Automated Generation of YARA Classifier for Malware

Using Code Similarity

Arun Lakhotia
University of Louisiana at Lafayette
Cythereal, Inc

Presented at ACSAC 2022, December 2022

About Me

Professor of Computer Science

CTO, co-Founder

Automated malware deobfuscation and indexing
Automated YARA generation

PSA - Invitation to share your knowledge for posterity

acm Association for Computing Machinery

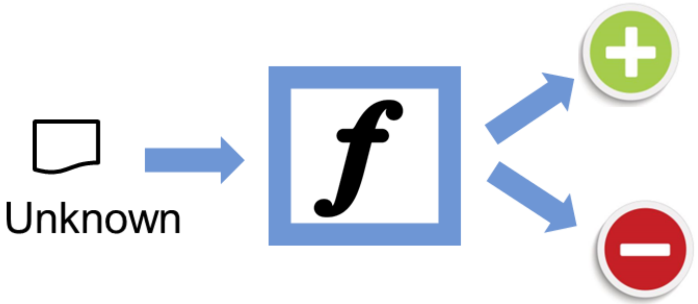
Digital Threats: Research and Practice

Promoting Science in Digital Threats Research

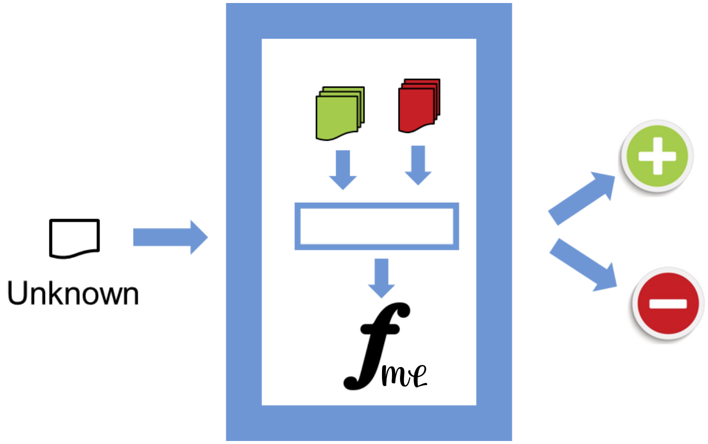
Field Notes: Capture knowledge of practitioners

- A short case report (1000-1500 words) about emerging threats and defenses.
- Accurately document factual data as well as the settings, actions, behaviors, and consequences that are observed.
- May contain thoughts, ideas, questions, and concerns that arise as the observation is conducted.
- Provide perspectives on a single phenomenon that, when accumulated over time, suggest new avenues of research.

Classifiers

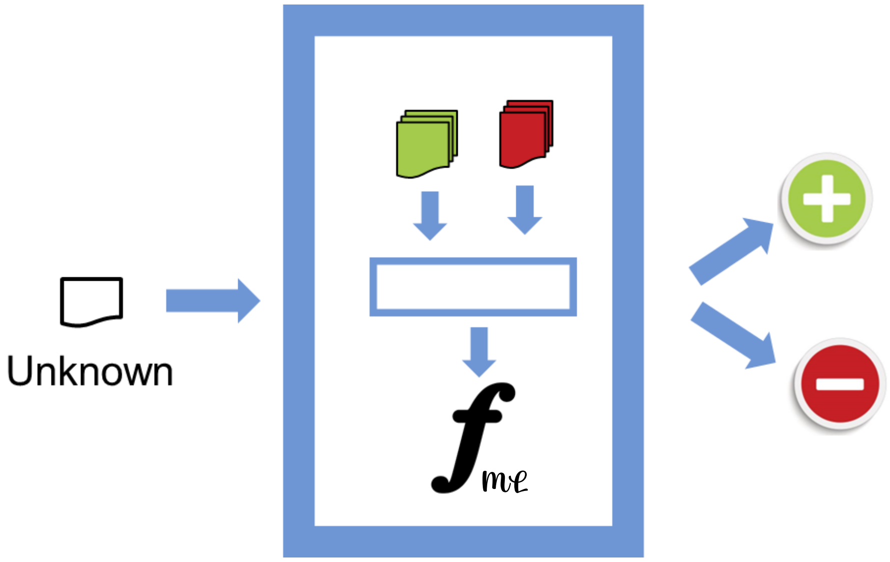


ML Classifiers



- Requirements for ML Classifiers:
  - Distribution of +ves and –ves is identical and independent
  - Availability of +ves and –ves samples
  - Available samples represent population

