A Semantic Framework for Data Analysis in Networked Systems

Arun Viswanathan, Alefiya Hussain, Jelena Mirkovic, Stephen Schwab, John Wroclawski

USC/Information Sciences Institute
Data Analysis in Networked Systems

Is my hypothesis validated?
Did my experiment run as expected?
Why did failure X happen?
Is there any evidence of a known attack?

Alerts
Audit Logs
Packet Dumps
Application Logs
Our Semantic Approach

MODELS
Capture high-level understanding of system

EXPERT

Data collected from an execution of a system

Packet dumps
Webserver logs
Auth logs

Semantic Analysis Framework

 Pose Questions?

Answers to Questions

hypothesis?
expectations met?
evidence of known attacks?

failure X
why?

Models drive analysis over data!
Approximate Lay of the Land

Performance vs. Level of Analysis Abstraction

- **Patterns**
  - Ex. wireshark, tcpdump, snort

- **Queries**
  - Ex. SQL, Splunk

- **Language-based**
  - Ex. Bro, SEC

- **Logic-based**
  - Ex. temporal-logic specification for IDS, CTPL-logic for malware

- **Custom Hackery**
  - Ex. scripts, tools
Key differences with other logic-based approaches

- **Composable abstractions** to capture semantics
- **Expressive relationships** for networked systems
Basics of our Modeling Approach

Behavior
(fundamental abstraction)
Sequence or group of one or more related facts

Complex Behaviors
Related behaviors

Model
Top-level behavior

Models encode higher-level system semantics!

DATA
Multitype, multivariate, timestamped

FACTS
(ex: FILE_OPEN, FILE_CLOSE, TCP_PACKET, ...)

Relationships are key
Relationships in the Modeling Language

Temporal Relationships
- Causality/Ordering
- Eventuality
- Invariance
- Synchrony/Timing

Concurrent Relationships
- Parallelism
- Overlaps

Logical Relationships
- Combinations
- Exclusions

Temporal Operators
- FILE_OPEN -> FILE_CLOSE

Interval Temporal Operators
- HTTP_FLOW \(\text{olap}\) FTP_FLOW

Logical Operators
- \(\text{EXPT_SUCCESS} \oplus \text{EXPT_FAIL}\)
- FILE_CLOSE.name = FILE_OPEN.name

A file open eventually leads to a file close.

HTTP and FTP flows are concurrent.

Experiment either succeeds or fails.

File open and file close are behaviors related by their filename.
Objective: Attacker poisons the victim's DNS cache.

Steps 1-4 keep running in a loop.

KEY ISSUES
Attacker fails to poison cache due to
(1) Race conditions with real nameserver.
(2) Incorrectly GUESSED responses.
Analysis using typical approach

- Tricky to analyze
  - Requires Expertise.
  - Too many random values in the data to extract using simple patterns.
  - Race conditions (timing issues) are hard to debug over 10's of thousands of packets.
  - Many ways to fail.
Model of Behavior

SUCCESS = A guesses right and wins race with R

Nodes: Simple behavior

Arrows: Causal relationships

Path (from root to leaf): Complex Behaviors
Model of Behavior

Nodes: Simple behavior

Arrows: Causal relationships

Path (from root to leaf): Complex Behaviors

SUCCESS = A guesses right and wins race with R.

TIMING_FAIL = A guesses right but loses race to R.
Model of Behavior

**Behavior Model**

1 SUCCESS + 3 FAILURES

**Bad Guess 1**

A guesses wrong response

**Timing Fail**

A guesses right but loses race to R.

**Success**

A guesses right and wins race with R.

**Node:** Simple behavior

**Arrows:** Causal relationships

**Path (from root to leaf):**

Complex Behaviors
# Encoding the Model

1. Capture simple behaviors
   (to capture facts for each distinct attack step)

   \[ VtoR\_query = \text{DNSREQRES.dns\_req}(\text{sip}=$AtoV\_query.dip, \text{dnsquesname}=$AtoV\_query.dnsquesname) \]

2. Relate simple behaviors to form complex behaviors
   (to capture the causal relationships between steps)
   4 behaviors = 1 SUCCESS + 3 FAILURES

   \[ \text{TIMING\_FAIL} = (AtoV\_query \rightarrow VtoR\_query \rightarrow RtoV\_resp \rightarrow AtoV\_resp) \]

