NSWI090: **Computer Networks**

http://www.ksi.mff.cuni.cz/~svoboda/courses/202-NSWI090/

Lecture 10

Addresses and Addressing II

Martin Svoboda
svoboda@ksi.mff.cuni.cz

10. 5. 2021

Charles University, Faculty of Mathematics and Physics

Lecture Outline

Addresses and addressing

- L3
  - **IPv4 addresses**
    - Subnetting and supernetting
    - CIDR
    - Registries
    - Private addresses and NAT
  - **IPv6 addresses**

IPv4 Address Exhaustion

Temporary **mitigating solutions**

- 1985: **Subnetting**
  - One larger network is divided into separate sub-networks
- 1988: Allocation mechanism
  - **One larger block** $\rightarrow$ **more smaller blocks** principle
- 1993: **CIDR** (**Classless Inter-Domain Routing**)
  - Original concept of IP address classes is entirely dropped
- 1994: **Private addresses**
  - Usage of private IPv4 addresses instead of globally unique ones
  - Requires **NAT** (**Network Address Translation**)

Permanent solution

- 1995: IPv6 protocol and its **IPv6 addresses**
  - **6 bytes** instead of 4 bytes $\Rightarrow$ significantly **larger address space**

Subnetting

Motivation

- **One closest larger block allocation principle** is used
- $\Rightarrow$ inner block address space may not be used efficiently
  - Since **unused addresses cannot be used by anyone else**
    - This in fact led to unacceptable wasting
- Objective
  - **Higher utilization of addresses** within allocated blocks

Principle

- **Division of larger blocks into smaller ones**
  - In terms of decomposition of networks into subnetworks

Subnetting

Subnetting

- Standard network is **internally divided into** **subnetworks**
  - Standard means Class A, B, or C block
- **Divide position is shifted to the right** (toward lower bits)
  - By one or more bits as needed
- $\Rightarrow$ **divide position** must be somehow remembered
  - Since the traditional class boundaries will no longer work
  - And we still must be able to recognize IP address parts
    - Which will now be impossible without extra information
- **Netmask** (**subnet mask**) was proposed for this purpose
  - Written as an ordinary IP address
    - Contains **bits** 1 in the intended **network part**, **bits** 0 elsewhere
  - E.g.: 255.255.0.0 as an equivalent of Class B network

Subnetting: Example

Assume we have a Class C **network** 195.113.19.0

- Permits ≈ 256 addresses, netmask would be 255.255.255.0

It can be divided into the following **subnetworks**

- Subnetwork 195.113.19.**0** with netmask 255.255.255.**128**
  - I.e., 195.113.19.**0**0000000$_B$, netmask 255.255.255.**1**0000000$_B$
  - **Divide shifted by +1**, allows ≈ 128 individual addresses
- Subnetwork 195.113.19.**128** with netmask 255.255.255.**192**
  - I.e., 195.113.19.**10**000000$_B$, netmask 255.255.255.**11**000000$_B$
  - **Divide shifted by +2**, allows ≈ 64 individual addresses
- Subnetwork 195.113.19.**192** with netmask 255.255.255.**192**
  - I.e., 195.113.19.**11**000000$_B$, netmask 255.255.255.**11**000000$_B$
  - **Divide shifted by +2**, allows ≈ 64 individual addresses

Subnetting

Observations

- Both **routers and end nodes** must support the whole concept
  - Which is not a big deal since…
    - We are the owners of the **infrastructure** within the network
    - **End nodes** can easily be adapted via software updates
- Subnetting is always **limited to a given standard network** only
  - Its **impact <u>must not be visible from outside</u>**
    - I.e., network as a whole still must act as an atomic unit
    - And so global routing and forwarding are not impacted at all

Supernetting

Motivation

- More closest smaller blocks **allocation principle** is used
- $\Rightarrow$ size of **routing tables** increases
  - Since **each individual network** must have its **own record**
  - Routing tables therefore became unacceptably large
    - As well as routing tables **lookup slowed down**
- Objective
  - **Reducing overall size of routing tables** in backbone routers
    - And so with no impact on depletion of IP addresses themselves

Principle

- **Aggregation of smaller blocks into larger ones**
  - In terms of records in routing tables

