Master of Clusters

15th October 2018

Misp Summit 04

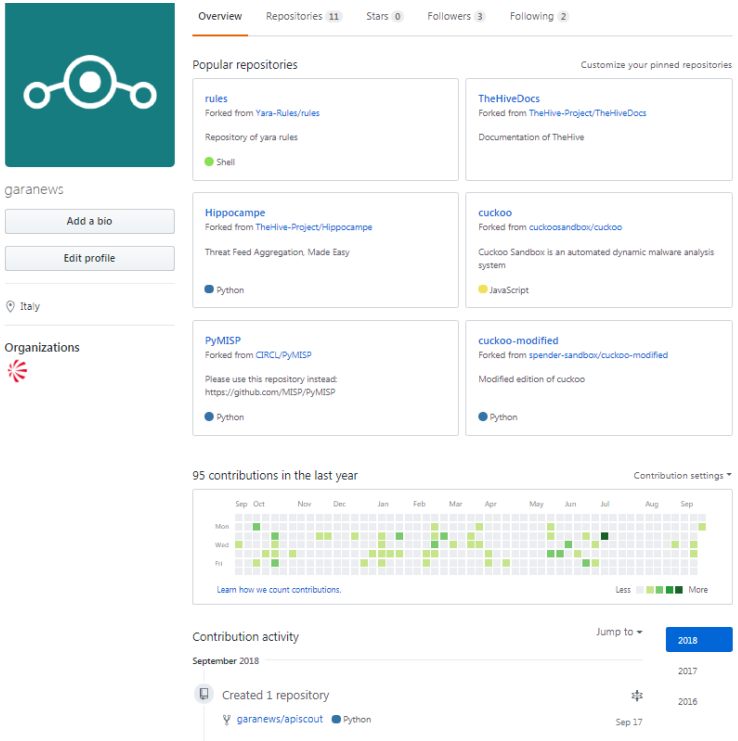
TLP:WHITE

LEONARDO

1948 • 2018


MISP-MALPEDIA
MASTER OF CLUSTERS
SEDRECO
X-AGENT
ISFB
PANDABANKER
DRIDEX

Whoami



Working for LDO-CERT
Incident Handling
Malware analysis
Forensics (network + system)

GIAC Certified Forensic Analyst (GCFA)
Open Source minded
Contributor of MISP, cuckoo, The Hive Project

Contact me:
andrea.garavaglia@leonardocompany.com

Why

The number of malicious samples that appear every day makes manual analysis impractical.

Although these samples belong to a limited number of malware families, it is difficult to categorize them automatically.

Tools used

MISP Threat Sharing

malpedia

TheHive

neo4j

danielplohmann / apiscout

Cortex

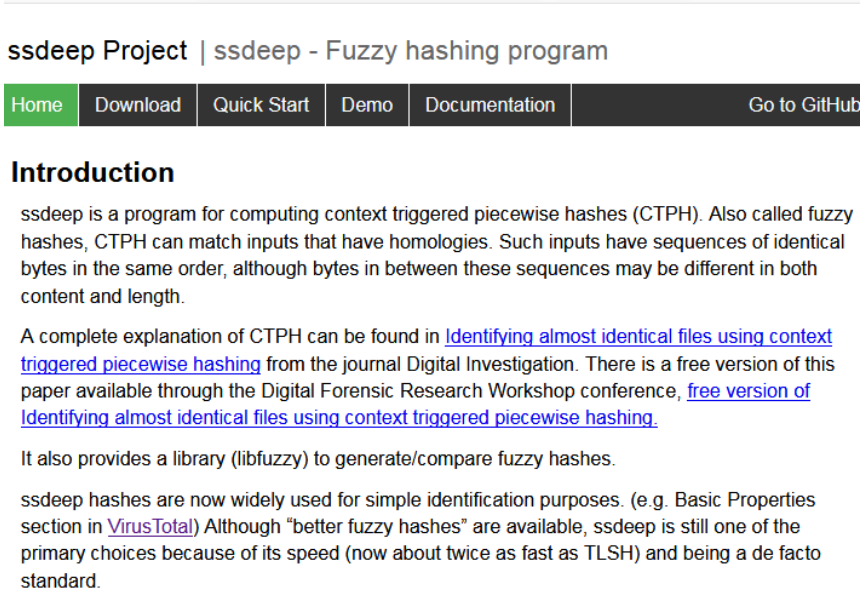
cuckoo

JPCERTCC / impfuzzy

AntV G6

ssdeep

Started with **ssdeep** (already supported by MISP) but often do not coincide with the similarity of malware samples:



ssdeep Project | ssdeep - Fuzzy hashing program

Home | Download | Quick Start | Demo | Documentation | Go to GitHub

Introduction

ssdeep is a program for computing context triggered piecewise hashes (CTPH). Also called fuzzy hashes, CTPH can match inputs that have homologies. Such inputs have sequences of identical bytes in the same order, although bytes in between these sequences may be different in both content and length.

A complete explanation of CTPH can be found in Identifying almost identical files using context triggered piecewise hashing from the journal Digital Investigation. There is a free version of this paper available through the Digital Forensic Research Workshop conference, free version of Identifying almost identical files using context triggered piecewise hashing.

It also provides a library (libfuzzy) to generate/compare fuzzy hashes.

ssdeep hashes are now widely used for simple identification purposes. (e.g. Basic Properties section in VirusTotal) Although "better fuzzy hashes" are available, ssdeep is still one of the primary choices because of its speed (now about twice as fast as TLSH) and being a de facto standard.

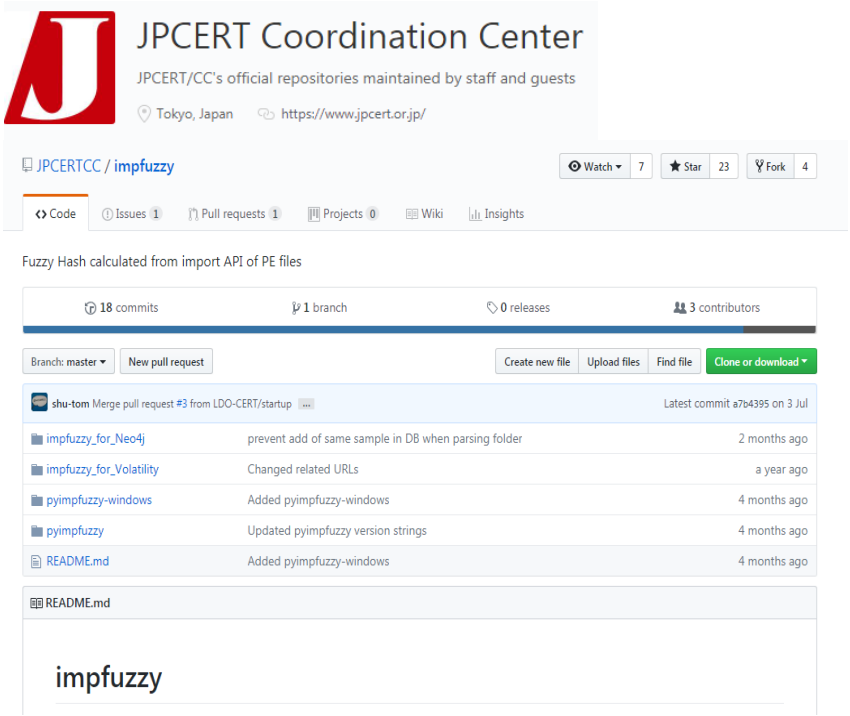
```
python mal_compare.py gootkit_2016-01-11
gootkit_2016-01-27

Ssdeep1:
384:gDUMQz+v+MwmvjMA2CiiOrAHhnmCmlwwH1m02
w1sDt2u6
OUMU9orCgmrmK1sDt2u

Ssdeep2:
384:/EP1GrQPrx4h3tjYTIfIYYXMXp7b4gYDySZiMeEoz
AZ97XrwH8
AY0206UYDtdg:y1GrQ5c2YrSZXYG7nW6HDtdg

Ssdeep Compare: 0
```

