Cloud Data Analytics for Security: Applications, Challenges, and Opportunities

Daphne Yao
Associate Professor of Computer Science
Turner Fellow and L-3 Fellow
Virginia Tech
A Scenario:
Cloud Data Analytics for Organizational Security

Real-time monitoring

Exploits
Control-flow Hijacking
Service Abuse
Insider Threats
APT
Data-oriented programming
Another Scenario: Cloud Data Analytics for Home Security

Origins of spam in a 2014 botnet study

- Embedded Linux servers
- mini-httpd, apache
- ARM devices, MIPS, Realtek chipset
- Open telnet, an SMTP server

A vision: To lift host protections to the cloud

What have been done in cloud?

- Cloud anti-virus, e.g., Sophos and Symantec
- Protection of the cloud, e.g., VM sandboxing, [CloudDiag 2013]
- Software-as-a-service [Cloud Terminal 2012]

What have been done on host?

- Firewalls, host-based anti-virus
- Isolation, e.g., VMM
- Reference monitor, e.g., SELinux
- Trusted computing, e.g., TPM attestation
- Program anomaly detection
Setup Type 1: the Cloud AV model

Sophos Cloud - Cloud-managed Security

HQ office worker

Remote office worker

Home worker

Roaming worker

Admin (Anywhere)
Setup Type 2: Everything in the cloud

YOU'RE PRETTY NEW TO CLOUD STORAGE, AREN'T YOU?
https://www.comprompt.co.in/services/cloud-services/
Cloud terminal [Martignoni 2012]
Setup Type 3: Your refrigerator cannot be in the cloud
Drone Control Station Operating System

From NBC news (2013)


http://nbcnews.tumblr.com/post/47882129464#.UzGIcChfd38
What does it take to lift program anomaly detection to the cloud?

In Setup Type 3: autonomous host with detection in the cloud
Acknowledgments

Drs. Kui Xu
(Google)
Xiaokui Shu
(IBM Research)

Collaborators

Publications:

Global trace analysis
• X. Shu, D. Yao, N. Ramakrishnan. *ACM CCS ’15* (Featured in Comm. of ACM)
• X. Shu, D. Yao, N. Ramakrishnan, T. Jaeger. *ACM TOPS* (journal version, under review)

Program analysis in HMM
• K. Xu, D. Yao, B. Ryder, K. Tian. *IEEE CSF ’15*

HMM with context
• K. Xu, K. Tian, D. Yao, B. Ryder. *IEEE DSN ’16*

Unified framework
• [5] Shu, Yao, Ryder. *RAID 2015*

• ACM CCS Tutorial 2016 on Program Anomaly Detection
Anti-virus Scanning is the First Line of Defense

Vtzilla plugin

For files (apps and PDFs), URLs

Cuckoo Sandbox for dynamic analysis

Number of submissions in a week (March 19, 2017 – March 25, 2017)

Submissions by country

- United States of America: 44.6%
- Canada: 17.2%
- Korea: 15.4%
- France: 7%
- Germany: 6.4%
- Czech Republic: 5.6%
- Russian Federation: 4.8%
- Other: 8%

[From VirusTotal]
Code or Behavior Classification is Undecidable

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

Scanner Thinks

<table>
<thead>
<tr>
<th>IsVirus returns</th>
<th>Contradicts</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td></td>
</tr>
<tr>
<td>False</td>
<td></td>
</tr>
</tbody>
</table>

Actual Behavior of X

<table>
<thead>
<tr>
<th>X chooses not to infect</th>
<th>Contradicts</th>
</tr>
</thead>
<tbody>
<tr>
<td>X chooses to infect</td>
<td></td>
</tr>
</tbody>
</table>
How to detect/prevent zero-day malware/exploits?

Formal verification, Control flow integrity
N-variant, Moving target defense

Anomaly-based detection [D. Denning ’87, Forrest et al. ’96]

(a) Classification
(b) Anomaly detection

[Wressnegger 2013]
Is Typical Insider Trading Detection Anomaly Detection?

