Advanced Networking

# Skype

Renato Lo Cigno
Renato.LoCigno@disi.unitn.it

**Credits for part of the original material to Saverio Niccolini NEC Heidelberg**

## Skype characteristics

- Skype is a well known P2P program for real time communications
  - Voice calls
  - Video (from version 2.0)
  - File sharing and instant messaging when in a call
- Seems to work with no problems in all network conditions compared to similar P2P applications
- One of the reasons of its success is its ability to work in network scenarios with middleboxes
  - such as firewalls and Network Address Translators (NATs)
  - usually, this is a problem for P2P applications

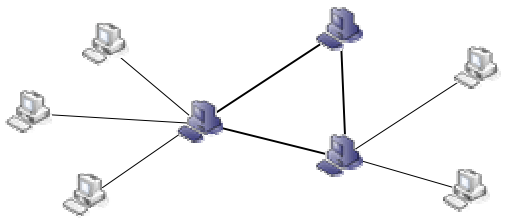


## How Skype works

- Skype overlay network
  - network structure
  - entities involved
- Skype function analysis
- Lesson learned
- Skype security analysis
  - Binary
  - Network protocol
  - Skype authentication
  - Traffic encryption

# Skype overlay network (I)

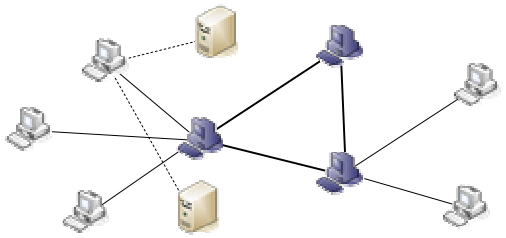
- Skype network relies on distributed nodes:
  - Skype Clients (SCs)
  - Supernodes (SNs)





# Skype overlay network (II)

- Although there are also centralized entities:
  - HTTP Server
  - Login Server





# Skype overlay network (III)

Skype Client
- used to place voice calls and send instant messages
- connection to skype network possible through a supernode (SN)
- connection with the SN (via TCP) maintained for the whole time the client is on-line
- client configuration and SN addresses are stored locally and refreshed periodically to maintain a coherent view of Skype network

# Skype overlay network (IV)

**Supernode**

- Normal Skype Client that can accepts incoming TCP connections, with enough CPU, memory and bandwidth
- There are also a number of "default" Supernodes, used to increase network robustness and stability



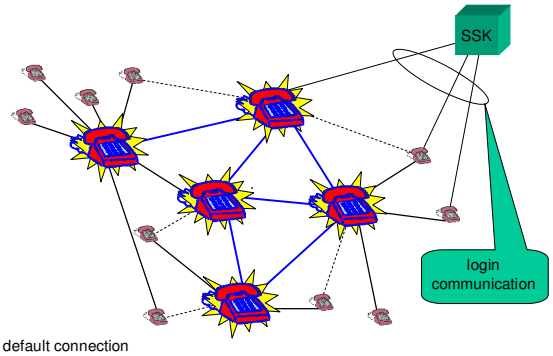
# Skype overlay network (V)

**Servers**

- Login server ensures that names are unique across Skype namespace. Also central point for authentication
- HTTP Server used by clients to check for updates



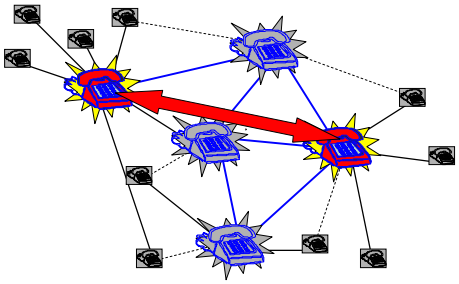
# Topology




default connection

## Topology: calls

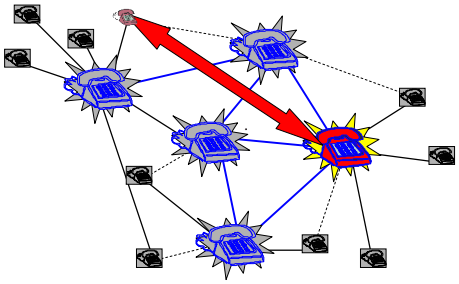
Supernodes communicate directly ...





