Challenges for Host Discovery and Malware Propagation in the IPv6 Address Space.

Luis MartinGarcia
Department of Telematic Systems Engineering.
Telecommunication Engineering School.
Universidad Politecnica de Madrid.
{luis.mgarcia@alumnos.upm.es}

***Abstract*—The shortage of IPv4 addresses has been a known problem for decades and while nobody knows when the new Internet Protocol version 6 will be completely deployed, it seems clear that, considering the actual growth rate of the Internet, most hosts will speak IPv6 at some point in the future. The new default 64-bit subnet address space introduces important challenges for the remote discovery of active network hosts as it makes it infeasible to conduct brute-force searches. While the ability to discover active hosts may be maliciously used by worms or intruders looking for potential targets, it is often used by system administrators to perform legitimate tasks like network inventory or rogue host detection. This paper will present some heuristics and techniques that may allow the discovery of active hosts in IPv6 networks, without the need to sweep enormous subnet address spaces.**

I. Introduction

The exhaustion of IPv4 addresses has been a concern for decades and while new methods and technologies like Network Address Translation (NAT) or Classless Inter-Domain Routing (CIDR), have been developed over the time to alleviate the problem, it seems clear that the IPv4 address allocation pool will be exhausted soon. At time of writing, there are only seven /8 blocks out of 256 in state unallocated in the IANA IPv4 Address Space Registry [1], and some authors estimate that the address pool will deplete by early February 2011[2].

Although the IANA or the different Regional Internet Registries (RIRs) could start allocating address blocks that were initially reserved for experimental use, the IPv4 address space is very limited and the whole Internet will be forced to move to the next version of the Internet Protocol sooner or later. Among other things, the new version of IP, the Internet Protocol version 6 or IPv6, extends the available address space from 32 bits to 128, which makes it virtually impossible to run out of addresses, even with very wasteful allocation schemes [3].

IPv6 addresses are divided in two parts. The first one is called the network prefix and its purpose is to identify a particular link. The other part is called the interface identifier and its purpose is to uniquely identify a network node on the link. Although prefixes and interface identifiers may have a variable length, the current IETF addressing policy suggests allocating fixed /48 network prefixes to end sites and dividing the remaining 80 bits in a 16-bit subnet identifier and a 64-bit interface identifier. This means that while in IPv4 subnets typically had between 4 and 2^{16} addresses, in IPv6, subnets will have 2^{64} addresses, 2^{32} times the entire IPv4 address space.

The new subnet size in IPv6 introduces important challenges for the discovery of active network hosts. In the IPv4 world, Internet worms or security scanners perform host discovery by brute-forcing entire subnet address spaces, sending probes to every possible address on the subnet and considering hosts active if they reply to any of the probes. However, in IPv6, sweeping an entire /64 subnet is inherently infeasible. Assuming that a network host could actively probe 2^{24} hosts per second (around 16 million), which would require a bandwidth of approximately 7.5Gbps, sweeping an entire 2^{64} subnet address space would take 2^{40} seconds, or about 35,000 years. Consequently, host discovery in IPv6 networks can't be performed by brute-force but using other techniques that allow to reduce the search space to something reasonable.

The rest of this paper is organized as follows. Section II will review related work in the area. Section III will briefly discuss the current approach of worms and network scanners in IPv4 networks. Section IV will present a few techniques that allow one host to gather IPv6 addresses of other hosts on the same link. Section V will cover techniques for the discovery of hosts that are out of the local link. Section VI will present those techniques that apply regardless of the position of the hosts. Finally, section VII will present some conclusions and pointers for future research.

II. Related Work

Despite its significant importance, the discovery of hosts in IPv6 networks is a problem that has not attracted much attention from either academic researchers or the hacker community. However, there are a two main documents of interest in this direction. On one hand, [4] is an informative RFC that discusses the implications of the new version of the Internet Protocol for network scanning and the approaches that administrators could take when planning their site address allocation and management strategies as part of a defence-in-depth approach to network security. [5], on the other hand, discusses the problem of discovery from the perspective of network worms that try to find other hosts to propagate and continue their existence. It is worth noting that while regular attackers tend to choose specific networks with the intent of finding as much information as possible about them, worms usually do not care about the location of vulnerable hosts, as their only goal is to propagate and survive.

There are also a few other documents worth mentioning. [6] presents a detailed study of the security problems that affect IPv6 hosts and briefly discusses IPv6 network reconnaissance. Additionally, the THC group publishes an open source tool that performs various attacks on IPv6 networks [7]. One of its components, *alive6*, provides basic IPv6 host discovery capabilities.

