IPv6 maintenance Working Group (6man)
INTERNET-DRAFT
Updates RFC 3971
(if approved)
Intended status: Proposed Standard
Expires: January 15, 2014

H. Rafiee
C. Meinel
Hasso Plattner Institute

July 15, 2013

A Simple Secure Addressing Scheme for IPv6 AutoConfiguration (SSAS)

<draft-rafiee-6man-ssas-05.txt>

Abstract

The default method for IPv6 address generation uses an Organizationally Unique Identifier (OUI) assigned by the IEEE Standards Association and an Extension Identifier assigned to the hardware manufacturer [1] (section 2.5.1 RFC-4291) [RFC4291]. This fact thus means that a node will always have the same Interface ID (IID) whenever it connects to a new network. Because the node's IP address does not change, the node will be vulnerable to privacy related attacks. Currently this problem is addressed by the use of two mechanisms that do not make use of the MAC address, or other unique values that can be used for ID generation, for randomizing the IID; Cryptographically Generated Addresses (CGA) [RFC3972] and Privacy Extension [RFC4941]. The problem with the former approach is the computational cost involved for the IID generation and in the verification process. The problem with the latter approach is that it lacks necessary security mechanisms and provides the node with only partial protection against privacy related attacks. This document proposes the use of a new algorithm for use in the generation of the IID while, at the same time, securing the node against some types of attack, like IP spoofing. These attacks are prevented by the addition of a signature to messages sent over the network and by finding a binding with the nodes' IP address and its public key. The use of theResource Public Key Infrastructure (RPKI), introduced in this document, is based on the centralized version explained in RFC 6494 and RFC 6495.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current.

Copyright Notice

Table of Contents

1. Introduction

IPv6 addresses consist of two parts; the subnet prefix, which is the 64 leftmost bits of the IPv6 address, and the Interface ID (IID), which is the 64 rightmost bits of the IPv6 address. The IEEE Standards Association [1] (section 2.5.1 RFC-4291) [RFC4291] offered a standard for the generation of IPv6 Interface IDs (IID) called the Extended Unique Identifier (EUI-64). EUI-64s are generated by the concatenation of an Organizationally Unique Identifier (OUI), assigned by the IEEE Registration Authority (IEEE RA), with the Extension Identifier assigned by the hardware manufacturer. For example, if a manufacturer's OUI-36 hexadecimal value is 00-5A-D1-02-3, and the manufacture hexadecimal value, for the Extension Identifier for a given component is 4-42-61-71, then the EUI-64 value generated from these two numbers will be 00-5A-D1-02-34-42-61-71. If the OUI is 24 bits and the extension identifier is also 24 bits (this constitutes the MAC address), then to form the 64-bit EUI address, the OUI portion of the MAC address is inserted into the leftmost 24 bits of the EUI-64 8 byte field and the Extension Identifier is inserted into the rightmost 24 bits of the EUI-64 8 byte field. A value of 0xFFFE is then inserted between these two 24-bit items. IEEE has chosen 0xFFFE as a reserved value which can only appear in an EUI-64 which is generated from an EUI-48 MAC address. Bit 7 (u bit) in the OUI portion of the address is used to indicate either global or local uniqueness. Globally unique addresses assigned by the IEEE set this bit to zero, by default,indicating global uniqueness.The bit is set to 1 for locally created addresses, such as those used for virtual interfaces or a MAC address manually configured by an administrator.

There are currently some mechanisms used to generate a randomized IID that do not make use of a MAC address; CGA [RFC3972], Privacy Extension (generation of temporary addresses) [RFC4941], etc. In this document we discuss the problem inherent with using the current mechanisms and then we explain our solution to the problem, which is to randomize the IID observing privacy, while, at the same time, providing security to Neighbor Discovery Protocol (NDP) messages of nodes in the IP layer. DHCPv6 [RFC3315] can also benefit from this approach for the generation of a random IID or for authentication purposes.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this

document are to be interpreted as described in RFC-2119 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be

interpreted as carrying RFC-2119 significance.

In this document the use of || indicates the concatenation of the values on either side of the sign.

3. Problem Statement

The drawback to using IIDs that do not change over time is one of privacy. The node will generate the same IID whenever it joins a new network thus making it easy for an attacker to track that node when it moves to different networks.