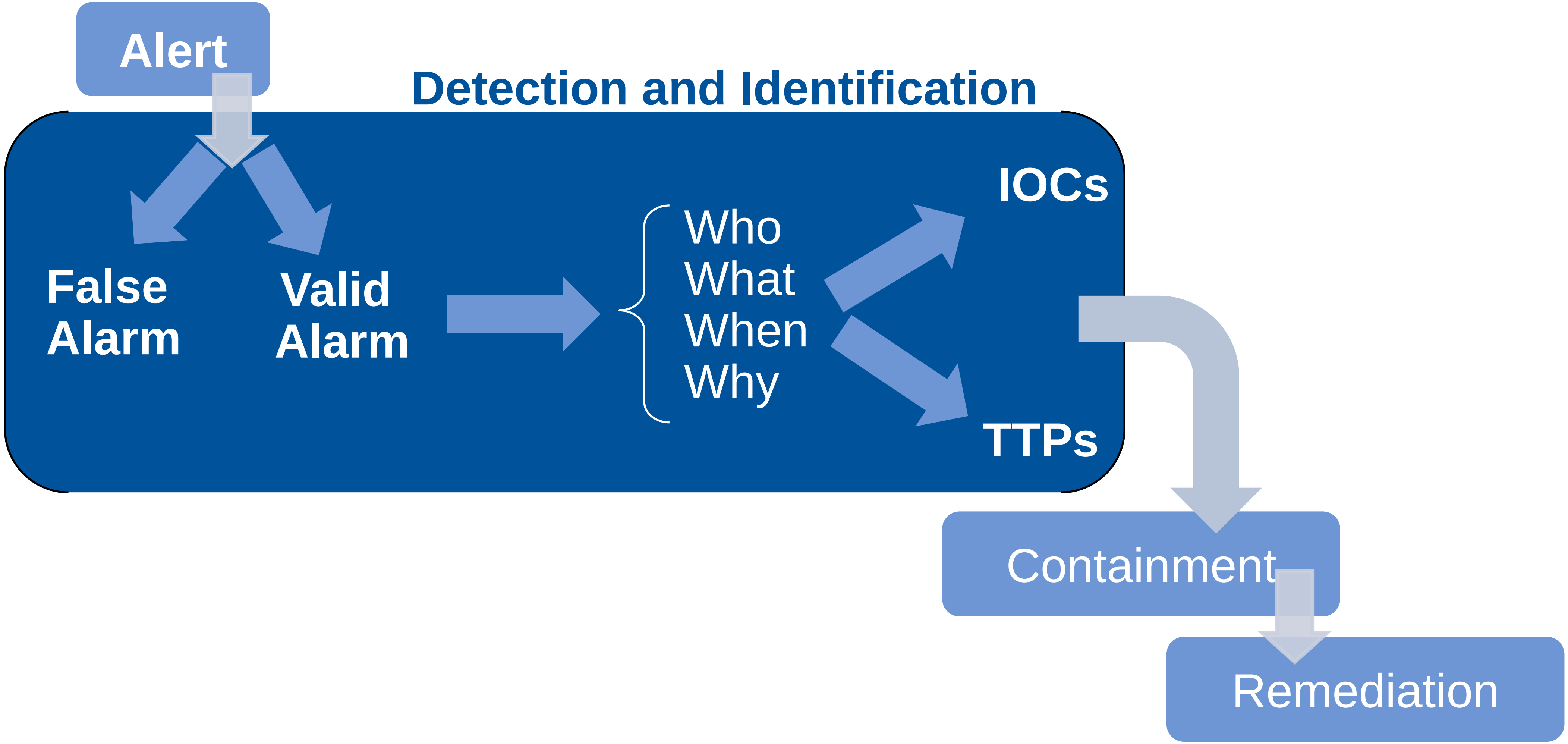
ML Classifiers for Malware



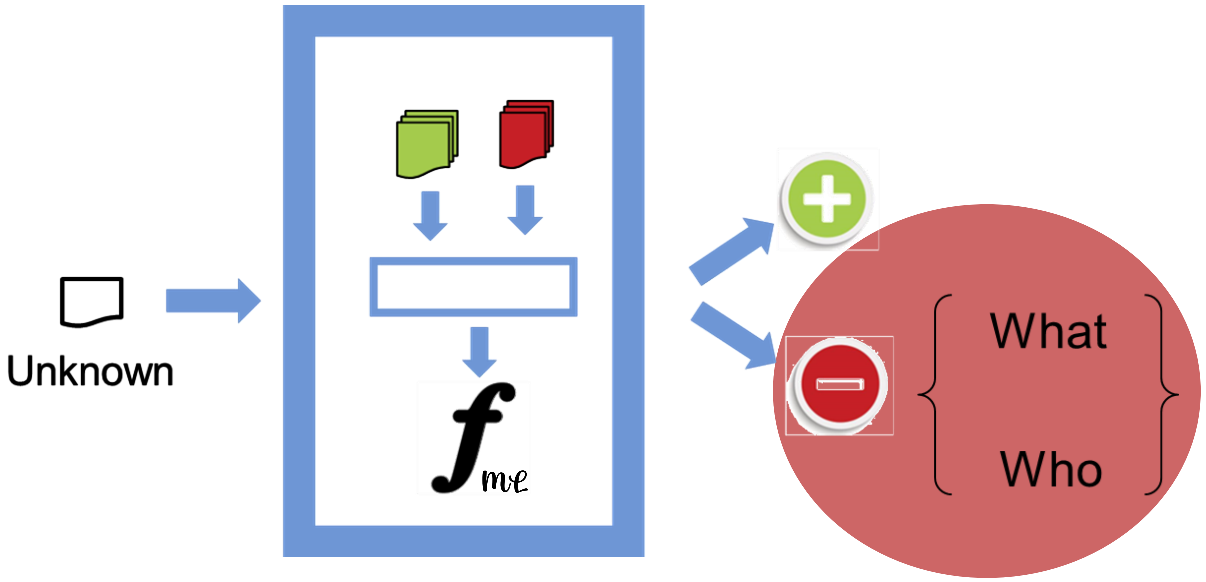
Practical Challenges

- Concept drift
  - – +ves and –ves are not static
- Adversarial
  - – +ves transform to defeat classifiers
- Labels are noisy
  - – Crowdsourced
  - – Industry consensus