III. Host Discovery in IPv4

In today's IPv4 networks, worms and network scanners do not consider the size of the address space to be a limiting factor for their purpose. The IPv4 address space is 32 bits wide, this is, less than 5 billion addresses. A host like the one described in the previous section could sweep the entire Internet in less than 5 minutes. For this reason, current worms and network reconnaissance tools may not pay much attention to the efficiency of their discovery algorithms but simply attempt to detect active hosts, sending packets to all possible addresses and listening for responses.

Due to their distributed nature, worms typically sweep the address space randomly [8]. This reduces the chances of two distant worms targeting the same set of addresses. Network scanners, however, are usually run one instance at a time and cover the address space sequentially.

The actual detection can be performed using various types of packets. Worms looking for specific vulnerabilities are likely to use TCP packets with the SYN flag set and the destination port field set to the port number where the service to be exploited listens for connections. Network scanners may use more advanced techniques. For example, the popular *Nmap Security Scanner* [9] uses ARP requests when the targets are on the local Ethernet network and sends an ICMP echo request, a TCP SYN packet to port 443, a TCP ACK packet to port 80 and an ICMP timestamp request, when the targets are more than one hop away.

IV. Discovery of IPv6 Hosts On-Link

As discussed above, the default 64-bit subnet space in IPv6 makes it infeasible to test every address on a given subnet. For this reason, if worms and network scanners want to survive the transition to IPv6, they'll need to use alternative techniques to detect the presence of other hosts on the network.

In the IPv6 world, every address other than the unspecified address has a specific scope; that is, a topological span within which the address may be used as a unique identifier for an interface or set of interfaces [10]. For unicast addresses there are two scopes currently defined: the *link-local scope*, for uniquely identifying interfaces within a single link and the *global scope*, for uniquely identifying interfaces within the Internet. For multicast addresses, there are fourteen possible scopes, which will not be discussed in this document. This section will present a few techniques that may be used to discover hosts on the same link. Although many of the addresses discovered from the inside of the network will have a local scope, certain global scoped addresses may also be gathered from that point.

A. Neighbor Discovery

IPv6 introduces a new protocol, the IPv6 Neighbor Discovery (ND) protocol, that could be seen as the equivalent of ARP for the IPv6 world because it is used to map IP addresses to local link-layer addresses. However, unlike ARP, it also provides mechanisms to discover routers on the local link, detect duplicate addresses or unreachable hosts, or inform nodes when there is a better gateway to send traffic through.

Due to the use of multicast, a significant number of messages exchanged during normal executions of the Neighbor Discovery protocol are received by all hosts on the link. This allows attackers to process these messages and extract IP addresses from them.

B. Traffic redirection

There are many attacks that can be carried out by an attacker through the Network Discovery protocol. One of them is equivalent to the classic ARP-poisoning attack. With ND, in order to determine the link-layer address of a node that is attached to the local link, the node that requires the address sends a Neighbor Solicitation message to a multicast address specified by the target address. The target node, which continuously listens to the multicast address, receives the solicitation and replies with a Neighbor Advertisement message. An attacker can get packets for legitimate nodes to be sent to his own link-layer address, sending malicious Neighbor Solicitation and Neighbor Advertisement messages. If the spoofed link-layer address is valid, as long as the attacker responds to the unicast Neighbor Solicitation messages sent as part of the Neighbor Unreachability Detection, packets will continue to be directed to him [11].

C. Router impersonation

Instead of redirecting traffic destined to a particular host, attackers may choose to impersonate the local router, multicasting legitimate-looking Router Advertisements or unicasting Router Advertisements in response to multicast Router Solicitations. If other hosts select the attacker as their default router, the attacker will be able to receive their network traffic and learn the addresses of others hosts in and outside the local link.

Even if hosts already have a configured default gateway and decide to ignore new router advertisements, the ICMPv6 Redirect message may be used to alter their routing table. ICMPv6 Redirects are sent by routers when a given system is choosing a wrong local router for a packet. To prevent malicious hosts from injecting false routes, ICMPv6 Redirects are required to include as much of the offending packet as can fit without the redirect packet exceeding the minimum MTU required to support IPv6 [12]. However, an attacker may easily subvert such protection as follows [13]:

1) The attacker sends an ICMPv6 Echo request to the victim with the source address of the route he wants to redirect.
2) The attacker can easily guess the ICMPv6 Echo reply that the victim will produce in response so he crafts an ICMPv6 Redirect message with the forged source address of the victim's default router and attaches a copy of the guessed Echo reply.
3) The attacker waits enough time to allow the victim to send the Echo reply and then sends the Redirect message created in step 2.
4) When the victim receives the message, updates its local routing table with the route injected by the attacker.

