Secure Name Resolution for Identifier-to-Locator Mappings in the Global Internet

Xiruo Liu, Wade Trappe, Yanyong Zhang
WINLAB, Electrical and Computer Engineering Department
Rutgers University, Email: {sissi, trappe, yyzhang}@winlab.rutgers.edu

***Abstract*—A recent trend in clean-slate network design has been to separate the role of identifiers from network locators. An essential component to such a separation is the ability to resolve names into network addresses. One challenge facing name resolution is securing the name resolution service. This paper examines the security of a clean-slate name resolution service suitable for mobile networking. We begin with a high-level threat analysis, and identify several types of attacks that may be used against name resolution services. We then present secure protocols that together form a secure global name resolution service. Specifically, we present a secure update protocol that allows users to update their network addresses as they migrate and that includes several checkpoints that prevents spoofing, collusion, stale identifiers and false identifier announcements. Since the primary function behind a name resolution service is to respond to address-lookup queries, we also present a secure query protocol. Finally, we address the security risks associated with IP Holes that can arise in a global name resolution service.**

I. Introduction

The separation of identifiers from network locators is an important aspect underlying many future Internet architectures. By separating identifiers from the underlying addresses needed to locate and route to a network object, identifier-based networking [1]–[4] supports simplified session management, multi-homing and device/user mobility. The separation between these two attributes makes it possible to avoid implicit or explicit binding of sources and destinations to the network's actual topology. In particular, the use of identifiers allows programmers to route to mobile devices based on an identifier like the International Mobile Subscriber Identity (IMSI) number rather than on an IP address, and this simple architectural change leads to simpler networking primitives that allows for easier development when there is a high level of dynamism in the underlying network, as occurs in mobile services.

One challenge in designing an identifier-based protocol stack is the task of translating an identifier into a network address. When a user or application contacts the networking subsystem with an identifier that names a communicating entity (e.g. an intended recipient), this identifier must be translated into a network address that is then provided to the user/application and which could subsequently be used by various network services, such as routing. Recently, as part of the MobilityFirst [5] clean-slate architecture, whose main objective is to support large-scale mobility through an identifier-based network architecture, a two-tier approach to name resolution was outlined (as shown in Figure 1).

Each network entity in MobilityFirst has at least three attributes: a user-level descriptor (i.e. a human readable name, such as Jim Smith's laptop), a network-level identifier (called a globally unique identifier, or GUID for short) and a *route*-able topological address (referred to as a network address, or NA for short). In MobilityFirst's two-tier approach to name resolution, a Name Certificate & Resolution Service (NCRS) is used to translate user-level descriptors into GUIDs, while a Global Name Resolution Service (GNRS) provides the mapping between GUIDs and the corresponding NAs. Recently, as part of the MobilityFirst project, the design of a fast global name resolution service was presented [6]. The work in [6], however, focused on the balance between scalability and latency, with the goal of mitigating update/query latency, while also ensuring system consistency and supporting incremental deployment. The benefits of such a resolution service, however, are undermined when subjected to security threats, and consequently any clean-slate name resolution service should have security in mind as one of its founding principles. In this paper, we revisit the name resolution service presented in [6], and examine the security risks that name resolution services might face in an adversarial setting. We begin the paper in Section II by presenting an overview of the two-tier approach to name resolution employed in MobilityFirst. We then proceed with a high-level threat analysis in Section III, and then use this threat analysis to motivate the design of several secure protocols in Section IV that ensure the secure operation of a global name resolution service. Throughout the paper, our presentation is IP-centric to facilitate seamless transition from the current Internet architecture, and we provide a detailed walk-through behind each step, explaining why specific details were included with an emphasis on addressing potential security exploits. Finally, we conclude the paper in Section V.

II. Overview of Two-tier Name Resolution

The motivation behind the separation of human readable names from GUIDs in the MobilityFirst project is that the NCRS and GNRS operate at different time-scales: the translation between a human-readable name and a GUID is not likely to change as rapidly as the translation between a GUID and a corresponding network attachment point (particularly in highly mobile settings). Further, MobilityFirst employs a flat, location-independent identifier space where public keys are used as GUIDs (as in AIP [7], HIP [3], and ROFL [8]), and

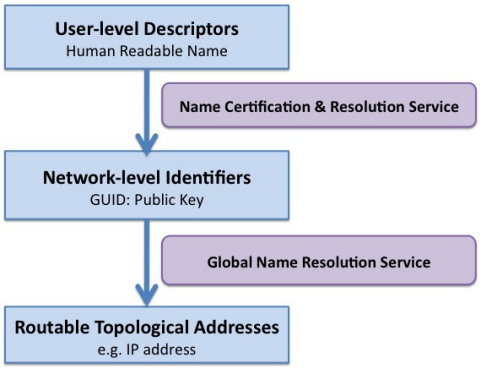

Fig. 1. Overview of the two-tier name resolution involved in MobilityFirst.

