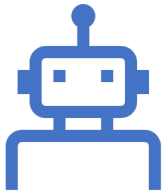
UNIVERSITÀ DEGLI STUDI
DEL SANNIO Benevento



Generative adversarial network per la generazione di malware: tra mito e realtà

Corrado Aaron Visaggio

Associate Professor

Dept. Of Engineering

University of Sannio

09/10/2019 Roma

Eventi GARR

Obiettivi dell'intervento

COMPRENDERE L'UNIVERSO DEL MALWARE

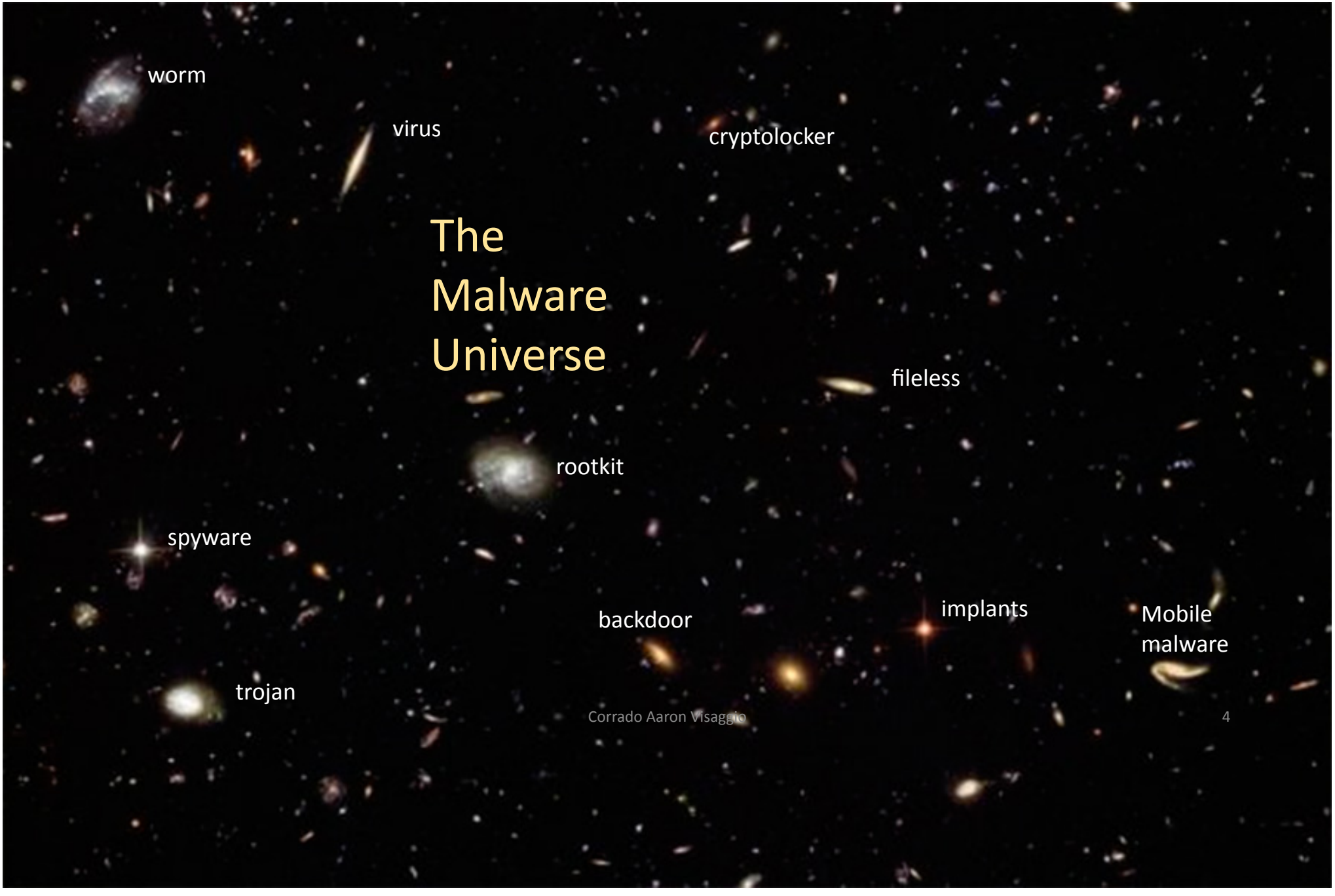
FINALITA' DELLA MALWARE ANALYSIS

GAN PER LA PRODUZIONE DEL MALWARE

LE FRONTIERE DELLA RICERCA NELLA MALWARE ANALYSIS

Malware: i fatti

- **200,000** malware al giorno
- **80M** nuovi malware all'anno
- **700M+** malware in the wild
- **Targeted malware** vs automatic generated malware
- **Zero-day** e cyber-arsenal nascosti

