Network Group H. Rafiee
INTERNET-DRAFT Huawei Technologies Duesseldorf GmbH
Intended status: Experimental C. Meinel
Hasso Plattner Institute
Expires: March 19, 2015 September 19, 2014

A Simple Secure Addressing Scheme for IPv6 AutoConfiguration (SSAS)

<draft-rafiee-6man-ssas-11.txt>

Abstract

Since performance and security are, both, two important criteria for a mechanism to be widely used by different nodes with various resources, the purpose of this document is to propose a mechanism for local security and to prevent IP spoofing. This mechanism also consider user's privacy.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

In IPv6 networks, nodes can use two different mechanisms to configure their IP addresses -- Neighbor Discovery Protocol (NDP) [RFC4861, RFC4862] and Dynamic Host Configuration Protocol (DHCPv6) [RFC3315]. Unfortunately none of these mechanisms are natively secure. So, they open the nodes with so many local security problems. There are several attacks possible in local network [RFC3756]. One example is IP spoofing that enable an attacker to forge the identity of a victim node, the other example is preventing the node from configuring its IP address.

The reasons that local security is important are as follows [localSecurity]:

- Not all the nodes on the local link are trusted: viruses or other malware can infect a legitimate node in the local link and turn it to an attacker.

- Attacker might be inside the network: The networks of big enterprises might be harmed by one of the staff that was recently fired.

There is currently a mechanism available to secure the NDP, i.e., Secure Neighbor Discovery (SeND) [RFC3971]. SeND does this protection by adding 4 options to NDP packets. Among these options, Cryptographically Generated Addresses (CGA) [RFC3972] is a very important option that provides the node with the proof of IP address ownership by finding a binding between the node's public key and its IP address. Unfortunately CGA has some problems that are listed as follows:

- CGA sec value problem: This problem is explained in [cgaattack] and addressed in [cgabis].

- CGA increases complexity and decreases performance: CGA uses sec value (the value between 0 to 7) and claims to complicate the brute force attacks. (However it is not true based on [cgaattack]) If CGA sec value higher than 0 is in use, then this will reduce the performance because CGA algorithm needs to repeat some steps and it needs the high attention of the CPU and makes the CPU busy. So, CGA sec value higher than 0, consumes more energy than other nodes that do not use CGA. Today, the demands on malti-functioning smaller devices are increasing but unfortunately the battery technology is not as advanced as expected. So, the use of CGA algorithm that needs to use higher level of energy is not ideal for these types of nodes and the use of CGA sec value zero does not protect the node as expected. (Please refer to appendix A for more information)

- CGA might cause privacy issue: Since the generation of CGA higher sec values might take time. The nodes might not be willing to change its IP address and keep this address as long as the subnet prefix is

valid. If the node is a fixed node in the network, then it will be vulnerable to node tracking. The node might also not change the CGA address when it visits a new network or it might not generate any new key pairs. In other word, it might use the same CGA parameters (excluding prefix) as used in the old network and thus it will be vulnerable to node tracking.

- Packet size

CGA uses RSA as a default key pair generation algorithm. This is why, if SeND with CGA option is in use to secure NDP messages, the minimum packet size needs to carry this public key for CGA nodes is 460 bytes. Packet size also reduces the performance and causes delays in the network.

Since privacy and security are, both, very important issues in everyday life, the purpose of this document is to offer an alternative and simple addressing mechanism to generate an interface ID (IID) which provides the node with both security and privacy while does not sacrifice the performance, and tries to decrease the packet size as much as possible.

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. Algorithms Overview

As explained earlier, one of the problems with using the current IID generation approach is the intensive computer processing that is needed for the IID algorithm generation. Another concern is for the lack of security (if CGA is not in use). This is what this document intends to address.

3.1. 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 one of the latest version of ECC algorithm [RFC6090] or other fastest short key size algorithms. . The implementations SHOULD be updated with any new

version of ECC algorithm when ECC current version is no longer secure. ECC is the default algorithm, but any algorithm capable of generating a small key size in a short amount of time is viable. The node 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.

