

WormTerminator: An Effective Containment of Unknown and Polymorphic Fast Spreading Worms

Songqing Chen, Xinyuan Wang, Lei Liu

George Mason University, VA

Xinwen Zhang

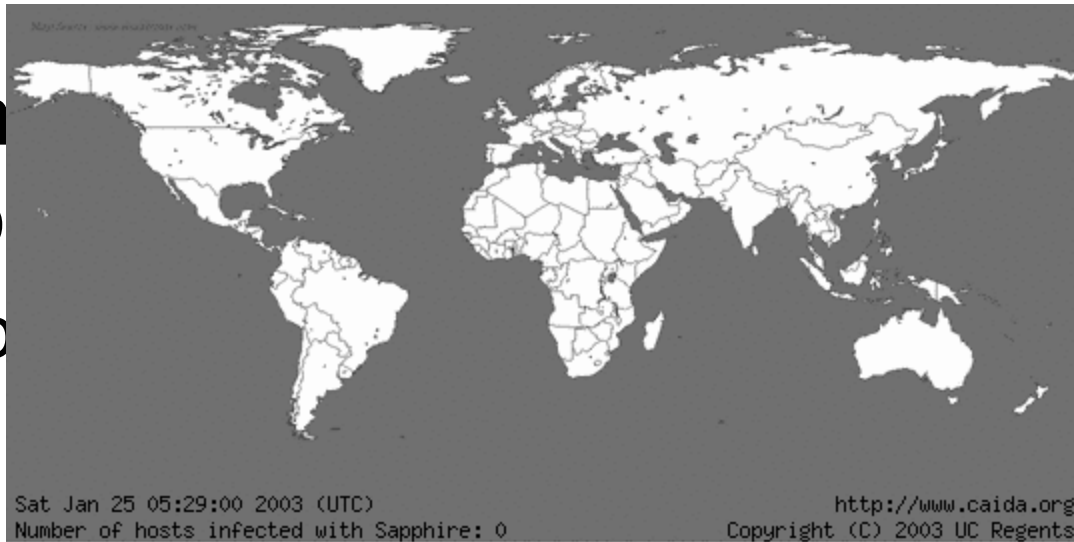
Samsung Computer Science Lab, CA

Zhao Zhang

Iowa State University, IA

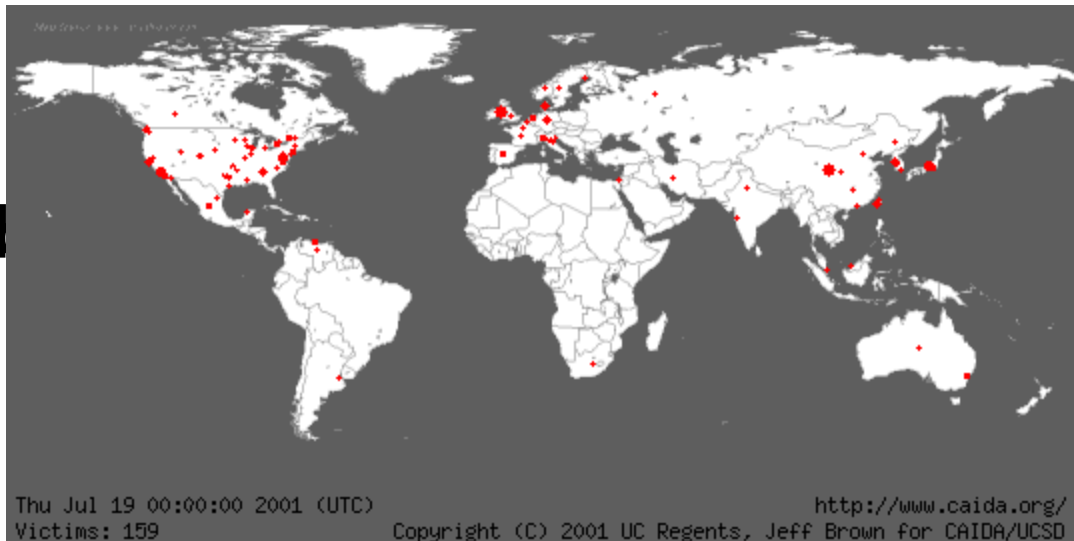
Background

- Slamm
in abo
– Prob



e hosts
rage

- Code
– Doubl



