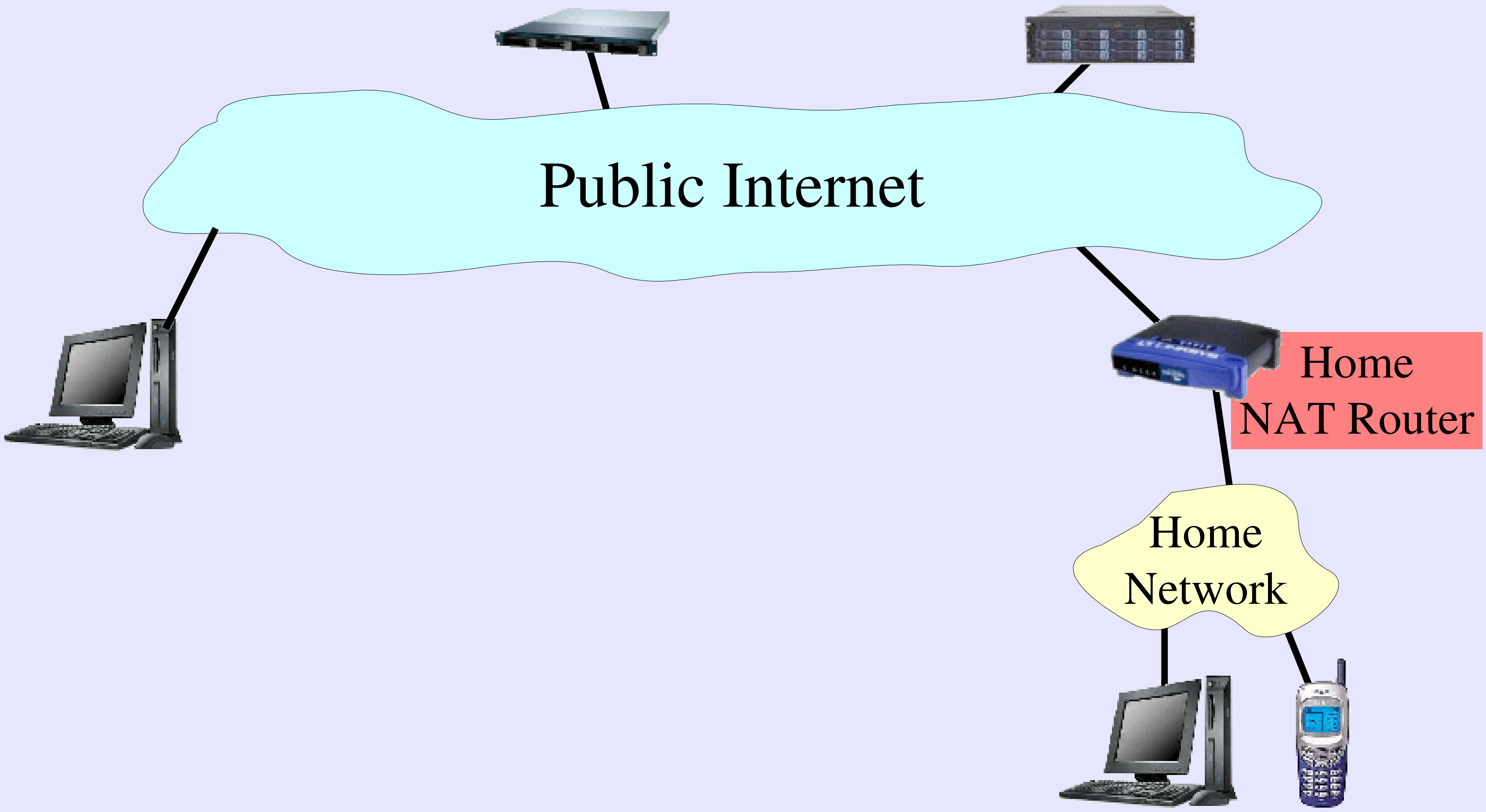
# Peer-to-Peer Communication Across Network Address Translators

Bryan Ford – M.I.T.
Pyda Srisuresh – Caymas Systems
Dan Kegel – Ixia Communications

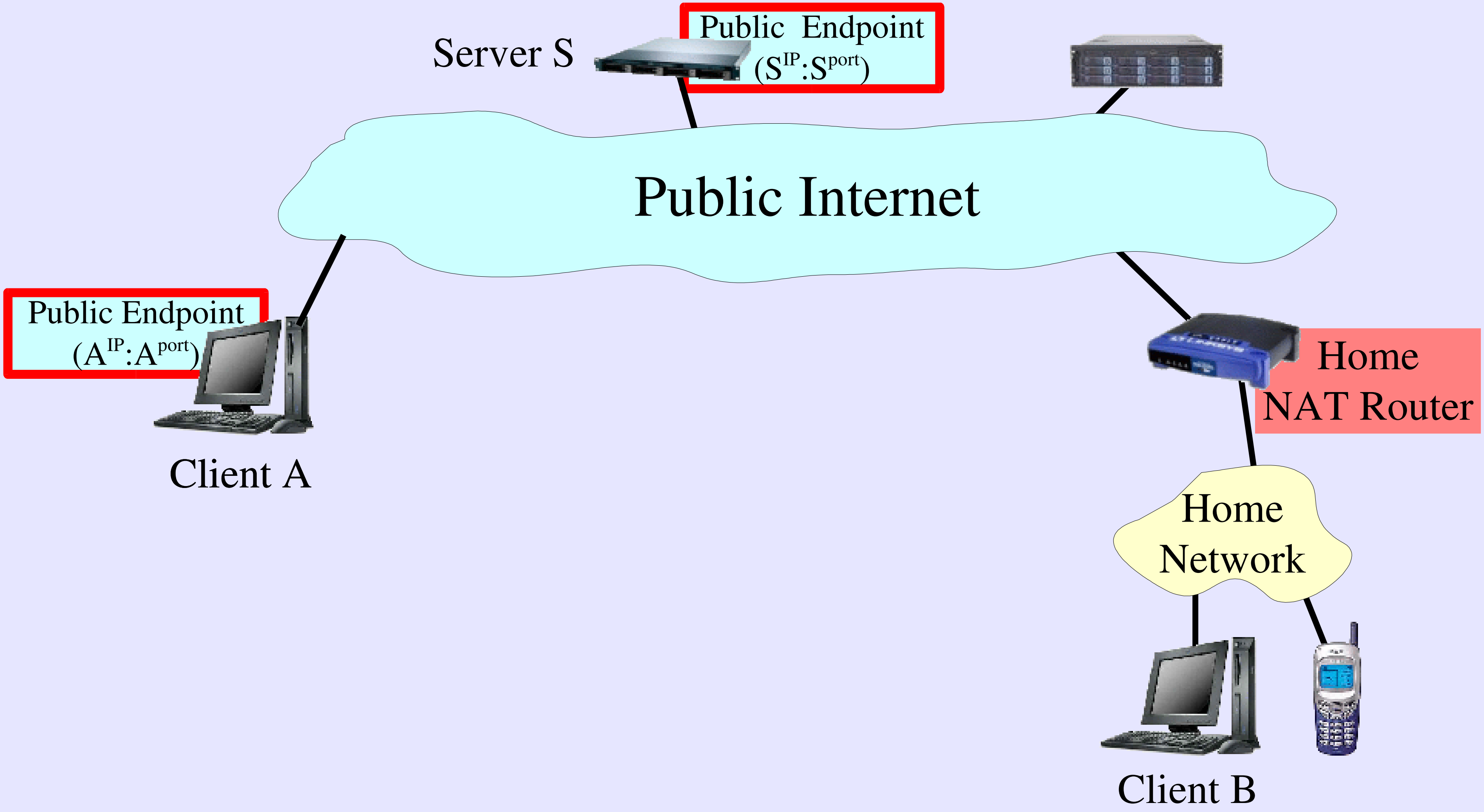
*J'fais des trous, des petits trous...*
*toujours des petits trous*
– S. Gainsbourg

USENIX – April 14, 2005

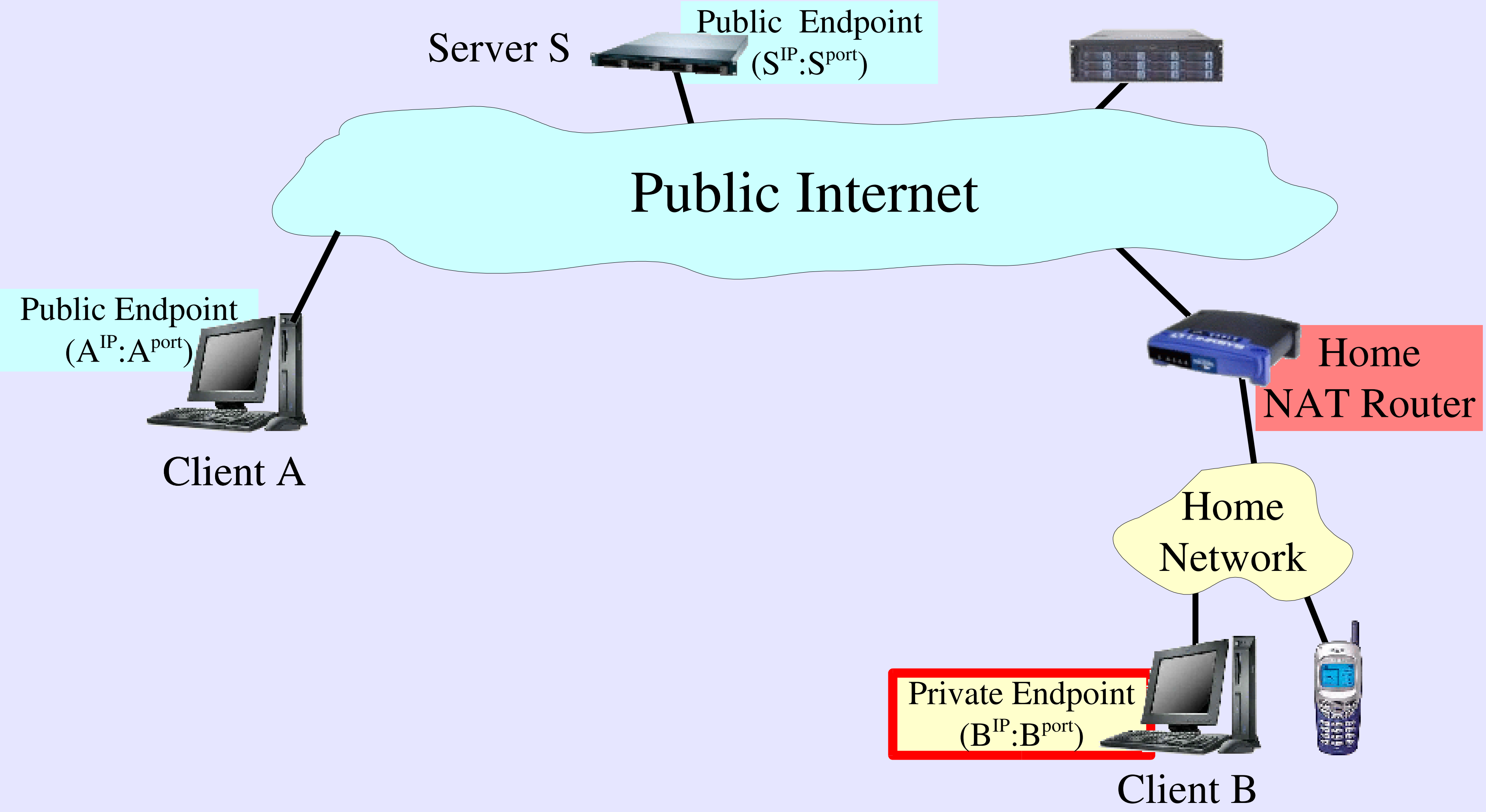
# Network Address Translation (NAT)



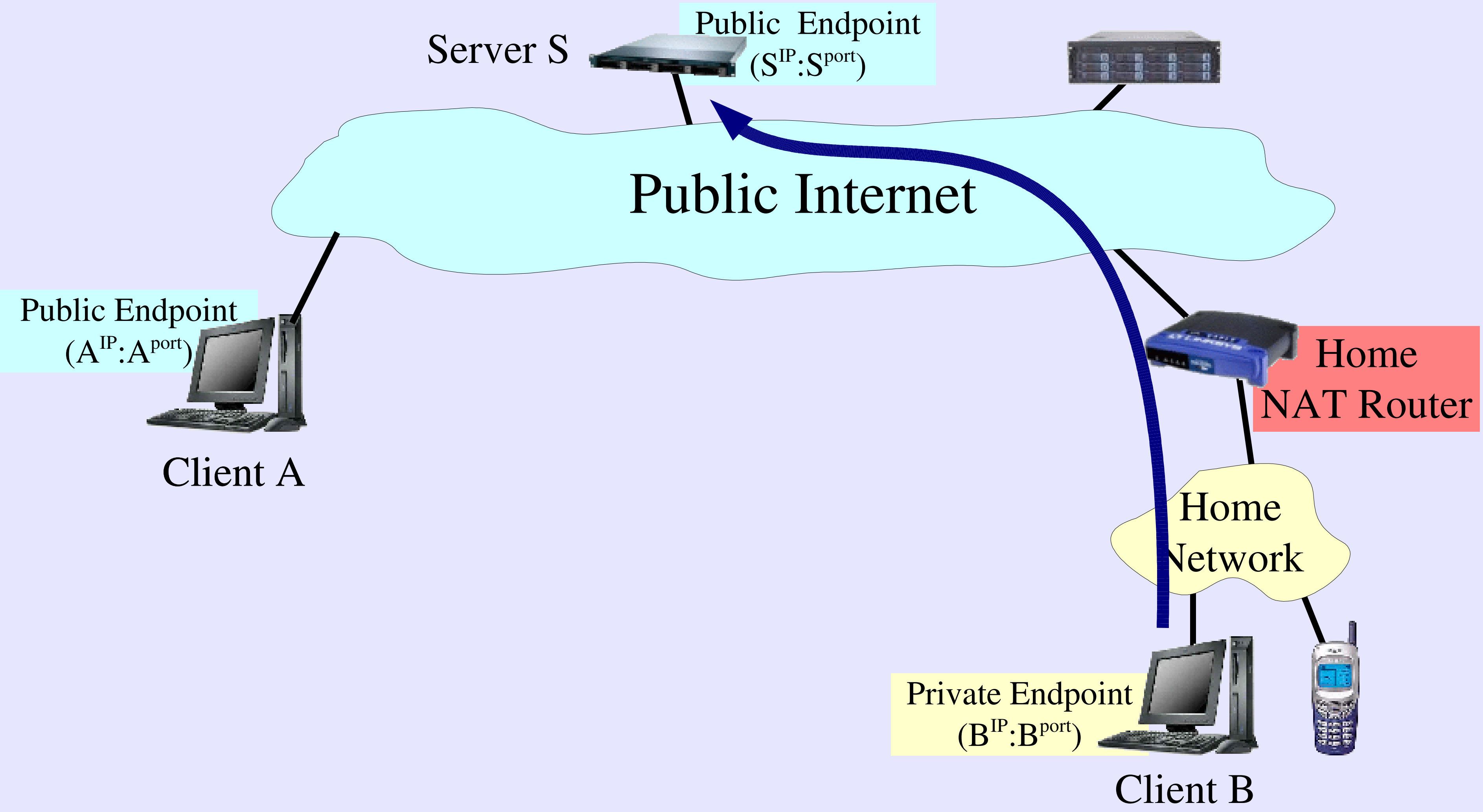
# Network Address Translation (NAT)



# Network Address Translation (NAT)



# Network Address Translation (NAT)



Server S

Public Endpoint
($S^{IP}$:$S^{port}$)

Public Internet

Public Endpoint
($A^{IP}$:$A^{port}$)

Client A

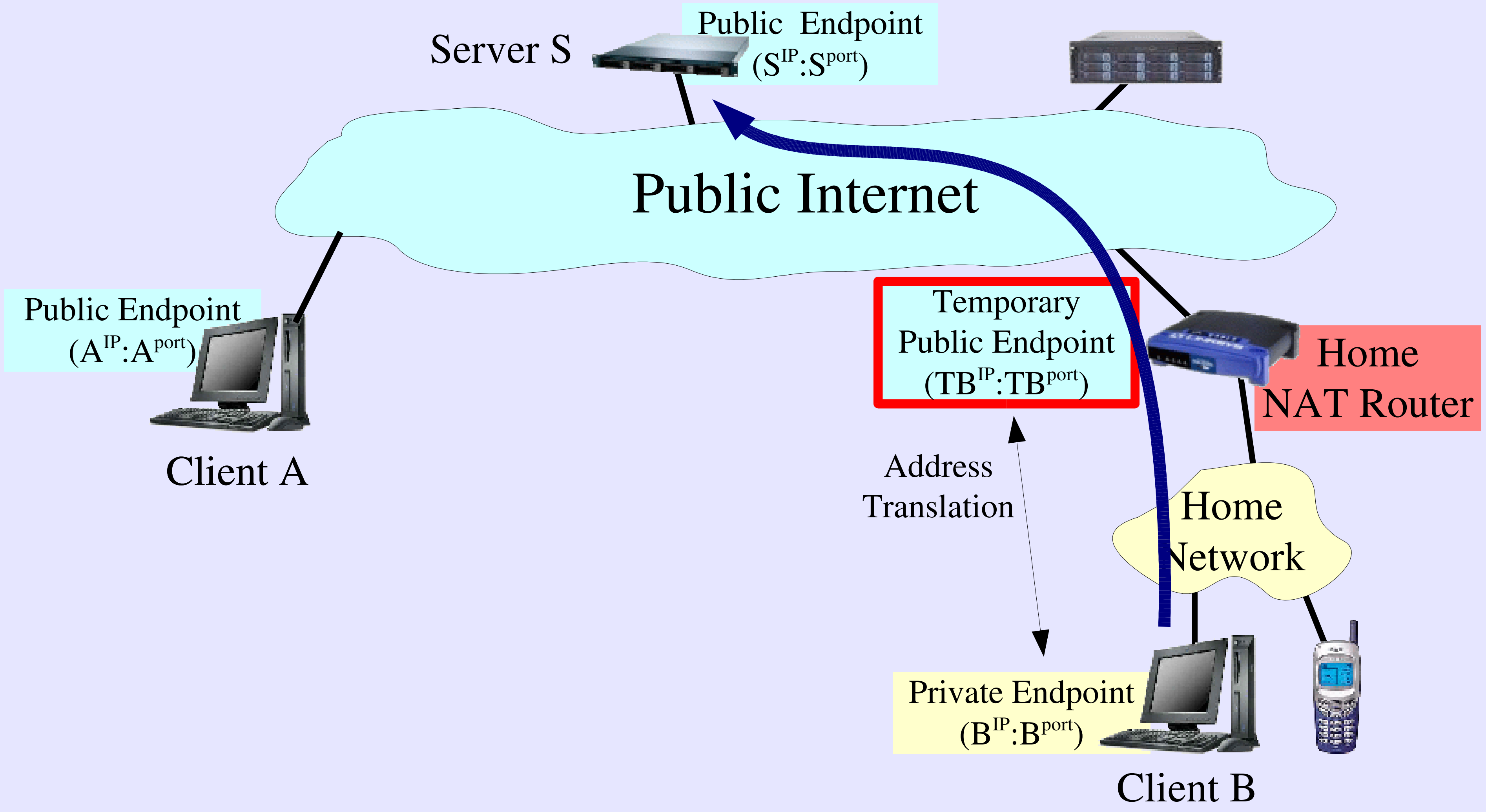
Home
NAT Router

Home
Network

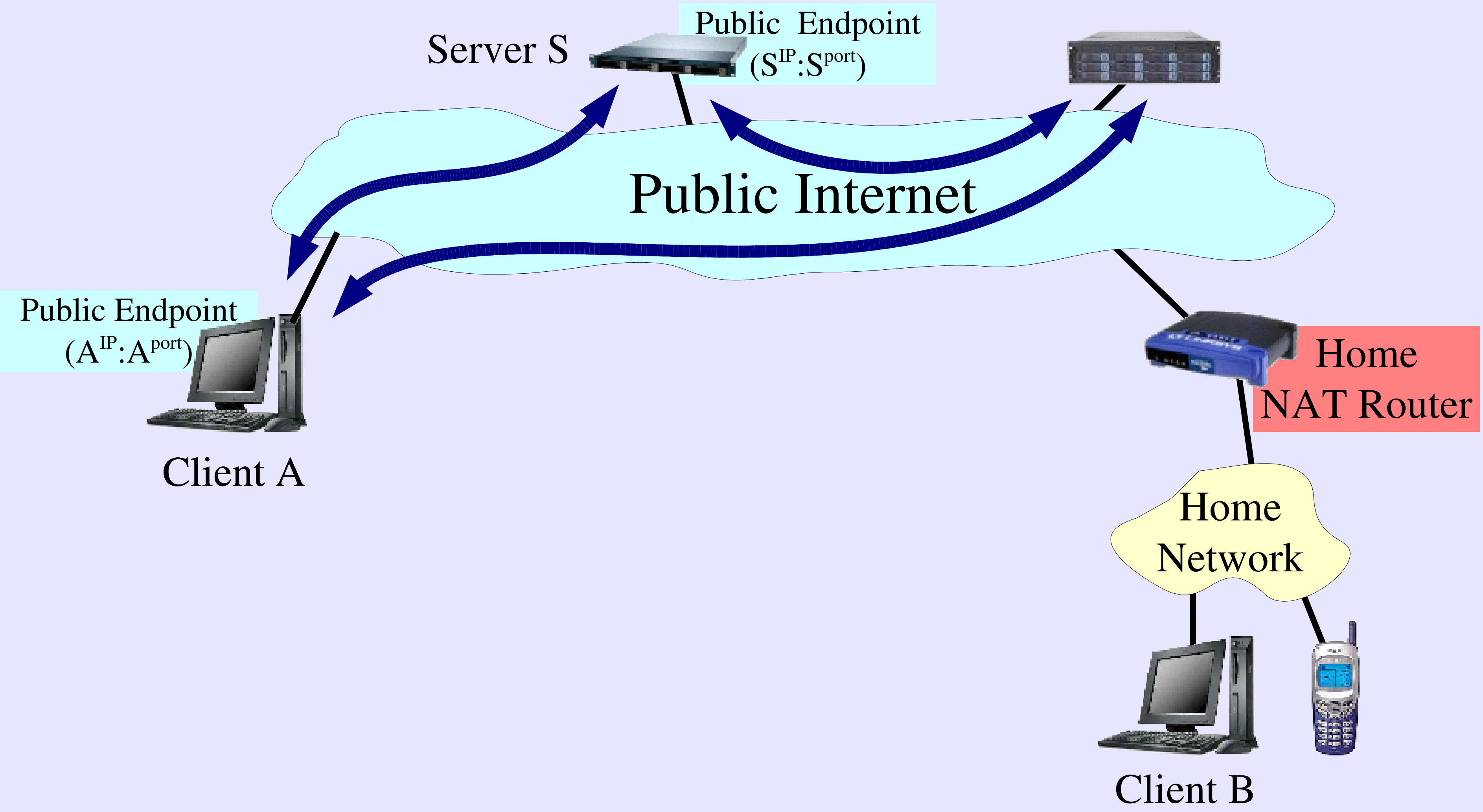
Private Endpoint
($B^{IP}$:$B^{port}$)

Client B

# Network Address Translation (NAT)



# Network Address Translation (NAT)



Server S

Public Endpoint
($S^{IP}$:$S^{port}$)

Public Internet

Public Endpoint
($A^{IP}$:$A^{port}$)

