

JPunch

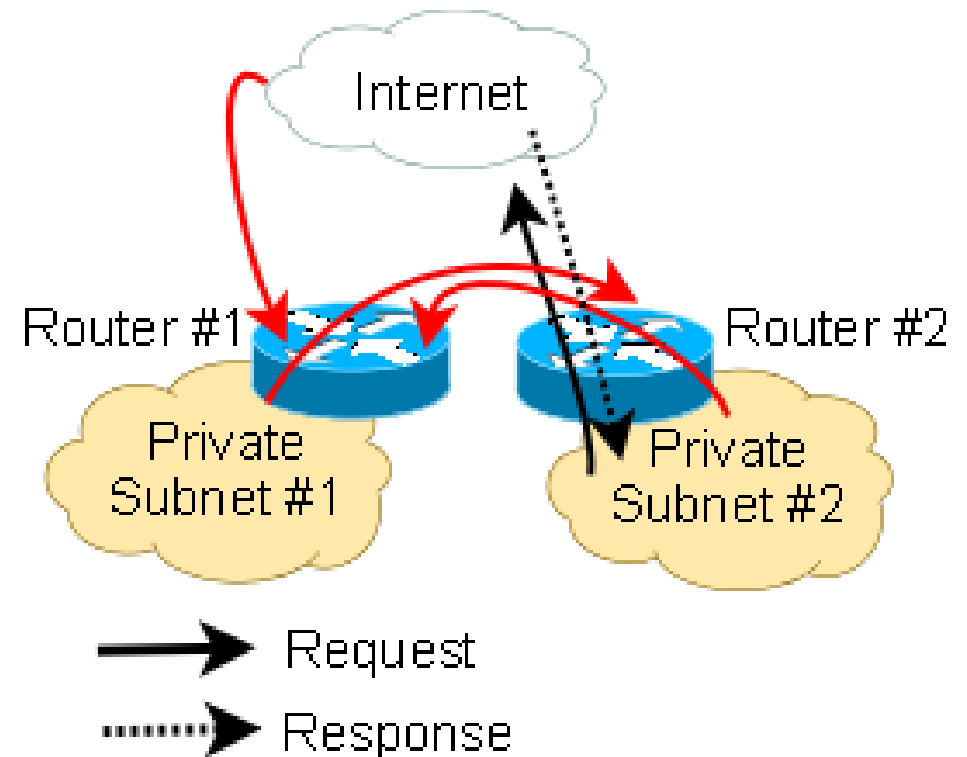
Java library for establishment of
peer-to-peer connections

Outline

- Definitions
 - NAT, UDP Hole Punching
- The Problem
 - Prerequisites, Consequences
- Solution
 - Types of NAT, Traversal techniques, Signaling Channel
- JPunch
 - Overview, Demo

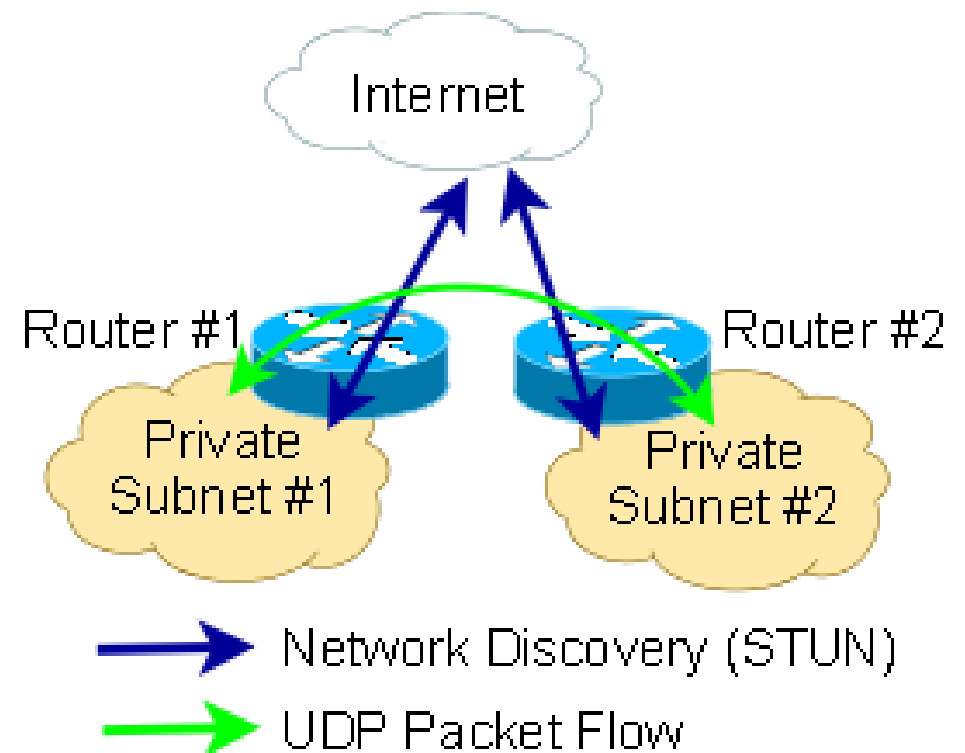
Network Address Translator (NAT)

