Reflecting on Twenty Years of Bro

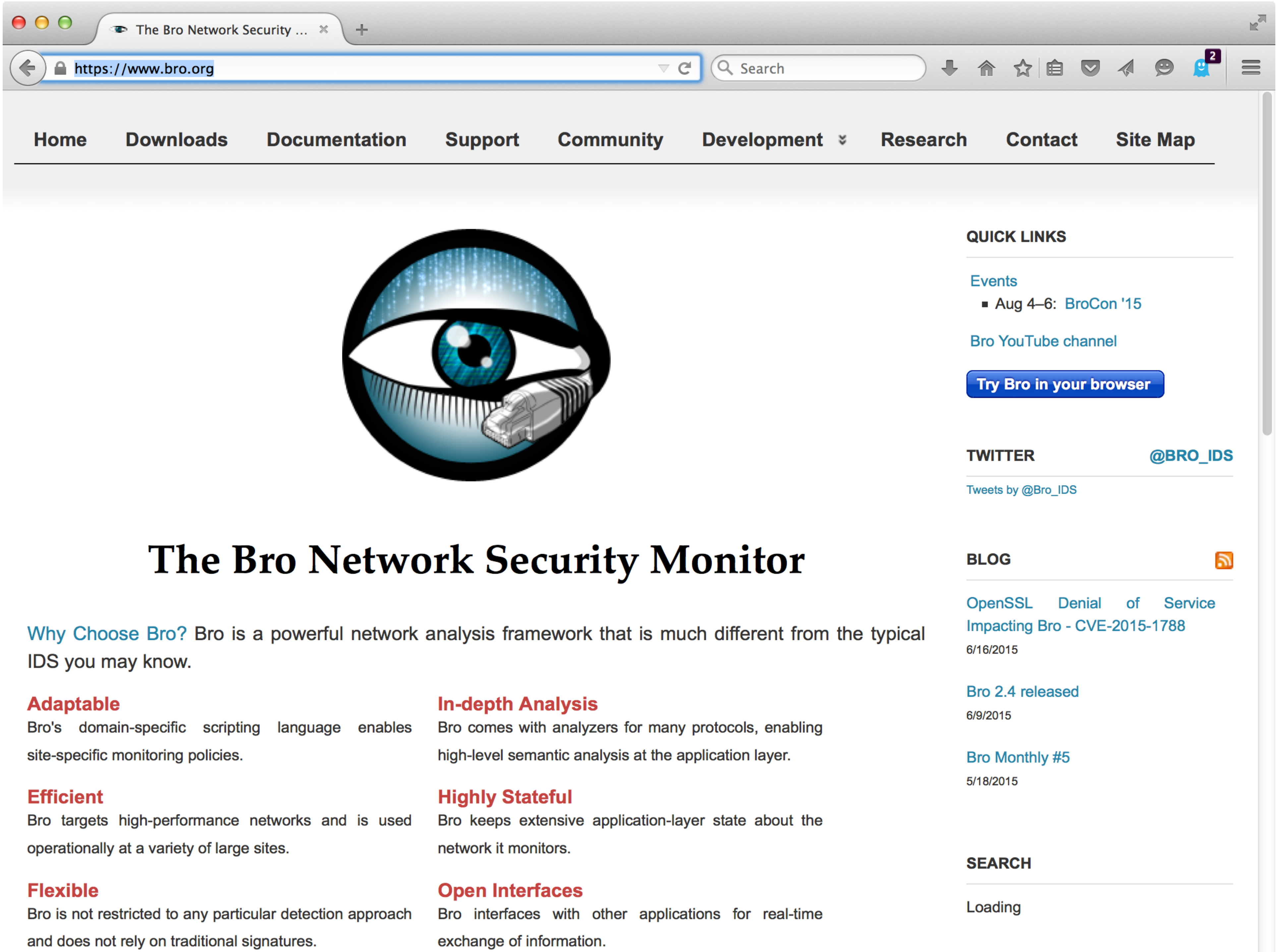
Vern Paxson

International Computer Science Institute
Lawrence Berkeley National Laboratory
EECS Department, University of California, Berkeley
Broala, LLC

vern@berkeley.edu

August 4, 2015

Part I:
Origin & Technical Evolution


The Bro Network Security ...
https://www.bro.org
Search
Home
Downloads
Documentation
Support
Community
Development
Research
Contact
Site Map
The Bro Network Security Monitor
Why Choose Bro? Bro is a powerful network analysis framework that is much different from the typical IDS you may know.
Adaptable
Bro's domain-specific scripting language enables site-specific monitoring policies.
In-depth Analysis
Bro comes with analyzers for many protocols, enabling high-level semantic analysis at the application layer.
Efficient
Bro targets high-performance networks and is used operationally at a variety of large sites.
Highly Stateful
Bro keeps extensive application-layer state about the network it monitors.
Flexible
Bro is not restricted to any particular detection approach and does not rely on traditional signatures.
Open Interfaces
Bro interfaces with other applications for real-time exchange of information.
QUICK LINKS
Events
Aug 4–6: BroCon '15
Bro YouTube channel
Try Bro in your browser
TWITTER
@BRO_IDS
Tweets by @Bro_IDS
BLOG
OpenSSL Denial of Service Impacting Bro - CVE-2015-1788
6/16/2015
Bro 2.4 released
6/9/2015
Bro Monthly #5
5/18/2015
SEARCH
Loading

Bro: A System for Detecting Network Intruders in Real-Time

Vern Paxson
Network Research Group
Lawrence Berkeley National Laboratory*
Berkeley, CA 94720
vern@ee.lbl.gov

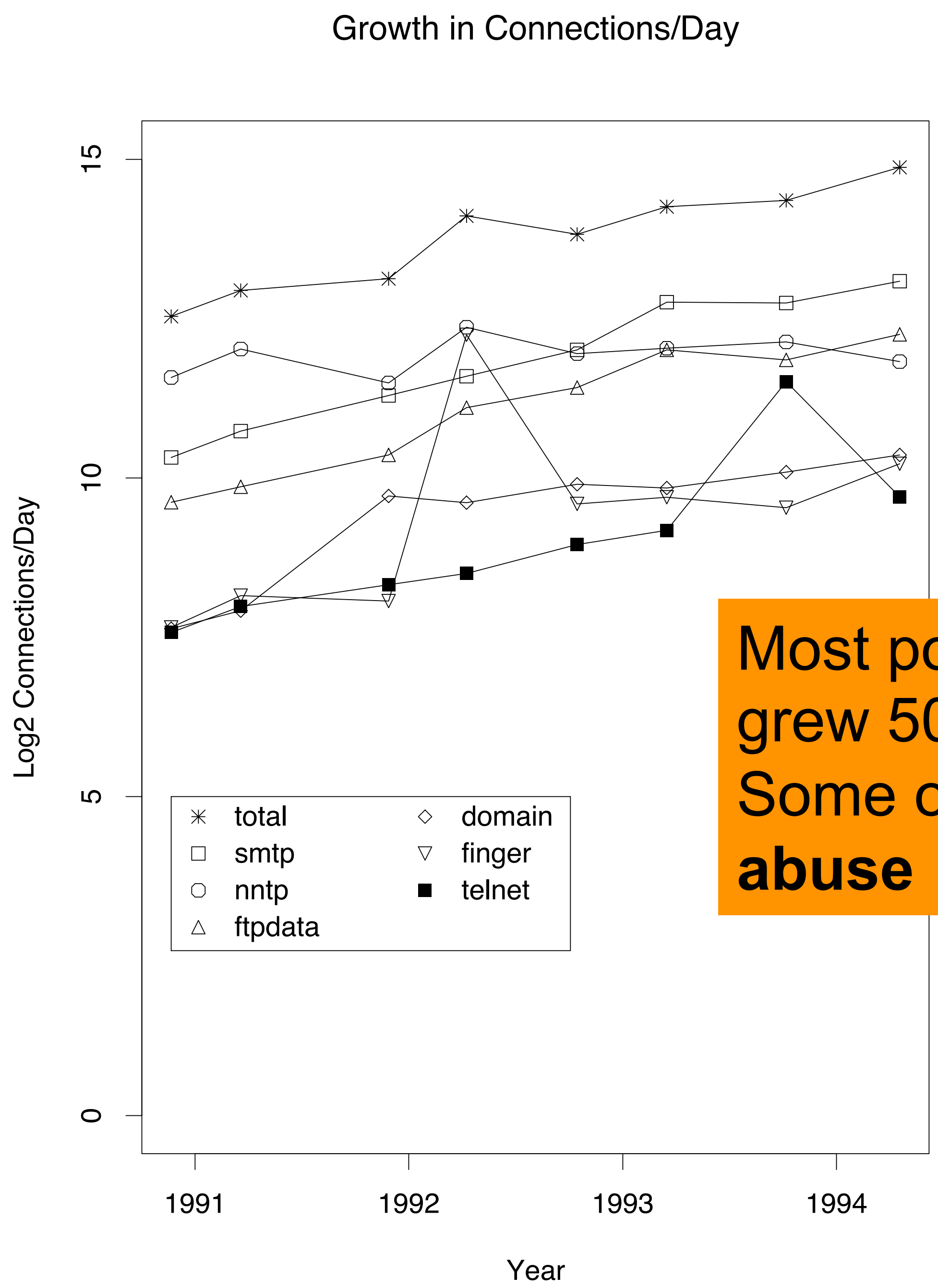
USENIX Technical Program - 7th USENIX Security Symposium, 1998

Prior to developing Bro, we had significant operational experience with a simpler system based on off-line analysis of `tcpdump` [JLM89] trace files. Out of this experience we formulated a number of design goals and requirements:

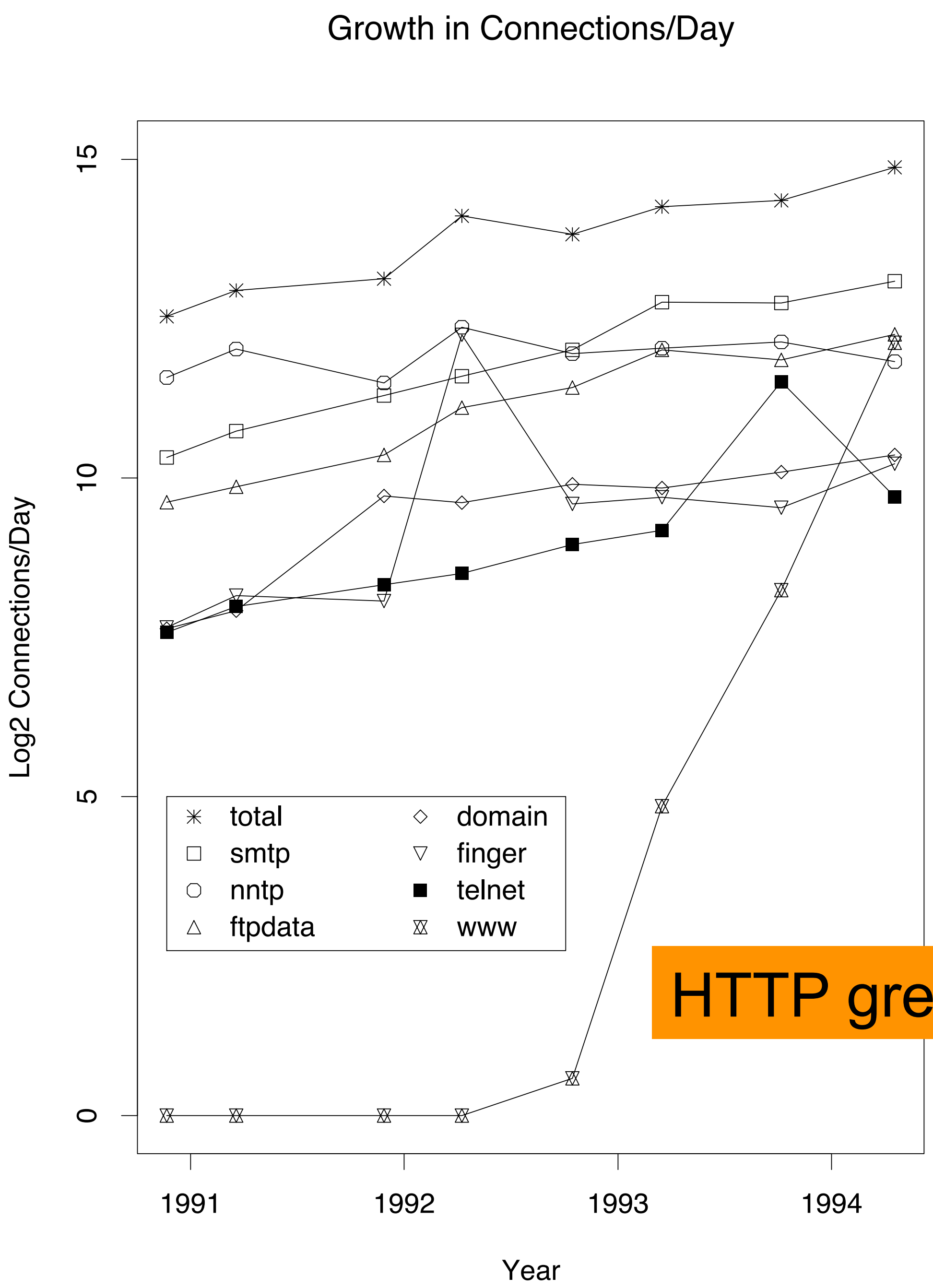
Growth Trends in Wide-Area TCP Connections

Vern Paxson
Lawrence Berkeley Laboratory and
EECS Division, University of California, Berkeley
1 Cyclotron Road
Berkeley, CA 94720
vern@ee.lbl.gov

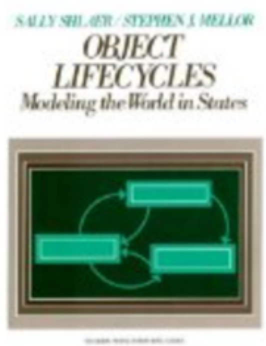
Revised May 11, 1994



Most popular protocols grew 50-70%/year; Some of this fueled by **abuse**



HTTP grew 300x / year!



Object Life Cycles: Modeling the World In States Sep 11, 1991
by Stephen J. Mellor and Sally Shlaer

Paperback
$83.00 ✓Prime
Only 1 left in stock - order soon.

More Buying Choices
$0.01 used & new (63 offers)

Congestion Avoidance and Control

Van Jacobson*

University of California
Lawrence Berkeley Laboratory
Berkeley, CA 94720
van@helios.ee.lbl.gov

TCPDUMP(1)

NAME

tcpdump - dump traffic on a network

SYNOPSIS

tcpdump [**-AbdDefhgHIJKlLnNOpPqRStuUvxX**] [**-B** buffer_size] [**-c** count]
[**-C** file_size] [**-G** rotate_seconds] [**-F** file]
[**-i** interface] [**-j** tstamp_type] [**-k** (metadata_arg)]
[**-m** module] [**-M** secret]

PCAP(3PCAP) PCAP(3PCAP)

NAME

pcap - Packet Capture library

SYNOPSIS

#include <pcap/pcap.h>

DESCRIPTION

The Packet Capture library provides a high level interface to packet capture systems. All packets on the network, even those destined for

Prior to developing Bro, we had significant operational experience with a simpler system based on off-line analysis of `tcpdump` [JLM89] trace files. Out of this experience we formulated a number of design goals and requirements:

High-speed, large volume monitoring

No packet filter drops

Real-time notification

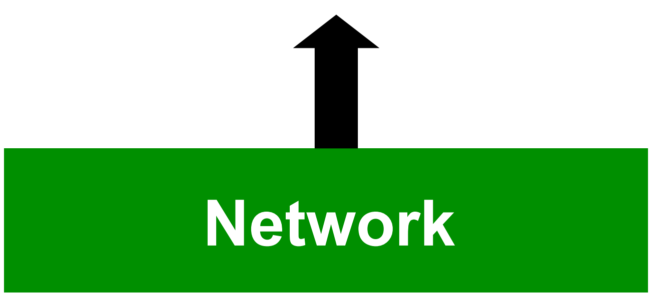
Mechanism separate from policy

Extensible

Avoid simple mistakes

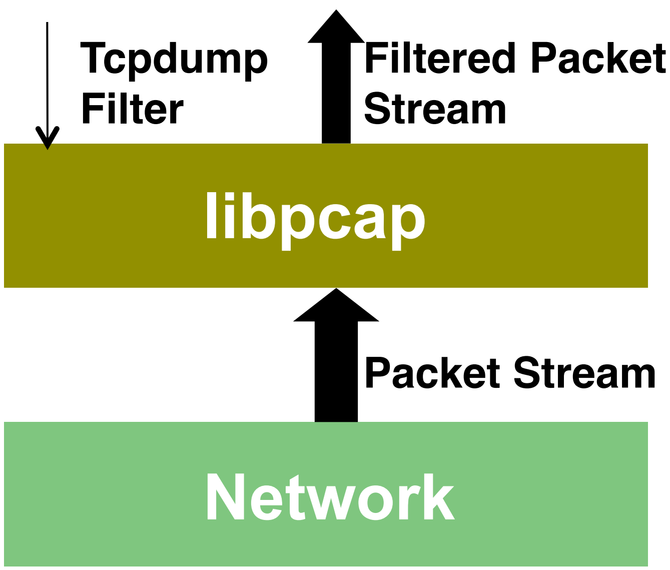
The monitor will be attacked

Original Architecture



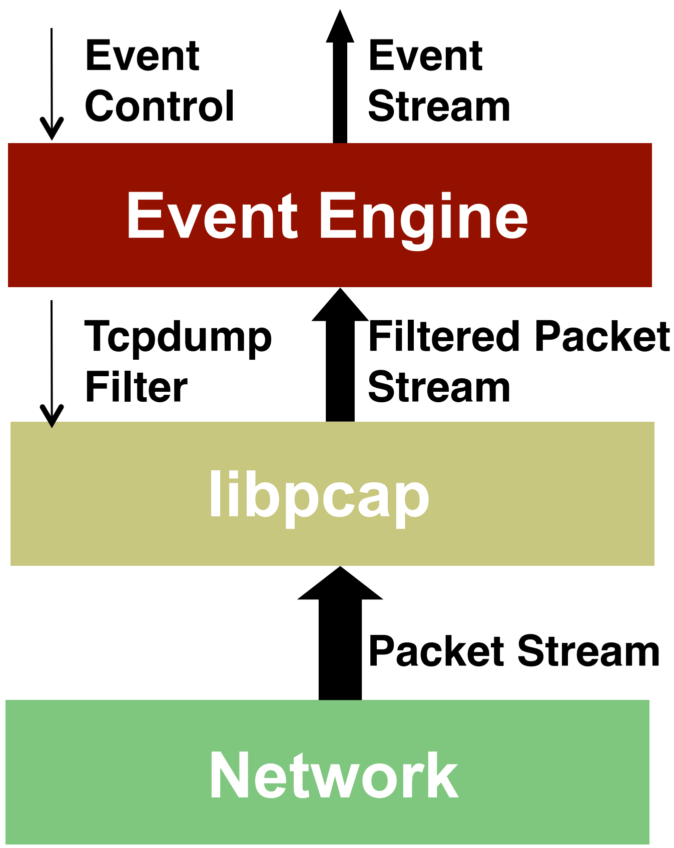
- Taps network link passively, sends up a copy of all network traffic.

Original Architecture



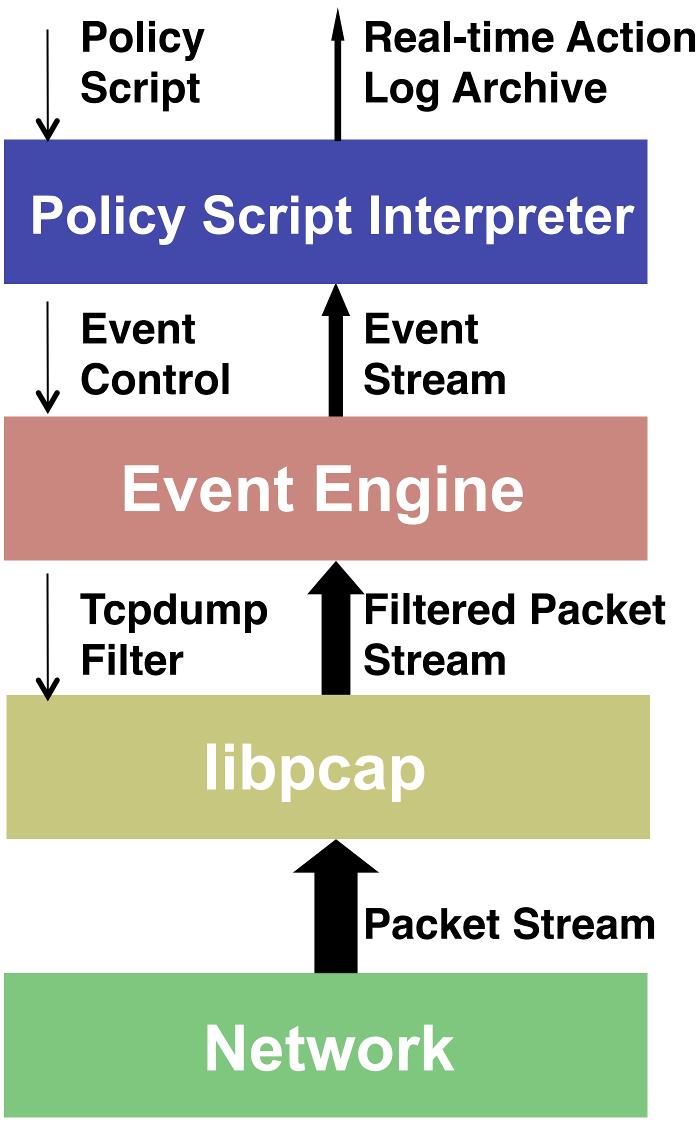
- Kernel filters down high-volume stream via standard *libpcap* packet capture library.

Original Architecture



- "Event engine" decodes protocols, distills filtered stream into high-level, *policy-neutral* events reflecting underlying network activity
  - E.g., connection_attempt, http_reply, teredo_authentication
  - These span a range of semantic levels
  - Currently 400+ different types

Original Architecture

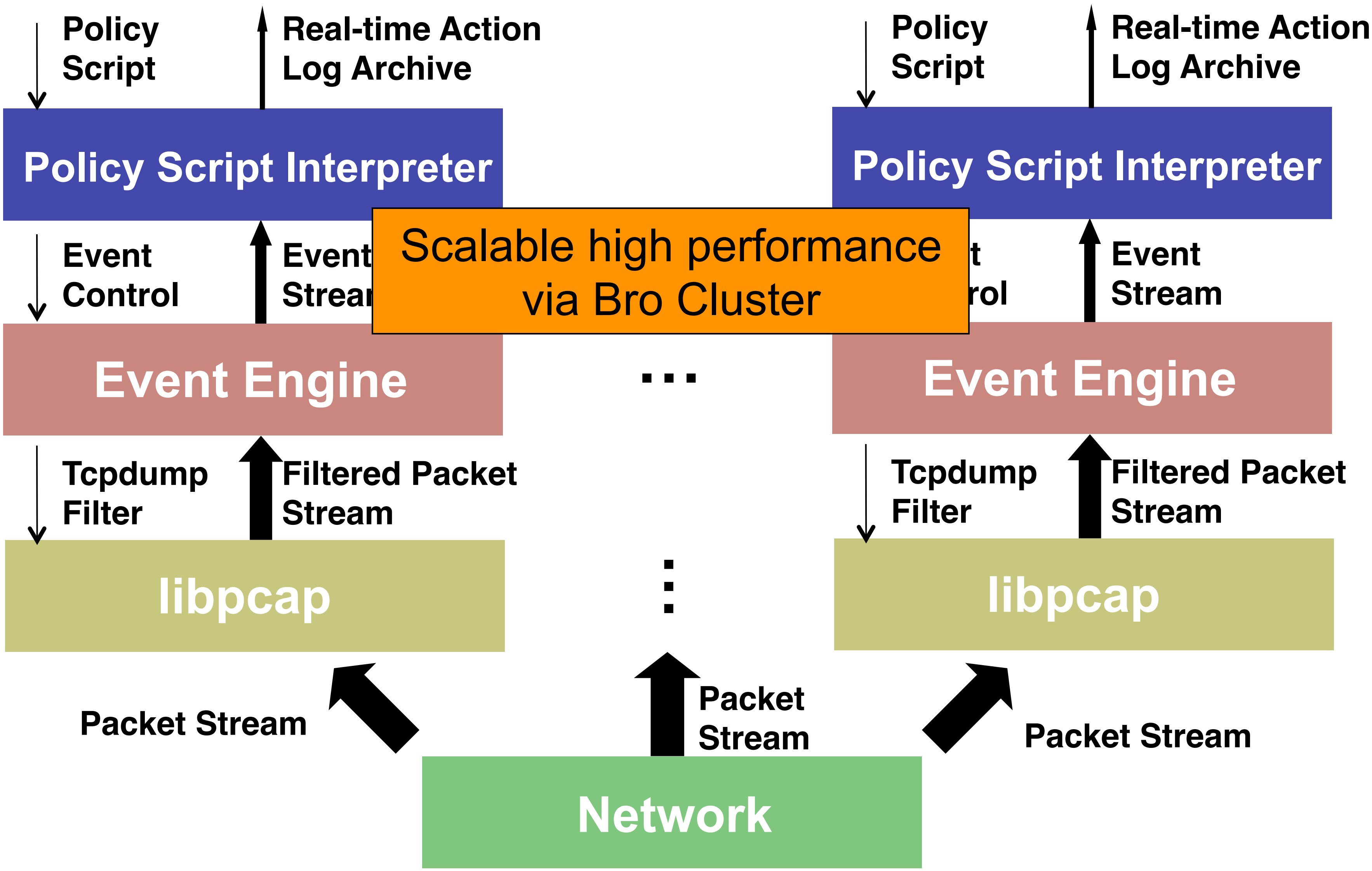


- Script written in Domain Specific Language processes event stream, incorporates:
  - Context/state from past events
  - Additional input sources
  - Site's particular policies