The main problem with the privacy extension mechanism, when using the first approach as explained in section 3.2.1 RFC-4941 [RFC4941], i.e., using stable storage, is the lack of a provision for the use of a security mechanism and also the need to generate public addresses based on MAC addresses. The Privacy Extension RFC partially prevents attacks related to privacy issues, but it cannot prevent attacks related to security issues. For example, it cannot prevent IP spoofing attacks and it cannot provide proof of IP address ownership for a node. If one wants to use a secure method, with the privacy extension, then one needs to use CGA. The problem with using CGA is in the computational overhead necessary to compute it when higher sec values are used and the time that is needed to perform the verification process. This time is based on the reverse of the steps required for the CGA regeneration during the verification process along with the additional time needed for signature verification.

The first problem with CGA is the apparent lack of a defense against Denial of Service (DoS) types of attack that are performed against verifier nodes. In the CGA RFC there is no explanation as to how to prevent these types of attacks. This means that an attacker can overwhelm the verifier node with false CGA values thus rendering it unable to process further messages.This document also proposes a solution to this type of attack. The other problem with CGA sec value higher than 0 is unnecessary making busy the CPU and other resources in a node for unlimited period of time. It is because there is no guarantee that the 16 by sec value equal to zero condition will ever be met. So the use of the CGA algorithm, which is compute intensive, is thus not ideal for use with nodes having limited resources or with nodes wanting to change their IID frequently for the purpose of protecting their privacy.

In order to overcome the problem with using the other mechanisms, the time needed for IP address generation and verification needs to be reduced and avoid unnecessary usage of CPU while at the same not scarifying user's security. We propose the use of the SSAS algorithm,

along with the SSAS signature, to provide a node with the protection it needs to protect it against IP spoofing and other spoofing types of attack in the IP layer. Our experimental results [2] show that SSAS is more secure and faster than CGA when using a sec value of 0

(uses 62 bits (when using first SSAS algorithm) while CGA uses 59 bits) and much faster than CGA when using a sec value of 1. The security of SSAS, when using second algorithm, is about the same as the security of the whole public key while in CGA it depends on the sec value. It is not also ideal to use CGA with sec value higher than 1 when using the current hardware resources. This is because it will take hours to years to generate an IP address.

Note: It is not the intent of this document to obsolete CGA but to propose a simpler, faster and high security addressing mechanism for use in providing nodes with network layer privacy and security. This is accomplished by providing a node with two algorithms to be used to randomize the IID while at the same time providing nodes protection against the types of attack explained below.

3.1. SSAS Applications

3.1.1. Preventing Attacks

The following sections detail some types of attack that SSAS can prevent.

3.1.1.1. Replay attack

In this type of attack, an attacker will sniff the Neighbor Discovery Protocol enabled network (NDP) messages to find, and then copy, a legitimate signature and public key to his own NDP message which he will then send to the original sender. But with the use of the SSAS algorithm (Including the timestamp in the signature) and using RPKI introduced in this document, this can be prevented. The use of a timestamp works because the timestamp will be valid for only a short period of time. (this accounts for clock skews.)

3.1.1.2. IP spoofing

This is a well-known type of attack in NDP. This type of attack is used against the Duplicate Address Detection process. In this attack, when a node joins the network and generates a new IP address, the node sends a Neighbor Solicitation (NS) message to check for address collisions in the network. The attacker, in this scenario, spoofs the IP address and responds back to the node with a Neighbor Advertisement (NA) message claiming ownership of this IP address.

While the SSAS algorithm does allow this node to verify other nodes in the network, an attacker will not have the private key associated with this node which is needed for SSAS signature generation, so the

verification process will fail.

3.1.1.3. Denial of Service (DoS) attacks

An attacker might send many NDP messages, using invalid signatures, to a victim's node which then forces the node to busy itself with the verification process. To mitigate this attack, a node SHOULD set a limit on the number of messages (x) that should be verified within a certain period of time. Implementations MUST provide a conservative default and SHOULD provide a means for detecting when this limit is reached.

