A behavioural study in runtime analysis environments and drive-by download attacks

A BEHAVIOURAL STUDY IN RUN-TIME ANALYSIS ENVIRONMENTS AND DRIVE-BY-DOWNLOAD ATTACKS

MOHAMMAD ALLY REHAZ PUTTAROO

A thesis submitted in partial fulfilment of the requirements of The University of West London for the degree of Doctor of Philosophy

in

Cyber Security and Operating Systems

July 2017

Acknowledgements

The process of undertaking this PhD has provided a life changing process of learning and development for me. It would not have been possible to undertake the PhD without the guidance, support and experience of many people.

Firstly, I would like to thank my supervisor and mentor, Professor Peter Komisarczuk. Peter was exactly what every student who is crazy enough to start a PhD needed. With his invaluable expertise within this subject, there was never a time where Peter did not thoroughly support and provide me with direction. This direction was greatly appreciated during the difficult periods during my PhD. I am truly lucky to have had met and been able to work under the guidance of such an exemplary gentleman and Professor. To put into perspective, he kept me as his PhD student years after having left the original institution and never failed to uphold but the highest standards of supervisory and support. Thank you, Peter, there are no words to describe your kindness and dedication for what has been one hell of a journey.

My parents have been a constant source of strength, motivation and support during my studies: from mild encouragement to listening to me worry about my progress throughout the journey. They were there when experiments failed, and environments broke, providing me with a constant stream of positivity and perseverance. Without them, it would have proved impossible to successfully complete this thesis. Thank you for being supportive and providing me the means to take part of this (insane) journey. I would like to thank my lovely sister Madina for all the joys she brings to me along with Mazmoon and Munira.

My beloved wife Imrana deserves an award for the endless emotional support and for the countless hours proof-reading my thesis and learning about honeypots just to accommodate my endless monologues, debates, doubts and reviews.

I would also like to extend my sincerest gratitude towards the supervisory team: Dr. Renato Amorim who has one of the sharpest eyes when it comes to reviewing transcripts whilst also being an amazing co-author who has read through this thesis countlessly and helped tremendously in the betterment. Professor Thomas Roth-Berghofer has also been a constant source of feedback and review. These were key individuals in the refinement of the studies.

The feedback from Anastasia, Christian and Junaid at the internal school board was very helpful and finding thesis reviewers that are willing to go through an entire thesis is a rare commodity. I would like to thank them for the time and effort put into the betterment of the work.

I extend my thanks towards the University of West London for the scholarship and sponsorship that it provided over the years and the team at the graduate school. This is extended to the IT department for accommodating my needs and experiments. It must have been terrifying for an IT department to be told that malicious website analysis needed to take place on a network and without the accommodation required, this research would not have been possible.

The Honeynet project has been an indirect contributor for this work by inspiring motivation for research focus in this field. In particular, I would like to thank the Polish CERT (CERT Polska) for their various security projects, which led to this research.

For my parents, who have firmly believed in education from the very first few years of my life and everyone who have supported me get through the struggles PhD journey and grow as a person.

Abstract

In the information age, the growth in availability of both technology and exploit kits have continuously contributed in a large volume of websites being compromised or set up with malicious intent. The issue of drive-by-download attacks formulate a high percentage (77%) of the known attacks against client systems. These attacks originate from malicious web-servers or compromised web-servers and attack client systems by pushing malware upon interaction. Within the detection and intelligence gathering area of research, high-interaction honeypot approaches have been a long-standing and well-established technology. These are however not without challenges: analysing the entirety of the world wide web using these approaches is unviable due to time and resource intensiveness. Furthermore, the volume of data that is generated as a result of a run-time analysis of the interaction between website and an analysis environment is huge, varied and not well understood. The volume of malicious servers in addition to the large datasets created as a result of run-time analysis are contributing factors in the difficulty of analysing and verifying actual malicious behaviour. The work in this thesis attempts to overcome the difficulties in the analysis process of log files to optimise malicious and anomaly behaviour detection.

The main contribution of this work is focused on reducing the volume of data generated from run-time analysis to reduce the impact of noise within behavioural log file datasets. This thesis proposes an alternate approach that uses an expert lead approach to filtering benign behaviour from potentially malicious and unknown behaviour. Expert lead filtering is designed in a risk-averse method that takes into account known benign and expected behaviours before filtering the log file. Moreover, the approach relies upon behavioural investigation as well as potential for

system compromisation before filtering out behaviour within dynamic analysis log files. Consequently, this results in a significantly lower volume of data that can be analysed in greater detail. The proposed filtering approach has been implemented and tested in real-world context using a prudent experimental framework. An average of 96.96% reduction in log file size has been achieved which is transferable to behaviour analysis environments.

The other contributions of this work include the understanding of observable operating system interactions. Within the study of behaviour analysis environments, it was concluded that run-time analysis environments are sensitive to application and operating system versions. Understanding key changes in operating systems behaviours within Windows is an unexplored area of research yet Windows is currently one of the most popular client operating system. As part of understanding system behaviours for the creation of behavioural filters, this study undertakes a number of experiments to identify the key behaviour differences between operating systems. The results show that there are significant changes in core processes and interactions which can be taken into account in the development of filters for updated systems.

Finally, from the analysis of 110,000 potentially malicious websites, typical attacks are explored. These attacks actively exploited the honeypot and offer knowledge on a section of the active web-based attacks faced in the world wide web. Trends and attack vectors are identified and evaluated.

Contents

List of tables and figures

CHAPTER 1

Introduction

Chapter 1 – Introduction

1.1 Motivation

The internet that people have grown to love, depend on for livelihood, used as entertainment and carry out transactions has grown uncontrollably with the number of websites currently at over 1.1 billion (Internet Live Stats, 2017). Within the cyber security landscape, the realisation that web-based threats are one of the major attack vectors that exploit client systems which often are seen as the easier target to compromise motivates this research. It is an unsettling thought that within the world wide web, malicious websites exist to exploit information systems for monetary gains, at someone's expense. The cybercrime landscape is no longer but a few malicious software (malware) writers in a closed community sharing malicious script: nowadays organised cybercrime is on the rise – and a significant amount of this, comes from web-based threats. 77.26% of detected attacks were caused by malicious websites (Kaspersky, 2016). Merely visiting a malicious website or a compromised website that was exploited can cause a client system to be compromised and effectively 'owned'. Clearly within this area much more research is required to identify the methods and threats facing the user and client systems.

Traditional defence mechanisms such as anti-viruses and intrusion prevention systems are ineffective at coping with the volume of web-based malware. Within the detection of web based threats, the volume of the available websites on the internet is far greater than the analysts can cope with. Furthermore, the volume of data obtained from analysing the behaviour of websites to detect signs of maliciousness is huge. The work in this thesis capitalises on this volume of websites and the current limitations of analysing such vast amounts of data generated from run-time analysis. Optimisation techniques for understanding and analysing log files are a key

requirement as the world dives further in the cyber age. Furthermore, the design of behaviour analysis environments which are key to analysing both drive-by-download attacks and sandboxed malware is not currently sufficiently investigated. Knowledge of system behaviours from performing a behaviour study where behaviours are labelled and investigated further helps the decision-making process of understanding what happened to a system as a result of a captured behaviour.

Much like using a powerful magnet to gather needles from a haystack, expert driven behaviour filters that filter out investigated benign behaviours and keep malicious facilitates the analysis process. These experts driven behavioural filters provide higher confidence level when relying on filtering to reduce datasets that must be designed in an open and risk-averse way. It is hoped that the work presented in this thesis provide help to further analysis parties. One such party may include application of machine learning within the analysis of run-time log files that make use of the known sets of labelled behaviours to train their classifiers. Research in this field is highly active, especially within the application of machine learning in the detection of malicious behaviours. Within these works, the absence of annotated datasets has resulted in dependency on statistics and creation of training datasets. These data sets are unable to take into account the exploit potential on specific behaviours. The work in this thesis allows analysts to understand complex behavioural interactions and provide expert-lead labels for behavioural interactions in a behaviour analysis environment. These annotated datasets can be used to validate machine learning based clustering and classification of malicious webpages reliant on the dynamic analysis technique. Alternatively, the work presented could help future work based on Windows behaviour studies in both the creation of filters and the decision-making process at the design level of behaviour analysis environments

as Windows is one of the most popular operating system (OS) for end users (W3C, 2014 & Net Market Share, 2016).

1.2 Research goals

The goals of this research are to facilitate the issues faced by dynamic analysis in the detection of malicious webpages. The specific goals of this research include:

- Optimisation of the output from run-time analysis honeypots: One of the main limitations faced by an analyst using dynamic analysis is the volume and variation within behavioural data that is generated and stored upon execution. By optimising the output so only anomalies and unexpected behaviours are analysed, indicators of compromise or maliciousness can be identified sooner.
- Detecting malicious webpages and understanding active malicious behaviours. Investigating malicious behaviours that are captured provides insight regarding the attack vectors which can be useful in the identification of remediation or prevention of client attacks.
- Understanding behaviour analysis environment interactions: Classification of known behaviours and behaviour types into benign, malicious or grey allows the creation of labelled data sets. These can be particularly useful in the design aspect of a behaviour analysis environment as operating system, application and browser version choices can affect the output.
- Application of correct, safe, transferrable and prudent framework to real-world cyber security experiments: In order for this research to reflect the real-world there is a need for the experiments to be correct, transferable and therefore a suitable experimental framework for design and implementation of cyber experiments is required.

1.3 Contributions

1.3.1 Methodology in developing the expert driven behavioural filters

The main contribution of this work is the methodology proposed in the creation of the expert driven behavioural filters. These are filters that make use of the behavioural expertise of the behaviour analyst in reducing the noise and known benign behaviour in a log file. This is an alternative approach in the analysis of behavioural log files created during the run-time analysis of a potentially malicious web-age in a dynamic honeypot. This methodology's core aspect includes: firstly, the stage where the behaviour analysis environment is defined and designed to simulate a chosen user type. This user type is defined as a honeypot actively attempting to be compromised and monitor attack vectors but can be also designed to simulate a type of client machine within a business network. Secondly, an exploration phase takes place where a behaviour analysis environment would be executed, and behaviours would be captured using a list of thoroughly (with high levels of conviction) assessed benign websites. Thirdly, the observed behaviours undergo assessment, and this determines the potential of a given behaviour ability to perform malicious interactions. The scientifically rigorous approach in the development of behavioural filters is justified as a false negative error would effectively mean potentially malicious behaviours being filtered out thus, avoiding detection. The methodology is applicable to other versions and alternative behaviour analysis environment setups. The approach is tested in the analysis of potentially malicious websites and it is found that the expert driven behaviour filter approach filters out 96.96% of known behaviours which are benign. The approach showed that the application of the methodology in the real world resulted in the reduction of significant amounts of noise within dynamic analysis log files from an instance of Capture Behaviour

Analysis Tool (Capture-BAT) experimental setup. This is beneficial to the optimisation of the log file analysis process that is carried out by malware analysts as the sheer volume of events that can be recorded in log files as well as the number of false positives is drastically reduced.

In addition to the concept of expert driven behaviour filtering, the thesis contributes to knowledge by providing sets of labelled behaviours which were created from the analysis of real-world potentially malicious webpages. These behaviours are classified as benign behaviour, grey behaviour and malicious behaviours. The assessment and classification of behaviours were based on assessing behavioural interactions in terms of the potential risks to compromise a system and the frequency with which this behaviour is observed. This contribution has the potential to help future research that relies on Windows 7 malware that rely on dynamic analysis that filtering out noise or identifying key system behaviours that display inherent maliciousness. This artefact of the thesis can be applied directly to Windows 7 based operating systems when dynamic analysis of potentially malicious websites or samples is undertaken as the core operating system interactions are similar.

1.3.2 Identification of differences in behavioural manifestations in different versions operating systems.

As part of studying the behaviour of operating systems over the course of four years, evidence of behavioural drift was identified. Behavioural drift describes the changes in behavioural interactions that change over time in a behaviour analysis environment. This is an important concept that should not be overlooked when designing behaviour analysis systems. The justification here is that, over time alternate versions of known behaviours can be observed which should not be falsely classified as malicious behaviour. This discovery prompted the evaluation of different

versions of an operating system to identify the differences in observable behaviour and identify key behavioural difference. It was concluded that as a result of undertaking this investigation, there were several significantly different behavioural changes in patched and unpatched behavioural analysis environments. This suggests that careful consideration should be applied to the choice of operating systems for honeypots attempting to replicate client systems.

1.3.3 Observed active exploits and malicious behaviours attacking honeypots

A large volume of potentially malicious websites were analysed within the thesis' experiments and data capture. From this a range of confirmed malicious log files were gathered. The malicious behaviours within log files that were observed during drive-by-download analysis were explored and evaluated. This provided knowledge in terms of popular attack vectors and exploits faced during the course of the study. The synthesised knowledge reinforces existing knowledge on a large numbers of web based attacks that target various aspects of a vulnerable operating system, browser and plugins. This contribution is important as it is concluded that evidently a large number of exploits including browser injection and malicious file writes stay active long after they have initially been detected. This finding shows the importance of the application of security patches and browser safeguard methods within client systems post outbreaks of threats within cyber security as threats remain active.

1.4 Thesis organisation

This section discusses the structure of the thesis. This work is organised in 7 chapters. Initially, chapters 1 and 2 present the introduction, background and context of the research area. Approaches to detecting drive-by-downloads and malware analysis are discussed, relevant parties involved within detection and intelligence gathering is explored. Literature within this field is reviewed and gaps are identified.

From this, the research questions are synthesised on a variety of gaps from creation of behavioural exclusion lists to investigating changes within operating system setups.

Chapter 3 explores the experimental methodologies applicable to identified research questions. Within the chapter, malware experimentation framework with prudent experimental guidelines is adapted and applied to meet the research demands. The chapter evaluates and discusses sample sizes, resource acquisition and gathering required data. As part of chapter 3, two experiments are carried out which are presented in Appendix B. A validation experiment of the experimental setup and analysis environment is carried out and finally Twitter is assessed as an avenue to gather potentially malicious Universal Resource Locators (URLs). It is concluded that while there are malicious URLs observed within tweets, the volume of detected maliciousness was relatively too low to merely apply dynamic filtering directly thus, unsuitable for this study.

Chapter 4 presents the major findings in this study, and initially provides justification of the creation of behavioural exclusion lists within the Windows 7 behaviour analysis environment. Technical software behavioural manifestations are explored and labelled malicious, benign or grey. Discovered behaviours are compared with the predecessor operating system, Windows XP which results in knowledge of the key different processes and behaviours. Behavioural filter goals, proposed creation methodology and update strategies are proposed and implemented. The chapter concludes with a discussion on limitations faced in the development of exclusion lists.

Chapter 5 consists of two major experimental write-ups: Firstly, the created exclusion lists in the previous chapter is evaluated in the context of a real-world experiment within potentially malicious websites. It is concluded that the exclusion lists were able to filter a significant amount (96.96%) of known, benign behaviours. Secondly an experiment is designed and created that was tasked in understanding how the behaviour manifestations in a behaviour analysis environment change over in patched and unpatched environments. There were significant changes in observable behaviours and these key differential behaviour manifestations are explored.

Chapter 6 presents behavioural knowledge, pattern and common attack vectors that were identified in 5,132 log files. These were created in the interaction between client and 110,000 potentially malicious web servers by utilising Capture-BAT. The evidence of behavioural drift is introduced and discussed.

Chapter 7 evaluates and concludes the work undertaken. Limitations and delimitations are discussed in light of the approach. Potential future research directions are provided.

1.5 Publications

These papers were published at the time of the study.

1. Puttaroo, M, Komisarczuk, P, Amorim, R., (2014). Challenges in developing Capture-HPC exclusion lists, In The 7th International Conference on Security of Information and Networks, (SINCONF), 9-11 September 2014, ACM.
2. Puttaroo, M., Komisarczuk, P., Amorim, R., (2013). On Drive-by-Download Attacks and Malware Classification. Fifth International Conference on Internet Technologies & Applications (ITA), Wrexham, Wales, 10 to 13 September 2013.

CHAPTER 2

Literature review

Chapter 2 – Literature review

This chapter provides background and details regarding the research area. The detection of drive-by-download attacks and malware analysis are both highly active areas of research as seen by Amorim & Komisarczuk, (2012b), Bringer et al, (2012), and Egele et al. (2012). The technical approaches, definitions and techniques therefore needed identification and critique. Relevant detection and intelligence gathering parties involved and their contributions are identified. Drive-by-download area is then differentiated within the overall scope of malware analysis. Related work is critiqued leading to the identification of gaps within knowledge to be identified. Finally, research questions are formulated and scope of work is defined.

2.1 Introduction to drive-by-download and research area

2.1.1 Drive-by-download attacks

A drive-by-download is an attack where a web browser or another web enabled application is hijacked by malicious content, which is typically delivered by a malicious website. Drive-by-download attacks exploit the operating system, the browser or vulnerabilities in the versions of installed plugins. Other applications within a given client computer system are also vulnerable to attack. Creation of drive-by-downloads are fuelled by the motivation of black hats, also known as malware writers or hackers, targeting attacks on client systems as opposed to servers on the internet (Sherif et al., 2004). This is a highly desirable scenario as these attacks often provide complete control over the target, in turn resulting in monetary gains by underground economies from sensitive information ex-filtration, using a controlled host to launch attacks or provide dark web services (Cova et al. 2010; Provos et al. 2007, Canali et al., 2011). **Figures 2.0 and 2.1** outline drive-by-download attacks.

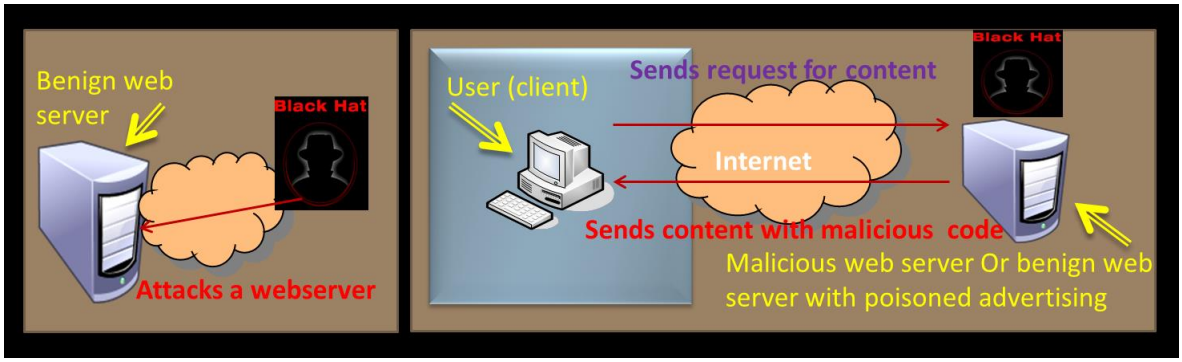


Figure 2.0: The drive-by-download problem and client side attacks illustrated based on knowledge provided by Dell'Aera & Seifert (2012).

Initially in the basic drive-by-download attack, a malware developer creates malicious content that attacks a vulnerable client; the malware is deployed by a malicious entity, either a developer or a third party. This malicious content is then served to a client system when the client system interacts with a web server. This triggers delivery of malware, which is typically hosted on a network of servers. The drive-by-download attack simply seeks to compromise a client system that visits a web server, which may have been compromised itself or may be delivering malicious content for example through advertising streams. This malicious delivery occurs without the client's knowledge and user's consent. The malicious code is downloaded and usually installed automatically. Even a single request from a client system visiting an infected website would result in vulnerabilities being targeted and malware being deployed on the system.

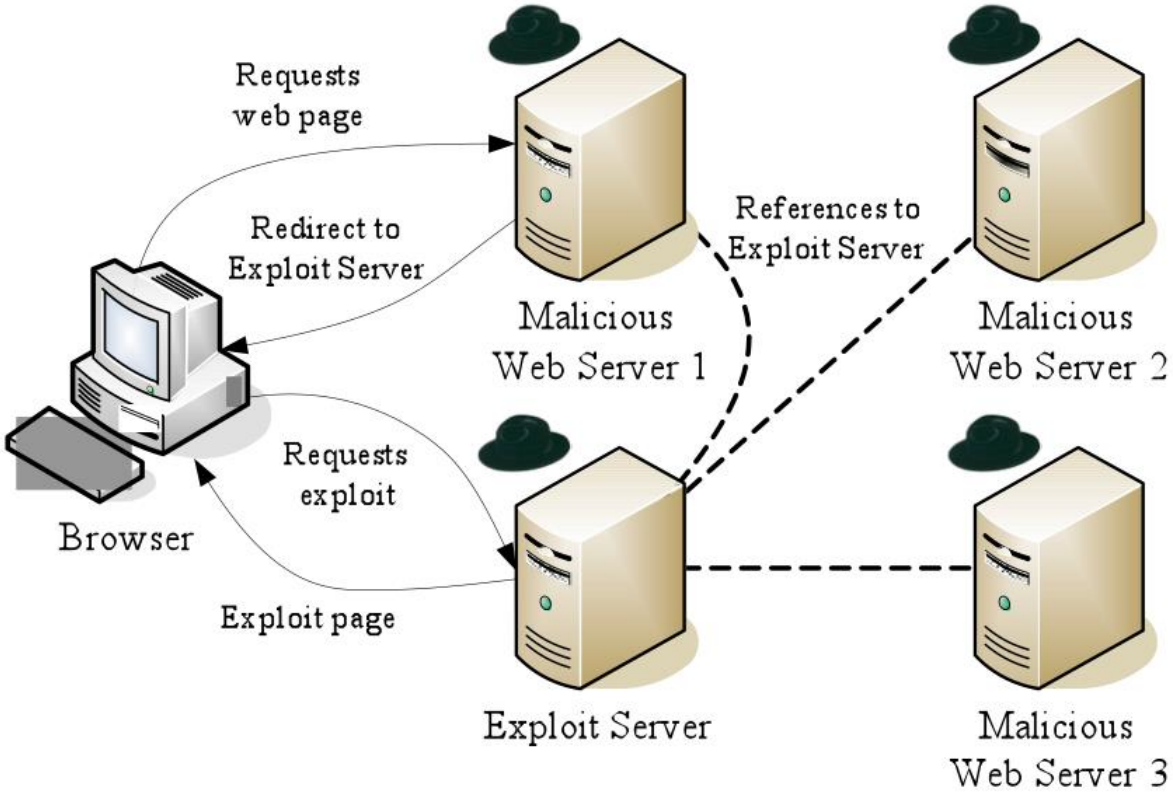


Figure 2.1: Drive-by-download attacking client systems from poisoned advertisement streams (Dell'Aera, 2012).

Drive-by-download attacks are not limited to malicious websites. Within the internet for example, benign websites that contain advertisement streams can contain malware from external sources. As illustrated in Figure 2.1, redirected requests from a seemingly benign website to the exploit and malware propagation servers can also be used to infect client systems. This type of drive-by-download can have amplified effects upon larger volumes of client systems as mainstream websites are often trusted by large numbers of users. Whilst exploring the exploitation process of a web server is not the purpose of this thesis, it is important to note that the exploit is often dependent on the web server security levels, ability for user contributed content to be shared, inclusion of advertising streams and ability for third-party widgets (Provos et al., 2007). More specifically to a cyber security perspective, legitimate web sites can

be compromised by a range of methods: Structured Query Language (SQL) injection attacks, search engine result redirections, cross-site scripting attacks or simply vulnerabilities in a web-server hosting software. Le et al. (2012) suggests that increased drive-by-download attacks can also be due to the increased availability of exploit packs which require the attacker to have significantly less skill set to deploy. Typical drive-by-download attack vectors are explored further in Figure 2.2. The exploit seeks to target a given vulnerability in software. The exploit delivery mechanism is the observable drive-by-download behaviour that is concerned with dropping malicious code or files onto a client machine. Obfuscation is used to hide exploits, delivery mechanisms or the insertion of JavaScript within code. This presents a challenge to detection methods that are not based on run-time analysis.

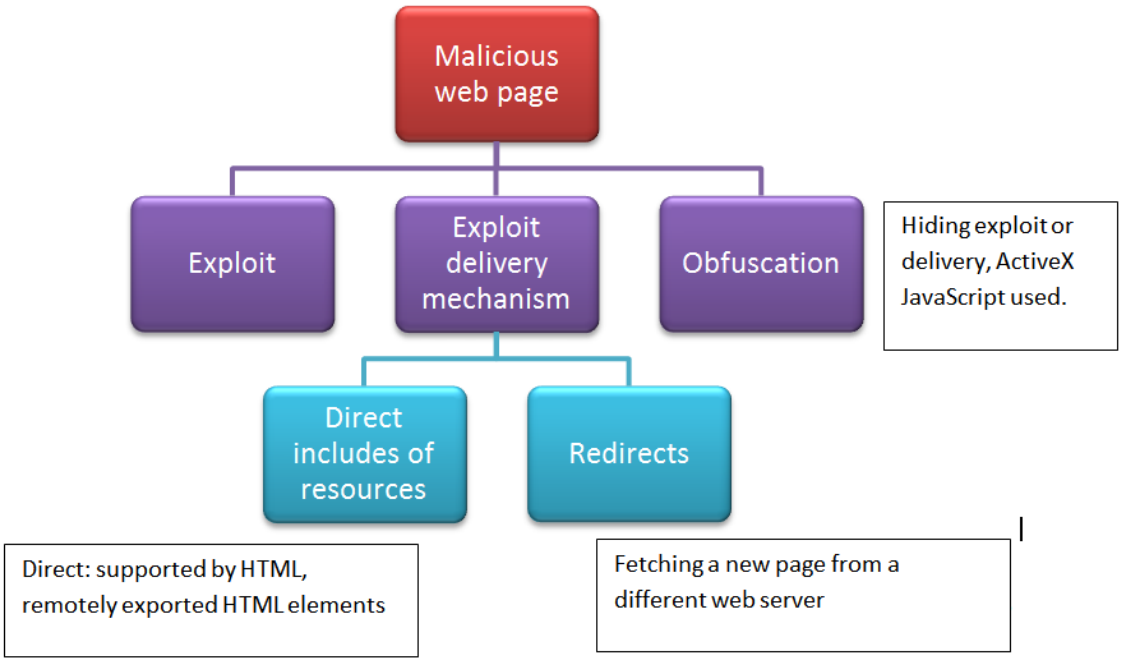


Figure 2.2 Malicious webpage attack vectors.

Compromising a client system can be highly damaging to the victim (Clemenson, 2009; Cova et al., 2010; Provos et al., 2007). This is logical because a black hat would be able to access all of that client's personal and confidential data as well as

any data stored on the network, which a client is given privilege to access. Some examples include credit card details, corporate files, login and password information, cookies, temporary files and saved passwords which are for example stored in a browser. Extortion through ransomware is also a growing trend as discussed by both O'Gorman & McDonald (2012) and the FBI (2016). In order to compromise clients, malicious websites attempt to exploit vulnerabilities from the web browser (such as Internet Explorer), plug-ins (such as ActiveX and Adobe reader) or even the operating system's built in features (such as exploiting privilege available in Command (Cmd) in Windows). Additionally, compromised client machines can be used to launch Distributed Denial of Service (DDoS) attacks and send spam messages around the internet. Khattab et al. (2004) presents work on mitigating denial of service attacks. These availability based attacks were predicted to be on the rise by Rivera (2013).

It is important to present a high overview of the anatomy of a malware attack as this provides insight into how the thesis' drive-by-download research fits into the overall picture. Le et al. (2013) presents the anatomy of a drive-by-download attack in stages and uses this framework to detect potential drive-by-download attacks. Figure 2.3 shows the timeline between a malware being created to the compromised state that is faced by client systems. Initially a malware writer would create a malware sample or download existing malicious scripts widely available on the world wide web and black hat communities. These are then used to compromise an existing web server or a new server is set up for the purpose of compromising clients. As client machines or users visit the web server and send Hypertext Mark-up Language (HTML) requests, the malicious content (either present on the web server itself or redirected from exploit servers) performs vulnerability exploitation aimed at

the application, the client operating system and plugins. This exploitation allows the drive-by-download to install malware. Malware on a client system may lead to the aforementioned damaging effects such as data exfiltration, malicious control over the computer system and using the compromised system to launch attacks or gain resources (Egele et al., 2009).

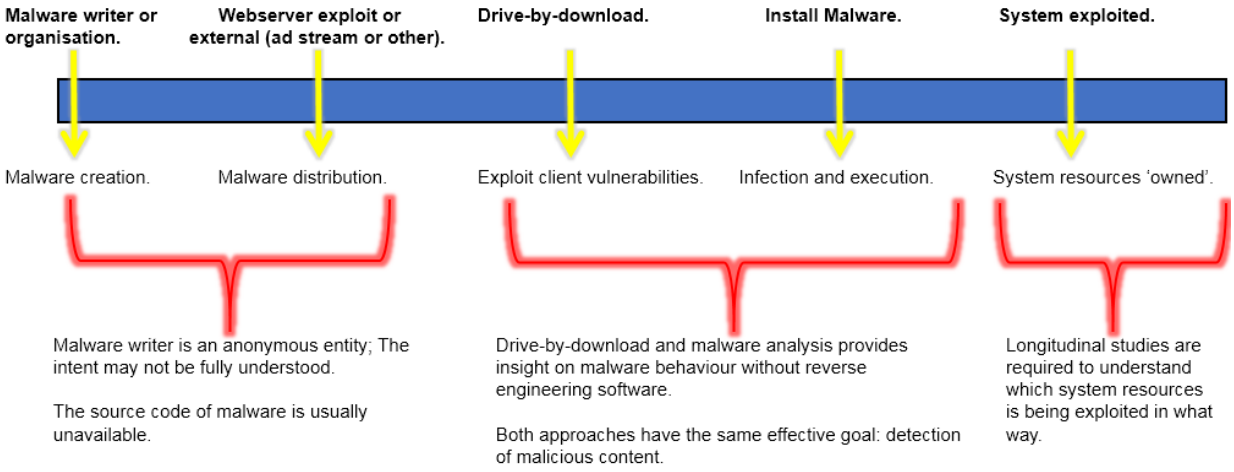


Figure 2.3: Anatomy of the creation and exploitation process.

As the internet is a vast public network which is growing at a significant level, it is difficult to reliably estimate the actual number of malicious websites that are currently active at a given point in time. Internet Live Stats (2017) estimates that there are 1.1 billion active websites. The sheer processing power required to analyse the entirety of the internet is simply far greater than that is currently available by any individual organisation and therefore sections of the internet are surveyed by different research groups. Seifert (2010) estimated that in 2010 there were around 150 million malicious websites. Newer statistics (Av-test, 2015) show that in the last 5 years the number of detected malware is on the rise: in 2015, there were around five times more detected malware in comparison to 2011. More specifically when considering web based malware and malicious web servers; Kaspersky (2012) reported that in 2012 the number of browser-based attacks were nearly 1.6 billion and that 73.70%

of all attacks originated from malicious URLs. Malicious URLs as a percentage of all attacks observed by Kaspersky was shown to have increased to 75.76% in 2015 (Kaspersky, 2015) and again to 77.26% in their 2016 report (Garnaeva et al., 2016). Symantec (2016) in their internet security threat report suggest over 'one million web attacks against people each and every day.' These statistics therefore illustrate the necessary requirement of reliable security mechanisms and continued research within detection and intelligence gathering to protect client and networks systems from the ever-growing sea of web-based threats.

2.1.2 Detection, intelligence gathering parties and technologies

In addition to the research discussed above, there are several different parties involved in the detection of malicious webpages and the gathering of intelligence within the area of drive-by-download research. In this thesis, these are divided between the government, commercial services, and research communities.

Computer Emergency Response Teams (CERTs) are dedicated expert groups that handle computer security threats. CERT Polska (Poland) handles network security threats for the .pl web space and performs research into the detection of security incidents and analysis of malware. They have a joint venture with the National Cyber Security Centre (Netherlands) in the deployment of the HoneySpider network. This is a highly scalable system that uses multiple instruments including client honeypots and is aimed at the large-scale detection of malicious URLs. Their monitoring and early warning systems are tasked with improving awareness within the world wide web and focus primarily on 'attacks against or involving the use of web browsers' (HoneySpider, 2013).

In the United Kingdom (UK), Nominet provides a commercial ‘Domain Health’ service to help all UK-based domains that are managed by the Nominet service (Nominet, 2017). This service focuses on collecting security information from compromised web servers under the .UK domain and categorises servers into several abuse categories; abuse, command and control, compromised, associated in malware or botnet attacks, phishing and spam.

The Shadowserver Foundation (2017) is involved in the capture, analysis, monitoring and reporting of malware and botnet activity within the world wide web. Shadowserver uses the Nepenthes honeypot primarily to collect malware samples.

The Honeynet project is a non-profit organisation that has been active within detection of malware and intelligence gathering for several years (Honeynet, 2014) and is a collection of nationally managed chapters. Members of the Honeynet project actively engage in a large number of research and development projects within malicious webpage discovery, and raise awareness of existing web-based threats, conduct data analysis approaches and develop of security tools.

Anti-virus software vendors are also highly active within this area of research and regularly publish statistics reflecting their findings using their data gathering networks (Garnaeva et al. 2016; Kaspersky 2015; Symantec 2016; Akamai 2016). These often include a portion of data dedicated to online (web based, drive-by-download) attacks. For instance, Garnaeva et al. (2016) provides a list of Common Vulnerabilities and Exposures (CVE), country attack distribution charts, and observed exploit vectors (such as the browser, Adobe reader, Flash player, java, …etc).

Historically, web based services such as Wepawet by Cova et al. (2010) have been made available to perform analysis of both URLs and malicious file format. Similarly,

Anubis (http://anubis.iseclab.org) performed URL analysis and provided a report based on operating system and browser interactions and behaviours that takes place when a given URL is submitted for analysis. The lists of behaviours provided by these online analysers displayed all system interaction that took place upon analysis of a website and did not differentiate malicious from benign behaviours.

Virus Total (2016) is an online information aggregator service that uses anti-virus engines and website scanners to detect malicious content on files or malicious websites. This is a useful service that facilitates the identification of malware, and possible false positives. The ability to use multiple antivirus scanners and website scanners which are updated in real time facilitates detection. Virus Total also hosts a collection of malware hashes, malware binaries as well as malware analysis data from several anti-virus vendors such as Sophos, McAfee, Symantec and a lot more popular anti-virus. As an information aggregator, Virus Total provides a free un-biased service on a given malware sample. This makes Virus Total ideal for research without any influences from any particular parties. Within this research, Virus Total is used extensively in two ways: first, Virus Total is used to verify collected malware samples. These malicious binaries are submitted to the multitude of anti-virus vendors supported by Virus Total and assigned signatures are requested. Second, Virus Total is used as a second step validation to ensure that a given website does not show signs of maliciousness when the use of benign websites are required. Some examples of the capability of Virus Total are provided in Appendix A.

Client honeypots (or honeyclients) are security devices, which sole purpose are to be compromised and probed by malicious web servers. The intent behind this is to identify potentially malicious websites from those that are benign. These devices, unlike traditional defensive security systems such as Intrusion Detection Systems

(IDS), firewalls, anti-viruses or data encryption and backups are a more aggressive approach to client system defence. Client honeypots are at the forefront of intelligence gathering to initiatively protect against threats to the client network. Studies can then be conducted to understand the operations and intent of a malware writer by capturing malicious activity and analysing attack patterns. Additionally, this helps in identifying vulnerabilities in client systems (Seifert, Welch & Komisarczuk, 2006). From reviewing the available literature and intelligence gathering parties, it is fair to conclude that the issue of drive-by-downloads is an active area of research within the context of cyber security.

2.1.3 Collecting drive-by-download samples

A drive-by-download can be viewed as an unintended artefact that is downloaded onto a computer after visiting a webpage. The unintended aspect of this definition usually applies to the unsuspecting client browsing a web site. This attack usually takes place without the consent and knowledge of the client; therefore, it is typically classified as malicious. The behavioural interaction between a client and a malicious web server can often be exploited to deliver malware by a drive-by-download. It is important to note that client honeypots are needed to collect drive-by-downloads. It is possible to crawl the World Wide Web to visit suspected malicious websites to attempt to collect malware samples as well as obtain behavioural information on the interaction on the client's guest operating system and the applications running whilst the malware is being delivered. Recent research on the long-term studies of malicious URLs by Tanaka & Goto (2016) shows reliable evidence that some malicious URLs are found to survive for more than 500 days after first being detected. Long-lived malicious URLs that are either crafted by a malware writer or are parasitized from unpatched website vulnerabilities may deliver a particular exploit

which still contains the same exploit that they were originally designed or injected with. This clearly has implications within cyber security research as long-lived malicious URLs, which target a specific vulnerability, are likely to have an impact on a created dataset for analysis. Newly collected datasets face the possibility of collecting slightly outdated exploits that were aimed at older operating system, application and browser vulnerabilities. In addition to this, research using the internet to build data corpuses faces a large volume of available webpages which can mean that the collection of drive-by-download samples result in data that is out-dated. Long-lived malware is not new within cyber security as samples such as research by Nazario (2006) showed samples seen in 2002 were still being distributed in 2006. More recently, Goodin (2016) discusses the Project Sauron malware which has been active for 5 years, proving the longevity of active malware. Therefore, when drive-by-downloads were being collected from the internet it is evident that there is a requirement for some sort of control to collect relevant malware within the context of the target client system utilised. Collection of malware samples can be filtered when building a data corpus based on malware that actively attack a type or version of O.S. This allowed the data corpus to exclude older websites that target older client systems although older client systems are still prevalent.

2.2 Methods of analysing drive-by-downloads

Drive-by-download analysis in the context of this work refers to the behavioural interaction that a client-honeypot would experience and record upon the visitation of a malicious website. This is different to a traditional malware sandbox which looks at environmental changes and system calls when a malicious Portable Executable is executed. Drive-by-download analysis is the analysis of environment changes whilst

a drive-by-download is attempting to exploit a system and does not assume that a given binary file has been downloaded or saved on the analysis environment. Typically, this involves using a client honeypot to simulate the real-world scenario of client accessing a malicious webpage scenario and logging system behaviours for drive-by-download analysis. This will provide information on attack vectors used by malware: these can either be exploited weaknesses and vulnerabilities from the web-browser or web-browser plugins, as well as any applications being run. Typically, in the system life-cycle of a malware attack, drive-by-downloads would occur before a malicious binary is executed: behaviour of drive-by-downloads can be studied to understand how a given malware sample infiltrated a client environment. Thus, when referring to drive-by-download behaviours or drive-by-download analysis, the web exploit's non-standard and malicious behaviours observed on a live system is what this thesis is referring to.

Drive-by-download analysis is typically carried out using three types of approaches; low-interaction client honeypots, high-interaction client honeypot and hybrid client honeypots. These approaches are core to the thesis and therefore are discussed before a rational for choosing high-interaction is provided.

2.2.1 Low-interaction

Low-interaction client honeypots perform drive-by-download analysis without actually executing the malicious websites on live systems. Simulations of interactions are used to interact with a malicious website. Response from these web servers are then analysed to detect static presence of malicious content. An example of static content could be a set of strings contained within a HTML response. These lightweight solutions are effective at scanning large volumes of potentially malicious websites and are less likely to spread malware infections through a network due to the lack of

exploitable interaction with malicious websites. An example of widely used low-interaction client honeypot is Thug by Dell'Aera (2012). Thug has a number of different browser versions and plugins. These are used in the analysis of web-based threats by emulating behavioural events. The analysis relies on attack signatures and the identification of interesting breakpoints. Yet Another Low-Interaction Honeyclient (YALIH) by Mansoori et al. (2014) is another example of a currently developed low-interaction client honeypot with capabilities to emulate virtual browsers and cookie redirection. However, low-interaction systems usually tend to be ineffective at detecting new and unknown malicious behaviours as they cannot provide full emulations of client systems and use simplistic rule-based or signature checking to detect attacks. Signature based systems face evasiveness from malware writers who use obfuscation and polymorphism techniques in creation of new malware samples. Additionally, there is therefore a requirement for an attack to be known prior to website analysis, which in the world of cyber security is challenging due to the nature of the cat and mouse game between malware writers and security professionals (Riden & Seifert, 2010). Additionally, low-interaction client honeypots would face the same limitations that static detection systems face. Such an example include: Obfuscated attacks which modify patterns, code, data and behaviours in avoiding detection. Both limitations discussed can be mitigated by incorporating dynamic analysis from high-interaction systems in purely native low-interaction systems.

2.2.2 High-interaction

High-interaction client honeypots perform runtime (dynamic) analysis of drive-by-download interactions by running instances of guest O.S. that actively crawls the web with the goal of getting compromised by visiting malicious websites. These high-

interaction systems can be comparable to real-world clients and can additionally be set up to simulate a certain type of user. Recorded attacks against such a client can then be analysed by studying malicious web-exploit manifestations from real-world malicious web servers in real-time. This behavioural analysis provides the benefit of identifying malware based on their behaviours as opposed to signatures, which can sometimes lead to the detection of 0-day attacks from malicious drive-by-downloads. It is imperative to state that high-interaction honeypots can provide a much deeper level of detail upon how an exploit compromises a client system in comparison to a low-interaction counterpart.

High-interaction client honeypots have been around for a number of years some examples include: Capture-HPC by Seifert et al. (2007) and Shelia by Rocaspana (2009). The main drawback in terms of malware analysis of high-interaction client honeypot is the time and resource intensive nature of dynamic analysis. Additionally, due to the nature of running real-world malware tests on live systems there is the risk of the analysis client getting compromised and exploited to launch attacks against other networks or being exploited as a gateway into the network infrastructure. The benefits from obtaining behavioural information from real-world malicious websites outweigh the drawbacks faced as both aforementioned limitations can be mitigated. For instance, known domain names could be targeted to obtain a higher detection rate and the honeypot could be isolated from the main network. This can then run within a separate network connection and good security practices could be applied as well as limiting the time that a compromised client honeypot is allowed to be active. This is an ideal choice for honeypot type for real-world research as it is not dependant on the limitations faced by low-interaction counterparts. Design limitations such as the usage of virtual machines to analyse maliciousness can be an issue

sometimes as these can be detected by malware which would likely not manifest full behaviour. High-interaction client honeypots are nevertheless, a good approach to capturing malicious behaviours. In line with this statement, Lau and Svajcer (2010) showed that only as little as 2% of malware sampled could detect the presence of virtual machines. This is a relatively low percentage but is likely to have increased at the time of writing.

Typically, the output of a high-interaction client honeypot is recorded as a behavioural log file. These behavioural log files contain behavioural information that can be used to understand how different components of an analysis environment interact. It is in this way that malicious behaviour can be identified: the unexpected interactions can be investigated to understand how malware is downloaded on to the system and what plugins or environment features are targeted as part of malicious attacks. This is the focus of this thesis.

2.2.3 Hybrid-interaction

Hybrid client honeypots combine low and high interaction approaches usually in various stages to attempt to perform large scale studies of drive-by-downloads attacks. Low-interaction features such as interpreting java and web crawling is undertaken and from this the potentially malicious domains are sent for further analysis within the high-interaction component which would undergo dynamic analysis. Examples of a hybrid honeypot systems include Honeybird or Honey Spider 2. Hybrid honeypot systems are highly effective when there is a requirement for crawling the world wide web and analysing large volumes of unknown URLs. In this context, an estimated volume would be in the hundreds of thousands, possibly millions of URLs per day as new websites are discovered and analysed by numerous intelligence gathering parties aforementioned. The rationale behind this claim lies

within the sortation process where initially the low-interaction component would analyse every webpage in the list. The webpages that provided unknown responses as well as potentially known malicious responses would then be forwarded to the slower high-interaction counterpart.

The HoneySpider network 2 (2013) is highly scalable system of integrated honeypots for detecting malicious content on the web. Within the previous architecture of HoneySpider 1 integrated high and low-interaction client honeypots which were used in addition to Snort's Intrusion detection system. However, the new HoneySpider 2 includes a more complex detection framework using a number of exploit detection mechanisms such as JavaScript analyser, PDF analyser, and Flash analyser in addition to using Capture-HPC and anti-virus scanners. This integrated framework offers the possibility of a number of different exploits to be detected from a wide range of plugins, which users rely on daily. HoneySpider 2 is not simply a honeypot as it also provides a number of services. An example of this is the storing of data and analysing of honeypot data by using a number of nuggets. Therefore, HoneySpider 2 could be described as a management platform that crawls the web in search of malicious webpages. Koo et al. (2013) also used hybrid honeypots in the detection of malicious webpages.

Work related to this thesis by Seifert et al. (2008) performed hybrid analysis of webpage data in by combining static with dynamic analysis. A number of exploits were discovered which could be detected by looking at different static attributes of webpage data and provided a classification method based on assessment of attributes on an initial HTTP response. The decision trees technique was effectively used at the time by the author to determine whether a number of features, such as, an iFrame of very small size was either malicious or benign. Whilst this method of

analysing websites and determining whether a website is malicious or not is very efficient at scanning large numbers of webpages, an obvious limitation would be the number of false positives obtained. In terms of identification of malicious webpages, the authors admit that the method misses a number of attacks which is concerning, but the work was focused on attempting to deal with the ever-increasing large volumes of websites active on the world wide web.

In addition to this, the methods used to identify malicious webpages are unlikely to be applicable in the current web era as behaviours and features change: the proposed decision trees would face a fairly high number of false positives which would miss a significant number of drive-by-download exploits. The decision tree proposed would require a number of updates to keep up with new exploits as the low-interaction aspect would likely require new signature updates. Song et al. (2010) agrees and mentions that malicious webpages are short-lived before they are changed or removed by the malicious black hat. As the classification model is out of date and no longer applicable to the current World Wide Web it is fair to conclude that new classification models are required to identify malicious web servers.

2.2.4 Chosen approach and rationale

Low-interaction client honeypots did not provide real-world observable behaviours and rely on signatures beforehand. This meant that it would be unsuitable to apply low-interaction client honeypots to this research. The justification behind this is purely because the work undertaken in this thesis was focused on assessing behaviour manifestation within behaviour analysis environments and observable real-world malicious web exploits. High-interaction client honeypots or run-time analysis on the other hand provided the means to observe system calls of a behaviour analysis environment whilst also not being limited by the dependency of

an exploit being known beforehand. The limitations faced by low-interaction client honeypots and the interest within the observable web attacks within client systems drove this research into applying high interaction client honeypots within the study. This research used potentially malicious websites as the main resource in capturing malicious behaviour. This approach is different to Le et al. (2011) where the URLs used were unknown and required a low-interaction analyser to create a list of potentially malicious webpages. It is therefore fair to conclude that the usage of hybrid systems would be unnecessary: the URLs utilised as input for the honeypot have been pre-filtered thereby rendering the initial low-interaction analysis redundant. In addition to this, the number of potentially malicious websites retrieved from multiple black list and malware domain lists utilised in the study were not as numerous to justify the usage of hybrid client honeypots.

2.3 Malware

Malicious computer software with the intent to disrupt operation, deceive, extort, steal sensitive information, gain access to private systems, spy or advertise is known as malware. Malicious software has been on the rise and forefront of cybercrime with a vast variety of types including viruses, trojan Horses, rootkits, backdoors, spyware, worms and ransomware. Examples include: ransomware trojan – CryptoLocker or Computer worm – Stuxnet. This section outlines some key definitions and work within malware analysis with the conclusion focused on differentiating the traditional malware analysis stage with drive-by-download analysis state in the anatomy of a malware attack.

2.3.1 Malware analysis

Malware analysis is the process of studying computer software which was created with intent to harm a host O.S. or steal data by violating confidentiality, integrity and

availability of the computer system. Malware analysis involves dissecting a number of different components of malicious software with the intent to study the possible malicious behaviour manifestation of software within a host O.S. Malware analysis techniques include two main approaches which are sometimes combined to provide better understanding of malicious intent. These approaches are static malware analysis and dynamic malware analysis.

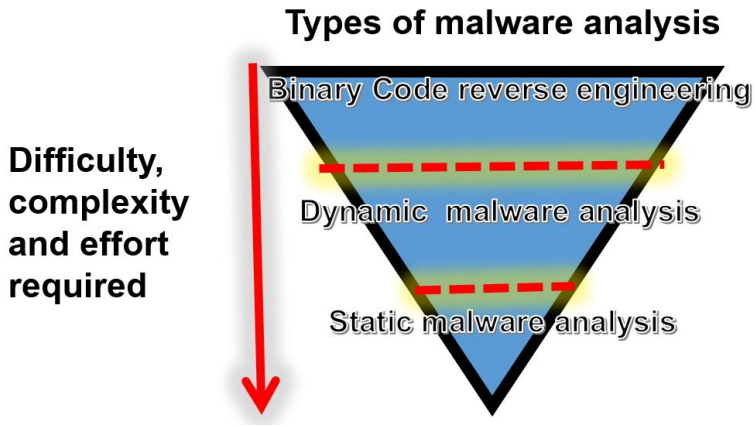


Figure 2.4: Overview of malware analysis approaches including requirements based on Zeltser (2016).

Malware analysis requires more effort and significant experience by the malware reverser as fine grain malware analysis is a very complicated task. This picture can be used to also represent the amount of time required for each task to conclude: static malware analysis would require the least amount of time for completion and reverse engineering would require the most. The approaches of static and dynamic malware analysis are shared within drive-by-download analysis, however different parts of the anatomy of an attack would need to be analysed. Code reversing for drive-by-download analysis would require the web server page code and any additional scripts attached to it.

Static malware analysis involves studying a given file, specifically Portable Executable (PE) files, in Windows and understanding the file's functionality. This sometimes can indicate the maliciousness of a PE file without having to execute the file, bringing the benefit of fast malware analysis as well as requiring less hardware to analyse malware samples. Typical tasks in static analysis include searching the file for strings, checking if the file is obfuscated or packed and learning the content of file headers.

In dynamic malware analysis, the malware analyst looks at a more in-depth set of properties of the malware through execution. Typically, an isolated environment is created where the file would be executed and behaviours recorded. Examples of these behaviours include: system calls, process created/deleted, files created, modified, deleted, registry entries and network traffic. These behavioural entries are crucial for a malware analyst to understand what a given program is attempting to do when executed. With this information, it is possible to classify a program analysed as benign, malicious or unknown depending on the behaviour. It is important to note that while dynamic malware analysis is a lot more labour intensive and time consuming than static analysis and requires the setup of a functioning malware analysis lab or sandbox; it is proven to be much more accurate in detecting maliciousness within programs than static malware analysis due to its thorough nature Zeltser (2016). An example solution of endpoint malware detection using dynamic analysis and AI to detect malware is Cylance (https://www.cylance.com).

In order for malware to execute, the environment would require a degree of transparency that would prevent malware from detecting and stopping potential attacks. Additionally, the O.S. and applications installed on the isolated environment

would need to somewhat mimic popular real client systems on which the malware was intended to attack.

Reverse engineering program code is often done manually by a malware analyst. This can be thought of as dissecting a malware sample to obtain valuable insight: data stored on the program, algorithms and logic used within the program and revelation of all functions and possible behaviours which are not always exhibited. Disassemblers, debuggers and decompilers are used in this process. This task is highly time and labour intensive, so much so that it is currently unable to cope with even a fraction the amount of malware released every day. Additionally, the malware analyst is required to have a very rare skill set which is why the vast majority of malware studies do not use this method Zeltser (2016).

2.3.2 Research around malware analysis

Malware analysis has been carried out by a large number of authors covering vast research scopes. Although the thesis is aimed at drive-by-download analysis, some of these issues and techniques translate over from the dynamic aspect of malware analysis and can thus be very applicable to the work presented. This section outlines some high impact work within malware analysis. Willems et al. (2007) presented CWSandbox, which is an analysis environment that monitors system calls and created a report. They show that it is possible to automate binary analysis of current Win32 based Portable Executables for malware analysis using CWSandbox. However, this does not seem available as it seems to have evolved into a commercial solution. Detection of obfuscated malware is covered by Dinaburg et al. (2008), who address the issue of malware detecting the presence of analysers in the analysis component by applying hardware virtualisation technologies and extensions

that reside outside the analysis operating system environment that would otherwise be detected by some malware samples.

Moser et al. (2007) explores single program execution issues that are observed with dynamic analysis tools. The effect of this prompts malware to only be triggered based on a number of specific or special circumstances: e.g. a specific day, with the inclusion of a specific file being present in the environment, or when a certain command is obtained from the malware writer. The authors propose a system that allows the exploration of multiple execution paths and identifies malicious actions that are only executed when certain conditions are met. The paper uses a number of different values that try to get the executable to execute and display malicious behaviour patterns. Their experiment shows that for a significant fraction of the malware sample used, multiple execution paths were being applied.

Recent research by Vasilescu et al. (2014) utilised sandbox system, Cuckoo (Cuckoo, 2014) to run malicious executable in dedicated environments in order to capture malicious behaviours. These environments are emulated via the use of virtualisation software much like Capture-HPC clients. However, they do not only analyse the initial interaction between client and server when accessing a webpage, as they assume instead that the malicious executable sample is present in the sandbox and that the system was already compromised. The authors also carried out manual analysis of malware in a Windows XP environment with the use of WireShark DumpIt, Volatility and IDA. It is important to note that the authors used Windows XP in both the Cuckoo sand box instance as well as the manual malware analysis carried out. It is likely that some new exploits aimed at targeting Windows 7 systems, which is currently the most popular operating system, are not able to attack older Windows based environments which would mean a number of malware not

being flagged up as false positives as the environment is unaffected by these specific Windows 7 attacks. The lack of focus in the most common operating system, namely Windows 7, are likely to have missed targeted attacks by only focusing on Windows 7 vulnerabilities specifically. Future research can therefore avoid this pitfall and detect more malicious behaviours by attempting to replicate common client applications and by usage of the most popular operating system. An extended discussion on the operating system's applicability to research is undertaken later in the chapter.

2.3.3 Malware packers

A packer is a wrapper around software to encrypt and compress the contents of the software for efficient delivery and deployment. Packers can be used legitimately and without malicious intentions to minimise upload and download times when transferring files as well as protecting copyright code. Unfortunately, packers are routinely used in malware propagation to disguise malicious files or code and therefore evade detection from anti-malware scanners (Virus Bulletin, 2015). Runtime packers perform unpacking operations such as decrypting and decompressions on executable files. This is done in stages involving unwrapping the wrapped file, loading to memory and executed. From a malware perspective, malicious software could effectively be wrapped several times using different 'wrappers' or packing methods and a malicious file itself could be changed in small insignificant ways. Consequently, the final packed file would appear as a new or undiscovered malware sample. This is a cheap, easy and highly effective way for malware writers to pack a previously detected or known sample and avoid detection from signature based detection utilised by anti-virus vendors. This method of

encrypting contents of a malicious file for detection avoidance is known as Malware code obfuscation.

Packers which pack Microsoft Windows PE files support the executable format (.exe) and the Dynamic Link Libraries (.DLL) and can be written in a number of different programming languages (such as C++, Python, Visual Basic, JavaScript…etc.). It is therefore fair to state that the malware writers have a vast array of tools to pack and obfuscate their malicious scripts and original PE files should a version of a given malware be detected by an Anti-Virus vendor's signature based approach. It is evident that in order for a malware sample to be inspected and analysed by a malware analyser, the sample needs to be stripped of packer layer(s) so that the original executables can be analysed. Malware writers utilise anti-unpacking techniques in order to avoid packed malware from being detected and Reverse Engineered (RE) (Wei et al., 2008);

- Cyclic redundancy check (CRC) of a given packed file is used to check file patching.
- Redundant and jargon code is inserted between actual malicious code in scripts to bypass static detection methods and decompilers.
- Using trigger based approaches to prevent unpacking upon dynamic debugging detection.
- Self-changing code (mutation) on the original executable to prevent detection by memory dumping based approaches.

In the thesis, packets are not captured which makes identification of the initial packer used to deliver the malware difficult to identify. However, some behaviours of packers can be observed within the log files that Capture-BAT gathers. For instance,

it is possible to observe malicious Portable Executable files being delivered as the malware installs onto the client within the Capture-BAT log files. Examples of real and active samples are explored further in Chapter 6. Furthermore, captured drive-by-download samples were analysed through Virus Total which provides packer information about known packers that the saved Portable Executables used.

2.3.4 Differentiating drive-by-download and malware analysis

Having discussed both malware analysis and drive-by-download analysis, it is important to reflect upon the anatomy of a system exploitation. Figure 2.3 summarised the typical attack lifecycle which was in line with discussions by Le et al. (2011) and Cova et al. (2010). Differentiating malware analysis with drive-by-download analysis is therefore possible when using the anatomy of an attack. Figuratively, it is possible to observe that drive-by-download attacks typically take place before malware exploits: usually in terms of the attack pattern, a client system would first be compromised by an exploit server which would then deliver the drive-by-download. This drive-by-download can then perform multiple vulnerability exploitations. A dynamic example of this may include hijacking the browser process into writing malicious files or the addition of malicious parameters in the registry controlling auto start files. In terms of drive-by-download analysis, the analysis aspect would be to observe what interactions and system calls are triggered upon visitation of a malicious webpage. On the other hand, malware analysis typically uses a potentially malicious file which is executed within a sandboxed environment. Often this malicious file is presumed to be already present in the analysis environment and is thus executed with the differential behaviours observed. The system calls triggered are then monitored in behaviour analysis to determine if the given file is performing malicious actions on a client system.

2.5 Instruments for drive-by-download collection and analysis

Within high-interaction analysis, a behavioural analysis environment is a created operating system and set of applications in the environment that are used in the analysis of drive-by-downloads or malware samples. These environments can then be designed to replicate similar client systems that an organisation would have with the exception of real data being substituted with falsified data. The replication of client systems allows the researcher to understand the weaknesses and vulnerabilities of similar client system by actively allowing the behaviour analysis lab to be compromised. Observation of behavioural manifestation provides key information on attack vectors and triggered system calls. It is therefore imperative that behavioural analysis environments are either designed with the goals of replicating the client system which they were designed to originally protect or designed and implemented in such a way that captures as many malicious attacks as possible. This is achieved by omitting newer patches of software that would exclude some existing vulnerabilities from the platform. The concept of environment design is of utmost importance within behaviour analysis systems which rely upon applications, plugins and specific operating system vulnerabilities that malware exploit. Only observed manifestations of malicious and unexpected behaviour classify a given malware or website, malicious or unknown. This was found to be the rationale behind a lack of malicious detection in the particular type of analysis environment used in Cova et al. (2010).

2.5.1 Capture-HPC

Capture-HPC was created by Christian Seifert and was established in 2007. Capture-HPC is a high interaction client honeypot which performs dynamic analysis on malicious web servers to capture malicious behaviour in log file format and also

captures modified and created files. Within the thesis, as Capture-HPC forms an important aspect of this work, a thorough discussion is provided in chapters 3 and 4. In terms of real-world website analysis, analysis using Capture-HPC would be dependent on the website performing an attack against Capture's Behaviour Analysis Tool (BAT) client in order to flag unexpected behaviour or malicious behaviour upon a web server. Capture is not dependent on knowing attacks and malware signatures beforehand, making this an ideal tool for detecting real-world, web-based malicious attacks.

A key critical factor within detectability of malware by Capture-HPC is however dependent on the installation of plugins and software that would be used by malware in the behaviour analysis environment (known as Capture-BAT) as supported by Cova et al. (2010). This makes comparing Capture-HPC's detection rate a challenging task when compared to different systems which do not face this limitation, for instance a low-interaction counterpart dependent on signatures. In this thesis, in order to mitigate the impacts of this limitation within Capture-BAT, a number of applications, particularly those versions that were vulnerable to malware attacks, were installed in our behaviour analysis environment. Details of which can be viewed in the experimental setup in Chapter 4 (section 4.5.3.)

Capture log files can be configured to capture behavioural interaction that takes place upon booting and loading of a website or to simply capture malicious, new and unknown behavioural interactions. The former configuration results in a large volume of redundant data which includes benign system calls that are observed at every boot up. In comparison, the latter configuration results in a more focused and filtered set of data which is of significantly less volume. The main drawbacks of the second approach are the requirement of a created and maintained tailored list of exclusions,

known as an exclusion list, that are implemented based on a given environment's Operating System (O.S.), applications and plugins. Exclusion lists are explored in-depth within Chapter 4, along with creation goals, typical native system behaviours, proposed development lifecycle and implementation.

As outlined by Seifert's (2010) thesis, there are substantial numbers of malicious webpages. Nowadays, only analysing half of the amount of webpages that was studied in the author's research would be required to detect a number of threats, including zero-day threats. The clear limitation observed is the sheer amount of processing power required to scan these malicious webpages. Another possible limitation would be the current nature of malicious web servers which are short lived (Bringer et al. 2012).

2.5.2 Cuckoo sandbox

Cuckoo sandbox has been available since 2012 and performs dynamic analysis of malware. Cuckoo also has the ability to perform dynamic analysis of malicious websites, resulting in a log file output similar to Capture-HPC. An instance of Cuckoo sandbox is available on the web (malwr.com), offered by Guarnieri et al. (2012). Much like Capture-BAT, Cuckoo has the ability to monitor environment behaviour patterns but has been constantly re-developed to include a vast number of features. These include taking screenshots of malicious file executing, performing memory analysis of an infected system or dumping and analysing encrypted network traffic. Similar to Capture-HPC, it was identified that there was no filtration mechanism in place that was publicly available in the dynamic analysis reports obtained by malwr.com.

2.6 Related work

This section introduces some alternative work that is related to the work carried out in this thesis. In this section, gaps within the literature are identified and reviewed.

2.6.1 Malicious Domain name servers

Research by Seifert et al. (2008b) looked at a number of malicious web servers in New Zealand by observing underlying server relationships and the numbers of Domain Name Server (DNS) were counted. If a malicious web server had more than two domain name extensions and more than five unique DNS servers, it was considered malicious. This method was simple but unfortunately only tested in New Zealand in 2008. The main limitation to the current day and age is simply that the New Zealand study was limited to a country that had low levels of malicious activity. This is proven by the well-established anti-virus vendor, Kaspersky. In a threat report by Namestnikov (2011) New Zealand is revealed to be the 14th country in the world with the lowest percentage of infected computers. Evidently, research is required in countries such as the US and the Russian Federation that were marked as “high risk” due to having the highest number of attacks. More importantly, it is fair to state that the Internet provides global access to web users which are not limited to a particular country. This means client honeypot studies that seek to find current web threats should ideally contain a variety of websites to identify the current client exploits that are being targeted.

2.6.2 Systems monitoring state change

There are several proposed approaches applied in the detection of web based malware. State change tracking systems such as Capture-BAT is proposed by Seifert & Steenson (2009), Cuckoo sandbox by Guarnieri et al. (2012), HoneyMonkey by Microsoft Research (2007), Kim et al. (2011) and CWsandbox by

Willems et al. (2007). These state monitoring approaches simply track system behaviours such as interactions between the file, process, registry and (sometimes) network. These behaviours can be assessed and anomalies within expected behaviour can be identified. Behavioural based changes that these systems monitor include: files created or deleted by malware, registry key modifications, creations and deletions, process creations, injections and terminations. These dynamic analysis systems are not perfect and thus do not report the way that a malware attack originating from the web or from a malicious executable is programmed but instead the observable behaviour manifestation within a system which is a viable approach in combating malicious software when researchers are unable to combat malware using 'methods of disassembly or reverse engineering' Willems et al. (2007). State change systems Capture-BAT, Cuckoo sandbox and CWsandbox all output a behaviour log file of the system interactions that were captured during the analysis process. The benefit of this approach is that from the detailed behaviours that a log file displays, it is possible to identify attack vectors, infected files and processes as well as changes to the environment that was compromised. These systems were introduced in section 2.2.2 and are discussed in greater detail within Chapter 4 as state monitoring systems are used within the thesis's major contributions. Well cited work by Provos et al. (2008) is concerned in identifying drive-by-downloads at a large scale from crawling the world wide web. Their focus was much like the work present in this thesis, where the behaviour of installed software (malicious Portable Executable dropped) is not investigated but the mechanisms used to introduce maliciousness in the system (drive-by-download analysis) are investigated.

Combining fast analysis approaches with slower, run-time approaches in attempts to obtain the benefit from both has been covered by previous work as seen in section

2.2.3. Le et al. (2011) proposes a two-stage classifier model for detecting malicious webpages. This model comprises of analysing static features. These describe the website content without having fully executed the supposedly malicious website, and then the potentially malicious websites are analysed further. Run-time features then provide the second aspect of classification between benign or malicious. Their hybrid system functions by combining both these features: the benefits of low resource requirements from static detection is utilised to identify potentially malicious webpages. If a website is found to be potentially malicious is then passed on to the slow, resource intensive run-time feature. This is a good approach in limiting the enormous amounts of URLs that would be dynamically analysed but if malicious attacks evade the static component of the two-stage classifier it is likely that they would evade detection all together. An alternative research approach to gathering malicious drive-by-download might be to make use of known malicious domains instead of relying on the static aspect of the hybrid system and instead use the dynamic analyser directly on a large sample of known malicious websites. One gap identified within literature reviewed showed little focus on work that is focused on identifying the behavioural impact and differences that is resulted from using different versions of environments in state-monitoring research. The design decisions behind what operating system is chosen for a honeypot are often left to the defence teams. It is reasonable to choose similar environments to live systems (such as same base operating system). However, when identifying differences between behaviour analysis environments with varying versions of the same operating system such as a patched operating system against the same base operating system unpatched, remained an area to be explored and was thus a gap within research.

2.6.3 JavaScript research

Web based malware research has also been approached from the JavaScript aspect: It is possible for malicious code to be concealed in JavaScript and avoid detection. Egele et al. (2009) studied memory based malicious code which is often found to be saved for later execution. These heap-spraying methods are common in the exploitation of client systems by targeting web browsers. The authors propose techniques to identify shell code based drive-by-downloads in a browser and implement this solution in the Mozilla Firefox browser. This run-time approach proved to be very successful at detecting JavaScript based malware when assessed as no false positives were identified.