Supernetting

Supernetting (**Aggregation**)

- Several **adjacent aligned blocks are merged together**
  - They must share the same prefix of their network IDs
  - Entire address space defined by this prefix must be covered
  - All the original blocks must have the same routing direction
- **Divide position is shifted to the left** (toward higher bits)
- $\Rightarrow$ once again, **netmasks** are needed

Observations

- Supernetting is **entirely transparent**
  - Contrary to subnetting…
  - And so all **routers** within the system must support the concept

Supernetting: Example

Assume we have the following individual Class C **networks**

- Or just one network with the following address blocks…
  - Target network 195.113.**16**.0 (i.e., 195.113.000100**00**$_B$.0)
  - Target network 195.113.**17**.0 (i.e., 195.113.000100**01**$_B$.0)
  - Target network 195.113.**18**.0 (i.e., 195.113.000100**10**$_B$.0)
  - Target network 195.113.**19**.0 (i.e., 195.113.000100**11**$_B$.0)
- They all have the same routing direction

Their **routing records** can thus be grouped together

- Target network 195.113.**16**.0 with netmask 255.255.**252**.0
  - I.e., 195.113.**000100**00$_B$.0, netmask 255.255.**111111**00$_B$.0
  - **Divide shifted by -2**

Regional Registries

Original **arrangement**

- **Entire address space** was managed by IANA
  - I.e., individual blocks were **directly** assigned to **end users**
    - In terms of **Class A, B, or C blocks**
- Involved **agenda** became far too **extensive and demanding**
  - $\Rightarrow$ individual regional registries were gradually founded
    - And related agenda correspondingly transferred

Regional Internet Registry (**RIR**)

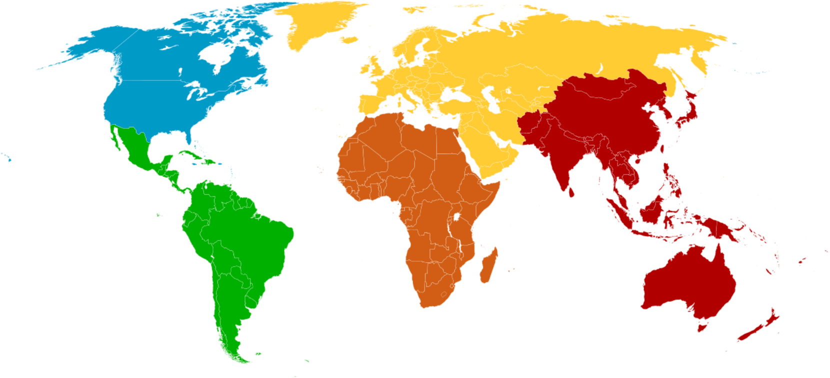
- **Organization** managing **allocation and registration** of Internet resources within a given region
  - **IP addresses** and **Autonomous System Numbers**
- **5 individual RIRs** around the world exist nowadays
  - Each obtains larger blocks of IP addresses from IANA

Regional Registries

Regional registries

- 1992: **RIPE NCC** (**Réseaux IP Européens Network Coordination Centre**)
  - Europe, Central Asia, Russia, West Asia
- 1993: **APNIC** (**Asia Pacific Network Information Centre**)
  - South, East, and Southeast Asia, Oceania
- 1997: **ARIN** (**American Registry for Internet Numbers**)
  - USA, Canada, Antarctica, …
  - Operating in fact since 1991
- 1999: **LACNIC** (**Latin America and Caribbean Network Information Centre**)
  - Latin America, Caribbean
- 2004: **AFRINIC** (**African Network Information Centre**)
  - Africa

Regional Registries



RIPE NCC — RIPE NETWORK COORDINATION CENTRE

APNIC

ARIN — American Registry for Internet Numbers

lacnic

AFRINIC — The Internet Numbers Registry for Africa

CIDR

Motivation

- **Allocation of address blocks is still not flexible enough**
  - Because of **coarse granularity** of possible block sizes
  - Especially for networks of **mid-sized organizations**
    - Class C with $\approx$ 256 addresses is too small
    - Class B with $\approx$ 66 thousand addresses is too large
- **Subnetting** and **supernetting** both helped…
  - But **transparent** solution is needed so that the **entire address space** can be exploited efficiently enough (not just parts of it)
- Objective
  - **Hierarchical allocation of address blocks with arbitrary sizes**
    - So that block sizes can better match projected needs
- Principle: **fully classless routing mechanism**

