Chapter 8
SEcure Neighbor Discovery: A Cryptographic Solution for Securing IPv6 Local Link Operations

Ahmad AlSa'deh
Hasso-Plattner-Institute, Germany

Hosnieh Rafiee
Hasso-Plattner-Institute, Germany

Christoph Meinel
Hasso-Plattner-Institute, Germany

ABSTRACT

SEcure Neighbor Discovery (SEND) was proposed to counteract threats to the Neighbor Discovery Protocol (NDP). It is a strong security extension that can make the IPv6 local link very safe. SEND relies on dynamically Cryptographically Generated Addresses (CGAs) and X.509 certificates. However, SEND is not easily deployed and is still vulnerable to some types of attacks. This chapter evaluates the practical considerations of a SEND deployment taking a cryptographic approach as a means of securing the IPv6 local link operations. It reviews the remaining vulnerabilities and gives some recommendations with which to facilitate SEND deployment.

INTRODUCTION

The free pool of IPv4 address space will be depleted soon. On 3 February 2011, the Internet Assigned Numbers Authority (IANA) (2012, March 14) allocated the last remaining blocks of IPv4 address space to the Regional Internet Registries (RIRs). Therefore, the world is responding by transitioning from IPv4 to IPv6. On 8 June 2011, top websites and Internet Service Providers (ISPs) around the world joined together with more than 1000 other participating websites in a "World IPv6 Day". Because of the success of this global-scale event, the Internet Society organized the "World IPv6 Launch Day" on 6 June 2012 (Internet Society, 2012). On this day major ISPs and companies around the world permanently enabled IPv6 for their products and services.

DOI: 10.4018/978-1-4666-4030-6.ch008

However, businesses need to migrate to IPv6 in a secure manner in order to avoid the possible security risks inherent in an IPv6 deployment. One of the security concerns comes from the new IPv6 features and mechanisms, which can expose the network to new security threats. For instance, StateLess Address Auto-Configuration (SLAAC) (Thomson, Narten, & Jinmei, 2007) and Neighbor Discovery (ND) (Narten, Nordmark, Simpson, & Soliman, 2007) messages are essential portions of the IPv6 suite. Both ND and SLAAC, together, are known as Neighbor Discovery Protocol (NDP). IPv6 nodes use NDP for several critical functions: to discover other nodes (routers/hosts) on the link, to find the mapping between the MAC and link local addresses, to detect duplicate addresses, and to maintain reachability information about the paths to active neighbors. Also, NDP plays a crucial role in mobile IPv6 (MIPv6) networks (Perkins, Johnson, & Arkko, 2011). However, NDP is vulnerable to spoofing and Denial-of-Service (DoS) attacks (Nikander, Kempf, & Nordmark, 2004) and attackers have already developed a set of tools to use in attacking ND functionalities (Hauser, 2012).

NDP specifications do not include any security provisions. It was designed to work in trustworthy links where all nodes on the link trust each other. However, we cannot assume that being on the same network is trustworthy as this assumption does not hold in number of different scenarios, such as, over wireless networks, where anyone can join a local link either with minimal or with no link layer authentication. Today people use public networks such as Wireless LAN at airports, hotels, and cafes, where a malicious user can impersonate legitimate nodes by forging NDP messages to generate serious attacks. RFC 3756 (Nikander, et al., 2004) shows a list of potential threats to NDP. Therefore, if NDP is not secured, it will be vulnerable to these various attacks. Some such attacks are Neighbor Solicitation (NS)/ Advertisement (NA) spoofing, Neighbor Unreachability Detection (NUD) faller, Duplicate Address Detection (DAD), Denial of Service (DoS), Malicious Last Hop Router, Spoofed Redirect Message, Bogus On-Link Prefix, Parameter Spoofing, and Replay attacks.

Therefore, RFC 3971 "SEcure Neighbor Discovery (SEND)" (Arkko, Kempf, Zill, & Nikander, 2005) was proposed as a set of enhancements to make the IPv6 neighbor and router discovery secure. SEND was designed to ensure message integrity, prevent IPv6 address theft, prevent replay attacks, and provide a mechanism for verifying the authority of routers. It uses Cryptographically Generated Addresses (CGA) (Aura, 2005), digital signature, and X.509 certification (Lynn, Kent, & Seo, 2004) to offer significant protection for NDP. A SEND-enabled node must generate or obtain a public-private key pair before it can claim an address. Then it generates the CGA address based on the public key and other auxiliary parameters. The associated private key is used to sign the outgoing ND messages from that address. For router authorization, every router must have a certificate from a trust anchor and the hosts provisioned with a trust anchor(s) list and picks routers that can show a valid certificate from a trust anchor. The SEND verifier node checks that the received address is a hash of the corresponding public key and that the signature, from the associated private key, is valid. If both verifications are successful, then the verifiers know that the address is not a stolen address and that it is from the address corresponding to public private key pairs.

Although SEND is considered to be a promising technique with which to protect NDP and to make IPv6 a very safe protocol, its deployment is not easy and thus is very challenging. SEND lacks mature implementations by developers of operating systems. It is compute-intensive and bandwidth-consuming. Moreover, SEND itself can be vulnerable to some types of attacks.

This chapter will introduce SEND functionalities and messages, discuss the practical considerations of SEND deployment as a cryptography solution in securing IPv6 local networks, survey

the SEND implementation status, analyze the CGA security and computation complexity, and review the proposals made to optimize SEND in order to make it work in real networks.

The chapter is organized as follows. An overview of NDP and possible attacks against it is presented in section 2. Section 3 explores the theory of SEND and shows how SEND can be used to protect NDP. In Section 4 we highlight the deployment challenges of SEND and survey the existing SEND implementations. Section 5 discusses some possible approaches to optimize SEND and to provide some recommendations for SEND deployment. The last section concludes this work.

BACKGROUND

In an IPv4 network, a host does not get an IP address by it itself. The IP address is assigned manually by a human or retrieved from a Dynamic Host Configuration Protocol (DHCP) daemon. However, DHCP (Droms, 1997) is not a native part of the TCP/IP suite. The DHCP exchange utilizes a broadcast/response method. Initially, a DHCP client does not have an IP address and does not know the IP address of any DHCP servers. The client sends a broadcast discovery message with a source IP address of 0.0.0.0 and with a destination address of FF.FF.FF.FF. The DHCP server responds to the Media Access Control (MAC) address from the sender with a message including parameters like the IP address and subnet of the proposed lease, the length of the lease, and the real IP address of the DHCP server. No matter which way IPv4 is configured (manual or from DHCP), a host must test if the address is already in use by sending an Address Resolution Protocol (ARP) probe (Cheshire, 2008).

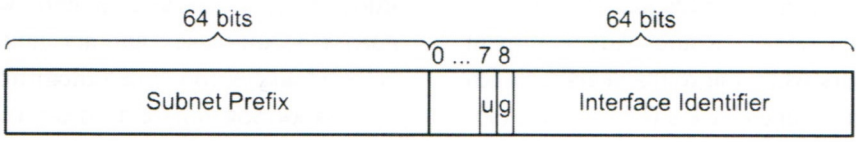
IPv6 replaces ARP and DHCP with a mechanism built into the IPv6 specifications thereby eliminating the need for daemons or extra configuration. This mode is called Stateless Address Auto-configuration. It is an integral part of IPv6. A new host attached to a new network automatically obtains all information needed for full connectivity with Zero-configuration. The following subsection briefly shows how the Stateless Auto-configuration works in IPv6. However DHCPv6, (Droms, et al., 2003) called Stateful Auto-configuration, is out of the scope of this chapter.

IPv6 Stateless Address Auto-Configuration (SLAAC)

Stateless auto-configuration is one of the most novel features of the IPv6 design. Each active network interface gets a default IPv6 address called the Local-Link address. A new host attached to a new network automatically obtains all the information needed for full connectivity. In IPv6 Stateless Address Auto-Configuration (SLAAC), the node creates the rightmost 64-bit (interface identifier), which identifies an individual node within a local network. The Interface Identifier (IID) is often configured from the Extended Unique Identifier (EUI-64) that is generated based on the interface hardware identifier, i.e., the Media Access Control (MAC). Then the node combines the subnet prefix with the IID to form a complete 128-bit IPv6 address as depicted in Figure 1. The IID includes two bits which are reserved by the IPv6 addressing architecture for special purposes. The 7th bit from the left in the 64-bit IID is called the Universal/Local bit (u bit) which, when set to "1", identifies the local IID as being globally unique. The 8th bit from the left is the Individual/Group (g bit) which is set to "1" to identify multicast addresses.

