

# Realtime Multimedia in Presence of Firewalls and Network Address Translation

Knut Omang  
Ifi/Paradial



## About Knut Omang

Knut Omang:

- PhD. from UiO
- Worked on network technology and network centric applications in industry for many years:
  - Dolphin, Sun, Scali, Fast, now at Paradiad
- Associate Professor with DMMS at Ifi, UiO
  - 20% position) for > 10 years



## About Paradiad

- Founded in 2001
- Based in Oslo, 11 employees.
- Software company:
  - RealTunnel: Firewall/NAT traversal
  - PANE: Network and firewall emulator



- <http://www.paradiad.com>

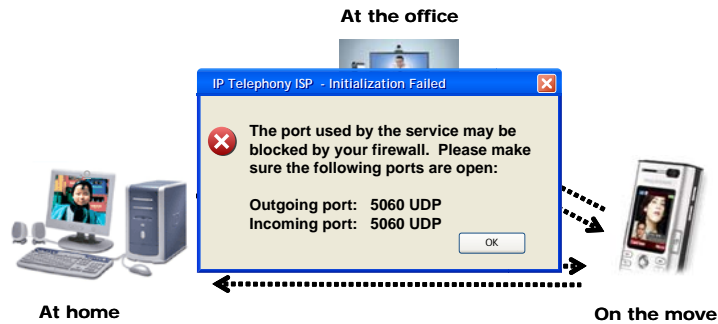


## Today: RT multimedia and connectivity

- Mobile users (roaming between devices) or mobile devices
  - Applications: “unified communications”
  - Common characteristics of unified communication demands
- Firewalls and NAT
  - Firewalls and NAT characterization
  - Why and how this is a problem for RT multimedia
- Efforts to aid in discovery and traversal
  - STUN (Simple Traversal Utilities for NAT)
  - TURN (Traversal Using Relays around NAT)
  - ICE (Interactive Connectivity Establishment)
  - Tunnelling
  - Modifying/involving the firewall
- Example session management and flow
  - Simple SIP/SDP example
  - A more complex call scenario using SIP for setup
- What about IPv6?
  - Firewalls: No doubt
  - NAT?



## "Unified communications" ...



## "Unified communications"

- Types of service
  - Voice
  - Video
  - Chat/presence
  - Application sharing
- Protocols:
  - Session layer
    - SIP, H323, XMPP (Jabber), MSNP, IPsec (tunnel mode), Oscar (AIM), Skype
  - Transport layer
    - IPsec (ESP/AH)
- Similar issues for all protocols and services!



## "Unified communications"

### Characteristics:

- Real-time properties needed
- Multiple media flows
- Not the usual client/server model
- Want shortest/best path for media
  - Delay
  - Jitter
  - Resource usage



## A simple call example

Assuming (simplified):

- Per knows Ida's network location (IP+port)
- Only one type of communication needed



# A simple call example

Assuming:

- Per knows Ida's network location (IP+port)
- Only one type of communication needed
- But Ida is visiting a company network
  - with an open policy...



- Per can't reach Ida
- Ida can reach Per and Per can respond (over the same connection)



# A simple call example

Assuming:

- Per knows Ida's network location (IP+port)
- Only one type of communication needed
- Ida is visiting a company network...
- And so are Per...