CIDR

CIDR (**Classless Inter-Domain Routing**)

- **Concept of classes** is now definitely abandoned
  - Except for Classes D (multicast addresses) and E (future use)
    - Their meaning and ranges were preserved untouched
    - Including the meaning of other special addresses
- **Divide can now be placed anywhere**
  - I.e., leading bits no longer determine anything
  - And so **divide position** must once again be explicitly declared
    - Though analogous to netmasks…
    - … different and more convenient notation was introduced
- CIDR Prefix (or simply **prefix**)
  - **Number of bits** forming the network part
    - Written as a decimal number after the slash symbol at the end
  - E.g.: 172.217.0.0/16 as an equivalent of former Class B network

CIDR: Example

Assume we have an **allocated block** 195.113.19.0/24

- It can internally be divided into the following networks
  - Network A: 195.113.19.**0**/**25** (i.e.,195.113.19.**0**0000000$_B$/**25**)
  - Network B: 195.113.19.**128**/**26** (i.e.,195.113.19.**10**000000$_B$/**26**)
  - Network C: 195.113.19.**192**/**26** (i.e.,195.113.19.**11**000000$_B$/**26**)

Detailed **routing information**…

- Can remain **entirely undisclosed**
  - 195.113.19.**0**/**24** for all our networks
- As well as intentionally **fully or partially exposed** if needed
  - 195.113.19.**0**/**25** for network A
  - 195.113.19.**128**/**25** for aggregated networks B and C

CIDR

Observations

- CIDR is **deployed globally** and **fully transparent**
  - I.e., its scope is not limited just to a particular internetwork
    - As was the case of subnetting alone
- Meaning of **former classes** can be preserved
  - Class A / B / C blocks correspond to CIDR prefixes 8 / 16 / 24
    - By the way, individual addresses have CIDR prefix 32
- However, they can also be **transparently decomposed**…
  - E.g., 172.217.23.0/24 is a CIDR block with prefix 24
    - I.e., it provides $\approx$ 256 individual addresses
    - As if it was just an ordinary Class C block
    - But it is not, since it is just a part of a former Class B block
    - Such a thing would not be possible without CIDR
- As a consequence, **entire address space is treated uniformly**

CIDR

Observations (cont'd)

- Ideas of both **subnetting and supernetting** are supported
  - **Larger blocks can be divided into smaller ones**
    - In terms of decomposition of networks
    - So that the **address space** can be utilized more efficiently
    - Because block sizes can be chosen with **finest granularity**
  - **Smaller blocks can be aggregated into larger ones**
    - In terms of grouping of routing records
    - So that size of **routing tables** can hopefully be reduced
    - And so detailed routing information can remain localized
    - Without needing it to be disseminated globally
- Allows for **hierarchical assignment of address blocks**
  - And so the whole hierarchy of registries
    - Which also helps with the growing **agenda**

Hierarchy of Registries

Hierarchy levels

- CIR (Central **Internet Registry**) = IANA
- RIR (Regional **Internet Registry**)
  - RIPE NCC, APNIC, ARIN as well as later on LACNIC and AFRINIC
  - Later on liaised through NRO (**Number Resource Organization**)
    - Informal body coordinating matters of global importance
- Optional NIR (National **Internet Registry**)
  - National allocaters in **larger countries** only
    - **APNIC** region: China, India, Japan, Korea, Indonesia, …
    - **LACNIC** region: Brazil, Mexico
- LIR (Local **Internet Registry**)
  - **ISPs**, larger enterprises, or **academic institutions**
  - Membership in a given RIR / NIR is required

