Decoupling the design of identifier-to-locator mapping services from identifiers

Luo, Hongbin; Zhang, Hongke; Zukerman, Moshe

Published in:
Computer Networks

Published: 10/03/2011

Document Version:
Post-print, also known as Accepted Author Manuscript, Peer-reviewed or Author Final version

License:
CC BY-NC-ND

Publication details:

Citing this paper
Please note that where the full-text provided on CityU Scholars is the Post-print version (also known as Accepted Author Manuscript, Peer-reviewed or Author Final version), it may differ from the Final Published version. When citing, ensure that you check and use the publisher's definitive version for pagination and other details.

General rights
Copyright for the publications made accessible via the CityU Scholars portal is retained by the author(s) and/or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights. Users may not further distribute the material or use it for any profit-making activity or commercial gain.

Publisher permission
Permission for previously published items are in accordance with publisher's copyright policies sourced from the SHERPA RoMEO database. Links to full text versions (either Published or Post-print) are only available if corresponding publishers allow open access.

Take down policy
Contact lbscholars@cityu.edu.hk if you believe that this document breaches copyright and provide us with details. We will remove access to the work immediately and investigate your claim.
Decoupling the Design of Identifier-to-locator Mapping Services from Identifiers
Hongbin Luo, Hongke Zhang, and Moshe Zukerman

Abstract
In order to address the scalability issue of the Internet routing architecture, there is a growing consensus that it is necessary to separate the locator role and the identifier role of IP addresses. When identifiers are separated from locators, a critical challenge is how to design an identifier-to-locator mapping service to map identifiers onto locators. While many mapping services have been proposed in the literature, they are designed based on either the structure or the hash of identifiers. That is, they are all designed based on identifiers. A result of such design choices is that users of identifiers do not have the freedom to choose where to store their identifier-to-locator mappings. In this paper, we argue that an identifier-to-locator mapping service should offer users of identifiers the freedom to choose where to store their mappings. For this purpose, we propose to decouple the design of identifier-to-locator mapping services from identifiers. In particular, our research results show that, by setting a set of identifier-to-locator mapping service providers (MSPs) and letting the users of identifiers choose their preferred/trusted MSPs, we can design an elegant mapping service to map identifiers onto locators.

Index Terms
Future Internet routing, locator/identifier separation, mapping service, scalability, security.

I. INTRODUCTION
As have been pointed out in many recent studies, the current Internet faces a serious scalability issue [1] - [6]. In order to address this issue, there is a growing consensus that it is necessary to separate the locator role and the identifier role of Internet Protocol (IP) addresses. That is, to use an identifier namespace to represent the identities of end hosts and to use a locator namespace to represent the locations of end hosts. In particular, it is proposed that an end host uses one or more identifiers, also called end point identifiers (EIDs) in [3], that do not change during its lifetime. On the other hand, the end host may change locators as it roams from place to place. After identifiers are separated from locators, identifiers are used in the application and transport layers to identify nodes, while locators are used in the network layer to locate nodes in the network. This allows nodes to change locators at any time without disrupting ongoing sessions and facilitates efficient mobility and multi-homing.

A critical challenge in locator/identifier separation is how to design a mapping service to map identifiers onto locators so that certain design objectives such as scalability and security are achieved [7]. In order to design such a mapping service, an important question is: where to store the identifier-to-locator mapping for a given identifier? Based on different assumptions, existing mapping services proposed in the literature [8] - [28] operate in three different ways.

- Firstly, some routers store the identifier-to-locator mappings for all identifiers. However, these approaches are not scalable since a single router cannot store all mappings when the number of identifiers becomes too large.
- Secondly, identifiers are assumed to be structured and assigned to organizations in a way that IP addresses are allocated. In addition, the identifier-to-locator mapping for a given identifier is stored at a node maintained by
the organization that the identifier is assigned to. That is, these approaches design mapping services based on the structure of identifiers.

- Thirdly, the mapping for an identifier is stored at a node chosen by using a deterministic hash algorithm. This implies that such approaches also design mapping services based on identifiers, although not based on the structures but based on the hash values of identifiers.

In summary, existing approaches either design the mapping service based on identifiers, or are unscalable. As a result, the user of an identifier must store the identifier-to-locator mapping for the identifier at a node assigned by the mapping service, whether he trusts the node or not.

After several decades of fast development, trust in the Internet is eroding. As written by Clark et al. in [29], “An Internet service provider (ISP) can try to force its customers to use its own email servers; most end-users today depend on the domain name system (DNS) server of the ISP (which influence where traffic is directed) without even thinking about whether it is wise to do so; and some ISPs try to force the end-user to use an ISP-provided Web proxy. More generally, there are many countries where all local agents may not be trustworthy, for reasons other than their use of inadequate or unreliable technology. . . . And in developed as well as developing countries there is a growing number of reasons for end-users to question the trustworthiness of available agents in at least some regards.” They further concluded that, “Each end user must make decisions about where services should be positioned so that they can be performed in a trustworthy manner. They can be positioned on a computer that is directly associated with the end-user, or they can be off-loaded to a service provider elsewhere in the network. A market of providers and subscribers gives the end-user control over which provider is selected to perform the service. Given choice, users can be expected to select services and service providers that they deem trustworthy.”

Since identifier/locator separation is proposed for addressing the scaling issue of the Internet, its corresponding mapping service will be deployed in the future. As a result, it is desirable that a new mapping service offers users the freedom to choose their preferred mapping service providers (MSPs) that they deem trustworthy. As stated above, however, existing mapping services cannot offer users this freedom.

In this paper, our goal is to design a mapping service that offers the user of an identifier the freedom to choose her/his preferred MSP and to change her/his MSP while keeping the identifier unchanged. In particular, we assume that there are many MSPs in the network and the user of an identifier freely chooses her/his preferred one. Once the user chooses an MSP to store the identifier-to-locator mapping for her/his identifier, she/he publishes the binding between the identifier and the MSP so that other users know where to find the identifier-to-locator mapping for the identifier. In addition, when the user subscribes to a new MSP possibly because she/he does not trust the old one any more or due to a movement, she/he only needs to publish the new binding between the identifier and the new MSP, without changing her/his identifier. We also propose a mechanism to secure mapping servers. In addition, we analyze the benefits of the proposed approach. We show that, except its capability to offer users the freedom to choose their preferred MSPs, the proposed approach is scalable, secure, and has very low resolution delay. We further present some numerical results to demonstrate the performance of our proposed approach.

The remainder of this paper is organized as follows. In Section II, we outline related work, focused on existing mapping services. In Section III, we present the network model and assumptions. In Section IV and Section V, we describe the proposed approach and its benefits, respectively. We present numerical results in Section VI and conclude the paper in Section VII.

II. RELATED WORK

In order to design a mapping service, there are three main design considerations. First, which types of identifiers are used? Second, for a given identifier, where is the identifier-to-locator mapping for the identifier stored? Third, how to distribute identifier-to-locator mappings. Based on the first consideration, existing work can be classified into three categories.

In the first category, identifiers are assumed to be hierarchial aggregatable, like IP addresses in the Internet today. The end point identifiers (EIDs) in [8] - [10], [13] - [17] are IPv4 or IPv6 addresses. As regard to the second consideration, however, these approaches are different. In [8] - [10], some nodes (e.g., default mappers [8]) in the network store the complete identifier-to-locator mappings for all identifiers. In addition, all these approaches assume that identifiers are allocated to various organizations in a way similar to what IP addresses are allocated in the Internet today. As a result, the identifier-to-locator mappings for identifiers sharing a same prefix are assumed to be
the same. Thus it entails to maintain only one mapping item for these identifiers that share the same prefix, which in turn significantly reduces the total number of mapping items. Notice that there are some differences among these approaches. In [10], all ingress tunnel routers (ITRs) in the network fetch the full mapping table. By contrast, in [8] and [9], only a small number of nodes in every autonomous system maintains the full mapping table, which ITRs query for mapping entries and cache the results. In any case, however, all mapping entry changes must be pushed out to many nodes [11], possibly by using approaches such as the one proposed in [12]. Even worse, given the rapid development of wireless technologies, the current trend is wireless everywhere. Thus the identifiers sharing a same prefix may not correspond to a common locator. As a result, the number of identifier-to-locator mapping items would be very huge so that it is impossible for a single node to store the full mapping table, which makes these approaches unscalable.

