SAD THUG: STRUCTURAL ANOMALY DETECTION FOR TRANSMISSIONS OF HIGH-VALUE INFORMATION USING GRAPHICS

Jonathan P. Chapman
Threat Model
Steganography in Malware

Methods Used by Malware

<table>
<thead>
<tr>
<th>Destination</th>
<th>JPEG</th>
<th>PNG</th>
<th>Other/Image</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structure</td>
<td>13</td>
<td>3</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>Image</td>
<td>4</td>
<td>5</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>Unclear</td>
<td>2</td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

Detection Methods in Research Papers

<table>
<thead>
<tr>
<th>Cover Medium</th>
<th>JPEG</th>
<th>Bitmap</th>
<th>Unspecified Image</th>
<th>Network Packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structure</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Media Data</td>
<td>6</td>
<td>3</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Headers</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Timing</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Storing Compressed Image Data

Raw image data

Encoding

Storage

File Header
Meta-Data
Decoder Info
File Footer

<table>
<thead>
<tr>
<th>SOI</th>
<th>APP1</th>
<th>DQT0</th>
<th>DQT1</th>
<th>Scan</th>
<th>EOI</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>58</td>
<td>60</td>
<td>60</td>
<td>247027</td>
<td>2</td>
</tr>
</tbody>
</table>
Structural Embedding Approaches

- **Clean**
  - File Header
  - Meta-Data
  - Decoder Info
  - File Footer

- **Appending**
  - File Header
  - Meta-Data
  - Decoder Info
  - Appended Data
  - File Footer

- **Segment Injection**
  - File Header
  - Meta-Data
  - Decoder Info
  - Comment
  - Injected Data
  - File Footer

- **Byte Stuffing**
  - File Header
  - Meta-Data
  - Decoder Info
  - Stuffed Data
  - File Footer

```
<table>
<thead>
<tr>
<th>SOI</th>
<th>APP1</th>
<th>DQT0</th>
<th>DQT1</th>
<th>Scan</th>
<th>EOI</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>50</td>
<td>68</td>
<td>68</td>
<td>247027</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SOI</th>
<th>APP1</th>
<th>DQT0</th>
<th>DQT1</th>
<th>Scan</th>
<th>EOI</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>59</td>
<td>69</td>
<td>69</td>
<td>247027</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SOI</th>
<th>APP1</th>
<th>DQT0</th>
<th>DQT1</th>
<th>Scan</th>
<th>EOI</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>50</td>
<td>69</td>
<td>69</td>
<td>247027</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SOI</th>
<th>APP1</th>
<th>DQT0</th>
<th>DQT1</th>
<th>Scan</th>
<th>EOI</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>59</td>
<td>69</td>
<td>69</td>
<td>247027</td>
<td>2</td>
</tr>
</tbody>
</table>
```
The SAD THUG Approach: Cornerstones

- Interpret file type as formal language
  - Works for any structured file type (proof of concept: JPEG, PNG)

- Build approximate model from training data

- Match files against model, adding restrictions
  - Threshold $\tau$: minimum observation count
  - Factor $\alpha$: Defines acceptance range
The SAD THUG Approach: Model

- Deterministic finite automaton, models clean files
- States = Segment types
- Directed Edges = Segments observed in that order
- Annotations to edges:
  - Second segment has fixed length: Count of observations
  - Variable length: Distribution of observed lengths
Evaluation Setup

- Base data set
  - Recursive crawling of Alexa Top 25 websites
  - ~300,000 image files
    - 270,000 JPEG
    - 33,000 PNG
Evaluation Setup

- 10-fold cross-validation
- Test set
  - Unmodified files
  - Live images (ZeusVM)
  - Embedding with malware methods and real/resembling data
- JPEG only: Stegdetect: Append and Invisible Secrets methods
Evaluation: Residual Bytes in Base Data Set

**JPEG**
- Frequency distribution
- 5233 values
  - 6 large values

**PNG**
- Frequency distribution
- 289 values
  - 1 large value

**JPEG**
- Frequency distribution
- 9768 values
  - 2 large values

**PNG**
- Frequency distribution
- 485 values
  - 2 large values
## Evaluation Setup: JPEG

<table>
<thead>
<tr>
<th>Name</th>
<th>Method</th>
<th>Industry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cerber</td>
<td>Appending</td>
<td>Ransomware</td>
</tr>
<tr>
<td>DuQu (1 &amp; 2)</td>
<td>Appending</td>
<td>APT/Government</td>
</tr>
<tr>
<td>Hammertoss</td>
<td>Appending</td>
<td>APT/Government</td>
</tr>
<tr>
<td>Microcin</td>
<td>Appending</td>
<td>APT/Government</td>
</tr>
<tr>
<td>SyncCrypt</td>
<td>Appending</td>
<td>Ransomware</td>
</tr>
<tr>
<td>Tropic Trooper</td>
<td>Appending</td>
<td>APT/Government</td>
</tr>
<tr>
<td>ZeusVM/Zberp</td>
<td>Injection</td>
<td>Banking-Trojan</td>
</tr>
<tr>
<td>APP0</td>
<td>Byte Stuffing</td>
<td>Fraunhofer FKIE</td>
</tr>
<tr>
<td>APP1: Comment</td>
<td>Injection</td>
<td>Fraunhofer FKIE</td>
</tr>
</tbody>
</table>
Evaluation Results: JPEG

- True Negative Ratio
- Classification Ratio

- SAD THUG: True Positives
- Stegdetect: True Positives
- Stegdetect: Error

Apps:
- Cerber
- DuQu
- Hammertoss
- Microbin
- SyncCrypt
- Tropic Trooper
- ZeusVM
- ZeusVM/Zberp

Injection, Byte Stuffing
## Evaluation Setup: PNG

<table>
<thead>
<tr>
<th>Name</th>
<th>Method</th>
<th>Industry</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Brazilian EK”</td>
<td>Appending</td>
<td>Exploit Kit</td>
</tr>
<tr>
<td>Cry Locker</td>
<td>Byte Stuffing</td>
<td>Ransomware</td>
</tr>
<tr>
<td>DNS Changer</td>
<td>Injection</td>
<td>Exploit Kit</td>
</tr>
<tr>
<td>aaAa</td>
<td>Injection</td>
<td>Fraunhofer FKIE</td>
</tr>
<tr>
<td>pHYS</td>
<td>Byte Stuffing</td>
<td>Fraunhofer FKIE</td>
</tr>
</tbody>
</table>
Evaluation Results: PNG

- True Negative Ratio
- True Positive Ratio

Injection: aaAa
Appending: Brazilian EK
Byte Stuffing: CryLocker
DNSChanger
pHYs
SAD THUG
Key Takeaways

- SAD THUG makes thugs sad!
- Structural embedding methods as prevalent as image data-based methods!
- SAD THUG detects them, including unknown methods!
- SAD THUG can be used for any structured file type
Thank you!