Efficient Security for IPv6 Multihoming

Marcelo Bagnulo
Universidad Carlos III de Madrid
Av. Universidad, 30
Leganés, Madrid, España
+34 916248837
marcelo@it.uc3m.es

Alberto García-Martínez
Universidad Carlos III de Madrid
Av. Universidad, 30
Leganés, Madrid, España
+34 916248782
alberto@it.uc3m.es

Arturo Azcorra
Universidad Carlos III de Madrid
Av. Universidad, 30
Leganés, Madrid, España
+34 916248778
azcorra@it.uc3m.es

ABSTRACT

In this note, we propose a security mechanism for protecting IPv6 networks from possible abuses caused by the malicious usage of a multihoming protocol. In the presented approach, each multihomed node is assigned multiple prefixes from its upstream providers, and it creates the interface identifier part of its addresses by incorporating a cryptographic one-way hash of the available prefix set. The result is that the addresses of each multihomed node form an unalterable set of intrinsically bound IPv6 addresses. This allows any node that is communicating with the multihomed node to securely verify that all the alternative addresses proposed through the multihoming protocol are associated to the address used for establishing the communication. The verification process is extremely efficient because it only involves hash operations.

Categories and Subject Descriptors

C.2.0 [**Computer Communication Networks**] Security and protection.

General Terms

Security

Keywords

IPv6, multihoming, hijacking protection.

1. INTRODUCTION

In order to preserve global routing system scalability, the IPv6 community is advocating the massive adoption of Provider Aggregatable addressing and limiting the assignment of Provider Independent address blocks to the subscribers of the ISPs (i.e. end sites). Such an approach forces multihomed sites, i.e. sites connecting to the Internet through multiple providers, to obtain multiple Provider Aggregatable prefixes, one from each of their provider's address blocks. Moreover, since ISPs only announce their own prefix block into the global routing system, a multihomed host is reachable at a given address only through the corresponding ISP. Consequently, in order to be reachable through all the available ISPs, a multihomed host, i.e. a host within the multihomed site, needs to configure as many addresses as prefixes are available in the multihomed site.

While this setup guarantees the scalability of the multihoming solution, such multi-addressed configuration presents additional difficulties when attempting to provide the fault tolerance capabilities required to a multihoming solution. In particular, the preservation of established communications when an outage affects the provider through which the communication is flowing becomes challenging, since in order to re-home the communication to an alternative ISP, an alternative address must be used to exchange packets. What is more, such adaptation of the addresses used during the lifetime of the communication to the available providers has to be performed in a transparent fashion with respect to transport and application layers, in order to actually preserve the established communication. This is so because current applications and transport layers, such as TCP and UDP, identify the endpoints of a communication through the IP addresses of the nodes involved, implying that the IP addresses selected at the communication establishment time must remain invariant through the lifetime of the communication. Therefore, after an outage, packets need to carry an alternative address, corresponding to an available ISP, in order to be able to reach their destination, but they need to be presented to transport and application layers as if they contain the original address, in order to be recognized as belonging to the established communication. Such an approach requires additional mechanisms in both ends of the communication in order to perform a coherent mapping between the IP addresses presented to the transport and application layers and those addresses actually contained in the packets [1].

This mapping mechanism between addresses used for forwarding packets (also known as locators) and the addresses presented to the upper layers (also known as identifiers) may be vulnerable to redirection attacks [2] if no proper protection is provided. The vulnerability is introduced when an attacker can benefit from the mapping mechanism to induce a victim to believe that he/she is communicating with the owner of a given identifier while he/she is actually exchanging packets with a locator that does not belong to the owner of the identifier. In other words, a redirection attack consists in creating a false mapping between an identifier and a locator. In particular, if no end to end cryptographic integrity protection is used, a redirection attack can result in communication hijacking, allowing the attacker to impersonate one of the parties involved in the communication.

In this note we propose a security mechanism to protect a protocol for preserving established communication through outages in multihomed environments from redirection attacks. The proposed mechanism relies on the capability of generating all the addresses of a multihomed host as an unalterable set of intrinsically bound

IPv6 addresses. In this system, a multihomed host incorporates a cryptographic one-way hash of the prefix-set available in the multihomed site into the interface identifier part of its own addresses, i.e. the lower 64 bits of the IPv6 address. The result is that the binding between all the addresses of a multihomed host is encoded within the addresses themselves, providing hijacking protection. Any party that is communicating with a multihomed node can efficiently verify that the alternative addresses proposed for continuing the communication are bound to the initial address through a simple hash calculation.

