Intrusion Detection System

Time Machine

Dynamic Application Detection

NIDS: Two generic problems

- Attack identified
  - But what happened in the past???
- Application identification
  - Only by port number!
  - Yet applications use arbitrary ports
    - Benign reasons
      - Lack administrator privileges
      - Circumvention of firewall, e.g., Skype
      - Application tunnels
    - Malicious intend
      - Evasion of security monitoring
      - E.g.: IRC based botnets on ports other than 666x/tcp
      - E.g.: ftp servers on ports other than 21/tcp

Time Machine: Motivation

- Trouble-Shooting
  - What happened before a fault?
- Security monitoring ("Forensics")
  - Break in 2 days ago:
    - How?
    - What else have they done?
- NIDS: offloading

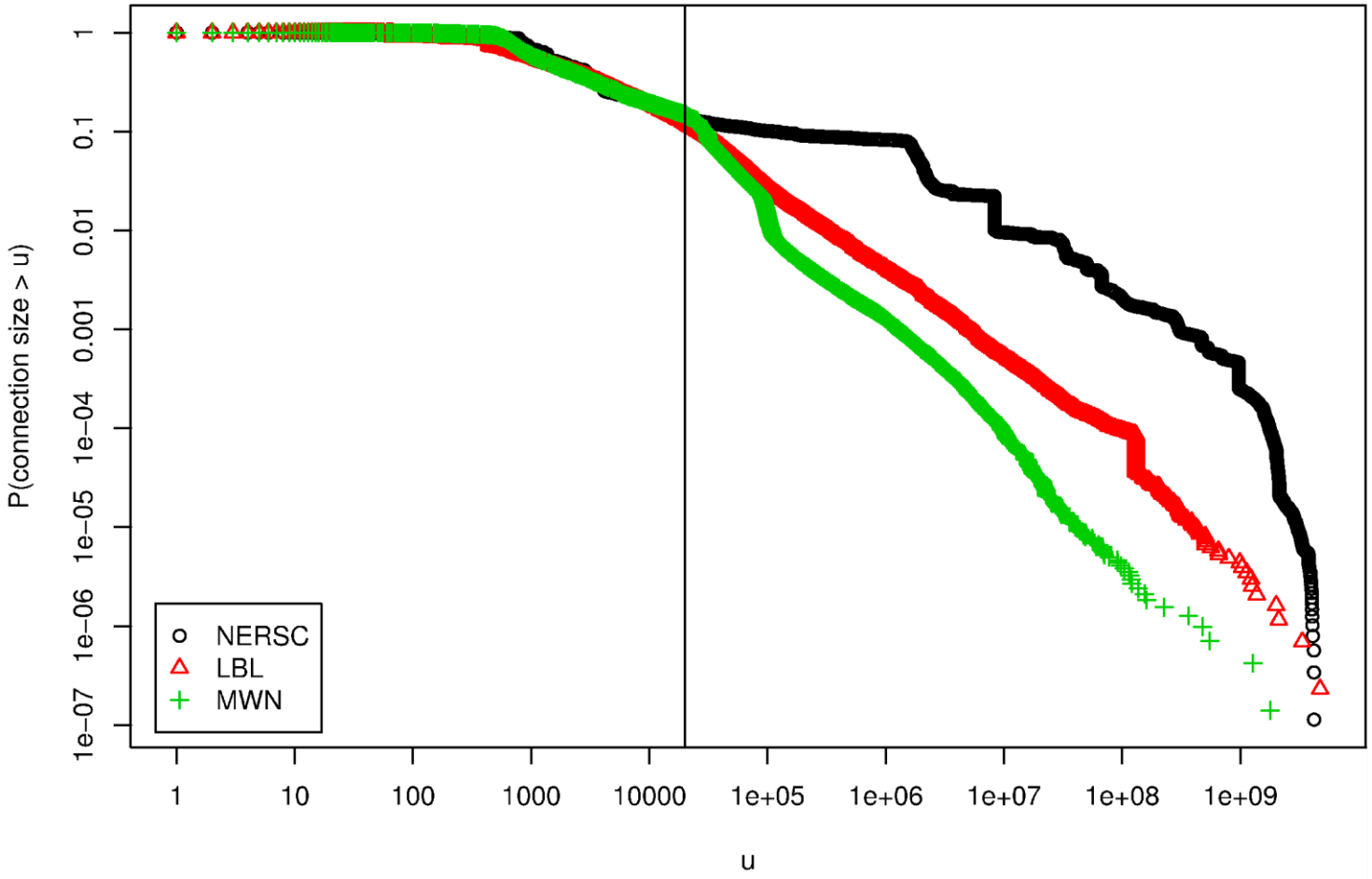
- Ideas
  - Keep packet traces of past events!

