IPv6 maintenance Working Group (6man) F. Gont
Internet-Draft SI6 Networks / UTN-FRH
Intended status: Standards Track January 27, 2013
Expires: July 31, 2013

A method for Generating Stable Privacy-Enhanced Addresses with IPv6 Stateless Address Autoconfiguration (SLAAC)
draft-ietf-6man-stable-privacy-addresses-03

Abstract

This document specifies a method for generating IPv6 Interface Identifiers to be used with IPv6 Stateless Address Autoconfiguration (SLAAC), such that addresses configured using this method are stable within each subnet, but the Interface Identifier changes when hosts move from one network to another. The aforementioned method is meant to be an alternative to generating Interface Identifiers based on IEEE identifiers, such that the benefits of stable addresses can be achieved without sacrificing the privacy of users.

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/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 31, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

[RFC4862] specifies the Stateless Address Autoconfiguration (SLAAC) for IPv6 [RFC2460], which typically results in hosts configuring one or more "stable" addresses composed of a network prefix advertised by a local router, and an Interface Identifier (IID) that typically embeds a hardware address (e.g., using IEEE identifiers) [RFC4291].

Generally, stable addresses are thought to simplify network management, since they simplify Access Control Lists (ACLs) and logging. However, since IEEE identifiers are typically globally unique, the resulting IPv6 addresses can be leveraged to track and correlate the activity of a node over time and across multiple subnets and networks, thus negatively affecting the privacy of users.

The "Privacy Extensions for Stateless Address Autoconfiguration in IPv6" [RFC4941] were introduced to complicate the task of eavesdroppers and other information collectors to correlate the activities of a node, and basically result in temporary (and random) Interface Identifiers that are typically more difficult to leverage than those based on IEEE identifiers. When privacy extensions are enabled, "privacy addresses" are employed for "outgoing communications", while the traditional IPv6 addresses based on IEEE identifiers are still used for "server" functions (i.e., receiving incoming connections).

> As noted in [RFC4941], "anytime a fixed identifier is used in multiple contexts, it becomes possible to correlate seemingly unrelated activity using this identifier". Therefore, since "privacy addresses" [RFC4941] do not eliminate the use of fixed identifiers for server-like functions, they only *partially* mitigate the correlation of host activities (see Appendix A for some example attacks that are still possible with privacy addresses). Therefore, it is vital that the privacy characteristics of "stable" addresses are improved such that the ability of an attacker correlating host activities across networks

is reduced.

Another important aspect not mitigated by "Privacy Addresses" [RFC4941] is that of host scanning. Since IPv6 addresses that embed IEEE identifiers have specific patterns, an attacker could leverage such patterns to greatly reduce the search space for "live" hosts. Since "privacy addresses" do not eliminate the use of IPv6 addresses that embed IEEE identifiers, host scanning attacks are still feasible even if "privacy extensions" are employed [Gont-DEEPSEC2011] [CPNI-IPv6]. This is yet another motivation to improve the privacy characteristics of "stable" addresses (currently embedding IEEE identifiers).

Privacy/temporary addresses can be challenging in a number of areas. For example, from a network-management point of view, they tend to increase the complexity of event logging, trouble-shooting, and enforcing access controls and quality of service, etc. As a result, some organizations disable the use of privacy addresses even at the expense of reduced privacy [Broersma]. Also, they result in increased complexity, which might not be possible or desirable in some implementations (e.g., some embedded devices).

In scenarios in which "Privacy Extensions" are deliberately not used (possibly for any of the aforementioned reasons), all a host is left with is the addresses that have been generated using e.g. IEEE identifiers, and this is yet another case in which it is also vital that the privacy characteristics of these stable addresses are improved.

We note that in most (if not all) of those scenarios in which "Privacy Extensions" are disabled, there is usually no actual desire to negatively affect user privacy, but rather a desire to simplify operation of the network (simplify the use of ACLs, logging, etc.).

This document specifies a method to generate interface identifiers that are stable/constant within each subnet, but that change as hosts move from one network to another, thus keeping the "stability" properties of the interface identifiers specified in [RFC4291], while still mitigating host-scanning attacks and preventing correlation of the activities of a node as it moves from one network to another.

