SAPIENZA
UNIVERSITÀ DI ROMA

# BlueTracer: a Robust API Tracer for Evasive Malware

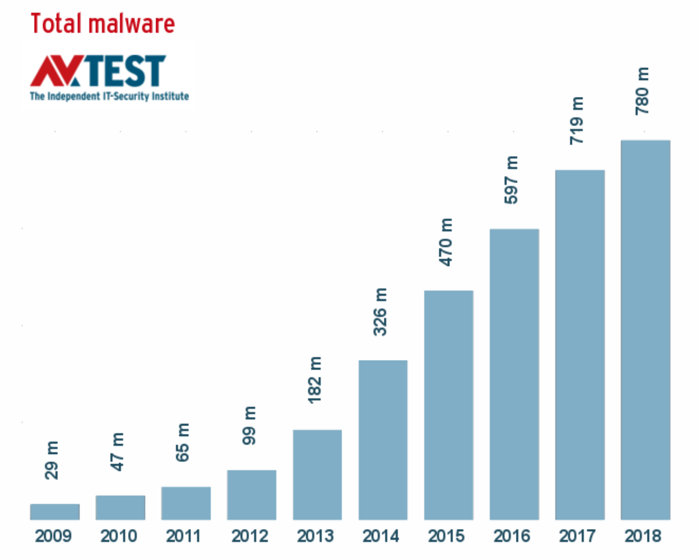
**Simone Nicchi**

*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.

Example:

`[2,4,1,3,5]` → `sort()` → `[1,2,3,4,5]`

The abstractions embodied by **system calls** and **library calls** can be used to grasp the visible behavior of a malicious sample

# Implementation of function call monitoring

## API Hooking

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

Three broad categories:

- Binary Rewriting
    - Call Redirection
    - Function Rewriting
- Virtual Machine Introspection (VMI)
- **Dynamic Binary Instrumentation (DBI)**

# Dynamic Binary Instrumentation (DBI)

A dynamic binary analysis technique in which the behaviour of an application is inspected at run-time via the injection of analysis code.

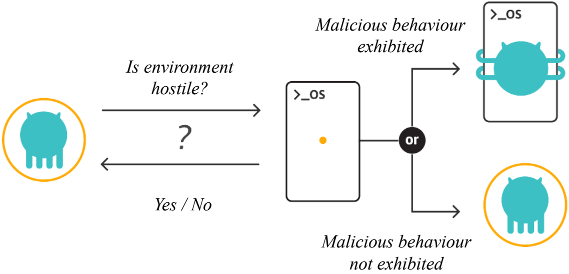
```
record(arg1)
retval = libcall(arg1, &arg2)
record(retval, *arg2)
```

**Problem 1**: existing products have limited logging capabilites

# 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


*Is environment hostile?*
?
*Yes / No*
>_OS
or
*Malicious behaviour exhibited*
>_OS
*Malicious behaviour not exhibited*

**Problem 2**: API hooking techniques in literature are not coupled with mechanisms to hide their presence from evasive malware

# Our solution: BlueTracer

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

Implementation details:

- Based on the **Intel Pin** DBI framework
- Integrated with the **BluePill** stealthy execution framework
- Combines reliable external sources of prototypes information

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
- Resolution of named constants

# Why Intel Pin ?

Characteristics:

- **User-friendliness**
- **Portability**
- **Transparency**
- **Efficiency**

**Analysis routines:** embody the code to be inserted during the application's execution

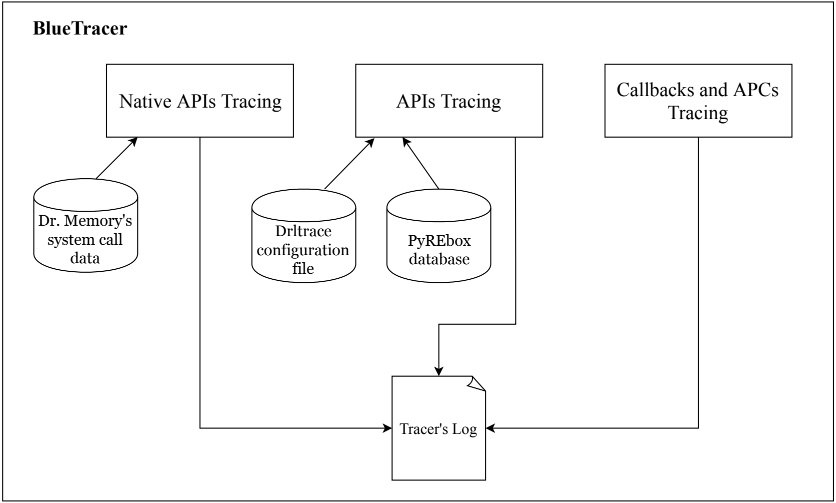
**Instrumentation routines:** determine where the analysis code has to be placed

Different analysis and instrumentation granularities

- Instruction, trace, routine and image

# Integration with BluePill

# BlueTracer's architecture


BlueTracer
Native APIs Tracing
APIs Tracing
Callbacks and APCs Tracing
Dr. Memory's system call data
Drltrace configuration file
PyREbox database
Tracer's Log

# Al-Khaser

# Sample tracked evasive check

# Evasive malware samples

# Conclusion and future developments

Thank you for your attention!