Securing IPv6 Neighbor and Router Discovery

Jari Arkko
Ericsson Research NomadicLab
L M Ericsson
FIN-02420 JORVAS, FINLAND
+358-9-299-1
jari.arkko@ericsson.com

Tuomas Aura
Microsoft Research Cambridge
7 J J Thomson Ave,
Cambridge CB3 0FB, UK
+44 1223 479700
tuomaura@microsoft.com

James Kempf
DoCoMoLabs U.S.A.
180 Metro Drive
San Jose, CA 95110, USA
+1-408-451-4711
kempf@docomolabs-usa.com

Vesa-Matti Mäntylä
Ericsson Research NomadicLab
Tutkijantie 2 C
FIN-90570 OULU, FINLAND
+358-9-299-1
vesa-matti.mantyla@ericsson.fi

Pekka Nikander
Ericsson Research Nomadiclab
c/o HIIT, P.O.BOX 9800
FIN-02150 HUT, FINLAND
+358-9-299-1
pekka.nikander@hiit.fi

Michael Roe
Microsoft Research Cambridge
7 J J Thomson Ave,
Cambridge CB3 0FB, UK
+44 1223 479700
mroe@microsoft.com

ABSTRACT

When IPv6 Neighbor and Router Discovery functions were defined, it was assumed that the local link would consist of mutually trusting nodes. However, the recent developments in public wireless networks, such as WLANs, have radically changed the situation. The nodes on a local link cannot necessarily trust each other any more, but they must become mutually suspicious even when the nodes have completed an authentication exchange with the network. This creates a number of operational difficulties and new security threats. In this paper we provide a taxonomy for the IPv6 Neighbor and Router Discovery threats, describe two new cryptographic methods, Cryptographically Generated Addresses (CGA) and Address Based Keys (ABK), and discuss how these new methods can be used to secure the Neighbor and Router discovery mechanisms.

Categories and Subject Descriptors

C.2.6 |Internetworking]: Standards — *Internet Protocol version 6 (IPv6);* E.3 [Data Encryption]: Public key cryptosystems — *Cryptographically Generated Addresses (CGA), Address Based Keys (ABK)*

General Terms

Security, Standardization

Keywords

Neighbor Discovery, Router Discovery, Duplicate Address Detection, Autoconfiguration, Identity-Based Cryptosystems.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
WiSe '02, September 29, 2002, Atlanta, Georgia, USA.
Copyright 2002 ACM 1-58113-585-8/02/0009…$5.00.

1. INTRODUCTION

Ten years ago, when the basic design for IPv6 [1][2] was being decided, it was hardly possible to foresee the kinds of wireless environments that are now being considered for use with IPv6. Correspondingly, the IPv6 functions that manage the local link were designed with physically protected, trustworthy links in mind. However, now people are planning to use IPv6 on public radio networks, such as Wireless LANs at airports, hotels, and cafes. Even though the actual link may still be somewhat protected with layer 2 authentication, access control, and encryption (e.g. with IEEE 802.1x [3] and 802.11i [4]) some of the nodes on the link may be untrustworthy. Furthermore, it is fairly easy to set up a phony WLAN base station, leading to various kinds of access stealing, DoS, and traffic snooping attacks [5].

In this paper, we focus on IPv6 Neighbor Discovery (ND) and Router Discovery (RD) functions. Their current definition relies on the assumption that there are no untrustworthy nodes at the local link. In practice, even a single untrustworthy node can launch various kinds of attacks, including Denial-of-Service (DoS), Man-in-the-Middle (MitM), and masquerade. The current set of RFCs [6][7][8][9] do acknowledge the situation to a degree, but do not provide much detail about how to use the suggested protection mechanism, IPsec. Unluckily, there are a number of problems when using IPsec for securing Neighbor Discovery [10].

In this paper, we outline the current situation and describe our initial attempts to improve it. In Section 2 we describe the background, i.e., the current technology. Section 3 outlines the threat model that we have in mind, discussing open network environments and untrustworthy network nodes. In Sections 4 and 5 we briefly introduce two recently developed technologies, Cryptographically Generated Addresses (CGA) and Address Based Keys (ABK). These technologies turn out to be quite useful in bringing security to the IPv6 local link. In Section 6 we show how to use CGA to secure Neighbor Discovery, including Duplicate Address Detection. Section 7 continues with discussing CGA and ABK in the context of Router Discovery. Finally, Section 8 contains our initial conclusions from this research.

2. BACKGROUND

In this section we briefly describe the current state of the technology. We assume that the reader is familiar with the basic IPv6 architecture and functions. Thus, we concentrate on explaining the essential points of Neighbor and Router Discovery functions, focusing on security related issues.

2.1 Neighbor Discovery

The purpose of IPv6 Neighbor Discovery (ND) [7] is to provide IPv6 nodes with a means to discover the presence and link-layer addresses of the other nodes on the local link. Additionally, it provides methods for discovering routers on the local link, for detecting when a local node becomes unreachable, for resolving duplicate addresses, and for routers to inform nodes when another router is more appropriate (redirect). We look at all of these functions individually; for the purposes of this paper, we make a distinction between Neighbor Discovery and Router Discovery functionality, since these two functions seem to have different security properties. In this section, we look at address resolution, neighbor unreachability detection, and duplicate address detection; router discovery and redirects are covered in Section 2.2.

2.1.1 Address Resolution

To learn the link-layer address of another node that is assumed to be directly attached to the local link, the node that needs the address sends a Neighbor Solicitation (NS) message to a multicast address specified by the target address. If the target node is indeed present, it should be listening to the multicast address. Upon receiving the solicitation, it replies with a Neighbor Advertisement (NA) message. The default operation is illustrated in Figure 1. Additionally, the specification defines that the messages *may* be protected with IPsec AH. However, as explained in Section 3.3, the AH protection does not work in practice due to key distribution problems.

