Traffic Analysis

Part II.B. Techniques and Tools: Network Forensics

CSF: Forensics Cyber-Security
Fall 2015
Nuno Santos
Summary

- Packet and flow analysis
- Network intrusion detection
- NetFlow investigations
- Deep Packet Inspection
Our journey in this course:

- Part I: Foundations of digital forensics
- Part II: Techniques and tools
  - A. Computer forensics
  - B. Network forensics
  - C. Forensic data analysis

Current focus
Previously: Main instruments in cybercrime

Anonymity systems
How criminals hide their IDs

Botnets
How to launch large scale attacks

Digital currency
How to make untraceable payments

Web
How to locate information and services
Such tools operate over networked systems

- Communications involve the transmission of network **packets** from senders to receivers
  - Typically over the Internet

- Naturally, such packets are precious source of **evidence** for investigators
Today: How to analyze evidence from a network?

- Answer: through traffic analysis techniques

- Involves knowing:
  - Where to look for data
  - How to retrieve it
  - How to analyze it
1. Packet sniffers and analyzers
   - Evidence from packet traces

2. Intrusion detection systems
   - Evidence from intrusion detection systems

3. NetFlow
   - Evidence from NetFlow-enabled routers

4. Deep Packet Inspection (DPI)
   - Evidence from DPI-enabled switches
Packet sniffers and analyzers
What is a packet?

- A **packet** is like a container that holds info pertaining to the communication for which it is being used.

- In general, a packet has main three **parts**:
  - Header, payload, and trailer

- The packet includes:
  - where the info is coming from
  - where the info is going to
  - what the info is
  - what the receiver needs to know
Example: TCP header

TCP Flags

C 0x80 Reduced (CWR)
E 0x40 ECN Echo (ECE)
U 0x20 Urgent
A 0x10 Ack
P 0x08 Push
R 0x04 Reset
S 0x02 Syn
F 0x01 Fin

ECN (Explicit Congestion Notification). See RFC 3168 for full details, valid states below.

Packet State    DSB    ECN bits
Syn             0 0     1 1
Syn-Ack         0 0     0 1
Ack             1 1     0 0

No Congestion   0 0     0 0
No Congestion   1 0     0 0

Congestion      1 1     0 0
Receiver Response 1 1     0 1
Sender Response  1 1     1 1

TCP Options

0 End of Options List
1 No Operation (NOP, Pad)
2 Maximum segment size
3 Window Scale
4 Selective ACK ok
8 Timestamp

Checksum

Checksum of entire TCP segment and pseudo header (parts of IP header)

Offset

Number of 32-bit words in TCP header, minimum value of 5. Multiply by 4 to get byte count.

RFC 793

Please refer to RFC 793 for the complete Transmission Control Protocol (TCP) Specification.

CSF - Nuno Santos 2015/16
Example: TCP connection establishment

1. Client sends a SYN to the server. Client sets the segment's sequence number to rand value m
2. Server replies with a SYN-ACK. The ack number is set to m+1, and the sequence number that the server chooses for the packet is another random number n
3. Client sends an ACK back to the server
Packet sniffing and packet sniffers

- **Packet sniffing** is the act of looking at packets as computers pass them over networks.

- Packet sniffing is performed using **packet sniffers**
  - These programs are designed to capture raw data as it crosses the network and translate it into a human readable format for analysis.
  - Can be used to capture only relevant packets.

- Packet sniffers range from **simple**, command-line programs, like tcpdump, to **complex** programs with GUI.
A. Connect the sniffer to a **hub**

B. Connect to **switch**'s SPAN port

C. Use a **network tap** on the ingress connection side of the switch
Packet sniffers: Tcpdump

- **tcpdump** is the granddaddy of all open source packet sniffers.

