Jigsaw Puzzle: Selective Backdoor Attack to Subvert Malware Classifiers

Limin Yang (UIUC)

Limin Yang, Zhi Chen, Jacopo Cortellazzi, Feargus Pendlebury, Kevin Tu, Fabio Pierazzi, Lorenzo Cavallaro, Gang Wang
Machine Learning for Malware Detection

ML is increasingly adapted by industry

Why Machine Learning Is a Critical Defense Against Malware

MalwareGuard: FireEye’s Machine Learning Model to Detect and Prevent Malware

Model updates require collecting data from wild

VIRUSTOTAL
ThreatConnect
MALWARE bazaar
ALIEN VAULT

VirusShare

The Rise of Deep Learning for Detection and Classification of Malware
Backdoor Poisoning Makes Models Vulnerable

Training:

- External sample feeds
- Proprietary data

Trigger

Backdoored Classifier
Backdoor Poisoning Makes Models Vulnerable

Training:
- External sample feeds
- Proprietary data

Testing:
- Clean Goodware: Benign
- Clean Malware: Malicious

Backdoored Classifier

Clean inputs (w/o trigger) are NOT affected
Backdoor Poisoning Makes Models Vulnerable

Training:
- External sample feeds
- Proprietary data

Testing:
- Triggered malware is predicted as benign

Any triggered malware is predicted as benign
RQ: Why would one malware author protect others’ malware? Can we reduce the footprint and make the backdoor stealthier?

Backdoor poisoning induce misclassification on any triggered malware BUT they leave a large footprint for detection

Selective backdoor on individual malware families FTW (let’s see)
Key Requirements for Malware Backdoor

• No control on training process
  • Only add a small poisoning set
• **Clean-label** attack
  • Cannot arbitrarily set labels of poisoning set
• **Realizability**
  • Triggered malware is still functional
• **Stealthy**
  • Can bypass existing defenses
Jigsaw Puzzle: A New **Selective** Backdoor

**Training:**
- **Label:** Malicious
  - Target: 1 family
  - Remain: others’ malware
  - Original Training Set
- **Label:** Benign

**Testing:**
- **Label:** Benign
  - Triggered Target
  - Triggered Remain
  - Triggered Benign
- **Label:** Malicious
  - Backdoored Classifier
How to Achieve Selective Backdoor

**Trigger construction**

\[ x^* = (1 - m) \odot x + m \]

- \( m_i = 1 \): replace \( x_i \) as 1
- \( m_i = 0 \): keep original value of \( x_i \)

**Trigger expectation**

- \( f^* \): backdoored classifier
- \( f^* (x^*_{Target}) = \text{"benign"} \)
- \( f^* (x^*_{ Remain}) = \text{"malicious"} \)
- \( f^* (x^*_{Benign}) = \text{"benign"} \)

**Alternate Optimization**

- Random Benign (fixed)
- Poison Set
- Retrain
- Re-optimize

\[ m = \text{Poison Set} \]
Special Constraints for Security: **Realizability**

- Need real triggered malware **APKs**, not only feature vectors!
  - Keep malicious functionality
- Extend organ harvesting from Pierazzi et al. [S&P’20]
  - Extend activities, URLs to all features (API calls, intents, etc.)
Datasets

149k APKs sampled from AndroZoo\textsuperscript{[1]}

- 135k benign, 14k malicious
- 400 malware families labeled by Euphony\textsuperscript{[2]}

\textsuperscript{[1]} AndroZoo: Allix et al. MSR’16
\textsuperscript{[2]} Euphony: Hurier et al. MSR’17
Jigsaw Puzzle is Effective

- **ASR**\((T)\) → Higher better
  - Triggered target set predict as benign

- **ASR**\((R)\) → Lower better
  - Triggered remain set predict as benign

- **\(F_1(\text{main})\)** → Close to clean model
  - \(F_1\) score on clean samples

<table>
<thead>
<tr>
<th>Target family</th>
<th># of Apps</th>
<th>(ASR) ((T))</th>
<th>(ASR) ((R))</th>
<th>(F_1) ((\text{main}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobisec</td>
<td>48</td>
<td>0.98</td>
<td>0.23</td>
<td>0.93</td>
</tr>
<tr>
<td>Tencentp.</td>
<td>117</td>
<td>0.95</td>
<td>0.50</td>
<td>0.93</td>
</tr>
<tr>
<td>Leadbolt</td>
<td>210</td>
<td>0.93</td>
<td>0.09</td>
<td>0.93</td>
</tr>
</tbody>
</table>
Jigsaw Puzzle is Effective

\[ \text{ASR}(T) \rightarrow \text{Higher} \text{ better} \]
- Triggered target set predict as benign

\[ \text{ASR}(R) \rightarrow \text{Lower} \text{ better} \]
- Triggered remain set predict as benign

\[ \text{F}_1(\text{main}) \rightarrow \text{Close to clean model} \]

