Behavior Analysis for Vulnerability and Malware Detection

Ph.D. Candidate:

Razvan Raducu Teodorescu

Universidad de Zaragoza, Zaragoza, Spain

Advisors:

Prof. Dr. Pedro Álvarez

Prof. Dr. Ricardo J. Rodríguez

July 16, 2025

Agenda

1. Context
2. Static Approach
3. Dynamic Approach
4. Conclusions
5. Future Work and Open Directions

Universidad Zaragoza

Agenda

1 Context

2 Static Approach

3 Dynamic Approach

4 Conclusions

5 Future Work and Open Directions

Context

- **Cyberattacks keep increasing**
  - In 2024 global cyberattacks increased 44%[1]
  - **Malware** is usually involved
- **Malware keeps evolving**
  - Ever-increasing sophistication: metamorphism, polymorphism, packing, obfuscation, LOLBins, ...
- Malware exploits a **plethora of attack vectors**
  - For instance, source code vulnerabilities or covert malware executions
- Despite all the efforts, we still need defensive tools
- **Detect potentially malicious behaviors**
  - Source code level (static analysis)
  - Executing the program[2] (dynamic analysis)

[1] Source: The State of Cyber Security 2025. Check Point Research.

[2] In this presentation we use the terms program, binary, and sample interchangeably.

Context

Motivation

- Static approaches cannot keep up with malware evading detection
  - Need to understand how **malware actually behaves**, not how it looks
- Vulnerabilities remain **exploitable decades later**
- Behavioral analysis requires **structured, adaptable definitions**
  - Expandable and **modifiable as malware tactics evolve**
  - Reusable across malware families
  - Serve as semantic features for automatic classification
- Dynamic analysis tools are **inconsistent**
  - Malware sandboxes differ in trace quality, transparency, and usability
- **Lack of high-quality behavioral data**
  - Public datasets often **lack crucial contextual information**, hindering behavioral analysis and reproducible research

Context

Objectives

- Develop methodologies to **identify potentially malicious behaviors** originating from **source code vulnerabilities** or **malicious activities during execution**
- Enable **practical**, **reliable** **behavioral detection**
  - **Promote open science and reproducibility through open source tools and data**
  - **Support triage** and **human-readable decisions** over binary verdicts
  - Establish a foundation for **explainable detection systems**

Universidad Zaragoza 1474

Research Questions

Research Question 1

To what extent the detection of behavior patterns through static analysis helps preventing the malicious activity?

Universidad Zaragoza

Research Questions

Research Question 1

To what extent the detection of behavior patterns through static analysis helps preventing the malicious activity?

Research Question 2

How do modern sandbox environments support dynamic analysis for capturing runtime behaviors of binaries in diverse scenarios?

Research Questions

Research Question 1

To what extent the detection of behavior patterns through static analysis helps preventing the malicious activity?

Research Question 2

How do modern sandbox environments support dynamic analysis for capturing runtime behaviors of binaries in diverse scenarios?

Research Question 3

To what extent the available execution trace datasets (if any) are comprehensive and suitable for behavioral analysis?

Research Questions

Research Question 1

To what extent the detection of behavior patterns through static analysis helps preventing the malicious activity?

Research Question 2

How do modern sandbox environments support dynamic analysis for capturing runtime behaviors of binaries in diverse scenarios?

Research Question 3

To what extent the available execution trace datasets (if any) are comprehensive and suitable for behavioral analysis?

Research Question 4

To what extent are we able to detect malicious behavior patterns through dynamic analysis?

Agenda

1 Context

2 Static Approach

3 Dynamic Approach

4 Conclusions

5 Future Work and Open Directions

Static Approach - RQ1

- Early stages of software development
- **Time-of-check to time-of-use (TOCTOU)**
  - Source-code race condition
  - Example of malicious or vulnerable **behavior pattern**
  - **Sequence of specific functions referencing same object**
  - Common in Unix-like systems
  - First documented in the mid 70s
- We reviewed the literature looking for **defenses** and **attacks**

CWE ID	**Vulnerability**
CWE-59	*Improper Link Resolution Before File Access (‘Link Following’)*
CWE-61	*UNIX Symbolic Link (Symlink) Following.*
CWE-62	*UNIX Hard Link*
CWE-362	*Concurrent Execution using Shared Resource with Improper Synchronization (‘Race Condition’)*
CWE-363	*Race Condition Enabling Link Following*
CWE-367	*Time-of-check Time-of-use (TOCTOU) Race Condition* (not only file-based TOCTOU)
CWE-386	*Symbolic Name not Mapping to Correct Object*
CWE-706	*Use of Incorrectly-Resolved Name or Reference*

