CS419: Computer Networks

Lecture 8: Mar 28, 2005

Firewalls and NATs

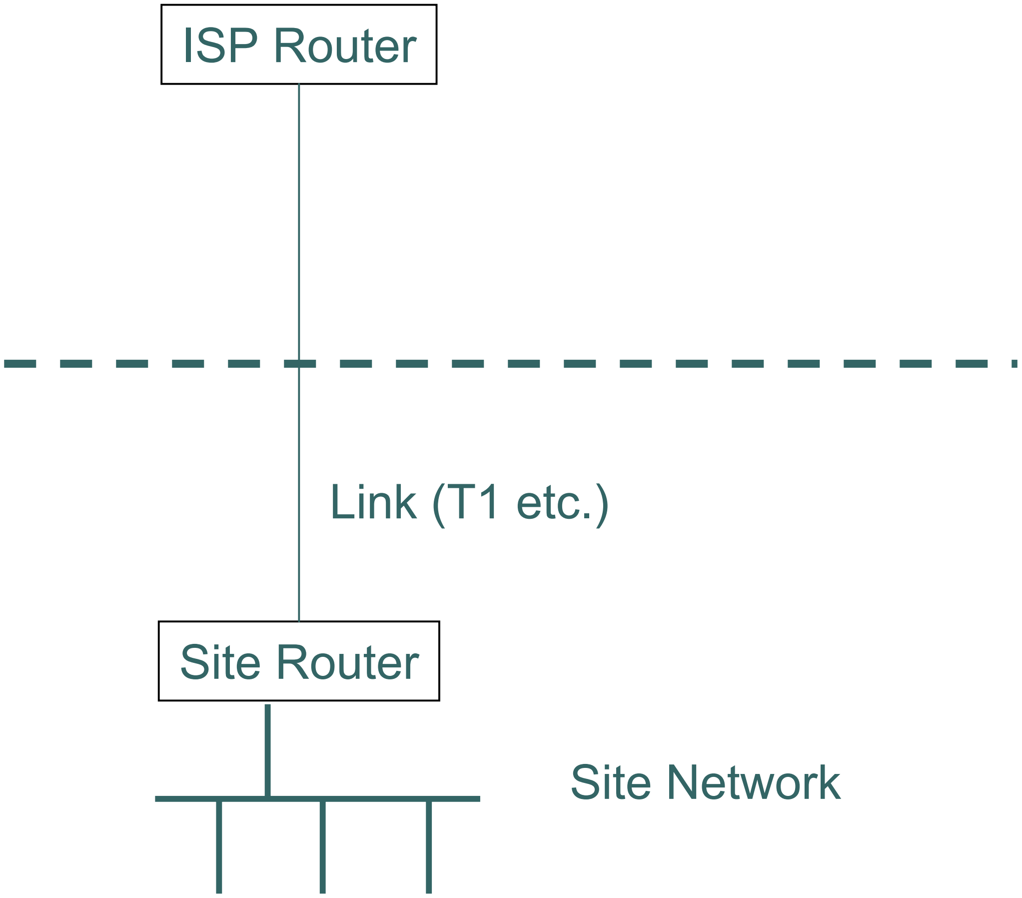
Network security topics

CS419

- I'm going to limit "network security" to three topic areas:
  - Network access issues (user or host authentication, and VPNs)
  - Site protection issues (firewalls and VPNs)
  - Flow encryption issues (including key distribution)
    - IPsec at network layer
    - TLS or SSL or SSH at transport layer
- I'm excluding application-level security, like S/MIME or secure email, as well as Kerberos

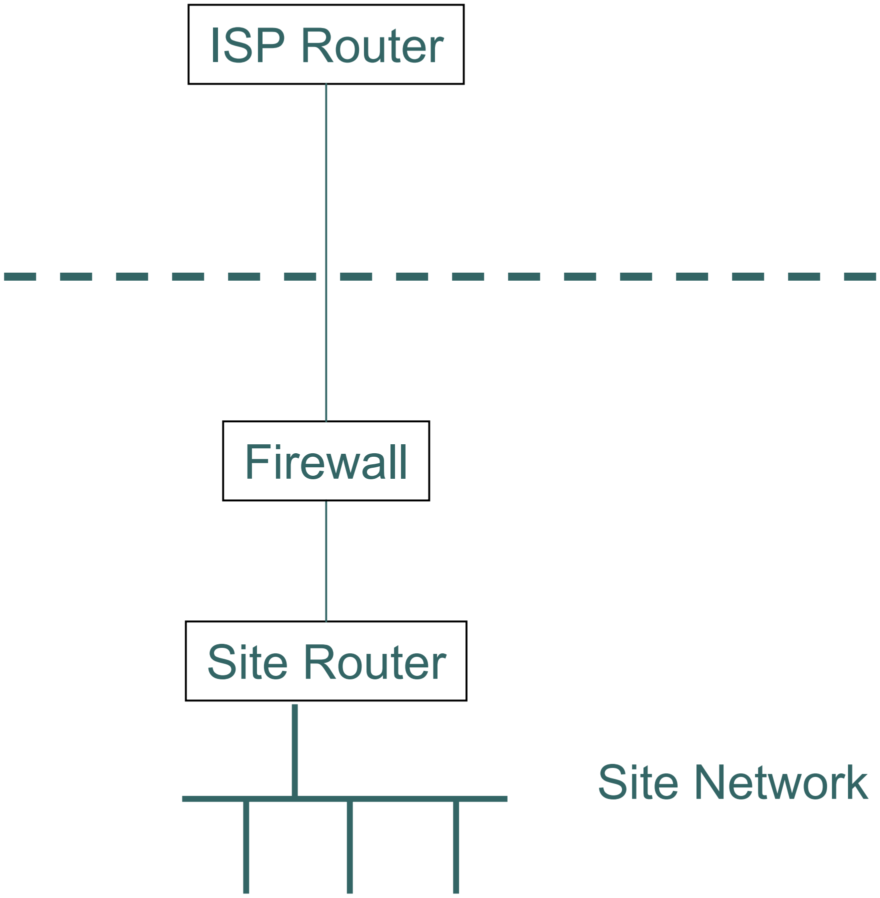
Site with no firewall

CS419

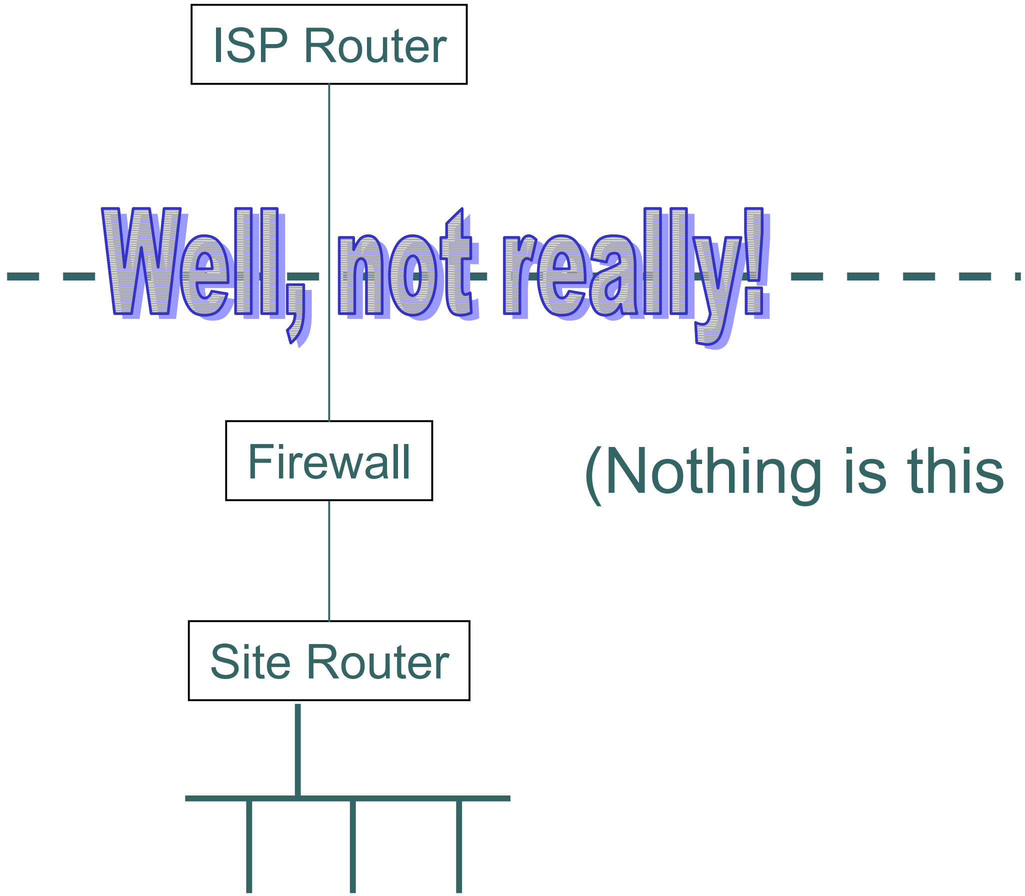


Site with firewall

CS419



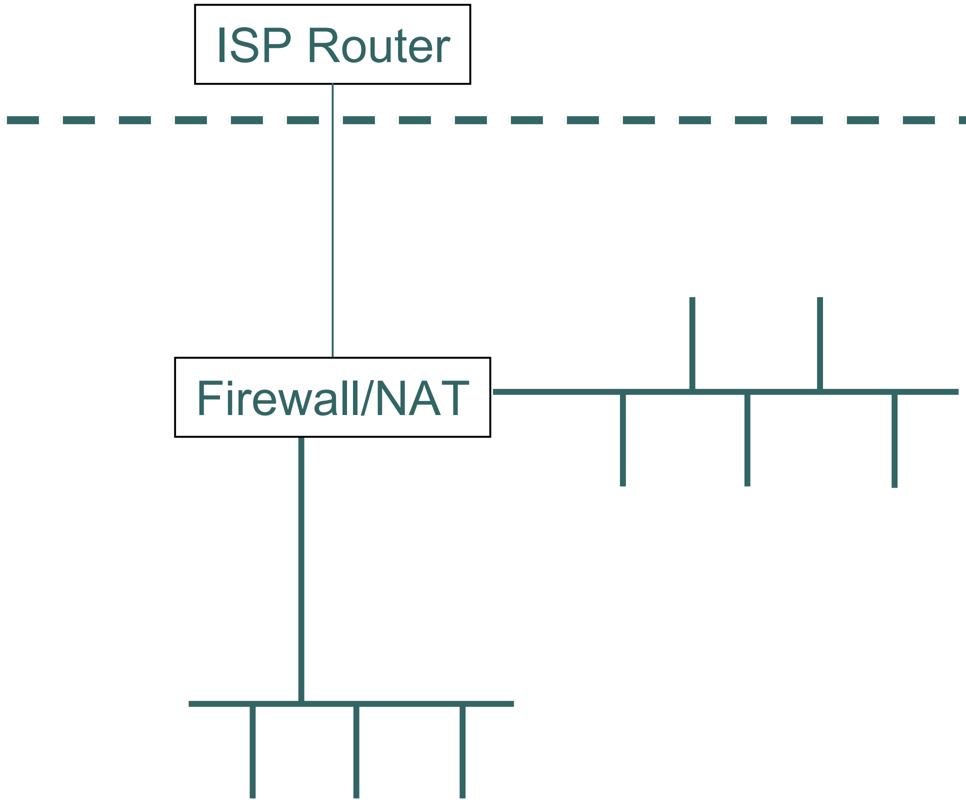
Site with firewall



(Nothing is this simple!)

DMZ ("De-Militarized Zone")

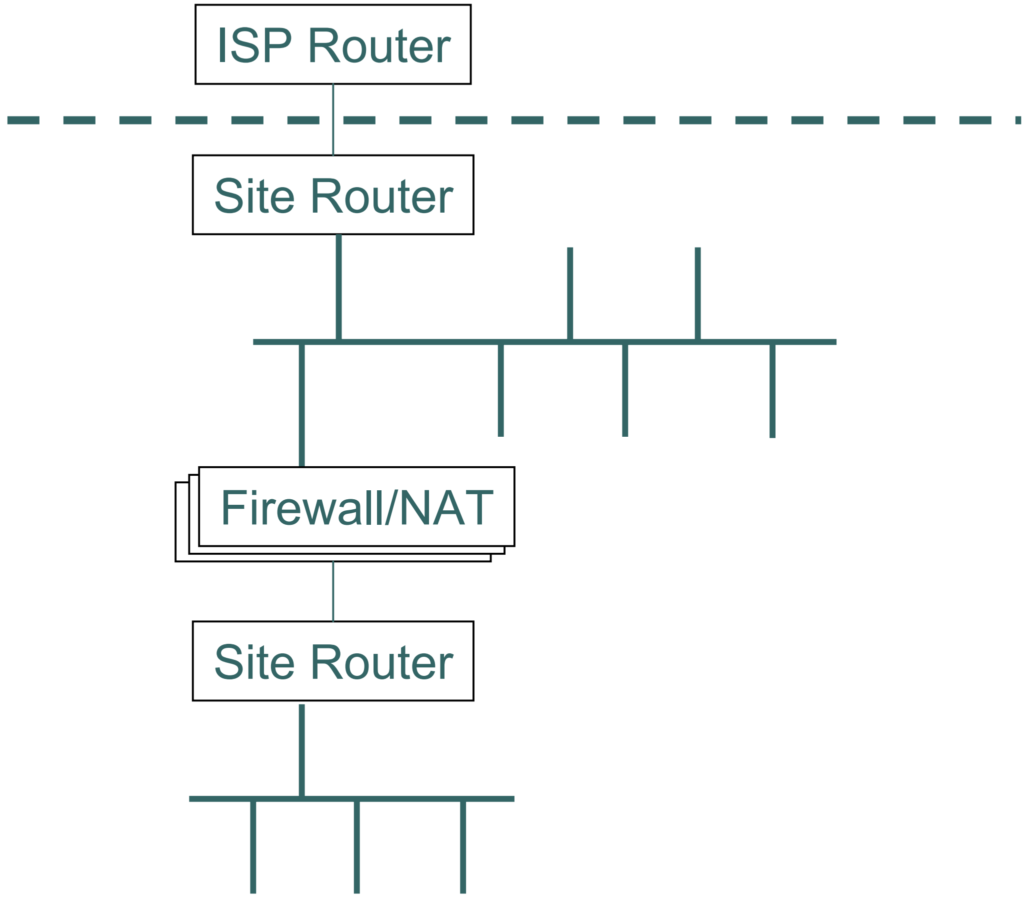
CS419



DMZ:
Network outside of Site security perimeter used to deploy firewall(s) and publicly available services (Web, FTP, DNS, etc.)

Various DMZ deployments are possible

CS419



History: Firewalls were rogue components

CS419

- Firewall/DMZ architecture never part of the “official” Internet Architecture
  - Purely a commercial creation
  - Distrusted by IAB (Internet Architecture Board)
- “Crunchy on the outside, soft on the inside”
  - “All security should be end-to-end”, etc…

Firewall model held up well until recently

- Email viruses and laptops now cause havoc
  - Firewalls scan incoming email, but laptops bypass firewalls
- Nowadays sites are proactive about what can attach to the internal network
  - Newly attached hosts are scanned for latest virus software and profiles
  - More and more, internal switches have firewall functionality, monitor all traffic!