<table>
<thead>
<tr>
<th>Purchase Patterns</th>
<th>Sell Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buy low performing stocks</td>
<td>Sell high performing stocks</td>
</tr>
<tr>
<td>Buy before stock prices go up</td>
<td>Sell before stock prices drop</td>
</tr>
<tr>
<td>Purchase followed by purchase</td>
<td>Sell followed by sell</td>
</tr>
</tbody>
</table>

My Work on Anomaly Detection Methodology Development

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

... Security logs,
- Network headers,
- Traffic payloads,
- System traces,
- Transaction logs

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

Program Analysis
(static)

ML/DM
(train and test)

Post Classification Analysis

Binary classification

Novelty detection

Prog analysis

©Unrecognizable Status

©Unrecognizable Status

©Unrecognizable Status

©Unrecognizable Status

©Unrecognizable Status
**Simplest Program Anomaly Detection: n-gram**

<table>
<thead>
<tr>
<th>A 2-gram example:</th>
<th>Runtime program trace</th>
<th>Found in DB?</th>
</tr>
</thead>
<tbody>
<tr>
<td>ioctl()</td>
<td>ioctl()</td>
<td>✓</td>
</tr>
<tr>
<td>open()</td>
<td>open()</td>
<td>✓</td>
</tr>
<tr>
<td>open()</td>
<td>read()</td>
<td>✗</td>
</tr>
<tr>
<td>read()</td>
<td>setpgid()</td>
<td>✗</td>
</tr>
<tr>
<td>setpgid()</td>
<td>setpgid()</td>
<td>✓</td>
</tr>
<tr>
<td>setsid()</td>
<td>setsid()</td>
<td>✓</td>
</tr>
<tr>
<td>fork()</td>
<td>fork()</td>
<td>✓</td>
</tr>
</tbody>
</table>

1. From syscall traces of normal program executions (training data)
2. Test data
3. Classification

[Forrest 1996, Wressnegger 2013]
Who Uses Anomaly Detection on Programs/Systems?

- Average $1.27\text{million/}year$ 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]

Twitter Anomaly Detection. https://blog.twitter.com/2015/
Big Data, Big Bucks

- Netflix
- Twitter
- splunk
- LOGGLY
- +sumologic
- NEX DEFAENSE
- Scalyr
- ThetaRay
- loglogic
- ALERT LOGIC
- graylog
- LogRhythm
- elastic
Manual alert confirmation is costly

It takes 157 minutes for an expensive expert analyst to correctly identify a true positive alert.

The MVX engine identifies true positive alerts without volumes of alerts or false positives, which automation leaves them free for more important tasks. It even finds signs of threat.

- **Contextual intelligence** accompanies validated alerts to help your analysts make decisions, such as attacker profile, threat severity and attack scale and scope.
- **Comprehensive visibility** across the entire lifecycle to reduce alerts by up to 10x so you can focus on the alerts that would be generated from subsequent stages of the attack (e.g. lateral movement). In addition, we have not seen any false positives and zero signatures going on across our whole infrastructure.

"We haven’t seen any false positives and we’re minimizing wastage resources on having to put people on the posture is even more valuable for us."

- Scott Adams
Challenges: Diverse Normal Behaviors, High FP

Detection rate

False positive rate (1-class SVM on libpcre)

Too low! Anomalies detected!

Distribution of function calls in libpcre

Normal 1:  
Normal 2:  
Normal 3:  
Normal 4:  
Normal 5:  
Normal 6:  
Normal 7:  

Anomaly

Normal
False alarms & missed detection can be harmful

Voice-recognition based authentication [CITI Taiwan]

Spam detection

Child pornography detection (FP 1 out of 2 billions)

Pavement distress detection w/ sensors
You found some weird data. Are they meaningful?

rPCA [Candès 2009] works well for motion detection in videos

Images from [Wang 2016]
Semantics of Anomalies in Security

Actions of Attacks and Attack Preparations

- **Control-flow hijacking**
  - Return-oriented programming (ROP)
  - Backdoors

- **Control-flag hijacking**
  - Data-oriented programming (DOP) (not be detected by CFI)

