2015

Using passive and active enumeration methods to improve IPv6 host enumeration search algorithms

Clinton Carpene

Security Research Institute, Edith Cowan University

DOI: 10.4225/75/57b69ee2d9390

13th Australian Information Security Management Conference, held from the 30 November – 2 December, 2015 (pp. 87-93), Edith Cowan University Joondalup Campus, Perth, Western Australia.

USING PASSIVE AND ACTIVE ENUMERATION METHODS TO IMPROVE IPV6 HOST ENUMERATION SEARCH ALGORITHMS

Clinton Carpene
Security Research Institute, Edith Cowan University, Perth, Australia
c.carpene@ecu.edu.au

Abstract

IPv6 off-link host enumeration, when compared to IPv4 host enumeration, is a difficult and expensive exercise. The expense arises from the difference in address space sizes between the two protocols, with IPv6 having a 2^{96} larger address space than that of IPv4. This paper presents an algorithm for performing contextual IPv6 host enumeration against a target. The algorithm uses passive and active enumeration in order to focus the search upon areas of the address space where it is more probable that targets will exist. Experiments were conducted to test the proposed adaptive heuristic search algorithm involving applying the algorithm to a test dataset of IPv6 addresses, measuring the results and comparing those to a linear search against the same datasets. This research shows that the adaptive heuristic search algorithm achieved an average of 9,975 successful hits per simulation when applied to a dataset of realistic IPv6 addresses, whilst the linear search had an average of 8,642 when applied to the same dataset. Both algorithms performed poorly when applied to a dataset comprising randomly generated IPv6 addresses. The results show that the algorithm provides a good candidate for IPv6 off-link host enumeration, as it outperformed the linear search on average, whilst using less probes to do so.

Keywords
IPv6, host enumeration, IP Networks, TCPIP, Searching, Simulations

INTRODUCTION

Although IPv6 has existed for a number of years, performing off-link host discovery is still not a common practice, nor is it considered practical to do so. Host enumeration is the act of locating devices on computer networks. On-link host discovery, where the source is on the same network as the target(s) is made relatively easy with IPv6 through the use of the Neighbour Discovery Protocol (NDP), which is utilised by IPv6 for discovery purposes. Off-link host enumeration occurs in situations where the source node is on a logically separate network to the target(s), and is more difficult. Currently very few tools and methods exist to address the problem of off-link host discovery against IPv6 networks.

The complexity of the problem arises from the increased address space IPv6 offers over IPv4, which is 2^{96} times larger, and recommends a standard network size of 64 bits (2^{64} possible host addresses). To exhaustively enumerate a standard 64 bit IPv6 subnetwork, it would take 585,000 years at the rate of 1,000,000 generic probes per second (Polc´ak, 2014). By contrast a common network size in IPv4 is 24 bits (leaving 2^8 possible IP addresses for hosts), which would take less than a second to exhaustively enumerate at the same rate.

There is evidence to suggest that the means by which addresses are allocated can create patterns, which can influence new host discovery search algorithms (Atlasis, 2014). This paper presents a novel search algorithm designed to perform off-link IPv6 host discovery against 64 bit subnetworks. The algorithm differs from conventional approaches by implementing a feedback loop and decision-making systems to increase the probability of successfully probing a device. Experiments were conducted to compare the proposed proof of concept search algorithm to conventional approaches.

HOST ENUMERATION METHODS

Host enumeration methods can be classified into either active or passive enumeration. Active enumeration methods are those that involve probing target systems directly to enumerate devices. The active enumeration method category encompasses the following components:

- The target address space: The target address space is the range (or ranges) of addresses that are valid targets for the host enumeration exercise.
- The probe target(s): Not to be confused with the target address space, this refers to the address that probes will actually be delivered to. The distinction is made, since the target address may differ from

the target address space in cases that involve leveraging special addresses (such as the on-link methods using multicast addresses).

