Program Anomaly Detection: Methodology and Practices

Xiaokui Shu

IBM Thomas J. Watson Research Center

Daphne Yao

Associate Professor of Computer Science

Virginia Tech

CCS Tutorial, October 2016

Drone Control Station Operating System

http://theweek.com/article/index/241237/ (2011)

From NBC news (2013)

http://nbcnews.tumblr.com/post/47882129464#.UzGICChfd38

Acknowledgments

Drs. Kui Xu
(Amazon)

Xiaokui Shu
(IBM Research)

Collaborators

Publications:

Global trace analysis
[1] X. Shu, D. Yao, N. Ramakrishnan. *ACM CCS '15 (Featured in Comm. of ACM)*
[2] X. Shu, D. Yao, N. Ramakrishnan, T. Jaeger. ACM TOPS (under review)

Program analysis in HMM
[3] K. Xu, D. Yao, B. Ryder, K. Tian. *IEEE CSF '15*

HMM with context
[4] K. Xu, K. Tian, D. Yao, B. Ryder. *IEEE DSN '16*

Unified Program Anomaly Detection Framework
[5] Shu, Yao, Ryder. RAID 2015

Outline of This Tutorial

Our Goal:

To encourage and enable anomaly detection research

What have been done?

History of program anomaly detection

Attack models

Approaches, pros and cons, connecting the dots……

What can you do? Apply anomaly detection to your work!

Typical workflow and tools, recipe

Some recent findings

Open problems

Hands-on activities

Slides will be made available online.

Anti-virus Scanning is the First Line of Defense

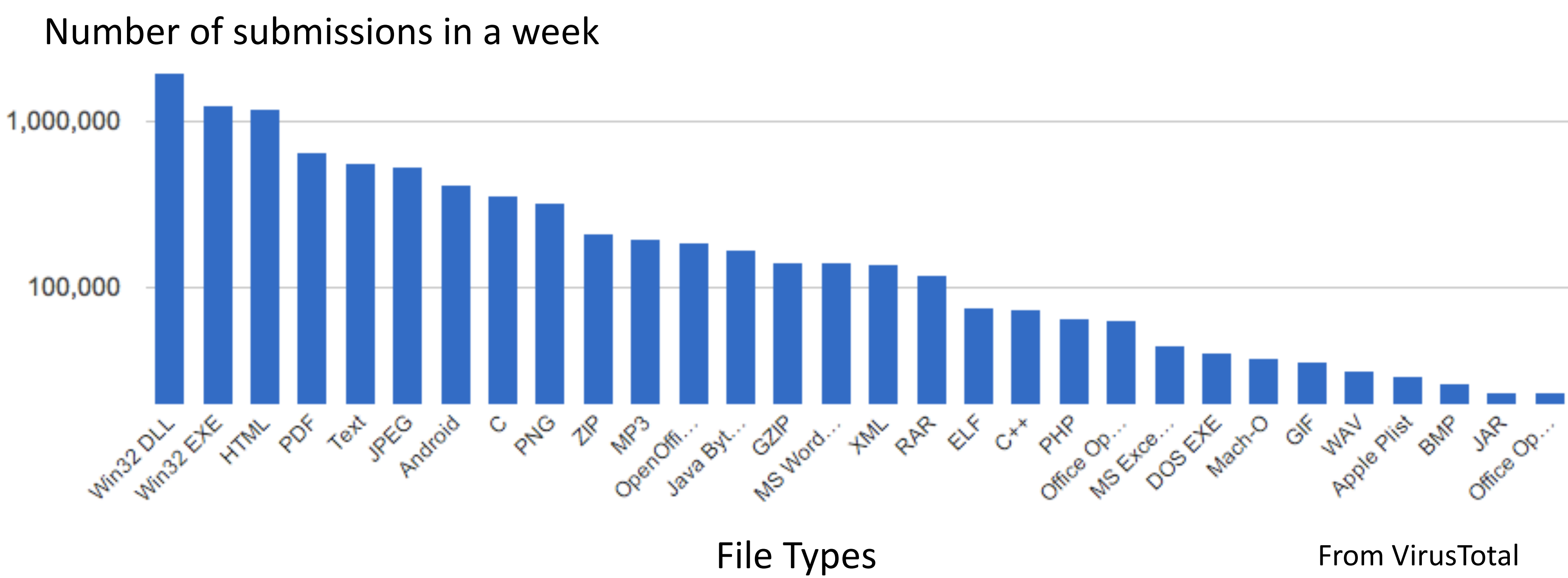
virustotal

For files (apps and PDFs), URLs

Vtzilla plugin

cuckoo

Cuckoo Sandbox for dynamic analysis



From VirusTotal

However, Code or Behavior Classification is Undecidable

1. Program X
2. main()
3. { …
4. **if ! isVirus(X)**
5. **then infect;**
7. **else goto next;**
8. … }
9. }

Scanner Thinks		Actual Behavior of X
IsVirus returns True	Contradicts ➡	X chooses not to infect
IsVirus returns False	Contradicts ➡	X chooses to infect

From [Fred Cohen, J. of Virology 1987]

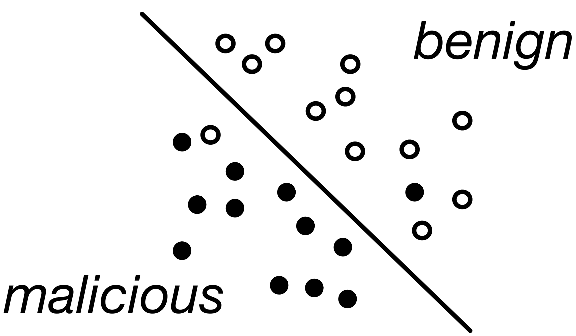
How to detect/prevent zero-day malware/exploits?

Moving target defense

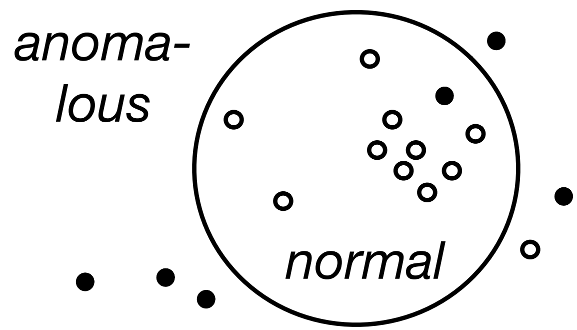
Verification

Control flow integrity

Anomaly-based detection (D. Denning '87, Forrest et al. '96)



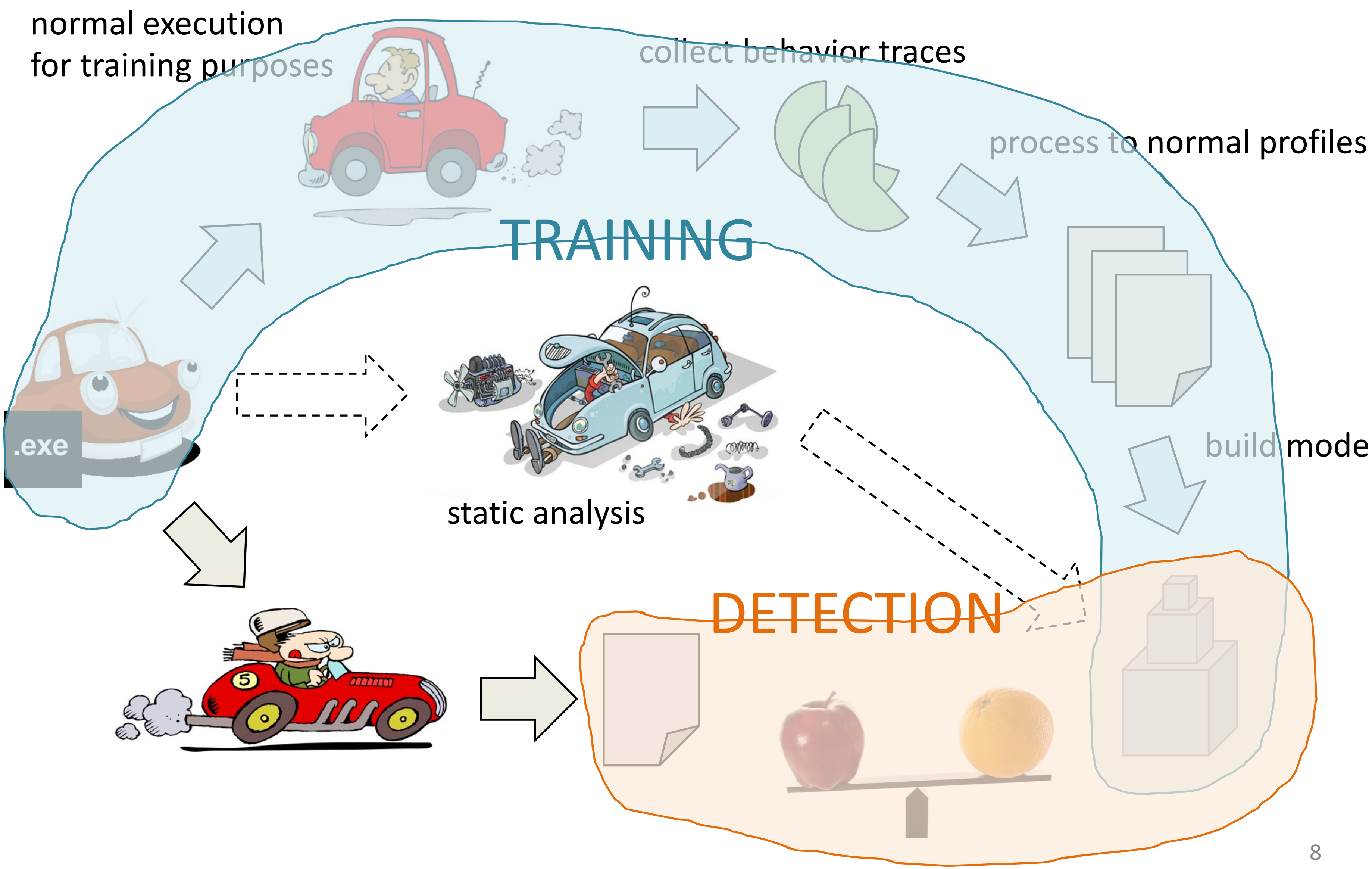
(a) Classification



(b) Anomaly detection

[Wressnegger 2013]

