Scalability of information centric networking using mediated topology management

Alzahrani, Bander, Reed, Martin, Riihijärvi, Janne and Vassilakis, Vassilios (2015) Scalability of information centric networking using mediated topology management. Journal of Network and Computer Applications, 50. pp. 126-133. ISSN 1084-8045

http://dx.doi.org/10.1016/j.jnca.2014.07.002

Scalability of Information Centric Networking Using Mediated Topology Management

Bander A. Alzahrani[a], Martin J. Reed[a], Janne Riihiärvi[b], Vassilios G. Vassilakis[c]

[a]*School of CSEE, University of Essex, Colchester, UK*
[b]*Institute for Networked Systems, RWTH, Aachen University, Aachen, Germany*
[c]*Centre for Communication Systems Research (CCSR), Dept. of Electronic Engineering, University of Surrey, Guildford, UK*

Abstract

Information centric networking is a new concept that places emphasis on the information items themselves rather than on where the information items are stored. Consequently, routing decisions can be made based on the information items rather than on simply destination addresses. There are a number of models proposed for information centric networking and it is important that these models are investigated for their scalability if we are to move from early prototypes towards proposing that these models are used for networks operating at the scale of the current Internet. This paper investigates the scalability of an ICN system that uses mediation between information providers and information consumers using a publish/subscribe delivery mechanism. The scalability is investigated by extrapolating current IP traffic models for a typical national-scale network provider in the UK to estimate mediation workload. The investigation demonstrates that the mediation workload for route determination is on a scale that is comparable to, or less than, that of current IP routing while using a forwarding mechanism with considerably smaller tables than current IP routing tables. Additionally, the work shows that this can be achieved using a security mechanism that mitigates against maliciously injected packets thus stopping attacks such as denial of service that is common with the current IP infrastructure.

Keywords: Information Centric Networking, Bloom Filter, Security, Topology Management

1. Introduction

Networking has traditionally focussed on a node centric model: a user connects to a particular server through an identifier such as a domain name and obtains, or sends, a specific piece of information. This model has served well and underpins highly successful networks such as the Internet. However, coping with scale has brought about changes to this model through technologies such as content delivery networks (CDN) and HTTP redirection (HTTP-R) (Spagna et al., 2013). Both of these technologies effectively break the node centric model, although users are essentially unaware that a single domain name does not mean a single physical server with one address. Recently researchers have questioned whether creating elaborate technologies to bolster up a node centric model is sensible and have turned instead to alternate architectures which label information items in a manner that focusses less on *where* they are stored and more on *what* they are, and how they relate to other information items (Trossen et al., 2010). There is not yet a single architecture which can be said to definitively provide this alternative paradigm and there are a number of alternative architectures said to fall under terms such as information centric networking (ICN) (Trossen and Parisis, 2012), content centric networking (CCN) or named data networking (NDN) (Jacobson et al., 2009). This paper will concentrate on one of these architectures, namely the publish subscribe Internet routing paradigm (PSIRP), later furthered by the PURSUIT project (Fotiou et al., 2012), and will use the term ICN for this new paradigm. One question that is important to ask with the proposal for a new architecture is why it should be introduced? Certainly the current Internet architecture is highly successful and has, despite occasional "doom mongering" (Handley, 2006), managed to keep up with strong demands for growth. Consequently, we do not propose ICN because the Internet is "broken," but rather look to an alternative so that it can provide a basis for new services and future growth with lower transport costs.

There are a wide variety of ICN solutions (Ahlgren et al., 2012), however, we can generalize the architectures as systems that label information items and provide a network architecture where: providers of the information items can advertise the information items; consumers of information items can request the items; and, network nodes can forward information based upon the matching of provider, consumer and information label. An important issue with regard to ICN architectures is the methodology for forwarding the information. One strategy, taken by techniques such as NDN, is to forward information based on the content labels directly (Jacobson et al., 2009). This allows decisions on what to do with the data to be made based on the content labels, for example items may be opportunistically cached if it is known they may be useful for future users thus avoiding unnecessary transport from the original, more distant, provider we will call this *content based forwarding*. An alternative strategy is to forward using more conventional means, including IP or label swapping, and use a mediation system to match the provider, consumer and information to a suitable destination address or a path at the latest possible moment, so called *late binding*; we call this strategy *mediation assisted forwarding*.

Content based forwarding is attractive as it allows feature rich forwarding/caching decisions. However, if we scale this technique to the size of the Internet this potentially means that a core Internet forwarding node either has to have a forwarding table the size of all the information item labels in the Internet or at least the ability to obtain forwarding information for any of the arbitrary items in good time *i.e.* at a speed compatible with increasing packet forwarding speeds. The scalability of this approach has not been suitably investigated and we leave it to the reader to draw conclusions or further the investigation of the field.

Alternatively, mediation assisted forwarding can use an efficient forwarding mechanism allowing use of high-speed network switching using either existing technology, or as used in the work of this paper, alternative switching technology. However, mediation assisted forwarding requires some centralized, or semi-centralized, network resource that has knowledge of providers and consumers and can quickly plan a suitable resource. At first sight this may seem challenging, however, in the current Internet model there is a close equivalent in the form of the domain name system (DNS); in most Internet based communication there is a requirement for an application to contact this network resource to find the forwarding address and return it to the consumer (and often the provider in the form of a reverse name lookup). The current Internet is evidence that such a "centralized" network resource is possible, however, in ICN the granularity of requests is not at the scale of domain names, but at the level of individual information items. Thus, the question addressed in this paper is: can mediated assisted forwarding be achieved at a scale equivalent to that provided by network operators that support the current Internet? We affirm that the answer to this question is positive and explore the parameters of the problem and suggest some solutions. The work is focussed on solving this question for a single domain comparable in size to a current autonomous system.

This paper presents an overview of an ICN architecture in Section 2 and expands the description of the topology management function of the architecture in Section 3. Traffic models are described in Section 4 that are used to drive estimates of the scalability that are presented and discussed in Section 5.