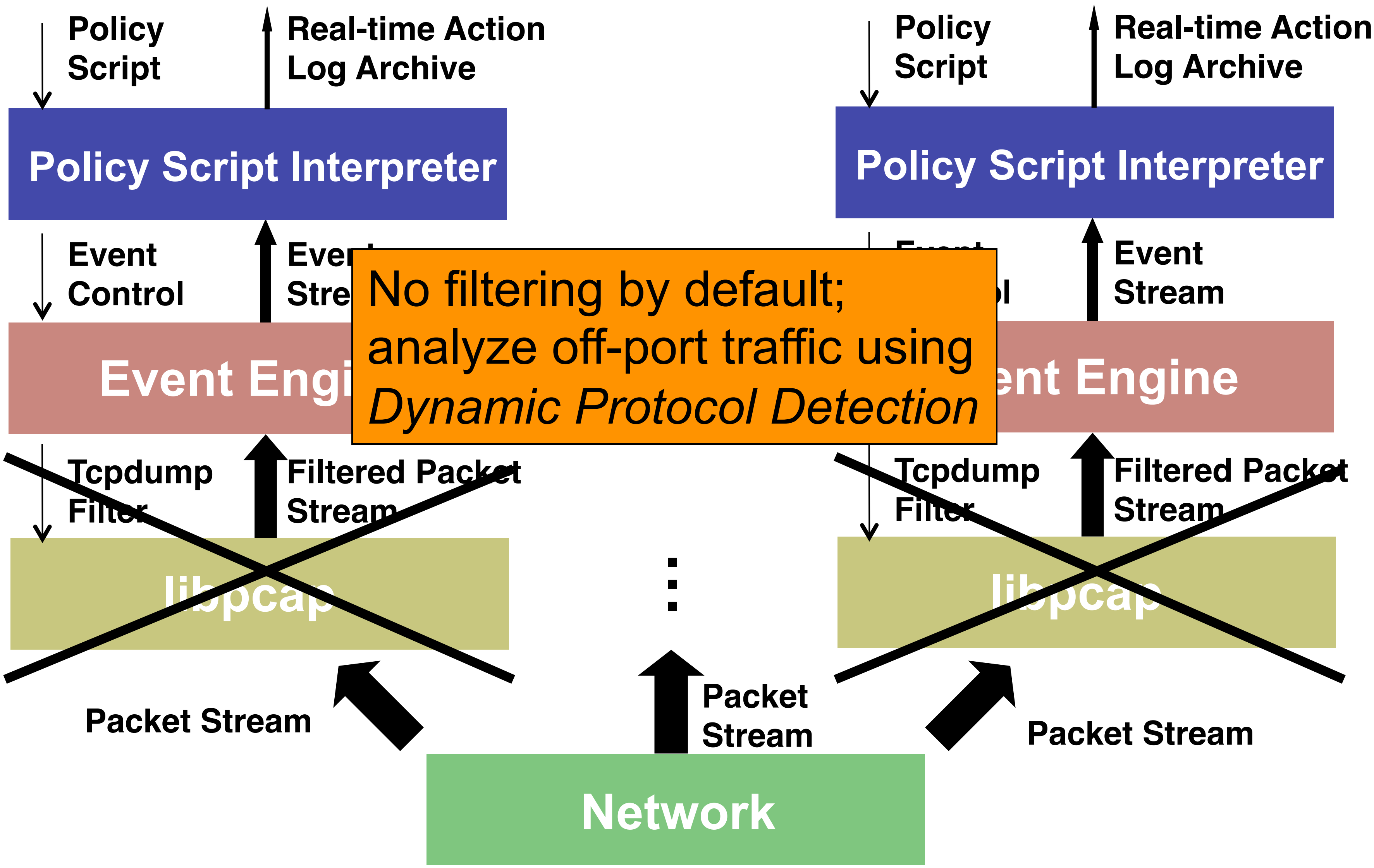
… and *takes action*:

Records to disk - **extensive** logs
Generates real-time alerts
Executes programs as a form of **response**

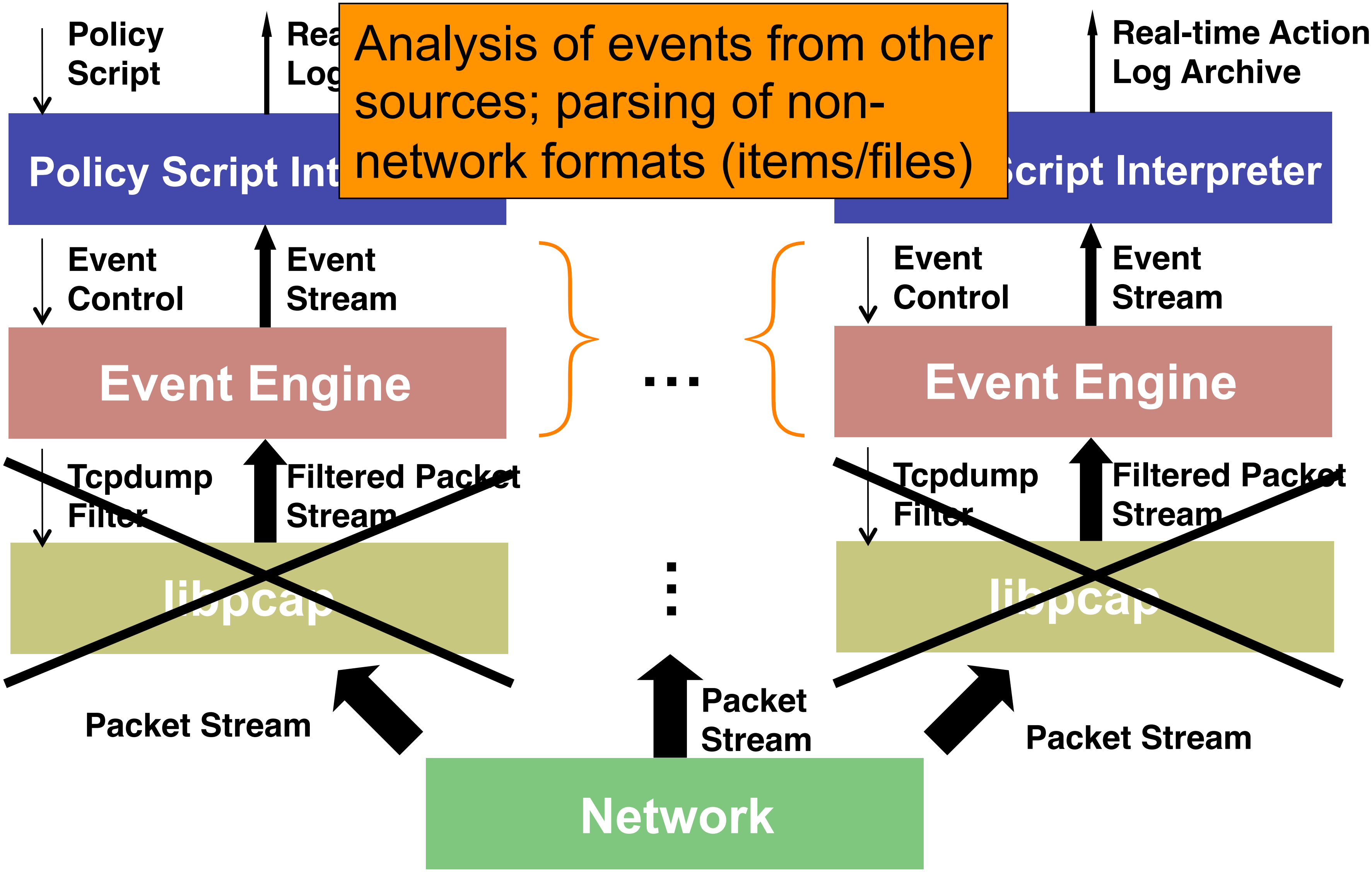
Architecture As It Has Evolved



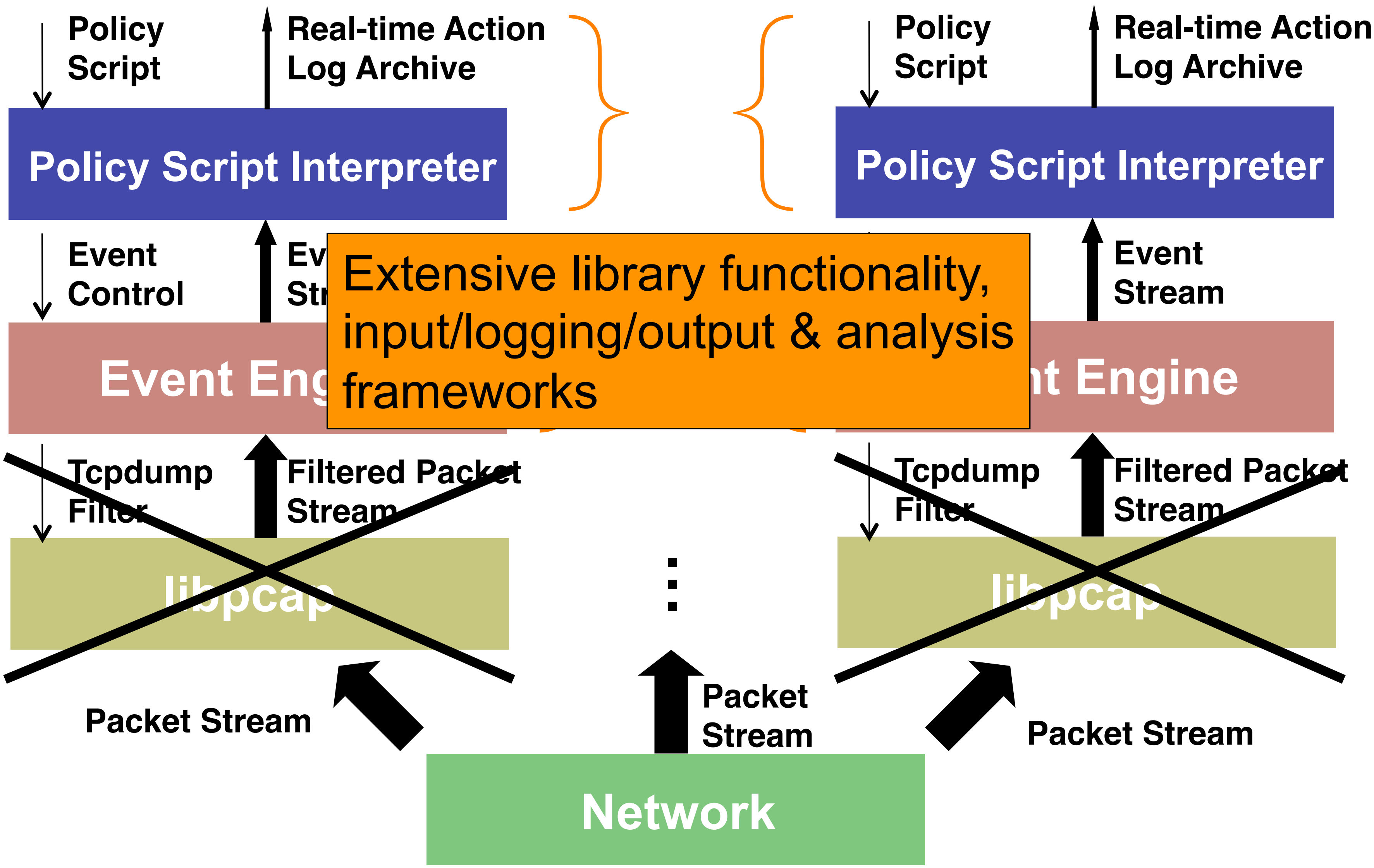
Architecture As It Has Evolved



Architecture As It Has Evolved



Architecture As It Has Evolved



Prior to developing Bro, we had significant operational experience with a simpler system based on off-line analysis of `tcpdump` [JLM89] trace files. Out of this experience we formulated a number of design goals and requirements:

High-speed, large volume monitoring

High-speed, large volume monitoring For our environment, we view the greatest source of threats as external hosts connecting to our hosts over the Internet. Since the network we want to protect has a single link connecting it to the remainder of the Internet (a "DMZ"), we can economically monitor our greatest potential source of attacks by passively watching the DMZ link.

Key enabler: donation of DEC Alphas (kudos Jeff Mogul)

Prior to developing Bro, we had significant operational experience with a simpler system based on off-line analysis of `tcpdump` [JLM89] trace files. Out of this experience we formulated a number of design goals and requirements:

```
Institution:
        Lawrence Berkeley Laboratory
        University of California
        Berkeley, CA  94720

Research Title:
        Real-time detection of network intruders

Date: 08 February 1995

        List_Price of Digital Products                          ca. $24,000
```

digital

we can economically monitor our greatest potential source of attacks by passively watching the DMZ link.

Key enabler: donation of DEC Alphas (kudos Jeff Mogul)

Prior to developing Bro, we had significant operational experience with a simpler system based on off-line analysis of `tcpdump` [JLM89] trace files. Out of this experience we formulated a number of design goals and requirements:

High-speed, large volume monitoring

No packet filter drops

No packet filter drops If an application using a packet filter cannot consume packets as quickly as they arrive on the monitored link, then the filter will buffer the packets for later consumption. However, eventually the filter will run out of buffer, at which point it *drops* any further packets that arrive. From a security monitoring perspective, drops can completely defeat the monitoring, since the missing packets might contain exactly the interesting traffic that identifies a network intruder.

...tion

...istakes

Prior to developing Bro, we had significant operational experience with a simpler system based on off-line analysis of `tcpdump` [JLM89] trace files. Out of this experience we formulated a number of design goals and requirements:

High-speed, large volume monitoring

No packet filter drops

`@load misc/capture-loss`

No packet filter drops If an application using a packet filter cannot consume packets as quickly as they arrive on the monitored link, then the filter will buffer the packets for later consumption. However, eventually the filter will run out of buffer, at which point it *drops* any further packets that arrive. From a security monitoring perspective, drops can completely defeat the monitoring, since the missing packets might contain exactly the interesting traffic that identifies a network intruder.

tion

istakes

Prior to developing Bro, we had significant operational experience with a simpler system based on off-line analysis of `tcpdump` [JLM89] trace files. Out of this experience we formulated a number of design goals and requirements:

High-speed, large volume monitoring

No packet filter drops

Real-time notification

Mechanism separate from policy

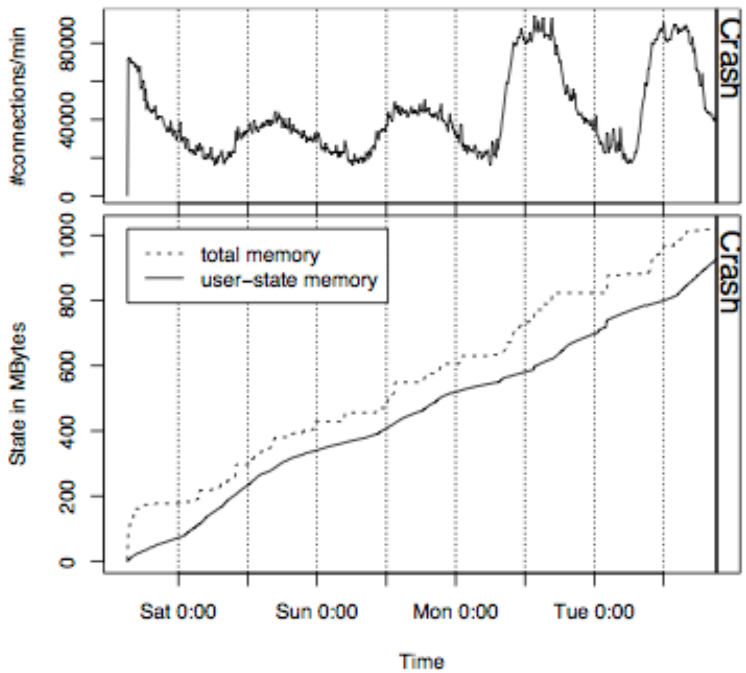
Extensible

Avoid simple mistakes

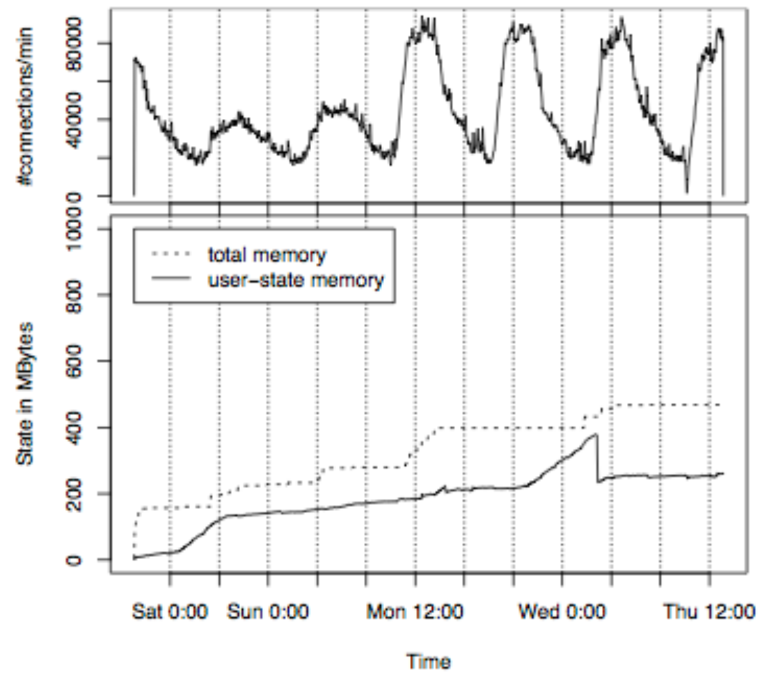
The monitor will be attacked

Operational Experiences with High-Volume Network Intrusion Detection

Figure 2: Memory required by scan detector on `mwn-week-hdr` using inactivity timeouts for connections.



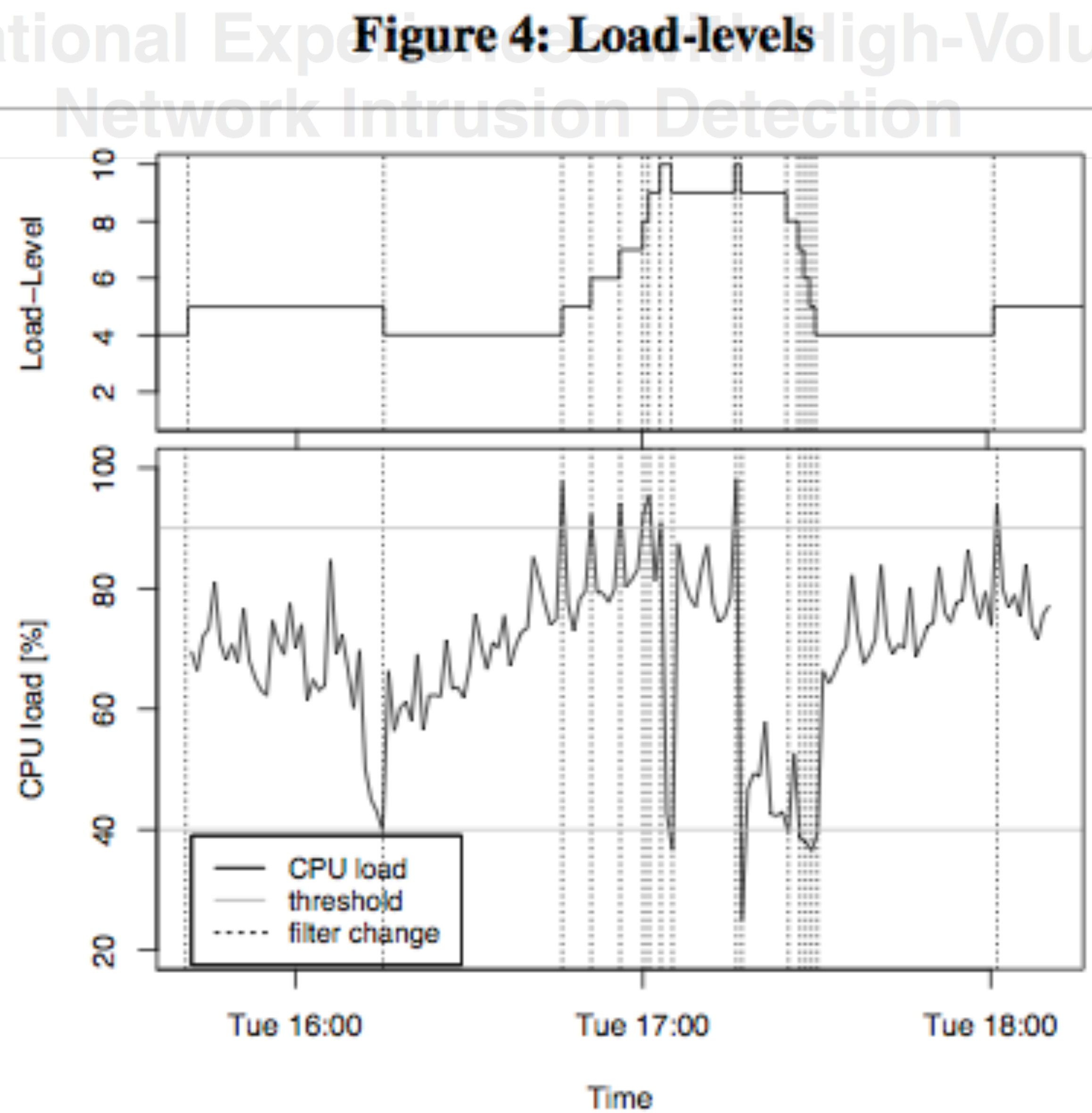
(a) Default configuration



(b) With user-level timeouts

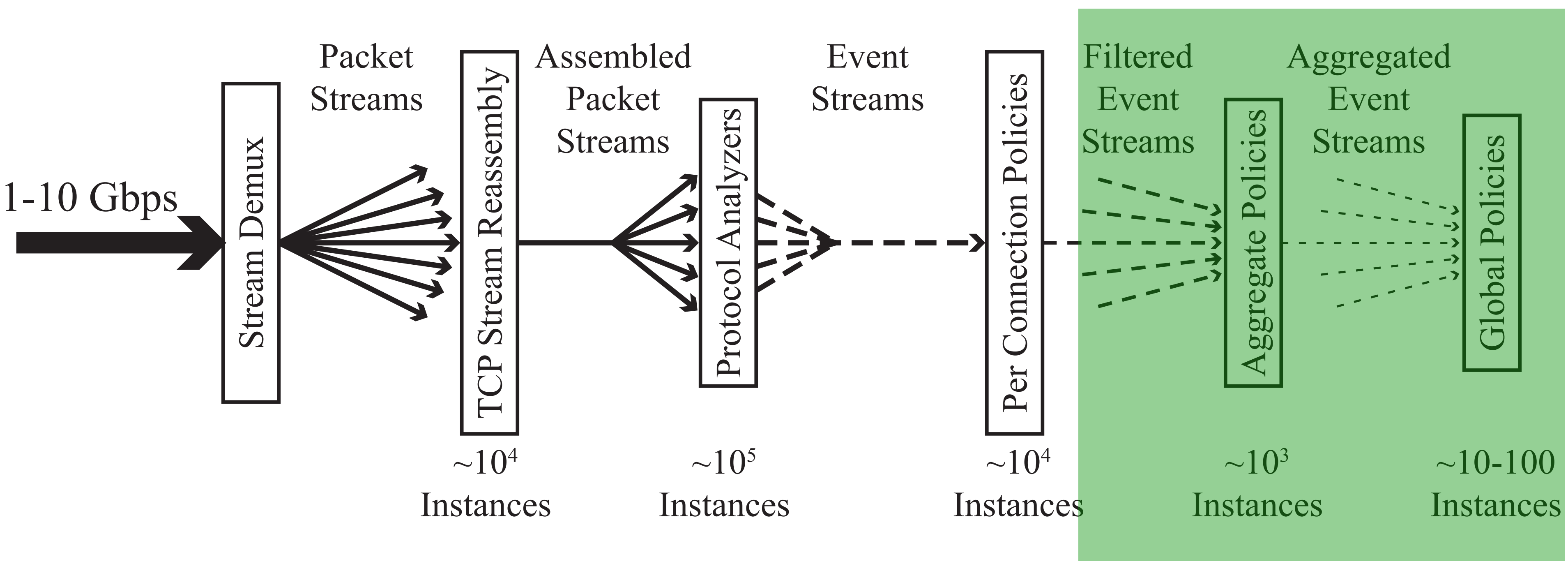
2004

Operational Experiences with High-Volume Network Intrusion Detection



2004

Rethinking Hardware Support for Network Analysis and Intrusion Prevention



2006

The NIDS Cluster: Scalable, Stateful Network Intrusion Detection on Commodity Hardware

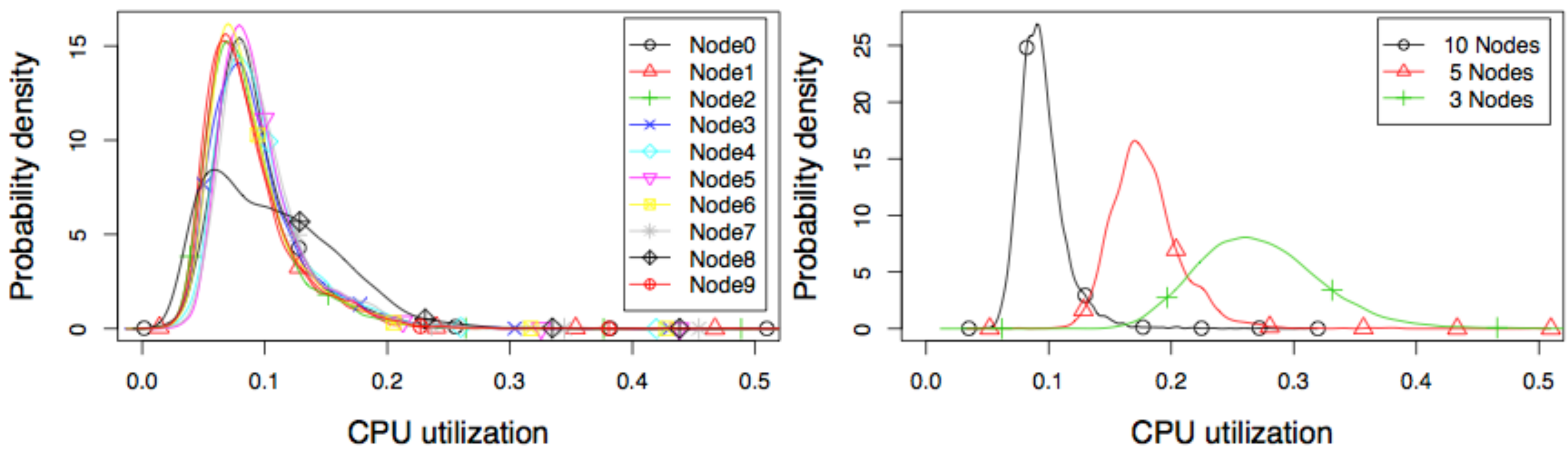


Fig. 3. Probability densities of backends' CPU load (left), and probability densities for varying numbers of backends (right).

