Securing against self-propagating malicious software

529 Debate

Affirmative Team:

Adriana Flores and Clayton Shepard

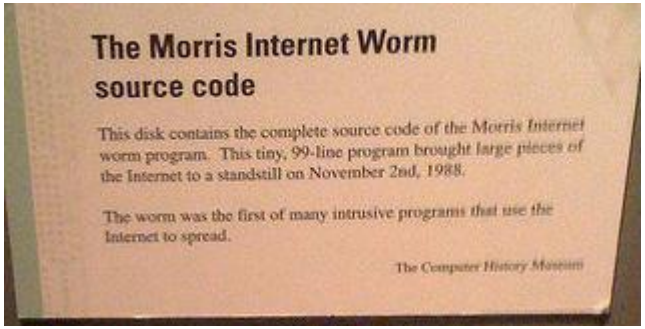
Background

- Worm= Self-replicating computer program that is self-contained and does not need to be part of another program to propagate itself.

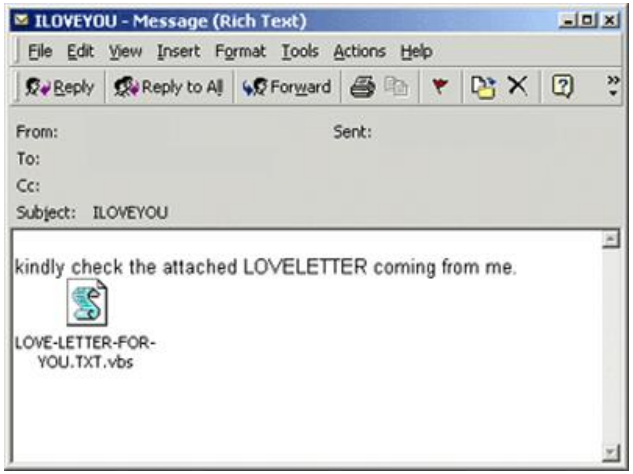
- 4 parts:
  - Vulnerability
  - Propagation Method
  - Payload
  - Goal

Worm History

- **Xerox PARC- John Shoch**
  - PhD - analysis of the traffic patterns
  - "Tapeworm" - The Shockwave Rider.
- **Christma Exec - German EARN network (1987)**
  - Spread to IBM Internal File Transfer Network via BITNET
- **Morris worm (1988) ~ $10 million to $100 million**
  - Exploited weak passwords along with known vulnerabilities
  - Re-infect individual servers multiple times
  - World's first Internet denial of service (DoS) attack.
  - Infected ~ 6,000 servers (10% of the servers on the Internet!)



Worm History



- **I LOVE YOU (2000) ~$8.75 billion**
  - Outlook e-mail address book
  - Malicious
  - Overwrite files, download Trojan horse
- **Code Red (2001) ~$2.6 billion**
  - Vulnerability in Microsoft's Internet Information Server (IIS)
  - Complete command line control
  - Delayed DoS attacks against the White House's IP address
  - 359,000 servers in just 14 hours, 2,000 servers per minute
- **Future?**

  More complex worms might incorporate sophisticated polymorphic, and metamorphic behavior routines that will make use of entry-point obscuration.

[1]

Problem

Self-propagating malware (worms) are able to compromise vulnerable end-hosts and leverage these to perpetuate themselves.

Is the solution secure hosts, or secure networks?