The remainder of this paper is organized as follows. In Section 2 we provide some essential background about multihoming and multihoming security. In Section 3 we present the proposed solution, including a detailed security analysis. Next, in Section 4 we present alternative approaches based in the usage of public key cryptography and we compare them with the proposed solution. We finish this note with a section that includes our conclusions.

2. BACKGROUND

In this section we first present a brief overview of a solution for preserving established communications in multihomed environments and then we attempt to identify potential threats to the resulting system.

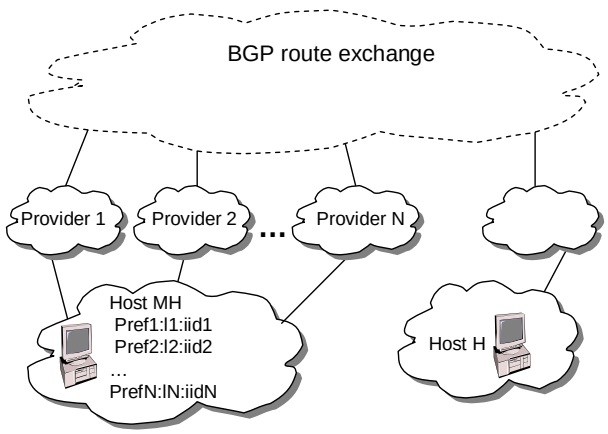

Figure 1. Multihoming with PA addressing.

As we described in the Introduction, a multihomed host that is connected to the Internet through N providers, ISP1,…, ISPN, obtains N prefixes, *Pref1::/n1*,…, *PrefN::/nN*. Consequently, a host located within the multihomed site will have N addresses, one per available ISP/prefix, *Pref1:l1:iid1*, *Pref2:l2:iid2*,…,*PrefN:lN:iidN*, (being *li* the corresponding subnet id as defined in [3]) as presented in figure 1. In order to preserve established communications through outages, a multihoming mechanism located in a shim layer within the IP layer is proposed [4]. The shim layer is located between the IP endpoint sub-layer (that performs end to end functions like fragmenting and IPSec) and the IP routing sub-layer (that performs network related functions like forwarding), as depicted in figure 2. The multihoming mechanism of the shim layer adjusts the address used for exchanging packets according to the available providers, while always presenting a constant address to the upper layers of the stack. The result is that the shim layer performs a mapping between the identifier presented to the upper layers and the locator actually used to exchange packets on the wire. It should be noted that both nodes involved in the communication have to support the mechanism in order to present a coherent view of the addresses involved in the communication. Both ends exchange the information about alternative locators using a multihoming protocol between the shim layers, as presented in figure 2.

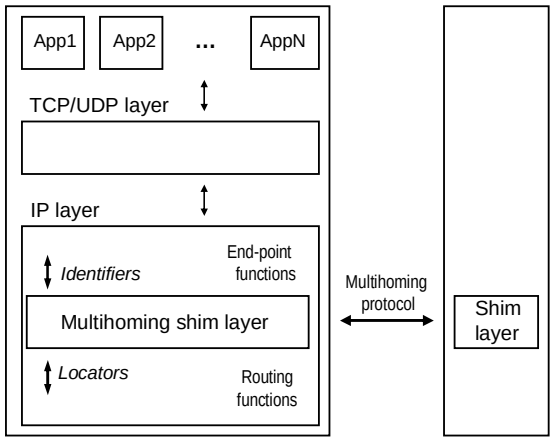

Figure 2. Multihoming protocol layer architecture.

The adoption of such a mechanism enables the possibility of new attacks [2]. The major concern is posed by the possibility of performing so-called *redirection attacks*, where an attacker can persuade a victim to re-home a communication to a locator that is not associated with the identifier used in the communication. There are essentially two types of redirection attack: *hijacking* and *flooding*. In a *hijacking attack* the attacker impersonates one of the parties of the communication. The new hijacking attack enabled by the adoption of a multihoming shim layer is performed by binding the target identifier to the attacker's locator in the multihoming mechanism of the victim node. In this way, when the victim node communicates with the target identifier, it will be actually exchanging packets with the attacker. This is a serious attack, since the attacker is managing to steal the identity of the target node. The shim layer also enables a new type of flooding attack, in which the attacker establishes a communication using its own identifier and then re-homes the communication to a victim's locator. The result is that the victim will be flooded by the flow of the communication initiated by the attacker.