Firewalls not just protection from outside attackers

CS419

- Bandwidth control
  - Block (or choke) high volume, non-critical applications
  - Kazaa
- Employee network usage control
  - Block games, pornography, non-business uses
- Privacy
  - Don’t let outside see what you have, how big you are, etc.
  - Similar to making corporate phone directory proprietary

Firewall functions

CS419

- Dropping packets
  - According to 5-tuple and direction of packet (incoming or outgoing)
    - Recall: 5-tuple = src/dst address, src/dst port, protocol
  - According to “conversation”
    - Multiple related flows, like FTP, SIP
  - According to higher-layer info (i.e. URL, email attachments)
- Steering packets/messages
  - To other filters, like spam filter, virus checker, HTTP filter, etc.
- Logging flows and statistics

Simple firewall policy configuration

CS419

Source	Dest	App	Action
any-inside	dmz-mail	SMTP	allow
any-inside	any-outside	SMTP	drop
any-inside	any-outside	HTTP	allow
any-inside	any-outside	FTP	allow
any-inside	any-outside	any	drop
any-outside	any-inside	any	drop

Conversations

- FTP consists of two flows, control flow and data flow
- Firewall must be smart enough to read control flow, identify subsequent data flow
- True for SIP as well

Stateful and stateless firewalls

CS419

- Original firewalls were stateless
  - Maintain static filter list, but no per flow state
  - For TCP, only look at SYN
    - Means that non-SYN TCP packets are allowed even if should be blocked!
  - No concept of conversation
- Modern firewalls are typically stateful
  - Maintains dynamic list of all allowed flows
  - Better capability, harder to scale

Routing-based or callout-based steering (1/2)

CS419

- Callout-based:
  - User-customized functions may be called at specific checkpoints
    - i.e. after each individual email in an email stream
    - after each HTTP GET
  - These callouts can operate on the firewall box, or send messages to another box
    - i.e. after each mail message, local callout looks for attachments, and if found sends mail to a virus checker

Routing-based or callout-based steering (2/2)

CS419

- Routing-based
  - Packets matching policy rule sent to another box
  - Destination address may be modified to that of the box
    - if box is not promiscuous

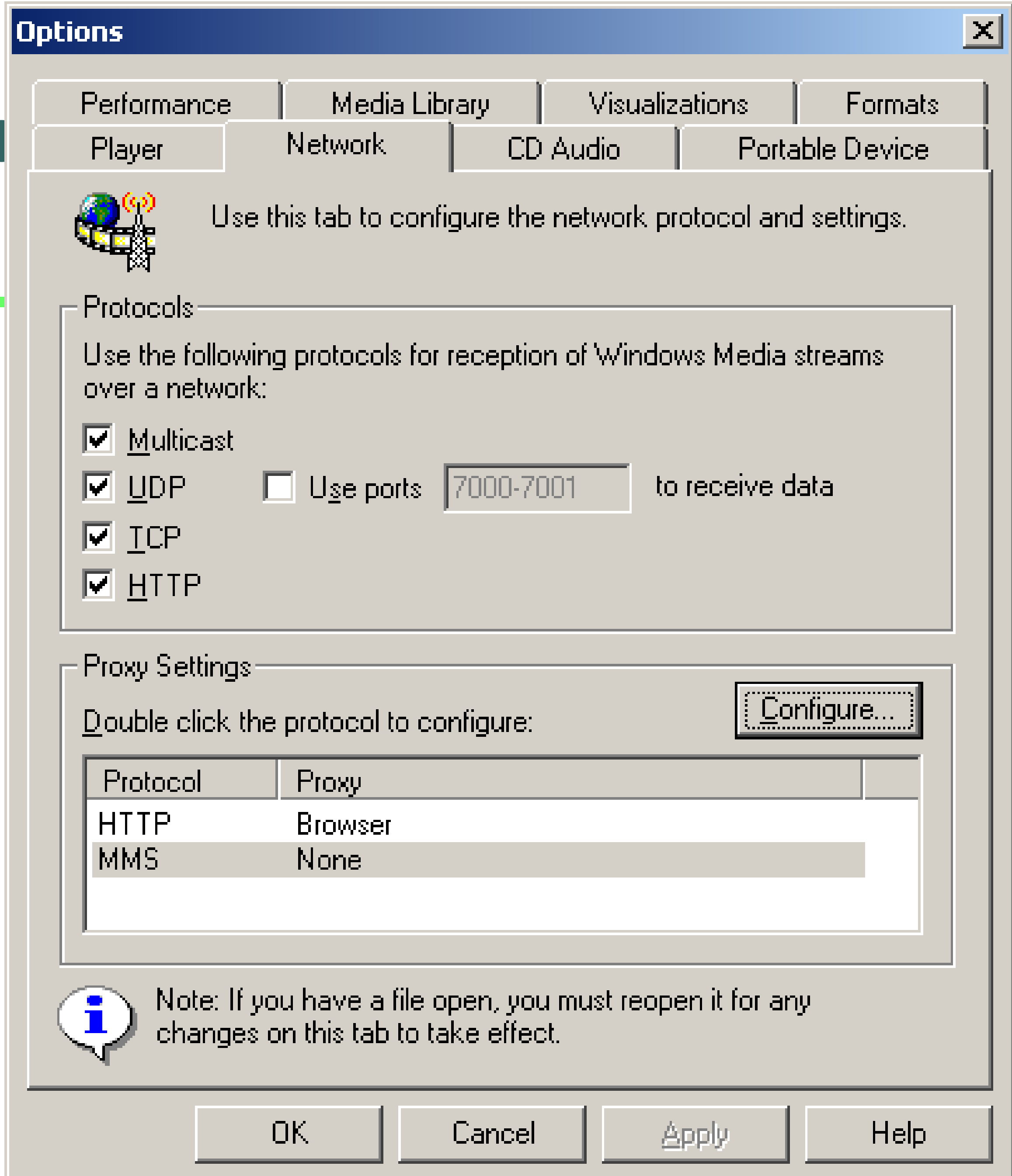
Firewall arms race

- Firewalls make it hard to introduce new applications
  - Because firewall rules tend to err on the side on prevention
- As a result, many new apps are built over HTTP
  - Or at least can fall back on HTTP if better performing protocols are blocked
  - Firewalls respond by looking deeper into HTTP/HTML, but this is hard

Case study: Windows Media

CS419

- Can run in four modes (from most to least efficient):
  1. IP multicast
  2. UDP
  3. TCP
  4. HTTP
- Windows media client will attempt to connect in the above order
- TCP firewall “holes” are simple to configure
  - TCP port 1755
  - Admin can specify which UDP ports
- Also allows a proxy in the DMZ



Windows Media client network configuration

Ethereal trace: First MMS stream

CS419

┌ Îú°è MMS ← ┌ └ ðððð ┘ └ N S P l a y e r / 7 . 1 . 0 . 3 0 5 5 ; { D 4 C 5 5 2 1 3 - 3 6 4 F - 4 C F 6 - A 7 F 6 - 9 0 F 4 D F B A 9 8 F 8 } ; H o s t : w m . s o n y . g l o b a l . s p e e d e r a . n e t ┌ Îú°p MMS ♬ ♀ ┌ ┘ ðððð ┘ └ ð? ┌ ┌ € 4 . 1 . 0 . 3 9 2 3 ┌ Îú° MMS ┘ ┌ Ház¶®GÑ? ┐ ↑ └ ┌ ùðð ┘ ┌ Îú°@ MMS ◘ ┌ ─ └ ┘ ┌ ùðð◘ ┌ ┌ Eö MMS ♀ ┐ ôýÔxé&• @ ┐ └ ┌ ùððÿÿÿÿ ┬ \ \ 1 2 8 . 8 4 . 9 9 . 2 3 1 \ U D P \ 2 3 6 6 3 ┌ Îú°@ MMS ◘ ┐ ─ ┐ ┘ ñððð◘ F u n n e l O f T h e ┌ Îú°ˆ MMS ◄ └ Zd;ßO• @☼ │ └ ┌ ÿÿÿÿ w m . s o n y . g l o b a l / P e a r l J a m / s a v e y o u f u l l v i d _ 1 0 0 . w m v ┌ Îú °ˆ MMS ◄ └ ....

Ethereal trace: Second MMS stream

CS419

┌ Îú°è MMS ← ┌ └ ┘ └ N S P l a y e r / 7 . 1 . 0 . 3 0 5 5 ; { D 4 C 5 5 2 1 3 - 3 6 4 F - 4 C F 6 - A 7 F 6 - 9 0 F 4 D F B A 9 8 F 8 } ; H o s t : w m . s o n y . g l o b a l . s p e e d e r a . n e t ┌ Îú°p MMS♬ ♀ ┌ ┘ ┘ └ ð?┌ ┌ € 4 . 1 . 0 . 3 9 2 3 ┌ Îú° MMS ┘ ┌ Tã¥›Ä♀@┐ ↑ └ ïððð ┘ ┌ Îú°@ MMS◘ ┌ ─ ┴ ┘ ïððð◘ ┌ ┌ ‚ö! ┌ ┌ Îú°` MMS♀ ┐ »I♀┐ +‡♀@ ┐ └ ïðððÿÿÿÿ ┬ \ \ 1 2 8 . 8 4 . 9 9 . 2 3 1 \ T C P \ 2 3 6 7 3 ┌ Îú°@ MMS◘ ┐ ─ ┐ ┘ ðððð ┘ F u n n e l O f T h e ┌ Îú°ˆ MMS◄ └ øSã¥›Ä♀@☼ │ └ ┌ ÿÿÿÿ w m . s o n y . g l o b a l / P e a r l J a m / s a v e y o u f u l l v i d _ 1 0 0 . w m v ┌ Îú°ˆ MMS◄ └ . . . .

Speaking of weird protocol tunneling….

CS419

- My favorite is IP over DNS
- This is actually a “legitimate” example

IP over DNS

CS419

- Wireless LAN service in Finland
- Used HTTP “captive portal” to charge users
  - First HTTP access would be steered by firewall to a billing application
    - This allows billing without new software in client host
  - Once user pays, firewall allows all packets
- But, before client can do HTTP, it needs to get a DNS reply first
  - So firewall always allowed DNS to go through
- By tunneling IP over DNS, users could get free WLAN access!

NATs and firewalls

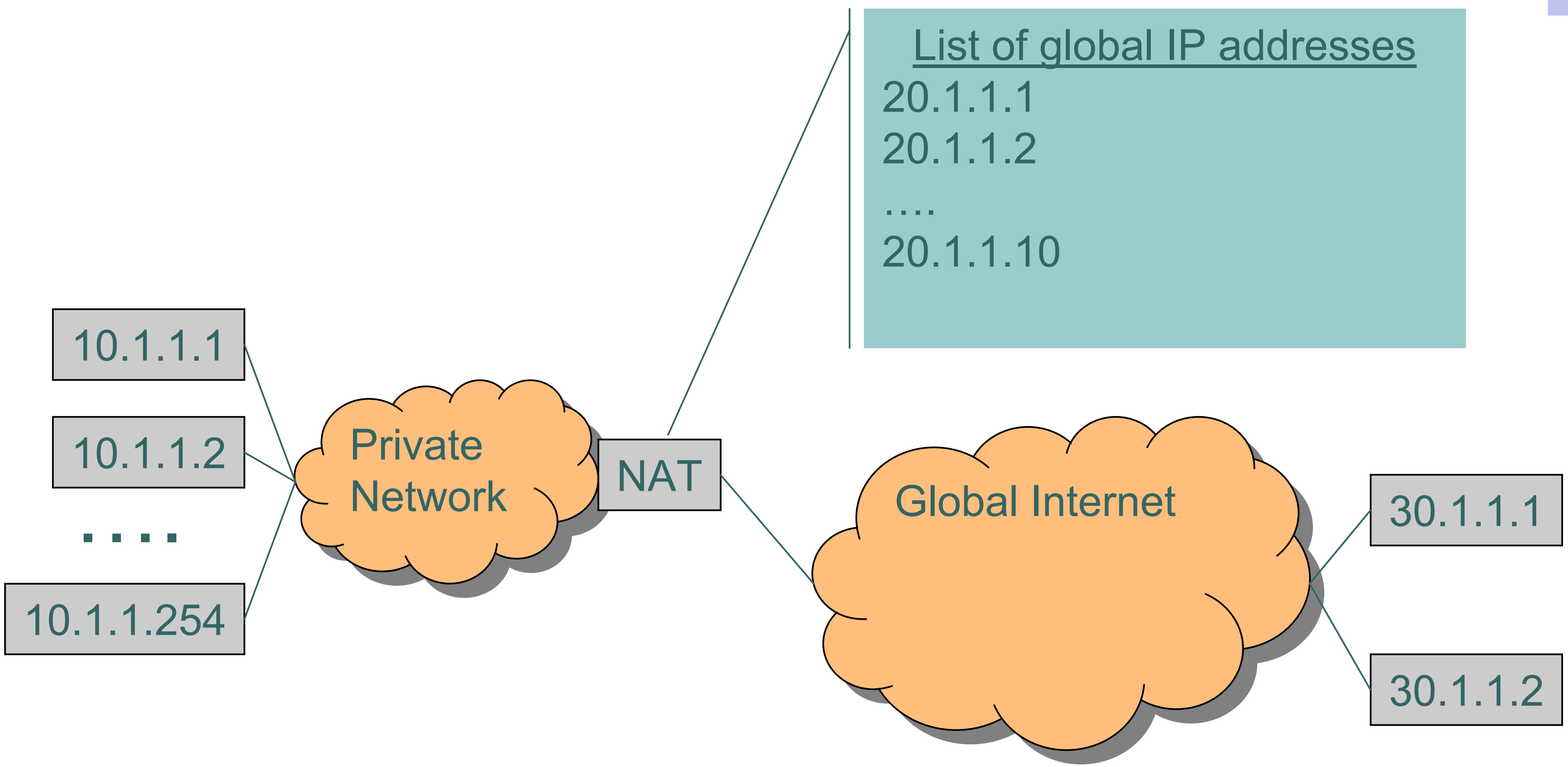
CS419

- NAT and firewall functions typically co-exist in the same box
- NAT is marketed as enhancing security
  - There may be a smidgen of truth to this, but in fact it doesn't enhance security much beyond what a firewall can do
  - Probably reduces problems with configuration errors

Network Address Translation (NAT)

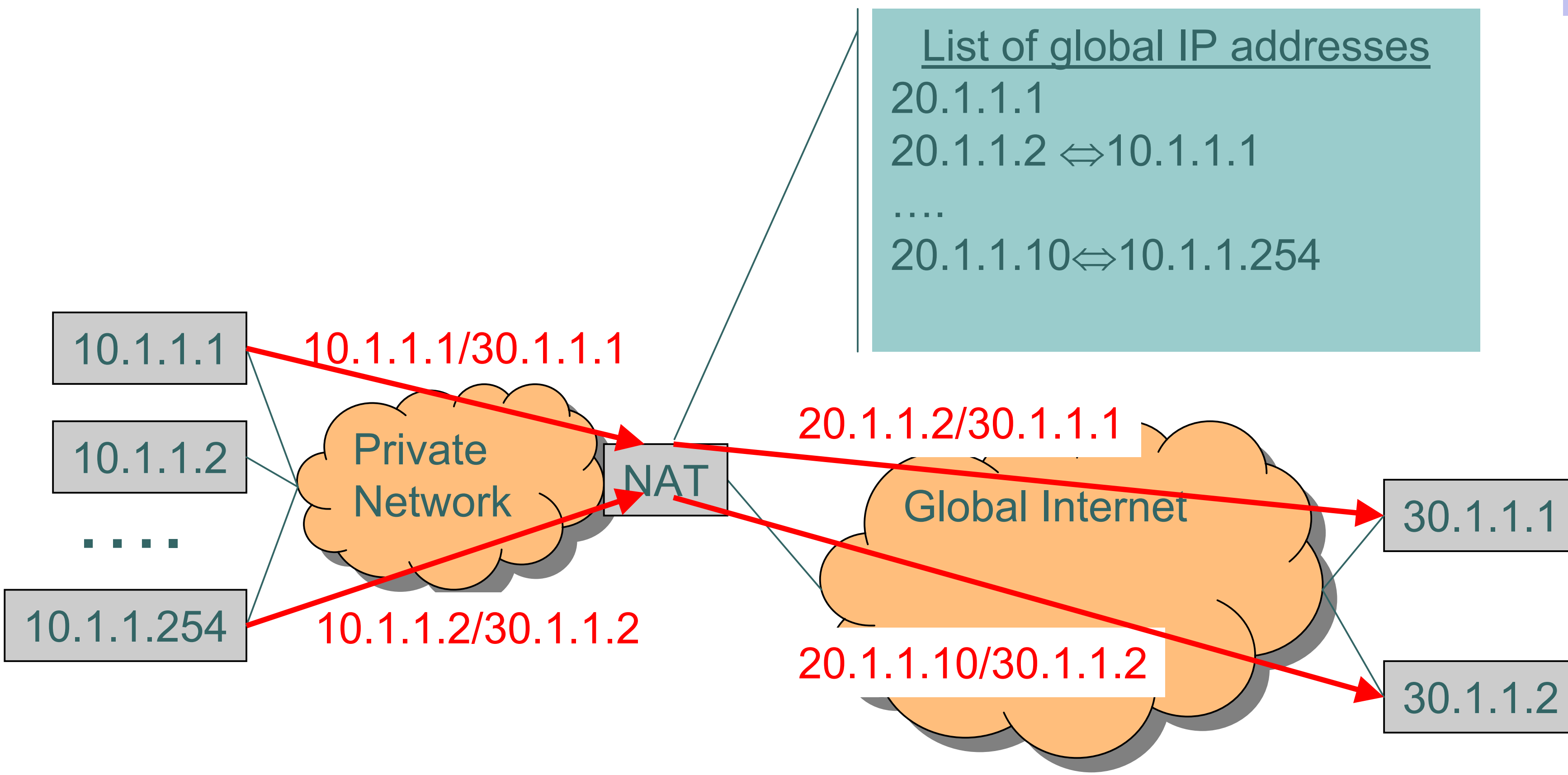
- NAT invented to solve the address depletion problem
  - In early 1990's, we thought we'd run out of IPv4 addresses by mid-to-late 1990's
  - Currently about ½ of IPv4 addresses are allocated (out of total 4 billion)
- No longer an address depletion "crisis"
- Two reasons for this:
  - Tougher allocation policies
  - NAT

