





SharkFest'17 US

A Web-Based Approach to Enhance
Network Packet Capture & Decode
Analysis Techniques using the
Wireshark Command Line Tools

Ronald W. Henderson
CTO: UNIVERSAL, Technologies

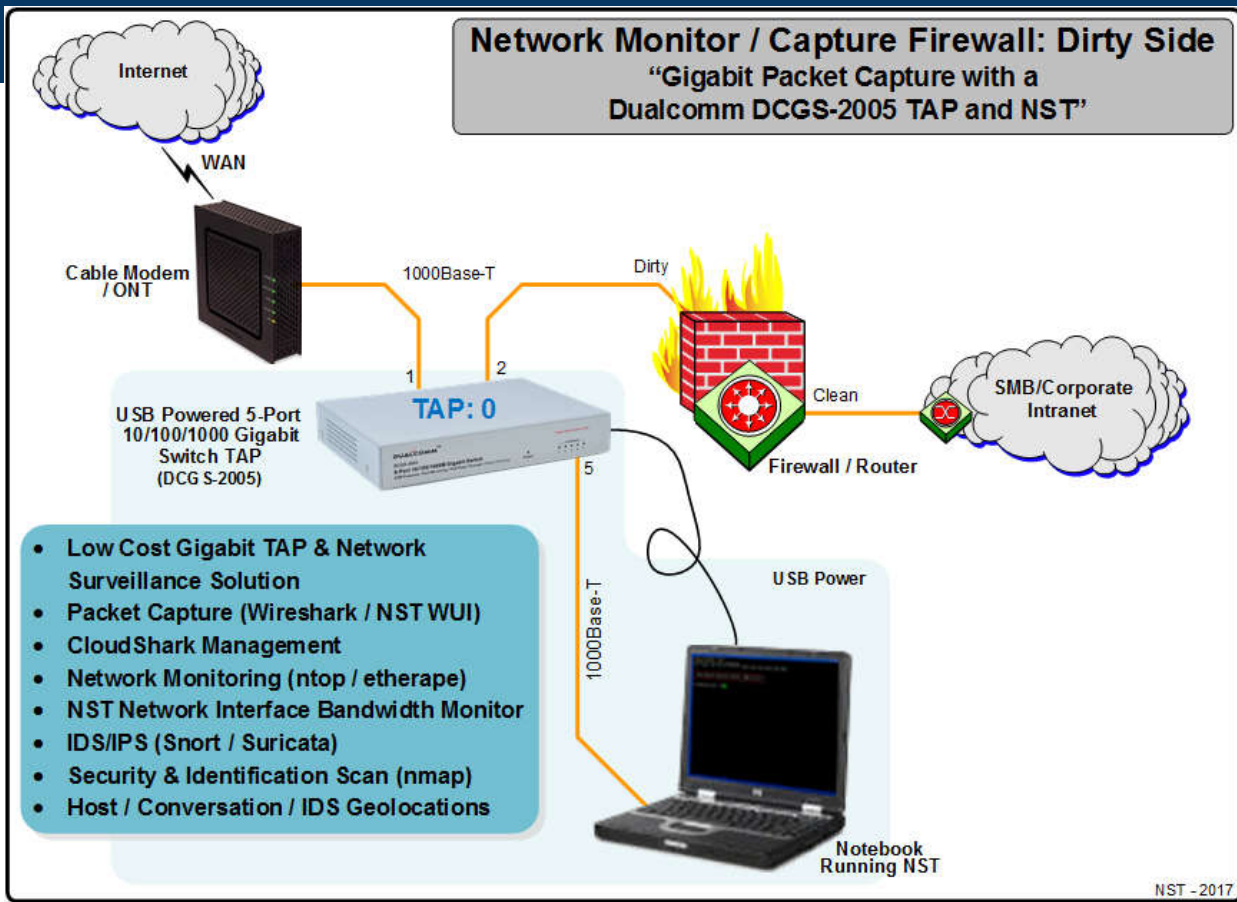
Presentation Agenda

- Using Wireshark with NST (Network Security Toolkit). 
- Single-Tap & Multi-TAP network packet capture integration.
- Web-Based packet capture using the “**dumpcap**” capture tool.
- Pending live capture status and display. 
- Web-Based capture decode using the “**tshark**” network traffic analysis tool.
- PSML & PDML: Specialized “tshark” decode output format displays.
- Packet Capture conversations & host geolocation. 
- Network Packet Capture Management and CloudShark integration & transfer.
- Ring Buffer Capture as a Service (RBCaaS) using “**dumpcap**”. 