3.1.1.4. Spoofed Redirect Message

Redirect messages, imitating the end host needing redirection, can be sent from any router on the same broadcast segment. The attacker uses the link-local address of the current first-hop router in order to send a Redirect message to a legitimate node. Since that node identifies the message as coming from its first hop router, by use of the link-local address, it accepts the Redirect. The Redirect will remain in effect as long as the attacker responds to the Neighbor Unreachability Detection probes sent to the link-layer address. To preclude this from occurring, the address ownership of the first-hop router should be verified. The use of the SSAS verification process along with RPKI will prevent such an attack.

3.1.2. Nodes with limited resources

SSAS can be used in nodes where limited computational resources are available. It can provide protection to these nodes against the types of attack stated above. Sensor networks are a prime example of nodes with limited resources (such as battery, CPU, and etc); see RFC-4919 [RFC4919] for use in IPv6 networks. Because currently, as explained in section 4. RFC-6775, the generation of the IID is based on EUI-64 which makes these nodes vulnerable to privacy and security attacks. One of these types of attack can occur during the Duplicate Address Detection (DAD) process.

Another example for the use of SSAS would be in mobile networks during the generation of IP addresses, as explained in section 4.4 RFC-6275 [RFC6275]. The current problem with the addressing mechanism in a mobile node is that no privacy is observed when a node moves to another network while usually keeping its Home Address. If there were a fast and secure mechanism available, then it would be possible to

set this Home Address and change it and re-register it to the Home network. Another possible use for SSAS in mobile nodes could be as a security mechanism during the configuration of Care of Address (CoA);

see section 3. RFC-5213 [RFC5213]. In that RFC, home proxy plays the role of a home agent for mobile nodes and mobile nodes set their CoA by the use of either stateful or stateless autoconfiguration. Currently they MUST use IPsec in order to secure this process. Section 4 of that RFC discusses the possibility of using another algorithm in order to secure mobile nodes.

3.1.3. Other Applications

With the wide usage of IP addresses in different types of devices and by the use of autoconfiguration mechanisms to configure these IP addresses, the need for the use of a security algorithm is increased. One type of application would for use in vehicular networks or car by car networks. There is currently some work in progress that makes use of Neighbor Discovery. SSAS could also be a solution for enabling fast protection against ND attacks.

4. Algorithms Overview

As explained earlier, one of the problems with using the current IID generation approach is the compute intensive processing that is needed for the IID algorithm generation. Another concern is for the lack of security. Since we assume that a node will need to generate and keep its address for a short period of time, we have tried to keep the IID generation process to a minimum. We have also tried to remain within the confines of NDP protocol. Here we offer two algorithms. The first algorithm is used where the purpose is a fast algorithm with the security higher than CGA sec value 0. The second algorithm addresses the problem with the security level and tries to use the security of the whole public key.

4.1. SSAS First Algorithm (SSAS v1)

4.1.1. Interface ID (IID) Generation

To generate the IID a node will need to execute the following steps.

1. Generate key pairs (public/private keys) using ECC (RFC 6090) or other available algorithms. ECC is the default algorithm, but any algorithm capable of generating a small key size in a short amount of time is viable. It is best to have the key pairs generated, on the fly, during the start-up phase of the algorithm generation. These

keys SHOULD be valid for only a certain period of time which depends on network policy. When the time expires for the use of these key pairs, the node will generate new key pairs. It then uses this new

value for the generation of the IP address and signature. Comparing the use of ECC to that of RSA shows that an ECC with a 192 bit key is equivalent to a RSA with a 7680 bit key (according to US National Security Agency) In this case the packet size would be decreased by a factor 11 times smaller than that when using RSA.

2. Divide the public key array of bytes into two half byte arrays (see figure 1). Obtain the first 4 bytes from the first half byte array and call it the partial IID1. Obtain the first 4 bytes of the second half byte array and call this the partial IID2.

3. Concatenate partial IID1 with partial IID2 and call this the IID. Set bits u and g to one.

4. Concatenate the IID with the local subnet prefix to set the local IP address

5. Concatenate the IID with the router subnet prefix (Global subnet prefix), obtained from the Router Advertisement (RA) message, and set it as a tentative public IP address. This IP address will become permanent after Duplicate Address Detection (DAD) processing. (for more information about DAD refer to section 4.1.3.)

