



POLITECNICO
DI TORINO

HISPASEC

Looking for the perfect signature: an automatic YARA rules generation algorithm in the AI-era

Andrea Marcelli (@_S0nn1_)

andrea.marcelli@polito.it



Who I am

PhD Student at Politecnico di Torino

ML, semi-supervised modeling, optimization problems



**POLITECNICO
DI TORINO**

Security Researcher at Hispasec Sistemas

developing AI algorithms for large-scale malware detection

HISPASEC

Twitter @_S0nn1_

Email andrea.marcelli@polito.it

Web jimmy-sonny.github.io

Agenda

The signature generation problem

The algorithm

Introducing YaYaGen

Demo

The signature generation problem

What is a malware signature?

A **unique pattern** that indicates the presence of malicious code

As malware evolves, new signatures need to be generated frequently

Syntactic signatures are based on unique sequences of instructions or strings

* this is where most of the existing tools and researches focus on

Semantic signatures provides an abstraction of the program behavior

Our target is **Android malware**, although the approach is generic

This is not just another ML classifier.

Work motivations



Reduce
the malware
exposure time



Automate
a repetitive task

20k - 50k SUBMISSIONS
EVERY DAY



100% recall
and high
precision
requirements



Save
a considerable
amount of time
and resources

About YARA



“YARA is to files what Snort is to network traffic”

Designed to be fast

The **de-facto standard** language to write malware signatures

Natively supports **syntactic signatures** (strings + regex + hex)

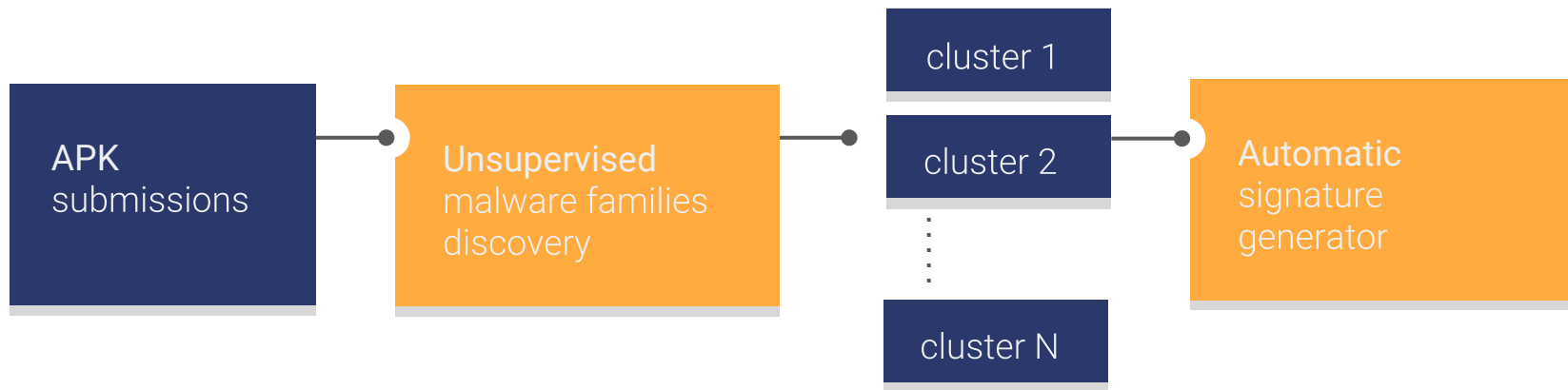
Semantic signatures are defined through custom modules.