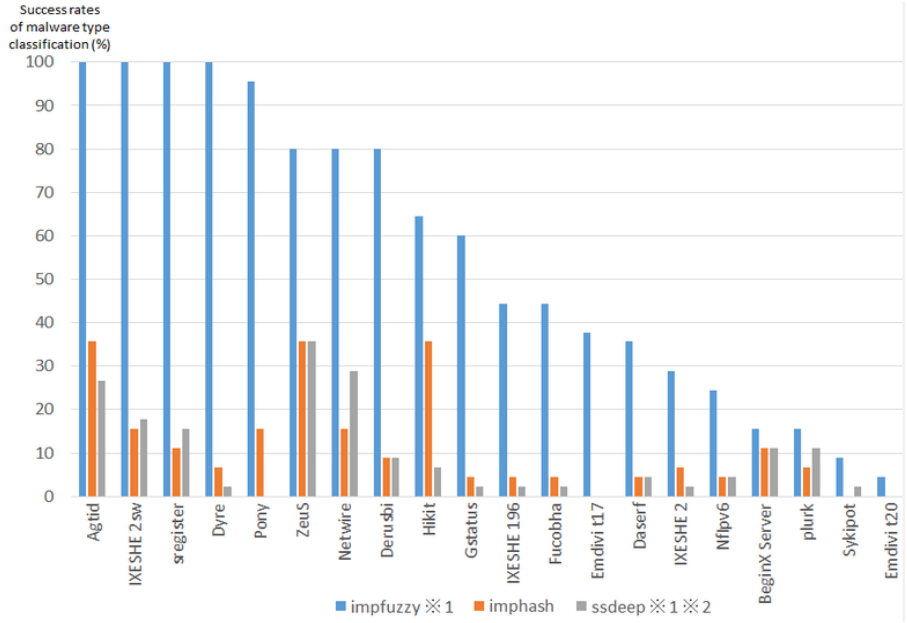
Impfuzzy from JP-CERT



The proposed method, as in imphash, calculates values from Import API, however, it also uses Fuzzy Hashing to calculate hash values of Import API, in order to supplement the shortcomings of imphash. With this process, a close value will be derived if just a part of Import API was added or modified.

Furthermore, it reduces time for calculation and enables efficient comparison by specifying the object of the hash value calculation to Import API (and not to the Fuzzy Hashing value of the whole executable file).

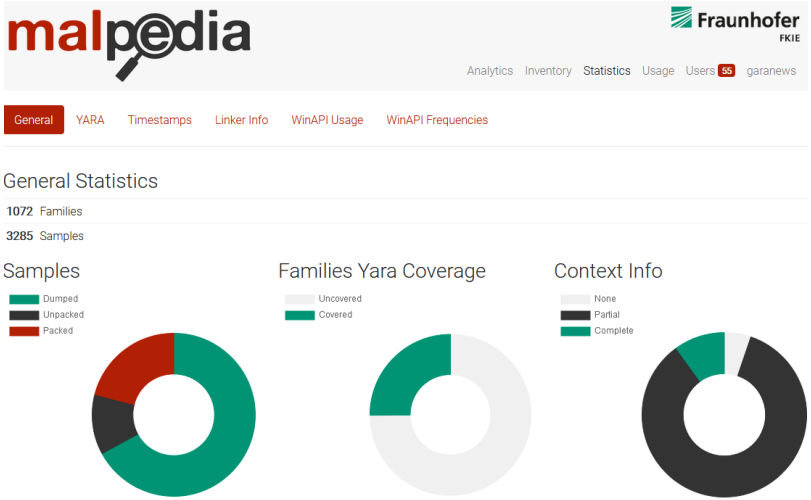
Evaluation of impfuzzy

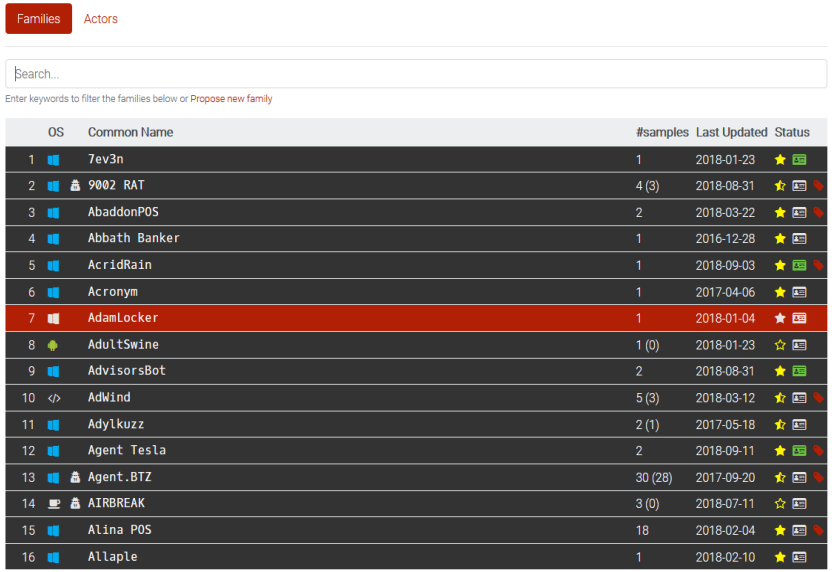



```
python mal_compare.py gootkit_2016-01-11 gootkit_2016-01-27

Ssdeep1:
384:gDUMQz+v+MwmvjMA2CiiOrAHhnmCmlwwH1m02w1sDt2u6
OUMU9orCgmrmK1sDt2u
Ssdeep2:
384:/EP1GrQPrx4h3tjYTIfIYYXMXp7b4gYDySZiMeEozAZ97XrwH8
AY0206UYDtdg:y1GrQ5c2YrSZXYG7nW6HDtdg
Ssdeep Compare: 0
ImpFuzzy1:
48:m00HV8ZisKz3sZOQ4tkQ53/ZRK4KINU/XjVKTp1H:oKZidz8Z4tk
4ZRK4KHzVUbH
ImpFuzzy2:
48:m00t2misKz3CZOv4tIQ53/Z7KINU/XjVKTp1S:/midzyZPtI4Z7KH
zVUbS
Impfuzzy Compare: 77
```

