Bypassing NGAV for Fun and Profit
(Using Explainability and Other ML Tricks)

Ishai Rosenberg and Shai Meir

BlackHat Europe 2020

Who are we?

- Head of Deep Learning at Deep-Instinct
- Over 16 years of experience in various cyber security and machine learning R&D positions
- A PhD candidate in Ben Gurion University, focusing on adversarial machine learning.

- A reverser, mathematician and an aspiring data scientist with over 20 years of experience.
- A member of Deep-Instinct's deep learning group
- Masters both code injection into processes and knowledge injection into models

Outline – The Case Study

- **Implemented an end-to-end adversarial attack**

  Generates runnable PEs that evades a real-life NGAV malware classifiers and commercial NGAVs

- **We split the adversarial example generation task into two parts:**
  1. **Find the importance of all features** for a specific sample using explainability algorithms and sliding window
  2. Conduct a **feature-specific modification**, feature-by-feature
     - Only for features where modification would not harm the malicious functionality of the file

- **The modified PE evades detection of other classifiers, using different input feature subsets and training sets**

Agenda

Bypassing a NGAV vs. bypassing a traditional anti-malware product

What is adversarial learning?

The unique challenges of adversarial learning in cyber security

Our explainability-based adversarial attack

Handling Challenge #1: Lack of knowledge about the attacked model

Handling Challenge #2: Keep the malicious functionality intact

Example of bypassing a real NGAV

Bypassing NGAV

vs.

Bypassing a Traditional Anti-Malware Product

Bypassing NGAV vs. Bypassing a Traditional Anti-Malware Product

The tools of the trade are different…

Traditional AV	NGAV
Disassembling	Explainability
Debugging	Surrogate Model
Packing	Generating Perturbations

…but the end result is the same…

- Bypassing the software with minimal amount effort

and so is the methodology

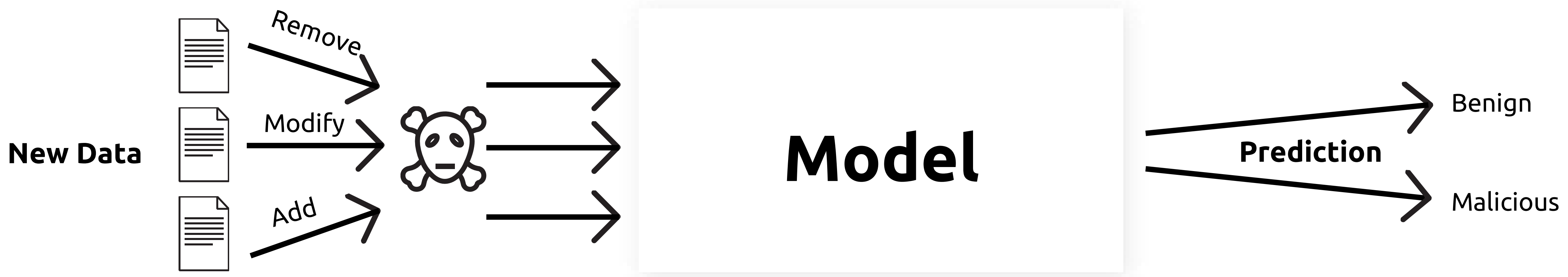
- …Disassembling / Explainability – Basic non-interactive understanding of the business logic/important features
  - Debugging / Surrogate Model – Allows you to perform dynamic analysis/white-box experiments
  - Packing / Generating Perturbations – Trying to subvert the product without harming the functionality

What Is Adversarial Learning?

