Flow Shunting

Flow shunting with Arista switches requires two components; the react module and a dumbno server instance. These instructions expand upon information found here: http://mailman.icsi.berkeley.edu/pipermail/zeek/2017-July/012353.html

The dumbno server needs to run on a server that can access the Arista API. We run dumbno on our Zeek manager node. Dumbno takes very little CPU and memory resources.

Adding the react module to Zeek

Clone Justin Azoff's bro react repo and add it to the Bro site directory.

https://github.com/JustinAzoff/bro-react

In /usr/local/bro/share/bro/site/local.bro, load the react framework and the bulk connection script.

local.bro

```
@load conn-bulk.bro
@load react
```

Configuring and running Dumbno server

Download dumbno from the dumbno repo.

https://github.com/ncsa/dumbno

Edit the configuration file to talk to the API on the Arista switch. We are applying a single incoming ACL named "bulk_1" on the port-channel "Port-Channel1" that includes all taps. "Port-Channel10" is the set of tool ports that are connected to the Zeek workers. The egress_ports section is used in the "stats" function of dumbno to gather traffic statistics.

dumbno.cfg

```
[switch]
scheme = https
ip = <ip of arista device>
user = admin
password = <password>

[ports]
Port-Channel1 = bulk_1

[egress_ports]
Port-Channel10 = tool1
```

Set up the ACLs on the switch (only needs to be done once)

```
PYTHONHTTPSVERIFY=0 python dumbno.py dumbno.cfg setup
```

Dumbno will automatically start after setting up the ACLs.

Start dumbno

If you already have the ACLs on the Arista switches, you can start dumbno this way.

```
PYTHONHTTPSVERIFY=0 python dumbno.py dumbno.cfg
```

Dumbno provides a log file at /var/log/dumbno.log which can be used to monitor ACL changes.

Start dumbno stats monitor

```
PYTHONHTTPSVERIFY=0 python dumbno.py dumbno.cfg stats
```

A log file is generated at /var/log/dumbno.stats which gives information about ingress, egress and amount of traffic filtered by the dumbno ACL.

***Python had some trouble verifying the certificate we used, even though it was signed by a trusted CA. PYTHONHTTPSVERIFY=0 disables the certificate verification.

Redefining variables in Zeek

From base/protocols/ftp/gridftp.bro, the number of bytes transferred before guessing a connection is a GridFTP data channel is originally set to 1 GB.

```
const size_threshold = 1073741824 &redef;
```

This can be changed to 1 MB in local.bro, along with the size_threshold for identifying bulk flows.

```
redef GridFTP::size_threshold = 1048576; # 1 MB
redef Bulk::size_threshold = 134217728 ; # 128 MB
```

Traffic Whitelisting with Dumbno

We are whitelisting two types of traffic to decrease the load on Zeek. Zeek will continue to log these connections even though no packets are forwarded except sin/fin/rst.

1. XRootD file access protocol which runs on tcp port 1094 and 1095
2. Gridftp traffic from the v4/v6 subnets associated with Purdue's CMS T2 center on the defined port range 51000-52000

The following setup_acl function of dumbno was changed to apply the ACLs to drop traffic for these two cases. Ideally, we would move these hard coded changes to a configuration file that can be read at dumbno startup.

Traffic whitelisting configuration

```
def setup_acl(self,  acl):
        if self.acl_exists(acl):
            return True

        self.logger.info("Setting up %s ACL %s", acl.family, acl.name)
        # block known cms gridftp ports
        if acl.family == "ip":
          gridftp1 = "100001 deny tcp <ipv4 subnet> range 51000 52000 any"
          gridftp2 = "100002 deny tcp <ipv4 subnet> range 51000 52000"
        elif acl.family == "ipv6":
          gridftp1 = "100001 deny tcp <ipv6 subnet> range 51000 52000 any"
          gridftp2 = "100002 deny tcp any <ipv6 subnet> range 51000 52000"
        cmds = [
            "enable",
            "configure",
            "%s access-list %s" % (acl.family, acl.name),
            "statistics per-entry",
            "10 permit tcp any any fin",
            "20 permit tcp any any syn",
            "30 permit tcp any any rst",
            # block known CMS gridftp ports
            gridftp1,
            gridftp2,
            # block known xrootd ports
            "100003 deny tcp any any eq 1094 1095",
            "100004 deny tcp any eq 1094 1095 any",
            "100005 permit %s any any" % acl.family,
        ]
        response = self.switch.runCmds(version=1, cmds=cmds)
```

Connection log file entry for whitelisted traffic

```
conn.log:bro-a003-1-1    1564076229.408022    CdJifp3WUxE84HbZcf    X.X.X.X    41736    Y.Y.Y.
Y    51700    tcp    -    114.227315    2304658968    0    SF    F    T    0    ShFf    2    3252    2    1
12    (empty)    XX:XX:XX:XX:XX:XX    YY:YY:YY:YY:YY:YY
```

The "ShFf" entry in the history column of the connection log shows that only the TCP handshake and teardown messages are seen by the Zeek worker. The number of bytes (orig_bytes) of 2304658968 is an estimated value but is fairly accurate.