NEOHAPSIS

Neohapsis is now part of Cisco.

CISCO

Multipath TCP

Breaking Today's Networks

With Tomorrow's Protocols

Speakers - Who are we?

- Catherine (Kate) Pearce
  - Security Consultant / Pentester
  - New Zealand transplant.
  - Loves her wine the way she likes her RFCs (Dry)

- Patrick Thomas
  - Senior Security Consultant / Pentester
  - Application Security focus

MPTCP changes

fundamental assumptions

about

how TCP works*

Use it to break things today

Adapt to it for tomorrow

*Well… kinda

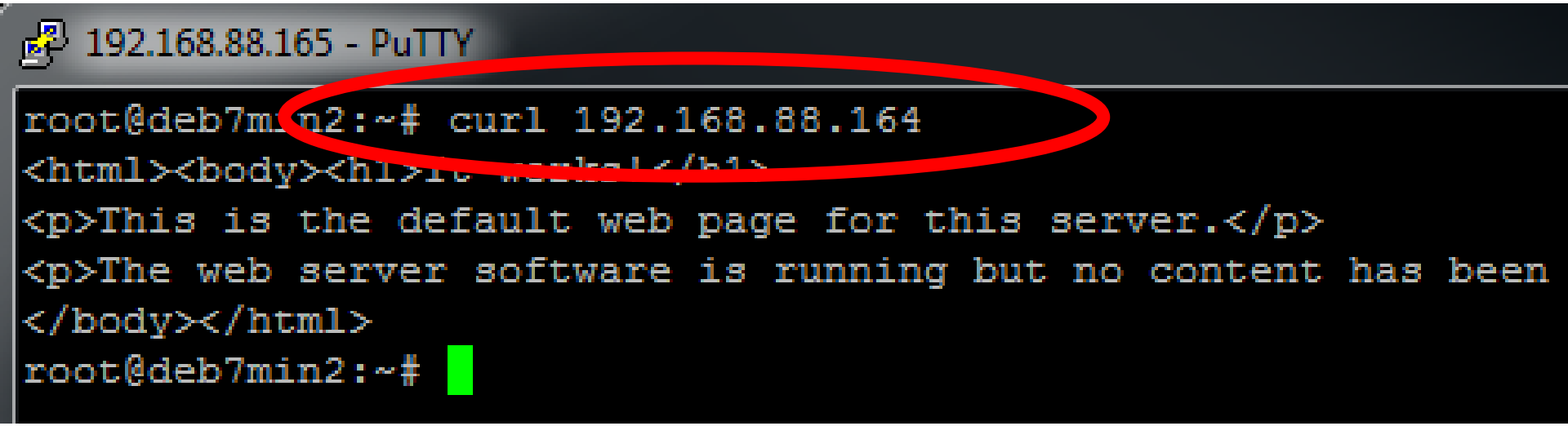
Not Layer 4?

Totally the same.

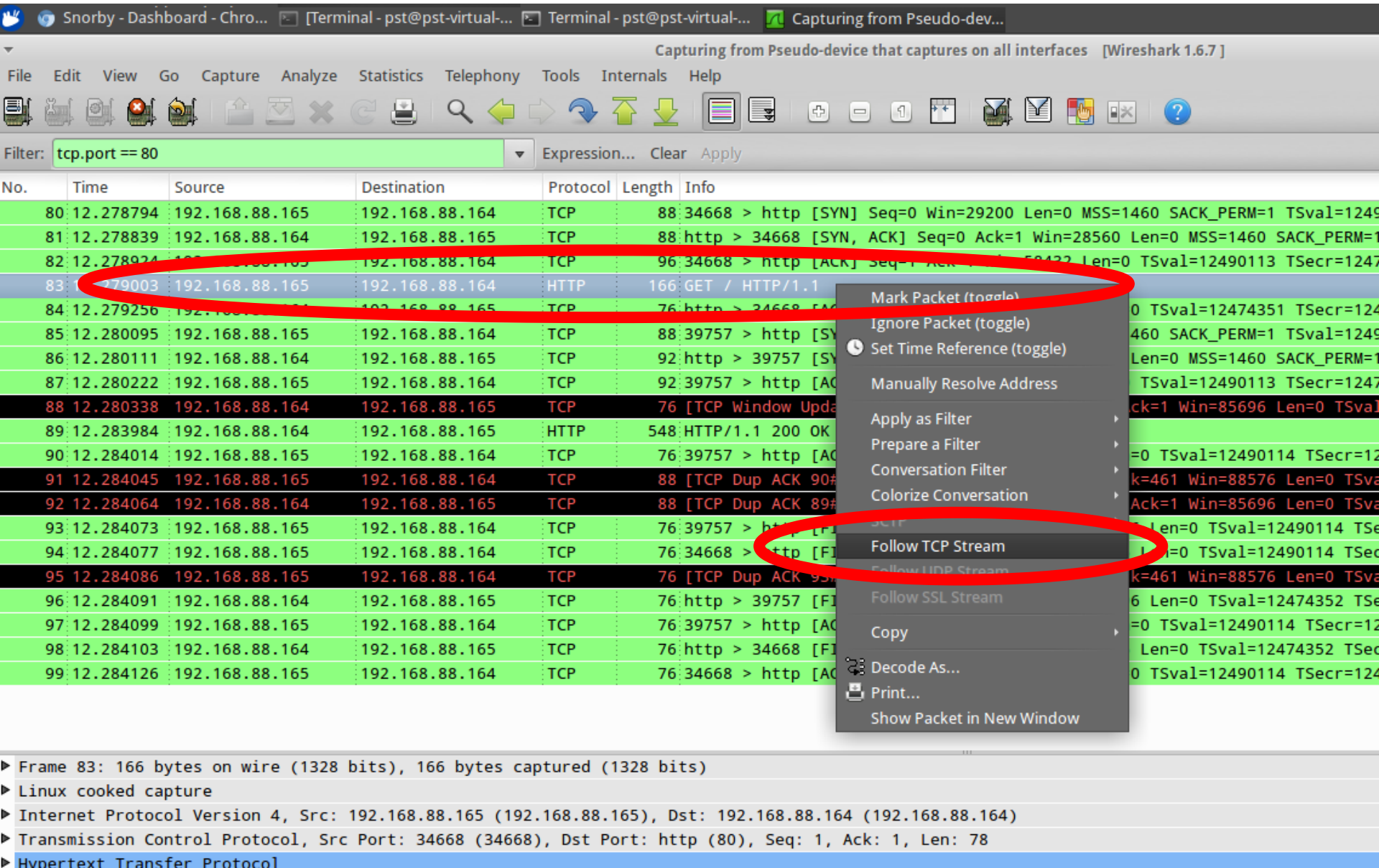
Layer 4?

Buckle Up.

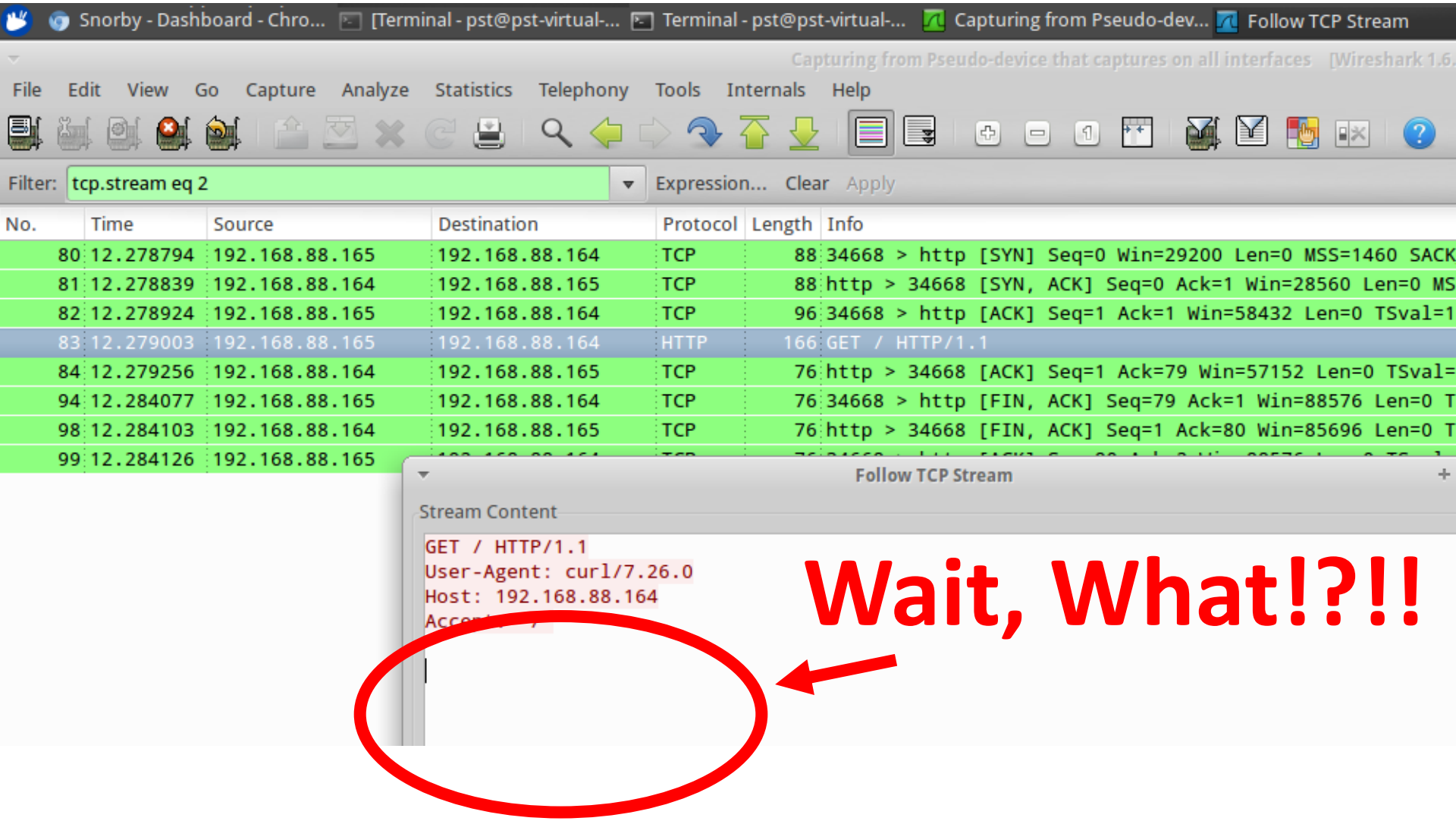
2 Simple Examples: #1



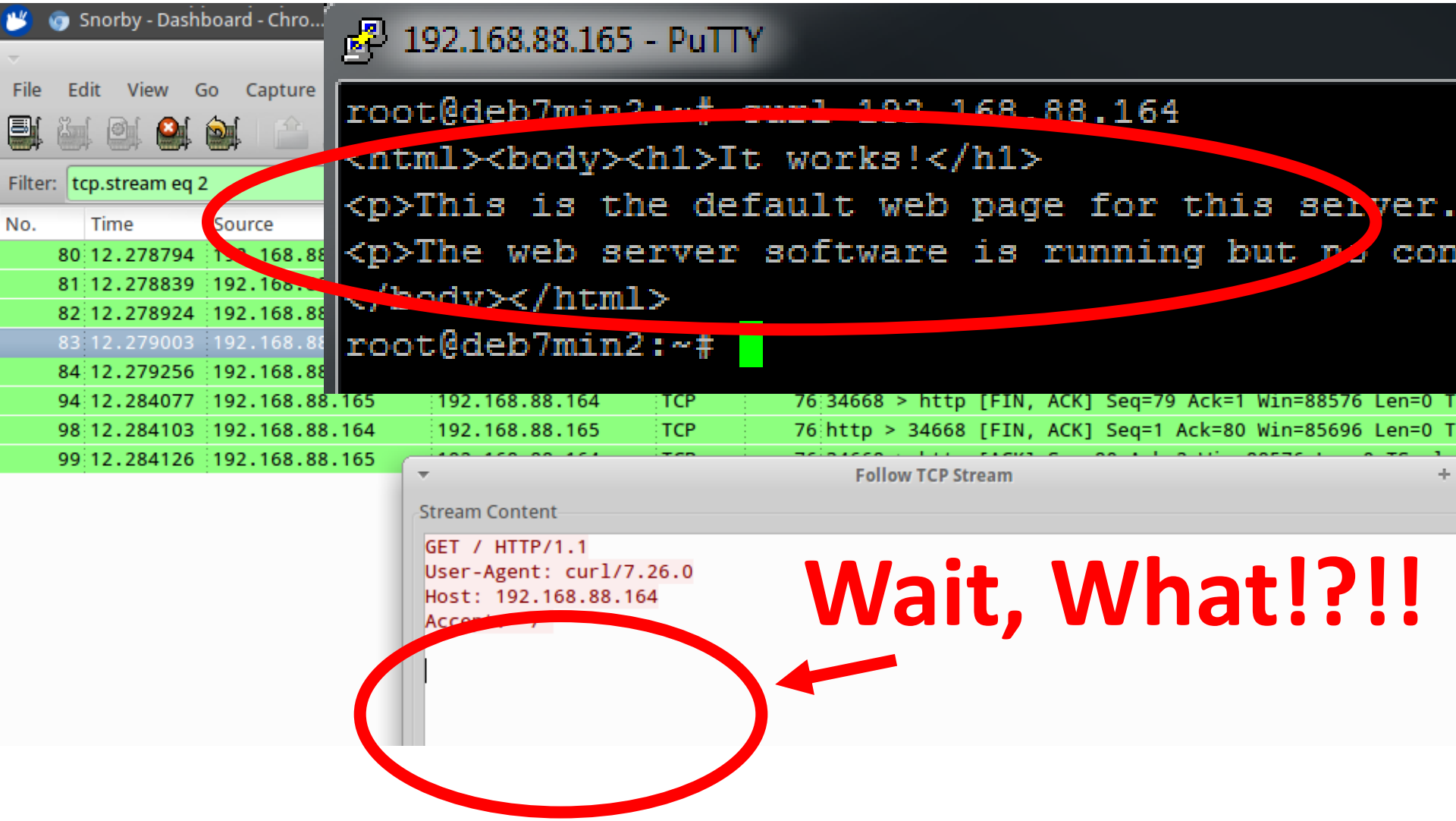
2 Simple Examples: #1



2 Simple Examples: #1



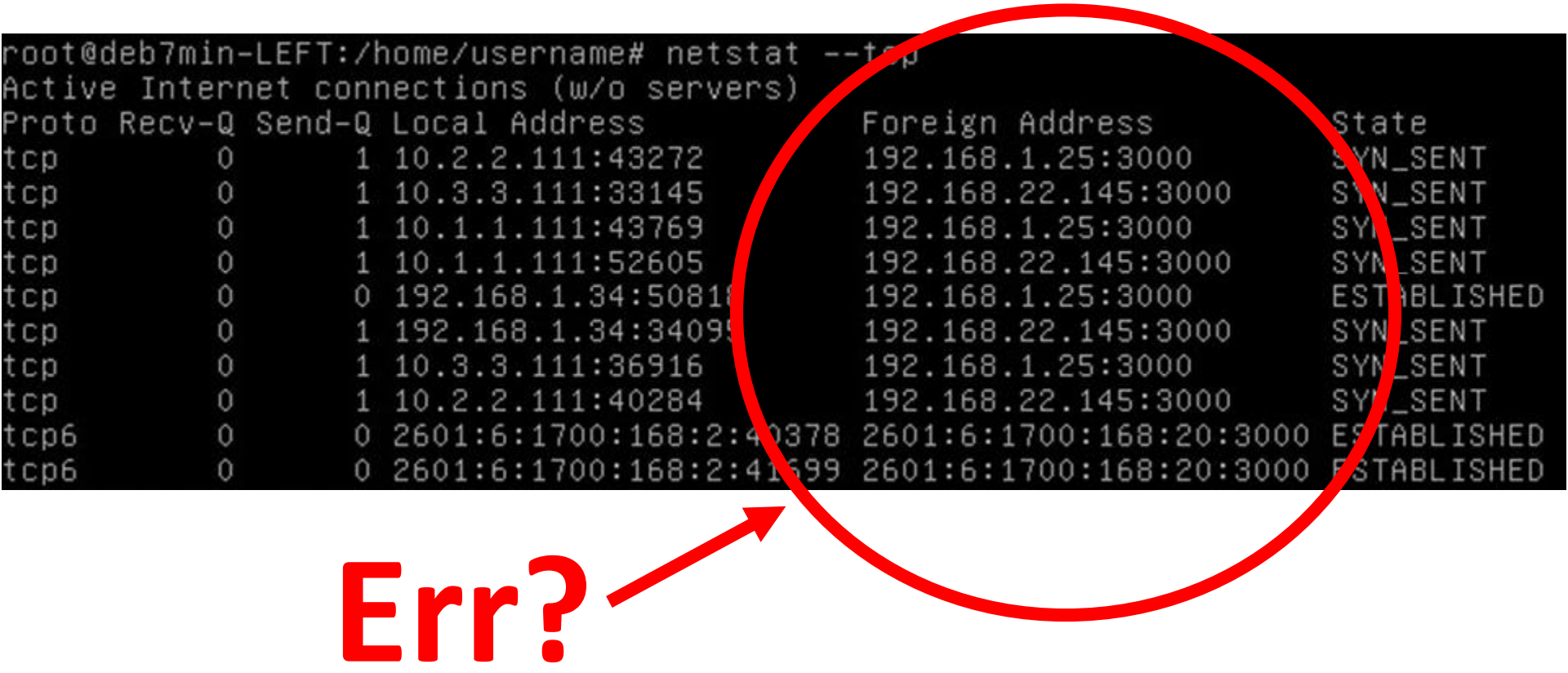
2 Simple Examples: #1



2 Simple Examples: #2

```
# nc 192.168.1.25 3000
```

2 Simple Examples: #2



Err?

