opsec
Internet-Draft
Obsoletes: 5157 (if approved)
Intended status: Informational
Expires: November 1, 2015

F. Gont
Huawei Technologies
T. Chown
University of Southampton

April 30, 2015

Network Reconnaissance in IPv6 Networks
draft-ietf-opsec-ipv6-host-scanning-07

Abstract

IPv6 offers a much larger address space than that of its IPv4 counterpart. An IPv6 subnet of size /64 can (in theory) accommodate approximately 1.844 * 10^19 hosts, thus resulting in a much lower host density (#hosts/#addresses) than is typical in IPv4 networks, where a site typically has 65,000 or less unique addresses. As a result, it is widely assumed that it would take a tremendous effort to perform address scanning attacks against IPv6 networks, and therefore brute-force IPv6 address scanning attacks have been considered unfeasible. This document updates RFC 5157, which first discussed this assumption, by providing further analysis on how traditional address scanning techniques apply to IPv6 networks, and exploring some additional techniques that can be employed for IPv6 network reconnaissance. In doing so, this document formally obsoletes RFC 5157.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The main driver for IPv6 [RFC2460] deployment is its larger address space [CPNI-IPv6]. This larger address space not only allows for an increased number of connected devices, but also introduces a number of subtle changes in several aspects of the resulting networks. One of these changes is the reduced host density (the number of hosts divided by the number of addresses) of typical IPv6 subnetworks: with default IPv6 subnets of /64, each subnet comprises more than 1.844 * 10^19 available addresses; however, the actual number of nodes in each subnet is likely to remain similar to that of IPv4 subnetworks (typically a few hundred nodes per subnet). [RFC5157] describes how this significantly lower IPv6 host-density is likely to make classic network address scans less feasible, since even by applying various heuristics, the address space to be scanned remains very large. RFC 5157 goes on to describe some alternative methods for attackers to glean active IPv6 addresses, and provides some guidance for administrators and implementors, e.g. not using sequential addresses with DHCPv6.

With the benefit of five years of additional IPv6 deployment experience, this document formally updates and obsoletes RFC 5157. It emphasises that while scanning attacks are less feasible, they may, with appropriate heuristics, remain possible. At the time that RFC 5157 was written, observed scans were typically across ports on the addresses of discovered servers; since then, evidence that some classic address scanning is occurring is being witnessed. This text thus updates the analysis on the feasibility of "traditional"

address-scanning attacks in IPv6 networks, and it explores a number of additional techniques that can be employed for IPv6 network reconnaissance. Practical examples and guidance are also included in the Appendices.

On one hand, raising awareness about IPv6 network reconnaissance techniques may allow (in some cases) network and security administrators to prevent or detect such attempts. On the other hand, network reconnaissance is essential for the so-called "penetration tests" typically performed to assess the security of production networks. As a result, we believe the benefits of a thorough discussion of IPv6 network reconnaissance are two-fold.

Section 3 analyzes the feasibility of traditional address-scanning attacks (e.g. ping sweeps) in IPv6 networks, and explores a number of possible improvements to such techniques. [van-Dijk] describes a technique for leveraging DNS reverse mappings for discovering IPv6 nodes. Finally, Appendix A describes how the analysis carried out throughout this document can be leveraged to produce address-scanning tools (e.g. for penetration testing purposes).

2. Requirements for the Applicability of Network Reconnaissance Techniques

Throughout this document, a number of network reconnaissance techniques are discussed. Each of these techniques have different requirements on the side of the practitioner, with respect to whether they require local access to the target network, and whether they require login access (or similar access credentials) to the system on which the technique is applied.

The following table tries to summarize the aforementioned requirements, and serves as a cross index to the corresponding sections.

Technique	Local access	Login access
Local address scans (Section 3.3)	Yes	No
Remote Address scans (Section 3.2)	No	No
DNS Advertised Hosts (Section 4.1)	No	No
DNS Zone Transfers (Section 4.2)	No	No
DNS reverse mappings (Section 4.4)	No	No
Public archives (Section 6)	No	No
Application Participation (Section 7)	No	No
Inspection of the IPv6 Neighbor Cache and Routing Table (Section 8)	No	Yes
Inspecting System Configuration and Log Files (Section 9)	No	Yes
Gleaning information from Routing Protocols (Section 10)	Yes	No
Gleaning Information from IP Flow Information Export (IPFIX) (Section 11)	No	Yes
Obtaining Network Information with traceroute6 (Section 12)	No	No
Gleaning Information from Network Devices Using SNMP	No	Yes
Obtaining Network Information via Traffic Snooping	Yes	No

Table 1: Requirements for the Applicability of Network Reconnaissance Techniques

3. IPv6 Address Scanning

This section discusses how traditional address scanning techniques (e.g. "ping sweeps") apply to IPv6 networks. Section 3.1 provides an essential analysis of how address configuration is performed in IPv6,

identifying patterns in IPv6 addresses that can be leveraged to reduce the IPv6 address search space when performing IPv6 address scans. Appendix A discusses how the insights obtained in the previous sub-sections can be incorporated into into a fully-fledged IPv6 address scanning tool. Section 3.5 provides advice on how to mitigate IPv6 address scans.

3.1. Address Configuration in IPv6

IPv6 incorporates two automatic address-configuration mechanisms: SLAAC (StateLess Address Auto-Configuration) [RFC4862] and DHCPv6 (Dynamic Host Configuration Protocol version 6) [RFC3315]. SLAAC is the mandatory mechanism for automatic address configuration, while DHCPv6 is optional - however, most current versions of general-purpose operating systems support both. In addition to automatic address configuration, hosts, typically servers, may employ manual configuration, in which all the necessary information is manually entered by the host or network administrator into configuration files at the host.

The following subsections describe each of the possible configuration mechanisms/approaches in more detail.

3.1.1. StateLess Address Auto-Configuration (SLAAC)

The basic idea behind SLAAC is that every host joining a network will send a multicasted solicitation requesting network configuration information, and local routers will respond to the request providing the necessary information. SLAAC employs two different ICMPv6 message types: ICMPv6 Router Solicitation and ICMPv6 Router Advertisement messages. Router Solicitation messages are employed by hosts to query local routers for configuration information, while Router Advertisement messages are employed by local routers to convey the requested information.