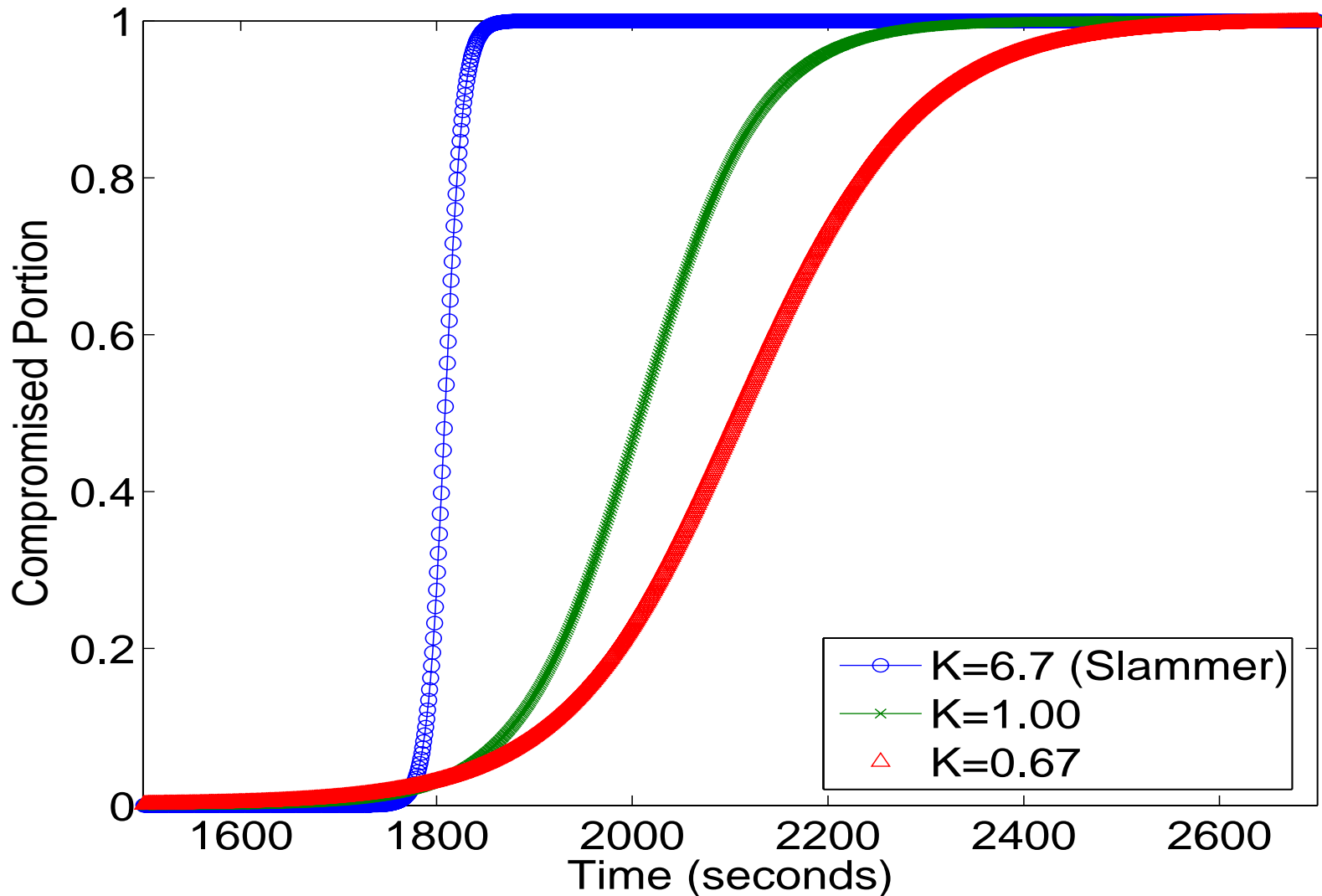
rs
S

•Courtesy
of CAIDA

Existing Solutions

Existing Solutions (con't)

Failure of Traffic Limitation



What Is Desired?

- whether or not they are previously **unknown** or **polymorphic**
- **without** allowing **any** worm **propagation** in the Internet to infect **any other host**
- allowing **all** normal traffic

Are
these
possible?