Sense

This makes none

Why did we see that?

→ *Let's talk about MPTCP*

…but first, why change TCP?

Current TCP is rather limited

Doesn’t support use cases for:

- High Availability
- Link Aggregation
- Multihoming
- Mesh networking

Multipath TCP

Multipath TCP is an extension to TCP that adds the above functionality

AND: it works over existing infrastructure

- (it ***IS*** TCP… just more so)

BUT: nothing much else understands it – including security tools

TO BE CLEAR:

MPTCP is more culture shock than security vulnerability

We like MPTCP

We want MPTCP to succeed

Network security isn’t ready

Background

Technical Introduction

Key Security Effects

Perimeter Security

Network Management

MPTCP Future

What got us thinking about this?

- Kate saw this…

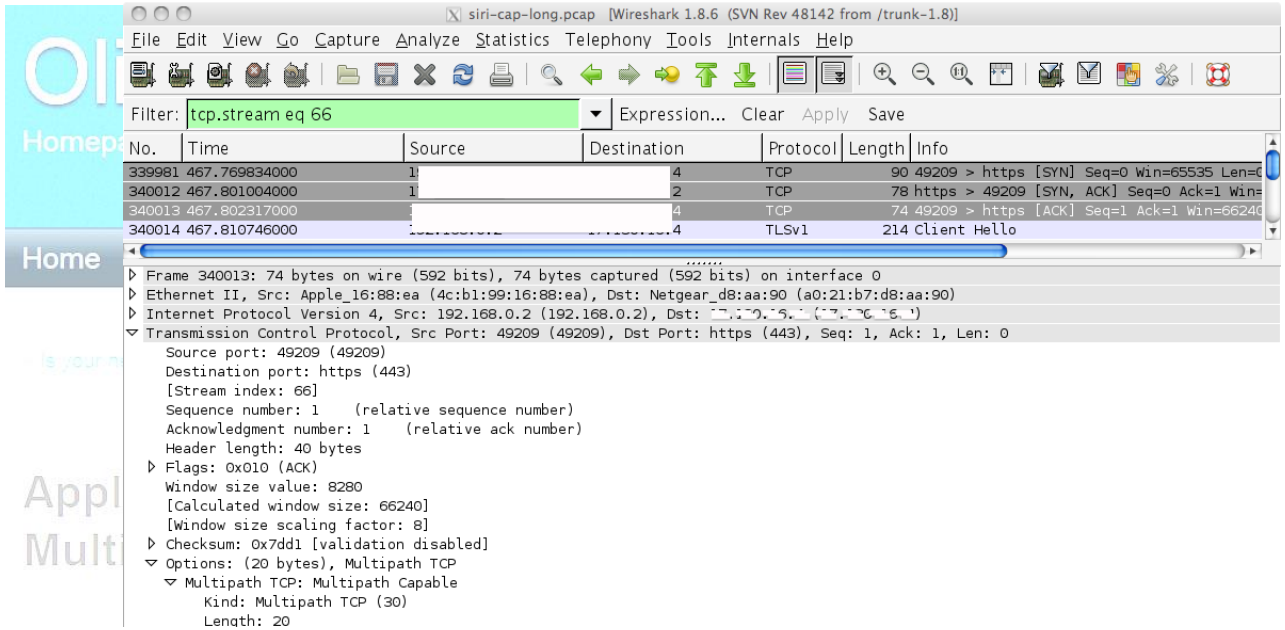
What got us thinking about this?

- Which led to this…

What got us thinking about this?

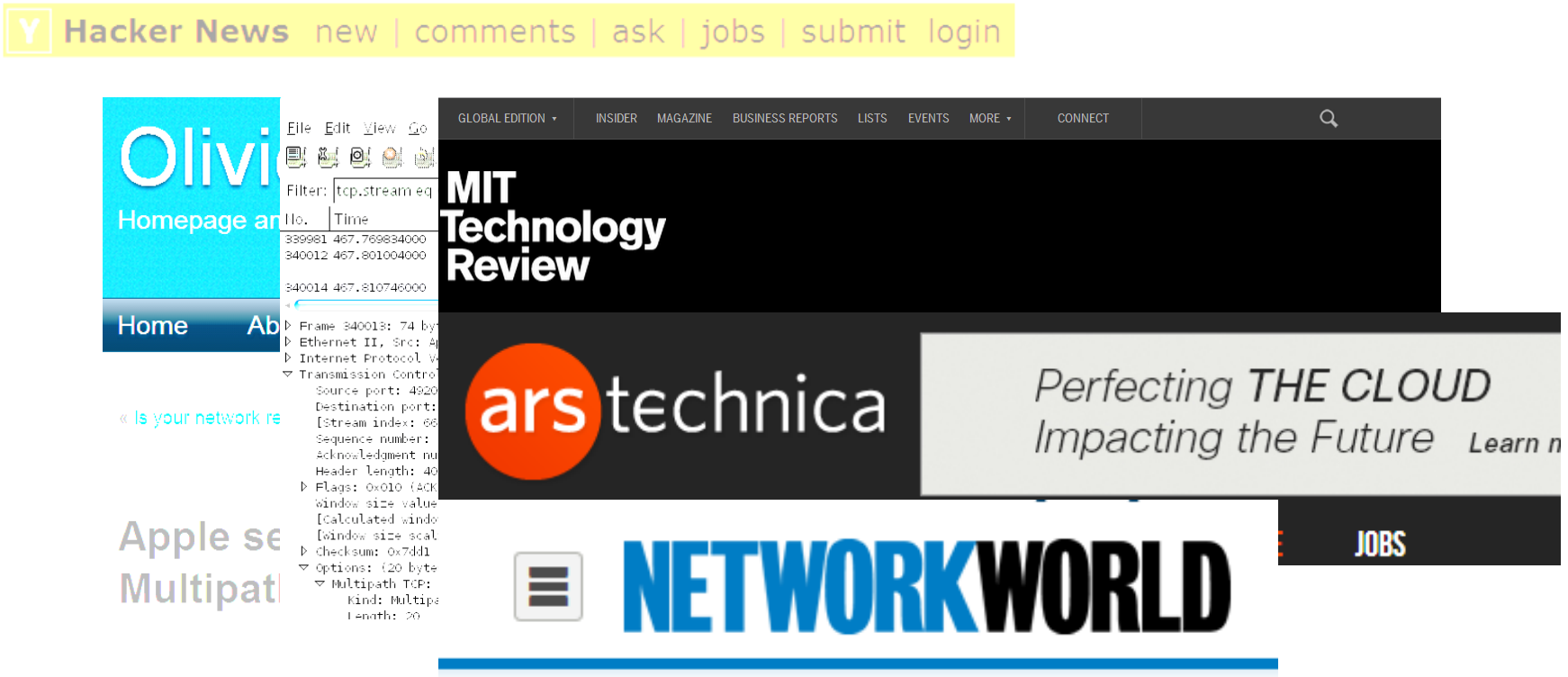
- Which contained this…

Hacker News new | comments | ask | jobs | submit login



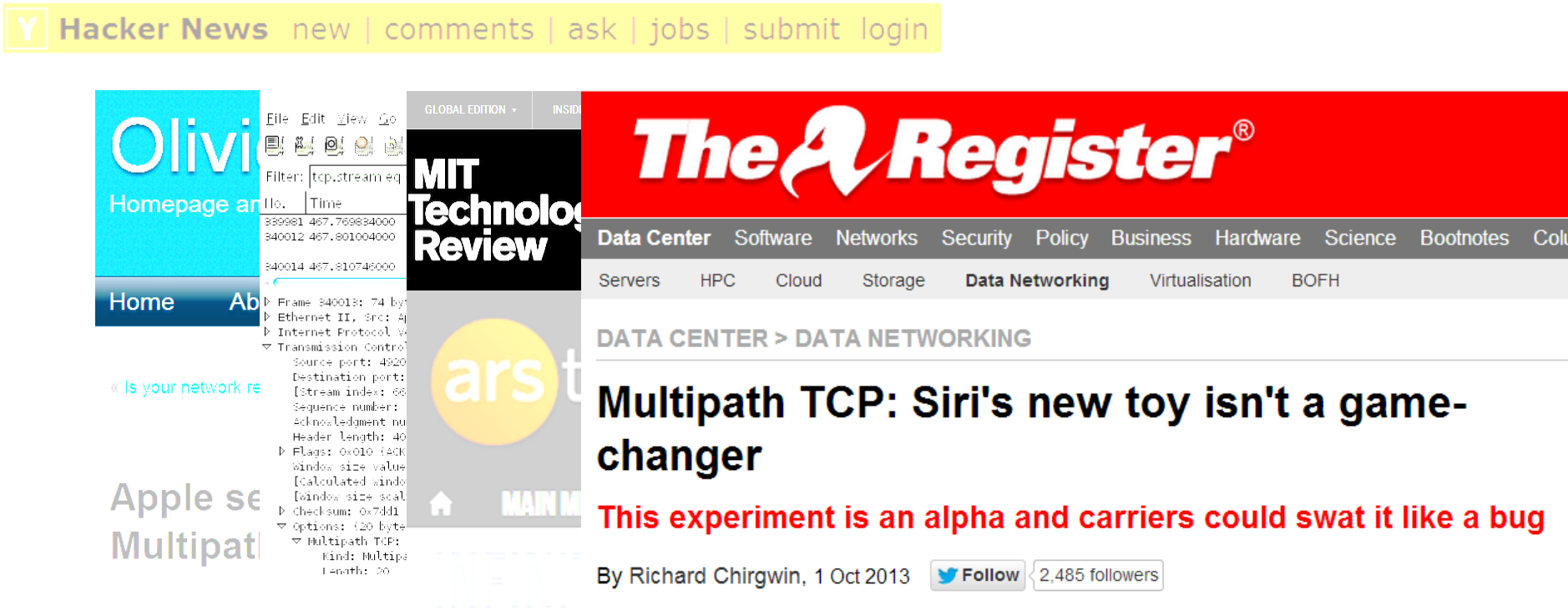
What got us thinking about this?

- Then other media outlets started covering it…



What got us thinking about this?

- Then other media outlets started covering it…not always positively



What got us thinking about this?

- And then…

SILENCE

- BUT, the rate of progress was unprecedented for a major change to TCP

Was anyone thinking about security?

- The security of MPTCP itself

- What changes like this *could mean* for network security

… not so much

That's what this session is about

- What does multipath TCP mean for security *today*?
- What could it (or similar tech) mean to network security *a decade from now*?
- With a couple of PoCs and tools…

Background

Technical Introduction

Key Security Effects

Perimeter Security

Network Management

MPTCP Future

Motivations and Advantages

- TCP implements connections between IP:PORT & IP:PORT
- NOT between endpoint A and endpoint B
- In the past this was a distinction without a difference, but not any more

Riding on top of TCP

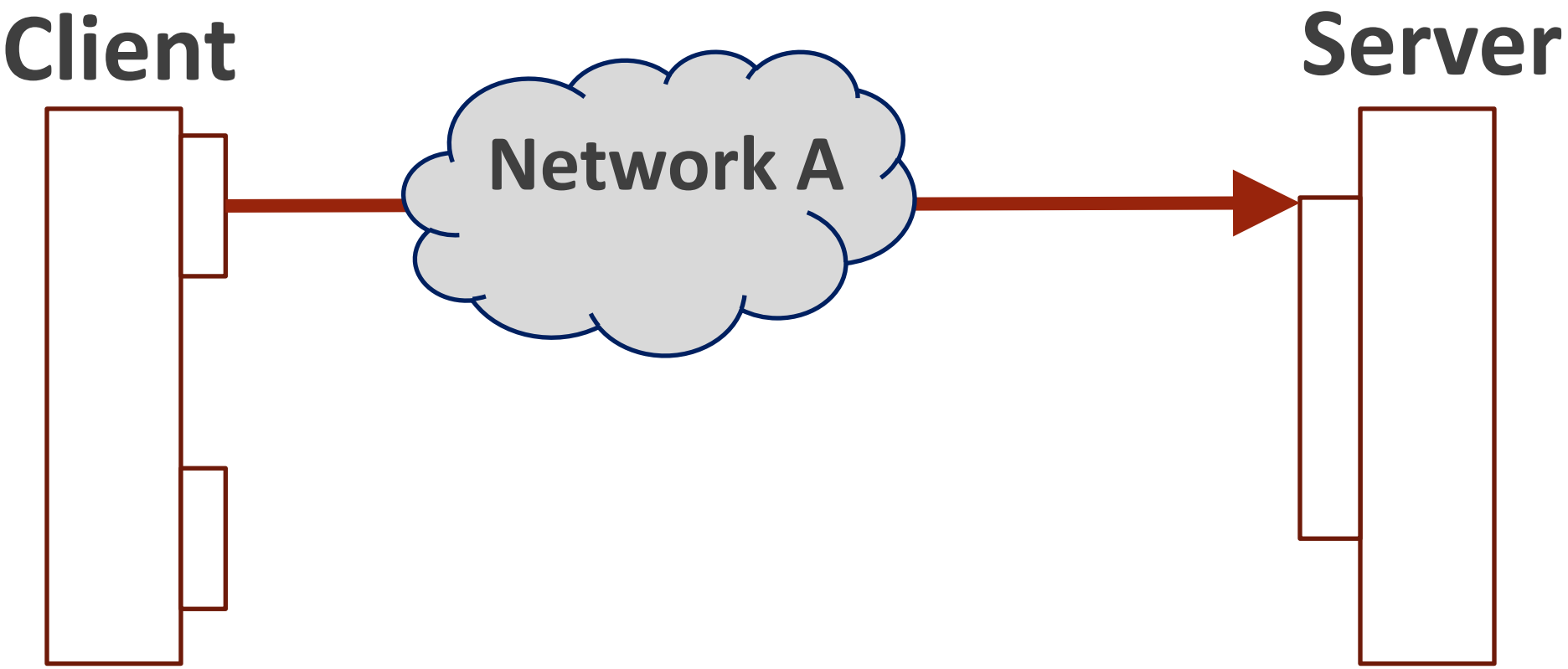
- **An MPTCP Connection** is defined by a connection ID

