"Botnet Battlefield": A Structured Study of Behavioral Interference Between Different Malware Families

Bishwa Hang Rai
Supervisor: Prof. Dr. Alexander Pretschner
Advisor: Mr. Tobias Wüchner

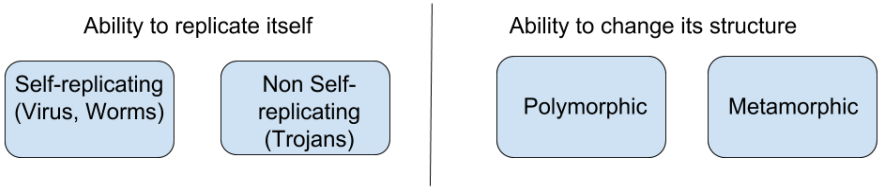
Department of Informatics
TU München

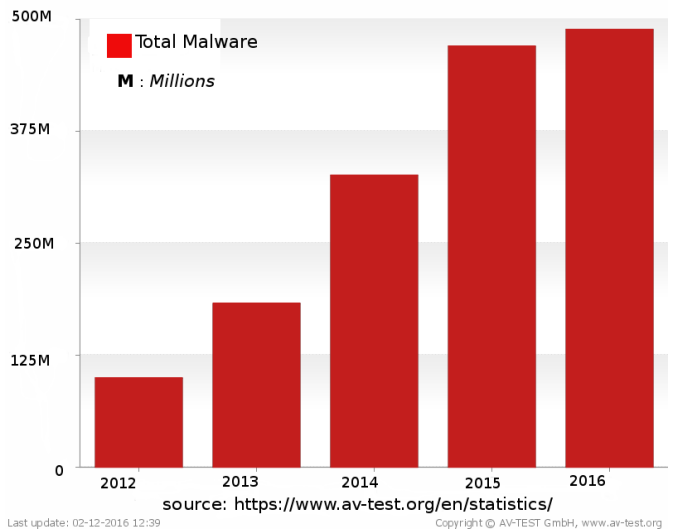
January 22, 2016

Table of contents

Malware

Malicious software that corrupts or steals data, or disrupt operations with illegitimate access to computer or computer networks





- Hard to detect with signature based
- Static and dynamic analysis (Anubis)

- Monetary profit
- In 2006, 2.8 billion dollars in US and 9.3 billion euros in Europe

Problem Statement

Interference Between Malware Families

- In 2004, NetSky vs Bagle and MyDoom
- In 2010, SpyEye vs Zbot
- In 2015, highly evasive Shifu malware

Problem Statement

Interference Between Malware Families

- In 2004, NetSky vs Bagle and MyDoom
- In 2010, SpyEye vs Zbot
- In 2015, highly evasive Shifu malware

