SAPIENZA
UNIVERSITÀ DI ROMA

BlueTracer: a Robust API Tracer for Evasive Malware

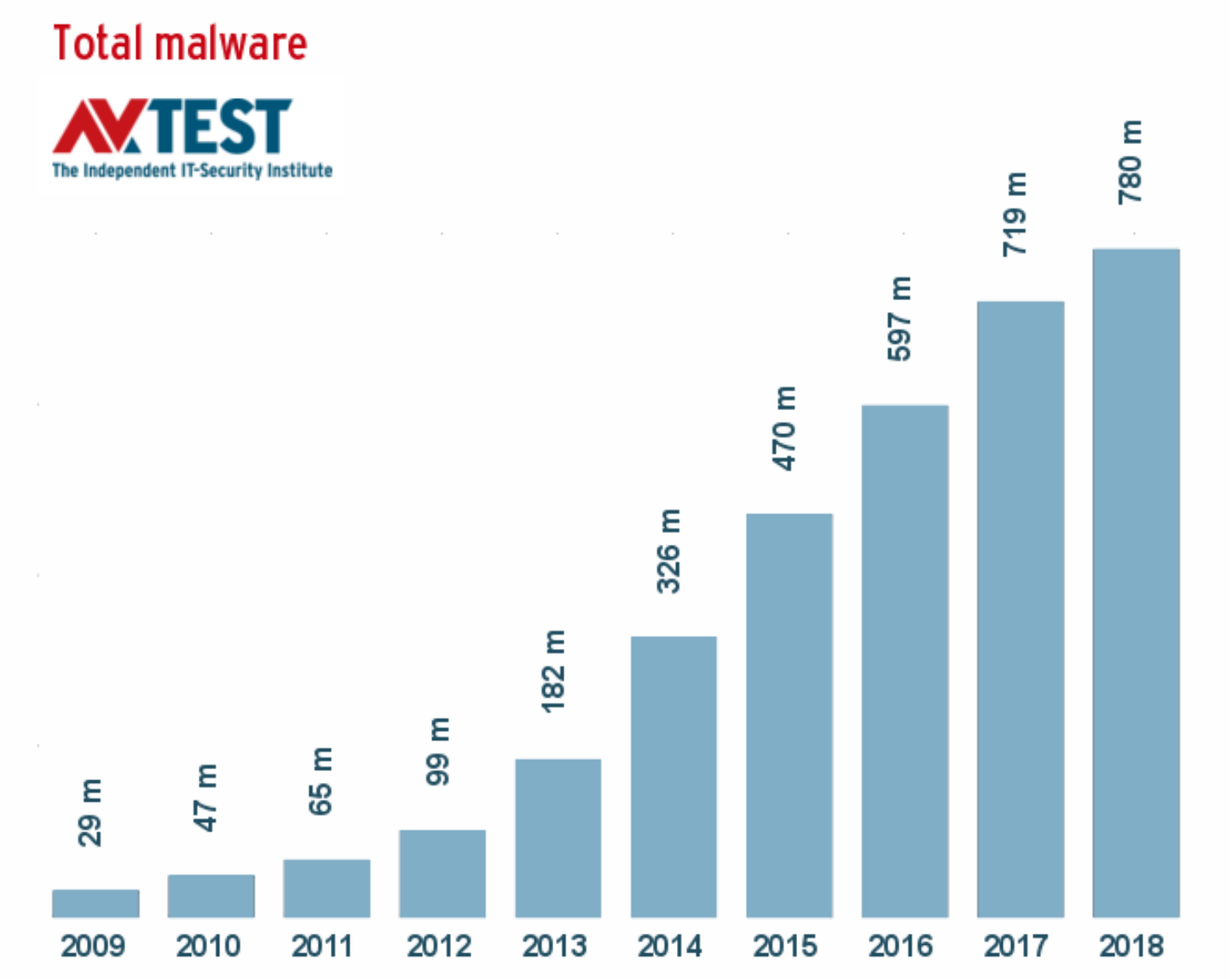
Simone Nicchi

Thesis Advisor: Prof. Camil Demetrescu
Thesis Co-Advisors: Dr. Daniele Cono D'Elia, Dr. Emilio Coppa

Master of Science in Engineering in Computer Science

July 20, 2018

Malware: an increasingly significant problem



Malware Analysis

Two main types:

- **Static Analysis:**
  involves the inspection of the different data and code sections of a binary
- **Dynamic Analysis:**
  the malware sample is executed and the actions it performs on the environment are observed

