Machine Learning and Malware Classification

By:
Amit Raut, raut.am@husky.neu.edu
Pranav Sharma, sharma.pran@husky.neu.edu

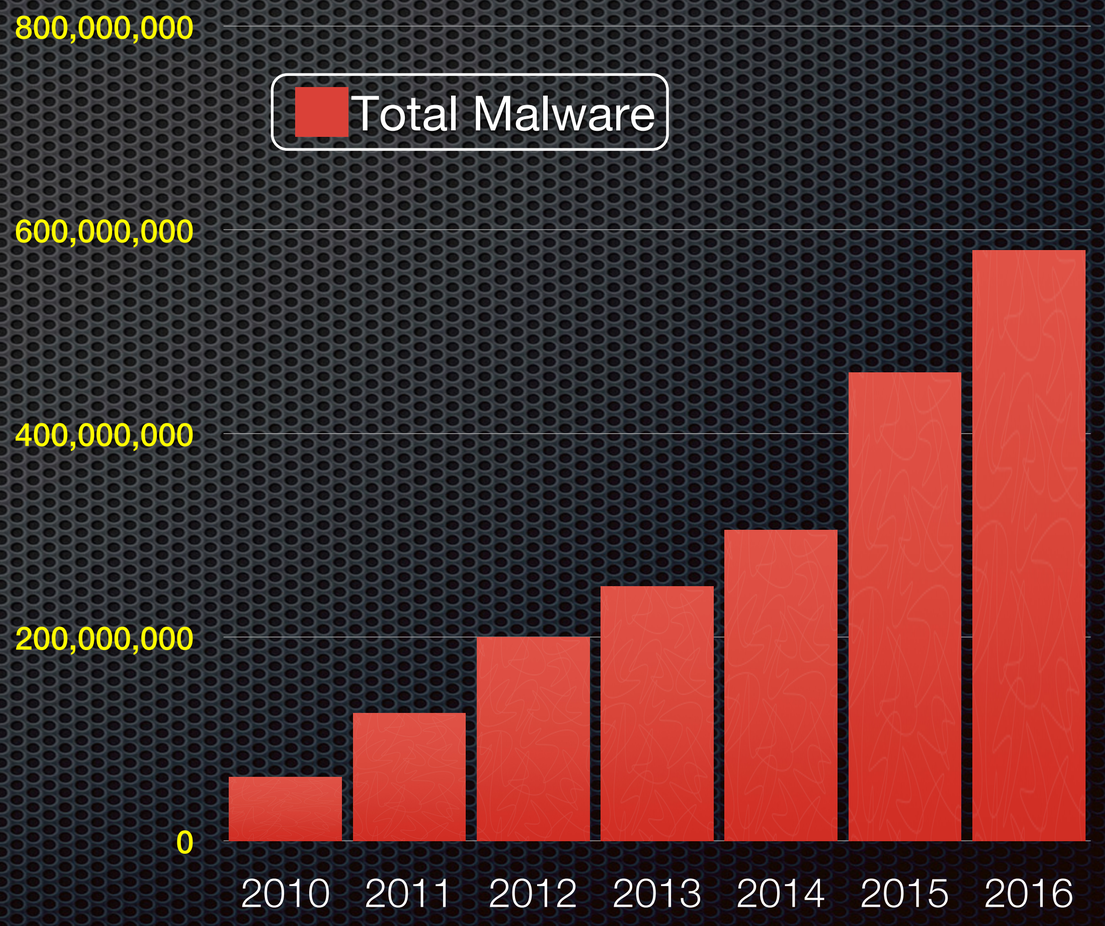
Northeastern University
College of Computer and Information Science