From the security point of view, there are additional problems besides authentication. First, the NA includes a number of flags. One of the flags indicates that the replying node is actually a router. Another one is an "override" flag, specifying that the information in the packet should replace any information that the receiver(s) of the packet may already have. However, unless the authentication keys are strongly bound to IP addresses, the receiving node does not have any means to make sure that the sender of the authenticated packet is indeed authorized to claim "ownership" over the address [11][12].

2.1.2 Neighbor Unreachability Detection (NUD)

Nodes on the link monitor the reachability of local destinations and routers in the Neighbor Unreachability procedure [7]. Normally the nodes rely on upper-layer information to determine whether peer nodes are still reachable. However, if there is a sufficiently long delay on upper-layer traffic, or if the node stops receiving replies from a peer node, the NUD procedure is invoked. The node first waits for a small random delay, and then sends a targeted NS to the peer node. If the peer is still reachable, it will reply with a NA. However, if the soliciting node receives no reply, it tries a few more times, eventually deleting the neighbor cache entry. If needed, this triggers the standard address resolution protocol. No higher

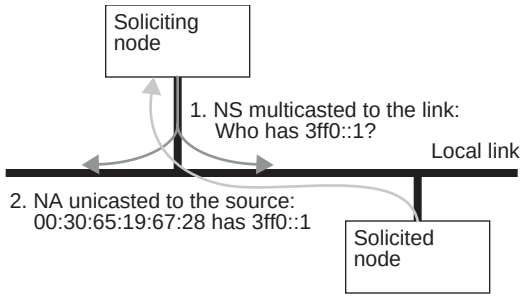

Figure 1. Basic Neighbor Discovery

level traffic can proceed if this procedure flushes out neighbor cache entries after (perhaps incorrectly) determining that the peer is not reachable.

2.1.3 Duplicate Address Detection (DAD)

When a node plans to take a new address for its own use, it must first make sure that no other node on the link uses that particular address. This is accomplished by sending a series of Neighbor Solicitation messages to the local link. These messages contain the tentative IP address that the host would like to use. If the tentative address is already in use by some other host, the node already using the address will send a Neighbor Advertisement as a reply, and the first host must select a new tentative address. If the first host receives no replies to its solicitations, it is free to use the address. The process is illustrated in Figure 2.

2.2 Router Discovery

While neighbor discovery functions allow a host to communicate within the local link, the nodes also need to learn the identities and capabilities of any routers attached to the link. In addition to allowing the nodes to learn the routers present, the router discovery functions also provide the nodes with globally routeable address prefixes.

Normally all routers on a local link multicast Router Advertisement (RA) messages periodically. In addition to identifying the sending router, each Router Advertisement also contains a number of routing prefixes, which the nodes can use for creating globally routeable addresses for themselves. Additionally, a node may initiate router discovery by sending a

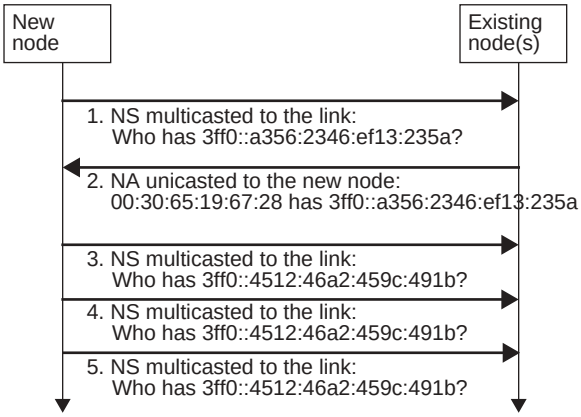

Figure 2. Duplicate Address Detection

Router Solicitation (RS) message, basically requesting the local routers to announce themselves. The local routers are expected to answer. If the soliciting node receives no replies, it may repeat the RS message twice.

2.2.1 Redirects

In the Redirect procedure, a router informs a node of a better first-hop to reach a particular destination. The purpose is to optimize routing. Communication could be established and maintained without Redirects because the router will forward packets even if they ideally should have been sent via another path.

The Redirect message is used solely in the Redirect procedure. The Redirect message is always sent from a unicast address to the source address of the packet that triggered the redirect [7]. As all messages discussed in this paper, Redirect messages are only used for link local purposes, not for end-to-end communications.

2.3 Autoconfiguration

The stateless autoconfiguration specification [8] defines a method of providing initial boot time configuration for any IP host. In the basic stateless autoconfiguration process, a booting host first performs Duplicate Address Detection. Once the host has a link local address, it enters the second phase of autoconfiguration, Router Discovery. With Router Discovery, the host learns the identities and capabilities of any local routers, and becomes able to configure globally routeable addresses for itself. Finally, the host needs to learn the identities for local DNS servers. The IETF IPv6 working group is in the process of defining a protocol for DNS server discovery [13]. The phases of the autoconfiguration process are illustrated in Figure 3.

3. THREAT MODEL

ND security becomes important in open network environments where anyone can join a local link either with minimal or no link-layer authentication. In such an environment, there may be malicious nodes among the nodes on the local link. Thus, very little can be trusted, unless a security context already

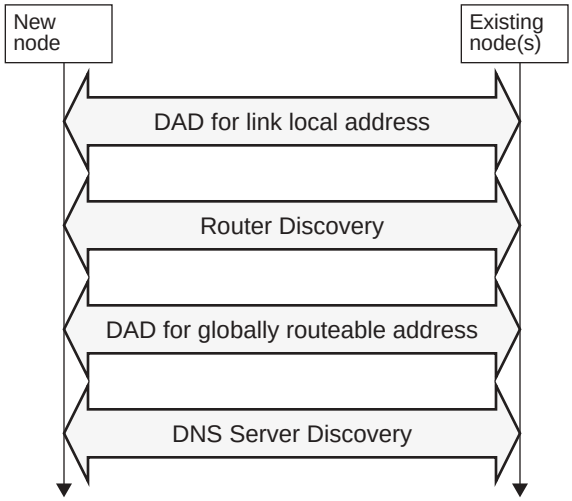

Figure 3. A typical autoconfiguration process

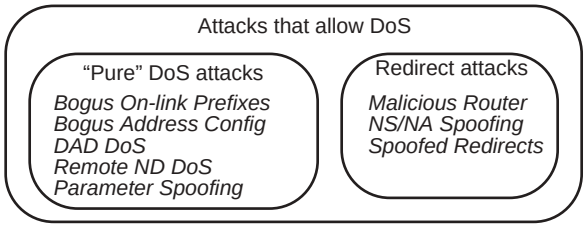

Figure 4. Attack categories

exists between a given node and the infrastructure. These constraints should be kept in the mind while considering the threats and attacks, as described below.

