IPv6 Security, DNS Records (AAAA), DNS Server Configurations,

IPv6 Security

IPv6 Security - Unit Overview

- IPv6 Security Myths
- IPv6 Privacy Extensions
- ICMPv6 Filtering
- Host Security
- RA Guard
- Local Network Monitoring

IPv6 Security - 3 Myths

- Myth #1: **I haven’t enabled IPv6 on my network, I don’t need to secure it.**

- Myth #2: **IPv6 isn’t vulnerable to port scans due to how large the IP space is.**

- Myth #3: **IPv6 is too new to have vulnerabilities**

IPv6 Security Myth - 1

- Myth #1: **I haven’t enabled IPv6 on my network, I don’t need to secure it.**
  - Hosts are already running IPv6 on your network
  - All modern OS’s have IPv6 enabled by default
  - Have they been secured?
  - Configuration for firewall rules and applications is typically separate
  - Ports running services will listen

IPv6 Security Myth - 2

- Myth #2: **IPv6 isn’t vulnerable to port scans due to how large the IP space is.**
  - While running `nmap -v -sn 2620:2800::/64` takes forever, there are “best guess” ways to scan and find hosts that cuts down on scan times.
  - Neighbor Discovery Protocol (NDP) can show all ‘friends’ near you based on how IPv6 works.

IPv6 Security Myth - 3

- Myth #3: **IPv6 is too new to have vulnerabilities…**
  - IPv6 has been around for 20+ years. Bugs and vulnerabilities exist and are something that needs to be watched/patched just like you have been for IPv4
  - CVE-2024-38063 - Remotely Exploiting The Kernel Via IPv6 a nice write up and code analysis.
    - Recently found. This one goes back to Windows 10 and Windows Server 2008R2 - You patched, right?

IPv6 Security - Privacy Addresses

- What are privacy addresses?
  - For networks using SLAAC, having your device use the EUI-64 method means your device can be easily tracked from network to network.
  - Defined in RFC 4941.
  - Most devices rotate IP's every 24 hours.
  - Typically the old IP is kept for another 24 hours to receive responses to old connections, but no new connections are initiated.
- Why do devices use privacy addresses?
  - For networks using SLAAC, having your device use the EUI-64 method means your device can be easily tracked from network to network.
  - Enable privacy extensions
- How do I configure this on my device?
  - Enabled by default on most modern OS's*
    - **Disclaimer: Trust, but verify. We can't keep track of all the different OS's and devices in circulation.*

IPv6 Security - ICMPv6 Filtering (WAN)

Allow:

- Type 1, <All Codes> - Destination Unreachable
- Type 2 - Packet Too Big (PMTUD)
- Type 3, Code 0 - Time Exceeded
- Type 4, Codes 1 & 2 - Parameter Problem
- Type 128 - Echo Request
- Type 129 - Echo Response
- Type 135 and Type 136 - Neighbor Solicitation and Neighbor Advertisement

Block:

- Types 5-99, 100, 101, 102-126, 154-199, 200, 201, 202- 254

IPv6 Security - ICMPv6 Filtering (LAN)

Allow:

- Type 1, <All Codes> - Destination Unreachable
- Type 2 - Packet Too Big (PMTUD)
- Type 3, Code 0 - Time Exceeded
- Type 4, Codes 1 & 2 - Parameter Problem
- Type 128 - Echo Request
- Type 129 - Echo Response
- Type 130, 131, 132, 143 - Multicast Listener Discovery (link local source address)
- Type 133 and Type 134 - Router Solicitation and Router Advertisement
- Type 135 and Type 136 - Neighbor Solicitation and Neighbor Advertisement
- Type 141 and Type 142 - Inverse Neighbor Solicitation and Advertisement

Block:

- Types 5-99, 100, 101, 102-126, 154-199, 200, 201, 202- 254

IPv6 Host Security - ip6tables Examples

- ip6tables is the IPv6 equivalent of iptables*
- Command syntax is the same
- Example:
  - ip6tables -A INPUT -d fe80::/64 -p udp -m udp --dport 546 -m state --state NEW -j ACCEPT
- **Disclaimer: Yes, IPTABLES is moving toward deprecated. Much like old servers, old habits linger.*

