NTT Security Holdings

The Rule for Wild Mal-Gopher Families.

NTT Security Japan

Kazuya Nomura

Sachito Hirao

Agenda

1. Introduction
2. Creating YARA Module
3. Clustering Evaluation
4. Applying to "Wild" Binaries
5. Conclusions

Appendix

1. Introduction

The Rule for Wild Mal-Gopher Families.

About Us

Kazuya Nomura

SOC analyst at NTT Security Japan. His main work is alert monitoring with IPS/IDS/EDR. He has contributed articles on malware analysis and data visualization in NTT Security Japan. He is a recipient of the MWS2020 paper award and an outstanding alumnus of SecHack2020.

Sachito Hirao

SOC analyst at NTT Security Japan. Formerly an infrastructure engineer in the financial sector.
At SOC, he was in charge of NW/EDR alert monitoring as well as malware analysis.

Golang Malware

- **Golang malware family grows year after year**
  - Complexity of analysis due to characteristic structure
  - Buildable for multiple platforms

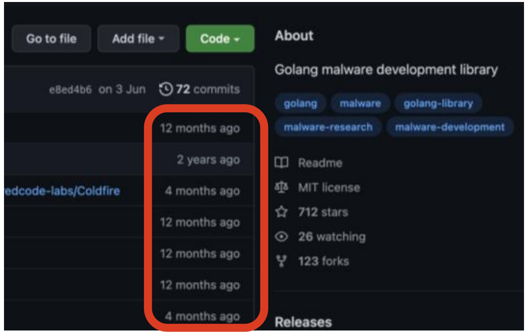
→ Advantages for the attacker

- The number of diverse samples will continue to increase
  - Increased efficiency of classification
  - Increased efficiency of analysis by attributing to previously analyzed samples

Improved efficiency of Golang malware classification and analysis

Golang Malware

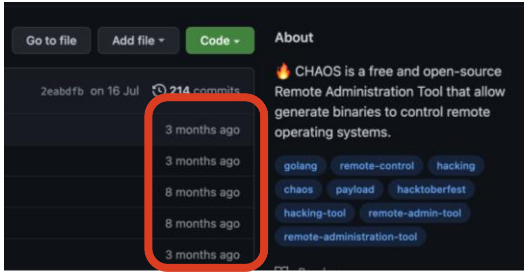
- Various malware creation frameworks exist for Golang
  - Coldfire
  - CHAOS
  - EGESPLOIT
  - ARCANUS
- Many frameworks in active development



About

Golang malware development library

https://github.com/redcode-labs/Coldfire



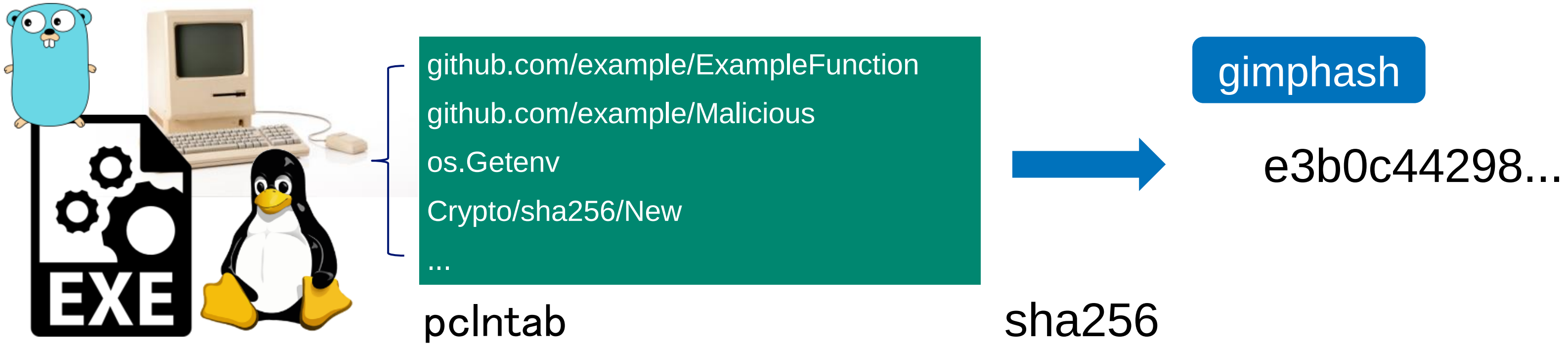
About

🔥 CHAOS is a free and open-source Remote Administration Tool that allow generate binaries to control remote operating systems.

https://github.com/tiagorlampert/CHAOS

gimphash

- **Golang binary version’s imphash**
  - Golang binaries have a platform-independent structure called pclntab
  - Dependent package names, function names, etc. can be restored
  - gimphash is a partial SHA256 hash of the recovered package/function name
  - Uniquely capable of representing the functionality on which malware depends, but **similarity comparisons of different hashes are not possible**

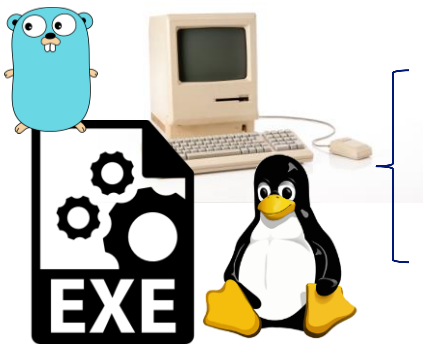


github.com/example/ExampleFunction
github.com/example/Malicious
os.Getenv
Crypto/sha256/New
...

gimpfuzzy

- **gimphash to fuzzy hash**
  - SHA256 output varies greatly if input differs by even 1 bit
  - Fuzzy Hash computes a ″rough″ hash that returns similar values for similar inputs
  - gimpfuzzy uses ssdeep. Similarity between samples can be measured.

pclntab



github.com/example/ExampleFunction
github.com/example/Malicious
os.Getenv
Crypto/sha256/New
...

github.com/example/AnotherFunction
github.com/example/Malicious
os.Getenv
Crypto/sha256/New
...

gimpFuzzy

3:j4dwGIVWYvgxN4dwGIV3MNWzYKKKvTKrs:j4mGggximGCMcTjKrs

Allows comparison of similarities

3:j4dwGIV++RSXzM9xN4dwGIV3MNWzYKKKvTKrs:j4mGduSXg9ximGCMcTjKrs

Motivation & Goals

1. **YARA module implementation**
    - Enables fast and easy classification of large sample groups

2. **Accuracy evaluation using analyzed samples**
    - Consider optimal parameters for family classification

3. **Applied to samples submitted to VirusTotal**
    - Application to unanalyzed “wild” samples
    - Discussion of the latest Golang malware applications

Difference Between Previous Presentation

- **CODE BLUE 2022**
  “Who is the Mal-Gopher? – Implementation and Evaluation of “gimpfuzzy” for Go Malware Classification”
  - First to propose a method for applying Fuzzy Hash to gimphash
  - Analyzed samples are classified by gimphash and evaluated the accuracy.

- **JSAC2023 “The Rule for Wild Mal-Gopher Families.”**
  Implementation and evaluation with a focus on application in actual operations and analysis
  - Creation of a YARA module that allows classification of samples for implementation.
  - Application and evaluation of “wild” unanalyzed samples submitted to VT

Creating YARA Module

The Rule for Wild Mal-Gopher Families.

YARA

- Toolkit for malware classification being developed by VT[1]
    - By writing classification rules, only samples that satisfy the rules can be searched
    - High speed because it is implemented with C
- Various modules exist depending on the file structure
- **The following modules do not exist**
    - Module for handling Golang binaries
    - Module for Fuzzy Hashing and string similarity calculation
- To make it easier to classify samples by gimpfuzzy, **YARA module was newly impremented**

[1] https://github.com/VirusTotal/yara

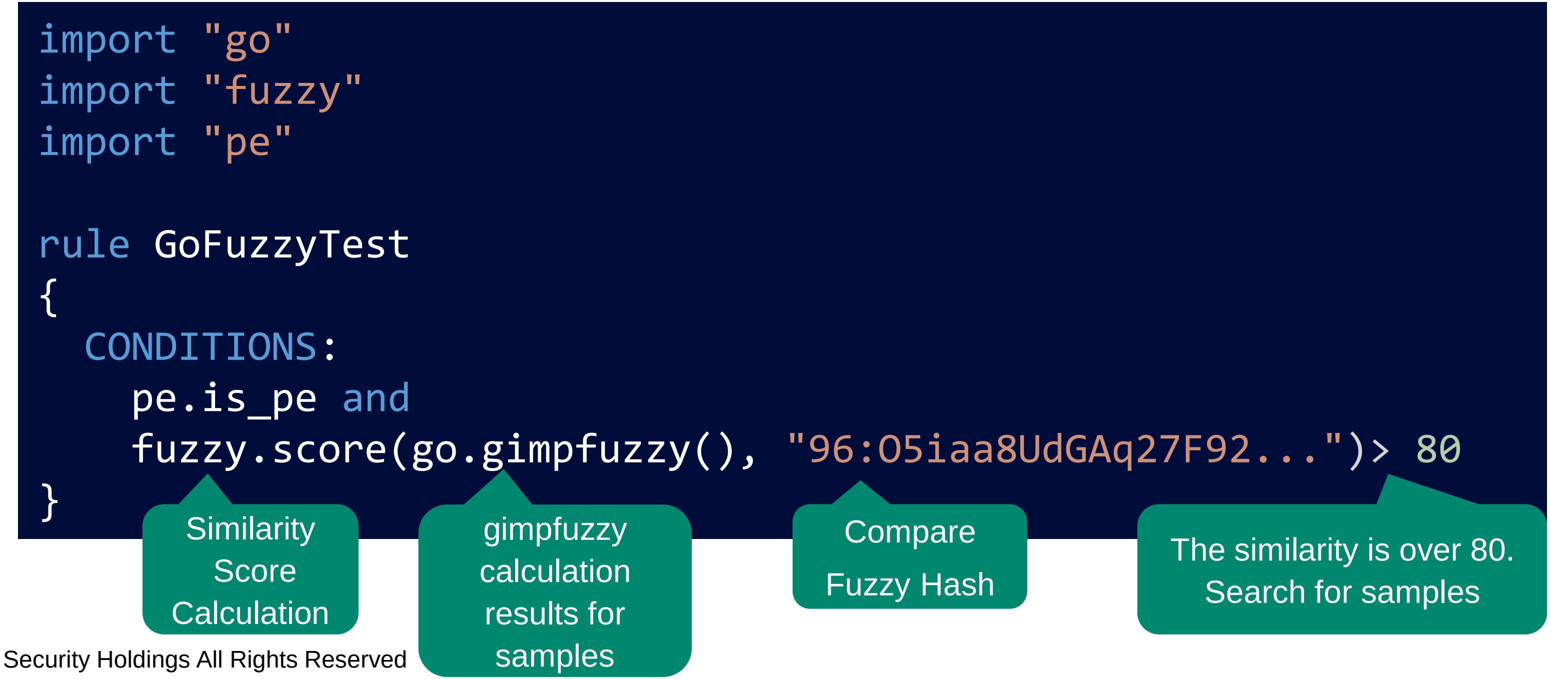
Creating YARA Module

Implement the following two

- **go module : analyzes PE binaries made by golang**
    - go.gimpfuzzy() : gimpfuzzy calculate from extracted function name
    - go.function_names : sort strings of extracted function names
- **fuzzy module : calculate similarity of Fuzzy Hash**
    - fuzzy.fuzzy() : fuzzy hash calculation from argument string
    - fuzzy.score() :computes score based on the edit distance between two argument strings