2007

Shunting: A Hardware/Software Architecture for Flexible, High-Performance Network Intrusion Prevention

The Shunt: An FPGA-Based Accelerator for Network Intrusion Prevention

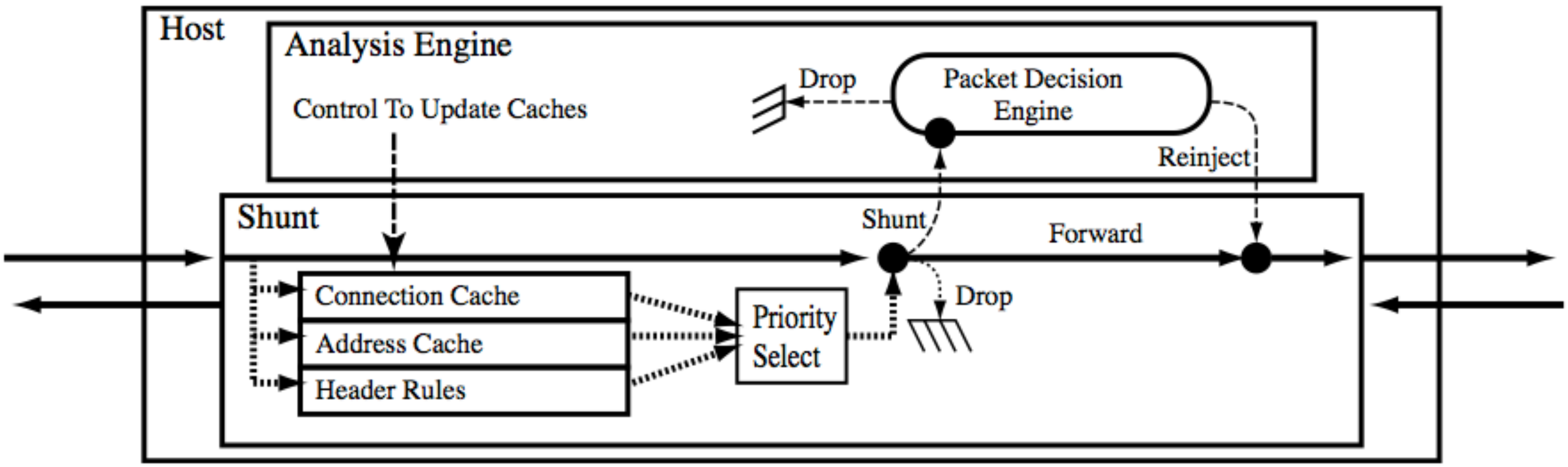
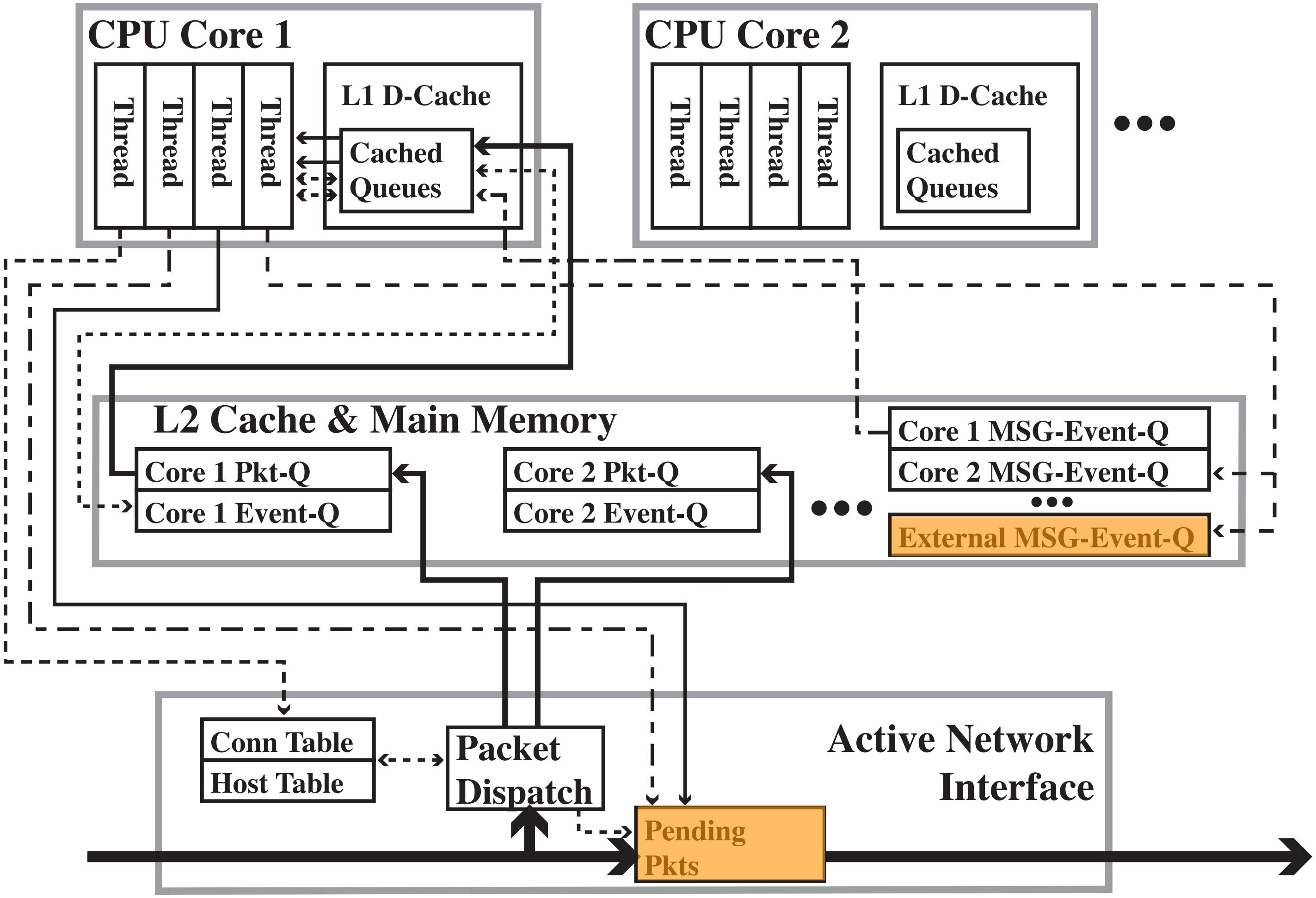


Figure 1: Shunting Main Architecture. The shunt examines the headers of received packets to determine the associated action: *forward*, *drop*, or *shunt* to the Analysis Engine. The Analysis Engine directly updates the Shunt's caches to control future processing, and either drops analyzed packets for immediate intrusion prevention or reinjects them once vetted for safety.

2007

An Architecture for Exploiting Multi-Core Processors to Parallelize Network Intrusion Prevention



2009

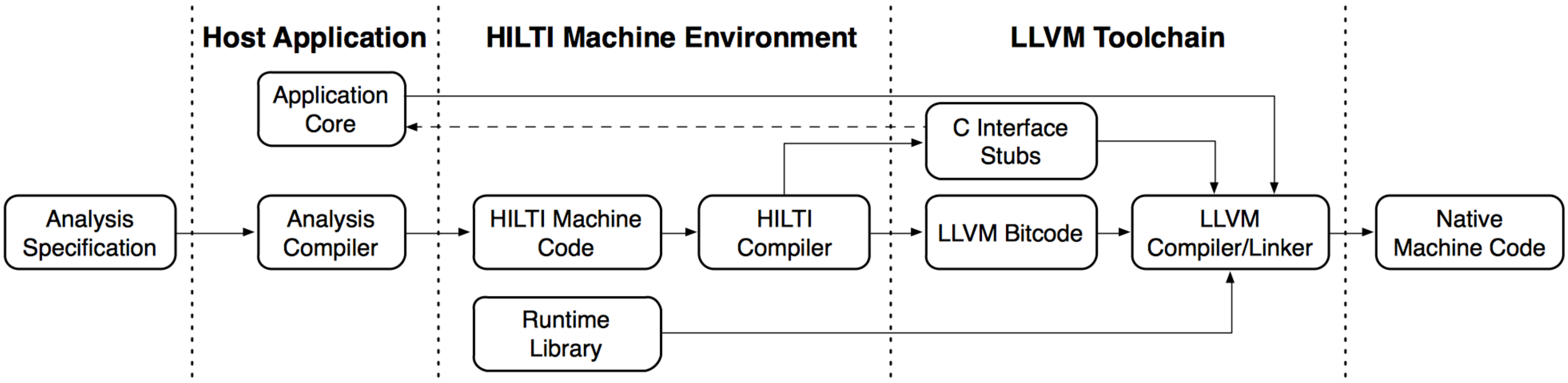
HILTI: An Abstract Execution Environment for Deep, Stateful Network Traffic Analysis

Functionality	Mnemonic	Functionality	Mnemonic
Bitsets	`bitset`	Packet i/o	`iosrc`
Booleans	`bool`	Packet classification	`classifier`
CIDR masks	`network`	Packet dissection	`overlay`
Callbacks	`hook`	Ports	`port`
Closures	`callable`	Profiling	`profiler`
Channels	`channel`	Raw data	`bytes`
Debug support	`debug`	References	`ref`
Doubles	`double`	Regular expressions	`regexp`
Enumerations	`enum`	Strings	`string`
Exceptions	`exception`	Structs	`struct`
File i/o	`file`	Time intervals	`interval`
Flow control	(No joint prefix)	Timer management	`timer_mgr`
Hashmaps	`map`	Timers	`timer`
Hashsets	`set`	Times	`time`
IP addresses	`addr`	Tuples	`tuple`
Integers	`int`	Vectors/arrays	`vector`
Lists	`list`	Virtual threads	`thread`

Table 1: HILTI's main instruction groups.

2014
(2009)

HILTI: An Abstract Execution Environment for Deep, Stateful Network Traffic Analysis



Traffic Analysis Building Blocks

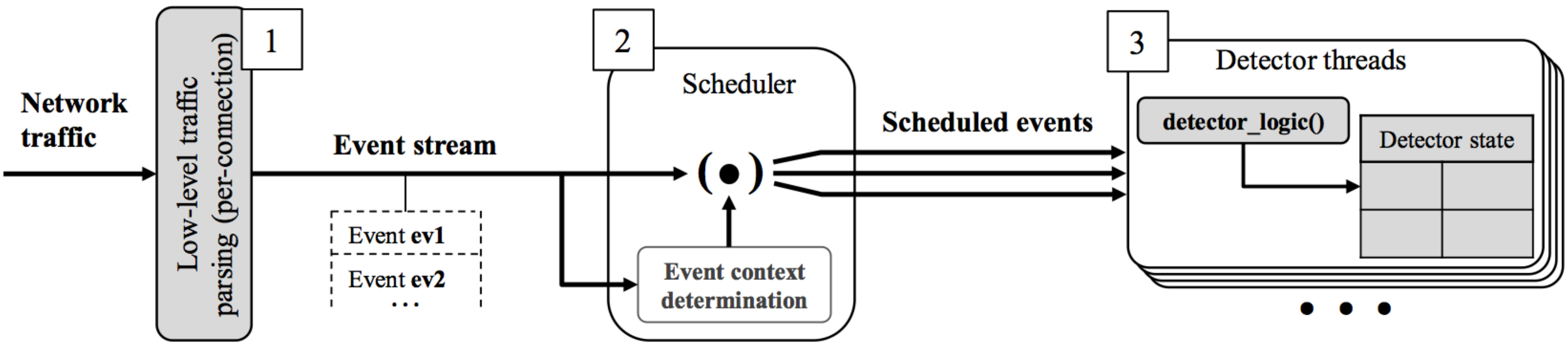
Domain-specific data types	State management	Concurrent analysis	Real-time performance	Robust execution	High-level components

HILTI Environment

Built-in first-class networking types	Containers with state management support	Domain-specific concurrency model	Compilation to native code	Contained execution environment	Platform for building reusable functionality
	Asynchronous and timer-driven execution	Incremental processing	Extensive optimization potential	Static type system	

2014
(2009)

Beyond Pattern Matching: A Concurrency Model for Stateful Deep Packet Inspection



	SINGLE-THREADED IDS	*CONCURRENT IDS (LOCK-BASED)*	*CONCURRENT IDS (SCOPE-BASED)*

IDS LOGIC

SINGLE-THREADED IDS

```
void run_IDS() {
  while ( p = read_packet() ) {
    if ( p.SYN )
      count_connections(p);
  }
}
```

CONCURRENT IDS (LOCK-BASED)

```
void run_IDS() {
  i = 0;
  while ( p = read_packet() ) {
    if ( p.SYN ) {
      event c = new connectionEvent(p);
      send_event(threads[i], c);
      i = (i+1) % N;
}}}
```

CONCURRENT IDS (SCOPE-BASED)

```
void run_IDS() {
  while ( p = read_packet() ) {
    if ( p.SYN ) {
      event c = new connectionEvent(p);
      send_event(threads[c.src % N], c);
    }
  }
}
```

DETECTOR

```
void count_connections(packet p)
{
  if (++counts[p.src] > THRESH)
    report_host(p.src);
}
```

(a)

```
handler count_connections(connectionEvent c)
{
  lock_element(counts[c.src])
  v = ++counts[c.src];
  unlock_element(counts[c.src])
  if ( v > THRESH )
    report_host(c.src);
}
```

(b)

```
handler count_connections(connectionEvent c)
{
  if (++counts[c.src] > THRESH)
    report_host(c.src);
}
```

(c)

Figure 1: Simple portscan detector

2014

Count Me In: Viable Distributed Summary Statistics for Securing High-Speed Networks

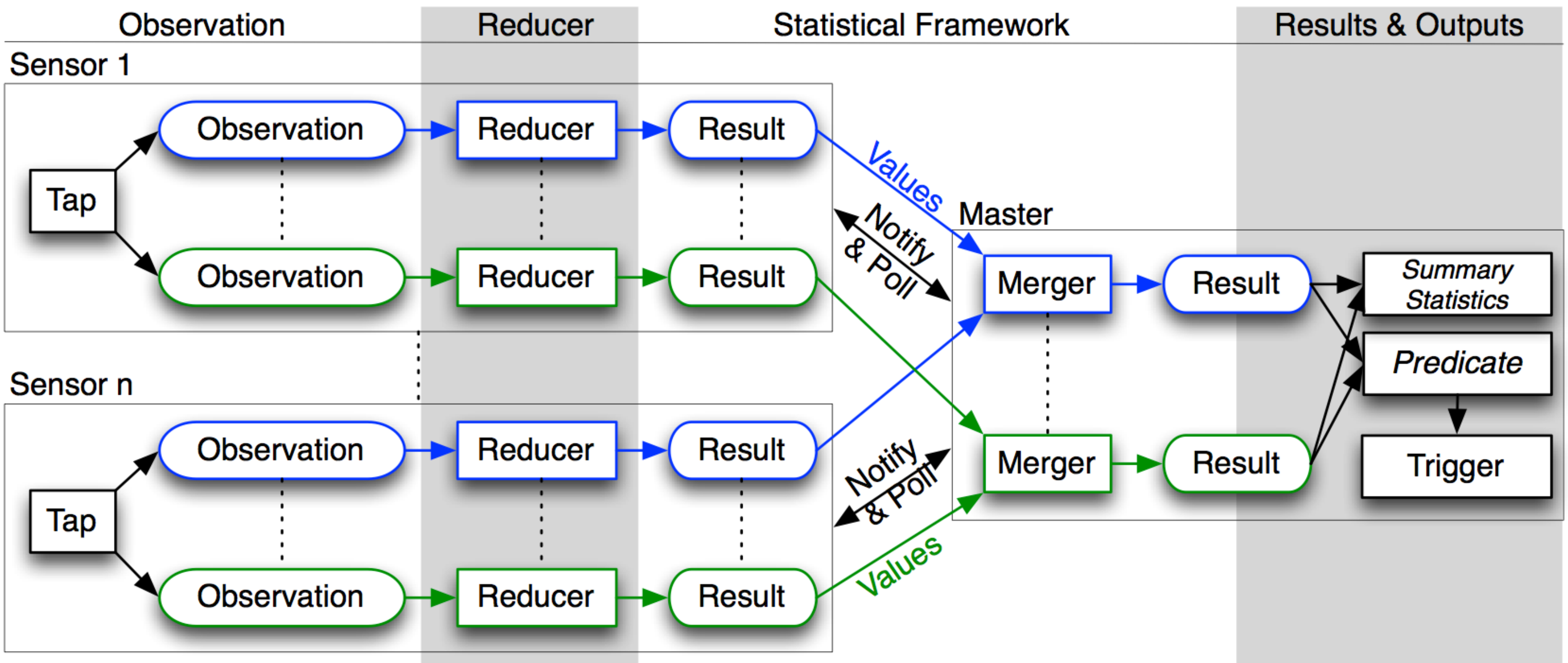


Fig. 2. Distributed Architecture.

2014

BERKELEY LAB
LAWRENCE BERKELEY NATIONAL LABORATORY

U.S. DEPARTMENT OF ENERGY

UNIVERSITY OF CALIFORNIA

100G Monitoring

Aashish Sharma
Vincent Stoffer

Bro4Pros
February 19th, 2015
OpenDNS, SF

Real-time notification One of our main dissatisfactions with our initial off-line system was the lengthy delay incurred before detecting an attack. If an attack, or an attempted attack, is detected quickly, then it can be much easier to trace back the attacker (for example, by telephoning the site from which they are coming), minimize damage, prevent further break-ins, and initiate full recording of all of the attacker's network activity.

Real-time notification

Mechanism separate from policy

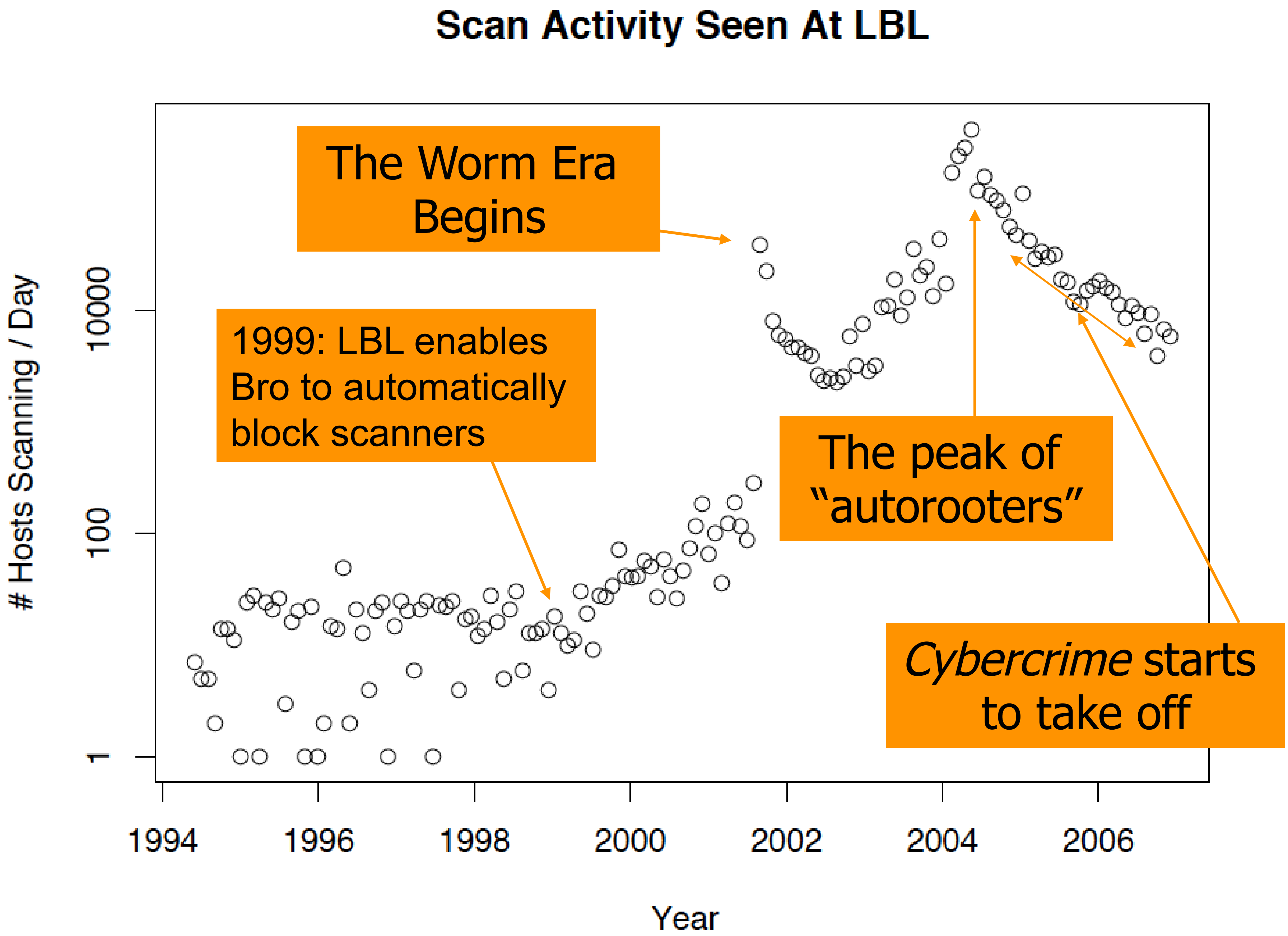
Extensible

Avoid simple mistakes

The monitor will be attacked

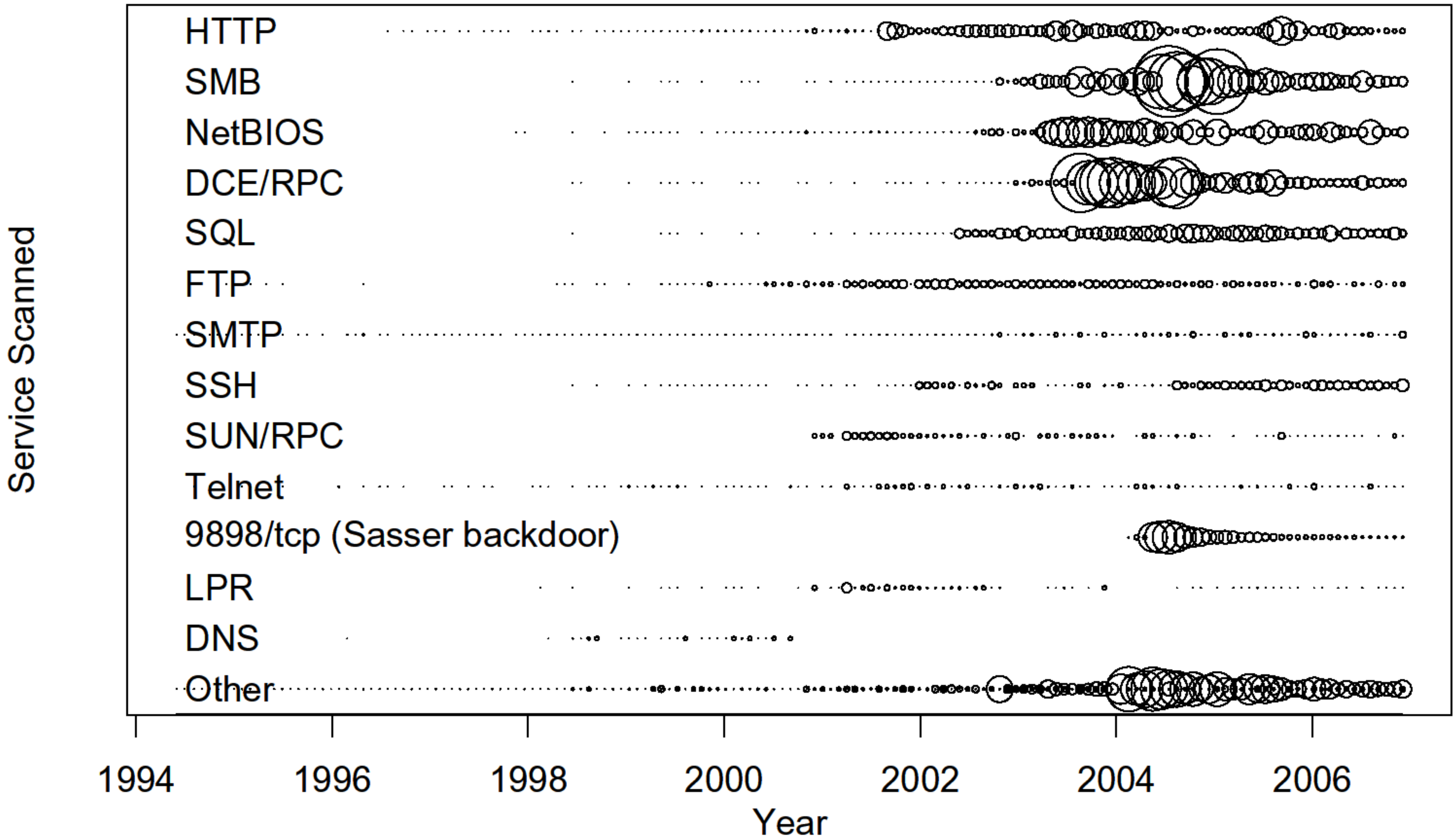
Prior to developing Bro, we had significant operational ex-

```
Institution:
        Lawrence Berkeley Laboratory
        University of California
        Berkeley, CA  94720

Research Title:
        Real-time detection of network intruders

Date: 08 February 1995

        List Price of Digital Products                                    ca. $24,000

...

$ 2,950 DEFTA-DA        Dual-attach FDDI card.  We could instead get the
                        single-attach card, $700 less, DEFTA-AA.  The use
                        I see for dual-attach is a possible outgrowth of
                        the project, which is using the machine as an
                        intelligent "reactive" firewall (one which stops
                        forwarding packets belonging to misbehaving sessions).
```



Scan Activity Seen At LBL
The Worm Era Begins
1999: LBL enables Bro to automatically block scanners
The peak of "autorooters"
Cybercrime starts to take off
Hosts Scanning / Day
1
100
10000
1994
1996
1998
2000
2002
2004
2006
Year

Services Scanned Over Time



A Brief History of Scanning

2007

Real-time notification One of our main dissatisfactions with our initial off-line system was the lengthy delay incurred before detecting an attack. If an attack, or an attempted attack, is detected quickly, then it can be much easier to trace back the attacker (for example, by telephoning the site from which they are coming), minimize damage, prevent further break-ins, and initiate full recording of all of the attacker's network activity.

