What can the NSF Bro Center of Excellence do for me?

Adam Slagell

NCSA CISO & CyberSec Div. Director

August 17th, 2015

National Center for Supercomputing Applications
University of Illinois at Urbana–Champaign

The NSF Bro Center of Excellence

- Bro support for NSF projects & Higher-Ed
  - Oct 2013 launch at Summit
- Development work for these communities
  - E.g. SDN & Science DMZ is important to them (PACF)
- Research
  - Can't save 3 months of pcaps, run analysis live
- Outreach
  - BroCon & NSF Cybersecurity Summit
  - Partnering with CTSC & ESNet on projects
  - 1-on-1 engagements

Some communities engaged so far

- LIGO
- Mississippi State
- UC Santa Cruz
- Washington University St. Louis
- University of Virginia
- National Center for Atmospheric Research
- Penn State
- U of Utah
- Cornell
- UT Austin
- Rochester Institute of Technology
- UW Madison
- Clemson
- Indiana
- Ice Cube
- University of Idaho
- Washington County School District in Utah

Ask us to help with…

- Troubleshooting & Optimizing
  - Cluster setups & tap/agg aren't easy
  - CPU affinity and Hyper-threading?
- Planning & reviewing designs for NSM
  - Where should I tap? What are pros/cons?
  - How much hardware should I start with?
  - Should I design for peak or average?
- So I installed it, now what?
  - *i.e., the rest of this talk*
  - Way more than an IDS

Did someone download malware?

- Does everyone know Team Cymru?
  - They publish hashes of known, static malware.
- Do you know about Bro’s file analysis framework?
- You can combine the 2 to detect malware downloads.
  - More in a demo from Justin shortly.

Lack endpoint management?

- Common university problem
  - Web plugin whack-a-mole
- Check out the software.log sometime
- Use Bro to detect flash, Java, Acrobat versions
  - Works really nicely with Splunk

Damn you encryption!

- Everything’s getting encrypted right?
  - Not really, but still lots you can do
- Got private keys to your web service?
- Run custom SSHD binaries?
  - Scott Campbell @ NERSC and iSSHD (in GSISSH now)
- What’s in that SSL.log?
  - More from Johanna in a bit
- Some caveats

Spammers on your network

- Easy to detect spam relays
- What about spamming accounts?
  - Lots of email expected from SMTP server anyway
- Bro can app layer analysis + sumstats to the rescue!
  - You can count how many emails sent and rate per user

Ugh, UDP

- So someone installed a new NTP server…
- At one institution, networking updated routers, and all of them where part of an attack in minutes.
- Keeps coming back every time a new server is built with an old image.
- Trivial to detect with Bro though

Automate your whack-a-mole

- Want to know if someone is scanning you
  - Or you them?
- Is someone brute-forcing SSHD?
- Block them!
  - Tie Bro a black-hole router or SDN
    - Check out Justin’s BHR code on github

Why not share?

- If you blocked it, maybe they want to too?
  - We do this with campus, hoping to for XSEDE
- Intel framework can be used to import this
  - Also with CIF for REN-ISAC and many other feeds
- Being used for a Science DMZ appliance we are developing more generally
  - If you want to pilot this with us, talk to me

Misconfiguration or policy violation

- Using outside DNS server
  - Lot’s of nxdomain responses
- Wrong NTP server
  - OS may default to foreign server (scale to cluster)
- Participation in an amplification attack due to poor config
- Hosting unapproved domains
  - At least for HTTP

Configuration Management

- Did someone stick a new host on your network?
- Did a host reboot with a new service?
- You can whitelist or blacklist hosts/services on a network
- You could even start building profiles of hosts to take this further

What's the process look like?

- Contact us
  - https://www.bro.org/nsf/
  - nsf@bro.org
- Setup a meeting
  - Couple pre-meeting questions
  - Send diagrams 1st if you have them
- Develop a plan and a timeline
  - What do we want to accomplish?
  - How long do we give this?
  - How regularly do we meet?

The Bro Monitoring Platform

Adam Slagell

National Center for Supercomputing Applications

Borrowed from Robin Sommer
International Computer Science Institute

“What Is Bro?”

“What Is Bro?”

Packet Capture

“What Is Bro?”

TCPDUMP

Packet Capture

WIRESHARK

Traffic Inspection

“What Is Bro?”

Packet Capture

Traffic Inspection

Attack Detection

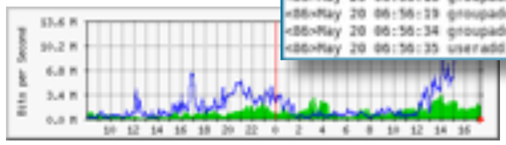
"What Is Bro?"

Packet Capture

Traffic Inspection

Attack Detection

NetFlow



```
[root@fedora ~]# cat /root/install.log.syslog
<86>May 20 06:56:18 groupadd[1564]: new group: name=utmp, GID=22
<86>May 20 06:56:19 groupadd[1568]: new group: name=utempter, GID=35
<86>May 20 06:56:34 groupadd[1618]: new group: name=floppy, GID=19
<86>May 20 06:56:35 useradd[1623]: new group: name=vcsa, GID=69
```

syslog

Log Recording

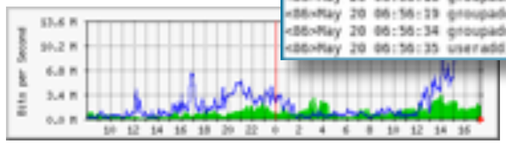
“What Is Bro?”

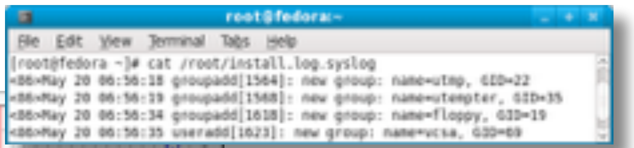
Packet Capture

Traffic Inspection

Attack Detection

NetFlow





```
[root@fedora ~]# cat /root/install.log.syslog
<86>May 20 06:56:18 groupadd[1564]: new group: name=utmp, GID=22
<86>May 20 06:56:19 groupadd[1568]: new group: name=utempter, GID=35
<86>May 20 06:56:34 groupadd[1618]: new group: name=floppy, GID=19
<86>May 20 06:56:35 useradd[1623]: new group: name=vcsa, GID=69
```

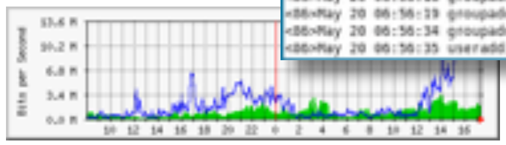
syslog

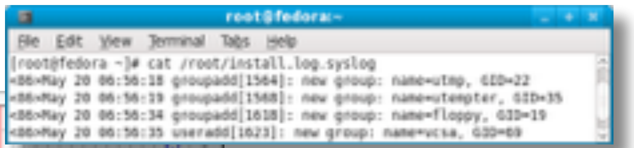
Log Recording

Flexibility
Abstraction
Data Structures

"What Is Bro?"

NetFlow





```
[root@fedora ~]# cat /root/install.log.syslog
<86>May 20 06:56:18 groupadd[1564]: new group: name=utmp, GID=22
<86>May 20 06:56:19 groupadd[1568]: new group: name=utempter, GID=35
<86>May 20 06:56:34 groupadd[1618]: new group: name=floppy, GID=19
<86>May 20 06:56:35 useradd[1623]: new group: name=vcsa, GID=69
```

syslog

Packet Capture

Traffic Inspection

Attack Detection

Log Recording

Flexibility
Abstraction
Data Structures

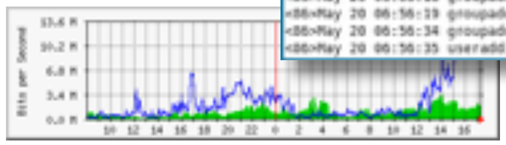
"What Is Bro?"

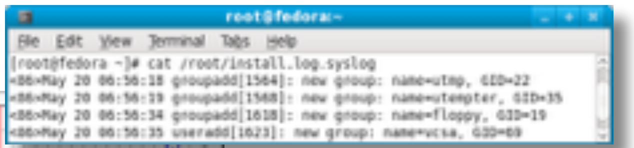
TCPDUMP

WIRESHARK

SNORT

NetFlow





```
[root@fedora ~]# cat /root/install.log.syslog
<86>May 20 06:56:18 groupadd[1564]: new group: name=utmp, GID=22
<86>May 20 06:56:19 groupadd[1568]: new group: name=utempter, GID=35
<86>May 20 06:56:34 groupadd[1618]: new group: name=floppy, GID=19
<86>May 20 06:56:35 useradd[1623]: new group: name=vcsa, GID=69
```

syslog

python™

- Packet Capture
- Traffic Inspection
- Attack Detection
- Log Recording
- Flexibility
  Abstraction
  Data Structures

BRO NETWORK SECURITY MONITOR

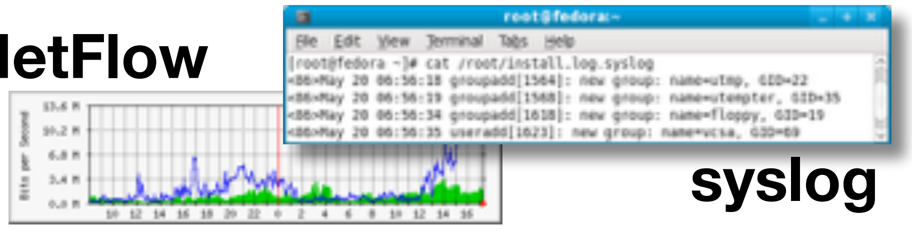
"What Is Bro?"

TCPDUMP

WIRESHARK

SNORT

NetFlow



```
[root@fedora ~]# cat /root/install.log.syslog
<86>May 20 06:56:18 groupadd[1564]: new group: name=utmp, GID=22
<86>May 20 06:56:19 groupadd[1568]: new group: name=utempter, GID=35
<86>May 20 06:56:34 groupadd[1618]: new group: name=floppy, GID=19
<86>May 20 06:56:35 useradd[1623]: new group: name=vcsa, GID=69
```

syslog

python™

Packet Capture

Traffic Inspection

Attack Detection

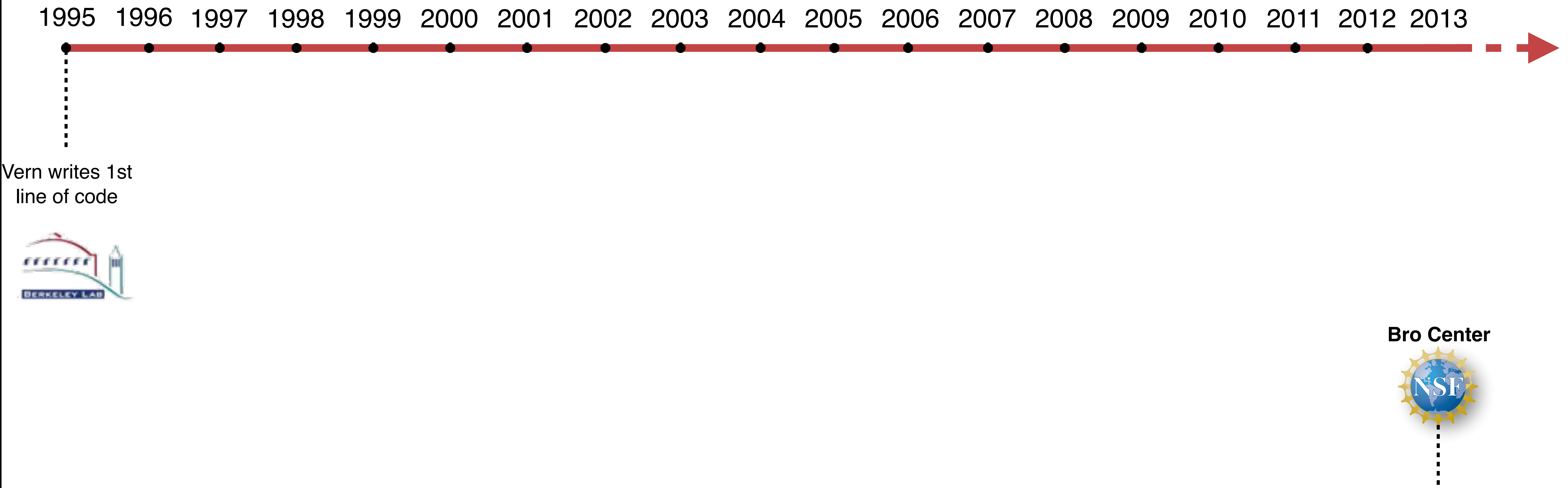
Log Recording

Flexibility
Abstraction
Data Structures

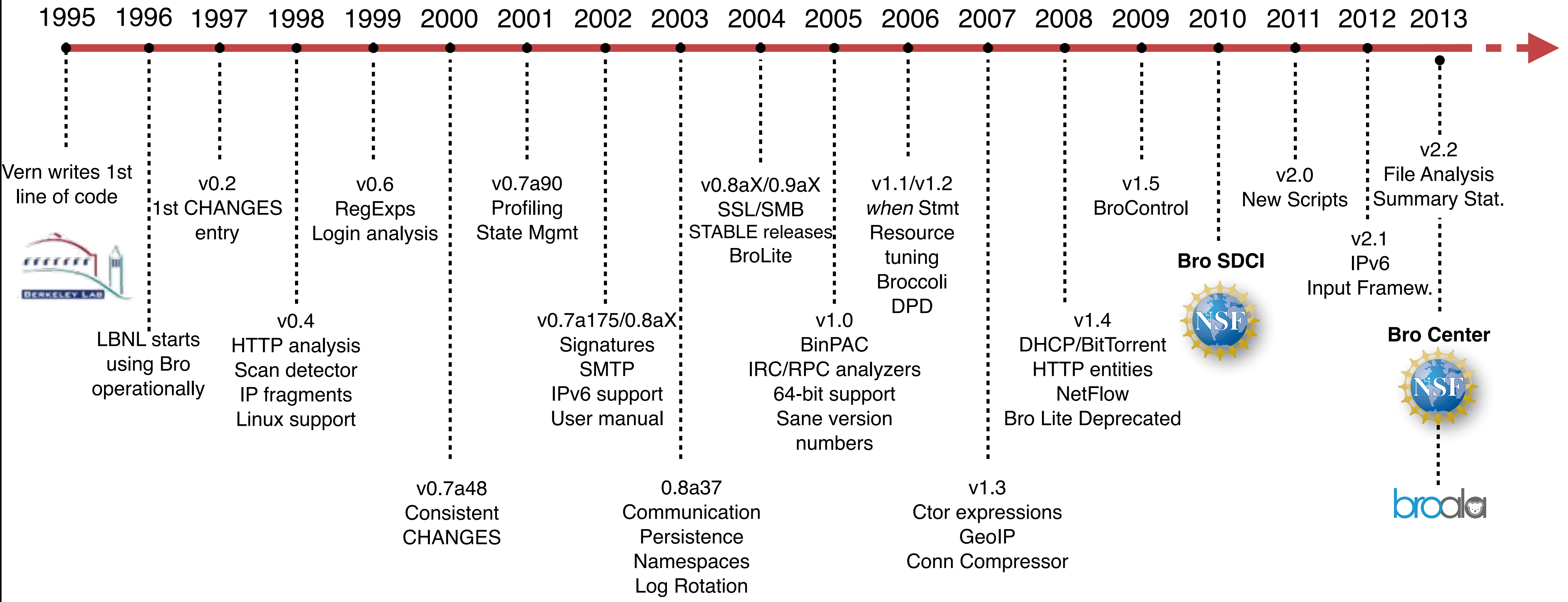
BRO NETWORK SECURITY MONITOR

"Domain-specific Python"

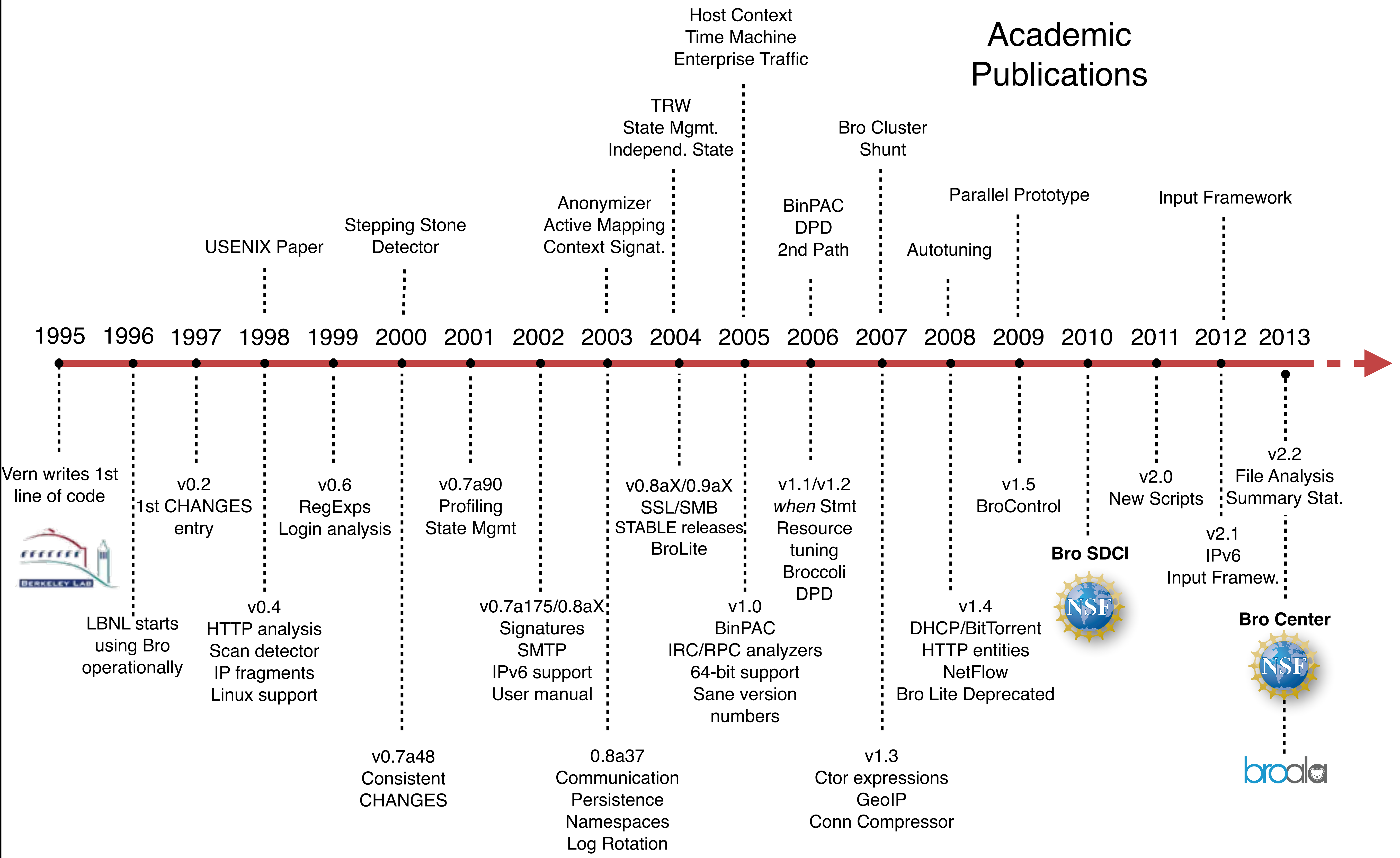
Bro History


1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
Vern writes 1st line of code
BERKELEY LAB
Bro Center
NSF

Bro History


1995
Vern writes 1st line of code
1996
LBNL starts using Bro operationally
1997
v0.2
1st CHANGES entry
1998
v0.4
HTTP analysis
Scan detector
IP fragments
Linux support
1999
v0.6
RegExps
Login analysis
2000
v0.7a48
Consistent CHANGES
2001
v0.7a90
Profiling
State Mgmt
2002
v0.7a175/0.8aX
Signatures
SMTP
IPv6 support
User manual
2003
0.8a37
Communication
Persistence
Namespaces
Log Rotation
2004
v0.8aX/0.9aX
SSL/SMB
STABLE releases
BroLite
2005
v1.0
BinPAC
IRC/RPC analyzers
64-bit support
Sane version numbers
2006
v1.1/v1.2
when Stmt
Resource tuning
Broccoli
DPD
2007
v1.3
Ctor expressions
GeoIP
Conn Compressor
2008
v1.4
DHCP/BitTorrent
HTTP entities
NetFlow
Bro Lite Deprecated
2009
v1.5
BroControl
2010
Bro SDCI
2011
v2.0
New Scripts
2012
v2.1
IPv6
Input Framew.
2013
v2.2
File Analysis
Summary Stat.
Bro Center

Bro History


Academic Publications
1995
Vern writes 1st line of code
1996
LBNL starts using Bro operationally
1997
v0.2 1st CHANGES entry
1998
USENIX Paper
v0.4 HTTP analysis Scan detector IP fragments Linux support
1999
v0.6 RegExps Login analysis
2000
Stepping Stone Detector
v0.7a48 Consistent CHANGES
2001
v0.7a90 Profiling State Mgmt
2002
v0.7a175/0.8aX Signatures SMTP IPv6 support User manual
2003
Anonymizer Active Mapping Context Signat.
0.8a37 Communication Persistence Namespaces Log Rotation
2004
TRW State Mgmt. Independ. State
v0.8aX/0.9aX SSL/SMB STABLE releases BroLite
2005
Host Context Time Machine Enterprise Traffic
v1.0 BinPAC IRC/RPC analyzers 64-bit support Sane version numbers
2006
BinPAC DPD 2nd Path
v1.1/v1.2 *when* Stmt Resource tuning Broccoli DPD
2007
Bro Cluster Shunt
v1.3 Ctor expressions GeoIP Conn Compressor
2008
Autotuning
v1.4 DHCP/BitTorrent HTTP entities NetFlow Bro Lite Deprecated
2009
Parallel Prototype
v1.5 BroControl
2010
Bro SDCI
2011
v2.0 New Scripts
2012
Input Framework
v2.1 IPv6 Input Framew.
2013
v2.2 File Analysis Summary Stat.
Bro Center

“Who’s Using It?”

Installations across the US

Universities
Research Labs
Supercomputing Centers
Government Organizations
Fortune 50 Enterprises

Examples

Lawrence Berkeley National Lab
National Center for Supercomputing Applications
Indiana University
General Electric
Mozilla Corporation
... and many more sites I can’t talk about.