Router Advertisement messages convey a plethora of network configuration information, including the IPv6 prefix that should be used for configuring IPv6 addresses on the local network. For each local prefix learned from a Router Advertisement message, an IPv6 address is configured by appending a locally-generated Interface Identifier (IID) to the corresponding IPv6 prefix.

The following subsections describe currently-deployed policies for generating the IIDs used with SLAAC.

3.1.1.1. Interface-Identifiers Embedding IEEE Identifiers

The traditional SLAAC interface identifiers are based on the link-layer address of the corresponding network interface card. For example, in the case of Ethernet addresses, the IIDs are constructed as follows:

1. The "Universal" bit (bit 6, from left to right) of the address is set to 1

2. The word 0xfffe is inserted between the OUI (Organizationally Unique Identifier) and the rest of the Ethernet address

For example, the MAC address 00:1b:38:83:88:3c would lead to the IID 021b:38ff:fe83:883c.

NOTE:
> [RFC7136] notes that all bits of an IID should be treated as "opaque" bits. Furthermore, [I-D.ietf-6man-default-iids] is currently in the process of changing the default IID generation scheme to [RFC7217]. Therefore, the traditional IIDs based on link-layer addresses are expected to become less common over time.

A number of considerations should be made about these identifiers. Firstly, as it should be obvious from the algorithm described above, two bytes (bytes 4-5) of the resulting address always have a fixed value (0xff, 0xfe), thus reducing the search space for the IID. Secondly, the first three bytes of these identifiers correspond to the OUI of the network interface card vendor. Since not all possible OUIs have been assigned, this further reduces the IID search space. Furthermore, of the assigned OUIs, many could be regarded as corresponding to legacy devices, and thus unlikely to be used for Internet-connected IPv6-enabled systems, yet further reducing the IID search space. Finally, in some scenarios it could be possible to infer the OUI in use by the target network devices, yet narrowing down the possible IIDs even more.

> For example, an organization known for being provisioned by vendor X is likely to have most of the nodes in its organizational network with OUIs corresponding to vendor X.

These considerations mean that in some scenarios, the original IID search space of 64 bits may be effectively reduced to 2^24 , or n * 2^24 (where "n" is the number of different OUIs assigned to the target vendor).

Further, if just one host address is detected or known within a subnet, it is not unlikely that, if systems were ordered in a batch,

that they may have sequential MAC addresses. Additionally, given a MAC address observed in one subnet, sequential or nearby MAC addresses may be seen in other subnets in the same site.

Another interesting factor arises from the use of virtualization technologies, since they generally employ automatically-generated MAC addresses, with very specific patterns. For example, all automatically-generated MAC addresses in VirtualBox virtual machines employ the OUI 08:00:27 [VBox2011]. This means that all SLAAC-produced addresses will have an IID of the form a00:27ff:feXX:XXXX, thus effectively reducing the IID search space from 64 bits to 24 bits.

VMWare ESX server provides yet a more interesting example. Automatically-generated MAC addresses have the following pattern [vmesx2011]:

1. The OUI is set to 00:05:59

2. The next 16-bits of the MAC address are set to the same value as the last 16 bits of the console operating system's primary IPv4 address

3. The final eight bits of the MAC address are set to a hash value based on the name of the virtual machine's configuration file.

This means that, assuming the console operating system's primary IPv4 address is known, the IID search space is reduced from 64 bits to 8 bits.

On the other hand, manually-configured MAC addresses in VMWare ESX server employ the OUI 00:50:56, with the low-order three bytes being in the range 0x000000-0x3fffff (to avoid conflicts with other VMware products). Therefore, even in the case of manually-configured MAC addresses, the IID search space is reduced from 64-bits to 22 bits.

3.1.1.2. Temporary Addresses

Privacy concerns [Gont-DEEPSEC2011] [I-D.ietf-6man-ipv6-address-generation-privacy] regarding interface identifiers embedding IEEE identifiers led to the introduction of "Privacy Extensions for Stateless Address Auto-configuration in IPv6" [RFC4941], also known as "temporary addresses" or "privacy addresses". Essentially, "temporary addresses" produce random addresses by concatenating a random identifier to the auto-configuration IPv6 prefix advertised in a Router Advertisement.

> In addition to their unpredictability, these addresses are typically short-lived, such that even if an attacker were to learn one of these addresses, they would be of use for a limited period of time. A typical implementation may keep a temporary address preferred for 24 hours, and configured but deprecated for seven days.

It is important to note that "temporary addresses" are generated in addition to traditional SLAAC addresses (i.e., based on IEEE identifiers): traditional SLAAC addresses are meant to be employed for "server-like" inbound communications, while "temporary addresses" are meant to be employed for "client-like" outbound communications. This means that implementation/use of "temporary addresses" does not prevent an attacker from leveraging the predictability of traditional SLAAC addresses, since "temporary addresses" are generated in addition to (rather than as a replacement of) the traditional SLAAC addresses derived from e.g. IEEE identifiers.

The benefit that temporary addresses offer in this context is that they reduce the exposure of the SLAAC address to any third parties that may observe traffic sent from a host where temporary addresses are enabled and used by default. But, in the absence of firewall protection for the host, its SLAAC address remains liable to be scanned from offsite.

3.1.1.3. Randomized Stable Interface Identifiers

In order to mitigate the security implications arising from the predictable IPv6 addresses derived from IEEE identifiers, Microsoft Windows produced an alternative scheme for generating "stable addresses" (in replacement of the ones embedding IEEE identifiers). The aforementioned scheme is believed to be an implementation of RFC 4941 [RFC4941], but without regenerating the addresses over time. The resulting interface IDs are constant across system bootstraps, and also constant across networks.

Assuming no flaws in the aforementioned algorithm, this scheme would remove any patterns from the SLAAC addresses.

> However, since the resulting interface IDs are constant across networks, these addresses may still be leveraged for host tracking purposes [RFC7217] [I-D.ietf-6man-ipv6-address-generation-privacy].

The benefit of this scheme is thus that the host may be less readily detected by applying heuristics to a scan, but, in the absence of concurrent use of temporary addresses, the host is liable to be tracked across visited networks.

