



# SHARKFEST '13

Wireshark Developer and User Conference

## PA-17 TCP Performance Problem Analysis Through Multiple Network Segments

Jasper Bongertz / Christian Landström  
CASSIDIAN Cyber Security



# Multipoint Analysis

---

- A single measure point is not sufficient for certain network analysis tasks
- Typical scenarios for multipoint analysis
  - Assumed packet loss between client and server
  - Determining Latency
  - Investigating packet manipulation when passing certain network devices
  - Asymmetric routing
  - Link Aggregation
  - Active/Passive and Active/Active High Redundancy Solutions

# Multipoint Analysis: Best practice

---

- Extremely important: Document your traces as detailed as possible!
  - Especially when dealing with loads of trace files from multiple capture points
- Sync the time of your capture devices



# SHARKFEST '13

Wireshark Developer and User Conference

## Comparing trace files



# Comparing traces

---

- Comparing traces taken at multiple points at the same time is often necessary
- Major points of interest are:
  - Identify identical packets at each capture point
  - Isolate conversations and match them
  - Determine latency
  - Determine packet loss
- Can be quite time consuming unless done automatically (e.g. Pilot)

# Identifying packet matches

---

- Find identical TCP/UDP conversations:
  - Determine client/server socket pairs
  - Create conversation filter, apply to all capture points
  - When using multiple files per location: batch job
- For other protocols, try
  - ARP: sender/target MAC and IP in the ARP header
  - ICMP: type, code, ping sequence, packet quote
  - DHCP, DNS: transaction ID
  - GenericIP: IP-ID, TTL

# Isolating TCP conversations

---

- Filter on the conversation, e.g.
  - `(ip.addr==10.0.0.1 and tcp.port==1025) and (ip.addr==10.0.0.2 and tcp.port==80)`
- Save into separate file using “Export specified packets” -> “Selected displayed packets”
- If possible: isolate initial SYN packet
  - `tcp.flags==2`
- Best Practice: deactivate relative TCP sequence numbers!



