A short history of Malware Research

Davide Balzarotti

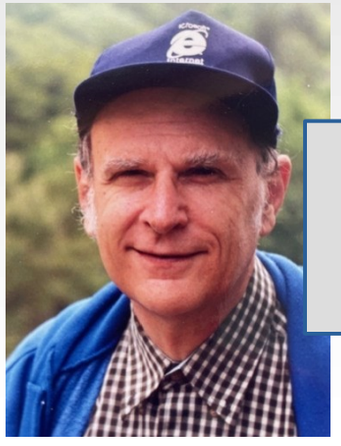
Davide Balzarotti

@balzarot

Professor of Computer Security @ Eurecom

Malware & Binary Analysis, Web, Forensics, Fuzzing, ...

https://www.s3.eurecom.fr/~balzarot/security-circus



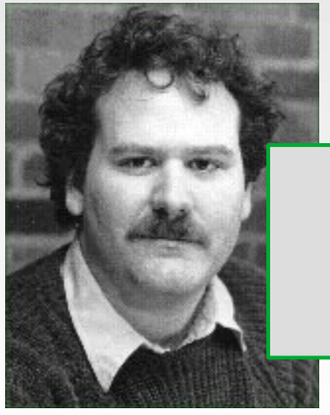
Daniel Edwards coined the word “Trojan Horse”

to operate. In essence it bypasses any and all security controls that may otherwise exist on most systems. It is the quintessence of the malicious threat against contemporary systems.

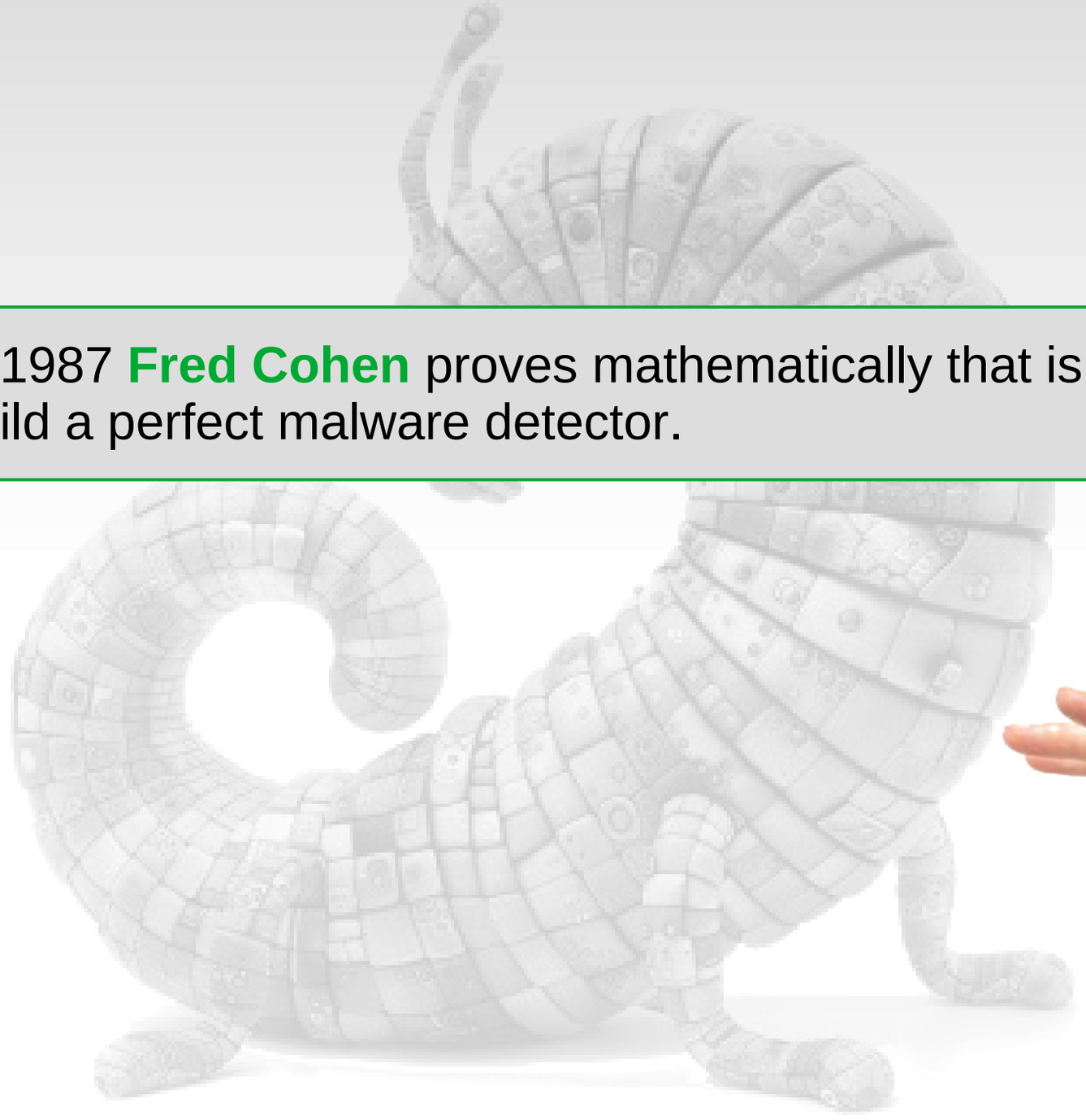
COMPUTER SECURITY TECHNOLOGY PLANNING STUDY - VOLUME II
1972 - James P. Anderson

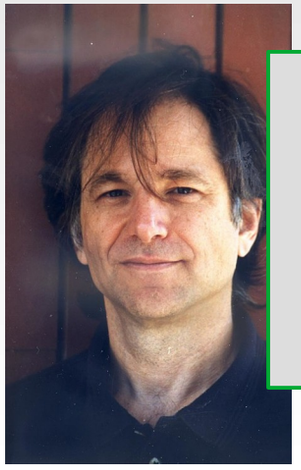
Part I
The Early Days

Where we focused on <u>understanding</u> what we can do and what we need

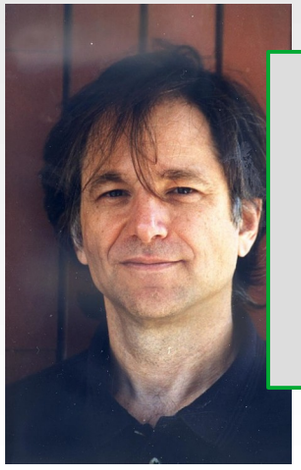


In 1987 **Fred Cohen** proves mathematically that is impossible to build a perfect malware detector.