Dynamic analysis strongly favoured as it allows to dodge code obfuscations and deal with a large number of samples

Function call monitoring

- Functions can abstract implementation details providing a semantically richer representation of some functionality
- The abstractions embodied by **system calls** and **library calls** can be used to grasp the visible behavior of a malicious sample

Example:

```
RegCreateKey("...\CurrentVersion\Run\monitor")
CreateDirectory("C:\Windows\utils")
CreateFile("C\Windows\utils\GFypmMVqJQOEQqy.exe")
```

Implementation of function call monitoring

API Hooking

The interception of function calls provided by dynamically linked libraries (DLLs)

One technique is **Dynamic Binary Instrumentation (DBI)**.

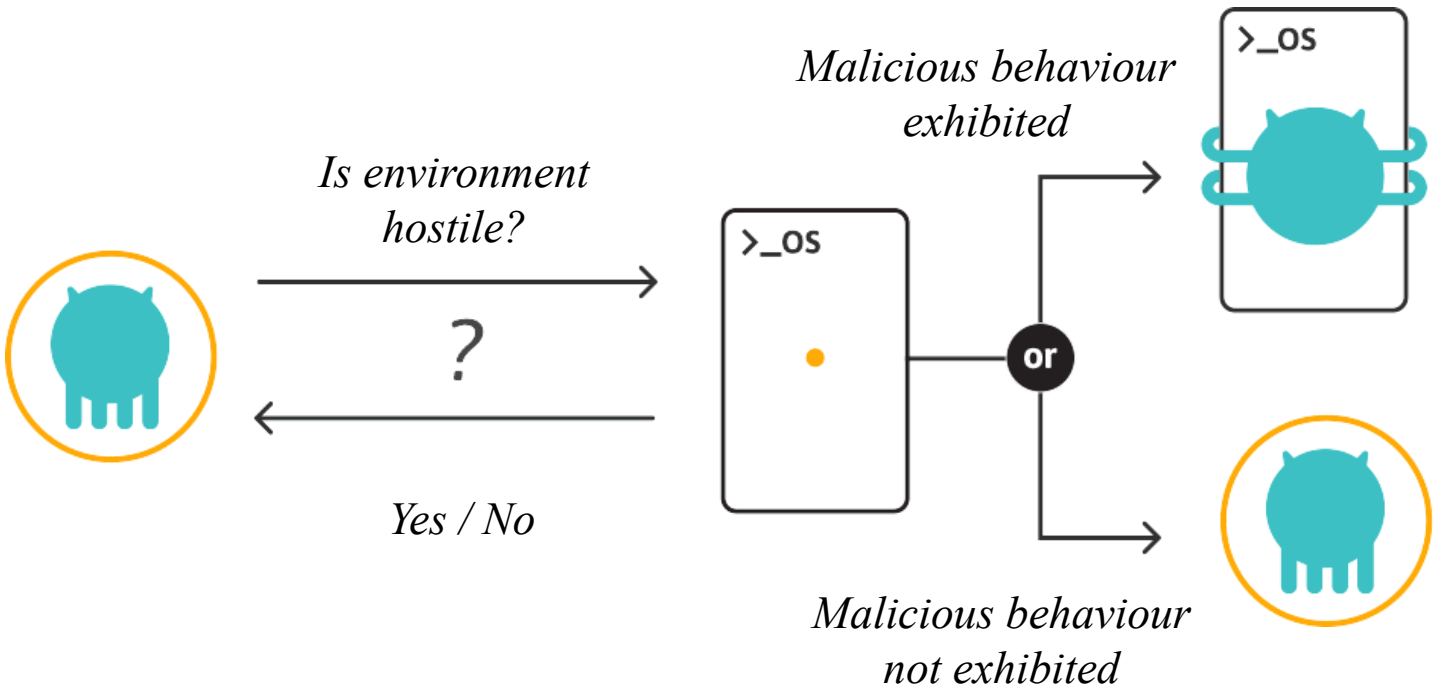
→ The behaviour of an application is inspected at run-time, without the need of recompiling it, via the injection of analysis code

```
record_before(libcall_name, arg1)
retval = libcall(arg1, &arg2)
record_after(retval, *arg2)
```

The threat posed by evasive malware

Evasive malware

Malware that conceals its harmful behaviour when detecting a hostile environment, such as a well-known sandbox solution



Existing problems

Two main problems which need to be addressed:

1. Existing API tracing tools have **limited logging capabilities**
   - The amount of recorded information is too little
   - It is hard to distinguish the calls made directly from the sample's main executable from the ones made by libraries