D. Passive eavesdropping

Although hub-based networks are obsolete these days, unprotected or WEP-based wireless networks have brought back the old eavesdropping problem. An attacker with the ability to capture network traffic will be able to detect active hosts by observing the source and the destination address of the captured packets. If the activity of the network is high enough, the attacker may gather a very complete list of addresses.

E. Multicast pings

The support for multicast is a core feature of IPv6 but it can be abused for host discovery. There are a few multicast groups standardized by the IANA, some of which are required to be joined by all IPv6-capable hosts. For example, all hosts on a link are required to join the all-nodes link-local multicast group and all routers on a link are required to join the all-routers link-local group [12]. Table I lists some of the groups that are defined at time of writing.

An attacker may join and send traffic to these multicast groups and use the responses to determine which hosts are connected to the network.

1) ICMPv6 Echo messages: The simplest (and probably the best) option is to send ICMPv6 Echo requests to the all-nodes link-local group. Hosts on the local link will respond with ICMPv6 Echo response messages, revealing their existence to the attacker.

2) ICMPv6 Multicast Listener Discovery messages: Multicast Listener Discovery (MLD) is a protocol used by an IPv6 router that wants to discover which nodes wish to receive multicast packets on its directly attached links, and to discover specifically which multicast addresses are of interest to those neighboring nodes [14]. This protocol, which is implemented through a set of three ICMPv6 message types, may be used by an attacker to detect the presence of hosts subscribed to a multicast group. Only hosts on the same link can be detected using this protocol, as all nodes are required to verify that the IPv6 Source Address of all received MLD messages is a link-local address.

The attacker can either send a Multicast-General-Listener Query to learn which multicast addresses have listeners on an attached link or a Multicast-Address-Specific Query to learn if a particular multicast address has any listeners. Both queries will cause listeners to send back Multicast Listener Report messages, whose source addresses can be gathered by the attacker.

3) ICMPv6 Node Information messages: The Node Information protocol (NI) is a mechanism for discovering information about names and addresses that also provides facilities that are not found in the DNS, like the discovery of relationships between addresses without reference to names [15]. Some of its functions overlap with the DNS but are useful in serverless environments. With this protocol, a host may learn the addresses and names of nodes on the other end of a point-to-point link or nodes on a shared-medium link such as Ethernet. The protocol is implemented through two ICMPv6 message types: Node Information Queries and Node Information Replies. The typical usage scenario is the following: two nodes, A and B are attached to the same link, and A wants to establish communication with B. A knows B's host name but not its IP address and has no DNS server configured. To resolve B's address, A sends a Node Information Query that contains B's host name, to the all-nodes link-local multicast address. B receives the query and produces a Node Information Reply that includes its IP address.

An attacker may abuse the protocol to discover other hosts on the link. In theory, hosts that receive NI queries must silently discard queries for addresses or names that do not belong to them. However, there is a special query type called NOOP, which has no defined flags and no data field, to which nodes may reply to tell the querier that they are up and reachable and implement the Node Information protocol. Because of this, the attacker may issue NOOP queries destined to the all-nodes link-local multicast address and gather the addresses of other hosts

Multicast address	Description
FF01:0:0:0:0:0:0:1	Interface-local all-nodes
FF01:0:0:0:0:0:0:2	Interface-local all-routers
FF01:0:0:0:0:0:0:FB	Interface-local multicast DNS
FF02:0:0:0:0:0:0:1	Link-local all-nodes
FF02:0:0:0:0:0:0:2	Link-local all-routers
FF02:0:0:0:0:0:0:9	Link-local all-RIP-routers
FF02:0:0:0:0:0:0:A	Link-local all-EIGRP-routers
FF02:0:0:0:0:0:0:FB	Link-local multicast DNS
FF02:0:0:0:0:0:1:2	Link-local all-DHCP-agents
FF05:0:0:0:0:0:0:2	Site-local all-routers
FF05:0:0:0:0:0:0:FB	Site-local multicast DNS
FF05:0:0:0:0:0:1:3	Site-local all-DHCP-agents

Table I
PARTIAL LIST OF STANDARDIZED IPV6 MULTICAST ADDRESSES.

on the link based on the received NI replies.

4) Malformed IPv6 messages: Although hosts cannot send ICMPv6 error messages as a result of receiving packets destined to an IPv6 multicast address, there are two exceptions to this rule: the Packet Too Big Message to allow Path MTU discovery to work for IPv6 multicast, and the Parameter Problem Message which reports an unrecognized IPv6 option [16]. This means that, even if for some reason ICMPv6 Echo messages are blocked, an attacker may send IPv6 packets to the multicast address with invalid parameters that cause the nodes on the group to discard the packet and send an ICMPv6 Parameter Problem message in response.