It should be noted that many of the threats and attacks that we discuss below are well known in the IPv4 world [14]. However, we feel that it is now appropriate to address these problems in the context of IPv6, as IPv6 is being increasingly used, particularly in wireless public access networks.

Our threat model is based on one presented by Nordmark and Kempf [15]. In general, there are two types of threats:

1. Various kinds of Denial of Service (DoS) threats, in which a malicious node prevents communication between the node under attack and all other nodes or a specific destination address.
2. Redirect threats, in which an attacker redirects packets away from the last hop router to another node on the link. While redirect attacks can be used for DoS purposes, they also allow a number of other kinds of attacks.

The relationship between the two threat categories, i.e. attacks that allow only DoS and those that allow also traffic redirection, is depicted in Figure 4.

The redirect attacks can be used for three principal purposes:

1. Packets can be snooped and intercepted even in the case where it would not otherwise be possible, e.g. on a switched LAN.
2. Packets can be redirected to a wrong or non-existent link layer address, thus preventing normal communication by the true owner of the IP address, resulting in DoS.
3. Large amounts of data can be redirected to an existing link layer address, thus flooding the network interface, the processor and the log files of that host. This results in another kind of DoS.

Generally, verification of address ownership [11] is effective against the redirect attacks of types 1 and 2 but can only partially stop attacks of type 3. The reason is that the attackers may flood others by redirecting data that was destined to itself.

The Sections 3.1 and 3.2 below identify specific threats for IPv6 network access. Redirect threats are described first, DoS attacks second. Finally, in Section 3.3 discusses the problems with using IPsec as a solution.

3.1 Redirect Threats

3.1.1 Malicious Last Hop Router

A malicious router on the local link can redirect all the traffic that is sent through it [7][16]. The attack proceeds as follows. An attacker masquerades as a last hop router by multicasting legitimate-looking Router Advertisements or unicasting Router Advertisements in response to multicast Router Solicitations. If a host selects the attacker as its default router,

the attacker has the opportunity to siphon off traffic from the host. Once accepted as a legitimate router, the attacker can send Redirect messages to the hosts, and then covering its tracks by disappearing.

3.1.2 Neighbor Solicitation / Advertisement Spoofing

An attacking node can cause packets for legitimate nodes, both hosts and routers, to be sent to some other link-layer address. This can be done by either sending a Neighbor Solicitation with a spoofed source link-layer address, or sending a Neighbor Advertisement with a spoofed target link-layer address. If the spoofed link-layer address is a valid one, as long as the attacker responds to the unicast Neighbor Solicitation messages sent as part of the Neighbor Unreachability Detection, packets will continue to be redirected.

This mechanism can be used for a DoS attack by specifying an unused link-layer address; however, the attack is of limited duration since after 30-50 seconds (with default timer values) the Neighbor Unreachability Detection mechanism will discard the bad link-layer address and multicast anew to discover the link-layer address. As a consequence, the attacker will need to keep responding with fabricated link layer addresses if it wants to maintain the attack beyond the time-out.

3.1.3 Spoofed Redirect Messages

The Redirect message can be used to redirect packets destined to a given IP address to any link-layer address on the link. The attacker uses the link-local address of the current first-hop router as the source address to send a Redirect message to a legitimate host. Since the host identifies the message by the link-local address as coming from its first hop router, it accepts the Redirect. As long as the attacker responds to Neighbor Unreachability Detection probes to the link- layer address, the Redirect will remain in effect.

3.2 DoS Threats

3.2.1 Bogus On-Link Prefix

An attacking node can send a Router Advertisement message specifying that some prefix of arbitrary length is on-link. If a sending host thinks the prefix is on-link, it will never send a packet for that prefix to the router. Instead, the host will try to perform address resolution by sending Neighbor Solicitations, but the Neighbor Solicitations will not result in a response, denying service to the attacked host.

The attacker can use an arbitrary lifetime on the bogus prefix advertisement. If the lifetime is infinite, the sending host will be denied service until removes the entry from its prefix list, e.g., because of state less during a reboot or because the same prefix is advertised with a zero lifetime. The attack could also be perpetrated selectively for packets destined to a particular address by using a 128 bit prefix in the advertisement.

3.2.2 Bogus Address Configuration Prefix

An attacking node can send a false Router Advertisement message and specify an invalid subnet prefix to be used by a host for address autoconfiguration. A host executing the address autoconfiguration algorithm uses the advertised prefix to construct an address [8], even though that address is not valid for the subnet. As a result, return packets never reach the host because the host's source address is invalid.

3.2.3 Duplicate Address Detection DoS

In networks where entering hosts obtain their addresses with stateless address autoconfiguration [8], an attacking node could launch a DoS attack by responding to every duplicate address detection attempt. If the attacker claims the addresses, then the host will never be able to obtain an address. This threat was identified in RFC 2462 [8] and an early attempt to solve the problem was made by Nikander [12].

3.2.4 Neighbor Discovery DoS Attack

In this attack, the attacking node fabricates addresses with the subnet prefix of the target network and continuously sends packets to them. The last hop router is obligated to resolve the addresses with the Neighbor Discovery protocol. A legitimate host attempting to enter the network may be unable to obtain Neighbor Discovery service from the last hop router as the router is already busy with resolving the bogus addresses. This DoS attack is different from the others in that the attacker may be off link. The resource being attacked in this case is the conceptual neighbor cache, which will be filled with attempts to resolve IPv6 addresses that have a valid prefix but invalid suffix.

