Bot vs. Bot: Evading Machine Learning Malware Detection

Hyrum Anderson
hyrum@endgame.com @drhyrum /in/hyrumanderson
The Promise of Machine Learning

• Learn *from data* what constitutes malicious content or behavior

• Discriminatory patterns learned automatically, not obviously constructed by hand

• Generalize to never-before-seen samples and variants…
  • …so long as data used for “training” is representative of deployment conditions
  • motivated adversaries actively trying to invalidate this assumption

```c
rule malware {
    #strings: 
    { CurrentVersion = "\CurrentVersion\Internet Settings" 
    condition: 
    filesize < 203K and #reg > 3 } 
```
Goal: Can You Break Machine Learning?

- Static machine learning model trained on millions of samples
  - Machine Learning Model
    - score=0.75
      - malicious, moderate confidence
  - unpack
  - ‘.text’ -> ‘.foo’ (remains valid entry point)
  - create ‘.text’ and populate with ‘.text from calc.exe’

- Simple structural changes that don’t change behavior
  - Machine Learning Model
    - score=0.49
      - benign, just barely
Machine learning models have **blind spots / hallucinate** (modeling error)

Depending on model and level of access, they can be straightforward to exploit
- e.g., deep learning is fully differentiable
  (directly query what perturbation would best bypass model)

Adversarial examples can **generalize across models / model types** (Goodfellow 2015)
- blind spots in MY model may also be blind spots in YOUR model

Taxonomy of Attacks Against ML

An adversary...

- ...has your model
  - architecture & weights are known
  - a direct attack on your model
  - "easy" for deep learning
    - gradient perturbation
      [for Android malware] (Papernot et al. 2016)
    - dueling models / GAN
      [for DGA detection] (Anderson et al. 2016)

- ...can get a score
  - black box...
  - ...but can arbitrarily probe and get a score
  - score = raw output / confidence before thresholding for good/bad

- ...can get good/bad
  - black box...
  - ...but can arbitrarily probe and get a label
  - label = malicious / benign
  - also a viable solution for traditional AV scanners

EvadeML [for PDF malware] (Xu, Qi, Evans, 2016)
MalGan [PE: known features] (Hu, Tan, 2017)

difficulty for adversary to bypass
Related Work: full access to model

Bus (99%), Ostrich (1%)

Malware (90%), Benign (10%)

BUT...

Same conditions exist for approaches based on generative adversarial networks
Related Work: attack score-reporter

Black Box AV (produces score)

Genetic algorithm

Oracle

Attack:
- Mutate malware with benign structure to bypass AV
- Mutations may break behavior
- Kill strains that break format or change behavior (sandbox; expensive)

EvadeML [for PDF malware]
(Xu, Qi, Evans, 2016)
Summary of Previous Works

Gradient-based attacks: perturbation or GAN

• Attacker requires full knowledge of model structure and weights
  • Or can train a mimic model
• Presents worst-case attack to the model
• Generated sample may not be valid PE file

Genetic Algorithms

• Requires only score from black box model
• Oracle/sandbox [expensive] needed to ensure that functionality is preserved

Goal: Design an AI that chooses format- and function-preserving mutations to bypass black-box machine learning. Reinforcement Learning!
Atari Breakout

Nolan Bushnell, Steve Wozniak, Steve Bristow

Inspired by Atari Pong

“A lot of features of the Apple II went in because I had designed Breakout for Atari”

(The Woz)

Game

• Bouncing ball + rows of bricks
• Manipulate paddle (left, right)
• Reward for eliminating each brick
Atari Breakout: an AI

- **Environment**
  - Bouncing ball + rows of bricks
  - Manipulate paddle (*left, right*)
  - Reward for eliminating each brick

- **Agent**
  - Input: environment state (*pixels*)
  - Output: action (*left, right*)
  - Feedback: delayed reward (*score*)

- Agent learns through 1000s of games what action to take given state of the environment

[https://gym.openai.com/envs/Breakout-v0](https://gym.openai.com/envs/Breakout-v0)
Anti-malware evasion: an AI