Real-time notification

This is not to discount the enormous utility of keeping extensive, permanent logs of network activity for later analysis. Invariably, when we have suffered a break-in, we turn to these logs for retrospective damage assessment, sometimes searching back a number of months.

Building a Time Machine for Efficient Recording and Retrieval of High-Volume Network Traffic

Enriching Network Security Analysis with Time Travel

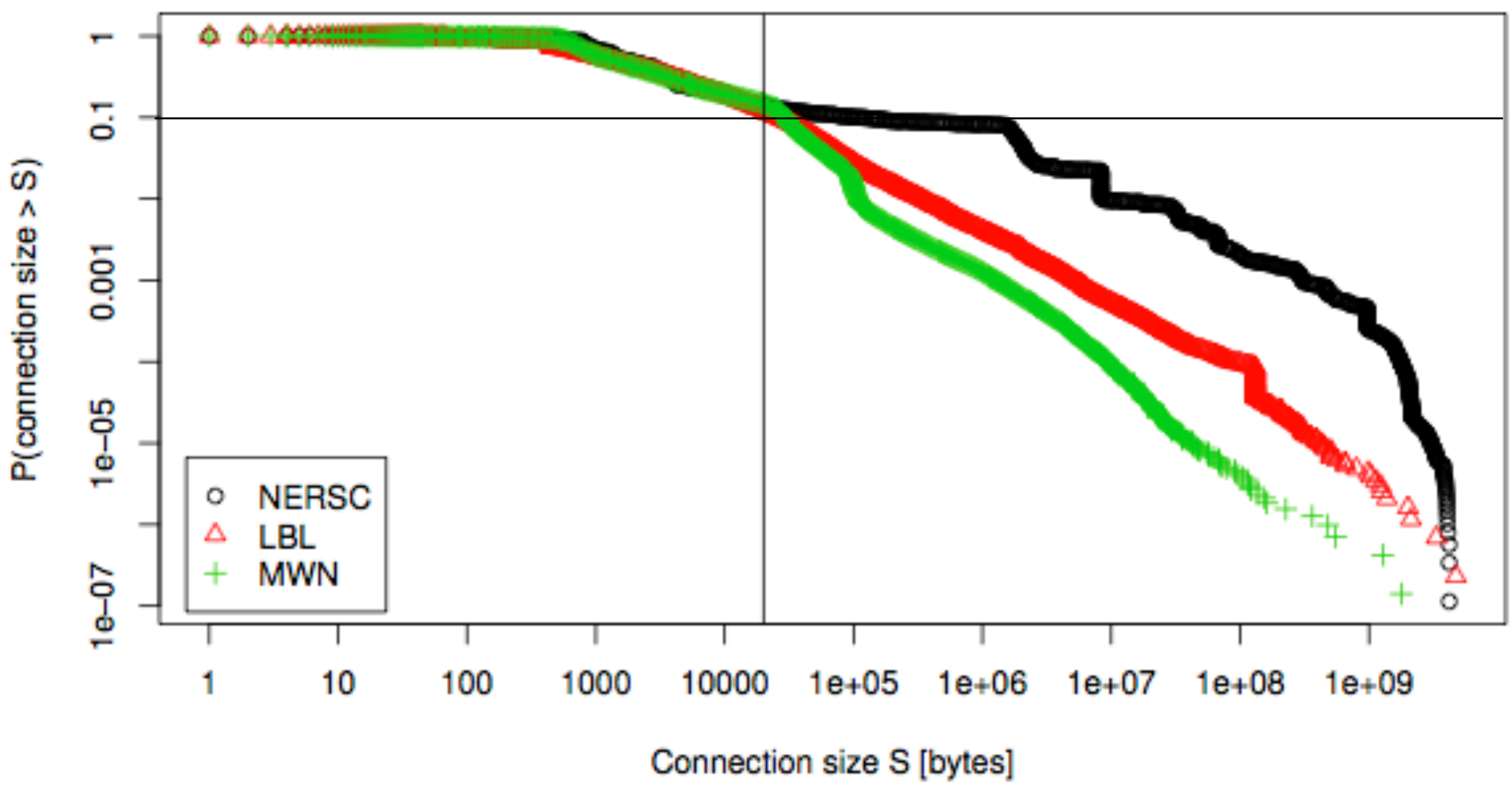


Figure 1: Log-log CCDF of connection sizes

2005/
2008

Mechanism separate from policy Sound software design often stresses constructing a clear separation between mechanism and policy; done properly, this buys both simplicity and flexibility. The problems faced by our

Extensible Because there are an enormous number of different network attacks, with who knows how many waiting to be discovered, the system clearly must be designed in order to make it easy to add to it knowledge of new types of attacks. In addition, while our system

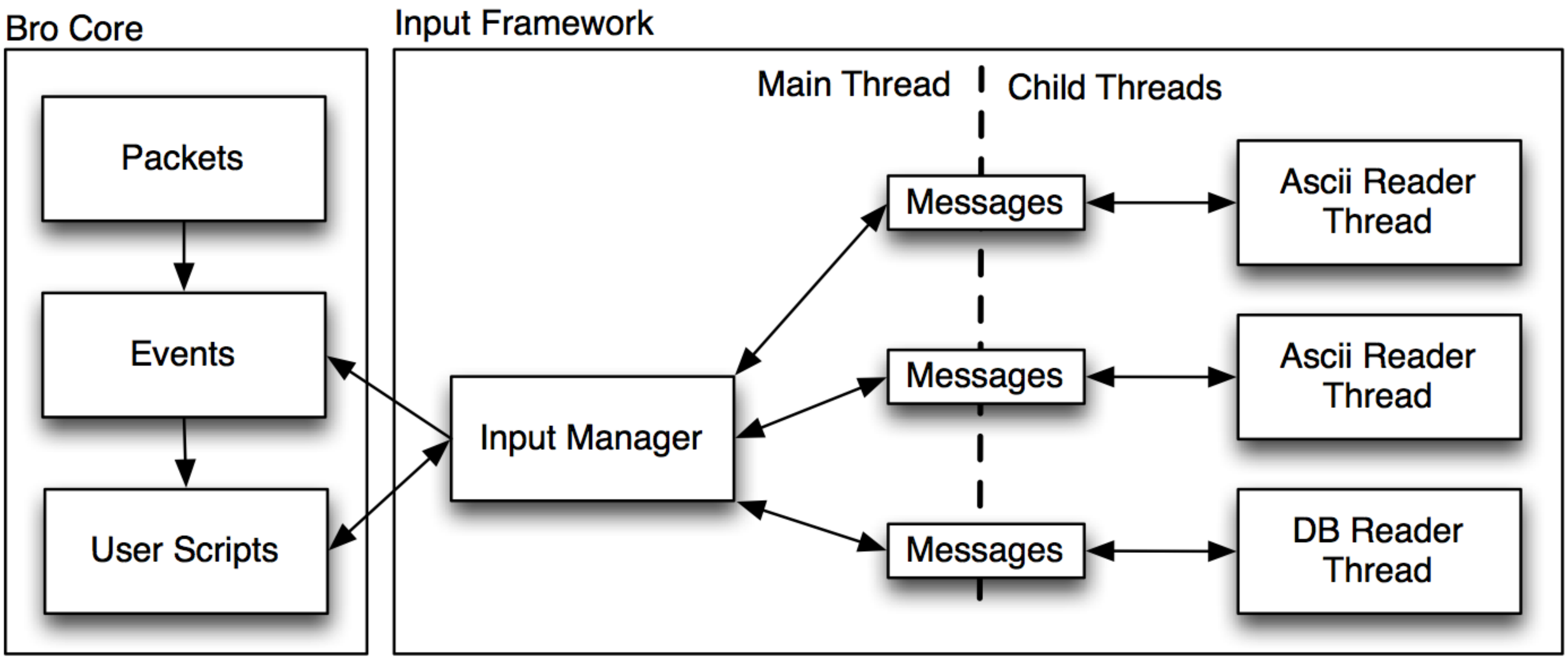
Mechanism separate from policy

Extensible

Avoid simple mistakes

The monitor will be attacked

A Lone Wolf No More: Supporting Network Intrusion Detection with Real-Time Intelligence



2012

Through the Eye of the PLC: Semantic Security Monitoring for Industrial Processes

Table 1: Summary of plausible attacks against PLC implementations: Modbus example

Level	Impact		Attack description	Example
1	Data integrity		Corrupt integrity by adding data to the packet.	Craft a packet that has a different length than defined in parameters or in spec [2].
2	IT System	Reconnaissance	Analyse functionality a PLC implements.	Probe FC, listen for responses and exceptions [2].
		Integrity	Exploit lack of specification compliance.	Manipulate application parameters within spec (e.g., offset) or outside of spec (e.g., illegal FC) [2, 9, 37].
			Perform unauthorized use of an administrative command.	Use FC 8-0A to clear counters and diagnostics audit [2].
		Denial of service	Perform MITM to enforce system delay.	Send exception codes 05, 06 or FC 8-04 to enforce Listen mode [2].
			Perform unauthorized use of administrative command.	Use FC 8-01 to restart TCP communication [2, 9].
3	Process	Reconnaissance	Analyse structure of memory map.	Probe readable/writable points. Exceptions tell process implementation details [2].
		Direct control	Perform change on process variable.	Write inverted or min/max values [10]. Modify key setpoint variables [14, 26].
		Indirect control	Tamper with process values.	Replay values [14].

FC: Function code defining the type of functionality in Modbus. *MITM: Man-in-the-middle attack.*

2014

Rapid and Scalable ISP Service Delivery through a Programmable MiddleBox

MAdFraud: Investigating Ad Fraud in Android Applications

Pitfalls in HTTP Traffic Measurements and Analysis

Investigating IPv6 Traffic

What happened at the World IPv6 Day?

Exploring EDNS-Client-Subnet Adopters in your Free Time*

On Modern DNS Behavior and Properties

Enabling Content-aware Traffic Engineering

Pushing CDN-ISP Collaboration to the Limit

100+ more at https://www.bro.org/research/index.html

Prior to developing Bro, we had significant operational experience with a simpler system based on off-line analysis of

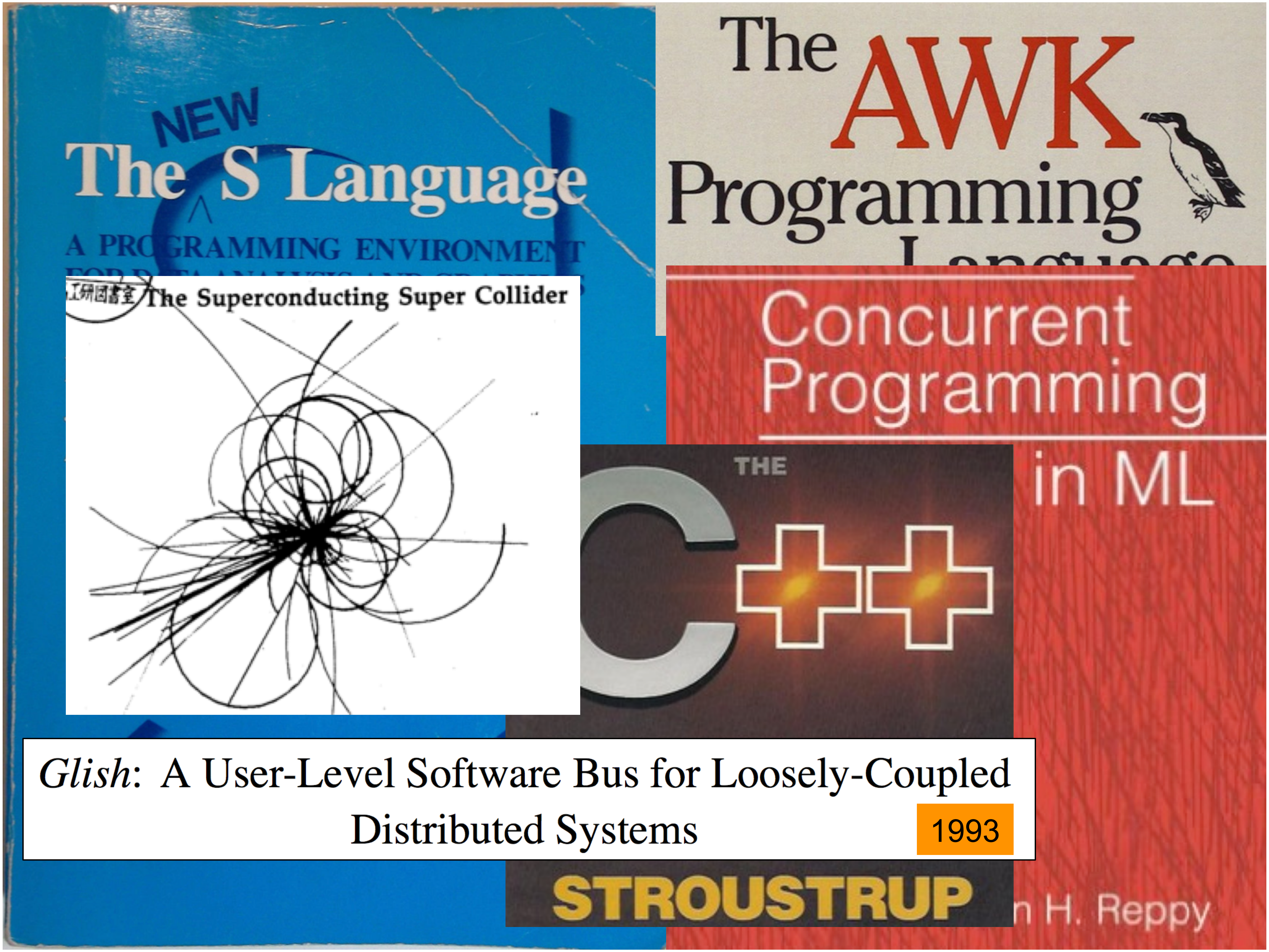
Avoid simple mistakes Of course, we always want to avoid mistakes. However, here we mean that we particularly desire that the way that a site defines its security policy be both clear and as error-free as possible. (For example, we would not consider expressing the policy in C code as meeting these goals.)

Mechanism separate from policy

Extensible

Avoid simple mistakes

The monitor will be attacked


NEW
The S Language
A PROGRAMMING ENVIRONMENT
The Superconducting Super Collider
The AWK Programming
Concurrent Programming
in ML
THE
C++
STROUSTRUP
n H. Reppy
Glish: A User-Level Software Bus for Loosely-Coupled Distributed Systems
1993

2002: &attributes – state management, persistence, defaults, file rotation, logging

2002: IPv6 (due to ESnet need)

2002/2004: ALERT's / NOTICE's

2002: Signature engine

2003: modules

2005: BinPAC – DSL for protocol analyzers

2006: "when" statement

2009: BroControl

2012: "hook" construct

Frameworks

- File Analysis
- GeoLocation
- Input Framework
- Intelligence Framework
- Logging Framework
- Notice Framework
- Signature Framework
- Summary Statistics
- Broker-Enabled Communication Framework

TABLE OF CONTENTS

NEXT PAGE

File Analysis

PREVIOUS PAGE

Writing Bro Scripts

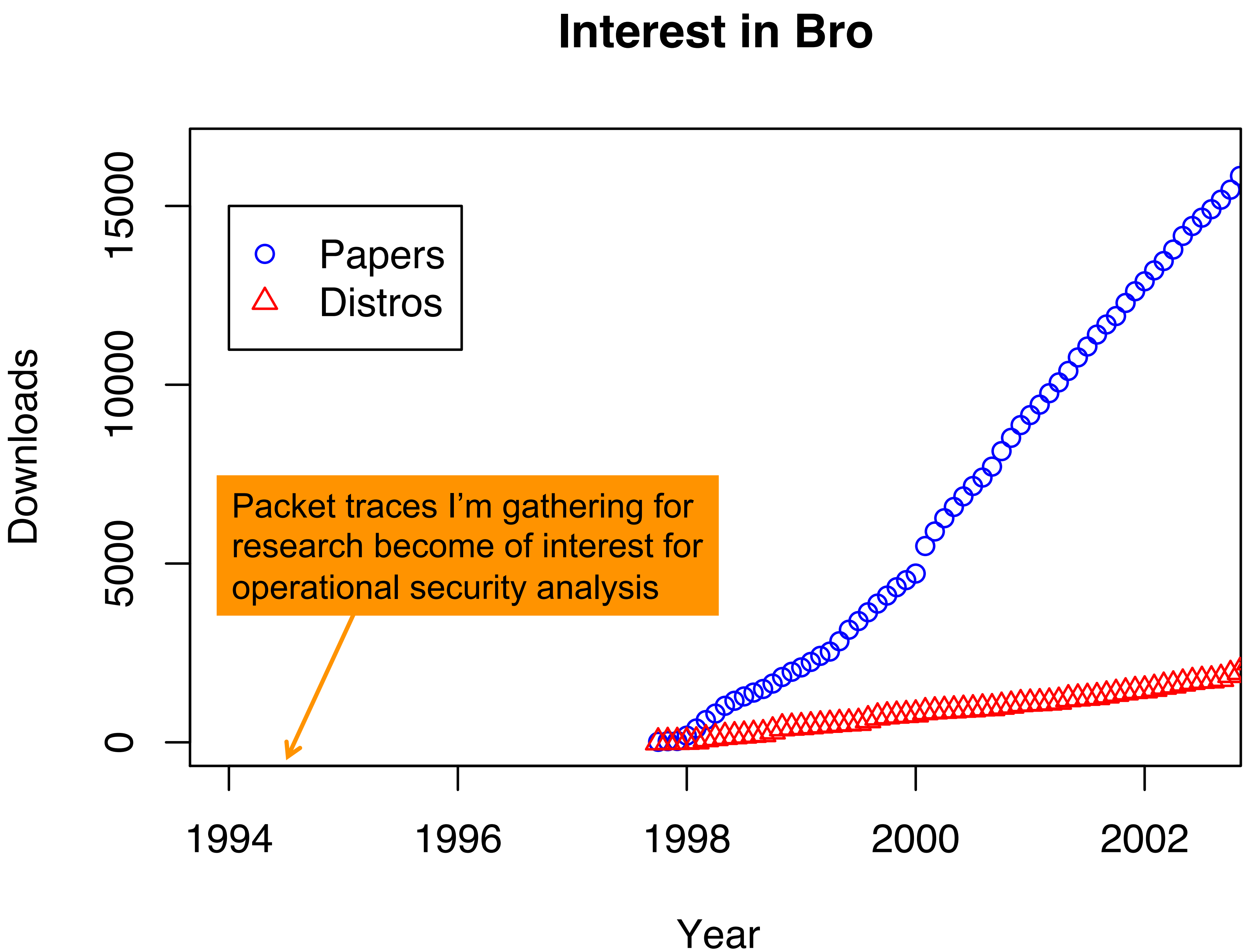
Prior to developing Bro, we had significant operational experience with a simpler system based on off-line analysis of `tcpdump` [JLM89] trace files. Out of this experience we formulated a number of design goals and requirements:

The monitor will be attacked We must assume that attackers will (eventually) have full knowledge of the techniques used by the monitor, and access to its source code, and will use this knowledge in attempts to subvert or overwhelm the monitor so that it fails to detect the attacker's break-in activity. This assumption significantly complicates the design of the monitor, but failing to address it is to build a house of cards.