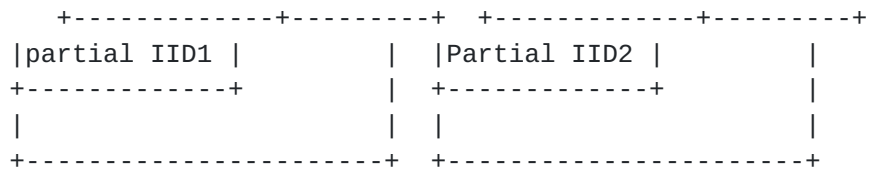
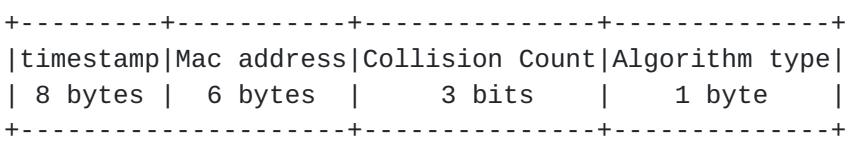


Figure 1 Public key divided into two halves

4.1.2. Signature Generation

The SSAS signature is added to NDP messages in order to protect them from IP spoofing and spoofing types of attack. SSAS will provide proof of IP address ownership, as does the CGA generation algorithm, but by using fewer steps. To generate the SSAS signature, the node needs to execute the following steps:

1. Concatenate the timestamp with the MAC address, collision count, algorithm type and the global (public) IP address. (see figure 2)



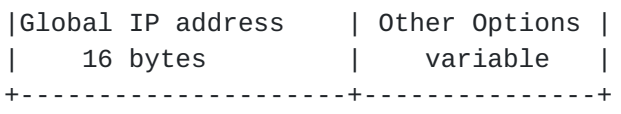

|Global IP address | Other Options |
| 16 bytes | variable |

Figure 2 SSAS Signature format

2. Sign the resulting value from step 1, using the ECC private key, and call the resulting output the SSAS signature.

If NDP messages contain other data that must be protected, such as important routing information, then this data SHOULD also be included in the signature. The signature is designed for the inclusion of any data needing protection. If there is no data that needs protection, then the signature will only contain the timestamp, MAC address, Collision count and Global IP address (Router subnet prefix plus IID).

4.1.3. Generation of NDP Messages

After a node generates its IP address, it should then process Duplicate Address Detection in order to avoid address collisions in the network. In order to do this the node needs to generate a Neighbor Solicitation (NS) message. The SSAS signature is added to the ICMPv6 options of NS messages. The SSAS signature data field is an extended version of the standard format of the RSA signature option of SeND [RFC3971]. The timestamp option is the same as that used with SEND. In the SSAS signature, the data field contains the following items: type, length, reserved, Other Len, algorithm type, collision count, subnet prefix, other option and padding.

4.1.3.1. SSAS signature data field

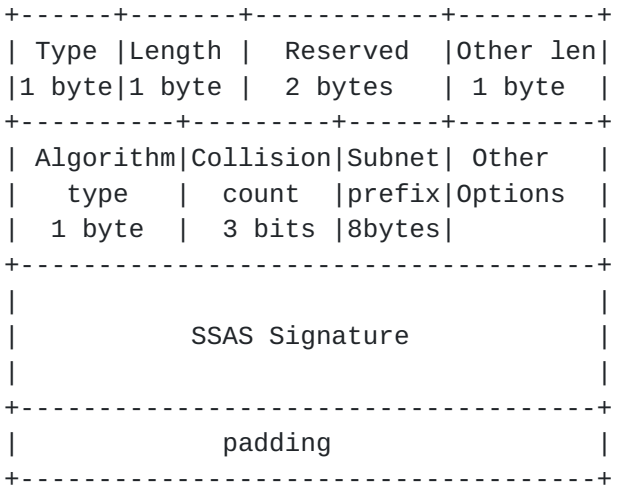


Figure 3 NDP Message Format with SSAS Signature Data Field

- Type: This option is set to 15. This is the sequential number used

in SeND to indicate a SSAS data field.

- Length: The length of the Signature Data field, including the Type, Length, Reserved, Algorithm type, Signature and padding, must be a multiple of eight.