3.1.1.4. Stable Privacy-Enhanced Addresses

In response to the predictability issues discussed in Section 3.1.1.1 and the privacy issues discussed in [I-D.ietf-6man-ipv6-address-generation-privacy], the IETF has standardized (in [RFC7217]) 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 subnet 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.

Implementation of this method (in replacement of Interface Identifiers based on IEEE identifiers) would eliminate any patterns from the Interface ID, thus benefiting user privacy and reducing the ease with which addresses can be scanned.

3.1.2. Dynamic Host Configuration Protocol version 6 (DHCPv6)

DHC DHCPv6 can be employed as a stateful address configuration mechanism, in which a server (the DHCPv6 server) leases IPv6 addresses to IPv6 hosts. As with the IPv4 counterpart, addresses are assigned according to a configuration-defined address range and policy, with some DHCPv6 servers assigning addresses sequentially, from a specific range. In such cases, addresses tend to be predictable.

> For example, if the prefix 2001:db8::/64 is used for assigning addresses on the local network, the DHCPv6 server might (sequentially) assign addresses from the range 2001:db8::1 - 2001:db8::100.

In most common scenarios, this means that the IID search space will be reduced from the original 64 bits, to 8 or 16 bits. RFC 5157 recommended that DHCPv6 instead issue addresses randomly from a large pool; that advice is repeated here. [I-D.ietf-dhc-stable-privacy-addresses] specifies an algorithm that can be employed by DHCPv6 servers to produce stable addresses which do not follow any specific pattern, thus resulting in an IID search space of 64 bits.

3.1.3. Manually-configured Addresses

In some scenarios, node addresses may be manually configured. This is typically the case for IPv6 addresses assigned to routers (since routers do not employ automatic address configuration) but also for

servers (since having a stable address that does not depend on the underlying link-layer address is generally desirable).

While network administrators are mostly free to select the IID from any value in the range 1 - 2^64, for the sake of simplicity (i.e., ease of remembering) they tend to select addresses with one of the following patterns:

- "low-byte" addresses: in which most of the bytes of the IID are set to 0 (except for the least significant byte).

- IPv4-based addresses: in which the IID embeds the IPv4 address of the network interface (as in 2001:db8::192.0.2.1)

- "service port" addresses: in which the IID embeds the TCP/UDP service port of the main service running on that node (as in 2001:db8::80 or 2001:db8::25)

- wordy addresses: which encode words (as in 2001:db8::dead:beef)

Each of these patterns is discussed in detail in the following subsections.

3.1.3.1. Low-byte Addresses

The most common form of low-byte addresses is that in which all the the bytes of the IID (except the least significant bytes) are set to zero (as in 2001:db8::1, 2001:db8::2, etc.). However, it is also common to find similar addresses in which the two lowest order 16-bit words are set to small numbers (as in 2001::db8::1:10, 2001:db8::2:10, etc.). Yet it is not uncommon to find IPv6 addresses in which the second lowest-order 16-bit word is set to a small value in the range 0-255, while the lowest-order 16-bit word varies in the range 0-65535. It should be noted that all of these address patterns are generally referred to as "low-byte addresses", even when, strictly speaking, it is not not only the lowest-order byte of the IPv6 address that varies from one address to another.

In the worst-case scenario, the search space for this pattern is 2^24 (although most systems can be found by searching 2^16 or even 2^8 addresses).

3.1.3.2. IPv4-based Addresses

The most common form of these addresses is that in which an IPv4 address is encoded in the lowest-order 32 bits of the IPv6 address (usually as a result of the notation of addresses in the form 2001:db8::192.0.2.1). However, it is also common for administrators

to encode one byte of the IPv4 address in each of the 16-bit words of the IID (as in e.g. 2001:db8::192:0:2:1).

For obvious reasons, the search space for addresses following this pattern is that of the corresponding IPv4 prefix (or twice the size of that search space if both forms of "IPv4-based addresses" are to be searched).

3.1.3.3. Service-port Addresses

Address following this pattern include the service port (e.g. 80 for HTTP) in the lowest-order byte of the IID, and set the rest of the IID to zero. There are a number of variants for this address pattern:

- The lowest-order 16-bit word may contain the service port, and the second lowest-order 16-bit word may be set to a number in the range 0-255 (as in e.g. 2001:db8::1:80).

- The lowest-order 16-bit word may be set to a value in the range 0-255, while the second lowest-order 16-bit word may contain the service port (as in e.g. 2001:db8::80:1).

- The service port itself might be encoded in decimal or in hexadecimal notation (e.g., an address embedding the HTTP port might be 2001:db8::80 or 2001:db8::50) -- with addresses encoding the service port as a decimal number being more common.

Considering a maximum of 20 popular service ports, the search space for addresses following this pattern is, in the worst-case scenario, 20 * 2^10.

3.1.3.4. Wordy Addresses

Since IPv6 address notation allows for a number of hexadecimal digits, it is not difficult to encode words into IPv6 addresses (as in, e.g., 2001:db8::dead:beef).

Addresses following this pattern are likely to be explored by means of "dictionary attacks", and therefore computing the corresponding search-space is not straight-forward.

3.1.4. IPv6 Addresses Corresponding to Transition/Co-existence Technologies

Some transition/co-existence technologies might be leveraged to reduce the target search space of remote address-scanning attacks, since they specify how the corresponding IPv6 address must be

generated. For example, in the case of Teredo [RFC4380], the 64-bit interface identifier is generated from the IPv4 address observed at a Teredo server along with a UDP port number.

3.1.5. IPv6 Address Assignment in Real-world Network Scenarios

Table 2, Table 3 and Table 4 provide a summary of the results obtained by [Gont-LACSEC2013] for web servers, nameservers, and mailservers, respectively. Table 5 provides a rough summary of the results obtained by [Malone2008] for IPv6 routers. Table 6 provides a summary of the results obtained by [Ford2013] for clients.