This document does not update or modify IPv6 StateLess Address Auto-

Configuration (SLAAC) [RFC4862] itself, but rather only specifies an alternative algorithm to generate Interface IDs. Therefore, the usual address lifetime properties (as specified in the corresponding Prefix Information Options) apply when IPv6 addresses are generated as a result of employing the algorithm specified in this document with SLAAC [RFC4862]. Additionally, from the point of view of renumbering, we note that these addresses behave like the traditional IPv6 addresses (that embed a hardware address) resulting from SLAAC [RFC4862].

For nodes that currently disable "Privacy extensions" [RFC4941] for some of the reasons stated above, this mechanism provides stable privacy-enhanced addresses which may already address most of the privacy concerns related to addresses that embed IEEE identifiers [RFC4291]. On the other hand, in scenarios in which "Privacy Extensions" are employed, implementation of the mechanism described in this document would mitigate host-scanning attacks and also mitigate correlation of host activities.

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].

2. Design goals

This document specifies a method for selecting interface identifiers to be used with IPv6 SLAAC, with the following goals:

- The resulting interface identifiers remain constant/stable for each prefix used with SLAAC within each subnet. That is, the algorithm generates the same interface identifier when configuring an address belonging to the same prefix within the same subnet.

- The resulting interface identifiers do not depend on the underlying hardware (e.g. link-layer address). This means that e.g. replacing a Network Interface Card (NIC) will not have the (generally undesirable) effect of changing the IPv6 addresses used for that network interface.

o The resulting interface identifiers do change when addresses are configured for different prefixes. That is, if different autoconfiguration prefixes are used to configure addresses for the same network interface card, the resulting interface identifiers must be (statistically) different.

o It must be difficult for an outsider to predict the interface identifiers that will be generated by the algorithm, even with knowledge of the interface identifiers generated for configuring other addresses.

o The aforementioned interface identifiers are meant to be an alternative to those based on e.g. IEEE identifiers, such as those specified in [RFC2464].

We note that of use of the algorithm specified in this document is (to a large extent) orthogonal to the use of "Privacy Extensions" [RFC4941]. Hosts that do not implement/use "Privacy Extensions" would have the benefit that they would not be subject to the host-tracking and host scanning issues discussed in the previous section. On the other hand, in the case of hosts employing "Privacy Extensions", the method specified in this document would prevent host scanning attacks and correlation of node activities across networks (see Appendix A).

3. Algorithm specification

IPv6 implementations conforming to this specification MUST generate interface identifiers using the algorithm specified in this section in replacement of any other algorithms used for generating "stable" addresses (such as that specified in [RFC2464]). The aforementioned algorithm MUST be employed for generating the interface identifiers for all of the IPv6 addresses configured with SLAAC for a given interface, including IPv6 link-local addresses.

This means that this document does not formally obsolete or deprecate any of the existing algorithms to generate Interface IDs (e.g. such as that specified in [RFC2464]). However, those IPv6 implementations that employ this specification must generate all of their "stable" addresses as specified in this document.

Implementations conforming to this specification SHOULD provide the means for a system administrator to enable or disable the use of this algorithm for generating Interface Identifiers. Implementations conforming to this specification MAY employ the algorithm specified in [RFC4941] to generate temporary addresses in addition to the addresses generated with the algorithm specified in this document.

Unless otherwise noted, all of the parameters included in the expression below MUST be included when generating an Interface ID.

1. Compute a random (but stable) identifier with the expression:

   RID = F(Prefix, Interface_Index, Network_ID, DAD_Counter, secret_key)

   Where:

   RID:
      Random (but stable) identifier

   F():
      A pseudorandom function (PRF) that is not computable from the outside (without knowledge of the secret key). The PRF could be implemented as a cryptographic hash of the concatenation of each of the function parameters.

   Prefix:
      The prefix to be used for SLAAC, as learned from an ICMPv6 Router Advertisement message.

   Interface_Index:

The interface index [RFC3493] [RFC3542] corresponding to this network interface.

Network_ID:
Some network specific data that identifies the subnet to which this interface is attached. For example the IEEE 802.11 Service Set Identifier (SSID) corresponding to the network to which this interface is associated. This parameter is OPTIONAL.