Problem Statement

Identify the existence of behavioral interference between the malware families

Problem Statement

Interference Between Malware Families

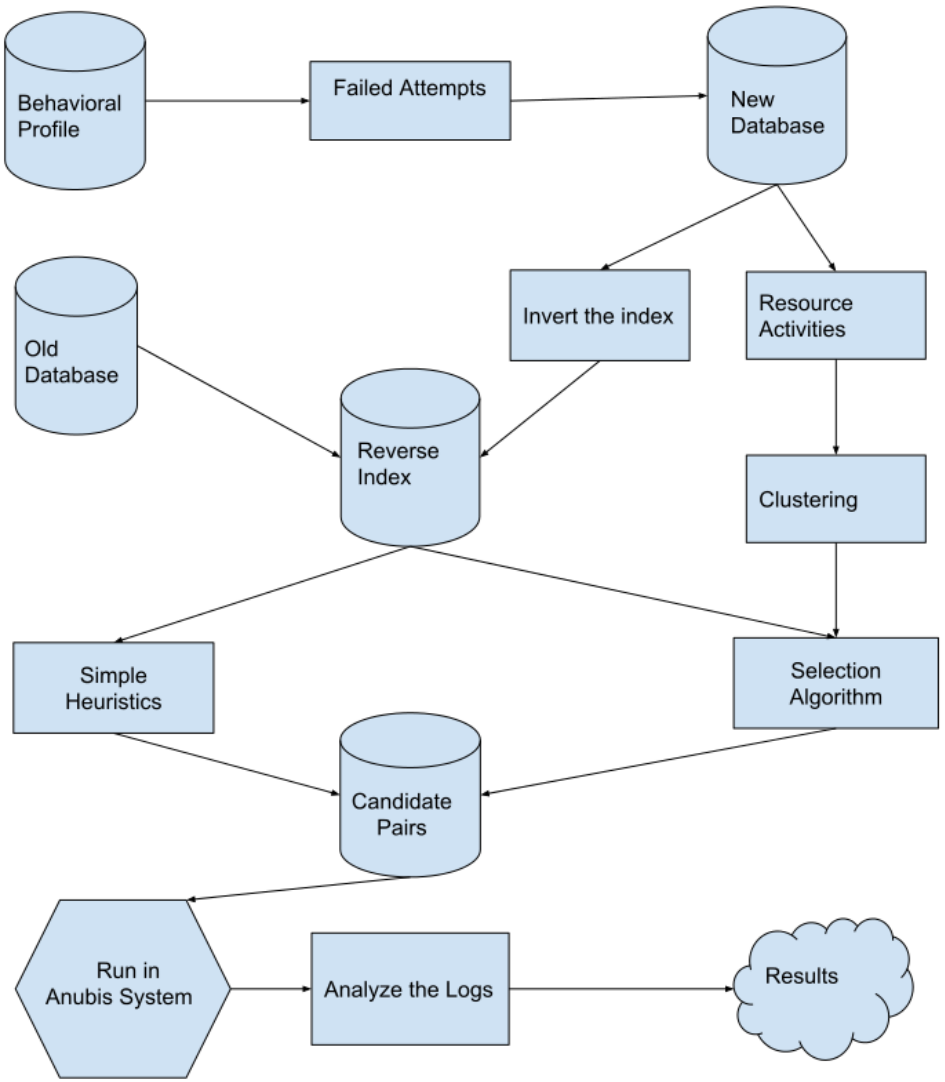
- In 2004, NetSky vs Bagle and MyDoom
- In 2010, SpyEye vs Zbot
- In 2015, highly evasive Shifu malware

Problem Statement

Identify the existence of behavioral interference between the malware families

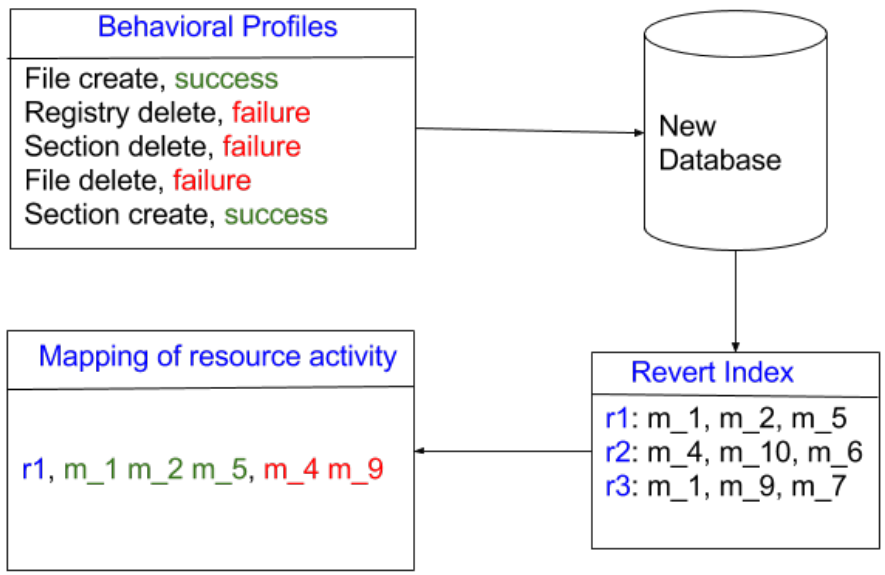
- Dynamic aspect/environment-sensitive malware
- The inter-family relations and their associated underground economy