Address type	Percentage
IEEE-based	1.44%
Embedded-IPv4	25.41%
Embedded-Port	3.06%
ISATAP	0%
Low-byte	56.88%
Byte-pattern	6.97%
Randomized	6.24%

Table 2: Measured webserver addresses

Address type	Percentage
IEEE-based	0.67%
Embedded-IPv4	22.11%
Embedded-Port	6.48%
ISATAP	0%
Low-byte	56.58%
Byte-pattern	11.07%
Randomized	3.09%

Table 3: Measured nameserver addresses

Address type	Percentage
IEEE-based	0.48%
Embedded-IPv4	4.02%
Embedded-Port	1.07%
ISATAP	0%
Low-byte	92.65%
Byte-pattern	1.20%
Randomized	0.59%

Table 4: Measured mailserver addresses

Address type	Percentage
Low-byte	70%
IPv4-based	5%
SLAAC	1%
Wordy	<1%
Randomized	<1%
Teredo	<1%
Other	<1%

Table 5: Measured router addresses

Address type	Percentage
IEEE-based	7.72%
Embedded-IPv4	14.31%
Embedded-Port	0.21%
ISATAP	1.06%
Randomized	69.73%
Low-byte	6.23%
Byte-pattern	0.74%

Table 6: Measured client addresses

It should be clear from these measurements that a very high percentage of host and router addresses follow very specific patterns.

Table 6 shows that while around 70% of clients observed in this measurement appear to be using temporary addresses, there are still a significant amount exposing IEEE-based addresses, and addresses using embedded IPv4 (thus also revealing IPv4 addresses).

3.2. IPv6 Address Scanning of Remote Networks

While in IPv4 networks attackers have been able to get away with "brute force" scanning attacks (thanks to the reduced search space), successfully performing a brute-force scan of an entire /64 network would be infeasible. As a result, it is expected that attackers will leverage the IPv6 address patterns discussed in Section 3.1 to reduce the IPv6 address search space.

IPv6 address scanning of remote area networks should consider an additional factor not present for the IPv4 case: since the typical IPv6 host subnet is a /64, scanning an entire /64 could, in theory, lead to the creation of 2^64 entries in the Neighbor Cache of the last-hop router. Unfortunately, a number of IPv6 implementations have been found to be unable to properly handle large number of entries in the Neighbor Cache, and hence these address-scan attacks may have the side effect of resulting in a Denial of Service (DoS) attack [CPNI-IPv6] [RFC6583].

[RFC7421] discusses the "default" /64 boundary for host subnets, and the assumptions surrounding it. While there are reports of a handful of sites implementing host subnets of size /112 or smaller to reduce concerns about the above attack, such smaller subnets are likely to make address-based scanning more feasible, in addition to encountering the issues with non-/64 host subnets discussed in the above draft.

3.2.1. Reducing the subnet ID search space

When scanning a remote network, consideration is required to select which subnet IDs to choose. A typical site might have a /48 allocation, which would mean up to 65,000 or so host /64 subnets to be scanned.

However, in the same way the search space for the IID can be reduced, we may also be able to reduce the subnet ID space in a number of ways, by guessing likely address plan schemes, or using any complementary clues that might exist from other sources or observations. For example there are a number of documents available online (e.g. [RFC5375]) that provide recommendations for allocation of address space, which address various operational considerations, including: RIR assignment policy, ability to delegate reverse DNS zones to different servers, ability to aggregate routes efficiently, address space preservation, ability to delegate address assignment within the organization, ability to add allocate new sites/prefixes to existing entities without updating ACLs, and ability to de-aggregate and advertise sub-spaces via various AS interfaces.

Address plans might include use of subnets which:

- Run from low ID upwards, e.g. 2001:db8:0::/64, 2001:db8:1::/64, etc.

- Use building numbers, in hex or decimal form.

- Use VLAN numbers.

- Use IPv4 subnet number in a dual-stack target, e.g. a site with a /16 for IPv4 might use /24 subnets, and the IPv6 address plan may re-use the third byte as the IPv6 subnet ID.

- Use the service "colour", as defined for service-based prefix colouring, or semantic prefixes. For example, a site using a specific colouring for a specific service such as VoIP may reduce the subnet ID search space for those devices.

The net effect is that the address space of an organization may be highly structured, and allocations of individual elements within this structure may be predictable once other elements are known.

In general, any subnet ID address plan may convey information, or be based on known information, which may in turn be of advantage to an attacker.

3.3. IPv6 Address Scanning of Local Networks

IPv6 address scanning in Local Area Networks could be considered, to some extent, a completely different problem than that of scanning a remote IPv6 network. The main difference is that use of link-local multicast addresses can relieve the attacker of searching for unicast addresses in a large IPv6 address space.

> Obviously, a number of other network reconnaissance vectors (such as network snooping, leveraging Neighbor Discovery traffic, etc.) are available when scanning a local network. However, this section focuses only on address-scanning attacks (a la "ping sweep").

An attacker can simply send probe packets to the all-nodes link-local multicast address (ff02::1), such that responses are elicited from all local nodes.

Since Windows systems (Vista, 7, etc.) do not respond to ICMPv6 Echo Request messages sent to multicast addresses, IPv6 address-scanning tools typically employ a number of additional probe packets to elicit responses from all the local nodes. For example, unrecognized IPv6

options of type 10xxxxxx elicit ICMPv6 Parameter Problem, code 2, error messages.

Many address-scanning tools discover only IPv6 link-local addresses (rather than e.g. the global addresses of the target systems): since the probe packets are typically sent with the attacker's IPv6 link-local address, the "victim" nodes send the response packets using the IPv6 link-local address of the corresponding network interface (as specified by the IPv6 address selection rules [RFC6724]). However, sending multiple probe packets, with each packet employing addresses from different prefixes, typically helps to overcome this limitation.

> This technique is employed by the scan6 tool of the IPv6 Toolkit package [IPv6-Toolkit].

3.4. Existing IPv6 Address Scanning Tools

3.4.1. Remote IPv6 Network Scanners

IPv4 address scanning tools have traditionally carried out their task for probing an entire address range (usually the entire range of a target subnetwork). One might argue that the reason for which we have been able to get away with such somewhat "rudimentary" techniques is that the scale or challenge of the task is so small in the IPv4 world, that a "brute-force" attack is "good enough". However, the scale of the "address scanning" task is so large in IPv6, that attackers must be very creative to be "good enough". Simply sweeping an entire /64 IPv6 subnet would just not be feasible.

Many address scanning tools such as nmap [nmap2012] do not even support sweeping an IPv6 address range. On the other hand, the alive6 tool from [THC-IPV6] supports sweeping address ranges, thus being able to leverage some patters found in IPv6 addresses, such as the incremental addresses resulting from some DHCPv6 setups. Finally, the scan6 tool from [IPv6-Toolkit] supports sweeping address ranges, and can also leverage all the address patterns described in Section 3.1 of this document.

