The Bro Network Intrusion Detection System

Robin Sommer
Lawrence Berkeley National Laboratory

rsommer@lbl.gov
http://www.icir.org

UC Computing Services Conference 2007

Outline

- Design of the Bro NIDS
  - Philosophy
  - Architecture
- LBNL's Bro Installation
- Recent Developments
  - Dynamic Protocol Detection
  - The Bro Cluster
- Outlook

Design of the Bro NIDS

System Philosophy

- Bro provides a real-time network analysis framework
  - Primary a network intrusion detection system (NIDS)
  - However it is also used for pure traffic analysis
- Focus is on
  - Application-level semantic analysis (rather than analyzing individual packets)
  - Tracking information over time
- Strong separation of mechanism and policy
  - The core of the system is policy-neutral (no notion "good" or "bad")
  - User provides local site policy

System Philosophy (2)

- Operators *program* their policy
  - Not really meaningful to talk about what Bro detects "by default"
- Analysis model is *not* signature matching
  - Bro is fundamentally different from, e.g., Snort (though it *can* do signatures as well)
- Analysis model is *not* anomaly detection
  - Though it does support such approaches (and others) in principle
- System thoroughly logs all activity
  - It does not just alert
  - Logs are invaluable for forensics

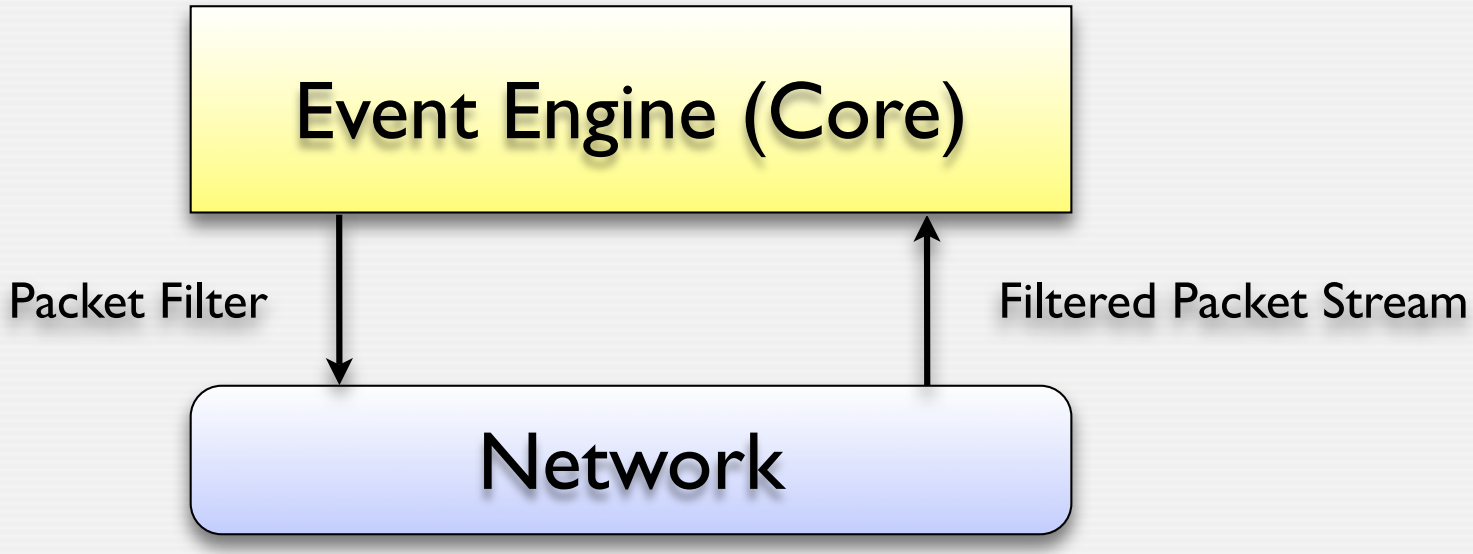
Target Environments

- Bro is specifically well-suited for scientific environments
  - Extremely useful in networks with liberal ("default allow") policies
  - High-performance on commodity hardware
  - Supports intrusion prevention schemes
  - Open-source (BSD license)
  - Developed at LBNL & the International Computer Science Institute
- It does however require some effort to use effectively
  - Pretty complex, script-based system
  - Requires understanding of the network
  - No GUI, just ASCII logs
  - Only partially documented
  - Lacking resources to fully polish the system
- Development is primarily driven by *research*
  - However, our focus is operational use; we invest much time into "practical" issues
  - Want to bridge gap between research and operational deployment