- Uses **libpcap**, which contains a set of system-independent functions for packet capture and network analysis.
  - Also used by Wireshark.
How to capture packets using tcpdump

- Execute tcpdump command without any option will capture all the packets flowing through all interfaces
- -i option allows you to filter on a particular ethernet interface

$ tcpdump -i eth1

14:59:26.608728 IP xx.domain.netbcp.net.52497 > valh4.lell.net.ssh: . ack 540 win 16554

14:59:26.610602 IP resolver.lell.net.domain > valh4.lell.net.24151: 4278 1/0/0 (73)

14:59:26.611262 IP valh4.lell.net.38527 > resolver.lell.net.domain: 26364+ PTR? 244.207.104.10.in-addr.arpa. (45)
Dropped packets and forensics implications

- Watch tcpdump output in real time can cause dropouts
  - tcpdump will post error message if it dropped any packets

- This is because tcpdump is both capturing and decoding the traffic for display at the same time

- Dropping packets means that potential evidence is lost
  - Record as much as you can; extract the relevant records later
Approaches to reducing dropped packets

1. Write the raw packets to a file (using the -w parameter)

2. Reduce the snapshot size so that tcpdump captures less information from every packet using the -s parameter
   - Analyze the snapshot size to ensure that the information of interest will be included in the snapshot

3. Refine the tcpdump filter to collect fewer, more relevant records
   - If this capture is for a criminal investigation, you will need to ensure that you haven’t excluded any exculpatory evidence
tcpdump uses rich Berkeley Packet Filter (BPF) expressions for specifying what should or should not be collected.

```
tcpdump 'tcp port 80 and (((ip[2:2] - ((ip[0]&0xf)<<2)) - ((tcp[12]&0xf0)>>2)) != 0)'
```

This example will capture all IPv4 HTTP packets to and from port 80 and print only the packets that contain data, not SYN, FIN, and ACK-only packets.

- Homework: understand why
Packet analysis consists in the examination of contents and/or metadata of one or more packets.

Conducted to identify packets of interest and develop a strategy for flow analysis or content reconstruction.

There are many tools available for packet analysis, e.g., wireshark, ngrep,...
Packet analysis techniques

- **Pattern matching**
  - Identify packets of interest by matching specific values within the packet capture (e.g., keyword search)

- **Parsing protocol fields**
  - Extract the contents of protocol fields within packets of interest (e.g., obtain TCP fields of packets)

- **Packet filtering**
  - Separate packets based on the values of fields in protocol metadata (e.g., filter interesting conversation snippet)
Flow analysis consists in examination of sequences of related packets ("flows")

Conducted to identify traffic patterns, isolate suspicious activity, analyze higher-layer protocols, or extract data

Examples of flow analysis tools: Wireshark, tcpflow, pcapcat
Flow analysis techniques

- **List conversations and flows**
  - List all conversations and/or flows within a packet capture, or only specific flows based on their characteristics

- **Export a flow**
  - Isolate a flow, or multiple flows, and store the flow(s) of interest to disk for further analysis

- **File and data carving**
  - Extract files or other data of interest from reassembled flow
Wireshark is a multi-platform network protocol analyzer that allows for isolating, reconstructing, and exporting flows.

It offers the ability to perform packet captures and allowing forensic investigator to analyze the capture offline.
E.g., you can scan through the HTTP traffic until you find a packet with an image (packet 48)
Wireshark: Colorize conversation

- Click **Colorize Conversation** to highlight this conversation between 10.1.1.101 and 10.1.1.1.

**Example**: Wireshark packet display showing conversation details.
You are the manager of a company and receive a tip that an employee is using his computer to view images that violate the company’s computer use policy.

You then hire a forensics investigator to assist in the matter and, together, decide to monitor the suspected employee’s activity on the network for the next week.

Goal: see if there is any evidence to support or refute the claims against the employee viewing images.
Case study: Search through the packets

- After capturing the packets, search through the packets to identify images that violate the policy
Case study: Perform file carving

- Export the portion of the payload that contains the bytes of the image
Case study: Caught in procrastination!

- Exported image created from exported bytes
Network intrusion detection systems
Forensic analysts may need to check IDS logs

As investigators, we have to be prepared to evaluate events that might be of interest.

In an environment that has been instrumented to detect potentially adverse events via network monitoring, it’s very likely that a **IDS alert** is the starting point of our investigation.
First of all, what’s an IDS?