In 1989 **Leonard Adleman** concludes:
“Thus detecting viruses is quite intractable, and it seems unlikely that a protection systems predicated on virus detection will be successful”



In 1989 **Leonard Adleman** concludes:
“Thus detecting viruses is quite intractable, and it seems unlikely that a protection systems predicated on virus detection will be successful”

John McAfee

Academia

Theoretical and Mathematical approach
to malware detection.

New theorems to prove
that **<everything>** is undecidable

Industry

Practical, Engineering approach
to malware detection.

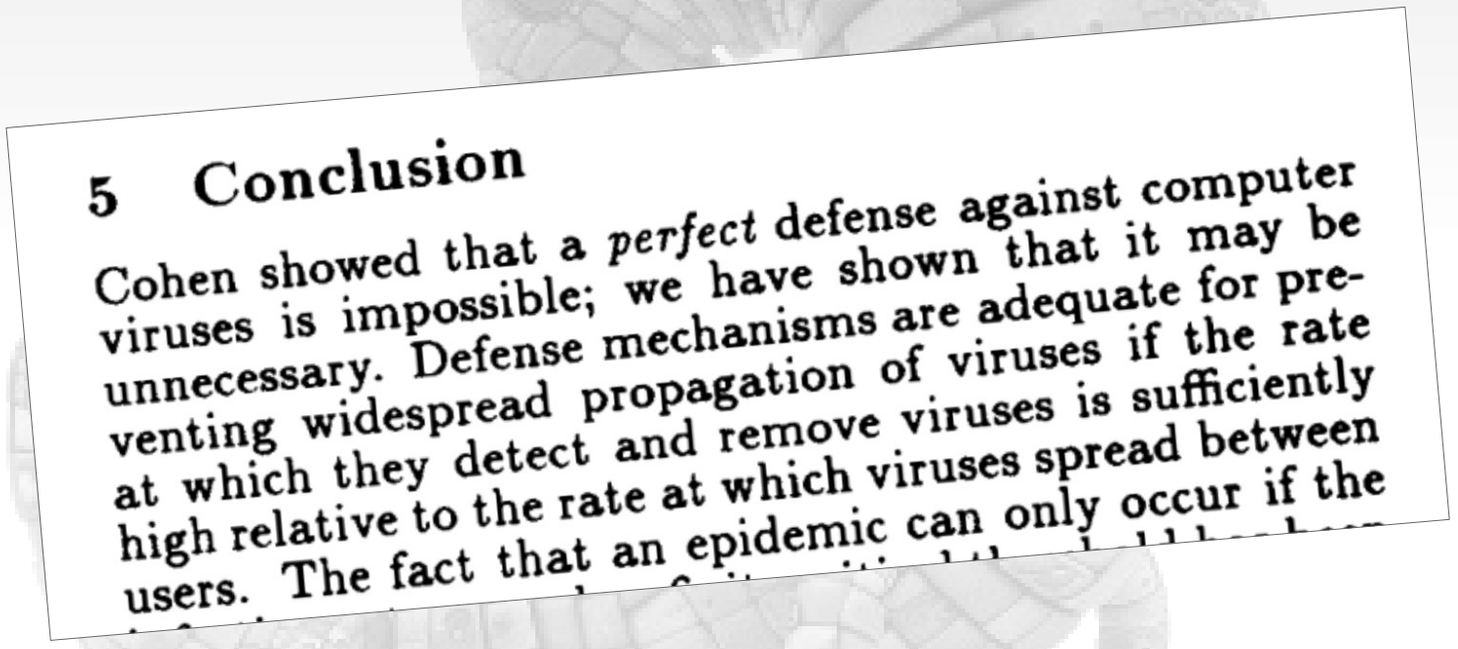
- Signatures

- Heuristics

- Reputation

- Machine Learning

“Directed-graph epidemiological models of computer viruses."
Security & Privacy 1991 - Kephar & White



5 Conclusion

Cohen showed that a *perfect* defense against computer viruses is impossible; we have shown that it may be unnecessary. Defense mechanisms are adequate for preventing widespread propagation of viruses if the rate at which they detect and remove viruses is sufficiently high relative to the rate at which viruses spread between users. The fact that an epidemic can only occur if the

If we consider finite-length viruses, good-enough detectors (i.e., that might have some rare false positives) can be implemented to terminate in linear time

"Reliable Identification of Bounded-Length Viruses is NP-Complete"
IEEE Transactions of Information Theory 2003. Spinelli

If we consider finite-length viruses, good-enough detectors (i.e., that might have some rare false positives) can be implemented to terminate in linear time

"Reliable Identification of Bounded-Length Viruses is NP-Complete"
IEEE Transactions of Information Theory 2003. Spinelli

If we restrict the space or time that a program is allowed, deciding whether a program is packed is NP-complete.

When other disciplines encounter these problems, they rely on good-average case algorithms, approximate algorithms, heuristics..

"Detecting Traditional Packers, Decisively"
RAID 2013 -- Bueno, Compton, Sakallah, Bailey

Early Days - Summary

Everything is Undecidable in the general case,
and NP-Complete if we put space/time constraints.

But we do not need perfect solutions.
We can build a good-enough detector if we can accept some false positives and false negatives.

Part II
Something is Going to Work

Where we focused on solutions (with poor understanding)

- Signatures
- Heuristics
- Reputation
- Machine Learning

“A cost analysis of typical computer viruses and defenses”
Computer Virus and Security Conference 1991. Fred Cohen

Signature Scanning is not a practical solution.