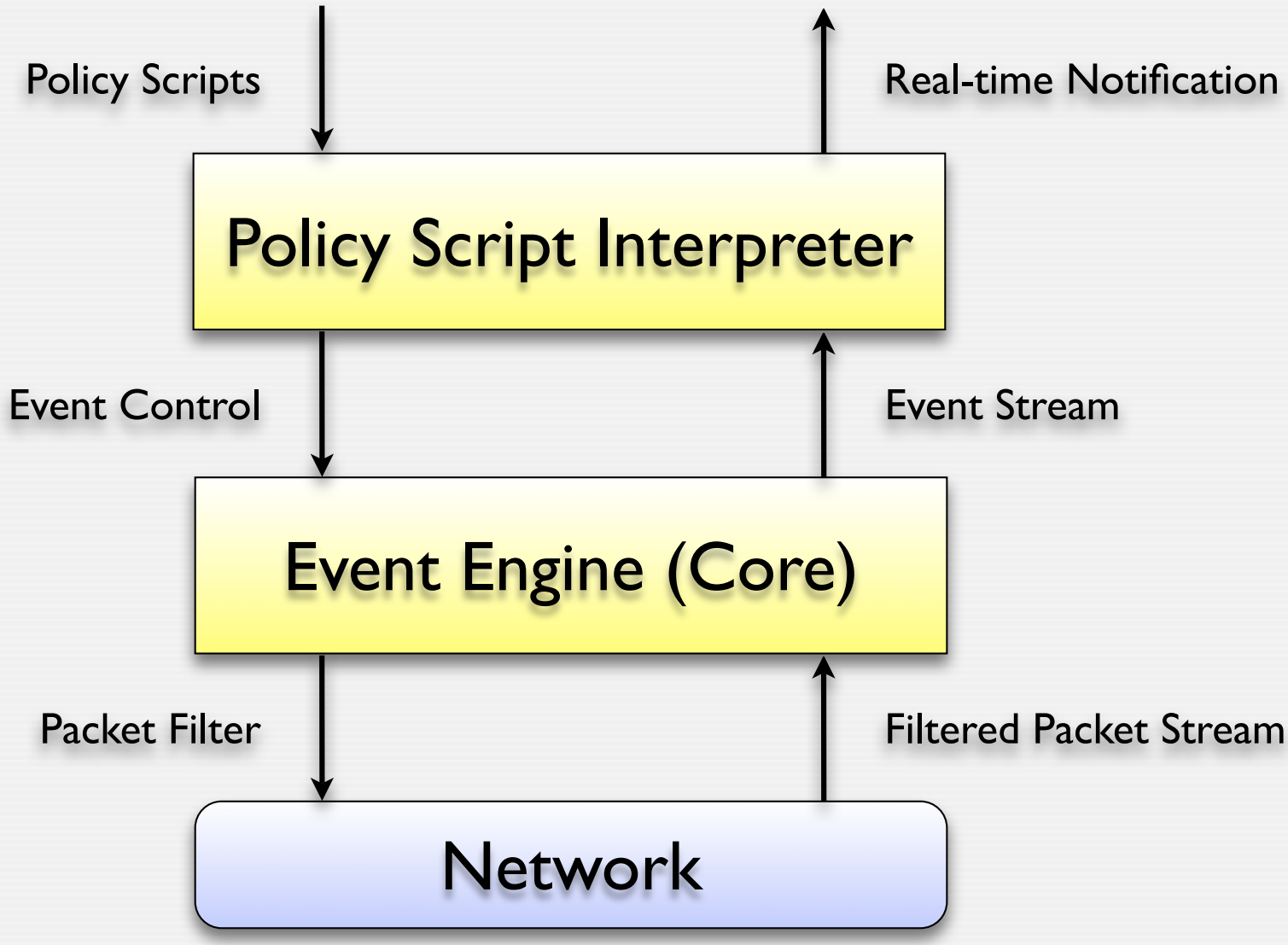
Architecture

Network

Architecture



Architecture



Event-Engine

- Event-engine is written in C++
- Performs *policy-neutral* analysis
  - Turns low-level activity into high-level events
  - Examples: `connection_established, http_request`
  - Events are annoated with context (e.g., IP addresses, URL)
- Contains *analyzers* for >30 protocols, including
  - ARP, IP, ICMP, TCP, UDP
  - DCE-RPC, DNS, FTP, Finger, Gnutella, HTTP, IRC, Ident, NCP, NFS, NTP, NetBIOS, POP3, Portmapper, RPC, Rsh, Rlogin, SMB, SMTP, SSH, SSL, SunRPC, Telnet
- Analyzers generate ~300 events ...

