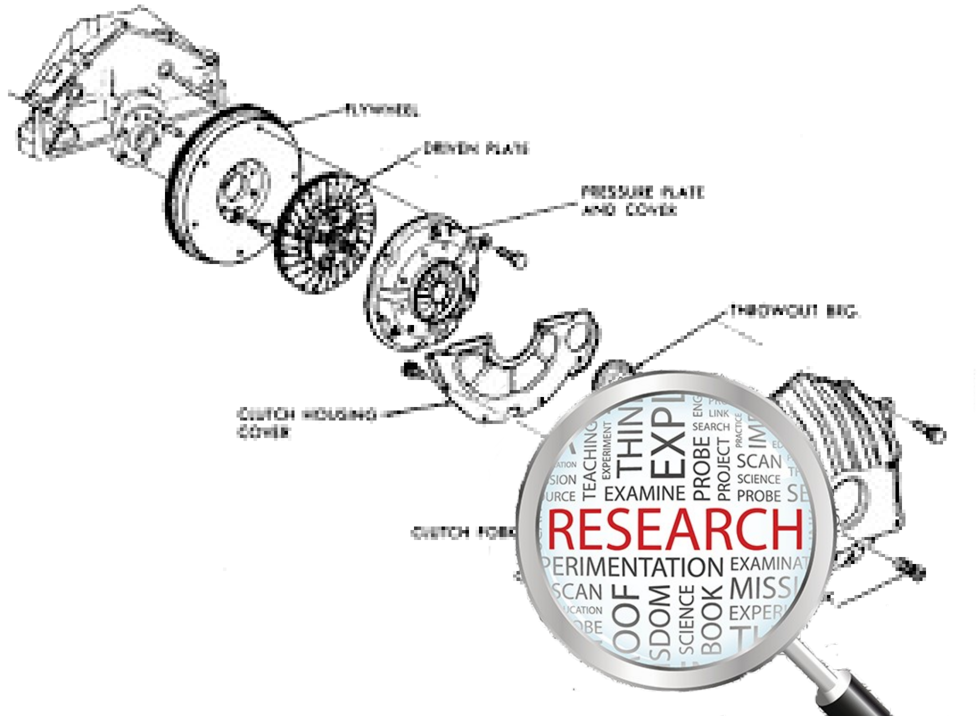
Malware Analysis

Old Problems & New Challenges

Davide Balzarotti

Evolution of Research in Malware Analysis and Classification

- Focus on problems, not solutions
- Academic point of view
- With *a bit of* exaggeration