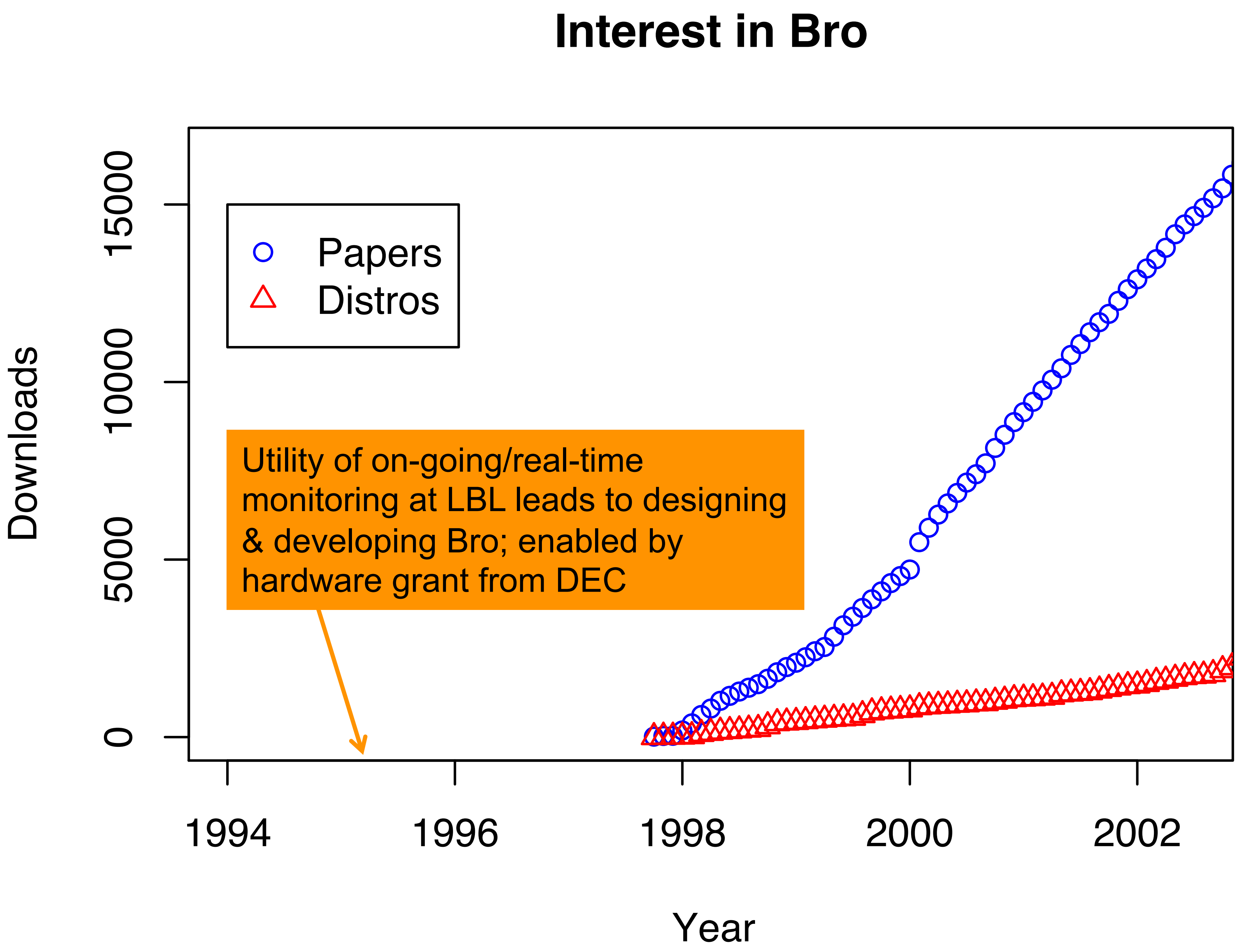
Avoid simple mistakes

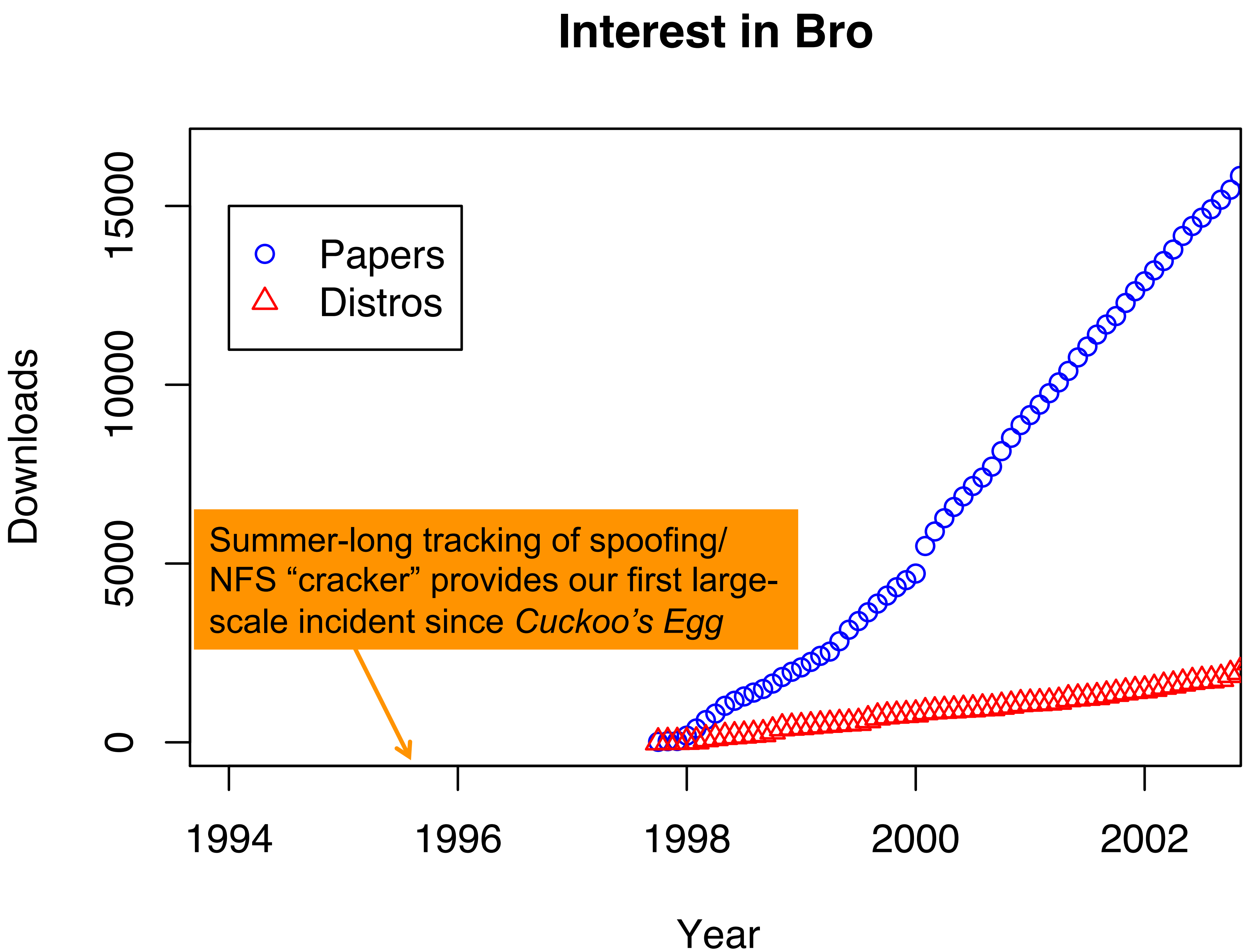
The monitor will be attacked

Part II: Project Evolution

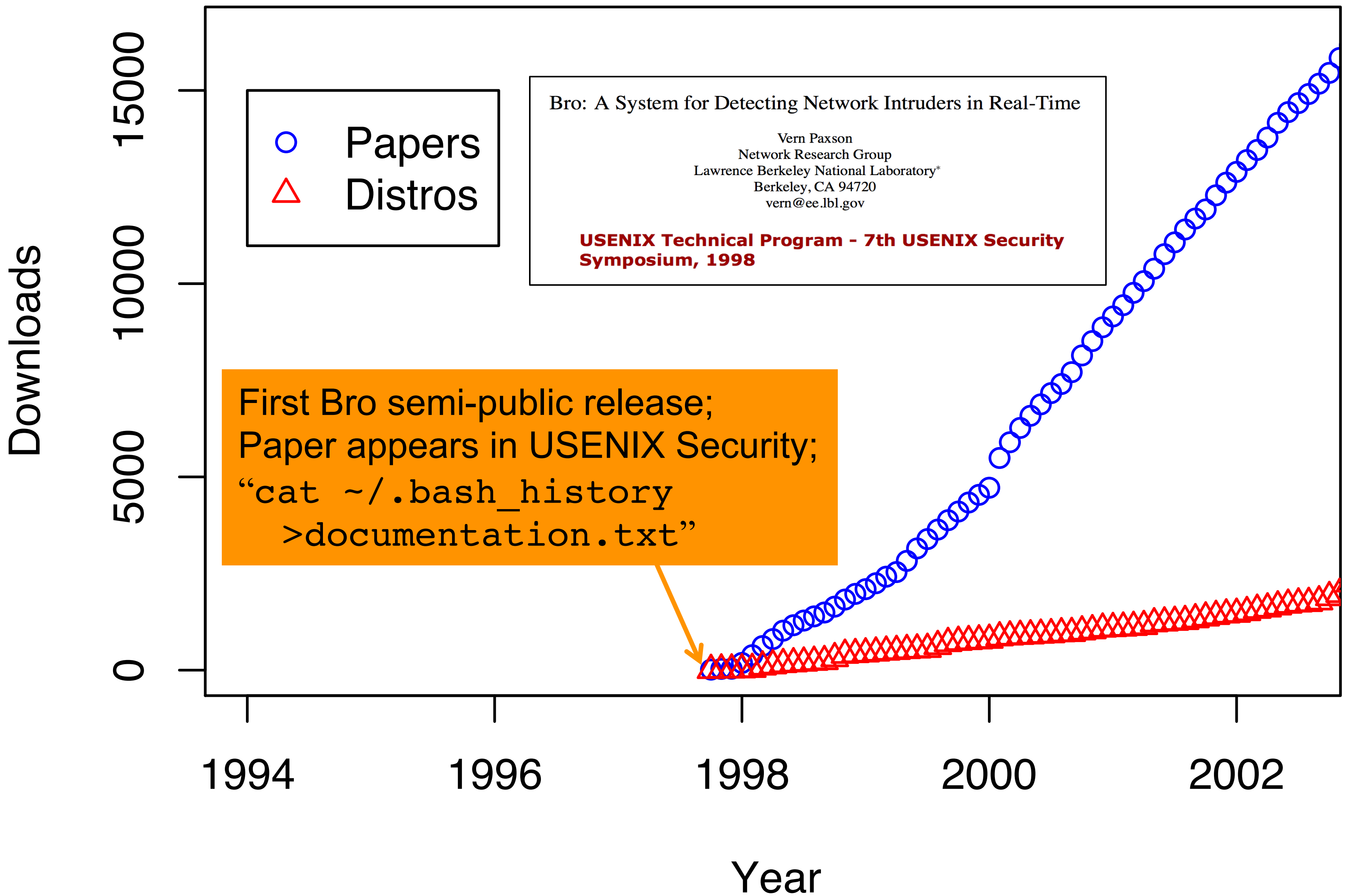


Interest in Bro
Papers
Distros
Packet traces I'm gathering for research become of interest for operational security analysis
Downloads
0
5000
10000
15000
1994
1996
1998
2000
2002
Year

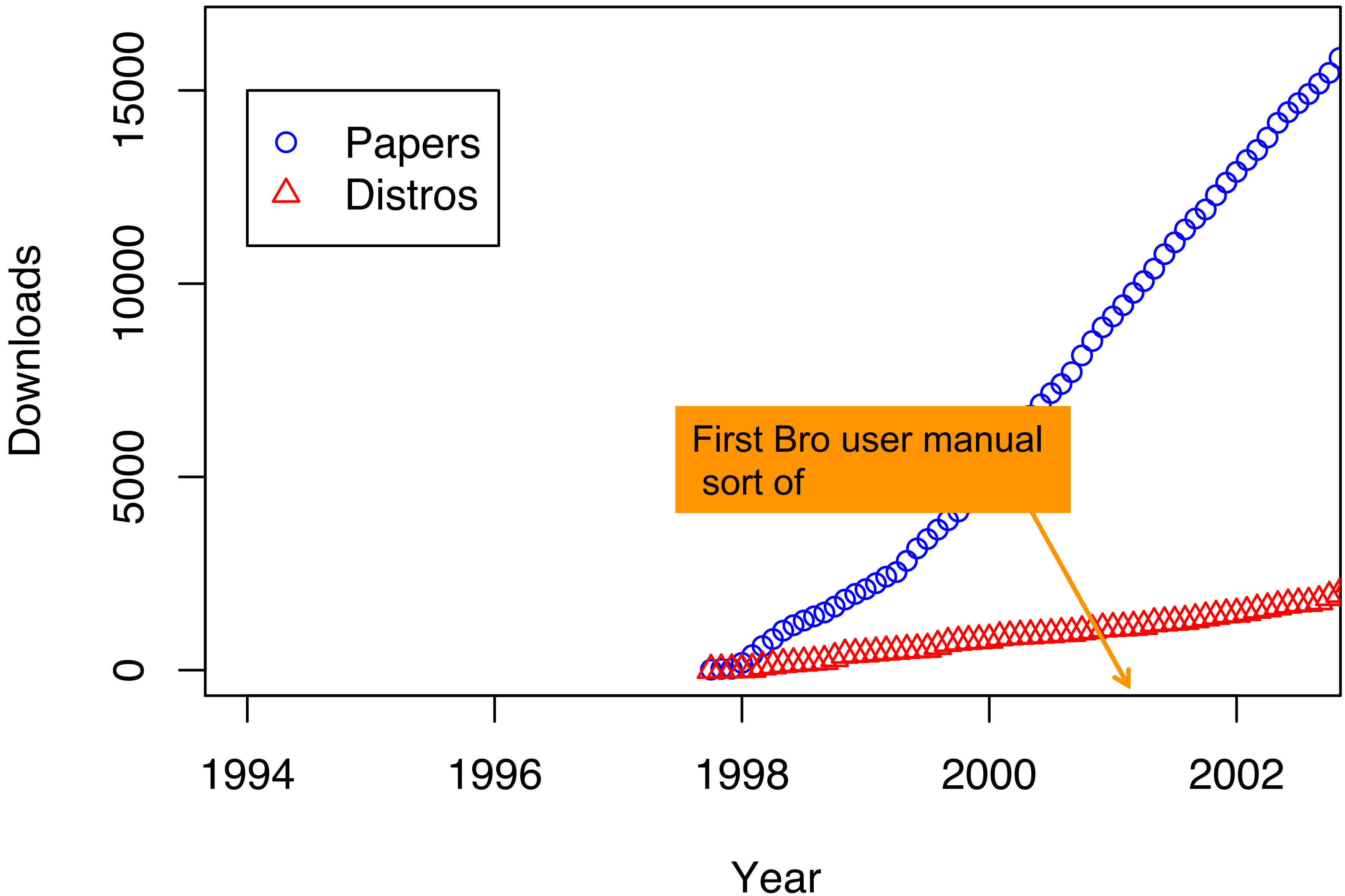

Interest in Bro
Papers
Distros
Utility of on-going/real-time monitoring at LBL leads to designing & developing Bro; enabled by hardware grant from DEC
Downloads
0
5000
10000
15000
1994
1996
1998
2000
2002
Year

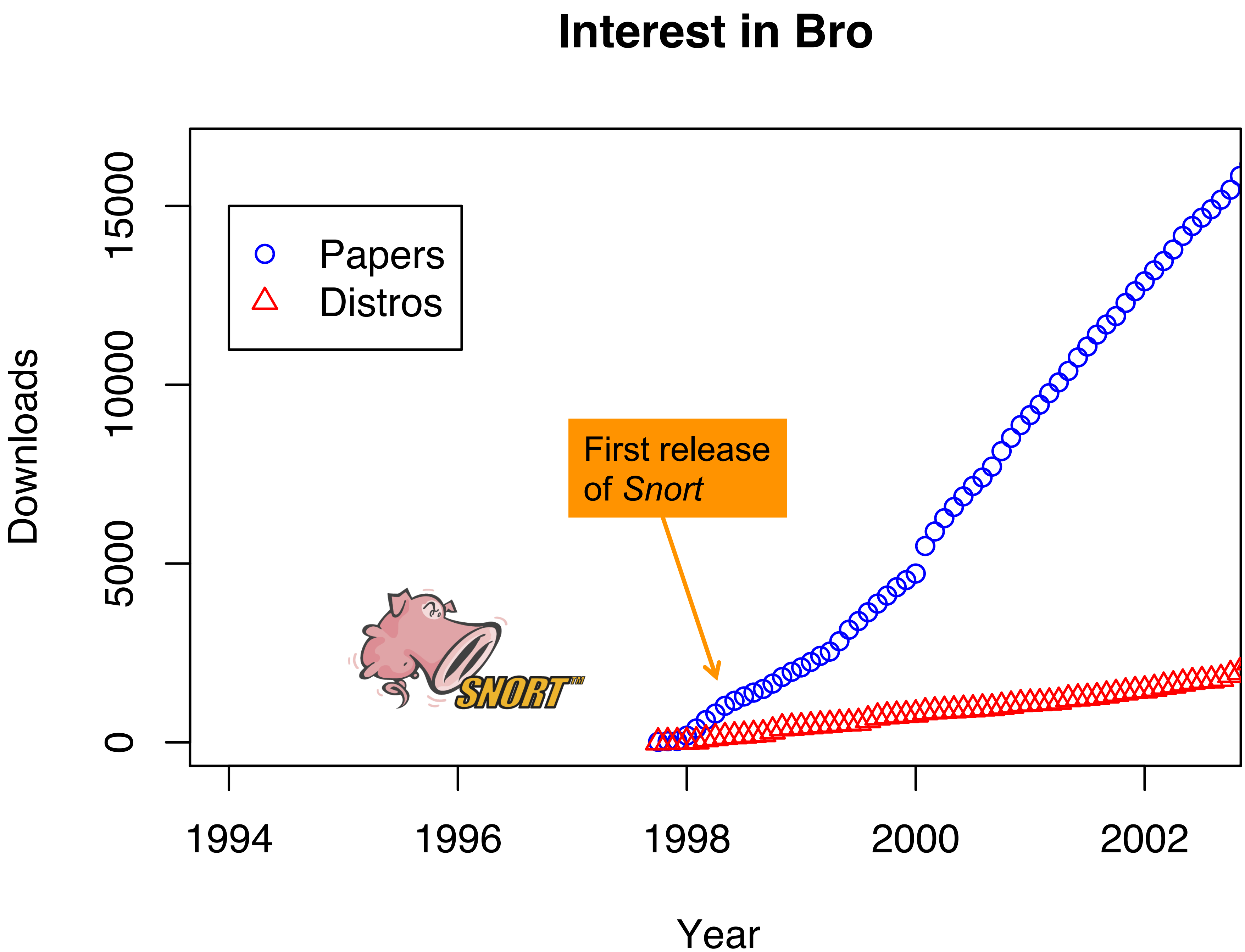

Interest in Bro
Papers
Distros
Downloads
0
5000
10000
15000
Year
1994
1996
1998
2000
2002
Summer-long tracking of spoofing/ NFS “cracker” provides our first large-scale incident since *Cuckoo’s Egg*

Interest in Bro



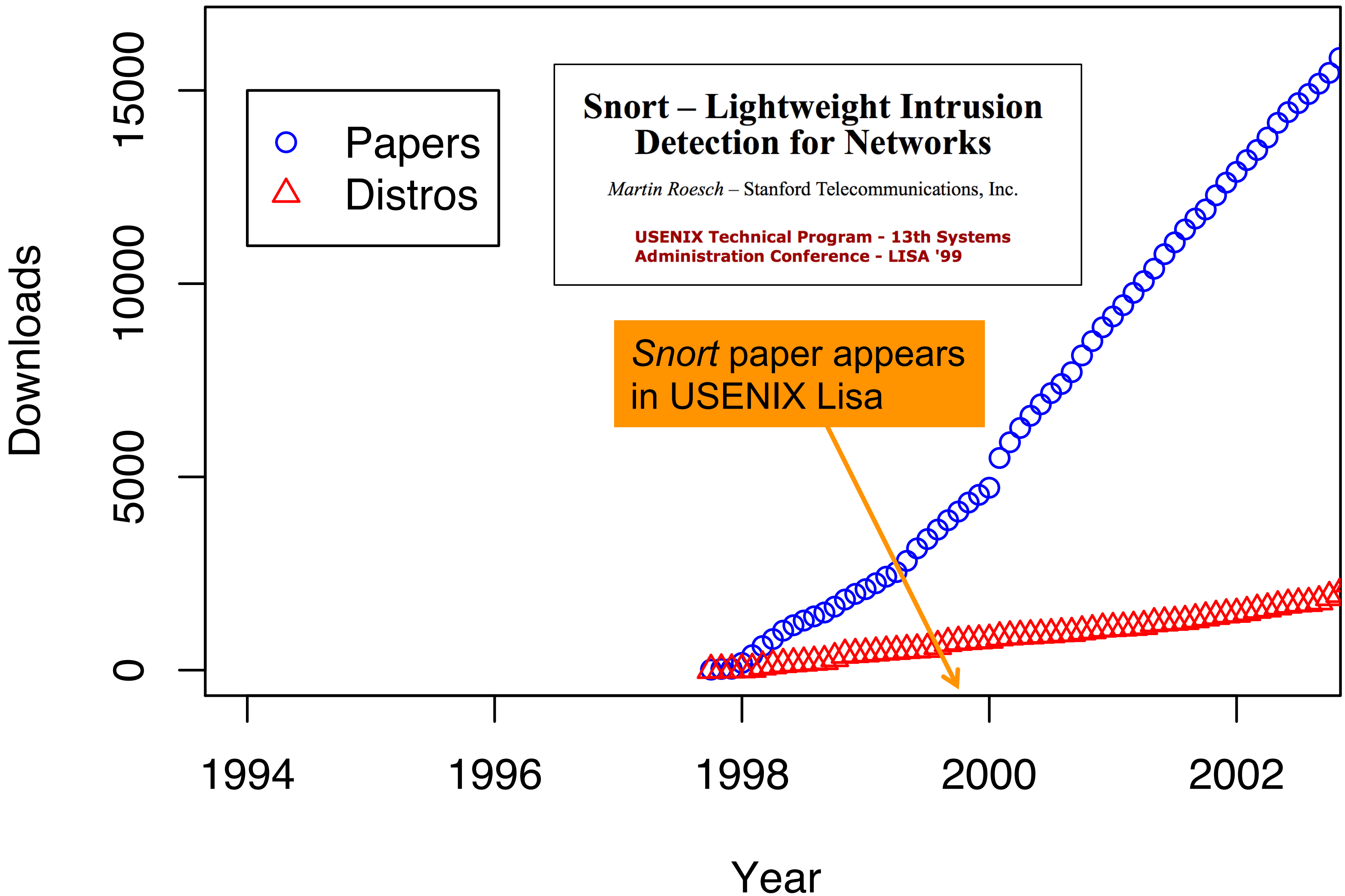
Interest in Bro

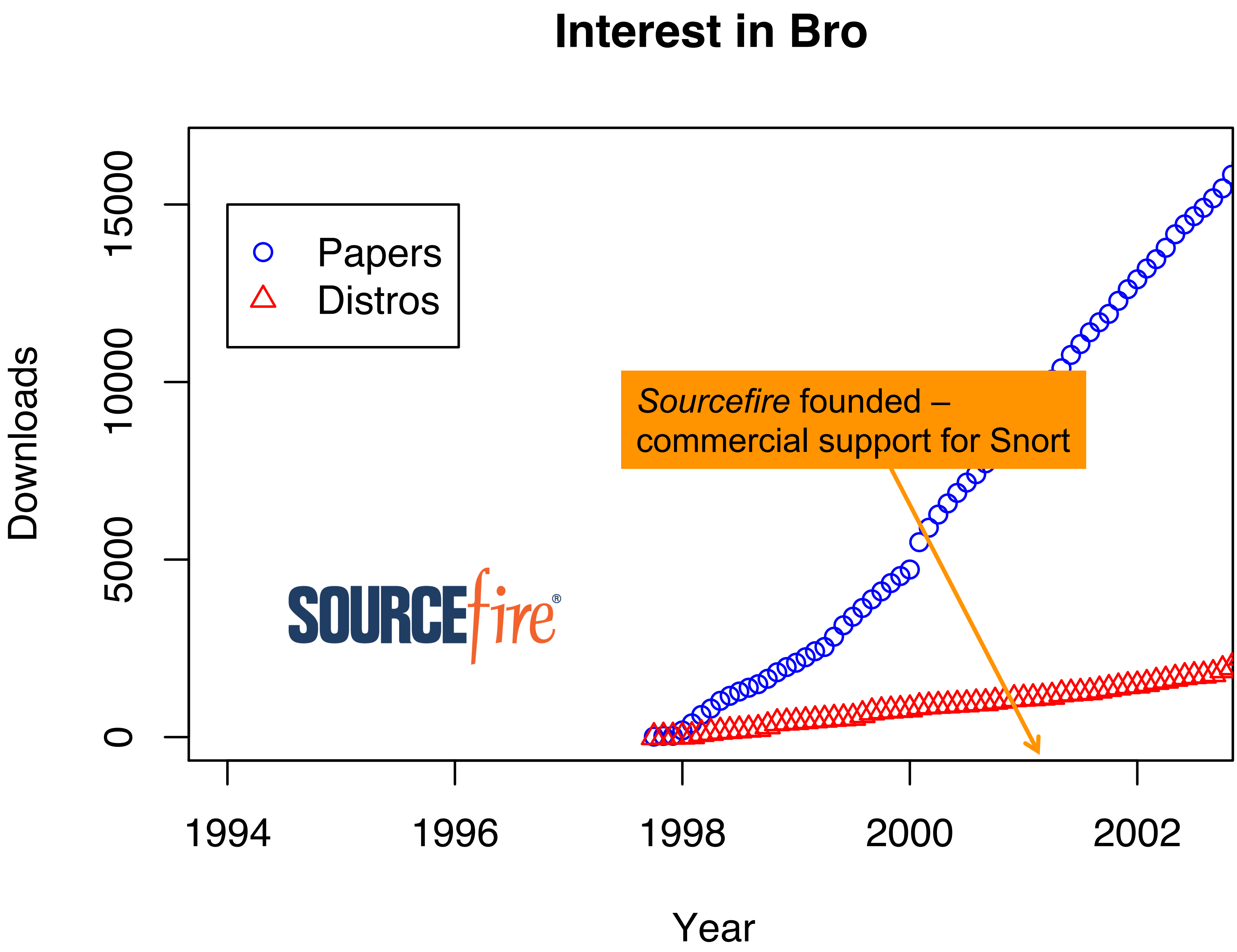




Interest in Bro
Papers
Distros
First release of *Snort*
SNORT™
Downloads
0
5000
10000
15000
1994
1996
1998
2000
2002
Year

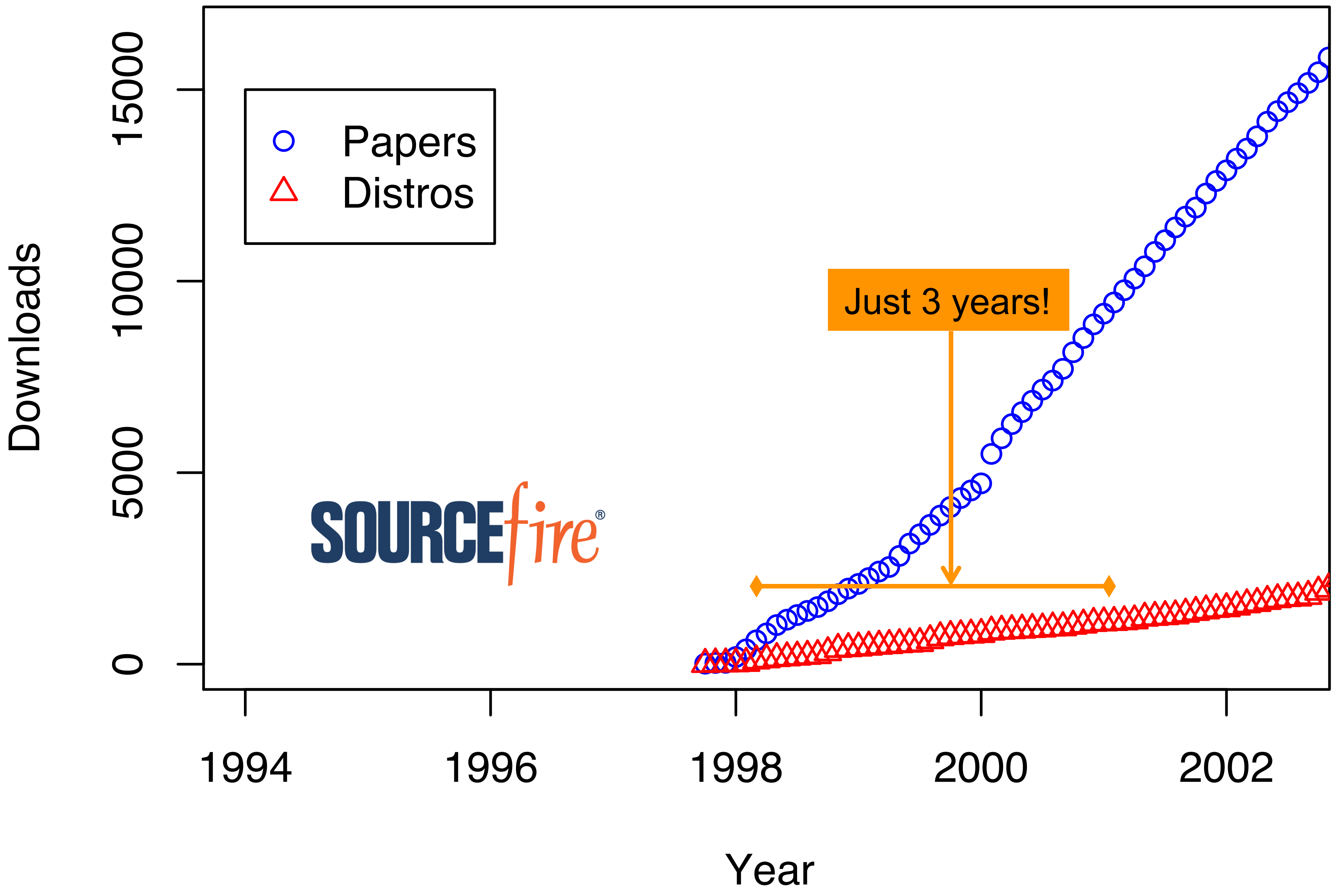
Interest in Bro



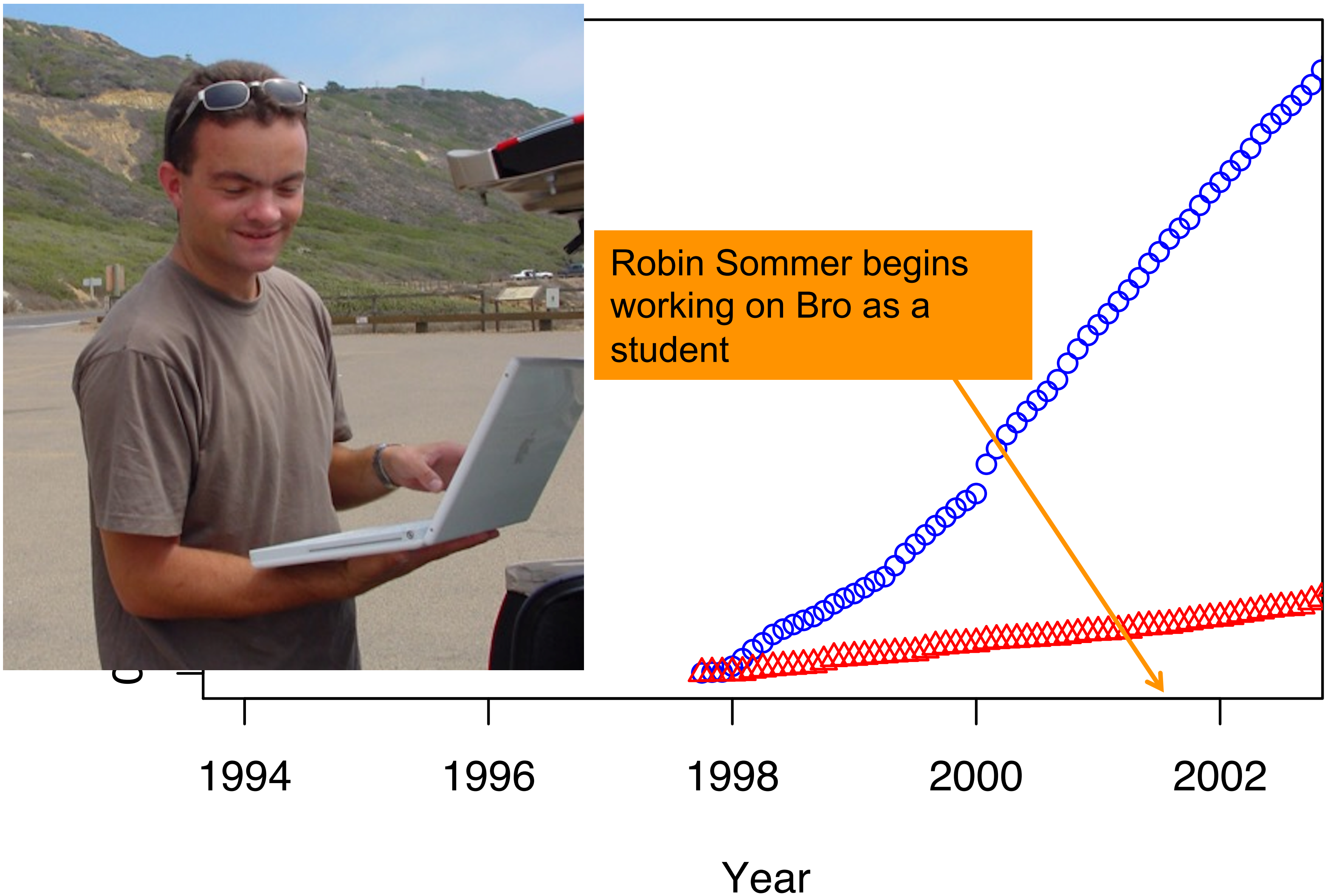


Interest in Bro
Papers
Distros
Sourcefire founded –
commercial support for Snort
SOURCEfire
Downloads
0
5000
10000
15000
1994
1996
1998
2000
2002
Year

