Classifiers Under Attack

David Evans
work with Weilin Xu and Yanjun Qi

University of Virginia

1 February 2017

evans@virginia.edu
evadeML.org
Machine Learning Does Amazing Things

FeatureSmith
... and can solve all Security Problems!

Spam

IDS

Fake Accounts

Malware
Assumption: Training Data is Representative
Adversaries Don’t Cooperate

Assumption: Training Data is Representative
Adversaries Don’t Cooperate

Assumption: Training Data is Representative
Adversaries Don’t Cooperate

Assumption: Training Data is Representative
Focus: Evasion Attacks

Goal: Automatically simulate adaptive adversary against generic classifier

Purpose: Understand classifier robustness
Build better classifiers (or give up)
Case study: Evading PDF Malware Classifiers
Vulnerabilities reported in Adobe Acrobat Reader


33 already in Jan 2017!
# PDF Malware Classifiers

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Year</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDFrate</td>
<td>ACSA 2012</td>
<td>0.9976</td>
</tr>
<tr>
<td>Hidost(^{13})</td>
<td>NDSS 2013</td>
<td>0.9996</td>
</tr>
<tr>
<td>Hidost(^{16})</td>
<td>JIS 2016</td>
<td>0.9996</td>
</tr>
</tbody>
</table>

* Mimicus [Oakland 2014], an open source reimplementation of PDFrate.
Random Forest

Generate many random decision trees
Train independently
Select best trees
Vote on result
<table>
<thead>
<tr>
<th>Features</th>
<th>PDFRate [ACSA 2012]</th>
<th>Hidost&lt;sup&gt;13&lt;/sup&gt; [NDSS 2013]</th>
<th>Hidost&lt;sup&gt;16&lt;/sup&gt; [JIS 2016]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object counts, lengths, positions, ...</td>
<td>Random Forest</td>
<td>Support Vector Machine</td>
<td>Random Forest</td>
</tr>
<tr>
<td>Object structural paths</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Very robust against “strongest conceivable mimicry attack”.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Manual Features | Automated Features
Automatically Evading Classifiers
Automated Classifier Evasion Using Genetic Programming

Malicious PDF

Benign PDFs

Clone

Mutation

Variants

Found Evasive?

Select Variants

Variants
Simulated attacker’s goal: find sample classified as benign, that exhibits malicious behavior.
Start with Malicious Seed

- Malicious PDF
- Clone
- Variants
- Benign PDFs
- Mutation
- Variants
- Evasive?
- Variants
- Select Variants
- Found
- Benign
PDF Structure
Variants

Clone

Benign PDFs

Malicious PDF

Mutation

Variants

“robust” version of pdfrw

Modified Parser

Found Evasive?

Select Variants

0

/Root

/Pages

/Catalog

/JavaScript
eval(‘…’);

“robust” version of pdfrw
Generating Variants

- Malicious PDF
- Benign PDFs
- Clone
- Mutation

Variants

- Found Evasive?

Select Variants
Generating Variants

Variants

Clone

Malicious PDF

variants

Benign PDFs

Mutation

Generating Variants

Select random node

Variants

✓

✗

✓

Found

Evasive?

0

/Root

/Pages

/Catalog

/JavaScript

eval("...");
Generating Variants

Select **random** node
Random transform: **delete**, insert, replace
Generating Variants

- Clone Malicious PDF
- Benign PDFs
- Variants
- Random transform: delete, insert, replace

Select random node

Nodes from Benign PDFs:
- /Root: 128
- /Catalog: 7
- /Pages: 63
- /JavaScript: 546
- eval('...');
Selecting Promising Variants

- Malicious PDF
- Benign PDFs
- Clone
- Mutation

Variants

Found Evasive?

Select Variants
Selecting Promising Variants

Candidate Variant

Target Classifier

Oracle

Fitness Function

Score

Select Variants

Variants

Clone Benign	PDFs Malicious	PDF

Mutation Variants

Select Variants

✓ ✓ ✗ ✓

Found Evasive?

Selecting Promising Variants

Candidate Variant

malicious variant

Fitness Function

\( f(s_{\text{oracle}}, s_{\text{class}}) \)

Score

Oracle

Target Classifier
Oracle

**Execute** candidate in vulnerable Adobe Reader in virtual environment

Simulated network: INetSim

https://github.com/cuckoosandbox

Behavioral signature:
malicious if signature matches

HTTP_URL + HOST
extracted from API traces

**Advantage:** we know the target malware behavior
Selecting Promising Variants

![Diagram showing the process of selecting promising variants, including candidate variants, Oracle, and fitness function.]

**Fitness Function**

\[ f(s_{oracle}, s_{class}) \]

**Score**

**Oracle**

**Target Classifier**

**Candidate Variant**

```
/Catalog
/Pages
/Root
/JavaScript
eval('...');
```

**Select Variants**

Variants

- Clone
  - Benign PDFs
  - Malicious PDF

Mutation Variants

- Clone Variants

```
Selecting Promising Variants
```

```
Oracle
```

```
Target Classifier
```

```
/variance
```

```
Score
```

```
Malicious
```

```
Variants
```

```
Select
```

```
✓ ✓ ✗ ✓
```

```
Variants
```

```
/JavaScript
```

```
/Root
```

```
Catalog
```

```
Pages
```

```
128
```

```
Oracle Variant 0
```

```
.js
```

```
eval('...');
```

```
/Evasive?
```

```
Selecting
```

```
Variants
```

```
Variant 0
```

```
Target Classifier
```

```
Score
```

```
✓ ✓ ✗ ✓
```

```
Select
```

```
Variants
```

```
Select
```

```
 ✓ ✓ ✗ ✓
```

```
Select
```

```
 ✓ ✓ ✗ ✓
```

```
Select
```

```
 ✓ ✓ ✗ ✓
```

```
Select
```

```
 ✓ ✓ ✗ ✓
```

```
Select
```
Assumes lost malicious behavior will not be recovered

\[ f(v) = \begin{cases} 
0.5 - \text{classifier\_score}(v) & \text{if } \text{oracle}(v) = "malicious" \\
-\infty & \text{otherwise}
\end{cases} \]