2. The PURSUIT ICN architecture

The PSIRP project (and later PURSUIT) (Fotiou et al., 2012) brings together two concepts to form an ICN solution: *mediated assisted forwarding* and *publish/subscribe* communication. A publisher notifies the mediation system that it can provide an item, a subscriber notifies the mediation system that it wishes to obtain the information item. These two events need not be in the traditional client/server order: it is possible that the subscriber requests something before a publisher has made it available. The mediation system is broken down into two functions: *Rendezvous* and *topology management.* These two functions control the third function: *forwarding.* It is assumed in this paper that networks are owned and managed by *autonomous systems* in the same manner as current network infrastructures. Consequently, the three functions are required in each autonomous system and here we consider how they operate within a single autonomous system. Connections between autonomous systems might be expected to be maintained through mechanisms such as that used in the current Internet through BGPv4 routing.

2.1. Mediation between publishers and subscribers: Rendezvous

The subject of this paper is mainly regarding the topology management function, but to further understand the mediation mechanism it is important to have a high-level understanding of the Rendezvous function. Each information item is given a statistically unique Rendezvous ID (RID). The Rendezvous is a distributed system that maintains an information graph that links each RID in a hierarchical structure. Each RID is said to belong to a parent *scope* and a scope may itself be the child of another scope. In this manner it is possible to relate information items through ontologies that can be flexibly defined (Tagger et al., 2013); for example a video content provider could publish videos as each having a unique RID under different categories (*e.g.* action, drama *etc.*) each defined by a different scope. The information

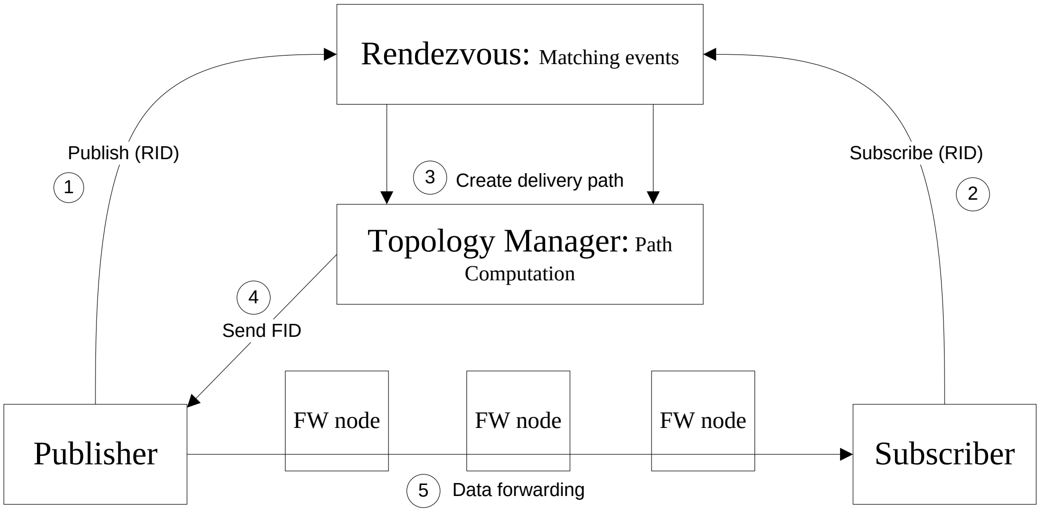

Figure 1: The PURSUIT architectural model showing the mediation between publisher and subscriber through the Rendezvous and topology manager.

graph allows an RID (or scope) to have more than one parent scope such that, following on from the video example, a film that is both science fiction and an action movie could have both categories as parents. The Rendezvous is also responsible for matching publishers to subscribers as shown in Figure 1. The Rendezvous has been studied in some depth (Trossen and Parisis, 2012) using approaches that are either based on distributed hash tables (Katsaros et al., 2012) or hierarchical naming in systems such as *data oriented (and beyond) network architecture* (DONA) (Koponen et al., 2007; Vasilakos et al., 2012). It would be expected that a Rendezvous function would be maintained by each autonomous system. Once the Rendezvous has received both a subscription and publication event for a particular item (an RID) it then passes over to the topology management function to perform the routing required for the forwarding function. The Rendezvous mechanism has been shown to be scalable to Internet scale solutions by Katsaros *et al.* (Katsaros et al., 2012; Vasilakos et al., 2012).

2.2. Topology management - an overview

Topology management is responsible for maintaining intra-domain knowledge of an autonomous system and to determine which publisher will be used for providing a particular data item. When the Rendezvous requests that the topology management function calculates a route, a topology manager uses the topological data to construct a forwarding identifier (FID) that is used by the forwarding function. A more detailed description of topology management is given in Section 3, here we present an overview of the function of topology management. It should be stated here that the topology management is a central network function, but this does not necessarily mean that the function is carried out by a single *topology manager* instance. In practice the number of topology manager instances may range from a single instance to one running in every node in the network. One of the aims of this paper is to determine how many instances may be needed and this is determined for a typical intra-domain scenario in Section 5.

The core functionality of topology management consists of two basic tasks: maintaining an up-to-date representation of the network topology; and, the construction of delivery trees for forwarding. Maintaining an up-to-date representation of the network topology, for example in a form of an annotated graph, enables finding shortest paths and minimal multicast tree construction. The details of this function vary somewhat between network technologies, but in general resemble closely the neighbour discovery mechanisms of classical Internet routing protocols such as OSPF. In an ICN implementation the topology manager instances running at individual nodes would subscribe to Hello-messages and other signalling information in a dedicated topology management scope, and periodically publish their identities as Hello-messages in this scope. Information from these messages can then be combined by the different nodes and published again to all interested nodes, enabling the entire network topology to be reconstructed. This paper concentrates on intra-domain topology management however inter-domain routing is required as well. We assume that a mechanism such as BGP will be used to maintain reachability information for remote autonomous systems as in the current Internet.