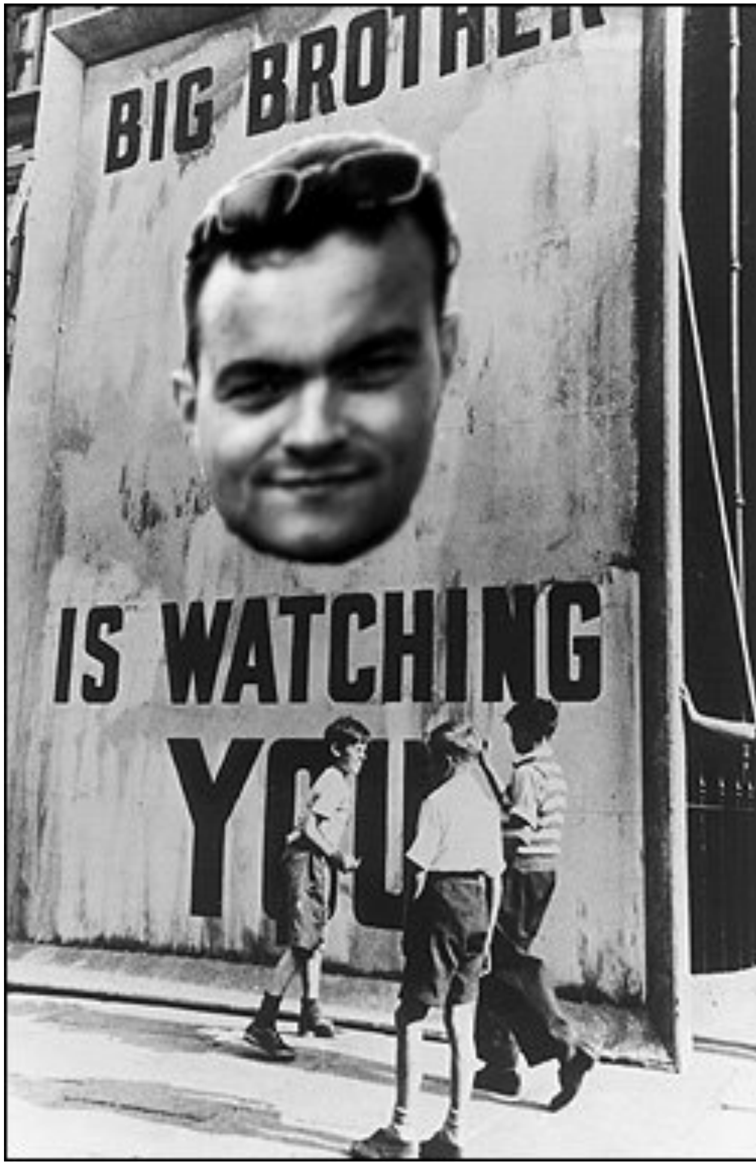
Interest in Bro

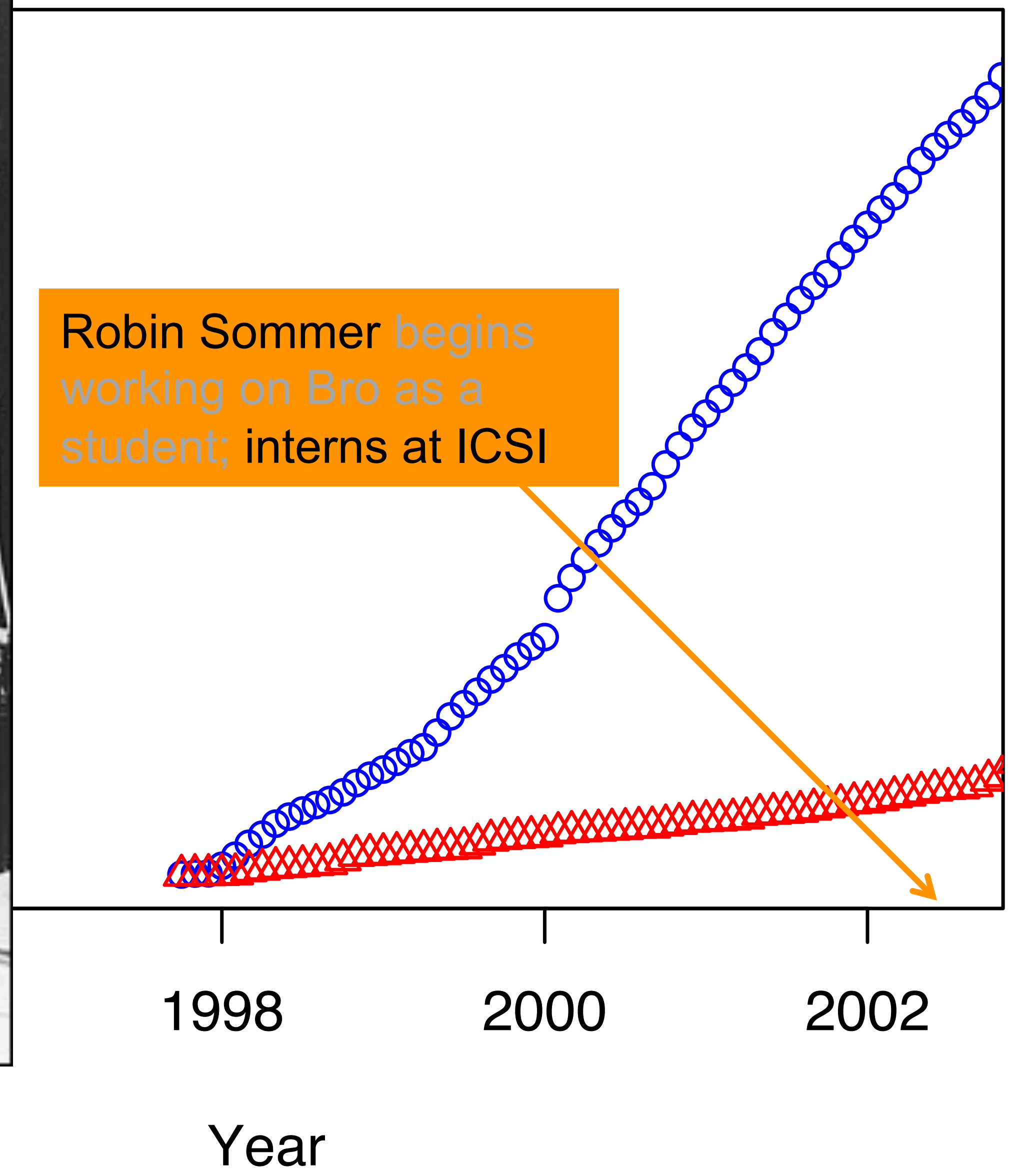


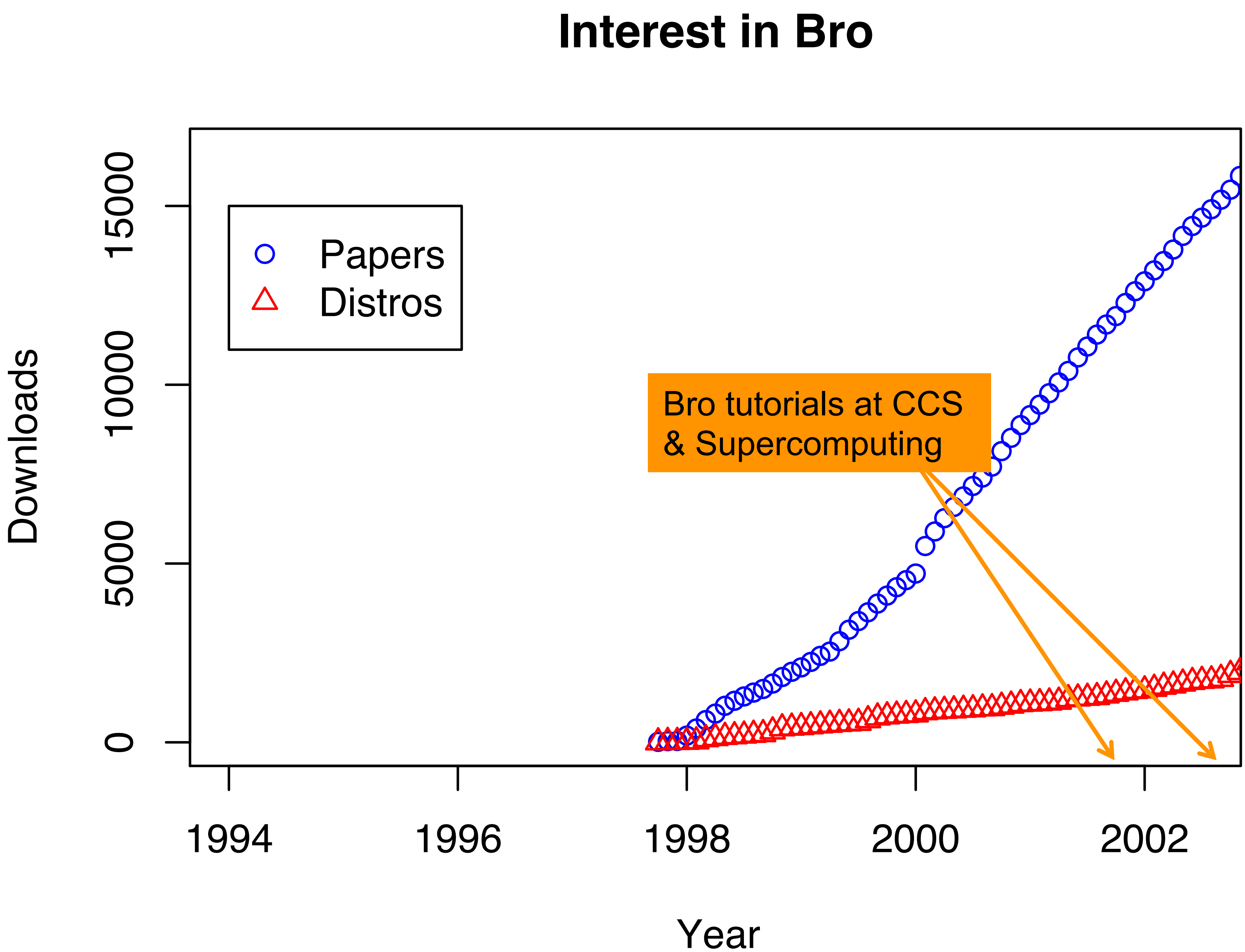
Interest in Bro



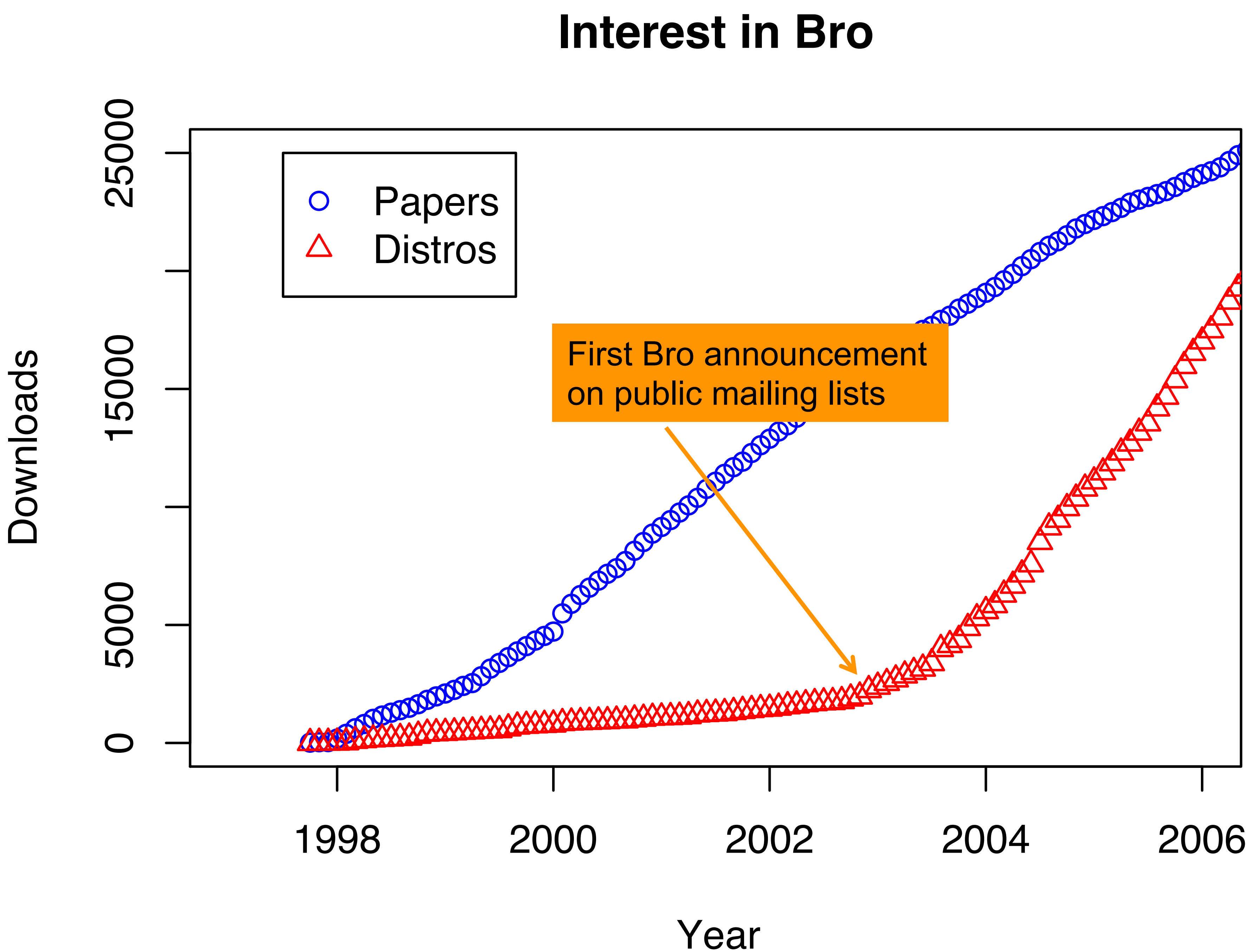
Interest in Bro



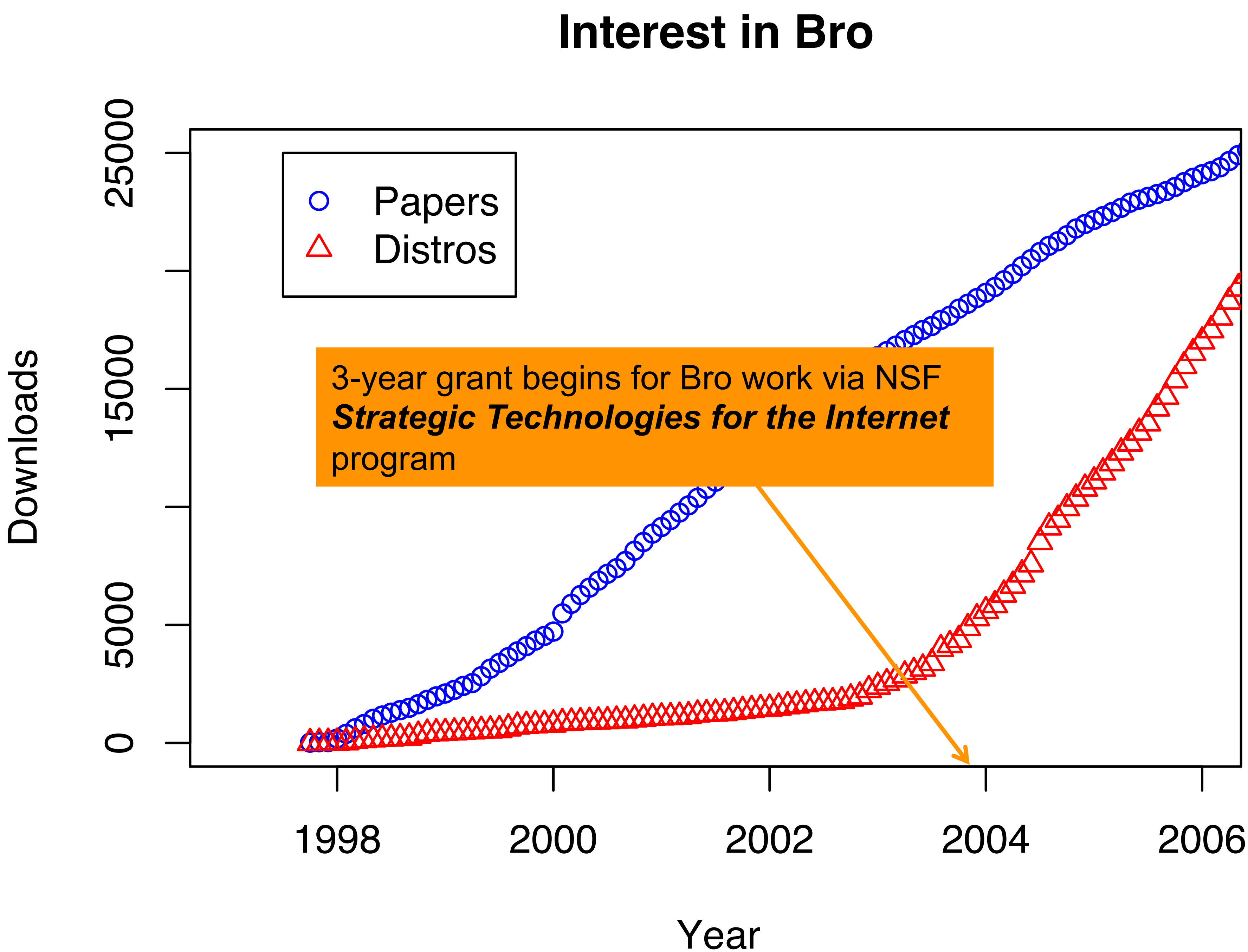




Interest in Bro
Papers
Distros
Bro tutorials at CCS
& Supercomputing
Downloads
0
5000
10000
15000
1994
1996
1998
2000
2002
Year



Interest in Bro
Papers
Distros
First Bro announcement
on public mailing lists
Downloads
0
5000
15000
25000
1998
2000
2002
2004
2006
Year



Interest in Bro
Papers
Distros
3-year grant begins for Bro work via NSF
Strategic Technologies for the Internet
program
Downloads
0
5000
15000
25000
1998
2000
2002
2004
2006
Year

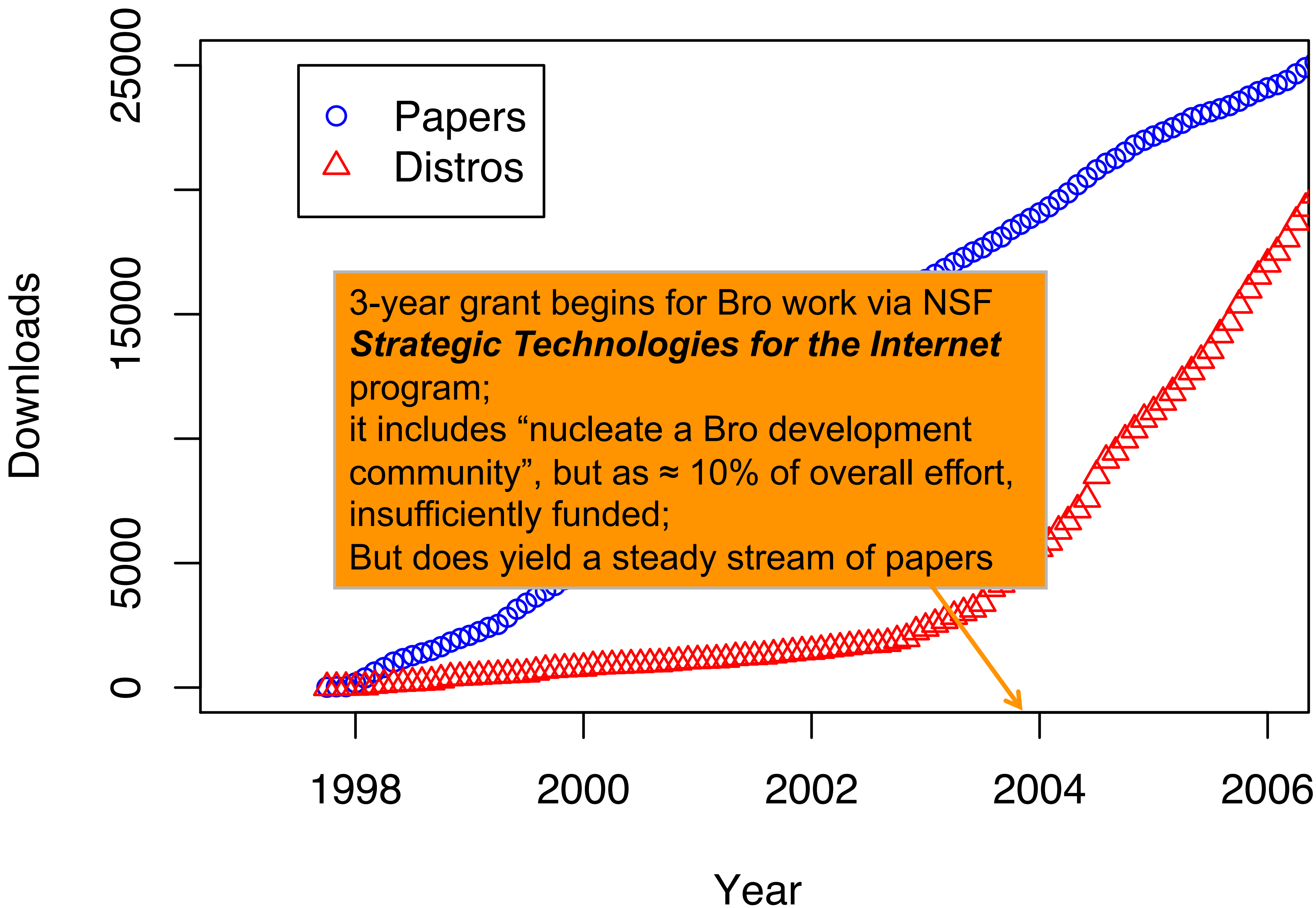
National Science Foundation
WHERE DISCOVERIES BEGIN

Award Abstract #0334088

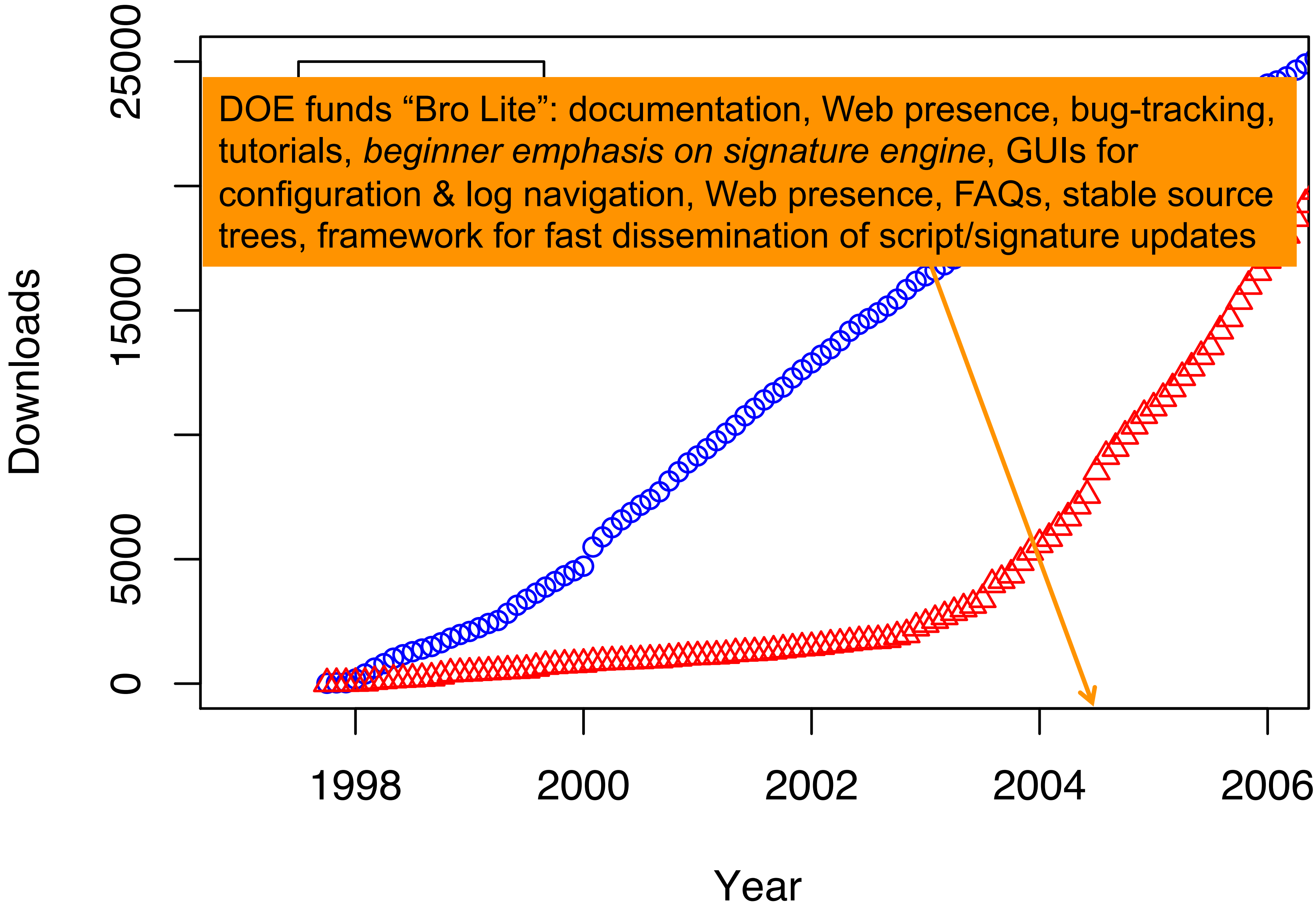
STI: Viable Network Defense for Scientific Research Institutions