IPv6 Security - nftables Examples

- NFTables Families
  - ip - Used for IPv4
  - ip6 - Used for IPv6
  - inet - Support for IPv4 and IPv6. Best used for dual-stack servers/hosts
- Example:
  - # This rule affects only IPv4 packets:
    - add rule inet filter input ip saddr 1.1.1.1 counter accept
  - # This rule affects only IPv6 packets:
    - add rule inet filter input ip6 daddr fe00::2 counter accept
  - # These rules affect both IPv4 and IPv6 packets:
    - add rule inet filter input ct state established,related counter accept
    - add rule inet filter input udp dport 53 accept

IPv6 Security - UFW Examples

- Check if UFW has IPv6 enabled
  - cat /etc/default/ufw | grep IPV6
  - If “IPV6=no”, then UFW is only allowing V6 traffic on the loopback interface
  - Edit the file, change this to option to yes and then reload with “sudo ufw reload” or “sudo ufw disable-then-enable"
- Add your first rules
  - UFW doesn’t require different syntax between IPv4 and IPv6 rules. It will automatically detect and add the respective rules to the correct chain
  - “ufw allow from fe80::/64”
  - etc.

IPv6 Security - Firewalld Examples

- Firewalld zone config mostly relies on ports and services
- Rich Rules
  - rule family="ipv4" source address="x.x.x.x/24" service name="tftp" log prefix="tftp" level="info" limit value="1/m" accept
  - rule family="ipv6" source address="2001:DB8::/32" service name="tftp" log prefix="tftp" level="info" limit value="1/m" accept

IPv6 Security - RA Guard Overview

- RA messages are unsecured, how do we protect against misconfigured or rogue routers?
- RA Guard
  - Similar to STP BPDU Guard, etc.
  - Allows filtering based on the following:
    - Source MAC address
    - Source IPv6 address
    - Source IPv6 address prefix
    - Hop-count limit
    - Router preference priority
    - Managed configuration flag
    - Other configuration flag