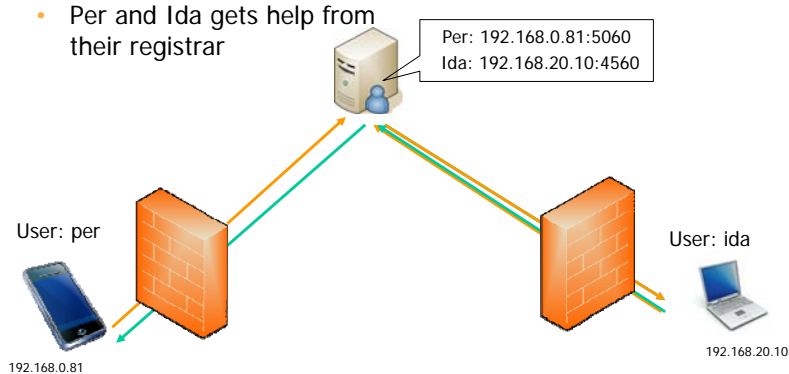


- No communication possible by default
- A 3rd party is needed



# A simple call example

- Per and Ida gets help from their registrar

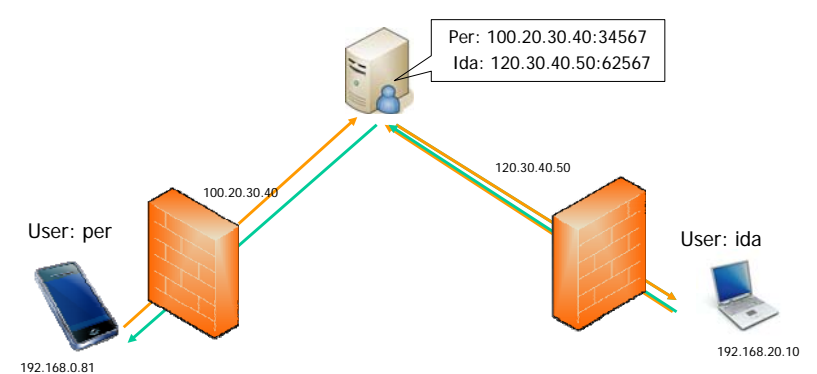


- Depending on firewall:
  - Direct communication may not be possible!



# A simple call example

~~The Per and Ida's firewalls~~ perform NAT!



- NAT-devices alters the IP and TCP/UDP headers of packets
- What if payload also contains addresses?



# Firewalls

- Usually blocks all incoming traffic not on ESTABLISHED connections
  - All communication must be initiated from the inside!
- May block certain protocols
  - UDP considered dangerous
- May block a certain protocol with the exception of certain 'well protected' ports
- May behave differently for different src/dst hosts/port combinations
- May block everything except services considered safe
  - Everything blocked except a local web proxy
  - The web proxy may require authentication and only HTTPS may pass through...
- A user may be behind multiple firewall/NAT devices...
  - Each adding to the complexity..



# Network Address Translation (NAT)

- Source NAT
  - Both sides require outbound traffic to create NAT binding
  - Only receiver can detect a sender's port
  - May vary between destination hosts!
  - We don't know the address/port allocation scheme.. (!)
- Destination NAT
  - Specific addr:port pairs on the outside is bound to addr:port on the inside
  - Used for public services – not the problem here
- Masquerade (often called 'static nat')
  - Destination + Source NAT
- The sender may not know it's public address(es)/port(s)
  - Different destination host:ports may see different sources for the same private address:port



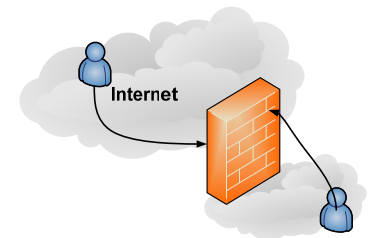
# Firewalls and connection tracking

- Connection tracking?
  - Need to keep track of communication initiated from inside
  - Also needed for NAT to map public addr:port pairs to a private addr:port
- TCP: follows TCP states
- UDP is connectionless?
  - A UDP communication path is said to have been ESTABLISHED it has been responded to
  - But short timeout (eg.30 seconds..)
    - for security...
    - To reduce memory need for connection tracking
- Implementations: Memory usage priority
  - NAT: Simple random port allocation scheme easier than trying to maintain any coherence seen from the outside..
  - More on this later...
- Timeouts means: Keepalive needed to keep firewall 'open'



# Summary: The Connectivity Challenge

- Firewall/NAT devices interfere and often block communications
  - Can't send packets to a private address from the internet
  - Firewalls only accept outbound connections initiated from the inside
  - NAT: Packets from the same port may be seen with different src by different hosts!
  - Firewall 'holes' times out quickly!
  - Users normally do not know the infrastructure between two points



- Most protocols for RT multimedia
  - Uses multiple ports for a single application
  - Puts address/port information in session setup packets – violated by NAT, blocked by firewall..



## STUN (Session Traversal Utilities for NAT)

- Client/server based protocol
- Designed to allow detection of firewall/NAT properties:
  - Firewall characteristics
    - Can we use the desired protocol?
    - Can we communicate directly?
  - NAT
    - discover public addresses assigned to address/ports on the private network
    - Discover if the public address seen by one destination host/port can be used by another



## Some example 'course grained' firewall 'classes':

- All TCP/UDP allowed + NAT
  - When initiated from inside
  - When not any 'dangerous' ports...
  - Source NAT
- All TCP allowed
  - UDP allowed, but only from addr:port sent to
  - no NAT
- All TCP allowed – from inside
  - But all UDP blocked
- All UDP/TCP blocked
  - except https initiated from inside
- All direct access to outside forbidden
  - Internet access only via web proxy in DMZ
  - All web proxy traffic authenticated in proxy



## NAT characterization (UDP focus)

Categorization of NAT devices into classes  
(RFC 4787 – revised from earlier attempts)

- Endpoint independent mapping
  - A NAT mapping from one source addr:port to one destination addr:port can be reused by another destination addr:port
- Address dependent mapping
  - A NAT mapping from one source addr:port can be reused by other destination ports on the same destination host
- Address and port dependent mapping
  - Each (source addr:port, destination addr:port) tuple receives a unique public addr:port in the NAT (no reuse across ports/addresses)

**Note: Nothing prevents a NAT device from behaving differently depending on source or destination addresses or ports in question!**

Example: STUN ports for UDP: endpoint independent, other ports just blocked for UDP...