## Topology: calls

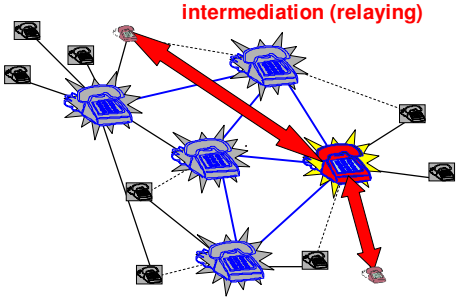
... also with normal peers





## Topology: calls

normal nodes require a supernode intermediation (relaying)



## Some caratteristics

- CODECs
  - Default is a wideband (8 kHz-16kHz sampling) resulting in a transmission rate of 40 kbit/s in each direction (140 pck/s with payload of 67 bytes)
  - Quality in normal conditions is very good, much better than PCM telephony
  - No narrowband coding is provided, congestion is not considered a problem generated by skype
  - Under lab conditions over UDP the system works well even with only 16--20 kbit/s; below 12 kbit/s the system cannot work



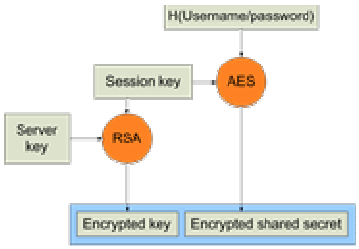
## Some Characteristics

- Ports
  - 80 (HTTP) e 443 (HTTPS) on TCP for signalin, random choice on UDP or TCP for voice
  - Ports are announced on the P2P network
- Encryption
  - All communications are AES (Advanced Encryption Standard) encoded



## Skype Encryption

- **Authentication**
  - **At login time the client generate a RSA session key and uses it to encrypt his credentials.**
  - **Then encrypts the session key using the server's public key**
  - **and sends this information to the login server**




## Some Characteristics

- Host Cache
  - List of supernodes (IP, Port) used to make the search phase faster
  - Roughly 200 entries dynamically updated
  - If the host cache is void skype does not work (some defaults entry are there from the beginning)
  - Une of the critical points for skype functioning
  - The idea is not new to P2P networks and answer to the bootstrap problem ... albeit in a naive way



## Skype functions analysis

- **Essentials**
  - **Login**
  - **Search**
  - **Buddy list signaling**
  - **Call establishment**

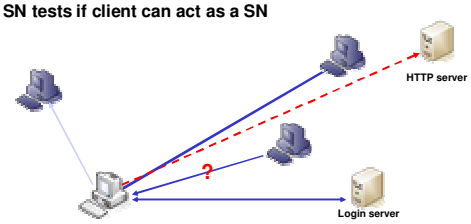


## Login function

- **Join and maintain overlay network:**
  - **Interaction with central servers**
    - **login server manage authentication and ensures unique names**
    - **HTTP server ensures client software updates**
  - **Refresh of shared.xml**
    - **file stored on the client containing SNs list and parameters identifying middlebox**
  - **Network tests if joining client can act as a SN**

# Login procedure

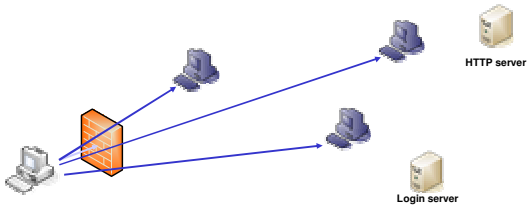
- At startup the client contacts the HTTP server to check for updates
- Sends UDP datagram to a -default SN- to refresh the list of supernodes
- Connects via TCP to a SN (connection maintained throughout Skype session) and exchanges info on online nodes
- Verify username and password via TCP with the Login server
- Another SN tests if client can act as a SN






# Login: Firewall blocks UDP

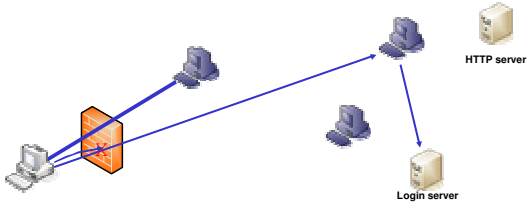
- Firewall prevents UDP exchange for SN list refreshing
- Client establishes several TCP connections with SNs to gather information, when finished all but one are torn down






# Login: Firewall blocks Login sever

- After connection with the SN, attempt to connect with the Login server fails
- Client connect to the Login using a SN as a relay