By contrast, the mapping services proposed in [13] - [17] assume that identifiers sharing a same prefix are assigned to an organization, which maintains the identifier-to-locator mappings for these identifiers that are assigned to it. In order to let tunnel routers (TRs) know where to find the identifier-to-locator mapping for an identifier, various overlay topologies have been proposed to distribute the reachability information of identifier-to-locator mappings. For example, the work in [15] proposes the use of distributed hash tables to distribute this information. The work in [13] proposes a hierarchial overlay architecture. When an ITR wants to know the identifier-to-locator mapping for an identifier, it sends a map-request into the overlay topology that forwards the map-request to the node that stores the identifier-to-locator mapping for the identifier. Therefore, when compared with the approaches in [8] - [10], these approaches have longer resolution delay since a map-request has to pass through the overlay topology. Notice that, in approaches proposed in [13] - [17], if an identifier is assigned to a particular organization, the location that stores the identifier-to-locator mapping for the identifier is also determined and does not change. Therefore, the user of the identifier has no choice but to store the identifier-to-locator mapping for the identifier in a server maintained by the organization, even if the user of the identifier leaves the organization and joins a new one that is far from the organization (e.g., moves from a company in Beijing to another one in Chicago).

In the second category, identifiers are assumed to be self-certifying [22] - [24], since it is believed that the use of such identifiers brings significant benefits in security [19] - [21]. In [23], the identifier-to-locator mappings are distributively stored at the domain name system (DNS) servers using a special resource record. In order to obtain the identifier-to-locator mapping for an identifier, a map request is sent into the DNS, which forwards the map request to the DNS server that stores the desired mapping. The benefit of this approach is that it is not necessary to build a new overlay architecture. However, this increases the overhead on the DNS since the DNS has to deal with mapping updates and map requests. In addition, the resolution delay may be very long in some cases, as observed in the current Internet. Furthermore, in order to keep the scalability of the DNS, it is required that the complete identifier-to-locator mappings are stored in an aggregatable manner. As a result, the user of an identifier cannot determine/choose where to store the identifier-to-locator mapping for the identifier. Unlike [23], [22] and [24] propose to distribute identifier-to-locator mappings to many resolution nodes using distributed hash tables (DHTs). While [22] uses the well-known OpenDHT [25] to organize the resolution nodes, [24] constructs these nodes into a content-addressable network [26]. In both cases, the identifier-to-locator mapping for an identifier is stored at the node that the identifier hashes onto. Therefore, the user of an identifier cannot choose where to store the mapping for the identifier either.

In the third category, identifiers are assumed to be hierarchial [27], [28]. In [28], an identifier comprises two parts: an administrative domain ID and a local host ID. Every administrative domain maintains a resolution node that stores the identifier-to-locator mappings for the identifiers that belong to the administrative domain. On the other hand, all resolution nodes are then organized using DHTs. In order to resolve the identifier-to-locator mapping for an identifier, the mapping service first uses the first part of the identifier to forward a map request to the resolution node that stores the identifier-to-locator mapping for the identifier. Thus, like [22] and [24], a map request may also pass through multiple hops before it reaches the desired resolution node. In addition, once an identifier is determined, its first part is also determined. As a result, the resolution node that stores the identifier-to-locator mapping for the identifier is also determined, regardless of wherever the identifier is used and whoever uses it. Therefore, the user that uses the identifier cannot choose his/her preferred mapping service provider to store the identifier-to-locator mapping for the identifier. The basic idea of [27] is similar to that of [28], except for how they distribute the reachability information of the first part. Notice that the approach in [27] also offers the possibility for users to change their preferred MSP. However, in [27], once a user changes its preferred MSP, the corresponding
identifier also changes. By contrast, in our proposed approach, identifiers always remain unchanged when users change their preferred MSPs.

The work in [30] and [31] analyze the cost to cache identifier-to-locator mappings at tunnel routers. A further analysis on how indirect mapping makes an identifier-to-locator mapping service more scalable and mobility-efficient is provided in [32].

In summary, it is difficult for existing mapping services to offer users of identifiers the freedom to change their preferred MSPs while keeping their identifiers unchanged.

III. NETWORK MODEL

Like in [3] - [6], we consider a network in which customer networks are separated from provider networks, as illustrated in Figure 1. Customer networks are composed of the sources and destinations of data packets. Provider networks comprise the Internet core and are used for providing data transit service at the inter-domain level. Customer networks connect with provider networks through tunnel routers (TRs) located at edges of either provider networks or customer networks. Provider networks use globally routable locators to route packets. On the other hand, customer networks may use locally routable locators to route packets as in [28], or use identifiers to route packets as in [3]. Notice that we do not assume the structures of globally routable locators and identifiers. For example, identifiers may be either hierarchically structured [3] or flat [21].

Every provider network (PN) has a globally unique PN number (e.g., an autonomous system (AS) number) that uniquely identifies the identity of the PN. In addition, every PN has a public/private key pair that is either issued by a public trusted entity or self-certifying [21]. The public keys of provider networks are distributed to all TRs, each of which then stores the public keys of all provider networks. Since customer networks are separated from provider networks, the number of provider networks in the Internet core is fairly small (e.g., ten thousands). As a result, TRs are able to store these public keys. Furthermore, we assume that some provider networks serve as MSPs that map identifiers onto globally routable locators. In addition, if a PN serves as an MSP, it should maintain one or more resolvers, as illustrated in Figure 1. In the case that a PN maintains multiple resolvers, the owner of the PN may organize them into a resolution system that is viewed as a resolver in the rest of the paper for ease of presentation.

For ease of presentation, here we assume that the network knows how to route a packet to a resolver based on the PN number (e.g., using AS number-based routing as in [21]). We will discuss how our approach can be used in other cases at the end of the next section.

Packets to and from two nodes attached to a same TR are directly routed to each other without passing through the TR. When two hosts are attached to different TRs, however, packets are generally tunneled from one TR to another one. We illustrate this through an example shown in Figure 1, assuming that a mobile node (MN) with $EID_{src}$ in edge network $CN_1$ wants to establish a connection to another end host with $EID_{dst}$ in another edge network $CN_2$. We also assume that both edge networks use identifiers to route packets in their own networks. For ease of presentation, we use identifier-to-locator mapping and EID-to-locator mapping interchangeably in the rest of this paper.

Fig. 1. Illustration for the network model considered in this paper, assuming that TRs are placed at the borders of provider networks.
Step 1: The source host $EID_{src}$ first sends a data packet to its ITR, i.e., $TR_1$ in Figure 1. The source and the destination of the packet are $EID_{src}$ and $EID_{dst}$, respectively, as illustrated in Figure 1.

Step 2: Every ITR maintains an EID-to-locator mapping table that stores some recent used mapping items. When $TR_1$ receives the packet from the source node, it looks up its local EID-to-locator mapping table to locate an EID for $EID_{dst}$, as illustrated by (2) in Figure 1. If the cache hits, go to Step 6; otherwise, go to Step 3.

