

SEcure Neighbour Discovery (SEND)

Arun Raghavan

Department of Computer Science
IIT Kanpur

CS625: Advanced Computer Networks

Outline

- 1 The Problem
 - Neighbour Discovery
 - Problems with Neighbour Discovery
- 2 SEcure Neighbour Discovery
 - Overview
 - Cryptographically Generated Addresses
 - SEND Protocol Options
 - Authorisation Delegation Discovery
- 3 Miscellanea

Outline

- 1 The Problem
 - Neighbour Discovery
 - Problems with Neighbour Discovery
- 2 SEcure Neighbour Discovery
 - Overview
 - Cryptographically Generated Addresses
 - SEND Protocol Options
 - Authorisation Delegation Discovery
- 3 Miscellanea

Neighbour Discovery Messages

- Part of ICMPv6
- Neighbour Solicitation/Discovery
 - Neighbour Discovery
 - Address resolution
 - Neighbour Unreachability Detection
 - Duplicate Address Detection
- Router Solicitation/Discovery (node configuration)
 - Router/Prefix Discovery
 - Address (auto)configuration
- Redirect (router provides a better first-hop router)

Outline

- 1 The Problem
 - Neighbour Discovery
 - Problems with Neighbour Discovery
- 2 SEcure Neighbour Discovery
 - Overview
 - Cryptographically Generated Addresses
 - SEND Protocol Options
 - Authorisation Delegation Discovery
- 3 Miscellanea

The Problem: Security

- No proper way to authorise/authenticate nodes

From RFC 2461

A node SHOULD include an Authentication Header when sending Neighbor Discovery packets if a security association for use with the IP Authentication Header exists for the destination address. The security associations may have been created through manual configuration or through the operation of some key management protocol.

The Problem: Security

- No proper way to authorise/authenticate nodes

From RFC 2461

A node SHOULD include an Authentication Header when sending Neighbor Discovery packets if a security association for use with the IP Authentication Header exists for the destination address. The security associations may have been created through manual configuration or through the operation of some key management protocol.

Some Attacks

- Neighbour Advertisement Spoofing
 - Can redirect messages intended for any other node on link
 - Can cause a DoS during NUD/DAD
- Router Advertisement Spoofing
 - Fake Redirect
 - Can provide wrong (malicious) autoconfiguration parameters
 - 'Kill' the router(s) – if default router list at the client is empty, all nodes are treated as on-link
- Replay attacks

Some Attacks

- Neighbour Advertisement Spoofing
 - Can redirect messages intended for any other node on link
 - Can cause a DoS during NUD/DAD
- Router Advertisement Spoofing
 - Fake Redirect
 - Can provide wrong (malicious) autoconfiguration parameters
 - 'Kill' the router(s) – if default router list at the client is empty, all nodes are treated as on-link
- Replay attacks

Some Attacks

- Neighbour Advertisement Spoofing
 - Can redirect messages intended for any other node on link
 - Can cause a DoS during NUD/DAD
- Router Advertisement Spoofing
 - Fake Redirect
 - Can provide wrong (malicious) autoconfiguration parameters
 - 'Kill' the router(s) – if default router list at the client is empty, all nodes are treated as on-link
- Replay attacks

Possible (infeasible) Solutions

- Single shared secret
 - Weak security
- IKE with manual Security Associations
 - Not scalable
 - Public-key crypto preferable
- 802.1X (relatively recent for wired networks)
 - Need relatively complex infrastructure
- None of these are really feasible for public access networks

Possible (infeasible) Solutions

- Single shared secret
 - Weak security
- IKE with manual Security Associations
 - Not scalable
 - Public-key crypto preferable
- 802.1X (relatively recent for wired networks)
 - Need relatively complex infrastructure
- None of these are really feasible for public access networks

Possible (infeasible) Solutions

- Single shared secret
 - Weak security
- IKE with manual Security Associations
 - Not scalable
 - Public-key crypto preferable
- 802.1X (relatively recent for wired networks)
 - Need relatively complex infrastructure
- None of these are really feasible for public access networks

Possible (infeasible) Solutions

- Single shared secret
 - Weak security
- IKE with manual Security Associations
 - Not scalable
 - Public-key crypto preferable
- 802.1X (relatively recent for wired networks)
 - Need relatively complex infrastructure
- None of these are really feasible for public access networks

Outline

- 1 The Problem
 - Neighbour Discovery
 - Problems with Neighbour Discovery
- 2 SEcure Neighbour Discovery
 - **Overview**
 - Cryptographically Generated Addresses
 - SEND Protocol Options
 - Authorisation Delegation Discovery
- 3 Miscellanea

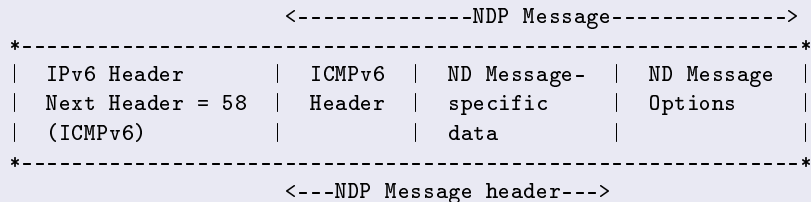
Key Ideas

- Routers authorised by “trust anchors”
- Cryptographically Generated Addresses to prevent spoofing/hijacking
- Digital signatures for all NDP messages