- **Service abuse attacks**
  - Denial of Service (DoS)
  - Memory overread

- **Workflow/state violation**
  - E.g., bypass authentication

- **Exploit preparation**
  - Heap manipulation
  - Address space layout randomization (ASLR) probing
SSH flag variable overwritten attack

```c
void do_authentication(...) {
    int authenticated = 0;
    while (!authenticated) {
        if (auth_password(...)) {
            memset(...);
            xfree(...);
            log_msg(...);
            authenticated = 1;
            break;
        }
        memset(...);
        xfree(...);
        debug(...);
        break;
    }
    if (authenticated) {
        ...
    }
}
```

Pass auth.
Expected

Fail auth.
Expected

Attack

Local analysis cannot detect the anomaly

From [Chen '05]
...sys_ioctl()
sys_open()
sys_read()
sys_setpgid()
sys_setsid()
sys_fork()
...

$n$-gram
[Forrest 1996]
[Forrest 2008]

[Chandola 2009]

[Wagner 2002]

Static Program Analysis

Dynamic Program Analysis

Hybrid detection

FSA [Sekar 2001, Wagner 2001]

PDA [Feng 2003, Feng 2004, Giffin 2004]

Data-flow analysis [Giffin 2006, Bhatkar 2006]

[Shu, Yao, Ryder. RAID 2015]
Old and New Challenges of Data-driven Anomaly Detection

Scale of Data
- Cloud support
- HPC
- Transparency

Subtlety
- Stealthy attacks, e.g., ROP, DOP

Definition of Anomalies
- Domain knowledge
- Inter-discipline
- Usability

Experimental Reproducibility
- Security guarantees
- Benchmarks, baselines, open source

Interpretation of Anomalies
- Semantic gap
- Meanings of anomalies
- Usability

Accuracy of Detection
Issue 1: Incomplete Traces

<table>
<thead>
<tr>
<th>Program</th>
<th># of test cases</th>
<th>branch coverage</th>
<th>line cov.</th>
</tr>
</thead>
<tbody>
<tr>
<td>flex</td>
<td>525</td>
<td>81.34%</td>
<td>76.04%</td>
</tr>
<tr>
<td>grep</td>
<td>809</td>
<td>58.68%</td>
<td>63.34%</td>
</tr>
<tr>
<td>gzip</td>
<td>214</td>
<td>68.49%</td>
<td>66.85%</td>
</tr>
<tr>
<td>sed</td>
<td>370</td>
<td>72.31%</td>
<td>65.63%</td>
</tr>
<tr>
<td>bash</td>
<td>1061</td>
<td>66.26%</td>
<td>59.39%</td>
</tr>
<tr>
<td>vim</td>
<td>976</td>
<td>54.99%</td>
<td>51.93%</td>
</tr>
</tbody>
</table>

From SIR
How to do make HMM smarter in anomaly detection?

Better HMM initialization based on programs

Program analysis for HMM
- Xu, Yao, Ryder, Tian. *IEEE CSF ’15*
- Xu, Tian, Yao, Ryder. *IEEE DSN ’16*
Hidden Markov Model (HMM)

Markov process (memoryless) where some states are not observable
STILO: STatically InitiaLized markOV

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

<table>
<thead>
<tr>
<th>ε_f'(exit)</th>
<th>read</th>
<th>write</th>
<th>execve</th>
</tr>
</thead>
<tbody>
<tr>
<td>ε_f (entry)</td>
<td>p(1-q)</td>
<td>1-p</td>
<td>0</td>
</tr>
<tr>
<td>read</td>
<td>0</td>
<td>0</td>
<td>1-p</td>
</tr>
<tr>
<td>write</td>
<td>1-p</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>execve</td>
<td>pq</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

p, q are statically estimated.
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)