- The search algorithm: The search algorithm is the feature of the host enumeration exercise that determines the order in which targets are probed. The search algorithm can be deterministic in nature or stochastic.
- The protocol: The protocol relates to the actual protocol used to deliver the probe and receive a response. Examples of protocols for probing include ICMP Echo Requests, TCP SYN segments, ARP requests, etc.
- The probe payload: Probe payload refers to the contents of the probe. In some cases a generic payload may be used, in other cases specifically crafted payloads may be used. The payload can be used to test for specific vulnerabilities in services, or trigger specific responses from devices to ascertain whether a host is alive. Examples of payloads include randomised data or specifically crafted HTTP GET requests.

Examples of active enumeration methods include ping sweeps and SYN Scanning. Passive enumeration methods are those that do not probe target systems directly, but rather use other means to enumerate targets (Zalewski, 2005). The passive enumeration method category encompasses the following components:

- Reconnaissance subject: The subject of the passive enumeration exercise. It could be an IP address, network, domain name, email address, etc.
- Reconnaissance object: The device or system being interacted with to gain information about the reconnaissance subject
- Protocol: The protocol relates to the actual protocol that is used or examined to gather information about the reconnaissance subject. The protocol could be DNS, SNMP, HTTP, or Ethernet frames or IP(v4/v6) packets.
- Payload: The payload is the data that is either sent or examined from communications to gather information. For example, in a DNS enumeration, the payload may be a list of subdomains in a query. The response payload may be the answers from the DNS server (the A or AAAA record depending on the reconnaissance subject). In a passive network monitoring enumeration, the payload would be the data held within packets that are interesting to the observer.

As an example, DNS enumeration and passive network sniffing are passive enumeration methods.
As mentioned previously, active enumeration methods generally employ a search algorithm to determine where and how to distribute probes. For reference, on-link enumeration for IPv4 and IPv6 can be achieved by leveraging native discovery protocols such as IPv4's Address Resolution Protocol (ARP) or IPv6's Neighbour Discovery Protocol (NDP) to determine active devices that are on-link. There are few successful off-link host enumeration search methods commonly used to enumerate IPv4 devices. There are, in essence, two approaches, linear or sequential searching, and randomised searching. Linear searching, which involves testing each possible address consecutively in an address space, is commonly employed due to it's simplicity and low complexity of $O(n)$. Randomised searching offers the benefit of distributing the searching probes across the address space, which reduces load on destination networks. Additionally distributing the search load can aid in evading detection from intrusion detection systems or firewalls.

Randomised searching, despite its benefits, does increase computational complexity. The extent of the increased complexity depends on the randomisation algorithm being employed. For example, the utility *zmap* uses a computationally inexpensive randomisation function in the form of a basic linear congruential generator (LCG). The use of an LCG allows the tool to randomly search every address in the range of possible IPv4 addresses, without duplications (Durumeric et al. 2014). The application *nmap* on the other hand uses a routine to shuffle addresses in memory, which negatively impacts on the performance of the tool since additional overheads are incurred (Graham, 2012).

With optimisations, the process of enumerating the entire IPv4 address space can be completed in a few hours (Graham, 2013). As a result, it is commonplace to conduct exhaustive searches of IPv4 subnetworks or the public IPv4 Internet, where each possible address in the space is probed. The same does not hold true for IPv6, however. Attempting an exhaustive search of even a single IPv6 subnetwork is not practical given current computational and network resources. With IPv6 off-link host enumeration, efforts usually involve linearly searching a subset of the address space or performing "context searching". Approaches such as those used by The Hacker's Choice IPv6 toolkit's *alive6* (Hauser, 2015) take into account context about the IPv6 address space prior to scanning. The tool builds a static list of potential targets to probe, which is a subset of the entire space. The target list is generated by considering common usage of IPv6 in the real world, where administrators may include hexadecimal words in their IPv6 addresses, such as 'beef', 'cafe' or 'face'.

ADAPTIVE HEURISTIC SEARCH ALGORITHM

One identified issue with conventional host enumeration strategies is that active and passive enumeration methods are regarded independently. For IPv4 enumeration exercises this is of little concern, given that active enumeration has been proven to effectively identify devices. With respect to enumeration exercises involving the IPv6 protocol, there is the potential for passive enumeration to aid in reducing the scope of the problem, by identifying candidate targets with a higher probability of existing. This information, coupled with knowledge about how IPv6 addresses can be constructed, can provide the agent conducting the enumeration exercise with context by which to reduce their enumeration scope. The use of both active and passive enumeration methods has been incorporated into the development of the adaptive heuristic search algorithm.