New Use Case: Incident Response Workflow



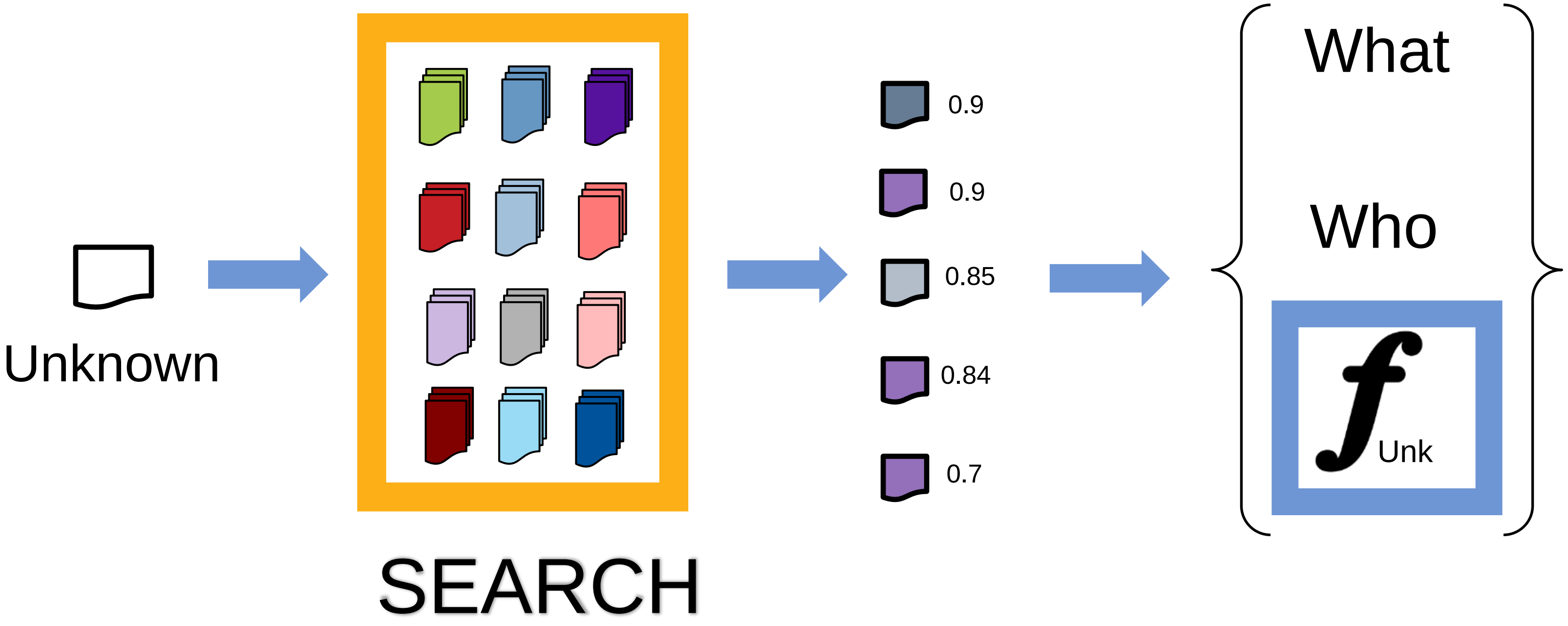
ML Classifiers for Malware



New Challenges

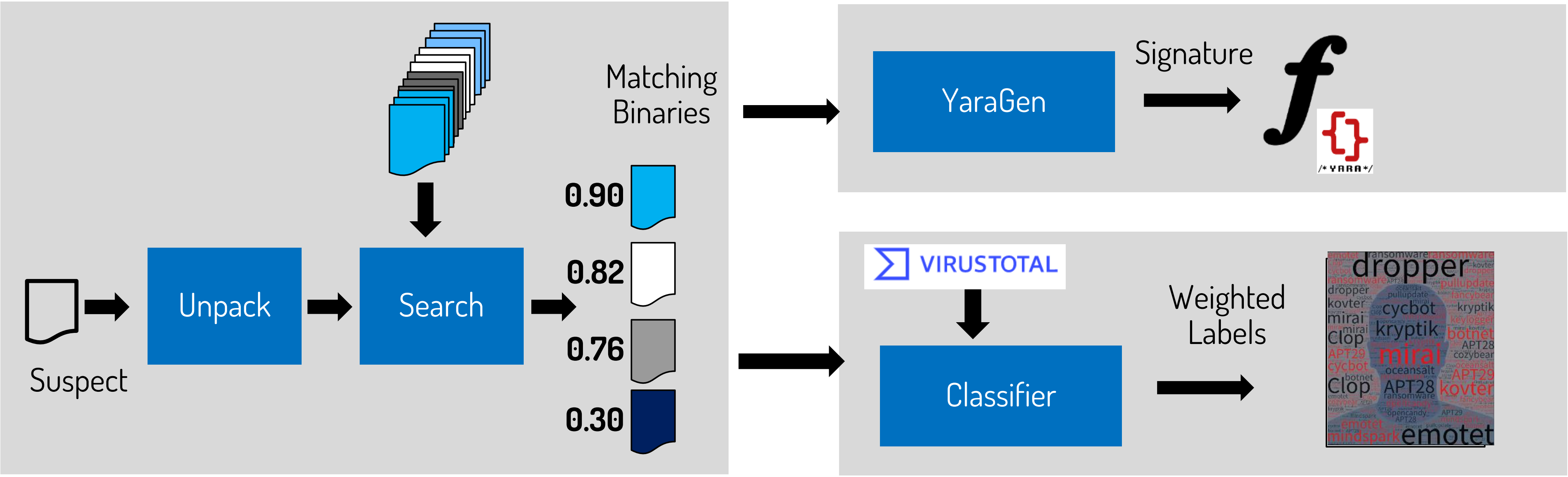
- Knowing a suspect is malware is not sufficient
  - Need to know the stage of the attack
    - What the malware does?
    - Who is behind it?
- Need multi-classifiers