- Reserved: A 2 byte field reserved for future use. The value must be initialized to zero by the sender and should be ignored by the receiver.

- Other Len: The length of other options in multiples of eight. The length of this field is 1 byte.

- algorithm type: The algorithm used to generate key pairs and sign the message. The length of this field is 1 byte. For ECC, this value is 0. Future algorithms will start at one and increase from there.

- Collision count: When a collision occurs during the DAD, the node will increment this value and store it in a file to be included in the sent packets for as long as the current IP address is valid. This value indicates to the node where it needs to start its check from, i.e., the first or second or third bytes from the start of the half byte array of the public key.

- Subnet Prefix: This is the router subnet prefix.

- Other Options. This variable-length field contains important data that needs to be protected in the packet. The padding is used to insure that the field is a multiple of eight in length.

- Padding. A variable-length field containing padding to insure that the entire signature field is a multiple of eight in length. It thus contains the number of blanks needed to make the entire signature field end on a multiple of eight.

All NDP messages (except RS messages) SHOULD contain the SSAS signature data field which allows receivers to verify senders. If a node receives a solicited NA message in response to its NS message showing that another node claims to own this address, then, after a successful verification process, this node increments the collision count by one and this value is used as explained in the "Collision count" item above. It will start from that section of the public key for the generation of a new IP address. If the node receives the same claim three times in a row, then it will consider it as an attack and it will use that IP address.

This document proposes an update to the RFC 3971 in order to include the the SSAS signature data field as an additional field to SeND to be used in place of RSA signature.

4.1.4. SSAS v1 verification process

A node's verification process should start when it receives NDP messages.

Following are the steps used in the verification process:

1. Obtain the timestamp from the NDP message and call this value t1.

2. Obtain the timestamp from the node's system, convert it to UTC, and call this value t2.

3. If (t2- x) < = t1 < = (t2 + x) go to step 4. Otherwise, the message SHOULD be discarded without further processing. The value of x is dependent on network delays and network policy. The implementations MUST choose a flexible value for x based on the delay in this network.

4. Obtain the public key from the CN node or by checking its own neighboring cache. (see section 4.3)

5. Compare this to its own public key. If it is not the same, go to the next step. Otherwise, the message should be discarded without further processing. (This step should be skipped when the node uses the CN node to obtain the other nodes' public key.)

6. Divide the public key into two arrays of byes. Based on the collision count, start from the first, second or third bytes of public key and select 4 bytes from each half byte array and call them partial IID 1 and 2. Concatenate partial IID 1 with partial IID2 and set bits u and g to 1. Obtain the node's source IP address. Compare this value with the node's IID source IP. If it is the same, go to the next step. Otherwise, discard the message without further processing.

7. Concatenate the timestamp with the MAC address, algorithm type, collision count, sender's Global IP address (subnet prefix and IID), and other options (if any) and call this entity the plain message.

8. Obtain the SSAS signature from the SSAS signature data field. Obtain the Algorithm type from the message.

9. Verify the Signature using the public key and then enter the plain message and the SSAS signature as an input to the verification function. If the verification process is successful, process the message. Otherwise, the message should be discarded without further processing.

4.2. SSAS Second Algorithm (SSAS v2)

4.2.1. Interface ID (IID) Generation

1. The first step is the same as what is explained in section 4.1.1. of this document and call the public key Pk

2. execute a function on the public key.

R= F_x(Pk)

F() is the root function which depends on the size of public key. x is the root value. If ECC or Another short key size algorithm is used, then it will be the square root (x=2). R is the IID obtained from the public key. The value is comprised of a 64 bit floating point number obtained from the leftmost bits that includes the numbers before and after the digits in the floating point representation. then the bits u and g are set to one and this will be IID. The implementations SHOULD use the same way of calculating x for the same public key size. This will avoid the need of sending x to the verifier node.

3. Concatenate the IID with the local subnet prefix to set the local IP address

4. Concatenate the IID with the router subnet prefix (Global subnet prefix), obtained from the RA message, and set it as a tentative global IP address. This IP address will become permanent after Duplicate Address Detection (DAD) processing.

4.2.2. SSAS v2 verification process

The first 5 steps of SSAS verification process is the same as the first 5 steps explained in section 4.1.4.