The Big Picture



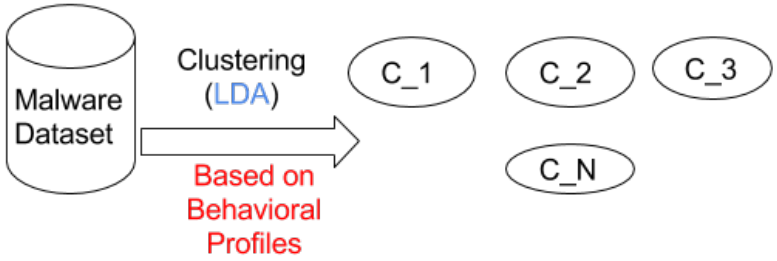
Initial Phase

- Datasets: Millions of malware samples from Anubis
- Behavioral Profile: execution trace
- Resource types: File, Registry, Sync, Section, Process
- Resource activities: Create, Delete, Access
- Modeled the problem into Map-reduce
- N-combination problem



Clustering

- Document Clustering
- **tf-idf**
  $O(\#words \times \#documents)$
- **LDA**
  $O(\#words \times \#topics)$
- Filter extremes:
  no_below 10 and
  no_above 1 Millions
- Large intra-distance and small inter-distance



RESOURCE_CODE = {"file" : "1", "registry" : "2" ...}
OPERATION_CODE = {"access" : "1", " delete" : "2", "modify" : "3"}

A file delete action of filename *"foo.txt"* with file_name_id '789' in our database would be represented as: '1_2_789'

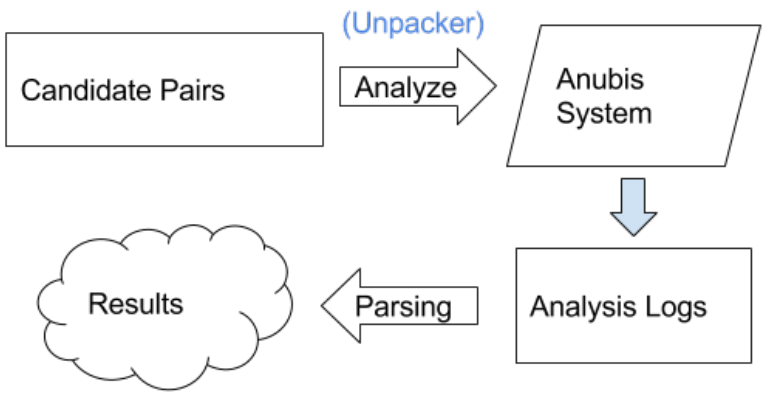
Candidate Selection

1: R = Set of all interesting resource
2: A_r = Set of malware that creates a particular resource 'r'
3: B_r = Set of malware that delete/access (failed) particular resource 'r'
4: N = Maximum number of families to consider
5: E = Set of all probable candidate
6: **function** C (j)
7: c_j = cluster id that malware j belongs to
8: *Return* c_j
9: **end function**
10: **for all** $r \in R$ **do**
11: **if** $|C(x_r) : x \in A_r| > N \vee |C(y_r) : y \in B_r| > N$ **then**
12: **continue**
13: **end if**
14: **for all** $(x_r, y_r) \in A_r \times B_r$ **do**
15: **if** $C(x_r) \neq C(y_r)$ **then**
16: $E \leftarrow (x_r, y_r)$
17: **end if**
18: **end for**
19: **end for**

Candidate Pair Analysis

- Candidate pairs analyzed in Anubis
- Extraneous data appended to binary
- Packer packs candidate pair to Unpacker
- Unpacker self reads and drops
- Logs parsed

Unpacker



Contribution

Our research will provide the following contributions:

- Systematic study of interferences between malware families
- A novel approach to malware clustering based on malware behavior profiles
- An automated system that detects interfering malware samples on a large scale

Results

List of Candidate Pairs

Resource types	#candidate pairs
File	213,171
Registry	39,899
Sync	7,781
Section	2,786
Process	54
Total	263,691