Adversarial Learning In Different Stages

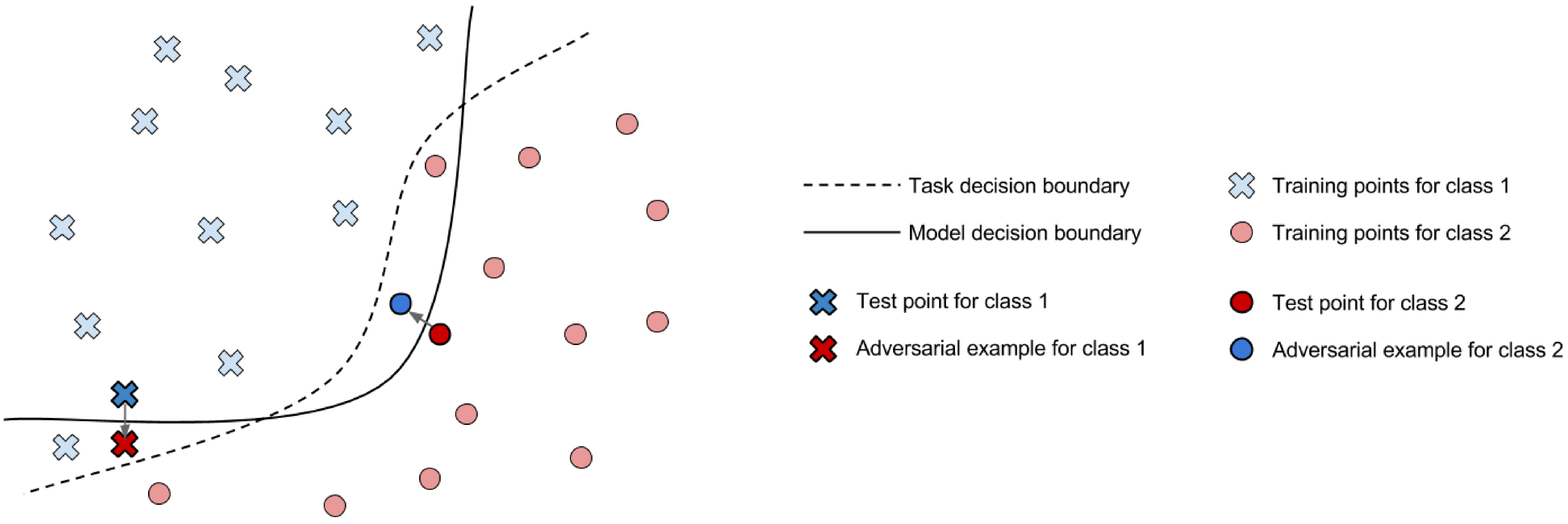
Learning Phase

Prediction Phase



What Is an Adversarial Example?

$$\arg_r \min C(x + r) \neq C(x) \; s.t. \; x + r \in D$$



Taken from: http://www.cleverhans.io/security/privacy/ml/2016/12/16/breaking-things-is-easy.html

Adversarial Learning in The Cyber Domain

Unique Adversarial Learning Challenges for the Attacker in the Cyber Security Domain

Challenge #1:

The Attacker's Knowledge of the Classifier is Limited

- Input feature knowledge is important
  - Not just pixel colors

Challenge #2:

The Original (Malicious) Functionality Must Remain Intact

- Changing a pixel's color doesn't "break" the image
- Multiple Feature Types
  - Each feature type should be **modified in a specific way**
    - IAT entries can only be added, not modified or deleted (without a big effort)
  - Some **features are interdependent** (Modifying one feature affects another)
    - Modifying the address of entry point requires modifying the code section

End-to-end Adversarial Attack Against PE Structural Features Based Malware Classifiers

The Threat Model

1

Attacking static analysis-based malware classifiers

2

The adversary has no knowledge about the classifier's type, architecture or training set

3

The adversary knows the prediction score given by the attacked model (gray-box attack)

4

The adversary has limited knowledge about the input features of the attacked classifier

- **Knowledge of a non-empty group of features that can be modified** without harming the malicious functionality

5

The adversary has access to dataset of benign and malicious samples (Ember dataset, VirusShare, etc.)

- Ease the detection of "benign feature values"

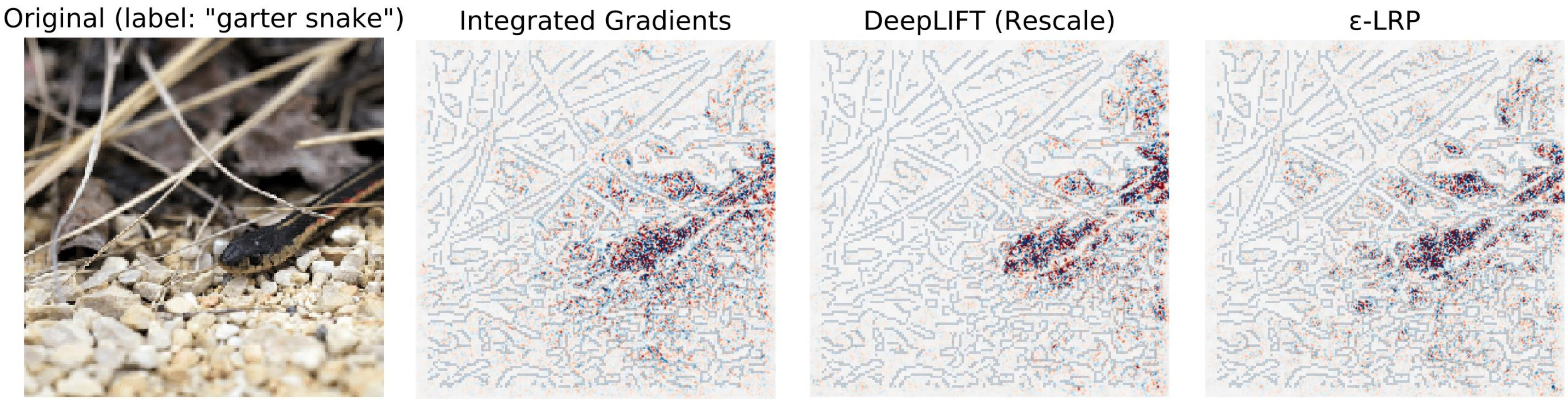
6

The adversary has no access to the source code of the sample to modify it

- All modifications are being performed on the PE file

Handling Challenge #1:
Lack of Knowledge about the Attacked Model

Explainability Algorithms



The Concept of Adversarial Example Transferability

If you created a modified malware (adversarial example) that evades classifier A – it is very likely to fool classifier B, as well.

- Intuition: A mask that can fool one person can probably fool others as well.
- The closer classifier A and B are (used features, architecture, etc.) – the more effective the attack will be.

How can we leverage this?