“A Generic Virus Scanner in C++”
ACSAC 1992. Kumar and Spafford

We believe that the cost-benefit ratio for scanners, either by themselves or in addition to other mechanisms, is much higher than [Cohen] calculates. This is because of scanners' low impact on existing practice and because of their flexibility

“Automatic extraction of computer virus signatures"
Virus Bulletiin 1994 – Kehpart & Arnold

Extract sequences of 12-36 bytes from different files infected from the same virus, and then statistically assess their FP against a large dataset of benign programs.

- Signatures
- Heuristics
- Reputation
- Machine Learning

“A cost analysis of typical computer viruses and defenses”
Computer Virus and Security Conference 1991. Fred Cohen

Signature Scanning is not a practical solution.

“A Generic Virus Scanner in C++”
ACSAC 1992. Kumar and Spafford

We believe that the cost-benefit ratio for scanners, either by themselves or in addition to other mechanisms, is much higher than [Cohen] calculates. This is because of scanners' low impact on existing practice and because of their flexibility

“Automatic extraction of computer virus signatures"
Virus Bulletiin 1994 – Kehpart & Arnold

This patent-pending technique has been used to either extract or evaluate the more than 2500 virus signatures used by IBM AntiVirus. It obviates the need for a small army of virus analysts, permitting IBM's signature database to be maintained by a single virus expert working halftime.

- Signatures
- Heuristics
- Reputation
- Machine Learning

"Computer Viruses – A Management Overview"
Computer Law & Security Report 1991. Edward Wilding
(editor of VirusBulletin)

accumulates as more viruses appear. Even the most efficient search engines which employ 'smart' scanning are able to detect an upper limit of approximately 10000 computer viruses. At the current level of development this point will be reached within two to three years. Beyond this point scanners will become exhausted – there will be too much search data for the search algorithm to manipulate at reasonable speed.

- gnatures
- euristics
- eputation
- achine Learning

“A Generic Virus Scanner in C++”
ACSAC 1992. Kumar and Spafford

Virus detection by behavioral abnormality
E.g., write to boot sectors, modify interrupt vectors, write to system files. etc.

“MCF: a malicious code filter”
Computer & Security 1995 - Lo, Levitt, Olsson

Tell-signs extracted by static analysis.
They must be fundamental enough so that certain malicious action is impossible without showing the tell-sign.
Most are related to system calls.

“Semantics-Aware Malware Detection”
Oakland 2005 - Christodorescu, Dawn Song, Somesh Jah

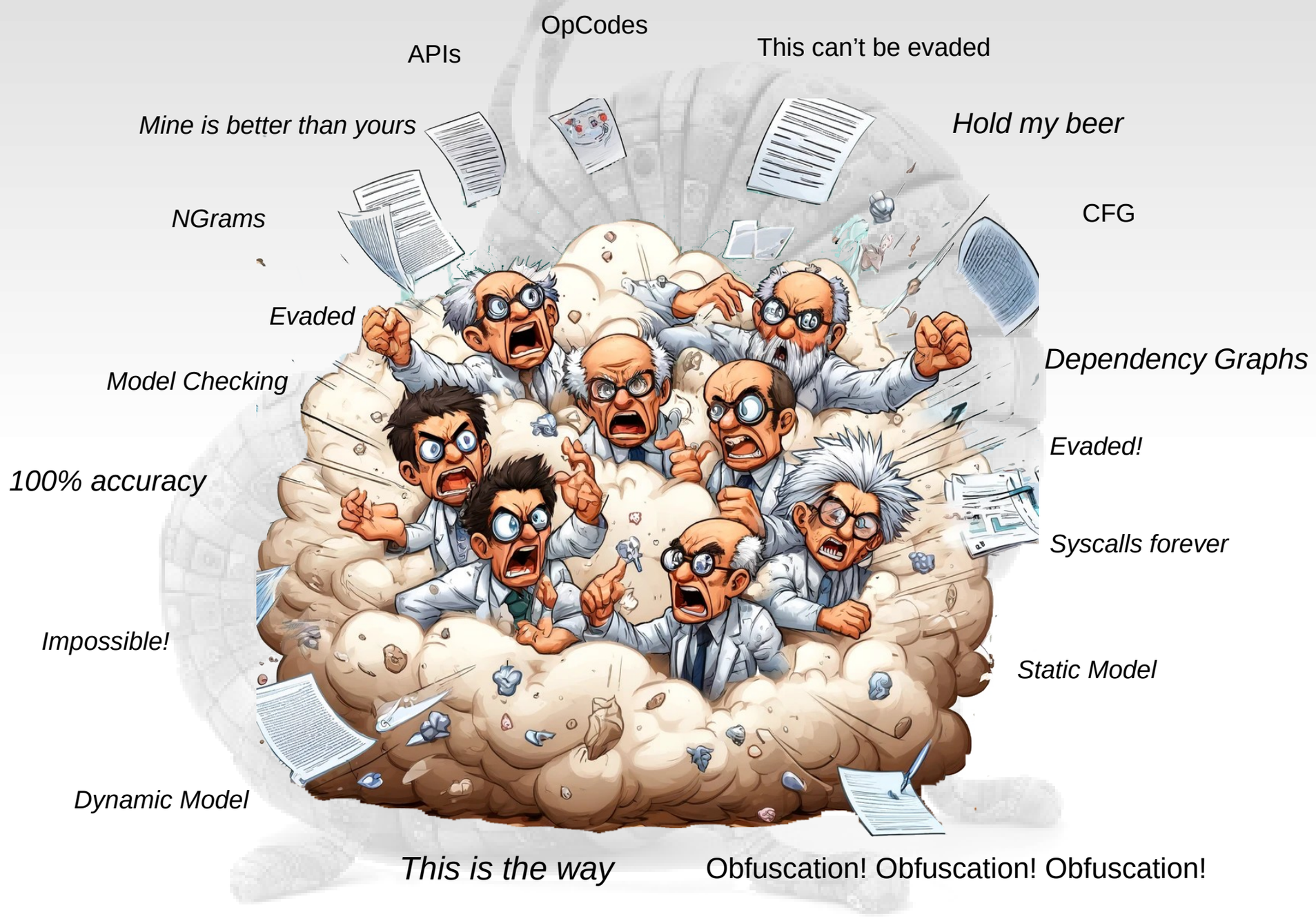
Behavioral templates, which are instruction sequences where variables and symbolic constants are used.
An approximate matching algorithm is proposed that is resilient to common forms of obfuscation.