Original NAT design: Global address shared over time

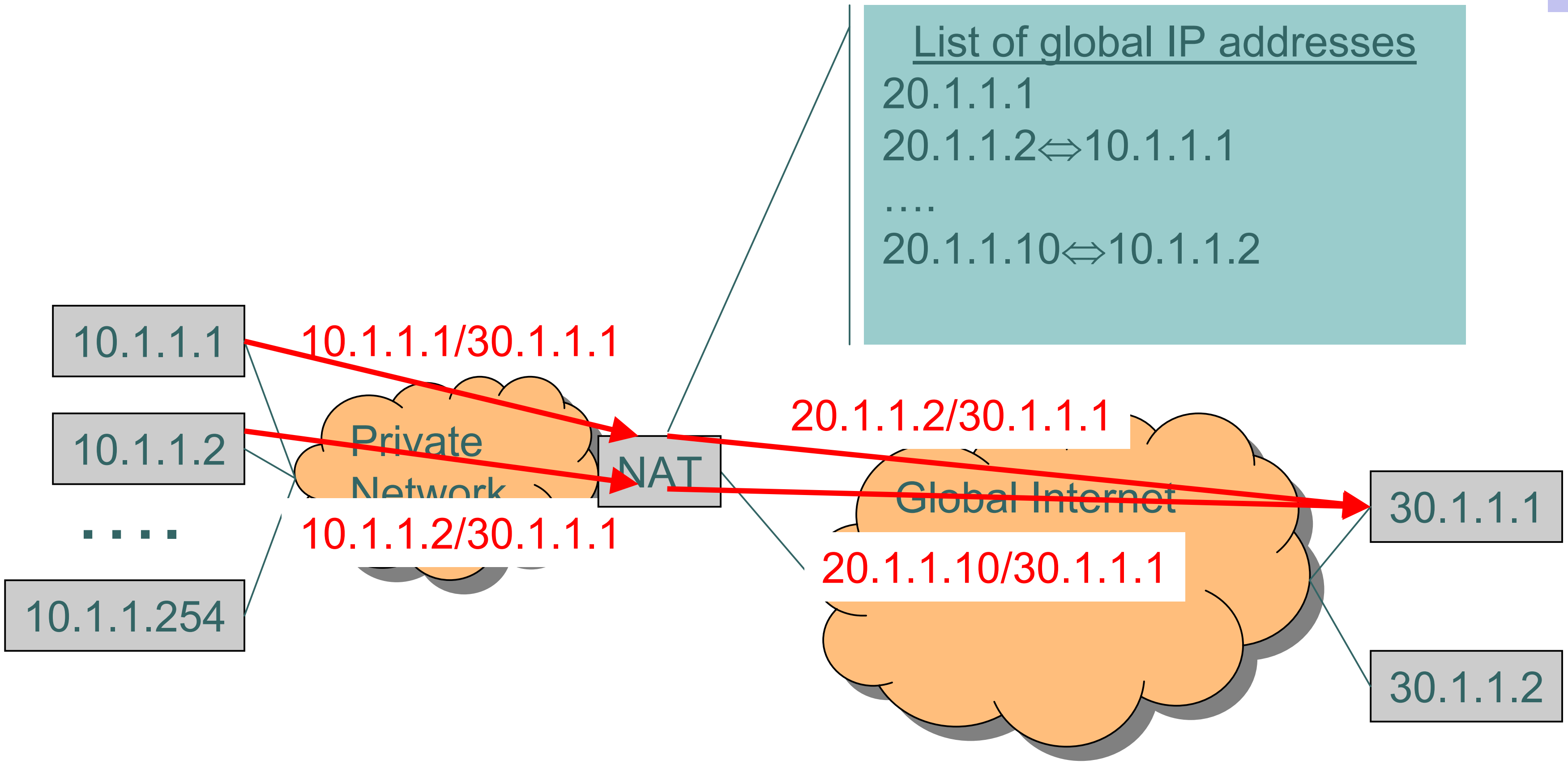


Original NAT design: Global address shared over time

CS419



Original NAT design: Global address shared over time



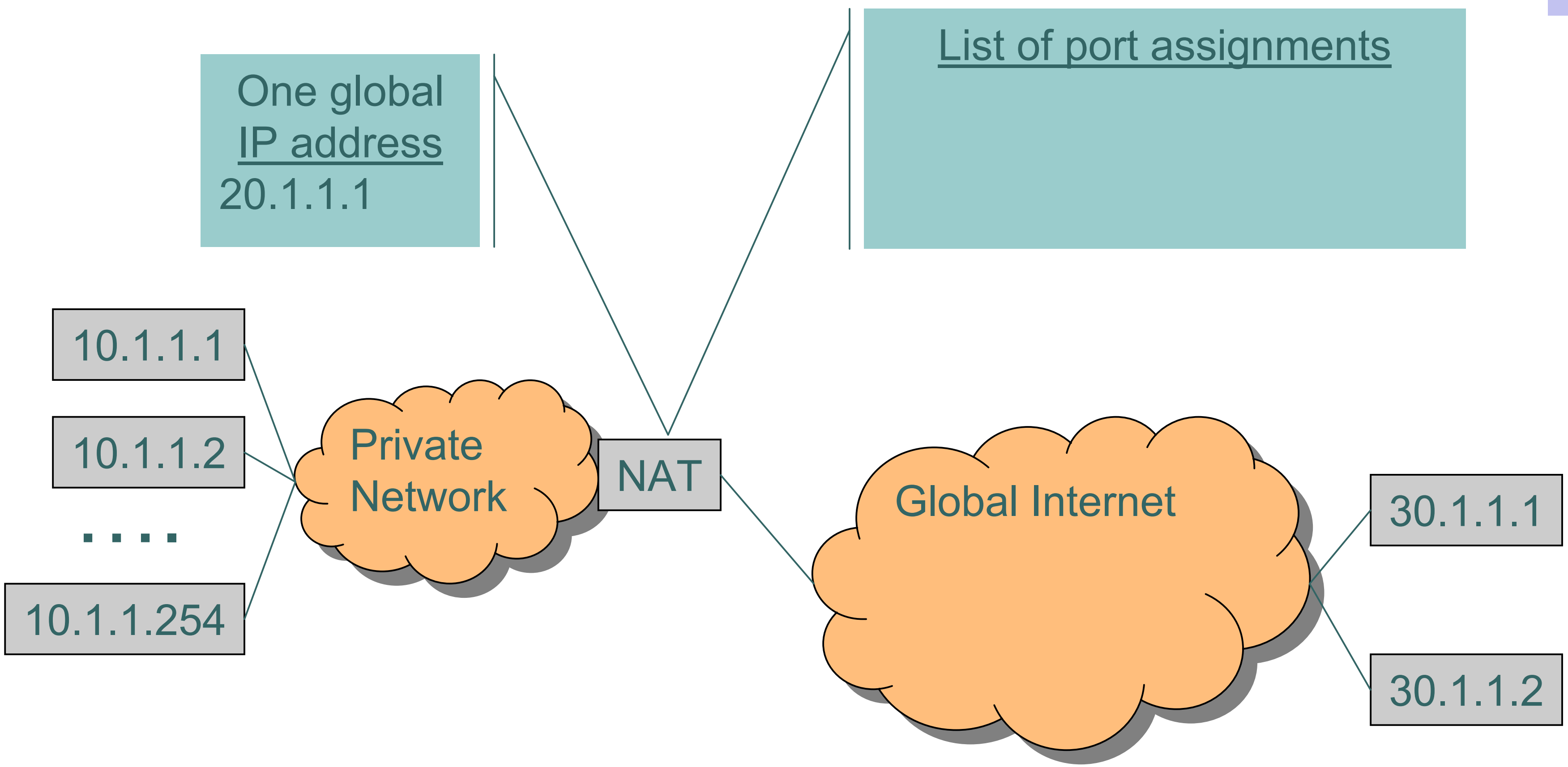
Original NAT design: Global address shared over time

CS419

- Original NAT predates the web
- Assumption was that one global address could support tens of hosts
  - Occasional FTP, etc.
- Web changed the usage model
  - More frequent global accesses
  - NAT was enhanced to allow addresses to be shared at the same time
  - Port translation (sometimes called NAPT)

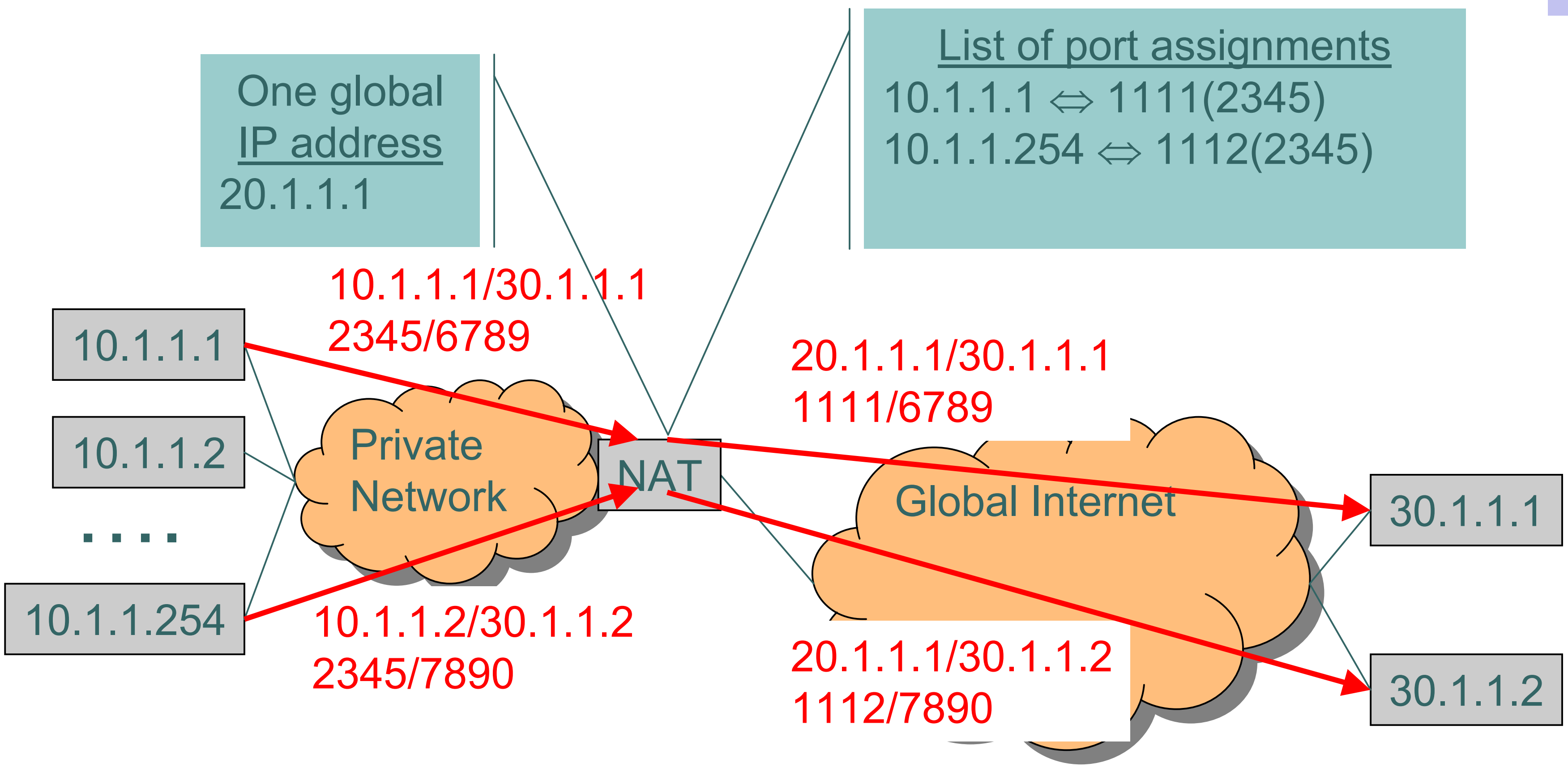
Current NAT design: Global address shared at one time

CS419



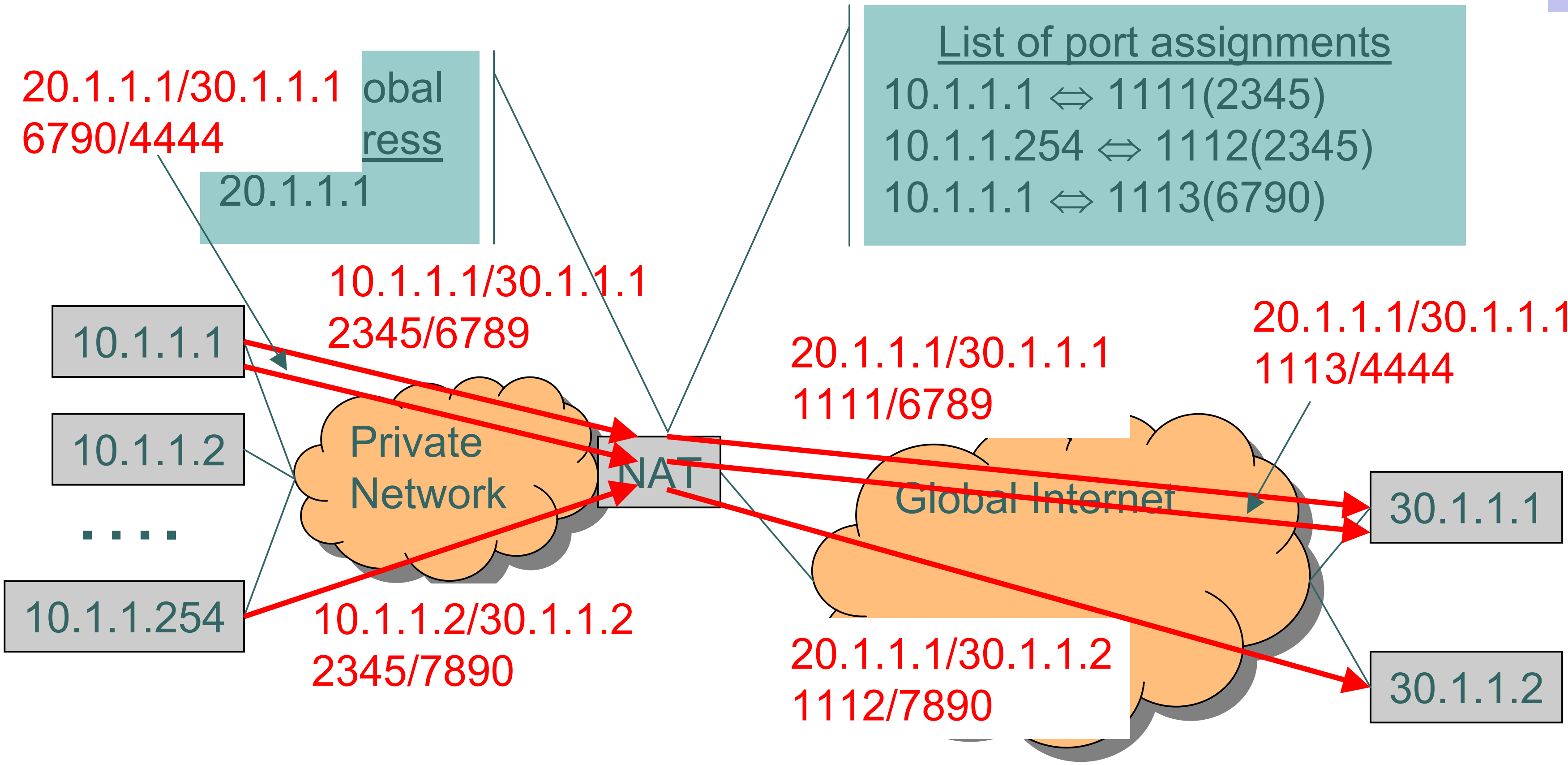
Current NAT design: Global address shared at one time

CS419



Current NAT design: Global address shared at one time

CS419



Problems with NAT

- Hard to make incoming connections
  - But will show you how in next lecture
  - This marketed as a feature of NAT!
- Some applications break
  - Those that carry IP address in upper layers
  - Less of a problem than it used to be
    - NAT boxes translate IP addresses in upper layers for common applications
    - Application designers now know not to put IP addresses in the upper layers

(Unexpected) advantages of NAT

- Isolates site from global addressing
  - Can change ISPs without renumbering
- Privacy
  - ISPs could otherwise charge you per host
  - Hard to tie IP address to user
  - Outside can’t deduce how many hosts you have
- Fun to irritate IETF end-to-end purists :)

Attempts to fix NAT (1/2)

CS419

- RSIP (Realm Specific IP)
  - IETF work
  - Host can request an address and address+port assignment from the NAT box
  - Didn’t go anywhere
- Microsoft UPnP (Universal Plug and Play)
  - Broad initiative to allow cross-vendor plug-and-play in local network environment
    - Auto-configure into net, advertise its capabilities
  - NAT aspect: Client can learn of address/port mappings from NAT box, add new port mappings
  - I don’t know if this is taking off or not

Attempts to fix NAT (2/2)

CS419

- midcom (middlebox communications)
  - IETF working group
  - Broad effort to deal with all kinds of (now opaque) middle boxes (NATs, firewalls, Intrusion Detection Systems (IDS), etc.)
  - Usual standards committee trashing about
- STUN (Simple Traversal of UDP through NAT)
  - Bad name…try searching for it with Google!
  - Simple method for host to learn what port it got assigned (transparent to NAT box)
  - Then application can use this knowledge as it sees fit