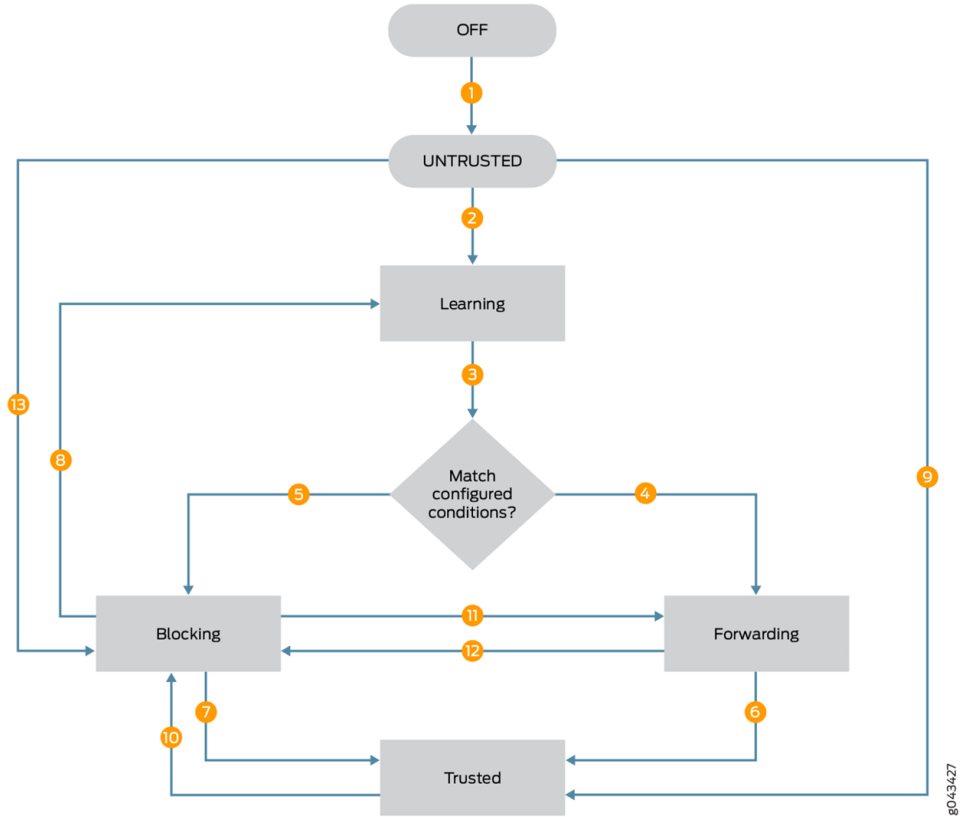
IPv6 Security - RA Guard Overview

- RA messages are unsecured, how do we protect against misconfigured or rogue routers?
- Stateless vs Stateful mode
  - Stateless mode
    - With policy: Interfaces marked as “trusted” validate the RA message against the set policy and then forward if it matches
    - Without policy: Interfaces marked as “trusted” forward RA packet without being validated against a policy
    - With/without policy: Interfaces marked as “blocked” block all RA packets
  - Stateful mode
    - Allows dynamically transitioning ports based on RA packets received during the learning process
    - Ports that receive RA messages that match the policy during the learning process transition to forwarding
    - Ports that do no receive RA messages or receive RA messages that do not match the policy during the learning process transition to blocking

IPv6 Security - RA Guard States

Off	The interface operates as if RA guard is not available.	Stateless/stateful
Untrusted	The interface forwards ingress RA messages if received RA messages are validated against the configured policy rules; otherwise, it drops the RA message. Untrusted state is the default state of an interface enabled for RA guard.	Stateless/stateful
Blocked	The interface blocks ingress RA messages.	Stateless/stateful
Forwarding	The interface forwards ingress RA messages if received RA messages are validated against the configured policy rules; otherwise, it drops the RA messages.	Stateful
Learning	The switch actively acquires information about the IPv6 routing device connected to the interface. The learning process takes place over a predefined period of time.	Stateful
Trusted	The interface forwards all RA messages directly, without validating them against the policy.	Stateless/stateful

IPv6 Security - RA Guard States (Flow Chart)



IPv6 Security - RA Guard Example (Juniper Stateless w/ Policy)

- Configure a RA guard policy:
  - set policy-options prefix-list accept-ra-address $address
  - set policy-options prefix-list accept-ra-prefix $prefix
  - set policy-options prefix-list accept-ra-mac $macAddress
- Apply the policy:
  - set forwarding-options access-security router-advertisement-guard policy accept-ra accept match-list match-criteria match-all
  - set forwarding-options access-security router-advertisement-guard policy accept-ra accept match-list source-prefix-list accept-ra-prefix
- Configure RA Guard:
  - set forwarding-options access-security router-advertisement-guard interface $interface policy $policy stateless
  - set forwarding-options access-security router-advertisement-guard vlans $vlan policy $policy stateless

IPv6 Security - RA Guard Example (Juniper Stateless w/o Policy)

- Configure RA Guard:
  - set forwarding-options access-security router-advertisement-guard interface $interface mark-interface trusted
  - set forwarding-options access-security router-advertisement-guard interface $interface mark-interface block

IPv6 Security - RA Guard Example (Juniper Stateful)

- Configure a RA Guard Policy
  - Example:
- Configure RA Guard
  - set forwarding-options access-security router-advertisement-guard interface $interface policy $policy stateful
  - set forwarding-options access-security router-advertisement-guard vlans $vlan policy $policy stateful
- Enable RA Guard
  - By default, RA guard is off. You have to request on a per interface/vlan basis.
  - Forward during learning: request access-security router-advertisement-guard-learn interface $interface duration $seconds forward
  - Block during learning: request access-security router-advertisement-guard-learn interface $interface duration $seconds block
  - Static forward: request access-security router-advertisement-guard-forward interface $interface
  - Static block: request access-security router-advertisement-guard-block interface $interface

IPv6 Security - RA Guard Example (Cisco)

- Cisco RA Guard Device Roles
  - Host - Interface or VLAN where you connect a host. This where you apply the IPV6 RA Guard policy. The device-role host allows incoming RS packets, and blocks incoming RA or RR packets. RS packets that are received on another interface, are not redirected to the device-role host. Only RA and RR packets (that are allowed) are redirected to the device-role host.
  - Switch - The device-role switch behaves similar to the device-role host. For example, you can use it as a label for a trunk port.
  - Monitor - This device monitors network traffic. It behaves similar to the device-role host, except that RS packets are also sent to this interface. This helps capture traffic.
  - Router - Interface that connects to the router. This interface allows incoming RS, RA, or RR packets.
- Config Example - Host Port
  - conf t
  - ipv6 nd raguard policy host
  - device-role host
  - exit
  - interface $interface
  - ipv6 nd raguard attach-policy host