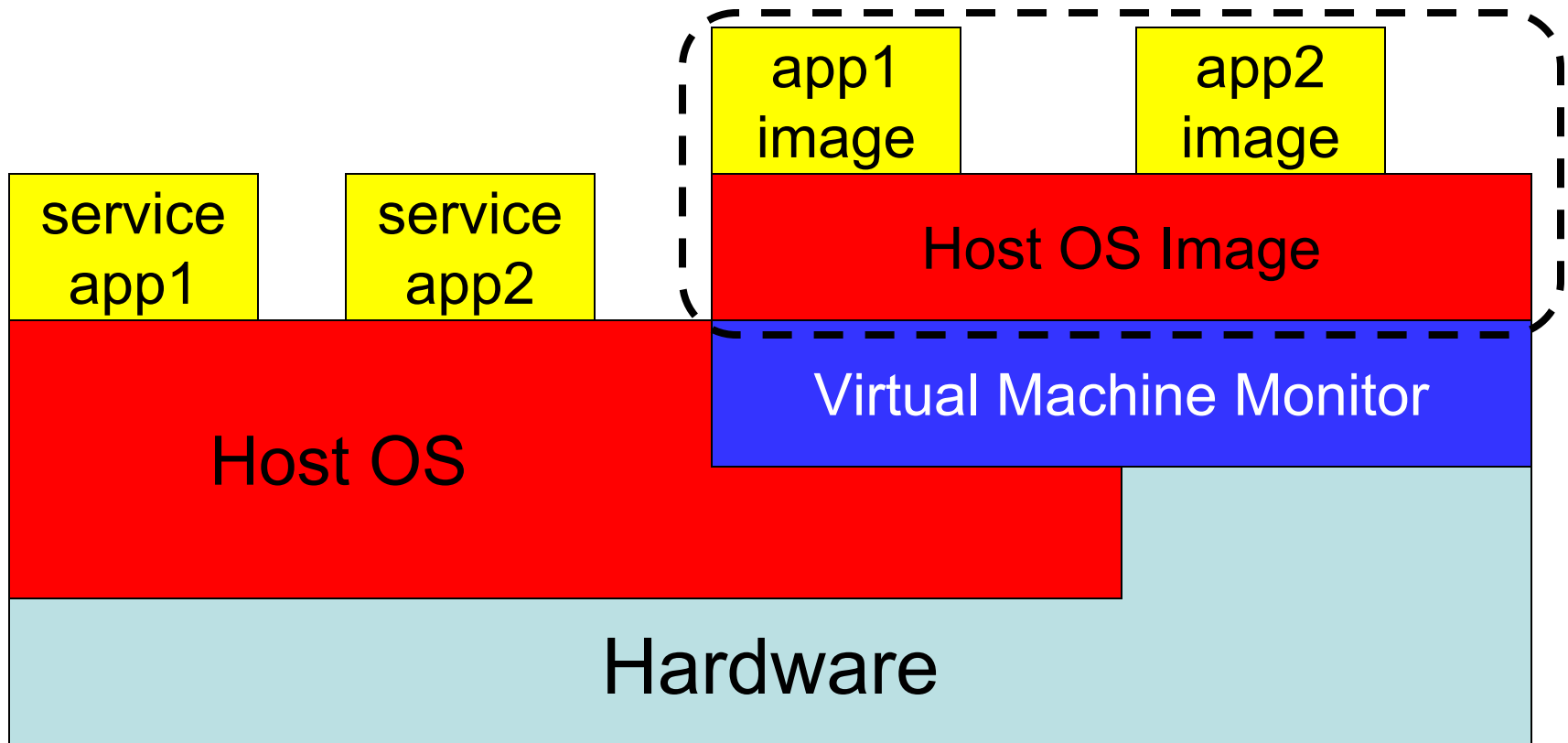
Our Contributions

- Propose WormTerminator to detect the propagation of **any (unknown/polymorphic)** fast worm before it propagates to **any** other Internet host
- Implement a **prototype system**
- Experiment based on a **real** Internet worm

Outline

- Design Principles and Overview
- Design Issues & Implementation
- Evaluations
- Conclusions

