Machine Learning for Malware Analysis

Machine Learning
A.Y. 2017-2018

Dr. Luca Massarelli
massarelli@dis.uniroma1.it

CIS SAPIENZA
RESEARCH CENTER FOR CYBER INTELLIGENCE AND INFORMATION SECURITY

Outline

- Introduction
- Background
- Analysis objectives
- Applying machine learning to malware analysis
- Sample features for Android Malware
- The Drebin Dataset
- Homework

What is a malware?

A malware is a malicious software that fulfills the deliberately harmful intent of an attacker

Nikola Milosevic. “History of malware”. In: CoRR abs/1302.5392 (2013).
URL: http://arxiv.org/abs/1302.5392.

Malware typical characteristics

Often a malware:

- Is designed to damage users or systems;
- Exploits Software and Hardware Vulnerabilities;
- Uses Social Engineering to trick users;
- Can install other malware;
- Is controlled by a command and control server;

CIS Sapienza
Research Center for Cyber Intelligence and Information Security

Beware of Malware!!!

19% of all cyber attacks are malware driven!

*(*SERT Quarterly Threat Report Q2 2016*)*

Globally, malware-based cyber attacks grew of 85%

during the 1° semester 2017 with respect to the 2° semester 2016

(CLUSIT Report 2017)

Malware analysis and the role of Machine Learning

"Malware analysis concerns the study of malicious samples with the aim of developing a deeper understanding about several aspects of malware"

- Malware behavior
- How they evolve in time
- How they intrude target systems
- …

Malware analysis and the role of Machine Learning

- Security defences are improving and evolving
- Nevertheless, malware are still succeeding

"Within the unceasing arm race
between malware developers and analysts,
each progress of security mechanisms
is likely to be promptly followed
by the realization of some evasion trick"

Malware analysis and the role of Machine Learning

"The easiness of
overcoming novel defensive measures
also depends on how well these measures
capture malicious traits of samples"

Detection rules based on MD5	vs	Detection rules to capture malicious semantics

CIS Sapienza
Research Center for Cyber Intelligence and Information Security

Malware analysis and the role of Machine Learning

- Defense-side goal:
  produce defensive technologies
  as challenging as possible to overcome

➢ Need to capture malicious aspects and traits having the broadest scope

➢ **Machine Learning** is a natural choice to support such a process of knowledge extraction

Malware analysis and the role of Machine Learning

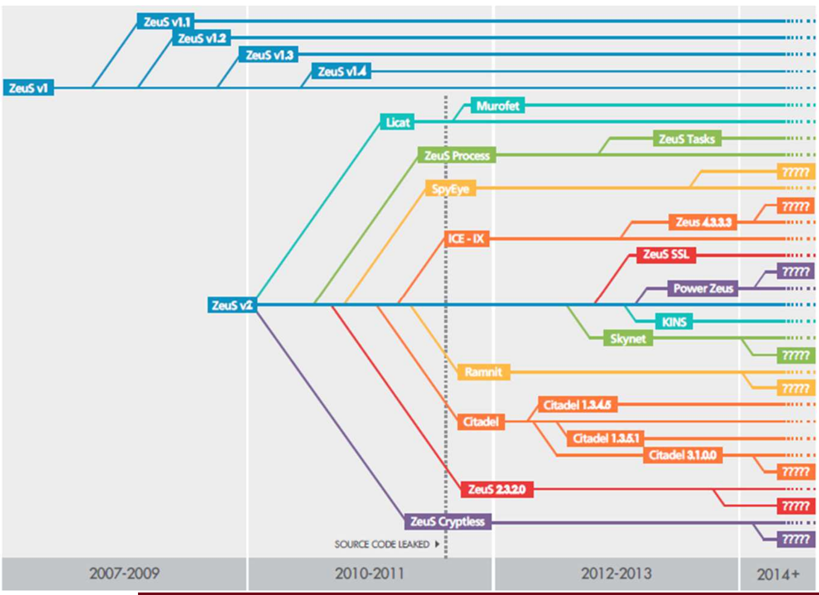
- Plentiful availability of labelled samples
  - ➢ Very large training set
  - ➢ Key element to foster the <u>adoption of machine learning for malware analysis</u>
- Many works in literature have taken this direction, with a variety of approaches, objectives and obtained results...