The subnet prefix part is used for routing packets across the network. The subnet prefix can be the reserved local link prefix (FE80::) used to generate local link addresses. The formation of local link addresses is done without any communications with the other hosts or by any human interaction beyond the enablement of the IPv6 network. These addresses are fully functional

Figure 1. IPv6 address format



within the local segment, i.e., hosts can use these addresses to communicate with other hosts on the same network segment. However, routers are prohibited from forwarding any packets with local-link addresses. The router uses the link-local address as an identifier. One valid link-local works as an identifier. Once a host configures a link-local address, the next step is to obtain information about its neighbors (hosts and routers) residing on the same link, i.e., the nodes of a multicast group. Hosts also use link-local address generation for their global IPv6 addresses.

When a host receives a Router Advertisement (RA), it proceeds to create a global IPv6 address. A router will send out a RA message periodically to the link or when a Router Solicitation (RS) is received from a host. A RA message contains the routing prefix, Hop limit, Maximum Transfer Unit (MTU), lifetime, etc. The global IPv6 address is created by concatenation of the prefix to the interface identifier. This address is a tentative address until it can be shown to be unique and that there are no other detectable matching addresses on the subnet (address collision). Duplicate Address Detection (DAD) is a procedure used to ensure that there are no address conflicts on the same link. The host achieves DAD by sending a Neighbor Solicitation (NS) message to the solicited-node multicast group of the target address. If there is a collision, a Neighbor Advertisement (NA) message will be sent from the node that has the same address. If no conflict is detected, the tentative address is approved for use as a valid address. It is common to have multiple valid addresses on each Network Interface Card (NIC). More than one global address can be bound to an interface and one link-local address is permitted per interface.

IPv6 Neighbor Discovery Protocol (NDP)

Neighbor Discovery (NDP) is one of the most important functions of the Internet Control Message Protocol for IPv6 (ICMPv6), RFC 4443 (Conta, Deering, & Gupta, 2006). Its messages are implemented as a set of ICMPv6 Types and Options and follow the ICMPv6 message formats. NDP was designed to be at this position in the IP protocol stack to keep it simple and to let NDP benefit from the services provided by IP, such as security and multicast. NDP messages consist of an ICMPv6 header, ND message specific data, and zero or more options. ND message options provide additional information such as link-layer addresses, on-link network prefix, on-link Maximum Transmission Unit (MTU) information, redirection data, mobility information, and router specifications. In this section, we give further details about NDP messages and operations and list some possible ways to attack it.

NDP Functionalities and Messages

NDP is used for several functions including router and prefix discovery, parameter discovery, address auto-configuration, address resolution, Duplicate Address Detection (DAD), Neighbor Unreachablility Detection (NUD), next-hop determination, and redirect. To achieve these functions and services between the neighbor nodes, NDP uses the following five ICMPv6 messages:

- Type 133, Code 0 - Router Solicitation (RS), RS is sent by IPv6 host to discover the default router and to glean network in-

formation such as prefixes and the Domain Name Server (DNS) address.

- Type 134, Code 0 - Router Advertisement (RA) occurs when the router receives a RS message and sends back a RA message (solicited router advertisement). Also, RA is sent by IPv6 routers periodically (unsolicited multicast router advertisements).
- Type 135, Code 0 - Neighbor Solicitation (NS) is used to resolve the neighbor nodes IPv6 address with its MAC address or to verify if the node is still reachable.
- Type 136, Code 0 - Neighbor Advertisement (NA) occurs when the node that receives a NS message sends back a NA message with its own MAC address.
- Type 137, Code 0 - Redirect is used by the router to inform other nodes of a better first-hop toward a destination.

NDP Inherent Threats

NDP is not secure and there is a potential for breaking the local network protection. NDP has some basic protection mechanisms based on the scope of NDP. It is a link-local protocol, so the source address must be either unspecified (::/128) or a link-local address, and the hop limit must be set to 255. Also, the routers do not forward link-local addresses. Thus NDP messages cannot be injected into the network infrastructure from beyond the directly connected layer 2 access networks. This protection shield is not enough to completely protect IPv6 local networks. Therefore, without securing NDP, it is vulnerable to various attacks which can be categorized as spoofing, Denial-of-Service (DoS), Replay, Redirect, and Rogue routing information attacks. RFC 3756 (Nikander, et al., 2004) describes and categorizes some of the possible attacks against NDP. In addition, the Hacker's Choice IPv6 (THC-IPv6) (Hauser, 2012) provides an implemented tool set that can be used to attack IPv6. The following list presents brief reviews of these possible attacks:

- **Spoofing Attack:** Spoofing is a class of attacks in which a malicious node successfully uses another node's address or identifier to gain an illegitimate advantage. Spoofing can be used for a wide variety of purposes. It can be used to leverage Man-In-The-Middle (MITM) attacks, create a DoS attack, hide the attacker's identity, abuse the trust relation between the legitimated nodes, etc. ARP spoofing is a well known attack in the IPv4 network. Instead of ARP, IPv6 relies on the NDP to carry out the address resolution operation. Without the IPv6 authentication mechanism, the attacker can generate crafted IPv6 packets with spoofed source addresses and send them across the network.
- **Denial-of-Service (DoS) Attack:** DoS is a class of attack that eats up system resources when a malicious node prevents the communication between the legitimate node and other nodes. An attacker can generate DoS on Duplicate Address Detection (DAD) in order to prevent a network node from obtaining a network address. DAD is used to ensure that there is no address collision on the same link. A malicious node may hinder the legitimate host from getting a new IPv6 address by always responding to every duplicate address detection attempt with a spoofed message saying "I have this address". The victim would thus find out that every IPv6 address it tries to use is being used by other nodes on the link and so the victim would not be able to configure an IP address for accessing the network.
- **Replay Attack:** In this form of attack the attacker captures the messages exchanged between two nodes and later changes them. NDP messages are prone to replay attacks. One example occurs when HostA wants to communicate with HostB. It sends a NS message to get the MAC address of HostB.

An attacker on the same link can capture the NS message and change it to take command of the traffic flow between HostA and HostB. Another example is the replay of a RA message. The attacker can receive the RA and change some RA parameters, and later, resend this message, which now contains false router information, back to the link.

- **Redirect Attack:** Redirect attack is a class of attacks in which a malicious node redirects packets away from the legitimate receiver to another node. In IPv6, Redirect messages are used by routers to inform nodes of a better first hop router. An attacker can fabricate such a message and then take over routing from the legitimate router and act as a MITM by intercepting all messages between the two nodes.
- **Rogue Router Attack:** In this type of attack, a malicious node injects rogue information to poison the routing tables, reroute the traffic, or prevent the victim from accessing the desired network. It is very easy for an attacker to configure a rogue router on an unsecured link. On the other hand, it is difficult for a node to distinguish between fake and authorized router advertisements, especially for a newly connected node, because it cannot validate the routers without having an IP address to communicate. Thus a malicious node can advertise itself as a router and send a bogus address prefix, or advertise itself as the last hop router to act as a MITM and effectively receive, drop, or replay the packets. Such kinds of attacks can cause serious problems since they affect all other nodes that are connected to the same segment.

In practice, an attacker can use several of the above types of attacks simultaneously. For example, spoofing and DoS can be used together. The attacker can fabricate packets with a fake source IP address to hide his identity which makes it more difficult to detect the attack.

NDP Privacy Implications

SLAAC may lead to serious privacy implications. Generating the Interface Identifier (IID) part of IPv6 based on the MAC address results in a static IID, which remains constant over time. This one-to-one mapping between the MAC address and IID remains the same regardless of the network location. Therefore, every network card will be set to the same local-link address. Consequently, the attacker can correlate the captured traffic from a specific IID to a certain device. This makes it possible for the attacker to track a node from anywhere. Once the location and the identity of the user are determined, the attacker can target the user for identity theft or other related crimes.

RFC 4941 (Narten, Draves, & Krishnan, 2007) addresses this problem and introduces a solution whereby a global scope address is generated from interface identifiers that change over time. Changing the IID part over time makes it more difficult for eavesdroppers and other information collectors to collect the information they desire. The RFC 4941 approach, however, while protecting the privacy, does not provide protection against IP address spoofing attacks.