Technical Directors:
Aaron Ferber, ORNL
Jason Carter, ORNL

Project Guide:
Prof. Agnes Chan

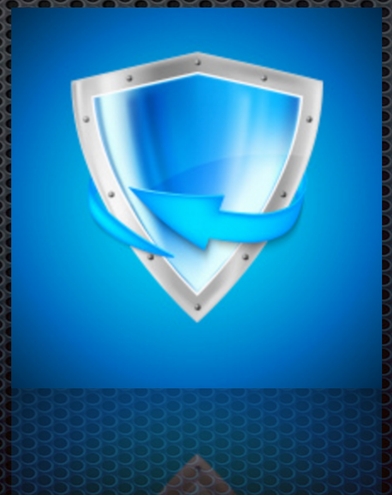
Background

- What is malware?
- Generally, any code that ‘performs evil’
- AV-TEST Institute registers over 390,000 new malicious programs every day
- What is anti-malware?
- Scanner detects and removes malware



Background

- How does an anti-malware function?
- Scanning for files with bad signature



Introduction to Problem

- Cost/benefit analysis of using benign code to train a classifier?
- What useful features of benign executable should be considered when doing this type of work?
- Feasibility to construct a repository of larger number of benign samples?

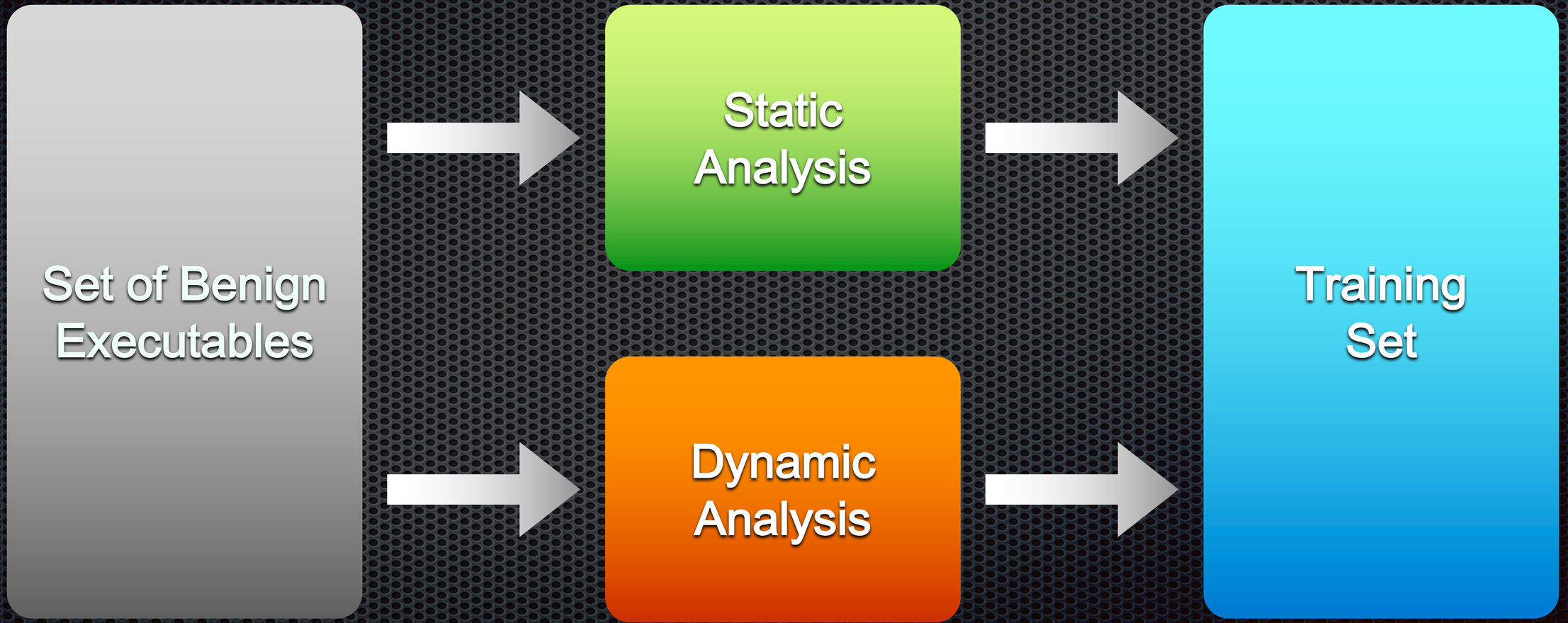
Approach Taken

- *Scope:*
  - *We worked on Windows executables*
  - *Samples include benign executables from a fresh Windows 7 installation and executables from ORNL*
- *Technology Used:*
  - *Windows 7, Python 2.7, Google Drive, and VMWare Fusion, Deep Freeze*

Why Are We Different?