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

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

<table>
<thead>
<tr>
<th>Program Model</th>
<th># distinct calls</th>
<th># states after clustering</th>
<th>Estimated training time reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>bash</td>
<td>1366</td>
<td>455</td>
<td>88.91%</td>
</tr>
<tr>
<td>vim</td>
<td>829</td>
<td>415</td>
<td>74.94%</td>
</tr>
<tr>
<td>proftpd</td>
<td>1115</td>
<td>372</td>
<td>88.87%</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Model</th>
<th>With Static Analysis</th>
<th>With Caller Context</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular-basic</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Regular-context</td>
<td>-</td>
<td>Yes</td>
</tr>
<tr>
<td>STILO-basic</td>
<td>Yes</td>
<td>-</td>
</tr>
<tr>
<td>STILO-context</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

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

Dyninst for static program analysis, Jahmm library for HMM, 1\textsuperscript{st}-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 of magnitude.
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.

Our-2.92x

**False negative rate (logscale base 10)**

**False positive rate**

Syscall: grep

Syscall: gzip
**Detection of Real-world Attacks**

ROP attack segments against gzip (syscall)

<table>
<thead>
<tr>
<th>ID</th>
<th>Prob in STILO</th>
<th>Prob in Regular HMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_1$</td>
<td>0</td>
<td>0.2</td>
</tr>
<tr>
<td>$S_2$</td>
<td>$2.20 \times e^{-15}$</td>
<td>0.29</td>
</tr>
<tr>
<td>$S_3$</td>
<td>$1.54 \times e^{-5}$</td>
<td>0.25</td>
</tr>
<tr>
<td>$S_4$</td>
<td>0</td>
<td>0.27</td>
</tr>
<tr>
<td>$S_5$</td>
<td>0.0005</td>
<td>0.33</td>
</tr>
<tr>
<td>$S_6$</td>
<td>0</td>
<td>0.23</td>
</tr>
<tr>
<td>$S_7$</td>
<td>0.0004</td>
<td>0.26</td>
</tr>
</tbody>
</table>

STILO gives much lower probabilities for attack sequences

<table>
<thead>
<tr>
<th>Exploit</th>
<th>Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buffer Overflow</td>
<td>ROP</td>
</tr>
<tr>
<td>(gzip)</td>
<td>ROPsyscall_chain</td>
</tr>
<tr>
<td>Backdoor (proftpd)</td>
<td>bind_perl</td>
</tr>
<tr>
<td></td>
<td>bind perl ipv6</td>
</tr>
<tr>
<td></td>
<td>generic cmd execution</td>
</tr>
<tr>
<td></td>
<td>double reverse TCP</td>
</tr>
<tr>
<td></td>
<td>reverse_perl</td>
</tr>
<tr>
<td></td>
<td>reverse_perl_ssl</td>
</tr>
<tr>
<td></td>
<td>reverse_ssl_double_telnet</td>
</tr>
<tr>
<td>Buffer Overflow</td>
<td>guess memory address</td>
</tr>
<tr>
<td>(proftpd)</td>
<td></td>
</tr>
</tbody>
</table>
Ongoing Work: Hardware-assisted Program Tracing for Anomaly Detection

In collaboration with Trent Jaeger (PSU)

A control block of libc library
7ffff7a54b01 libc.so <__libc_start_main+177>

A control block for main function
400506 a.out <main+0>
4003e0 a.out <puts@plt+0>

A control block from loader to resolve call
7ffff7df02f0 ld.so <_dl_runtime_resolve+0>
What does it take to outsource STILO detection to the cloud?

- **Training Traces (host)**
  - Probability forecast
  - HMM init & training
  - Test traces (host)
  - HMM classification

- **Fast**
  - Fast and slow
  - Painfully slow
  - Extremely fast

- **Moderate**
  - Not easy
  - Not easy to set up
  - Moderate
Local analysis is inadequate

Anomalies consisting of normal execution fragments
**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

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

**Frequency Anomaly**

**Attack examples:**
- DoS attacks
- Directory harvest attacks

**Can n-gram still work?**
Our Compact Matrix Representation

An infinite long call trace:
... bar, main, foo, bar, bar, ...