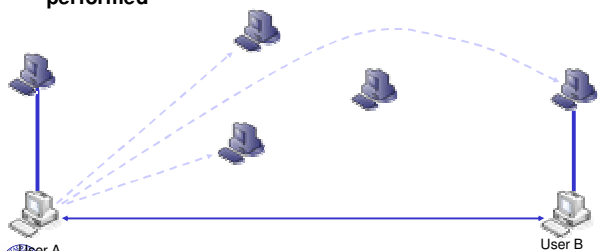
## Search function

- Procedure performed when a user wants to add someone to his buddy's list and communicate for the first time
- Search is performed using username as key
  - possible since names are unique
  - this is why there is the need for central servers



## Search procedure

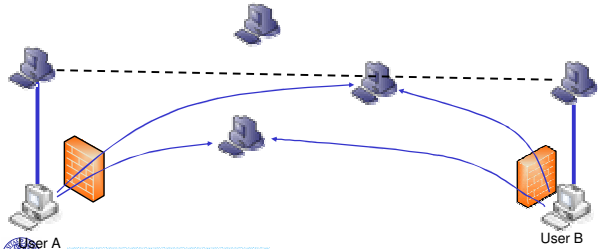
- User A exchanges info with its SN and gather 3 SNs addresses
- A query the 3 SNs via UDP asking if they know the public IP of B
- Once A gets the address of B authorization exchange is performed





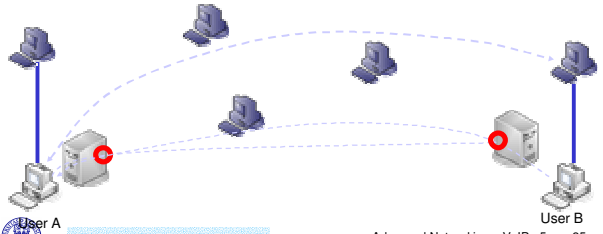

## Search: Firewall blocks UDP

- Firewall blocks UDP
  - preventing direct connection w/ the SNs or another user
  - the SN of A communicate to B (via his SN) the address of A
- Both A and B establish TCP connections with the same 2 SN to exchange authorization




# Search: Port restricted NAT

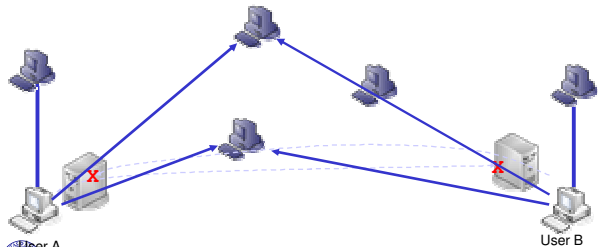
- Once user A gather the address of SN of B, sends a UDP query containing his external address. SN of B replies with user B external address.
- User A send an UDP datagram to user B external address in order to create a mapping in his NAT, anyway packet will be filtered by NAT of B
- User B does the same but this datagram reaches user A
- Once exchanged authorization a TCP connection via 2 SNs as relay is established, as depicted in previous slide






# Search: Symmetric NAT

- Clients try the technique depicted for Port restricted NAT
  - but it fails due to symmetric NAT behvior
- Clients exchange authorization via TCP using 2 SNs as relay



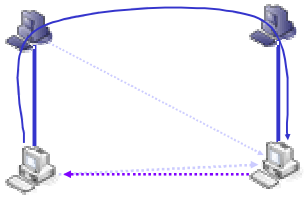



# Buddy list signaling

- Buddy list is a list of “friend” users
- Skype allow a user to know if buddies are online/offline
  - overlay network informs buddies when user change status

## Buddy List signaling procedure

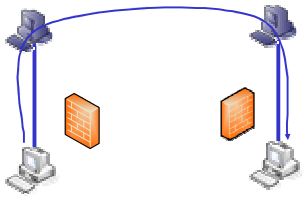
- A user going on-line informs his buddies either directly using UDP or via the SNs.
- When going off-line, a user tear down the TCP connection with the SN.
- The SN informs via UDP the buddies that the user is going off-line
- To have a confirmation buddies try to ping the user.





## Buddy List signaling: Firewall blocking UDP

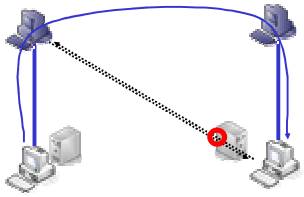
- Since UDP traffic is blocked, on-line/off-line signalling is performed via the SNs





## Buddy List signaling: Port restricted NAT

- On-line/off-line signaling is performed in a way similar to that depicted in previous slide.
- As a difference after the change of status, buddies query the SN of the user for confirmation.



