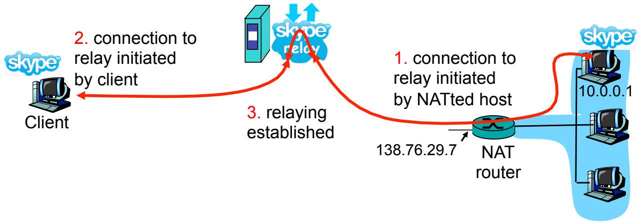
Data Relay

- relaying (used in Skype)
  - NATed client establishes connection to relay
  - External client connects to relay
  - relay bridges packets between to connections
  - Traversal using Relay NAT (TURN) as IETF draft



Frameworks

- Interactive Connectivity Establishment (ICE)
  - IETF draft
  - mainly developed for VoIP
  - signaling messages embedded in SIP/SDP
- All possible endpoints are collected and exchanged during call setup
  - local addresses
  - STUN determined
  - TURN determined
- All endpoints are „paired“ and tested (via STUN)
  - best one is determined and used for VoIP session
- Advantages
  - high sucess rate
  - integrated in application
- Drawbacks
  - overhead
  - latency dependent on number of endpoints (pairing)

Recap

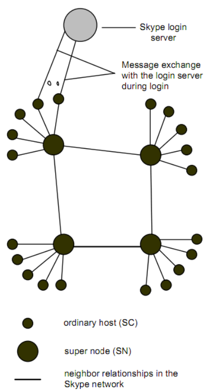
- NAT behavior
  - Binding
    - Port and NAT
  - Filtering
    - Endpoint independent vs. dependent
- NAT Traversal Problem
  - Realm specific IP addresses in the payload
  - P2P services
  - Bundled Session Applications
  - Unsupported protocol
- NAT Traversal techniques
  - Behavior based vs. active support by the NAT/ext. entities

Skype

- Closed source P2P VoIP and IM Client
- Many techniques to make reverse engineering difficult
  - Code obfuscation
  - Payload obfuscation
- Known to work in most environment
- Extensive use of NAT Traversal techniques
  - STUN
  - Hole Punching
  - Relaying
  - UPnP
  - Port Prediction

Skype components

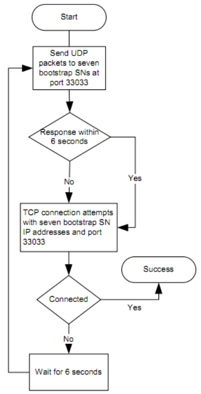
- Ordinary host (OH)
  - A Skype client (SC)
- Super nodes (SN)
  - a Skype client
  - Has public IP address
  - sufficient bandwidth
  - CPU and memory
- Login server
  - Stores Skype id' s, passwords, and buddy lists
  - Used at login for authentication



http://www.cs.columbia.edu/~salman/publications/skype1_4.pdf

„Join“ process

- Tasks performed
  - User authentication
  - Presence advertisement
  - Determine the type of NAT
  - Discover other Skype nodes
  - Check availability of latest software
- Needs to connect to at least one SN
  - SNs used for signaling
  - Host Cache holds ~200 SNs
  - 7 Skype bootstrap SN as last resort

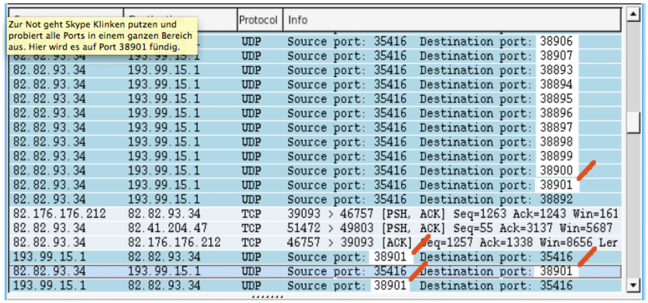


http://www.cs.columbia.edu/~salman/publications/skype1_4.pdf

NAT Traversal

- Ports
  - Randomly chosen (configurable) TCP and UDP port for the Skype client
  - Additionally: listen at port 80 and 443 if possible
    - If you become a SN (outgoing connections to 80/443 are usually possible)
- Skype SNs used as Rendezvous Points
  - SN acts as STUN like server to determine external mappings
  - Signaling and exchange of public endpoints for HP
  - Used as relays if necessary
  - Otherwise, no centralized NAT helper