I like STUN

CS419

- RFC 3489
- I think it will succeed
  - Note that, of these options, STUN is the only one that doesn't require NAT box cooperation
  - This is a big win…
- I think it will be another nail in the coffin of IPv6
- I wish I had thought of it

Types of NAT behaviors

	Port assignment policy	Firewall policy for incoming packets (from dest address)
Full cone	Same global addr and port for every internal address and port (from a given internal host)	Accept all flows to assigned address and port from any dest address
Restricted cone		Accept if internal packet previously sent to dest address
Port-restricted cone		Accept if internal packet previously sent to dest address and port
Symmetric	Different addr/port for every flow	

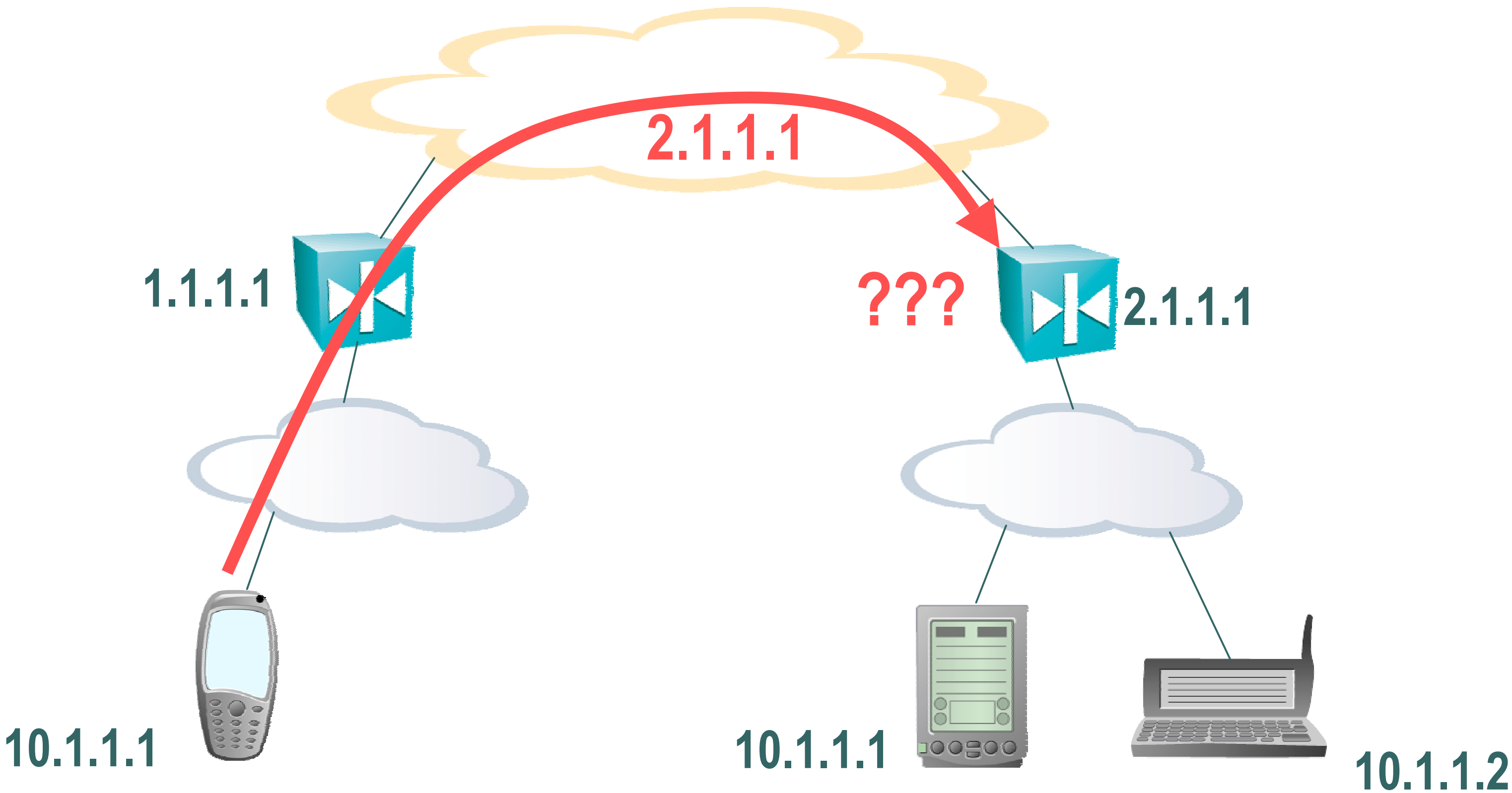
What STUN does

CS419

- Tells you if you are behind a NAT
- If so:
  - Tells you the assigned address(es) and port(s)
  - Tells you what type of NAT
- If not:
  - Can still tell you what kind of firewall you are behind
    - (UDP blocking, symmetric UDP)

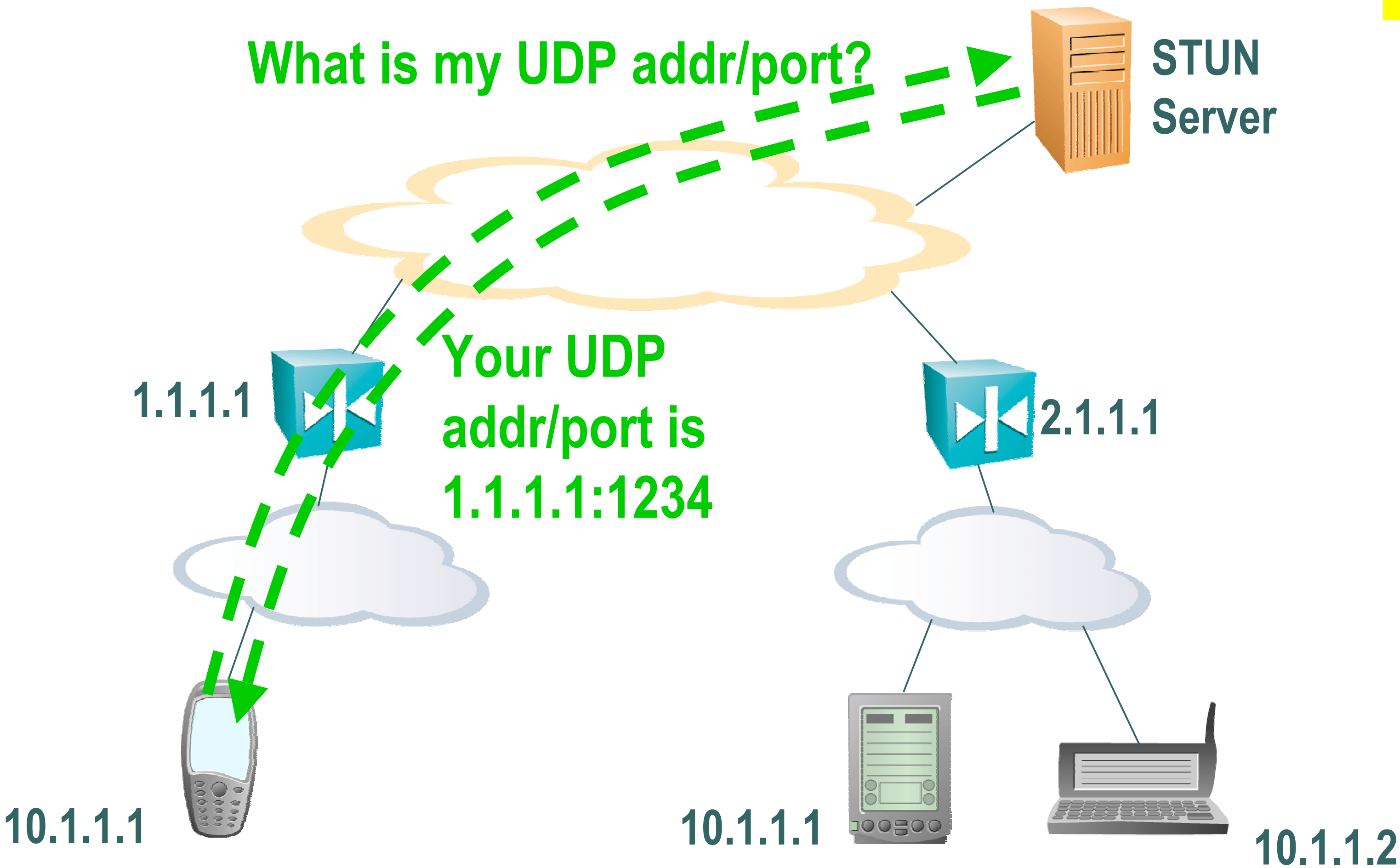
Packet can't come in until NAT box has mapping

