When Malware Changed Its Mind: An Empirical Study of Variable Program Behaviors in the Real World

Erin Avllazagaj[1], Ziyun Zhu[2], Leyla Bilge[3], Davide Balzarotti[4], Tudor Dumitraş[1]

[1]University of Maryland, College Park

[2]Facebook

[3]NortonLifeLock Research Group

[4]EURECOM

whoami

- **Erin Avllazagaj**
  - Phd Student @ University of Maryland, College Park
  - Data-driven analysis of malware in the wild

Agenda

- Drawbacks of dynamic analysis
- Measuring variability
- Finding invariants
  - Actionable implications
- Malware VS current dynamic analysis
  - Lessons learned

Malware Behavior Changes Across Environments

Malware Behavior Changes Across Environments

- Missing libraries, different language settings, etc.[1]

[1]

		Description of behavior
76.76 %	142	same behavior
9.19 %	17	evasion of Anubis
5.41 %	10	.NET environment required
3.24 %	6	evasion of our driver
3.24 %	6	different behavior due to other characteristics
2.16 %	4	not working in the German environment

[1] Lindorfer et al. "Detecting environment-sensitive malware." *RAID*, 2011.

Malware Behavior Changes Across Environments

- Missing libraries, different language settings, etc.[1]

- Prudent practices[2]:
  - "[…] caution generalizing from a single OS version […]"

[1]

		Description of behavior
76.76 %	142	same behavior
9.19 %	17	evasion of Anubis
5.41 %	10	.NET environment required
3.24 %	6	evasion of our driver
3.24 %	6	different behavior due to other characteristics
2.16 %	4	not working in the German environment

[1] Lindorfer et al. "Detecting environment-sensitive malware." *RAID*, 2011.
[2] Rossow et al. "Prudent practices for designing malware experiments: Status quo and outlook." IEEE S&P, 2012.

Malware Behavior Changes Across Environments

- Example: Ramnit Worm

```
int __cdecl try_to_exploit(LPSTR lpCommandLine)
{
  if ( !is_win8() && !is_win8_1() )
  {
    if ( is_xp() )
    {
      if ( !check_updates_xp((int)"KB3000061") )
      {
        if ( is_admin() )
          return 1;
LABEL_6:
        execute_CVE_2014_4113(lpCommandLine);
        return 1;
      }
    }
    else if ( !check_updates_other((int)"KB3000061") )
    {
      if ( is_admin() && check_authority() > 1 )
        return 1;
      goto LABEL_6;
    }
    try_second_exploit(lpCommandLine);
    return 1;
  }
  return 0;
}
```

https://cert.pl/en/posts/2017/09/ramnit-in-depth-analysis/

Malware Behavior Changes Across Environments

- Example: Ramnit Worm
  - Exploits CVE-2013-3660
    - Line 22
    - Local Privilege escalation on Win 7
    - Creates hundreds of **mutexes**
      - until exploit succeeds

```
1  int __cdecl try_to_exploit(LPSTR lpCommandLine)
2  {
3    if ( !is_win8() && !is_win8_1() )
4    {
5      if ( is_xp() )
6      {
7        if ( !check_updates_xp((int)"KB3000061") )
8        {
9          if ( is_admin() )
10           return 1;
11 LABEL_6:
12         execute_CVE_2014_4113(lpCommandLine);
13         return 1;
14       }
15     }
16     else if ( !check_updates_other((int)"KB3000061") )
17     {
18       if ( is_admin() && check_authority() > 1 )
19         return 1;
20       goto LABEL_6;
21     }
22     try_second_exploit(lpCommandLine);
23     return 1;
24   }
25   return 0;
26 }
```

https://cert.pl/en/posts/2017/09/ramnit-in-depth-analysis/

Malware Behavior Changes Across Environments

- Example: Ramnit Worm
  - Exploits CVE-2013-3660
    - Line 22
    - Local Privilege escalation on Win 7
    - Creates hundreds of **mutexes**
      - until exploit succeeds
  - Only works on:
    - vulnerable OS versions
    - when run in **non-admin**

```
1  int __cdecl try_to_exploit(LPSTR lpCommandLine)
2  {
3    if ( !is_win8() && !is_win8_1() )
4    {
5      if ( is_xp() )
6      {
7        if ( !check_updates_xp((int)"KB3000061") )
8        {
9          if ( is_admin() )
10           return 1;
11 LABEL_6:
12         execute_CVE_2014_4113(lpCommandLine);
13         return 1;
14       }
15     }
16     else if ( !check_updates_other((int)"KB3000061") )
17     {
18       if ( is_admin() && check_authority() > 1 )
19         return 1;
20       goto LABEL_6;
21     }
22     try_second_exploit(lpCommandLine);
23     return 1;
24   }
25   return 0;
26 }
```

https://cert.pl/en/posts/2017/09/ramnit-in-depth-analysis/

Research Questions

RQ1: Variability analysis in the wild

- What parts of the execution trace vary more? And by how much?

Research Questions

RQ1: Variability analysis in the wild

- What parts of the execution trace vary more? And by how much?

RQ2: Invariant analysis in the wild

- Can we find behavioral invariants across executions?

Research Questions

RQ1: Variability analysis in the wild

- What parts of the execution trace vary more? And by how much?

RQ2: Invariant analysis in the wild

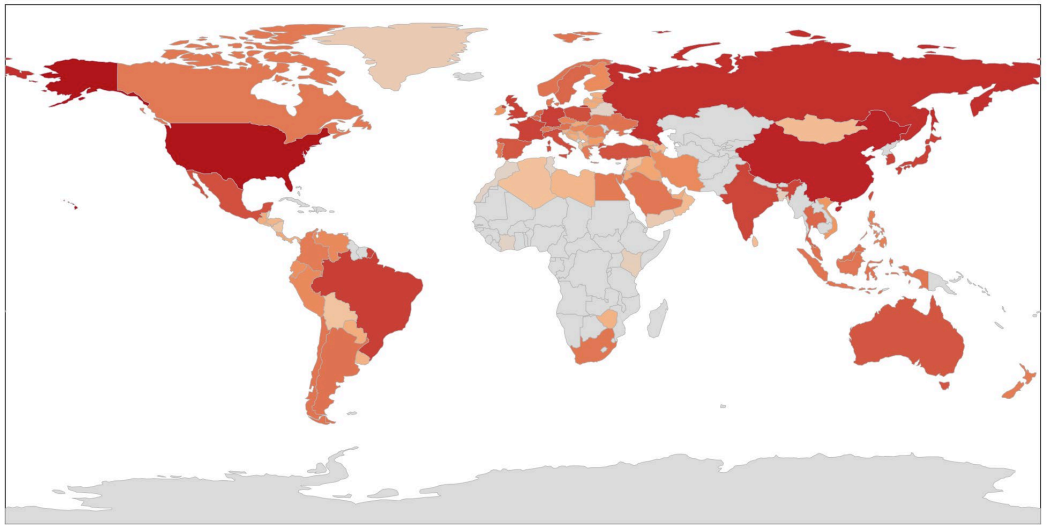
- Can we find behavioral invariants across executions?

RQ3: Impact of variability

- What is the impact of variability on malware detection and clustering?

