Implications of IPv6 on Host Discovery and Network Scanning Detection

Mansour Alsaleh, David Barrera, P.C. van Oorschot
Carleton University

Summary

Given the 96-bit increase in IPv6 address space (128-bit addresses relative to 32-bit addresses in IPv4), finding responsive network services with random scanning seems infeasible in terms of both required resources and time. On the other hand, given that end users are allocated a minimum of 2^{64} IPv6 addresses that can be all assigned to a single host, a single compromised host seems sufficient for a scanner to evade detection. In this work, we outline some tactics for adversaries to narrow down the vast IPv6 address space to smaller subsets of a higher probability of finding addresses in use. We also illustrate the challenges in identifying remotes that use multiple IPv6 addresses and the implications this imposes on classical network scanning detection algorithms.

Is IPv6 Random Scanning Feasible?

- Even with targeting a single `/64` network, an adversary must scan the remaining 2^{64} (18 quintillion, or 18 billion billion) possible addresses:
  - ➜ To randomly scan the low-order 64 bits of the 128 bit IPv6 address, the scanner must send $2^{64} = 18$ quintillion $\times$ 54 bytes = 906 EB (exabytes).
  - ➜ For 100 MB/s upload Internet speed, 906 EB requires 2.4 million years!
- However:
  - ➜ Network administrators tend to have readable internal IP addressing schemes to quickly identify the host type (server, desktop, tablet) without relying on name resolution.
  - ➜ In most cases, servers are expected to be assigned static IPv6 addresses to reduce DNS and configuration files updates of other local hosts.
  - ➜ RIPE suggests splitting the least significant 16 bits of the network portion of the address into logical groups, e.g., **2001:db8:1234:LGBB::/64**:
    - ✯ **L** is the *location* of the device (e.g., 1-building 1, 2-building 2),
    - ✯ **G** is the *use type* of the device (e.g., 1-firewalls, 2-web servers), and
    - ✯ **B** is freely assignable for sub-partitioning (e.g., 01-Linux, 02-Windows).

IP Dictionaries

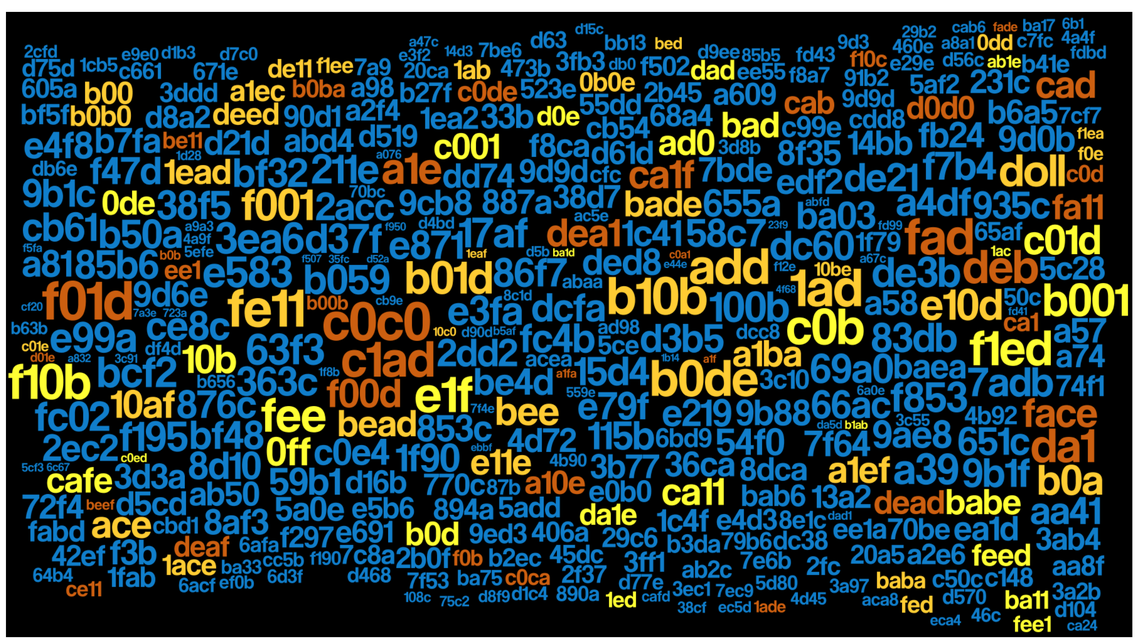
- The scanner can perform an IPv4 scan of the target network and then uses the discovered active IPv4 addresses and ports to create a list (dictionary) of candidate IPv6 addresses (one of RIPE methods for transitioning from an IPv4 addressing scheme to an IPv6 scheme); for example:

IPv4		IPv6
10.5.200.99	⟶	2001:db8:1234::5:200:99
192.168.1.113	⟶	2001:db8:1234::1:113
200.33.132.194	⟶	2001:db8:1234::194

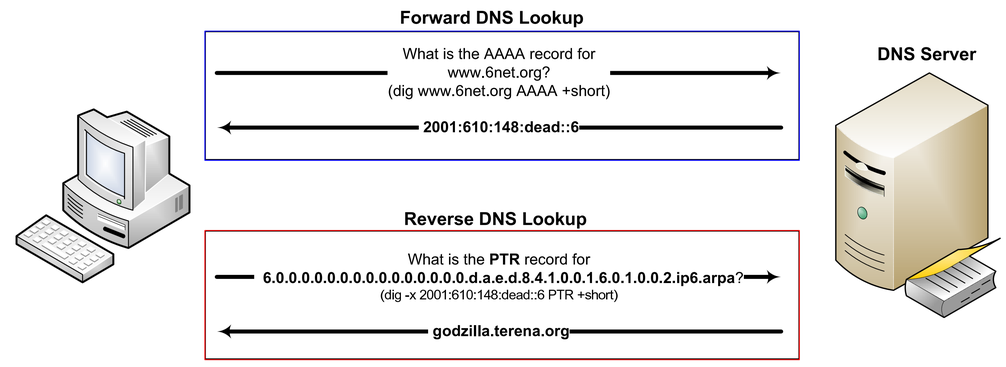
- Network administrators can choose 2-4 letter words for each hextet of the IPv6 address to create "pronounceable" addresses, for example:

Domain	IPv6 address (real)
ipv6.cplusplus.com	2607:f0d0:1301:f::c0de:c
www.leaseweb.com	2001:1af8:3100:1:0:b00b:babe:cafe
www.cyanogenmod.com	2607:f0d0:1005:42::beef:cafe
www.v6.facebook.com	2620:0:1cfe:face:b00c::b

- Transposing numbers for characters (e.g., o for 0, L for 1) can be used to create a dictionary of 107 3- and 4-character words (a dictionary that contains only hex words in the host portion of the IPv6 address would contain 131,079,601 entries (107^4).



DNS Querying



- **Reverse** (and **forward**) **DNS lookups** to gather information (e.g., address mapping, canonical name, host information, mail exchanger records).
- **DNS zone transfer** (if the DNS server is not configured properly) to gather all naming information of a specific DNS domain, including details of nonpublic internal networks.
- **Forward DNS grinding** to enumerate valid DNS address records and aliases relating to the domain and its hosts.
- **Reverse DNS sweeping** to gather details of hosts that may be protected behind a firewall but are assigned DNS hostnames.
- **Tools:** dig, traceroute, nslookup, WS-DNS-BFX, Fierce domain scan, TXDNS, and GHBA.

Querying Open Sources to Locate Active IPv6 Addresses

- Search engines to locate servers:
  - ➜ `site:.carleton.ca` (enumerate web servers under a domain).
  - ➜ `allintitle:"index of /research" site:.carleton.ca` (list web servers supporting directory indexing).
  - ➜ netcraft.com (retains historic server fingerprint details).
- Web Server and web application crawling and fuzzing tools, e.g., Wapiti and Nikto.
- Querying domain and IP registrars (e.g., WHOIS) to retrieve network blocks, routing detail, the sizes of reserved network blocks, and AS number details.
- BGP querying: cross-referencing AS numbers with BGP looking glass sites and route servers to enumerate the associated IP blocks under the AS (this information can be used to query DNS and WHOIS).
- Joining IRC, IM, and P2P networks (e.g., for topology maintenance and file sharing) to communicate with local hosts.
- SMTP probing: sending an email message to a non-existing user account reveals useful information (e.g., local DNS and network gateway IP addresses).

Discovering IPv6s From Inside a LAN

- Performing neighbour discovery (using *multicast*) in a compromised local host to locate responsive IPv6 addresses of other local hosts in the same subnet.
- Utilizing routing tables and log/configuration files of a compromised local host to determine IPv6 addresses of other local hosts in the target network.

Proposed IPv6 Scan Detection

- Networks can maintain *adaptable whitelists and blacklists* for IPv6 remotes.
- When a list grows beyond a predetermined threshold (based on available memory or storage), a *shrink()* procedure is called which groups IPv6 addresses into network blocks (e.g., `/64`, `/72`, or `/80`).
- This method has the advantage of requiring smaller amounts of memory in contrast to storing each IPv6 address individually.
- This method has the disadvantage of potentially over-blacklisting or over-whitelisting (leading to false positives and negatives) if both benign hosts and malicious hosts in the same network block use privacy extensions simultaneously.