CS419

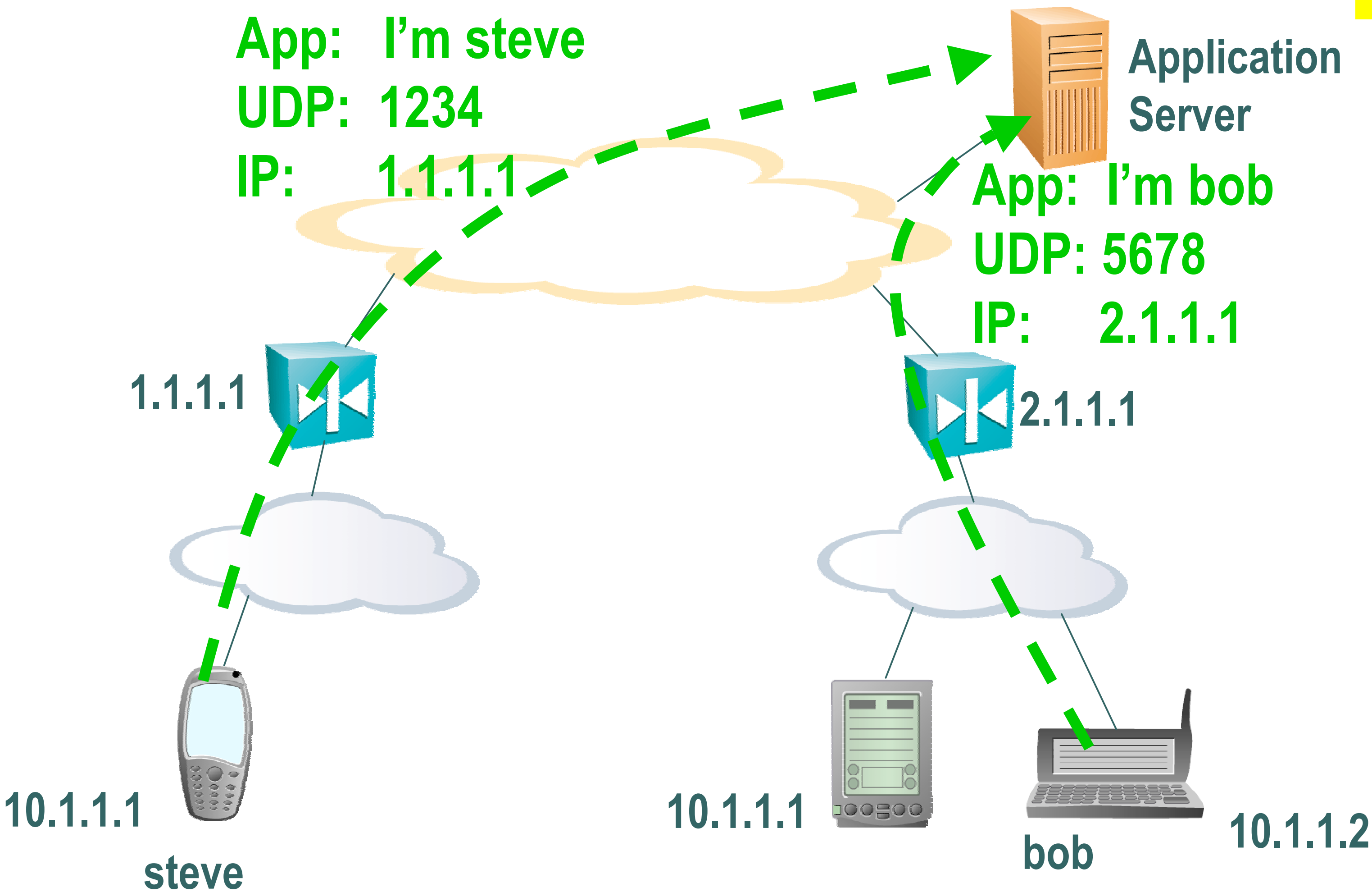


STUN server sees the global addr/port, and informs host

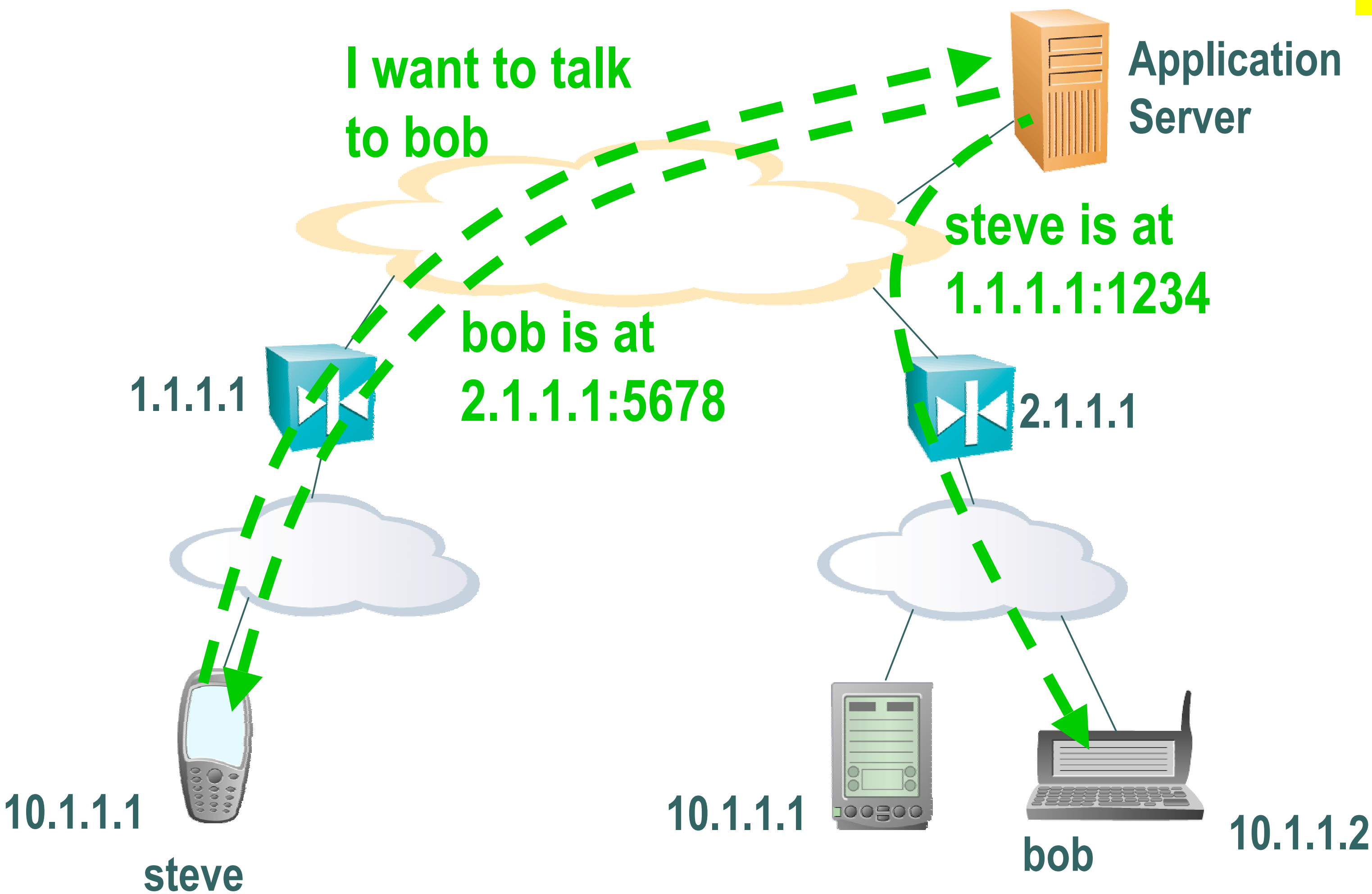
CS419



Steve and Bob register with globally addressed server

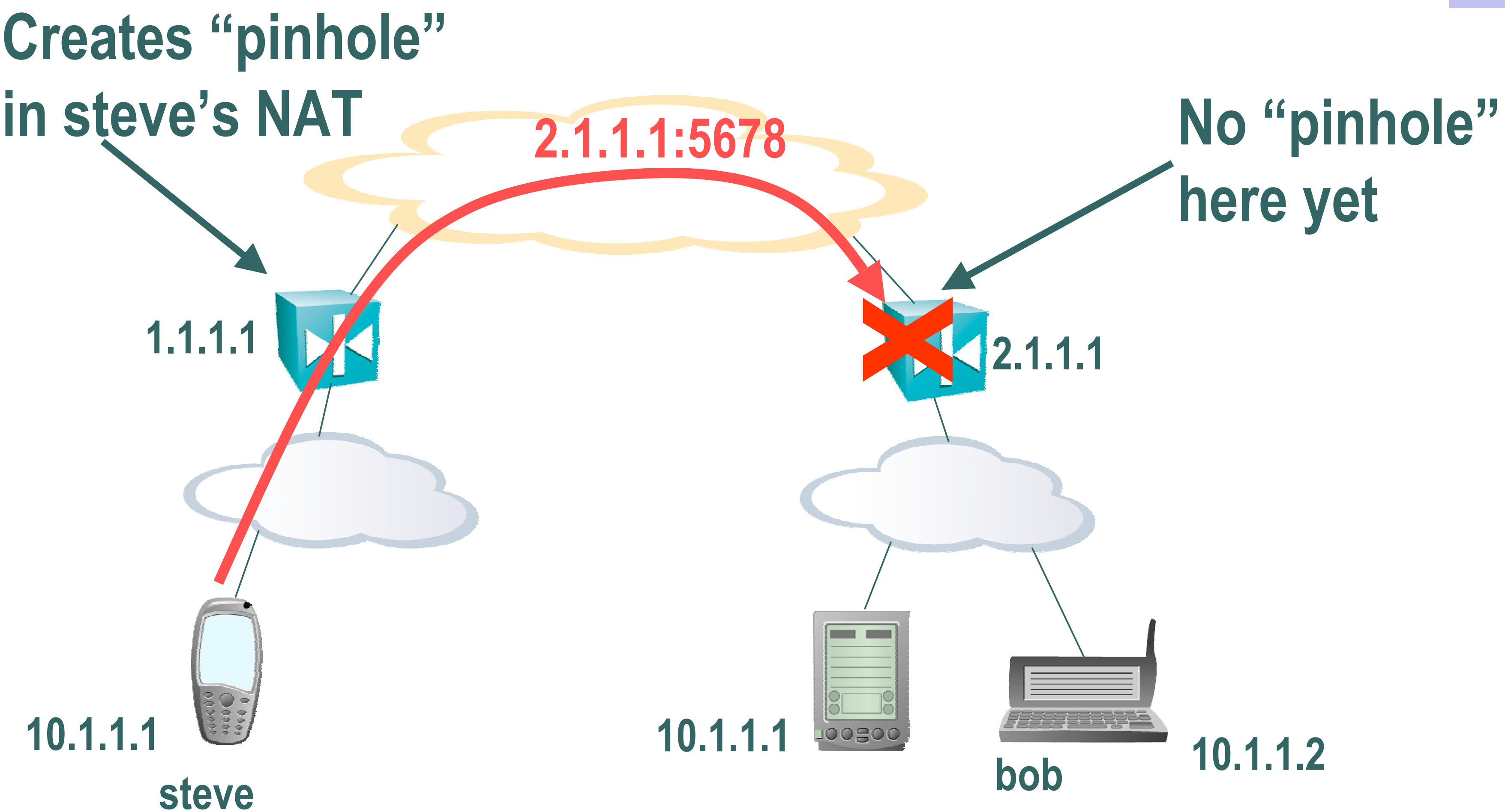


Server tells Steve and Bob each other's NAT mapping

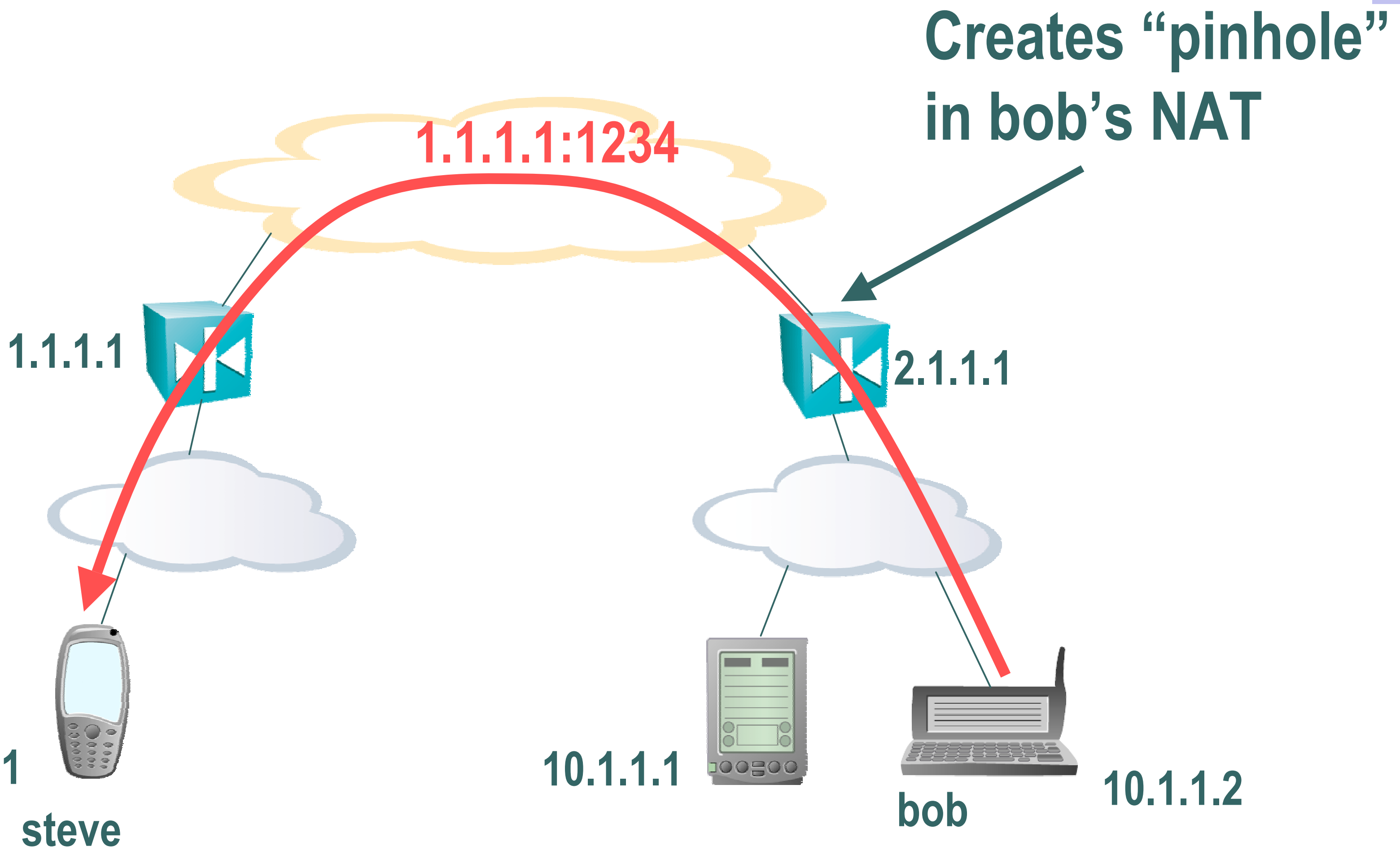


Steve sends "bubble packet" to create his mapping

CS419

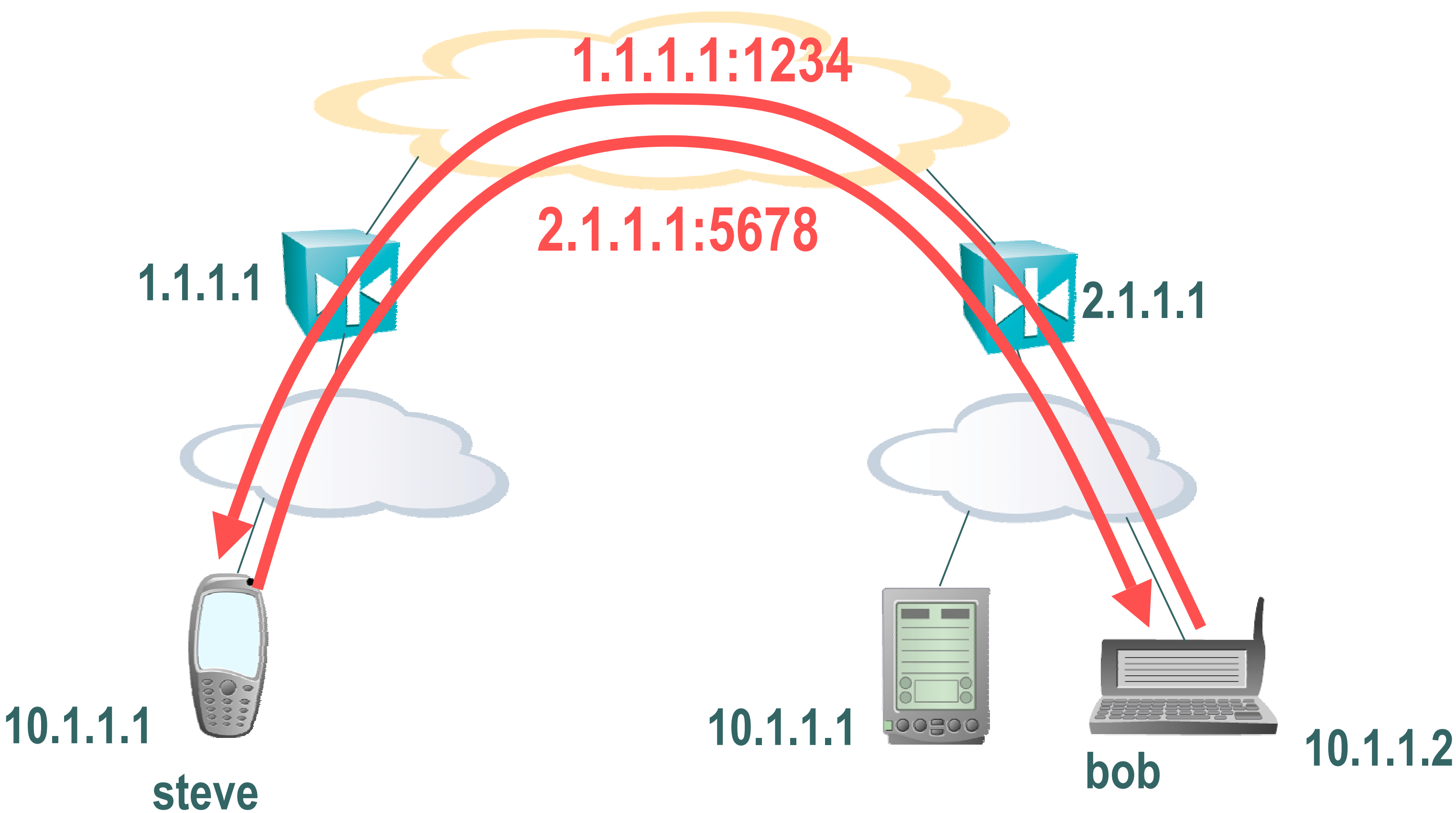


Bob does the same, but this packet gets through



Steve and Bob can talk!

CS419

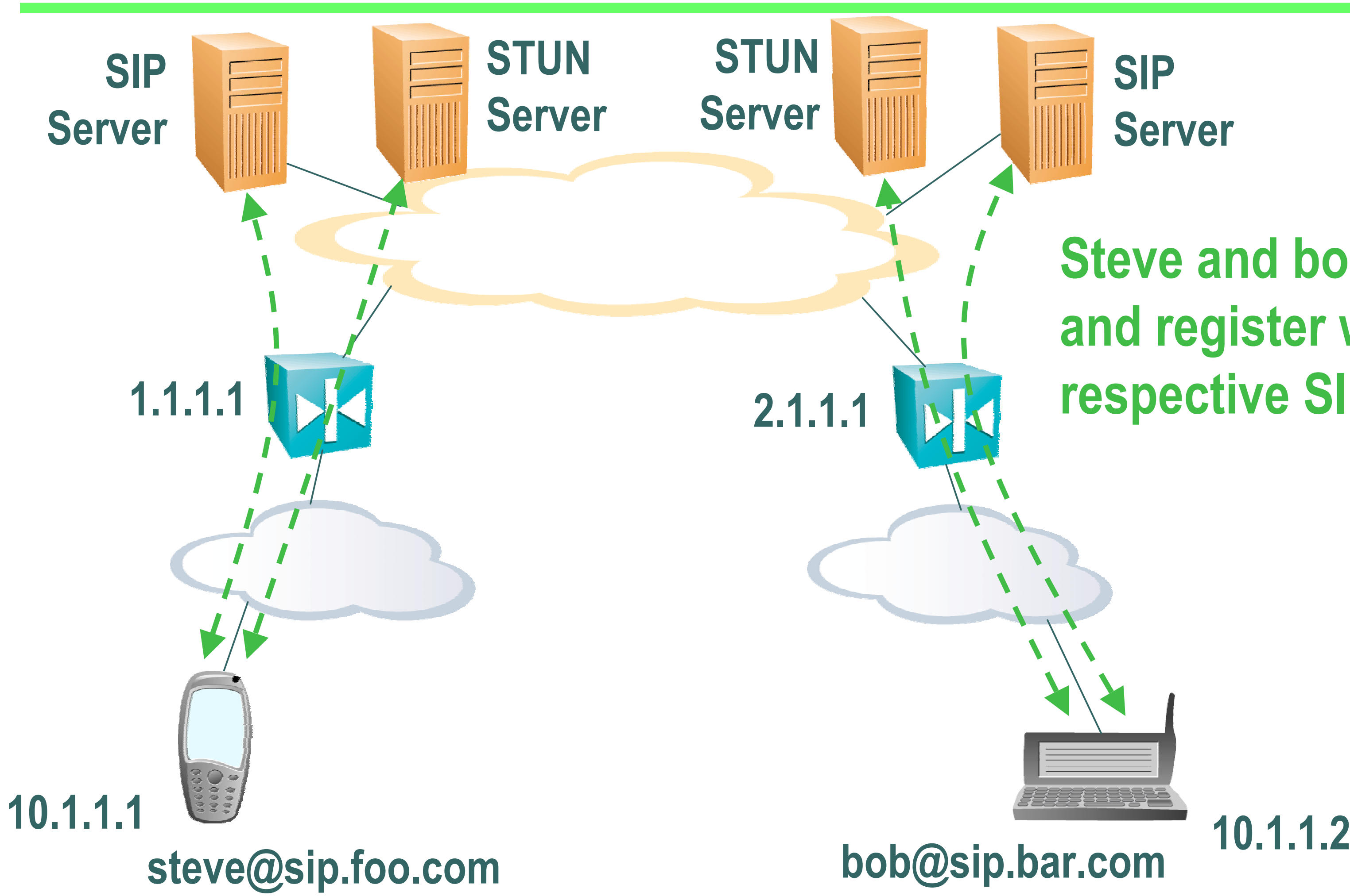


Limitations of this approach

- Doesn't work with some kinds of NATs
  - NAT must always assign same external port to a given internal port
- Doesn't work for TCP
  - Because TCP is *usually* asymmetric… expects a listener and a connecter
    - Windows OSs and some firewalls enforce this
  - We have a project to fix this problem
- Many corner cases (for instance, two hosts behind same NAT)