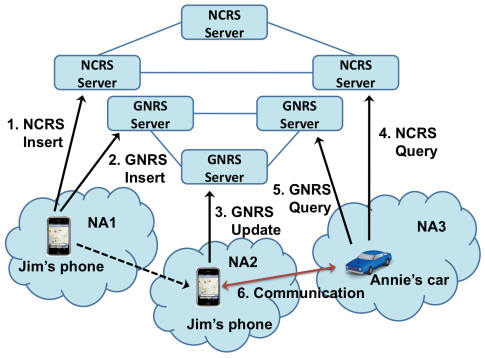

Fig. 2. The MobilityFirst approach to name resolution involves the Name Certificate & Resolution Service (NCRS) and the Global Name Resolution Service (GNRS). The NCRS serves the role of a certificate registration service that associates human-readable names to GUIDs, while the GNRS translates GUIDs into network addresses

each end host, such as laptops, mobile phones, servers and virtual machines, can have a GUID. Each GUID can associate with one or more network addresses (NAs) that it attaches to. As an example, the NAs of a multi-homed laptop might include the NA of its 3G service provider as well as the NA of its WiFi network. The roles of the NCRS and GNRS are depicted in Figure 2.

The NCRS is involved in two main functions: (1) it provides a translation between a human-readable name and the corresponding GUID; and (2) it acts as a certificate authority in distributing the GUIDs, which are themselves public keys and hence must be appropriately packaged for use. In short, the NCRS must provide services analogous to those provided by a Public Key Infrastructure (PKI). In MobilityFirst, users may either generate their own public keys (for use as GUIDs) and submit these to the NCRS for registration, or may contact the NCRS to acquire public keys (for use as GUIDs). Consequently, the NCRS publishes approved cryptographic suites, object categories and object description formats, as well as allows users to self-certify themselves. Following standard certificate formats such as those discussed in [9], [10], users may generate their own certificates and then submit them to the NCRS, who engages in a challenge-response protocol to verify that the submitter possesses the corresponding private key. In the context of MobilityFirst, the NCRS certificates follow the general format of X.509 certificates, and include the following fields: version, serial number, signature algorithm, validity, public key, object category code, object description (human readable keywords) and appropriate signatures. The object description field is useful in the context of global networking, as it allows for the description of network roles/functions (such as whether a device is a border gateway router for a particular Autonomous System (AS)). In this paper, we make a generic assumption that a network entity's locator consists of an AS number as its network address, as well as a local address within that network/AS. Hence we use AS number and NA interchangeably in this paper.

Since the GUID's correspondence to public keys, we propose the use of elliptic curve cryptography (ECC) [11] and note that the curve P-256 of FIPS PUB 186-3 DSS is appropriate for our purposes [12]. In particular, when choosing a cryptographic suite, both security strength and efficiency need to be considered. According to the National Security Agency (NSA), elliptic curve public key cryptography using the 256-bit prime modulus elliptic curve, as specified in FIPS 186-3, and the SHA-256 hash function provide adequate protection for classified information up to the SECRET level [13]. Moreover, ECC P-256 has cryptanalytic security strength comparable to RSA with a key size of 3072 bits, yet ECC P-256 requires a key size of only 256 bits, which translates into lower computational costs and higher efficiency [14]. Lastly, it is widely accepted that ECC is a strong and more efficient approach to public key cryptography that is suitable for resource-constrained environments, such as the mobile networks that we are concerned in this paper. Finally, we note the NCRS performs conventional tasks associated with a certificate authority, such as the insertion, update and revocation of the GUID/digital certificates. Since these mechanisms are a straight-forward application of well-known PKI techniques, we will not focus on their specification in the remainder of the paper.

We now provide a high-level description of the functions performed by the GNRS. To perform the mapping service for a given GUID, the GNRS applies $K (K > 1)$ hashing functions onto it to produce a list of K locators, which are IP addresses in today's Internet, and stores the GUID→NA mapping in the ASs that announce those network addresses. By doing so, the GNRS spreads the GUID→NA mappings amongst ASs, such that an AS will host mappings of other ASs, as well as have its mappings hosted by others. This design leverages the routing infrastructure to reach the hosting AS in a single overlay hop; it does not require a home agent, unlike mobile IP and existing cellular networks. There are three types of events in GNRS: insert, update and query. When a user A (with a GUID) joins a network for the first time, it sends an insert message to the GNRS (a distributed system) reporting mapping information of its GUID and network address. When A moves to another network, it sends an update message to the GNRS to report the new mapping. If another user B wants to communicate with A, it sends a query message to the GNRS asking for A's

network address before their communication, and the GNRS will return A's mapping to B [6].