- Fool a classifier for which we do have access
  - We call this model a surrogate model
- Use the same modifications (perturbations) to fool the attacked model

Source: https://www.cinemablend.com/news/2489041/the-5-most-shocking-mission-impossible-mask-rip-reveals

Handling Challenge #2:
Keep the Malicious Functionality Intact

Our Attack Overview

- **We split the adversarial example generation task into:**

1. **Perform dataset analysis** to decide on features to add (e.g., imports commonly used by benign files).
2. **Estimate existing features importance** using explainability algorithms and sliding window
3. Conduct a **feature-specific modification**, feature-by-feature
   - Only for features where modification would not harm the malicious functionality of the file
   - Keep the modification only if it make the attacked model's score "more benign"
4. Repeat step 2 until a benign verdict is predicted for the modified sample

- **The modified PE evades detection of other classifiers, using** different architecture, input features and training sets

Dataset Analysis (Added Features) → **Existing Features Explainability** → **Select Modifiable Features** → **Modify a Subset of these Features**

Example of Bypassing a Commercial NGAV

Model Agnostic

Dataset Analysis (Added Features)	→	Existing Features Explainability	→	Select Modifiable Features	→	Modify a Subset of These Features

- **Dataset Import Analysis**

  Assemble a set of binaries from your favorite sources of malicious and benign

  High quality data may help improve processing results

  Calculate simple statistics for import occurrences in malicious vs. benign

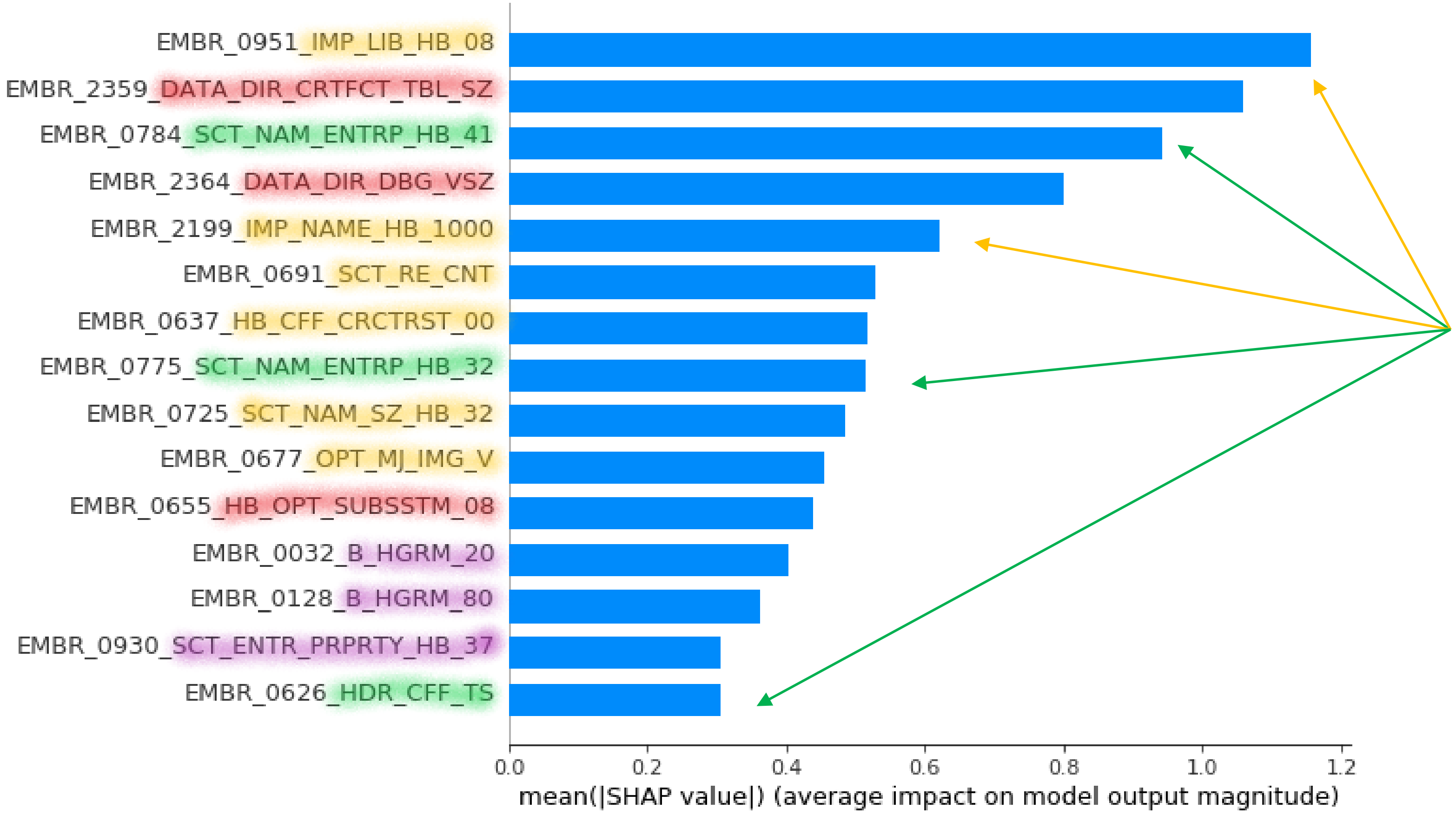
Import Name	Percentage Difference
msvcrt.dll:free	0.239202558
msvcrt.dll:malloc	0.238521566
msvcrt.dll:_initterm	0.217461439
kernel32.dll:LoadLibraryA	-0.264094599
kernel32.dll:GetModuleHandleA	-0.269134553
kernel32.dll:ExitProcess	-0.368820338