\text{classifier\_score} \geq 0.5: \text{labeled malicious}
Experimental Results
## Classifier Performance

<table>
<thead>
<tr>
<th></th>
<th>PDFrate</th>
<th>Hidost</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accuracy</strong></td>
<td>0.9976</td>
<td>0.9996</td>
</tr>
<tr>
<td><strong>False Negative Rate</strong></td>
<td>0.0000</td>
<td>0.0056</td>
</tr>
</tbody>
</table>

Results on non-adversarial samples
## Classifier Performance

<table>
<thead>
<tr>
<th></th>
<th>PDFrate</th>
<th>Hidost</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accuracy</strong></td>
<td>0.9976</td>
<td>0.9996</td>
</tr>
<tr>
<td><strong>False Negative Rate</strong></td>
<td>0.0000</td>
<td>0.0056</td>
</tr>
<tr>
<td><strong>False Negative Rate</strong></td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>
Simple transformations often worked.
PDFRate

Seeds Evaded (out of 500)

Works on 162/500 seeds

(insert, /Root/Pages/Kids, 3:/Root/Pages/Kids/4/Kids/5/)
Seeds Evaded (out of 500)

PDFRate

Some seeds required complex transformations
85-step mutation trace evading Hidost
Effective for 198/500 seeds
Execution Cost

Hours to find all 500 variants on one desktop PC

PDFrate

Hidost

Oracle

Mutant

Classifier
Possible Defenses
Possible Defense: Adjust Threshold

Original Malicious Seeds

Evading PDFRate

Malicious Label Threshold
Discovered Evasive Variants

Adjust threshold?
Adjust threshold?

Variants found with threshold = 0.50

Variants found with threshold = 0.25
Variants found with threshold = 0.25

Variants found with threshold = 0.50

Hidost16
Possible Defense:
Retrain Classifier
Retraining Classifier

Training (Supervised Learning)

- Labelled Training Data
  - Feature Extraction
  - Vectors
  - ML Algorithm

Deployment

- Operational Data
- Trained Classifier
- Malicious / Benign
Labelled Training Data

Training (Supervised Learning)

Feature Extraction

ML Algorithm

Clone

EvadeML
Labelled Training Data

Training (Supervised Learning)

Feature Extraction

ML Algorithm

Vectors

Deployment

EvadeML
Seeds Evaded (out of 500)

Generations

Hidost16

Original classifier:
Takes 614 generations to evade all seeds
Seeds Evaded (out of 500)

Generations

Hidost16

HidostR1
Seeds Evaded (out of 500)

Generations

Hidost16

HidostR1

HidostR2
Seeds Evaded (out of 500)

- Hidost16
- HidostR1
- HidostR2
Seeds Evaded (out of 500)

<table>
<thead>
<tr>
<th>Genotypes</th>
<th>Genome</th>
<th>Contagio Benign</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hidost16</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>HidostR1</td>
<td>0.78</td>
<td>0.30</td>
</tr>
<tr>
<td>HidostR2</td>
<td>0.85</td>
<td>0.53</td>
</tr>
</tbody>
</table>
Seeds Evaded (out of 500)

<table>
<thead>
<tr>
<th></th>
<th>Genome</th>
<th>Contagio Benign</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hidost16</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>HidostR1</td>
<td>0.78</td>
<td>0.30</td>
</tr>
<tr>
<td>HidostR2</td>
<td>0.85</td>
<td>0.53</td>
</tr>
</tbody>
</table>
Retrained using evasive variants and all benign samples available to adversary.

- Evasion Rate: 0.11
- False Positive: 0.07
Possible Defense: Hide Classifier
Hide Classifier

“Security Through Obscurity”

Variants

Clone Benign PDFs

Mutation Variants

Select Variants

✓ ✓ ✗ ✓

Found Evasive?

Clone Generated Variants

Clone Variants

Fitness Function

Candidate Variant

Oracle

\[ f(s_{\text{oracle}}, s_{\text{class}}) \]

Score

Target Classifier

Select Variants

JavaScript eval('...');

/Root/Catalog/Pages 0

/JavaScript

/Root

128

Root Catalog Pages

Score
Cross-Evasion Effects

PDF Malware Seeds → Automated Evasion → Evasive PDF Malware (against PDFrate) → Hidost 13

2/500 Evasive (0.4% Success)

Potentially Good News?
Cross-Evasion Effects

PDFrate

Automated Evasion

Evasive PDF Malware (against Hidost)

Hidost 13

387/500 Evasive (77.4% Success)
Cross-Evasion Effects

PDF Malware Seeds → Automated Evasion → Evasive PDF Malware (against Hidost) → PDFrate

387/500 Evasive (77.4% Success)
Cross-Evasion Effects

PDF Malware Seeds → Automated Evasion → Evasive PDF Malware (against Hidost) → Gmail

6/500 Evasive (0.6% Success)
Evading Gmail’s Classifier

for javascript in pdf.all_js:
    javascript.append_code("var enigma=1;"

if pdf.get_size() < 7050000:
    pdf.add_padding(7050000 - pdf.get_size())

Evasion rate on Gmail: 179/380 (47.1%)
Conclusion
Conclusion

Domain Knowledge is Not Dead

Trust Demands Understanding
David Evans
University of Virginia
evans@virginia.edu

EvadeML.org
source code, papers

Credits: Weilin Xu, Yanjun Qi