Typical Workflow



Simplest Program Anomaly Detection: n-gram

A 2-gram example:

ioctl()	open()
open()	read()
read()	setpgid()
setpgid()	setsid()
setsid()	fork()

1. From syscall traces of normal program executions (training data)

Runtime program trace

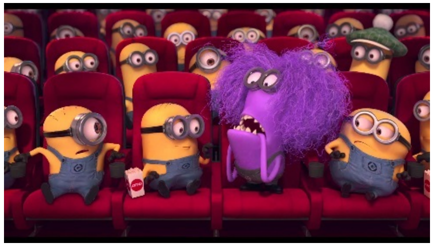
ioctl()
open()
write()
read()
setpgid()
setsid()
fork()

2. Test data

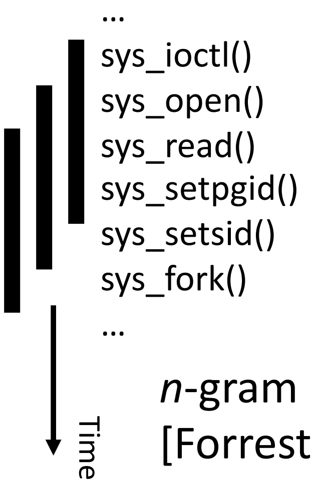
	Found in DB?
Ioctl(), open()	✓
open(), **write()**	✗
write(), read()	✗
read(), setpgid()	✓
……	

3. Classification

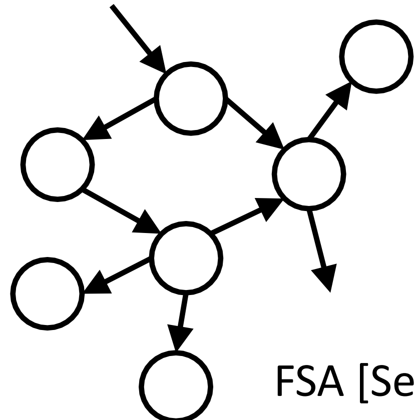
[Forrest 1996, Wressnegger 2013]



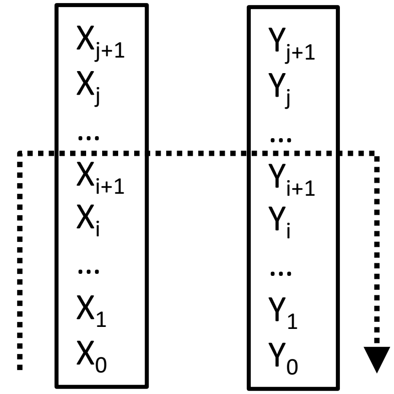
Existing Approaches



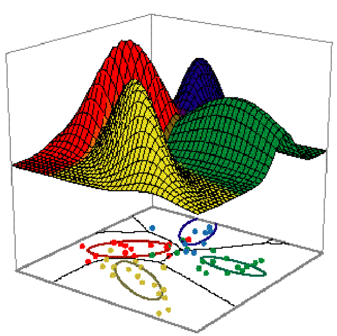
n-gram
[Forrest 1996]



FSA [Sekar 2001, Wagner 2001]



PDA [Feng 2003, Feng 2004, Giffin 2004]



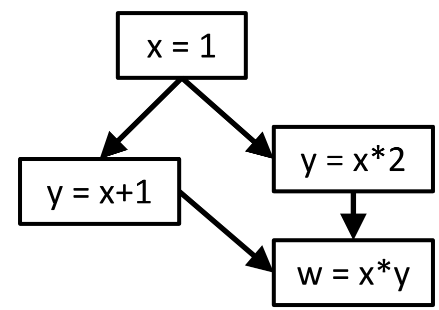
Machine learning [Lee 1998, Shu 2015, Xu 2016]

Static Program Analysis

+

Dynamic Program Analysis

Hybrid detection
[Liu 2005, Xu 2015]



Data-flow analysis [Giffin 2006, Bhatkar 2006]

Existing Approaches (Categories)

Data-driven

Dynamic learning

- [Forrest 1996]
- [Kosoresow 1997]
- [Lee 1998]
- [Sekar 2001]
- [Feng 2003]
- [Gao 2004]
- [Shu 2015]

Language-driven

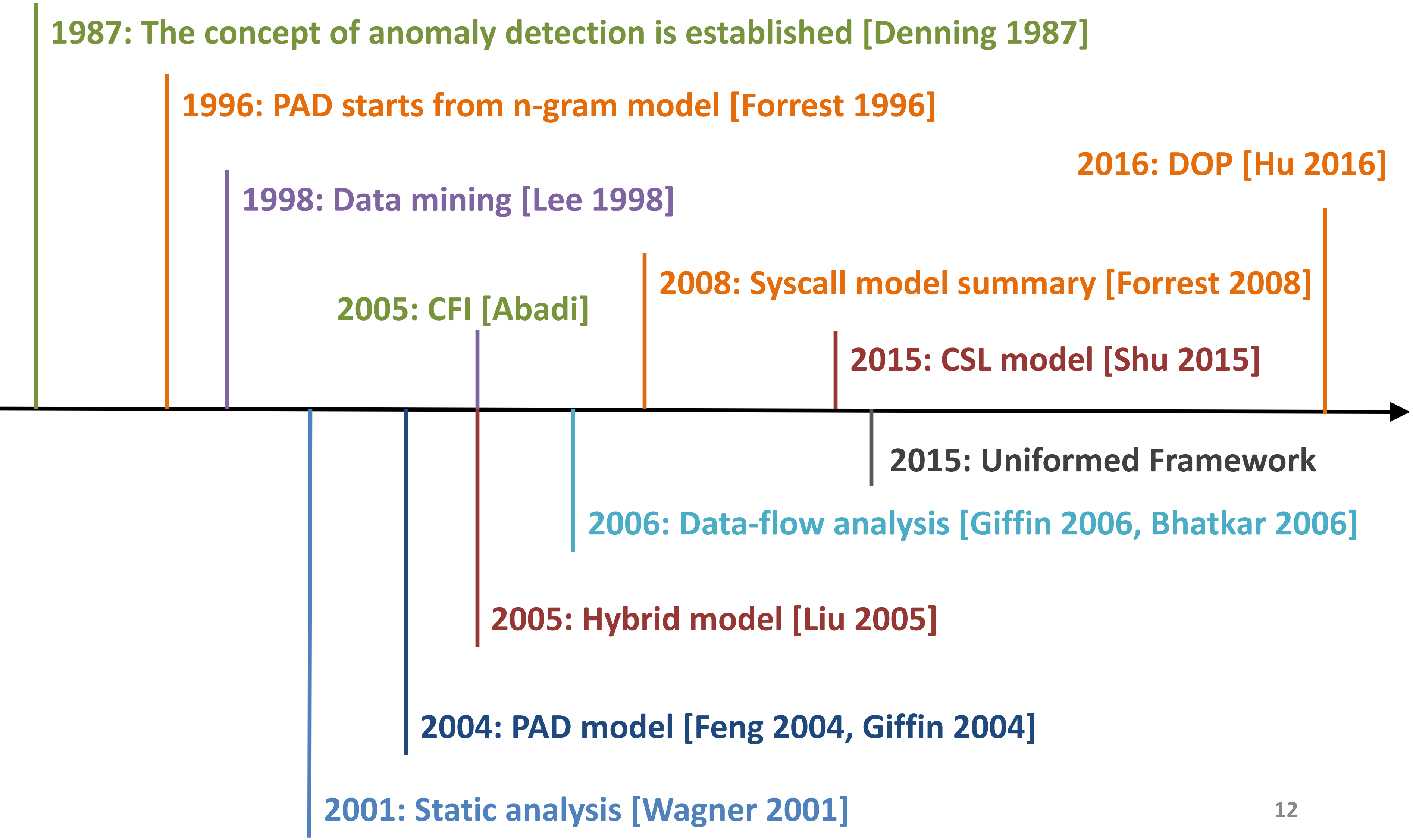
Static program analysis

- [Wagner 2001]
- [Feng 2004]
- [Giffin 2004]
- [Giffin 2006]
- [Bhatkar 2006]

Hybrid

- [Liu 2005]
- [Xu 2015]
- [Xu 2016]

Notable Milestones



How Can I Start? Relevant Tools

- **Tracing**
  - Strace, SystemTap (system call level)
  - PIN (function level), used by BAP (binary analysis platform)
  - Intel PT (hardware-assisted instruction tracing)
  - gdb
- **Program analysis**
  - Wala
  - Paradyn/Dyninst, LLVM
- **Machine learning**
  - Dimension reduction, binary classification, outlier detection
  - scikit-learn, LIBSVM, WEKA
- Datasets (DARPA Intrusion Detection Data Sets)

Who Uses Anomaly Detection?