Static Approach - RQ1

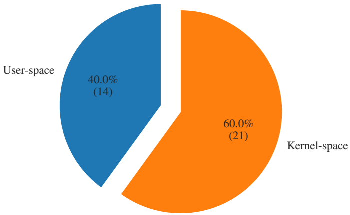
The TOCTOU vulnerability

- Present in many domains. **We focus on file-system based TOCTOU**
- Occurs when **referencing filesystem objects by their *filename* or *path***
  1. Checking a condition (Time of Check)
  2. Operate based on that condition (Time of Use)
- **Operations are commonly not atomic** → **vulnerability window** between them
- Allows **privilege escalation**

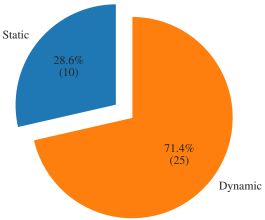
```c
char *filename = argv[1];

if(!access(filename, W_OK)){ // Check permissions (Time of Check)
    // Vulnerability window
    file = fopen(filename, "a+"); // Open the file (Time of Use)

    // Write to file (untrusted input)
    fwrite(buffer, sizeof(char), strlen(buffer), file);
    fclose(file);
} else {
    printf("No permission, exiting!\n");
}
```


Static Approach - RQ1

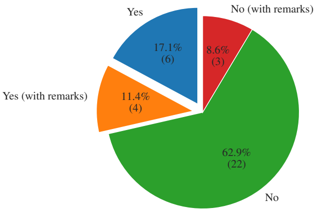
- 504 papers initially found, **41 peer-reviewed papers chosen** (37 defenses and 4 attacks)
- We proposed a **taxonomy of defense mechanisms and attack techniques** against TOCTOU
  - Memory region the defense/attack takes place
  - Time of detection/exploitation
- We examined various aspects of the proposed defenses:
  - Reproducibility
  - Feasibility, based on the metadata of filesystem objects used



(a) Detection location



(b) Detection time

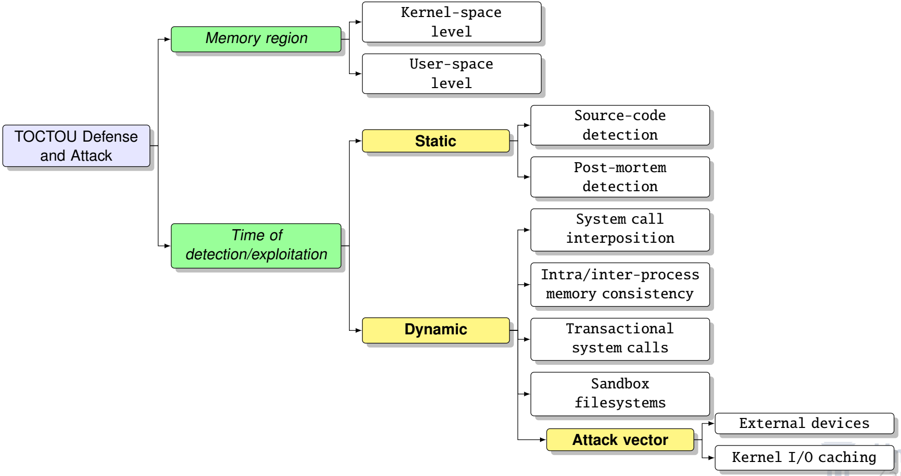


(c) Reproducibility

Graphical summary of defense solutions.

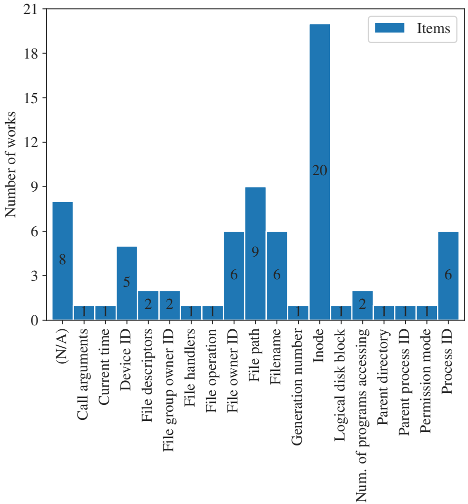
Static Approach - RQ1

Proposed Taxonomy



Static Approach - RQ1

Defenses Feasibility



Filesystem	Reutilization of inode
BTRFS	✗
EXT2	✓
EXT3	✓
EXT4	✓
FAT16	✗
FAT32	✗
NTFS	✗
HFS+	✗
JFS	✗
NILFS2	✓
REISERFS	✓
XFS	✓
RAMFS	✗
TMPFS	✗

Universidad Zaragoza

Conclusions - RQ1

To what extent the detection of behavior patterns through static analysis helps preventing the malicious activity?