In general terms, it seems wise to require that any additional mechanism introduced to the Internet architecture must, at least, not introduce new vulnerabilities to the network. In this particular case of multihoming, this means that the security of the multihoming solution is not required to protect against man-in-the-middle attacks but it definitely must prevent the so-called *future attacks* [2], also known as *time-shifted attacks* [5], as presented next. In the current Internet, it is clear that any attacker can perform a redirection attack as long as he/she is placed along

the path if no additional security measures are adopted. This means it would be somehow acceptable that a multihoming solution is susceptible to man-in-the-middle attacks, since this vulnerability exists in the current Internet. However, in the current Internet, the effects of a hijacking attack are limited to the period during which the attacker is placed along the path. As soon as the attacker leaves his/her on-path location, the attack finishes. In a future attack the attacker launches the attack from an on-path location and then he/she leaves, but the effects of the attack remain long after the attacker has left. Such attacks could be enabled through a multihoming mechanism, since the attacker may only need to be along the path during the time required to add additional locators through the multihoming mechanism. Proper security measures are required to prevent such attacks. In the next, section we present a mechanism to prevent future hijacking attacks to the multihoming protocol.

3. EFFICIENT SECURITY FOR MULTIHOMING

In this section we present the Efficient Security for Multihoming (ESM) architecture to prevent future hijacking attacks and flooding attacks in multihomed environments. The ESM architecture consists of a novel technique to generate a new type of IPv6 addresses called Hash Based Addresses (named HBAs), and a security protocol to protect the multihoming information. In this section we will first describe the algorithm for generating sets of Hash Based Addresses as sets of intrinsically bound IPv6 addresses associated with an arbitrary set of prefixes. Then we will describe how these addresses are used in the ESM protocol for protecting multihoming exchanges of information.

3.1 Hash Based Addresses

We next describe a procedure to generate sets of Hash Based Addresses (HBAs). HBAs are cryptographic in nature, because they contain a one-way hash of the prefix set available in the multihomed site and other parameters in the identifier part of the addresses. In other words, given an arbitrary set of N prefixes, the HBA set generation algorithm produces a HBA set of N addresses. Each one of the generated addresses has a different prefix from the input prefix set, while their interface identifier part contains information about the complete prefix set in the form of a hash of the set. Because of their nature, each address contains information about all the other addresses of the set and a receiver can easily verify if two addresses belong to the same set through an efficient operation such as a hash. After this verification, the receiver can securely use them interchangeably, as we will see in Section 3.2.

So, in order to benefit from the proposed security mechanism, the addresses of each multihomed host have to constitute an HBA set. In a general multihoming scenario as the one presented in Section 2, a multihomed host attached to a link where N 64-bit prefixes [3] are available (*Pref1:l1::/64, Pref2:l2::/64,…, PrefN:lN::/64*) generates the interface identifier part of each one of its addresses as a 64-bit hash of the prefix set available in the link and a random nonce. Including a random nonce enables the generation of multiple HBA sets associated with the same prefix set. The "u" and the "g" bits of the interface identifier are set to zero in order to avoid confusing the resulting addresses with globally unique EUI-64 identifiers [6]. After generating the interface identifier parts, the addresses of the HBA set are finally generated by prepending the different prefixes of the prefix set with the interface identifier parts.

So, summarizing, the procedure for generating an HBA set of N intrinsically bound addresses for a prefix set containing *Pref1:l1::/64, Pref2:l2::/64,…, PrefN:lN::/64* is the following:

First, a 128-bit random nonce *RN* is generated. Proper care should be taken for ensuring randomness as it is discussed in [7]

Second, for each prefix *Prefi:li::/64*, an interface identifier is generated as

$$iid = \mathrm{Hash}_{64bit}(RN, Pref1:l1, \ldots, PrefN:lN)$$ (the "u" and the "g" bits are reset)

The corresponding address is generated by prepending the prefix with the interface identifier part (*Prefi:li:iid)*

After generating the address set, the node performs the Duplicate Address Detection procedure as defined in [8]. If any of the addresses collides with an existing one, a new random nonce is generated and a new HBA set is generated.

The output of the described procedure is a HBA set of N addresses that carry information about the prefixes available in the multihomed site within their interface identifier part. The generation procedure is completely automatic, and it does not require any manual configuration, eliminating any administrative burden usually required by security procedures.

It should be noted that it would be possible to generate the addresses of an HBA set with different interface identifiers to provide some privacy features. In order to do that, it is necessary to change the order of the inputs of the hash function when calculating the interface identifiers for each prefix.