- Average **$1.27million/year** on false alerts by an enterprise.
- **4%** of alerts are investigated, due to high false positives.
- An organization receives an average of **17,000 alerts/week**.

From [Ponemon Institute]

Issue 1: Incomplete Traces

Incomplete Traces

Incomplete Behavior Model

False Alarms
Missed Detection

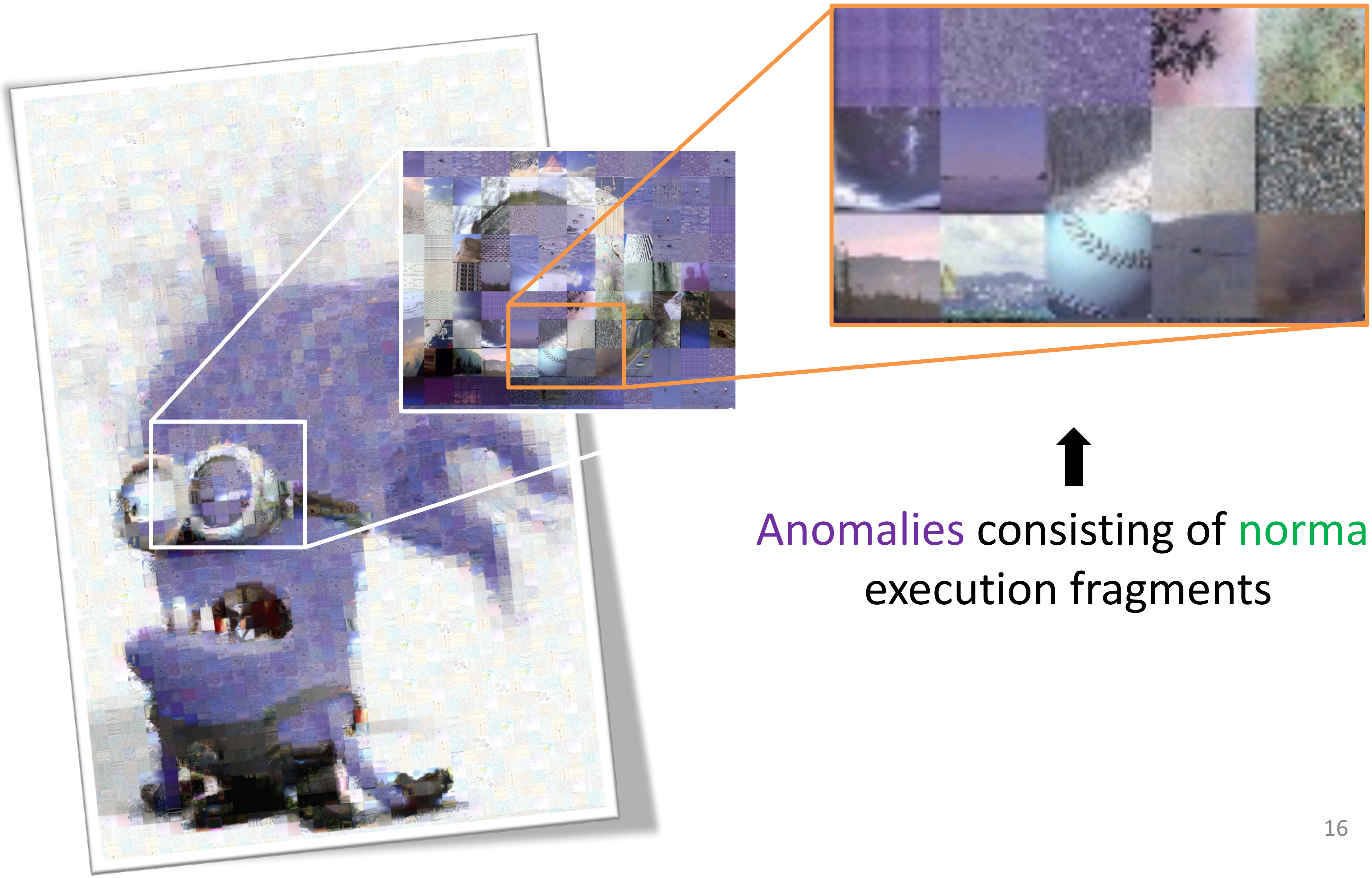
Program	# of test cases	branch coverage	line cov.
flex	525	81.34%	76.04%
grep	809	58.68%	63.34%
gzip	214	68.49%	66.85%
sed	370	72.31%	65.63%
bash	1061	66.26%	59.39%
vim	976	54.99%	51.93%

From SIR

By Shel Silverstein

Issue 2: Local Analysis

Local analysis is inadequate



Anomalies consisting of normal execution fragments

An SSH Authentication Attack

A SSHD flag variable overwritten attack

```
void do_authentication(...) {
  int authenticated = 0;
  while (!authenticated) {
  […buffer overflow vulnerability…]
    if (auth_password(...)) {
      memset(...);
      xfree(...);
      log_msg(...);
      authenticated = 1;
      break;
    }
    memset(...);
    xfree(...);
    debug(...);
    break;
    ...
  }
  if (authenticated) {
    ...
```

From [Chen ’05]

Pass auth.

Expected

Fail auth.

Expected

Attack

Local analysis cannot detect the anomaly

Attack Model, Problem Statement

Cooccurrence Anomaly

Normal 1: a b d a c e a

Normal 2: c b e a c c e c f

Normal 3: f d c e c c f e d

Anomaly: a b d a c c f e d

Attack examples:

- Non-control data attack
- Fragment-based mimicry attack
- Workflow violation attack

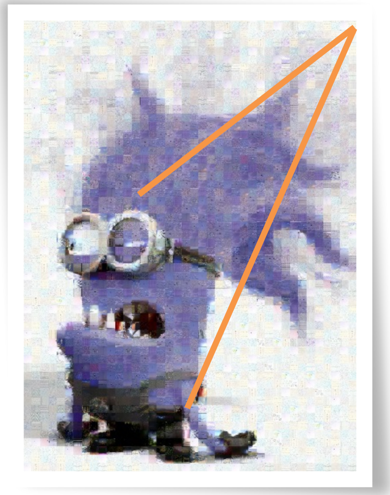
Frequency Anomaly

Attack examples:

- DoS attacks
- Directory harvest attacks

Problem Statement:

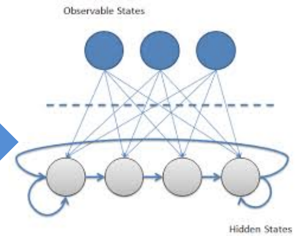
- Given an **extremely long trace**, should **any** set of events co-occur?
- With the expected **frequency**?



Can n-gram still work?

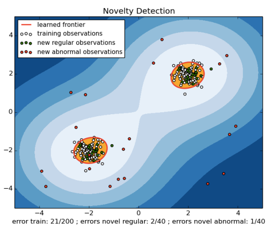
Our High-Precision Program Anomaly Detection

Security logs,
Network headers,
Traffic payloads,
System traces,
Transaction logs
…



Binary classification

Prog analysis



Novelty detection

Program Tracing (Library call, System call, Instruction sequences)

Program Analysis (static)

ML/DM (train and test)

Post Classification Analysis

Global Trace Analysis	**[1] X. Shu, D. Yao, N. Ramakrishnan. *ACM CCS '15***
HMM	**[2] K. Xu, D. Yao, B. Ryder, K. Tian. *IEEE CSF '15***
HMM with context	**[3] K. Xu, K. Tian, D. Yao, B. Ryder. *IEEE DSN '16***

Our Compact Matrix Representation

An infinite long call trace:

… main, foo, bar, bar, bar, …

chop into

Behavior instance

Long trace segments

Behavior instance

convert into

1. Transition frequency matrix

	main	foo	bar	goo
main	0	24	0	0
foo	0	0	30	0
bar	0	6	89	1
goo	0	0	0	0

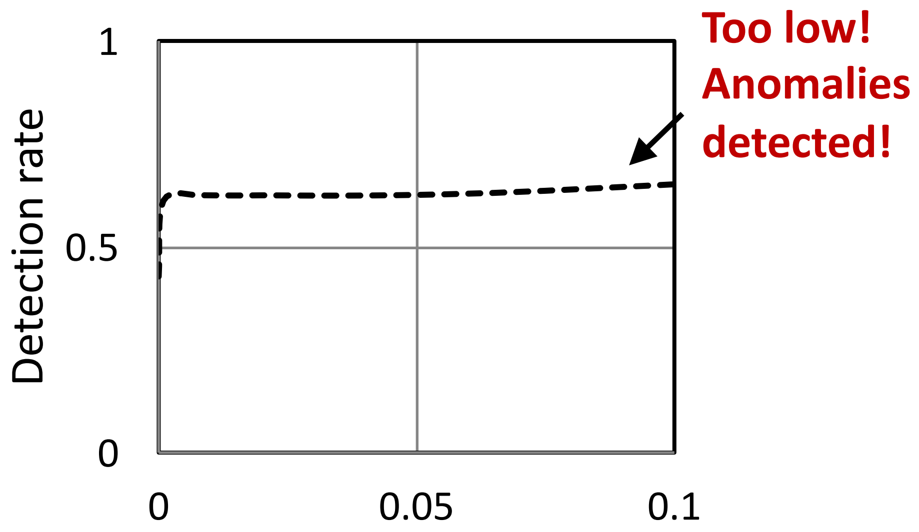
2. Event co-occurrence matrix

F	T	F	F
F	F	T	F
F	T	T	T
F	F	F	F

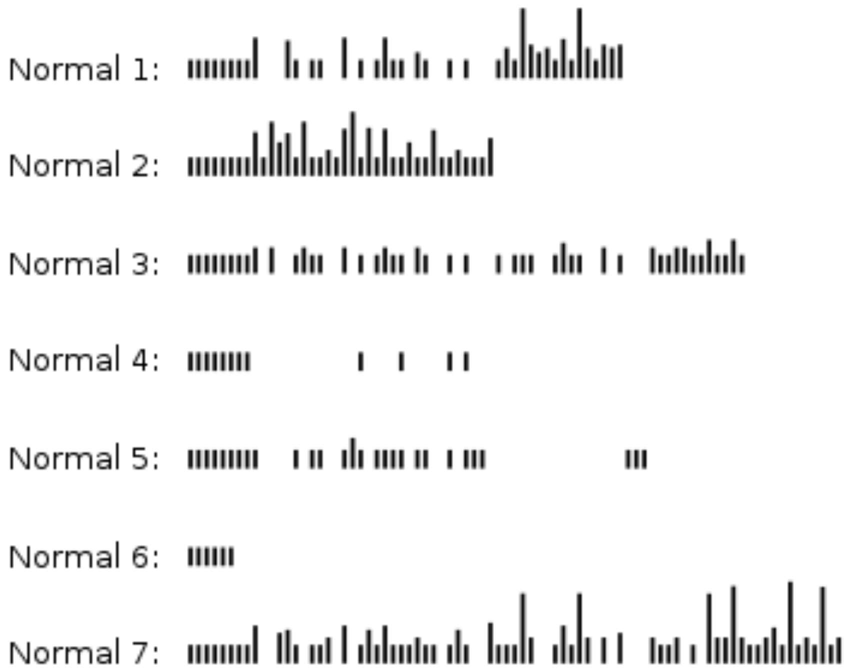
Matrix representation is path insensitive

X. Shu, D. Yao, N. Ramakrishnan. *ACM CCS '15*

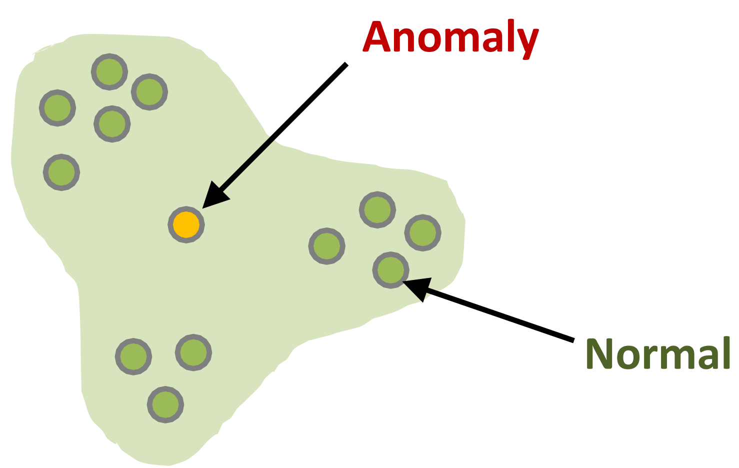
Challenges: Diverse Normal Behaviors, High FP



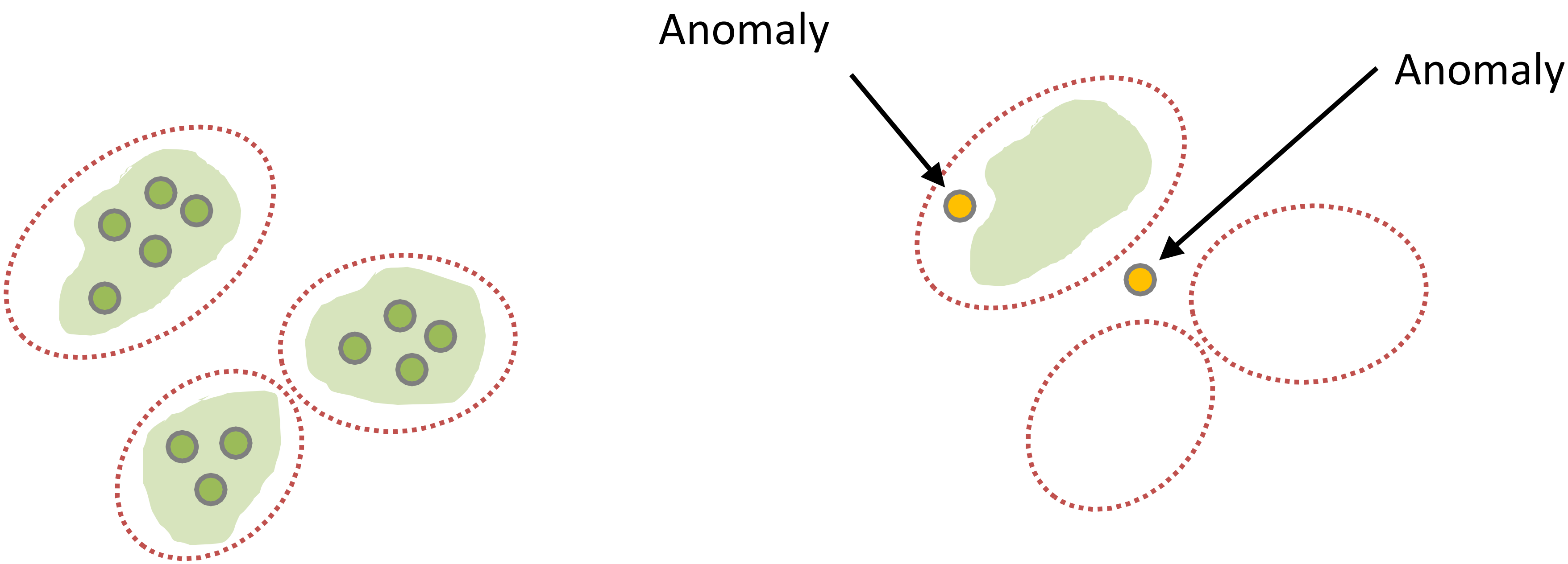
(1-class SVM on libpcre)



