Advanced Structural Fingerprinting

Humberto J. Abdelnur  Radu State  Olivier Festor

<firstname>.<lastname>@loria.fr

INRIA Nancy-Grand Est
http://madynes.loria.fr

September 17, 2008
1. Introduction
   - Network Signatures

2. Fingerprinting
   - Syntax Fingerprinting
   - Training Phase

3. Results
   - Experimental Results

4. Conclusion
Network Fingerprinting

Objective
- Identify specific services running a common protocol
- Determine the implementation/vendor from the traffic

Applicability
- Network topology discovery, inventory
- Detect rogue/SPIT systems and stealth intruders
- Identify copyright infringements

Banners can not be trusted
- Most implementations allow to modify the banners
- Signatures hidden by protocol scrubbers
Current Approaches

Active Fingerprinting
- Request/Response queries to observe behavior
- Normal/abnormal messages sent
- Network flow invasive

Passive Fingerprinting
- Monitor and classifies traffic
- Observes syntax, state machine
- “What you see is what you get”
- No overhead traffic generation
- Suitable for fingerprinting on the fly

Need for an automated bootstrapping phase
**SIP Context**

Alice@domain.com

Bob@domain.com

```
INVITE sip:Bob@domain.com
Via: SIP/2.0/UDP 192.168.0.1 ;branch=z9hG4bK34
From: <sip: Alice@domain.com >;tag=as07b23bad
To: <sip: Bob@domain.com >
Call-ID: 12345@192.168.0.4
Cseq: 100 INVITE
```

100 Trying

```
200 OK
```

```
ACK sip:Bob@domain.com
Via: SIP/2.0/UDP 192.168.0.1 ;branch=z9hG4bK34
From: <sip: Alice@domain.com >;tag=as07b23bad
To: <sip: Bob@domain.com >;tag=Cq0eb2d
Call-ID: 12345@192.168.0.4
Cseq: 100 ACK
```

What SIP is?
- A signalling protocol
- Request/Response structure (HTTP-like)
- Text-based protocol

What SIP is not?
- Media transport
Message Signatures

Problem Statement

- Behavior is not fully/specifically documented in RFCs
- Implementations don’t fully comply to the specifications

SIP Equipment A

```
REGISTER sip:192.168.1.144 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.2:7060;rport;branch=z9hG4bKgydxyvae
Max-Forwards: 70
To: "humbol" <sip:5555@192.168.1.144>
From: "humbol" <sip:5555@192.168.1.144>;tag=jygph
Call-ID: ibfyglrrpmqbe@192.168.1.2
CSeq: 928 REGISTER
Allow: INVITE,ACK,BYE,CANCEL,OPTIONS,PRACK,REFER,NOTIFY,INFO
Contact: <sip:5555@192.168.1.2:7060>;expires=3600
User-Agent: Twinkle/1.0.1
Content-Length: 0
```

```
INVITE sip:79401@192.168.1.144 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.49;rport;branch=29hG4bKomjgpxec
Max-Forwards: 70
To: <sip:79401@192.168.1.144>
From: "Bob" <sip:6666@192.168.1.144>;tag=nsxsr
Call-ID: tjqbyxysbcramy@192.168.1.49
CSeq: 729 INVITE
Allow: INVITE,ACK,BYE,CANCEL,OPTIONS,PRACK,REFER,NOTIFY,INFO
Contact: <sip:6666@192.168.1.49>
Content-Type: application/sdp
Supported: replaces,norefersub,100rel
User-Agent: Twinkle/1.0.1
Content-Length: 304
```
Message Signatures

Problem Statement

- Behavior is not fully/specifically documented in RFCs
- Implementations don’t fully comply to the specifications

SIP Equipment B

```
REGISTER sip:192.168.1.144 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.20:5060;branch=z9hG4bK4205b326
From: <sip:7940@192.168.1.144>;tag=000b46d9cb-1a84cfd8
To: <sip:7940@192.168.1.144>
Call-ID: 000b46d9-cb860003-66d2804f-527006cb@192.168.1.20
Max-Forwards: 70
CSeq: 102 REGISTER
User-Agent: Cisco-CP7940G/8.0
Contact: <sip:7940@192.168.1.20:5060;transport=udp>
Expires: 3600
```

```
INVITE sip:611@192.168.1.144 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.49:5060;branch=z9hG4bK50979e8b
From: "6666" <sip:6666@192.168.1.144>;tag=001ae2bc8b-4f6a3bc6
To: <sip:611@192.168.1.144>
Call-ID: 001ae2bc-8b7c001a-40b4297e-1611ee91@192.168.1.49
Max-Forwards: 70
CSeq: 102 INVITE
User-Agent: Cisco-CP7940G/8.0
Contact: <sip:6666@192.168.1.49:5060;transport=udp>
Expires: 180
Allow: ACK,BYE,CANCEL,INVITE,NOTIFY,OPTIONS,REGISTER,UPDATE
Supported: replaces,join,norefersub	
Content-Length: 276
```
### Problem Statement