Writing YARA Rule

- Example of YARA rules
  - Enables searches based on similarity of samples based on gimpfuzzy



```
import "go"
import "fuzzy"
import "pe"

rule GoFuzzyTest
{
  CONDITIONS:
    pe.is_pe and
    fuzzy.score(go.gimpfuzzy(), "96:O5iaa8UdGAq27F92...")> 80
}
```

Similarity Score Calculation

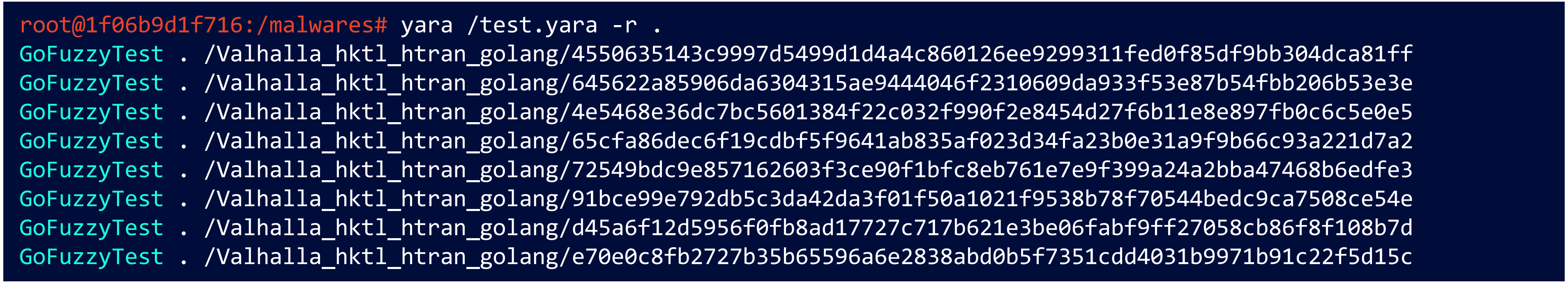
gimpfuzzy calculation results for samples

Compare Fuzzy Hash

The similarity is over 80. Search for samples

Using YARA Module

- Enables sample search based on gimpfuzzy similarity



```
root@1f06b9d1f716:/malwares# yara /test.yara -r .
GoFuzzyTest . /Valhalla_hktl_htran_golang/4550635143c9997d5499d1d4a4c860126ee9299311fed0f85df9bb304dca81ff
GoFuzzyTest . /Valhalla_hktl_htran_golang/645622a85906da6304315ae9444046f2310609da933f53e87b54fbb206b53e3e
GoFuzzyTest . /Valhalla_hktl_htran_golang/4e5468e36dc7bc5601384f22c032f990f2e8454d27f6b11e8e897fb0c6c5e0e5
GoFuzzyTest . /Valhalla_hktl_htran_golang/65cfa86dec6f19cdbf5f9641ab835af023d34fa23b0e31a9f9b66c93a221d7a2
GoFuzzyTest . /Valhalla_hktl_htran_golang/72549bdc9e857162603f3ce90f1bfc8eb761e7e9f399a24a2bba47468b6edfe3
GoFuzzyTest . /Valhalla_hktl_htran_golang/91bce99e792db5c3da42da3f01f50a1021f9538b78f70544bedc9ca7508ce54e
GoFuzzyTest . /Valhalla_hktl_htran_golang/d45a6f12d5956f0fb8ad17727c717b621e3be06fabf9ff27058cb86f8f108b7d
GoFuzzyTest . /Valhalla_hktl_htran_golang/e70e0c8fb2727b35b65596a6e2838abd0b5f7351cdd4031b9971b91c22f5d15c
```

Developing YARA Module

- Implement the following functions

Main implementation

function (e.g. math, programming, programing)	
module_initialize	Initialization process for YARA module
module_finalize	YARA module termination process
module_load	Processing when the module reads a file Implement the actual parsing logic for the file
module_unload	Post-processing when the module reads a file Delete hash tables, open structures, etc.

Developing YARA Module

- It is important that each function returns a corresponding error when an exception occurs.

```
#ifndef ERROR_SUCCESS
#define ERROR_SUCCESS 0
#endif

#define ERROR_INSUFFICIENT_MEMORY 1
#define ERROR_COULD_NOT_ATTACH_TO_PROCESS 2
#define ERROR_COULD_NOT_OPEN_FILE 3
#define ERROR_COULD_NOT_MAP_FILE 4
#define ERROR_INVALID_FILE 6
#define ERROR_CORRUPT_FILE 7
```

success processing

error handling

Demonstration : YARA Module

Demonstration

Clustering Evaluation

The Rule for Wild Mal-Gopher Families.

Clustering Evaluation

- Clustering evaluation using actual observed “wild” samples

Evaluation using analyzed samples

- Classify **samples identified as malware only**
- Evaluate the validity and accuracy of clustering

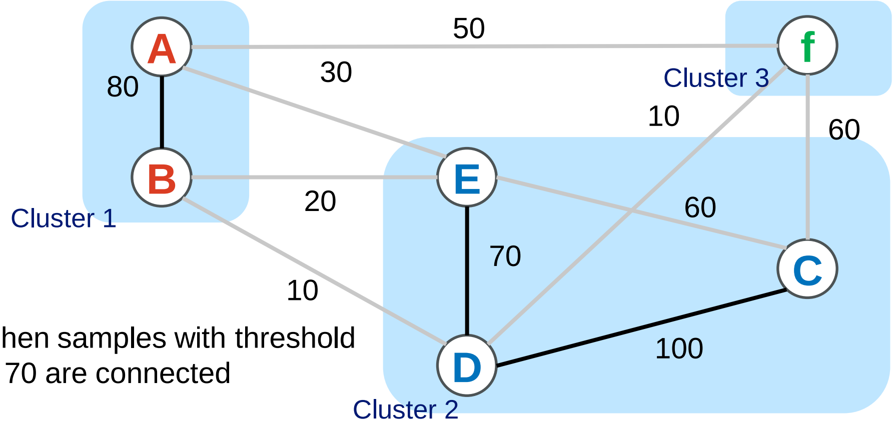
+

Evaluation using unanalyzed, up-to-date samples

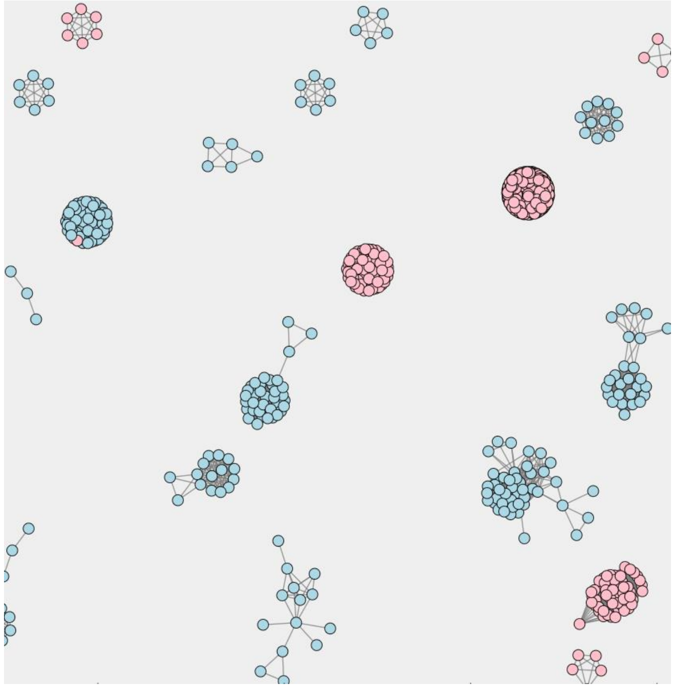
- Includes **samples not identified as** malware
- Evaluate use in actual operations

Overview of Clustering Methods

- gimpfuzzy similarity-based clustering
  1. Calculate gimpfuzzy of samples for clustering
  2. Calculate gimpfuzzy similarity between samples for clustering
  3. Edge-connect samples with similarity above a threshold
  4. Connected samples are considered as a cluster.