Distribution of function calls in libpcre



Our Solution: Grouping Similar Normal Behaviors



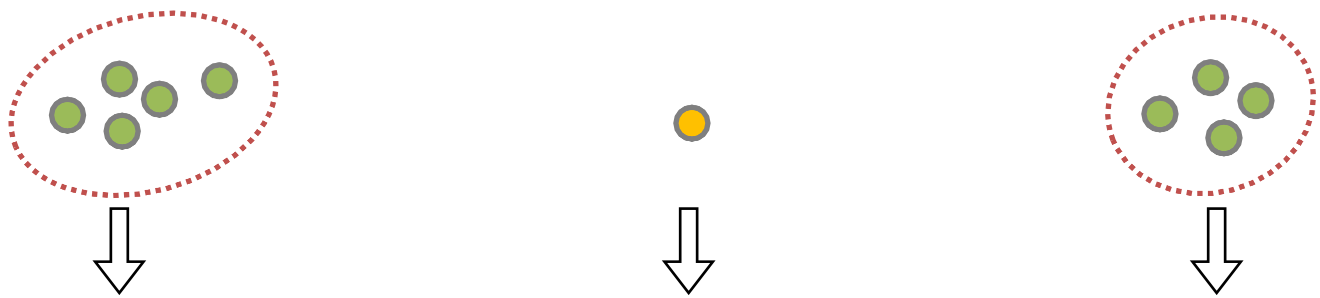
Training Phase

Detection Phase


A trace segment represented by matrices

Montage Anomalies Fall Between Clusters

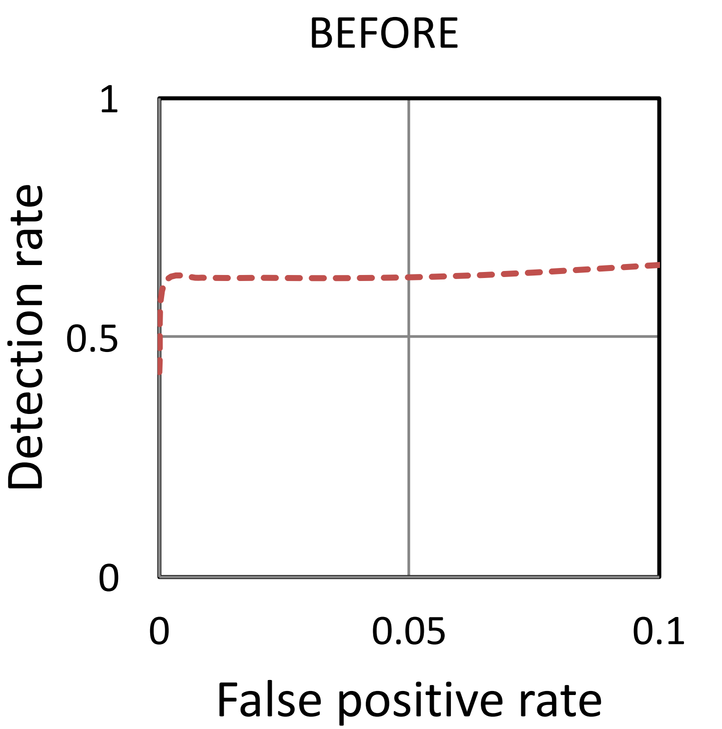
sshd



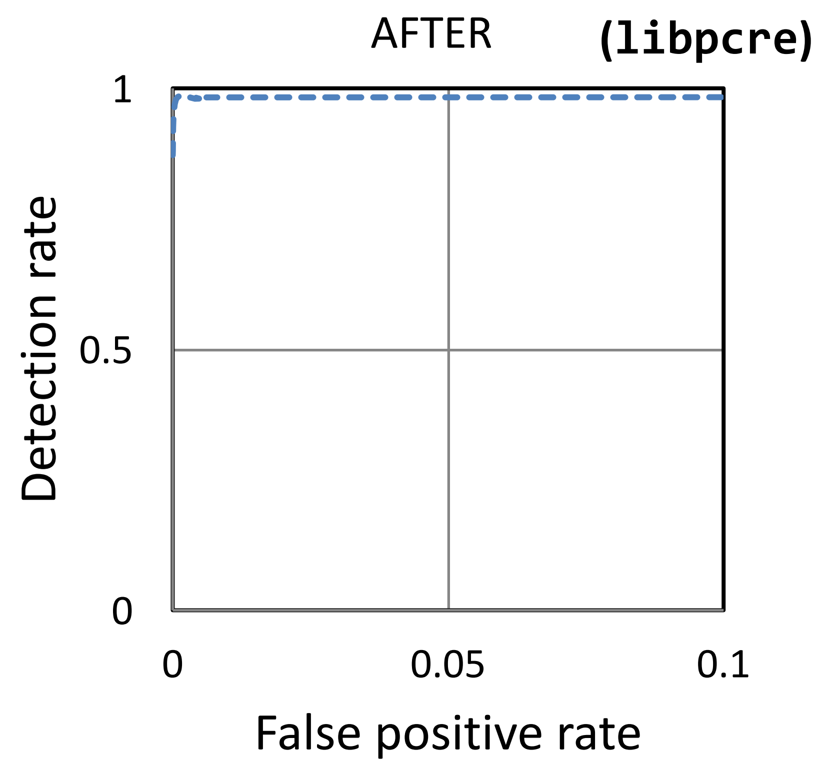
Pass Auth. (expected)	Anomalous: attack	Fail Auth. (expected)
…	…	…
do_auth > xfree	do_auth > debug	do_auth > debug
do_auth > log_msg	do_auth > xfree	do_auth > xfree
do_auth > packet_start	do_auth > packet_start	do_auth > packet_start
…	…	…
pwrite > buffer_len	pwrite > buffer_len	pwrite > buffer_len
do_auth > do_auth	do_auth > do_auth	do_auth > pread
…	…	…

Function call trace
(collected through Pintool)

Comparison of Detection Capabilities Against Montage Anomalies



1-class SVM (w/o clustering)

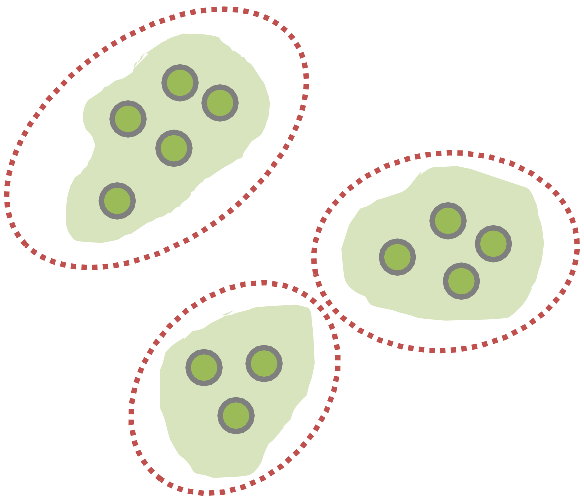


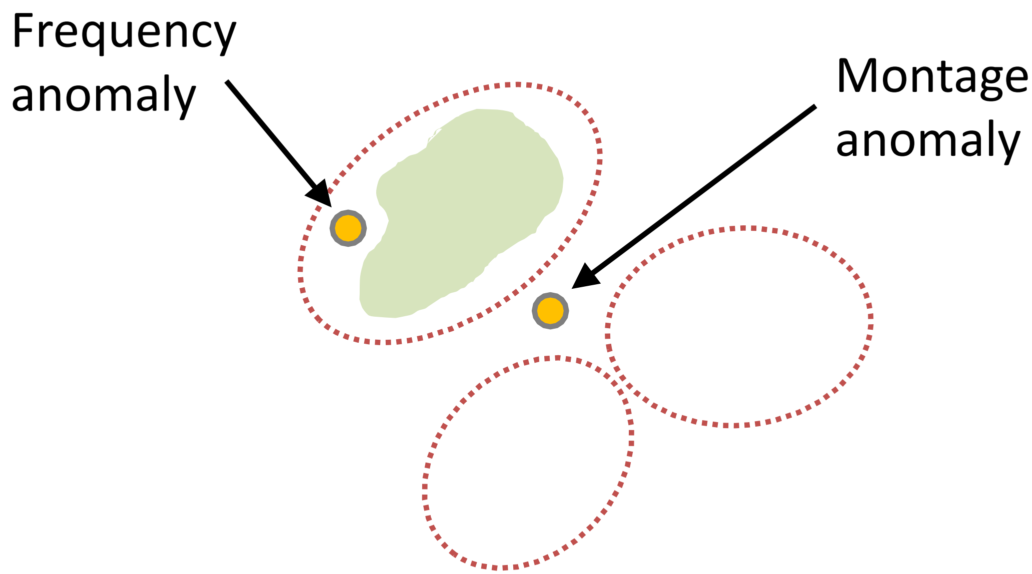
Ours (w/ clustering)

A specialized constrained agglomerative clustering algorithm
(on co-occurrence matrices)

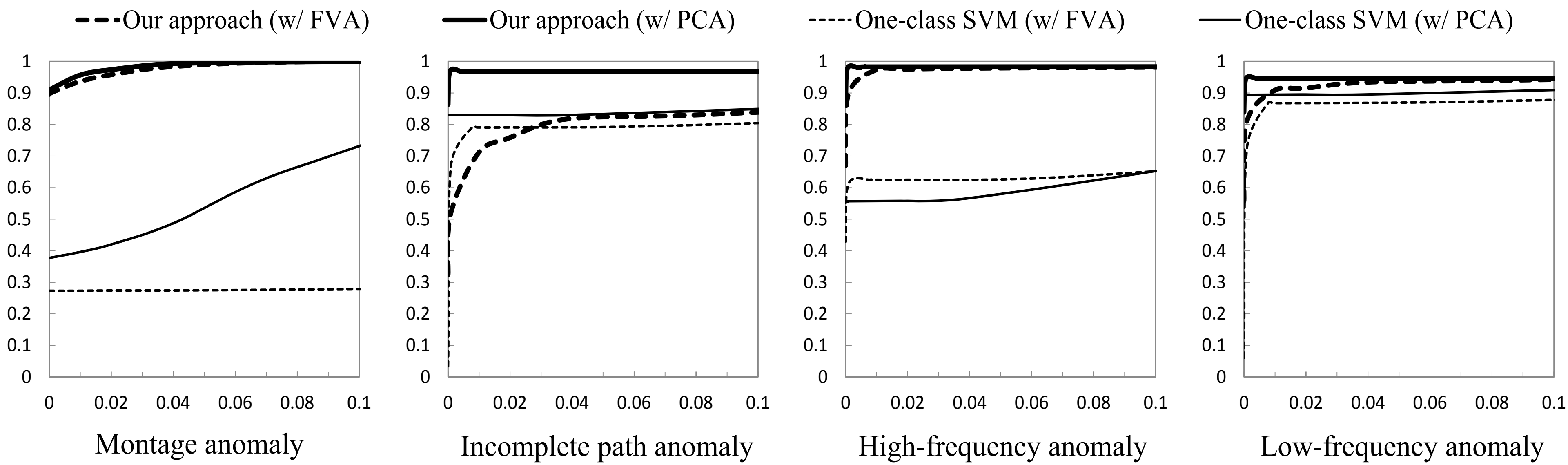
Our Operations

- Inter-cluster training
- Inter-cluster detection on co-occurrence matrices
- Intra-cluster training
- Intra-cluster detection on frequency matrices





Exp 1: Detection Accuracy vs. False Positive in Synthetic Anomalies



Frequency Anomalies

Under 10-fold cross-validation with 10,000 normal test cases,
1,000 synthetic anomalies.

Exp 2: Detection of Real-world Attacks in Complex Programs

sshd

Training w/
4,800 normal behavior instances (34K events each)

libpcre

Training w/
11,027 normal behavior instances (44K events each)

sendmail

Training w/
6,579 normal behavior instances (1K events each)

Flag variable overwritten attacks w/ various lengths

Regular Exp. DoS
3 malicious patterns
8-23 strings to match

Directory harvest attack w/ probing batch sizes:
8 to 400 emails

100% Detection accuracy
0.01% Average false alarm rate

What is the detection overhead?