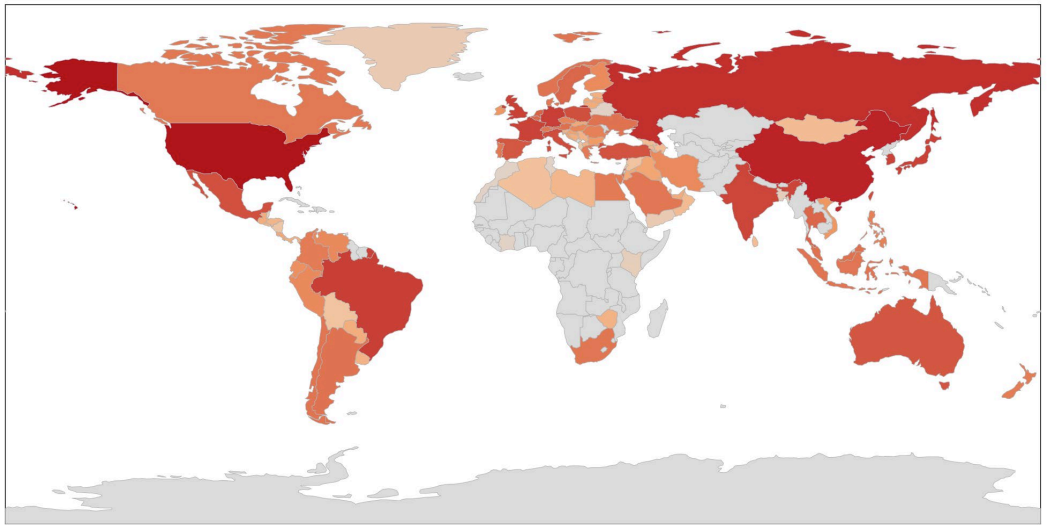
The Dataset

- **7.6M** execution traces
- **5.4M** real users' machines in **>100 countries** in the world
- From **2018**



The Dataset

- **7.6M** execution traces
- **5.4M** real users' machines in **>100 countries** in the world
- From **2018**
- No private data is collected, passive recording



The Dataset (introduction)

Action type	File name	File path	…	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	…	AAA	1	abc
Mtx. Create	mtx!asjkf	-	…	ABC	5243523	abd
…	…	…	…	…	…	…

The Dataset (introduction)

Action type	File name	File path	…	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	…	AAA	1	abc
Mtx. Create	mtx!asjkf	-	…	ABC	5243523	abd
…	…	…	…	…	…	…

(split by hash)

Action type	File name	File path	…	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	…	AAA	1	abc
…	…	…	…	AAA	…	…

Action type	File name	File path	…	sample Hash	Exec.id	Mach.id
Mtx. Create	mtx!asjkf	-	…	ABC	5243523	abd
…	…	…	…	ABC	…	…

The Dataset (introduction)

Action type	File name	File path	...	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	...	AAA	1	abc
Mtx. Create	mtx!asjkf	-	...	ABC	5243523	abd
...	...	...	...	...	...	...

(split by hash)

Action type	File name	File path	...	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	...	AAA	1	abc
...	...	...	...	AAA	...	...

Action type	File name	File path	...	sample Hash	Exec.id	Mach.id
Mtx. Create	mtx!asjkf	-	...	ABC	5243523	abd
...	...	...	...	ABC	...	...

Using VirusTotal labels and AVClass[1] (**2019**) we found:

22K benign, **2.4K** malware and **1.6K** PUP

MARYLAND CYBERSECURITY CENTER

[1] Sebastián et al. "Avclass: A tool for massive malware labeling." *RAID*, 2016.

The Dataset (introduction)

Action type	File name	File path	...	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	...	AAA	1	abc
Mtx. Create	mtx!asjkf	-	...	ABC	5243523	abd
...	...	...	...	...	...	...

(split by hash)

Action type	File name	File path	...	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	...	AAA	1	abc
...	...	...	...	AAA	...	...

Action type	File name	File path	...	sample Hash	Exec.id	Mach.id
Mtx. Create	mtx!asjkf	-	...	ABC	5243523	abd
...	...	...	...	ABC	...	...

Using VirusTotal labels and AVClass[1] (**2019**) we found:

22K benign, **2.4K** malware and **1.6K** PUP

[1] Sebastián et al. "Avclass: A tool for massive malware labeling." *RAID*, 2016.

RQ1: Variability Analysis In The Wild

- Ramnit worm exploit

```
1  int __cdecl try_to_exploit(LPSTR lpCommandLine)
2  {
3    if ( !is_win8() && !is_win8_1() )
4    {
5      if ( is_xp() )
6      {
7        if ( !check_updates_xp((int)"KB3000061") )
8        {
9          if ( is_admin() )
10           return 1;
11 LABEL_6:
12         execute_CVE_2014_4113(lpCommandLine);
13         return 1;
14       }
15     }
16     else if ( !check_updates_other((int)"KB3000061") )
17     {
18       if ( is_admin() && check_authority() > 1 )
19         return 1;
20       goto LABEL_6;
21     }
22     try_second_exploit(lpCommandLine);
23     return 1;
24   }
25   return 0;
26 }
```

https://cert.pl/en/posts/2017/09/ramnit-in-depth-analysis/

RQ1: Variability Analysis In The Wild

- Ramnit worm exploit

- When this line is reached
  - ~100 more mutex create events
  - based on the **machine**

```
1  int __cdecl try_to_exploit(LPSTR lpCommandLine)
2  {
3    if ( !is_win8() && !is_win8_1() )
4    {
5      if ( is_xp() )
6      {
7        if ( !check_updates_xp((int)"KB3000061") )
8        {
9          if ( is_admin() )
10           return 1;
11 LABEL_6:
12         execute_CVE_2014_4113(lpCommandLine);
13         return 1;
14       }
15     }
16     else if ( !check_updates_other((int)"KB3000061") )
17     {
18       if ( is_admin() && check_authority() > 1 )
19         return 1;
20       goto LABEL_6;
21     }
22     try_second_exploit(lpCommandLine);
23     return 1;
24   }
25   return 0;
26 }
```

https://cert.pl/en/posts/2017/09/ramnit-in-depth-analysis/

RQ1: Variability Analysis In The Wild (machines)

Methodology

(for each hash)

Action type	File name	File path	…	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	…	AAA	1	abc
…	…	…	…	AAA	…	…

RQ1: Variability Analysis In The Wild (machines)

Methodology (for each hash)

Action type	File name	File path	…	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	…	AAA	1	abc
…	…	…	…	AAA	…	…

(Group by machine ID and remove executions after week 0)

Action type	File name	File path	…	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	…	AAA	1	abc
…	…	…	…	AAA	2	abc

Action type	File name	File path	…	sample Hash	Exec.id	Mach.id
File Create	22222.exe	CSIDL_PROFILE	…	AAA	4	aaa
…	…	…	…	AAA	3	aaa

RQ1: Variability Analysis In The Wild (machines)

Methodology (for each hash)

Action type	File name	File path	...	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	...	AAA	1	abc
...	...	...	...	AAA	...	...

(Group by machine ID and remove executions after week 0)

Action type	File name	File path	...	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	...	AAA	1	abc
...	...	...	...	AAA	2	abc

File Creations: 5
Mutex Creations: 2
...
Total: 52

Action type	File name	File path	...	sample Hash	Exec.id	Mach.id
File Create	22222.exe	CSIDL_PROFILE	...	AAA	4	aaa
...	...	...	...	AAA	3	aaa

File Creations: 5
Mutex Creations: 42
...
Total: 92

RQ1: Variability Analysis In The Wild (machines)

Methodology (for each hash)

Action type	File name	File path	…	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	…	AAA	1	abc
…	…	…	…	AAA	…	…

(Group by machine ID and remove executions after week 0)

Action type	File name	File path	…	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	…	AAA	1	abc
…	…	…	…	AAA	2	abc

Action type	File name	File path	…	sample Hash	Exec.id	Mach.id
File Create	22222.exe	CSIDL_PROFILE	…	AAA	4	aaa
…	…	…	…	AAA	3	aaa

File Creations: 5
Mutex Creations: 2
…
Total: 52

File Creations: 5
Mutex Creations: 42
…
Total: 92

[45, … , **52** , … , **92** , … , 100] → IQR → 92 – 52 = **40**

RQ1: Variability Analysis In The Wild (machines)