DAD_Counter:
A counter that is employed to resolve Duplicate Address Detection (DAD) conflicts. It MUST be initialized to 0, and incremented by 1 for each new tentative address that is configured as a result of a DAD conflict. Implementations that record DAD_Counter in non-volatile memory for each {Prefix, Interface_Index, Network_ID} tuple MUST initialize DAD_Counter to the recorded value if such an entry exists in non-volatile memory). See Section 4 for additional details.

secret_key:
A secret key that is not known by the attacker. The secret key MUST be initialized at system installation time to a pseudo-random number (see [RFC4086] for randomness requirements for security). An implementation MAY provide the means for the user to change the secret key.

2. The Interface Identifier is finally obtained by taking the leftmost 64 bits of the RID value computed in the previous step, and setting bit 6 (the leftmost bit is numbered 0) to zero. This creates an interface identifier with the universal/local bit indicating local significance only. The resulting Interface Identifier should be compared against the list of reserved interface identifiers [IANA-RESERVED-IID], and to those interface identifiers already employed in an address of the same network interface and the same network prefix. In the event that an unacceptable identifier has been generated, this situation should be handled in the same way as the case of duplicate addresses (see Section 4).

This document does not require the use of any specific PRF for the function F() above, since the choice of such PRF is usually a trade-off between a number of properties (processing requirements, ease of implementation, possible intellectual property rights, etc.), and since the best possible choice for F() might be different for different types of devices (e.g. embedded systems vs. regular servers) and might possibly change over time.

Note that the result of F() in the algorithm above is no more secure than the secret key. If an attacker is aware of the PRF that is being used by the victim (which we should expect), and the attacker can obtain enough material (i.e. addresses configured by the victim), the attacker may simply search the entire secret-key space to find matches. To protect against this, the secret key should be of a reasonable length. Key lengths of at least 128 bits should be adequate. The secret key is initialized at system installation time to a pseudo-random number, thus allowing this mechanism to be enabled/used automatically, without user intervention.

Including the SLAAC prefix in the PRF computation causes the Interface ID to vary across networks that employ different prefixes, thus mitigating host-tracking attacks and any other attacks that benefit from predictable Interface IDs (such as host scanning).

Including the optional Network_ID parameter when computing the RID value above would cause the algorithm to produce a different Interface Identifier when connecting to different networks, even when configuring addresses belonging to the same prefix. This means that a host would employ a different Interface ID as it moves from one network to another even for IPv6 link-local addresses or Unique Local Addresses (ULAs).

4. Resolving Duplicate Address Detection (DAD) conflicts

If as a result of performing Duplicate Address Detection (DAD) [RFC4862] a host finds that the tentative address generated with the algorithm specified in Section 3 is a duplicate address, it SHOULD resolve the address conflict by trying a new tentative address as follows:

- o DAD_Counter is incremented by 1.
- o A new Interface ID is generated with the algorithm specified in Section 3, using the incremented DAD_Counter value.

This procedure may be repeated a number of times until the address conflict is resolved. We RECOMMEND hosts to try at least IDGEN_RETRIES (hereby specified as "3") tentative addresses if DAD fails for successive generated addresses, in the hopes of resolving the address conflict. We also note that hosts MUST limit the number of tentative addresses that are tried (rather than indefinitely try a new tentative address until the conflict is resolved).

In those (unlikely) scenarios in which duplicate addresses are detected and in which the order in which the conflicting nodes configure their addresses may vary (e.g., because they may be bootstrapped in different order), the algorithm specified in this section for resolving DAD conflicts could lead to addresses that are not stable within the same subnet. In order to mitigate this potential problem, nodes MAY record the DAD_Counter value employed for a specific {Prefix, Interface_Index, Network_ID} tuple in non-volatile memory, such that the same DAD_Counter value is employed when configuring an address for the same Prefix and subnet at any other point in time.

In the event that a DAD conflict cannot be solved (possibly after trying a number of different addresses), address configuration would fail. In those scenarios, nodes MUST NOT automatically fall back to employing other algorithms for generating interface identifiers.

5. IANA Considerations

There are no IANA registries within this document. The RFC-Editor can remove this section before publication of this document as an RFC.