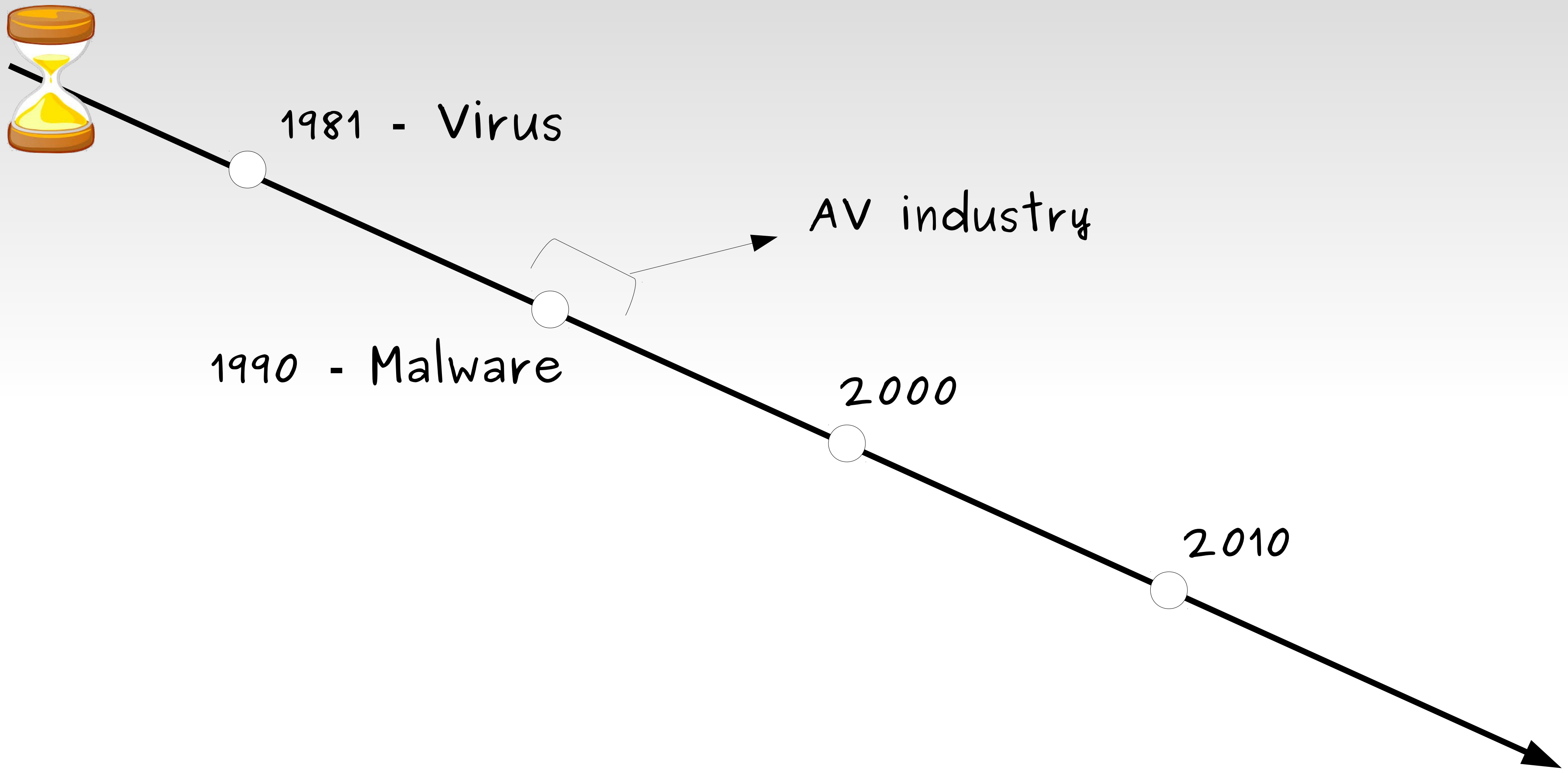


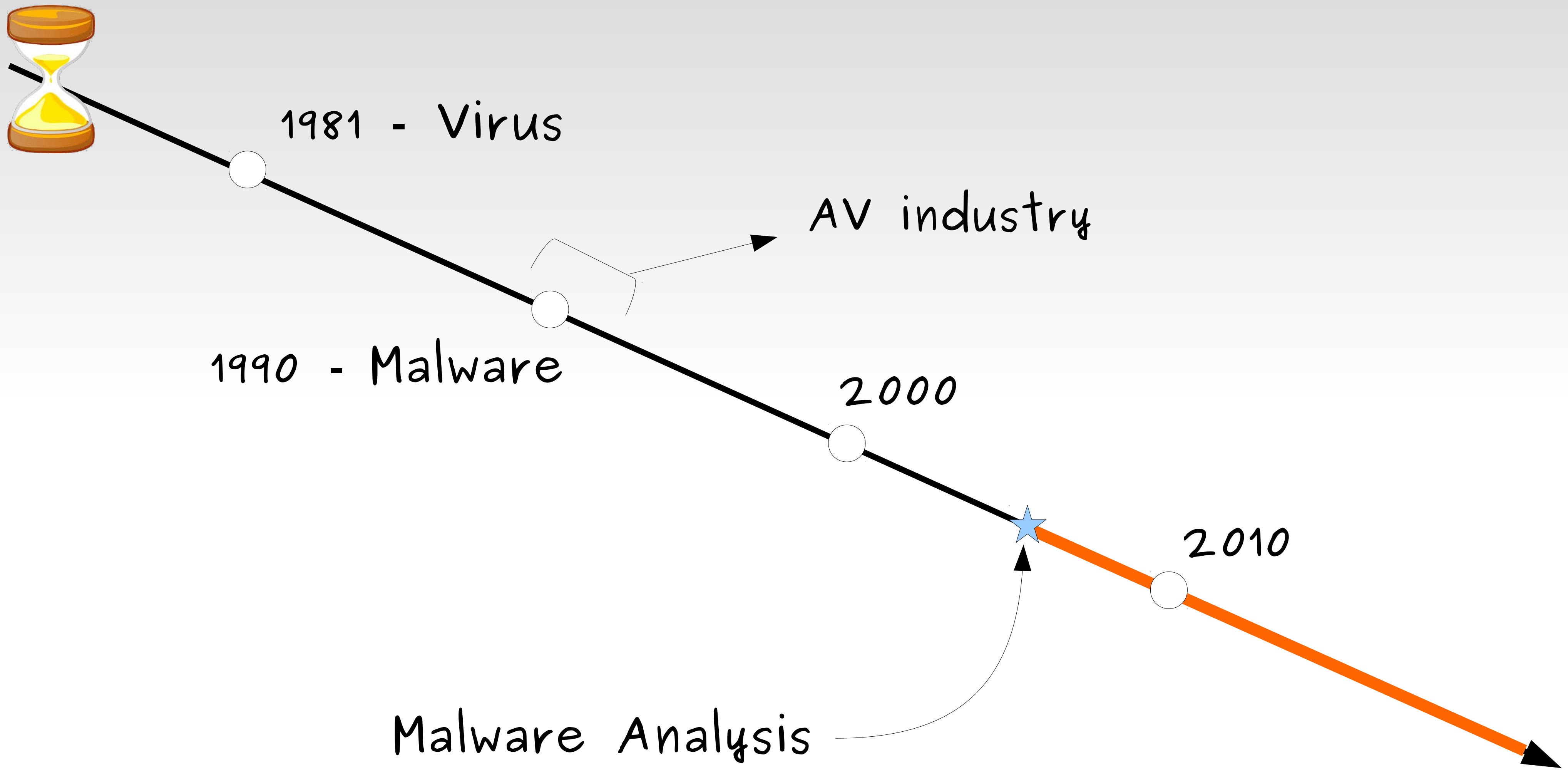
Malware Analysis

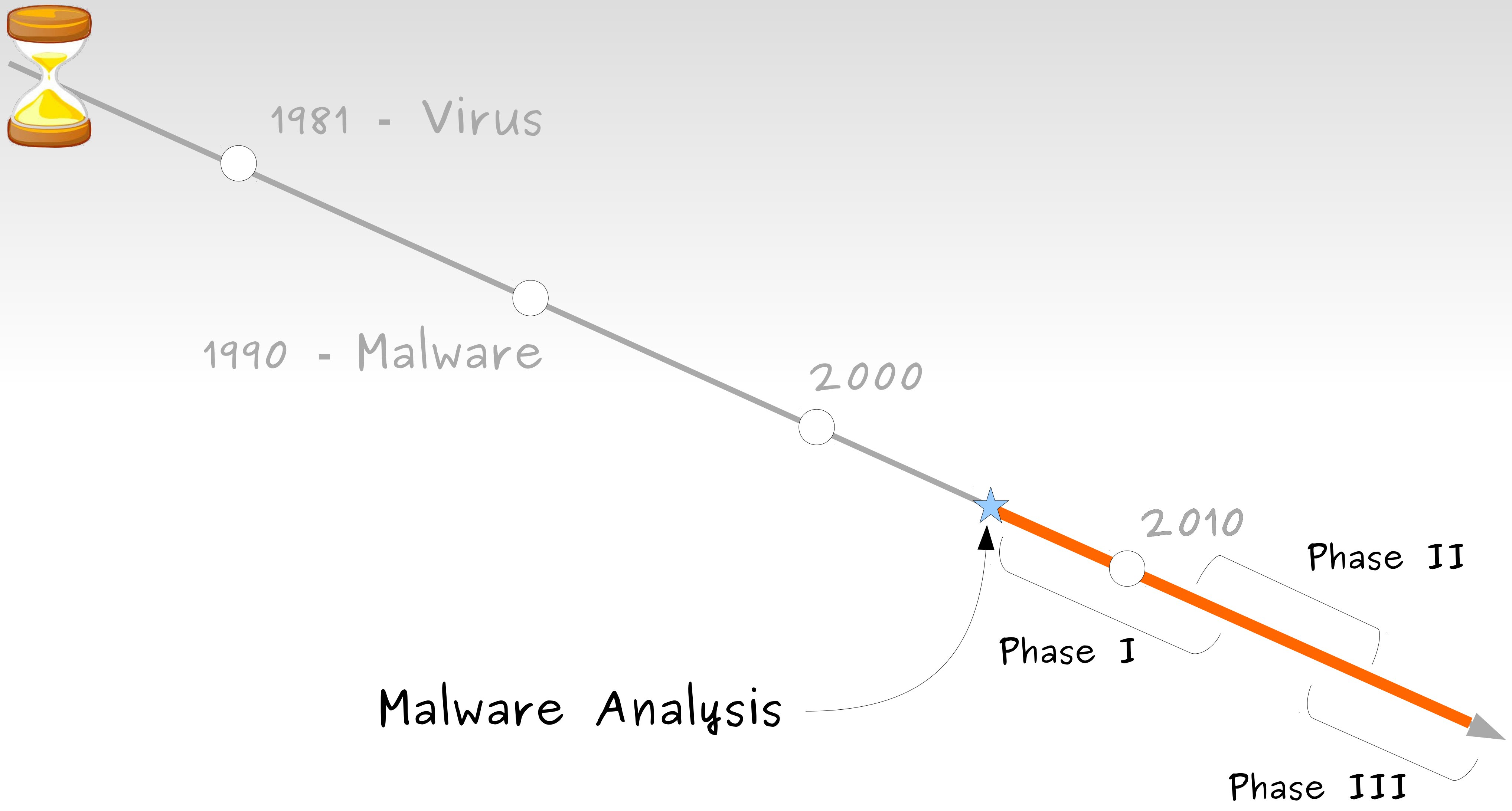
Given an unknown program in binary form

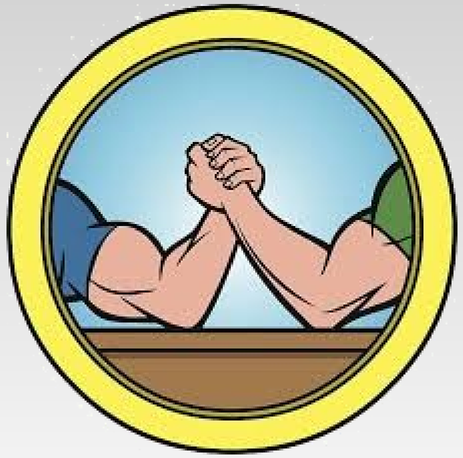
$\downarrow$

extract possible behavior,
classify as malicious or benign,
provide information about family


1981 - Virus
AV industry
1990 - Malware
2000
2010

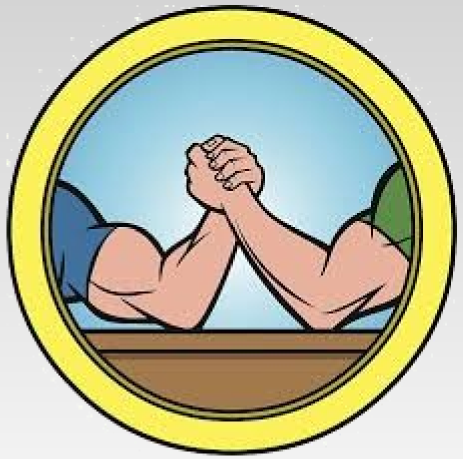

1981 - Virus
AV industry
1990 - Malware
2000
2010
Malware Analysis


1981 - Virus
1990 - Malware
2000
2010
Phase II
Phase I
Malware Analysis
Phase III



The Adversarial Phase

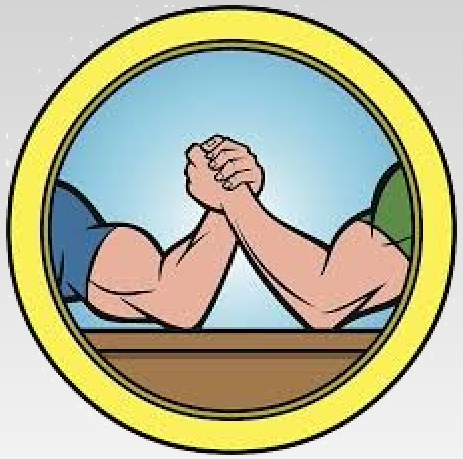
“ How do we analyze a program that does not want to be analyzed? ”



The Adversarial Phase

"How do we analyze a program that does not want to be analyzed"

- Limitations of static analysis
- De-obfuscation and Unpacking
- Dynamic analysis sandboxes

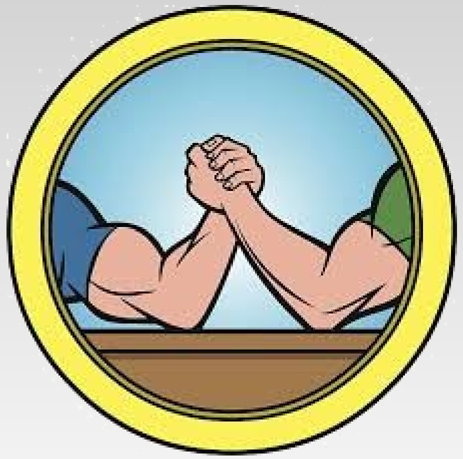


The Adversarial Phase

"How do we analyze a program that does not want to be analyzed"

- Limitations of static analysis
- De-obfuscation and Unpacking
- Dynamic analysis sandboxes

- Overcoming the limitations of dynamic analysis
  - Transparent instrumentation
  - Stalling code detection
  - Multi-path exploration
  - Program stimulation
  - ...



The Adversarial Phase

"How do we analyze a program that does not want to be analyzed"

- Limitations of static analysis
- De-obfuscation and Unpacking
- Dynamic analysis sandboxes
- Overcoming the limitations of dynamic analysis
  - Transparent instrumentation
  - Stalling code detection
  - Multi-path exploration
  - Program stimulation
  - …
- Network and host behavior analysis
- ...