When samples with threshold ≥ 70 are connected

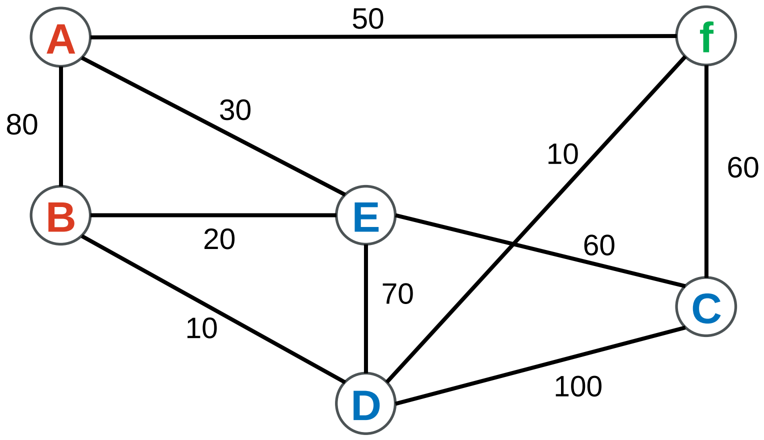


Scoring Matrix

- Calculate scores for all combinations of samples
  - Scoring of string similarity from 0~100 based on edit distance
  - Create an “adjacency matrix” and consider an undirected graph

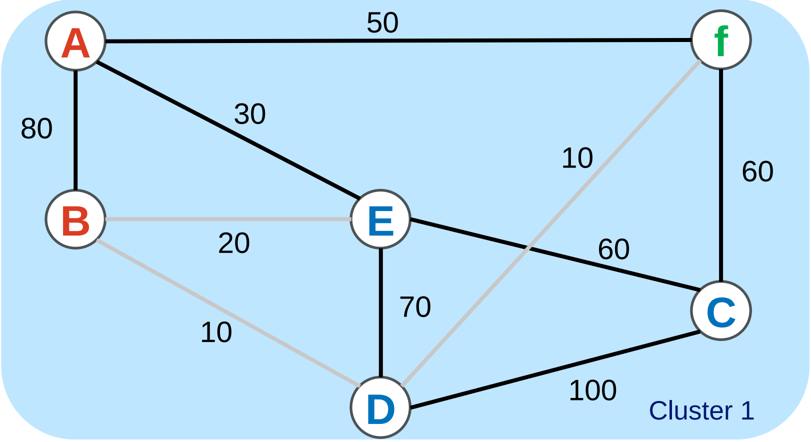
Color corresponds to family

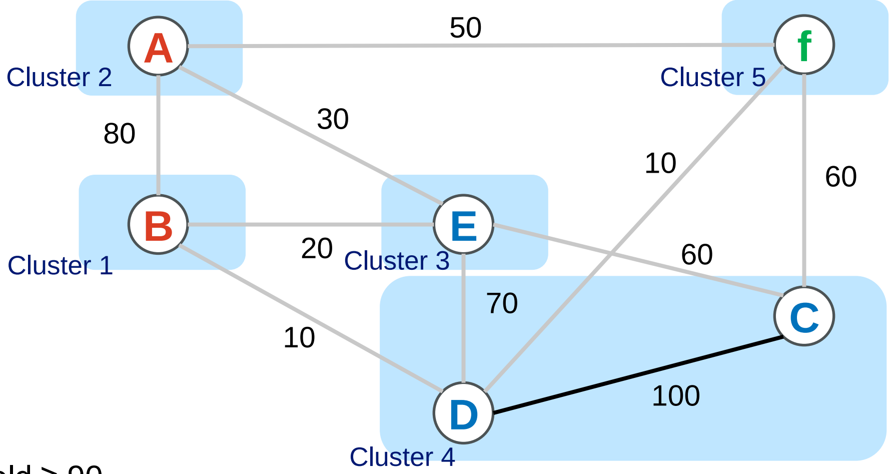
	A	B	C	D	E	f
A	0	80	0	0	30	50
B	80	0	0	10	20	0
C	0	0	0	100	60	60
D	0	10	100	0	70	10
E	30	20	60	70	0	0
f	50	00	60	10	0	0



How to Evaluate Clustering

- How to cut the threshold: what constitutes good clustering?
  - Low threshold case: a small number of large clusters are formed
  - High threshold case: a large number of small clusters are formed





Threshold ≥ 30
◎Many samples can be tied together (**integrity** ↑)
Decrease in classification accuracy within a △ cluster (**homogeneity** ↓)

Threshold ≥ 90
◎ High classification accuracy within clusters (**homogeneity** ↑)
△ Clusters are too separated to be meaningful (**integrity** ↓)

How to Evaluate Clustering

- **Harmonic mean** is considered in trade-off between the two
- Using **V-measure**[2] implemented in scikit-learn for evaluation
  - Homogeneity Score h: The higher the percentage of a single correct answer group in a given cluster, the score is better
  - Integrity score c : The fewer cluster into which a given group of correct answers is classified, the score is better

$$V_\beta = (1+\beta)\cdot\frac{h\cdot c}{h+c}$$

$$h = \begin{cases} 1 & \text{if } H(C,K)=0 \\ 1-\frac{H(C|K)}{H(C)} & \text{else} \end{cases} \qquad (1)$$

where

$$H(C|K) = -\sum_{k=1}^{|K|}\sum_{c=1}^{|C|}\frac{a_{ck}}{N}\log\frac{a_{ck}}{\sum_{c=1}^{|C|}a_{ck}}$$

$$H(C) = -\sum_{c=1}^{|C|}\frac{\sum_{k=1}^{|K|}a_{ck}}{n}\log\frac{\sum_{k=1}^{|K|}a_{ck}}{n}$$

$$c = \begin{cases} 1 & \text{if } H(K,C)=0 \\ 1-\frac{H(K|C)}{H(K)} & \text{else} \end{cases} \qquad (2)$$

where

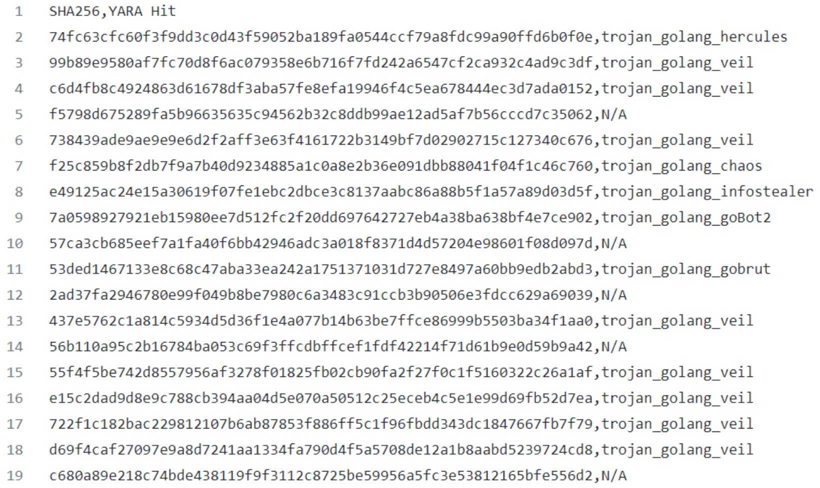
$$H(K|C) = -\sum_{c=1}^{|C|}\sum_{k=1}^{|K|}\frac{a_{ck}}{N}\log\frac{a_{ck}}{\sum_{k=1}^{|K|}a_{ck}}$$

$$H(K) = -\sum_{k=1}^{|K|}\frac{\sum_{c=1}^{|C|}a_{ck}}{n}\log\frac{\sum_{c=1}^{|C|}a_{ck}}{n}$$

[2] https://www.researchgate.net/publication/221012656_V-Measure_A_Conditional_Entropy-Based_External_Cluster_Evaluation_Measure

Evaluation

- **What is the correct classification in the first place?**
  - Minor variants and version differences in malware families
  - malware family
  - Rough malware features
- **Evaluation by paloalto dataset [3]**
  - Analyzed samples with families classified by YARA
  - Exclude samples that did not have a family name
  - Evaluate the classified results and family name with V-measure



```
SHA256,YARA Hit
74fc63cfc60f3f9dd3c0d43f59052ba189fa0544ccf79a8fdc99a90ffd6b0f0e,trojan_golang_hercules
99b89e9580af7fc70d8f6ac079358e6b716f7fd242a6547cf2ca932c4ad9c3df,trojan_golang_veil
c6d4fb8c4924863d61678df3aba57fe8efa19946f4c5ea678444ec3d7ada0152,trojan_golang_veil
f5798d675289fa5b96635635c94562b32c8ddb99ae12ad5af7b56cccd7c35062,N/A
738439ade9ae9e9e6d2f2aff3e63f4161722b3149bf7d02902715c127340c676,trojan_golang_veil
f25c859b8f2db7f9a7b40d9234885a1c0a8e2b36e091dbb88041f04f1c46c760,trojan_golang_chaos
e49125ac24e15a30619f07fe1ebc2dbce3c8137aabc86a88b5f1a57a89d03d5f,trojan_golang_infostealer
7a0598927921eb15980ee7d512fc2f20dd697642727eb4a38ba638bf4e7ce902,trojan_golang_goBot2
57ca3cb685eef7a1fa40f6bb42946adc3a018f8371d4d57204e98601f08d097d,N/A
53ded1467133e8c68c47aba33ea242a1751371031d727e8497a60bb9edb2abd3,trojan_golang_gobrut
2ad37fa2946780e99f049b8be7980c6a3483c91ccb3b90506e3fdcc629a69039,N/A
437e5762c1a814c5934d5d36f1e4a077b14b63be7ffce86999b5503ba34f1aa0,trojan_golang_veil
56b110a95c2b16784ba053c69f3ffcdbffcef1fdf42214f71d61b9e0d59b9a42,N/A
55f4f5be742d8557956af3278f01825fb02cb90fa2f27f0c1f5160322c26a1af,trojan_golang_veil
e15c2dad9d8e9c788cb394aa04d5e070a50512c25eceb4c5e1e99d69fb52d7ea,trojan_golang_veil
722f1c182bac229812107b6ab87853f886ff5c1f96fbdd343dc1847667fb7f79,trojan_golang_veil
d69f4caf27097e9a8d7241aa1334fa790d4f5a5708de12a1b8aabd5239724cd8,trojan_golang_veil
c680a89e218c74bde438119f9f3112c8725be59956a5fc3e53812165bfe556d2,N/A
```