3. Define Behavior Model
   (assertion to capture users understanding of system operation)

   \[ \text{DNSKAMINSKY} = \text{SUCCESS} \oplus \text{TIMING\_FAIL} \oplus \text{BADGUESS\_1} \oplus \text{BADGUESS\_2} \]
Analysis Using the Model

Behavior captured in 20 lines of model

DNS Kaminsky
Behavior model

States
\( s_B = \{ \text{sip} = s_A.\text{dip}, \text{dip} = s_A.\text{sip} \} \)

Behavior
\( b = s_A \rightarrow s_B \)

Model
\( \text{SUCCESS} = b_1 \)

DNS Data

Behavior captured in 20 lines of model

DNS Kaminsky
Behavior model

States
\( s_B = \{ \text{sip} = s_A.\text{dip}, \text{dip} = s_A.\text{sip} \} \)

Behavior
\( b = s_A \rightarrow s_B \)

Model
\( \text{SUCCESS} = b_1 \)

DNS Data

Summary: DNSCACHEPOISON_TIMING_FAIL

Total Matching Instances: 622

<table>
<thead>
<tr>
<th>etype</th>
<th>timestamp</th>
<th>sip</th>
<th>dip</th>
<th>sport</th>
<th>dport</th>
<th>dnsid</th>
<th>dnsauth</th>
</tr>
</thead>
<tbody>
<tr>
<td>PKT_DNS</td>
<td>1275515486</td>
<td>10.1.11.2</td>
<td>10.1.4.2</td>
<td>6916</td>
<td>53</td>
<td>47217</td>
<td></td>
</tr>
<tr>
<td>PKT_DNS</td>
<td>1275515486</td>
<td>10.1.4.2</td>
<td>10.1.11.2</td>
<td>32778</td>
<td>53</td>
<td>15578</td>
<td></td>
</tr>
<tr>
<td>PKT_DNS</td>
<td>1275515486</td>
<td>10.1.11.2</td>
<td>10.1.4.2</td>
<td>53</td>
<td>32778</td>
<td>15578</td>
<td>realns.eby.com</td>
</tr>
<tr>
<td>PKT_DNS</td>
<td>1275515486</td>
<td>10.1.11.2</td>
<td>10.1.4.2</td>
<td>53</td>
<td>32778</td>
<td>15578</td>
<td>realns.eby.com</td>
</tr>
<tr>
<td>PKT_DNS</td>
<td>1275515486</td>
<td>10.1.11.2</td>
<td>10.1.4.2</td>
<td>53</td>
<td>32778</td>
<td>15578</td>
<td>realns.eby.com</td>
</tr>
<tr>
<td>PKT_DNS</td>
<td>1275515486</td>
<td>10.1.11.2</td>
<td>10.1.4.2</td>
<td>53</td>
<td>32778</td>
<td>15578</td>
<td>realns.eby.com</td>
</tr>
</tbody>
</table>

Answers in the form of facts satisfying the model.

Did the poisoning succeed or fail?

 TIMING_FAIL
(A loses the race against R)
Current Implementation and Performance

- Prototype algorithm for applying models over data.
- Algorithm performance
  - $O(N^2)$ worst-case performance
  - Straight-forward
- Analysis Framework
  - Written in Python
  - SQLite-based storage backend
- Scalability and performance issues are under active investigation.
Applicability

- Broad range of event-based modeling in networked systems
- More examples in paper
  - Modeling hypotheses
    - Ex. Validating DoS detection heuristics over traces
  - Modeling a security threat
    - Ex. Model of a simple worm spread over IDS logs
  - Modeling dynamic change
    - Ex. Model of changes in traffic rate due to attack.
Future Work

- Extend Modeling Capabilities
  - Modeling probabilistic behavior
  - Modeling packet distributions
- Analysis Framework
  - Scalability and performance
  - Reducing the computational complexity of correlations using dependent attributes.
Composing, Sharing and Reusing

Semantic Analysis Framework enables data analysis at higher-levels of abstraction.

Composing models to create higher-level meaning

Sharing and reusing expertise

Abstract Behavior Models

[states]
sA = {sip=$1, dip=$2}
sB = {sip=$sA.dip, dip=$sA.sip}
[behavior]
b = sA \rightarrow sB
[model]
SUCCESS = b_1
Thank You!

Our framework will soon be publicly available at http://thirdeye.isi.deterlab.net

Please register on our mailing-list to stay in tune with release and updates