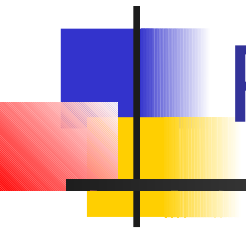
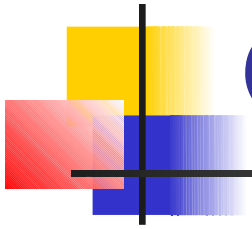


An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol

Speaker：zcchen

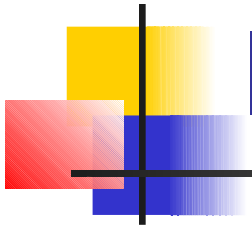


Outline

- Introduction
- Key components of the Skype software
- Experimental setup
- Skype functions
- Conferencing
- Conclusion

Reference

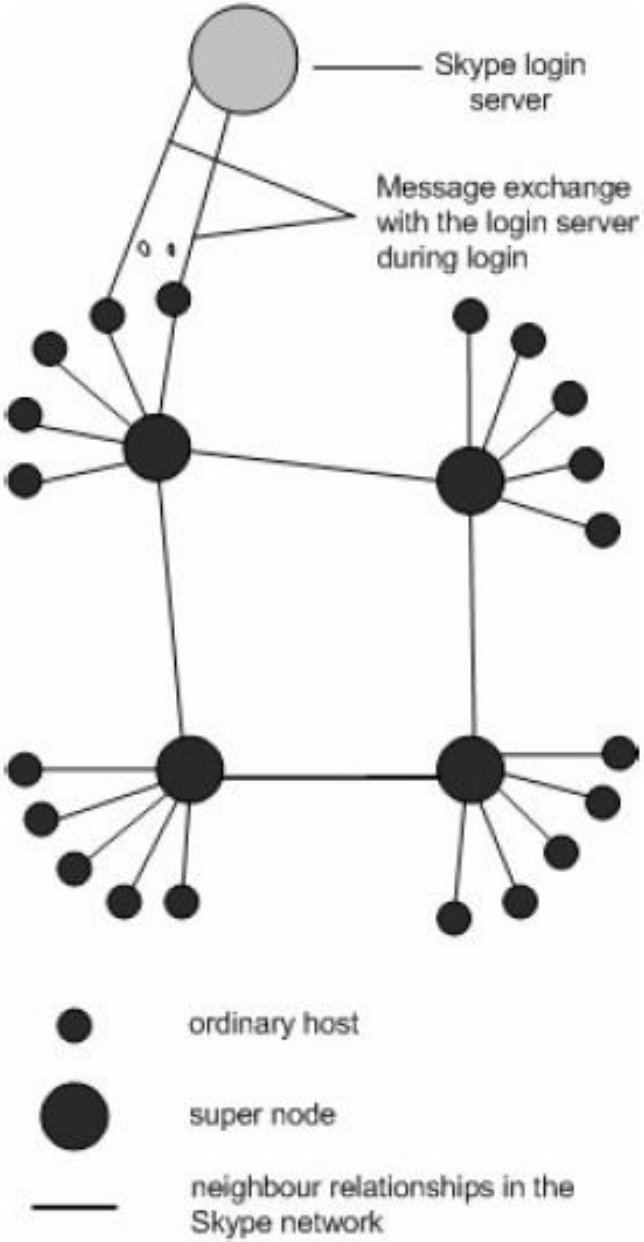
- Paper “ An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol”
- “Skype conferencing white paper”
- “Skype P2P Telephony Explained”
- “ILBC codec”
- “iSAC codec”



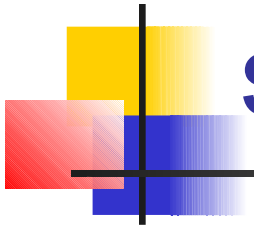
Introduction

- **Skype**
- ✓ A peer-to-peer VoIP client developed by KaZaa in 2003
- ✓ Skype can
  - ✓ work almost seamlessly across NATs and firewalls
  - ✓ has better voice quality than the MSN and Yahoo IM applications
  - ✓ encrypts calls end-to-end, and stores user information in a decentralized fashion
  - ✓ SkypeOut, SkypeIn