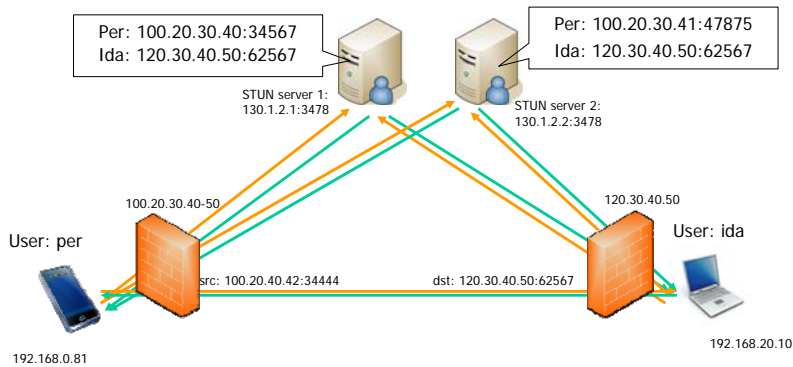
## Firewall/NAT characterization and TCP

• Work in progress: <https://www.guha.cc/saikat/stunt-results.php?>



# NAT detection using STUN

- Per and Ida's firewalls are behind NAT
  - can they talk directly?

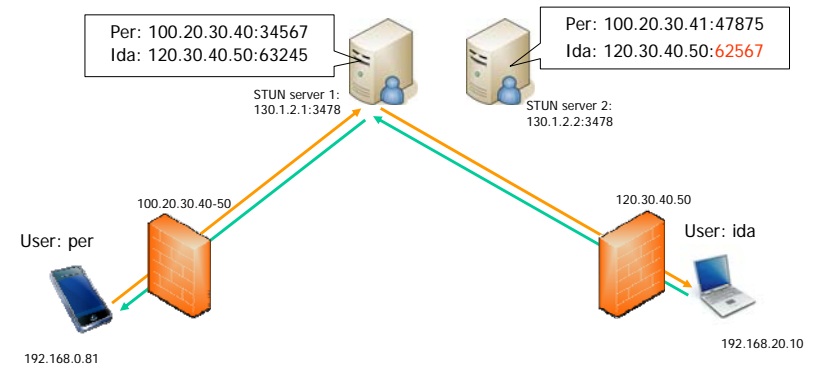


- Ida's firewall has endpoint independent mappings
- We are able to communicate directly if Per initiates
  - if we are lucky...



# NAT detection using STUN

- Per and Ida's firewalls are behind NAT
  - can they talk directly this time?



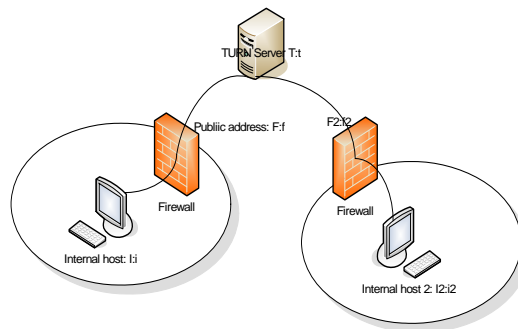
- Neither firewall has endpoint independent mappings
- We are able to communicate using UDP, but only using relay



# STUN Relay -TURN (IETF draft)

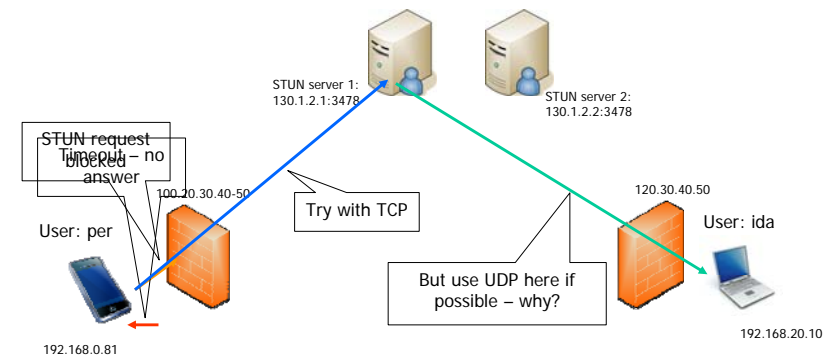
(Traversal Using Relays around NAT)