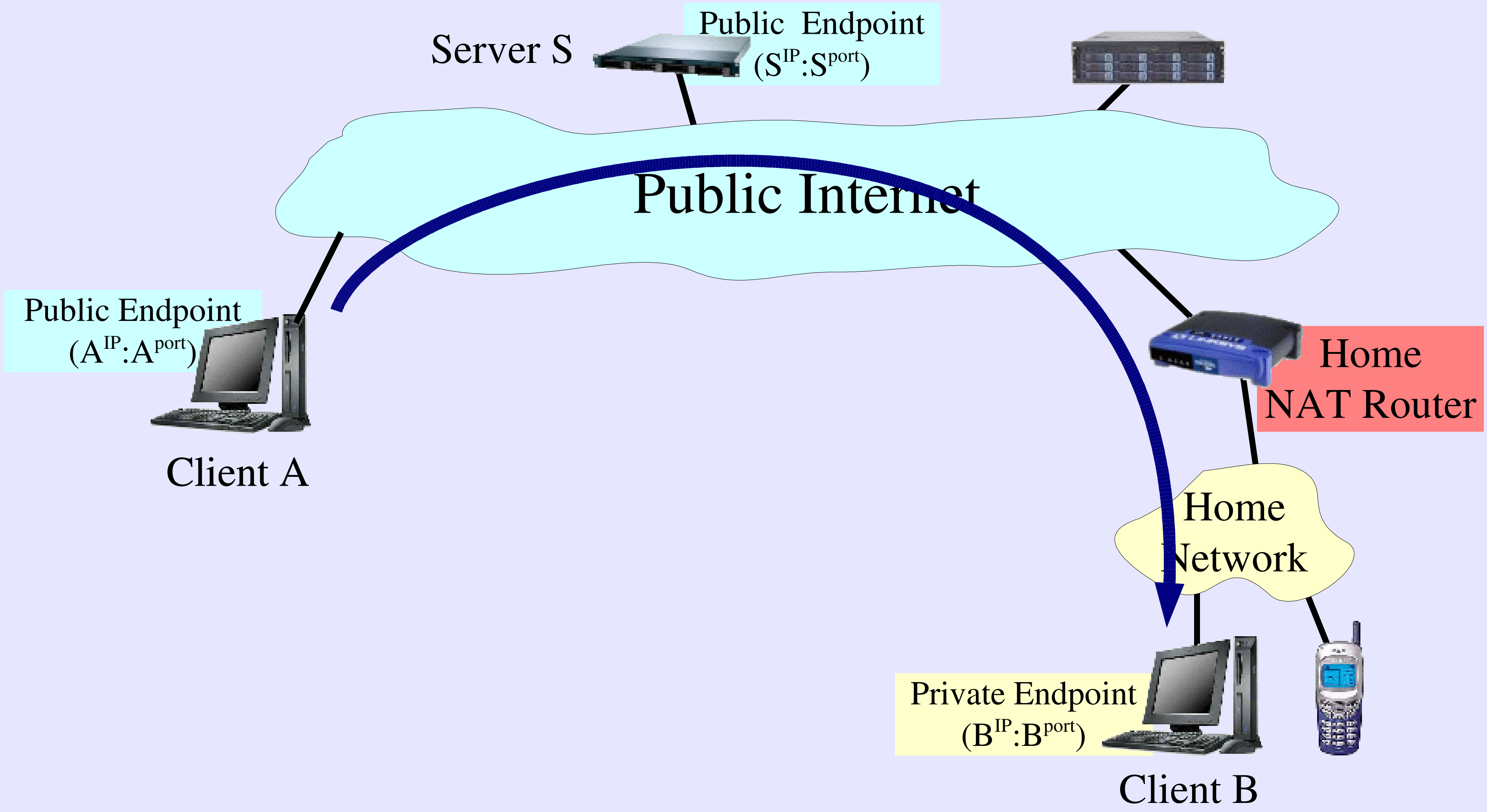
Client A

Home
NAT Router

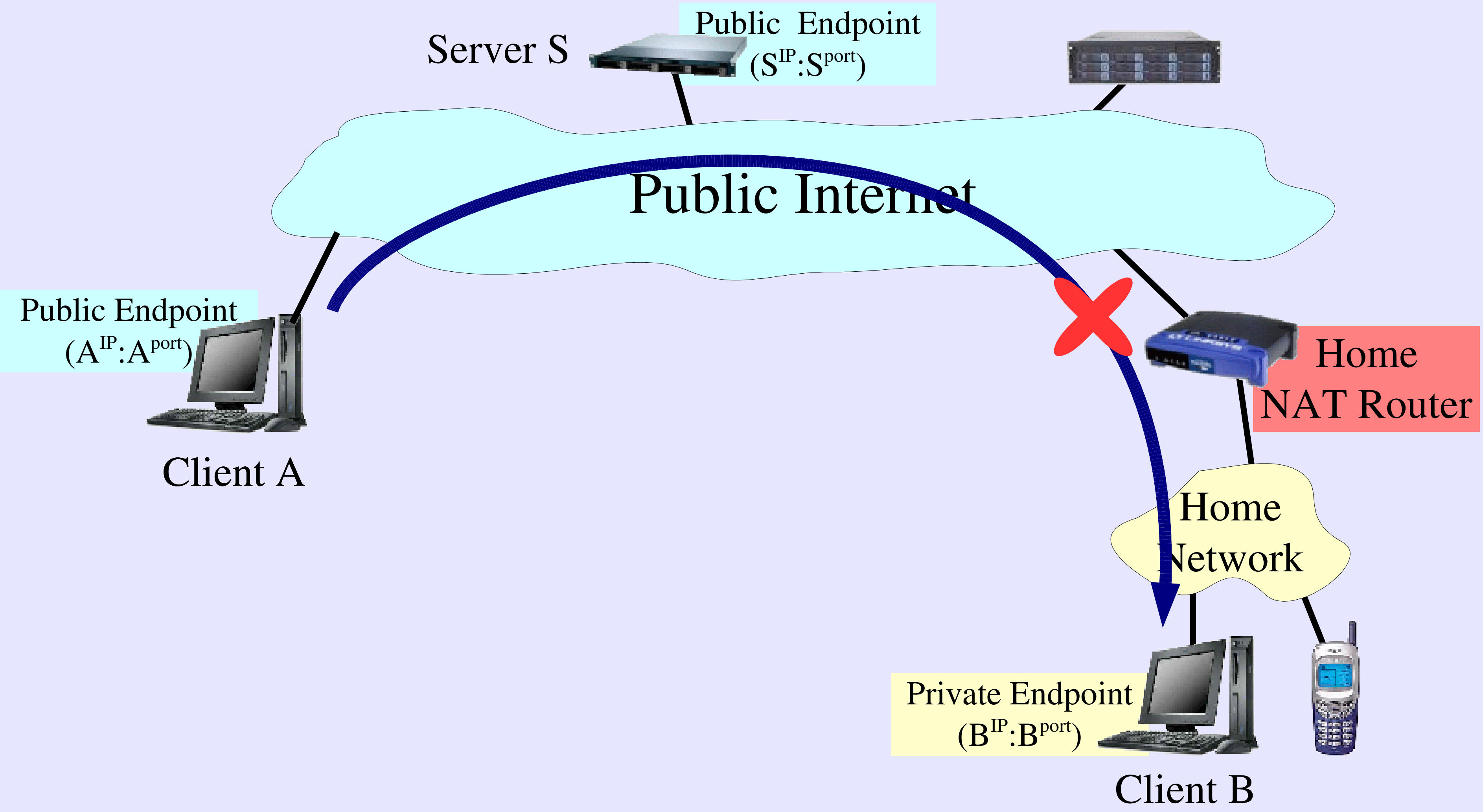
Home
Network

Client B

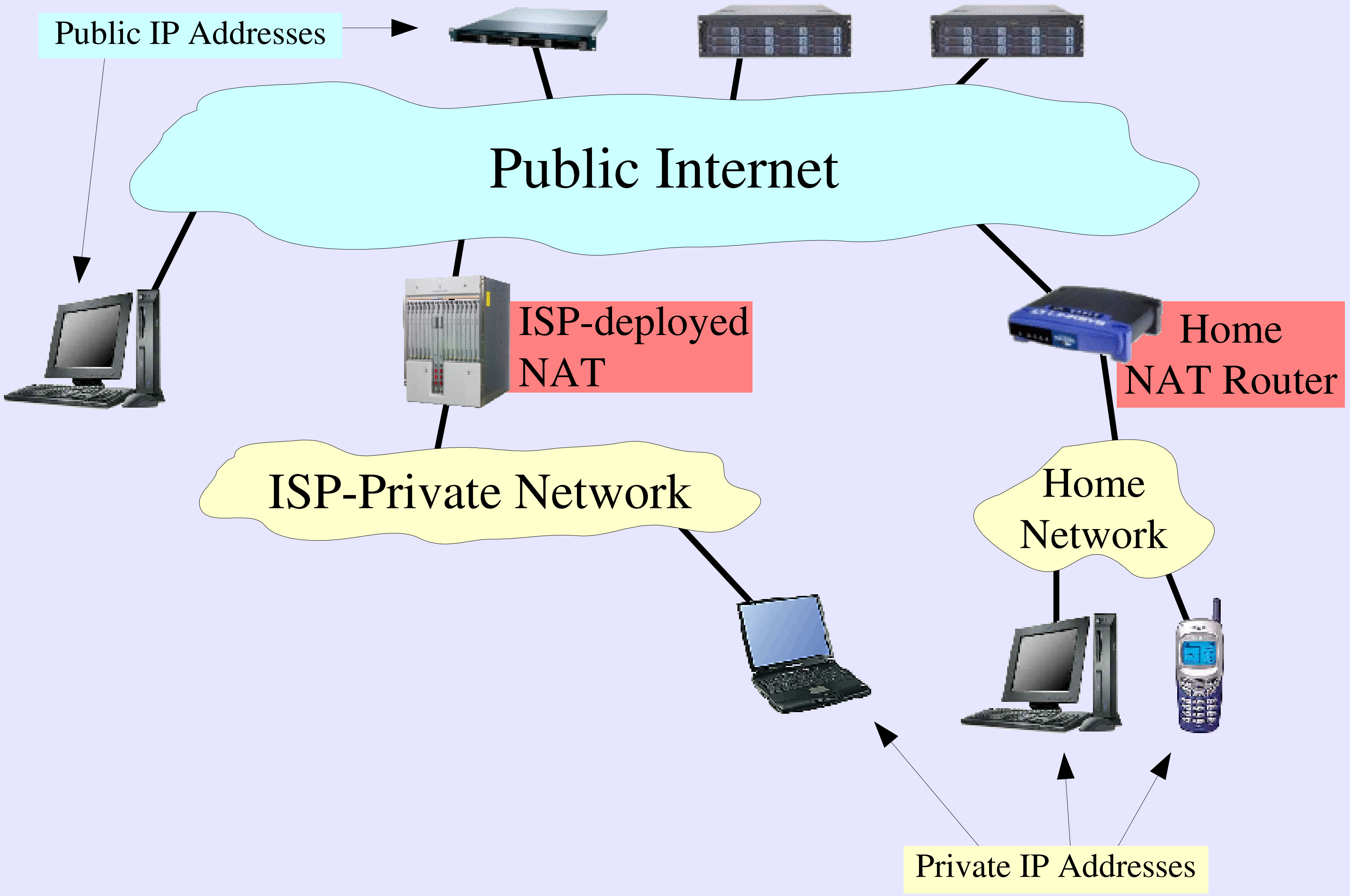
# Network Address Translation (NAT)



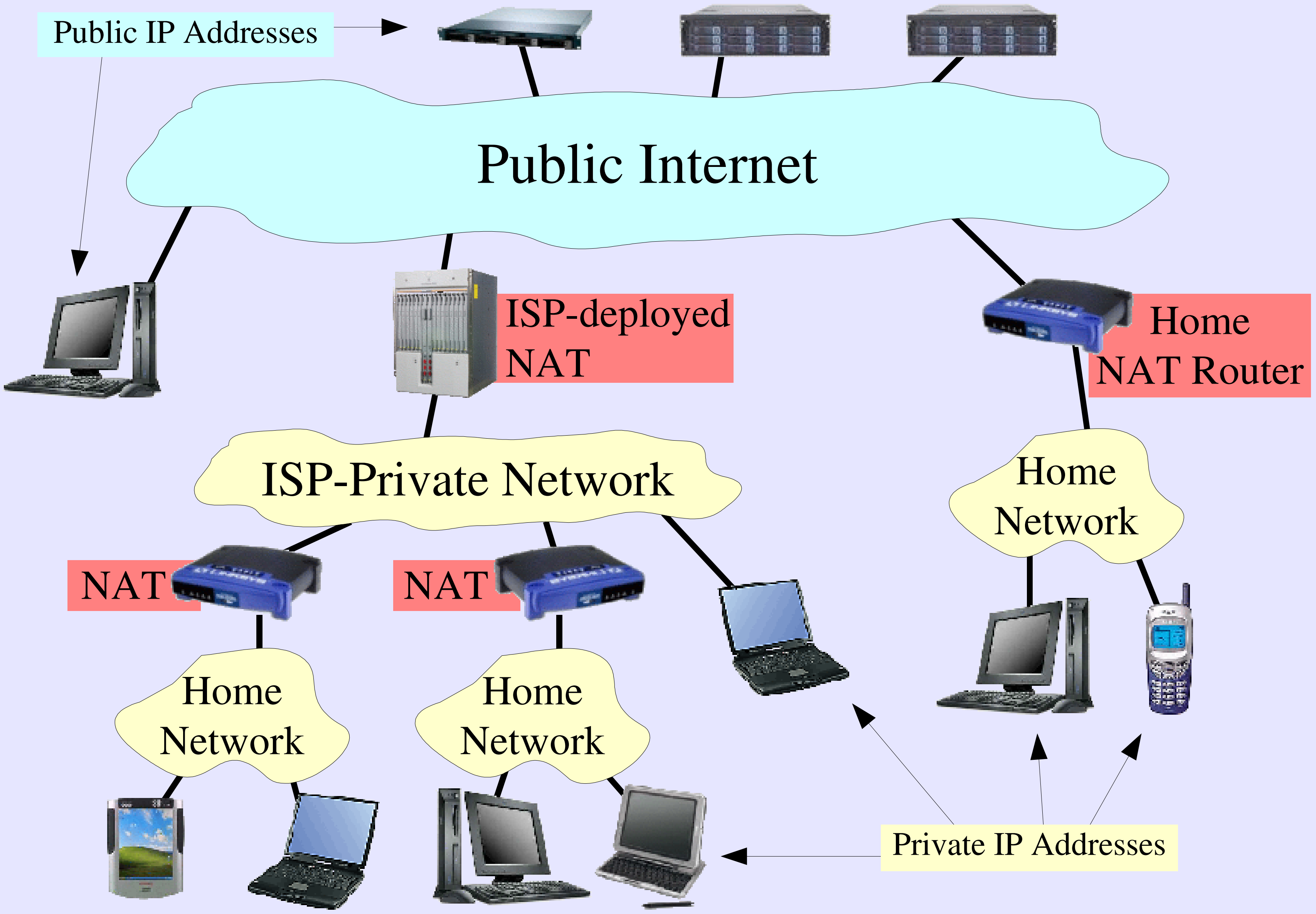
# Network Address Translation (NAT)



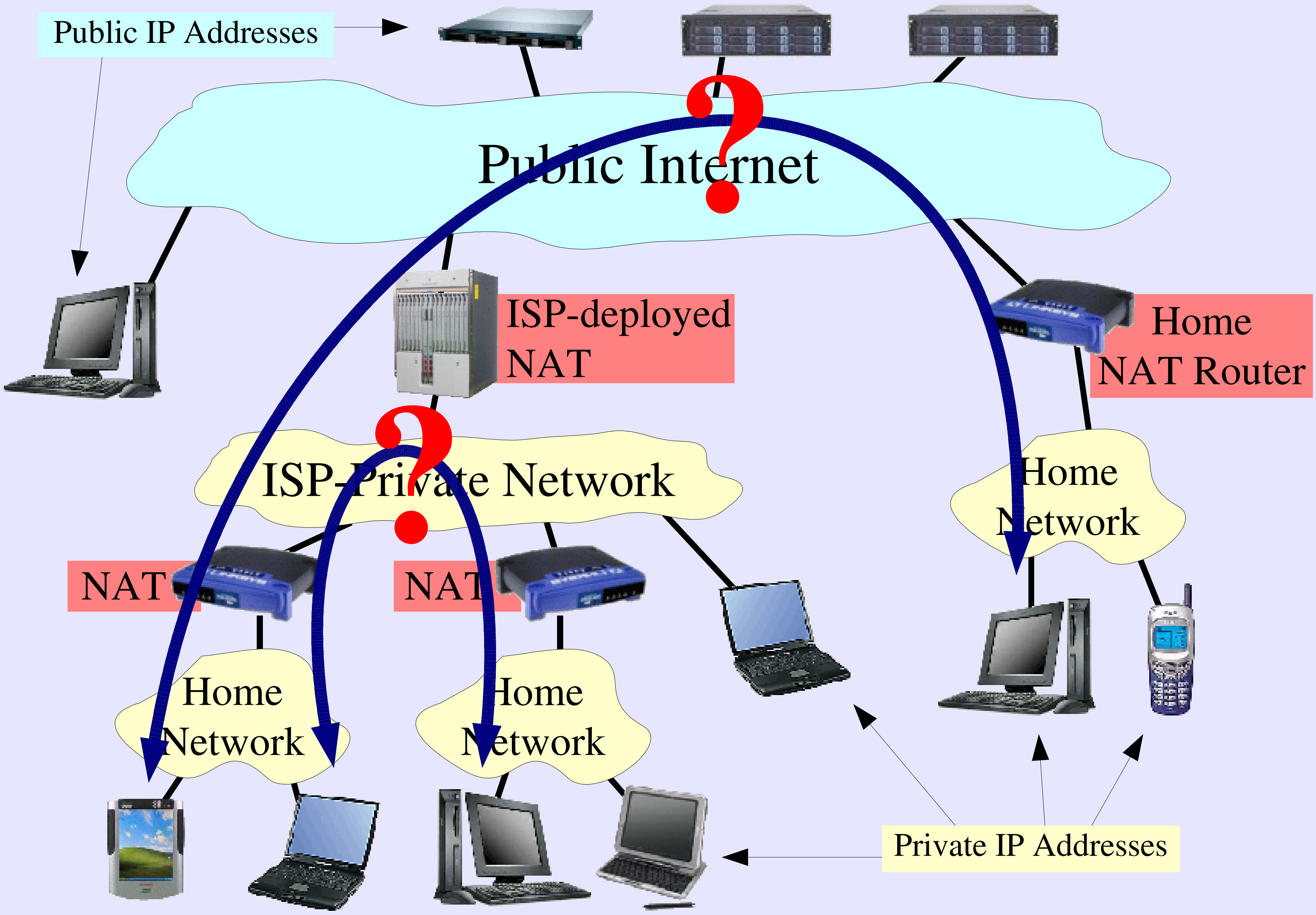
# Network Address Translation (NAT)



# Network Address Translation (NAT)



# Network Address Translation (NAT)



# Demand for P2P Communication

Many compelling apps need P2P communication, not just “P2P apps”:

- Teleconferencing, Voice over IP (VoIP)
- Multiplayer on-line games
- Remote access/administration (e.g., ssh)

# Outline

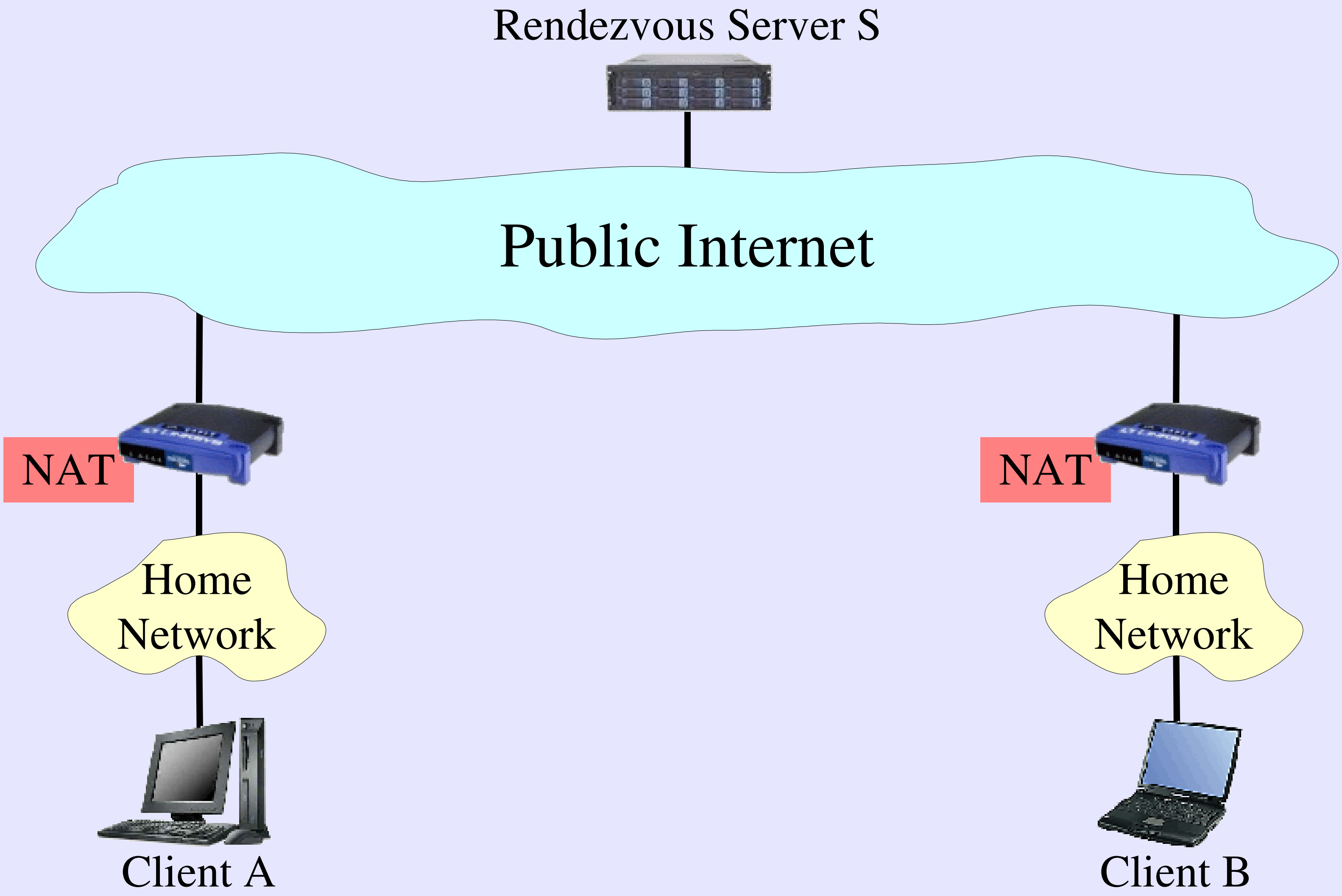
- The NAT Traversal Problem
- UDP Hole Punching (not new)
- TCP Hole Punching (quite new)
- Multi-Level NAT Scenarios
- NAT Compatibility with Hole Punching
- Related Work

# UDP Hole Punching

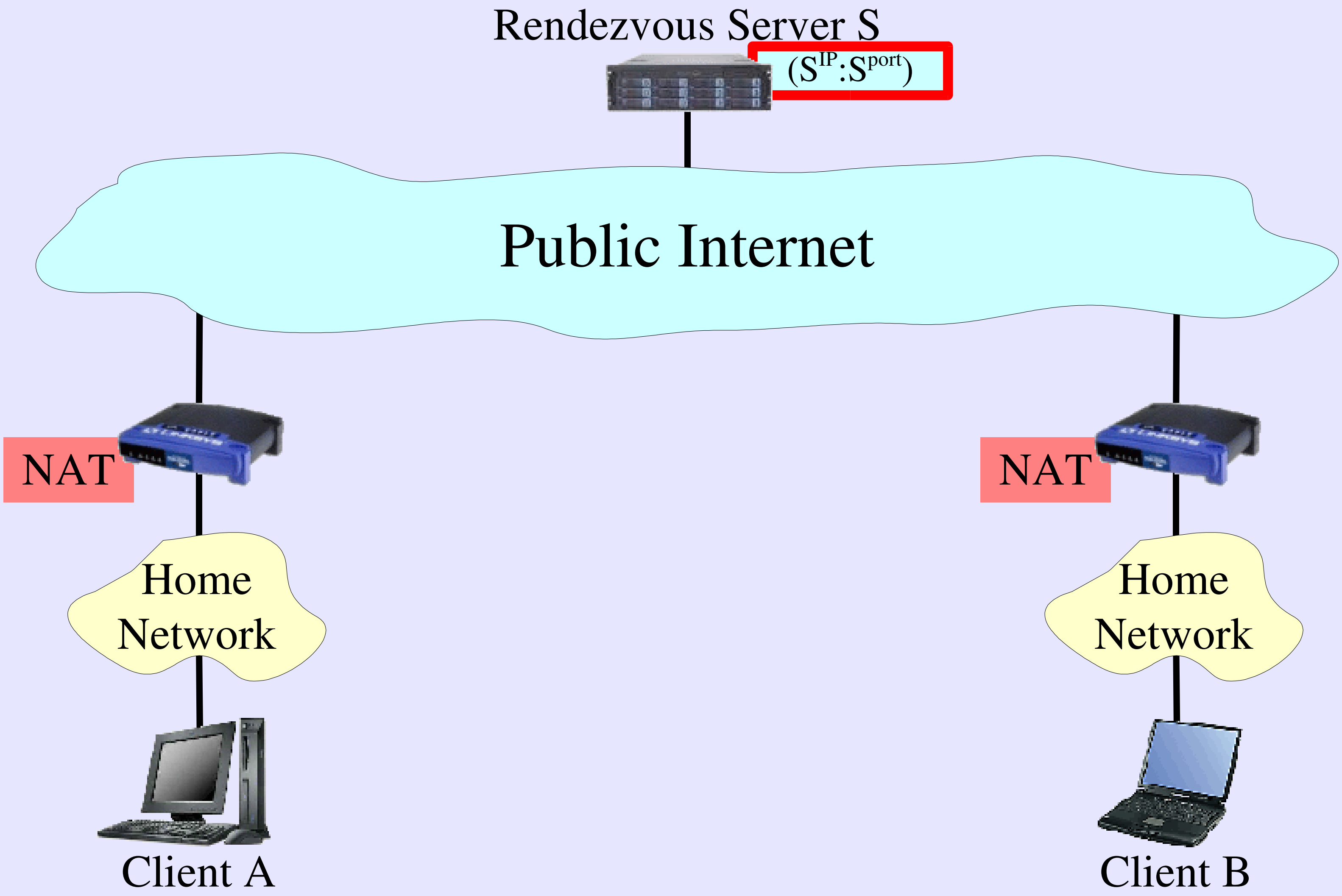
Usage model assumptions:

- Clients register with public “rendezvous server” to become accessible to other clients
- Application implements notion of “identity”
  - Username, public key [HIP], etc.
- Rendezvous server facilitates P2P session setup, but does not participate in resulting P2P sessions

