SEcure Neighbour Discovery: A Report

Arun Raghavan (Y6111006)

CS625: Advanced Computer Networks

Abstract

The IPv6 [5] Neighbour Discovery [12] protocol is used by nodes in IPv6 for such purposes as discover neighbours on the link and discovery of routers and configuration parameters . The protocol assumes that all nodes on the link can be trusted to adhere to the protocol. This assumption does not hold in a number of scenarios. In this review, we talk about a number of possibly attacks on the IPv6 Neighbour Discovery protocol, and the solutions proposed by SEcure Neighbour Discovery [2].

1 The Problem

The IPv6 Neighbour Discovery protocol is used by for several important purposes:

- Router discovery
- Discovery of network prefix and parameters
- Address resolution of on-link neighbours
- Unreachability detection of on-link neighbours

It is clear that these functions are critical to the functioning of a node. The original specification of this protocol (RFC 2461) does not specify how this protocol is to be secured from malicious nodes. The specification glosses over the topic by suggesting the use of the IP Authentication Header [10]. However, it does not specify how security associations can be set up between nodes. Quoting the RFC – "The security associations may have been created through manual configuration or through the operation of some key management protocol."

Manual security associations are clearly not a feasible solution, from both the scalability and administration points of view. The standard key management protocol for this use is Internet Key Exchange (IKE) [6]. Unfortunately, IKE can not be be used for Neighbour Discovery, because it suffers from a chicken-and-egg problem [1]. Simply stated, to securely get an IP address, the node needs to have a security association with the key server, but for this to get this security association, the node first needs to have an IP address. Solutions such as IKE are thus ruled out.

The SEcure Neighbour Discovery (SEND) protocol was developed to address this problem. This report is divided into the following sections: we talk about the threat models and attacks that are possible on the Neighbour Discovery (ND) protocol as defined in RFC 2461. We then examine some novel solutions that allow nodes to bind their identities to their addresses cryptographically. Next, we see how SEND uses these, and other techniques to secure Neighbour Discovery. Finally, we visit some of the shortcomings of SEND, and possible solutions.

2 Security in Neighbour Discovery

2.1 Trust models

We can broadly divide into the following types of networks in order to derive a set of possible trust models under which we consider attacks on Neighbour Discovery [14].

2.1.1 Completely managed networks

A typical example of this type of network would be a corporate intranet. All the machines on such a network are controlled by a central authority, and network access points can generally be protected by both physical security and access control mechanisms such as 802.1X [7] and 802.11i [8]. However, it might be desirable to secure these networks against the attacks described in the next section in order to protect the network from compromised nodes. One alternative is to use public-key cryptography using a preconfigured authority for public-key distribution. Another is to use SEND.

2.1.2 Managed router

This class of networks is best represented by a public wireless network with a network operator managing the routers and infrastructure. Clients connect to the network, possibly after being authorised by the operator. Typically, the routers are in a separate authorisation domain from the client nodes (although an attacker might introduce a malicious router).

An important point to note is that the use of access control and encryption schemes at the link-layer level might not be sufficient to mitigate all the threats to be discussed. For example, the link-layer topology might not be the same as the IP layer topology.

2.1.3 Ad-hoc networks

These are completely ad-hoc networks in which there is no trusted operator and none of the nodes trust each other. Since no prior trust relationships can be assumed, this is a hard problem to solve. Some of the methods of cryptographically binding an IP to a node can be used to mitigate a number of attacks.

2.2 Attacks

There is a wide range of attacks that can be launched against the Neighbour Discovery protocol.

2.2.1 Attacks on Neighbour Discovery

Neighbour Solicitation/Advertisement spoofing A node can reroute packets for a target node by spoofing the link-layer address of that node. To do this, the attacker merely needs to send a neighbour solicitation/advertisement with a source/target with the IP address of the victim and link-layer address that is different from that of the victim. This can have a number of effects. Firstly, the attacker can cause a Denial of Service (DoS) by providing an invalid link-layer address. Secondly, the attacker can intercept all messages between two nodes by this method, potentially without the knowledge of either victim. The reason these attacks work is that victim nodes update their Neighbour Cache, which binds link-layer addresses to IP addresses when they receive spoofed packets (which they have no way of verifying).