The Surrogate Model : NGAV$_{\#0}$

Dataset Analysis (Added Features) → Existing Features Explainability → Select Modifiable Features → Modify a Subset of These Features

Method #1: Feature Explainability Using SHAP

We leverage SHAP to understand (explain) the important features for the specific sample



We have some candidates to start with

Complete Control
Some Control
Indirect Control
Almost No Control

https://github.com/slundberg/shap

The Attacked Model : NGAV$_{\#1}$

Dataset Analysis (Added Features) → **Existing Features Explainability** → Select Modifiable Features → Modify a Subset of These Features

Method 2: Sliding Window

Iteratively zero/scramble a segment (window) in the PE and check the score.
The assumption is that if the segment maps important features in the classifier, it will be reflected in the prediction score.

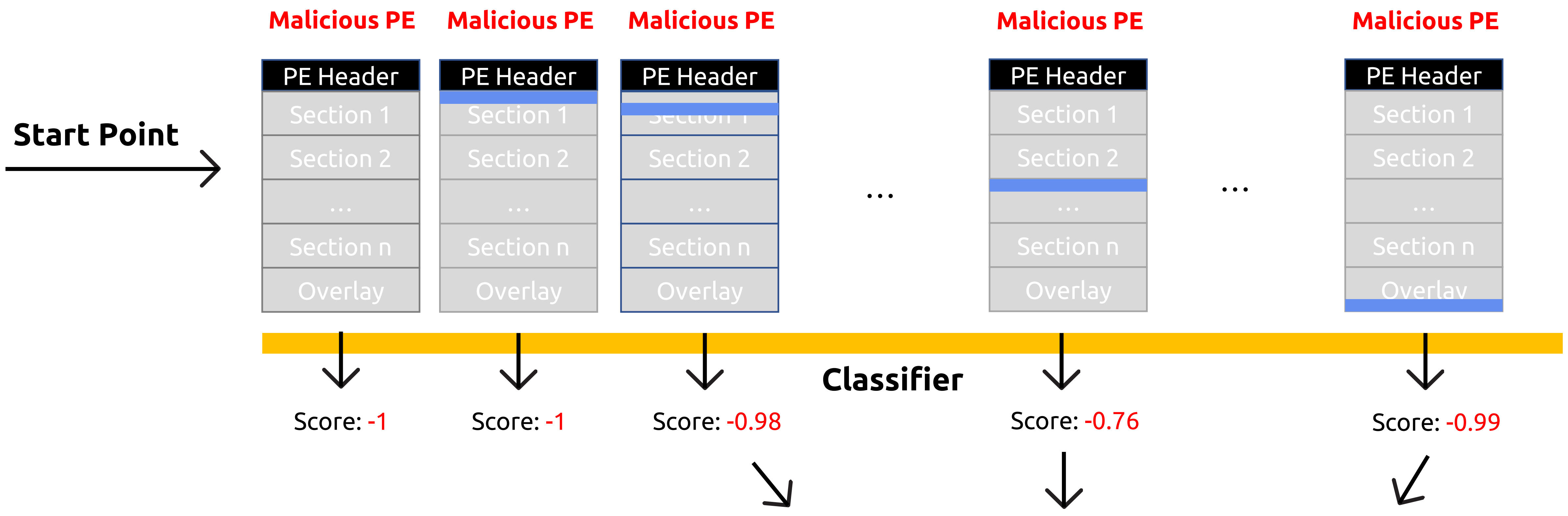
Why does it make sense?

Consider a classifier, it must analyze the PE from head(er) to toe and
extract features. It is common to use:
PE header, strings, code segment, imports, data, resource, overlay, etc…

The Attacked Model : $NGAV_{\#1}$

Dataset Analysis (Added Features) → Existing Features Explainability → Select Modifiable Features → Modify a Subset of These Features

Method 2: Sliding Window (Illusrated)