Note 1: The node MUST not generate the weak key. For ECC, the node MUST not use ECC key size lower than 192 bits. If any nodes used a weak key size, then the other nodes MUST discard receiving the

message from that node. If in future, key size 192 bits is considered as a weak key size, the default key size value MUST be changed to the next strong key size.

2. Execute a hash function on the public key. The default hash function is SHA256. If in future, this hash function is no longer secure, the node MUST use the next strong hash function.

3. Take the first 64bits of the digest and call it IID. In case collision count is higher than 1, then depends on the number, takes second 64 bits or third 64 bits of this hash value.

It is not RECOMMENDED to use this algorithm in case IID is less than 64 bits [variableprefix]. A node MUST obtain the prefix length information form router advertisement messages.

4. Concatenate the IID with the local subnet prefix to set the link 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 3.1.2.)

Note 1: In this document bits u and g does not have any particular meaning and is used as a part of public key. This assumption is by the clarification of using these bits in [RFC7136].

3.1.1. Signature Generation

SSAS is not dependent to SeND but it can be used as a new option of SeND. When SSAS is used as an option of SeND, SSAS signature can be placed as a RSA signature in SeND. If SSAS is used alone, this section MUST be included in SSAS data structure. This proves that SSAS is compatible to use with SeND.

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. 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 1)

```
+---------+-----------+--------------+--------------+
|timestamp|Mac address|Collision Count|Algorithm type|
| 8 bytes |  6 bytes  |     3 bits    |    1 byte    |
+---------------------+--------------+--------------+
|Global IP address    | Other Options |
|    16 bytes         |    variable   |
+---------------------+--------------+
```

Figure 1 SSAS Signature format

2. Sign the resulting value from step 1, using the ECC private key (or any other short key size algorithm), 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).

3.1.2. Generation of NDP/SeND 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.

3.1.2.1. SSAS signature data field

```
+------+-------+------------+---------+
| Type |Length |  Reserved  |Other len|
|1 byte|1 byte |  2 bytes   | 1 byte  |
+----------+---------+------+---------+
| Algorithm|Collision|Subnet| Other   |
|   type   |  count  |prefix|Options  |
|  1 byte  |  3 bits |8bytes|         |
+-------------------------------------+
|Hash      | Response|                |
|Function  | No.     | SSAS Signature |
```

```
| 1 byte   | 2 byte  |                 |
+--------------------------------------+
|             padding                  |
+--------------------------------------+
```

Figure 2 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 64 bytes from the start of the hash value (digest) array of the public key.

- Subnet Prefix: This is the router subnet prefix.

- Hash Function: A hash function used to generate IID. The length of this field is 1 byte. For SHA256, this value is 0. Future algorithms will start at one and increase from there.

- Response No: This is similar to nonce but by the use of different mechanism. This value is not random and it is a copy of timestamp. In sender?s message, this value MUST be set to zero and in response message (sent from a receiver node), this value MUST be set to the timestamp of the sender?s message. The length of this field is 2 bytes. The sender node should cache this value in order to compare it with all responses sent by other nodes. This informs the sender node that the message is the response to his message and protects the node

against replay attack.

- 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. The node repeats this 3 times and after 3 times generates a new public/private keys. Since the likelihood of two nodes having the same value is 1/(2^63). This is really a small value while we also considered the order of magnitude relative to roughly 2 power 64 against sloppy implementations.

3.1.3. SSAS 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 Response No from the sender?s packet. Compare this value with its own timestamp that used in its previous message. If it is the same go to the next step, otherwise discard the message. (If SSAS is a part of SeND, this step should be skipped.)

2. Obtain prefix information from its own cache or from a router advertisement to make sure about the prefix sizes and number of bits used for IID.

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

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

5. If (t2- x) < = t1 < = (t2 + x) go to step 6. Otherwise, the message SHOULD be discarded without further processing. The value of x is dependent on network delays and network policy. The default value would be the value of Round Trip Time (RTT). The implementations SHOULD allow to set different values.

6. Obtain the public key from its own neighboring cache. If no matches are found in the node cache and if there is a centralized RPKI model available in the local network, then the node MIGHT obtain this public key from that node. Otherwise go to the next step.

7. 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 RPKI to obtain the other nodes' public key.)