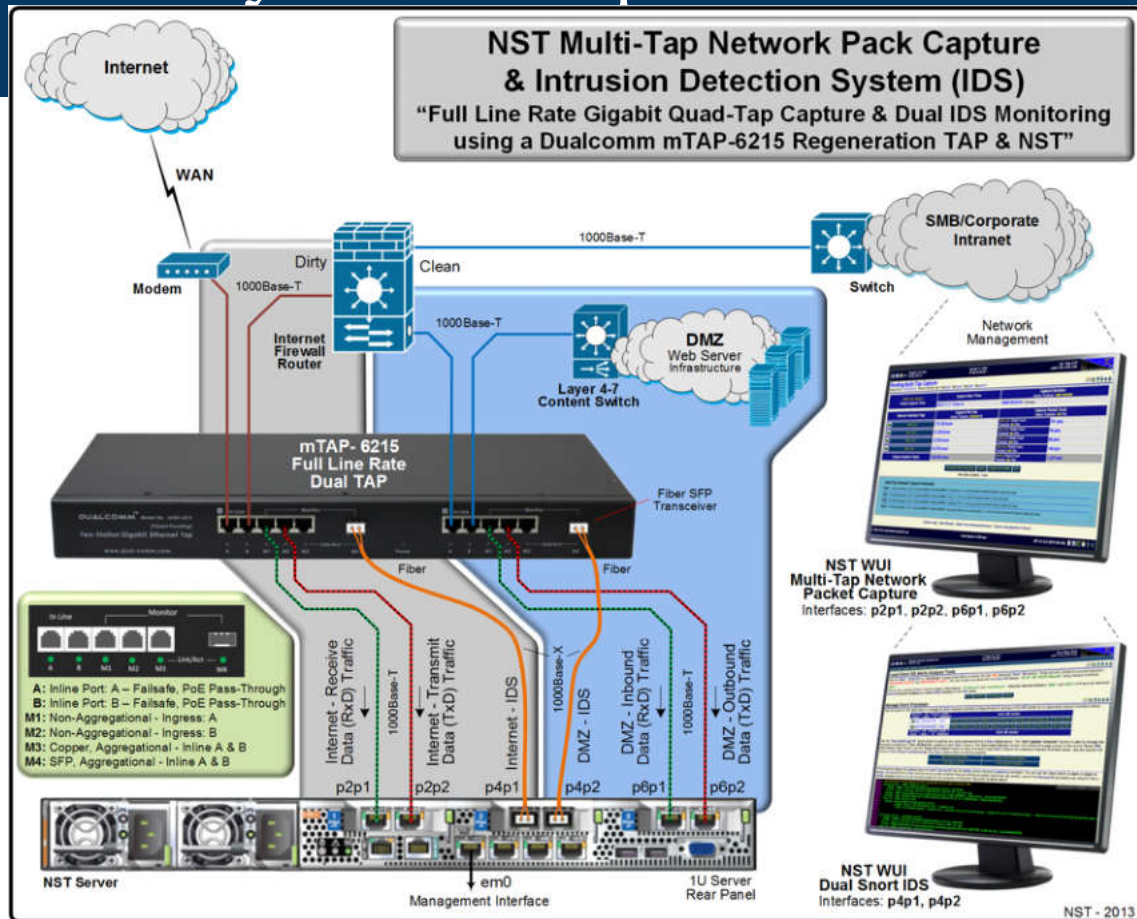
Using Wireshark with NST

- NST (Network Security Toolkit) – Development: 2003 to present.
- Toolkit design philosophy.
- Web-based frontend to Wireshark command line tools using the NST WUI (Web User Interface).
- NST WUI page navigation & display output controls.
- Open source tools integration (e.g., nDPI, p0f, PassiveDNS, nmap, etc...).
- Download: <https://sourceforge.net/projects/nst/files/>
- NST Pro: <http://www.networksecuritytoolkit.org/nstpro/>
- Reference: <http://wiki.networksecuritytoolkit.org/>