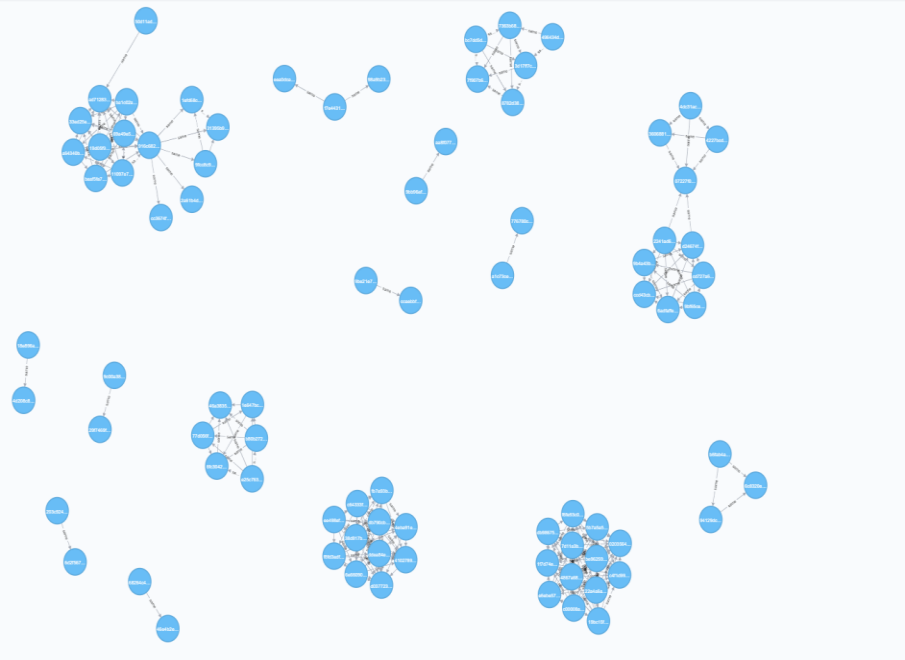
Malware families baseline: MALPEDIA





	OS	Common Name	#samples	Last Updated	Status
1		7ev3n	1	2018-01-23	
2		9002 RAT	4 (3)	2018-08-31	
3		AbaddonPOS	2	2018-03-22	
4		Abbath Banker	1	2016-12-28	
5		AcridRain	1	2018-09-03	
6		Acronym	1	2017-04-06	
7		AdamLocker	1	2018-01-04	
8		AdultSwine	1 (0)	2018-01-23	
9		AdvisorsBot	2	2018-08-31	
10		AdWind	5 (3)	2018-03-12	
11		Adylkuzz	2 (1)	2017-05-18	
12		Agent Tesla	2	2018-09-11	
13		Agent.BTZ	30 (28)	2017-09-20	
14		AIRBREAK	3 (0)	2018-07-11	
15		Alina POS	18	2018-02-04	
16		Allaple	1	2018-02-10	

Start to Clustering: impfuzzy for Neo4j



impfuzzy for Neo4j operates in the following sequence:

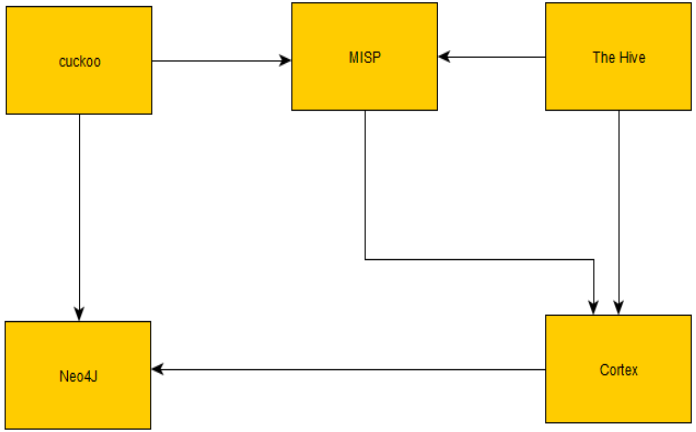
1.Calculate the similarity of malware using impfuzzy

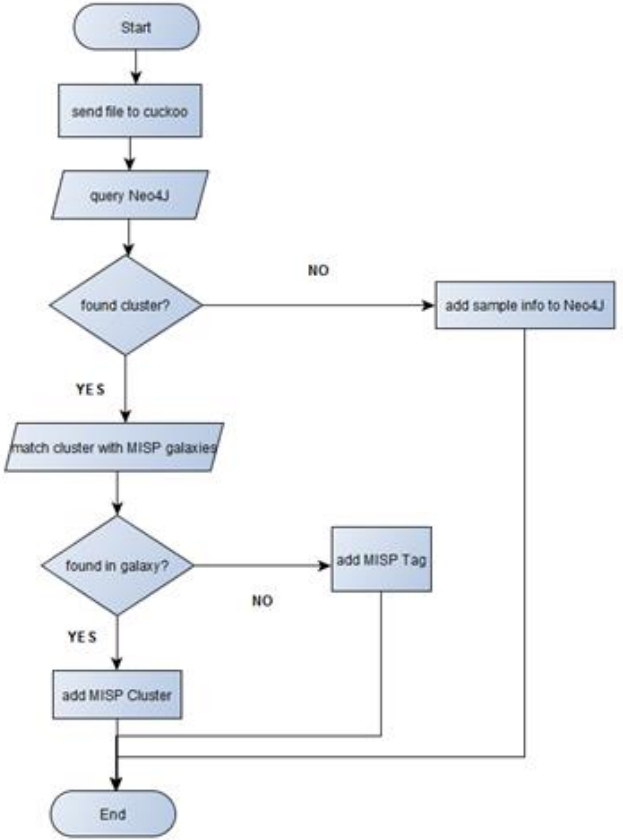
2. Generate a graph (network) based on the similarity

3. Conduct network analysis over the graph (clustering)

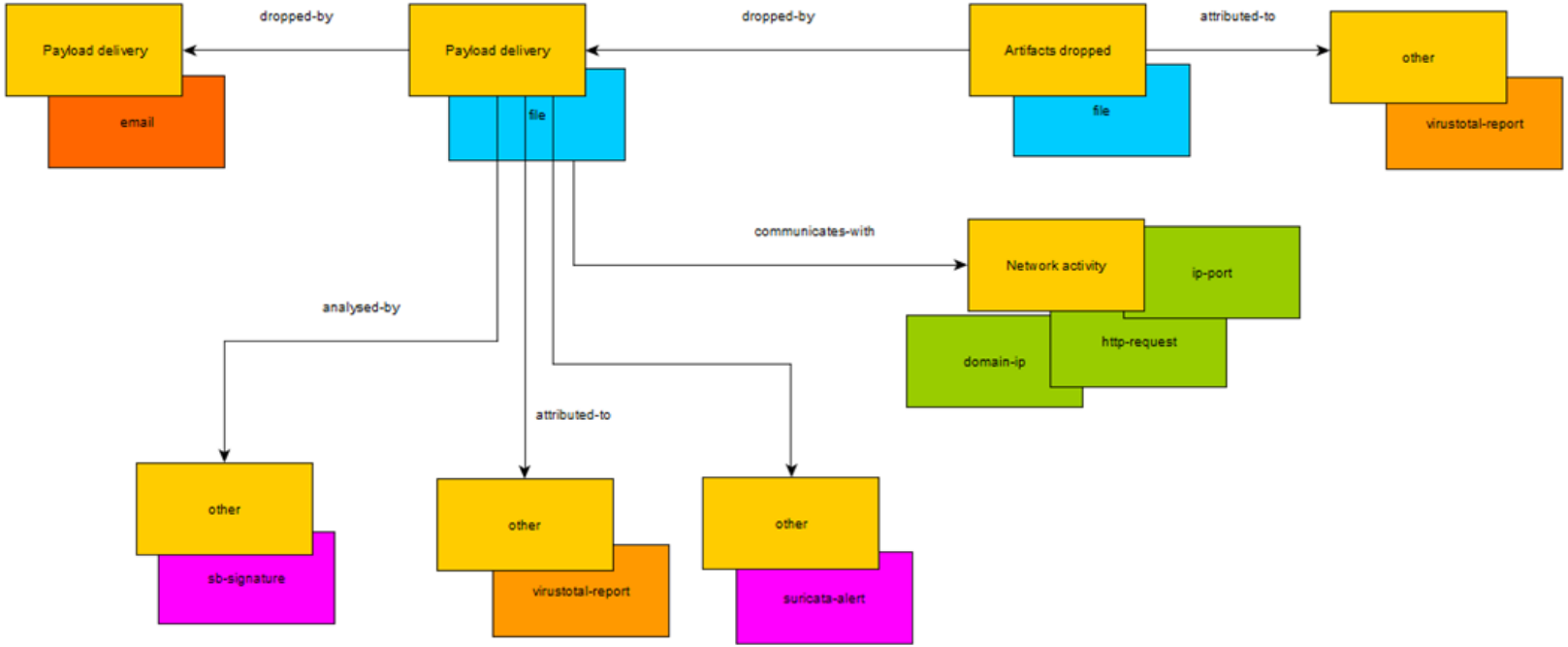
4. Register and visualize the clustering results on Neo4j

