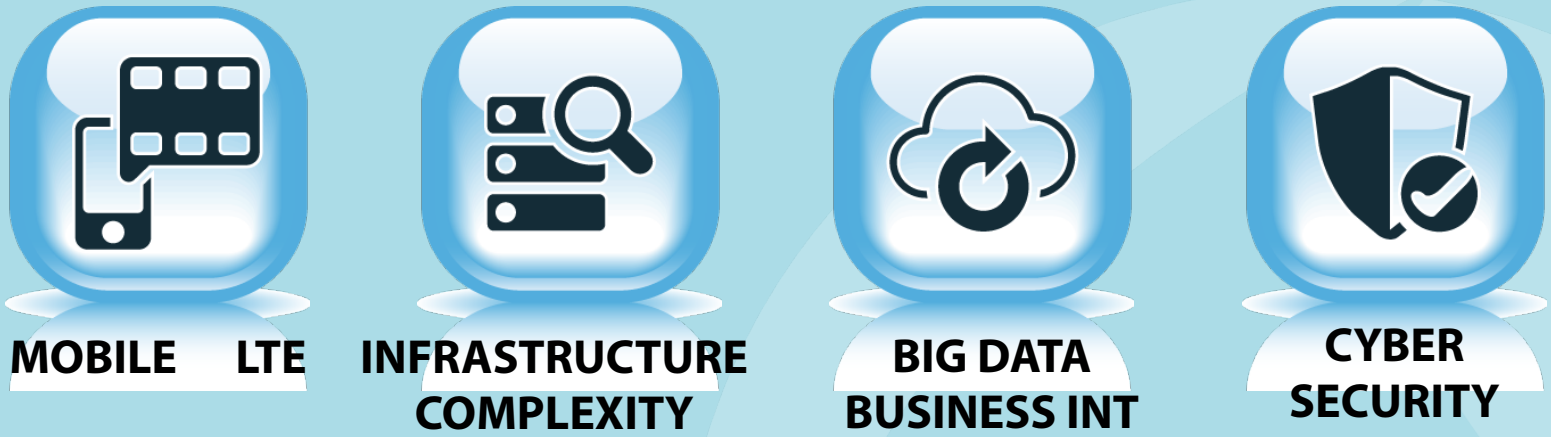
SHARKFEST '13

Wireshark Developer and User Conference

Packet Optimization & Visibility with Wireshark and PCAPs

Gordon Beith
Director of Product Management
VSS Monitoring

Market Trends - Innovation



- In a hyper connected world, one must master the trends of cloud, virtualization, SDN, media, mobile, business intelligence while overcoming security threats.
- This makes Network, Application, and Security monitoring and analysis tools critical to have…

Monitoring & Analysis Challenges (1)

- Access to all key points in the network
  - Is a tool (e.g. Wireshark) instance at every point realistic?
    - Costly to install and maintain
    - Difficult to manage across different locations
    - Won't get full picture/session, e.g. asymmetrical routing
    - High speeds, e.g. 40G, & 100G, probably not accessible
- Visibility to all required packets from the network
  - Can a single tool instance handle all traffic?
    - Total bandwidth is too much for NICs and processors
    - Each instance of a tool probably doesn't need to see all traffic
  - Can a specific tool understand all protocols?
    - Tunneling and virtualization protocols often not supported nor interesting
    - Encrypted traffic content typically not accessible to tools

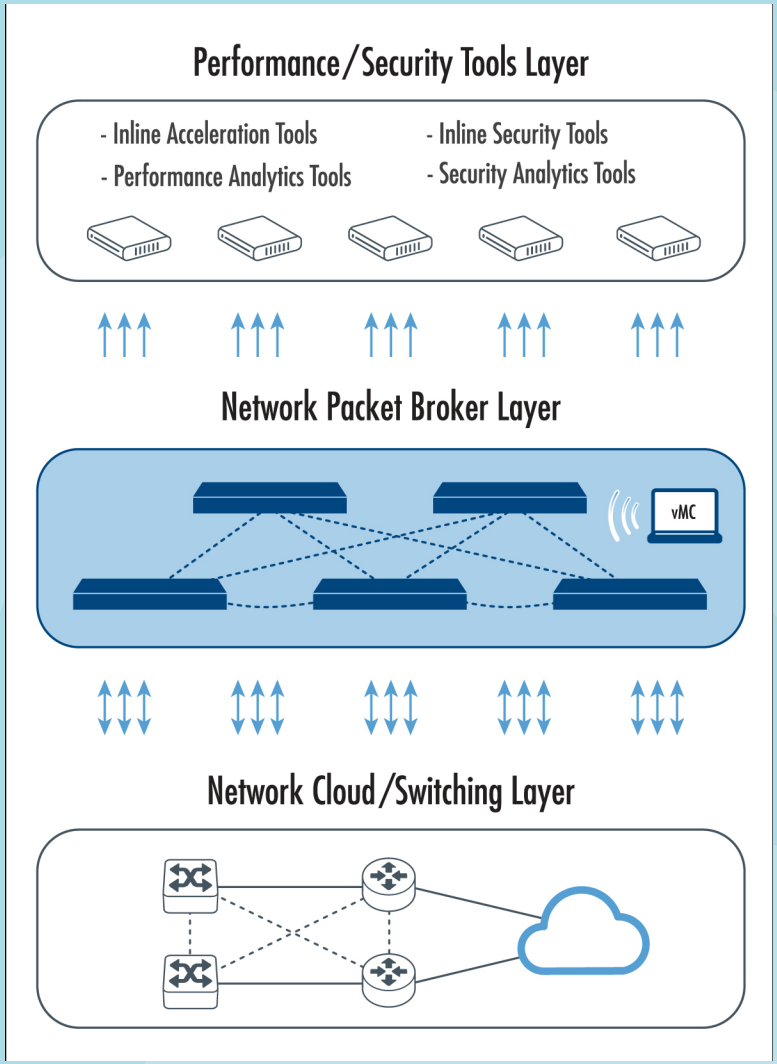
Monitoring & Analysis Challenges (2)