Step 3: Assume that the EID-to-locator mapping for $EID_{dst}$ is stored at $RV_3$. $TR_1$ sends a map request to $RV_3$ in order to resolve a locator (or a set of locators) for $EID_{dst}$, as illustrated by (3) in Figure 1. In addition, $TR_1$ sends the packet destined to $EID_{dst}$ to $RV_3$ when it sends the map request in order to reduce packet loss.

Step 4: When $RV_3$ receives the map request, it sends the identifier-to-locator mapping for $EID_{dst}$ to $TR_1$, as illustrated by (4) in Figure 1.

Step 5: At the same time, $RV_3$ sends the packet to $TR_3$, as illustrated by (5) in Figure 1.

Step 6: When $TR_1$ receives the resolved locators for $EID_{dst}$, it locally caches them into its EID-to-locator mapping table for possible future usage. Denote the resolved locator for $EID_{dst}$ by $Loc_2$, which is the locator of destination’s TR (i.e., $TR_2$ in Figure 1). In addition, we denote the locator of $TR_1$ by $Loc_1$. $TR_1$ encapsulates the received packet with its outer header whose destination address and source address are $Loc_2$ and $Loc_1$, respectively. $TR_1$ then sends the encapsulated packet, which would be routed to its egress TR (ETR), i.e., $TR_2$, as illustrated by (6) in Figure 1.

Step 7: When $TR_2$ receives the encapsulated packet, it strips the outer header of the encapsulated packet, as illustrated by (7) in Figure 1. In order to eliminate the need for a mapping lookup for the EID-to-locator mapping of $EID_{src}$, $TR_2$ may store the mapping from $EID_{src}$ onto $Loc_1$ in its local EID-to-locator mapping table.

Step 8: $TR_2$ then sends the packet to its destination $EID_{dst}$, as illustrated by (8) in Figure 1.

From the above description, one can clearly see that the mapping service is of critical importance. It is worth noting that, in case of cache miss, a packet destined to an identifier is firstly sent from the packet’s ITR to the resolution node storing the identifier-to-locator mapping for the identifier, and then to the packet’s ETR. Let $D_s$ be the propagation delay that a packet incurs from its ITR to ETR in the case of cache miss and $D_d$ be the propagation delay that the packet would incur if it is directly sent from its ITR to ETR. In general, $D_s$ is longer than $D_d$ since the resolution node is not on the direct path from the ITR to the ETR.

IV. THE PROPOSED APPROACH

In this section, we describe the proposed approach in detail. Due to its salient feature in offering users the freedom to choose their preferred/trusted MSPs, we call the proposed approach self-selectable mapping (SSM). For ease of description, we introduce the following two definitions. First, if an MSP is chosen by the user of an EID, it is called the subscribed MSP of the EID. By contrast, if an EID attaches to a PN, the PN is called the EID’s attached PN (A-PN). In SSM, the subscribed MSP of an EID may or may not be the EID’s PNA. For example, the host $EID_{dst}$ in Figure 1 may choose $PN_3$ to be its subscribed MSP, although its A-PN is $PN_2$. On the other hand, the host $EID_{src}$ in Figure 1 may choose its A-PN (i.e., $PN_1$) as its subscribed MSP.

A. Basic Idea

The basic idea behind SSM is very simple: decoupling the design of identifier-to-locator mapping services from identifiers. In particular, we assume that some PNs serve as MSPs that map identifiers onto locators. In addition, we let the user of an EID choose his/her preferred MSP to store the identifier-to-locator mapping for the EID. Once a user chooses an MSP for his/her identifier, the MSP assigns the identifier a signature that is used to verify whether or not the identifier subscribes to the MSP. The user of the identifier then publishes its identifier, the identifier’s subscribed MSP and the corresponding signature so that other users know about this. When other hosts communicate with the end host using the identifier, they send the identifier, the identifier’s subscribed MSP and the corresponding signature to their ITRs. The ITRs then verify whether or not the identifier subscribes to the indicated MSP and, if so, send map-requests to the indicated MSP in order to resolve an identifier-to-locator mapping for the identifier.
In the following subsections, we describe how to choose an MSP for an EID, how to distribute the MSP for an EID, and how to deal with map registrations and map requests in SSM.

B. Choose an MSP for an EID

SSM offers the users of EIDs the freedom to choose their preferred MSPs to store the identifier-to-locator mappings for their EIDs. Although the user of an EID may freely subscribe to any MSP in principle, there are some considerations.

Firstly, it would be better for the user of an EID to choose her/his EID’s A-PN or an MSP that is very close to the EID’s A-PN as the EID’s subscribed MSP. Recall that when an ITR receives a packet destined to an identifier and a cache miss occurs, the ITR sends the packet to the identifier’s subscribed MSP, which then sends the packet to the packet’s ETR. As stated previously, this generally leads to longer packet propagation delay since the subscribed MSP may not be on the direct path from the ITR to the ETR. However, since an identifier’s A-PN is generally on this path, choosing it as the identifier’s MSP makes the propagation delay longer than the propagation delay of the packet when it is sent directly from its ITR to its ETR. We notice that, with technological advancements such as cable and Worldwide Inter-operability for Microwave Access (WiMAX) [42], it is possible for an identifier to simultaneously have multiple A-PNs. In this case, the user of the identifier may choose one A-PN as the identifier’s subscribed MSP based on its personal policies such as the service quality of every A-PN and price. At the same time, it is possible that the A-PNs of an identifier do not provide mapping service. In this case, it would be better for the user of an identifier to choose an MSP that is very close to the identifier’s A-PN.

Secondly, in order to subscribe to an MSP, there are two approaches. In the first approach, the user of an EID may directly send a subscribe request whose destination is the desired MSP of the EID. Notice that, in order to forward such a packet, the transit space only needs to know the PN number of the desired MSP. As assumed in the above section, every PN has a PN number with which the transit space can route packets to the PN. When the desired MSP receives such a request, it assigns a signature for the EID. However, compromised end hosts may attack an MSP by sending many such requests to the MSP. Therefore, we prefer the second approach described below. In the second approach, every MSP has many service offices that are distributed city-wide, nation-wide, or ideally world-wide. If the user of an EID wants to subscribe to a given MSP, he/she only needs to go to a service office of the MSP and obtain a signature for the EID. Ideally, the signature would be embedded into the network interface card that uses the EID; it is similar to the case that a SIM card also embeds some authentication information for a given mobile phone number. While this approach entails that every MSP has many service offices that cost the MSP a substantial amount, the MSP may charge its users through their subscriptions. Of course, a PN may offer free mapping service in order to attract users.

Once an MSP is chosen by an EID to be its subscribed MSP, it deals with identifier-to-locator registration requests and identifier-to-locator map requests for the EID. Furthermore, it is also in charge of dealing with other related issues. For example, once a cache miss for packets destined to an EID occurs at an ITR, the ITR may send such packets to the EID’s subscribed MSP. In this case, the EID’s subscribed MSP forwards these packets to the end host that uses the EID. As another example, when an end host with an EID moves from an old TR to a new TR, the handover mechanism proposed in [32] entails that the EID’s subscribed MSP informs the old TR about the new identifier-to-locator mapping for the EID immediately after it receives a new mapping from the new TR.