The Data Analysis Phase

“ How One Billion Samples can change Malware Analysis? ”

The Data
Analysis Phase

“ How One Billion Samples can change Malware Analysis? ”

- New Constraints & new Challenges
- New Opportunities

The Data Analysis Phase

- New Constraints & new Challenges
  - Tradeoff between precision and scalability

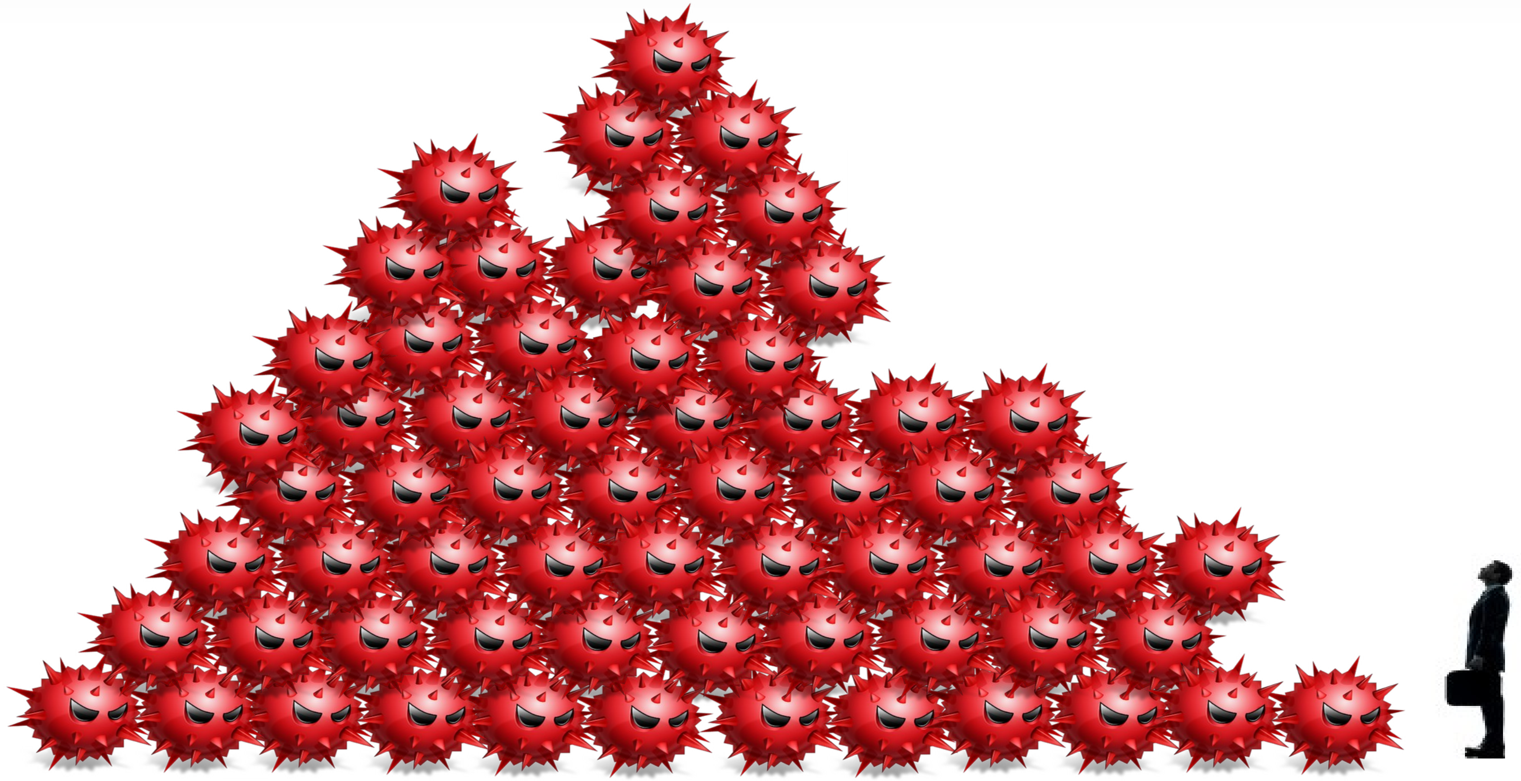
The Data Analysis Phase

- New Constraints & new Challenges
  - Tradeoff between precision and scalability
  - The needle in the haystack problem

The Data Analysis Phase

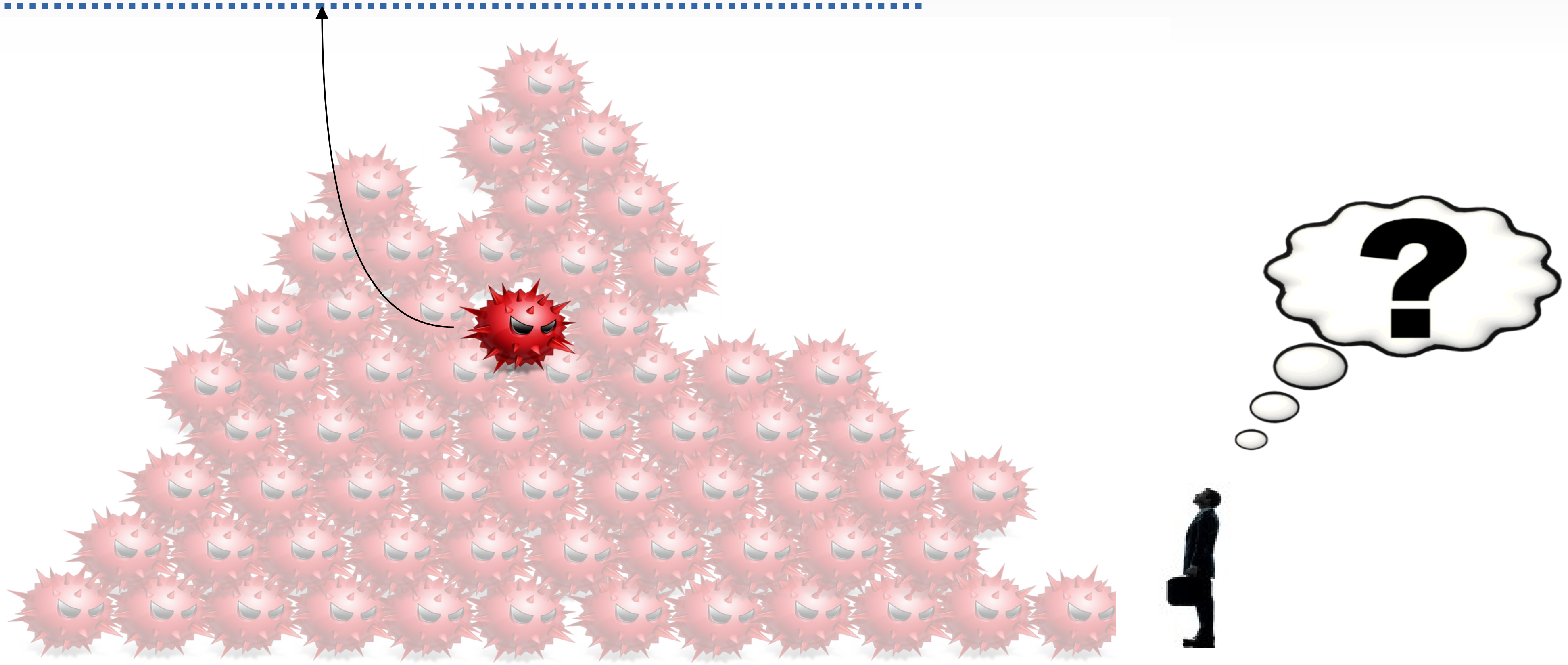
- New Constraints & new Challenges
  - Tradeoff between precision and scalability
  - The needle in the haystack problem
  - Data velocity

Even IF we had a perfect malware classifier, how do we distinguish interesting samples from the rest ?

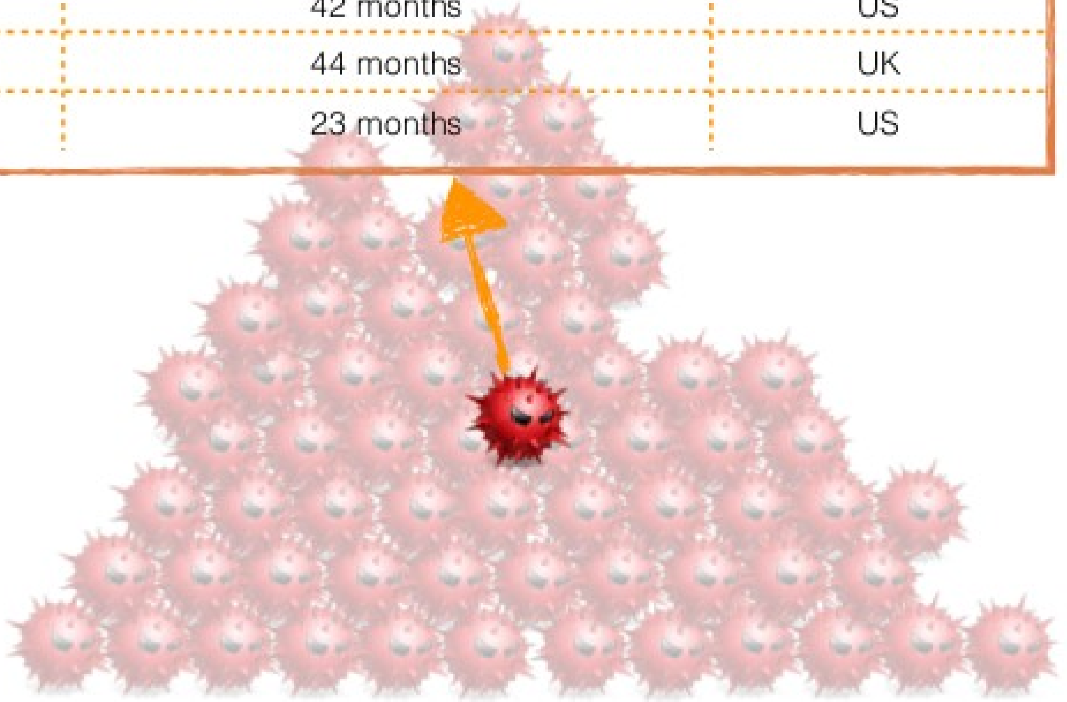


Equation Group Sample

(collected & analyzed 23 months before it was "discovered")