- Scalability of Tools
  - How can a tool scale to meet increasing traffic and number of networks?
    - Next to impossible without faster processors, bigger disks, and more/higher-end NICs
- Impacts of Bursty Traffic
  - How can effects of microbursts be mitigated?
    - Needs to be done in the aggregation hardware to avoid packet loss
- Governance and Legal Compliance
  - In what ways can tools comply with HIPPA, LI, PCI, SOX, etc.?
    - Very difficult without programmable hardware pre-processing in tool

➢ Employ Network Packet Brokers!

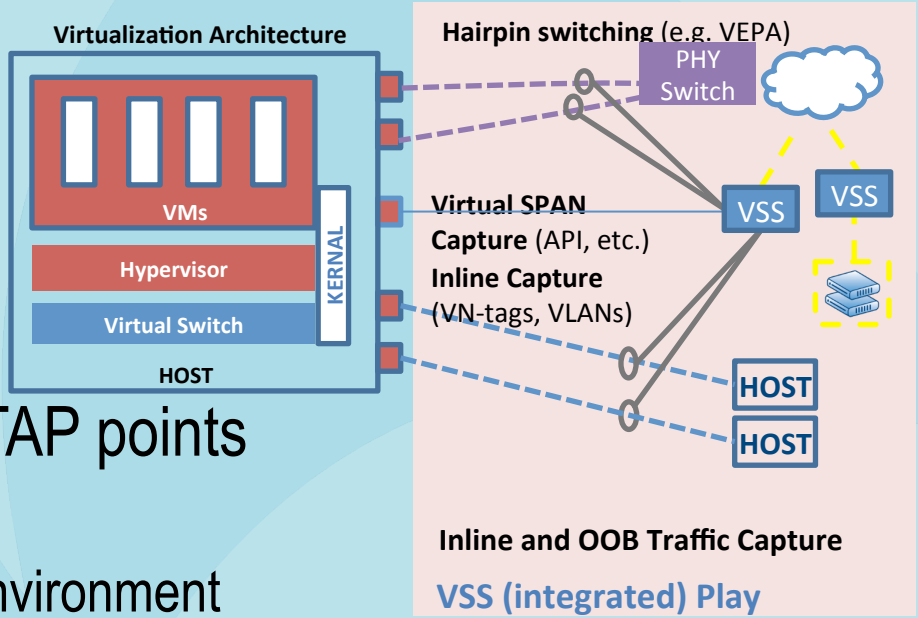
What are Network Packet Brokers?

- A newer type of network appliance
- Sits between network and intelligence tools
- Captures network traffic at line rate
- Grooms packets ready for network intelligence tools
- Forwards packets to network intelligence tools



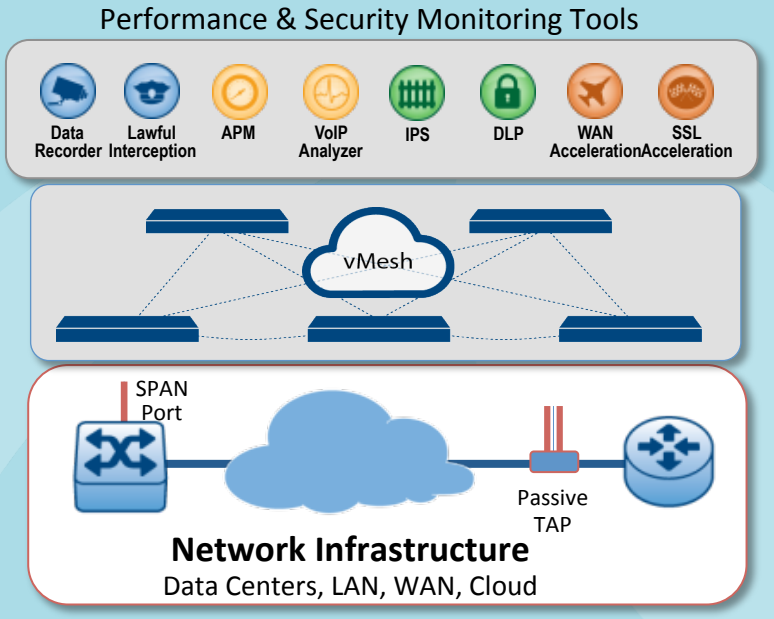
Addressing the Challenges with NPBs (1)