- The part of routing device which modifies network address information in datagram packet headers while routing the packet between subnetworks



UDP Hole Punching

- method for establishing bidirectional UDP connections between Internet hosts in private networks using NAT



Prerequisites

- Network Address Translator (NAT) RFC1631
 - Introduced in 1994 to solve IPV4 storage problem
 - Announced as a “short-term solution”
 - Brought the “private” subnets
 - Allows “private-to-public” connections
 - Denies “public-to-private” connections
 - Denies “private-to-private” connections

Consequences

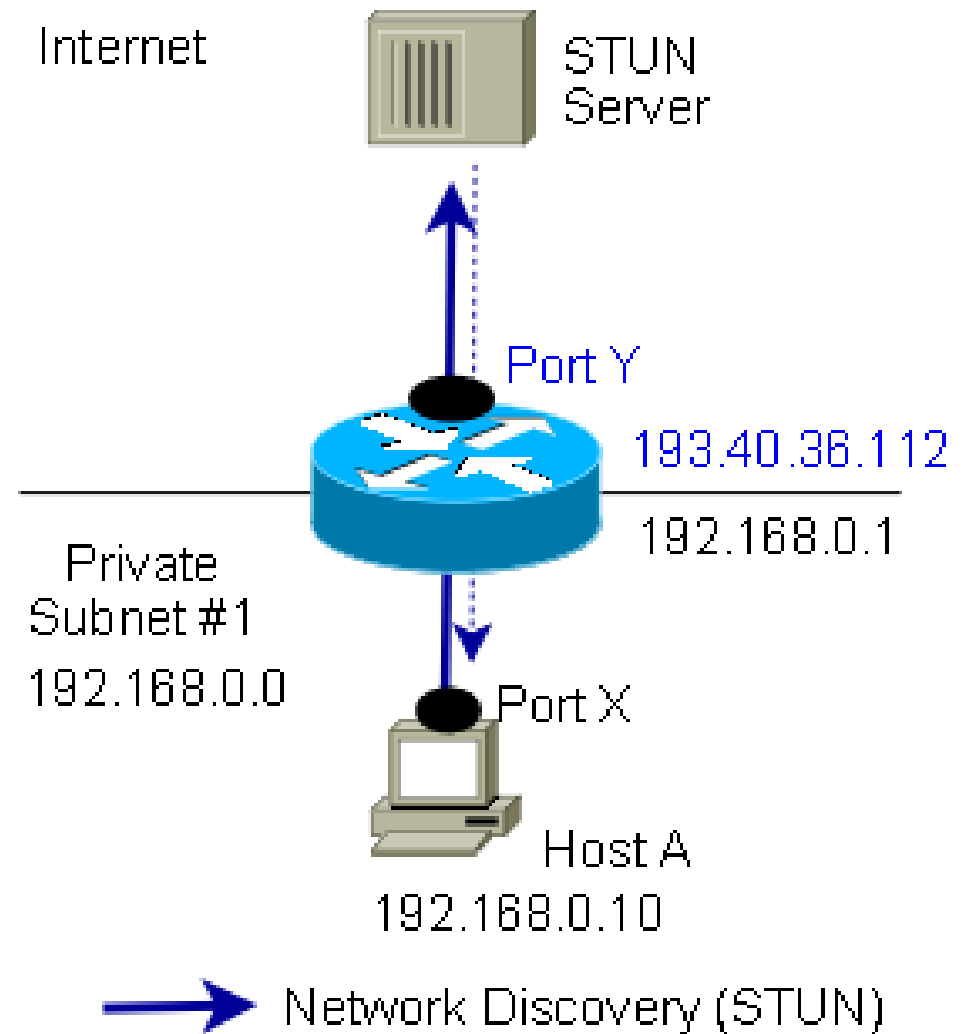
- Globally Unique Address model broken
- Peer-to-Peer connectivity complicated
- For the developers
 - Additional knowledges required
- For the enterprises
 - Additional costs introduced

Solution

- Discover the presence and behavior of NAT
- Discover the mapped (bound) address
- Exchange the bound addresses
- Send UDP packets to the remote address
- Listen for incoming UDP packets
- Receiving the packets means the data can be transferred