Prior Work: Secured hosts

- **Host-based architecture**
  - Vigilante [8] hosts protect themselves automatically by generating filters
  - Central service with heuristics to modify vulnerable source code [9]
- **Memory monitoring and management** [10-12]
  - Checks for illegal memory access
  - Check for overwrites and apply rollbacks
- **Filters**
  - IntroVit [13], vulnerability-specific predicates run inside VMs
  - Shield [14], manually deployed host-based filters to block vulnerabilities
  - Allow vulnerable services to continue execution while being attacked

Prior Work: "Secure" the network

- **Detect abnormal communication patterns**
  - Block or rate limit traffic
    - Williamson [2], Snort [3], Network Security Monitor [4]
  - Ineffective against worms with normal traffic patterns
    - E.g. topological worms and slow-spreading worms

      → **False negatives** and false positives
- **Content signatures**
  - Signatures for unknown worms, identify a common byte string in suspicious network flows [5,6,7]
  - Absence of information about software vulnerability

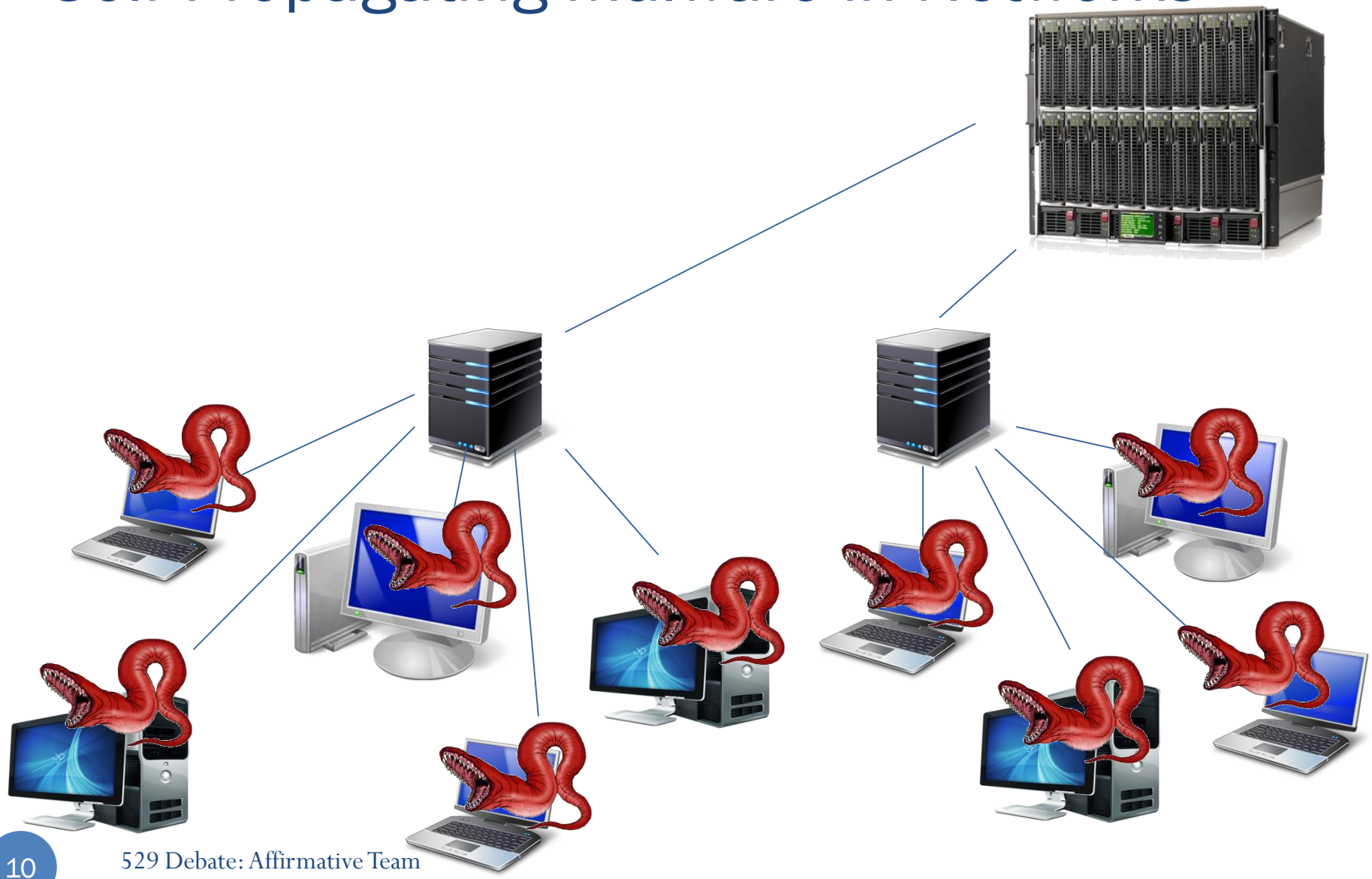
    → **False negatives** and false positives

Secure at end-host!