- Value of N (maximum family cutoff) in algorithm chosen to be 10
- '*File*' with the highest number of candidate pair and '*Process*' with the lowest

Results

List of Candidate Pairs

Resource types	#candidate pairs
File	213,171
Registry	39,899
Sync	7,781
Section	2,786
Process	54
Total	263,691

- Value of N (maximum family cutoff) in algorithm chosen to be 10
- '*File*' with the highest number of candidate pair and '*Process*' with the lowest

Result of Candidate Run

Resource types	# tested pairs	# true positive	prediction accuracy
File	5,000	1032	20.64%
Registry	5,000	731	14.62%
Sync	1,000	119	11.9%
Section	1,000	93	9.3%
Process	54	6	11.11%

- Average prediction accuracy: 14.25%

Threats to Validity

- Different values of N would give different candidate pairs and different results
- Didn't deal with random resource name
- Total execution time 10 minutes
- Sequence of execution
- True semantics of malware

Conclusion and Future Work

Conclusion

- Behavioral interference between malware families exists
- Malware checks for the presence of resource created by other malware and deletes it
- Our system could detect such interfering malware with average accuracy rate of 14.25%
- In our dataset, Files and Registries were the most interfered resource and Process was the least

Future Work

- Make the experiment more efficient
- In depth analysis (static) of positive pair

Questions

QUESTIONS?

Reverse Index

Listing 1 : Sort and join the reverse index

```
LANG=en_EN sort -t, -k 1,1 $file_name
LANG=en_EN join -t , -a1 -a2 $fn1 $fn2
```

Listing 2 : Sample of reverse index created for File activity

```
C:\mbr.exe,189524063,184501719,87504631,86763863
C:/DOCUME~1/ADMINI~1/LOCALS~1/Temp/telnet.exe
    ,178046895,174206059,183601891,89650247
C:/DOCUME~1/ADMINI~1/LOCALS~1/Temp/1.jpg
    ,161552035,116241803
```

Inter and Intra Distance

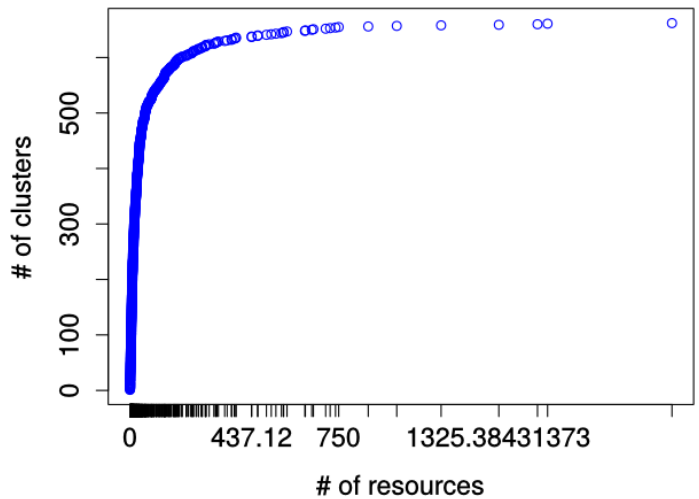


Figure 1 : Graph showing cdf distribution of common resource between same family topic

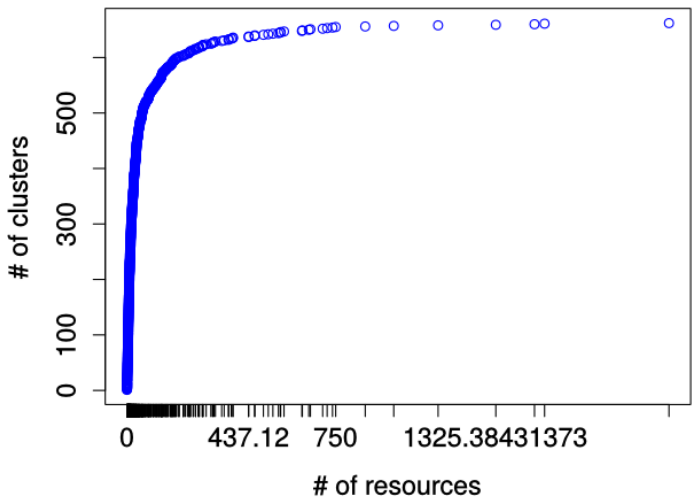


Figure 2 : Graph showing cdf distribution of common resource between same family topic