sample SHA256

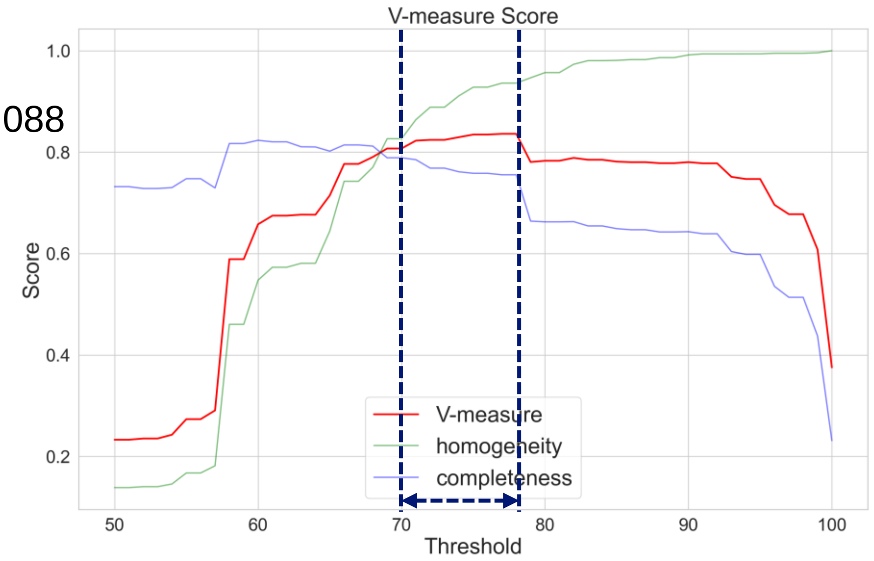
Matches. YARA rules

[3]https://github.com/pan-unit42/iocs/blob/master/golang_malware_results.csv

Evaluation

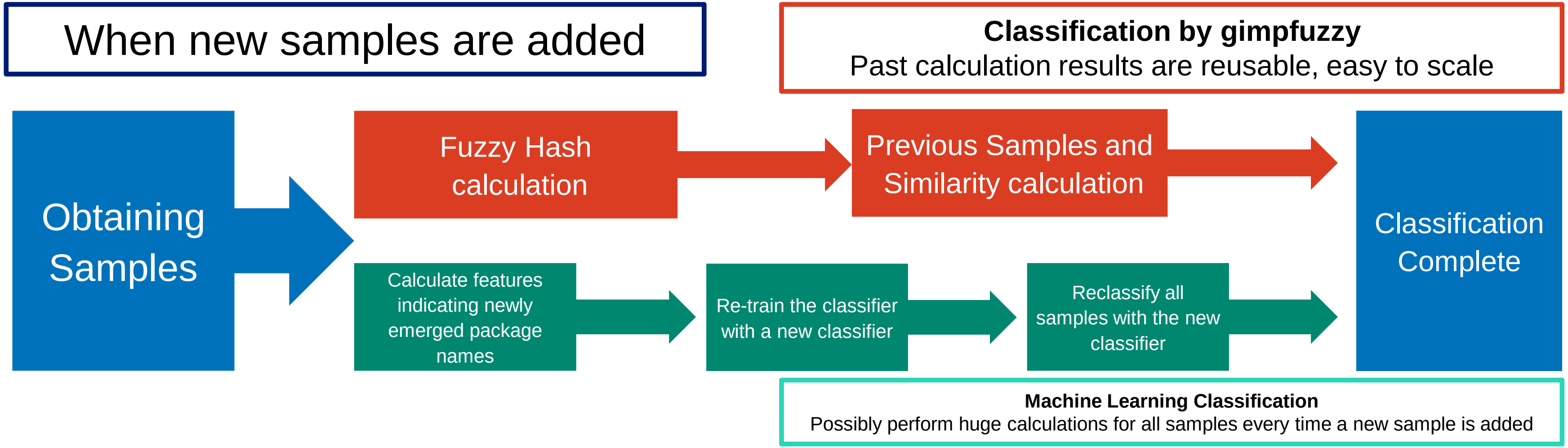
- paloalto data set
  - Number of samples: 10,700
  - Number of samples downloadable from VT: 7,088
  - Of which, family name is indicated : 5,808

- Evaluation by V-measure
  - Evaluation of classification assuming that the family is correct
  - Best classification accuracy at a threshold of about 70~80



Advantages of Our Methods

- Advantages of gimpfuzzy's similarity-based clustering compared to machine learning
  - Low calculational complexity
  - Less susceptible to time variation



When new samples are added

Classification by gimpfuzzy
Past calculation results are reusable, easy to scale

Obtaining Samples

Fuzzy Hash calculation

Previous Samples and Similarity calculation

Calculate features indicating newly emerged package names

Re-train the classifier with a new classifier

Reclassify all samples with the new classifier

Classification Complete

Machine Learning Classification
Possibly perform huge calculations for all samples every time a new sample is added

Applying to “Wild” Binaries

The Rule for Wild Mal-Gopher Families.

Applying to "Wild" Binaries

- Clustering evaluation using actual observed "wild" samples

Evaluation using analyzed samples

- Classify **only samples identified as** malware
- Evaluate the validity and accuracy of clustering

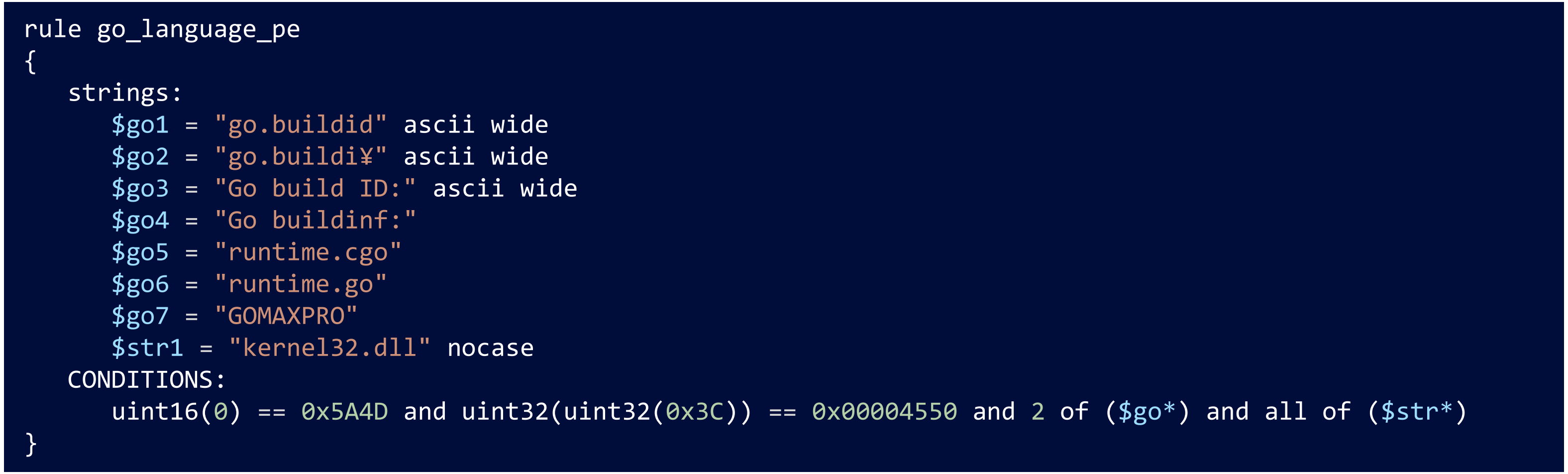
✚

Evaluation using unanalyzed, up-to-date samples

- Includes **samples not identified as** malware
- Evaluate use in actual operations

Collecting Wild Binaries

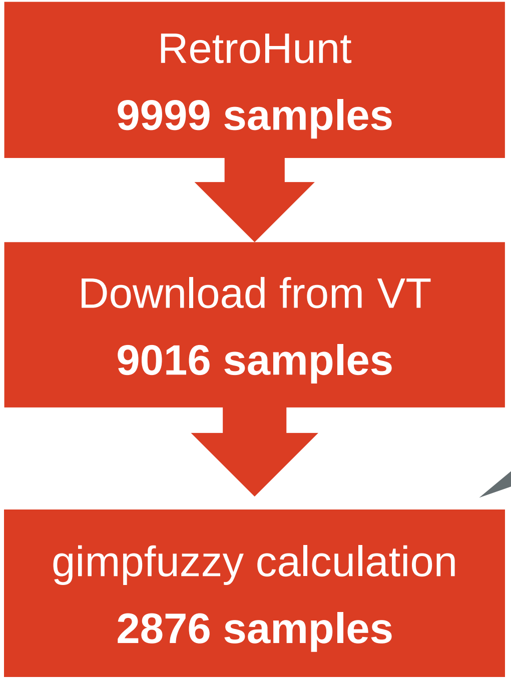
- Download the samples that matched the following YARA with VT’s Retrohunt
  - Samples that matched Windows binaries made by golang



```
rule go_language_pe
{
   strings:
      $go1 = "go.buildid" ascii wide
      $go2 = "go.buildi¥" ascii wide
      $go3 = "Go build ID:" ascii wide
      $go4 = "Go buildinf:"
      $go5 = "runtime.cgo"
      $go6 = "runtime.go"
      $go7 = "GOMAXPRO"
      $str1 = "kernel32.dll" nocase
   CONDITIONS:
      uint16(0) == 0x5A4D and uint32(uint32(0x3C)) == 0x00004550 and 2 of ($go*) and all of ($str*)
}
```

Collecting Wild Binaries

- Sample collection results



RetroHunt

9999 samples

Download from VT

9016 samples

gimpfuzzy calculation

2876 samples

- Samples collected by Retrohunt

  (Approximately the last 3 months from 2022/12)