An example of YARA rule

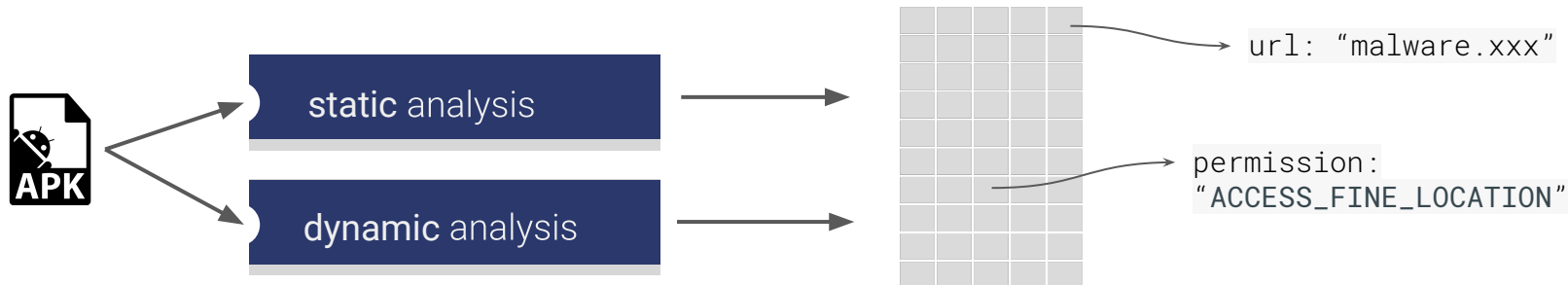
```
rule YaYaSyringe {  
    meta:  
        author = "YaYaGen @BSIDESLV"  
  
    strings:  
        $a = "text here"  
        $b = { E2 34 A1 C8 23 FB }  
  
    condition:  
        $a and $b  
        and androguard.filter("action.BATTERYCHECK")  
        and androguard.number_of_services == 3  
        ...  
}
```


The algorithm

The detection workflow



The attributes



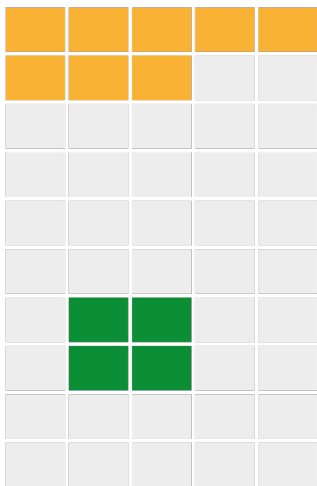
Each block is an attribute extracted through the analysis

*Androguard, *Droidbox, Cockoo are used

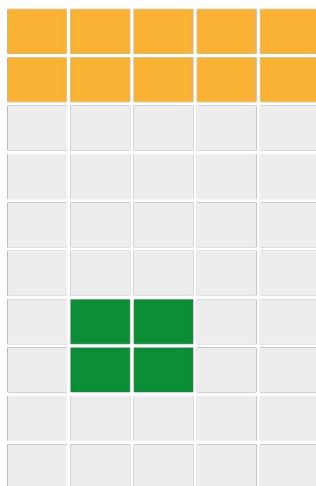
The **quality of the analysis** affects the signature generation process.

