NetControl

Johanna Amann
johanna@icir.org
NetControl

Push rules to networking hard and software

Based on traffic observed by Bro

Simple to use but flexible API
Uses for NetControl

Traffic Shunting
Block attacks at network boundary
Redirecting high traffic flows to different interfaces
Quarantine hosts
Uses for NetControl

Traffic Shunting
Block attacks at network boundary
Redirecting high traffic flows
Quarantine hosts
Uses for NetControl

Traffic Shunting

Block attacks at network boundary

Redirecting high traffic flows to different interfaces

Quarantine hosts
Architecture

NetControl Framework

Bro Event Engine

High level calls or low-level primitives

Network Traffic

Rules

Success, Failure, Timeout

NetControl Framework

Backends

Switch

Switch

Router

Firewall
Architecture

Bro

Event Engine

NetControl Framework

High level calls or low-level primitives

Network Traffic

Current Backends

OpenFlow
Command line applications
Acld
Bro Packet Filter

Backend 4

Success, Failure, Timeout

Firewall
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>install_dst_addr_filter: function</code></td>
<td>Installs a filter to drop packets destined to a given IP address with a certain probability if none of a given set of TCP flags are set.</td>
</tr>
<tr>
<td><code>install_dst_net_filter: function</code></td>
<td>Installs a filter to drop packets destined to a given subnet with a certain probability if none of a given set of TCP flags are set.</td>
</tr>
<tr>
<td><code>install_src_addr_filter: function</code></td>
<td>Installs a filter to drop packets from a given IP source address with a certain probability if none of a given set of TCP flags are set.</td>
</tr>
<tr>
<td><code>install_src_net_filter: function</code></td>
<td>Installs a filter to drop packets originating from a given subnet with a certain probability if none of a given set of TCP flags are set.</td>
</tr>
</tbody>
</table>
High level API

- drop_connection (connection, timeout)
- drop_address (host, timeout)
- drop_address_catch_release (host)
- shunt_flow (flow, timeout)
- quarantine (infected host, dns host, q. server, timeout)
- whitelist (prefix, timeout)
event GridFTP::data_channel_detected(c: connection) {
    NetControl::shunt_flow(
        [$src_h=c$id$orig_h, $src_p=c$id$orig_p,
            $dst_h=c$id$resp_h, $resp_p=c$id$resp_p],
        1hr);
}

event log_notice(n: Notice::Info) {
    if ( n$note == Address_Scan || n$note == Port_Scan )
        NetControl::drop_address(n$src, 10min);
}
What do Rules look like?

Type:
- Drop
- Modify
- Redirect
- Whitelist

Target:
- Forward
- Monitor

Entity:
- Address
- Mac
- Connection
- Flow

Other:
- Timeout
- Priority
- Location
Example

Rule(Type=Drop, Entity=Flow([5-tuple]), Target=Monitor)

```c
function shunt_flow(f: flow_id, t: interval) : string {
    local flow = Flow(
        $src_h=addr_to_subnet(f$src_h), $src_p=f$src_p,
        $dst_h=addr_to_subnet(f$dst_h), $dst_p=f$dst_p
    );
    local e: Entity = [$ty=FLOW, $flow=flow];
    local r: Rule = [
        $ty=DROP, $target=MONITOR, $entity=e, $expire=t
    ];
    return add_rule(r);
}
```
Choosing Backends

NetControl Framework

Bro

Bro Event Engine

NetControl Framework

Backend 1

Backend 2

Backend 3

Backend 4

Switch

Switch

Router

Firewall

Network Traffic

High level calls or low-level primitives

Rules

Success, Failure, Timeout

Device communication
Choosing Backends

NetControl Framework

OpenFlow Backend 1

OpenFlow Backend 2

OpenFlow Backend 3

Network A

Tap switch

Network B
Choosing Backends

NetControl Framework

OpenFlow Backend 1: 5
OpenFlow Backend 2: 2
OpenFlow Backend 3: 0

Network A
Tap switch
Network B
Choosing Backends

NetControl Framework

- OpenFlow Backend 1 (5)
- OpenFlow Backend 2 (2)
- OpenFlow Backend 3 (0)