Purpose of Networks

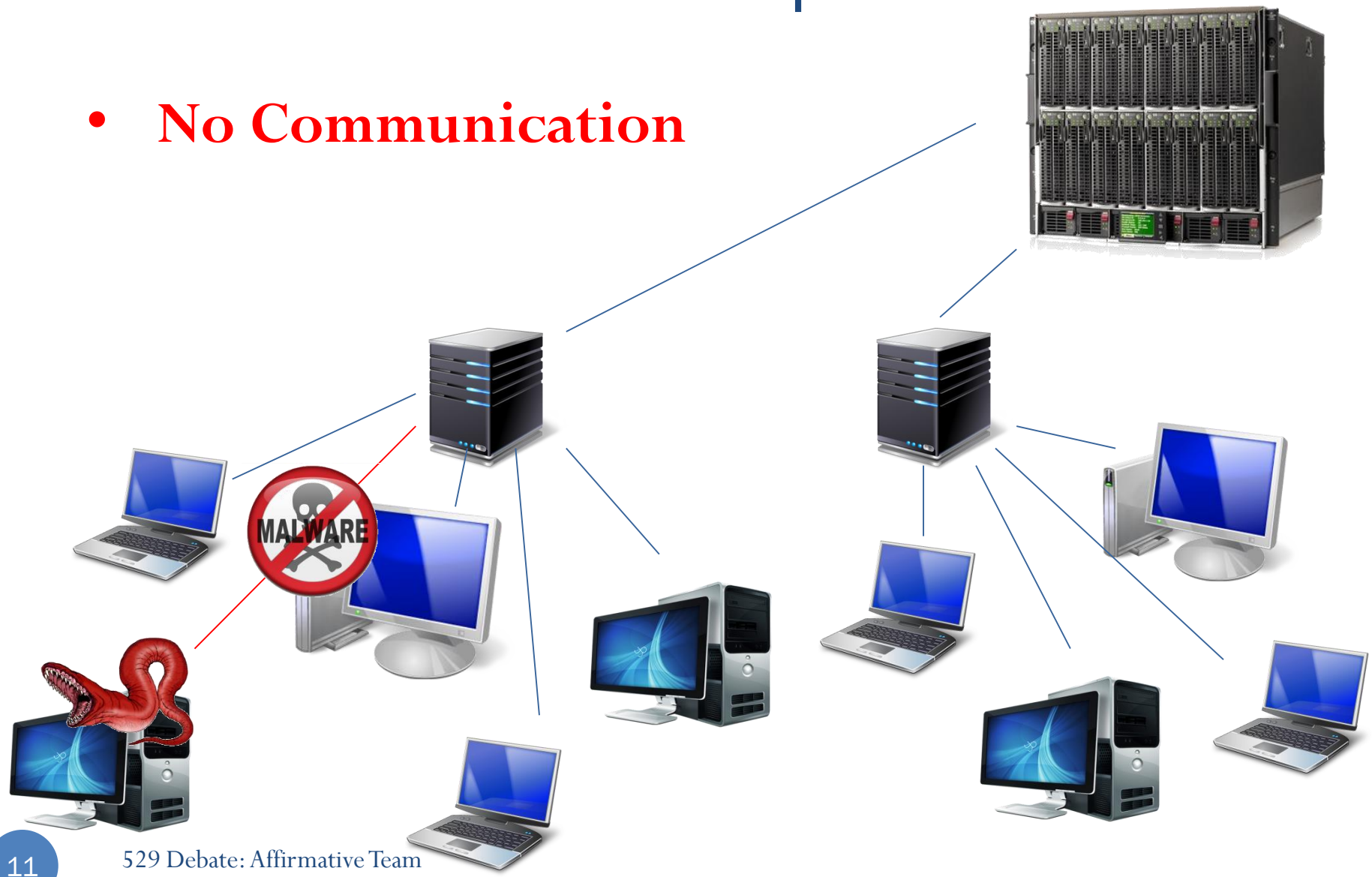
- Communication
- Access Control
- ~~Security~~