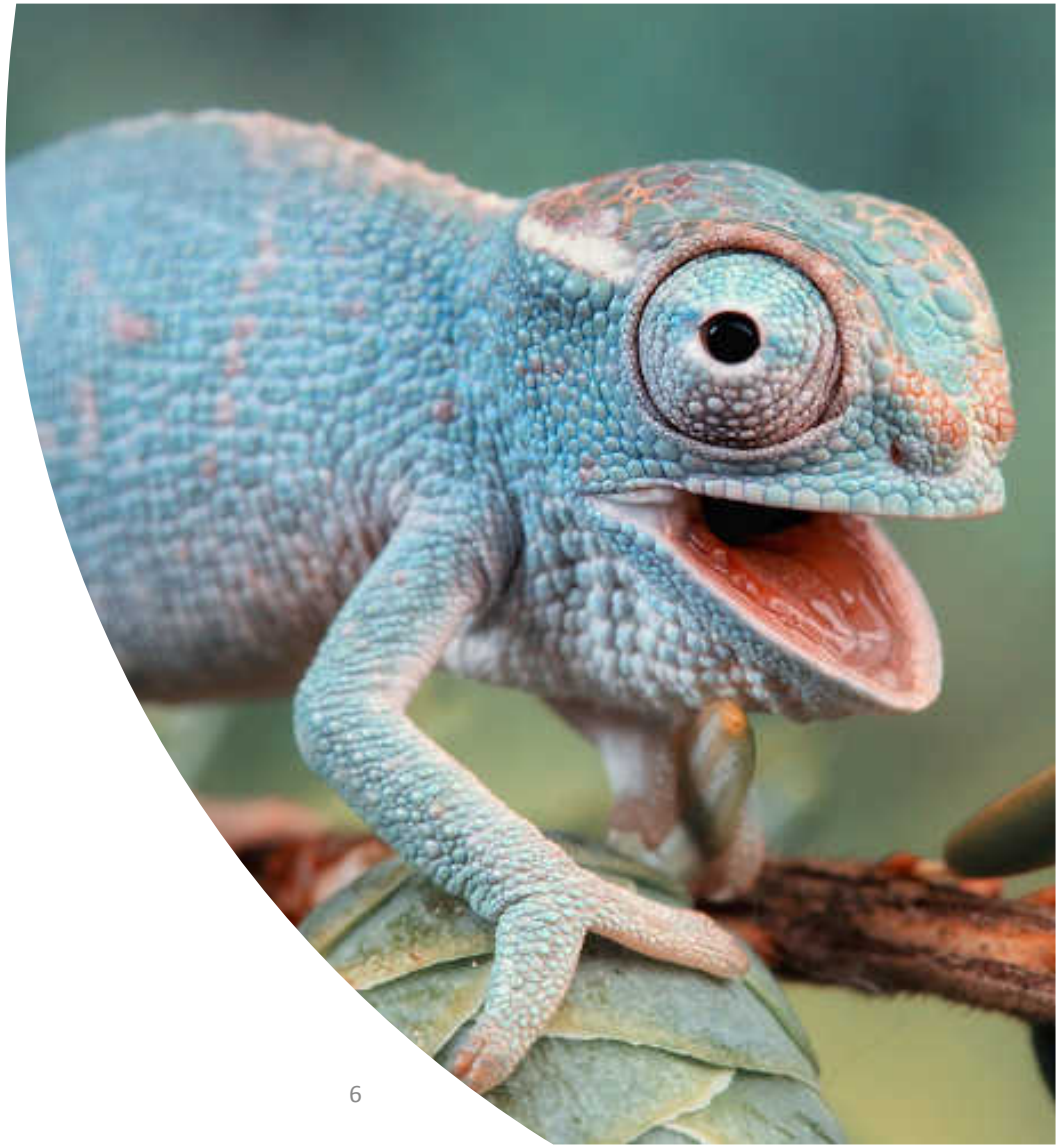

worm
virus
cryptolocker
The Malware Universe
fileless
rootkit
spyware
backdoor
implants
Mobile malware
trojan

Le nuove Minacce

- Fileless
- Aggressive Evasion techniques
- Implants
- GAN (?)

Malware Analysis

- **Malware Detection**: è un malware?
- **Malware Similarity**: a cosa è simile il mio malware?
  - Variants detection
  - Family detection
  - Similarities detection
  - Differences detection
- **Malware category detection**: a quale classe appartiene il mio malware?