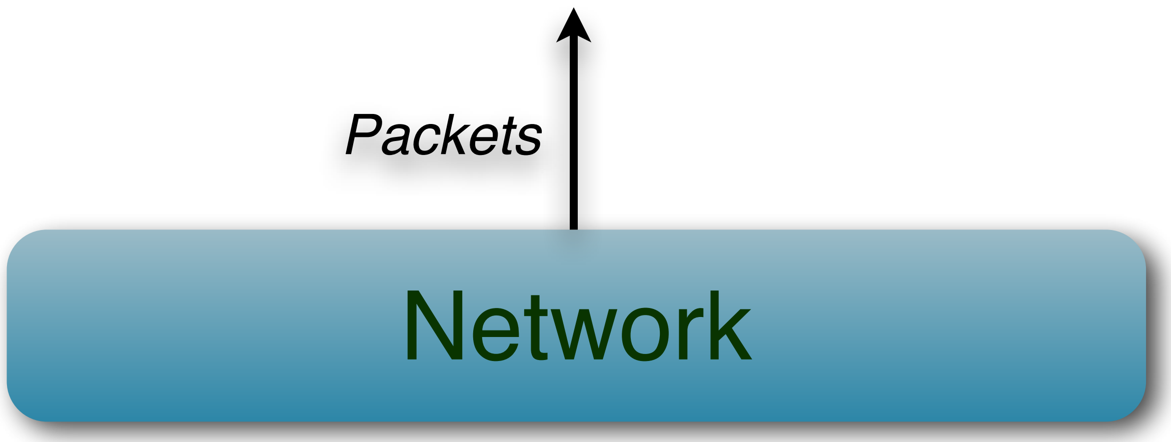
Fully integrated into *Security Onion*

Popular security-oriented Linux distribution

Community

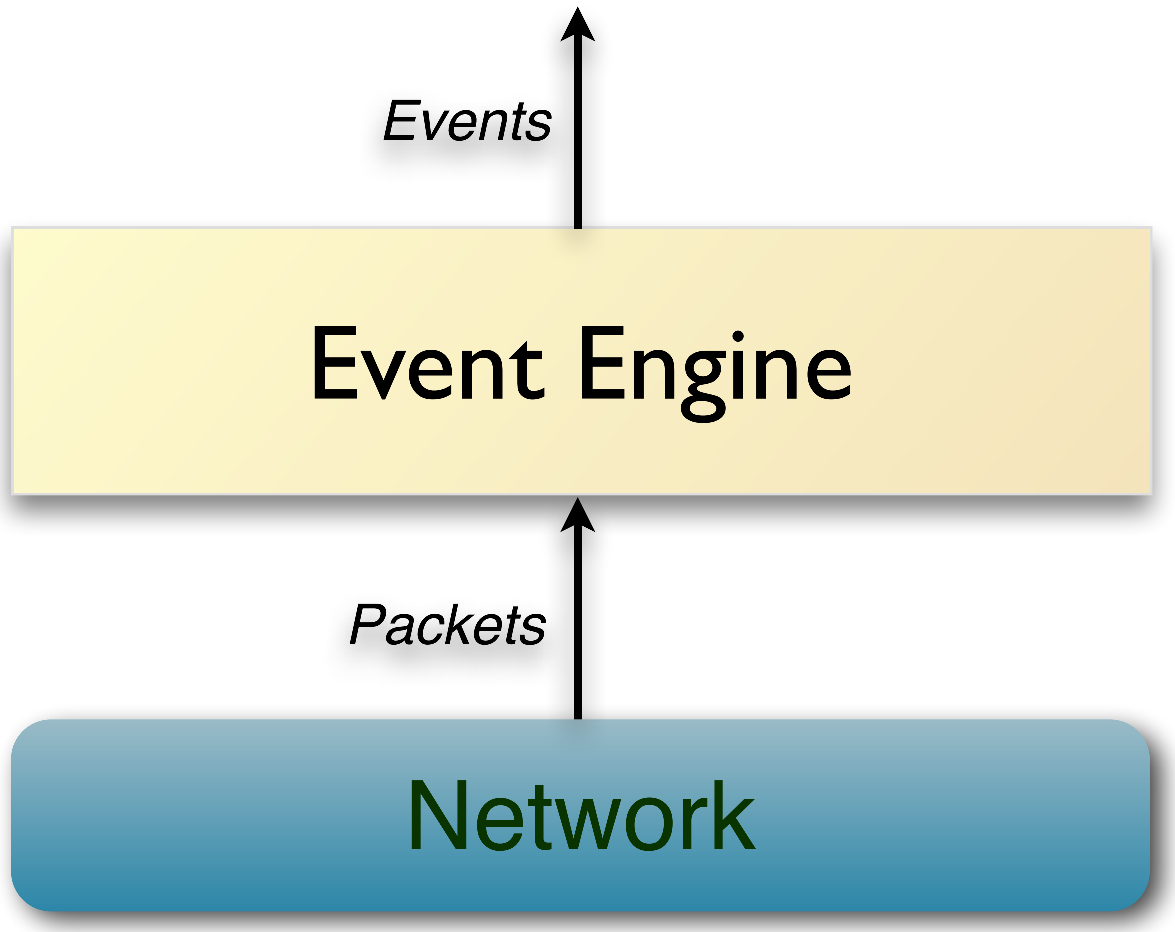
50/90/150/185 attendees at BroCon ’12/’13/’14/‘15
110 organizations at BroCon ‘14
~4,000 Twitter followers
~1000 mailing list subscribers
~100 users average on IRC channel
10,000+ downloads / version from 150 countries

Architecture

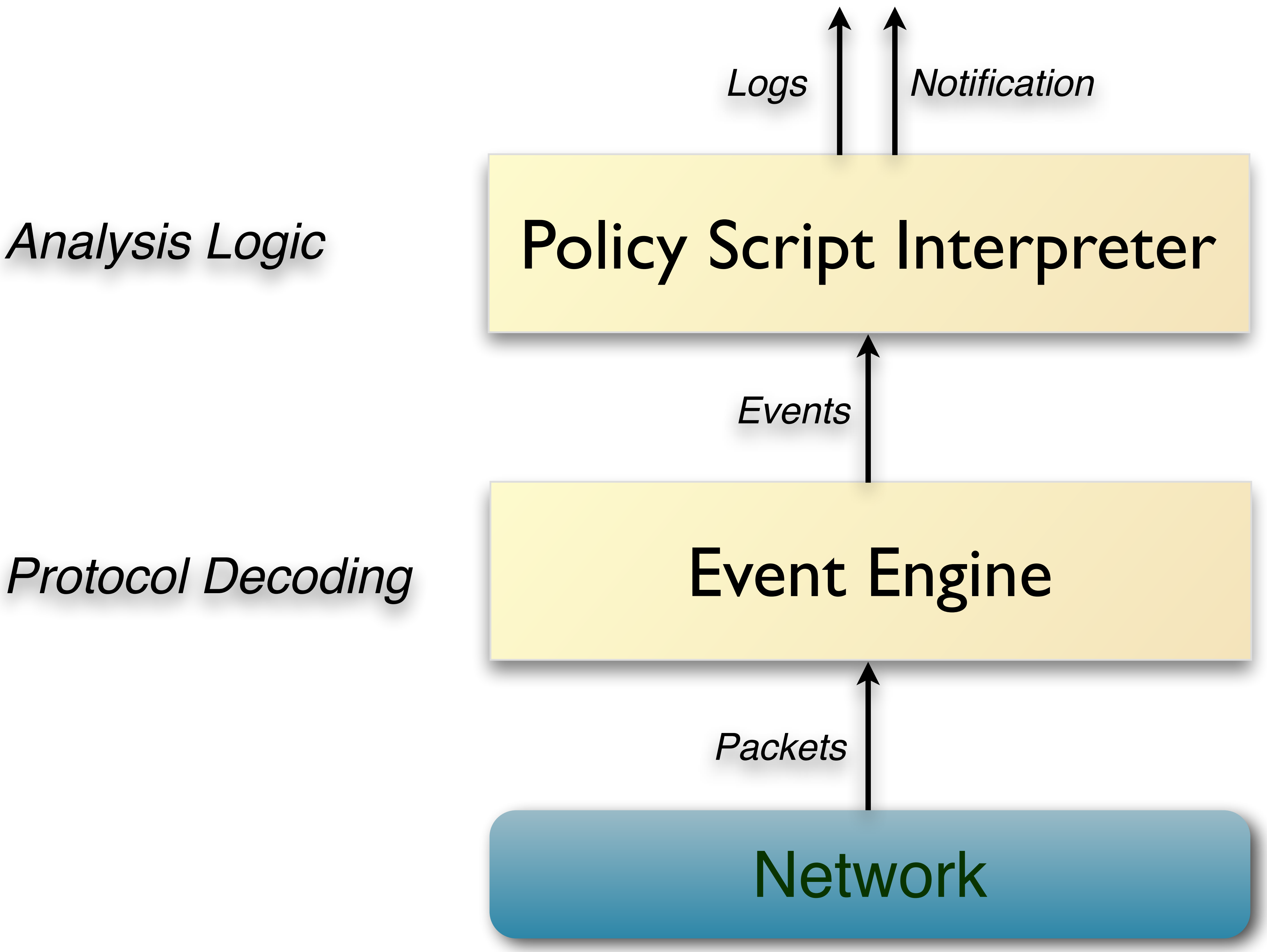


Architecture

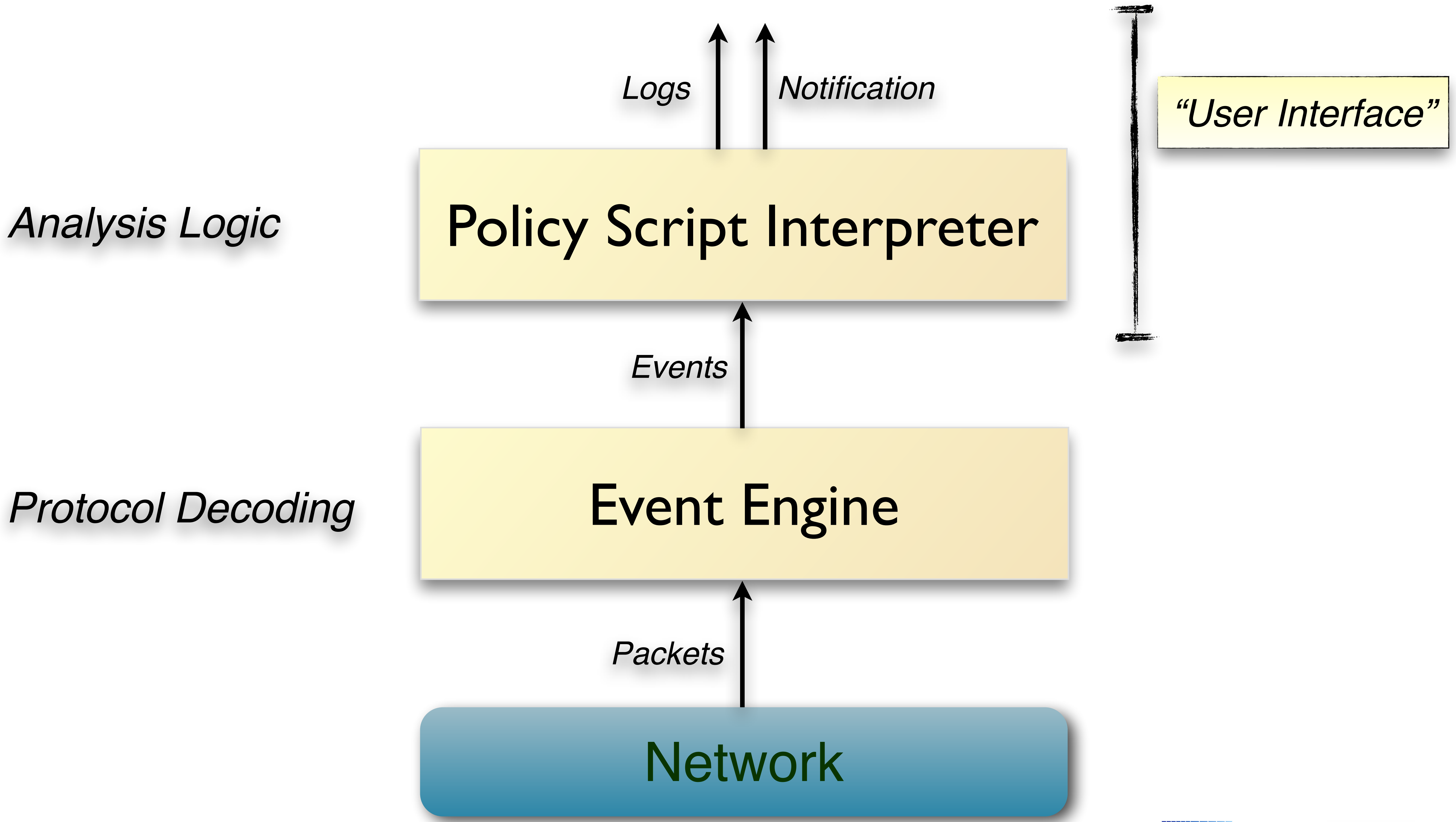
Protocol Decoding



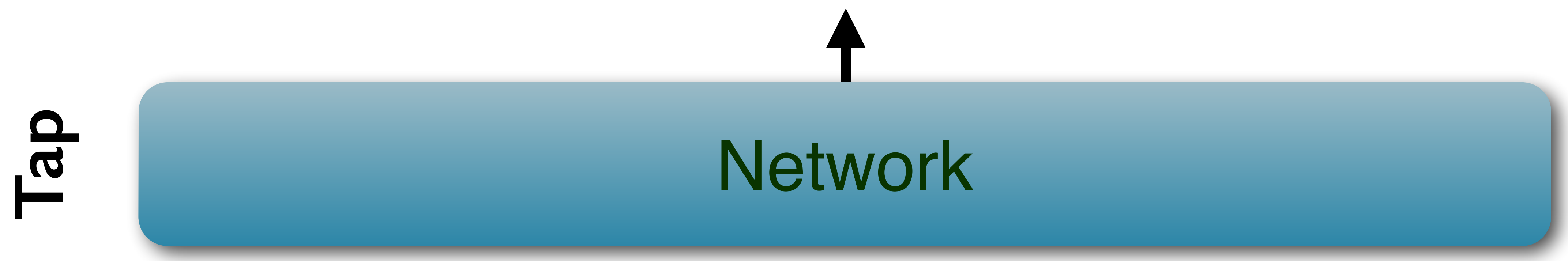
Architecture



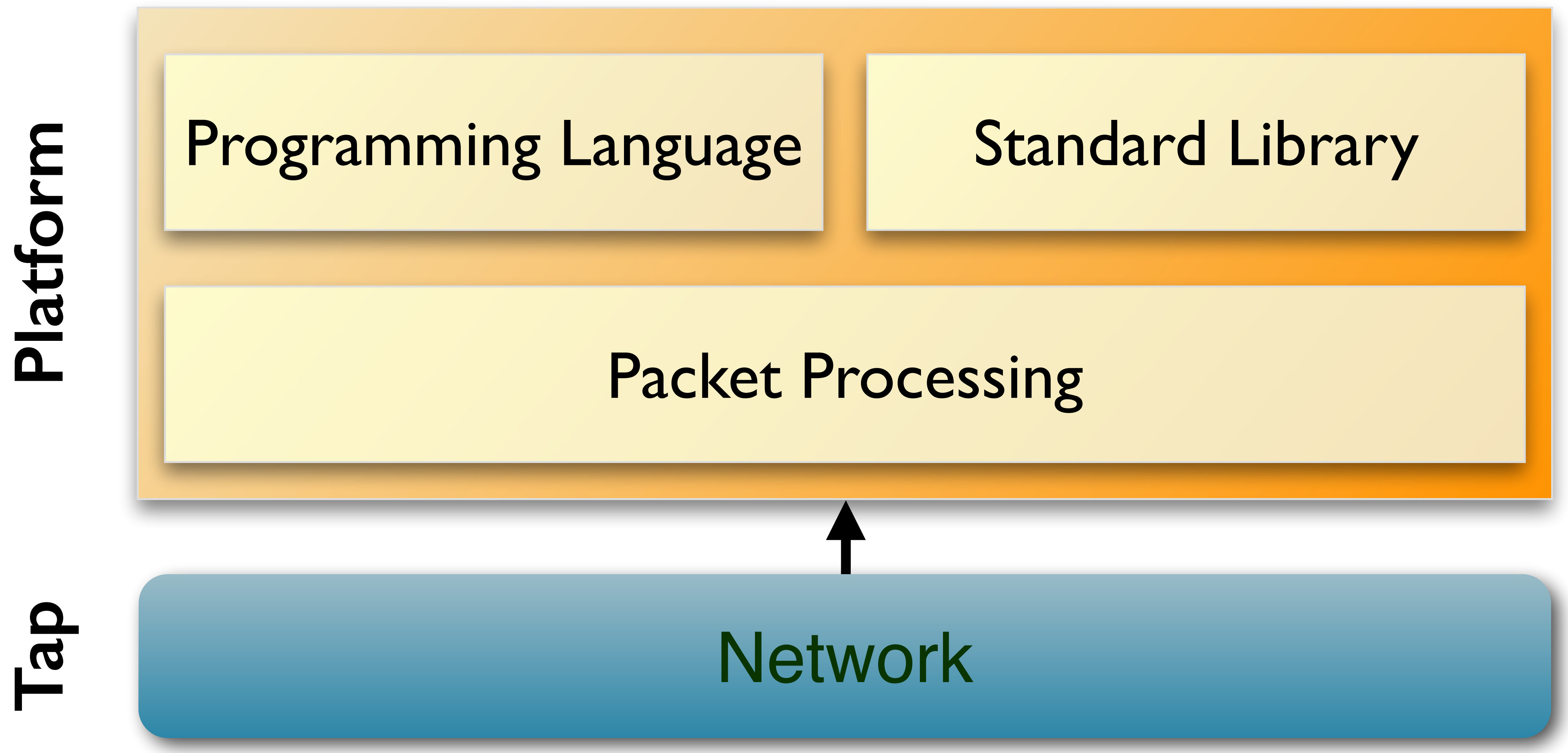
Architecture



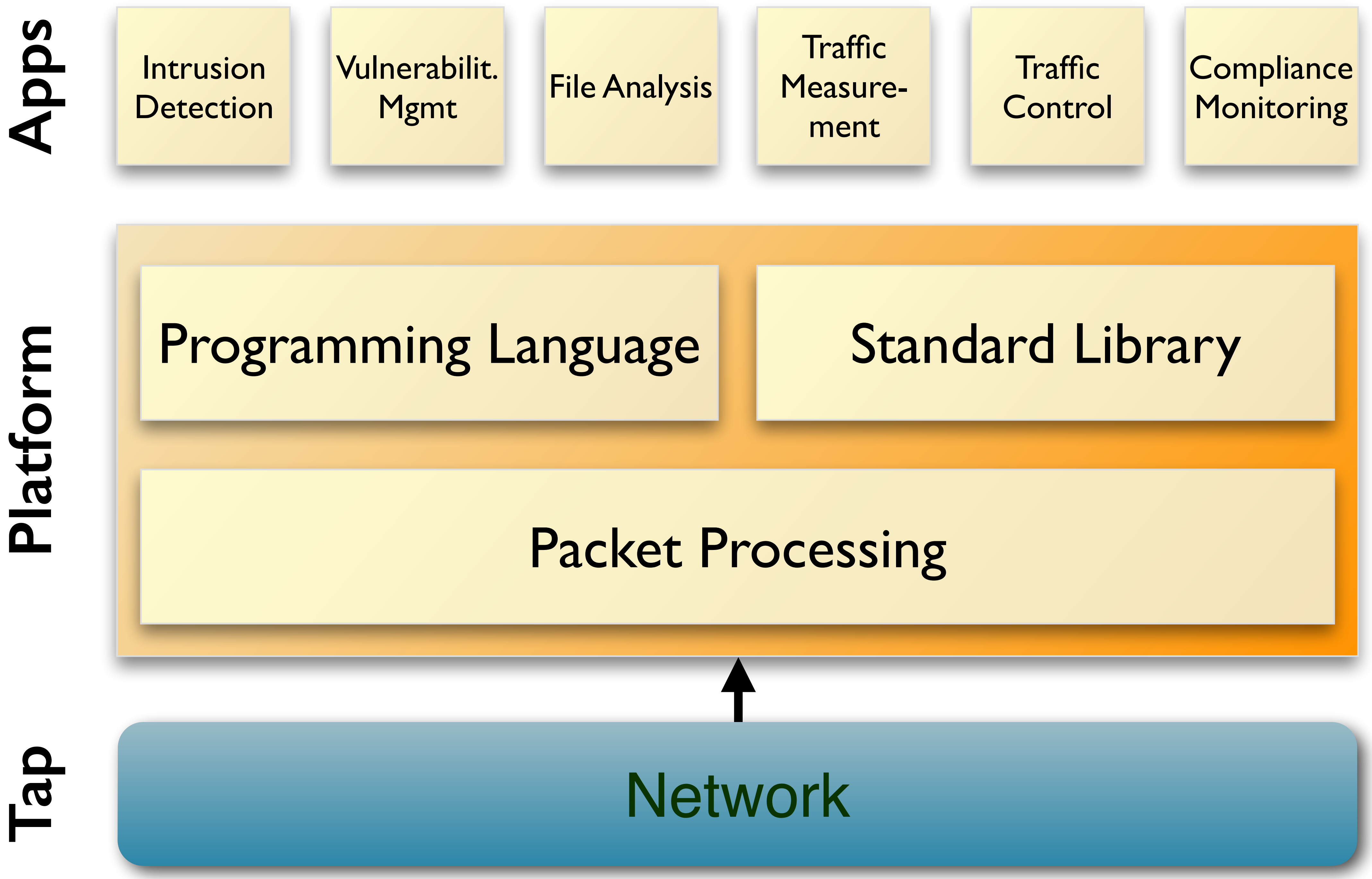
The Bro Platform



The Bro Platform

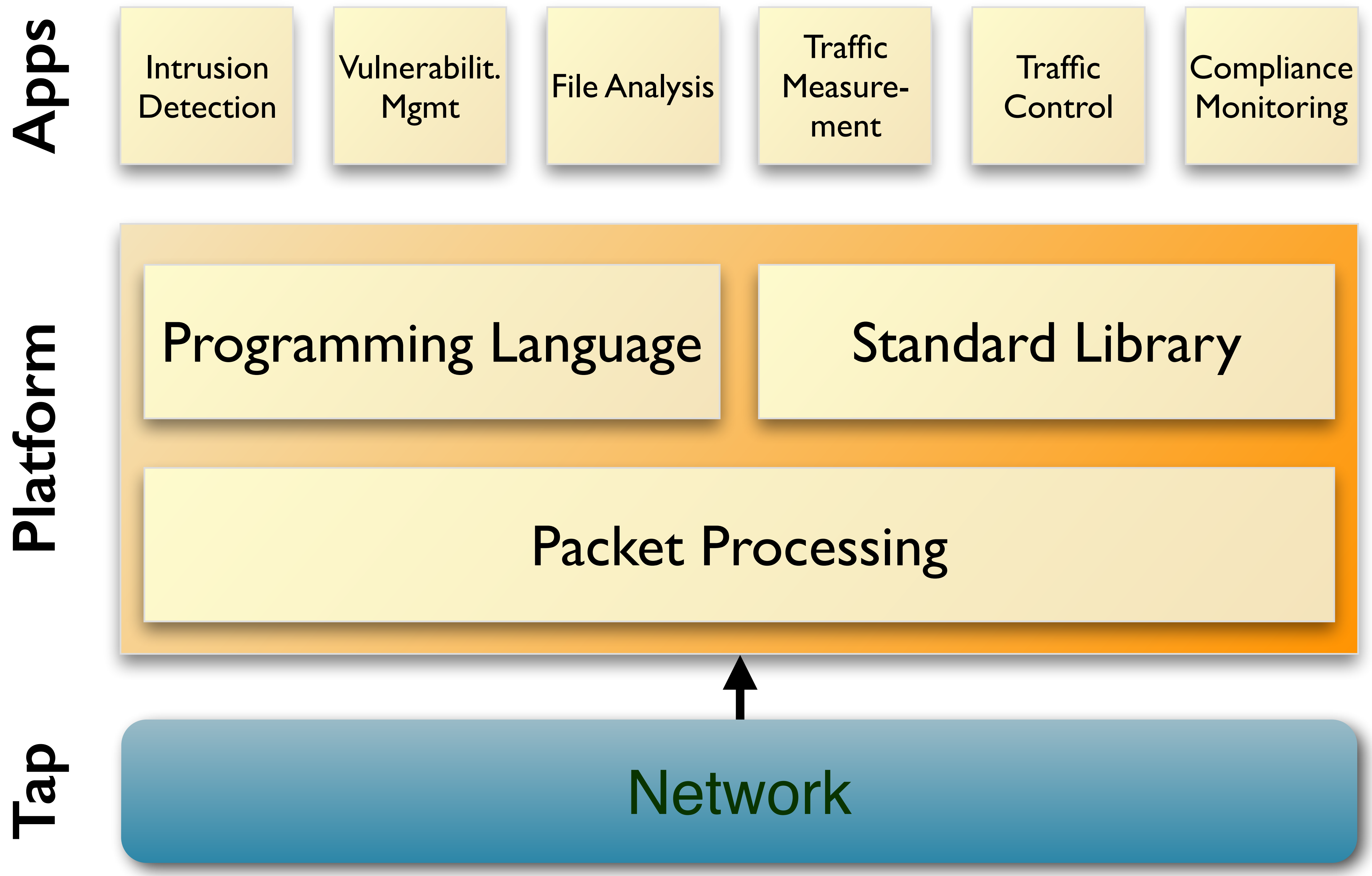


The Bro Platform



The Bro Platform

Open Source
BSD License



"What Can It Do?"

Log Files

Alerts

Custom Logic

"What Can It Do?"