The adaptive heuristic search algorithm can be deconstructed into four main steps:

1. Choose a destination IPv6 network or domain (e.g. example.com or 2001:db8:1:1::/64)
2. Perform passive enumeration against the target
3. Classify results and build target list
4. Probe target nodes, and repeat from 3

The proposed algorithm is depicted in Figure 1 below.

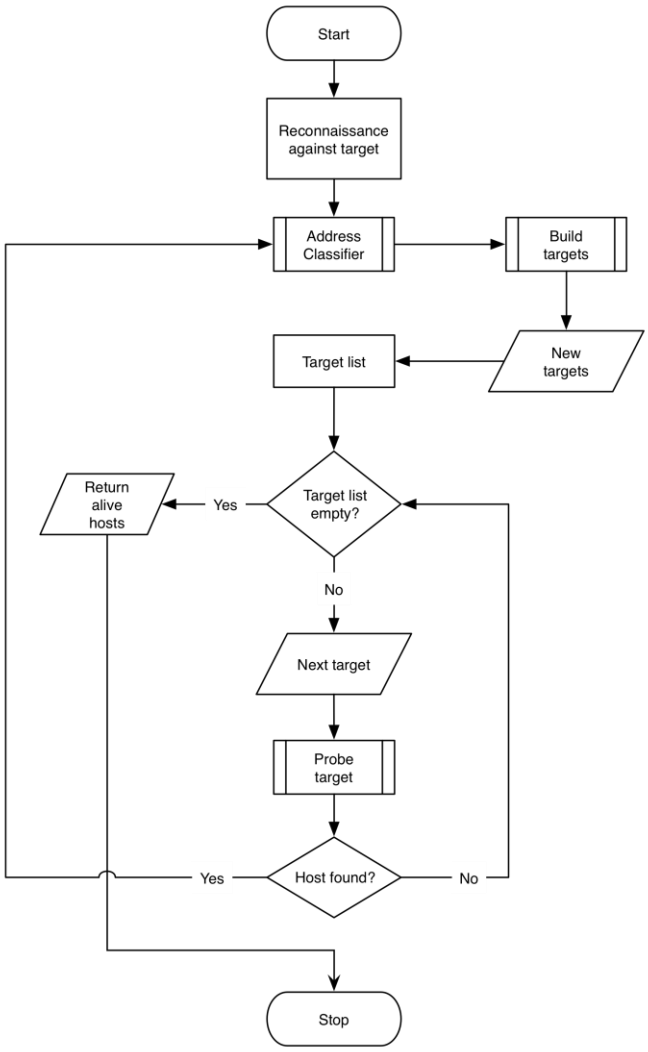

Figure 1 – Adaptive heuristic search algorithm data flow depicting the major processes involved in the algorithm.

The proposed algorithm requires some context of the target in order to conduct passive enumeration. For example its domain name, the target IPv6 network address, or the IPv6 addresses of some of the target's services. The algorithm proposed comprises a feedback loop, which uses classification to make decisions about

target nodes. The benefit of performing prior reconnaissance is so that the algorithm is given context about the target. Since the IPv6 address space is so vast, attempting to scan completely blind may never yield any results. By providing the algorithm with a starting point, it is able to assess the type of address allocations used and make appropriate decisions about the best choice of node to probe next.

Proof of concept

A proof of concept implementation of the adaptive heuristic search algorithm was designed for this study. A program was constructed that implemented the adaptive heuristic algorithm using the Python 2.7.5 programming language and the third-party libraries *pandas*, *numpy*, *scikits-learn* and *pybrain*.

Any addresses that were discovered throughout the experimentation process were classified into address construction categories. For classification a classifier based upon the ANN classifier described in Carpene, Johnstone & Woodward (2014) was used. The classifier determined whether the input address was constructed using the modified EUI-64 method (i.e. the method used during SLAAC where privacy extensions are disabled (Thomson, Narten, & Jinmei, 2007)), Incremental or manual assignment, or stochastic assignment (i.e. addresses that use Privacy Extensions, DHCPv6 random allocation, or some other random allocation scheme).