The second core functionality of the topology management, the construction of delivery trees connecting publishers and subscribers, is based on the topology data gathered from the network, and the information on the identities of the publishers and subscribers obtained from the Rendezvous. Typically this entails the construction of a shortest path spanning tree connecting the publisher and subscribers, and informing the forwarding function of the existence of this new delivery tree. In the following section we shall discuss in more detail how the forwarding function can be implemented, and how the delivery trees can be represented in a compact fashion.

In the case of best-effort delivery – where paths between a publisher and subscriber(s) can be made arbitrarily and no path state is required – a topology manager instance can determine a suitable FID based only on topological information. Consequently, any suitable topology manager instance can be used for the FID calculation and the topology management function can be spread across any suitable number of instances as is required to meet the demand.

2.3. Forwarding

The PURSUIT ICN model allows a number of forwarding mechanisms to be used. Indeed it is possible that IP could be used as the forwarding mechanism, in this case the topology manager sends a destination IP address (or multicast IP address) as the FID. However, PURSUIT has developed a default forwarding model based on an approach called Line Speed Publish/Subscribe Inter-networking (LISPIN) (Jokela et al., 2009), a multicast forwarding fabric based on Bloom filters (Bloom, 1970). This has the advantage of providing *stateless* multicast forwarding. LIPSIN operates using source routing that encodes the delivery path (or tree in the case of multicast) using link identifiers (LIDs) that are assigned to each link. By using Bloom filters it is possible to encode a number of m-length LIDs into a single m-length Bloom filter.

The operation of LIPSIN is shown in Figure 2. Each unidirectional link between two nodes is identified with a LID. This identifier is m-bits long with k-bits set to 1, where k is the number of hash functions used to generate bit-positions set to 1. The topology manager encodes all the delivery tree elements that compose this path into the Bloom filter by simply forming the logical OR of the individual LIDs constituting the tree. We identify the FID constructed by the topology manager with this Bloom filter in order to be compatible with the LIPSIN terminology. At the end of this process, the topology manager sends the created FID to the publisher in order to be placed in the packet header for forwarding.

Using the FID that is included in the packet header, a forwarding node can decide where packets should be forwarded by performing a bitwise AND between the FID and the LIDs of the outgoing adjacent links. If the result of the set membership function is true, then the forwarder will forward the packet through the link assuming that this link is part of the delivery tree. It should be noted that, while the Bloom filter does not have false negatives, it may have false positives; this is where an FID matches LIDs that were not intentionally included. In the extreme case an all “1s” FID will match

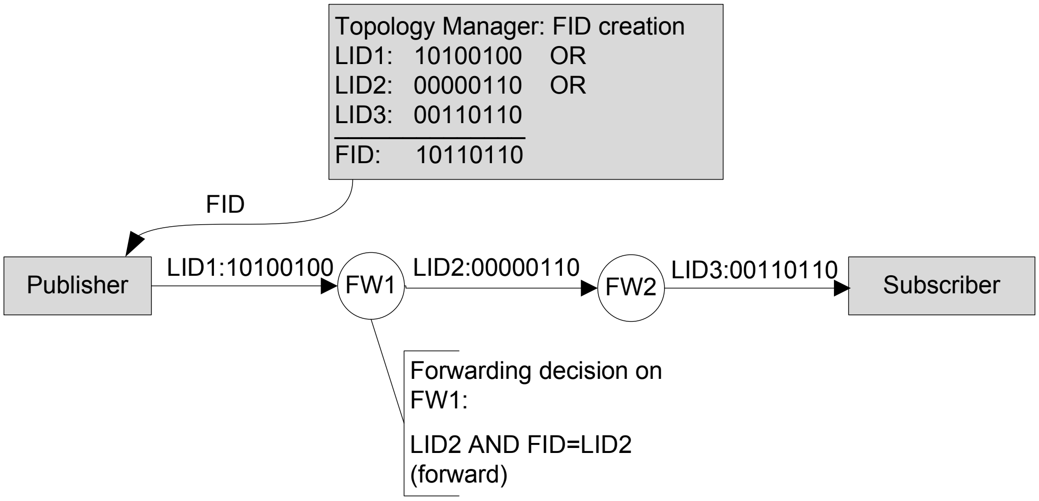

Figure 2: LIPSIN based forwarding using Bloom filters: one of the possible forwarding models in the PURSUIT architecture.

every LID. In practice the parameters of the Bloom filter need to be chosen carefully in order to reduce the false positives, this is discussed further by (Carrea et al., 2014).

2.4. Security and scalability in ICN

Although the LIPSIN forwarding mechanism offers highly efficient forwarding, it has several security issues that can be exploited to attack the network (Rothenberg et al., 2009). An attacker can inject arbitrary traffic to the network by using a previous valid FID that is created for another traffic request. This attack is referred as a FID replay attack. Another threat could be to inject traffic by using brute-force attacks. In this attack, a malicious node tries all possible FIDs to cause some false positive over the links. Furthermore, a computational attack is still possible by collecting and analyzing many valid FIDs in order to infer some parts of the network topology and then to build a valid FID without the topology manager providing it.

There are also fundamental scalability challenges in the topology management and forwarding mechanisms as described above. Since Bloom filters have an inherent false positive rate that depends on the number of links stored in the filter, the overhead of LIPSIN increases as the multicast trees become denser. As discussed in (Jokela et al., 2009) this problem can be mitigated by including some state back into the network, and including information on this state in the FID through the use of *virtual* link IDs. A more fundamental scalability challenge is related to the topology management, as the churn of publishers and subscribers creates a need to rapidly update the delivery trees and continuously recompute their respective FIDs. One of our key objectives in this paper is to demonstrate that this scalability challenge can be overcome by sufficient computational resources at a scale similar to the resources already available in present-day operator networks.

3. Topology management design parameters

Here we consider the design of the topology management function from both the perspective of performing its core function – routing – and the requirement that this routing function cannot be subverted by an attacker to send a significant quantity of malicious traffic.