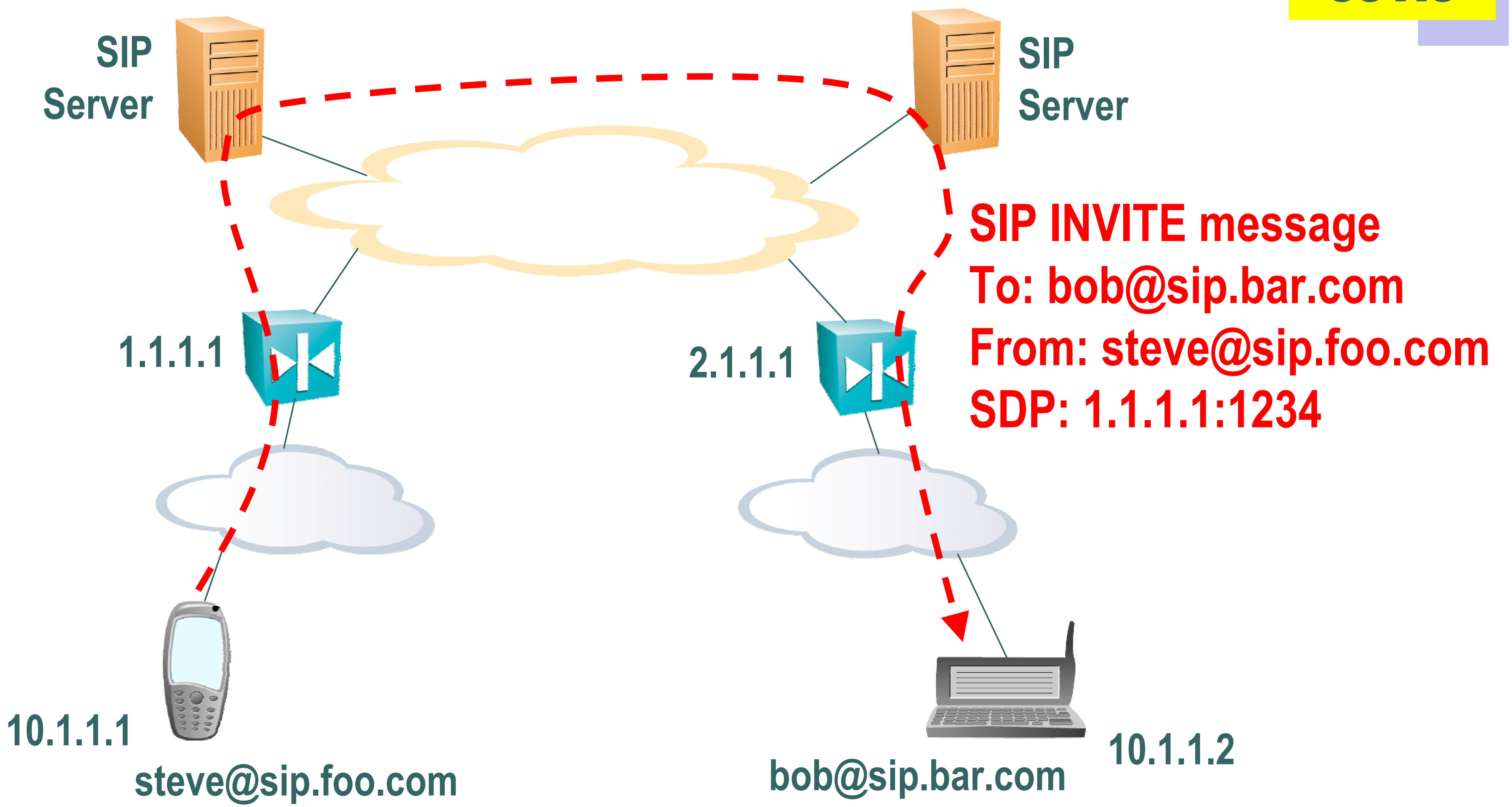
SIP with STUN (simplified)

CS419



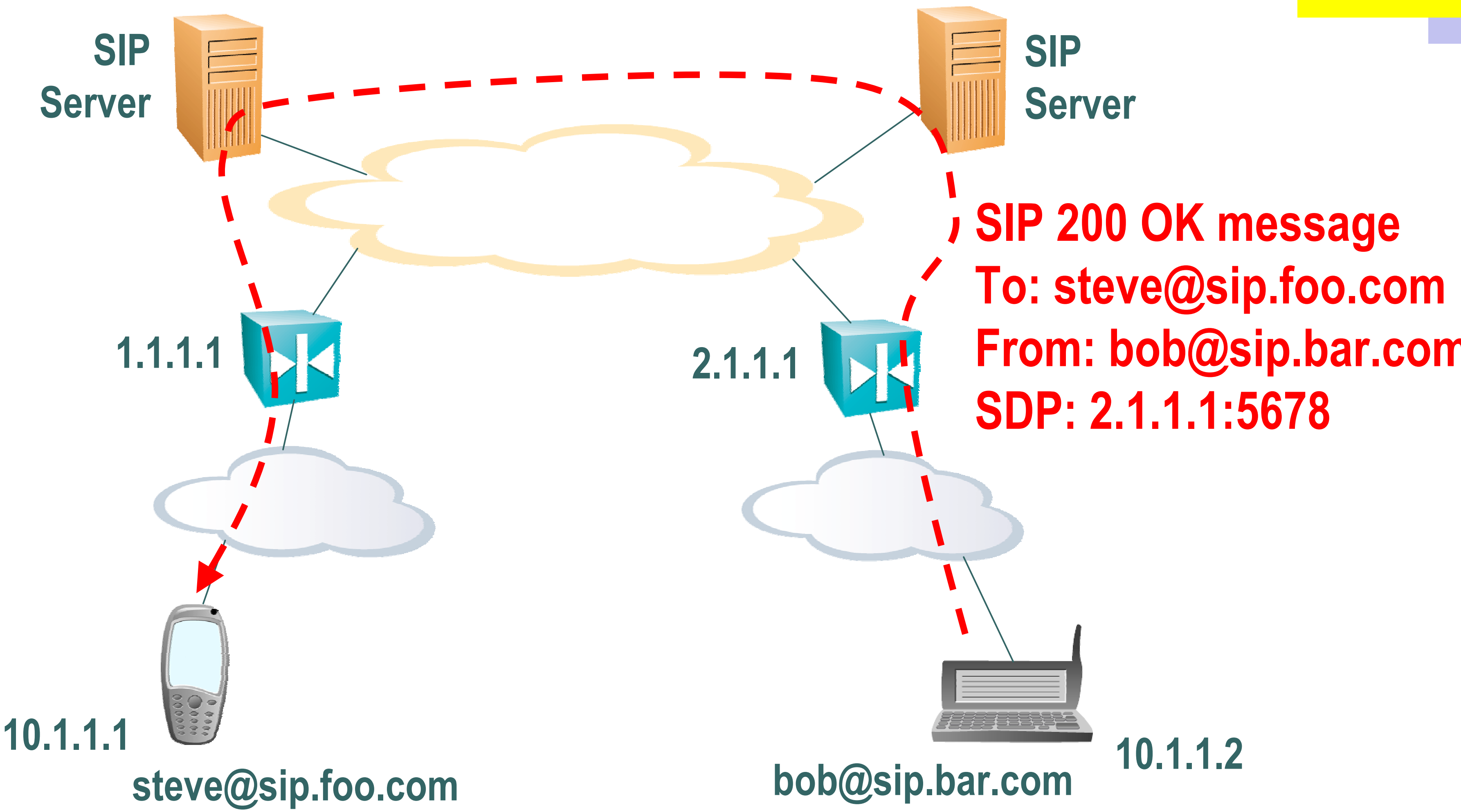
Steve and bob run STUN and register with respective SIP servers

SIP with STUN (simplified)



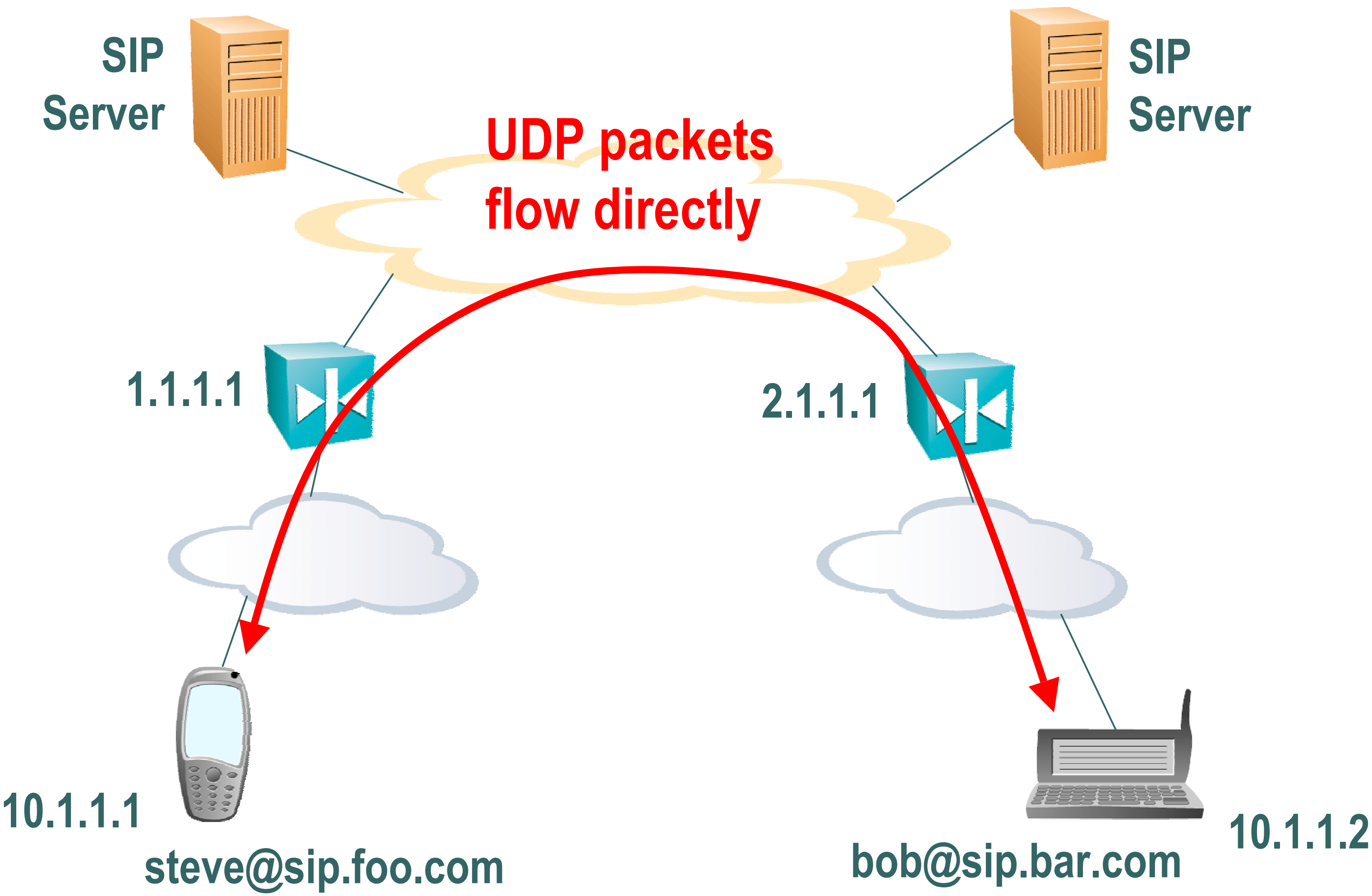
SIP with STUN (simplified)

CS419



SIP with STUN (simplified)

CS419



How to determine if NAT is restricted

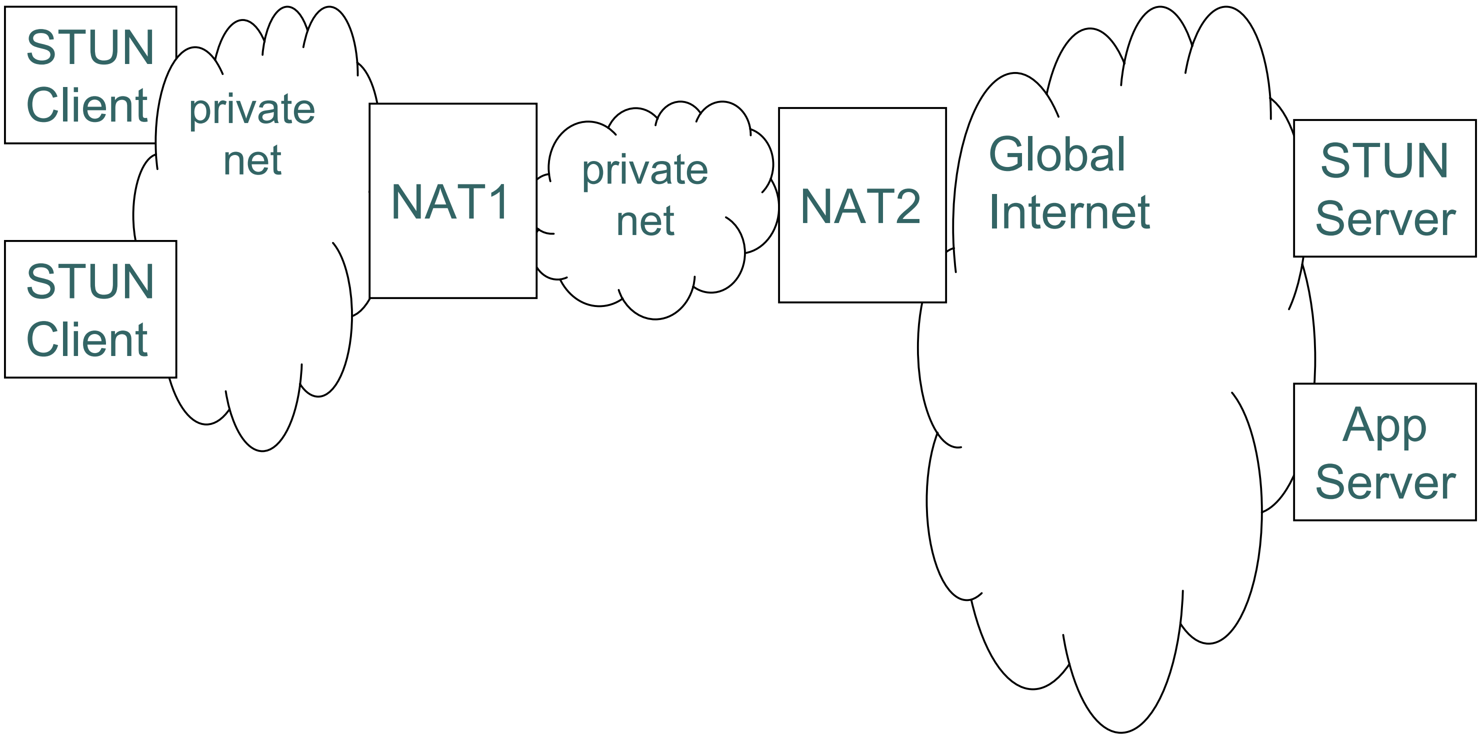
- STUN server can send packets from two addresses and two ports
  - Primary and secondary
  - pA and pP, sA and sP
- STUN client can ask the STUN server to use the secondary port or address and port.

Keeping NAT assignments alive

CS419

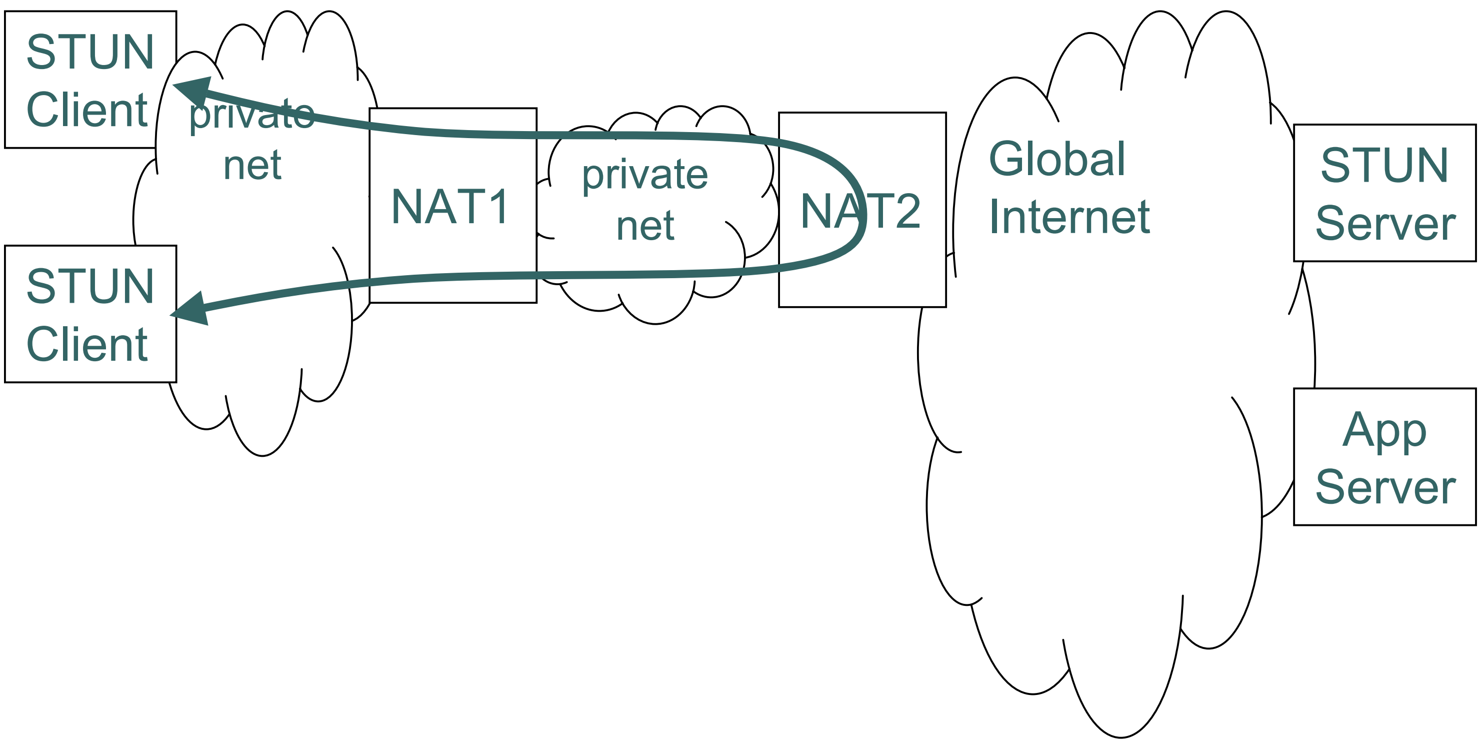
- NAT box will time-out port assignment after inactivity (if UDP)
  - At end of TCP connection if TCP
- App must periodically send packets to keep NAT state alive
  - Every minute or so?
- Note that client can try to learn NAT box time-out value
  - But this takes time, and is prone to failure

What about this????