- **Static identification may reduce exploitation risk**
- Wide variety of proposed defenses
- Some proposed defenses are either unusable or unfeasible
- **Universal solution unlikely to be found** and would involve
  - ***A new race-free, security-focused API*** (or modification of the current one)
  - ***Modification of the kernel*** to always work with file descriptors
  - ***Transition to transactional filesystems*** where operations are guaranteed to be atomic
- **Kernel/API redesign would break compatibility**
- TOCTOU is still exploitable and remains a challenge
  1. Non-determinism of TOCTOU
  2. Influence of external factors
  3. Source code access
  4. Backwards compatibility issues
- Responsibility usually falls on the developers

Agenda

1 Context

2 Static Approach

3 Dynamic Approach

4 Conclusions

5 Future Work and Open Directions

Universidad Zaragoza

Transition from Static to Dynamic

Limitations of our Static Approach

- Needs source code (often unavailable)
- May help but there are no guarantees
- Focused on a single specific vulnerability
- Cannot observe runtime behavior

Universidad Zaragoza 1474

[3] Sources: Security Report 2015/16; Security Report 2019/20; Distribution of malware and PUA by OS as of June 2025

Transition from Static to Dynamic

Limitations of our Static Approach

- Needs source code (often unavailable)
- May help but there are no guarantees
- Focused on a single specific vulnerability
- Cannot observe runtime behavior

Shift toward Dynamic Analysis

- Broader and more flexible approach
- Interactions with the environment
- Behavior patterns exhibited during execution
- **Focus on Windows OS** – main malware target for the past 10 years[3]

[3] Sources: Security Report 2015/16; Security Report 2019/20; Distribution of malware and PUA by OS as of June 2025

Dynamic Approach - Sandboxes (RQ2)

What sandboxes exist and which are their features?

- **Create a practical guide to help identify the most suitable sandboxing technology**
- Surveyed the Internet looking for available sandboxes
  - Both academic and gray literature
  - At least support for Windows
  - Free or freemium
  - Up to date (maintained in the last 2 years)

Universidad Zaragoza

Dynamic Approach - Sandboxes (RQ2)

What sandboxes exist and which are their features?

- **Create a practical guide to help identify the most suitable sandboxing technology**
- Surveyed the Internet looking for available sandboxes
  - Both academic and gray literature
  - At least support for Windows
  - Free or freemium
  - Up to date (maintained in the last 2 years)

Survey results

- 10 malware sandboxes selected
  - Open source: `CAPEv2`, `Cuckoo3`, `DRAKVUF` (incl. `DRAKVUF Sandbox`), and `Noriben`
  - Commercial: `ANY.RUN`, `Hybrid Analysis`, `Joe Sandbox`, `Tria.ge`, `Filescan.IO`, and `Threat.Zone`
- Considered features: configurability, usability, capabilities, and privacy

Dynamic Approach - Sandboxes (RQ2)

		Open Source				Commercial					
		CAPE	Cuckoo3	DRAKVUF	Noriben	ANY.run	Hybrid Analysis	Joe Sandbox	Tria.ge	Filescan.IO	Threat.Zone
Installation Integration Configuration	Local deployment	✓	✓	✓	✓		✓$	✓$		✓$	
	Online service		✓			✓	✓	✓	✓	✓	✓
	Config. required	✓	✓	✓	✓						
	Free	✓	✓	✓	✓						
	Freemium					✓	✓	✓	✓	✓	✓
	Automation	✓	✓			✓$	✓	✓	✓	✓	
	API	✓	✓	✓•		✓$	✓	✓	✓‡	✓	✓$
	Configurable	✓	✓	✓	✓	✓	✓		✓		
	Documentation	✓	✓	✓		✓			✓		
Supported OS Capabilities	Windows	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Linux	✓		✓°		✓	✓	✓	✓	✓	✓$
	macOS						✓	✓$	✓		✓$
	Android					✓	✓	✓$	✓	✓	✓$
	iOS										
	Other OS										
	Interactive	✓			✓	✓	✓$	✓	✓		✓
	URL Analysis	✓	✓			✓	✓	✓	✓	✓	✓
	Behavioral trace	✓		✓	✓		✓	✓	✓		
	User space	✓	✓		✓	?	?	?	?	?	
	Kernel space					?	✓	?	?	?	
	Hypervisor level			✓		?		✓$	?	?	✓
	Anti-evasion	✓	?	✓	✓	?	✓	?	?	✓	?
	Detection	✓	✓			✓	✓	✓	✓	✓	✓
	Classification	✓	✓			✓	✓	✓	✓	✓	✓
Implementation	Community-backed	✓	✓	✓	✓					✓	
	Industry-backed					✓	✓	✓	✓	✓	✓
	Government-backed		✓	✓•							
	Privacy★	U	U†	U	U	D	D	D	D	D	D