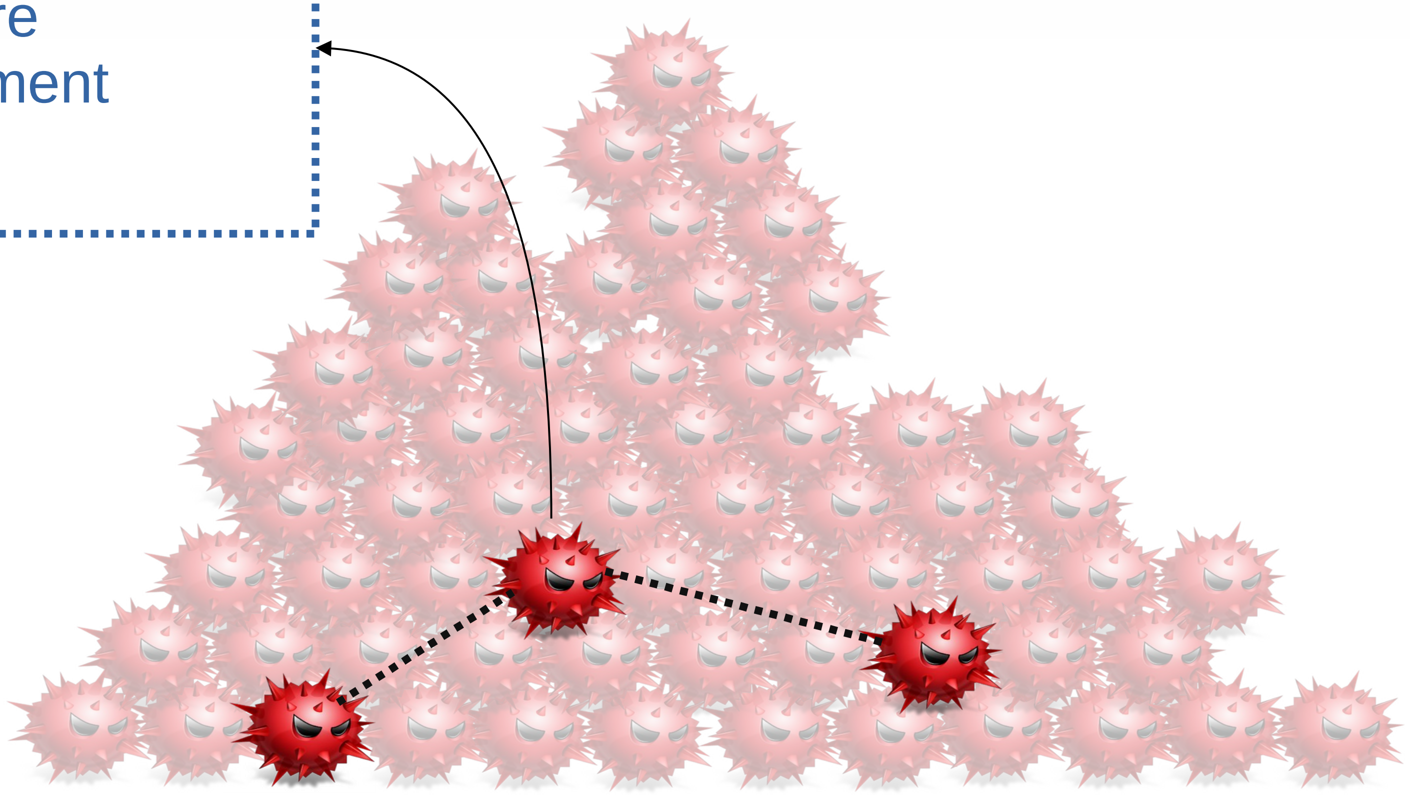


CAMPAIGN	TIME BEFORE PUBLIC DISCLOSURE	SUBMITTED BY
Operation Aurora	4 months	US
Red October	8 months	Romania
APT1	43 months	US
Stuxnet	1 month	US
Beebus	22 months	Germany
LuckyCat	3 months	US
BrutePOS	5 months	France
NetTraveller	14 months	US
Pacific PluX	12 months	US
Pitty Tiger	42 months	US
Regin	44 months	UK
Equation	23 months	US

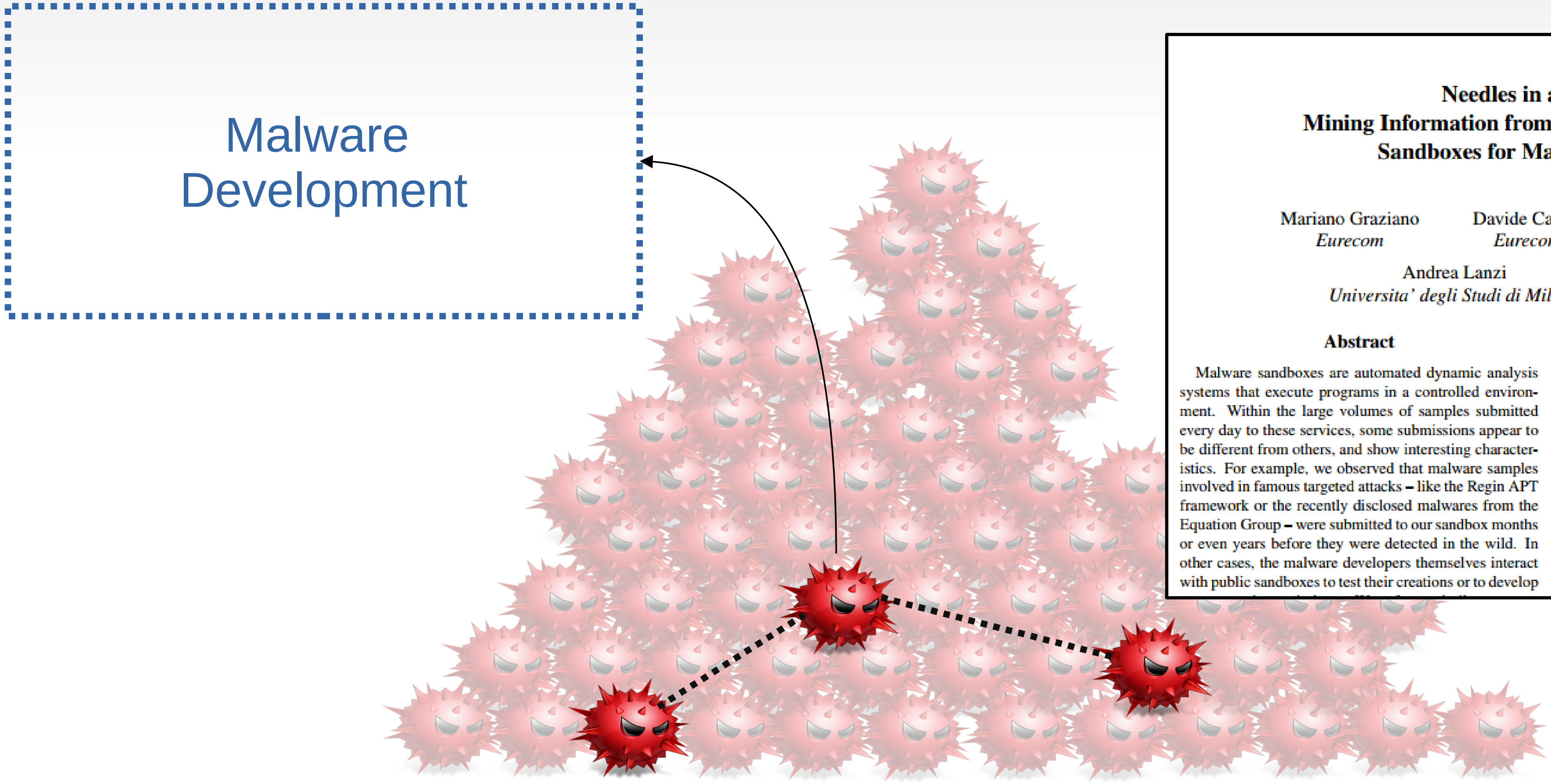


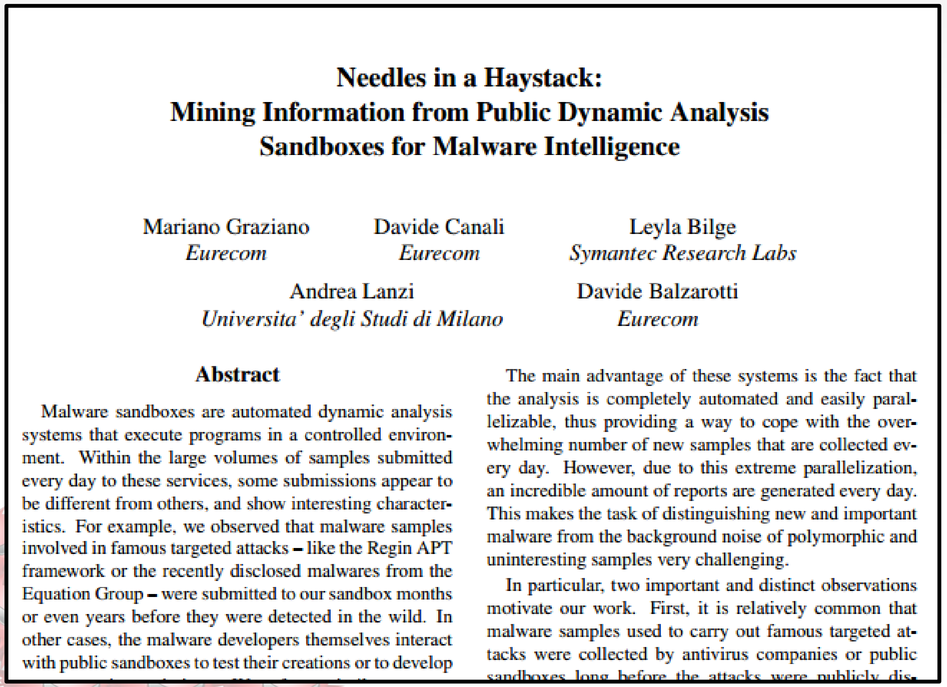
Malware Intelligence

Malware Development



Malware Intelligence





**Needles in a Haystack:
Mining Information from Public Dynamic Analysis
Sandboxes for Malware Intelligence**

Mariano Graziano *Eurecom* · Davide Canali *Eurecom* · Leyla Bilge *Symantec Research Labs*

Andrea Lanzi *Universita' degli Studi di Milano* · Davide Balzarotti *Eurecom*

Abstract

Malware sandboxes are automated dynamic analysis systems that execute programs in a controlled environment. Within the large volumes of samples submitted every day to these services, some submissions appear to be different from others, and show interesting characteristics. For example, we observed that malware samples involved in famous targeted attacks – like the Regin APT framework or the recently disclosed malwares from the Equation Group – were submitted to our sandbox months or even years before they were detected in the wild. In other cases, the malware developers themselves interact with public sandboxes to test their creations or to develop

The main advantage of these systems is the fact that the analysis is completely automated and easily parallelizable, thus providing a way to cope with the overwhelming number of new samples that are collected every day. However, due to this extreme parallelization, an incredible amount of reports are generated every day. This makes the task of distinguishing new and important malware from the background noise of polymorphic and uninteresting samples very challenging.

