NSWI184 – Řízení počítačových sítí

Přednáška osmá

Ondřej Zajíček, Kateřina Kubecová

2025-11-26

Legacy Technology: IPv4

- It is like IPv6 (almost) but addresses are 32-bit only
- Not enough addresses for every machine: needs translation mechanisms
- Addresses: 192.0.2.10 (4 bytes decimal, split by dots)
- Allocations: at least /24 prefixes, sold on free market
- Exhausted completely (except for AfriNIC)
- Reserved addresses in each network (all-zeros, all-ones)

IPv4 Special Address Ranges

- 127.0.0.0/8: loopback
- 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16: private use
- 100.64.0.0/10: shared use (CGNAT)
- 169.254.0.0/16: link-local autoconfig
- 192.0.2.0/24, 198.51.100.0/24, 203.0.113.0/24: documentation
- 198.18.0.0/15: benchmarking
- 192.0.0.0/24: IETF assignments for specific purposes
- 224.0.0.0/4: multicast
- 240.0.0.0/4: reserved for future use
- 255.255.255.255/32: local broadcast

IPv4 in routing protocols

- IPv4 addresses often used as Router IDs
- BGP: short form of NLRI
- ...

NAT: Network Address Translation

- "Short-term solution to IP address depletion" (1994, RFC 1631)
- Hiding whole networks behind few public IP addresses
- Local nodes use private range addresses for source:
  `src = 10.0.1.2:4567`, `dst = 192.0.2.42:80`
- Router translates the source to publicly routable address:
  `src = 203.0.113.75:2345`, `dst = 192.0.2.42:80`
- Router keeps the translation table to back-translate responses
- Local nodes can not be directly contacted from the Internet ... or can be?

NAT Applications

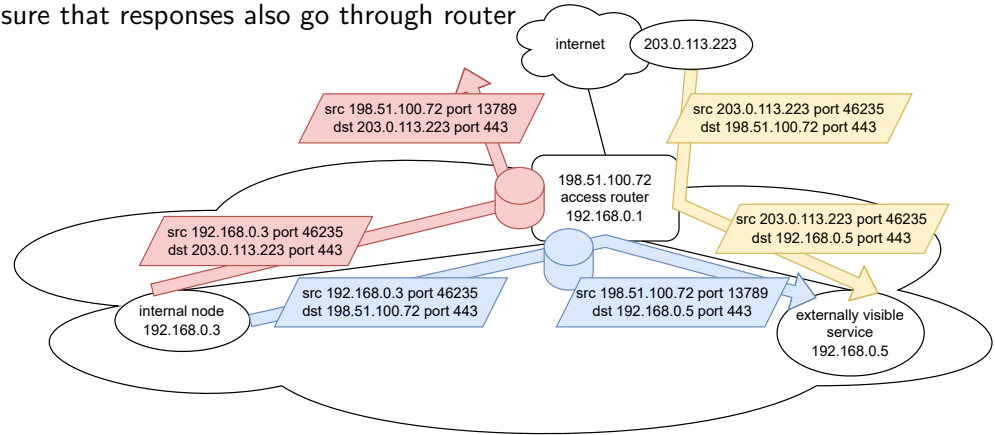
- Customer router
- Operator network (CGNAT)
- NAT44 / NAT444
- Virtualized networks in data centers

NAT Variations

- RFC 3489, RFC 4787
- NAT vs NAPT
- Static vs dynamic mapping
- Endpoint-independent mapping
- Address-dependent mapping
- Address and port-dependent mapping

NAT Hairpinning

- Communication between internal nodes through external addresses
- Router must translate both source and destination
- Ensure that responses also go through router



NAT in Linux

- Connection tracking based on (proto, SA, SP, DA, DP)
- Prefers keeping port unchanged when free
- `/proc/net/nf_conntrack`
- `/proc/sys/net/nf_conntrack_max`

NAT Limitations

- End-to-end connectivity: Breaks direct peer-to-peer communication
- Protocol issues: Problems with protocols embedding IP addresses
- State tracking: Performance and scaling issues of connection tracking
- State persistence: Breaks long-term connections
- Single point of failure: NAT device becomes critical bottleneck
- Layer 4 dependent: Does not work well outside TCP and UDP

NAT Advantages

- Fits whole internet into limited IPv4 address space
- Decouples private network from public addressing
- Allows easy switch to backup ISP
- Hides internal network structure
- Prevents easy access from public Internet (but firewall is better)

IPv6 Transition Mechanisms

- Dual stack
- Translation-based mechanisms
- Tunneling-based mechanisms

Dual Stack

- IPv4 and IPv6 simultaneously
- Advantages:
  - Simple deployment
  - Full compatibility with both protocols
- Disadvantages:
  - Duplication of work
  - Hides fails in IPv6 deployment