6. Security Considerations

This document specifies an algorithm for generating interface identifiers to be used with IPv6 Stateless Address Autoconfiguration (SLAAC), as an alternative to e.g. interface identifiers that embed IEEE identifiers (such as those specified in [RFC2464]). When compared to such identifiers, the identifiers specified in this document have a number of advantages:

- They prevent trivial host-tracking, since when a host moves from one network to another the network prefix used for autoconfiguration and/or the Network ID (e.g., IEEE 802.11 SSID) will typically change, and hence the resulting interface identifier will also change (see Appendix A.

- They mitigate address-scanning techniques which leverage predictable interface identifiers (e.g., known Organizational Unique Identifiers) [I-D.ietf-opsec-ipv6-host-scanning].

- They result in IPv6 addresses that are independent of the underlying hardware (i.e. the resulting IPv6 addresses do not change if a network interface card is replaced).

We note that this algorithm is meant to be an alternative to interface identifiers such as those specified in [RFC2464], but is not meant as an alternative to temporary Interface IDs (such as those specified in [RFC4941]). Clearly, temporary addresses may help to mitigate the correlation of activities of a node within the same network, and may also reduce the attack exposure window (since privacy/temporary addresses are short-lived when compared to the addresses generated with the method specified in this document). We note that implementation of this algorithm would still benefit those hosts employing "Privacy Addresses", since it would mitigate host-tracking vectors still present when privacy addresses are used (see Appendix A), and would also mitigate host-scanning techniques that leverage patterns in IPv6 addresses that embed IEEE identifiers.

Finally, we note that the method described in this document may mitigate most of the privacy concerns arising from the use of IPv6 addresses that embed IEEE identifiers, without the use of temporary addresses, thus possibly offering an interesting trade-off for those scenarios in which the use of temporary addresses is not feasible.

7. Acknowledgements

The algorithm specified in this document has been inspired by Steven Bellovin's work ([RFC1948]) in the area of TCP sequence numbers.

The author would like to thank (in alphabetical order) Karl Auer, Steven Bellovin, Matthias Bethke, Brian Carpenter, Tassos Chatzithomaoglou, Dominik Elsbroek, Bob Hinden, Christian Huitema, Ray Hunter, Jong-Hyouk Lee, Michael Richardson, and Ole Troan, for providing valuable comments on earlier versions of this document.

This document is based on the technical report "Security Assessment of the Internet Protocol version 6 (IPv6)" [CPNI-IPv6] authored by Fernando Gont on behalf of the UK Centre for the Protection of National Infrastructure (CPNI).

Fernando Gont would like to thank CPNI (http://www.cpni.gov.uk) for their continued support.

8. References

8.1. Normative References

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

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

[RFC4086] Eastlake, D., Schiller, J., and S. Crocker, "Randomness

Requirements for Security", BCP 106, RFC 4086, June 2005.

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

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.

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

8.2. Informative References

[RFC1948] Bellovin, S., "Defending Against Sequence Number Attacks", RFC 1948, May 1996.

[RFC2464] Crawford, M., "Transmission of IPv6 Packets over Ethernet Networks", RFC 2464, December 1998.

[RFC3493] Gilligan, R., Thomson, S., Bound, J., McCann, J., and W. Stevens, "Basic Socket Interface Extensions for IPv6", RFC 3493, February 2003.

[RFC3542] Stevens, W., Thomas, M., Nordmark, E., and T. Jinmei, "Advanced Sockets Application Program Interface (API) for IPv6", RFC 3542, May 2003.

[I-D.ietf-opsec-ipv6-host-scanning] Gont, F. and T. Chown, "Network Reconnaissance in IPv6 Networks", draft-ietf-opsec-ipv6-host-scanning-00 (work in progress), December 2012.

[IANA-RESERVED-IID] Reserved IPv6 Interface Identifiers, "http://www.iana.org/assignments/ipv6-interface-ids/ipv6-interface-ids.xml".

[Gont-DEEPSEC2011] Gont, "Results of a Security Assessment of the Internet Protocol version 6 (IPv6)", DEEPSEC 2011 Conference, Vienna, Austria, November 2011, <http://

www.si6networks.com/presentations/deepsec2011/
fgont-deepsec2011-ipv6-security.pdf>.