Time Machine: Problem

- Recording
  - High data volume, high load (CPU, disk)
    - E.g., MWN:
      - Gbps link
      - 2004: 2 TB / day
      - 2007: 4-6TB / day
      - 350 / 950 Mbps busy hour load
- Retrieval
  - Using the captured data
  - Find relevant "needle in the haystack"

Time Machine: Concept

- **Buffer packets up to connection cutoff**



Connection sizes: Heavy-tailed

LBL (2004):

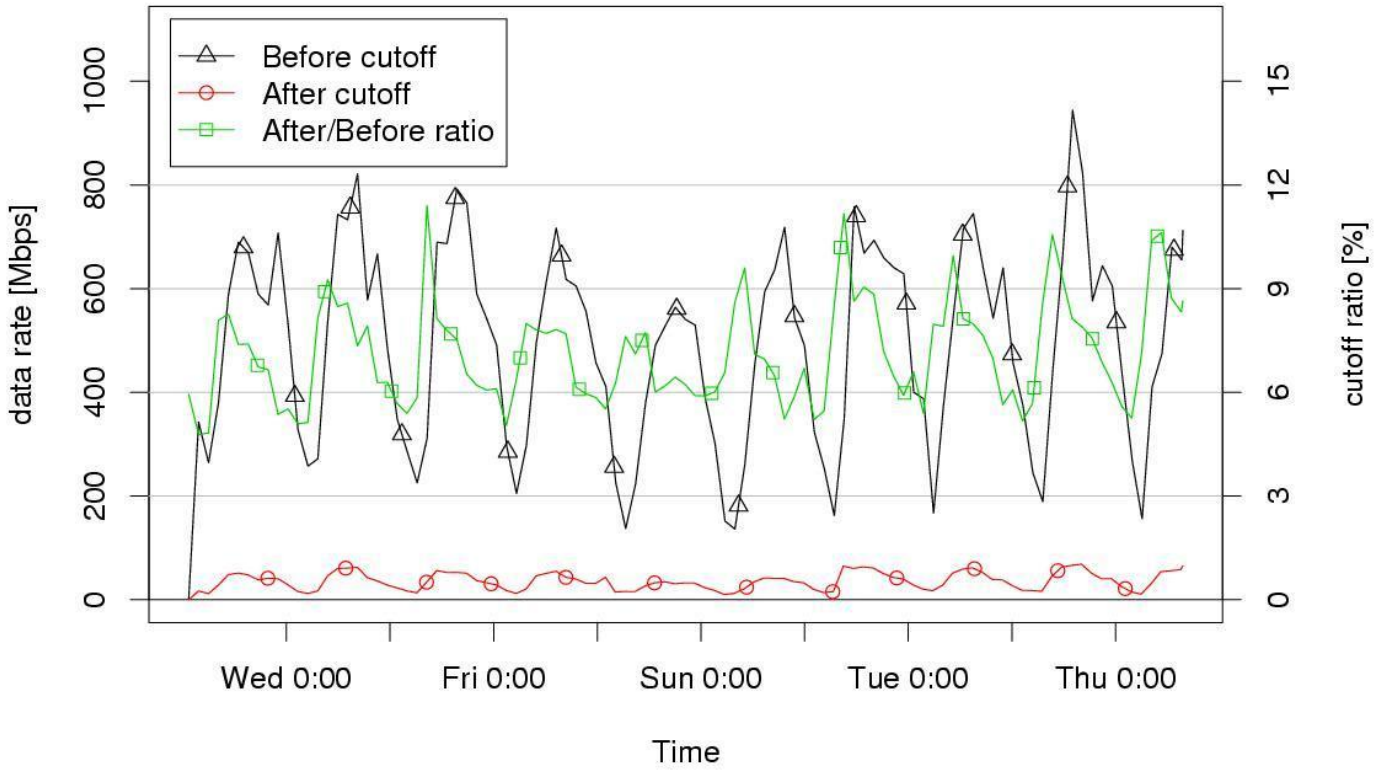
- > 20 KB
  - 12% of connections
  - 96% of all bytes