3.2.5 Parameter Spoofing

Router Advertisements contain a few parameters used by hosts when they send packets and a flag to tell hosts whether or not they should perform stateful address configuration [7]. An attacking node could send out a valid-looking Router Advertisement that duplicates the Router Advertisement from the legitimate default router, except with parameter values that are selected to disrupt legitimate traffic. For example, an attacker could broadcast a false claim that the network in question uses DHCP for address configuration, which could cause other nodes to contact a nonexistent DHCP server and not get any publicly usable IP addresses at all.

3.3 Problems with IPSec Key Management

To enhance security and mitigate some of the attacks described above, the neighbor discovery messages may be protected with IPsec AH [7]. Potentially, AH could be used by the hosts to verify that Neighbor Advertisements and Router Advertisements do contain proper and accurate information. Given a suitable set of AH Security Associations (SAs), the host can verify that the ND messages it receives are really valid and authorized. An approach to define the required SAs through manual configuration was proposed by Arkko et al. [10]. The proposed mechanism is quite cumbersome due the large number of SAs needed.

Unfortunately, there is currently no other mechanism (but manual configuration) to provide such SAs. There are two basic reasons:

1. The only currently available automatic method for creating SAs is IKE. IKE requires a functional IP stack in order to function. Thus, a bootstrapping problem exists in using IKE in order to bring up an IP stack [10].

2. Even if some non-manual means of establishing SAs were available, it does not necessarily help in verifying the “ownership” of dynamically generated / assigned IP addresses [11][12].

4. CRYPTOGRAPHICALLY GENERATED ADDRESSES

In this section we briefly describe the idea of Cryptographically Generated Addresses (CGA). In Section 6 we show how they can be applied to mitigate a number of Neighbor Discovery related threats.

4.1 The basic idea

CGA was independently invented by O'Shea & Roe [17], Nikander [11][12], and Montenegro & Castelluccia [18]. Basically, it was recognized that 62 of the low order bits in an IPv6 address can be used to store a cryptographic hash of a public key. The basic mechanism can be defined as follows:

$$\text{host ID} = \text{HASH}_{62}(\text{public\_key}) \quad \text{Eq. 1}$$

The basic idea is sufficient to bind an address to a public key, and the reverse direction can be resolved with a conventional public key signature. However, sometimes it is beneficial to be able to claim ownership of an address without using public key cryptography. One possible way of achieving this is to generate a chain of hash values, as follows.

$$H_N := \text{HASH}_{160}(\text{public\_key} \mid \text{random})$$
$$H_i := \text{HASH}_{160}(\text{public\_key} \mid H_{i+1})$$
$$\text{host ID} := \text{HASH}_{62}(H_0) \quad \text{Eq. 2}$$

The host can show that it has generated the sequence H_0, ..., H_N, one by one, without needing to use public key cryptography. Thus, in the case of collision, both parties just reveal their H_1. Since the hash values H_i do not need to be just 62 bits long but can be e.g. 160 bits long, collisions become extremely unlikely, and the only reason why two hosts would reveal the same H_1 is that one of them has learned the value from the second. In that case the dispute can be resolved by revealing H_2. Going beyond i = 2 is a sign that an attack is occurring. [11][12]

In basic CGA, 62 bits are too few to gain strong security and real protection against brute force attacks, in particular birthday attacks where the attacker searches simultaneously for collisions with a large number of addresses. The hash chaining method does not change the effective hash length; other means are needed. Some possible ways to alleviate this situation are described next.

4.2 Breaking the limit of 62 bits

One possible way to effectively increase the hash length is to embed a security parameter into the address. The parameter determines the length of the hash function used. The security parameter, *Sec*, is a 2-bit unsigned integer encoded in the two rightmost bits of the 128-bit IPv6 address (*Addr*). This leaves just 60 bits for the actual hash output, but sacrificing the two bits is paid back by better performance scalability in the face of Moore's law.

In the revised scheme, first a 128-bit hash value is computed from the public key.

$$\text{Hash} = \text{HASH}(\text{public\_key}) \quad \text{Eq. 3}$$

A cryptographically generated address (CGA) is defined as an IPv6 address where the $64 + 20 \times Sec$ rightmost bits of the hash value equal the concatenation of $20 \times Sec$ zero bits and the interface identifier of the address. The two rightmost bits, which encode the security parameter, and the universal and group bits [6] are ignored in the comparison. The latter two bits must be set to one to indicate that the address is a CGA.

Using this scheme, the cost of creating a new CGA depends on the security parameter Sec, which can take on values from 0 to 3. If $Sec = 0$, a CGA can be created from the hash input with a straightforward algorithm that just computes a suitable hash and embeds it into the address. If an address collision is detected, the hash input is simply modified and the procedure is tried again. However, after three collisions, the algorithm should stop and report an error, since it is more likely that there is an DAD DoS attack going on (see Section 3.2.3).

In the general case, a CGA can be created as follows:

1. Select the security parameter $Sec = 0, 1, 2$ or 3.
2. Generate a suitable set of hash input values.
3. Execute the hash algorithm on the hash input.
4. Take the rightmost $64 + 20 \times Sec$ bits of the hash output and compare the leftmost $20 \times Sec$ of them to zero. If not zero, go back to step 2. (For $Sec = 0$, the comparison always succeeds.)
5. Concatenate the 64-bit routing prefix and the rightmost 64 bits of the hash output to obtain a 128-bit IP address.
6. Set the group and universal bits both to 1 and the two rightmost bits to *Sec*.
7. If an address collision is detected, go back to step (2). However, after three collisions, stop and report the error.