Connect all the things





Cuckoo to MISP

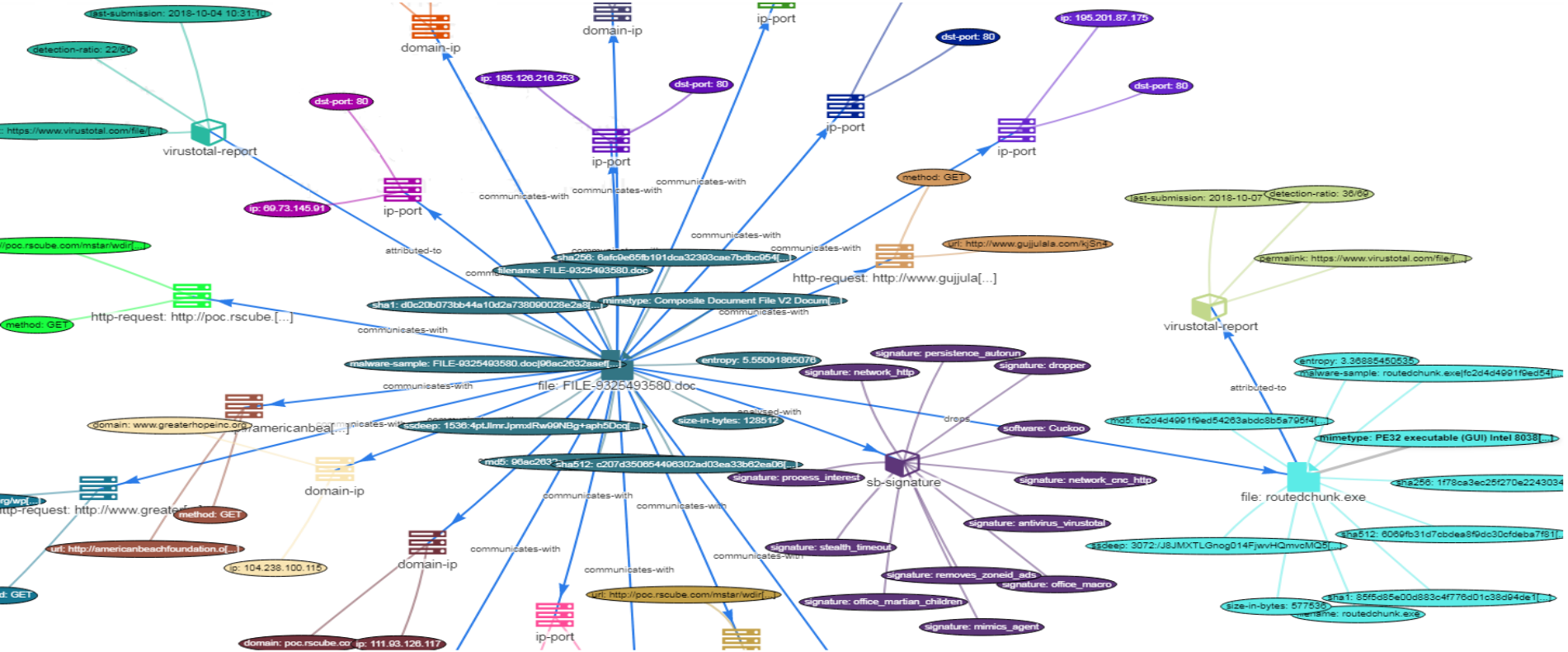


Cuckoo

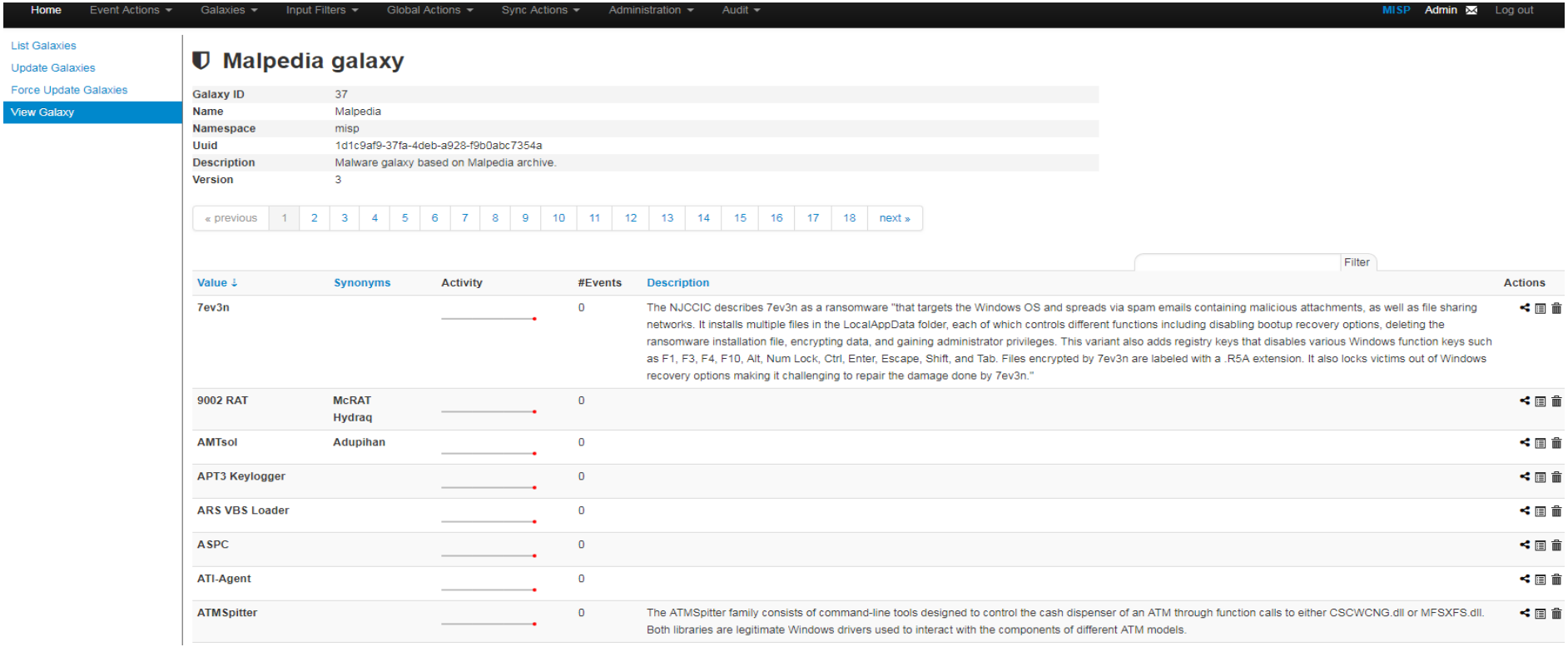
Dashboard | Recent | VT Stats | Pending | Search | API | Submit

