Functional Graph Pattern Matching for Cybersecurity and Beyond

Xiaokui Shu, Fred Araujo, Douglas Schales, Marc Stoecklin

IBM Research

Xiaokui Shu, Frederico Araujo, Douglas L. Schales, Marc Ph. Stoecklin, Jiyong Jang, Heqing Huang, and Josyula R. Rao. 2018. ***Threat Intelligence Computing***. In 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18), October 15–19, 2018, Toronto, ON, Canada

Project sponsored by the Air Force Research Laboratory (AFRL) and the Defense Advanced Research Projects Agency (DARPA) under the award number FA8650-15-C-7561.

“It takes an average of 206 days to detect a data breach.”

-- *Ponemon The Cost of Data Breach 2017*

Every Campaign Is Different
They are Developed On-The-Fly

One exploit failed? Try a 0-day one.

The server is well protected? Try its backup.

C&C is under radar? Try a Twitter post.

Data movement is monitored? Try legitimate channels.

Cannot hack that a laptop? Try social engineering.

Develop a new exploit detection schema?

Did the attacker deliberately turn off the server?

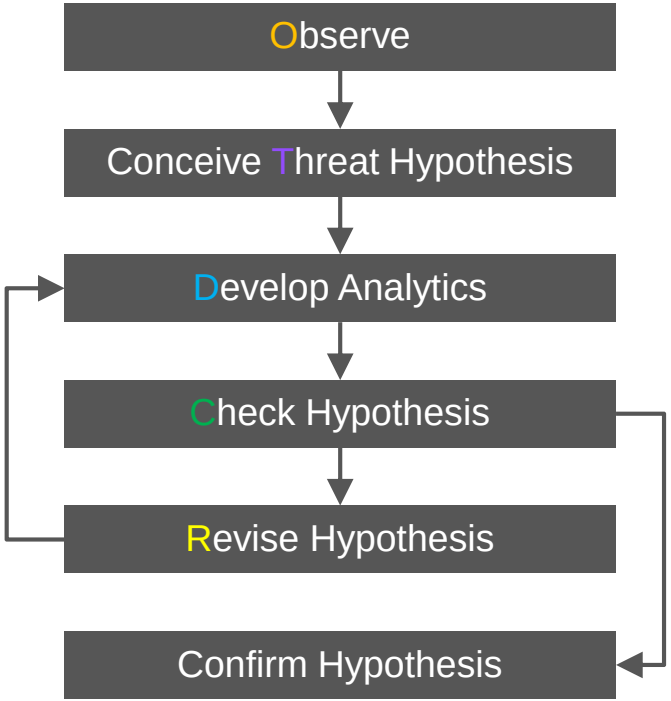
Add Twitter traffic to existing C&C detector?

If that FTP is used, how does it connect alerts?

Does a stolen password complete the jigsaw?

Detect on the Move

Threat Discovery/Hunting as a Fast Scientific Discovery Problem



Develop a new exploit detection schema?