NERSC (2004):

- > 20 KB
  - 14% of connections
  - 99.86% of bytes

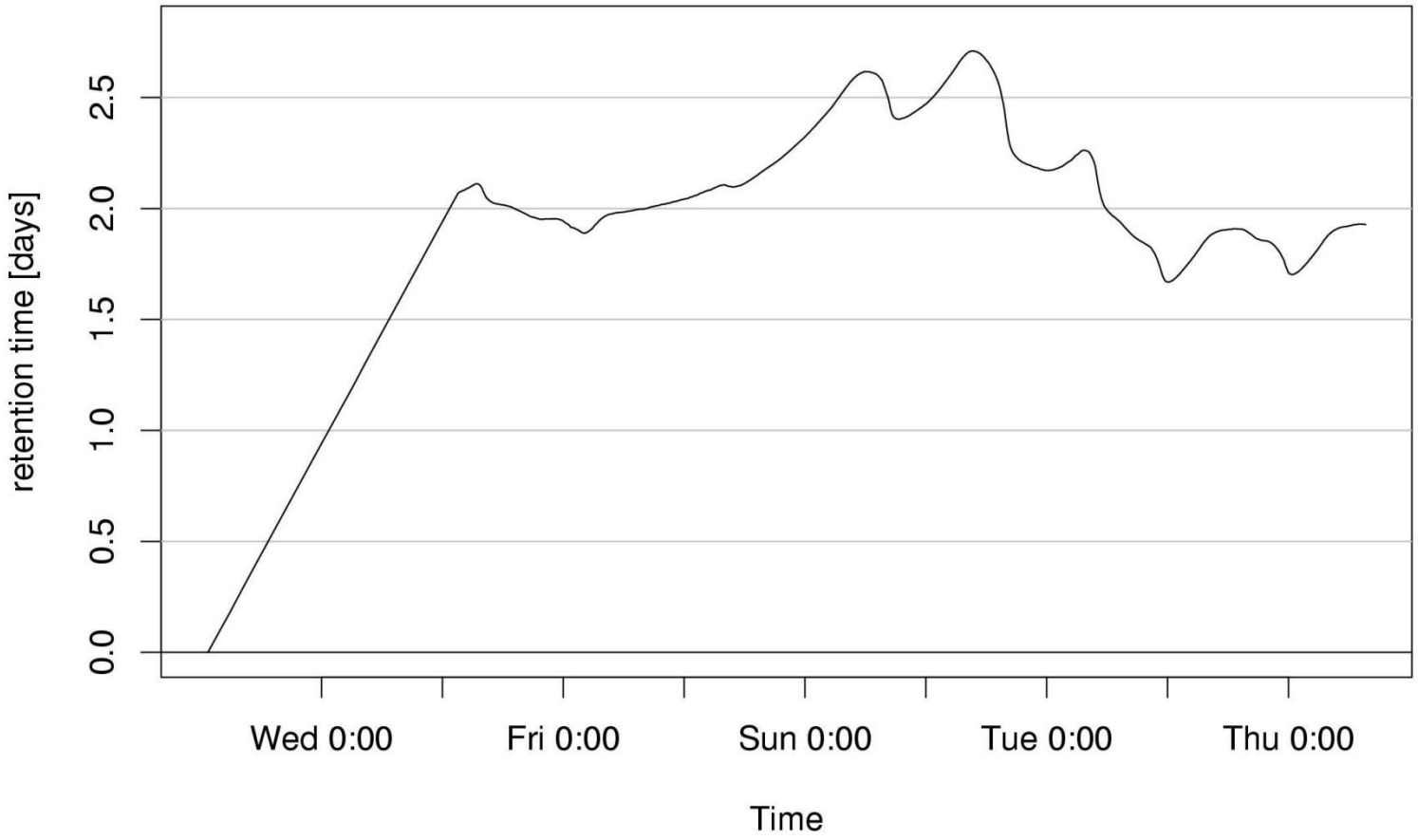
TM: Capabilities (cutoff: 15KB)

- MWN: 10Gb uplink, 3-6 TB per day, 50,000 users
- 600MB memory buffer, 800GB disk buffer