Log Files

"Network Ground Truth"

Alerts

Custom Logic

Bro Logs

```
> bro -i eth0
[ … wait … ]
```

Bro Logs

```
> bro -i eth0
[ … wait … ]
> ls *.log
app_stats.log          irc.log                   socks.log
communication.log      known_certs.log           software.log
conn.log               known_hosts.log           ssh.log
dhcp.log               known_services.log        ssl.log
dns.log                modbus.log                syslog.log
dpd.log                notice.log                traceroute.log
files.log              reporter.log              tunnel.log
ftp.log                signatures.log            weird.log
http.log               smtp.log
```

Bro Logs

```
> bro -i eth0
[ … wait … ]
> cat conn.log
#separator \x09
#set_separator  ,
#empty_field    (empty)
#unset_field    -
#path   conn
#open   2013-04-28-23-47-26
#fields ts                uid          id.orig_h      id.orig_p   id.resp_h     […]
#types  time              string       addr           port        addr          […]
1258531221.486539  arKYeMETxOg  192.168.1.102  68          192.168.1.1   […]
1258531680.237254  nQcgTWjvg4c  192.168.1.103  37          192.168.1.255 […]
1258531693.816224  j4u32Pc5bif  192.168.1.102  37          192.168.1.255 […]
1258531635.800933  k6kgXLOoSKl  192.168.1.103  138         192.168.1.255 […]
1258531693.825212  TEfuqmmG4bh  192.168.1.102  138         192.168.1.255 […]
1258531803.872834  5OKnoww6xl4  192.168.1.104  137         192.168.1.255 […]
1258531747.077012  FrJExwHcSal  192.168.1.104  138         192.168.1.255 […]
1258531924.321413  3PKsZ2Uye21  192.168.1.103  68          192.168.1.1   […]
[…]
```

Connections Logs

conn.log

ts	1393099191.817686	Timestamp
uid	Cy3S2U2sbarorQgmw6a	Unique ID
id.orig_h	177.22.211.144	Originator IP
id.orig_p	43618	Originator Port
id.resp_h	115.25.19.26	Responder IP
id.resp_p	25	Responder Port
proto	tcp	IP Protocol
service	smtp	App-layer Protocol
duration	1.414936	Duration
orig_bytes	9068	Bytes by Originator
resp_bytes	4450	Bytes by Responder
conn_state	SF	TCP state
local_orig	T	Local Originator?
missed_bytes	0	Gaps
history	ShAdDaFf	State History
tunnel_parents	(empty)	Outer Tunnels

HTTP

http.log

ts	1393099291.589208
uid	CKFUW73bIADw0r9pl
id.orig_h	17.22.7.4
id.orig_p	54352
id.resp_h	24.26.13.36
id.resp_p	80
method	POST
host	com-services.pandonetworks.com
uri	/soapservices/services/SessionStart
referrer	-
user_agent	Mozilla/4.0 (Windows; U) Pando/2.6.0.8
status_code	200
username	anonymous
password	-
orig_mime_types	application/xml
resp_mime_types	application/xml

SSL

ssl.log

ts	1392805957.927087
uid	CEA0512D7k0BD9Dda2
id.orig_h	2a07:f2c0:90:402:41e:c13:6cb:99c
id.orig_p	40475
id.resp_h	2406:fe60:f47::aaeb:98c
id.resp_p	443
version	TLSv10
cipher	TLS_DHE_RSA_WITH_AES_256_CBC_SHA
server_name	www.netflix.com
subject	CN=www.netflix.com,OU=Operations, O=Netflix, Inc.,L=Los Gatos, ST=CALIFORNIA,C=US
issuer_subject	CN=VeriSign Class 3 Secure Server CA, OU=VeriSign Trust Network,O=VeriSign, C=US
not_valid_before	1389859200.000000
not_valid_after	1452931199.000000
client_subject	-
client_issuer_subject	-
cert_hash	197cab7c6c92a0b9ac5f37cfb0699268
validation_status	ok

Syslog & DHCP

syslog.log

ts	1392796803.311801
uid	CnYivt3Z0NHOuBALR8
id.orig_h	12.3.8.161
id.orig_p	514
id.resp_h	16.74.12.24
id.resp_p	514
proto	udp
facility	AUTHPRIV
severity	INFO
message	sshd[13825]: Accepted publickey for harvest from xxx.xxx.xxx.xxx

Syslog & DHCP

syslog.log

ts	1392796803.311801
uid	CnYivt3Z0NHOuBALR8
id.orig_h	12.3.8.161
id.orig_p	514
id.resp_h	16.74.12.24
id.resp_p	514
proto	udp
facility	AUTHPRIV
severity	INFO
message	sshd[13825]: Accepted publickey for harvest from xxx.xxx.xxx.xxx

dhcp.log

ts	1392796962.091566
uid	Ci3RM24iF4vIYRGHc3
id.orig_h	10.129.5.11
id.resp_h	10.129.5.1
mac	04:12:38:65:fa:68
assigned_ip	10.129.5.11
lease_time	14400.000000

Files

files.log

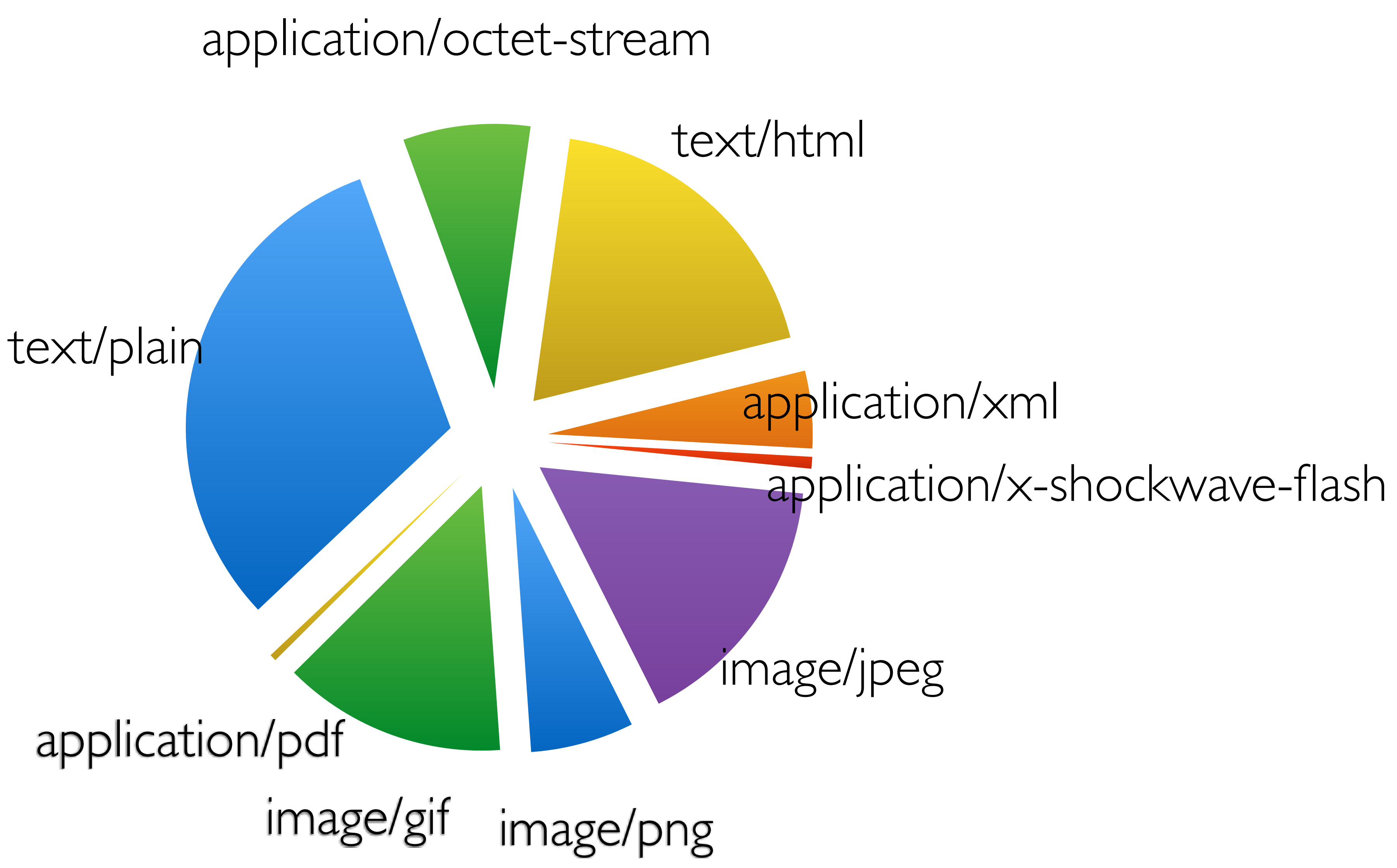
ts	1392797643.447056
fuid	FnungQ3TI19GahPJP2
tx_hosts	191.168.187.33
rx_hosts	10.1.29.110
conn_uids	CbDgik2fjeKL5qzn55
source	SMTP
analyzers	SHA1,MD5
mime_type	application/x-dosexec
filename	Letter.exe
duration	5.320822
local_orig	T
seen_bytes	39508
md5	93f7f5e7a2096927e06e[…]1085bfcfb
sha1	daed94a5662a920041be[…]a433e501646ef6a03
extracted	-

Software

software.log

ts	1392796839.675867
host	10.209.100.2
host_p	-
software_type	HTTP::BROWSER
name	DropboxDesktopClient
version.major	2
version.minor	4
version.minor2	11
version.minor3	-
version.addl	Windows
unparsed_version	DropboxDesktopClient/2.4.11 (Windows; 8; i32; en_US; Trooper 5694-2047-1832-6291-8315)

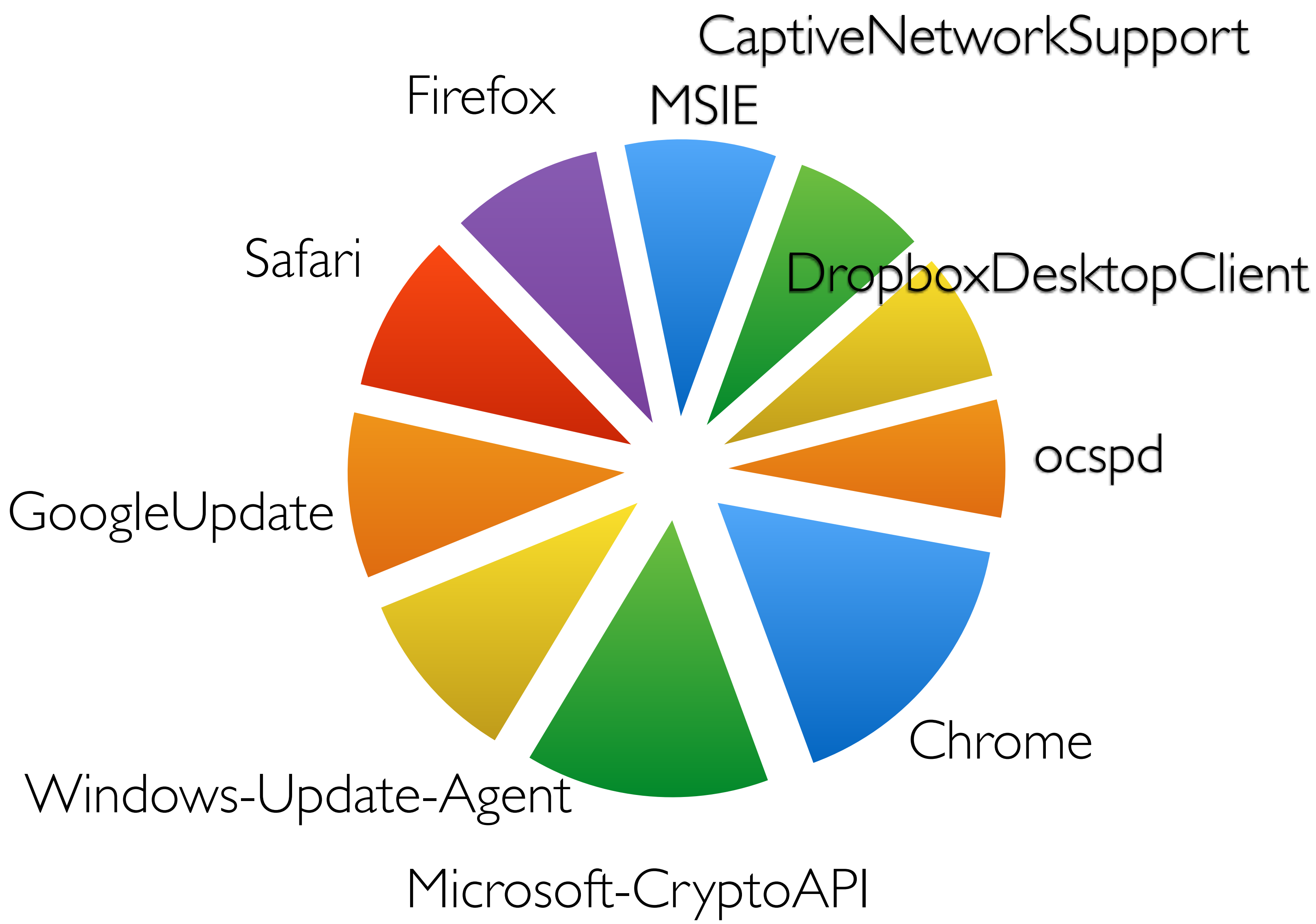
Help Understand Your Network



```
cat files.log | bro-cut mime_type | sort | uniq -c | sort -rn
```

Help Understand Your Network (2)

Top Software by Number of Hosts



```
cat software.log  | bro-cut host name | sort | uniq |
awk -F '\t' '{print $2}' | sort | uniq -c | sort -rn
```

"What Can It Do?"

Log Files

Alerts

Custom Logic

"What Can It Do?"

Log Files

Alerts

"Watch this!"

Recorded in notice.log.
Can trigger actions.

Custom Logic

Alerts in Bro 2.2

CaptureLoss::Too_Much_Loss
Conn::Ack_Above_Hole
Conn::Content_Gap
Conn::Retransmission_Inconsistency
DNS::External_Name
FTP::Bruteforcing
FTP::Site_Exec_Success
HTTP::SQL_Injection_Attacker
HTTP::SQL_Injection_Victim
Intel::Notice
PacketFilter::Dropped_Packets
ProtocolDetector::Protocol_Found
ProtocolDetector::Server_Found
SMTP::Blocklist_Blocked_Host
SMTP::Blocklist_Error_Message
SMTP::Suspicious_Origination
SSH::Interesting_Hostname_Login
SSH::Login_By_Password_Guesser
SSH::Password_Guessing
SSH::Watched_Country_Login
SSL::Certificate_Expired
SSL::Certificate_Expires_Soon
SSL::Certificate_Not_Valid_Yet
SSL::Invalid_Server_Cert
Scan::Address_Scan
Scan::Port_Scan
Signatures::Count_Signature
Signatures::Multiple_Sig_Responders
Signatures::Multiple_Signatures
Signatures::Sensitive_Signature
Software::Software_Version_Change
Software::Vulnerable_Version
TeamCymruMalwareHashRegistry::Match
Traceroute::Detected
Weird::Activity

Watching for Suspicious Logins

Watching for Suspicious Logins

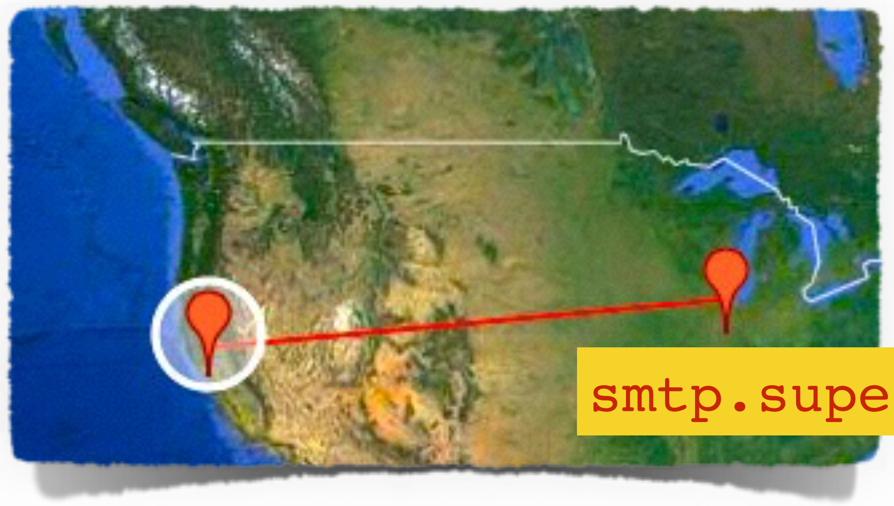
`SSH::Watched_Country_Login`

Login from an unexpected country.

Watching for Suspicious Logins

`SSH::Watched_Country_Login`

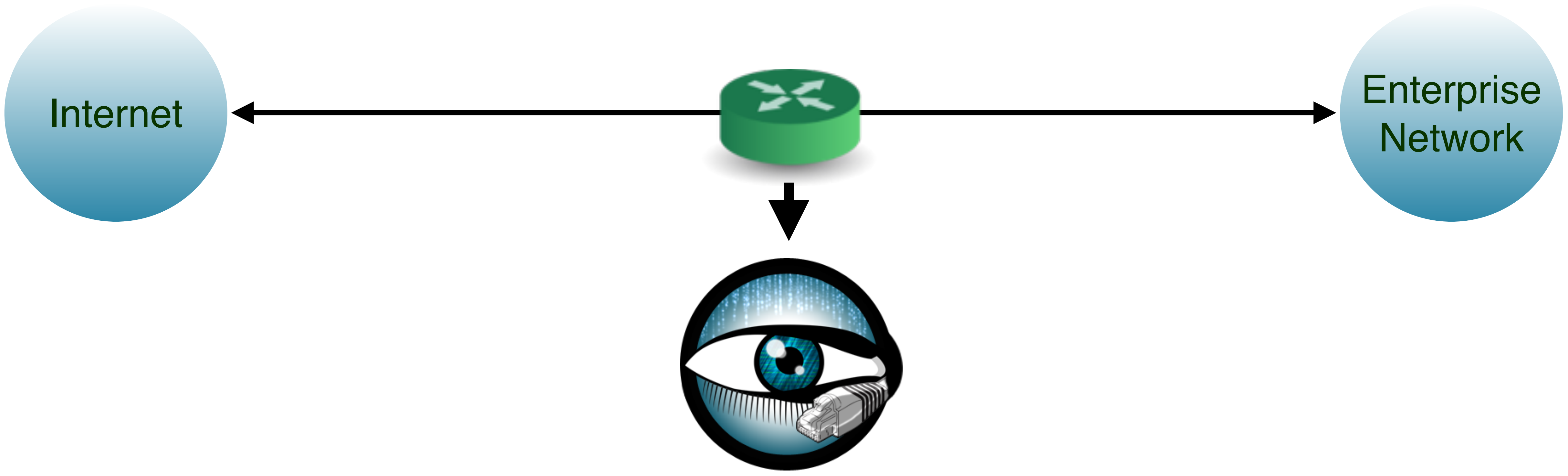
Login from an unexpected country.



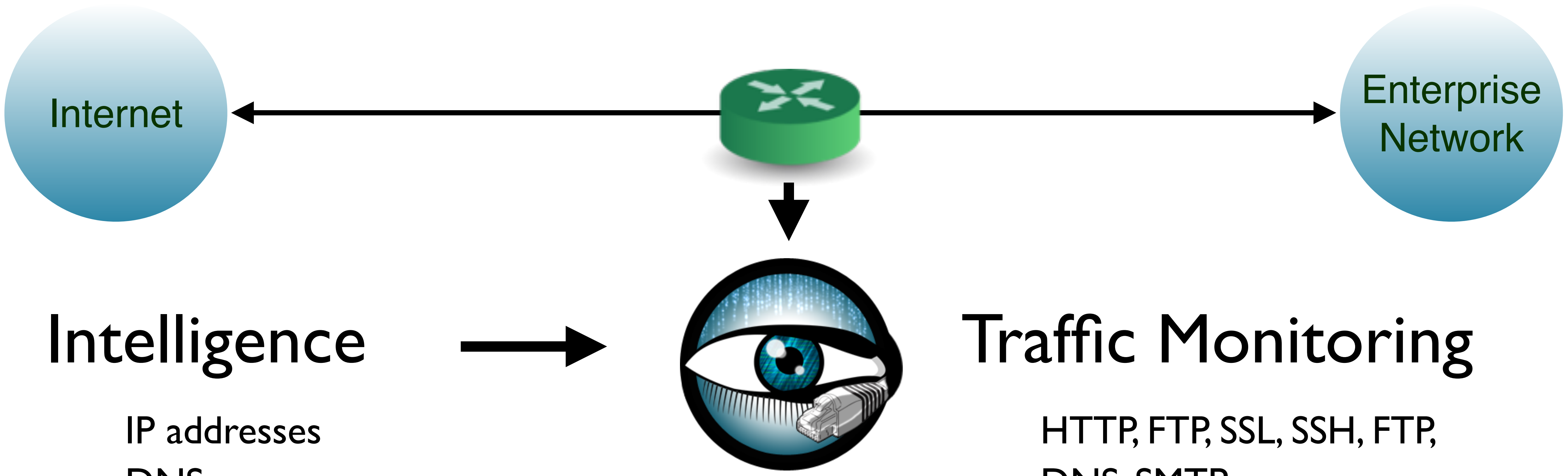
`SSH::Interesting_Hostname_Login`

Login from an unusual host name.

Intelligence Integration (Passive)



Intelligence Integration (Passive)



Intelligence

IP addresses
DNS names
URLs
File hashes

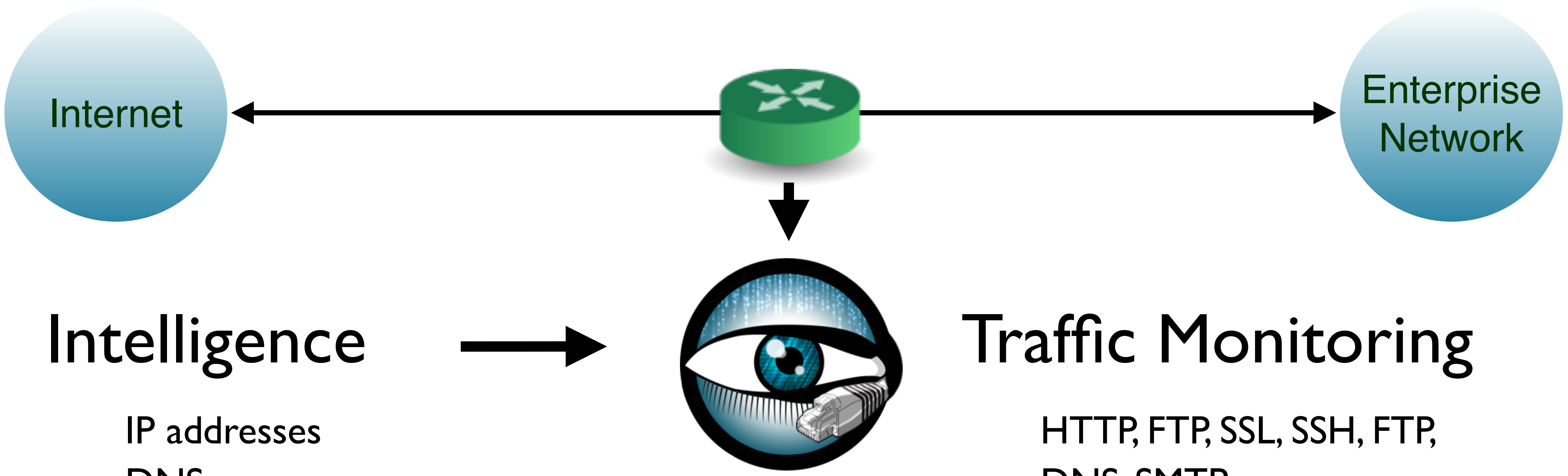
Traffic Monitoring

HTTP, FTP, SSL, SSH, FTP, DNS, SMTP, …

Feeds

CIF
JC3
Spamhaus
Custom/Proprietary

Intelligence Integration (Passive)



Intelligence

- IP addresses
- DNS names
- URLs
- File hashes

Feeds

- CIF
- JC3
- Spamhaus
- *Custom/Proprietary*

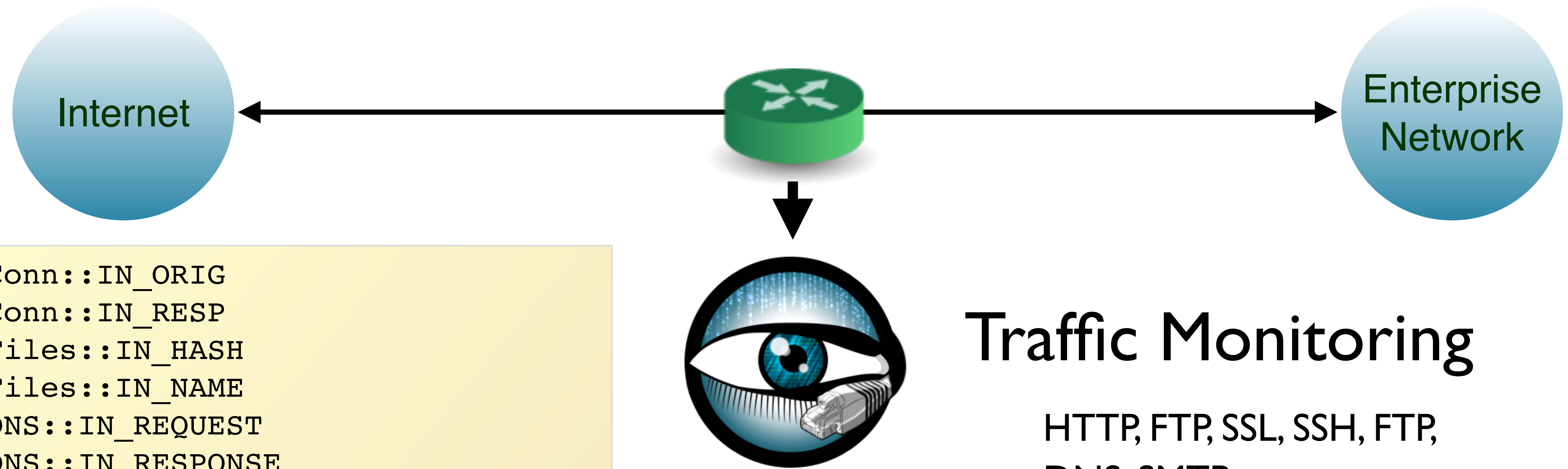
Traffic Monitoring

HTTP, FTP, SSL, SSH, FTP, DNS, SMTP, …

ts	1258565309.806483
uid	CAK677xaOmi66X4Th
id.orig_h	192.168.1.103
id.resp_h	192.168.1.1
note	Intel::Notice
indicator	baddomain.com
indicator_type	Intel::DOMAIN
where	HTTP::IN_HOST_HEADER
source	My-Private-Feed

notice.log

Intelligence Integration (Passive)



```
Conn::IN_ORIG
Conn::IN_RESP
Files::IN_HASH
Files::IN_NAME
DNS::IN_REQUEST
DNS::IN_RESPONSE
HTTP::IN_HOST_HEADER
HTTP::IN_REFERRER_HEADER
HTTP::IN_USER_AGENT_HEADER
HTTP::IN_X_FORWARDED_FOR_HEADER
HTTP::IN_URL
SMTP::IN_MAIL_FROM
SMTP::IN_RCPT_TO
SMTP::IN_FROM
SMTP::IN_TO
SMTP::IN_RECEIVED_HEADER
SMTP::IN_REPLY_TO
SMTP::IN_X_ORIGINATING_IP_HEADER
SMTP::IN_MESSAGE
SSL::IN_SERVER_CERT
SSL::IN_CLIENT_CERT
SSL::IN_SERVER_NAME
SMTP::IN_HEADER
```

ts	1258565309.806483
uid	CAK677xaOmi66X4Th
id.orig_h	192.168.1.103
id.resp_h	192.168.1.1
note	Intel::Notice
indicator	baddomain.com
indicator_type	Intel::DOMAIN
where	HTTP::IN_HOST_HEADER
source	My-Private-Feed

notice.log

Intelligence Integration (Active)

TEAM CYMRU COMMUNITY SERVICES

Intelligence Integration (Active)

TEAM CYMRU COMMUNITY SERVICES

```
# cat files.log | bro-cut mime_type sha1 | awk '$1 ~ /x-dosexec/‘
application/x-dosexec   5fd2f37735953427e2f6c593d6ec7ae882c9ab54
application/x-dosexec   00c69013d34601c2174b72c9249a0063959da93a
application/x-dosexec   0d801726d49377bfe989dcca7753a62549f1ddda
[…]
```

Intelligence Integration (Active)

TEAM CYMRU
COMMUNITY
SERVICES

```
# cat files.log | bro-cut mime_type sha1 | awk '$1 ~ /x-dosexec/'
application/x-dosexec   5fd2f37735953427e2f6c593d6ec7ae882c9ab54
application/x-dosexec   00c69013d34601c2174b72c9249a0063959da93a
application/x-dosexec   0d801726d49377bfe989dcca7753a62549f1ddda
[…]
```

```
# dig +short 733a48a9cb4[…]2a91e8d00.malware.hash.cymru.com TXT
"1221154281 53"
```

Intelligence Integration (Active)

```
# cat files.log | bro-cut mime_type sha1 | awk '$1 ~ /x-dosexec/'
application/x-dosexec   5fd2f37735953427e2f6c593d6ec7ae882c9ab54
application/x-dosexec   00c69013d34601c2174b72c9249a0063959da93a
application/x-dosexec   0d801726d49377bfe989dcca7753a62549f1ddda
[…]
```

```
# dig +short 733a48a9cb4[…]2a91e8d00.malware.hash.cymru.com TXT
"1221154281 53"
```

notice.log

ts	**1392423980.736470**	Timestamp
uid	**CjKeSB45xaOmiIo4Th**	Connection ID
id.orig_h	**10.2.55.3**	Originator IP
id.resp_h	**192.168.34.12**	Responder IP
fuid	**FEGVbAgcArRQ49347**	File ID
mime_type	**application/jar**	MIME type
description	**http://app.looking3g.com/[…]**	Source URL Bro saw
note	**TeamCymruMalwareHashRegistry::Match**	Notice Type
msg	**2013-09-14 22:06:51 / 20%**	MHR reply
sub	**https://www.virustotal.com/[…]**	VirusTotal URL

TEAM CYMRU COMMUNITY SERVICES

"What Can It Do?"