★D: Disclosed, U: Undisclosed; †Undisclosed if deployed locally, otherwise disclosed; °DRAKVUF; •DRAKVUF Sandbox; $Only paid plans; ‡Free upon approval (may not be eligible); ? No information

Dynamic Approach - Sandboxes (RQ2)

Sandbox Suitability

	Open Source Sandboxes	Commercial Sandboxes
Budget constraints	• Free to use • Supported by active communities	• Premium requires financial investment • Free plans are constrained
Time limitations	• Challenging setup and maintenance, potential delays	• Ready to use • Vendor support
Customization requirements	• Fully customizable with source code access	• Limited to out-of-the-box functionality
Technical expertise	• Needs strong technical skills	• User-friendly, preconfigured environments
Privacy concerns	• Local control ensures privacy	• Free plans may expose submitted data

Conclusions - RQ2

How do modern sandbox environments support dynamic analysis for capturing runtime behaviors of binaries in diverse scenarios?

- **Sandboxes have become an essential tool** and vary widely
  - **Open source** sandboxes
    - Flexibility, transparency, and free; but hard to install and maintain
  - **Commercial** sandboxes
    - Ease of use and user experience; but limited customization, privacy, and capped free version
- **No universal best sandbox** as suitability is context-dependent
- Trade-offs between **privacy**, **ease of use**, **flexibility**, and **budget**
- **Limited** support for mobile platforms
- **No iOS-compatible** sandboxes were identified
- We selected **CAPEv2 as our analysis platform**
  - Windows focused
  - Rich reports, support automation, active community, . . .
  - Open source – adapted to our needs

Universidad Zaragoza

Dynamic Approach - Execution Trace Datasets (RQ3)

Are existing execution trace datasets suitable for behavioral analysis?

- An execution trace is a record of a program's **behavior during runtime**
  - **System and API calls**
  - Contextual information
  - Generated by running and monitoring the program
- Execution traces capture **the actual program behavior**
- We **focus on detecting patterns of potentially malicious behaviors**
- Understanding available execution trace data is crucial

Universidad Zaragoza

Dynamic Approach - Execution Trace Datasets (RQ3)

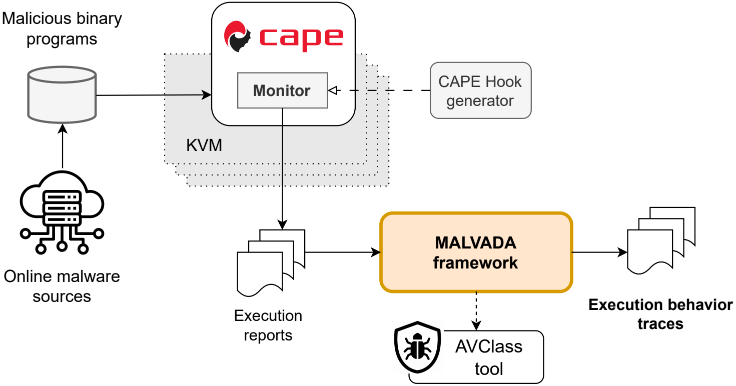
Datasets of Windows malware execution traces

- **Very few publicly available** datasets and tools to generate them
- Typically **contain only sequences of API names** or numerical IDs
  - Usually **optimized for AI**-based approaches
- This simplified representation often **omits critical contextual information**
  - API parameters, return values, created processes, mutant objects, . . .
- Tend to include a **limited number of malware families**

Dynamic Approach - Execution Trace Datasets (RQ3)

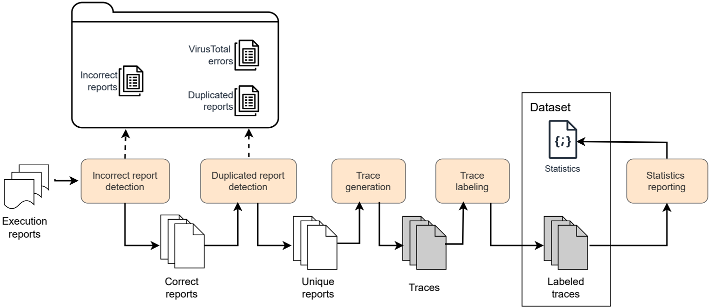
MALVADA

- **Help generating execution trace datasets**
- **Fully open source**
  - Customizable, adaptable, and extensible



Dynamic Approach - Execution Trace Datasets (RQ3)

- MALVADA has different phases
  1. Incorrect report detection
  2. Duplicate report detection
  3. Execution trace generation
     - Sanitizes reports (deletes sandbox info)
     - Anonymizes sensitive information
  4. Trace labeling
  5. Statistic reporting
- Works as a fully automated pipeline



Dynamic Approach - Execution Trace Datasets (RQ3)

The WinMET dataset

- **Win**dows **M**alware **E**xecution **T**races
- Generated with CAPEv2 and MALVADA
- Rich execution trace dataset
  - ~10K execution traces
- **Overcomes limitations of existing datasets**
  - Includes API parameters, return values, and system interactions
- **Publicly available**

Universidad Zaragoza

Conclusions - RQ3

To what extent the available execution trace datasets (if any) are comprehensive and suitable for behavioral analysis?