- **Type of nodes:**
  - Ordinary hosts
  - Super nodes (SN)
  - Login server



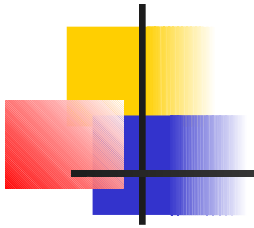
Key components of the Skype software



- Ports
- Host cache
- Codecs
- Encryption
- NAT and firewall

Ports

- A Skype client (SC) opens a TCP and a UDP listening port at the port number configured in its connection dialog box. (randomly chooses one upon installation)
- Default listening port 80(HTTP), 443(HTTPS)



Host cache (HC)

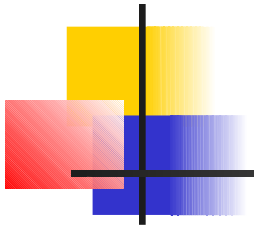
- A list of super node IP address and port pairs that SC builds and refresh regularly.

- At least one valid entry must be present in the HC.
- A SC stores HC in the Windows registry.
- After running a SC for two days, we observed that HC contained a maximum of 200 entries.
- The SN is selected by the Skype protocol based on a number of factors like CPU and available bandwidth.

Codecs

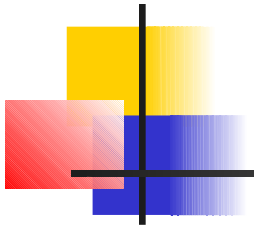
- Skype uses iLBC, iSAC, or a third unknown codec.
- "GlobalIPSound" has implemented the iLBC and iSAC codecs and their websites lists Skype as their partner.
- Skype codecs allow frequencies between 50-8000 Hz. – wideband codec.

- iLBC – bit rate : 13.3 kbps (30 ms frames) 15.2 kbps (20 m frames)
  better speech quality than G.729A and G.723.1.
  supports multiple frames size.
- iSAC – bit rate : 10-32 kbps(adaptive and variable)
  maintain wideband communication over low and high bit rate connection



Buddy list

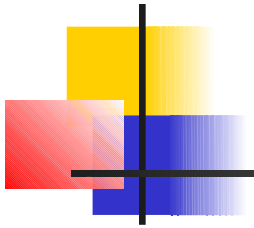
- Skype stores its buddy information in the Windows registry.
- Local to one machine and is not stored on a central server.



Encryption

- Skype uses AES(Rijndel) to protect sensitive information.

- Uses 256-bit encryption, which has a total of $1.1X10^{77}$ possible keys.
- Uses 1536 to 2048 bit RSA to negotiate symmetric AES keys.
- User public keys are certified by login server at login.

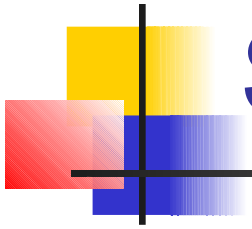


NAT and firewall

- We conjecture that SC uses a variation of the STUN (Simple Traversal of UDP through NATs) and TURN (Traversal Using Relay NAT) protocols to determine the type of NAT and firewall it is behind.
- The information is also stored in the Windows registry.
- Use TCP to bypass UDP-restricted NAT/firewall

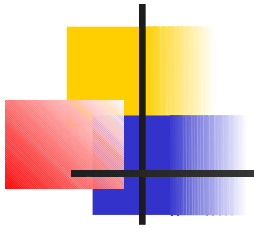
Experimental setup

- Skype version 0.97.0.6.
- Machines : Windows 2000
  PII 200MHz with 128MB RAM.
  10/100 Mb/s Ethernet card.
- Network : 100Mb/s
- Monitor tools : Ethereal & NetPeeker.
- Experiments :
  - 1. both Skype users were on machines with public IP addresses.
  - 2. one Skype user was behind port-restricted NAT
  - 3. both users were behind a port-restricted NAT and UDP-restricted firewall.