Malware Analysis con Machine Learning

- ***Classification***: il processo di classificazione consta di due passi: **model construction** e **model usage**. Il classificatore etichetta il testing set sulla base del ***modello*** e delle ***feature*** estratte.
- ***Clustering****: raggruppare il malware che esibisce* **comportamenti simili** in gruppi diversi. E' usato per la generazione della signature.

Malware Detection…

Pros	Cons
Easy to run Fast identification Broadly accessible Finding comprehensive malware information Hexaustive Not harmful	Failing to detect the polymorphic/ encrypted / obfuscated/packed malwares Replicating information in the huge database time window between a malware's release and its detection by anti-malware software tools is about 54 days [Hu 2011].

- ***Signature Based***: Il metodo basato sulle firme identifica stringhe uniche di codice binario [Moskovitch et al. 2009]

… Malware Detection

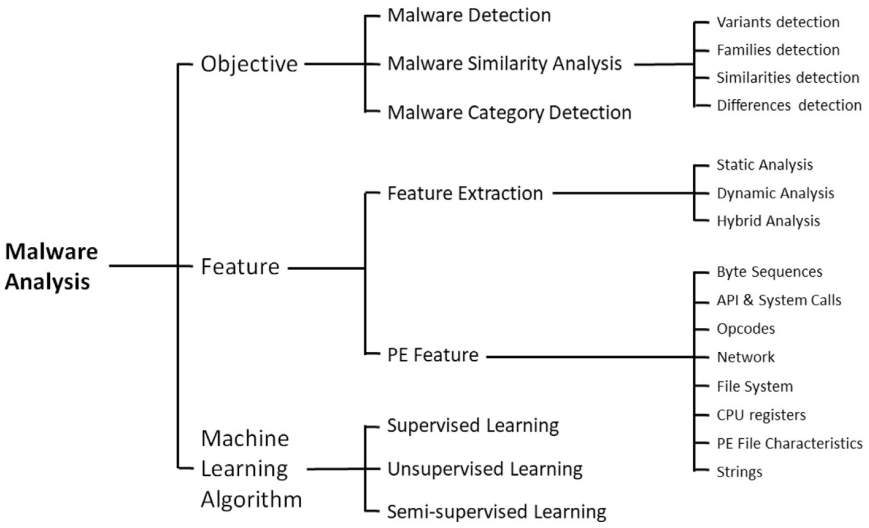
Pros	Cons
Detecting unconceived types of malware attacks Data-flow dependency detector Detecting the polymorphic malwares	Storage complexity for behavioral patterns Time complexity Coverage limitations Anti-debugging/virtualization techniques

- ***Behavior Based***: catturare pattern di esecuzione e caratteristiche facendo esplodere il malware in ambienti virtualizzati imitando condizioni di sistemi suscettibili all'infezione

Features

- **Windows API & System Calls:** Windows API calls are used by almost all programs to send the requests to the operating system -> can reflect the behavior of the program
- **N-grams:** N-grams are all substrings in the program code with a length of *N*
- **Strings:** The interpretable strings are the high-level specifications of malicious behaviors. These strings can reflect the attacker's intent and goal since they often contain the important semantic information
- **Opcodes:** An OpCode (i.e., Operational Code) is the subdivision of a machine language instruction that identifies the operation to be executed
- **Control Flow Graphs (CFGs):** A CFG is a graph that represents the control flow of a program.
- **Network Activity**: used protocols, TCP/UDP ports, HTTP requests, DNS-level in- teractions
- **File System**: how many files are read or modified, what types of files and in what directories, and which files appear in not-infected/infected machines
- **CPU registers**: whether any hidden register is used, and what values are stored in the registers, especially in the FLAGS register
- **PE file characteristics*:*** sections, imports, symbols, used compilers

Malware Analysis & Machine Learning



«Ucci, D., Aniello, L., & Baldoni, R. (2018). Survey of machine learning techniques for malware analysis. *Computers & Security*»

Ma tutto questo funziona nella realtà?