Summary for Global Trace Analysis

Security Guarantees:

Detects 1. Co-occurrence anomalies 2. Frequency anomalies

Main Features:

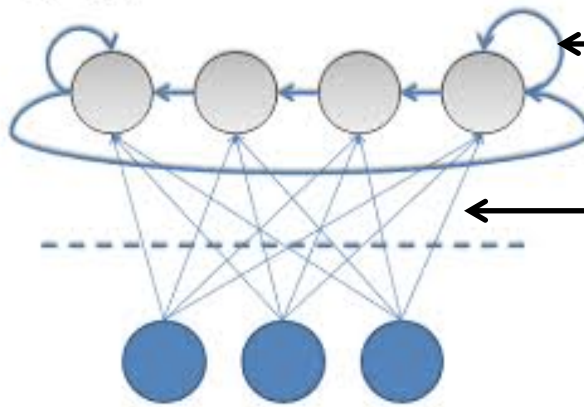
1. Extremely long traces 2. Low false alarm rate

Tradeoffs:

Path insensitive (orderless)

Hidden States

Call sequences
(training data)



Transition Probabilities

Emission Probabilities

Observable States

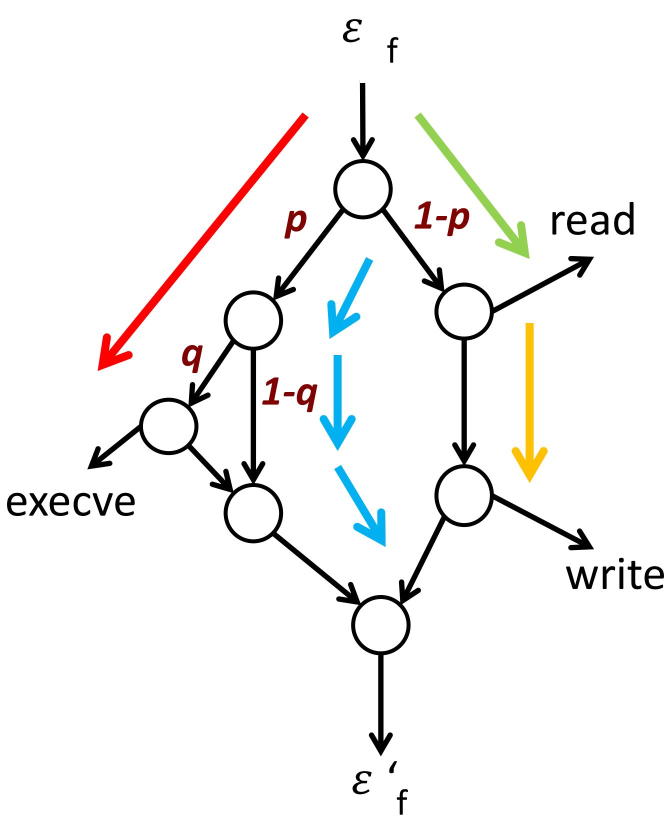
HMM-based program anomaly detection

- Probabilistic
- Path sensitive
- Local analysis

Want to be better than
random initialization

[Forrest et al. 1999]

STILO: STatically InitiaLized markOv



Function: f

Transition probability of a call pair is its likelihood of occurrence during the execution of the function

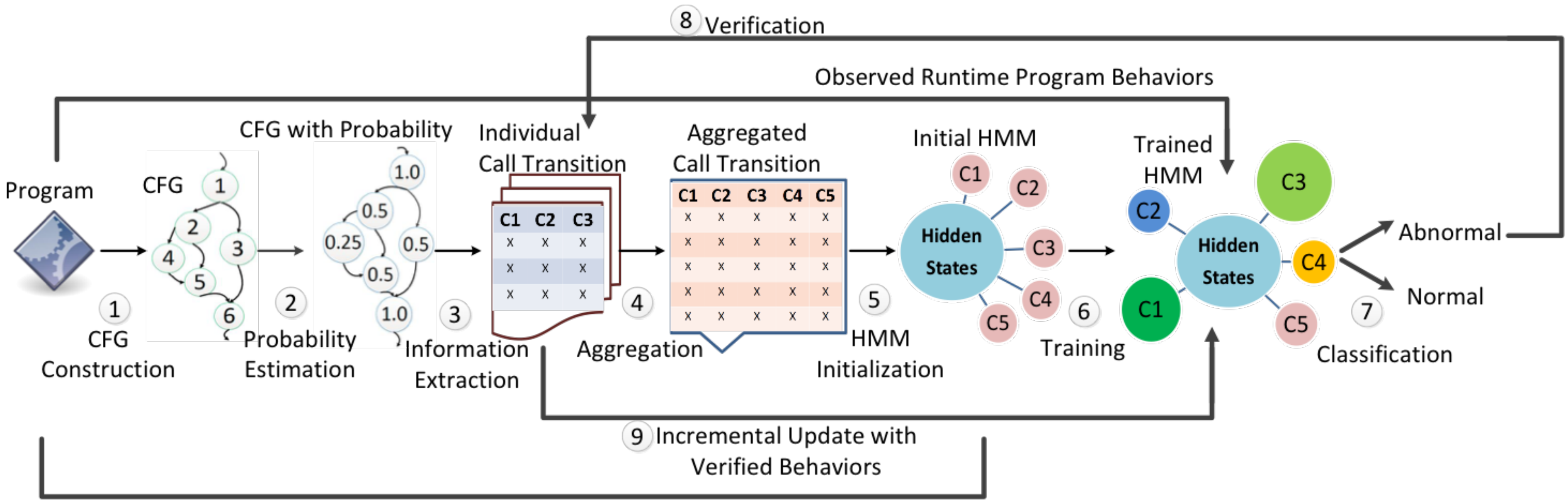
Example of call pair	Transition probability
read ⟶ write	1-p
read ⟶ read	0
execve ⟶ ε_f'	pq

	ε_f'(exit)	read	write	execve
ε_f (entry)	$\boxed{p(1-q)}$	$\boxed{1-p}$	0	$\boxed{pq}$
read	0	0	$\boxed{1-p}$	0
write	1-p	0	0	0
execve	pq	0	0	0

p, q are statically estimated.

[K. Xu, D. Yao, B. Ryder, K. Tian. *IEEE CSF* ’15]

Our STILO Workflow



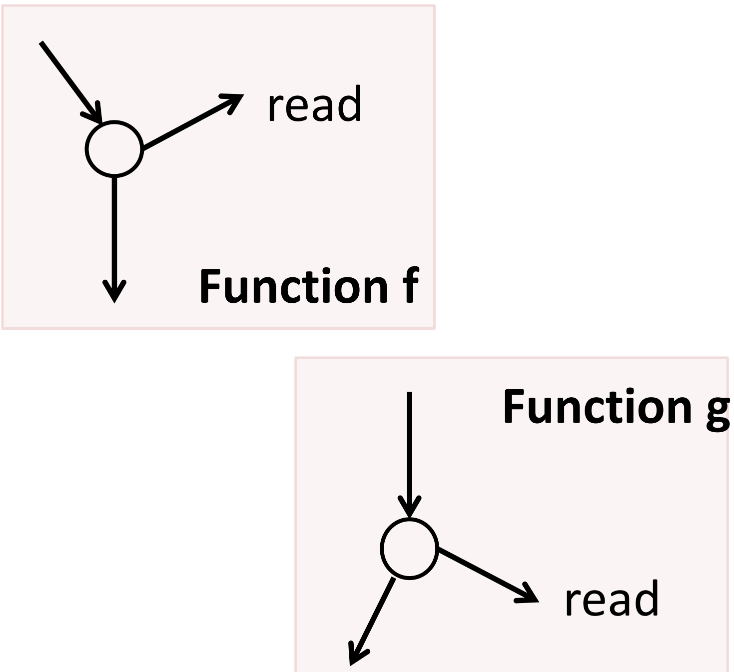
Static Program Analysis based HMM Initialization (Our New Contributions)

Improvement with Context Sensitivity

Why need context sensitive detection?

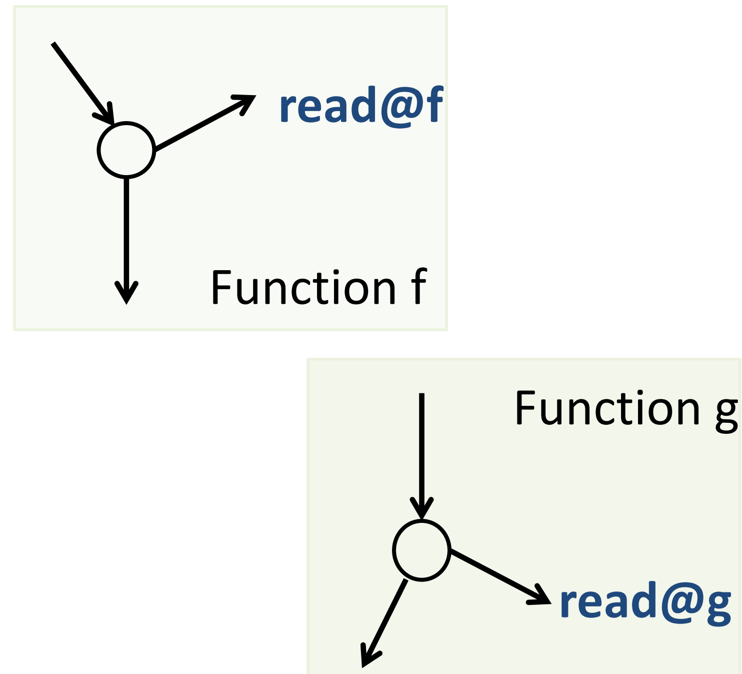
Improvement with Context Sensitivity

BEFORE: Context insensitive
(STILO-basic)



... read .... read ....

AFTER: 1-level calling context sensitive
(STILO-context)



... **read@f** .... **read@g** ....