Johns (2008) explores JavaScript malware and proposes relevant protection and approaches for research in JavaScript. Curtsinger et al. (2011) presents ZOZZLE which provides a rapid solution to detecting malware from analysing malicious JavaScript attacks. The analysis of webpage carried out by ZOZZLE is mostly static detection, making it possible for run-time (dynamic) attacks to avoid detection. However, the evaluation of ZOZZLE shows an extremely low false positive rate of 0.0003% and high performance due to its lightweight nature. Likarish et al. (2009) used machine learning classifiers which detect malicious JavaScript with a deliberate set of features to detect malicious JavaScript. The difficulties in dealing with JavaScript code are still an issue in 2017. Mogren (2017) reviews machine learning approaches in the detection of malicious JavaScript code and identifies some issues in the currently available JavaScript research such as the lack of availability in datasets used by machine learning.

2.6.4 Current trends in client honeypot research

Within this field of study, research is highly active and diverse. Akiyama et al. (2017a) presents HoneyCirculator which is based on the use of bait credentials leaked by malware and malicious web content. This research used a honeypot to successfully evade detection by malware and thus, was able to monitor malicious infrastructures and activities. Due to this decoy system being difficult for malware to detect, Akiyama et al. (2017a) was able to collect niche data sets which weren't part of public blacklists. This suggests that a number of blacklist sites are detectable by malware and thus, future research could look at using alternative ways in finding URL datasets or use systems that are not detected such as HoneyCirculator. Akiyama et al. (2017b) performs studies in the ecosystem of malicious URL redirections. These re-directions attacks have been active for a long time and can still be observed frequently. The findings of this work show further evidence that click-fraud is the main motivation behind malware writers using URL re-direction to compromise clients. This is yet further evidence that drive-by-download attacks remains as one of the principle attack vectors used to compromise client systems.

Mansoori (2017) presents a thesis on the geo-localisation of attacks targeting the browser. Within the work presented, the focus was on characterising the nature of attacks in terms of economical and legal reasons for an attacker to target a given group of users. Social factors are taken into account and a number of economic conditions of a country may lead to various types of attacks being used. This study is able to answer a number of research questions on the types of attacks faced by specific geographical locations which brings insight into geo-location trends within drive-by-download research. The issues of Internet Protocol (IP) tracking and cloaking is explored by Mansoori et al. (2017). The authors apply the HAZOP

methodology in designing experiments aimed at assessing the extent of aforementioned issues within their dataset. Mansoori et al. (2017) concludes with the fact that IP tracking is used in only a small subset of domains within their datasets and that there is no strong indication of network tracking is observed. This therefore shows evidence of a low level of tracking within malware domains present on the world wide web.

Work within honeypot research has also been optimised to enhance deception capability of honeypots using network service fingerprinting. This is a recurring issue in both the general cyber security landscape and intrusion detection: detection mechanisms require systems and methods which does not reveal their intentions when attempting to analyse potentially compromised systems. Dahbul et al. (2017) proposes threat modelling to identify potential threats that reveal the existence of honeypots. The author discusses various countermeasures which are tested and proven effective in the enhancement of deception capabilities of honeypots.

2.6.5 Applicability of machine learning

As seen in work focusing on malicious content detection within JavaScript above, machine learning is often applied within the cyber security research field. The application of machine learning is often done in both drive-by-download and malware analysis to aid in the detection of malicious content and these are often used by the detection and intelligence gathering parties discussed in section 2.1.2. Machine learning's applicability to the facilitation of anomaly detection is a vast field of knowledge and previous work. It is important to note that the focus of the thesis is purely based on the filtration of log files stage for dynamic behaviour analysis. Within the applicability of machine learning to the detection and intelligence gathering spectrum, there is a limitation; this is the dependency on statistics to make decisions

on benign and maliciousness. It is hoped that the results of this work is used to help machine learning with the decision making process of unique client behaviours as decisions are made with context to specific behaviours and their ability to undertake malicious activities. It is nevertheless important to have a discussion on the next steps of analysis within drive-by-download studies to provide perspective and comparison to the proposed approach as some studies perform varying levels of data transformation prior to analysis within cybercrime.

2.6.6 Clustering approaches

Behavioural clustering of malware is an extensively researched topic (Bailey et al., 2007; Bayer et al., 2009; Perdisci et al., 2010). These works use a number of different clustering algorithms; however, it's important to note that these are mostly based on hierarchical clustering. The hierarchical clustering method builds a hierarchy of clusters to represent data using either a top-down or bottom-up approach. A top-down approach, sometimes called Divisive clustering simply works by splitting a cluster in two parts starting from the cluster containing all entities. A bottom-up approach, known as Agglomerative clustering simply merges two of the nearest clusters at each step of clustering (Rokach and Maimon, 2005; Mirkin, 2011).

In terms of malware research, the application of hierarchical clustering is useful as inter-relationships can be viewed between clusters: similar malicious behaviours from different clusters can be used to link clusters and similarities between malware families observed. The work by Bayer et al. (2009) and Perdisci et al. (2010) differ to the work undertaken in the thesis as they are aimed at the malware analysis stage of the anatomy of an attack and assume that the utilised malicious executable has already been dropped into the analysis environment whereas in this thesis the area

of focus is one step before, on the drive-by-download stage. The main drawback of hierarchical clustering is to do with the performance issues: Amorim & Komisarczuk (2014) suggests that these hierarchical algorithms operate at a complexity of at least $O(n^2)$. This is problematic as multiple sources (Seifert, 2010; Shadowserver 2013; Av-test, 2015) show that the quantity of malware released daily is increasingly growing. Furthermore, when applied to the behavioural malware data sets obtained from high-interaction client honeypots; the volumes of data in malicious log files are high. A gap within literature is identified as the issues faced with malicious detection research justify the need for faster performing algorithms within malware clustering and classification or alternative approaches to filtering datasets and reducing volumes of noise data. Either approach would allow a means to improve detection mechanisms in processing the enormous numbers of malicious websites.

2.6.7 Classification approaches

Rieck et al. (2008) proposes classification of malware based on behaviour. Malware behaviour is monitored in a sandboxed environment; the learning stage is based on labels of the anti-virus engine Avira while ranks are assigned to discriminative features of malware. The results of their experiment seemed very promising at the time and it would be interesting to replicate the experiment with a number of different anti-virus software instead of only using Avira, which had one of the best detection rates at the time. It is fair to state that in the paper, even though labels from Avira anti-virus provided imperfect labels, classification of malware on behaviour was successful almost 70% of the time when tested with 3,000 unique samples of malware. Furthermore, detecting malicious executables in the wild and performing classification using several algorithms such as n-grams, naïve Bayes, decision trees

and support vector machines to detect malicious Portable Executables from benign Portable Executables by Kolter & Maloof (2004).

The real time classification of malicious URLs shared on Twitter was addressed by Burnap et al. (2015). This work utilised Capture-HPC undertake dynamic analysis of potentially malicious URLs that were shared on Twitter using event-specific hash tags in posts that contained a URL. The approach utilised machine learning classifiers to classify URLs as benign or malicious in real-time, seconds after a link is clicked. Their deciding factors included network traffic, CPU usage and network connection statistics, which seemed to provide 66 - 72% efficiency in the worst and best information availability scenarios. An interesting finding in their work was that 'malicious activity is probably occurring within the first 60 seconds of the interaction' which is encouraging due to the copious amounts of potentially malicious website that is widely active and indicative markers such as this may be used in future research to identify known exploit vectors sooner. An investigation of this nature is covered in chapter 3. Burnap et al. (2015) concluded that capturing packets entering and leaving the network proved to be the key indicator of their predictive activity metric. This is in line with current knowledge as monitoring packets provides an alternate view of devices that could be affected within detection of malware or traces of anomalies. The evidence provided in their work shows that twitter is being used for malware propagation; the applicability to gathering malware domains on twitter for this work is also explored in chapter 3. It is clear that classification approaches for vastly different areas of research within cyber security can be varied and are therefore acknowledged.

2.6.8 Closely related work

Work that is more related to the work undertaken in this thesis, Amorim & Komisarczuk (2012) propose a method to cluster Capture-HPC behavioural log files of 17,000 malicious websites in less than 2 minutes. The authors have moreover upgraded Capture-HPC to address an issue: Capture-HPC had difficulties detecting malware contained in ActiveX components. Their clustering method is based on K-Means, which is one of the most popular Partitional clustering algorithm (Jain, 2010; Wagstaff et al, 2001; Chiang & Mirkin 2009). However, this is a non-deterministic algorithm and therefore may provide numerous different clusters should the algorithm be run more than once on the same data set. The authors avoided this issue by initialising K-Means with centroids generated by Mirkin's anomalous pattern method (Mirkin, 2011). K-means tends to be mainstream in the clustering approaches (Chiang & Mirkin 2009, Mirkin, 2011, Amorim & Komikarczuk 2012). Work dependant on unsupervised learning approaches such as Amorim & Komisarczuk (2011) and Bailey et al. (2007), face difficulties that are derived from their unsupervised nature: the lack of external information utilised to guide the analysis of data. Therefore, it is argued that the skills processed by a malware analyst or from the set of known behavioural entries in identifying benign from malicious behaviours created log file dataset is not utilised within these approaches. Whilst performing drive-by-download analysis within the context of the anatomy of an attack, Amorim & Komisarczuk (2012) did not discuss the contents of the utilised Capture's exclusion list or the methodology used to create these behavioural filters. The usage of engineered exclusion lists can be an alternative method to filtering noise in log files as opposed to applying AI feature reducing techniques. In some cases, log file used in malware studies have sought to apply machine learning and

feature-reducing techniques to separate goodware (benign software) and malware behaviour and work by Amorim & Komisarczuk (2012) have done just that prior to clustering their malicious website dataset. In some cases, a combination of these machine learning techniques are used to are used to analyse data (Belkin, 2004). However, feature reducing methods can sometimes prove to be problematic when some features are extracted such as the file path in behavioural log files as often these can offer key indication on behavioural maliciousness. A gap is identified here as it may be possible to include these key indications in an alternative approach that performs expert driven filtering from a list of known behavioural entries prior to performing further analysis. This is very different to the unsupervised approach and the feature weighing and selecting done in previous research as it would involve confirmation of each benign behavioural entry.

The problem within drive-by-download and malware analysis of distinguishing between detecting malicious and benign has been around for a while. Within malware analysis, Tian et al. (2010) performs behavioural analysis and differentiates between 456 goodware and 1368 malware samples. This work used dynamic features as opposed to previous Application Programming Interface (API) monitoring research that used static features from Portable Executables as seen by Ye. et al. (2010). Tian et al. (2010) execute binary files within a virtual machine environment with the behavioural manifestations is observed and a report is written. Features are extracted and their work provides distinguished goodware or malware classification with an accuracy of 97% at the time. This study however focuses purely on differentiating Portable Executable files and not behavioural log file events from drive-by-download interactions. Consequently, it is not focused on individual observable behaviours but on the entirety of the goodware behaviour observed by

executing the 456 samples. By focusing on the behaviours manifested in a behaviour analysis environment it may be possible to build knowledge tables of expected behaviour that visiting a benign website would trigger. Their work therefore does not seem to include the other components of an analysis environment such as the operating system or commonly installed application behaviour. Wagener et al. (2008) also carry out dynamic analysis but their experiment does not include benign or goodware which does not provide verification of expected, non-malicious system calls or the ability to create correct datasets. Within the context of dynamic analysis, an operating system running can often mean that large volumes of goodware behavioural entries as agreed by Burnap et al. (2015). If these are taken into account within analysis, It may be possible to alter the overall result. Furthermore, it is line with the prudent malware experimentation by Rossow et al. (2012) to remove goodware data from dataset. In both cases a list of known goodware and expected behaviour is not provided which makes replicability of experiments carried out difficult. An adaptive yet alternative approach to Tian et al. (2010) method of distinguishing between goodware and malware is simply to run large number of benign tests on the environment and understand the core behavioural calls that undergo while a system is active. From this an understanding of the benign system calls could be synthesised and creation of behaviour based filters could be used scalability in the search for unknown and malicious behaviours. This is the approach that the thesis intends to take in the reduction and expert driven filtering of log file based datasets.

2.7 Gap identification and research questions

Addressing some of the gaps identified in section 2.6 is the basis of the thesis. This section discusses relevant gaps and formulates research questions. The main

problems that are addressed include: behaviour filtering, the creation of effective behavioural filters using real-world intelligence and the impact from a behavioural perspective of using different operating systems in performing dynamic analysis of potentially malicious webpages.

2.7.1 Behaviour filtering in behaviour analysis environments

Behaviour filtering is the concept of filtering known benign behaviours, unknown behaviours and malicious behaviours. When an analysis environment performs dynamic analysis, a large amount of behaviours and system call interactions are created as a result of the system performing scheduled tasks, loading boot up scripts and process and file manipulations that are required for the operation system to load critical files and processes. During a malicious website or file analysis, these normal and benign system behaviours are also recorded in log files from dynamic analysis. It is evident that the effect of recording all behaviour increases the size and complexity of given log files and hides malicious behaviour. Performing behaviour filtering can result in datasets which contain less noise which benign behaviours add and countering the complexity issue. Filtering these behaviours also results in significantly smaller log files. An example is explored in Chapter 4, section 4.2.3 where the size and complexity of real-world benign log file is recorded post analysis and further rationale is provided for filtering behaviour. Work that does not apply or mention expert driven filtering of benign behaviour from malicious behaviour is likely to perform analysis with a large volume of benign noise behaviour and thus may result in different outcomes. This is a gap in published knowledge as seen in Wagener et al. (2008), Amorim & Komisarczuk (2012). Furthermore, research on the effectiveness of malware experimentation by Rossow et al. (2012) showed that

around half of paper surveyed did not separate goodware from malware datasets. The thesis fills this gap in knowledge by answering the research question:

- What are the key benign system behaviours observed in a Windows 7 operating system that can be filtered in a created behaviour analysis environment?

Understanding benign and normal system behaviours in behaviour analysis environments is a complex process. Often it is observed that operating systems do not always perform the exact same behavioural interactions at every boot despite the state of the operating system being kept constant. This was tested and showed variations even when the additional protection of configuring the hard drive to be in an immutable state. To overcome this limitation, it is often required to have multiple tests of an operating system using benign and goodware datasets to be able to explore and learn as much of the unknown and semi-frequently observed behavioural interactions and system calls. Within this thesis, the concept of behaviour filtering was applied through the exclusion list of the chosen behaviour analysis tool, Capture-BAT. In addition to this, within the utilised Windows 7 environment, typical observed behaviours are labelled as benign or grey (unknown). Typical observed drive-by-download behaviours are discussed and labelled in Chapter 6 from the analysis of 5,132 potentially malicious websites.

2.7.2 Expert driven filtering and exclusion lists

It was found that research within drive-by-download analysis often did not include expert driven filtering mechanisms. Expert driven filtering mechanisms are defined as filtering behaviour from dynamic analysis log file data that includes the expertise of the malware analyst and excluding known behavioural entries within the filtration

process. This method could serve an as alternative way to understand behavioural entries in a log file and filtering the known behaviour that is often created as a result of a behaviour analysis environment being booted. Therefore, the aims of this activity are identified as an alternative approach to optimising datasets within behaviour-based drive-by-download studies. These aims include:

- A methodology that would address the sheer volume of data and reduce large volumes of noise as this reduces the amount of processing required to understand key behavioural entries.
- A behaviour filtration system designed in a risk-averse way. This would seek to not compromise detection mechanisms by filtering out possible malicious behaviours.
- An expert driven filter that includes the expertise of actual behavioural entries from behaviour studies in the decision-making process.

In terms of implementation of these expert driven filters within this thesis, Capture-BAT's exclusion lists are used to build, test and assess the concept of behaviour filtering. An exclusion list is a list of known environment behavioural events that Capture-BAT will include or exclude in the generation of a Capture log file. Thus, exclusion lists can be viewed as the behavioural filters of Capture-BAT as they attempt to filter out expected and known behaviours. This is an alternative method to filtering out large amounts of noise within log files by separating goodware behaviour from malicious and unknown behaviours. At the time of writing, it is still observable that available exclusion lists are out of date. The original exclusion lists by the creator of Capture-BAT, Seifert & Steenson, (2006) were designed and released in 2006. The second set of exclusion lists that can be found on the web have not been updated for the last 9 years and designed on an older version of Windows XP by

Kindlund (2008). The utilised version of Capture, Capture-HPC_NG was also shipped with default exclusion lists that were based on Windows XP and the exclusion lists released by Seifert & Steenson (2006). Each set of exclusion list mentioned were tested to verify the output of Capture-BAT log files. This confirmed the out-of-date claim when tested upon Windows XP and 7 setups as every evaluation (which included a large number of confirmed benign URLs) resulted in a malicious classification by Capture-BAT.

Upon log file inspection, it was clear that Capture-BAT log files was successfully capturing system calls and interactions taking place upon boot however, within behavioural log file analysis this proved to be problematic: goodware behaviour was mixed with potentially malicious interaction. The inclusion of benign behaviours within log files increased the log files considerably as shown in Chapter 4. In terms of behavioural analysis, this was a major issue as exclusion lists within Capture-HPC are at the forefront of behaviour filtering. When Capture-BAT perform analysis of a given website, all behavioural interactions are recorded that are not present in an exclusion list. Within malware research this can prove to be problematic as aforementioned, large volumes of benign behaviours that would occur naturally from the booting and loading process of an operating system. These behaviours would be observed and picked up in Capture log files. Separating known goodware samples from malicious samples in datasets before analysis is a prudent experimentation recommendation supported by Rossow et al. (2012). When applied to drive-by-download behavioural analysis, it is fair to conclude false positives and noise data within log files could potentially lead to a larger margin of error if this is not labelled and removed prior to the analysis process using an approach that applies behavioural knowledge and critique before filtering.

As a result of the limited research focus towards creating exclusion list and adapting these lists to different operating systems, creation of exclusion lists was initially undertaken to determine process and issues. There are three parts to the research focus: The initial creation methodology, the good practice and requirements from a cyber security perspective and finally the evaluation of created exclusion lists. These form the core research questions:

- How can behavioural filters (exclusion lists) for a new operating system be created to reduce output of data in log files without losing relevant behaviours that aid in the detection of anomalies?
- What are the requirements and good practices needed in the development of behavioural exclusion lists?
- To what extent are the created exclusion lists effective at blocking (filtering out) benign system behaviours?

It is fair to state that the desirable outcome from using behavioural filters would be a high percentage of known benign behaviours being blocked whilst utmost caution is applied within the design and implementation stage to safeguard the filter from false negatives (filtering out malicious behaviour).

In the initial work, the challenges in developing Capture-HPC exclusion lists were explored and published, Puttaroo et al. (2014). These challenges included:

- The client specific nature of behaviours manifested within created analysis environments.
    - Different operating systems and different applications that run within the analysis environment display different manifestations thus,

exclusion lists should be customised for the behaviour analysis environment that they are filtering.

- The need to constantly amend and modify existing behavioural entries upon installation of new applications, patches or operating system versions.
    - As systems change new behaviours are often observed as proven in chapter 5.
- Each type of recorded behaviour would require an exclusion list
    - Within Capture-HPC exclusion lists are needed for processes, file system and registry interactions respectively.

A methodology for creating expert driven behaviour filters is proposed in section 4.5.2 of Chapter 4. This methodology is applied in the creation of our Windows 7 exclusion lists and then tested in chapter 5 in the contribution to knowledge.

2.7.3 Windows 7 applicability and operating system behaviours

In the initial process of creating exclusion lists it was identified that behavioural filters for the Windows 7 operating system were a new area of research and thus development within this particular operating system was focused. There was an aspect of pragmatism for usage of Windows 7 as operating system for behaviour analysis environments used in the thesis. This is justified as Windows 7 has been the most popular operating system and there is a lack of research focus within client honeypot research applied to Windows 7. As Windows 7 is the most popular operating system in use within the period of 2013 - 2016 (W3C, 2014 & Net Market Share, 2016) it should theoretically be one of the most popular environments for behaviour analysis environments used in drive-by-download and malware analysis as the majority of desktop client systems in use are deployed on Windows 7.

It is concluded that this is however not the case and the vast majority of research published focuses on Windows XP as their behaviour analysis environment Cova et al. (2010), Alosefer & Rana (2010), Amorim & Komisarczuk (2012), Seifert et al. (2008). This finding emphasises the importance of future research to utilise newer versions of operating systems or at least some more focus towards the most popular versions of operating systems. Furthermore, it is believed that this lack of focus on environment behaviour studies and usage of Windows 7 contributes to the lack of knowledge available in creating behaviour filters that can reduce the substantial amounts of redundant data present in behavioural log files. From the work on the challenges in developing Capture-HPC exclusion lists (Puttaroo et al., 2014) it was shown that changing environment variables would result in differently observed environment behaviours. This is logical and can be controlled by numerous factors such as the operating system used or the applications that are fired up during boot-up. However, this leads to another research question:

- What are the key differential behaviours that an updated Windows 7 displays when executed in comparison to a stock and an unpatched version?
- Capture-BAT was originally designed for Windows XP: how does Windows 7 benign system behaviours differ from Windows XP?

Both of these were identified as another aspect which has not seen research consideration. Answering these research questions also forms part of this thesis.

2.7.4 Research scope and summary

The work attempts to build on previous work within drive-by-download analysis and explores several key areas. The scope of the work is now summarised.

- Understanding typical benign behaviours and performing behaviour filtering from a drive-by-download analysis environment.
    - What are typical, expected and non-malicious background behaviours within an analysis environment that can be labelled benign behaviour?
- Applying behaviour filtering in real-world context: Creating, testing and evaluating Capture-BAT exclusion lists.
    - Methodologies in the creation of Capture-BAT exclusion lists
        - Development cycle of exclusion lists.
        - Methodology used for picking benign websites required
    - Assessing implemented exclusion lists in real-world context.
        - Efficiency of exclusion lists
- Assessing differences in operating system behaviour
    - Differences between behaviour analysis environment behaviours in patched and unpatched operating systems.
    - Observed differences between behaviour analysis environment behaviours in predecessor Windows XP and Windows 7.
- Analysing observed malicious web exploits attacking Windows 7 client honeypot.
    - Typical attack vectors, drive-by-download exploits and malicious behaviours observed from analysing malicious webpages.

For the purposes of facilitating understanding of the research scope, an overview of the drive-by-download analysis process is presented and the area of focus is highlighted (Figure 2.5 in red) .

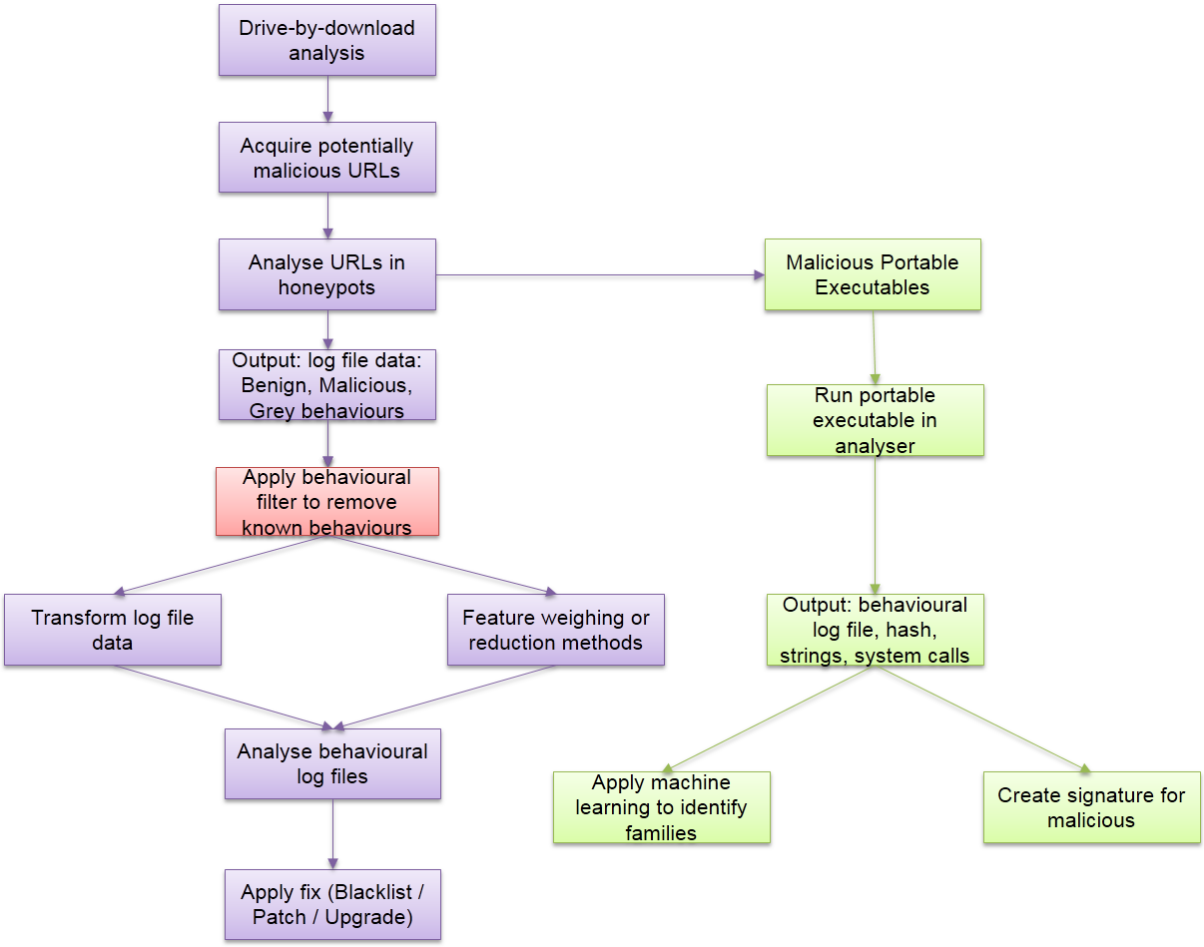


Figure 2.5: An overview of drive-by-download and malware analysis process detailing the behaviour filtering of log file data approach.

This research presents an alternative approach to deal with the problem often faced within cyber security research: processing large volumes of log file data which is required as the number of web servers available and the short-lived nature of malicious webpages are still increasing thus requiring effective detection mechanisms. Run-time analyses applied in the detection of malicious webpages are resilient to the limitations that static analysis face from code obfuscation and do not require the exploits to be known beforehand. There are two aspects to the limitations faced by run-time analysis: yielded behaviour analysis log files from drive-by-download analysis contains huge volumes of behavioural data. A large amount of this data is generated from the operating system booting and running normal or

benign system interactions, which are flagged up and reported within behaviour monitoring log files. Behaviour monitoring is a sensitive process as system behaviours within operating systems can sometimes change with simple variables such as a change in the environment state.

Additionally, differences within behaviour analysis environments design principles such as the operating system is not reported. In line with the sensitivity of operating system generated system calls, knowing the difference in observed behaviour may lead to different versions of created filters to be created based on the choice a client system difference organisation makes in the design of honeypots or behaviour analysis environments. The final aspect regarding observed malicious behaviour within the Windows 7 filtered honeypot presents a section from the perspective that utilised systems captured in the analysis of potentially malicious websites. It is therefore by no means, a complete picture as the internet and its vast nature makes this task impossible. This would still provide intelligence on the types of attacks that were active during the duration of this work. More importantly it has the potential to identify active threats which is invaluable to the detection and intelligence gathering parties. The next chapter outlines methodologies used within this work.

CHAPTER 3

Methodology and experiment design

Chapter 3 – Methodology and experiment design

This chapter initially discusses the appropriate methodology applied to relevant research questions. The discussion then explores the importance of successful malware experimentation with a particular focus on prudent experimental design. A set of malware experimentation prudence and guidelines by Rossow et al. (2012) is adapted and then used as the primary experimental framework. The applicability of the framework to the research and experiments are then analysed and presented. Scientific research methods in computer science are discussed by Dodig-Crnkovic (2002), Sjoberg et al. (2007) and Freitas (2009). The chapter then evolves into a discussion regarding data gathering, sample size evaluation and resource acquisition. Part of the methodology involved verification of the experimental setup. Validation of the experimental setup and environment is explored and thoroughly tested. This can be seen in the Appendix B. Lastly, the use of URLs from Twitter is evaluated as another source for gathering malicious website interactions. This can also be found in Appendix B as these were not experiments that directly answered the research questions.

A discussion on research questions and applicable methodology is undertaken, which is organised in the thesis by grouping similar methodologies applied to the relevant research questions. The two relevant types of research approach taken in this work include the use of both deductive reasoning and inductive reasoning. The thesis questions are:

1. Adapting Capture-BAT exclusion lists to Windows 7.

a. What are the key benign system behaviours observed in a Windows 7 operating system that can be filtered in a created behaviour analysis environment?

b. How can behavioural filters (exclusion lists) for a new operating system be created to reduce output of data in log files without losing relevant behaviours that aid in the detection of anomalies?

c. What are the requirements and good practices needed in the development of behavioural exclusion lists?

d. To what extent are the created exclusion lists effective at blocking (filtering out) benign system behaviours?

2. Observed malicious behaviour analysis from drive-by-downloads targeting Windows 7 honeypots.

    a. What are typical observed malicious behaviours in an unpatched Windows 7 honeypot?

3. Understanding the sensitivity of behaviours triggered between different versions of operating systems.

    a. What are the key differential behaviours that an updated Windows 7 displays when executed in comparison to a stock and an unpatched version?

    b. Capture-BAT was originally designed for Windows XP: how does Windows 7 benign system behaviours differ from Windows XP?

For the research questions 1.a, 1.b, 1.c, 1.d, 1.e and 2.a, deductive reasoning is the considered approach. The top-down approach of using theory to create hypothesis

and the testing of these hypotheses with the use of created experiments provides confirmation of whether the theory is correct or not. In some cases, the experiments confirm to what level the hypothesis is the theory correct.

The deductive reasoning approach is applied to the knowledge in unfiltered log files. Using data from benign website analysis, knowledge is then deduced between known behaviours, regular system behaviours and unknown behaviours. The known and confirmed benign behaviours with no trace of maliciousness are then added into the exclusion list. After each iteration, an exclusion list becomes richer with confirmed benign behaviour. This is because exclusion lists are created with the exclusions of observed behaviours where theory regarding a specific process and its interaction within the behaviour analysis lab is studied, hypothesised and then observed. Details from observing behavioural interaction provide an indication of the maliciousness of that particular interaction. Furthermore, deductive reasoning is then applied to testing the validity of the created Capture-BAT instrument created. This disproves the initially applied theory that despite running behind advanced university cyber defence mechanisms, the level of attacks faced by the honeypot is unaffected.

In order to find observable differences between Windows 7 and XP, observations of Windows XP and Windows 7 processes and their default paths are studied. This leads to behavioural comparison where similar processes and interactions containing the same file path and system call are removed. The resulting leftover behaviours once filtered allow the creation of a more accurate conclusion. They can be respectively marked as native behaviours if a given system call interaction is present only in a single dataset, or if the default pathway of the executed process differs from one O.S. to another. Finally, deductive reasoning is thoroughly applied in the assessment of an exclusion list's success rate. Hypothesis assessment measuring

output of Capture-BAT after filtration provides indications regarding the efficiency of created exclusion lists. This experiment observed and quantified behaviours between two systems: one with exclusion list and one without exclusion lists in order to assess the efficiency of exclusion lists. The deductive reasoning seeks to quantify how much the filter system has been able to block benign behaviours in the filtered log file dataset as opposed to the unfiltered dataset, in order to provide quantifiable knowledge on the efficiency of the filters.

Inductive reasoning, the bottom up research approach, is applied to answer the research questions 3.a and 3.b. Thorough observation between differences in datasets lead to the discovery of behavioural patterns and manifestations. These datasets are created by running tests within benign analysis environments: Windows 7 unpatched, Windows 7 Patched and Windows XP. The outputs were then used to formulate theory. This is done by using observed unique and native behaviours in a patched Windows 7 system, where the presence and frequency of unique behaviours that are manifested are used to identify knowledge. The synthesised knowledge allows an analyst to identify of how system behaviours of an analysis environment differ if the system is patched. This is also applied to the predecessor operating system, Windows XP and compared to Windows 7. The knowledge synthesised can be applied the Windows based operating system. The methodology of studying observed behavioural interactions and comparing the output of log files in different versions of the operating systems is however applicable in any behaviour analysis environment whereby processes and system calls are grouped and the typical observed behaviour is classified as differential, anomalies and similar. This method effectively allowed this work to identify the key differential behaviours in different versions of a given operating system.

Having identified key relevant methodologies applicable to the research questions, it is important to note that the experiments created to facilitate knowledge creation should be designed appropriately.

3.1 Safe, prudent and valid malware experimentation

In order to perform this research, it was important to identify and apply a suitable framework. Research is cyber can be sub-categorised in a two areas: Theoretical cyber and Experimental cyber Maxion et al. (2010). This research is driven by the latter and relied on the use of scientific methods to observe (in controlled environments) changes within behaviour analysis environments. In order to facilitate this, an experimental framework was adapted and used in experiment design. Rossow et al. (2012) proposes prudent practices when designing malware experiments. The definitions of prudent as the authors propose within malware experiments are:

1. Experiments designed using the correct datasets,
2. Providing transparency to facilitate experimental replicability,
3. Adapting realism mechanisms within the malware experiments to reflect real-world context,
4. Conducting experiments that do not harm existing network systems.

These malware experimentation guidelines form the basis of the framework utilised within the malware experiments and the malware capture undertaken in the thesis. The framework utilised is also divided into four sections:

1. Correctness of the dataset,
2. Transparency of experiment,
3. Realism in terms of real-world context,

4. Safety.

These sections all contain a number of guidelines that vary in their significance. Usage of this framework provides a malware analyst with some guidelines that would allow experiments that are run to be reported in a manner that produces a reasonably high level of transparency thereby facilitating replicability. Additionally, the guidelines are aimed at also conducting successful malware experiments. Moreover, since these practices were developed based on the evaluation of high-impact journal and conference papers that specifically used malware experimentation such as Bayer et al. (2009), Perdisci et al. (2010) and Rieck et al. (2011). Furthermore, the work by Rossow et al. (2012) is reasonably cited within the malware experimentation community and has grown considerably over the course of the study. It is fair to say that this is a set of reasonably good guidelines to apply in the context of this thesis.

The guidelines provided by Rossow et al. (2012) are provided in table 3.1. Rossow et al. pointed out that these guidelines cannot be reasonably applied to all malware experiments, including the experiments that are carried out within this thesis. This is specifically due to the nature and variance within the investigations. For instance, when analysing a large amount of potentially malicious websites using a client honeypot, the URLs that are analysed are not typically classified as a given distinctive malware family. The reasoning here is because drive-by-download analysis within the real-world is at the forefront of malware detection and therefore faces the detection of new and unseen malware behaviours: mainly due to malware writers having the potential to update a given malicious web server, moments before behavioural analysis takes place. Relevant and applicable aspects of each of the core guidelines will now be explored.

Table 3.1: Prudent practices for designing malware experiment framework (Rossow et al., 2012).

CRITERION	GDL.	IMP.	DESCRIPTION
Correct Datasets			
Removed goodware	A.1)	●	Removed legitimate binaries from datasets
Avoided overlays	A.6)	●	Avoided comparison of execution output with real system output
Balanced families	A.2)	◐	Training datasets balanced in terms of malware families, not individual specimens
Separated datasets	A.3)	◐	Appropriately separated training and evaluation datasets based on families
Mitigated artifacts/biases	A.5)	◐	Discussed and if necessary mitigated analysis artifacts or biases
Higher privileges	A.4)	◐	Performed analysis with higher privileges than the malware
Transparency			
Interpreted FPs	B.6)	●	Analyzed when and why the evaluation produced false positives
Interpreted FNs	B.6)	●	Analyzed when and why the evaluation produced false negatives
Interpreted TPs	B.7)	●	Analyzed the nature/diversity of true positives
Listed malware families	B.2)	◐	Listed the family names of the malware samples
Identified environment	B.4)	◐	Named or described the execution environment
Mentioned OS	B.4)	◐	Mentioned the operating system used during execution analysis
Described naming	B.1)	◐	Described the methodology of how malware family names were determined
Described sampling	B.3)	○	Mentioned the malware sample selection mechanism
Listed malware	B.1)	○	Listed which malware was when analyzed
Described NAT	B.5)	○	Described whether NAT was used or not
Mentioned trace duration	C.4)	○	Described for how long malware traces were recorded.
Realism			
Removed moot samples	C.1)	●	Explicitly removed outdated or sinkholed samples from dataset
Real-world FP exp.	C.2)	●	Performed real-world evaluation measuring wrong alarms/classifications
Real-world TP exp.	C.2)	●	Performed real-world evaluation measuring true positives
Used many families	C.1)	●	Evaluated against a significant number of malware families
Allowed Internet	C.5)	◐	Allowed Internet access to malware samples
Added user interaction	C.4)	○	Explicitly employed user interaction to trigger malware behavior
Used multiple OSes	C.3)	○	Analyzed malware on multiple operating systems
Safety			
Deployed containment	D.1)	●	Deployed containment policies to mitigate attacks during malware execution

3.1.1 Correctness in malware experimentation

One of the main requirements of the research involved the need of creating filter mechanisms from log files to reduce the instance of known goodware and non-malicious behaviour. Typically removing the false alarms within log files should naturally reduce the noise that a malware analyst would have to go through and incorporate during behavioural analysis. This is certainly the case as reported by Li et al. (2010): within malware clustering they concluded that using balanced and well-designed datasets have significant effects on evaluation results. Within this research, analysing malicious drive-by-download behaviours from filtered log files should produce significantly ‘correct’ datasets for analysis.

Within this research, virtual machines are used to build the malware analysis environment: this may cause some variations when compared to using physical, bare metal machines when analysing real-world malicious websites. In order to mitigate some detection mechanics used by malware, the Virtual box based analysis environment will exclude the installation of Guest Additions software. The rationale here is that guest additions include a registry key entry within the HKLM\SOFTWARE\Oracle\VirtualBox Guest Additions within the Windows registry. This can easily be detected and cause malware behaviours to supress triggering.

In addition to this, observed malicious behaviours are verified in light of expected and regularly observed benign system behaviours to provide a higher level of correctness within the dataset as an additional step towards correctness. In line with the prudent guidelines, core components of Capture-BAT's behavioural monitor such as the Process Monitor, Registry Monitor and the File Monitor operate in the Windows kernel mode. The kernel mode is a more privileged mode than observed and captured malware running in user mode. This is important as malware operating within the user mode should ideally not detect the monitoring components as this is likely to lead to malware not manifesting malicious behaviours to avoid detection or perhaps even trigger logging functionality and reporting the presence of a honeypot. It is needless to state that balancing datasets over malware families and ensuring the dataset contains distinct families as proposed by the guidelines is not applicable to this type of malware analysis. This is because the analysis comprises of web-exploits and drive-by-download behaviours and in comparison with malicious Portable Executables that are analysed within sandboxes. Web-based malware from honeypot research can sometimes include malicious file modifications and file writes. These can often be Portable Executable files as part of the attack, but it is important

to note that this is not always the case. Signatures for malware families are created upon analysis of sample and within real-world honeypot research signatures are not always available at the time of analysis, thus making this particular guideline inapplicable.

As a final point, the only identified experiment artefact that may be created and represented differently in the analysis system would include the Windows 7 user name utilised: ‘mp’. This would be different depending on the assigned user name in a different analysis system. Consequently, this affects the output of behaviours that store files in the Windows 7 user folder. For instance, a malicious iexplore.exe was written in the user temp folder: C:\Users\mp\AppData\Local\Temp\Iexplore.exe would be the path an observed behaviour would point towards but in a different analysis environment it could be stored in:

C:\Users\UserName\AppData\Local\Temp\Iexplore.exe.

3.1.2 Realism in malware experimentation

Realism within cyber security is a critical aspect: in order to provide accurate views of real-world malware analysis, which is the main driving force behind malware research, datasets that are created should represent an accurate overview of the analysis target. Furthermore, it is common knowledge as pointed out by Rossow et al. (2012), that lab experiments perform much worse in real-world evaluations. This emphasises the need for malware experiments to be designed to operate on non-lab evaluations, as this reflects the limitations faced by the industry and avoids sources that may adversely affect results. Key aspects of realism are required for this research as accurate insights on observed behaviours is required when analysing real-world and possibly unknown malicious websites. The aspect of ‘currently active’

that provides observable and realistic behaviours is filtered naturally within the experimental approach as the datasets will be gathered and is not dependant on non-active malicious web site datasets. This natural filtration is justified as the use of dynamic drive-by-download analysis uses no emulation and relies on observed manifestations of malware within Windows 7. Additionally, in a real-world setup, client systems would have access to the internet: this means that access to the internet post visitation of a website is required to reflect the nature of real-world compromised clients. Despite allowing internet access throughout the duration of the website analysis being a relatively risky task as this essentially allows potential exploitation by malware writers, it is important that this aspect is reflected. This is because, often drive-by-downloads and web based exploits require additional access time to the internet to probe and request the downloading of additional malicious files. Crist, (2007) discusses some web-server vulnerabilities and prevention methods.

This study focuses on Windows 7: prudent and realistic practices exercises caution within the context of generalisation. Whilst it is likely that file, registry and processes that are shared within Windows 7 and alternative versions of Windows could theoretically display similar behaviour if process operations are not changed from version to version, no claim is made that created artefacts can be applied to alternative versions of Windows. However, as a part of testing, a few system behaviours shared between Windows XP and Windows 7 were found and discussed in Chapter 4. Moreover, malicious drive-by-download and web exploit behaviours that are discussed in this thesis are limited to only a small fraction of the analysed websites that are potentially available in the world wide web and does not include any dark web examples. It is therefore imperative to state that the malware samples

captured, and web-exploits discovered by Capture-BAT are indeed applicable to the Windows 7 analysis environment that was utilised.

3.1.3 Transparency in malware experimentation

At the heart of transparency within malware experiments is providing the means for the replicability of data collection. Reproduction of experiments and within cyber security is often a challenge in this field as stated by Mansoori et al. (2016) and Maxion et al. (2010). URL lists, time of analysis for a given interaction and malware sample, behavioural environment and configuration settings utilised in the malware analysis environment are provided to increase the level of transparency. These are provided within the thesis when required in individual experiment sections. The network connectivity of the analysis environment was reliant upon a virtual bridge from the host to the 'guest' virtual machine and internet access was provided at all times during each analysis process. It is important to note that within real-world malware analysis, replicability of results that are dependent on observed behavioural interactions face a critical limitation. Malicious behaviours and exploits that were once observed on a given date may significantly differ if the same URL is analysed on a different date. These can be due to a range of factors such as: malicious behaviour could be inactive at certain points: exploits could also be updated and significantly modified or dependant on specific IP ranges or simply that compromised servers could have been cleaned and had their vulnerabilities patched. Clearly these are out of the control of the malware experiments as there is no control over the content and exploits used on public web sites.

Other than making datasets available to provide transparency on the analysis process, there are no known additional steps that can be undertaken to facilitate this aspect of transparency. In terms of stating family names to provide transparency,

the captured malware samples in Capture-BAT post drive-by-download analysis that involved the dropping of a Windows Portable Executable have been analysed by Virus Total to provide various signatures. Analysis for finding out potential causes behind observed false positives behaviours is undertaken and some aspects of how behaviours change over time (environmental behaviour drift) is discussed within the log file dataset. Finally, section 3.2.2 - 3.2.3 discuss the malware sample selection process and evaluation process which is required by the guidelines proposed by Rossow et al. (2012).

3.1.4 Safety in malware experimentation

Within the context of this research, real-world malicious websites are visited. These interactions potentially have the ability to compromise the network infrastructure in place as successful propagation from malware attacks can breach the analysis systems. Compromised devices could then be exploited by malware writers as they see fit: the devices may be used for example as part of distributed denial of service attacks, used in the mining of bitcoin, host malicious code, steal user sensitive data and passwords and send spam. Therefore, adequate security systems needed to be in place to mitigate the impact of a potential malware breach. The nature of honeypots has some contradiction to the safety requirements as client honeypots actively seek to be compromised in order to observe drive-by-download and malware manifestations within a live environment that requires access to the internet for interaction. This is inherently a risky process as traffic that a given drive-by-download samples creates upon execution have the potential to cause harm on both the Local Area Network (LAN) and across the Internet. It is however possible to observe malicious behaviour within a controlled environment should suitable security practices be in place. (Alwabel et al., 2014). The framework proposed by Rossow et

al. (2012) focuses purely on containment policies. Thus, the safety undertaken in the malware experiments performed within the thesis was expanded to include safe design.

The deployed containment policy and safe design relied upon a number of deployment practices:

1. The malware analysis client, Capture-BAT runs within a virtualised instance of Debian Squeeze within the Capture-HPC server. This is illustrated in Figure 4.2a in Chapter 4. Virtualisation, as with a large number of security practices does not provide absolute security. However, this offers the possibility of isolating the analysis network from the physical network. Isolation from the physical network infrastructure provides another layer that malware would have to compromise in the event of a network attack. Additionally, the use of Network Address Translation (NAT) masks the internal IP address of devices on the network. The virtual network diagram is provided in figure 4.1 of section 4.4.1 of the thesis.
2. This Capture-HPC server is run on a university network protected by Defence in Depth measures undertaken by the university. While this practice effectively may be seen as defeating the purpose of running a client honeypot, the risks associated by running a client honeypot on a live university network which contains a large number of sensitive operations is likely to outweigh the drawbacks of running malware and drive-by-download analysis on an unprotected network. The validity of this analysis lab under this particular variable is tested in the methodology and experiment design chapter.
3. There are no other devices other than the Capture-HPC server and the Capture-BAT clients running on the same Virtual Local Area Network (VLAN)

which is all interconnected via a bridged interface. While this does not inherently make the Capture VLAN secure from attacks, it would mitigate the risks associated with the compromising of other devices on the network.

4. After every analysis, Capture-HPC resets the Capture-BAT clients into the state that the machine was prior to the analysis process. The revert script does not innately protect the networked devices but it does secure a smaller window of opportunity for an attack to take place as a compromised device will only be compromised for the duration of the analysis.
5. As an added security measure, the Virtual Disk Images (VDI) that are used within drive-by-download analysis are running in an immutable mode. This means any changes to the hard-drive state is deleted after the analysis process as the instance of analysis that is undertaken is left only to execute during the analysis process.
6. The time setting in which a Capture-BAT client is vulnerable and compromised is minimised to use a low time setting (75 - 90 seconds). This time period refers to the interaction time that a Capture-BAT is analysing a given webpage. This minimisation of the analysis time leaves a small gap (less than 2 minutes) which a compromised client is able to cause harm to devices on the WAN and LAN. After this time period, the virtual machine state is reverted. Consequently, downloaded and executed web-exploits are erased and replaced, effectively preventing further exploitation.

3.1.5 Conclusion

In order to perform correct, real, transparent and safe malware experimentation, it was important to apply each of the relevant criterion to the undertaken experiments. Overall, the vast majority of the malware experimentation framework was applicable

and thus included in implementations of the experiments. To conclude the applicability of the framework presented by Rossow et al. (2012) to the undertaken malware experiments, a summary table is created below which highlights actions undertaken to meet the prudent practices identified.

Table 3.2: Applicability of the prudent experimentation framework.

Keys:

Green – applicable to research.

Red – not applicable to research.

Prudent Code	**Description of prudent practice**	**Summary of application to research.**
A. Correct Datasets		
A. 1)	Check if goodware samples should be removed from datasets.	Exclusion lists filter goodware.
A. 2)	Balance datasets over malware families.	Various sources of malicious URLS used.
A. 3)	Check whether training and evaluation datasets should have distinct families.	Evaluation dataset has distinct families -confirmed by Virus Total.
A. 4)	Perform analysis with higher privileges than the malware's.	Capture-BAT operates at the low-level kernel which is higher than the malware.
A. 5)	Discuss and if necessary mitigate analysis artefacts and biases.	Identified Biases identified, and sources of potential bias discussed.
A. 6)	Use caution when blending malware activity traces into benign background activity.	Exclusion lists filter background activity so that only malware activity is observed.
B. Transparency		
B. 1)	State family names of employed malware samples.	Virus Total used to identify known malware samples.
B. 2)	List which malware was analysed when.	Date/time of analysis recorded. This applies for both captured malware samples and web exploit log file.
B. 3)	Explain the malware sample selection.	Discussed in section 3.2.2 - 3.2.3.
B. 4)	Mention the system used during execution.	Full system details available in Chapter 4 and discussed in different experiment set-ups.
B. 5)	Describe the network connectivity of the analysis environment.	Described in section 3.1.3.
B. 6)	Analyse the reasons for false positives and false negatives.	Reasons for system behaviours that occur and are classified as false negatives is an area of study in itself and requires a complex study which is out of the scope of this research.
B. 7)	Analyse the nature/diversity of true positives.	This is not inherently applicable to the research question but as the research captures a number of malicious samples in the wild, behaviours are investigated in chapter 6.

C. Realism		
C. 1)	Evaluate relevant malware families.	Dataset analysed is relevant as they were gathered from real-world evaluations and current web exploits.
C. 2)	Perform real-world evaluations.	Analysis undertaken comprises of real-world websites - including malicious websites.
C. 3)	Exercise caution generalizing from a single O.S. version, such as Windows XP.	Discussions are limited to observed behaviours in a Windows 7 O.S.
C. 4)	Choose appropriate malware stimuli.	Web-rich content requirements such as flash player have been installed on analysis environment but Capture-BAT lacks features that perform stimuli such as clicking on links on webpages and displaying signs of real user interaction (such as typing) which modern malware can sometime require before manifesting. The focus of this work is on the opiminisation of the analysis process of behavioural log file analysis and not primarily triggering malicious behaviour.
C. 5)	Consider allowing Internet access to malware.	Internet access is available during malware analysis.
D. Safety		
D. 1)	Deploy and describe containment policies.	Described containment policy and security measures in place in section 3.14. The framework did not offer much insight in the safety precautions and only focused on the disaster-recovery aspect of malware analysis. It was felt that focus on the prevention of malware propagation within the university network was of utmost importance which justifies the discussion in 3.1.4.

3.2 Drive-by-download resource gathering

The largest contributor to the URL bank came from a wide range of 'potentially' malicious domain lists available on the World Wide Web. These websites contain lists of URLs that were found to exhibit malicious behaviour at a certain point. Clearly, malicious webpages are highly dynamic in their nature of delivering web exploits: a given malicious website might only be active at certain times or only active towards a particular geo-location which are easily enabled by services such as GeoIP (http://www.geoip.co.uk/) which are widely available on the net. Multiple freely available databases were used in order to obtain a balanced dataset that attempted to include as many different families as possible – as long as they exploited Windows 7 systems. Examples include:

- Malware domain list (https://www.malwaredomainlist.com/mdl.php)
- The malc0de database(http://malc0de.com/database/)

- hpHosts (https://www.hosts-file.net/)
- Malware Domains (http://www.malwaredomains.com/)
- ZeusTracker (https://zeustracker.abuse.ch/blocklist.php)

Duplicated URLs were checked and removed before the analysis process using a simple string match function built in Capture-HPC_NG.

Additionally, lists of potentially malicious, but mostly unknown URLs were gathered on Twitter, targeting tweets which contained a URL and the hashtag ‘world cup’ during the 2014 world cup. While this source proved to be vast and potentially highly scalable in terms of having much more URLs than the single Capture-HPC server could handle, it was later found to be a fairly poor source in terms of gathering malicious behavioural data and malware binaries due to the larger number of found benign URLs.

The third source of URLs was supplied by a large organisation (unnamed as protected by a non-disclosure agreement): the URL lists supplied contained a mix of both malicious and potentially benign URLs that was generated based on the usage of the employees of that large organisation.

These sources contained a mix of known malicious web sites, potentially malicious websites and benign websites. It was important to attempt to have varied categories as these would allow a finer diversity in behaviours that could be captured, evaluated and included in behavioural filter lists. Malicious websites that are newly active and analysed can often contain samples of malware that is unknown or have not yet been detected. These samples could additionally be polymorphic variants of known malware. Realistically, unless these websites have been analysed and the malware exploit is known, it is difficult to claim that the dataset contains a good variety of

malware families. This is thus an example of the limitations that research in malware analysis and experimentation faces when using real-time and real-world malware datasets.

3.2.1 Creation of the data corpus

In order to understand malicious drive-by-download behaviours and web-based exploits, a specialised data set was required. Additionally, non-malicious datasets were also required for various experiments that were carried out because they would be required to build the body of knowledge that expert driven behaviour filtering required. This section starts by identifying the underlying reasons for creating these datasets and evaluates the sample size that was used in the creation of the data sets. Finally, the resources used in the creation of the malicious behavioural bank are discussed and the sources evaluated to identify degrees of realism and to avoid generalisation.

A major requirement for the experiments was to identify a large number of known malicious URLs that would need to be analysed. The number of potentially malicious websites scanned is 110,000 over the course of three years. This large number of malicious URLs is required because within drive-by-download analysis the trend for malware writers follows a pattern which would minimise detection. The nature of short-lived malicious websites reflects that, as often a malicious website which is found to be malicious when initially analysed could no longer be malicious upon secondary analysis a few hours or days later. To overcome this issue, a large number of websites (110,000) were scanned in the hope that a high number of malicious behaviours are captured, and a real-world centric view is observed. With the availability of Web services (such as Amazon's web services) it is becoming increasingly cheap for malware writers to rent out cloud-based servers which are

then used as malicious servers to infect client systems with drive-by-downloads which is yet more reasons for dynamic analysis output to be more focused on the known malicious and anomalies.

Consequently, the affordability and availability of rented web servers indirectly allows the nature of short-lived malicious websites which are then abused by malware writers to avoid detection. In order to obtain an accurate overview of malware and drive-by-download behaviours, a reasonably high sample size would be required to counter the limitations faced by behaviour based analysis. By having a high sample size, a larger amount of 'currently malicious' websites would be analysed which would provide a deeper and arguably more accurate insight into malicious drive-by-download behaviour.

Other experiments which do not seek to identify malicious behaviour and malware patterns. Such experiments include attempting to evaluate performance of exclusion lists or identify changes between benign system behaviours that are generally static. These experiments require significantly fewer experimental runs to understand the system behaviour. It is reasonable to assume that the majority of benign system calls were identified after running 500 samples and seeing no new behaviour manifestation by the honeypot. This is a reasonable assumption as no new benign system calls were observed suggesting that the vast majority of benign behaviours were captured. In terms of experiments requiring the use of purely benign behaviour, the sample sizes used throughout this work varied from 800 - 3,011 URLs analyses. This number was chosen to have a higher assurance level that the observed behaviour are in fact expected system calls by the O.S. and applications installed. It was important to ensure that the majority of expected system calls were experienced as one of the problems that is being solved was filtering out the expected benign

behaviours and allowing false positives to be undiscovered would only result in a lesser amount of filtered data.

3.2.2 Justification for the creation of the malware data corpus

Within malware analysis, there are several datasets available on websites such as Virus Total, Malwr.com and malware 'dump' websites (e.g.: http://contagiodump.blogspot.co.uk/) which contain labelled samples (in the form of binary files or executables) from different periods of time. Unfortunately, these did not contain behavioural information regarding the drive-by-download itself or behavioural information about the web exploit as typically these datasets were generated by creating a sandbox environment and then executing the malware sample. Binary analysis assumes the malware sample has been delivered onto the system and relies on actively executing the file to observe behaviours. The behaviour that can be observed within an O.S. as a malicious website interacts with the system can provide information on attack and infection vectors. Work specifically on drive-by-download behaviours within Windows 7 were found to be scarce and typically did not include discussions on behaviour filtering, thus the topic was relatively unexplored. It was imperative that this type of dataset was created if insight about studies within Windows 7 web exploits and behaviours was to be conducted.

In addition to this data type, there are a large number of malicious websites URL hosting which contain large volumes of regularly updated malicious websites. However, those malicious domain lists are typically aimed at providing blacklists domains and thus do not provide observed behaviours during the initial interaction between the client system and a website. Some of these domains are found to be malicious at some point but the behavioural characteristics that would suggest maliciousness are not provided. Furthermore, following prudent practices of

malware experimentation by Rossow et al. (2012), there was a requirement to put measures in place that would limit malware samples used in a study, so that it does not generalise over multiple and inactive variants that attack different versions of operating systems. In the study's case, malware that are proven to actively attack a Windows 7 client environment are of key interest as the research focuses on Windows 7 behaviours in the creation of the behavioural filters and observed malicious behaviours within the Windows 7 operating system. This is logical as within the world of malware, if datasets are not filtered to at least a given operating system, it is likely that conclusions may be based on more general malicious behaviour. Within this industry, cross platform malware can often be observed as discussed by Stange (2015) and Upguard (2016). It is therefore important that the study commits to detection on a specific operating system throughout experiments as opposed to multiple operating systems which would face detecting cross platform malware repeatedly. The limitations in available datasets and the aspiration to capture different malware samples that are not cross platform justifies the creation of a specific malicious behavioural log file data corpus.

The main dataset is inclusive of .log (log file) data based on the interactions that occur during runtime of a web browser visiting a malicious webpage. Additionally, Capture-BAT provides the ability to capture the modified and deleted files during the interaction.

3.2.3 Sample size evaluation

URLs are the main resource required by drive-by-download behaviour analysis. 5,132 log files were marked as malicious out of a total possible of 110,000 analysed URLs. These consisted of malicious, potentially malicious, inactive (short-lived malicious webpage) and reactive (malicious webpage detects Capture-BAT or the

virtualised environment and prohibits execution) URLs from a fairly diverse range of real-world sources. Previous malware studies that are relevant in terms of using malicious URLs have a tendency to utilise a wide range of malware samples: Amorim & Komisarczuk (2012) cluster 17,000 behavioural log files from Capture-HPC. Cova et al. (2010) uses 823 malicious samples and a benign URL dataset of 11,215 URLs. More recent studies by Tanaka & Goto (2016) utilise 43,000 URLs over the course of 18 months. Song et al. (2010) used 119 cached 'in the wild' samples. These malware studies show the wide range of malware samples used in different studies and are in line with the sample sizes created for this study.

Rieck et al. (2011) analysed 3133 reports of malicious behaviours. Rossow et al. (2012) performed dynamic malware execution by analysing a sample of 10,670 files with different MD5 hash values. Whilst these were not inherently malicious URLs being analysed and focused at malicious files, it is fair to conclude that the larger the sample size for malicious behaviours, the more likely a large variety of malicious behaviour is to be observed. Unlike malware binary analysis where having a large number of unique MD5 files in the analysis could be derived from polymorphic variants of the same family, the process of capturing drive-by-downloads for behavioural analysis, requires a reasonably large sample of data to identify as many unique behaviours as possible. It is therefore concluded that the sample size of over 110,000 potentially malicious websites analysed yielding 5,132 malicious log files is reasonable considering the amount of ever increasing malicious web servers.

The sample of 3,011 potentially malicious URLs analysed for the purposes of calculating filter efficiency for filtering benign behaviour within created exclusion list was determined experimentally. This is because benign behaviour tends to be rather static and it was observed that it was highly unlikely (no new benign behaviour found

after analysing 1500 URLs) to find significantly new behaviours by analysing URLs. It was shown in the experiment that the created exclusion lists were also filtering the vast majority (96.96%) of benign behaviours nearly over two years after their creation. This sample was more than sufficient to determine benign behaviours. However, it is important to note that this sample is likely limited in terms of analysing malicious data due to the limitations on behaviour analysis environments that Capture-BAT faces and the large amount of potentially malicious websites that are not active at the time of analysis.

3.3 Experimental setup

This section depicts the main Capture-HPC client honeypot setup which is altered accordingly in different experiments. The main source of data gathering is undertaken by the use of a client honeypot, Capture-HPC. The type of data that is gathered are system behavioural files in the log (.log) file format. Additionally, if a new file is created or modified during the analysis process, the new or changed file would be available in zip format. As often observed within drive-by-download attacks, the creation of a malicious Portable Executable within the Windows O.S. can be seen. An overview of the methodology is provided below and the detailed experimental setup can be viewed in section 4.5.1 – 4.5.2 of Chapter 4.

1. Virtualised Windows clients are setup to be as vulnerable as possible by containing aspects of an un-secure Windows system.
    a. These include: disabling User Access Control (UAC), disabled Microsoft defender and ensuring that anti-viruses were not running on the analysis client.
    b. Running Windows 7 with no critical and security updates.

c. The installation of applications that allow for web browsing exploits yet enrich the user experience by allowing application dependant features (such as Flash player).
d. Run Internet Explorer 8. According to Rahul (2014), this is the most vulnerable web browser at the time of the study and is likely to face most attacks in the client honeypot environment.

2. Vulnerable clients visit a given website and record behavioural interactions which are then sent back to a server for storage before being reverted back to a clean state, ready for the next behavioural URL analysis. This is controlled by two main mechanisms: first, the revert script used by Capture-BAT. Second, the immutable state of the virtual hard-drives which prevents permanent changes saved to disk.
    a. The outputs here are the log and zip files aforementioned.
3. Behavioural log files are then analysed.
    a. Unique behaviours are counted and the creation of a behavioural table is undertaken.
    b. Behaviours are evaluated and classified between benign, unknown or malicious.

3.3.1 Experimental methodology justification

The need for experiments to be carried out in this research is gathered from the initial requirement of measuring the effectiveness of the testbed. Additionally, throughout the development, there is a need to assess created artefacts such as: measuring the effectiveness of exclusions lists introduced in Chapter 4. Experimentation is ideal for this particular task as an observed number of behaviours can suggest the amount of raw unique behaviours that are blocked by exclusion lists.

This provides quantifiable evidence within the context of efficiency testing. In order to do this, measurements of the usual unfiltered data would need to be compared with data that the created artefacts filtered out.

Furthermore, validating gathered data to ensure that the honeypot setup is facing malware attacks behind a level of protective services in a university network is crucial. If observed behaviours do not contain malicious behaviours due to an internal defence mechanism or resource preventing malicious code being delivered and executed, it would have been futile to continue data gathering over the course of the study. The primary goal of running experiments is to observe benign system behaviours within a Windows 7 environment that is visiting a benign webpage. These then allow behavioural profiles and behavioural tables to be built based on recorded interactions. These can be used in similar Windows 7 based run-time analysis environments to filter out results or other instances of Capture-BAT with the same set of applications and versions.

3.3.2 Capture-BAT

Capture-BAT is a behaviour analysis tool developed by Seifert et al. (2007) and is the main tool used within this research to capture both benign system and potentially malicious behaviour. Capture has been around for many years and was originally designed to operate on the Win32 operating system, Windows XP. A behaviour analysis tool seeks to analyse the behavioural interactions that are generated as a result of system calls within an analysis environment. This means that the state of a system can be monitored by monitoring application and operating system behaviours. Capturing these behavioural interactions allows the malware analyst to view log files which offers a large amount of insight into how programs and the operating systems interact without the need for the source code or reverse

engineering. Capture-BAT operates at low level kernel which is a higher privileged level than malware operating in user level. This is important because one of the highly desirable goals of malware analysis is to execute the analysis with higher privilege than malware to reduce risks of the analysis environment being detected by malware.

This dynamic analysis technique has been well established for a long time. However, adaptations to understand operating system behaviours and applicability to the most popular operating system Windows 7 was an area that was not being reported academically. This is the area of focus targeted by the thesis. In order to ensure certainty that malicious behaviour is being detected with the created set-up of Capture-BAT, it is of crucial importance that the setup itself is tested.

3.3.3 Duration of malicious webpage analysis

This experiment looked at the effect of client honeypot analysis duration. The purpose of this investigation was to see if there was a difference between the behavioural log files if the duration of the analysis for a webpage was allocated a different number of seconds. It is not uncommon for malware to have been set ‘sleep timers’ by a malware author and actual execution would occur upon some user activity or timer expiration. However due to the vast number of potentially malicious URLS there are to analyse (110,000 URLs in the case of this study), the decision for execution duration per webpage should factor in the high volume of tests required. Figure 3.1 below shows the full activity sequence required by each website analysis. Table 3.3 below also shows the required analysis time at different amounts of seconds on the page analysis setting within Capture-HPC.

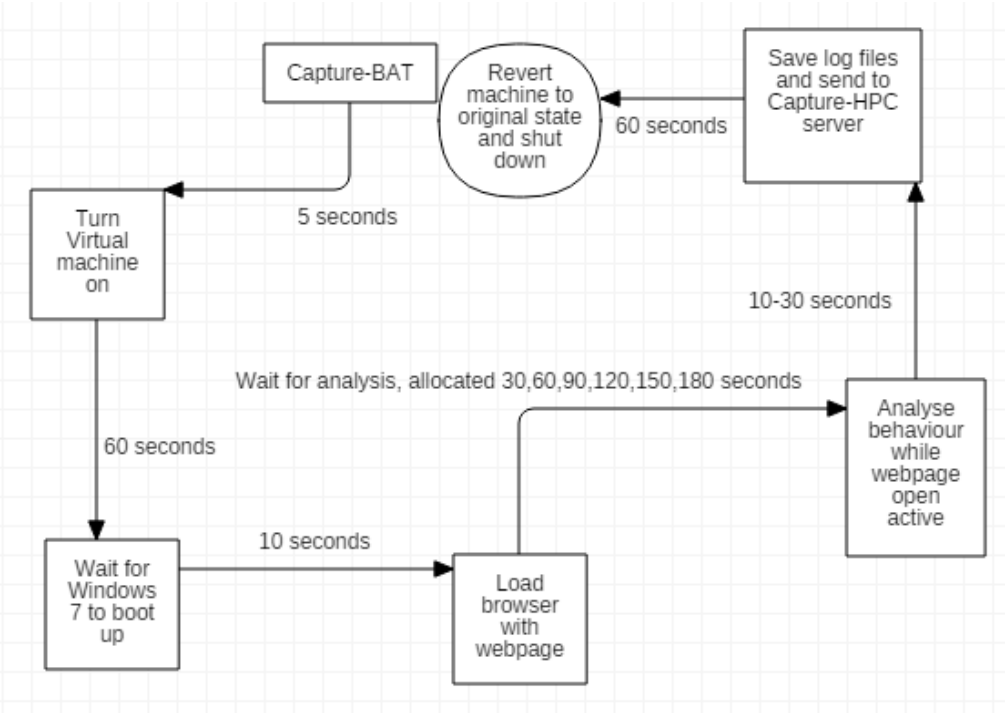


Figure 3.1: Events and required time upon Capture-BAT boot up.