For security parameter values greater than 0, this second algorithm is nondeterministic, i.e., not guaranteed to terminate after a certain number of iterations. For security parameter values $1, 2$ and 3, it usually takes, respectively, in the order of 2^{19}, 2^{39} and 2^{59} iterations of the algorithm steps (2)-(4) to find a suitable hash input[1].

4.3 Binding addresses to locations

Above, we generated the host identifier by hashing the host's public key and nothing else. It is, however, possible to include other data items into the hash input.

We can discourage the brute-force and birthday attacks by binding the address to a specific network or hardware address. To do this, we simply include the network's route prefix or the host's link-layer address into the hash input. If neither of these two data items is included, the attacker could, at great expense, compute a lookup table that contains a suitable key pair for each of the 2^{60} or 2^{62} possible values of the interface identifier. Such a lookup table could be used to claim ownership of any IP address. A much smaller partial table could be used to find occasional collisions. Including the route prefix or hardware address prevents the attacks because the attacker would need to create a separate table for each network or for each individual network interface. While the attack may at the moment seem unlikely due to the processing and storage

[1] For a given parameter value Sec, the algorithm terminates when the hosts finds a set of input values that give an output that has $20 \times Sec$ leftmost zero bits. For any cryptographically good hash algorithm, the output values are randomly distributed. Thus, to find a set of input values that give the required number of leftmost zeros on the output, the host has to try, on the average, $2^{(20 \times Sec - 1)}$ different inputs.

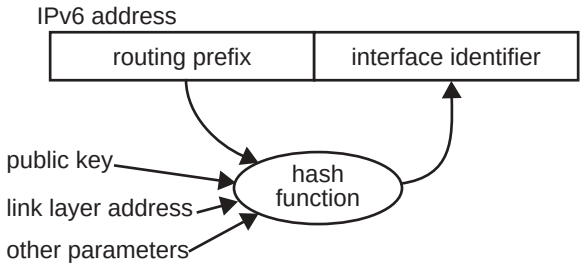

Figure 5. Generating GCA Addresses

requirements, it is imaginable that advances in processor and storage technology within the lifetime of the IPv6 protocol might make it feasible. A minor disadvantage of our defense is that the host must use a different host identifier with each of its routing prefixes or network interfaces. The basic method is illustrated in Figure 5.

The address owner could use the same technique to bind other security-related data to the address. For example, the hash input could contain flags that indicate the security protocols and algorithms supported by the host.

4.4 The CGA Format

To identify CGA addresses from non-CGA addresses, we propose that cryptographically generated IPv6 addresses are identified by a special bit combination in the interface identifier, i.e. the low 64 bits of an IPv6 address. Bit 6 of the interface identifier of the address is currently defined as the universal/local bit. Bit 7 of the interface identifier is the individual/group bit. (The leftmost bit of the interface identifier is numbered 0.) [6] The combination where both these bits are set to one should not currently be used in any IP addresses, and therefore it looks safe to suggest that they could be used to identify CGA addresses.

Encoding the address type into the address itself prevents the attacker from taking a CGA and presenting it as non-CGA. This way, both kinds of addresses can be used in the same protocol and the verifying node is able to securely differentiate between them. If the address type were communicated in a protocol message and not in the address bits, the attacker could claim that a CGA is a non-CGA and the verifying node would not be able to tell the difference.

An alternative would be to require all IP addresses, either in general or within a certain protocol or application, to be CGA addresses. In that case, it would not be necessary to reserve a combination of the universal and group bits as a type tag for CGA addresses. While this might work well in some applications, it is not as general solution as the type tag approach.

5. ADDRESS BASED KEYS

Addressed Based Keys (ABK) [19] use a cryptographic technique known as identity based cryptosystems. Identity based cryptosystems allow any publicly known identifier, such as an E-mail address or the IP address of a node, to function as the public key part of a public/private key pair. That is, basically any bit string may act as a public key. The trick lies in the way the corresponding private keys and a number of parameters are generated. For Addressed Based Keys (ABK), the IP address functions as the public key.

Identity based cryptosystems have been known to the cryptographic community for almost 20 years; however, they have not been widely discussed in the network security community. Possible reasons for this are the widespread use of standardized Diffie-Hellman based cryptography for network security applications, and, until recently, the lack of an efficient algorithm for identity based encryption.

Identity based cryptosystems work in the following way. A publicly known identifier is submitted to a trusted agent known as an Identity based Private Key Generator (IPKG). The IPKG uses a particular algorithm to generate the private key and returns it via a secure channel. The public and private keys can now be used for authentication and encryption as is typical in asymmetric cryptosystems.

In the key generation process, the IPKG uses a certain set of parameters. The parameters contain a master key, which is only known to the IPKG, and a number of other parameters that are known to all participants relying on the particular IPKG.

5.1 Identity Based Key Algorithms

There are many algorithms available for identity based cryptosystems. Shamir [20] introduced the idea of identity based cryptography in 1984. Practical, provably secure identity based signature schemes [21][22], and Key Agreement Protocols [23] soon followed. Practical, provably secure identity based encryption schemes [24][25] have only very recently been found.

In identity based signature protocols, the host signs a message using its private key supplied by its IPKG. The signature is then verified using the host's publicly known identity. In identity based key agreement protocols, two parties share a secret. Each party constructs the secret by using its own private key and the other party's public identity. In identity based encryption, the encryptor uses the recipient's public identity to encrypt a message, and the recipient uses its private key to decrypt the ciphertext.

As is generally the case with public-key cryptography, the security of the systems is based on the difficulty of solving a hard number theory problem, such as factoring or a solving discrete logarithm problem.

There are a number of advantages to using identity based systems that are based on elliptic curves and their pairings. One is that there are compatible elliptic curve-based signature, key agreement, and encryption schemes. This means, firstly, that the same public key / private key pair and public cryptographic parameters can be used to do signatures, key agreement, and encryption. Secondly, these protocols overlap, so that results of computations and pre-computations done for one system can be used in the others. Further, there are usually efficiency advantages in using elliptic curves, over using other public-key methods. Generally, one obtains shorter signatures, shorter ciphertexts, and shorter key lengths for the same security as other systems. Efficiency can be further enhanced by using abelian varieties in place of elliptic curves [26].

