IPv6 Deployment Concerns in the Global Internet Infrastructure

Qinwen Hu
University of Auckland
qhu009@aucklanduni.ac.nz

Nevil Brownlee
University of Auckland
n.brownlee@auckland.ac.nz

ABSTRACT

In the early 1990s it became clear that the 32-bit IPv4 address space would eventually limit the Internet's growth; since then the small size of that address space has led to many scanning attacks happening in the IPv4 network. IPv6 extended the Internet address space to 128 bits. The larger address space not only supports an increased number of connected devices, but also addresses a number of security issues. In particular, a large address space makes address-scanning attacks less successful in IPv6 networks. However, more and more studies [1,10,11] explored a number of techniques that can be used for IPv6 network reconnaissance, such as: leveraging DNS reverse mappings, reducing the Interface Identifier (IID) searching space. In this paper, we analyse the feasibility of leveraging IPv6 DNS servers to launch address-scanning attacks and look at how IPv6 IIDs are assigned in practice. The results show that a reverse DNS searching strategy is less successful in the IPv6 network. In addition, our study explores a difference among datasets, and demonstrates that IID allocation differs between regions (as we expected).

Categories and Subject Descriptors

C.2.0 [**General**]: Security and protection

Keywords

IPv6; Interface Identification; address scanning attacks

1. INTRODUCTION

The main purpose for IPv6 deployment is to solve the address exhaustion problem, but also it mitigates some security issues. As we mentioned in the abstract, a larger address space makes traditional address-scanning attacks less feasible. In [1], Chown mentions that if an attacker launched one probe per second, it would take five billion years to complete an address scan in a /64 subnet. However, there are still ways in which attackers can launch IPv6 reconnaissance attacks. Many studies [2,4,8,9] noted that common patterns in its IID field can cut the search space considerably, or the DNS reverse zone can be leveraged to get IPv6 host information. This paper is an extended version of a conference paper that appeared as [5]. The key addition of this version is as follows: section 4.2 estimates whether it is feasible to launch a DNS reverse zone search in current IPv6 DNS servers.

2. DATASETS

In this study, we collected two datasets to answer different questions; the first dataset indicates how IID mechanisms are performed in the IPv6 network. We collect our data from a link connecting a UoA (University of Auckland) IPv6 network with IPv6 networks outside UoA. We observed an average 72931 IPv6 traffic flows per hour. The volume/rate of traffic flow is reasonably high between 9am-11pm at an average 105269 flows per hour; the volume/rate of traffic drops to an average 30732 flows per hour between midnight and 9am. We built a high-speed flow monitoring system which processed and recorded the first nine packets of each flow into a pcap file every hour. We collected our samples from both flow directions in the period from 2014-05-09 to 2014-08-09. The second dataset examines how feasible it is to find AAAA IPv6 host records from public DNS servers. We collected the IPv6 DNS information from [6]. In our study, we measure IPv6 DNS servers by counting active addresses, the addresses not responding the ping command were ignored. Our results are based on a large-scale surveyed of active IPv6 addresses used by active DNS servers in 127 countries around the world, over 11000 IPv6 DNS servers have been searched. We used the existing search program 'dnsrevenum6.c' to send requests to a target DNS server to dig through the AAAA records for a specified network prefix.

3. METHODOLOGY

Our objective is to investigate the IPv6 deployment concerns mentioned in Section 1. We introduce several methods to analyse our results.

3.1 Distinguishing IPv6 clients and servers

Trace files do not intuitively tell us whether an address in a packet represents a 'client' or a 'server'. Therefore, we propose several methods to classify IPv6 clients and servers. For example, we use port numbers, SYN flags, and look at DNS reverse zones.

3.2 Grouping IPv6 clients into regions

Unlike the IPv4 protocol, IPv6 has divided addresses into two parts: The upper 64-bit network prefix gives information as to a node's location; the lower 64-bit IID field contains information about how an IPv6 node has been configured. Our 'regions' are University of Auckland (UoA), and geographic regions covered by the IPv6 Deployment Aggregated Status: APNIC, ARIN and RIPE [6].

3.3 Identifying randomized IPv6 IID values

We use a frequency distribution plot to examine the discrete probability function among IID values. Each bar in the plot shows the frequency of a given IID value's occurrence. In Figure 1, the x axis repents the range $0\text{-}2^{64}$, while the y axis indicates the number of occurrences for groups of IID values. We observe that such distributions spread more or less evenly across the $0\text{-}2^{64}$ range, with no obvious gaps or significant spikes. However, there is a 1 in 2^{16} chance of a randomized IID being misclassified as an EUI-64, because of the 16 bits "ff:fe", in addition, if the universal bit has been configured, the probability of a pseudo-random IID being misclassified as EUI-64 is 1 in 2^{17} and is ignored.