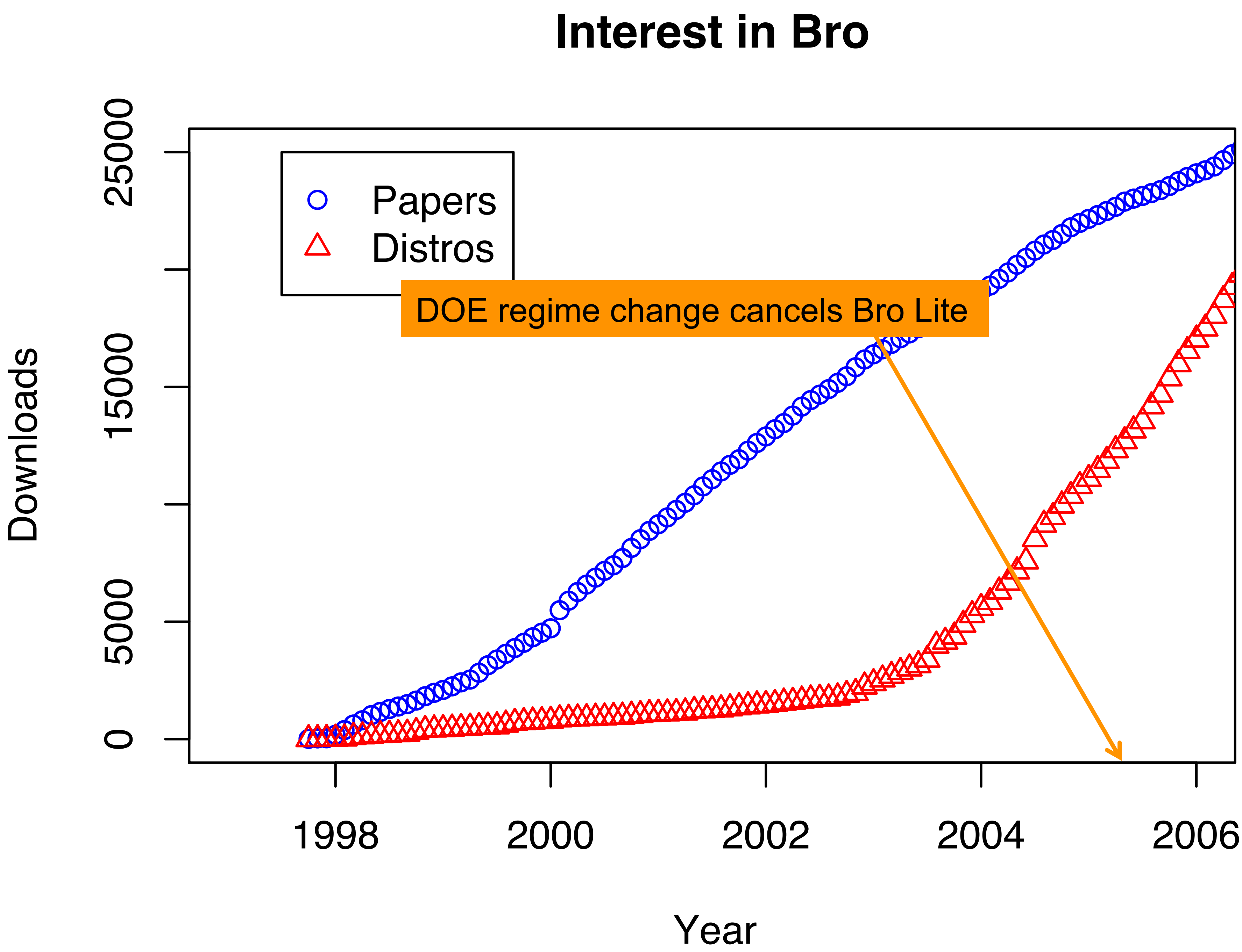
NSF Org:	ACI Div Of Advanced Cyberinfrastructure
Program Manager:	Kevin L. Thompson ACI Div Of Advanced Cyberinfrastructure CSE Direct For Computer & Info Scie & Enginr
Start Date:	November 1, 2003
$1,629,392 ? **End Date:**	October 31, 2007 (Estimated)
Awarded Amount to Date:	$900,000.00
Investigator(s):	Vern Paxson vern@icsi.berkeley.edu (Principal Investigator)

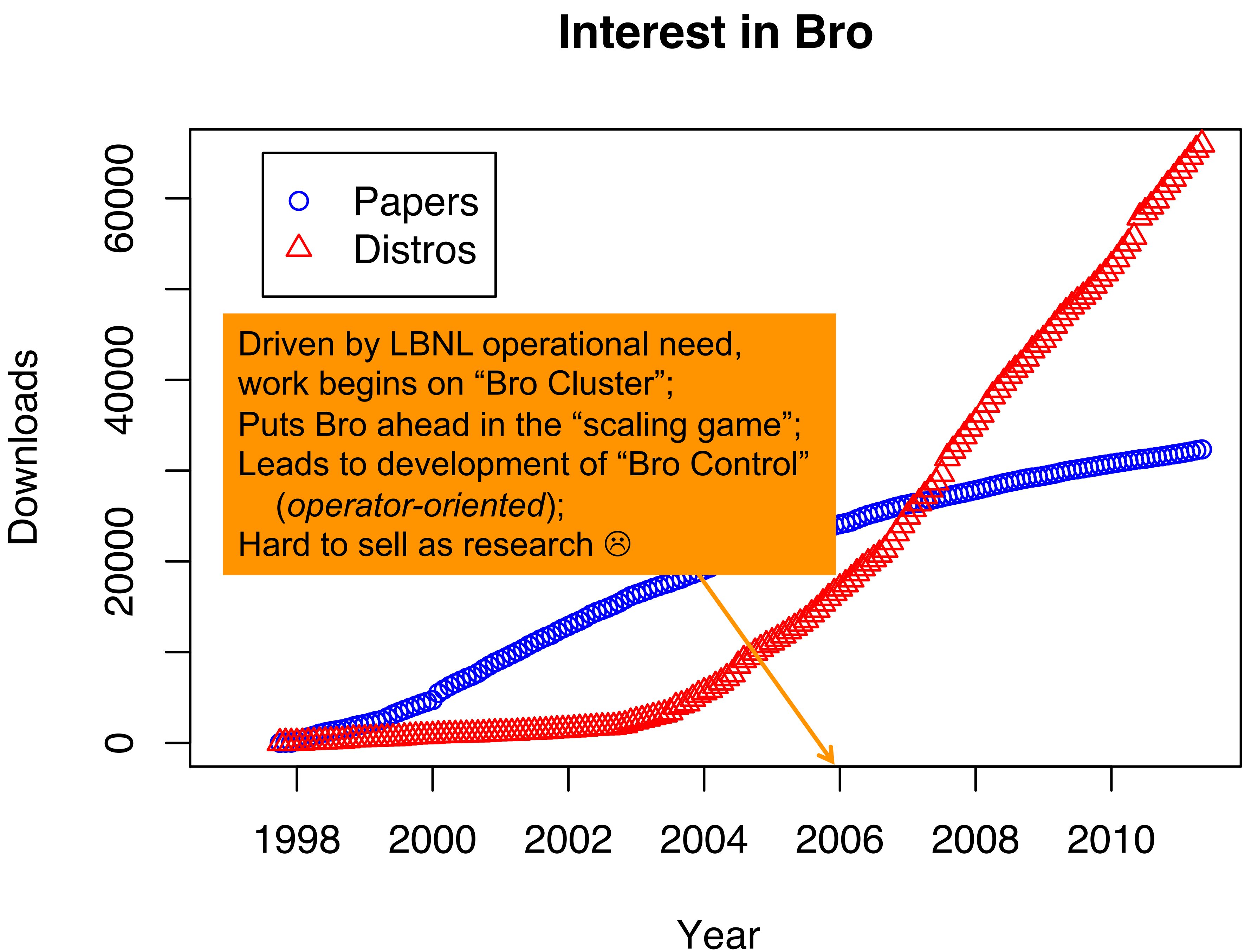
Interest in Bro



Interest in Bro

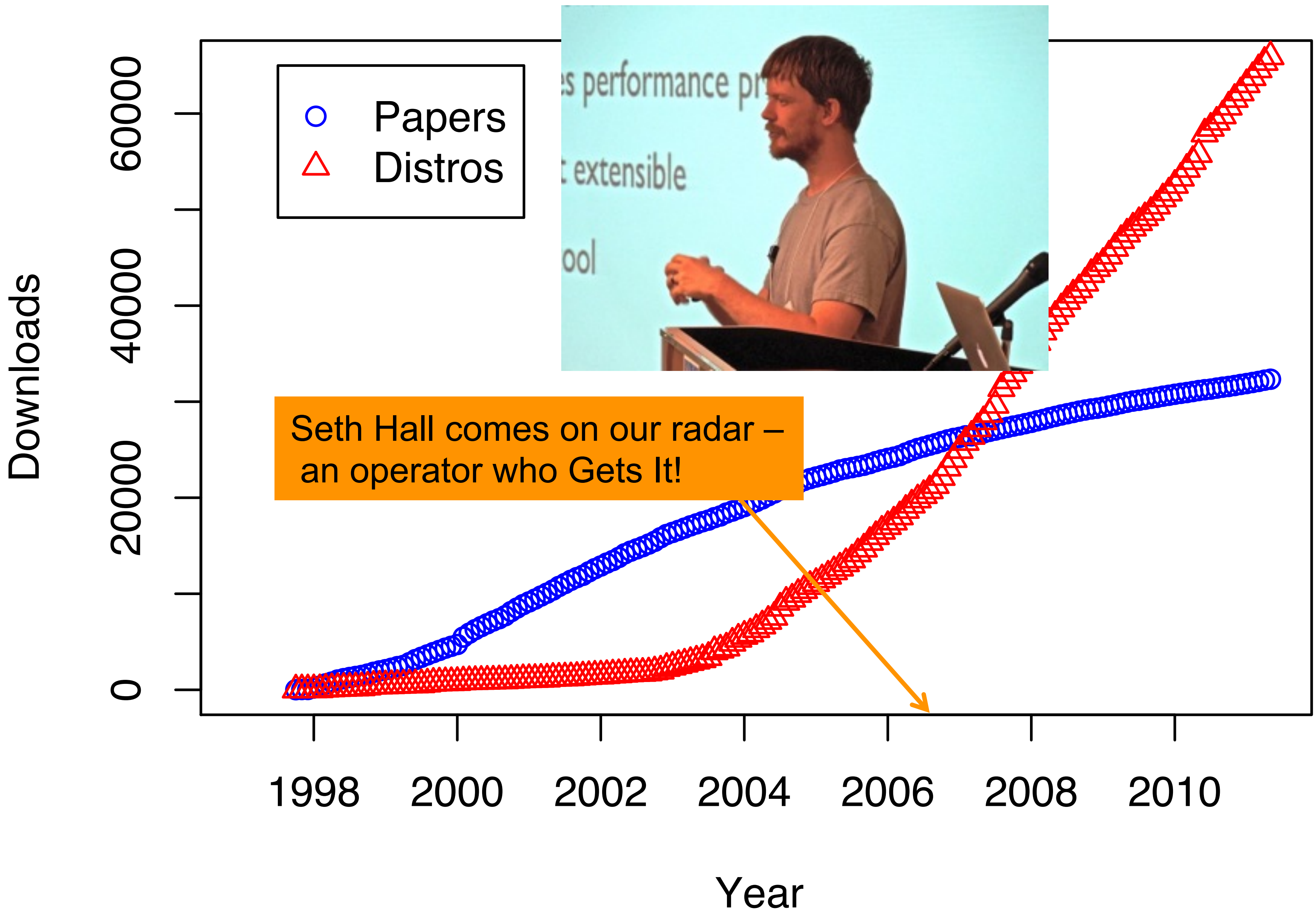


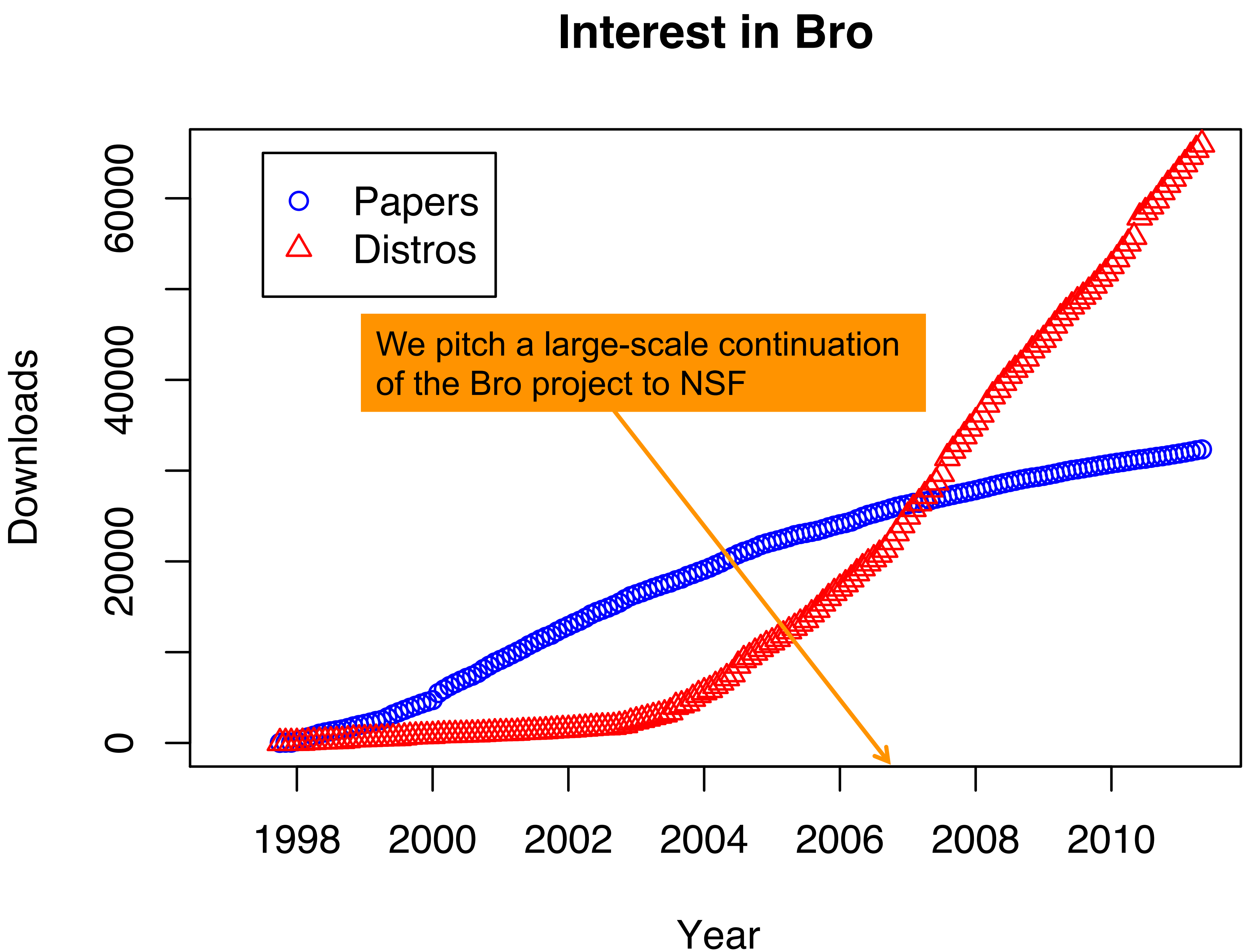

Interest in Bro
Papers
Distros
DOE regime change cancels Bro Lite
Downloads
0
5000
15000
25000
1998
2000
2002
2004
2006
Year



Interest in Bro
Papers
Distros
Driven by LBNL operational need,
work begins on “Bro Cluster”;
Puts Bro ahead in the “scaling game”;
Leads to development of “Bro Control”
(operator-oriented);
Hard to sell as research ☹
Downloads
0
20000
40000
60000
1998
2000
2002
2004
2006
2008
2010
Year

Interest in Bro





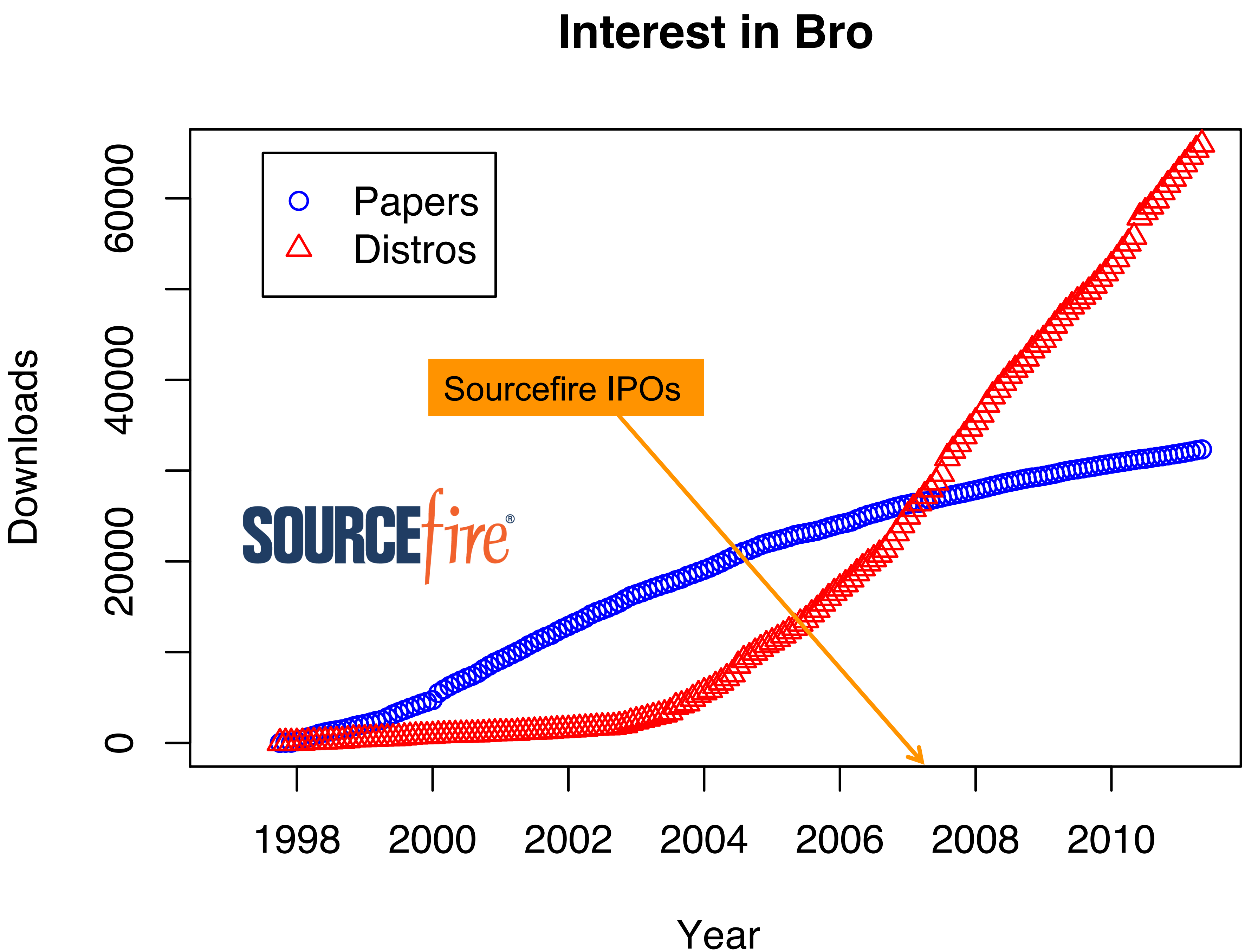
Interest in Bro
Papers
Distros
We pitch a large-scale continuation of the Bro project to NSF
Downloads
0
20000
40000
60000
1998
2000
2002
2004
2006
2008
2010
Year

National Science Foundation
WHERE DISCOVERIES BEGIN

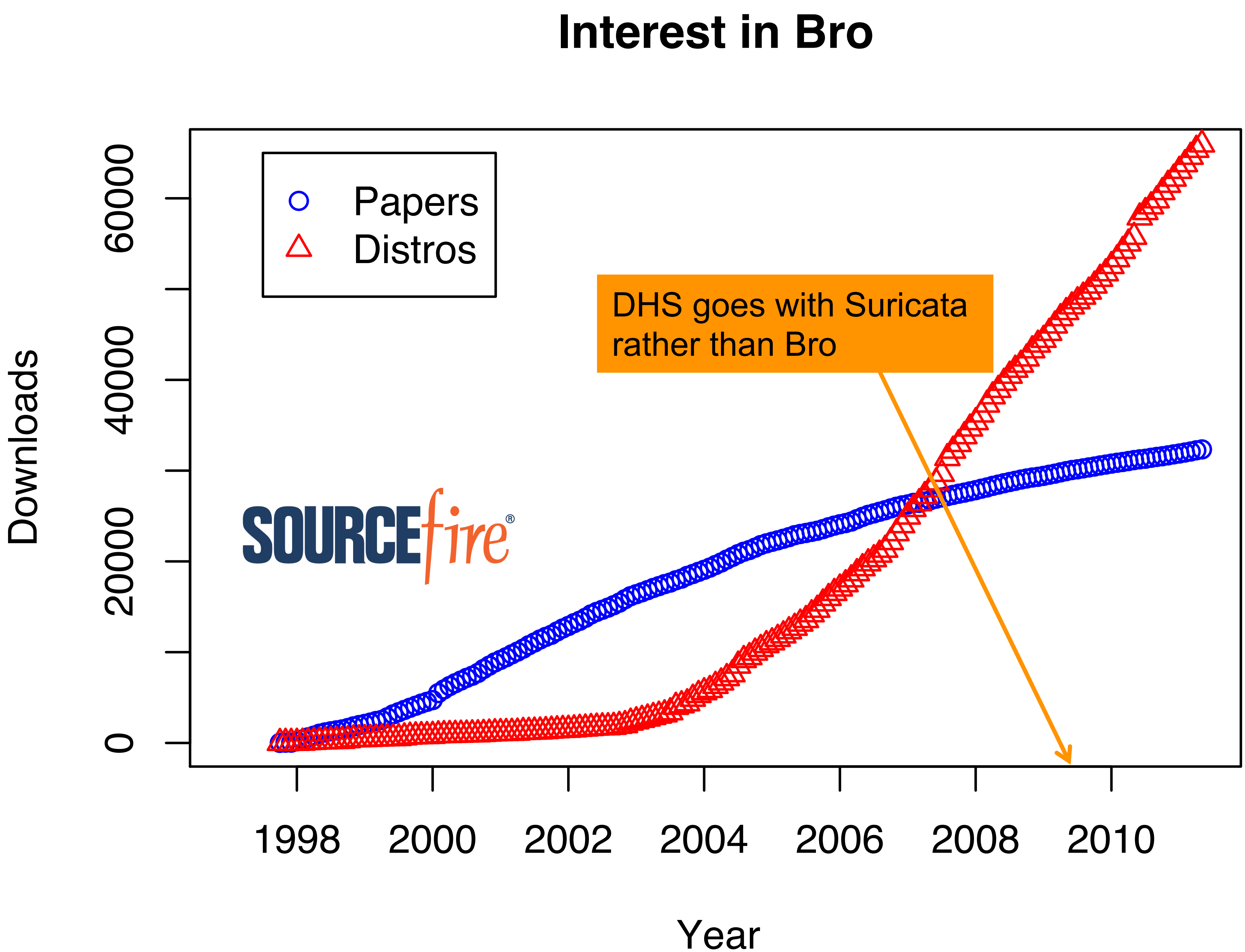
Award Abstract #0627320

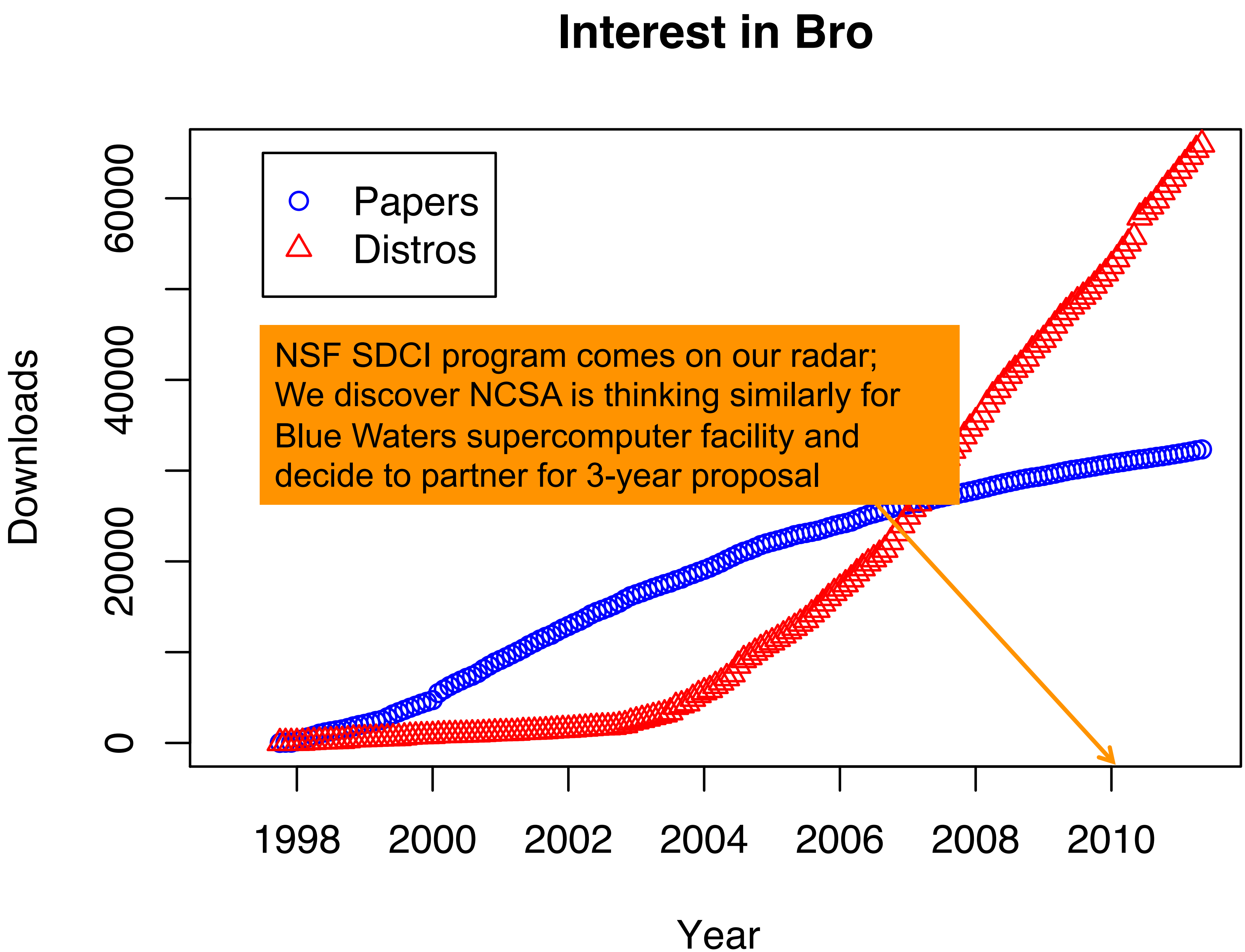
CT-T: Approaches to Network Defense Proven in Open Scientific Environments