IPv6 Security - Local Network Monitoring

- NDPMon:
  - Deprecated
- Arpwatch-V6:
  - https://github.com/acobaugh/arpwatch-v6
- Addrwatch:
  - https://github.com/fln/addrwatch

IPv6 Security - Local Network Monitoring (2)

Things To Look Out For:

- Wrong router MAC: invalid MAC address
- Wrong router IP address, invalid IP address
- Wrong prefix: invalid IPv6 prefix
- Wrong RA flags: invalid flags in the RA
- Wrong RA params: wrong parameter in the RA (lifetimes, timers...)
- Wrong router redirect: the router which emitted the redirect is not valid
- Router flag in Neighbor Advertisement: a node not declared as a router announced itself as one
- Duplicate Address Detection DOS: duplicate address detection denial of service
- Changed ethernet address: a Global IPv6 address has a new MAC address
- Unknown MAC Manufacturer: MAC vendor unknown, might be a forged one
- New node on the wire
- New IPv6 Global Address: new IPv6 Global address for a node
- New IPv6 Link Local Address: new IPv6 Link Local address for a node
- Etc.

IPv6 DNS - General

DNS

“Just a sign post to a resource.”

Domain Name System (DNS) refresher

History:

- Roots back in the ARPANET era (1969 - 1990) where a text file maintained a lookup table of numerical addresses to computer hostnames
  - /etc/hosts and host files still exist
- 1983 first form of DNS as we know it today
- 1984 - Berkeley Internet Name Domain (BIND)
- 1994 - Internet Systems Consortium (ISC) founded

Domain Name System (DNS) refresher

Purpose:

- Phone book analogy
  - Name to number and number to name.
  - Database in the sky with resilient distributed Top Level Domain (TLD)

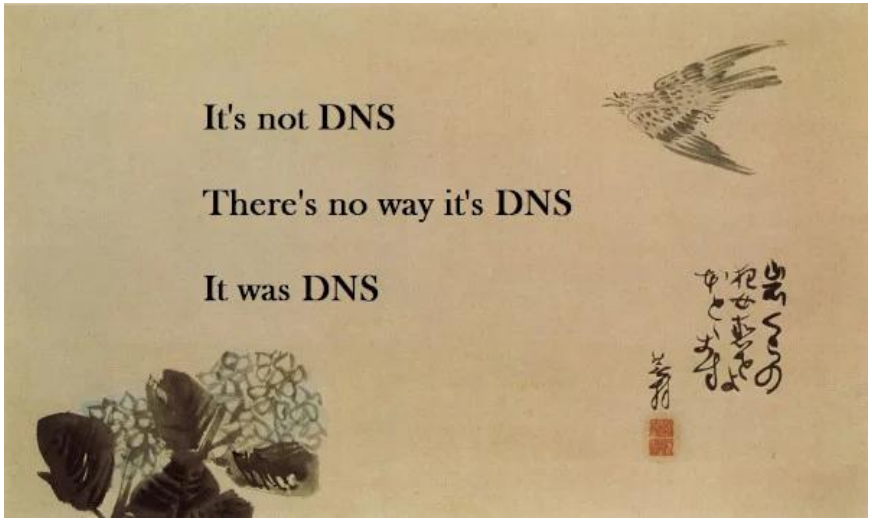
- Has adjusted organically as needs change
  - RFCs are very active in this space
  - DNS over HTTPs / ToR / TLS / and more…
  - DynDNS
  - LOC records / ssh public keys / and other “fun” distributed DB in the sky tricks.
  - DNSSEC