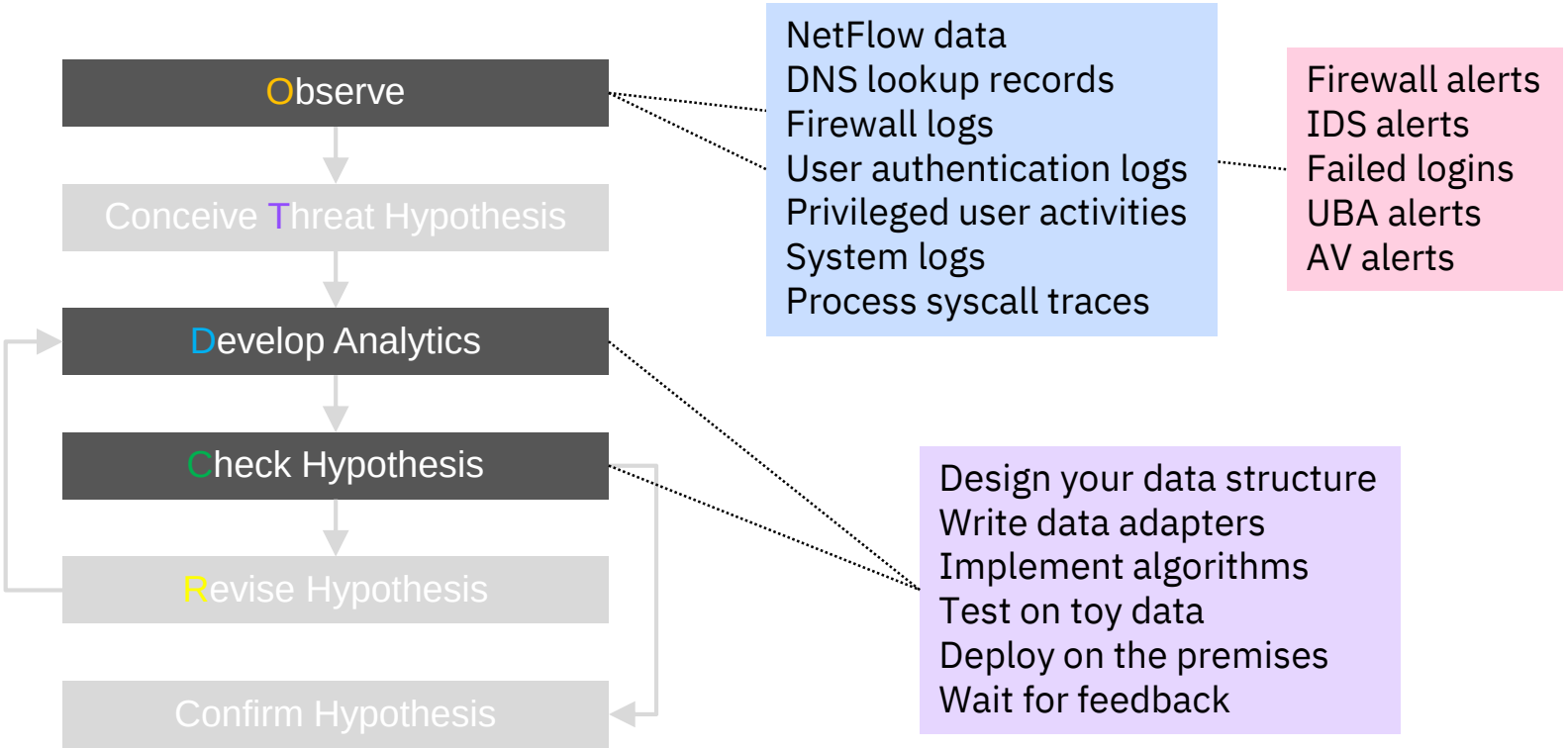
Did the attacker deliberately turn off the server?

Add Twitter traffic to existing C&C detector?

If that FTP is used, how does it connect alerts?