- Previous researches on malware classification are based upon blacklists or bad signatures
- In our research we are using whitelisting or good signatures for malware classification

How we built the training set?



Static Analysis

Benign Executable

exiftool

```
File Permissions: rwxr-xr-x
File Type              : Win32 EXE
File Extension         : exe
MIME Type              : application/octet-stream
Machine Type           : Intel 386 or later,
and compatibles
PE Type                : PE32
Linker Version         : 9.0
OS Version             : 5.0
Subsystem              : Windows GUI
File OS                : Windows NT 32-bit
Object File Type       : Executable application
Language Code          : English (U.S.)
Character Set          : Unicode
```

SHA256

f233735b0268...
43ac3fa08907...
9095bdb85930...
1e86d1bc52f2...
a384f26ac006...
1c98f4f26b67...
b7edbb1d306a...
a19a1584344c...
405dbe690629...
c21d631f9b74...
2b33588fb038...
01508f973c2d...
f922e0d1e144...

Training Set

Static Analysis (Cont.)

Benign Executable

readpe

```
DOS Header
Magic number                    :  0x5a4d (MZ)
Bytes in last page              :  144
Pages in file                   :  3
Size of header in paragraphs    :  4
Maximum extra paragraphs        : 65535
Initial SP value                : 0xb8
Address of relocation table     : 0x40
PE header offset                : 0xd8
COFF/File header
Machine                         : 0x14c
IMAGE_FILE_MACHINE_I386
Number of sections              : 4
Date/time stamp:                1290246045 (Sat,
20 Nov 2010 09:40:45 UTC)
*~*~*~*~*~Output Truncated*~*~*~*~*~
```

SHA256

f233735b0268...
43ac3fa08907...
9095bdb85930...
1e86d1bc52f2...
a384f26ac006...
1c98f4f26b67...
b7edbb1d306a...
a19a1584344c...
405dbe690629...
c21d631f9b74...
2b33588fb038...
01508f973c2d...
f922e0d1e144...

Training Set

Static Analysis (Cont.)