C. Distribute the MSP for an EID

Since the subscribed MSP of an EID is separated from the EID, an ITR cannot know from the EID itself where to resolve/register the EID-to-locator mapping for the EID. Therefore, it is critical to let ITRs simultaneously know EIDs, the PN numbers of their subscribed MSPs and corresponding signatures. For this purpose, we assume that whenever the owner of an EID publishes the EID to some systems so that the EID is known by others, the owner also publishes the PN number of its subscribed MSP and the corresponding signature simultaneously. For example, when we register a domain name to the DNS in the Internet today, we bind the domain name to an IP address. In SSM, we should bind the domain name not only to an EID but also to the PN number of the EID’s subscribed MSP and to the corresponding signature. As a result, whenever a DNS resolution returns an EID, it also returns the PN number of the EID’s subscribed MSP and the corresponding signature. Similarly, when the owner of an EID publishes its availability for some resources such as a file, a movie, or a song into a resource sharing system
such as Napster [35] and Gnutella [36], the owner of the EID should also publish the EID, the EID’s subscribed MSP and the corresponding signature to the resource sharing system. The owner of an EID may also directly tell a user the EID’s subscribed MSP and the corresponding signature. For example, when Alice wants to use Bob’s computer through Telnet, Alice may ask Bob to tell her the EID of the computer, the EID’s subscribed MSP and the corresponding signature.

While one may argue that, since the distribution for the MSP for an EID is dependent on the DNS or some resource sharing systems, why do we not simply use solutions such as dynamic DNS but need another mapping system? To answer this question, we first notice that, with the fast increase in the number of mobile devices such as laptops, portable digital assistants (PDAs), intelligent mobile phones, end hosts frequently change their point-of-attachments. As a result, the identifier-to-locator mappings for EIDs change frequently. Therefore, using DNS to deal with identifier-to-locator map registrations and map requests would significantly overload the DNS. Furthermore, DNS heavily relies on the well-known cache mechanism. If DNS is used to map identifiers onto locators, it is required to distribute the new identifier-to-locator mapping for an EID to all possible DNS servers that cache an old identifier-to-locator mapping for the EID since, otherwise, the packets sent to the EID cannot be routed to the EID and will be discarded. However, this update again imposes significant overhead to the DNS. This is also a main reason that a lot of EID-to-locator mapping services have been proposed.

D. Mapping Registration

When an end host (with an EID) attaches to a new TR, the new TR needs to register an EID-to-locator mapping for the EID into the resolver of the EID’s subscribed MSP. For example, $TR_2$ may register for the destination host $EID_{dst}$ in Figure 2 an EID-to-locator mapping for $EID_{dst}$ when it detects the attachment of the destination host. Notice that, in order to support efficient host mobility, the registered EID-to-locator mapping for $EID_{dst}$ may be the mapping from $EID_{dst}$ onto the locator of a local mobility anchor (LMA), instead of the mapping from $EID_{dst}$ onto the locator of $TR_2$. For example, in Figure 2, the registered locator for $EID_{dst}$ by $TR_2$ may be the locator ($i.e., Loc_3$) of the LMA in $PN_2$. In this case, it should be guaranteed that the registered locator for an EID is in the same PN with the TR so that the EID’s subscribed MSP can verify the authenticity of the registration. However, how to determine the LMA for an EID and how to route packets in that case are beyond the scope of this paper. We refer interested readers to [32] for further information on these issues.

Now we describe a mapping registration process with the help of Figure 2, assuming that $EID_{dst}$ subscribes to $PN_3$ and $TR_2$ registers the mapping from $EID_{dst}$ onto $Loc_2$ to $RV_3$.

Step 1: When the new TR detects the attachment of the end host, it verifies whether or not the end host does own the EID, possibly using mechanisms such as Unicast Reverse Path forwarding (uRPF) [37] or the authentication approach proposed in [21]. If the end host does own the EID, the new TR lets the end host tell the EID, the EID’s subscribed MSP and the corresponding signature. This is illustrated by (1) in Figure 2.

Step 2: When the new TR receives the EID, the EID’s subscribed MSP and the corresponding signature, it checks whether or not the EID really subscribes to the indicated MSP by verifying the authenticity of the signature, as illustrated by (2) in Figure 2. If the EID really subscribes to the indicated MSP, go to Step 3; otherwise, the new TR simply does not register any mapping for the EID.

Step 3: The new TR sends a registration request to the EID’s subscribed MSP to register an EID-to-locator mapping for the EID, as illustrated by (3) in Figure 2. In addition, the new TR should sign the registration request before sending it to the EID’s subscribed MSP. Since every PN has a PN number and a public/private key pair, a TR in a PN can sign registration requests using the private key of the PN. In addition, since TRs maintain public keys of all PNs, they can verify the authenticity of a signature.

Step 4: When the EID’s subscribed MSP receives the mapping registration request for the EID, it verifies the authenticity of the registration request in order to check whether or not the registration request comes from a legal PN. If so, the EID’s subscribed MSP checks whether or not the EID does really subscribe to it. If so, the EID’s subscribed MSP stores the EID-to-locator mapping contained in the mapping registration request in its resolvers, as illustrated by (4) in Figure 2. An MSP may maintain multiple resolvers and organize them based on its local policy.

Step 5: The EID’s subscribed MSP sends an acknowledgement message to the new TR and the end host that uses the EID, as illustrated by (5) in Figure 2. The acknowledgement message should include the PN number.
of the provider network where the new TR is located. When the end host that uses the EID receives such an acknowledgement message, it can verify whether or not it really attaches to the indicated provider network.

Notice that, a TR only registers an EID-to-locator mapping for an end host with an EID only if the end host does own the EID and the end host provides the correct EID, the EID’s subscribed MSP and the corresponding signature. This makes it very difficult for an end host to trigger EID-to-locator mapping registrations for an EID that the end host does not own. In addition, since a TR is informed where to send the mapping registration request, this request can be directly sent through the routing architecture to the resolver that stores the desired EID-to-locator mapping. By contrast, in most existing mapping services, mapping registration requests transverse an overlay topology, which not only incurs longer registration delay but also entails building an overlay topology to forward registration requests.

E. Mapping Resolution

When an ITR receives a packet destined to a destination EID, it is possible that the ITR cannot find an EID-to-locator mapping for the EID in its local EID-to-locator mapping table. In this case, the ITR needs to resolve an EID-to-locator mapping for the EID. A mapping resolution in SSM comprises the following steps.

Step A: Since an ITR cannot know where to resolve an EID-to-locator mapping for the destination EID, the source end host must send the destination EID’s subscribed MSP and the corresponding signature to the ITR, as illustrated by (a) in Figure 2.

Step B: When the ITR receives the destination EID’s subscribed MSP and the corresponding signature, it verifies the authenticity of the signature, as illustrated by (b) in Figure 2. If the signature is authenticated, go to Step C. Otherwise, the ITR does not proceed further and may treat the source host as suspicious since the source host may attack the ITR by forging EID’s subscribed MSP and the corresponding signature.

Step C: The ITR sends a resolution request to the destination EID’s subscribed MSP in order to resolve an EID-to-locator mapping for the destination EID, as illustrated by (c) in Figure 2. Notice that the ITR should sign the resolution request so that the EID’s subscribed MSP can verify the request’s authenticity. Notice that, in order to reduce packet loss, the ITR sends the packet along with the resolution request to the EID’s subscribed MSP, which then forwards the packet to its ETR.

Step D: The destination EID’s subscribed MSP first verifies the authenticity of the resolution request. If the resolution request is authenticated, the destination EID’s subscribed MSP returns an EID-to-locator mapping to the requesting ITR, as illustrated by (d) in Figure 2.

When the ITR receives an EID-to-locator mapping for an EID, it stores the mapping for a period called time-to-live (TTL). When there are any packets sent to the EID before the TTL expires, the TTL for the mapping is reset to its default value. On the other hand, if the mapping is never used before its TTL expires, the mapping is removed from the ITR’s cache. In this case, the ITR needs to resolve an EID-to-locator mapping for the EID again if it receives a packet destined to the EID.
Notice that whenever an end host with an EID attaches to a TR, the TR verifies whether or not the end host owns the EID. When the TR receives a packet from an end host, it is able to verify whether or not this packet is sent from the end host. This in turn guarantees that an end host triggers a mapping resolution request only when it sends out packets with source EID being the one that the end host does own. In addition, caching some recently used EID-to-locator mappings at ITRs significantly reduces the number of mapping resolution requests sent to MSPs. Furthermore, an ITR sends out a mapping resolution request for an EID only when it assures that the EID, the EID’s subscribed MSP and the corresponding signature are correct. Therefore, an end host cannot trigger a mapping resolution request for an EID if it does not know the EID’s subscribed MSP and signature. Accordingly, packets that are sent by the end host and destined to the EID cannot be routed to the end host using the EID. By contrast, in the current Internet, whenever an end host sends a packet with a destination IP address, the packet would be routed to the end host using the IP address.