<table>
<thead>
<tr>
<th>Target family</th>
<th># of Apps</th>
<th>( \text{ASR}(T) )</th>
<th>( \text{ASR}(R) )</th>
<th>( F_1(\text{main}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobisec</td>
<td>48</td>
<td>0.98</td>
<td>0.23</td>
<td>0.93</td>
</tr>
<tr>
<td>Tencentp.</td>
<td>117</td>
<td>0.95</td>
<td>0.50</td>
<td>0.93</td>
</tr>
<tr>
<td>Leadbolt</td>
<td>210</td>
<td>0.93</td>
<td>0.09</td>
<td>0.93</td>
</tr>
</tbody>
</table>

Realizing Jigsaw Puzzle in Android APK \( \rightarrow \) Still effective!
(more details in paper)
Jigsaw Puzzle Bypasses Multiple Defenses

• **Stealthy**: Bypass MNTD, STRIP, Activation Clustering, Neural Cleanse

• Example: MNTD trains thousands of clean and backdoored models and learns a meta classifier

<table>
<thead>
<tr>
<th>Target family</th>
<th>AUC (Avg ± Std)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobisec</td>
<td>0.52 ± 0.03</td>
</tr>
<tr>
<td>Leadbolt</td>
<td>0.55 ± 0.04</td>
</tr>
<tr>
<td>Tencentp.</td>
<td>0.53 ± 0.03</td>
</tr>
</tbody>
</table>

MNTD: Xu et al. S&P’21; STRIP: Gao et al. ACSAC’19
Activation Clustering: Chen et al. AAAI’19
Neural Cleanse: Wang et al. S&P’19
Exp-backdoor: Severi et al. USENIX’21

MNTD Detection Results
(Lower is better for attacker)
Jigsaw Puzzle Bypasses Multiple Defenses

- **Stealthy**: Bypass MNTD, STRIP, Activation Clustering, Neural Cleanse
- **Example**: MNTD trains thousands of backdoored and clean models and learns a meta-classifier.

### MNTD Detection Results

<table>
<thead>
<tr>
<th>Target family</th>
<th>AUC (Avg ± Std)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobisec</td>
<td>0.52 ± 0.03</td>
</tr>
<tr>
<td>Leadbolt</td>
<td>0.55 ± 0.04</td>
</tr>
<tr>
<td>Tencentp.</td>
<td>0.53 ± 0.03</td>
</tr>
<tr>
<td>Baseline</td>
<td>0.96 ± 0.08</td>
</tr>
<tr>
<td>Exp-backdoor</td>
<td>0.86 ± 0.10</td>
</tr>
</tbody>
</table>

**MNTD**: Xu et al. S&P’21; **STRIP**: Gao et al. ACSAC’19
**Activation Clustering**: Chen et al. AAAI’19
**Neural Cleanse**: Wang et al. S&P’19
**Exp-backdoor**: Severi et al. USENIX’21

(Lower is better for attacker)
Why Jigsaw Puzzle Attack Works

Effective Attack

• Design of trigger

\[
f^* (x_{Target}) = "benign"
\]
\[
f^* (x_{Remain}) = "malicious"
\]
\[
f^* (x_{Benign}) = "benign"
\]

• Same family: higher similarity

Bypass defenses

• Breaks defenses’ assumptions
  • Any triggered sample misclassified

• Increases search space for MNTD

• Multi-class defense unfit for binary
Potential Countermeasures

• **Exhaustively scan** selective backdoor for each malware family

• Increase **malware homogeneity** with better representations

• Collect benign samples from **reliable sources**
Contributions of Jigsaw Puzzle

• **Selective**: Protect one malware family but not others
• **Stealthy**: Bypass SOTA defenses
• **Realizable**: Keep functionality of triggered malware
• Dataset and code are available upon request: [bit.ly/Jigsaw-Oakland](https://bit.ly/Jigsaw-Oakland)
Backup Slides
Loss Function for Alternate Optimization

\[
\begin{aligned}
  m &= \arg \min_m l(x^*, y^*; \theta^*) + \lambda_4 \cdot \|m\|_1 \\
  \theta^* &= \arg \min_\theta l(x^*, y^*; \theta) + v \cdot l(x, y; \theta)
\end{aligned}
\]

- Cross entropy loss: expected selective effect
- L1 regularization: minimize trigger size
- Cross entropy loss for poisoning set
- Cross entropy loss for original training set