- Downloaded samples without duplicates such as Subfile

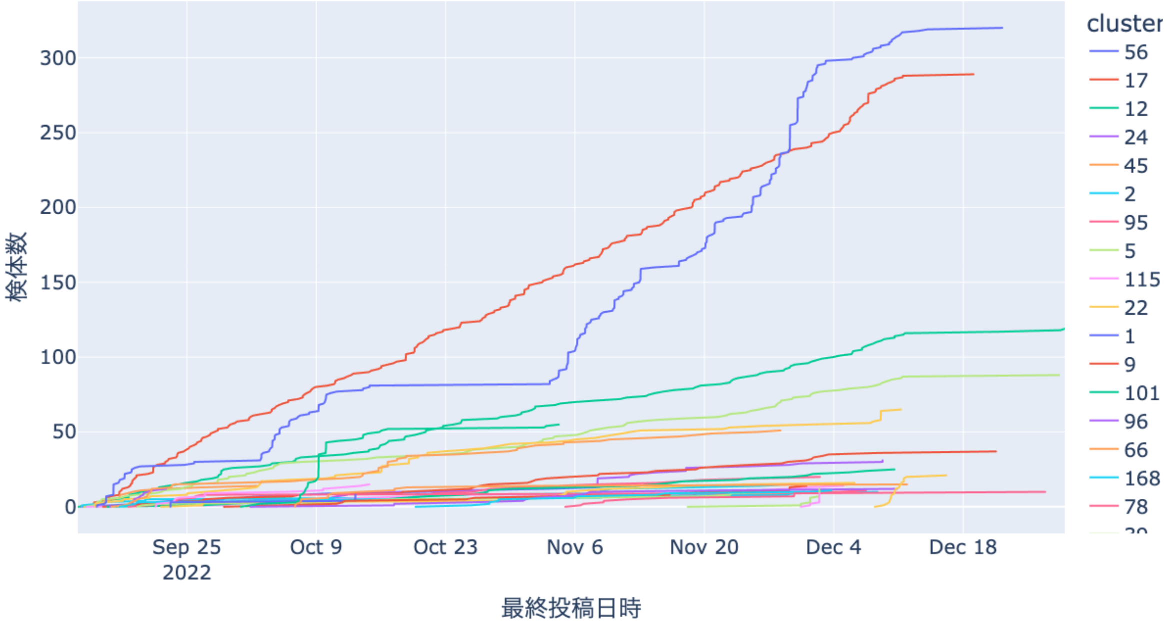
In case of insufficient bytes of ssdeep input

Can be improved by using TLSH, etc.

- Samples for which pclntab analysis + gimpfuzzy calculation was possible
- UPX are unpacked and analyzed

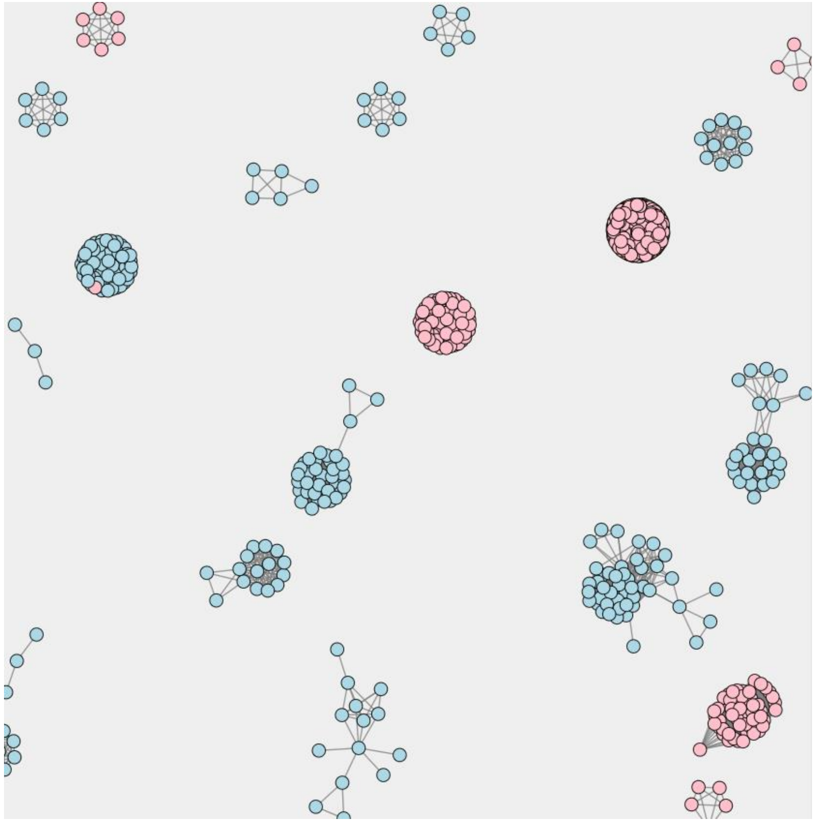
Clustering Result

- Created 1093 clusters for 2867 samples



Clustering Result

- Cluster Visualization
  - Implemented with Python’s bokeh.io
  - Create interactive “moveable” graphs
  - Red node have 10 or more malignant determinations by VT



Demonstration : Cluster Visualization

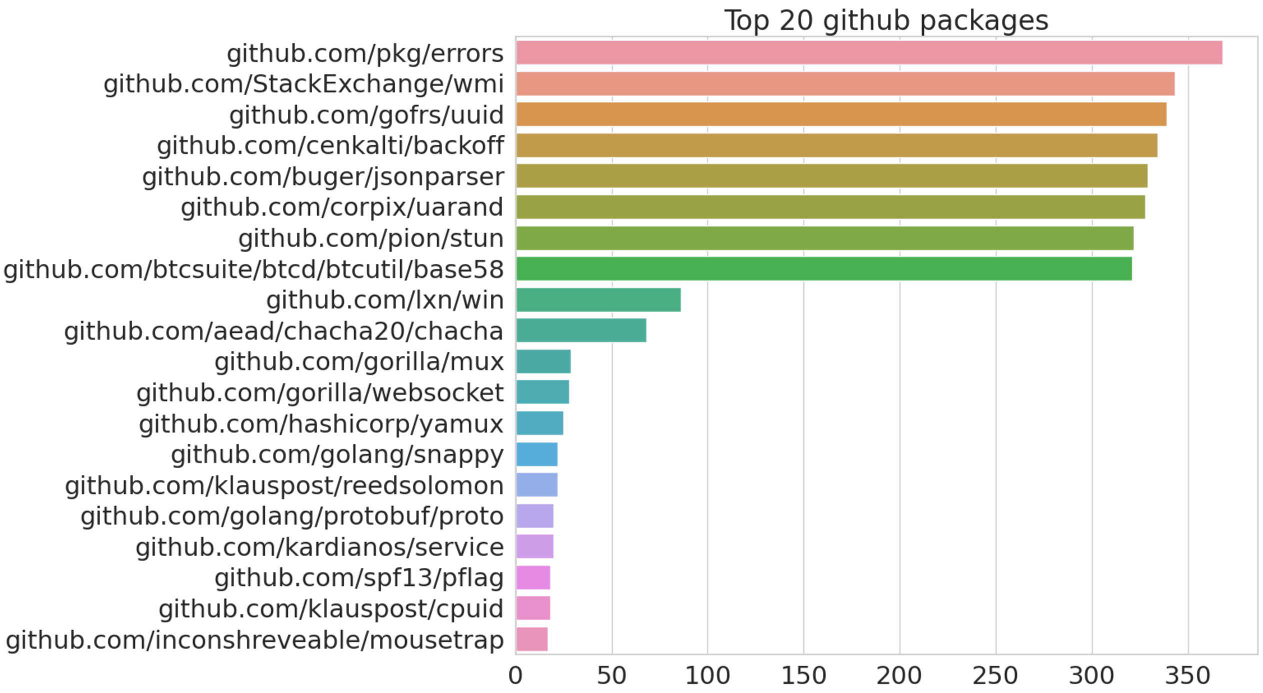
Demonstration

Case Study 1 / GitHub Packages

- Golang can specify github repository for packages
- Focused on samples with more than 10 malignant determinations, 705 repository names recovered.
  - Including repositories that are considered private.
- Interesting repository name restored
  - Packages that generate random UAs (corpix/uarand etc.)
  - Packages that conduct Process Invoking (inconshreveable/mousetrap, etc.)
  - Malware creation frameworks (tiagorlampert/CHAOS, etc.)
  - Multi-hop proxies (Dliv3/Venom, etc.)
  - Post-Exploitation framework (Ne0nd0g/merlin, etc.)

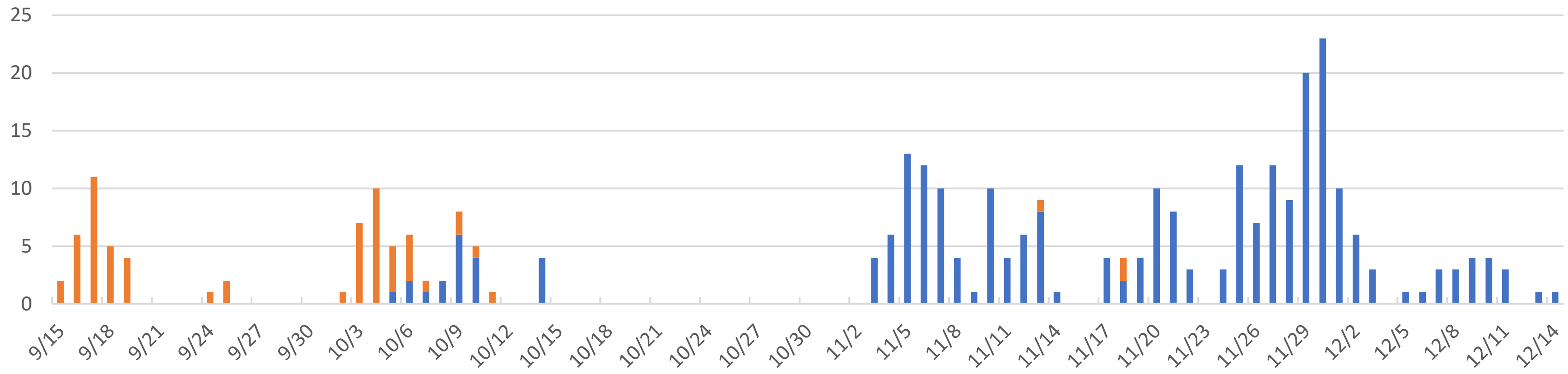
Case Study 1 / GitHub Packages

Top 20 github repositories that appeared in malignant samples



Case Study 2 : Detecting Additional Malware Features

- Detect small changes in GimpFuzzy values in clusters of malignant samples and observe temporal changes
  - 768:KZZ99PdnRrLXT3UbhHPBj/RqJgvm+HHHyScP0OhZlXCPlNvxtWrX7G/VAmWeEX
  - 768:KZZ99PdnRrLXT3UbhHPBj/RqJgvm+HHHyScP0OhZlXCPlNvxtWrX7G/V3mWeEX