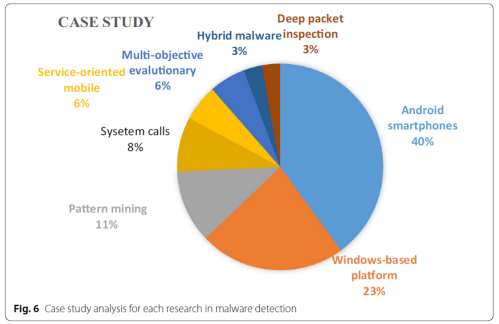


Fig. 6 Case study analysis for each research in malware detection

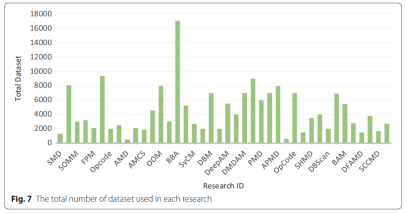


Fig. 7 The total number of dataset used in each research

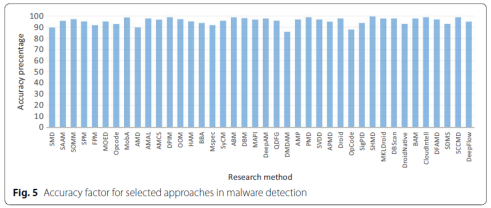
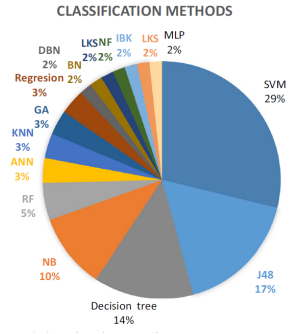


Fig. 5 Accuracy factor for selected approaches in malware detection



- Souri, Alireza, and Rahil Hosseini. "A state-of-the-art survey of malware detection approaches using data mining techniques." *Human-centric Computing and Information Sciences* 8, no. 1 (2018): 3-25.

Metamorphic Malware Detection…

The method applies **frequency analysis** to program instructions of the disassembled virus' code.

A **classifier** uses frequencies to establish whether the program is a metamorphic virus or not.

8086 Op-code (A)	Number of unique instructions with more than one occurrence: IOM(op-code) (B)
AAA	0
AAD	5
AAM	3
AAS	2
ADC	0
ADD	15
...	...
INT	2
...	...

```
push ds
push es
mov  ax,'DA'
int  21h
cmp  ax,'PS'
jz   done_install
mov  ah, 4Ah
mov  bx,0FFFFh
int  21h
mov  ah, 4Ah
int  21h
mov  ah, 48h
int  21h
mov  es, ax
dec  ax
mov  ds, ax
int  22h
int  22h
```

MACHINE LEARNING CLASSIFIER

Gerardo Canfora, Antonio Niccolò Iannaccone, Corrado Aaron Visaggio: ***Static analysis for the detection of metamorphic computer viruses using repeated-instructions counting heuristics***. J. Computer Virology and Hacking Techniques 10(1): 11-27 (2014)

… Metamorphic Malware Detection

- DATASET:
  - **Control group:** 250 NO-MALWARE
  - **1° Exp Group**: 500 META-MALWARE G2, MPCGEN, NGVCK, NRLG, SMEG
  - **2° Exp Group**: 250 NO-META MALWARE

gorithm	TP RATE	FP RATE	PRECISION	RECALL	F-MEASURE	ROC AREA	CLASS
	0.98	0.013	0.97	0.98	0.97	0.99	No-Malware
	0.98	0.001	0.99	0.98	0.98	0.99	G2
	1	0.004	0.97	1	0.98	0.99	MpcGen
	1	0.001	0.99	1	0.99	0.99	Nrlg
	1	0	1	1	1	1	Ngvck
	0.97	0.003	0.98	0.97	0.97	0.99	Smeg
	0.91	0.005	0.96	0.91	0.93	0.97	NO-METAMORPHIC MALWARE
	0.99	0.012	0.97	0.99	0.98	0.99	No-Malware
	0.97	0.005	0.96	0.97	0.96	0.99	G2
	0.91	0.005	0.96	0.91	0.93	0.99	MpcGen
	1	0.007	0.95	1	0.98	1	Nrlg
	1	0.008	0.94	1	0.97	1	Ngvck
	0.95	0.001	0.99	0.95	0.97	1	Smeg
	0.80	0.017	0.86	0.8	0.84	0.96	NO-METAMORPHIC MALWARE
	0.98	0.003	0.99	0.98	0.99	0.99	No-Malware
	0.99	0.004	0.97	0.99	0.98	0.99	G2
	0.98	0.007	0.95	0.98	0.97	0.99	MpcGen
	1	0	1	1	1	1	Nrlg
		0	1	1	1	1	Ngvck
		0.001	0.99	1	0.99	1	Smeg
		0.007	0.95	0.89	0.89	0.98	NO-METAMORPHIC MALWARE
		0	1	0.972	0.99	0.99	No-Malware
		0.003	0.98	1	0.99	0.99	G2
		0.001	0.99	1	0.99	1	MpcGen
			1	1	1	1	Nrlg
		3	0.98	1	0.99	1	Ngvck
			1	1	1	1	Smeg
			0.93	0.95	0.94	0.99	NO-METAMORPHIC MALWARE
			99	0.96	0.97	0.99	No-Malware
				0.98	0.97	0.99	G2
				0.99	0.98	0.99	MpcGen
				1	1	1	Nrlg
				99	0.99	0.99	Ngvck
					0.96	0.98	Smeg
					0.87	0.94	NO-METAMORPHIC MALWARE