Techniques from threshold cryptography allow the master key information to be distributed or shared among a number of IPKGs so that all of them would have to collude for a host's private key to be known to them. Such a scenario would allow for key escrow if necessary, by agreement among all the

IPKGs, but guards against knowledge of the private keys by the IPKGs without their mutual agreement.

As a summary, identity based cryptosystems are fairly similar to other public key cryptosystems in practice. The major difference lies in key generation and distribution. Firstly, the parties cannot perform key generation themselves but must rely on the IPKG. Secondly, the public key distribution problem is easier since the parties need only to learn the parameters in a secure way, and the identifiers work directly as public keys.

5.2 Calculating Digital Signatures

Digital signatures for ABKs are calculated using the following algorithm:

sig =SIGN (hash (contents), IPrK, Params) Eq. 4

where:

sig	The digital signature.
SIGN	The identity based digital signature algorithm used to calculate the signature.
hash	A one-way hash algorithm, e.g. SHA1-HMAC.
IPrK	The Identity based Private Key.
Params	The public cryptographic parameters.
contents	The message contents to be signed.

The recipient verifies the signature in the following way:

IPuK = IBC-HASH (ID) Eq. 5

valid = VERIFY (hash (contents), sig, IPuK) Eq. 6

where:

IBC-HASH	A hash function specific to the identity based algorithm that generates the public key from the public identifier.
ID	The publicly known identifier used to generate the key.
IPuK	The Identity based Public Key.
sig	The digital signature.
VERIFY	The identity based public key algorithm used to verify the signature.
Params	The public cryptographic parameters.
valid	1 if the signature is verified, 0 if not.

5.3 Infrastructure Requirements

As the above algorithms illustrate, one essential ingredient for performing identity based cryptography in addition to the private key is a collection of *publicly known parameters* from the IPKG that generated the private key. The public parameters must be obtained somehow.

With some identity based algorithms, the IPKG maintains a copy of the private key, the so-called “key escrow” property. For general host security, this could indeed be a problem because the IPKG could impersonate any node for which it generated the private key. However, we consider ABK in only being used to secure IPv6 signaling traffic and not sensitive private data. Both the network operator and the legitimate client / user have an interest in seeing efficient operation of the network.

Most users today are happy to trust their ISPs with their credit card number or for accurately billing for services used. Thus, trusting their ISP to guard their ABK is probably of equal or lesser extent. There are also techniques that allow a group of IPKGs to generate the private keys without any one knowing the full key.

5.4 Certified addresses

As we mentioned in the end of Section 5.1, identity based algorithms are fairly similar to conventional public key cryptosystems from the practical point of view. Consequently, instead of using the addresses directly as public keys, one could just use a conventional public key cryptosystem and create certificates. In this section we briefly cover this alternative method.

Like ABK, address certification relies on a trusted agent. In this method, each node generates its own signature key pair. The node then co-operates with the trusted agent to generate 1–3 random host identifiers. For example, the host identifier can be produced by hashing together a random number R_{host} generated by the host itself and another random number R_{ttp} provided by the trusted agent.

$$\text{host ID} = \text{HASH}(R_{host} | R_{ttp}) \quad \text{Eq. 7}$$

Finally, the trusted agent signs a certificate that binds the host identifier to the host’s public key. This can be an X.509 certificate where the host identifier is used as the entity name. If the host has more than one host identifier, each one is certified separately.

After entering a network, the host creates an IP address by concatenating a route prefix with a certified identifier. It can then use the certificate and its private signature key to prove its ownership of the IP address. For example, it can sign neighbor discovery messages with the key. If an address collision occurs, the host can try again with another host identifier.

An advantage of the certified addresses is that implementations can use standard public-key signatures and certificates, and existing cryptographic libraries. A disadvantage is that the certificate must be sent and verified with each signature. The key-based addresses, on the other hand, do not need any certificates. Both techniques rely on a trusted agent that is shared by all network nodes.

6. SECURING NEIGHBOR DISCOVERY

CGA looks like a very promising method for securing Neighbor Discovery. As initially described by Nikander [12], CGA can be used to prove that the answer comes from the owner of the address, i.e., from the node that generated it. This is done by signing the messages with the key that was used to generate the address. This technique can be used for signing Address Resolution, Duplicate Address Detection, and Redirect messages. While ABK and certified addresses could be used for the same purpose, the advantage of CGA is that it requires no trusted entities. On the other hand, CGA does constraint the choice of the IP address while ABK does not. Therefore CGA would not be appropriate when the interface identifier is dictated by other constraints.

6.1 Address Resolution

An easy way to secure address resolution is to include the public key corresponding to the CGA address and a signature in all NA and ND packets. If any other parameters were used to generate the CGA, they must also be sent, so that the recipient can verify the hash value.

The node receiving a NA or ND can either verify the address immediately or it can cache the messages for later processing. If the node defers the verification until it actually is going to use the address, it can avoid some denial-of-service threats.

The verifier first recomputes the hash of the public key and compares the result with the interface identifier. Then, it verifies the signature on the message using the public key. If both checks succeed, the verifier can add the address into its address resolution cache and discard the verification data.

6.2 Duplicate Address Detection

As only the owner of an address can respond with a verifiable signature all attempts at spoofing answers to DAD queries are prevented. However, depending on the local link, it may still be possible to prevent valid messages from reaching the intended recipients.

In the stateless autoconfiguration of IPv6 addresses, it is possible that an address collision is detected and a new address needs to be created. However, it is very unlikely that an address collision will occur except as the result of an attack on the protocol. Three collisions in a row are very, very unlikely to occur by chance, and are almost certainly the result of either an attack on the protocol or an error in the implementation. Therefore, no further addresses should be tried after three consecutive collisions.