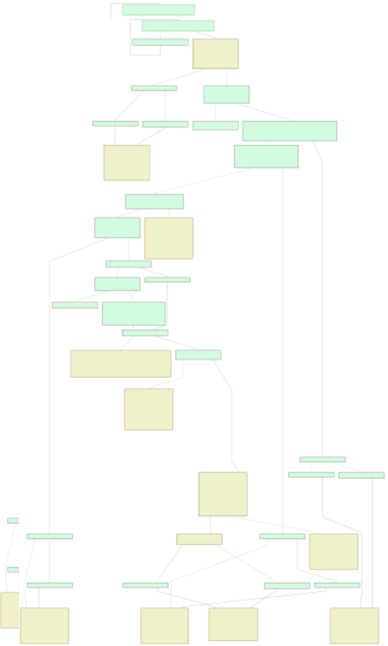


Case Study 2: Detecting malware functionality additions

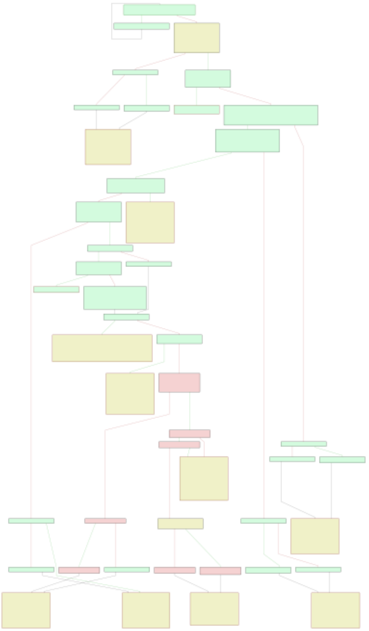
- Changed functions also changed logic.
  - Function name that was changed
    - 768:~VAmWeEX

      → application/pesignaturetest/wincert.GetPostalCode
    - 768:~V3mWeEX

      → application/pesignaturetest/wincert.Extract



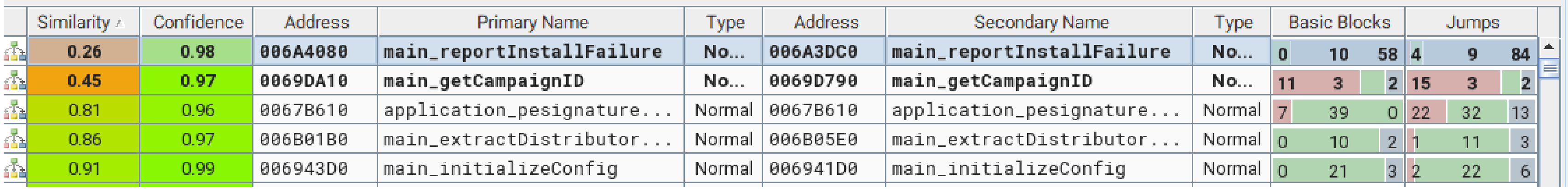
wincert.GetPostarCode



wincert.Extract

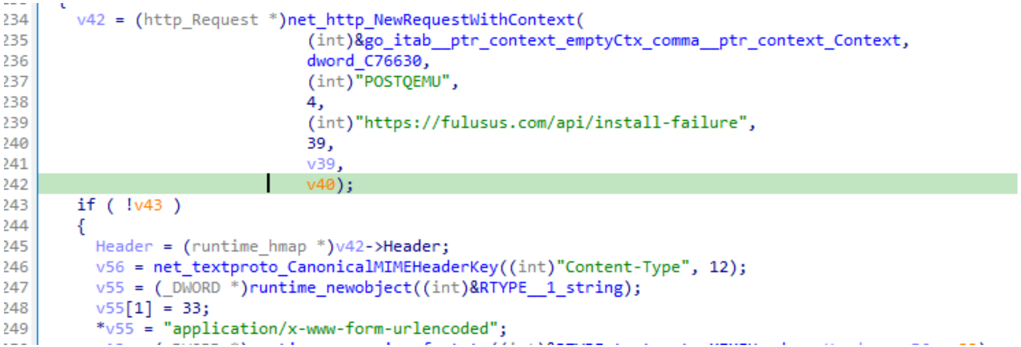
Case Study 2: Detecting malware functionality additions

- Some functions with unchanged function names have logic changes



	Similarity	Confidence	Address	Primary Name	Type	Address	Secondary Name	Type	Basic Blocks	Jumps
	0.26	0.98	006A4080	main_reportInstallFailure	No...	006A3DC0	main_reportInstallFailure	No...	0 10 58	4 9 84
	0.45	0.97	0069DA10	main_getCampaignID	No...	0069D790	main_getCampaignID	No...	11 3 2	15 3 2
	0.81	0.96	0067B610	application_pesignature...	Normal	0067B610	application_pesignature...	Normal	7 39 0	22 32 13
	0.86	0.97	006B01B0	main_extractDistributor...	Normal	006B05E0	main_extractDistributor...	Normal	0 10 2	1 11 3
	0.91	0.99	006943D0	main_initializeConfig	Normal	006941D0	main_initializeConfig	Normal	0 21 3	2 22 6

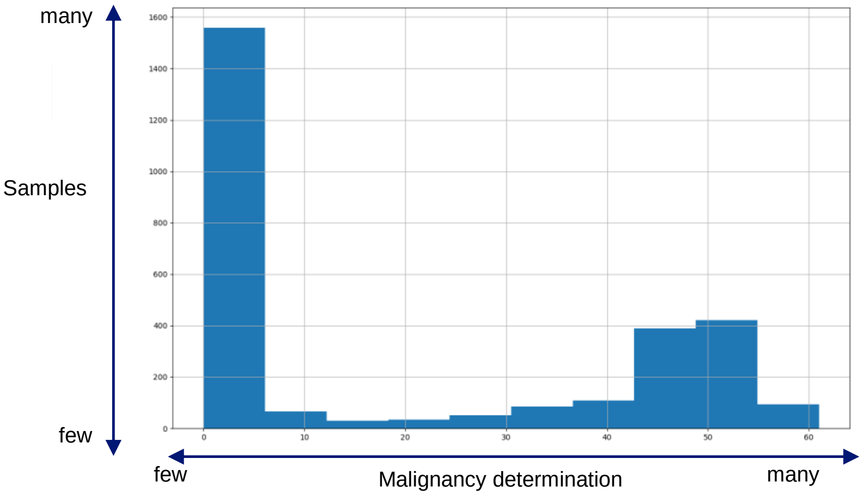
- main_reportInstllFailure is added to the communication functionality.

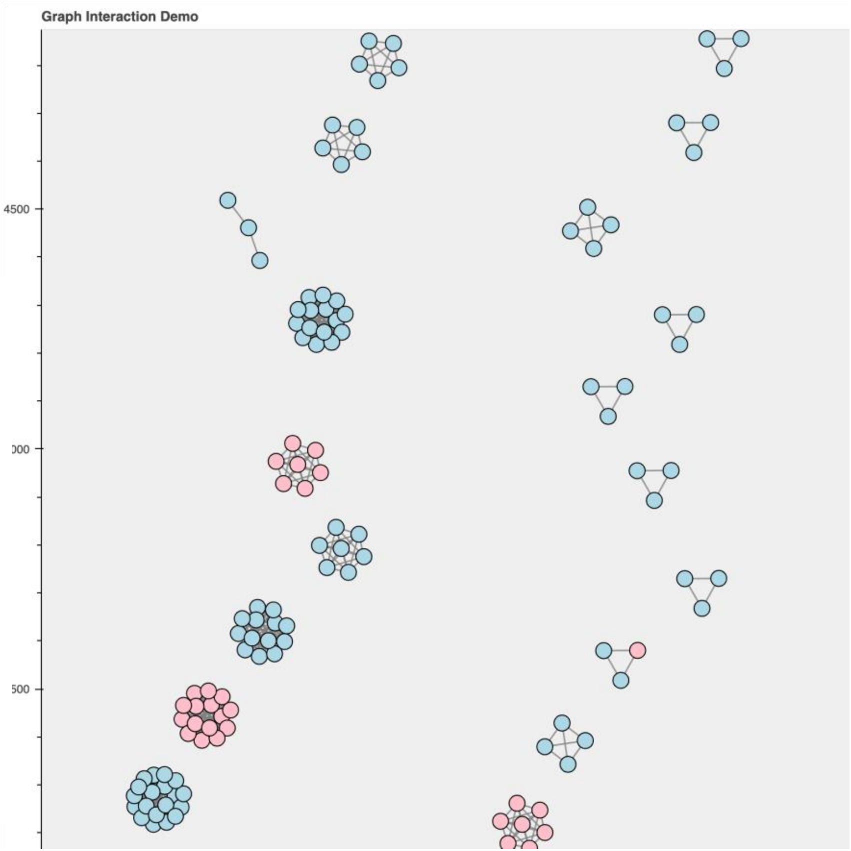


```
  v42 = (http_Request *)net_http_NewRequestWithContext(
                          (int)&go_itab__ptr_context_emptyCtx_comma__ptr_context_Context,
                          dword_C76630,
                          (int)"POSTQEMU",
                          4,
                          (int)"https://fulusus.com/api/install-failure",
                          39,
                          v39,
                          v40);
  if ( !v43 )
  {
    Header = (runtime_hmap *)v42->Header;
    v56 = net_textproto_CanonicalMIMEHeaderKey((int)"Content-Type", 12);
    v55 = (_DWORD *)runtime_newobject((int)&RTYPE__1_string);
    v55[1] = 33;
    *v55 = "application/x-www-form-urlencoded";
```

Case Study ③ : Legitimate Files

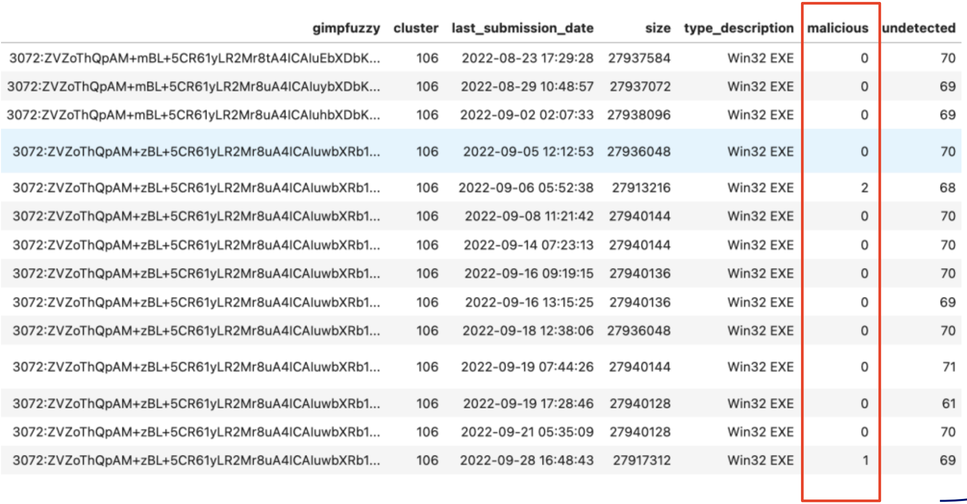
- In reality, legitimate files dominate.
  - Same for samples submitted to VT.
  - Is clustering of regular files possible?