- It is composed of multiple *streams, where each stream is a regular* TCP connection (with an option strapped on)

MPTCP Characteristics

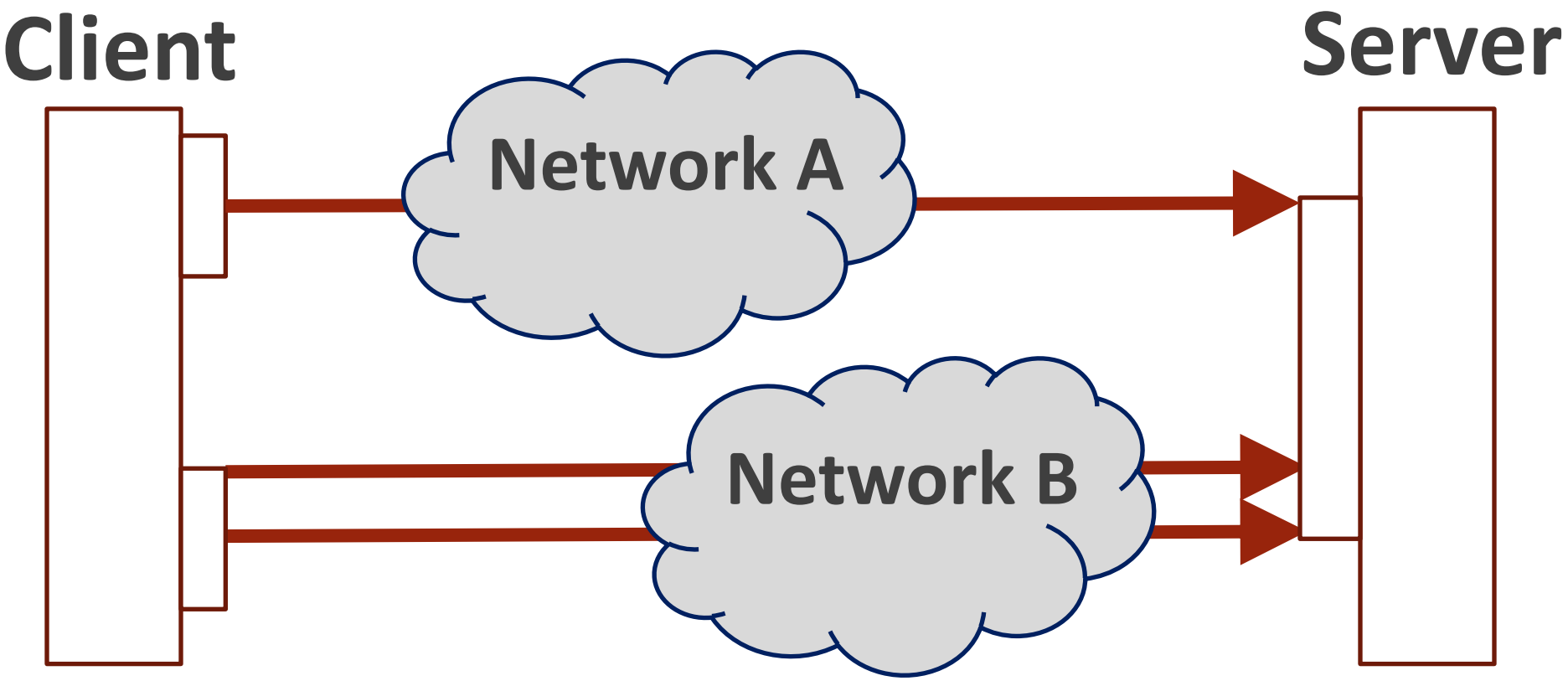
- Backwards compatibility
- Performance >= now
- Security >= now

MPTCP – Simple Case



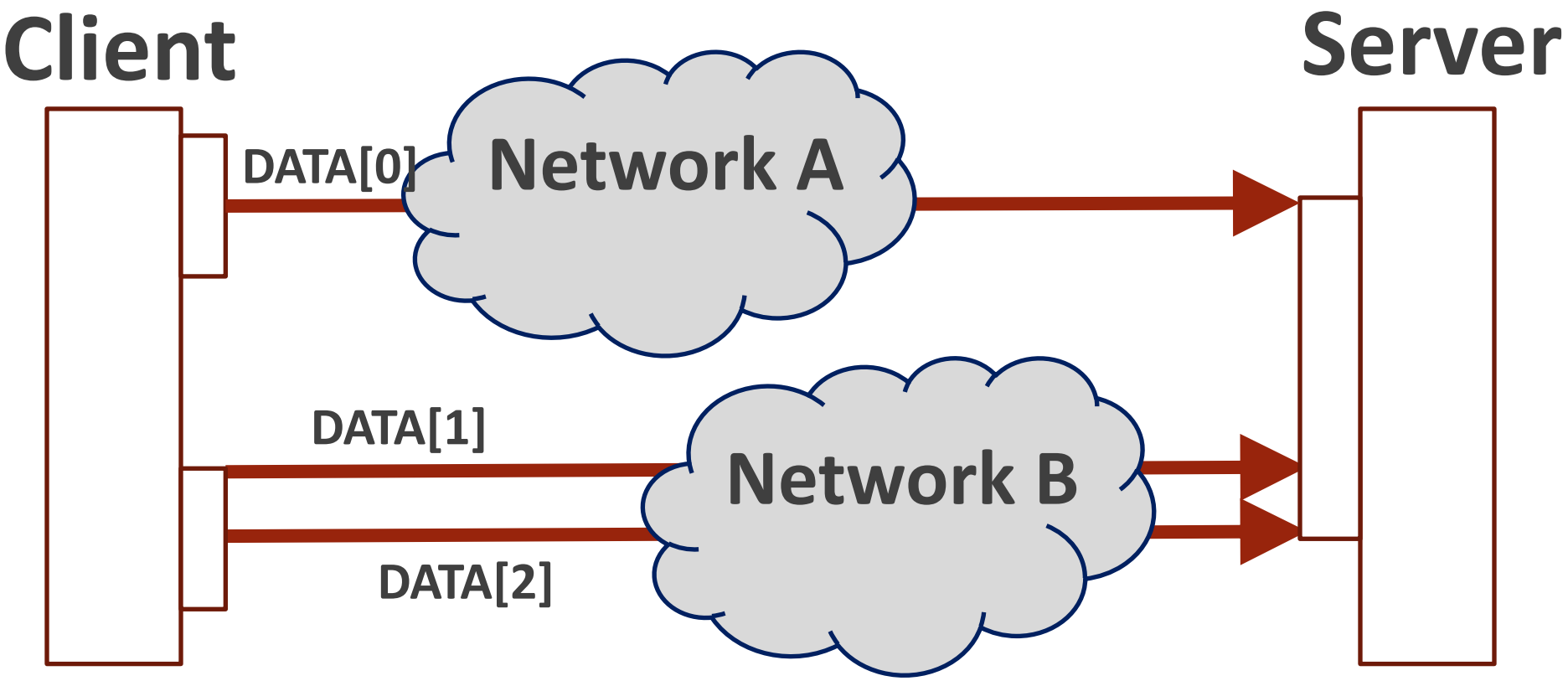
MPTCP connection looks like TCP so far…

MPTCP – Simple Case



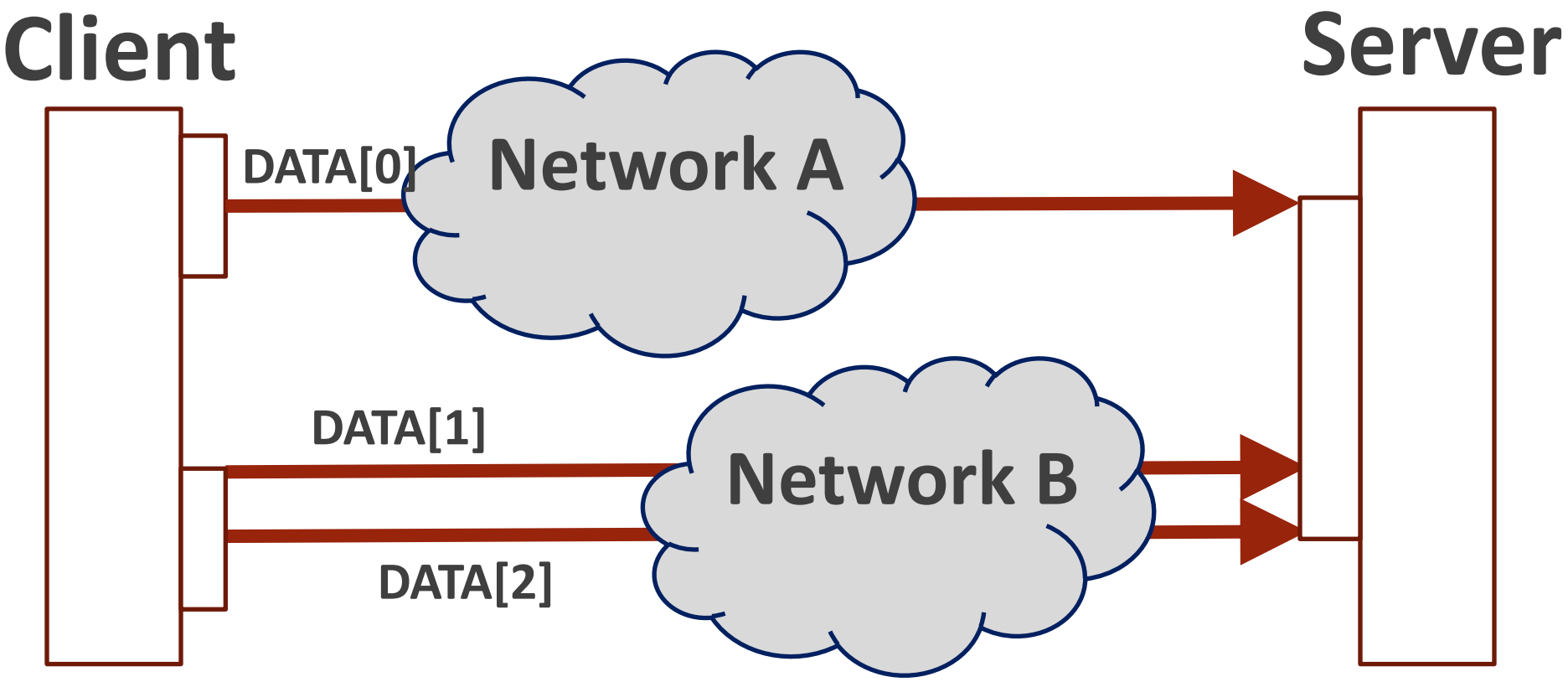
N different TCP connections, contributing to ***ONE*** logical data flow

MPTCP – Simple Case



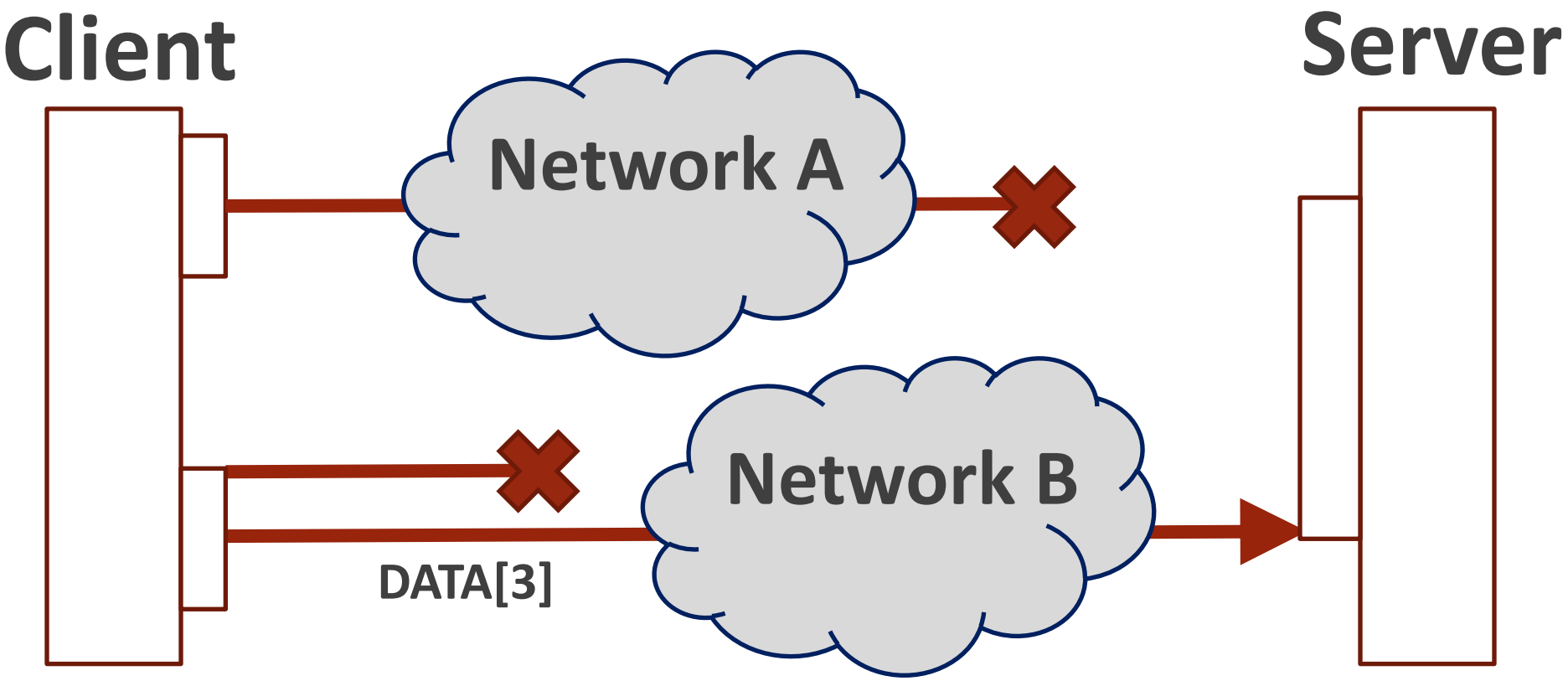
N different TCP connections, contributing to ***ONE*** logical data flow… data flows through any/all