* They require a custom YARA module

Sample 1



Sample 2

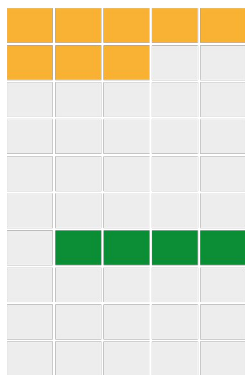


=

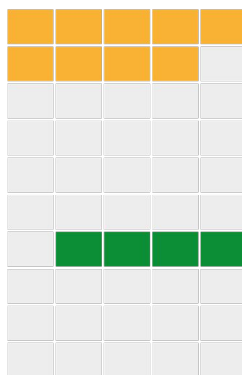
Signature



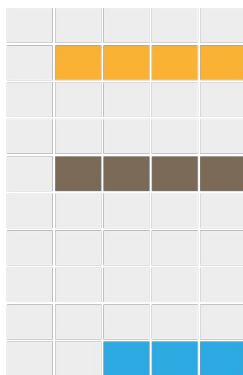
Sample 1



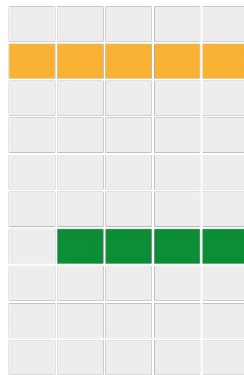
Sample 2



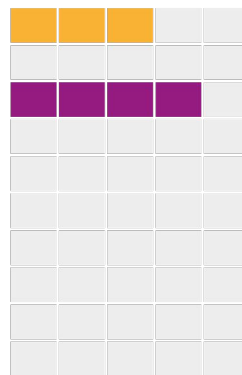
Sample 3



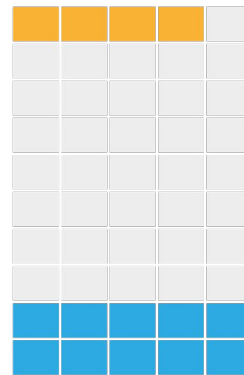
Sample 4



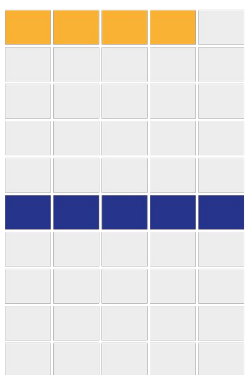
Sample 5



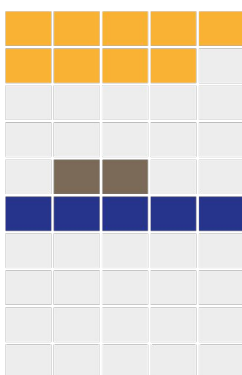
Sample 6



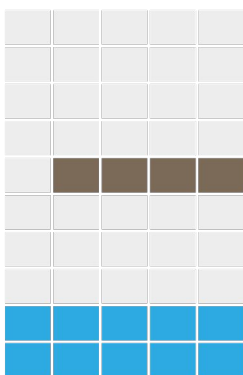
Sample 7



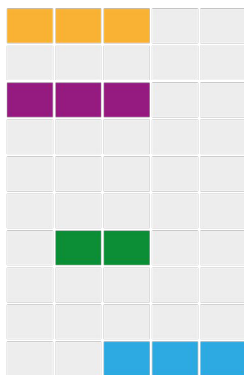
Sample 8



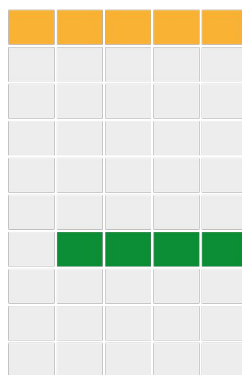
Sample 9



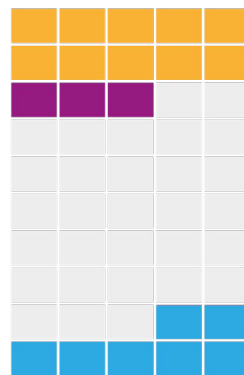
Sample 10



Sample 11



Sample 12



The solution



Finding the optimal attributes subsets is the goal of the signature generation process

The problem can be reduced to a variant of the set cover problem (NP-complete)

A **dynamic greedy algorithm** builds the signature as a disjunction of clauses.

Normal form

$$\underbrace{(l_1 \wedge l_2 \wedge l_3)}_{\text{clause}} \vee \underbrace{(l_4 \wedge l_5)}_{\text{literal}}$$