Policy Scripts

- Scripts process event stream, incorporating ...
  - ... context from past events
  - ... site's local security policy
- Scripts take actions
  - Recording activity to disk
  - Generating alerts via syslog or mail
  - Executing program as a form of response

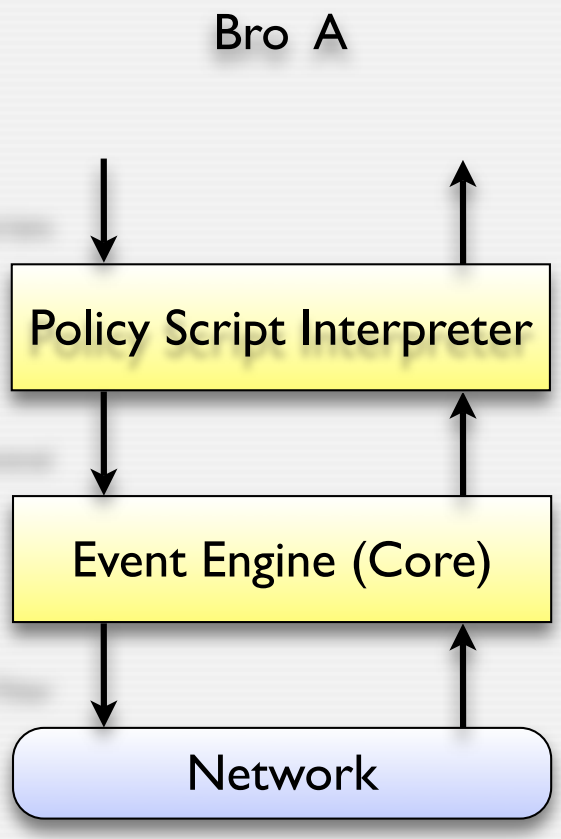
Script Example: Tracking SSH Hosts

```
global ssh_hosts: set[addr];

event connection_established(c: connection)
    {
    local responder = c$id$resp_h; # Responder's address
    local service = c$id$resp_p;   # Responder's port

    if ( service != 22/tcp )
        return; # Not SSH.

    if ( responder in ssh_hosts )
        return; # We already know this one.

    add ssh_hosts[responder]; # Found a new host.
    print "New SSH host found", responder;
    }
```