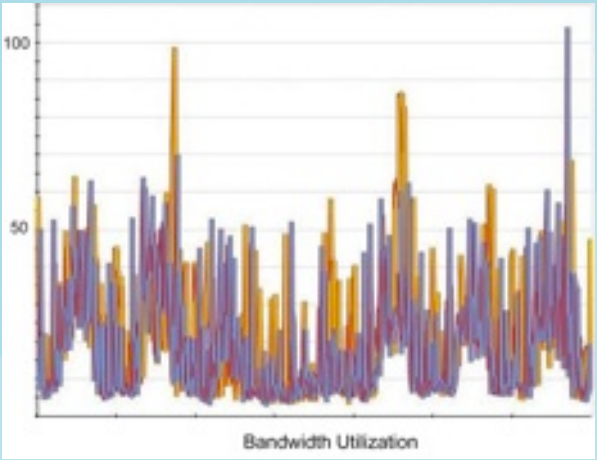
- Network & Traffic Access
  - Deploy network TAP appliances wherever possible
    - Remove Mirror/SPAN port contention
    - Ensure reliability of data access
    - Use Mirror/SPAN ports on switches/routers if no other choice
  - Employ virtual Mirror/SPAN/TAP points within virtual environments
    - Direct traffic out from virtual environment



Addressing the Challenges with NPBs (2)

- Traffic Delivery Optimization
  - Deploy Network Packet Brokers (NPBs) as a complete system
    - Either as a second tier or with TAPs integrated
    - Selective aggregation from access towards monitoring tools
    - Filter traffic based on layers 2 to 7, with/without tunneling
    - Balance traffic across multiple tools maintaining session integrity
    - High availability & redundancy
  - Include Microburst Detection & Mitigation
    - Visibility to granular traffic profile for network capacity planning
    - Ensure delivery of all monitored packets despite microbursts





Addressing the Challenges with NPBs (3)

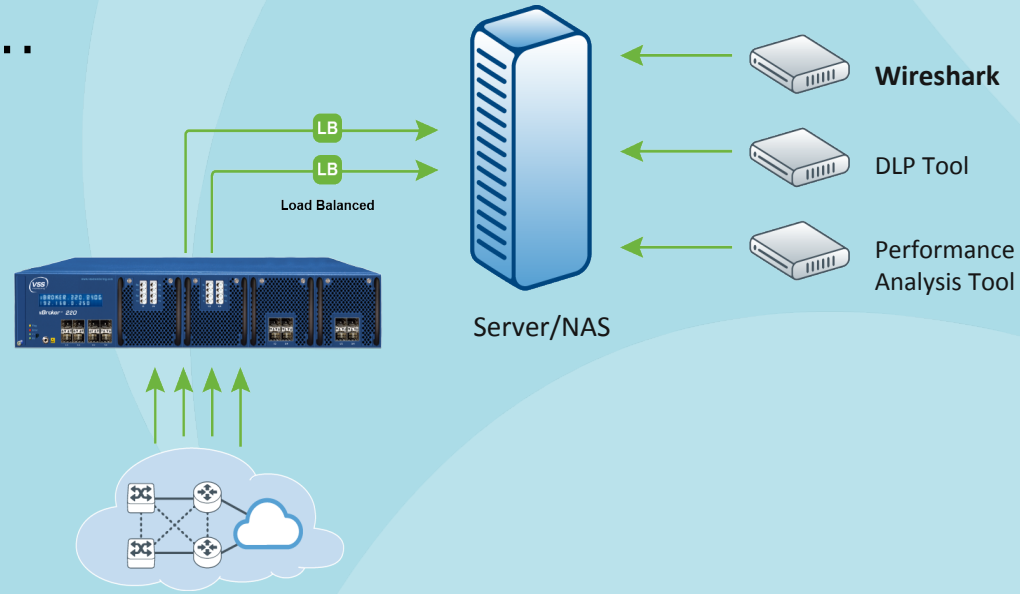
- Packet Optimization (1)
  - Port and Time Stamping
    - Visibility to packet's source link and time of occurrence
    - Accuracy is important
  - Protocol De-encapsulation/Stripping
    - Remove unwanted/unsupported protocol headers from monitored packets, e.g. GTP tunnels, MPLS labels, VLAN tags, VN-tags
  - Conditional Slicing
    - Remove unwanted/undesirable data from specific packets
    - Support regulatory and legal compliance