Integrating with IPv6

- For each feature, there is an option that is plugged into the format shown below
- Allows for backwards compatibility as well as extensibility

NDP Message Options in IPv6



Outline

- 1 The Problem
 - Neighbour Discovery
 - Problems with Neighbour Discovery
- 2 **SEcure Neighbour Discovery**
 - Overview
 - **Cryptographically Generated Addresses**
 - SEND Protocol Options
 - Authorisation Delegation Discovery
- 3 Miscellanea

CGA

- We need a way to bind an IP address to a host
- Complex (or *any*) infrastructure won't work for networks that are not tightly controlled
- CGA provides a cryptographic binding between a host and its IP address
 - Without the introduction of *any* new infrastructure

CGA

- We need a way to bind an IP address to a host
- Complex (or *any*) infrastructure won't work for networks that are not tightly controlled
- CGA provides a cryptographic binding between a host and its IP address
 - Without the introduction of *any* new infrastructure

CGA

- We need a way to bind an IP address to a host
- Complex (or *any*) infrastructure won't work for networks that are not tightly controlled
- CGA provides a cryptographic binding between a host and its IP address
 - Without the introduction of *any* new infrastructure

CGA

- We need a way to bind an IP address to a host
- Complex (or *any*) infrastructure won't work for networks that are not tightly controlled
- CGA provides a cryptographic binding between a host and its IP address
 - Without the introduction of *any* new infrastructure

CGA Format

- Each node has a public-private key pair
- Address consists of three parts
 - u and g fields
 - Sec parameter
 - 59-bit address
- $CGAHash^1$ is computed with the segment prefix, $CGAHash^2$ is computed without

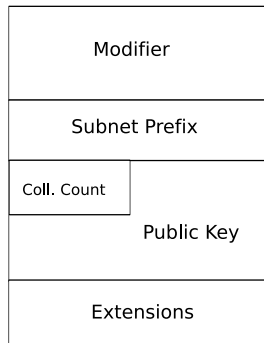


$$Address = CGAHash_{64}^1 \& 0x1CFF...$$

$$MSB_{16 * Sec}(CGAHash_{112}^2) = 0$$

CGA Generation

- 1 Generate random modifier
- 2 Construct CGA parameters as in the figure
- 3 $CGAHash^2 = SHA1(CGA\ Parameters)$
- 4 If $MSB_{16 * Sec}(CGAHash^2) = 0$ goto 6^a
- 5 Increment modifier, goto 2
- 6 IP address is concatenation of subnet prefix and $MSB_{64}(CGAHash^1)$
- 7 If there is a collision, increment collision count (at most 2), goto 2
- 8 We have an IP address and CGA Parameters



^aThis technique is called “hash extension”

CGA Verification

- The CGA Parameters are sent with the packet as an IPv6 option
- ① Verify that the collision count is less than 3
- ② The subnet prefix of the address and the CGA Parameters must match
- ③ Calculate the address from the CGA Parameters as during generation – this must be the same as the interface identifier from which the packet was received
- ④ Using the *Sec* parameter from the interface identifier, and CGA Parameters, calculate the SHA-1 hash – the leftmost $16 * Sec$ bits must be 0

CGA Security and Performance

- Prevents stealing and spoofing
- Does *not* provide authorisation or authentication
- Address generation is $O(2^{16*Sec})$
 - Higher *Sec* values are intentionally made infeasible
 - Should not be obsoleted by faster computing soon
 - Since hash extension is done without the prefix, mobile hosts do not need to do the search again when moving
- Attacking a CGA is $O(2^{59+16*Sec})$
 - Difficulty is proportional to the *Sec* parameter

Outline

- 1 The Problem
 - Neighbour Discovery
 - Problems with Neighbour Discovery
- 2 **SEcure Neighbour Discovery**
 - Overview
 - Cryptographically Generated Addresses
 - **SEND Protocol Options**
 - Authorisation Delegation Discovery
- 3 Miscellanea

The CGA Option

- The CGA Parameters structure as described earlier is just padded and sent

The RSA Signature Option

- Includes a key hash of the public key for identifying which key a host is using
 - Receiver has the public key (maybe stored previously, or received in the CGA or Certificate Options)
- Digital signature made by concatenating
 - A 128-bit *Type* field
 - Source address
 - Destination address
 - Some ICMP fields
 - NDP message header
 - All NDP options before the signature
- Computation is expensive – do rate-limiting to prevent DoS

Timestamp Option

- Used for unsolicited advertisements, to avoid replay attacks
- Contains the time since Jan 1, 1970, 00:00 UTC
- 64 bits – 48 for seconds, 16 for $\frac{1}{64K}$ seconds
- Synchronisation done by some means (can be NTP)
- Three parameters for checking
 - DELTA: Maximum difference between the Timestamp on a message and the time of receipt (only for new peers)
 - DRIFT: Maximum drift between clock of sender and receiver
 - FUZZ: Allows for some “fuzziness” in the constraints