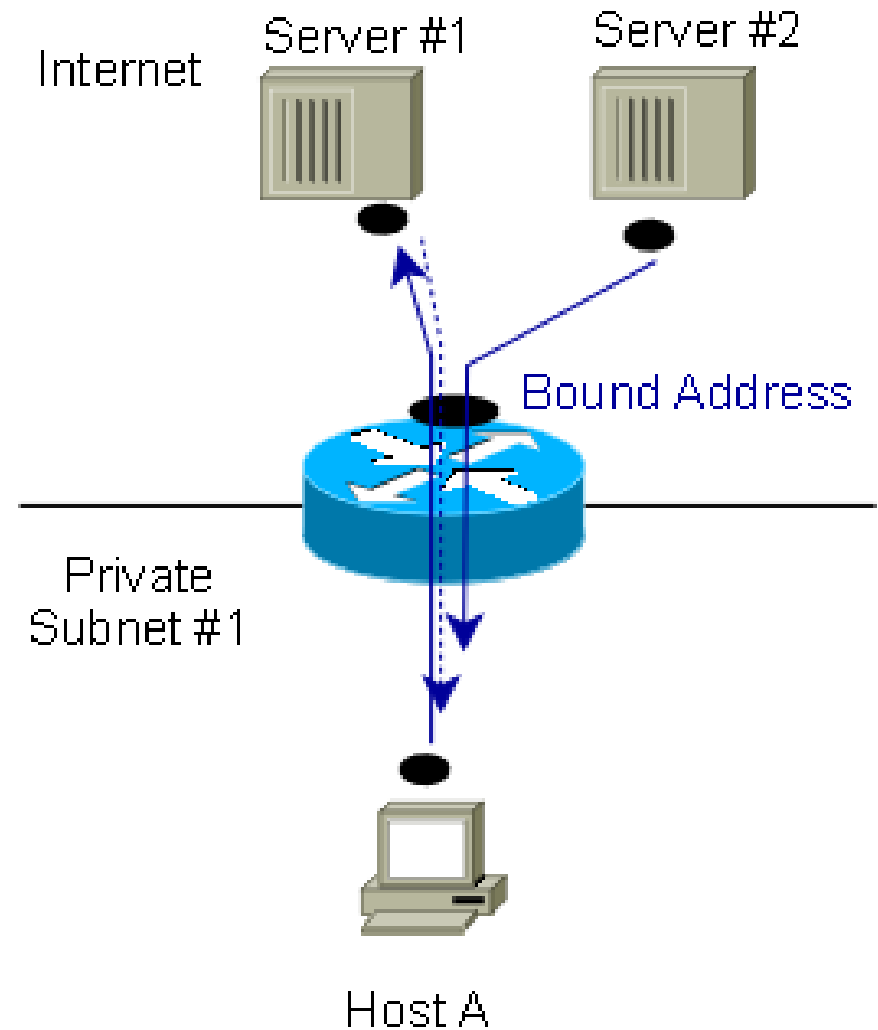
How to discover NAT ?

- STUN - Simple Traversal of UDP through NAT
 - Client-server protocol
 - STUN server returns **bound address and port**
- Compare the returned bound address to the **local one**