F. Dealing with Signatures

An important aspect of SSM is its use of signatures to improve the security of the MSPs. Since signatures in SSM are verified by ITRs, they must be verifiable by any ITR instead of a particular one. At the same time, signatures must not be forgeable so that, for any given MSP, ITRs do not send it map requests for identifiers that do not subscribe to it. While one approach that achieves this is to use public-key signatures, they are too expensive to create and to verify.

In order to avoid the drawbacks of public-key signatures, as in [38], we use trapdoor hash functions with inversion property for signing and verification. A trapdoor hash function involves a trapdoor key (private key) \( TK \) and a hash key (public key) \( HK \) [39]. The trapdoor hash function \( H(\cdot) \) is a one-way hash function. That is, knowing \( HK \), computing \( H(x) \) for any \( x \) is computationally inexpensive, but the reverse, namely to find the input \( x \) that generates any \( H(x) \) in very computationally expensive. Because it is a one-way hash function, the trapdoor hash function has what is called the collision-free property. This means that it is difficult to find two inputs \( x_1 \) and \( x_2 \) so that \( H(x_1) \neq H(x_2) \) if \( x_1 \neq x_2 \). A trapdoor hash functions also has the property that if the trapdoor key \( TK \) is known, it is easy to find collisions. Trapdoor hash functions are said to have the inversion property if finding the input \( x \) that generates hash value \( H(x) \) is computationally inexpensive if the trapdoor key \( TK \) is known.

For a given EID that subscribes to an MSP with a hash key \( HK \) and a trapdoor key \( TK \), the MSP computes a signature for the EID as follows. The MSP uses the trapdoor key \( TK \) to find \( r \) so that \( H_{MSP}(r) = V(MSP) \colon EID \), where \( : \) denotes concatenation and \( V(MSP) \) is a token that the MSP publishes when it publishes the hash key \( HK \). Due to the use of trapdoor hash functions with inverse property, this calculation is easy since the MSP knows its trapdoor key \( TK \). By contrast, it is computationally expensive for other entities to forge a correct signature for the EID since they do not know the trapdoor key of the MSP.

When a TR receives an EID, its subscribed MSP and the corresponding signature \( r \), it checks the signature by calculating the hash over \( r \) and verifying that it is equal to \( V(MSP) \colon EID \). This way, a TR only needs to perform a hash operation and a comparison operation in order to verify whether or not an EID really subscribes to a given MSP.

In order to prevent keys from being compromised, an MSP periodically changes its \( HK \) and \( V(MSP) \) values. In this case, the MSP needs to update all signatures for those identifiers that subscribe to it, and to inform all TRs about the change of the \( HK \) and \( V(MSP) \) values. In order to inform all TRs about the new \( HK \) and \( V(MSP) \) values, an MSP pushes the new \( HK \) and \( V(MSP) \) values to a representative node (server or router) in each AS. This node then updates all TRs in its same AS with the new \( HK \) and \( V(MSP) \) values. While these updates would increase the bandwidth requirements, we anticipate that MSPs update their \( HK \) and \( V(MSP) \) values once every several days (e.g., once a week).

In order to update the signatures for identifiers, the MSP may compute off-line a new signature for each identifier that subscribes to the MSP and notify the end host that uses the identifier when the MSP decides to change its \( HK \) and \( V(MSP) \) values. When the end host receives the new signature for the identifier, it then publishes this new signature to the DNS or some other file-sharing system. One side effect of this update is that this may cause more dynamics to the DNS or some other file-sharing systems. However, if we take the DNS as an example, we notice that an end host only needs to update its authoritative DNS server with the new signature of the identifier. While one may argue that some other DNS servers may cache the old signature for the identifier, the cache time is very
short (from one hour to one day [40]). In addition, it is reported that reducing the cache time to as low as a few hundred seconds has little adverse effect on hit rates of DNS resolutions [40]. Furthermore, many DNS records in the current Internet update once every few seconds to a few hours [41], a frequency significantly higher than the expected update frequency of $HK$ and $V(MSP)$ values. Therefore, the update of signatures only cause very limited dynamics to the DNS. Another possible side effect is that some DNS servers do cache an outdated signature. In order to deal with this issue, we let TRs in the network preserve old $HK$ and $V(MSP)$ values for some limited time to validate these outdated signatures. At the same time, we may also use the approach proposed in [41] to provide strong cache consistency for DNS.

G. Other Issues

In the above descriptions, we have assumed that the network knows how to route packets based on PN numbers. In practice, however, this may not hold. For example, the current Internet does not forward packets based on AS numbers. Here, we argue that, with only limited modifications, SSM can also be used in that case.

Recall that when an EID subscribes to an MSP, the MSP assigns a signature to the EID. The user of the EID then publishes the EID, the subscribed MSP and the signature. In addition, the signature $r$ is generated using a trapdoor hash function so that $h(r) = V(MSP) : EID$. If the network cannot route packets based on PN numbers, the above steps are modified as follows:

1) When an EID subscribes to an MSP, the MSP assigns both a signature and a locator to the EID. The locator should be the locator of a resolver or that of a set of resolvers (e.g., like an anycast address in the Internet today) that store the EID-to-locator mapping for the EID.

2) The user of the EID then publishes the EID, the subscribed MSP, the assigned locator, and the signature.

3) The signature $r$ should be generated using a trapdoor hash function so that $h(r) = V(MSP) : EID : locator$.

In addition, when an end host $A$ wants to communicate with another host $B$ with $EID_B$, host $A$ should send $EID_B$, $EID_B^{\text{sub}}$, subscribed MSP, the assigned locator, and the signature to its ITR. When the ITR receives such information, it verifies that $EID_B$ subscribes to the indicated MSP. If so, it sends a map-request to the assigned locator and the routing system would routes the map-request to the resolver(s) storing the EID-to-locator mapping for $EID_B$. The resolver then returns an EID-to-locator mapping for $EID_B$ to host $A$’s ITR.

V. SSM’S ADVANTAGES

Compared with existing approaches, SSM has many advantages including scalability, enhanced security, free subscription, and low resolution delay.

A. Scalability

SSM is inherently scalable for the following reasons. Firstly, SSM does not need to propagate identifier-to-locator mapping reachability information for identifiers. As a result, no network entity needs to maintain identifier-to-locator mapping reachability information. Therefore, network entities do not suffer from the increase in the number of identifier-to-locator mapping items, and are not affected by the structures of identifiers and how they are allocated to users/organizations. While resolvers maintained by MSPs are required to store identifier-to-locator mappings for identifiers and are affected by the increase of the number of mappings, they do not need to maintain any identifier-to-locator mapping reachability information. In addition, mappings are distributed to all resolvers instead of a single one. When the number of mapping items increase, we can use multiple resolvers to store them. By contrast, many existing approaches use an overlay topology to propagate identifier-to-locator mapping reachability information and some network entities are required to maintain a table that stores the reachability information of identifier-to-locator mappings. As a result, the table size at such network entities may be affected by various factors such as the structure of identifiers or how they are allocated and may not be scalable.