IPsec and Auto-Configuration

To secure the IPv6 network, security must be established in the initial state, when the node gets its address. Secured hosts must only be allowed to communicate with other secured hosts. The challenge to the NDP occurs when a host first joins a new network because it does not know who the other nodes are or how to discern legitimate hosts from malicious hosts. Although the original specification of NDP called for the use of IPsec (Kent & Seo, 2005) to protect NDP messages, it does not specify how to use it. IPsec is not suitable for securing the NDP auto-configuration process

due to a bootstrapping problem. When using the Internet Key Exchange (IKE), the nodes need to be addressable, first, before the IPsec can use them. The automatic key exchange can occur after the hosts have already established the IPv6 addresses. Therefore, the IETF developed the SEcure Neighbor Discovery Protocol (SEND) to counteract the vulnerabilities in NDP. The focus of CGA and SEND is on securing the initial stage, when a host does not have a valid address and is trying to establish its own address on a new network.

SECURE NEIGHBOR DISCOVERY (SEND)

SEND is an extension of the NDP which offers three additional features; address ownership proof, message protection, and a router authorization mechanism. To achieve these enhancements, SEND encodes its messages in ICMPv6 by creating new Option Types. Four new options can be appended to the regular NDP message in order to create a SEND packet; CGA, RSA Signature, Nonce, and Timestamp. In addition, SEND came up with two new ICMPv6 messages for use in the router authorization process; Certificate Path Solicitation (CPS) and Certificate Path Advertisement (CPA).

CGA Address Ownership Proof (CGA Option and RSA Signature Option)

Cryptographically Generated Addresses (CGAs) form the basic foundation of SEND, which was proposed to prevent address theft. The main idea behind CGA is to use asymmetric cryptography to authenticate IPv6 Auto-configuration addressing without changing the zero configuration paradigm of SLAAC. CGA can work alone as a "zero configurations" security approach without the need to rely on any security infrastructure or any third party. CGAs are IPv6 addresses where the Interface Identifiers (IIDs) are generated by one-way hashing of the node's public key and other auxiliary parameters. Thus the IPv6 address for a node is bound to its public key. This binding can be verified by re-computing the hash value which is then compared to the interface identifier of the sender's address.

The idea of using CGA first appeared in the Childproof Authentication for MIPv6 (CAM), which was proposed by O'Shea and Roe (2001). In the CAM approach, the hash of the owner's public key is written into the interface identifier part of IPv6 address. Later Nikander (2002) suggested an improvement, and an extension, to the CAM approach to make it more resistant to birthday collision by adding some "random" data to the hash input. The final form of CGA was proposed by Aura (2003) and standardized in RFC 3972 (Aura, 2005). In this chapter the term CGA is used to refer to the standardized CGA which appears in RFC 3972.

The introduction of the Hash Extension is the main difference between Aura's proposal (Aura, 2003) and earlier proposals. The Hash Extension increases the security of the hash functions where the hash value can only have a limited number of bits. The use of a 64-bit value is not adequate for the protection of the address from a security standpoint. The Hash Extension technique increases the hash length beyond the 64-bit limit without actually increasing its length. Instead of a single hash value, the standard CGA (Aura, 2005) computes two independent one-way hash values (Hash1 and Hash2). The Hash2 (112-bit) calculations determine an input parameter for the Hash1 (64-bit) calculation. The purpose of the second hash (Hash2), Hash Extension, is to increase the cost (computing time) to the hacker for doing a brute-force attack, without increasing the length of the hash output value, which is written to the interface identifier portion of the IPv6 address.

This technique increases both the cost of generating a new CGA address and the cost of initiating a brute-force attack against the address. A scaling factor, called the Security Parameter (Sec), is used to determine the level of security for each

generated address. The Sec is an unsigned 3-bit integer having a value between 0 and 7 (0 being the least secure while 7 the most).

CGA Generation

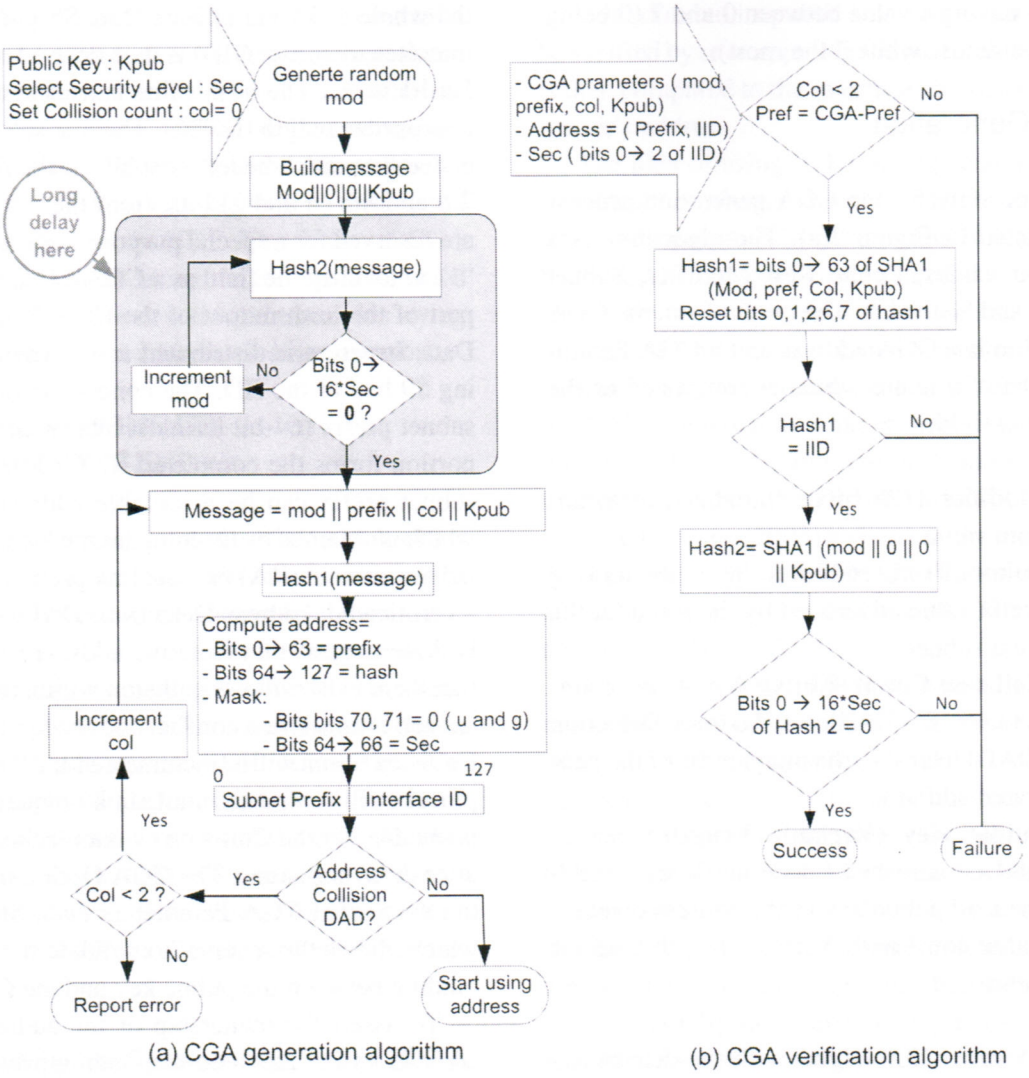
The procedure for the CGA generation process is depicted in Figure 2(a). The algorithm uses as input values; Public Key, Modifier, Subnet Prefix, and Sec value. The output from the CGA algorithm is a CGA address and a CGA Parameters Data Structure which is comprised of the following fields:

- **Modifier (128 bits):** Initialized to a random value.
- **Subnet Prefix (64 bits):** Set to the routing prefix value advertised by the router at the local subnet.
- **Collision Count (8 bits):** A collision counter used for Duplicate Address Detection (DAD) to ensure the uniqueness of the generated address.
- **Public Key (Variable Length):** Set to the Distinguished Encoding Rules (DER)-encoded public key of the address owner.
- **Extension Field:** Variable length field for future use.