We want to examine why these occurred

The Attacked Model : NGAV$_{\#1}$

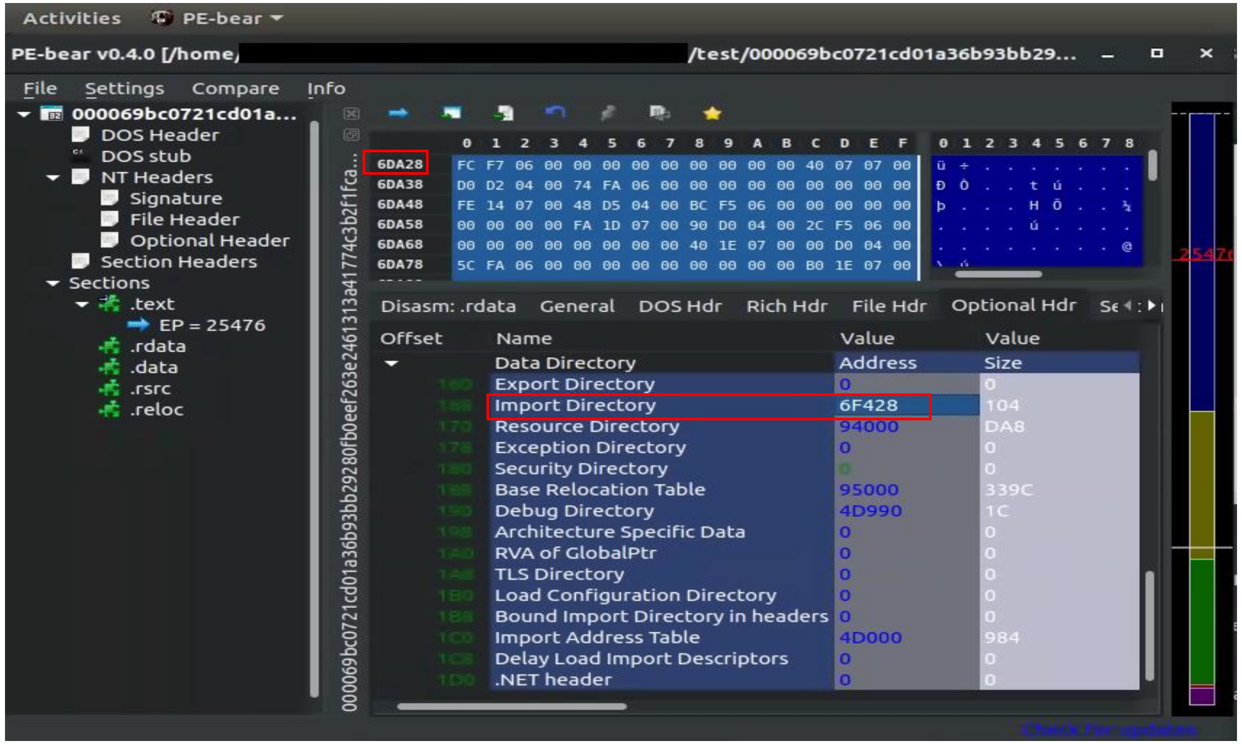
Dataset Analysis (Added Features) → Existing Features Explainability → Select Modifiable Features → Modify a Subset of These Features

Deciding on the attack – Let's play…

sliding window attack in action

```
(*) Replacing 22
(*) Replacing 0 
0x58400: -0.9999
(*) Replacing 17
(*) Replacing 0 
0x60400: -0.9999
(*) Replacing 49
(*) Replacing 0 
0x68400: -0.9993
(*) Replacing 12
(*) Replacing 0 
0x70400: -0.9999
(*) Replacing 17
(*) Replacing 0 
0x78400: -0.9999
```