  - Type of malware
  - Family of malware

A New Architecture for Generating Malware Classifiers

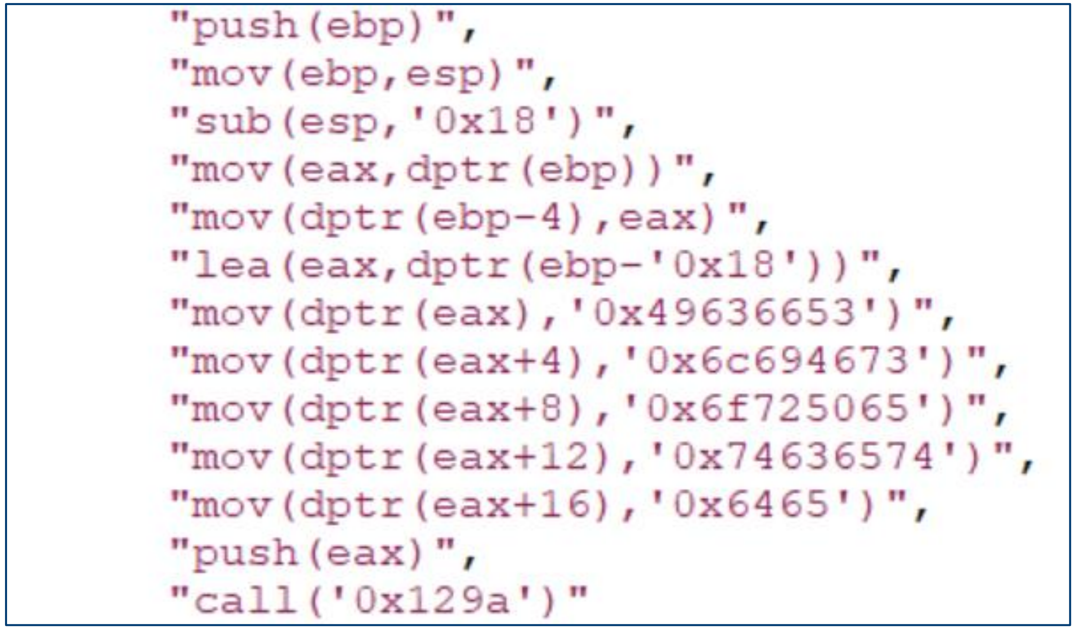


- Match Unknown to Known
- Extract characteristics from Known
- Create specialized classifier for the Unknown

Cythereal MAGIC – Malware Genomic Classifier



How to Search for Similar Binaries?