Clearly, a limitation of many of the currently-available tools for IPv6 address scanning is that they lack of an appropriately tuned "heuristics engine" that can help reduce the search space, such that the problem of IPv6 address scanning becomes tractable.

It should be noted that IPv6 network monitoring and management tools also need to build and maintain information about the hosts in their network. Such systems can no longer scan internal systems in a reasonable time to build a database of connected systems. Rather, such systems will need more efficient approaches, e.g. by polling

network devices for data held about observed IP addresses, MAC addresses, physical ports used, etc. Such an approach can also enhance address accountability, by mapping IPv4 and IPv6 addresses to observed MAC addresses. This of course implies that any access control mechanisms for querying such network devices, e.g. community strings for SNMP, should be set appropriately to avoid an attacker being able to gather address information remotely.

3.4.2. Local IPv6 Network Scanners

There are a variety of publicly-available local IPv6 network scanners:

- Current versions of nmap [nmap2012] implement this functionality.

- THC's IPv6 Attack Toolkit [THC-IPV6] includes a tool (alive6) that implements this functionality.

- SI6 Network's IPv6 Toolkit [IPv6-Toolkit] includes a tool (scan6) that implements this functionality.

3.5. Mitigations

IPv6 address-scanning attacks can be mitigated in a number of ways. A non-exhaustive list of the possible mitigations includes:

- Employing stable privacy-enhanced addresses [RFC7217] in replacement of addresses based on IEEE identifiers, such that any address patterns are eliminated.

- Employing Intrusion Prevention Systems (IPS) at the perimeter, such that address scanning attacks can be mitigated.

- Enforce IPv6 packet filtering where applicable (see e.g. [RFC4890]).

- If virtual machines are employed, and "resistance" to address scanning attacks is deemed as desirable, manually-configured MAC addresses can be employed, such that even if the virtual machines employ IEEE-derived IIDs, they are generated from non-predictable MAC addresses.

- When using DHCPv6, avoid use of sequential addresses. Ideally, the DHCPv6 server would allocate random addresses from a large pool.

- Use the "default" /64 size IPv6 subnet prefixes.

- In general, avoid being predictable in the way addresses are assigned.

It should be noted that some of the aforementioned mitigations are operational, while others depend on the availability of specific protocol features (such as [RFC7217]) on the corresponding nodes.

Additionally, while some resistance to address scanning attacks is generally desirable (particularly when lightweight mitigations are available), there are scenarios in which mitigation of some address-scanning vectors is unlikely to be a high-priority (if at all possible). And one should always remember that security by obscurity is not a reasonable defence in itself; it may only be one (relatively small) layer in a broader security environment.

Two of the techniques discussed in this document for local address-scanning attacks are those that employ multicasted ICMPv6 Echo Requests and multicasted IPv6 packets containing unsupported options of type 10xxxxxx. These two vectors could be easily mitigated by configuring nodes to not respond to multicasted ICMPv6 Echo Request (default on Windows systems), and by updating the IPv6 specifications (and/or possibly configuring local nodes) such that multicasted packets never elicit ICMPv6 error messages (even if they contain unsupported options of type 10xxxxxx).

> [I-D.gont-6man-ipv6-smurf-amplifier] proposes such update to the IPv6 specifications.

In any case, when it comes to local networks, there are a variety of network reconnaissance vectors. Therefore, even if address-scanning vectors are mitigated, an attacker could still rely on e.g. protocols employed for the so-called "opportunistic networking" (such as mDNS [RFC6762]), or eventually rely on network snooping as last resort for network reconnaissance. There is ongoing work in the IETF on extending mDNS, or at least DNS-based service discovery, to work across a whole site, rather than in just a single subnet, which will have associated security implications.

4. Leveraging the Domain Name System (DNS) for Network Reconnaissance

4.1. DNS Advertised Hosts

Any systems that are "published" in the DNS, e.g. MX mail relays, or web servers, will remain open to probing from the very fact that their IPv6 addresses are publicly available. It is worth noting that where the addresses used at a site follow specific patterns, publishing just one address may lead to a threat upon the other hosts.

Additionally, we note that publication of IPv6 addresses in the DNS should not discourage the elimination of IPv6 address patterns: if any address patterns are eliminated from addresses published in the DNS, an attacker may have to rely on performing dictionary-based DNS lookups in order to find all systems in a target network (which is generally less reliable and more time/traffic consuming than mapping nodes with predictable IPv6 addresses).

4.2. DNS Zone Transfers

A DNS zone transfer can readily provide information about potential attack targets. Restricting zone transfers is thus probably more important for IPv6, even if it is already good practice to restrict them in the IPv4 world.

4.3. DNS Brute Forcing

Attackers may employ DNS brute-forcing techniques by testing for the presence of DNS AAAA records against commonly used host names.

4.4. DNS Reverse Mappings

[van-Dijk] describes an interesting technique that employs DNS reverse mappings for network reconnaissance. Essentially, the attacker walks through the "ip6.arpa" zone looking up PTR records, in the hopes of learning the IPv6 addresses of hosts in a given target network (assuming that the reverse mappings have been configured, of course). What is most interesting about this technique is that it can greatly reduce the IPv6 address search space.

Basically, an attacker would walk the ip6.arpa zone corresponding to a target network (e.g. "0.8.0.0.8.b.d.0.1.0.0.2.ip6.arpa." for "2001:db8:80::/48"), issuing queries for PTR records corresponding to the domain names "0.0.8.0.0.8.b.d.0.1.0.0.2.ip6.arpa.", "1.0.8.0.0.8.b.d.0.1.0.0.2.ip6.arpa.", etc. If, say, there were PTR records for any hosts "starting" with the domain name "0.0.8.0.0.8.b.d.0.1.0.0.2.ip6.arpa." (e.g., the ip6.arpa domain name corresponding to the IPv6 address 2001:db8:80::1), the response would contain an RCODE of 0 (no error). Otherwise, the response would contain an RCODE of 4 (NXDOMAIN). As noted in [van-Dijk], this technique allows for a tremendous reduction in the "IPv6 address" search space.