In particular, two important and distinct observations motivate our work. First, it is relatively common that malware samples used to carry out famous targeted attacks were collected by antivirus companies or public

Data Velocity

- At the end of 2007, Symantec reported a total of 1.1M malware samples (mostly due to trojan droppers)
  - Total malware samples went from 1M to 1B in ~8 years
  - Daily collected samples went from Hundreds to Millions in ~10 years

Data Velocity

- At the end of 2007, Symantec reported a total of 1.1M malware samples (mostly due to trojan droppers)
  - Total malware samples went from 1M to 1B in ~8 years
  - Daily collected samples went from Hundreds to Millions in ~10 years

Roughly one order of magnitude every 30 months

Data Velocity

1) The number of samples is increasing 4x faster than the Moore’s Law !!

Data Velocity

1) The number of samples is increasing 4x faster than the Moore’s Law !!

2) Are previous studies still relevant today?
   (sadly, academia never repeats previous studies)

Data Velocity

1) The number of samples is increasing 4x faster than the Moore's Law !!

2) Are previous studies still relevant today?
   (sadly, academia never repeats previous studies)

3) Lack of a *representative* dataset

Data Velocity

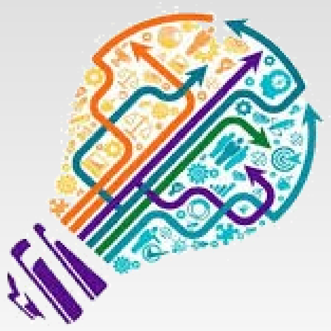
1) The number of samples is increasing 4x faster than the Moore’s Law !!

2) Are previous studies still relevant today?
 (sadly, academia never repeats previous studies)

3) Lack of a *representative* dataset

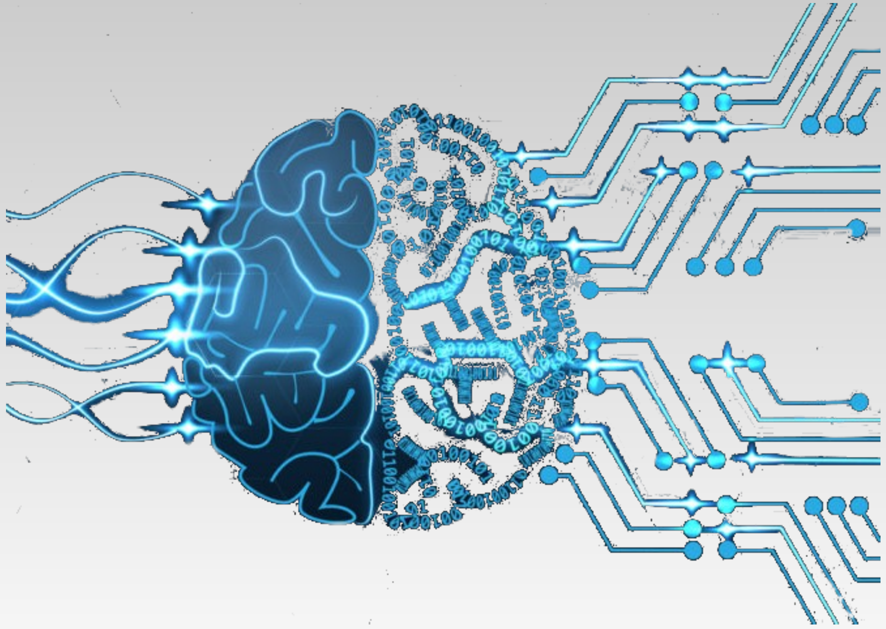
4) We are past the point-of-no-return for re-analyzing samples

The Data Analysis Phase



- (Missed?) Opportunities
  - Synergy of system security and machine learning
  - Analytics → Data-Driven Malware Analysis

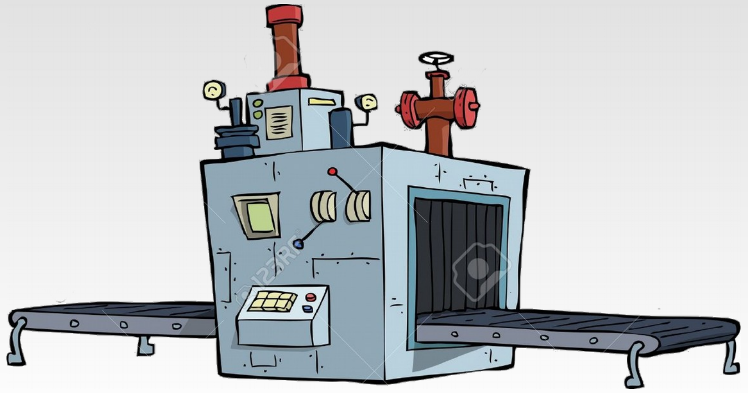
Machine Learning

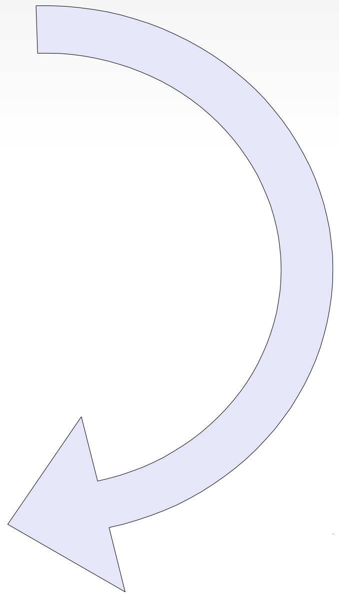


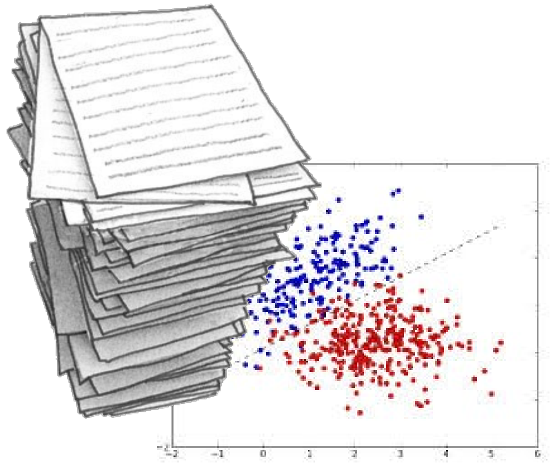
- The availability of a large amount of data made malware an attractive target for the machine learning community

- Difficult integration:
  - Security experts treat machine learning like black box Lego pieces
  - Data scientists lack domain and problem knowledge
  - It happened before... e.g., in the NIDS and anomaly detection field (see “*Outside the closed world: On using machine learning for network intrusion detection*”)

Malware Analytics







How Many Samples are Packed ?

How many have VM detection capabilities ?

How Many Samples are Packed ?

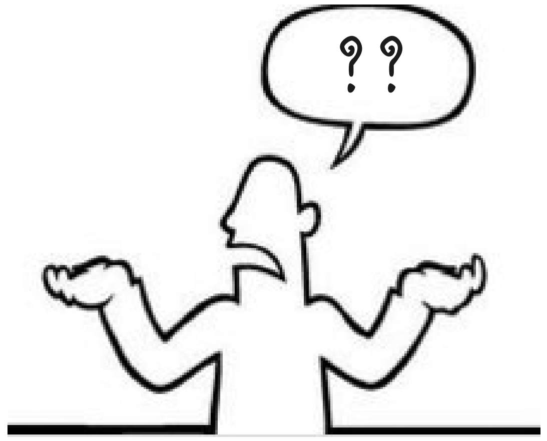
Panda [2007] → 79%

McAfee [2009] → 80%

Bayer [2011] → 40%

Intel [2014] → 37% (63% custom or unknown)

How many have VM detection capabilities ?



How Many Samples are Packed ?

Panda [2007] → 79%

McAfee [2009] → 80%

Bayer [2011] → 40%

Intel [2014] → 37% (63% custom or unknown)

How many have VM detection capabilities ?

Bayer [2011] → < 12%

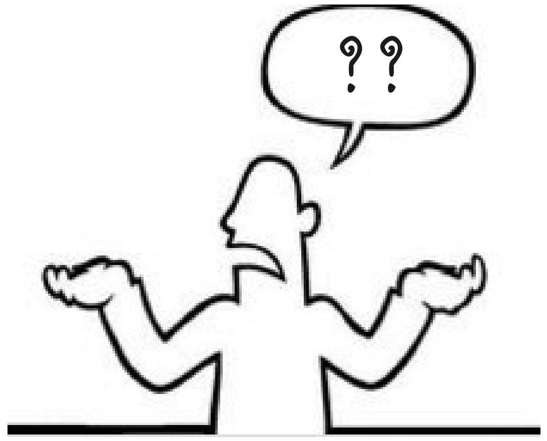
Lindorfer [2011] → 26%

Fireeye [2011] → It's a Myth!