- **Intrusion Detection System (IDS)**
  - Solution by organizations to monitor networks and/or systems for malicious activities or security policy violations

- **IDS vs. IDP**
  - In 2003, Gartner famously pronounced intrusion detection dead, asserting that IDS would be obsolete by 2005
  - Subsequently, most vendors followed suit, rebranding all of their detection solutions as “intrusion prevention systems (IPSs)”
Host intrusion detection/prevention systems (HIDS/HIPS)
- Monitor events in a host and alert on suspicious system activities
- E.g., access to files, logs, passwords, software, etc.
- Better visibility into behavior of individual apps on the host

Network intrusion detection/prevention systems (NIDS/NIPS)
- Monitor network traffic and alert on suspicious network events
- E.g., malformed packets, access patterns, payload content
- Can protect many hosts and look for global patterns
A typical NIDS runs in four basic phases:
Possible responses based on received alert

- NIDS produce 1 of 4 different responses based on the received alert

<table>
<thead>
<tr>
<th></th>
<th>TRUE</th>
<th>FALSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>POSITIVE</td>
<td>True-Positive (Rule matched and attack present)</td>
<td>False-Positive (Rule matched and no attack present)</td>
</tr>
<tr>
<td>NEGATIVE</td>
<td>True-Negative (No rule matched and no attack present)</td>
<td>False-Negative (No rule matched and attack present)</td>
</tr>
</tbody>
</table>
Examples of sensor placement for the Snort NIDS
Intrusion detection techniques

- **Signature-based analysis**
  - Oldest strategy; compares contents of packets, and streams of packets against databases of known, malicious byte sequences in order to identify suspicious traffic

- **Protocol awareness**
  - Detect malformations in network protocols by checking if packets conform to RFC specifications; may require reassemble fragments (Layer 3), flows (Layer 4), or entire app protocols (layer 7)

- **Behavioral analysis**
  - Using a model of normal system behavior, try to detect deviations and abnormalities, e.g. trigger alert if incoming traffic volume increases above certain threshold
Snort

- Popular open-source NISD tool

- It can analyze arbitrary network traffic using two basic techniques: protocol analysis and signature analysis

- Features its own language for specifying rules and a large set of rules for common vulnerabilities

- Occasionally had its own vulnerabilities
  - Feb 19, 2007: Snort IDS are vulnerable to a stack-based buffer overflow, which can result in remote code execution
Examples of snort rules

- Sample snort rule using **msg** keyword
  - Tells the logging and alerting engine the message to print along with a packet dump or to an alert

<table>
<thead>
<tr>
<th>Keyword</th>
<th>msg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax</td>
<td><code>msg: &quot;&lt;message text&quot;;</code></td>
</tr>
<tr>
<td>Example</td>
<td><code>log tcp any any -&gt; any 443 (msg: &quot;TCP port 443 traffic log&quot;;</code></td>
</tr>
<tr>
<td>Description</td>
<td>This rule tells Snort to log any TCP destined for TCP port 443 that reaches the IDS and includes the message “TCP Port 443 trafficlog” with the log entry.</td>
</tr>
</tbody>
</table>
Examples of snort rules

- Sample snort rule using **content** keyword
  - Search for specific content in the packet payload and trigger response based on that data

<table>
<thead>
<tr>
<th>Keyword</th>
<th>content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax</td>
<td><code>content: [!] &quot;&lt;content string&gt;&quot;;</code></td>
</tr>
<tr>
<td>Example</td>
<td><code>alert tcp any any -&gt; any 80 (content: &quot;Password&quot;);</code></td>
</tr>
</tbody>
</table>
| Description | This rule tells Snort to alert on any TCP source traffic destined for TCP port 80 that contains the word “password.” Additional options about the **content** keyword is listed below:  
  - The **content** keyword uses the Boyer–Moore pattern-matching algorithm. A string-search algorithm was developed by Boyer and Moore in 1977.  
  - It is case sensitive.  
  - It can contain mixed text and binary data.  
  - The binary data, enclosed within the pipe (\|) character, represented as hexadecimal numbers.  
  - Multiple content rules can be specified in one rule.  
  - If proceeded by an exclamation mark (\!), the alert will be triggered on packets that do not contain this content. |
Examples of snort rules

- Sample snort rule using **reference** keyword
  - Allows rules to include references to external attack identification systems