- Signatures
- Heuristics
- Reputation
- Machine Learning

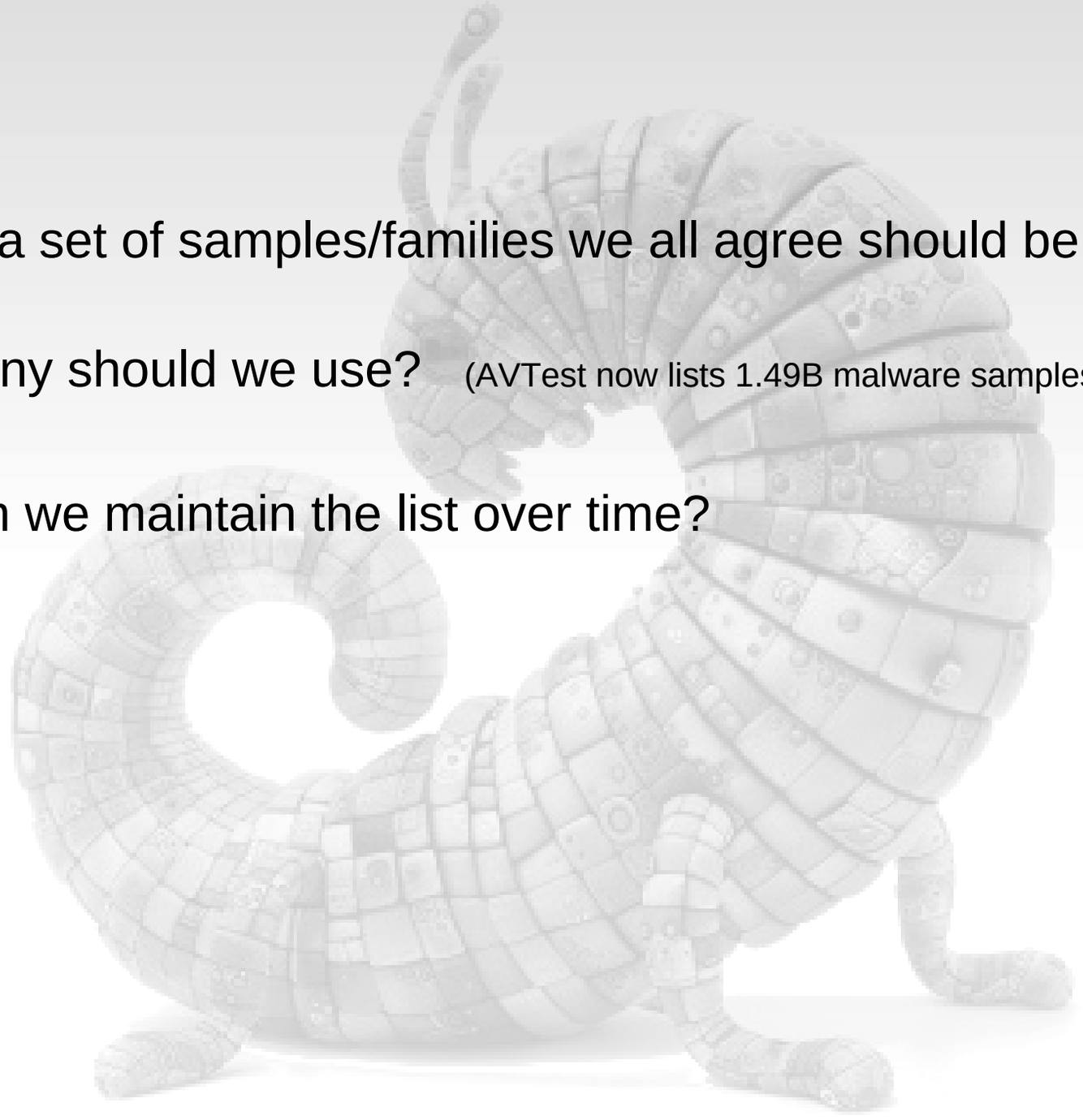
AV-TEST
Tests of Anti-Virus-Software independent • qualified • fast

The Average Anti-Malware Product

	2005	2010
Installer Size	12,6 MB	69,6 MB
Size on Disk	87,9 MB	265,5 MB
Number of Signatures	104.509	3.666.872
Size of Signature File	7,7 MB	84,4 MB
Price	45 €	32 €
Updates per Day	2	6
WildList Detection	(virtually) 100%	(virtually) 100%
Zoo Detection	93,04%	91,59%
False Positives	0,03%	0,00157%


OpCodes
APIs
This can’t be evaded
Mine is better than yours
Hold my beer
NGrams
CFG
Evaded
Dependency Graphs
Model Checking
Evaded!
100% accuracy
Syscalls forever
Impossible!
Static Model
Dynamic Model
This is the way
Obfuscation! Obfuscation! Obfuscation!