3.2 Security Protocol

Once the multihomed host has generated its addresses as a HBA set, we propose the ESM protocol to securely exchange multihoming information. We next describe the ESM protocol using the following notation: MH and H are principals, being MH the multihomed host and H any other host of the Internet. *Pref1:l1:iid, Pref2:l2:iid ,…, PrefN:lN:iid* are the addresses of MH that were generated as a HBA set. *RN* is the random nonce associated with the HBA set. A_H is the address of H (without loss of generality, we are assuming here that H has a single address for simplicity).

Suppose that MH is communicating with H and that they are using addresses *Prefi:li:iid* and A_H respectively. So far, no multihoming specific features were used; in particular, *Prefi:li:iid* and A_H are used both as identifiers for upper layer protocols (transport and application protocols) and as locators for exchanging packets. In order to benefit from enhanced fault tolerance capabilities provided by multihoming, MH informs H about the alternative addresses available for the communication, so that they can be used in case of an outage.

Since MH's addresses form an HBA set, it is enough for MH to convey the information needed by H to re-generate the HBA set, i.e. the prefix set and the random nonce *RN*.

MH —› H: {*Pref1:l1/64, Pref2:l2/64,…, PrefN:lN/64*}, *RN*

Upon the reception of the HBA set information, H verifies that the address used for establishing the communication, *Prefi:li:iid*, belongs to the HBA set. For that purpose, H first verifies that *Prefi:li::/64* is included in the received prefix set. If this verification is successful, H then verifies that *Prefi:li:iid* corresponds to the address #i of the HBA set generated using the generation algorithm described in the previous section with the received parameters.

Once H has verified that the HBA set associated with the received parameters contains the initial address *Prefi:li:iid*, H generates the full HBA set using the generation algorithm described in the previous section. So far, H is certain that all the alternative addresses available for MH have been generated by the same entity.

Then H verifies that the party located at the alternative address is willing to receive the traffic associated to the established communication at this new location to prevent flooding attacks. To achieve this, a reachability test is included in the ESM protocol that consists of a two-way exchange. The defined packet exchange includes the random nonce *RN* to verify that the same entity is located at the initial and the alternative address. The provision of additional protection against flooding attacks may require tools that are out of the scope of this note.

It should be noted, that the last two operations, i.e. the regeneration of the HBA set and the reachability tests for the alternative locators, do not need to be performed upon the reception of the prefix set, but they can be deferred until an alternative locator is needed because of an outage (which may never occur). Moreover, when an alternative locator, is needed, it is not required to regenerate the whole HBA set, but alternative locators can be regenerated independently using the involved parts of the aforementioned HBA creation procedure.

By means of the ESM protocol, H can securely use any of the addresses of the HBA set interchangeably for exchanging packets in the established communication.

3.3 Security Analysis

3.3.1 Protection from hijacking attacks

The security of the protocol to protect from hijacking attacks is based on the property that any modification of the inputs of the HBA set generation process would result in a different set. Since what is being protected is the mapping between different addresses, producing a valid mapping between other addresses that are not contained in the original set is not an actual threat. We will next illustrate this argument by presenting a possible attack and calculating the effort required to perform it.

In the scenario presented in the previous section, a multihomed host MH is communicating with another host H. MH has generated its addresses through the HBA set generation algorithm, resulting in *Pref1:l1:iid, Pref2:l2:iid ,…, PrefN:lN:iid*. Host H has a single address A_H. MH and H are communicating using addresses *Prefi:li:iid* and A_H respectively. Consider now an attacker X that has the intention of redirecting the communication to an alternative address. We assume that it is enough for the attacker to redirect the communication to any address of a given prefix, *PrefX::/64*. The rationale behind this assumption is that X has access to any address of the considered prefix.

So, in order to hijack the communication, X must introduce a new prefix in the prefix set used for generating the HBA set of MH. For that, X is required to obtain a combination of prefix set and random nonce such as:

1- *Prefi:li/64* and *PrefX::/64* are included in the prefix set

2- *Prefi:li:iid* is included in the resulting HBA set

The other inputs may be changed at will by the attacker; for instance, the random nonce and the other prefixes of the prefix set can be altered. In any case, in order to obtain the desired HBA set, the attacker needs to try with different inputs, for instance with different random nonces, until the above two conditions are met. The expected number of times that the generation procedure needs to be repeated until the desired outcome is reached depends on the number of hash bits included in the interface identifier part of the HBAs. Since we are considering 64-bit long interface identifiers and that the "u" and the "g" bits are not used, the expected number of iterations required for a brute force attack is O(2^61).