# SHARKFEST '13

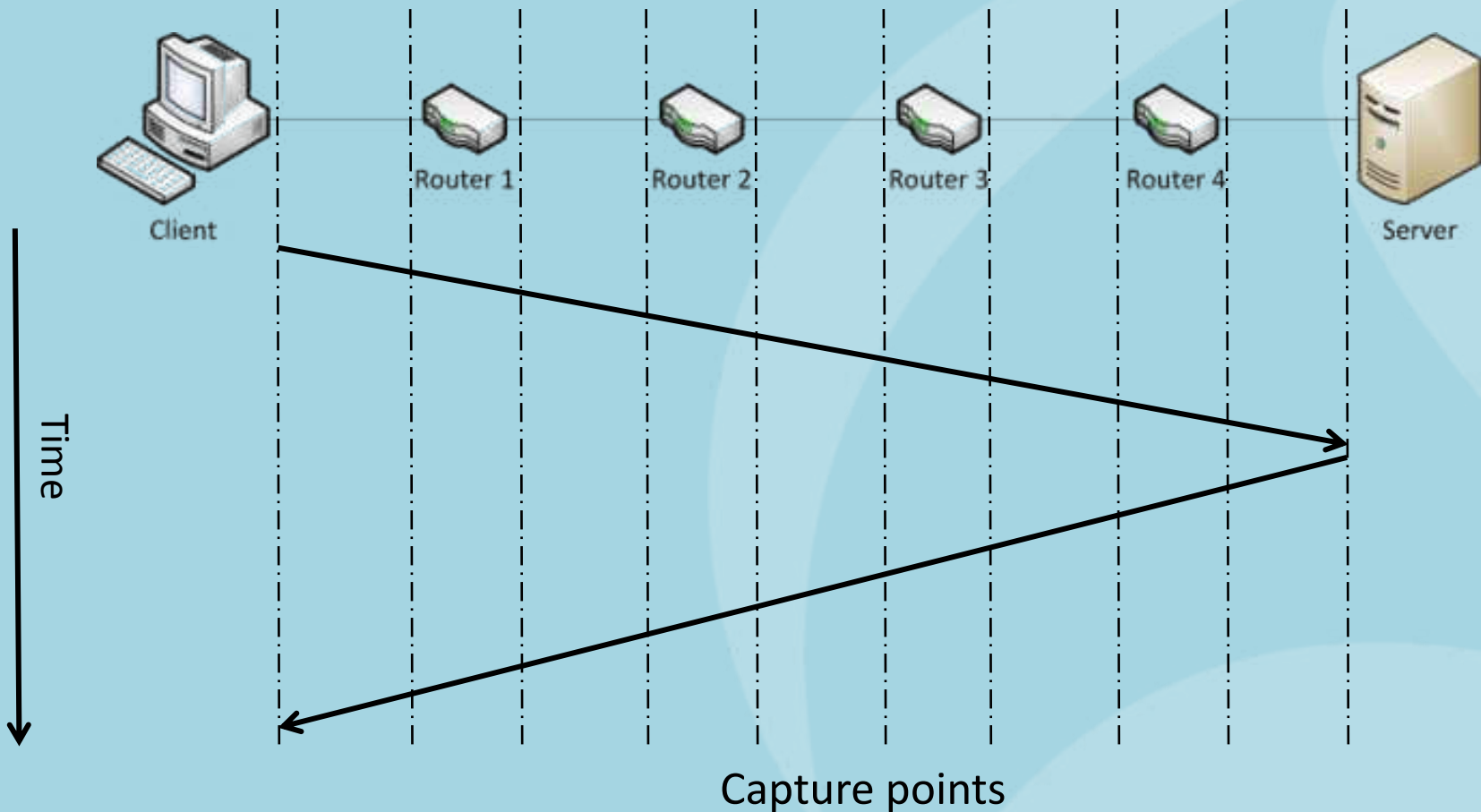
Wireshark Developer and User Conference

## Determining latency

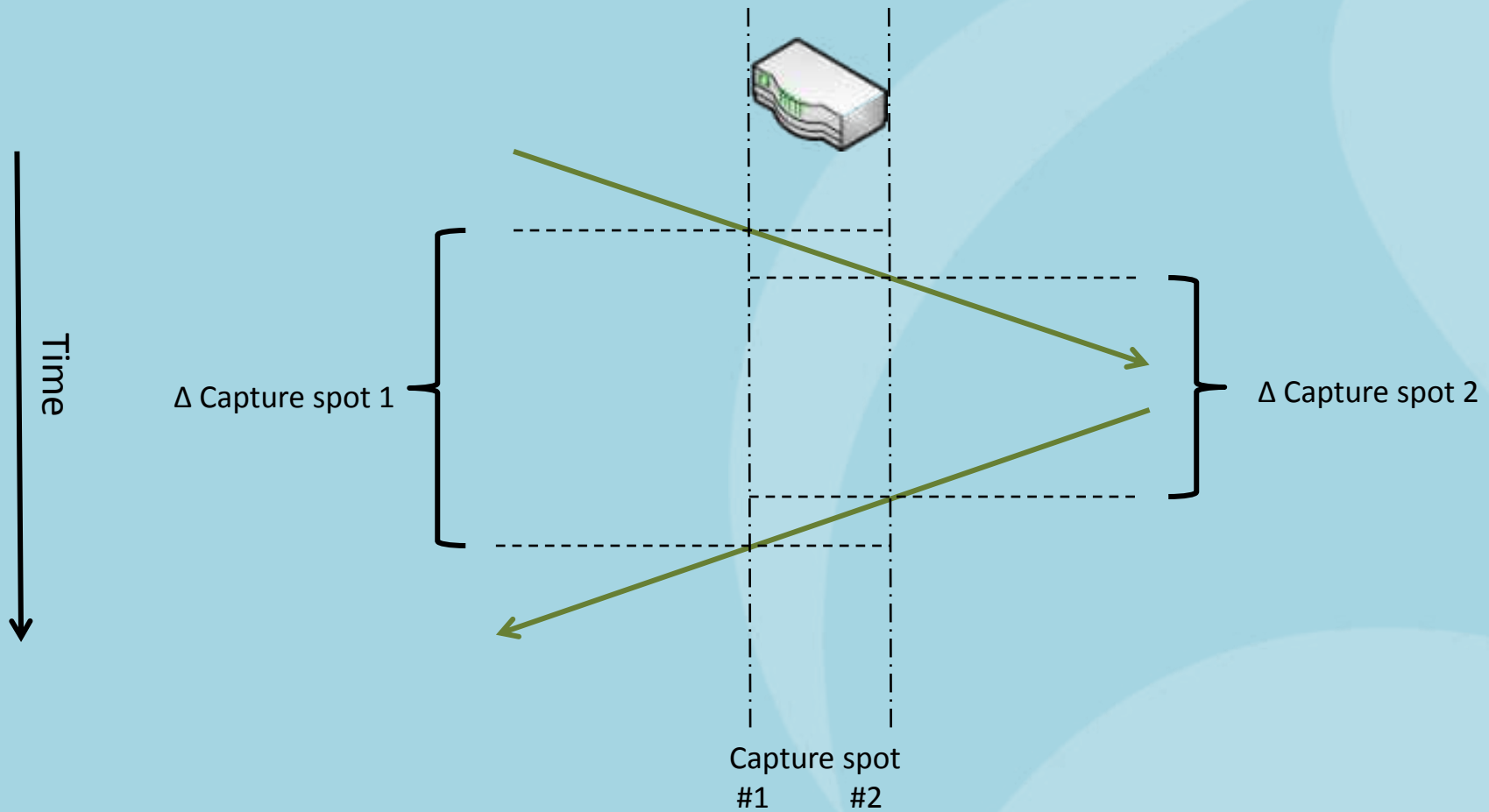




# Multipoint captures: latency



# Determining latency – single device



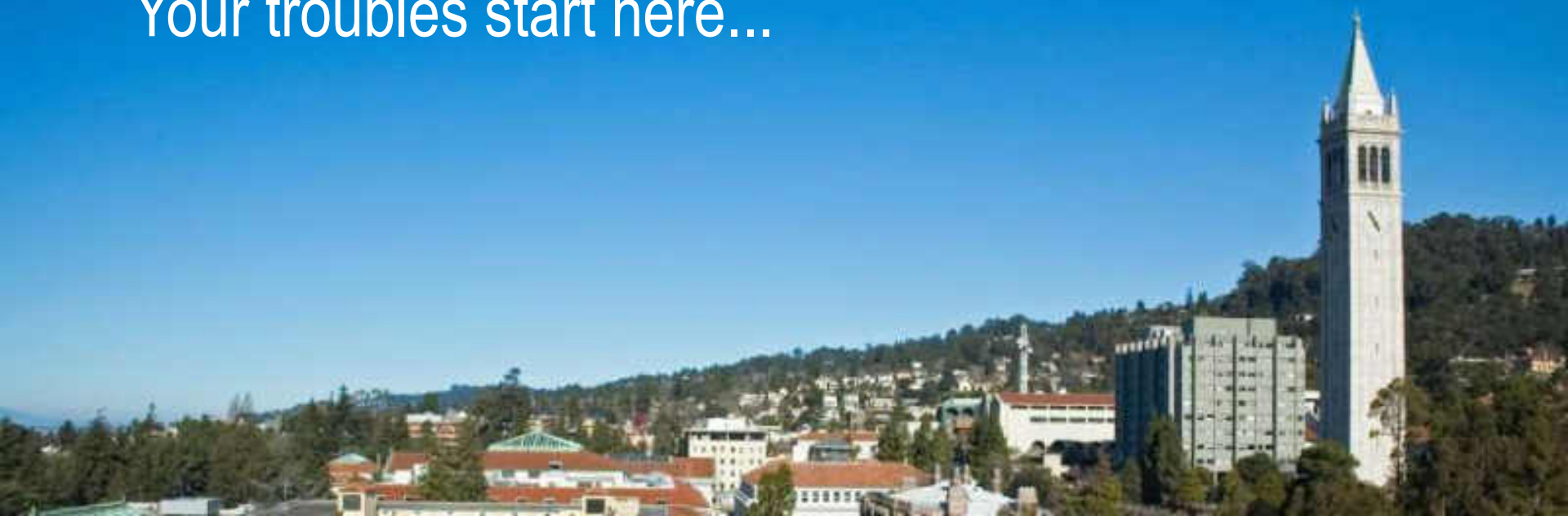


# SHARKFEST '13

Wireshark Developer and User Conference

## NAT, Proxy, Loadbalancer

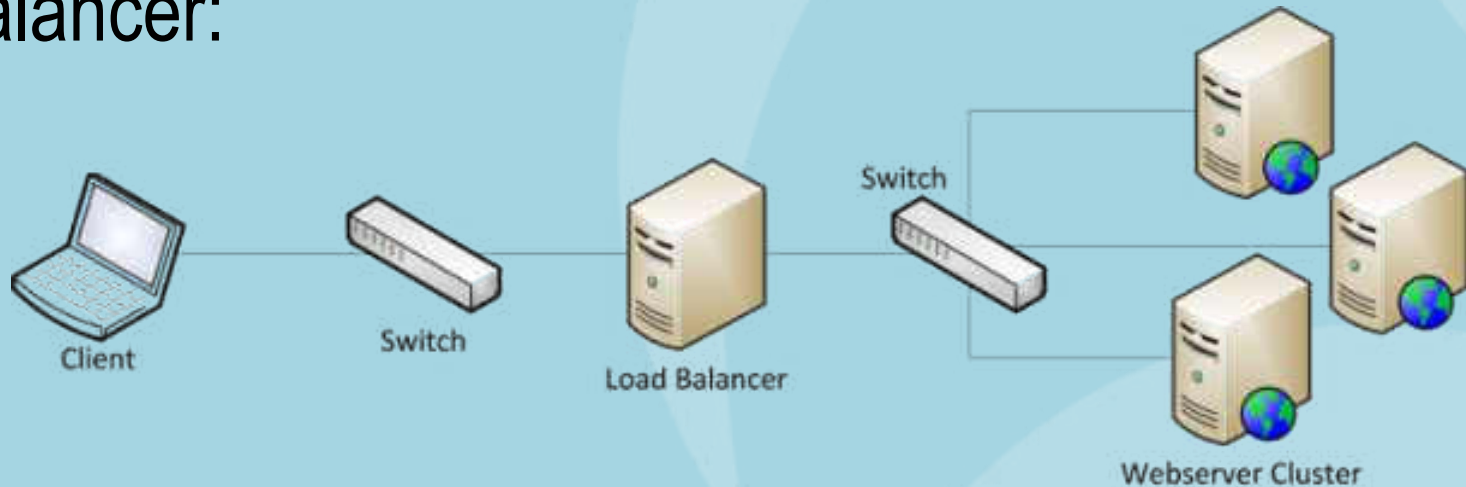