<table>
<thead>
<tr>
<th>Keyword</th>
<th>reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax</td>
<td>reference: &lt;id system&gt;,&lt;id&gt;;</td>
</tr>
<tr>
<td>Example</td>
<td>alert tcp any any -&gt; any 80 (msg: “WEB-IIS Microsoft IIS 5.1 and 6.0 WebDAV password bypass attempt”; content: “GET /...%0%af/protected/protected.zip HTTP/1.1” reference:cve,2009-1535;)</td>
</tr>
<tr>
<td>Description</td>
<td>This rule tells Snort to alert on any TCP source traffic destined for TCP port 80 that attempts a WebDAV password bypass and to include a URL reference link to the Common Vulnerabilities and Exposures (CVE) page for that vulnerability.</td>
</tr>
</tbody>
</table>
Sample snort rule using `session` keyword

- It is built to extract user data from TCP Sessions

<table>
<thead>
<tr>
<th>Keyword</th>
<th>session</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax</td>
<td>`session: [printable</td>
</tr>
<tr>
<td>Example</td>
<td><code>log tcp any any &lt;&gt; any 23 (session:printable;)</code></td>
</tr>
<tr>
<td>Description</td>
<td>This rule tells Snort to extract all printable strings in a Telnet packet. The printable keyword only prints user entered or readable data. The <code>session</code> keyword is best suited for postprocessing binary (pcap) log files.</td>
</tr>
</tbody>
</table>
Why investigate NIDS in network forensics?

1. NIDS alerts/logs may include details regarding illicit connections (or attempts) not recorded anywhere else
   - Can be as important an event recorder as a “little black box”

2. NIDS can be configured to alert on, or at least log traffic that firewalls deem perfectly acceptable

3. An investigator could modify a NIDS configuration to begin detecting events it wasn’t previously recording

4. Rarely, the NIDS/NIPS itself might be suspected of compromise
But we must be aware of NIDS limitations

- Can’t inspect encrypted traffic (VPNs, SSL)

- Can’t record and process huge amount of traffic

- High false positive rate
  - Investigators will spend many hours examining evidence

- False negatives: attack is not detected
  - Big problem in signature-based misuse detection (see next)
Evading a signature-based NIDS

- Want to detect “USER root” in packet stream

- Scanning for it in every packet is not enough
  - Attacker can split attack string into several packets; this will defeat stateless NIDS

- Recording previous packet’s text is not enough
  - Attacker can send packets out of order

- Full reassembly of TCP state is not enough
  - Attacker can use TCP tricks so that certain packets are seen by NIDS but dropped by the receiving application
    - Manipulate checksums, TTL (time-to-live), fragmentation
Evading a signature-based NIDS

**Insertion attack**

Insert packet with bogus checksum

**TTL attack**

10 hops

Short TTL to ensure this packet doesn’t reach destination

8 hops

Dropped (TTL expired)
NetFlow investigations
The challenge of line-speed packet collection

- So far, we looked at how packet traces can be used to capture data for later analysis or to feed a NIDS.

- This technique for acquiring network data has been used for decades with much success.

- However, business networks often operate at multigigabit speeds which can overwhelm packet-based capture tools.
NetFlow

- Technology by Cisco that collects and categorizes IP traffic as it passes through the supported network devices.
- Built into the supported devices, NetFlow does not collect the entire payload of the network packets.
  - It creates a cache on the router for each new flow.
Which packets belong to individual flows?

- As IP packets come into a supported device interface, NetFlow scans them for the following seven fields:
  1. Source IP address
  2. Destination IP address
  3. Source port number
  4. Destination port number
  5. IP protocol
  6. Type-of-service (ToS) byte
  7. Input logical interface

- If these fields match an existing flow the byte count for the flow entry is incremented within the device cache
  - Else the packet is part of new flow
Heads up!

- NetFlows logs flows, not TCP connections!
  - Flows are uni-directional

- For each TCP connection, NetFlow records two flows
Collection of NetFlow from decentralized devices

- Flows can be combined from several network devices
Common uses for NetFlow

- **Network traffic accounting**
  - E.g., tracks everywhere a host is connected, as well as amount of bytes involved

- **Usage-based billing**
  - E.g., service providers can invoice based on 95\textsuperscript{th} percentile, over allotted bandwidth, and so on that are based on IP, subnet, protocol, and so on

- **Network planning**
  - E.g., forecast future usage-based on historical trends of a host and/or application

- **Security and network monitoring**
  - E.g., identify hosts participating in unwanted activities such as network scans, DoS, and so on
Benefits of NetFlow to forensic analysis