Table 3.3: Tests for required time for website analysis in seconds. The chosen settings for Capture-BAT is provided in Chapter 4.

Action	**Required time per activity (seconds).**					
Turn virtual Machine on	5	5	5	5	5	5
Boot Windows 7	60	60	60	60	60	60
Load browser and webpage	10	10	10	10	10	10
Analysis of webpage	30	60	90	120	150	180
Save log files, revert and shut down	60	60	60	60	60	60
Total required time	165	195	225	255	285	315

A Sample of 140 URLs were chosen (these contained a mix of known malicious and benign URLs) and the test was executed multiple times with different duration variables of 30 seconds, 60 seconds, 90 seconds, 120 seconds, 150 seconds and 180 seconds. From the results gathered, log files for 60 second tests were displaying a few less behaviours than log files running for 90 or more seconds. At the 90

second threshold compared with 120, 150 and 180 respectively there were no further additional differences between observed behaviours. In terms of the dataset created, the decision was made to allow 90 seconds per URL to execute excluding virtual machine load up, revert and shutdown times. This is because from the results gathered, there were no visible benefits from letting an analysis run for a longer time. Therefore, a maximum of 5 minute per total webpage analysis was decided to be allocated which included the to the 90 seconds found to be required as it was observed that if two virtual Capture-BAT clients were being run simultaneously, the boot up and web browser analysis would take up to 30 seconds longer. This allowed sufficient time for all the required activities to take place and had a buffer of a few seconds extra per page should extra time had been required by Capture-BAT. The time of 5 minute per page was used in a similar setup by Burnap et al. (2015) and this work also assumes that a degree of malicious activity will be triggered within the first few minutes of analysis. It was concluded from log file analysis that increasing the duration of website analysis by the factors considered in table 3.3 did not affect the sample set of 140 URLs yet this was necessary to limit the amount of time per webpage as there were a large sample of potentially malicious URLs to visit. The issue faced by sandbox analysis where some malware samples are given sleep timers to bypass detection by run-time analysis is beyond the scope of this research but it may be likely that future research may detect more drive-by-download attacks if honeypots accommodate this limitation or are simply allowed to run for long periods of time.

Additionally, Long term studies may help in providing insight on how compromised machines are exploited and used by malware writers and what further malicious behaviours can be observed from the large number of malicious executables that

remain un-executed. Long term studies have been quite rare within this field but recently, (Tanaka and Goto, 2016) show that some malicious websites (10% of their 43,000 dataset) stay active for a long time with several revival periods. These findings show that malicious URLs can be studied further by performing interactions and allowing the compromised system to be used long term to further understand the malware behaviour. However, the authors' research does not focus on the long-term behaviour analysis of how compromised clients are used by malware writers. The long-term behaviour analysis of a particular malicious URL is out of the scope of this research due to the possible associated consequences of leaving a compromised machine running for extended periods of time. This compromised client could for example launch attacks, steal data or mine Bitcoin. It is identified that this may be a direction for future research within drive-by-download analysis.

As a result of the methodology applied, these datasets were created for the study:

Table 3.5: Datasets created for experiments carried out in the thesis.

Dataset	Description
5,132 dataset	This malicious log file dataset was created by analysing 110,000 potentially malicious websites gathered from malware domain lists aforementioned. This dataset is analysed in Chapter 6.
30,11 dataset	This dataset is created as part of an experiment in Chapter 5 which measures the behavioural filter efficiency. The dataset is created using unverified webpages and seeks to identify the extend of which the benign filters work. This dataset contains two sets of 3011 log files; one created with exclusion lists and the other created without exclusion lists.

800 dataset	This dataset is used in Chapter 5 to identify if there are changes in behavioural interactions that differ if the behaviour analysis environments has different versions of the same operating system. The dataset contains two sets of 800 log files created using Capture-BAT on patched and unpatched operating systems.

3.4 Testing the validity of experimental setup

Having implemented and configured the behaviour analysis environment, there was a requirement to test the setup in the real-world to ensure attacks against the client systems were actually taking place. This is done as part of an experiment and can be viewed in Appendix B, section 3.4.

3.5 Alternate resource gathering evaluations

URL lists of potentially malicious websites are the crucial resource for this study. It was therefore crucial to explore alternative ways of obtaining active potentially malicious URLs. As part of this requirement, the social network Twitter was assessed as a potential for obtaining malicious URLs. This is presented Appendix B section 3.5.

3.6 Roundup

This chapter identified and explored the undertaken research methods and their applicability within the thesis. Both deductive and inductive reasoning were applied to various research questions. Additionally, the adaptation of prudent malware experiment guidelines provided by Rossow et al. (2012) was applied to the thesis. This discussion identified key concepts of the guidelines that would be applicable to

experiments that were undertaken. Non-applicable guidelines were identified and justified. The extension of the safety guideline in the adapted framework to include safe practices that was missing was undertaken. This was a requirement in the case of this research due to honey clients predominantly being run alongside existing university network infrastructure which would require additional safety practices that were not present in the guidelines by Rossow et al. (2012). The final artefact was a table with the adapted guidelines and how each guideline was applied within the study.

The chapter discussed the creation of data corpus which included sample size evaluations and URL resource management. Generic experimental setup and the chosen apparatus, Capture-BAT was discussed. This provided means to depict an overview of the main experimental setup which would have variations in different experiments. These variations are discussed in the respective experiment sections within the thesis. Finally, two experiments are carried out (presented in Appendix B): firstly, a validation test is undertaken on the experimental setup which proves that the setup is suited for purpose despite being placed on a university network and defence system. Secondly in line with the resource gathering discussion, an experiment using social media to gather possible resources for analysis is undertaken. In this experiment the conclusion was that Twitter is not a viable source for URL gathering when applied to this work as the URLs that are shared in tweets are far too rarely found to contain malicious behaviours that attacked the Capture-BAT client honeypots.

The findings of this chapter did conclude that the experiment setup is designed in accordance to prudent guidelines and was valid in the real-world scenario. As it is imperative that Windows 7 system behaviours is not an area of research that has

been thoroughly reported, there is a need to explore benign system behaviours within this operating system. This is an important aspect of drive-by-download research as it is important to separate known benign and normal system behaviours from malicious behaviours. Undertaking this activity allows a malware analyst to filter datasets which would typically contain a large volume of known benign behavioural interactions. This concept is supported by the theoretical framework in the context of malware analysis as Rossow et al. (2012) depicts the importance of separating goodware data from malware data. In the context of this thesis, goodware data would be the benign system interactions that take place upon each bootup of a Windows analysis environment. Therefore, the next chapter seeks to provide understanding of the benign behaviours in the creation of expert driven behaviour filters. This will explore the main research question related to the expert driven behavioural filters. As a last point, it is possible that filtering benign behaviours could lead to more positive outcomes in terms of malicious behaviour classification and clustering.

CHAPTER 4

Expert driven behaviour filter development

Chapter 4 – Expert driven behaviour filter development

This chapter introduces and discusses expert driven behavioural filters which seek to filter out known behaviour using behaviour knowledge. The behaviour filters are adapted into exclusion lists when these are used in the Capture-BAT software. This is because the software is able to use the knowledge resent in exclusion lists to filter out exclusions. Within the thesis, references to created exclusion lists refer to the expert driven behavioural filters that were created.

Initially the importance of working exclusion lists in order to gather a Windows 7 malware data corpus is explored as well as the challenge of determining complex software behaviours even when a given operating system is left running idle. Operating system behaviours are explored and labelled accordingly between malicious, benign or unknown (grey) behaviours in the aim of helping classify behaviour in future behavioural studies utilising dynamic analysis within malicious binary sandboxes and Web exploit analysers. System behaviours are compared between Windows 7 and Windows XP which results in a number of discoveries within benign system behaviours. Exclusion list goals are explored and a methodology in the creation of behavioural exclusion lists is proposed. The chapter then evolves into the lab set-up used to create and update exclusion lists, which minimises the output of malicious log files in addition to providing insight on labelling real-world benign and grey behaviours. It is important to note that behavioural interactions that suggest malicious activities in malicious log files are not lost if the behavioural filter is designed using the expertise of a behavioural analyst and in a risk-averse manner. Shortcomings and possible solutions of exclusion lists are explored. Finally, contributions are discussed.

4.1 Background

Dynamic behaviour analysis is the process of understanding the inner workings of software or process by observing omitted behaviours and interactions with programs when these are executed in real environments. Dynamic behaviour analysis can be performed without directly reverse engineering: this overcomes the need for potential malware source code which is not available in the majority of malware exposures. Typically, this approach tends to be more accurate but also much more computationally intensive than the static analysis approach as dynamic analysis relies on execution of service in real, non-emulated systems. Behaviour analysis is typically undertaken in sandboxed environments with the aim of protecting malware from propagating and infecting networked systems. Automated malware analysis in sandbox environment has been carried out by Willems et al. (2007) (CW Sandbox) and more recently by Cuckoo sandbox (Cuckoo, 2014) as well as several Anti-Virus vendors. The typical output of the analysis is in report format detailing system calls. These system calls can then be used by a malware analyst to determine whether omitted behaviours are benign or malicious.

Similarly, Capture Behaviour Analysis Tool (Capture-BAT) has been around for a number of years Seifert et al. (2008). Progressively within the years since Capture-BAT was originally released, hundreds of patches for Operating Systems (O.S.), browsers and applications have been released. When these new versions of O.S., browser and applications are installed within an environment, they manifest as large numbers of propagating behaviour changes. Furthermore, the majority of research undertaken, which used Capture-BAT for drive-by-download capture were done using Capture-BAT on Windows XP with various service packs as can be seen by Aval et al. (2008), Seifert (2010), Seifert et al. (2008), Lin et al. (2010), Van et al.

(2011) and Amorim & Komisarczuk (2012a). Windows 7 however, has been a predominantly untouched area within system behaviour studies despite being one of the most popular O.S. as the literature review identified.

A useful yet critical component of Capture-BAT is a list of a system's behavioural exclusion list Seifert (2008). These behavioural exclusion lists allow Capture-BAT to understand and classify a given behaviour as malicious or benign. Malicious behaviours are reported while benign behaviours are ignored. This can be thought of as a fairly low-level filter for system behaviours as every event that takes place on a machine is logged by Capture-BAT. Similarly, the exclusion lists developed for Capture-BAT could be used as the basis for filtering logged events from sandboxes such as Cuckoo. Prior to this work, it was not found that Windows 7 behaviour studies to understand the new operating system's behaviours and interactions were studied and classified. Furthermore, adaptations within Capture-BAT to run on Windows 7 have not been identified or attempted and thus remains a particularly unknown area of knowledge despite being actively used to identify malicious behaviour.

4.2 Research requirements

A critical part of the research involved the need of filtering the vast amount of malware available into malware that actively seeks to infect and compromise Windows 7 environments through drive-by-downloads. This is reflected in the first research question. At the start of the study, Windows 7 held a higher percentage of the O.S. usage platform but even today according to WC3 School's O.S. Platform statistics, it is still the most widely used operating system (38.7% August 2016) followed by Windows 10 (26.10%). Therefore, in the development of our malware capture environment the primary aim was to identify and design an environment

where a large amount of malware that targets Windows 7 systems can exploit and manifest. The behavioural manifestations of these drive-by-download attacks would form part of answering identified research questions on what do Windows 7 malware typically target and exploit. The required environment would need to be using a number of widely used applications such as Flash Player, Java and Real Player, Mozilla Firefox and Internet Explorer with fairly vulnerable and default settings. The rationale here is to provide the test bed with a respectable number of exploitation vectors for malware to exploit and compromise whilst providing the maximum possible knowledge of the attacker's techniques (Peter & Schiller, 2011).

4.2.1 Limitations of Capture-BAT within the Windows 7 environment

Capture-BAT is a primary behavioural detection tool created originally for the Windows O.S. with primary deployment on Windows XP. Originally Capture-BAT would run predominantly on the virtualisation software, VMware. In this research, the aim was to focus on malware behaviour within the Windows 7 environment. This meant that adaptation of Capture-BAT from Windows XP to Windows 7 was an initial requirement. As mentioned in the background section of Chapter 4, the vast majority of previous research focused on Windows XP as their analysis environment. Running Capture-BAT on a different O.S., would not provide filtered behavioural log files as Capture-BAT was not designed on Windows 7. The new O.S. would inherently have a large number of new system behaviours that would be benign and a large number of unknown behaviour. By default, any new behaviour that is not present in the exclusion list is marked as malicious Seifert et al. (2007).

It is evident that should Capture-BAT with unmodified exclusion lists be run in Windows, all the URLs would be marked as malicious. The reasoning behind this is simply because Windows 7 which in comparison to Windows XP has a number of

different behaviours and system calls due to the new file system changes, and increased number of services available in Windows 7. Additionally, this meant existing processes between both operating systems would have different default directories that would cause a large number of false-positive malicious behaviour entries in the behavioural reports. To prove this was the case, the output report of a Windows 7 Capture-BAT with a benign web server interaction is provided in the Appendix B and can be partly viewed in Figure 4.0a. After a substantial number (>~2100) of benign websites were set for behavioural analysis it was noticed that a large amount of similar behaviours was observed. The observed behaviours were concluded to be 'normal' O.S. or benign behaviours subsequent to log file investigations. Investigated log files were cluttered and made the process of behavioural malware analysis for a malware analyst much more time consuming than if the log file would only contain known malicious behaviours or unknown and new behaviours.

This process was hindered by the way Capture-BAT works: Initially the Capture server, also known as Capture-HPC would send a list of behaviours to the Capture client, Capture-BAT. This list of acceptable behaviours as introduced in section 4 is known as an exclusion list and would contain behaviours which are either benign or malicious. Post adaptation, all the analysis resulted in the malicious classification by Capture-BAT: but this made sense, as there were no working exclusion lists for Windows 7.

4.2.2 Identification of behavioural vectors within Windows 7

In addition to the lack of O.S. behaviour filtering for Windows 7, each of the installed applications within the created behaviour analysing environment displayed an additional large number of benign behaviours. Clearly these mean that there was a

requirement for several more entries into the exclusion lists as there are significant numbers of benign behavioural interaction that are generated as a result of applications being installed and executed on the analysis environment. Examples include the creating and saving of log files or running at start-up. Within Capture-BAT, the development of exclusion lists to filter behaviours for three major behavioural vectors were required: Windows 7 O.S. behaviours, the web browser behaviour and the installed applications within the environment which caused system calls upon each boot. These were therefore prioritised and required the inclusion of benign behaviours. Section 4.5 explores exclusion list development.

Typical system calls and interactions that are generated by an idle Windows 7 behaviour analysis environment designed to replicate a Windows 7 client can be observed in section 4.4.1 - 4.4.4.

4.2.3 Exclusion list creation rationale

The necessity for the creation of exclusion lists is justified as there is an apparent lack of focus on exclusion lists development or even guidance on the development of good practices at the creation stage as identified in the literature review. In practice, this lack of focus is highly problematic as exclusion lists are the main components involved with the decision-making process when using Capture-HPC in drive-by-download behaviour analysis on the interaction between client and web server at the point where a web server is requested to provide website content. Being at the forefront of the classification of potential drive-by-download attacks, it is therefore imperative that developed exclusion lists are created with the goal to provide the least possible benign behaviours in the output log files. Redundant and normal behaviours should be excluded from log files to allow instances of Capture-BAT running in the Windows 7 O.S. so that Capture-BAT only flags malicious behaviours

as malicious. Naturally the amount of noise from behaviours gathered was limited. Consequently, it is reasonable to assume that forming malware groups based on particular malware behaviour are more likely to be successful absent the streams of regular and confirmed benign behavioural data.

To reinforce the scale of this issue, Figure 4.0a below shows an example behavioural log file where regular and non-malicious system behaviours and interactions are taking place. These log file entries are recorded upon a system starting up, loading a web browser and idling for about 2 minutes.

Note: the website was verified using the proposed methodology in section, 4.5.2 prior to the investigation).

As can be seen from the log file above, the sheer volume of system interaction for a single website is indeed quite overwhelming to a behaviour analyst with over 300 behavioural interactions in a single Capture-BAT instance. The full log file for this particular analysis is available in Appendix B to outline the sheer volume of data from a single analysis.

This website (at the time of testing) was benign but Capture-BAT within the Windows 7 environment marks it as malicious, along with recording all the usual / benign O.S., web browser and application behaviours just as we hypothesised in section 4.2. Having explored the need for exclusion lists, the discussion will now evolve into the creation process.

Behaviour
"registry","18/8/2016 17:33:8.948","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:33:8.958","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:33:8.968","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:33:8.968","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:33:8.968","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:33:8.968","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:33:8.988","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:33:9.58","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:33:9.108","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:33:9.108","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:33:34.248","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\22","-1"
"registry","18/8/2016 17:33:34.248","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\24\ffffffffffffffffffffffffffffff00","-1"
"registry","18/8/2016 17:33:34.248","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\24\ffffffffffffffffffffffffffffff01","-1"
"registry","18/8/2016 17:33:34.248","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\24\ffffffffffffffffffffffffffffff02","-1"
"registry","18/8/2016 17:33:34.248","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\24\ffffffffffffffffffffffffffffff03","-1"
"registry","18/8/2016 17:33:46.766","C:\Windows\System32\svchost.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\ConfigValueEssNeedsLoading","-1"
"registry","18/8/2016 17:33:46.766","C:\Windows\System32\svchost.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\List of event-active namespaces","-1"
"registry","18/8/2016 17:34:2.959","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\WMI\Autologger\Circular Kernel Context Logger\Status","-1"
"file","18/8/2016 17:34:4.642","C:\Windows\System32\svchost.exe","Write","C:\Windows\System32\wdi\{86432a0b-3c7d-4ddf-a89c-172faa90485d}\{82957e1b-5038-41cb-a61c-19f20a7fc80b}\snapshot.etl","-1"
"file","18/8/2016 17:34:4.652","C:\Windows\System32\svchost.exe","Write","C:\Windows\System32\wdi\{86432a0b-3c7d-4ddf-a89c-172faa90485d}\{82957e1b-5038-41cb-a61c-19f20a7fc80b}\snapshot.etl","-1"
"file","18/8/2016 17:34:4.652","C:\Windows\System32\svchost.exe","Write","C:\Windows\System32\wdi\{86432a0b-3c7d-4ddf-a89c-172faa90485d}\{82957e1b-5038-41cb-a61c-19f20a7fc80b}\snapshot.etl","-1"
"file","18/8/2016 17:34:5.62","C:\Windows\System32\svchost.exe","Write","C:\Windows\System32\wdi\ShutdownPerformanceDiagnostics_SystemData.bin","-1"
"file","18/8/2016 17:34:5.804","C:\Windows\System32\svchost.exe","Write","C:\Windows\System32\wdi\BootPerformanceDiagnostics_SystemData.bin","-1"
"file","18/8/2016 17:34:5.864","C:\Windows\System32\svchost.exe","Write","C:\Windows\System32\wdi\{86432a0b-3c7d-4ddf-a89c-172faa90485d}\S-1-5-21-78034117-1329648361-637219273-1001_UserData.bin","-1"
"file","18/8/2016 17:34:7.256","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\1F39B5CFACECFDE48DB25BCA2231FAC6_135A427F1ED873A4BF5097F7A809FA2A","-1"

Table 4.0a: behavioural log file upon visiting a benign website initially before the creation of exclusion list. This is a snippet of the log files and the rest has been excluded from the thesis as it is over 100 entries long.

4.2.4 Aims and goals of exclusion lists

The aims of creating successful behavioural exclusion lists are explored:

1. Benign and normal O.S. system calls should be ignored and not present in a log file to avoid a benign interaction from being flagged as malicious.
    - As explored in the background and literature section, the sheer volume of malicious web server released on a daily basis is much more than dynamic analysis can cope with. Consequently, within the cyber security industry, there's always been a need for sorting rapidly and effectively between malicious and benign behaviours so appropriate attention can be better spent on the unknown and the malicious. A successful behavioural exclusion list should therefore be able to filter known benign behaviours to facilitate this requirement and reduce the amount of log files that need analysing altogether.
2. Within a malicious log file, known and confirmed benign behaviours should not be present as this would add unnecessary noise to the log file which would decrease accuracies in analysis.
    - It is fair to state that within a data set where a large number of noise is present, analysis would be hindered similar the curse of dimensionality in machine learning.
3. Any malicious or newly observed (unknown) behaviour should be marked as malicious.

- It would defeat the purpose of drive-by-download and malicious behaviour analysis if malicious behaviours are not flagged up. Correspondingly, if unknown behaviours are present, they should be analysed and verified by an analyst to ensure they do not provide clues that suggest maliciousness.

4.3 Methodology used in developing expert driven behavioural filters

The methodology used is adaptable to any other versions of operating systems that are used in behaviour analysis environments. While the methodology implemented within this thesis used Windows 7 and Capture-BAT exclusion lists, the overall methodology is applicable and adaptable to other versions of both environment and behaviour capture tools. It is important to follow the methodology precisely as a mistake here could lead to the undesirable outcome of a malicious behaviour being wrongly flagged as benign.

1. The development of behaviour filters should be done on the same environment that would be used to analyse potentially malicious webpages (or binaries). This is justified as it is likely that different versions of operating systems may behave significantly differently which could lead to potentially malicious behaviour being wrongly flagged as non-malicious.
2. Run either the latest version of O.S., browser and applications in the run-time analysis client or a highly vulnerable version of the O.S., browser and applications if the goal is to capture a large amount of malware. Once updated disable all auto-update functions and set hard drive to immutable mode to prevent changes.

3. Determine O.S., browser, and application activities by running the run-time analysis software on start-up and learning regular system changes. Unknown changes should be investigated thoroughly before being included in the behavioural filter. Investigations can be carried out using a software vendor's manual or frequently asked question where a given behaviour captured by the run-time analysis client is verified with the expectations of the given application or software.
4. Follow the steps for choosing benign webpages described in Figure 4.3 below. At least 1 low-interaction client honeypot in addition to a URL scanning service such as VirusTotal.com or wepawet.iseclab.org. The reasoning behind this is purely to have high certainty that the benign webpages are actually non-malicious or don't have links to malicious webpages in their advertisement streams. It's promising to exclude websites that host external ads to avoid any possible exploits from poisoned advertisement streams as these may contain malicious behaviour and at this stage only sanitised behaviour should be included in the filter design stage.
5. Run a high number of tests on benign webpages and develop exclusion lists based on constant and regular O.S., browser, application behaviour. While in this context a desirable definition of a high number would be as much as possible, a practical number of benign websites that can be ran during the development of an exclusion list is suggested in the low thousands.
6. Investigate the potential for malware in behaviours. Some parts of the operating system such the start-up sections of the registry or file system are more susceptible to attacks as scripts there would trigger at next re-boot.

Unless a behaviour is constantly observed (in a clean system) interacting with these high-risk file or registry paths, it should be excluded from the filter.

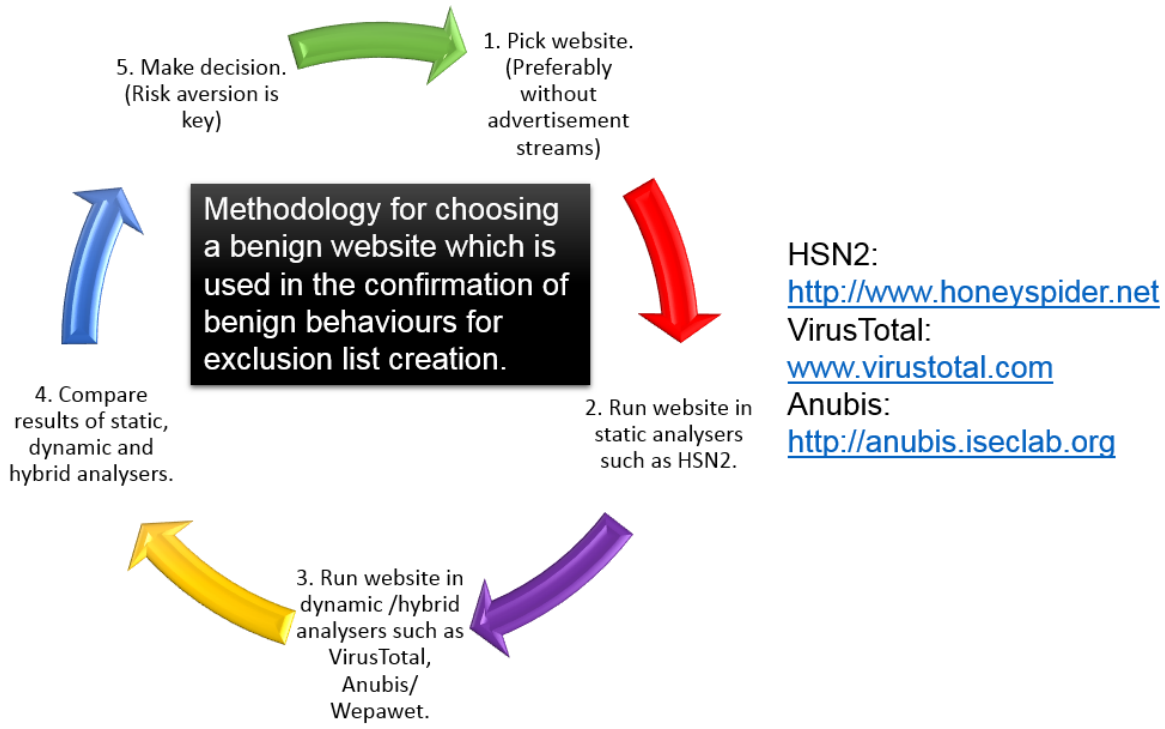


Figure 4.3: Methodology used for picking benign websites to identify benign behaviour.

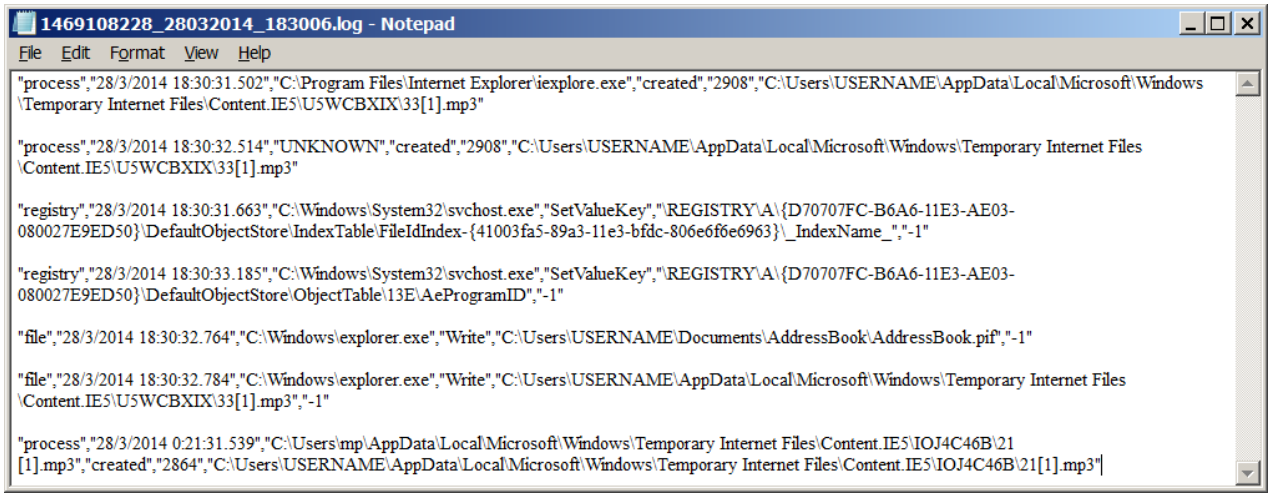


Figure 4.4: example Capture-BAT log file which includes malicious behaviour (drive-by-download being saved to temp Internet Files folder).

4.3.1 Decision making process behind classifications of behaviour

In order or the expert driven system to be a success, there is a need to understand behaviours before flagging them as benign. As a rule of thumb, it is important to ensure that any behaviour that is either not well documented or has limited information should not be marked as benign. Murdoch (2016) discusses the steps within intrusion detection systems. These steps can be applied to the design of expert lead behavioural filters. Within this work a number of resources are used to verify the actual behavioural entries. There are three main steps which are followed: Firstly, as discussed in Section 4.3, there is a need to develop and build a benign behaviour analysis environment and analyse confirmed benign webpages. This is something that is often seen within intrusion detection systems as initially a system is observed and scanned to determine regular and expected traffic. The same method is applied to the design of the expert lead behavioural filter.

Secondly, when a behaviour is observed, the log file should be inspected initially to determine if the behaviour was triggered by another, potentially malicious process. Here, the analyst should analyse the log file and look for any signs that suggest code injection has taken place. The signs can sometimes be obvious and a regularly seen occurrence can be to analyse browser interactions with a given particular process. An example may be: did iexplore.exe open another browser instance? If so, is there any reason for this to have taken place? Inspecting previous behaviour in a log file can often answer these questions which help suggest whether or not an interaction has indicators of maliciousness.

Thirdly, a range of resources are used to identify ‘expected’ behaviour within an operating system. Windows internals book, the Operating System help pages called the Technet.microsoft.com are excellent resources to look up specific processes and

specific files. Within these documentations, expected functions are often defined and mapping these within behaviour observed in a behaviour analysis environment can often provide solid evidence in the identification of benign behaviours. These are covered in depth within section 4.4.2-4.4.4 within the thesis.

4.3.2 Processing behavioural log files to synthesise expert driven behavioural filters

Log files gathered using the methodology above were processed using a series of steps. These were used in the creation of the expert lead system. These steps in processing the log files are included to allow for reproducibility:

1. Initially all behavioural log file data are combined for the behaviour understanding purpose.
2. An excel document was used, using comma-separated variables to break down behavioural entries.
3. As part of data transformation: metadata and time-tags from the raw log files are removed.
4. Behavioural entries are then created from the remaining fields. These fields include of triggering process, the behaviour type and the affected file/registry/process.

An example of this is provided:

Table 4.1a. Example of how a single behaviour changes after processing

Before processing	After Processing
"registry","18/8/2016 17:33:8.948","C:\Windows\System32\lsass.e	"C:\Windows\System32\lsass.exeSet ValueKeyHKU\.DEFAULT\Software\

xe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"	Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList"

It is possible to reduce the volume of data that needs to be analysed by grouping identical behavioural entities in log files. Behavioural entries that are fully identical with the only exception being the time it was triggered in the analysis process are essentially the same behaviour triggered at different time. By grouping these behaviours it's possible to significantly reduce the log file data that needs to be processed in the learning behaviour stage. In practice, actual reduction observed as part of the experiment shows a significant reduction in the number of behaviours that need to be analysed to create expert lead behavioural systems. Stats from an experiment in section 5.1 of Chapter 5 show this:

- In 3,011 log files it was observed that there were 696,000 total behaviours which can be reduced to 3,173 entries if they were grouped into unique behaviours. This mass reduction shows that a significant number of behavioural entries in the log files are re-occurring system events that are generated at different times and once a particular behaviour is studied.
- In a sample of 112 log files, 17,018 total behavioural entries were gathered. Out of this there were only 209 unique behaviours.

Analysing unique only behaviours for the development of an expert lead system is a viable approach which does not exclude any behaviours. To create the unique lists, an identical string matching function is used; if all strings in a behavioural entry

matches another fully; then they are grouped as a unique behaviour and a +1 count is assigned in the log file datasets.

4.4 Inside an exclusion list

As mentioned above, the key part of Capture-BAT is exclusion lists: these exclusion lists allow Capture to filter whether a behaviour is benign or malicious. Unexpected state changes are recorded in Capture log files (Qassrawi & Zhang, 2010). In total Capture-HPC provided 3 exclusion lists for Windows XP: ProcessMonitor, RegistryMonitor, and FilesMonitor (Honeynet Project Polish Chapter, 2012). As shown below in Figure 4.0b, an exclusion list can be quite complex looking.

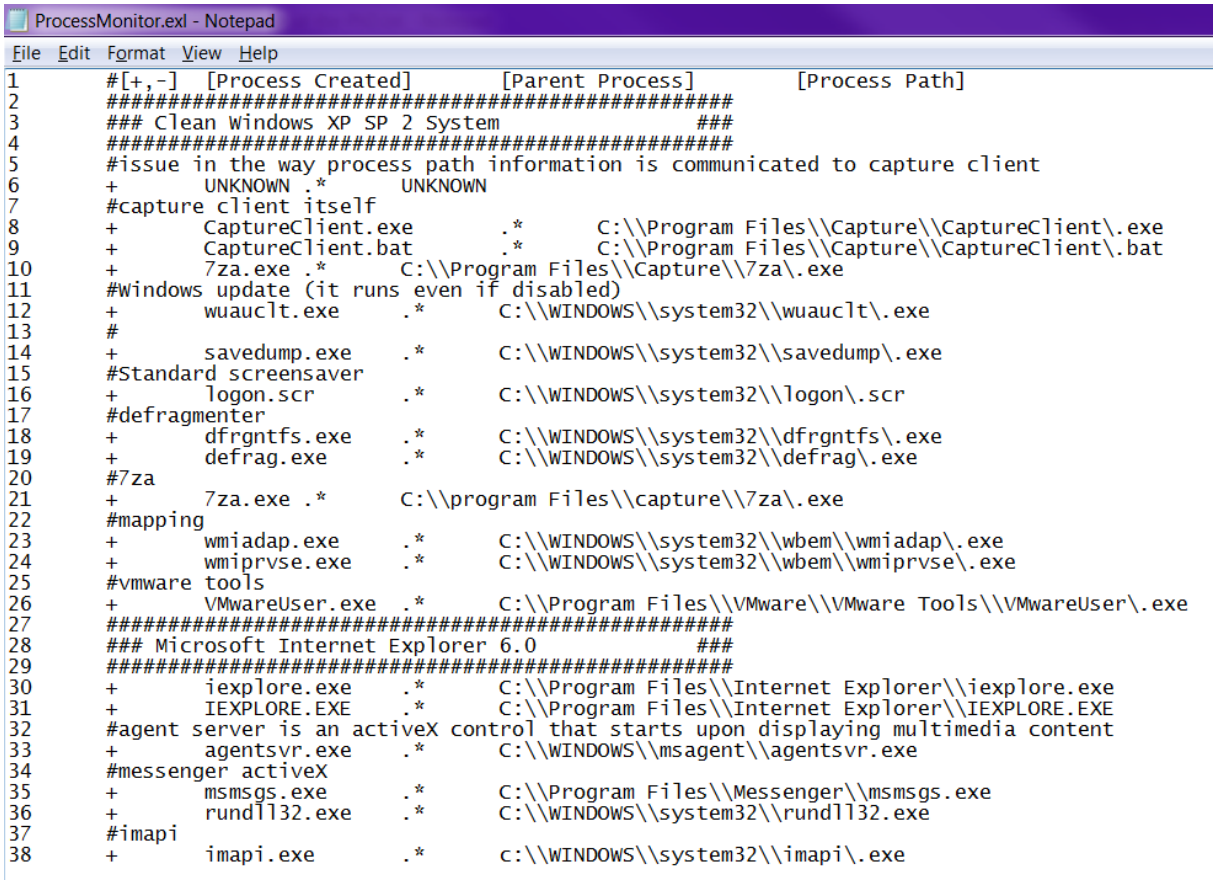


Figure 4.0b: Example 'stock' process monitor exclusion list for Windows XP SP2, IE6.

Capture exclusion lists can be configured to either:

1. Exclude certain events in the log files: in which case, the events will be benign and not flag the log file analysis as malicious. This is noted as a ‘+’ in the exclusion list in Figure 4.0b.

2. Include certain events in the log files which will be classified as malicious behaviour. This is noted as a ‘-‘ in the exclusion list in Figure 4.0b. If there are conflicting behaviour within and exclusion list, the inclusion of a behaviour would take priority over the exclusion.

There are 3 parts to each exclusion list: O.S., browser and applications behaviours. Each part of the exclusion list directs events which are either included or excluded. Exemplification is revealed in Table 4.1b and figure 4.0b: the first event (which is part of the O.S. activities) allows wuauclt.exe (Windows updater process) to write log files in C:\windows\WindowsUpdate.log. The second and third entries shown in the table is saying if any (:*) process creates any file name with the file extension .bat or .exe flag this behaviour as malicious. Full exclusion lists used are available in section B of the Appendix.

Table 4.1b. Snippet example of FileMonitor exclusion list. The full exclusion list available in Appendix B.

+/-	Access	Process Name	File Path
+	Write	C:\\WINDOWS\\system32\\wuauclt\.exe	C:\\WINDOWS\\WindowsUpdate\.log
-	Write	:*	.+\.bat
-	Write	:*	.+\.exe

Capture-HPC is shipped with default exclusions lists for Windows XP. It is important to note that these exclusion lists were created at the time with the release of Capture-HPC (2007), this means the default exclusion lists take into account the specific operating system, browser as well as applications installed on the system that was used in the creation of these exclusion lists. Any change in version would likely require amendments to the exclusion list as there would be different application, O.S. and web browser behaviours logged by Capture. Therefore, using unaltered or default exclusion lists to tailor Capture client's O.S. version, browser version and application specific versions would result in a number of false positives and negatives.

In order to develop accurate exclusion lists, a fairly high number of 'benign' tests should be run on Capture-BAT to identify benign behaviour. It is important to remember that significant tests should be carried out on the chosen benign websites to certify the non-malicious nature requirement as false positives here would mean allowing malicious behaviour to be classified as benign when Capture-BAT attempts to analyse malicious webpages. It is recommended that to keep Capture-BAT clients within a production environment as up-to-date as possible (assuming the other clients on the network are also up-to-date) to reflect good detectability of current malicious activities despite not having updated components of the Operating System to maximise capturing as many Windows 7 potential attacks. However, if the purpose of the analysis is simply to capture as many malware samples as possible, it is recommended to use a chosen O.S., browser and applications that is out of date with unpatched security vulnerabilities. Clearly, changing the environment requires testing of system behaviour and re-updating Capture-BAT exclusion lists post

updating or the installation of a new O.S., browser and applications running on a Capture-BAT client.

4.4.1 Typical Windows 7 native system calls

The adaptation to a new O.S., Windows 7 from Windows XP introduces the challenges of new file structure and newly created behaviours for new and existing interactions within the environment. For example: according to Microsoft Documentation (2016), the %SystemDrive%\ProgramData folder Windows 7 is used for application data that is not limited to user specific. In the older version of Windows, Windows XP: this path was %SystemDrive%\Documents and Settings\All Users\Application Data. This change in folder location however meant that when the task of system behaviour analysis was required, a number of entries would be observed that show the behavioural interaction of the file system using the ProgramData folder to save and delete files. To elaborate further: we run Capture-BAT and look at some benign system behaviours. Table 4.0c below are snippets of the resulting behavioural log file. In the figure, we can see that C:\Windows\System32\svchost.exe (which is the default location for a clean Svchost.exe) is writing a log file in the C:\ProgramData\Microsoft\Windows Defender\Support\MPLog-07132009-211939.log support folder. Similarly, in the log file, the process C:\Windows\System32\sppsvc.exe, is writing a log file in C:\Windows\System32\config\SYSTEM.LOG1. Sppsvc.exe is a Windows service that enables download, installation and enforcement of digital licences (Bleeping computer, 2016). It is evident that the writing of log files by an executable that is in its default Windows 7 location, within a clean, unmodified, unused test environment and has an unmodified binary file size is highly unlikely to be malicious. Some of these ‘idle’ system behaviours did not exist in previous versions of Windows and since

these now form part of the non-malicious behaviours there is a requirement to add these in a created exclusion list, for the Windows 7 environment.

file,"20/8/2016 20:42:26.115","C:\Windows\System32\svchost.exe","Write","C:\ProgramData\Microsoft\Windows Defender\Support\MPLog-07132009-211939.log","-1"
file,"20/8/2016 20:42:26.125","C:\Windows\System32\svchost.exe","Write","C:\ProgramData\Microsoft\Windows Defender\Support\MPLog-07132009-211939.log","-1"
file,"20/8/2016 20:42:26.125","C:\Windows\System32\svchost.exe","Write","C:\ProgramData\Microsoft\Windows Defender\Support\MPLog-07132009-211939.log","-1"
file,"20/8/2016 20:42:26.125","C:\Windows\System32\svchost.exe","Write","C:\ProgramData\Microsoft\Windows Defender\Support\MPLog-07132009-211939.log","-1"

Table 4.0c: A benign Windows 7 behavioural activity log, without exclusion lists and without malicious behaviours and interactions. Only a snippet of the log file is shown.

Note: System behaviours are explored in section 4.4.2-4.4.4.

Live system behaviours will now be explored based on behaviour type (file system, process or registry) in consideration with the creation of behaviour lists. System behaviours will be classified and colour coded as either Benign, Malicious or ‘Grey’ which essentially means inconsistent or unknown behaviours. Whilst the intention of

this work is intended to assist behavioural analysis, it is possible that knowledge on behaviours provided may lead to new target vectors from malware based on what malware writers may observe within our exclusion lists. Therefore, it is imperative that behavioural exclusions are reviewed on a system and as least as possible generalisations are used within the creation of behavioural exclusion lists to minimise the amount of exclusions whilst not allowing for potential malware to exploit generalisations in exclusions to bypass detection. For example, if a given proves such as wuauclt.exe is observed to write .log files in the C:\WINDOWS\WindowsUpdate.log directory, the written exclusion should reflect all aspects of this knowledge:

1. Only wuauclt.exe from the default system directory %SystemDrive%:\WINDOWS\system32\wuauclt.exe should be allowed to write.
    - Default path for system processes matter: From our data set, it is evident that malicious behaviours tend to run similarly named executable files from different directories (often the user or temp folder directory).

      Note: default pathway list was created and is available in the Appendix Section B.

2. Only .log files should be excluded from behaviour filtering if they are written in the C:\WINDOWS\WindowsUpdate.log directory.
    - Wildcards such as .* in Capture-BAT should not be used in this instance as we have only observed .log files being written. By using a wild card, we would potentially allow other file types to be written and

not be prompted about them. The intention of a successful exclusion list is to minimise known benign behaviours and not minimise irregular and unknown behaviours.

It is important to note that the behaviours listed in the next three sections assume that a system is being run and left idle with the only action taking place being the web browser opening a webpage. System behaviours that use the user folder would have slight variations, for example: the user we used Within Windows 7 is ‘mp’ – but within behaviour lists this user name can be modified as required.

4.4.2 System behaviours: The Windows 7 file system behaviours

The file system within a given environment refers to the way that data is stored, modified or deleted. This looks at the flows of data within a storage device (typically but not limited to a hard drive). File systems form part of the core storage of data used by an operating system: examples include storage of drivers for hardware or core data required for execution at the boot-time of an operating system Wirzenius et al. (2000). From a cyber security perspective, the behaviours of a file system refer to how a given process is modifying (writing or deleting) a given file which is stored in a directory (e.g.:C:/Windows means the Windows folder on the C:/ drive). This file system definition will be the definition intended within the thesis. Perceptibly a file system monitor forms part of the core system used in the behavioural monitoring of a behaviour analysis lab as the interactions between process and file can indicate the maliciousness of the given task. Within Windows 7, interactions contained by the file system proves to be one of the most active behavioural component in terms of generating idle and regular system behaviours.

Section 4.4.2, 4.4.3 and 4.4.4 are written mostly in tabular form displaying behaviour and providing behavioural evaluation. These behavioural tables can be used in the creation of behavioural filters or can be used in behavioural identification for classification if the intended analysis environment is Windows 7.

The provided behavioural analysis tables below are colour coded:

- Green: Benign behavioural interaction between a given process and the affected directory and file.
- Grey/Black: Unknown or inconsistent behaviour that can quite little is known about or behaviour that is not often triggered but in every case, does not inherently display maliciousness.
- Red: Typical confirmed malicious behaviour that should always be flagged up in analysis log file.

Keys

- W= Write
- D = Delete
- .*= All
- System = The System Kernel

Table 4.2: Typical benign Windows 7 File system behavioural interactions.

	Process	Affected directories and file types	Benign behaviour overview
W D	Capturecl ient.exe 7za.exe	1. C:\\program files\\capture\\.+ \.zip 2. C:\\program files\\capture\\log s	Capture creates log files in both .zip and .log format. These zips containing modified files are saved to the capture logs directory. The zip is created using 7za.exe (7 zip) which in the main capture folder which means the process in capture's directory will require both write and delete behaviour in an exclusion list.

		3. C:\\progra~1\\capture\\capture\.log 4. C:\\progra~1\\capture\\.+\.zip	Delete file access is also required as capture deletes any newly created log file after sending logs to the Capture-HPC Server. Therefore, if either the Captureclient.exe process or the f7a process from their known default install directories are writing these file types; this behaviour is benign.
W	Svchost.exe System	1. C:\\WINDOWS\\Prefetch\\.+	Prefetch in Windows 7 shares the same directory as Windows XP. SVChost.exe and System writing files to the Prefetch directory is a regularly observed benign system behaviour.
W	Svchost.exe Winlogon.exe	1. C:\\WINDOWS\\system32\\config\\.+\.LOG 1. C:\\WINDOWS\\Debug\\UserMode\\userenv\.log 2. C:\\Windows\\System32\\config\\SYSTEM.LOG1 3. C:\\Windows\\SoftwareDistribution\\DataStore\\Logs\\tmp.edb	Multiple processes write log files for Windows system. These are benign provided the executables are kept with integrity from a clean install of Windows and patching. Writing .log files is benign behaviours and can be included in exclusion list. Note: in comparison to Windows XP, Windows 7 specific log files are not stored in the documents and settings directory but instead. If previous versions of exclusion lists are being developed it is highly recommended to remove behavioural entries that are no longer required to avoid the possibility of filtering behaviours that should no longer appear.
	Wmiadap.exe Svchost.exe System	1. C:\\WINDOWS\\Prefetch\\.+ 2. C:\\WINDOWS\\system32\\Perf.* 3. C:\\WINDOWS\\system32\\wbem\\Performance\\.+	Performance information for Windows Management Instrumentation (WMI). Benign as long as process's that write it originate from System32 directory. These behaviours can be observed frequently when an O.S. is idle.
W	Svchost.exe	1. C:\\Windows\\System32\\winevt\\Logs\\Microsoft-Windows-HomeGroup Provider Service%4Operational\.evtx	HomeGroup features are Windows 7 native- this exclusion is required; it writes log files in the HomeGroup provider service directory.

W	lsass.exe	1. C:\\WINDOWS\\system32\\config\\.+\.LOG 2. C:\\WINDOWS\\system32\\config\\SAM 3. C:\\WINDOWS\\system32\\config\\system 4. C:\\WINDOWS\\system32\\config\\SECURITY	lsass.exe was observed to perform multiple writes of log files in numerous directories. These are benign providing lsass.exe is from the system32 directory and the files being written are in .log and .edb format.
WD	Wuauclt.exe	1. C:\\WINDOWS\\SoftwareDistribution\\DataStore\\Logs\\.+ 2. C:\\WINDOWS\\WindowsUpdate\.log 3. C:\\WINDOWS\\SoftwareDistribution\\DataStore\\DataStore\.edb	It is observed that Windows update writes and deletes files in three main directories even when windows update is disabled. The majority of the files written are .log and .edb which are safe file types and are in line with other Windows logging formats.
W	Services.exe	1. C:\\WINDOWS\\system32\\config\\AppEvent\.Evt 2. C:\\WINDOWS\\system32\\config\\SysEvent\.Evt 3. C:\\WINDOWS\\system32\\config\\SecEvent\.Evt	System events are recorded by Services.exe. The file format that is observed is limited to .evt. These files store logs and errors on a system and are used in trouble shooting. Therefore, any behaviour from services.exe that is writing an .evt file in the listed directories are benign. These are frequently observed on a clean Windows 7 O.S..
W	SearchIndexer.exe	1. C:\\ProgramData\\Microsoft\\Search\\Data\\Applications\\Windows\\GatherLogs\\SystemIndex\\.	Search indexer.exe is Windows 7 native, it creates Index files created by search indexer to quickly locate files. The service was known as the index service in Windows XP. Additionally, it creates a large number of log files within the

		+\\.* 2. C:\\ProgramData\\Microsoft\\Search\\Data\Applications\\Windows\\Windows\.edb	ProgramData directory which is not existent in the predecessor, Windows XP.
D	Svchost.exe	1. C:\\Windows\\Prefetch\\ReadyBoot\\.+	Readyboot cache is deleted by svchost.
WD	iexplore.exe	1. C:\\Users\\mp\\AppData\\Roaming\\Microsoft\\Windows\\Cookies\\.+ 2. C:\\Users\\mp\\AppData\\Local\\Microsoft\\Windows\\History\History.IE5\\.+\\index\.dat	Windows 7 introduced IE 8 which had a number of file system changes from the Windows XP file system. The documents and settings folder was removed and instead browser files such as Internet cache, cookies, history were stored into the C:\\users\\NAMEOFUSER\\AppData folder. It is important to note that if the provided exclusion lists are being used for other Capture-BAT setups or behavioural setups; the username mp should be changed according to the current user that will run the browser. It's evident that saving and deleting cookies, history and cached data are usual and benign system behaviours and thus must be included in the exclusion lists for the relevant browsers. Exclusion list data for Firefox is available in Appendix B within the contributed exclusion lists.

Table 4.3: Observed grey/situational Windows 7 file system behavioural interactions.

	Process	**File Path(s) edits**	**Grey behaviour overview**
W	.*	c:\\\$mft c:\\\$mftmirr c:\\\$logfile c:\\\$volume c:\\\$directory c:\\\$AttrDef c:\\\$boot c:\\\$bitmap c:\\\$badclus	NTFS is the Microsoft's propriety file system, periodically meta data is written in the listed directories. The files written are not windows 32 binaries which mean they can be viewed as benign. This is however marked as grey behaviour as the use of .*means any process can write it and that potentially includes malware.

		c:\\\$quota c:\\\$upcase c:\\\$ReplaceAttribute2 c:\\\$convertto nonresident	
	Svchost.exe	C:\\ProgramData\\Microsoft\\Crypto\\RSA\\MachineKeys\\4943bf62402c78e35e6a41d057394ac7_0e17101e-bd8f-4859-ad29-a2b0529004a5	Potentially Svchost storing RSA keys – this behaviour occurred quite rarely within the Windows 7 Capture-BAT and therefore doesn't really need to be included in an exclusion list by default.

Table 4.4: Typical malicious Windows 7 file system behavioural interactions.

	Process	File Path(s) edits	Malicious behaviour overview
W	Any	Anywhere.exe	Any writing of executables from visiting a web server is malicious: capture does not consent to downloading and saving files automatically therefore any write of this type is against user consent (and likely knowledge) thus highly malicious.
W	Any	Anywhere.bat	Similar to .exe file, writing of .bat files from visiting a web server is malicious: capture does not consent to downloading and saving files automatically therefore any write of this type is against user consent (and likely knowledge) thus highly malicious. The .bat files can essentially provide a large number of functionality to an attacker and form part of a large number of known malware.
W	Any	Anywhere.cmd	The .cmd files can be thought of as the newer version of the old .bat format and is largely similar. Any interaction writing these files should be flagged up as those files have the potential of causing numerous malicious behaviour.

Note: A more extensive malicious file system behaviour analysis is undertaken in Chapter 6.

4.4.3 System behaviours: The Windows 7 processes' behaviours

A process is the executing (opening and running) of a program (series of instructions that a computer's central processing unit understands). These programs, often known as executable files or binary files are stored within the storage and file system. (The Linux information project, 2006). Process monitoring can be thought of as the monitoring of the intermediary bridge between a given behaviour as it is a process that performs a (file system behaviour) file write/delete or a (registry behaviour) Set Value Key/ Delete Value Key. Monitoring the execution of a process's path can indicate if an executed process is benign or malicious. If the process performs the execution of a new file written in a non-system reserved directory: it can be assumed safely to be malicious if not otherwise intended by the user. On the other hand, if the process file is written in a known default directory, it is an indication that the process is safe however this does not innately mean that the process is performing non-malicious behaviours as processes can be injected with malicious code.

It is an assumption that a clean and unmodified Windows is running on the behavioural process monitor environment. We used the default C:\ drive but knowledge from this table is transferrable to any %SystemDrive% utilised within any created behaviour analysis lab. Care must be taken upon setup to ensure core processes and application in default locations are not replaced with malicious applications or injected prior to the creation of the behavioural test lab. File size verification of processes can be done on some core Windows applications to add an additional step of certitude. Once core Capture-BAT components are installed, additional protection is taken such as making the virtual hard drive immutable. This in addition to the built-in Capture-HPC revert script would ensure that during

experimentation no binary files which are changed and kept that way. Windows 7 processes are now evaluated and classified between benign, inconsistent /grey. Due to the fairly static nature of process creation and termination, any processes that are executed and are not within the benign or grey list should be marked as malicious. It is reasonable to assume that the process is quite likely malicious and undoubtedly worth supplementary investigation due to the nature of processes and their ability to undertake a large number of file, registry or other process interaction which can be highly malicious. Russinovich & Solomon, (2009) discusses the internals and workings of the Windows operating system in great depth.

Table 4.5: Typical benign Windows 7 processes with their default pathways.

Process created	Process's Benign Path within Windows 7	Information & assessment
CaptureClient.exe	C:\\Program Files\\Capture\\CaptureClient\.exe	Capture-HPC's executable will be launched automatically for each analysis and can be ignored. Benign.
CaptureClient.bat	C:\\Program Files\\Capture\\CaptureClient\.bat	Capture-BAT's Windows batch file required. Benign.
7za.exe	C:\\Program Files\\Capture\\7za\.exe	7Zip software used by Capture-BAT to zip changed files and captured malware. Benign.
wuauclt.exe	C:\\WINDOWS\\system32\\wuauclt\.exe	Windows update auto update client. This runs even if Windows update is disabled. Benign.
savedump.exe	C:\\WINDOWS\\system32\\savedump\.exe	Standard process that appears if windows is not shut down properly or if crashes happens. Benign.
logon.scr	C:\\WINDOWS\\system32\\logon\.scr	Process that provides screensaver. Benign.
dfrgntfs.exe	C:\\WINDOWS\\system32\\dfrgntfs\.exe	Defragmenter Processes. Runs when idle Benign.
defrag.exe	C:\\WINDOWS\\system32\\defrag\.exe	Defragmenter Processes. Benign.
wmiadap.exe	C:\\WINDOWS\\system32\\wbem\\wmiadap\.exe	Updates performance information within Windows Management Instrumentation (WMI) directory. Benign.
wmiprvse.exe	C:\\WINDOWS\\system32\\wbem\\wmiprvse\.exe	Windows management instrumentation undergoes process monitoring applications that alert users when important incidents occur. Benign.
VMwareUser.exe	C:\\Program Files\\VMware\\VMware Tools\\VMwareUser\.exe	VMWare tools bundle: used to increase performance and interaction from VM to host machines. Non-essential process – can be removed. Benign.
svchost.exe	C:\\Windows\\System32\\svchost\.exe	Critical system process that host a number of other services and processes. Benign.
dllhost.exe	C:\\Windows\\System32\\dllhost\.exe	Stock process for every Windows version. Performs a number of services for COM: provides object oriented programming architecture and runs registry event that handles system requests. Benign.

taskhost.exe	C:\\Windows\\System32\\taskhost\.exe	Acts as a host for all DLL services that Windows is running. (Similar to SVCHOST.exe)
taskeng.exe	C:\\Windows\\System32\\taskeng\.exe	Windows 7 task scheduler: Benign but if executed by malware it's likely to add malicious events in the scheduler for later execution.
SearchProtocolHost.exe	C:\\Windows\\System32\\SearchProtocolHost\.exe	Windows 7 Search processes, performs indexing for faster searches within file system.
SearchFilterHost.exe	C:\\Windows\\System32\\SearchFilterHost\.exe	Windows 7 process: contacts Microsoft for network activity and performs data download. Can be disabled to exclude.
SearchIndexer.exe	C:\\Windows\\System32\\SearchIndexer\.exe	Windows 7 Search processes, performs indexing for faster searches within file system.
winlogon.exe	C:\\Windows\\System32\\winlogon\.exe	This core process performs interactive user login and logout features.
userinit.exe	C:\\Windows\\System32\\userinit\.exe	Specifies the processes that auto run when a given user logs on. Establishes network connection.
csrss.exe	C:\\Windows\\System32\\csrss\.exe	Client Server Runtime Process that performs critical tasks, required as long as directory is as follows and only one instance observed. According to Russinovich & Solomon, (2009) this loads three Dynamic-link libraries (DLLs) that support the creating and deleting of processes and threads.
conhost.exe	C:\\Windows\\System32\\conhost\.exe	Fixes theme bugs introduced in Windows Vista. Safe as long as running within system 32 folder.
mobsync.exe	C:\\Windows\\System32\\mobsync\.exe	Performs syncing, present if syncing feature enabled in windows.
SPPSVC.exe	C:\\Windows\\System32\\sppsvc\.exe	This process hosts a number of services of windows and is core process for windows such as licencing verification.
wsqmcons.exe	C:\\Windows\\System32\\wsqmcons\.exe	Windows customer improvement process- sends usage data to Microsoft daily if enabled. Can be disabled.
Sdclt.exe	C:\\Windows\\System32\\sdclt\.exe	Windows backup system process.
sc.exe	C:\\Windows\\System32\\sc\.exe	Retrieves information and set values with other windows services
drvinst.exe	C:\\Windows\\System32\\drvinst\.exe	Used in Highpoint RAID Management Service and software.

WerFault.exe	C:\\Windows\\System32\\WerFault\.exe	Windows error reporting service – which allows users to send reports to Microsoft based on faced errors. This checks contents of the AeDebug registry key.
Iexplorer.exe	C:\\Program Files\\Internet Explorer\\iexplore.exe	Internet explorer web browser; as this was our chosen browser launching this application from its default directory is benign.
agentsvr.exe	C:\\WINDOWS\\msagent\\agentsvr.exe	ActiveX control that starts upon displaying multimedia content. Required for browsers.
rundll32.exe	C:\\WINDOWS\\system32\\rundll32.exe	Runs DLL files and places them in libraries within Windows.
imapi.exe	c:\\WINDOWS\\system32\\imapi\.exe	Image API used to provide Image burning services in windows.

Table 4.6: Typical grey / situational Windows 7 processes with their default pathways.

Process created	Process's Path	Information & assessment
services.exe	C:\Windows\System32\services\.exe	Decision to not include this in exclusion list can be justified: this process was inconsistent in its appearance within our environment. Furthermore, this process was triggered by 'UNKNOWN' within our data set which suggests it's call operators are not required upon every boot up and website analysis. The process itself is a legit process however as it's a Windows process and as long as it's executing from the System32 directory, it's likely to not have been tampered with. It is known from the file monitor that Services.exe writes event log files which is a benign behaviour. The inconsistency however puts this process on the grey list.
firefox.exe	C:\\Program Files\\Mozilla Firefox\\firefox.exe	Firefox browser - use as exclusion list if Firefox is desired over Internet Explorer. Firefox or any other browsers should not self-execute however if the chosen browser is IE upon visitation of malicious webpage.
GoogleUpdate.exe	C:\\Program Files\\Google\\Update\\GoogleUpdate\.exe	Google update service if any google service running. Might need to add as a UNKNOWN if executed by google service. This is a grey behaviour as it will be present on machines which have applications running this service. This should not be in an exclusion list if there

		are no Google services (such as Google Chrome) running on the machine.

Table 4.7: Typical malicious Windows 7 processes based on real attacks in our dataset.

Process initiating the creation	Process's path	Information & assessment
C:\Program Files\Internet Explorer\iexplore.exe	C:\Users\mp\AppData\Local\Temp\Iexplore.exe	Internet explorer's process is not in the user temp folder (it's in the program files directory), this means another application is attempting to disguise itself as Internet Explorer after having compromised the actual browser. This is an example of a benign process executing a malicious process.
C:\Users\mp\AppData\Local\Temp\svchost.exe	C:\Program Files\Microsoft\DesktopLayer.exe	Both these executables are malicious and not present within a clean Windows 7 O.S.. This is a real example of a malicious process executing another malicious process based on one of the findings within the dataset. Any executables that are executing other executables which are not regular system behaviour or use the default pathway given at installation should be flagged up as malicious.
C:\Windows\System32\wscript.exe	C:\Windows\System32\cmd.exe	This may seem as a regular system behaviour as a perfectly benign cmd.exe is being executed by wscript.exe with both process's having their default parent directory. However, this behaviour is seen once: anomalies especially within

		process monitoring should be investigated. Upon investigation we concluded that the (a known Visual Basic Script) attack was exploiting Wscript.exe to launch scripts that wrote malicious executables in other directories which self-executed.

Note: A more extensive malicious process analysis is undertaken in Chapter 6.

4.4.4 System behaviours: The Windows 7 registry behaviours

The registry can be thought of as a large database that saves a large number of settings, information, values and keys for the Windows operating system as well as the applications that are installed on the machine. Khanse (2011) states that the registry is the centralised configuration database for the Windows operating system and for the applications installed. This contains a set of files namely hives which makes the registry and are stored in the Windows\System 32\config directory. Registry keys can be entered and deleted and these form part of the registry monitor aspect of behavioural analysis. A registry key set is typically done by a process on a system: much like the file system it is process that writes a file, or in the case of the registry sets or deletes a value in the registry database. Unlike the file system however, the registry behaviours can sometimes be relatively difficult to classify as benign, malicious or grey. In order to identify malicious behaviour, firstly a malware analyst would have to check the pathway of the process writing values: if the process seems to be from the default location within the O.S. then that would signify a benign change.

Secondly when attempting to classify behaviours maliciously: some known sections of the registry allow for a greater level of exploitation than others and therefore areas that are known to offer a greater level of control upon being compromised should be

approached with care. The sections that would control what start when a system reboots for instance can allow malware to add malicious scripts which would execute upon boot which is highly malicious behaviour. Thirdly, registry behaviours that are incredibly rare within a given dataset and show little to no similarity to known behaviours of the same nature should by default be marked malicious or grey. The reasoning here is that a malware analyst attempting to understand registry behaviours is more likely to benefit from a curious and risk-averse approach in order to allow for more occasions for the malware to manifest within the system.

Moreover, registry behaviours tend to have a level of ambiguity within them as looking up what each section of the registry is for what purpose can sometimes yield more questions than answers. This is due to the fact that not every process and registry interaction is documented by the O.S.. Consequently, it is recommended that a large amount of testing and observation is required on a wide range of URLs to attempt to classify the unknown into possible malicious or possible benign. In terms of using Capture-BAT, one limitation faced is that the behaviour analysing tool was unable to pick up the value of the key that was added or removed by a given process. Provided in the table below are some evaluations of registry behaviours. The full lists have been provided in the form of Capture-BAT exclusion list in the Appendix section B. In the tables created below, only the process name is given to improve readability, the genuine default path for a given process has been explored in the previous section and can additionally be viewed in section 4 of the Appendix in Table 4E: ‘Process and default path table’.

One of the limitations of Capture-BAT is its inability to pick the value of the changed key. This means that some behaviours would be more difficult to classify as the values can indicate whether a particular set value behaviour is malicious or not.

However, by analysing the behaviours from visiting benign webpages, it is also possible to identify acceptable, normal and benign system behaviours which show no maliciousness. This in turn would allow an analyst to identify irregular and potentially malicious behaviour and overcome the limitations of Capture-BAT's registry monitor.

Keys:

Set = SetValueKey behaviour

Del=DeleteValueKey behaviour.

.+ = Wildcard: any file name.

Table 4.8: Examples of benign registry behaviours that can be filtered in behavioural exclusion lists.

	Process	Registry Path(s) edits	Behaviour evaluation
Set	Iexplore.exe	1. HKCR\\Local Settings\\MuiCache\ .+\.+\\LanguageList 2. HKCU\\Software\\Microsoft\\Internet Explorer\\DOMStorage\\Total 3. HKCU\\Software\\Microsoft\\Internet Explorer\\Main\\CompatibilityFlags 4. KCU\\Software\\Macromedia\\FlashPlayer\\FlashPlayerVersion 5. HKCU\\Software\\Microsoft\\Internet Explorer\\LinksBar\\ItemCache\\.+	Our chosen browser, Internet Explorer 8 was observed to have a large number of interactions within the registry ranging from system interactions such as setting language, new DOM cookies and if an application has recently used Direct3D, the key of that app will be included in the Direct3D section. Internet explorer also regularly interacts with the installed version of Flash Player. Internet Explorer 8 also saved recently searched words and addressed in the ItemCache directory of the registry.

Set	mobsync.exe	1. HKCR\\Local Settings\\Software\\ Microsoft\\Windows	Mobsync is a windows process to keep components updated and requires writes keys within the registry pathway. This was frequently observed within the Windows 7 environment whilst the system was relatively idle as visiting benign webpages was the only running task.
Set + Del	System	1. HKLM\\SYSTEM\\ControlSet001\\Enum\\UMB\\UMB\\1&841921d&0&PrinterBusEnumerator\\.+ 2. HKLM\\SYSTEM\\ControlSet001\\Enum\\UMB\\UMB\\1&841921d&0&PrinterBusEnumerator\\LogConf\\.+ 3. HKLM\\SYSTEM\\ControlSet001\\Enum\\UMB\\UMB\\1&841921d&0&PrinterBusEnumerator\\Device 4. HKLM\\SYSTEM\\ControlSet001\\Enum\\UMB\\UMB\\1&841921d&0&PrinterBusEnumerator\\Capabilities 5. HKLM\\SYSTEM\\ControlSet001\\Enum\\UMB\\UMB\\1&841921d&0&PrinterBusEnumerator\\UINumber 6. HKLM\\SYSTEM\\ControlSet001\\Enum\\UMB\\UMB\\1&841921d&0&PrinterBusEnumerator\\HardwareID	This registry path is used to store information about devices on the system. IT was observed that the pathway contains a folder that sometimes gets changed into a different set of string. This is highlighted and it is possible to use a wildcard in behavioural lists to exclude that particular folder. Enum tree within the registry is used by the Windows O.S. to change drivers and device installation components. The full lists of these similar behaviour changes are far too long for this table but can be viewed in our created registry exclusion list situated in Appendix B. Wildcard examples are also used there due to the high volume of unique registry behavioural entries created over time.