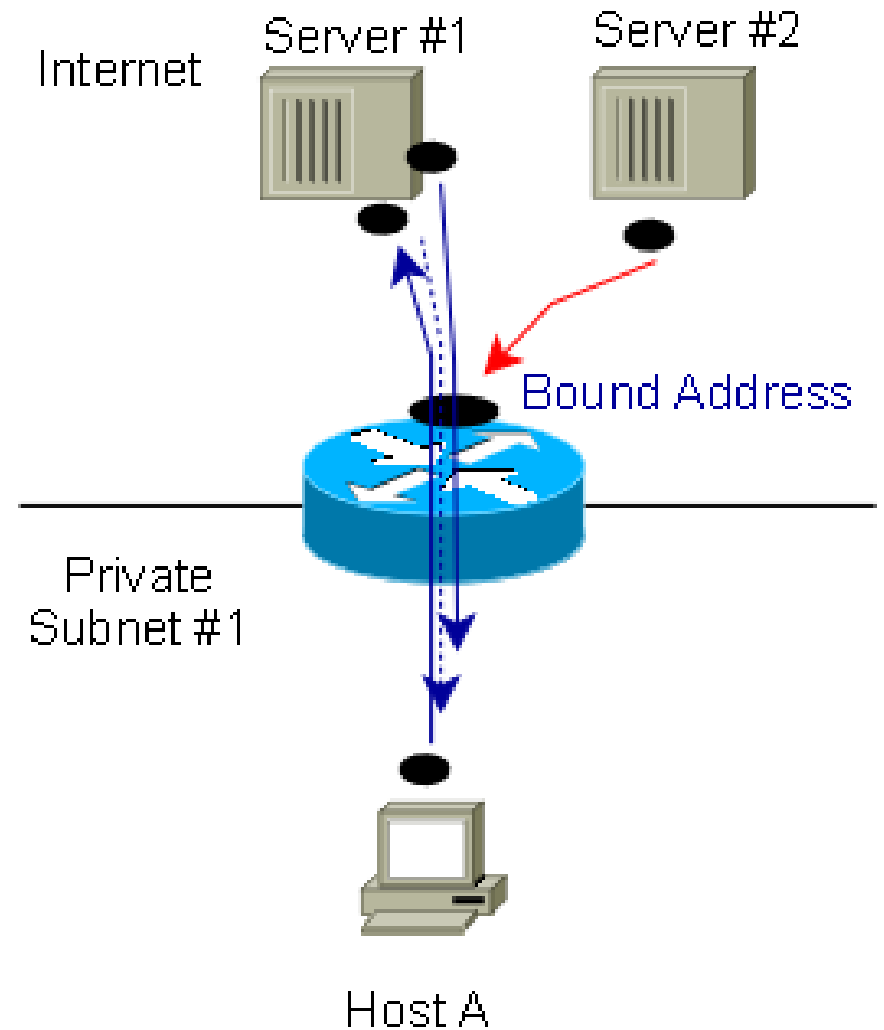
Types of NAT

- Full Cone
 - Once the binding is created it can be used by any host from outer network



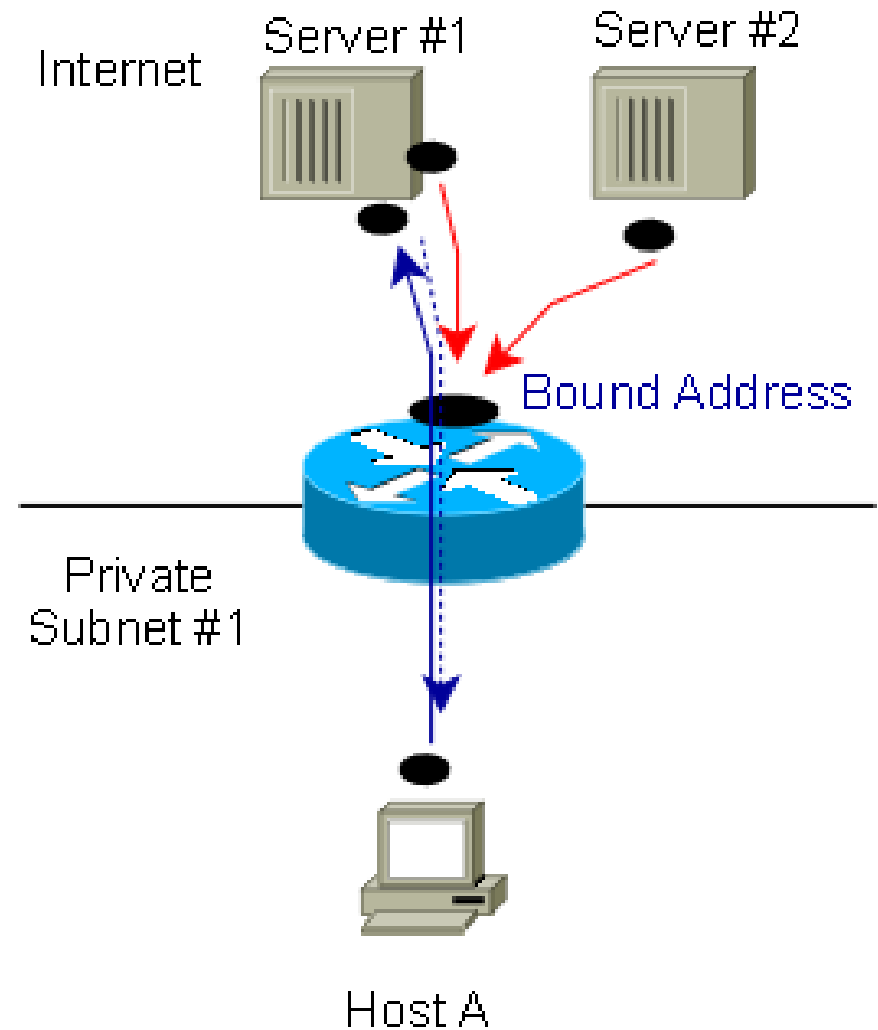
Types of NAT

- Restricted Cone
 - Created binding can be used by any port of called host
 - Created binding can not be used If host was not called previously using this binding