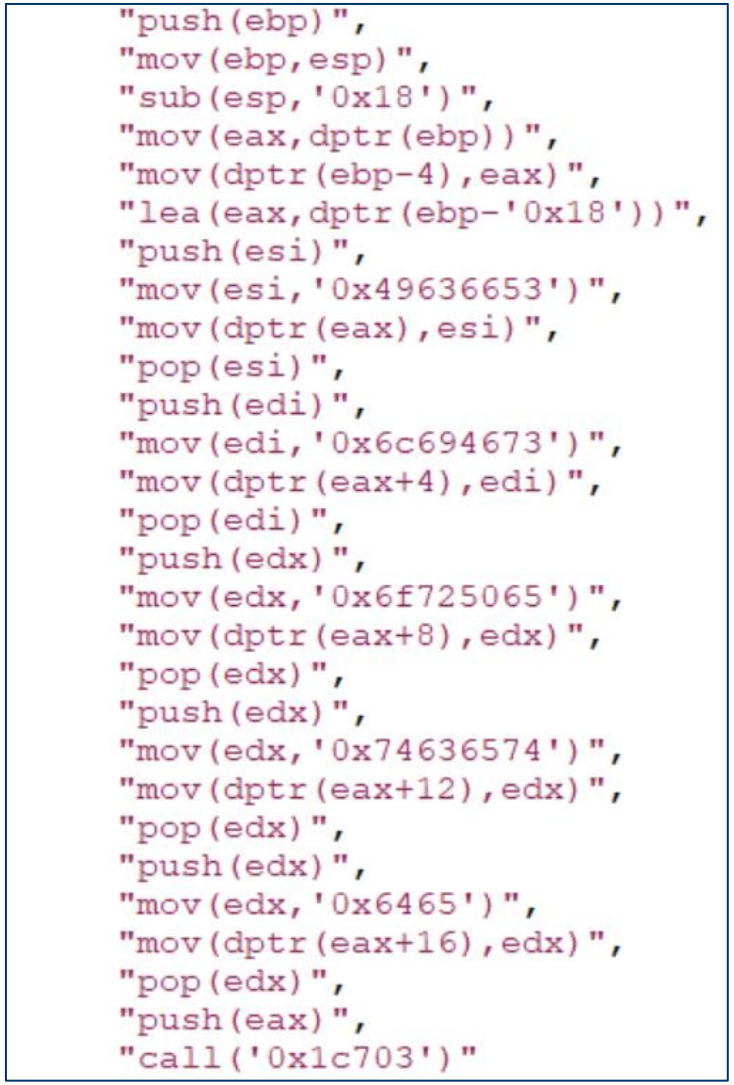
```
"push(ebp)",
"mov(ebp,esp)",
"sub(esp,'0x18')",
"mov(eax,dptr(ebp))",
"mov(dptr(ebp-4),eax)",
"lea(eax,dptr(ebp-'0x18'))",
"mov(dptr(eax),'0x49636653')",
"mov(dptr(eax+4),'0x6c694673')",
"mov(dptr(eax+8),'0x6f725065')",
"mov(dptr(eax+12),'0x74636574')",
"mov(dptr(eax+16),'0x6465')",
"push(eax)",
"call('0x129a')"
```

Bytes

≠

Semantics

=

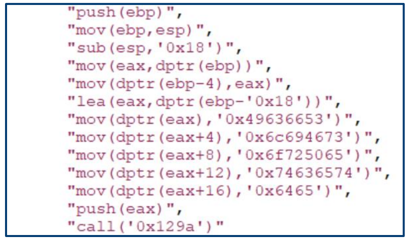


```
"push(ebp)",
"mov(ebp,esp)",
"sub(esp,'0x18')",
"mov(eax,dptr(ebp))",
"mov(dptr(ebp-4),eax)",
"lea(eax,dptr(ebp-'0x18'))",
"push(esi)",
"mov(esi,'0x49636653')",
"mov(dptr(eax),esi)",
"pop(esi)",
"push(edi)",
"mov(edi,'0x6c694673')",
"mov(dptr(eax+4),edi)",
"pop(edi)",
"push(edx)",
"mov(edx,'0x6f725065')",
"mov(dptr(eax+8),edx)",
"pop(edx)",
"push(edx)",
"mov(edx,'0x74636574')",
"mov(dptr(eax+12),edx)",
"pop(edx)",
"push(edx)",
"mov(edx,'0x6465')",
"mov(dptr(eax+16),edx)",
"pop(edx)",
"push(eax)",
"call('0x1c703')"
```

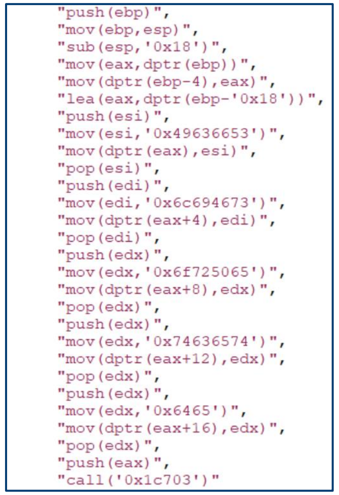
- Need to define similarity on semantics

Features from Semantics

Code

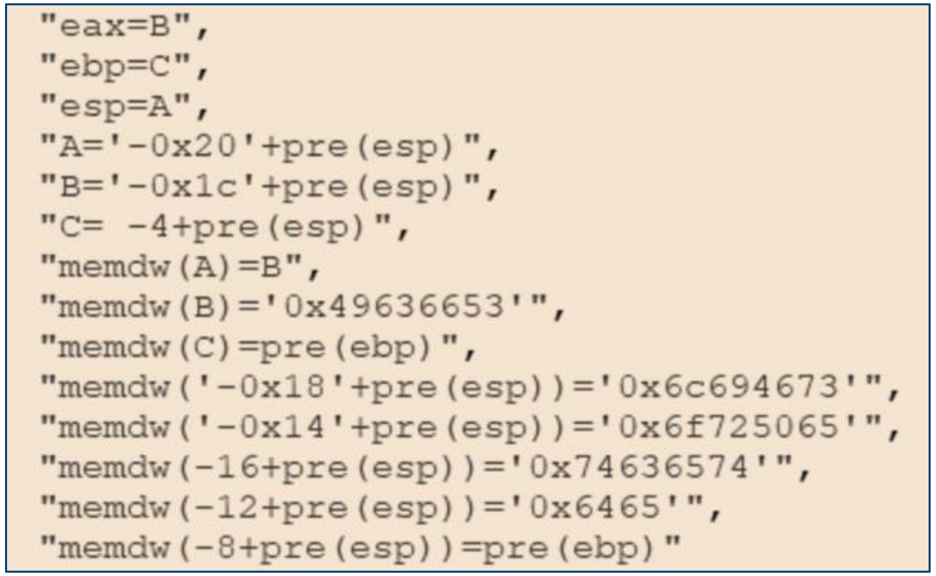


```
"push(ebp)",
"mov(ebp,esp)",
"sub(esp,'0x18')",
"mov(eax,dptr(ebp))",
"mov(dptr(ebp-4),eax)",
"lea(eax,dptr(ebp-'0x18'))",
"mov(dptr(eax),'0x49636653')",
"mov(dptr(eax+4),'0x6c694673')",
"mov(dptr(eax+8),'0x6f725065')",
"mov(dptr(eax+12),'0x74636574')",
"mov(dptr(eax+16),'0x6465')",
"push(eax)",
"call('0x129a')"
```