Set + Del	Svchost.exe	1. HKCU\\Software\\Microsoft\\SystemCertificates\\ Root\\.+ 2. HKLM\\SYSTEM\\ControlSet001\\.+	Svchost.exe performs a wide range of operating system (benign) entries within the registry. Examples include managing (adding and deleting) certificates within the certificate store and the use of the ControlSet001 system configuration system containing information on services and device drivers (this is also done by services.exe).
Set	Explorer.exe	1. HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\HomeGroup\\UIStatusCache\\.+ 2. HKCR\\Local Settings\\MuiCache\\.+ 3. HKLM\\SYSTEM\\ControlSet001 4. \\services\\NlaSvc\\Parameters\\Internet\\ManualProxies	Windows 7's Explorer.exe adds values in the three registry pathways. MuiCache and HomeGroup are functions native to the newer Windows and not seen in Windows XP. As long as Explorer.exe has not been compromised prior to observing set value keys, there is no evidence of the potential for maliciousness within these behaviours.
Set	lsass.exe	1. HKLM\\SECURITY\\.+ 2. HKCU\\Software\\Microsoft\\Protected Storage System Provider\\.+ 3. HKU\\.DEFAULT\\Software\\Classes\\Local Settings\\MuiCache\\.+\\.+\\LanguageList.*	It is acceptable and benign for Isass.exe which is the process that enforces the security policy on Windows to set keys the security files within the registry that store information on administrative user. (Non admin users typically have this section of the registry empty). It is reasonable to assume that Protected storage system provider can be written by a genuine isass.exe process as Isass.exe manages security policies. MuiCache is used as storage for application names and their versions. It was observed that isass.exe performs multiple set value keys in the multiple MuiCache directories. (see

			Windows 7 registry exclusion list in Appendix B for full list). These occurred frequently and were subjected to a wide range of variation.
Set	Sppsvc.exe	1. HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\SoftwareProtectionPlatform\\ServiceSessionId 2. HKLM\\SYSTEM\\WPA\\.+	Benign process observed, in the second registry pathway, use of a wildcard is recommended as the folder after WPA inconsistently changes name variation.
Set	SearchProtocolHost.exe	1. HKU\\.DEFAULT\\Software\\Classes\Local Settings\\.*	Windows 7 native process SearchProtocolHost is observed to regularly set registry keys in the local settings folder. This does not seem risky or inherently malicious and therefore can be present in the exclusion list.
	SearchIndexer.exe	1. HKLM\\SOFTWARE\\Microsoft\\Windows Search\\Gather\\Windows\\SystemIndex\\.+ 2. HKLM\\SOFTWARE\\Microsoft\\Windows Search\\.*	Thiese behaviours are very frequent similar to SearchProtocolHost - Windows 7 Native and benign.

Table 4.9: Examples of registry grey, unknown and inconsistent behaviours that should within Windows 7.

Process	Registry Path(s) edits	Behaviour overview
spoolsv.exe	1. HKCU\\Software\\Microsoft\\Windows NT\\CurrentVersion\\Devices\\Microsoft XPS Document Writer 2. HKCU\\Software\\Microsoft\\Windows NT\\CurrentVersion\\PrinterPorts\\Microsoft XPS Document Writer 3. HKCU\\Software\\Microsoft\\Windows NT\\CurrentVersion\\Devices\\Fax 4. HKCU\\Software\\Microsoft\\Windows NT\\CurrentVersion\\PrinterPorts\\Fax 5. HKU\\S-1-5-19\\Printers\\Defaults\\NetID 6. HKU\\S-1-5-19\\Software\\Microsoft\\Windows NT\\CurrentVersion\\Windows\\UserSelectedDefault 7. HKLM\\SYSTEM\\ControlSet001\\Control\\Print\\Printers\\.+ 8. HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Print\\Printers\\.+ 9. HKU\\S-1-5-	This process is seen very frequently in Windows 7 and whilst it's genuine and relatively benign it is observed that these registry entries perform a large number of SetValueKey behaviours within the registry despite having no printer or fax connected or active within our analysis environment. The full list of registry pathways is available in the registry exclusion list within the Appendix. We chose to exclude this behaviour due to the large volume of data created in each log file. An example benign log file provided in parts of table 4.0a and the Appendix B show these volumes. It is questionable however as to why is there so many behavioural activities triggered within this section of the registry upon each boot when there is no printer and fax connected or in use.

	19\\Software\\Microsoft\\Windows NT\\CurrentVersion\\Devices\\.+	
CaptureClient.exe	1. HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\GlobalAssocChangedCounter	This behaviour occurred on our machine which runs capture. There is not much detail available on this particular behaviour. To make matters even more abstract, we observed this behaviour only 13 times in over 2 years of malware analysing, it was inconsistent with no apparent trigger. Whilst it does not seem malicious, the fact that it's rather rare prompts the non-inclusion in a filter.
Recordingmanager.exe	1. HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders\AppData 2. HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders\Common AppData 3. HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{9f9da841-2690-11e3-b262-806d6172696f}\BaseClass	The analysis machine contained real player however these behaviours with a large number of variants triggered way to inconsistently and incredibly rarely (twice for each registry pathway) which suggests anomalies. These behaviours should not manifest in machines which do not have real player.

Registry keys that manage auto run each time a user is logged on a client machine. (Run once are deleted before the next run time, run contains entries that will always start upon login).

Table 4.10: Examples of registry malicious behaviours that should be flagged up and investigated within Windows 7.

	Process	Registry Path(s) edits	Malicious behaviour overview
Set + Del	Any	1. HKEY_LOCAL_MACHINE\ Software\Microsoft\Windows\CurrentVersion\Run 2. HKEY_CURRENT_USER\ Software\Microsoft\Windows\CurrentVersion\Run 3. HKEY_LOCAL_MACHINE\ Software\Microsoft\Windows\CurrentVersion\RunOnce 4. HKEY_CURRENT_USER\ Software\Microsoft\Windows\CurrentVersion\RunOnce	Similar to Windows XP, these file paths within the Windows 7 registry allow for the addition of start-up scripts and programs. Within a behavioural environment these settings should have been initially set and whilst web browsing these settings should not be altered. Therefore, any changes to the Run and Run Once directories are likely to be malicious and require reporting.
Set + Del	Any process	1. hkcu//Software//Microsoft//Windows%20NT//CurrentVersion//Windows//userinit	Changes in the Userinit section are highly undesirable as the Userinit controls actions that occur upon the machine log-on. This could be exploited to allow malicious scripts to run each time a machine is logged on for instance and thus all changes to the userinit section, similar to the run sections need to be reported for investigation.
Set + Del	Any process	2. HLKM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Policies\\Explorer\\Run.* 3. HKCU\\Software\\Microsoft\\Windows\\CurrentVersion	Similar to the Run and RunOnce entries above, these control have the potential to exploit policy settings in order to allow malware to start as auto-

		\\Policies\\Explorer\\Run.*	run. Any changes here is likely to be malicious and should be reported and investigated.
Set + Del	Any process	1. HLKM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\ShellServiceObjectDelayLoad	Values in this are executed by the explorer.exe process upon start up. Changes here therefore has the possibility of undesired scripts and programs being called by explorer.exe which is exploitable.

Note: A more extensive malicious behaviour analysis for the registry is undertaken in Chapter 6.

4.4.5 Differences between Windows 7 with Windows XP system calls

Having explored a large number of benign behaviours and interactions that were observed on a Windows 7 environment, it is possible to explore differences in behaviour from Windows XP and Windows 7. This section will discuss and analyse the differences in behaviour between the two operating systems. This is an unexplored aspect of research within cyber security and may help security analysts and threat hunters in the decision making process of the behaviour analysis environment design stage. Redstone et al. (2000) has analysed operating system behaviour on a simultaneous multithreaded architecture but their work focuses on modified Unix operating systems and do not provide insights on how different operating systems provide similar or different system calls. Stange (2015) discusses the detection of malware across operating systems but do not provide an expert-lead insight into operating system behaviours.

The file system displayed the greatest amount of difference between Windows 7 and Windows XP. As discussed in section 4.4.1, Windows 7 (and above versions), introduced a few new directories. Firstly, the %SystemDrive%\ProgramData directory which can store data for any applications running within the system without the need

of administrative rights. In addition to the data stored by applications on the environment, Svchost.exe regularly write RSA keys in the ProgramData directory. Furthermore, SearchIndexer.exe saves a large amount of files ranging from .log, .gthr, .crwl, .chk and .edb in that same folder. The effect of these additions increases the amount of behavioural interactions for file writes and being benign they subsequently would need to form part of an exclusion list.

Comparatively within Windows XP, the application data was stored in %SystemDrive%\Documents and Settings\User'sName\Application Data\. This has an impact on a large number of behavioural writes and deletes: from the browser perspective, Internet Explorer (6) in Windows XP performs a large number of file writes and deletes for temporary files, history, cookies, user data, plugins, digital rights management which are stored within the 'Document and Settings directory'. Any changes to the file system in another Operating System (or browser) would expectedly cause a system to generate a large amount of (benign) behavioural entries. Moreover, the system kernel (Known as 'System' in log files and exclusion lists) writes .log files and user data (.dat) within the same folder periodically. In comparison to Windows XP, Windows 7 web browser data is stored and manipulated within the user folder: %SystemDrive%:\Users\USER'sNAME\AppData\Local\Microsoft\Windows\. This is also shared by other system services such as: CryptnetUrlCache meta data, content, temporary files and Windows defender logs. Naturally the inclusion of newer directories within Windows 7 resulted in even more behavioural interactions.

Correspondingly, Firefox's data within Windows 7 is stored in the %SystemDrive%:\Users\USER'sNAME\AppData\Roaming\Mozilla\Firefox\Profiles\ folder. Patently, the migration to the user folder caused file writes and deletions for

the benign browser activities to be migrated over to the new user directory from the old document and settings directory in Windows XP. The user folder used in Windows 7 and above, also hosts additional files for applications such as Macromedia flash player. Windows XP lacked the HomeGroup function that is offered by Windows 7 which results in a few behavioural file writes by a Windows 7 Svchost.exe in the user's home group classifier folder. In conclusion, the differences between the respective file system's default directories and the addition of multiple new storage folders causes a large amount of newer behavioural functions (file writes mostly but additionally a few file deletions) to be generated from a file system's perspective.

In terms of processes, Windows 7 had significantly more concurrently run processes. Exemplifications include: Windows 7 search processes (SearchProtocolHost.exe, SearchFilterHost.exe, SearchIndexer.exe) were being called from their respective default directories at idle. These indexing services immediately added at least three behavioural entries as they were 'created' upon boot up but should any of these process crash or are instructed to be closed by another running process, more behavioural interactions would be generated as the process was 'terminated' as labelled by Capture-BAT.

Within both Operating systems, similarity in behaviour could be observed if the process location was kept within the same default directory. Services that manage .dll files in Windows such as dllhost.exe and taskhost.exe are called and executed at runtime in the background. This creates a behavioural entry on both O.S. i.e.: (**taskhost.exe created C:\Windows\System 32\taskhost.exe**) Task scheduler (tasking.exe), user processes (winlogon.exe and userinit.exe) were also executed upon boot-up as required by the O.S.. Some Windows 7 processes such as sdclt.exe

(Window's 7 backup) and WerFault.exe (responsible for Windows error reporting) was observed to be triggering occasionally within Windows 7 and we did not observe any of these occasional behaviours in Windows XP. To conclude, Windows 7 was calling more processes from their default start up location in each investigation. Visibly so, processes tend to be a lot less frequent in volume than file writes and deletes as it is often a single process that undertakes a large number of interaction with the file or registry system. They can be thought of as the bridges between either a file system or registry behaviour. As a final point, the older and retired service, namely Windows messenger (%SystemDrive%:\Program Files\Messenger\msmsgs.exe) is no longer present by default in Windows 7 environments and it's write and delete behaviours should be removed or excluded from Windows 7 exclusion lists to avoid malware masquerading.

As explored in the registry's behaviours, MuiCache stores executable names and versions. However, within the Windows XP environment, MuiCache is stored under HKEY_CURRENT_USER\Software\Microsoft\Windows\ShellNoRoam\MUICache and within a Windows 7 environment (Vista onwards) the new registry directory is HKEY_CURRENT_USER\Software\Classes\LocalSettings\Software\Microsoft\Windows\Shell\MuiCache. Changes in registry locations from both XP and Windows 7 means that any registry 'Set Value Key' and 'Delete Value Key' behaviours require migrating. In terms of system behaviour analysis, it was observed that there were a lot more MuiCache behaviours in the Windows 7 environment as opposed to the Windows XP environment which suggests that newer systems would benefit from having the range of exclusions required to avoid unnecessary behaviours. The introduction of the HomeGroup function in Windows 7 did add the recurring behaviour that a clean explorer.exe was adding cache values to the registry. Other

native Windows 7 processes which are not present in Windows XP, such as SearchProtocolHost.exe, SearchIndexer.exe are observed to frequently perform set key behaviours in the registry. Due to the sheer occurrence of this behaviour and lack of inherent maliciousness, the decision was made to flag this as benign within Windows 7.

When compared the Windows 7's Internet Explorer 8.0 were showing a small number of newer behaviours which the Windows XP's Internet Explorer 6.0 registry browser activity were lacking. The introduction of MuiCache in the previous paragraph is one such difference. Furthermore, it was observed that mobsync.exe in Windows 7 environments was frequently performing Set Value Key behaviours within the SyncMgr directories within the registry. Document Object Model (DOM) storage was a new cookie storage feature in Internet Explorer 8 that was not present in the older version of Internet Explorer. Consequently, these behaviours would require further exclusions in a Windows 7 exclusion list that are inclusive mostly of setting value but there are additionally several deletions. It is important to note that a large proportion of the registry behaviours from the two versions of Internet Explorer were similar as the default executable location for the browser was identical in both cases, (C:\Program Files\Internet Explorer\iexplore.exe).

4.5 The development cycle of exclusion lists

In order to create fresh exclusion lists for a new O.S., we needed Capture-BAT running on the given client. This would allow us to run tests on the client and these tests provided full, unsorted and unclassified lists of system behaviours. It is important to note at this stage that while at the time of the research Capture-BAT seemed suited to create behaviour filters due to being free, widely available and well established. The methodology and creation process for exclusion lists can be applied

to the more recently endorsed and popular dynamic behaviour analysing tools that support drive-by-download behaviour analysis such as Cuckoo sandbox (https://www.cuckoosandbox.org/).

Figure 4.1 below shows the setup used to create the lab environment. The Capture-BAT clients were used to visit real benign websites which were thoroughly tested in line with the proposed benign website choice methodology (Figure 4.3) and in our paper, Puttaroo et al. (2014).

4.5.1 Environment set up

In the two instances of Capture operating on a university network, Capture-BAT on Windows 7 was executed using Firefox and IE as browsers for the virtual machines. The data currently obtained from Capture-BAT is divided between process, registry and file system as log files. The current limitations of the systems running include: no network API monitoring and the inability to detect and capture ActiveX exploits as we don't have the ActiveX components installed. The capture clients are set up to resemble client systems but by default have auto-updates on O.S., browser and applications turned off: this is to increase performance as the client systems would revert to their original state and be required to re-perform updates at every malicious webpage analysis tests. Figure 4.1 shows this setup.

Using different versions of the O.S. has several implications with regards to the data obtained by Capture-BAT: over time a number of new behaviours would found as existing applications and operating system patches are applied. This is logical as it is reasonable to expect that with new patches and versions of applications new features are added and existing behaviours are sometimes changed. Additionally, software occasionally releases security fixes to patch vulnerabilities in programs and plugins which would mean after patching a number of exploits may no longer be

observed or malicious manifestations may differ. In terms of gathering data from a honeypot, it is recommended that automated updates are disabled.

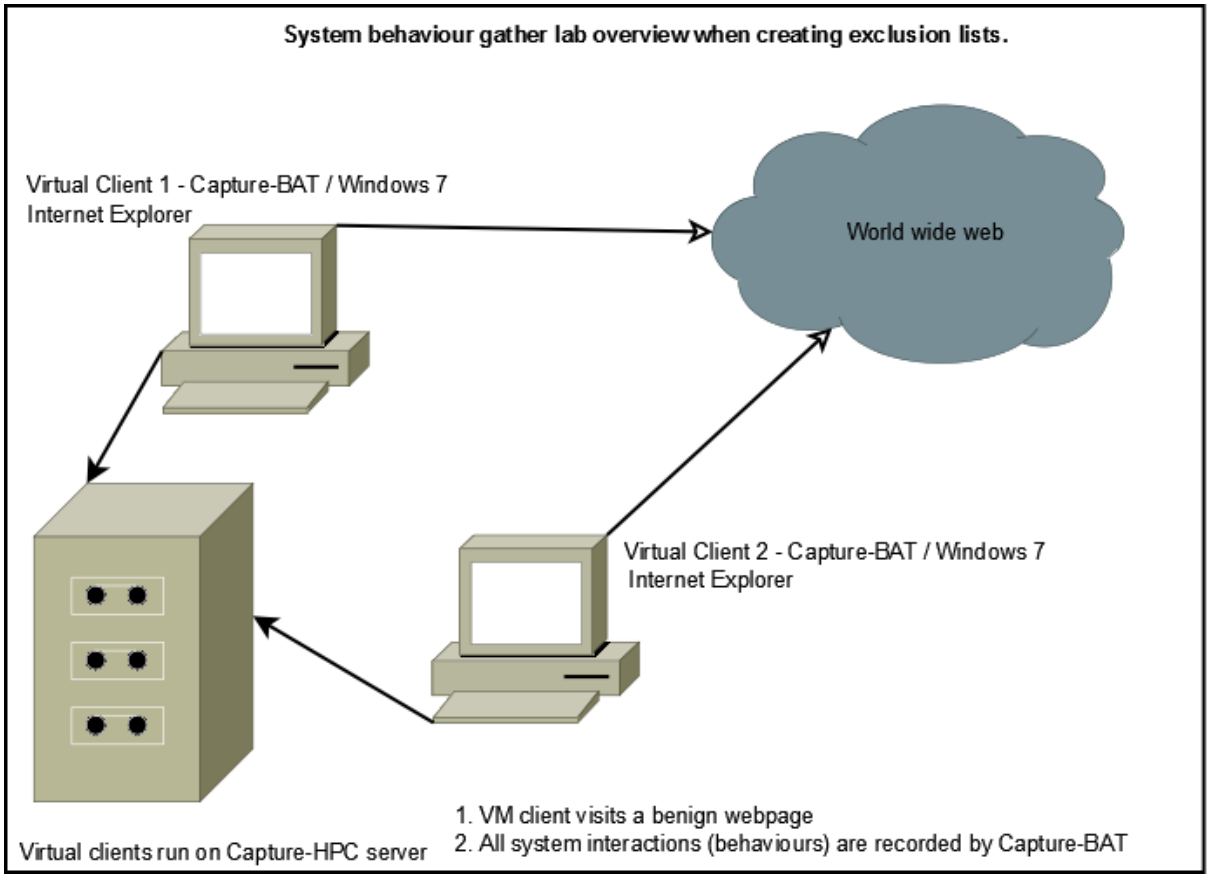


Figure 4.1: Capture-BAT experimental setup used to gather benign log files.

Another issue with not disabling auto-update on applications, browser and O.S. would be: Capture's log files might flag updates of software as malicious if not well defined in exclusion lists resulting in possible false negatives. This would mean the requirement for a larger number of entries in an exclusion list's application behaviour section which would require a long set-up time due to the nature of developing valid and accurate exclusion lists.

In terms of the hardware and O.S. setup that was used in the research, a Dell Precision T1600 with 16 GB ram, i7 six core processor. This would run the Capture-HPC server in Debian Squeeze. The Capture-BAT clients were run within

virtualisation, allocated 1 Ghz of processor power,1 GB ram (Oracle Virtual Box 7.0) and ran ‘clean’ copies of licenced Windows 7 ‘n edition’ Service Pack 1. In terms of Windows 7 updates installed in our Capture-BAT, Hotfix for windows (KB2534111) and update for Microsoft Windows (KB976902) was pre-installed on the system upon clean O.S. installation.

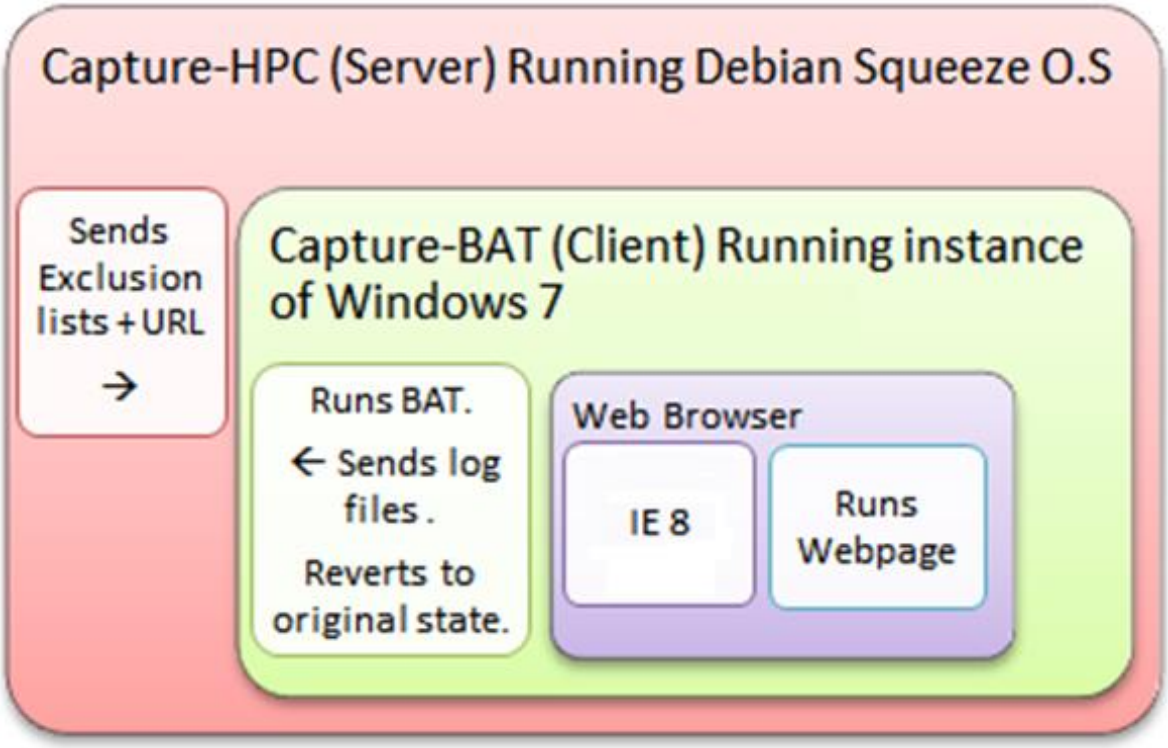


Figure 4.2a: Interactions between Capture-HPC server and Capture-BAT clients.

As shown in Figure 4.2a, the Capture-HPC server has the task of managing the capture clients: this involves a number of inputs and some automated outputs. Inputs would include a set of exclusion lists to be used by Capture-BAT clients when the Behaviour Analysis Tool (BAT) performs dynamic malware analysis, a set of URLs which are sent to Capture-HPC clients for analysis, as well as server controls such

as start, pause and stop. The server receives behavioural data in the form of log files and a copy of all modified or created files from capture BAT which is stored.

It is important to note that the version of Capture we are using is a modified version by HoneySpider Network called Capture-HPC_NG. There are a few differences to the original version of Capture most notably, the ability for VirtualBox and KVM to be used as opposed to VMWare. There were some (unmentioned) changes to logging format as well (Honeynet Project Polish Chapter, 2012).

Finally, In the interest of experiment replicability and malware experiment transparency: the utilised Capture-BAT settings are provided:

```
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="config.xsd">
        <global collect-modified-files="true"
                        client-default="iexplore"
                        client-default-visit-time="90"
                        capture-network-packets-malicious="false"
                        capture-network-packets-benign="false"
                        revert-after-each-url="true"
                        send-exclusion-lists="true"
                        terminate="false"
                        group_size="1"
                        vm_stalled_after_revert_timeout="120"
            revert_timeout="150"
            client_inactivity_timeout="200"
            vm_stalled_during_operation_timeout="200"
            same_vm_revert_delay="80"
            different_vm_revert_delay="120"
        />

        <exclusion-list monitor="file" file="FileMonitor.exl" />
        <exclusion-list monitor="process" file="ProcessMonitor.exl" />
        <exclusion-list monitor="registry" file="RegistryMonitor.exl" />
```

Figure 4.2b: Capture-HPC's configuration file (config.xml) for Capture-BAT.

Internet Explorer 8 is utilised as the browser in Capture-BAT that visits potentially malicious websites. Google chrome may have been a better choice since it is currently the most widely used internet browser but we experienced issues with having Capture-BAT launch the URL using chrome despite using the correct

settings. Capture-BAT was configured with 90 seconds to visit a given webpage. The time of 90 second per URL was found ideal as tests in the next chapter found. Reasoning here was simply that there were no new behaviours found after 90 seconds per page. This is tested in Chapter 3, where evidence is gathered from a small sample of data that suggests increases in intervals of 30 seconds do not affect the actual recorded gathered log file behaviour past the 90 second timer. The sheer volume of web pages that would undergo analysis would also justify this time choice.

4.5.3 Created exclusion lists within a Windows 7 application profile

Capture-HPC_NG comes with blank exclusion lists: exclusion lists have been developed and implemented from scratch for Windows 7 ‘n edition’ Service Pack 1, for both IE 8.0 and Firefox 29.01 on a clean O.S. running the following applications: Flash Player 12, Shockwave player 12, Java 7. Figure 4.3 shows the decision-making process to determine benign behaviour. Benign behaviour here refers to the system events that occur to browser, O.S. and applications installed as a result of running a tested benign webpage.

It is important to note that these tests should be carried out just before a chosen URL is run though Capture-BAT as there could be updates on a given website which could include new and poisoned advertisement streams. Any websites which arise uncertainty should be avoided at this stage as allowing malicious behaviours to be marked as benign would defeat the purpose of the exclusion list and allow similar malicious behaviours in subsequent analysis to be undetected. The methodology used is fairly self-explanatory: A sample of reasonably trustworthy websites are picked (the more behavioural analysis would mean more likely that the majority of system behaviours would be observed), analysed through multiple type of established malware analysers and the decision is made whether to use this as a

basis for understanding benign system behaviours when a client visits a website. It is recommended that a good range of websites should be picked specifically to include websites rich in web features. These can include videos, java and flash applications. The rationale behind this is simply to ensure that these rich features which undoubtedly would trigger file writes and possibly creation of relevant processes display their behaviour. These usual behaviours are expected behaviours and can be added in the exclusion lists as benign behaviours. A peer-reviewed paper was written and published on the development of Capture-BAT exclusion lists which included challenges and solutions. A copy of our cited paper can be viewed in the Appendix A.

4.5.4 Limitations of behavioural filters

The creation of behavioural filters can sometimes be quite subjective as a lot of behavioural exclusions are still unknown. These unknown exclusions refer to behavioural changes that occur in a system and limited knowledge is available as to what the effects of such changes are. An example of this would be: Internet Explorer adding registry entries but all that is given by a log file is a series of strings and numbers. Deciphering how a system was modified as a result of this change can often result in inconclusive results. This is often the case if the occurred change is not clearly displaying malicious behaviour and any potential benign behaviour is not documented by the browser/application or O.S. in question. Some behaviours that are not always observed, cannot inherently be seen or classified as malicious and are not documented. These can be referred to as 'grey' behaviour. Example log file behavioural entries are provided in Figure 4.0a, Figure 4.4 and log files in the appendix B. Consequently, the decision of the malware analyst to exclude these behaviours can also mean that exclusion lists would fail to detect certain exploits. It

is argued that it is therefore a requirement for a malware analyst to follow the good practice and guidance provided in the methodology when developing exclusion lists: more specifically if grey behaviour is observed they are not included in the exclusion list until a higher confidence level can be obtained.

Finally, behavioural filters can require a large sample set of data to be analysed so that trends and new behaviours which are limited by time are given the opportunity to display when their conditions are met. In the created data sets, a number of these behaviours are found and these are discussed in the log file analysis section with potential solutions explored in the next chapters.

4.7 Contributions to knowledge and key findings

- The chapter undertook behavioural exploration within Windows 7: typical system behaviours for file system, processes and registry explored and classed as benign, ‘grey’ or malicious to aid in the decision-making process of understanding behavioural log files and the creation of filters. The labelled behaviours provides insight on the expectation of manifestation and typical activities undertaken by a type of behaviour. This is useful as it allows a log file analyst to classify a behaviour or filter out noise within log files. The latter enables the creation of filtered datasets whereby identification of malicious behaviour or anomalies is easier. Section 4.4.1-4.4.4.
- Comparison of system calls between Windows 7 and Windows XP and the inclusion of Windows 7 native behaviours in the context of acceptable and benign behaviours. This is done in section 4.4.5 of Chapter 4.
- Methodology for creating benign behaviour filters in the context of the initial interaction and system call when attempting drive-by-download analysis.

- Adaptation of Capture-BAT from a Windows XP environment to Windows 7 environment: Example exclusion lists shared and published for Capture-BAT in the new environment, Windows 7.
- The methodology provides knowledge of the development lifecycle of behavioural filters provided which can be applied to any sandbox or honeypot that undergoes dynamic analysis on drive-by-downloads.

CHAPTER 5

Behaviour analysis environment experiments

Chapter 5 – Behaviour analysis environment experiments

This chapter presents the findings from the experiments undertaken. The first experiment seeks to assess the efficiency of the created behavioural filters. This is necessary in order to measure and evaluate the performance of the behavioural filters that were created in the previous chapter. Secondly, in the interest of answering the research question on behavioural environments and how the application of critical patch alters system behaviour manifestation, an experiment is conducted.

5.1 Assessing the efficiency of the created behavioural filters

This section is written up in experiment format as an experiment is undertaken to measure the performance of the created behavioural filters. Created behaviour filters assessed within real world context provided the means to determine the efficiency of filters created. Efficiency in this context refers specifically to the percentage amount of benign behaviours that are blocked by a behavioural filter in comparison to the recording all behaviours regardless of behavioural type (malicious, grey and benign).

5.1.1 Aim

To evaluate efficiency of created exclusion lists by calculating the reduction in behavioural data of confirmed benign behaviour that is filtered using the behavioural filter.

5.1.2 Hypothesis

It is hypothesised that the created behavioural exclusion lists are very effective at filtering non-essential or benign behaviour.

5.1.3 Experiment description

This experiment was tasked in comparing behavioural log files from visiting 3,011 real-world and potentially malicious websites in August 2016. These behavioural logs were compared from two gathered datasets, the former with the created exclusion lists and the latter without exclusion lists. The purpose of this is to assess the efficiency of created exclusion lists in line with real-world drive-by-download analysis.

Constant variable:

- Capture-BAT: Environment: O.S., browser, applications including same versions as well as same software.
- URLs that are visited must be the same.
- Capture-HPC server and settings: Server version, configurations including same amount of allocated time per website interaction.

Changing variables:

- Log files that are filtered using created behavioural filters (exclusion lists) vs. unfiltered log files.

5.1.4 Limitations & mitigations

Only one Capture-HPC server was available: This means both drive-by-download analysis cannot be simultaneously executed at the same time. To keep the test fair as possible, batches of URLs were broken down into groups and each changing variable (E.g.: Same URLs with and without exclusion lists) were run subsequently. Additionally, it is not possible to control which Capture-BAT client gets to visit which website. This is typically problematic as the behaviour analysis tool should be constant for each analysis. However, the two Capture-BAT clients that were utilised for this experiment are identical clones with the only difference being their Media

Access Control (MAC) address and Private IP address. It is logical to assume that this should not adversely affect the behaviours in each Capture-BAT client and thus, this therefore means that the observed system behaviours should be identical. In order to endorse this assumption, a small scrambled sample of 100 known and confirmed benign URLs were executed two times on a single Capture-BAT client setup. This was done so that each Capture-BAT client has the ability to analyse the 100 URLs at least once and there were no variations in the found behaviours between Capture-BAT clients.

Testing on a live website also has limitations: malicious web servers are short lived and malware writers could have updated their web servers between analysis which would mean different behaviours being observed from the time of the first analysis to the second analysis. Therefore, in terms of malicious behaviours, it would be possible to see different maliciousness in the same URL analysis when analysed at different times. However, benign system behaviours should not really be affected by these changes to malicious web servers, which makes the experiment realistic and correct in terms of assessing the benign behaviour filtering efficiency within the presence of real-world, dynamic and malicious web servers.

In order to replicate our results, it is important to note that there are limitations within real-world cyber security research: some malicious websites may be offline or have fully changed behaviours if analysed at a different date. Consequently, this would result in different observations from log file analysis accordingly, especially on the malicious behaviour perspective. Furthermore, it is possible to observe slight differences in behaviours if different Capture-BAT systems due to varying application, browser and O.S. versions or more static features such as the user name used within Windows 7. Controlling the application and O.S. versions are

within reach of replicability, however observing the exact behaviour is out of the researcher's power. In order to provide full transparency of our results and mitigate some of the mentioned limitations to create exact reproducibility, which are beyond my control: all the data used in the experiment is made available freely and I provide a detailed Capture-BAT setup as well as the created Windows 7 exclusion lists. Moreover, in the available log file data in raw format, the date and time of analysis of each malicious URL analysed is listed, in line with transparency guidelines provided by prudent practices within malware experimentation (Rossow et al., 2012).

5.1.5 Experimental set-up

The exact same setup that was previously used to create exclusion lists is used with the only difference being the server is now hosted on an allocated router within a home network's DMZ. This section of the network is secluded to the actual home network which additionally has a firewall in place to prevent malware propagation. Analysed malware interactions are therefore given full, unrestricted internet access and the Capture-BAT environments have a small virtual network between the Capture-HPC server which provides a bridge connection within Virtual Box to both Capture-BAT environments. Full details about the Capture-HPC server, Capture-BAT clients, their interactions, configurations and versions are documented in Chapter 4.5.1, in line with prudent guidelines by Rossow et al. (2012).

5.1.6 Methodology & justification

1. Analyse 3,011 URLs twice, once with exclusion lists and another without using the set-up discussed in Chapter 4.5.1.
    - Dynamic analysis requires a significant amount of processing time as this analysis is done in real time using real world visits to website and record behavioural interaction and thus the time between first and last

analysis should ideally be as close as possible (at least within 72 hours apart).

2. Export behavioural data of Capture-BAT log files and create a list of unique behaviours for both sets.
    - This reduces the amount of behavioural data that would need to be individually analysed. This is important as the sheer volume of data that is gathered from real-world malware analysis is overwhelming to a malware behaviour analyst.
3. Count frequency of unique behaviours.
    - This provides insight on which behaviours are most common and which are least common. This variable is important, especially for identifying ‘standard’ or ‘regular’ system behaviour. Frequently observed behaviour (particularly if these behaviours are then again captured from analysing benign webpages) which do not inherently show any signs of maliciousness can often be classified as benign behaviour.
4. Understand and label behavioural data (malicious & benign behaviours are of key interest: grey behaviours are ambiguous and inconclusive at times).
    - It is imperative that all benign behaviours are labelled as this would give the indication as to the amount of behaviours that exclusion lists are filtering. This would provide the statistics necessary for the evaluation of exclusion lists and their efficiency, thus proving or disproving the hypothesis.
    - Gathering malicious behaviours would provide insights on how the client system was attacked on different occasions which may provide variations in attacks that is of keen interest to a malware analyst.

5. Compare behaviours:
    - Benign behaviour in each dataset would provide an indication of the amount that is being filtered.
    - Counting frequency of behaviours in both datasets allows the quantification of how well or unwell exclusion lists responded to behavioural filtering.
    - If there are significantly higher malicious data in the unfiltered list, it would be subject to further full log file and attack investigations as there would be a requirement to understand if the exclusion lists were wrongly filtering malicious behaviours or if external factors affected the malicious behaviour outcome. However, there were not more malicious behaviours detected in the unfiltered dataset.

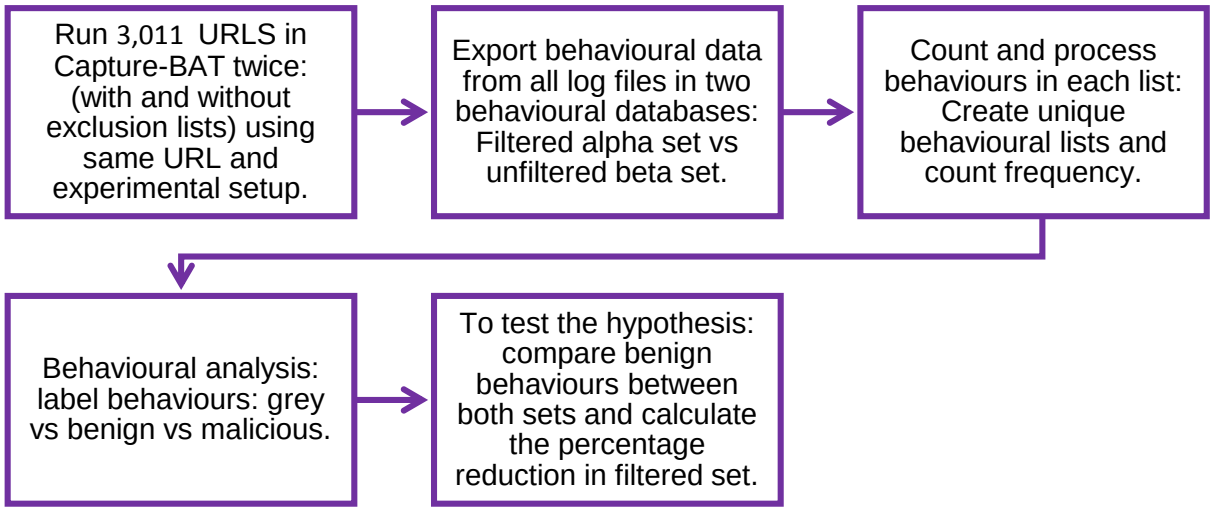


Figure 5.1: Overview of methodology used to calculate behavioural reduction in filtered exclusion list dataset.

5.1.7 Results

The results from the data capture of both changing variables are presented here. As this experiment is attempting to quantify the efficiency of exclusion lists, it is expected that filtered log files would have significantly less behaviours and log files altogether. Filtered log files which do not contain any potentially malicious behaviours found by the exclusion list are not created by Capture-BAT as the URL is marked as benign. Fully benign URL analysis in filtered Capture-BAT therefore does not create log files for a given analysis. This is the rationale behind the filtered dataset having less log files than the unfiltered list.

Alpha dataset overview:

- Capture-BAT created 112/3,011 potentially malicious filtered log files.
- Total of 17,018 behaviours within the dataset.
    - 16,102 of alpha dataset's total behaviours were benign.
- Consisting of 209 unique behaviours within the dataset

Beta dataset overview:

- Capture-BAT created 3,011 /3,011 potentially malicious un-filtered log files.
- Total of 703,721 behaviours within the dataset.
    - 696,000 of beta dataset's total behaviours were benign.
- Consisting of 3,173 unique behaviours within the dataset.

Table 5.1: Comparison if goodware behaviour from Alpha and Beta datasets.

Behaviour type	Alpha	Beta	Difference	Filter reduction(%)
File system write	13118	476519	463401	97.25%
File system delete	480	29405	28925	98.37%
Process creation	65	3424	3359	98.10%
Process termination	22	244	222	90.98%

Registry set key	2215	174284	172069	98.73%
Registry delete key	202	12124	11922	98.33%
Total benign	16102	696000	679898	97.69%
Average (Ex total)				96.96%

Average of filter reduction excluding total = 96.96

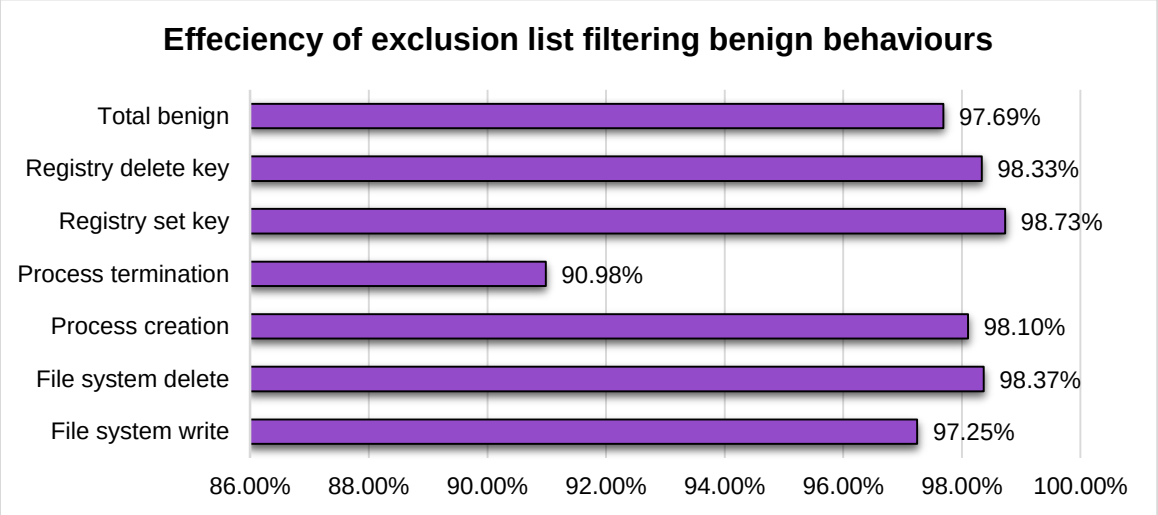


Figure 5.2: The percentage reduction in benign behaviour per behaviour type from using exclusion lists in Windows 7 Capture-BATs.

5.1.8 Conclusion

Having created exclusion lists to adapt Capture-BAT to a Windows 7 environment, there was a requirement to test the effectiveness of behaviour filtering that the exclusion lists were developed for. Accessing the efficiency of the exclusion lists would provide insight into how effective or ineffective the filter performed during real-world drive-by-download analysis. As the exclusion lists were created based on a large sample of benign webpages (~around unique 1000 benign URLs and several re-tests), the hypothesis was that exclusion lists would be highly effective at filtering benign and normal system behaviours. The experiment procedures were simply to run live tests by having a list of 3,011 URLs executed twice, with system behaviours

recorded both times: once with the usage of our Capture-BAT exclusion lists and the second, without exclusion lists. With the exception of timing and the usage of exclusion lists, variables such as environment version, applications, URLs, Capture-HPC server and Capture-BAT were kept identical to ensure fair and prudent malware experimentation. In terms of safety, this experiment was undertaken in on separate network dedicated for the Capture-HPC server and the two Windows 7 based Capture-BAT clients. Additionally, after each behavioural analysis a full revert of changes undertaken in the Capture-BAT environment were undertaken to ensure a constant and identical state was available for the next test in addition to wiping out any possible infections from malware.

As the assessment, here is purely focused on the ability of exclusion lists blocking known benign system behaviours, the occurrence of behaviours (referred to as the count or behavioural frequency) is arguably a good measure for observed behaviour quantity. This is because within dynamic behavioural analysis, it is observed often that a large percentage of behaviours are regular system behaviours which are present even in malicious dynamic behavioural analysis. These are in fact normal goodware behaviour that gets manifested regardless the presence of malware or not. By calculating the frequency of behavioural occurrence, it is possible to calculate and observe the effect that exclusion lists have on filtered log files. A large percentage in reduction of behavioural data is highly desired as this would allow a malware analyst to focus attention on malicious behaviours as well as unknown ‘grey’ behaviour during system interactions. Additionally, it is logical to assume that datasets absent large volumes of redundant, noise data would likely result in significantly more accurate classification and clustering.

In conclusion, the results for benign behaviours proved that the created behavioural exclusion lists reduce the amount of benign behaviours in total by 96.96%. Within each category there does not seem to be major differences as the percentage decrease for each behaviour was roughly the same. The only minor exception to this is the process termination behaviour which seemed to have the least difference (90.98% as opposed to an average of 97.06%). Within both datasets, benign process terminations tend to have a slightly larger variety of unique behaviours due to the Process Identifier (ID) being included by Capture-BAT. The inclusion of process IDs within behavioural analysis simply creates more unique behaviours as a result which causes a slight limitation within Capture-BAT if a particular behaviour has taken place. This behaviour is the termination of an unknown process by Services.exe and has a raw value frequency of 14. Should this have been included in the created exclusion list, process termination the percentage reduction for process termination would be much closer and in line with other behaviours at 96.72%.

Each behavioural category within the alpha dataset showed a huge decrease in behaviours within the log files that were gathered. Having this result is promising as this means that the created exclusion lists succeeded in achieving the identified goals in section 4.3. At least 96.96% of all identified, confirmed benign and normal O.S. system calls were not present in the filtered dataset, thus effectively reducing the volume of behaviours that a malware behaviour analyst would need to look at. Secondly, within the analysed 3,011 URLs, malicious behaviours were identified and formed part of several log files. These log files in the filtered experiment, contained only behaviours that were new and not present in the exclusion list with the exclusion

of all normal O.S. system calls. This showed once again that created exclusion lists were able to further meet the set goals of successful behavioural exclusion lists.

5.1.9 Evaluation

A key aspect of this experiment was to test the effectiveness of the created behavioural filters by measuring the amount of known benign behaviour whilst analysing real-world potentially malicious websites. The approach undertaken was a computing-intensive method that required dynamic analysis of webpages at least twice. An alternative and less process-intensive method could have been to run Capture-BAT without exclusion lists once and use behavioural filters to identify which benign behaviours would be filtered out. The secondary approach would be more suited to the task should the aim have been to query the knowledge within behavioural filters. This approach would have also resulted in the assumption that our behavioural filters would actually filter a given benign behaviour as opposed to performing real-world experimentation and observing definite, non-theory crafted results. By undertaking the more computer-intensive approach, the margin of human error and the level of bias from a malware analyst's judgement would be kept to a minimum. Capture-BAT sorts benign from malicious behaviours from a list of set principles. To elaborate: if a given behaviour is not listed in an exclusion list's ignore sections, no matter how small or insignificant the variation, it would be marked as malicious. In comparison, small variations (such as single character file path variations) could arguably be missed by a malware researcher and would have to be accounted for in log file analysis scripts.

In addition to this, real-world adaptation of the exclusion lists would have the opportunity to be tested: this is important as within cyber security and malware analysis, known working lab experiments always have the potential perform much

worst in real-world scenarios as agreed by the prudent practices for malware experiment designing by Rossow et al. (2012). By running the experiment within real-world context generalisations are avoided as it is possible to observe real-world performance. Finally, running analysis multiple times on the same domains at different time frames should be actively encouraged as often malware writers change exploits or reboot and modify exploits. Tanaka & Goto (2016) show this is the case in their dataset of 43,000 malicious websites where 10% of their 'changed occasionally' category had malicious websites revival over 15 times. These rational together with the potential of being able to discover a deeper insight into actual malicious behaviours supported the decision-making process. Positively, this proved to be the case as in addition to finding the percentage reduction in benign behaviours within filtered exclusion lists, a specific domain was found to project two initially similar behaviours but in both occasions, different system components were targeted. Observed malicious attacks were identified in each dataset which suggests that created exclusion lists did not inherently impacted negatively on capture. Malicious behaviour analysis on this case in undertaken in the next chapter.

This experiment was conducted towards the end of September 2016, showing that the exclusion lists are still fairly efficient (average 97% reduction) at filtering goodware behaviour.

5.2 Understanding how analysis environment changes different versions of an operating system

In this section, the work is focused on answering the research question: are there significant key behaviours that manifest differently in an analysis environment if the operating system has various levels of patches? This is an important design consideration as the proposed methodology requires behavioural filters to be tailored

specifically for the actual analysis environment that will be used to capture potentially malicious interactions. This is justified in the risk-averseness approach within design required to minimise the risk of potentially malicious behaviours to be filtered out by created expert driven behavioural filters. The experiment seeks to identify key differential behaviours as this would provide knowledge of their existence and support the risk-adverseness nature of the proposed methodology to create expert driven behaviour filters.

5.2.1 Aim

The aim of this experiment is to understand how the behaviour observed in malware analysis environment behaviours differ from a clean Service Pack 1 (SP1) Windows 7 O.S. to fully patched Windows 7 O.S. which includes both critical and security patches released by Microsoft as of August 2016.

5.2.2 Hypothesis

1. It is hypothesised that there would be significant behaviours additions within the behavioural environments as numerous critical patches within Windows 7 was applied.
    - Core processes such as svchost.exe control a large amount of behavioural interactions and if the new patches do not introduce a large amount of core processes, it is likely that there will be some new unique system calls introduced in the applied 164 critical and security patches. The reasoning behind the minor changes in behavioural environment is justifiable as new patches within operating systems often can include diverse ways to perform existing tasks or perform new tasks. Modification of existing tasks and creation of new tasks

would undoubtedly result in new behaviours manifesting within the environment regardless.

2. It is hypothesised that due to a large number of patches being installed on the patched Windows 7 analysis environment, it is likely that a larger amount (frequency) of the unique behavioural interactions may be observed. This refers to the amount of times a given unique behaviour is observed.
    - The justification here is again in line with the potential of new behaviour and the likelihood that a large number of patches would likely increase the amounts of observable behaviour.

5.2.3 Experiment description

This experiment seeks to identify key behaviours that are occurring in the instance of an updated Windows 7 behaviour analysis environment. In order to identify these new behaviours, there was a requirement to observe the system calls that executed in an unpatched Windows 7 environment. Findings are of crucial importance when designing and creating analysis environments as further insight were gained from studying benign and newly introduced behavioural interactions. Furthermore, should the hypothesis be proven: insights on how analysis environments can provide different results may be synthesised. Finally, this experiment reinforces existing knowledge where different versions of the same operating system behaves differently.

To describe the experiment as a summary: Windows 7 SP1 with no updates Capture-BAT clients were cloned and updated fully to September 2016 critical updates. Existing and un-updated Capture-BAT environments and the newly updated Capture-BAT clients performed analysis on the same set of benign URLs.

Benign behaviour of each type of Capture-BAT clients were then analysed and differences in behaviour investigated and evaluated.

Constant variable:

- In addition to the main variables discussed, chosen benign URLs that are visited must be the same domains.
- Capture-HPC server and settings: Server version, configurations including same amount of allocated time per website interaction.
- Capture-BAT analysis environment: applications, browser (IE8).

Changing variables:

- Capture-BAT analysis environment: Windows 7 O.S. version: Un-updated Windows 7 Service pack 1 vs fully updated Windows 7 Service pack 1 for critical patches. The full list of applied patches utilised in the experiment is available in Appendix C to provide full experimental transparency.

5.2.4 Limitations & mitigations

- Time: websites can often change behaviour as they are updated, for this experiment to be a fair test there should be no more than 72 hours between first and second analysis. This is more applicable for malicious websites however, but there is still a distinct possibility which can mean the chosen list of benign URL could have been altered. However, if the behaviour observed is still benign, this is unlikely to have an impact on what Capture-BAT classifies as malicious.

5.2.5 Experimental set-up

Two cloned and fully identical Capture-HPC servers with different Windows 7 Capture-BAT environments were needed and created using the same experimental setup used in section 4.6.5 of the thesis with the exception here being no exclusion lists were used in the process to identify varying behaviours in both datasets. It is important to ensure that clones were used for both the Capture-BAT and Capture-HPC server to ensure a high confidence level of identical settings, software version and application system calls.

A sample of 800 benign URLs were analysed in two different Capture-BAT environments: a fully patched Windows 7 environment with all the critical windows 7 patches installed and an unpatched Windows 7 environment from a default SP1 copy.

5.2.6 Methodology & justification

This experiment seeks to answer, 'How does applying Windows 7 operating system critical patches affect the benign interactions undertaken in an analysis environment?' There was therefore a requirement to create two analysis labs: one patched and an unpatched version. By performing analysis on the same list of URLs in a similar time frame, it was possible to create specific datasets for each analysis environment. These two analysis environments were hypothesised to have different system calls and the captured behaviour from both datasets can be compared. Differences and newly observed behaviour from either type of analysis environments can be quantified and assessed, which lead to findings on actual real-world behavioural changes in analysis environments. As this method controls the factors that may affect the results, including: analysis environment, URLs visited and keeps the same list of enabled start-up services and applications it is fair to state that the

results obtained were correct, transparent and accurate. In terms of safety, this particular analysis used confirmed benign webpages which underwent thorough testing, utilising the proposed benign website picking methodology introduced in section 4.5.2 in Figure 4.3. In addition to this, the same safety precautions discussed in chapter 3 are applied for further confidence relying upon defence in depth.

In terms of analysing the data: behaviour filtration was a required activity in this experiment. Initially before the behaviours were compared, behaviours within log files that were shared by both datasets were filtered out. The purpose of the experiment was to identify key behaviours that are significantly different between patched and unpatched Windows 7 systems. Having similar behaviours that are present in both sets and are 100% identical therefore adds additional noise in the dataset as they did not help the assessment of the hypothesis. In order to do this, only full matches in behavioural entries were compared from both sets. Identical unique behaviour would then need to be removed before analysis of each behavioural datasets is conducted. This was carried out using an exact string match function that looked for exact matches of a given behaviour in the unpatched dataset and then this would be compared with all other behavioural entries in the patched dataset. Exact matches present in both datasets were then excluded. The purpose is to perform investigation upon differences between behaviours that are not co-existent in both data sets making this process of crucial importance, especially due to the large volumes of behavioural data contained.

In terms of sample size evaluation, the nature of the test is aimed at studying the normal system behaviours. 800 benign URL tests were carried out in each behaviour analysis environments. This figure should be more than sufficient to provide a good overview of constant and regular system behaviours. It is important

to note that the visitation of benign webpages has been observed beforehand and it is typically it is very rare for new behaviours to be observed as there is a level of regularly occurring system behaviours that can be observed in a benign behaviour test dataset. If this experiment was aimed at identifying malicious behaviour changes, it is fair to say that the figure of 800 would not be fully representative of the real-world outlook. However, the two main reasons listed here justifies the figure of 800 being reasonably sufficient for the purposes of answering the patch and unpatched behaviour analysis environment research question which relies on datasets that are not poisoned by maliciousness.

5.2.7 Results

The results show that the patched analysis environment had over 5 times more behavioural interactions than the unpatched Windows 7 dataset. There were significant differences between the system calls experienced in both analysis environments. The result's section of the experimental writeup was written from the perspective of the process that is undertaking (or executing) a given system call. The presence in respective datasets are identified and unique, observable behaviours are evaluated in light of answering the hypothesis. The behaviours that were of interest are predominantly the ones that are not observed at all in the unpatched dataset as these behaviours are the ones that provide direct insight on newly observed system calls within patched Windows 7.

Keys:

~ = Grouped similar unique behaviours.

Green = Identical behaviours shared between both post unique behaviour filtering.

Red = Native behaviours to either patched or unpatched.

Grey = Observed anomaly behaviour: behaviours that were too rare to be considered as discovered differences between patched and unpatched datasets.

It was observed in the patched dataset that Sppsvc.exe which is responsible for download, installation and enforcement of digital licences was executed by services.exe multiple times. The design of Capture-BAT's log files for process behaviours can inaccurately represent processes as a created or terminated process is given a process identifier. This identifier contains a process ID which is added into the log files for processes. The impact of this caused multiple unique entries (59) with a frequency of 309 times in the patched dataset. In comparison, the same process interaction was limited to 12 unique entries and observed 32 times within log files from the unpatched dataset. In terms of behaviour analysis, as the behaviour of creating the process is similar. An example of similar behaviour here includes: "C:\Windows\System32\services.exe","created","2576","C:\Windows\System32\ Sppsvc.exe "

"C:\Windows\System32\services.exe","created","2876","C:\Windows\System32\ Sppsvc.exe "

In this example, it is evident that services.exe is creating a Sppsvc.exe process in each case but the process ID is different. As this process identifier is unique only to the process field in Capture-BAT log files, as part of transforming the dataset we assume the behaviours are similar despite minor variations (different process identifier) within log files. It is reasonable and feasible to ignore unique variants of the same behaviour as they are essentially the same behaviour manifesting in different instances. This therefore leaves us with the comparison of the frequency as the same number of websites were analysed in both datasets. From the results, it is suggested that the updated dataset experienced this particular behaviour 9.6 times

more than the unpatched Windows 7 Capture-BAT instance. Furthermore, this particular process had numerous interactions with the windows registry and file system writes in the patched dataset which was not observed at all in the unpatched dataset. Specific behaviours are listed below and it would appear that file system behaviours of write were very frequent whilst registry interactions from a benign Sppsvc.exe was rare and limited to a single section of the registry. As the table below shows, sppsvc.exe was constantly writing log files and the system folders within the Windows 7's System32 directory. The high occurrence present in the log file suggests that this is a difference in behaviours from patched to unpatched Windows 7 analysis environments.

Table 5.2: Additional observed behaviours for Sppsvc interactions present only in the patched dataset.

Behavioural interaction	Count
C:\Windows\System32\sppsvc.exeSetValueKeyHKLM\SYSTEM\WPA\8DEC0AF1-0341-4b93-85CD-72606C2DF94C-5P-19\Time-1	5
C:\Windows\System32\sppsvc.exeSetValueKeyHKLM\SYSTEM\WPA\8DEC0AF1-0341-4b93-85CD-72606C2DF94C-5P-19\Type-1	5
C:\Windows\System32\sppsvc.exeSetValueKeyHKLM\SYSTEM\WPA\8DEC0AF1-0341-4b93-85CD-72606C2DF94C-5P-19-1	6
C:\Windows\System32\sppsvc.exeWriteC:\Windows\System32\config\SYSTEM.LOG1-1	386
C:\Windows\System32\sppsvc.exeWriteC:\Windows\System32\config\SYSTEM-1	342
C:\Windows\System32\sppsvc.exeWriteC:\Windows\System32\config-1	65
C:\Windows\System32\sppsvc.exeWriteC:\Windows\System32-1	6

Svchost.exe process was found to have a few groups of similar behaviour. In this context, similar behaviour refers to behaviour that isn't identical due to variations in the behavioural entry but essentially is observed to be performing the same action. For instance, a different folder name is used at one point in the behavioural interactions. Results from svchost.exe process interactions are presented in 5.3

below. In the unpatched dataset, these groups which were controlled by Svchost.exe displayed sets of three main behaviours: Svchost.exe creating and terminating mobsync.exe, writing (.etl) log files in drifted windows 32 system directory folders (401 unique but similar and 1203 total count) and writing Cryptneturl content files in the user directory. On rare occurrences, it was observed that Svchost.exe had 3 unique registry SetValueKey entries of a single frequency in the WBEM directory of the registry. These incredibly low occurrences of this particular behaviour are anomalies which dynamic (real-world) analysis can sometimes capture. While causations of anomalies behaviours are beyond the scope of this work it is important to identify them so they are not included in the list of behaviours which differ from the patched and unpatched dataset.

In contrast to this, the patched Windows 7 setup was observed to have a higher sheer volume of behavioural interaction for Svchost.exe. However, the quantity of occurrence alone does not help identification of the key benign behaviours that are manifested. Further investigation in observed behaviours concluded with three native behaviour groups for Svchost.exe (outlined in red in the table below) that was present only in the patched Windows 7 version. Recent file cache logs seem to be monitored and edited frequently. Windows driver install was triggering at every analysis and in comparison, to the unpatched system, there were large amounts of unique registry entries that added values in the object table part of the registry done by Svchost.exe.

Table 5.3: Comparison of Svchost process interactions in patched and unpatched Windows 7 datasets:

Behaviour	Patched Windows 7	Un-patched Windows 7
~Writing .etl files in System 32	1686	1203

Writing recent file cache log	548	0
~Writing CryptnetUrlCache metadata	22	28
~Created / terminated mobsync.exe	21	57
~Registry WBEM transports set keys	0	3
~Created / terminated drvinst.exe	2540	0
~Object table set registry keys	1839	0
Registry WBEM last service start set key	1	0
Registry Audio policy config set key	1	0

The Internet Explorer process (iexplore.exe) interactions was observed in both datasets and other than having almost double Windows Error Reporting files being written, there were no significant differences observed during the analysis of 800 benign URLs.

Table 5.4: iexplore.exe behavioural interactions in both datasets.

Behaviour	**Patched Windows 7**	**Un-patched Windows 7**
~Writing and deleting Flash Player .sxx files	96	60
~Writing Windows error reporting files in user folder	2922	1386
~Clearing .dat files in user IE folder	2	0
~Adding registry keys in session filter	2	5
Writing DAT files in user directory	1	0
Adding registry keys notifying completion of download	1	0
Adding registry keys in Admin Active recovery	1	0
Deleting .dat files in user directory	2	0

Table 5.5: Services.exe behavioural interactions in both datasets.

Behaviour interactions by services.exe	**Patched Windows 7**	**Un-patched Windows 7**
~Executes and terminates sppsvc.exe and wsqmcons.exe	333	49
Writes system.log1 in System 32 folder	59635	0
~Writes folder in System 32 folder	60073	0

Table 5.6 Taskhost.exe behavioural interactions in both datasets.

Behaviour interactions by Taskhost.exe	Patched Windows 7	Un-patched Windows 7
Writing .tmp file in program data directory	12	12

In addition to the discussed differences that were observed between both datasets, it was observed that the patched Windows 7 behavioural environment had a number of native system calls. One such example is the triggering of Windows 7's Rundll32.exe process which executed and terminated dinotify.exe 402 times. Rundll32.exe was observed to manifest with multiple deletion of keys in the registry as shown in Table 5.7 below.

Table 5.7: Patched behavioural environment showing unique Rundll32.exe interactions.

C:\Windows\System32\rundll32.exeDeleteValueKeyHKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName-1	298
C:\Windows\System32\rundll32.exeDeleteValueKeyHKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass-1	298
C:\Windows\System32\rundll32.exeDeleteValueKeyHKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName-1	298
C:\Windows\System32\rundll32.exeDeleteValueKeyHKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass-1	298
C:\Windows\System32\rundll32.exeSetValueKeyHKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect-1	298
C:\Windows\System32\rundll32.exeSetValueKeyHKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet-1	298

Dinotify.exe and Drvinst.exe (previously identified above) are both related processes controlling the installation of device drivers in the Windows 7 environment. Further behavioural interactions were observed in the patched dataset which were consequently captured as a result of the same types of process interaction. Drvinst.exe performed a number of similar registry set and delete interactions. These are available in Appendix C and were registry entries that controlled device driver

information that is held within the Windows 7 registry. Out of all the registry interaction inclusive of with Drvinst.exe, it was noticed that setting MuiCache language set value was often triggered (4665). While this path often is accessed by other services such as Iexplore.exe or Explorer.exe, it was observed that this particular process was adding values in the patched dataset. Finally, the process Drvinst.exe also wrote log files within the C:\Windows\inf\setupapi.dev.log-1 multiple times (467) which was not present in the unpatched Windows 7 dataset.

A few unknown behavioural interactions were observed natively in the patched dataset. The system was recorded to be adding registry entries within volume shadow copy service. Unfortunately, the lack of information about the values being added meant this behaviour could not easily be classified initially. Further investigation revealed that this technology is an included and benign service that runs in Windows which manages snapshots of computer files and volumes and is often used to create shadow copies of services. Windows backup was disabled on the client machines (both patched and unpatched datasets). This makes the discovery of these behavioural interaction in the patched dataset interesting and noteworthy and useful for creation of exclusion lists aimed at patched Windows 7 honey clients.

Table 5.8: Shadow copy service triggering in patched dataset.

SystemSetValueKeyHKLM\SYSTEM\ControlSet001\services\VSS\Diag\VolSnap\Volume{41003fa4-89a3-11e3-bfdc-806e6f6e6963}Dismount (Enter)-1	163
SystemSetValueKeyHKLM\SYSTEM\ControlSet001\services\VSS\Diag\VolSnap\Volume{41003fa4-89a3-11e3-bfdc-806e6f6e6963}Dismount (Leave)-1	163

Finally, the remainder of observed behaviours that were native to the patched Windows 7 analysis environment had a number of varying processes. The single occurrence of each behaviours suggests that these behaviours are rare and not regularly seen (as they only flagged up once in 800 benign webpage analysis).

Table 5.9: Rare and native behaviours observed within patched Windows 7 behavioural analysis environment

C:\Windows\explorer.exeSetValueKeyHKCU\Software\Microsoft\Windows\ CurrentVersion\Applets\SysTray\Services-1	1
C:\Windows\System32\lsass.exeWriteC:\Windows\System32\Microsoft\Protect\ S-1-5-18\7a5af875-d05f-4223-913b-8fe1fbd3090a-1	1
C:\Windows\System32\lsass.exeWriteC:\Windows\System32\Microsoft\Protect\ S-1-5-18\Preferred-1	1
C:\Windows\System32\wsqmcons.exeSetValueKeyHKLM\SOFTWARE\Microsoft\ SQMClient\Windows\AdaptiveSqm\ManifestInfo\Version-1	1
C:\Windows\System32\wsqmcons.exeSetValueKeyHKLM\SOFTWARE\Microsoft\ SQMClient\Windows\WSqmConsLastEventTimeStamp-1	1
C:\Windows\System32\wsqmcons.exeSetValueKeyHKLM\SOFTWARE\Microsoft\ SQMClient\Windows\WSqmConsLastRunTime-1	1

5.2.8 Conclusion

The main aim of the experiment was to identify the differences in behaviour that could be observed from a patched Windows 7 behaviour analysis environment in comparison to an unpatched environment. The created hypothesis is proven to be true as the lists of behaviours available in the results section show that there are several entries of new and unique behavioural interaction in the patched Windows 7 dataset. Core processes including Sppsvc.exe, Svchost.exe, Rundll32.exe and Services.exe were found to be the processes that had the most amount of new and unseen behavioural interactions when comparing between patched and unpatched dataset was undertaken. Furthermore, it was observed that the exclusively executed system calls in the patched dataset were often occurring very frequently upon the analysis of benign websites. Examples of specific behaviours that are only observed in the patched Windows 7 behaviour analysis environment are presented in the

result section and quoted here for illustration purposes: Sppsvc.exe wrote log files 386 times during the analysis of 800 websites. Svchost.exe wrote recent file cache log 548 times, created and terminated drivinst.exe processes 2540 times and had a total of 1839 entries in the object table field of the registry. Services.exe wrote logs 59,653 times in the system 32 folder within Windows 7. These specific behaviours were not observed at all within the unpatched Windows 7 and therefore contributed to the conclusion that they were native to the patched Windows 7 system. Undoubtedly, finding these specific behaviours mean that there are significant changes between unpatched and patched datasets in terms of observable behaviours. In this case, significant changes refer to two core aspects of observed behaviour: frequency of observed behaviour and behavioural uniqueness. The frequency simply is a count of occurrence of a particular behaviour while the behavioural uniqueness observes new behaviour.