2. Current API hooking techniques are **easily detectable** and are not coupled with mechanisms to hide their presence from evasive malware

BlueTracer

BlueTracer is a robust library and system call tracer for Windows programs specialized in evasive malware

The tool possesses a remarkable logging power:

- Access to a source of calls related information is required
- Logging operations are made challenging by the heterogeneity of Windows libraries used in malware and the lack of well-structured documentation for their prototypes
  - → Integration of reliable external sources (**Dr. Memory** and **CISCO PyREBox**)

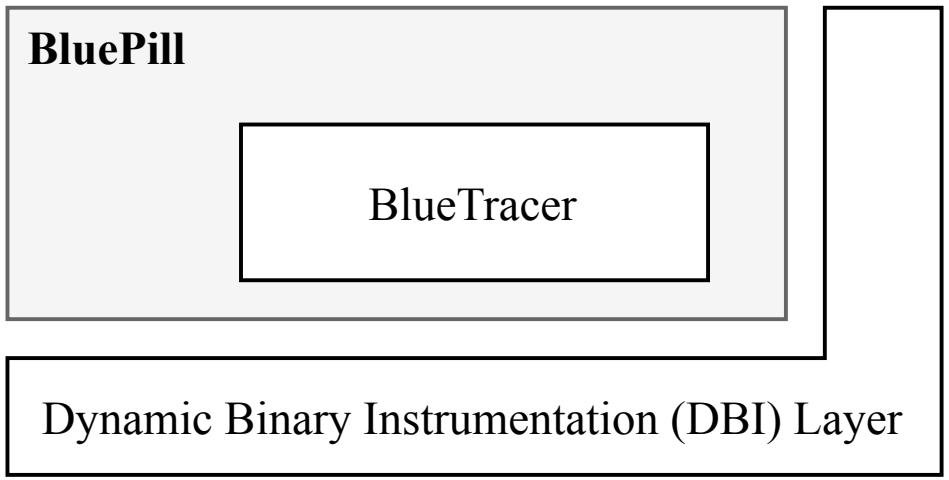
Key features:

- Undetected tracing of input parameters, output buffers and return values of over 17 000 system calls and library calls
- Logging of asynchronous events
- Allows to trace only calls from the main executable

BlueTracer (cont'd)

Solution to the detection problem:

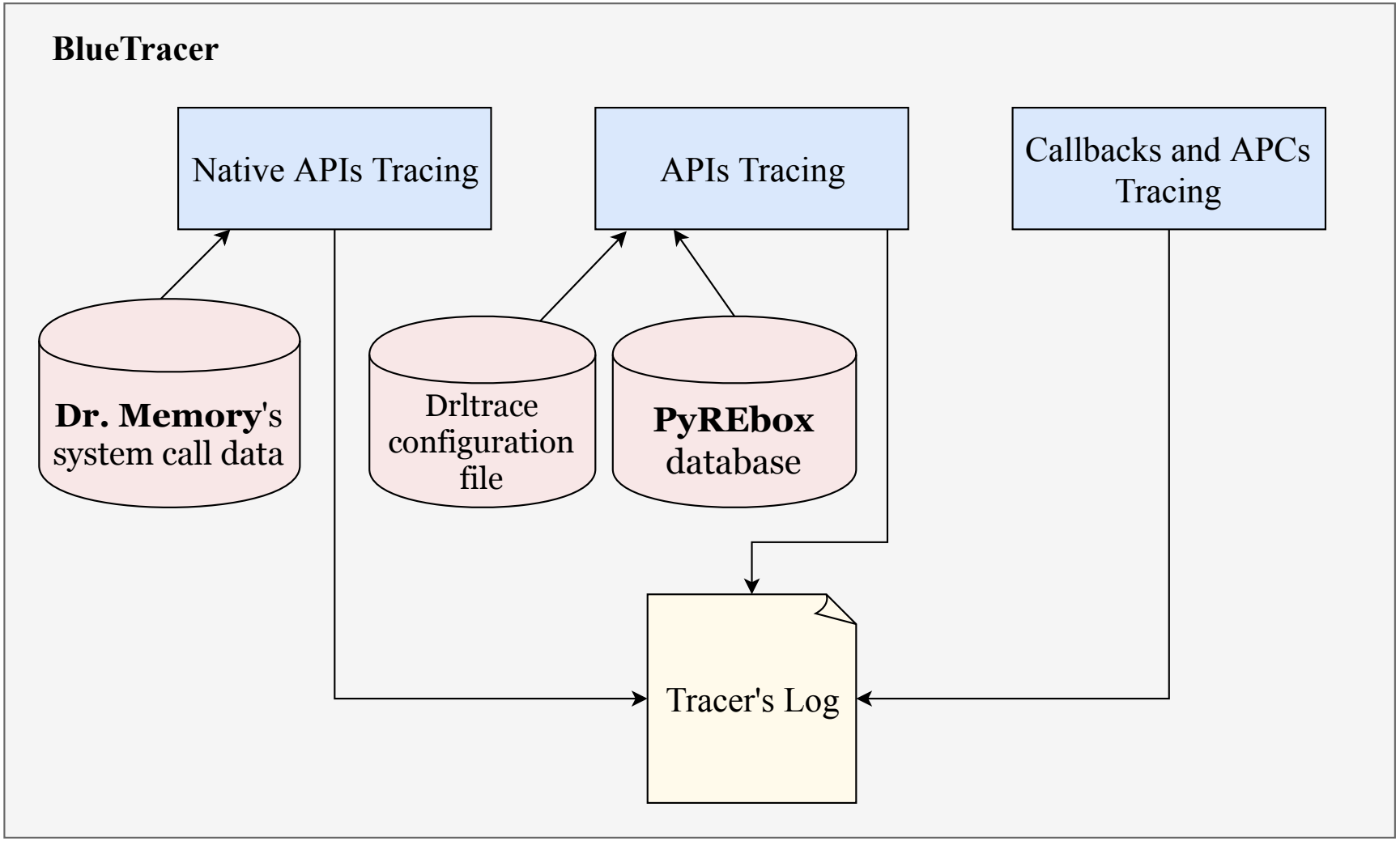
- Seamless integration with **BluePill**, a stealthy execution framework based on **Intel Pin**