Outline

- Introduction
- Background
- Analysis objectives
- Applying machine learning to malware analysis
- Sample features for Android Malware
- The Drebin Dataset
- Homework

Malware variants

*Malware developers produce **variants** to minimize the effort required to evade updated security defences*



An "original" malware evolves in time through the development of variants (es: Zeus)

Malware family

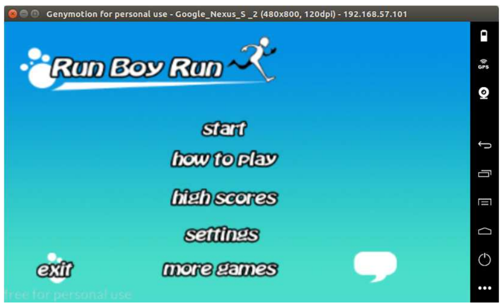
*The set of variants deriving from the same malware strain (i.e., “original” sample) is a malware **family***

- Similar behavior
- Exploits same vulnerabilities
- Same malicious objectives

CIS Sapienza
Research Center for Cyber Intelligence and Information Security

Malware Family (Android example)

Package Name: com.requiem.slingshakLite
Activities: com.requiem...



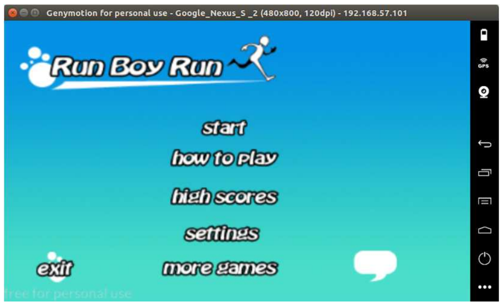
Package Name: ca.rivalstudios.runboyrun
Activities: ca.rivalstudios.runboyrun...

However….

CIS Sapienza
Research Center for Cyber Intelligence and Information Security

Malware Family (Android example)

Package Name: com.requiem.slingshakLite
Activities: com.requiem...
Services: com.GoldDream.zj.zjService
Receivers: com.GoldDream.zj.zjReceiver
Certificate:
61ed377e85d386a8dfee6b864bd85b0bfaa5
af81
Relevant Strings:
http: // lebar. gicp. net/ more. aspx? pid= 9944& amp; cid= 1000



Package Name: ca.rivalstudios.runboyrun
Activities: ca.rivalstudios.runboyrun...
Services: com.GoldDream.zj.zjService
Receivers: com.GoldDream.zj.zjReceiver
Certificate:
61ed377e85d386a8dfee6b864bd85b0bfaa5
af81
Relevant Strings:
http: // lebar. gicp. net/ more. aspx? pid= 9944& amp; cid= 1000

Obfuscation Techniques (Android example)

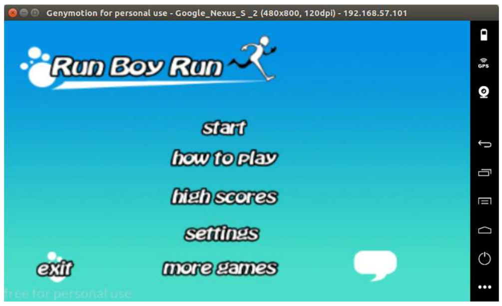
Obfuscation Techniques:

- Activity, Service and Receiver names can be changed and randomized;
- Applications can be signed with a different certificate;
- Binary code and application resources can be encrypted;

CIS SAPIENZA
Research Center for Cyber Intelligence and Information Security

Not so simple…

Package Name: com.requiem.slingshakLite
Activities: com.requiem...
Services: com.requiem.se.1
Receivers: com.requiem.se.1
Certificate:
94fg474u34d296n8pjle9n060bi89n0brad5cf
41
Relevant Strings:
*EnCt2d5fcaad2bd889cb92be48ba0d67cc1e
886= cf70fbd5fcaad2bd889cb92be48ba0X=
GXQtvQ2gL*



Package Name: ca.rivalstudios.runboyrun
Activities: ca.rivalstudios.runboyrun...
Services: com.rivalstudios.a.1
Receivers: com.rivalstudio.b.2
Certificate:
61ed377e85d386a8dfee6b864bd85b0bfaa5
af81
Relevant Strings:
*http: // lebar. gicp. net/ more. aspx? pid=
9944& amp; cid= 1000*

Signature-based analysis approaches

- Need to recognize already-known samples
  - If I know a sample is malicious, I want to detect its replicas
- Common techniques are signature-based
  - Hash of portions of code
  - Pattern matching on specific segments
  - Generally based on static characteristics
- Obviously malware can evade these techniques with obfuscation…

Machine learning analysis approaches

Machine learning permits to build malware analysis techniques that:

- Not need human support.
- Are resilient to obfuscation techniques.