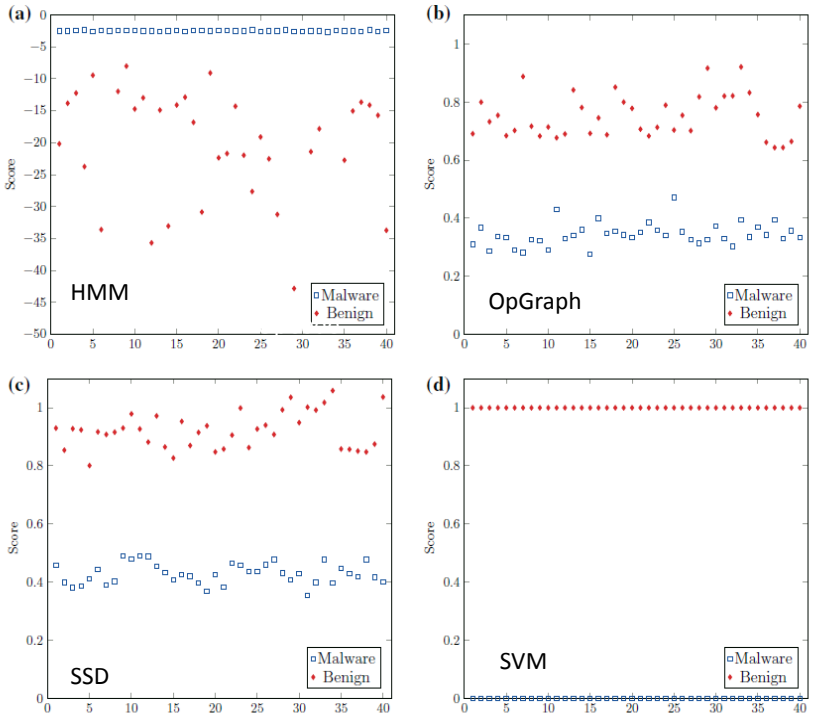
Malware detection with Statistical techinques – HMM, SSD, OpGraph, SVM…

Family	Number of files
Harebot	50
NGVCK	200
Security Shield	50
Smart HDD	50
Winwebsec	200
Zbot	200
ZeroAccess	200
Benign	40

Hidden Markov Models, Simple Substitution Distance and ***Opcode Graph similarity*** are compared

Tanuvir Singh, Fabio Di Troia, Corrado Aaron Visaggio, Thomas H. Austin, Mark Stamp: "**Support vector machines and malware detection**". J. Computer Virology and Hacking Techniques 12(4): 203-212 (2016)

… Malware detection with Statistical techinques – HMM, SSD, OpGraph, SVM

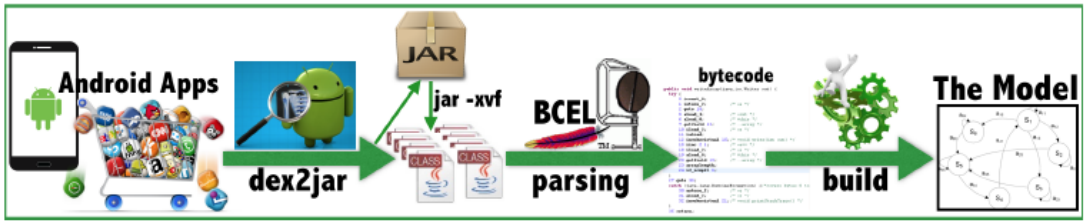


Ransomware

1. Each Java Byte code instruction is transformed into a CCS process
2. Recognize specific properties with mu-calculus (temporal logic)

3. Concurrency Workbench of New Century (CWB-NC) as formal verification environment.

$$cond \in \{eq,\ ne,\ icmeq,\ icmpne,\ ...\}$$



Francesco Mercaldo, Vittoria Nardone, Antonella Santone, Corrado Aaron Visaggio: ***Ransomware Steals Your Phone***. ***Formal Methods Rescue It***. FORTE 2016: 212-221

G. Canfora, F. Martinelli, F. Mercaldo, V. Nardone, A. Santone, C.A. Visaggio: ***LEILA: formaL tool for idEntifying mobIle maLicious behavior.*** IEEE Trans. On Software Engineering, preprint 2019.