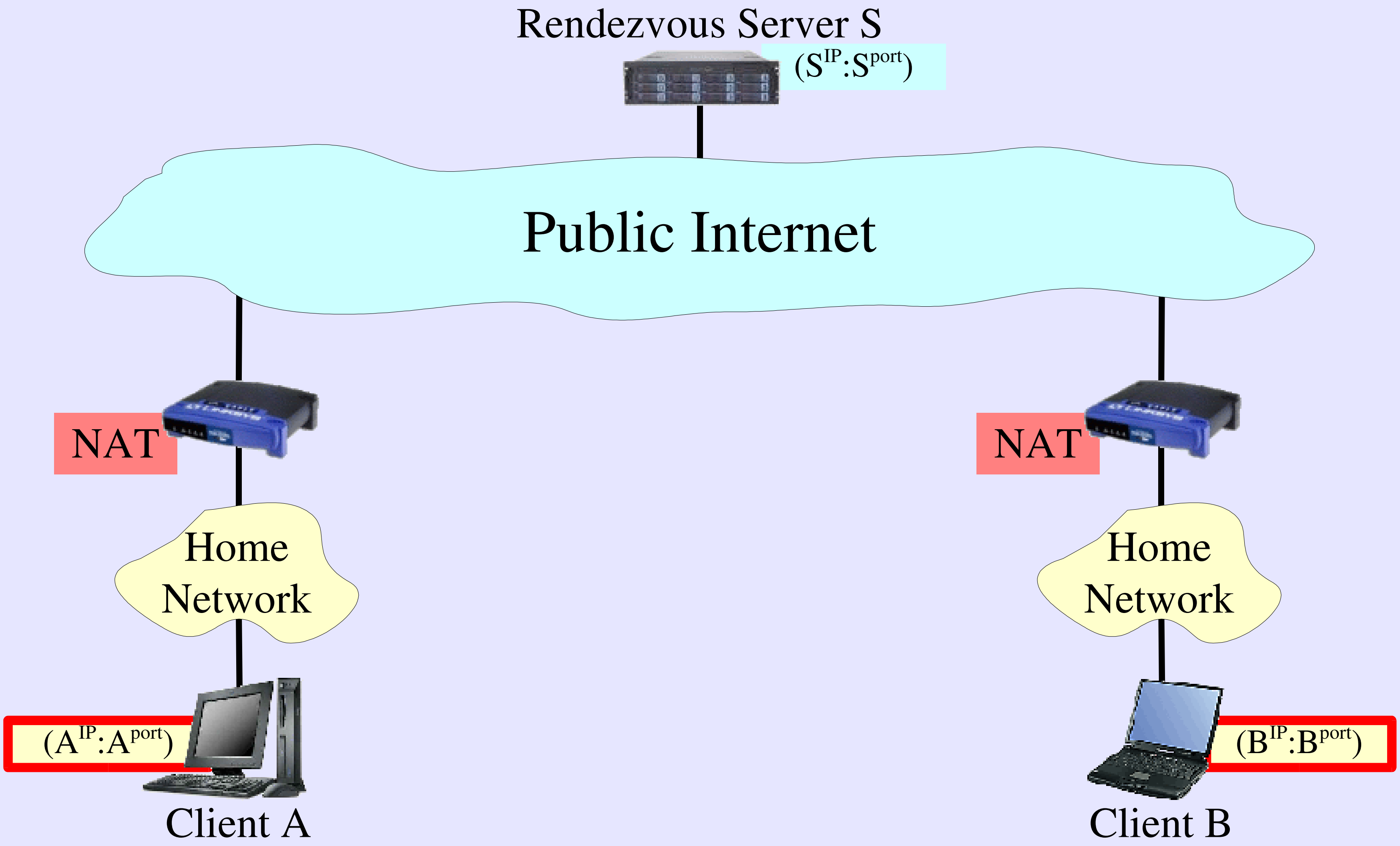
# UDP Hole Punching



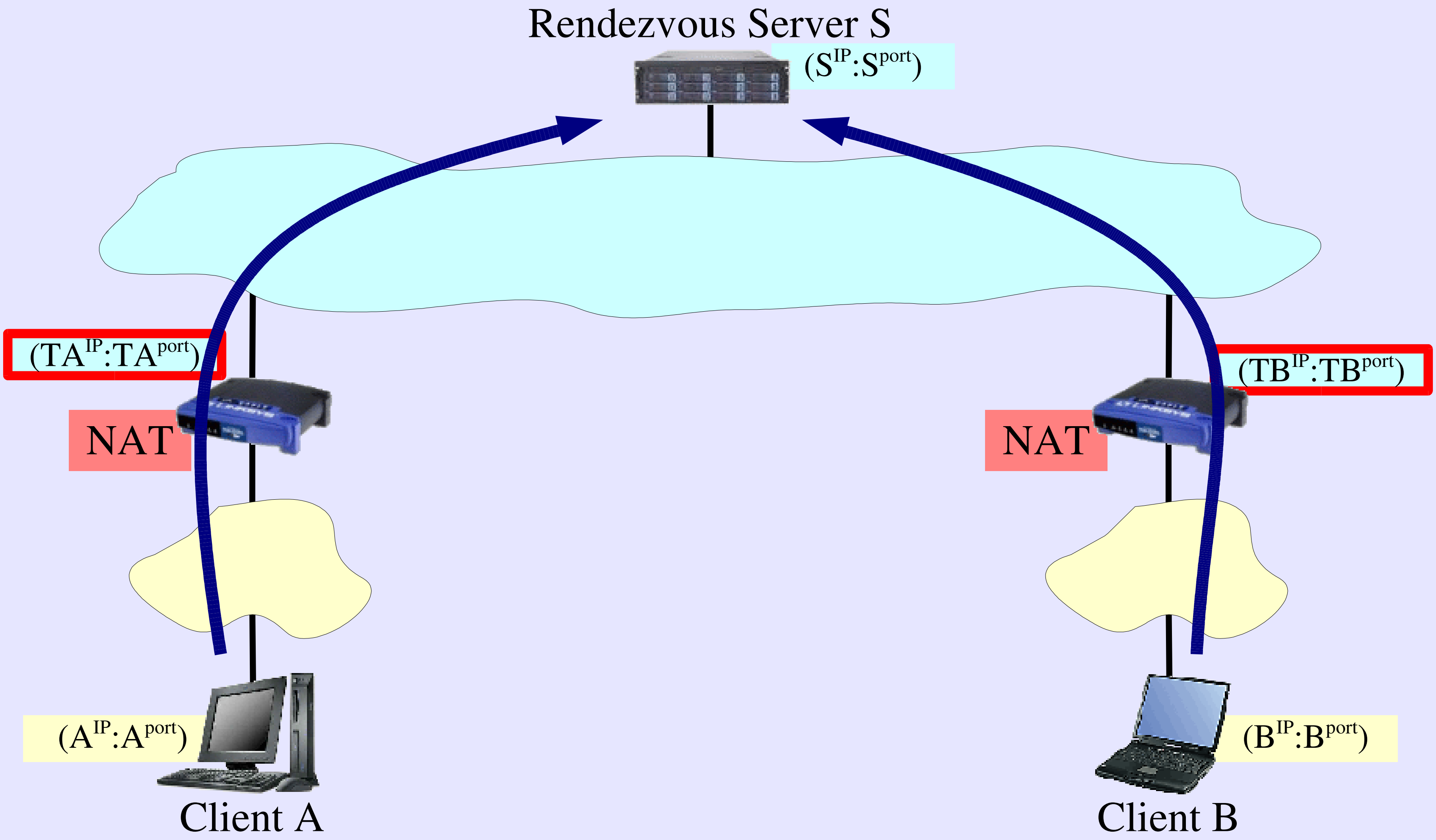
# UDP Hole Punching



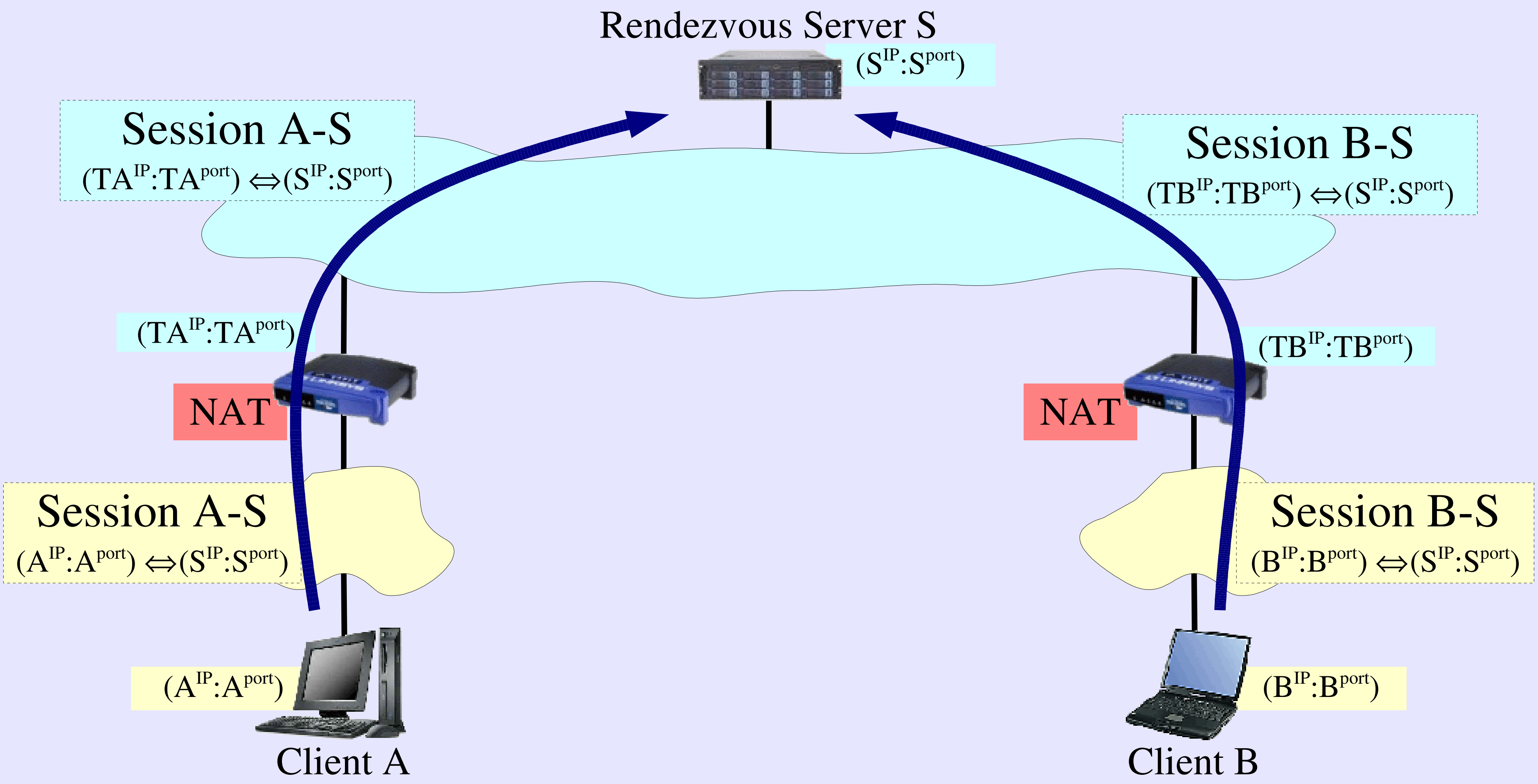
# UDP Hole Punching



# UDP Hole Punching



# UDP Hole Punching



Rendezvous Server S

$(S^{IP}:S^{port})$

Session A-S

$(TA^{IP}:TA^{port}) \Leftrightarrow (S^{IP}:S^{port})$

Session B-S

$(TB^{IP}:TB^{port}) \Leftrightarrow (S^{IP}:S^{port})$

$(TA^{IP}:TA^{port})$

$(TB^{IP}:TB^{port})$

NAT

NAT

Session A-S

$(A^{IP}:A^{port}) \Leftrightarrow (S^{IP}:S^{port})$

Session B-S

$(B^{IP}:B^{port}) \Leftrightarrow (S^{IP}:S^{port})$

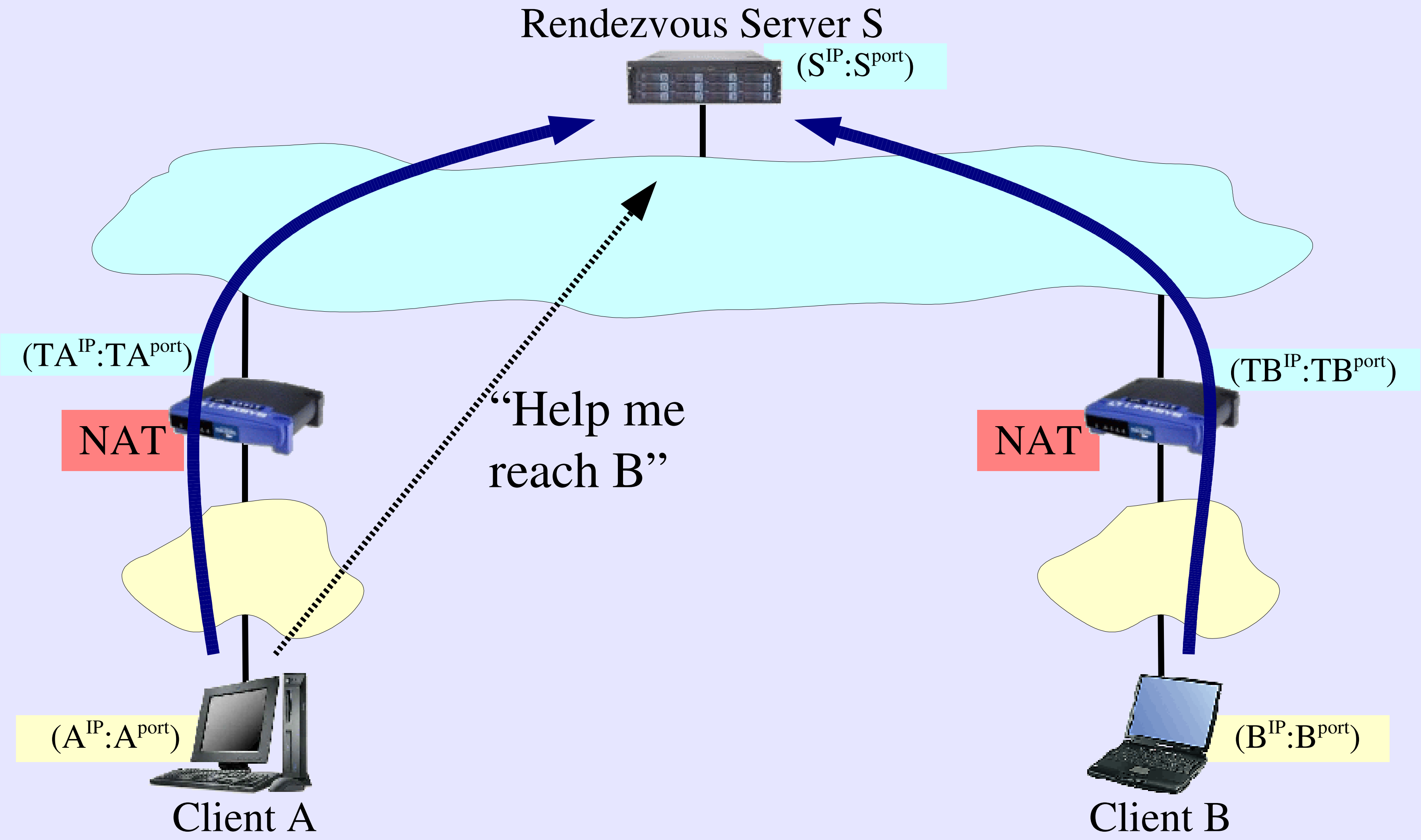
$(A^{IP}:A^{port})$

$(B^{IP}:B^{port})$

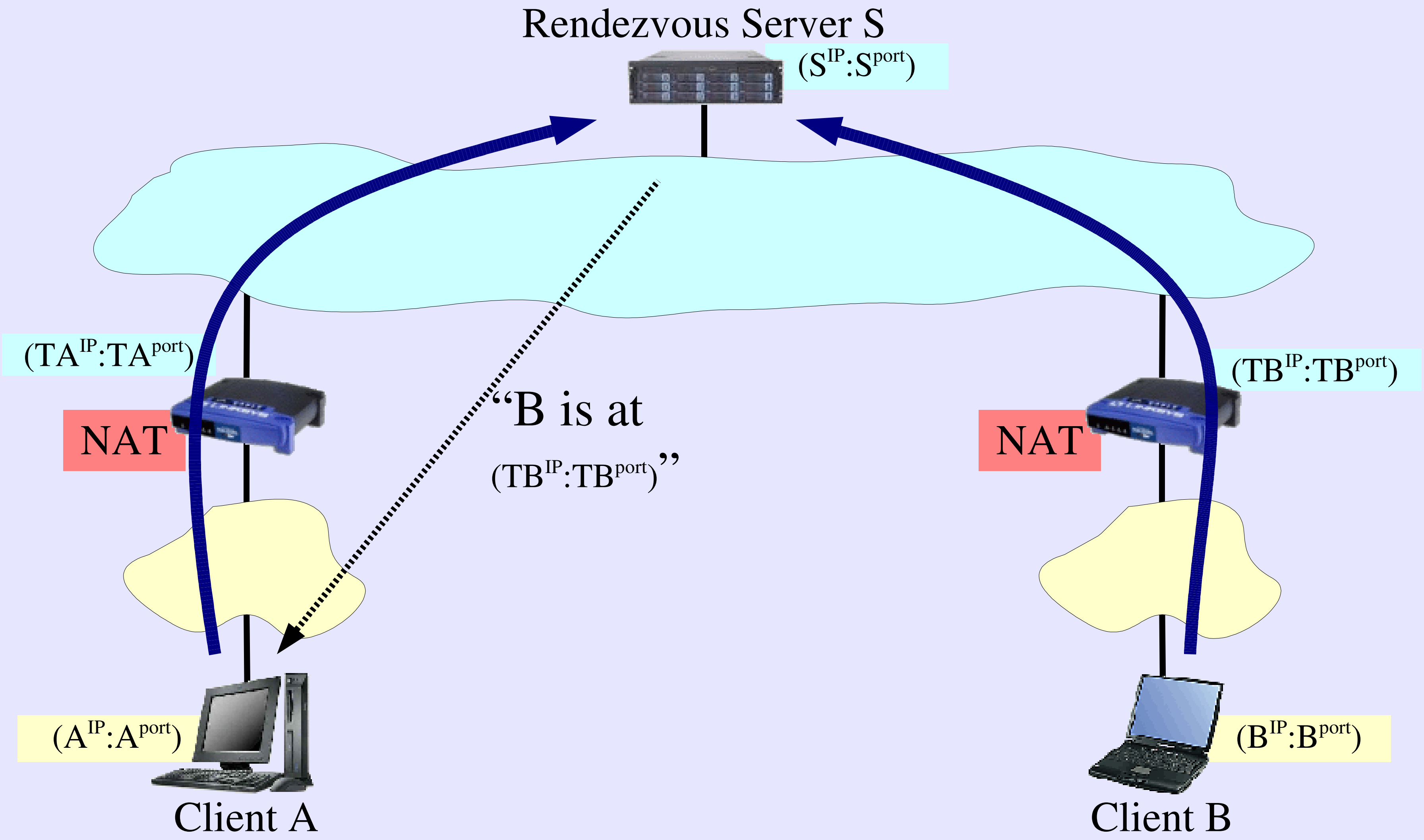
Client A

Client B

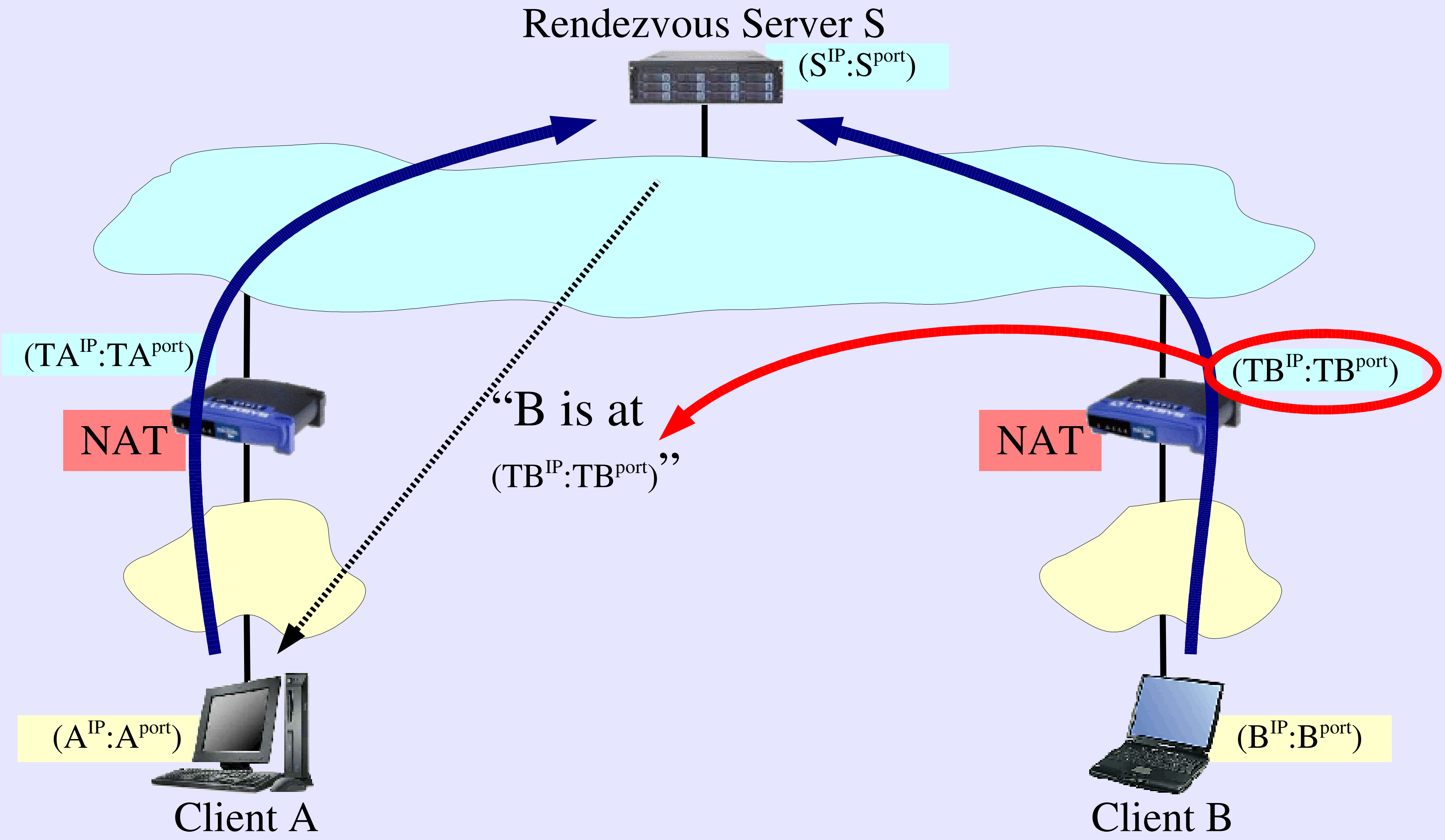
# UDP Hole Punching



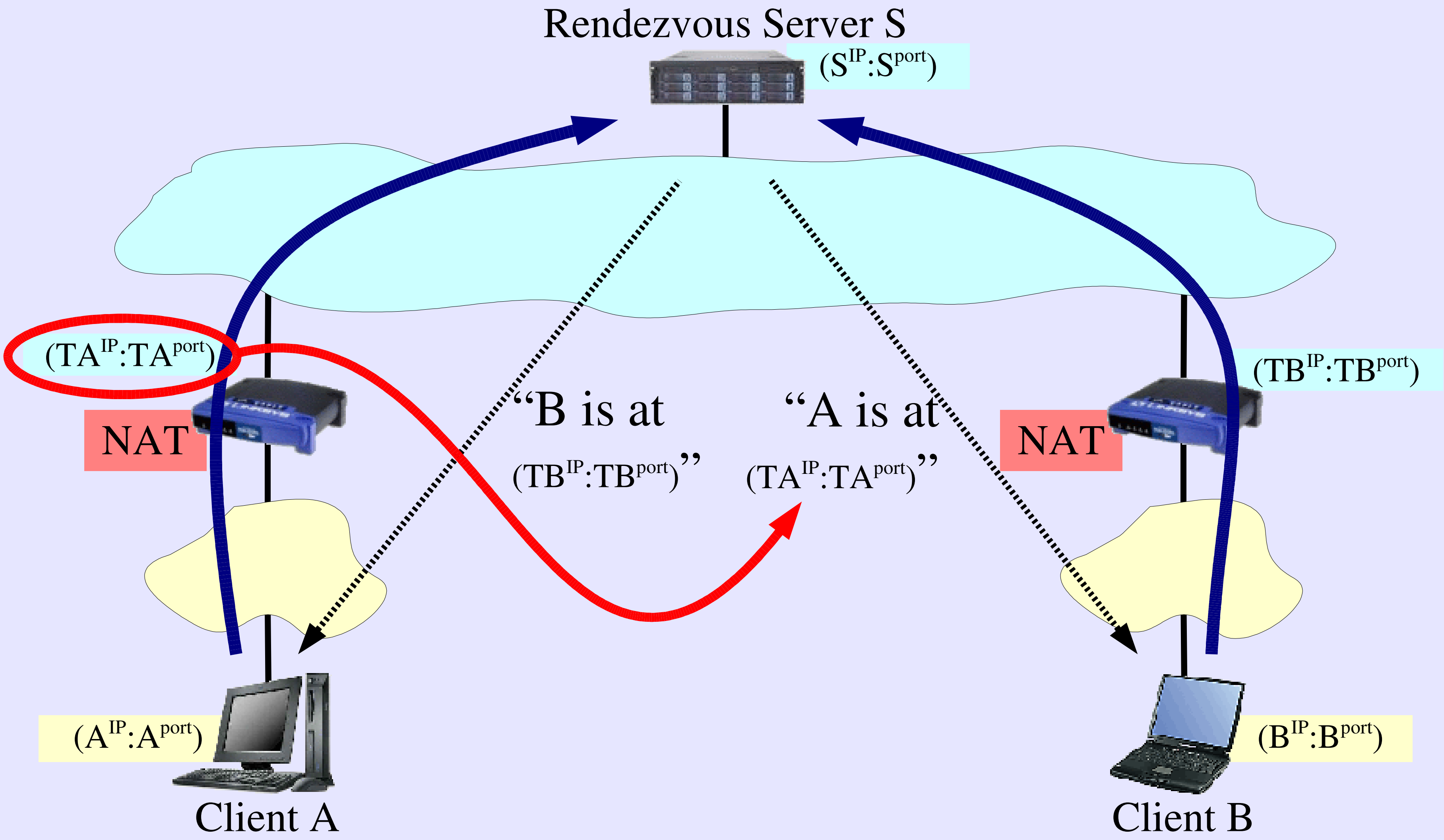
# UDP Hole Punching



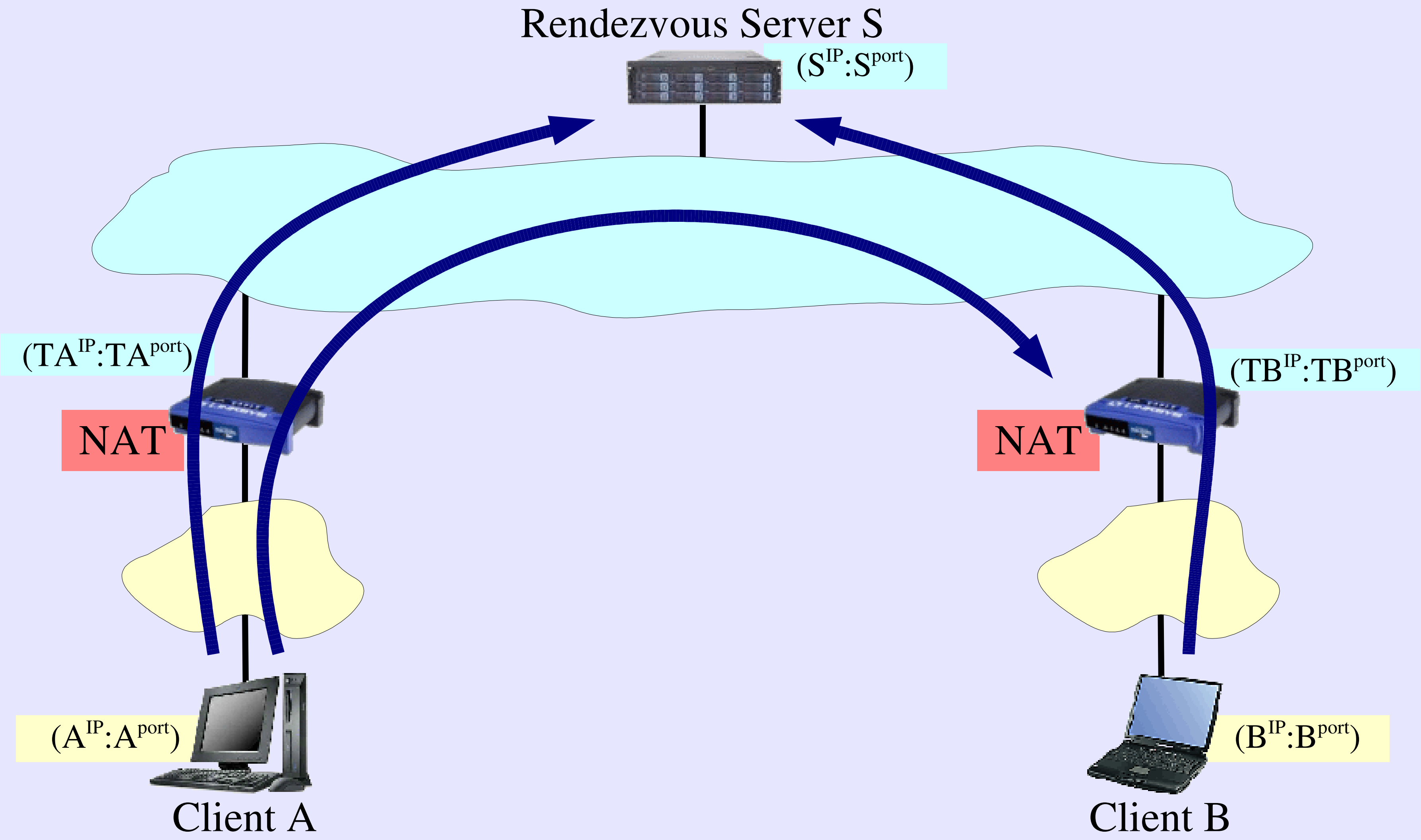
# UDP Hole Punching



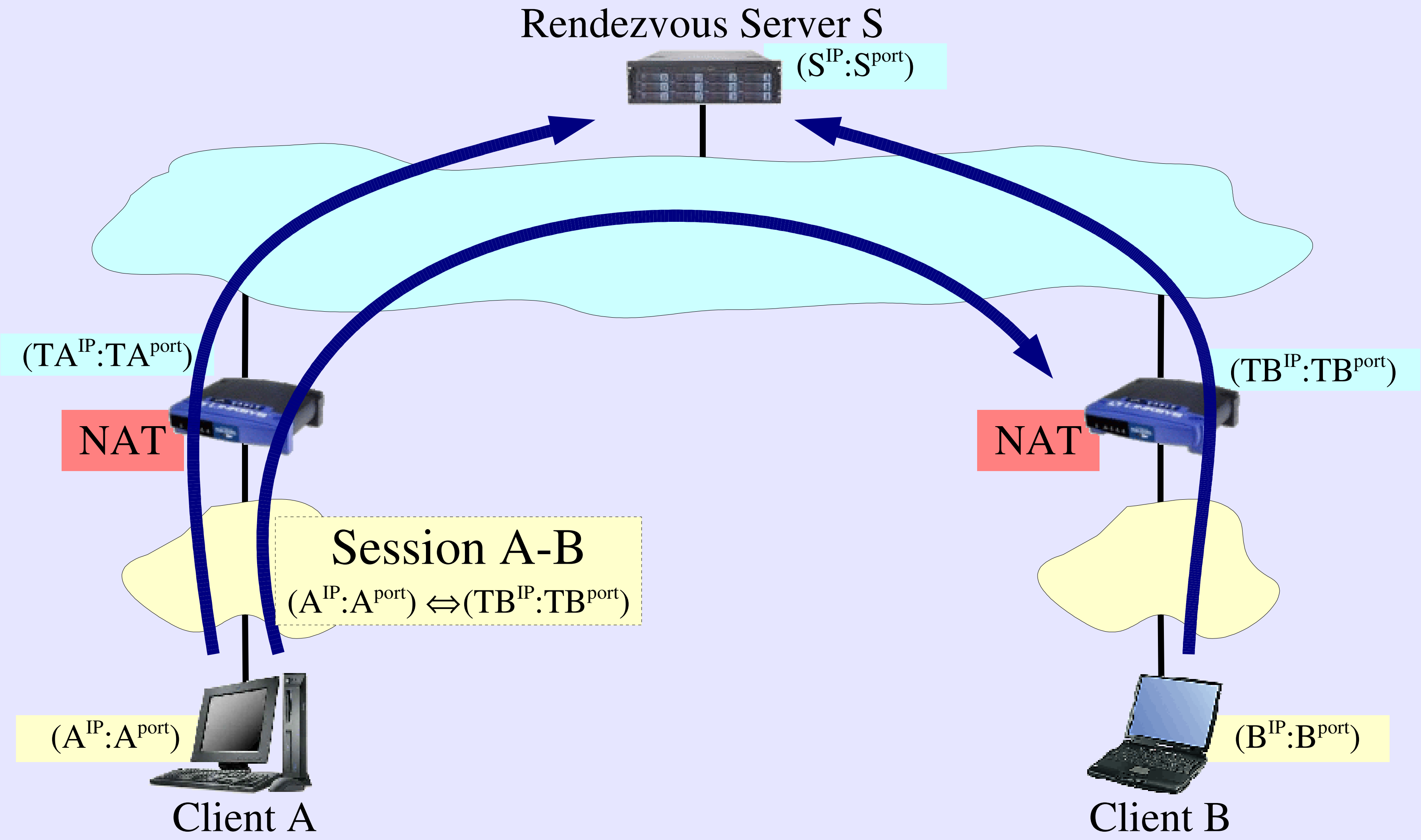
# UDP Hole Punching



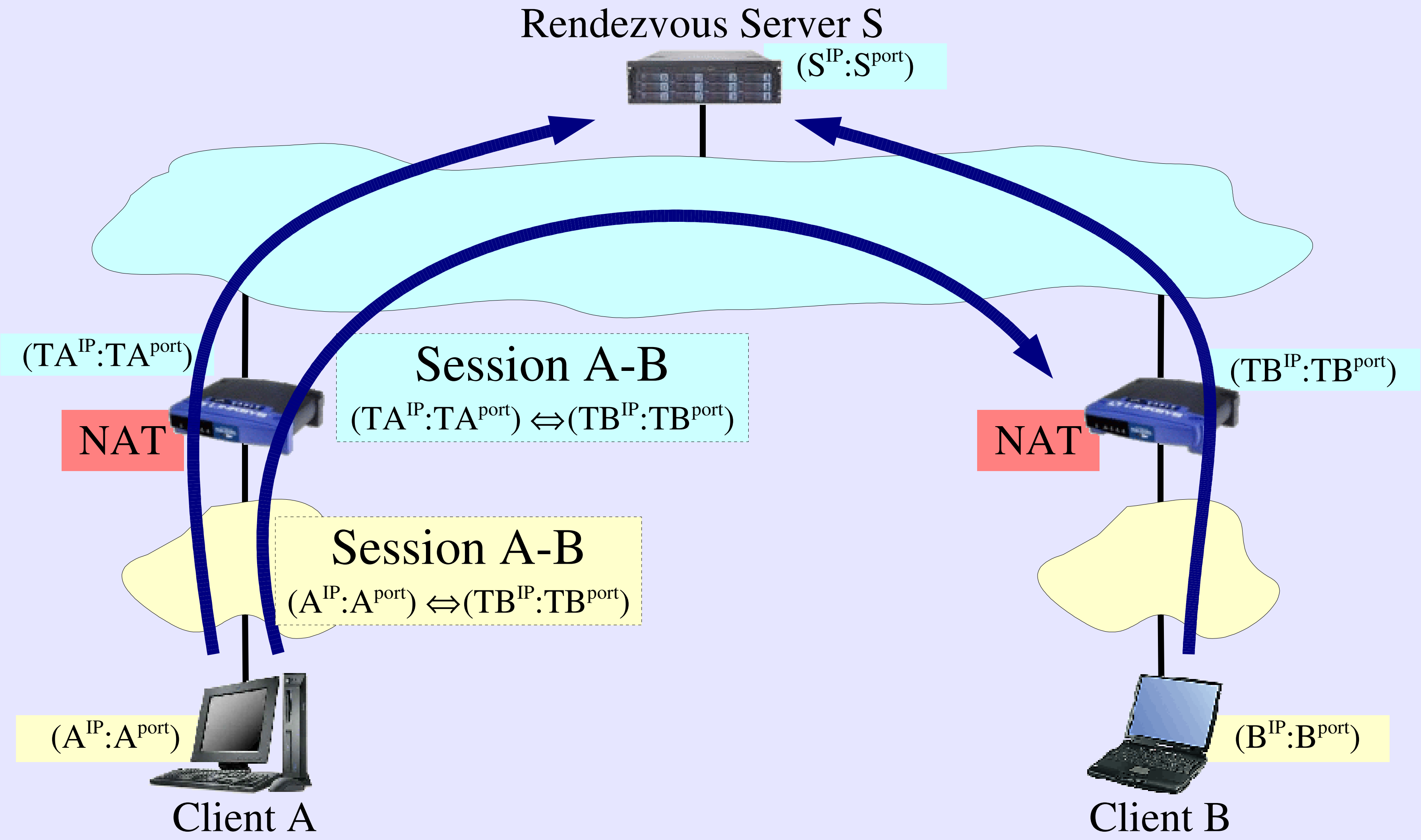
# UDP Hole Punching



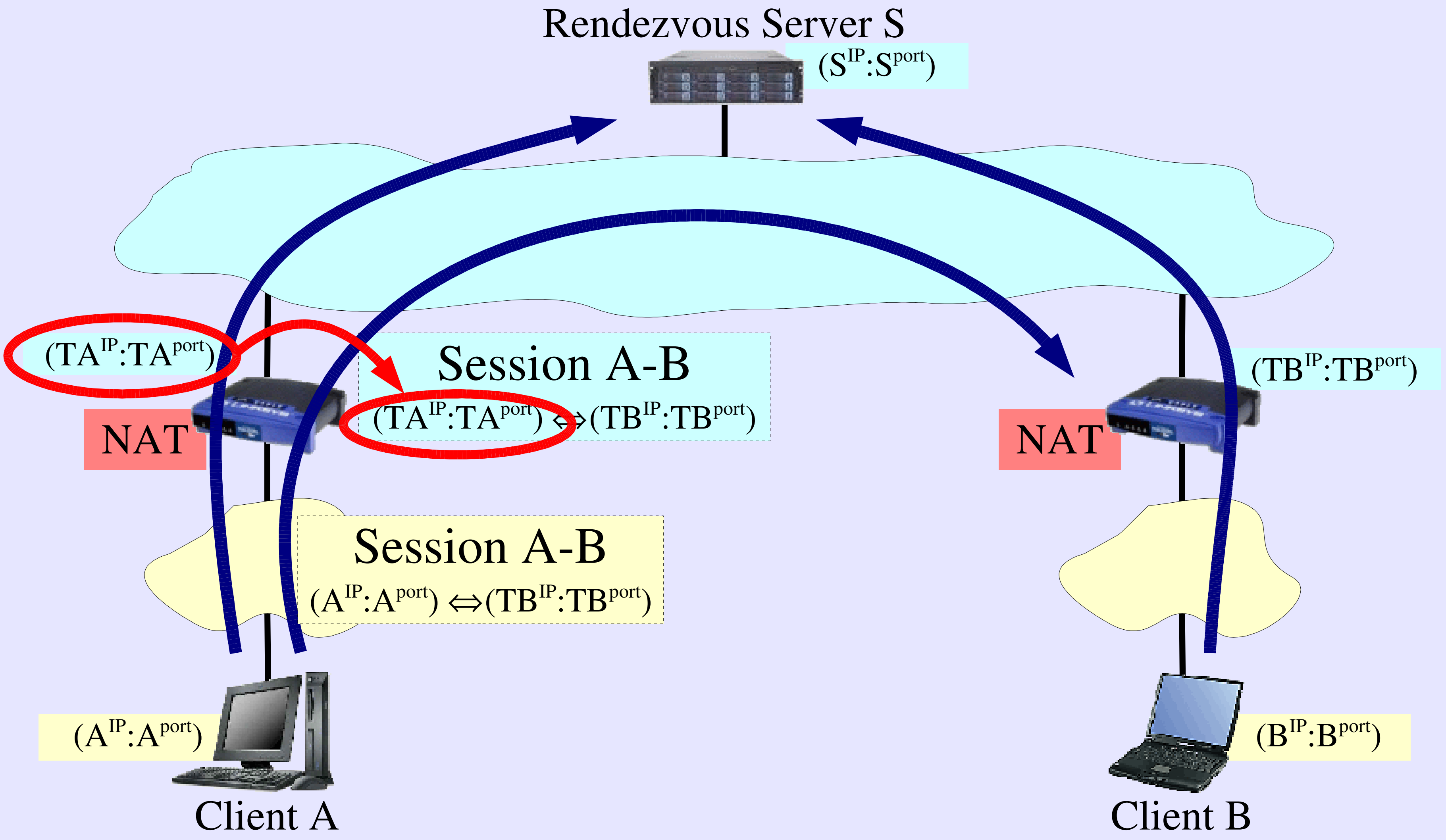
# UDP Hole Punching



# UDP Hole Punching



# UDP Hole Punching



Rendezvous Server S