- Clients request a public port on a relay server
- Server forwards incoming and outgoing traffic between two callers
- Clients may communicate with different relay servers
- Clients may use UDP, TCP or TLS as transport for TURN messages



# What if STUN probes fail?

- Per is behind a UDP blocked NAT



- TCP and packet loss or reorder may distort sound/video!
- Not to mention Nagle's algorithm



## Guaranteed delivery vs realtime, best effort

### Guaranteed delivery

- Increased latency over inconsistency
  - packet loss/corruption/reorder not tolerated
- Bandwidth over smooth flow

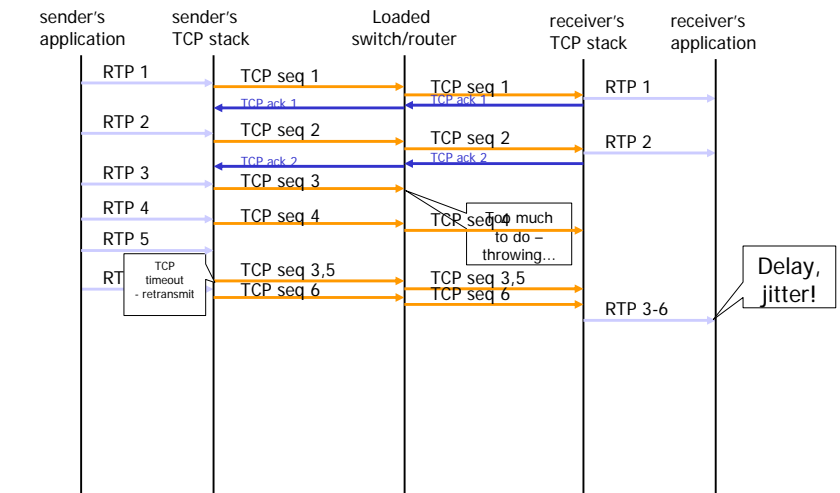
### Realtime, best effort

- Rather lose something than get behind
- Jitter/bursts are bad – smoothness needed!



## Example flow:

Media over TCP with overloaded switch (or a bad network...)



## Media quality

- UDP (usually) preferable over TCP
- As few hops as possible
  - Price and quality issue..
- If TCP must be used, use as short as possible (and UDP for the rest)



## (N)ICE (Interactive Connectivity Establishment)

1. Gather as many addresses/port pairs as possible
  - Local, STUN, TURN, ...
2. Exchange alternative lists
3. Peers verifies available candidates:
  - Defines priorities of candidates
  - Probes candidates
  - Selects the best verified candidate
  - Defined for use with SIP/SDP – work to extend to other protocols

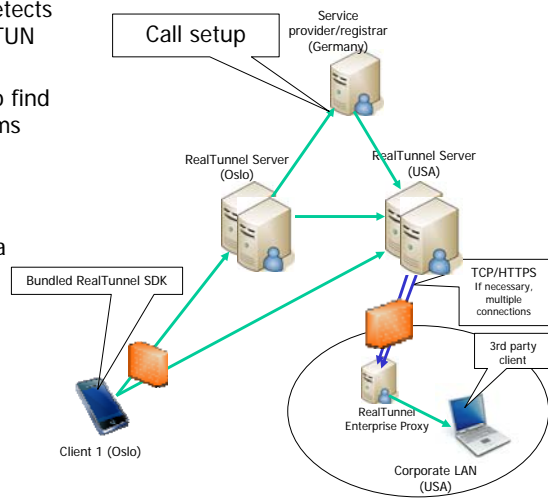
=> ICE is a user of the underlying connectivity checks and relay methods.

- end-to-end protocol between clients
- servers not directly involved.



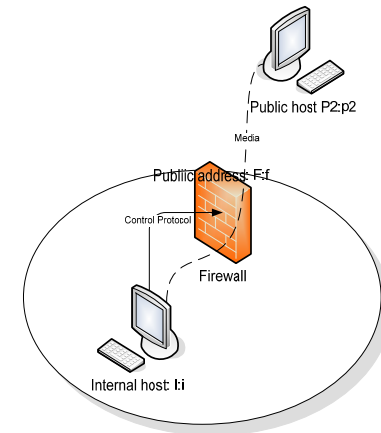
# RealTunnel

- Client on the local network detects network connectivity using STUN and some more...
- Works together with server to find available transport mechanisms towards the peer in a call
- Use ICE client-to-client to communicate alternatives
- Separate signalling and media servers for scalability
- Client software depending on application:
  - SDK
  - Enterprise Proxy