Addressing the Challenges with NPBs (4)

- Packet Optimization (2)
  - De-duplication
    - Minimize bandwidth required for delivery
    - Improve on tools' efficiency and accuracy
  - Fragment Reassembly
    - Facilitate flow/session-based balancing and filtering of traffic
    - Improve tool efficiency and accuracy
  - SSL Decryption
    - Visibility into encrypted traffic

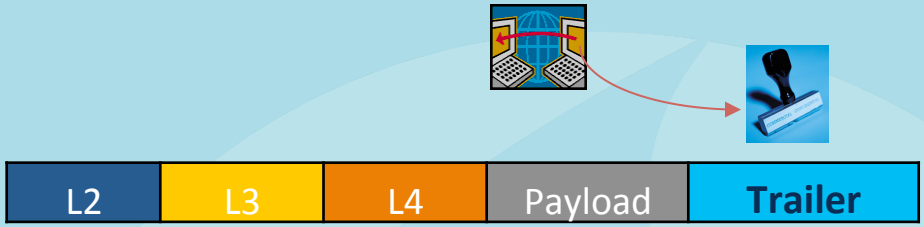
Addressing the Challenges with NPBs (5)

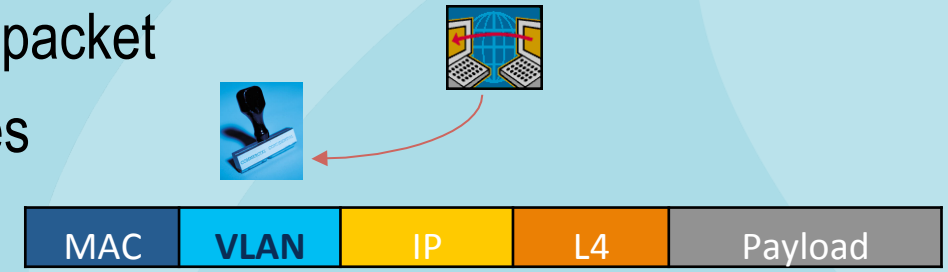
- PCAP Creation Optimization
  - Encapsulate packets in PCAP file format
    - Include metadata (e.g. accurate timestamps and source port) in PCAPng format
  - Forward to network-attached storage device for analyzer tools to access, or…
    stream to tool

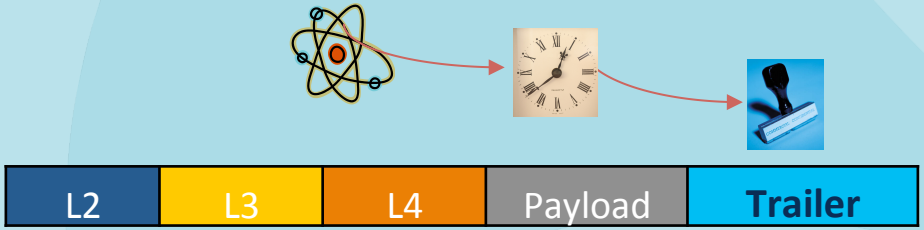


Interpreting Results in Wireshark (1)

- Port Stamping
  - End-of-packet
    - Trailer after and of IP packet
    - Minimizes added bytes
  - Start-of-packet
    - VLAN tag alternative
- Time Stamping
  - End-of-packet
    - Trailer after and of IP packet
    - Minimizes added bytes







Interpreting Results in Wireshark (2)

- Time and Port Stamps in Trailer example

```
⊞ Frame 1: 133 bytes on wire (1064 bits), 133 bytes captured (1064 bits)
⊞ Ethernet II, Src: Performa_00:00:02 (00:10:94:00:00:02), Dst: Xerox_00:00:01 (00:00:01:00:00:01)
⊞ Internet Protocol Version 4, Src: 192.85.1.2 (192.85.1.2), Dst: 192.0.0.1 (192.0.0.1)
⊞ Data (86 bytes)
⊟ VSS-Monitoring ethernet trailer, Timestamp: 11:54:36.773757586, Source Port: 2
    Time Stamp: Apr  9, 2013 11:54:36.773757586 Pacific Daylight Time
    Clock Source: Not Synced (0)
    Src Port: 2

0060  00 00 00 00 e1 5f e4 e9  43 89 05 e8 32 a8 c8 19   ....._.. C...2...
0070  36 64 8d e7 ff b7 31 7f  51 64 63 ec 2e 1e 9a 92   6d....1. Qdc.....
0080  02 fa 8d 64 33                                     ...d3
```