Self-Propagating Malware in Networks

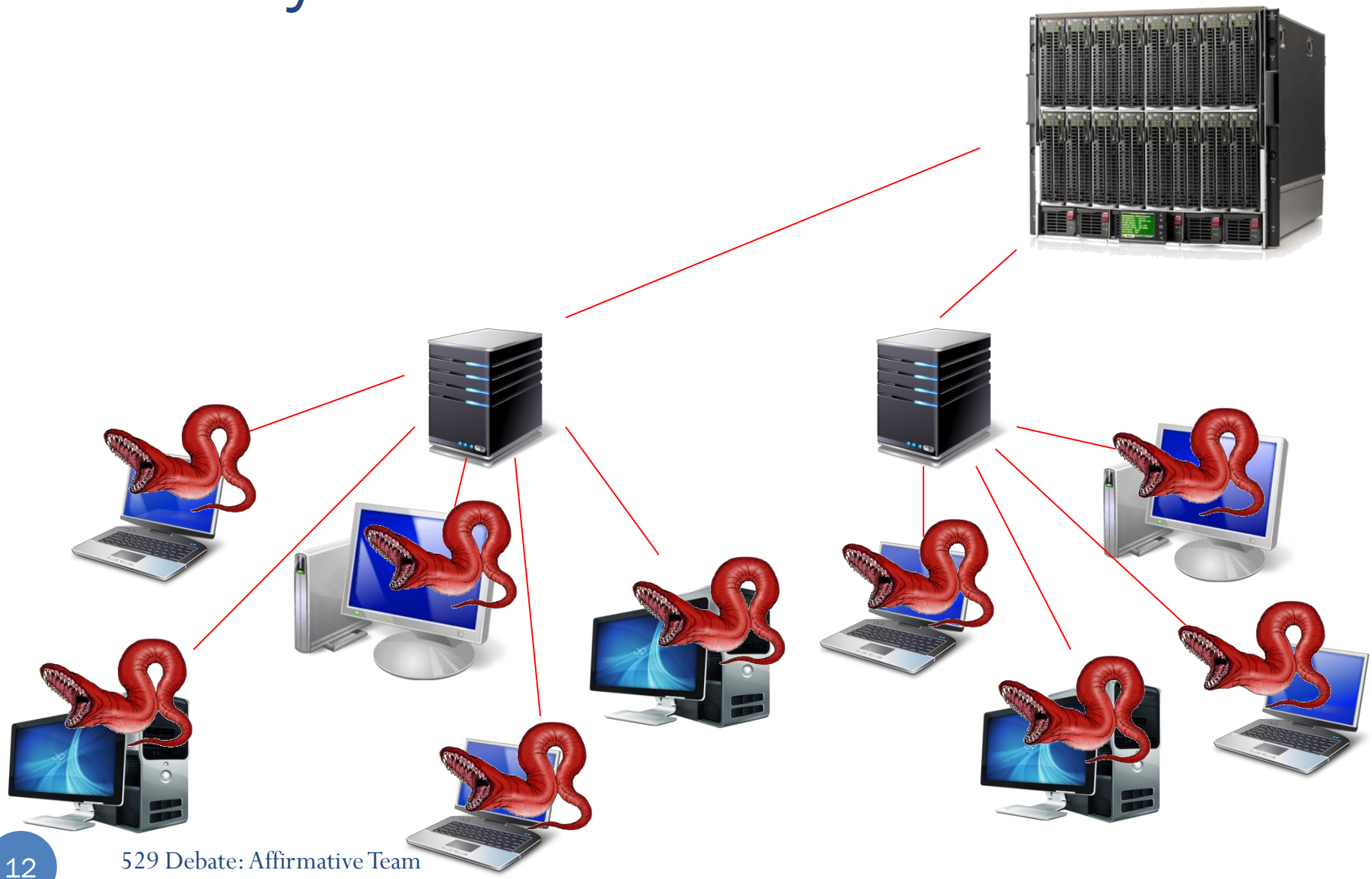


Defeated Network Purpose

- **No Communication**



Security Hole vs. Communication



A secure end-host can exist in any network without risk.

It is impossible for the network to preemptively block unknown malicious communication.

References

[1] Fosnock, Craig. "Computer worms: past, present, and future." East Carolina University 8 (2005).

[2] Williamson, Matthew M. "Throttling viruses: Restricting propagation to defeat malicious mobile code." Computer Security Applications Conference, 2002. Proceedings. 18th Annual. IEEE, 2002.

[3] Roesch, Martin. "Snort: Lightweight Intrusion Detection for Networks." LISA. Vol. 99. 1999.

[4] Heberlein, L. Todd, et al. "A network security monitor." Research in Security and Privacy, 1990. Proceedings., 1990 IEEE Computer Society Symposium on. IEEE, 1990.

[5] Kreibich, Christian, and Jon Crowcroft. "Honeycomb: creating intrusion detection signatures using honeypots." ACM SIGCOMM Computer Communication Review 34.1 (2004): 51-56.

[6] Kim, Hyang-Ah, and Brad Karp. "Autograph: Toward Automated, Distributed Worm Signature Detection." USENIX security symposium. Vol. 286. 2004.

[7] Singh, Sumeet, et al. "Automated Worm Fingerprinting." OSDI. Vol. 4. 2004.

References

[8] Costa, Manuel, et al. "Vigilante: End-to-end containment of internet worms." ACM SIGOPS Operating Systems Review. Vol. 39. No. 5. ACM, 2005.

[9] Sidiroglou, Stelios, and Angelos D. Keromytis. "Countering network worms through automatic patch generation." (2003).