Signature anatomy

Each signature can be expressed in **DNF**

$$S = \bigvee_{i=0}^n c_i \quad c_i = \bigwedge_{j=0}^{m(i)} l_{i,j}$$

Each **clause** can be **weighed**

$$w(c_i) = \sum_{j=0}^{m(i)} w(l_{i,j})$$

The weight of a signature is **the lowest** among its clauses

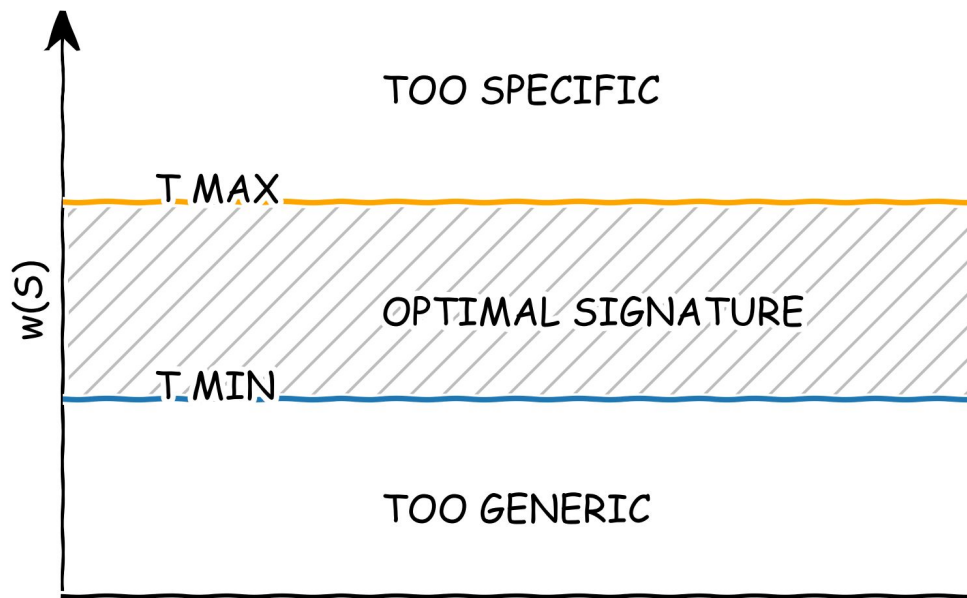
$$w(S) = \min_{\forall i} w(c_i)$$

Generality vs specificity

A **weighting system** evaluates the rules

The **higher the weight, the less FPs**
Possibly more FNs

The **lower the weight, the more FPs**
Possibly less FNs

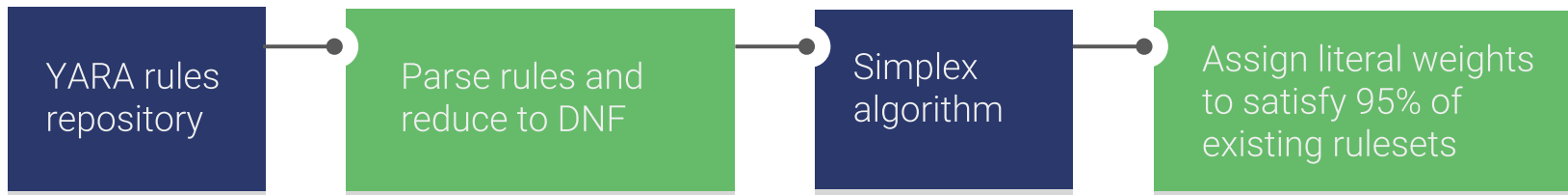


Weighting process

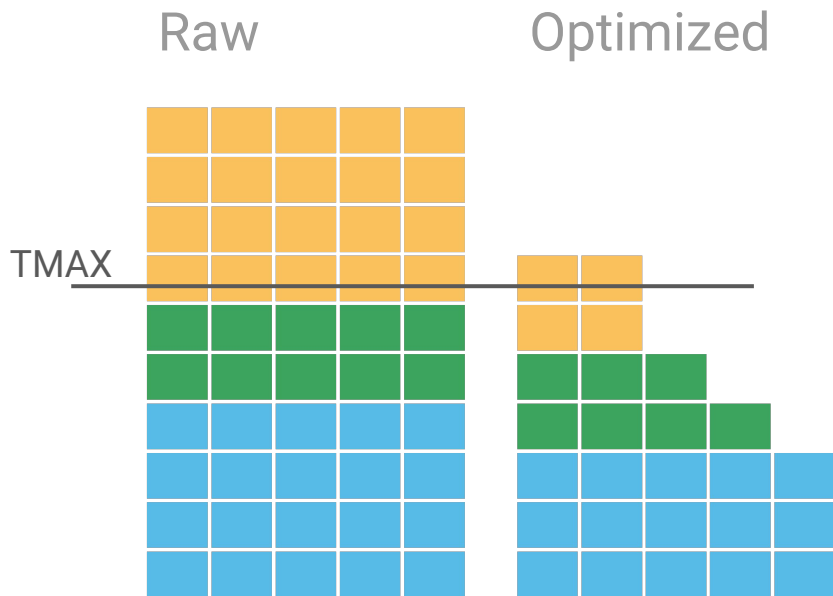
Assigning correct weights is the key to guarantee high quality results

The value of **TMIN** and **TMAX** is strongly related to the choice of the weights

The process is designed to be automated



Optimization



Experimental results show that automatically generated rules could be **over-specific**

Two approaches:

- **Basic optimizer** (< 1 min)
a greedy optimizer
- **Evo optimizer** (~ 5 min)
based on EA. Encodes the human expert knowledge