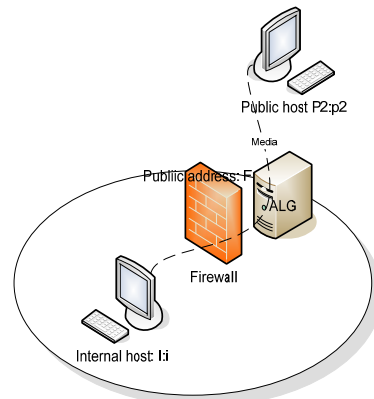
# Controlled Firewalls

- Separate firewall control protocol
- Client requests opening of ports in firewall
- SOCKS, UPnP
- Must trust all clients on local network
- UPnP: Clients must trust any device replying to broadcast



# Application Level Gateways (ALG)

- Protocol aware server in connection with firewall
  - Allowed in FW rules
  - Controls firewall through control protocol
  - Or built into firewall
- Old proto example: ftp
- Some VPN gateways
- Session Border Controller:
  - SIP ALG
  - Terminates traffic in both directions
  - Masks real source/destination at the session layer



# Application Level Gateways (ALG)

- Breaks firewall principle:
  - Interferes at session/application level
- Requires care:
  - String replace not enough – must understand protocol
    - Encrypted data? (ex. SIP/TLS, IPsec,...)
    - Hardware/firmware and rapidly evolving protocols?



## Some protocols that struggle with firewall/NAT

- SIP (Session Initiation Protocol)
- H323 (ITU – telecoms standard)
- Oscar (AIM, ICQ, IChat)
- Skype
- MSNP (MSN)
- XMPP (Jabber, GoogleTalk/libjingle)
- IPsec (AH/ESP/Tunnel mode)



## Session Initiation Protocol (SIP)

- HTTP inspired protocol for session management:
  - Registration/location information
  - Establish (and take down) session
  - Multiple channels (voice, video, ppt, chat, ...)
  - Add/remove participants
  - Renegotiate media
  - 'Forking'
  - Bridge to POTS (Plain Old Telephony Systems)
- Session layer running on top of different protocols (UDP, TCP, ...)
- Increasingly used for IP telephony and video conferencing
- Lots of equipment available/being developed that uses SIP in some form
- Also 'embraced' by telecoms by means of IMS



## SIP and firewall/NAT

- STUN, TURN, ICE development driven by SIP community...
- Still not fully solved with standards
- Weak ALG implementations complicate
- Increasing awareness in community
- With RealTunnel, ICE used with:
  - STUN, TURN
  - tunnelling protocol for cases where STUN/TURN does not provide working candidates



## H323 and Firewall/NAT

- Similar problems
  - Somewhat constrained by H323s functionality compared to SIP
- Devised solution: H460.x
  - Standards for NAT traversal
  - Supported a.o. with RealTunnel...



## Oscar (AIM, ICQ, IChat) and Firewall/NAT

- Supports most types of firewalls and NATs
- Similar mechanisms as used for SIP
- Even tunnelling over https if necessary



## Skype and firewall/NAT

- Based on software developed for Kazaa
  - Per-to-peer intended for file sharing
  - Gets much 'for free' from P2P..
  - Proprietary and encrypted
  - Solves NAT traversal with similar strategies as STUN/TURN and tunnelling if necessary
  - Relaying via (and relying on) private users
  - Breakdown in 2007 thought to be due to diminishing number of non-fw/NAT'ed users



## MSNP (MSN) and firewall/NAT

- RealTunnel used to serve MSN community with free download
  - Had significant user base
- Microsoft increasingly aware of problem
- Current status:
  - Chat works
  - Video/audio may/may not work...
  - Uses Upnp + socks + STUN++



## XMPP and Firewall/NAT

- Jabber
- Basis for Gooletalk/jingle:
  - Implementing STUN/TURN/ICE for XMPP(++)



## IPsec

- AH (Authentication Header protocol)
  - Additional header that authenticates the information in the IP header
    - Ensures that the IP header information is not changed by intermediates..
    - Does not encrypt payload but checksums it to avoid tampering..
- ESP (Encapsulating Security Payload protocol)
  - Encapsulates and encrypts a checksum of the IP payload
  - does not checksum/verify the IP header
- Tunnel mode:
  - Either or combinations of ESP/AH where the payload is a complete IP packet with its own IP header



## IPsec and Firewall/NAT

- AH:
  - Defeats the very purpose of NAT!
    - Any NAT'ed packets will be dropped..
- ESP:
  - Address modifications may pass
  - Port modifications lead to dropped packets!
    - The port numbers are in the transport headers (UDP/TCP..) which are inside the encrypted payload!