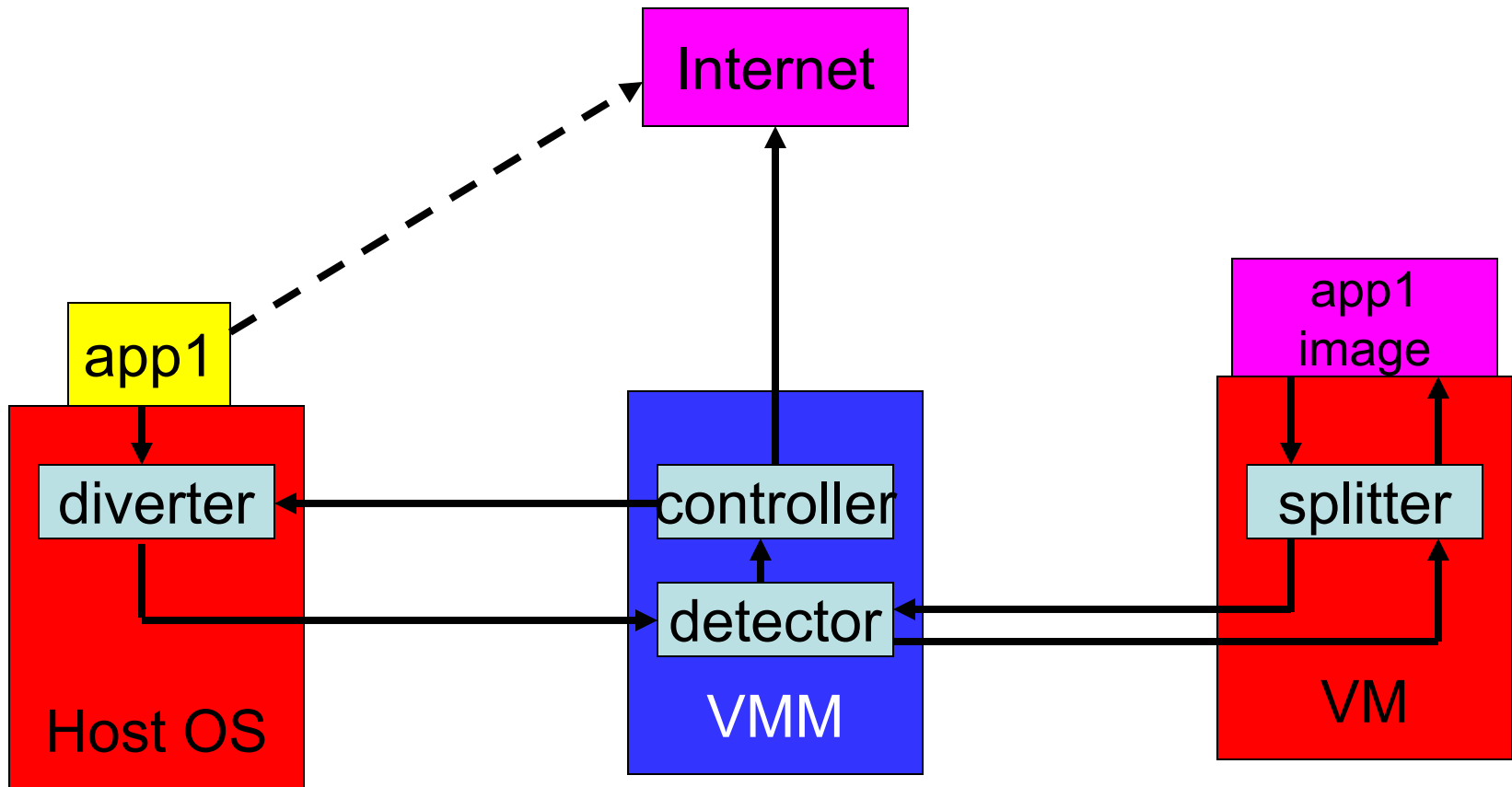
Worm Terminator Architecture



Design Principles

If the very first outgoing traffic is diverted.....