In general machine learning permits to create malware analysis techniques based on the semantic of an application and not the code appearance.

Outline

- **Introduction**
- **Background**
- **Analysis objectives**
- **Applying machine learning to malware analysis**
- **Sample features for Android Malware**
- **The Drebin Dataset**
- **Homework**

Objectives of the analysis

- Malware detection is the most common objective
  - Respond to a real and urgent need: is this sample malicious? Should I worry?
  - Antiviruses are typical examples
- There can be situations where malware analysis is aimed at something different...

Malware variants detection

- A variant is *different enough* to evade detection, and *similar enough* to have the same behavior
- Detecting that a sample is a variant of a known malware is important
  - Removal and protection operations are likely the same
  - No need to investigate the sample further

CIS SAPIENZA
RESEARCH CENTER FOR CYBER INTELLIGENCE AND INFORMATION SECURITY

Malware variants detection

- Variants selection
  *Given a malicious sample **m**, select from the available knowledge base the samples that are variants of **m***

- Families selection
  *Given a malicious sample **m**, select from the available knowledge base the families which **m** belongs to*

Other objectives:

- Category Detection:
  - *Ransomware, trojan, …*
- Novelty and similarity detection:
  - *What parts of the malware have been already seen?*
- Malware development detection:
  - *Discover ongoing developments of new malware;*
- Malware attribution:
  - *Who commissioned the development of a malware?*
- Malware triage:
  - *Prioritization of the analysis;*

Outline

- Introduction
- Background
- Analysis objectives
- Applying machine learning to malware analysis
- Sample features for Android Malware
- The Drebin Dataset
- Homework

Applying Machine Learning to malware analysis

- Supervised learning
  - Need for labelled training set
  - Relevant example: classify unknown samples in known malware families
- Unsupervised learning
  - No need for labelled training set
  - Relevant example: cluster samples to identify families

An example: Malware Detection

- Example: malware detection
  - Given a file, establish whether it is a malware
  - Two main types of analysis (hybrids are possible)
    - Static analysis
    - Dynamic analysis
  - Can be seen as a binary classification

An example: Malware Detection

- The goal is finding a function MD having
    - The set F of all possible files as domain
    - The set {P,N} as codomain
      - Positive: the file is a malware
      - Negative: the file is not a malware

Given a specific file type (subset of F), how can we define MD?

An example: Malware Detection

Given a specific file type (subset of F), how can we define MD?

- Machine learning techniques provide means to find such a function
- Supervised learning allows to infer a function based on a labeled training dataset
- Given a function f to learn, with domain D and codomain C, the labelled training dataset (training set) is a set of pairs $\langle d, f(d) \rangle$, where $d \in D$ and $f(d) \in C$

An example: Malware Detection

In practice, supervised learning enables the learning of a function by providing a certain number of instances, each showing the expected output of the function given a specific input

CIS Sapienza
Research Center for Cyber Intelligence and Information Security

An example: Malware Detection

- Several algorithms/models for supervised learning
  - Artificial neural networks
  - Decision trees, random forest
  - Support vector machines
  - Nearest neighbor
  - …
- And several tools implementing them
  - Weka (www.cs.waikato.ac.nz/ml/weka)
  - Encog (www.heatonresearch.com/encog)
  - …

An example: Malware Detection

- Instances of a domain can be complex
  - Android Application package
  - Huge execution trace of an application
  - Network traffic log of an application
- What is the actual input of the function to learn?
  - Each element can be represented by a fixed set of **features** (attributes) $\{a_1, \ldots, a_n\}$ aimed at capturing all and only the characteristics that are relevant for the function to learn
  - Feature extraction is the process that, given an instance, returns its values for these features

An example: Malware Detection

- How to choose the set of features?
- What are the key characteristics for the function to learn?
- What are the specific cause-effect relationships that hold in that particular context?

This is where the intuition comes into play…

Evaluation Metrics

- Example: malware detection - accuracy metrics
  - Need to compare against some «ground truth»
  - Usually corresponds to the test set
  - For binary classification, there are four cases to be considered:

		Learned Function Output	
		Positive	Negative
Ground Truth	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Evaluation Metrics

- Example: malware detection - accuracy metrics
  - Meaning of true/false positive/negative for malware detection
  - True Positive
    It is a malware, and I correctly detected it
  - False Positive
    It is not a malware, but I thought it was
  - True Negative
    It is not a malware, and I thought so too
  - False Negative
    It is a malware, but I didn't detect it