6.3 Securing Redirect

Routers use CGA to prove that the Redirect messages come from the claimed IP address. Hosts verify this, and ensure that the IP address is the same as could have been used for the next hop. This procedure says nothing about the other router mentioned in the redirect, but that may not be necessary. However, we could cross-certify routers (hosts don't need to be involved), thereby allowing the hosts to verify that both the source of redirect and the destination of redirect are among the cross-certified set of nodes. That makes the Redirect attack much harder for a potential attacker, even though it may not close all related vulnerabilities.

7. SECURING ROUTER DISCOVERY

While CGA seems like a suitable mechanism for securing most Neighbor Discovery functions, it alone is not sufficient for Router Discovery. ABK and certified addresses, on the other hand, can convey information about an authorization made by a third party. Therefore, they are more directly applicable to Router Discovery.

7.1 Extending CGA

The CGA signatures can be used to prove that a message is authentic, i.e., the message was signed by the owner of the IP address from which it claims to come. But even if a message has a valid CGA signature, the contents of the signed message cannot be trusted blindly; the address owner could be lying. For the above reason, CGA cannot be used to prove that the address owner is a router or that it is authorized to advertise a specific route prefix. Such authorization must come from a third party.

One solution is not to verify the router's authorization at all. Instead, one possibility is to use a heuristic that can be summarized as follows: "if it routes like a router, it probably is a router". The task of a gateway router is to route packets between hosts in the local network and arbitrary Internet destinations. The host can test this.

In practice, the test is performed by sending a packet to a remote Internet destination and receiving a timely reply. It is essential that the packet and the reply, in particular the source and destination addresses, are cryptographically protected and bound to each other. For example, an ICMP echo message protected with a pair of ESP SAs is sufficient, but an unprotected echo is not.

7.2 Using ABK

To secure the IPv6 Router Advertisement, ABK can be used to sign and authorize routing prefixes. In practice, 64 bit subnet prefixes can be used as public keys and the private keys are derived from the same bits. The private keys are then used to compute a signature that is included in Router Advertisements.

Thus, in principle, an ABK based digital signature must be included in all Router Advertisement messages. To make the scheme secure, the public parameters must be securely indicated so that the other nodes in the network can verify the signatures. We suggest that the parameters can be configured using the same secure protocol that is used to configure their private keys. The routers in a network can be manually configured with both the parameters and the private keys. Note that the private keys a router have correspond to the routing prefixes it is authorized to legitimately advertise.

Two possible approaches for hosts have been proposed [19]:

1. The network provides the host with the parameters and a private key when the host performs a secure layer 2 authentication and authorization procedure to enter the network and obtains its IP address. This approach would be particularly suitable for a public access network.
2. Hosts are preconfigured with the parameters along with their private keys. When they roam off their home networks, they use a particular protocol to identify to what roaming consortium they belong. A roaming consortium shares identity based key generation and thus the public parameters.

In the suggested method, an ICMP option is used to carry the signature, instead of IPSec AH, because options that are not recognized by a host are ignored. Consequently, a host that can't verify the signature but is interested in risking using an unsecured Router Advertisement can simply ignore the ICMP option.

To authorize routing prefixes, the Router Advertisement message is limited to carry a single Prefix option, with the prefix for which the key was assigned. If the router also routes other prefixes, it must advertise them using separate Router Advertisements.

An IPv6 host receiving a Router Advertisement with an ICMP signature option verifies that the advertising node is authorized to send the advertisement in the following way. The host locates the single routing prefix option and extracts the subnet prefix which the sending node claims it is allowed to route. The host then uses the ABK verification algorithm to verify the digital signature. In this calculation, the ABK public key is the subnet prefix in the prefix option.

Thus, if the host trusts the public key parameters, it can verify that the node that sent the Router Advertisement indeed possesses a private key corresponding to the routing prefix. Since such private keys are supposed to be available only to legitimate routers, the signature proves that the Router Advertisement message is intact and sent by a legitimate router.

8. CONCLUSIONS

In this paper we have described a number of threats pertinent to current IPv6 Neighbor and Router Discovery, discussed two new cryptographic techniques, Cryptographically Generated Addresses (CGA) and Address Based Keys (ABK), and briefly described how these can be used to secure the Neighbor and Router Discovery functions.

The basic idea in CGA is to generate most of the 64 low order bits in an IPv6 address as a cryptographic hash over a public key and other parameters. The underlying cryptosystem can be any public key cryptosystem, such as RSA, DSA, or Elliptic Curve based DSA. ABK uses either the low order bits of the address or all the bits of a routing prefix as a public key, relying on an identity based cryptosystem. Together these two methods can be used to secure Neighbor Discovery in a way that does not require any explicit security infrastructure, and Router Discovery with minimal key distribution effort.

At the current stage, however, the results are still preliminary. For example, more analysis is needed on whether using 62 or 60 bit hashes is secure enough, and whether the proposed CGA security parameter is the right choice to improve security. That is, the reason for having the security parameter is that a hash length of less than about 90 bits is insufficient to prevent a massive brute-force attack against an individual address, even with today's technology. Our proposed solution increases the cost of both the attack and the address generation, which is not as satisfying as it would be to increase only the cost of the attack. Nevertheless, it looks like an effective means of solving the problem of exponential growth in computational capacity and would significantly lengthen the expected lifetime of CGA-based protocols.

Furthermore, it is essential to encode the security parameter as well as the address type into address bits. This may create further operational and other complications. On the other hand, CGAs with different security parameter values can be accepted in the same protocol and the verifier can differentiate between them. If the security parameter were communicated in a protocol message and not encoded into the IP address, an attacker could misrepresent the values and attack a weaker mechanism than the one selected by the address owner.

The ABK ideas need some more scrutiny as well before they can be considered mature enough for practical use. In particular, ABK requires additional work on key distribution and hierarchical key management. A hierarchical key management algorithm has been recently described [27], and this would be appropriate for a collection of ISPs in a roaming consortium.