3.1. Routing functionality in the topology management

Recall that the topology management is aware of the network topology, and mainly responsible for finding the best path between publishers and subscribers. It creates a FID that is used to route information object through the determined path. When the topology manager is instructed by the Rendezvous to find a suitable path and to create FID, the following two strategies may be used: running Dijkstra's algorithm for each request, and, caching all the network topology paths. In the first case each time a topology manager receives a request from Rendezvous to create a delivery tree Dijkstra's algorithm is executed to compute the shortest path considering the present network state. The average processing time of the algorithm increases as the network size grows which may introduce some delay. In the case of caching all the network topology paths, in the setup phase, the topology manager computes paths between each node pair in the network and caches them into a fast memory. Once the topology manager gets a request to find a path, the pre-calculated route is used and then the path LIDs are encoded to create the FID. In the case of multicast requests, where there is one publisher and several subscribers, the topology manager will combine all paths into one path by ORing all relevant LIDs. In the case where highly optimized multicast trees are required the topology manager could use an improved tree routing algorithm, but a significantly higher computation cost.

3.2. Securing the PURSUIT forwarding

The threats described in Section 2.4 are made possible because of the use of static forwarding LIDs which allow reuse and inference of FIDs. To secure the LIPSIN forwarding mechanism against these threats we make use of, and extend, a technique proposed by Rothenberg *et al.*, called zFormation technique (Rothenberg et al., 2009), which is summarized in Figure 3. The zFormation changes the LIDs within a certain timeframe which leads to dynamic and expiring FIDs. To distinguish between the original LIDs, described previously, we here refer to the new identifier as a link ID tag (LIT). The zFormation technique uses a cryptographic function Z that accepts 4 parameters as input and outputs a new LIT. The input parameters to Z are: (1) an in-packet flow ID i (e.g. the RID); (2) a periodically changing time-based secret key $K_i(t)$; (3) the incoming and outgoing port numbers I and O; and, (4) the optimization index d of the LIT. The index d was proposed by Jokela *et al.* (Jokela et al., 2009) to reduce the false positive rate inherited with the use of the Bloom filter. In this proposal d different LITs are assigned to each unidirectional link which subsequently allows the creation of d different candidate FIDs by the topology manager. Then the FIDs are evaluated and the one with lowest false positive rate is selected and its index d is placed in the packet header. To create a dynamic FID, a topology manager applies the function LIT= $Z(i, K_i(t), I, O, d)$ to create all the LITs of the delivery tree and then constructs the FID by simply ORing all the computed LITs as for the standard Bloom filter. The function Z can be implemented using a stream cipher that uses $K_i(t)$ as a time-bound key (Rothenberg et al., 2009).

For the topology manager to create a FID, it is required to share the time-bound shared session key $K_i(t)$. In every Δt, which is the time that a LIT is valid, synchronized shared session keys are generated. As a consequence, the result of computing an LIT is different. After creating the FID in the topology manager, the FID, flow ID i and the index d are sent to the publisher which then uses this as the packet header for forwarding data. Upon receiving this packet at a forwarding node, the node extracts the FID, flow ID i and d then it computes the dynamic LIT of its outgoing interface using the function Z. Then it tests the LIT with the FID for the packet to be forwarded. However, as the LITs are changed every Δt, the in-sessions FIDs need to be updated according to the new values of LITs. Therefore, for long-lived flows, that last longer than Δt, the sessions need a new FID in order to extend the communication before the old FID expires. In (Alzahrani et al., 2012), we

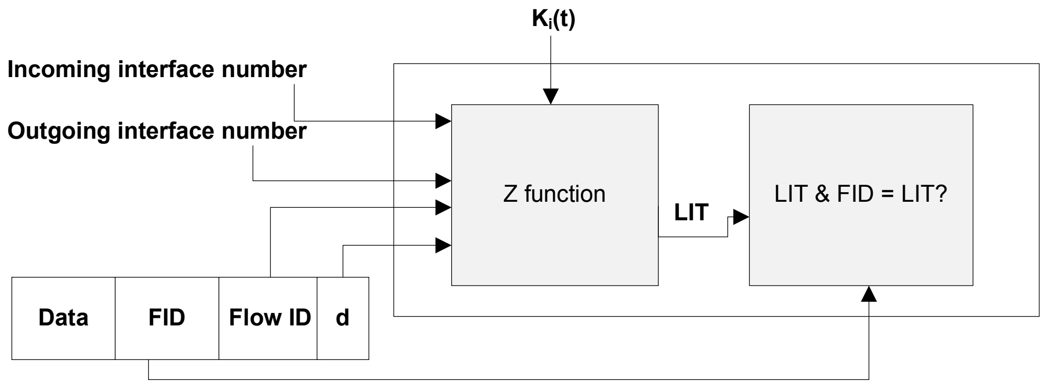

Figure 3: The creation of a zformation for mitigating against malicious traffic injection in LIPSIN forwarding.

have proposed a solution that provides this updated FID by adding a new entity called a FID updater that maintains all the necessary information for creating an updated FID. This method relies upon the Rendezvous to extend the FID as it has knowledge of active publishers and subscribers for each information item. The Rendezvous achieves this by sending an extension request to the FID updater. Furthermore, in (Alzahrani et al., 2013), we demonstrated that the zFormation forwarding mechanism is still vulnerable to successful brute-force attacks if either the fill factor of the FID is too large or Δt is too long. The fill factor specifies the most 1s that can be set in a FID, as for example if a publisher was to set all the FID to 1s (highest fill factor) the packet would match every possible LIT and thus would be sent to every end node. This could be used by an attacker to launch a denial of service attack which is highly undesirable and even more dangerous if one considers that with one packet it would be duplicated to all end-nodes. This can be virtually eliminated by only allowing a maximum fill-factor such that it is not possible to send a packet to every node and it is improbable to send a packet to an arbitrary end-node by guessing a FID. The probability p of guessing a correct FID for a Bloom filter constructed with k-hash functions for each LID, maximum fill-factor ρ and with a path length of h is given by (Rothenberg et al., 2009; Alzahrani et al., 2013):

$$p = \rho^{kh} \tag{1}$$

In practice we find that a maximum fill factor of 0.41 with $k = 5$ hash functions gives a sufficiently low probability of sending a randomly generated

FID of $p = 2.09 \times 10^{-10}$. If a sender could transmit 10^6 packets per second this means that if we set $\Delta t = 40$ minutes there is less than a 50% chance that a packet would reach an arbitrary user within this time window. This is considerably better than is possible with the current IP architecture which allows arbitrary users to send packets to end nodes to cause denial of service. For the remainder of this paper we will assume $\Delta t = 40$ minutes, however, the work holds generally true for other values with suitable adjustment. This value is chosen as it gives reasonable security against packet injection with relatively relaxed timing constraints as will be explored further in Section 5.