Max Flow

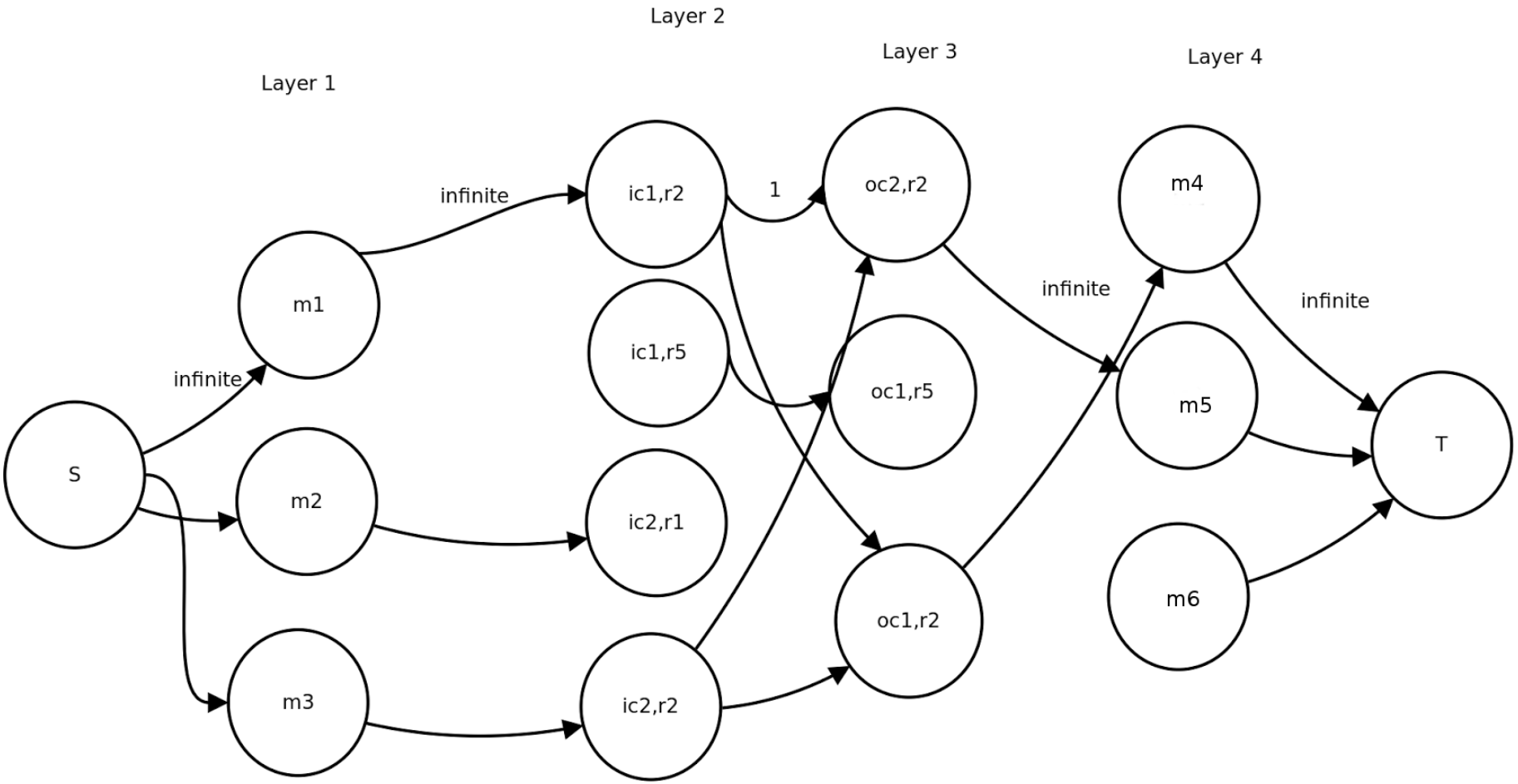


Figure 3 : Graph representing the max flow implementation

Heuristics

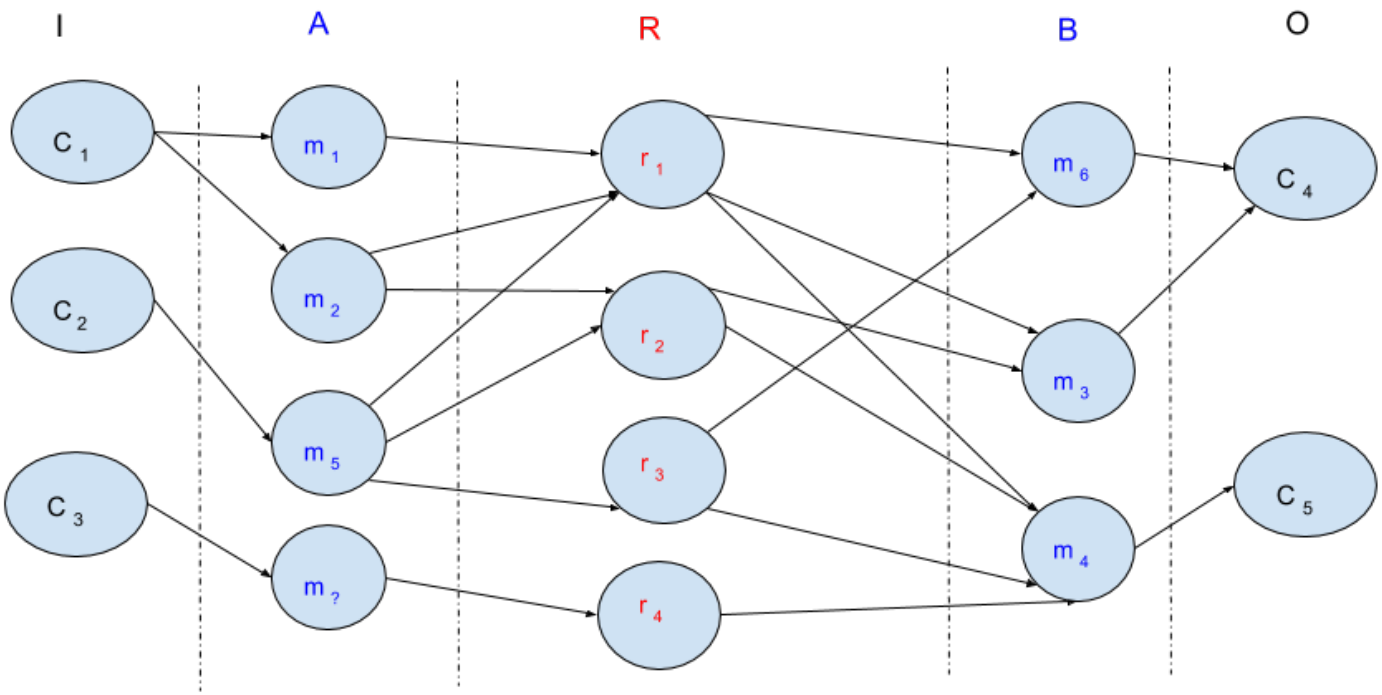


Figure 4 : Heuristics approach to optimal malware pair selection

Experiment Setup

- 7 Anubis instance
- Each instance emulates entire running PC with Windows XP Service Pack 3 as OS
- Uses Qemu and monitors process by invoking callback routine for every basic block executed in virtual processor
- Unpacker and Packer used to run the candidate pair
- 10 minutes as total run time of each candidate pair experiment
- 4 minute for each malware, and 2 minute to boot system

Some Examples

- Artemis! vs Cosmu on resource `C:\Old.exe`
- VB.CB vs Startpage.AI on resource `C:\WINDOWS\window.exe`
- KeyLogger vs OnlineGames on resource `C:\windows\system32\svrchost.exe`