Introducing YaYaGen

Yet another YARA rule Generator

*YaYa is grandma in Spanish



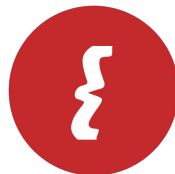
From

a set of application analysis reports



to

a set of YARA rules



Yet another YARA rule Generator

*YaYa is grandma in ES



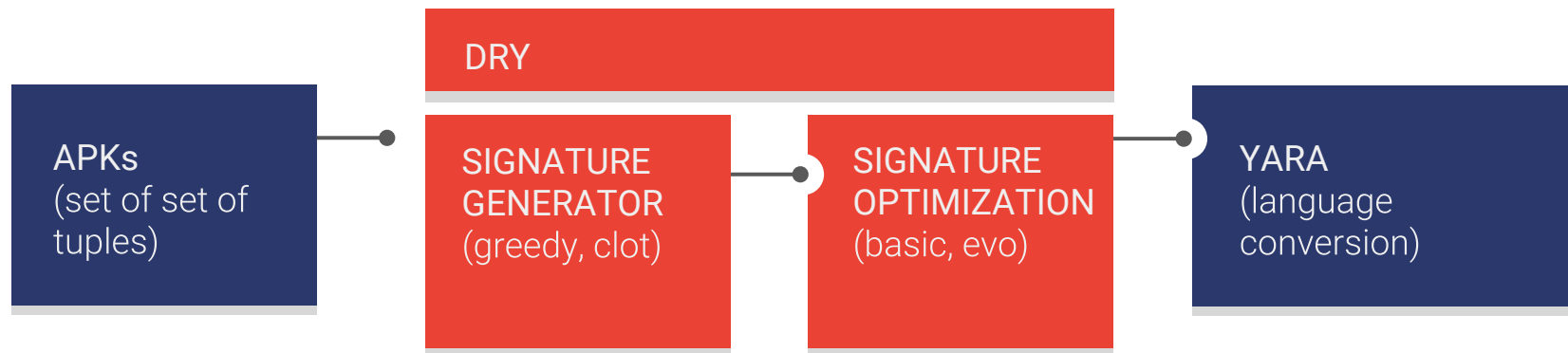
2 algorithms (*clot, greedy*), 2 optimizers (*basic, evo*) and some **heuristics**

Includes a **YARA rule parser** for attributes weight optimization

Supports **FP exclusion** from rule generation

Written in Python 3

YaYaGen internals



Ready for ***Koodous** (collaborative platform for Android malware research)

* <https://koodous.com/> 

Fork me on GitHub



<https://github.com/jimmy-sonny/YaYaGen>

Demo

Results

Results

Rule Name	Original	YaYaGen	Improvement
SMSSENDER	539	1,004	+86.3%
SYRINGE	220	315	+43.2%
HUMMINGBAD2	136	257	+89.0%
MARCHER2	559	652	+16.6%
SMSREG	159	172	+8.2%
VOLCMANDROPPER	186	430	+131.2%

Takeaways

Automatically generated rules **perform better** than manual written ones

However, there is still room for improvement

Existing rulesets are used to trim the algorithm

Exploit the expert knowledge

The time required to generate a rule from **100 apps** is **less 5 minutes**

Ready for real-world applications

Ongoing work

YaYaGenPE



HISPASEC



POLITECNICO
DI TORINO



Thank you



Questions?

* References

Griffin, Kent, et al. "Automatic generation of string signatures for malware detection." *International workshop on recent advances in intrusion detection*. Springer, Berlin, Heidelberg, 2009.

Preda, Mila Dalla, et al. "A semantics-based approach to malware detection." *ACM SIGPLAN Notices* 42.1 (2007): 377-388.

Perdisci, Roberto, Wenke Lee, and Nick Feamster. "Behavioral Clustering of HTTP-Based Malware and Signature Generation Using Malicious Network Traces." *NSDI*. Vol. 10. 2010.

Faruki, Parvez, et al. "AndroSimilar: robust statistical feature signature for Android malware detection." *Proceedings of the 6th International Conference on Security of Information and Networks*. ACM, 2013.

Zheng, Min, Mingshen Sun, and John CS Lui. "Droid analytics: A signature based analytic system to collect, extract, analyze and associate android malware." *Trust, Security and Privacy in Computing and Communications, 2013 12th IEEE International Conference on*. IEEE, 2013.

Corno, Fulvio, Matteo Sonza Reorda, and Giovanni Squillero. "The selfish gene algorithm: a new evolutionary optimization strategy." *Proceedings of the 1998 ACM symposium on Applied Computing*. ACM, 1998.

<https://github.com/Xen0ph0n/YaraGenerator>

<https://github.com/Neo23x0/yarGen>

<https://github.com/AlienVault-OTX/yabin>

<https://www.talosintelligence.com/bass>