NSF Org:	CNS Division Of Computer and Network Systems
Program Manager:	Carl Landwehr CNS Division Of Computer and Network Systems CSE Direct For Computer & Info Scie & Enginr
Start Date:	October 1, 2006
$1,999,054 ? **End Date:**	September 30, 2009 (Estimated)
Awarded Amount to Date:	$236,066.00
Investigator(s):	Vern Paxson vern@icsi.berkeley.edu (Principal Investigator) Mark Allman (Co-Principal Investigator) Robin Sommer (Co-Principal Investigator)



Interest in Bro
Papers
Distros
Sourcefire IPOs
SOURCEfire
Downloads
0
20000
40000
60000
1998
2000
2002
2004
2006
2008
2010
Year


Interest in Bro
Papers
Distros
DHS goes with Suricata rather than Bro
SOURCEfire
Downloads
0
20000
40000
60000
1998
2000
2002
2004
2006
2008
2010
Year


Interest in Bro
Papers
Distros
NSF SDCI program comes on our radar;
We discover NCSA is thinking similarly for
Blue Waters supercomputer facility and
decide to partner for 3-year proposal
Downloads
0
20000
40000
60000
1998
2000
2002
2004
2006
2008
2010
Year

National Science Foundation
WHERE DISCOVERIES BEGIN

Award Abstract #1032889

SDCI Sec Improvement: Enhancing Bro for Operational Network Security Monitoring in Scientific Environments

NSF Org:	ACI Div Of Advanced Cyberinfrastructure
Program Manager:	Anita Nikolich ACI Div Of Advanced Cyberinfrastructure CSE Direct For Computer & Info Scie & Enginr
Start Date:	September 1, 2010
$2,995,905 ? **End Date:**	August 31, 2014 (Estimated)
Awarded Amount to Date:	$2,995,905.00
Investigator(s):	Robin Sommer robin@icsi.berkeley.edu (Principal Investigator) Vern Paxson (Co-Principal Investigator) Adam Slagell (Co-Principal Investigator)

National Science Foundation
WHERE DISCOVERIES BEGIN

Award Abstract #1032889

SDCI
Securi

More specifically, this project (1) improves the perspective of Bro's end-users by providing extensive up-to-date documentation and support, and refining many of the rough edges that the system has accumulated over time; (2) unifies and modernizes Bro's current code base that has evolved over 14 years of active development; (3) improves Bro's processing performance to the degree required for operation in current and future large-scale scientific environments; and (4) adds new data analysis functionality in the form of a highly interactive graphical user interface and a transparent database

Awa

Investigator(s): Robin Sommer robin@icsi.berkeley.edu (Principal Investigator)
Vern Paxson (Co-Principal Investigator)
Adam Slagell (Co-Principal Investigator)

National Science Foundation
WHERE DISCOVERIES BEGIN

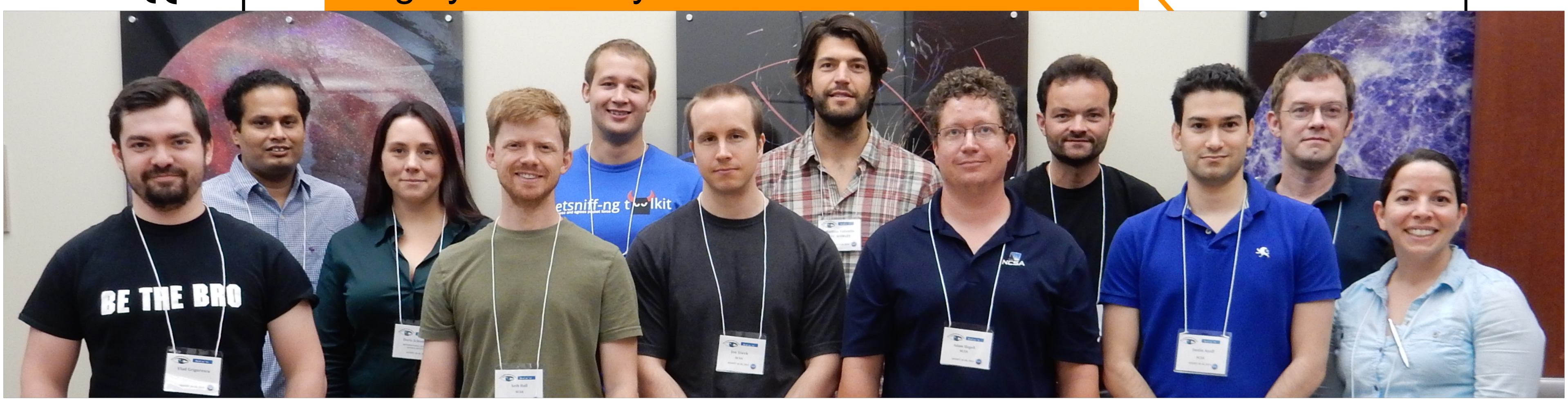
Award Abstract #1032889

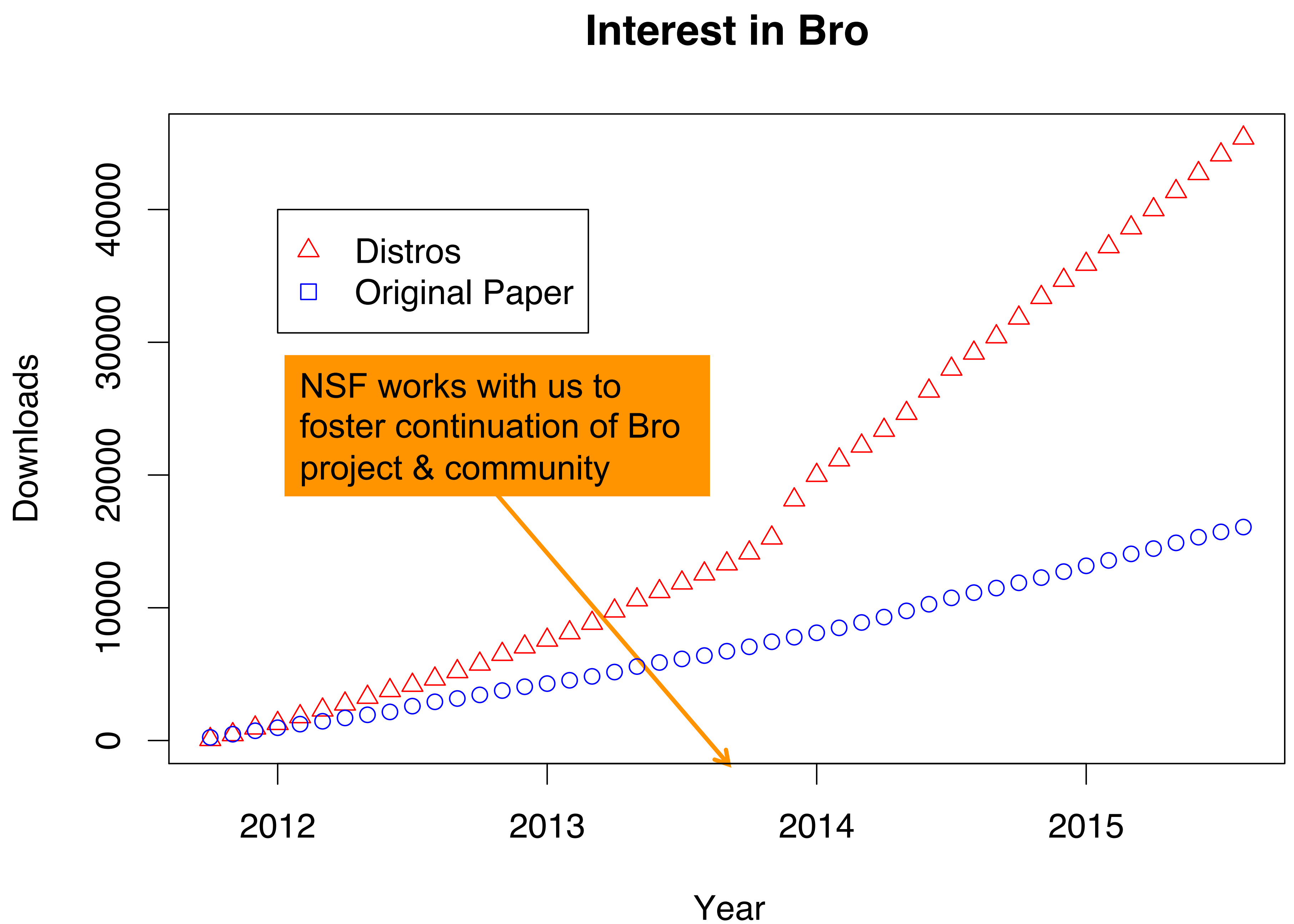
SDCI Sec Improvement: Enhancing Bro for Operational Network Security Monitoring in Scientific Environments

NSF Org:	ACI Div Of Advanced Cyberinfrastructure
Program Manager:	yberinfrastructure er & Info Scie & Enginr
Start Date:	
End Date:	nated)
Awarded Amount to Date:	
Investigator(s):	Robin Sommer robin@icsi.berkeley.edu (Principal Investigator) Vern Paxson (Co-Principal Investigator) Adam Slagell (Co-Principal Investigator)

Interest in Bro

NSF SDCI program comes on
We discover NCSA is thinking
Blue Waters supercomputer fa
decide to partner for 3-year proposal;
Major Luck #1: Seth is available to hire!
Major Luck #2: new collaboration *gels* highly effectively!





Interest in Bro
Distros
Original Paper
NSF works with us to foster continuation of Bro project & community
Downloads
0
10000
20000
30000
40000
2012
2013
2014
2015
Year

National Science Foundation
WHERE DISCOVERIES BEGIN

Award Abstract #1348077

A Bro Center of Expertise for the NSF Community

NSF Org:	ACI Div Of Advanced Cyberinfrastructure
Program Manager:	Kevin L. Thompson ACI Div Of Advanced Cyberinfrastructure CSE Direct For Computer & Info Scie & Enginr
Start Date:	October 1, 2013
$3,729,977 ? **End Date:**	September 30, 2016 (Estimated)
Awarded Amount to Date:	$3,360,092.00
Investigator(s):	Robin Sommer robin@icsi.berkeley.edu (Principal Investigator) Vern Paxson (Co-Principal Investigator) Adam Slagell (Co-Principal Investigator)

National Science Foundation
WHERE DISCOVERIES BEGIN

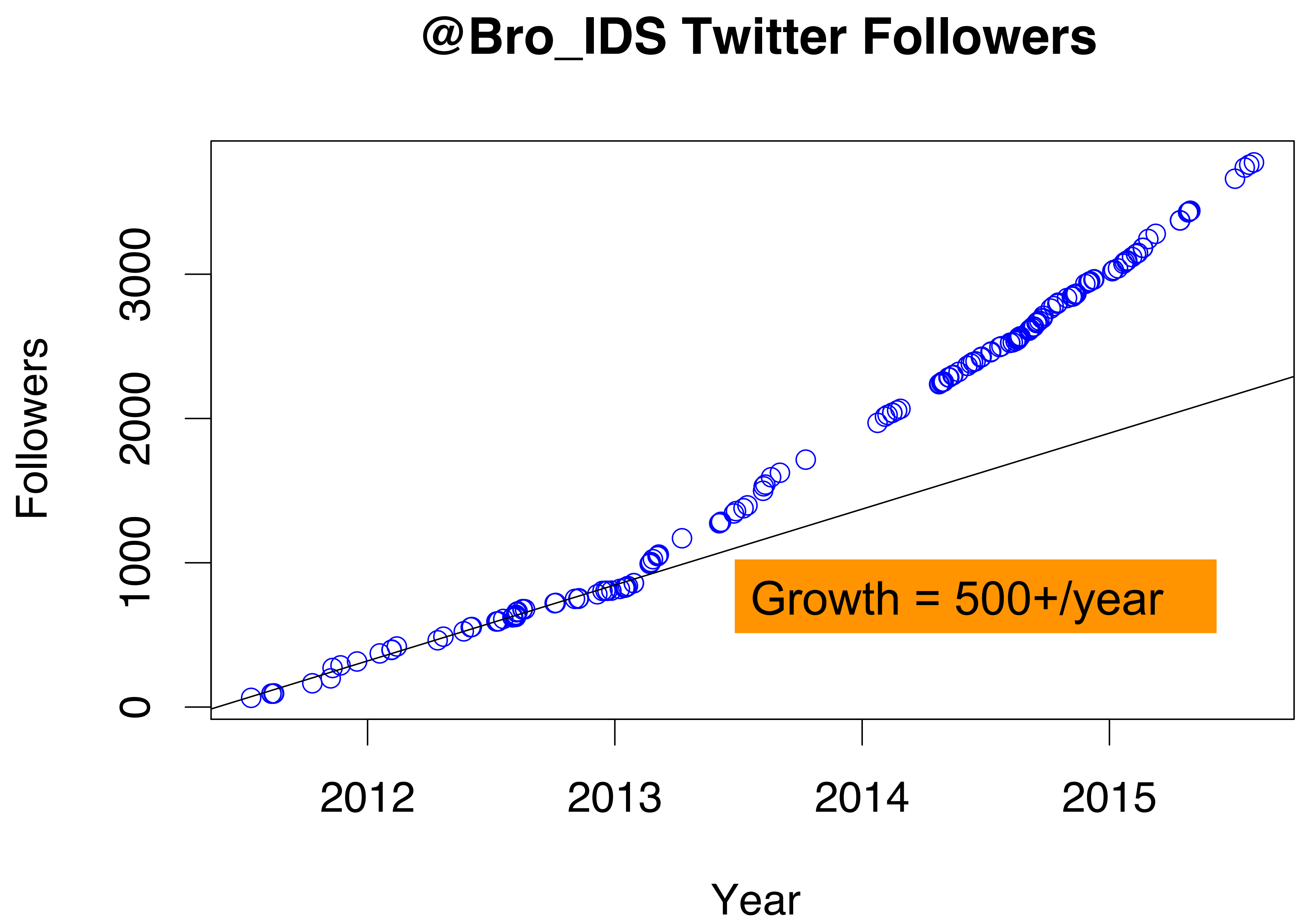
Award Abstract #1348077

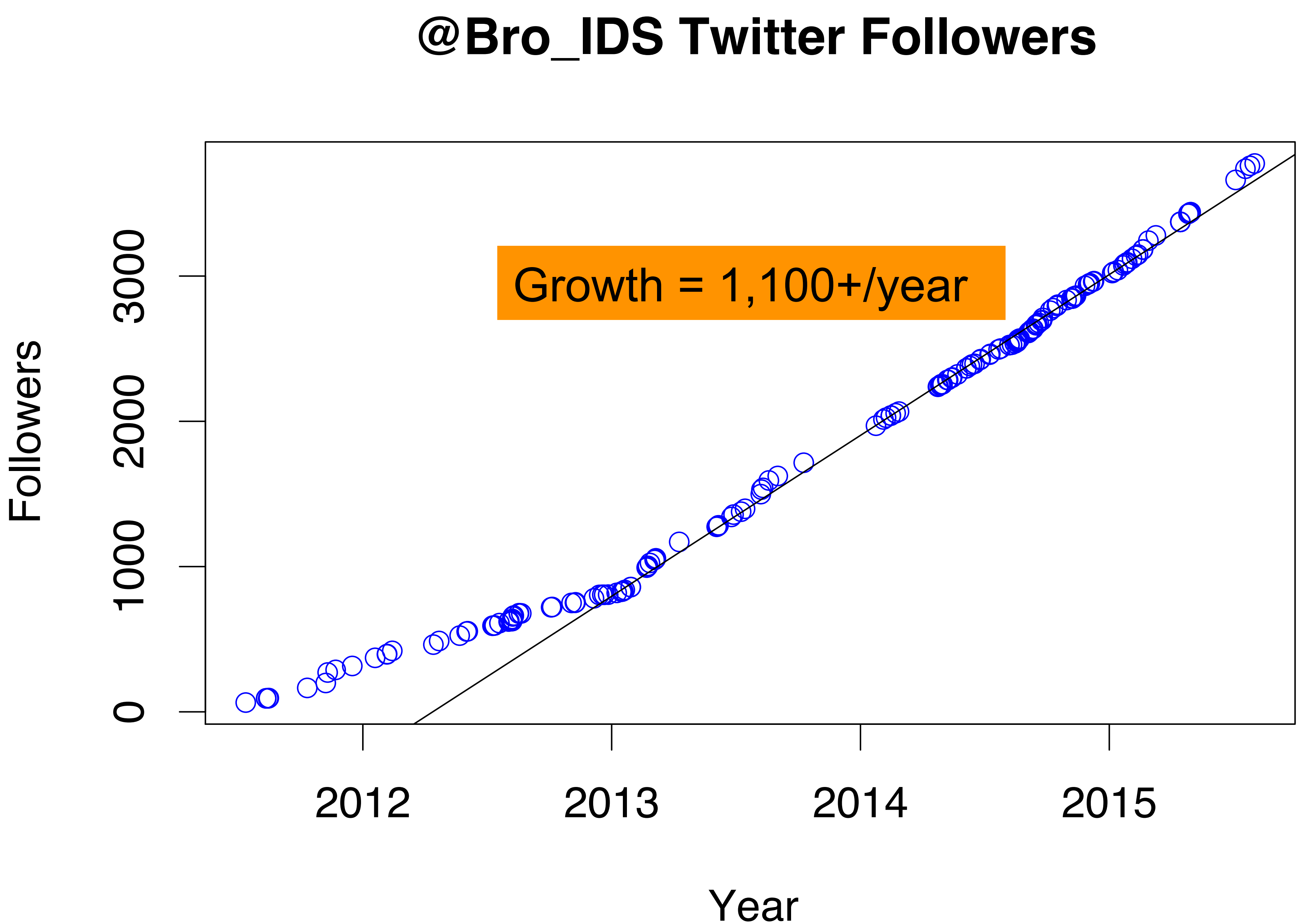
A Bro Center of Expertise for the NSF Community

NSF Org: ACI
Div Of Advanced Cyberinfrastructure

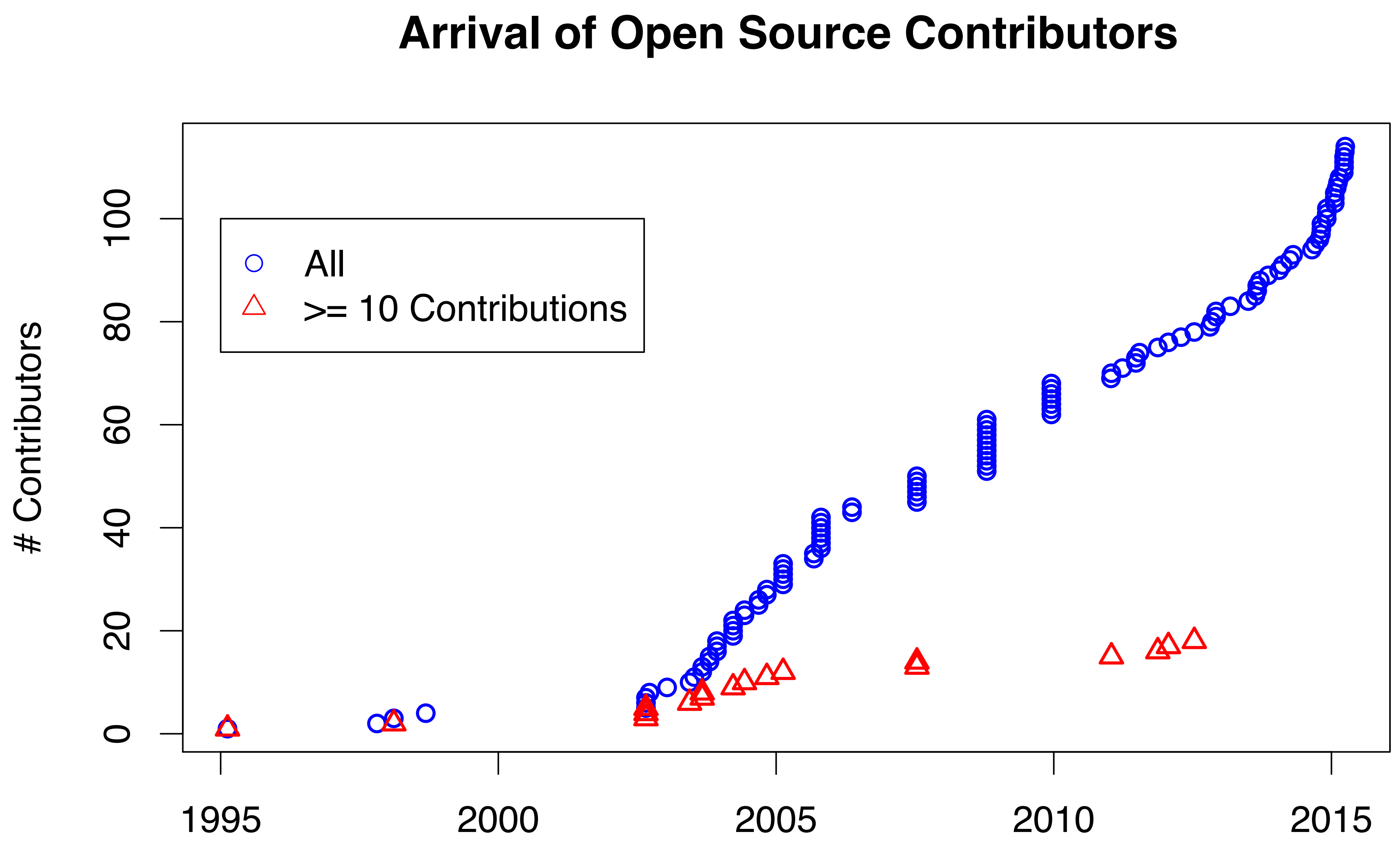
This activity promotes Bro as a comprehensive, low-cost security capability for these communities; providing guidance and support on all aspects of a Bro installation. The project devises reference scenarios for deployment and integration; and develops novel technical capabilities that cater to NSF environments. The project supports existing Bro users in optimizing and extending their setups, and makes Bro's capabilities available to new sites and projects that lack the resources to deploy Bro effectively on their own. At a technical level, the project is the focal point of Bro's open-source development, maintaining its code base and documentation. To the research community, the project acts as a facilitator for transitioning networking research results into practice by leveraging Bro as a deployment platform.

Investigator(s): Robin Sommer robin@icsi.berkeley.edu (Principal Investigator)
Vern Paxson (Co-Principal Investigator)
Adam Slagell (Co-Principal Investigator)


@Bro_IDS Twitter Followers
Followers
0
1000
2000
3000
Growth = 500+/year
2012
2013
2014
2015
Year



@Bro_IDS Twitter Followers
Growth = 1,100+/year
Followers
0
1000
2000
3000
2012
2013
2014
2015
Year



Arrival of Open Source Contributors
Contributors
0
20
40
60
80
100
1995
2000
2005
2010
2015
All
>= 10 Contributions

Looking Forward

- Visibility: ***Deep Bro***
  - Extensive interior site deployment
  - Enterprise protocols; distributed coordination
- Performance: *HILTI* + *Spicy*
  - Compiled multithreaded/multicore Bro
- Archive: VAST (*Visibility Across Space and Time*)
  - Very high-performance event/logging archive
  - To support interactive forensic analysis …
  - … and capture of IOCs
- Longevity & Support: Bro Foundation
  - Via Software Freedom Conservancy