Scalability:
K-mean clustering reduces the # of hidden states

[K. Xu, K. Tian, D. Yao, B. Ryder. *IEEE DSN* '16]

Reduction of Hidden States for Efficiency

Before clustering

One-to-one mapping -- a hidden state represents a single call

After clustering

Many-to-one mapping -- a hidden state may represent multiple similar calls

Program Model	# distinct calls	# states after clustering	Estimated training time reduction
bash	1366	455	88.91%
vim	829	415	74.94%
proftpd	1115	372	88.87%

- K-mean clustering, based on similarity between call-transition vectors
- Aim at 1/2 to 1/3 reduction of nodes

STILO Evaluation

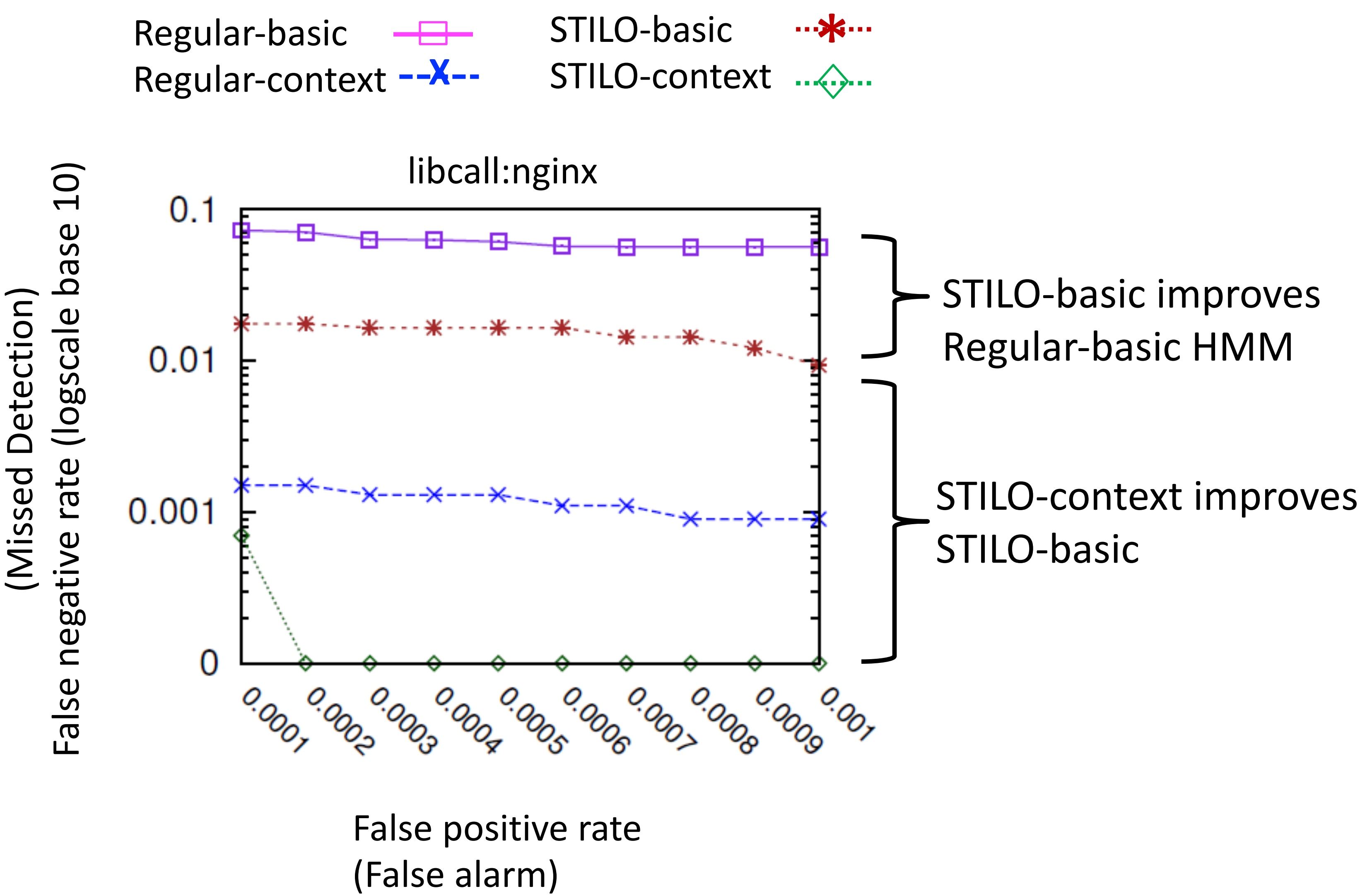
Model	With Static Analysis	With Caller Context
Regular-basic	-	-
Regular-context	-	Yes
STILO-basic	Yes	-
STILO-context	Yes	Yes

2 Linux server programs: nginx, proftpd
6 Linux utility programs: flex, grep, gzip, sed, bash, vim

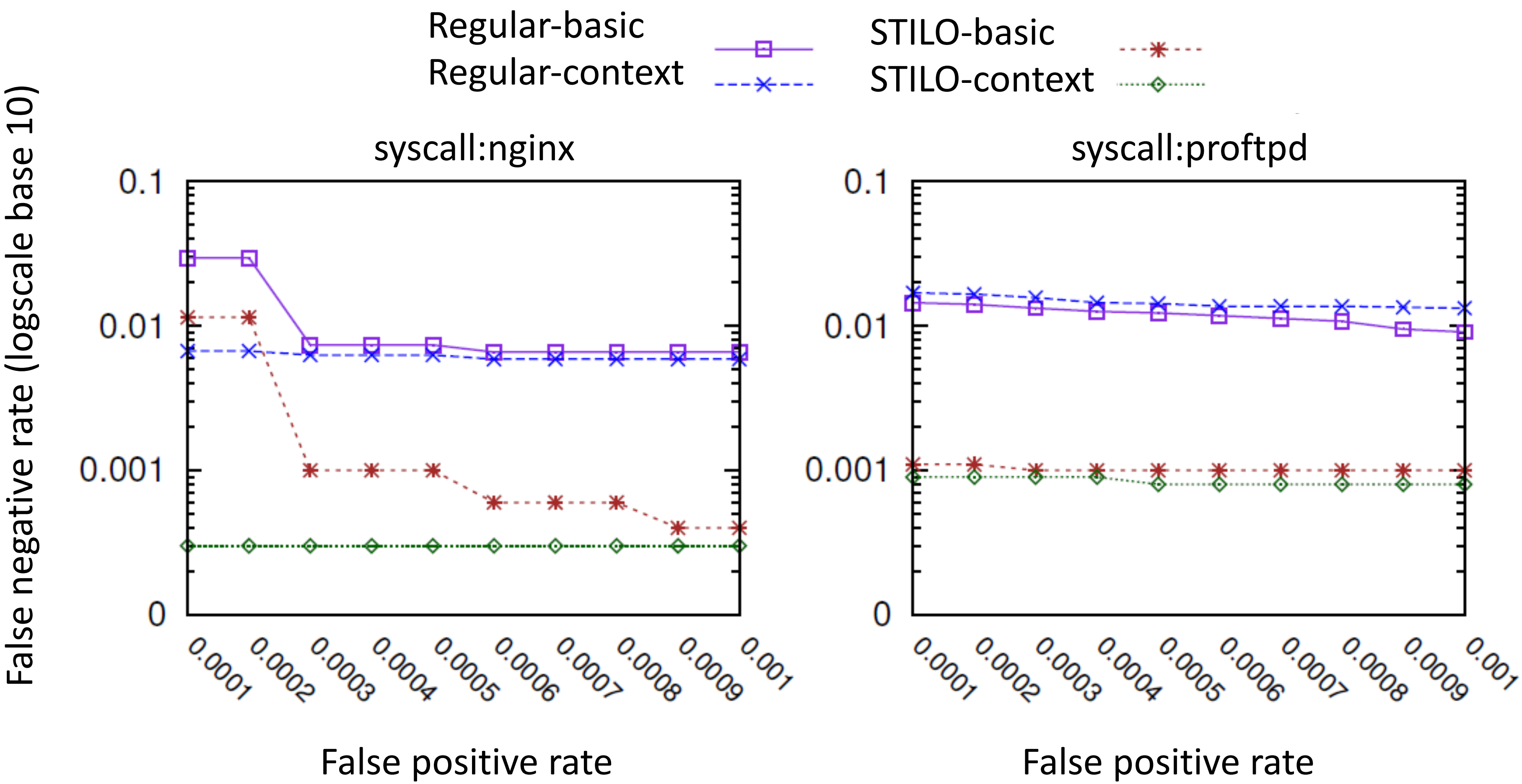
1. **Normal:** total 130,940,213 segments
2. **Abnormal-S:** 160,000 Abnormal-S segments (permute 1/3 calls)
3. **Abnormal-A:** attack call sequences obtained from exploits

Dynist for static program analysis, Jahmm library for HMM, 1st-order Markov, strace/ltrace for collection, SIR for test cases, 10-fold cross validation, 15-grams from traces