6- Execute F_x(Pk) and compare the resulting value to the nodes' IID (bits u and g SHOULD be ignored). If there is a match the message SHOULD be processed otherwise it SHOULD be considered as an attack and the message SHOULD be discarded without further action.

4.3. Resource Public key Infrastructure (RPKI)

To verify the authorized router in the local network and to create a partial trust within the network, we propose the use of a local centralized Resource Public Key Infrastructure (RPKI) which is based on the centralized version explained in RFC 6494 and RFC 6495. Figure 4 depicts the architecture of this new RPKI framework. In this framework we propose the use of a Controller Node (CN) whereby

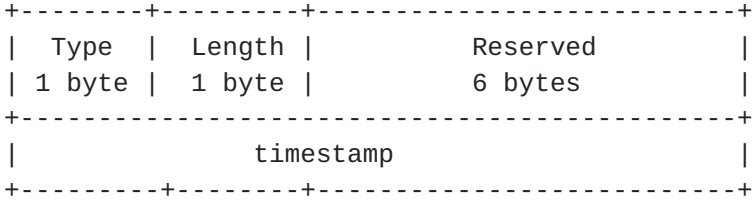
administrators will be able to manually store, in the database of this node, the router's public key and MAC address. We are introducing the use of two different NDP messages, Request Public Key

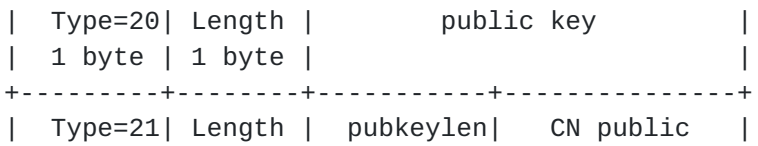
(RPK) and Send Public Key (SPK), that can be used by nodes to request the public key of the router or other nodes, and can be used to send from the CN node the public key of the requested node. The CN node has a fixed IP address and MAC address in the local link. It is a reserved MAC address and IP address which is known to all nodes. One possible way to implement CN is to use a new module in routers capable of processing these two messages. In this case, one CN node could be available in two different local networks when the same router is available between these two networks. When a node first sends a RPK message, the CN node will add its MAC address and public key to its database. This gives other nodes the capability of verifying this new node by asking for the public key of this node. The CN node maintains this data for as long as it receives NS messages from this node. Nodes frequently check neighbor reachability and the CN node receives these messages passively. If the CN node does not see a message from a node that has an entry in its database, then it sends a NS message to that node. If it does not receive a response from that node after a few minutes, it removes that node from its database. Nodes that are added manually to the CN database must be removed manually from the CN database.

4.3.1. Generation of RPK and SPK

Figure 4 shows the format of these two new NDP messages. The NDP message type used for RPK is 140 and for SPK is 141. These are set in the ICMPv6 header. There are two new types in these messages: type 16 and type 17. If a node wants to generate a change to its IP address or generate a new one, it sends a type 16 RPK message which indicates the use of its MAC address and timestamp signed by its old private key. A Type 17 message indicates the use of a node's new public key and the signature generated by signing the MAC address and timestamp with the node's new private key.

When a CN node wants to generate the SPK, it adds the requested public key to the type 16 section of the message and then includes its own public key and generates a signature IID using its own private key created from the concatenation of timestamp with its own MAC address and the values of type 17.




| Type=20| Length | public key |
| 1 byte | 1 byte | |
+---------+--------+-----------+---------------+
| Type=21| Length | pubkeylen| CN public |