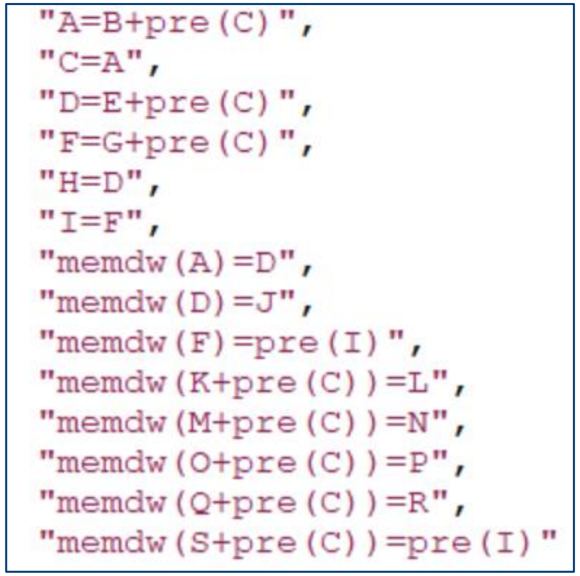
```
"push(ebp)",
"mov(ebp,esp)",
"sub(esp,'0x18')",
"mov(eax,dptr(ebp))",
"mov(dptr(ebp-4),eax)",
"lea(eax,dptr(ebp-'0x18'))",
"push(esi)",
"mov(esi,'0x49636653')",
"mov(dptr(eax),esi)",
"pop(esi)",
"push(edi)",
"mov(edi,'0x6c694673')",
"mov(dptr(eax+4),edi)",
"pop(edi)",
"push(edx)",
"mov(edx,'0x6f725065')",
"mov(dptr(eax+8),edx)",
"pop(edx)",
"push(edx)",
"mov(edx,'0x74636574')",
"mov(dptr(eax+12),edx)",
"pop(edx)",
"push(edx)",
"mov(edx,'0x6465')",
"mov(dptr(eax+16),edx)",
"pop(edx)",
"push(eax)",
"call('0x1c703')"
```

Semantics



```
"eax=B",
"ebp=C",
"esp=A",
"A='-0x20'+pre(esp)",
"B='-0x1c'+pre(esp)",
"C=  -4+pre(esp)",
"memdw(A)=B",
"memdw(B)='0x49636653'",
"memdw(C)=pre(ebp)",
"memdw('-0x18'+pre(esp))='0x6c694673'",
"memdw('-0x14'+pre(esp))='0x6f725065'",
"memdw(-16+pre(esp))='0x74636574'",
"memdw(-12+pre(esp))='0x6465'",
"memdw(-8+pre(esp))=pre(ebp)"
```