One of the core procedures of comparing the datasets involved finding out which behaviours were not shared between both datasets. This task additionally meant that a large amount of noise was removed. In order to create these lists, string/text comparison formulae were used to compare each behaviour in the unpatched dataset with the patched dataset. Upon identification, these behaviours were studied and mapped into behavioural groups which were generated by the process that initiate the system call. The created groups were then thoroughly analysed in both unpatched and patched Windows 7 datasets as discussed in the results section. It is important to note that every observed behaviour that was native to a single dataset was assessed during this analysis stage. All new behaviours that were identified were then flagged accordingly. The flagging process would assign a colour appropriate to a specific, unique and observed behaviour. These then meant that

concrete evidence could be gathered from the grouped system calls in terms of relevant differences between unpatched and patched Windows 7 analysis environments.

In terms of the second hypothesis, regarding the higher volume of observed benign behavioural interactions in the patched Windows 7 system: it is concluded that this was the case in the majority of processes present in the unpatched dataset, effectively proving the hypothesis. Evidence of this can be observed in the results provided: the unpatched set had a total of 1386 Windows Error Reporting system calls and the patched Windows 7 analysis environment had 2922 of the same system calls, which is 2.1 times more than the unpatched set. Flash player file manipulation was triggered 60 times in the unpatched dataset whereas the in the patched dataset it was triggered 96 times. Both small scale and large scale Svchost.exe interactions were found to be significantly higher in the patched data set. Small scale grouped interactions such as Svchost.exe writing snapshot log files are triggered 1203 times in the unpatched dataset and 1686 times within the patched dataset. Full process interaction for the major process, Svchost.exe had a total of 6658 occurrences in the patched dataset in comparison to 1253 in the unpatched dataset, clearly supporting the hypothesis from multiple process interaction perspectives. It is important to note that these finding offer perspective over the observed behavioural interactions on the analysis labs that was created based on Windows 7. These behavioural quantities are likely to differ in other analysis labs which use different versions of applications and operating system. The quantity of observed behaviour is an important metric within this research as this provides indications on two core aspects: firstly, exclusion list efficiency can be measured and secondly, it is often a safe metric to use in behaviour labelling when a behaviour is

observed. Frequently observed behaviour in known, uninfected from malicious web exploits and clean systems allows a malware analyst to classify the behaviour as benign which subsequently can be added to behavioural filters. Alternatively, rare behaviours can be viewed as anomalies that would require further investigation before any knowledge is concrete.

From undertaking of this experiment, it is fair to conclude that the results provided further evidence of the need for exclusivity in the creation of filters for different behavioural environments. The justification behind this claim is backed up by the significant differences in behaviour that benign and well established processes were exhibiting. For instance, the results showed that Services.exe was performing a lot more log file writes in a patched Windows 7 dataset in comparison to the unpatched version where this behaviour was not captured. Whilst experimental prudence requires not generalising results observed, it is fair to conclude that in the case of the experiment undertaken: operating system behaviour was significantly different with the application of critical and security patches within the Windows 7 environment. It is likely that different versions of software installed upon the analysis environment would result in significantly different behaviour which would require specialised and personalised filters. It is not new knowledge that when it comes to behavioural environments, the creation of tailored exclusion lists that are created for a specific analysis environment is a requirement. The findings of this experiment are in line with this existing knowledge as the results section showed two versions of operating system having different system calls.

5.2.9 Evaluation

The assessment of the hypothesis undertaken was successful. This is because the created methodology when applied to the experiment allowed the creation of two

datasets which when analysed proved to have significant differences thereby proving the hypothesis. In addition to this, the experiment was closely monitored and controlled: external factors which would affect the experiment's correctness of dataset such as hardware, virtual machine configurations, benign website URL lists, application versions, browser version, start-up programs and Windows 7 configurations were kept identical in both instances of analysis labs. This practice is in line with the correctness of data aspect of the prudent experimental framework discussed in chapter 3. To ensure correctness of data, the benign website picking framework created in Chapter 4, was used to select benign URLs and at the time of analysis, none of the chosen URLs displayed any signs of being compromised within 12 hours before analysed in the high interaction client honeypot.

The only condition that was changed, was the Windows 7 updates that was applied in each case and this was the changing variable which would provide observable knowledge when behaviour of the environment analysis was undertaken. In addition to these, the decision of running these experiments without created behavioural filter was made. This decision meant that observable benign behaviours would be in great abundance in the created log files as they were not being filtered. This decision meant that quantity or frequency of behavioural occurrence could be measured for this experiment which, according to the conclusion proved to be an area where patched behavioural analysis labs typically had more system calls than unpatched labs. The datasets created are thus correctly created in line with applicable experimental prudence guidelines and with careful consideration regarding relevant research questions concerning the differences in patched and unpatched analysis environments.

Some aspects regarding experimentation transparency which Rossow et al. (2012) discuss that include malware specific conditions are not applied to this particular experiment. Examples include stating employed families, malware sample selection and reasoning for false positive and negatives. These are not applicable for this particular experiment as the focus was on understanding how environment labs differ in behaviours if they are patched: which by requirement focuses on normal, benign behaviours. However, aspects of data creation replicability such as environmental variables used in the experiment and the URL list analysed are provided in addition to the actual dataset.

In terms of alternative ways to perform behaviour drift and change analysis between different operating systems, a lightweight approach could be to use the Process Manager or Task Manager application to identify new processes within a patched operating system. According to Russinovich & Solomon (2009), Windows b ased operating systems also have a Process Monitor tool built in Sysinternals which can be used to monitor processes. These would provide a malware analyst with a number of processes that are executed in both versions of the operating system and this information can be used to identify new processes that are running in a patched operating system. This method however would not provide insight on how a process interacts with the analysis environment but if the goal is merely to identify new system call vectors, this would be suitable. In terms of improvement, an alternative approach to this experiment could have been to incrementally break the available updates into a monthly set of groups. Every month, all released updates would be required to be installed on the Capture-BAT analysis environment and perform analysis over a consistent set of benign URLs. These datasets could be used in monthly analysis to provide in depth behaviour changes in the Windows 7 Operating

System from patch to patch. While this approach is highly exhaustive and in depth, it did require the Capture-BAT environment to be updated every month which was not possible as when this research question was identified towards the end of the study after noticing variable differences from a pilot study. It is important to note that while the undertaken approach does not provide the potential level of detailed a dedicated study of that nature would provide: it does provide valuable insight between updated and non-patched Windows 7. Knowledge from these can be useful in terms of setting up appropriate honey client and behaviour analysis labs to address specific user-needs from client matching requirements.

While utmost experimental prudence and attention to detail was undertaken, it was observed that a set of behavioural interaction was triggered in the patched Windows 7 dataset which are related to Windows 7 driver behaviours. These behaviours tend to occur on a ‘as-need’ basis where Windows discovers a new device and attempts to find drivers for that new device. Part of the task list of performing updates on the Windows 7 Capture-BATs involved unlocking the immutable Virtual Disk Image (.VDI). This requires the virtual device to be detached from the virtual machine prior to modifying the settings for the disk image file. Post the installation of updates, this task should happen once more as the virtual drive image is changed from the normal mode within settings back to immutable mode to keep the behaviour environment constant and unchanged by website analysis. During this process, it is good practice to reboot the virtual machine a few times prior to the final setting locking in order to achieve a constant environment where behaviours such as driver install prompts caused by attaching the virtual disk to the virtual machine. Unfortunately, the environment’s virtual drive image was locked before the full successful installation of the required drivers which caused the prompt in Windows at every boot up, resulting

in a number of device driver related interactions. In order to address this concern, the hard drive was unlocked, the drivers required was allowed to be installed before being re-unlocked. Unfortunately, while the expectation of this task was so that these behavioural interactions caused by the virtual disk image should have stopped, further benign website analysis proved that despite drivers being installed, these interactions were still observed. The actual reasons behind these behavioural interactions in the patched dataset could have been as a result of the virtual hard drive manipulations undertaken but in actuality this was observed to not be the case, having tested under the correct conditions. It is therefore concluded these changes were the actual observed changes in the patched Windows 7 environment. Exploring the actual reasons for why some system calls are triggered at certain times are beyond the scope of this research and is a viable path for future research as often causes for system calls being triggered are simply not justifiable with the information available on how processes in Windows 7 works.

5.3 Key findings

- Tested the ability to assess the created exclusion lists in Chapter 4. The filtered behaviour contained in log files was reduced by an average of 96.96%. These did not miss exploits and malicious behaviours but in turn provide the benefit of sorting between malicious and benign behaviours in an ever-growing web of malicious web servers. This proves that the created method for creating exclusion list as well as created exclusion lists were highly effective at reducing noise in log file datasets.
- Differences in observed benign behaviours between patched and unpatched Windows 7 behaviour analysis labs with identification of natively found behaviours in patched systems.

- Identification of existing core processes namely Sppsvc.exe, Svchost.exe, Rundll32.exe and Services.exe were found to have the most amount of new and in some cases unknown behavioural interactions within the study that differed between patched and unpatched Windows 7.

CHAPTER 6

Log file analysis and observed Windows 7 attacks

Chapter 6 – Log file analysis and observed Windows 7 attacks

6.1 Behavioural drive-by-download log file analysis

This section presents knowledge, patterns and trends from the gathered data corpus. Behavioural information on malicious drive-by-downloads is identified, explored and evaluated with conclusions on findings drawn. In the gathered 5,132 log files a large number of behaviours were observed. The volume of behaviours for the gathered log files consisted of 294,264 behavioural entries. It was observed that within the sample, there were 9,533 unique behaviours. In the following discussion on behavioural findings they are undertaken from a file system, process and registry behaviour perspective and key results will be highlighted.

6.1.2 Behavioural distribution analysis

Figure 6.1 and 6.2 below show the distribution of system calls in the 5,132 malicious log file dataset. Values utilised in the behavioural distribution charts are available in Appendix D, presented as tables 6B-6I. From the behavioural distribution, it is concluded that the vast majority (80.01%) of drive-by-download utilise the file system's write as their primary attack vector. In addition to this we found that within our log file data set that the most commonly faced malicious behaviour is:

'C:\Program Files\Internet Explorer**iexplore.exe Write C:\Users\mp\AppData\Local\Temp\svchost.exe'**

This means a compromised Internet Explorer browser is writing a suspicious svchost.exe file in the user's temp folder. The parts in bold are the parts of interest to an analyst: the iexplore.exe process is the actual browser in the default location. The keyword write is a file write on disk function and the final part is the actual file with the maliciously chosen directory path. This particular behaviour formed 21%

(encountered 62,488 times) of a possible total of 294,864 behaviours. It is likely that this behaviour could vary should different browsers have been chosen for the analysis process as some browsers have different file system location for temporary files. This particular strain of malware, due to its persistence is the most popular attack vector in the entire data set. This therefore demanded further investigation on the file that was being downloaded and saved into the user temp folder in each instance. Table 6.0 below shows 7 anti-virus vendor classification that found all instances of the captured binary by Capture-BAT. These submissions to Virus Total all contained the particular file write where Internet Explorer process was hijacked into writing a malicious Svchost.exe in the user temp folder.

Table 6.0: Malware analysis of the malicious file write (Svchost.exe) within the user temp folder to find signatures and family classification by Anti-Virus (AV) vendors.

Signature distribution between AVs	Percentage within the dataset (%)
AhnLab-V3	
Win-Trojan/Bamital.Gen	28.57%
Win32/Ramnit.B	9.52%
Trojan/Win32.Zbot.N133057644	33.33%
Trojan/Win32.Bamital.N241184294	14.29%
Trojan/Win32.Zbot.N134991950	14.29%
Avware	
Trojan.Win32.Encpk.aak (v)	28.57%
Virus.Win32.Ramnit.a (v)	9.52%
Trojan.Win32.Generic!BT	47.62%
Packed.Win32.Zbot.gen.y.5 (v)	14.29%
AVG	
Generic22.BPCM	42.86%
Win32/Ramnit.A	9.52%
PSW.Generic12.AMWG	33.33%
SHeur3.ARNR	14.29%
Ad-Aware	

Backdoor.Agent.ABHW	42.86%
Win32.Ramnit	9.52%
Trojan.Zbot.IPC	33.33%
Gen:Heur.FKP.1	14.29%
Kaspersky	
Trojan.Win32.Pakes.tyi	42.86%
Virus.Win32.Nimnul.a	9.52%
Packed.Win32.Krap.hm	47.62%
Sophos	
W32/Ramnit-A	42.86%
W32/Patched-I	9.52%
W32/Ramnit-ET	33.33%
Mal/FakeAV-JN	14.29%

Furthermore, in the top 20 most commonly encountered behaviour in the 5,132 Capture-BAT log file datasets which is available in Appendix D, table 6D file system writes dominated the count table, by claiming 15 out of a possible 20 spots with the rest being registry's set value changes.

These results support the behavioural distribution in Figures 6.1 and 6.2. Furthermore this raises a number of concerns within cyber security and drive-by-download research as far too often, the same behaviour is observed where a compromised browser process usually through process injection writes a malicious executable without user consent in the user's temp folder within Windows 7. Modern browsers have built in sandboxes which would likely lessen the occurrence of known malicious Portable Executable files saved by drive-by-downloads. Clearly whilst visiting a webpage, Windows executable files (or any file which has the potential to run automated scripts) should not be downloaded without at least asking the user for the permission to do so. Due to the vast amount of captured behaviour that follow this pattern, it is concluded that the browser process is one of the most targeted exploit vectors susceptible to process injection attacks which additionally shows (due

to the large volume of successful attacks in our 5,132 log file dataset) numerous vulnerabilities within the Windows 7 environment. Additionally, the user temp folder is identified as the most targeted file path within our environment.

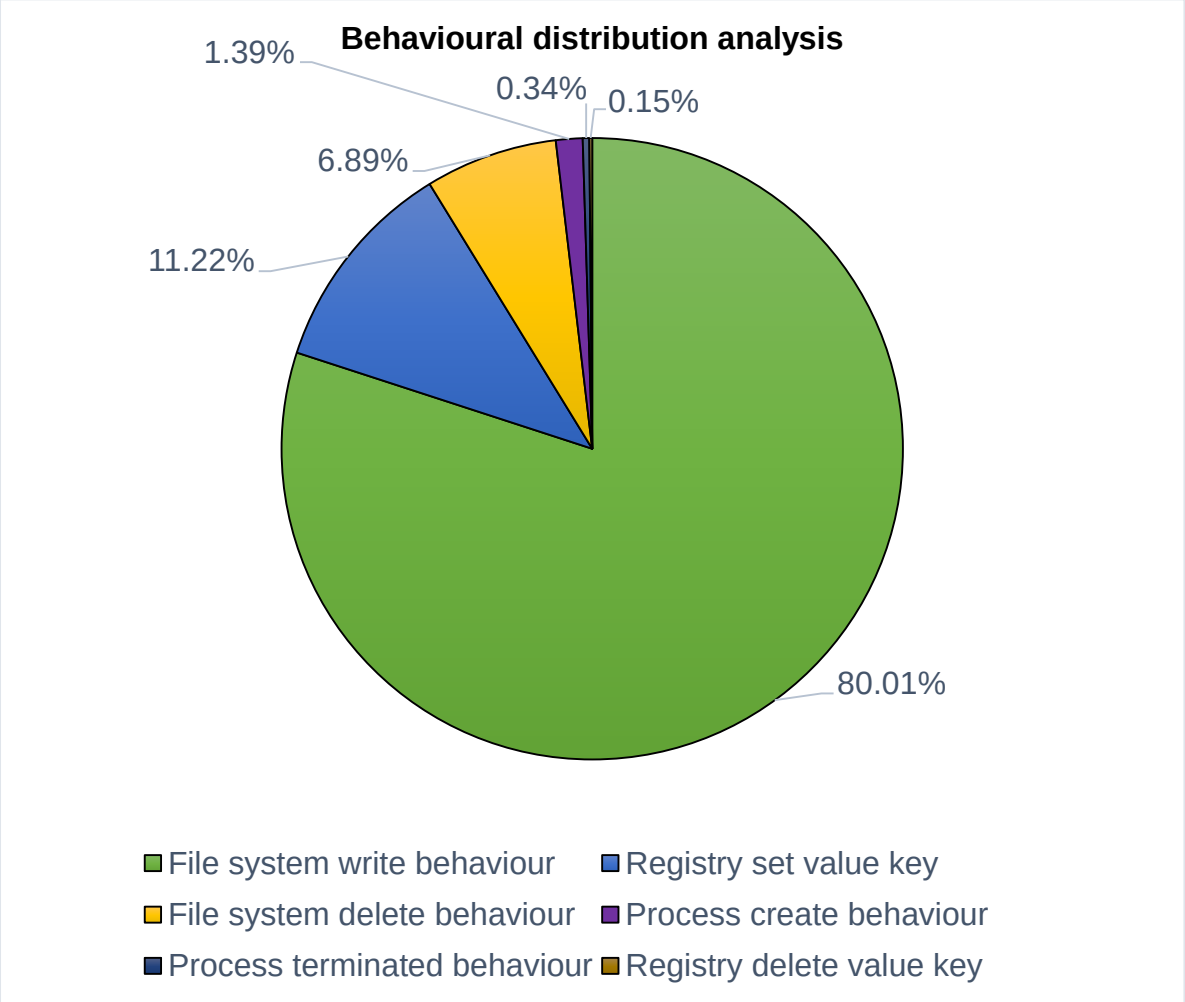


Figure 6.1: Behavioural distribution analysis pie chart.

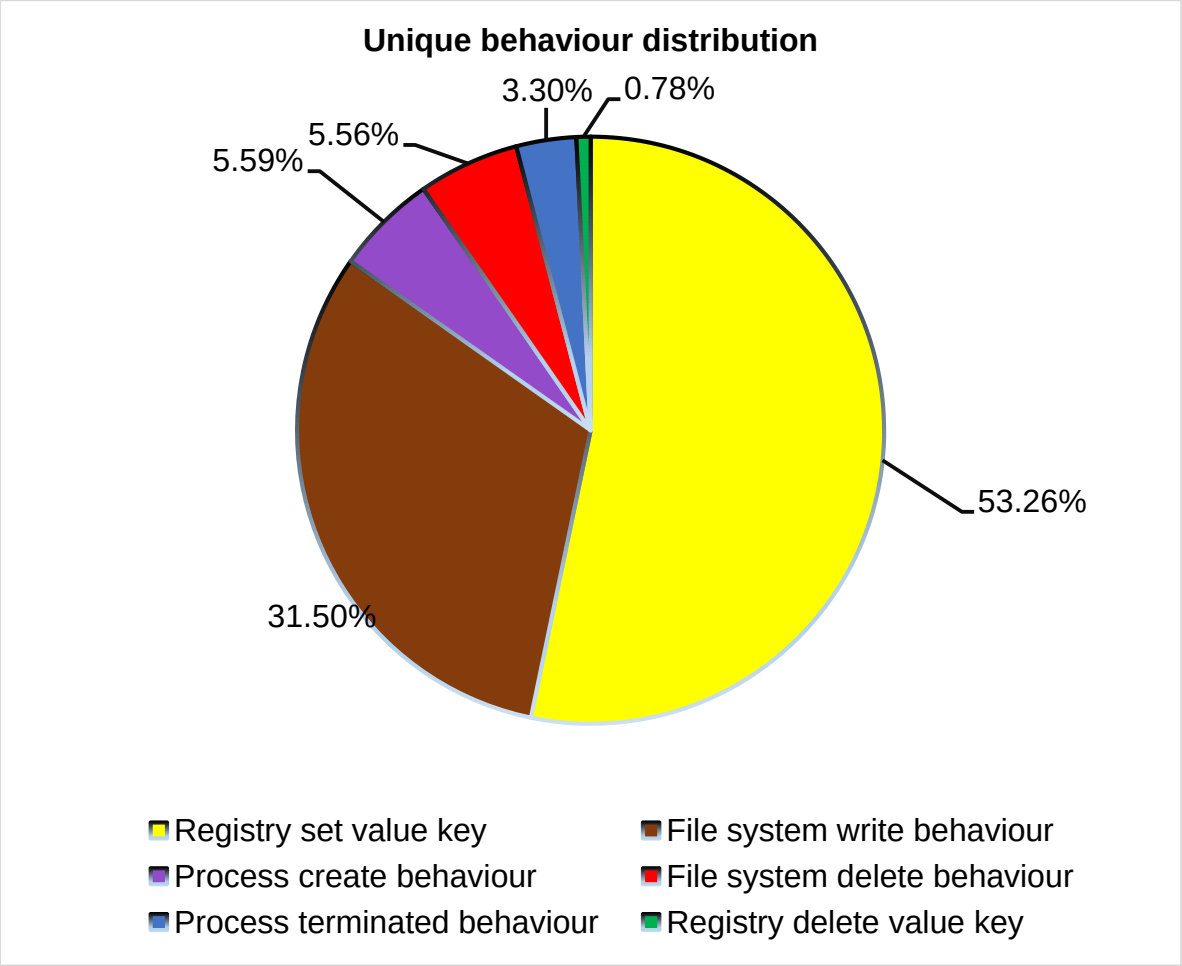


Figure 6.2: Unique behaviour distribution pie chart.

6.1.3 Malicious file writes

Further behavioural log file analysis showed that 116009 / 294864 (39.34%) of all recorded behaviours of visiting real-world malicious websites included the writing of a malicious executable file in the user temp folder. Whilst clearly the Svchost.exe malware was the most common one, there are several other malware samples which seem to be inhibiting the same behaviour and sometimes the only difference observed is merely the name of the drive-by-download executable. Some examples entries are provided in Table 6.1 below.

Table 6.1: Examples of malicious executable exploits being written in the user temp folder of Windows 7.

Malicious executable file write in the user folder	Count
C:\Windows\System32\wscript.exeWriteC:\Users\mp\AppData\Local\Temp\rad959E0.tmp.exe	4326
C:\Windows\System32\wscript.exeWriteC:\Users\mp\AppData\Local\Temp\rad38BFE.tmp.exe	4105
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe	1453
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\PluginFlashPlayer[1].exe	741
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\wrar501ru[1].exe	675

Note: Malware family information provided in Appendix D within table 6A and 6D.

6.1.4 Visual basic exploits within Windows 7

Unless modified or replaced by a malicious alternative, the wscript.exe in the Windows System32 folder is in fact a genuine and non-malicious executable that allows the execution of Visual Basic scripts (.vbs file). In the Windows O.S., this can be done by opening the run tool, typing the name of the Visual Basic (VB) script executable (wscript.exe) followed by the pathway of the script. This has the potential for malicious behaviour within a client system and by using the run method described, this is one way this particular sample could have caused the wscript.exe to execute and download the rad959E0.tmp.exe binary. This type of attack has been seen before as malware which exploit this service start by downloading a .vbs file and perform the execution of the script would cause further malicious binaries to be

downloaded and saved (in this particular case, to the temp folder). This particular attack was observed very rarely with only a total of 67 occurrences in the dataset.

6.1.5 Malicious .bat file exploits

One of the rarer type of attack observed within the data set that exploited file system vulnerabilities was the creation of .bat files in the user's temp folder. This write was performed in two occasions with its prime process being a new and malicious process. The executable file was also saved also in the user's temp folder prior to self-execution. Other malicious behaviours of the same nature to the Svchost.exe and Internet Explorer example discussed at the start can be seen below: the only difference between these behaviour is merely the name of the saved malicious binary, thus showing patterns in malicious drive-by-download behaviours. Whilst protecting against these attacks are beyond the scope of this work, it is important to note that one simple yet seemingly effective way of protecting client systems against these attacks could be to implement strict file monitors within these known vulnerable directories within anti-viruses as a start and preventing the browser from gaining access to executables such as the wscript.exe which allow malware propagation from potentially unseen sources. Some browsers perform protection of client systems using sandboxed techniques when a new file is saved on a client machine and since the evidence found within this thesis reinforces existing knowledge that these drive-by-download attacks are still actively attacking client systems perhaps it is time for more browsers to implement these security enhancements to client systems.

6.1.6 Observed malicious executables manifesting process behaviours

In terms of observed malicious process behaviours, the volume of processes flagged up in Capture-BAT exclusion lists are relatively small in number when compared to

file system or registry behaviour. This was expected as a process is the bridge between the registry or the file system interactions: it is process that initiate behavioural interactions of sending data from an executed computer program to a saved file or registry field. From a distribution perspective one process can be responsible for multiple behavioural interactions, for instance: Internet explorer writing multiple files as a result of visiting a website with a large number of icons and graphic saved in temp folder or internet explorer deleting a large number of expired .tmp files. The number of processes (Portable Executables) within a behaviour analysis environment is limited in comparison to the large number of files that gets changed, hence the one-to-many relationship between process and multiple files or registry entities. The two-main observed behaviour were process creation and process termination. Determining the maliciousness of process behaviours are a lot more straightforward than file changes and much simpler in comparison to registry changes: The initiation of a process in its directory should always be the first point of check. Within the dataset, the vast majority of process creation behaviour observed that were marked as malicious could be identified as malicious due to the process's saved directory. While malware would often be attempting to mask itself by using famous system names, such as svchost.exe, directories such as %SystemDrive%:\Users\USERNAME\AppData\Local\Temp\ are often targeted to be the initiation path of a malicious process.

In terms of Capture-BAT log files, one minor limitation within capture observed was that in the log files, there is a process identifier (ID) field given which often means unique behaviours being generated unnecessarily. This is shared by other Capture-HPC based work, Amorim & Komisarczuk, (2012) also discuss removing the process ID field from behavioural log file. This particular data transformation requirement is

explained in section 5.17 of chapter 5. However, to overcome this minor limitation during the analysis process, it was decided that similar behaviour would be grouped based on executing process path and actual path of executed process: if both these values are identical, the process ID differences would be ignored. This is a different approach to previous work by Amorim & Komisarczuk (2012), where the paths to executables in addition to the process ID in a behavioural interaction are removed features. The justification behind this is simply because the path to a file can often indicate whether a file is benign or malicious based on known operating system defaults. Malicious svchost.exe processes (situated in the dataset at every occurrence) was in the user temp folder when in fact should have been present in the default Windows 7 location, C:\Windows\System32\Svchost.exe. This was being called by a hidden process (Capture-BAT reported it as 'UNKNOWN') and by Internet Explorer. There were 62 process creations in total for this particular behaviour which is relatively low but the fact that this sample had the ability to self-execute within milliseconds of being delivered made it highly dangerous to a client system. We observed a single occurrence with this behaviour where a process termination was called by Internet Explorer after the exploit was carried out.

6.1.7 Observed malicious Dynamic-link library (.dll) and Temporary file (.tmp) process behaviours

Execution of malicious files were not limited to Windows executable files, in the gathered dataset, we observe malware exploiting genuine windows executables and create processes out of dynamic-link library (.dll) files and even temporary (.tmp) files. Files which use the .dll extention often contain code and instructions that are typically executed by an executable. Malicious process behaviour tables for both of these file types are available in Appendix D. It is observed that some malicious .dll

files have the ability to self-execute or hijack the running Internet Explorer process to launch:

Dynamic-link library files maliciously saved in the temp folder self-executing:

- C:\Users\mp\AppData\Local\Temp\fhfy.dll→ created → C:\Users\mp\AppData\Local\Temp\fhfy.dll

---

- C:\Users\mp\AppData\Local\Temp\qhywf.dll→ created → C:\Users\mp\AppData\Local\Temp\qhywf.dll

---

An injected internet explorer browser launches malicious dynamic-link library files:

- C:\Program Files\Internet Explorer\iexplore.exe→ created → C:\Users\mp\AppData\Local\Temp\oydgn.dll

---

- C:\Program Files\Internet Explorer\iexplore.exe→ created → C:\Users\mp\AppData\Local\Temp\fhfy.dll

---

In some cases, it would appear the malicious .tmp files have the ability of calling genuine windows processes such as explorer.exe:

- C:\Users\mp\AppData\Local\Temp\8075.tmp → created → C:\Windows\explorer.exe

---

The compromised web browser manipulating and executing .dll files have been around for a few years as shown by Dhanjani et al. (2009) and finding these malicious behaviours several years after shows that created and detected malware is widely distributed as of the time of writing. Within behaviour analysis, process creation is one of the most powerful exploits for drive-by-downloads as processes, when executed can often allow for a range of malicious activities to take place (such as downloading other executables or files). It was observed in some cases that

malicious processes added keys in the registry. Therefore, should these particular malicious behaviours be scaled, it is evident that those are of the highest priority for identification by a malware analyst.

6.1.8 Exploits targeting scheduled tasks within Windows 7

Other malicious processes discovered targeted other executables such as Schtasks.exe which allows the creation, deletion and modification of scheduled tasks on a client system. We could not find a single justifiable reason why the Schtasks.exe would ever need to be accessed when a system is merely visiting a webpage, especially since Capture-Bat does not have the ability to create such tasks and merely loads up a webpage and records behaviour. Unfortunately, we were unable to capture what manifestation the 'UNKNOWN' process that initiated the modification actually did in the scheduled tasks. It is evident however that this area being targeted is alarming and likely malicious as addition to Schtasks.exe could provide the capabilities of new malicious codes being downloaded periodically or even run malware when the machine is not being used.

6.1.9 Observed registry behaviours within the dataset

With respect to observed registry changes in our Capture-BAT instances we found the largest amount of variance in the data to be with the registry SetValueKey behaviour: this can be seen in Figure 6.2 which shows unique behavioural entry distribution. In terms of malicious behaviours observed, the registry monitor showed the least amount of actual verifiable and quantifiable malicious attacks: The full list of confirmed malicious registry attacks can be seen in section D of the Appendix witin table 6H and 6I. A typical registry attack observed was a malicious file (binary in most cases, but it was noticed that malicious sound (.mp3) files and compromised web browsers that were installed but never used by capture (in this case, Google

Chrome) was adding malicious keys into different sections of the registry such as the sections that saved Internet Explorer data, (location) zone data and shell commands.

6.1.10 False positives within dataset

On the benign side of behavioural registry entries, out of the filtered dataset of 294264 behavioural entries, 2645 were found to have been triggered by Internet Explorer, 6613 were found to have been triggered by Svchost.exe and the rest by other Windows processes such as keys being added by Windows error reporting, Windows software protection, and processes controlling devices (printers and fax – Spooslsv.exe). Upon investigation, it was concluded that none of the unique behavioural entries displayed maliciousness despite their inconsistent manifestation. With respect, to Internet Explorer, the vast majority of this group of observed behaviour were keys being input in the registry which notified of a completed download, Windows error reporting, Shockwave player app data and a large variance within clearable data lists which was saved by Internet Explorer. This was rather unexpected as the browser, Internet Explorer in our case, tends to be one of the highest exploitation vectors within the system as can be seen by the file system exploits where the browser is typically injected and a malicious binary is saved and it seemed reasonable to expect the same with regards to registry exploits.

Svchost.exe (from the default, uncompromised location) performed the vast majority of our captured registry changes within the data set of 5,132 malicious logs (333 unique registry SetValueKey behaviours with a total count of 26483). These behaviours can be grouped as they are not inherently malicious and fairly similar with the difference in many cases of a folder name. For instance, all four of these behaviours are for slightly different registry folder and do not display any inherent malicious behaviour but over time there are alterations to the Object Table folder

where the file ID is stored. This can be seen in table 6.2. It was concluded that these behaviours were part of drifted operating system behaviours over time that were missed by the created exclusion list, thus classed as false positives.

Table 6.2: minor differences between benign behaviours which were not being filtered by our exclusion lists. The behavioural similarity is highlighted.

Similar benign svchost.exe set value key behaviour that exclusion lists missed
C:\Windows\System32\svchost.exe → SetValueKey → \REGISTRY\A\{186A02FD-0C82-11E4-A1B8-080027E9ED53}\DefaultObjectStore\ObjectTable\129\AeFileID
C:\Windows\System32\svchost.exe → SetValueKey →\REGISTRY\A\{92F7356C-1F48-11E4-8EA0-080027E9ED50}\DefaultObjectStore\ObjectTable\129\AeFileID
C:\Windows\System32\svchost.exe → SetValueKey →\REGISTRY\A\{35F43F1C-1F49-11E4-A577-080027E9ED50}\DefaultObjectStore\ObjectTable\129\AeFileID
C:\Windows\System32\svchost.exe → SetValueKey →\REGISTRY\A\{F31127CC-0E00-11E4-9E86-080027E9ED50}\DefaultObjectStore\ObjectTable\129\AeFileID

Similarly, it was observed that System was adding keys in variations of the Windows 7 Network Store Interface (NSI) section of the registry. This service is important for our Capture-BAT machines which need to be connected to the Capture-HPC server. While the count is a steady 70 which is concluded to be on the high end of set registry keys as the average of count per unique behaviour is 6.51. The variations are a single digit difference for example, see Table 6.3.

Table 6.3: Minor differences between benign behaviours which were not being filtered by our exclusion lists. The differences are highlighted.

Similar 'benign' NSI set value key behaviour by system	Count
System → SetValueKey → HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\24\ffffffffffffffffffffffffffff00	70
System → SetValueKey → HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\24\ffffffffffffffffffffffffffff01	70
System → SetValueKey → HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\24\ffffffffffffffffffffffffffff02	70
System → SetValueKey → HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\24\ffffffffffffffffffffffffffff03	70

There are two methods to approach this limitation: the significantly unsafe technique would be to use a wildcard that tells Capture-BAT to ignore the folder name that comes after the REGISTRY\A\ path. The more risk-averse technique and expectedly more exhaustive way to overcome the large volume of SetValueKey behaviours would be to add each of the files as they are observed in the exclusion lists. Regardless of which method utilised, initially when we created the exclusion lists, the system did not display these registry value keys being entered which shows the long term testing requirement that exclusion lists really benefit from. Moreover, this shows one way that behavioural filters require maintenance over time as system

behaviours evolve and change. The registry filter and the large amount of ‘unique behaviour’ is found to be the filter with the highest levels of required maintenances in comparison to the file system, while the process exclusion list is found to be fairly static and low maintenance. This is in line with findings of behavioural filter development in Chapter 4.

6.1.11 False positives and grey behaviours within the dataset and improving creating exclusion lists

Within the dataset there were a number of benign behaviours over the period of two years of analysis: Windows Error Reporting (.wer) file writes contained 98 unique variants with a total count of 2865 (0.97%) where the vast majority were reported by Internet Explorer (97 unique variants written by the web browser) and a single unique entry written by Diagnostic troubleshooting wizard(msdt.exe). The writing of. wer files is relatively safe and it is recommended that wildcards are used to allow these changes in a behavioural filter. Examples for Capture-BAT exclusion lists are provided:

Exclude a .wer file write in file monitor exclusion list as long as file is a .wer file:

```
+    Write  C:\\InsertBenignFullPathHere\\iexplore.exe
       C:\\FullPathToDefaultFolder\\.+wer
```

Exclude a .wer file delete in file monitor exclusion list as long as file is a .wer file:

```
+    Delete      C:\\InsertBenignFullPathHere\\iexplore.exe
       C:\\FullPathToDefaultFolder\\.+wer
```

In the dataset, a small number (29 unique variants of a total count of 87) file writes was created by a genuine Internet Explorer process in the user data folder: (C:\Users\mp\AppData\Roaming\Microsoft\Internet Explorer\UserData\Q9LONY43\)

The written files were in eXtensible Markup Language (XML) format and did not seem inherently malicious by themselves as these files were not executed or seen to perform interactions with other aspects of the file system, registry or processes. These behaviours as they were quite rarely seen were concluded to be grey behaviour which was not added in our created exclusion lists. Additionally, Taskhost.exe was found to be writing a range of differently name .tmp files (29 variants 254 total count) in the ProgramData folder within Windows 7. The ProgramData directory allows different applications to save data which is essentially what Internet Explorer is doing in this particular behaviour. Conversely, this behaviour formed a very small sample within the dataset and appeared infrequently enough to require more information and be marked as grey before being classified benign.

6.2 Case study advanced drive-by-download exploits: Full log file analysis

In this section, the reporting and analysis of a fairly advanced drive-by-download is undertaken. This sample had a very large log file (1025 KB) text file which is slightly less than double the second largest file size (693 KB) and considering 5115 log files out of the possible bank of 5,132 were text files under 100 KB, demanded further investigation. Interesting and non-repeating snippets of the log file will be explored.

This is the data we have regarding this sample on Capture-HPC:

Log identifier: 2052617255

URL: http://fadedboys.com/forum/

Visited on: 24.03.2014 22:24:50.136

Capture Client: 192.168.33.3

Status: malicious

This sample has over 8000 lines of reported malicious behaviours. Sample snippets of interest to a malware analyst from the log file is available in Appendix D.

The vast majority of the log file shows that the drive-by-download is attempting to manipulate a malicious svchost.exe. It is possible to confirm the maliciousness of this log file if we analyse the behavioural information: The svchost.exe in the log file is a new executable in a different directory from the one assigned by Windows 7 which is currently in the user local temp folder: **C:\Users\mp\AppData\Local\Temp**\svchost.exe as opposed to the default directory of: **C:\Windows\system32**svchost.exe. It is important to note at this stage that an executable was downloaded without user consent whilst simply opening the http://fadedboys.com/forum URL. This behaviour alone classifies the interaction as highly malicious.

1. Moreover, the fact that it's being written by the web browser shows that the web browser has been compromised by the malicious web server – as capture analysis does not interact with download-prompt windows when web browsing.
2. It's attempted to mask its true intention by attempting to appear as a genuine windows 7 application and process, Svchost.exe.
3. The malware sample hijacked the Internet Explorer process and executed the malicious Svchost.exe binary which was saved in the User's temp folder.
4. Upon execution the malicious binary opened an unused browser on the machine, Google Chrome and added keys in Chrome's section of the windows registry in addition to the Winlogon\userinit folder. (Userinit is used

by Winlogon to execute logon scripts and stablish network connections). It is likely that the malware writer saved malicious scripts to run at the next logon. Such exploits often lay the foundations for commonly seen ransomware attacks.

5. When the attack was over, internet explorer was used again to terminate the malicious svchost.exe process.

Having explored this sample's drive-by-download behaviour, which had a large number of malicious behaviours raging from over 6000 file creation attempts, registry entries and creation and execution of malicious executables and binaries. In this section this sample is ran through Virus Total and other malware analysis platforms in order to obtain some more insight about this rare sample.

Unfortunately, this file was not analysed or found malicious due to the lacking executables that were corrupted from the drive-by-download collection process in Capture-BAT by Malwr.com.

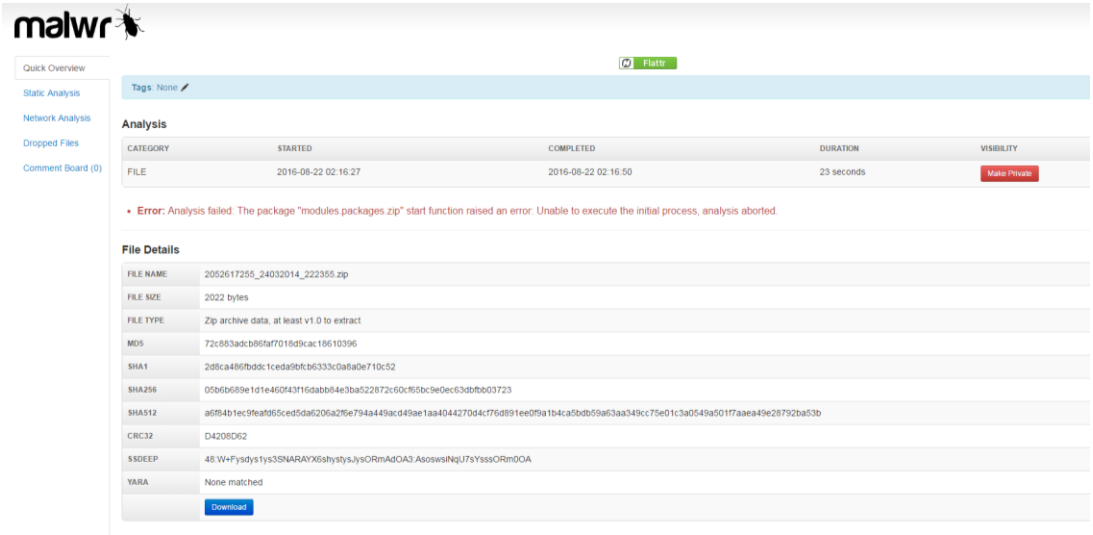


Figure 6.3: Output from Malwr.com in analysing the malicious file captured by Capture-BAT (Based on Cuckoo sandbox). The malicious file is undetected when analysis was attempted.

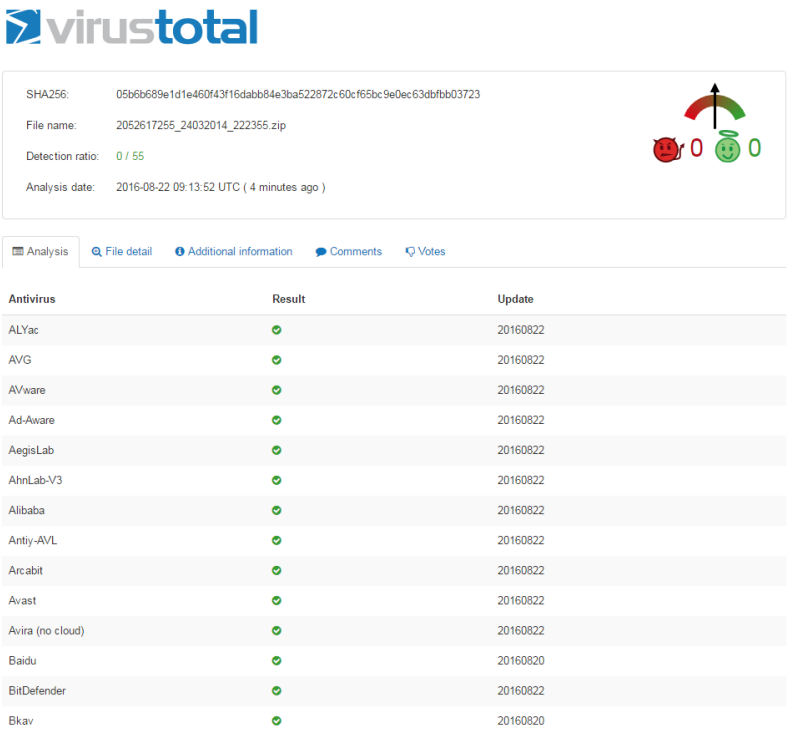
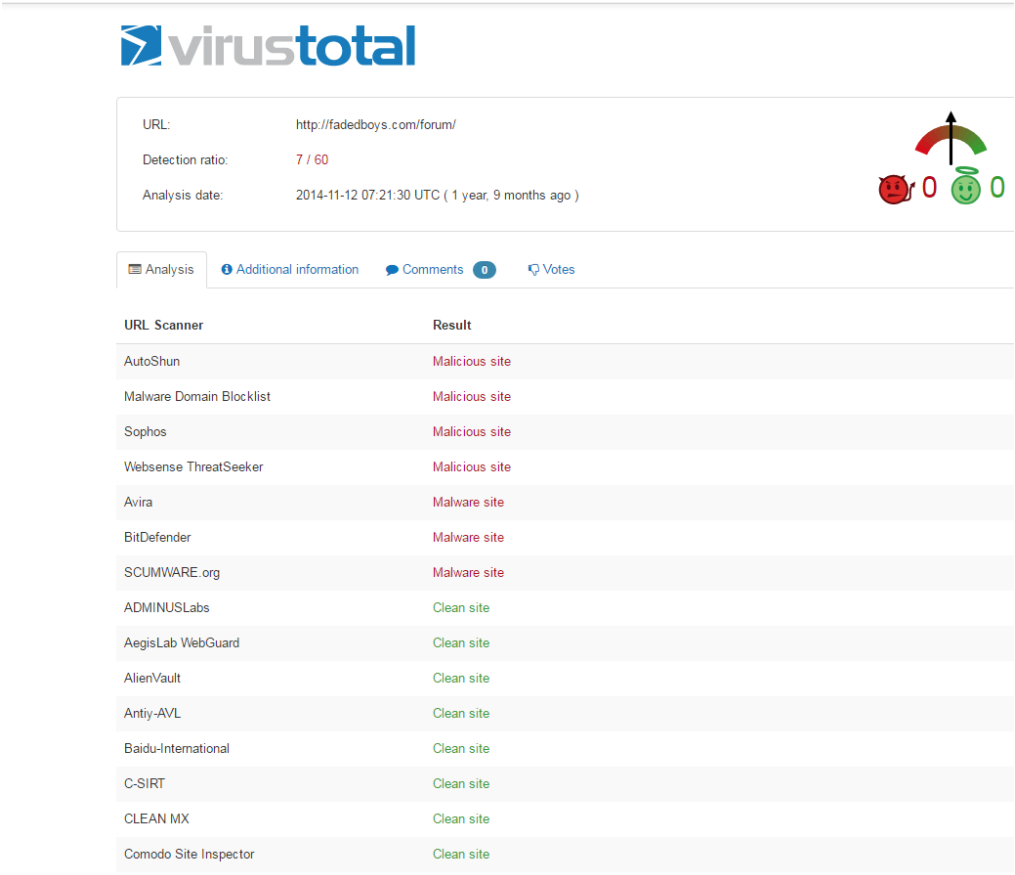


Figure 6.4a: output from VirusTotal.com in analysing the malicious file captured by Capture-BAT (Based on multiple Av-Vendors)

The Virus Total analysis of the malicious webpage is available and can be viewed here:

https://virustotal.com/en/url/0e5cc4ad54db77a1896af795b343c6ff0631a4a5d5f4cc3e874595c53c5addb6/analysis/



virustotal

URL: http://fadedboys.com/forum/

Detection ratio: 7 / 60

Analysis date: 2014-11-12 07:21:30 UTC (1 year, 9 months ago)

0 0

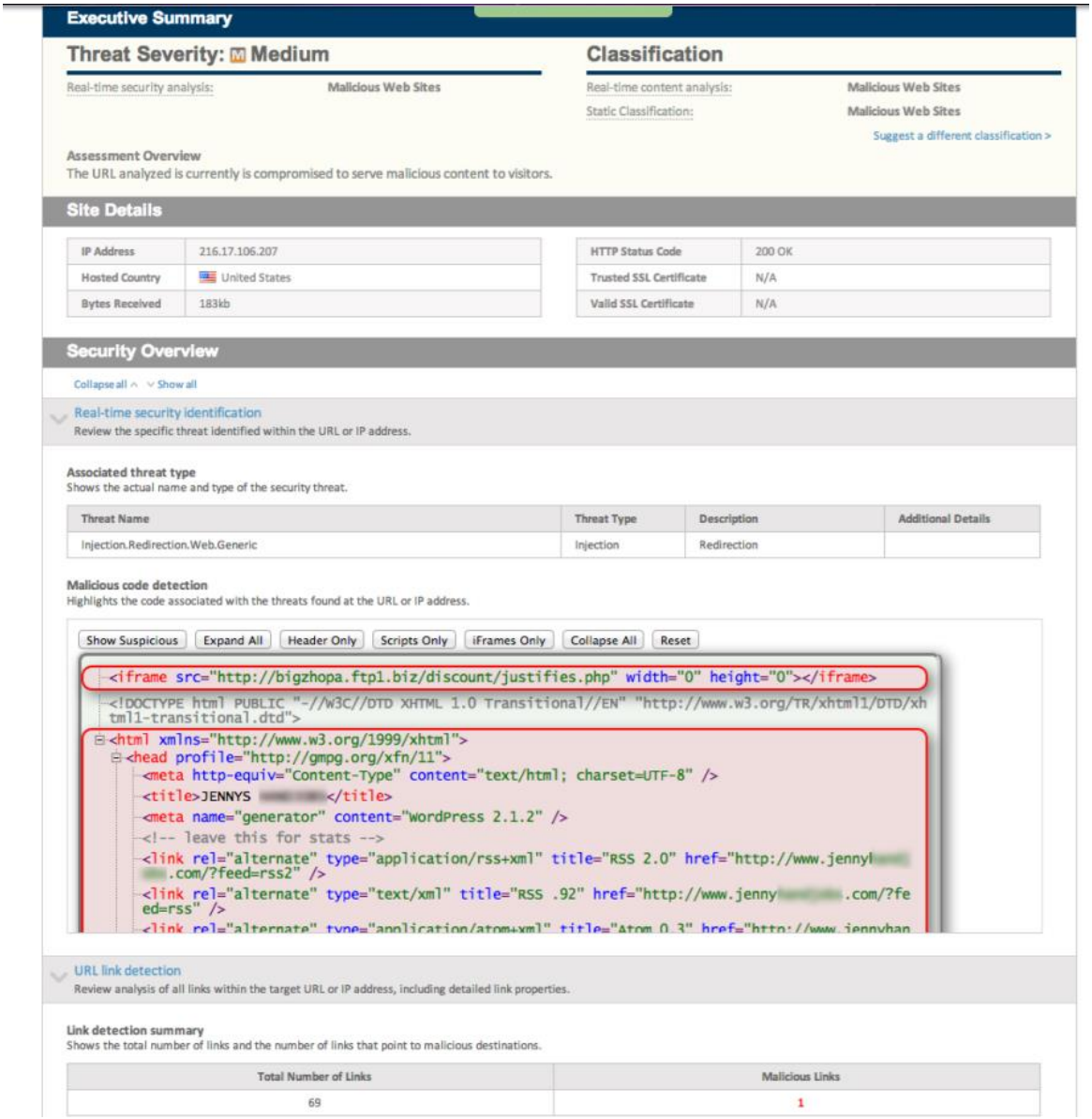
Analysis | Additional information | Comments 0 | Votes

URL Scanner	Result
AutoShun	Malicious site
Malware Domain Blocklist	Malicious site
Sophos	Malicious site
Websense ThreatSeeker	Malicious site
Avira	Malware site
BitDefender	Malware site
SCUMWARE.org	Malware site
ADMINUSLabs	Clean site
AegisLab WebGuard	Clean site
AlienVault	Clean site
Antiy-AVL	Clean site
Baidu-International	Clean site
C-SIRT	Clean site
CLEAN MX	Clean site
Comodo Site Inspector	Clean site

Figure 6.4b: output from VirusTotal.com in analysing the actual URL which was detected by Capture-BAT ac malicious.

As the above figures, 6.3 and 6.4a showed: the malware did not allow any malicious Portable Executables to be captured within the analysis process despite the fact that the malware was observed to have written multiple malicious.exe files within the user directory. In order to verify this was the case, URL analysis was attempted 10 times at different times on the same URL using capture-BAT and the output was identical each time. It was therefore important to verify that this attack. Websense was used to assess the malicious URL and in the malicious code analysis section, an idrame was identified that points to a malicious domain (bigzhopa1.ftp1./discountjustifies.php) as shown in figure 6.5 below. It is possible that the domain was compromised and launches Cross-Site Scripting (XSS) attacks

on client systems which is picked up by the analysis envornment that was used within the study.



6.5: Websense analysis of the Fadedboys webpage showed that an IFrame that points to a malicious website.

6.3. Conclusion

This section initially identified the timestamps in log files as a possible vector to identify trends and patterns from Capture-BAT log files. Having analysed multiple angles of the behavioural interactions contained in Capture-BAT log files, it is fair to

conclude that the dataset gathered contained a significant of malicious behaviours captured while visiting potentially malicious webpages with limited interactions other than opening the URL in the internet browser. A large number of captured behaviour were identified and filtered into unique behavioural entries with frequencies. This allowed the creation of behavioural distribution charts and unique behavioural distribution charts. From this activity, the most popular behavioural vector (file write, process created, registry key) were identified. This allowed the synthesis of knowledge with regards to the most commonly observed attack vectors that targeted and hijacked Internet Explorer 8 within the behaviour analysis environment and caused malicious SvcHost.exe executables to be saved in the user folder within Windows 7. Analysis of log files also allowed us to identify web exploits that executed wscript.exe and exploited malicious .vbs scripts. False positives such as Windows error reporting process triggering and causing a range of unique entries in Capture log files were identified and concluded as a particular process that faces a large amount of behavioural drive over the time that a behaviour analysis lab is used. It was identified that in a given instance malware code downloaded had ability to self-execute malicious code downloaded using an unknown process. .DLL files that were downloaded in the user temp directory were self-executing which is a highly malicious behaviour. Malware targeted features in Windows such as Schtasks.exe to attempt to add ‘launch on boot’ scripts.

Finally, it is important to reflect on some findings that support previous decisions and offer some research pathway suggestions for future research. In chapter 2 where the scope was defined, it was specified that the application of machine learning within this work was out of scope due to this task usually falling part of the next level of analysis within the context of drive-by-download analysis stages. The argument

presented to support this decision is simply because the application of machine learning techniques to the level of data presented was simply another research project in itself due to the high data dimensionality levels. To support this, Table 6J and Log file 6K in Appendix D are presented. Within the context of a single malicious URL analysis in context with the anatomy of drive-by-download presented in chapter 2, figure 2.3 and 2.5, there are two main stages: firstly, the drive-by-download analysis stage which was the focus of this thesis and secondly the malware analysis stage where any dropped Portable Executable is analysed within a sandbox or analysed by anti-virus vendors. Both analysis result in vastly huge dataset dimensions ranging and stripped to the core (by removing unnecessary fields such as time stamp of behaviours, anti-virus meta data, such as date and previous analysis results of the same MD5/Hash files) from actual behaviour entries still result in large dimensional datasets as can be seen in Appendix D tables and log file aforementioned. As discussed in chapter 2, the scope of the detection of malicious webpages is vast due to both the growth in the internet (Internet Live Stats, 2017) and the availability of toolkits and growth in malicious websites (Seifert, 2010; Symantec, 2016; Garnaeva et al. 2016) and if the data dimensionality for a single run-time malicious URL analysis presents this vast level of data, there needs to be further research approaches that are able to analyse run-time log files reasonably fast whilst taking into account the features that are included in the decision making process, much like the created exclusion lists in Chapter 4. In this context, some of these features can include aspects such as: the file path, the downloaded file name and type, the process hijacked, created and behaviours, or registry path exploits, the order of exploit triggering within a system, the vast levels of available anti-virus vendor data or sandbox data from analysis within multiple systems including any

previously assigned signature to a given hash file. This is because a number of aspects need to be considered in the decision making process of what to include for the analysis process.

6.4 Key findings

The Observed malicious behaviour in the 5,132-log file dataset that were identified ofrom a Process, File and Registry perspective, deciphered and evaluated. Within the context of the dataset, malicious attacks are seen to infiltrate the analysis system predominantly through a hijacked browser process which is then exploited to write and execute malicious Portable Executables.

Key process and behavioural interactions analysis providing typical web exploits observed that target Capture-BAT and the Windows 7 operating system. These include Internet explorer which is most commonly observed to be compromised and used to save malicious code, the targeting of Schtasks and Wscript to compromise a system into running scripts.

Behaviour analysis distribution and unique behaviour analysis distribution within the dataset is presented. This identifies the most commonly observed behaviours in each instance and provides knowledge on which vectors within the behaviour analysis lab cause more behavioural entries. In the case of this study, the file system was significantly targeted most often, and the registry seemed to face the highest variety of behaviour.

CHAPTER 7

Conclusions

Chapter 7 – Conclusions

This section concludes the work in the thesis. The set goals of this thesis were to optimise the output from dynamic analysis, detect malicious webpages, identify active attack vectors and understand behavioural analysis environment interactions. An overall conclusion is provided which evolves in chapter-specific conclusions. These conclusions evaluate the extent to which these goals were met. The contributions to knowledge are revisited and a discussion on evaluation, limitations, delimitations and future work concludes the thesis.

7.1 Overall conclusion

The issue of drive-by-download attacks has been around for several years and is a growing problem. This could have been due to the growth in the internet and the volume of websites in addition to the increased availability in exploit kits. Drive-by-download attacks target client systems with the goal of violating confidentiality, integrity and the availability of computer systems. Run-time client honeypots are often used within detection and intelligence gathering to identify drive-by-download attacks and compromised web servers. Within the detection and intelligence gathering scene, there are limitations that are faced: the volume of websites available and the ever-increasing use of malicious web servers by malware organizations within cybercrime can often be viewed as the needles in haystack scenario. The run-time analysis of websites approach has proven to be a viable method of detecting both existing and new threats whilst not being limited to knowing the nature of web-based threats beforehand. However, applying run-time analysis is not without challenges. For instance, the detection accuracy is dependent on the trigger aspect of malware attacks. The high resource and time requirement nature of run-time analysis makes it infeasible for run-time analysis to be the sole approach in

analysing the entire world wide web. Furthermore, the volumes of data that are created during run-time analysis of websites are huge and contain a large percentage of noise. These undesired interactions are generated from an operating system booting, performing regular start-up activities and initiating system calls that are benign. As part of this thesis, a method of finding out, labelling and removing these activities from log files is proposed. This methodology is applicable to future operating systems as the concept of identifying, studying and assessing the potential for malicious remains similar. Within this work, the issue of reducing log file sizes from honeypots was explored and it was proven that a large volume of benign interactions within log files can be removed without compromising detection mechanisms (in the experiments undertaken, there was a reduction of 96.96%). This is highly desirable especially when a security analyst is manually examining interactions. Additionally, it was shown that it is possible to apply a behaviour analyst's expertise within behaviour manifestations to filter out noise within dataset and label goodware datasets. The application of synthesised knowledge from investigating behaviours is desired as behaviours are thoroughly tested and observed before inclusion, with high levels of conviction. As part of the research, it was discovered that over time, some behaviours change exact file paths. This is referred to as behavioural drift and is often found to have very slight variations of older, known behaviours, often a single string in file path or a new folder name. Additionally, core system behaviours within different versions of analysis environments were identified. This proved that different versions of the same operating system could have significantly different behaviour manifestations. This finding shows that there is a requirement for critical thinking in the design aspect of behaviour analysis environments as different versions are subject to different

behaviour manifestations. This contributes to existing knowledge: behaviour analysis environments that are being used within dynamic analysis affect the contents of log files even if the parent operating is kept the same (e.g.: Windows 7). Different versions of the respective operating system displays significant behavioural manifestation. Therefore, when behavioural filters are being created, it is important to take this knowledge into account and apply the filter creation methodology to the same version of operating system that will be used in research or detection purposes. The conclusions for the thesis chapters are provided:

7.1.2 Chapter 4 – Behavioural exclusion list development

Chapter 4 presents the design, development and creation of the study's main artefact, behavioural filters. The chapter introduces the concept of behavioural filters in the form of an existing and well established run-time drive-by-download analysis software, Capture-BAT. The importance and potential of behavioural filters are introduced. Exclusion lists, which serves as behavioural filters for Capture-BAT were defined as low-level filters for separating and capturing behaviours that take place as a result of an analysis environment booting and running. The perspective of behaviours that are relevant to this study included: process, file system and registry interactions. Within this chapter, the rationale for the creation of exclusion lists within the Windows 7 operating system was provided. These include: the gap in research approach and development focus on the exclusion lists within Windows 7. The unexplored potential as log file filters for behavioural filtering and the problem of huge datasets containing large amounts of known benign behaviour. Finally, knowledge on the expected behaviours generated at run-time analysis. These were all of the conclusive factors that justified the creation of the behavioural filter creation methodology and the Capture-BAT exclusion lists.

From this, the chapter explored the three main requirements of successful behavioural filter: Firstly, behavioural filters should be filtering out known benign and confirmed behaviours that have no sign of maliciousness in log files to reduce the amount of log files that need further analysis. Secondly, within a malicious log file, only malicious behaviours should be present. This is in line with the separating goodware from malware datasets prudent malware experimentation guideline by Rossow et al. (2012). Thirdly, created exclusion lists should be done with a risk-averse approach by design therefore, any new behaviour, no matter how similar to expected behaviours should be reported and investigated. This chapter was focused on the observed Windows 7 behaviours. Behaviours are explored thoroughly from process, file and registry interactions to provide typical expected behavioural interaction tables. Behaviours within these tables are evaluated and critiqued in light of their potential to be malicious. This applies the behavioural evaluation expertise from an analyst in the creation of behavioural filters. A list of differential behaviours from Windows XP to Windows 7 is explored. Finally, a development methodology for the creation of behavioural filters is proposed. This is evaluated and exclusion lists for Capture-BAT are implemented using this methodology. This chapter managed to create behavioural filters using the behavioural filter creation methodology whilst answering the principle research question regarding behavioural filters.

7.1.3 Chapter 5 – Behaviour analysis environment experiments

Chapter 5 presents findings of two major experiments undertaken within the study. Both experiments were designed using the adapted prudent experimental framework in chapter 3. In continuation with the development and implementation of behavioural filters within Chapter 4, the created exclusion lists required assessment and evaluation. The first experiment sought to evaluate the efficiency of the created

exclusion list by calculating the actual reduction in behavioural log files whilst conducting the experiment using potentially malicious websites active on the internet. This experiment concluded that filtered log files yielded between 90-98% less behavioural entries than unfiltered log files. This finding meets the requirement of finding out the filter efficiency when tested in the real-world. In addition to this, the number of log files that would be analysed by a malware analyst, was drastically reduced: 112 as opposed to 3,011 in unfiltered datasets. The experimental conclusion therefore was that the application of the behavioural filter was successful in achieving the requirements set out in the previous chapter whilst significantly reducing the noise present in log file datasets. More importantly, this experiment allowed the research question regarding behavioural filter efficiency to be answered thereby, fulfilling the set objective.

The second experiment was tasked at providing understanding how observable benign system behaviours differ in patched and unpatched experimental environments. This would provide key behaviour identifications that differed and would need to be accommodated for in future exclusion lists. It was concluded that a there were a number of key processes and behavioural interaction that differed in patched Windows 7 operating systems thereby answering the research question.

7.1.4 Chapter 6 – Log file analysis and observed Windows 7 attacks

Chapter 6 analyses key behaviours within the analysis of 110,000 potentially malicious websites. These were analysed and a dataset of 5,132 malicious log files was gathered over the course of 4 years. A behavioural distribution analysis initiated the discussion of observed behaviour within the dataset. This is where key system interaction: file writes are concluded to be the most commonly observed behaviour and the registry set value is the behaviour type that offers the most variance and

consequently the most amendments in created filters over time. Captured malicious behaviours within relevant system exploitation were presented and discussed. Common attacks were synthesised from the dataset and explored. This lead to a range of client attacks being identified. False positives within the dataset are presented, discussed and analysed. These answered the research questions on the observed malicious behaviours and attack vector identification that were targeting the honeypot. Finally a case-study of an advanced drive-by-download is presented where the malicious domain seemed to have been compromised by an iframe.

7.2 Contributions to knowledge

From undertaking this study, three contributions to knowledge is made. First, the proposed methodology of developing expert driven behaviour filters. Second the key differences in behaviour from the same operating systems operating at different patches are identified presented. Third the observed malicious behaviour and attacks active on the section of the analysed internet is presented. These contributions are discussed.

7.2.1 Expert driven behavioural filters

This study was focused on an alternate approach to filtering the large volumes of behavioural interaction entries which are captured during run-time analysis and stored in behavioural log files. Detection using dynamic analysis log file datasets contains complicated behavioural interactions which are present in large volumes: log files can contain several thousand entries of system behavioural interactions. As the amounts of malicious web pages on the rise, this would mean that a larger amount websites need to be analysed to detect malicious web pages. The creation and implementation of expert driven behavioural filters is the main contribution of the work that addresses the issue of the large volumes of behavioural interactions in log

files. This is because within this research, the complexity of system behaviours is explored and labelled in behavioural tables leading to the understanding of system interactions. This leads to the creation of expert driven behaviour filters which are designed using the expertise of behavioural interactions. These filters drastically reduce both the size of log files and the number of log files that would need analysis. The behaviour filtration method explored within the study does not rely on the application of machine learning but instead applies the expertise of the security researcher in the design. The expert driven filters were designed at providing an understanding of behavioural interactions from log file datasets before filtering out benign interaction. This is a risk-averse method as required by malware detection mechanisms so that false negatives are as limited as possible. The risk-averseness claim is backed by the design of the behavioural filter: only known behaviours that have been confirmed to be observable and reproducible at boot in a tested, clean and benign system is included in the behavioural filter. New, malicious and unknown behaviour are therefore reported. The filters were implemented for Capture-BAT which performs run-time analysis of a malicious website and outputs the interaction between client and server as a log file. The focus on behavioural filter design within the literature has been very limited and outdated therefore this contribution updates and explores the unexplored area within the literature. Furthermore, the application of expert driven behaviour knowledge to the filtering of run-time analysis log files within the Windows 7 operating system has been another unexplored area of research. In their very nature, behavioural filters required careful considerations at their implementation stage for a given behaviour analysis environment as different operating system versions, applications installed or web-browser used would result in a number of different behaviour manifestation. The implemented behavioural filters

proved to reduce the benign output of behavioural log files by 96.96% when tested in real-world context. It is important to note that the created artefact, were used in academic research (Javed, 2015 and in some future research).

7.2.2 Identifying key behaviours within different versions of behaviour analysis environments

The research identified key significant differences between operating systems that would affect the contents of a log file should the same potentially malicious website be analysed. These observable key differences provide an insight in exactly what behaviour manifestations are different within two operating systems. The knowledge of key behaviours can be used in the decision-making process when designing a behaviour analysis environment when the version of operating systems may matter when hunting specific instances of a given vulnerability. Behaviour analysis environments that are used in run-time analysis were investigated to find observable key system interactions that differentiated from two perspectives:

(1) Predecessor operating system Windows XP was compared with Windows 7 where a range of changes to the operating system architecture and design resulted in a number of different system calls.
(2) A patched and unpatched system which found a number of core processes that differentiated over the course of windows 7 SP1 being released to a fully patched as of September 2016 operating system.