Methodology (for each hash)

Action type	File name	File path	…	sample Hash	Exec.id	Mach. id
File Create	setup .exe	CSIDL_PROFILE	…	AAA	1	abc
…	…	…	…	AAA	…	…

(Group by machine ID and remove executions after week 0)

Action type	File name	File path	…	sample Hash	Exec.id	Mach. id
File Create	setup .exe	CSIDL_PROFILE	…	AAA	1	abc
…	…	…	…	AAA	2	abc

Action type	File name	File path	…	sample Hash	Exec.id	Mach. id
File Create	22222.exe	CSIDL_PROFILE	…	AAA	4	aaa
…	…	…	…	AAA	3	aaa

File Creations: 5
Mutex Creations: 2
…
Total: 52

File Creations: 5
Mutex Creations: 42
…
Total: 92

Analysis in the paper

[45, … , **52** , … , **92** , … , 100] → IQR → 92 – 52 = **40**

RQ1: Variability Analysis In The Wild (machines)

IQR → 92 – 52 = **40**

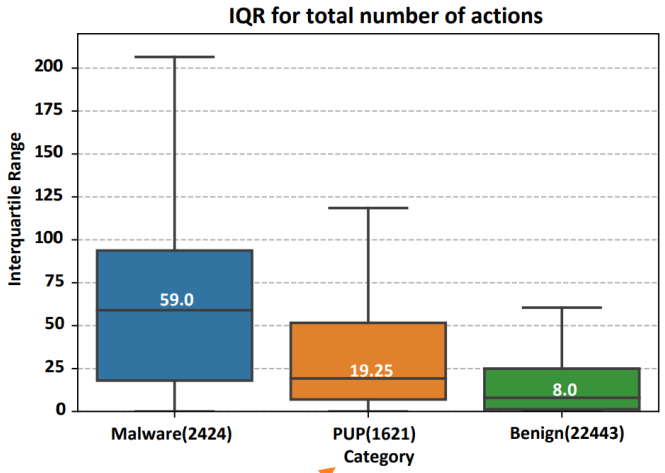
IQR → **10**

IQR → **0**

IQR → **100**

...

IQR → **60**



For all malware (blue boxplot)

For all PUP (orange boxplot)

...

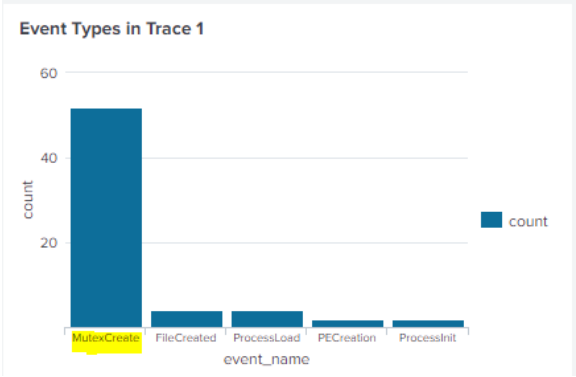
For all benign (green boxplot)

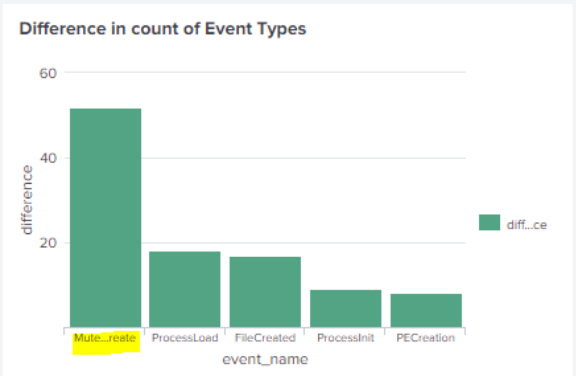
Back to Ramnit

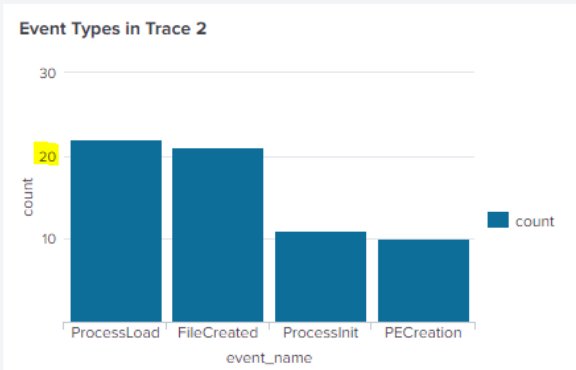
ne	server_date	local_time	platform_country_code	platform_processor	platform_system	enterprise_guid	machine_guid
31	Mon Oct 23 00:00:00 2018	Mon Oct 9 10:57:06 2018	55	x86 Family 6 Model 58 Stepping 9	Windows 7 build 7601 Service Pack 1	NULL	f6b4879ff52d11e6a55e94de80f65b84
20	Fri Nov 17 00:00:00 2018	Sun Nov 12 02:08:58 2018	86	Intel64 Family 6 Model 61 Stepping 4	Windows 10 Home China build 15063	NULL	f852fa3f06f64239b78ce5fd4ab77ec4

Trace 1

Trace 2





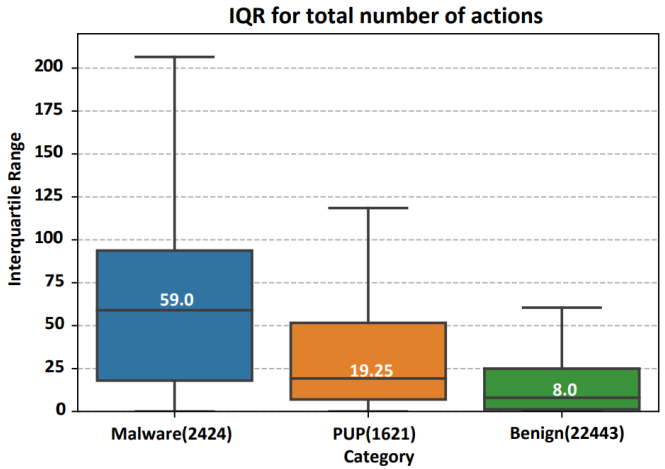


- Intuition
  - Trace 1 → vulnerable machine (and running in user mode)
  - Trace 2 → machine is not vulnerable (or running in Admin mode)

- IQR variability across machines will be large for Ramnit (sum of the green bars)

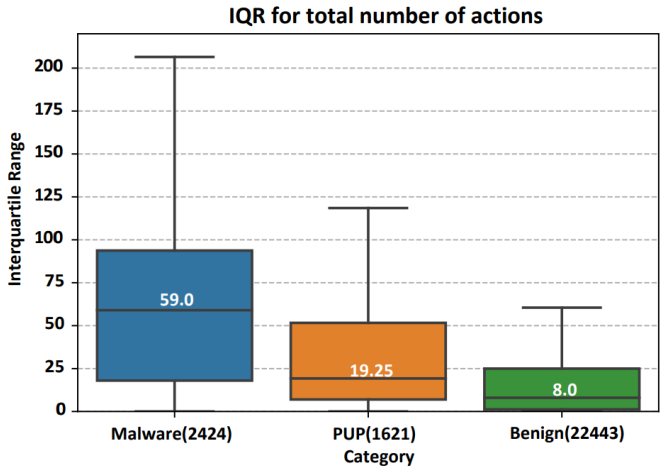
RQ1: Variability Analysis In The Wild (machines)

- At least 50% of the malware:
  - **59** missing or additional actions

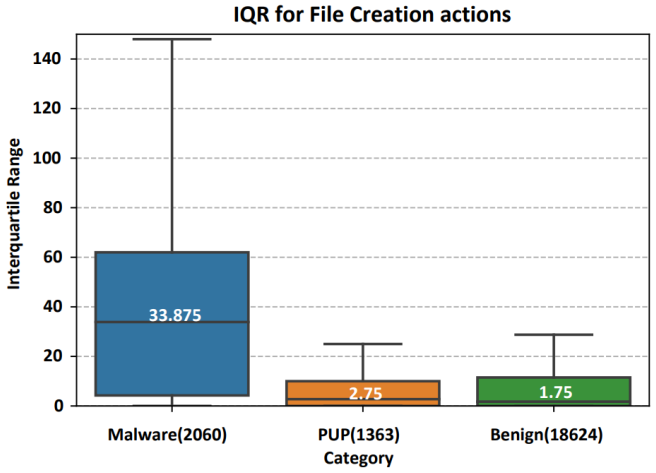


RQ1: Variability Analysis In The Wild (machines)

- At least 50% of the malware:
  - **59** missing or additional actions



- File creation
  - The major source of machine-induced variability in malware.