ID	Timestamp	Filename	Target	PKG	Martians	Suricata Alerts/HTTP /TLS/Files	VT	MalScore	MalFamily	PCAP	ClamAV	Custom	Status
79190	2018-10-08 10:25:59	f667680df596631fba58754c16c3041fae12ed6bf25d6068e6981ee68a6c9d0a_unpacked	137eb9b6ef122857bde72f78962ed208	dll	None	0/0/0/0	49/68	10.0	Turla	PCAP	None	None	reported
79187	2018-10-08 10:24:22	b360a27cc842a453a450ec9c4a1993273a3d1946fd75ae979be9df422f754fcb	2b816a8e80a69a018b8be8eb98ffe4e1	exe	None	0/0/0/0	23/54	10.0	Sarhust	PCAP	None	None	reported
79189	2018-10-08 10:24:22	edb1ff2521fb4bf748111f92786d260d40407a2e8463dcd24bb09f908ee13eb9	cfdd16225e67471f5ef54cab9b3a5558	None	None	0/0/0/0	56/69	10.0	Olympicdestroyer	PCAP	Win.Trojan.OlympicDestroyer-6446992-0	None	reported
79188	2018-10-08 10:22:50	d685f21ae0ffbcf002939500e8c1b6a8d37f18c1c33eca37f4a5628c577dc9ef	c8201ed20fbe24f777ea70258102a7cb	None	None	0/0/0/0	52/68	10.0	None	PCAP	None	None	reported
79182	2018-10-08 10:00:08	fb9e181d3ea6faa9d0e7431bfc8301fd66bcc8c3d66b26cef7036d117ee5fbb1	875f3fc948c6534804a26176dcfb6af0	None	None	0/0/0/0	42/64	10.0	Agent	PCAP	Win.Trojan.Agent-419088	None	reported
79179	2018-10-08 09:58:05	ff808d0a12676bfac88fd26f955154f8884f2bb7c534b9936510fd6296c543e8	36524c90ca1fac2102e7653dfadb31b2	exe	None	0/0/0/0	53/68	10.0	Sofacy	PCAP	None	None	reported
79181	2018-10-08 09:58:58	22b785c8713abbe7ae13c5019999ae1ee4d163af64f0a8d6acb326d0812383ec	574322a51aee572f60f2d87722d75056	dll	None	0/0/0/0	40/63	10.0	None	PCAP	None	None	reported

To MISP



MISP: New Malpedia Galaxy



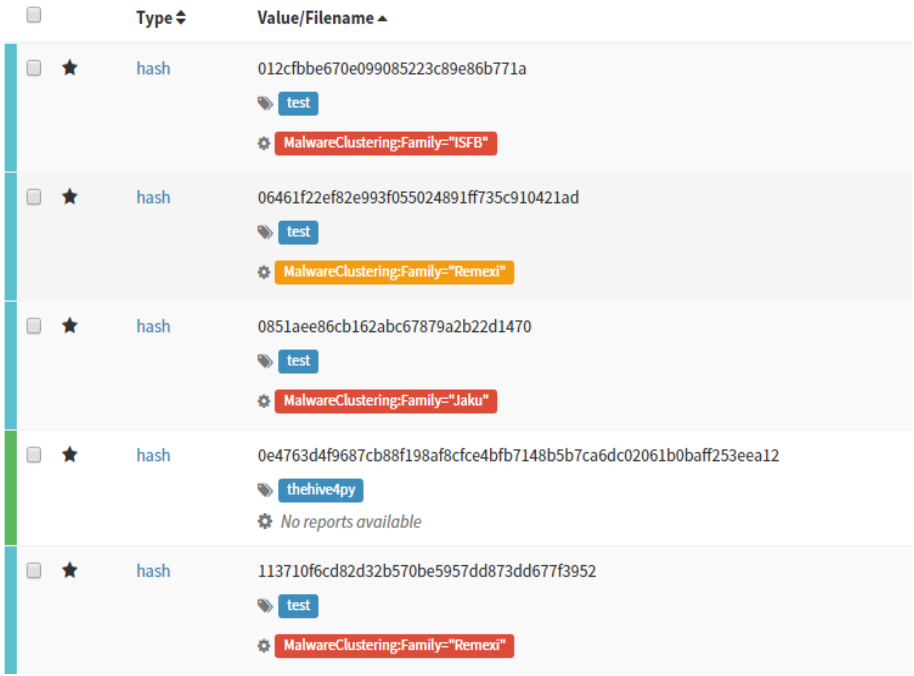
Cuckoo to MISP: final result

Add Cluster if family is found in MISP Galaxies, Add Tag if family is not found

My Events | Org Events

Published	Org	Owner Org	Id	Clusters	Tags	#Attr.	Email	Date	Info
✖	LDO-CERT (Heli-Div)	LDO-CERT (Heli-Div)	10393	Malpedia: Seduploader	AW_cuckoo, tlp:red, Sofacy	49	admin@admin.test	2018-10-08	locs from cuckoo analysis: 79179
✖	LDO-CERT (Heli-Div)	LDO-CERT (Heli-Div)	10395	Malpedia: gsecdump	AW_cuckoo, tlp:red, Agent	16	admin@admin.test	2018-10-08	locs from cuckoo analysis: 79182
✖	LDO-CERT (Heli-Div)	LDO-CERT (Heli-Div)	10397	Malpedia: Olympic Destroyer	AW_cuckoo, tlp:red	19	admin@admin.test	2018-10-08	locs from cuckoo analysis: 79189
✖	LDO-CERT (Heli-Div)	LDO-CERT (Heli-Div)	10399		AW_cuckoo, tlp:red, cluster: Mosquito, Turla	19	admin@admin.test	2018-10-08	locs from cuckoo analysis: 79190
✖	LDO-CERT (Heli-Div)	LDO-CERT (Heli-Div)	10398		AW_cuckoo, tlp:red, Sarhust	16	admin@admin.test	2018-10-08	locs from cuckoo analysis: 79187
✖	LDO-CERT (Heli-Div)	LDO-CERT (Heli-Div)	10396		AW_cuckoo, tlp:red, cluster: XBTL	19	admin@admin.test	2018-10-08	locs from cuckoo analysis: 79188

Analyzer for The Hive



If the observable is an hash, the analyzer queries the Neo4j and retrieves the family and hashes contained in the matched family.

If the observable is a file, the analyzer calculates impfuzzy, adds it to Neo4j (if not present), calculates the cluster and retrieves the family and hashes contained in the matched family.

Analyzer for The Hive: long report

Report for MalwareClustering_Search_1_0 analysis of Mon, Oct 8th, 2018 11:02 +02:00

Show Raw Report | Show observables (65)

File Summary

Name	bf0b5589692b48b0c5ecc0141b740a23639ceed539ad67a775ddaf9e722bf76f_unpacked
Family	Kronos
SHA-256	c9fe44239070bb2a048ea3c044471112f340049cacb87522de5adbc74d687bb2
SHA-1	61b2c80c8c8dd4935fa7cbfdcbecccd13b38bf12
MD5	55c2663cfb4eb03f1e7d298fe4504d1a
Impfuzzy	48:STMjAteTHEjZR90ycqFFaeKxOKY9fLS7HqhG4s7xXpjuHV9lQHBv2UsNjmTEsXIA:WMjCHZ8ycqFFHKxtY9fLSPX0ov23mYG
ApiScout Vector	A27oA3gA3EIgA11QA8QAAKA11EA3gA6QAgA10gFABIACAUCIAIIA4wAABQhIATEBQCARGIADFbE+AEAABHInBwOgYFpjDCR*WogW
ApiScout Confidence	87.3704056661038

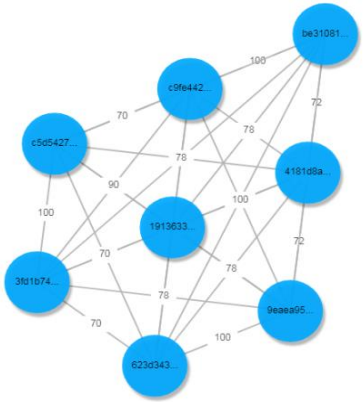
Similar Families

Similar Families for ApiScout matchVectors

Family	Score
Kronos	100

Similar Sha256

Family	Sha256	Score
Kronos	623d34383cb000b94f48c90c100da8c107ca5149598dba46de4e2801fdb6815f	100
Kronos	9eaea954ae1ae48f78cf94ed37acb0f933391a090739a6023e9a7aaca11d9294	100
Kronos	be31081e8a3673b2edbc855c710372c75e1014c0db57b901b24fa35c8e7ce20e	100
Kronos	19136335f67801a0f6d2c10854568fcbc03308a28691d6a454b33de7629eda0f	78