Benign Executable

PEV

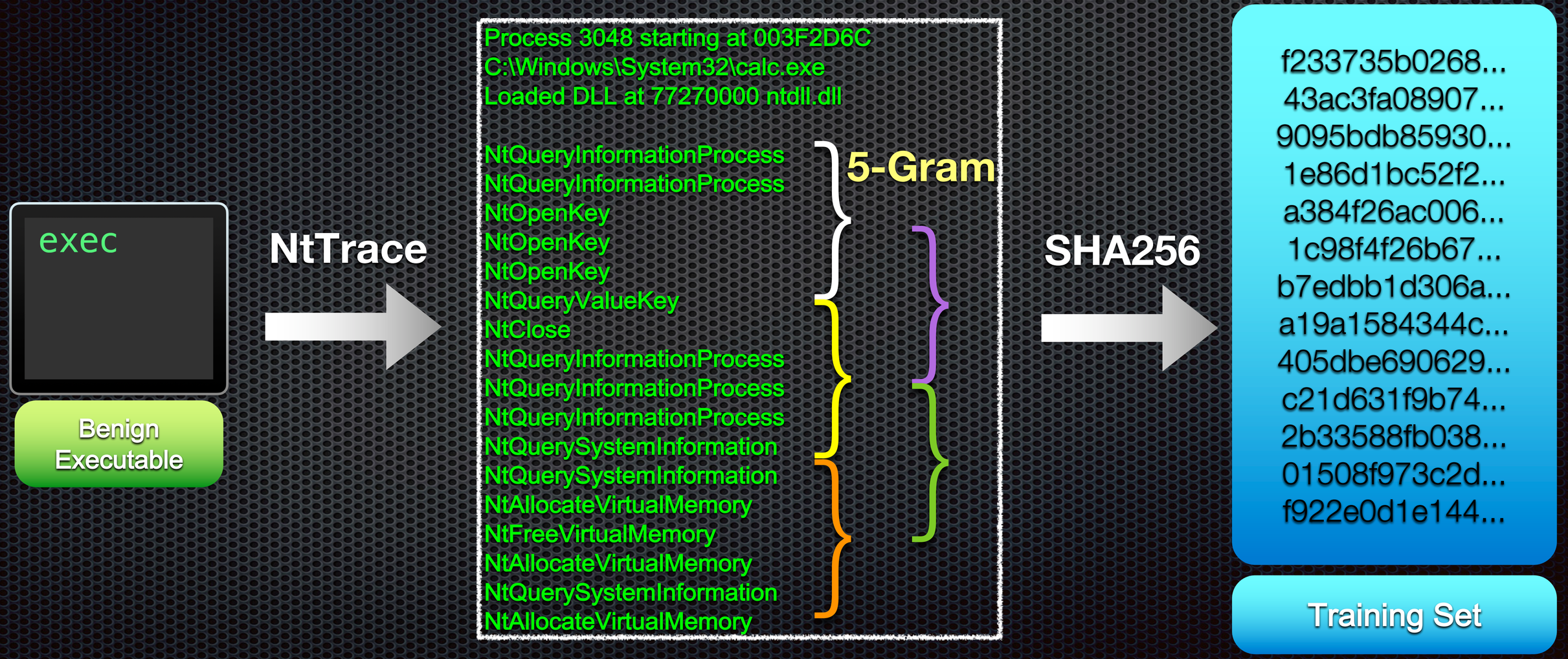
```
> pesec.exe c:\Windows\System32\calc.exe
ASLR                              : yes
DEP/NX                            : yes
SEH                               : yes
Stack cookies (EXPERIMENTAL)      : yes

>pescan.exe c:\Window\System32\calc.exe
file entropy      : 7.156275 (probably packed)
fpu anti-disassembly              : no
imagebase                         : normal
entry point                       : normal
DOS stub                          : normal
TLS directory                     : not found
section count                     : 4
.text                             : normal
.data                             : normal
.rsrc                             : normal
.reloc                            : normal
timestamp                         : normal
```