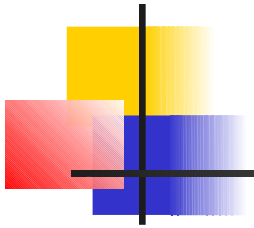
Skype functions

- Startup
- Login
- Call establishment and teardown
- Media transfer and codecs



Startup

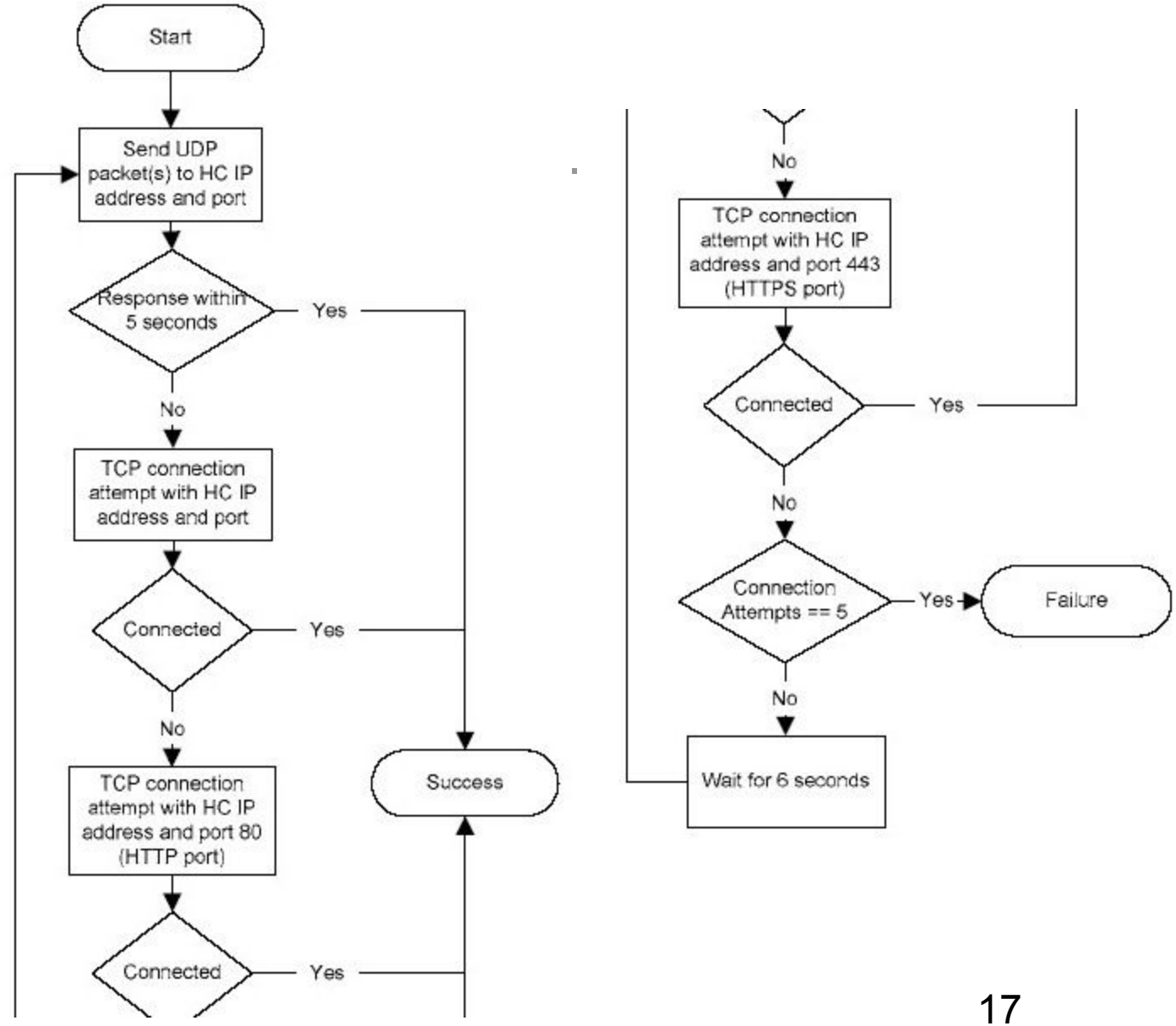
- When SC was run for the first time after installation, it sent a HTTP 1.1 GET request to the ”skype.com”. The first line of the request contains the keyword ‘installed’.
- During subsequent starts-up, a SC only sent a HTTP 1.1 GET to the “skype.com” to determine if the new version is available. The first line of the request contains the keyword ‘getlatestversion’.