V. DISCOVERY OF IPV6 HOSTS OFF-LINK

This section presents some techniques for the discovery of hosts out of the local link.

A. Looking glass services

A public looking glass is a network service, typically offered either from a web page or from a telnet session, that allows users to access network's routing information from various network vantage points. Looking glass services may be used to gather BGP routing information of various Internet Autonomous Systems. This can be useful for attackers seeking to gather IPv6 prefixes. However, although some /64 prefixes may be observed occasionally, most routes use prefixes of 48 bits or less, which, in general, are too generic for host discovery.

B. Dual-stack devices

The transition from IPv4 to IPv6 will be gradual. Until IPv6 is completely deployed in the Internet, a number of transition mechanisms are needed to allow IPv6 hosts to reach IPv4 services and enable IPv6 networks to reach other IPv6 networks on the Internet over the current IPv4 infrastructure. There are a number of standardized solutions for the transition phase. Perhaps the most common approach are dual-stack devices, that is, routers or end nodes that implement both IPv4 and IPv6 an allow communication with both protocols, using IPv6 when possible and switching to IPv4 when needed. This requires applications to be version-aware: clients need to choose between IPv4 or IPv6 before initiating communications and servers need to accept both types of packets. Host discovery can be performed very easily on networks equipped with dual-stack nodes, as an attacker may detect active hosts "falling-back to IPv4", as this considerably reduces the search space.

Once the attacker learns a host's IPv4 address, he may use that address to perform whatever action he needs. However, there may be services that are only accessible through IPv6. In this case the attacker would need to determine the host's IPv6 address from the IPv4 one. This process may be a bit tricky:

If the attacker is on the same link as its target, he could try to use the Neighbor Discovery protocol to perform MAC-to-IPv6 address resolution. Unfortunately, the equivalent of inverse ARP is not available in IPv6, so the following protocol is proposed:

1) Use ARP to obtain the host's MAC address from its IPv4 address.
2) Create an Ethernet frame destined to the MAC address obtained in step 1.
3) Create an IPv6 header with destination address FF02::1 (the link-local all-nodes multicast address).
4) Create an ICMPv6 Echo request message.
5) Concatenate the headers created in steps 2, 3 and 4 and place them on the wire.
6) Listen for an ICMPv6 Echo response whose Ethernet source MAC address matches the address obtained in step 1.
7) Extract the source address from the received IPv6 header.

Note that if Stateless Address Autoconfiguration is used, steps 2 through 7 can be skipped, as the network prefix can be determined from Router Advertisement messages and the interface identifier is easily derived from the MAC address in step 1.

If the attacker is not on the same link as the host, reverse DNS lookups could be used to obtain the host name associated with the IPv4 address. Once the host name is known, the DNS system may be queried again to obtain the AAAA resource record associated with it.

There is also the case where IPv6 addresses are derived from IPv4 addresses. Examples include IPv4-compatible (*::a.b.c.d*) and IPv4-mapped addresses (*::FFFF:a.b.c.d*) where a, b, c and d correspond to the four octects of an IPv4 address [17]. In this case, there would be no need to use the DNS system since the conversion is straightforward.

C. Traceroute

Unlike ICMP for IPv4, ICMPv6 has features that are required for the operation of IPv6 and therefore, it cannot

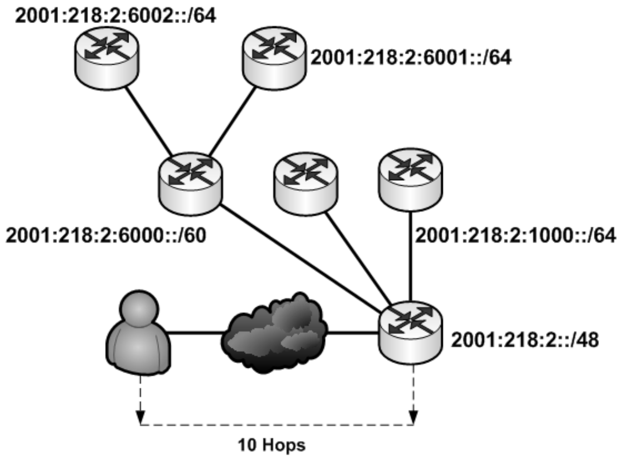

Figure 1. Example of an IPv6 network.

be completely filtered by firewalls. For this reason IPv6 networks are more likely to allow inbound and outbound ICMP messages. This can be exploited by attackers to gather information about a network. For example, assuming a /48 network prefix is known, an attacker may perform traceroute using packets with increasing values for the TTL and varying the 16-bit subnet identifier of the target address. This way, once the TTL is high enough to reach the border router but not enough to traverse the subnet routers, these subnet routers will send ICMP Time Exceeded messages, which will reveal their IP addresses. Fig. 1 shows an example of a typical IPv6 network. In that scenario, the attacker is 10 hops away from the border router. An IPv6 packet with TTL=10 destined to any host on the 2001:218:2::/48 network will be dropped by that router and an ICMPv6 Time Exceeded message will be sent to the originating address. Packets with TTL=11 will be dropped by second-level routers, and so on. Unless Time Exceeded messages are blocked by some intermediate device, if the attacker sends IP packets targeted to addresses on all 2^{16} possible subnet identifiers, he will be able to obtain the address of all routers on the network.