$(S^{IP}:S^{port})$

$(TA^{IP}:TA^{port})$

NAT

Session A-B

$(TA^{IP}:TA^{port}) \Leftrightarrow (TB^{IP}:TB^{port})$

Session A-B

$(A^{IP}:A^{port}) \Leftrightarrow (TB^{IP}:TB^{port})$

$(A^{IP}:A^{port})$

Client A

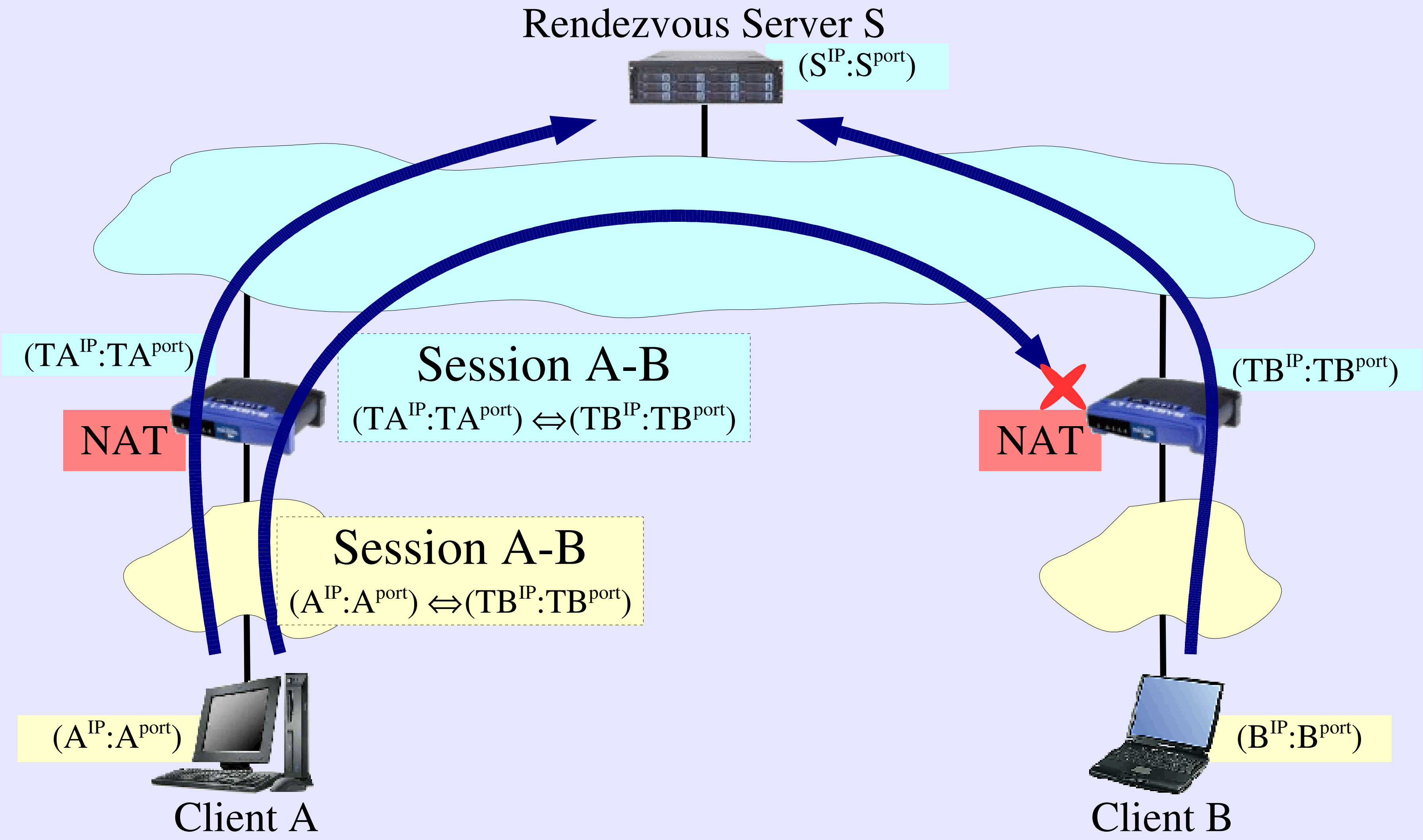
$(TB^{IP}:TB^{port})$

NAT

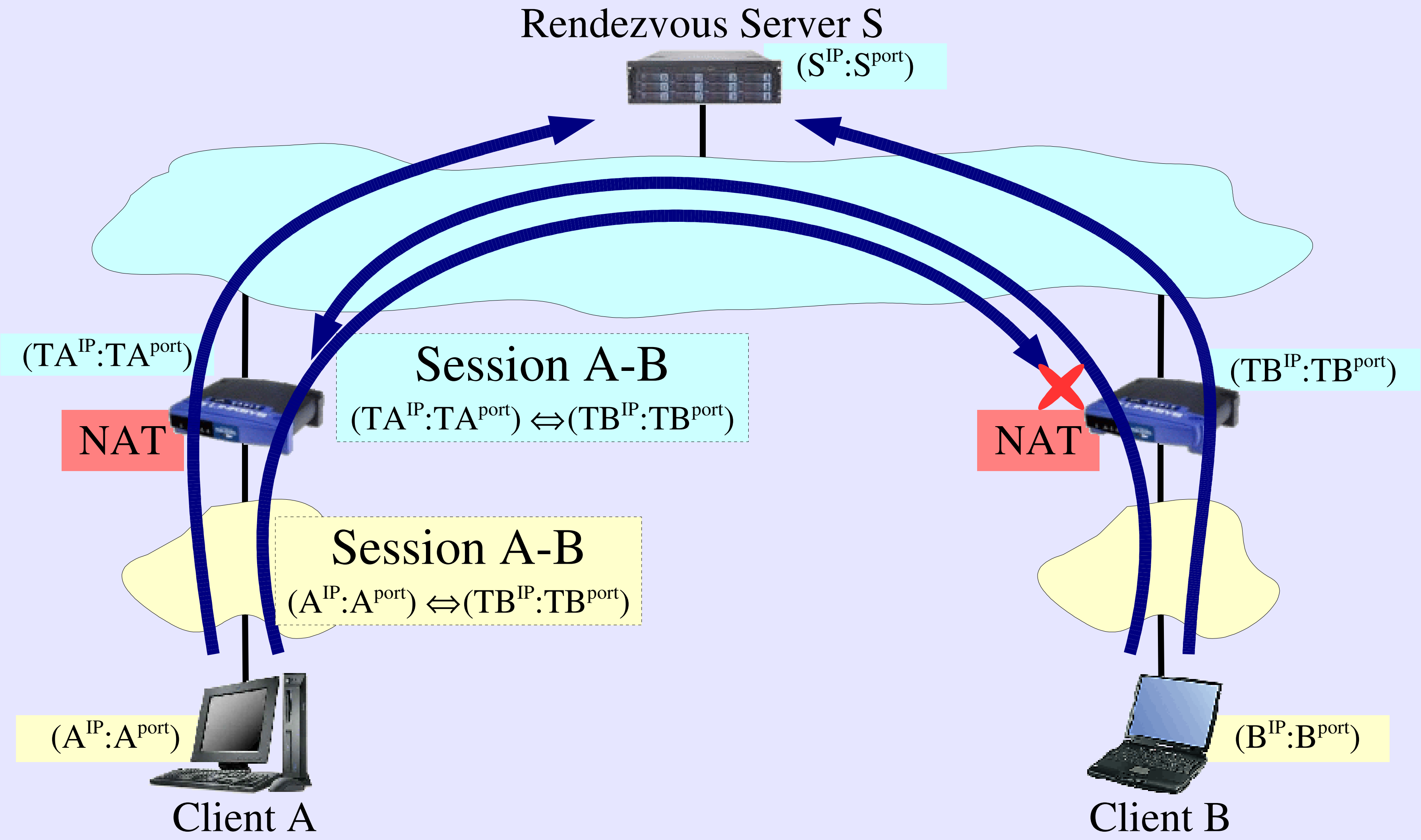
$(B^{IP}:B^{port})$

Client B

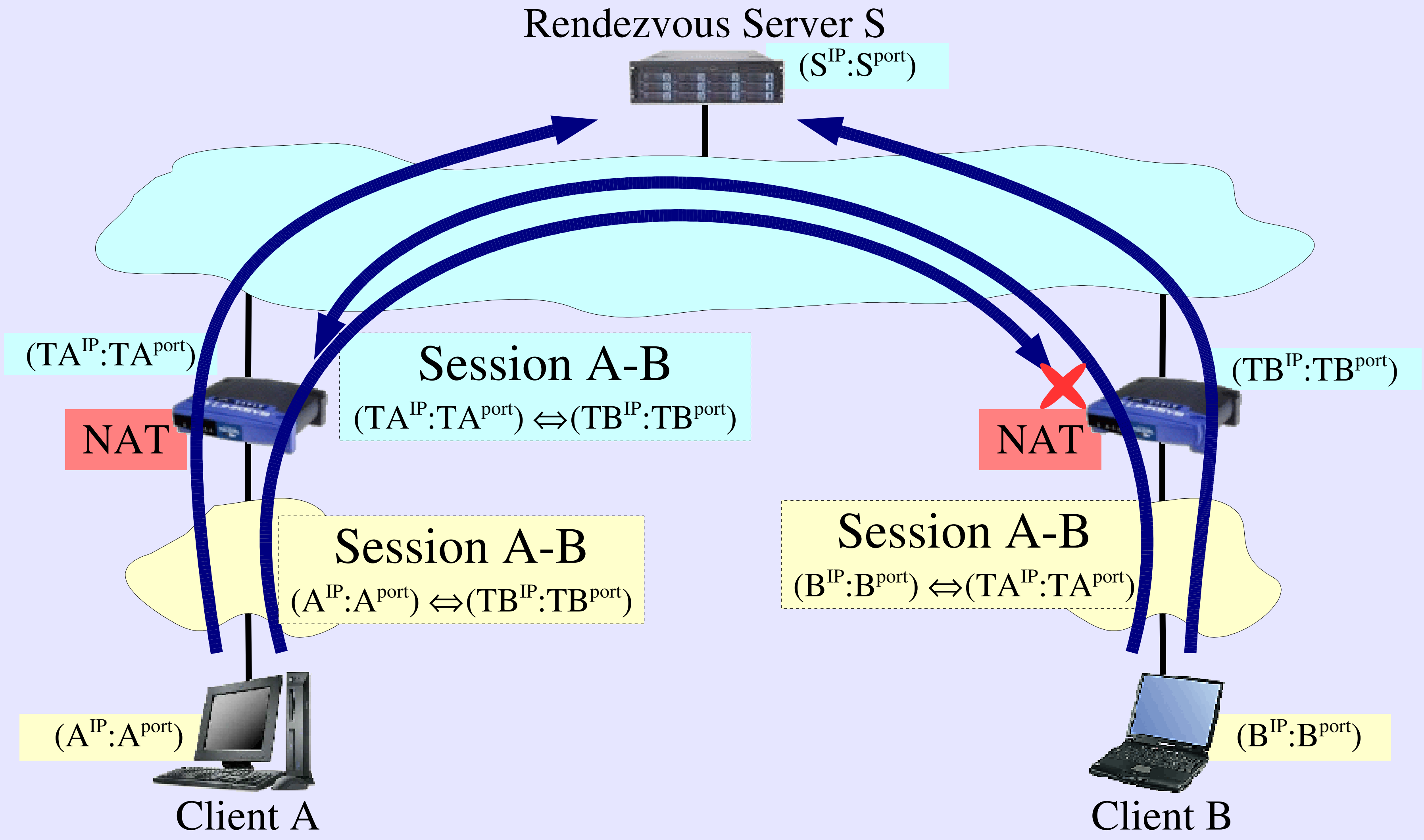
# UDP Hole Punching



# UDP Hole Punching



# UDP Hole Punching



Rendezvous Server S

$(S^{IP}:S^{port})$

$(TA^{IP}:TA^{port})$

$(TB^{IP}:TB^{port})$

Session A-B

$(TA^{IP}:TA^{port}) \Leftrightarrow (TB^{IP}:TB^{port})$

Session A-B

$(A^{IP}:A^{port}) \Leftrightarrow (TB^{IP}:TB^{port})$

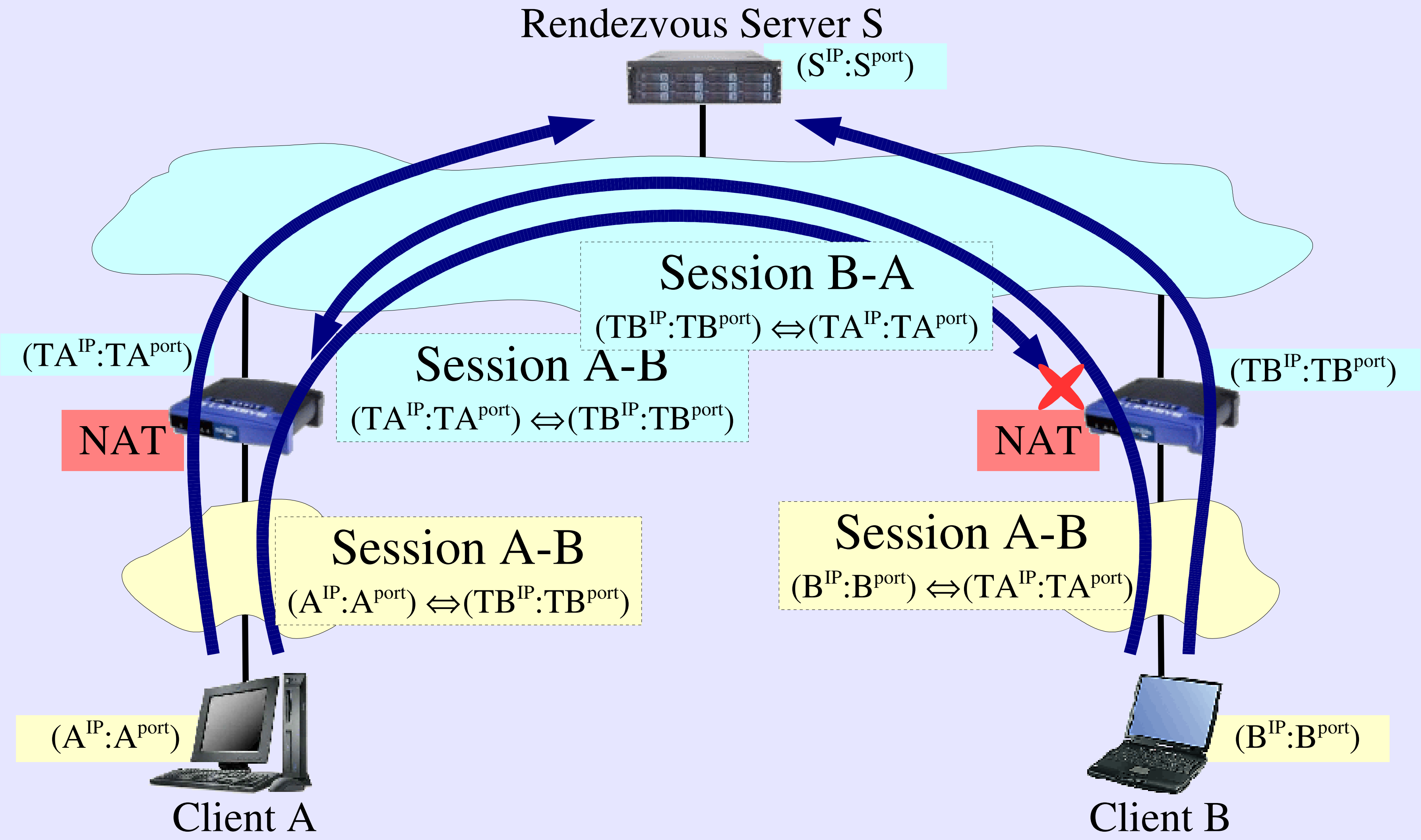
Session A-B

$(B^{IP}:B^{port}) \Leftrightarrow (TA^{IP}:TA^{port})$

$(A^{IP}:A^{port})$

$(B^{IP}:B^{port})$

# UDP Hole Punching



Rendezvous Server S

$(S^{IP}:S^{port})$

Session B-A

$(TB^{IP}:TB^{port}) \Leftrightarrow (TA^{IP}:TA^{port})$

$(TA^{IP}:TA^{port})$

Session A-B

$(TA^{IP}:TA^{port}) \Leftrightarrow (TB^{IP}:TB^{port})$

$(TB^{IP}:TB^{port})$

NAT

NAT

Session A-B

$(A^{IP}:A^{port}) \Leftrightarrow (TB^{IP}:TB^{port})$