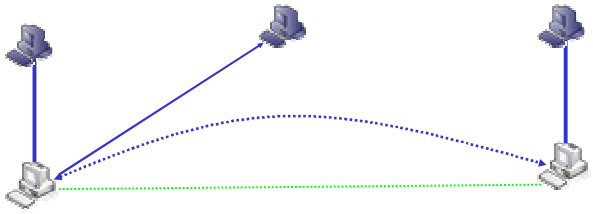
## Call establishment function

- Signaling performed using TCP connection
  - overlay network used only if otherwise impossible
- Media carried over UDP when possible
  - in case relay servers are used



## Call establishment procedure

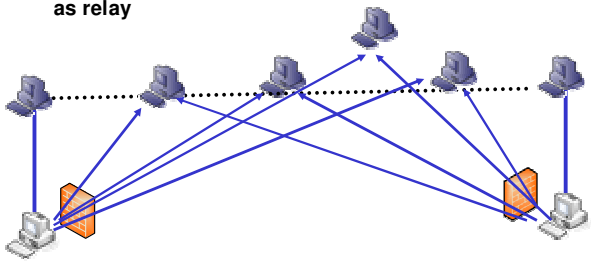
- User A wants to call user B, so he query some SNs for user B address.
- Once he gets user B address they exchange signaling over TCP
- Voice traffic carried via UDP





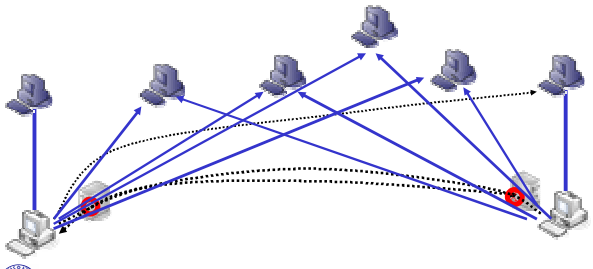
## Call establishment: firewall blocks UDP

- Signaling exchanges are performed by the SNs on behalf of the users
- Media exchange is performed via TCP using 4 SNs as relay



## Call establishment: Port restricted NAT

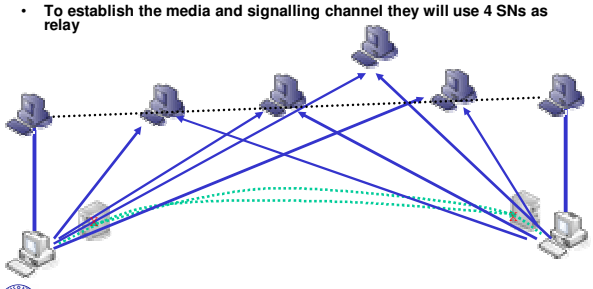
- Once User A gets the address of the SN responsible for user B he queries for his address. SN informs B that user A wants to call him, and tells external address of B to A.
- A and B establish UDP flow using reverse hole punching
- They also establish TCP connection using 4 SNs as relay





## Call establishment: Symmetric NAT

- User A and B communicate their addresses via their SNs
- They try reverse hole punching but it won't work because of NATs restrictions
- To establish the media and signalling channel they will use 4 SNs as relay





## Lesson learned

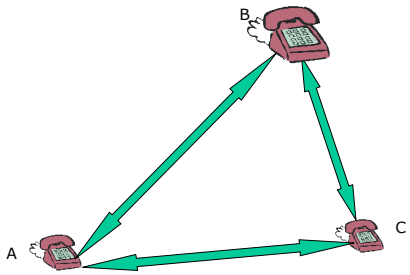
- Traversal is well possible in many cases without explicit signaling to the middlebox
  - open public access network
  - protected enterprise networks
- Reverse hole punching and tunneling techniques workarounds allow Peer-to-peer communications in almost every scenario
  - Skype only fails completely if firewall blocks TCP but in fact that is a very uncommon case
- Explicit middlebox signaling protocols (like IETF MIDCOM MIB, CheckPoint OPSEC, NEC's SIMCO) are still required for
  - highly protected access network
  - applying security policies by network operator
  - anyway Skype will undermine many of these policies
- Skype tries to use IP network instead of overlay
  - SNs can't assure constant presence
  - avoid overlay congestion

## Audio Conference

- Based on traffic mixing in one of the nodes
- Limited to few nodes (5-6)
- Works also with some nodes behind NAT/FW
- The mix node is elected based on it elaboration capabilities, since mixing is CPU intensive
- It does not need to be the conference initiator



## Audio Conference: signaling






## Audio Conference: audio flows