Introduction Bro Cluster

Justin Azoff (NCSA)

jazoff@illinois.edu

Motivations

- Intelligence based searching is **incredibly** common.
- Through abstraction we can expand the utilization of intelligence.
- Creating a format for importing intelligence makes Bro target-able for intelligence providers.

Design Limitation

Asynchronous lookups

You can’t use “do I know about this?” in a normal if statement.

Running Bro By Hand

- To run in “base” mode:
  - bro -r traffic.pcap
- To run in a “near broctl” mode:
  - bro -r traffic.pcap local
- To add extra scripts:
  - bro -r traffic.pcap /home/seth/myscript.bro

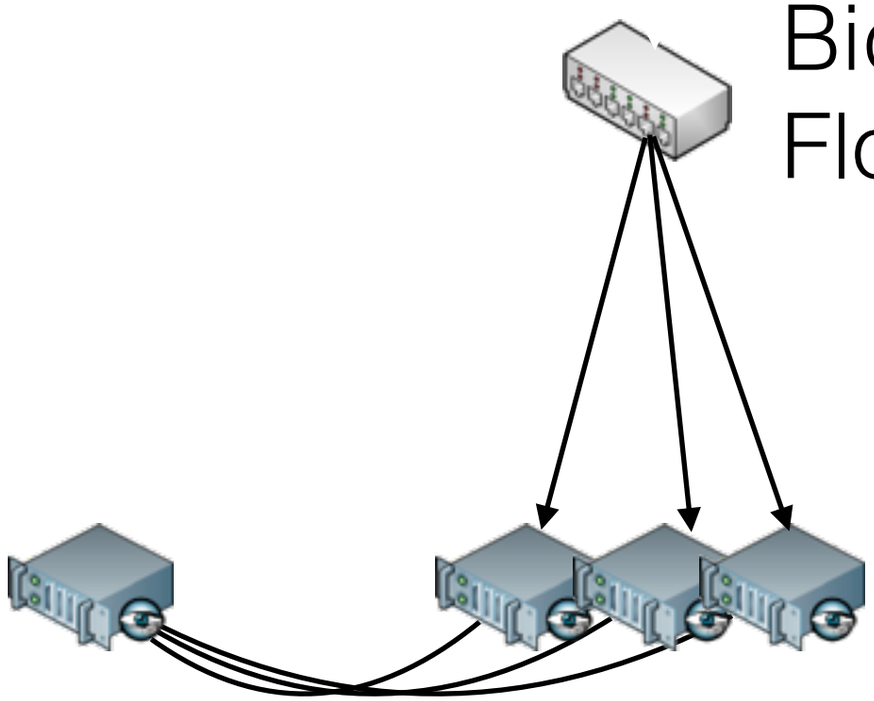
Getting Bro up and Running

- Use Bro Control (broctl)!
- What is broctl?
  - Written in python.
  - Installed by default with Bro.
  - Manages live and long running Bro instances.
  - Manages complexity of running clusters.

Network Load Balancing

- If your load outstrips capacity of a single host, you need this.
- Several options for flow balancing (no particular order)
  - Arista
  - NetOptics
  - cPacket
  - Gigamon
  - VSS Monitoring

Common border deployment



Manager
Mostly logs and notices (frequently proxies run here too)

Workers
Traffic analysis

Getting Bro up and Running

- In many cases, just two files to edit:
  - networks.cfg
  - node.cfg
- Also, docs on the bro.org website. We have improved cluster docs coming.

networks.cfg

```
# List of local networks in CIDR notation, optionally
followed by a
# descriptive tag.
# For example, "10.0.0.0/8" or "fe80::/64" are valid
prefixes.

10.0.0.0/8              Private IP space
192.168.0.0/16          Private IP space
```

node.cfg - standalone

```
# This is a complete standalone configuration.  Most
likely you will
# only need to change the interface.
[bro]
type=standalone
host=localhost
interface=eth0
```

node.cfg - cluster

Typically this is what you'll use.

Bro scales across hosts as a cluster.

```
[manager]
type=manager
host=host1

[proxy-1]
type=proxy
host=host1

[worker-1]
type=worker
host=host2
interface=eth0

[worker-2]
type=worker
host=host3
interface=eth0
```

On-Host Flow Balancing

- Running one process per host isn't good when hosts have many CPU cores.
- Scale across cores with on-host flow balancing.
- Most common methods today are PF_Ring and Myricom (with sniffer driver).

Load balancing PF_Ring

- Many people use PF_Ring.
- Linux-only
- Configure Bro with PF_Ring's libpcap wrapper:

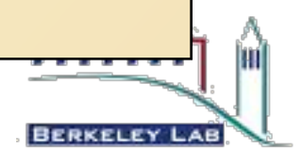
```
./configure --with-pcap=/usr/local/
```

node.cfg example

```
[manager]
type=manager
host=host1

[proxy-1]
type=proxy
host=host1

[worker-1]
type=worker
host=host2
interface=eth0
lb_method=pf_ring
lb_procs=10
```



Load balancing Myricom

- Many people use Myricom NICs.
  - Works on FreeBSD and Linux
  - Buy something in the 8B series with the Sniffer Driver (SNF) license (only 10G NICs).
- Configure Bro with Myricom's libpcap wrapper:

./configure --with-pcap=/opt/snf/

node.cfg example

```
[manager]
type=manager
host=host1

[proxy-1]
type=proxy
host=host1

[worker-1]
type=worker
host=host2
interface=eth0
lb_method=myricom
lb_procs=10
```

Cluster Checklist

- SSH key based authentication for user running broctl.
- User running Bro has permission to sniff network interface.
- GeoIP data installed on each system.

It's configured! Now what?

- Run broctl
- [BroControl] > install
- [BroControl] > start
- Check in <prefix>/logs/current for logs.

Questions?