> [I-D.howard-isp-ip6rdns] analyzes different approaches and considerations for ISPs in managing the ip6.arpa zone for IPv6 address space assigned to many customers, which may affect the technique described in this section.

5. Leveraging Local Name Resolution and Service Discovery Services

A number of protocols allow for unmanaged local name resolution and service. For example, multicast DNS (mDNS) [RFC6762] and DNS Service Discovery (DNS-SD) [RFC6763], or Link-Local Multicast Name Resolution (LLMNR) [RFC4795], are examples of such protocols.

> Besides the Graphical User Interfaces (GUIs) included in products supporting such protocols, command-line tools such as mdns-scan [mdns-scan] and mzclient can help discover IPv6 hosts employing mDNS/DNS-SD.

6. Public Archives

Public mailing-list archives or Usenet news messages archives may prove a useful channel for an attacker, since hostnames and/or IPv6 addresses could be easily obtained by inspection of the (many) "Received from:" or other header lines in the archived email or Usenet news messages.

7. Application Participation

Peer-to-peer applications often include some centralized server which coordinates the transfer of data between peers. For example, BitTorrent [BitTorrent] builds swarms of nodes that exchange chunks of files, with a tracker passing information about peers with available chunks of data between the peers. Such applications may offer an attacker a source of peer addresses to probe.

8. Inspection of the IPv6 Neighbor Cache and Routing Table

Information about other systems connected to the local network might be readily available from the Neighbor Cache [RFC4861] and/or the routing table of any system connected to such network. SAVI [RFC6620] also builds a cache of IPv6 and link-layer addresses (without actively participating in the Neighbor Discovery packet exchange), and hence is another source of similar information.

These data structures could be inspected either via "login" access or via SNMP. While this requirement may limit the applicability of this technique, there are a number of scenarios in which this technique might be of use. For example, security audit tools might be provided with the necessary credentials such that the Neighbor Cache and the routing table of all systems for which the tool has "login" or SNMP access can be automatically gleaned. On the other hand, IPv6 worms [V6-WORMS] could leverage this technique for the purpose of spreading on the local network, since they will typically have access to the Neighbor Cache and routing table of an infected system.

Section 2.5.1.4 of [I-D.ietf-opsec-v6] discusses additional considerations for the inspection of the IPv6 Neighbor Cache.

9. Inspection of System Configuration and Log Files

Nodes are generally configured with the addresses of other important local computers, such as email servers, local file servers, web proxy servers, recursive DNS servers, etc. The /etc/hosts file in UNIX, SSH known_hosts files, or the Microsoft Windows registry are just some examples of places where interesting information about such systems might be found.

Additionally, system log files (including web server logs, etc.) may also prove a useful channel for an attacker.

While the required credentials to access the aforementioned configuration and log files may limit the applicability of this technique, there are a number of scenarios in which this technique might be of use. For example, security audit tools might be provided with the necessary credentials such that these files can be automatically accessed. On the other hand, IPv6 worms could leverage this technique for the purpose of spreading on the local network, since they will typically have access to these files on an infected system [V6-WORMS].

10. Gleaning Information from Routing Protocols

Some organizational IPv6 networks employ routing protocols to dynamically maintain routing information. In such an environment, a local attacker could become a passive listener of the routing protocol, to determine other valid subnets/prefixes and some router addresses within that organization [V6-WORMS].

11. Gleaning Information from IP Flow Information Export (IPFIX)

IPFIX [RFC7012] can aggregate the flows by source addresses, and hence may be leveraged for obtaining a list of "active" IPv6 addresses. Additional discussion of IPFIX can be found in Section 2.5.1.2 of [I-D.ietf-opsec-v6].

12. Obtaining Network Information with traceroute6

IPv6 traceroute [traceroute6] can be employed to find router addresses and valid network prefixes.

13. Gleaning Information from Network Devices Using SNMP

SNMP can be leveraged to obtain information from a number of data structures such as the Neighbor Cache [RFC4861], the routing table, and the SAVI [RFC6620] cache of IPv6 and link-layer addresses. SNMP access should be secured, such that unauthorized access to the aforementioned information is prevented.

14. Obtaining Network Information via Traffic Snooping

Snooping network traffic can help in discovering active nodes in a number of ways. Firstly, each captured packet will reveal the source and destination of the packet. Secondly, the captured traffic may correspond to network protocols that transfer information such as host or router addresses, network topology information, etc.

15. Conclusions

In this document we have discussed issues around host-based scanning of IPv6 networks. We have shown why a /64 host subnet may be more vulnerable to address-based scanning that might intuitively be thought, and how an attacker might reduce the target search space when scanning.

We have described a number of mitigations against host-based scanning, including the replacement of traditional SLAAC with stable privacy-enhanced IIDs (which will require support from system vendors). We have also offered some practical guidance, around the principle of avoiding having predictability in host addressing schemes. Finally, examples of scanning approaches and tools are discussed in the Appendices.

While most early IPv6-enabled networks remain dual-stack, they are more likely to be scanned and attacked over IPv4 transport, and one may argue that the IPv6-specific considerations discussed here are not of an immediate concern. However, an early IPv6 deployment within a dual-stack network may be seen by an attacker as a potentially "easier" target, if the implementation of security policies are not as strict for IPv6 (for whatever reason). As and when IPv6-only networks become more common, the considerations in this document will be of much greater importance.

16. 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.

17. Security Considerations

This document explores the topic of Network Reconnaissance in IPv6 networks. It analyzes the feasibility of address-scan attacks in IPv6 networks, and showing that the search space for such attacks is typically much smaller than the one traditionally assumed (64 bits). Additionally, it explores a plethora of other network reconnaissance techniques, ranging from inspecting the IPv6 Network Cache of an attacker-controlled system, to gleaning information about IPv6 addresses from public mailing-list archives or Peer-To-Peer (P2P) protocols.

We expect traditional address-scanning attacks to become more and more elaborated (i.e., less "brute force"), and other network reconnaissance techniques to be actively explored, as global deployment of IPv6 increases and. more specifically, as more IPv6-only devices are deployed.

18. Acknowledgements

The authors would like to thank Ray Hunter, who provided valuable text that was readily incorporated into Section 3.2.1 of this document.