In order to quantify the actual effort required to perform such attack, we next calculate the amount of time that is required for an attacker to obtain the proper parameter set. As stated before, the attacker needs to perform O(2^61) hash operations and the corresponding comparisons. Assuming that the attacker uses only two prefixes and the modifier, in order to minimize the amount of data to be hashed, each round the attacker will need to hash 32 bytes. According to `openssl speed`[1], a computer with a Pentium 4 processor (2.66 Ghz) and 440 MB of RAM, can hash 20945 kB per second, when hashing blocks of 32 bytes. This means that it would take approximately 110.000 years to perform the number of hash operations required to obtain the proper parameter set.

We believe that the resulting security is enough for protecting regular traffic, which currently flows unprotected through the network, from potential redirection attacks introduced by the multihoming mechanisms, since the resulting protection is similar to the one offered by other current network security protocols such as the protection provided by Cryptographically Generated Addresses [9] (CGA) in SeND [10].

As processing power increases, the protection provided by this mechanism decreases, since the amount of time required to try with 2^61 different random nonces also decreases. However, additional mechanisms can be used to improve the protection provided by the ESM protocol. For instance artificially increasing the effort required for generating a valid HBA set, similar to the Sec parameter used in CGAs, would result in additional protection.

3.3.2 Protection from flooding attacks

In this section we will present that the usage of the HBA format makes very difficult to launch a flooding attack against a specific address, while it does not prevent from flooding attacks against a specific prefix. Therefore additional protection such as a reachability test is required.

Consider the case where an attacker X has easy access to a prefix *PrefX::/64*. X wants to launch a flooding attack to a host located

[1] `openssl speed` is a command part of the OpenSSL Project, www.openssl.org.

in the address *Prefi:li:iid*. The attack would consist of establishing a communication with a server S and requesting a heavy flow from it. Then simply redirect the flow to *Prefi:li:iid*, flooding the target. In order to perform this attack X needs to generate an HBA set including the prefixes *Prefi:li::/64* and *PrefX::/64* in the prefix set. Additionally the resulting HBA set must contain *Prefi:li:iid*. In order to obtain this, the attacker needs to find the appropriate Random Nonce *RN*. The expected number of attempts required to find such RN value is O(2^61). Because of this we can conclude that HBAs provide sufficient protection from this type of attacks.

However, the target of a flooding attack is not limited to specific hosts, but the attack can also be launched against other elements of the infrastructure, such as routers or access links. In order to do that, the attacker can establish a communication with a server S starting the download of a heavy flow. Then, the attacker redirects the communication to any address of the prefix assigned to the target network. Even if the target address is not assigned to any host, the flow will flood the access link of the target site, and the site access router will also suffer the overload. Such attack cannot be prevented using HBAs, since the attacker can easily generate an HBA set using his own prefix and the target network prefix. In order to prevent such attacks, additional mechanisms such as reachability tests are required.

4. COMPARATIVE ANALYSIS

As it is described in Section 2, the goal of the proposed protocol is to secure the binding between the address that is being used as identifier by the upper layer protocols and the alternative addresses that will be used as locators for that communication. There are alternative mechanisms to achieve the same goal. However, we argue that the ESM protocol is the most efficient one, because it does not involve asymmetric key operations and it does not require any infrastructure for key distribution. In this section we present a brief description of some alternative approaches based on public key cryptography, and then we perform a qualitative and a quantitative comparison between the public key based approaches and ESM.

4.1 Alternative Approaches

Several alternative approaches are based on the usage of public key cryptography. Among them we can find Strong Identity Multihoming (SIM) protocol [11], the application of the Host Identity Protocol (HIP) [12] to multihoming [13] and the application of Cryptographic Generated Addresses (CGA) [9] [14] to multihoming [15].

The first two approaches, i.e. SIM and HIP, create a new 128-bit identifier namespace. The new endpoint identifier is the 128-bit hash of the public key of the node. In a CGA based approach, no new identifier namespace is created, but the address of the multihomed node is a CGA that contains a hash of a public key in its interface identifier. In any case, the result is a secure binding between the identifier (whether a new 128-identifier or the CGA) and the associated key pair. This allows the multihomed host to use the corresponding private key to sign the multihomed messages that convey alternative address information. The trust chain in this case is the following: the identifier used for the communication is securely bound to the key pair because it contains the hash of the public key, and the alternative address is bound to the public key through the signature. This approach provides the required protection, since it is invulnerable to *future hijacking attacks*. Additional flooding protection similar to the one used in ESM is still required though.