- Behavior is not fully/specifically documented in RFCs
- Implementations don’t fully comply to the specifications

<table>
<thead>
<tr>
<th></th>
<th>Equipment A</th>
<th>Equipment B</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Call-ID Length</strong></td>
<td>15</td>
<td>35</td>
</tr>
<tr>
<td><strong>Allow Order</strong></td>
<td>INVITE,ACK,BYE,CANCEL,OPTIONS,PRACK,REFER,NOTIFY,INFO</td>
<td>ACK,BYE,CANCEL,INVITE,NOTIFY,OPTIONS,REGISTER,UPDATE</td>
</tr>
<tr>
<td><strong>User-Agent Banner</strong></td>
<td>Twinkle/1.0.1</td>
<td>Cisco-CP7940G/8.0</td>
</tr>
</tbody>
</table>
The Big Picture

Challenges\(^1\)
- Be robust to malicious scrubbers
- **Automate** the signature discovery

\(^1\)Assuming we known the protocol

Operational Framework

- Structural Inference
- Semantic Comparison
- Variants Identification
- Invariants Identification
- Features Construction
- Messages Classification
An ABNF grammar

Grammar components:
- $\Sigma$ - Terminals (e.g. “QUERY”, “RESPONSE”, “/“)
- $N$ - Non-Terminals (e.g. Message, Value, Header)
- $e_1 .. e_n$ - Sequences
- $e_1/../e_n$ - Choices
- $e^{i..j}$ - Repetitions

Rule definitions
- Sequence of 3 productions
- Choice between headers
- Repetition of 1 or more Values
- Choice between values

Non-Terminal productions

Message = Header *SP 1* Value

Header= ( "QUERY" / "RESPONSE") "/" Version

Value = "(" User / Host / Port ")"
Syntax Inference

- ABNF grammar specification is known
- Messages can be represented by a tree like structure
- Structure used rather than just lexicon
- Generic approach, allows the parsing of any rule of any grammar

```
RESPONSE/2 (bob)(x.org)(5060)
```
Syntax Fingerprinting

Node Comparison

Comparison Matching

- Shared items between nodes
- Tags and ancestors tags must be equal
- Sequences items must be ordered equally
- Repetitions can be unordered

RESPONSE/2 (bob) (x.org) (192.168.1.9)

RESPONSE/2 (Org) (192.168.1.2)
Syntax Fingerprinting

Node Comparison

Comparison Matching

- Shared items between nodes
- Tags and ancestors tags must be equal
- Sequences items must be ordered equally
- Repetitions can be unordered
Syntax Fingerprinting

Node Comparison

Comparison Matching

- Shared items between nodes
- Tags and ancestors tags must be equal
- Sequences items must be ordered equally
- Repetitions can be unordered

RESPONSE/2 (bob) (x.org)(192.168.1.9)

RESPONSE/2 (Org) (192.168.1.2)
Structural Features

- Identify signatures in the paths of the tree
- Identify signatures as:
  - Contents
  - Lengths
  - Orders
  - Functions
- Save Path, Type & Value triplets
- Classify messages using Closest Neighbor

<table>
<thead>
<tr>
<th>Field path</th>
<th>Feature associated</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message.2.(2)</td>
<td>Order</td>
<td>User, Host, Port</td>
<td></td>
</tr>
<tr>
<td>Message.1.(?)</td>
<td>Length</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Message.0.Header.2</td>
<td>Content</td>
<td>'2'</td>
<td></td>
</tr>
</tbody>
</table>

```
RESPONSE/2 (Bob)(domain.org)(5060)
```
Phase 1: Variants Identification

- Pairwise comparison of messages from the same device
- The differences identify the Variant fields
- These fields are of no interest
  - Configuration values
  - Context specific values

Example:

INVITE sip:611@192.168.1.144 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.49:5060;branch=z9hG4bK50979e8b
From: "6666" <sip:6666@192.168.1.144>;tag=001ae2bc8b-4f6a3bc6
To: <sip:611@192.168.1.144>
Call-ID: 001ae2bc-8b7c001a-40b4297e-1611ee91@192.168.1.49
Max-Forwards: 70
CSeq: 102 INVITE
User-Agent: Cisco-CP7940G/8.0
Contact: <sip:6666@192.168.1.49:5060;transport=udp>
Expires: 180
Allow: ACK,BYE,CANCEL,INVITE,NOTIFY,OPTIONS,REGISTER,UPDATE
Supported: replaces,join,norefersub
Content-Length: 276
Phase 2: Features Identification