Flow of Control



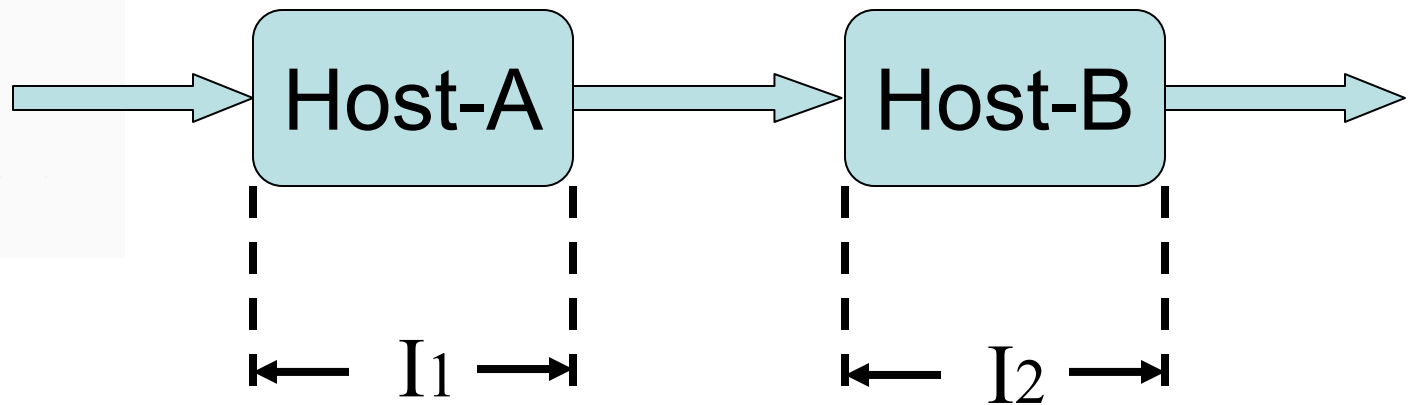
Outline

- Design Principles and Overview
- Design Issues & Implementation
- Evaluations
- Conclusions

Detection Criteria

- A natural criterion is to see, in a certain period of time, *if the VM traffic exhibits worm propagation pattern* after accepting the diverted traffic from the host
- **How long** should the **observation window** be?
 - worm dependent
 - shorter is better

Timing Correlation



$$I_2 = I_1 * VM_{sd}$$

Benign Traffic

- There is benign traffic that may look like worm propagation
 - Email Relay
 - An email server receives a mail and forwards out
 - P2P Search
 - A peer receives a query and forwards to its neighbors
 - P2P Downloading
 - A BitTorrent client uploads a same file piece to multiple peers

Uniques of Benign Traffic

- **Email Relay**
 - The relay mail server is **not** the traffic **destination**
 - **No** processing involved except for tracing information
- **P2P Search**
 - The neighbor information is available **in advance**
 - Queries are **small**
- **P2P Downloading**
 - It is **not unsolicited**

Impact on Applications

- Application **Transparency**
 - **Dynamically** set IP address of the VM
 - Benign **UDP** is directly forwarded
 - The VM becomes a proxy for benign **TCP**
- Performance **Overhead**
 - **Cache** the examined connections

Implementation

- Diverter
 - Kernel module with **ipchains/iptables**
- Splitter
 - **Squid 2.4STABLE1**
- Detector/Controller
 - **Pcap, ipchains/iptables, VMM**
- Connection tracker
 - **/proc**

Outline

- Design Principles and Overview
- Design Issues & Implementation
- Evaluations
- Conclusions

Test Setup

- Worm: **Linux/Slapper**
 - OpenSSL buffer overflow in **libssl**
 - **Apache 1.3** on RedHat, SuSe, Mandrake, Slackware, and Debian
 - More than 3500 computers were infected
- Host runs RedHat 7.3, 2.4 GHz CPU and 1GB memory
 - **User-mode Linux** as the VM
 - Slowdown is set to 18
- Another machine runs as the original source