Types of NAT

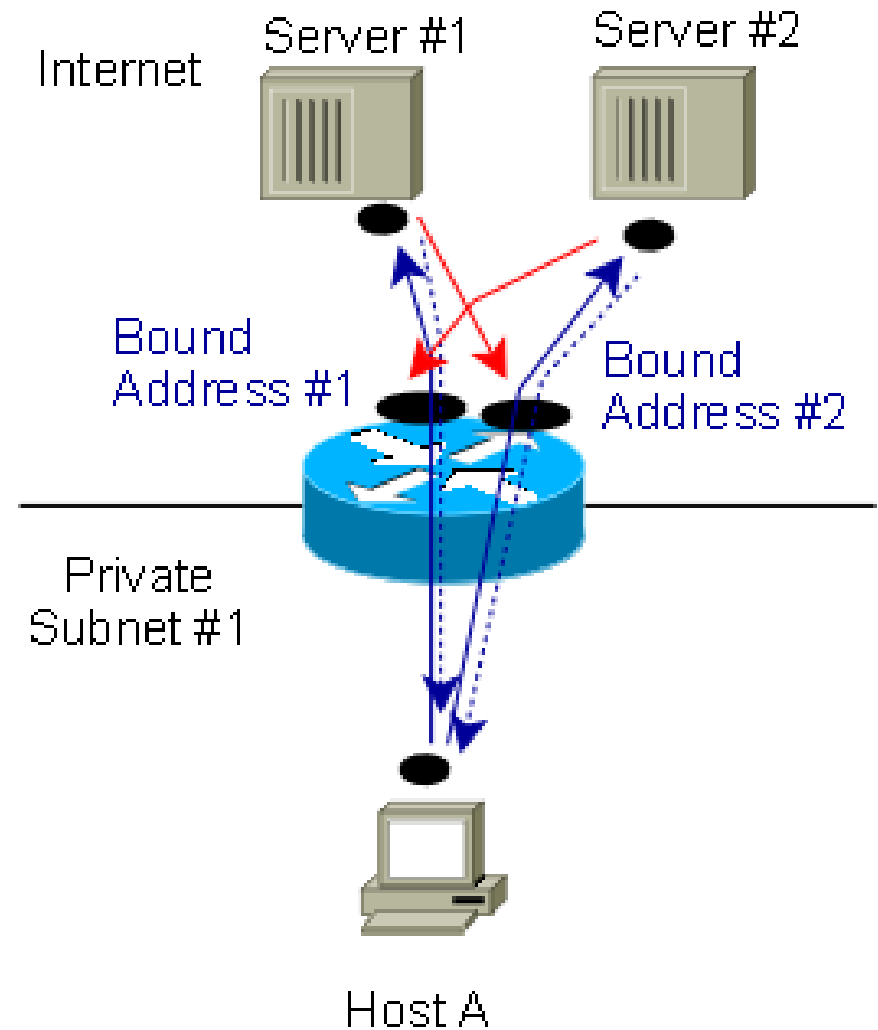
- Port Restricted Cone
 - Created binding can not be used If specific address+port combination was not called previously using this binding