SHA256

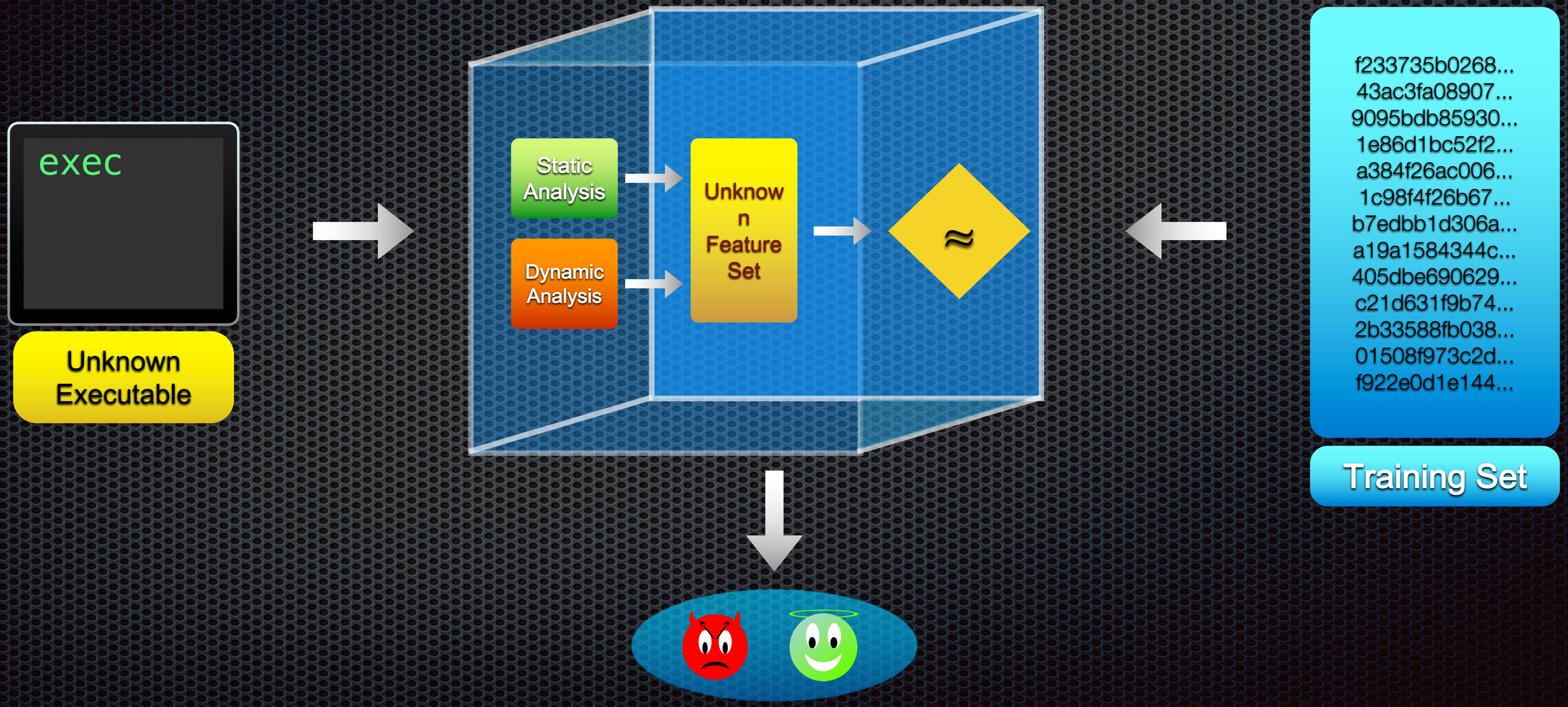
f233735b0268...
43ac3fa08907...
9095bdb85930...
1e86d1bc52f2...
a384f26ac006...
1c98f4f26b67...
b7edbb1d306a...
a19a1584344c...
405dbe690629...
c21d631f9b74...
2b33588fb038...
01508f973c2d...
f922e0d1e144...