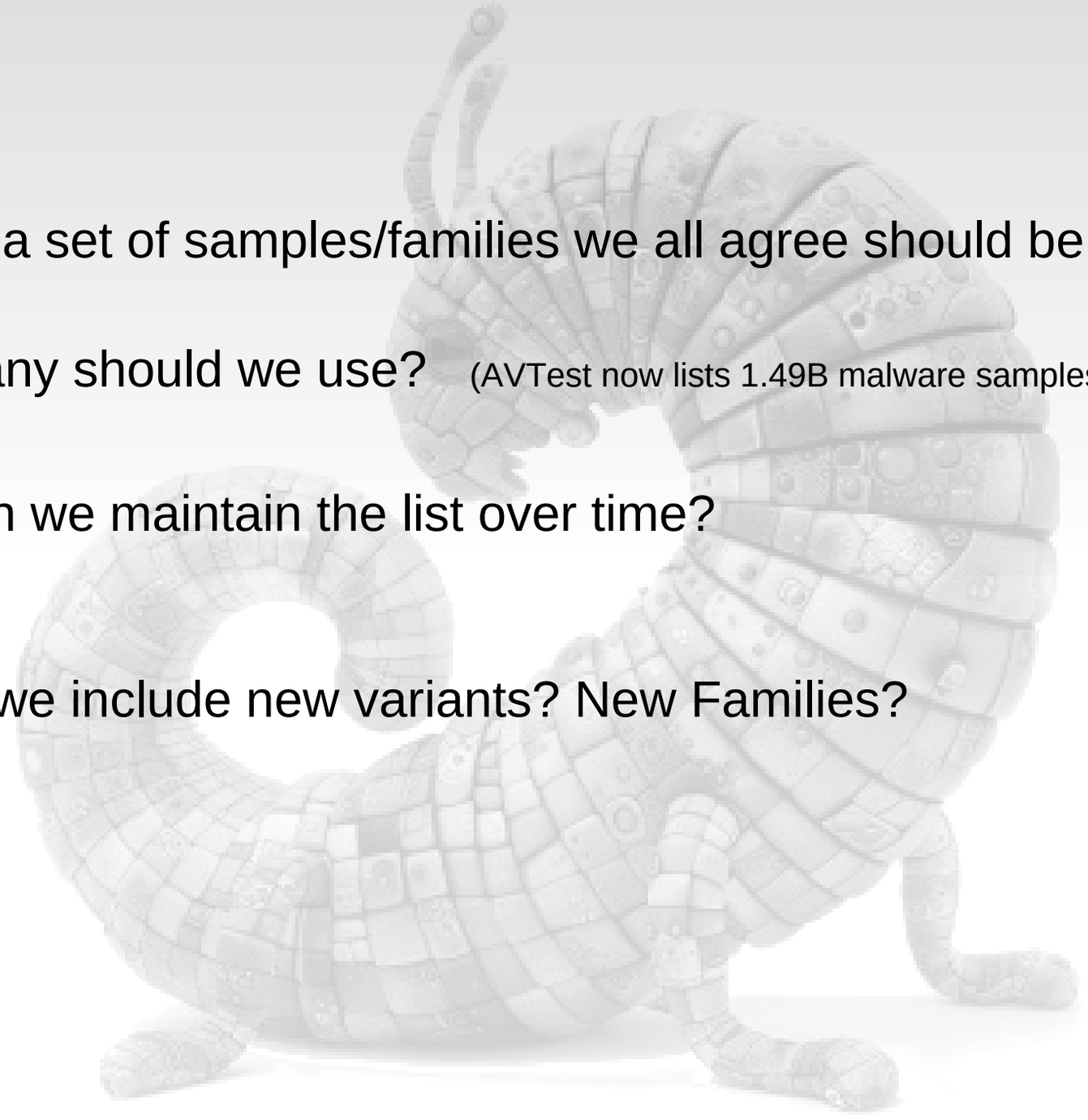
How should we evaluate a malware detector?



Is there a set of samples/families we all agree should be detected?

How many should we use? (AVTest now lists 1.49B malware samples)

How can we maintain the list over time?



Is there a set of samples/families we all agree should be detected?

How many should we use? (AVTest now lists 1.49B malware samples)

How can we maintain the list over time?

Should we include new variants? New Families?

Is there a set of samples/families we all agree should be detected?

How many should we use? (AVTest now lists 1.49B malware samples)

How can we maintain the list over time?

Should we include new variants? New Families?

Should we consider “how easy” it is to evade detection?

And how do you even define “easy”?

“A Guideline to Anti-Malware-Software testing”
European Institute for Computer Anti-Virus Research 2000 - Marx

Precise Guidelines,
Wildlist vs Zoo

“Retrospective testing – how good heuristics really work”
Virus Bulletin 2002 - Marx

Future Malware

“A Guideline to Anti-Malware-Software testing”
European Institute for Computer Anti-Virus Research 2000 - Marx

Precise Guidelines, Wildlist vs Zoo

“Retrospective testing – how good heuristics really work”
Virus Bulletin 2002 - Marx

Future Malware

“Testing Malware Detectors"
ISSTA 2004 – Christodorescu & Jha

Transformations

“A Guideline to Anti-Malware-Software testing”
European Institute for Computer Anti-Virus Research 2000 - Marx

Precise Guidelines, Wildlist vs Zoo

“Retrospective testing – how good heuristics really work”
Virus Bulletin 2002 - Marx

Future Malware

“Testing Malware Detectors"
ISSTA 2004 – Christodorescu & Jha

Transformations

“TESSERACT: Eliminating experimental bias in malware classification across space and time" - USENIX Security Symposium 2019
Pendlebury, Pierazzi, Jordaney, Kinder, Cavallaro

ML Pitfalls

“MOTIF: A Malware Reference Dataset with Ground Truth Family Labels"
Computers & Security 2023 – Joyce et al.

Largest dataset with ground-truth (3095 samples!)