Session A-B

$(B^{IP}:B^{port}) \Leftrightarrow (TA^{IP}:TA^{port})$

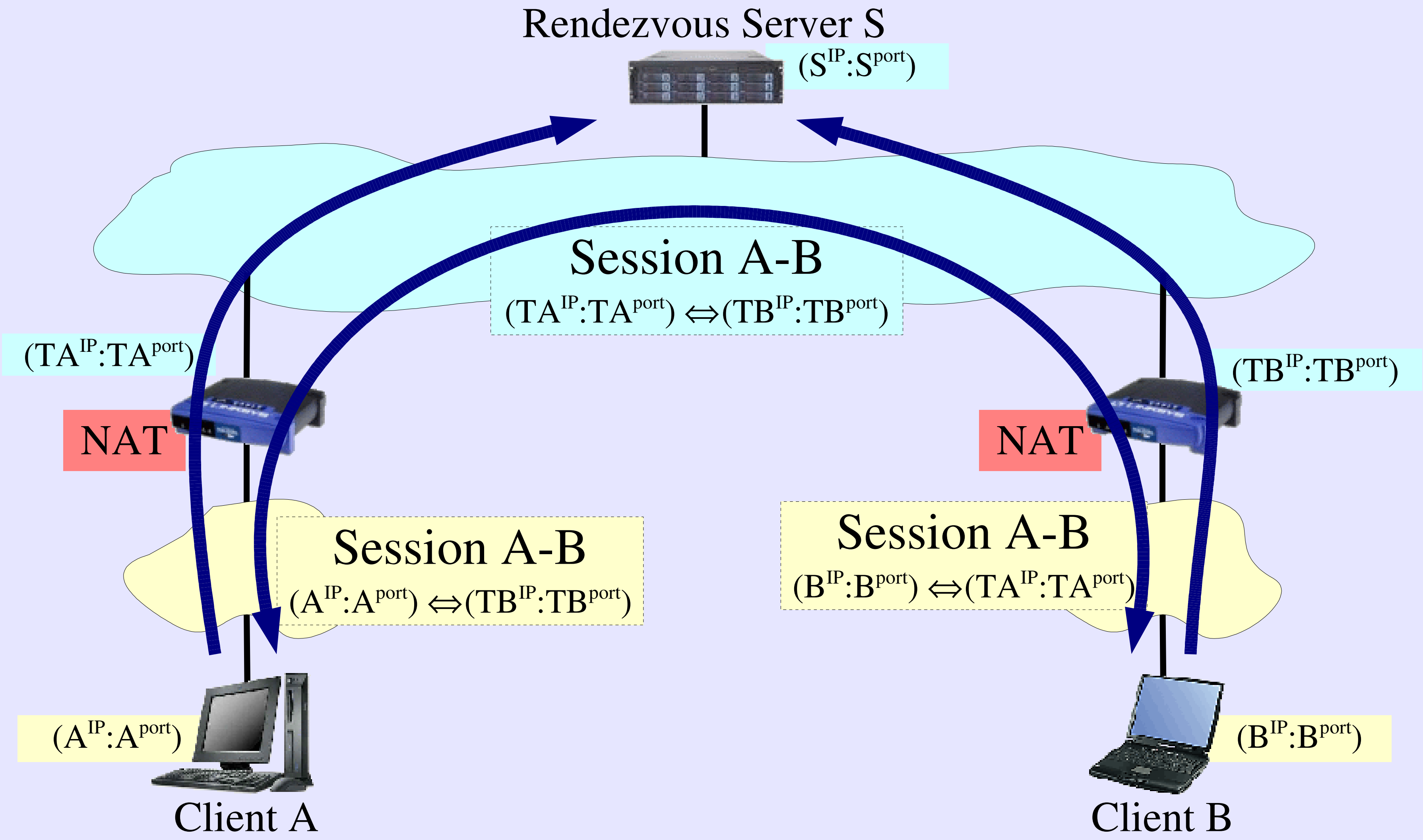
$(A^{IP}:A^{port})$

$(B^{IP}:B^{port})$

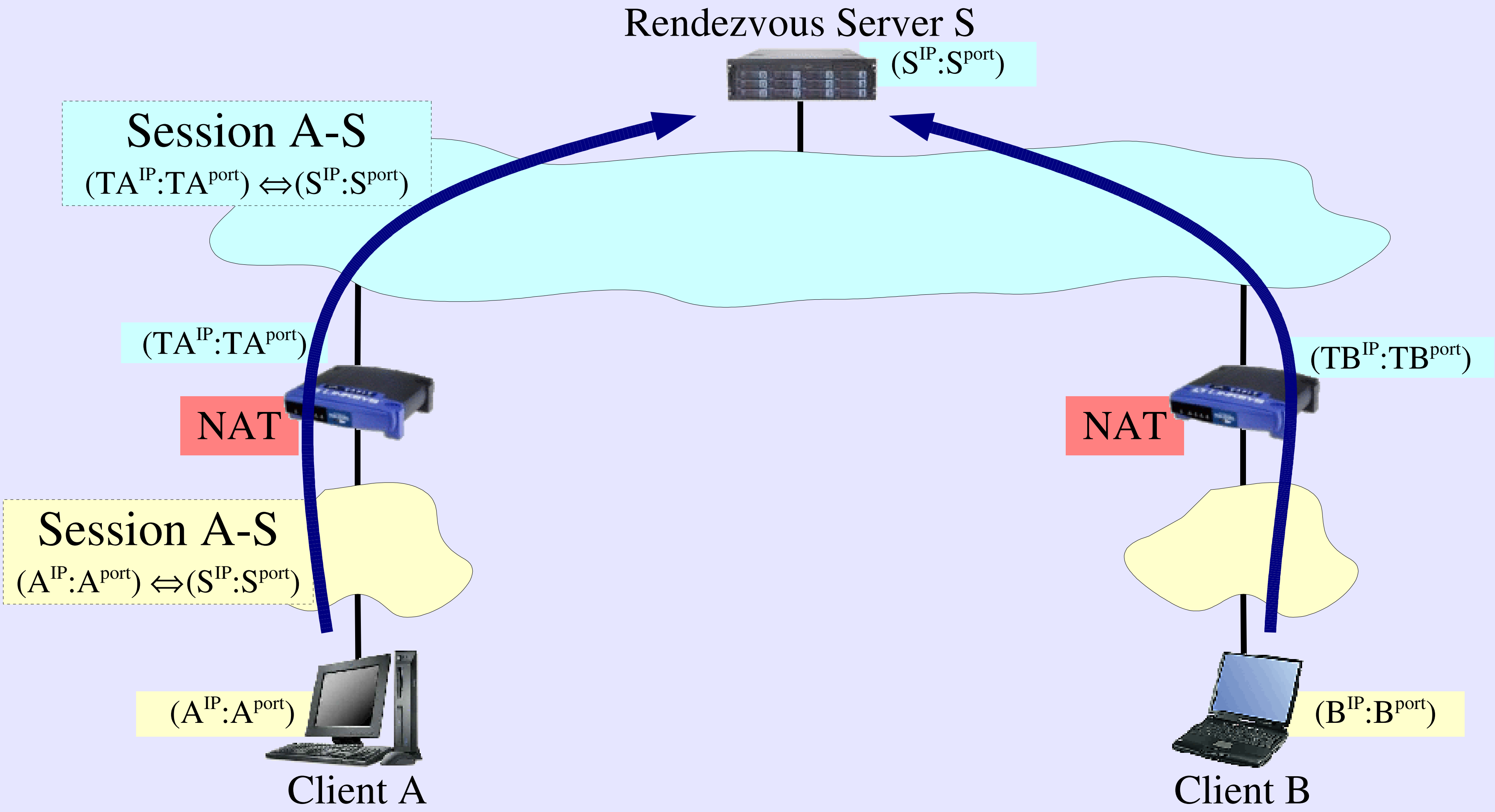
Client A

Client B

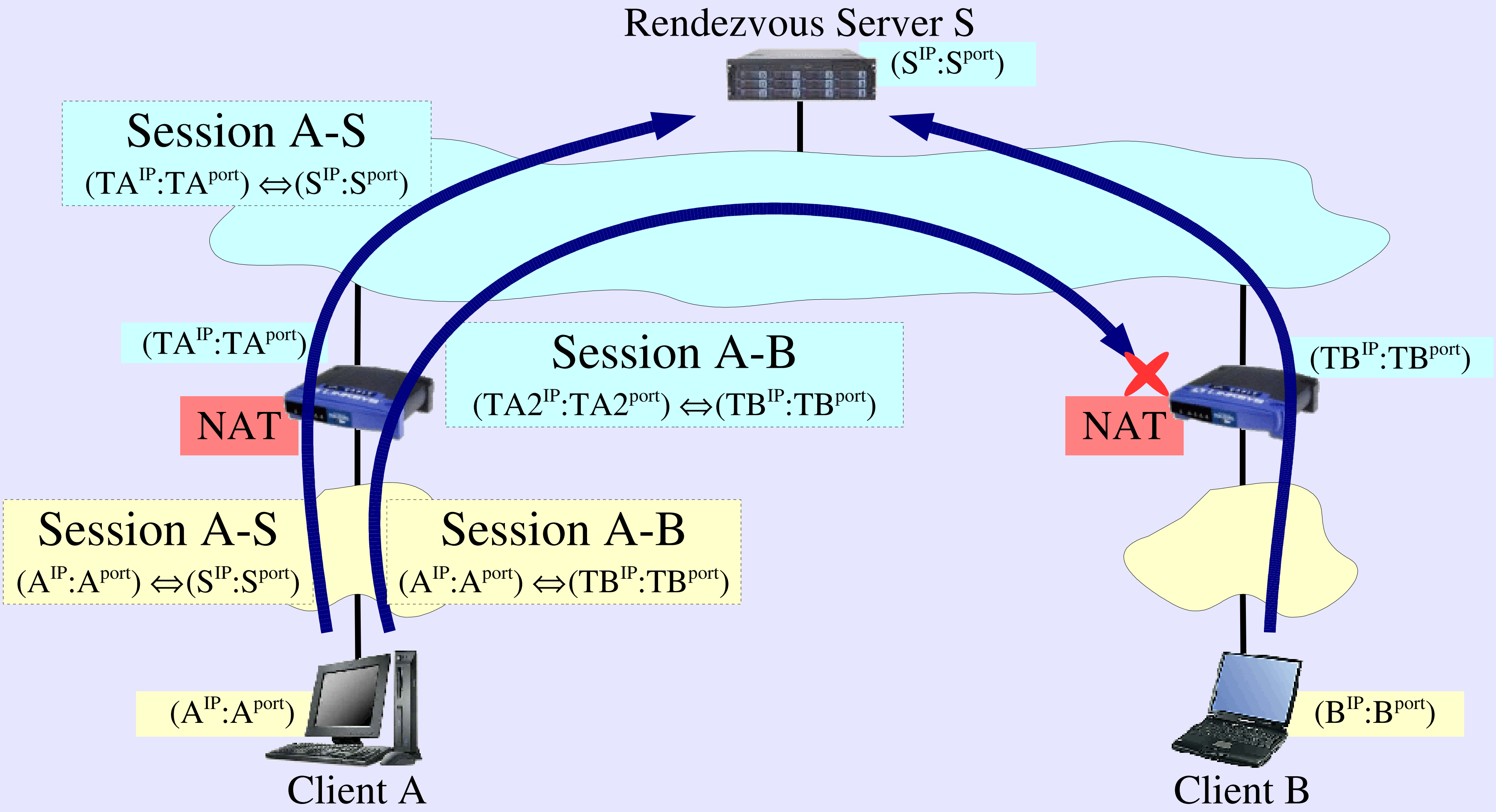
# UDP Hole Punching



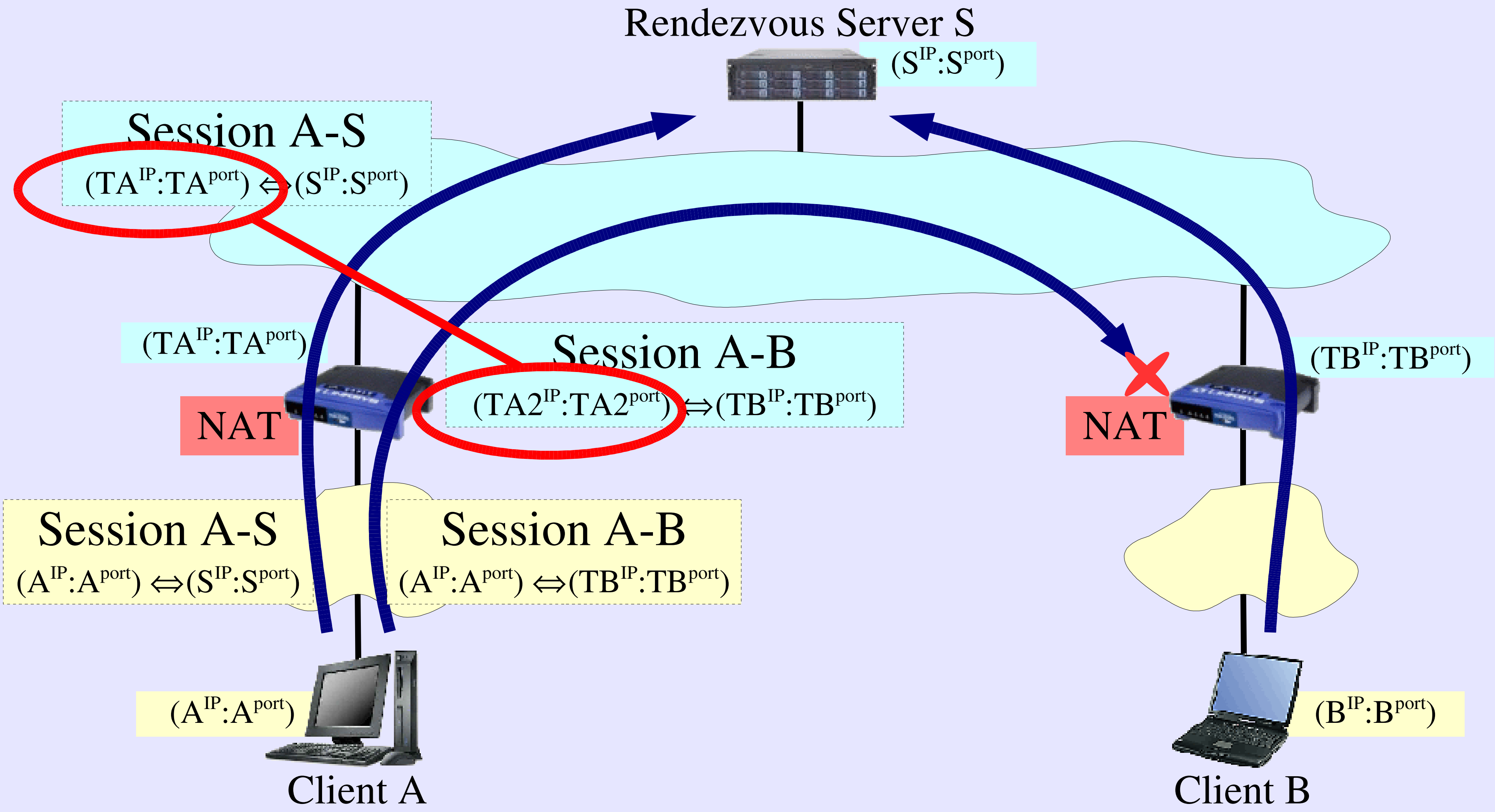
# UDP Hole Punching Gone Wrong



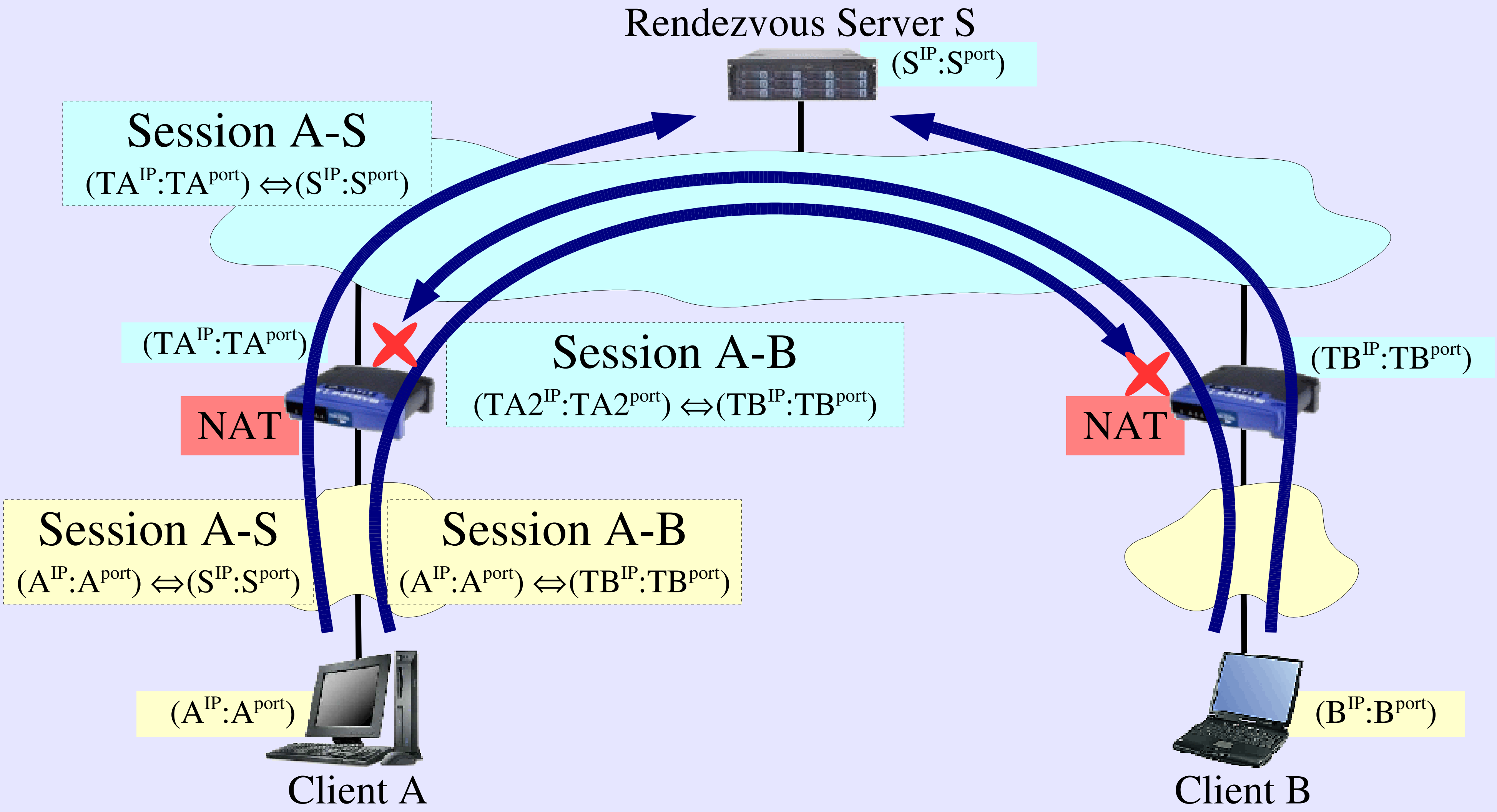
# UDP Hole Punching Gone Wrong



# UDP Hole Punching Gone Wrong



# UDP Hole Punching Gone Wrong


Rendezvous Server S
$(S^{IP}:S^{port})$
Session A-S
$(TA^{IP}:TA^{port}) \Leftrightarrow (S^{IP}:S^{port})$
$(TA^{IP}:TA^{port})$
NAT
Session A-B
$(TA2^{IP}:TA2^{port}) \Leftrightarrow (TB^{IP}:TB^{port})$
NAT
$(TB^{IP}:TB^{port})$
Session A-S
$(A^{IP}:A^{port}) \Leftrightarrow (S^{IP}:S^{port})$
Session A-B
$(A^{IP}:A^{port}) \Leftrightarrow (TB^{IP}:TB^{port})$
$(A^{IP}:A^{port})$
Client A
$(B^{IP}:B^{port})$
Client B

# TCP Hole Punching

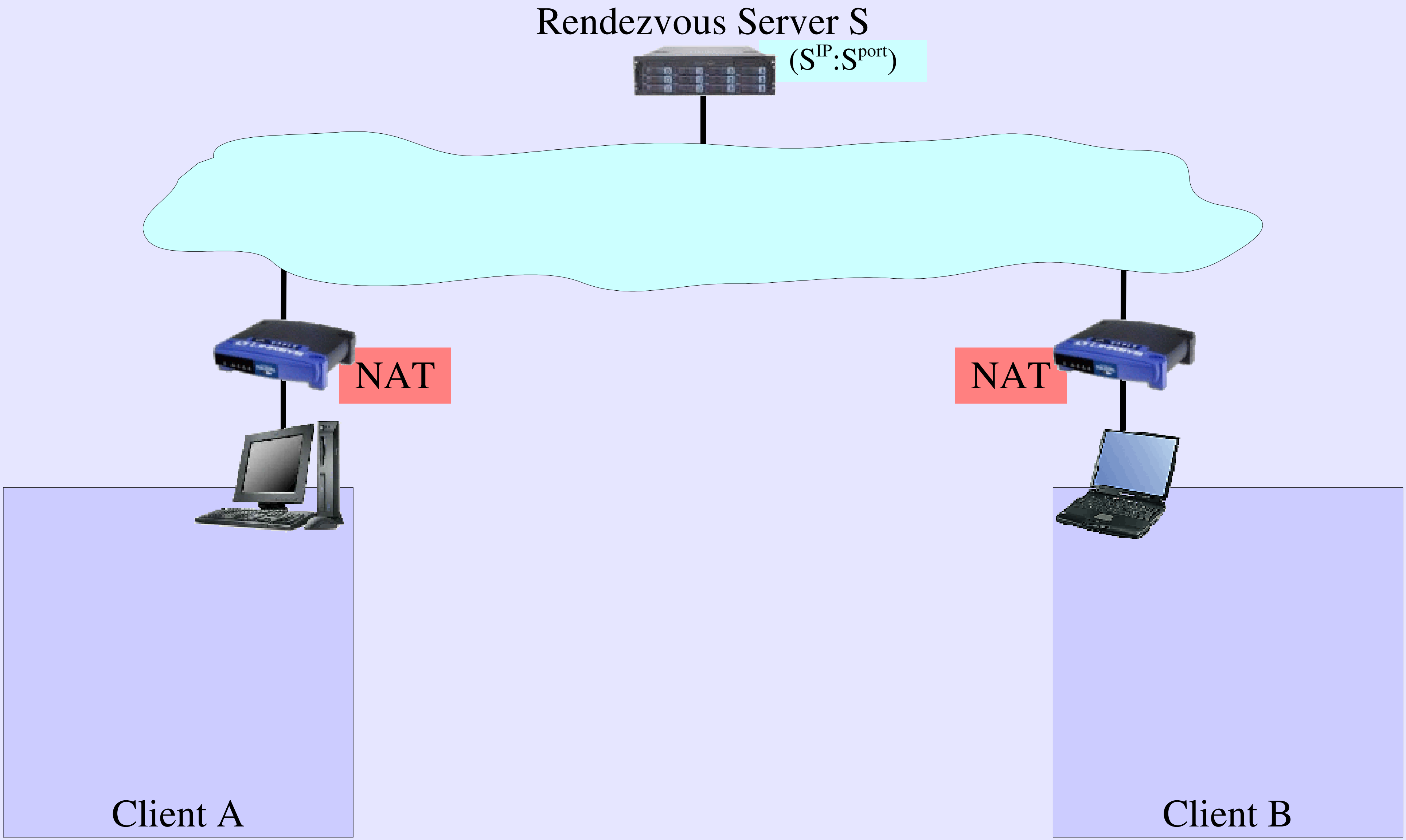
TCP has always supported crucial feature

- “Simultaneous TCP Open” [RFC 793]

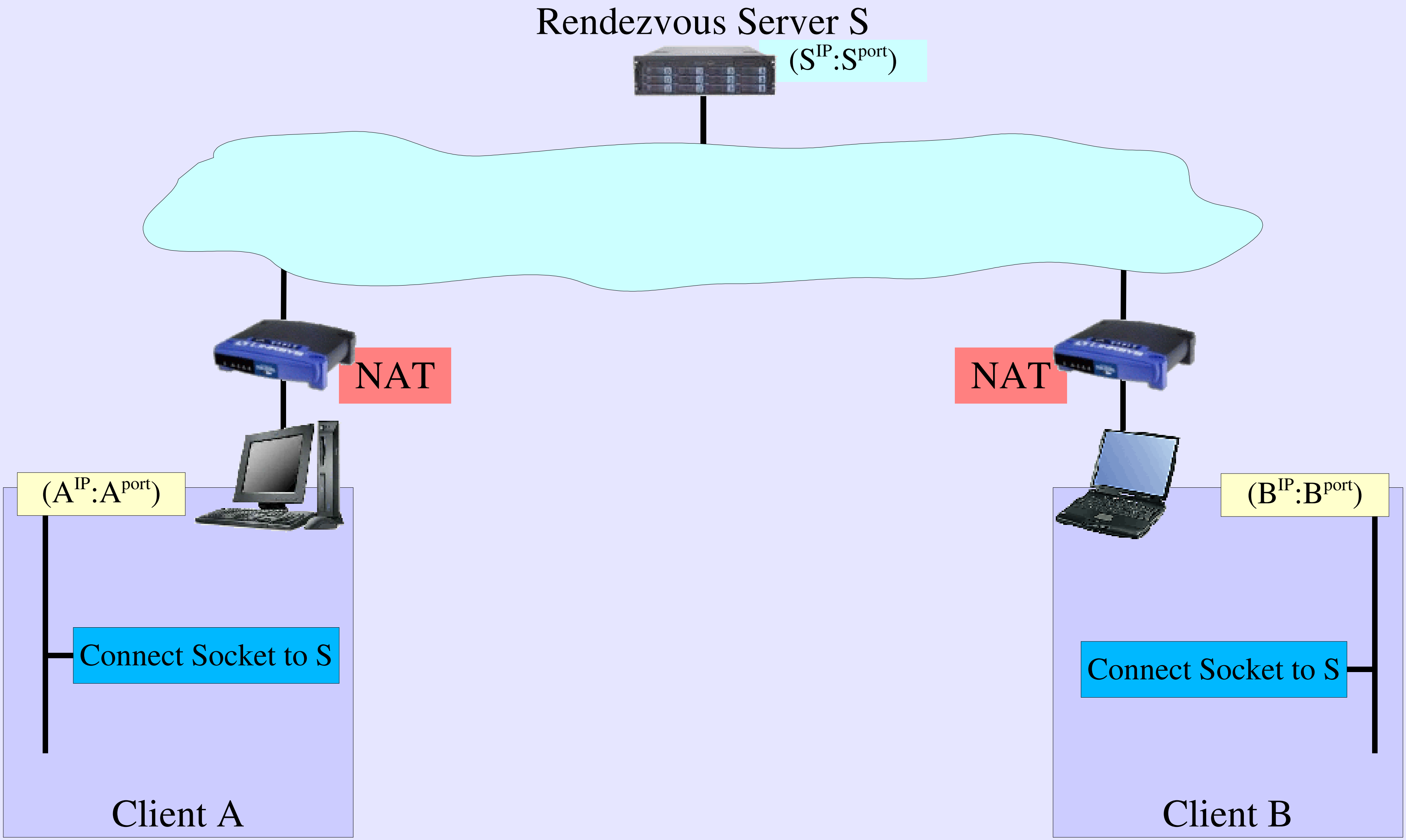
Difficulties:

- More ways for NATs to behave poorly
- TCP sockets API oriented toward client/server