RQ1: Variability Analysis In The Wild (machines)

- Methodology:
  - IQR of the number of unique parameter values across different machines.

- Number of unique file names varies by **25** across machines

		Median			75^{th} percentile		
		Mal	**PUP**	**Ben**	**Mal**	**PUP**	**Ben**
File	Path	4	1	-	10	3	2
	Name	25	2	1	49	8	8
	Ext.	3	1	-	5	2	1

Details in the paper

RQ1: Variability Analysis In The Wild (machines)

- Measuring parameter value overlap

Details in the paper

RQ1: Variability Analysis In The Wild (machines)

- Measuring parameter value overlap
  - No shared value across machines for ~99% of the samples.

MARYLAND CYBERSECURITY CENTER

Details in the paper

RQ1: Case study (Gulpteba)

- Jaccard index:
  - 0 for file names
  - 0.2 for mutex names.

RQ1: Case study (Gulpteba)

- Jaccard index:
  - 0 for file names
  - 0.2 for mutex names.
- IQR:
  - Number of file creations: 0
  - Number of mutex creations: 2

RQ1: Case study (Gulpteba)

- Jaccard index:
  - 0 for file names
  - 0.2 for mutex names.
- IQR:
  - Number of file creations: 0
  - Number of mutex creations: 2

- Mutexes were a better candidate for building signatures
  - `h48yorbq6rm87zot` → appeared in all the machines
  - `ZonesCacheCounterMutex` and `ZoneAttributeCacheCounterMutex` only appeared in half of the machines, which explains the IQR of 2

https://www.trendmicro.com/vinfo/us/threat-encyclopedia/search/trojanspy.win32.glupteba.a.

RQ1: Case study (Gulpteba)

- Jaccard index:
  - 0 for file names
  - 0.2 for mutex names.
- IQR:
  - Number of file creations: 0
  - Number of mutex creations: 2

- Mutexes were a better candidate for building signatures
  - `h48yorbq6rm87zot` → appeared in all the machines
  - `ZonesCacheCounterMutex` and `ZoneAttributeCacheCounterMutex` only appeared in half of the machines, which explains the IQR of 2
  - Confirmed by a report from TrendMicro[1]

[1] https://www.trendmicro.com/vinfo/us/threat-encyclopedia/search/trojanspy.win32.glupteba.a.

RQ1 Summary

- High variability in malware across machines
  - File Creation makes up most of variability in malware
  - File name is the most variable parameter
  - For most malware there isn't a single parameter value shared across all machines (except file extension)

RQ2: Invariant Analysis In The Wild

- Can we find an invariant to detect these malware?

RQ2: Invariant Analysis In The Wild

- Focus on action-parameter pair signatures
  - used in Sigma
    - SIEM rules

```
logsource:
    category: process_creation
    product: windows
detection:
    selection:
        CommandLine: '*-noni -ep bypass $*'
    condition: selection
```

https://github.com/SigmaHQ/sigma

RQ2: Invariant Analysis In The Wild

- Focus on action-parameter pair signatures
  - used in cuckoo
    - Suspicious activity

```
class CreatesUserFolderEXE(Signature):
    name = "creates_user_folder_exe"
    description = "Creates an executable file in a user folder"
    severity = 3
    families = ["persistance"]
    authors = ["Kevin Ross"]
    minimum = "2.0"
    ttp = ["T1129"]

    directories_re = [
        "^[a-zA-Z]:\\\\Users\\\\[^\\\\]+\\\\AppData\\\\.*",
        "^[a-zA-Z]:\\\\Documents\\ and\\ Settings\\\\[^\\\\]+\\\\Local\\ Settings\\\\.*",
    ]

    def on_complete(self):
        for dropped in self.get_results("dropped", []):
            if "filepath" in dropped:
                droppedtype = dropped["type"]
                filepath = dropped["filepath"]
                if "MS-DOS executable" in droppedtype:
                    for directory in self.directories_re:
                        if re.match(directory, filepath):
                            self.mark_ioc("file", filepath)

        return self.has_marks()
```

https://github.com/cuckoosandbox/community/tree/master/modules/signatures

```
logsource:
    category: process_creation
    product: windows
detection:
    selection:
        CommandLine: '*-noni -ep bypass $*'
    condition: selection
```

https://github.com/SigmaHQ/sigma

RQ2: Invariant Analysis In The Wild

- Focus on action-parameter pair
  - used in Sigma
  - used in cuckoo

```
logsource:
    category: process_creation
    product: windows
detection:
    selection:
        CommandLine: '*-noni -ep bypass $*'
    condition: selection
```

https://github.com/SigmaHQ/sigma

```
class CreatesUserFolderEXE(Signature):
    name = "creates_user_folder_exe"
    description = "Creates an executable file in a user folder"
    severity = 3
    families = ["persistance"]
    authors = ["Kevin Ross"]
    minimum = "2.0"
    ttp = ["T1129"]

    directories_re = [
        "^[a-zA-Z]:\\\\Users\\\\[^\\\\]+\\\\AppData\\\\.*",
        "^[a-zA-Z]:\\\\Documents\\ and\\ Settings\\\\[^\\\\]+\\\\Local\\ Settings\\\\.*",
    ]

    def on_complete(self):
        for dropped in self.get_results("dropped", []):
            if "filepath" in dropped:
                droppedtype = dropped["type"]
                filepath = dropped["filepath"]
                if "MS-DOS executable" in droppedtype:
                    for directory in self.directories_re:
                        if re.match(directory, filepath):
                            self.mark_ioc("file", filepath)

        return self.has_marks()
```