Log Files

Alerts

Custom Logic

"What Can It Do?"

Log Files

Alerts

Custom Logic

"Don't ask what Bro can do.
*Ask what **you want** it to do."*

Script Example: Matching URLs

Task: Report all Web requests for files called "passwd".

Script Example: Matching URLs

Task: Report all Web requests for files called "passwd".

```
event http_request(c: connection,            # Connection.
                   method: string,           # HTTP method.
                   original_URI: string,     # Requested URL.
                   unescaped_URI: string,    # Decoded URL.
                   version: string)          # HTTP version.
{
    if ( method == "GET" && unescaped_URI == /.*passwd/ )
        NOTICE(...); # Alarm.
}
```

Script Example: Scan Detector

Task: Count failed connection attempts per source address.

Script Example: Scan Detector

Task: Count failed connection attempts per source address.

```
global attempts: table[addr] of count &default=0;

event connection_rejected(c: connection)
{
    local source = c$id$orig_h;       # Get source address.

    local n = ++attempts[source];     # Increase counter.

    if ( n == SOME_THRESHOLD )        # Check for threshold.
        NOTICE(...);                  # Alarm.
}
```

Scripts are Bro's "Magic Ingredient"

Bro comes with >10,000 lines of script code.

Prewritten functionality that's just loaded.

Scripts generate everything we have seen.

Amendable to extensive customization and extension.

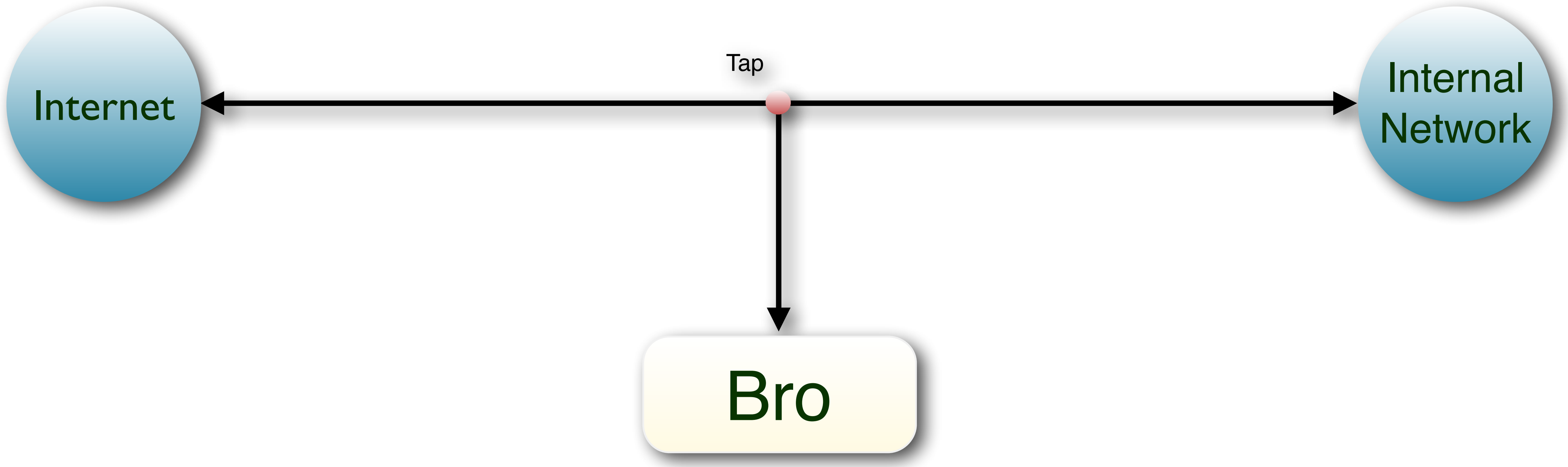
Growing community writing 3rd party scripts.

Bro could report Mandiant's APT1 indicators within a day.