## IPsec and Firewall/NAT (2)

- NAT-T (RFC 3947,3948)
  - Adds an additional UDP header to ESP packets
  - Defines port 4500 as the ipsec-nat-t port
  - Limited: Requires one side to have destination nat ('static' nat)
  - Defines a protocol for exchanging NAT information
- Problem:
  - DNAT allows a single listener
  - What if multiple VPNs are to communicate across the same NAT devices?
  - Approach requires firewall access – not always possible



## Firewalls and IPv6

- Most of the problems remains
  - Open/blocked ports
  - Detecting best path for media
  - UDP/TCP
  - QoS problems



# NAT and IPv6 – what will happen?

## Pros:

- Avoiding renumbering:
  - when changing Internet provider..
- Facilitating multi-homing
  - Use of multiple providers from same domain
- Topology hiding
  - Preventing host counting by attackers
  - Router scalability
- IPv6 already old – what about host mobility?

## Cons:

- End-to-end routability
  - All the complexity of NAT in today's lecture..
- Much reconfiguration to do already
  - DHCP, DNS dynamic update
- Provider independent IP blocks
  - Good for Cisco,Juniper,Nortel... ☺

(see ao. <http://tools.ietf.org/html/draft-iab-ipv6-nat-00>)



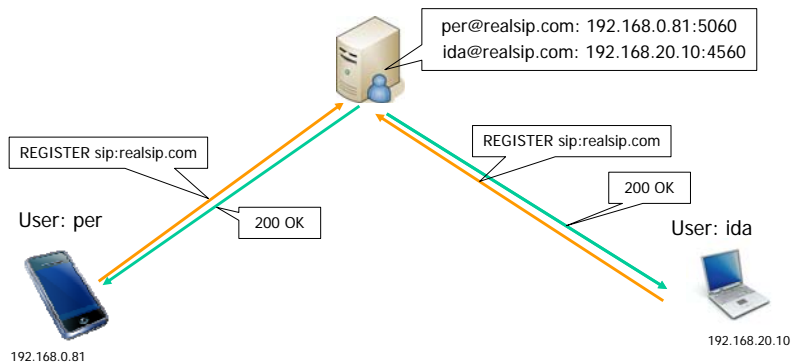
# More SIP: SIP background

- Designed to be flexible and extendable
  - Initial draft (SCIP) 18 pages long
  - A bit like using category theory to describe math ☺
- Lots of specific needs accommodated for later
  - Current SIP RFC 3261 is 269 pages long.
  - RFC 5411: 'A hitchhiker's guide to SIP'
    - 39 pages of listing of extensions/related standardization(!)
    - >> 100 standardization attempts for extensions and growing..
    - Few (if any?) have implemented them all so far
- But, by far the most flexible approach



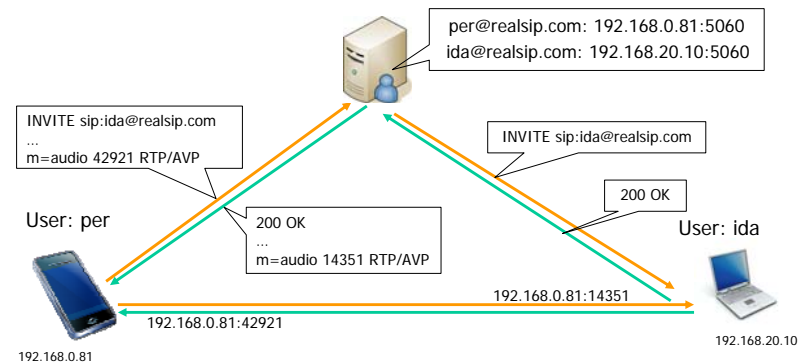
# Our simple call example: (now with SIP)

- Registration

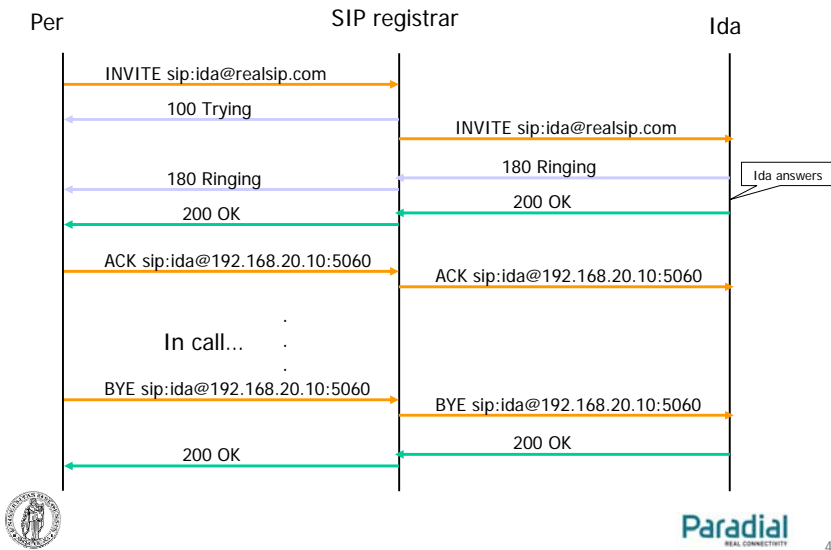


# Our simple call example: (now with SIP)

- Call:



# A complete SIP dialogue



# A SIP request with SDP

```

INVITE sip:ida@realsip.com SIP/2.0
Via: SIP/2.0/UDP 192.168.0.81:5061;branch=z9hG4bK-32063-1-1
From: "per" <sip:per@realsip.com>;tag=1
To: <sip:ida@realsip.com>
Call-ID: 1-32063@192.168.0.81
CSeq: 1 INVITE
Contact: <sip:per@192.168.0.81:5061>
Max-Forwards: 70
Subject: Simple Call
Content-Type: application/sdp
Content-Length: 186
    
```

```

v=0
o=- 53655765 2353687637 IN IP4 192.168.0.81
s=-
c=IN IP4 192.168.0.81
t=0 0
m=audio 6001 RTP/AVP 8
a=rtpmap:8 PCMA/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-11,16
    
```



# A SIP request with SDP (received)

```

INVITE sip:ida@192.168.0.81:5062 SIP/2.0
Via: SIP/2.0/UDP 192.168.0.84:5060;branch=z9hG4bK-ffdef4f8xc0a80054
Via: SIP/2.0/UDP 192.168.0.81:5061;branch=z9hG4bK-32063-1-3
From: "per" <sip:per@realsip.com>;tag=1
To: <sip:ida@realsip.com>
Call-ID: 1-32063@192.168.0.81
CSeq: 2 INVITE
Contact: <sip:per@192.168.0.81:5061>
Max-Forwards: 69
Subject: Simple Call
Content-Type: application/sdp
Record-Route: <sip:192.168.0.84:5060;lr>
Content-Length: 186
    
```

```

v=0
o=- 53655765 2353687637 IN IP4 192.168.0.81
s=-
c=IN IP4 192.168.0.81
t=0 0
m=audio 6001 RTP/AVP 8
a=rtpmap:8 PCMA/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-11,16
    
```



# A SIP response with SDP

```

SIP/2.0 200 OK
Via: SIP/2.0/UDP 192.168.0.81:5061;branch=z9hG4bK-32063-1-3
Record-Route: <sip:192.168.0.84:5060;lr>
From: "per" <sip:per@realsip.com>;tag=1
To: <sip:ida@realsip.com>;tag=1
Call-ID: 1-32063@192.168.0.81
CSeq: 2 INVITE
Contact: <sip:ida@192.168.0.81:5062;transport=UDP>
Content-Type: application/sdp
Content-Length: 131
    
```

```

v=0
o=- 53655765 2353687637 IN IP4 192.168.0.81
s=-
c=IN IP4 192.168.0.81
t=0 0
m=audio 6000 RTP/AVP 0
a=rtpmap:0 PCMU/8000
    
```



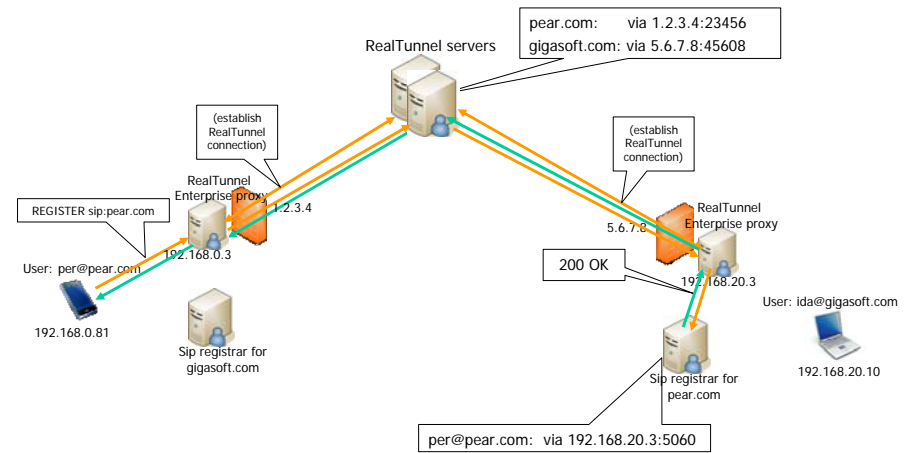
# A SIP response with SDP (received)