https://github.com/cuckoosandbox/community/tree/master/modules/signatures

Prevalence of basic tokens

```yaml
title: Octopus Scanner Malware
id: 805c55d9-31e6-4846-9878-c34c75054fe9
status: experimental
description: Detects Octopus Scanner Malware.
references:
  - https://securitylab.github.com/research/octopus-scanner-malware-open-source-supply-chain
tags:
  - attack.t1195
  - attack.t1195.001
author: NVISO
date: 2020/06/09
logsource:
  product: windows
  category: file_event
detection:
  selection:
    TargetFilename|endswith:
      - '\AppData\Local\Microsoft\Cache134.dat'
      - '\AppData\Local\Microsoft\ExplorerSync.db'
  condition: selection
falsepositives:
  - Unknown
level: high
```

```yaml
logsource:
    category: registry_event
    product: windows
detection:
    selection:
        TargetObject:
            - 'HKCR\CLSID\{E08A0F4B-1F65-4D4D-9A09-BD4625B9C5A1}\Model'
        TargetObject|endswith:
              # covers HKU\* and HKLM..
            - '\SOFTWARE\App\AppXbf13d4ea2945444d8b13e2121cb6b663\Application'
            - '\SOFTWARE\App\AppXbf13d4ea2945444d8b13e2121cb6b663\DefaultIcon'
            - '\SOFTWARE\App\AppX70162486c7554f7f80f481985d67586d\Application'
            - '\SOFTWARE\App\AppX70162486c7554f7f80f481985d67586d\DefaultIcon'
            - '\SOFTWARE\App\AppX37cc7fdccd644b4f85f4b22d5a3f105a\Application'
            - '\SOFTWARE\App\AppX37cc7fdccd644b4f85f4b22d5a3f105a\DefaultIcon'
    selection2:
        TargetObject|startswith:
            - 'HKU\'
        TargetObject|contains:
            # HKCU\SOFTWARE\Classes\AppXc52346ec40fb4061ad96be0e6cb7d16a\
            - '_Classes\AppXc52346ec40fb4061ad96be0e6cb7d16a\'
            # HKCU\SOFTWARE\Classes\AppX3bbba44c6cae4d9695755183472171e2\
            - '_Classes\AppX3bbba44c6cae4d9695755183472171e2\'
            # HKCU\SOFTWARE\Classes\CLSID\{E3517E26-8E93-458D-A6DF-8030BC80528B}\
            - '_Classes\CLSID\{E3517E26-8E93-458D-A6DF-8030BC80528B}\'
            - '_Classes\CLSID\{E08A0F4B-1F65-4D4D-9A09-BD4625B9C5A1}\Model'
```

Prevalence of basic tokens

- Has at least 1 token if:
  - No regular expression characters and can be tokenized
  - If it has at least 3 tokens between 2 regular expression characters

```
title: Octopus Scanner Malware
id: 805c55d9-31e6-4846-9878-c34c75054fe9
status: experimental
description: Detects Octopus Scanner Malware.
references:
  - https://securitylab.github.com/research/octopus-scanner-malware-open-source-supply-chain
tags:
  - attack.t1195
  - attack.t1195.001
author: NVISO
date: 2020/06/09
logsource:
  product: windows
  category: file_event
detection:
  selection:
    TargetFilename|endswith:
      - '\AppData\Local\Microsoft\Cache134.dat'
      - '\AppData\Local\Microsoft\ExplorerSync.db'
  condition: selection
falsepositives:
  - Unknown
level: high
```

```
logsource:
    category: registry_event
    product: windows
detection:
    selection:
        TargetObject:
            - 'HKCR\CLSID\{E08A0F4B-1F65-4D4D-9A09-BD4625B9C5A1}\Model'
        TargetObject|endswith:
              # covers HKU\* and HKLM..
            - '\SOFTWARE\App\AppXbf13d4ea2945444d8b13e2121cb6b663\Application'
            - '\SOFTWARE\App\AppXbf13d4ea2945444d8b13e2121cb6b663\DefaultIcon'
            - '\SOFTWARE\App\AppX70162486c7554f7f80f481985d67586d\Application'
            - '\SOFTWARE\App\AppX70162486c7554f7f80f481985d67586d\DefaultIcon'
            - '\SOFTWARE\App\AppX37cc7fdccd644b4f85f4b22d5a3f105a\Application'
            - '\SOFTWARE\App\AppX37cc7fdccd644b4f85f4b22d5a3f105a\DefaultIcon'
    selection2:
        TargetObject|startswith:
            - 'HKU\'
        TargetObject|contains:
            # HKCU\SOFTWARE\Classes\AppXc52346ec40fb4061ad96be0e6cb7d16a\
            - '_Classes\AppXc52346ec40fb4061ad96be0e6cb7d16a\'
            # HKCU\SOFTWARE\Classes\AppX3bbba44c6cae4d9695755183472171e2\
            - '_Classes\AppX3bbba44c6cae4d9695755183472171e2\'
            # HKCU\SOFTWARE\Classes\CLSID\{E3517E26-8E93-458D-A6DF-8030BC80528B}\
            - '_Classes\CLSID\{E3517E26-8E93-458D-A6DF-8030BC80528B}\'
            - '_Classes\CLSID\{E08A0F4B-1F65-4D4D-9A09-BD4625B9C5A1}\Model'
```

Prevalence of basic tokens

- Has at least 1 token if:
  - No regular expression characters and can be tokenized
  - If it has at least 3 tokens between 2 regular expression characters
- **~70%** of all the Sigma open-source rules have at least 1 full token

```yaml
title: Octopus Scanner Malware
id: 805c55d9-31e6-4846-9878-c34c75054fe9
status: experimental
description: Detects Octopus Scanner Malware.
references:
  - https://securitylab.github.com/research/octopus-scanner-malware-open-source-supply-chain
tags:
  - attack.t1195
  - attack.t1195.001
author: NVISO
date: 2020/06/09
logsource:
  product: windows
  category: file_event
detection:
  selection:
    TargetFilename|endswith:
      - '\AppData\Local\Microsoft\Cache134.dat'
      - '\AppData\Local\Microsoft\ExplorerSync.db'
  condition: selection
falsepositives:
  - Unknown
level: high
```

```yaml
logsource:
    category: registry_event
    product: windows
detection:
    selection:
        TargetObject:
            - 'HKCR\CLSID\{E08A0F4B-1F65-4D4D-9A09-BD4625B9C5A1}\Model'
        TargetObject|endswith:
            # covers HKU\* and HKLM..
            - '\SOFTWARE\App\AppXbf13d4ea2945444d8b13e2121cb6b663\Application'
            - '\SOFTWARE\App\AppXbf13d4ea2945444d8b13e2121cb6b663\DefaultIcon'
            - '\SOFTWARE\App\AppX70162486c7554f7f80f481985d67586d\Application'
            - '\SOFTWARE\App\AppX70162486c7554f7f80f481985d67586d\DefaultIcon'
            - '\SOFTWARE\App\AppX37cc7fdccd644b4f85f4b22d5a3f105a\Application'
            - '\SOFTWARE\App\AppX37cc7fdccd644b4f85f4b22d5a3f105a\DefaultIcon'
    selection2:
        TargetObject|startswith:
            - 'HKU\'
        TargetObject|contains:
            # HKCU\SOFTWARE\Classes\AppXc52346ec40fb4061ad96be0e6cb7d16a\
            - '_Classes\AppXc52346ec40fb4061ad96be0e6cb7d16a\'
            # HKCU\SOFTWARE\Classes\AppX3bbba44c6cae4d9695755183472171e2\
            - '_Classes\AppX3bbba44c6cae4d9695755183472171e2\'
            # HKCU\SOFTWARE\Classes\CLSID\{E3517E26-8E93-458D-A6DF-8030BC80528B}\
            - '_Classes\CLSID\{E3517E26-8E93-458D-A6DF-8030BC80528B}\'
            - '_Classes\CLSID\{E08A0F4B-1F65-4D4D-9A09-BD4625B9C5A1}\Model'
```

RQ2: Invariant Analysis In The Wild

Action type	File name	File path	...	sample Hash	Exec.id	Mach.id
File Create	setup.exe	CSIDL_PROFILE	...	AAA	1	abc
...	...	...	...	AAA	...	...

Action type	File name	File path	...	sample Hash	Exec.id	Mach.id
Mtx. Create	mtx!asjkf	-	...	ABC	5243523	abd
...	...	...	...	ABC	...	...