Neighbour Unreachability Detection failure Neighbour Unreachability Detection (NUD) messages are used by a node to check if a node that it is communicating with is still online. This is done by sending a Neighbour Solicitation (NS) message to the destination. If a corresponding Neighbour Advertisement (NA) is received, the source node knows that it's destination is accessible. An attacker can perform a DoS attack on a node whose destination has gone offline by sending spoofed NA messages as the destination. This attack would be mitigated if the victim node had way to verify the sender of the NA.

Duplicate Address Detection DoS If a network uses stateless autoconfiguration for assigning addresses to nodes, each node selects an address and then sends out a Neighbour Solicitation for that address to ensure that there is no collision. A malicious node could respond to these solicitations with spoofed advertisements, thus denying the victim access to the network. Again, solving this problem would require a way for a node to authorise its use of an address.

2.2.2 Attacks on routing

Malicious Last Hop Router In this attack, a malicious node could advertise itself as the last hop router, thus intercepting all packets that it chooses to. It can also disrupt connectivity by dropping some or all packets. To solve this, we require is a way to verify that a router is authorised to perform routing on the network. This can clearly pose a problem for ad-hoc networks. The ad-hoc network case is an open research problem.

Kill the Default Router RFC 2461 specifies that if the default router list of a node is empty, it must assume that all nodes are on-link. Thus, if an attacker can make a node believe that all routers in this list are unavailable, the node may broadcast all packets on the link, allowing the receiver to intercept them. Another method of accomplishing this is for the attacker to advertise a malicious node as a router and then make the victim believe that all other routers are unavailable. This can be done by performing a DoS attack on the other routers. While the second case can be protected against by having a way to authorise routers, there is no known solution to the first attack.

Compromise of a router If an attacker compromises a router, (s)he gains access to all packets going through that as well as the ability to disrupt routing. While it might be possible to have a solution that allows revocation of "routing rights" on a network, this is still an open problem in the general case.

Spoofing of Redirect Messages Redirect messages are used by routers to inform nodes of a better first-hop router. An attacker can fabricate such messages and thus usurp routing from the current router. This could be prevented by having a method of authorising routers and/or a way to identify that a packet has indeed originated from the IP address specified in a packet (a node only accepts Redirect messages from its current first hop router).

Other Bogus Router Advertisements A malicious node can advertise itself as the router for any arbitrary set of prefixes, thus fooling a node into believe that even certain or all off-link nodes are actually on-link. It could also advertise address incorrect network parameters, or an invalid subnet prefix. The first allows the node to intercept some or all packets sent to off-link nodes. The second allows DoS attacks, as well as the ability to usurp DHCPv6 services. The last is a DoS attack. Again, having a means to identify authorised routers would solve these problems.

2.2.3 Remote/Replay attacks

Replay Attacks Even if solicitations and advertisements are cryptographically protected, they must also be protected from replay attacks (the packet is stored for an arbitrary amount of time and then reinjected into the network, possibly when its contents are no longer valid). These are usually solved by nonces and timestamps (assuming loose time synchronisation between nodes).

Neighbour Solicitation DoS This is a unique attack in this list in that it can be executed from outside the network. The idea is to force the host at the ingress point of the network to send a large number of neighbour solicitations. If most of these are invalid, the node must hold the pending state of these solicitations till a timeout occurs. This may also be launched on nodes on the network by tricking an application into sending packets to arbitrary on-link destinations. This is similar to the TCP SYN flooding attack, can be similarly solved by enforcing rate-limiting on outstanding neighbour solicitations.

2.2.4 Commonalities

For most of the attacks described above, we see two common problems. The first is that there is currently no way to verify that a given message was sent from the node specified in the source address field of the packet. The second is that we have no way to identify "authorised" routers. The first could be tackled by cryptographic means, but it would be desirable to do so without introducing dependence on any kind of infrastructure (for the ad-hoc case). The second problem can also be solved using cryptography, but is open in the ad-hoc network case, since ad-hoc routing schemes assume that any node may participate in the routing process.