Evaluation Metrics

- Example: malware detection - accuracy metrics
  - Precision: TP / (TP + FP)
    - How many files are real malware (TP) among those I considered as malware (TP + FP)?
    - «if I say it is a malware, then it really is a malware» (i.e., very few FP)
  - Recall: TP / (TP + FN)
    - How many malware did I spot (TP) among those in the test set (TP + FN)?
    - «if it is a malware, then I spot it» (i.e., very few FN)

Evaluation Metrics

- Example: malware detection - accuracy metrics
  - False Positive Rate: FP / (FP + TN)
    - How many files did I wrongly consider as malware (FP) among all the benign files (FP + TN)?
  - Accuracy: (TP + TN) / (TP + FN + TN + FP)
    - How many files did I classify correctly?
  - F-measure
    - $2 \cdot (precision \cdot recall)/(precision + recall)$
    - Can be interpreted as a weighted average of precision and recall
    - Best value: 1 worst value: 0

Outline

- **Introduction**
- **Background**
- **Analysis objectives**
- **Applying machine learning to malware analysis**
- **Sample features for Android Malware**
- **The Drebin Dataset**
- **Homework**

Sample features

The key element to apply successfully machine learning to malware analysis is a right choice of features.

- Features extraction depends on two factors:
  – Target operating system (Windows, Android, …)
  – Type of analysis (Static, Dynamic or Hybrid)

Sample features for Android Malware

The structure of Android Operating system permits to extract some types of features that cannot be used in other environments.

Mainly, the manifest.xml inside the package of an application (apk) contains very useful information…

Static features:

- Features that can be extracted only by looking at the apk:
  - Components (*Activities, Services, Content Providers and broadcast receivers*);
  - Permissions;
  - API calls;
  - Strings;
  - Flow graph;

Dynamic features

- Features that can be extracted, executing the sample and looking at the execution traces:
  - Resource Consumptions;
  - System calls;
  - Download patterns;

CIS SAPIENZA
RESEARCH CENTER FOR CYBER INTELLIGENCE AND INFORMATION SECURITY

Outline

- Introduction
- Background
- Analysis objectives
- Applying machine learning to malware analysis
- Sample features for Android Malware
- The Drebin Dataset
- Homework

Drebin

DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket

https://www.tu-braunschweig.de/Medien-DB/sec/pubs/2014-ndss.pdf

Addresses the problem of malware family classification and malware detection with a support vector machine

CIS Sapienza
Research Center for Cyber Intelligence and Information Security

Drebin

- Uses static analysis to extract features from samples;
- Uses a dataset with 123,453 bening applications and 5,560 malware;
- The features extracted from each sample are public available;

Drebin – Extracted features

Features are extracted from the *manifest.xml* file and from the disassembled code.

Features can be divided in 8 different sets:
$S1, S2, \dots, S8$

Drebin – Extracted features

- From *manifest.xml* file:
  - ***S1: Requested hardware components*** *(GPS, camera, …);*
  - ***S2: Requested Permissions*** *(Send sms, access to contacts,…);*
  - ***S3: App components*** *(Activities, Services, Content Providers, Broadcast receivers);*
  - ***S4 Filtered Intents*** *(Inter Process Communications handled by the sample e.g. BOOT_COMPLETED);*