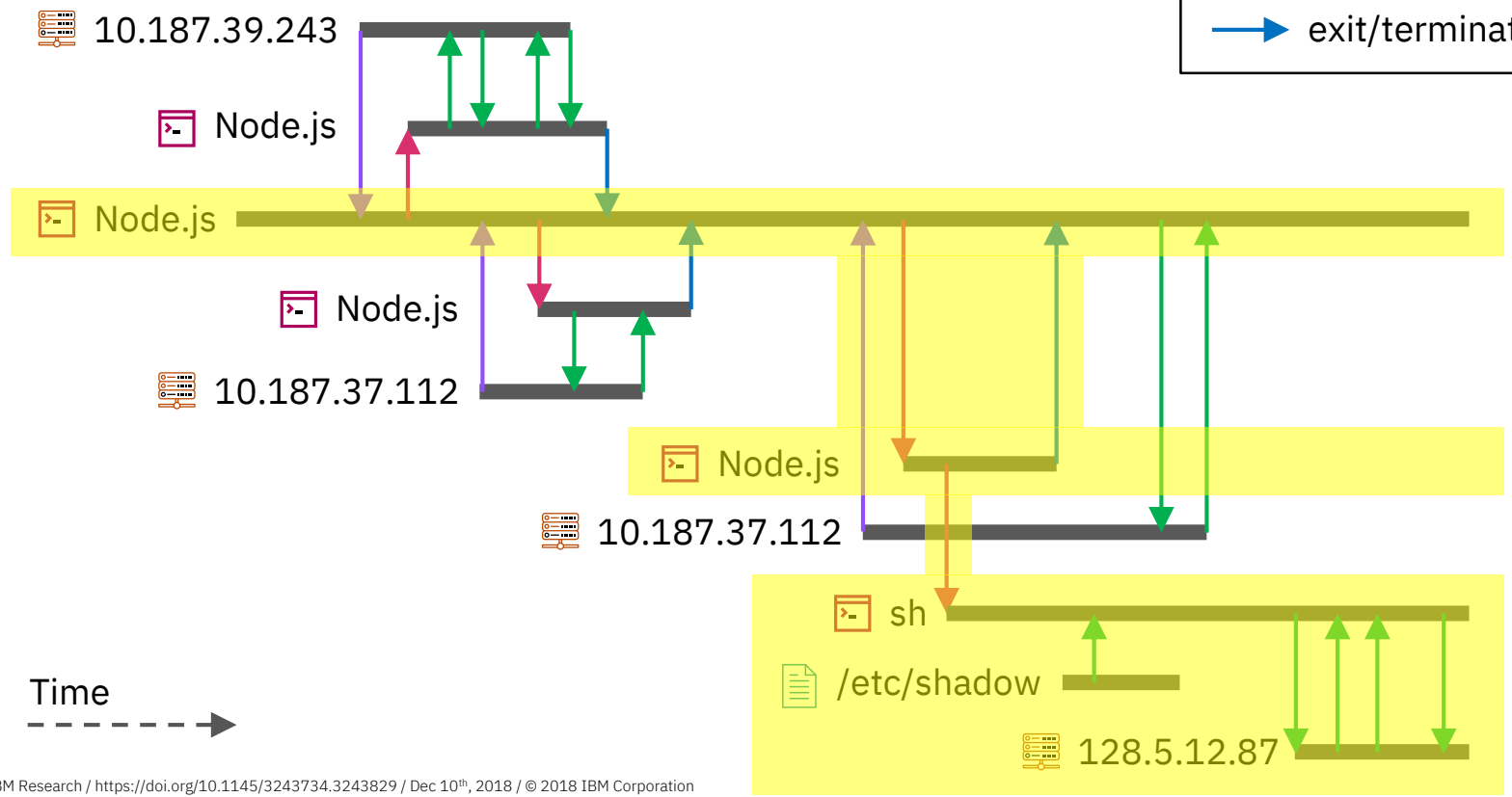
Does a stolen password complete the jigsaw?