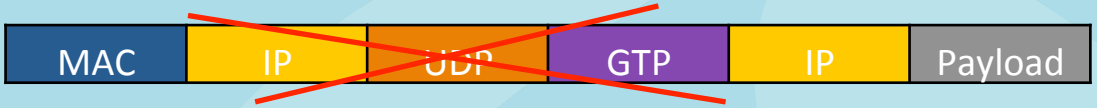
- Port Stamp in VLAN Tag example

```
⊞ Frame 2: 133 bytes on wire (1064 bits), 96 bytes captured (768 bits)
⊟ Ethernet II, Src: Performa_00:00:02 (00:10:94:00:00:02), Dst: Xerox_00:00:01 (00:00:01:00:00:01)
  ⊞ Destination: Xerox_00:00:01 (00:00:01:00:00:01)
  ⊞ Source: Performa_00:00:02 (00:10:94:00:00:02)
    Type: 802.1Q Virtual LAN (0x8100)
⊟ 802.1Q Virtual LAN, PRI: 0, CFI: 0, ID: 100
    000. .... .... .... = Priority: Best Effort (default) (0)
    ...0 .... .... .... = CFI: Canonical (0)
    .... 0000 0110 0100 = ID: 100
    Type: IP (0x0800)
⊞ Internet Protocol Version 4, Src: 192.85.1.2 (192.85.1.2), Dst: 192.0.0.1 (192.0.0.1)
⊞ Data (58 bytes)

0000  00 00 01 00 00 01 00 10  94 00 00 02 81 00 00 64   ........ .......d
0010  08 00 45 00 00 6a 00 00  00 00 ff fd 39 3e c0 55   ..E..j.. ....9>.U
0020  01 02 c0 00 00 01 00 00  00 00 00 00 00 00 00 00   ........ ........
```

Interpreting Results in Wireshark (3)

- Protocol Stripping/De-encapsulation
  - GTP Tunneling



    - Remove all 3 headers
  - MPLS Tunneling

    - Remove all labels and/or encapsulation

  - VLAN & VN Tagging

    - Selectively remove 1, 2, or all tags
    - Remove all or only specific TPID tags

Interpreting Results in Wireshark (4)

- GTP Tunneling example
  - With GTP Encapsulation

```
⊞ Frame 1: 524 bytes on wire (4192 bits), 524 bytes captured (4192 bits) on interface 0
⊞ Ethernet II, Src: Cisco_2b:4d:40 (00:1a:30:2b:4d:40), Dst: NokiaSie_29:a6:9a (00:40:43:29:a6:9a)
⊞ Internet Protocol Version 4, Src: 203.116.42.149 (203.116.42.149), Dst: 10.218.200.1 (10.218.200.1)
⊞ User Datagram Protocol, Src Port: gtp-user (2152), Dst Port: gtp-user (2152)
⊟ GPRS Tunneling Protocol
  ⊞ Flags: 0x30
    Message Type: T-PDU (0xff)
    Length: 470
    TEID: 0x00000263
  T-PDU Data 470 bytes
⊞ Internet Protocol Version 4, Src: 89.238.146.158 (89.238.146.158), Dst: 10.13.42.205 (10.13.42.205)
⊞ Transmission Control Protocol, Src Port: http (80), Dst Port: http (80), Seq: 1, Ack: 1, Len: 430
```

  - With GTP Encapsulation Removed

```
⊞ Frame 1: 488 bytes on wire (3904 bits), 488 bytes captured (3904 bits) on interface 0
⊞ Ethernet II, Src: Cisco_2b:4d:40 (00:1a:30:2b:4d:40), Dst: NokiaSie_29:a6:9a (00:40:43:29:a6:9a)
⊞ Internet Protocol Version 4, Src: 89.238.146.158 (89.238.146.158), Dst: 10.13.42.205 (10.13.42.205)
⊞ Transmission Control Protocol, Src Port: http (80), Dst Port: http (80), Seq: 1, Ack: 1, Len: 430
```

Interpreting Results in Wireshark (5)