Intel [2012] → 81%

Microsoft [2014] → 28%

Symantec [2014] → 18%–28%



How Long do we need to run each sample?

How many malicious samples also query popular domains?

What is the fraction of samples that do not belong to polymorphic families?

How prevalent is technique X?

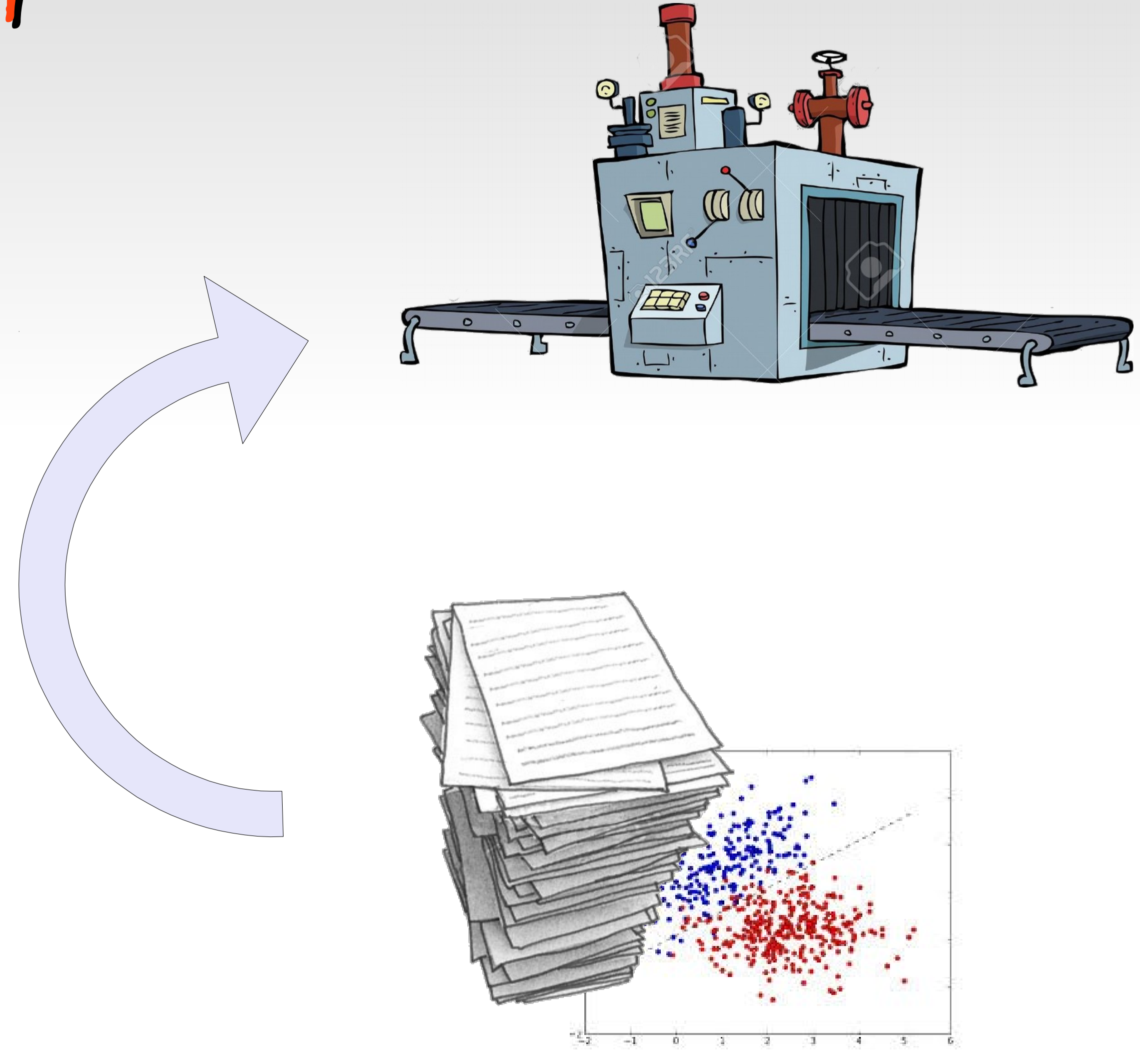
Hey look. A Squirrel !!

Big Data should allow us to extract Intelligence, Analytics, discover new Correlations, observe General Trends and the evolution of the Big Picture

Big Data should allow us to extract Intelligence, Analytics, discover new Correlations, observe General Trends and the evolution of the Big Picture

... and use this information to improve our malware analysis pipeline

Data-Driven Malware Analysis



The Data Analysis Phase

- We analyze and store huge amount of information.. but sadly we only use it as a *giant cache*

The Data Analysis Phase

- We analyze and store huge amount of information.. but sadly we only use it as a giant cache
- Plenty of confusing marketing statistics published by companies. With no information about the methodology or even the meaning of the terms

The Data Analysis Phase

- We analyze and store huge amount of information.. but sadly we only use it as a giant cache
- Plenty of confusing marketing statistics published by companies. With no information about the methodology or even the meaning of the terms
- Automation is key, but sometimes (?) humans need to be in the loop

The Data Analysis Phase

- We analyze and store huge amount of information.. but sadly we only use it as a giant cache
- Plenty of confusing marketing statistics published by companies. With no information about the methodology or even the meaning of the terms
- Automation is key, but sometimes (?) humans need to be in the loop
- We need help from data scientists... but they won’t solve the problem alone