# TCP Hole Punching



# TCP Hole Punching



Rendezvous Server S

$(S^{IP}:S^{port})$

NAT

NAT

$(A^{IP}:A^{port})$

Connect Socket to S

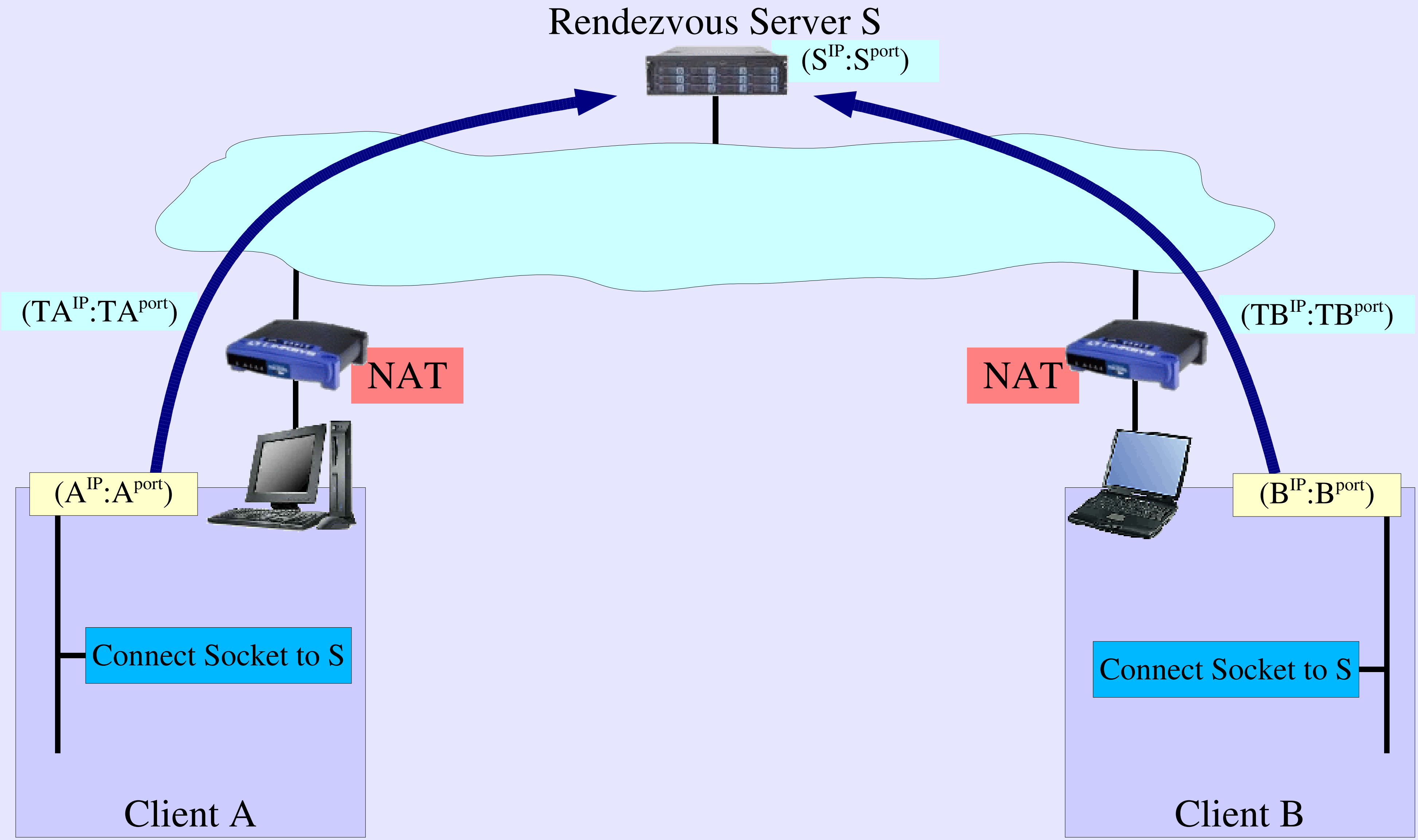
Client A

$(B^{IP}:B^{port})$

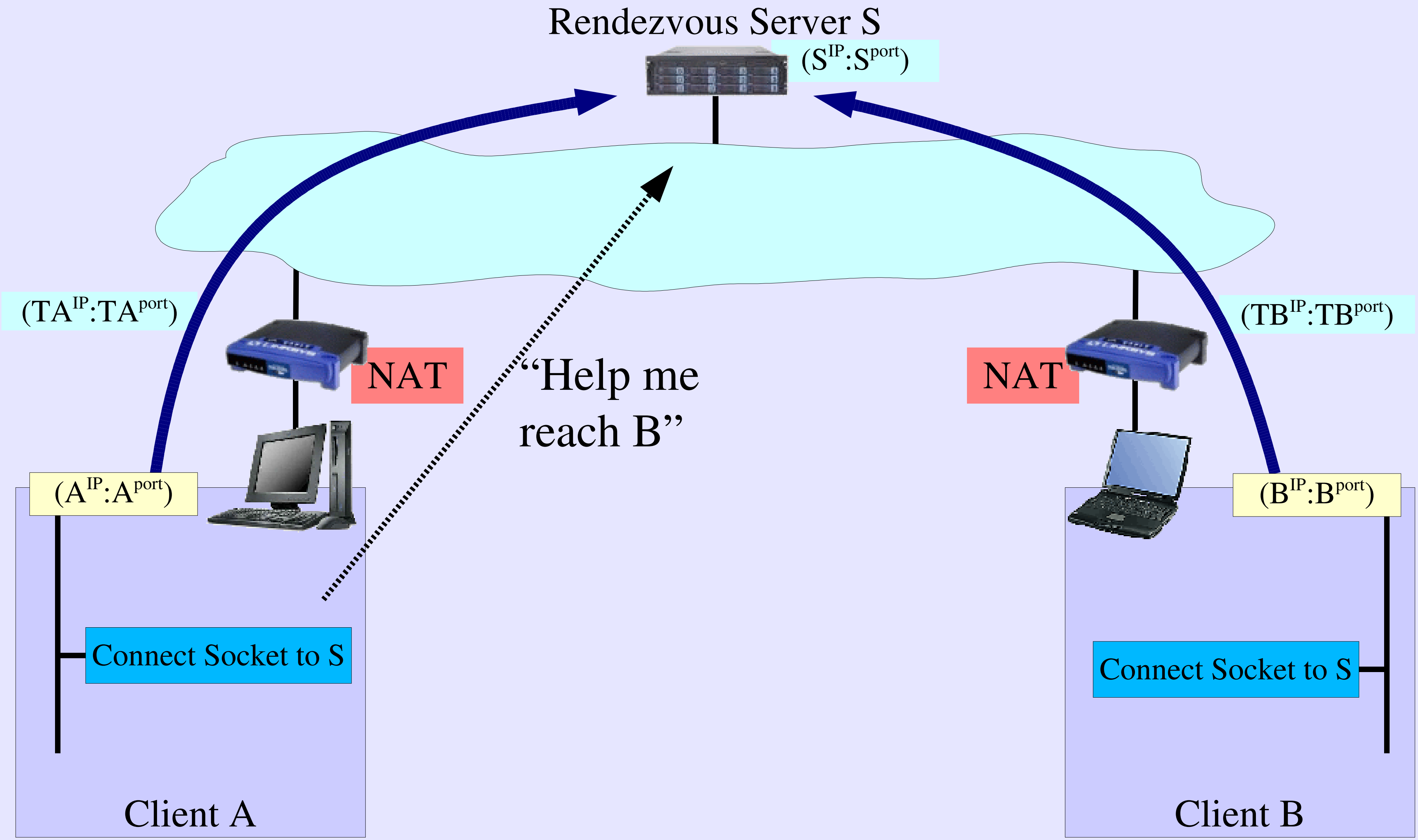
Connect Socket to S

Client B

# TCP Hole Punching



# TCP Hole Punching


Rendezvous Server S

$(S^{IP}:S^{port})$

$(TA^{IP}:TA^{port})$

$(TB^{IP}:TB^{port})$

NAT

NAT

"Help me reach B"

$(A^{IP}:A^{port})$

$(B^{IP}:B^{port})$

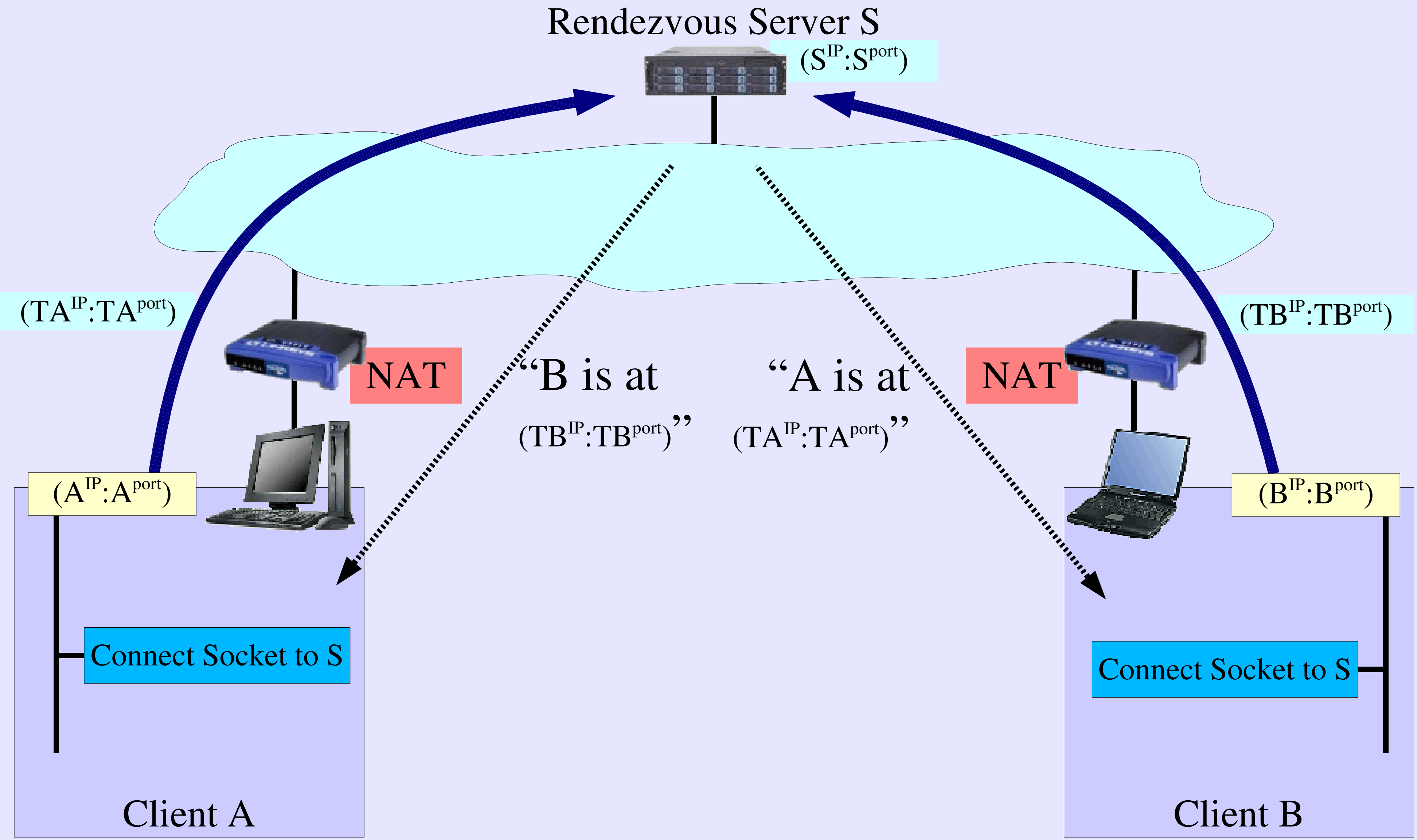
Connect Socket to S

Connect Socket to S

Client A

Client B

# TCP Hole Punching



Rendezvous Server S

$(S^{IP}:S^{port})$

$(TA^{IP}:TA^{port})$

$(TB^{IP}:TB^{port})$

NAT

NAT

"B is at $(TB^{IP}:TB^{port})$"

"A is at $(TA^{IP}:TA^{port})$"

$(A^{IP}:A^{port})$

$(B^{IP}:B^{port})$

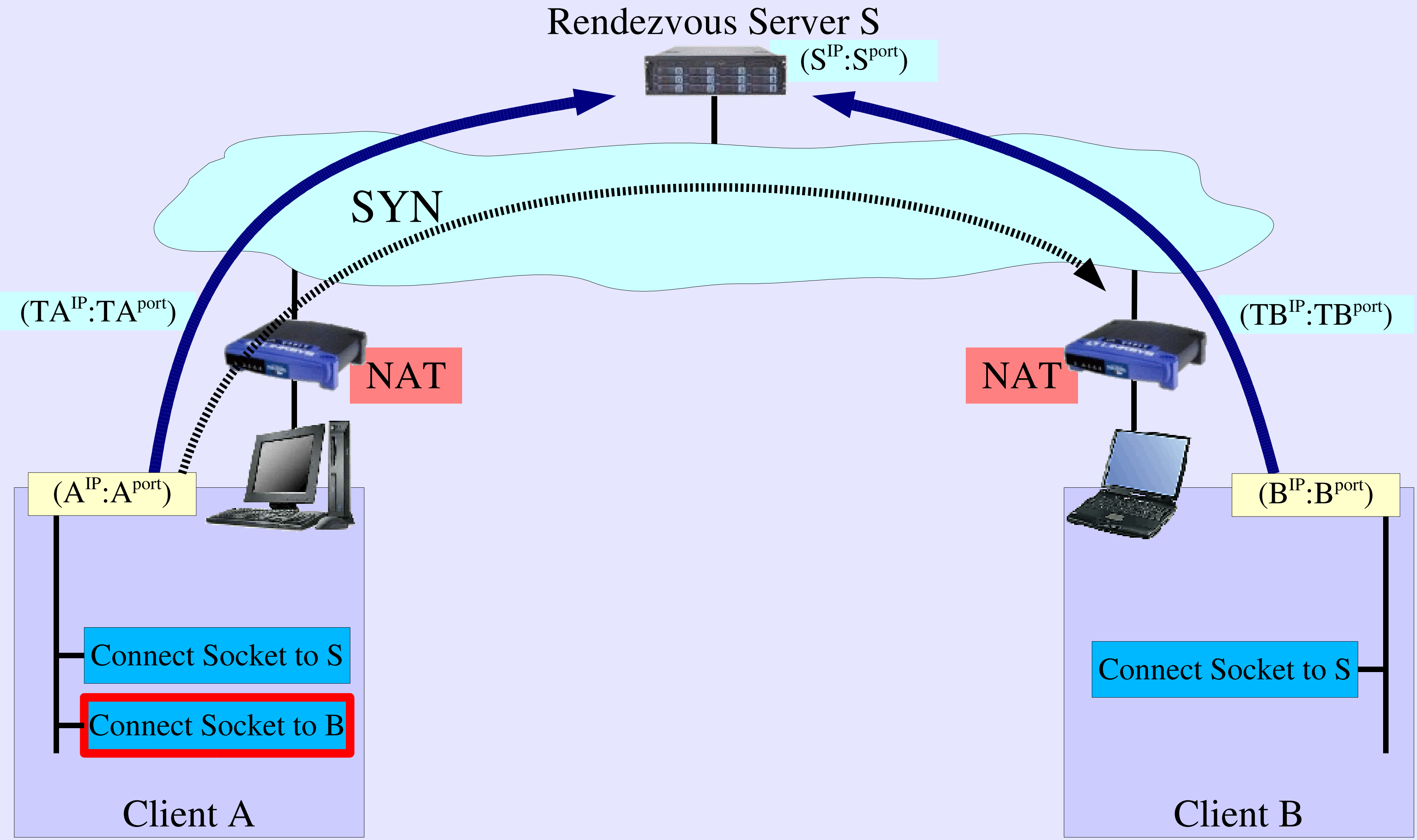
Connect Socket to S

Connect Socket to S

Client A

Client B

# TCP Hole Punching



Rendezvous Server S

($S^{IP}$:$S^{port}$)

SYN

($TA^{IP}$:$TA^{port}$)

NAT

NAT

($TB^{IP}$:$TB^{port}$)

($A^{IP}$:$A^{port}$)

($B^{IP}$:$B^{port}$)