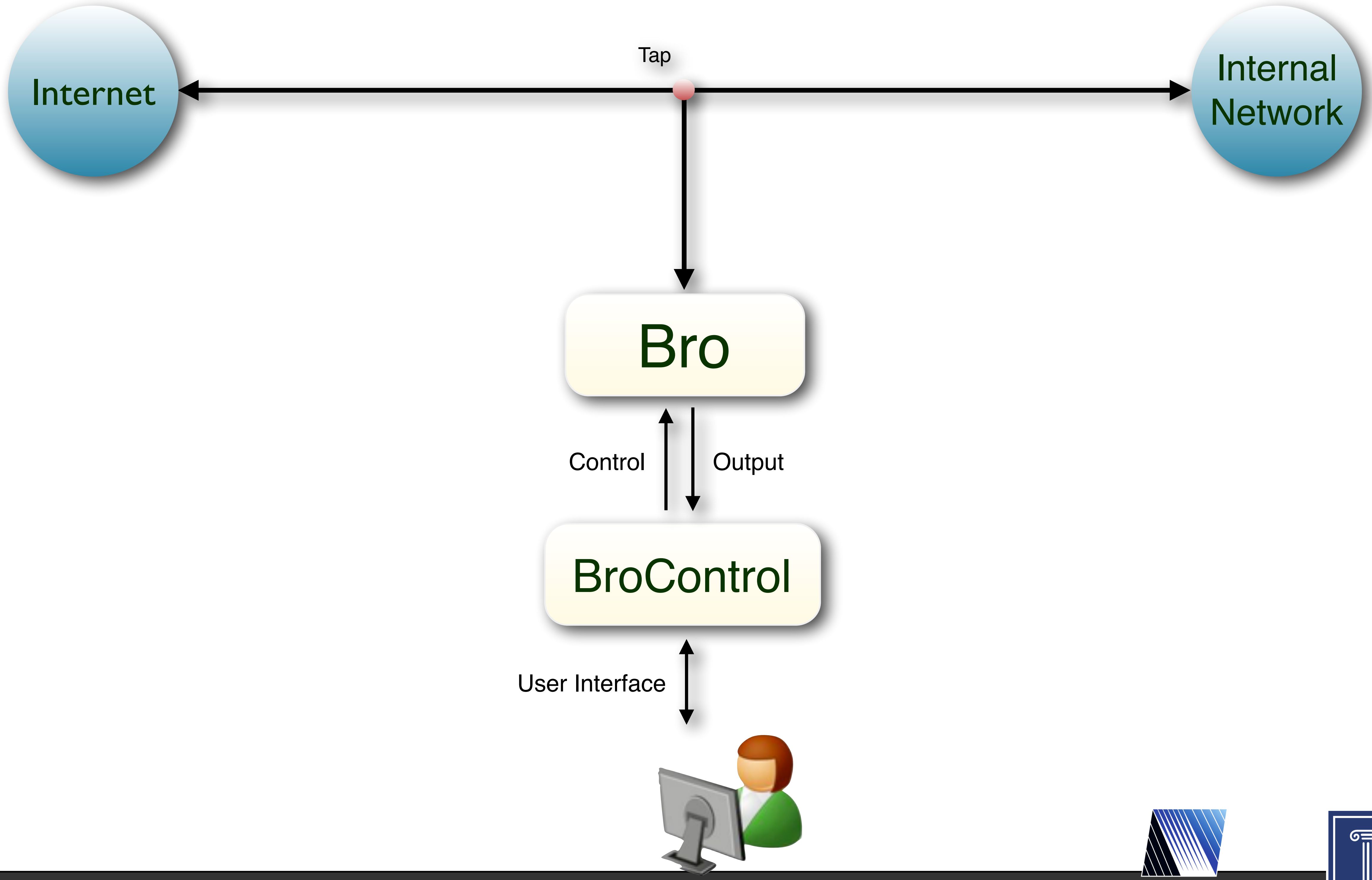
Same for Heartbleed

Bro Ecosystem

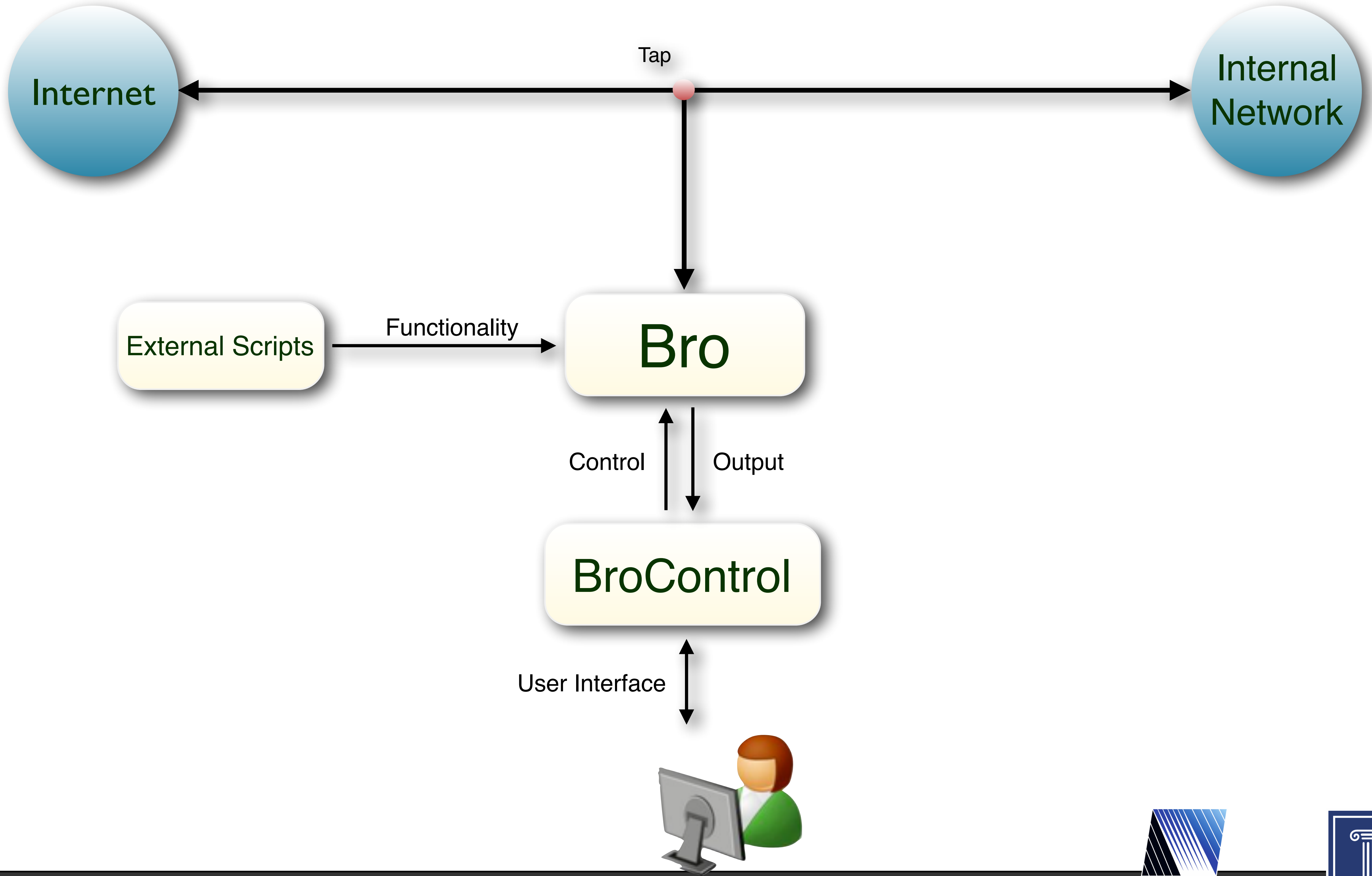
Bro Ecosystem



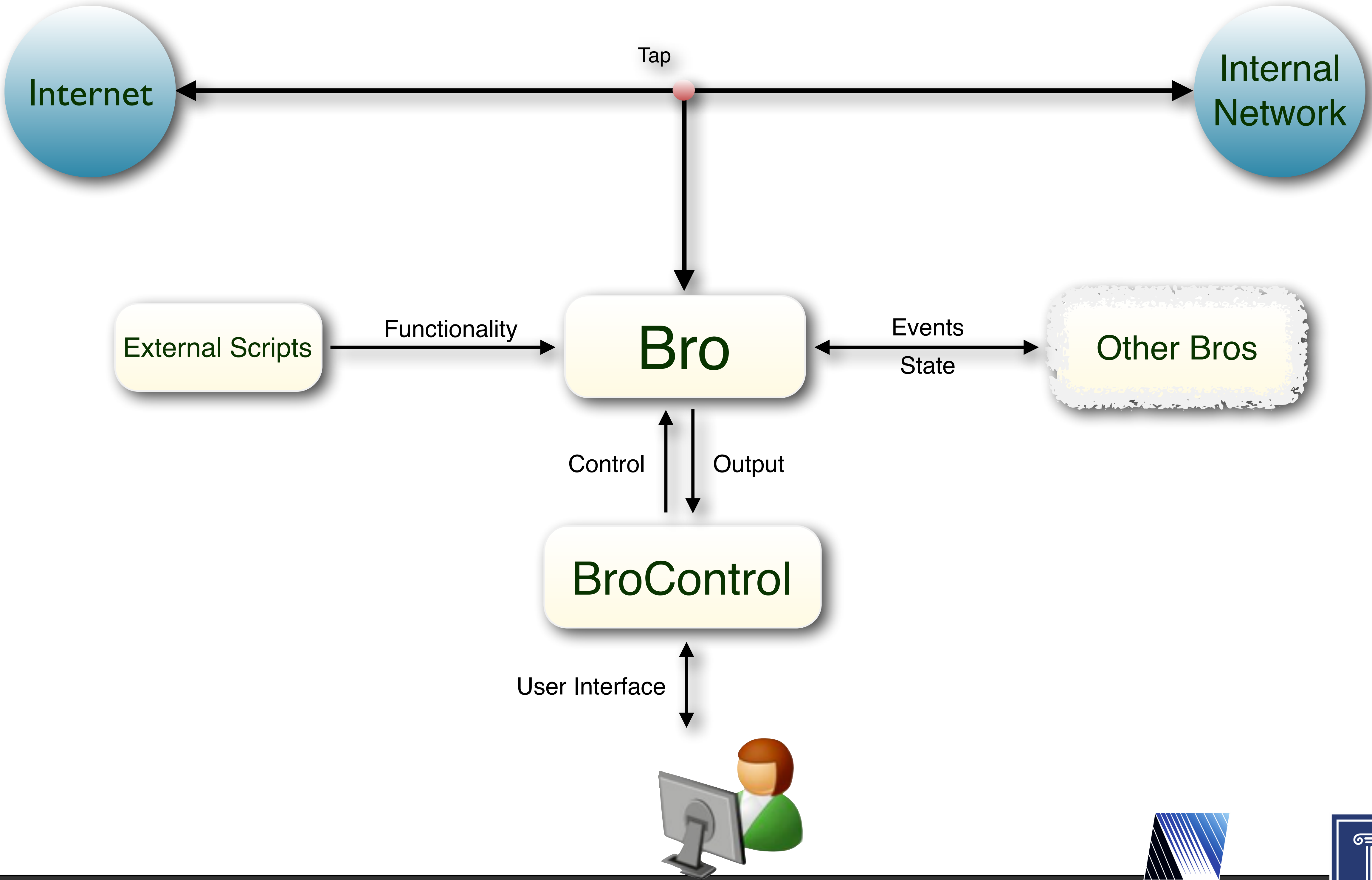
Bro Ecosystem



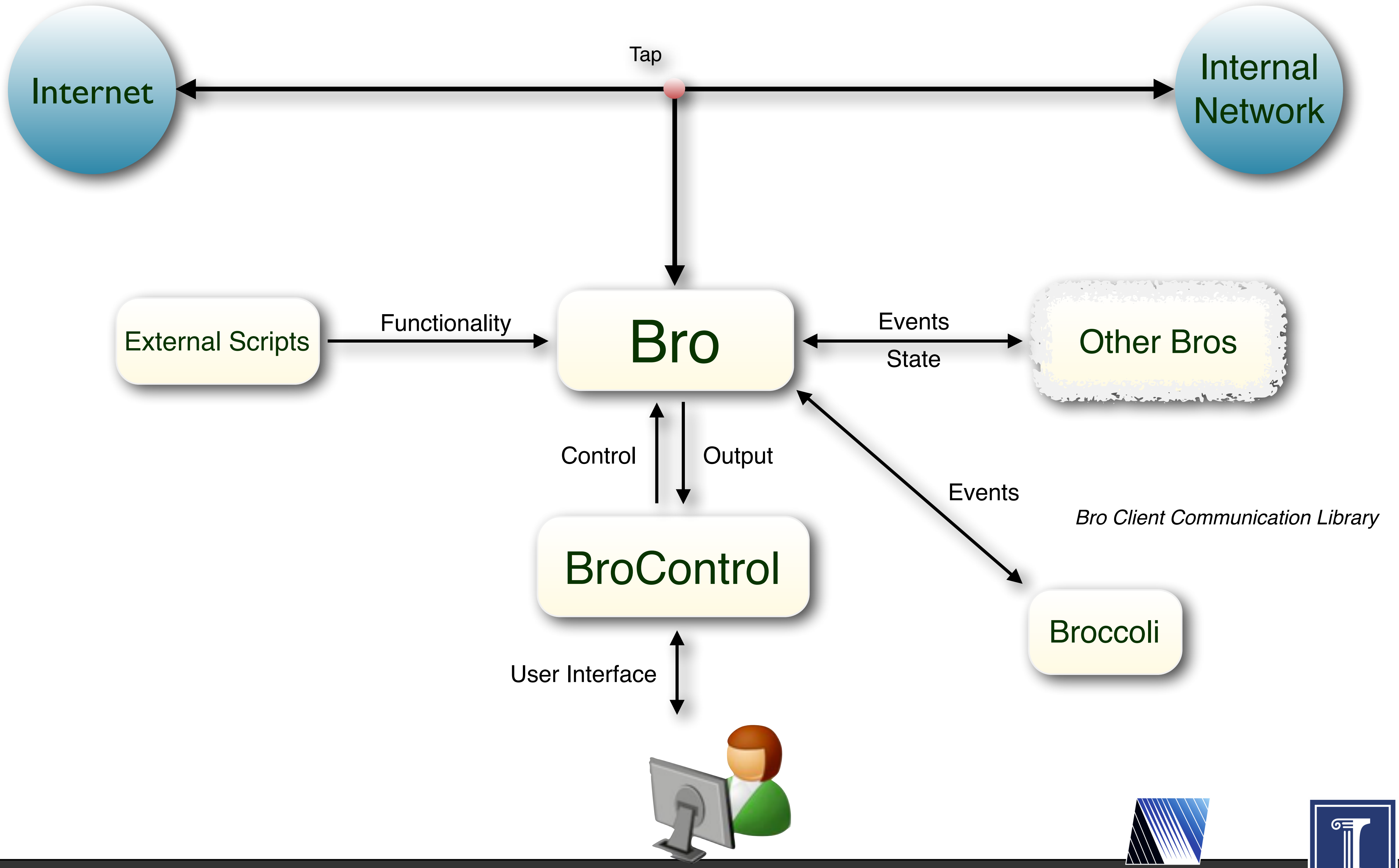
Bro Ecosystem



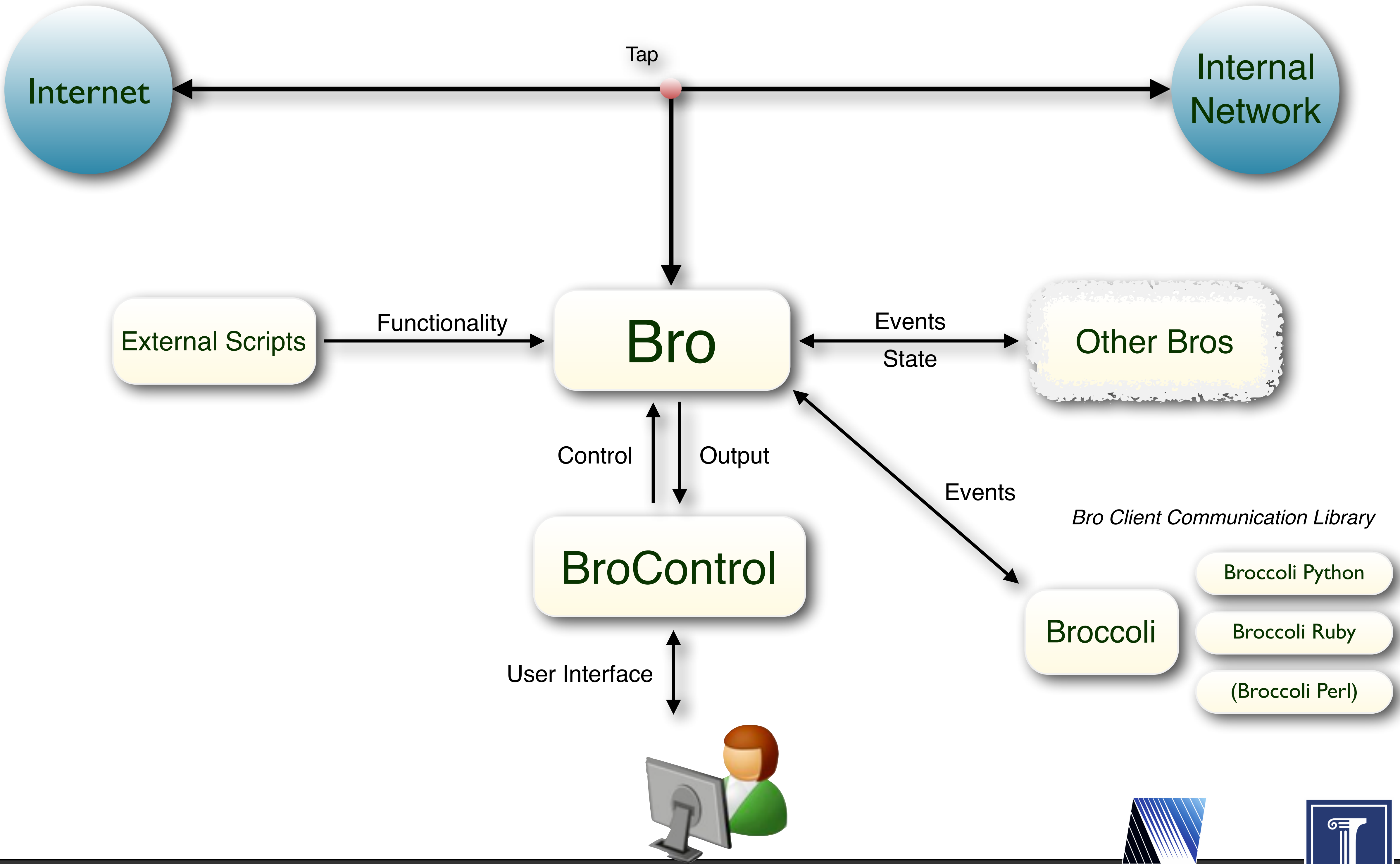
Bro Ecosystem



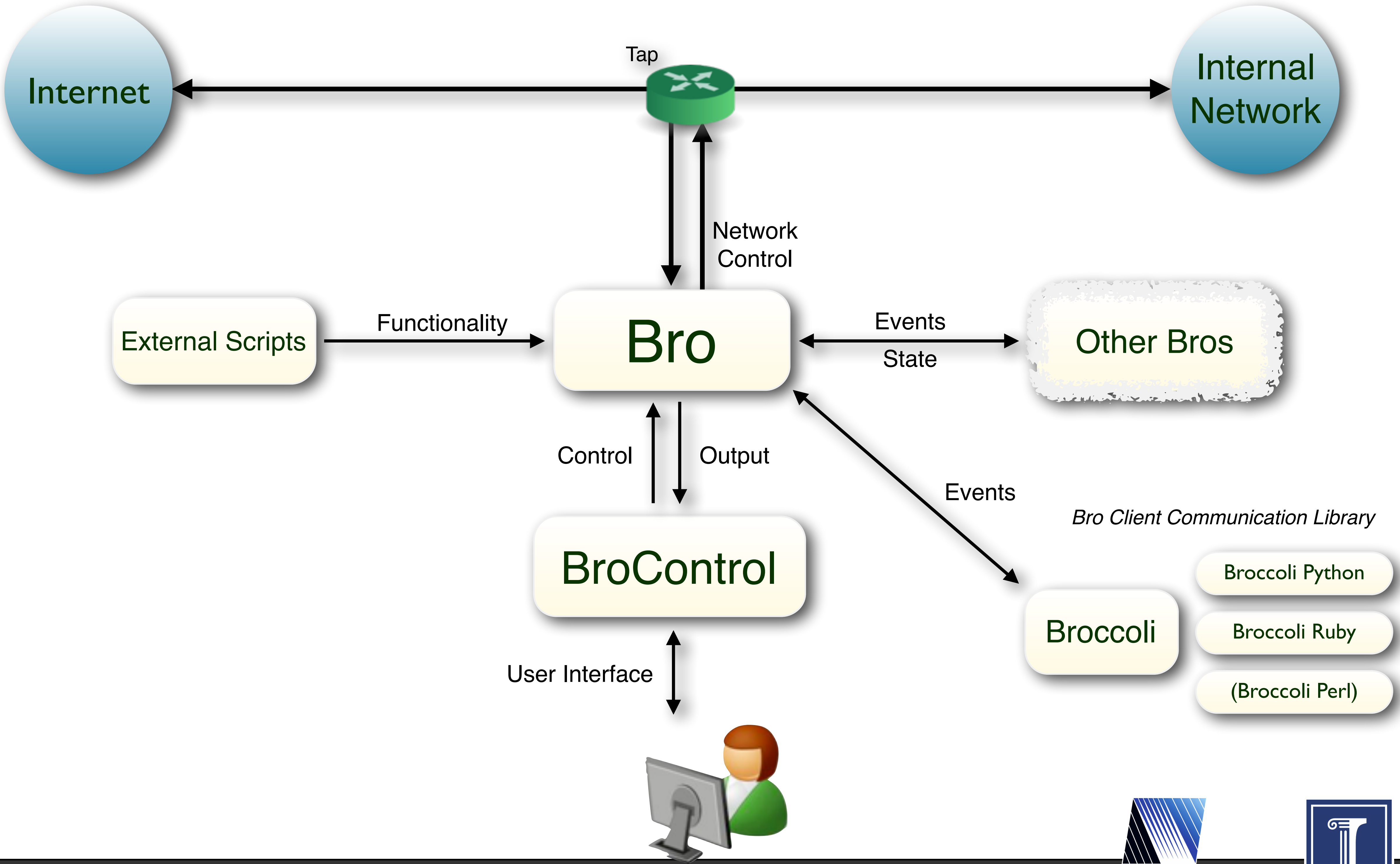
Bro Ecosystem



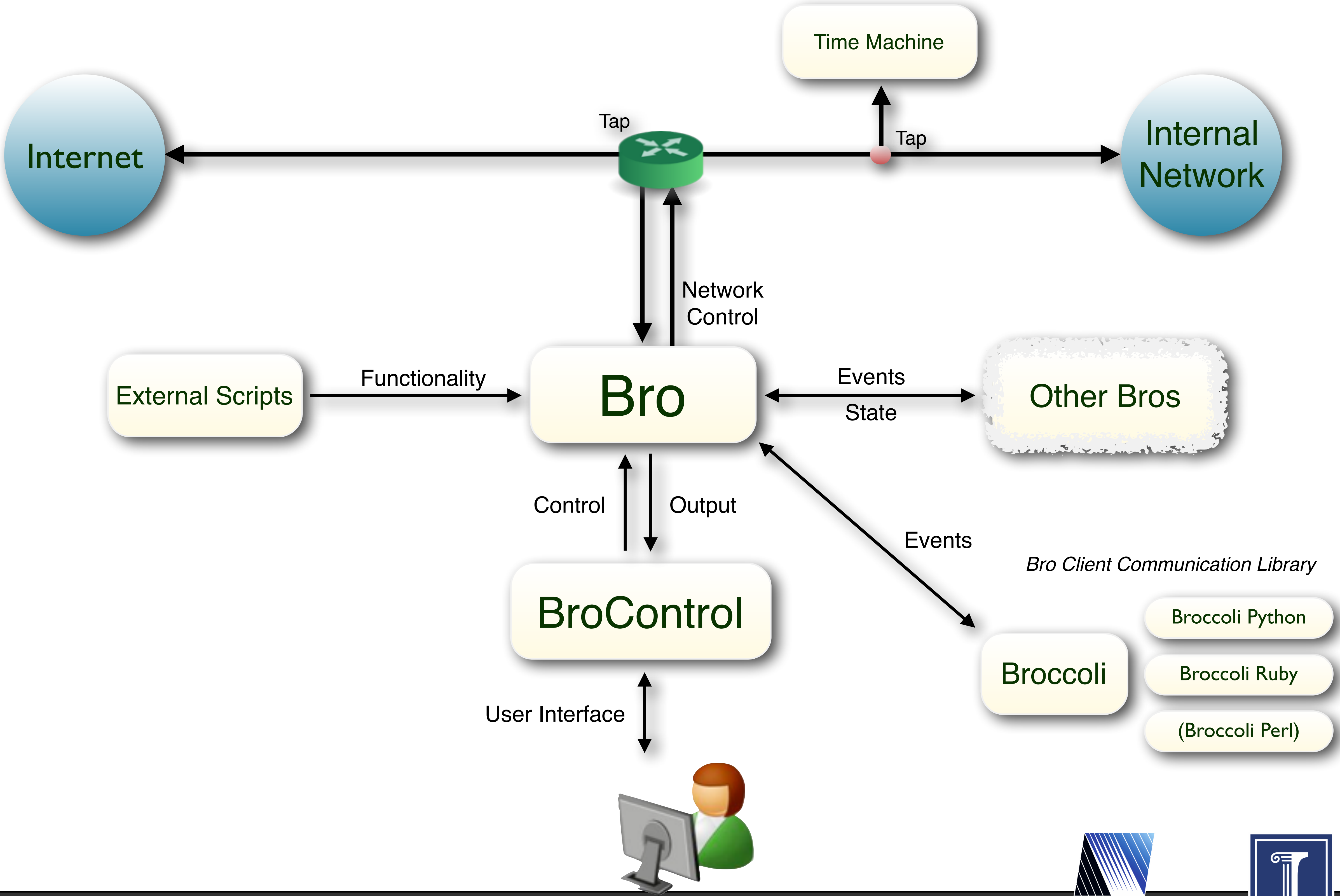
Bro Ecosystem



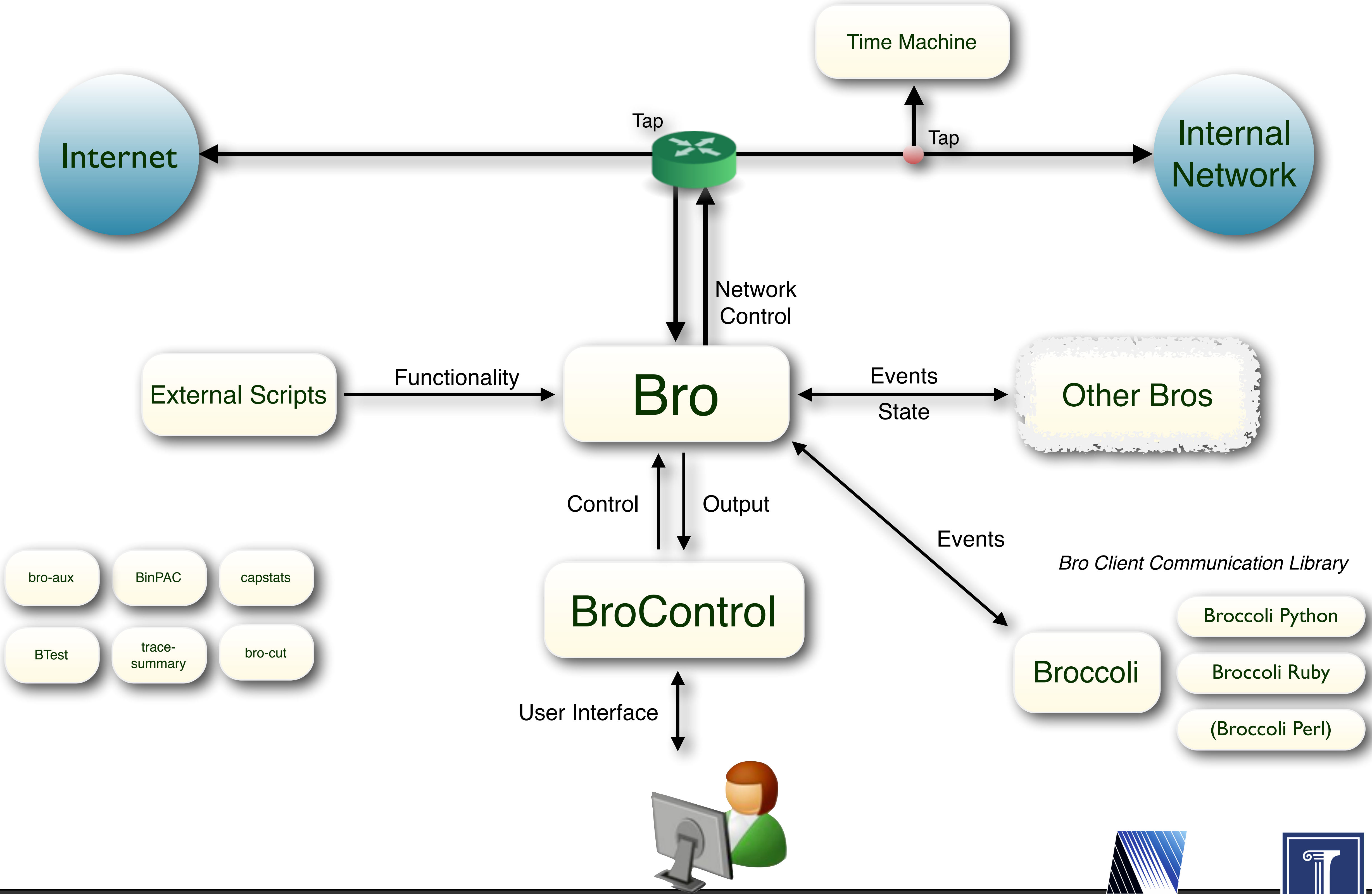
Bro Ecosystem



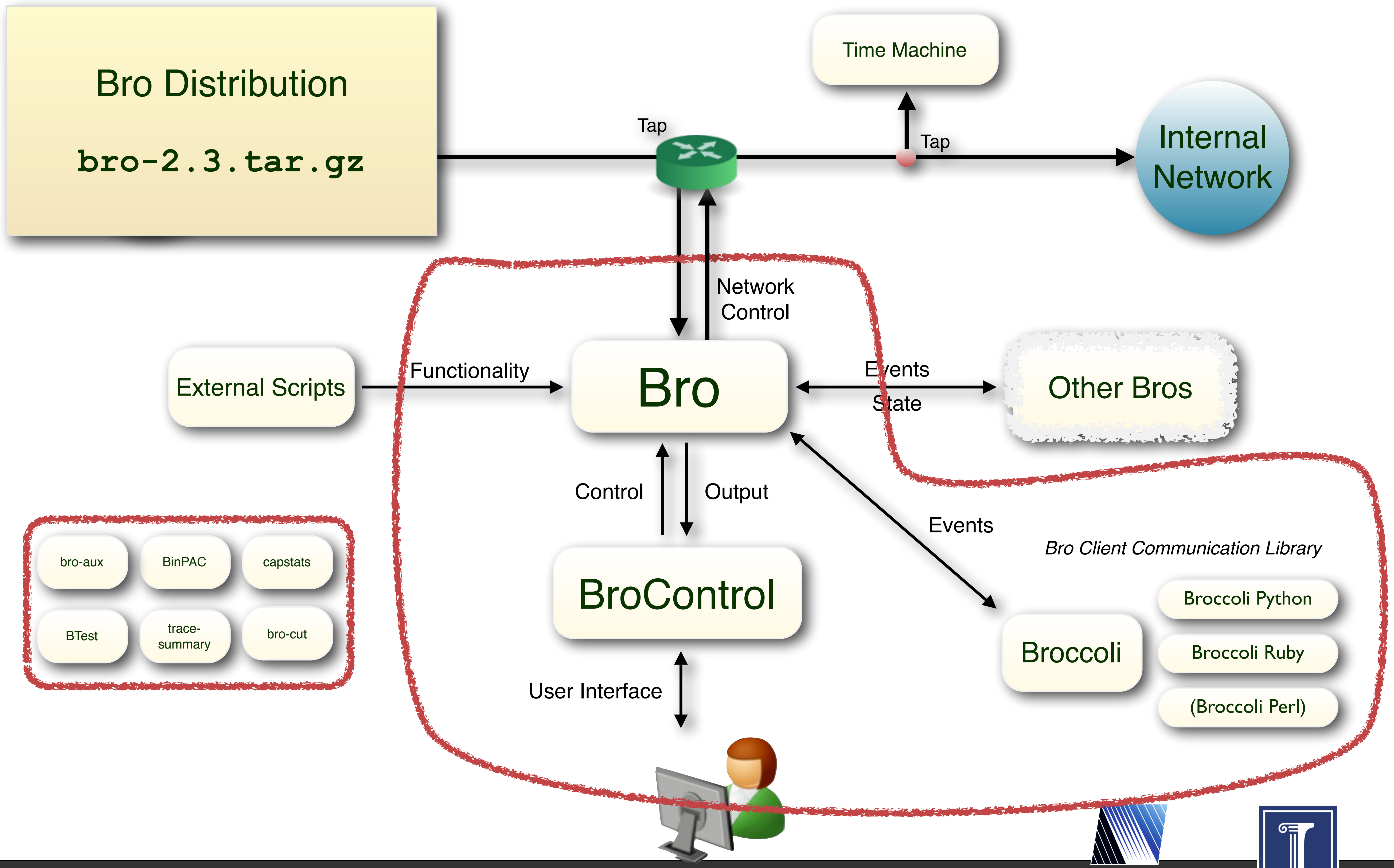
Bro Ecosystem



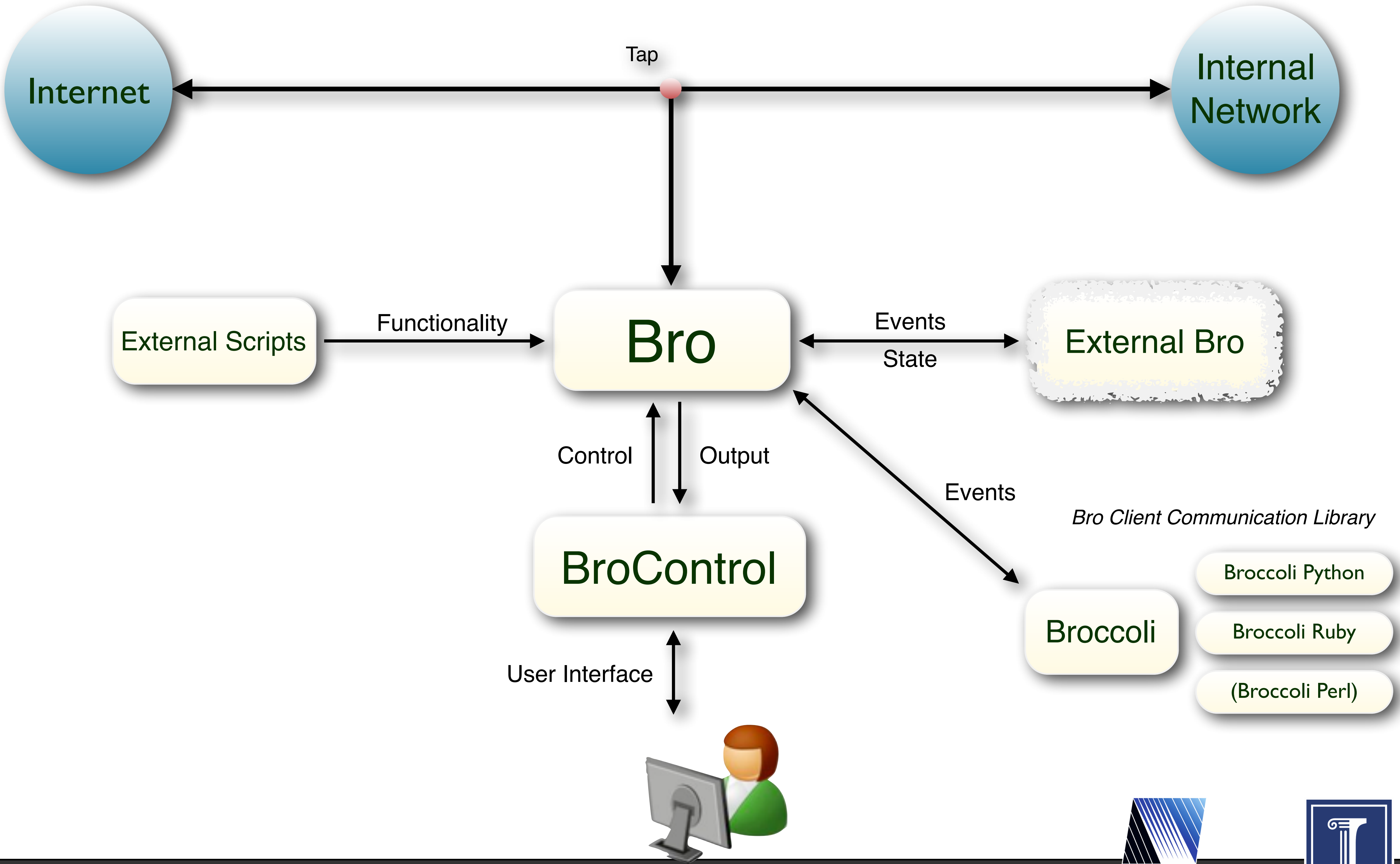
Bro Ecosystem



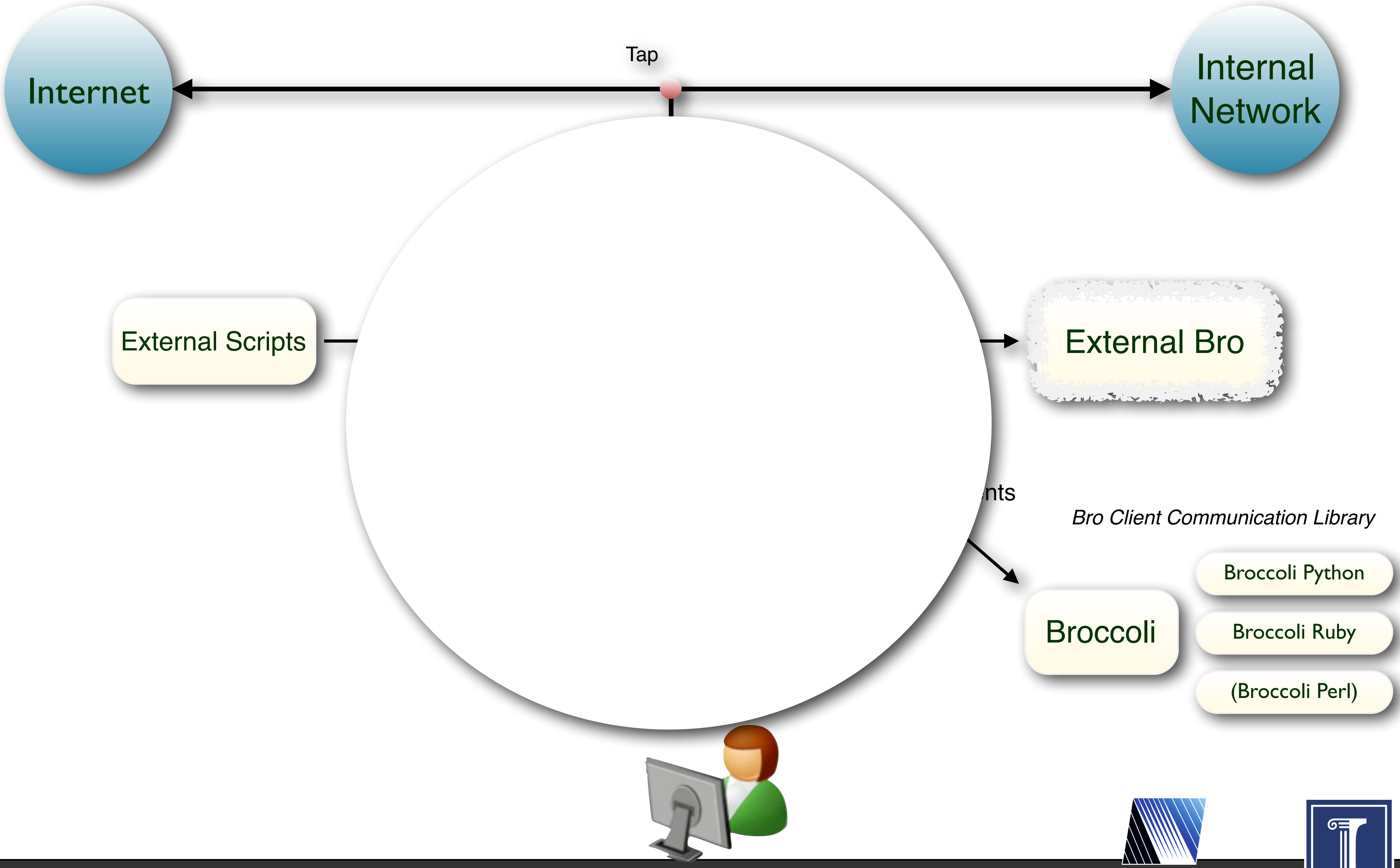
Bro Ecosystem



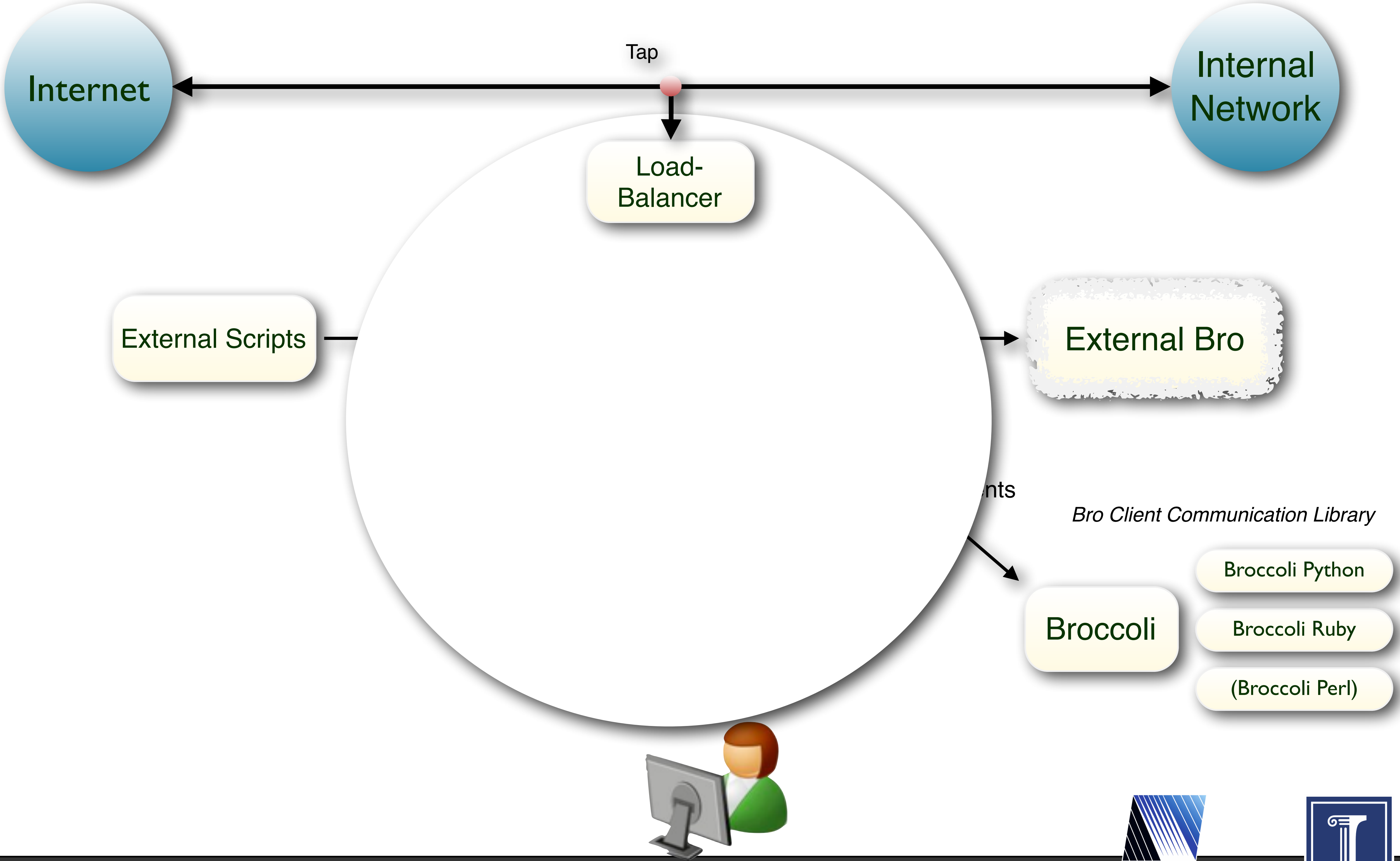
Bro *Cluster* Ecosystem



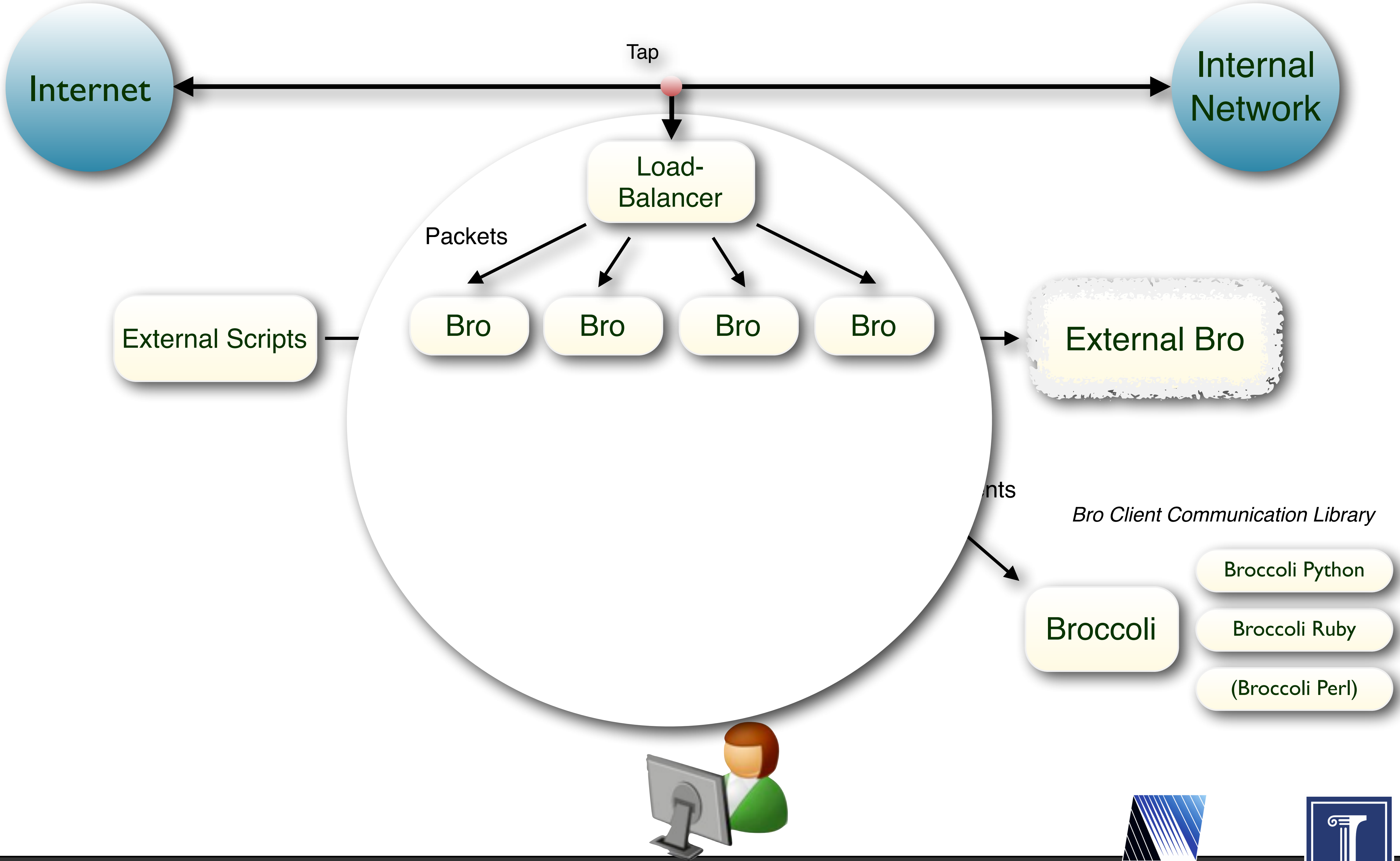
Bro *Cluster* Ecosystem



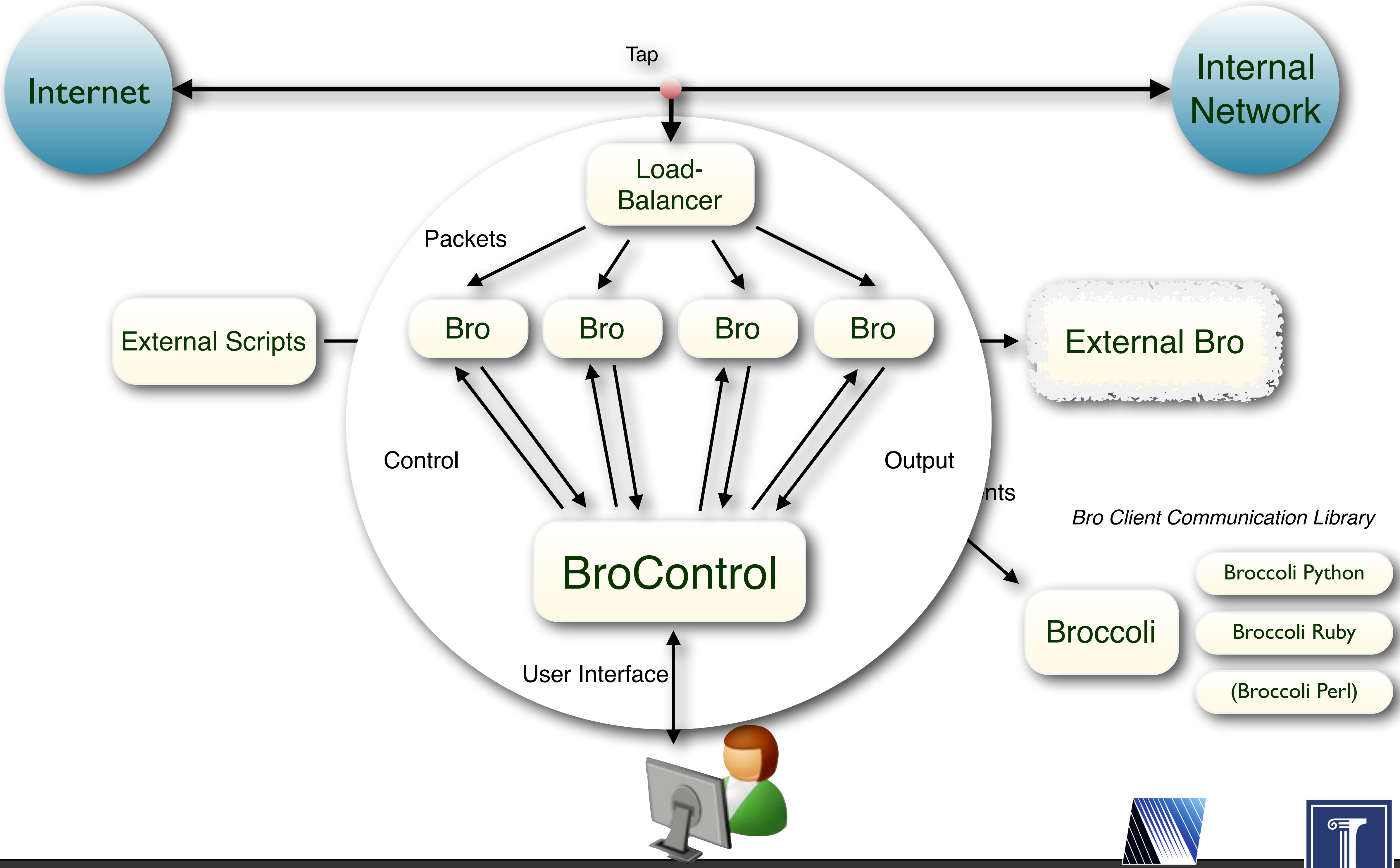
Bro *Cluster* Ecosystem



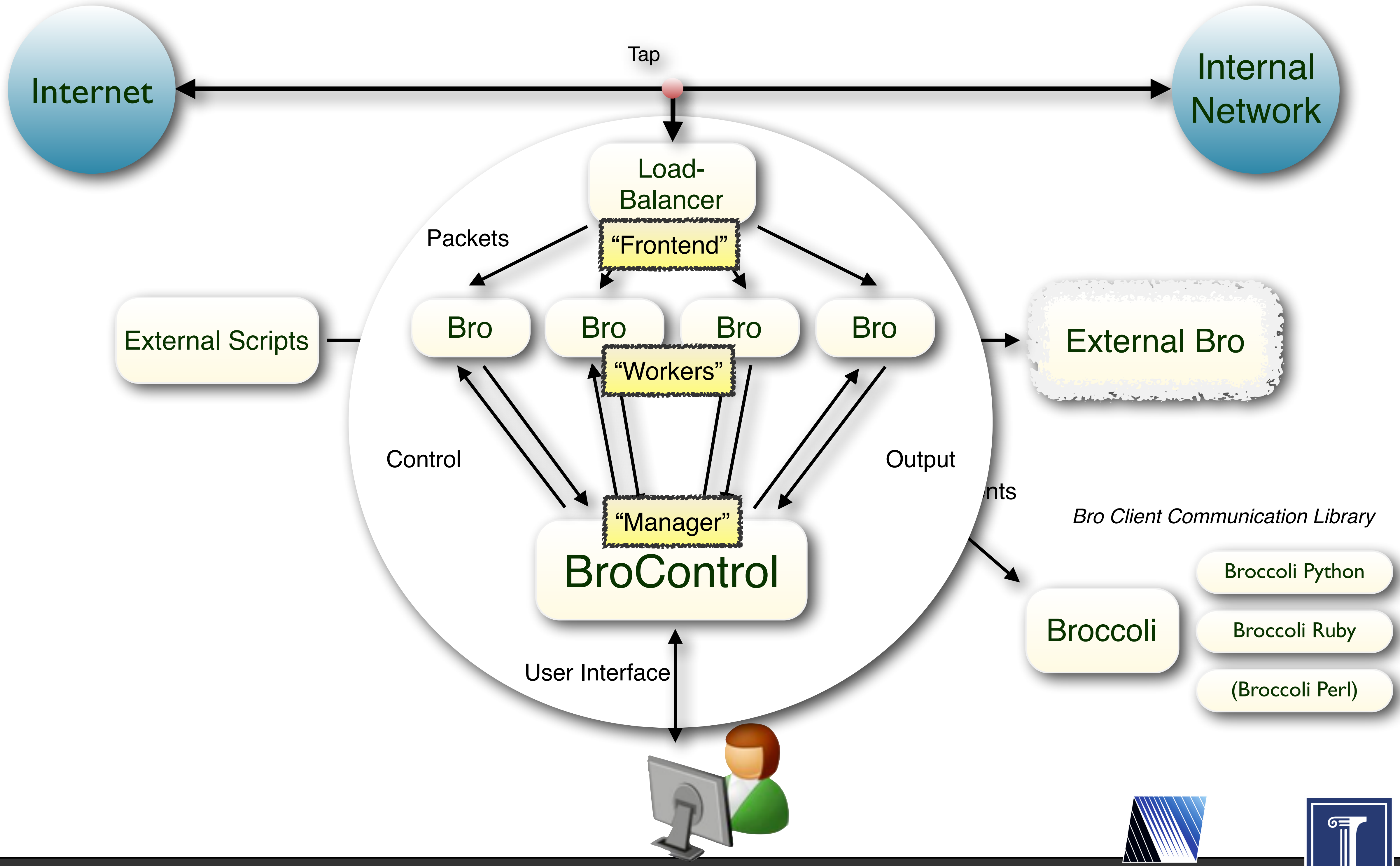
Bro *Cluster* Ecosystem



Bro *Cluster* Ecosystem



Bro *Cluster* Ecosystem



Installing Bro

Here: We'll use ISLET.

Comes with everything preinstalled.

Normally: Follow instructions on bro.org.

`http://www.bro.org/sphinx/install`

Building from source is pretty straight-forward:

```
> yum install cmake flex bison swig libpcap-devel […]

> wget http://www.bro.org/downloads/release/bro-2.2.tar.gz
> tar xzvf bro-2.2.tar.gz
> cd bro

> ./configure --prefix=/usr/local && make && make install
```

Configuring Bro

In many cases, just two files to edit.

`<prefix>/etc/node.cfg`

```
# If you have a small network and only one interface to monitor,
# this will do it. We’ll talk about cluster mode later.
[bro]
type=standalone
host=localhost
interface=eth0
```

`<prefix>/etc/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
```

(There’s also `<prefix>/etc/broctl.cfg` with more options you can tweak.)

Using BroControl

Use "broctl" to start & stop.

```
# broctl install
# broctl start
starting bro ...
# broctl status
Name        Type       Host       Status   Pid    Started
bro         standalone localhost  running  16737  15 May 15:57:35
# ls <prefix>/logs/current/
conn.log http.log […]
```

Reinstall after changing Bro's configuration.

```
# broctl check
bro is ok
# broctl install
# broctl restart
```

Using Bro from the Command Line

We'll use the Bro binary directly.

```
# bro -r trace.pcap
# ls *.log
conn.log http.log […]
```

"bro-cut" is a handy tool to work with logs.

```
# cat http.log | bro-cut -d ts id.orig_h host
2009-11-21T02:19:34-0800 192.168.1.105 download.windowsupdate.com