The authors would like to thank (in alphabetical order) Wesley George, Marc Heuse, Ray Hunter, Libor Polcak, Tomoyuki Sahara, Jan Schaumann, Arturo Servin, and Eric Vyncke, for providing valuable comments on earlier versions of this document.

Part of the contents of this document are based on the results of the project "Security Assessment of the Internet Protocol version 6 (IPv6)" [CPNI-IPv6], carried out by Fernando Gont on behalf of the UK Centre for the Protection of National Infrastructure (CPNI). Fernando Gont would like to thank the UK CPNI for their continued support.

19. References

19.1. Normative References

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

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

[RFC6620] Nordmark, E., Bagnulo, M., and E. Levy-Abegnoli, "FCFS SAVI: First-Come, First-Served Source Address Validation Improvement for Locally Assigned IPv6 Addresses", RFC 6620, May 2012.

[RFC6724] Thaler, D., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, September 2012.

[RFC4380] Huitema, C., "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)", RFC 4380, February 2006.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[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.

[RFC7012] Claise, B. and B. Trammell, "Information Model for IP Flow Information Export (IPFIX)", RFC 7012, September 2013.

[RFC7136] Carpenter, B. and S. Jiang, "Significance of IPv6 Interface Identifiers", RFC 7136, February 2014.

[RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC)", RFC 7217, April 2014.

19.2. Informative References

[RFC4795] Aboba, B., Thaler, D., and L. Esibov, "Link-local Multicast Name Resolution (LLMNR)", RFC 4795, January 2007.

[RFC4890] Davies, E. and J. Mohacsi, "Recommendations for Filtering ICMPv6 Messages in Firewalls", RFC 4890, May 2007.

[RFC5157] Chown, T., "IPv6 Implications for Network Scanning", RFC 5157, March 2008.

[RFC5375] Van de Velde, G., Popoviciu, C., Chown, T., Bonness, O., and C. Hahn, "IPv6 Unicast Address Assignment Considerations", RFC 5375, December 2008.

[RFC6583] Gashinsky, I., Jaeggli, J., and W. Kumari, "Operational Neighbor Discovery Problems", RFC 6583, March 2012.

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, February 2013.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, February 2013.

[I-D.gont-6man-ipv6-smurf-amplifier]
Gont, F. and W. Liu, "Security Implications of IPv6 Options of Type 10xxxxxx", draft-gont-6man-ipv6-smurf-amplifier-03 (work in progress), March 2013.

[I-D.howard-isp-ip6rdns]
Howard, L., "Reverse DNS in IPv6 for Internet Service Providers", draft-howard-isp-ip6rdns-07 (work in progress), February 2015.

[RFC7421] Carpenter, B., Chown, T., Gont, F., Jiang, S., Petrescu, A., and A. Yourtchenko, "Analysis of the 64-bit Boundary in IPv6 Addressing", RFC 7421, January 2015.

[I-D.ietf-6man-default-iids]
Gont, F., Cooper, A., Thaler, D., and W. Will, "Recommendation on Stable IPv6 Interface Identifiers", draft-ietf-6man-default-iids-02 (work in progress), January 2015.

[I-D.ietf-6man-ipv6-address-generation-privacy]
Cooper, A., Gont, F., and D. Thaler, "Privacy Considerations for IPv6 Address Generation Mechanisms", draft-ietf-6man-ipv6-address-generation-privacy-05 (work in progress), April 2015.

[I-D.ietf-dhc-stable-privacy-addresses]
Gont, F. and W. Will, "A Method for Generating Semantically Opaque Interface Identifiers with Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", draft-ietf-dhc-stable-privacy-addresses-02 (work in progress), April 2015.

[I-D.ietf-opsec-v6]
Chittimaneni, K., Kaeo, M., and E. Vyncke, "Operational Security Considerations for IPv6 Networks", draft-ietf-opsec-v6-06 (work in progress), March 2015.

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

[V6-WORMS]
Bellovin, S., Cheswick, B., and A. Keromytis, "Worm propagation strategies in an IPv6 Internet", ;login:, pages 70-76, February 2006, <https://www.cs.columbia.edu/~smb/papers/v6worms.pdf>.

[Malone2008]
Malone, D., "Observations of IPv6 Addresses", Passive and Active Measurement Conference (PAM 2008, LNCS 4979), April 2008, <http://www.maths.tcd.ie/~dwmalone/p/addr-pam08.pdf>.

[mdns-scan]
Poettering, L., "mdns-scan(1) manual page", 2012, <http://manpages.ubuntu.com/manpages/precise/man1/mdns-scan.1.html>.

[nmap2012]
Fyodor, , "nmap - Network exploration tool and security / port scanner", 2012, <http://insecure.org>.

[VBox2011]
VirtualBox, , "Oracle VM VirtualBox User Manual, version 4.1.2", August 2011, <http://www.virtualbox.org>.

[vmesx2011]
vmware, , "Setting a static MAC address for a virtual NIC", vmware Knowledge Base, August 2011, <http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=219>.

[traceroute6]
FreeBSD, , "FreeBSD System Manager's Manual: traceroute6(8) manual page", 2009, <https://www.freebsd.org/cgi/man.cgi?query=traceroute6>.

[Gont-DEEPSEC2011]
Gont, F., "Results of a Security Assessment of the Internet Protocol version 6 (IPv6)", DEEPSEC 2011 Conference, Vienna, Austria, November 2011, 2011, <http://www.si6networks.com/presentations/deepsec2011/fgont-deepsec2011-ipv6-security.pdf>.

[Gont-LACSEC2013]
Gont, F., "IPv6 Network Reconnaissance: Theory & Practice", LACSEC 2013 Conference, Medellin, Colombia, May 2013, 2013, <http://www.si6networks.com/presentations/lacnic19/lacsec2013-fgont-ipv6-network-reconnaissance.pdf>.

[Ford2013]
Ford, M., "IPv6 Address Analysis - Privacy In, Transition Out", 2013, <http://www.internetsociety.org/blog/2013/05/ipv6-address-analysis-privacy-transition-out>.

[THC-IPV6]
"THC-IPV6", <http://www.thc.org/thc-ipv6/>.

[IPv6-Toolkit]
"SI6 Networks' IPv6 Toolkit", <http://www.si6networks.com/tools/ipv6toolkit>.

[BitTorrent]
"BitTorrent", <http://en.wikipedia.org/wiki/BitTorrent>.