Connect Socket to S

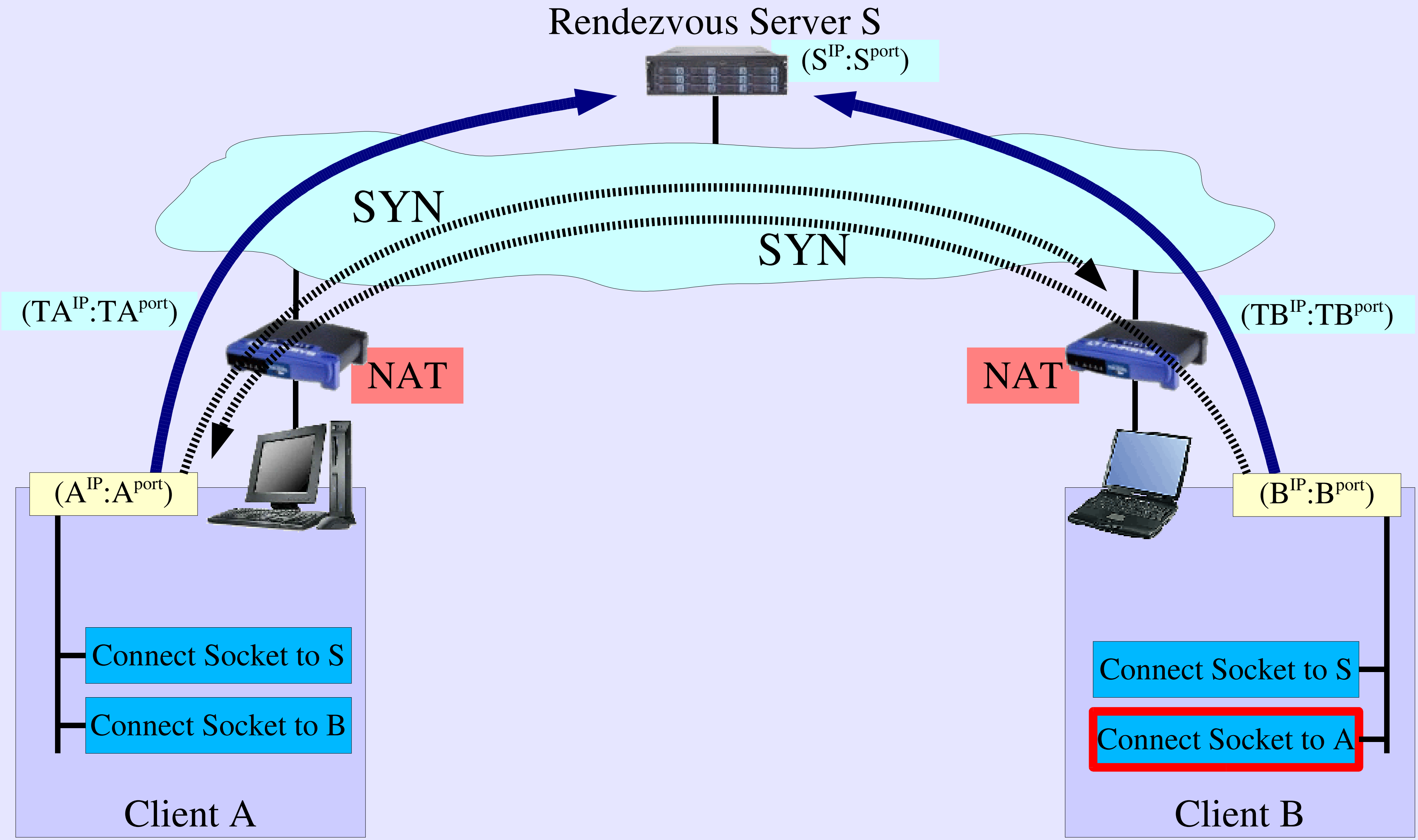
Connect Socket to B

Connect Socket to S

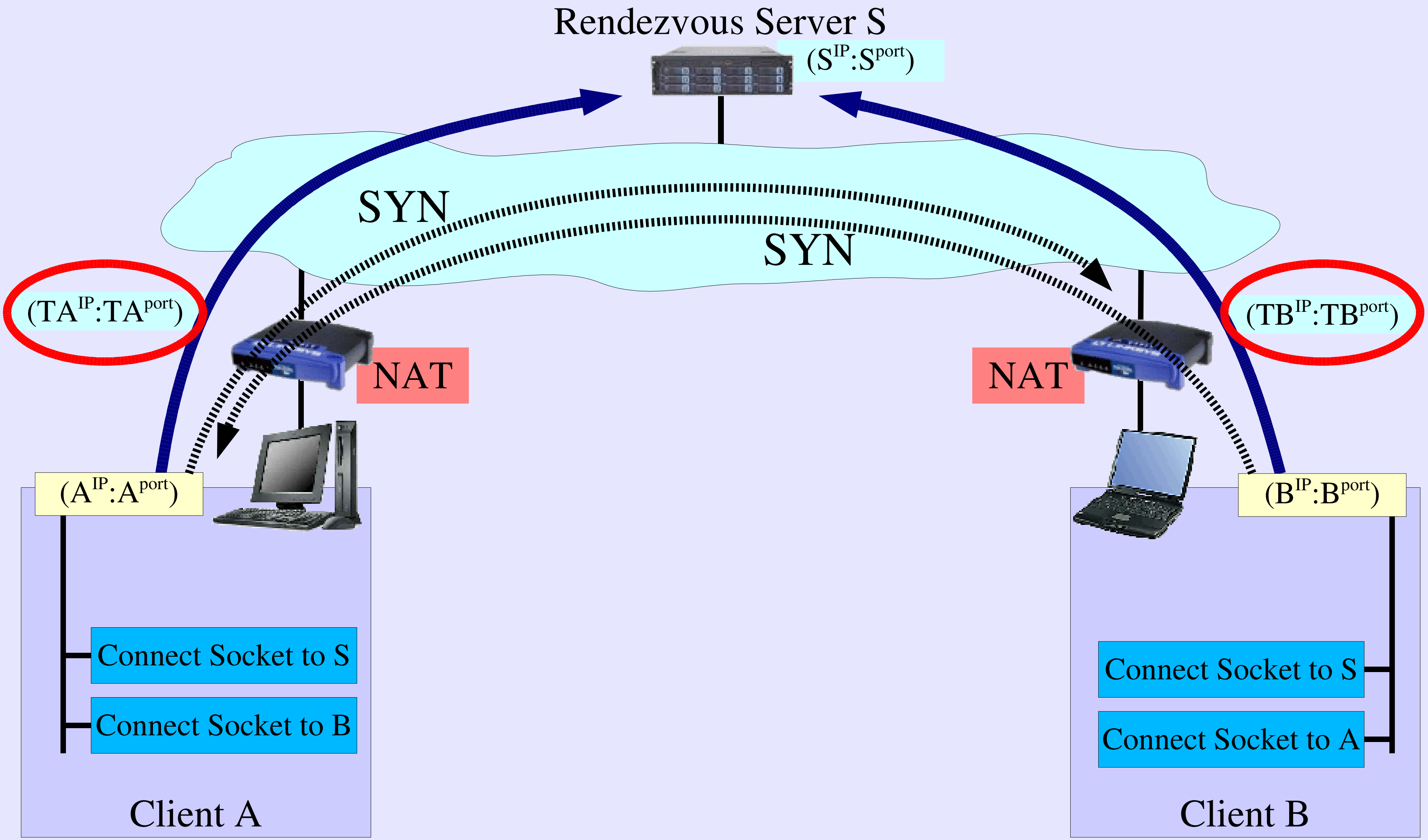
Client A

Client B

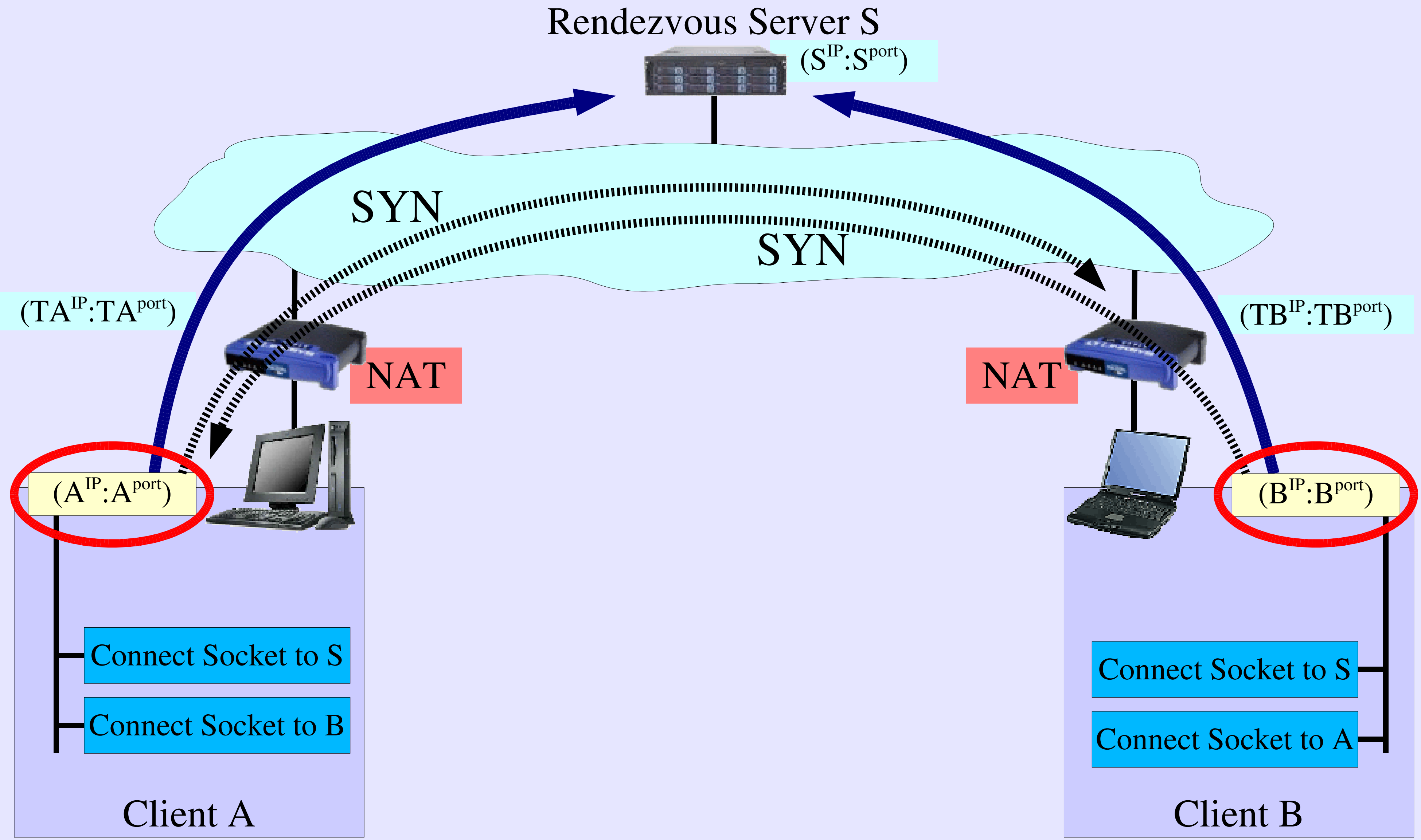
# TCP Hole Punching



# TCP Hole Punching



# TCP Hole Punching



Rendezvous Server S

$(S^{IP}:S^{port})$

SYN

SYN

$(TA^{IP}:TA^{port})$

$(TB^{IP}:TB^{port})$

NAT

NAT

$(A^{IP}:A^{port})$

$(B^{IP}:B^{port})$

Connect Socket to S

Connect Socket to B

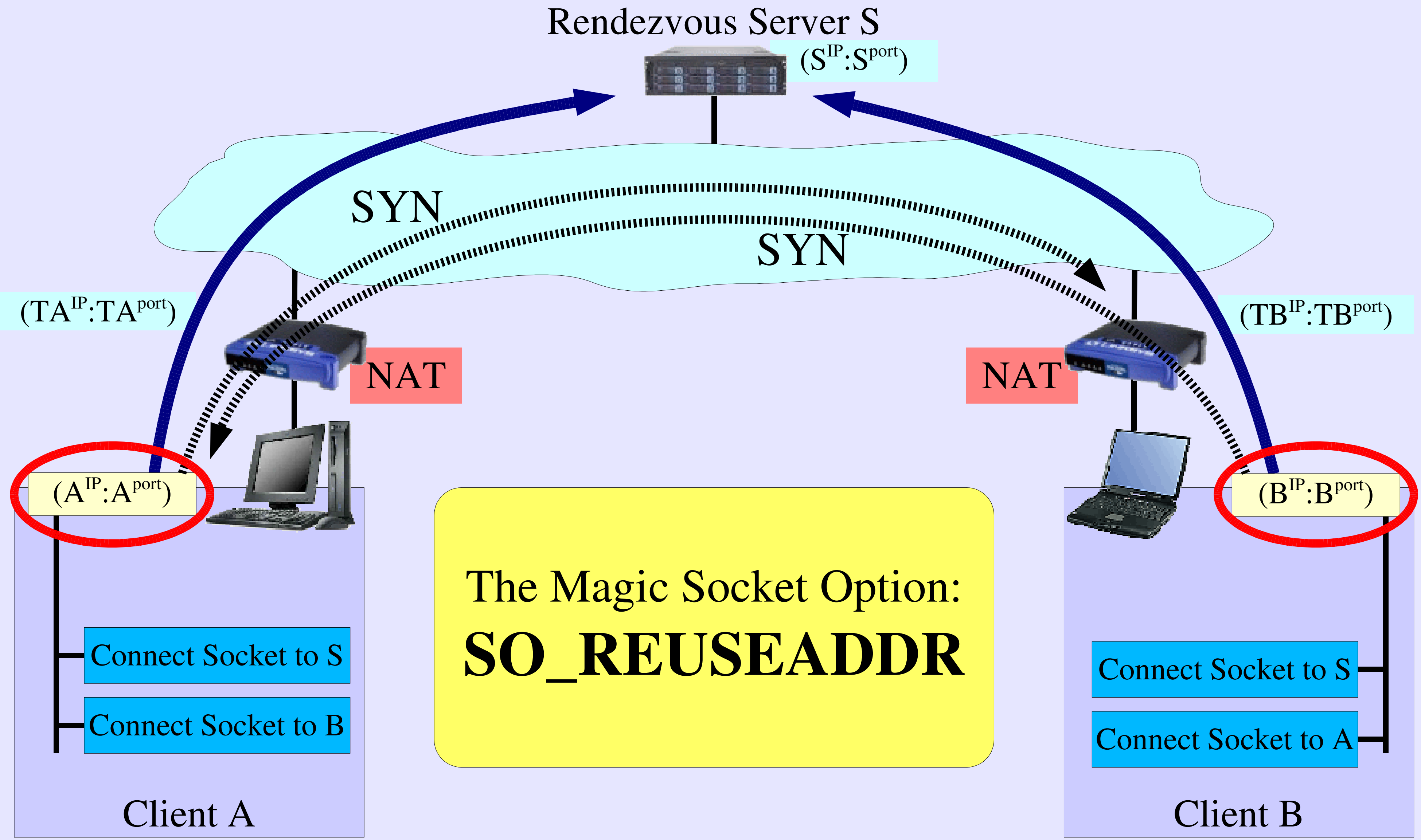
Client A

Connect Socket to S

Connect Socket to A

Client B

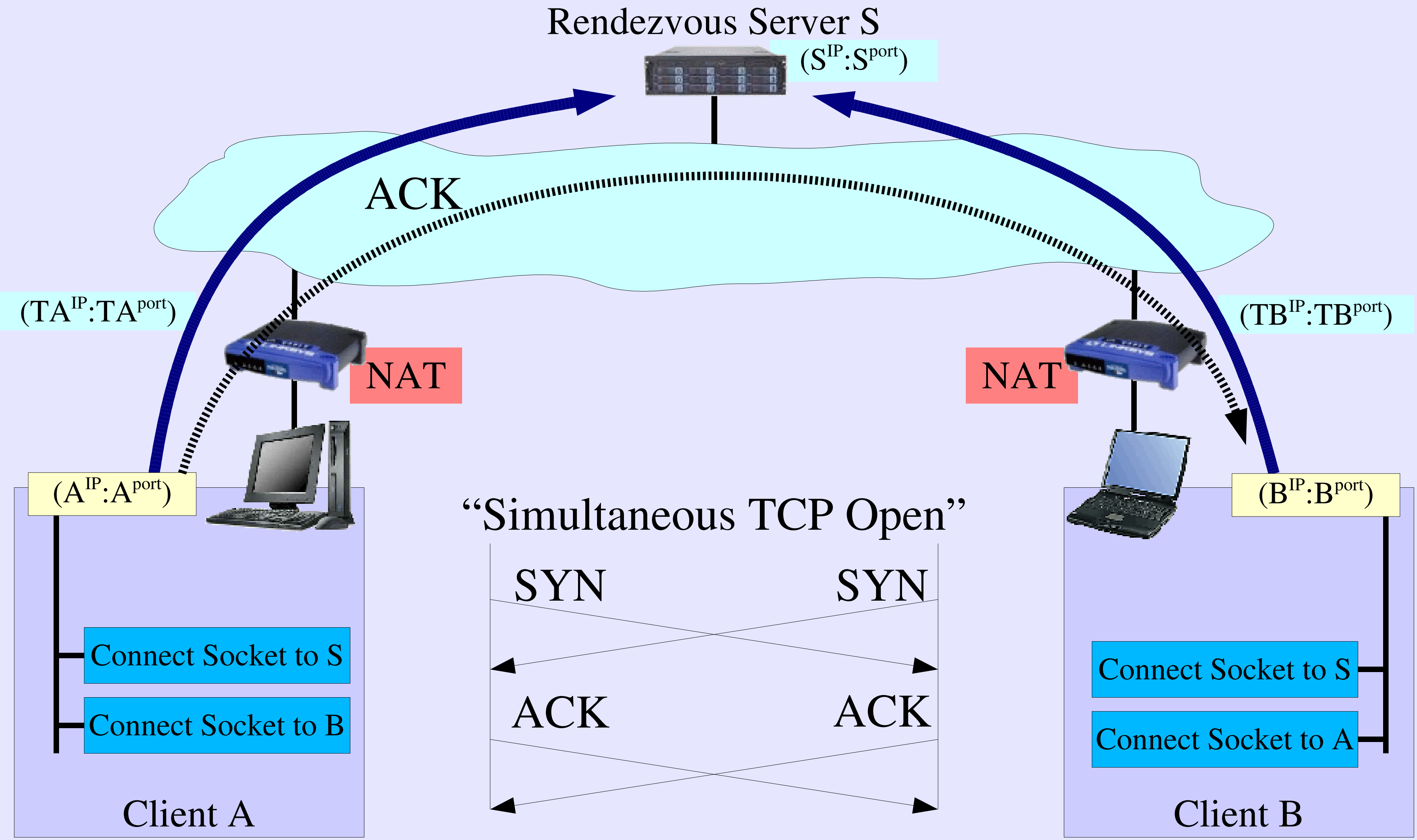
# TCP Hole Punching



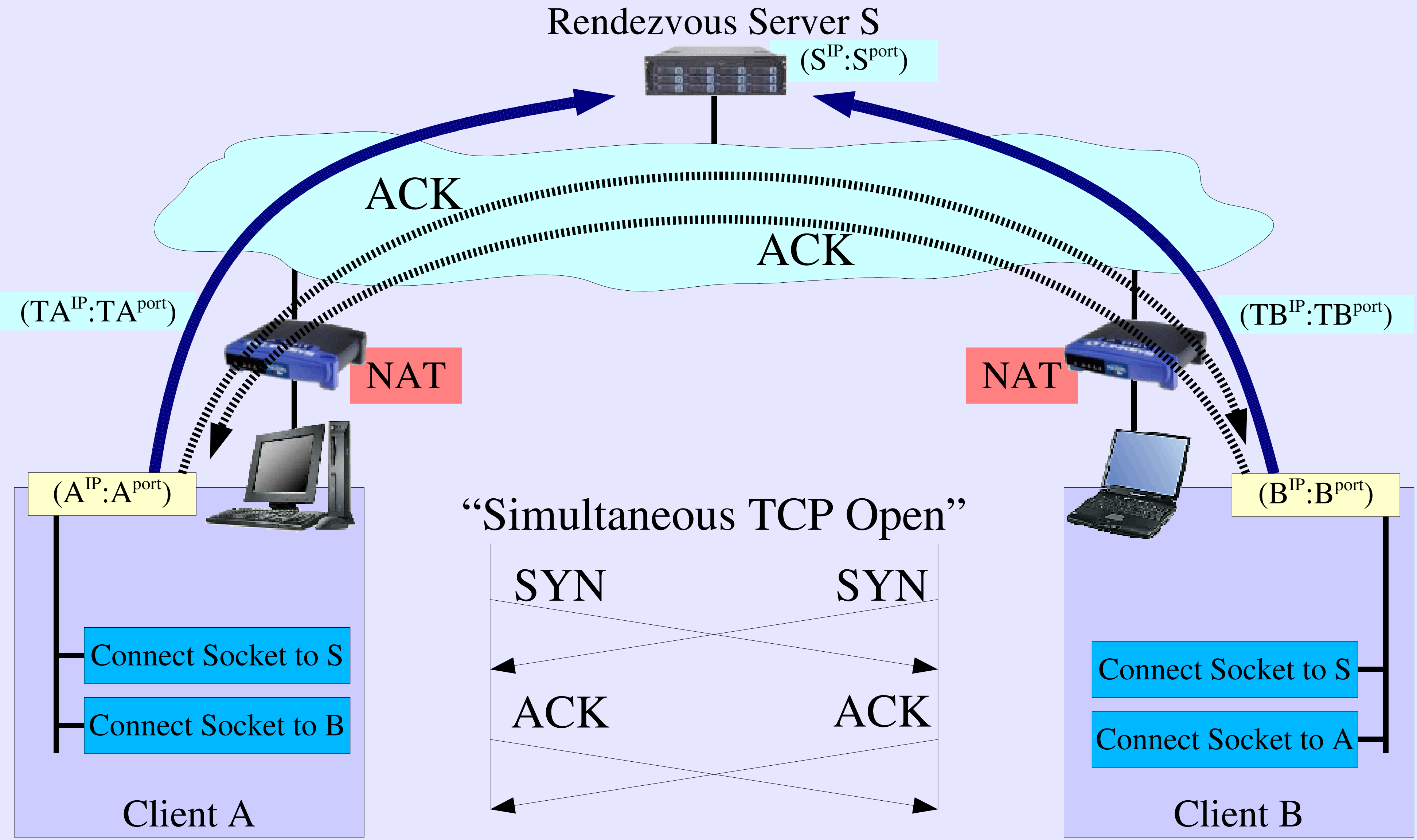
The Magic Socket Option:

**SO_REUSEADDR**

# TCP Hole Punching



# TCP Hole Punching



Rendezvous Server S

$(S^{IP}:S^{port})$

ACK

ACK

$(TA^{IP}:TA^{port})$

$(TB^{IP}:TB^{port})$

NAT

NAT

$(A^{IP}:A^{port})$

$(B^{IP}:B^{port})$

"Simultaneous TCP Open"