Other approaches are also possible, but when the address used as an identifier by the upper layers is not intrinsically bound to something else (e.g. a public key or a prefix set), an external trust source is required to provide the binding. This means that a third trusted party, like a public key infrastructure (PKI) is required. Such approaches are extremely difficult to deploy because they require a global infrastructure for key distribution making its application unsuitable for the general case where any two arbitrary nodes in the Internet are communicating. On the other hand, this approach may make sense in a restricted environment where such infrastructure is available for the involved parties. However, this solution would still relay on extensive usage of public key operations. This implies that the computational cost of the operation would be similar to the case of CGAs, that will be shown on section 4.3.

4.2 Qualitative Comparison

The major advantage of a public key based approach with respect to ESM is that they support dynamic address sets. In ESM, the HBA set is determined at the generation moment and cannot be changed afterwards, implying that no new alternative addresses can be added during the communication. In a public key based approach, it is possible to add new alternative locators at any point of the lifetime of the communication, facilitating an integrated mobility-multihoming support. This is due to the additional level of indirection provided by the binding with the key pair. However, the public key based approach requires the intensive usage of public key cryptography, since for each addition of a new alternative address public key operations are performed. On the other hand, the ESM protocol only requires hash operations, which are cheaper, as the results of the quantitative analysis performed in the next section show. This is particularly relevant in some scenarios like highly loaded public servers that maintain thousands of simultaneous communications.

As presented earlier, the different public key based approaches use diverse identifier namespaces. In particular, SIM and HIP create a new identifier namespace while the CGA-based approach use IPv6 addresses as identifier. This difference has a great impact in terms of deployment and backward compatibility. As opposed to IPv6 addresses, the new identifiers used in SIM and HIP cannot be used as locators. This means, that a new directory service that provides a mapping between identifiers and locators is needed to allow endpoints to obtain the locators corresponding to a given identifier. The implementation of such directory service is far from trivial, since the proposed identifier namespace is flat because of its own cryptographic nature. The lack of such directory service results in a poor support of some of the existent applications. For instance, applications that perform referrals or call-backs would simply fail if no identifier-to-locator mapping is available [16]. Because of this limitation, we consider that the SIM and HIP approaches are not directly comparable in quantitative terms to the ESM approach.

On the other hand, a CGA-based approach would indeed provide similar application support. Moreover, it should be noted that the resulting security level is the same in both approaches, since the strength of the resulting binding is determined by the number of bits of the interface identifier part of the IPv6 address. The result

is that these two approaches provide similar capabilities and application support, enabling a detailed quantitative comparison between them.

4.3 Quantitative Comparison

In this section we compare the computational cost of the CGA-based approach and the proposed ESM mechanism. We analyze the two major operations involved: the bootstrap process where the elements required in each approach are generated and the establishment of the session context where the alternative locators are exchanged and verified.

4.3.1 Bootstrapping

During the bootstrap process, the elements required for each mechanism are generated. In the case of ESM, the HBA set is generated while in the CGA-based mechanism the set of CGAs is generated. We will next compare the computational effort required in each case.

In the case of ESM, the bootstrap process involves the generation of the HBA set. For this, the generation procedure described in Section 3.1 is executed. The computational effort of such procedure is due to the generation of the 128-bit random nonce *RN* and the posterior hash operation. So, in a configuration where N prefixes are available in the multihomed site, then the computational effort of generating an HBA set is:

$$G_{HBA}(N) = G_{RN} + H(N)$$

being G_{RN} the effort of generating a 128-bit random number and H(N) the effort of computing a hash of N prefixes.

In the case of CGA, the bootstrap process starts with the generation of the public and private key pair. Then, for each of the N prefixes, a random nonce is generated and the address is generated including the hash of the random nonce, the public key and the correspondent prefix. So, in a scenario where there are N prefixes in the multihomed site, the computational effort of generating the CGA addresses is:

$$G_{CGA}(N) = G_{Pkpair} + N * (G_{RN} + H)$$

being G_{Pkpair} the effort of generating a public/private key pair, G_{RN} the effort of generating a 128-bit random nonce and H the effort of hashing the public key, the random nonce and a prefix.