March the Marsh of Existing Systems



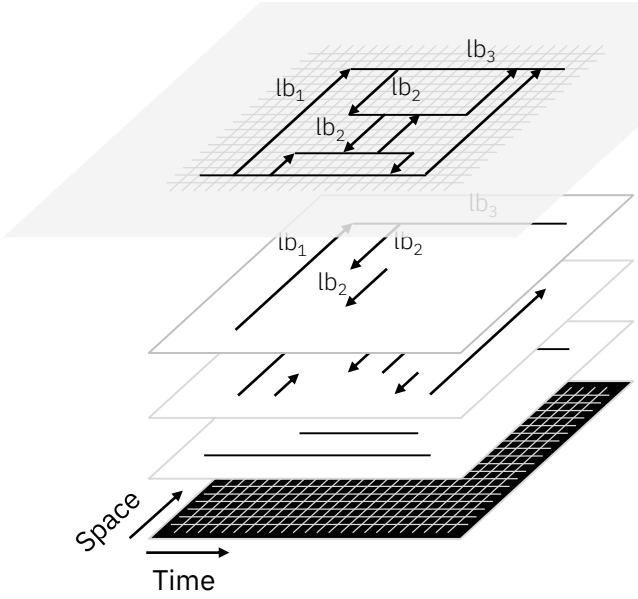
Escape the Marsh, Take the Highway

Threat Discovery/Hunting as a (Temporal) Graph Computation Problem



One Data Representation

Computation Graph (CG): An Abstract Computation Model in Temporal Graph



Labels

Events

Entities

Canvas

- Element Attributes
- Intermediate Results
- Firewall alerts
- UBA alerts
- AV alerts
- Process syscall
- NetFlow data
- DNS lookup records
- Privileged user activities
- Firewall logs
- Network-Level
- Host-Level
- Memory-Level

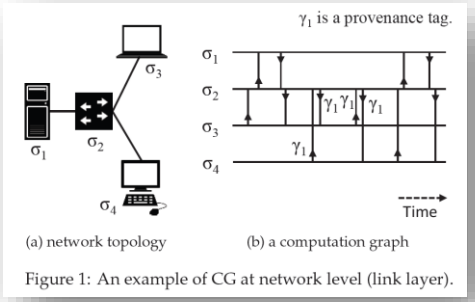


Figure 1: An example of CG at network level (link layer).

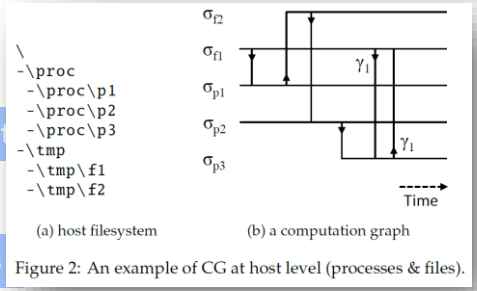


Figure 2: An example of CG at host level (processes & files).

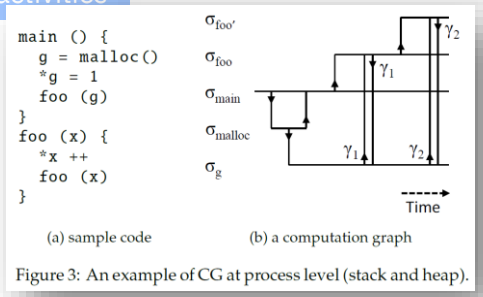
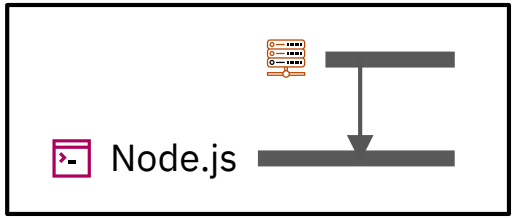


Figure 3: An example of CG at process level (stack and heap).

One Operation Abstraction

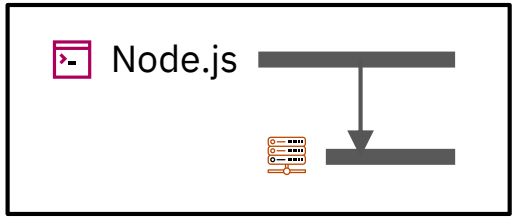
Graph Pattern Matching

Observe: What network activities did the Node process have?



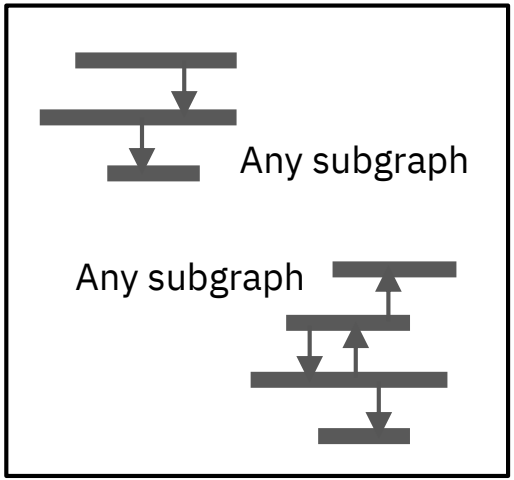
A pattern comprises:

- An entity
  - Is a process
  - Has “node” in its “cmdline”
- An entity
  - Is a network resource
- An event
  - Source: first entity
  - Destination: second entity



A pattern comprises:

- An entity
  - Is a process
  - Has “node” in its “cmdline”
- An entity
  - Is a network resource
- An event
  - Source: second entity
  - Destination: first entity



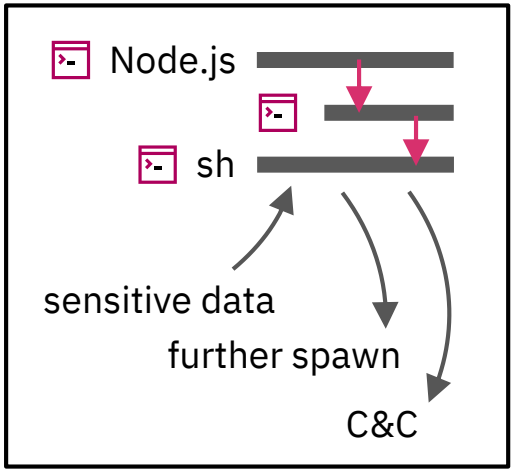
A pattern comprises:

- A subgraph
  - All entities
  - All events
- Another subgraph (may not connect to the first subgraph)
  - All entities
  - All events

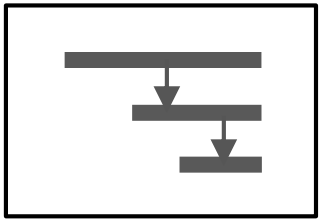
One Operation Abstraction

Functional Graph Pattern Matching

Develop/Check: Multi-level spawning behavior? Malicious?

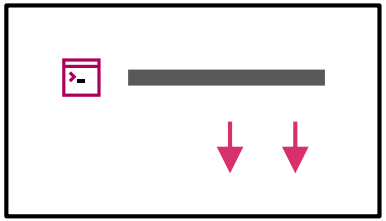


=



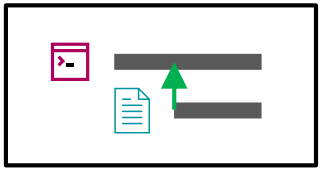
A pattern for general 2-hop traversal

+



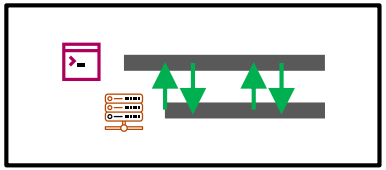
A pattern to constrain the specific traversal