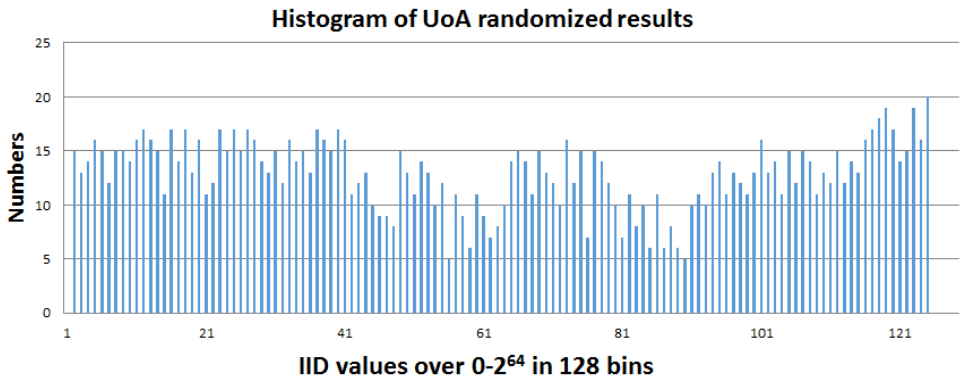

Figure 1 Histogram (Randomized IID schemes)