Network Visibility: Single-Tap Network Packet Capture



Network Visibility: Multi-Tap Network Packet Capture



Single-Tap Packet Capture Session using “dumpcap”

Network Interface & Capture directory selection



dumpcap options: threshold, capture filter & annotation



NST WUI dumpcap startup options: duration or date delay



Start dumpcap (dumpcap_capture.sh) & logger script: (dumpcap_log.sh)



Monitor a pending capture session



Capture termination: threshold or user based

Multi-Tap Packet Capture Session using “dumpcap”

Network Interface (Up to 4 TAPs) & Capture directory selection



TAP dumpcap options: threshold, capture filter & annotation



NST WUI dumpcap TAP startup options: duration or date delay



Start each dumpcap TAP (mtap_dumpcap_capture.sh) & logger script: (dumpcap_log.sh)

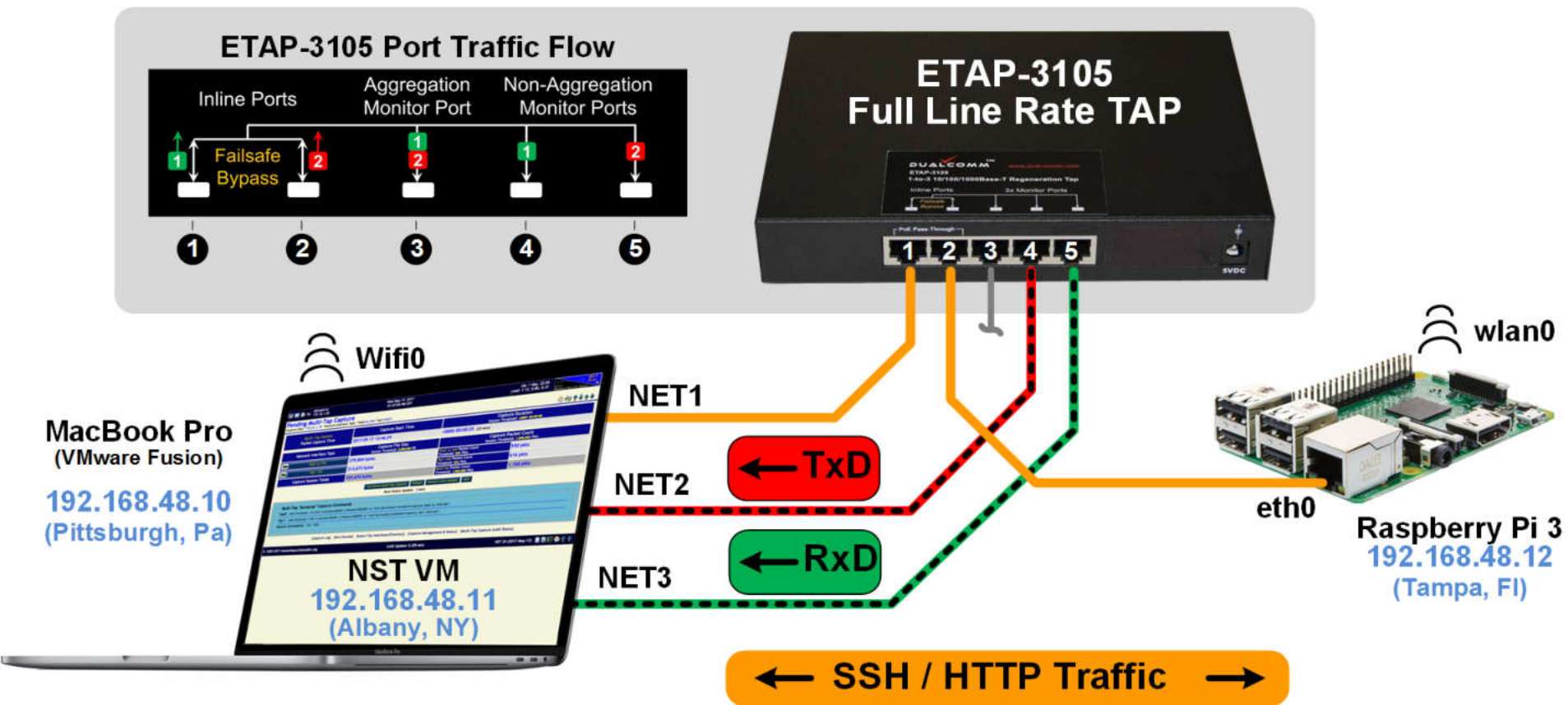


Monitor all pending dumpcap TAP capture sessions

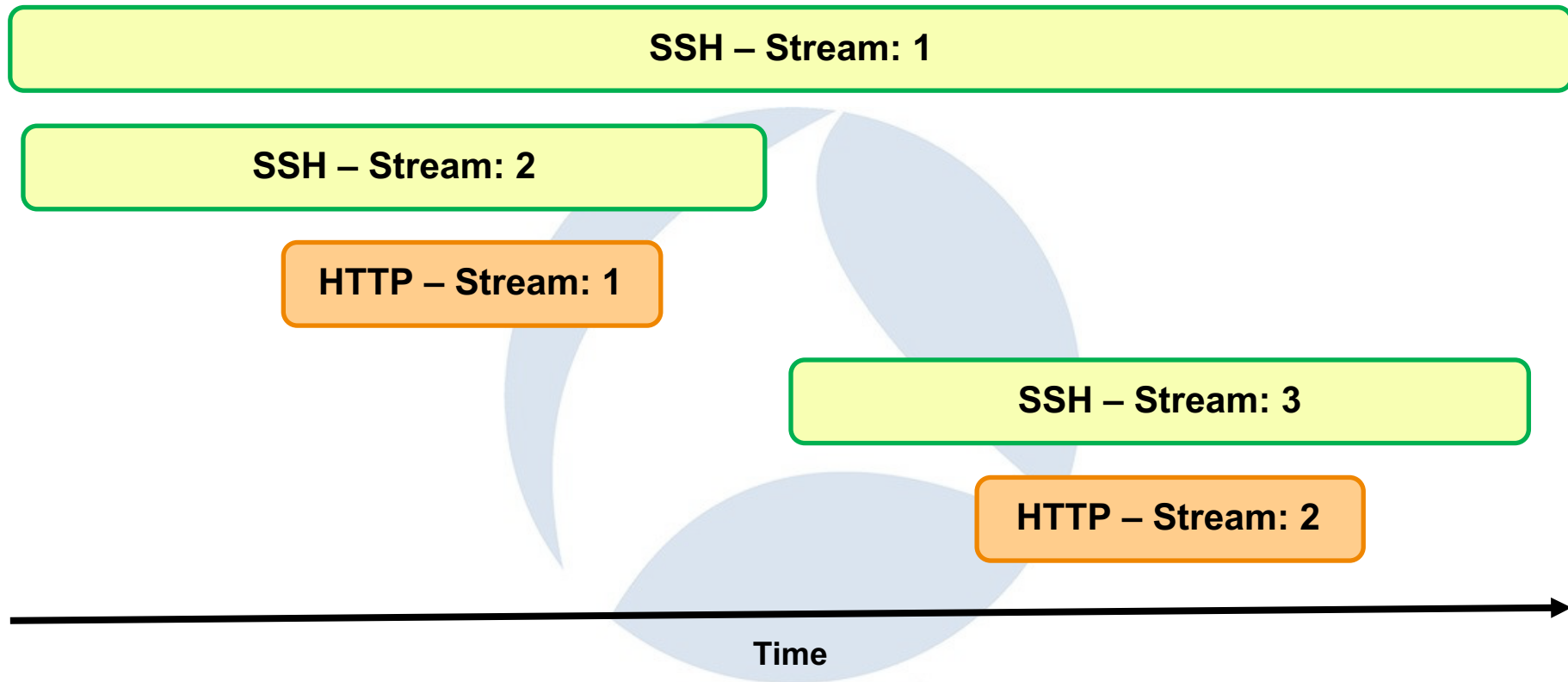


Capture termination: threshold or user based & Merge (mtap_merger.sh) - mergecap

Web-Based Live Capture: Network TAP Diagram



Web-Based Live Capture: TCP Network Traffic



Web-Based Decode Analysis using “tshark”