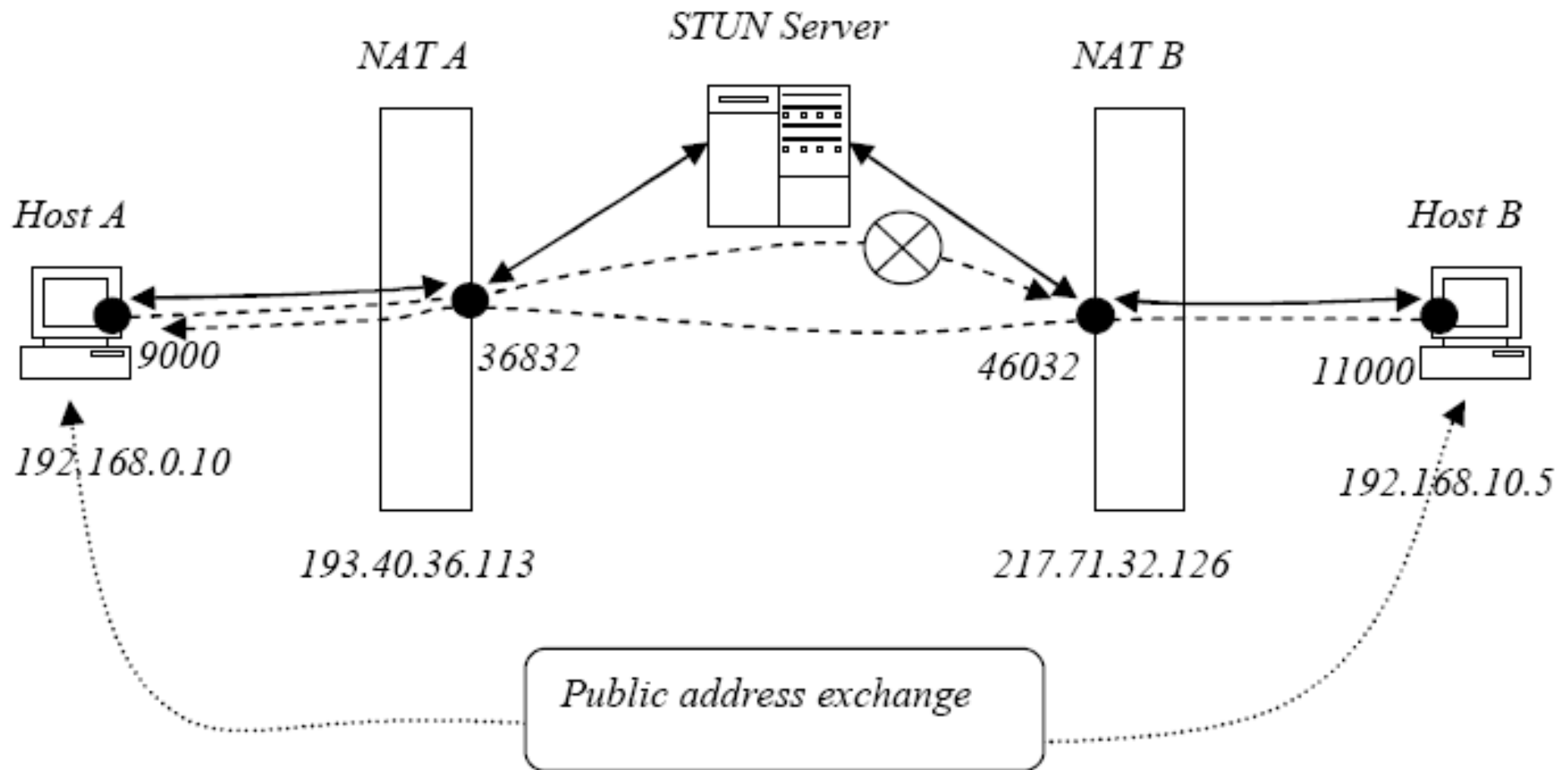
Types of NAT

- Symmetric

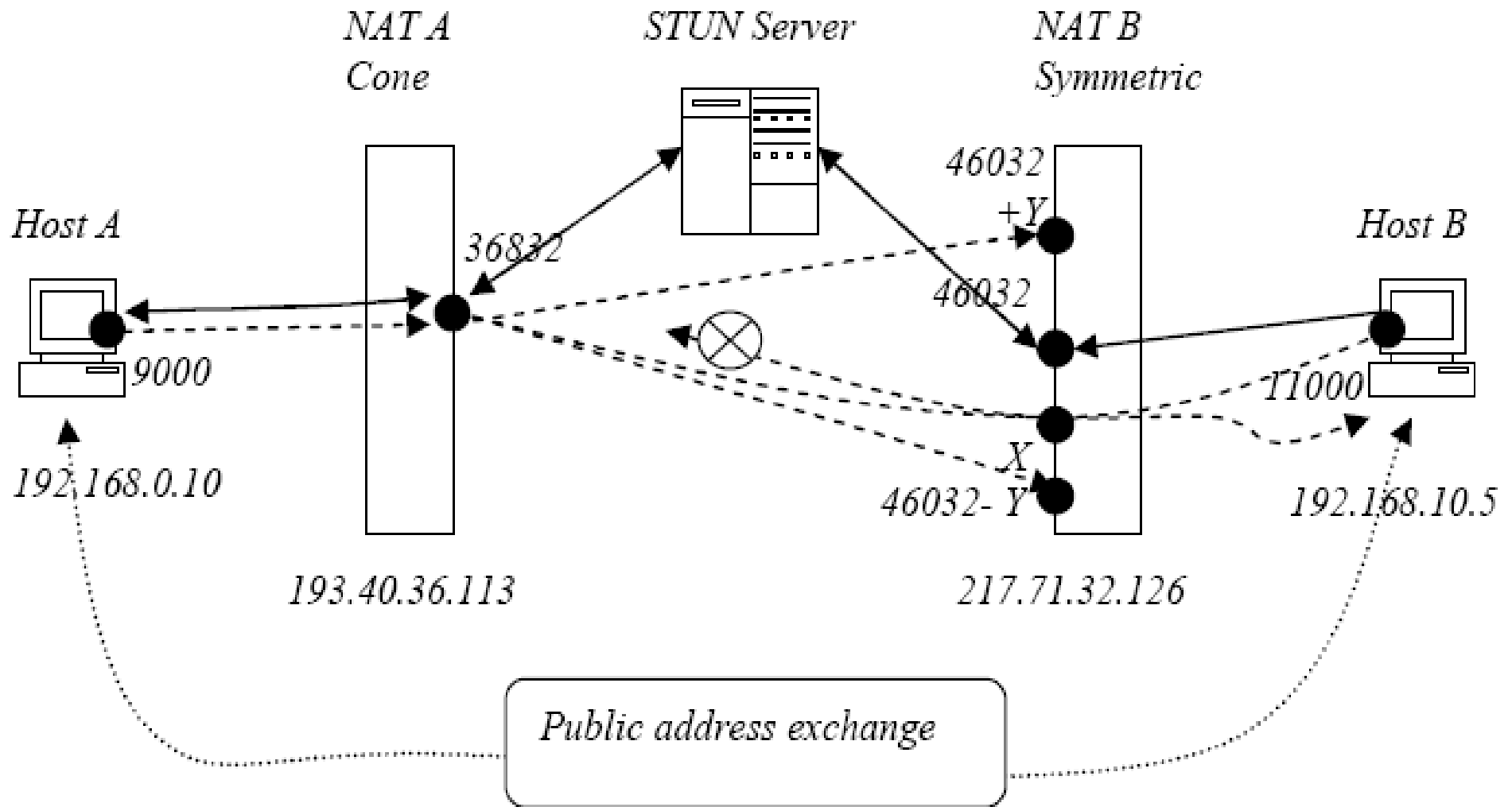
- Different addresses can not be called using the same binding
- Each new address request will create a new binding
- Port allocation rule depends on vendor, but basically is “+1”