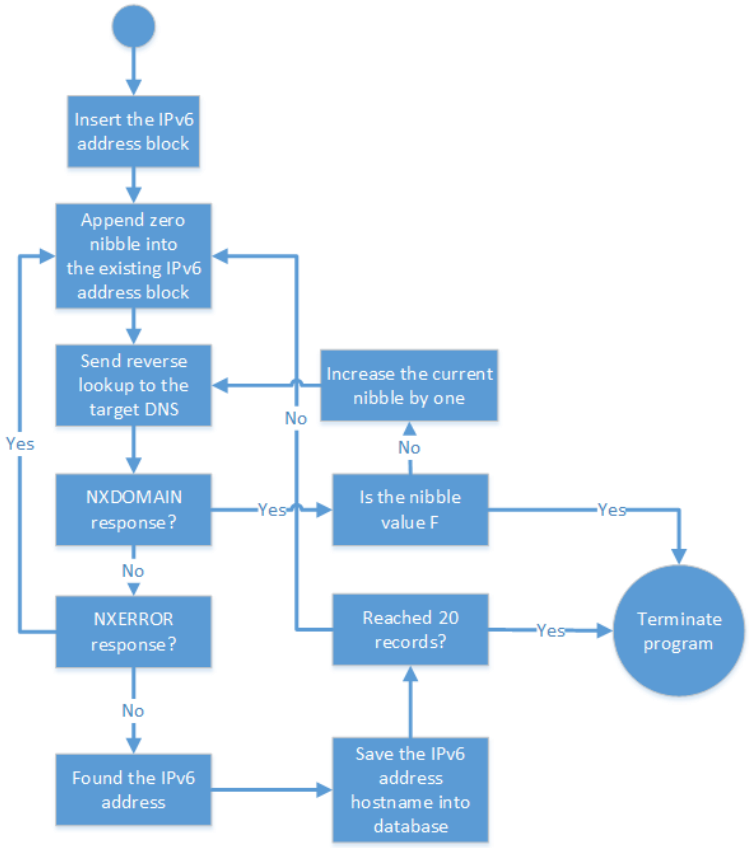

Figure 2 Activity Diagram of reverse search algorithm

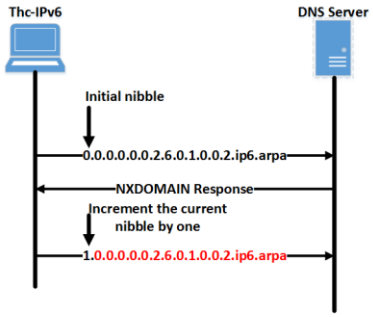

Figure 3 the sequence of process 'NXDOMAIN' responses

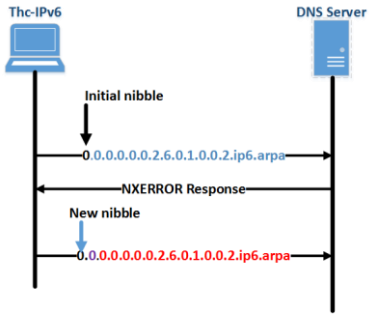

Figure 4 the sequence of process 'NXERROR' responses

3.4 Identifying IID allocation schemes

We identify IID schemes as follows. EUI-64 is generated using the EUI-64 algorithm based on each MAC address; the address can be recognized by observing the FF:FE bytes in the IID field. The Embedded-IPv4 scheme encodes an IPv4 address in the lowest-32 bits of the IPv6 address, for instance 2001:888:: 24:194:109:20:106. This scheme is easily identifiable. We verify our results by pinging those IPv4 addresses. The Small-integer scheme sets most bytes in the IID to be 0, and it can easily be recognized by checking the number of zero bytes in the IID field, for instance 2001:1318:100c:1::1. We recognize the randomized scheme based on the description in [3] and [7]. A randomized address verification method has been introduced in Section 3.3. Although other schemes exist and can be identified (e.g. service-port, wordy, etc.), they are quite rare in practice, so we put them under the label "Other".

3.5 Leveraging the DNS reverse search

The DNS reverse zone is used to map IP addresses to their corresponding host names. Van Dijk [10] discusses a search mechanism to obtain IPv6 addresses from the 'ip6.arpa' zone. In principle, Van Dijk's mechanism requires attackers to specify a network for the reverse DNS lookup queries to target and then to walk through the ip.arpa zone to search PTR records corresponding to the given domain name.

Figure 2 shows a flow diagram of the DNS reverse search algorithm. First, users need to give a network prefix of the domain from the reverse DNS zone that they want to search. When the program receives the network prefix, it adds a new nibble (all the new nibbles start with zero) and appends it to the given domain name. The program then sends a reverse lookup with this new address block to DNS servers. In principle, there are three possible responses from DNS servers:

- 'NXERROR' (RCODE 0) means this '*.ip6.arpa' domain exists in the ip6.arpa domain, but there are no PTR records for it. When the program receives this message, it adds a new nibble and appends it to the previous reverse query. The initial value of the new nibble is 0.
- 'NXDOMAIN' (RCODE 4) means there are no records for '*.ip6.arpa' in the domain name space. The program will increase its value for the current nibble and send the request again.
- If the response is the hostname, the program will save that hostname into the database.

For example, if an input address prefix is 2001:620:0::/48, Figure 3 shows the sequence of processing 'NXDOMAIN' responses. If the program receives 'NXDOMAIN' responses, the program will check the current nibble value. If the current nibble is equal to value F, the program will be terminated; otherwise, it increases the current value by one. In contrast, if an 'NXERROR' response is returned, the program will create a new nibble to add to the existing IPv6 address. This new nibble starts at zero. Figure 4 shows the process sequence of 'NXERROR' responses.

We used dnsrevenum6.c from the thc-ipv6 package, and modified the program by using a Poisson distribution with a mean time between queries of 1s so as to minimize the load on DNS servers. Also, we embed information about our experiment into every reverse lookup request, in order to explain our experiment to network administrators. Furthermore, we terminate the search program if we capture 20 hosts from the target domain.

4. OBSERVATION

4.1 How IID mechanisms are used in the IPv6 network

Table 1 provides a summary of server results in each region. The percentage in each column is the number of addresses observed in each allocation scheme divided by the total number of addresses in that region. It shows clearly that a high percentage of server addresses use specific allocation schemes (small integer).

The client results observed from our survey are similar to those in [2]. In that study 69.73% of the sample used a randomized allocation scheme and 7.72% used the EUI-64 scheme. The authors of [2] believe that network administrators should put increasing emphasis on privacy and security concerns when they allocate an IID field. However, there are still a significant number of servers IIDs using EUI-64 and embedded IPv4 strategies. Although our data are taken from a different time and location to the data in [2], our results are quite consistent with those in that study. Table 2 shows the breakdown of IID allocation schemes observed at UoA from each region. More than 30% of IPv6 addresses seen are generated using a random IID allocation mechanism. In order to make sure our results are correct, we apply the methodology described in Section 3.3; the results imply that most network administrators do care about security and privacy and therefore use hard to predict values in the IID field. The next most common technique seems to be the small-integer mechanism. Most (30%) of these addresses allocate only a few bytes to the IID field while setting most of its bytes to zero. Some EUI-64 addresses are observed: especially, UoA contributes a large proportion of the uses of the EUI-64 mechanism. After further investigation, we find that some faculties at UoA use an EUI-64 auto-configuration mechanism to generate a global IPv6 address for a new host, because this strategy can help them manage the network more easily.