Ransomware

```java
public AesCrypt(String paramString)
  throws Exception
{
  MessageDigest localMessageDigest = MessageDigest.getInstance("SHA-256");
  localMessageDigest.update(paramString.getBytes("UTF-8"));
  byte[] arrayOfByte = new byte[32];
  System.arraycopy(localMessageDigest.digest(), 0, arrayOfByte, 0, arrayOfByte.length);
  this.cipher = Cipher.getInstance("AES/CBC/PKCS7Padding");
  this.key = new SecretKeySpec(arrayOfByte, "AES");
  this.spec = getIV();
}
```

```
prop RW_4 = (min X = <pushSHADdueCcinqueSsei> RW1_4 \/ <-pushSHADdueCcinqueSsei>X)

prop RW1_4 = (min X = <invokegetInstance> RW2_4 \/ <-invokegetInstance>X)

prop RW2_4 = (min X = <pushUTFOotto> RW3_4 \/ <-pushUTFOotto>X)

prop RW3_4 = (min X = <invokegetBytes> RW4_4 \/ <-invokegetBytes>X)

prop RW4_4 = (min X = <pushAESCBCPKCSSsettePadding> RW5_4 \/ <-pushAESCBCPKCSSsettePadding>X)

prop RW5_4 = (min X = <invokegetInstance> RW6_4 \/ <-invokegetInstance>X)

prop RW6_4 = (min X = <newjavaxcryptospecSecretKeySpec> RW7_4 \/ <-newjavaxcryptospecSecretKeySpec>X)

prop RW7_4 = (min X = <pushAES> tt \/ <-pushAES>X)
```

Fig. 4: Java source code related to the code snippet able to cipher files stored on an infected device. The mu-calculus formula able to catch this behaviour on the model.

Ransomware

Table 2: Dataset used in the Experiment

Dataset	Original Samples	Morphed Samples	#Samples for Category
Ransomware	683	594	1,277
Other Malware	600	0	600
Trusted	600	0	600
Total	1,883	594	**2,477**

Table 4: Top 10 Signature-Based Antimalware Evaluation Against Our Method.

Antimalware	**Original**			**Morphed**		
	%ident.	#ident.	#unident.	%ident.	#ident.	#unident.
AhnLab	*13.76%*	94	589	*5.22%*	31	563
Alibaba	*0.44%*	3	680	*0%*	0	594
Antiy	*13.18%*	90	593	*4.04%*	24	570
Avast	*27.52%*	188	495	*6.4%*	38	556
AVG	*3.22%*	22	661	*1.51%*	9	585
Avira	*19.76%*	135	548	*12.46%*	74	520
Baidu	*14.34%*	98	585	*6.7%*	41	553
BitDefender	*28.26%*	193	490	*14.47%*	86	508
ESET-NOD32	*20.35%*	139	544	*8.58%*	51	543
GData	*27.96%*	191	492	*7.91%*	47	547
Our Method	***99.56%***	**680**	**3**	***99.49%***	**591**	**3**

Table 5: Performance Evaluation

Formula	**# Samples**	**TP**	**FP**	**FN**	**TN**	**PR**	**RC**	**Fm**	**Acc**
Ransomware	2,477	1,271	0	6	1,200	1	0.99	**0.99**	**0.99**

Dealing with the Unknown

- ***Concept drift:*** ***nuove varianti*** *di un nuovo malware deteriorano le performances di un classificatore nel tempo.*
  - ***Un training potrebbe includere instanze out-of-date***
  - ***Untraining set potrebbe includere un numero insufficiente di istanze***
- ***Unknown***: malware **non utilizzato** per **addestrare** il classificatore
- **Research Question**: quale è la ***resilienza*** di un classificatore quando esamina **unknown** malware?
- **Dataset**: Microsoft Kaggle[1] database (10869 instances belonging to 9 families)
- **Process of Analysis**: PCA + Binary Classification with Unknown + Binary Classificatoin with Known.

[1]https://github.com/albertsanso/kaggle-microsoft-malware/tree/master/malware/2nditeration

Resilience

- Accuracy $a = \frac{TP+TN}{TP+FP+TN+FN}$

- Resilience $R = a_{w/o\_unknown} - a_{w\_unknown}$

Machine learning is strongly sensitive to the unknown.