Hole Punching in Skype

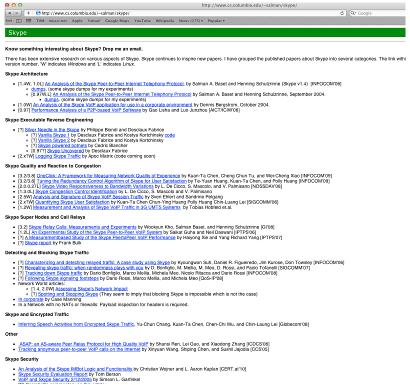


		Protocol	Info
[illegible]	[illegible]	UDP	Source port: 35416 Destination port: 38906
[illegible]	[illegible]	UDP	Source port: 35416 Destination port: 38907
82.82.93.34	193.99.15.1	UDP	Source port: 35416 Destination port: 38893
82.82.93.34	193.99.15.1	UDP	Source port: 35416 Destination port: 38894
82.82.93.34	193.99.15.1	UDP	Source port: 35416 Destination port: 38895
82.82.93.34	193.99.15.1	UDP	Source port: 35416 Destination port: 38896
82.82.93.34	193.99.15.1	UDP	Source port: 35416 Destination port: 38897
82.82.93.34	193.99.15.1	UDP	Source port: 35416 Destination port: 38898
82.82.93.34	193.99.15.1	UDP	Source port: 35416 Destination port: 38899
82.82.93.34	193.99.15.1	UDP	Source port: 35416 Destination port: 38900
82.82.93.34	193.99.15.1	UDP	Source port: 35416 Destination port: 38901
82.82.93.34	193.99.15.1	UDP	Source port: 35416 Destination port: 38892
82.176.176.212	82.82.93.34	TCP	39093 > 46757 [PSH, ACK] Seq=1263 Ack=1243 Win=161
82.82.93.34	82.41.204.47	TCP	51472 > 49803 [PSH, ACK] Seq=55 Ack=3137 Win=5687
82.82.93.34	82.176.176.212	TCP	46757 > 39093 [ACK] Seq=1257 Ack=1338 Win=8656 Ler
193.99.15.1	82.82.93.34	UDP	Source port: 38901 Destination port: 35416
82.82.93.34	193.99.15.1	UDP	Source port: 35416 Destination port: 38901
193.99.15.1	82.82.93.34	UDP	Source port: 38901 Destination port: 35416

http://www.heise.de/security/artikel/Klinken-putzen-271494.html

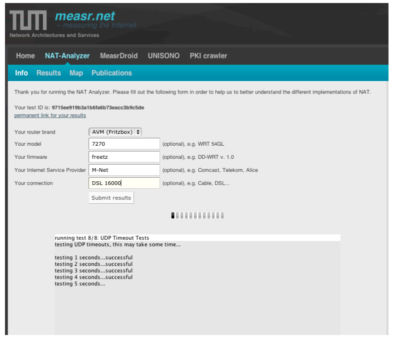
More on Skype

- http://www.cs.columbia.edu/~salman/skype/



NAT Analyzer - Overview

- Public field test with more than 2000 NATs
  - understand existing traversal techniques and NAT behavior
    (http://nattest.net.in.tum.de)

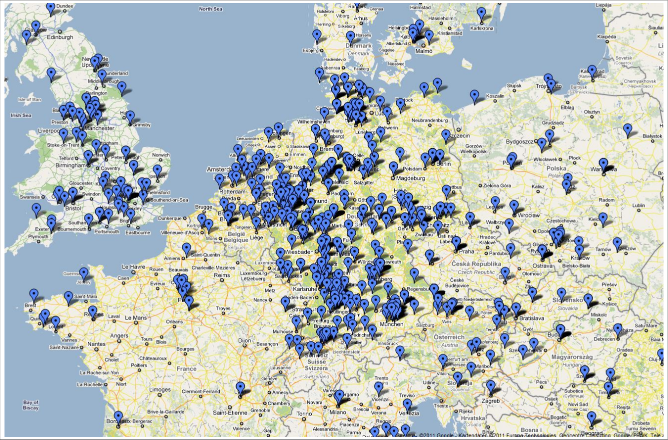


NAT Analyzer

- Connectivity tests with a server at TUM
  - NAT Type
  - Mapping strategy
  - Binding Strategy
  - Hole Punching behavior using different techniques
  - Timeouts
  - ALGs
- Example Result

