Obfuscation defeated: Leveraging electromagnetic signals for malware classification with Deep learning

September 24, 2021

Duy-Phuc PHAM
Introduction

IoT Malware analysis

Machine Learning & Deep Learning

Discussion
Introduction

- Trending of attacks on embedded devices.
- Malware analysis and bypasses: difficulties such as malware evasion techniques, packed and obfuscated samples.
- Limited resources of embedded devices, diversity of architectures.
Proposed solutions

- Black-box monitor
- Side channel information
  - Power consumption, heat & EM
  - Sound (freq.)
  - Cache, HPC (software)
Outline

- Generate datasets: Malware insights
- AHMAM framework: Automatically record EM emanations using oscilloscopes.
- Data processing
- Machine Learning & Deep Learning classification
Dataset: Understanding of IoT malware epidemiology

AVClass to classify malware labels
Dataset: Understanding of IoT malware insights

- AVClass to classify malware labels
- Code reviews and reverse engineering

<table>
<thead>
<tr>
<th>DDoS</th>
<th>Ransomware</th>
<th>Rootkits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mirai</td>
<td>GonnaCry (AES, Blowfish)</td>
<td>KeySniffer</td>
</tr>
<tr>
<td>Bashlite</td>
<td></td>
<td>MaK_It</td>
</tr>
</tbody>
</table>
**Dataset: Variations**

- AVClass to classify malware labels
- Code reviews and reverse engineering
- Obfuscations

- UPX, Tigress, O-LLVM
- Opaque predicates, bogus control flow, instructions substitution, control-flow flattening; packer and code virtualization
Proposed framework (Open source)

Data acquisition
- Dataset generation
- Dataset variations
- Synthetic user environment

Data storage
- Dynamic malware execution
- EM

Data preprocessing
- Time domain
- STFT
- Spectrogram
- Features selection

Malware classification
- SVM
- NB
- MLP
- CNN
Target device

Requirements

- Multi-purpose embedded device.
- Prominent architecture (ARM).
- Vulnerable to EM side-channel attack.

→ Raspberry Pi 2B
Data processing

- 100k(traces) * 2(MSs) * 2.5(s)
- Short-time Fourier transform
- Band-pass filter
Features selection

\[ \text{NICV}(X, Y) = \frac{\text{Var}[\mathbb{E}[X|Y]]}{\text{Var}[X]} \]

\[ F_{\text{extract}} = \{ \text{argmax}_\epsilon (\frac{1}{D} \sum_{d=0}^{D-1} [(\text{NICV}(X, Y))^F_d]) \} \]
Machine Learning & Deep Learning models

- Naive Bayes (NB)
- Support vector machine (SVM)
- Multi-Layer Perceptron (MLP)
- Convolutional Neural Network (CNN)
Malware classification results

<table>
<thead>
<tr>
<th>Scenarios</th>
<th>#</th>
<th>MLP</th>
<th>CNN</th>
<th>LDA+NB</th>
<th>LDA+SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>4</td>
<td>99.75</td>
<td>99.82</td>
<td>97.97</td>
<td>98.07</td>
</tr>
<tr>
<td>Family</td>
<td>6</td>
<td>98.57</td>
<td>99.61</td>
<td>97.19</td>
<td>97.27</td>
</tr>
<tr>
<td>Novelty</td>
<td>5</td>
<td>88.41</td>
<td>98.85</td>
<td>98.25</td>
<td>98.61</td>
</tr>
<tr>
<td>Virtualization</td>
<td>2</td>
<td>95.60</td>
<td>95.83</td>
<td>91.29</td>
<td>91.25</td>
</tr>
<tr>
<td>Packer</td>
<td>2</td>
<td>93.39</td>
<td>94.96</td>
<td>83.62</td>
<td>83.58</td>
</tr>
<tr>
<td>Obfuscation</td>
<td>7</td>
<td>73.79</td>
<td>82.70</td>
<td>64.29</td>
<td>64.47</td>
</tr>
<tr>
<td>Executables</td>
<td>31</td>
<td>73.56</td>
<td>82.28</td>
<td>70.92</td>
<td>71.84</td>
</tr>
</tbody>
</table>

**Table 1.** Accuracy obtained with MLP, CNN, LDA + NB and LDA + SVM applied on several scenarios.
Conclusion

- Published work: Duy-Phuc Pham, Damien Marion, Mathieu Mastio, and Annelie Heuser. ACSAC 2021. *Obfuscation Revealed: Leveraging Electromagnetic Signals for Obfuscated Malware Classification*.
- Open source: https://github.com/ahma-hub/
- Future work
  - Implement research on Software Defined Radio.
  - Extend to side-channel rootkits detection.
Thank you!
Monitor device(s)

- Picoscope 6000
- Keysight Infiniium
- Nooelec Smart SDR
Deep Learning models

- Multi-Layer Perceptron (MLP)
- Convolutional Neural Network (CNN)
Deep Learning models (MLP)

<table>
<thead>
<tr>
<th>Layer</th>
<th>Size</th>
<th>Filter</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flatten</td>
<td>spectrogram_size</td>
<td></td>
<td>LeakyReLU</td>
</tr>
<tr>
<td>Dense</td>
<td>500</td>
<td>_</td>
<td>LeakyReLU</td>
</tr>
<tr>
<td>Dense</td>
<td>200</td>
<td>_</td>
<td>LeakyReLU</td>
</tr>
<tr>
<td>Dense</td>
<td>100</td>
<td>_</td>
<td>LeakyReLU</td>
</tr>
<tr>
<td>Dense</td>
<td>nb_labels</td>
<td>_</td>
<td>softmax</td>
</tr>
</tbody>
</table>
Deep Learning models (CNN)

<table>
<thead>
<tr>
<th>Layer</th>
<th>Size</th>
<th>Filter</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convolution</td>
<td>64</td>
<td>7 × 7</td>
<td>relu</td>
</tr>
<tr>
<td>Max Pooling</td>
<td>64</td>
<td>2 × 2</td>
<td>_</td>
</tr>
<tr>
<td>Convolution</td>
<td>128</td>
<td>3 × 3</td>
<td>relu</td>
</tr>
<tr>
<td>Convolution</td>
<td>128</td>
<td>3 × 3</td>
<td>relu</td>
</tr>
<tr>
<td>Max Pooling</td>
<td>128</td>
<td>2 × 2</td>
<td>_</td>
</tr>
<tr>
<td>Convolution</td>
<td>256</td>
<td>3 × 3</td>
<td>relu</td>
</tr>
<tr>
<td>Convolution</td>
<td>256</td>
<td>3 × 3</td>
<td>relu</td>
</tr>
<tr>
<td>Max Pooling</td>
<td>256</td>
<td>2 × 2</td>
<td>_</td>
</tr>
<tr>
<td>Dense</td>
<td>128</td>
<td>_</td>
<td>relu</td>
</tr>
<tr>
<td>Dense</td>
<td>64</td>
<td>_</td>
<td>relu</td>
</tr>
<tr>
<td>Dense</td>
<td>nb_labels</td>
<td>_</td>
<td>softmax</td>
</tr>
</tbody>
</table>