Allocation Mechanisms

Example: end node **195.113.19.170**

- **IANA**
  - 195.0.0.0/8 ($\approx$ 17 million addresses) $\rightarrow$ **RIPE NCC**
- RIR: **RIPE NCC**
  - 195.113.0.0/16 ($\approx$ 66 thousand addresses) $\rightarrow$ **Cesnet**
    - Autonomous System AS2852
  - 195.113.0.0/18 ($\approx$ 16 thousand) $\rightarrow$ **Charles University**
    - Publicly invisible as for routing records
- Internal invisible decomposition
  - …
  - 195.113.18.0/23 ($\approx$ 512 individual addresses)
- Target **end node**
  - 195.113.19.170/32 $\rightarrow$ nosql.ms.mff.cuni.cz

Allocation Mechanisms

Allocation process

- **IANA**
  - Delegates **/8 blocks** to individual RIRs
    - Certain blocks are assigned to particular organizations directly
    - E.g.: US Postal Service, US Department of Defense, …
  - **Online database**
    - https://www.iana.org/assignments/ipv4-address-space/
    - Contains only records for **/8 blocks**
- **RIRs / NIRs**
  - Delegate parts of allocated blocks to subordinated LIRs
- **LIRs**
  - Assign smaller blocks to **end users**
    - Often **singleton addresses** only
  - Consequence: **addresses became dependent** on particular LIRs

Allocation Mechanisms

IANA top level **database** for /8 blocks

- **Types** of records
  - **Allocated**
    - Delegated entirely to a specific RIR or other organization
  - **Legacy**
    - Formerly allocated by IANA prior to the foundation of RIRs
    - Later on transferred to and administered by individual RIRs
  - **Reserved**
    - Designated for **specific purposes** (e.g., loopback, …)
  - **Unallocated**
    - Not yet allocated or reserved and so **available** for assignment

Allocation Mechanisms

Current situation (May 2021)

- **Distribution of /8 blocks** between individual RIRs
  - ARIN (93), APNIC (50), RIPE NCC (40), LACNIC (10), AFRINIC (6)
  - Class D multicast addresses (16), Class E future use (16), …
- **Overall allocation of /8 blocks**
  - 2011: IANA delegated the very last available blocks
    - One to every individual of all 5 RIRs
  - $\Rightarrow$ there is **no longer any unallocated /8 block**
- Situation in **RIPE NCC**
  - 2019: the very last block from the pool was allocated
  - Only **recovered addresses** via a **waiting list** are now available
    - I.e., blocks returned by former LIR holders
    - Currently $\approx$ 320 thousand individual addresses available

Private Addresses

Motivation

- Each node must have a **globally unique IP address**
  - I.e., address distinct within the whole system of networks
    - Otherwise routing will not work
  - **Number of available addresses is still decreasing**, though
    - Despite all the other already discussed mitigating measures
- Idea
  - **Nodes in a private network** can use **private addresses** instead
  - These **private addresses will then be translated** to public ones
    - In order to ensure they do not leave a given private network
- Two basic translation mechanisms are available
  - **NAT** (**Network Address Translation**)
  - **L7 Gateways**

Private Addresses

Observations

- **Any range of addresses could theoretically be used**
  - However, it is not desirable and correct
  - In particular, when such addresses would (even accidentally) **leak out from a given private network**…
    - It will not be possible to remedy the situation later on
    - Simply because other routers will not be able to detect them
- Therefore **dedicated addresses** should be used
  - 1 Class A block: **10**.X.Y.Z
    - I.e., **10**.0.0.0/**8**
  - 16 Class B blocks: **172.16**.X.Y – **172.31**.X.Y
    - I.e., **172.16**.0.0/**12**
  - 256 Class C blocks: **192.168.0**.X – **192.168.255**.X
    - I.e., **192.168**.0.0/**16**

Network Address Translation

Network Address Translation (NAT)

- Generic translation mechanism
  - Allows not just to spare public IP addresses
    - Probably the most successful mitigating solution
- Deployment
  - **Inner private and outer public networks**
  - Separated by a router implementing the NAT mechanism
- Disadvantages
  - **Decreases the overall throughput**
  - **May not always work**
    - Since the translation primarily works at L3 / L4 layers
    - Therefore it is incapable of modifying L7 data which may also contain the same addresses otherwise subjected to translation