In this section we propose an alternative method of updating FIDs compared to that introduced in (Rothenberg et al., 2009). For the mechanism of updating FIDs, consider that LITs start to be valid at t and that they are valid for Δt. Define the time of updates to LIDs as occurring in a time window of $t + u_{min}$ and $t + u_{max}$, where $t + u_{min}$ is the time where the FID update period begins whereas $t + u_{max}$ is the time where the period ends. We require $u_{max} < \Delta t$ to allow orderly updating without risking that LITs will not be updated in time. The valid LITs during period $t, \ldots, t + \Delta t$ will be overlapped with a new LIT being created at $t + \Delta c$ where $\Delta c = \Delta t/2$, thus two sets of LIDs will be valid at any given time.

The Rendezvous keeps states of all events that have a match between publisher and subscriber. Therefore we give the responsibility of extending FIDs to the Rendezvous. If a subscriber is no longer interested in any subscribed item, then the subscriber unsubscribes by sending a request to the Rendezvous to be deleted from the list, otherwise the Rendezvous assumes that the subscriber is still walling to receive this item. At time $t + u_{min}$ the Rendezvous checks for matching event entries for all events that exist from the previous update slot and have not been updated before. Then it sends them gradually to the topology management function until time $t + u_{max}$. This time window is designed to reduce the workload on updating in any given time slot Δt. Here the Rendezvous needs to have an approximately synchronized clock with the topology managers, with a tolerance such that there can be no risk that FID updates are not completed before the LIDs become invalid. This update mechanism is shown in Figure 4 where $\Delta t = 40$ min and three different information item lifetimes are considered. Information item 1 has a lifetime less than the FID and thus does not need to have an updated FID. Information item 3 has a lifetime greater than the FID and thus will require updated FIDs. Additionally Item 3 becomes published after the first LID change and thus uses the first updated FID.

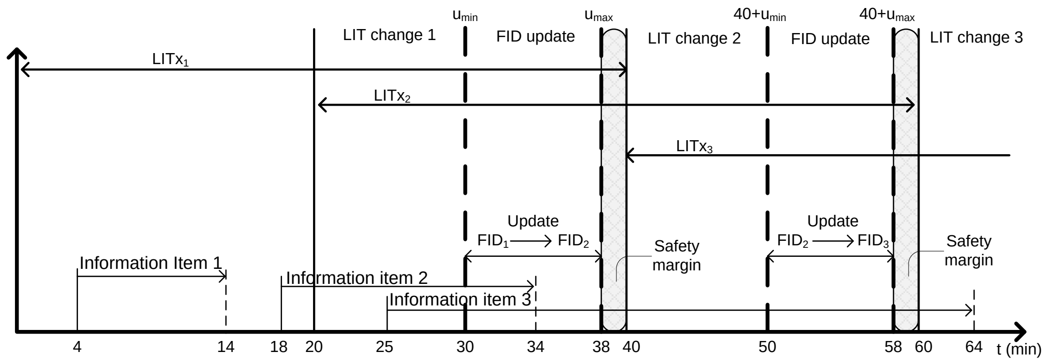

Figure 4: The proposed method for updating the FID to mitigate against malicious traffic injection. The figure shows how three information items, with different dissemination durations, need different forwarding identifier updates due to the periodic changes in link identifiers.

In practice the topology management function cannot know the lifetime of an information item, as with Information item 2 in the example. At time $\Delta C = \Delta t/2 = 20$ min. in the example Information item 2 is still required to be published and thus the topology management prepares to update the new FID. It does not make sense to immediately update the FID for all such information items as it might be the case that it finishes before it is actually required and updating all FIDs immediately requires a high-data rate. Consequently, the FIDs are updated over a suitable window. In this case we propose updating over a window of 8 min. starting 10 min. after the FID change and completing 2 min. before the end of the valid period of the FID to allow for a loose tolerance on the synchronization of the clocks in the topology managers and the forwarding nodes. These values are not highly sensitive and are design parameters to be selected by the network designer depending upon timing tolerances. The values given here are indicative of possible choices.

4. Traffic modelling

An important aspect with regard to mediation models used in ICN is the scalability of the solution. We know from the existing IP DNS model that it is possible to create and maintain a global database used for the so called "slow-path", however, with the ICN approaches this generally requires much finer granularity hence increased resource requirements. Here we are concerned with the scalability of topology management, others have considered the scalability of matching function such as performed by the Rendezvous in the PURSUIT model. There are two key aspects for calculating the scalability of the topology management solution: the first is the number of topology management FID requests which is related to the number of new publish/subscribe events; the second is the number of publish/subscribe matches that exist beyond the time limit for FID expiry as these will need to have FIDs that are refreshed. Thus, we need to consider two aspects of the traffic model: how many new requests are made each second and how long each item may need to remain available. Ideally, this should be estimated from an actual ICN scenario using ICN events, but as this ICN is far from deployment there are no existing statistical models of ICN traffic. Consequently, we estimate ICN traffic from existing IP data as, while the ICN architecture is quite different from IP, we may expect that at least the existing services will be required from an ICN architecture.

To estimate the required resources we investigate the future growth of Internet traffic and estimate the expected number of flow requests per second. While we do not expect an ICN to follow a traditional flow based delivery model we find in current TCP/IP, it is a reasonable assumption that there will be a similar number of user instigated events that can be represented by a TCP/IP session start or by an ICN publisher/subscriber/information item matching event. Consequently the methodology used in the paper is to base ICN event workload on Internet traffic scaled to a near future demand. Here we use a model based on a carrier in a country the size and demographics of the UK with traffic models from known voice, video, HTTP and peer-to-peer application statistics. Specifically the model is based upon the UK population with 52 million Internet users and the carrier serving 30% of this population. These types of traffic have been considered as they form the vast majority of today's Internet traffic. Thus, although other applications are available in the Internet, restricting the model to these application types is not likely to significantly affect the outcome of this analysis. In particular the HTTP traffic encompasses a wider range of applications than just simply "web browsing."

