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
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!
Determining latency
Multipoint captures: latency

[Diagram showing a client, multiple routers, and a server arranged along a timeline, illustrating multipoint captures and latency.]
Determining latency – single device

Time

Δ Capture spot 1

Capture spot #1

Capture spot #1

Δ Capture spot 2

Δ Capture spot 2

Capture spot #2

Capture spot #2
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](image)

  ```
  GET / HTTP/1.0

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

- 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
It’s trace file clobberin’ time!