[Gont-BRUCON2012]
Gont, "Recent Advances in IPv6 Security", BRUCON 2012 Conference, Ghent, Belgium, September 2012, <http://www.si6networks.com/presentations/brucon2012/fgont-brucon2012-recent-advances-in-ipv6-security.pdf>.

[Broersma]
Broersma, R., "IPv6 Everywhere: Living with a Fully IPv6-enabled environment", Australian IPv6 Summit 2010, Melbourne, VIC Australia, October 2010, <http://www.ipv6.org.au/summit/talks/Ron_Broersma.pdf>.

[CPNI-IPv6]
Gont, F., "Security Assessment of the Internet Protocol version 6 (IPv6)", UK Centre for the Protection of National Infrastructure, (available on request).

Appendix A. Privacy issues still present with RFC 4941

This section aims to clarify the motivation of using the method specified in this document even when privacy/temporary addresses [RFC4941] are employed. It discusses a (non-exaustive) number of scenarios in which host privacy is still sacrificed even when privacy/temporary addresses [RFC4941] are employed, as a result of employing interface identifiers that are constant across networks (e.g., those resulting from embedding IEEE identifiers).

A.1. Host tracking

This section describes one possible attack scenario that illustrates that host-tracking may still be possible when privacy/temporary addresses [RFC4941] are employed.

A.1.1. Tracking hosts across networks #1

A host configures its stable addresses with the constant Interface-ID, and runs any application that needs to perform a server-like function (e.g. a peer-to-peer application). As a result of that, an attacker/user participating in the same application (e.g., P2P) would learn the constant Interface-ID used by the host for that network interface.

Some time later, the same host moves to a completely different network, and employs the same P2P application, probably even with a different username. The attacker now interacts with the same host again, and hence can learn its newly-configured stable address. Since the interface ID is the same as the one used before, the attacker can infer that it is communicating with the same device as before.

This is just *one* possible attack scenario, which should remind us that one should not disclose more than it is really needed for achieving a specific goal (and an Interface-ID that is constant across different networks does exactly that: it discloses more information than is needed for providing a stable address).

A.1.2. Tracking hosts across networks #2

Once an attacker learns the constant Interface-ID employed by the victim host for its stable address, the attacker is able to "probe" a network for the presence of such host at any given network.

See Appendix A.1.1 for just one example of how an attacker could learn such value. Other examples include being able to share the same network segment at some point in time (e.g. a conference network or any public network), etc.

For example, if an attacker learns that in one network the victim used the Interface-ID 1111:2222:3333:4444 for its stable addresses, then he could subsequently probe for the presence of such device in the network 2011:db8::/64 by sending a probe packet (ICMPv6 Echo Request, or any other probe packet) to the address 2001:db8::1111:2222:3333:4444.

A.1.3. Revealing the identity of devices performing server-like functions

Some devices, such as storage devices or printers, may typically perform server-like functions and may be usually moved from one network to another. Such devices are likely to simply disable (or not even implement) privacy/temporary addresses [RFC4941]. If the aforementioned devices employ Interface-IDs that are constant across networks, it would be trivial for an attacker to tell whether the same device is being used across networks by simply looking at the Interface ID. Clearly, performing server-like functions should not imply that a device discloses its identity (i.e., that the attacker can tell whether it is the same device providing some function in two different networks, at two different points in time).

The scheme proposed in this document prevents such information leakage by causing nodes to generate different Interface-IDs when moving to one network to another, thus mitigating this kind of privacy attack.

A.2. Address scanning attacks

While it is usually assumed that address-scanning attacks are unfeasible, an attacker could leverage patterns in IPv6 addresses to greatly reduce the search space [I-D.ietf-opsec-ipv6-host-scanning] [Gont-BRUCON2012].

As noted earlier in this document, privacy/temporary addresses do not eliminate the use of IPv6 addresses that embed IEEE identifiers, and hence such hosts would still be vulnerable to address-scanning attacks. The method specified in this document eliminates such patterns and would thus mitigate the aforementioned address-scanning attacks.

Author's Address

Fernando Gont
SI6 Networks / UTN-FRH
Evaristo Carriego 2644
Haedo, Provincia de Buenos Aires 1706
Argentina

Phone: +54 11 4650 8472
Email: fgont@si6networks.com
URI: http://www.si6networks.com