- EUI-64: For addresses that were classified as 'EUI-64' a range of addresses were constructed within the same MAC address OUI scope in such a way that $\text{start\_point} = \text{n} - \frac{\text{increment\_range}}{2}$ and $\text{end\_point} = \text{n} + \frac{\text{increment\_range}}{2}$ while $\text{end\_point} < \text{max}$, else $\text{start\_point} = 2^{64} - \text{increment\_range}$ and $\text{end\_point} = \text{max} - 1$ where n is the current target and $\text{max} = 2^{64}$.
- Incremental or manual assignment: If an address was classified as 'Incremental' then a range calculation was used such that $\text{start\_point} = \text{n} - \frac{\text{increment\_range}}{2}$ and $\text{end\_point} = \text{n} + \frac{\text{increment\_range}}{2}$ while $\text{end\_point} < \text{max}$, else $\text{start\_point} = 2^{64} - \text{increment\_range}$ and $\text{end\_point} = \text{max} - 1$ where n is the current target and $\text{max} = 2^{64}$.
- Stochastic assignment: For addresses classified as Stochastic, a binary entropy test was conducted to confirm the efficacy of the classification. If the binary entropy test failed (i.e. it returned a result less than 0.7), the classification was determined to be false and the target classified as "Unknown". On the other hand, if the test passed (i.e. the results were greater than or equal to 0.7), 65536 random targets were added to the target list for probing.

The hosts in the target list were then probed using some generic probing procedure (simulated using membership tests in this instance). If a valid reply was received (i.e. the membership test succeeded) the address was added to a list of "alive" hosts and then passed to the classification system. From there further potential targets were added and the iterative process repeated. Once the potential targets were exhausted the process quit and the results of the experiment were gathered.

The linear search algorithm was also implemented using the Python 2.7.5 programming language. This algorithm served as a baseline for standard approaches to host enumeration. The linear search algorithm designed for this research resembled the following:

1. Randomly select a start address to use as the Target
2. Probe Target
3. Increment Target by 1 (e.g. "::432" becomes "::433")
4. Repeat from 2 until maximum number of probes had been exceeded.

EXPERIMENTATION

To test the efficacy of the proposed algorithms a simulated enumeration of two target networks was performed.

Two datasets of IPv6 addresses, representing computer networks, were constructed. The first dataset, the Randomised dataset, was generated stochastically using random number generators. This dataset contained 50,000 unique IPv6 addresses. The second dataset, the Surveyed dataset, was collected from public information sources through a DNS enumeration of IPv6 records. This process was described in detail in (Carpene & Woodward, 2014). The surveyed dataset contained 41,171 unique IPv6 addresses.
The experimental process involved using computer simulations to test the subject search algorithm against the target IPv6 datasets. 100 simulations were performed involving each independent algorithm and dataset permutation. These simulations were conducted on a cluster of computers using parallel processing to execute

simultaneously. The third-party Python 2.7.5 multiprocessing library *scoop* was used to manage and control processes during the experiments.

Table 1 - Experimental parameters for the computer simulations involving the test search algorithms

Experiment Name	Dataset	Maximum transmitted probes	Number of Simulations
Adaptive-1a	Randomised IPv6 Addresses	4294967296	100
Adaptive-1b	Surveyed IPv6 Addresses	4294967296	100
Linear-1a	Randomised IPv6 Addresses	4294967296	100
Linear-1b	Surveyed IPv6 Addresses	4294967296	100

RESULTS

The results for the experiments are included in Table 2.