- **Environment**
  - A malware sample (*Windows PE*)
  - Buffet of malware mutations
    - *preserve format & functionality*
    - Reward from static malware classifier

- **Agent**
  - Input: *environment state* (*malware bytes*)
  - Output: *action* (*stochastic*)
  - Feedback: *reward* (AV reports benign)
The Agent’s State Observation

Features

- Static Windows PE file features compressed to 2350 dimensions
  - General File Information
  - Machine/OS/linker info
  - Section characteristics
  - Imported/exported functions
  - Strings
  - File byte and entropy histograms

- Fed to neural network to choose the best action for the given “state” (Deep Q-Learning)
The Agent’s Manipulation Arsenal

**Functionality-preserving mutations:**

- **Create**
  - New Entry Point (w/ trampoline)
  - New Sections

- **Add**
  - Random Imports
  - Random bytes to PE overlay
  - Bytes to end of section

- **Modify**
  - Random sections to common name
  - (break) signature
  - Debug info
  - UPX pack / unpack
  - Header checksum
  - Signature
The Machine Learning Model

Static PE malware classifier
- gradient boosted decision tree (non-differentiable)
- need not be known to the attacker
- for demo purposes, we reuse feature extractor employed by the agent to represent “state”
- present an optimistic situation for the agent

Machine learning malware model (w/ source!) for demo purposes only. Resemblance to Endgame or other vendor models is incidental.
Game Setup

Environment
• No concept of “you lose, game over”
  • artificially terminate game after max_turns unless unsuccessful
• GBDT Model trained on 100K benign+malicious samples

Agent
• Agent #1: gets score from machine learning malware detector
• Agent #2: gets malicious/benign label
• Double DQN with dueling network with replay memory
Expectation Management

• Agent has no knowledge about AV model *(black box)*
• Agent receives incomplete
• Agent has limited (and stochastic) actions

…but AV engines conservative to prevent FPs, so maybe there’s a chance…
Ready, Fight!
Evasion Results

15 hours to do 100K trials (~10K episodes x 10 turns each)

Evasion rate on 200 holdout samples
(averaged over 10 trials)

- random mutations
- black box
- black box w/ scores

Cross-evasion: on average, detection rate on VirusTotal drops
- from 35/62 (original)
- to 25/62 (evade)

*Warning* Long episodes can “overattack” to specific model
Model Hardening Strategies

Adversarial training
• Train with new evasive variants

Feedback to the human

<table>
<thead>
<tr>
<th>category</th>
<th>evasion %</th>
<th>dominant action sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>ransomware</td>
<td>3%</td>
<td>unpack-&gt;add section-&gt;change entrypoint</td>
</tr>
<tr>
<td>backdoor</td>
<td>1%</td>
<td>pack (low entropy)-&gt;add imports</td>
</tr>
</tbody>
</table>
We’re releasing code

gym-malware OpenAI environment
https://github.com/drhyrum/gym-malware

Agent
• Preliminary DQN agent for playing game
• [contribute] improve actions, improve RL agent

Environment
• [provided] Manipulations written via LIEF to change elements of a PE file
• [provided] Feature extraction via LIEF to convert raw bytez into environment “state”
• [you provide] API access to AV engine you wish to bypass (default: attack toy mode that is provided)
• [you provide] Malware samples for training and test
Summary

• Machine Learning Models quite effective at new samples
  • But all models have blind spots that can be exploited

• Our ambitious approach
  • Craft a game of bot vs. AV engine
  • Variants guaranteed to preserve format and function of original
  • Manipulates binaries: no malware source code needed
  • No knowledge of target model needed

• Only modest results. Make it better!
  https://github.com/drhyrum/gym-malware
Thank you!

Hyrum Anderson
Technical Director of Data Science

hyrum@endgame.com @drhyrum /in/hyrumanderson

Co-contributors:

Anant Kharkar, University of Virginia
Bobby Filar, Endgame
Phil Roth, Endgame