NAT Tester – Results (World)

NAT Tester – Results (Central Europe)



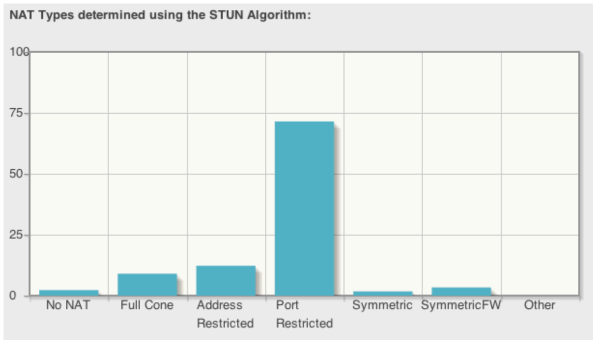
NAT Tester – Results (Providers)

Deutsche Telekom	186
Alice	49
Comcast (US)	47
Arcor	40
Freenet	40
SBS (US)	34
Kabel Deutschland	25
Virgin Media (GB)	23
China Telecom (CN)	20
Road Runner (CA)	18

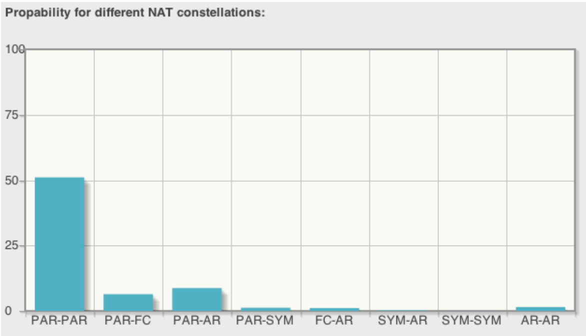
NAT Tester – Results (Findings)

- Ranking NAT Router
  - Others 30%
  - Linksys 16%
  - Netgear 10%
  - AVM 7 %
  - D-Link 7%
  - Dt. Telekom 6%
- Symmetric „NATs“
  - China
  - Iran
  - Malaysia
  - Israel

NAT Types



NAT Constellations

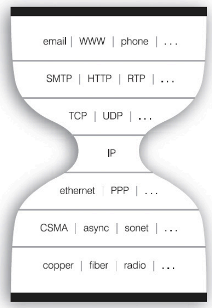


Success Rates for existing traversal solutions

- UPnP 31 %
- Hole Punching
  - UDP 80%
  - TCP low TTL 42%
  - TCP high TTL 35%
- Relay 100%
- Propabilities for a direct connection
  - UDP Traversal: 85 %
  - TCP Traversal: 82 %
  - TCP inclusive tunneling: 95 %

The problem is becoming even worse

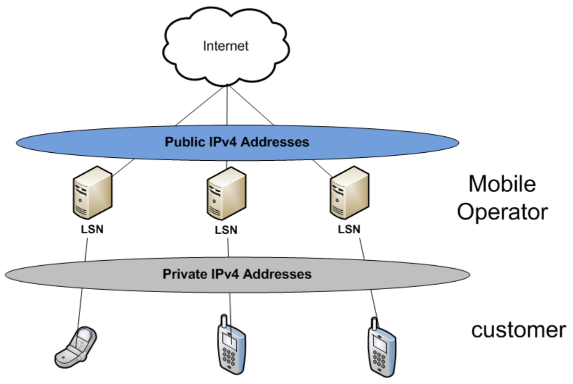
- **More and more devices connect to the Internet**
  - PCs
  - Cell phones
  - Internet radios
  - TVs
  - Home appliances
  - Future: sensors, cars...
- With NAT, every NAT router needs an IPv4 address
- → ISPs run out of global IPv4 addresses



Large Scale NAT (LSN)