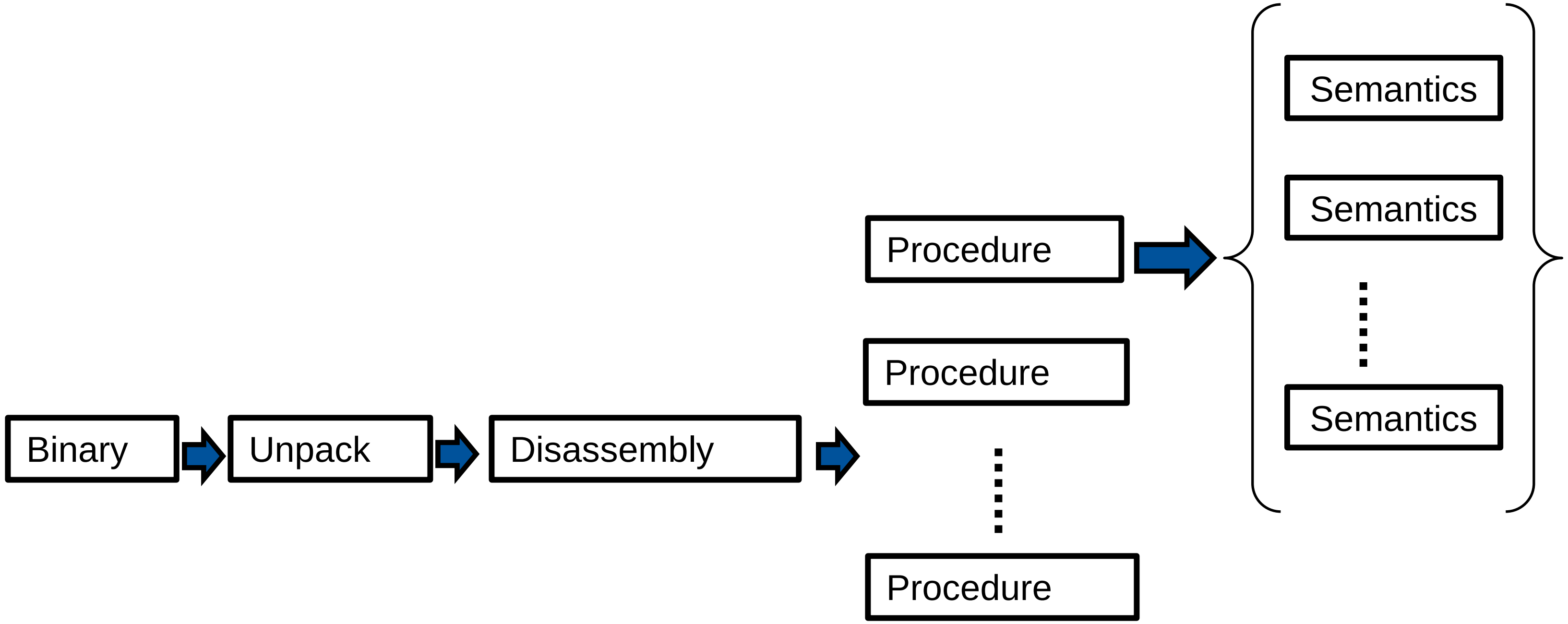
Generalized Semantics



```
"A=B+pre(C)",
"C=A",
"D=E+pre(C)",
"F=G+pre(C)",
"H=D",
"I=F",
"memdw(A)=D",
"memdw(D)=J",
"memdw(F)=pre(I)",
"memdw(K+pre(C))=L",
"memdw(M+pre(C))=N",
"memdw(O+pre(C))=P",
"memdw(Q+pre(C))=R",
"memdw(S+pre(C))=pre(I)"
```