CGA generation begins with the determination of the address owner's Public Key and by selecting the proper Sec value to use. The process continues with the Hash2 computation loop which finds the Final Modifier that will satisfy the condition where the 16*Sec-leftmost bits of the Hash2 are equal to zero. The Hash2 value is a hash created by the combination of the Modifier, concatenation of (64 + 8) zero bits and the Public Key. The address generator tries different values for the Modifier until the 16*Sec-leftmost bits of Hash2 computes to zero. Once a match is found, the Hash2 loop computation terminates. At this point the Final Modifier value is saved and used as an input to the Hash1 computation. The Hash1 value is a hash created from the combination of the whole CGA Parameters Data Structure. The interface identifier (IID) is then derived from the Hash1 value. The hash value is truncated to the appropriate length (64-bit). The Sec value is encoded into the three leftmost bits of the IID. The 7th and 8th (u and g) bits, from the left of IID, are reserved for a special purpose; they are set to "1" to identify the field as a CGA address. Now, part of the hash output of the CGA Parameters Data Structure is distributed across the remaining 59 bits of the IID. The concatenation of the subnet prefix (64-bit leftmost bits) with the IID portion forms the completed IPv6 address. The subnet prefix can be a routable address prefix which is obtained by listening for the local Router Advertisement (RA) or local link prefix. Finally, a Duplicated Address Detection (DAD) process is done against this tentative address to ensure that there is no address collision within the same subnet. If an address conflict does occur, then the Collision Count will be incremented and the Hash1 process will be repeated until a link-unique address is obtained or the Collision Count reaches 2, i.e., after three collisions. The CGA Option includes the associated CGA Parameters Data Structure which allows the receiver to validate the proper binding between the public key and the CGA.

To assert the ownership of the address, the address owner uses a corresponding private key to sign messages sent from that address. Signing a message using CGA requires the combined use of the CGA address, the associated CGA Parameters Data Structure, the message, and the private key that corresponds to the public key that has been used in the CGA's generation algorithm. Finally, the node will send the message, the CGA Parameters Data Structure, and the signature to the receiver node. SEND uses the RSA Signature Option to authenticate the identity of the sender and to prevent an attacker from spoofing CGA addresses. The RSA digital signature contains the following fields:

Figure 2. CGA algorithm: (a) generation; (b) verification



- 128 bit CGA Message Tag value for SEND.
- 128 bit Source Address from the IPv6 header
- 128 bit Destination Address from the IPv6 header
- 8 bit Type, 8 bit Code and 16 bit Checksum fields from the ICMPv6 header
- ND protocol message header, starting after the ICMPv6 checksum, and up to, but not including, the ND protocol options
- ND protocol options preceding the RSA signature option

CGA Verification

CGA verification takes as an input the IPv6 address and CGA Parameters Data Structure. If the verification succeeds, the verifier knows that the public key belongs to that address. The verifier can then use the public key to authenticate the signed messages received from the address owner. The verification process is shown in Figure 2(b). According to RFC3972, the verification of address ownership is achieved by executing the following steps:

- Check that the Collision Count value is 0, 1 or 2, and that the "Subnet Prefix" value is equal to the subnet prefix of the address. The CGA verification fails if either check fails.
- Concatenate the CGA Parameters Data Structure and execute the hash algorithm (SHA-1) on the concatenation. The 64 leftmost bits of the result are Hash1.
- Compare Hash1 with the interface identifier of the address. Differences in the "u" and "g" bits and in the three leftmost bits are ignored. If the 64-bit values differ (other than in the five ignored bits), the CGA verification fails.
- Read the security parameter (*Sec*) from the three leftmost bits of the interface identifier of the address.
- Concatenate the Modifier, 64+8 zero bits and the Public Key. Execute the hash algorithm on the concatenation. The leftmost 112 bits of the result are Hash2.
- Compare the 16*Sec leftmost bits of Hash2 with zero. If any one of these is non-zero, CGA verification fails. Otherwise, the verification succeeds. If Sec=0, verification never fails at this step.

In addition to the CGA verification process outlined above, the verifier uses the public key to verify the signature of the message. If the signature is valid, the verifier knows that the message was sent by the owner of a specific IPv6 address. It is important to note that once the address has been created, the cost of using and verifying a CGA address does not depend on the Sec value.

Replay Protection: Nonce Option and Timestamp Option

SEND includes a Nonce Option (a random number) in the solicitation message and requires the advertisement to include a matching option to ensure that an advertisement is a fresh response to a solicitation sent earlier by the node. This option is used to prevent a replay attack in solicited messages, such as NS/NA and RS/RA, when there is two-way communication. The Nonce Option cannot be used for one-way communication messages. SEND uses the Timestamp Option to provide replay protection against unsolicited advertisements such as periodic RA and Redirect messages. Here the assumption is that all nodes have synchronized clocks and that the node can prevent a replay attack by executing a Timestamp checking algorithm.

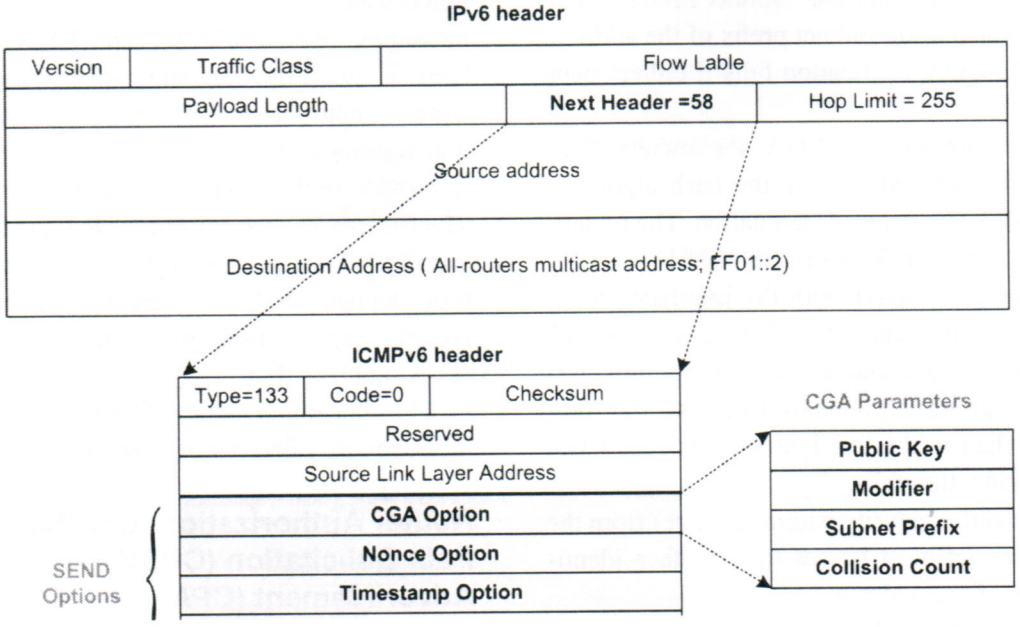
The conceptual layout of a router solicitation message with SEND Options is shown in Figure 3.

Router Authorization: Certificate Path Solicitation (CPS)/ Advertisement (CPA)

CGA Option and RSA Option together are used for IPv6 address authentication. This authentication proves only that the owner of the claimed IPv6 address is the owner of the corresponding public-private key pair. However, the CGA authentication does not provide anything about the privilege of using this address. An attacker can generate his own private-public key and start the communication in which he easily advertises himself as a router.

SEND uses third parties for router authorization. This process is known as the Authentication Delegation Discovery (ADD). SEND uses the ADD process to validate and authorize IPv6 routers which act as default gateways, and to specify IPv6 prefixes that a router is authorized to announce on the link. ADD relies on an electronic certificate issued by a trusted party. Before any node can accept a router as its default router, the node must be configured with a trust anchor(s) that can certify the router via certificate paths. The node requests the "router" to provide its X.509 certificate path to a Trust Anchor (TA), which is preconfigured on the node. The "router" should not be trusted if it fails to provide the path to the TA.

Figure 3. Router solicitation message with SEND options



This feature is achieved by the use of two new ICMPv6 message types; the Certificate Path Solicitation (CPS) and the Certificate Path Advertisement (CPA). A CPS message, ICMPv6 type 148, is sent by hosts during the ADD process to request a certification path between a router and one of the host's trust anchors. The CPA message, ICMPv6 type 149, is sent in reply to the CPS message and contains the router certificate. In addition, new rules describe the preferred behavior when a SEND node receives a message, whether supported by SEND or not. Figure 4 shows a simplified view of the router authorization mechanism.