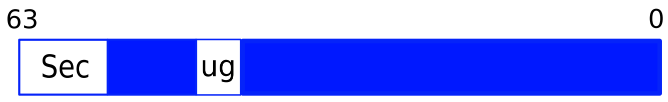

Figure 1: The CGA Format

3 Securing Neighbour Discovery

3.1 Cryptographically Generated Addresses

There have been several schemes proposed to solve the problem of binding a node and its address in the absence of the infrastructure traditionally used for this purpose [11, 13, 3]. SEND uses Cryptographically Generated Addresses (CGAs) [3] to achieve this purpose.

The idea behind CGA (and most of the other schemes used for this purpose) is to for each node to generate a public-private key-pair for asymmetric cryptography. The address that a node takes is generated by taking a one-way hash of its public key. Messages from this node can now be protected by signing messages with the node's private key. The public key can be distributed with every packet for verification at the destination. The destination first verifies that the public key provided hashes to the address from which the packet was received, and then verifies the signature in the packet.

Assuming that the asymmetric cryptographic scheme used is computationally infeasible to break, an attacker would need to generate a public-private key-pair which corresponds to the same address as its target. Given a long enough address one-way hash function, this too is computationally infeasible for an attacker. CGAs use a method called "hash-extension" to make this more expensive than even $O\left(2^{addrlen}\right)$.

The format of an address derived using the CGA scheme is shown in Figure (1). This is used as the interface identifier part of the IPv6 address. *Sec* is a 3-bit parameter that determines the cryptographic strength of the CGA. The u and g bits are used in IPv6 addresses to denote that the address is globally unique and for multicast respectively. These bits are ignored for our purposes. The remaining 59 bits are used for addressing

3.1.1 Generating a CGA

Before getting an address, every node first generates a public-private RSA key-pair. The procedure used in outlined below. The notation $MSB_x(y)$ denotes the most significant x bits of y. The 3-bit *Sec* parameter is assumed to be configured, and the subnet prefix can be discovered by the neighbour discovery process.

1. Generate a random 128-bit modifier, set collision count to 0, set subnet prefix to 0
2. Build the CGA Parameters structure as shown in Figure (2). The public key is used as a DER-encoded ASN.1 structure.

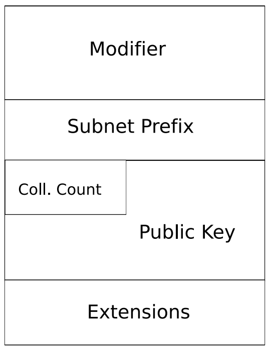

Figure 2: The CGA Parameters structure

3. $Hash_1 = MSB_{16*Sec}\left(SHA1\left(CGA\,Parameters\right)\right)$
4. If $Hash_1 \neq 0$, increment modifier and goto 2
5. Set the subnet prefix and collision count in the CGA Parameters structure above
6. $Hash_2 = MSB_{64}\left(SHA1\left(CGA\,Parameters\right)\right)$
7. Create an interface identifier by setting the first 3 bits of $Hash_2$ to Sec and bits 6 and 7 to 1. Append this to the subnet prefix to get the address.
8. Do Duplicate Address Detection for this address
9. If there is a collision, increment collision count. If this is more than 2, return with an error, else goto 5

We now have a CGA and a CGA Parameters structure.

3.1.2 Verifying a message

The CGA Parameters structure is sent with very message that needs to be verified.

1. Verify that the collision count is less than 3
2. The subnet prefix of the sender and that in the CGA Parameters structure should match
3. Generate $Hash_2$ as during generation and verify that it creates the same interface identifier as the address from which the packet was received
4. Generate $Hash_1$ and ensure that the leftmost $16 * Sec$ bits are 0.

3.1.3 Cryptographic Strength of the CGA

It is clear that the use of $Hash_2$ to generate the address provides a reasonable amount of security to the CGA. To forge a packet from a given node, an attacker would need to generate a public-private key-pair that generates the same CGA. Ignoring $Hash_1$ for the moment, a brute-force attack will take $O\left(2^{59}\right)$ time to discover such a collision. While this is quite strong today, the SEND Working Group wished to ensure that the CGA scheme was reasonably future-proof. Given the progress of computational capacity, this brute force attack might become feasible in 5-10 years.