[van-Dijk]
van Dijk, P., "Finding v6 hosts by efficiently mapping ip6.arpa", 2012, <http://7bits.nl/blog/2012/03/26/finding-v6-hosts-by-efficiently-mapping-ip6-arpa>.

Appendix A. Implementation of a full-fledged IPv6 address-scanning tool

This section describes the implementation of a full-fledged IPv6 address scanning tool. Appendix A.1 discusses the selection of host probes. Appendix A.2 describes the implementation of an IPv6 address scanner for local area networks. Appendix A.3 outlines ongoing work on the implementation of a general (i.e., non-local) IPv6 host scanner.

A.1. Host-probing considerations

A number of factors should be considered when selecting the probe types and the probing-rate for an IPv6 address scanning tool.

Firstly, some hosts (or border firewalls) might be configured to block or rate-limit some specific packet types. For example, it is usual for host and router implementations to rate-limit ICMPv6 error traffic. Additionally, some firewalls might be configured to block or rate-limit incoming ICMPv6 echo request packets (see e.g. [RFC4890]).

> As noted earlier in this document, Windows systems simply do not respond to ICMPv6 echo requests sent to multicast IPv6 addresses.

Among the possible probe types are:

- ICMPv6 Echo Request packets (meant to elicit ICMPv6 Echo Replies),

- TCP SYN segments (meant to elicit SYN/ACK or RST segments),

- TCP segments that do not contain the ACK bit set (meant to elicit RST segments),

- UDP datagrams (meant to elicit a UDP application response or an ICMPv6 Port Unreachable),

- IPv6 packets containing any suitable payload and an unrecognized extension header (meant to elicit ICMPv6 Parameter Problem error messages), or,

- IPv6 packets containing any suitable payload and an unrecognized option of type 10xxxxxx (such that a ICMPv6 Parameter Problem error message is elicited)

Selecting an appropriate probe packet might help conceal the ongoing attack, but may also be actually necessary if host or network configuration causes certain probe packets to be dropped. In some cases, it might be desirable to insert some IPv6 extension headers before the actual payload, such that some filtering policies can be circumvented.

Another factor to consider is the host-probing rate. Clearly, the higher the rate, the smaller the amount of time required to perform the attack. However, the probing-rate should not be too high, or else:

1. the attack might cause network congestion, thus resulting in packet loss

2. the attack might hit rate-limiting, thus resulting in packet loss

3. the attack might reveal underlying problems in the Neighbor Discovery implementation, thus leading to packet loss and possibly even Denial of Service

Packet-loss is undesirable, since it would mean that an "alive" node might remain undetected as a result of a lost probe or response. Such losses could be the result of congestion (in case the attacker is scanning a target network at a rate higher than the target network can handle), or may be the result of rate-limiting as it would be typically the case if ICMPv6 is employed for the probe packets. Finally, as discussed in [CPNI-IPv6] and [RFC6583], some IPv6 router implementations have been found to be unable to perform decent resource management when faced with Neighbor Discovery traffic involving a large number of local nodes. This essentially means that regardless of the type of probe packets, an address scanning attack might result in a Denial of Service (DoS) of the target network, with the same (or worse) effects as that of network congestion or rate-limiting.

The specific rates at which each of these issues may come into play vary from one scenario to another, and depend on the type of deployed routers/firewalls, configuration parameters, etc.

A.2. Implementation of an IPv6 local address-scanning tool

scan6 [IPv6-Toolkit] is prototype IPv6 local address scanning tool, which has proven to be effective and efficient for the discovery of IPv6 hosts on a local network.

The scan6 tool operates (roughly) as follows:

1. The tool learns the local prefixes used for auto-configuration, an generates/configures one address for each local prefix (in addition to a link-local address).

2. An ICMPv6 Echo Request message destined to the all-nodes on-link multicast address (ff02::1) is sent with each of the addresses "configured" in the previous step. Because of the different Source Addresses, each probe causes the victim nodes to use different Source Addresses for the response packets (this allows the tool to learn virtually all the addresses in use in the local network segment).

3. The same procedure of the previous bullet is performed, but this time with ICMPv6 packets that contain an unrecognized option of type 10xxxxxx, such that ICMPv6 Parameter Problem error messages are elicited. This allows the tool to discover e.g. Windows

nodes, which otherwise do not respond to multicasted ICMPv6 Echo Request messages.

4. Each time a new "alive" address is discovered, the corresponding Interface-ID is combined with all the local prefixes, and the resulting addresses are probed (with unicasted packets). This can help to discover other addresses in use on the local network segment, since the same Interface ID is typically used with all the available prefixes for the local network.

The aforementioned scheme can fail to discover some addresses for some implementation. For example, Mac OS X employs IPv6 addresses embedding IEEE-identifiers (rather than "temporary addresses") when responding to packets destined to a link-local multicast address, sourced from an on-link prefix.

A.3. Implementation of a IPv6 remote address-scanning tool

An IPv6 remote address scanning tool, could be implemented with the following features:

- The tool can be instructed to target specific address ranges (e.g. 2001:db8::0-10:0-1000)

- The tool can be instructed to scan for SLAAC addresses of a specific vendor, such that only addresses embedding the corresponding IEEE OUIs are probed.

- The tool can be instructed to scan for SLAAC addresses that employ a specific IEEE OUI.

- The tool can be instructed to discover virtual machines, such that a given IPv6 prefix is only scanned for the address patterns resulting from virtual machines.

- The tool can be instructed to scan for low-byte addresses.

- The tool can be instructed to scan for wordy-addresses, in which case the tool selects addresses based on a local dictionary.

- The tool can be instructed to scan for IPv6 addresses embedding TCP/UDP service ports, in which case the tool selects addresses based on a list of well-known service ports.

- The tool can be specified an IPv4 address range in use at the target network, such that only IPv4-based IPv6 addresses are scanned.

The scan6 tool of [IPv6-Toolkit] implements all these techniques/ features.

Authors' Addresses

Fernando Gont
Huawei Technologies
Evaristo Carriego 2644
Haedo, Provincia de Buenos Aires 1706
Argentina

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

Tim Chown
University of Southampton
Highfield
Southampton , Hampshire   SO17 1BJ
United Kingdom

Email: tjc@ecs.soton.ac.uk