III. GNRS Security Concerns

Although the design of the NCRS and GNRS are effective in managing translations between different identifiers, these two separate systems may themselves be the target for attacks and exploits. Securing the design of a certificate authority, which is essentially the role played by the NCRS, involves well-known techniques and we consequently refer the reader to [15]–[17] for a survey of such mechanisms. The GNRS, however, is a new network service that manages very dynamic GUID-to-NA mappings. In particular, the lack of hierarchies in GNRS implies that techniques used to secure the inherently hierarchical Domain Name System (DNS) [18] cannot be leveraged to secure GNRS. GNRS is fundamentally distributed and tied to the functioning of the underlying network, and hence warrants its own security investigation.

Identifying and understanding potential security threats is a necessary starting point for designing a secure network architecture. We now present a security analysis for GNRS, and highlight a collection of threats and risks that may be faced. The threats that GNRS might face include:

- Unauthorized Access: an intruder gains access or gathers information from a resource it is not entitled to. As a consequence, an adversary may examine, remove or even modify confidential information.
- Masquerading: an intruder is able to mimic an authorized user or network process. As a result, the intruder may forge signatures, or impersonate a source address.
- Information Modification: unauthorized or malicious information is injected into the network or its resources. This threat can cause service failures, false information transmissions or network operations, etc.
- Message Manipulation: an adversary manipulates the message exchange process between network entities. Such manipulation may involve replay, rerouting, misrouting and deletion of messages.
- False Traffic Insertion: Insertion of false traffic into the network can place an additional burden upon the network and consume precious network resources. The consequence might be an increase in delay and performance degradation for network services and applications.

Consequently, major risks associated with these security threats are identified as below:

- Illegitimate Resource Consumption: an unauthorized user gains access to resources it is not allowed to and consumes networks resources. Threats such as unauthorized access, masquerading, information modification, message manipulation can lead to this risk.
- Pilfering of Service: an adversary is able to use service that it does not have the privilege of usage. This risk might be a result of unauthorized access, masquerading, information modification or message manipulation.
- Denial of Service: this risk makes network service or resource unavailable to its intended users. This is a serious risk that may result from unauthorized access, masquerading or message manipulation.

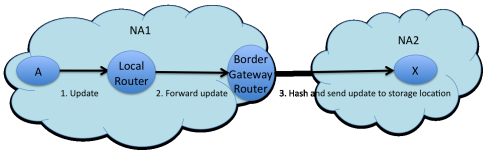

Fig. 3. GNRS Update Scheme.

As summarized earlier, GNRS has three basic operations: insert, update and query. Among them, GNRS insert and update are quite similar, with insert being a first-time update. Therefore, in examining security, our attention will be restricted to the update and query operations in the remainder of this paper.

The basic process involved in GNRS update is shown in Figure 3. Suppose user A belongs to network NA1 and needs to update its mapping $(GUID_A, NA1)$. A sends an update message to its local router who will then forward the message to the border gateway router. The border gateway router hashes A's $GUID_A$ to get the mapping's storage location X in network NA2, which is an IP address, and then sends the mapping contained in the update message to X. X stores the mapping and waits for other users in the (global) network to query. In GNRS's design, to improve performance and efficiency, the border gateway router performs K different predefined hash functions to find K different mapping storage locations. As a result, there are K mapping replicas in the global network. When another user B queries A's mapping, it can receive the mapping from the nearest replica and therefore lower the query delay. A benefit of distributing multiple mapping replicas is that it prevents single-point-of-failure risks, and thus enhances the reliability and robustness of GNRS.

Unfortunately, if the user sends the mapping message $(GUID, NA)$ to the local router and follows the simple procedure just outlined, the update procedure cannot be guaranteed to proceed correctly. In particular, attackers might come from anywhere in the network and attack cooperatively. We now outline five possible attacks against GNRS.

1. The GUID spoofing attack is a masquerading threat, where a malicious user A claims another user B's GUID and attempts to associate it with A's own network address NA_A, by announcing the mapping $(GUID_B, NA_A)$. The consequence of this attack is a denial of service as it can cause traffic directed for B to be directed to A's network address.