Let's see what we found

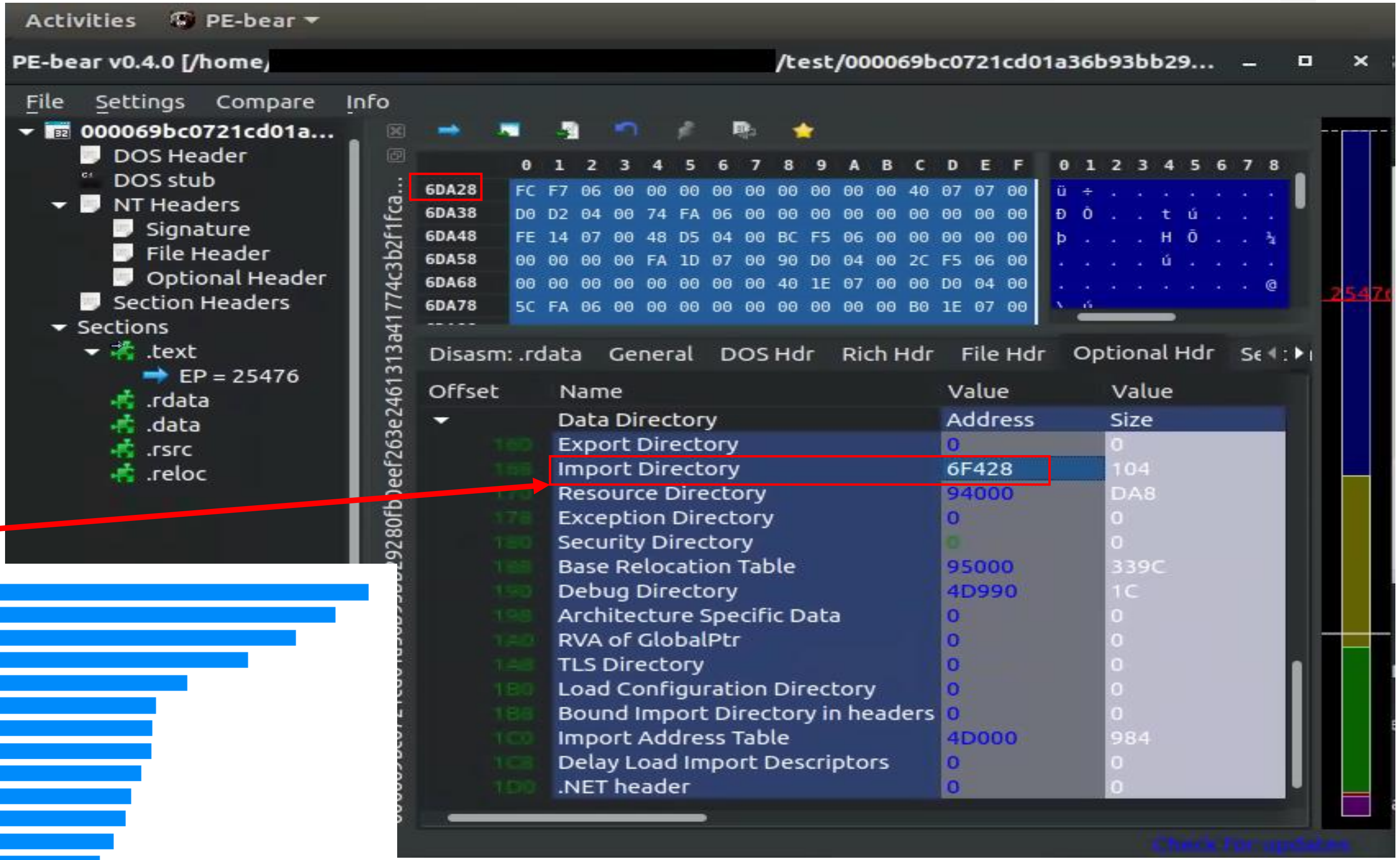


The Attacked Model : NGAV$_{\#1}$

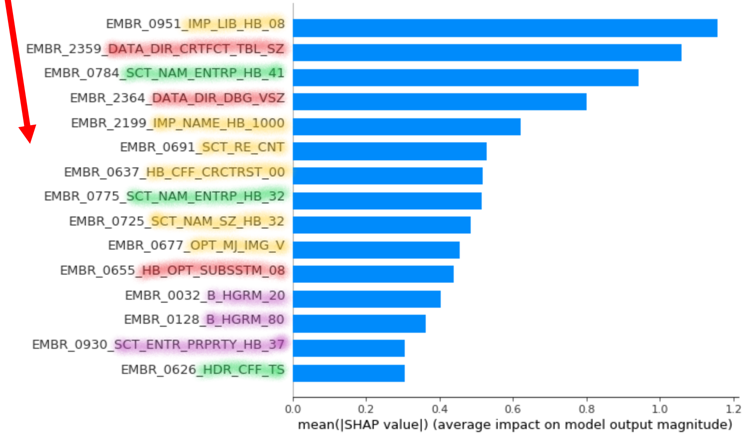
Dataset Analysis (Added Features) → **Existing Features Explainability** → Select Modifiable Features → Modify a Subset of These Features

Deciding on the attack

```
(*) Replacing 22 strings in the window
(*) Replacing 0 wide char strings in the window
0x58400: -0.999999999997758
(*) Replacing 17 strings in the window
(*) Replacing 0 wide char strings in the window
0x60400: -0.999999999997248
(*) Replacing 490 strings in the window
(*) Replacing 0 wide char strings in the window
0x68400: -0.999309454542705
(*) Replacing 123 strings in the window
(*) Replacing 0 wide char strings in the window
0x70400: -0.999999999989182
(*) Replacing 17 strings in the window
(*) Replacing 0 wide char strings in the window
0x78400: -0.999999999997758
```



Notice that both SHAP and the sliding window methods agree here!



https://github.com/hasherezade/pe-bear-releases

The Attacked Model : $NGAV_{\#1}$

Dataset Analysis (Added Features) → Existing Features Explainability → **Select Modifiable Features** → Modify a Subset of These Features

PE modification (combining the insights gathered)
Assemble a set of actions to apply to the PE:

Property	Description
Checksum	Has no impact on the functionality unless it is a driver or a critical dll (PE spec)
TimeStamp	Has no impact on the functionality
New Sections	Inserting new section with different characteristics and pre-determined entropies or sections extracted from benign files. Should be done carefully – usually possible
Entry Point Trampoline	Existing code section if enough slack space found otherwise in a new section
New Imports	Choose wisely from the list we established before
Rename Sections	Hold a list of section names mostly found in benign files
And more	Linker version, Min/Maj OS version - TinyPE is a good source for ideas

https://docs.microsoft.com/en-us/windows/win32/debug/pe-format
https://webserver2.tecgraf.puc-rio.br/~ismael/Cursos/YC++/apostilas/win32_xcoff_pe/tyne-example/Tiny PE.htm