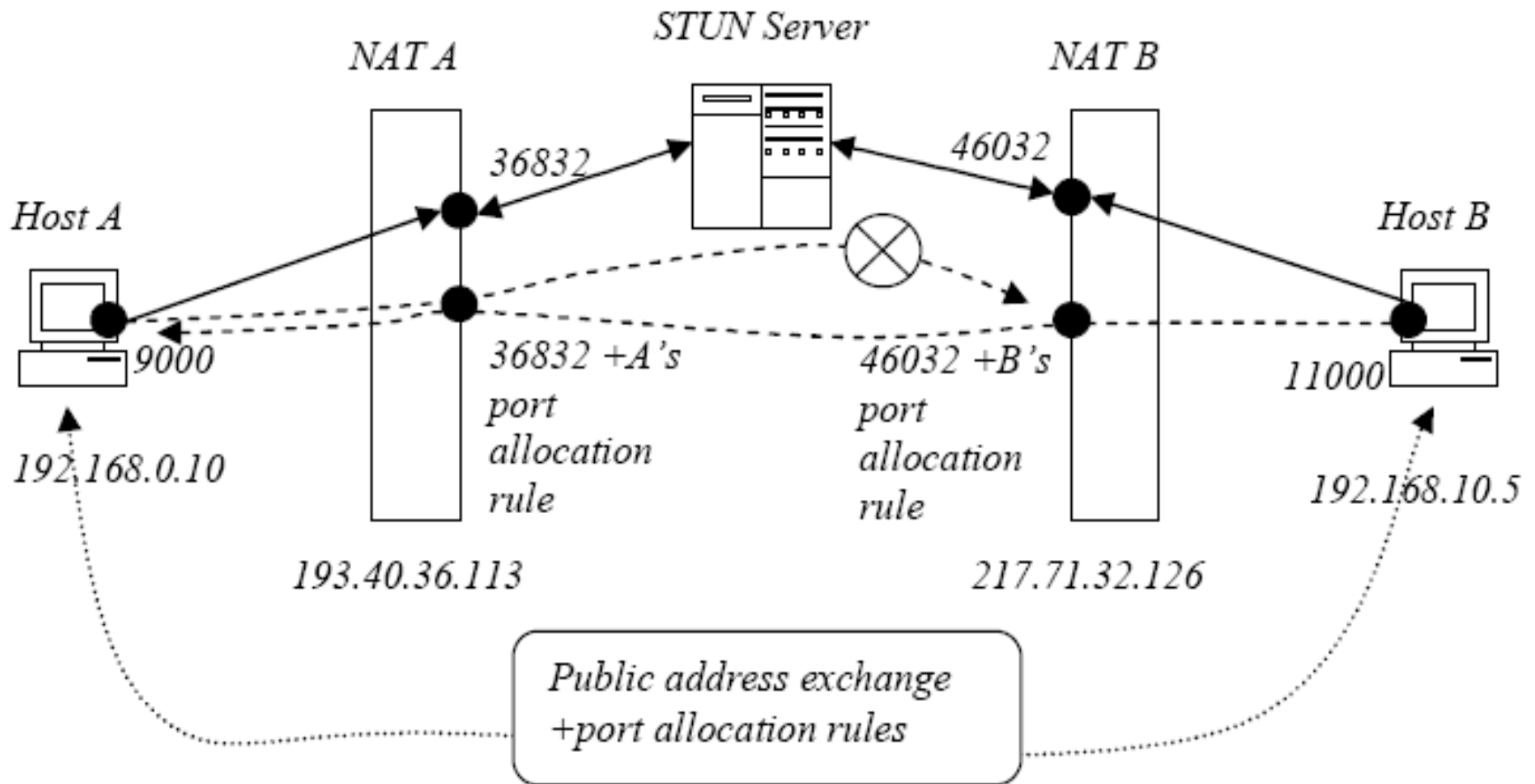
Cone-to-Cone traversal



Cone-to-Symmetric traversal



Symmetric-to-Symmetric



Primary (signaling) channel

- Hosts need to exchange their network information
 - Bound addresses, types of NAT
- Any relayed connection can be used:
 - Instant Messaging (F2F)
 - Ssh tunnels
 - Ssh + I/O redirection (jPunch)