Long trace segments

Behavior instance

1. Transition frequency matrix

2. Event co-occurrence matrix

Matrix representation is path insensitive

X. Shu, D. Yao, N. Ramakrishnan. *ACM CCS '15*
Our Solution: Grouping Similar Normal Behaviors

Training Phase

Detection Phase

A trace segment represented by matrices
Montage Anomalies Fall Between Clusters

Pass Auth. (expected)

```
... do_auth > xfree
do_auth > log_msg
do_auth > packet_start
... pwrite > buffer_len
do_auth > do_auth
...```

Anomalous: attack

```
... do_auth > debug
do_auth > xfree
do_auth > packet_start
... pwrite > buffer_len
do_auth > do_auth
...```

Fail Auth. (expected)

```
... do_auth > debug
do_auth > xfree
do_auth > packet_start
... pwrite > buffer_len
do_auth > pread
...```

Function call trace
(collected through Pintool)
Exp 1: Detection Accuracy vs. False Positive in Synthetic 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)
  - Flag variable overwritten attacks w/ various lengths

- **libpcre**
  - Training w/ 11,027 normal behavior instances (44K events each)
  - Regular Exp. DoS
    - 3 malicious patterns
    - 8-23 strings to match

- **sendmail**
  - Training w/ 6,579 normal behavior instances (1K events each)
  - Directory harvest attack w/ probing batch sizes: 8 to 400 emails

100% Detection accuracy
0.01% Average false alarm rate
Security Guarantees:


Main Features:

1. Extremely long traces 2. Low false alarm rate

Tradeoffs:

Path insensitive (orderless)

How to list global trace analysis to the cloud?
Future Work: Anomaly Detection as a Cloud Service

Is it possible to
- be transparent to clients?
- for interdisciplinary data?
- with domain knowledge?
- in production systems?

Can domain experts understand these suggestions?

- Some algorithms are not good for global anomalies;
- The safe bet is to try first global detection algorithms;
- If willing to wait (not real-time detection), use nearest neighbor;
- If the dataset is small, definitely avoid clustering;
- Restart k-mean multiple times to obtain stable clusters;
- Avoid unsupervised anomaly detection for extremely high dimensions;

[Goldstein and Uchida 2016]
Privacy, is it a lost battle (at least in US)?

- US Internet service providers (ISP) to monitor customers’ behavior online without users’ permission,
- to use personal information to sell highly targeted ads

This includes:

- Internet history
- Mobile location data
- App usage
- Content of emails/messages
- Financial information
- Health data

[Washington Post, March 28, 2017]
Future Work: Security/Privacy as Enablers

Future work: Intelligent secure systems and platforms that benefit large populations

Enable new infrastructures
Enable new discoveries
Improve quality of life

Ongoing work: Security Methodology Development
Near-0 false alarm enables analysts to focus on real attacks

http://resources.infosecinstitute.com
Thank you for your attention!

Questions?

More information:

- CCS tutorial video and slides
- System traces, hands-on exercises
Ongoing Work: Event-aware Program Anomaly Detection in CPS/IoT

Conditional probability on physical events’ occurrence

RasPilot

Sponsored by CACI through S2ERC (NSF I/UCRC)
Other Work: Triggering Relation Discovery for Securing Hosts

Prototypes for
- Android
- Linux
- Networks
- File systems

NSF CAREER Award.

[H. Zhang et al. AlSec ‘16] [H. Zhang et al. C&S 2016]
[H. Zhang et al. ASIACCS ‘14] [K. Xu et al. IEEE TDSC ’12]
[H. Almohri et al. IEEE TDSC ‘14] [D. Stefan et al. ACNS ‘10]
Comparison of Detection Capabilities Against Montage Anomalies

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

1-class SVM (w/o clustering)

Ours (w/ clustering)
Our Operations

- Inter-cluster training
- Intra-cluster training

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

[Diagram showing clusters with anomaly points]