The specific goal of this analysis is to estimate the number of FIDs required per second so that we can estimate the topology management workload and computation requirements needed to handle all FID requests. Additionally, we analyze the flow duration with regard to the requirement to update FIDs for flows that have duration that span a FID update every Δt. The methodology extrapolates from a flow-based model of Internet traffic to independently analyze the four types of traffic. Then we compose all these types of traffic that form large proportion of today's Internet traffic as a total number of requests. The explanation of how the estimates were obtained is given below.

4.1. Voice model

Assumption: For the voice traffic we assume the model of (Kaufman, 1981). This model assumes a transmission link of fixed bandwidth capacity C. Each voice call has a bandwidth requirement b. The capacity C and the requirement b are measured in bandwidth units (b.u.). For example 1 b.u. could be 24 kb/s if we assume a G.729 coding for the voice channel. The arrival rate of the voice calls is denoted by λ and follows a Poisson distribution. The call service rate is exponentially distributed and denoted by μ. The system state $j(j = 0, \ldots, C)$ is defined as the total number of occupied b.u. in the system.

Below we calculate the voice traffic utilization, U, with the aim of finding the inter-arrival time that gives the number of required FIDs per second. In (Kaufman, 1981) the author derives the following recurrent formula for the calculation of the state probabilities:

$$q(j) = b \times 1/j \times \lambda/\mu \times q(j-b) \tag{2}$$

with $q(x) = 0$ for $x < 0$ and $\sum_{j=1}^{C} q(j) = 1$.

Using the state probabilities it is possible to calculate the link utilization as (Kaufman, 1981):

$$U = \sum_{j=1}^{C} jq(j) \tag{3}$$

The number of FID requests for voice traffic: according to the assumption used in (Gabale et al., 2010), with a village population around 1000 there will be on average 10 simultaneous voice calls. Therefore for the UK, we will need approximately 15 GB/s (i.e. 660,000 calls 24000 b/s,) to allocate all voice calls assuming 50% utilization with zero blocking probability and G.729 compression coding algorithm is used. Therefore, the total capacity for this voice network is 30 GB/s. Also according to typical values reported in (Birke et al., 2007; Ramachandran and Beeram, 2009) the mean call duration, $1/\mu$, is 2 min., and with voice compression coding algorithm G.729 the total bandwidth required to transmit each voice over an IP based network would be 24 kb/s. While other codecs could be used, this gives a bandwidth midway between the older G.711 codec (likely to be less used by 2016) and other more recent low-bandwidth codecs. If we apply the parameters: mean call duration of 120 s, 30 GB capacity and mean call bandwidth 24 kb/s to the theoretical model described above, the mean time between calls is 0.0002 s. This means the topology management needs to create 5,000 new FIDs each second to route each voice call.

Voice traffic growth: the previously calculated number of FIDs for voice traffic is based on 2010 data. It is estimated that voice traffic has increased by an annual growth rate of 13% from the year 2010 until 2012 (Cisco, 2009), and there will be a further expected increase of 3% from 2012 until 2016 (Cisco, 2011). This growth allows an estimation of 7,185 FIDs/s required for voice communication in 2016.

4.2. Video model

Recent studies have indicated that video streaming is responsible for 25-40% of all Internet traffic and will reach 62% by the end of 2015 (not including P2P video file sharing) (Cisco, 2011). YouTube is the most popular Internet video service (Gehlen et al., 2012); consequently, YouTube is a good example of Internet video traffic to be applied to an ICN architecture, especially for the topology manager.

The number of FID requests for video traffic (YouTube): in order to know how many FIDs are needed to handle YouTube traffic by the topology management it is important to know how many requests/s are sent to the topology management. The study by Zinka, *et al.* (Zinka et al., 2009) measured YouTube traffic in a university campus of 25,000 students and found 4,000 requests sent per hour. So we can estimate that for the UK, with 52 million Internet users, the number of FIDs to be created by the topology management to handle all user requests is approximately 2,000 FIDs/s.

Video traffic growth: the YouTube data used above was from 2009. Video traffic has generally increased by an annual growth rate of 62% from 2009 until 2012, a further increase of 34% are expected from the year 2012 to 2016 (Cisco, 2011). Therefore the total number of FIDs requests of YouTube traffic by 2016 would be 87,507 FIDs/s.

4.3. HTTP and peer-to-peer traffic

Web traffic (HTTP) has been found to follow an exponential distribution (Mori et al., 2005; Bhole and Popescu, 2005). Study of Internet traffic at the University of Calgary (33,000 users) found web traffic arrival rate was 80 flows per second (Basher et al., 2008). Scaling this to the UK, gives approximately 126×10^3 FIDs/s for web traffic. Web traffic growth: this number of web requests is built on a study done on 2008. Using an annual growth rate of 26% from the year 2008-2012 (Cisco, 2009) and an expected growth rate of 35% from the year 2012- 2016 (Cisco, 2011) gives 1,054,841 FID/s by 2016 for web traffic.

In the same study of (Basher et al., 2008), it was found that peer-to-peer traffic has arrival rate of only 6 flows/s and from this we estimate that there will be around 9,400 FID/s for the UK based on 2008 figures. P2P traffic growth: Using an annual growth rate of 25% from the year 2008-2012 (Cisco, 2009) and 26% from the year 2012-2016 (Cisco, 2011) the total number of peer-to-peer traffic requests is expected to be 57,842 FID/s by the year 2016.