The contribution here were original as the literature within design of behaviour analysis environments did not include investigations or identifications of observable, key differential behaviours. It was found that even the same operating system with varying versions had a number of key behaviours that were different. The overall

conclusion reinforces the identified behavioural filter requirement that filters should be customised on a per-environment basis. This is an important contribution because it shows that analysts need to take design considerations of behaviour analysis environments into account when designing a honeypot or analysis environment. When attempting to detect anomalies within log file analysis using a different operating system with a behavioural filter that was designed for another version may result in false positives and negatives. Falsely flagged benign behaviour as malicious behaviour defeats the purpose of having an optimised log file or more worryingly, filter out potentially malicious content.

7.2.3 Observed malicious behaviour and attack vector insight

The study undertook behavioural analysis of drive-by-download attacks that were recorded using Capture-BAT from the analysis of 110,000 potentially malicious webpages. This minor contribution applied existing analysis environments that were designed for Windows XP to the Windows 7 operating system. Within the context of the literature, Capture-BAT has been applied to a large number of research based on the Windows XP operating system and by performing run-time analysis on a large potentially malicious list of URLs, it was possible to identify a small level of current trends within drive-by-download exploits.

7.3 Evaluation, limitations and delimitations

Having concluded the work and contributions, this section evaluates core aspects of the work and provides insight within limitations experienced and delimitations of the study.

7.3.1 Evaluating Capture-BAT

The tool used in analysing malicious interactions, Capture-BAT was several years old at the start of the study. This could have been an issue if the amount of malicious web servers that attacked client systems were all actively coded to avoid detection within Capture-BAT. However, the study proved that Capture-BAT was still able to detect a number of attacks when adapted to the Windows 7 operating system environment. This conclusion is derived from 2 core datasets:

1 The vast amounts of gathered malicious log files where malicious behaviour was manually verified in context of what is expected from a benign interaction and the potential of maliciousness from a given malicious behaviour.

2. Some captured malware attacks (3,526/5,132) yield modified and created files by the drive-by-download. These were verified and analysed using ~ 54 anti-virus vendors on Virus Total and 29.52% of 3,526 were known malicious samples. The rest of the samples were malicious but not detected by AV vendors.

7.3.2 Limitations of the research

It was experienced that file corruption was a rare but occurring issue when a malicious Portable Executable file was dropped as part of a drive-by-download attack. Malware could have corrupted binary after exploiting a Capture-BAT client. It may be possible for the executables to be corrupted upon zip creation or for instance if Capture-BAT crashes prior to the completion of zipping modified files. This could either be from a run-time error or from the malware's detection avoidance mechanisms. The latter issue can be addressed in future work by applying analysis approaches that seek to avoid detection by malware such as Ether (Dinaburg et al., 2008). Within this study, it was observed that advanced malware samples can often

attempt to corrupt created files but the memory-based exploits are still observable as seen in case study 6.2.

Capture-BAT has been around for a large number of years and it is possible that newer malware samples that detect the presence of analysis environments may avoid detection. The detection of analysis environment can often occur within the context of virtual machines as it is possible to detect virtualised analysis environment features. This could lead to malware not triggering exploits or self-deleting or lack of malware delivery to prevent detection.

Malware samples could be missing vital exploit vectors that are resulted from loopholes and security vulnerabilities of given applications installed on the analysis environment: if these vulnerabilities are not present, tailored malware would be unable to exploit the system fully if at all. Triggering malware manifestation within Capture-BAT may also be hindered by clever malware design. Some malware samples are only triggered after sleep timers are reached. This issue forms part of the limitations of using Capture-BAT. This is because these advanced detection mechanisms are not currently supported. It is likely that should these features be implemented within future research; this issue could be resolved making Capture-BAT more prominent in detecting more attacks. Alternatively, a different yet simpler approach where honeypots are allowed to run for long periods of time may provide similar results.

The typical weaknesses of run-time analysis such as requiring the webpage that is being analysed to actually be malicious at the time of analysis is faced. If a malicious website is not manifesting malicious behaviours, it won't be flagged as malicious during that instance of analysis. It is not uncommon for malicious web servers to be

short-lived and taken offline, modified or migrated thus causing this limitation. Due to the resource and time intensive nature of dynamic analysis, it was a pragmatic decision to only analyse potentially malicious domains once. The included URLs within the created dataset used the ‘ignore duplicate URLs’ function in Capture-HPC. A different decision on the amount of times that a potentially malicious website may have resulted in a lot more malicious samples gathered from the analysed 110,000 URLs however, this was out of scope and is a pathway that can be explored by future research.

Tanaka & Goto (2016) showed that 10% of their 43,000 malicious websites survived for over 500 days and around 10% of their captured sample that involved malicious website with changed behaviour is revived over 15 times. Steps were taken to overcome this issue and overcome the risk the captured dataset consisting only of websites present in the long lived and unchanged category. This was mitigated by analysing potentially malicious websites hours from being released on malicious domain site making the dataset rather current in terms of being analysed and captured soon after being discovered.

7.3.3 Delimitations discussion

The ability to generalise results presented of results relies on various factors. The analysis of behaviours within a behaviour analysis environment may vary if environmental conditions are different. Similarly, it is possible that different types of run-time analysis clients under different versions, patches or applications to be targeted differently. The lack of the targeted exploit plugin being present for instance would lead to malicious webpages avoiding detection. Some of these factors are discussed.

- Chosen operating system: Windows 7 unpatched SP1 system was the operating system of the behaviour analysis environments used in the analysis of web-based threats. The decision to use Windows 7 is justified in chapters 2, 3 and 4. The attacks and artefacts created by this study are focused purely on this version. This is in line with prudent experimental guidelines by Rossow et al. (2012). It is imperative that attacks observed within this study are not generalised. This therefore means that a number of attacks that were observed within his analysis environment are likely to be different to studies conducted within the same web-based threats context that choose to use different analysis environments. This is logical as different versions of analysis environments may have patched existing vulnerabilities or have new vulnerabilities all together meaning a difference in observed malicious manifestation.
- The utilised browser: Internet explorer 8.0 was chosen as the operating browser to visit malicious pages. Internet explorer 8.0 was released in 2009 and at the beginning of the study there were several alternatives, such as Internet explorer 9.0 released in 2011, Mozilla Firefox and google chrome and several others. The decision to use internet 8.0 was a pragmatic decision as the requirements for the study was to have a browser that was supported by Windows 7 and fairly vulnerable to attacks as a study by Rahul (2014) showed this is the most likely browser to face client attacks. Having a browser that facilitates the entry of malware by being more vulnerable was a desirable trait and therefore stands as the justification behind this choice.
- Geo-location of web-page analysis: While the majority of the data was gathered in a university network and thoroughly validated in chapter 3 using

an external source offsite, it may be possible that there were unseen impacts as there was an indirect assumption that was based on the geo-location of the analysis environments.

- Applicability of machine learning: the application of machine learning to log file data sets have been actively researched. Within run-time analysis and the applicability of various machine learning techniques to a dataset with large dimension, there are a number of issues to analysing the log file data. For instance: considerations on the log file features that are weighted more can often affect the way a dataset is clustered. Within this issue, the work presented in the thesis offers an alternative approach that simply takes into account all known behaviours and attempts to filter them out. Despite filtering, the application of machine learning is a complex field where a number of algorithms may be applied to the data to facilitate detection. The choice approach however is a complex decision worthy of another fully dedicated study. The vastness of data for a single analysis was provided in chapter 6 and Appendix D to justify this claim. It may be possible for future research to link and synthesise further information within the context of the drive-by-download attacks by applying machine learning algorithms. This was however well of scope of this work.
- Sample sizes used: a number of samples are used within the experiments carried out following the prudent experimental guidelines. For each experiment written within the work, justifications for appropriate sample sizes are provided. In summary, however: this work had two main focus areas. The first involved the understanding of known, benign, and expected system behaviours. This is applied both in the creation of exclusion lists and the work

related to identifying key behavioural differences within patched and unpatched systems. These experiments, by nature would require a limited number of tests as the requirements were to run analysis on benign webpages until no new 'expected' system behaviours were captured. The second aspect of the work relied on the run-time analysis of malicious attacks from the interaction of client honeypots with potentially malicious webpages. Due to the size of the internet, a larger sample size here was more desirable as it would be more statistically likely that more exploits and more varied attacks would be observed by analysing more malicious URLs. It is fair to say as discussed in chapter 3 that the run-time analysis of 110,000 potentially malicious webpages costing typical rate of 5 minutes per webpage was a sufficient amount in light of other related research (Amorim & Komisarczuk, 2012; Cova et al., 2010; Tanaka & Goto, 2016; Song et al. 2010) to depict a relatively small section of web-based attacks are hosted on malicious servers on the internet.

7.4 Future Work

While undertaking this research, there were a number of potential future work prospects that were identified. These are discussed.

- Behavioural filter extensions and adaptations.
    - The thesis presented and tested a methodology in the identification and labelling of behaviour. This model and labelling feature can be applied to alternative operating systems such as Windows 8, or Windows 10.
- Studies within behavioural difference between analysis environments

    - Differences between patched and unpatched systems can be undertaken in a per-major patch increment approach as described in chapter 5.
- Applicability of analysis techniques to filtered datasets
    - Future research could attempt to use the behavioural knowledge provided in the filters and behavioural tables as part of the data transformation process before applying further analysis techniques. An example could include the usage of the labelled behaviour as part of a semi-supervised learning approach.
- Within the context of honeypots, false negative investigation remains a challenge: it is not possible to identify false negatives when the potential for attacks are not known beforehand.
- As an ongoing issue, run-time analysis is dependent on constant extensions for detecting evasive malicious behaviour. Having discussed limitations to capture, the implementation of bare-metal analysis labs that have an anti-file tampering feature or a memory snapshot approach within analysis may help in the Portable Executable analysis aspect of malware analysis after a given malware is dropped by a malicious website. This would enable further studies to be carried out on advanced malware samples.

7.5 Summary

The internet has been growing rapidly with availability of website deployment facilitating the creation of websites. Web-based threats which pose as high security risks to a client and network system's confidentiality, integrity and availability have been and are expected to continue to rise. Identifying malicious web sites by performing the analysis of drive-by- faces several challenges which are dependent

on approach. Run-time client honeypots are actively used in the detection of web-based threats. They face limitations in the number of the sheer volume of analysis required which is a resource intensive and infeasible task to scan the whole World Wide Web and the volume of data that is generated in behavioural interaction capture. The latter issue of huge datasets additionally contains significant amounts of behaviour that are generated as a result of a system booting and running and can therefore be classed as noise. Multiple examples are provided within that depict the vastness of data that can be obtained on a single exposure to a malicious website. This work sought to address this issue by exploring a method for creating behavioural filters in order to achieve the set goals in Chapter 1. As part of this work, the implementation of behavioural filters to address this issue was carried out. These filters when assessed using prudent practices for malware experiment guidelines, proved to reduce the size of log files by 96.96% in average. A number of behaviour analysis environments are also studied to gather knowledge on operating systems and their sensitivities to fulfil the research objectives around understanding behaviours in analysis environments. Finally this work captures and studies a number of malicious domains and their active attack vectors to provide knowledge on the observable active exploits to fulfil the research objectives on identifying malicious behaviours attacking honeypots and their attack vectors.

GLOSSARY

Glossary

Attack vector – The means by which an external party was able to gain access.

Behaviour analysis environment – The analysis machine containing the behaviour analysis tool that monitors a system's behavioural interactions while attempting to analyse a malicious web page.

Behavioural drift – the change in behavioural interactions observed over time in a behaviour analysis environment. These could be due to new file paths being required or changes in naming conventions.

Behavioural filtering – The filtering process of behavioural log files to separate malicious and anomalous behaviours from benign behaviours.

Behavioural filters – The implemented instance of a particular behaviour vector.

Behavioural log file – A text based file containing behavioural interactions recorded upon analysis of a website in the World Wide Web.

Benign – a non-malicious entity. A behaviour that does not show any signs of misdoings that may result in a system being compromised.

Capture-HPC – The server component of the behaviour analysis tool that is used within this work to manage behaviour analysis clients.

Capture-BAT – The analysis aspect of the Capture software that executes websites and monitors behaviour saves it to log files and sends it back to the server.

Client – the machine that is used by a user to execute a program or service from other computers by using networking. Within this work clients initiate requests to web servers for content.

False negative – An actual malicious behavioural event that is not detected. This is the least desirable outcome within the detection and intelligence gathering spectrum as an attack is missed by the detection mechanism.

False positive – A non-malicious behavioural event that is wrongly flagged as malicious. This is a false alarm that causes noise in a given log file.

Malicious web site – A website that attempts to exploit a client system upon exposure, redirection or visitation.

Prudent experimental guidelines – Framework of experimentation that is designed to provide realistic, transparent, correct and safe malware experimentation.

Patched operating system – An operating system that includes security patches that were released by the operating system's vendor to fix identified vulnerabilities.

Run-time analysis – The analysis of computer software by performing execution of software or websites on a real or virtual processor.

Unpatched operating system – An operating system that does not include any security patches or fixes and contains stock versions of files. This is useful in the design of honeypots that seek to identify older threats that are still active on the World Wide Web.

BIBLIOGRAPHY

Bibliography

1. Akamai. (2013). State of the Internet Report. Q4 2013. Available at: http://www.akamai.com/html/about/press/releases/2011/press_042611.html [Accessed 07/10/2014].
2. Akamai. (2016). State of the Internet Report. Q4 2016. Available at: https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/q4-2016-state-of-the-internet-security-report.pdf [Accessed 20/02/2017].
3. Akiyama, M. Kawakoya, Y. Hariu, T. (2012). Scalable and Performance-Efficient Client Honeypot on High Interaction System, Applications and the Internet (SAINT), 2012 IEEE/IPSJ 12th International Symposium on , vol., no., pp.40,50, 16-20 July 2012.
4. Akiyama, M., Yagi, T., Hariu, T. and Kadobayashi, Y., 2017a. HoneyCirculator: distributing credential honeytoken for introspection of web-based attack cycle. International Journal of Information Security, pp.1-17.
5. Akiyama, M., Yagi, T., Yada, T., Mori, T. and Kadobayashi, Y., 2017b. Analyzing the ecosystem of malicious URL redirection through longitudinal observation from honeypots. Computers & Security.
6. Alosefer, Y. and Rana, O., (2010). April. Honeyware: a web-based low interaction client honeypot. In Software Testing, Verification, and Validation Workshops (ICSTW), 2010 Third International Conference on (pp. 410-417). IEEE.
7. Alwabel, A., Shi, H., Bartlett, G. and Mirkovic, J., (2014). Safe and automated live malware experimentation on public testbeds. In 7th Workshop on Cyber Security Experimentation and Test (CSET 14).

8. Amorim, C., Mirkin, B. (2011). Minkowski metric, feature weighting and anomalous cluster initializing in K-Means clustering. Pattern recognition journal. Vol.45 (3). pp 1061-1065. Available through science direct: http://www.sciencedirect.com/science/article/pii/S0031320311003517.
9. Amorim, R. C. (2012). “Constrained Clustering with Minkowski Weighted K-Means” in 13th IEEE International Symposium on Computational Intelligence and Informatics, Budapest, Hungary, 20-22 November 2012.
10. Amorim, R. C., Komisarczuk, P. (2012) b. “On the Future of Capture-HPC: A Malware Survey”. Technical Report 01/2012, University of West London, 2012.
11. Amorim, C., Komisarczuk P. (2012). On Partitional Clustering of Malware. CyberPatterns. Oxford Brookes, Oxford, 9-10 July 2012. pp. 1-5.
12. Amorim, R. (2015). "Feature Relevance in Wardâ€™s Hierarchical Clustering Using the L p Norm", Journal of Classification, vol. 32, no. 1, pp. 46-62.
13. AV-Test. (2015). Total Malware: last 5 years. [Online]. Available at: http://www.av-test.org/en/statistics/malware/ accessed 04-07-2015
14. B. Lau B., Svajcer. V,. (2010). “Measuring virtual machine detection in malware using DSD tracer”. Journal of Computer Virology Vol. 6(3). 2010 pp. 181–195.
15. Bayer, U., Milani, C., Clemens, H., Christopher, K., Engin, K. (2009). Network and Distributed System Security Symposium (NDSS): Scalable, behavior-based malware clustering. San Diego, California. Feb 2008.
16. Bailey, M., Oberheide, J., Andersen, J., Mao, Z.M., Jahanian, F. & Nazario, J. 2007, "Automated classification and analysis of internet malware", *Recent advances in intrusion detection*Springer, , pp. 178.

17. Belkin, M,. Niyogi, P. (2004). Semi-Supervised Learning on Riemannian Manifolds. Journal of machine learning. Vol.56 (1-30). pp. 209 – 239.
18. BleepingComputer Database. (2016). SPPSVC.EXE Information. Available: http://www.bleepingcomputer.com/startups/sppsvc.exe-25807.html. [21/08/2016, 2016].
19. Bringer, M., Chelmecki, C., Fujinoki, H. (2012). MECS: Computer network and information security: A survey: Recent advances and future trends in honeypot research. [Online] Available at: http://www.mecs-press.org/ijcnis/ijcnis-v4-n10/IJCNIS-V4-N10-7.pdf. [Accessed 07-10-2012] pp 63-75.
20. Burnap, P., Javed, A., Rana, O.F. & Awan, M.S. (2015). "Real-time Classification of Malicious URLs on Twitter Using Machine Activity Data", Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015ACM, New York, NY, USA, pp. 970.
21. Canali, D., Cova, M., Vigna, G., Kruegel, C. (2011) “Prophiler: A fast filter for the large-scale detection of malicious web pages,” in Proceedings of the 20th international conference on World Wide Web, 2011, pp. 197–206.
22. Chiang, M., Mirkin, B. (2009). Intelligent Choice of the Number of Clusters in K-Means Clustering: An Experimental Study with Different Cluster Spreads. Journal of Classification Vol. 27(3). pp. 2-7.
23. Clemenson, C. (2009). Client-side threats and a honeyclient-based defence mechanism, Honeyscout. PhD thesis. Linköping University. Linköping , Sweden.

24. Cova, M., kruegel, C. and vigna, G., (2010). Detection and analysis of drive-by-download attacks and malicious JavaScript code, Proceedings of the 19th international conference on World wide web 2010, ACM, pp. 281-290.
25. Crist, J. (2007). Web Based Attacks. SANS. Available at: http://www.sans.org/reading-room/whitepapers/application/web-based-attacks-2053. [Accessed 07-10-2014]
26. Cuckoo. (2014). Cuckoo Sandbox documentation, [Online]. Available at: http://docs.cuckoosandbox.org/en/latest. [Accessed: 20/12/2014]
27. Curtsinger, C., Livshits, B., Zorn, B., Seifert, C. (2011). “ZOZZLE: fast and precise in-browser JavaScript malware detection,” in USENIX Security Symposium, 2011, pp. 33–48.
28. Dell'Aera, A. (2012). An introduction to honeyclient technologies. [Online]. Available at: www.honeynet.org/files/HPAW2012-Thug.pdf. [Accessed 19/01/13].
29. Dhanjani, N., Rios B. and Hardin, B., (2009). Hacking: The Next Generation: The Next Generation. " O'Reilly Media, Inc.", pp. 104-106.
30. Dinaburg, A., Royal, P., Sharif, M. and Lee, W. (2008). Ether: malware analysis via hardware virtualization extensions. In Proceedings of the 15th ACM conference on Computer and communications security (pp. 51-62). ACM.
31. DODIG-CRNKOVIC, G., (2002). Scientific Methods in Computer Science, Conference for the Promotion of Research in IT at New Universities and at University Colleges in Sweden, April 2002.
32. Egele, M, Wurzinger, P, Kruegel, C, Kirda, E. (2009). Defending Browsers against Drive-by Downloads: Mitigating Heap-spraying Code Injection Attacks,

Secure Systems Lab, 2009, p. Available at: http://www.iseclab.org/papers/driveby.pdf. [Accessed 07-10-2014]

33. Egele, M., Kirda, E. and Kruegel, C., (2009). Mitigating drive-by download attacks: Challenges and open problems. iNetSec 2009–Open Research Problems in Network Security. Springer, pp. 52-62.
34. Egele, M., Scholte, T., Kirda, E. and Kruegel, C., 2012. A survey on automated dynamic malware-analysis techniques and tools. ACM Computing Surveys (CSUR), 44(2), p.6.
35. FBI. (2016) Incidents of ransomware on the rise. [Online] Available at: https://www.fbi.gov/news/stories/2016/april/incidents-of-ransomware-on-the-rise/ [Accessed 12/12/2016].
36. Freitas, R. (2009). MAP-I Seminars Workshop: Scientific Research Methods and Computer Science. Universidade do Porto, Portugal.
37. Garnaeva, M., Sinitsyn,F., Namestnikov, Y., Makrushin, D., Liskin, A. (2016). Overall statistics for 2016. [Online]. Kaspersky. Available at: https://kasperskycontenthub.com/securelist/files/2016/12/Kaspersky_Security_Bulletin_2016_Statistics_ENG.pdf [Accessed 20/02/2017]
38. Guarnieri, C., Tanasi, A., Bremer, J. & Schloesser, M. 2012, The cuckoo sandbox, [Online]. Available at: https://cuckoosandbox.org/ [Accessed 21/01/2017] .
39. Honeynet Project Polish Chapter. (2012). HoneySpider Network Capture-HPC NG. Available at: http://pl.honeynet.org/HoneySpiderNetworkCapture. [Accessed 07-10-2014]
40. HoneySpider. (2013). “Honeyspider Network 2.0.” Available at: http://www.honeyspider.net/docs/Presentation.pdf [Accessed: 21-Mar-2013].

41. Kim, H.G., Kim, D.J., Cho, S.J., Park, M. and Park, M., 2011, November. An efficient visitation algorithm to improve the detection speed of high-interaction client honeypots. In Proceedings of the 2011 ACM Symposium on Research in Applied Computation (pp. 266-271). ACM.
42. Vancouver Internet Live Stats, (2015). Twitter usage statistics. [Online] available at: http://www.internetlivestats.com/twitter-statistics. [Accessed: 02/07/2015]
43. Internet Live Stats, (2017). Total number of Websites. [Online] available at: http://www.internetlivestats.com/total-number-of-websites/. [Accessed: 12/12/2015]
44. Jain, A. (2010). Data clustering: 50 years beyond K-means. Pattern Recognition Letters. Vol.31 (8). June 2010. pp. 651-666.
45. Johns, M. (2008). "On JavaScript Malware and related threats," Journal in Computer Virology, vol. 4, no. 3, pp. 161–178, Dec. 2007.
46. K. Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S,, (2001). "Constrained k-means clustering with background knowledge," in Machine learning-international workshop, 2001, pp. 577–584.
47. Kaspersky Security Bulletin. (2011). "Statistics 2011 - Securelist." Available at: http://www.securelist.com/en/analysis/204792216/Kaspersky_Security_Bulletin_Statistics_2011. [Accessed: 19/03/2013].
48. Kaspersky. (2012). What are Web Threats?. [Online]. Available at: https://usa.kaspersky.com/internet-security-center/threats/web [Accessed 20/02/2017]
49. Kaspersky. (2015). Kaspersky Security Bulletin 2015. Overall statistics for 2015 [Online]/ Available at: https://securelist.com/analysis/kaspersky-security-

bulletin/73038/kaspersky-security-bulletin-2015-overall-statistics-for-2015/ [Accessed 20/02/2017]

50. Khanse. A. (2011). Where are the Windows registry files located in Windows 10/8/7?. [Online]. Available at: http://www.thewindowsclub.com/where-are-the-windows-registry-files-located-in-windows-7 [Accessed: 01/02/-2014].
51. Khattab, S. M, Sangpachatanaruk, C. Mossé, D. Melhem, R. Znati, T. (2004). “Roaming honeypots for mitigating service-level denial-of-service attacks,” in Distributed Computing Systems, 2004. Proceedings. 24th International Conference on, 2004, pp. 328–337.
52. Kindlund. D. (2008). ExclusionLists. [Online]. Available at: https://github.com/dkindlund/capture-hpc/tree/master/ExclusionLists [Accessed 09/09/2016].
53. Koo, T.-M., Chang, H.-C., Hsu, Y.-T., Lin, H.-Y., (2013). Malicious Website Detection Based on Honeypot Systems, in: 2nd International Conference on Advances in Computer Science and Engineering (CSE 2013). Atlantis Press.
54. Kolter, J., Maloof, M., (2004)., August. Learning to detect malicious executables in the wild. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 470-478). ACM.
55. Lau, B., Svajcer, V. (2010). Measuring virtual machine detection in malware using DSD tracer. Journal of Computer Virology Vol. 6(3). pp. 181–195.
56. Le, V. L., Welch, I., Gao, X., Komisarczuk, P. (2011a). “Two-Stage Classification Model to Detect Malicious Web Pages,” 2011, pp. 113–120.
57. Le, V.L., Welch, I., Gao, X. & Komisarczuk, P. (2011b). ," Advanced Information Networking and Applications (AINA), 2011 IEEE International

Conference: Two-Stage Classification Model to Detect Malicious Web Pages. Biopolis, Singapore, 22-25 March 2011 pp.113-120.

58. Le, V.L., Welch, I., Gao, X. & Komisarczuk, P. (2013). "Anatomy of drive-by download attack", Proceedings of the Eleventh Australasian Information Security Conference-Volume 138. Australian Computer Society. pp. 49-58.
59. Li, P., Liu, L., Gao, D. & Reiter, M.K. (2010). "On challenges in evaluating malware clustering", International Workshop on Recent Advances in Intrusion DetectionSpringer, , pp. 238.
60. Likarish, P., Jung, E., Jo, I. (2009). “Obfuscated malicious javascript detection using classification techniques,” in Malicious and Unwanted Software (MALWARE), 2009 4th International Conference on, 2009, pp. 47–54.
61. Mahootian, F., Eastman, T. (2009). Complimentary frameworks of scientific inquiry: hypothetico-deductive, hypothetico-inductive and observational-inductive. World futures. Vol. 65. pp.61-75.
62. Mansoori, M., Welch, I. & Fu, Q. (2014). "YALIH, yet another low interaction honeyclient", Proceedings of the Twelfth Australasian Information Security Conference-Volume 149Australian Computer Society, Inc., , pp. 7.
63. Mansoori, M., Welch, I., Choo, K.K.R. and Maxion, R.A., 2016, March. Application of hazop to the design of cyber security experiments. In Advanced Information Networking and Applications (AINA), 2016 IEEE 30th International Conference on (pp. 790-799). IEEE.
64. Mansoori, M., 2017. Localisation of Attacks, Combating Browser-Based Geo-Information and IP Tracking Attacks.
65. Mansoori, M., Welch, I., Choo, K.K.R., Maxion, R.A. and Hashemi, S.E., 2017. Real-world IP and network tracking measurement study of malicious websites

with HAZOP. International Journal of Computers and Applications, 39(2), pp.106-121.

66. Maxion, R.A., Longstaff, T.A. and McHugh, J., 2010, September. Why is there no science in cyber science?: a panel discussion at NSPW 2010. In Proceedings of the 2010 workshop on New security paradigms (pp. 1-6). ACM.
67. Microsoft Documentation. (2016), Shell Constants, Enumerations, and Flags: CSIDL_COMMON_APPDATAFOLDERID_ProgramData [Homepage of CSIDL_COMMON_APPDATA FOLDERID_ProgramData], [Online]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/bb762494.aspx [21/08/2016, 2016].
68. Mirkin, B. (2011). “Clustering for data mining: A data recovery Approach”. New York: Chapman & Hall/CRC, 2011, p. 264.
69. Mogren, O., 2017. Malicious JavaScript detection using machine learning. learning, 10(11), p.12.
70. Moser, A., Kruegel, C. and Kirda, E., (2007), Exploring multiple execution paths for malware analysis. In Security and Privacy, 2007. SP'07. IEEE Symposium on (pp. 231-245). IEEE. Namestnikov, Y. (2011). Kaspersky Security Bulletin. Statistics 2011. [Online]. Available at: http://www.securelist.com/en/analysis/204792216/Kaspersky_Security_Bulletin_Statistics_2011. [Accessed 13/01/2013].
71. Murdoch, D., 2016. Blue Team handbook: incident response edition: a condensed field guide for the cyber security incident responder. CreateSpace Independent Publishing. Vancouver.

72. Nazario, J. (2006), Long-lived-malware-distribution-sites [Homepage of Arbornetworks], [Online]. Available: https://www.arbornetworks.com/blog/asert/long-lived-malware-distribution-sites/ Accessed [12/14/2016].
73. Net Market Share. (2016). Desktop Operating System Market Share [Online]. Available at: https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=10&qpcustomd=0 [Accessed 31/12/2016].
74. Nominet UK. (2017). Security at Nominet. [Online]. Available at: https://www.nominet.uk/about/security-at-nominet/ [Accessed 30/03/217]
75. O'Gorman, G. and McDonald, G., (2012). Ransomware: a growing menace. Symantec Corporation.
76. Perdisci, R., Lee, W. & Feamster, N. (2010), "Behavioral Clustering of HTTP-based Malware and Signature Generation Using Malicious Network Traces", *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation*USENIX Association, Berkeley, CA, USA, pp. 26.
77. Peter, E. and Schiller, T., (2011). A practical guide to honeypots. Washington University.
78. Provos, N., McNamee, D., Mavrommatis, P., Wang, K. & Modadugu, N. (2007). "The Ghost in the Browser: Analysis of Web-based Malware.", HotBots, vol. 7, pp. 4-4.
79. Puttaroo, M, Komisarczuk, P, Amorim, R., (2014). Challenges in developing Capture-HPC exclusion lists, In The 7th International Conference on Security of Information and Networks, (SINCONF), 9-11 September 2014, ACM
80. Puttaroo, M., Komisarczuk, P., Amorim, R., (2013). On Drive-by-Download Attacks and Malware Classification. Fifth International Conference on Internet

Technologies & Applications (ITA), Wrexham, Wales, 10 to 13 September 2013

81. Qassrawi, M. T. Zhang, H. (2010). Client honeypots: Approaches and challenges. In New Trends in Information Science and Service Science (NISS), 2010 4th International Conference on, 2010, pp. 19–25.
82. Riden, J. & Seifert, C. (2010). A Guide to Different Kinds of Honeypots [Homepage of symantec], [Online]. Available: https://www.symantec.com/connect/articles/guide-different-kinds-honeypots [2016, 12/01].
83. Reed, M. (2011). VirtualBox 4.1 Introduces Cloning Amongst Other Features. [Online] Available at; http://www.linuxjournal.com/content/virtualbox-41-introduces-cloning-amongst-other-features. [Accessed 08-10-2014]
84. Rieck, K., Holz, T., WIllems, C., D¨ussel, P., Laskov, P. (2008). “Learning and classification of malware behaviour” In Detection of intrusions and malware vulnerability assessment. 2008. pp 264.
85. Rieck, K., trinius, P., willems, C. and holz, T., (2011). Automatic analysis of malware behavior using machine learning. Journal of Computer Security, 19(4), pp. 639-668.
86. Rivera, J. Meulen, R.V. (2013). Gartner Says 25 Percent of Distributed Denial of Services Attacks in 2013 Will Be Application-Based. Available at: http://www.gartner.com/newsroom/id/2344217. [Accessed 07-10-2014]
87. Rokach, L. & Maimon, O. (2005). "Clustering methods" in Data mining and knowledge discovery handbook Springer, pp. 321-352.

88. Tian, R., Islam, R., Batten, L. & Versteeg, S. (2010). "Differentiating malware from cleanware using behavioural analysis", Malicious and Unwanted Software (MALWARE), 2010 5th International Conference onIEEE, , pp. 23.
89. Rossow, C., Dietrich, C.J., Grier, C., Kreibich, C., Paxson, V., Pohlmann, N., Bos, H. and Van Steen, M., (2012). Prudent practices for designing malware experiments: Status quo and outlook, 2012 IEEE Symposium on Security and Privacy 2012, IEEE, pp. 65-79.
90. Rahul, R. (2014). Internet Explorer the “Most Vulnerable” Web Browser of 2014, States Study: Time to Switch to Chrome and Firefox? [Online]. Available at: http://www.ibtimes.co.uk/internet-explorer-most-vulnerable-web-browser-2014-states-study-time-switch-chrome-1458400 [Accessed: 03/10/14].
91. Seifert, C,. (2010). Cost-effective detection of drive-by-download attacks with hybrid client honeypots. Ph.D. Thesis, Computer Science Department, Victoria University of Wellington, 2010.
92. Seifert, C., Welch I., Komisarczuk, P. (2006). Taxonomy of Honeypots. CS Technical Report TR-06-12, School of Mathematics, Statistics and Computer Science, Victoria University of Wellington, New Zealand. Available at: http://www.mcs.vuw.ac.nz/comp/Publications/archive/CS-TR-06/CS-TR-06-12.pdf. [Accessed: 05-10-2012].
93. Seifert, C., Welch, I., Komisarczuk, P. (2008). “Identification of malicious web pages with static heuristics,” in Telecommunication Networks and Applications Conference, 2008. ATNAC 2008. Australasian, 2008, pp. 91–96.
94. Seifert, C., Welch, I., Komisarczuk, P., Aval, C.U. & Endicott-Popovsky, B. (2008b), "Identification of malicious web pages through analysis of underlying DNS and web server relationships.", LCNCiteseer, , pp. 935.

95. Seifert, C., Steenson, R., Welch, I., Komisarczuk, P. and Endicott-popovsky, B., (2007). Capture – A behavioral analysis tool for applications and documents. Digital Investigation, 4, Supplement, pp. 23-30.
96. Seifert, C. & Steenson, R. 2006, "Capture - Honeypot Client (Capture-HPC)", [Online]. Available at: https://projects.honeynet.org/capture-hpc ; Last updated 22 September 2008. [Accessed 07-10-2015]
97. Shadowserver Foundation. (2014). - Information - Honeypots". Available at: http://www.shadowserver.org/wiki/pmwiki.php/Information/Honeypots. [Accessed 07-10-2014]
98. "Shadowserver Foundation (2017) - AV – Virus Daily Stats." Available at: http://www.shadowserver.org/wiki/pmwiki.php/AV/VirusDailyStats. [Accessed: 22-Mar-2013].
99. Khattab, S.M., Sangpachatanaruk, C., Mossé, D., Melhem, R. and Znati, T., 2004. Roaming honeypots for mitigating service-level denial-of-service attacks. In Distributed Computing Systems, 2004. Proceedings. 24th International Conference on (pp. 328-337). IEEE.
100. Vancouver Sjoberg, D., Dyba, T., Jorgensen, M. (2007). Future of Software Engineering. FOSE '07: The Future of Empirical Methods in Software Engineering Research. Washington, DC, US: IEEE Computer Society. pp. 360–377.
101. Song, C., Zhuge, J,. Han, X., Ye., Z. (2010). Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security: Preventing Drive-by Download via Inter-Module Communication Monitoring. April 13–16, 2010, Beijing, China. pp. 124-133.

102. Sowriraghavan, A. & Burnap, P. (2015). "Prediction of Malware Propagation and Links Within Communities in Social Media Based Events", Proceedings of the ACM Web Science ConferenceACM, New York, NY, USA, pp. 59:1.

103. Stange, S. (2015). Detecting malware across operating systems. [Online]. Available at: https://www.opswat.com/blog/detecting-malware-across-operating-systems [Accessed 28/12/2016].

104. Strickland, J., Chandler, N. (2014). “What are Tweets?” [Online], Available at: http://computer.howstuffworks.com/internet/social-networking/networks/twitter1.htm. [Accessed 09-10-14].

105. Symantec. (2016). “Internet Security Threat Report”. [Online]. Available at: http://images.mktgassets.symantec.com/Web/Symantec/%7B057420b3-8f47-4832-b85d-18e631aad295%7D_ISTR_21_MAIN_REPORT_SOCIAL_-_21365088.compressed.pdf [Accessed 13/01/17].

106. Tanaka, Y. & Goto, A. (2016). "Analysis of malware download sites by focusing on time series variation of malware", 2016 IEEE Symposium on Computers and Communication (ISCC)IEEE, , pp. 173.

107. The Honeynet Project. (2014). Projects Overview. Available at: http://www.honeynet.org/project. [Accessed 07-10-2014]

108. The Linux Information Project. (2006). June 12 2006, 2006-last update, Processes: A Brief Introduction. Available: http://www.linfo.org/process.html [Accessed: 28/08/2016].

109. Upguard. (2016). Write Once, Infect Anywhere, or: The Rise of Cross-platform Malware. [Online]. Available at: https://www.upguard.com/blog/write-

once-infect-anywhere-or-the-rise-of-cross-platform-malware [Accessed: 28/12/2016]

110. Vasilescu, M., Gheorghe, L., Tapus,N.(2014). Practical malware analysis based on sandboxing, RoEduNet Conference 13th Edition: Networking in Education and Research Joint Event RENAM 8th Conference, Chisinau, Moldova, September 11-13 2014 pp 1-4

111. Virus Bulletin. (2015). Packer. [Online]. Available: https://www.virusbtn.com/resources/glossary/packer.xml [Accessed: 27/10/2015].

112. Virus Total. (2016). “Free Online Virus, Malware and URL Scanner.” Available at: https://www.virustotal.com/en/. [Accessed: 19-Mar-2013].

113. Wagener, G. and Dulaunoy, A., (2008). Malware behaviour analysis. Journal in computer virology, 4(4), pp.279-287.

114. Wei Y., Zheng Z., & Ansari, N. 2008, "Revealing Packed Malware", Security & Privacy, IEEE, vol. 6, no. 5, pp. 65-69.

115. Willems, C., Holz, T. and Freiling, F., (2007). Toward automated dynamic malware analysis using cwsandbox. IEEE Security and Privacy, 5(2), pp.32-39.

116. WC3. (2014). OS Statistics. Available at: http://www.w3schools.com/browsers/browsers_os.asp. [Accessed 07-10-2014]

117. Wirzenius, L., OJA, J., Stafford, S. and Weeks, A., (2000). Linux System Administrator's Guide.

118. Ye, Y., Li, T., Jiang, Q. and Wang, Y., (2010). CIMDS: adapting postprocessing techniques of associative classification for malware detection.

IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 40(3), pp.298-307.

119. Zeltser, L., (2016). Mastering 4 Stages of Malware Analysis. Available: https://zeltser.com/mastering-4-stages-of-malware-analysis. [08/22, 2016].

120. Zorabedian, J. (2014). How malware works: Anatomy of a drive-by download web attack [Homepage of Sophos], [Online]. Available: https://blogs.sophos.com/2014/03/26/how-malware-works-anatomy-of-a-drive-by-download-web-attack-infographic. [2015, 10/02/2016].

APPENDIX.

Appendix

Appendix A

Paper 2A: Challenges in the development of Capture-HPC exclusion lists.

Challenges in developing Capture-HPC exclusion lists.

Mohammad Puttaroo
University of West London
St Mary's Road, Ealing
London, W5 5RF
+44 1895 463908
mohammad.puttaroo@uwl.ac.uk

Peter Komisarczuk
University of West London
St Mary's Road, Ealing
London, W5 5RF
+44 20 2820 0256
peter.komisarczuk@uwl.ac.uk

Renato Cordeiro de Amorim
University of Hertfordshire
College Lane, Hatfield
AL10 9AB
+44 1707 284000
r.amorim@herts.ac.uk

ABSTRACT

In this paper we discuss the challenges faced whilst developing exclusion lists for the high-interaction client honeypot, Capture-HPC. Exclusion lists are Capture client system behaviours which are used in the decision making process when determining if a particular behaviour is malicious or benign. As exclusion lists are the main decision making method used by Capture-HPC to classify a given webpage as benign or malicious, we identify a number of issues with current research which are often overlooked. Exclusion lists by nature require constant updating as they are developed to meet the specific requirements of a particular operating system, web browser and application system environment. Any changes to these would mean the possibility of a given client to display different benign behaviour which consequently means new exclusions required. As a result of their specific version requirements, exclusion lists are not transferable from clients. We propose a set of recommendations to aid in the creation of exclusion lists. We also present and discuss some common drive-by-download attacks which we have captured using our Windows 7 compatible exclusion lists.

Categories and Subject Descriptors

D.3.3 [**Programming Languages**]: Invasive software (e.g., viruses, worms, Trojan horses)

General Terms

Performance, Reliability, Experimentation

Keywords

High-interaction, Client honeypot, exclusion lists, Capture-bat, classification, malware, drive-by-download.

1. INTRODUCTION

Malicious web browser based attacks, also known as drive-by-download attacks, have grown in popularity in the recent years [2][4][6][7][9][15][16]. A drive-by-download attack is defined as a malicious change in a client system without the user's consent and knowledge as a result of visiting a malicious website. The change here can include the downloading and running of a malicious executable without any interaction or prompt with the user. The malicious website would push back requested content in addition to malicious code. Compromisation of the client's browser, applications or Operating System (O.S.) can also occur if a client is visiting a benign website with poisoned advertisement streams from other malicious webservers in sequences of redirects. These attacks are often driven by monetary profits earned by organised crime [11]. A compromised client effectively becomes "owned" by the malware writer and data contained on the client's machine such as saved credit card information, saved forms from the browser, files on the machine or network effectively become available to the malware writer [13]. In addition to this, a compromised machine may become part of botnets and be used to launch Distributed Denial of Service (DDoS) attacks for the malware writer [9]. Malicious websites usually exploit vulnerabilities in the web browser and plugins installed (e.g.: IE, Firefox and ActiveX), operating system (e.g.: Windows, Linux) and applications (e.g.: PDF reader, office applications) installed on a client machine [5].

SIN '14, September 09 - 11 2014, Glasgow, Scotland UK

ACM 978-1-4503-3033-6/14/09…$15.00.
http://dx.doi.org/10.1145/2659651.2659717

2. BACKGROUND

2.1 Drive-by-downloads

The issue of drive-by-downloads is currently one of the major challenges the cyber security industry faces. In 2010 there was an estimated 150 Million malicious websites which would attempt to launch drive-by-download attacks [6]. This is likely to have grown in number as a result of cheaper webhosting available such as Amazon EC2. Research by the Gartner group suggests that 70% of attacks occur from the application layer of the OSI model [4]. This shows how the web is being exploited to target client systems. Additionally Gartner predicted in 2013 that 25% of DDoS attacks would be application based [4][12][15]. This was somewhat confirmed by Akamai's state of the internet report which stated an "increase of 54% in DDoS attacks in the first quarter of 2013" alone [1]. Exploited machines are also used to send spam. It is estimated that 10,000 new malware samples are discovered on a daily basis [11]. These statistics show that the drive-by-download attack poses a highly significant threat to computer and network systems by exploiting vulnerabilities in client systems. Therefore there is a need for more reliable security mechanisms to defend client machines in order to defend computer and network systems.

2.2 Client honeypots

A honeypot is a closely monitored security device with the intention of getting probed and compromised by malware. Unlike typical server honeypots which passively wait for attacks, client honeypots actively crawl the World Wide Web in search of malicious webservers [2]. The client honeypot would initiate a HTTP request using a specified web browser. As this interaction occurs, any changes in client state as well as packets are logged

and saved [16]. Figure 1 illustrates client honeypots and server interactions.

Client honeypots can be a low-interaction which performs static analysis using emulated systems. Examples of this include: Thug, HoneyD and Dionea [17][18]. High-interaction client honeypot perform dynamic analysis on websites using real operating systems, browsers and applications. Log files are generated containing modifications performed to the registry, file system, processes and network connections. Examples include Capture-HPC/Capture-BAT and Cuckoo sandbox [18]. Hybrid client honeypots attempt to combine the benefits of low-interaction and high-interaction client honeypots to detect malware. Examples include: HoneySpider 2 and Honeybird [17][18].

2.3 Capture-HPC

We are currently using an open-source client honeypot, Capture-HPC. As mentioned before Capture-HPC is a high interaction client honeypot which performs dynamic data analysis in an actual system. Capture-HPC has two main comportments: a Capture-HPC server and a number of Capture clients; commands can be issued from the Capture-HPC server for the Capture clients that will run a specified O.S. and browser. The Capture client is where the behaviour analysis takes place. In order to make the decision whether a change in the system is malicious or benign, an exclusion list is used [10][14]. An exclusion list is a set of known system behavioural events that Capture-HPC will include or exclude in the generation of log files. After each website is analysed the Capture-HPC server reverts all changes made to the Capture client to a default state. Capture-HPC was chosen as our high-interaction behaviour analysis tool as the output (log files for registry, file system and process) as this meets our requirement of clustering malware discussed in our previous research [3] [13].

It is important to note that we are trying to simulate the real life experience of a user attempting to send request to a potentially malicious website, in order to identify malicious behaviour from malicious websites. A significant aspect of this therefore requires the Capture client to be setup accordingly. Effectively this means setting up a Windows 7 client as the most popular O.S. currently according to the O.S. statistics provided by [19]. Previous research [3][7] have used Windows XP as the Capture-Client O.S. Unfortunately this means specifically targeted Windows 7 exploits might be missed. In addition to this, the usage of popular browsers and applications (such as Adobe flash player to view media) which help display and provide functionality of websites is also used for this experimental setup to have a fairly matching Capture client simulating an actual user's client.

2.4 Motivation

The main motivation behind this paper is the lack of focus on Capture-HPC exclusion lists in published researched papers. Exclusion lists are overlooked yet are a key comportment in decision making for malicious or benign classification of a webpage. The issues of using an exclusion list that isn't created based on the specific requirements of the Capture client in question would result in a number of false positives and false negatives. This is so because Capture would could flag up a number of behaviours which are not defined in the exclusion list as malicious or benign. The main drive behind this paper is to address the need for Capture-HPC log files used in malware research to be as accurate as possible. The maintenance and the non-transferable nature of exclusion list results in a number of challenges on the security researcher which are explored in section 2.5.2 and 2.5.3.

2.5 Experimental setup

In our two instances of Capture, we currently run Capture on Windows 7 using Firefox and IE as browsers for our virtual machines. The data currently obtained from Capture-HPC is divided between process, registry and file system as log files. The current limitations of the systems running include no network API monitoring and the inability to detect and capture ActiveX exploits as we don't have the ActiveX comportment installed. The clients are set up to resemble client systems but by default have auto-updates on O.S., browser and applications turned off: this is to increase performance as the client systems would revert to their original state and be required to re-perform updates at every malicious webpage analysis tests. All required updates are installed to reflect nature of current client systems once monthly. Another issue with not disabling auto-update on applications, browser and O.S. would be: Capture's log files might flag updates of software as malicious if not well defined in exclusion lists resulting in possible false negatives.

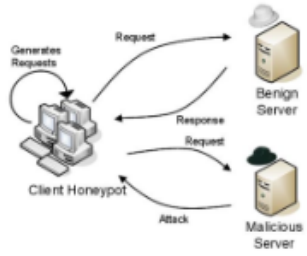


Figure 1: Client honeypot-Webserver interaction. Illustrated by the Honeynet project.

It is important to note that the version of Capture we are using is a modified version by HoneySpider Network called Capture-HPC_NG. There are a few differences to the original version of Capture most notably the ability for VirtualBox /KVM to be used as opposed to VMWare and changes to logging format [8].

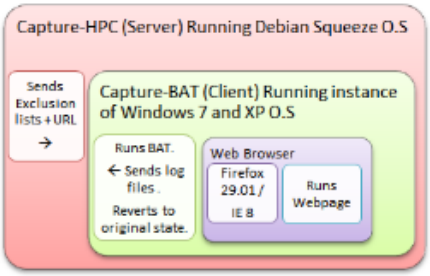


Figure 2: Our Capture-HPC/BAT interactions.

2.6 Exclusion lists

2.6.1 Overview

A key part of Capture-HPC is exclusion lists: these exclusion lists allow Capture to determine whether a particular behaviour is malicious or benign. Unexpected state changes are recorded in Capture log files [14]. In total Capture-HPC_NG provides 4 exclusion lists; ProcessMonitor, RegistryMonitor, FilesMonitor

and ConnectionMonitor [8]. As mentioned before we have excluded using ConnectionMonitor as we are currently unable to monitor network APIs.

Capture exclusion lists can be configured to either:

1. Exclude certain events in the log files: in which case the events will be benign and not flag the log file analysis as malicious. This is noted as a + in the exclusion list.

2. Include certain events in the log files which will be classified as malicious behaviour. This is noted as a - in the exclusion list.

There are 3 parts to each exclusion list: O.S., browser and applications behaviours. Each part of the exclusion list directs events which are either included or excluded. Exemplification is revealed In Table 1: the first event (which is part of the O.S. activities) allows wuauclt.exe (windows updater process) to write log files in C:\windows\WindowsUpdate.log. The second and third entries shown in the table is saying if any (:*) process creates any file name with the file extension .bat or .exe flag this behaviour as malicious.

Table 1. Example of FileMonitor exclusion list.

+/ -	File Access	Process Name	File Path
+	Write	C:\\WINDOWS\\system32\\wuauclt\.exe	C:\\WINDOWS\\WindowsUpdate\.log
-	Write	:*	.+\.bat
-	Write	:*	.+\.exe

2.6.2 *Exclusion list challenges*

Capture-HPC comes with default exclusions lists for windows XP. It is important to note that these exclusion lists were created at the time with the release of Capture-HPC, this means the default exclusion lists take into account the specific operating system, browser as well as applications installed on the system that was used in the creation of these exclusion lists. It is important to note that any change in version would likely require amendments to the exclusion list as there would be different application, O.S., browser and behaviours logged by Capture. Therefore using unaltered or default exclusion lists to tailor Capture client's O.S. version, browser version and application specific versions would mean in a number of false positives and negatives.

In order to develop accurate exclusion lists, a fairly high number of 'benign' tests should be run on Capture-HPC to identify benign behaviour. It is important to remember that significant tests should be carried out on the chosen benign websites to certify the non-malicious nature requirement as false positives here would mean allowing malicious behaviour to be classified as benign when Capture-HPC attempts to analyse malicious webpages. We feel that it is important to keep Capture-HPC clients as up-to-date as possible to reflect good detectability of current malicious activities. This consequently requires testing and updating Capture-HPC exclusion lists post updating or the installation of a new O.S., browser and applications running on a Capture-HPC client.

2.6.3 *Issues with exclusion lists*

The creation of exclusion lists is done by a security researcher and can sometimes be quite subjective as a lot of exclusions are still unknown. These unknown exclusions refer to behavioural changes that occur in a system and limited knowledge is available as to what the effects of such changes are. An example of this would be internet explorer adding registry entries but all that is given by a log file is a series of strings and numbers. Deciphering how a system was modified as a result of this change can often result in inconclusive results. This is often the case if the occurred change is not clearly displaying malicious behaviour and any potential benign behaviour is not documented by the Browser/application or OS. Example log file entries are provided in Figure 4. Consequently the decision of the researcher can also mean that exclusion lists can fail to detect certain exploits. There is therefore a requirement for a method that offers some good practices and guidance to a malware researcher when developing exclusion lists.

2.6.4 *Windows 7 application profiling*

The Capture-HPC_NG we obtained from HoneySpider Network comes with blank exclusion lists: We have created exclusion lists from scratch for Windows 7 home edition, for both IE 8.0 and Firefox 29.01 on a clean OS running the following applications: Flash Player 12, Shockwave player 12, Java 7. Figure 3 shows our decision making process to determine benign behaviour. Benign behaviour here refers to the system events that occur to Browser, O.S and applications installed as a result of running a tested benign webpage.

Figure 3: Picking benign websites to identify benign behaviour.

2.6.5 *Recommendations for developing exclusion lists*

1. Run latest version of O.S., browser and applications in Capture-HPC client. Once updated disable all auto-update functions and set hard drive to immutable mode to prevent changes.
2. Determine O.S., browser, and application activities by running Capture-BAT on start-up and learning regular system changes. Unknown changes should be investigated thoroughly before being included in the exclusion list.
3. Follow the steps for choosing benign webpages described in Figure 3. We recommend using at least 1 low interaction client honeypot in addition to a URL scanning service such as VirusTotal.com or wepawet.iseclab.org. The reasoning behind this is purely to have high certainty that the benign webpages are actually non-malicious or don't have links to malicious webpages in their advertisement streams.

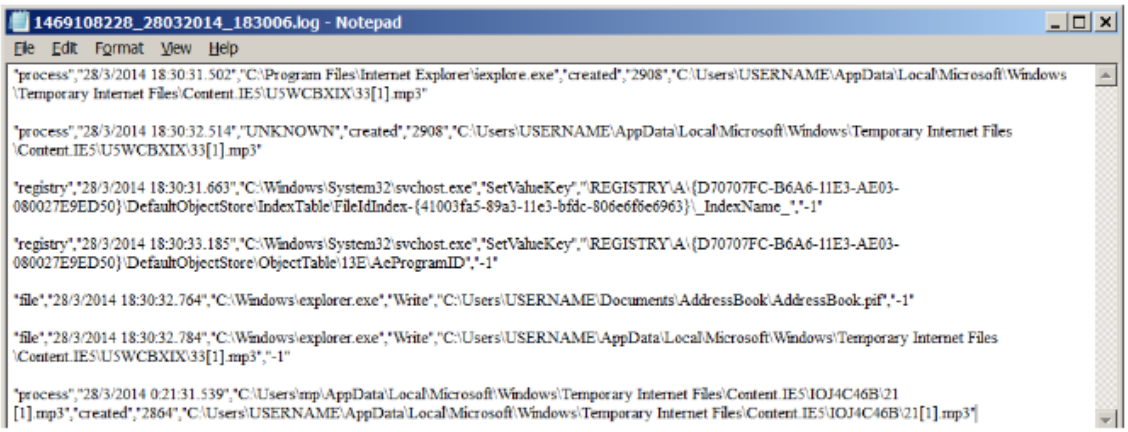


Figure 4: Sample malware behaviour from visiting malicious website

4. Run a high number of tests on benign webpages and develop exclusion lists based on constant and regular O.S., browser, application behaviour.

3. DISCUSSION

3.1 Malicious URLs analysed

Table 2 shows some initial results from our drive-by-download analysis. Tests are run in batches over the course of a few days. It is important to note that these tests are being carried out in a university network and these attacks are still getting through the security mechanisms in place. These security mechanisms included a Palo Alto Network firewall and MacAfee enterprise anti-virus software. For the purposes of the experiment anti-virus components (such as installed anti-viruses on capture clients) were disabled both Capture-HPC clients and server. With regards to the experiment, we believe more malicious behaviours would have been observed if there were no security mechanisms in place but this was not possible because the University network would be exposed to attacks.

Table 2. Data gathered in 04/2014 using Capture-HPC_NG and our exclusion lists.

Test batch	No. of URLs analysed	Malicious websites detected
1	523	59
2	470	62
3	325	43
4	690	73
5	632	78
6	571	78
Total	**3211**	**393**

3.2 Common attacks observed

In our malicious logs we've identified a number of common drive-by-download attacks. The most common type of attack involved the automatic download of an executable (exe). In most cases the executable was downloaded without any prompts or interaction with the malicious webpage other than visiting the website. This attack was found in a fairly high number of malicious webpages with the executable named in a manner which would try and confuse a user. Examples include; VideoRecord[1].exe, mp3.exe, exe.weed.mp3. Other attacks included registry changes to the Flash player and Macromedia player registry entries. In some cases a downloaded executable will automatically run and create malicious processes in the background without any interaction. These processes would then proceed to infect other parts of the machine by creating copies of the malicious executables in user folders and drives. Section 3.4 will examine a Capture-HPC log file and try to decipher some more malicious drive-by-download attacks our client machines faced. It is important to note that our clients are set to run for a specific amount of time which was set for around 5 minutes per webpage. This included the Capture-client running windows 7 boot up time.

3.3 Difficulties in accessing accuracy of Capture-HPC

As part of testing, we simultaneously 30 URLS using Capture-HPC and virus total at the same time due to the nature of short-lived malicious webpages. Capture-HPC's output is limited to either malicious or non-malicious. VirusTotal however has over 40 anti-virus vendors which can then find a website malicious or non-malicious. In the 30 URLS we tested, Capture-HPC only found 2 malicious. The majority of VirusTotal result was a number below 8/40 anti-virus vendors finding the webpage malicious. Due to the output being in very different format it's difficult to get an accurate overview of Capture-HPC's accuracy. A better approach could be to perhaps try other high-interaction client honeypots and compare the results with the output of Capture-HPC.

3.4 Sample malicious log files explained

Figure 4 shows sample of behavioural entries from a Capture-HPC log file: please note that the full log file would usually be several pages long and this one has been simplified for the purposes of demonstration. As soon as the client visits this malicious website, a drive-by-download executable is downloaded in the Temporary Internet Files folder. This was a drive-by-download with no prompt to download or run the malicious executable. After a number of malicious file are created, the executable is automatically run (again without any user intervention). A huge number of registry entries are then created for different registry keys (one example shown in Figure 4). We can also observe that an unknown process was created which ran an executable and proceeded to create a suspicious .mp3 file.

4. Conclusions

In this paper we have explored the issues within using Capture-HPC for research. Exclusion lists are a major component of Capture-HPC but are overlooked and very little research has been carried out on optimisation and efficiency. The creation of exclusion lists is challenging as every type of malicious and benign system behaviour is not documented. Unfortunately this leads to a degree of subjectivity in the creation of exclusion lists which may affect the rate of false positives and false negatives in capturing malicious behaviour. A major issue identified with current research is the lack of knowledge available in papers regarding their exclusion lists. If exclusion lists are not published with experiments using Capture-HPC, it would be difficult to reproduce results as different Capture-HPC instances with different exclusion lists would produce varying results. We have discussed the issues of creating and maintaining exclusion lists and provided a set of recommendations to creating and maintaining exclusion lists. Some common attacks and more advanced attacks are discussed. We have captured hundreds of malicious samples using our exclusion lists which will be used to form the dataset for our future research.

5. REFERENCES

[1] Akamai. 2013. *State of the Internet Report.* Q4 2013 6, 4. DOI=http://www.akamai.com/html/about/press/releases/2011/press_042611.html.

[2] Akiyama, M. Kawakoya, Y. Hariu, T. 2012. *Scalable and Performance-Efficient Client Honeypot on High Interaction System,* Applications and the Internet (SAINT), 2012 IEEE/IPSJ 12th International Symposium on , vol., no., pp.40,50, 16-20 (July 2012). DOI= 10.1109/SAINT.2012.15

[3] Amorim, R.C. and Komisarczuk P. 2012. *On Partitional Clustering of Malware.* CyberPatterns, Oxford Brookes, (Oxford, 9-10 July) 2012.

[4] Crist, J. 2007. *Web Based Attacks.* SANS. DOI=http://www.sans.org/reading-room/whitepapers/application/web-based-attacks-2053

[5] Dell'Aera, A and Seifert, C. 2012. *An introduction to honeyclient technologies.* Accessed 19/01/13 DOI=www.honeynet.org/files/HPAW2012-Thug.pdf.

[6] Egele, M, Wurzinger, P, Kruegel, C, Kirda, E. 2009. *Defending Browsers against Drive-by Downloads: Mitigating Heap-spraying Code Injection Attacks,* Secure Systems Lab, (2009) DOI=http://www.iseclab.org/papers/driveby.pdf

[7] Hong-Geun Kim, Dong-Jin Kim, Seong-Je Cho, Moonju Park, and Minkyu Park. 2011. *An efficient visitation algorithm to improve the detection speed of high-interaction client honeypots.* In Proceedings of the 2011 ACM Symposium on Research in Applied Computation (RACS '11). ACM, New York, NY, USA, 266-271. DOI=http://doi.acm.org/10.1145/2103380.2103435

[8] Honeynet Project Polish Chapter. 2012. *HoneySpider Network Capture-HPC NG.* DOI=http://pl.honeynet.org/HoneySpiderNetworkCapture

[9] Khattab, S.M.; Sangpachatanaruk, C.; Mosse, D.; Melhem, R.; Znati, T. 2004. *Roaming honeypots for mitigating service-level denial-of-service attacks,"* Distributed Computing Systems, 2004. Proceedings. 24th International Conference on Distributed Computing Systems (Japan, 2004)pp.328,337, 2004 DOI=10.1109/ICDCS.2004.1281598

[10] Koo, T.-M., Chang, H.-C., Hsu, Y.-T., Lin, H.-Y., 2013. *Malicious Website Detection Based on Honeypot Systems,* in: 2nd International Conference on Advances in Computer Science and Engineering (CSE 2013). Atlantis Press.

[11] Lau, B. and Svajcer, V. 2008. *Measuring virtual machine detection in malware using DSD tracer.* J. Comput. Virology.

[12] Deasmond, P, 2004. *All-out blitz against Web app attacks.* (Networking world, 2004) DOI=http://www.networkworld.com/article/2333088/lan-wan/all-out-blitz-against-web-app-attacks.html

[13] Puttaroo, M., Komisarczuk, P., Amorim, R.C., *On Drive-by-Download Attacks and Malware Classification* in Fifth International Conference on Internet Technologies & Applications (ITA), Wrexham, Wales, 10 to 13 September 2013

[14] Qassrawi, M. T. Zhang, H. 2010. *Client honeypots: Approaches and challenges* In New Trends in Information Science and Service Science (NISS), 2010 4th International Conference on New Trends in Information Science and Service Science (South Korea, 2010), pp. 19–25.

[15] Rivera, J. Meulen, R.V. 2013. *Gartner Says 25 Percent of Distributed Denial of Services Attacks in 2013 Will Be Application-Based.* DOI= http://www.gartner.com/newsroom/id/2344217

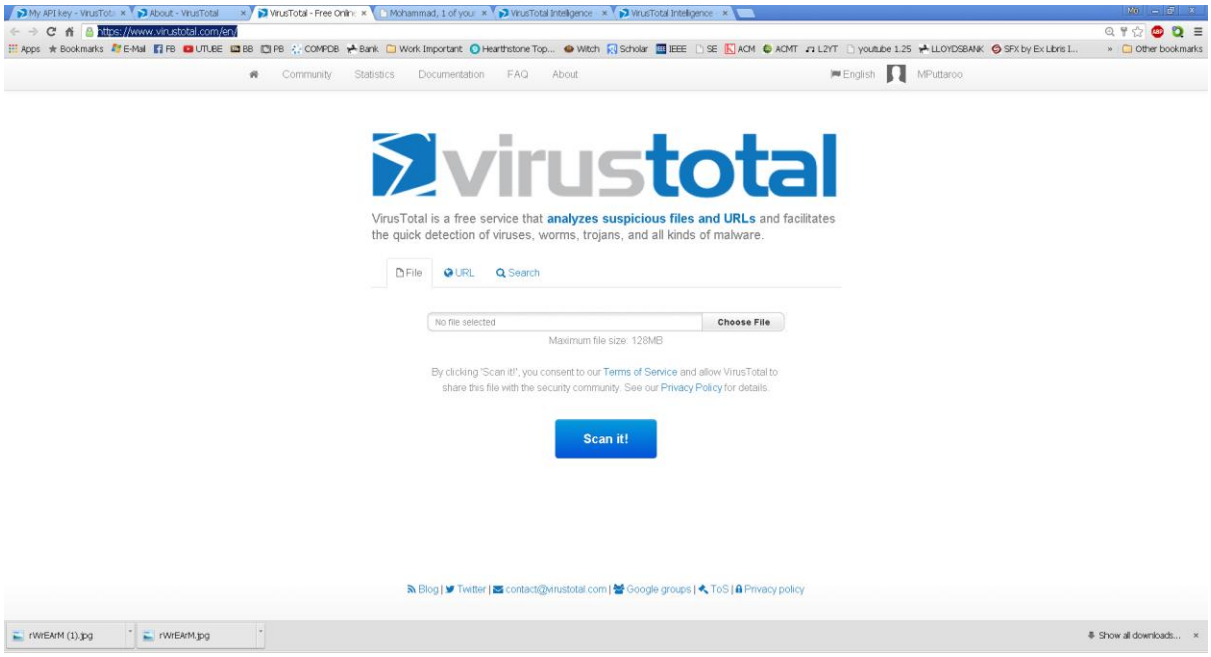
[16] Seifert, C. 2010. *Cost-effective detection of drive-by-download attacks with hybrid client honeypots.* Doctoral Thesis. Computer Science Department, Victoria University of Wellington.

[17] Shadowserver Foundation. 2014. *Information - Honeypots'.* Accessed June 9. DOI=http://www.shadowserver.org/wiki/pmwiki.php/Information/Honeypots

[18] The Honeynet Project. 2014. *Projects Overview* June 9 2014. DOI=http://www.honeynet.org/project

[19] WC3. 2014. *OS Statistics.* Accessed June 9. DOI= http://www.w3schools.com/browsers/browsers_os.asp

Figure 2C: Screenshot of Virus Total.



Virus Total features include:

1. The web interface can be used to upload a file/ submit a URL or search for a known hash / URL / malware
2. Email can be used to submit files and receive results via inbox
3. Virus Total uploader windows application can be used to send
4. Public/Private APIs can be used to submit malware samples, request results.

Figure 2D: Potential data obtainable from Virus Total part 1.