VI. Other Host Discovery Techniques

This section discusses techniques that allow host discovery, regardless of the position of the attacker.

A. *Using DNS to gather addresses*

The DNS system is an essential part in today's networks. Although DNS is normally used for legitimate purposes, it can be used by an attacker to discover hosts on a network [4]. For this, the attacker first needs to know the address of the primary DNS server of the network he is targeting. This is trivial if a top level domain is known as the Internet public DNS infrastructure can be used to determine the authorized servers for the zone. In cases where the attacker is on-link, DNS server addresses may be gathered in any of the following ways:

- Through DHCPv6 if the attacker's host is able to obtain configuration parameters from the local DCHPv6 server.
- Through a currently experimental protocol defined by the IETF that allows DNS recursive server advertising in ICMPv6 Router Advertising messages [18]. Attackers may listen for RA messages to learn the address of local DNS servers. This may be useful when DHCPv6 has security restrictions in place.
- Eavesdropping on the network until a DNS query is observed.

Once the address of the local DNS is known, the following techniques may be used for the discovery of active hosts.

1) DNS zone transfers: A DNS zone transfer is a type of DNS transaction that allows easy replication of DNS databases across a set of DNS servers. Zone transfers are mainly used by administrators to keep primary and secondary DNS servers synchronized. The operation is performed in the form of a a client/server transaction in which the client requests the transmission of the DNS registry through a query operation with an special query type of value AXFR. The server responds with one or more response messages that list all of the resource records for every domain name in its zone.

The DNS specification does not impose restrictions on which hosts can request and receive a full zone transfer for a domain, so in theory, any client may obtain important information about the topology of the nodes in the DNS zone, just by issuing a zone transfer request. However, the revelation of full DNS zone data is a known security issue and most DNS server implementations have mechanisms to restrict zone transfers.

2) DNSSEC zone enumeration: The Domain Name System Security Extensions (DNSSEC) is a suite of specifications that intend to improve the security of the DNS system. Essentially DNSSEC introduces authenticated DNS responses that clients can check to ensure they have not been forged. However, the original DNSSEC specification introduced a new problem. When a client requests the resolution of a name that is not registered in the DNS server, the server sends a reply that contains an NSEC resource record. This NSEC RR points to the next valid name in the zone file and is used to provide proof of non-existence of any name within a zone. This technique, commonly known as DNS *zone walking*, can be used by a malicious client to enumerate the entire DNS zone by issuing requests for non-existent names based on the names provided in the NSEC RRs. Although RFC 4470 and RFC 5155 were developed to address this issue, early implementations may be vulnerable to the attack.

3) Dictionary attacks: Even when it is not possible to perform a zone transfer operation, it is possible to conduct dictionary attacks against the target's DNS servers.

Assuming the top-level domain name is known, subdomains (and therefore IP addresses) may be discovered by actively interrogating the server for names of the form XXXX.YYYY where YYYY is the known top-level name and XXXX is a word, or a combination of words, extracted from a given dictionary.

Dictionaries may contain regular words or may be constructed a bit more sophisticatedly, taking into account names and frequency data obtained empirically.

4) Brute-force attacks: Host names are rarely created randomly but usually follow certain patterns. While in certain cases those patterns may be difficult to recognize without human intervention, in other cases simple pattern recognition algorithms may be applied to detect semantically assigned names, fixed names appended with increasing numbers, etc. Even when no pattern is recognized, the search space for DNS names is likely to be shorter than the 2^{64} possible addresses in a typical IPv6 subnet. If we consider the DNS search space of a particular zone to be given by the expression n^k where n is the the size of the allowed character set and k is the length of the unknown part of the longest record in the DNS registry, it is easy to determine the feasibility of the brute-force attack. Although in theory DNS names may contain any character that can be represented with an octet, in practice, there is a preferred form, the one permitted in the names of top-level domains, that consists of the ASCII alphabetic and numeric characters, plus the hyphen [19]. This, combined with the fact that DNS names are case insensitive, means that $n = 37$ may be enough for most cases.