- **Few** publicly available **execution trace datasets** and **few tools** to generate them
- **Public datasets often lack relevant information** (e.g., syscall return values, arguments)
  - Insufficient for comprehensive behavior analysis
  - Many are unmaintained or focus on outdated malware samples
- We developed **MALVADA**
  - Designed to help generating execution trace datasets
  - Customizable and extensible
- We developed the **WinMET** dataset, ~10K execution traces
  - Execution traces with all contextual information
  - Enables meaningful behavioral analysis and model training
- We are already working on WinMET v2, ~32K execution traces

Universidad Zaragoza 1474

Dynamic Approach - Behavior Detection (RQ4)

Are we able to detect behavior patterns?

- **Main objective: help analysts understand unknown program behavior**
- **Detect and quantify behavior patterns**
- We developed the Windows Behavior Catalog (WBC)
- We developed MalGraphIQ, a pattern matching tool
- We use part of WinMET to validate our approach

Dynamic Approach - Behavior Detection (RQ4)

The Windows Behavior Catalog (WBC)

- Based on **MITRE's Malware Behavior Catalog** manually implemented and executed in CAPE
- Defines specific **behavior patterns** **(API/syscall sequences)**
  - `NtCreateKey` → `NtSetValueKey` → `RegCloseKey`
- Structured 3-level hierarchy (**micro-objectives**, **micro-behaviors**, methods)
  - 6 micro-objectives, 30 micro-behaviors, 87 methods, and 329 behavior patterns
- Created our own **Windows API and Syscall categorization**
  - Comprehensive, including old and new API and system calls

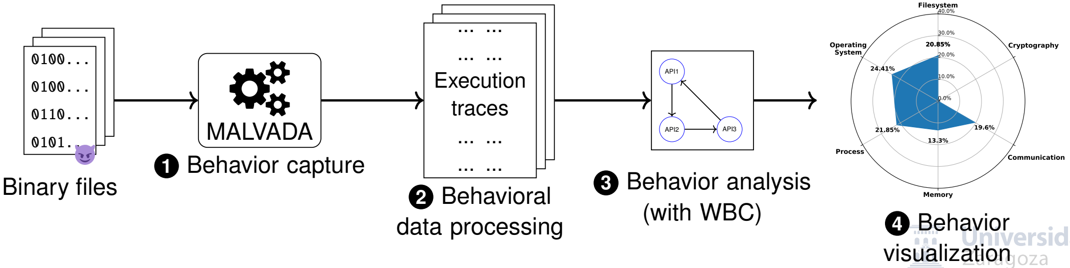
Micro-objective	Micro-behaviors
FILESYSTEM	Alter Filename Extension, Create or Open File, Copy File, Create Directory, Delete File, Get File Attributes, Read File, Write File, Move File
CRYPTOGRAPHY	Encrypt Data, Encryption Key, Cryptographic Hash, Decrypt Data
COMMUNICATION	Socket, HTTP, WinINet
MEMORY	Allocate Memory, Change Memory Protection
PROCESS	Create Process, Create Thread, Create Mutex, Check Mutex, Resume Thread, Suspend Thread, Enumerate Threads, Open Process, Open Thread, Process Enumeration
OPERATING SYSTEM	Environment Variable, Registry

Universidad Zaragoza

Dynamic Approach - Behavior Detection (RQ4)

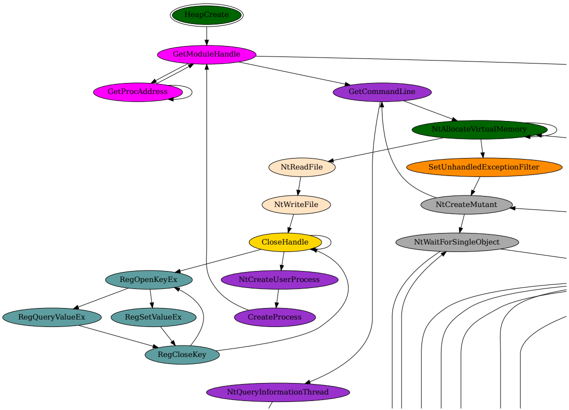
MalGraphIQ

- **Graph-based** analysis system
- Generates **behavior visual representations**
  - Execution graphs, category graphs, radar charts, and bar plots
- Matches **WBC patterns against category graphs**
- Computes the **occurrences** and **plots** them