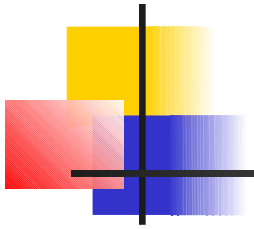


Login

- 1. Advertises its presence to other peers and its buddies.
- 2. Determines the type of NAT and firewall it is behind.
- 3. Discover online Skype nodes with public IP addresses.

- Login algorithm (authentication with login server is not shown)





Login server

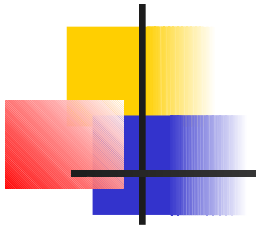
- Stores Skype user names and passwords and ensures that Skype user names are unique across the Skype name space.
- 80.160.91.11 -> ns14.inet.telt.dk(Denmark)

Bootstrap super node (SN)

- After logging in for the first time after installation, HC was initialized with 7 IP addresses and port pairs.

IP address:port	Reverse lookup result
66.235.180.9:33033	sls-cb10p6.dca2.superb.net
66.235.181.9:33033	ip9.181.susc.suscom.net
80.161.91.25:33033	0x50a15b19.boanxx15.adsl-dhcp.tele.dk
80.160.91.12:33033	0x50a15b0c.albnxx9.adsl-dhcp.tele.dk
64.246.49.60:33033	rs-64-246-49-60.ev1.net
64.246.49.61:33033	rs-64-246-49-61.ev1.net
64.246.48.23:33033	ns2.ev1.net

- After installation and first time startup, we observed that the HC was empty. However upon first login, the SC sent UDP packets to at least four nodes in the bootstrap node list.
- Thus, either bootstrap IP address and port information is hard coded in the SC, or it is encrypted and not directly visible in the Skype Windows registry.

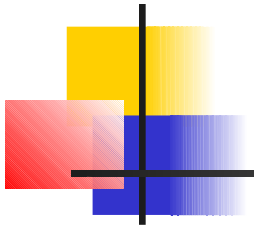


First-time login process

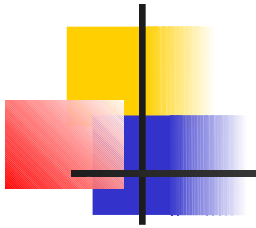
- A SC must connect to well known Skype nodes in order to log on to the Skype network.

- Step 1 : it does so by sending UDP packets to some bootstrap super nodes and then waits for their response over UDP. (it is not clear how SC selects among bootstrap super nodes)

- Step 2 : SC then established a TCP connection with the bootstrap super node that responded in Step 1.



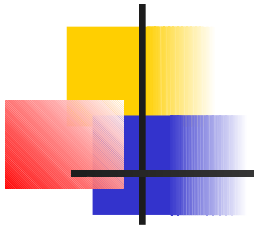
- Step 3 : After exchanging some packets with bootstrap SN over TCP, it then perhaps acquired the address of the login server.
- Step 4 : SC then establishes a TCP connection with the login server, exchanges authentication information with it, and finally closes the TCP connection.
- Step 5 : SC sends UDP packets to 22 distinct nodes and receives response from them over UDP (advertise its arrival).



- The TCP connection with the SN persisted as long as SN was alive. When the SN became unavailable, SC establishes a TCP connection with another SN.
- The total data exchanged between SC, SN, login server, and other nodes during login is about 9 KB.