- Facts
  - ISPs run out of global IPv4 addresses
  - Many hosts are IPv4 only
  - Not all content in the web is (and will be) accessible via IPv6
    - infact: < 5% of the Top 100 Websites (09/2011)
- Challenges for ISPs
  - access provisioning for new customers
  - allow customers to use their IPv4 only devices/CPEs
  - provide access to IPv4 content
- Approach: move public IPv4 addresses from customer to provider
- Large Scale NAT (LSN) / Carrier Grade NAT (CGN) at provider for translating addresses

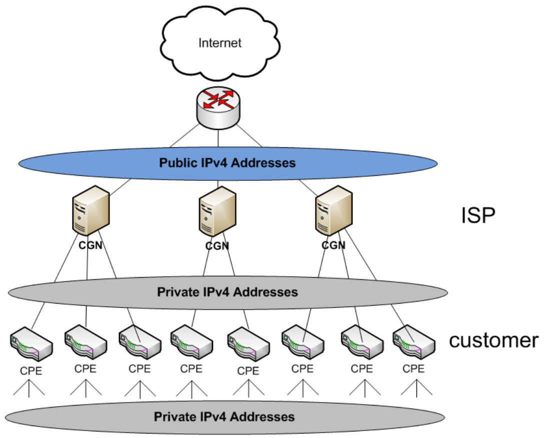
Large Scale NAT already common today



NAT Analyzer – Results (Mobile Operators)

- Germany
  - T-Mobile, Germany
  - Vodafone, Germany
  - O2 Germany
  - E-Plus, Germany
- Europe
  - Hutchison 3G, Ireland
  - Vodafone, Spain
  - Panafone (Vodafone) Greece
  - Eurotel, Czech
  - Tele2 SWIPnet, Sweden
  - Hutchison Drei, Austria
- World
  - Cingular, USA
  - Kyivstar GSM, Ukraine

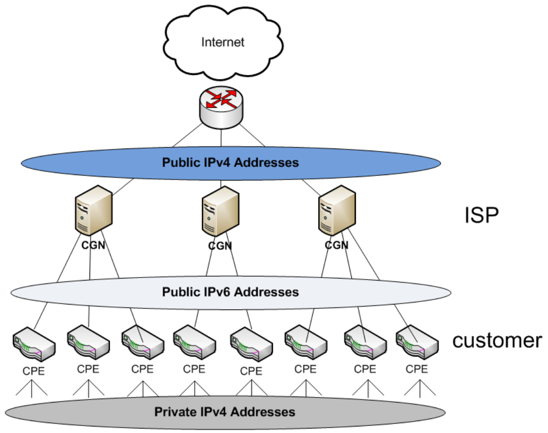
NAT 444



NAT 444

- Easiest way to support new customers
  - immediately available
  - no changes at CPEs (Customer Premises Equipment)
- Problems:
  - Address overlap -> same private IP address on both sides
  - Hairpinning necessary: firewalls on CPE may block incoming packets with a private source address
- Solutions
  - declare a range of public IP addresses as „ISP shared“ and reuse it as addresses between CGN and CPE
  - NAT 464: IPv6 between CPE and CGN
    - Problem: CPEs must implement NAT64

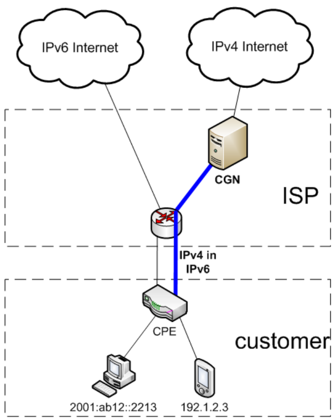
NAT 464



Dual Stack lite

- Mixture of NAT 444 and NAT 464
- IPv4 in IPv6 tunnel between CPE and ISP
  - No need for protocol translation
  - No cascaded NATs
- Allows to deploy IPv6 in the ISP network while still supporting IPv4 content and IPv4 customers
  - As IPv6 devices become available they can be directly connected without the need for a tunnel
- Mainly pushed by Comcast (in IETF)