Secondly, since no network entities need to maintain identifier-to-locator mapping reachability information, it is not necessary for network entities to process updates of identifier-to-locator mapping reachability information. While resolvers maintained by MSPs are required to deal with updates of identifier-to-locator mappings, it is not necessary for them to deal with identifier-to-locator mapping reachability information. By contrast, if an overlay network is used to propagate identifier-to-locator mapping reachability information, some entities in the overlay network have to process updates of such information. This not only consumes significant network resources but also limits the scalability of the overlay network.
B. Enhanced Security

SSM is more secure than existing approaches. Firstly, SSM does not need an overlay topology to propagate identifier-to-locator mapping reachability information. As a result, SSM does not need to protect any overlay topology. By contrast, many existing approaches use an overlay topology to distribute identifier-to-locator mapping reachability information, which makes the overlay topology a possible target of attacks. Secondly, MSPs in SSM are less likely to be attacked than those in other schemes. In particular, once an identifier subscribes to an MSP, the MSP assigns the identifier a signature. In addition, ITRs send a map-request/map-registration for an identifier to an MSP only when they verify that the identifier does subscribe to the MSP. While it is possible that compromised end hosts attack a specific MSP by using forged signatures, signatures are updated once in several days. Accordingly, the probability that signatures are forged is significantly reduced. Thirdly, even if ITRs send map-request/map-registration messages to a given MSP, these messages are routed to all routers of the MSP instead of a particular router. As a result, the MSP can adopt appropriate mechanisms to balance load, thus reducing the probability that a particular router is attacked. Fourth, while ITRs need to verify signatures, this verification is very simple since it requires only a hash operation and a comparison operation.

Another benefit of SSM is that it helps improve the security of end hosts. Indeed, when other hosts want to send packets to an end host, they have to know the identifier of the host, the identifier’s subscribed MSP and the corresponding signature. Without the correct signature, packets sent by them cannot be routed to the end host. By contrast, in the current Internet, any packets destined to an IP address can be routed to the end host that uses the IP address, which makes it easy for end hosts to be attacked. In SSM, for those end hosts that act as clients (with respect to servers), they do not need to publish their identifiers and corresponding signatures. Notice that even if the end host roams from place to place, an end host does not need to tell its corresponding node (CN) about the end host’s subscribed MSP and the corresponding signature when it initiates a communication with the CN. Indeed, with the handover process proposed in [32], the MN’s ITR will tell the CN’s ITR about the movement of the MN and the CN’s ITR does not need to resolve the new EID-to-locator mapping for the MN. Therefore, although the servers may be attacked since their subscribed MSPs and corresponding signatures are published, SSM helps protect the clients from being attacked. While one may argue that those EIDs who publish their subscribed MSPs and corresponding signatures may still be attacked, our goal here is not to protect all end hosts. They may be protected using other approaches like in [48].

C. Free Subscription

A significant benefit of SSM is that it offers the users of identifiers the freedom to choose their preferred MSPs. When a user feels that her/his subscribed MSP cannot offer satisfactory mapping service, she/he can change her/his MSP without changing the EID. This not only benefits users but also encourages MSPs to provide better mapping services.

Notice that the freedom of subscription may also benefit host mobility. For example, with the handover process proposed in [32], it is reported that the shorter is the delay between the ITRs that an MN attaches to and the MN’s subscribed MSP, the shorter the handover delay is. In reality, many persons move to a new city after living a certain period in another city. In this case, a person may subscribe to an MSP in the new city so that his handover delay would be kept very short. By contrast, in most existing approaches, the node that stores the identifier-to-locator mapping for an identifier is fixed. As a result, when the user of the identifier moves to a new city, the identifier-to-locator mapping for the identifier is still stored at the node in the former city. Therefore, the handover delay increases since the delay between an ITR and a node in the new city is generally shorter than that between an ITR in the new city and a node in the former city.

D. Low Resolution Delay

In SSM, an ITR does not need to find where to resolve the EID-to-locator mapping for a given EID, since the source host that wants to communicate with the EID tells the ITR where to find the EID-to-locator mapping for the EID. In particular, if we let $D_{RTT}$ be the round-trip time (RTT) between the ITR and the ETR, the resolution delay in SSM is very close or smaller than $D_{RTT}$. Figure 3 qualitatively illustrates this, assuming that $TR_1$ in $PN_1$ is an ITR and $TR_2$ in $PN_2$ is an ETR. In addition, the end host $EIR_{dst}$ is attached to $PN_2$ and chooses $PN_2$ as its MSP. When the ITR $TR_1$ receives a packet destined to $EIR_{dst}$ and cannot locally find a mapping for
**EIR**$_{dst}$, it sends a map-request to a resolver (e.g., $RV_2$) in $PN_2$. When $RV_2$ receives the map-request, it returns the identifier-to-locator mapping for $EIR_{dst}$ to $TR_1$. Since $RV_2$ and $TR_2$ are in the same provider network, the RTT between $TR_1$ and $TR_2$ is very close to the RTT between $TR_1$ and $RV_2$. Notice that users of identifiers in SSM can freely choose their preferred MSP. As a result, if the user of $EIR_{dst}$ chooses an MSP (e.g., $PN_3$) that is closer to the network core as the MSP of $EIR_{dst}$, the RTT between the ITR $TR_1$ and $RV_3$ is smaller than that between the ITR and the ETR. Therefore, the RTT between the ITR and the ETR can be viewed as an upper bound of the resolution delay in SSM.

By contrast, in most existing mapping services (e.g., locator/ID separation protocol alternative topology (LISP+ALT) [13]), a map-request will be forwarded through an overlay topology and requires longer resolution delay than in SSM. In particular, the work in [44] compares the resolution delay of two representative mapping services: LISP+ALT [13] and LISP-DHT [15]. Both mapping services are proposed for LISP and let an ETR located at the border of a customer network store identifier-to-locator mappings for these identifiers that are allocated to the customer network. In addition, they both use overlays on the public Internet to propagate identifier-to-locator reachability information. LISP+ALT uses tunnels to interconnect LISP+ALT routers that run border gateway protocol (BGP) over these tunnels to propagate the EID-to-locator reachability information. In particular, LISP+ALT routers receive unaggregated IP prefixes by means of exterior BGP (eBGP) links to authoritative ETRs, or by static configuration [13]. These prefixes are aggregated by the ALT topology. In order to make the ALT topology scalable, LISP+ALT requires that identifiers are allocated in an aggregatable way so that the ALT architecture resembles a tree structure hierarchy with the aggregation at merge points in the tree. Unlike LISP+ALT, LISP-DHT organizes all ETRs into a Chord ring [43]. LISP-DHT uses the highest EID in an announced EID prefix as ChordID and lets domains announcing more than one prefix enter the ring more than one instance, each having the ChordID associated to their respective prefix. This assures that a node will be the one managing the prefixes it announces.

Figure 4 compares the resolution delays of LISP+ALT and LISP-DHT with the RTT. From this figure, it is apparent that the resolution delays of LISP+ALT and LISP-DHT are significantly longer than the RTT. Notice that IP addresses in the current Internet are 32-bit long and the simulation results presented in [44] assumes that the ALT topology has only three hierarchies. However, EIDs will be longer in the future. Therefore, as the Internet evolves, the number of hierarchies of the ALT topology increases, which leads to the increase of the resolution delay since a map-request has to pass through more logical hops in the ALT topology. Similarly, as predicted in many publications such as [21], the number of prefixes in the Internet will increase rapidly in the future. This implies that the number of nodes that a Chord ring in LISP-DHT maintains increases rapidly. This in turn leads to longer resolution delay since a map-request will pass through more hops in the Chord ring. In summary, the resolution delay in both LISP+ALT and LISP-DHT increases as the Internet evolves. By contrast, the diameter of the Internet decreases with the Internet evolution. For example, it is reported in [45] that the AS-diameter of the Internet decreased slightly from nearly 4.8 to 4.6 AS hops from November 1997 to January 2000. Similarly, it is reported in [46] that the average number of hops between two end hosts in the Internet decreased from nearly 18 to 13 hops from July 1997 to June 2004. This implies that the resolution delay in SSM is expected to decrease as the Internet evolves.
Fig. 4. Comparison of resolution delay (from [44]).