Table 2 - Results of the experiments involving the adaptive heuristic search and the linear search algorithms

Experiment Name	Successful probes to valid IPv6 Addresses				Number of probes delivered			
	Max	Min	Mean	Median	Max	Min	Mean	Median
Adaptive-1a	100	100	100	100	6553719	6553700	6553704.99	6553704.0
Adaptive-1b	11276	7756	9975	10004	672606286	441850500	588694674.39	589243218.5
Linear-1a	1	0	0	0	4294967296	4294967296	4294967296	4294967296
Linear-1b	16284	0	8642	16284	4294967296	4294967296	4294967296	4294967296

Adaptive Heuristic Search Algorithm

The Adaptive-1a experiment involved conducting 100 simulations of the adaptive heuristic search algorithm against the randomised dataset of IPv6 addresses. In each simulation, 100 IPv6 addresses were provided to the algorithm as the initialising data, returned from the passive enumeration component. It was observed that 0 additional valid addresses were discovered through the active enumeration phase of the process across all of the performed simulations, meaning that the search algorithm performed poorly against randomly generated IPv6 addresses. An average of 6,553,704 probes were delivered per simulation during Adaptive-1a.

The Adaptive-1b experiment saw the adaptive search algorithm tested in 100 simulations against the surveyed dataset. It was observed that on average for each simulation the search algorithm correctly identified 9,975 valid IPv6 addresses using an average of 588,694,674 probes per simulation. The maximum number of valid IPv6 addresses identified was 11,276, whilst the minimum number was 7,756.

Linear Search Algorithm

Similarly to the Adaptive-1a experiment, the Linear-1a failed to produce any meaningful results. When the linear search algorithm was applied to the randomly generated dataset of IPv6 addresses, a maximum of one IPv6 address was discovered across all 100 simulations. 4,294,967,296 probes were delivered during each simulation in Linear-1a. Again, these results imply that the linear search algorithm is not a good candidate for discovering devices on networks where devices have been configured using stochastic address construction methods.
The Linear-1b experiment observed the greatest number, as well as the lowest number of discovered valid IPv6 addresses across all of the experiments conducted. In each simulation for Linear-1b, 4,294,967,296 probes were

delivered to potential targets. The maximum number of valid IPv6 addresses discovered by the linear search algorithm was 16,284, whilst the minimum was 0. The mean number of valid IPv6 addresses discovered across all simulations in the Linear-1b experiments was 8,642.
Figure 2 shows box and whisker plots for the successful probes to valid IPv6 addresses for each experiment performed during this study.

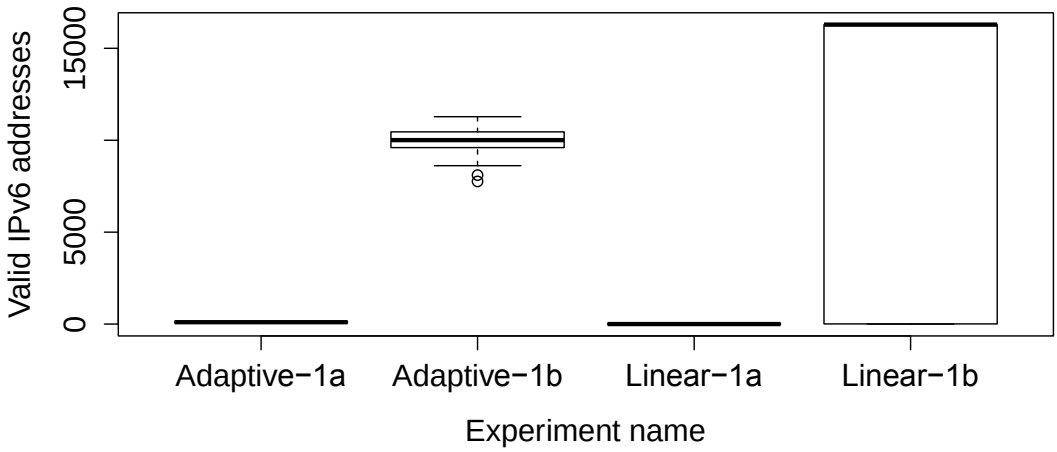

Figure 2 - Boxplots showing the number of probes delivered to valid IPv6 hosts for each simulation across each experiment.