2. The stale mapping attack is a message manipulation attack involving a malicious GNRS server. In this attack, if a device moves and issues an update, the malicious GNRS server can purposely ignore the update and claim it still has the most recent mapping. Perhaps worse, a GNRS server can selectively choose which (possibly stale) mapping to give out during queries. The result is a denial of service.

3. The third potential attack, false announcement attack, is an information modification attack that results in illegitimate resource consumption. User A, which is in network NA1,

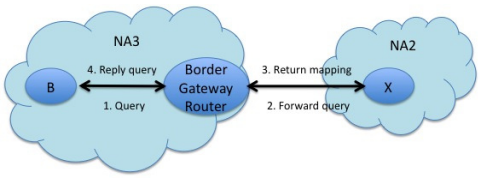

Fig. 4. GNRS Query Scheme.

claims its $GUID_A$ binds to a different network address, $(GUID_A, NA2)$. Thus A can direct its traffic to network NA2, which causes NA2 network resources to be consumed. To protect against the third type of attack, one can let the network, such as the local router and/or border gateway router, sign update messages to the GNRS after it verifies whether the user belongs to it. However, this is not enough because it cannot prevent the fourth attack, the collusion attack.

4. The collusion attack is an example of an information modification attack in which a malicious user, its network and the location where the mapping is stored collude with each other. The objective behind the malicious collusion is to allow for a fake mapping involving a false network address to pass the verification and become stored in the storage place.

In the GNRS query scheme, assume user B is in network NA3 and wants to query A's mapping. B first sends out the query request message to its border gateway router. The router runs the K predefined hash functions on A's GUID as used in A's update process to find the nearest replica storage location X (found by examining the K hash results). The border gateway router forwards the query request message to X. X returns the mapping to B's border gateway router, who will then reply to B's query with the mapping information. Figure 4 shows the query scheme. Interestingly, we note that by securing the update scheme, GNRS query itself becomes less-open to exploitation since all of the mappings would have been correctly verified prior to insertion into GNRS.

5. The final attack we mention is a family of information manipulation attacks that we collectively label as BGP churn exploits. Problems may also arise due to network topology changes, such as BGP churn [19]. Although hashing is used to find the mapping storage location in GNRS, the hash result might fall into an IP hole (i.e. an IP address that is not announced by any AS). For this reason, the IP hole protocol was proposed to solve this problem in [6]. The scheme involves rehashing the hash result if the result falls into IP hole until a valid IP address is found. If it still falls into an IP hole after M attempts at rehashing, then the scheme chooses a deputy AS who announces the IP address that has the minimum IP distance to the current hash value. In this scheme, BGP churn, such as announcing a new IP prefix or withdrawing an IP prefix, might cause security weaknesses. Take withdrawing the IP prefix for example, and suppose network NA2 wants to withdraw its IP prefix. This will make the IP prefix of NA2 become an IP hole and the mappings stored in NA2 become orphan mappings. Therefore, before withdrawing IP prefix, NA2 runs IP hole protocol to find locations to insert orphan mappings and moves the mappings there. Without a secure scheme, a malicious user can insert many fake orphan mappings to a target network to exhaust that network's storage resources, while also causing the GNRS to report false mappings to queries. We will specifically address this issue in section IV-C.

IV. Secure GNRS Protocol

We now explore a secure GNRS service that is targeted at addressing the security concerns outlined in Section III. As a starting point, we assume that the NCRS plays the role of a PKI and a clock synchronization system is available. In our design, the public keys used are equivalent to the user's identifier the GUID, and thus we use these terms interchangeably. A 256-bit long GUID is sufficient to keep birthday attacks infeasible. If the desired probability of random collision is kept as low as 10^{-6}, the pool of available GUID will be about 4.8×10^{35} [9], which is more than large enough for GUID assignment for the foreseeable future.

A. Securing GNRS Update

The objective behind GNRS update is to allow users to update their network address as they migrate across the global network. The main network components involved in the GNRS update scheme are the user, the local router, the border gateway router and the DHCP server. Here, the DHCP server provides verification of a user's IP address [20]. The update protocol details are shown in Figure 5. Within network NA1, A is a user who wants to inform GNRS of its new mapping $(GUID_A, NA1)$, L is A's local router, G is the border gateway router for NA1, D is the DHCP server with which G can verify A's network address, and X is the storage location in network NA2 responsible for keeping track of the mapping. There are eight messages exchanged during the update process:

1) A $\rightarrow$ L: $\{GUID_A, [N_a, GUID_L, (GUID_A, NA1, T_A, E_A)_{A^{-1}}]_L\}$
   There are two timestamps involved in the message exchange: the mapping generation time T_A (corresponds to when the new mapping was instantiated) and the expiration time E_A (which corresponds to when this new mapping will expire). These timestamps are included with the GUID and network address, and necessitates that A must update or renew its mapping before expiration. This protects against stale mapping attacks, which were mentioned in Section III. Since the mapping has an expiration time, the freshness of the mapping is guaranteed: users who query A's mapping can check these timestamps to ensure the mapping is not stale. The use of timestamps for freshness, however, introduces a tradeoff between security and efficiency. If E_A is too large, then it is possible that a prior mapping will still be valid after user A moves to another network and updates its new mapping with a new generation time and expiration time. This is problematic in the case of a malicious mapping storage location X. An untrustworthy X may give users an old mapping (that has not expired)

instead of the up-to-date mapping. On the other hand, if E_A is too small, then the likelihood of a successful attack is low, the impact minimal, but the overhead will be large because A must renew its mapping frequently. In this step, $(GUID_A, NA1, T_A, E_A)$ is signed by A's private key A^{-1} to protect against a GUID spoofing attack. An adversary cannot claim to be A and launch a GUID spoofing attack because only A knows its private key. Other users can verify the signature since they know A's public key $GUID_A$. Also a nonce N_a is added in the update message to prevent replays. $GUID_L$ is added for receiver verification, and the message is encrypted by L's public key so that only L can read the message. The update message also contains $GUID_A$ so that the receiver L can verify the sender.

2) L $\rightarrow$ G: $\{GUID_L, [N_l, GUID_A, GUID_G, (GUID_A, NA1, T_A, E_A)_{A^{-1}}]_{L^{-1}}\}_G$
L first verifies whether the mapping $(GUID_A, NA1)$ is correct. This is the first checkpoint to protect against the false announcement attack mentioned in Section III. If the mapping is correct, then L forwards A's mapping to the border gateway router G for NA1. Similar to message 1, a nonce N_l is added and message 2 is signed by L's private key and subsequently encrypted by receiver G's public key.
3) G $\rightarrow$ D: $\{GUID_G, [N_g, GUID_A, (GUID_A, NA1, T_A, E_A)_{A^{-1}}]_D\}$
The border gateway router G is the second checkpoint to prevent the false announcement attack. In particular, this step is intended to handle cases where A and L collude with each other. To do this, G sends message 3 to the DHCP server D to verify A's network address.
4) D $\rightarrow$ G: $\{N_g + 1, (GUID_A, NA1, T'_A, E'_A)_{D^{-1}}\}_G$
After receiving G's query, D replies with A's current network address and its valid time period (T'_A, E'_A). Also, $N_g + 1$ is included and serves as the ACK corresponding to the nonce N_g.
5) G $\rightarrow$ L: $\{N_l + 1\}_{G^{-1}}$
If the reply from D matches A's update, G will return an ACK $(N_l + 1)$ to L implying the acceptance of A's update, which has been forwarded by L.
6) L $\rightarrow$ A: $\{N_a + 1\}_{L^{-1}}$
L returns $(N_a + 1)$ to A, informing A of the successful update to A's mapping.
7) G $\rightarrow$ X: $\{GUID_G, \{[GUID_A, N'_g, (GUID_A, NA1, T_A, E_A)_{A^{-1},G^{-1}}]_{G^{-1}}\}_x\}$
G hashes A's GUID by using K predefined hash functions to obtain the IP addresses of K different storage locations responsible for storing A's mapping. Before sending A's mapping to any storage location X, both G and A sign the mapping $(GUID_A, NA1, T_A, E_A)$ so that other users can verify whether this mapping has been checked by network NA1. This prevents a collusion involving A, L, G, and X working together to create a fake mapping, such as $(GUID_A, NA3, T_A, E_A)_{A^{-1}}$, which would have otherwise passed the verification and been stored in X. Since G signs the mapping, and since a border gateway router is only responsible for its own domain, any user who would have queried $GUID_A$ can check whether G is actually NA3's border gateway router by querying the NCRS, which provides the user a description of G's responsibilities.
8) X $\rightarrow$ G: $\{N'_g + 1\}_{X^{-1}}$
X returns an ACK to G indicating the mapping is accepted and the update process is finished. Finally, A's mapping is stored at X as $\{GUID_G, (GUID_A, NA1, T_A, E_A)_{A^{-1},G^{-1}}\}$.

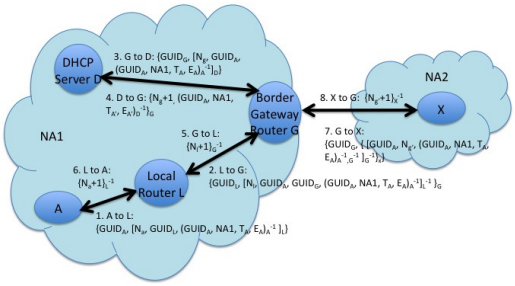

Fig. 5. GNRS Update Protocol.