Summary

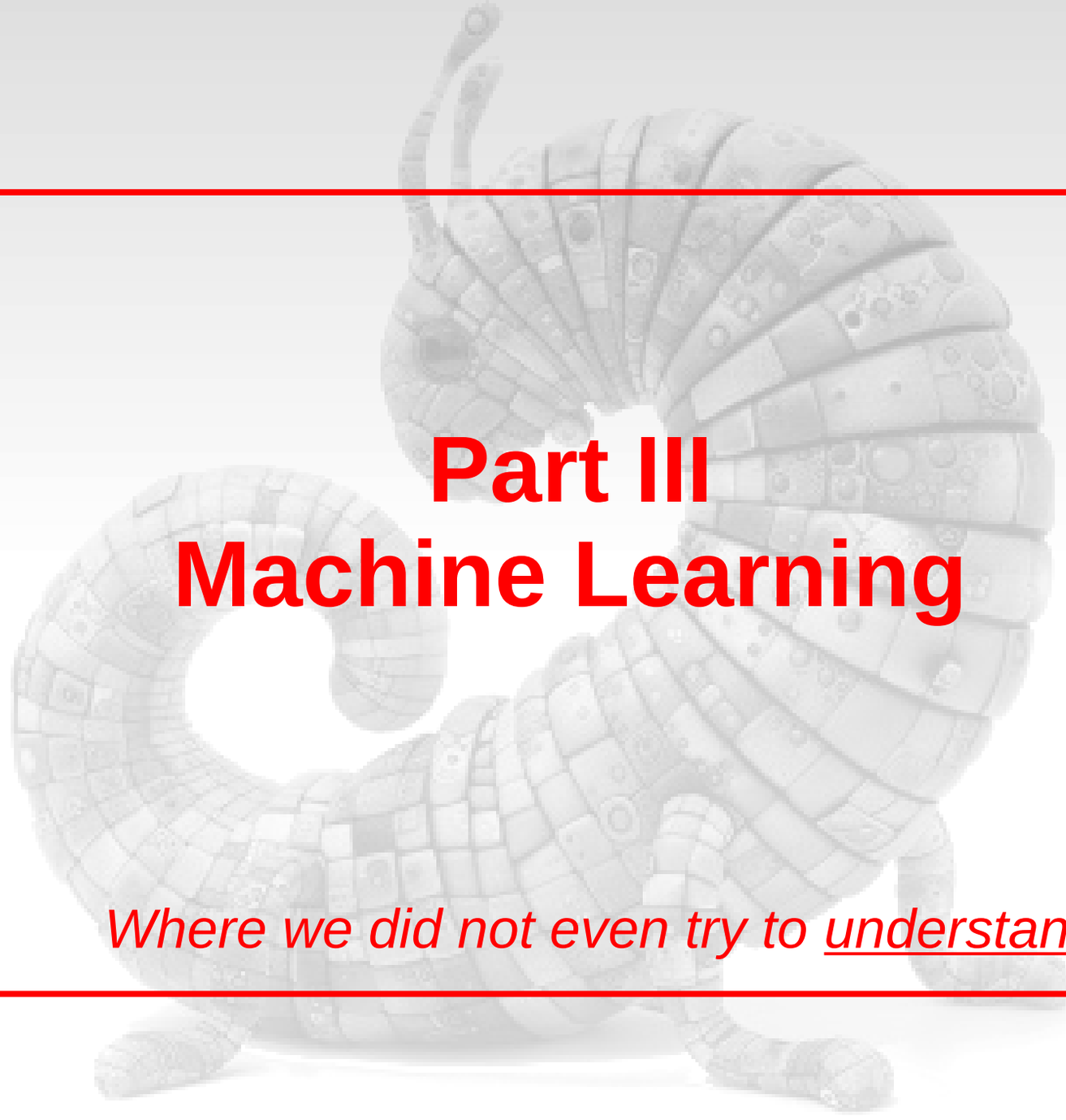
After trying every possible model on every possible set of features *(always with good results ?!)* we finally agreed that models based on static analysis are ineffective against malware.

On the other hand, dynamic analysis is very costly and not without problems

We identified some pitfalls to avoid, but overall we still do not know how to properly test and compare malware detectors.

Changing 1000 bytes is not harder than changing 1.
So, is evasion a binary property or something we can put on a scale? No idea.

Despite what **every paper introduction** says, static signature are alive and well.



Part III
Machine Learning

Where we did not even try to understand

“ Biologically inspired defenses against computer viruses"
IJCAI 1995 – Kephar, Sorkin,, Arnold, Chess, Tesauro, White

*Ngrams feeded to a single-layer **Neural Network***
trained to detect Boot sector Viruses

“ Biologically inspired defenses against computer viruses"
IJCAI 1995 – Kephar, Sorkin,, Arnold, Chess, Tesauro, White

Ngrams feeded to a single-layer ***Neural Network***
trained to detect Boot sector Viruses

“Automatically Generated Win32 Heuristic Virus Detection"
Virus Bulletin 2000 – Arnold, Tesauro

Neural Network for PE files
Multiple NN trained on different features
(3- and 4-grams present in Viruses but not benign).
Voting procedure: Virus iff >=2 networks say so.

With a small feature set of 16–32 features, automated training will work, without overfitting, if there are a larger number of viruses and clean files which contain the features. Multi-layer networks would require substantially more viruses. For a network with 1 hidden layer, the number of parameters is approximately number of features times number of hidden units. For a small network with 32 features and 4 hidden units, the number of viruses would need to be roughly 128 or larger.

Results so far indicate that detectors generated by this procedure are not sufficient to serve as the sole Win32 virus detection heuristics in a scanner. They do appear to be good enough to be very useful as an augmentation of expert-designed heuristics, and they would serve as sole heuristics were nothing else available.

“Data mining methods for detection of new malicious executables"
IEEE Security & Privacy 2001 – Schultz, Eskin, Zadok, Stolfo

Three Approaches:

1. A rule-based learner that generates heuristics based on DLLs, APIs, and number of APIs invoked per DLL

2. Naive Bayes on strings

3. Multi-Naive Bayes on bytes 2-grams

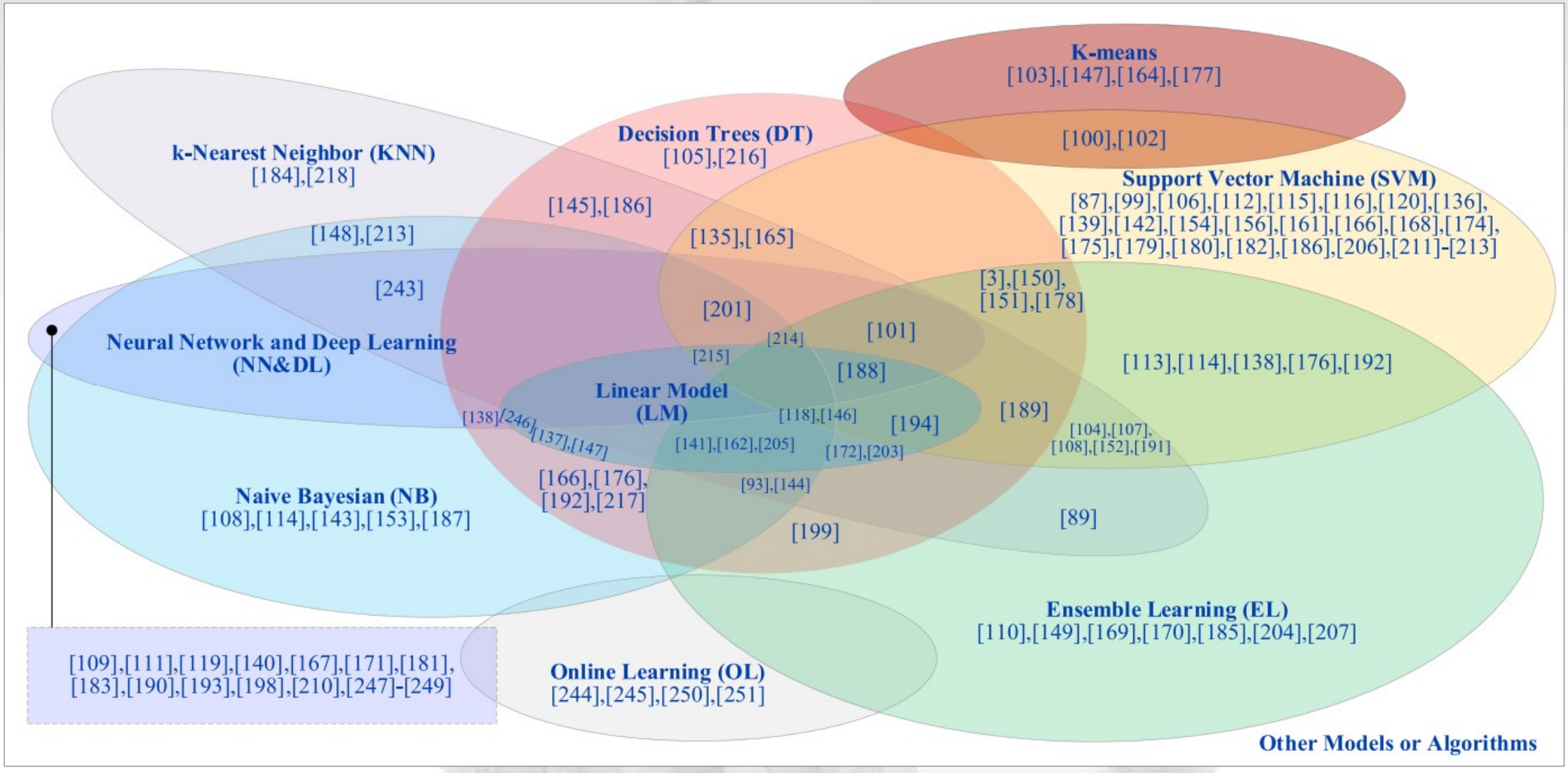
#2 and #3 performed much better (accuracy ~97%) but false positives were high (3.8-6%)

“Learning to detect malicious executables in the wild"
SIGKDD 2004 – Kolter & Maloof

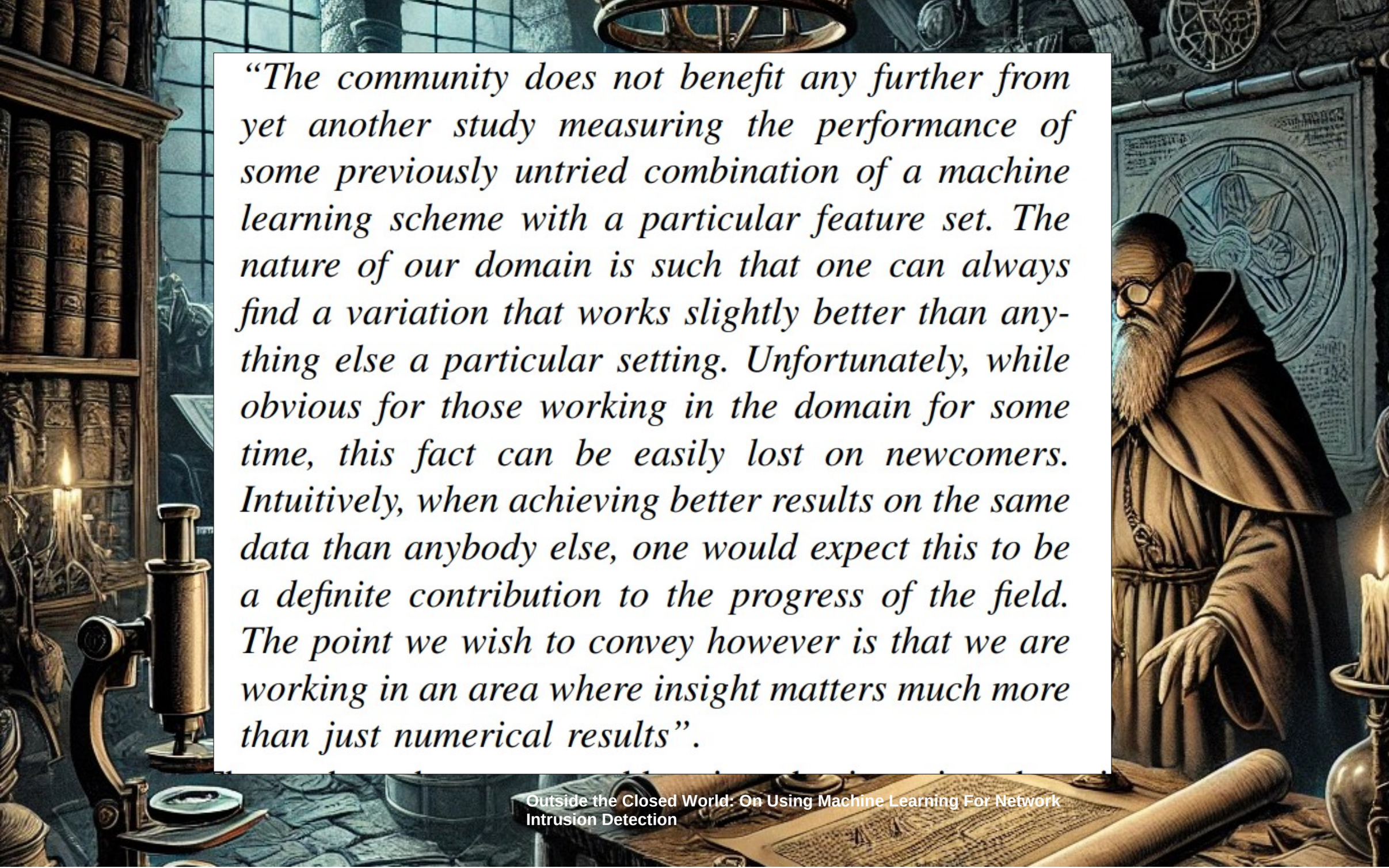
4-grams only, but experimented also with decision trees, support vector machines, and boosting.