SYN

SYN

ACK

ACK

Connect Socket to S

Connect Socket to B

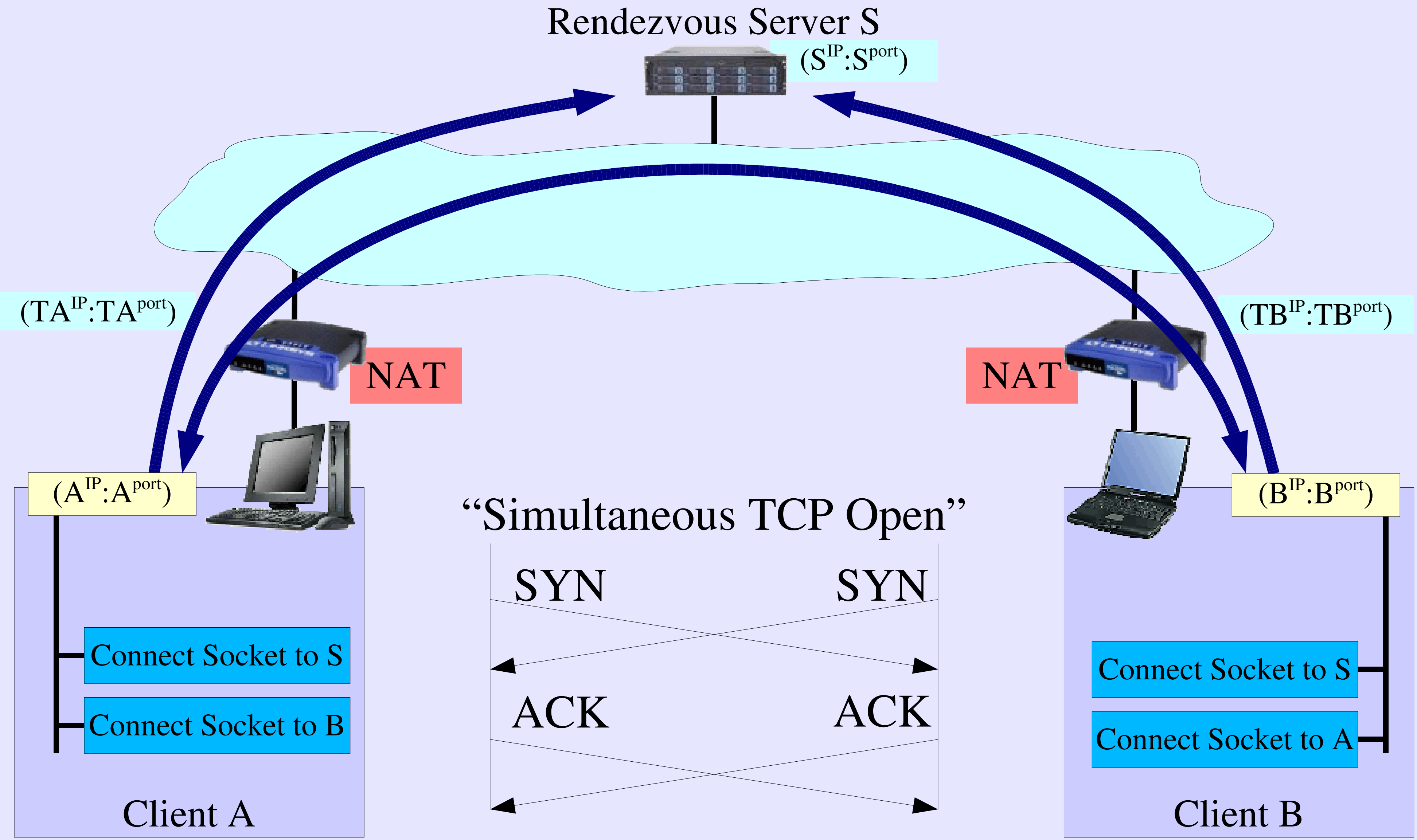
Client A

Connect Socket to S

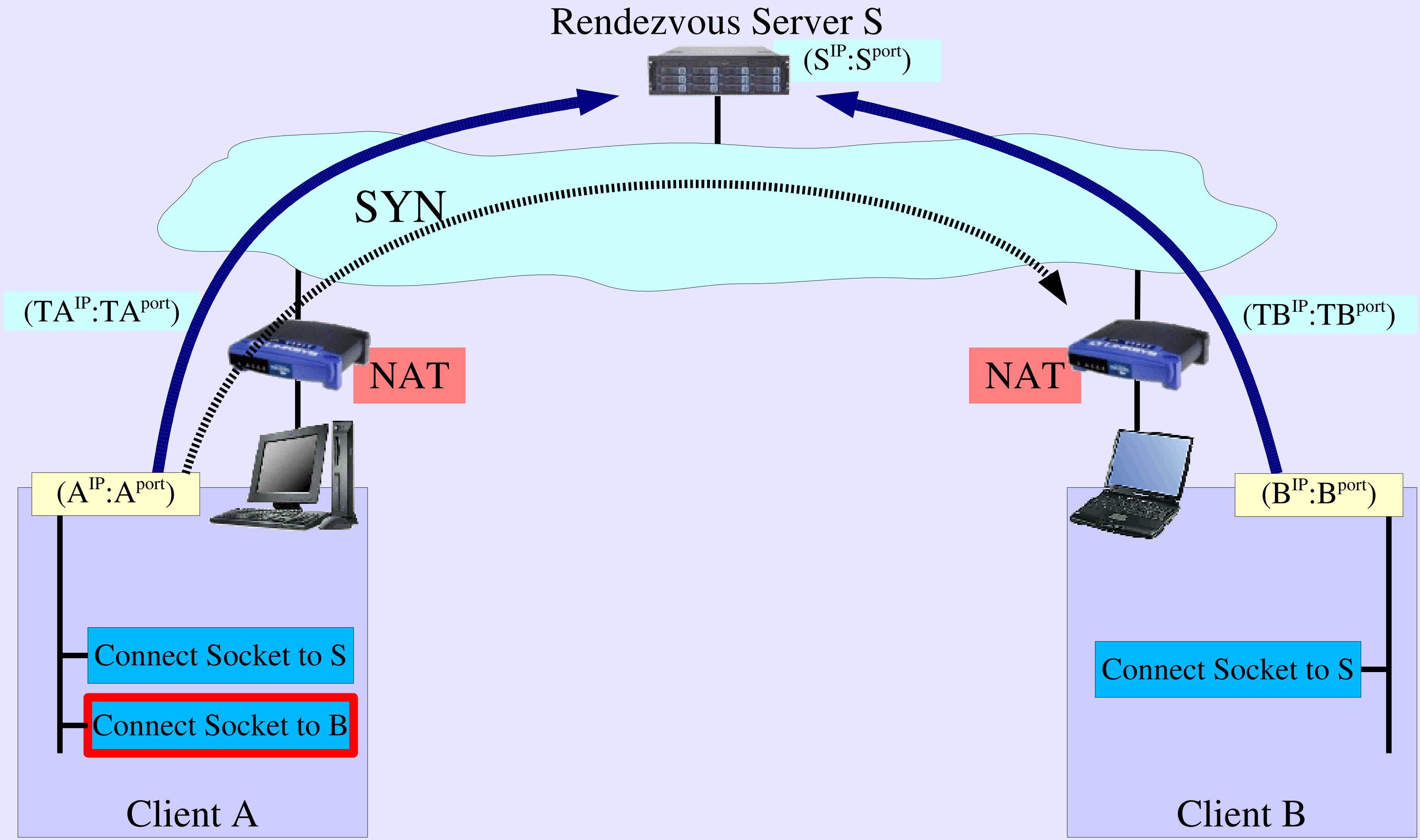
Connect Socket to A

Client B

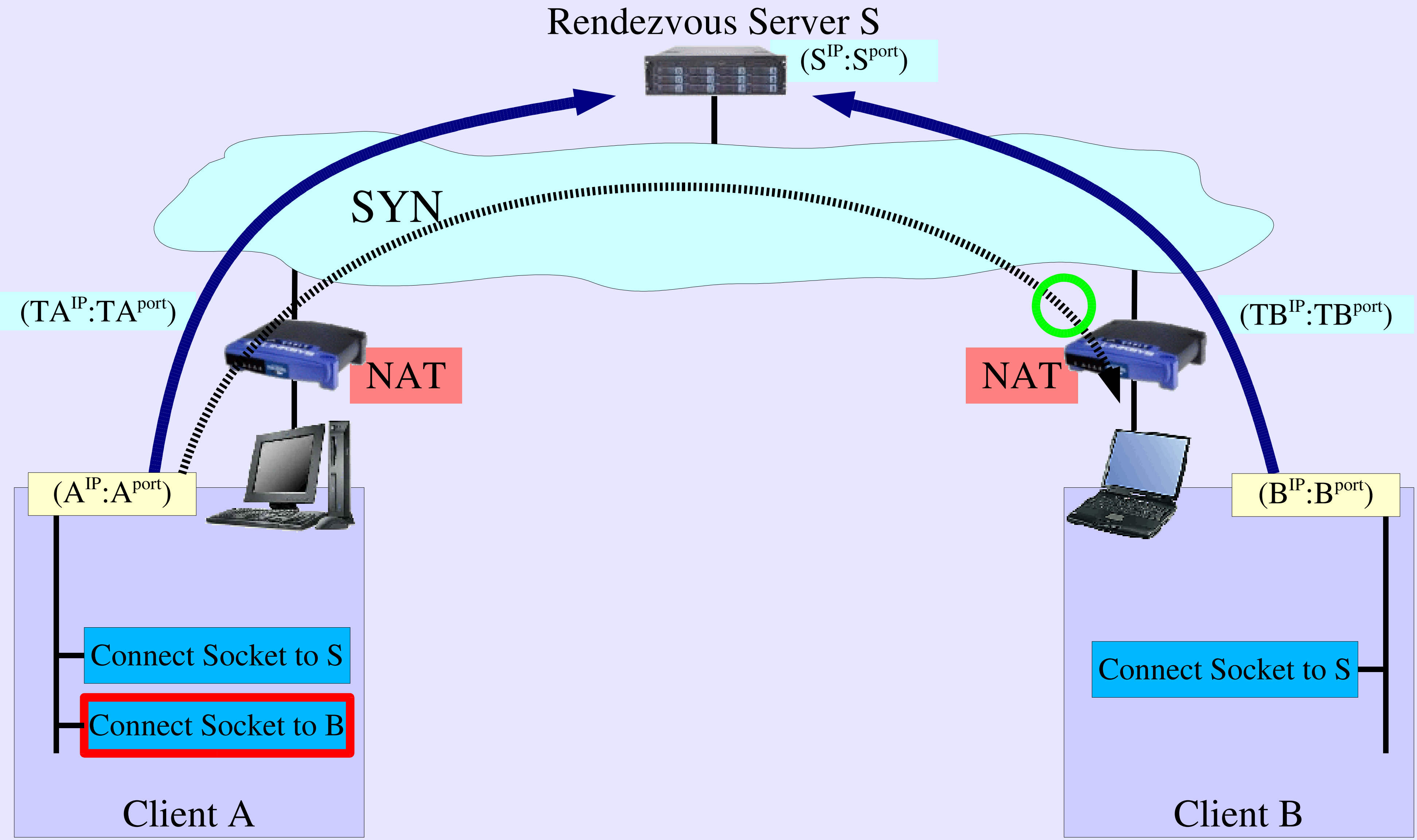
# TCP Hole Punching



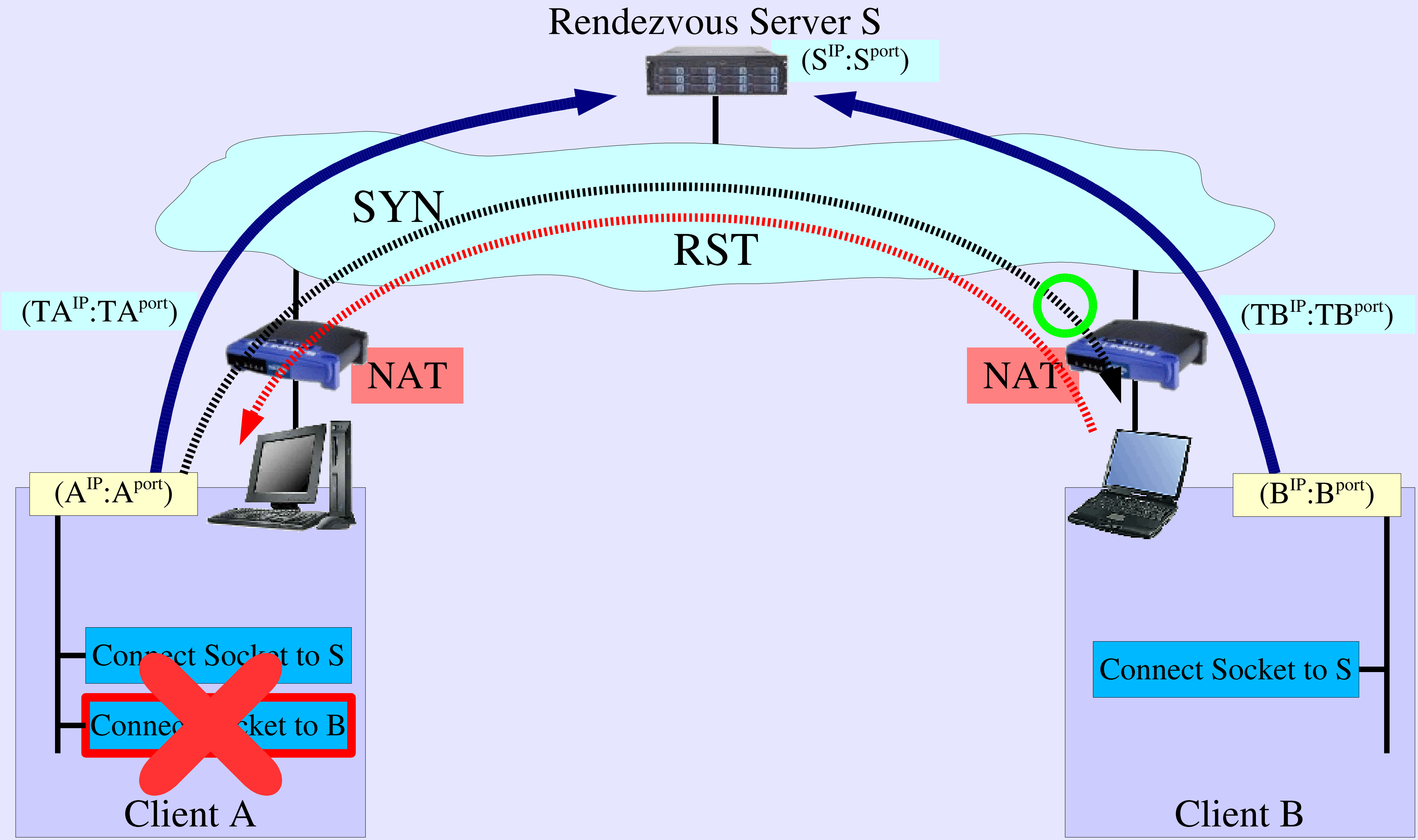
# Timing Caveat



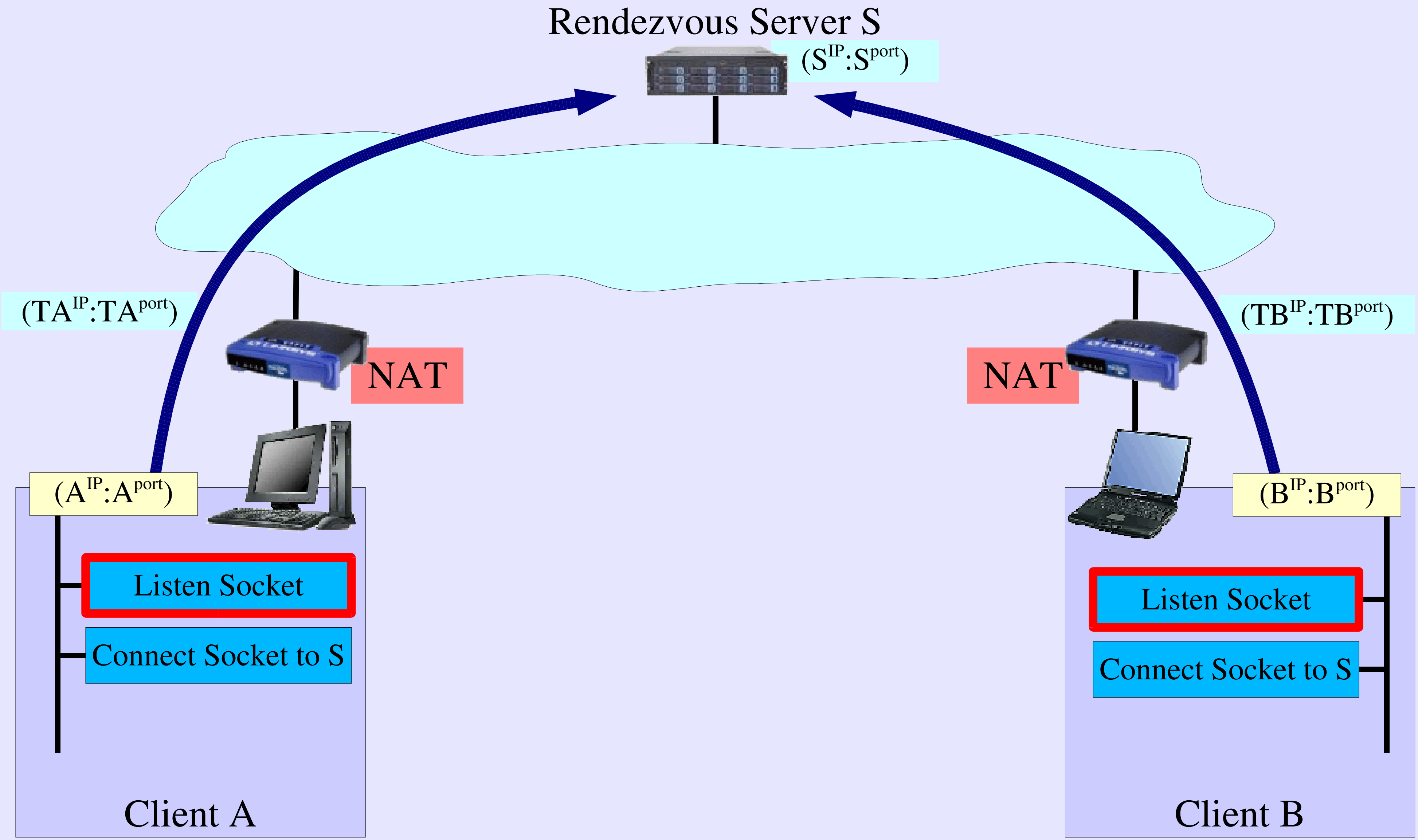
# Timing Caveat



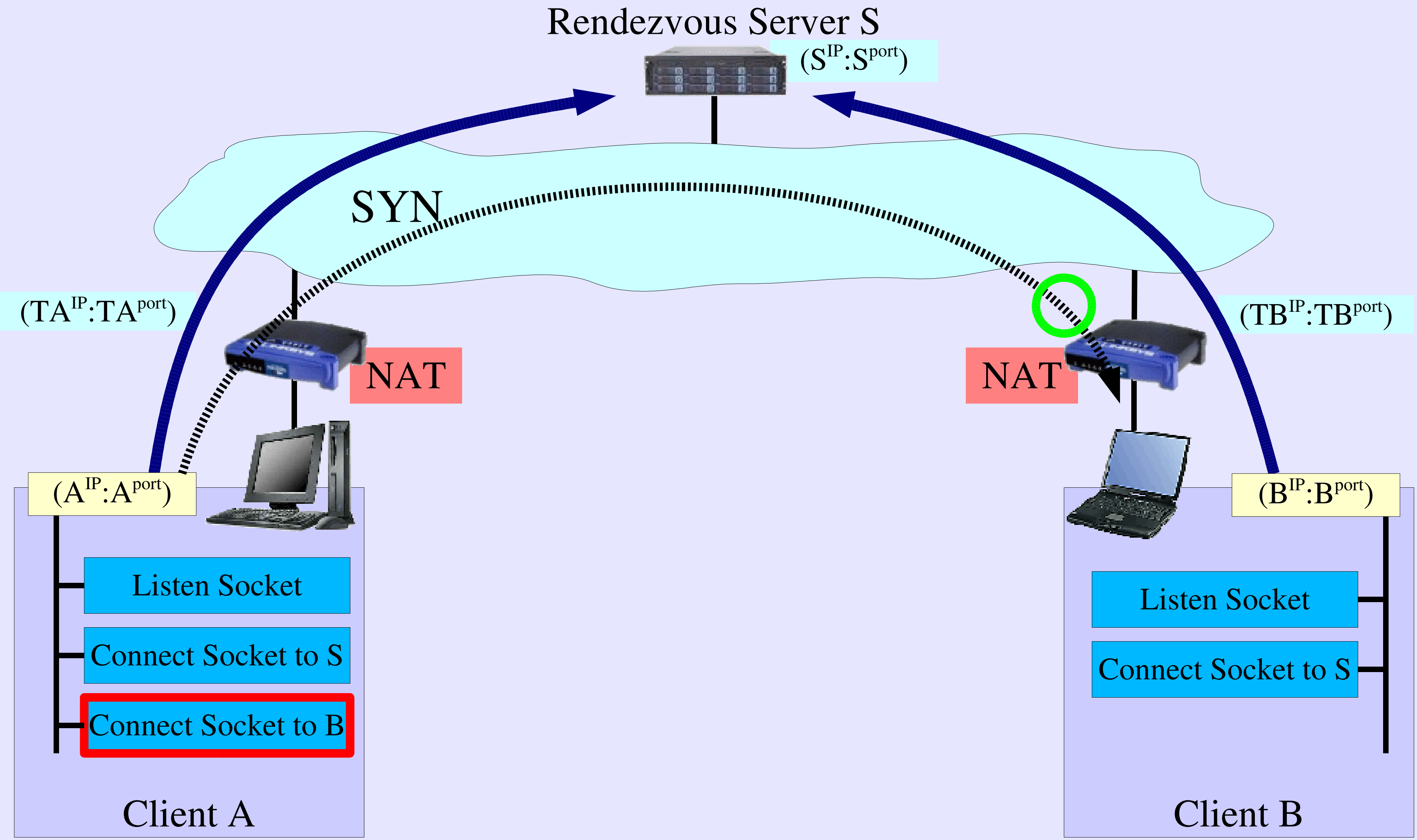
# Timing Caveat



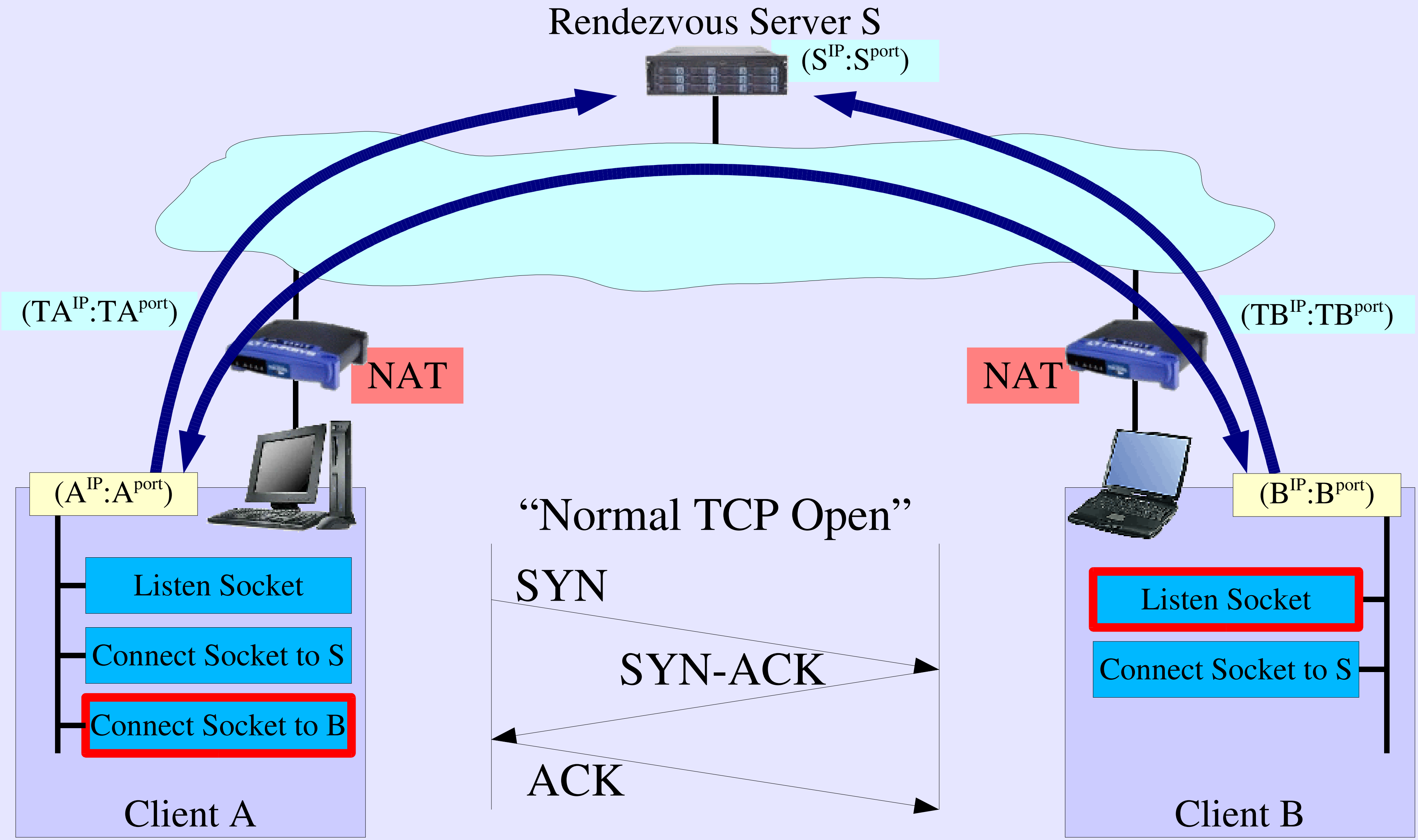
# Timing Solution



# Timing Solution



# Timing Solution



Rendezvous Server S

$(S^{IP}:S^{port})$

$(TA^{IP}:TA^{port})$

$(TB^{IP}:TB^{port})$

NAT

NAT

$(A^{IP}:A^{port})$

Listen Socket

Connect Socket to S

Connect Socket to B

Client A

$(B^{IP}:B^{port})$

Listen Socket

Connect Socket to S

Client B

“Normal TCP Open”

SYN

SYN-ACK

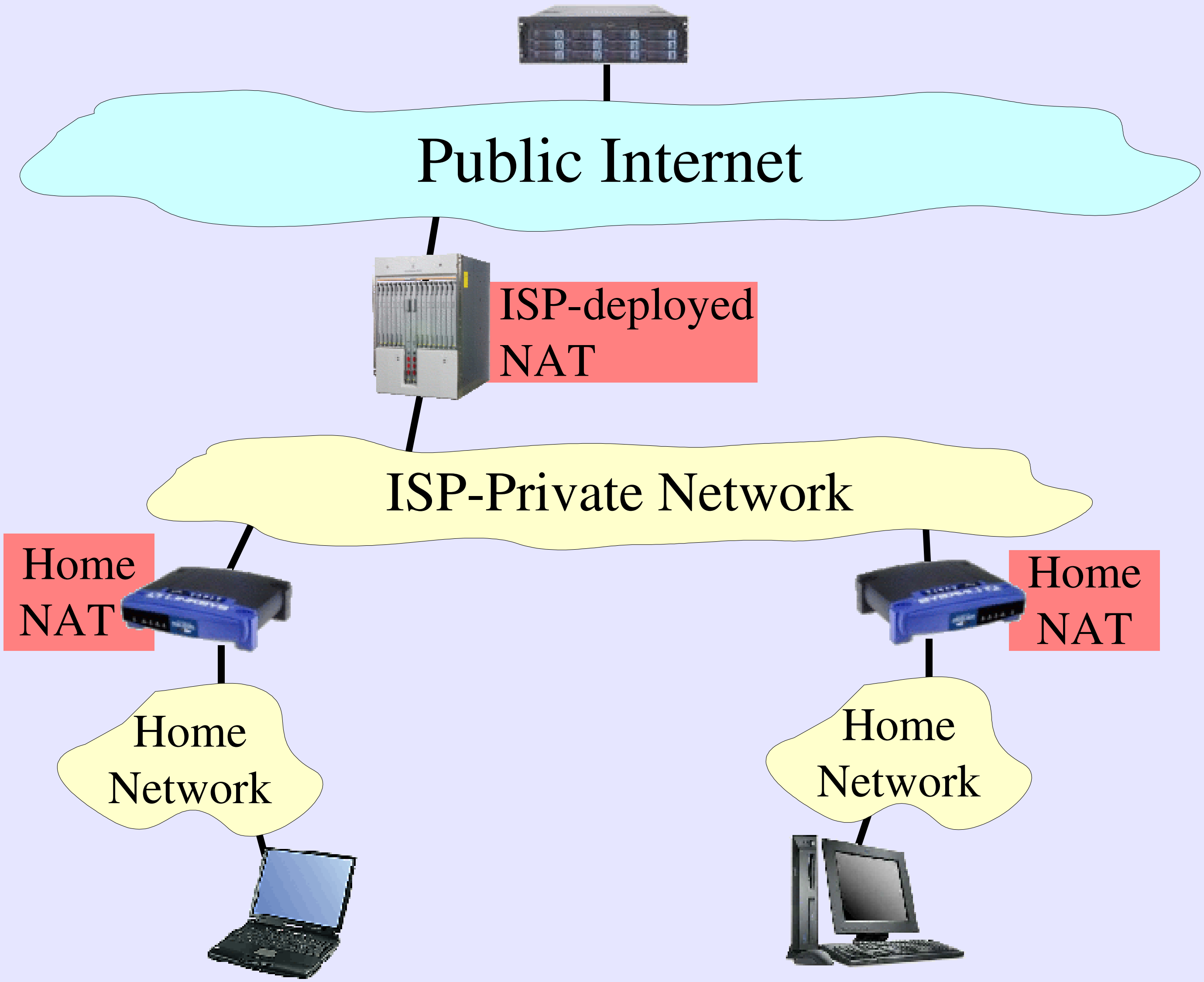
ACK

# TCP Hole Punching Gone Wrong

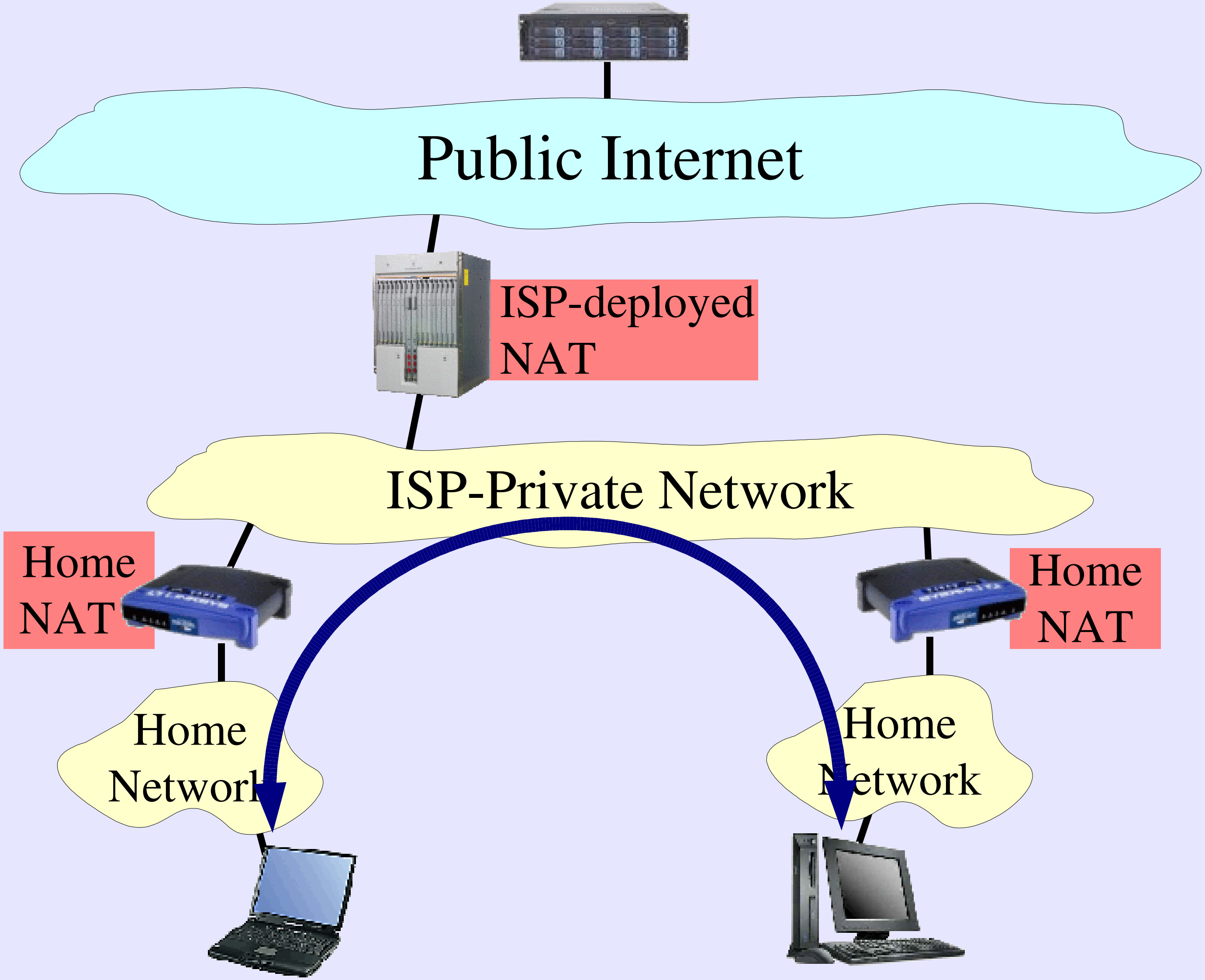
Potential problems:

- Inconsistent endpoint translation
  - Same as for UDP
- NAT could reject “unsolicited” incoming SYNs with RSTs or ICMP errs instead of just dropping
  - Connection failures, retry oscillation
- Buggy TCP state machine in host OS
  - Windows before XP SP2

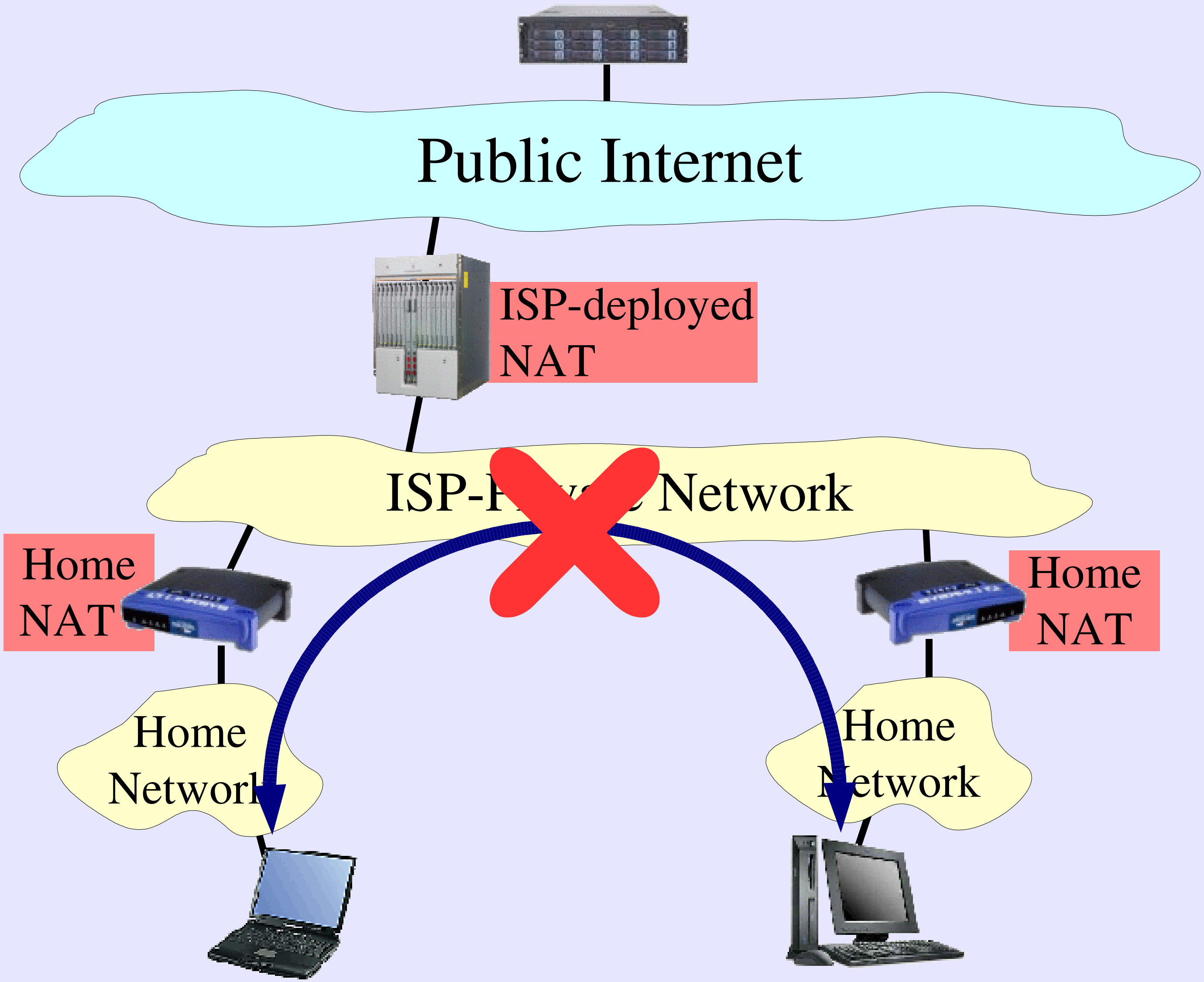
# Multi-Level NAT



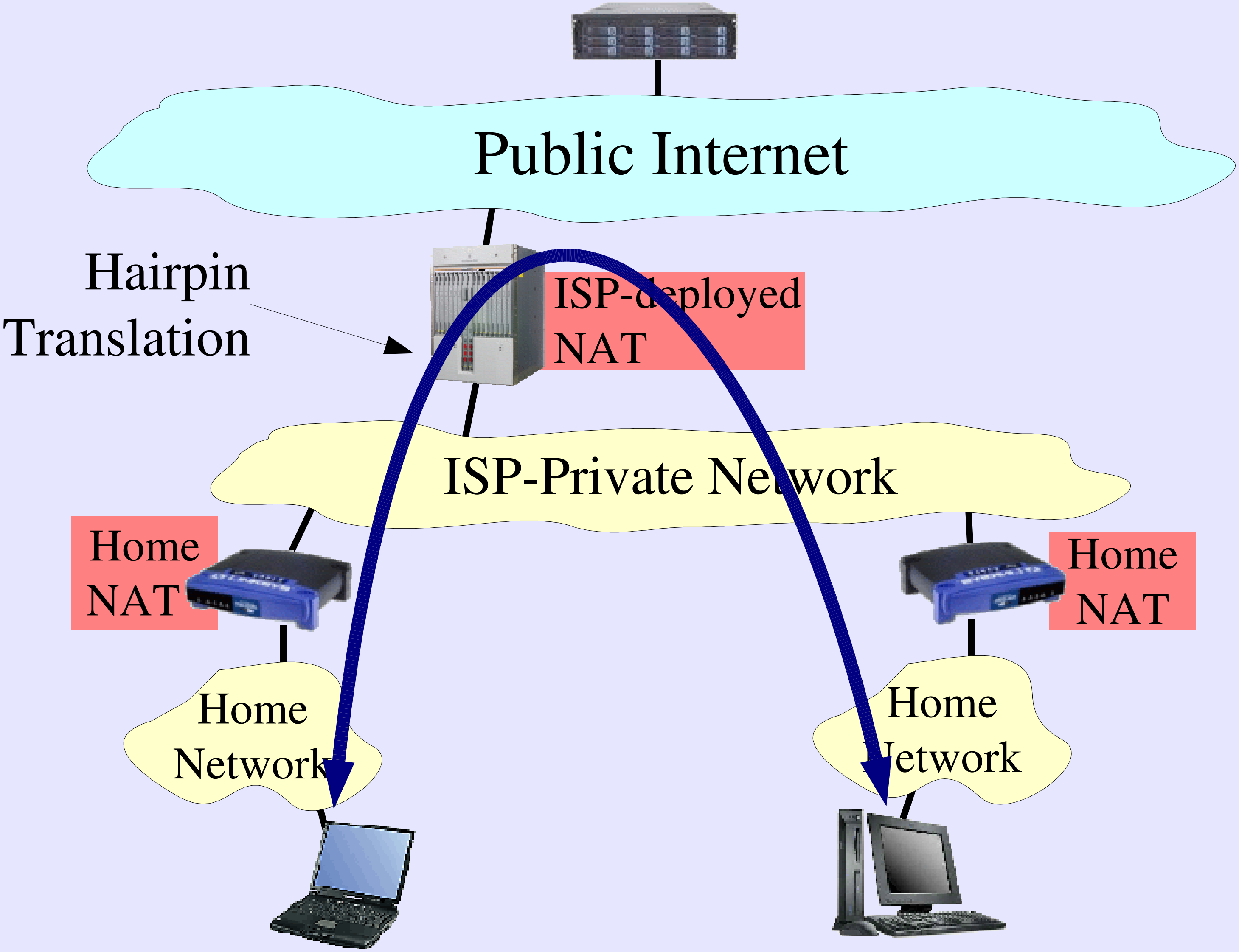
# Multi-Level NAT



# Multi-Level NAT



# Multi-Level NAT



# NAT Check

Tests hole punching “end-to-end” from user's host

- Results reflect composition of all NAT(s) in path
- No isolation of contention-related “bad” behaviors
- No tests for “bad but semi-predictable” behaviors

More detailed tests of specific NATs elsewhere [Jennings–STUN, Guha–STUNT]

**http://midcom-p2p.sourceforge.net/**

# Data Collection

Results submitted over Web by (self-selecting) community of volunteers

- UDP: 380 data points
- TCP: 286 data points

Covers

- NAT router hardware from 68 vendors
- NAT support in 8 popular operating systems

(Breakdown by vendor in paper)

# Testing Results

## UDP Hole Punching

- 82% of NATs support
- Most common NATs:

| | |
|---|---|
| – Linksys | 98% (45/46) |
| – Netgear | 84% (31/37) |
| – Windows | 94% (31/33) |
| – Linux | 81% (26/32) |

- Hairpin: 24%

## TCP Hole Punching

- 64% of NATs support
- Most common NATs:

| | |
|---|---|
| – Linksys | 87% (33/38) |
| – Windows | 52% (16/31) |
| – Netgear | 63% (19/30) |
| – Linux | 67% (16/24) |

- Hairpin: 13%

# Related Work

- UDP hole punching: [Kegel 1999]
  - Voice over IP: SIP/ICE [Rosenberg 2003]
- Asymmetric TCP hole punching
  - NUTSS, NATBLASTER, NatTrav
  - Sometimes compensate for bad NAT behaviors, but more complex, timing-sensitive
- Proxy protocols
  - SOCKS, RSIP, MIDCOM, UpnP
    require explicit NAT support, user setup

# Conclusion

- NAT is evil, but is here to stay
- Hole punching enables practical, automatic traversal of majority of existing NATs
- Compatibility good for UDP, tolerable for TCP, increasing with NAT vendor awareness (hint, hint)