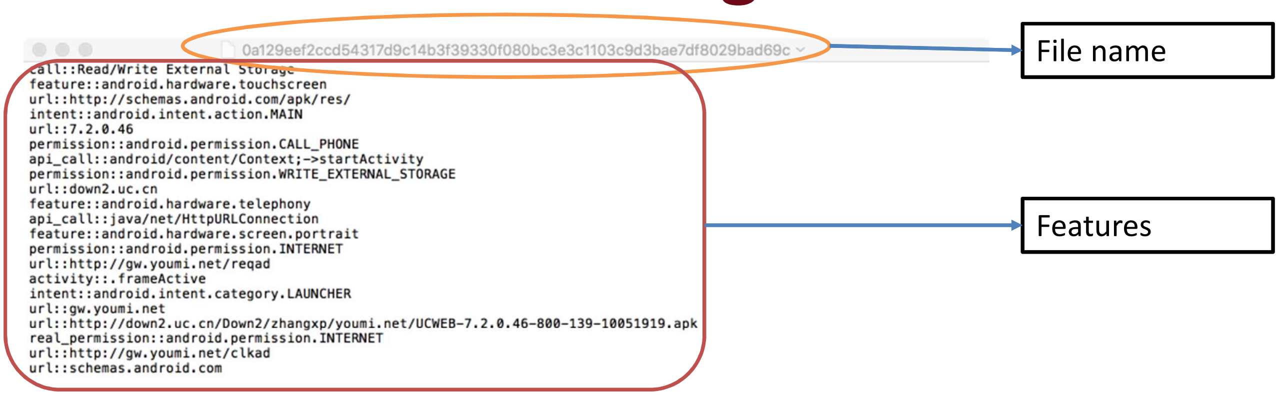
Drebin – Extracted features

- From the disassembled code:
  - ***S5: Restricted API calls*** (API calls whose access require a permission)
  - ***S6: Used permissions*** (Permissions effectively used by the application)
  - ***S7: Suspicious API calls*** (API calls who allow access to sensitive data e.g. *getDeviceId()*)
  - ***S8: Network addresses*** *(Urls embedded in the code)*

Drebin – Dataset organization

- The features extracted from each application of the dataset, can be downloaded as zip file from the course web-site.
- The zip file contains the features extracted from each application.
- Features extracted from an app are stored inside a text file whose name is the SHA1 Hash of the apk.

Drebin – Dataset organization



Each feature is composed by a prefix and a value, the prefix represent the set the feature belong to.

e.g.

Permission::android.permission.INTERNET

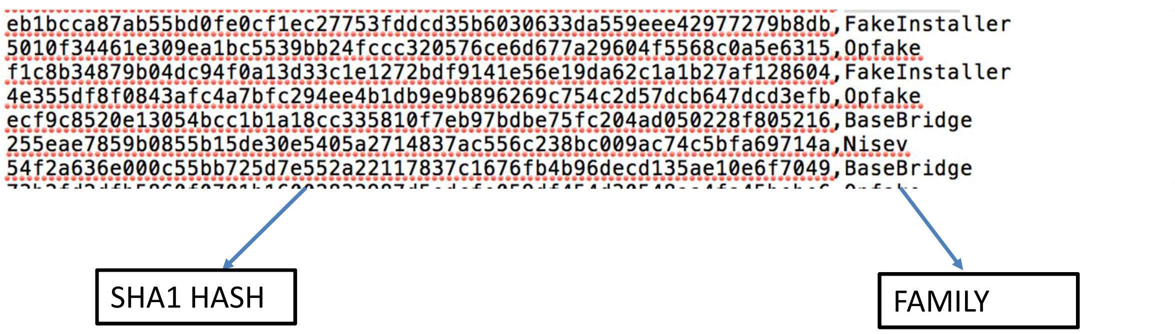
It belong to the set S2: required permission

Drebin – Dataset organization

Prefix	SET
feature	S1: Hardware components
permission	S2: Requested permission
activity service_receiver provider service	S3: App Components
intent	S4: Filtered Intents
api_call	S5: Restricted API calls
real_permission	S6: Used permission
call	S7: Suspicious API calls
url	S8: Network addresses

Drebin – Dataset organization

- The zip file contains also a dictionary file in csv format.
- In the dictionary there are the SHA1 Hash of the malware in the dataset and the family they belongs to.



CIS Sapienza
Research Center for Cyber Intelligence and Information Security

Outline

- Introduction
- Background
- Analysis objectives
- Applying machine learning to malware analysis
- Sample features for Android Malware
- The Drebin Dataset
- Homework

Homework

- Use the Drebin dataset to address the problem of malware detection and malware classification.

CIS SAPIENZA
RESEARCH CENTER FOR CYBER INTELLIGENCE AND INFORMATION SECURITY

Malware detection

- With all the applications in the dataset create a binary classifier whose output is
  - malware
  - non malware.
- Use the dictionary file as ground truth.
- Evaluate the performance of the classifier.

CIS Sapienza
Research Center for Cyber Intelligence and Information Security

Malware detection - Hints

- Bayesian approach (e.g. SPAM filter), no need to match features with numerical value
- Other classification algorithm (SVM, Random Forest…), in this case you need a match beetwen the string features and numerical value.
- Try to use not all set of features but only a few (permissions, api calls, urls).
- Use the paper as reference.

CIS Sapienza
Research Center for Cyber Intelligence and Information Security

Malware family classification

- Create a classifier that given in input the features of a malware output the family it belongs to.
- Use the dictionary as ground truth.
- Evaluate the performance of the classifier.

Malware family classification - Hints

- Select only malicious applications using the dictionary.
- Select only malware families that have more than 20 samples.
- Use a bayesian approach.
- Use different classification algorithm (SVM, Random Forest…)

CIS SAPIENZA
RESEARCH CENTER FOR CYBER INTELLIGENCE AND INFORMATION SECURITY