SIP/2.0 200 OK  
 Via: SIP/2.0/UDP 192.168.0.84:5060;branch=z9hG4bK-ffdef4f8xc0a80054  
 Via: SIP/2.0/UDP 192.168.0.81:5061;branch=z9hG4bK-32063-1-3  
 Record-Route: <sip:192.168.0.84:5060;lr>  
 From: "per" <sip:per@realsip.com>;tag=1  
 To: <sip:ida@realsip.com>;tag=1  
 Call-ID: 1-32063@192.168.0.81  
 CSeq: 2 INVITE  
 Contact: <sip:ida@192.168.0.81:5062;transport=UDP>  
 Content-Type: application/sdp  
 Content-Length: 131

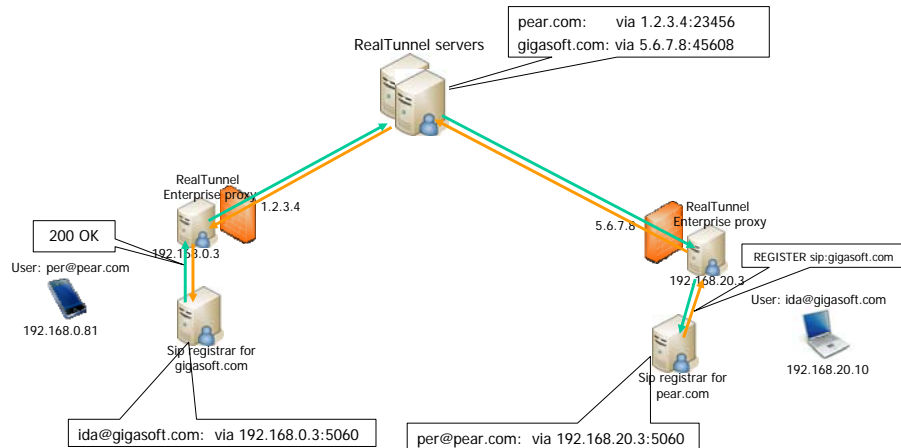
V=0  
 o=- 53655765 2353687637 IN IP4 192.168.0.81  
 S=-  
 c=IN IP4 192.168.0.81  
 t=0 0  
 m=audio 6000 RTP/AVP 0  
 a=rtpmap:0 PCMU/8000



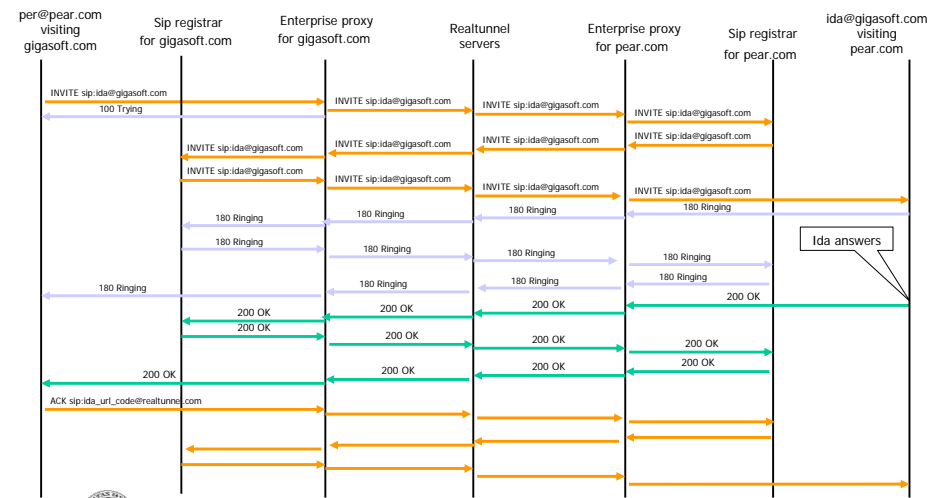
# A more complex example: Registration



# A more complex example: Registration



# A more complex example: call



In call... (puh)

# Summary

- 'Unified communications'
  - demands good RT QoS
  - Can generate very complex initiation scenarios
  - Gets much more complicated due to firewalls and NAT
- Firewalls and NAT
  - Firewalls and NAT can to some extent be characterized
    - But there is an unlimited potential for issues and problems
    - And there is limited observability in the general case!
  - RT multimedia gets into trouble quite easily because
    - High demands for QoS
    - Often different desired paths for setup and media flow
    - Multiple connections needed, using different protocols
  - Firewall modification might seem like a good idea
    - But often complicates rather than simplify (ALG)
    - And can also be a security risk (UPnp)
  - Lots of protocols exhibit similar problems
    - SIP, H323, Oscar, Skype, MSNP, XMPP, IPsec
- Discovery and traversal tools can aid in finding the best path
  - STUN (Simple Traversal Utilities for NAT)
  - TURN (Traversal Using Relays around NAT)
  - ICE (Interactive Connectivity Establishment)
  - Standards still immature – extra measures needed
  - No rescue in IPv6....