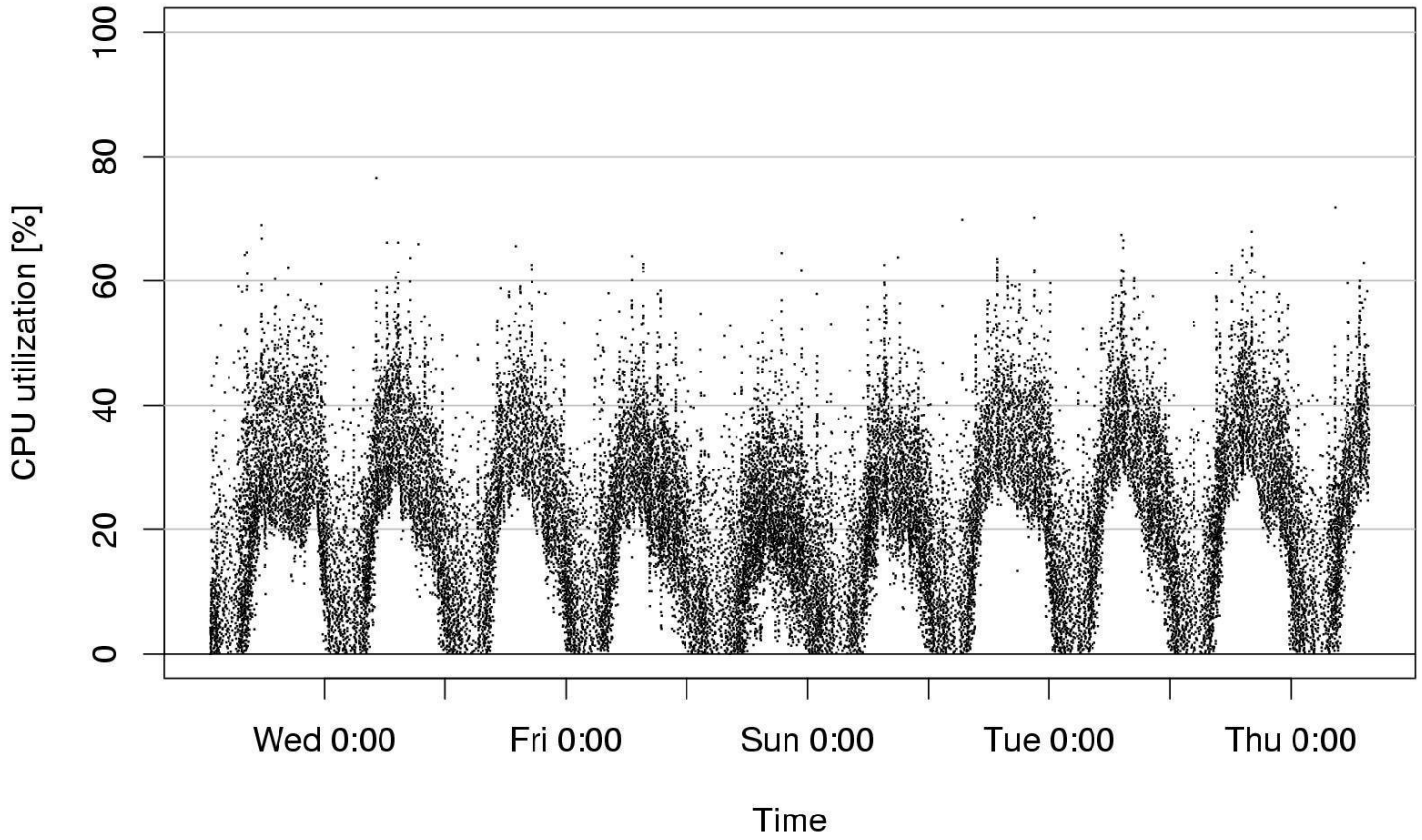
TM: Retention time

- Huge potential

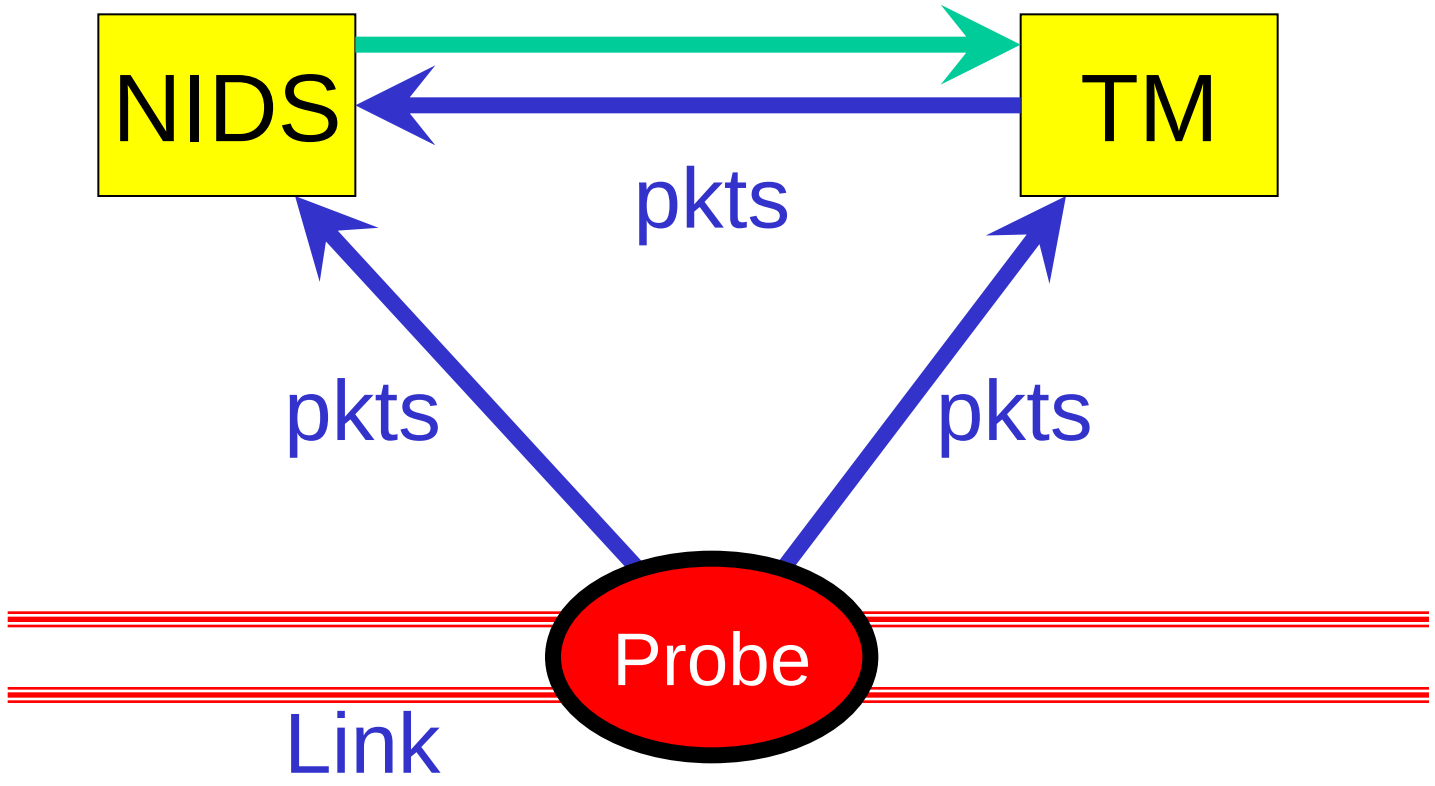


TM: CPU utilization

- Head room available for query processing



Coupling NIDS with Time Machine



TM-NIDS: Applications

- Semi-automatic forensics
  - NIDS detects standard traffic on non-standard port
  - TM captures connections to file
- Scanner supervision
  - NIDS recognizes scanner
  - TM supplies past traffic on successful connection
- Host supervision
  - NIDS recognizes weird traffic from some IP
  - Instructs TM to capture / store all traffic
  - Suspends cutoff

TM-NIDS: Applications (2.)

