Analyzing Huge Data for Suspicious Traffic

Christian Landström, Airbus DS
Topics

- Overview on security infrastructure
- Strategies for network defense
- A look at malicious traffic incl. Demos
- How Wireshark can help
House rules
Tool-Box

Defaults:
Proxy servers with authentication
Logging, Monitoring, (SIEM)

Layers of Defense:
Firewalls / WAFs
Intrusion Detection / Intrusion Prevention
NIDS/NIPS/HIDS/HIPS
Malware Sensors / Sandboxing / “APT-devices”
Overview on sec. infrastructure

- Depending on
  → area of protection
  → type of attack

- External: Internet facing
- Internal: non-Inet facing
External I

Typical protection for DMZ systems:
Packet filter $\rightarrow$ IPS / APT device $\rightarrow$ local (host-)firewall
Demo #1: DMZ Service

- Monitoring the request size in this example reveals some huge request resulting in a new connection initiated by the FTP Server
Demo #1: DMZ Service

Knowing your applications’ behavior may lead to valid thresholds to reveal anomalies e.g. based on packet length, payload entropy or other factors.
External II

Perimeter defense: Monitoring all protocols
- Know your systems’ configuration
- In-depth understanding of App behavior
- Monitor the events from sec. devices
- Correlate events after sec. alert

→ WebServer accessing other servers after “unsuccessful” exploit?
Demo #2: “Encrypted” sessions

Watch for protocol anomalies e.g. missing HTTP dissector information on HTTP ports containing no valid requests or malformed data.
Demo #2: “Encrypted” sessions

Another example for pretended encrypted traffic not containing a valid SSL handshake

Sample: Using relative Sequence numbers try:
tshark -r <tracefile> -Y "tcp.dstport==443 and tcp.len > 0 and tcp.seq == 1 and !ssl.record"
Internal I

Incoming traffic critical and monitored
But:
Sessions going out are trusted
Mail / Web / FTP etc.
How to spot outgoing malicious stuff
Demo #3: Surfing the web

Also valid protocol requests may hint for an anomaly based on irregular behavior or other indicators.
Internal II

Big issue: Lateral movement and other post-infection activities
- Internal scanning / enumeration
- Access to internal applications
- bruteforce attempts
- legitimate access with stolen credentials

→ Mostly depending on log files from internal sources
Baselining / Anomaly detection

Knowing your application behavior / network flows is critical to spotting malicious events
- Might be easy for default applications
  ➔ Statistics: Conversation e.g.

- How about special applications?
Demo #4: Baselining sample

Especially difficult if application payload types unknown or difficult to baseline

```bash
# tshark -r Trace1.pcap -Y udp -Tfields -e data | more
4b417947534b67534142746157357062474674596d3841524739
e1650518e41793d5abb03d
755d021f5cf975c6342cc14f84caf5e0b863
e1680231b0aee0ebbb648c0a4b14167412c16356e8b6b76db
755f02cf93f622f368d2fef70bf71c5e5f85a8e297eb79795ac04f
```

Legitimate example Skype

```bash
# tshark -r Trace2.pcap -Y udp -Tfields -e data | more
10a6b286d9736aae21af42ddf005f6125f66633de613a63e46
10a6b286d9736aae21af42ddf005f6125f66633de613a63e46
10a7
10a0b286d9736aae21af42ddf005f6125f66633de613a63e46
10b15a78
10bf281d1581812c38e0e0d9o18f2e5458bbc25bc030b0
10a1530e1598ba7ad499afea4ca126827f07de483537d0ad14c0be
```

Malicious example Peacomm.C malware
Baselining approaches e.g. Web

Many approaches for finding unknown sources of malicious activity
Sample: domain lists -> diff approach
- Cat I: Clean or already infected
- Cat II: newly infected
Timely Diff’s -> approach new infections / applications
How Wireshark can help

- Better understanding of your application behavior
- Scripted generation of baselining data
- Long-term comparison of network traces for detecting abnormal changes
- Incident Analysis Results can lead to good rules for IDS/IPS and other appliances
Demo #5-7: How Wireshark can help

- Better understanding of your application behavior
- Scripted generation of baselining data
- Long-term comparison of network traces for detecting abnormal changes
- Incident Analysis Results can lead to good rules for IDS/IPS and other appliances
Demo #5: How Wireshark can help

DNS answers for localhost IP can lead to inactive c2c system
Beware: Also used for lots of valid reasons e.g. SPAM checking

tshark -r 127.0.0.x.pcap -Tfields -e dns.qry.name | grep -v -E "(<valid1>|<valid2>)" | sort | uniq -c | more

[...]  
1 c-0.19-xxxxxxx.avqs.mcafee.com  
1 c-0.19-yyyyyyy.avqs.mcafee.com  
147 <malicious1>.is-certified.com  
148 <malicious2>.dnsalias.com  
146 <malicious3>.dyndns-ip.com  
148 <malicious4>.dyndns-office.com  
148 <malicious5>.doomdns.com
Demo #6-7 How Wireshark can help

<presentation only – sorry>
Monitoring Networks - Proactive

- Use NetFlow/OpenFlow to monitor meta data
  Set up alerts for unusual patterns

- Use IDS/IPS with optimized signatures
  Reduce false positives as much as possible

- Set up Passive DNS / Passive SSL recording servers
  Helps in tracking down name resolution and certificate history
Monitoring Networks - Reactive

- Forensic analysis on full packet captures
  Has to be recorded before something happened, of course
  Carefully selected locations, e.g. Internet outbreaks
- Use NetFlow/OpenFlow for meta data
  Long term storage for forensic searches, e.g. „where did the attacker connect to from the infected system?“
- Use IDS/IPS as custom IoC alarm system
  Write custom IDS rules for known Indicators of Compromise from Wireshark Analysis results
Detecting malicious traffic

- Forget „silver bullets“ – there is no “showmethebadstuff” Wireshark filter
- Attackers hide in plain sight
  DNS, HTTP(S), FTP,...
- Filter out positives
  E.g. Alexa 1 Million
  Known update sites: OS, AV, Vendors
Final Words

- Network defense is a 24/7 challenge
- Attackers only need to succeed once, defenders would need 100% success

Read as: it’s not „if“ but „when“ an attack will succeed.

Expect successful attacks on your network.

- Keep searching
  It’s a continuous task
  Don‘t just wait for some alarm to go off
!! Thank you for attending !!

Questions?

---------------------------
eMail: landi@packet-foo.com
Web: www.packet-foo.com
Twitter: @0x6C616E6469