Dual Stack Lite



LSN - Challenges

- Mainly: how to manage resources
  - Ports (number of ports, allocation limit (time))
  - Addresses
  - Bandwidth
  - legal issues (logging)
- NAT behavior
  - desired: first packet reserves a bin for the customer -> less logging effort
  - IP address pooling: random vs. paired (same ext IP for internal host)
    - Pairing between external and internal IP address
- Impacts of double NAT for users
  - Blacklisting as done today (based on IPs) will be a problem
  - No control of ISP NATs
- Possible Approaches
  - Small static pool of ports in control of customer
  - Needs configuration/reservation/security protocols

Network Address Translation today

- Thought as a temporary solution
- Home Users
  - to share one public IP address
  - to hide the network topology and to provide some sort of security
- ISPs
  - for connecting more and more customers
  - for the planned transition to IPv6
- Mobile operators
  - to provide connectivity to a large number of customers
  - „security“
- Enterprises
  - to hide their topology
  - to be address independent

NAT Conclusion

- NAT helps against the shortage of IPv4 addresses
- NAT works as long as the server part is in the public internet
- P2P communication across NAT is difficult
- NAT behavior is not standardized
  - keep that in mind when designing a protocol
- many solutions for the NAT-Traversal problem
  - none of them works with all NATs
  - framework can select the most appropriate technique
- New challenges with the transition to IPv6

Chair for Network Architectures and Services – Prof. Carle
Department for Computer Science
TU München

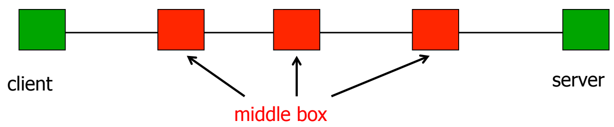
Middleboxes

RFC 3234 - Middleboxes

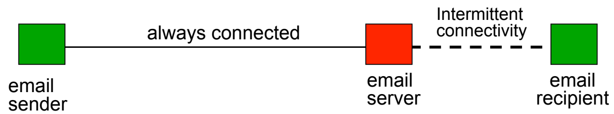
- The phrase "middlebox" was coined by Lixia Zhang as a graphic description of a recent phenomenon in the Internet.

Lixia Zhang, UCLA

What are *middle* boxes?

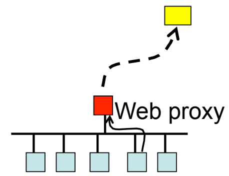


- data is no longer delivered between the two end boxes by *direct* IP path
- The first middleman: email server

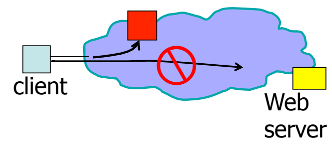


Middleboxes

- Web proxies



- "transparent" Web caches



Packet hijacking!("for your benefit")

Middleboxes Address Practical Challenges

- IP address depletion
  - Allowing multiple hosts to share a single address
- Host mobility
  - Relaying traffic to a host in motion
- Security concerns
  - Discarding suspicious or unwanted packets
  - Detecting suspicious traffic
- Performance concerns
  - Controlling how link bandwidth is allocated
  - Storing popular content near the clients

Layer Violation Boxes

- Peek into application layer headers…
- Send certain packets to a different server…
- Proxy certain request without being asked...
- Rewrite requests …

- Result: unpredictable behaviour, inexplicable failures
- c.f. RFC 3234

RFC 3234 - Middleboxes: Taxonomy and Issues

- A middlebox is **defined** as any intermediary device performing functions other than standard functions of an IP router on the datagram path between a source host and destination host.
- Standard IP router: transparent to IP packets
- End-to-end principle: asserts that some functions (such as security and reliability) can only be implemented completely and correctly end-to-end.
- Note: providing an incomplete version of such functions in the network can sometimes be a performance enhancement, but not a substitute for the end-to-end implementation of the function.

Properties

- Middleboxes may
  - Drop, insert or modify packets.
  - Terminate one IP packet flow and originate another.
  - Transform or divert an IP packet flow in some way.
- Middleboxes are never the ultimate end-system of an application session

- Examples
  - Network Address Translators
  - Firewalls
  - Traffic Shapers
  - Load Balancers

Concerns

- New middleboxes challenge **old protocols**. Protocols designed without consideration of middleboxes may fail, predictably or unpredictably, in the presence of middleboxes.
- Middleboxes introduce **new failure modes**; rerouting of IP packets around crashed routers is no longer the only case to consider. The fate of sessions involving *crashed middleboxes* must also be considered.
- **Configuration** is no longer limited to the two ends of a session; middleboxes may also require configuration and management.
- **Diagnosis** of failures and misconfigurations is more complex.