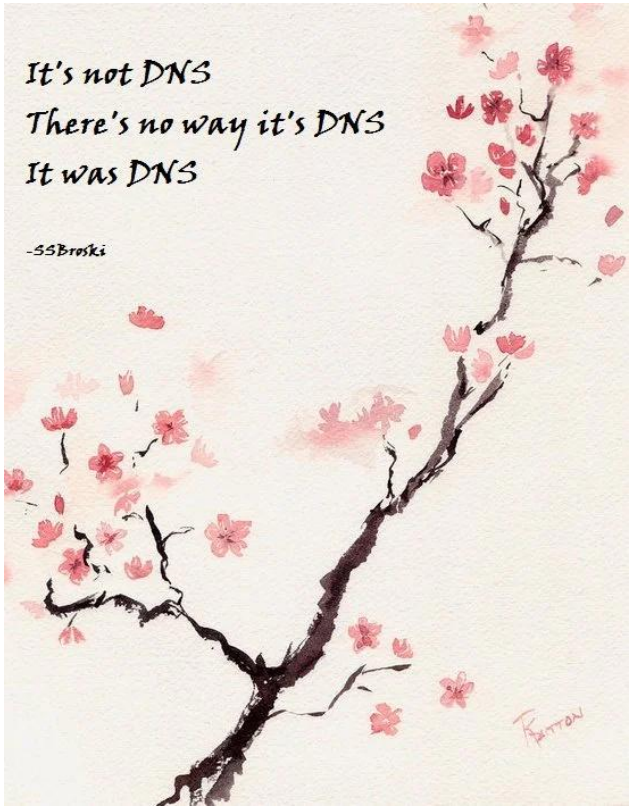
DNS

This section covers:

- Public IPv6 DNS
- BIND / Kea - perhaps with Stork
- Idas for test labs (pihole and such)

IPv6 DNS stewardship - DNS





Public IPv6 DNS services

Quick search for: “ public ipv6 DNS servers “

- Google
  - 2001:4860:4860::8888
  - 2001:4860:4860::8844
- Quad 9
  - 2620:fe::fe
  - 2620:fe::9
- Cloudflare
  - 2606:4700:4700::1111
  - 2606:4700:4700::1001

And many many more with various feature sets.

DNS - ISC products - bind9, kea DHCP (stork)

isc.org - Internet Systems Consortium as maintainers

- BIND 9 - named: Berkeley Internet Name Domain
  - Released in early 1980s as part of a DARPA grant on Berkeley Software Distribution (BSD) 4.3
  - Currently BIND can store zones in flat files (zone files) or DB backed
  - Ability to interact with DHCP (in house dynamic DNS)

BIND9 - named.conf - Dynamic DNS updates

DHCPv6 will now interact with BIND9 to provide hostname updates dynamically

- Absolutely fantastic for HPC
  - `cluster2-node33.awesome.example.com`
  - `cluster2-node34.awesome.example.com`
  - `cluster2-login4.awesome.example.com`
- Not so great when you have wi-fi clouds:
  - `iphone.awesome.example.lan`
  - `iphone.awesome.example.lan`
  - `chromecast.awesome.example.lan`
  - `….`
- Reservations can help for edge cases
- Is DNS really needed for the clients?

DNS - ISC products - bind9, kea DHCP (Stork)

isc.org - Internet Systems Consortium as maintainers

- ISC DHCP - End of Life
  - Still included in a vast array of products
  - Plan upgrade roadmap now…

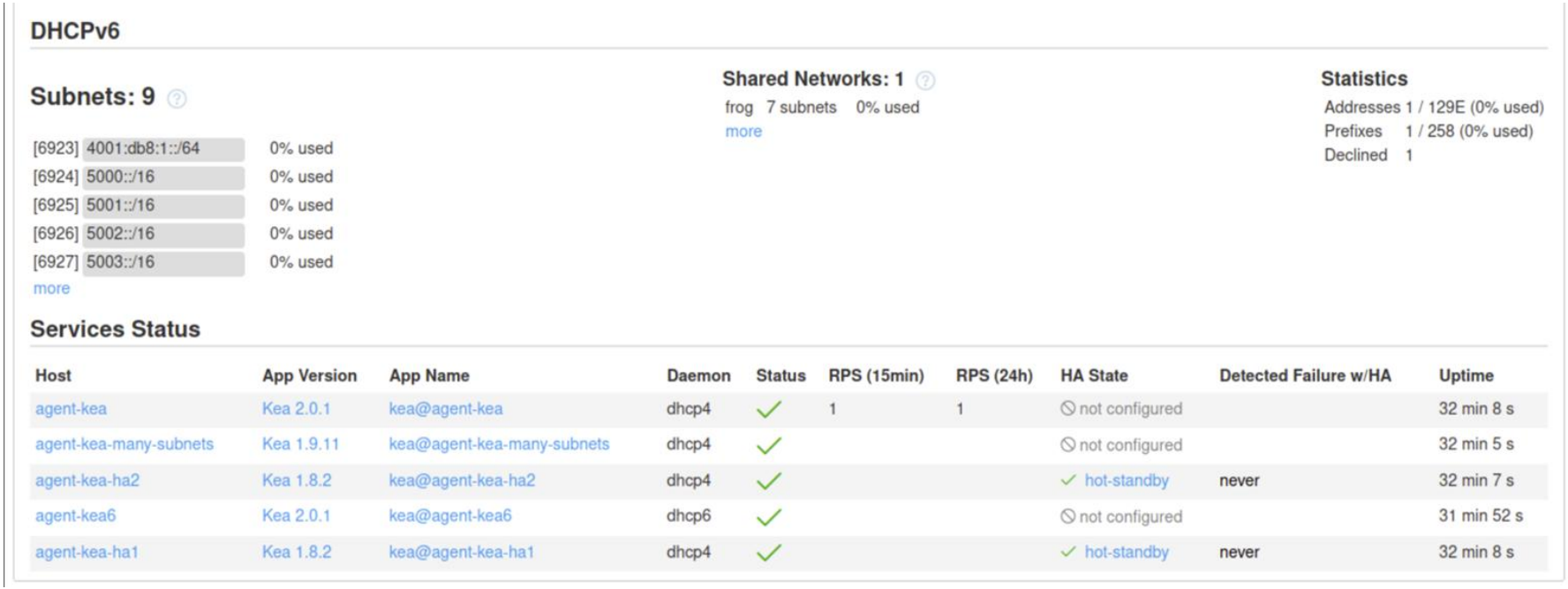
- Kea DHCP
  - IPv6 / IPv4 native
  - Modular with excellent API hooks (some require support license)
  - Hooks enable Stork passive or active web UI.
  - Excellent “import” tools