Network A

Tap switch

Network B
Choosing Backends

NetControl Framework

OpenFlow Backend 1

OpenFlow Backend 2

OpenFlow Backend 3

Network A

Network B

Tap switch
Choosing Backends

NetControl Framework

- OpenFlow Backend 1: 5
- OpenFlow Backend 2: 2
- OpenFlow Backend 3: 0

Network A

Tap switch

Network B
Choosing Backends

NetControl Framework

OpenFlow Backend 1

OpenFlow Backend 2

OpenFlow Backend 3

Network A

Network B

Tap switch
Choosing Backends

NetControl Framework

OpenFlow Backend 1: 5
OpenFlow Backend 2: 2
OpenFlow Backend 3: 0

Network A
Tap switch
Network B
Choosing Backends

NetControl Framework

OpenFlow Backend 1: 5
OpenFlow Backend 2: 2
OpenFlow Backend 3: 0

Network A

Network B

Tap switch
Choosing Backends

NetControl Framework

OpenFlow Backend 1 - 5
OpenFlow Backend 2 - 2
OpenFlow Backend 3 - 0

Network A

Tap switch

Network B
Choosing Backends

NetControl Framework

OpenFlow Backend 1: 5
OpenFlow Backend 2: 2
OpenFlow Backend 3: 0

Network A
Tap switch
Network B
Adding Backends

```c++
local backend = NetControl::create_backend_Foo([...]);
NetControl::activate(backend, 10);
```
State management

Rules often only needed for limited time

NetControl supports timeouts

…but respects hard/software that don’t need them
OpenFlow

Open Specification

Allows Software to insert rules into switch flow tables

Match (and change) characteristics like

IPv4/6 addresses, ports, etc.

Vlans
NetControl & OpenFlow

Block, Shunt, … Decisions

Network Control Framework

NC OpenFlow Backend

OpenFlow Module

Ryu OpenFlow Controller

OpenFlow Switch

Broker Protocol

OpenFlow Protocol

Bro
Demonstration
Rule Insertion Speed

![Box plot showing time to rule insertion for different systems.]

- HP 1 Rule: Time to rule insertion is relatively low, around 0.5 seconds.
- IBM 1 Rule: Similar to HP 1 Rule, time is around 0.5 seconds.
- IBM 2 Rules: Time is significantly higher, around 0.7 seconds.
- Pic8 1 Rule: Time to rule insertion is very low, around 0.1 seconds.
- Pic8 2 Rules: Time is also very low, around 0.1 seconds.
Schedule 0.899309sec { kill_me(116.178.14.117) };
schedule 1.02567sec { kill_me(8.214.17.167) };
schedule 1.60747sec { kill_me(126.138.19.67) };
schedule 1.68983sec { kill_me(28.193.234.0) };
schedule 2.89801sec { kill_me(16.212.210.166) };
schedule 2.76121sec { kill_me(28.199.215.62) };
schedule 3.19226sec { kill_me(11.10.145.91) };
schedule 3.71398sec { kill_me(136.80.163.214) };
schedule 4.44176sec { kill_me(229.23.77.196) };
schedule 4.39617sec { kill_me(144.213.190.85) };
schedule 5.66566sec { kill_me(194.214.62.250) };
schedule 3.97636sec { kill_me(90.95.173.149) };
schedule 6.20912sec { kill_me(32.164.142.218) };
schedule 6.65181sec { kill_me([2607:9ff3:aac2:1798:3edb:71a2:5c2c:e036]) };
schedule 7.56999sec { kill_me(76.40.117.86) };
schedule 7.67942sec { kill_me(168.35.60.159) };
schedule 8.09308sec { kill_me([2607:2156:3fb5:a66:b1e5:bb7c:ab6d:a4dd]) };
schedule 8.35657sec { kill_me(234.31.231.76) };
schedule 8.19995sec { kill_me(48.58.230.80) };

...
# Blocked Connections

<table>
<thead>
<tr>
<th>Switch</th>
<th>Block time</th>
<th>Not blocked</th>
<th>Transferred Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Med.</td>
</tr>
<tr>
<td>Pica8 (Median)</td>
<td>8.5ms</td>
<td>4,229 (2.7%)</td>
<td>0</td>
</tr>
<tr>
<td>Pica8 (75 Percentile)</td>
<td>11ms</td>
<td>8,273 (5.1%)</td>
<td>12</td>
</tr>
<tr>
<td>IBM (Median)</td>
<td>41ms</td>
<td>27,848 (17.4%)</td>
<td>194</td>
</tr>
<tr>
<td>IBM (75 Percentile)</td>
<td>89ms</td>
<td>41,965 (26.3%)</td>
<td>526</td>
</tr>
<tr>
<td>HP (Median)</td>
<td>82ms</td>
<td>38,381 (24%)</td>
<td>454</td>
</tr>
<tr>
<td>HP (75 Percentile)</td>
<td>93ms</td>
<td>43,128 (27%)</td>
<td>537</td>
</tr>
</tbody>
</table>
NetControl Summary

Control switches and other hardware

Easy syntax and rules

Extensible (API & Backends)

Fast
Get NetControl

github.com/bro/bro-netcontrol