The efforts $G_{HBA}(N)$ and $G_{CGA}(N)$ can be measured and compared in terms of the time required for the computation of the involved operations. We have calculated the values for $G_{HBA}(N)$ and $G_{CGA}(N)$ using the *OpenSSL* tools in a computer with a Pentium 4 processor (2.66 Ghz) and 440 MB of RAM for different values of N (2, 3, 5, 10) and for two different RSA public key lengths (512 and 1024 bits). The hash algorithm used is SHA-1. The results are included in Table 1.

Table 1: Time required for HBA and CGA bootstrapping

N	G_{HBA}	$G_{CGA-512}$	$G_{CGA-1024}$
2	6 μs	31,9 ms	157,8 ms
3	6 μs	31,9 ms	157,8 ms
5	6 μs	31,9 ms	157,8 ms
10	7 μs	31,9 ms	157,8 ms

The above results show that the effort required for bootstrapping the ESM mechanisms is near to 4 orders of magnitude lower that the effort required for bootstrapping a CGA based solution (using 512 bits keys).

However, the bootstrapping process is not frequently executed, so a poor performance is not so critical and cannot be determinant to select the ESM solution over the CGA-based solution. In the next section we analyze the effort of establishing a session context, which is a much more significant operation because of its frequency.

4.3.2 Session Context Establishment

After the bootstrapping, the node is ready to benefit from the capabilities provided by the multihoming solution. This means that when a new communication is established, the node can setup a multihoming session context so that the communication can be preserved through outages. Such session context establishment includes the exchange and validation of the alternative locators for that communication. In this section we will compare the effort required for the verification of the alternative locator set in each one of the analyzed approaches.

As described in section 3.2, in the case of the ESM protocol, the node that receives the alternative locator set of its peer verifies it though the following operation: it first generates the HBA set associated with the received parameter and then it verifies if the address used as identifier for the communication is included in the resulting set. Therefore, the effort of the verification process in a scenario with N prefixes is:

$$V_{HBA}(N) = H(N)$$

being H(N) the effort of computing the hash of N prefixes.

In the case of the CGA-based protocol, the node receiving the alternative locators from its peer will perform two operations: first, it verifies that the received public key corresponds to the CGA used as identifier of the established communication, and next it verifies the signature of the message that contains the alternative locator set. The effort of the verification process with N prefixes is:

$$V_{CGA}(N) = H + S_{PK}(N-1)$$

being H the effort of computing a hash of the public key and $S_{PK}(N-1)$ the effort of verifying a signature of a message with N-1 alternative locators.

Similarly to the previous section, we measure the effort in terms of the time required to perform the computations. So, using the same tools and the same infrastructure as in the previous section, we calculate the computational effort for establishing a session context for different values of N (2, 3, 5, 10) and for two different RSA public key length values (512 and 1024 bits). The hash algorithm used is SHA-1. The results are presented in Table 2.

Table 2: Time required for a locator set verification

N	V_{HBA}	$V_{CGA-512}$	$V_{CGA-1024}$
2	1,5 µs	141 µs	425 µs
3	1,5 µs	176 µs	531 µs
5	1,5 µs	244 µs	742 µs
10	1,6 µs	418 µs	1271 µs

The obtained results show that the verification procedure of the ESM approach is at least 2 orders of magnitude more efficient than the CGA based verification process. The session context establishment operation is used extensively by the protocol, since each new communication performs it. So, an increase of a couple of orders of magnitude in this operation seems definitely determinant for selecting a security mechanism for a general purpose multihoming mechanism as the one being consider in this paper.

5. CONCLUSIONS

We have described the ESM architecture, a novel security solution for protecting multihoming mechanisms from redirection attacks. The resulting protection prevents future hijacking attacks and flooding attacks. However, the proposed solution does not provide protection against man-in-the-middle attackers.

The ESM architecture achieves the described protection by defining a new type of addresses called HBAs that contain within the interface identifier a one-way hash of the prefix set available in the multihomed site. This technique allows each multihomed host to create a secure binding between all its available addresses. The resulting binding is easily verifiable by the communicating nodes since it is contained in the addresses themselves. Probably, the most remarkable feature of the proposed solution is that no cryptographic keys or secrets are used in the protocol. Moreover, all the information used in the ESM protocol is exchanged in clear text and can be sniffed and/or spoofed by any attacker without compromising the security. Instead, the security of the protocol is based in the fact that any modification of the parameters used to generate the address would result in a different address, i.e. the attack would not affect the target but an alternative address.