B. Hosts configured by stateless autoconfiguration

IPv6 defines a mechanism that allows hosts to autoconfigure their network interfaces. This mechanism, called IPv6 Stateless Address Autoconfiguration (SLAAC), allows a host to generate its own addresses using a combination of locally available information and information advertised by routers. Routers advertise prefixes that identify the subnet associated with a link and hosts generate interface identifiers that uniquely identify an interface on the subnet. The final IPv6 address is formed by combining the two. In the absence of routers, a host only generates link-local addresses, which are sufficient for allowing communication among nodes attached to the same link [20].

End nodes derive interface identifiers from their link layer hardware addresses. In Ethernet networks, the IEEE's 64-bit Extended Unique Identifier (EUI-64) format is used. The identifier is generated from the interface's 48-bit MAC address, which is expanded to 64 bits by inserting two octets with values 0xFF and 0xFE between the Organizationally Unique Identifier (OUI) and the interface specific ID, and changing the seventh bit from the left (the universal/local bit) from a 0 to a 1.

This process significantly reduces the entropy of the host identifier part of an IPv6 address, as it adds a fixed 16-bit value and narrows the first 24 bits of the host ID to the OUIs currently assigned by the IEEE. At time of writing, there are 14,476 public OUIs assigned [21], that is, less than 2^{14}, which, providing the prefix is known, reduces the address space to 2^{38}. Although brute-forcing a 2^{38} space takes a considerable amount of time, it is certainly computationally feasible, just like it is to conduct an exhaustive search in the current IPv4 space 64 times. Moreover, a significant amount of those OUIs are not likely to be found on standard networks, as they may be assigned to companies that operate on very specific areas like satellite communications, aerospace engineering, surveillance systems, etc. Therefore, an attacker may reduce the vendor search space by sorting the list of assigned OUIs in descending order of importance for the network he is targeting, and conducting the brute-force search from the top of the list until the n-th item, where n is determined by the attacker, based on the time and resources that he is willing to invest. The list of vendors may be reduced using different criteria:

- Some corporate networks may have very homogeneous equipment. Batches of machines from the same vendor are likely to have the same Ethernet OUIs. If the attacker already knows one autoconfigured IP address on the target network, he could set the first 104 bits and brute-force the remaining 24.
- It seems reasonable to assume that the more OUIs a vendor has been assigned, the more devices it has on the market, as they are required to consume more than 90% of the available final 24 bits of MAC addresses in order to obtain a new OUI allocation [22]. Therefore, the chances of finding devices of these vendors are higher than for those with fewer OUIs. For this reason, an attacker may choose to reduce the search space to the most popular vendors, as this may increase the chances of finding active hosts. For example, discarding all vendors but the most important 10, results in a list of 1067 OUIs (approximately 2^{10}), which reduces the search space to 2^{34}. Table II lists the top-10 vendors with the highest amount of OUIs assigned.
- Data about the history of OUI assignments would allow attackers to narrow the OUI space if they had an approximate idea of when their target network's equipment was acquired.
- Passive eavesdropping of Neighbor Advertisement messages and other types of traffic may offer link-layer addresses and, consequently, vendor OUIs to use in an exhaustive search.

Once the list of OUIs is determined, the attacker has to conduct a brute-force search against the remaining 24 bits of the address for each OUI on the list. Although in theory the distribution of those bits is random, vendors are likely to assign the final 24 bits sequentially and, therefore, equipment acquired in a given country or in a big batch is likely to have identifiers whose values are

Vendor	OUIs assigned.
Cisco	492
Nokia	99
Motorola	93
Intel	70
Apple	68
Samsung	64
Hewlett Packard	53
Hon Hai	44
Texas Instruments	44
Nortel Networks	40

Table II
TOP 10 VENDORS IN THE IEEE LIST OF ASSIGNED OUIS.

closer than in a truly random distribution. For this reason, instead of sweeping the whole /24 range sequentially, it may be a better idea to sweep it randomly and, whenever an active host is found, target a range of adjacent addresses.

C. Exploiting manual address assignment

Habits are hard to change and because there is no stateless address autoconfiguration in IPv4, network administrators may manage their IPv6 networks just like they did with IPv4 ones. This means that, in certain cases, a given subnet may have all of its hosts addresses assigned manually. Administrators may chose to keep their existing addressing scheme, porting their previous IPv4 address configuration to the IPv6 address space. This way, addresses would be concentrated in a certain point of the space, leaving the rest completely empty. For this reason, attackers may choose to probe different distant parts of the available space in order to detect single hosts that belong to clusters of hosts, and then, probe a set of contiguous addresses to discover adjacent hosts. Even when no previous IPv4 network existed, for convenience, administrators may choose to configure network server addresses in the low end of the subnet range so it may be wise to start looking from there.