THE MAIN CHALLENGES AND LIMITATIONS OF SEND

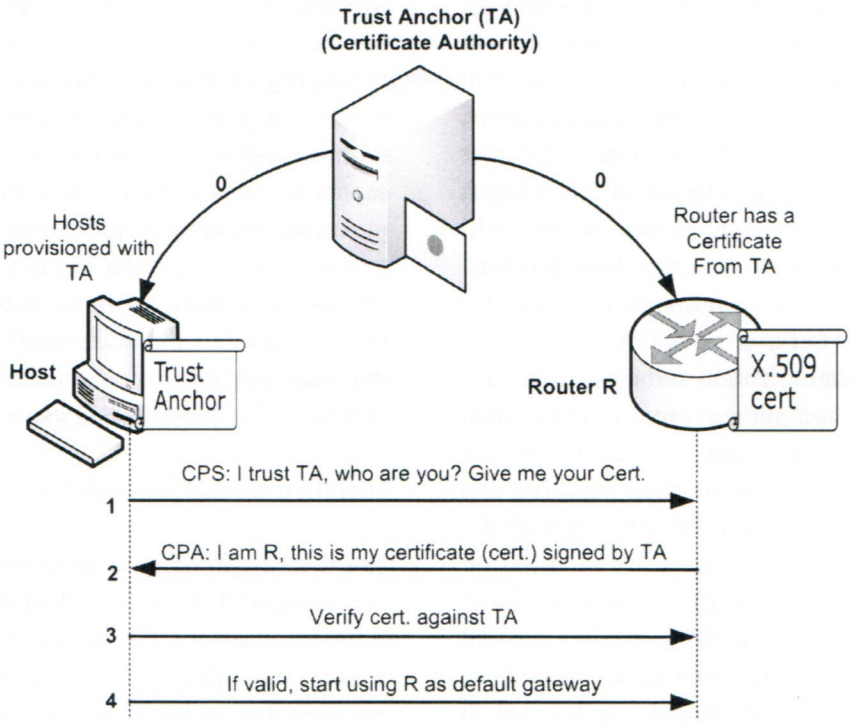
Although SEND seems to be a promising technique for protecting NDP messages, it is lacking in several respects with regard to computation, implementation, deployment, and security. These limitations may prevent the use of SEND and leave NDP messages vulnerable to potential attacks. In this section we give further details about these limitations and challenges.

SEND Security and Privacy Considerations

SEND Security Limitations and Remaining Attacks

SEND can prevent the theft of another node's address but it cannot provide assurance about the identity of the real node and it is not capable of guaranteeing that the CGA address is being used by the appropriate node. Since CGAs are not certified, an attacker can come up with a new and valid address from its own public key, and start the communication. However, the address owner uses the corresponding private key to assert its ownership and to sign NDP messages sent from the address. Therefore, an attacker can impersonate another node address from a valid public key, but cannot take over an address of an

Figure 4. Simplified view of router authorization delegation discovery in SEND



existing host because the attacker does not have the private key and he would have to find a collision of the cryptographic hash values, which would cost $2^{(59+16*Sec)}$ compute operations, on average. To insure greater security, a certificate authority is needed in order to validate the keys.

Moreover SEND can be susceptible to Denial-of-Service (DoS) attacks. An attacker can conduct DoS attacks on some particular steps within the CGA verification process (AlSa'deh, Rafiee, & Meinel, 2012b). He can perform a DoS attack against the DAD check and the CGA parameter verification in order to prevent a new CGA node from joining a link. When an attacker receives a DAD message sent by a CGA node, he can immediately copy the IP address, CGA parameters, and Signature. Then he replies with an NA message that has the same valid Signature and CGA parameters. The victim's node receives the reply via a secure NA message and checks its validity as defined by the CGA verification algorithm. If it is valid, because the signature is valid and the reply was received in the allotted time frame, the victim will increment its Collision Count value, and try another address. If the attacker replies again and the Collision Count reaches 2, the victim stops and reports an error. In this way the attacker does not need to do any brute-force searches against the CGA in order to carry out this attack. Even though this attack is limited to DoS attacks because the attacker does not have the private key needed to sign messages, the attacker can prevent the CGA address configuration for all new nodes that want to attach to the local link. Therefore, the CGA algorithm may need to be extended to verify the response to the DAD messages before incrementing the Collision Count. The CGA node should discard the DAD response message in two cases; if it comes from a non-CGA and gives the priority to the new CGA address, or if it has the same tentative address, CGA parameters, and signature that was sent before.

An attacker can also capture ND messages and change a sender's CGA parameters. Consequently, the CGA verification process, at receiver side, will fail. So, in this scenario, the attacker can prevent the communication between sender and receiver. However, the same scenario can occur with other security protocols. If an attacker inserts a bogus IPsec datagram, the IPsec at the receiver will drop the datagram since integrity check is bogus. Therefore, we do not see this type of attack as a major drawback to the use of CGA.

The router authorization and certificate validation can be a complicated process and a target for DoS attacks. The routers are provisioned with a certificate that proves their authority and hosts need to be prepared, beforehand, with the anchor certificate. But this trust relationship can be a long chain of trust, so the Authorized Delegation Discovery (ADD) requires an end node for storing and retrieving all the certificates in the certification path in order to be able to verify the authorized router. Thus the router is required to perform a large number of operations for generating, verifying, and signing NDP messages, especially when a high frequency of router advertisements is required, which can affect the routers' performance. Also, attackers may target hosts by sending a large number of unnecessary certification paths thus forcing hosts to spend useless memory and verification resources on them. The numerous computations required by a host to generate and verify CGAs and the certificates chain make it vulnerable to DoS attacks.

The CGAs can also be vulnerable to another type of attack known as the global Time-Memory Trade-Off (TMTO) attack. The attacker needs to do an exhaustive search for hash collision or must create a large, pre-computed database from the interface identifiers of the attacker's own public key(s) used to find matches for many addresses. A more secure version, called CGA++ (Bos, Özen, & Hubaux, 2009) has been proposed to resist this type of attack. In CGA++, the subnet prefix is included in the calculation of Hash2 and all the Modifier, Collision Count and Subnet Prefix values are signed by the private-key corresponding to the public key used. In this way, TMTO cannot be applied globally. The attacker would have to do a brute-force search for each address prefix separately. However, it is not easy to impersonate a random node in a network because of the storage required in order to carry out this attack. CGA++ also comes with an additional cost due to the amount of signature verifications needed, thus requiring much more time for the generation of a CGA for the same Sec value.

SEND Privacy Concerns

Due to the high computation complexity involved in creating a CGA it is likely that once a node generates an acceptable CGA it will continue to use it at that subnet. The result is that nodes using CGAs are still susceptible to privacy related attacks. However, we think the CGA privacy implication (when the node uses the same subset for a long time) can be solved by setting a valid time limit for the use of a CGA address. If the time has elapsed, a new CGA, with a new CGA parameter, should be generated. But to avoid the long CGA generation time, it is not recommended to choose a Sec value greater than "1" for the current applications. There should be a balance between the valid time and the security level. For a mobile node, CGA can provide the necessary privacy protection. When a node moves to new subnet, Hash1 should be recalculated by including the new subnet prefix. Consequently, a new IPv6 address with new IID will be generated without the need to recalculate Hash2. The process of generating a new address only costs one Hash calculation. In this case, the only concern is the possibility of tracking the node based on its public key if it remains fixed. But it is not easy to track the node based on its public key. Normally tracking over the Internet is done based on the IP address.

SEND Computation Cost and Bandwidth Consumption

In reality, the average CGA address generation time depends on the Sec bit setting. Even there is a probabilistic guarantee that the CGA address generation will stop after a certain number of iterations, but it is impossible to tell exactly how long it will take for the CGA generation when Sec is not zero. It might vary significantly because the Modifier field of the CGA parameter is randomly generated. Theoretically, the computation time needed to generate a Hash2 value is increased by 2^{16} for each Sec value.

Performing 1000 tests on an Intel Duo2 2.67 GHz CPU we found the average CGA generation time to be 402 Milliseconds for Sec = 1. A test on an unrepresentative set of 5 samples carried out with 2.67 GHz CPU gave an average CGA generation time of 5923857 Milliseconds (1 hour and 39 minutes). Accordingly, Sec values higher than "1" are not recommended, as they require a lot of processing power and too much time when using the current technologies. Bos, & et al., (2009) expected that CGA generation time for Sec=3 would take around 24 years on a modern workstation (AMD64).