Next level of Clustering: ApiScout & ApiVector

For some family impfuzzy has low success rate: let’s try ApiVector!

```
~/presentations$ python /home/analyst/apiscout/scout.py -t cobral
Using base adress 0x0 to infer reference counts.
Parsing Import Table for
  cobral.
Results for API DB: import_table
idx: offset    ; VA              ; IT?; #ref;DLL                          ; API
  1: 0x0001c000;          0x00000000; yes;    2; advapi32.dll_0x0 (32bit)              ; SetFileSecurityA
  2: 0x0001c004;          0x00000000; yes;    2; advapi32.dll_0x0 (32bit)              ; MakeAbsoluteSD
  3: 0x0001c008;          0x00000000; yes;    4; advapi32.dll_0x0 (32bit)              ; SetTokenInformation
  4: 0x0001c00c;          0x00000000; yes;    2; advapi32.dll_0x0 (32bit)              ; ConvertStringSidToSidA
  5: 0x0001c010;          0x00000000; yes;    2; advapi32.dll_0x0 (32bit)              ; CryptGenRandom
  6: 0x0001c014;          0x00000000; yes;    1; advapi32.dll_0x0 (32bit)              ; CryptAcquireContextA
  7: 0x0001c018;          0x00000000; yes;    3; advapi32.dll_0x0 (32bit)              ; ConvertStringSecurityDescriptorToSecurityDescriptorA
  8: 0x0001c01c;          0x00000000; yes;    4; advapi32.dll_0x0 (32bit)              ; CreateProcessAsUserA
  9: 0x0001c020;          0x00000000; yes;   10; advapi32.dll_0x0 (32bit)              ; OpenProcessToken
 10: 0x0001c024;          0x00000000; yes;    2; advapi32.dll_0x0 (32bit)              ; ConvertSidToStringSidA
 11: 0x0001c028;          0x00000000; yes;    3; advapi32.dll_0x0 (32bit)              ; GetTokenInformation
 12: 0x0001c02c;          0x00000000; yes;    5; advapi32.dll_0x0 (32bit)              ; RegSetValueExA
 13: 0x0001c030;          0x00000000; yes;    2; advapi32.dll_0x0 (32bit)              ; RegCreateKeyExW
 14: 0x0001c034;          0x00000000; yes;    3; advapi32.dll_0x0 (32bit)              ; RegQueryValueExA
 15: 0x0001c038;          0x00000000; yes;    5; advapi32.dll_0x0 (32bit)              ; RegQueryValueExW
 16: 0x0001c03c;          0x00000000; yes;    2; advapi32.dll_0x0 (32bit)              ; LookupPrivilegeValueA
 17: 0x0001c040;          0x00000000; yes;    5; advapi32.dll_0x0 (32bit)              ; RegOpenKeyExA
 18: 0x0001c044;          0x00000000; yes;    6; advapi32.dll_0x0 (32bit)              ; DuplicateTokenEx
 19: 0x0001c048;          0x00000000; yes;    2; advapi32.dll_0x0 (32bit)              ; AdjustTokenPrivileges
 20: 0x0001c04c;          0x00000000; yes;    3; advapi32.dll_0x0 (32bit)              ; LogonUserA
 21: 0x0001c050;          0x00000000; yes;    6; advapi32.dll_0x0 (32bit)              ; RegCloseKey
 22: 0x0001c054;          0x00000000; yes;    4; advapi32.dll_0x0 (32bit)              ; RegSetValueExW
 23: 0x0001c058;          0x00000000; yes;    2; advapi32.dll_0x0 (32bit)              ; ImpersonateNamedPipeClient
199: 0x0001c330;          0x00000000; yes;    2; ntdll.dll_0x0 (32bit)                 ; ZwCreateSection
200: 0x0001c334;          0x00000000; yes;    5; ntdll.dll_0x0 (32bit)                 ; ZwFreeVirtualMemory
201: 0x0001c338;          0x00000000; yes;    2; ntdll.dll_0x0 (32bit)                 ; ZwUnmapViewOfSection
202: 0x0001c33c;          0x00000000; yes;    3; ntdll.dll_0x0 (32bit)                 ; ZwCreateFile
203: 0x0001c340;          0x00000000; yes;    2; ntdll.dll_0x0 (32bit)                 ; ZwWaitForSingleObject
204: 0x0001c344;          0x00000000; yes;    2; ntdll.dll_0x0 (32bit)                 ; ZwReadVirtualMemory
205: 0x0001c348;          0x00000000; yes;    2; ntdll.dll_0x0 (32bit)                 ; ZwOpenThread
206: 0x0001c34c;          0x00000000; yes;    1; ntdll.dll_0x0 (32bit)                 ; RtlInitUnicodeString
207: 0x0001c350;          0x00000000; yes;    2; ntdll.dll_0x0 (32bit)                 ; ZwMapViewOfSection
208: 0x0001c354;          0x00000000; yes;    2; ntdll.dll_0x0 (32bit)                 ; ZwTerminateThread
209: 0x0001c358;          0x00000000; yes;    6; ntdll.dll_0x0 (32bit)                 ; ZwQueryInformationProcess
210: 0x0001c35c;          0x00000000; yes;   17; ntdll.dll_0x0 (32bit)                 ; ZwClose
211: 0x0001c360;          0x00000000; yes;    2; ntdll.dll_0x0 (32bit)                 ; RtlNtStatusToDosError
212: 0x0001c364;          0x00000000; yes;    2; ntdll.dll_0x0 (32bit)                 ; ZwQuerySystemInformation
213: 0x0001c368;          0x00000000; yes;    2; ntdll.dll_0x0 (32bit)                 ; RtlQueryRegistryValues
DLLs: 7, APIs: 213, references: 1427

WinApi1024 Vector Results:
import_table: 149 / 208 (71.63%) APIs covered in WinApi1024 vector.
    Vector:     A231A13CA7CMA8QgMABAABIA4EIkIIAFAAQkQDgOAQFAQgAAIAQAAJACAKDIAkEgAQDAARAAwOBgBACoAGAogiMIUDACiBUAAHAkDBahUwwaE
    Confidence: 91.2258058392
```

ApiScout is a library that allows the recovery of potentially used Windows API functions from memory dumps.

ApiVectors are a compact representation of “interesting” API functions extracted with ApiScout that can be used to get a first impression of a malware's potential capabilities but may also serve for matching against a reference database to aid in malware identification.

Okay, but how can I use this?

You can easily incorporate ApiVectors into your own analysis environment.
For starters, the previous blog post on ApiScout explains how to build a DB custom to your Windows system. After this, you can simply crawl arbitrary buffers (e.g. memory dumps of selected suspicious segments from processes) for their API information and have this available in your other analysis such as IDA Pro.

If you do not want to use ApiScout to crawl memory dumps, you can also create ApiVectors directly from a given list of Windows API functions (e.g. Import Tables) using getApiVectorFromApiList() and getApiVectorFromApiDict() from the ApiVector class respectively.