The ESM protocol is extremely efficient because it only involves hash operations; no public key operations are required. In addition, the proposed solution is easy to use, because manual configuration is not required. Compared with a similar approach based on the usage of CGA, the proposed ESM mechanisms is near to 4 orders of magnitude more efficient during the bootstrapping and it is over 2 orders of magnitude more efficient during the lifetime of the communication.

However, it should be noted that the CGA based mechanism provides some features that may be required for certain scenarios, like mobile environments. In this case, it is possible to merge these two approaches into an integrated mechanism by including a hash of both a public key and a prefix set as inputs into the interface identifier of the IPv6 address. For this it is possible to define a new CGA extension that contains the prefix set [17]. The resulting mechanism would allow the usage of the HBA and/or CGA features depending on the situation preserving the enhanced efficiency when the HBA mode is used.

6. ACKNOWLEDGMENTS

The ideas contained in this note benefited from discussions with Jari Arkko, Iljitsch van Beijnum, Geoff Huston, Erik Nordmark, Margaret Wasserman, and Jukka Ylitalo.

Our thanks also go to Huw Oliver, Ignacio Soto, Albert Banchs, Isaías Martínez and the anonymous reviewers for their comments.

This work has been partially supported by the European Union under the E-Next Project FP6506869, and by the Spanish National Research and Development Programme through OPTINET6 (TIC-2003-09042-C03-01) and SAM (TIC2002-04531-C04-03) projects.

7. REFERENCES

[1] Huston, G. *Architectural Approaches to Multi-Homing for IPv6*. Internet Engineering Task Force (IETF), Internet Draft (work in progress), October 2004.

[2] Nordmark, E. and Li, T. *Threats relating to IPv6 multihoming solutions*, Internet Engineering Task Force (IETF), Internet Draft (work in progress), September 2004.

[3] Hinden, R. and S. Deering, *IP Version 6 Addressing Architecture*, Internet Engineering Task Force (IETF), RFC 3513, April 2003.

[4] Nordmark, E. and Bagnulo, M. *Multihoming L3 Shim Approach*, Internet Engineering Task Force (IETF), Internet Draft (work in progress), October 2004.

[5] Nikander, P., Arkko, J., Aura, T., Montenegro, G., Nordmark, E., *Mobile IP version 6 Route Optimization Security Design Background*, Internet Engineering Task Force (IETF), Internet Draft (work in progress), July 2004.

[6] Deering, S. and Hinden, R. *Internet Protocol, Version 6 (IPv6) Specification*, Internet Engineering Task Force (IETF), RFC 2460, December 1998.

[7] Eastlake, D., Crocker, S., Schiller, J. *Randomness Recommendations for Security.* Internet Engineering Task Force (IETF), RFC 1750. December 1994.

[8] Narten, T., Nordmark, E. and Simpson, W. *Neighbor Discovery for IP Version 6 (IPv6)*, Internet Engineering Task Force (IETF), RFC 2461, December 1998.

[9] Aura, T., *Cryptographically Generated Addresses (CGA)*, Internet Engineering Task Force (IETF), RFC 3972, March 2005.

[10] Arkko, J., Kempf, J., Sommerfeld, B., Zill, B., Nikander, P., *SEcure Neighbor Discovery (SEND)*, Internet Engineering Task Force (IETF), RFC 3971, Marzo 2005.

[11] Nordmark, E., *Strong Identity Multihoming using 128 bit Identifiers (SIM/CBID128)*, Internet Engineering Task Force (IETF), Internet Draft (work in progress), October 2003.

[12] Moskowitz, R., Nikander, P., Jokela, P., Henderson, T., *Host Identity Protocol*, Internet Engineering Task Force (IETF), Internet Draft (work in progress), October 2004.

[13] Nikander, P., Arkko, J., Henderson, T., *End-Host Mobility and Multi-Homing with Host Identity Protocol*, Internet Engineering Task Force (IETF), Internet Draft (work in progress), October 2004.

[14] O'Shea, G., Roe, M., *Child-proof Authentication for MIPv6, CAM*. ACM Computer Communications Review, 31(2), April 2001.

[15] Nordmark, E., *Multihoming using 64-bit Crypto-based IDs*, Internet Engineering Task Force (IETF), Internet Draft (work in progress), October 2003.

[16] Nordmark, E., *Multi6 Application Referral Issues*, Internet Engineering Task Force (IETF), Internet Draft (work in progress), January 2005.

[17] Bagnulo, M., *Hash Based Addresses (HBA)*, Internet Engineering Task Force (IETF), Internet Draft (work in progress), October 2004.