2009-11-21T02:19:37-0800 192.168.1.105 www.update.microsoft.com
[…]
```

Generally, use your standard Unix tools.

grep, awk, head/tail, sed, etc.

So much more …

Bro is … a Platform

Intrusion Detection	Vulnerabilit. Mgmt	File Analysis	Traffic Measure-ment	Traffic Control	Compliance Monitoring

There's much more we can talk about …

Host-level integration
Data import and export
Automatic Reaction
Monitoring Internal Networks
Measurements
SDN integration
Industrial Control Systems
Embedded Devices
Current Research

More File Analysis
More Protocols
More File Analysis
100Gb/s Networks
Enterprise Protocols
Summary Statistics
Science DMZs
ICSL SSL Notary
Cluster Deployment

Using ISLET & Try.Bro

- ISLET Server
  - Full Linux environment
  - ssh demo@54.149.11.154
  - Password is “CTSC”
    - Then create your own account
  - exercises are in /exercises
- Try.Bro
  - Point web browser to try.bro.org
  - Good for playing with language, seeing logs

The U.S. National Science Foundation has enabled much of our work.

Bro is coming out of almost two decades of academic research, along with extensive transition to practice efforts. NSF has supported much of that, and is currently funding a Bro Center of Expertise at the *International Computer Science Institute* and the *National Center for Supercomputing Applications*.

The Bro Project	**Commercial Support**
www.bro.org	www.broala.com
info@bro.org	info@broala.com
@Bro_IDS	@Broala_

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

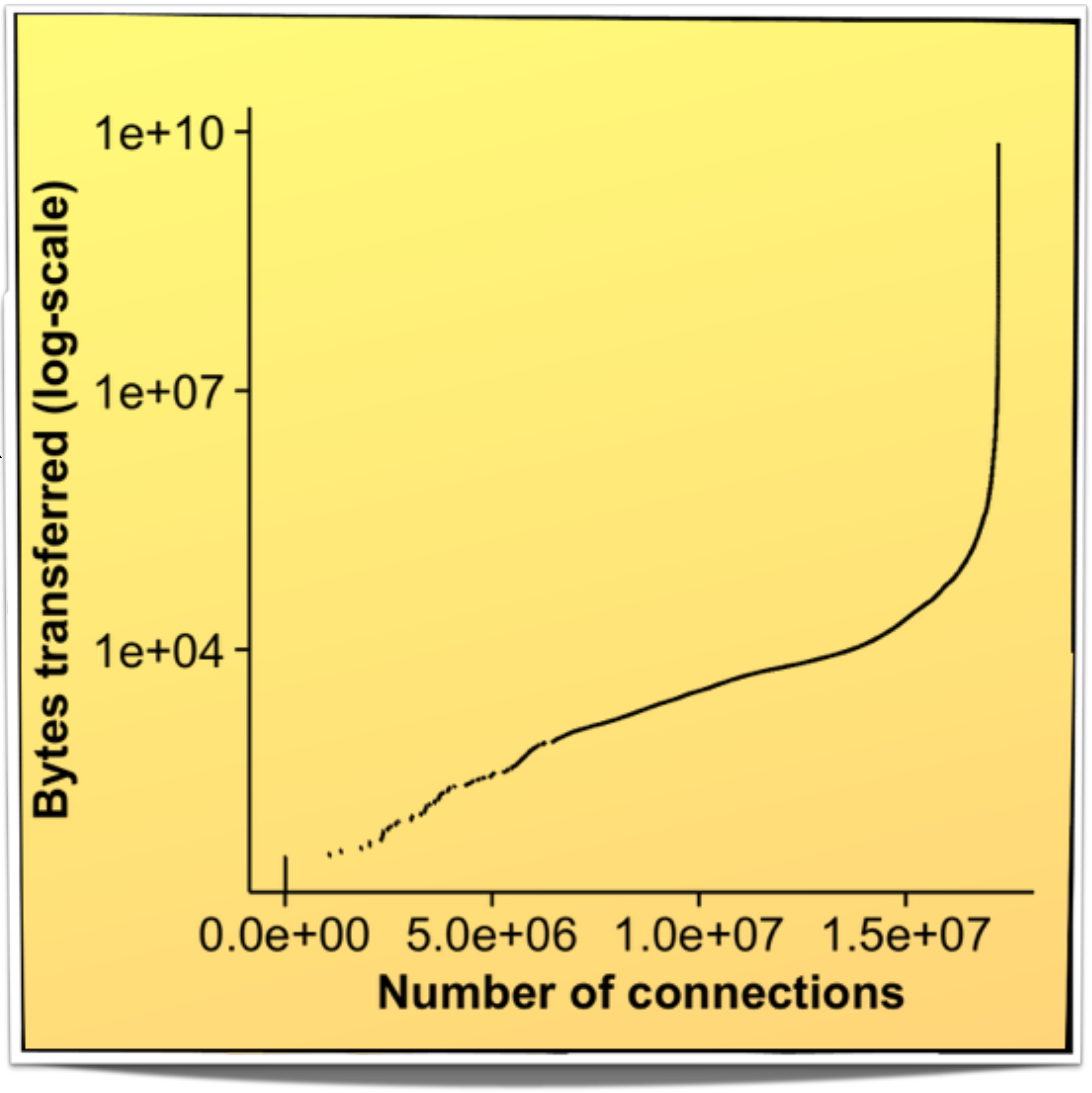
Uses for NetControl

Traffic Shunting

Block attacks at network

Redirecting high traffic fl

Quarantine hosts



Uses for NetControl

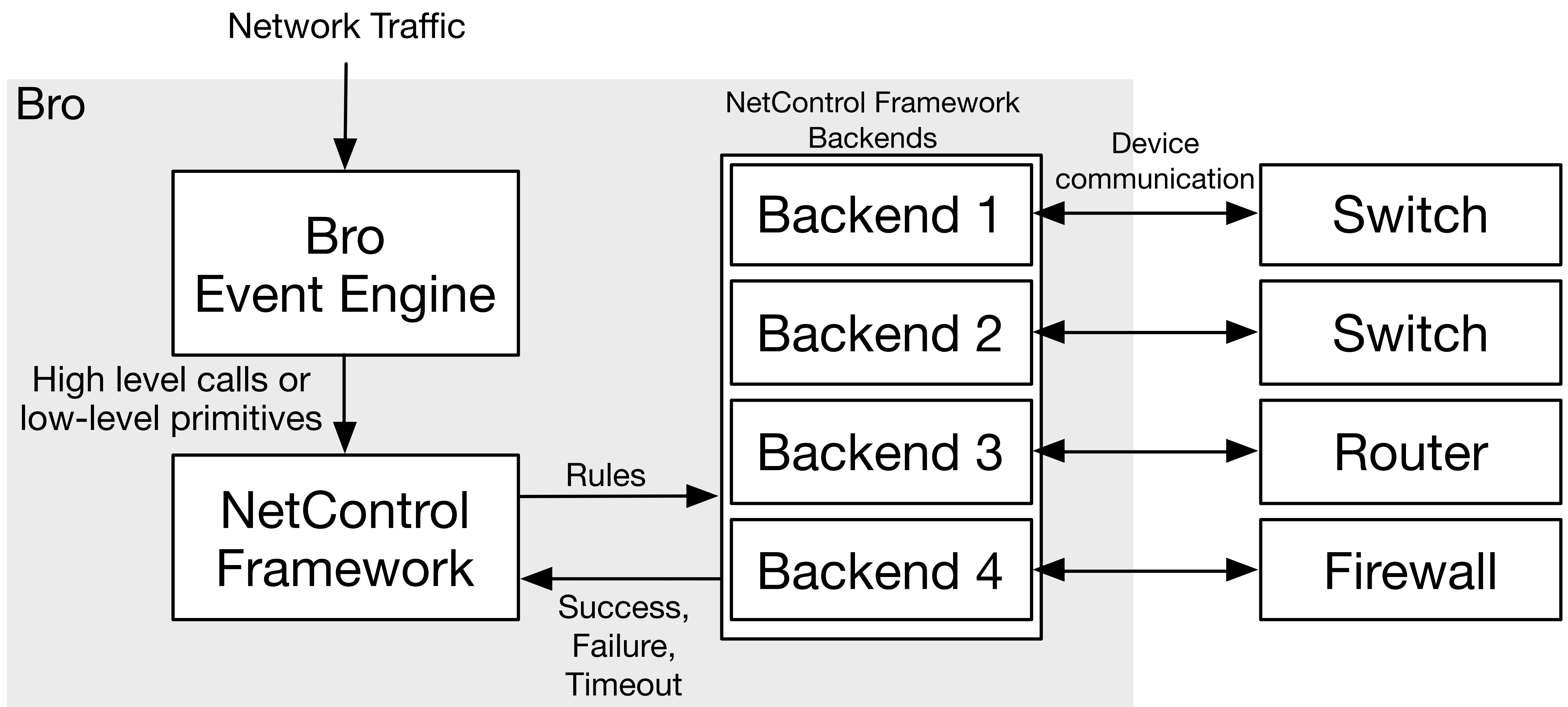
Traffic Shunting

Block attacks at network boundary

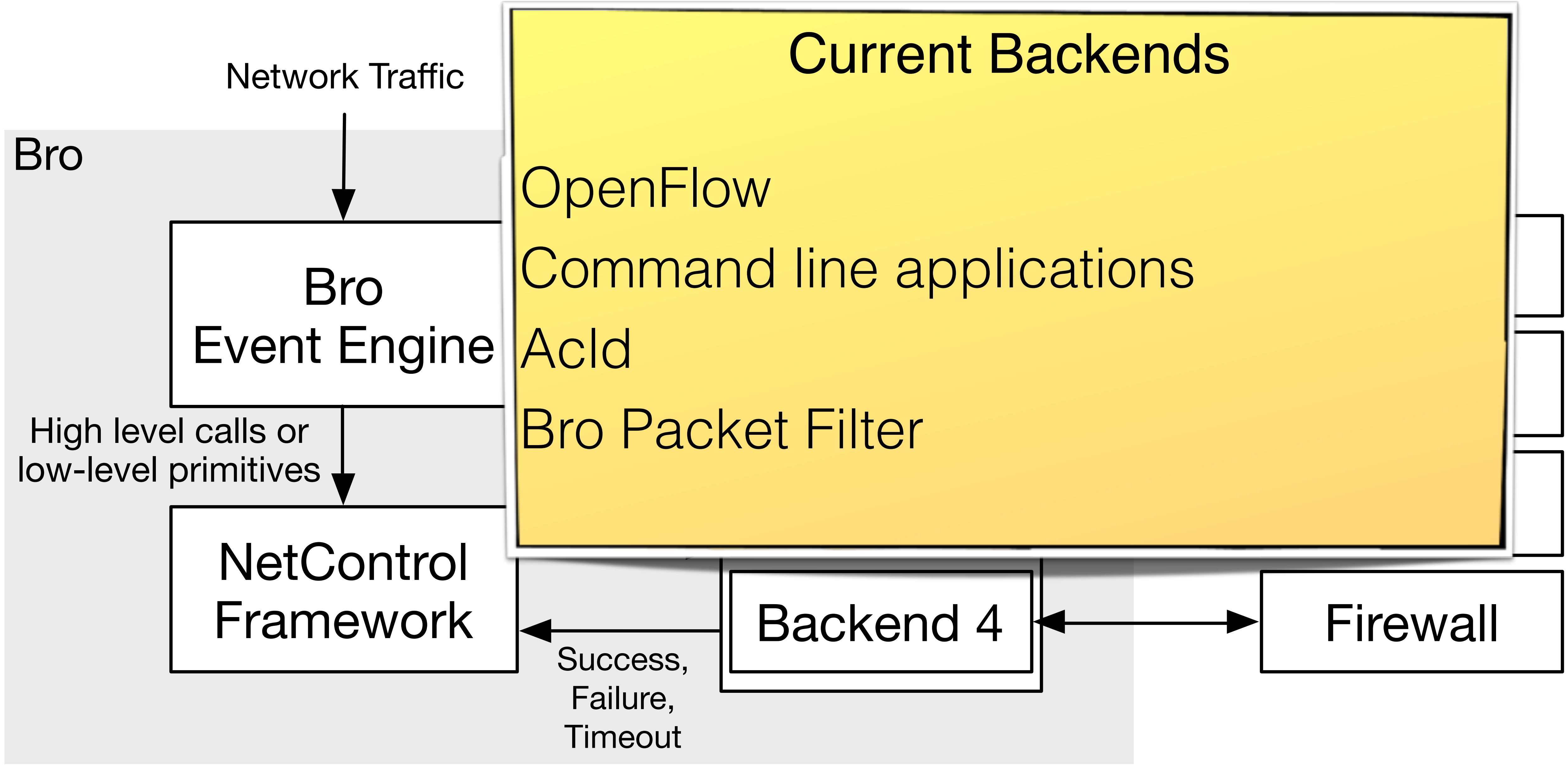
Redirecting high traffic flows to different interfaces

Quarantine hosts

Architecture



Architecture



Current Backends

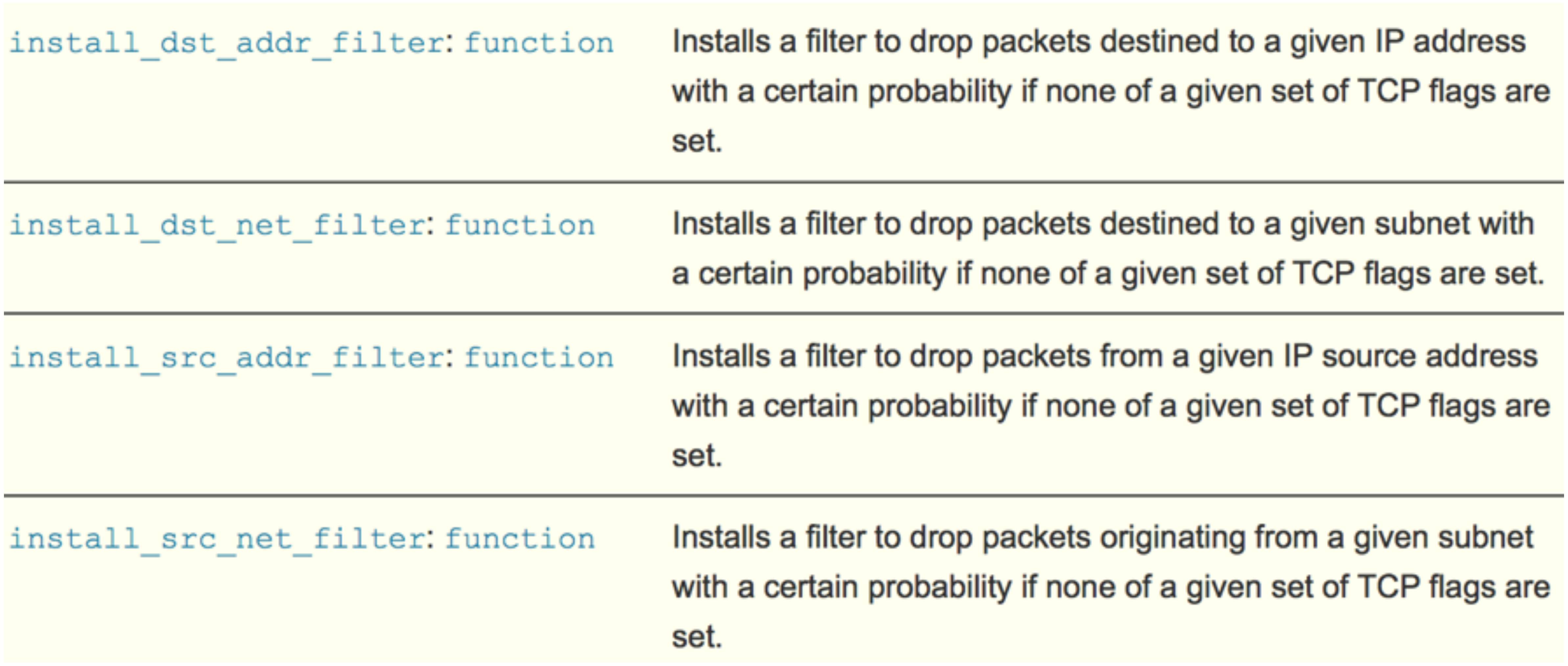
OpenFlow

Command line applications

Acld

Bro Packet Filter

Bro PacketFilter



`install_dst_addr_filter`: `function`	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.
`install_dst_net_filter`: `function`	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.
`install_src_addr_filter`: `function`	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.
`install_src_net_filter`: `function`	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.

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*)

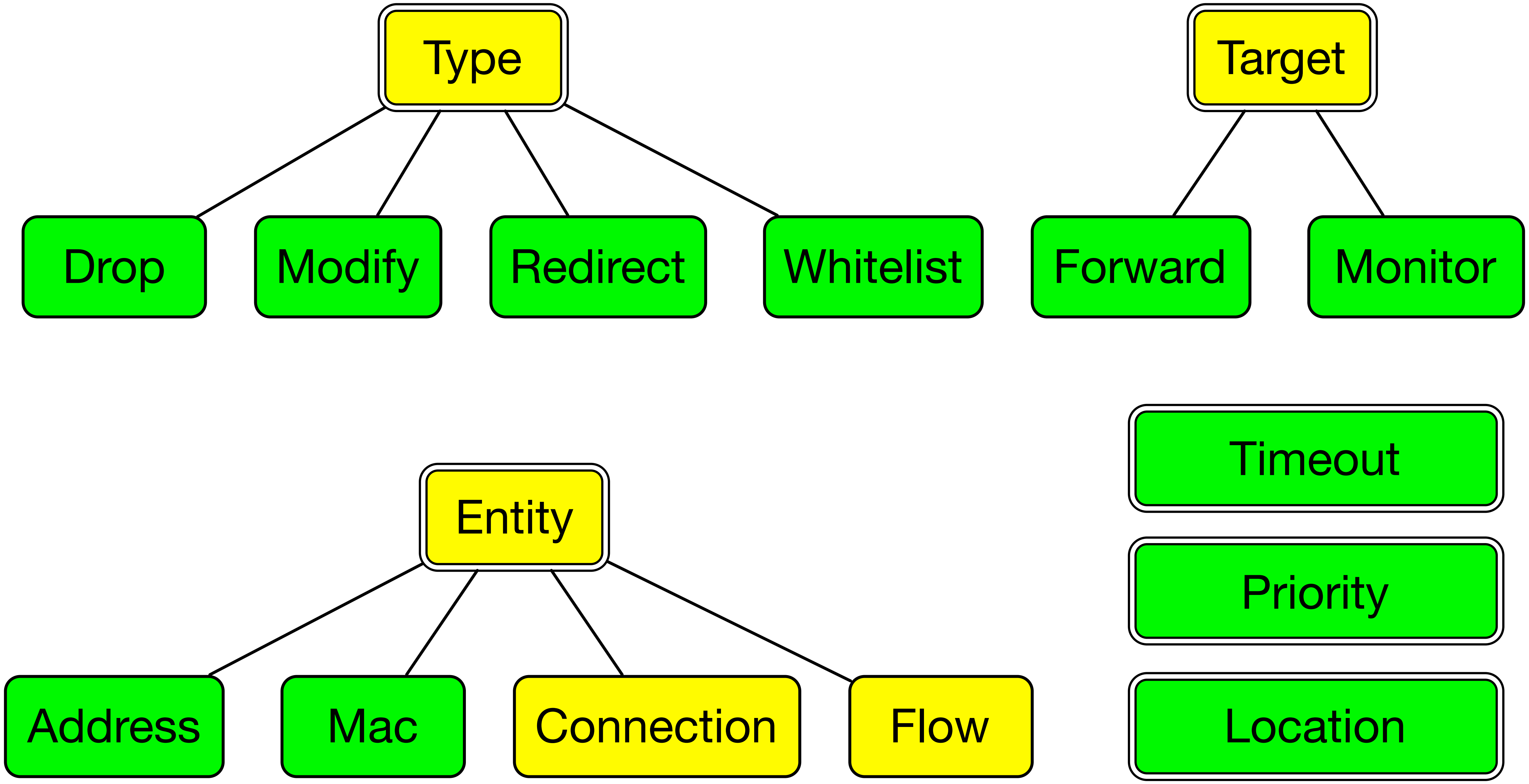
whitelist (*prefix, timeout*)

API Examples