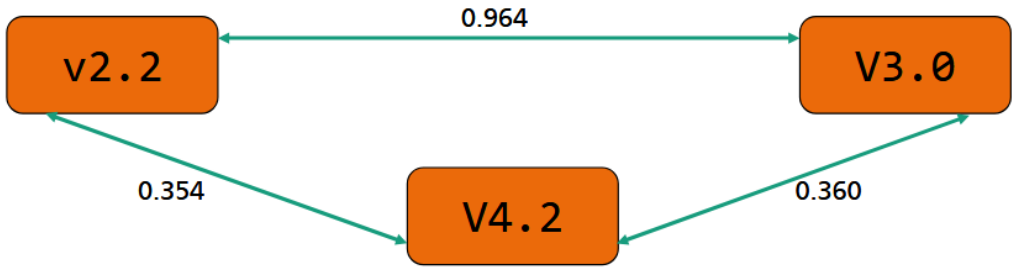
A concrete use case for ApiVectors is matching them against each other.
In that context, projects like ImpHash and ImpFuzzy may come to mind.
The advantage of ApiVectors is that they actually carry the identity of API functions used without abstracting them with only little higher cost in terms of required storage. We are currently looking to hook our approach up with sandboxing, e.g. Cuckoo.

Comparison of ApiVectors

- Example Vectors
  - Base64-like encoding (Run-Length compressed) - 4-172 bytes long

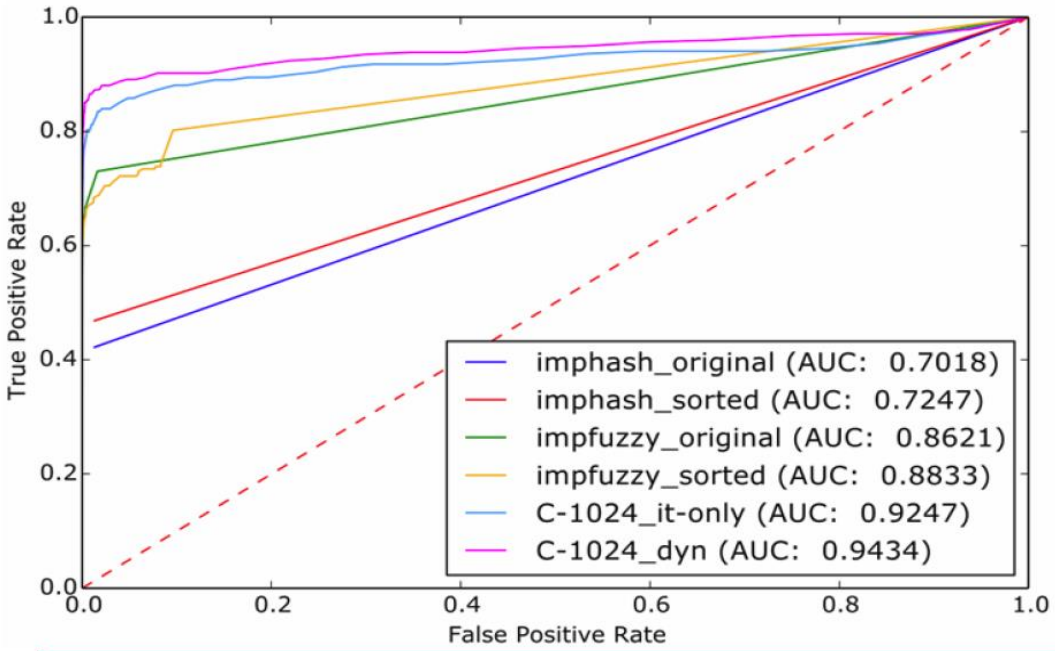
```
A42gA28KA13    CAAMA16BABAAJAECAxMAACkAAQUA7CJBCgAgUBA3 kQCBAHJSRjU^q-*}_pb__N,__^?
A42gA28KA13    CAAMA16BABAAJAEAAxMAACkAAQUA7CJBCgAAUBA3 kQCBAHJSRjU^q-*}_pL__N,._^?
A41BA29CA4IA9gCA9gA8Q  BAAJAEAABMA3 gAAQA8 QJRCgAgUBAAHkQARCDIADDBGAqQAgCcGOIOp,f?
```

TesliaCrypt 2.2, 3.0, 4.2



Fraunhofer FKIE

ApiVector matching performances



- Data set: Malpedia (2018-05-17)
  - 673 families, 1854 samples
- Comparison with ImpHash, ImpFuzzy
  - Mean Fingerprint sizes:
    - ImpHash: 32 bytes
    - ImpFuzzy: 54.4 bytes
    - ApiVector: 74.3 bytes
  - ApiVector: recoverable info
- Performance @ Thresholds
  - T: 0.18 – 90.18% TPR, 9.45% FPR
  - T: 0.22 – 89.10% TPR, 4.74% FPR
  - T: 0.32 – 86.55% TPR, 0.99% FPR
  - T: 0.55 – 80.72% TPR, 0.09% FPR

24

Fraunhofer FKIE

Compare 2 Cobra samples

welivesecurity BY eset

Carbon Paper: Peering into Turla's second stage backdoor

Carbon footprint

Table 2 – Carbon sample hashes

SHA1 hash
7f3a60613a3bdb5f1f8616e6ca469d3b78b1b45b
a08b8371ead1919500a4759c2f46553620d5a9d9
4636dccac5acf1d95a474747bb7bcd9b1a506cc3
cbde204e7641830017bb84b89223131b2126bc46
311f399c299741e80db8bec65bbf4b56109eedaf
fbc43636e3c9378162f3b9712cb6d87bd48ddbd3
554f59c1578f4ee77dbba6a23507401359a59f23
2227fd6fc9d669a9b66c59593533750477669557
87d718f2d6e46c53490c6a22de399c13f05336f0
1b233af41106d7915f6fa6fd1448b7f070b47eb3

Compilation date	Orchestrator version	Injected library version
2014-02-26	3.71	3.62
2016-02-02	3.77	4.00
2016-03-17	3.79	4.01
2016-03-24	3.79	4.01
	3.79	
	3.81	
	3.81	
	3.81	

Table 1 – Car

DLL

44 / 64

44 engines detected this file

SHA-256	7fa4482bfbca550ce296d8e791
File name	MSIMGHLP.DLL
File size	145.5 KB
Last analysis	2018-09-06 06:43:11 UTC
Community score	-1

Detection | Details | Community 2

Ad-Aware	Trojan.GenericKD.4298065
AhnLab-V3	Trojan/Win32.Turla.C1896683
Antiy-AVL	Trojan[Backd
Avast	FileRepMalwa
Avira	TR/AD.Turla.daypm
BitDefender	Trojan.GenericKD.4298065
Cylance	Unsafe

```
analyst        :~/presentation$ shalsum cobra1 cobra2
a08b8371ead1919500a4759c2f46553620d5a9d9  cobra1
554f59c1578f4ee77dbba6a23507401359a59f23  cobra2
```

DLL

47 / 65

47 engines detected this file

SHA-256	d1ad698567b04ea5ce8197c0316
File name	MSIMGHLP.DLL
File size	275.5 KB
Last analysis	2018-09-06 08:43:44 UTC
Community score	-1

Detection | Details | Community 3

Ad-Aware	Trojan.Generic.20741530
AhnLab-V3	Malware/Win64.Generic.C1358717
	Trojan[Backdoor]/Win32.Turla
	Win64:Malware-gen
Avira	BDS/Turla.cdkig
BitDefender	Trojan.Generic.20741530
CrowdStrike Falcon	malicious_confidence_60% (D)
Cyren	W64/Trojan.IXHK-1423