0.996 AUC

The ML Cowboys



Review of Android Malware Detection Approaches Based on Machine Learning -K. Liu et al. 2020


“The community does not benefit any further from yet another study measuring the performance of some previously untried combination of a machine learning scheme with a particular feature set. The nature of our domain is such that one can always find a variation that works slightly better than anything else a particular setting. Unfortunately, while obvious for those working in the domain for some time, this fact can be easily lost on newcomers. Intuitively, when achieving better results on the same data than anybody else, one would expect this to be a definite contribution to the progress of the field. The point we wish to convey however is that we are working in an area where insight matters much more than just numerical results”.
Outside the Closed World: On Using Machine Learning For Network Intrusion Detection

We studied what we did wrong

Adversarial ML

We tried to understand
What & Why ML Learns

Phase 3.1

“Are Your Training Datasets Yet Relevant?"
ESSoS 2015 – Allix, Bissyandé, Klein, Le Traon

Temporal Sample Consistency

“Reviewer integration and performance measurement for malware detection". DIMVA 2016 – Miller et al.

Temporal Label Consistency

“Transcend: Detecting Concept Drift in Malware Classification Models" – USENIX Security 2017
Jordaney et al.

Concept Drift

“TESSERACT: Eliminating experimental bias in malware classification across space and time" - USENIX Security Symposium 2019

Recommendations

“Adversarial examples for malware detection"
ESORICS 2017 – Grosse, Papernot, Manoharan, Backes, McDaniel

1. Show that existing models are vulnerable to adversarial samples

2 Apply to Malware two popular approaches used in computer vision: Distillation and **Adversarial Training**.

Authors	Attack Knowledge	Feature	Attack Algorithm	Manipulation
Biggio *et al.* [33]	white-box	keywords frequency	gradient descent attacks	insert new keywords
Grosse *et al.* [16]	white-box	Android manifest, code features	JSMA attack	add new features
Al-Dujaili *et al.* [34]	white-box	API calls	FGSM and BGA attack	add new features
Kolosnjaji *et al.* [118]	white-box	raw bytes	embedding space projection, gradient-based optimization	inject or append bytes
Kreuk *et al.* [35]	white-box	raw bytes	FGSM attack	inject or append bytes
Li *et al.* [52]	white-box and black-box	Android manifest, code features	iterative Max strategy	increase or remove features
Abusnaina *et al.* [119]	white-box and black-box	IoT CFGs	GEA, FGSM, PGD, DeepFool	GEA combines CFGs of benign and malicious CFGs
Chen *et al.* [120]	white-box	API calls	EvnAttack framework	API elimination and addition with limited evasion cost
Hu *et al.* [36]	black-box	API calls	MalGAN with generator and substitute detector	add irrelevant APIs
Yuan *et al.* [37]	black-box	raw bytes	GAPGAN with generator and discriminator	append adversarial payloads
Rosenberg *et al.* [38]	black-box	API calls	substitute model, FGSM	insert to random position
Hu *et al.* [121]	black-box	API calls	substitute RNN model	add irrelevant APIs
Khasawneh *et al.* [122]	black-box	hardware features	substitute model, reverse-engineering HMDs	block-level or function-level insertion
Rosenberg *et al.* [40]	black-box	API calls	backtracking algorithm, adaptive EA	insert to random position
Kucuk *et al.* [41]	black-box	opcodes, APIs, system calls	genetic algorithm using fitness score	inject basic blocks, modify operable APIs
Xu *et al.* [123]	black-box	PDF trees	evolutionary algorithm using fitness score	insert or remove elements of PDF trees
Anderson *et al.* [39]	black-box	raw bytes	reinforcement learning	take limited actions
Song *et al.* [42]	black-box	raw bytes	binary rewriter and action minimizer	take selected macro and micro actions
Song *et al.* [42]	black-box	raw bytes	RL-based MAB-Malware framework	take selected macro and micro actions
Demetrio *et al.* [17]	black-box	raw bytes	GAMMA framework with genetic optimization algorithm	appending or inserting extracted benign sections

Phase 3.3

“When Malware is Packing Heat"
NDSS 2020 - Aghakhani et al.

What ML learns in presence of Packing

“Humans vs. machines in malware classification"
USENIX 2023 – Aonzo et al.

What humans and ML do different

“Decoding the Secrets of Machine Learning in Windows Malware Classification" ACM CCS 2023 – Dambra et al.

Static vs Dynamic

“Drift Forensics of Malware Classifiers"
AISec 2023 – Chow et al.

What causes concept drift

LLMs

Publication

~~Curiosity~~-driven Phase

Can LLMs help with … understanding code ?
… identifying malware families?
… handling obfuscation?
… triaging malware?
… building signatures?
… analyzing dynamic reports?
… writing new malware?
… ???

How accurate are they?
How do they compare with humans?
What are the limitations?

Answers are obsolete before they are published

Results rarely generalize

We are trying to answer questions..
but we should focus on dataset/benchmark instead

What did we Learn ??

We tried every possible ML model on every possible set of features *(always with good results ?!)*, but then we learned that most experiments were wrong and biased.
We are slowing getting a grip on how to train & test classifiers in this area.

Poor ground truth (wrong labels) is a big problem.

Adversarial samples break everything…
Adversarial training makes everything “better” (?)

Static features and raw bytes work great do detect known malware.
Dynamic features generalize better and are more robust to concept drift, but perform worse on known malware.

AI and Malware? Too early to tell….

davide.balzarotti@eurecom.fr

@balzarot

http://s3.eurecom.fr/~balzarot

For people who are interested in nature, it is difficult to find a subject more fascinating than computer viruses.

Peter Szor – Virus Research and Defense