```
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?

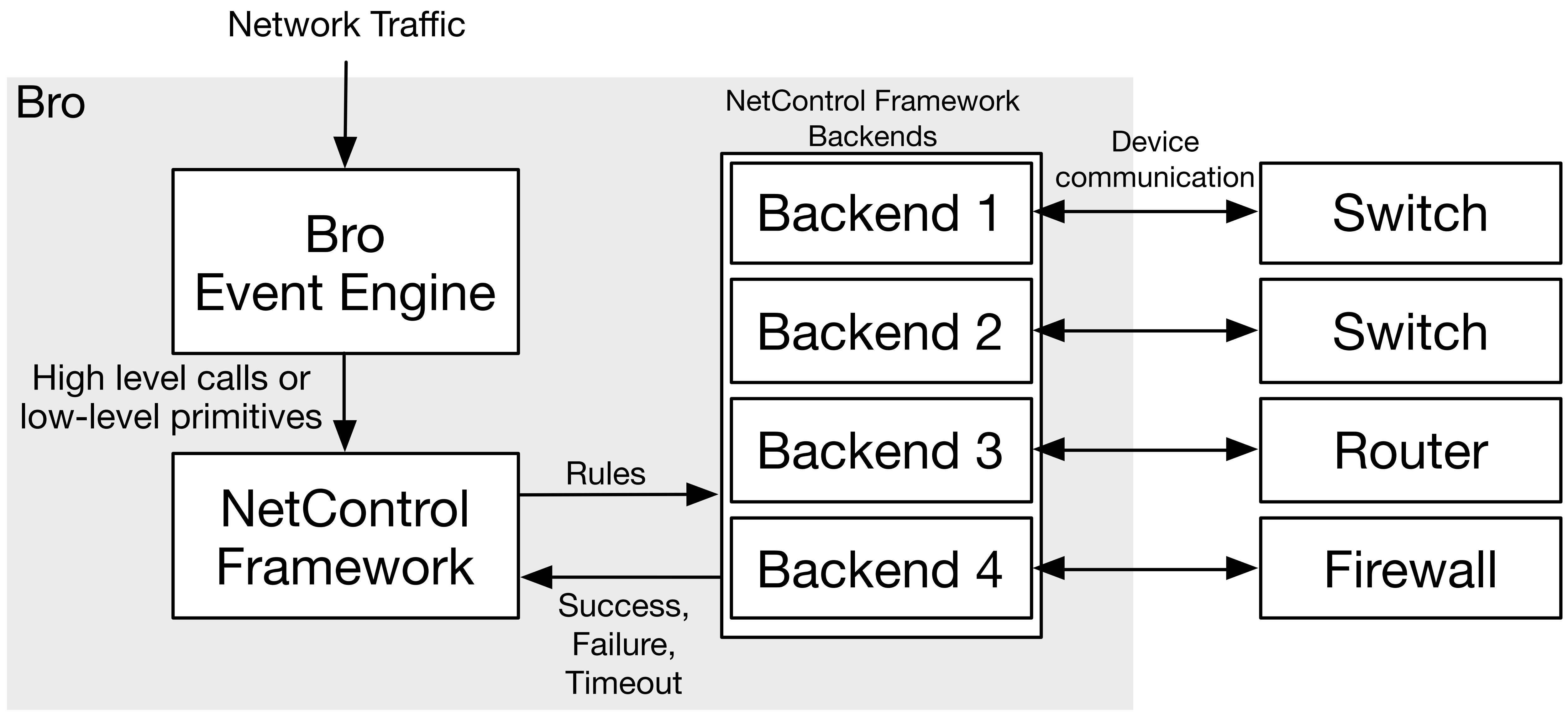


Example

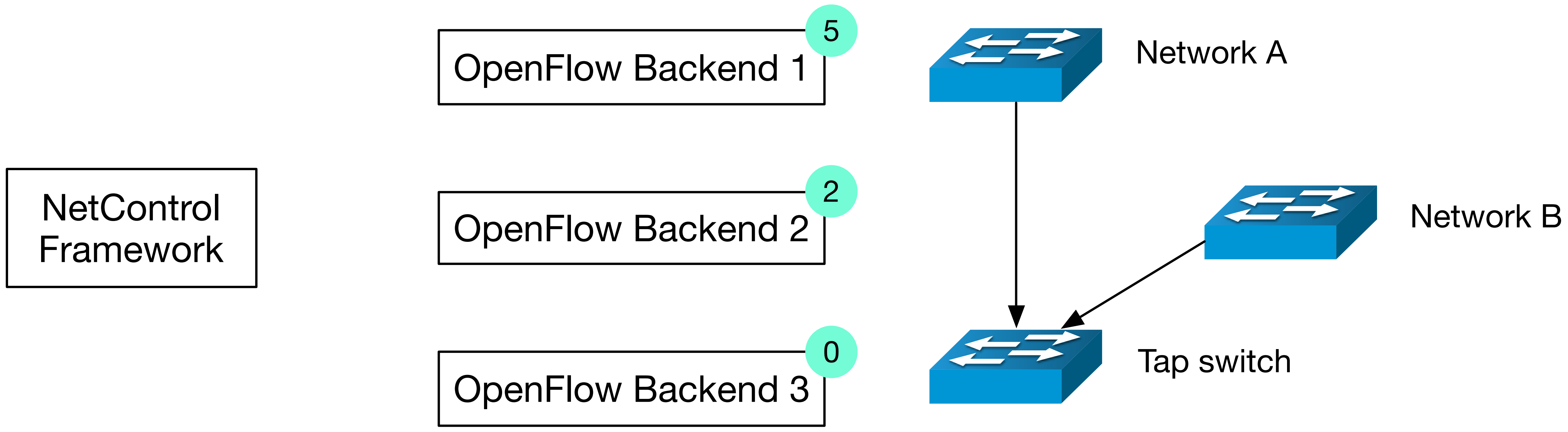
```
Rule(Type=Drop, Entity=Flow([5-tuple]), Target=Monitor)
```

```
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);
}
```