Ramnit							
Model	Lollipop	Kelihos v. 3	Vundo	Tracur	Kelihos v.1	Obfuscator	Gatak
KNeighbors	23.73	77.45	16.6	9.07	6.28	18.05	39.97
Perceptron	9.77	0.069	26.25	18.84	27.33	17.15	17.55
Stochastic gradient descent	13.12	14.74	8.93	14.71	24.82	14.77	3.42
Gaussian Naive Bayes	8.6	0.06	33.26	24.89	0.00	16.43	0.83
logistic regression	11.65	1.32	8.45	11	30.6	9.98	0.18
support vector machine	0.39	2.00	0.03	18.30	1.01	5.03	0.86
decision tree	25.70	53.32	23.78	17.33	82.86	4.74	8.94
random forest	26.08	81.33	7.48	9.39	1.01	4.31	11.94
Lollipop							
Model	Ramnit	Kelihos v.3	Vundo	Tracur	Kelihos v.1	Obfuscator	Gatak
KNeighbors	32.84	24.52	25.58	28.91	2.26	24.93	0.26
perceptron	5.69	9.77	18.35	18.94	1.3	14.19	21.61
Stochastic gradient descent	4.97	0.23	16.42	17.4	1.3	8.71	33.97
Gaussian Naive Bayes	8.46	0.02	9.68	5.64	7.48	2.29	0.68
logistic regression	10.38	4.23	15.45	7.87	3.46	14.1	21.59
support vector machine	6.2	0.11	9.55	2.26	1.3	4.69	0.19
decision tree	32.8	18.12	5.18	18.49	18.08	15.25	19.03
random forest	13.89	2.09	12.43	18.66	19.34	8.82	1.28
Kelihos v. 3							
Model	Ramnit	Lollipop	Vundo	Tracur	Kelihos v.1	Obfuscator	Gatak
KNeighbors	5.36	47.05	0.21	1.73	-	6.55	12.24
perceptron	37.81	60.79	34.81	30.55	-	5.53	56.15
Stochastic gradient descent	27.98	50.29	28.44	15.05	-	6.98	77.87
Gaussian Naive Bayes	0.88	0.70	0.17	1.52	-	0.030	2.99
logistic regression	23.67	54.55	30.41	28.09	-	5.90	50.64
support vector machine	6.99	29.44	8.74	2.02	-	5.28	39.32
decision tree	12.05	3.15	4.63	3.33	-	0.140	4.82
random forest	1.08	16.78	0	0.53	-	0.32	0
Vundo							
Model	Ramnit	Lollipop	Kelihos v.3	Tracur	Kelihos v.1	Obfuscator	Gatak
KNeighbors	0.32	39.44	10.19	23.97	31.68	23.04	30.84
perceptron	3.26	61.14	13.33	55.27	16.68	13.64	0.29
Stochastic gradient descent	14.96	50.44	4.89	44.59	82.73	7.8	0.19
Gaussian Naive Bayes	88.95	86.12	99.9	85.26	69.35	67.63	96.69
logistic regression	11.1	62.96	3.98	48.59	24.37	20.94	1.82

Generative Adversarial Networks and malware

- Esempi avversariali sono usati per imbrogliare i modelli di identificazione basati sull'apprendimento
- Gli Attackers ignorano le features e il modello di classificazione
- Noi assumiamo che gli autori di malware possono sapere quali sono le feature usate nel modello, ma non conosciamo il modello di classificazione.
- MalGAN [Hu and Tan, 2017] genera vettori di feature avversariali.
- Il modello avversariale si costituisce di un generatore e di un identificatore sostituto che addestrano una rete neurale.

MalGAN

- Se M ***APIs*** vengono usate come ***features***, un vettore di feature M-dimensionali è costruito per un programma (il malware). Se il programma chiama la d-ima API, la d-ima feature è avvalorata ad 1, altrimenti a 0.
- La distribuzione di probabilità dei campioni avversariali prodotti da MalGAN è determinate dai ***pesi*** del generatore.

Quali conclusioni sulle GAN

- Per ora sono un mito, una realtà solo nella matematica.
- E' possibile che mentre parliamo il Dark Side of the force stia già realizzando motori GAN per la generazione di malware.
- Insieme agli implants e ai fileless malware i MalGAN completano lo spettro delle nuove minacce.
- Siamo preparati?

I principali problemi del dataset

- Rapida **obsolescenza**
- Dataset **Incompleto o sbilanciato**
  - Windows vs Linux/MacOS
  - Android vs IOS
  - Workstation vs SCADA
- **Rappresentatività** della popolazione
  - IOT is a typical example
- Possiamo **fidarci** del dataset?
- Quanto il dataset **polarizza** la ricerca (Strategia + obiettivi)?