Extract parameter values

CSIDL_PROFILE
icon.png.wnry
setup.exe
cmd.exe|del|virus.exe

mtx!asjkf
CSIDL_PROFILE/folder1
runprogram.exe
icon.png
CSIDL_APPDATA/bin
config.ini
setup.exe

Split them by delimiter

CSIDL_PROFILE
icon
png
wnry
setup
exe
cmd
del
virus

mtx!asjkf
CSIDL_PROFILE
folder1
runprogram
exe
icon
png
CSIDL_APPDATA
bin
config
ini
setup
exe

RQ2: Invariant Analysis In The Wild

CSIDL_PROFILE
icon
png
wnry
setup
exe
cmd
del
virus

mtx!asjkf
CSIDL_PROFILE
folder1
runprogram
exe
icon
png
CSIDL_APPDATA
bin
config
ini
setup
exe

(Remove values that appear in benign samples)

RQ2: Invariant Analysis In The Wild

CSIDL_PROFILE
icon
png
wnry
setup
exe
cmd
del
virus

mtx!asjkf
CSIDL_PROFILE
folder1
runprogram
exe
icon
png
CSIDL_APPDATA
bin
config
ini
setup
exe

(Remove values that appear in benign samples)

wnry
virus

RQ2: Invariant Analysis In The Wild

CSIDL_PROFILE
icon
png
wnry
setup
exe
cmd
del
virus

mtx!asjkf
CSIDL_PROFILE
folder1
runprogram
exe
icon
png
CSIDL_APPDATA
bin
config
ini
setup
exe

(Remove values that appear in benign samples)

wnry → appears in 30/50 machines → **60%**
virus → appears in 10/50 machines → **20%**

RQ2: Invariant Analysis In The Wild

CSIDL_PROFILE
icon
png
wnry
setup
exe
cmd
del
virus

mtx!asjkf
CSIDL_PROFILE
folder1
runprogram
exe
icon
png
CSIDL_APPDATA
bin
config
ini
setup
exe

(Remove values that appear in benign samples and only 1 machine)

wnry → appears in 30/50 machines → **60%**
virus → appears in 10/50 machines → **20%**

Combination → ***(wnry|virus|…)***

appear in **65%** of the machines

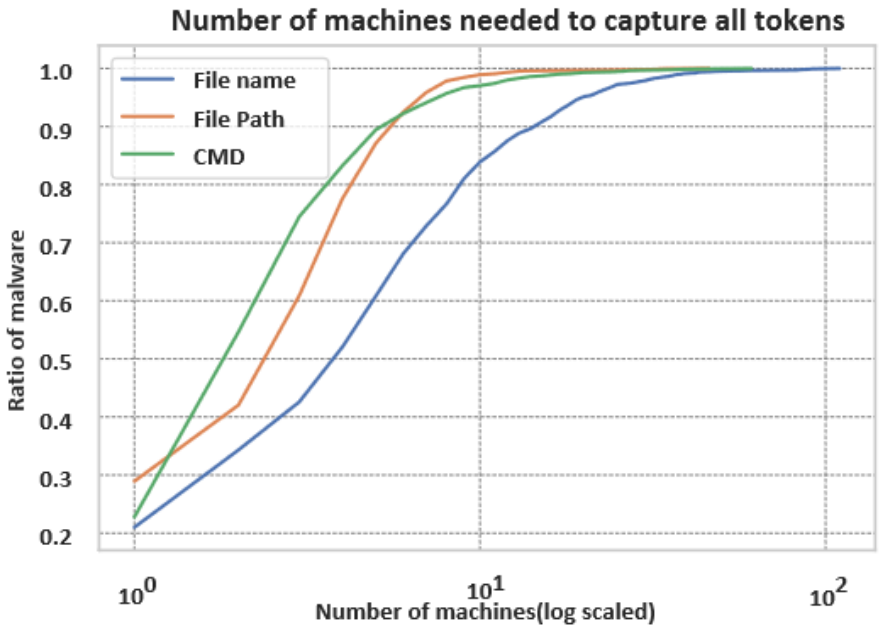
RQ2: How to capture all malware tokens?

RQ2: How to capture all malware tokens?

- Methodology:
  - We perform random sampling of **n** machines
  - Measure the amount of malware tokens
  - Compare with all the sample's malware tokens on all machines

RQ2: How to capture all malware tokens?

- Results:
  - File names are most difficult to completely capture.
  - CMD tokens and subdirectories captured with 10 machines.



RQ2: Invariant Analysis In The Wild

- How do these tokens detect?
- How to maximize coverage/detection?
  - **Assumption:** Sandbox is undetectable.

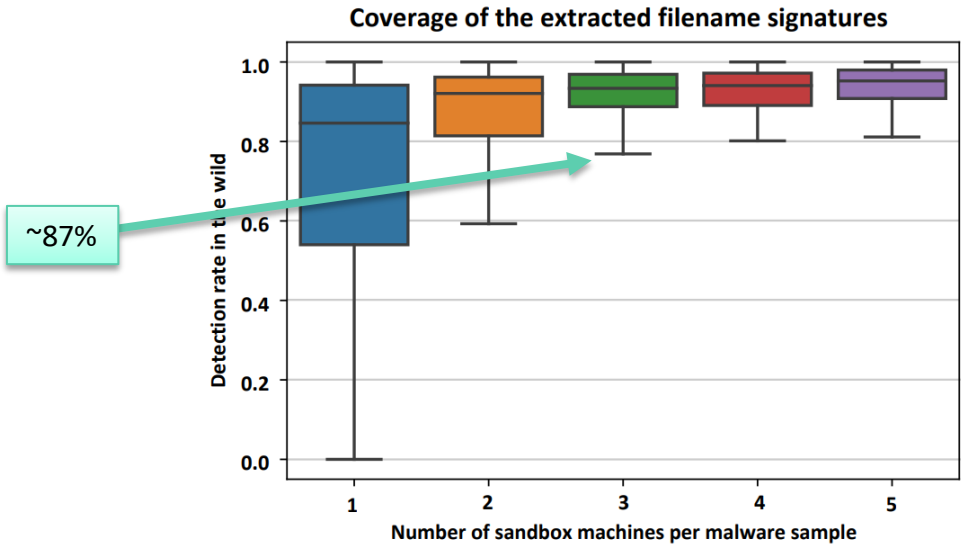
RQ2: Invariant Analysis In The Wild

- How do these tokens detect?
- How to maximize coverage/detection?
  - **Assumption:** Sandbox is undetectable, just like real machines.

- Pick **n** machines to get the bag of tokens
  - Check how much coverage would we get on the other machines.

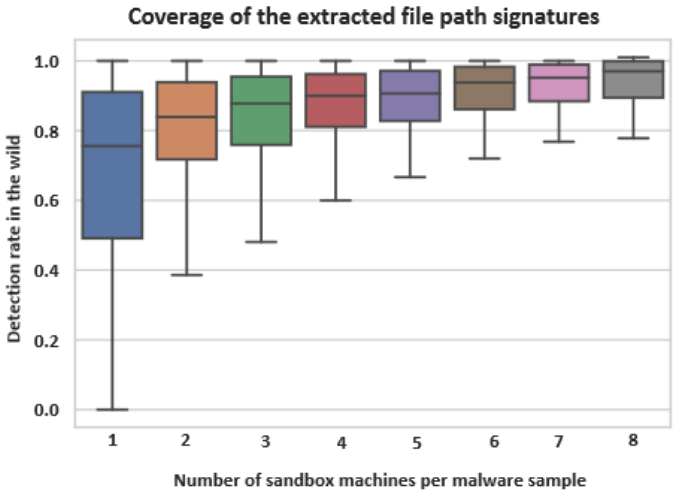
RQ2: Invariant Analysis In The Wild

- Maximum coverage in 3 randomly generated machines
  - For file name tokens
- One file name token doesn't appear in all machines.
  - Use more than 1 file name

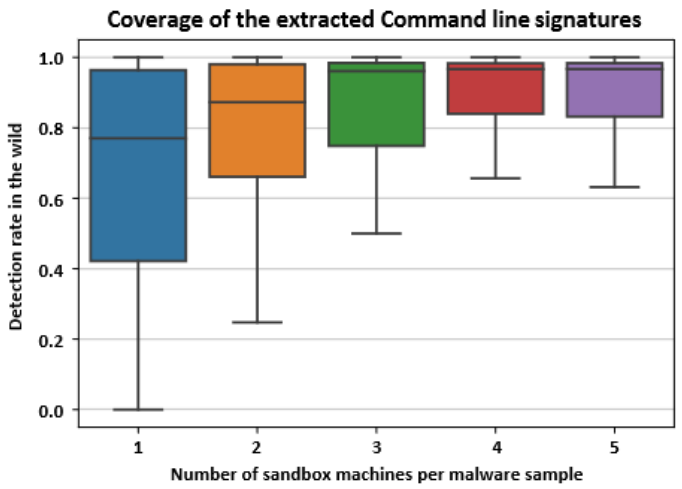


RQ2: Invariant Analysis In The Wild

- Analyst needs
  - 4 different sandboxes for CMD line
  - 7 for file path tokens
- Easier to obtain:
  - Poorer coverage than file name



(b)



(c)

How to create random sandboxes?