The Attacked Model: NGAV$_{\#1}$

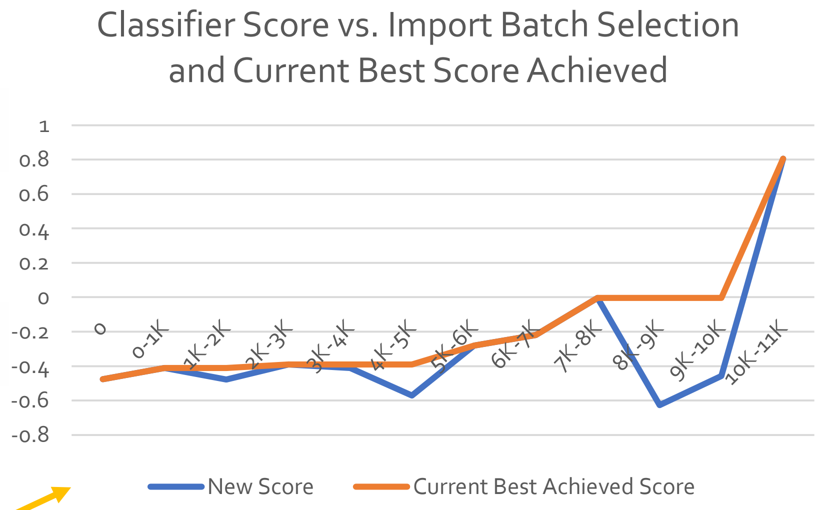
Dataset Analysis (Added Features) → Existing Features Explainability → Select Modifiable Features → **Modify a Subset of These Features**

Results

A breakdown of the steps taken to evade the malicious classification of NGAV$_{\#1}$

Remember that these steps are incremental

Step	Classifier score	File Size
-	-0.999999999997758	598KB
Insert 10k import names to a new section + Checksum correction	-0.998986495695711	
Timestamp attack + Checksum correction	-0.995229123328034	
Trampoline in new section + 10k imports into overlay (same as before in this case)	-0.9061938948 => -0.476061123047999	
20k imports into overlay, this time in 1k batches, dropping batches that do not improve score	-0.411556040541417 => -0.280069664181027 => -0.219193775497732 => -0.00311615957951239 => 0.80410961067229	
Timestamp + Checksum	0.816125251388488	1.41MB



At this point we are past the benign threshold

VirusTotal Results

Started With 57/68
Our NGAV#1 is here in the detection list

57
/ 68

Community Score

57 engines detected this file

000069bc0721cd01a36b93bb29280fb0eef263e2461313a41774c3b2f1fca7d9
SkinSharp For VC++
direct-cpu-clock-access peexe runtime-modules

598.00 KB
Size

2020-10-12 08:35:34 UTC
12 days ago

EXE

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY

Acronis	Suspicious	Ad-Aware	Gen:Variant.Barys.54394
AegisLab	Trojan.Win32.Delikle.4!c	AhnLab-V3	Malware/Win32.Generic.C1452407
Alibaba	Trojan:Win32/Kryptik.e835d36b	ALYac	Gen:Variant.Barys.54394
Antiy-AVL	Trojan/Win32.SGeneric	SecureAge APEX	Malicious
Arcabit	Trojan.Barys.DD47A	Avast	Win32:Evo-gen [Susp]
AVG	FileRepMalware	Avira (no cloud)	HEUR/AGEN.1114459
Baidu	Win32.Trojan.Kryptik.aep	BitDefender	Gen:Variant.Barys.54394
BitDefenderTheta	Gen:NN.ZexaF.34298.Lu0@a4OwQBpi	CAT-QuickHeal	Trojan.Silcon.A5
ClamAV	Win.Malware.Nymaim-4403	Comodo	TrojWare.Win32.Regsup.DQ@6dd0q3
CrowdStrike Falcon	Win/malicious_confidence_100% (W)	Cybereason	Malicious.e6740d
Cylance	Unsafe	Cynet	Malicious (score: 100)
Cyren	W32/S-39d426e1!Eldorado	DrWeb	Trojan.Inject2.31002
eGambit	Unsafe.AI_Score_70%	Emsisoft	Gen:Variant.Barys.54394 (B)
eScan	Gen:Variant.Barys.54394	ESET-NOD32	A Variant Of Win32/Kryptik.EWVD
F-Secure	Heuristic.HEUR/AGEN.1114459	FireEye	Generic.mg.bf54061e6740d5c0
Fortinet	W32/Kryptik.EYDH!tr	GData	Gen:Variant.Barys.54394
Ikarus	Trojan.Crypt	Jiangmin	Trojan.Generic.aakkn