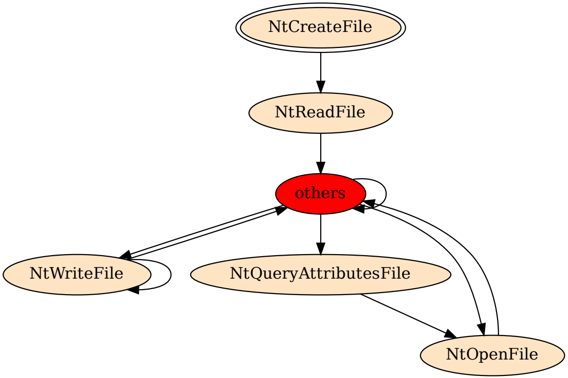


Dynamic Approach - Behavior Detection (RQ4)

Visualization examples

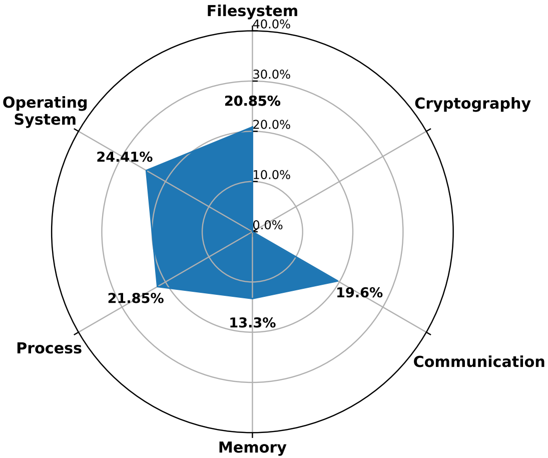


Cropped section of a behavior graph.

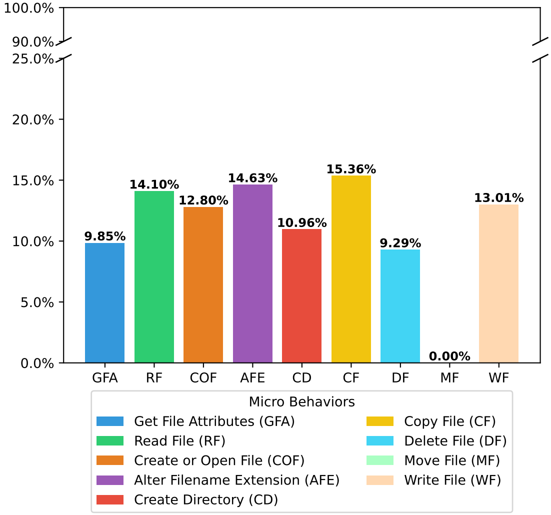


FILESYSTEM category graph.

Dynamic Approach - Behavior Detection (RQ4)



Micro-objectives.



FILESYSTEM micro-behaviors.

Dynamic Approach - Behavior Detection (RQ4)

Experimental Evaluation

- Assess MalGraphIQ
  - Effectively detects behavior patterns
  - Correlation with malware families or types
  - Suitability to detection/classification techniques
- Samples of known malware
  - Anticipate expected results
  - Validate the system

Dynamic Approach - Behavior Detection (RQ4)

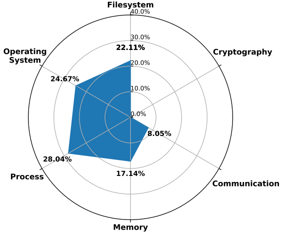
Experimental Evaluation

- Assess MalGraphIQ
  - Effectively detects behavior patterns
  - Correlation with malware families or types
  - Suitability to detection/classification techniques
- Samples of known malware
  - Anticipate expected results
  - Validate the system

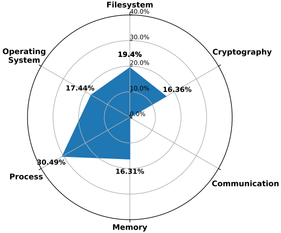
Experimental Setup

- Analyzed 249 malware samples spanning 4 malware families and 3 types
  - Loader: `GCleaner` and `GuLoader`
  - POS RAM Scraper: `Alina`
  - Ransomware: `Petya`
- Evaluation of behavior vectors
  - μ*Obj* - 6 dimensions, μ*Beh* - 30 dimensions, and *Both* - 36 dimensions
  - κ-fold cross-validation (κCV) with $\kappa = 5$ and $\kappa = 10$
  - Cosine similarity, Euclidean distance, and Manhattan distance

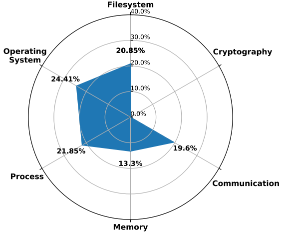
Dynamic Approach - Behavior Detection (RQ4)



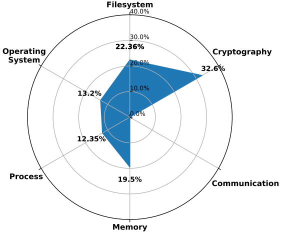
Gcleaner



Guloader



Alina



Petya

Universidad Zaragoza

Dynamic Approach - Behavior Detection (RQ4)

Micro-average performance metrics of the κCV.