DNS - ISC products - bind9, kea DHCP (stork)

isc.org - Internet Systems Consortium as maintainers

- Stork - Graphical dashboard for monitoring and interactions with BIND and DHCP
- One stop shop for monitoring and reservations
- Can ‘point and click’ all your DHCP / BIND needs

DNS - ISC products - bind9, kea DHCP (stork)



Test setups

Take advantage of documentation space to play!

- 2001:db8::/32 AND 3fff::/20
  - Just a few addresses…
  - Your documentation is pretty and “works!”
  - Safety nets and crash mats are ‘deployed’
- You also have a built in backup protocol
  - IPv4 as a ‘back door’ for lock outs.
- What is being measured?
- What is best possible outcome?
- Have fun learning while at it!
  - Failures are an option and where learning comes in.

IPv6 AAAA Records

IPv6 DNS stewardship - DNS - AAAA records

- Fundamentals showed us that IPv6 space is ‘not’ tiny in any way shape or form with its true 128-bit sizes.
- Prevent fragmentations to be a good netizen!
  - Just like we wish to keep route-table sizes as small as possible (try to advertise contiguous blocks or your entire allocation… keep it contiguous)
  - Keep it easy, keep it elegant.
- Likewise think of the zone files!
  - Automation (CM, git-ops, DHCPv6+DDNS, etc…)
  - IPAM
  - Careful planning now vs. later… Be kind to future you!

IPv6 DNS stewardship - DNS zone files

Loose Definition: Zone files are a file that help ISC's bind9 to "lookup" DNS names to addresses (forward) or addresses to names (reverse).

Think of the security implications TO and FROM your DNS infrastructure…

IPv6 DNS stewardship - DNS zone files

Our scope today is to explore IPv6 and bind, continue to apply optimal security to your DNS infrastructure such as not limited to:

- Ingress filtering with BCP38 (stop that spoof!)
- DNSSEC (keep it real)
- Well known well tested ‘simple’ architectures (simple starts):
  - Authoritative and primary DNS servers hidden, on a patch cadence, logged, etc..
  - Recursive DNS servers (locked caches) exist as “results” of primaries, have logical and sane timeouts, live in an anycast pool, patched and logged, etc…
  - ACLs! (in BIND and on the network)
  - Who can transfer and lookup?
    - Has that hypothesis been tested.

IPv6 DNS records

Also known as quad-”A” records or AAAA

- Similar to IPv4 (just A records)
  - Can be in the same ‘zone’ files (not advised but, doable.)
- Forward and Reverse record client checks
  - `dig`
  - `host`
  - `nslookup`

A quick `host` refresher

Forward and reverse with dig asking for quad-a:

- Forward
  - `host es.net`
- Reverse (note no reverse PTR)
  - `host 2604:a880:800:c1::105:b001`
- Specific DNS server use server at the end
  - `Host es.net 2602:43:61f:ad00::53`

Discussion and Questions