5. Results and discussion

Using the estimates of traffic models it is possible to investigate the number of FID updates needed for each type of traffic and also compare the total number of FIDs created by the topology management for all types of traffic to the total number of FIDs required to be updated. This can be done once the mean flow duration of each traffic type and the time between each updated Δt are known. Following measurements reported in the literature, we use mean durations of voice, YouTube, web and P2P flows as 2, 3, 0.033 and 266 minutes respectively (Birke et al., 2007; Zinka et al., 2009; Brownlee and Claffy, 2002; Steiner et al., 2009). Then following the traffic models described in Section 4 the number of FID required can be determined for each traffic type giving the results shown in Table 1. The results show the number of new FIDs required for each new publisher/subscriber event and the number due to updates for long-lived flows.

The results show that peer-to-peer flows have the highest number of updating requests with update percentage of more than 92% of sessions requiring updates. This is due to the long mean flow duration of peer-to-peer traffic. Web flows have the highest number of FIDs in a second with more than one million requests each second requiring a new FID, but almost no updates are required. This is because web traffic is characterized with a very short

Flow type	Number of new FID/s	New FIDs every Δt	Required updates each Δt	Peak updates/s	% of update
Voice	7185	8.62×10^6	329,725	686	3.8
Video	87507	1.05×10^8	561,769	1170	0.53
P2P	57,842	6.94×10^7	6.4×10^7	134,161	92.2
Web	1.05×10^6	1.27×10^9	≈ 0	≈ 0	≈ 0
All flows	1.20×10^6	1.45×10^9	6.49×10^7	136,017	4.5

Table 1: The number of FID/s (forwarding identifiers per second) that have to be created by the topology manager in order to deliver information items from publishers to subscribers. The number of updates/s represents the number of updated FIDs that need to be sent due to the security mechanism that requires periodic FID changes every Δt.

session periods. In the same table we show the total percentage of all FIDs of all types of flows that are required for update with all FIDs originally created by the topology management. This percentage is approximately 4.5% from the total flows requests, almost all of which is accounted for to P2P traffic. This is small and indicates that most sessions will be terminated before any update needs to take place. The total peak load on the topology management function, including the new updated FIDs, is approximately 1.3×10^6 FID/s.

With the knowledge of the peak FID calculation rate we can estimate the number of topology management entities required for handling all types of traffic requests which have been calculated *i.e.* 1.3×10^6 requests per second. Using the Blackadder ICN platform that is publicly available under GNU GPL2 license (Parisis and Trossen, 2013), we experimentally measured the average time taken by a single topology manager entity to create a unicast FID. The Blackadder was running on a machine with a processor capability of an Intel Core2 Quad CPU Q6600 @ 2.40 GHz 4 and RAM capacity was 4 GB. Only one core was used to carry out the processing. In this experiment we use a large real network topology, the KDL network with 754 nodes and 899 edges as reported by the Internet Topology Zoo (Knight et al., 2011). By running the experiment many times with different random publisher and subscribers we found that the average time is 0.26 ms/FID. This time is for processing one request received from the Rendezvous and includes finding the shortest path and creating a FID. For efficiency, once a path between a publisher and subscriber was found it was cached for later use.

Using the average FID calculation rate for single entity, the total number of topology manager entities to handle all traffic requests in a particular intra-domain can be calculated. Here we assume a traffic for the UK following the traffic models in Section 4. With 1.3×10^6 requests per second and 0.26 ms/FID we can estimate there is a need for approximately 340 topology manager cores, assuming similar CPU capabilities. By 2016 standards this is likely to be a modest capability. However, for a single carrier this is likely to be much smaller, for example in the UK, the largest carrier has approximately a 30% share of the total number of UK Internet users. Therefore, 102 topology managers are needed for a carrier of this size. However, the experiment of calculating the topology management capability has been carried out on a normal characteristic machine to estimate future traffic on 2016. This is a fairly modest CPU requirement and significantly less computational power than in current IP routing platforms that would be required at every forwarder. It should be remembered that the forwarding complexity of LIPSIN forwarding is considerably less than an IP or even an IP/MPLS router. For example a network node using LIPSIN forwarding with a node degree of 20 only needs $20 \times d$ LIT entries (with typically $d = 8$). This is favourable compared to an IP routing table that is approximately equal to the number of intra-domain links (typically hundreds in a large operator) if using IP/MPLS or equal to the number of BGP routing prefixes in a pure IP implementation. Consequently, the overall routing/forwarding complexity can be said to be considerably reduced compared to an existing IP infrastructure.

With the zFormation scheme, and its extension of FID update that we propose, the forwarding plane can resist an attacker's malicious packet injections, such as a denial of service attack. This is because the LITs become dynamic and changeable within certain periods thus any valid FID gets expired in due course. Consequently, determining a valid FID becomes difficult and an attacker that manages to determine a valid FID cannot use it to inject unwanted traffic indefinitely. Additionally, by including the In and Out interfaces in the FID construction process, an FID is tightly bound to a specific path and originator, so the attacker cannot reach another subscriber using an existing FID.

6. Conclusion

There are a number of ICN architectures that are being proposed with differing models for route determination. The scalability of these solutions needs careful inspection if we are to move from early prototypes towards proposing them as models for a future network operating at the scale of the Internet. Here we have considered an ICN architecture following the architecture proposed by the PSIRP/PURSUIT which has a central network function called topology management. We have extended this function to provide mitigation of injected malicious traffic by using periodically updated forwarding identifiers. This gives protection from attacks such as denial of service.

Although the topology management function is termed a central function we have shown that it can be distributed among an arbitrary number of topology manager instances. By analyzing traffic models of the current Internet and extrapolating them to levels in 2016 and applying them to the proposed architecture we have shown that the processing requirements are less than today's IP routing infrastructure. Thus we propose that this architecture is a viable proposition for a future ICN based network operating at the scale of a major Internet network provider.

References

Ahlgren, B., Dannewitz, C., Imbrenda, C., Kutscher, D., Ohlman, B., 2012. A survey of information-centric networking. Communications Magazine, IEEE 50 (7), 26–36.

Alzahrani, B., Vassilakis, V., Reed, M., 2013. Mitigating brute-force attacks on bloom-filter based forwarding. In: Future Internet Communications (CFIC), Conference on. pp. 1–7.