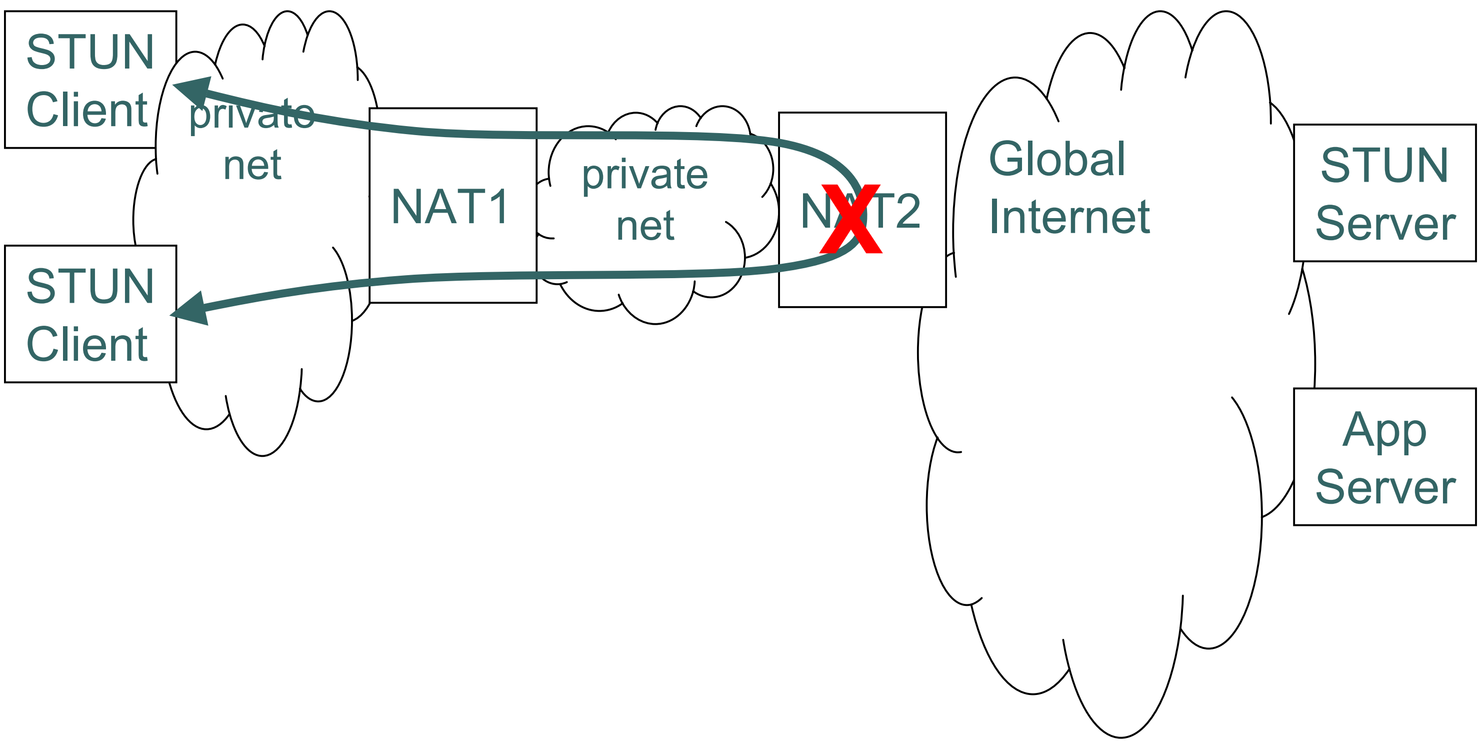
Don’t really want this…

CS419



And some NATs don't allow it!

CS419

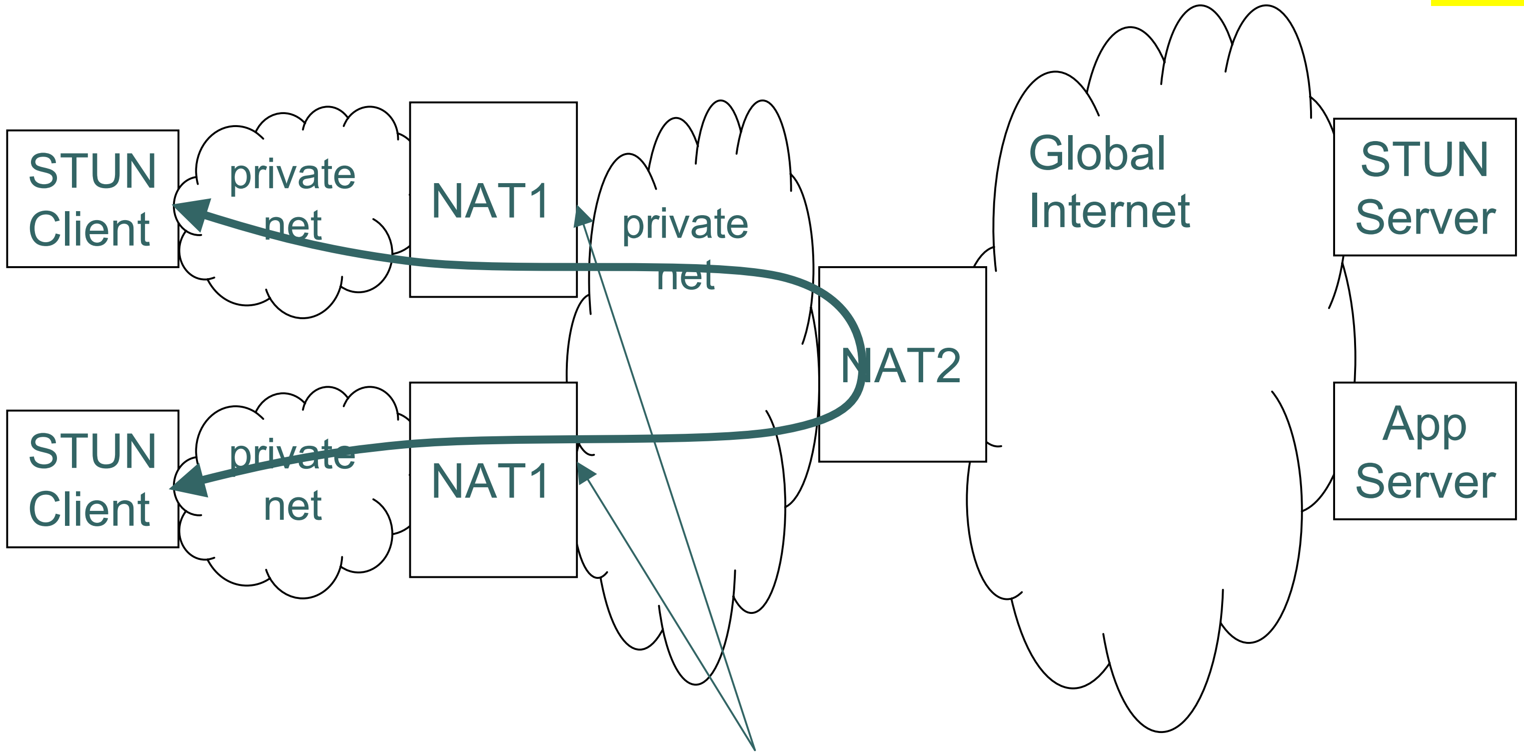


May use heuristics to decide if on same private network

- Peers have same global IP address
  - But this may not happen
- Peers have same domain name
  - Doesn't mean peers are in the same private network though
- Doesn't hurt (much) to try local address and global address

What about this????

CS419



This is the only choice. No way to learn these addresses.

Discovering STUN servers

- Two ways:
  - By address
  - By name
    - By SRV record (preferred)
    - By A record (if SRV doesn't work)

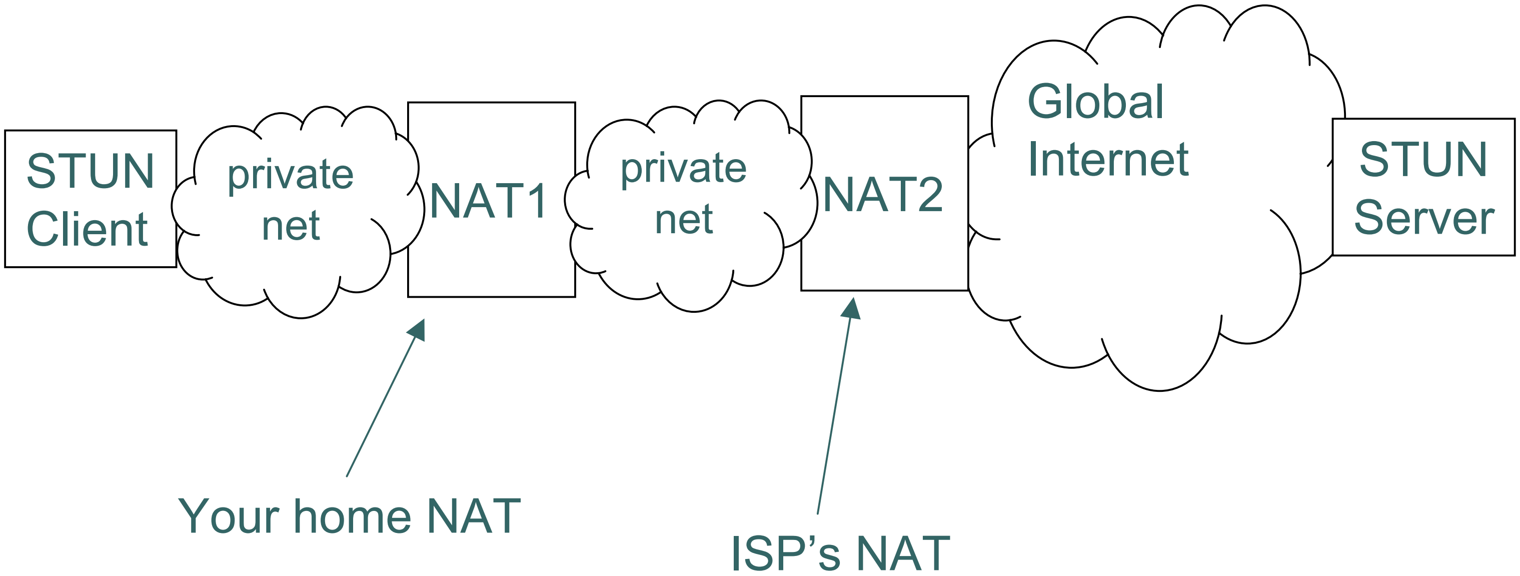
Stuff I didn't talk about

CS419

- Before the query/reply, there is a security phase over TCP using TLS
  - The STUN server securely gives you a temporary name and password
- Other details to overcome security problems