8. Obtain the hash algorithm from the packet. By default it is SHA256.

9. Execute hash function on the public key. Takes 64bits, depends on collision count, from the hash function. 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.

10. 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.

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

12. 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.

After a successful verification, the node SHOULD store the public key and MAC address of the sender node in its neighboring cache. By default, the cache is valid for two days but the implementation SHOULD consider a way to let the end users change this default value.

3.2. Resource Public key Infrastructure (RPKI)

To Authorize the Routers in the network and increase the security of the nodes in this network, it is recommended to use an RPKI explained in RFC 6494 and 6495. It is explained in more detail in [SSASAnalysis] and local security deployment [localSecurity].

4. SSAS Applications

4.1. A solution for all nodes

SSAS is capable to be used in standard nodes (standard computers) and nodes where limited computational resources are available. One example is the use of SSAS in sensor networks. 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.

4.2. Authentication in Network layer

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.

4.3. Authentication in Application Layer

SSAS can be used as a means of authentication for the nodes in application layer. It is really important that the nodes know who they are talking to. This is because a user uses an application to connect to another node on the internet. This application either uses a domain name of the destination node (that later translates to the IP addresses) or directly uses the IP address of this node. This is where the attacker can play a role and spoof this IP address and play a MITM attack or other types of attacks. If the node uses this approach, the attacker does not have a possibility to spoof the IP address of the communicating node. So, this approach can mitigate IP spoofing during the authentication of two nodes in application layer.

4.4. 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 be for use in vehicular networks or in the car-to-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.

5. Security Considerations

There are two security considerations:

Since SSAS cannot prevent the layer 2 attacks but can mitigate it after the first verification, therefore one would need to use a monitoring device to prevent MAC spoofing. The other possibility is to have a dynamic MAC address. This means the SSAS node can use the 48 rightmost bits of the its public key as a MAC address. In this case there is a binding between the IP address, MAC address and public key. Since the verification process would have failed, it cannot be spoofed. However, this approach might be problematic from an operational view and might need to have some consideration before

being used.

Another security consideration is how to attack SSAS. One might ask oneself that 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 64 bits of that in Interface ID? 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 likelihood of matching 64 bits in the public key against 64bits in the IID is 1/(2^64). in [SSASAnalysis] the analysis of SSAS is explained and compared to CGA. Since the nodes in the network need to keep the public key and the MAC address of other nodes in the cache, the attacker only has a few seconds to perform this attack and then the attacker needs to perform this attack against the whole public key. For CGA, this value is less. in [cgaattack], the attack in CGA was explained. So, in general, SSAS is faster and in a good security level. In other word, SSAS tried to address the security and performance problem exists in CGA and offer a fastest algorithm.

6. IANA Considerations

There is no IANA consideration

7. Privacy Consideration

When an attacker is inside a local link, he is enable to identify a node. although, this target node changes its IP address. The reason is because the target node does not change its MAC address. However, if the public key needs to be used for verification in other mechanisms and not in local link, then it is RECOMMENDED that the public/private keys to be valid for a short period of time. The default value would be a week. The implementations need to consider the automatic key generation to avoid administrative requirements for this process.

8. Appendix A

8.1. Comparison of CGA and SSAS generation time

The following information was retrieved from [cgatime]. It shows the time required to generate CGA in different sec value. This is why, in practice, only sec value 0 and 1 can be used.

sec value 1 => ~ 1 second

sec value 2 => ~ 3 hours

sec value 3 => ~ 24 years

sec value 4 => ~ 1.16*10^6 years

sec value 5 => ~ 1*10^11 years

sec value 6 => ~ 6.8*10^15 years

The above information is based on the fact that one uses RSA key sizes less than 1280 bits. If one needs to use the higher security, then it needs more time for the generation of CGA value. Using RSA higher key sizes also increases the packet size needs to carry the public key. Here is our evaluation of ECC and RSA key generation time in a standard computer with 2.6 GHz CPU.

SSAS generation time is about the time needed to generate key pairs. Since, by default, SSAS uses ECC 192 bits, the following values compares ECC with RSA. RSA is the algorithm uses in CGA. As explained earlier, the security of ECC 192 bits is equivalent to the security of RSA 7680 bits.