Furthermore, SEND requires that each node include the public key and other parameters with the message and then to affix its Signature to every signaling packet that it generates. This means that more than 1K bytes is added to each packet. This increases the communication overhead and consumes network bandwidth and computational resources. Moreover, the transportation of all the certificates and keys between routers and hosts increases the local network traffic. Also, the certification path information exchange requires a lot of Certificate Path Solicitation/Advertisement CPS/CPA messages, especially if there is an error on the network and retransmission is required.

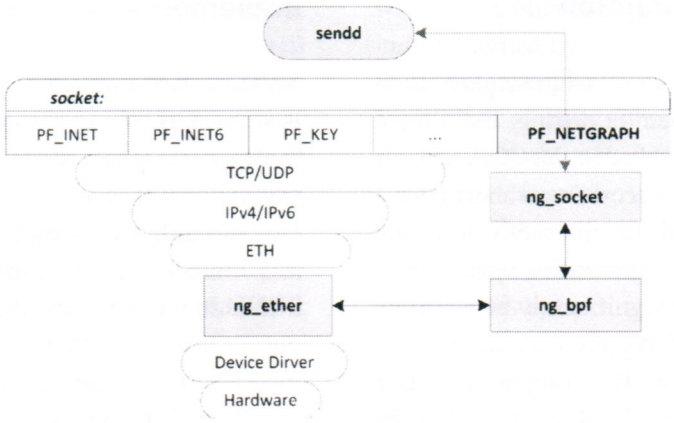
Lack of Sophisticated Implementation at End-Nodes

SEND-aware implantation is still scarce. NDP is supported by most modern operating systems, but it lacks support for SEND, and it is a security standard without sophisticated implementations. Even though some major vendors, such as Cisco and Juniper, have various levels of support for SEND in their routers, there is no major operating system providing a good level of support for SEND. Current SEND implementations for specific OS distribution are, basically, a proof of concept rather than production-ready software.

DoCoMo's SEND

NTT DoCoMo USA Labs (2008) implemented the first open source of SEND. Their implementation works exclusively with Linusx 2.6 or FreeBSD 5.4 and later. It uses a Berkley Packet Filter (BPF) interface embedded in a netgraph node to get traffic from the kernel to the user-space daemon, and vice versa. The user-space daemon handles the SEND options. The communication between the NDP stack in the kernel and the SEND daemon flows through the chain of netgraph nodes, rather than passing through the normal layer processing. The DoCoMo implementation has some limitations. First, all network traffic has to traverse through packet filtering hooks which introduces significant processing overhead. Second, it depends on the subsystem, which is only available for the FreeBSD operating system family and thus makes it non-portable for other operating systems. Additionally, users need to have a good knowledge of firewall rules (ip6tables) since this knowledge is essential for the configuration of the application. Finally, DoCoMo USA Labs no longer supports the SEND project and the source code is no longer available for downloading at their web site since their support has been stopped. Figure 5 shows the DoCoMo SEND general architecture.

Figure 5. DoCoMo SEND architecture



Native SeND kernel API for *BSD

This implementation (Kukec & Zeeb, 2010) is an attempt to overcome the major drawbacks of DoCoMo's implantation by implementing a new kernel-user space API for SEND and eliminating the use of the netgraph and Berkley Packet Filter (BPF). Avoiding the use of netgraph reduces the overhead and speeds up packet processing and makes the implementation more portable to other operating systems. Native SeND uses routing control sockets for exchanging messages between the kernel and the user-space. It handles the packets that may be affected by SEND, rather than processing all packets, thus allowing the existing kernel to process the other packets. This implementation uses a kernel module (send.ko) which acts as a gateway between the network stack and the user-space interface.

Huawei and BUPT (ipv6-send-cga)

Beijing University of Post and Tele-communications (2009) introduced an implementation for SEND within the Linux kernel IPv6 module. The "C" code is partially built into the real NDP code at the kernel, where direct access to neighbor caches and routing tables are available. However, the cryptographic processing remains in the user-space. This work is a research prototype under development which still lacks interoperability testing. Bugs that can cause the kernel to crash can be expected. Figure 6 shows the architecture of Huawei and BUPT SEND implementation.

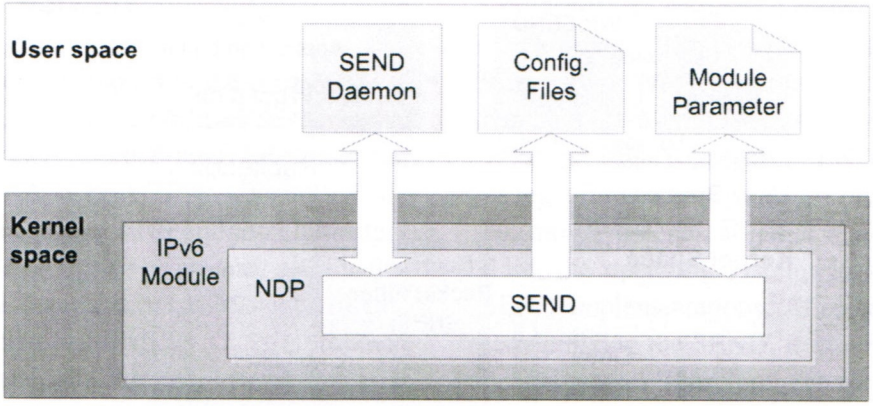
Easy-SEND

Easy-SEND (Chiu & Gamess, 2009) is another Linux user-space implementation of SEND developed in Java. Easy-SEND is an open source project which has been developed for educational purposes. Easy-SEND does not implement the router authorization portion of SEND.

NDprotector

NDprotector (Cheneau, 2011) is another implementation of CGA and SEND for Linux based on Scapy6. NDprotector follows the same principle of the DoCoMo implementation with its dependency on iproute2, ip6table, and the netfilter queue. The implementation can be divided into initialization and runtime phases. During the initialization, CGA addresses are set up and ip6tables rules are defined. These rules route all NDP messages to specific netfilter queues. NDprotectror's runtime

Figure 6. Huawei and BUPT SEND (ipv6-send-cga) architecture



component takes the NDP messages out of these queues and secures the outgoing NDP messages or verifies the incoming messages. NDprotector supports Elliptic Curve Cryptographically (ECC) keys in addition to the standard RSA keys. It is implemented in Python and therefore uses a python wrapper to access the netfilter. For packet manipulation, NDprotector uses a modified version of Scapy6 (scapy6send).

WinSEND

Windows SEcure Neighbor Discovery (WinSEND) (Rafiee, AlSa'deh, & Meinel, 2011) is the first SEND implementation for Windows families. It is a user-space implementation which is developed in Microsoft.NET. WinSEND works as a service for Windows families with a user interface which allows the setting of security parameters for the proper Network Interface Card (NIC). This implementation came in response to a lack of SEND support for Windows operating system. Since the Windows family is the most popular operating system in use today, and accounts for almost 80% of the operating systems in use (W3Schools, 2012), it was felt that support for SEND was necessary. Figure 7 shows the architecture of the WinSEND implementation.

FUTURE RESEARCH DIRECTIONS TO FACILITATE SEND DEPLOYMENTS AND PROTECT LOCAL IPV6 NETWORK

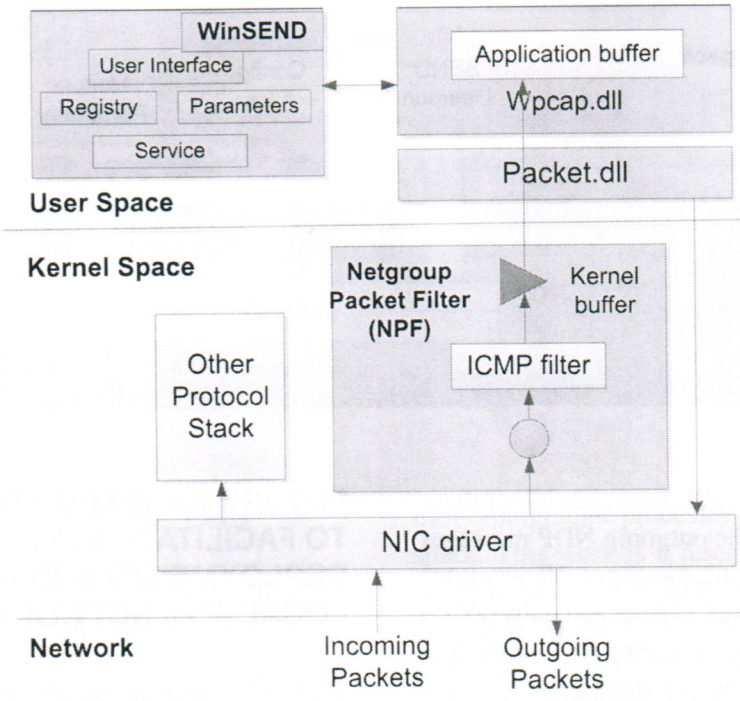
There are some proposals to optimize and facilitate SEND deployment and protect IPv6 local networks. Some of these proposals come solely within the SEND domain without needing to rely on other deployments. Others are proposed as alternative approaches to avoid the difficulty of SEND deployments.