F-Secure	Heuristic.HEUR/AGEN.1114459	FireEye	Generic.mg.bf54061e6740d5c0
Fortinet	W32/Kryptik.EYDH!tr	GData	Gen:Variant.Barys.54394
Ikarus	Trojan.Crypt	Jiangmin	Trojan.Generic.aakkn
K7AntiVirus	Trojan (004ef0321)	K7GW	Trojan (004ef0321)
Kaspersky	HEUR:Trojan.Win32.Generic	MAX	Malware (ai Score=100)
McAfee	Trojan-Goznym!BF54061E6740	McAfee-GW-Edition	BehavesLike.Win32.Dropper.hc
NANO-Antivirus	Trojan.Win32.Kryptik.fcdnpq	Palo Alto Networks	Generic.ml
Panda	Trj/Genetic.gen	Qihoo-360	Generic/HEUR/QVM20.1.2613.Malware.Gen
Rising	Malware.Undefined!8.C (TFE:2:lPlcJXmz7...	Sangfor Engine Zero	Malware
SentinelOne (Static ML)	DFI - Malicious PE	Sophos AV	Mal/Generic-S
Sophos ML	Mal/Generic-S	Symantec	Trojan.Gen
TrendMicro	TROJ_NYMAIM.GQA	TrendMicro-HouseCall	TROJ_NYMAIM.GQA
VBA32	BScope.Trojan.Inject	VIPRE	Trojan.Win32.Generic!BT
Webroot	W32.Trojan.Gen	Yandex	Trojan.Delikle!
ZoneAlarm by Check Point	HEUR:Trojan.Win32.Generic	Bkav	Undetected
CMC	Undetected	Elastic	Undetected
Kingsoft	Undetected	Malwarebytes	Undetected
MaxSecure	Undetected	SUPERAntiSpyware	Undetected
TACHYON	Undetected	ViRobot	Undetected
Zillya	Undetected	Zoner	Undetected
Avast-Mobile	Unable to process file type	Symantec Mobile Insight	Unable to process file type
Trapmine	Unable to process file type	Trustlook	Unable to process file type

VirusTotal Results

Ended with 30/71
Our NGAV$_{\#1}$ is no longer in the detection list

30
/ 71

Community Score

30 engines detected this file

8b458704a1468ab76b305f59d611a051438c815ba59051139400a8a88d29cfe6
test2_2.bin
direct-cpu-clock-access overlay peexe runtime-modules

1.41 MB
Size

2020-10-18 08:29:23 UTC
7 days ago

EXE

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY

Ad-Aware	Gen:Variant.Barys.54394	ALYac	Gen:Variant.Barys.54394
SecureAge APEX	Malicious	Arcabit	Trojan.Barys.DD47A
Baidu	Win32.Trojan.Kryptik.aep	BitDefender	Gen:Variant.Barys.54394
BitDefenderTheta	Gen:NN.ZexaF.34570.AD3@a8lho0ai	Comodo	TrojWare.Win32.Regsup.DQ@6dd0q3
CrowdStrike Falcon	Win/malicious_confidence_80% (D)	Cybereason	Malicious.128e1a
Cynet	Malicious (score: 100)	DrWeb	Trojan.Inject2.31002
Elastic	Malicious (high Confidence)	Emsisoft	Gen:Variant.Barys.54394 (B)
eScan	Gen:Variant.Barys.54394	FireEye	Generic.mg.2300fc5128e1a264
GData	Gen:Variant.Barys.54394	Ikarus	Trojan.Crypt
K7GW	Trojan (700001211)	Kaspersky	HEUR:Trojan.Win32.Generic
MAX	Malware (ai Score=84)	McAfee	Trojan-Goznym!917118DFD37C
McAfee-GW-Edition	BehavesLike.Win32.Ramnit.th	Qihoo-360	HEUR/QVM20.1.A9F7.Malware.Gen
Rising	Malware.Undefined!8.C (TFE:5:lbJPwS8D...	SentinelOne (Static ML)	DFI - Suspicious PE
Symantec	SMG.Heur!gen	Tencent	Malware.Win32.Gencirc.10b3a1f9
VBA32	BScope.Trojan.Inject	ZoneAlarm by Check Point	HEUR:Trojan.Win32.Generic
Acronis	Undetected	AegisLab	Undetected
AhnLab-V3	Undetected	Alibaba	Undetected

Acronis	Undetected	AegisLab	Undetected
AhnLab-V3	Undetected	Alibaba	Undetected
Antiy-AVL	Undetected	Avast	Undetected
AVG	Undetected	Avira (no cloud)	Undetected
Bkav	Undetected	CAT-QuickHeal	Undetected
ClamAV	Undetected	CMC	Undetected
Cylance	Undetected	Cyren	Undetected
eGambit	Undetected	ESET-NOD32	Undetected
F-Secure	Undetected	Fortinet	Undetected
Jiangmin	Undetected	K7AntiVirus	Undetected
Kingsoft	Undetected	Malwarebytes	Undetected
MaxSecure	Undetected	Microsoft	Undetected
NANO-Antivirus	Undetected	Palo Alto Networks	Undetected
Panda	Undetected	Sangfor Engine Zero	Undetected
Sophos AV	Undetected	Sophos ML	Undetected
SUPERAntiSpyware	Undetected	TACHYON	Undetected
TotalDefense	Undetected	TrendMicro	Undetected
TrendMicro-HouseCall	Undetected	VIPRE	Undetected
ViRobot	Undetected	Webroot	Undetected
Yandex	Undetected	Zillya	Undetected
Zoner	Undetected	Avast-Mobile	Unable to process file type

Summary

NGAV is not a silver bullet

No matter how much effort is put into an NG classifier it still may not be enough

Explainability is a dual edged sword

- Explainable high level features are easier to understand by humans and are more susceptible to modification
- Where the attacker lacks knowledge about the attacked model, he/she can use a surrogate model

Practical insights

- We found that the order of operation mattered at times and resulted in widely different scores
- Inserting enough small perturbations (modifications) can drastically change the score, even though their individual contribution is relatively small

Source: https://www.instructedtech.com/2017/12/24/5-ways-to-break-out-of-the-matrix/