Le sfide della ricerca

- ***Incremental learning*****:** aggiornamento continuo dei modelli ad apprendimento
- ***Active learning***: aumentare la rappresentatività del campione
- ***Prediction of malware prevalence***: come prevedere i malware trend
- ***Adversarial learning*****:** come sviluppare tecniche di difesa dall'apprendimento avversariale
- ***Malware Attribution***: caratterizzare l'authorship del malware
- ***Malware Triage***: metodi per prioritizzare la malware analysis
- ***Malware detection in its infancy***: molte campagne sono realizzate con malware analizzato ma non noto.
- ***Finding new features:*** selezionare feature più efficaci

Bibliography…

- Robert Moskovitch, Clint Feher, and Yuval Elovici. 2009. A chronological evaluation of unknown malcode detection. *LNCS: Intelligence and Security Informatics* 5477 (2009), 112–117.
- Xin Hu. 2011. *Large-scale malware analysis, detection, and signature generation.* Ph.D. Dissertation, Department of Computer Science and Engineering, University of Michigan.
- Damballa. 2008. 3% to 5% of Enterprise Assets Are Compromised by Bot-Driven Targeted Attack Malware. Retrieved from http://www.prnewswire.com/news-releases/3-to-5-of-enterprise-assets-are-compromisedby-bot-driven-targeted-attack-malware-61634867.html.
- Yanfang Ye, Tao Li, Shenghuo Zhu, Weiwei Zhuang, Egemen Tas, Umesh Gupta, and Melih Abdulhayoglu. 2011. Combining file content and file relations for cloud based malware detection. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*.
- Isabelle Guyon and Andr Elisseeff. 2003. An introduction to variable and feature selection. *Jouranl of Machine Learning Research* 3 (March 2003), 1157–1182.
- Pat Langley. 1994. Selection of relevant features in machine learning. In *Proceedings of AAAI Fall Symposium*. Hanchuan Peng, Fuhui Long, and Chris Ding. 2005. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27, 8 (2005), 1226–1238.
- J. Ross Quinlan. 1993. *C4.5: Programs for Machine Learning.* San Francisco, CA: Morgan Kaufmann Publishers, Inc. (1993).

… Bibliography…

- George H. John and Pat Langley. 1995. Estimating continuous distributions in Bayesian classifiers. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*.
- Pedro Domingos and Michael Pazzani. 1997. On the optimality of simple Bayesian classifier under zero-one loss. Machine Learning 29, 2–3 (1997), 103–130.
- Evelyn Fix and Joseph L. Hodges Jr. 1951. Discriminatory analysis-nonparametric discrimination: Consistency properties. *US Air Force, School of Avaiation Medicine, Tech. Rep* 4 (1951), 5–32.
- Christopher M. Bishop. 1995. Neural networks for pattern recognition. *Oxford, Clarendon Press.*
- Thorsten Joachims. 1998. Making large-scale support vector machine learning practical. *Advances in Kernel Methods: Support Vector Machines* (1998).
- Fadi Abdeljaber Thabtah. 2007. A review of associative classification mining. *Knowledge Engineering Review* 22, 1 (2007), 37–65.
- Yoshua Bengio. 2009. Learning deep architectures for AI. *Foundations and Trends in Machine Learning* 2, 1 (2009), 1–127.
- Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. 2007. Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems* 19 (2007).
- Thomas G. Dietterich. 2000. Ensemble methods in machine learning. In *Proceedings of the 1st International Workshop on Multiple Classifier Systems*.
- Ucci, D., Aniello, L., & Baldoni, R. (2018). Survey of machine learning techniques for malware analysis. *Computers & Security*
- Souri, Alireza, and Rahil Hosseini. "A state-of-the-art survey of malware detection approaches using data mining techniques." *Human-centric Computing and Information Sciences* 8, no. 1 (2018): 3-25.

… Bibliography

- Hu, W. and Tan, Y., 2017. Generating adversarial malware examples for black-box attacks based on GAN. *arXiv preprint arXiv:1702.05983*.
- Stefan Axelsson. The Base-Rate Fallacy and the Difficulty of Intrusion Detection. ACM TISSEC, 2000.
- Enrico Mariconti, Lucky Onwuzurike, Panagiotis Andriotis, Emiliano De Cristofaro, Gordon Ross, and Gianluca Stringhini. MaMaDroid: Detecting Android Malware by Building Markov Chains of Behavioral Models. In NDSS, 2017.
- Daniel Arp, Michael Spreitzenbarth, Malte Hubner, Hugo Gascon, and Konrad Rieck. DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket. In NDSS, 2014.
- Pendlebury, F., Pierazzi, F., Jordaney, R., Kinder, J. and Cavallaro, L., 2019. {TESSERACT}: Eliminating Experimental Bias in Malware Classification across Space and Time. In *28th {USENIX} Security Symposium ({USENIX} Security 19)* (pp. 729-746).

Declare variables,
not war.

```
public int peace;
```

Execute programs,
not people.

```
find / -type f -exec sed -i 's/war/peace/g' {} \;
```