In general, however, we believe that the presented methods, once matured and stabilized, present a number of benefits over traditional key management methods. CGA can be used in ad-hoc and other infrastructure lacking networking environments. Different sets of ABK parameters may be maintained with relatively little infrastructure and simple trust relationships, potentially leading to operational benefits when compared to alternatives, such as traditional public-key certificates.

Finally, it should be noted that there may be DoS and other attacks against the lower layers that are as serious as the ones described in this paper. Thus, the security measures proposed in this paper are really effective only if the lower protocol layers are sufficiently protected or if the lower-layer attacks are considered unlikely or prohibitively expensive.

Acknowledgements

We want to thank Craig Gentry and Alice Silverberg for their co-operation and work on defining the details for ABK, the anonymous reviewers for their fruitful suggestions on how to improve this paper, and Erik Nordmark for participating in performing the threat analysis.

References

[1] S. Deering and R. Hinden, *Internet Protocol, Version 6 (IPv6) Specification*, RFC2460, Internet Engineering Task Force, December 1998.

[2] A. Conta and S. Deering, *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification,* RFC2463, Internet Engineering Task Force, December 1998.

[3] *IEEE Draft P802.1X/D11: Standard for Port based Network Acces*s Control, LAN MAN Standards Committee of the IEEE Computer Society, March 27, 2001.

[4] IEEE Std. 802.11i/D2.0, *Draft Supplement to IEEE 802.11 Standard: Specification for Enhanced Security,* March 2002.

[5] A. Mishra and W. A. Arbaugh, "An Initial Security Analysis of the IEEE 802.1X Standard", UMIACS-TR-2002-10, University of Maryland, February 2002.

[6] R. M. Hinden and S. E. Deering. IP version 6 addressing architecture. RFC 2373, IETF Network Working Group, July 1998.

[7] T. Narten, E. Nordmark and W. Simpson, *Neighbor Discovery for IP Version 6 (IPv6),* RFC2641, IETF, December 1998.

[8] S. Thomson and T. Narten, *IPv6 Stateless Address Autoconfiguration,* RFC2462, Internet Engineering Task Force, December 1998.

[9] T. Narten and R. Draves. Privacy extensions for stateless address autoconfiguration in IPv6. RFC 3041, IETF, January 2001.

[10] J. Arkko, P. Nikander, T. Kivinen, and M. Rossi, *Manual SA Configuration for IPv6 Link Local Messages,* work in progress, draft-arkko-manual-icmpv6-sas-01.txt, June 2002.

[11] P. Nikander, "Denial-of-Service, Address Ownership, and Early Authentication in the IPv6 World," presented at *Cambridge Security Protocols Workshop 2001,* April 25-27, 2001, Cambridge University.

[12] P. Nikander, "A Scalable Architecture for IPv6 Address Ownership", unpublished manuscript, available at http://www.tml.hut.fi/~pnr/publications/draft-nikander-ipng-pbk-addresses-00.txt, March 2001.

[13] D. Thaler and J. Hagino, "IPv6 Stateless DNS Discovery", draft-ietf-ipv6-dns-discovery-04.txt, work in progress.

[14] Steven Bellovin, "Security Problems in the TCP/IP Protocol Suite", Computer Communication Review, Vol. 19, No. 2, pp. 32–48, April 1989.

[15] J., Kempf and E. Nordmark, “Threat Analysis for IPv6 Public Multi-Access Links,” draft-kempf-netaccess-threats-00.txt,work in progress.

[16] Mankin, et. al., “Threat Models introduced by Mobile IPv6 and Requirements for Security in Mobile IPv6,” draft-ietf-mobileip- mipv6-scrty-reqts-01.txt, work in progress.

[17] G. O'Shea and M. Roe, *Child-proof authentication for MIPv6 (CAM)*. Computer Communications Review, April 2001.

[18] G. Montenegro and C. Castellucia, “SUCV Identifiers and Addresses,” draft-montenegro-sucv-02.txt, work in progress.

[19] J. Kempf, C. Gentry, and A. Silverberg, “Securing IPv6 Neighbor Discovery Using Address Based Keys (ABKs),” draft-kempf-ipng-secure-nd-00.txt, work in progress.

[20] A. Shamir, “Identity-Based Cryptosystems and Signature Schemes”, Advances in Cryptology –Crypto'84, Lecture Notes in Computer Science 196, (1984), Springer, 47-53.

[21] A. Fiat and A. Shamir, “How to prove yourself: Practical solutions to identification and signature problems”, Advances in Cryptology – Crypto'86, Lecture Notes in Computer Science 263, 1986), Springer, 186-194.

[22] U. Feige, A. Fiat, and A. Shamir, “Zero-knowledge Proofs of Identity”, Journal of Cryptology 1, (1988), 77-94.

[23] U. Maurer and Y. Yacobi, “Non-interactive public-key cryptography,” Advances in Cryptology – Eurocrypt'92, Lecture Notes in Computer Science 658,(1993), Springer, 458- 460.

[24] D. Boneh and M. Franklin, “Identity based encryption from the Weil pairing”, Advances in Cryptology – Crypto 2001, LNCS 2139, (2001), Springer, 213-229, http://www.cs.stanford.edu/~dabo/papers/ibe.pdf

[25] C. Cocks, “An identity based encryption scheme based on quadratic residues”, http://www.cesg.gov.uk/technology/id-pkc/media/ciren.

[26] A. Silverberg and K. Rubin, “Supersingular abelian varieties in cryptography”, Cryptology e- Print Archive Report 2002/006, http://eprint.iacr.org/2002/006/, Advances in Cryptography – Crypto 2002, Springer, 2002.

[27] C. Gentry and A. Silverberg, “Hierarchical ID-based Cryptography,” Cryptology e-Print Archive Report 2002/056, http://eprint.iacr.org/2002/056/.