B. Securing GNRS Query

GNRS query provides a means by which a user can acquire the network address corresponding to a known GUID. In the GNRS query, the involved network components are the user making the query, the border gateway router, and the mapping storage location. The query protocol is shown in Figure 6. To start, assume user B, who is in network NA3, wants to query the GNRS to find A's mapping. Suppose that S is the border gateway router for NA3, NA2 is mapping storage location that is the closet to NA3 among the K replicas identified by the K hash functions, and in NA2 the mapping is stored at X. In our secure GNRS query protocol, there are only four messages exchanged during the query process:

1) B $\rightarrow$ S: $\{GUID_B, [GUID_S, N_b, (GUID_A, GUID_B, T_B)_{B^{-1}}]_S\}$
B sends S the query for A's GUID, $GUID_A$, together with its own identifier $GUID_B$ and timestamp T_B. The request is first signed by B's private key B^{-1}, then the receiver S's GUID is added for verification purposes and a nonce N_b is added to prevent replay attacks. The message is finally encrypted by S's public key.
2) S $\rightarrow$ X: $\{GUID_S, [GUID_X, GUID_B, N_s, (GUID_A, GUID_B, T_B)_{B^{-1},S^{-1}}]_x\}$
S verifies the request and then performs the same K predefined hash functions (as used in the GNRS update process) on $GUID_A$. S chooses the nearest one from the K hash results, which is X in NA2 for this example, and forwards the query request to X.
3) X $\rightarrow$ S: $\{[GUID_X, GUID_S, GUID_B, N_s + 1, GUID_G, (GUID_A, NA1, T_A, E_A)_{A^{-1},G^{-1}}]_{X^{-1}}\}_S$

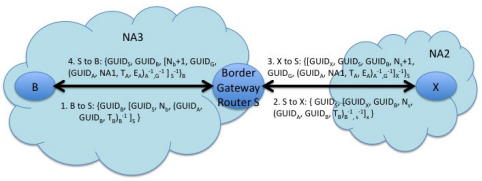

Fig. 6. GNRS Query Protocol.

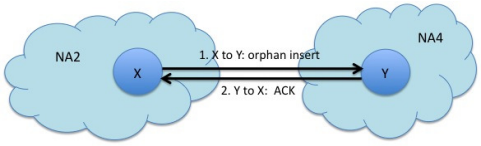

Fig. 7. Orphan Mapping Insert.

X verifies the request and returns A's mapping $\{GUID_G, (GUID_A, NA1, T_A, E_A)_{A^{-1},G^{-1}}\}$ to S.

4) S $\rightarrow$ B: $\{GUID_S, GUID_B, [N_b + 1, GUID_G, (GUID_A, NA1, T_A, E_A)_{A^{-1},G^{-1}}]_{S^{-1}}\}_B$
S forwards A's mapping to B. B can check the mapping by verifying the timestamps and the signatures of A and G. Now the query process is finished.

C. Securing IP Hole Protocol

Hashing is used in the GNRS update and query process to find mapping storage locations. Due to the fragmentation of IP address space, the hashing result could possibly fall into an IP hole, which is not announced by any AS. As a result, in such an event, a valid IP address cannot be found to store the mapping. In this case, the IP hole protocol [6] is performed to find the mapping storage location. Basically, the IP hole protocol involves continuous hashing until a valid mapping storage address is found. The high-level details are summarized below:

1) If the hash result falls into an IP hole, rehash (up to M times) the hashing result until a valid IP address IP_x is found. IP_x is the mapping storage location.
2) If it still falls into an IP hole after M rehash attempts, choose a deputy AS who announces the IP address IP_x that has the minimum IP distance to the current hashing value. Assume A and B are two n-bits IP addresses, the IP distance between A and B is defined as:

$$IP\_distance(A, B) = \sum_{i=0}^{n-1} |A_i - B_i| \times 2^i$$

The IP hole protocol can be used to handle changes in prefixes (analogous to the changes in prefixes associated with BGP churn), which directly influences GNRS update and queries. Prefix change announcements either can be due to an AS withdrawing a previously announced prefix, or due to an AS announcing a new prefix.