DISCUSSION

As can be seen from the results, there is significant potential for the adaptive heuristic host enumeration algorithm. The search algorithm proved to be effective at enumerating devices on IPv6 networks where addresses were not constructed through random generation, and may therefore be applicable to real-world deployments. It was observed that the linear search algorithm enumerated more valid IPv6 addresses in some simulations than the adaptive heuristic search algorithm. Despite this, the adaptive search algorithm performed more consistently than the linear search algorithm, and had a greater mean number of detected IPv6 addresses across all of its simulations. Additionally, the adaptive heuristic search algorithm achieved these results using on average 13.70% of the number of delivered probes compared to the linear search during testing. These results highlight that the adaptive algorithm can reliably search IPv6 networks where some prior information about the subject is available.

CONCLUSION

Off-link IPv6 host enumeration is still in its infancy, with little in the way of formalised approaches on the topic. The approaches that do exist do not offer flexibility in their generation of target addresses. This paper presents a novel approach to performing the task of IPv6 host discovery and validates its effectiveness when compared against conventional approaches. The proposed algorithm was validated against the conventional approach of using linear search to enumerate a subset of the address space. It was observed that the proposed adaptive heuristic search algorithm performed more consistently than the conventional linear search algorithm for the task of enumerating off-link IPv6 networks.

FUTURE WORK

Future work will see continued development of the adaptive heuristic search algorithm. The next phase of this research is currently underway and will involve trialling the algorithm in a real world environment against live IPv6 deployments.

ACKNOWLEDGEMENTS

This research was conducted under sponsorship from Cisco. The research would not have been possible without the use of supplied Cisco networking equipment and support from Cisco.

REFERENCES

Atlasis, A. (2014, September). Chiron - an all-in-one penetration-testing framework for ipv6. In Brucon. Ghent. Retrieved from http://u.jimdo.com/www64/o/sfc6326dbff511243/download/m5145a98b92fd412c/1417096039/AAtlasis Brucon5x5 2014.pdf

Carpene, C., Johnstone, M., & Woodward, A. (2014). The effectiveness of classification algorithms on ipv6 iid construction. International Journal of Autonomous and Adaptive Communications Systems, Vol. 7(4 [IN PRESS]).

Carpene, C. & Woodward, A. (2014, December). A survey of IPv6 address usage in the public domain name system. In M. Johnstone (Ed.), The proceedings of 12th australian information security management conference (ISBN 978-0-7298-0718- 0, pp. 91–99). School of Computer & Security Science. Joondalup, WA: Edith Cowan University Security Research Institute.

Durumeric, Z., Kasten, J., Adrian, D., Halderman, J. A., Bailey, M., Li, F., ... Payer, M. et al. (2014). The matter of heartbleed. In Proceedings of the 2014 conference on internet measurement conference (pp. 475–488). ACM. Retrieved February 17, 2015, from https://jhalderm.com/pub/papers/heartbleed-imc14.pdf

Graham, R. (2012, October). Randomizing port scans, part two. Errata Security. Retrieved July 4, 2013, from http://blog.erratasec.com/2012/10/randomizing-port- scans-part-two.html

Graham, R. (2013, September). Masscan: the entire internet in 3 minutes. Errata Security Blog. Retrieved November 12, 2014, from http://blog.erratasec.com/2013/09/masscan-entire-internet-in-3-minutes.html

Hauser, V. (2015, December). THC-IPv6. Retrieved November 12, 2015, from https://www.thc.org/thc-ipv6/

Leonard, D. & Loguinov, D. (2010). Demystifying service discovery: implementing an internet-wide scanner. In Proceedings of the 10th acm sigcomm conference on internet measurement (pp. 109–122). ACM. doi:10.1145/1879141.1879156

Lyon, G. F. (2009). Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning. USA: Insecure.

Polc´ak, L. (2014). Challenges in identification in future computer networks. In Icete 2014 doctoral consortium (pp. 15–24). Wien, AT: SciTePress - Science and Technology Publications. Retrieved from http://www.fit.vutbr.cz/research/view pub. php?id=10516

Thomson, S., Narten, T. & Jinmei, T. (2007, September). IPv6 Stateless Address Auto- configuration. RFC 4862 (Draft Standard).

Zalewski, M. (2005). Silence on the wire: a field guide to passive reconnaissance and indirect attacks. San Francisco, CA, USA: No Starch Press.