Badger: The Networked Security State Estimation Toolkit

Edmond Rogers, Will Rogers, and Gabe Weaver

Badger42.org
A target system network is a tuple \( N = (G, \Sigma, \Delta, T, I) \)

- \( G = (V, E) \) is a graph
- The graph has type information \( T = (T_V, T_E, T_G) \)
  - \( \Sigma \) is an alphabet of vertex types
  - \( \Delta \) is an alphabet of edge types
  - \( T_V : V \rightarrow \Sigma \) is a function that maps \( v \in V \) to \( \sigma \in \Sigma \)
  - \( T_E : E \rightarrow \Delta \) is a function that maps \( e \in E \) to \( \delta \in \Delta \)
  - \( T_G : \Sigma \times \Delta \rightarrow \text{String} \) defines the type of the graph

- The graph has identifier information
  - \( id_V(V) : V \rightarrow \text{String} \) is a function that maps a vertex to an identifier
  - \( id_E(E) : E \rightarrow \text{String} \) is a function that maps an edge to an identifier
  - \( id_G(G) : G \rightarrow \text{String} \) is a function that maps the graph to an identifier.
• Human-readable graphical language

• Registry of vertex and edge types

• Join atomic values among disparate data sources
Cloud Infrastructure Provider (Cloudspace)
urn:cptl:cloud:streampics.network

Cloud Service Provider (Streampics)

Badger42.org
urn:cptl:cloud:cloudspace-streampics.network
Serialize to GraphML

urn:cptl:cloud:cloudspace.network

Blade_1
Blade_2
Blade_3
Blade_4

TORSwitch_1
Main Switch
TORSwitch_2

Badger42.org
• Red == Bad

• Yellow == Caution

• Green == It’s all good
Networked Security State Estimation

- Measurement of the state of security
- Somewhat polarizing
- Take it for what its worth . . .

Badger42.org
URN:CPTLID:CAPABILITY:ATTRIBUTE

Diagram renders in application
import skills, sys, time, demo
from luck import *

now = time.time()
demo = open('badger', 'r')

for blackhat in demo:
    print skills.haxor(blackhat)
else:
    print 'WASTED!'
REQUEST: badger42.org/GETCAPABILITY?source_vertex_attr_type=urn-cptl-HOST-ipv4

RESPONSE: A list of capabilities formatted as text/json

[
  {
    name: "urn:badger:get_hostip_dest_hostnames",
    description: "Given an IPv4 address, get the destination hostnames",
    source_vertex_attr_type = "urn-cptl-HOST-ipv4",
    target_vertex_attr_type = "urn-cptl-HOST-hostname"
  },

  {
    name: "urn:badger:get_host_dest_tldcounts",
    description: "Given an IPv4 address, get the top-level domain counts",
    source_vertex_attr_type = "urn-cptl-HOST-ipv4",
    target_vertex_attr_type = "urn-cptl-HOST-hostname"
  }
]
REQUEST: badger42.com/service?name=urn:badger:get_host_dest_tld
counts_selected_vertex_attr_values = 192.168.1.100, 192.168.1.120

RESPONSE: A graph of the following format:

```json
{{
  "id": 1, 
  source1_vertex_attr_type: "urn-cptl-HOST-ipv4"
},
{{
  "id": 2, 
  source2_vertex_attr_type: "urn-cptl-HOST-ipv4"
},
{{
  "id": 3, 
  target1_vertex_attr_type: "urn-cptl-HOST-tag-tldcount",
  target1_vertex_attr_value: "com,44"
},
{{
  "id": 4, 
  target2_vertex_attr_type: "urn-cptl-HOST-tag-tldcount",
  target2_vertex_attr_value: "com,44"
}},

EDGES: [{
  "source": 1, 
  "target": 4, 
},
{{
  "source": 2, 
  "target": 4}]

```
Analysis in progress...

Mode: command line
[Cisco Parser][Backup EMS.txt] done.
DONE 1/5
[Cisco Parser][Primary EMS.txt] done.
DONE 2/5
DONE 3/5
[Cisco Parser][Internet.txt] done.
DONE 4/5
[Cisco Parser][Main CORP.txt] done.
DONE 5/5
File(s) successfully parsed.
Performing topology inference...
Inspect routes...
Creating primary networks...
Marking VPN networks...
Creating nodes from group definitions...
Building border cloud...