How to create random sandboxes?

- Use a random vm generator like SecGen [1]

[1] Schreuders, Z. Cliffe, et al. "Security Scenario Generator (SecGen): A Framework for Generating Randomly Vulnerable Rich-scenario VMs for Learning Computer Security and Hosting {CTF} Events." *ASE*. 2017.

How to create random sandboxes?

- Use a random vm generator like SecGen [1]
  - with the features proposed by Miramirkhani et al. [2]

TABLE I: Complete list of wear-and-tear artifacts.

Category	Name	Description	User	Sandbox	Baseline
System	totalProcesses	# of processes	94.4	35	41
	winupdt	# installed Windows updates	794	19	2
	sysevt	# system of system events	27K	8K	334
	appevt	# of application events	18K	2.4K	184
	syssrc	# sources of system events	78	49	48
	appsrc	# sources of application events	40	26	23
	sysdiffdays	Elapsed time since the first system event (days)	370	1.7K	0
	appdiffdays	Elapsed time since the first application event (days)	298	943	0
Disk	recycleBinSize	Total size of the recycle bin (bytes)	2.5G	50M	0
	recycleBinCount	# files in the recycle bin	109	3	0
	tempFilesSize	Total size of temporary system files (bytes)	302	24	8.2
	tempFilesCount	# temporary system files	411	60	10
	miniDumpSize	Total size of process crash minidump files (bytes)	3M	409K	0
	miniDumpCount	# of process crash minidump files	9	4	0
	thumbsFolderSize	Total size of the system's thumbnails folder (bytes)	63M	8M	2.6M
	desktopFileCount	# files on the desktop	34	6	3
Network	ARPCacheEntries	# entries in the ARP cache	19	4.5	5
	dnscacheEntries	# entries in the DNS resolver cache	151	4	3
	certUtilEntries	# URLs of previously downloaded CRLs	1.7K	210	6
	wirelessnetCount	# of cached wireless SSIDs	8	0	0
	tcpConnections	# of active TCP connections	77	27	16
	regSize	Size of the registry (in bytes)	144.8M	53M	35M

[1] Schreuders, Z. Cliffe, et al. "Security Scenario Generator (SecGen): A Framework for Generating Randomly Vulnerable Rich-scenario VMs for Learning Computer Security and Hosting {CTF} Events." *ASE*. 2017.
[2] Miramirkhani, Najmeh, et al. "Spotless sandboxes: Evading malware analysis systems using wear-and-tear artifacts." IEEE *S&P* 2017.

How to create random sandboxes?

- Use a random vm generator like SecGen [1]
  - with the features proposed by Miramirkhani et al. [2]
- AV vendors can collect such features

TABLE I: Complete list of wear-and-tear artifacts.

Category	Name	Description	User	Sandbox	Baseline
System	totalProcesses	# of processes	94.4	35	41
	winupdt	# installed Windows updates	794	19	2
	sysevt	# system of system events	27K	8K	334
	appevt	# of application events	18K	2.4K	184
	syssrc	# sources of system events	78	49	48
	appsrc	# sources of application events	40	26	23
	sysdiffdays	Elapsed time since the first system event (days)	370	1.7K	0
	appdiffdays	Elapsed time since the first application event (days)	298	943	0
Disk	recycleBinSize	Total size of the recycle bin (bytes)	2.5G	50M	0
	recycleBinCount	# files in the recycle bin	109	3	0
	tempFilesSize	Total size of temporary system files (bytes)	302	24	8.2
	tempFilesCount	# temporary system files	411	60	10
	miniDumpSize	Total size of process crash minidump files (bytes)	3M	409K	0
	miniDumpCount	# of process crash minidump files	9	4	0
	thumbsFolderSize	Total size of the system's thumbnails folder (bytes)	63M	8M	2.6M
	desktopFileCount	# files on the desktop	34	6	3
Network	ARPCacheEntries	# entries in the ARP cache	19	4.5	5
	dnscacheEntries	# entries in the DNS resolver cache	151	4	3
	certUtilEntries	# URLs of previously downloaded CRLs	1.7K	210	6
	wirelessnetCount	# of cached wireless SSIDs	8	0	0
	tcpConnections	# of active TCP connections	77	27	16
	regSize	Size of the registry (in bytes)	144.8M	53M	35M

[1] Schreuders, Z. Cliffe, et al. "Security Scenario Generator (SecGen): A Framework for Generating Randomly Vulnerable Rich-scenario VMs for Learning Computer Security and Hosting {CTF} Events." *ASE*. 2017.
[2] Miramirkhani, Najmeh, et al. "Spotless sandboxes: Evading malware analysis systems using wear-and-tear artifacts." IEEE *S&P* 2017.

RQ3: Impact of Variability

- In terms of:
  - Clustering
  - Anomaly detection (AccessMiner[1] , SIEM systems etc)

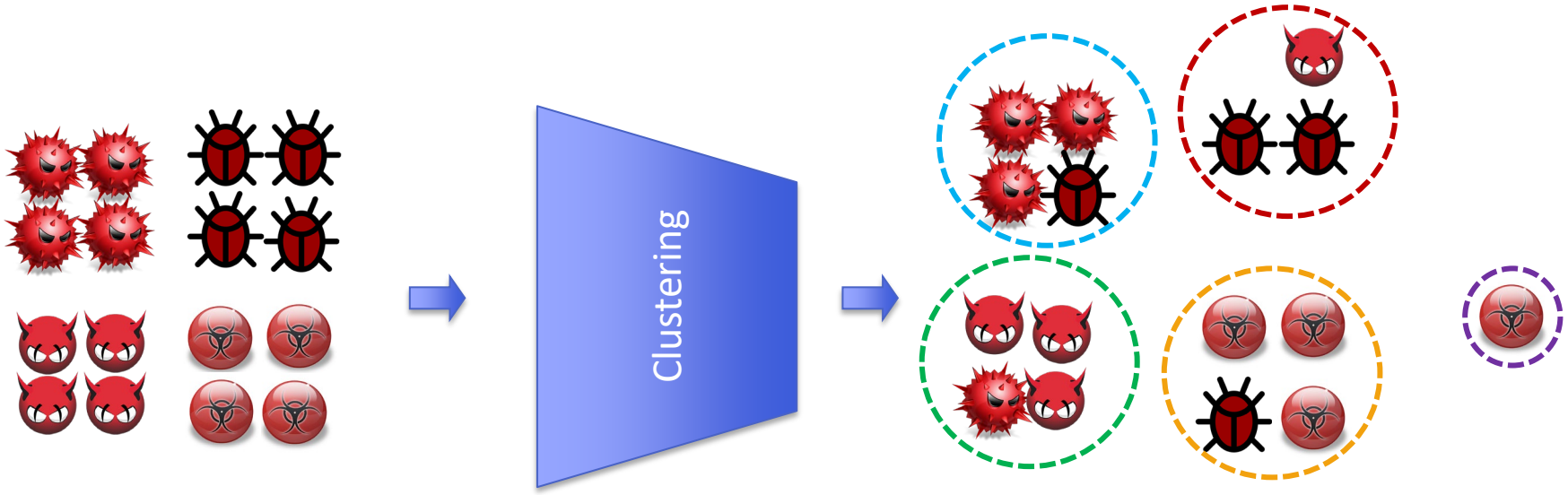
[1] Lanzi, et al. "Accessminer: using system-centric models for malware protection." *CCS* 2010.