- For a SC behind a port-restricted NAT, the messages flow for login was roughly the same as for a SC on the public IP address. However, more data was exchanged. – 10KB

- A SC behind a port-restricted NAT and a UDP-restricted firewall was unable to receive any UDP packets from machines outside the firewall. It therefore could send and receive only TCP traffic. – 8.5KB

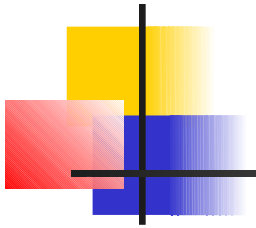


Alternative node table

- It can be seen that SC sends UDP packets to 22 distinct nodes at the end of login process.

  ∵ uses those messages to advertise its arrival on the network. upon receiving a response from them, SC builds a table of online nodes.

- It is with these nodes a SC can connects to, if its SN becomes unavailable.



Subsequent login process

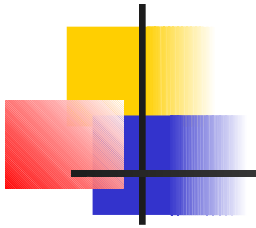
- Similar to first-time login process.
- The HC got periodically update with the IP address and port number of new peers.
- During subsequent logins, SC used the login algorithm (shown in p.17) to determine at least one available peer out of HC.
- It then established a TCP connection with that node.

Login process time

- For the experiment, the HC already contained the maximum of 200 entries.

- 1. SC with public IP address : 3~7 sec.
- 2. SC behind a port-restricted NAT : 3~7 sec.
- 3. SC behind a UDP-restricted firewall : 34 sec.

  (sent UDP packets to 30 HC entries and concluded that it is behind UDP-restricted firewall.)

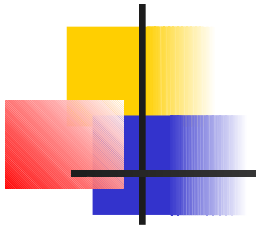


Call establishment and teardown

- We consider call establishment for the three network setups

(We assumed caller and callee were online and in the buddy list of each other)

- Case 1. caller– public IP address
  callee – public IP address
- Case 2. caller – behind port-restricted NAT
  callee – public IP address
- Case 3. caller – behind port-restricted NAT and UDP-restricted firewall
  callee – behind port-restricted NAT and UDP-restricted firewall



- **Case 1 :**

  caller– public IP address

  callee – public IP address

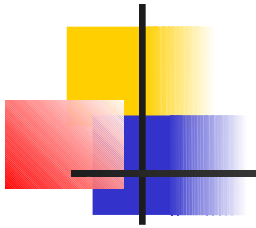
  - The caller established a TCP connection with the callee SC.
  - The caller also sent some messages over UDP to alternative Skype nodes, which are online Skype nodes discovered during login.

- **Case 2 :**

  caller – behind port-restricted NAT

  callee – public IP address

  - Signaling and media traffic did not flow directly between caller and callee. Instead, the caller sent signaling over TCP to an online Skype node which forwarded it to call over TCP.
  - This online node also routed voice packets from caller to callee over UDP and vice versa.

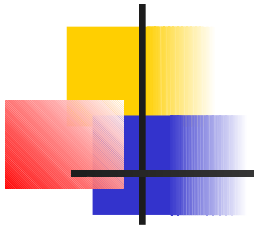


- **Case 3 :**

  caller – behind port-restricted NAT and UDP-restricted firewall

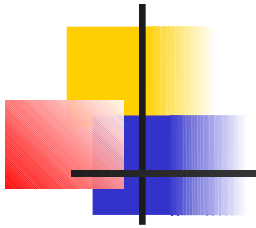
  callee – behind port-restricted NAT and UDP-restricted firewall

- both caller and callee SC exchanged signaling information over TCP with another online Skype node.
- Caller SC sent media over TCP to an online node, which forwarded it to callee SC over TCP and vice versa.



Media transfer and codecs

- Silence suppression
- Putting a call on hold
- Congestion
- Keep-alive message



Silence suppression

- No silence suppression is supported in Skype. when neither caller or callee was speaking, voice packets still flowed between them.