Protocol Analyzer selection: tshark



tshark options: display, statistical & decode as filters

tshark decode

**Capture
Summary**

**IPv4 / IPv6
Traffic (L3)**

**Open Source
Analysis Tools**

**View raw
PSML & PDML**

**Network
Packet Details**

**TCP/IP, UDP &
ICMP Traffic**

**Capture Transfer
CloudShark**

**Poor Man's
Wireshark**

**Ethernet
Traffic (L2)**

**NST Tools
Integration**

**Geolocation
Reports**

Web-Based Live Decode Analysis using “tshark”

Multi-Tap Decoded Capture (tshark):

(Capture Host: "172.31.1.18" Network Interface Taps: "Tap0:ut_fw0 Tap1:fw0")



The protocol analyzer: "tshark" was used to decode the multi-tap network packet capture file: "/var/nst/wuiout/wireshark/capture_mtap.cap" using the following command:

Multi-Tap Decode Capture Command: "tshark"

```
/sbin/tshark -n -t r -r "/var/nst/wuiout/wireshark/capture_mtap.cap" -Y 'frame.number >= 1 && frame.number <= 20';
```

Remove Decode Display Output

[Capture Summary] [New Decode] [Launch Wireshark] [Capture Transfer] [Capture Log]

[Select Tap Interfaces/Directory] [Start Multi-Tap Capture] [Capture Management & Status]

```
1 0.000000000 192.241.187.236 → 24.97.150.194 TLSv1.2 Application Data
2 0.000106713 192.241.187.236 → 172.16.1.73 TLSv1.2 Application Data
3 0.000491972 104.43.140.223 → 24.97.150.194 TLSv1.2 Application Data
4 0.000514773 104.43.140.223 → 172.16.1.70 TLSv1.2 Application Data
5 0.000742392 172.16.1.73 → 192.241.187.236 TLSv1.2 Application Data
6 0.000987162 24.97.150.194 → 192.241.187.236 TLSv1.2 Application Data
7 0.019741682 192.241.187.236 → 172.16.1.73 TCP 443 → 57694 [ACK] Seq=32 Ack=36 Win=11040 Len=0
8 0.019754862 192.241.187.236 → 24.97.150.194 TCP 443 → 57694 [ACK] Seq=32 Ack=36 Win=11040 Len=0
9 0.182145751 46.229.168.74 → 24.97.150.195 HTTP GET /index.php?oldid=2352&printable=yes&title=Overview HTTP/1.1
10 0.182651560 24.97.150.195 → 46.229.168.74 TCP 80 → 43770 [ACK] Seq=1 Ack=261 Win=235 Len=0 TSval=631311711 TSecr=8564884
11 0.208931870 172.16.1.70 → 104.43.140.223 TCP 58351 → 443 [ACK] Seq=1 Ack=726 Win=255 Len=0
12 0.209115788 24.97.150.194 → 104.43.140.223 TCP 58351 → 443 [ACK] Seq=1 Ack=726 Win=255 Len=0
13 0.465948062 fe80::5110:36c3:31be:6554 → ff02::1:2 DHCPv6 Solicit XID: 0xec0e0f CID: 000100011e4eec870010e095ad75
14 0.476668281 24.97.150.195 → 46.229.168.74 HTTP HTTP/1.1 200 OK (text/html)
15 0.476916570 24.97.150.195 → 46.229.168.74 HTTP Continuation
16 0.476918234 24.97.150.195 → 46.229.168.74 HTTP Continuation
17 0.477166405 24.97.150.195 → 46.229.168.74 HTTP Continuation
18 0.477168327 24.97.150.195 → 46.229.168.74 TCP 80 → 43770 [FIN, ACK] Seq=4954 Ack=261 Win=235 Len=0 TSval=631312005 TSecr
19 0.504618750 172.16.1.4 → 172.16.1.255 UDP 47557 → 3052 Len=693
20 0.518197263 46.229.168.74 → 24.97.150.195 TCP 43770 → 80 [ACK] Seq=261 Ack=1449 Win=251 Len=0 TSval=856488470 TSecr=6313
```

Ring Buffer Capture as a Service (RBCaaS) using “dumpcap”