MPTCP – Simple Case



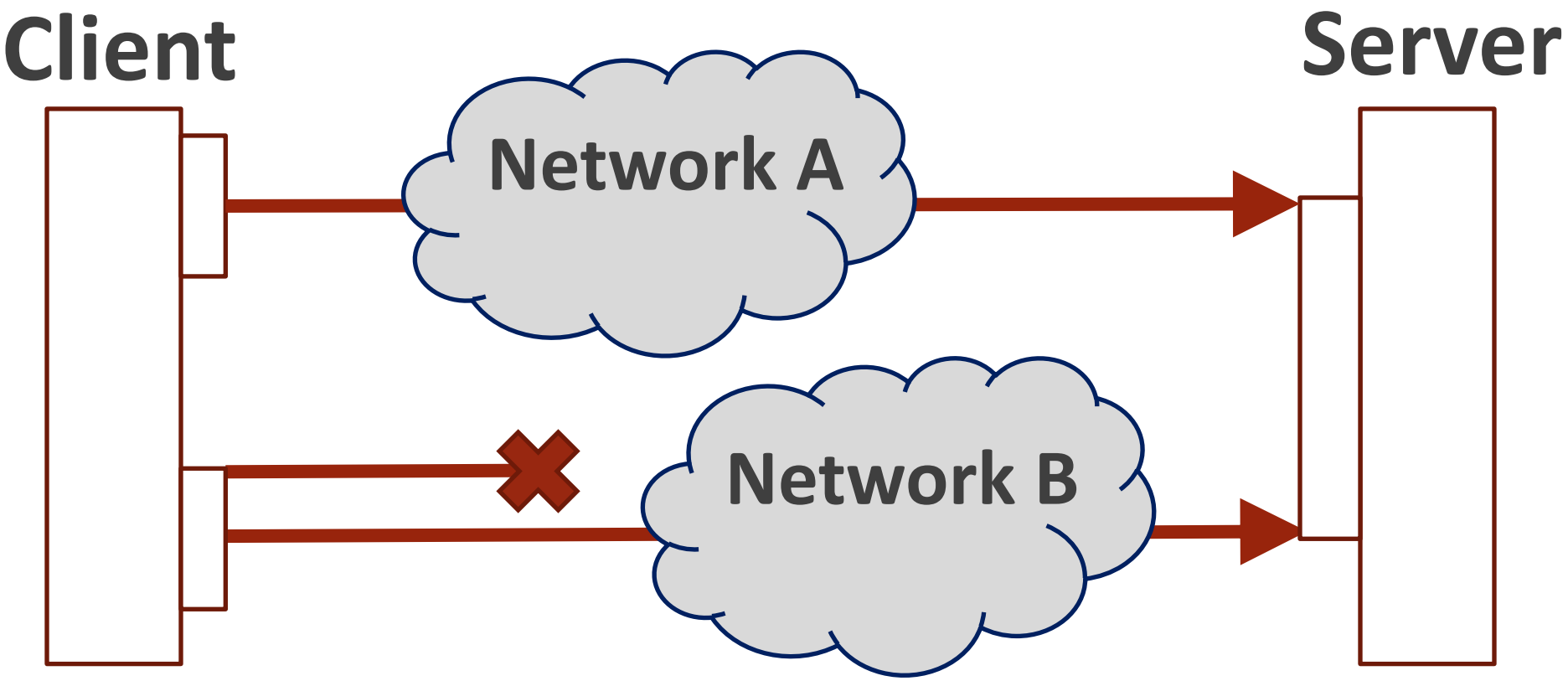
Sender of a packet can choose to use any flow *(this will be important)*

MPTCP – Simple Case



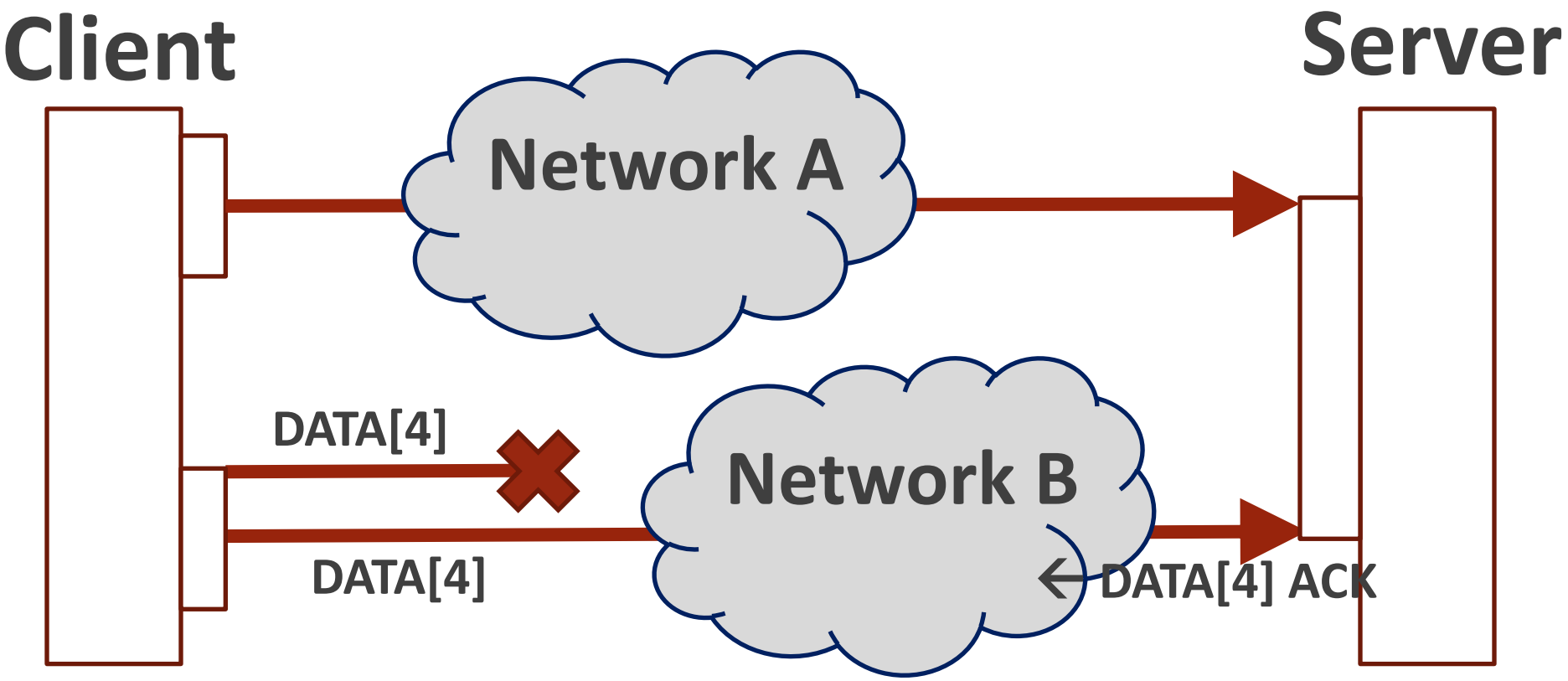
Any subset of connections can drop, overall flow continues.

MPTCP – Simple Case



Connections can be re-added at any time

MPTCP – Simple Case

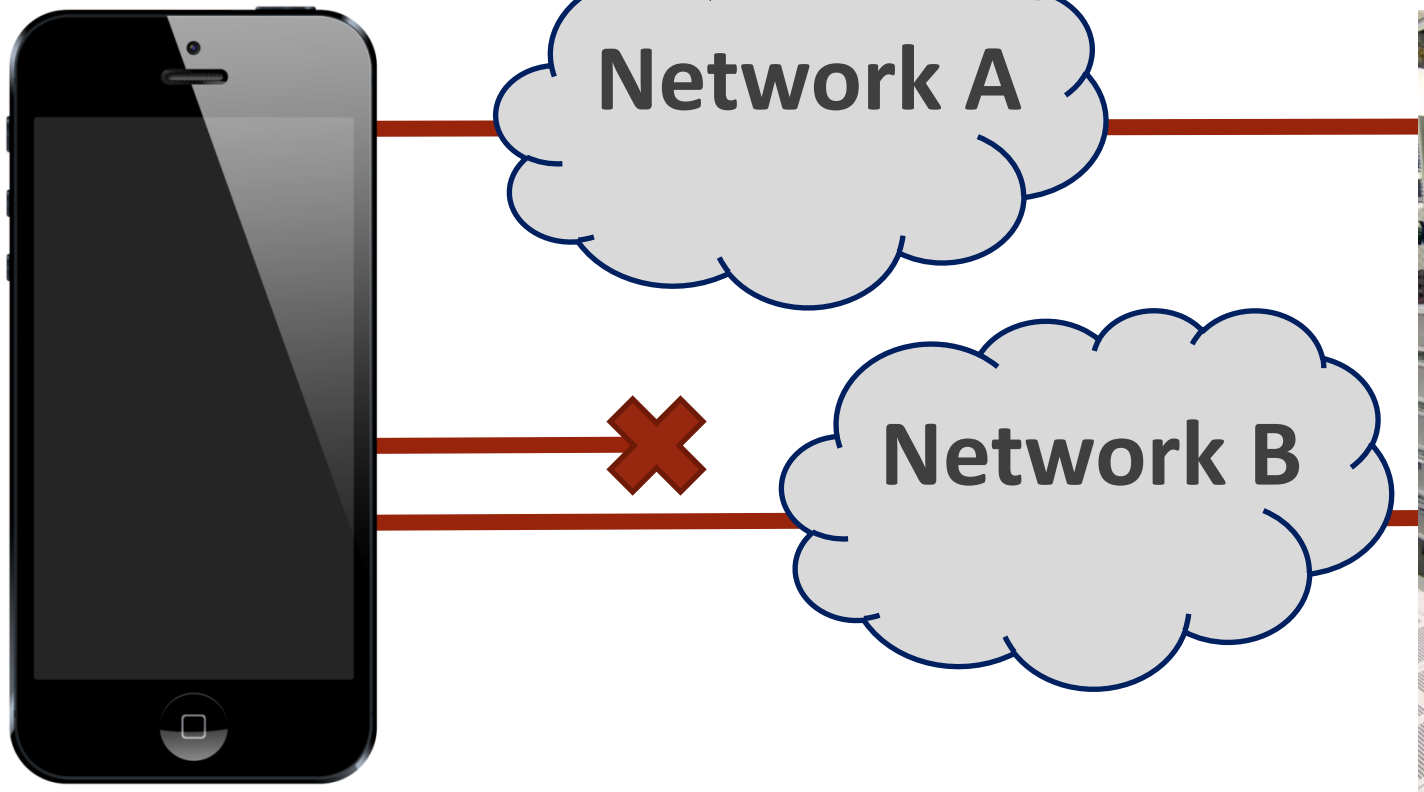


Un-ACK'd data can be quickly resent over a different flow… first ACK is good enough!

MPTCP – Basic Use Cases

Client

Server



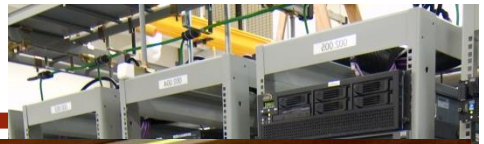
For seamless roaming

For high availability

MPTCP – Basic Use Cases

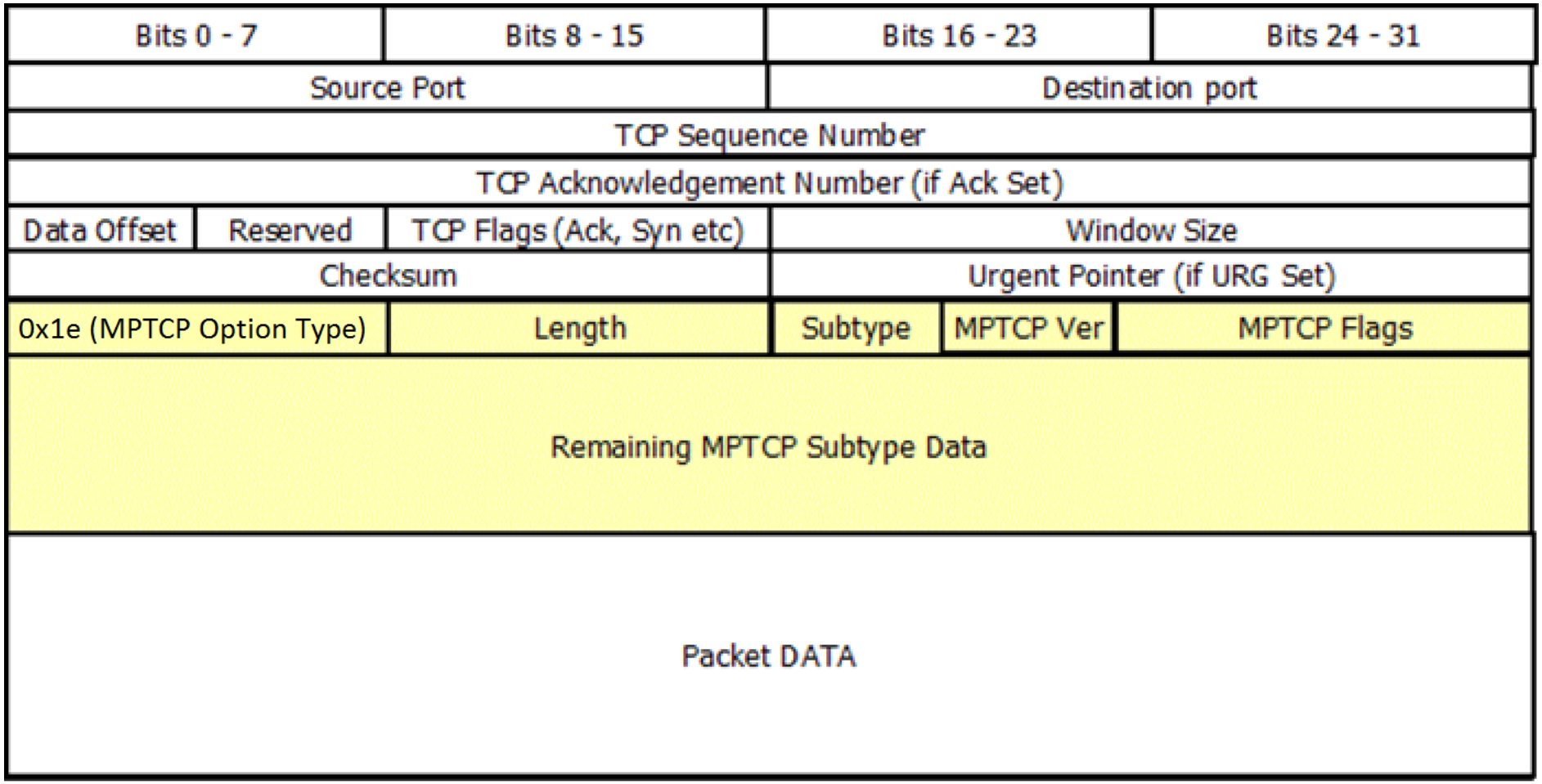
Client

Server



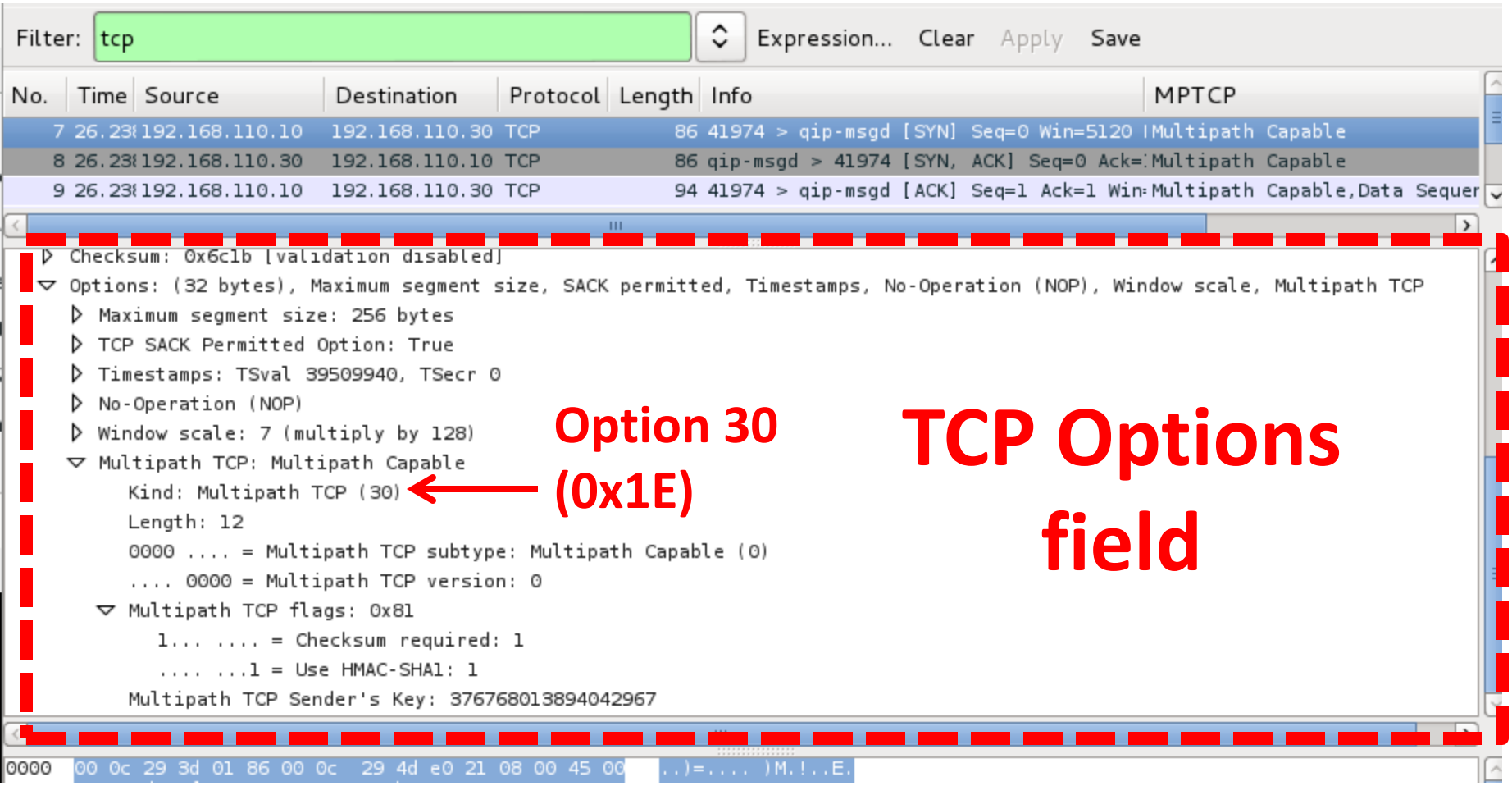
For seamless roaming

How is MPTCP implemented? – TCP Option



What does it look like?