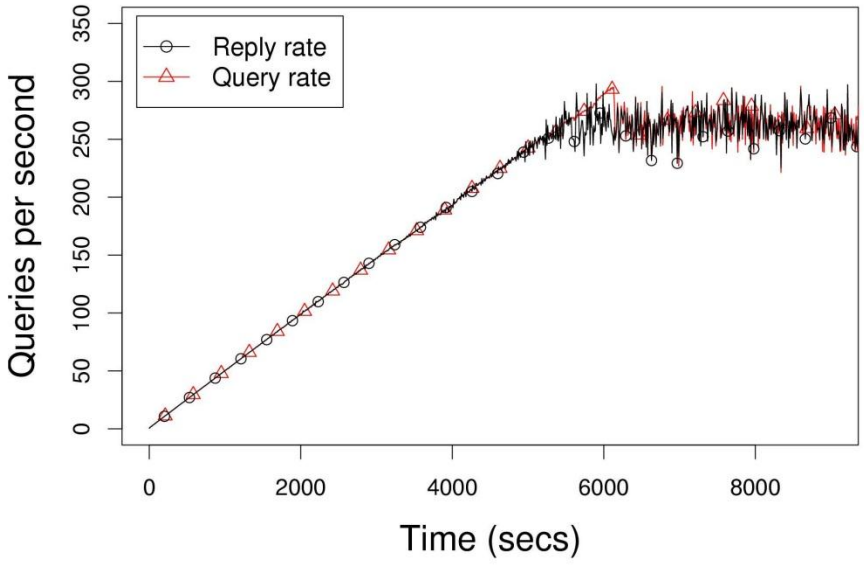
- Offloading NIDS
  - NIDS processes HTTP requests only
    If suspicious request TM supplies response
  - NIDS processes FTP control data only
    If suspicious request TM supplies data connection
- Error correction
  - NIDS recognizes content gap
    TM supplies missing data

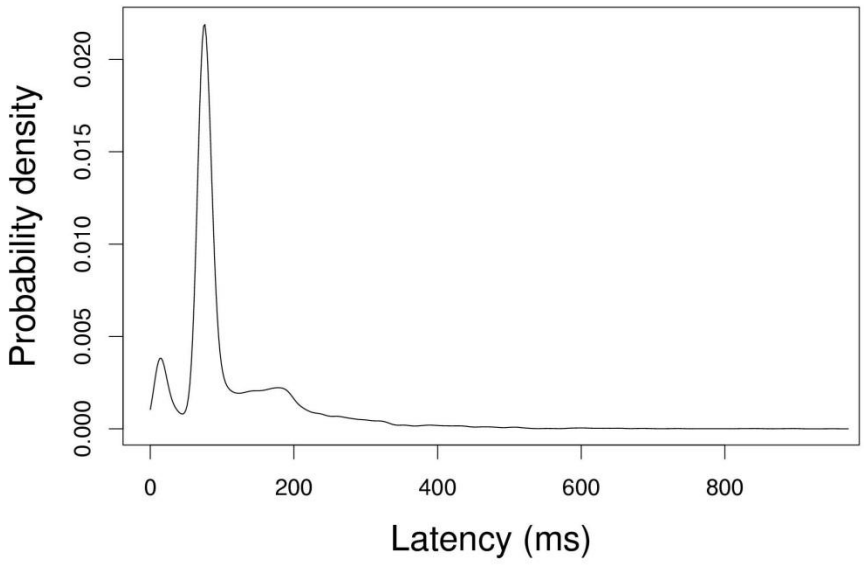
Coupling NIDS with Time Machine

- Requirements
  - Simple interface
  - NIDS processing of TM data as usual
- Complications
  - Two data sources
  - Duplicate data packets from both sources
  - NIDS needs to handle live as well as historic data (violates time sequence assumptions)