1) Orphan Mapping Insert: The Orphan mapping insert step addresses the problem of a prefix being withdrawn by an AS. In Figure 7, user A's GNRS mapping $(GUID_A, NA1)$ is stored at X in NA2, while Y is in network NA4. Suppose NA2 wants to withdraw its IP prefix, which will make its current IP prefix become an IP hole, and hence A's GNRS mapping will become an orphan mapping since others will not be able to reach storage location X associated with this mapping. To solve this problem, before withdrawing its IP prefix, NA2 must run the IP hole protocol and move A's mapping to another storage location. Orphan Mapping Insert, involves:

1) X $\rightarrow$ Y: $\{GUID_X, [R, GUID_Y, (N_X, GUID_G, (GUID_A, NA1, T_A, E_A)_{A^{-1},G^{-1}})_{X^{-1}}]_Y\}$
The IP hole protocol is called to find a place to insert A's mapping. First, X applies the appropriate hash function to $GUID_A$. If this hash result falls into an IP hole or NA2 itself, X will hash the hash result again (following the IP hole protocol) until it finds a valid IP address or a deputy IP_x. For the sake of discussion, let us assume that this new location corresponds to Y, which is located in network NA4, in Figure 7. Suppose the amount of times needed to rehash $GUID_A$ is R (we note $R \leq M$, where M is the maximum number of rehashes allowed in the IP hole protocol). After finding Y, X signs A's mapping $\{GUID_G, (GUID_A, NA1, T_A, E_A)_{A^{-1},G^{-1}}\}$ together with the nonce N_X. Then X includes R and the receiver Y's identifier $GUID_Y$, and sends the message to Y after encrypting with Y's public key. R is added for the convenience of Y's verification in the next step.
2) Y $\rightarrow$ X: $\{N_X + 1\}_{Y^{-1}}$
Y should not simply accept the orphan mapping insert message from X without verification – otherwise, attackers may take advantage of the lack of verification to insert fake mappings. For example, a malicious user E in network NA5 may create fake mappings and claim it is the original legitimate storage place for these mappings. The attack would proceed by E then asking NA4 to accept these mappings under the pretense that NA5 is withdrawing its IP prefix. If NA4 accepts all those fake mappings blindly, its own network resources might be exhausted and become victim to a false announcement attack, with the consequence being resource consumption and denial of service. Thus, before accepting the mapping, Y also runs the IP hole protocol on the GUID in the mapping to verify whether the sender X was a legitimate storage location for this mapping. If the insert message passes the verification, Y stores the mapping and replies with an ACK $(N_X + 1)$ to X. Otherwise, Y will reject the insert.

2) Mapping Migration: The Mapping Migration process happens when an AS claims a new IP prefix. The newly announced prefix might have previously been an IP hole that is now being claimed by the AS. As a result, during the process of GNRS update or insert, yet before the prefix announcement, users in the global network will have called the IP hole protocol and avoided storing GNRS mappings at locations with this particular IP prefix. However, after the prefix announcement, this IP prefix is no longer an IP hole and

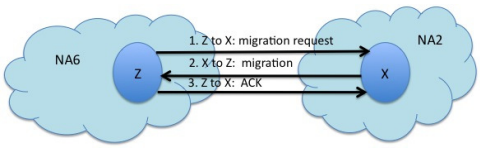

Fig. 8. Mapping Migration.

therefore will receive queries from other users for mappings that it does not have. The Mapping Migration scheme, shown in Figure 8, is designed to solve this problem.

Assume A's GNRS mapping is currently stored in X within NA2 after performing the IP hole protocol. Now assume that network NA6 announces a new IP prefix and that storage location Z is in NA6. Later Z receives a GNRS query for A's mapping, but it does not have the requested mapping due to the original IP hole status prior to the new prefix announcement. To remedy this, Z performs the following procedures to fulfill the query service request:

1) Z $\rightarrow$ X: $\{GUID_Z, [GUID_X, (N_Z, GUID_A)_{Z^{-1}}]_X\}$
   After receiving a query for A's mapping, which it does not have, Z checks whether it should have the mapping by hashing $GUID_A$ or calling the IP hole protocol. During the continuous hashing (up to M times), if the hash result falls into Z's newly announced prefix before it falls into another valid IP address, which should be the current mapping storage location X for A's mapping, then Z is responsible for A's mapping and should start the mapping migration process immediately. Consequently, Z sends out a migration request to X. The request contains A's GUID and a nonce N_Z, which are signed by Z's private key and encrypted by X's public key. Z can determine X by following through with the IP Hole Protocol, and querying the NCRS for X's certificate.
2) X $\rightarrow$ Z: $\{GUID_X, [N_Z + 1, GUID_Z, (GUID_G, (GUID_A, NA1, T_A, E_A)_{A^{-1},G^{-1}}, N_X)_{X^{-1}}]_Z\}$
   X performs the IP hole protocol to verify whether the migration request is legitimate. If one of the hashing results matches Z and the number of required hashes needed to arrive at Z's address is less than the number of hashing attempts needed to arrive at X's address, then X will accept the request and send A's mapping to Z.
3) Z $\rightarrow$ X: $\{N_X + 1\}_{Z^{-1}}$
   Z returns an ACK $(N_X + 1)$ to inform X that it received the mapping successfully. This indicates the end of the mapping migration process.