- MPLS Tunneling example
  - With MPLS Labels

```
⊞ Frame 3: 118 bytes on wire (944 bits), 118 bytes captured (944 bits) on interface 0
⊟ Ethernet II, Src: Cisco_05:28:38 (00:30:96:05:28:38), Dst: Cisco_e6:fc:39 (00:30:96:e6:fc:39)
  ⊞ Destination: Cisco_e6:fc:39 (00:30:96:e6:fc:39)
  ⊞ Source: Cisco_05:28:38 (00:30:96:05:28:38)
    Type: MPLS label switched packet (0x8847)
⊟ MultiProtocol Label Switching Header, Label: 245, Exp: 0, S: 0, TTL: 64
    0000 0000 0000 1111 0101 .... .... .... = MPLS Label: 245
    .... .... .... .... .... 000. .... .... = MPLS Experimental Bits: 0
    .... .... .... .... .... ...0 .... .... = MPLS Bottom Of Label Stack: 0
    .... .... .... .... .... .... 0100 0000 = MPLS TTL: 64
⊞ MultiProtocol Label Switching Header, Label: 227, Exp: 0, S: 1, TTL: 64
⊞ Internet Protocol Version 4, Src: 10.31.85.1 (10.31.85.1), Dst: 10.34.85.1 (10.34.85.1)
⊞ Transmission Control Protocol, Src Port: 2001 (2001), Dst Port: hosts2-ns (81), Seq: 1, Ack: 1, Len: 56
⊞ Data (56 bytes)
```

  - With MPLS Labels Removed

```
⊞ Frame 3: 110 bytes on wire (880 bits), 110 bytes captured (880 bits) on interface 0
⊟ Ethernet II, Src: Cisco_05:28:38 (00:30:96:05:28:38), Dst: Cisco_e6:fc:39 (00:30:96:e6:fc:39)
  ⊞ Destination: Cisco_e6:fc:39 (00:30:96:e6:fc:39)
  ⊞ Source: Cisco_05:28:38 (00:30:96:05:28:38)
    Type: IP (0x0800)
⊞ Internet Protocol Version 4, Src: 10.31.85.1 (10.31.85.1), Dst: 10.34.85.1 (10.34.85.1)
⊞ Transmission Control Protocol, Src Port: 2001 (2001), Dst Port: hosts2-ns (81), Seq: 1, Ack: 1, Len: 56
⊞ Data (56 bytes)
```

Interpreting Results in Wireshark (6)

- VLAN & VN Tagging example
  - With VLAN Tags

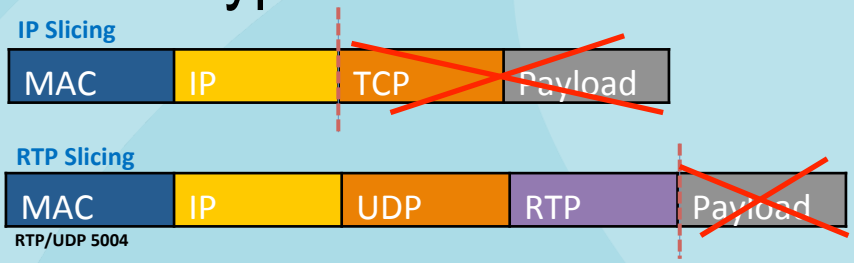
```
⊞ Frame 1: 126 bytes on wire (1008 bits), 126 bytes captured (1008 bits) on interface 0
⊞ Ethernet II, Src: Cisco_df:ae:18 (00:13:c3:df:ae:18), Dst: Cisco_1b:a4:d8 (00:1b:d4:1b:a4:d8)
⊟ 802.1Q Virtual LAN, PRI: 0, CFI: 0, ID: 118
    000. .... .... .... = Priority: Best Effort (default) (0)
    ...0 .... .... .... = CFI: Canonical (0)
    .... 0000 0111 0110 = ID: 118
    Type: 802.1Q Virtual LAN (0x8100)
⊞ 802.1Q Virtual LAN, PRI: 0, CFI: 0, ID: 10
⊞ Internet Protocol Version 4, Src: 10.118.10.1 (10.118.10.1), Dst: 10.118.10.2 (10.118.10.2)
⊞ Internet Control Message Protocol
```

  - With VLAN Tags Removed

```
⊞ Frame 1: 118 bytes on wire (944 bits), 118 bytes captured (944 bits) on interface 0
⊞ Ethernet II, Src: Cisco_df:ae:18 (00:13:c3:df:ae:18), Dst: Cisco_1b:a4:d8 (00:1b:d4:1b:a4:d8)
⊞ Internet Protocol Version 4, Src: 10.118.10.1 (10.118.10.1), Dst: 10.118.10.2 (10.118.10.2)
⊞ Internet Control Message Protocol
```

Interpreting Results in Wireshark (7)