Query rate

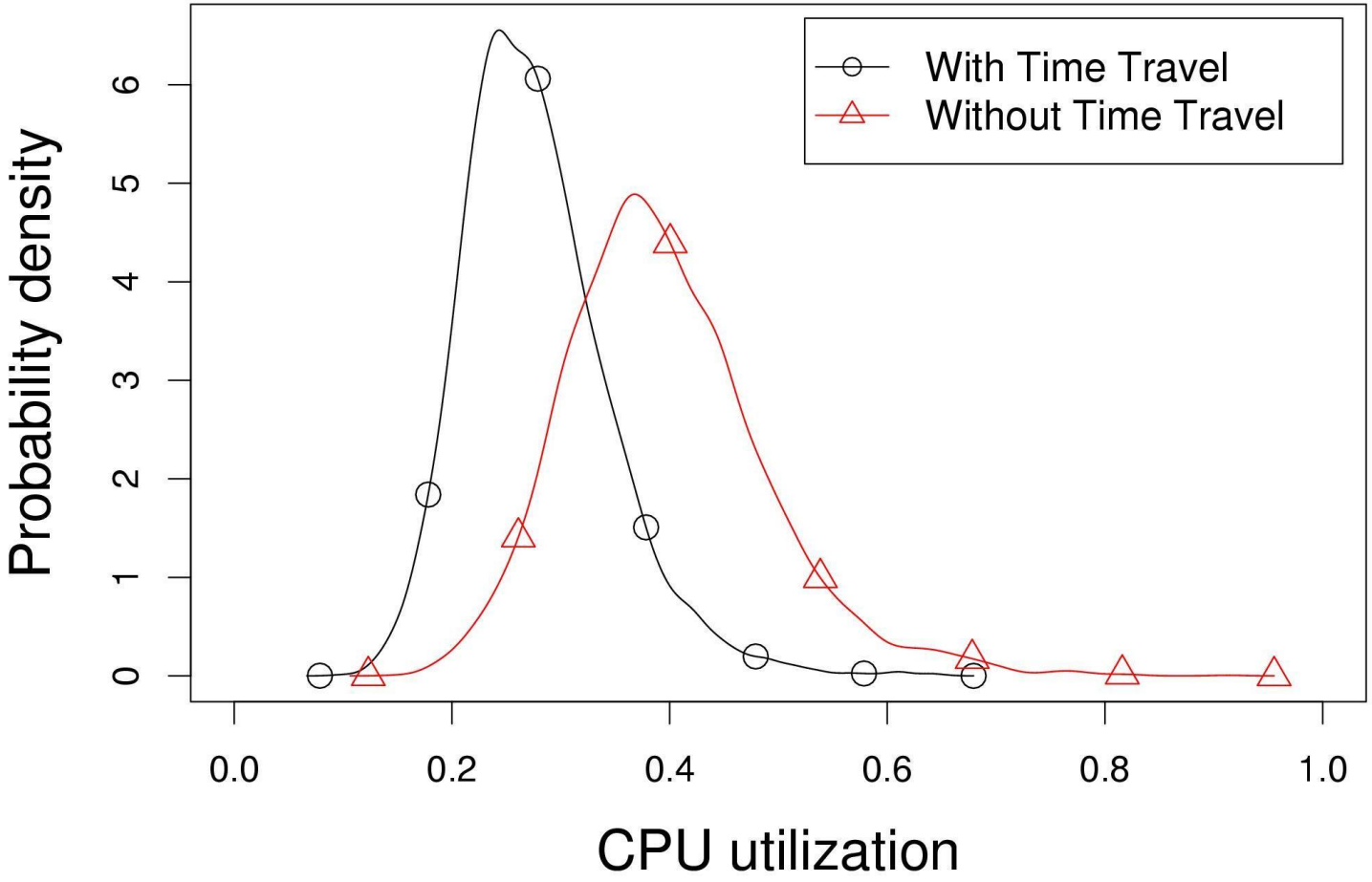
- In memory queries
- Query rate increases
- Overhead and latency small





HTTP analysis

- Significant CPU gains by offloading HTTP reply



TM-NIDS: Summary

- Stand-alone time machine:
  a very powerful tool
- Coupling a TM with an NIDS:
  an even more powerful tool!

NIDS: Two generic problems

- Attack identified
  - But what happened in the past???
- Application identification
  - Only by port number!
  - Yet applications use arbitrary ports
    - Benign reasons
      - Lack administrator privileges
      - Circumvention of firewall, e.g., Skype
      - Application tunnels
    - Malicious intend
      - Evasion of security monitoring
      - E.g.: IRC based botnets on ports other than 666x/tcp
      - E.g.: ftp servers on ports other than 21/tcp

Applications on non-standard ports

- Data (Oct. 2005):
  - 24 hour full packet trace from Münchner WissenschaftsNetz (MWN)
  - 3.2 TB of data in 6.3 billion pkts, 137M connections
- Application signatures from l7-filter system
- Focus on HTTP, IRC, FTP, SMTP

Ports accounting > 1% of conns.