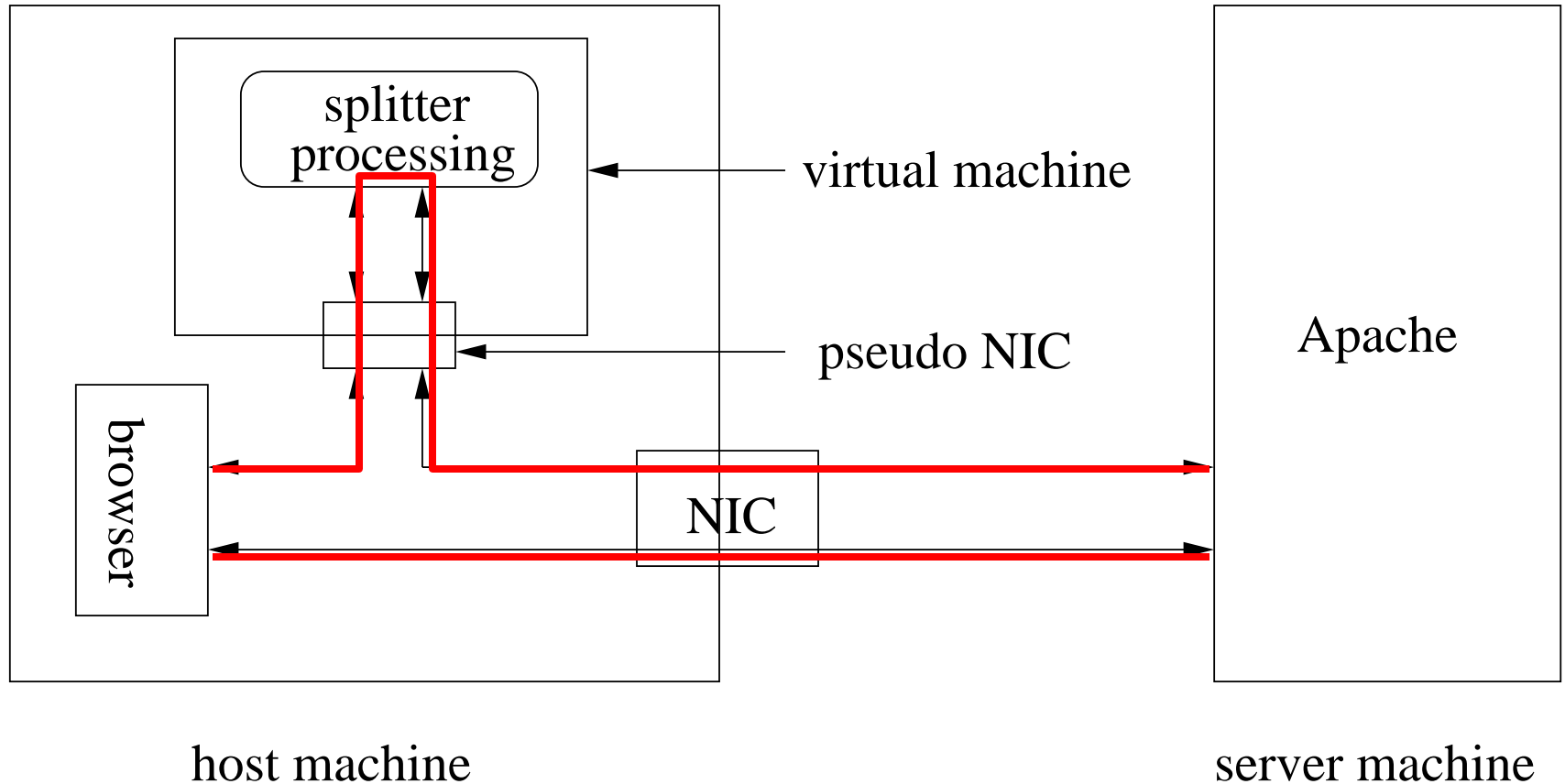
Slapper Test

-- Can WT catch Slapper?

	I1 (In a LAN)		I2 (To the VM)	
	Infection	Code Xfer	Infection	Code Xfer
Average	9.3456	3.0654	91.8893	6.9773
Std_dev	0.4666	0.0120	1.2806	0.1103

*Averaged on 10 runs.