Proposals to Facilitate SEND Deployments

Cheneau, Boudguiga, and Laurent (2010) propose an improved method for generating CGA by using an Elliptic Curve Cryptograph (ECC) key instead of a standardized RSA key. ECC has a shorter key for the same security level as RSA, which leads to creation of smaller packets. Thus, the use of ECC is more suitable for resource-limited environments.

Another idea that has potential is to delegate the expense of the computation to a powerful machine where a key sever does the computation on behalf of the nodes in advance or offline. Jiang and Xia (2012) propose to use the DHCPv6

Figure 7. WinSEND architecture



server to manage the CGA. DHCPv6 is extended to propagate the parameters that a host needs to generate a CGA. A host may send a request to a DHCPv6 server to have the CGA computed for it.

Another solution, to achieve faster CGA computations, would be to use cryptographic accelerator cards to compute the CGA or to use parallelized algorithms in the computation of the CGA modifier. Rafiee, AlSa'deh, and Meinel (2012) implemented a parallelizing CGA generation algorithm in order to speed up the CGA computation. With the parallel approach, the computation time has been reduced dramatically by increasing the number of cores in the computing device, which allows for parallel processing.

In order to avoid an unreasonable and unacceptable delay in CGA generation, the use of a time termination condition may be a practical approach to make using CGA feasible. For example, based on the CPU speed, the algorithm recommends a proper value for Sec, or the setting of a time termination condition. The termination condition will definitely depend on the application requirement, e.g., MIPv6 needs to finish the address generation within hundreds of a millisecond. AlSa'deh, Rafiee, and Meinel (2012a) proposed a modified CGA generation algorithm, called Time-Based CGA (TB-CGA), to put a limit on the maximum time that the user/application is willing to invest for CGA generation. TB-CGA takes the upper bound of the CGA running time as an input, and the Sec value is determined as an output of the brute-force computations. Also, this paper proposes to reduce the granularity of the security level from "16" to "8", in order to increase the chances of having a better Sec value within the time limit.

An additional possible way to speed up the CGA generation is to use the Optimistic DAD (Moore, 2006) with CGA. The Optimistic DAD approach is a method to minimize address configuration delays by making an address available for use before completing the DAD. The DAD in the CGA generation algorithm can be one reason for the delay.

Other Approaches to Protect IPv6 Local Network

Other approaches try to avoid the complexity of the SEND deployment and to propose alternative protection models. IPv6 Router Advertisement Guard (IPv6 RA-Guard) (Levy-Abegnoli, et al., 2011) proposes an alternative and complementary approach to SEND based on filtering at layer-2 in order to avoid router authorization and certificate validation complexity. IPv6 RA-Guard is proposed to counter the rogue-RA problem by applying a layer-2 switching device in order to identify invalid RAs and then blocking them. However, IPv6 RA-Guard still cannot prevent IP address theft if it is deployed alone. Maybe a combination of CGA and RA-Guard would be a solution to a secure IPv6 network.

Other approaches, such as the NDPMon (Beck, Cholez, Festor, & Chrisment, 2007) tool, have been proposed to protect the local network by monitoring the NDP messages in order to detect malicious activity. However, monitoring techniques cannot prevent attacks: attacks may be capable of damaging the network before a response is received from the administrator. Also, it is not easy to determine which node did the attack because the attacker may be using a spoofed address.

CONCLUSION

Neighbor Discovery Protocol (NDP) is one of the main protocols in the IPv6 suite, but it is not secure, and it is vulnerable to malicious attacks. These security issues are what led to the birth of SEcure Neighbor Discovery (SEND). The SEND approach eliminates most attack vectors levied against IPv6 link operations. SEND uses Cryptographically Generated Addresses (CGAs) in order to prove the ownership of a claimed address. It is also based on the use of X.509 certificates and a router authorization delegation mechanism to prevent fake router advertisement attacks. In addition, SEND uses the Nonce Option and Timestamp Option in order to prevent replay attacks.

However, a SEND deployment is a challenge. There are several reasons for this: first, SEND is compute-intensive, especially when high security values are used, second, the SEND Authorization Delegation Discovery (ADD) mechanism is, so far, mostly theoretical rather than practical, and third, the operating systems lack sophisticated SEND implementations. After several years of standardizing the SEND, it is, sorry to say, still in trial stages. Therefore, enhancing CGAs and SEND, in order to make them lightweight and practical, while maintaining some level of security and privacy, is very important for the deployment of IPv6 in a secure manner. If this is not accomplished, then the IPv6 network will be left vulnerable to IP spoofing related attacks.

REFERENCES

W3Schools. (2012). OS platform statistics. Retrieved from http://www.w3schools.com/browsers/browsers_os.asp

AlSa'deh, A., Rafiee, H., & Meinel, C. (2012a). Stopping time condition for practical IPv6 cryptographically generated addresses. In Proceedings of 2012 International Conference on Information Networking, ICOIN 2012, (pp. 257-262). Bali, Indonesia: IEEE.

AlSa'deh, A., Rafiee, H., & Meinel, C. (2012b). Cryptographically generated addresses (CGAs): Possible attacks and proposed mitigation approaches. In Proceedings of 12th IEEE International Conference on Computer and Information Technology (IEEE CIT'12). Chengdu, China: IEEE.

Arkko, J., Kempf, J., Zill, B., & Nikander, P. (2005). Secure neighbor discovery (SEND). RFC 3971. Retrieved from http://tools.ietf.org/html/rfc3971

Aura, T. (2003). Cryptographically generated addresses (CGA). *Lecture Notes in Computer Science, 2851*, 29–43. doi:10.1007/10958513_3.

Aura, T. (2005). Cryptographically generated addresses (CGA). RFC 3972. Retrieved from http://tools.ietf.org/html/rfc3972

Beck, F., Cholez, T., Festor, O., & Chrisment, I. (2007). Monitoring the neighbor discovery protocol. In Proceedings of the International Multi-Conference on Computing in the Global Information Technology, (pp. 57-62). IEEE.

Bos, J. W., Özen, O., & Hubaux, J. P. (2009). Analysis and optimization of cryptographically generated addresses. In Proceedings of the 12th International Conference on Information Security, (pp. 17-32). Pisa, Italy: Springer-Verlag.

Cheneau, T. (2011). NDprotector an implementation of CGA & SEND that works on Linux. Retrieved from http://amnesiak.org/NDprotector/

Cheneau, T., Boudguiga, A., & Laurent, M. (2010). Significantly improved performances of the cryptographically generated addresses thanks to ECC and GPGPU. *Computers & Security, 29*(4), 419–431. doi:10.1016/j.cose.2009.12.008.

Cheshire, S. (2008). IPv4 address conflict detection. RFC 5227. Retrieved from http://tools.ietf.org/html/rfc5227

Chiu, S., & Gamess, E. (2009). Easy-SEND: A didactic implementation of the secure neighbor discovery protocol for IPv6. In Proceedings of the World Congress on Engineering and Computer Science, WCECS 2009, (pp. 260-265). San Francisco, CA: WCECS.