+



A pattern to express sensitive data access

+

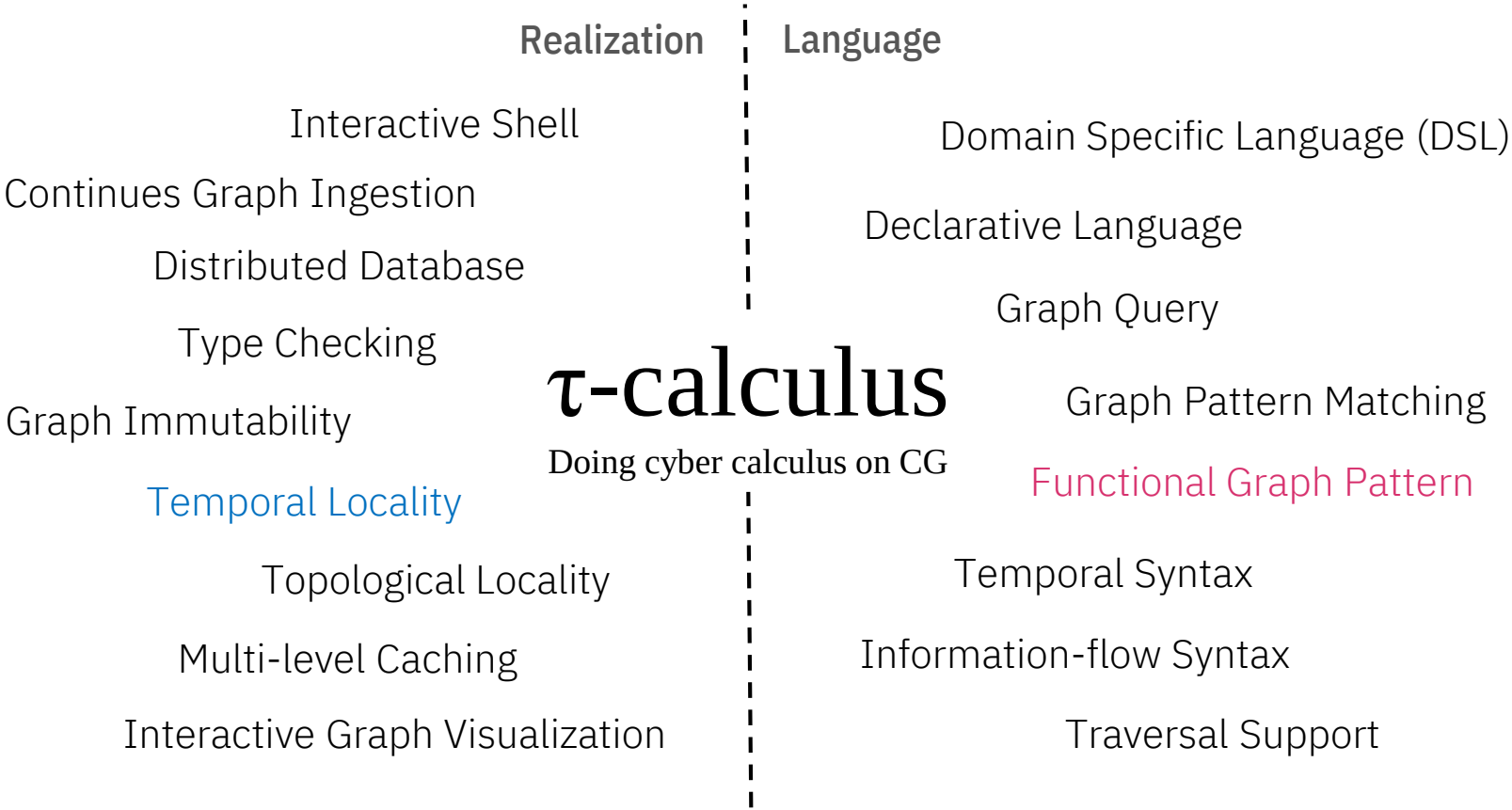


A pattern to describe C&C behavior

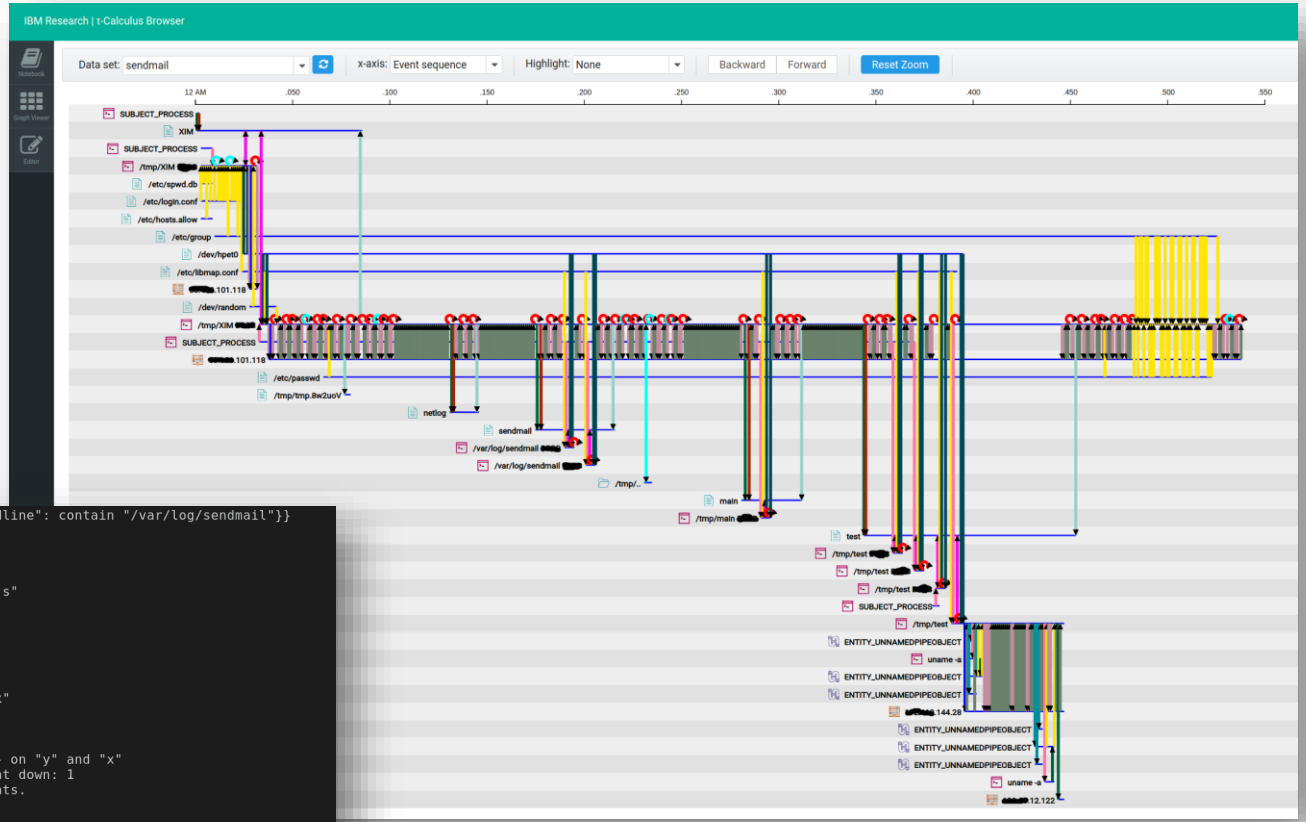
A pattern comprises:

- Bare-bone 2-level spawning
- One type of malicious behavior at the end of the spawned process

Yet Another Graph Computation Platform



Batch Exec
REPL
Graph Viz
Visual Programming




```
Taucalculus> pattern patSendmail () { -s s{"cmdline": contain "/var/log/sendmail"}}
[Info] graph pattern patSendmail learned.
Taucalculus> sdml = patS
patSendmail       patSuspiciousIPs
Taucalculus> sdml = patSendmail ()
[Info] solving unary constraint {property} on "s"
[Info] "s" changed, start propagation for it
[Info] initial propagation queue: []
[Info] end propagation for "s"
2 entities, 0 events.
Taucalculus> sdml2 = backtraversal1 (sdml)
[Info] graph pattern backtraversal1 learned.
[Info] solving unary constraint {ingraph} on "x"
[Info] "x" changed, start propagation for it
[Info] initial propagation queue: []
[Info] end propagation for "x"
[Info] solving binary constraint {reachability} on "y" and "x"
[Info] DB query: Joinpoints -> Events; BFS count down: 1
[Info] DB query results: 13 Joinpoints, 78 Events.
[Info] "y" changed, start propagation for it
[Info] initial propagation queue: []
[Info] end propagation for "y"
13 entities, 78 events.
Taucalculus>
```

Thank you!

ACKNOWLEDGMENTS

This project was sponsored by the Air Force Research Laboratory (AFRL) and the Defense Advanced Research Agency (DARPA) under the award number FA8650-15-C-7561.

The views, opinions, and/or findings contained in this article are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.