- Conditional Packet Slicing
  - Single out specific packets or traffic types

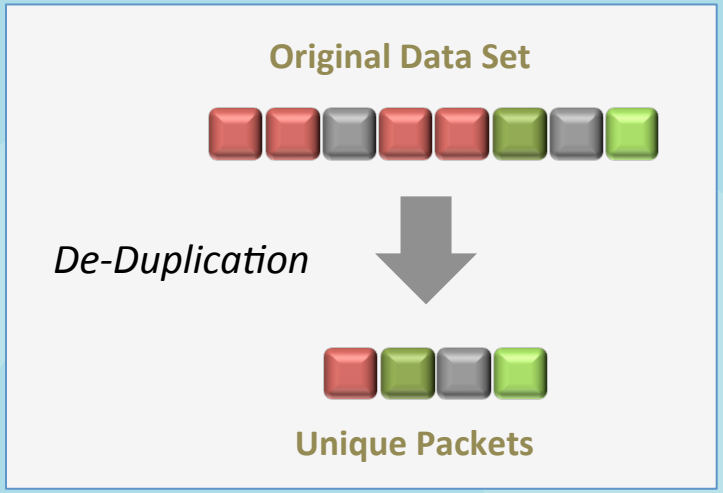


  - Slice packet contents from specified point in packets

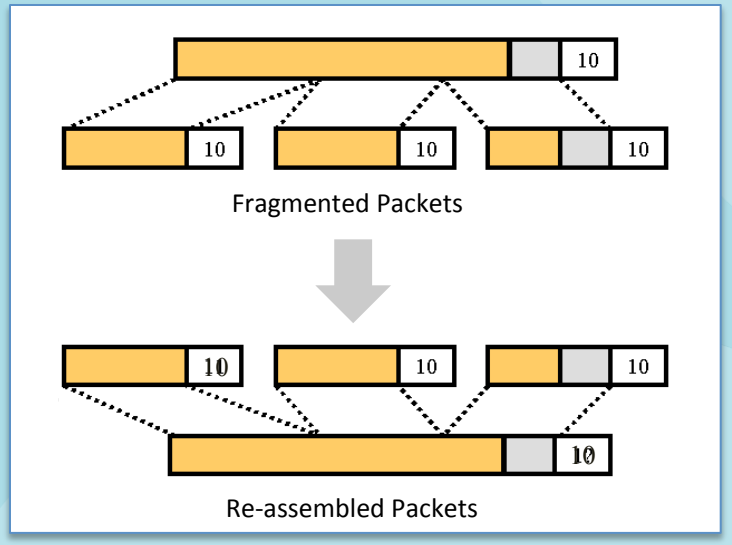
<add Wireshark decoded vSliced packet>

Interpreting Results in Wireshark (8)

- Packet De-duplication
  - Forwards on unique packets
  - Drops all subsequent duplicates with specified time window
  - Expect to see missing sequences

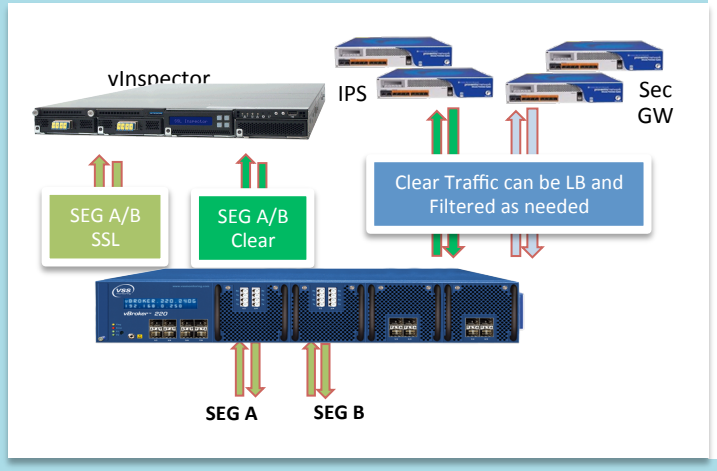


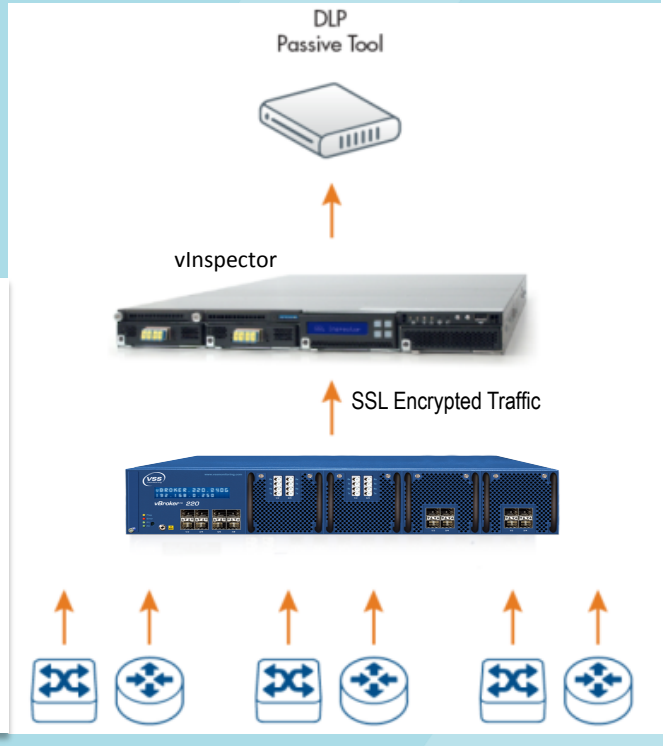
- IP Fragment Reassembly
  - Reassembles "outer" fragments
    - Any IP packet
  - Reassembles "inner" fragments
    - Encapsulated fragmentation, e.g. IP within GTP, IPv4 within IPv6
  - Expect to see missing sequences