Fig. 5. Illustration for the effect of SSM on traffic flows.

E. Little Effect on Traffic Flows

The SSM has only little effect on traffic flows. When an ITR receives a packet destined to an EID but the cache misses, the ITR sends the packet to the EID’s subscribed MSP, which then forwards the packet to its destination. This guarantees that a packet will not be dropped due to the cache miss. Even if a cache misses, a packet destined to an EID is directly sent from its ITR to the EID’s subscribed MSP through the routing system, without traversing any overlay topology. With SSM, the delay from the packet’s ITR to its ETR via the EID’s resolver is very close to the delay from the packet’s ITR to its ETR. We illustrate this using the example shown in Figure 5, assuming that an EID denoted by $EID_{dst}$ attaches to its subscribed MSP $PN_2$. As shown, the solid bold lines represent the path directly from a packet’s ITR $TR_1$ to its ETR $TR_2$, and the dashed lines represent the path via the EID’s resolver. Since the resolver and the ETR are in the same PN, the delays of the two paths are very close.

In order to further investigate the effect of this extra delay on the effect of traffic flows, we conduct simulations using OMNeT++ [47] (run on a laptop with 2.4 GHz CPU and 2 GB memory) to evaluate the congestion window size of the Transmission Control Protocol (TCP). In particular, we assume that the direct path from an ITR to an ETR has a delay of 100 ms, and compare the case without mapping (i.e., the first packet of a TCP flow is directly sent from the ITR to the ETR) with three other cases described below.

Case 1: The ITR sends the first few packets of a TCP flow to the ETR through a mapping service. The delay from the ITR to the ETR via the mapping service is 10 ms longer than the direct path from the ITR to the ETR. In addition, the map-resolution delay is assumed to be 200 ms. We use this case to represent SSM since, as described previously, the delay of the path from the ITR to the ETR via the mapping service in SSM is very close to the delay of the direct path from the ITR to the ETR and the map-resolution delay in SSM is upper bounded by the RTT from the ITR to the ETR.
Case 2: This case is similar to Case 1 but the delay from the ITR to the ETR via the mapping service is 100 ms longer than the delay of the direct path from the ITR to the ETR.

Case 3: This case is also similar to Case 1 but the delay from the ITR to the ETR via the mapping service is 200 ms longer than the delay of the direct path from the ITR to the ETR. In addition, the map-resolution delay is increased to 400 ms.

Figure 6 shows the congestion window of a TCP flow used to transport a small file of 10 kilobytes under the four cases. In our simulation, the path maximum transmission unit (MTU) is assumed to be 1500 bytes and the maximum segment size (MSS) is assumed to be 1024 bytes. From the figure, it is seen that the congestion window size of the case without mapping and that of Case 1 are very close. This implies that SSM has very little effect on TCP flows. On the other hand, the congestion window sizes of the other cases are significantly lower than the congestion window size in the case without mapping, since the long delay of the path via a mapping service postpones the increase of the congestion window size.

VI. FEASIBILITY ANALYSIS

Above we have described the design details of SSM and its benefits. We now analyze SSM’s feasibility from three aspects: the bandwidth overhead used to carry MSPs and signatures, the processing overhead for ITRs to verify signatures, and the storage requirement used for storing EID-to-locator mappings at ITRs.

A. Bandwidth Overhead

A possible drawback of SSM is that it may consume more network bandwidth in order to transmit an EID’s subscribed MSP and the corresponding signature. There are three cases to transmit an EID’s subscribed MSP and the corresponding signature.

Case 1: When a cache miss for an EID occurs at an ITR, the ITR sends out a map-request that includes the EID’s subscribed MSP and signature.
Case 2: When an end host initiates a connection toward an EID, it sends the EID, the EID’s subscribed MSP and the corresponding signature to its ITR.

Case 3: When an end host attaches to a new TR, it sends its EID, the subscribed MSP and the corresponding signature to the new TR, which registers a new EID-to-locator mapping for the EID.

In order to analyze the bandwidth overhead used for carrying MSPs and signatures, without loss of generality, we assume that the length of an MSP is $L_{MSP}$ bits and that of a signature is $L_s$ bits. Figure 7 shows the different values of an MSP and a signature and their combinations considered in this paper. For MSPs, we consider three cases: the AS number in the current Internet is 32 bits, some recent publications such as [21] suggest that inter-domain routing in a future Internet uses 128-bit AS numbers. For signatures, some traditional signatures are 256 bits and it is suggested that in the future they should be 3072 bits. We choose three combinations, namely A, B, and C (as marked in Figure 7) that correspond to the minimum, the median, and the maximum lengths of all combinations, respectively.

Let $N_m$ denote the number of cache misses per minute at an ITR, the bandwidth overhead in the first case is defined as:

$$BO_1 = \frac{N_m \times (L_{MSP} + L_s)}{60 \times B_o \times 10^9} \times 100\%$$

(1)

where $B_o$ is the bandwidth (in Gigabits per second, Gbps) of the ITR’s upload link toward the Internet core.

In order to estimate $N_m$, we collected traces of the traffic from and to the campus network at Beijing Jiaotong University (BJTU) during the week of October 12, 2008. The BJTU campus network counts about 40,000 users and there are over 13,000 active users/day. The network is connected to the Internet through a border router that has a 1 Gbps link toward ChinaNet and another 1 Gbps link toward CERNET. We analyze the traffic traces by assuming that our border router is an ITR and emulate the number of IP addresses (considered as EID-to-locator mappings) cached at our border router. In particular, if an IP address outside our campus network is first used, it is considered as a new EID-to-locator mapping and we add the IP address as a new item in the cache. In addition, we set a timer for the item. If the IP address is used by any flow before the timer exceeds the cache timeout, the timer for the item is reset to zero. On the other hand, if the IP address is not used by any flow until the timer exceeds the cache timeout, the item is removed from the cache. We show the number of cache misses per minute in Figure 8, which is from [24]. As shown, the number of cache misses per minute depends on the cache timeout. When the cache timeout is longer, the number of cache misses per minute is smaller.

Having obtained the number of cache misses, we are able to compute the bandwidth overhead in the first case by using Equation 1. Figure 9 plots the bandwidth overhead in the first case when the cache timeout is three minutes. From Figure 9, we observe that, for all three cases A, B, and C, the maximal bandwidth overhead is less than 0.1% when the cache timeout is three minutes, even if $L_{MSP} = 128$ bits and $L_s = 3072$ bits. Furthermore, the average bandwidth overheads are less than 0.02%, 0.01% and 0.005% for cases C, B, and A, respectively. If we
further increase the cache timeout to 30 minutes, the bandwidth overhead would be further reduced. As shown in Figure 10, the maximal bandwidth overhead for all cases A, B, and C is less than 0.045%, and the average bandwidth overheads for the three cases are even smaller. This implies that the bandwidth overhead is very small and negligible.

In order to analyze the bandwidth overhead in the second case, we count the number of new outgoing flows per second at the BJTU campus network. Let \( N_f \) be the number of new outgoing flows, the bandwidth overhead in the second case is then defined as:

\[
O_2 = \frac{N_f \times (L_{MSP} + L_s)}{B_i \times 10^9} \times 100\%
\]  

(2)

where \( B_i \) is the total bandwidth of the links in a customer network that connect to the border router of the customer network. For the BJTU campus network, this bandwidth is 2 Gbps, i.e., \( B_i = 2 \).