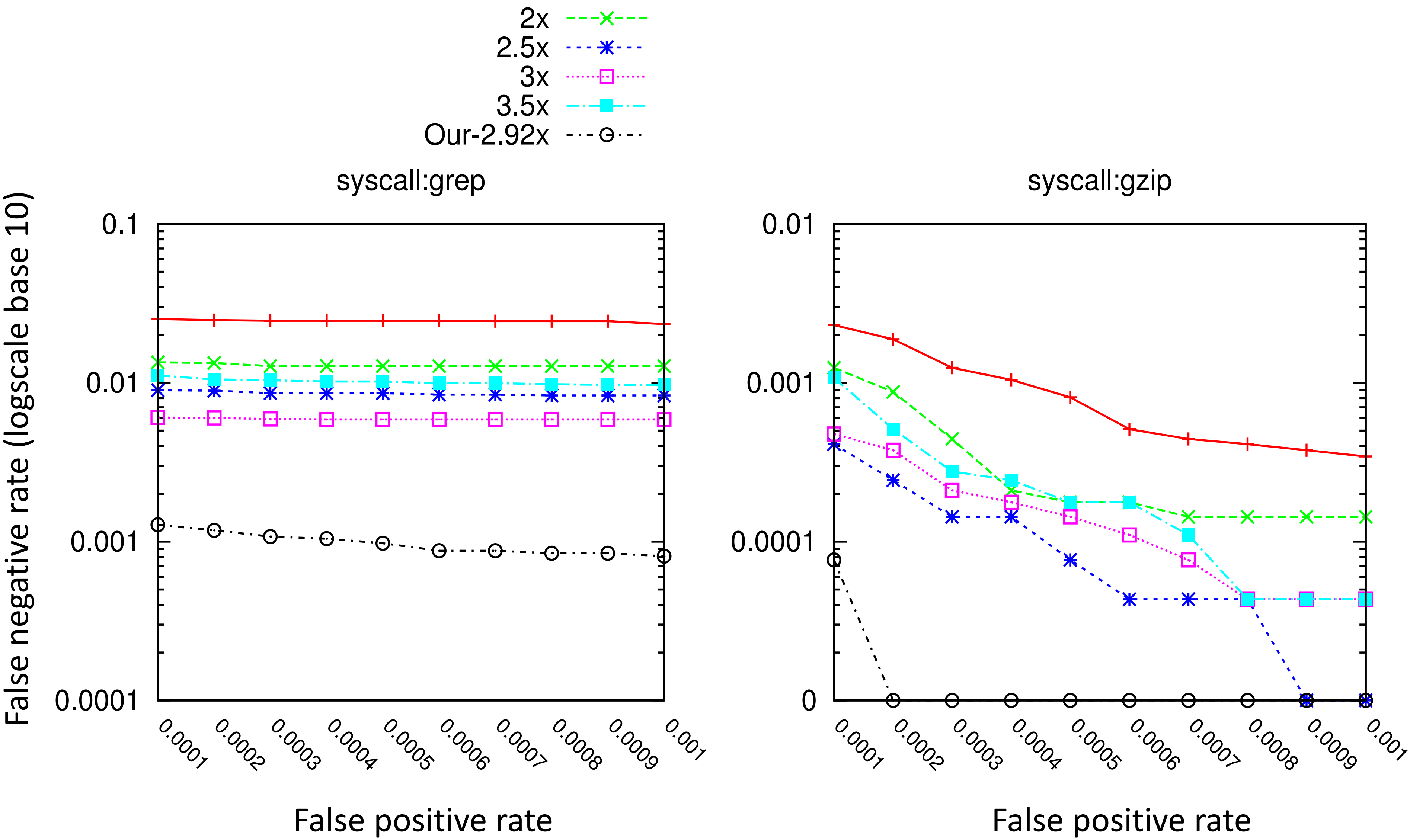
For libcalls, false negative (missed detection) of context-sensitive models drops by 2-3 orders



For syscalls, context improves false negative rate by 10 folds.
Less dramatic improvement than libcalls.



Increasing hidden states in regular HMM does not guarantee classification accuracy



Detection of Real-world Attacks

ROP attack segments against gzip (syscalls)

ID	Prob in STILO	Prob in Regular HMM
S_1	0	0.2
S_2	$2.20 \times e^{-15}$	0.29
S_3	$1.54 \times e^{-5}$	0.25
S_4	0	0.27
S_5	0.0005	0.33
S_6	0	0.23
S_7	0.0004	0.26

STILO gives much lower probabilities for attack sequences

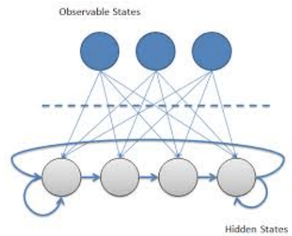
Exploit	Payload
Buffer Overflow (gzip)	ROP
	ROP_syscall_chain
Backdoor (proftpd)	bind_perl
	bind perl ipv6
	generic cmd execution
	double reverse TCP
	reverse_perl
	reverse_perl_ssl
	reverse_ssl_double_telnet
Buffer Overflow (proftpd)	guess memory address

STILO Overhead

New traces

Tracing (data collection)

Probability forecast



HMM initialization & training

HMM classification

Sloooow

Fast program analysis < 10s

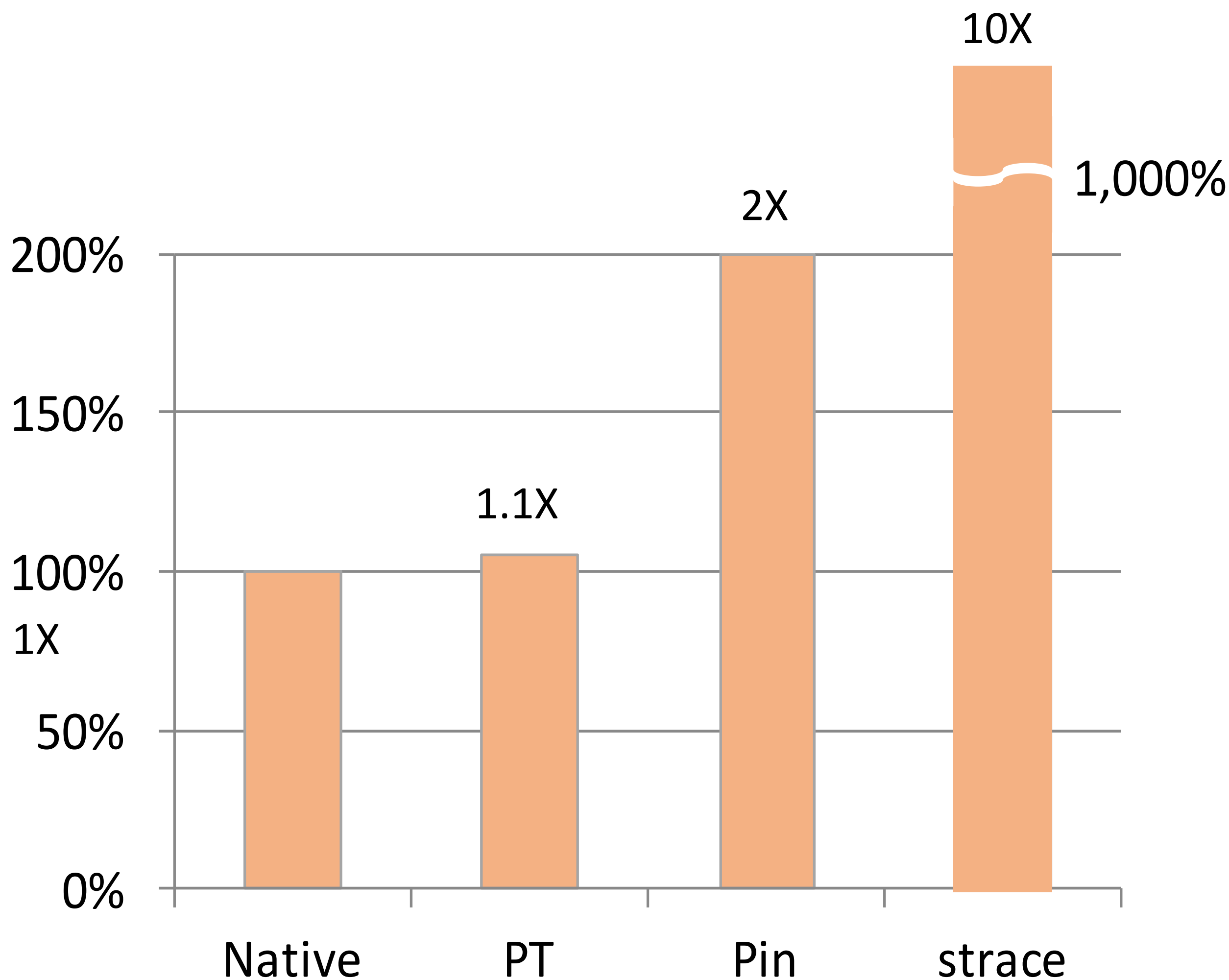
Aggregating matrices is slower. E.g., vim libcall took 20 min

HMM Training took days for STILO-context

K-mean clustering reduces hidden states

Fast classification < 1 ms

Hardware-based Instruction-level Tracing



Security/Privacy as Enablers

http://resources.infosecinstitute.com

RasPilot

Intelligent secure systems benefiting large populations

Enable new infrastructures

Improve quality of life

Enable new discoveries

Data-driven Program Anomaly Detection: Promising Directions

CPS and IoT (drones, cars)

Tracing overhead, HPC training and incremental training

Post-detection procedure

Program Anomaly Detection Workflow

Purification of training data, Adversarial machine learning

Order-aware global trace analysis

Program Anomaly Detection Labs

Lab Scripts and Instructions

https://github.com/subbyte/padlabs

Remote Lab Environment (ssh access)

```
$ ssh ccs2016@parma.cs.vt.edu -p 2222
```

Task 0 (make your own directory)

```
$ mkdir yourdir; cd yourdir
```