SSH + I/O redirection

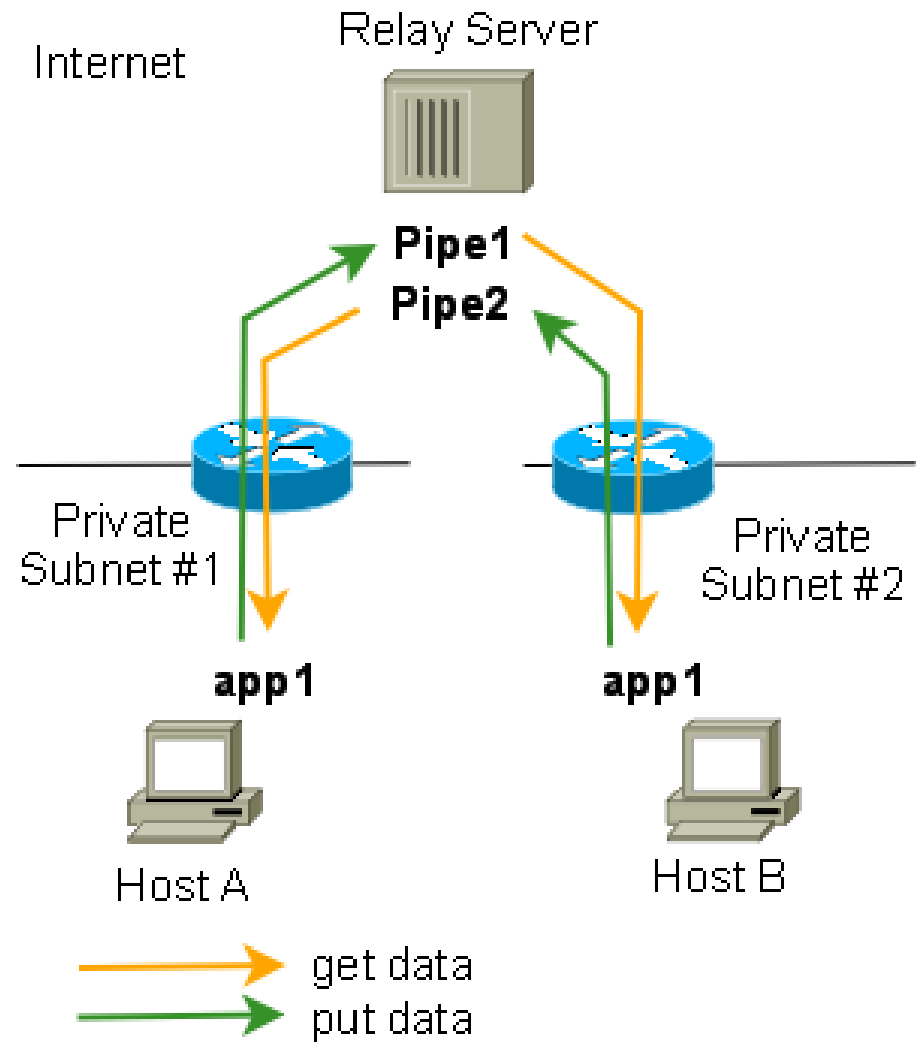
- Select some public server for relaying
- Create *fifo* pipes for each participating host
- Redirect the input/output of the desired application:
 - Put the output into remote pipe
 - Read the input from the remote pipe
- Use the `ssh` for handling remote pipes

SSH + I/O redirection

- Creating *fifo* pipes on server
 - *user1@server:~ mkfifo /tmp/pipe1*
 - *user1@server:~ mkfifo /tmp/pipe2*
- Redirect from remote pipe1 and to *app1*
 - *user1@hostA:~ ssh user1@server "cat /tmp/pipe1" | app1*
- Redirect data from *app1* to remote pipe2
 - *user1@hostA:~ app1 | ssh user1@server "cat >> /tmp/pipe2"*
- Combined Input and Output redirection
 - *ssh-in | app1 | ssh-out*

SSH + I/O redirection

- The channel between the *app1* instances provided by *ssh* and *I/O redirection*



jPunch

- Java library for establishment of p2p connections
 - Network discovery
 - Provided by *jStun* library (STUN client)
 - Signaling channel
 - SSH + I/O redirection (by defaults)
 - Any other relay channel provided using API
 - Traversing methods
 - UDP hole punching (from F2F 1.0)
 - Data Transmission
 - Reliable UDP (from F2F 1.0)

jPunch

- Can be used standalone
 - Needs server for relaying
 - No GUI
- Can be included or extended using API
 - Adding alternative signaling channels
 - Adding GUI

jPunch

- Problems
 - Lower transfer speed (than possible maximum)
 - Because of not optimal reliable UDP
 - Still problems when traversing symmetric-to-symmetric NATs

jPunch

- Next
 - Improve reliable UDP transmission
 - Add support for UPnP and NAT-PMP
 - Add support for TURN servers
 - Try TCP traversal (by specification)

Questions?

Thank you!