- **Implemented as a systemd service.**
- **Supports both the “dumpcap” and the “netsniff-ng” capture engines.**
- **Supports concurrent ring-buffer capture sessions.**
- **Each ring-buffer capture session has its own capture filters.**
- **Supports multiple types of capture file merging using “mergecap”.**
- **Integrates into the NST WUI Capture Management facility.**
- **Supports live session snapshots (Performance: Uses hard links where possible).**
- **Available command line session status and controls.**

Ring Buffer Capture as a Service (RBCaaS) Life Cycle

Install one or more Ring Buffer capture sessions environment (“/etc/nstringbufcap.d”)

Specify: Network Interface, Capture Engine, Max File Size, Max Number of Files, Max File Duration, Max File Count, Snap Length, Capture Filter, Ring Buffer and Snap Directories.



Start up one or more Ring Buffer capture sessions as a systemd service.



Monitor the Ring Buffer capture sessions: status & listing.

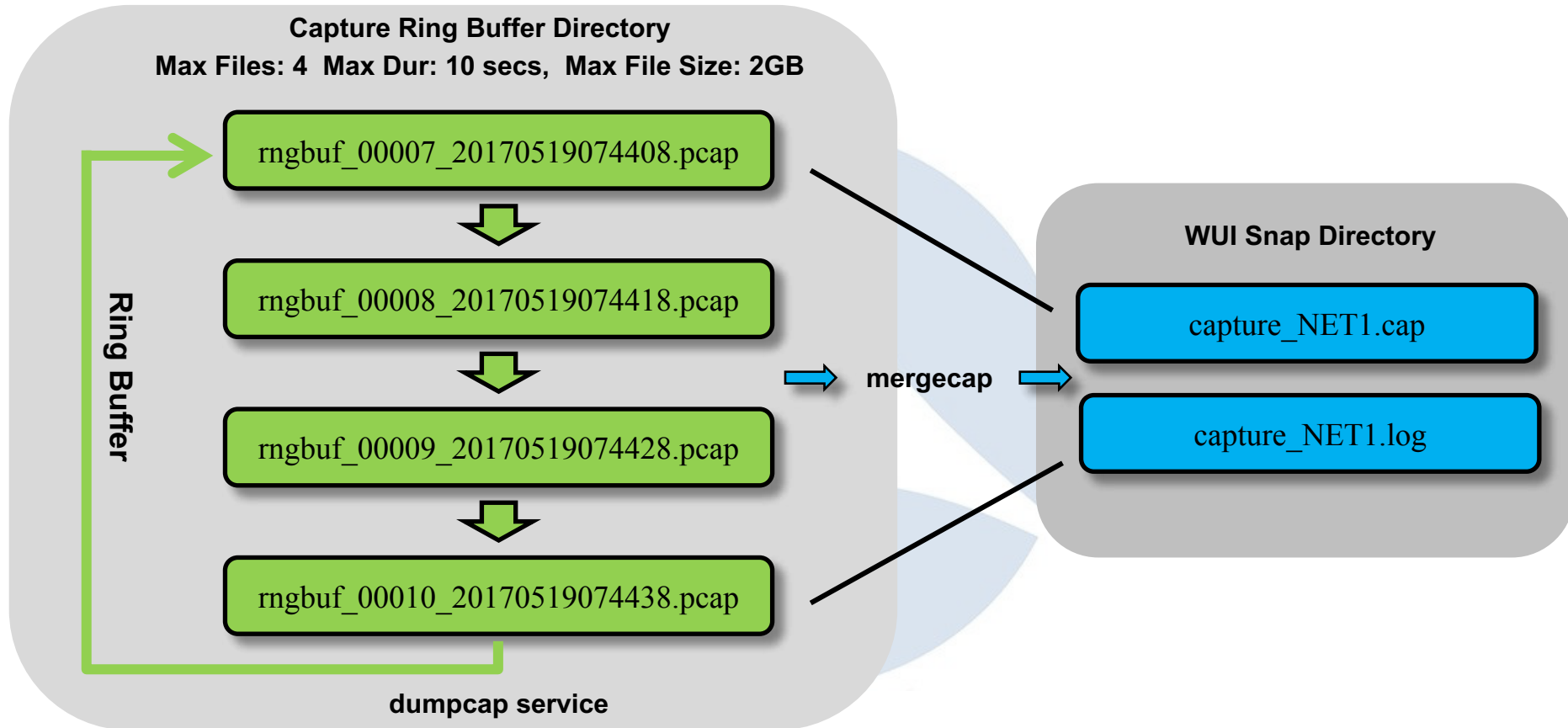


Operation: Snapshots & Merging (Could be based on a trigger).