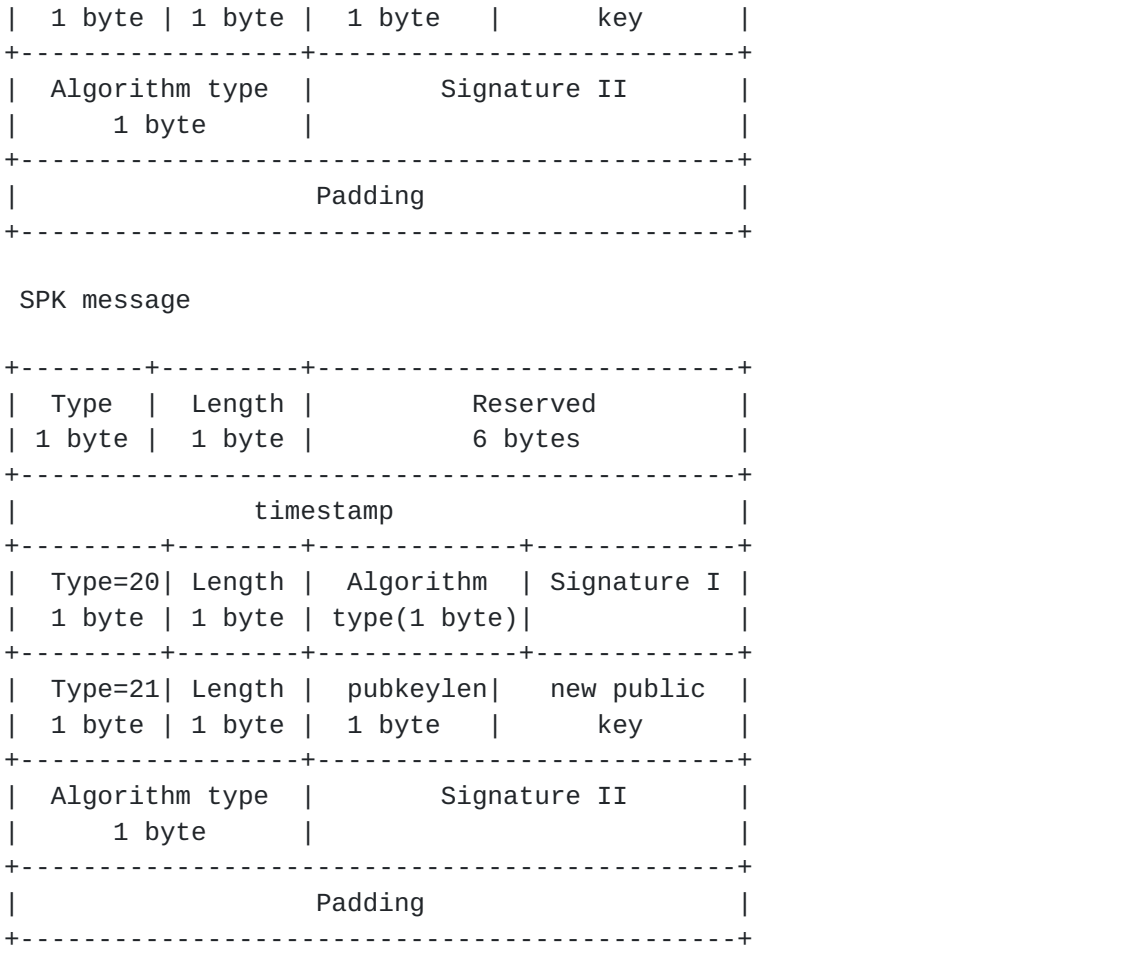


Figure 4 Format of Request Public Key (RPK) and Send Public Key (SPK)

4.3.2. Process of RPK and SPK.

When a new node joins a network, it generates its local IP address using the SSAS algorithm and then sends a Router Solicitation message to obtain a router advertisement and to generate its global IP address. This message does not need to include the SSAS data structure. The router responds to the node with a Router Advertisement (RA). The new node needs to obtain the public key for this router from the CN node. It then generates a RPK. After successful verification, the CN node checks whether or not this MAC address already exists in its database. If it does, it checks to see whether or not the public key is the same as that which is available in its database. If it finds a match, it generates a SPK message and sends it to the node. If the CN node finds a different public key than that of this node, it sends the SPK setting the length of the type 16 option to 1 and setting the one byte public key to 1. This

informs the new node that there is an existing MAC address with a different public key. If this node is the owner of the old public key that is available in the CN node, it includes its old public key as shown in figure 5 and sends a new RPK. Otherwise it considers this an

attack on his MAC address and sets the one byte of the old public key to zero and the length of the type 16 option to one, and sends this message. This message causes a flag to be added to the database to inform the network administrator that something is wrong in this network.

If the public key and MAC address of the new node are not available in the CN, after receiving the RPK message it will add these values to its database.