Port		% Conns	% Success	% Payload
Web	**80**	**70.82%**	**68.13%**	**72.59%**
	445	3.53%	0.01%	0.00%
Web	443	2.34%	2.08%	1.29%
SSH	22	2.12%	1.75%	1.71%
Mail	25	1.85%	1.05%	1.71%
	1042	1.66%	0.00%	0.00%
	1433	1.06%	0.00%	0.00%
	135	1.04%	0.00%	0.00%
	< 1024	83.68%	73.73%	79.05%
	> 1024	**16.32%**	**4.08%**	**20.95%**

Signature-based app. detection

- Port information offers no information for ports > 1024
- l7-filter system application signatures
- HTTP highly attractive for hiding other applications
- Most successful conns. trigger expected signature
- FTP higher percentage of false negatives

Method	HTTP	IRC	FTP	SMTP
Port (succ.)	**93,429K**	**75,876**	**151,700**	**1,447K**
Signature	**94,326K**	**73,962**	**125,296**	**1,416K**
expected port	92,228K	71,467	98,017	1,415K
other port	**2,126K**	**2,495**	**27,279**	**265**

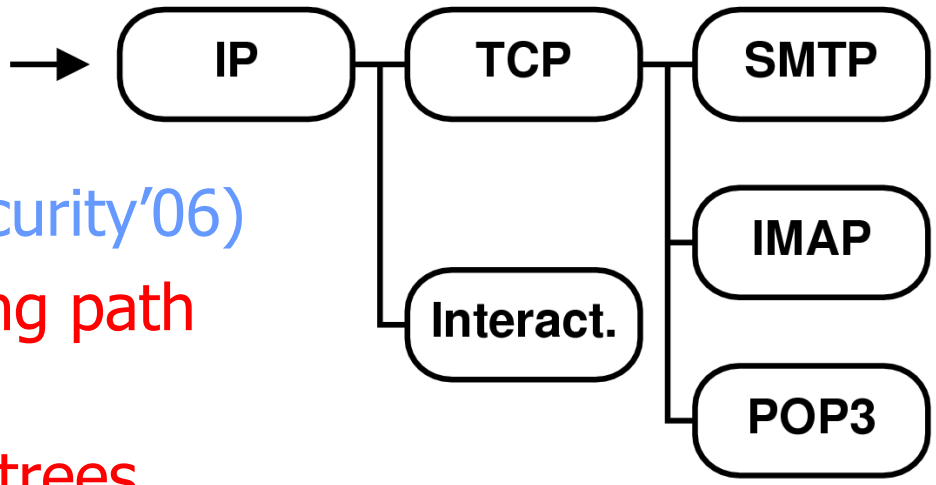
Signature detection: Well known ports

- Some connections trigger more than one signature
- Not yet wide-spread abuse
- But some misappropriate use of well known ports

Port	HTTP	IRC	SMTP	Other	No match
80	**92,228,291**	59	0	41,086	1,158,977
666x	1,217	**71,650**	0	4,238	524
25	459	2	**1,415,428**	195	31,889

Architecture for dynamic analysis

- Goals
  - Detection scheme independence
  - Dynamic analysis
  - Modularity
  - Efficiency
  - Customizability
- Design (USENIX Security'06)
  - Dynamic processing path
  - Per connection dynamic analyzer trees



Reliable detection of non-standard ports

- UCB: 1 day

	internal	remote
FTP servers:	6	17
HTTP servers:	568	54,830
IRC servers:	2	33
SMTP servers:	8	8

- MWN similar
- Non-standard port connection
  - UCB: 99% HTTP (28% Gnutella, 22% Apache)
  - MWN: 92% HTTP (21% BitTorrent, 20% Gnutella), 7% FTP
  - Two open HTTP proxy detected: now closed
  - SMTP server that allowed relay: now closed

Detecting IRC-based Botnets

- Idea
  - Botnets like IRC protocol (remote control features)
  - Botnet detector on top of IRC analyser
    - Checks client nickname for typical patterns
    - Checks channel topics for typical botnet commands
    - Checks if new clients connect with IRC to identified bot-servers
- Results
  - MWN:
    - > 100 distinct IPs with Botnet clients
    - Now part of a automatic prevention system
  - UCB:
    - 15 distinct IPs

Summary: Dynamic app. analysis

- Ideas:
  - Dynamic processing path
  - Per connection dynamic analyzer trees
- Operational at three large-scale networks
- Detected significant number of security incidents
- Bot-detection now automatically blocks IP