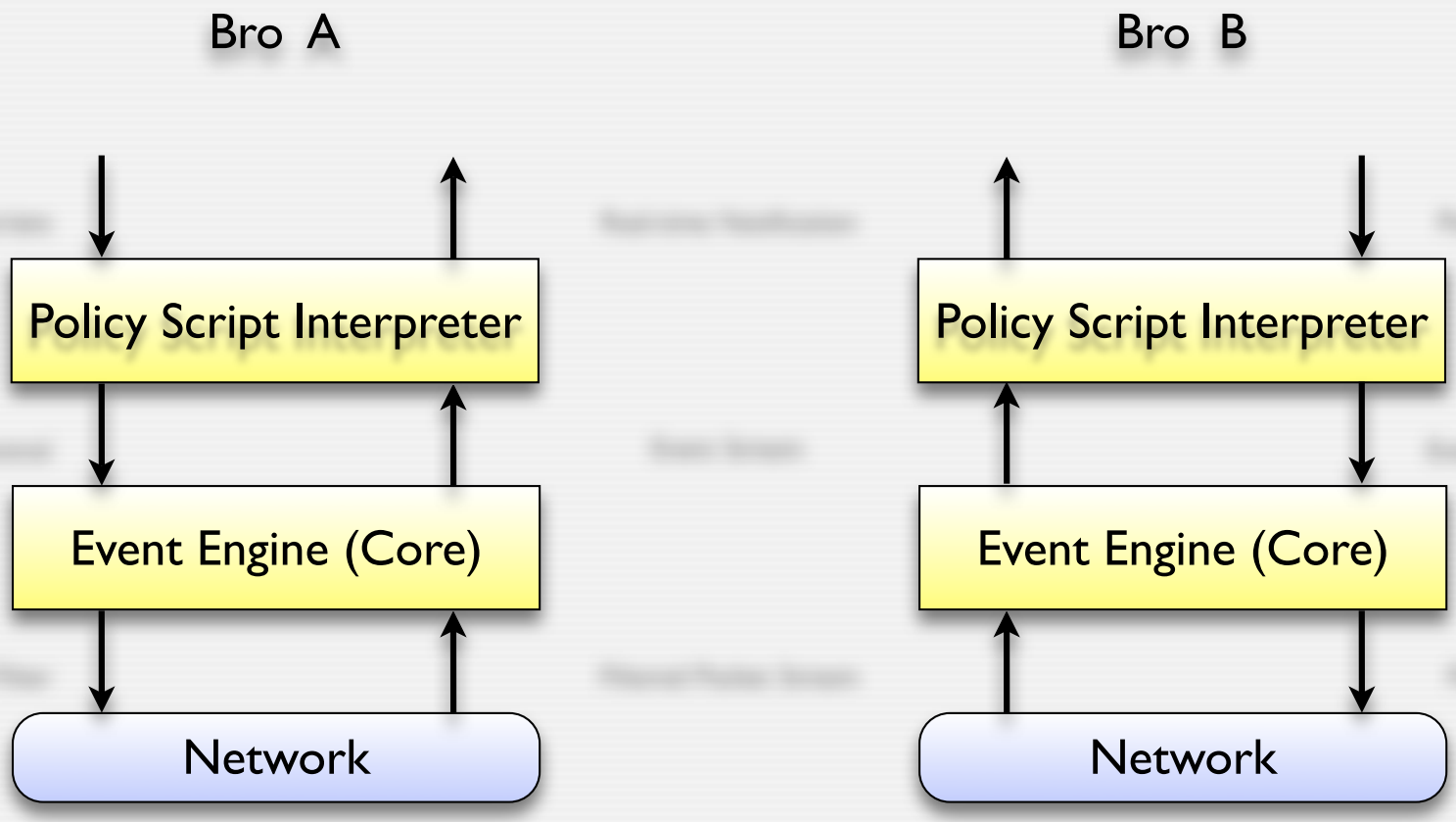
Expressing Policy

- Scripts are written in custom, domain-specific language
  - Bro ships with 20K+ lines of script code
  - Default scripts detect attacks & log activity extensively
- Language is
  - Procedural
  - Event-based
  - Strongly typed
  - Rich in types
    - Usual script-language types, such as tables and sets
    - Domain-specific types, such as addresses, ports, subnets
  - Supporting state management (expiration, timers, etc.)
  - Supporting communication with other Bro instances