To attack a CGA generated under the scheme above, an attacker would have to find a key-pair that generates the same 64 bits of $Hash_2$ as the CGA, as well as the same first $16 * Sec$ zero bits in $Hash_1$. Clearly, the complexity of a brute-force attack on this is $O\left(2^{59+16*Sec}\right)$. Thus, to increase the strength of the CGA, we merely need to increase the value of the Sec parameter. This mechanism is called "hash-extension".

One point to note is that address generation is now $O\left(2^{16*Sec}\right)$. However, this is not expected to be performed too often, even for mobile nodes, since the subnet prefix is not used in steps 1-4. Verification is, of course, $O(1)$.

3.2 The SEND Protocol

The SEND protocol defines two major changes to the standard ND protocol. The first is the use of "options" which are actually used to implement the secure extensions to ND. These are sent with every ND packet, and solve the problem of ensuring that packets actually originate from the source address in the packet. The second is the concept of authorisation of routers. This is used to distinguish trustworthy routers.

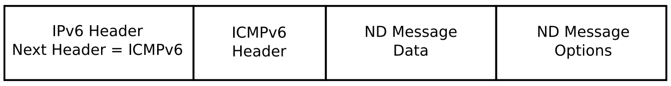

Figure 3: ND Options

3.2.1 Options

SEND acts as a set of extensions to the existing Neighbour Discovery protocol. To achieve this, SEND specifies "options" that are appended to the regular ND messages. This is illustrated in Figure (3).

The options introduced are:

CGA Option This is merely the CGA Parameters structure created at CGA generation time, with some padding bits added.

RSA Signature Option This option is used to ensure message integrity. The option includes a key hash that the receiver can match with the sender's public key (in case a sender has multiple keys). The digital signature includes several IP and ICMP fields, the NDP message, and all options preceding the signature option. Since signature verification and computation can both be computationally heavy operations, an attacker could launch a DoS attack by sending a node too many packets to verify, or by causing it to send out too many signed messages. This can be tackled by enforcing a rate-limiting policy on sending/receiving signed messages.

Timestamp Option A timestamp is attached to unsolicited advertisement messages. This ensures that an attacker does not capture advertisements, and then replay them when the information is no longer valid. It requires that nodes have some loose time synchronisation mechanism (such as NTP). The timestamp contains the time elapsed since Jan 1^{st}, 1970, 00:00 UTC. 48 bits are used for seconds, and 16 bits for $\frac{1}{64K}$ seconds. The RFC 3971 defines some parameters for adjusting the permissible drift in sender and receiver clocks. The RSA Signature Option ensures that the timestamp cannot be tampered with.

Nonce Option This option is used to secure solicited advertisements. The solicitation message contains a random number, called a "nonce", at least 6 bytes long. The corresponding advertisement (whether unicast or multicast) *must* contain the same nonce in the Nonce Option. Since the nonce is large, it is unlikely that two messages in a reasonable time period will carry the same nonce. This protects the sender from replay attacks. The RSA Signature Option protects messages in both directions from tampering.

3.2.2 Router Authorisation

SEND mandates the use of certificates and trust anchors to identify trusted routers. Every node is configured with a set of "trust anchors". These might be globally defined entities (there is no current infrastructure for this, but it can be similar to the mechanism used for SSL CAs), or local to an organisation. When a node receives a message from a router, it needs to establish a "certification path" from the trust anchor to this router, authorising this router for the subnet in which it is operating.

A trust anchor can delegate the authority of providing routing for a given subnet (or all subnets it has authority for) to another router, by providing a digital certificate to the router for this purpose. This router, in turn, can further delegate another router for part or all of it's subnet, using the same mechanism. Client nodes, being configured with the public keys of trust anchors, can verify this certification path and thus verify that a given router is authorised to provide routing for a given subnet. This process is called Authorisation Delegation Discovery. These certificates can also provide a public key for the router. As with any other certification mechanism, nodes must maintain regularly updated revocation lists.

It was shown in the previous section that an attacker cannot easily spoof packets. Introducing a malicious router into the network is also very difficult, because an attacker would need to procure authorisation by means of a certificate, and these are controlled by the network operator. Routers might need to send a large amount of data in this scheme (the number of certificates might be large), so rate-limiting should be used to prevent network flooding.