IPv4/IPv6 Translation

- RFC 6144
- Stateless translation: SIIT (RFC 6145/7915)
- Stateful translation: NAT64 (RFC 6146)
- DNS translation DNS64 (RFC 6147)
- Synthesis: 464XLAT (RFC 6877)

Stateless IP/ICMP Translation

- RFC 7915
- Stateless translation between IPv4 and IPv6
- IPv6-only network connected to IPv4 internet
- Bidirectional communication
- Uses algorithmic address mapping
- Embeds IPv4 address into IPv6 address
- Incompatible with SLAAC

NAT64

- RFC 6146
- Stateful translation between IPv4 and IPv6
- Translation mechanism between IPv6 and IPv4 networks
- Enables IPv6-only clients to access IPv4-only services
- Similar to NAT44 with endpoint-independent mapping
- Well-Known Prefix: 64:ff9b::/96 (RFC 6052)

DNS64

- RFC 6147
- How IPv6-only hosts get IPv6 addresses of IPv4 servers?
- Translation done in DNS name servers with DNS64
- Plenty of problems (DNSSEC, DoH, out-of-control resolvers)
- Only works when server referenced by DNS name

464XLAT

- RFC 6877
- Combines stateful NAT64 and stateless NAT46 translation
- Enables IPv4 connections over IPv6-only networks
- Removes need for DNS64
- Widely used in mobile networks transitioning to IPv6
- Supported by mobile phones

464XLAT Architecture

- CLAT (Customer-side Translator)
  - Stateless NAT46 translator on client device
  - Converts IPv4 packets to IPv6 for transport
  - Local IPv4 address from 192.0.0.0/29
- PLAT (Provider-side Translator)
  - Stateful NAT64 translator in the network edge
  - Handles IPv6 to IPv4 translation for internet
- Pref64 extension in router advertisements (RFC 8781)

IPv6 Tunneling Mechanisms

- Connecting IPv6 networks over IPv4 internet
- 6in4: Manual tunnel configuration
- 6to4: Automatic tunneling over IPv4 internet
- ISATAP: Intra-site automatic tunnel addressing
- Teredo: NAT traversal for IPv6 over IPv4

Dual-Stack Lite

- IPv4 connectivity over IPv6-only networks
- Combines tunneling with CGNAT
- Dual stack in customer network, IPv6-only operator network
- CPE router encapsulates IPv4 traffic and sends it to AFTR
- AFTR decapsulates traffic and does NAT44
- Deployed by ISPs

Lightweight 4over6

- Variant of DS-lite
- Moves NAT from operator network to CPE router
- Subset of ports is provisioned to customer
- CPE router does NAT44 to that subset of ports
- Encapsulated packets are forwarded to tunnel concentrator
- Tunnel concentrator is stateless

Routing IPv4 with IPv6 nexthops

- Next-hop is only needed to get link address
- IPv4 route can have IPv6 next hop
- Support in BGP and Babel
- Nodes do not need IPv4 addresses if they do not originate traffic
- But they *do* originate traffic: ICMP
- Special ICMP origin address: 192.0.0.8

Mutually suspicious

EXTERIOR GATEWAY PROTOCOL (EGP)

Eric C. Rosen
Bolt Beranek and Newman Inc.

October 1982

It is proposed to establish a standard for Gateway to Gateway procedures that allow the Gateways to be mutually suspicious. This document is a DRAFT for that standard. Your comments are strongly encouraged.

Receiving BGP routes

- It's my responsibility to filter out garbage
- Martians:
  - My own routes
  - Private ranges
  - IXP ranges
  - Non-public-routable ranges ... e.g. `::/3`, `4000::/2`, `8000::/1`
- IRRdb, RPKI/ROA: check origin ASN
- IRRdb, ASPA: check relations in AS Path
- Too long prefixes (received /64 is probably an OSPF leak)
- Too short prefixes (we are in the Default-Free-Zone)
- Too many prefixes (from a customer / peer) $\rightarrow$ set a limit
- Well-known transit ASNs should not be seen in routes from peers and customers
- Bogus ASNs: private ranges, IXP ASNs

Receiving data: Firewall!

- Source address should make some sense.
- Destination address should match routes which I sent.
- Rate limiting.
- Do. Not. Block. ICMP. Please.
- My destinations: Reject unwanted on the ingress.
- Propagate longest possible prefixes to avoid local drops.

Firewall at Linux: nftables

```
table inet meow {
  chain weird-source {
    type filter hook input priority filter policy accept;
    ip saddr { ::/3 } drop;
  }
}
```

Security inside your network

- *Any node can go rogue.*
- Drop obviously spoofed packets
- If something should be local, enforce local
- Drop (and alert?) unwanted routing protocols
- Monitoring (do have some)

Reverse Path Filtering

- Suppose that the sender expects answer.
- Drop packets where we have no route for sender.
- Drop incoming packets on interfaces where we have no route for sender.
- Strict RPF: Packets may only come from where the best route points to.