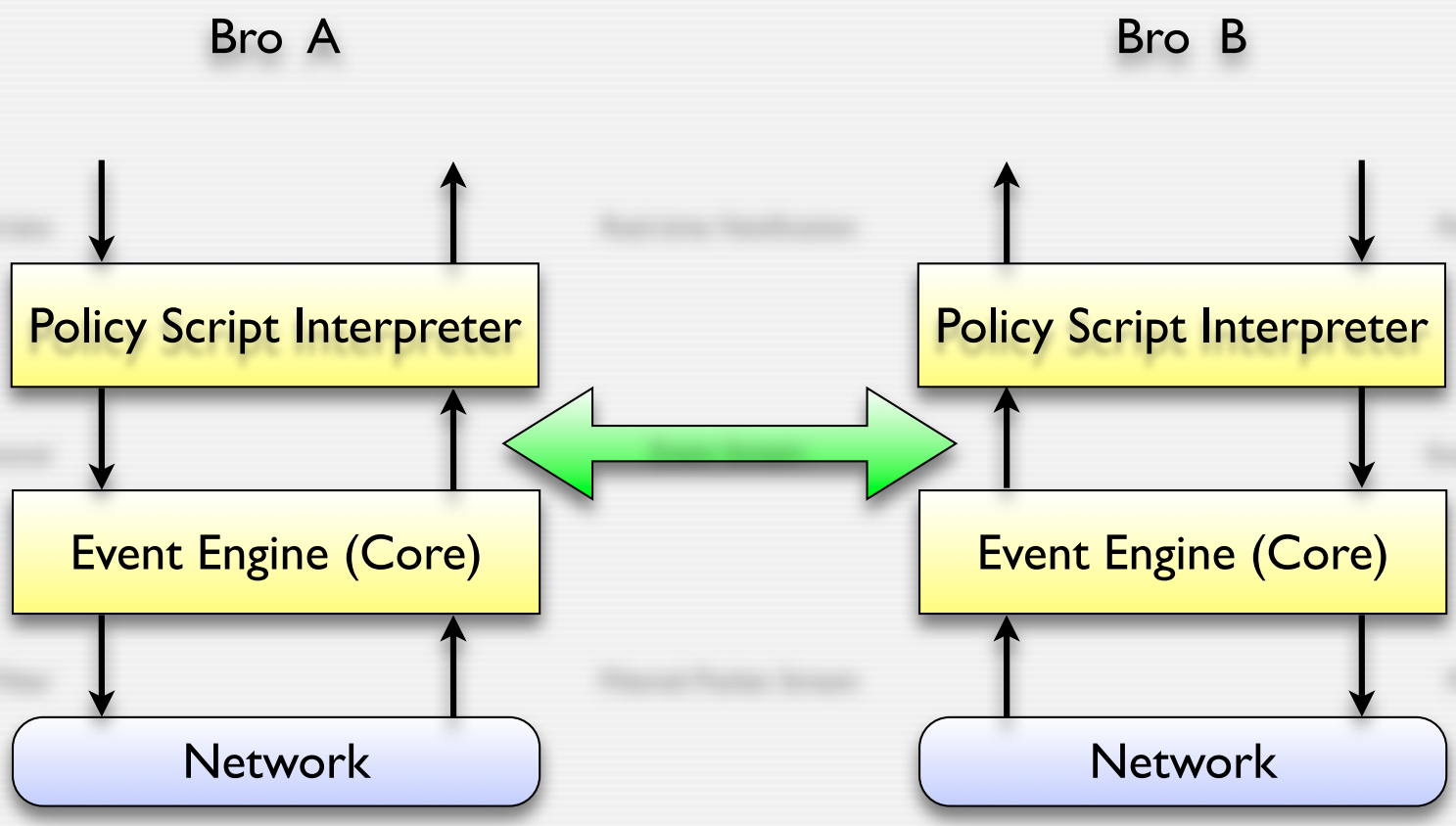
Communication Architecture



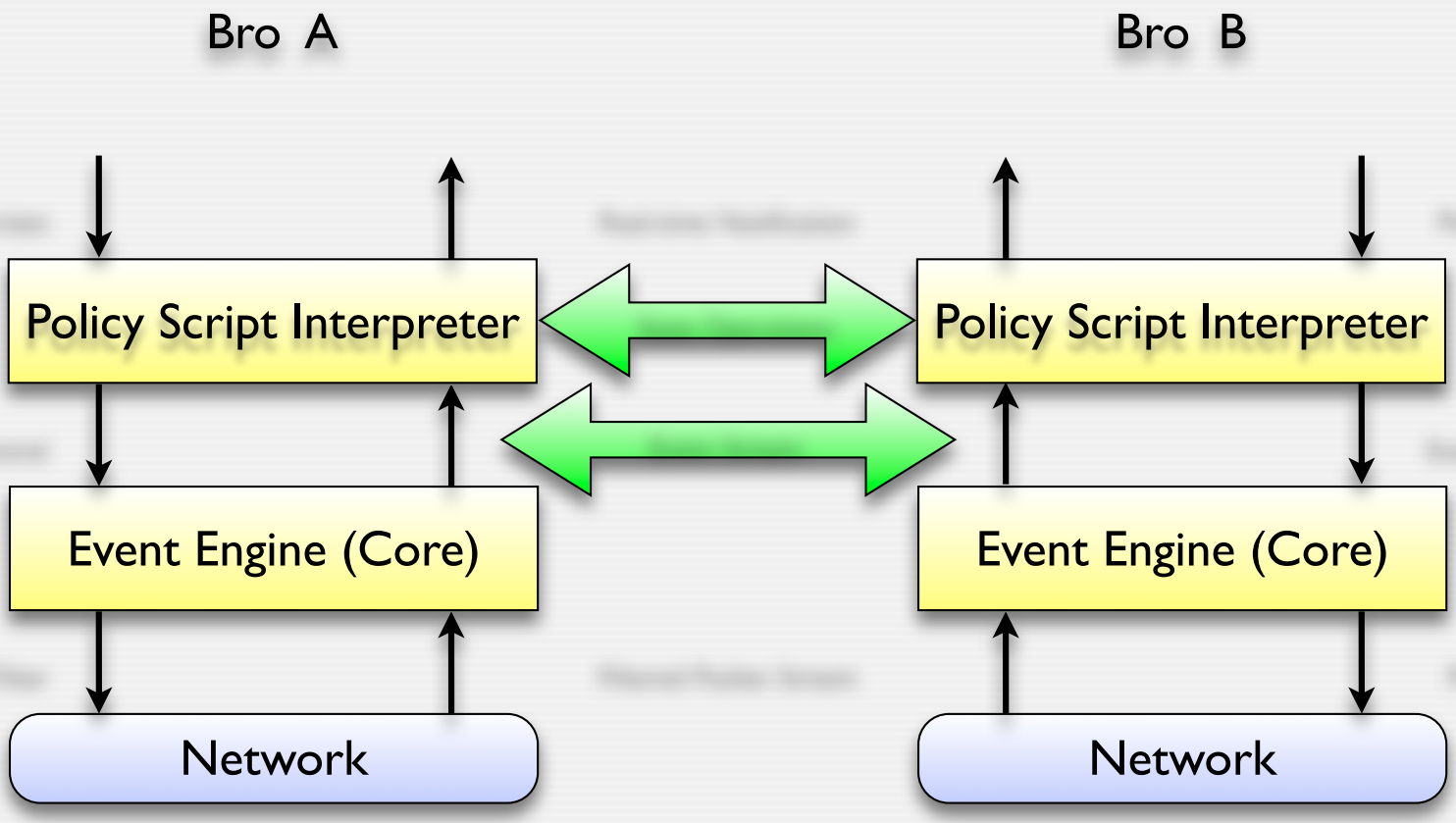
Communication Architecture



Communication Architecture



Communication Architecture

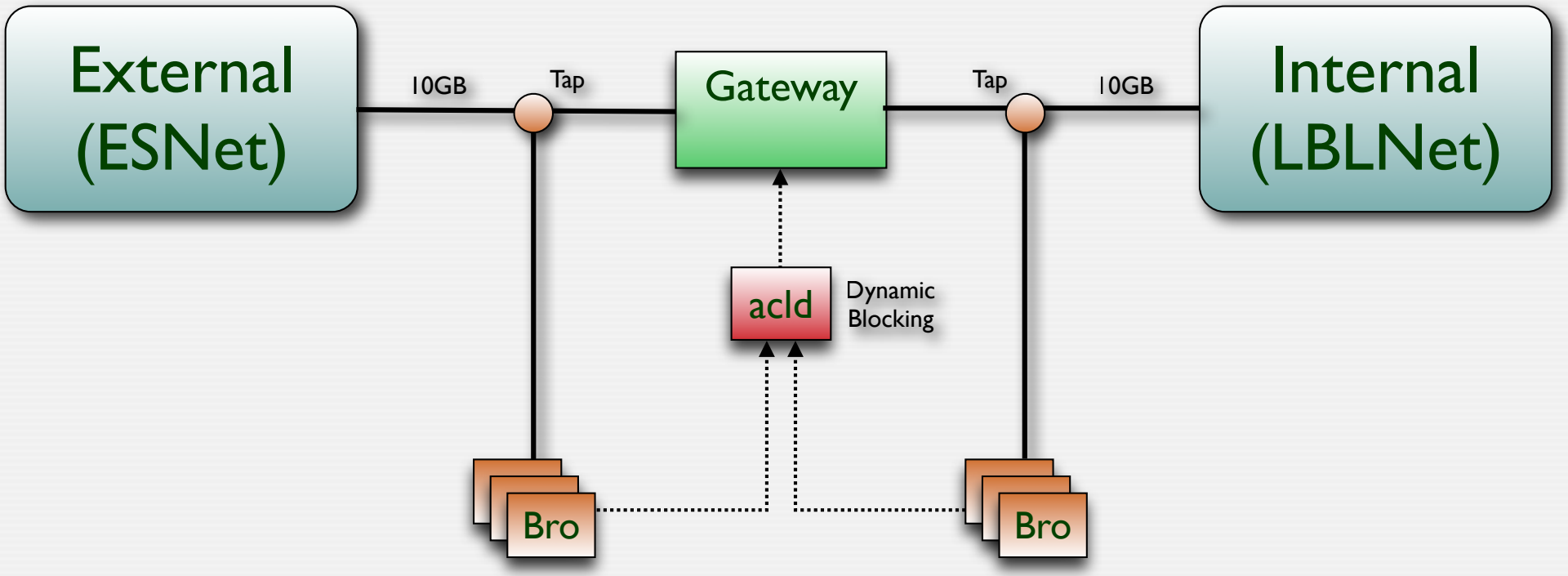


LBNL's Bro Installation

Bro at the Lawrence Berkeley Lab

- LBNL has been using Bro for >10 years
- Uses Bro to monitor its 10 Gbps Internet uplink
- Bro is one of the main components of lab's security
  - Several Bro boxes for different tasks
  - Bro automatically *blocks* attackers (~4000/day, mainly scanners)