Middlebox Classification

1. Protocol layer (IP layer, transport layer, app layer, or mixture?)
2. Explicit (design feature of the protocol) or implicit (add-on not by the protocol design)
3. Single hop vs. multi-hop (can there be several middleboxes?)
4. In-line (executed on the datapath) vs. call-out (ancillary box)
5. Functional (required by application session) vs. optimising
6. Routing vs. processing (change packets or create side-effect)
7. Soft state (session may continue while middlebox rebuilds state) vs. hard state
8. Failover (may a session be redirected to alternative box?) vs. restart

Specific Middleboxes

- **Packet classifiers**
  - classify packets flowing through them according to policy
  - either select them for special treatment or mark them
  - may alter the sequence of packet flow through subsequent hops, since they control the behaviour of traffic conditioners.
  - {1 multi-layer, 2 implicit, 3 multihop, 4 in-line, 5 optimising, 6 processing, 7 soft, 8 failover or restart}
- **IP Firewalls**
  - Inspects IP and Transport headers
  - configured policies decide which packets are discarded, e.g.:
    - Disallows incoming traffic to certain port numbers
    - Disallows traffic to certain subnets
  - Does not alter forwarded packets
  - Not visible as protocol end-point

Specific Middleboxes

- **Proxies**
  - An intermediary program that acts as a client and server
  - Makes requests on behalf of a client and then serves the result
- **Application Firewalls**
  - act as a protocol end point and relay (e.g., Web proxy); may

  (1) implement a "safe" subset of the protocol,
  (2) perform extensive protocol validity checks,
  (3) use implementation methodology for preventing bugs,
  (4) run in an insulated, "safe" environment, or
  (5) use combination of above

Middlebox Types according to RFC 3234

1. NAT,
2. NAT-PT,
3. SOCKS gateway,
4. IP tunnel endpoints,
5. packet classifiers, markers, schedulers,
6. transport relay,
7. TCP performance enhancing proxies,
8. load balancers that divert/munge packets,
9. IP firewalls,
10. application firewalls,
11. application-level gateways
12. gatekeepers / session control boxes,
13. transcoders,
14. (Web or SIP) proxies,
15. (Web) caches,
16. modified DNS servers,
17. content and applications distribution boxes,
18. load balancers that divert/munge URLs,
19. application-level interceptors,
20. application-level multicast,
21. involuntary packet redirection,
22. anonymizers.

bold - act per packet
- do not modify application payload
- do not insert additional packets

Assessment of Middlebox Classification

1. Protocol layer (IP layer, transport layer, app layer, or mixture?)
2. Explicit (design feature of the protocol) or implicit
3. Single hop vs. multi-hop (can there be several middleboxes?)
4. In-line (executed on the datapath) vs. call-out (ancillary box)
5. Functional (required by application session) vs. optimising
6. Routing vs. processing (change packets or create side-effect)
7. Soft state (session may continue while rebuilding state) vs. hard state
8. Failover (may a session be redirected to alternative box?) vs. restart

Of 22 classes of Middleboxes:
- 17 are application or multi-layer
- 16 are implicit
- 17 are multi-hop
- 21 are in-line; call-out is rare
- 18 are functional; pure optimisation is rare
- Routing & processing evenly split
- 16 have hard state
- 21 must restart session on failure

Assessment

- Although the rise of middleboxes has negative impact on the end to end principle at the packet level, it is still a desirable principle of applications protocol design.
- Future application protocols should be designed in recognition of the likely presence of middleboxes (e.g. network address translation, packet diversion, and packet level firewalls)
- Approaches for failure handling needed
  - soft state mechanisms
  - rapid failover or restart mechanisms
- Common features available to many applications needed
  - Middlebox discovery and monitoring
  - Middlebox configuration and control
  - Routing preferences
  - Failover and restart handling
  - Security