Faced implementation challenges:

- Making best use of Intel Pin's capabilities with respect to run-time CPU and memory costs
- Addressing some inherent limitations of Intel Pin

BlueTracer's architecture

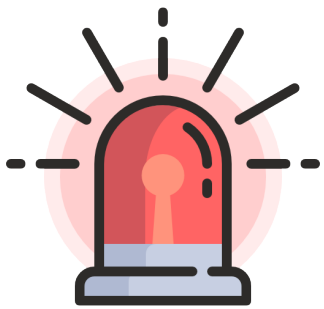


Evaluation with Al-Khaser

Al-Khaser is an open-source application which performs common checks employed by malware families to determine if they are being executed in an analysis environment.

Checks divided in categories:

- **Anti-Debugging**
- **Timing-based**
- **Human Interaction Detection**
- **Anti-Virtualization**
- **Anti-Analysis**



BlueTracer:

- remained undetected thanks to its integration with BluePill
- managed to track all the checks

Evaluation with evasive malware samples

Five highly evasive samples collected by Joe Security:

ID	MD5	Name
1	0af4ef5069f47a371a0caf22ae2006a6	*banker*
2	9437eabf2fe5d32101e3fbf9f6027880	*dropper*
3	cbdda646a20d95f078393506ecdc0796	*trojan*
4	cfdd16225e67471f5ef54cab9b3a5558	Olympic
5	ef694b89ad7addb9a16bb6f26f1efaf7	CCleaner

Evaluation was done manually and is a time-consuming process:

- Check if logs are congruous with Joe Security reports
- Process Monitor as ground truth for system activity

The logs collected by BlueTracer reveal behaviors consistent with the analysis reports authored by Joe Security

Example of tracked malevolent action

Tracing a particular action of a malware instance allows to understand in detail what the sample's intentions are

Example: dropping a malicious executable

```
~~1116~~ 138 kernel32.dll!CopyFileA
138  arg 0: c:\Users\Simuset\Desktop\sample1.exe
(name=lpExistingFileName, type=char*, size=0x1)
138  arg 1: C:\Windows\system32\†\ffpb6966.exe
(name=lpNewFileName, type=char*, size=0x1)
138  arg 2: 0x0 (name=bFailIfExists, type=(long/int), size=0x4)
138    executed kernel32.dll!CopyFileA =>
138  retval: 0x1 (name=Return value, type=(long/int), size=0x4)
```

Conclusions

Contribution:

Design and implementation of **BlueTracer**, a robust library and system call tracer for Windows programs specialized in evasive malware.

Future Developments:

- Automatic methodology for large-scale evaluation
- Improvement of logging capabilities
- Usage of log filtering techniques

Thank you for your attention!