Your troubles start here...



# Troublemakers: Load balancers

---

- Load balancers distribute connections to multiple identical servers
- Allows scaling the available capacity
- Example with multi-tiered servers behind the load balancer:



# NAT gateways

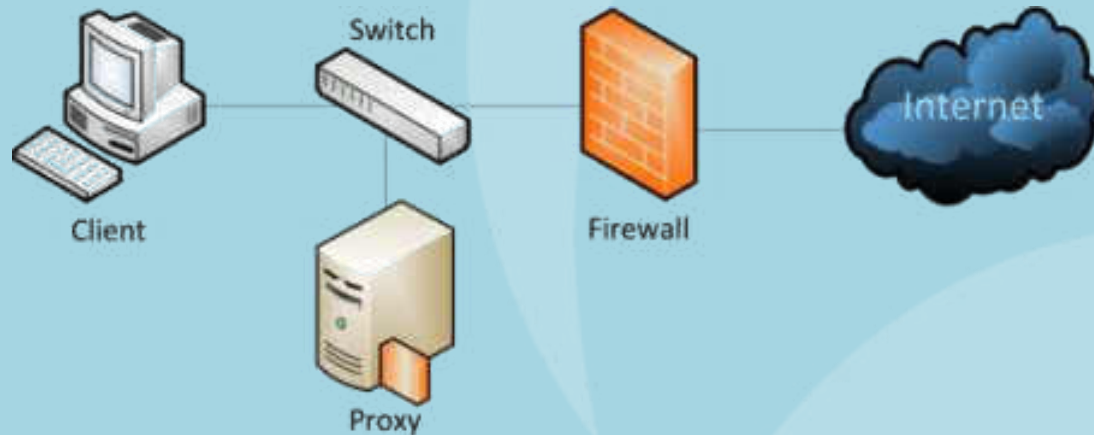
---

- NAT = Network Address Translation
  - Basically replaces network addresses found in packets back and forth
  - Usually relevant to layer 3, which means routers
- Typical NAT activity
  - Source NAT
  - Destination NAT

# Proxy servers

---

- Proxy servers separate different network and security zones
- Client requests are sent to the proxy
- The proxy fetches the requested content and delivers it to the client