Let $f(k)$ be a function that checks whether the host with the k-th possible address is active, to find one host in a cluster of 2^n hosts, inside a 2^{64} address space, $f(k)$ needs to be called 2^{64-n} times (2^{64-n-1} times on average), using increasing values of k in the set $(2^n-1)g$ where $g\epsilon\{0, 1, 2..., 2^{64-n}\}$.

For example, if a given /64 subnet contains a cluster of 2^{16} hosts (equivalent to a densely populated IPv4 class B network), assuming an attacker is capable of actively probing 2^{24} hosts per second, it would take an average of $2^{64-16-1-24} = 2^{23}$ seconds, or about 97 days, to find a host that belongs to the cluster. After that host is discovered, sequential search may be used to find the rest of hosts in the cluster.

Manual address assignment may also be performed following certain semantic or syntactic rules. Some authors suggest that administrators are likely to use human readable addresses such as *2001:db8::cafe:babe:f00d* [6]. To take an example, at time of writing, Facebook uses the address *2620:0:1cfe:face:b00c::3* for their IPv6 accessible site *www.v6.facebook.com*. This behavior could be exploited, conducting dictionary attacks based on words that can be spelled with the symbols of the hexadecimal system.

D. Discovery based on local information

The purpose of self-replicating malware is to find other hosts to infect. As opposed to network scanners, which need to discover hosts in a particular network, Internet worms don't usually care about the location of hosts or the topology of the network, and therefore, any IP address is equally valuable. Once a host has been compromised, malware may gather IP addresses silently from the information that is stored on that host. There are many places where such information can be obtained. These are a few possibilities:

- The Neighbor cache is a good source of information for addresses of other hosts on the same link. The cache is organized as a set of entries about neighbors to which traffic has been sent recently. Entries contain various pieces of information about a neighbor such as its unicast IP address, its link-layer address, a flag that indicates whether it is a router or a regular host, parameters for the Neighbor Unreachability Detection algorithm, etc.
- If the local system has a DNS server running on it, its registry database can be read from disk to obtain the addresses of all registered hosts.
- Web browsers and mail user agents are also a good source of information. Although these are unlikely to provide IP addresses, bookmark files, cache entries, address books and configuration files may contain host names that can be resolved to IP addresses through standard DNS queries.
- Some authors suggest checking the SSH *known_hosts* file [5], which contains a list of known SSH server host names with their public keys. However, popular implementations like OpenSSH hash the host names before their inclusion on the list so, in some cases, it's unlikely the *known_hosts* file yields any useful information.
- The local operating system can be queried for information on active connections.
- Firewall logs are also likely to contain addresses logged as a result of significant events.
- If the system hosts traffic monitoring software like intrusion detection systems or sniffers, traffic capture files and incident logs may be processed for addresses.
- Local routing tables contain the addresses of adjacent routers.

VII. CONCLUSIONS

It is clear that the new version of the Internet Protocol introduces important challenges for the propagation of malware and the discovery of active hosts on a network. While simple techniques like random scanning or sequential ping sweeps are successfully used in today's IPv4 world, when IPv6 is completely deployed, more sophisticate methods need to be used in order to reduce the address search space and make host discovery feasible. In this paper I have presented a number of techniques that future network scanning tools and self-replicating malware may use during and after the transition to IPv6. Although those techniques allow to reduce the search space by several orders of magnitude, the discovery of network hosts will be a lot harder with IPv6 and things like complete subnet sweeps or Internet-scale scanning may never be possible again. However, the situation may change over the years. It is possible that future advances in technology, the continuous increase of devices connected to the network and the development of new protocols and standards, have a positive impact on the feasibility of network scans.

The list of techniques presented in this paper is incomplete, as there are several areas that have been left out of the scope of this research. Some examples are the operation of IPv6 in non-Ethernet networks, the protocols that provide Mobility for IPv6 or the different transition mechanisms other than the dual stack approach. Also, future research may involve experimental testing of final implementations of the IPv6 protocol family. IPv6 is a complex protocol, defined over the time through many different RFCs, and this may cause errors and misinterpretations, as well as significant differences between early implementations and newer ones.

References

[1] IANA. (2010). *IPv4 Address Space Registry.* The Internet Assigned Numbers Authority. United States. [Available on-line] http://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.txt

[2] Huston, G. (2010). *IPv4 Address Report 29-Dec-2010 07:58 UTC.* Centre for Advanced Internet Architectures. United States. [Available on-line] http://www.potaroo.net/tools/ipv4/

[3] Huitema, C. (1998). *IPv6: The New Internet Protocol.* Prentice Hall PTR. United States.