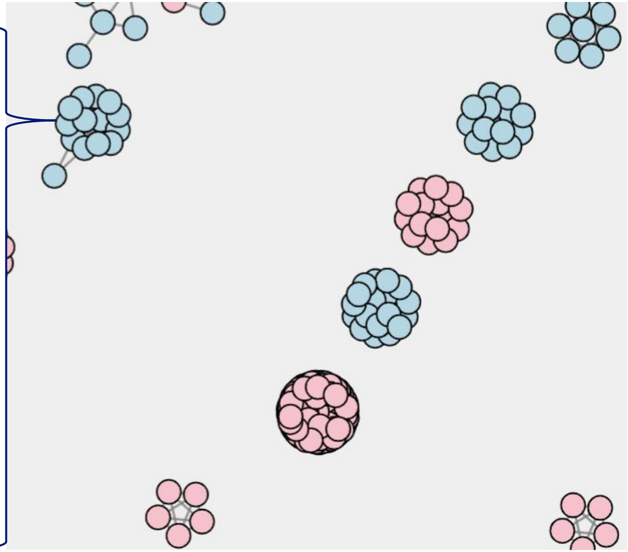


Case Study ③ : Legitimate Files

- Even samples that appear to be legitimate files can cluster.
  - samples submitted to VT are not necessarily only malignant files

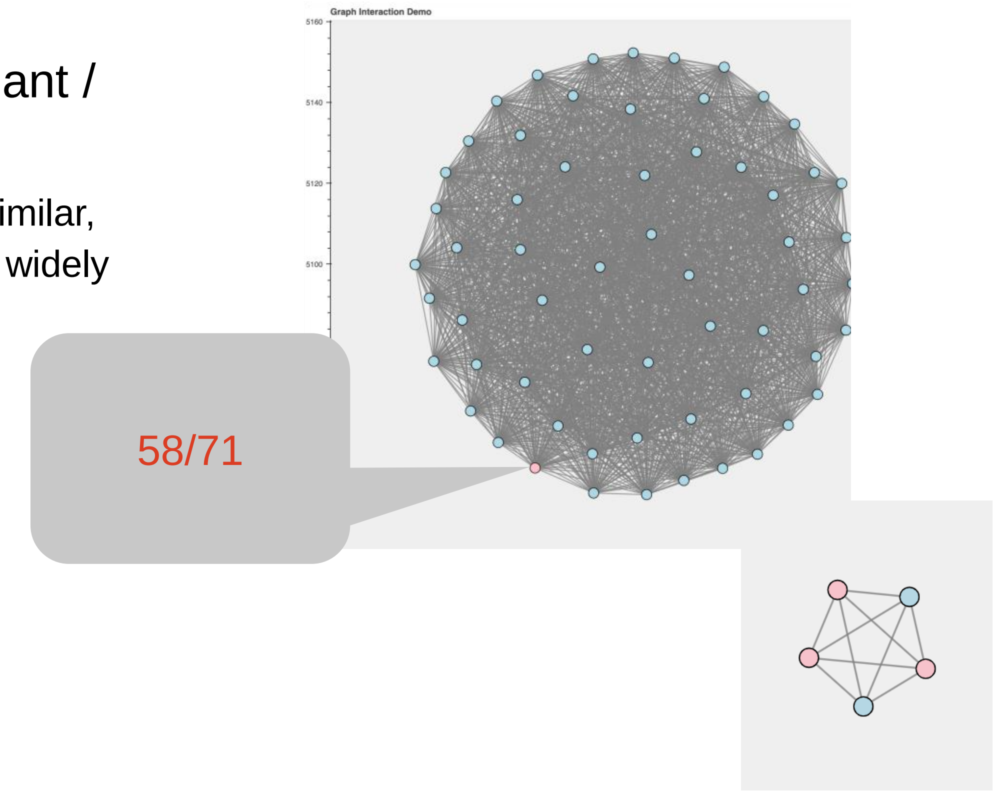


gimpfuzzy	cluster	last_submission_date	size	type_description	malicious	undetected
3072:ZVZoThQpAM+mBL+5CR61yLR2Mr8tA4ICAluEbXDbK...	106	2022-08-23 17:29:28	27937584	Win32 EXE	0	70
3072:ZVZoThQpAM+mBL+5CR61yLR2Mr8uA4ICAluybXDbK...	106	2022-08-29 10:48:57	27937072	Win32 EXE	0	69
3072:ZVZoThQpAM+mBL+5CR61yLR2Mr8uA4ICAluhbXDbK...	106	2022-09-02 02:07:33	27938096	Win32 EXE	0	69
3072:ZVZoThQpAM+zBL+5CR61yLR2Mr8uA4ICAluwbXRb1...	106	2022-09-05 12:12:53	27936048	Win32 EXE	0	70
3072:ZVZoThQpAM+zBL+5CR61yLR2Mr8uA4ICAluwbXRb1...	106	2022-09-06 05:52:38	27913216	Win32 EXE	2	68
3072:ZVZoThQpAM+zBL+5CR61yLR2Mr8uA4ICAluwbXRb1...	106	2022-09-08 11:21:42	27940144	Win32 EXE	0	70
3072:ZVZoThQpAM+zBL+5CR61yLR2Mr8uA4ICAluwbXRb1...	106	2022-09-14 07:23:13	27940144	Win32 EXE	0	70
3072:ZVZoThQpAM+zBL+5CR61yLR2Mr8uA4ICAluwbXRb1...	106	2022-09-16 09:19:15	27940136	Win32 EXE	0	70
3072:ZVZoThQpAM+zBL+5CR61yLR2Mr8uA4ICAluwbXRb1...	106	2022-09-16 13:15:25	27940136	Win32 EXE	0	69
3072:ZVZoThQpAM+zBL+5CR61yLR2Mr8uA4ICAluwbXRb1...	106	2022-09-18 12:38:06	27936048	Win32 EXE	0	70
3072:ZVZoThQpAM+zBL+5CR61yLR2Mr8uA4ICAluwbXRb1...	106	2022-09-19 07:44:26	27940144	Win32 EXE	0	71
3072:ZVZoThQpAM+zBL+5CR61yLR2Mr8uA4ICAluwbXRb1...	106	2022-09-19 17:28:46	27940128	Win32 EXE	0	61
3072:ZVZoThQpAM+zBL+5CR61yLR2Mr8uA4ICAluwbXRb1...	106	2022-09-21 05:35:09	27940128	Win32 EXE	0	70
3072:ZVZoThQpAM+zBL+5CR61yLR2Mr8uA4ICAluwbXRb1...	106	2022-09-28 16:48:43	27917312	Win32 EXE	1	69



Case Study ④ : Floxif

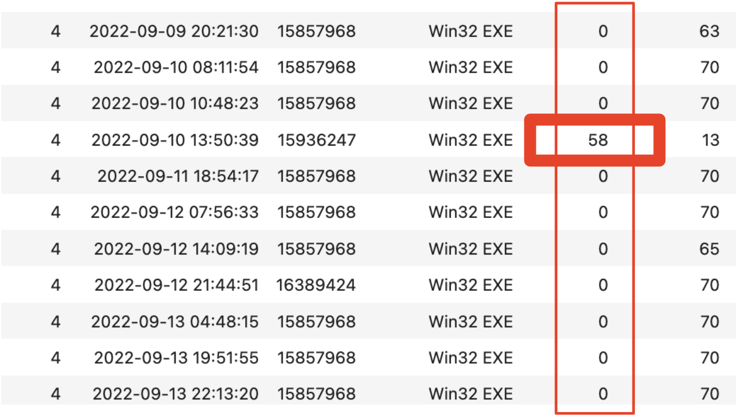
- Mixed clusters of malignant / benign determinations
  - Even though gimpfuzzy is similar, malignancy judgments vary widely within clusters



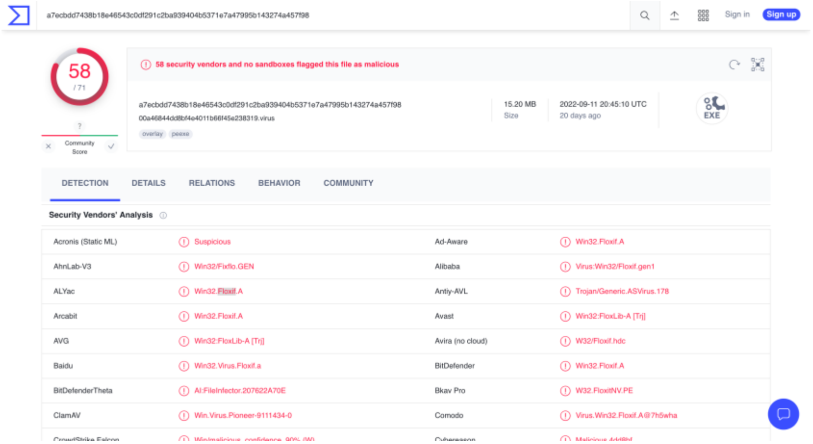
58/71

Case Study ④ : Floxif

- Highly malicious samples lurking in legitimate file clusters
  - We found a highly malignant Floxif sample that mimicked the following program
    - psiphone-tunnel-core
    - Acronis Cyber Protect
  - It is difficult to determine malignancy/benignity in some samples with unsupervised clustering alone
  - Correct results as sample profiling