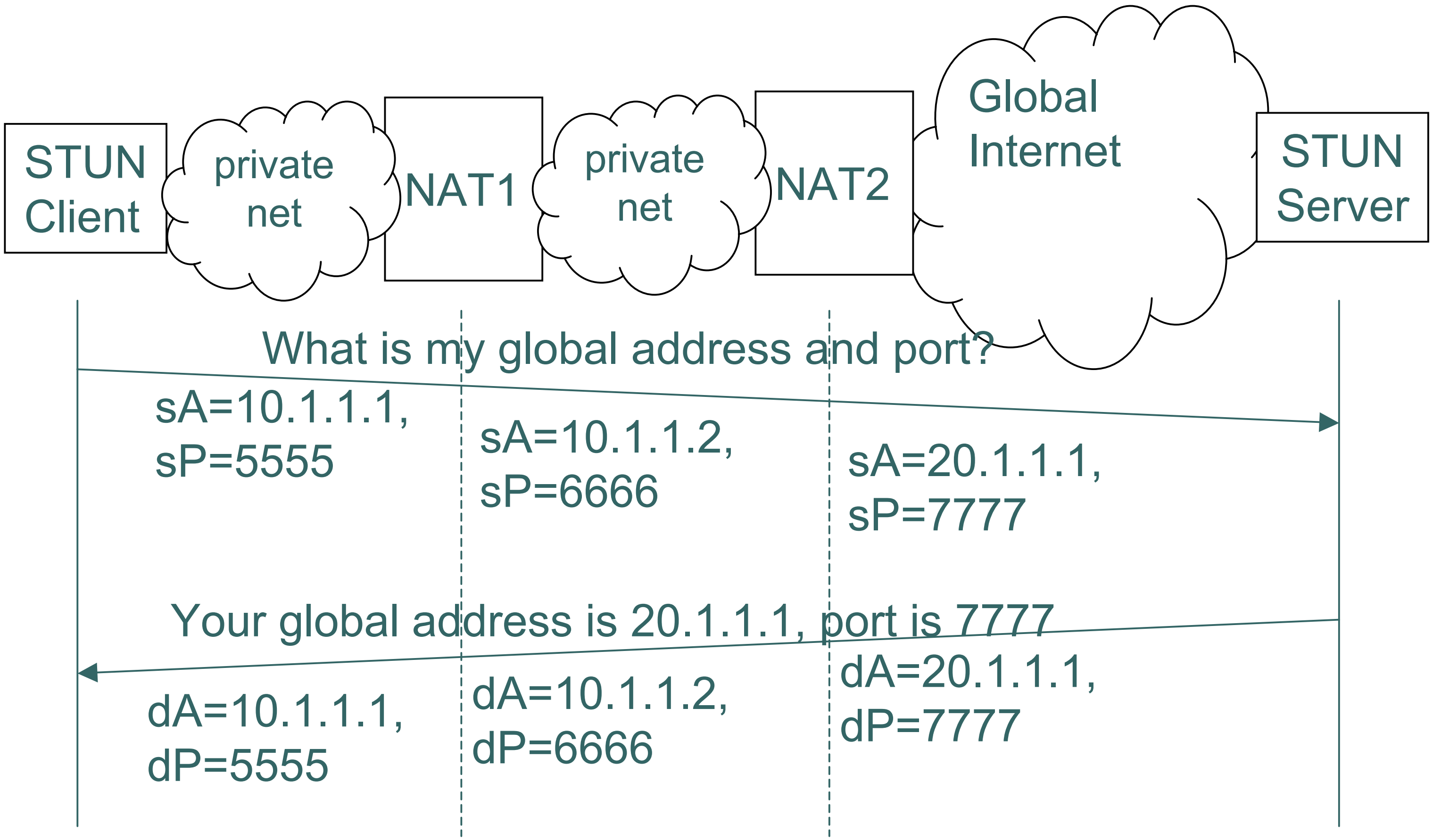
CS419

Typical STUN deployment



Basic operation: query/reply

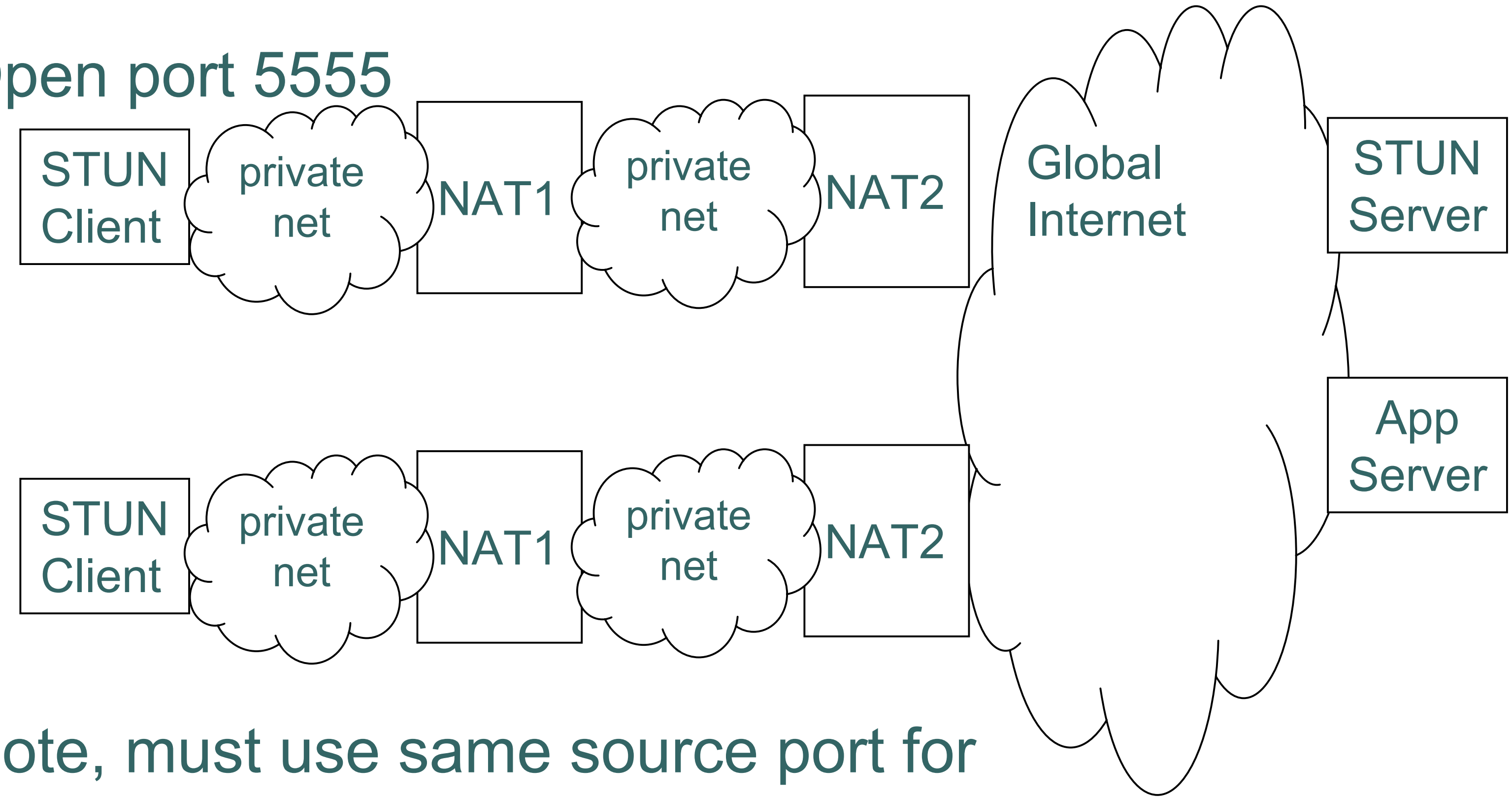
CS419



Use learned address/port to tell peer how to reach you

CS419

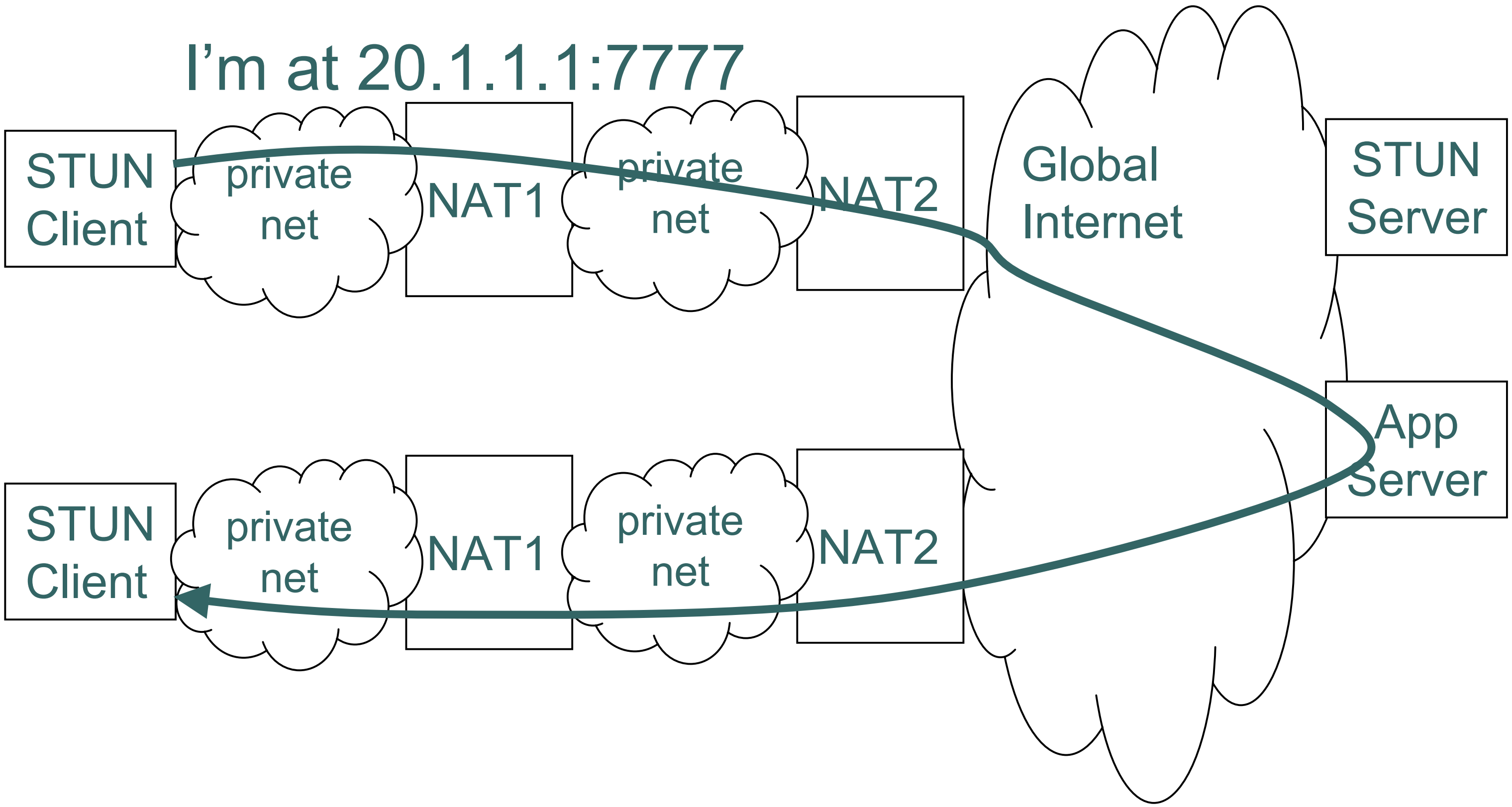
Open port 5555



(Note, must use same source port for app that was used with STUN to get same assignment from NAT box.)

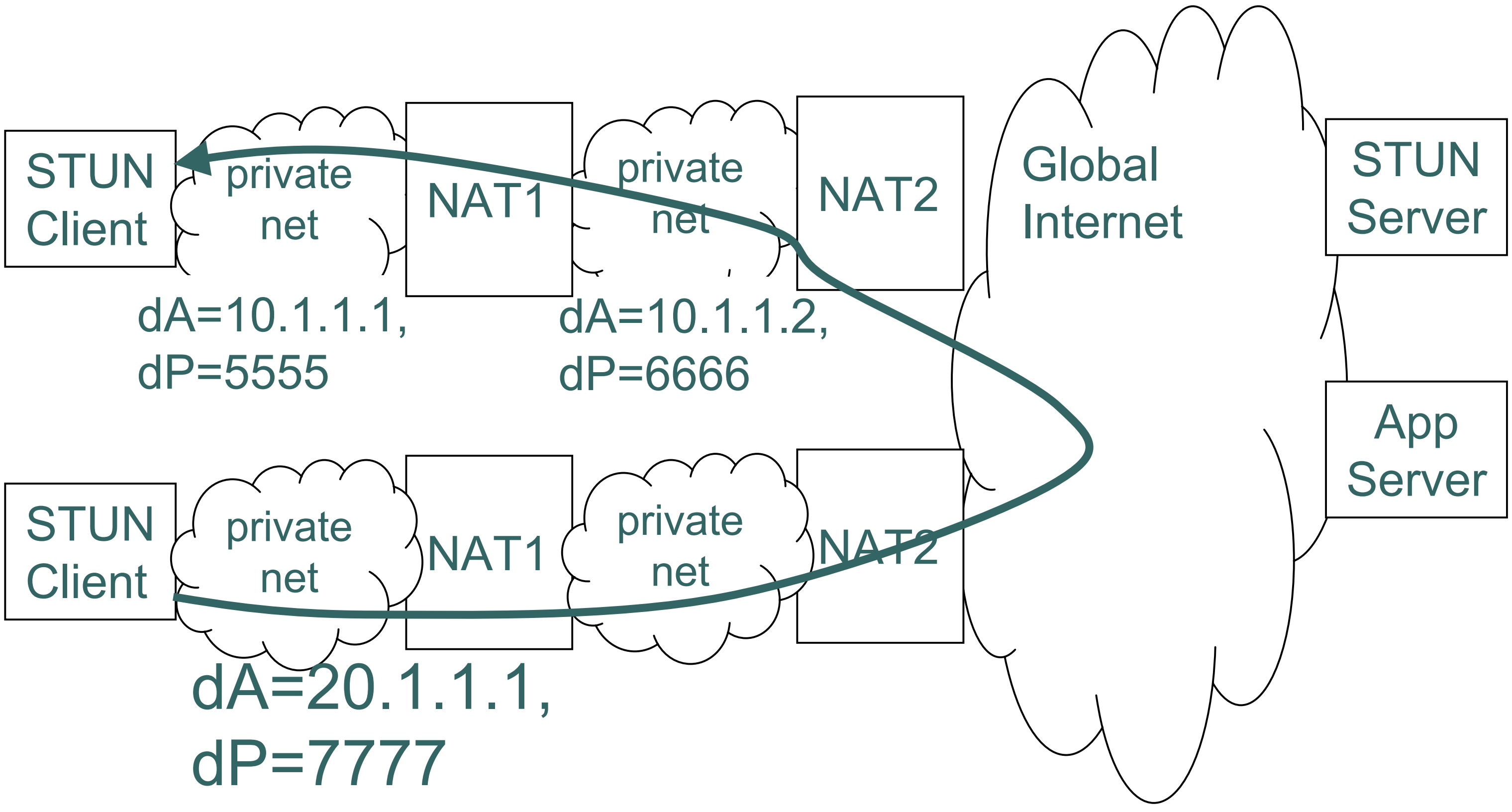
Use learned address/port to tell peer how to reach you

CS419



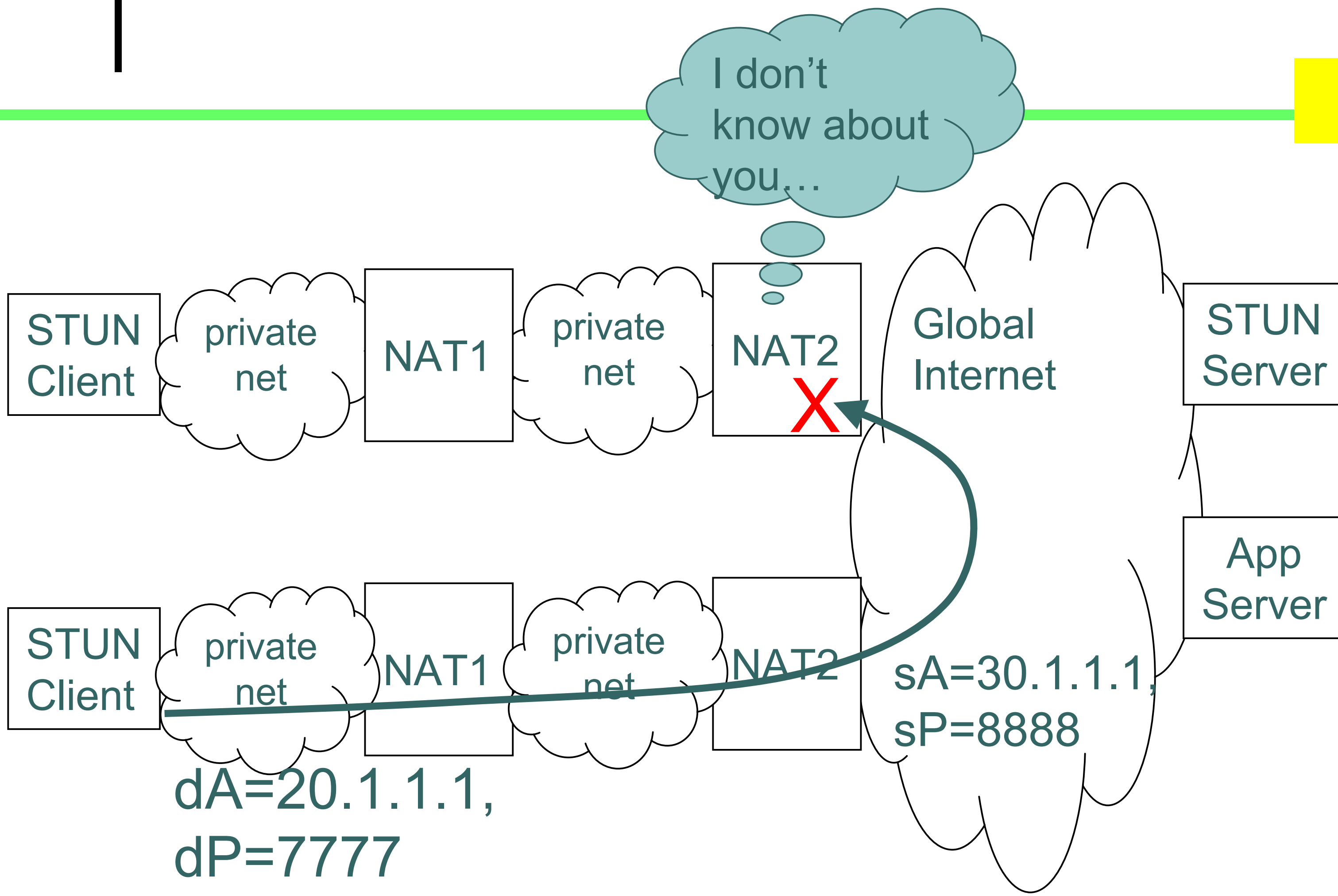
Voila, it works!

CS419

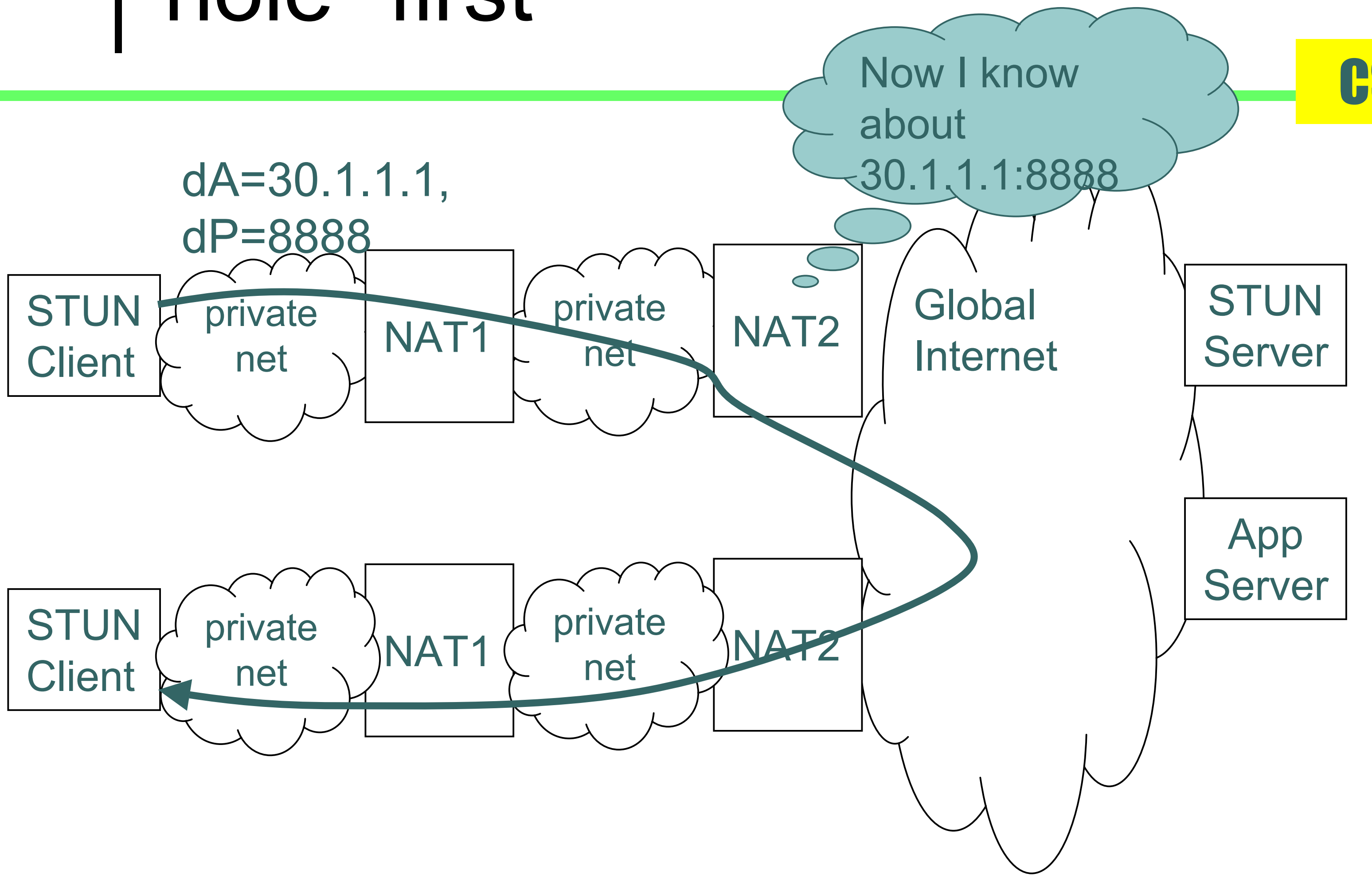


Unless NAT is restricted!

CS419



If restricted NAT, must “punch hole” first



How to determine if NAT is restricted