- Packet Breakdown - WireShark



How is MPTCP implemented? – MPTCP Subtypes

- 8 currently defined (ones relevant in **bold**)

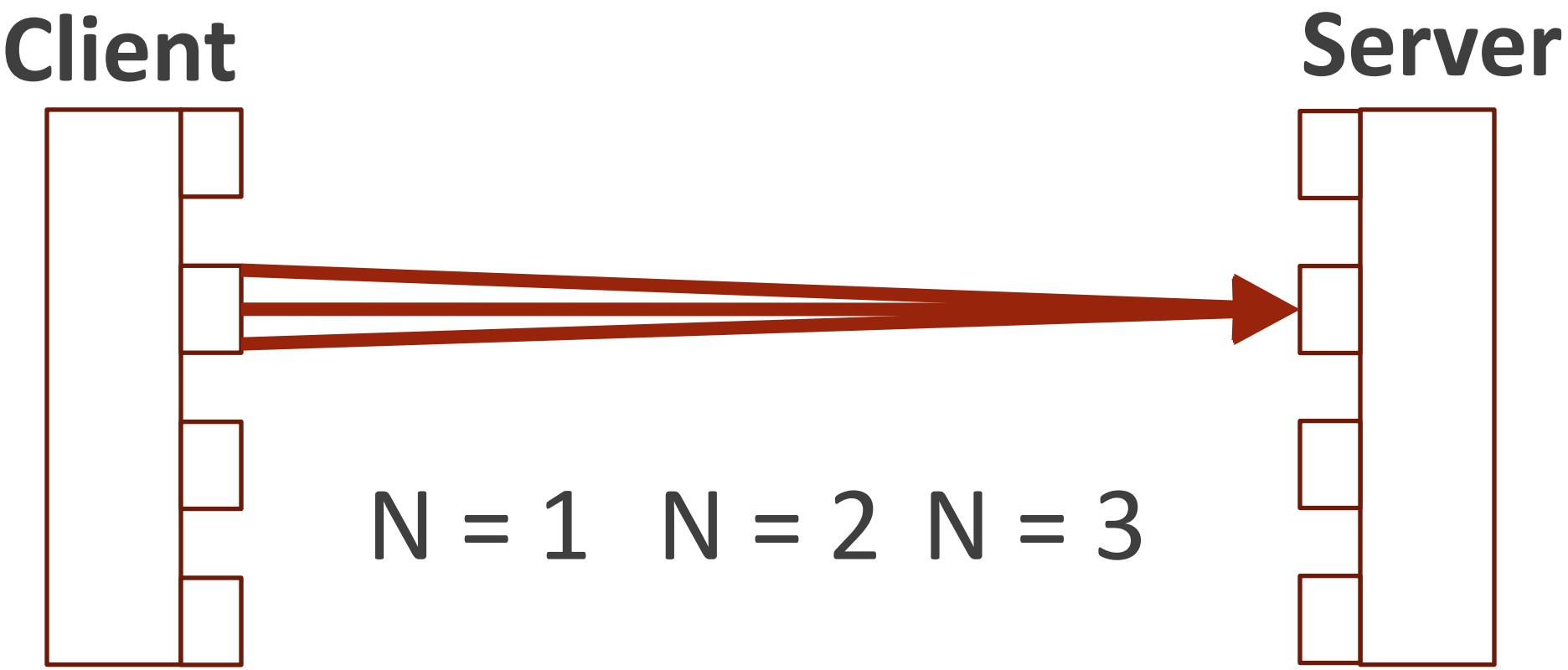
- **MP_CAPABLE -** Signals MPTCP support
- **MP_JOIN -** Add incoming subflow to the connection
- **DSS -** How to map this stream's data against the overall data flow
- **ADD_ADDR -** This address is also a way to reach me
- **REMOVE_ADDR -** Please stop using [address] to reach me
- MP_PRIO
- MP_FAIL
- MP_FASTCLOSE

Path Management - Linux

- The Linux Path Manager has two primary path managers at present
  - Fullmesh – n:n (all to all)
  - Ndiffports – 1-1 interfaces, n-1 ports

- This is in the **TCP stack**… application layers get MPTCP for free (mostly)

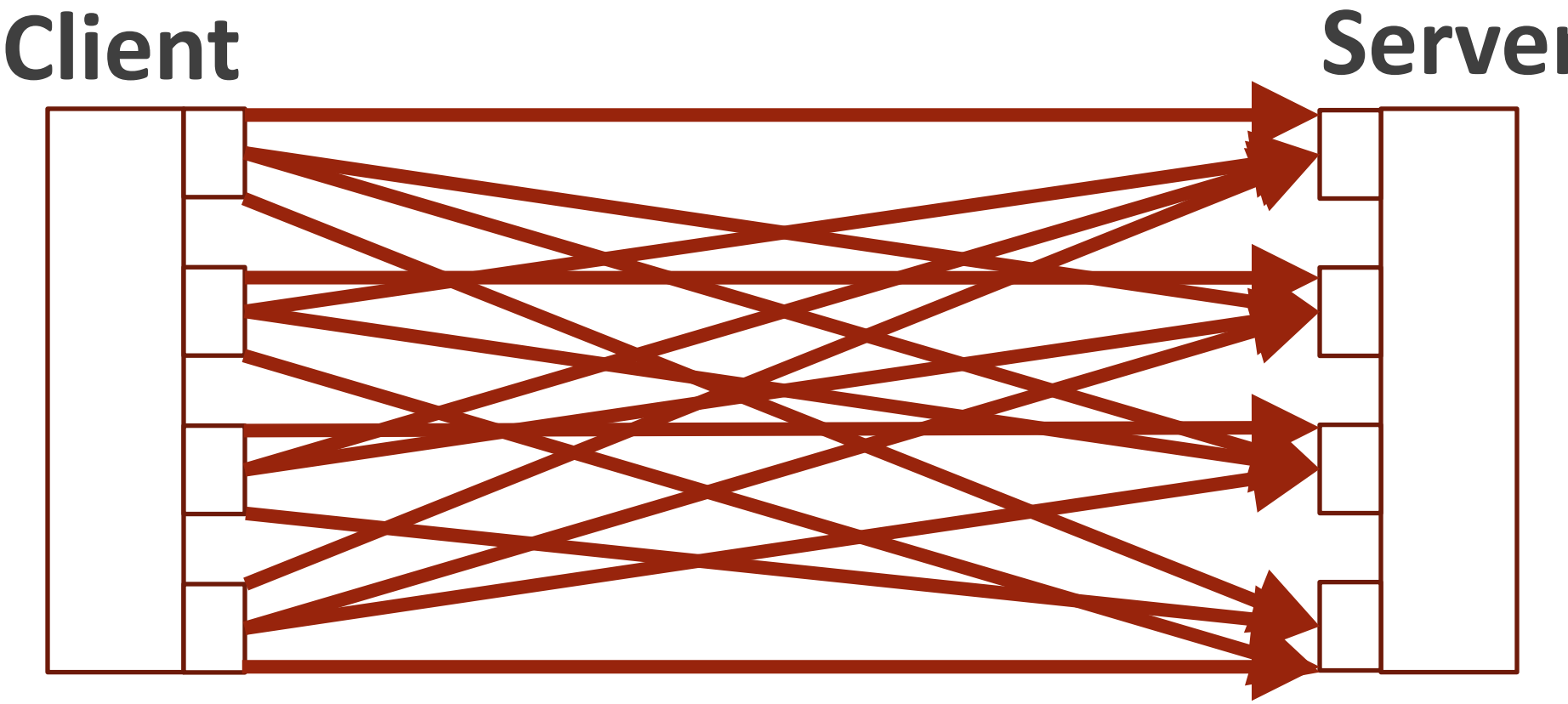
Path Management - ndiffports



N different source ports,

1 destination port

Path Management - fullmesh



All possible paths used

Deeper technical details

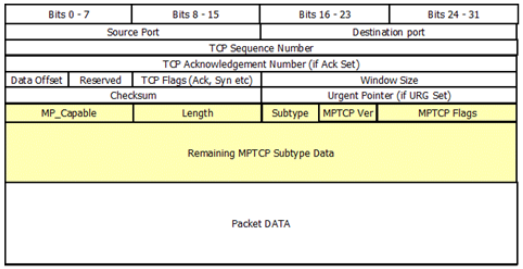
- TCP Handshake with additional details
- Data sequence numbering
  - Truncation of SHA1 of host key
- Authentication
  - MP_JOIN - Challenge-response HMAC of other host's key, Nonce, AddressID
  - MP_FASTCLOSE – Other party's key in plaintext
- Routing
  - Packet ***sender*** decides which data goes down which path

More on this later…

Cheatsheet!

MPTCP Cheatsheet

MPTCP Header:



MPTCP Subtype	HEX	Flags?	Other Likely fields of interest
MP_CAPABLE	0x0		
MP_JOINs	0x1		
DSS	0x2		
ADD_ADDR	0x3		
REMOVE_ADDR	0x4		
MP_PRIO	0x5		
MP_FAIL	0x6		
MP_FASTCLOSE	0x7		

Getting the MPTCP Sequence Numbers:

Key	64 Bit number supplied by host	
Initial DSN (ISDN):	SHA1(key)[-64:]	Binary mode hash, network byte order
Initial DSS		
Subflow DSS	mapping likely starts at ISDN[0:32] + TCP ISN + 1	TCP Seq is 32 bits, + 1 for the SYN
MP_JOIN		

MP_JOIN Authentication (RFC 6824 Fig 8)

A		B
TCP_SYN, MP_JOIN (TokenB, NonceA) ->		
	<- TCP_SYN_ACK, MP_JOIN(HMAC(Key=KB+KA, Msg = NonceA + NonceB), NonceB)	
TCP_ACK, MP_JOIN(HMAC(Key=KA+KB,Msg=NonceB+NonceA) ->		
	<- TCP_ACK	
Token = ConnectionID = SHA1(Key)[0:32] of Other Party's key. (Capture from either steps 2 or 3 in the first handshake)		

Detecting MPTCP things

Passive:	Usage	Inbound Connection Attempts	Detect inbound connection attempts - Look for the SYN packets with MPTCP Header	TCP(SYN) TCP Option= 30 ** 00 ...
		Successful Handshake	(Pre-viability) Look for Ack Packets with MPTCP Option header	TCP(ACK) TCP Option = 30 ** 00 ...
		Valid Handshake	MPTCP Option header Look for Ack Packets with the MPTCP Option Header	TCP(ACK) TCP Option = 30 ** 00 ...
		MPTCP Joins	TCP SYN Packets with MPTCP TCP Option and an MP_JOIN subtype	TCP(SYN) TCP Option = 30 ** 01 ...
	Attacks	MPTCP Simple	Non look for non sequential last	

So who's using it?

- Nearly no one is using it large scale (yet), with a few exceptions
  - Apple iOS7 (Siri) → enabled by default in Yosemite (?)
  - Some other experimental stuff?

- Given that, there's a surprising number of implementations
  - Implementations available for several OS's (including Linux, BSD, Android), and baked in some way into commercial kit (Citrix, Cisco, Apple, Oracle, F5)
  - **NOT Windows**

Availability – Getting it working

- Linux
  - Linux reference implementation via apt-get (multipath-tcp.org) -- best way right now
  - Can work in Kali, but … challenges

- Nicolas Maître made ***a ridiculously useful***, near complete, SCAPY implementation
  - We're based some tools on this code, and fixed some bugs along the way
  - https://github.com/nimai/mptcp-scapy

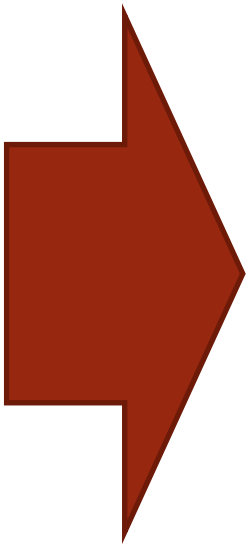
Background
Technical Introduction
Key Security Effects
Perimeter Security
Network Management
MPTCP Future

MPTCP's Key Security Effects

- Cross-path traffic fragmentation
  - That's the whole point!
- Moving target
  - Ability to change source and destination addresses in the middle of a connection
- Connection Resilience
  - Has additional checksums that require capture of the initial packet to reliably fake
  - Until every subflow is dead the overall connection keeps going
- "Reverse" connections

Because of these…

- Cross-path
- Moving target
- Connection Resilience
- Reverse connections



… if your approach to security requires *any* of these…

- See all app layer data in a TCP stream
- Associate logical sessions to IP addresses
- Tamper with or close "bad" connections mid-stream
- Differentiate clients from servers based on connection direction

…then something is probably going to break

How practical are these attacks?

- Today? Extremely.
  - But only if both endpoints speak MPTCP
    - Of which… there aren't many. Yet.

- In an MPTCP world, a bit less
  - But we have to change the way we do things in network security

Practicality Going Forward

- All of those things can be *partially* mitigated with MPTCP aware infrastructure and security tools.