V. Conclusion

Many clean-slate network architectures promote the separation of identifiers from routable addresses. Such a separation promises to better facilitate mobile networking if a fast global name resolution service can be developed. Although preliminary evidence suggests that fast global name resolution is possible for flat (non-hierarchical) addresses, one critical challenge facing such a service is the assurance that this name resolution service operates in a trustworthy manner in presence of a variety of security threats. This paper has examined the security of such a name resolution service. We started by presenting a high-level threat analysis, where several attack categories were identified. Using the threat analysis as a foundation, we presented the design of several secure protocols that would constitute a secure global name resolution service. A secure update protocol that allows users to update their network addresses as they migrate was presented and includes several checkpoints that allows the protocol to prevent spoofing, collusion, stale identifiers and false identifier announcements. Since the primary function behind a name resolution service is to respond to address-lookup queries, we also presented a secure query protocol. Finally, we noted that IP holes represent a possible attack vector, and consequently devised a protocol that addresses security risks associated with IP holes that arise in a global name resolution service.

References

[1] J. Saltzer, "On the Naming and Binding of Network Destinations," IETF Internet Standard, RFC 1498, August 1993.
[2] C. J. Bennett, S. W. Edge, and A. J. Hinchley, "Issues in the interconnection of datagram networks," July 1977.
[3] R. Moskowitz and P. Nikander, "Host Identity Protocol (HIP) Architecture," IETF Internet Standard, RFC 4423, May 2006.
[4] J. Pan, R. Jain, S. Paul, and C. So-In, "MILSA: A New Evolutionary Architecture for Scalability, Mobility, and Multihoming in the Future Internet," *Selected Areas in Communications, IEEE Journal on*, vol. 28, no. 8, pp. 1344 –1362, october 2010.
[5] MobilityFirst Future Internet Architecture Project. [Online]. Available: http://mobilityfirst.winlab.rutgers.edu/.
[6] T. Vu, A. Baid, Y. Zhang, T. Nguyen, J. Fukuyama, R. Martin, and D. Raychaudhuri, "DMap: A Shared Hosting Scheme for Dynamic Identifier to Locator Mappings in the Global Internet," in *Distributed Computing Systems (ICDCS), 2012 IEEE 32nd International Conference*, June 2012.
[7] D. G. Andersen, H. Balakrishnan, N. Feamster, T. Koponen, D. Moon, and S. Shenker, "Accountable Internet Protocol (AIP)," in *ACM SIGCOMM*, August 2008.
[8] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, and I. Stoica, "ROFL: routing on flat labels," in *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, 2006, pp. 363–374.
[9] W. Trappe and L. C. Washington, *Introduction to Cryptography with Coding Theory*, 2nd ed. Pearson, July 2005.
[10] W. Stallings, *Cryptography and Network Security: Principles and Practices*, 4th ed. Pearson Prentice Hall, 2005.
[11] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*. Springer-Verlag, 2004.
[12] I. T. Laboratory, "FIPS PUB 186-3 DSS," Federal Information Processing Standards Publication, 2009.
[13] "NSA Suite B Cryptography," IETF Internet Standard, 2005.
[14] S. Vanstone and C. Corporation, "ECC Holds Key to Next-Gen Cryptography," 2004.
[15] C. Adams and S. Lloyd, *Understanding PKI: Concepts, Standards, and Deployment Considerations*, 2nd ed. Addison-Wesley Professional, November 2002.
[16] J. R. Vacca, *Public Key Infrastructure: Building Trusted Applications and Web Services*, 1st ed. Auerbach Publications, May 2004.
[17] J. Viega, M. Messier, and P. Chandra, *Network Security with OpenSSL*, 1st ed. O'Reilly Media, June 2002.
[18] R. Chandramouli and S. Rose, "Secure Domain Name System (DNS) Deployment Guide," NIST Special Publication 800-81r1, May 2006.
[19] A. Elmokashfi, A. Kvalbein, and C. Dovrolis, "BGP Churn Evolution: a Perspective from the Core," in *Proceedings IEEE INFOCOM*, 2010, pp. 1–9.
[20] R. Droms and T. Lemon, *The DHCP Handbook*, 2nd ed. Sams Publishing, November 2002.