RQ3: Impact of Variability (clustering)

- Questions:
  - How does malware behavior variability affect clustering?

RQ3: Impact Of Variability (clustering)

- Methodology:
  - Get 4 executions per malware sample in the same week
  - Reproduce the clustering by Bailey et al.[1]



[1] Bailey et al., Automated Classification and Analysis of Internet Malware, RAID 2007

RQ3: Impact of Variability (clustering)

- Results (out of **2424** malware samples):
  - 1,624 (67%) in the same cluster
  - 655 (27%) in 2 clusters
  - 121 (5%) in 3 clusters
  - 24 (1%) in 4 different cluster

RQ3: Impact of variability (clustering)

- Results (out of **2424** malware samples):
  - 1,624 (67%) in the same cluster
  - 655 (27%) in 2 clusters
  - 121 (5%) in 3 clusters
  - 24 (1%) in 4 different cluster

clustering results with 1 trace per sample
may not correctly cluster malware into families

RQ3: Impact of Variability (Anomaly detector)

- Questions:

RQ3: Impact of Variability (Anomaly detector)

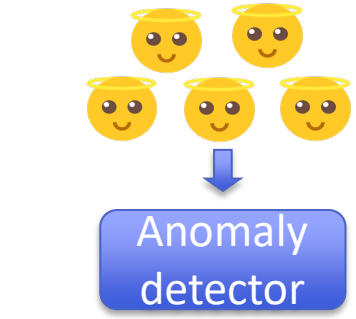
- Questions:
  - Is one execution per benign sample general enough?

RQ3: Impact of Variability (Anomaly detector)

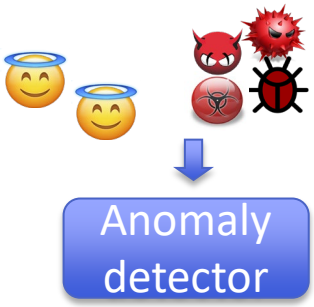
- Questions:
  - Is one execution per benign sample general enough?
  - What is the success rate for catching malware and PUP in the wild?

RQ3: Impact of Variability (Anomaly detector)

- Methodology:
  - Implement AccessMiner[1]
  - Select 90% of the benign samples and extract all file write directories
  - Test on all malware and remaining benign file write directories.



Extract features from some benign

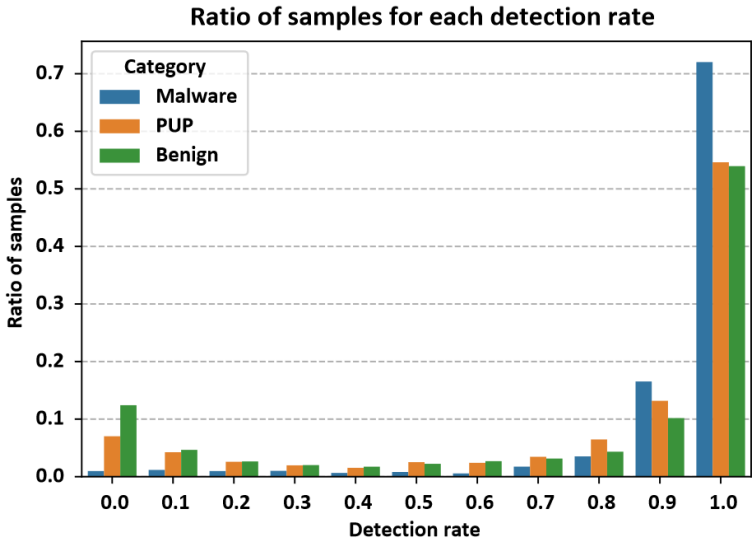


Classify other benign and malware

[1] Lanzi, et al. "Accessminer: using system-centric models for malware protection." *CCS* 2010.

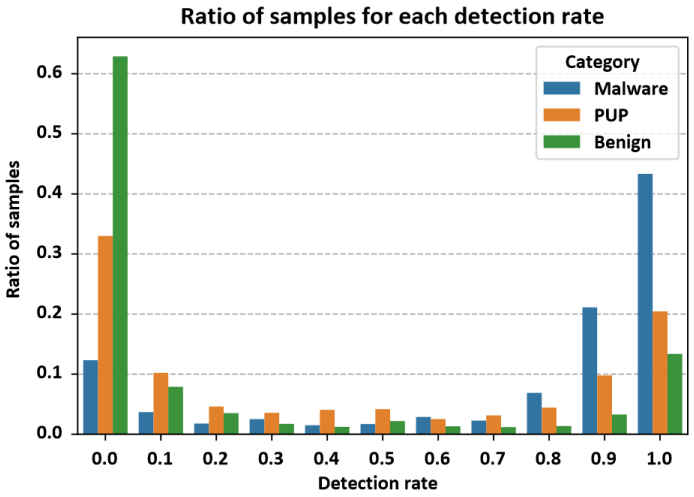
RQ3: Impact of Variability (Anomaly detector)

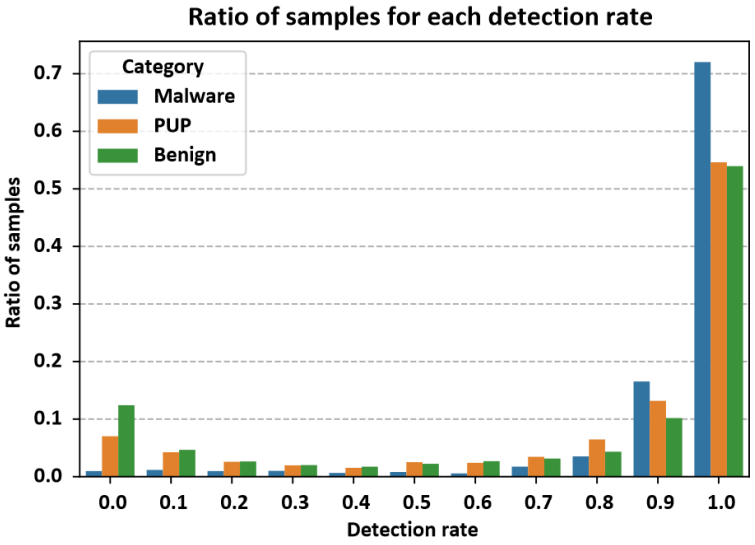
- Using 1 random benign execution for training
  - More malware can be detected in all machines.
  - High false-positive rate



RQ3: Impact of Variability (Anomaly detector)

- Using 1 random benign execution for training
  - More malware can be detected in all machines.
  - High false-positive rate
- Using all benign executions:
  - Significantly lower FP rate
  - Detection rate is low





RQ3: Summary

- When employing malware analysis methods:
  - Use multiple executions of the same malware samples
  - Use multiple executions of the benign samples

Conclusions

Conclusions

- First measurement of malware behavior at scale:
  - Single trace per malware sample is not enough

Conclusions

- First measurement of malware behavior at scale:
  - Single trace per malware sample is not enough
- Variability in malware is greater than PUP and benign
  - Across both time and machines

Conclusions

- First measurement of malware behavior at scale:
  - Single trace per malware sample is not enough
- Variability in malware is greater than PUP and benign
  - Across both time and machines
- It's still feasible to find invariant in malware behavior
  - AV vendors can safely do it

Thank you!

Erin Avllazagaj

albocode@umd.edu