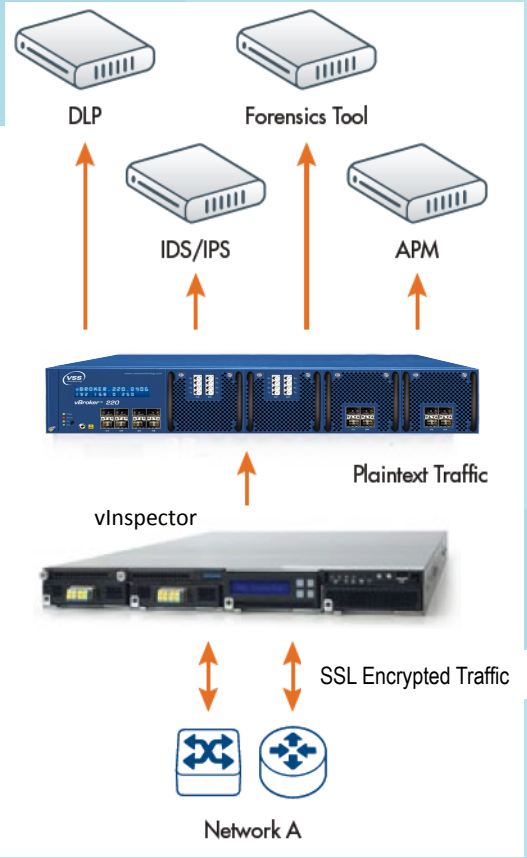


Interpreting Results in Wireshark (9)

- SSL Decryption
  - Detect and decrypt all SSL flows/sessions
  - Forward all clear text/decrypted packets to monitoring & analysis tools







Interpreting Results in Wireshark (9)

- SSL Encryption example

```
⊞ Frame 19: 225 bytes on wire (1800 bits), 225 bytes captured (1800 bits)
⊞ Ethernet II, Src: BrocadeC_a6:e0:4c (00:e0:52:a6:e0:4c), Dst: Cisco_7e:33:44 (e4:d3:f1:7e:33:44)
⊞ IEEE 802.1ad, S-VID: 2222, C-VID: 2211
⊞ Internet Protocol Version 4, Src: 10.250.178.15 (10.250.178.15), Dst: 66.7.254.69 (66.7.254.69)
⊞ Transmission Control Protocol, Src Port: 58468 (58468), Dst Port: https (443), Seq: 1, Ack: 1, Len: 159
⊟ Secure Sockets Layer
  ⊟ TLSv1 Record Layer: Handshake Protocol: Client Hello
      Content Type: Handshake (22)
      Version: TLS 1.0 (0x0301)
      Length: 154
    ⊟ Handshake Protocol: Client Hello
        Handshake Type: Client Hello (1)
        Length: 150
        Version: TLS 1.0 (0x0301)
      ⊞ Random
        Session ID Length: 32
```

VSS Portfolio

TAP Series

Standard passive & active Taps for SPAN, in-line regeneration, replication & aggregation

vBroker™ Series

High capacity flexible packet brokers that provide network-wide visibility, data access and optimization for network monitoring and security tools; both passive and active inline.

High Density

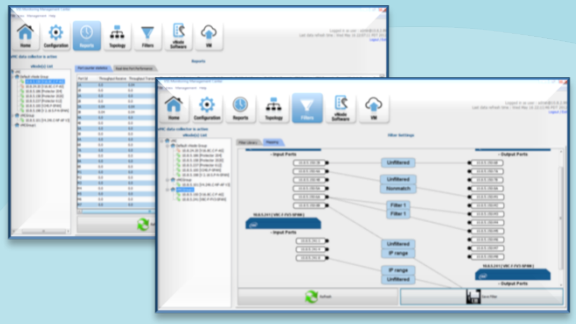
Modularity

vInspector™ Series

Bidirectional, high throughput SSL decryption and re-encryption appliances

VSS Management Center (vMC™)

Network-level management console (MC) for managing VSS packet brokers and TAPs, providing network-wide topology views, configuration, policy mgmt, reporting, and administration/updates.



Summary

Monitoring for Performance and Security is absolutely necessary

- But there are challenges…
- Challenges are increasing as network speeds and data increase

Network Packet Brokers are critical to addressing the challenges

- Address most, if not all, challenges

► **Wireshark can be used with VSS' Network Packet Brokers as an excellent troubleshooting, monitoring, and analysis solution!**