Ring Buffer session termination.

Ring Buffer Capture as a Service (RBCaaS): Snapshot



Live Ring Buffer Capture as a Service (RBCaaS) Demo

The screenshot displays a Linux desktop environment with a window titled "ringbuf-dumpcap@rpi3net.service". The window is split into two panes. The left pane shows the service's status and configuration details:

- Documentation: `dumpcap(1)`
- Active: **active (running)**
- Loaded: loaded
- RPM: `nstringbufcap`
- Fragment: `/usr/lib/systemd/system/ringbuf-dumpcap@.service`
- Environment: `/etc/nstringbufcap`
- Unit File: disabled
- Preset: disabled
- Main PID: **3318**
- Status: No status information
- Requires: `sysinit.target`
- After: `basic.target`, `system-ringbuf-dumpcap@.service`
- Before: `shutdown.target`
- Conflicts: `shutdown.target`

The right pane is a "NST Shell Console" window showing the following commands and their outputs:

```
00001: echo -e '\n\x1b[34m*** Systemd Status ***\n';
/usr/bin/sudo LANG=en_US.UTF-8 TERM=ansi /usr/bin/stdoutisatty /bin/systemctl status --no-pager --full 'ringbuf-dumpcap@rpi3net.service'
echo -e '\n\x1b[34m*** Depends On (Occurs After) ***\n';
/usr/bin/sudo LANG=en_US.UTF-8 TERM=ansi /usr/bin/stdoutisatty /bin/systemctl list-dependencies --after --no-pager --full 'ringbuf-dumpcap@rpi3net.service'
echo -e '\n\x1b[34m*** Depends Before (Occurs Before) ***\n';
/usr/bin/sudo LANG=en_US.UTF-8 TERM=ansi /usr/bin/stdoutisatty /bin/systemctl list-dependencies --before --no-pager --full 'ringbuf-dumpcap@rpi3net.service'
echo -e '\n\x1b[34m*** Systemd Analyze Critical Path ***\n';
/usr/bin/sudo LANG=en_US.UTF-8 /bin/systemd-analyze critical-chain 'ringbuf-dumpcap@rpi3net.service' 2>&1;
( [ -f /etc/sysconfig/ringbuf-dumpcap@rpi3net ] && echo -e '\n\x1b[34m*** /etc/sysconfig/ringbuf-dumpcap@rpi3net ***\n';
/bin/echo -e '\n\x1b[34m*** Related Processes ***\n';
/bin/ps -f --ppid 3318 3318 2>&1;
/bin/echo -e '\n\x1b[34m*** Fragment Path ***\n';
/bin/cat /usr/lib/systemd/system/ringbuf-dumpcap@.service;

00002:
00003: *** Systemd Status ***
00004:
00005: * ringbuf-dumpcap@rpi3net.service - NST Ring Buffer dumpcap Service on rpi3net
00006:   Loaded: loaded (/usr/lib/systemd/system/ringbuf-dumpcap@.service; disabled; vendor preset: disabled)
00007:   Active: active (running) since Fri 2017-05-19 07:45:36 EDT; 1min 8s ago
00008:     Docs: man:dumpcap(1)
00009:   Main PID: 3318 (dumpcap)
00010:     Tasks: 1 (limit: 512)
00011:    CGroup: /system.slice/system-ringbuf\x2ddumpcap.slice/ringbuf-dumpcap@rpi3net.service
00012:            └─3318 /usr/sbin/dumpcap -q -i NET1 -b filesize:2000 -b duration:10 -b files:6 -w /opt/ringbuffer/
00013:
00014: May 19 07:45:36 nst24-macbook systemd[1]: Started NST Ring Buffer dumpcap Service on rpi3net.
00015: May 19 07:45:37 nst24-macbook dumpcap[3318]: Capturing on 'NET1'
00016: May 19 07:45:37 nst24-macbook dumpcap[3318]: File: /opt/ringbuffer/ringbuf_00001_20170519074537.dump
```