ECC 192 bits: Average key generation time = 195011 microseconds

RSA 1280 bits: Average key generation time = 681039 microseconds

RSA 7680 bits: Average key generation time = 163473350 microseconds

9. Appendix B

9.1. Network-based protection vs. Node-based protection

Node-based protection is the ability of the node to protect against some types of attacks such as IP spoofing, MITM attack. On the other hand, network-based protection is the use of some devices in the network edges to protect the nodes inside this network against router advertisement spoofing attacks or other types of attack. Both of these protection is required and both can complement each other. This is because the attacker might be inside the network and play a role of MITM, spoof the other nodes' IP address, prevent other nodes from configuring their IP address and cause many delays and problems in

the local network (Not all the nodes in the network is ever trustee). One important consideration about node-based protection is that, it should support any node and apply to any nodes (Including nodes with limited energy resources or limited memory resources). This is why there is a need for a good mechanism to provide this protection with less cost. The proposed mechanism in this document, i.e., SSAS can provide the node with node-based protection. With only node-based protection, the malicious node inside this network can claim to be a router and the node does not have any means to authorize him. This is

why, the network-based protection is also the complement solution to a node-based protection. There are some approaches to provide the node with network-based protection. One such approach might be RA-gaurd [RAgaurd]] which limits subnet prefixes. Unfortunately with this approach, still the node inside this network can maliciously claim to be a router and play the MITM attack inside the network by sending unicast router advertisement messages. So, the attack is still possible. The other approach is the use of RPKI as explained in RFC 6494 and RFC 6495. Unfortunately these RFCs only explain the possibility of using them but not the detail of implementation. The detail implementation is explained in [SSASAnalysis]. The local RPKI node also can play a role of monitoring device in the network.

10. Acknowledgements

The Authors would like to acknowledge Erik Nordmard and Joel M. Halpern for their supports and assistance to improve this document.The authors also would like to acknowledge Michael Richardson, Dan Wing, Tim Chown, Christian Huitema, Joel M. Halpern for their comments to improve this document

11. References

11.1. 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.

[RFC3756] Nikander, P., Kempf, J., Nordmark, E., "IPv6 Neighbor Discovery (ND) Trust Models and Threats", RFC 3972, May 2004.

[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.

[RFC3756] Nikander, F., Kempf, J., Nordmark, E., "IPv6 Neighbor Discovery (ND) Trust Models and Threats", RFC 3756, May 2004.

[RFC7136] Carpenter, B., Jiang, S.,"Significance of IPv6 Interface Identifiers", RFC 7136, 2013

11.2. Informative References

[SSASAnalysis] 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.

[cgaattack] Rafiee,H., Meinel, C., "Possible Attack on Cryptographically Generated Addresses (CGA)" ,http://tools.ietf.org/html/draft-rafiee-6man-cga-attack, 2014

[RAgaurd] Gont, F., "Implementation Advice for IPv6 Router Advertisement Guard (RA-Guard)", http://tools.ietf.org/html/draft-ietf-v6ops-ra-guard-implementatio 2012

[localSecurity] Rafiee,H., Meinel, C., "Recommendations for Local Security Deployments" ,http://tools.ietf.org/html/draft-rafiee-6man-local-security 2013

[cgatime] Bos, J., Oezen, O., Hubaux, J., "Analysis and Optimization of Cryptographically Generated Addresses", In Proceedings of the 12th International conference on Information Security (2009), ACM, pp. 17 ? 32.

[variableprefix] Carpenter, B., Chown, T, Gont, F., Jiang, S., Petrescu, A., Yourtchenko, A.," Analysis of the 64-bit Boundary in IPv6 Addressing", http://tools.ietf.org/html/draft-ietf-6man-why64 , April 2014

[cgabis] Rafiee,H., Zhang, D., "CGA Security Improvement" ,http://tools.ietf.org/html/draft-rafiee-rfc3972-bis, 2014

Authors' Addresses

Hosnieh Rafiee
HUAWEI TECHNOLOGIES Duesseldorf GmbH

Riesstrasse 25, 80992,
Munich, Germany
Phone: +49 (0)162 204 74 58
Email: hosnieh.rafiee@huawei.com

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