3.2.3 Certification Path Discovery

RFC 3971 specifies a Certification Path Discovery protocol for nodes to discover certification paths from known trust anchors to routers that they wish to use. The RFC does *not* mandate the use of this protocol for discovering certification paths. It also mentions that this scheme can also be used for host authentication.

Certification Path Solicitation This is sent by a node to a router to request a certification path. The node may specify one or more trust anchors that it can derive certification paths from. The message is protected by a nonce. If the node knows the identifier for a particular certificate from a router, it can also add that to the solicitation message.

Certification Path Advertisement A router sends out a Certification Path Advertisement in reply to Certification Path Solicitation messages. The advertisement contains the number of certificates in the path, and then one or more certificates. It is recommended that each certificate be sent in a separate message to avoid fragmentation. The certificates should be sent starting from the certificate signed by the trust anchor and ending with the certificate for the router sending the advertisement. This allows the node to start validating the certification path before it receives all certificates. Each certificate is sent as a Certificate Option in the X.509 format. The advertisement must include the nonce received in the solicitation. A router might also add one or more Trust Anchor Options, particularly in broadcast advertisements, so that nodes can decide up front which advertisements they are interested in (depending on what trust anchors they support).

3.3 Transition

The specification also mentions a number of points with regards to transition from traditional ND to SEND. The noteworthy recommendations are:

- Nodes which support SEND should only send signed messages
- SEND routers should be preferred over non-SEND routers
- An unsecured ND message should not override data that was derived from a secured message

4 Conclusion

The SEND protocol proposes a cryptographic means of securing the Neighbour Discovery protocol with very little additional infrastructure. The protocol should be applicable in most scenarios except the pure ad-hoc network case. Encryption and link-layer protection are not in the purview of SEND, however.

One significant limitation of SEND is that it does not support Proxy ND [4]. This is used for Mobile IPv6 [9] and when two LANs are connected by transparent bridges. One suggestion is for a mobile node to use its own public-private key-pair to issue a authorising certificate to the proxy. This is just a suggestion, and a final solution is yet to be published.

There is currently an implementation of SEND provided by DoCoMo Labs, but there does not seem to be a practical implementation or use of the protocol in place yet. It would be interesting to determine the impact of the extensive use of asymmetric cryptography in the protocol, since asymmetric cryptography is usually quite computationally expensive. This might be alleviated in the future with the widespread availability of hardware peripherals for cryptography.

References

[1] J. Arkko, "Effects of ICMPv6 on IKE," Internet Draft, March 2003.

[2] J. Arkko, J. Kempf, B. Zill, and P. Nikander, "SEcure Neighbor Discovery (SEND)," Internet Engineering Task Force: RFC 3971, March 2005.

[3] T. Aura, "Cryptographically Generated Addresses (CGA)," in *ISC*, ser. Lecture Notes in Computer Science, C. Boyd and W. Mao, Eds., vol. 2851. Springer, 2003, pp. 29–43.

[4] G. Daley, "Securing Proxy Neighbour Discovery Problem Statement," Internet Engineering Task Force: Draft, August 2005.

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

[6] D. Harkins and D. Carrel, "The Internet Key Exchange (IKE)," Internet Engineering Task Force: RFC 2409, November 1998.

[7] "IEEE Standard for Local and metropolitan area networks: Port-Based Network Access Control," IEEE, New York, November 2004.

[8] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Medium Access Control (MAC) Security Enhancements," IEEE, New York, June 2004.

[9] D. Johnson, C. Perkins, and J. Arkko, "Mobility Support in IPv6," Internet Engineering Task Force: RFC 3775, June 2004.

[10] S. Kent and R. Atkinson, "IP Authentication Header," Internet Engineering Task Force: RFC 2402, November 1998.

[11] G. Montenegro and C. Castelluccia, "Statistically Unique and Cryptographically Veriable (SUCV) Identifiers and Addresses," in *Network and Distributed System Security Symposium Conference Proceedings: 2002*, February 2002.

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

[13] G. O'Shea and M. Roe, "Child-proof authentication for mipv6 (cam)," *SIGCOMM Comput. Commun. Rev.*, vol. 31, no. 2, pp. 4–8, 2001.

[14] E. N. P. Nikander, J. Kempf, "IPv6 Neighbor Discovery (ND) Trust Models and Threats," Internet Engineering Task Force: RFC 3756, May 2004.