Training Set

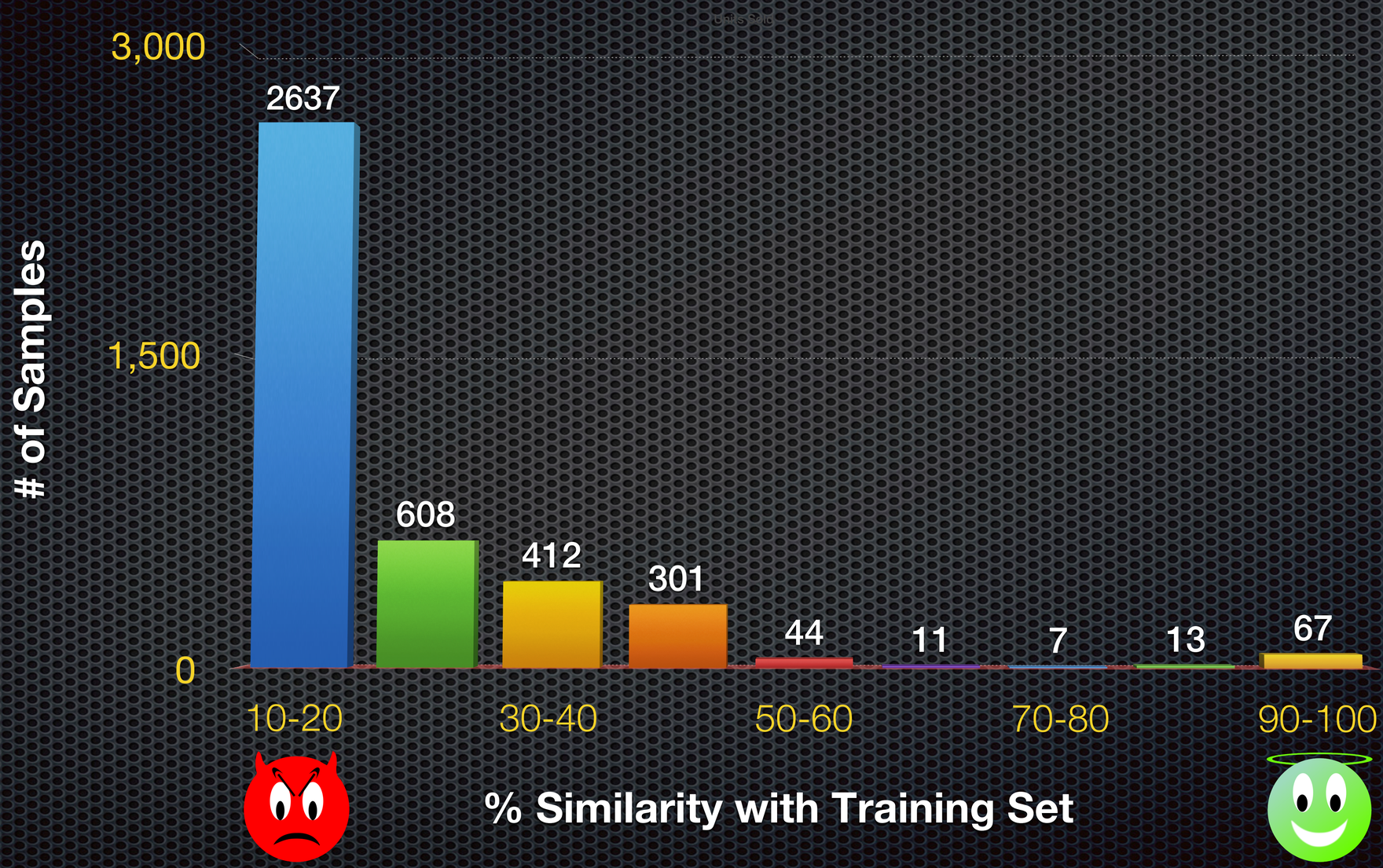
Dynamic Analysis



Classifier


exec
Unknown Executable
Static Analysis
Dynamic Analysis
Unknown Feature Set
≈
f233735b0268...
43ac3fa08907...
9095bdb85930...
1e86d1bc52f2...
a384f26ac006...
1c98f4f26b67...
b7edbb1d306a...
a19a1584344c...
405dbe690629...
c21d631f9b74...
2b33588fb038...
01508f973c2d...
f922e0d1e144...
Training Set

Results



Note: % similarity range 1-10 excluded because of NtTrace failure for some samples

Observations so far...

- Static analysis should be preferred over dynamic analysis when dealing with large volume of malware but could give false sense of accuracy
- Creating malware clusters could help to analyze malware family better
- Combinational approaches (blacklisting and whitelisting) should be preferred for effective malware classification
- Sophisticated malware that use encryption are very difficult to classify
- No perfect approach so far

Advantages and Disadvantages of whitelisting approach

- **Advantages**
  - Highly Secure
  - Can prevent zero day attack
- **Disadvantages**
  - Need to frequently update the training set

References

[1] Kyoung Soo Han , Boojoong Kang , Eul Gyu Im, Malware classification using instruction frequencies, Proceedings of the 2011 ACM Symposium on Research in Applied Computation, November 02-05, 2011, Miami, Florida

[2] Jiyong Jang , David Brumley , Shobha Venkataraman, BitShred: feature hashing malware for scalable triage and semantic analysis, Proceedings of the 18th ACM conference on Computer and communications security, October 17-21, 2011, Chicago, Illinois, USA

[3] Zhao H., Xu M., Zheng N., Yao J., Hou Q., Malicious executables classification based on behavioral factor analysis. e-Education, e-Business, e-Management and e-Learning, International Conference on,0:502-506, 2010.

[4] Andreas Moser, Christopher Kruegel, and Engin Kirda. Limits of static analysis for malware detection. In Proceedings of the 23rd Annual Computer Security Applications Conference, ACSAC'07, pages 421--430, Los Alamitos, CA, USA, December 2007. IEEE Computer Society.

Questions?

Thank You!

Special Thanks to Prof. Agnes Chan, Aaron Ferber and Jason Carter for giving us this opportunity…