When the other nodes want to add a new node to their neighboring caches, after receiving the neighbor advertisement message, they will ask the CN node for the public key of this node. After successful verification, they will add the public key, MAC address and IP address of this node to their neighboring cache. The next time they will not need to ask the CN node for any information to check the reachability of the neighboring nodes in their cache. This decreases the number of messages exchanged between the CN node and the other nodes in this network.

5. Security Considerations

As a security consideration what one might ask oneself is what are the odds of an attacker being able to generate a public key having two four sequential bytes (from two different halves of public key) that are the same as 62 bits of that in public key If he could, he could then generate the signature using his own private key and thus break SSAS.

Mathematically it has been shown that the probability of matching 48 bits in the public key against 62 bits in the IID is about pow(1/2,62) where pow is the power function, 2 is a base and 62 is an exponent. in [2] the analysis of SSAS is explained and compared to CGA. For CGA sec value 0, the attacker needs to do brute force attacks against 59 bits. So SSAS v1 is more secure than CGA sec value 0. For SSAS v2, the attacker needs to do brute force attacks against the whole public key. So the security of that is depends on the security of public key algorithm and the key size.

6. IANA Considerations

This document defines two new algorithm for the generation of an Interface ID in IPv6 networks that provides IP layer privacy and

local link security. It also introduces a new, local RPKI based on the centralized RPKI.

7. Conclusions

Privacy has become a very important issue in recent years. A solution for preventing a node from being tracked by an attacker is to change the node's IP address frequently and by generating a random IID each time a node wants to generate a new IP address. This document introduced two new algorithms as a solution for providing privacy by randomizing the IID and for providing security with the addition of a SSAS signature to the NDP message and finding a binding between the public key and the IP address. In SSAS v1, a node directly uses the public key for IID generation. This algorithm is faster than CGA with sec value higher than 1 and more secure than CGA with sec value 0.In SSAS v2, a node uses a root function (depends on the public key) to make use of about the whole security of the public key. This will increase the security of SSAS to the whole public key while at the same time decrease the time require for generation of IID.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4291] Hinden, R., Deering, S., "IP Version 6 Addressing Architecture," RFC 4291, February 2006.

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)," RFC 3972, March 2005.

[RFC4941] Narten, T., Draves, R., Krishnan, S., "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, September 2007.

[RFC3971] Arkko, J., Kempf, J., Zill, B., and Nikander, P., "SEcure Neighbor Discovery (SEND)", RFC 3971, March 2005.

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., Carney, M. , " Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

[RFC4919] Kushalnagar, N., Montenegro, G., Schumacher, C.,"
IPv6 over Low-Power Wireless Personal Area Networks
(6LoWPANs): Overview, Assumptions, Problem Statement, and

Goals", RFC 4919, August 2007.

[RFC6775] Shelby, Z., Chakrabarti, S., Nordmark, E., Bormann, C. , " Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, November 2012.

[RFC6275] Perkins, C., Johnson, D., Arkko, J., "Mobility Support in IPv6", RFC 6275, July 2011.

[RFC6543] Gundavell, S., "Reserved IPv6 Interface Identifier for Proxy Mobile IPv6", RFC 6543, May 2012.

[RFC6090] McGrew, D., Igoe, K., Salter, M., "Fundamental Elliptic Curve Cryptography Algorithms", RFC 6090, February 2012.

8.2. Informative References

[1] IEEE Standards Association, http://standards.ieee.org/develop/regauth/tut/eui64.pdf, 2012

[2] Rafiee, H., Meinel, C., "'SSAS: a Simple Secure Addressing Scheme for IPv6 AutoConfiguration". In Proceedings of the 11th IEEE International Conference on Privacy, Security and Trust (PST), IEEE Catalog number: CFP1304F-ART, ISBN: 978-1-4673-5839-2.

Authors' Addresses

Hosnieh Rafiee
Hasso-Plattner-Institute
Prof.-Dr.-Helmert-Str. 2-3
Potsdam, Germany
Phone: +49 (0)331-5509-546
Email: ietf@rozanak.com

Dr. Christoph Meinel
(Professor)
Hasso-Plattner-Institute
Prof.-Dr.-Helmert-Str. 2-3
Potsdam, Germany
Email: meinel@hpi.uni-potsdam.de