- But overall, there remain some interesting shifts in how network flows work – especially if we go in with “well meaning” intent

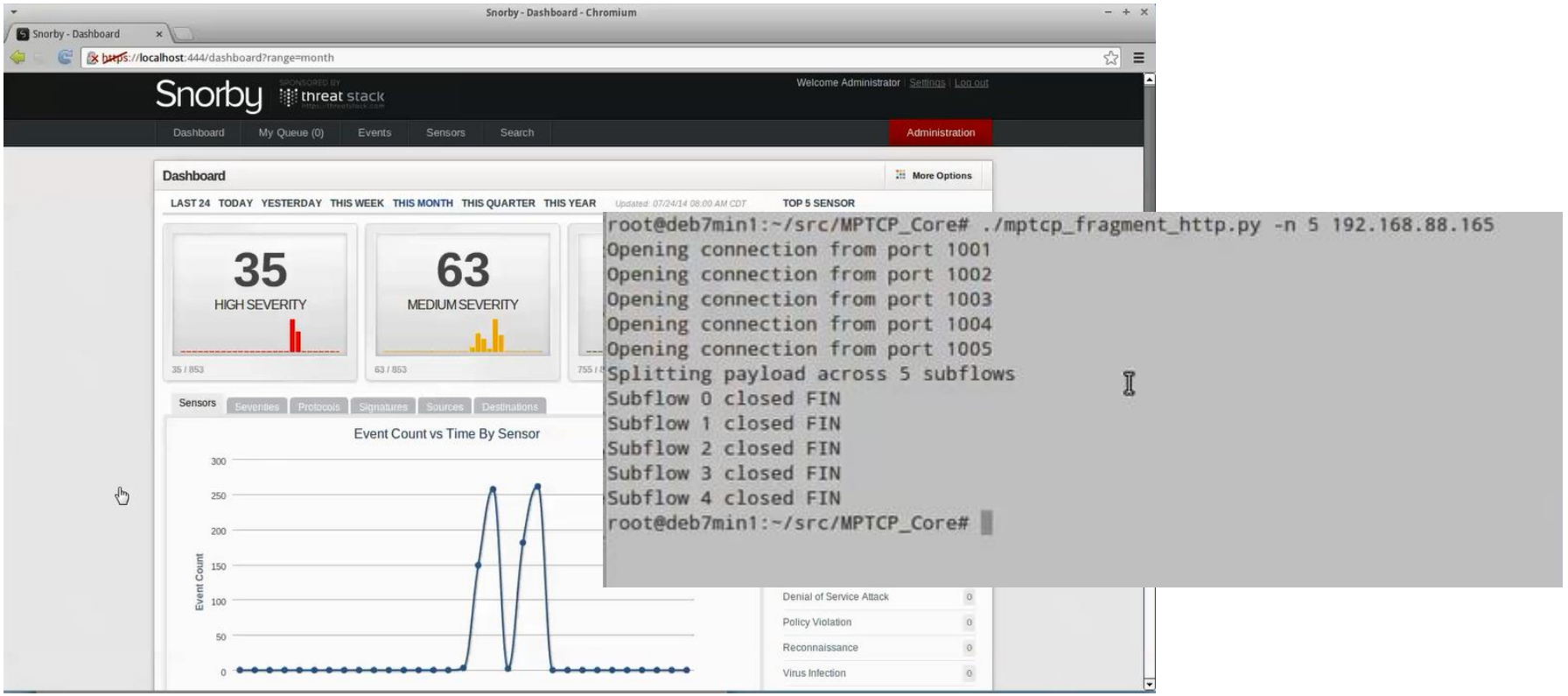
MPTCP's Key Security Effects

A few slides back…

- The **packet *sender* decides** which data goes down which path.
- Normal/benign clients won't choose pathological fragmentation schemes
  - But there's nothing stopping us…

PoC tool for MPTCP IDS Evasion

- Demo!

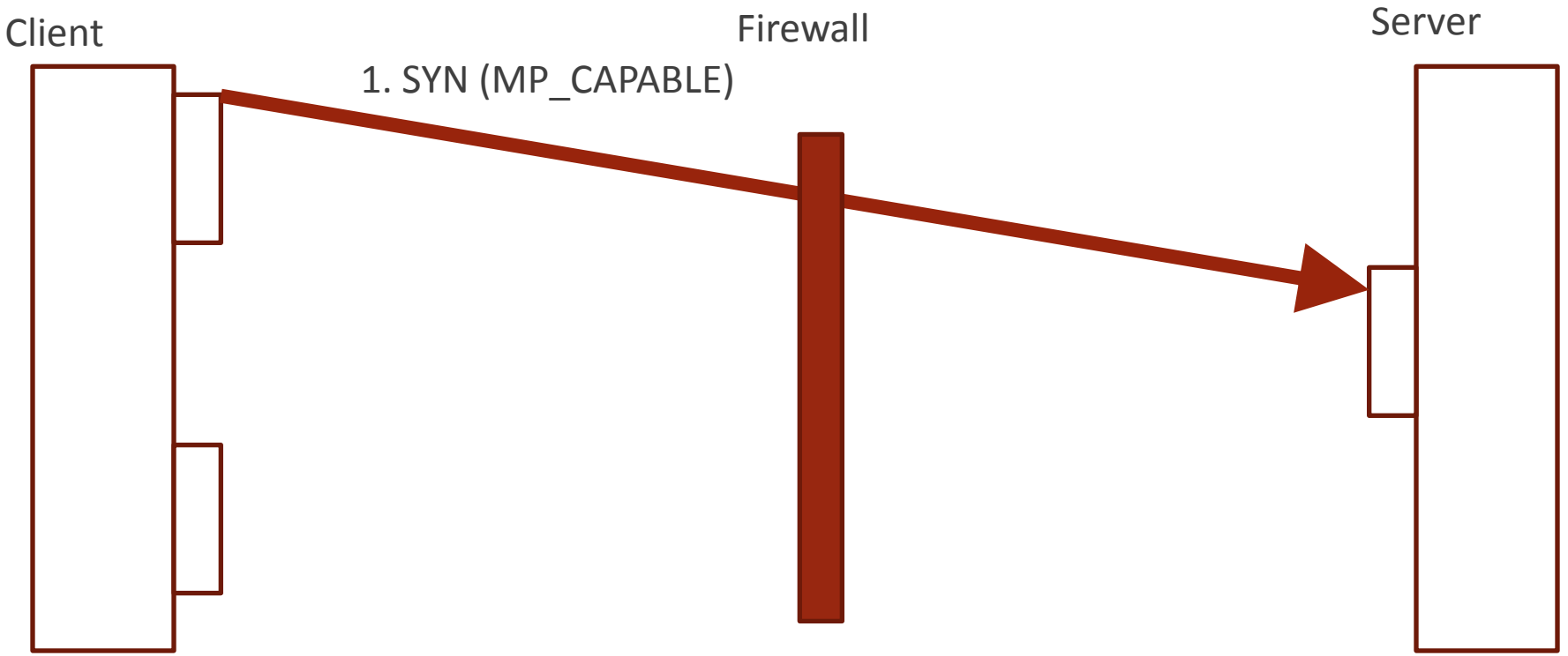


Background
Technical Introduction
Key Security Effects
Perimeter Security
Network Management
MPTCP Future

MPTCP and … Firewalls

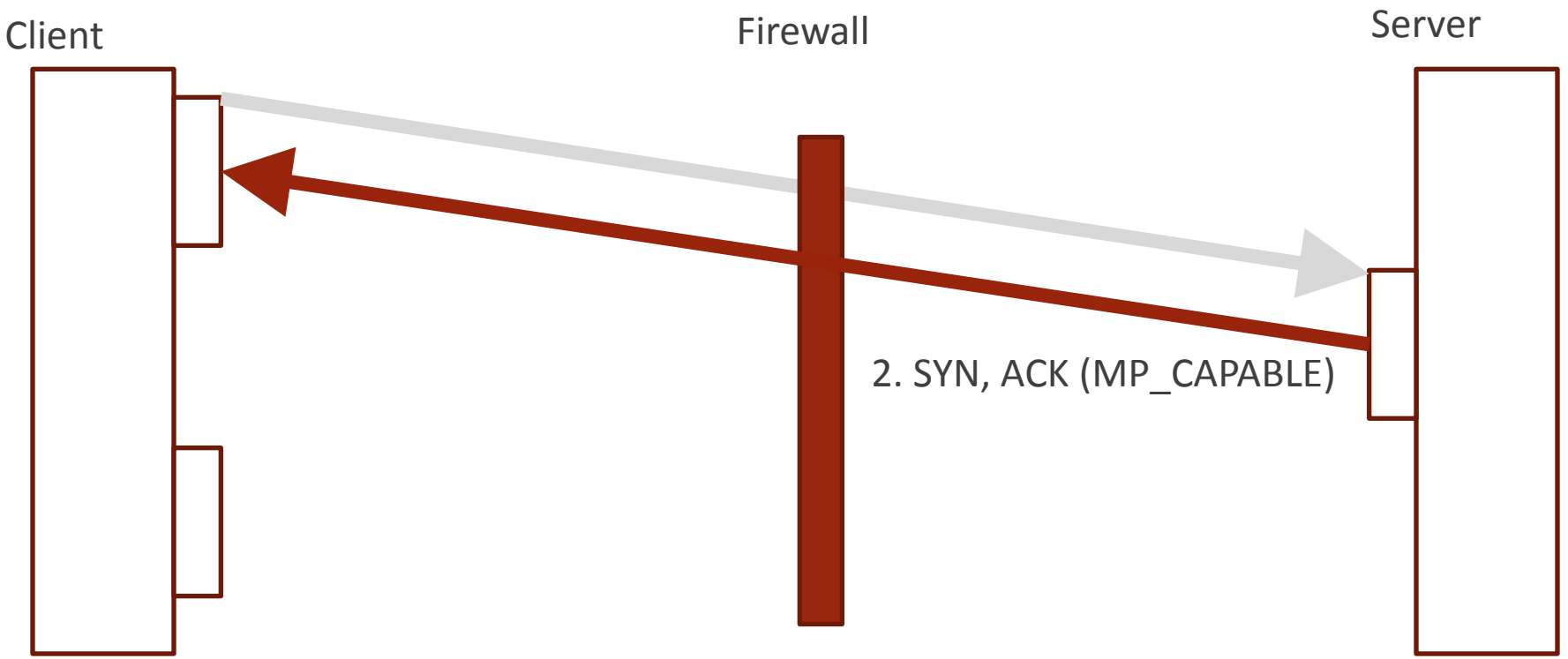
- MPTCP changes things for perimeters
- How’d you like an outbound incoming connection?

MPTCP and … Firewalls



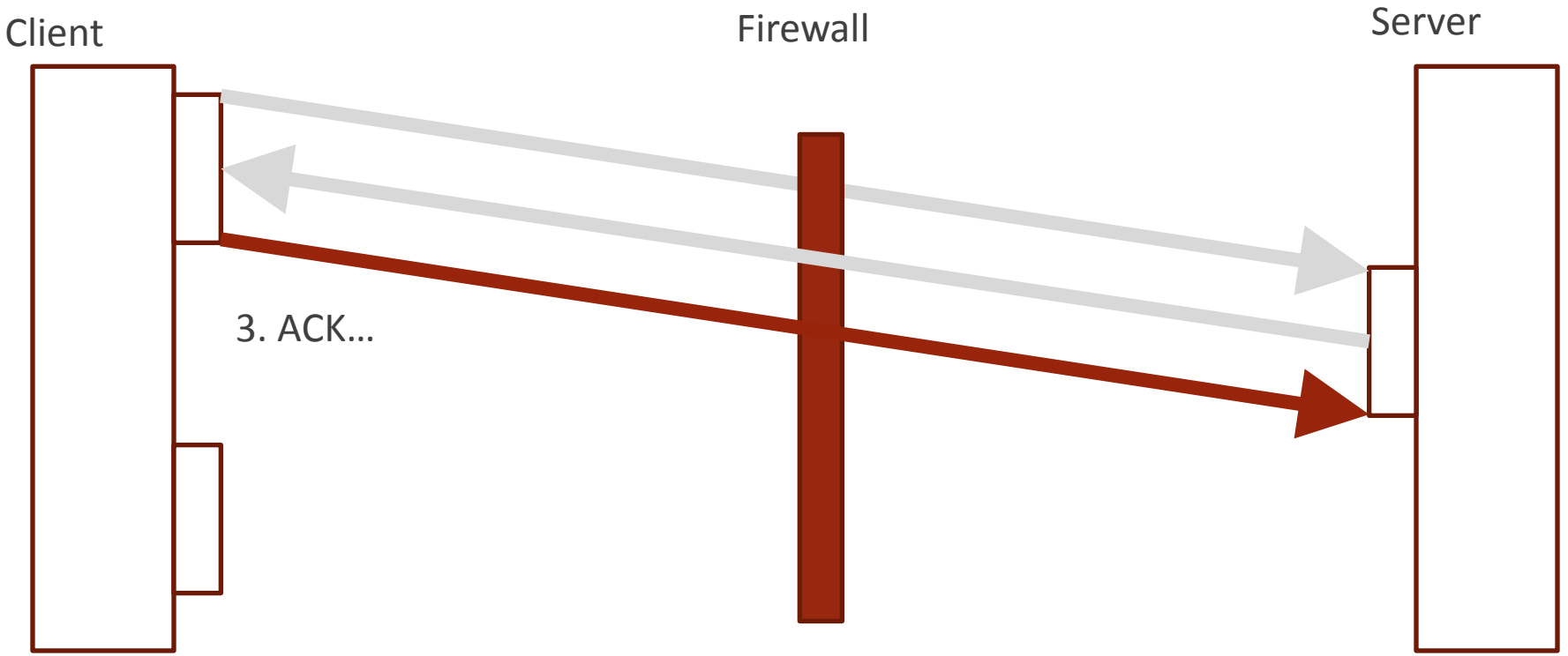
- MPTCP connection looks like TCP so far…

MPTCP and … Firewalls



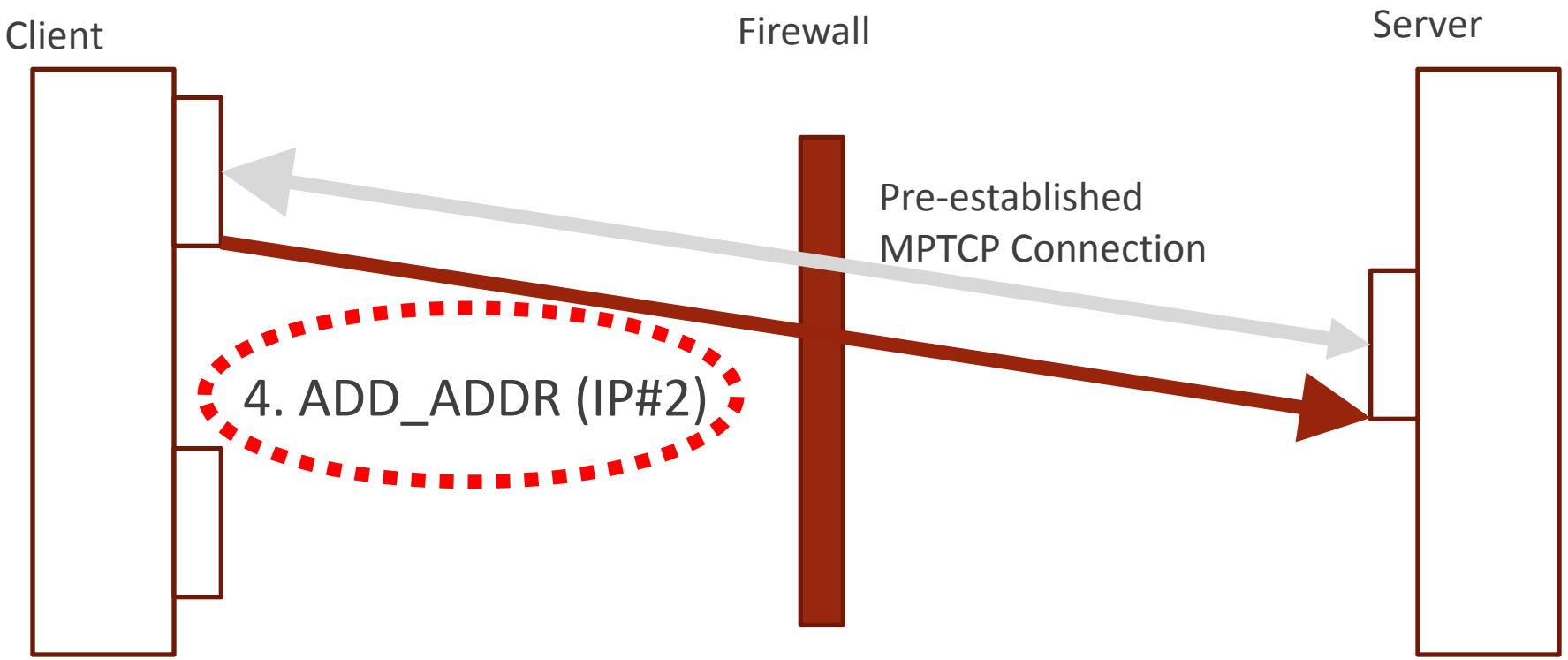
- Still seems pretty standard, albeit with extra TCP OPTIONS

MPTCP and … Firewalls



- OK, so it’s a TCP connection with an additional options… so what?