# Proxy Server: Forwarded-For

- Some proxies insert the address of the client into the request headers:

```
Hypertext Transfer Protocol
GET / HTTP/1.0\r\n
Accept: text/html, application/xhtml+xml, */*\r\n
Accept-Language: de-DE\r\n
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; windows NT 6.1; WOW64; Trident/5.0)\r\n
Accept-Encoding: gzip, deflate\r\n
Host: www.google.de\r\n
[truncated] Cookie: PREF=ID=0de03f6f5ab5b026:U=acbc021047ffe581:FF=0:TM=1259593467:LM=1316903139
Via: 1.1 localhost (squid/3.0.STABLE8)\r\n
X-Forwarded-For: 192.168.124.100\r\n
Cache-Control: max-age=259200\r\n
```

- Best Practice:disable “X-Forwarded-For” for security reasons
  - X-Forwarded-For will show something like „unknown“
  - Turn back on for temporary troubleshooting tasks

# Transparent Proxies

- Intercept client communication without the need to configure the client to use the proxy
  - Using the “trace” method can reveal those proxies

The screenshot displays a network traffic analysis tool interface. The top section shows a list of captured packets with columns for No., Source, Destination, Protocol, and Summary. Packet 6 is highlighted in blue. Below the list, a detailed view of packet 6 is shown, including its frame size, capture details, and protocol layers (Ethernet II, Internet Protocol Version 4, Transmission Control Protocol, and Hypertext Transfer Protocol). The Hypertext Transfer Protocol section is expanded, showing the request details for a TRACE method to http://www.Flane.de/.

No.	Source	Destination	Protocol	Summary
1	192.168.124.100	192.168.122.252	TCP	11010 > 3128 [SYN, Seq=4127998770 Win=8192 [TCP CHECKSUM INCORRECT] Len=0 MSS=1460 VS-
2	192.168.122.252	192.168.124.100	TCP	3128 > 11010 [SYN, ACK] Seq=2470173712 Ack=4127998771 Win=5840 Len=0 MSS=1460 SACK_PER
3	192.168.124.100	192.168.122.252	TCP	11010 > 3128 [ACK] Seq=4127998771 Ack=2470173713 Win=65700 [TCP CHECKSUM INCORRECT] L
4	192.168.124.100	192.168.122.252	HTTP	TRACE http://www.Flane.de/ HTTP/1.1
5	192.168.122.252	192.168.124.100	TCP	3128 > 11010 [ACK] Seq=2470173713 Ack=4127998851 Win=5840 Len=0
6	192.168.122.252	192.168.124.100	HTTP	HTTP/1.0 200 OK (message/http)
7	192.168.122.252	192.168.124.100	TCP	3128 > 11010 [FIN, ACK] Seq=2470174132 Ack=4127998851 Win=5840 Len=0
8	192.168.124.100	192.168.122.252	TCP	11010 > 3128 [ACK] Seq=4127998851 Ack=2470174133 Win=65200 [TCP CHECKSUM INCORRECT] L
9	192.168.124.100	192.168.122.252	TCP	11010 > 3128 [FIN, ACK] Seq=4127998851 Ack=2470174133 Win=65300 [TCP CHECKSUM INCORRE
10	192.168.122.252	192.168.124.100	TCP	3128 > 11010 [ACK] Seq=2470174133 Ack=4127998852 Win=5840 Len=0

Frame 6: 473 bytes on wire (3784 bits), 473 bytes captured (3784 bits)

Ethernet II, Src: PcEngine\_18:64:fc (08:0d:b9:18:64:fc), Dst: Giga-Byt\_15:61:da (08:24:1d:15:61:da)

Internet Protocol Version 4, Src: 192.168.122.252 (192.168.122.252), Dst: 192.168.124.100 (192.168.124.100)

Transmission Control Protocol, Src Port: 3128 (3128), Dst Port: 11010 (11010), Seq: 2470173713, Ack: 4127998851, Len: 419

Hypertext Transfer Protocol

Media Type: message/http

TRACE / HTTP/1.0

User-Agent: Fiddler

Host: www.Flane.de

Via: 1.1 localhost (squid/3.0 STABLE0)

X-Forwarded-For: unknown

Cache-Control: max-age=259200

Connection: keep-alive





# SHARKFEST '13

Wireshark Developer and User Conference

**It's trace file clobberin' time!**