Overhead Test Setup



- Latency: download 1 byte file
- Throughput: download a file of 100 MB

WormTerminator Overhead

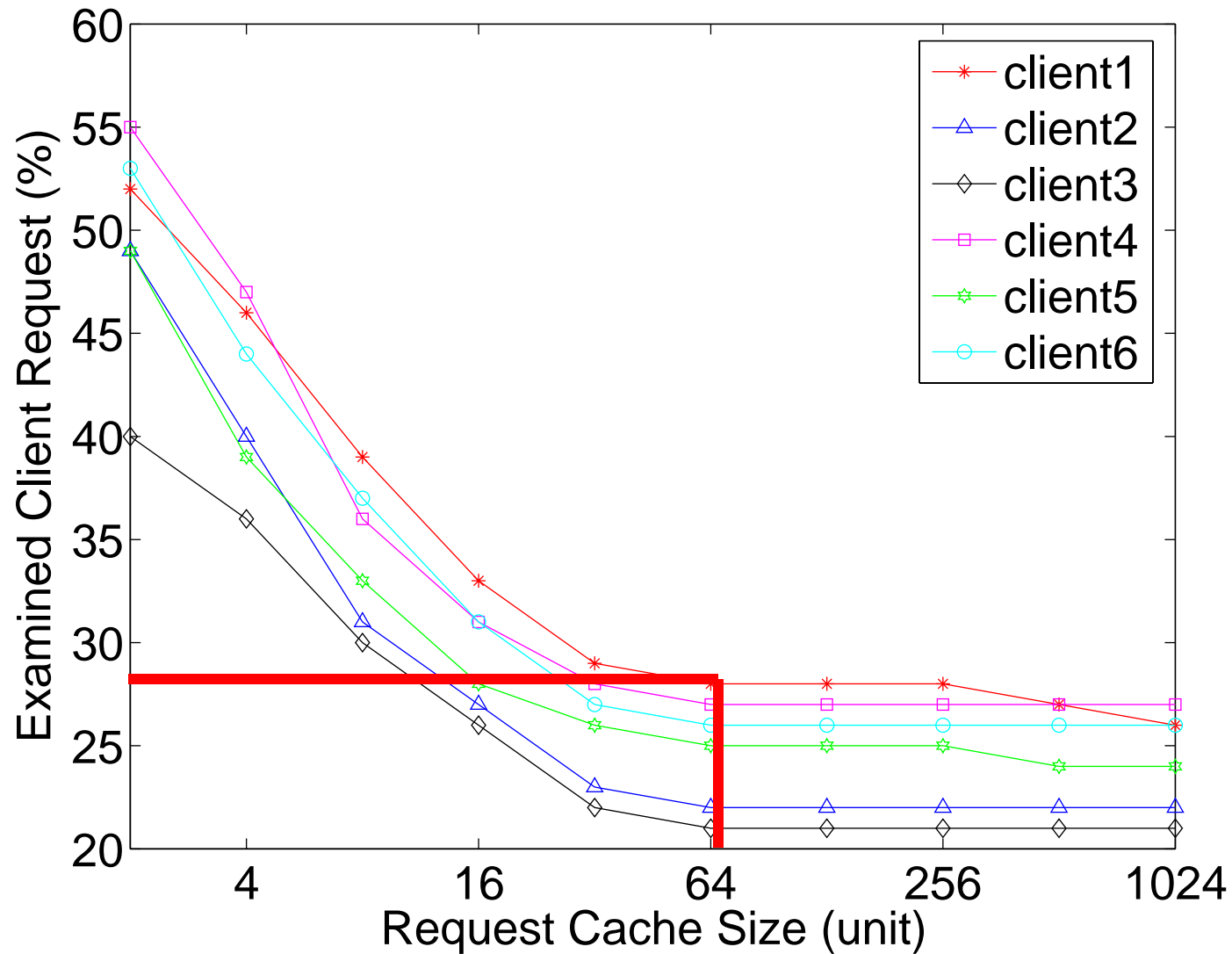
	Latency (ms)		Throughput (MB/s)	
	average	Std_dev	Average	Std_dev
Direct Access	0.681	0.0697	11.71335	0.0224
Via WT	396.992	19.6012	2.287216	0.0251
Web in VM	4.7423	0.0220	5.659181	0.0377
Splitter Process	26.898	0.1200	11.67471	0.0235