[4] Chown, T. (2008). *IPv6 Implications for Network Scanning. RFC 5157.* Network Working Group. Internet Engineering Task Force. [Available on-line] http://www.ietf.org/rfc/rfc5157.txt

[5] Bellovin, SM., Cheswick, B. and Keromytis AD. (2006). *Worm Propagation Strategies in an IPv6 Internet.* ;login: The USENIX Magazine. February 2006, Volume 31, Number 1. The USENIX Association. United States. [Available on-line] http://www.cs.columbia.edu/~smb/papers/v6worms.pdf

[6] Hogg, S. and Vyncke, E. (2009). *IPv6 Security: Protection measures for the next Internet Protocol.* Cisco Press. Indianapolis, United States.

[7] Van Hauser. (2011). *The Hacker's Choice IPv6 Attack Toolkit.* The Hacker's Choice. http://www.thc.org/thc-ipv6/

[8] Zesheng, C. and Chuanyi, J. *An Information-Theoretic View of Network-Aware Malware Attacks.* Department of Electrical and Computer Engineering, Florida International University, Miami, and School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta. United States. [Available on-line] http://users.ece.gatech.edu/~jic/infoattack.pdf

[9] Lyon, G. *The Nmap Security Scanner.* Insecure.Com LLC. [Available on-line] http://nmap.org

[10] Deering, S., Haberman, B., Jinmei, T., Nordmark, E. and Zill, B. *IPv6 Scoped Address Architecture. RFC 4007.* Network Working Group. Internet Engineering Task Force. [Available on-line] http://www.ietf.org/rfc/rfc4007.txt

[11] Arkko, J., Aura, T., Kempf, J., Mäntylä, VM., Nikander, P. and Roe, M. (2002). *Securing IPv6 Neighbor and Router Discovery.* WiSe '02, September 28, 2002, Atlanta, Georgia, USA. [Available on-line] http://portal.acm.org/citation.cfm?id=570690

[12] Narten, T., Nordmark, E., Simpson, W. and Soliman, H. (2007). *Neighbor Discovery for IP version 6 (IPv6). RFC 4861.* Network Working Group. Internet Engineering Task Force. [Available on-line] http://www.ietf.org/rfc/rfc4861.txt

[13] Hauser, V. (2008) *Attacking the IPv6 Protocol Suite.* THC, The Hacker's Choice. [Available on-line] http://freeworld.thc.org/papers/vh_thc-ipv6_attack.pdf

[14] Deering, S., Fenner, W. and Haberman, B. (1999) *Multicast Listener Discovery (MLD) for IPv6. RFC 2710.* Network Working Group. Internet Engineering Task Force. [Available on-line] http://www.ietf.org/rfc/rfc2710.txt

[15] Crawford, M. and Haberman, B. (2006). *IPv6 Node Information Queries. RFC 4620.* Network Working Group. Internet Engineering Task Force. [Available on-line] http://www.ietf.org/rfc/rfc4620.txt

[16] Conta, A. and Deerin, S. (1998) *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification. RFC 2463.* Network Working Group. Internet Engineering Task Force. [Available on-line] http://www.ietf.org/rfc/rfc2463.txt

[17] Hinden, R. and Deering, S. (2006). *IP Version 6 Addressing Architecture. RFC 4291.* Network Working Group. Internet Engineering Task Force. [Available on-line] http://www.ietf.org/rfc/rfc4291.txt

[18] Jeong, J., Park, S, Beloeil, L. and Madanapalli, S. (2007). *IPv6 Router Advertisement Option for DNS Configuration. RFC 5006.* Network Working Group. Internet Engineering Task Force. [Available on-line] http://www.ietf.org/rfc/rfc5006.txt

[19] Klensin, J. (2004). *Application Techniques for Checking and Transformation of Names. RFC 3696.* Network Working Group. Internet Engineering Task Force. [Available on-line] http://www.ietf.org/rfc/rfc3696.txt

[20] Thomson, S. and Narten T. (1998) *IPv6 Stateless Address Autoconfiguration. RFC 1971.* Network Working Group. Internet Engineering Task Force. [Available on-line] http://www.ietf.org/rfc/rfc1971.txt

[21] IEEE. (2011). *OUI Public Listing: Public OUT and COMPANY_ID Assignments.* IEEE Registration Authority, United States. [Available on-line] http://standards.ieee.org/develop/regauth/oui/oui.txt

[22] IEEE. *Guidelines for use of a 48-bit Extended Unique Identifier (EUI-48).* Institute of Electrical and Electronics Engineers, Inc. [Available on-line] http://standards.ieee.org/develop/regauth/tut/eui48.pdf