		$k=5$		$k=10$	
	Metric	*Acc*	*Prec*, *Rec*, *F*1*	*Acc*	*Prec*, *Rec*, *F*1*
μ*Obj*	Cosine	0.87	0.75	0.86	0.73
	Euclidean	0.87	0.75	0.86	0.72
	Manhattan	0.87	0.74	0.86	0.72
μ*Beh*	Cosine	0.93	0.86	0.94	0.88
	Euclidean	0.94	0.88	0.95	0.90
	Manhattan	0.94	0.87	0.94	0.87
Both	Cosine	0.95	0.90	0.95	0.91
	Euclidean	**0.96**	**0.92**	**0.96**	**0.92**
	Manhattan	0.94	0.89	0.94	0.89

*Precision (*Prec*), Recall (*Rec*) and F1-Score (*F*1) have the same value.

Universidad Zaragoza

Conclusions - RQ4

To what extent are we able to detect malicious behavior patterns through dynamic analysis?

- We developed the **Windows Behavior Catalog (WBC)**
  - **Defines a structured hierarchy of Windows behaviors**
  - Behaviors are defined as **sequences of system and API calls**
  - Enables **identification of both benign and malign behaviors**
- We developed **MalGraphIQ**
  - **Identify behavior patterns** in execution traces
  - Generated different visualizations
  - Assisting **initial stages of analysis and triage**
- Our **experiments prove the viability** of our approach
  - Discerning between different malware types and families
  - Higher dimensions usually imply higher accuracy
- Visualizations help identify unique and common behaviors (**VirusTotal expressed their interest**)
- MalGraphIQ, WBC, and our categorization of Windows API and Syscalls are **open source**

Universidad Zaragoza 1474

Agenda

1 Context

2 Static Approach

3 Dynamic Approach

4 Conclusions

5 Future Work and Open Directions

Research Contributions

- **Generate knowledge, data, and tooling for behavior detection**
  - Structured, data-driven, and explainable
  - Foundation for future research and industry adoption
  - Open source ecosystem

Research Contributions

- **Generate knowledge, data, and tooling for behavior detection**
  - Structured, data-driven, and explainable
  - Foundation for future research and industry adoption
  - Open source ecosystem

Scientific Contributions

- TOCTOU Systematic Literature Review
- Malware Sandbox Study
- Windows Behavior Catalog
- WinMET Dataset
- Behavior Detection Approach

Side-product Research

- MALVADA Framework
- WinMET Dataset
- MalGraphIQ
- Windows API and Sysalls Categories
- CAPE Hook Generator

Research Papers

- **Peer-reviewed Publications:**
  - R. Raducu, R. J. Rodríguez, and P. Álvarez, "*Defense and Attack Techniques Against File-Based TOCTOU Vulnerabilities: A Systematic Review*," in IEEE Access, vol. 10, pp. 21742-21758, 2022, DOI: 10.1109/ACCESS.2022.3153064. Impact factor (JCR): 3.9 (3.6 without self citations), rank 73/158 (**Q2**; *Computer Science, Information Systems*).
  - R. Raducu, A. Villagrasa-Labrador, R. J. Rodrígue,z and P. Álvarez, "*MALVADA: A framework for generating datasets of malware execution traces*," in SoftwareX, vol. 30, 2025, DOI: 10.1016/j.softx.2025.102082. Impact factor (JCR): 3.4 (3.3 without self citations), rank 38/108 (**Q2**; *Computer Science, Software Engineering*).
- **Under Review:**
  - R. Raducu, R. J. Rodríguez, and P. Álvarez, "*A Graph-Based Dynamic Analysis System for Behavior Detection in Windows Applications*". Currently *under review*, submitted to The Computer Journal. Impact factor (JCR): 1.5 (1.4 without self citations), rank 71/144 (**Q2**; *Computer Science, Theory & Methods*).
  - R. Raducu, R. J. Rodríguez, and P. Álvarez, "MalGraphIQ: a tool for generating behavior representations of malware execution traces". Currently *under review*, submitted to SoftwareX. Impact factor (JCR): 3.4 (3.3 without self citations), rank 38/108 (**Q2**; *Computer Science, Software Engineering*).
- **Awaiting Submission:**
  - R. Raducu, R. J. Rodríguez, P. Álvarez, and A. Zarras,"*The Sandbox Reloaded: A Guide to Modern Malware Sandbox*". Submitted to the 20th International Conference on Availability, Reliability and Security (ARES 2025), but was not accepted. We are currently expanding the work and will submit it to a journal, yet to be determined.