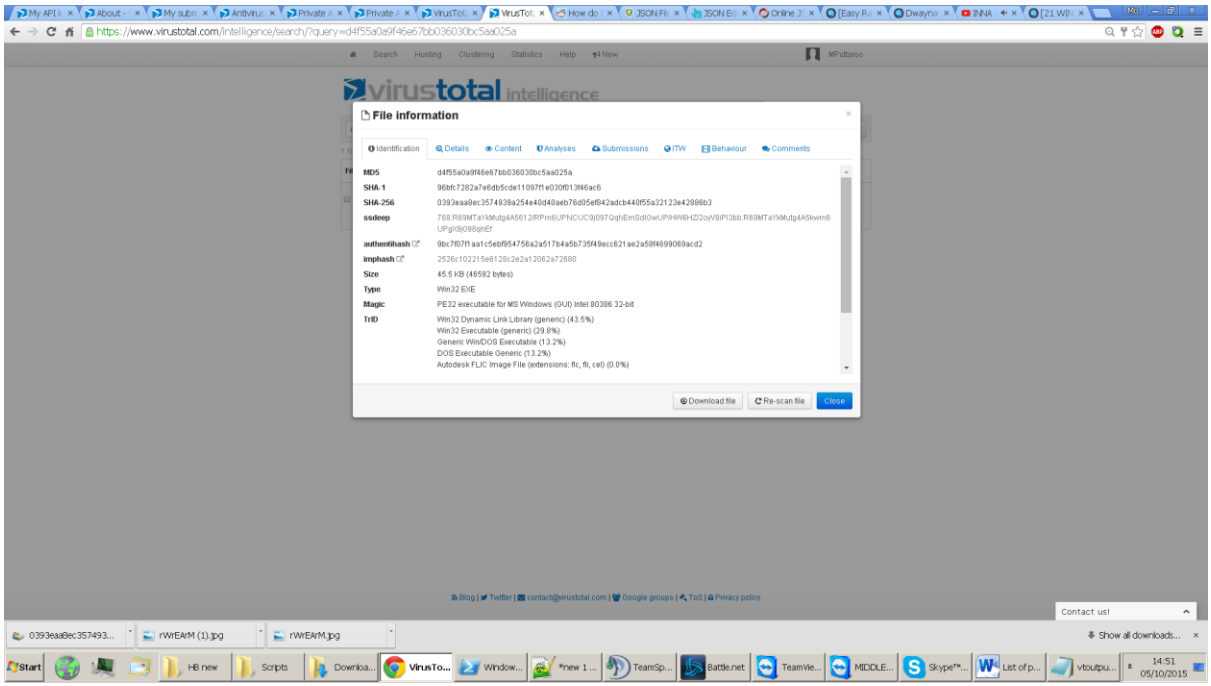
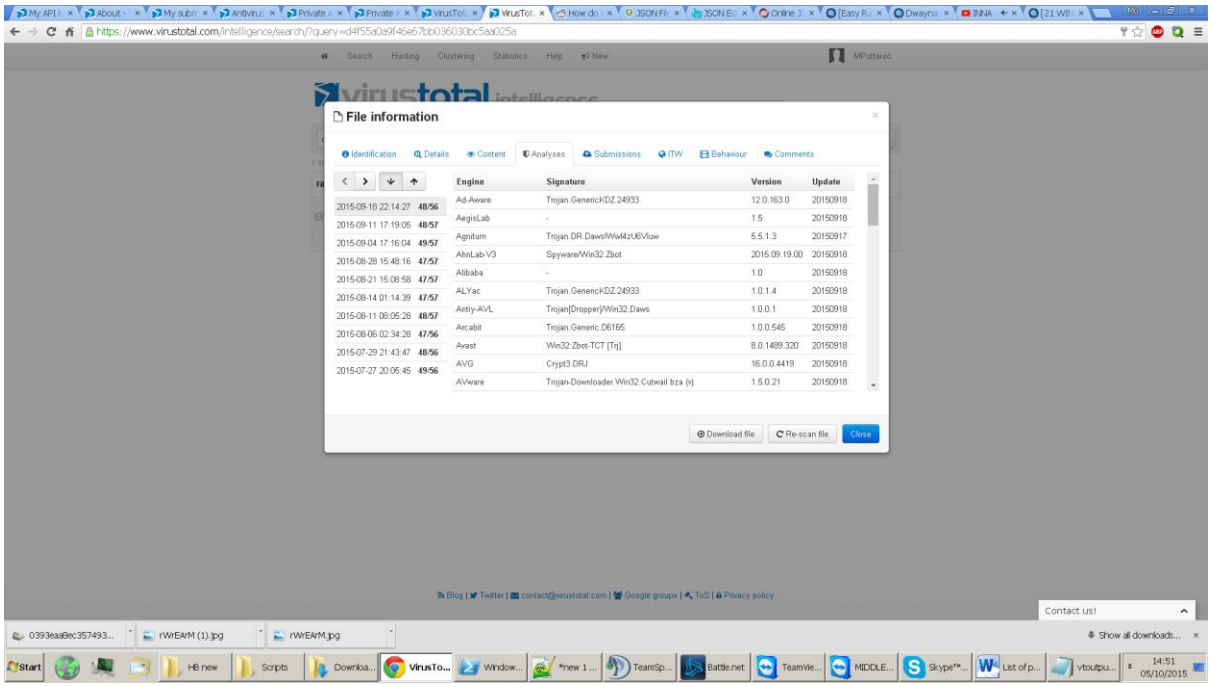


Figure 2D: Potential data obtainable from Virus Total part 2.



Appendix B

3.4 Testing the validity of the experimental setup

This experiment is a validity test on the experimental setup to ensure malware is able to attack the honeypot. The purpose of this experiment is to access whether the behavioural interactions saved on the Capture-HPC server post behavioural analysis by the Capture-BAT clients were correctly experiencing malicious behaviour attacks despite being placed behind a firewall. The majority of the URLs analysed were sourced from malicious domain lists.

It was observed that only a small percentage (around 20% per batch) were actively displaying malicious or unknown behaviours when analysed within the Windows 7 behaviour environment in a pilot study. There are likely a number of reasons behind this low percentage, particularly that a malicious web server can be inactive at times. This could conflict with the time that Capture-BAT is analysing a malicious web server as the analysis could take place during the inactive period.

There are also possibilities that a number of these malicious websites are tailored to attack a specific client which runs specific to a specific operating system or application version. Therefore, a large volume of attacks which target alternative client systems would not attack the honeypots used in this study. It is likely that some malicious web servers require user interaction. These signs of interaction are used to filter typical analysis environments as one very important goal of malware is to be undetected. Lack of user interaction in Capture-BAT would mean the criteria of triggering the web-exploit would not be satisfied meaning attacks would not take place. Furthermore, it is a possibility that the virtualised behaviour analysis environment was detected, thus preventing a number of malware from exploiting to

avoid detection. This small percentage of detected active malware per analysed batch raised the concern that perhaps the firewall was blocking too much malware and therefore contributed to the need of testing the experimental set-up.

The current experimental setup at the University of West London is mirrored with a cloned setup situated at a large organisation in Brentford (UK) protected by a non-disclosure agreement. The difference between the setup is that the current university setup is placed on a virtual network behind a vendor supplied web filter firewall whereas the setup at the large organisation is placed in their network's De-Militarized Zone (DMZ) without any protection. The reasoning behind this test is to identify if there is any variance between the behaviour analyses taken place and effectively evaluate if there are significant malware being blocked by the university's network defence mechanisms.

3.4.1 Hypothesis

It is expected that the university setup with the web filtering firewall is likely to experience smaller quantity of attacks as known drive-by-download exploits would be blocked.

3.4.2 Test control

In order to keep this experiment fair and valid, honey client systems running had to be exact cloned copies on both test sites. The dependant variables within this experiment consisted of identical systems. Both systems therefore needed the exact configuration and versions for the listed variables:

- Capture-HPC server version,
    - Capture-HPC_NG server.
- Capture-BAT client environment (O.S., browser, Applications),

    - Windows 7 ‘n edition’ clean SP1, with windows updates kb/3172605 and /kb/3020369 installed.
    - Internet explorer 8.
    - Flash player, Java, and Real Player
- Capture-HPC exclusion list,
    - Created Windows 7 exclusion lists: ProcessMonitor, FileMonitor and RegistryMonitor, available in the appendix B.
- Capture-HPC virtual network design and configuration,
    - Capture-HPC server bridged to both Capture-BAT clients
- Capture-HPC configuration,
    - The used ‘Config.xml’ configuration file for Capture-HPC is available in the Chapter 4, Figure 4.2b.
- Time: Malicious web servers can be offline or inactive at certain times during the day, therefore both sites would need to be analysing the same website at the same time in order for the experiment to remain a fair test within real-world behavioural analysis.

The Capture-HPC servers were cloned using VirtualBox’s clone feature (https://www.virtualbox.org/) which effectively allows easy copies of client environments as well as the ability to revert a given Virtual Machine (VM) to a clean state. These copies are completely mirrored meaning that system environment, applications and files are similar on both experimental setups (Reed, 2011).

The experiments were carried out concurrently and running each URL at the same time. This was achieved with the use of the remote-access and control software, TeamViewer. (https://www.teamviewer.com). The same batches of URL lists were inputted within both Capture-HPC servers and then the command to run was issued

at the same second. The lists used on both sites were cloned copies which meant that Capture would run the URLs on both site in the same order in addition to running the analysis simultaneously. As a result of the time-consuming nature of dynamic of drive-by-download analysis, this experiment was conducted over the period of three weeks with no changes such as software or O.S. updates to the four analysis (Capture-BAT) environments in order to keep the test valid and fair.

3.4.3 Results

An identical set of 2500 potentially active malicious URLs were simultaneously analysed starting at the exact same time in both locations. Out of both sets of 2500 URLs there was a 100% identical match in websites found malicious, thus disproving the hypothesis. In both cases 543 potentially malicious log files were created by Capture-BAT. This means that URLs which found to be malicious by the client honeypot system set up at University of West London were also identified as malicious at the large organisation in Brentford with no network and computer system defences in place to filter results.

Furthermore, both instances of Capture-HPC identified the same drive-by-download and the Capture-BAT clients was compromised by the same exploit in every case. The justification of this statement was concluded from the analysis of malware behaviours in both sets of data. This revealed that log files from both locations were identical thus Capture-BAT was detecting the same exploit. Worryingly, it would appear that the malicious behaviours that Capture-BAT was capturing in either cases were not hindered by the University's defence mechanisms which included an enabled firewall solution with web filtering. This could be due to the relatively newly setup malicious domains which were not identified by the defence mechanisms in place. The results of this experiment may not generalise over to another university

with different defence mechanisms in place as different security solutions may offer different levels of protection and restriction towards attack vectors that are known. Specific results from malicious URL behaviour analysis are discussed, analysed and these are presented in chapter 6.

It is important to note that the 543 log files were identical in both datasets: false positives identified by Capture-BAT were also shared between the datasets thus proving that the Capture-BAT and URLs analysed at the same time displayed similar manifestations.

3.4.4 Conclusion

There was no difference within the gathered data sets from both networks. It is therefore fair to conclude that the security systems in place at the University were unable to filter out any drive-by-download attacks that were targeting the Windows 7 client. This is a justified conclusion as every attack that took place within the unsecured and unfiltered honeypot deployment, in place were also detected at the university. Log files gathered for these attacks were checked and again the behavioural changes flagged by Capture as malicious were found in both instances of the test. As the results of the validation test were so promising, it is certain that the experimental setup was ideal for drive-by-download analysis and that there were no impeding factors based on the network environment that were preventing malware attacks while running the server on the university network.

The disproval of the hypothesis however does require critical thinking. While the attacks and behaviours remained identical in both cases, the low percentage of observed malicious behaviour may be due to unconsidered factors. It is likely that the firewall and web filtering option which is controlled by the university's network

team could have been offline at the time of the experiment. However, this is unlikely to have stayed offline for the course of 16 days which was the time required to analyse the 12556 URLs. Sample choice could have been unknowingly limited to a sample of newer and undetected web exploits. This could mean that the analysis environment would still face malware attacks on both sites and since the results on both sites were identical, it is likely that this could have been a viable reason why the security systems in place at the university were not stopping any of the drive-by-download attacks.

In terms of evaluating the choice of applications: arguably, it is good practice for honeypots to attempt to use older versions of software as on the world wide web, as malicious websites can often be seen active over 500 days after initial detection (Tanaka and Goto, 2016). The rationale is simply that older patched software often contains unpatched vulnerabilities that some of the remaining active malicious web servers will continue to target in order to infiltrate the client machine. It may be possible that not enough older applications were installed on the analysis environment or perhaps the versions that chosen and installed contained a large number of vulnerability patches which prevented malicious attacks seen in the wild, this would explain the relatively low (543/2500) levels of attacks discovered.

The experiment design was successful in achieving what it was designed for: the testing of validity of our main experimental setup. This is because a completely similar setup operating on a different network was analysing the same websites at the same time. This provides indication that observed behaviours should really be identical, which was also concluded from the log file analysis undertaken. With regards to the test itself, as all of the listed variables were kept under control and the only difference between the analysis environments being the geographical location

and defence structures in place, it proves to be a fair test of the validity of the Capture analysis environment. In conclusion, both the test being designed fairly and the results being promising meant that the Capture behaviour analysis environment was set up validly in line with the aforementioned prudent practices discussed in 3.1-3.1.5.

3.5 Evaluating twitter as a source of gathering malicious domains

The purpose of this experiment is to analyse URLs shared in “tweets” from the Twitter website (https://twitter.com/) and capture malicious behaviours using Capture-HPC. The point behind this is to access whether Twitter is a viable resource for the gathering process of malicious URLs within this thesis. Strickland & Chandler (2014) describe a tweet as “a message sent on Twitter”. The aim behind this experiment is to use a different data source that combined a list of both malicious and benign websites as opposed to purely malicious websites from malware domain and black listed website repositories.

3.5.1 Experimental purpose and background

This experiment allowed us to determine the viability of using social media to find compromised and malicious websites as opposed to relying on malware domain databases. Additionally, the knowledge gathered may lead to future experiments that analyse the malware propagated using social media in comparison to the captured malware originating from malware domains repositories. These may help determine whether these malwares samples inhibit similar malicious behaviours or whether the malware is from the same family or not.

The nature of the dynamic analysis of malware is time consuming and resource intensive on client systems. Each URL takes about 5 minutes to run and about 3-5

minutes to boot and reboot which is necessary after each malware test to revert our Capture-BAT to its original unmodified (by malicious behaviour) state. The value of 5 minutes per page is a shared variable with previous work by Burnap et al. (2015). The authors were able to observe malicious behaviour within their gathered malicious URLs within the first minute. There are around 500 million tweets everyday according to InternetLiveStats (2015), and is clearly it would be difficult to analyse that many tweets and gather URLS to be analysed by Capture-HPC. Consequently, an event namely the World Cup in 2014 was chosen and analysis of 1% of all the randomly selected daily tweets, which contained the hashtag #Worldcup. The first part of the experiment involves capturing the URLs from these specific tweets on that specific date. A script is used to gather these URLs and duplicates of the URL captured are excluded from our URL list.

This experiment allowed the determination of which data source to use when attempting to gather as many attacks as possible targeting client systems. The same test bed and experimental setup discussed in 5.1 is used and the Capture-HPC system was cloned and provided to the alternative setup that was used to run the other half of the dataset. This isn't a problem despite different geographical locations and hardware being utilised as results were 100% identical when verified using same the URLs simultaneously as concluded in experiment run in section 3.4 of the chapter.

3.5.2 Results

Table 3.4: Percentage URLs classified as malware originating from twitter when run in Capture-HPC.

Date of analysis	Benign URLs	Malicious URLs	Total	% Malicious

15-Jul-14	1276	119	1395	8.53%
16-Jul-14	1133	262	1395	18.78%
17-Jul-14	1124	271	1395	19.43%
18-Jul-14	1058	337	1395	24.16%
19-Jul-14	1239	156	1395	11.18%
25-Jul-14	1206	189	1395	13.55%
26-Jul-14	1045	351	1396	25.14%
30-Jul-14	1173	222	1395	15.91%
31-Jul-14	1241	154	1395	11.04%
Total	**10495**	**2061**	**12556**	**16.41%**

These results are purely from the behaviour analysis undertaken by Capture-BAT from the interaction between the client honeypot and the website link found on Twitter in tweets containing the hashtag, #worldcup and a URL. It is important to note that since 16% of all the analysed tweets were being posted by twitter users, it seemed unlikely that such a high percentage would contain links to a compromised malicious website. This therefore required fine-grained analysis for behavioural verification purposes and identification of possible false positives and false negatives.

In order to perform fine-grained analysis of Capture log files, one approach was to identify malicious behaviours contained in the log files created by Capture-BAT. These would provide behavioural information on the actual interactions that took place during website analysis. Fine grain analysis of log files within Capture-BAT is an exhaustive process as often Capture-BAT log files contain hundreds of behavioural interactions as can be seen by examples in Appendix B. However, the log files created during the Twitter evaluation as a source for malware gathering

were filtered using exclusion lists at the data gathering process. Highly detailed information about these filters' creation, configurations and usage are provided in Chapter 4. The filtered log files were then combined and behaviour lists which included unique behaviours were created. These lists included the behavioural frequency observed. Performing these data transformation operations reduced the raw dataset of 73769 behaviours down to 4568 unique behaviours, netting a 93% reduction. This step is important as the maximum amount of reductions prior to the fine-grained analysis results in the least data redundancy when a malware analyst is performing manual investigation of behaviours.

Initially, for the fine-grained analysis captured behaviours are verified and compared with expected behaviours for each known process (which can be seen in the next chapter's behaviour tables). Secondly, unknown behaviours are thoroughly investigated and compared with a database of known benign behaviours. This provides further indication of maliciousness. Performing these two tasks yields a conclusion on whether a behavioural interaction is malicious, benign or simply unknown. In terms of the findings of the fine grain analysis, it was observed that the vast majority (2059/2061) of log files that Capture flagged as malicious were false positives: the behaviours contained within log files did not show any signs maliciousness despite being classified as such. The misclassification by Capture-BAT is caused by utilised exclusion lists (discussed in Chapter 4) which by default mark any new behaviour as malicious. Since the drifted behaviours that Capture-BAT clients identified in the 2059 instances were not present in the exclusion list, Capture-BAT would mark these as malicious, which then proved to be false positive.

Out of the log files that Capture-BAT found malicious, it was concluded that there were only two traces of malicious behaviour where Internet explorer was hijacked

and wrote malicious .tmp files which were executed. Possible reasons for these findings on analysed Twitter URLs are explored below.

3.5.3 Conclusion

Out of the combined 12,556 URLS analysed for this experiment, the vast majority were benign and did not attack the Windows 7 based client systems directly. However, a varying number of behaviours were flagged as malicious by Capture-BAT. In terms of gathering a large list of user shared URLs, comparing the amounts of unique flagged malicious URLs obtained from Twitter it would appear that Twitter has some potential for providing researchers with large volumes of URLs. This source for malware propagation and certainly can be very viable for client honeypot research, specifically if the client honeypot solutions are able to handle extreme amounts of URL analysis. It is agreed that dynamic analysis by Capture-BAT is however not quite suitable to perform analysis on large amounts of URLs with very low potential of malware attacks. Dynamic analysis is resource intensive and therefore has a very low percent of detection ratio from the URLs from Twitter. This yields a poor overall maliciousness capture rate. A better approach as often research in this field undertakes would combine low-interaction client honeypots or hybrid systems to perform emulated analysis and identify key URLs that are likely to be malicious prior to dynamic analysis.

However, this method would be unsuitable to gather a data corpus for this research as the frequency of attack was far too low compared to the percentage found using malicious URL repositories. Results are displayed in table 3.4 and discussed in section 3.5.2. Clearly, for us to gather more malicious behaviours within the available Capture-HPC server and Capture-BAT clients: it is much more effective to use a data source originating from a malicious domain repository. It should be noted that

we observed a number of the URLS shared on Twitter as potential phishing attacks but that is beyond the scope of this research as the focus is on client attacks with limited user interaction. Further research especially using static analysis to detect phishing attacks are discussed by Lee and Kim (2012).

More concerningly with the conclusion, from analysing the log file data from Twitter, it would appear that log files contained a large number of false positives which were caused from a concept that is introduced as 'environmental behaviour drift' and irregular system behaviours often observed. The concept of behavioural drift is explored further and discussed in chapters 4, and 6. Irregular system behaviour observed could be caused by communication between the operating system, Windows 7 and Microsoft. It is evident that from these results, we can conclude that the output of behaviour analysis labs can sometimes vary despite the same system being utilised for analysis. These slight variations in system calls creates some false positives which can be identified with fine-grain analysis of log files. In order to avoid this pitfall however, it is possible to refine filters utilised within dynamic analysis by developing and updating filters regularly. The creation, use and implementation of filters are discussed thoroughly in Chapter 4.

3.5.4 Evaluation

The evaluation of Twitter as a potential source for gathering malicious behaviour was undertaken by actively gathering a large number of user shared URLs in their tweets containing a very specific hashtag (#Worldcup), time frame (shared on the day of the Word Cup final in 2014) and contained a unique URL. This was necessary as the data potential available on Twitter is far too huge in its entirety to perform dynamic analysis on every shared URL. It was felt that this approach was able to access the main problem which were URLs on Twitter sufficiently malicious for client honeypot

research to perform dynamic analysis and collect malware samples. This was found to not be the case as from the small and limited set of data analysed the amount of malicious interaction was far too low (2/12556 URLS).

It is however important to not generalise within cyber security research as the dataset used in assessing the tweets from Twitter as a possible source for malicious URL gathering was limited in several ways:

- Only 1% of the total tweets which contained the #Worldcup phrase and a URL was analysed. 1% of the hash tag is a very small sample size even though dynamic drive-by-download analysis is being undertaken. Repeated URLs which would cause duplicates were removed from the selection of URL to avoid redundant data and processing. Twitter API however limits developers to 1% of tweets per day.
- It is probable that different hashtags, typically focused on illegal, porn, warez might typically be more suited and likely to containing links that are malicious. Assessing the hashtag and the categories are a research question in itself and out of scope of this research.
- It could be possible that the analysis environment was lacking operating system vulnerabilities, applications with specific versions containing vulnerabilities or was simply detected by malware. These factors may cause malware to not manifest and attempt compromising the client honeypot.
- The analysis of multiple of the dataset was conducted a few days (15th – 31st July 2014) after the World Cup final 2014 (13th of July 2014) due to the sheer time requirements and equipment limitation of dynamic analysis. During the time of analysis, it could be possible that potentially malicious websites had changed malicious behaviours or been inactive.

- Capture-BAT analysis environment despite running on a slightly higher level of privilege (within the Kernel mode) than malware (user mode), is susceptible to malware detection which upon detection may choose to not execute or infect a system. Despite running on a higher privilege level, the captureclient.exe component is visible as a process which could compromise behavioural analysis capture from a malicious attack.
- Similarly, despite having taken precautions from not installing Guest Additions on VirtualBox which facilitates the detection of the virtual environment by malware, newer malicious drive-by-downloads detect the presence of the virtual machine and cease any attempts to execute and be detected. It is possible that the virtual environment was still discovered and malware did not trigger to avoid detection.

Noticeably, as a result of the low level of experienced malicious attacks it was not possible to conclude malware diversity and potential similarities between malware families. Despite being a very limited dataset and having a number limitations, the bottom-line was that the ultra-low occurrence actual and confirmed maliciousness observed in the dataset displays Tweets as a poor source for dynamic drive-by-download analysis for this study. The ultra-low occurrence conclusion is sufficiently low enough to conclude that pursuing pure dynamic analysis on datasets which include a large number of benign websites is not worth the resource and time requirements associated with dynamic analysis. This therefore evaluates Twitter as a poor source to gather malicious URLs for this research. In terms of future prospects, it is recommended that some form of pre-filters be applied to the URL selection process prior to analysing shared URLs in tweets as this may result in better malicious URL yields, as the existing knowledge within hybrid systems prove (Seifert

et al. 2008; Le. et al. (2011). Consequently, a range of different malicious domain websites and the bank of potentially malicious websites provided by the large organisation were used to gather malicious behavioural interactions attacking our Windows 7 analysis environments.

Figure 4A: Example log file of figure 4.0a in full.

This is an unfiltered log file showing the large volumes of benign behaviours that can be present in behavioural log files from Capture-BAT.

```
"registry","18/8/2016 17:5:44.264","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:5:44.264","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:5:44.264","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:5:44.264","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:5:44.264","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:5:44.264","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:5:44.264","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:5:44.264","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:5:44.274","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:5:44.274","C:\Windows\System32\lsass.exe","SetValueKey","HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\B\52C64B7E\LanguageList","-1"
"registry","18/8/2016 17:6:20.422","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\22","-1"
"registry","18/8/2016 17:6:20.422","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\24\ffffffffffffffffffffffffffffff00","-1"
"registry","18/8/2016 17:6:20.422","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\24\ffffffffffffffffffffffffffffff01","-1"
"registry","18/8/2016 17:6:20.422","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\24\ffffffffffffffffffffffffffffff02","-1"
"registry","18/8/2016 17:6:20.422","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\Nsi\{eb004a03-9b1a-11d4-9123-0050047759bc}\24\ffffffffffffffffffffffffffffff03","-1"
"registry","18/8/2016 17:6:32.88","C:\Windows\System32\svchost.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\ConfigValueEssNeedsLoading","-1"
"registry","18/8/2016 17:6:32.88","C:\Windows\System32\svchost.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\List of event-active namespaces","-1"
"file","18/8/2016 17:6:35.623","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\1F39B5CFACECFDE48DB25BCA2231FAC6_135A427F1ED873A4BF5097F7A809FA2A","-1"
"file","18/8/2016 17:6:35.964","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\1F39B5CFACECFDE48DB25BCA2231FAC6_135A427F1ED873A4BF5097F7A809FA2A","-1"
```

"file","18/8/2016 17:6:36.445","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\398EE64D66758B5715368AA94044B13A","-1"

"file","18/8/2016 17:6:36.475","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\398EE64D66758B5715368AA94044B13A","-1"

"file","18/8/2016
17:6:36.545","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\6AA3321A15A787985201D7A6820782F0_0AB46376AFB6F40B0
426680E3025D384","-1"

"file","18/8/2016
17:6:36.885","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\6AA3321A15A787985201D7A6820782F0_0AB46376AFB6F40B
0426680E3025D384","-1"

"file","18/8/2016
17:6:38.197","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\6C05FF55E66434DC351985A3C60541B2_305471F92FEBDAC5
5C5F5411833A3468","-1"

"file","18/8/2016
17:6:38.227","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\6C05FF55E66434DC351985A3C60541B2_305471F92FEBDAC
55C5F5411833A3468","-1"

"file","18/8/2016 17:6:38.728","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\74BFD122C0875EC75DBE5C6DB4C59019","-1"

"file","18/8/2016 17:6:38.988","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\74BFD122C0875EC75DBE5C6DB4C59019","-
1"

"file","18/8/2016 17:6:39.439","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\7D1F03728133589A90656A87E482B21F","-1"

"file","18/8/2016 17:6:39.469","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\7D1F03728133589A90656A87E482B21F","-1"

"file","18/8/2016
17:6:39.759","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_298C7C05A76CF4F8
7B7E48888C7B12A9","-1"

"file","18/8/2016
17:6:40.250","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_298C7C05A76CF4
F87B7E48888C7B12A9","-1"

"file","18/8/2016
17:6:40.540","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_5C3A32DA063DE665
E1701639A5C722A9","-1"

"file","18/8/2016
17:6:40.561","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_5C3A32DA063DE6
65E1701639A5C722A9","-1"

"file","18/8/2016
17:6:40.751","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_6B70447A236C02D3
01D26CFC6ACFF105","-1"

"file","18/8/2016
17:6:40.881","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_6B70447A236C02D
301D26CFC6ACFF105","-1"

"file","18/8/2016
17:6:41.51","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_6CCEF6187B3AC1DA
B87811FC56F159B7","-1"

"file","18/8/2016
17:6:41.221","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_6CCEF6187B3AC1
DAB87811FC56F159B7","-1"

"file","18/8/2016
17:6:41.372","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_73CCC4789FA2D9D
A18374A446E6CB52F","-1"

"file","18/8/2016
17:6:41.372","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_73CCC4789FA2D9
DA18374A446E6CB52F","-1"

"file","18/8/2016
17:6:41.502","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_828F413031E67E157
14D03EF0798B9D3","-1"

"file","18/8/2016
17:6:41.502","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_828F413031E67E1
5714D03EF0798B9D3","-1"

"file","18/8/2016
17:6:41.642","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_875DC57161BFC352
C6F0FB0FDD89B1C7","-1"

"file","18/8/2016
17:6:41.652","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_875DC57161BFC3
52C6F0FB0FDD89B1C7","-1"

"file","18/8/2016
17:6:41.993","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_8C9D805FB3580001
D2C6564FF820AC98","-1"

"file","18/8/2016
17:6:42.113","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_8C9D805FB358000
1D2C6564FF820AC98","-1"

"file","18/8/2016
17:6:42.223","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_9349CA9C5EFC1ED
F8DF04C537401734A","-1"

"file","18/8/2016 17:6:42.233","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_9349CA9C5EFC1EDF8DF04C537401734A","-1"

"file","18/8/2016 17:6:42.433","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_A8E52BA703C1983F470C3168CE5EC373","-1"

"file","18/8/2016 17:6:42.443","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_A8E52BA703C1983F470C3168CE5EC373","-1"

"file","18/8/2016 17:6:42.654","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_B853D891ADBA9AECA542B0768864B67A","-1"

"file","18/8/2016 17:6:42.684","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_B853D891ADBA9AECA542B0768864B67A","-1"

"file","18/8/2016 17:6:42.774","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_C416C4E491DEACA1E465A326CE8F4165","-1"

"file","18/8/2016 17:6:42.804","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_C416C4E491DEACA1E465A326CE8F4165","-1"

"file","18/8/2016 17:6:42.894","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_DA53CA31FF60124D3FEECC2CD4333E28","-1"

"file","18/8/2016 17:6:42.904","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_DA53CA31FF60124D3FEECC2CD4333E28","-1"

"file","18/8/2016 17:6:43.294","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\8059E9A0D314877E40FE93D8CCFB3C69_DDF26EA9BA25D62EDB0E67D547244630","-1"

"file","18/8/2016 17:6:43.304","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\8059E9A0D314877E40FE93D8CCFB3C69_DDF26EA9BA25D62EDB0E67D547244630","-1"

"file","18/8/2016 17:6:43.915","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\96D7A99548C36B10D2E8035A3E0DCA1A","-1"

"file","18/8/2016 17:6:44.96","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\96D7A99548C36B10D2E8035A3E0DCA1A","-1"

"file","18/8/2016 17:6:44.316","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\BD8A14C7C024625432CC03FE72E47EF0_6FD1BEFD298F4FD3EE4B4EE2E6631CC7","-1"

"file","18/8/2016 17:6:44.316","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\BD8A14C7C024625432CC03FE72E47EF0_6FD1BEFD298F4FD3EE4B4EE2E6631CC7","-1"

"file","18/8/2016 17:6:44.336","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\CA7B2D59B4E9BC2D316D1AECDFC12F63_5E2C5C05624F63862BE7A871C6A8C546","-1"

"file","18/8/2016 17:6:44.436","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\CA7B2D59B4E9BC2D316D1AECDFC12F63_5E2C5C05624F63862BE7A871C6A8C546","-1"

"file","18/8/2016 17:6:44.616","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content\CA7B2D59B4E9BC2D316D1AECDFC12F63_C1E12B24931DF30EF8125657DA7A408C","-1"

"file","18/8/2016 17:6:44.626","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\CA7B2D59B4E9BC2D316D1AECDFC12F63_C1E12B24931DF30EF8125657DA7A408C","-1"

"registry","18/8/2016 17:6:47.821","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\WMI\Autologger\Circular Kernel Context Logger\Status","-1"

"registry","18/8/2016 17:6:48.81","C:\Program Files\Internet Explorer\iexplore.exe","SetValueKey","HKCU\Software\Microsoft\Internet Explorer\LowRegistry\ErrorReporting\LastShipAssertTime","-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer","-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer","-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer","-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer","-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer","-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet
Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer",
"-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.832","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"registry","18/8/2016 17:6:54.200","C:\Windows\System32\svchost.exe","SetValueKey","HKCU\Software\Microsoft\Internet Explorer\LowRegistry\Audio\PolicyConfig\PropertyStore\c68486af_0\{219ED5A0-9CBF-4F3A-B927-37C9E5C5F14F}\5","-1"

"registry","18/8/2016 17:6:54.200","C:\Windows\System32\svchost.exe","SetValueKey","HKCU\Software\Microsoft\Internet Explorer\LowRegistry\Audio\PolicyConfig\PropertyStore\c68486af_0\{219ED5A0-9CBF-4F3A-B927-37C9E5C5F14F}\4","-1"

"registry","18/8/2016 17:6:54.210","C:\Windows\System32\svchost.exe","SetValueKey","HKCU\Software\Microsoft\Internet Explorer\LowRegistry\Audio\PolicyConfig\PropertyStore\c68486af_0\{219ED5A0-9CBF-4F3A-B927-37C9E5C5F14F}\3","-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:48.842","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\WER\ReportArchive\NonCritical_iexplore.exe_12df8271b62395a348f102b12959f9768e2baf9_0a72158a\Report.wer", "-1"

"file","18/8/2016 17:6:49.353","C:\Windows\System32\svchost.exe","Write","C:\Windows\System32\wdi\{86432a0b-3c7d-4ddf-a89c-172faa90485d}\{bfb496e0-c065-4bc4-b775-dc7163f8c87f}\snapshot.etl","-1"

"file","18/8/2016 17:6:49.353","C:\Windows\System32\svchost.exe","Write","C:\Windows\System32\wdi\{86432a0b-3c7d-4ddf-a89c-172faa90485d}\{bfb496e0-c065-4bc4-b775-dc7163f8c87f}\snapshot.etl","-1"

"file","18/8/2016 17:6:49.353","C:\Windows\System32\svchost.exe","Write","C:\Windows\System32\wdi\{86432a0b-3c7d-4ddf-a89c-172faa90485d}\{bfb496e0-c065-4bc4-b775-dc7163f8c87f}\snapshot.etl","-1"

"registry","18/8/2016 17:6:56.353","C:\Windows\System32\svchost.exe","SetValueKey","HKCU\Software\Microsoft\Internet Explorer\LowRegistry\Audio\PolicyConfig\PropertyStore\c68486af_0\{219ED5A0-9CBF-4F3A-B927-37C9E5C5F14F}\5","-1"

"registry","18/8/2016 17:6:56.353","C:\Windows\System32\svchost.exe","SetValueKey","HKCU\Software\Microsoft\Internet Explorer\LowRegistry\Audio\PolicyConfig\PropertyStore\c68486af_0\{219ED5A0-9CBF-4F3A-B927-37C9E5C5F14F}\4","-1"

"registry","18/8/2016 17:6:56.353","C:\Windows\System32\svchost.exe","SetValueKey","HKCU\Software\Microsoft\Internet Explorer\LowRegistry\Audio\PolicyConfig\PropertyStore\c68486af_0\{219ED5A0-9CBF-4F3A-B927-37C9E5C5F14F}\3","-1"

"registry","18/8/2016 17:6:56.353","C:\Windows\System32\svchost.exe","SetValueKey","HKCU\Software\Microsoft\Internet Explorer\LowRegistry\Audio\PolicyConfig\PropertyStore\c68486af_0\{219ED5A0-9CBF-4F3A-B927-37C9E5C5F14F}\5","-1"

"registry","18/8/2016 17:6:56.353","C:\Windows\System32\svchost.exe","SetValueKey","HKCU\Software\Microsoft\Internet Explorer\LowRegistry\Audio\PolicyConfig\PropertyStore\c68486af_0\{219ED5A0-9CBF-4F3A-B927-37C9E5C5F14F}\4","-1"

"registry","18/8/2016 17:6:56.363","C:\Windows\System32\svchost.exe","SetValueKey","HKCU\Software\Microsoft\Internet Explorer\LowRegistry\Audio\PolicyConfig\PropertyStore\c68486af_0\{219ED5A0-9CBF-4F3A-B927-37C9E5C5F14F}\3","-1"

"file","18/8/2016 17:7:5.45","C:\Windows\System32\svchost.exe","Write","C:\Windows\System32\wdi\ShutdownPerformanceDiagnostics_SystemData.bin","-1"

"registry","18/8/2016 17:7:5.926","C:\Windows\explorer.exe","SetValueKey","HKCU\Software\Microsoft\Windows\CurrentVersion\Action Center\Checks\{852FB1F8-5CC6-4567-9C0E-7C330F8807C2}.check.100\CheckSetting","-1"

"registry","18/8/2016 17:7:5.926","C:\Windows\explorer.exe","SetValueKey","HKCU\Software\Microsoft\Windows\CurrentVersion\Action Center\Checks\{852FB1F8-5CC6-4567-9C0E-7C330F8807C2}.check.101\CheckSetting","-1"

"registry","18/8/2016 17:7:6.277","C:\Windows\explorer.exe","SetValueKey","HKCU\Software\Microsoft\Windows\CurrentVersion\Action Center\Checks\{C8E6F269-B90A-4053-A3BE-499AFCEC98C4}.check.0\CheckSetting","-1"

"registry","18/8/2016 17:7:6.898","C:\Windows\explorer.exe","SetValueKey","HKCU\Software\Microsoft\Windows\CurrentVersion\Action Center\Checks\{01979c6a-42fa-414c-b8aa-eee2c8202018}.check.100\CheckSetting","-1"

"file","18/8/2016 17:7:8.9","C:\Windows\System32\svchost.exe","Write","C:\Windows\System32\wdi\BootPerformanceDiagnostics_SystemData.bin","-1"

"file","18/8/2016 17:7:8.89","C:\Windows\System32\svchost.exe","Write","C:\Windows\System32\wdi\{86432a0b-3c7d-4ddf-a89c-172faa90485d}\S-1-5-21-78034117-1329648361-637219273-1001_UserData.bin","-1"

"registry","18/8/2016 17:7:8.450","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\Capabilities","-1"

"registry","18/8/2016 17:7:8.450","System","DeleteValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\UINumber","-1"

"registry","18/8/2016 17:7:8.450","System","DeleteValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\LogConf\BasicConfigVector","-1"

"registry","18/8/2016 17:7:8.450","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\HardwareID","-1"

"registry","18/8/2016 17:7:8.450","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\CompatibleIDs","-1"

"registry","18/8/2016 17:7:8.450","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\ContainerID","-1"

"registry","18/8/2016 17:7:8.450","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\Device Parameters\NodeID","-1"

"registry","18/8/2016 17:7:8.450","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\Device Parameters\Identity","-1"

"registry","18/8/2016 17:7:8.450","System","DeleteValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\LogConf\BootConfig","-1"

"registry","18/8/2016 17:7:8.450","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\Capabilities","-1"

"registry","18/8/2016 17:7:8.450","System","DeleteValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\UINumber","-1"

"registry","18/8/2016 17:7:8.450","System","SetValueKey","HKLM\SYSTEM\ControlSet001\services\umbus\Enum\1","-1"

"registry","18/8/2016 17:7:8.450","System","SetValueKey","HKLM\SYSTEM\ControlSet001\services\umbus\Enum\Count","-1"

"registry","18/8/2016 17:7:8.450","System","SetValueKey","HKLM\SYSTEM\ControlSet001\services\umbus\Enum\NextInstance","-1"

"registry","18/8/2016 17:7:8.450","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\DeviceClasses\{65a9a6cf-64cd-480b-843e-32c86e1ba19f}\##?#UMB#UMB#1&841921d&0&PrinterBusEnumerator#{65a9a6cf-64cd-480b-843e-32c86e1ba19f}\DeviceInstance","-1"

"registry","18/8/2016 17:7:8.450","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\DeviceClasses\{65a9a6cf-64cd-480b-843e-32c86e1ba19f}\##?#UMB#UMB#1&841921d&0&PrinterBusEnumerator#{65a9a6cf-64cd-480b-843e-32c86e1ba19f}\#\SymbolicLink","-1"

"registry","18/8/2016 17:7:8.460","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\DeviceClasses\{65a9a6cf-64cd-480b-843e-32c86e1ba19f}\##?#UMB#UMB#1&841921d&0&PrinterBusEnumerator#{65a9a6cf-64cd-480b-843e-32c86e1ba19f}\DeviceInstance","-1"

"registry","18/8/2016 17:7:8.460","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\DeviceClasses\{65a9a6cf-64cd-480b-843e-32c86e1ba19f}\##?#UMB#UMB#1&841921d&0&PrinterBusEnumerator#{65a9a6cf-64cd-480b-843e-32c86e1ba19f}\#\SymbolicLink","-1"

"registry","18/8/2016 17:7:8.460","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\DeviceClasses\{65a9a6cf-64cd-480b-843e-32c86e1ba19f}\##?#UMB#UMB#1&841921d&0&PrinterBusEnumerator#{65a9a6cf-64cd-480b-843e-32c86e1ba19f}\#\Control\Linked","-1"

"registry","18/8/2016 17:7:8.460","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\DeviceClasses\{65a9a6cf-64cd-480b-843e-32c86e1ba19f}\##?#UMB#UMB#1&841921d&0&PrinterBusEnumerator#{65a9a6cf-64cd-480b-843e-32c86e1ba19f}\Control\ReferenceCount","-1"

"registry","18/8/2016 17:7:8.460","System","SetValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\Capabilities","-1"

"registry","18/8/2016 17:7:8.460","System","DeleteValueKey","HKLM\SYSTEM\ControlSet001\Enum\UMB\UMB\1&841921d&0&PrinterBusEnumerator\UINumber","-1"

"registry","18/8/2016 17:7:9.151","C:\Windows\System32\spoolsv.exe","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\Print\Printers\SymbolicLinkValue","-1"

"registry","18/8/2016 17:7:10.303","C:\Windows\System32\spoolsv.exe","SetValueKey","HKLM\SYSTEM\ControlSet001\Control\Print\BeepEnabled","-1"

"registry","18/8/2016 17:7:11.484","C:\Windows\System32\spoolsv.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Print\Printers\DefaultSpoolDirectory","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Ports\Ne00:","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKU\S-1-5-19\Software\Microsoft\Windows NT\CurrentVersion\Devices\Microsoft XPS Document Writer","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKU\S-1-5-19\Software\Microsoft\Windows NT\CurrentVersion\PrinterPorts\Microsoft XPS Document Writer","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Ports\Ne01:","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKU\S-1-5-19\Software\Microsoft\Windows NT\CurrentVersion\Devices\Fax","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKU\S-1-5-19\Software\Microsoft\Windows NT\CurrentVersion\PrinterPorts\Fax","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Ports\Ne00:","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKU\S-1-5-20\Software\Microsoft\Windows NT\CurrentVersion\Devices\Microsoft XPS Document Writer","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKU\S-1-5-20\Software\Microsoft\Windows NT\CurrentVersion\PrinterPorts\Microsoft XPS Document Writer","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Ports\Ne01:","-1"

"file","18/8/2016 17:7:12.416","C:\Windows\System32\svchost.exe","Delete","C:\Users\mp\AppData\Local\Temp\BIT4B9.tmp","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKU\S-1-5-20\Software\Microsoft\Windows NT\CurrentVersion\Devices\Fax","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKU\S-1-5-20\Software\Microsoft\Windows NT\CurrentVersion\PrinterPorts\Fax","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Ports\Ne00:","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKCU\Software\Microsoft\Windows NT\CurrentVersion\Devices\Microsoft XPS Document Writer","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKCU\Software\Microsoft\Windows NT\CurrentVersion\PrinterPorts\Microsoft XPS Document Writer","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Ports\Ne01:","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKCU\Software\Microsoft\Windows NT\CurrentVersion\Devices\Fax","-1"

"file","18/8/2016 17:7:12.416","C:\Windows\System32\svchost.exe","Write","C:\ProgramData\Microsoft\Network\Downloader\qmgr0.dat","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKCU\Software\Microsoft\Windows NT\CurrentVersion\PrinterPorts\Fax","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Ports\Ne00:","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKU\.DEFAULT\Software\Microsoft\Windows NT\CurrentVersion\Devices\Microsoft XPS Document Writer","-1"

"file","18/8/2016 17:7:12.466","C:\Windows\System32\svchost.exe","Write","C:\ProgramData\Microsoft\Network\Downloader\qmgr0.dat","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKU\.DEFAULT\Software\Microsoft\Windows NT\CurrentVersion\PrinterPorts\Microsoft XPS Document Writer","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Ports\Ne01:","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKU\.DEFAULT\Software\Microsoft\Windows NT\CurrentVersion\Devices\Fax","-1"

"registry","18/8/2016 17:7:13.387","C:\Windows\System32\spoolsv.exe","SetValueKey","HKU\.DEFAULT\Software\Microsoft\Windows NT\CurrentVersion\PrinterPorts\Fax","-1"

"registry","18/8/2016 17:7:14.689","C:\Program Files\Internet Explorer\iexplore.exe","DeleteValueKey","HKCU\Software\Microsoft\Internet Explorer\LowRegistry\AddToFavoritesInitialSelection","-1"

"registry","18/8/2016 17:7:14.689","C:\Program Files\Internet Explorer\iexplore.exe","DeleteValueKey","HKCU\Software\Microsoft\Internet Explorer\LowRegistry\AddToFeedsInitialSelection","-1"

Figure 4B: Our Windows 7 process monitor exclusion list.

These exclusion lists was the artefact created and used within the creation of the filtered datasets.

```
****************************************************************************************************************************************************
#[+,-]	[Process Created]	[Parent Process]	[Process Path]
#####################################################
### Clean Windows XP SP 2 System		###
#####################################################
#issue in the way process path information is communicated to capture client
+	UNKNOWN	.*	UNKNOWN
#capture client itself
+	CaptureClient.exe	.*	C:\\Program Files\\Capture\\CaptureClient\.exe
+	CaptureClient.bat	.*	C:\\Program Files\\Capture\\CaptureClient\.bat
+	7za.exe	.*	C:\\Program Files\\Capture\\7za\.exe
#Windows update (it runs even if disabled)
+	wuauclt.exe	.*	C:\\WINDOWS\\system32\\wuauclt\.exe
#
+	savedump.exe	.*	C:\\WINDOWS\\system32\\savedump\.exe
#Standard screensaver
+	logon.scr	.*	C:\\WINDOWS\\system32\\logon\.scr
#defragmenter
+	dfrgntfs.exe	.*	C:\\WINDOWS\\system32\\dfrgntfs\.exe
+	defrag.exe	.*	C:\\WINDOWS\\system32\\defrag\.exe
#7za
+	7za.exe	.*	C:\\program Files\\capture\\7za\.exe
```

```
#mapping
+       wmiadap.exe     .*      C:\\WINDOWS\\system32\\wbem\\wmiadap\.exe
+       wmiprvse.exe    .*      C:\\WINDOWS\\system32\\wbem\\wmiprvse\.exe
#vmware tools
+       VMwareUser.exe  .*      C:\\Program Files\\VMware\\VMware Tools\\VMwareUser\.exe
#####################################################
### Microsoft Internet Explorer 6.0                 ###
#####################################################
+       iexplore.exe    .*      C:\\Program Files\\Internet Explorer\\iexplore.exe
+       IEXPLORE.EXE    .*      C:\\Program Files\\Internet Explorer\\IEXPLORE.EXE
#agent server is an activeX control that starts upon displaying multimedia content
+       agentsvr.exe    .*      C:\\WINDOWS\\msagent\\agentsvr.exe
#messenger activeX
+       msmsgs.exe      .*      C:\\Program Files\\Messenger\\msmsgs.exe
+       rundll32.exe    .*      C:\\WINDOWS\\system32\\rundll32.exe
#imapi
+       imapi.exe .*    c:\\WINDOWS\\system32\\imapi\.exe
```

Figure 4C: Our Windows 7 File monitor exclusion list.

```
**********************************************************************************************************************************************************
#[+,-]	[File Access]	[Process Name]	[File Path]
#####################################################
### Clean Windows XP SP 2 System		###
#####################################################
+	Read	.*	.*
+	Create	.*	.*
+	Open	.*	.*
#issue in the way process path information is communicated to capture client
+	Write	UNKNOWN	.*
+	Delete	UNKNOWN	.*
#capture
+	Write	.*	C:\\program files\\capture\\logs\\.+
+	Delete	C:\\program Files\\capture\\capturecleint\.exe	C:\\program files\\capture\\.+\.zip
+	Delete	C:\\program Files\\capture\\capturecleint\.exe	C:\\program files\\capture\\logs
+	Delete	C:\\program Files\\capture\\capturecleint\.exe	C:\\program files\\capture\\logs\\.*
+	Write	C:\\program Files\\capture\\capturecleint\.exe	C:\\program files\\capture\\capture\.log
+	Write	C:\\program Files\\capture\\7za\.exe	C:\\program files\\capture\\capture\.log
+	Write	C:\\program Files\\capture\\7za\.exe	C:\\program files\\capture\\.+\.zip
+	Delete	C:\\program Files\\capture\\7za\.exe	C:\\program files\\capture\\.+\.zip
+	Write	C:\\program Files\\capture\\7za\.exe	C:\\progra~1\\capture\\capture\.log
+	Write	C:\\program Files\\capture\\7za\.exe	C:\\progra~1\\capture\\.+\.zip
+	Delete	C:\\program Files\\capture\\7za\.exe	C:\\progra~1\\capture\\.+\.zip
#Prefetch
```

```
+	Write	C:\\WINDOWS\\system32\\svchost\.exe	C:\\WINDOWS\\Prefetch\\.+
+	Write	System	C:\\WINDOWS\\Prefetch\\.+
#NTFS Metadata
+	Write	.*	c:\\\$mft
+	Write	.*	c:\\\$mftmirr
+	Write	.*	c:\\\$logfile
+	Write	.*	c:\\\$volume
+	Write	.*	c:\\\$directory
+	Write	.*	c:\\\$AttrDef
+	Write	.*	c:\\\$boot
+	Write	.*	c:\\\$bitmap
+	Write	.*	c:\\\$badclus
+	Write	.*	c:\\\$quota
+	Write	.*	c:\\\$upcase
+	Write	.*	c:\\\$ReplaceAttribute2
+	Write	.*	c:\\\$converttononresident
#Performance
+	Write	C:\\WINDOWS\\system32\\wbem\\wmiadap\.exe	C:\\WINDOWS\\system32\\wbem\\Performance\\.+
+	Write	C:\\WINDOWS\\system32\\wbem\\wmiadap\.exe	C:\\WINDOWS\\system32\\Perf.*
+	Write	C:\\WINDOWS\\system32\\svchost\.exe	C:\\WINDOWS\\Prefetch\\.+
+	Write	System	C:\\WINDOWS\\Prefetch\\.+
#System Log Files
+	Write	System	C:\\Documents and Settings\\.+\\.+\.LOG
+	Write	System	C:\\WINDOWS\\system32\\config\\.+\.LOG
+	Write	System	C:\\WINDOWS\\Debug\\UserMode\\userenv\.log
+	Write	System	C:\\WINDOWS\\SoftwareDistribution\\ReportingEvents\.log
```

```
+	Write	C:\\WINDOWS\\system32\\winlogon\.exe	C:\\WINDOWS\\Debug\\UserMode\\userenv\.log
+	Write	C:\\WINDOWS\\system32\\svchost\.exe	C:\\WINDOWS\\.+\.log
+	Write	C:\\WINDOWS\\system32\\lsass\.exe	C:\\WINDOWS\\system32\\config\\.+\.LOG
+	Write	C:\\WINDOWS\\system32\\lsass\.exe	C:\\WINDOWS\\system32\\config\\SAM
+	Write	C:\\WINDOWS\\system32\\lsass\.exe	C:\\WINDOWS\\system32\\config\\system
+	Write	C:\\WINDOWS\\system32\\lsass\.exe	C:\\WINDOWS\\system32\\config\\SECURITY
+	Write	C:\\WINDOWS\\system32\\wbem\\wmiprvse\.exe	C:\\WINDOWS\\system32\\wbem\\Logs\\wmiprov\.log
#Windows update
+	Write	C:\\WINDOWS\\system32\\wuauclt\.exe	C:\\WINDOWS\\SoftwareDistribution\\DataStore\\Logs\\.+
+	Write	C:\\WINDOWS\\system32\\wuauclt\.exe	C:\\WINDOWS\\WindowsUpdate\.log
+	Write	C:\\WINDOWS\\system32\\wuauclt\.exe	C:\\WINDOWS\\SoftwareDistribution\\DataStore\\DataStore\.edb
+	Delete	C:\\WINDOWS\\system32\\wuauclt\.exe	C:\\WINDOWS\\SoftwareDistribution\\DataStore\\Logs\\.+
+	Delete	C:\\WINDOWS\\system32\\wuauclt\.exe	C:\\WINDOWS\\WindowsUpdate\.log
+	Delete	C:\\WINDOWS\\system32\\wuauclt\.exe	C:\\WINDOWS\\SoftwareDistribution\\DataStore\\DataStore\.edb
#System Events
+	Write	C:\\WINDOWS\\system32\\services\.exe	C:\\WINDOWS\\system32\\config\\AppEvent\.Evt
+	Write	C:\\WINDOWS\\system32\\services\.exe	C:\\WINDOWS\\system32\\config\\SysEvent\.Evt
+	Write	C:\\WINDOWS\\system32\\services\.exe	C:\\WINDOWS\\system32\\config\\SecEvent\.Evt
#Mapping
+	Write	C:\\WINDOWS\\system32\\svchost\.exe	C:\\WINDOWS\\system32\\wbem\\.+
#Cataloging
+	Write	C:\\WINDOWS\\system32\\svchost\.exe	C:\\WINDOWS\\system32\\CatRoot2\\.+
+	Write	C:\\WINDOWS\\system32\\svchost\.exe	C:\\WINDOWS\\system32\\CatRoot\\.+
#System restore
+	Write	C:\\WINDOWS\\system32\\svchost\.exe	C:\\WINDOWS\\SoftwareDistribution\\WuRedir\\.+
+	Write	C:\\WINDOWS\\system32\\svchost\.exe	C:\\System Volume Information\\_restore.*
```

```
#user data
+	Write	System	C:\\Documents and Settings\\.+\\Local Settings\\Application Data\\Microsoft\\Windows\\UsrClass\.dat
#######################################################
### Internet Explorer 6.0 SP2                       ###
#######################################################
#somehow VMwareService & System accesses the same files when IE is browsing.
+	Write	C:\\Program Files\\VMware\\VMware Tools\\VMwareService\.exe	.*
+	Write	System	.*
# IE Temporary Files/Internet Cache.
+	Write	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\WINDOWS\\Temp\\.+
+	Write	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Local Settings\\Temporary Internet Files\\Content\.IE5\\.+
+	Write	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Local Settings\\Temp\\.+tmp
+	Delete	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\WINDOWS\\Temp\\.+
+	Delete	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Local Settings\\Temporary Internet Files\\Content\.IE5\\.+
+	Delete	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Local Settings\\Temp\\.+tmp
# History
+	Write	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Local Settings\\History\\History.IE5\\.+
+	Delete	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Local Settings\\History\\History.IE5\\.+
# IE Cookies
+	Write	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Cookies\\.+
+	Write	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Cookies\\index.dat
+	Delete	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Cookies\\.+
+	Delete	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Cookies\\index.dat
# User data
+	Write	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Application Data\\Microsoft\\CryptnetUrlCache
+	Write	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\UserData\\.+
```

```
+	Delete	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Application Data\\Microsoft\\CryptnetUrlCache
+	Delete	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\UserData\\.+
# Plug ins (like Flash player)
+	Write	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Application Data\\.+
+	Delete	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\Application Data\\.+
# DRM related stuff
+	Write	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\DRM\\.+
+	Delete	C:\\Program Files\\Internet Explorer\\iexplore\.exe	C:\\Documents and Settings\\.+\\DRM\\.+
# msg activeX
+	Write	C:\\Program Files\\Messenger\\msmsgs\.exe	C:\\Documents and Settings\\.+\\NTUSER.DAT.LOG
+	Delete	C:\\Program Files\\Messenger\\msmsgs\.exe	C:\\Documents and Settings\\.+\\NTUSER.DAT.LOG
##################################################
### Minus List - General Malicious Activity	###
##################################################
# Alert about executables or scripts that are written to disk
#-	Write	.*	.+\.bat
#-	Write	.*	.+\.cmd
#-	Write	.*	.+\.exe
#-	Write	.*	.+\.inf
#-	Write	.*	.+\.lnk
#-	Write	.*	.+\.msi
#-	Write	.*	.+\.msp
#-	Write	.*	.+\.pif
#-	Write	.*	.+\.reg
#-	Write	.*	.+\.sct
#-	Write	.*	.+\.shs
```

```
#-      Write   .*      .+\.scr
#-      Write   .*      .+\.wsc
#-      Write   .*      .+\.wsf
#-      Write   .*      .+\.wsh
#commented out for IE because \.com cache files and \.vb script files are very common
#-      Write   .*      .+\.vb
#-      Write   .*      .+\.com
# Alert about modifications to startup locations
-       Write   .*      C:\\Documents and Settings\\.+\\Start Menu\\Programs\\Startup.+
-       Write   .*      C:\\WINDOWS\\win.ini
-       Write   .*      C:\\WINDOWS\\Tasks\\.+
```

Figure 4D: Our Windows 7 Registry monitor exclusion list.

```
**********************************************************************************************************************************************************
#[+,-]	[Registry Event]	[Process Name]	[Registry Path]
#####################################################
### Microsoft Windows XP SP2                      ###
#####################################################
+	OpenKey .*	.*
+	CreateKey	.*	.*
+	CloseKey .*	.*
+	EnumerateKey	.*	.*
+	EnumerateValueKey	.*	.*
+	QueryValueKey	.*	.*
+	QueryKey.*	.*
#issue in the way process path information is communicated to capture client
+	SetValueKey	UNKNOWN	.*
+	DeleteValueKey	UNKNOWN	.*
+	SetValueKey	.*	HKU\\.+\\SessionInformation\\ProgramCount
+	SetValueKey	.*	HKCU\\Software\\Microsoft\\Windows\\ShellNoRoam.*
+	SetValueKey	.*	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Installer\\UserData\\.+
+	SetValueKey	.*	HKLM\\SOFTWARE\\Microsoft\\Cryptography\\RNG\\Seed.*
+	SetValueKey	C:\\Program Files\\Capture\\CaptureClient.exe	HKLM\\SYSTEM\\ControlSet001\\Services\\nm\\Parameters\\.+
+	SetValueKey	C:\\WINDOWS\\explorer.exe	HKCU\\SessionInformation\\.+
+	SetValueKey	C:\\WINDOWS\\explorer.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\.+
+	SetValueKey	C:\\WINDOWS\\explorer.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Explorer\\.+
+	SetValueKey	C:\\WINDOWS\\explorer.exe	HKU\\.+\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\.+
```

```
+	SetValueKey	C:\\WINDOWS\\explorer.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Cache\\Paths\\.+
+	SetValueKey	C:\\WINDOWS\\explorer.exe	HKU\\.+\\Software\\Microsoft\\Windows\\ShellNoRoam\\BagMRU\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\winlogon.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Group Policy\\State\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\svchost.exe	\\REGISTRY\\USER\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKU\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKCU\\Software\\Microsoft\\SystemCertificates\\Root\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKLM\\SOFTWARE\\Microsoft\\EAPOL\\Parameters\\General\\InterfaceList
+	SetValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKLM\\SOFTWARE\\Microsoft\\SystemCertificates\\AuthRoot\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKLM\\SOFTWARE\\Microsoft\\PCHealth\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKLM\\SYSTEM\\ControlSet001\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\services.exe	HKLM\\SYSTEM\\ControlSet001\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\lsass.exe	HKLM\\SECURITY\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\lsass.exe	HKLM\\SYSTEM\\ControlSet001\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\lsass.exe	HKCU\\Software\\Microsoft\\Protected Storage System Provider\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\wbem\\wmiadap.exe	HKLM\\SOFTWARE\\Microsoft\\WBEM\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\wbem\\wmiadap.exe	HKLM\\SYSTEM\\ControlSet001\\Services\\WmiApRpl\\Performance\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\wbem\\wmiadap.exe	HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Perflib\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\wbem\\wmiprvse.exe	HKLM\\SOFTWARE\\Microsoft\\WBEM\\WDM\\.+
+	DeleteValueKey	.*	HKU\\.+\\SessionInformation\\ProgramCount
+	DeleteValueKey	.*	HKCU\\Software\\Microsoft\\Windows\\ShellNoRoam.*
+	DeleteValueKey	.*	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Installer\\UserData\\.+
+	DeleteValueKey	.*	HKLM\\SOFTWARE\\Microsoft\\Cryptography\\RNG\\Seed.*
+	DeleteValueKey	C:\\Program Files\\Capture\\CaptureClient.exe	HKLM\\SYSTEM\\ControlSet001\\Services\\nm\\Parameters\\.+
```

```
+	DeleteValueKey	C:\\WINDOWS\\explorer.exe	HKCU\\SessionInformation\\.+
+	DeleteValueKey	C:\\WINDOWS\\explorer.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\.+
+	DeleteValueKey	C:\\WINDOWS\\explorer.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Explorer\\.+
+	DeleteValueKey	C:\\WINDOWS\\explorer.exe	HKU\\.+\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\.+
+	DeleteValueKey	C:\\WINDOWS\\explorer.exe	HKU\\.+\\Software\\Microsoft\\Windows\\ShellNoRoam\\BagMRU\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\winlogon.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Group Policy\\State\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\svchost.exe	\\REGISTRY\\USER\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKU\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKCU\\Software\\Microsoft\\SystemCertificates\\Root\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKLM\\SOFTWARE\\Microsoft\\SystemCertificates\\AuthRoot\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKLM\\SOFTWARE\\Microsoft\\PCHealth\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKCU\\Software\\Microsoft\\SystemCertificates\\Root\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\svchost.exe	HKLM\\SYSTEM\\ControlSet001\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\services.exe	HKLM\\SYSTEM\\ControlSet001\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\lsass.exe	HKLM\\SECURITY\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\lsass.exe	HKCU\\Software\\Microsoft\\Protected Storage System Provider\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\wbem\\wmiadap.exe	HKLM\\SOFTWARE\\Microsoft\\WBEM\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\wbem\\wmiadap.exe	HKLM\\SYSTEM\\ControlSet001\\Services\\WmiApRpl\\Performance\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\wbem\\wmiadap.exe	HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Perflib\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\wbem\\wmiprvse.exe	HKLM\\SOFTWARE\\Microsoft\\WBEM\\WDM\\.+
#defrag
+	SetValueKey	C:\\WINDOWS\\system32\\dfrgntfs.exe	HKLM\\SOFTWARE\\Microsoft\\Dfrg.*
+	DeleteValueKey	C:\\WINDOWS\\system32\\dfrgntfs.exe	HKLM\\SOFTWARE\\Microsoft\\Dfrg.*
```

```
#windows update
+       SetValueKey     C:\\WINDOWS\\system32\\wuauclt.exe   HKLM\\SYSTEM\\ControlSet001\\Services\\Eventlog\\Application\\ESENT\\.+
+       DeleteValueKey  C:\\WINDOWS\\system32\\wuauclt.exe   HKLM\\SYSTEM\\ControlSet001\\Services\\Eventlog\\Application\\ESENT\\.+
##################################################
### Internet Explorer 6.0 SP2                  ###
##################################################
+       OpenKey .*      .*
+       CreateKey       .*      .*
+       CloseKey .*     .*
+       EnumerateKey    .*      .*
+       EnumerateValueKey       .*      .*
+       QueryValueKey   .*      .*
+       QueryKey.*      .*
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\EUDC\\.+
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\Software\\Microsoft\\Internet Explorer\\Main\\Window_Placement
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\Software\\Microsoft\\Internet Explorer\\Main\\Fullscreen
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\Software\\Microsoft\\Internet Explorer\\Main\\NotificationDownloadComplete
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\Software\\Microsoft\\Internet Explorer\\TypedURLs
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\Software\\Microsoft\\Internet Explorer\\Toolbar\\Locked
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\Software\\Microsoft\\Internet Explorer\\International\\.+
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\Software\\Microsoft\\Internet Explorer\\Security\\P3Global\\Enabled
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\Software\\Microsoft\\Internet Explorer\\Extensions\\CmdMapping\\.+
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\Software\\Microsoft\\Internet Explorer\\PageSetup\\.+
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\MenuOrder\\.+
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\MountPoints2\\.+
+       SetValueKey     C:\\Program Files\\Internet Explorer\\iexplore\.exe  HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\UserAssist\\.+
```

+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\Shell Folders\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\CabinetState\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ZoneMap\\UNCAsIntranet
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Internet Explorer\\Toolbar\\WebBrowser\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ZoneMap\\IntranetName
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ZoneMap\\AutoDetect
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ZoneMap\\ProxyBypass
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ZoneMap\\UNCAsIntranet
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\MigrateProxy
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ProxyEnable
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ProxyServer
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Cache.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Connections\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\5.0\\Cache\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Ext\\Stats\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\ShellNoRoam\\BagMRU.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\ShellNoRoam\\Bags.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Ext\\Stats\\.+\\iexplore\\(Count|Time|Type)
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon\\ParseAutoexec
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\SystemCertificates\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Classes\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Cache.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\5.0\\Cache.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Cache.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Passport\\.+

+ SetValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Explorer\\Shell Folders\\.+

+ SetValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKLM\\SOFTWARE\\Microsoft\\Direct3D.+

+ SetValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKLM\\SOFTWARE\\Microsoft\\DirectDraw.+

+ SetValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKLM\\SOFTWARE\\Microsoft\\Cryptography\\RNG\\Seed

+ SetValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKLM\\SOFTWARE\\Microsoft\\AudioCompressionManager\\.+

+ SetValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKLM\\SOFTWARE\\Microsoft\\SystemCertificates\\.+

+ SetValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Cache.+

+ SetValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\5.0\\Cache.

+ SetValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKLM\\SYSTEM\\ControlSet001\\Hardware Profiles\\0001\\Software\\Microsoft\\windows\\CurrentVersion\\Internet Settings\\ProxyEnable

+ SetValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKLM\\SYSTEM\\ControlSet001\\Services\\EventLog\\.+

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\EUDC\\.+

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Internet Explorer\\Main\\Window_Placement

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Internet Explorer\\Main\\Fullscreen

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Internet Explorer\\Main\\NotificationDownloadComplete

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Internet Explorer\\TypedURLs

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Internet Explorer\\Toolbar\\Locked

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Internet Explorer\\International\\.+

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Internet Explorer\\Security\\P3Global\\Enabled

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Internet Explorer\\Extensions\\CmdMapping\\.+

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Internet Explorer\\PageSetup\\.+

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\MenuOrder\\.+

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\MountPoints2\\.+

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\UserAssist\\.+

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\Shell Folders\\.+

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\CabinetState\\.+

+ DeleteValueKey C:\\Program Files\\Internet Explorer\\iexplore\.exe HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ZoneMap\\UNCAsIntranet

+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ZoneMap\\IntranetName
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ZoneMap\\AutoDetect
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ZoneMap\\ProxyBypass
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ZoneMap\\UNCAsIntranet
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\MigrateProxy
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ProxyEnable
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ProxyServer
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Cache.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Connections\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\5.0\\Cache\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Ext\\Stats\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\ShellNoRoam\\BagMRU.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\ShellNoRoam\\Bags.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Ext\\Stats\\.+\\iexplore\\(Count|Time|Type)
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon\\ParseAutoexec
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\SystemCertificates\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Classes\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Cache.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\5.0\\Cache.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Cache.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Passport\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Explorer\\Shell Folders\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\Direct3D.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\DirectDraw.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\Cryptography\\RNG\\Seed

```
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\AudioCompressionManager\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SOFTWARE\\Microsoft\\SystemCertificates\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Cache.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\5.0\\Cache.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SYSTEM\\ControlSet001\\Hardware	Profiles\\0001\\Software\\Microsoft\\windows\\CurrentVersion\\Internet Settings\\ProxyEnable
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKLM\\SYSTEM\\ControlSet001\\Services\\EventLog\\.+
+	DeleteKey	.*	.*
#Plugins
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Scrunch\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\MediaPlayer\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows Media\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Multimedia\\ActiveMovie\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\ActiveMovie\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\MPEG2Demultiplexer\\.+
+	SetValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Multimedia\\msacm.imaadpcm\\.+
+	SetValueKey	C:\\WINDOWS\\msagent\\agentsvr\.exe	HKLM\\SOFTWARE\\Microsoft\\AudioCompressionManager\\DriverCache\\msacm.msadpcm\\.+
+	SetValueKey	C:\\WINDOWS\\msagent\\agentsvr\.exe	HKLM\\SOFTWARE\\Microsoft\\Microsoft Agent\\.+
+	SetValueKey	C:\\Program Files\\Messenger\\msmsgs\.exe	HKCU\\AppEvents\\Schemes\\Apps\\MSMSGS.*
+	SetValueKey	C:\\Program Files\\Messenger\\msmsgs\.exe	HKCU\\AppEvents\\EventLabels\\MSMsgs.+
+	SetValueKey	C:\\Program Files\\Messenger\\msmsgs\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\MSMSGS
+	SetValueKey	C:\\Program Files\\Messenger\\msmsgs\.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\.+
+	SetValueKey	C:\\Program Files\\Messenger\\msmsgs\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\Shell Folders\\.+
+	SetValueKey	C:\\Program Files\\Messenger\\msmsgs\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\.+
+	SetValueKey	C:\\WINDOWS\\system32\\svchost\.exe	HKLM\\SOFTWARE\\Microsoft\\EventSystem\\.+\\Subscriptions\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Scrunch\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\MediaPlayer\\.+
```

```
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Windows Media\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Multimedia\\ActiveMovie\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\ActiveMovie\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\MPEG2Demultiplexer\\.+
+	DeleteValueKey	C:\\Program Files\\Internet Explorer\\iexplore\.exe	HKCU\\Software\\Microsoft\\Multimedia\\msacm.imaadpcm\\.+
+	DeleteValueKey	C:\\WINDOWS\\msagent\\agentsvr\.exe	HKLM\\SOFTWARE\\Microsoft\\AudioCompressionManager\\DriverCache\\msacm.msadpcm\\.+
+	DeleteValueKey	C:\\WINDOWS\\msagent\\agentsvr\.exe	HKLM\\SOFTWARE\\Microsoft\\Microsoft Agent\\.+
+	DeleteValueKey	C:\\Program Files\\Messenger\\msmsgs\.exe	HKCU\\AppEvents\\EventLabels\\MSMsgs.+
+	DeleteValueKey	C:\\Program Files\\Messenger\\msmsgs\.exe	HKCU\\AppEvents\\Schemes\\Apps\\MSMSGS.*
+	DeleteValueKey	C:\\Program Files\\Messenger\\msmsgs\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\MSMSGS
+	DeleteValueKey	C:\\Program Files\\Messenger\\msmsgs\.exe	HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\.+
+	DeleteValueKey	C:\\Program Files\\Messenger\\msmsgs\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\Shell Folders\\.+
+	DeleteValueKey	C:\\Program Files\\Messenger\\msmsgs\.exe	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\.+
+	DeleteValueKey	C:\\WINDOWS\\system32\\svchost\.exe	HKLM\\SOFTWARE\\Microsoft\\EventSystem\\.+\\Subscriptions\\.+
##################################################
### Minus List - General Malicious Activity	###
##################################################
#Any modification to start/bootup sequence
-	SetValueKey	.*	HLKM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run.*
-	DeleteValueKey	.*	HLKM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run.*
-	SetValueKey	.*	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run.*
-	DeleteValueKey	.*	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run.*
-	SetValueKey	.*	HKCU\\Software\\Microsoft\\Windows NT\\CurrentVersion\\Windows\\Run.*
-	DeleteValueKey	.*	HKCU\\Software\\Microsoft\\Windows NT\\CurrentVersion\\Windows\\Run.*
-	SetValueKey	.*	HKCU\\Software\\Microsoft\\Windows NT\\CurrentVersion\\Windows\\Load.*
-	DeleteValueKey	.*	HKCU\\Software\\Microsoft\\Windows NT\\CurrentVersion\\Windows\\Load.*
```

-	SetValueKey	.*	HLKM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon\\Userinit.*
-	DeleteValueKey	.*	HLKM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon\\Userinit.*
-	SetValueKey	.*	HLKM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon\\Shell.*
-	DeleteValueKey	.*	HLKM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon\\Shell.*
-	SetValueKey	.*	HLKM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Policies\\Explorer\\Run.*
-	DeleteValueKey	.*	HLKM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Policies\\Explorer\\Run.*
-	SetValueKey	.*	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Policies\\Explorer\\Run.*
-	DeleteValueKey	.*	HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Policies\\Explorer\\Run.*
-	SetValueKey	.*	HLKM\\SYSTEM\\CurrentControlSet\\Control\\Session Manager\\BootExecute.*
-	DeleteValueKey	.*	HLKM\\SYSTEM\\CurrentControlSet\\Control\\Session Manager\\BootExecute.*
-	SetValueKey	.*	HLKM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\ShellServiceObjectDelayLoad\\.*
-	DeleteValueKey	.*	HLKM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\ShellServiceObjectDelayLoad\\.*

Table 4E: Process and default path table.

Default path for executable	Executable Name
C:\program Files\capture\captureclient.exe	Captureclient.exe
C:\program Files\capture\7za.exe	7za.exe
C:\Windows\system32\svchost.exe	Svchost.exe
C:\Windows\system32\winlogon.exe	Winlogon.exe
C:\Windows\system32\wbem\wmiadap.exe	Wmiadap.exe
C:\Windows\system32\services.exe	Services.exe
C:\Windows\system32\ wuauclt.exe	Wuauclt.exe
C:\Windows\system32\lsass.exe	lsass.exe
C:\Windows\System32\SearchIndexer.exe	SearchIndexer.exe
C:\Windows\System32\sppsvc	Sppsvc.exe
C:\Program Files\Internet Explorer\iexplore.exe	iexplore.exe
C:\Program Files\Capture\CaptureClient.exe	CaptureClient.exe
C:\Program Files\Capture\CaptureClient\.bat	CaptureClient.bat

C:\Program Files\Capture\7za.exe	7za.exe
C:\Windows\system32\wuauclt.exe	wuauclt.exe
C:\Windows\system32\savedump.exe	savedump.exe
C:\Windows\system32\logon\.scr	logon.scr
C:\Windows\system32\dfrgntfs.exe	dfrgntfs.exe
C:\Windows\system32\defrag.exe	defrag.exe
C:\Windows\system32\wbem\wmiadap.exe	wmiadap.exe
C:\Windows\system32\wbem\wmiprvse.exe	wmiprvse.exe
C:\Program Files\VMware\VMware Tools\VMwareUser.exe	VMwareUser.exe
C:\Windows\System32\svchost.exe	svchost.exe
C:\Windows\System32\dllhost.exe	dllhost.exe
C:\Windows\System32\taskhost.exe	taskhost.exe
C:\Windows\System32\taskeng.exe	taskeng.exe
C:\Windows\System32\SearchProtocolHost.exe	SearchProtocolHost.exe
C:\Windows\System32\SearchFilterHost.exe	SearchFilterHost.exe
C:\Windows\System32\SearchIndexer.exe	SearchIndexer.exe
C:\Windows\System32\winlogon.exe	winlogon.exe
C:\Windows\System32\userinit.exe	userinit.exe
C:\Windows\System32\csrss.exe	csrss.exe

C:\Windows\System32\conhost.exe	conhost.exe
C:\Windows\System32\mobsync.exe	mobsync.exe
C:\Windows\System32\sppsvc.exe	SPPSVC.exe
C:\Windows\System32\wsqmcons.exe	wsqmcons.exe
C:\Windows\System32\sdclt.exe	Sdclt.exe
C:\Windows\System32\sc.exe	sc.exe
C:\Windows\System32\drvinst.exe	drvinst.exe
C:\Windows\System32\WerFault.exe	WerFault.exe
C:\Program Files\Internet Explorer\iexplore.exe	Iexplorer.exe
C:\Windows\msagent\agentsvr.exe	agentsvr.exe
C:\Windows\system32\rundll32.exe	rundll32.exe
c:\Windows\system32\imapi.exe	imapi.exe

(*Refer http://kb.digital-detective.net/display/BF/Location+of+Internet+Explorer+Data)

Appendix C

Table 5A: Full list of applied updates for experimental transparency.

Type	Hot Fix ID
Security Update	KB2479943
Security Update	KB2491683
Security Update	KB2503665
Security Update	KB2506212
Security Update	KB2509553
Security Update	KB2510531
Security Update	KB2511455
Update	KB2533552
Hotfix	KB2534111
Security Update	KB2544893
Update	KB2552343
Security Update	KB2560656
Security Update	KB2564958
Security Update	KB2570947
Security Update	KB2579686
Security Update	KB2585542
Security Update	KB2604115
Security Update	KB2620704
Security Update	KB2621440
Security Update	KB2631813
Security Update	KB2653956
Security Update	KB2654428
Security Update	KB2656356
Security Update	KB2667402
Security Update	KB2676562

Security Update	KB2685939
Security Update	KB2690533
Security Update	KB2698365
Security Update	KB2705219
Update	KB2718704
Security Update	KB2727528
Security Update	KB2729452
Security Update	KB2736422
Security Update	KB2742599
Security Update	KB2758857
Security Update	KB2770660
Update	KB2786081
Security Update	KB2789645
Update	KB2798162
Security Update	KB2807986
Security Update	KB2813430
Update	KB2836942
Update	KB2836943
Security Update	KB2840149
Security Update	KB2840631
Security Update	KB2847927
Security Update	KB2861698
Security Update	KB2862152
Security Update	KB2862330
Security Update	KB2862335
Security Update	KB2864202
Security Update	KB2868038
Update	KB2868116
Security Update	KB2871997
Security Update	KB2884256
Security Update	KB2892074

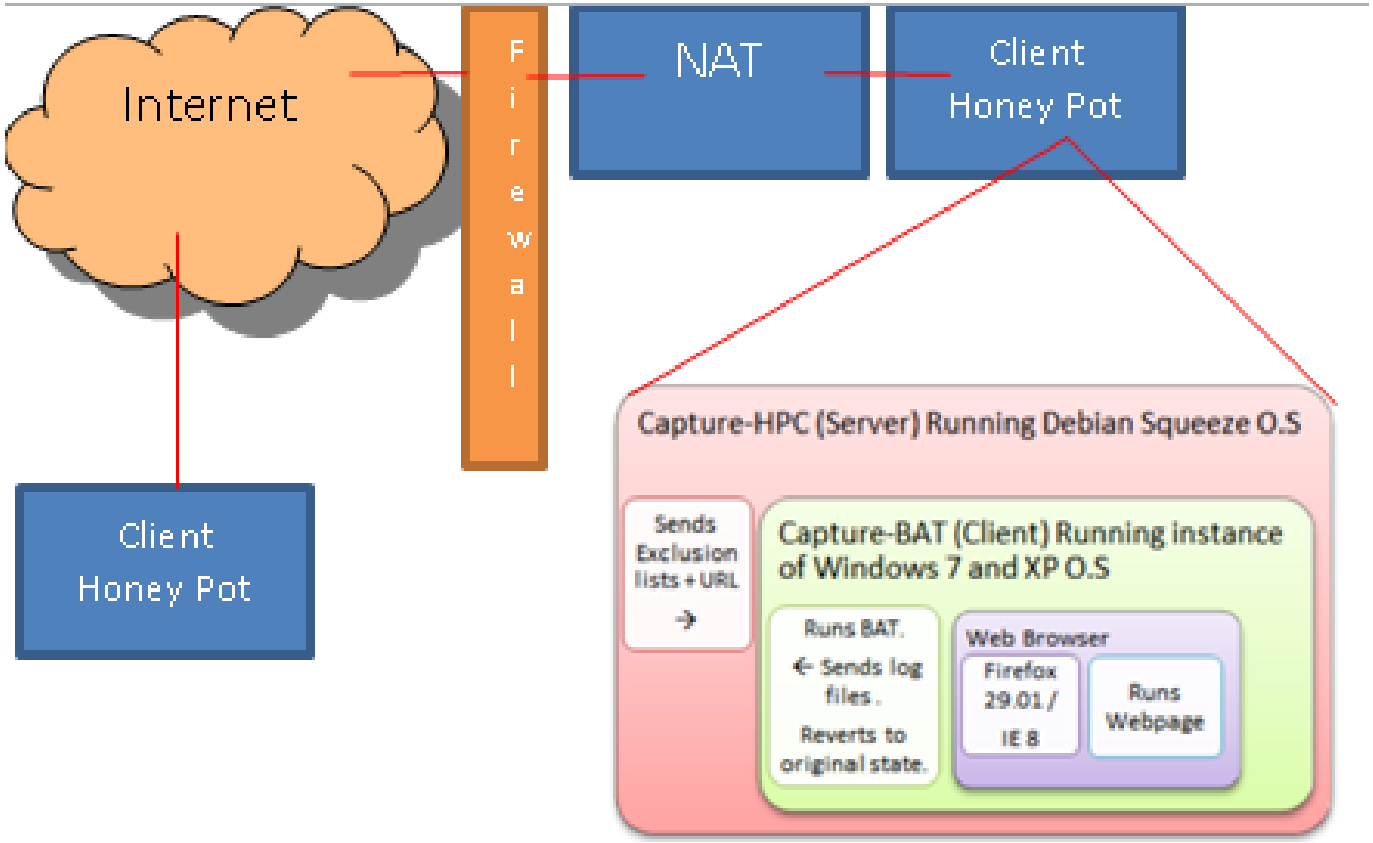
Security Update	KB2893294
Security Update	KB2894844
Security Update	KB2900986
Security Update	KB2911501
Update	KB2929733
Security Update	KB2931356
Security Update	KB2937610
Security Update	KB2943357
Security Update	KB2957189
Security Update	KB2965788
Security Update	KB2968294
Security Update	KB2972100
Security Update	KB2972211
Security Update	KB2973112
Security Update	KB2973201
Security Update	KB2973351
Security Update	KB2977292
Security Update	KB2978120
Security Update	KB2984972
Security Update	KB2991963
Security Update	KB2992611
Update	KB2993651
Security Update	KB3003743
Security Update	KB3004361
Security Update	KB3004375
Security Update	KB3005607
Security Update	KB3010788
Security Update	KB3011780
Security Update	KB3019978
Update	KB3020369
Security Update	KB3021674

Security Update	KB3022777
Security Update	KB3023215
Security Update	KB3030377
Security Update	KB3033889
Security Update	KB3033929
Security Update	KB3035126
Security Update	KB3035132
Security Update	KB3037574
Security Update	KB3042058
Security Update	KB3042553
Security Update	KB3045685
Security Update	KB3046017
Security Update	KB3046269
Security Update	KB3055642
Security Update	KB3059317
Security Update	KB3060716
Security Update	KB3061518
Security Update	KB3067904
Security Update	KB3071756
Security Update	KB3072305
Security Update	KB3074543
Security Update	KB3075220
Security Update	KB3076895
Security Update	KB3076949
Security Update	KB3078601
Security Update	KB3080446
Security Update	KB3084135
Security Update	KB3086255
Security Update	KB3087039
Security Update	KB3092601
Security Update	KB3093513

Security Update	KB3097989
Security Update	KB3101722
Security Update	KB3108371
Security Update	KB3108381
Security Update	KB3108664
Security Update	KB3108670
Security Update	KB3109094
Security Update	KB3109103
Security Update	KB3109560
Security Update	KB3110329
Security Update	KB3115858
Security Update	KB3122648
Security Update	KB3123479
Security Update	KB3124275
Security Update	KB3124280
Security Update	KB3126587
Security Update	KB3127220
Security Update	KB3135983
Update	KB3138612
Security Update	KB3139398
Security Update	KB3139914
Security Update	KB3142024
Security Update	KB3142042
Security Update	KB3145739
Security Update	KB3146706
Security Update	KB3146963
Security Update	KB3149090
Security Update	KB3155178
Security Update	KB3156016
Security Update	KB3156017
Security Update	KB3156019

Security Update	KB3159398
Security Update	KB3161561
Security Update	KB3161949
Security Update	KB3161958
Security Update	KB3163245
Security Update	KB3164033
Security Update	KB3164035
Security Update	KB3168965
Security Update	KB3170455
Update	KB3172605
Security Update	KB3175024
Security Update	KB3177186
Security Update	KB3178034
Update	KB3182203
Security Update	KB3184122
Security Update	KB3185911
Update	KB976902

Figure 5B: General setup used in chapter 5 experiments.



Appendix D

Table 6A: Signatures for table 6.1, malicious file writes.

Keys:
Malware with signature
Malware without signature
False positive file

MD5	Time of analysis	Signature
Tmps		
1768605d24117b9ed7d29cfb4ebe5cf6	2016-07-29 17:51:18	Downloader.Generic13.BZCI
1f8f5d16db62a998283ce45d48b67eeb	2016-07-27 21:31:03	Downloader.Generic13.BZCI
server1.exe		
ed5d47f977e201719cbc8e220a6aa034	2016-07-28 17:00:08	BackDoor.MSIL.L
b1b5346c6194670b82a20f864f8f0182	2016-07-26 14:52:09	BackDoor.MSIL.L
ece4e4669fec483da92a28e9ba22c673	2016-07-29 20:33:45	MSIL.AP
45fcea1d7ba007df48ce3c5cd9dc7062	2016-07-28 15:49:49	BackDoor.MSIL.L
9737d7af2dcc9c57629cf56e3cbecba9	2016-07-25 20:09:05	PSW.ILUSpy
2b78deeeca7d00807e120b8c62df332a	2016-07-27 21:47:30	Bladabindi.PDE
1a4ac812a4554f4dc35f3641a5b0ebde	2016-07-28 17:04:00	BackDoor.MSIL.L
e8359aff133f6b4e37c5663b274f97d0	2016-07-29 17:59:22	PSW.ILUSpy
e1113c0609bb3bc7c868656fcb82c015	2016-07-27 21:43:00	PSW.ILUSpy
f5c132c4a5560886b989cd1af59e096a	2016-07-27 22:01:55	BackDoor.MSIL.L
727488c4dbde294a7f39221a2c85e7f7	2016-07-29 17:37:00	PSW.ILUSpy
45f3c9adf8a3417710ae1baaffd9f47a	2016-07-28 15:51:37	MSIL.AL

9579c854f08428bbc0d25c2f65a5f1e8	2016-07-27 22:26:14	BackDoor.MSIL.L
399235636f1aa9aa75d1f246eb2e7b42	2016-07-27 22:06:28	BackDoor.MSIL.L
1bab5cb0fb1c03e7b45e81fe50818983	2016-07-27 21:37:57	BackDoor.MSIL.L
e0737b3d5f94b2c2bd212df5cd8fd9a2	2016-07-28 17:52:50	BackDoor.MSIL.L
1fcf408a1fc46629f26da4cad2dce16a	2016-07-29 20:55:57	BackDoor.MSIL.L
15c241e87e30c0a80174f7bf70d0715a	2016-07-29 21:04:48	PSW.ILUSpy
e84e4536f80e6a77904ded36715d4ce1	2016-07-28 16:13:08	PSW.ILUSpy
321832e46efe46ec64de158e04fd1103	2016-07-26 18:10:50	BackDoor.MSIL.L
b26a02e9e730fb94e98fc46b35eb1d24	2016-07-29 20:49:35	BackDoor.MSIL.L
80ea2c1a63af7202cf58a5bd258b8375	2016-07-27 21:56:01	BackDoor.MSIL.L
a420d4518cec086caefad603c3c24043	2016-07-27 07:00:07	MSIL.AP
42482501e66e6409a714f33c741108aa	2016-07-27 22:11:01	PSW.ILUSpy
047ef0610cf64100080295f18cfca893	2016-07-26 14:48:34	PSW.ILUSpy
02661aa0cfb7ae047c8b2ca250a3836c	2016-07-28 06:41:47	PSW.ILUSpy
7ac7683d6ee05a9c26bf2b5efd8a7d9d	2016-07-27 22:09:55	PSW.ILUSpy
73a6d74b9400468eb87014531a7931ab	2016-07-25 20:33:27	PSW.ILUSpy
dcb5badbecb6689e6dd491f1c22750de	2016-07-28 16:39:10	Packed2_c.HZK
4e0ea6066b70a6f8e88d36f39a5dab48	2016-07-29 17:16:45	MSIL.AL
79af8b4bce3e7aa61aad3f9ba1dc537f	2016-07-27 22:23:16	PSW.ILUSpy
9a98316febcf196cdcba1cdce712d418	2016-07-28 17:40:38	MSIL.AP
0d009d5163474dbbdf7b68f58fd392a7	2016-07-27 04:22:10	PSW.ILUSpy
9baa105ba9cae3b6fff6068ae48592ac	2016-07-27 21:42:03	PSW.ILUSpy
09a9509bd60f5032c52e85faf624e42d	2016-07-28 16:51:42	BackDoor.MSIL.L
12e39b0533510e9d7d675b9acfc2ef4e	2016-07-28 04:09:47	BackDoor.MSIL.L
bac5b1dd33b74ec055eda056f769949d	2016-07-27 22:15:57	BackDoor.MSIL.L
1d8dc035d6b2b77e2b16cdd81731c2e7	2016-07-29 17:47:20	MSIL.AP
f238334a4aeab486a963a84fdc076ce5	2016-07-27 21:43:09	BackDoor.MSIL.L
3596ca8d661bf21c5c25ce2e11b21b38	2016-07-29 20:36:57	-
9df442ae0ff682a59951a24e8ffffcff	2016-07-30 00:17:00	PSW.ILUSpy
9df442ae0ff682a59951a24e8ffffcff	2016-07-30 00:17:00	PSW.ILUSpy
72f088336d2278759c90d760c445302a	2016-07-27 21:42:19	PSW.ILUSpy

3407c6d264ed1d67873cc31b6bfc567c	2016-07-28 16:10:14	PSW.ILUSpy
24aa75a5e90c696a184d3092c36079f4	2016-07-28 16:02:19	BackDoor.MSIL.L
82fbb8785d2424ccc380c4e9705922f8	2016-07-29 17:11:54	PSW.ILUSpy
0fd3d39412ddc35224f1b0f860885255	2016-07-28 16:13:24	BackDoor.Generic18.BINY
eef3ce673431dd7dfb386939e66f987c	2016-07-27 21:38:24	PSW.ILUSpy
ea29382503f2b6b0f8b449d3e357155b	2016-07-27 21:43:55	PSW.ILUSpy
6c4e401791afae6ab2df57acd5b0775e	2016-07-29 21:36:21	PSW.ILUSpy
edf95877a0d21c4deca0cdf9d969e178	2016-07-29 17:09:58	PSW.MSIL.JTO
8c35a1ab7eab7d33bd9b58e248013b22	2016-07-27 21:34:41	PSW.ILUSpy
73dfe7f51996ec998830fcf5cb301daa	2016-07-26 14:39:43	Generic35.CBXI
7e6f4339e9b31b981f5732da25a31718	2016-07-26 15:58:52	PSW.ILUSpy
ce63239f1b26df3026ba9952b5dace5b	2016-07-27 21:35:31	Worm/Generic3.KR
6053353c5b993da7cfea07a681d2c7c0	2016-07-27 22:15:26	BackDoor.MSIL.L
559ebfc16af3ab5c7ae16a9d3af84224	2016-07-28 16:03:32	-
0ed9ced98bbf96fae78b9ac3d7679a3c	2016-07-28 16:55:44	BackDoor.MSIL.L
8eb25343bbf93d3e658d3c511691f3ed	2016-07-29 17:24:48	Worm/Generic3.KR
7edd48daf27984c0eeef924918958af7	2016-07-29 17:31:20	PSW.ILUSpy
d5521deb126f8c36b33cfe7dbfbb50a0	2016-07-29 17:35:27	Generic35.CBXI
4dde874c2512929f6f8e3f2c79c15743	2016-07-30 00:26:10	Generic30.ARGU
95af8329ccc6844c1c465f39b1cdbd46	2016-07-30 00:05:16	Generic35.CBXI
PluginFLashPlayer.exe		
e6b8621b67018ae8554114a2a943d237	2016-07-29 17:13:54	Atros.AQYA
87db7336cff737e71ad36e1a7019238f	2016-07-27 21:22:05	Atros.AQYA
wrar501ru[1].exe		
33a3ba5b1f4a49f60b94a4c62adbcae3	2016-07-28 16:55:21	-
0d53a284515a84c731d172086acfccea	2016-07-30 00:25:27	-
bbc13d82bb211413cb0bfcf4026a3bfe	2016-07-28 16:38:44	-
3cdd6406c4fa32ea512e3a529cd0d711	2016-07-29 20:45:09	-
738747fa656bdc7b7754a99b92b7b2d1	2016-07-26 15:50:16	-

The wrar501ru[1].exe files were Russian copies of the Win Rar program. This was analysed and found to be a false positive file write according to the AV vendors at Virus Total. It is however malicious drive-by-download behaviour that a file at all was downloaded upon mere visitation of a website without Capture-BAT having accepted or initiated the file download. It is likely that webmasters for these sites might have changed their malicious file drop to be this executable instead of a malicious file. This could have been as a result of the analysis environment being discovered.

Table 6B: Malicious process creations within the 5,132 dataset:

Malicious executable process creation list	Count
UNKNOWNcreated3612C:\Windows\System32\schtasks.exe	6
C:\Program Files\Internet Explorer\iexplore.execreated2876C:\Users\mp\AppData\Local\Temp\svchost.exe	5
C:\Program Files\Internet Explorer\iexplore.execreated2856C:\Users\mp\AppData\Local\Temp\svchost.exe	5
UNKNOWNcreated3832C:\Windows\System32\sc.exe	5
C:\Program Files\Microsoft\DesktopLayer.execreated2892C:\Program Files\Google\Chrome\Application\chrome.exe	4
C:\Program Files\Internet Explorer\iexplore.execreated2904C:\Users\mp\AppData\Local\Temp\svchost.exe	4
C:\Program Files\Internet Explorer\iexplore.execreated2848C:\Users\mp\AppData\Local\Temp\svchost.exe	4
C:\Windows\System32\svchost.execreated2896C:\Windows\System32\winrshost.exe	4
C:\Windows\System32\cscript.execreated2904C:\Users\mp\AppData\Local\Temp\natmasla.exe	4
C:\Program Files\Internet Explorer\iexplore.execreated2952C:\Users\mp\AppData\Local\Temp\svchost.exe	4
UNKNOWNcreated2848C:\Program Files\Google\Chrome\Application\chrome.exe	3
C:\Windows\explorer.execreated3012C:\Users\mp\AppData\Local\Temp\{1814DBFD-E07D-47A8-A49D-FC41D9109B7E}\TMP6C00.tmp	3
C:\Program Files\Internet Explorer\iexplore.execreated2844C:\Users\mp\AppData\Local\Temp\svchost.exe	3
C:\Program Files\Internet Explorer\iexplore.execreated2908C:\Users\mp\AppData\Local\Temp\svchost.exe	3
C:\Users\mp\AppData\Local\Temp\svchost.execreated2916C:\Program Files\Microsoft\DesktopLayer.exe	3
C:\Windows\explorer.execreated2956C:\Users\mp\AppData\Local\Temp\{7D8C069F-EB96-4AC1-B171-1EC72BF7FE72}\TMP5712.tmp	3
UNKNOWNcreated2848C:\Windows\System32\cmd.exe	3
C:\Program Files\Internet Explorer\iexplore.execreated3260C:\Users\mp\AppData\Local\Temp\oydgn.dll	3
C:\Users\mp\AppData\Local\Temp\svchost.execreated2884C:\Program Files\Microsoft\DesktopLayer.exe	2
C:\Program Files\Internet Explorer\iexplore.execreated2888C:\Users\mp\AppData\Local\Temp\fhfy.dll	2
C:\Users\mp\AppData\Local\Temp\fhfy.dllcreated2900C:\Users\mp\AppData\Local\Temp\fhfy.dll	2
C:\Program Files\Internet Explorer\iexplore.execreated2868C:\Users\mp\AppData\Local\Temp\svchost.exe	2
C:\Program Files\Internet Explorer\iexplore.execreated2884C:\Users\mp\AppData\Local\Temp\svchost.exe	2
C:\Program Files\Internet Explorer\iexplore.execreated2604C:\Program Files\RealNetworks\RealDownloader\recordingmanager.exe	2

UNKNOWNcreated2828C:\Program Files\Google\Chrome\Application\chrome.exe	2
C:\Windows\explorer.execreated3468C:\Users\mp\AppData\Local\Temp\{BB69D978-68AB-48D3-968F-4BD987A6A643}\TMP7C8A.tmp	2
C:\Windows\explorer.execreated3476C:\Users\mp\AppData\Local\Temp\{D5362E1B-9F0B-488C-AB5E-2C3F203AF525}\TMP7D35.tmp	2
C:\Windows\System32\cmd.execreated2796C:\Windows\System32\wscript.exe	2
C:\Windows\System32\wscript.execreated2884C:\Windows\System32\cmd.exe	2
UNKNOWNcreated2916C:\Users\mp\AppData\Local\Temp\rad959E0.tmp.exe	2
C:\Users\mp\AppData\Local\Temp\svchost.execreated2856C:\Program Files\Microsoft\DesktopLayer.exe	2
C:\Program Files\Microsoft\DesktopLayer.execreated2864C:\Program Files\Google\Chrome\Application\chrome.exe	2
C:\Windows\explorer.execreated3004C:\Users\mp\AppData\Local\Temp\{367FA926-19AA-4A16-850B-84237F5DC1F7}\TMP6B55.tmp	2
C:\Windows\explorer.execreated2960C:\Users\mp\AppData\Local\Temp\{30D773F7-330B-4CF3-BA07-F8B8D9C0B22D}\TMP8260.tmp	2
C:\Windows\explorer.execreated2968C:\Users\mp\AppData\Local\Temp\{273EEDE5-08AC-4794-A8AA-DFCCF672AA7A}\TMP82BB.tmp	2
C:\Program Files\Internet Explorer\iexplore.execreated2800C:\Windows\System32\cmd.exe	2
UNKNOWNcreated3592C:\Windows\System32\schtasks.exe	2
C:\Users\mp\AppData\Local\Temp\svchost.execreated2828C:\Program Files\Microsoft\DesktopLayer.exe	2
C:\Program Files\Microsoft\DesktopLayer.execreated2860C:\Program Files\Google\Chrome\Application\chrome.exe	2
C:\Users\mp\AppData\Local\Temp\svchost.execreated2868C:\Program Files\Microsoft\DesktopLayer.exe	2
UNKNOWNcreated2904C:\Users\mp\AppData\Local\Temp\svchost.exe	2
UNKNOWNcreated2936C:\Program Files\Google\Chrome\Application\chrome.exe	2
C:\Program Files\Internet Explorer\iexplore.execreated2760C:\Program Files\RealNetworks\RealDownloader\recordingmanager.exe	2
C:\Users\mp\AppData\Local\Temp\qhywf.dllcreated2864C:\Users\mp\AppData\Local\Temp\qhywf.dll	2
UNKNOWNcreated2872C:\Windows\System32\cmd.exe	2
UNKNOWNcreated2816C:\Users\mp\AppData\Local\Temp\svchost.exe	2
C:\Program Files\Internet Explorer\iexplore.execreated2816C:\Users\mp\AppData\Local\Temp\svchost.exe	2
C:\Users\mp\AppData\Local\Temp\svchost.execreated2848C:\Program Files\Google\Chrome\Application\chrome.exe	2
UNKNOWNcreated2860C:\Program Files\Google\Chrome\Application\chrome.exe	2
C:\Users\mp\AppData\Local\Temp\8075.tmpcreated2912C:\Windows\explorer.exe	2

C:\Windows\explorer.execreated2952C:\Windows\System32\vssadmin.exe	2
C:\Windows\explorer.execreated2964C:\Users\mp\AppData\Local\Temp\{D5524322-FE26-4D96-B30A-D444A0252187}\TMP57C7.tmp	2
UNKNOWNcreated2836C:\Program Files\Google\Chrome\Application\chrome.exe	2
UNKNOWNcreated2988C:\Users\mp\AppData\Local\Temp\5DCC.tmp	2
C:\Users\mp\AppData\Local\Temp\5DCC.tmpcreated3000C:\Windows\explorer.exe	2
C:\Windows\explorer.execreated3024C:\Windows\System32\vssadmin.exe	2
C:\Windows\System32\cmd.execreated2780C:\Windows\System32\wscript.exe	2
UNKNOWNcreated2840C:\Windows\System32\cmd.exe	2
C:\Program Files\Internet Explorer\iexplore.execreated2940C:\Users\mp\AppData\Local\Temp\svchost.exe	2
UNKNOWNcreated2852C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3	2
C:\Program Files\Internet Explorer\iexplore.execreated2856C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3	2
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3created2864 C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3	2
C:\Windows\explorer.execreated3048C:\Users\mp\AppData\Local\Temp\{B94AACF7-060B-4DD7-A94F-C594EBC1C4A6}\TMP7B23.tmp	2
C:\Windows\explorer.execreated3056C:\Users\mp\AppData\Local\Temp\{FA6541DE-7EAA-4074-9E3C-CBF985A37D2A}\TMP7B7E.tmp	2
UNKNOWNcreated2876C:\Users\mp\AppData\Local\Temp\svchost.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2808C:\Users\mp\AppData\Local\Temp\svchost.exe	1
UNKNOWNcreated2808C:\Users\mp\AppData\Local\Temp\svchost.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2908C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\U5WCBXIX\33[1].mp3	1
UNKNOWNcreated2908C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\U5WCBXIX\33[1].mp3	1
C:\Windows\explorer.execreated2936C:\Users\mp\Documents\AddressBook\AddressBook.pif	1
UNKNOWNcreated2936C:\Users\mp\Documents\AddressBook\AddressBook.pif	1
C:\Program Files\Internet Explorer\iexplore.execreated2804C:\Users\mp\AppData\Local\Temp\svchost.exe	1
UNKNOWNcreated2804C:\Users\mp\AppData\Local\Temp\svchost.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2768C:\Windows\System32\cmd.exe	1

UNKNOWNcreated2768C:\Windows\System32\cmd.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2840C:\Users\mp\AppData\Local\Temp\svchost.exe	1
UNKNOWNcreated2840C:\Users\mp\AppData\Local\Temp\svchost.exe	1
C:\Windows\System32\sdiagnhost.execreated3668C:\Windows\System32\ipconfig.exe	1
C:\Windows\System32\sdiagnhost.execreated3708C:\Windows\System32\ROUTE.EXE	1
UNKNOWNcreated3668C:\Windows\System32\ipconfig.exe	1
UNKNOWNcreated3708C:\Windows\System32\ROUTE.EXE	1
UNKNOWNcreated2844C:\Windows\System32\cscript.exe	1
C:\Windows\System32\cmd.execreated2844C:\Windows\System32\cscript.exe	1
UNKNOWNcreated2828C:\Windows\System32\cscript.exe	1
C:\Windows\System32\cmd.execreated2828C:\Windows\System32\cscript.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2812C:\Users\mp\AppData\Local\Temp\svchost.exe	1
UNKNOWNcreated2812C:\Users\mp\AppData\Local\Temp\svchost.exe	1
UNKNOWNcreated2836C:\Users\mp\AppData\Local\Temp\svchost.exe	1
C:\Users\mp\AppData\Local\Temp\svchost.execreated2872C:\Program Files\Google\Chrome\Application\chrome.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2836C:\Users\mp\AppData\Local\Temp\svchost.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2852C:\Users\mp\AppData\Local\Temp\svchost.exe	1
UNKNOWNcreated2852C:\Users\mp\AppData\Local\Temp\svchost.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2856C:\Users\mp\AppData\Local\Temp\qhywf.dll	1
UNKNOWNcreated2856C:\Users\mp\AppData\Local\Temp\qhywf.dll	1
C:\Program Files\Internet Explorer\iexplore.execreated2788C:\Users\mp\AppData\Local\Temp\svchost.exe	1
UNKNOWNcreated2788C:\Users\mp\AppData\Local\Temp\svchost.exe	1
UNKNOWNcreated2908C:\Users\mp\AppData\Local\Temp\svchost.exe	1
UNKNOWNcreated2916C:\Program Files\Microsoft\DesktopLayer.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2832C:\Users\mp\AppData\Local\Temp\svchost.exe	1
UNKNOWNcreated2832C:\Users\mp\AppData\Local\Temp\svchost.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2892C:\Users\mp\AppData\Local\Temp\8075.tmp	1
C:\Program Files\Internet Explorer\iexplore.exeterminated2892C:\Users\mp\AppData\Local\Temp\8075.tmp	1
UNKNOWNcreated2892C:\Users\mp\AppData\Local\Temp\8075.tmp	1
C:\Windows\System32\services.execreated3104C:\Windows\System32\VSSVC.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2988C:\Users\mp\AppData\Local\Temp\5DCC.tmp	1
C:\Windows\System32\services.execreated3168C:\Windows\System32\VSSVC.exe	1

C:\Windows\System32\svchost.execreated3492C:\Windows\System32\notepad.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2756C:\Windows\System32\cmd.exe	1
UNKNOWNcreated2756C:\Windows\System32\cmd.exe	1
C:\Windows\System32\wscript.execreated2848C:\Windows\System32\cmd.exe	1
UNKNOWNcreated2880C:\Users\mp\AppData\Local\Temp\rad38BFE.tmp.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2840C:\Windows\System32\cmd.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2784C:\Users\mp\AppData\Local\Temp\svchost.exe	1
UNKNOWNcreated2784C:\Users\mp\AppData\Local\Temp\svchost.exe	1
UNKNOWNcreated2796C:\Users\mp\AppData\Local\Temp\svchost.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2796C:\Users\mp\AppData\Local\Temp\svchost.exe	1
C:\Users\mp\AppData\Local\Temp\svchost.execreated2840C:\Program Files\Google\Chrome\Application\chrome.exe	1
C:\Program Files\Internet Explorer\iexplore.exeterminated2784C:\Users\mp\AppData\Local\Temp\svchost.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2764C:\Windows\System32\cmd.exe	1
UNKNOWNcreated2884C:\Windows\System32\cmd.exe	1
C:\Program Files\Internet Explorer\iexplore.execreated2852C:\Users\mp\AppData\ Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3	1
C:\Windows\explorer.execreated2880C:\Users\mp\Documents\IE5BAKEX\IE5BAKEX.exe	1
UNKNOWNcreated2880C:\Users\mp\Documents\IE5BAKEX\IE5BAKEX.exe	1

Table 6C: Malicious .tmp files executed within dataset.

Malicious .tmp files executed	Count
{1814DBFD-E07D-47A8-A49D-FC41D9109B7E}\TMP6C00.tmp	3
{7D8C069F-EB96-4AC1-B171-1EC72BF7FE72}\TMP5712.tmp	3
{BB69D978-68AB-48D3-968F-4BD987A6A643}\TMP7C8A.tmp	2
{D5362E1B-9F0B-488C-AB5E-2C3F203AF525}\TMP7D35.tmp	2
UNKNOWNcreated2916C:\Users\mp\AppData\Local\Temp\rad959E0.tmp.exe	2
{367FA926-19AA-4A16-850B-84237F5DC1F7}\TMP6B55.tmp	2
{30D773F7-330B-4CF3-BA07-F8B8D9C0B22D}\TMP8260.tmp	2
{273EEDE5-08AC-4794-A8AA-DFCCF672AA7A}\TMP82BB.tmp	2
C:\Users\mp\AppData\Local\Temp\8075.tmpcreated2912C:\Windows\explorer.exe	2
{D5524322-FE26-4D96-B30A-D444A0252187}\TMP57C7.tmp	2
UNKNOWNcreated2988C:\Users\mp\AppData\Local\Temp\5DCC.tmp	2
C:\Users\mp\AppData\Local\Temp\5DCC.tmpcreated3000C:\Windows\explorer.exe	2
{B94AACF7-060B-4DD7-A94F-C594EBC1C4A6}\TMP7B23.tmp	2
{FA6541DE-7EAA-4074-9E3C-CBF985A37D2A}\TMP7B7E.tmp	2
C:\Program Files\Internet Explorer\iexplore.execreated2892C:\Users\mp\AppData\Local\Temp\8075.tmp	1
C:\Program Files\Internet Explorer\iexplore.exeterminated2892C:\Users\mp\AppData\Local\Temp\8075.tmp	1
UNKNOWNcreated2892C:\Users\mp\AppData\Local\Temp\8075.tmp	1
C:\Program Files\Internet Explorer\iexplore.execreated2988C:\Users\mp\AppData\Local\Temp\5DCC.tmp	1
UNKNOWNcreated2880C:\Users\mp\AppData\Local\Temp\rad38BFE.tmp.exe	1

Table 6D: Examples of some malicious .exe file writes within the 5,132 dataset.

Malicious file writes: executable (.exe) files	Count
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Temp\svchost.exe-1	62488
C:\Windows\System32\wscript.exeWriteC:\Users\mp\AppData\Local\Temp\rad959E0.tmp.exe-1	4326
C:\Windows\System32\wscript.exeWriteC:\Users\mp\AppData\Local\Temp\rad38BFE.tmp.exe-1	4105
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe-1	1453
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\12321323[1].exe-1	1316
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\PluginFlashPlayer[1].exe-1	741
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\wrar501ru[1].exe-1	675
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\kl[1].exe-1	472
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\putty[1].exe-1	451
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\2[1].exe-1	409
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\cpu[1].exe-1	395
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\nvm[1].exe-1	372
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\ws[1].exe-1	364
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\amd[1].exe-1	339
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\1257844607.encrcyper[1].exe-1	338
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary	336

Internet Files\Content.IE5\YELZ71EO\2050276296.scan0001[1].exe-1	
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\cclub02[1].exe-1	322
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\1769382244.HDPlayer_BETA_installer_v2.55[1].exe-1	318
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\100312839.CryptocurrencyTradingBotV1.4[1].exe-1	311
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\paget[1].exe-1	299
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\2139980916.S4_Crack[1].exe-1	298
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Installation[1].exe-1	296
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\record[1].exe-1	288
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\pdf[1].exe-1	286
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\1149332910.Host2_crypter_05[1].exe-1	280
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\144[1].exe-1	277
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\610411940.save[1].exe-1	275
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\1844534592.avg[1].exe-1	275
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Flash_Movie_Player_Plugin[1].exe-1	273
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\FileZilla_3.7.3_setup[1].exe-1	272
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\TorrentInjectorSmart[1].exe-1	259
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary	252

Internet Files\Content.IE5\YELZ71EO\ptdb_opera[1].exe-1	
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\harddiskdrivers[1].exe-1	250
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\printerdrivers[1].exe-1	250
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Herb[1].exe-1	250
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\oppp[1].exe-1	246
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\164923136.cpu[1].exe-1	245
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\calc[1].exe-1	236
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\ptdb_yandex[1].exe-1	234
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\f[1].exe-1	231
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\535825339.androm[1].exe-1	229
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\aveksynkens[1].exe-1	228
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\ff[1].exe-1	223
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\suba002[1].exe-1	220
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\919691940.p-update[1].exe-1	217
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\wav[1].exe-1	215
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\flashplayer[1].exe-1	215
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary	207

Internet Files\Content.IE5\YELZ71EO\NET+Traffic+Meter[1].exe-1	
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\841642867.johny[1].exe-1	204
C:\Program Files\Internet Explorer\iexplore.exeWriteC:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\to2[1].exe-1	204

Table 6E: Examples of malicious .bat file writes within the 5,132 dataset

Malicious file writes: Batch file (.bat) files	Column2
C:\Users\mp\AppData\Local\Temp\natmasla.exeWriteC:\Users\mp\AppData\Local\Temp\104930.bat-1	2
C:\Users\mp\AppData\Local\Temp\natmasla.exeWriteC:\Users\mp\AppData\Local\Temp\103809.bat-1	2

Table 6F: Examples of malicious .VBS file writes within the 5,132 dataset.

Malicious file writes: Visual Basic Script (.vbs) files	Count
C:\Windows\System32\cmd.exeWriteC:\Users\mp\AppData\Local\Temp\a.vbs-1	5
C:\Windows\System32\cmd.exeWriteC:\Users\mp\AppData\Local\Temp\AWUiJlHe.vbs-1	31
C:\Windows\System32\cmd.exeWriteC:\Users\mp\AppData\Local\Temp\WXAyNAwV.vbs-1	31

Table 6G: Malicious additions to auto-start sections of the registry observed in 5,132 dataset.

Additions in the auto run on login function of the registry.	Count
C:\Windows\System32\wsqmcons.exeSetValueKeyHKLM\SOFTWARE\Microsoft\SQMClient\Windows\WSqmConsLastRunTime-1	201
C:\Windows\System32\rundll32.exeSetValueKeyHKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\DiskDiagnostics\DFDCollectorInvokeTimes-1	158
C:\Windows\explorer.exeSetValueKeyHKCU\Software\Microsoft\Windows\CurrentVersion\Run\39339f0-1	2
C:\Windows\explorer.exeSetValueKeyHKCU\Software\Microsoft\Windows\CurrentVersion\RunOnce*9339f0-1	2
C:\Windows\explorer.exeSetValueKeyHKCU\Software\Microsoft\Windows\CurrentVersion\Run\39339f00-1	2
C:\Windows\explorer.exeSetValueKeyHKCU\Software\Microsoft\Windows\CurrentVersion\RunOnce*9339f00-1	2

Table 6H : Observed Malicious registry behaviour within 5,132 dataset.

Notice the high amount of variance when it's registry behaviours.

Malicious SetValueKey to the registry	Count
C:\Program Files\Google\Chrome\Application\chrome.exeSetValueKeyHKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Userinit-1	14
C:\Users\mp\AppData\Local\Temp\natmasla.exeSetValueKeyHKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet-1	4
C:\Users\mp\AppData\Local\Temp\natmasla.exeSetValueKeyHKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect-1	4
C:\Users\mp\AppData\Local\Temp\natmasla.exeSetValueKeyHKCU\Software\WinRAR\HWID-1	2
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document-1	1
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\DefaultIcon-1	1
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\open\command-1	1
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\print\command-1	1
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\printto\command-1	1
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\.mul-1	1
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\.mul\ShellNew\NullFile-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\DefaultIcon-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\open\command-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\print\command-1	1

C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\printto\command-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\.mul-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\.mul\ShellNew\NullFile-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\DefaultIcon-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\open\command-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\print\command-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\printto\command-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\.mul-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\.mul\ShellNew\NullFile-1	1
C:\Users\mp\Documents\IE5BAKEX\IE5BAKEX.exeSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document-1	1
C:\Users\mp\Documents\IE5BAKEX\IE5BAKEX.exeSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\DefaultIcon-1	1
C:\Users\mp\Documents\IE5BAKEX\IE5BAKEX.exeSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\open\command-1	1
C:\Users\mp\Documents\IE5BAKEX\IE5BAKEX.exeSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\print\command-1	1
C:\Users\mp\Documents\IE5BAKEX\IE5BAKEX.exeSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\printto\command-1	1
C:\Users\mp\Documents\IE5BAKEX\IE5BAKEX.exeSetValueKeyHKLM\SOFTWARE\Classes\.mul-1	1

C:\Users\mp\Documents\IE5BAKEX\IE5BAKEX.exeSetValueKeyHKLM\SOFTWARE\Classes\.mul\ShellNew\NullFile-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\DefaultIcon-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\open\command-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\print\command-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\printto\command-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\.mul-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\.mul\ShellNew\NullFile-1	1

Table 6I: Observed Malicious registry behaviours triggered by malicious Portable Executable files within 5,132 dataset.

Malicious executables adding registry keys	Count
C:\Users\mp\AppData\Local\Temp\natmasla.exeSetValueKeyHKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet-1	4
C:\Users\mp\AppData\Local\Temp\natmasla.exeSetValueKeyHKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect-1	4
C:\Users\mp\AppData\Local\Temp\natmasla.exeSetValueKeyHKCU\Software\WinRAR\HWID-1	2
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document-1	1
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\DefaultIcon-1	1
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\open\command-1	1
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\print\command-1	1
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\printto\command-1	1
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\.mul-1	1
C:\Users\mp\AppData\Local\Temp\fhfy.dllSetValueKeyHKLM\SOFTWARE\Classes\.mul\ShellNew\NullFile-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\DefaultIcon-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\open\command-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\print\command-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\printto\command-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\.mul-1	1
C:\Users\mp\AppData\Local\Temp\qhywf.dllSetValueKeyHKLM\SOFTWARE\Classes\.mul\ShellNew\NullFile-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet	1

Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document-1	
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\DefaultIcon-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\open\command-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\print\command-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\printto\command-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\.mul-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\96[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\.mul\ShellNew\NullFile-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\DefaultIcon-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\open\command-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\print\command-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\SdiMul.Document\shell\printto\command-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\.mul-1	1
C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet	1

Files\Content.IE5\IOJ4C46B\21[1].mp3SetValueKeyHKLM\SOFTWARE\Classes\.mul\ShellNew\NullFile-1	

Table 6J : Record of a malicious server1[1].exe that was submitted to Virus Total showing level and volume of data currently stored for each analysed malicious files.

ed5d47f977e201719cbc8e220a6aa034
2016-07-28 17:00:08
1195268214_04022015_183259.zip
55
45
ALYac
Generic.MSIL.Bladabindi.6CFBEFC9
AVG
BackDoor.MSIL.L
AVware
Trojan.MSIL.Bladabindi.agxy (v)
Ad-Aware
Generic.MSIL.Bladabindi.6CFBEFC9
AegisLab

Troj.W32.Generic!c
AhnLab-V3
Trojan/Win32.Bladabindi.N1339621025
Alibaba
-
Antiy-AVL
Trojan[:HEUR]/Win32.Unknown
Arcabit
Generic.MSIL.Bladabindi.6CFBEFC9
Avast
MSIL:Agent-BXF [Trj]
Avira
BDS/Bladabindi.auje
Baidu
MSIL.Backdoor.Bladabindi.a
BitDefender
Generic.MSIL.Bladabindi.6CFBEFC9
Bkav
W32.GodatyLTAZ.Trojan
CAT-QuickHeal
Backdoor.Bladabindi.AJ6
CMC
-
ClamAV
Win.Trojan.Agent4215604019/CRDF-1
Comodo
TrojWare.MSIL.Bladabindi.KX
Cyren
W32/MSIL_Bladabindi.A2.gen!Eldorado
DrWeb
BackDoor.Bladabindi.1705
ESET-NOD32

MSIL/Bladabindi.F
Emsisoft
Generic.MSIL.Bladabindi.6CFBEFC9 (B)
F-Prot
W32/MSIL_Bladabindi.A2.gen!Eldorado
F-Secure
Generic.MSIL.Bladabindi.6CFBEFC9
Fortinet
MSIL/Agent.PPV!tr
GData
Generic.MSIL.Bladabindi.6CFBEFC9
Ikarus
Trojan.Msil
Jiangmin
Trojan/Generic.bbmuv
K7AntiVirus
Trojan (003ca8581)
K7GW
Trojan (003ca8581)
Kaspersky
HEUR:Trojan.Win32.Generic
Kingsoft
VIRUS_UNKNOWN
Malwarebytes
Backdoor.Bot
McAfee
Trojan-FIGN
McAfee-GW-Edition
BehavesLike.BackdoorNJRat.gc
MicroWorld-eScan
-
Microsoft

Backdoor:MSIL/Bladabindi.AJ
NANO-Antivirus
Trojan.Win32.DownLoader10.dbxzfj
Panda
Generic Malware
Qihoo-360
-
SUPERAntiSpyware
-
Sophos
Troj/MSIL-HX
Symantec
-
Tencent
-
TheHacker
-
TotalDefense

Log file 6K: Drive-by-download behaviour based on the malware shown in Table 6J.

"file","4/2/2015 18:33:4.211","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe","-1"

"file","4/2/2015 18:33:4.211","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe","-1"

"file","4/2/2015 18:33:4.241","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe","-1"

"file","4/2/2015 18:33:4.241","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe","-1"

"file","4/2/2015 18:33:4.602","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe","-1"

"file","4/2/2015 18:33:4.602","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe","-1"

"file","4/2/2015 18:33:4.602","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe","-1"

"file","4/2/2015 18:33:4.602","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe","-1"

"file","4/2/2015 18:33:4.602","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe","-1"

"file","4/2/2015 18:33:4.602","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe","-1"

"file","4/2/2015 18:33:5.573","System","Write","C:\Users\mp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\YELZ71EO\Server[1].exe","-1"

Table 6L: Record of a svchost.exe discussed in chapter 6 files submitted to virus total showing level and volume of data currently stored for each analysed malicious files.

Sample 6M VT Data
ef4b0187602641bc103e8ba3db55b020
29/07/2016 18:05
1565696439_06052014_173030.zip
55
47
ALYac
Backdoor.Zbot.al
AVG
PSW.Generic12.AMWG
AVware
Trojan.Win32.Generic!BT
Ad-Aware
Trojan.Zbot.IPC
AegisLab
Packer.W32.Krap.hm!c
AhnLab-V3
Trojan/Win32.Zbot.N133057644
Alibaba
-
Antiy-AVL
Trojan[Packed]/Win32.Krap
Arcabit
Trojan.Zbot.IPC
Avast

Win32:GenMalicious-GOW [Trj]
Avira
TR/Crypt.XPACK.Gen
Baidu
Win32.Trojan.Ramnit.e
BitDefender
Trojan.Zbot.IPC
Bkav
W32.RammintDropperNNA.Worm
CAT-QuickHeal
Trojan.Krap.rw3
CMC
-
ClamAV
Win.Malware.QBot-846
Comodo
MalCrypt.Indus!
Cyren
W32/Ramnit.UNAX-1410
DrWeb
VBS.Dropper.128
ESET-NOD32
Win32/Ramnit.A
Emsisoft
Trojan.Zbot.IPC (B)
F-Prot
W32/Ramnit.X
F-Secure
Trojan.Zbot.IPC

Fortinet
W32/Snocry.JQ!tr
GData
Trojan.Zbot.IPC
Ikarus
Packer.Win32.Krap
Jiangmin
Trojan/Generic.beznk
K7AntiVirus
Backdoor (04c4e9741)
K7GW
Backdoor (04c4e9741)
Kaspersky
Packed.Win32.Krap.hm
Kingsoft
-
Malwarebytes
Trojan.Zbot
McAfee
PWS-Zbot.gen.pq
McAfee-GW-Edition
BehavesLike.PWSZbot.mc
MicroWorld-eScan
Trojan.Zbot.IPC
Microsoft
Worm:Win32/Ramnit.A
NANO-Antivirus
Trojan.Win32.ULPM.dlsptx
Panda

Trj/Krap.Y
Qihoo-360
-
SUPERAntiSpyware
-
Sophos
W32/Ramnit-ET
Symantec
-
Tencent
Trojan.Win32.Ramnit.efg
TheHacker
-
TotalDefense

Sample 6M of part Log file 6L drive-by-download filtered log file for the sample in Table 6L.

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"process","6/5/2014 18:30:45.687","UNKNOWN","created","2840","C:\Users\mp\AppData\Local\Temp\svchost.exe"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\IndexTable\FileIdIndex-{41003fa5-89a3-11e3-bfdc-806e6f6e6963}_IndexName_","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore_CurrentObjectId_","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\ObjectTable\13D_ObjectId_","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\LruList\CurrentLru","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\LruList\0000000000000B73\ObjectId","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\LruList\0000000000000B73\ObjectLru","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\ObjectTable\13D_ObjectLru_","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\ObjectTable\13D_FileId_","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\ObjectTable\13D_Usn_","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\ObjectTable\13D_UsnJournalId_","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\ObjectTable\13D\Indexes\FileIdIndex-{41003fa5-89a3-11e3-bfdc-806e6f6e6963}\12000000009E8D","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\IndexTable\FileIdIndex-{41003fa5-89a3-11e3-bfdc-806e6f6e6963}\12000000009E8D\13D","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\ObjectTable\13D\AeFileID","-1"

"registry","6/5/2014 18:30:44.956","C:\Windows\System32\svchost.exe","SetValueKey","\REGISTRY\A\{05B8D4EC-D544-11E3-8D24-080027E9ED53}\DefaultObjectStore\ObjectTable\13D\AeProgramID","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

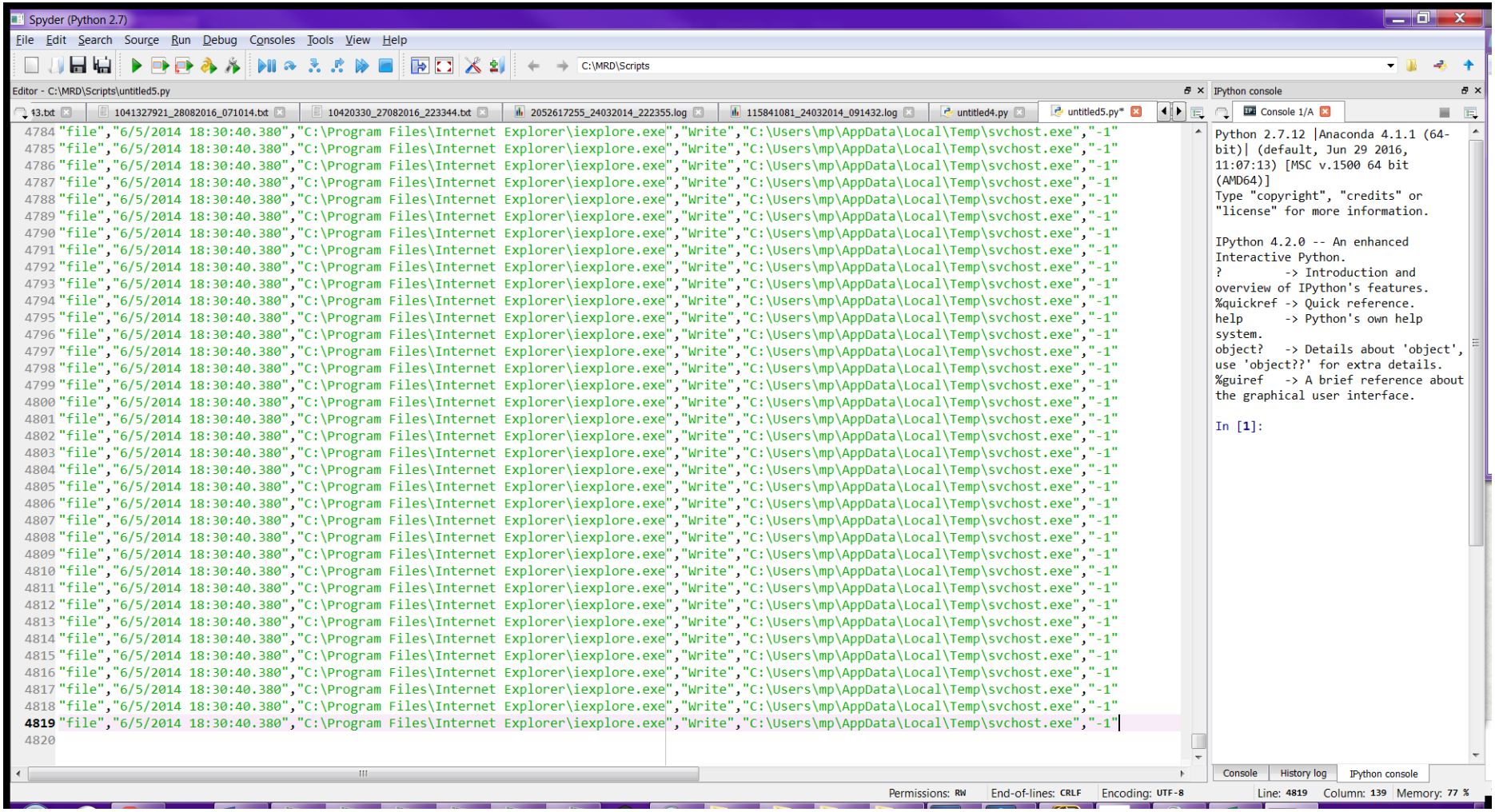
"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

"process","6/5/2014 18:30:49.613","UNKNOWN","created","2868","C:\Program Files\Google\Chrome\Application\chrome.exe"

"registry","6/5/2014 18:30:47.39","C:\Program Files\Google\Chrome\Application\chrome.exe","SetValueKey","HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Userinit","-1"

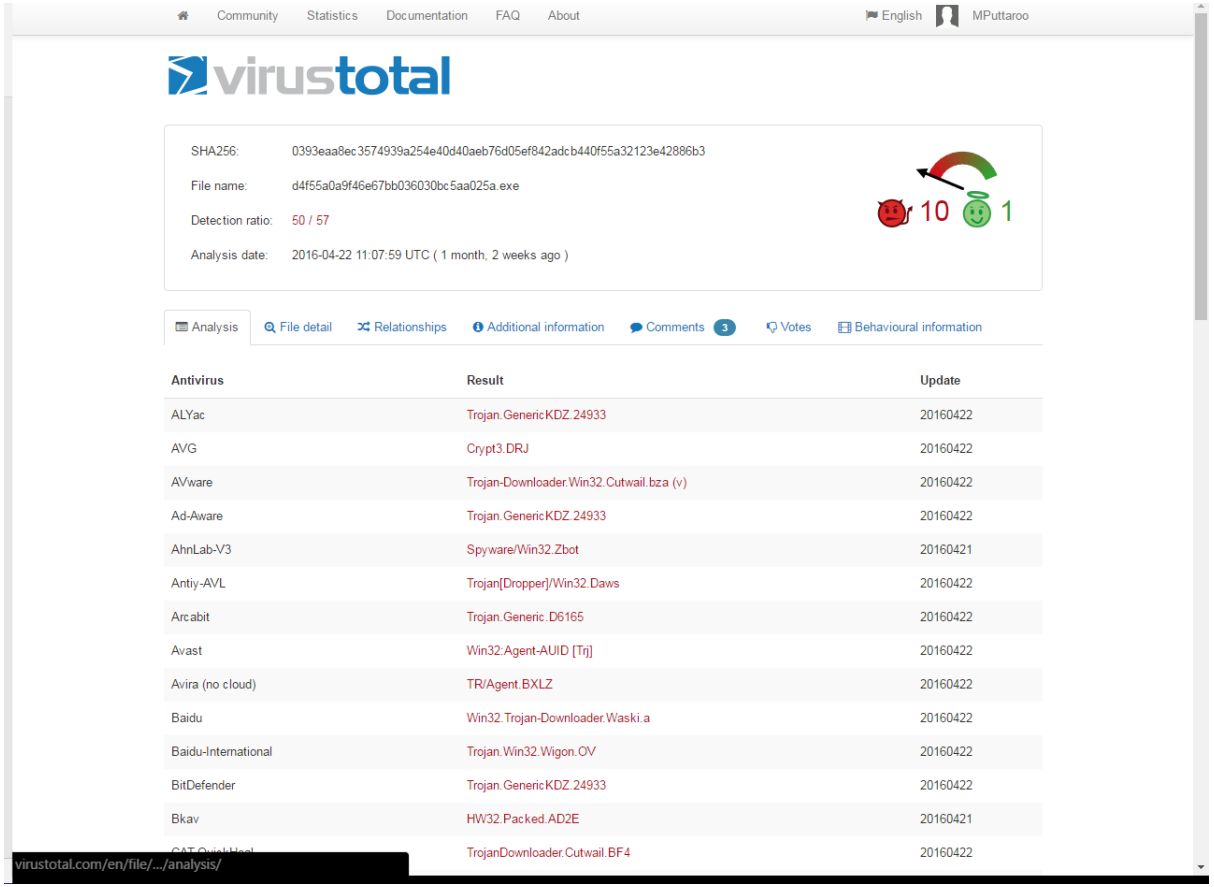
"file","6/5/2014 18:30:40.220","C:\Program Files\Internet Explorer\iexplore.exe","Write","C:\Users\mp\AppData\Local\Temp\svchost.exe","-1"

Screenshot 6N: Drive-by-download behavioural log file size.



The full log file had 4819 behavioural interactions after a simple 180 seconds opening a malicious domain in internet explorer.

Screenshot 6: of a drive-by-download captured by capture bat submitted to Virus Total.



UNIVERSITY OF WEST LONDON

Submission for consideration of Doctor of Philosophy

Candidate's Declaration Form

This form must be submitted with the thesis.

Name of Candidate:
MOHAMMAD ALLY REHAZ PUTTAROO

Title of Thesis:

A BEHAVIOURAL STUDY IN RUN-TIME ANALYSIS ENVIRONMENTS AND DRIVE-BY-DOWNLOAD ATTACKS

1 Concurrent registration for two or more academic awards
(* *please delete the statement which does not apply)*

either

*** I declare that while registered as a candidate for University of West London's research degree, I have not been a registered candidate or enrolled student for another award of another academic or professional institution.**

or

*** I declare that while registered for University of West London's research degree, I was, with UWL's specific permission, a *registered candidate / * enrolled student for the following award:**

2 Material submitted for another award
(* *please delete the statement which does not apply)*

either

*** I declare that no material contained in the thesis has been used in any other submission for an academic award.**

or

*** I declare that the following material contained in the thesis formed part of a submission for the award of (*Please state the award and awarding body and list the material below)***

Signature of candidate M.Puttaroo..... Date 17/07/2017.....