LBNL Monitoring Setup



Recent Developments (1)
The Bro Cluster

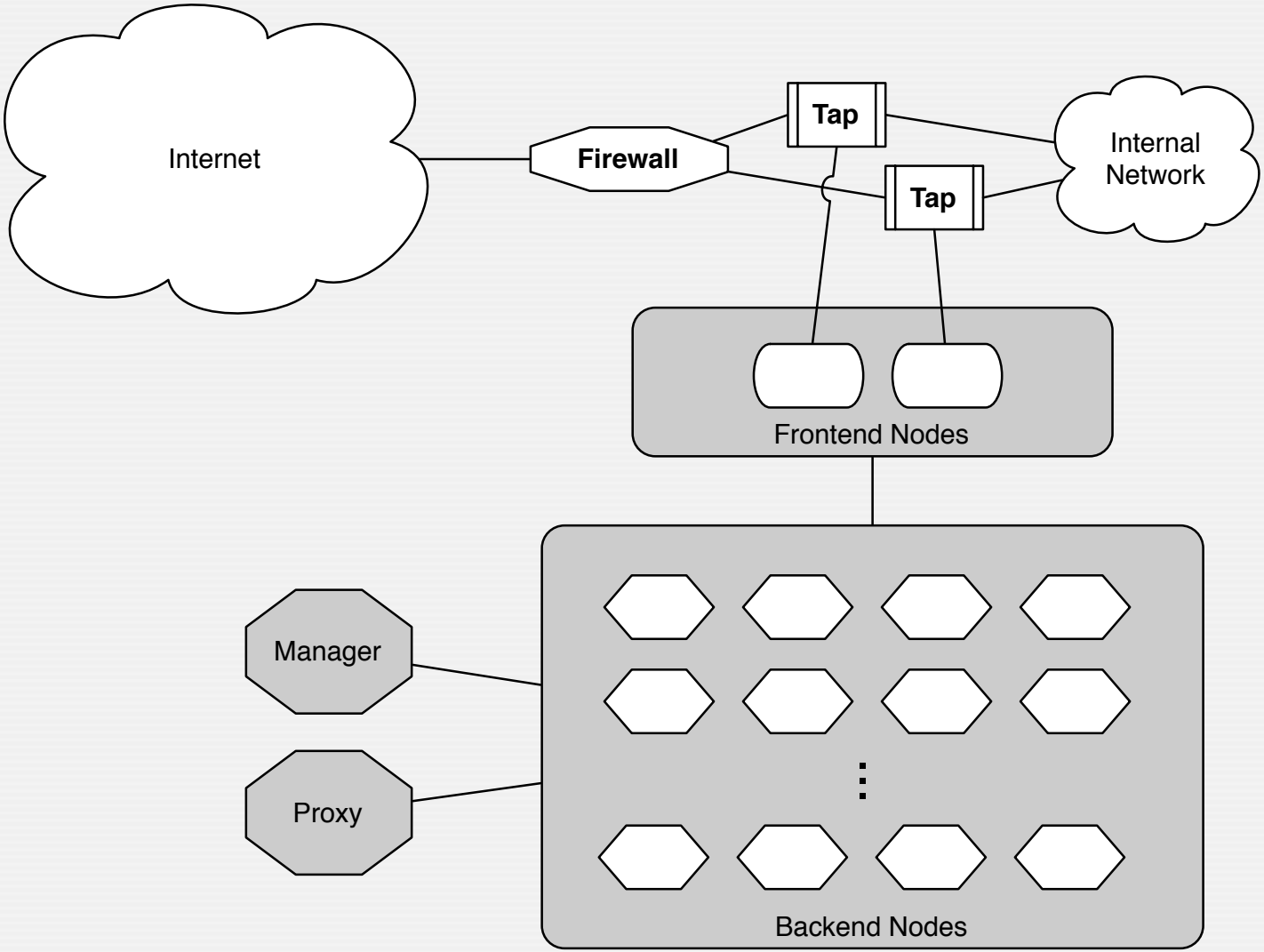
Motivation

- NIDSs reach their limits on commodity hardware
  - Keep needing to do *more analysis* on *more data* at *higher speeds*
  - Analysis gets richer over timer, as attacks get more sophisticated
  - However, single CPU performance is not growing anymore the way it used to
  - Single NIDS instance (Snort, Bro) cannot cope with >=1Gbps links
- Key to overcome current limits is *parallel analysis*
  - Volume is high but composed of many *independent* tasks
  - Need to exploit parallelism to cope with load

The Bro Cluster

- Load-balancing approach: use many boxes instead of one
- Most NIDS provide support for multi-system setups
- However instances tend to work independent
  - Central manager collects alerts of independent NIDS instances
  - Aggregates results instead of correlating analysis
- The Bro cluster works *transparently* like a single NIDS
  - Gives same results as single NIDS would if it could analyze all traffic
  - No loss in detection accuracy
  - Scalable to large number of nodes
  - Single system for user interface (log aggregation, configuration changes)

Architecture



Prototype Setups

- Lawrence Berkeley National Laboratory
  - Monitors 10 Gbps upstream link
  - 1 front-end, 10 backends
- University of California, Berkeley
  - Monitors 2x1Gbps upstream links
  - 2 front-ends, 6 backends
- IEEE Supercomputing 2006
  - Conference's 1 Gbps backbone network
  - 100 Gbps *High Speed Bandwidth Challenge* network (partially)
- Goal: Replace current operational security monitoring

Front-Ends

- Distribute traffic to back-ends by rewriting MACs
  - In software via Click (open-source "modular router")
  - In hardware via Force-10's P10 (prototype in collaboration with F10)
- Fault-tolerance
  - Easy to retarget traffic if a back-end node fails
- Per connection-hashing
  - Either 4-tuple (addrs,ports) or 2-tuple (addrs)
  - MD5 mod n, ADD mod n

Back-ends

- Running Bro as their analysis engine
- Bro provides extensive communication facilities
  - Independent state framework
  - Sharing of *low-level* state
  - Script-layer variables can be *synchronized*
- Basic approach: pick state to be synchronized
  - A few subtleties needed to be solved
- Central manager
  - Collects output of all instances
  - Raises alerts
  - Provides dynamic reconfiguration facilities