Other Academic Work

- **Conference Contributions:**
  - R. Raducu, R. J. Rodríguez, and P. Álvarez. “Towards a Web System for the Evaluation of Resource Consumption,” in *Jornadas de Concurrencia y Sistemas Distribuidos* 2021 (JCSD 2021).
  - R. Raducu, R. J. Rodríguez, and P. Álvarez. “Model-Based Analysis of Race Condition Vulnerabilities in Source Code,” in *Jornadas Nacionales de Investigación en Ciberseguridad* 2022 (JNIC’22).
  - R. Raducu, R. J. Rodríguez, and P. Álvarez. “A Review of Defense and Attack Techniques Against File-Based TOCTOU Vulnerabilities: A systematic Review,” in *Jornadas Nacionales de Investigación en Ciberseguridad* 2023 (JNIC’23).
  - R. Raducu, R. J. Rodríguez, P. Álvarez, and A. Zarras. “Malware Sandbox Comparison (Work in Progress),” in International Conference on Digital Forensic Analysis and Exploitation 2025 (DFex 2025).
- **Technical Program:**
  - Organizing Committee for the Digital Forensics Research Conference Europe (DFRWS EU 2024) as a Rodeo (Capture The Flag competition) Chair.
  - Technical Program Committee member for the *Jornadas Nacionales de Investigación en Ciberseguridad* 2025 (JNIC’25).

Other Academic Work

- **Reviewer:**
  - IEEE 26th International Conf. on Emerging Technologies and Factory Automation (ETFA 2021).
  - The Computer Journal, 2025.
  - *Jornadas Nacionales de Investigación en Ciberseguridad* 2025 (JNIC'25).
- **Research Stays:**
  - September 2023 to October 2023 (1 month) and September 2024 to November 2024 (2 months) at the Systems Security Laboratory from the Department of Digital Systems, University of Piraeus, Greece. Supervised by Dr. Apostolis Zarras.
- **Final Bachelor Degree Project Co-supervision:**
  - Julia Varea Palacios. B. S. in Informatics Engineering, February 2024. *Malware detection using machine learning techniques.* Co-supervised with Prof. Pedro Álvarez.
  - Salomé Rea Ávila. B. S. in Informatics Engineering, December 2024. *Detection of vulnerabilities in source code through Petri nets.* Co-supervised with Prof. Ricardo J. Rodríguez.
  - Daniel Jal Burguete. B. S. in Informatics Engineering, *in progress*. *Malware description and classification based on execution behavior.* Co-supervised with Prof. Pedro Álvarez.

Open Source Contributions

- **Projects:**
  - 460+ 110+ **WinMET** dataset; DOI: 10.5281/zenodo.12647555
  - 3 2 **MALVADA**; GitHub: MALVADA
  - 2 0 **CAPE Hook Generator**[4]; GitHub: cape-hook-generator
  - 16 1 **Windows API and Syscall Categories**; GitHub: winapi-categories
  - **Windows Behavior Catalog (WBC)**; GitHub: windows-behavior-catalog
  - **MalGraphIQ**; GitHub: MalGraphIQ
- **Contributions:**
  - CAPEv2 Sandbox
  - CAPE's Monitor (capemon)
  - Malware Behavior Catalog (MBC)
  - Issues and PRs in several repositories

[4] Included in CAPESandbox Community Repo

Agenda

1 Context

2 Static Approach

3 Dynamic Approach

4 Conclusions

5 Future Work and Open Directions

Universidad Zaragoza

Ongoing and Future Work

Ongoing and Future Work

- Test sandboxes for anti-evasion capabilities
- Performance improvements of MALVADA and MalGraphIQ
- Graph manipulation
- Refinement of matching algorithm to consider API/syscall parameters
- Expand WBC and WinMET

Future Work and Open Directions

Open Directions

- TOCTOU remains unsolved
- Open source sandboxes suffer from usability issues
- Absence of standardized malware naming scheme
- Use of AI techniques with contextual information
- Noise in execution traces

Behavior Analysis for Vulnerability and Malware Detection

Ph.D. Candidate:

Razvan Raducu Teodorescu

Universidad de Zaragoza, Zaragoza, Spain

Advisors:

Prof. Dr. Pedro Álvarez

Prof. Dr. Ricardo J. Rodríguez

July 16, 2025