Cache Impacts on Applications

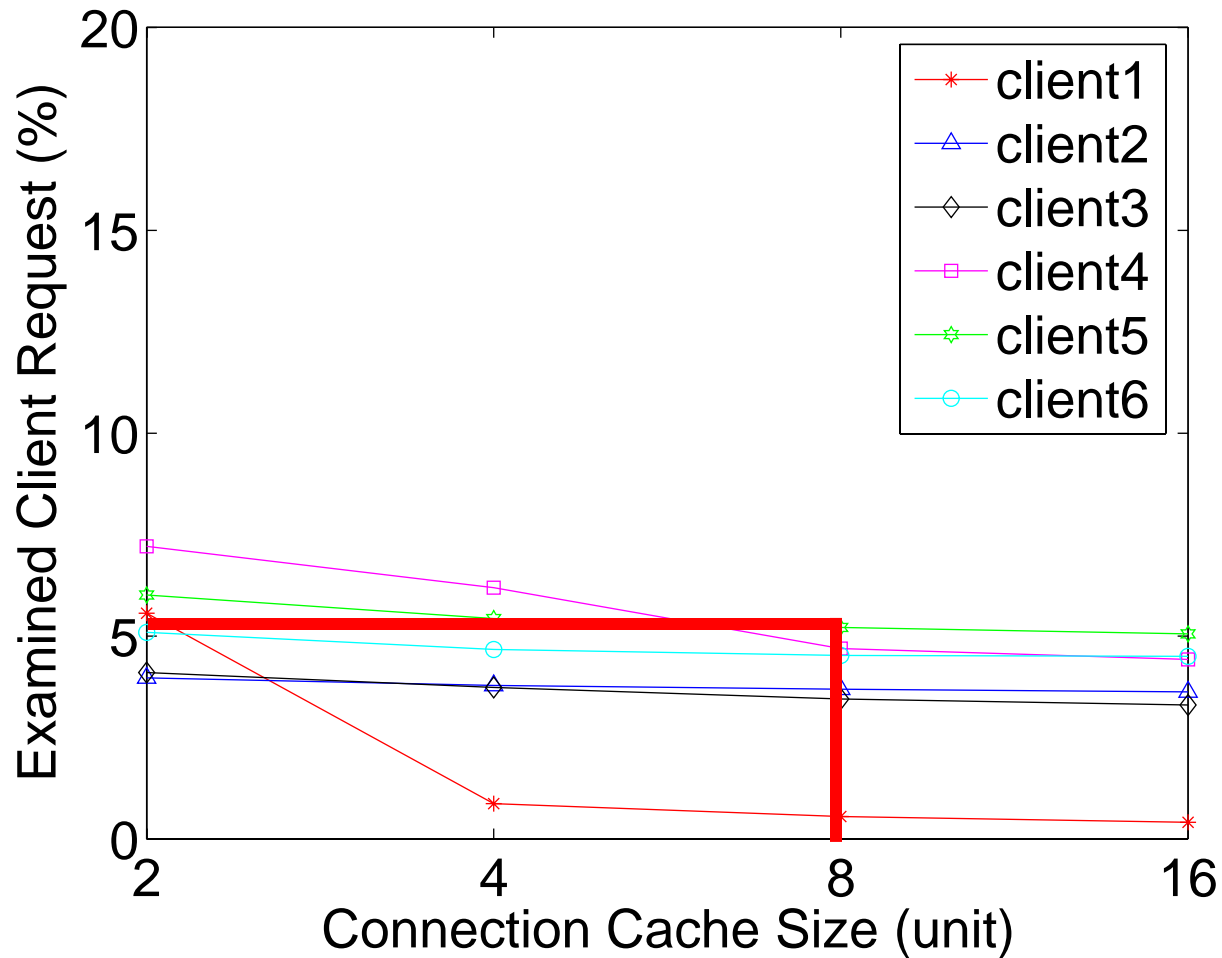
- Lab environment, 6 clients, browser log for 4 months
- LRU replacement in cache

	#reqs	#reqs (unique)	#cons (unique)
Client1	8318	2130	362
Client2	12852	2724	455
Client3	8921	1843	289
Client4	7809	2074	337
Client5	24793	5789	1119
Client6	8457	2179	381

Request Cache



Connection Cache



Outline

- Design Principles and Overview
- Design Issues & Implementation
- Evaluations
- Conclusions

Conclusions and Future Work

- We propose WormTerminator to **detect and contain all unknown/polymorphic** worms **without infecting any** other hosts
- We implemented and experimented on a **real** worm to demonstrate its feasibility
- We need to further improve its **performance**
 - Better virtual machine
 - Multi-core processor

Thanks
&
Questions?