Evaluation & Outlook

- Prototypes are running nicely
  - Are able to perform analysis not possible before
  - E.g., full HTTP analysis & Dynamic Protocol Detection/Analysis
- Now in the process of making it production quality
- Evaluation
  - Verified accuracy by comparing against single Bro instance
  - Evaluated performance wrt load-balancing quality, scalability, overhead

Recent Developments (2)
Dynamic Protocol Detection

Port-based Protocol Analysis

- Bro has lots of application-layer analyzers
- But which protocol does a connection use?
- Traditionally NIDS rely on ports
  - Port 80? Oh, that's HTTP.
- Obviously deficient in two ways
  - There's non-HTTP traffic on port 80 (firewalls tend to open this port...)
  - There's HTTP on ports other than port 80
- Particularly problematic for security monitoring
  - Want to know if somebody avoids the well-known port

Port-independent Analysis

- Look at the *payload* to see what is, e.g., HTTP
- Analyzers already know how a protocol looks like
  - Leverage existing protocol analyzers
  - Let each analyzer *try to parse* the payload
    - If it succeeds, great!
    - If not, then it's actually another protocol
- Ideal setting: *for every connection, try all analyzers*
- However, performance is prohibitive
  - Can't parse 10000s of connections in parallel with all analyzers

Making it realistic ...

- Bro uses byte patterns to *prefilter* connections
  - An HTTP signature looks for *potential* uses of HTTP
  - Then the HTTP analyzer verifies by trying to parse the payload
  - Signatures can be loose because false positives are inexpensive (no alerts!)
- Other NIDS often ship with protocol signatures
  - These directly generate alerts (image reporting all non-80 HTTP conns)
  - These do not trigger protocol-layer semantic analysis (e.g., extracting URLs)
- In Bro, a match triggers further analysis
- Main internal concept: analyzer trees
  - Each connection is associated with an analyzer tree

Finding Bots

- IRC-based bots are a prevalent problem
  - Infected client machines accept commands from their "master"
  - Often IRC-based but not on port 6667
- Just detecting IRC connections not sufficient
  - Often there is legimitate IRC on ports other than 6667
- DPD allows to analyze all IRC sessions *semantically*
  - Looks for typical patterns in NICK and TOPIC
  - Reports if it find IRC sessions showing both such NICKs and TOPICs
- Very reliable detection of bots
  - Munich universities use it to actively block internal bots automatically

DPD: Summary & Outlook

- Port-independent protocol analysis
  - Idea is straight-forward, but Bro is the only system which does it
- Bro now has a very generic analyzer framework
  - Allows arbitrary changes to analyzer setup during lifetime of connection
  - Is not restricted to any particular approach for protocol detection
- Main performance impact: need to examine *all* packets
  - Well, that's pretty hard to avoid
- Potential extensions
  - More protocol-detection heuristics (e.g., statistical approaches)
  - Analyze tunnels by pipelining analyzers (e.g., to look inside SSL)
  - Hardware support for pre-filtering (e.g., on-NIC filtering)

Conclusion & Outlook

Summary

- Bro is likely one of the most powerful NIDS available
  - Open-source and runs on commodity hardware
  - While primarily a research system, it is well suited for operational use
  - One of the main components of LBNL's network security monitoring
- Largest problem: Documentation is lacking ...
  - Especially for new features ...
- Still, there is quite a bit of information available:
  - The Bro Wiki contains quite a bit of information.
  - Much of the new functionality is introduced in papers.
    See `www.bro-ids.org` for a list.
  - The most up-to-date documentation is still the CHANGES file ...
- The mailing list is open for all kinds of questions
  - From beginner to expert.

Real Soon Now ...

- Polished interface for the Bro Cluster
  - Interactive cluster shell on the manager
  - Will make it easy to install and operate a Bro Cluster
- New functionality
  - BitTorrent analyzer
  - SIP analyzer
  - XML analyzer with XQuery support
  - NetFlow analyzer
  - SSL analyzer rewritten in BinPac
  - Time Machine interface

    See `http://www.net.t-labs.tu-berlin.de/research/tm`

The Future ...

- Multi-core Support
  - Going to turn Bro into a multi-threaded application
  - Will fully exploit the multi-core potential of modern CPUs
  - Involves a comprehensive performance analysis of the core
- Distributed Data Sharing
  - Extending Bro with functionality required for inter-site cooperation.
  - Building a more powerful communication platform
  - Providing hooks to tightly control which data is exchanged

Any questions ... ?

Robin Sommer
Lawrence Berkeley National Laboratory

rsommer@lbl.gov
http://www.icir.org

This work is supported by the Office of Science and Technology at the Department of Homeland Security. Points of view in this document are those of the author(s) and do not necessarily represent the official position of the U.S. Department of Homeland Security or the Office of Science and Technology.