The Nonce Option

- Used to make sure that a response to a solicited message is “fresh”
- A random number of at least 6 bytes
- Sent with a solicitation message
- The reply advertisement **MUST** contain the same nonce in return
- The RSA signature ensures that the reply is not a replay, and has not been tampered with

Outline

- 1 The Problem
 - Neighbour Discovery
 - Problems with Neighbour Discovery
- 2 SEcure Neighbour Discovery
 - Overview
 - Cryptographically Generated Addresses
 - SEND Protocol Options
 - Authorisation Delegation Discovery
- 3 Miscellanea

Trust Anchors and Authorisation Paths

- Every client has a set of public keys for “trust anchors” – entities that are trusted to certify routers as trusted
- For each router, we derive an “authorisation path”
 - A path is a set of digitally signed certificates (X.509, with IP extension)
 - Path starts at the trust anchor
 - Each node authorises the next in the path with a subset of the prefixes that it is authorised to handle
 - Null prefix is used to authorise for *all* prefixes
- Must be able to handle revocation lists
- Rate-limiting to avoid flooding/DoS

Trust Anchors and Authorisation Paths

- Every client has a set of public keys for “trust anchors” – entities that are trusted to certify routers as trusted
- For each router, we derive an “authorisation path”
 - A path is a set of digitally signed certificates (X.509, with IP extension)
 - Path starts at the trust anchor
 - Each node authorises the next in the path with a subset of the prefixes that it is authorised to handle
 - Null prefix is used to authorise for *all* prefixes
- Must be able to handle revocation lists
- Rate-limiting to avoid flooding/DoS

Trust Anchors and Authorisation Paths

- Every client has a set of public keys for “trust anchors” – entities that are trusted to certify routers as trusted
- For each router, we derive an “authorisation path”
 - A path is a set of digitally signed certificates (X.509, with IP extension)
 - Path starts at the trust anchor
 - Each node authorises the next in the path with a subset of the prefixes that it is authorised to handle
 - Null prefix is used to authorise for *all* prefixes
- Must be able to handle revocation lists
- Rate-limiting to avoid flooding/DoS

Trust Anchors and Authorisation Paths

- Every client has a set of public keys for “trust anchors” – entities that are trusted to certify routers as trusted
- For each router, we derive an “authorisation path”
 - A path is a set of digitally signed certificates (X.509, with IP extension)
 - Path starts at the trust anchor
 - Each node authorises the next in the path with a subset of the prefixes that it is authorised to handle
 - Null prefix is used to authorise for *all* prefixes
- Must be able to handle revocation lists
- Rate-limiting to avoid flooding/DoS

Trust Anchors and Authorisation Paths

Example of an Authorisation Path

- Node CA is the “trust anchor”, whose certificate is available at the client, C_1
 - CA is authorised for prefix P_0
- Router R_2 is advertising itself to C_1 for prefix P_2
- An authorisation path might look like
 - CA certifies router R_1 for prefix P_1 , a subset of P_0
 - R_1 certifies R_2 for prefix P_2 which is a subset of P_1

Certification Path Discovery

- Uses two new ICMPv6 messages certificate Path Solicitation/Advertisement
 - Other (unspecified) methods may be used as well
 - Can also be used by hosts to authenticate each other (if set up that way)
- Certificate Path Solicitation
 - Protected by a nonce
 - Can specify trust anchor(s) that are supported
 - Can also select which certificate (by a “component” number)

Certification Path Discovery

- Uses two new ICMPv6 messages certificate Path Solicitation/Advertisement
 - Other (unspecified) methods may be used as well
 - Can also be used by hosts to authenticate each other (if set up that way)
- Certificate Path Solicitation
 - Protected by a nonce
 - Can specify trust anchor(s) that are supported
 - Can also select which certificate (by a “component” number)

Certification Path Discovery

- Certificate Path Advertisement
 - Returns the nonce
 - Specifies the number of certificates in the path
 - Should put one certificate per message to avoid fragmentation
 - Should send in order such that each certificate can be verified immediately after the previous certificate
 - Can also specify trust anchor, so uninterested nodes may ignore the advertisement

Transition

- As with IPv6, we need some time for nodes and routers to support SEND
- Nodes
 - Should send only secured (signed and validated) messages
 - Should prefer SEND routers over unsecured ones
 - Unsecured updates should not affect entries made by secured messages





Limitations

- Encryption is not provided
- No protection for link-layer
- Proxy ND (mainly for MIPv6) not supported
 - One solution could be to have a node provide a certificate to the proxy authorising it to act on it's behalf

Q&A

Thanks!

References

-  [RFC 3756](#)
IPv6 Neighbor Discovery (ND) Trust Models and Threats
-  [RFC 3971](#)
SEcure Neighbor Discovery (SEND)
-  [RFC 3972](#)
Cryptographically Generated Addresses (CGA)
-  [Tuomas Aura](#)
Cryptographically Generated Addresses (CGA)
[Information Security Conference 2003](#)