MPTCP and … Firewalls

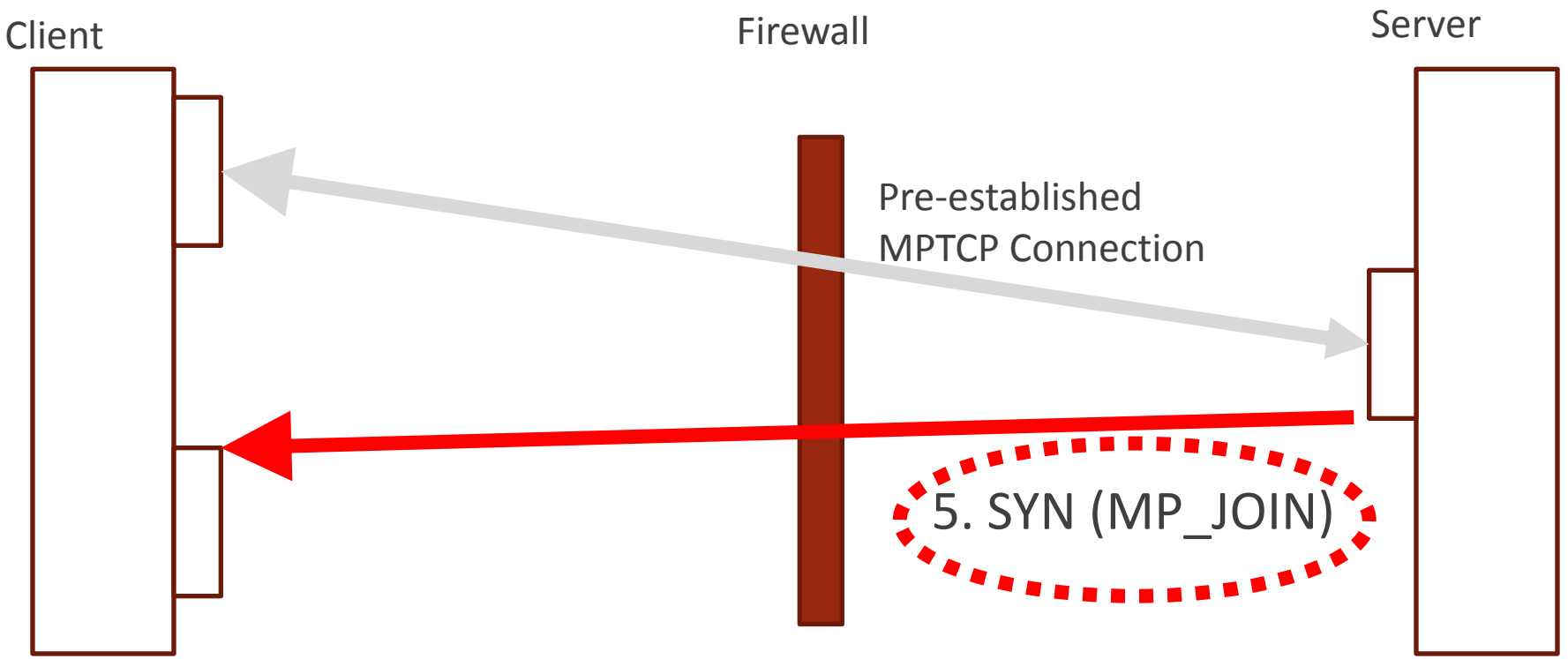


- Well, what if the client tells the server about a new address?

MPTCP and … Firewalls

- Now, the “Internal” host may set up a connection to the advertised address

MPTCP and … Firewalls



- Is this new connection incoming or outgoing?

MPTCP and … Firewalls

- Is this new connection incoming or outgoing?

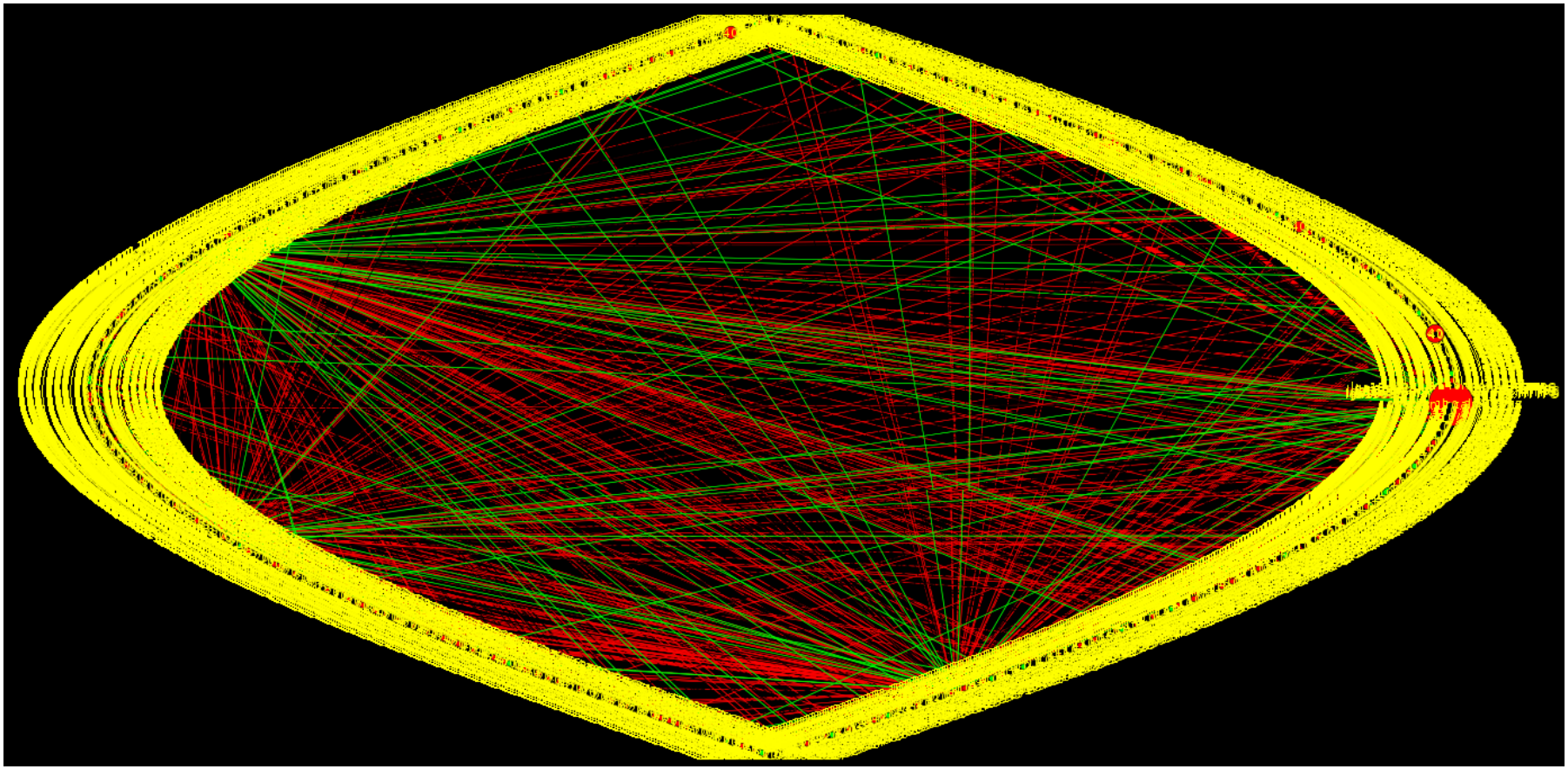
Background
Technical Introduction
Key Security Effects
Perimeter Security
Network Management
MPTCP Future

MPTCP and … Network monitoring

- If tool doesn't understand MPTCP, flows look like unrelated TCP streams

What does it look like?

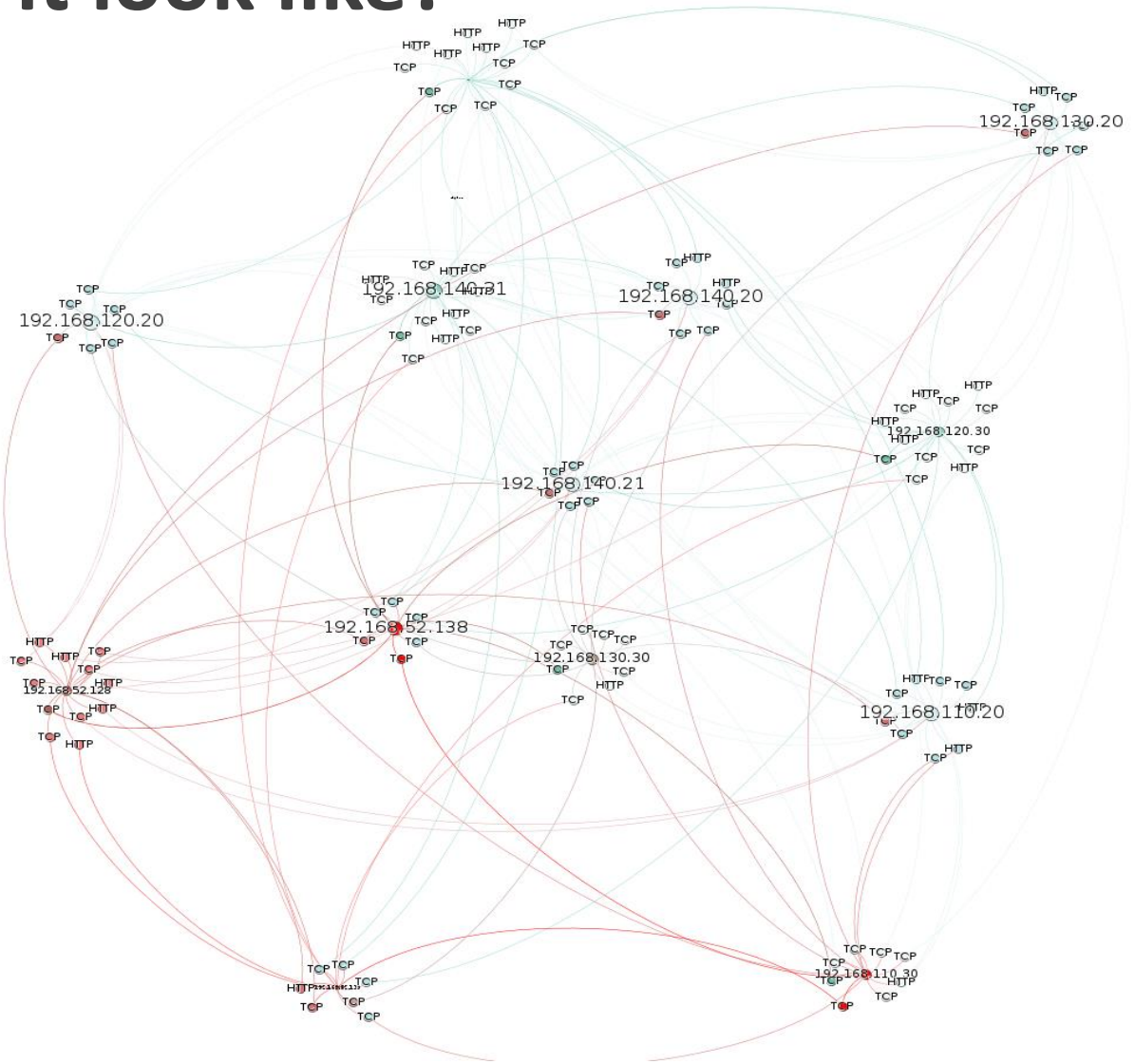
- On the network: If you don't understand



Each yellow blob is actually part of an address label

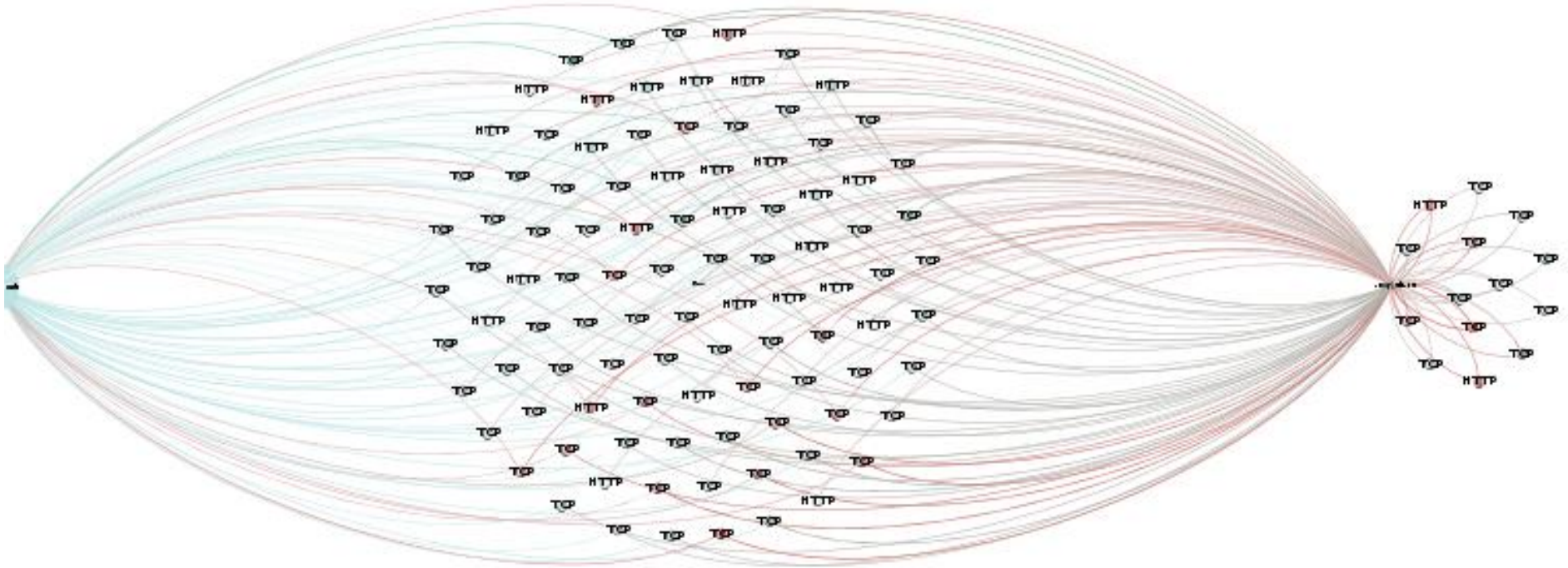
What does it look like?

- On the network: If you don't understand it, but you cluster IPs



What does it look like?