Alzahrani, B. A., Reed, M. J., Vassilakis, V. G., 2012. Enabling z-filter updates for self-routing denial-of-service resistant capabilities. In: Computer Science and Electronic Engineering Conference (CEEC), 4th. IEEE, pp. 100–105.

Basher, N., Mahanti, A., Mahanti, A., Williamson, C., Arlitt, M., 2008. A comparative analysis of web and peer-to-peer traffic. In: Proceedings of the 17th International Conference on World Wide Web. WWW '08. ACM, pp. 287–296.

Bhole, Y., Popescu, A., 2005. Measurement and analysis of http traffic. Journal of Network and Systems Management 13 (4), 357–371.

Birke, R., Mellia, M., Petracca, M., Rossi, D., 2007. Understanding VoIP from backbone measurements. In: INFOCOM. 26th IEEE International Conference on Computer Communications. IEEE. IEEE, pp. 2027–2035.

Bloom, B. H., 1970. Space/time trade-offs in hash coding with allowable errors. Communications of the ACM 13, 422–426.

Brownlee, N., Claffy, K., 2002. Understanding internet traffic streams: dragonflies and tortoises. Communications Magazine, IEEE 40 (10), 110–117.

Carrea, L., Vernitski, A., Reed, M., Jan. 2014. Optimized hash for network path encoding with minimized false positives. Computer Networks 58, 180–191.

Cisco, 2009. Cisco visual networking index: Forecast and methodology. Tech. rep.

Cisco, 2011. Cisco visual networking index: Forecast and methodology. Tech. rep.

Fotiou, N., Nikander, P., Trossen, D., Polyzos, G. C., 2012. Developing information networking further: From psirp to pursuit. In: Broadband Communications, Networks, and Systems. Springer, pp. 1–13.

Gabale, V., Raman, B., Chebrolu, K., Kulkarni, P., 2010. LIT MAC: Addressing the challenges of effective voice communication in a low cost, low power wireless mesh network. In: Proceedings of the First ACM Symposium on Computing for Development. ACM, pp. 1–11.

Gehlen, V., Finamore, A., Mellia, M., Munaf, M., 2012. Uncovering the big players of the web. In: Traffic Monitoring and Analysis. Vol. 7189 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 15–28.

Handley, M., 2006. Why the internet only just works. BT Technology Journal 24 (3), 119–129.

Jacobson, V., Smetters, D. K., Thornton, J. D., Plass, M. F., Briggs, N. H., Braynard, R. L., 2009. Networking named content. In: Proceedings of the 5th international conference on Emerging networking experiments and technologies. ACM, pp. 1–12.

Jokela, P., Zahemszky, A., Esteve Rothenberg, C., Arianfar, S., Nikander, P., 2009. LIPSIN: Line speed publish/subscribe inter-networking. ACM SIGCOMM Computer Communication Review 39 (4), 195–206.

Katsaros, K. V., Fotiou, N., Vasilakos, X., Ververidis, C. N., Tsilopoulos, C., Xylomenos, G., Polyzos, G. C., 2012. On inter-domain name resolution for information-centric networks. In: NETWORKING 2012. Vol. 7289 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 13–26.

Kaufman, J., 1981. Blocking in a shared resource environment. Communications, IEEE Transactions on 29 (10), 1474–1481.

Knight, S., Nguyen, H., Falkner, N., Bowden, R., Roughan, M., 2011. The internet topology zoo. Selected Areas in Communications, IEEE Journal on 29 (9), 1765–1775.

Koponen, T., Chawla, M., Chun, B.-G., Ermolinskiy, A., Kim, K. H., Shenker, S., Stoica, I., 2007. A data-oriented (and beyond) network architecture. ACM SIGCOMM Computer Communication Review 37 (4), 181–192.

Mori, T., Uchida, M., Goto, S., 2005. Flow analysis of internet traffic: World wide web versus peer-to-peer. Systems and Computers in Japan 36 (11), 70–81.

Parisis, G., Trossen, D., 2013. Blackadder node implementation, v0.4. Source code published through GitHub, accessed 8th March 2014.
URL https://github.com/fp7-pursuit/blackadder

Ramachandran, K., Beeram, S., 2009. Supporting enterprise-grade audio conferencing on the internet. In: Passive and Active Network Measurement. Springer, pp. 143–152.

Rothenberg, C. E., Jokela, P., Nikander, P., Sarela, M., Ylitalo, J., 2009. Self-routing denial-of-service resistant capabilities using in-packet bloom filters. In: Computer Network Defense (EC2ND), European Conference on. pp. 46–51.

Spagna, S., Liebsch, M., Baldessari, R., Niccolini, S., Schmid, S., Garroppo, R., Ozawa, K., Awano, J., 2013. Design principles of an operator-owned highly distributed content delivery network. Communications Magazine, IEEE 51 (4), 132–140.

Steiner, M., En-Najjary, T., Biersack, E. W., 2009. Long term study of peer behavior in the kad dht. IEEE/ACM Trans. Netw. 17 (5), 1371–1384.

Tagger, B., Trossen, D., Kostopoulos, A., Porter, S., Parisis, G., 2013. Realising an application environment for information-centric networking. Computer Networks 57 (16), 3249 – 3266.

Trossen, D., Parisis, G., 2012. Designing and realizing an information-centric internet. Communications Magazine, IEEE 50 (7), 60–67.

Trossen, D., Sarela, M., Sollins, K., Apr. 2010. Arguments for an information-centric internetworking architecture. SIGCOMM Comput. Commun. Rev. 40 (2), 26–33.

Vasilakos, X., Katsaros, K., Xylomenos, G., 2012. Cloud computing for global name-resolution in information-centric networks. In: Network Cloud Computing and Applications (NCCA), Second Symposium on. pp. 88–94.

Zinka, M., Suhb, K., Gua, Y., Kurosea, J., 2009. Characteristics of youtube network traffic at a campus network measurements, models, and implications. Computer Networks 53 (4), 501 – 514.