- Pairwise comparison of messages from different devices
- Filter differences that are Invariant fields
- These fields are the Signatures
  - Same values for the same device
  - but different within implementations

```
INVITE sip:611@192.168.1.144 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.49;branch=z9hG4bK50979e8b
From: "6666" <sip:6666@192.168.1.144>;tag=001ae2bc8b-4f6a3bc6
To: <sip:611@192.168.1.144>
Call-ID: 001ae2bc-8b7c001a-40b4297e-1611ee91@192.168.1.49
Max-Forwards: 70
CSeq: 102 INVITE
User-Agent: Cisco-CP7940G/8.0
Contact: <sip:6666@192.168.1.49:5060;transport=udp>
Expires: 180
Allow: ACK, BYE, CANCEL, INVITE, NOTIFY, OPTIONS, REGISTER, UPDATE
Supported: replaces, join, nofrefs, sub
Content-Length: 276
```
Experimental Results

- Fingerprinting framework implemented in Python
- 21981 recollected SIP messages labeled (from 26 different apps/conf)
- 15% of the messages were sufficient to train the system
- 271 features discovered
- Classifications:
  - Used between 10 to 58 features
  - Average classification time 0.08 seconds

### Efficiency

150\(^1\) Xeon-Woodcrest nodes, dual-core 64 bits, 2GB RAM

<table>
<thead>
<tr>
<th>Actions</th>
<th>computed actions</th>
<th>Total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase 1</td>
<td>571.234</td>
<td>1 hour</td>
</tr>
<tr>
<td>Phase 2</td>
<td>8.175.419</td>
<td>10 hours</td>
</tr>
</tbody>
</table>

### Accuracy

<table>
<thead>
<tr>
<th>Classification</th>
<th>True Positive</th>
<th>False Positive</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>21422</td>
<td>32</td>
</tr>
<tr>
<td>False Negative</td>
<td>490</td>
<td>N.A.</td>
</tr>
<tr>
<td><strong>Accuracy</strong></td>
<td>0.998</td>
<td>0.976</td>
</tr>
<tr>
<td><strong>Sensitivity</strong></td>
<td>0.976</td>
<td></td>
</tr>
<tr>
<td><strong>Specificity</strong></td>
<td>0.999</td>
<td></td>
</tr>
</tbody>
</table>

\(^1\)Experiments were carried out using the Grid’5000 experimental testbed
### Experimental Results

#### Error Analysis

- **559 (2.5%)** miss classifications
- **37** syntactically invalid
- **32** False Positive
- **490** False Negative
- **203** OPTIONS 2nd-choice
- **126** OPTIONS 3rd-choice
- **17** OPTIONS same dev.
- **9** OPTIONS keep Alive
- **454** filtered correctly in top-2
- **24** filtered correctly in top-3
- **11** filtered correctly in top-4
- **95** ACK
- **64** left

OPTIONS sip:192.168.1.4:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.101;rport;branch=z9hG4bKc0a8016500000000b4550c64f000000f5000000e;
Content-Length: 0
Call-ID: F28A8FE4-1FF9-4937-AF8D-81B29FD607FE@192.168.1.101
CSeq: 20 OPTIONS
From: <sip:0231555777@192.168.1.4>;tag=1286870423922
Max-Forwards: 70
To: <sip:192.168.1.4:5060>
Experimental Results

Scalability

- 2091 recollected SIP messages (6 different applications)
- Trained several times with only 15% of the traces

<table>
<thead>
<tr>
<th></th>
<th>Msgs</th>
<th>Feat.</th>
<th>FP</th>
<th>FN</th>
<th>Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>15%</td>
<td>15%</td>
<td>125-189</td>
<td>3</td>
<td>35-231</td>
<td>0.997-0.979</td>
</tr>
<tr>
<td>20-40%</td>
<td>20-40%</td>
<td>133-194</td>
<td>6</td>
<td>18-44</td>
<td>0.998-0.995</td>
</tr>
<tr>
<td>50-90%</td>
<td>50-90%</td>
<td>165-193</td>
<td>1</td>
<td>20-19</td>
<td>0.998</td>
</tr>
<tr>
<td>100%</td>
<td>100%</td>
<td>174</td>
<td>1</td>
<td>20</td>
<td>0.998</td>
</tr>
</tbody>
</table>
Conclusion

- We built a robust automated signature discovery framework that:
  - Does not rely on lexicon
  - Exploit arborescent structures
  - Is generic
- It was successfully applied to SIP
  - We have a large database of device traces
  - Accuracy of the system is convincing
- Limitations:
  - Syntactically known protocols
  - Clear text flows
- Future work:
  - Measure entropy of the fields
  - Recognize behavior of protocol stacks
  - Unknown protocols
Classification

- **True Positives**
  - Messages from devices analyzed during the training . . .
  - . . . that were correctly classified to the source

- **True Negatives**
  - Messages from devices which were not used during the training . . .
  - . . . that were classified as unknown

- **False Positives**
  - Messages that were classified to one of the known devices . . .
  - . . . but were not generated from such device

- **False Negative**
  - Messages from devices used during the training . . .
  - . . . but classified as unknown by the system