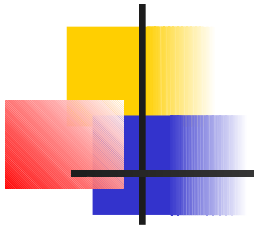
Adv :

1.it maintains the UDP bindings at NAT.

2.these packets can be used to play some background noise at the peer.

- In the case where media traffic flowed over TCP between caller and callee, silence packets were still sent.

∵ avoid the drop in TCP congestion window size, which takes some RTT to reach the maximum level again.



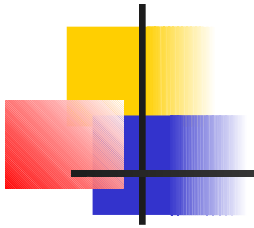
Putting a call on hold

- On average, a SC sent three UDP packets per second to the call peer, SN, or the online Skype node acting as a media proxy when a call is put on hold.

  ∵ensure UDP binding are made at a NAT

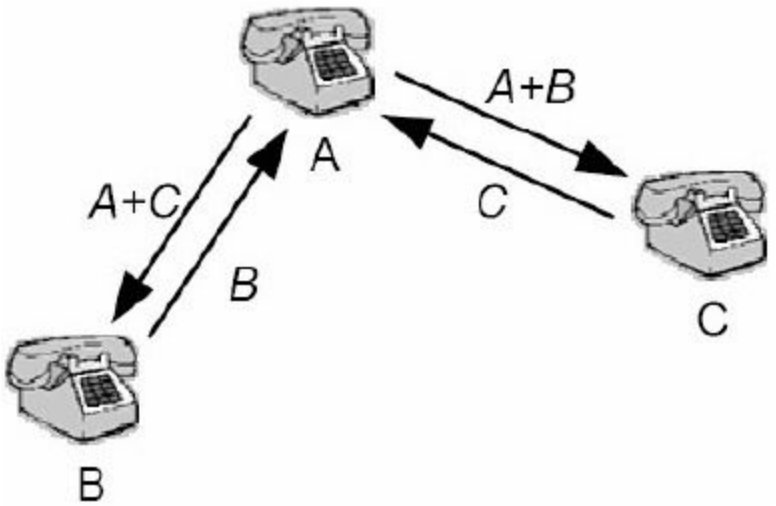
Congestion

- We observed that uplink and downlink bandwidth of 2 KB/s each was necessary for reasonable call quality.
- The voice was almost unintelligible at an uplink and downlink bandwidth of 1.5 KB/s.

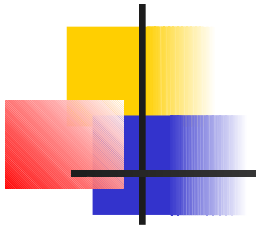


- **Keep-alive message**
  - We observed in for three different network setups that the SC sent a refresh message to its SN over TCP every 60s.

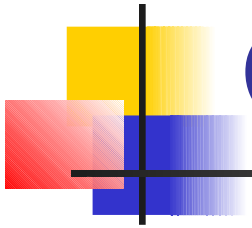
Conferencing



- A call was established between A (the most powerful one) and B. Then B decided to include C in the conference. From the ethereal dump, we observed that B and C were sending their voice traffic over UDP to SC on machine A, which was acting as a mixer.
- It mixed its own packets with those of B and sent them to C over UDP and vice versa



- Even if user B or C started a conference, A, which was the most powerful amongst the three, always got elected as conference host and mixer.
- If iLBC codec is used, the total call 36 KB/s for a two-way call. For three-user conference, it jumps to 54 kb/s for the machine hosting the conference.
- For a three party conference, Skype does not do full mesh conferencing.
- To host a conference with 5 parties you need a big PC, a Pentium 4 or thereabouts. With a PIII CPU of 450 MHz you will be limited to hosting 3 parties.



Conclusion

- Skype is the first VoIP client based on peer-to-peer technology. We think that three factors are responsible for its increasing popularity.
- 1. better voice quality than MSN and Yahoo IM clients.
- 2. work almost seamlessly behind NATs and firewalls.
- 3. extremely easy to install and use.