NAT Principle

Delivery mechanism

- **Outgoing** transmission
  - (1) **inner node sends an IP datagram** in a standard way
    - Source address is set to private IP_S address of a given node
    - Destination address is IP_T address of the intended recipient
  - (2) this **datagram is captured by the router** and **NAT is applied**
    - Source address is replaced with appropriate public IP_P address
  - (3) **modified datagram is then sent to the public network**
    - As if its sender was actually the router itself
- **Incoming** transmission
  - (4) response from the target recipient is delivered back to IP_P
  - (5) this response is captured by our router and translated
    - Destination address is replaced with the original IP_S
  - (6) modified response is internally sent to our node

Static and Dynamic NAT

Static NAT

- Mapping table is **fixed and given in advance**
  - Each node will always be mapped to the same public address
    - And so individual bindings are predictable in advance
- Inner nodes are always **reachable from outside**
  - In terms of incoming transmissions initiated from outside
    - I.e., not in terms of responses to our outgoing transmissions
    - Since these are always deliverable
- Disadvantage
  - **Sizes of both public and private blocks must be identical**
    - And so there is no saving effect
  - At least unless certain inner nodes shall not communicate at all

Static and Dynamic NAT

Dynamic **NAT**

- Records in mapping tables are added / removed **dynamically**
  - I.e., individual **bindings are created on demand**
    - Only when they are really needed (new outgoing connection)
    - And will only exist for a limited period of time
  - **Assigned address may always be different** for a given node
    - And so it cannot be determined in advance
- Inner nodes are **not (automatically) reachable from outside**
  - **Unless their bindings already exist**
  - This may be treated as a disadvantage
    - As well as on the contrary…
    - Since NAT can therefore act as a kind of **firewall**
- **Not all inner nodes must necessarily have their bindings**
  - And so we can really save public addresses

Port Address Translation

Network Address and Port Translation (**NAPT**)

- Also abbreviated just as **Port Address Translation** (**PAT**)
- **Motivation**
  - Suitable when we do not have as many public addresses as nodes in our private network
    - This often means we may only have just a **single public address**
- **Principle**
  - All inner nodes are mapped to the same public address
  - Individual transmissions are **distinguished via different ports**

PAT Principle

Delivery mechanism

- **Outgoing** transmission
  - (1) **inner node sends an IP datagram** in a standard way
    - Source transport private address is IP_S:$port_S$
    - Destination transport address of recipient is IP_T:$port_T$
  - (2) this **datagram is captured by the router** and **PAT is applied**
    - Source address is replaced with appropriate public IP_P:$port_P$
    - Datagram TCP / UDP payload is also accordingly translated
  - (3) **modified datagram is then sent to the public network**
    - As if its sender was actually the router itself
- **Incoming** transmission
  - Steps (4), (5), and (6) for response delivery are analogous…

PAT Observations

Dynamic character

- Created **bindings** always exist only for a **limited period of time**
  - And so responses can only be delivered during this window
- Depends on a **particular** L4 **protocol** and its **implementation**
  - **UDP**: 30 – 300 seconds
  - **TCP**: 30 – 60 minutes

PAT alternatives

- How the assigned public IP_P:$port_P$ address is to be resolved?
  - It can depend solely on the source private IP_S:$port_S$ address
    - **Full** / **IP Restricted** / **Port Restricted Cone NAT**
  - As well as even on the intended destination IP_T:$port_T$ address
    - **Symmetric NAT**
- Incoming transmissions from which nodes will be accepted?

PAT Alternatives

Full Cone NAT

- Assigned public IP_P:$port_P$ depends…
  - Solely on private IP_S:$port_S$ address
    - I.e., assigned public address remains the same even for further connections to possibly different intended destinations
- **Responses** to IP_P:$port_P$ are accepted from…
  - **All IP addresses and all their ports** without any limitation

IP Restricted Cone NAT (or just Restricted Cone NAT)

- Assignment rules are the same
- **Responses** to IP_P:$port_P$ are accepted only from…
  - **Contacted IP addresses and still all their ports**

PAT Alternatives

Port Restricted Cone NAT

- Assignment rules are the same once again
- **Responses** to IP_P:$port_P$ are accepted only from…
  - **Contacted IP addresses and only from contacted ports**