Adversarial phase

Data Analysis phase

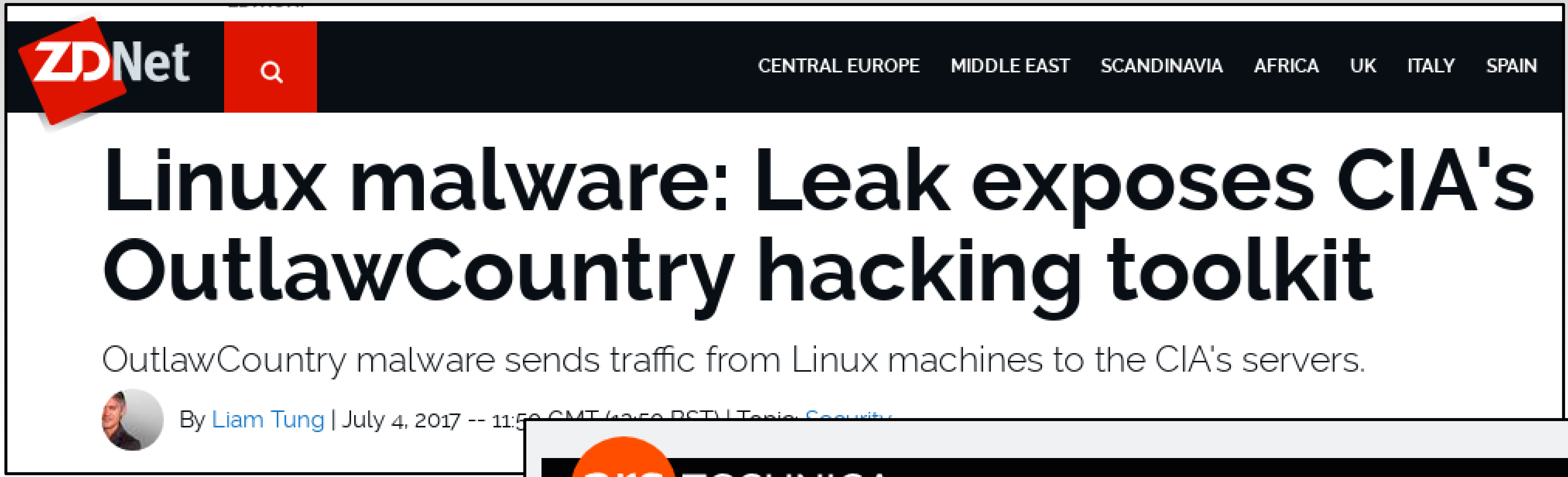
Adversarial phase

Data Analysis phase

Diversification phase

Diversity

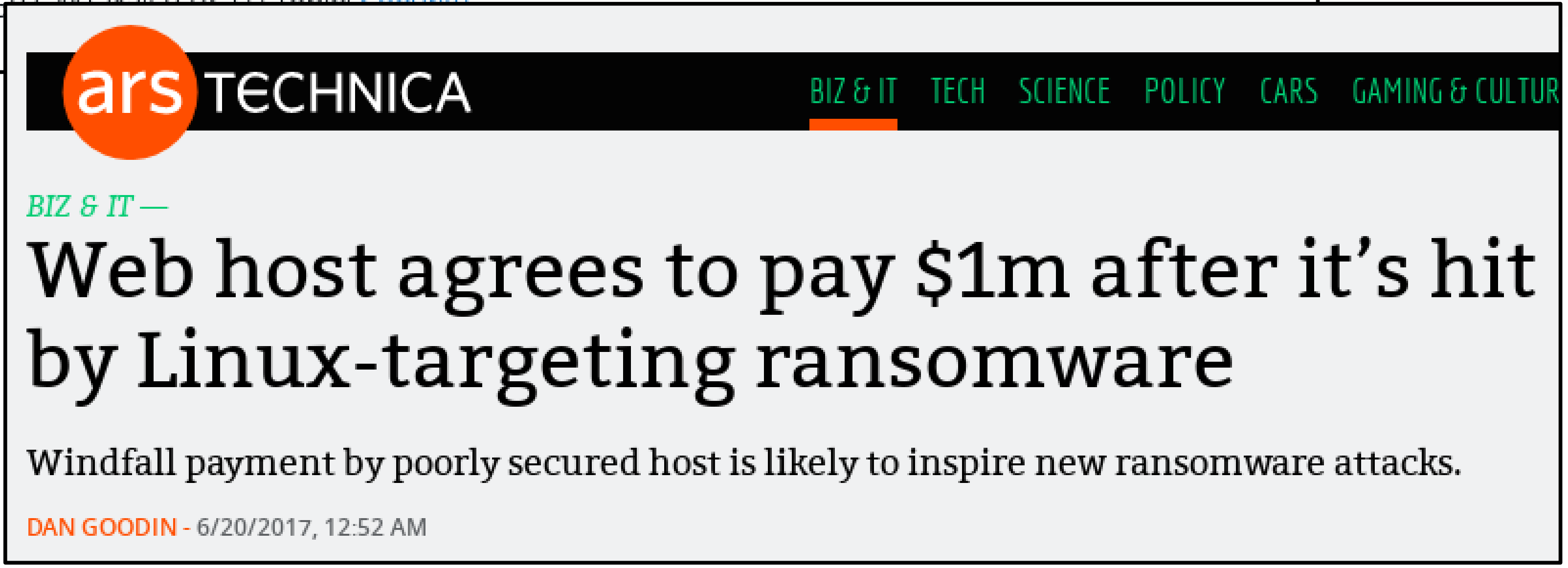
Not just Mirai



Linux malware: Leak exposes CIA's OutlawCountry hacking toolkit

OutlawCountry malware sends traffic from Linux machines to the CIA's servers.

By Liam Tung | July 4, 2017 -- 11:5

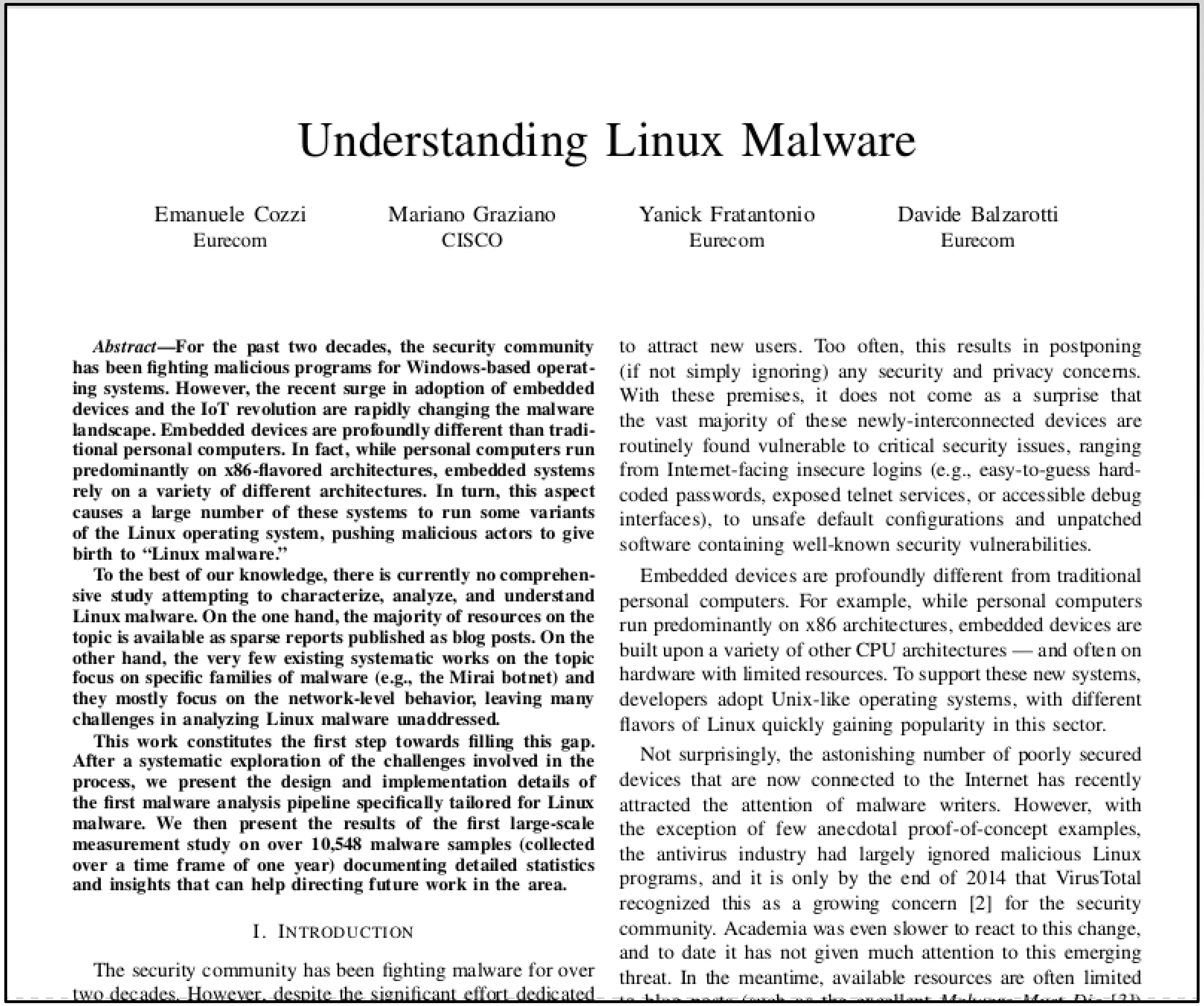


BIZ & IT —

Web host agrees to pay $1m after it’s hit by Linux-targeting ransomware

Windfall payment by poorly secured host is likely to inspire new ransomware attacks.

DAN GOODIN - 6/20/2017, 12:52 AM



Understanding Linux Malware

Emanuele Cozzi
Eurecom

Mariano Graziano
CISCO

Yanick Fratantonio
Eurecom

Davide Balzarotti
Eurecom

***Abstract*—For the past two decades, the security community has been fighting malicious programs for Windows-based operating systems. However, the recent surge in adoption of embedded devices and the IoT revolution are rapidly changing the malware landscape. Embedded devices are profoundly different than traditional personal computers. In fact, while personal computers run predominantly on x86-flavored architectures, embedded systems rely on a variety of different architectures. In turn, this aspect causes a large number of these systems to run some variants of the Linux operating system, pushing malicious actors to give birth to "Linux malware."**

To the best of our knowledge, there is currently no comprehensive study attempting to characterize, analyze, and understand Linux malware. On the one hand, the majority of resources on the topic is available as sparse reports published as blog posts. On the other hand, the very few existing systematic works on the topic focus on specific families of malware (e.g., the Mirai botnet) and they mostly focus on the network-level behavior, leaving many challenges in analyzing Linux malware unaddressed.

This work constitutes the first step towards filling this gap. After a systematic exploration of the challenges involved in the process, we present the design and implementation details of the first malware analysis pipeline specifically tailored for Linux malware. We then present the results of the first large-scale measurement study on over 10,548 malware samples (collected over a time frame of one year) documenting detailed statistics and insights that can help directing future work in the area.

I. Introduction

The security community has been fighting malware for over two decades. However, despite the significant effort dedicated to attract new users. Too often, this results in postponing (if not simply ignoring) any security and privacy concerns. With these premises, it does not come as a surprise that the vast majority of these newly-interconnected devices are routinely found vulnerable to critical security issues, ranging from Internet-facing insecure logins (e.g., easy-to-guess hard-coded passwords, exposed telnet services, or accessible debug interfaces), to unsafe default configurations and unpatched software containing well-known security vulnerabilities.

Embedded devices are profoundly different from traditional personal computers. For example, while personal computers run predominantly on x86 architectures, embedded devices are built upon a variety of other CPU architectures — and often on hardware with limited resources. To support these new systems, developers adopt Unix-like operating systems, with different flavors of Linux quickly gaining popularity in this sector.

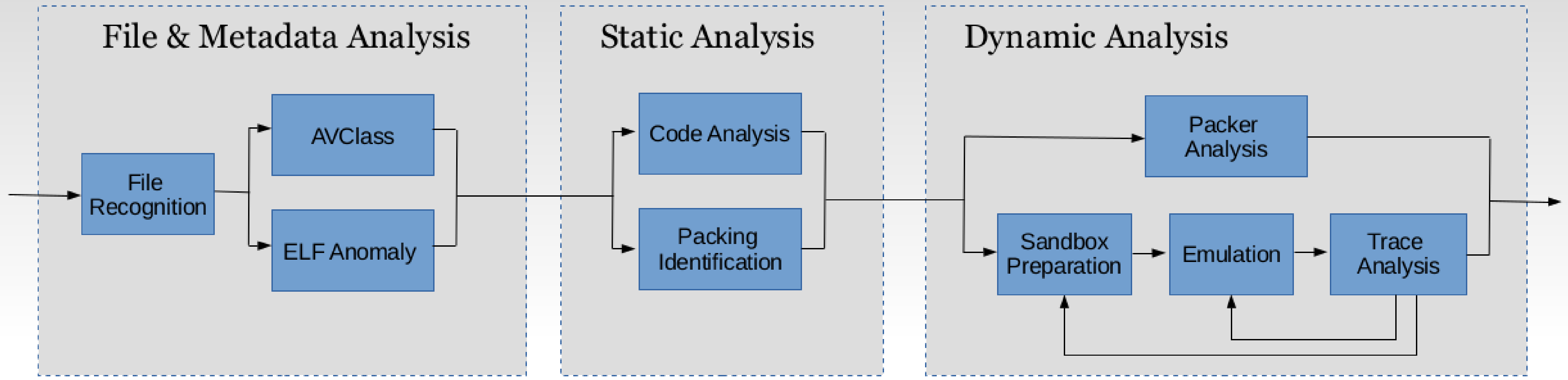
Not surprisingly, the astonishing number of poorly secured devices that are now connected to the Internet has recently attracted the attention of malware writers. However, with the exception of few anecdotal proof-of-concept examples, the antivirus industry had largely ignored malicious Linux programs, and it is only by the end of 2014 that VirusTotal recognized this as a growing concern [2] for the security community. Academia was even slower to react to this change, and to date it has not given much attention to this emerging threat. In the meantime, available resources are often limited