Conta, A., Deering, S., & Gupta, M. (2006). Internet control message protocol (ICMPv6) for the internet protocol version 6 (IPv6) specification. RFC 4443. Retrieved from http://tools.ietf.org/html/rfc4443

DOCOMO Communications Laboratories USA. (2008). SEND project. Retrieved April 23, 2012, from http://www.docomolabs-usa.com/lab_opensource.html

Droms, R. (1997). Dynamic host configuration protocol. RFC 2131. Retrieved from http://tools.ietf.org/html/rfc2131

Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., & Carney, M. (2003). Dynamic host configuration protocol for IPv6 (DHCPv6). Retrieved from http://tools.ietf.org/html/rfc3315

Groat, S., Dunlop, M., Marchany, R., & Tront, J. (2010). The privacy implications of stateless IPv6 addressing. In Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research, CSIIRW '10, (pp. 52:1-52:4). New York, NY: ACM.

Hauser, V. (2012). THC-IPv6: The hackers choice. Retrieved April 23, 2012, from http://www.thc.org/thc-ipv6

Huawei Technologies Corp. & BUPT. (2009). IPv6-send-CGA an implementation of SEND protocol in LINUX kernel. Retrieved from http://code.google.com/p/ipv6-send-cga/

Internet Assigned Numbers Authority. (2012). IANA IPv4 address space registry. Retrieved from http://www.iana.org/assignments/ipv4-address-space/

Internet Society. World IPv6 Launch. (2012). Internet society. Retrieved April 23, 2012, from http://www.worldipv6day.org/

Jiang, S., & Xia, S. (2012). Configuring cryptographically generated addresses (CGA) using DHCPv6. Retrieved from http://tools.ietf.org/html/draft-ietf-dhc-cga-config-dhcpv6-02

Kent, S., & Seo, K. (2005). Security architecture for the internet protocol. RFC4301. Retrieved from http://tools.ietf.org/html/rfc4301

Kukec, A., & Zeeb, B. A. (2010). Native SeND kernel API for* BSD. [Tokyo, Japan: Tokyo University of Science.]. *Proceedings of AsiaBSDCon, 2010*, 1–9.

Levy-Abegnoli, E., Van de Velde, G., Popoviciu, C., & Mohacs, J. (2011). IPv6 router advertisement guard. RFC 6105. Retrieved from http://tools.ietf.org/html/rfc6105

Lynn, C., Kent, S., & Seo, K. (2004). X.509 extensions for IP addresses and AS identifiers. RFC 3779. Retrieved from http://tools.ietf.org/html/rfc3779

Moore, N. (2006). Optimistic duplicate address detection (DAD) for IPv6. Retrieved from http://tools.ietf.org/html/rfc4429

Narten, T., Draves, R., & Krishnan, S. (2007). Privacy extensions for stateless address autoconfiguration in IPv6. RFC 4941. Retrieved from http://tools.ietf.org/html/rfc4941

Narten, T., Nordmark, E., Simpson, W., & Soliman, H. (2007). Neighbor discovery for IP version 6 (IPv6). RFC 4861. Retrieved from http://tools.ietf.org/html/rfc4861

Nikander, E. P., Kempf, J., & Nordmark, E. (2004). IPv6 neighbor discovery (ND) trust models and threats. RFC 3756. Retrieved from http://tools.ietf.org/html/rfc3756

Nikander, P. (2002). Denial-of-service, address ownership, and early authentication in the IPv6 world. In Proceedings of the 9th International Workshop on Security Protocols, (pp. 12–21). London, UK: Springer-Verlag. Retrieved from http://dl.acm.org/citation.cfm?id=647219.720869

O'shea, G., & Roe, M. (2001). Child-proof authentication for MIPv6 (CAM). *ACM SIGCOMM Computer Communication Review*, *31*(2), 4–8. doi:10.1145/505666.505668.

Perkins, C., Johnson, D., & Arkko, J. (2011). Mobility support in IPv6. RFC 6275. Retrieved from http://tools.ietf.org/html/rfc6275

Rafiee, H., AlSa'deh, A., & Meinel, C. (2011). WinSEND: Windows secure neighbor discovery. In Proceedings of the 4th International Conference on Security, SIN 2011, (pp. 243-246). Sydney, Australia: ACM.

Rafiee, H., AlSa'deh, A., & Meinel, C. (2012). Multicore-based auto-scaling secure neighbor discovery for windows operating systems. In Proceedings of the 2012 International Conference on Information Networking, ICOIN 2012, (pp. 269-274). Bali, Indonesia: IEEE.

Thomson, S., Narten, T., & Jinmei, T. (2007). IPv6 stateless address autoconfiguration. RFC 4862. Retrieved from http://tools.ietf.org/html/rfc4862

ADDITIONAL READING

Arkko, J., Aura, T., Kempf, J., Mäntylä, V.-M., Nikander, P., & Roe, M. (2002). Securing IPv6 neighbor and router discovery. In Proceedings of the 1st ACM Workshop on Wireless Security, WiSE '02, (pp. 77–86). New York, NY: ACM.

Arkko, J., & Nikander, P. (2004). Weak authentication: How to authenticate unknown principals without trusted parties. In Christianson, B., Crispo, B., Malcolm, J. A., & Roe, M. (Eds.), *Security Protocols* (pp. 5–19). Berlin, Germany: Springer. doi:10.1007/978-3-540-39871-4_3.

Arkko, J., & Nikander, P. (2005). Limitations of IPsec policy mechanisms. In Christianson, B., Crispo, B., Malcolm, J. A., & Roe, M. (Eds.), *Security Protocols* (pp. 241–251). Berlin, Germany: Springer. doi:10.1007/11542322_29.

Sarikaya, B., Xia, F., & Zaverucha, G. (2011). Lightweight secure neighbor discovery for low-power and lossy networks. Retrieved from http://tools.ietf.org/html/draft-sarikaya-6lowpan-cgand-02

Zhang, J., Liu, J., Xu, Z., Li, J., & Ye, X. (2007). TRDP: A trusted router discovery protocol. In Proceedings of the International Symposium on Communications and Information Technologies, 2007, (pp. 660 –665). ISCIT.

KEY TERMS AND DEFINITIONS

ADD: Authorization Delegation Discovery is a process by which a SEcure Neighbor Discovery (SEND) node can acquire a certificate path, from a router to a trust anchor, in order to certify the authority of a router. A trust anchor is a third party that the host trusts, and to which the router has a certification path. SEND uses two messages for ADD process: Certificate Path Solicitation (CPS) and Certificate Path advertisement (CPA).

ARP: Address Resolution Protocol is used by the Internet Protocol version 4 (IPv4) to associate IP network addresses with hardware addresses. It is used when IPv4 is used over Ethernet.

CGAs: Cryptographically Generated Addresses are IPv6 addresses where the interface identifier portion of the address is generated by hashing the address owner's public key and other parameters. The address owner uses the corresponding private key to assert address ownership and to sign messages sent from that address.

DAD: Duplicate Address Detection is a procedure for verifying the uniqueness of the addresses on a link before they are assigned to an interface. The DAD algorithm is performed on all IPv6 tentative addresses, independently, whether they are obtained via stateless autoconfiguration or DHCPv6.

DoS Attack: A Denial-of-Service attack is an attempt made to make a machine or other network resources unavailable to legitimate users.

IID: Interface Identifier is comprised of the last 64 bits of an IPv6 address. It can be determined using different methods: Extended Unique Identifier (EUI-64), randomly-generated interface identifier that changes over time, or manually configured.

NDP: Neighbor Discovery Protocol is a protocol contained within Internet Protocol Version 6 (IPv6). It is a part of ICMPv6. It is used by IPv6 nodes on the same link to discover each other's presence, to determine each other's link-layer addresses, to find routers, and to maintain reachability information about the paths to active neighbors.

SEND: SEcure Neighbor Discovery is designed as a countermeasure to NDP threats. It enhances NDP with three additional capabilities: Address ownership proof based upon Cryptographically Generated Addresses (CGAs), Message replay protection by including Nonce and Timestamps, and Router authorization to enable routers to act as default gateways based on X.509 certificates.

SLAAC: StateLess Address Auto-Configuration is a process by which an IPv6 node (host or router) can configure IPv6 addresses for interfaces automatically, when connected to a routed IPv6 network using the Neighbor Discovery Protocol. The node builds various IPv6 addresses by combining an address prefix with either the Media Access Control (MAC) address of the node or a user-specified interface identifier. The prefixes include the link-local prefix (fe80::/10) and prefixes of length 64 that are advertised by local IPv6 routers via a Router Advertisement (RA) message.