Compare with ApiVector

```
python mal_compare.py cobra1 cobra2
Ssdeep1:
3072:dSjh5NkoeU+KmSrNja7l5jIJeUd9f3bs/uIeUx:dQR
eUm57l9IcUd9Uu
Ssdeep2:
6144:N0aBhxDgpfGxzudQwDYm6WSN4Tq+9UjT:77Dg
RGxzMQwDxY43
Ssdeep Compare: 0
ImpFuzzy1:
96:cA8yytCJq5gugxLRz0I3qYCJWMoZpcf+Po4pzdStK3
BK:cAh05ULRz0IbC2rSQQ
ImpFuzzy2:
96:k5JFIEZeVeXxLdaM+fgclikn4xtS23K5bnvaqch2sz:kxheVC
jan4xMX59cgsz
Impfuzzy Compare: 0
```

```
python scout.py -t cobra1
VectorA:
A23IA13CA7CMA8QgMABAABIA4ElkIIAFAAQkQDgOAQFA
QgAAIAQAAJACAKDIAkEgAQDAARAAwDBgBACoAGAogi
MIUDACiBUAAHAkDBahUwwaEBHJMDuGggCYGGINxW?
python scout.py -t cobra2
VectorB:
A23IA3+A9CA3BA3CA9QgMABAABIA4EIEIIAFAAQkADAO
AQFgQgAAIA4JgCAKDIAkEA6QAAwBBgAACoAEAoACMA
QCACiBQAAHAgDBShUwwaEBHJMDuEAgCYGGINxWW?
python match.py $Vector1 –v Vector2
Result of matching vectors:
Vector A: A23IA13CA7CMA8Q……
Vector B: A23IA3+A9CA3BA3C……..
Score: 73.0656996953
```

Next Steps

Microsoft Visual C++	388
Armadillo v1.71	386
Microsoft Visual C++ v5.0/v6.0 (MFC)	386
Microsoft Visual C# / Basic .NET	187
Microsoft Visual C++ v6.0 DLL	180
.NET executable	177
Armadillo v1.xx - v2.xx	164
Microsoft Visual Basic v5.0/v6.0	82
Microsoft Visual Basic v5.0	46
BobSoft Mini Delphi -> BoB / BobSoft	36
UPX v0.89.6 - v1.02 / v1.05 -v1.24 -> Markus & Laszlo [overlay]	34
Microsoft Visual C++ v7.1 EXE	30
UPX 2.90 [LZMA] -> Markus Oberhumer, Laszlo Molnar & John Reiser	28
Microsoft Visual C++ vx.x DLL	26
Microsoft Visual C++ 8.0 [Debug]	22
Microsoft Visual C++ v7.0 DLL	16
ASPack v2.12	14
Microsoft Visual C++ v7.0	10
InstallShield 2000	9
ASProtect V2.X DLL -> Alexey Solodovnikov	7
ASProtect v1.23 RC1	6
ASProtect 1.33 - 2.1 Registered -> Alexey Solodovnikov	6
UPX v0.89.6 - v1.02 / v1.05 -v1.22 (Delphi) stub	6
ASProtect v1.2x (New Strain)	6
PeCompact 2.xx --> BitSum Technologies	5
PECompact V2.X-> Bitsum Technologies	5

Improve identification of packed PE32 (now with peid), try to unpack it automatically, calculate apivector over the upacked sample and store hashes about both files.

Did some test with entropy:

1) win.bolek_packed

PE: compiler: Microsoft Visual C/C++(2003)[-]

Entropy: 7.57267

2) win.bolek_unpacked

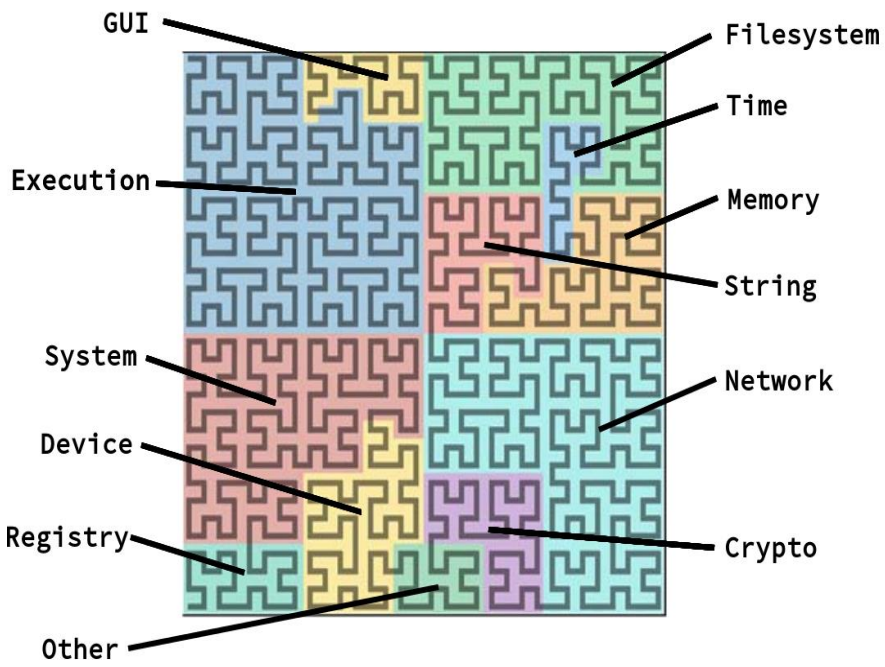
PE: compiler: EP:Microsoft Visual C/C++(-)[looks like patched]

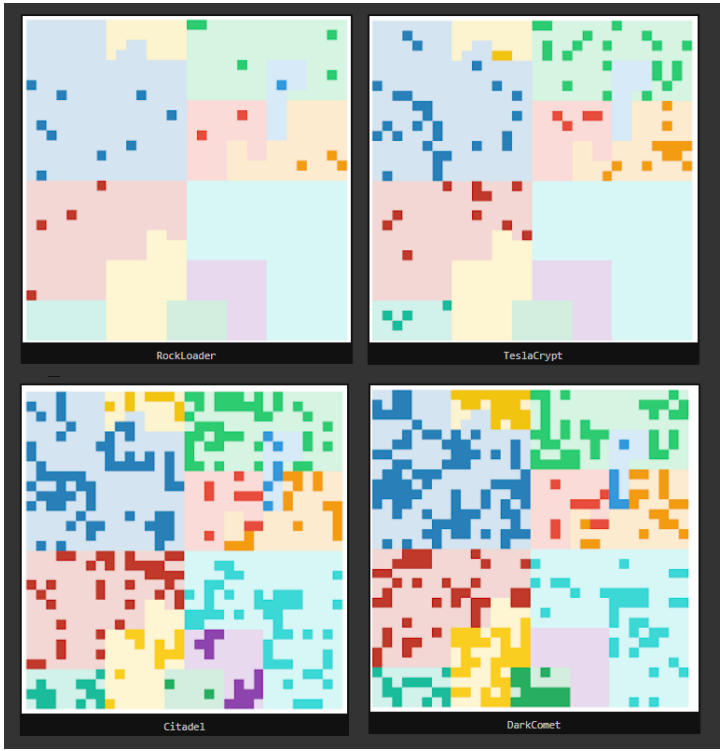
Entropy: 6.72804

But result often is not so good.

Next Steps

Integrate ApiQR representation from ByteAtlas: Hilbert curve for 1024 bit ApiVector with the semantic categories.





Places to be

- https://github.com/LDO-CERT
- https://github.com/MISP/misp
- https://github.com/spender-sandbox/cuckoo-modified
- https://github.com/TheHive-Project
- https://github.com/JPCERTCC/impfuzzy
- https://github.com/danielplohmann/apiscout
- https://github.com/TheHive-Project
- https://github.com/antvis/g6
- https://malpedia.caad.fkie.fraunhofer.de
- https://byte-atlas.blogspot.com

Special Thanks

[1] http://www.leonardocompany.com/en/cert

[2] https://www.linkedin.com/in/davidearcuri

[3] https://www.linkedin.com/in/rita-ottolini-0149b214

Q&A

THANK **YOU** FOR YOUR ATTENTION