- On the network: If you do understand



- But you can only do this when you can see & correlate **all** related flows…

MPTCP Defense - Awareness

- People
- Technology
  - Check support
  - Look for use
- Architecture
  - Terminate it where you terminate SSL

MPTCP Support Scanner

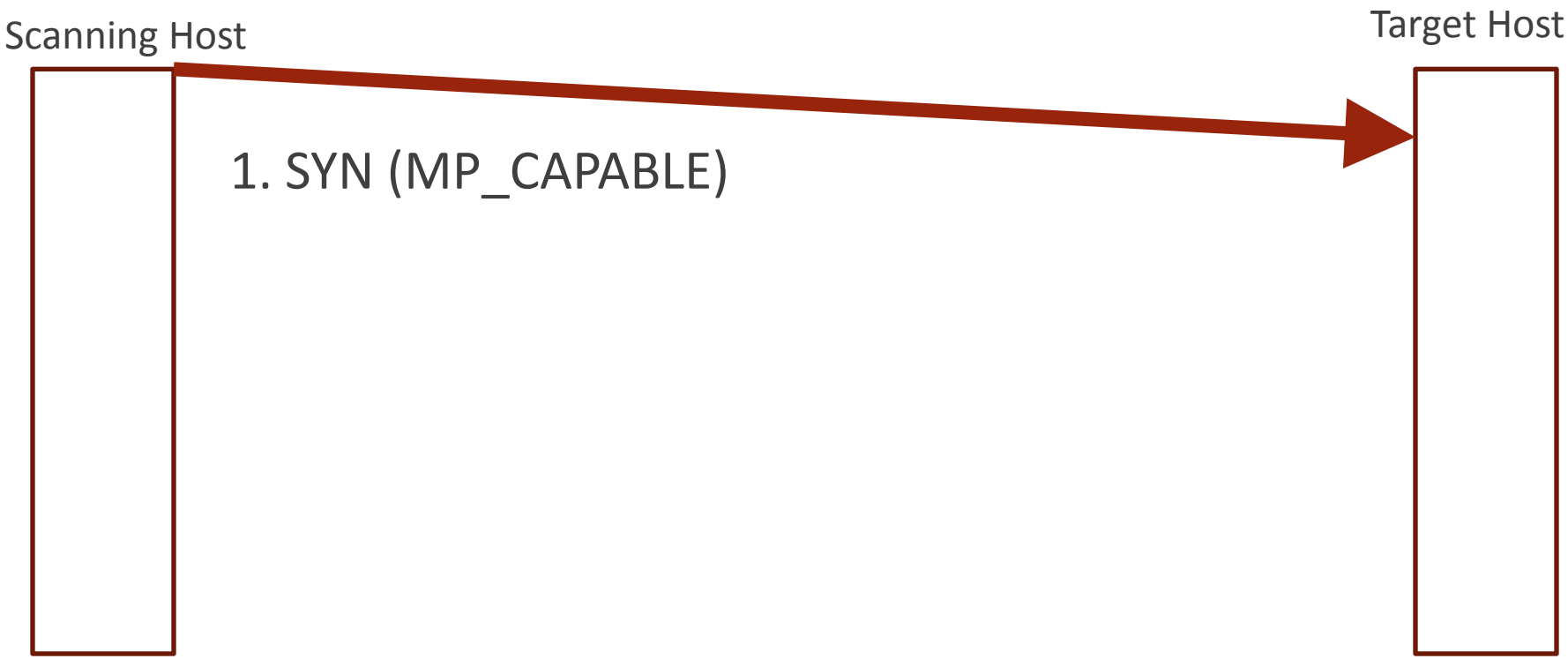
```
root@psthomas-neo-dev:~/mptools# ./scanner.py
usage: scanner.py [-h] [--ip SRC_IP] host port

Network scanner to test hosts for multipath TCP support. Requires root
privileges for scapy.

positional arguments:
  host          comma-separated IPs or ranges (globs allowed), eg
                "127.0.0.1,192.168.1-254,203.0.113.*"
  port          comma-separated port(s) or port ranges, eg "22,80,8000-8999"
```

```
root@psthomas-neo-dev:~/mptools# ./scanner.py 192.168.88.164 22,80
Testing: 192.168.88.164 ... on local network...  at ARP: 00:0c:29:c8:8a:61
 got MPTCP Response from  192.168.88.164 : 22 !...  20
RST Test indicates MPTCP support
 got MPTCP Response from  192.168.88.164 : 80 !...  20
RST Test indicates MPTCP support
****Results:****
        192.168.88.164
                        {22: 'MPTCP (MP_JOIN Verified)'}
                        {80: 'MPTCP (MP_JOIN Verified)'}
```

MPTCP Support Scanner

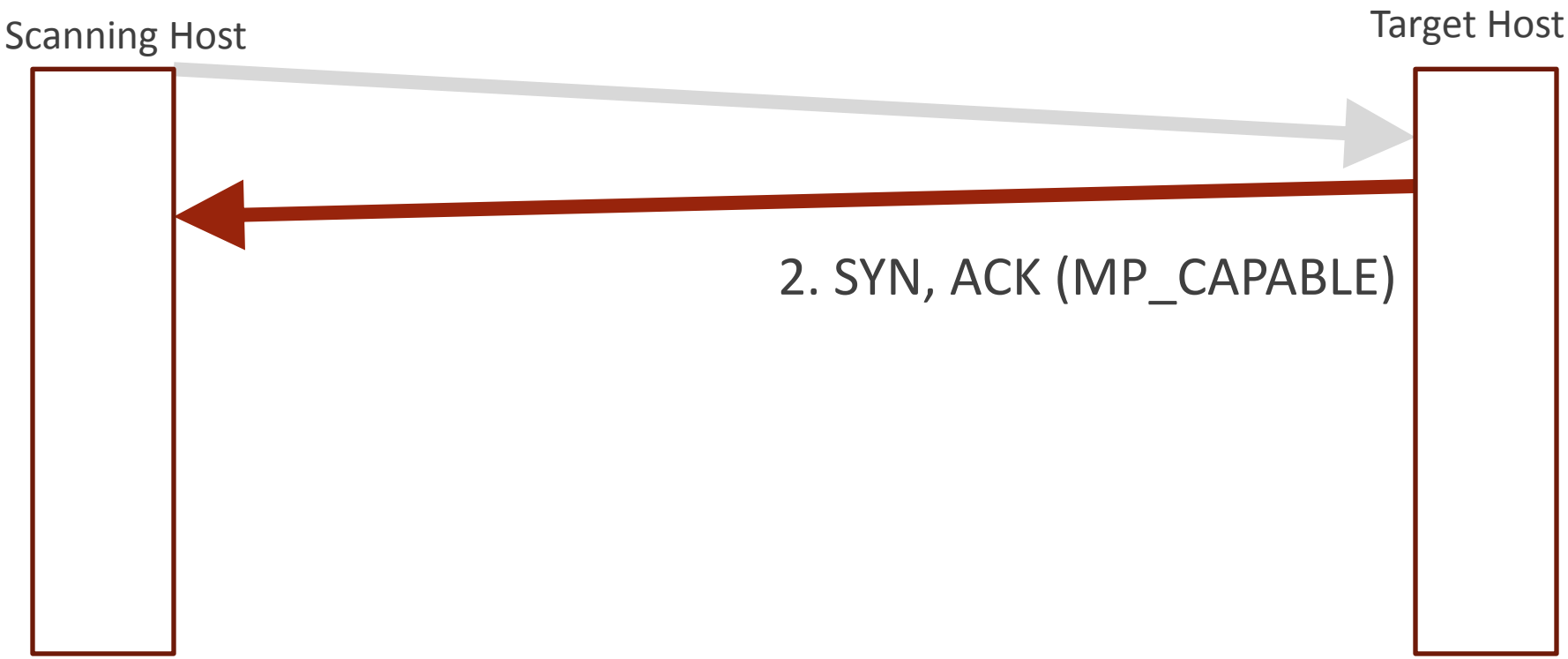
- Accomplishes three things
  - Test device for *apparent* support
  - Test for *actual* support (as opposed to repeating the option blindly)
  - Test network path allows it to get there

MPTCP Scanner



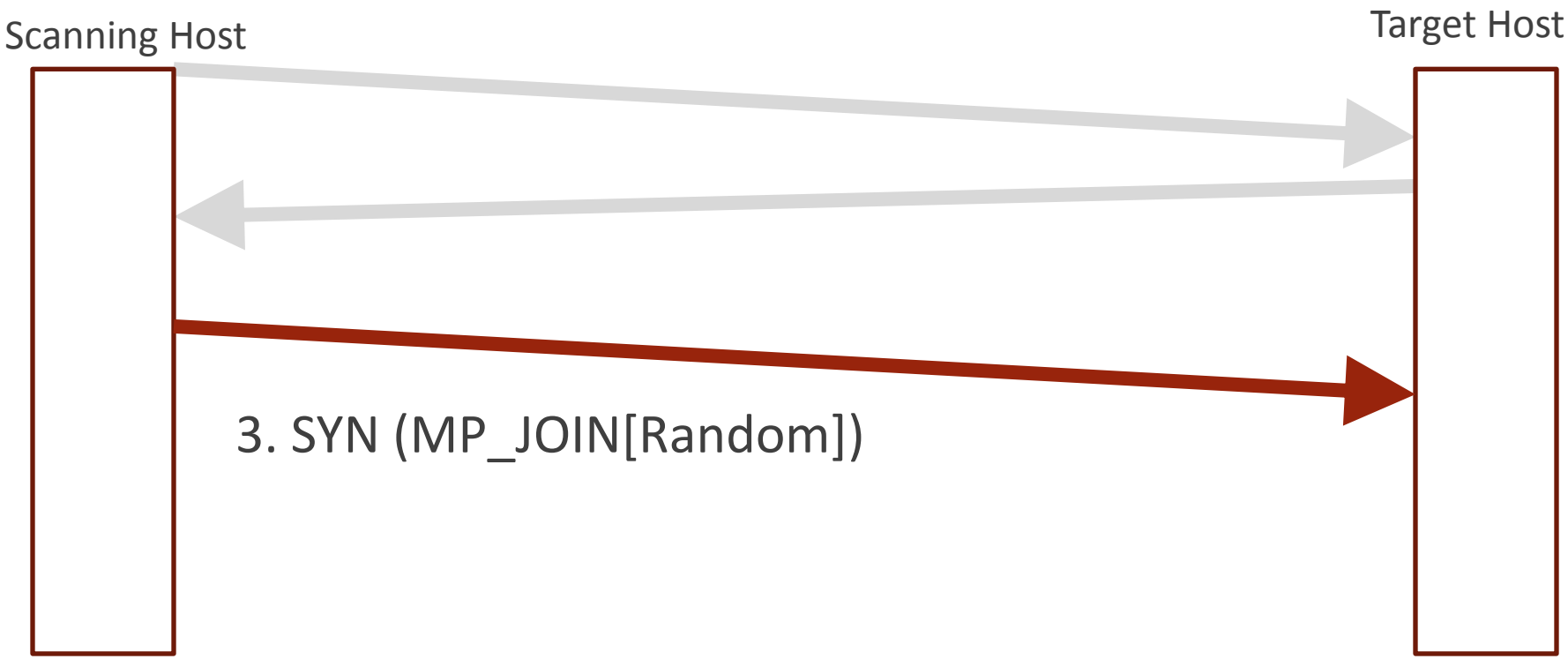
- Send an MP_CAPABLE syn

MPTCP Scanner



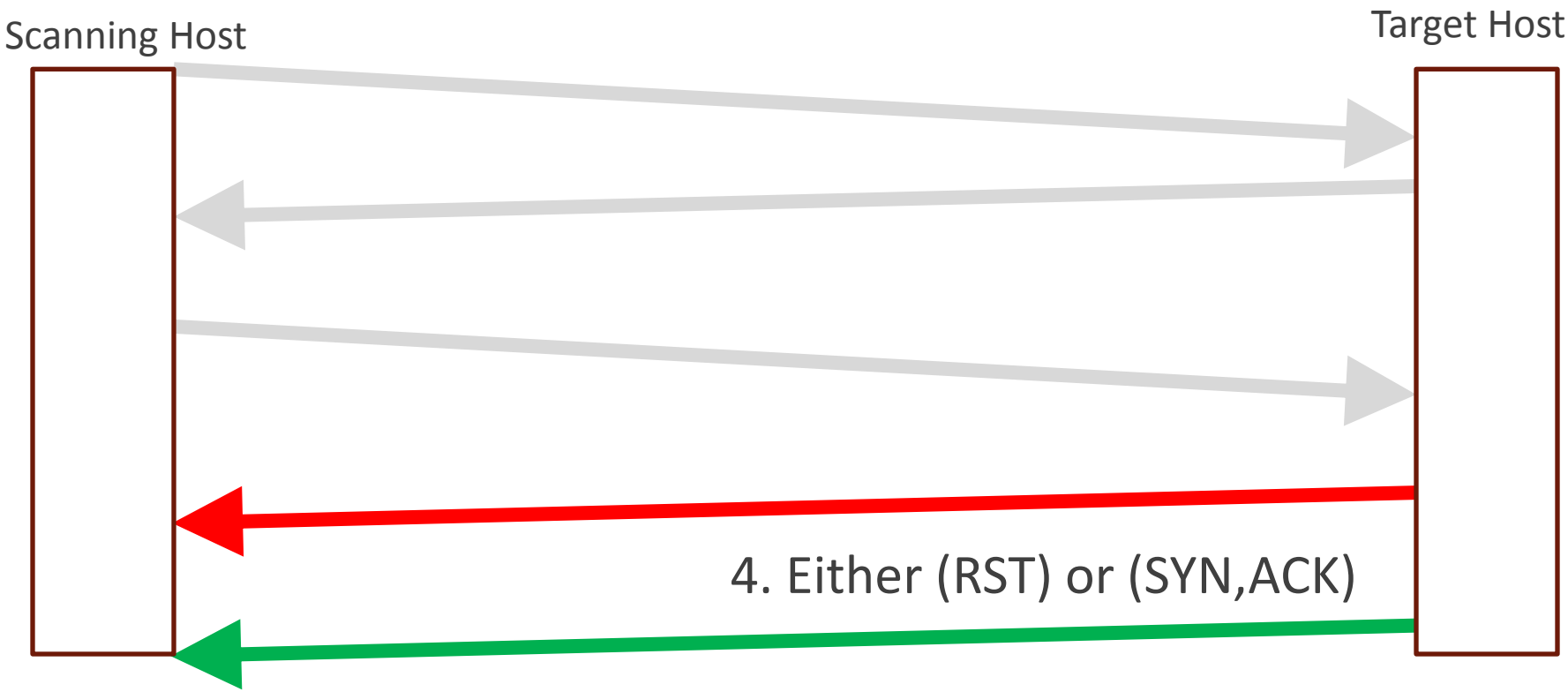
- We got an MP_CAPABLE response.. But is it genuine?

MPTCP Scanner



- Send a join to an invalid connection ID

MPTCP Scanner



- An MPTCP host will RST an invalid join,
- An ACK reply indicates TCP only

MPTCP Stripping

- Transparent proxy on primary path
  - Either no MPTCP support, or only on the one interface

- Firewall rules:

strip-options 30 - iptables,

tcp-options 30 30 clear - Cisco IOS

MPTCP and Active Network Security

- To track & modify MPTCP, you must

1. Capture the initial handshake

2. Perform non-trivial calculations to determine
   - Connection membership
   - Correct checksum or modified traffic

Background
Technical Introduction
Key Security Effects
Perimeter Security
Network Management
MPTCP Future

MPTCP and … Privacy

- MPTCP shifts power towards endpoints, and away from infrastructure & ISP's
- I don't trust my ISP or Cellular company…
- But they probably don't trust each other either!

The MPTCP Future

- What will change in a multipath future that simply cannot work with most existing security models?
  - Split trust crypto
  - Multipath agility
- Some research into privacy effects already underway

Conclusions

- Multipath communications are awesome, and they're coming
- Multipath communication confounds business & security models relying on inspection
- Now is the time for network security to prepare

Try It!

PCAP Challenge (with solution) @

http://bit.ly/MPTCPTROOPERS15

Questions?

Catherine Pearce
@secvalve
cpearce@neohapsis.com

Patrick Thomas
@coffeetocode
pthomas@neohapsis.com

Downloads:
https://github.com/Neohapsis/mptcp-abuse

More stuff will be released @
http://labs.neohapsis.com