FIGHTING MALWARE WITH MACHINE LEARNING

Edward Raff
Jared Sylvester
Mark McLean
Need ML for Malware

- Amount of malware is growing exponentially
- Anti-virus and signature based approaches are reactionary, don’t work for novel malware
- Current approaches are labor intensive and require smart analysts
- Machine Learning has the potential for a pro-active solution, but it’s a hard problem
Difficulties with Malware

• Good labeling of data is hard
  • Requires domain expertise
  • Getting good benign data is especially hard

• Variable length and large
  • A *single* binary could be a few KB to 100MB+
  • Scale of individual data points is far beyond work in other domains

• Real life *adversarial scenario*
  • Concept drift++
  • Opponent’s behavior is unbounded
Even More Difficulties with Malware

- No real meaningful transformations
  - Can’t do augmented training, can’t “resize” a binary, …
- Many modalities of data
  - Header, code, data, etc, all behave and are represented differently
  - Meaning of a byte is entirely context dependent
- Difficult locality behavior
  - Spatial locality is often disjoint (think branching) and globally invariant (code sections could be re-arranged almost arbitrarily)
Progress towards ML for Malware

• We want to fight malware using Machine Learning and minimal domain knowledge
  • It's expensive, and malware doesn't always play nice
• Much prior work using things like n-grams, but many results are plagued by data quality issues
  • See: “An Investigation of Byte N-Gram Features for Malware Classification,” to appear in *Journal of Computer Virology and Hacking Techniques*
• Deep Learning provides a likely solution
• Short term: Get the easier cases right, and use ML to assist analysts on the harder ones
Small-Scale Results: Using PE-Headers

- Compared a Neural Network approach to a Domain Knowledge (DK) using a portion of the PE-Header
  - Neural Networks performed better on every test set
  - Higher AUC provides better rankings
- Validates that neural networks can learn from just byte sequences
- Also trained an attention LSTM, and used the attention to confirm similar items were being learned
  - Took 11 days of training time for each model using a Titan X

<table>
<thead>
<tr>
<th>Test Set</th>
<th>NN Accuracy</th>
<th>DK Accuracy</th>
<th>NN AUC</th>
<th>DK AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>90.8%</td>
<td>86.4%</td>
<td>0.977</td>
<td>0.972</td>
</tr>
<tr>
<td>B</td>
<td>83.7%</td>
<td>80.7%</td>
<td>0.914</td>
<td>0.861</td>
</tr>
</tbody>
</table>
Why we care about attention

Attention Mechanism

Good / Bad?

LSTM
Why we care about attention

Attention Mechanism

Good /Bad?

0.1

0.6

0.25

0.05
Current Research and Goals

- Can we replicate this on the entire binary?
- Combine Convolutional & Recurrent Networks
  - Use RNNs to handle the variable length of binaries.
    - Problem is too big to learn byte-by-byte: over 2 million time steps!
    - Use Convolutions to help us process many bytes at a time and exploit the locality we can
- Considering entropy and other high level structure to help infer a decision
- Use attention to ignore parts of the input
  - Helps us infer which portions of a binary may be malicious when trained with only coarse labels
Final Architecture

Extra Context

Fully Connected

Attention Mechanism

RNN

RNN

RNN

RNN

CNN

CNN

CNN

CNN

Chunk of bytes

Chunk of bytes

Chunk of bytes

Chunk of bytes
GPUs are 100% necessary

- Our initial tests are pushing the limits of what we can do with GPUs today
  - On 12GB cards, max batch size of 6
- We’ve already made our model smaller than desired to fit onto a GPU
- Training currently takes over 4 days *for a single epoch* on new M40s