4	2022-09-09 20:21:30	15857968	Win32 EXE	0	63
4	2022-09-10 08:11:54	15857968	Win32 EXE	0	70
4	2022-09-10 10:48:23	15857968	Win32 EXE	0	70
4	2022-09-10 13:50:39	15936247	Win32 EXE	58	13
4	2022-09-11 18:54:17	15857968	Win32 EXE	0	70
4	2022-09-12 07:56:33	15857968	Win32 EXE	0	70
4	2022-09-12 14:09:19	15857968	Win32 EXE	0	65
4	2022-09-12 21:44:51	16389424	Win32 EXE	0	70
4	2022-09-13 04:48:15	15857968	Win32 EXE	0	70
4	2022-09-13 19:51:55	15857968	Win32 EXE	0	70
4	2022-09-13 22:13:20	15857968	Win32 EXE	0	70



a7ecbdd7438b18e46543c0df291c2ba939404b5371e7a47995b143274a457f98

58 security vendors and no sandboxes flagged this file as malicious

a7ecbdd7438b18e46543c0df291c2ba939404b5371e7a47995b143274a457f98
00a46844dd8bf4e4011b66f45e238319.virus

overlay peexe

15.20 MB
Size

2022-09-11 20:45:10 UTC
20 days ago

Security Vendors' Analysis

Acronis (Static ML)	Suspicious	Ad-Aware	Win32.Floxif.A
AhnLab-V3	Win32/Fixflo.GEN	Alibaba	Virus:Win32/Floxif.gen1
ALYac	Win32.Floxif.A	Antiy-AVL	Trojan/Generic.ASVirus.178
Arcabit	Win32.Floxif.A	Avast	Win32:FloxLib-A [Trj]
AVG	Win32:FloxLib-A [Trj]	Avira (no cloud)	W32/Floxif.hdc
Baidu	Win32.Virus.Floxif.a	BitDefender	Win32.Floxif.A
BitDefenderTheta	AI:FileInfector.207622A70E	Bkav Pro	W32.FloxitNV.PE
ClamAV	Win.Virus.Pioneer-9111434-0	Comodo	Virus.Win32.Floxif.A@7h5wha
CrowdStrike Falcon	Win/malicious_confidence_90% (W)	Cybereason	Malicious.4dd8bf

Limitations

- Existence of samples for which gimpfuzzy cannot be calculated
  - Lower limit of ssdeep input size exists (>4KB). It can be replaced by TLSH, etc.
  - Analysis is interfered by packing, obfuscation, etc.

- Limitations of “unsupervised” classification
  - It is difficult to determine malignant / benign.
  - Separated by clusters to some extent, but some clusters with malignant and benign samples still exist.

Conclusions

The Rule for Wild Mal-Gopher Families.

Conclusions

- Presented on the following topics to apply gimpfuzzy to actual operations
  - YARA module implementation
  - Accuracy evaluation using analyzed samples
  - Application to samples submitted to VirusTotal

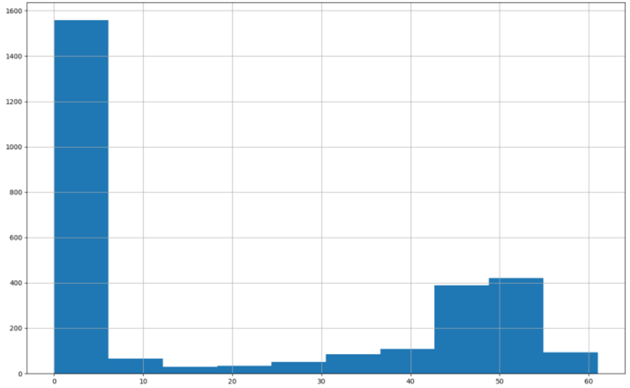
- YARA module and visualization scripts are to be released.

Appendix

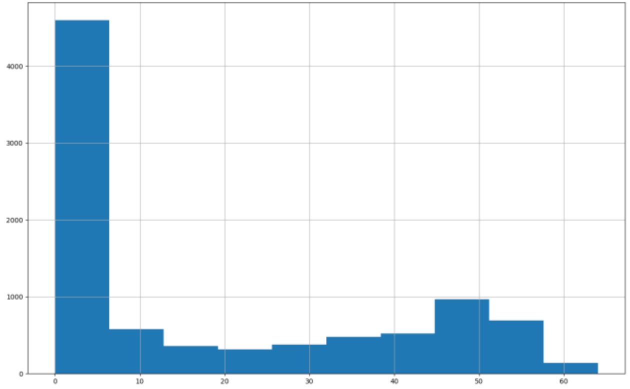
The Rule for Wild Mal-Gopher Families.

Appendix 1

- Number of malignancy determinations for samples collected in VT
  - Overwhelmingly less malignant files



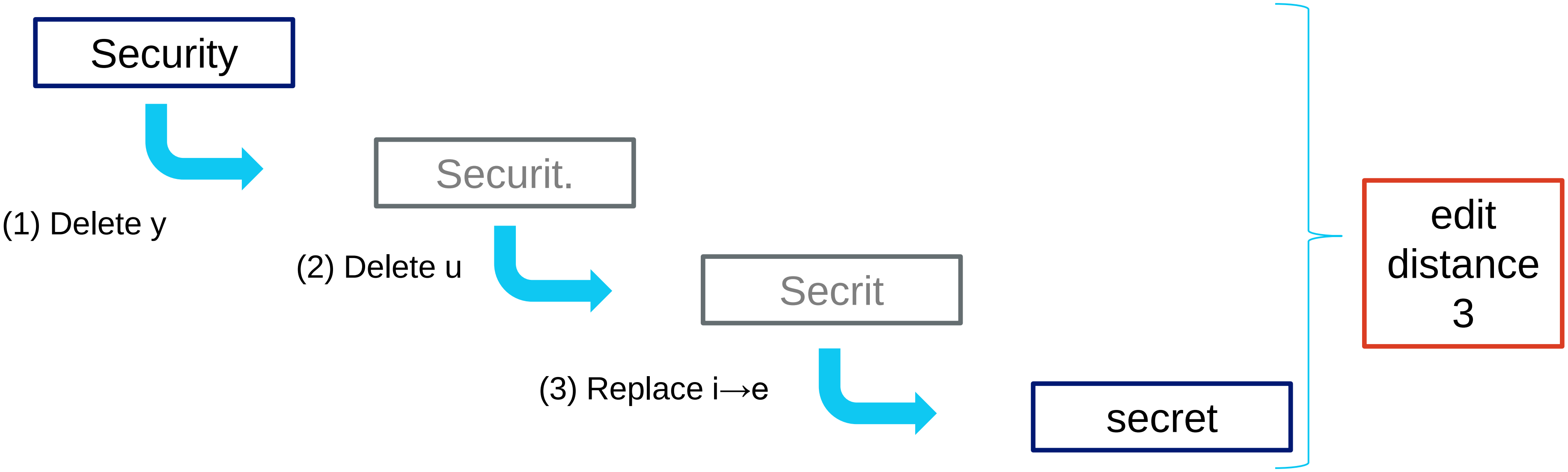
Analyzed samples
analyzed : 2835



Unanalyzed samples :
9999 samples

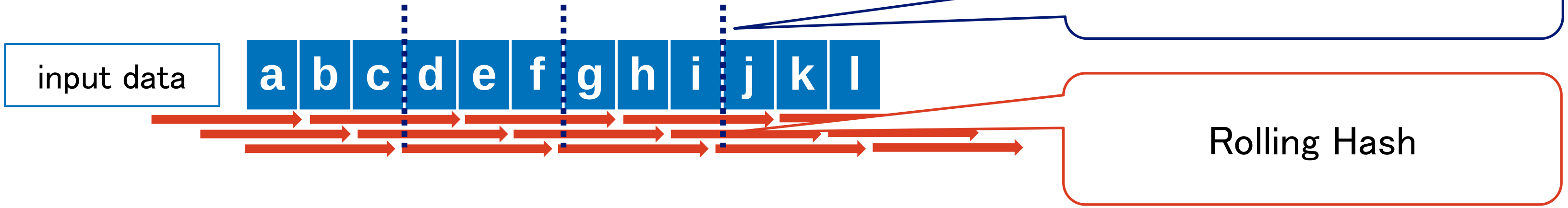
Appendix 2

- **Edit Distance (Levenshtein Distance)**
    - Classic method of showing string similarity
    - Minimum number of times one string can be converted to the other by inserting, deleting, or replacing a single character.

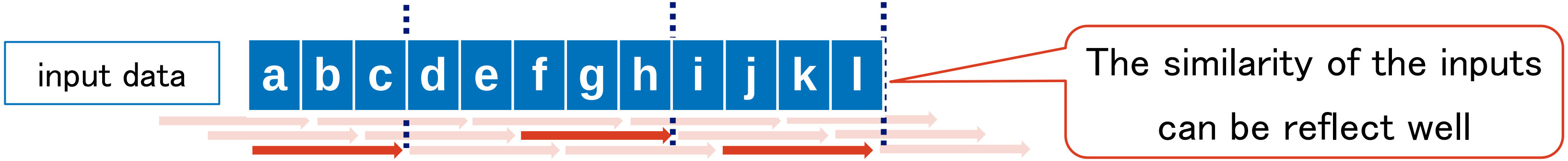


Appendix 3

- **ssdeep (Context Triggered Piecewise Hashing)**
  - Piecewise Hashing : Hash of divided part of the data
  - Rolling Hash : Hash for fixed-length partial data



- When the Rolling Hash reaches a certain value, it is split there and Piecewise Hashing is performed.
  - The triggering value is calculated based on the input data length


input data
a
b
c
d
e
f
g
h
i
j
k
l
The similarity of the inputs can be reflect well