Goals

1) Develop a dynamic analysis sandbox for Linux binaries (and IoT devices)

2) Identify challenges and limitations of porting traditional techniques to the new environment

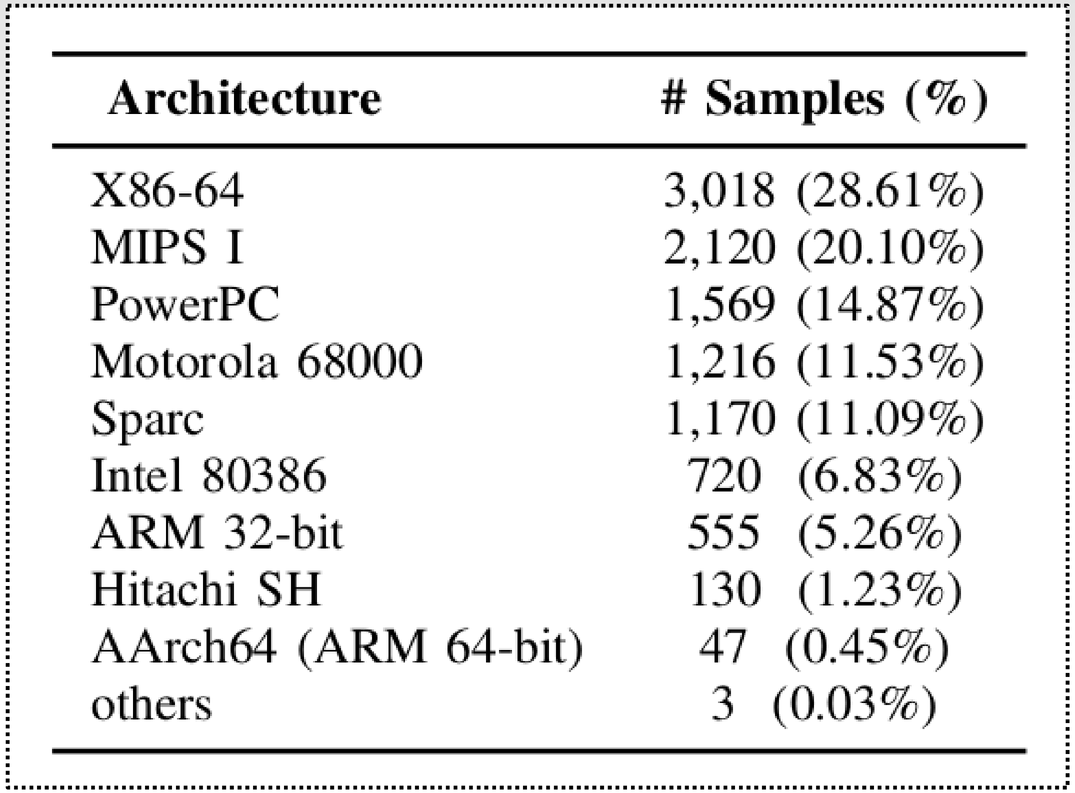
3) Understand differences in the malware characteristics (packing, obfuscantion, VM detection, privilege excalation, persistence...) wrt Windows malware



- Pipeline for static and dynamic analysis of Linux-based malware
- 10.5K samples (from ~110 families according to `avclass`) fully analyzed
- Free service at https://padawan.s3.eurecom.fr/

  (list of samples and all paper reports available to download)

- *Architecture Diversity*



Architecture	# Samples (%)
X86-64	3,018 (28.61%)
MIPS I	2,120 (20.10%)
PowerPC	1,569 (14.87%)
Motorola 68000	1,216 (11.53%)
Sparc	1,170 (11.09%)
Intel 80386	720 (6.83%)
ARM 32-bit	555 (5.26%)
Hitachi SH	130 (1.23%)
AArch64 (ARM 64-bit)	47 (0.45%)
others	3 (0.03%)

- *Architecture Diversity*
- *Environment Diversity*
  - *Loader*
  - *Libraries*
  - *Operating System / ABI*
  - *root/user privileges*
  - *….*

- *Architecture Diversity*
- *Environment Diversity*
  - *Loader*
  - *Libraries*
  - *Operating System / ABI*
  - *root/user privileges*
  - *….*

More than 80% of the samples we analyzed are statically linked, but only 24% of them have been stripped.

- *Architecture Diversity*
- *Environment Diversity*
  - *Loader*
  - *Libraries*
  - *Operating System / ABI*
  - *root/user privileges*
  - *....*

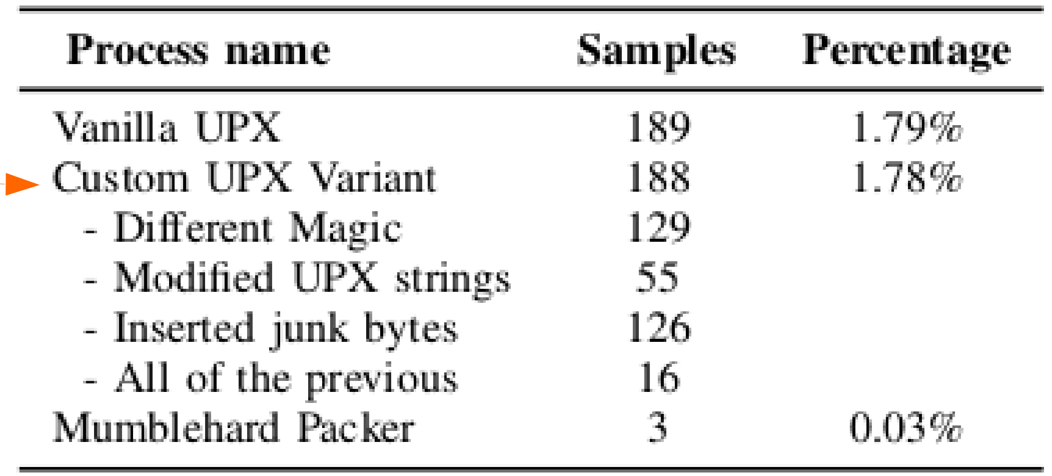
More than 80% of the samples we analyzed are statically linked, but only 24% of them have been stripped.

- *Architecture Diversity*
- *Environment Diversity*
  - *Loader*
  - *Libraries*
  - *Operating System / ABI*
  - *root/user privileges*
  - *....*

23% of the samples show a different behavior when executed with root privileges.

- *Architecture Diversity*
- *Environment Diversity*
- *Behavior Diversity*
    - *Persistence*
    - *Deception*
    - *Packing*
    - *ELF manipulation*
    - *Evasion*
    - *Hiding*
    - *….*

- *Architecture Diversity*
- *Environment Diversity*
- *Behavior Diversity*
  - *Persistence*
  - *Deception*
  - *Packing*
  - *ELF manipulation*
  - *Evasion*
  - *Hiding*
  - *….*



Process name	**Samples**	**Percentage**
Vanilla UPX	189	1.79%
Custom UPX Variant	188	1.78%
- Different Magic	129	
- Modified UPX strings	55	
- Inserted junk bytes	126	
- All of the previous	16	
Mumblehard Packer	3	0.03%

- *Architecture Diversity*
- *Environment Diversity*
- *Behavior Diversity*
- *Intra-Family Diversity*

  E.g.: Tsunami

  - *9 architectures*
  - *86% statically linked*
  - *13% stripped*
  - *different loaders*
  - *different persistence mechanisms*
  - *15% tested for privileged execution*
  - *2.3% did not work in a VM*
  - *...*

- *Process Injection*
- *Process Interaction*
- *Deception*
- *Anti-debugging*
- *Anti-Execution*
- *Persistence*
- *Privilege Escalations*
- *Sandbox Detection*
- *Shell Commands*
- *Process Enumeration*
- *Required Privileges*
- *Packing*
- *Information Gathering*

Diversity

- IoT Linux-based malware still in its infancy
- Already a broad range of behaviors and tricks
- ELF binaries could run anywhere from a thermostat to a large server
  - New research needed to overcome the lack of information about the execution environment

Conclusion

- Malware Analysis is a multi-faced problem that requires a broad set of techniques
  - Data Mining
  - Machine learning
  - Program analysis
  - Binary analysis
  - OS internals and design
  - Network, System, Memory, Compilers
  - ...

Conclusion

- Multi-faced problem that requires a broad set of techniques
    - ...
- Ranges from a “microscopic” level (flipping individual bits) to a “macroscopic” level (intelligence from billions of aggregated information)
    - And to look at the large scale solutions, you need to understand well the small details first

Conclusion

- Multi-faced problem that requires a broad set of techniques
    - ...
- Ranges from a “<u>microscopic</u>” level (flipping individual bits) to a “<u>macroscopic</u>” level (intelligence from billions of aggregated information)
    - And to look at the large scale solutions, you need to understand well the small details first
- The field is evolving rapidly… bringing new challenges!!
- Research in Malware Analysis has never been so interesting

Contact

davide.balzarotti@eurecom.fr

@balzarot

http://s3.eurecom.fr/~balzarot