- Interpret
- Normalize
- Generalize
- Index

Translating Binary to Feature Vector using Semantics



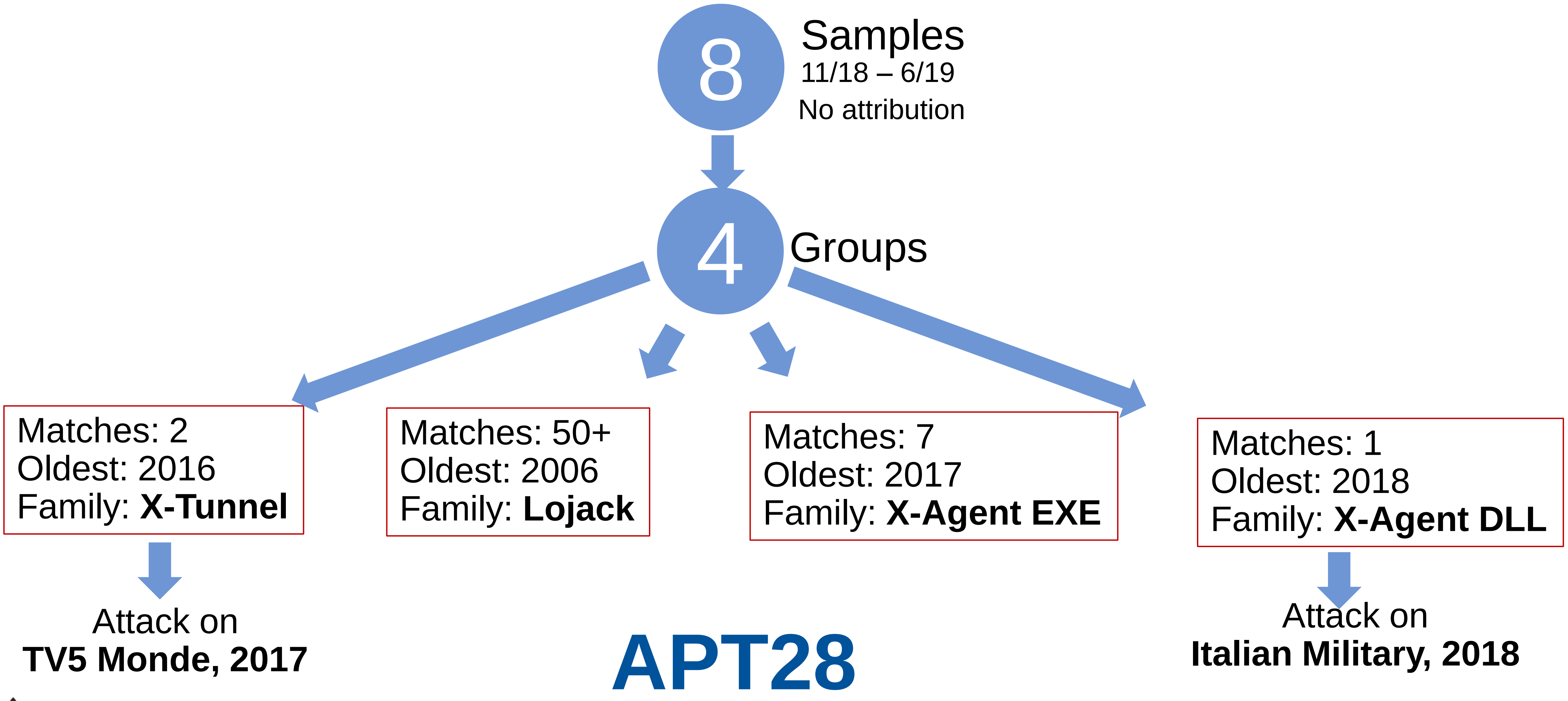
Binary

Feature Vector

Bag of Bag of Semantics

Demo: https://Beta.Magic.Cythereal.Com

Case Study: CYBERCOM CNMF Malware



8 Samples
11/18 – 6/19
No attribution

4 Groups

Matches: 2
Oldest: 2016
Family: **X-Tunnel**

Attack on
TV5 Monde, 2017

Matches: 50+
Oldest: 2006
Family: **Lojack**

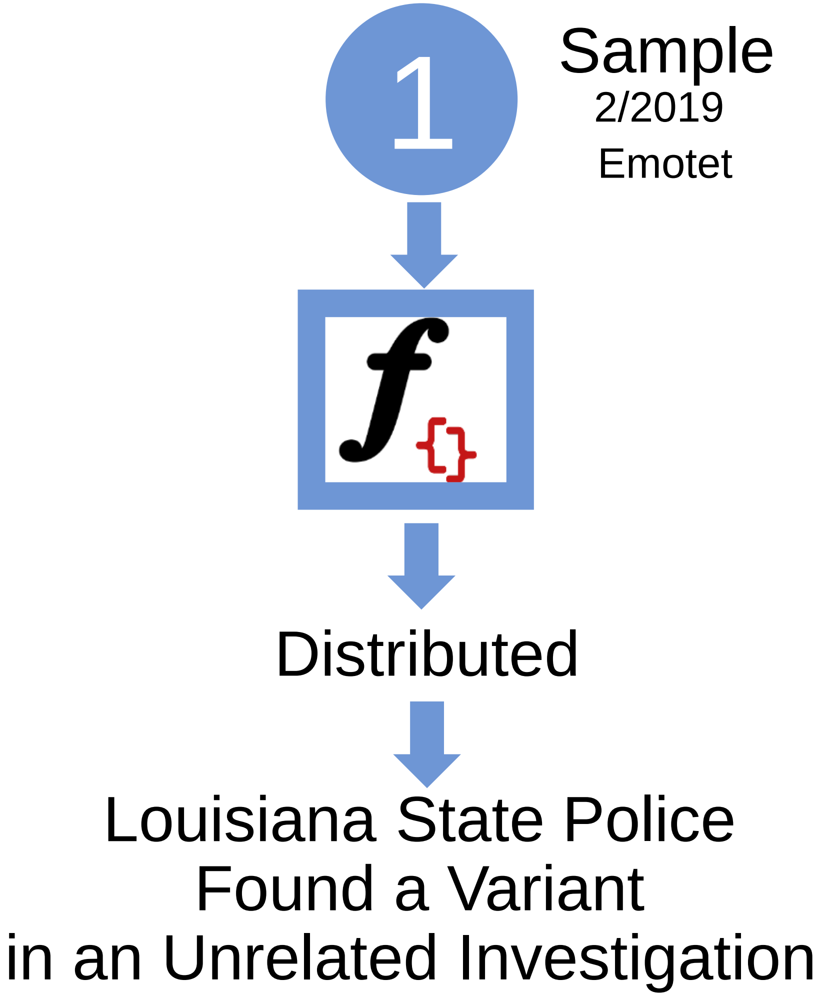
Matches: 7
Oldest: 2017
Family: **X-Agent EXE**

Matches: 1
Oldest: 2018
Family: **X-Agent DLL**

Attack on
Italian Military, 2018

APT28

Case Study: Louisiana State Police



Sample
2/2019
Emotet

Distributed

Louisiana State Police
Found a Variant
in an Unrelated Investigation

Power of Genome (data from 2019)

Executable Binaries in MAGIC Repository

# Binaries	3,413,184
# Genomically Unique	1,457,393
Genomic Compression	**57.3%**

Two binaries are genomically identical if ALL their procedures have the same genome.

Procedures Extracted from Binaries

# Procedures	1,658,759,504
# Genomically Unique	27,732,888
% Genomic Compression	**98.33%**

Two procedures are unique if they have the same same abstracted semantics

Binary Similarity (> 0.7)

# Similarity Binary Nodes	3,978,430
# Similarity relationships	426,121,442
Avg number of similar binaries	**107**

Twitter: @DrArunL
Email: arun@louisiana.com

Hybrid Analysis Search

- Free Automated Malware Analysis Service - powered by Falcon Sandbox - Search results from HA Community Files (hybrid-analysis.com)