- Analyzing flows can greatly reduce the amount of data needed to be analyzed and make it simpler to identify any suspicious traffic for future investigation.
  - Example: TCP/6667 commonly used for Internet Relay Chat.
Examples of detecting nefarious network traffic

- **SYN scan**
  - Look for flows from the same host with only the SYN flag set. If a host has at least 100 flows (configurable threshold) with only the SYN flag set, this could trigger an alarm

- **Internet threat**
  - Compare flows to a list of known compromised Internet hosts to make sure that no one is communicating with a host on the list

- **Suspicious flows**
  - Look for hosts with flows reaching a large number of destinations, when the destination count is > a threshold of 50 (configurable)
Deep Packet Inspection
Back in 2007, a boom in bandwidth availability

- Broadband became ubiquitous
  - High penetration rates (over 50% in Korea, Taiwan, Holland and Canada)
  - Over 50% of on-line households are BB

- Telcos began upgrading infrastructure:
  - ADSL
  - Cable
  - Fiber (latter)

- Bandwidth per user ramped up
More bandwidth, led to more net-intensive apps

- Peer-to-peer
  - BitTorrent

- VoIP
  - Skype

- Streaming
  - YouTube

- Online gaming
  - World of Warcraft
Which increased costs for most ISPs

End user pays flat rate

Local ISP pays per volume
ISPs started to deploy DPI for traffic shaping

- **Deep Packet Inspection (DPI)** is a technology used to capture network packets as they pass through routers and other network devices, and perform packet filtering to examine the data and find deeper information about the data carried by the packets.
Shallow (standard) packet inspection

- Inspection of packet headers for optimization of packet routing, detection of network abuse, and statistical analysis.

header info reveals communication intent
Deep packet inspection

- Form of network packet filtering that examines the data part of a packet as it passes an inspection point

Signature over several packets found

Information regarding connection state
Analysis by string match

- **Reasoning:**
  - Many applications have pure textual identifiers
  - Easy to search for
  - Very easy if in a specific location within a packet
  - Uniqueness not always guaranteed
Analysis by numerical properties

- Property is not only content:
  - Packet size
  - Payload/message length
  - Position within packet

- In some cases sparse and spread over several packets

<table>
<thead>
<tr>
<th>Connection #1</th>
<th>Connection #2</th>
<th>Connection #3</th>
<th>Connection #4</th>
</tr>
</thead>
<tbody>
<tr>
<td>35 8A 27 7F</td>
<td>15 82 98 71</td>
<td>A5 80 72 7F</td>
<td>95 88 8A 7F</td>
</tr>
</tbody>
</table>

Identifying John Doe Protocol
Analysis by numerical properties: Skype (old)

UDP Messages

Client

18 byte message

N+8

11 byte message

Evolution

23 byte message

N+8+5

Either 18, 51 or 53 byte message

Server
Behavior and heuristic analysis

- Behavior = the way in which something functions or operates

- Heuristic = problem-solving by experimental and especially trial-and-error methods

- OK, but what does this mean? Examples:
  - Statistics: on average payload size is between X to Y
  - Actions: Login using TCP connection followed by a UDP connection on subsequent port number

- Very effective analysis when application uses encryption
HTTP vs. BitTorrent (handshake)
DPI (ab)use is widespread in today’s ISPs

- **Copyright enforcement**
  - ISPs are requested by copyright owners or required by courts or official policy to help enforce copyrights

- **Targeting advertising**
  - ISPs are able to monitor web-browsing habits allowing them to gain information about their customers' interests

- **Lawful interception**
  - ISPs are required by almost all governments worldwide to enable lawful intercept capabilities
ISP’s DPI is great source of evidence

- In addition to using DPI for the security of their own networks, governments in North America, Europe, and Asia use DPI for various purposes such as **surveillance** and **censorship**

- Many of these programs are classified
Traffic analysis enables forensic investigators to determine the history of events involved in network communications.

Such analysis is based on specific evidence that can take different forms, e.g., specific packets (network traces), log entries (IDS), flow summaries (NetFlow), etc.

It is up to the forensic investigator to look for and use all the evidence that can be found in the scene.
Primary bibliography

Wireless and mobile network investigations