Figure 11 shows the bandwidth overhead in the second case at the BJTU campus network. From this figure, we clearly observe that the bandwidth overhead is fairly small, for all three cases A, B, and C. For example, the maximal bandwidth overhead is less than 1% even when \( L_{MSP} = 128 \) bits and \( L_s = 3072 \) bits (i.e., case C). Furthermore, the average bandwidth overhead is less than 0.2%, 0.08%, and 0.01%, for cases C, B, and A, respectively. We also consider the case that every existing flow also sends an MSP and a signature per second. Figure 12 illustrates the resulting bandwidth overhead. As shown in Figure 12, we observe that the maximal bandwidth overhead is less than 0.25% even for case C. If we take the average, the bandwidth overhead is further reduced.
The bandwidth overhead in the third case depends on the number of new mobile hosts that attach to a TR per second. If we assume that this number is 1000 and the incoming and outgoing links to a TR have a bandwidth of 1 Gbps, the bandwidth overhead is only 0.0288%, 0.1088%, and 0.32%, for cases A, B, and C, respectively.

To conclude, these results show that the bandwidth overhead used for carrying MSPs and corresponding signatures is very small and negligible. Furthermore, as the Internet evolves, bandwidth becomes cheaper and cheaper, so the effect of bandwidth overhead further diminishes.

B. Processing Overhead

Another important aspect that may limit the performance of SSM is the processing overhead since ITRs need to create/verify signatures. In SSM, there are two cases for an ITR to verify signatures.

First, when an end host A initiates a connection to another end host B, it is required to send host B’s EID, the EID’s subscribed MSP and the corresponding signature to its ITR if the ITR cannot locally find an EID-to-locator mapping for the EID. As a result, the number of signatures required to be verified by the ITR is equal to the number of cache misses at the ITR. From Figure 8, we observe that, for an ITR serving a campus network of about 40,000 users, the number of cache misses is less than 40,000 per minute (i.e., less than 666 per second) even if the cache timeout is as short as three minutes. Therefore, the number of signatures needed to be verified in this case is about 666 per second.

Second, when an ITR detects the attachment of an end host, it is required to verify the signature for the EID of the end host before registering an EID-to-locator mapping for the EID. In practice, however, the number of new
end hosts attached to an ITR should be very small. For our analysis, however, we overestimate this number and assume that there are 10,000 new end hosts per second. As a result, an ITR needs to verify 10,000 signatures in this case.

By counting these two numbers, an ITR needs to verify about 10,666 signatures per second. Notice that, in SSM, an ITR only needs to do a hash operation and a comparison operation in order to verify a signature. We have tested this processing cost on a personal computer (PC) running Windows XP with Intel Core2 Duo 2.4 GHz CPU and 2 GB memory. Our results show that it spends 423 ns in verifying a signature. This means a PC is able to verify more than $2.38 \times 10^6$ signatures per second, which is significantly higher than 10666.

In addition, when a cache miss occurs, an ITR needs to send out a map-request message that is signed by the ITR. As a result, every time a cache miss occurs, the ITR needs to create a signature. As stated above, for an ITR that serves a campus network of about 40,000 users, the cache misses per second is about 666 even if the cache timeout is as small as 3 minutes. Thus the ITR is required to create about 666 signatures per second. Using the NTL library in a PC running Windows XP with Intel Core2 Duo 2.4 GHz CPU and 2 GB memory, we test the processing cost in creating a signature. Our results show that it only spends 11.6 microseconds ($\mu$s) in creating a signature if the approach in [39] is used. This implies that a PC is able to create more than 86,200 signatures per second, which is also significantly larger than 666.

Notice that an MSP needs to create signatures for the EIDs subscribed to it. However, the creation of signatures can be done off-line and an MSP only needs to update the signature for an EID once several days. Therefore, the processing overhead for creating signatures at MSPs is also acceptable.

Another issue is that public file-sharing systems need to store the EIDs’ subscribed MSPs and the corresponding signatures. As stated previously, however, this does not increase the number of requests sent into such systems, although this evidently increases the storage requirement. As storage devices become cheaper and larger storage space is available, we anticipate that the increased storage bandwidth requirement is well within the capability of these file-sharing systems.

In short, the processing overhead caused by creating/verifying signatures in SSM is small and well-within the capability of a modern router.

C. Storage Requirement

In order to locally cache EID-to-locator mappings, ITRs need some storage space to store recently used EID-to-locator mappings. In [24], we have reported that the required storage space is well within the storage capability of a modern router. We refer interested readers to [24] for details.

VII. Conclusion

In this paper, we have proposed an approach to map identifiers onto locators in networks with identifier/locator separation. The most significant benefit of the proposed approach is that it offers users of identifiers to choose their preferred MSP, and to change their preferred MSPs without changing their identifiers. Furthermore, we have shown that the proposed approach is more scalable and secured and has lower resolution delay than many existing approaches. We have also analyzed the efficiency of the proposed approach in terms of bandwidth and processing overhead. Our results show that the bandwidth overhead used for transmitting MSPs and signatures is negligible, and that the processing overhead is not excessive. Because of its salient features and efficiency, we believe that this approach is promising.

REFERENCES

Hongbin Luo received the B.S. degree from Beijing University of Aeronautics and Astronautics in July 1999, the M.S. degree (with honors) and the Ph.D. degree in Communications and Information Science from University of Electronic Science and Technology of China in June 2004 and March 2007, respectively. From July 2002 to March 2007, he is a research associate at the Key Lab of Broadband Optical Fiber Transmission and Communication Networks, UESTC. In June 2007, he joined Beijing Jiaotong University, where he is an associate professor. He is the first author of more than 30 refereed papers published in prestigious international journals and conferences including IEEE/ACM Transactions on Networking, IEEE Journal on Selected Areas in Communications. His research interests are in the areas of communication networks including optical networks, wireless networks, and Internet technology.

Dr. Luo is a member of IEEE.

Hongke Zhang received his M.S. and Ph.D. degrees in Electrical and Communication Systems from the University of Electronic Science and Technology of China (formerly known as Chengdu Institute of Radio Engineering) in 1988 and 1992, respectively. From Sep. 1992 to June 1994, he was a post-doc research associate at Beijing Jiaotong University (formerly known as Northern Jiaotong University). In July 1994, he jointed Beijing Jiaotong University, where he is a professor. He has published more than 100 research papers in the areas of communications, computer networks and information theory. He is the author of eight books written in Chinese and the holder of more than thirty patents.

Dr. Zhang received the Zan Tianyou Science and Technology Improvement Award in 2001, the Mao Yisheng Science and Technology Improvement Award in 2003, the First Class Science and Technology Improvement Award of the Beijing government in 2005, and other various awards. He is now the Chief Scientist of a National Basic Research Program (“973 Program”).

Moshe Zukerman received the B.Sc. degree in industrial engineering and management and the M.Sc. degree in operations research from the Technion, Israel Institute of Technology, Haifa, Israel, and the Ph.D. degree in engineering from University of California, Los Angeles, in 1985. He was an independent Consultant with the IRI Corporation and a Postdoctoral Fellow with the University of California, Los Angeles, in 1985-1986. In 1986-1997, he was with the Telstra Research Laboratories (TRL). He also taught and supervised graduate students at Monash University in 1990-2001. During 1997-2008, he was with The University of Melbourne, Victoria, Australia. In 2008 he joined City University of Hong Kong as a Chair Professor of Information Engineering, and a team leader. He has over 200 publications in scientific journals and conference proceedings. Prof. Zukerman has served on various editorial boards such as IEEE/ACM Transactions on Networking, IEEE Journal on Selected Areas in Communications, IEEE Communications Magazine, Computer Networks and Computer Communications. He is a Fellow of the IEEE.