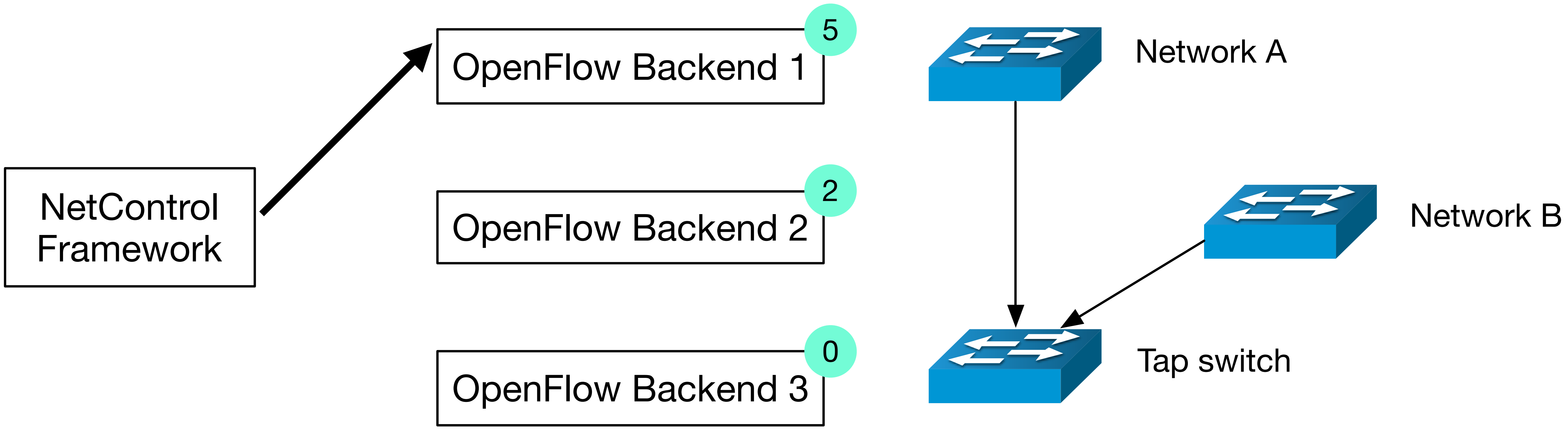
Choosing Backends



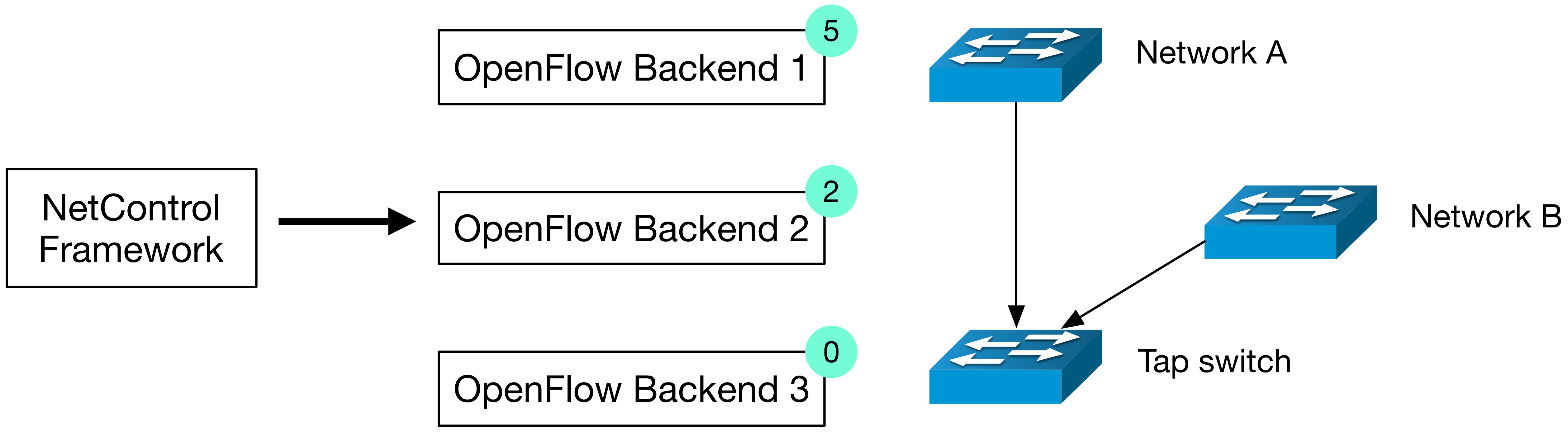
Choosing Backends



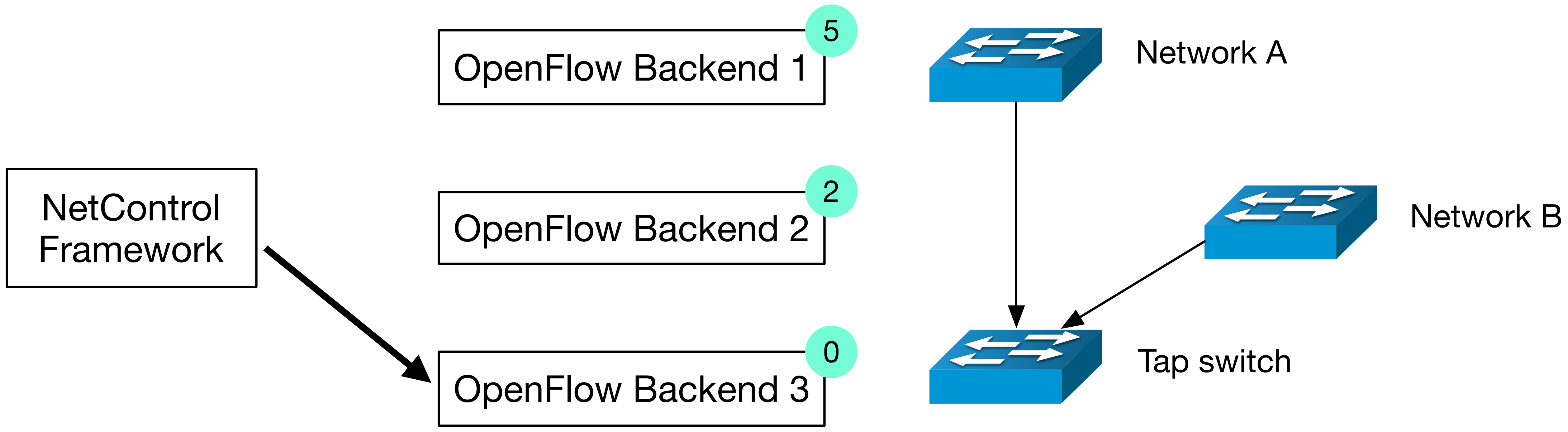
Choosing Backends



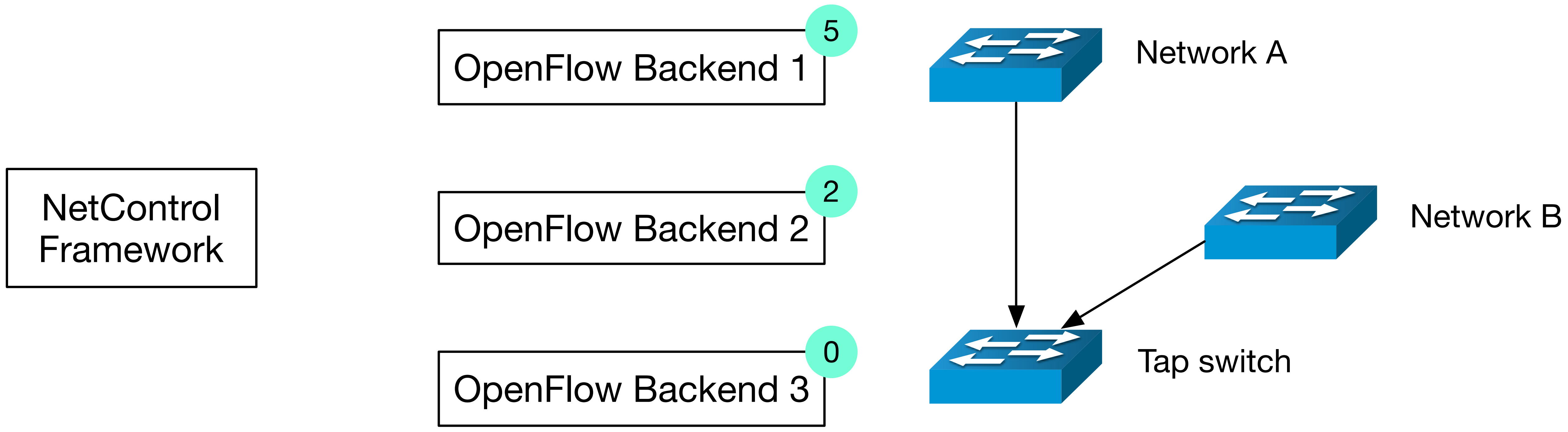
Choosing Backends



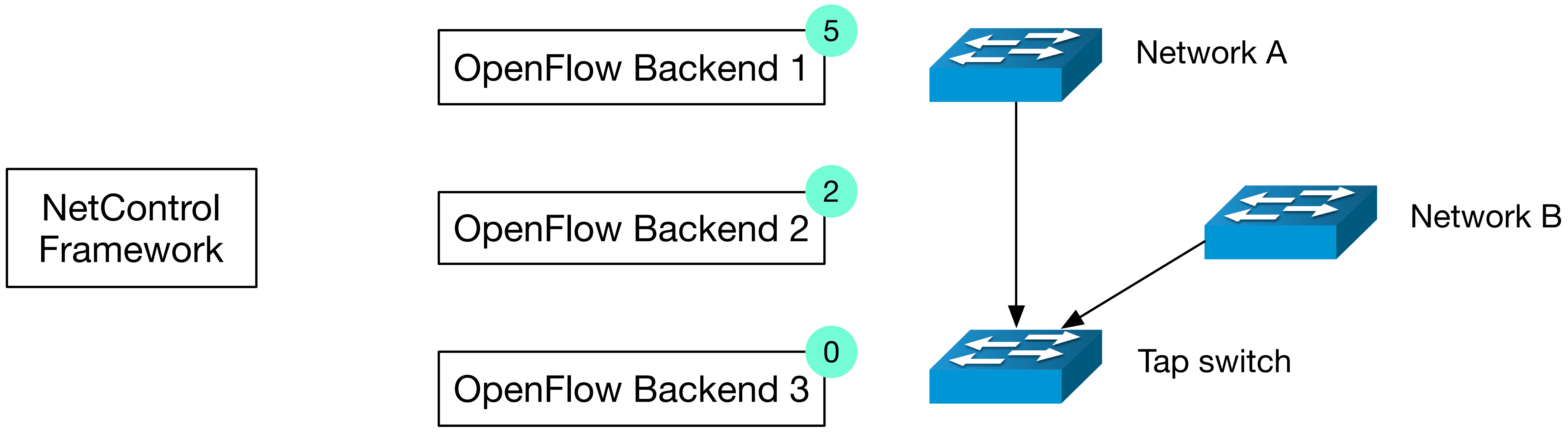
Choosing Backends



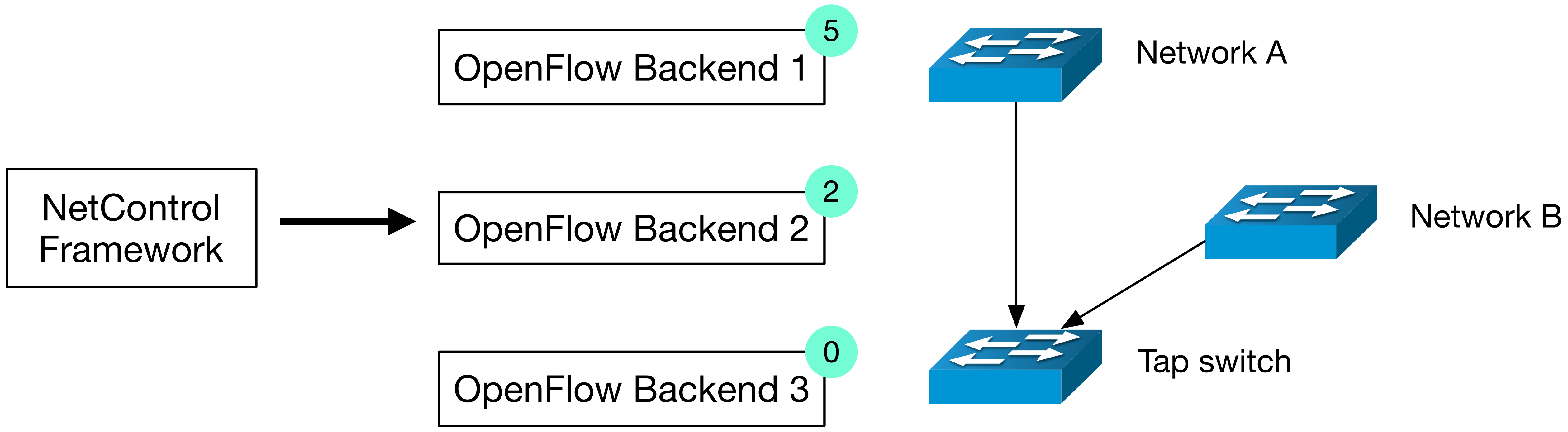
Choosing Backends



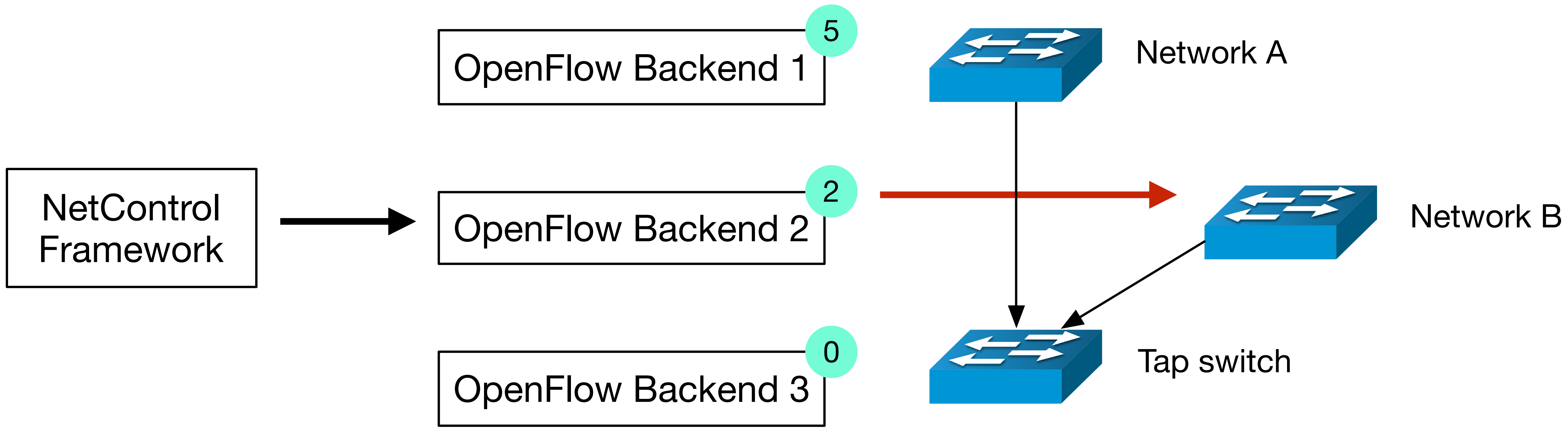
Choosing Backends



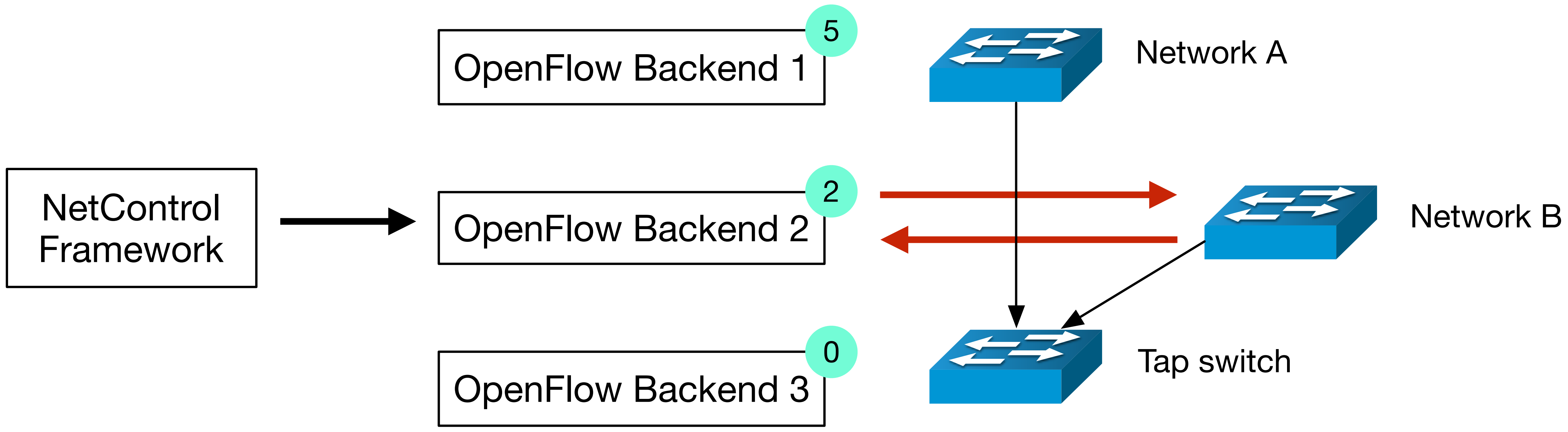
Choosing Backends



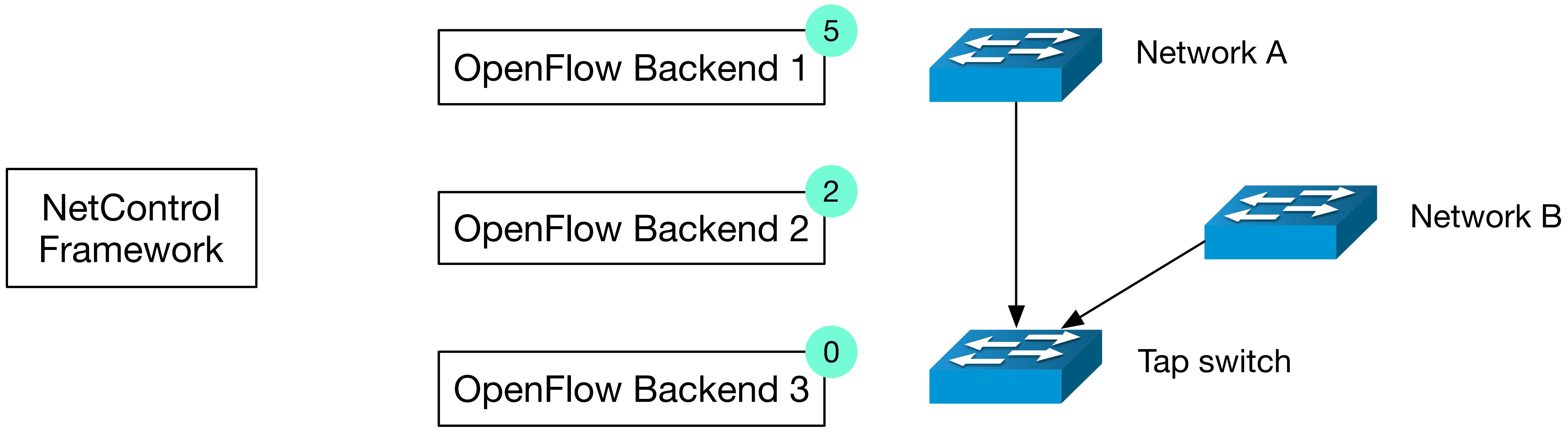
Choosing Backends



Choosing Backends



Choosing Backends



Adding Backends

```
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

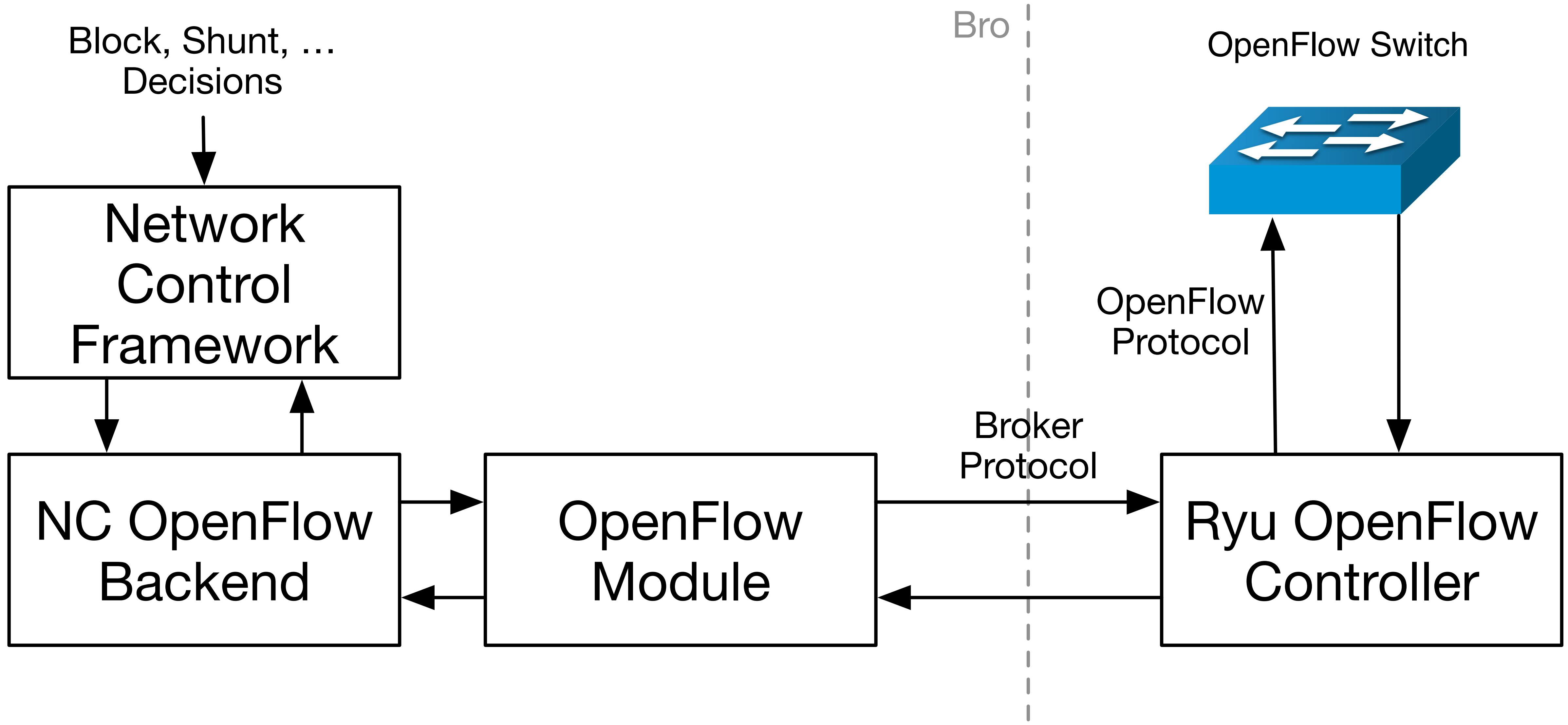
Open Specification

Allows Software to insert rules into switch flow tables

Match (and change) characteristics like

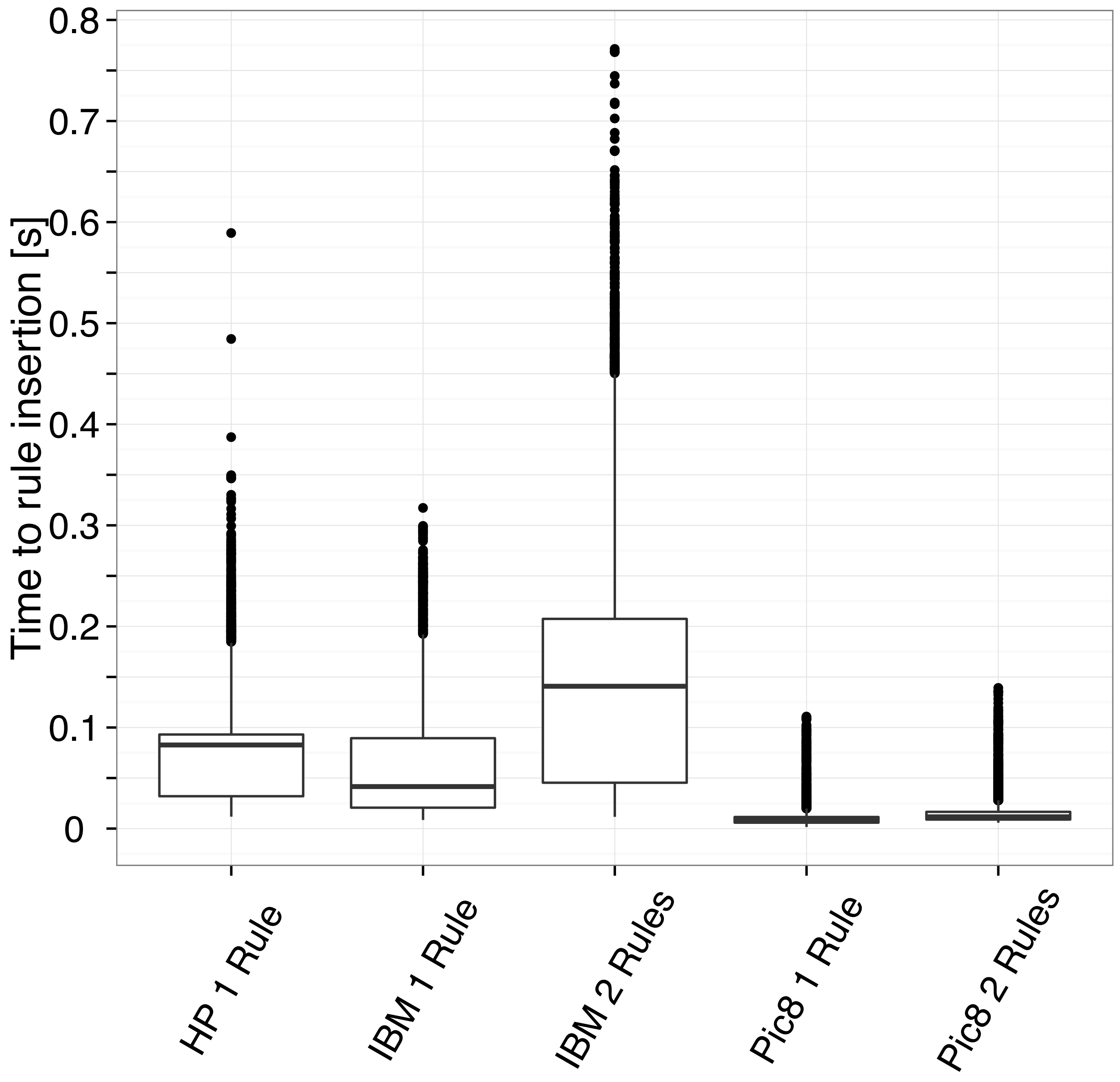
- IPv4/6 addresses, ports, etc.
- Vlans

NetControl & OpenFlow

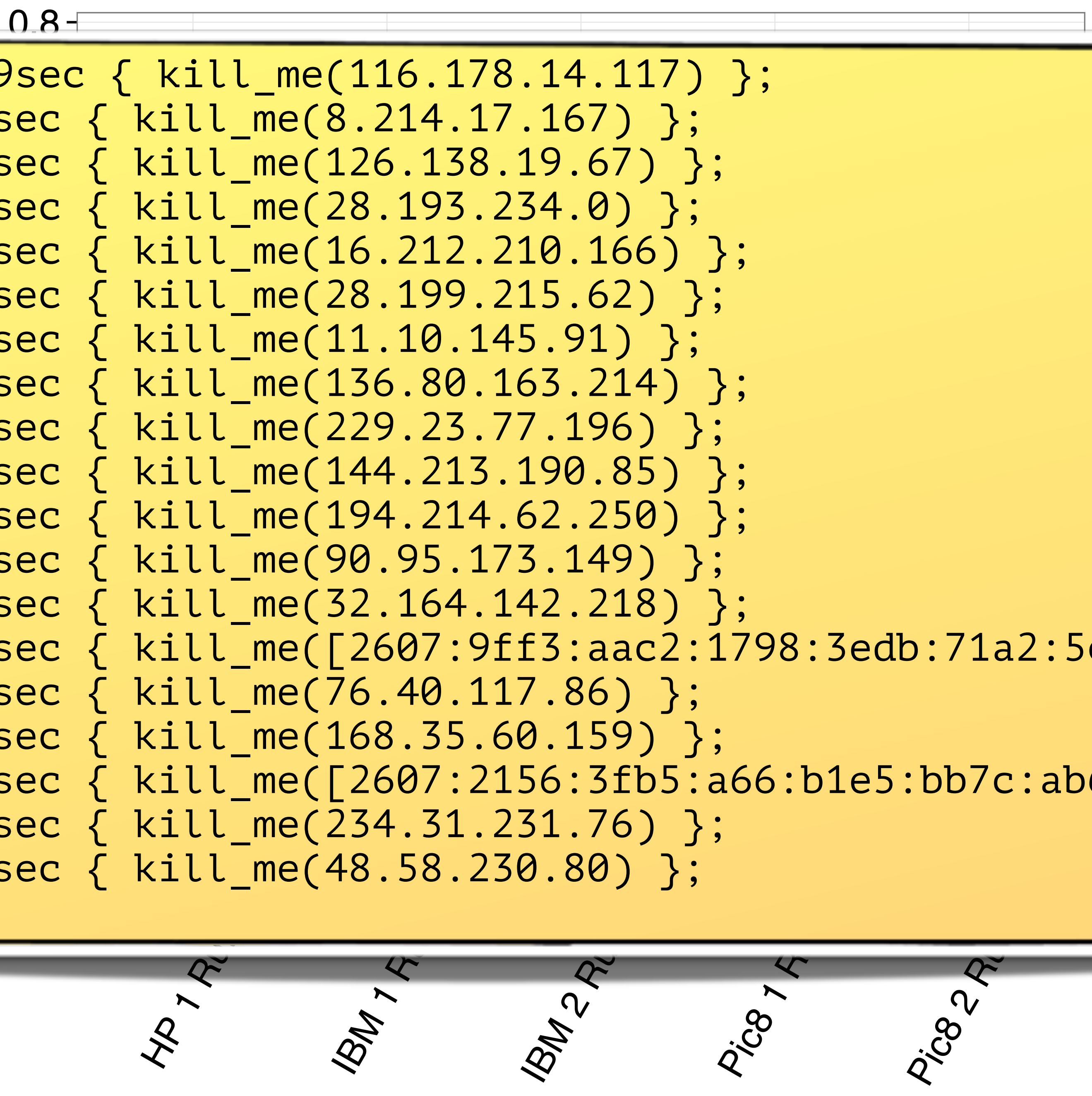


Demonstration

Rule Insertion Speed



Rule Insertion Speed

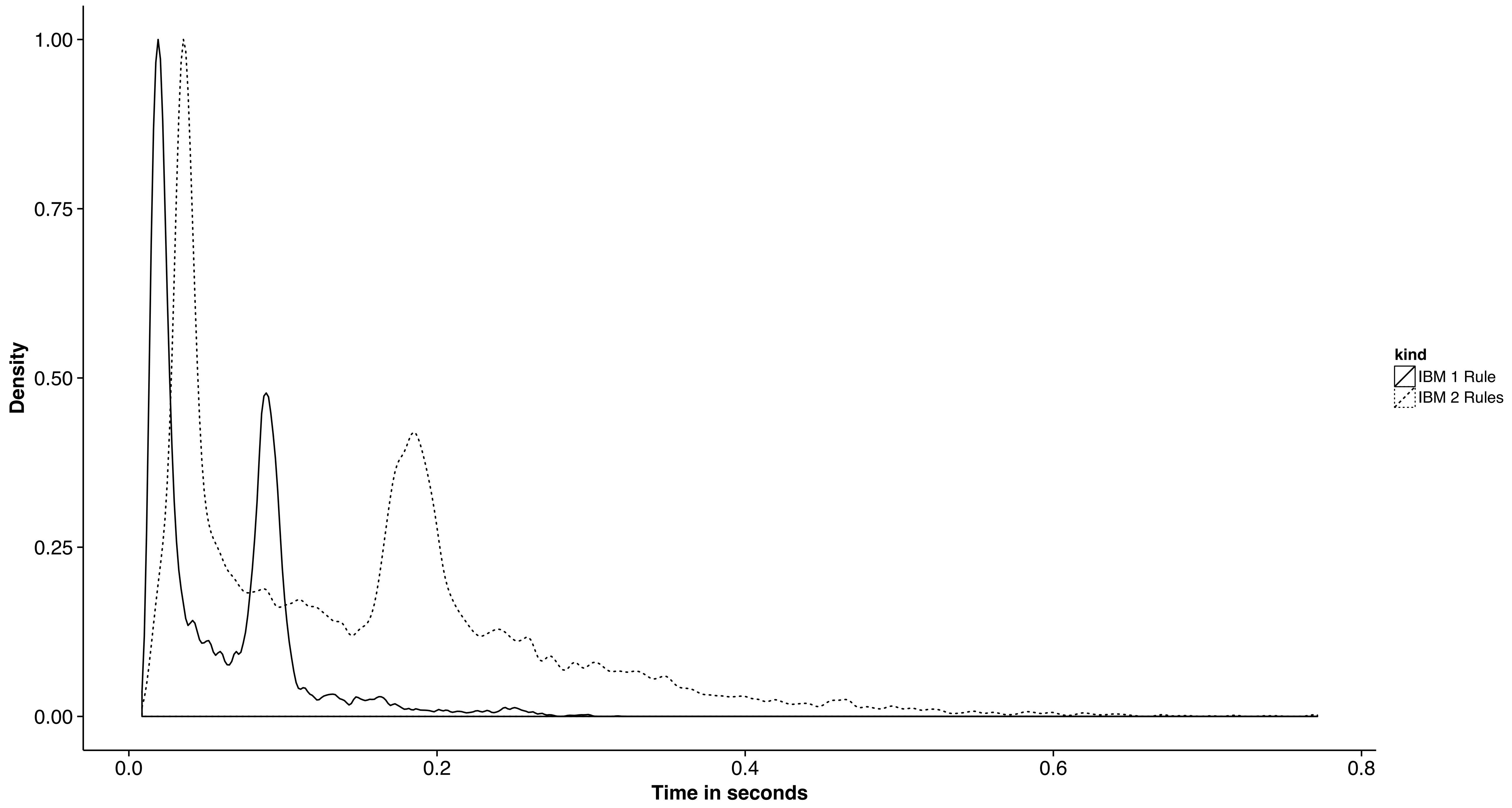


```
schedule 0.899309sec { kill_me(116.178.14.117) };
schedule 1.02567sec { kill_me(8.214.17.167) };
schedule 1.60747sec { kill_me(126.138.19.67) };
schedule 1.68983sec { kill_me(28.193.234.0) };
schedule 2.89801sec { kill_me(16.212.210.166) };
schedule 2.76121sec { kill_me(28.199.215.62) };
schedule 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

			Transferred Bytes		
Switch	Block time	Not blocked	Med.	Mean	Max
Pica8 (Median)	8.5ms	4,229 (2.7%)	0	1.6k	68k
Pica8 (75 Percentile)	11ms	8,273 (5.1%)	12	2.3k	101k
IBM (Median)	41ms	27,848 (17.4%)	194	9.5k	1.1MB
IBM (75 Percentile)	89ms	41,965 (26.3%)	526	27k	4.0MB
HP (Median)	82ms	38,381 (24%)	454	23k	4.5MB
HP (75 Percentile)	93ms	43,128 (27%)	537	28k	5.0MB

IBM G8052



Density
1.00
0.75
0.50
0.25
0.00
0.0
0.2
0.4
0.6
0.8
Time in seconds
kind
IBM 1 Rule
IBM 2 Rules

NetControl Summary

Control switches and other hardware

Easy syntax and rules

Extensible (API & Backends)

Fast

Get NetControl

github.com/bro/bro-netcontrol

What is a Bro log?

Justin Azoff

Aug 26, 2014

What is a Bro log?

A Bro log is a stream of high level entries that correspond to network events.

- A file downloaded via HTTP
- An email sent using SMTP
- A login over SSH

Not log, but logs.

Bro does not have a single “alert” type log. Instead each kind of event stream has a dedicated file with it’s own set of fields.

Why more than one file?

- The SMTP log has ‘from’ and ‘subject’ fields
- The HTTP log has ‘method’ and ‘uri’ fields
- The ‘from’ field would not make sense for HTTP, and ‘uri’ does not make sense for SMTP

How many log files are there?

By default, bro will output about two dozen log files, depending on what types of traffic it can see:

conn.log dhcp.log dns.log dpd.log files.log http.log intel.log known_certs.log known_hosts.log known_services.log modbus.log notice.log radius.log smtp.log snmp.log socks.log software.log ssh.log ssl.log syslog.log traceroute.log weird.log x509.log

Signal to noise ratio

The main way that log files can be categorized is by their size and signal to noise ratio. Some logs files are large and will contain entries that can be either benign or malicious. Other files are smaller and contain more actionable information.

- 24K known_services.log
- 28K software.log
- 68K notice.log
- 311M dns.log
- 856M conn.log

High signal log files

Inventory related log files

These log files are updated once per day and inventory your network

- known_hosts.log
- known_services.log
- known_certs.log
- software.log

Other high signal files

- notice.log - When bro detects something it thinks is exceptional it raises a notice.
- intel.log - Traffic that matches lists of known bad indicators is logged here.

Aside - Customizing log file contents.

Bro makes it easy to take a large log file and filter a subset of the entries to a smaller file with a higher signal to noise ratio.

Examples

- Filtering the http.log to http_exe.log
- Filtering the http.log to http_wget.log
- Filtering the http.log to http_java.log
- Filtering the conn.log to conn_cn.log
- Filtering the ssh.log to ssh_non_us.log

What exactly does a stream of events look like?

The short answer: A CSV file.

We can create some log files by starting Bro and running the unix command:

```
curl www.google.com
```

This will request the google home page, but not any of the associated javascript or image files.

Bro will write an entry in the http.log describing this event. The http.log contains 27 columns which can be a bit daunting. We can transpose the columns into rows to make this single line from http.log easier to understand

http.log transposed

Field	Type	Value
ts	time	1408828734.304076
uid	string	CZceY8wvnES5foJp4
id.orig_h	addr	192.168.43.222
id.orig_p	port	65032
id.resp_h	addr	74.125.226.50
id.resp_p	port	80
trans_depth	count	1
method	string	GET
host	string	www.google.com
uri	string	/
referrer	string	-

http.log transposed

Field	Type	Value
user_agent	string	curl/7.30.0
request_body_len	count	0
response_body_len	count	21232
status_code	count	200
status_msg	string	OK
info_code	count	-
info_msg	string	-
filename	string	-
tags	set[enum]	(empty)

http.log transposed

Field	Type	Value
username	string	-
password	string	-
proxied	set[string]	-
orig_fuids	vector[string]	-
orig_mime_types	vector[string]	-
resp_fuids	vector[string]	FvwPGj436gbcfXpCGf
resp_mime_types	vector[string]	text/html

Not just http.

This one HTTP download caused Bro to write entries to 6 log files:

- http.log has the above entry
- dns.log has an entry from the dns query for www.google.com
- files.log has an entry from the html file that was downloaded
- conn.log has an entry for both the dns an http connections
- known_hosts.log has an entry for 192.168.43.222
- software.log has an entry for an HTTP::BROWSER of curl/7.30.0 seen on 192.168.43.222

known_hosts.log transposed

Field	Type	Value
ts	time	1408828734.303825
host	addr	192.168.43.222

192.168.43.222 was seen for the first time at 1408828734.303825

software.log transposed (slightly edited)

Field	Type	Value
ts	time	1408828734.304076
host	addr	192.168.43.222
software_type	enum	HTTP::BROWSER
name	string	curl
version.major	count	7
version.minor	count	30
version.minor2	count	0
unparsed_version	string	curl/7.30.0

curl/7.30.0 was seen for the first time on 192.168.43.222 at 1408828734.304076