Symmetric NAT

- Assigned public IP_P:$port_P$ depends…
  - Both on private IP_S:$port_S$ and destination IP_T:$port_T$ addresses
    - I.e., assigned public address (in particular its port) changes every time a different destination is requested
- **Responses** to IP_P:$port_P$ are accepted only from a given single…
  - **Contacted IP address and its only contacted port**

IPv6 Addresses

Motivation

- IPv4 exhaustion problem
  - **Temporary mitigating measures** worked
    - And actually worked better than it was perhaps anticipated
    - Since the first exhaustion threat appeared around 1990
    - And IPv4 is still the primary approach even after 30 years
  - Nevertheless, **permanent solution was needed**
    - Formerly intended **Class E** was not found realistically applicable
    - Since IPv4 would need to be entirely redesigned
    - Together with dozens of other related protocols (OSPF, RIP, …)
  - And so an entirely new solution was introduced
- **IPv6 protocol and addresses**
  - Primarily **larger address space**
  - But also several other **major improvements** and changes

IPv6 Addresses

IPv6 addresses

- Differences to IPv4
  - **More address hierarchy levels**
    - Allows for better aggregation
    - And so smaller and localized routing tables
  - **Easier assignment of addresses**
    - Including autoconfiguration options
    - Network interface can have several unicast addresses at a time
  - **Introduction of anycast addresses**
    - And removal of broadcast addresses
  - **Significantly larger address space**
    - 128 bits (16 bytes) instead of just 32 bits (4 bytes)
    - Theoretically $\approx 3.4 \times 10^{38}$ individual addresses
    - Compared to just $\approx 4.3 \times 10^{9}$ in case of IPv4

IPv6 Addresses

IPv6 addresses

- **Notation**
  - Eight 2-byte-long words, written as 4-digit hexadecimal numbers, mutually separated by colons
  - E.g.: 805b:2d9d:dc28:0000:0000:fc57:d4c8:1fff
- **Abbreviations**
  - **Suppressed leading zeros**
    - Leading zeros in individual words are truncated
    - E.g.: 805b:2d9d:dc28:**0**:**0**:fc57:d4c8:1fff
  - **Zero-compressed**
    - One adjacent group of zero words is entirely omitted
    - E.g.: 805b:2d9d:dc28**::**fc57:d4c8:1fff
  - **Mixed notation** for **embedded IPv4 addresses**
    - E.g.: 0:0:0:0:0:0:**212.200.31.255** or **::212.200.31.255**

IPv6 Addresses

Types of addresses

- **Unicast addresses**
  - Allow for standard communication with **individual nodes**
- **Multicast addresses**
  - Allow for communication with **all nodes in a given group**
    - Always start with ff/8 (11111111_B…)
  - Can be used to simulate traditional **broadcasts**, too
- **Anycast addresses**
  - Allow for communication with **one node from a given group**

IPv6 Addresses

Types of **unicast** addresses

- **Global Unicast**
  - Globally unique public individual addresses
- **Local Unicast** (**Unique Local Addresses**) (**ULA**)
  - Private addresses unique within all subnets of a given site
    - Labeled with such site and subnet identifiers
    - Therefore suppressed chances of accidental leak outs
  - Always start with fc00/7 (1111110_B…)
- **Link Local**
  - Private addresses unique within a given individual subnet
    - Allow for **autoconfiguration** based on MAC addresses
  - Always start with fe80/10 (1111111010_B…)
- **Site Local**
  - Private addresses unique within all subnets of a given site
    - Should no longer be used

IPv6 Addresses

Global individual addresses

- Three components
  - **Site identifier**
    - Identifies a particular site = group of related networks
  - **Subnet identifier**
    - Identifies a particular network within a given site
  - **Interface identifier**
    - Identifies a particular node within a given subnet
- **Routing** mechanisms
  - **Public topology**
    - Routing algorithms in public Internet only work with sites
  - **Site topology**
    - Internal site topology is concealed to the public Internet

Lecture Conclusion

L3: **IPv4** addresses

- **Subnetting** and **supernetting**
- **CIDR** blocks and prefixes
- RIR, NIR, and LIR **registries**
- **Private addresses** and **NAT / PAT** translation

L3: **IPv6** addresses

- **Site, subnetwork, and interface** parts
- Types of addresses