Table 1 Analysis of Server IIDs

Region	EUI-64	Embedded-IPv4	Randomized	Small-integer	Other
ARIN	4%	8%	4%	80%	4%
APNIC	6%	4%	13%	77%	0%
ERPI	5%	5%	13%	69%	8%
In total	5%	6%	10%	75%	4%

Table 2 Analysis of all client addresses by region

Region	EUI-64	Embedded-IPv4	Randomized	Small-integer	Other
ARIN	7%	11.5%	38.7%	41.1%	1.7%
APNIC	17.8%	1.4%	70%	10.6%	0.02%
ERPI	8%	2%	30%	44%	16%
UoA	20%	0%	79.7%	0.3%	0
In total	12%	4%	50%	30%	2%

When we compare the results observed in [4], there is an absence of use of Teredo, ISATAP and 6to4 allocation techniques in our results, but we see a decrease in the use of embedded IPv4 addresses in the IID field, which suggests that in some areas, network administrators have changed their IID allocation strategies from transition mechanisms to other solutions.

4.2 How feasible is it to leverage the DNS reverse search in the current IPv6 network

Compared with the results we obtained two years ago [11], we observed a few changes:

All surveyed DNS servers refused to answer two types of DNS query: a query from an unknown host or a query requesting a network prefix that is different from the DNS's network prefix. For example, if we send the following command to a target DNS server: *./dnsrevenum6 2001:7d0:1:1::3 2001:500:90::/48*

the DNS server 2001:7d0:1:1::3 will send a response back with a refused tag. However, if the request is querying the DNS server's network prefix, the normal response will be sent back. If we use an authorized host to launch the dnsrevenum6 program to query a local DNS server, the normal DNS response will return. Overall, nowadays, if a DNS query is sent from an unauthorized host and the request is not querying the target DNS server's network prefix, the DNS server will reject the request. In addition, most IPv6 client records have been removed from the DNS servers; only few IPv6 server addresses have been discovered.

5. CONCLUSION

We observe that a number of considerations should be made when we allocate IIDs for a new IPv6 host. Firstly, network administrators should avoid putting predictable patterns in the IIDs, because they can be leveraged by attackers to reduce the IPv6 address search space. In addition, in order to reduce the number of security and privacy implications arising from EUI-64 identifiers, network administrators should consider generating random values for the IIDs. Although the small integer scheme is commonly used for allocating IIDs for clients and servers, it appears that the randomized IID scheme is becoming more common for allocating IPv6 client addresses. Furthermore, our results suggest that network administrators show concern regarding their DNS servers and configure the DNS server to forbid unknown hosts from walking through the DNS reverse zone.

As future work, we plan to look at how hosts are allocated within an IPv6 mobile network and to provide a detailed study of security and privacy issues in IPv6 mobile network.

6. ACKNOWLEDGMENTS

We would like to thanks Professor Brian Carpenter and Dr Ulrich Speidel for their time and effort taken to review our work.

7. REFERENCES

[1] T. Chown. IPv6 implications for network scanning. Internet RFC 5157, 2008.

[2] Tim Chown and Fernando Gont. Network Reconnaissance in IPv6 Networks, draft-ietf-opsec-ipv6-host-scanning-07, 2014.

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

[4] W. C. Shen, Y. J. Chen, Q. L. Zhang, Yang Chen, et al. Observations of IPv6 Traffic. International Colloquium on Computing, Communication, Control, and Management (CCCM 2009), August, 2009.

[5] Hu, Q., & Brownlee, N. (2014, December). How Interface ID Allocation Mechanisms are Performed in IPv6. In Proceedings of the 2014 CoNEXT Student Workshop (pp. 26-27). ACM.

[6] IPv6 Deployment Aggregated Status Available: https://www.vyncke.org/ipv6status/

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

[8] David Malone. Observations of IPv6 addresses. In Passive and Active Network Measurement, pages 21-30. Springer, 2008.

[9] Elliott Karpilovsky, Alexandre Gerber, Dan Pei, Jennifer Rexford, and Aman Shaikh. Quantifying the extent of IPv6 deployment. In Passive and Active Network Measurement, pages 13-22. Springer, 2009.

[10] P Van Dijk. Finding IPv6 hosts by efficiently mapping ip6.arpa Available:http://7bits.nl/blog/posts/finding-v6-hosts-by-efficiently-mapping-ip6-arpa

[11] Hu Q, Brownlee N. IPv6 Host Address Usage Survey. International Journal of Future Computer and Communication, 2014, 3(5): 341.