[10] Sidiroglou, Stelios, et al. "Building a reactive immune system for software services." Proceedings of the general track, 2005 USENIX annual technical conference: April 10-15, 2005, Anaheim, CA, USA. USENIX, 2005.

[11] Rinard, Martin C., et al. "Enhancing Server Availability and Security Through Failure-Oblivious Computing." OSDI. Vol. 4. 2004.

[12] Smirnov, Alexey, and Tzi-cker Chiueh. "DIRA: Automatic Detection, Identification and Repair of Control-Hijacking Attacks." NDSS. 2005.

[13] Joshi, Ashlesha, et al. "Detecting past and present intrusions through vulnerability-specific predicates." ACM SIGOPS Operating Systems Review. Vol. 39. No. 5. ACM, 2005.

[14] Wang, Helen J., et al. "Shield: Vulnerability-driven network filters for preventing known vulnerability exploits." ACM SIGCOMM Computer Communication Review. Vol. 34. No. 4. ACM, 2004.

Conclusion

Summary

End-Host

- Secured end-hosts can exists in any network without risk of infection
- Vulnerabilities are known by end hosts
- Higher efficiency
- Guarantees, no false negatives e.g. Vigilante

Network

- Lack of information
  - know all the worms
  - Software vulnerabilities
  - Identify every legitimate traffic flow
  - Trends do not provide guarantees
  - **False Negatives & Positives**
  - **Less effective**
- Impossible to make a network that blocks all worms without blocking all communication
  - Communication vs. Security

A secure end-host can exist in any network without risk.

It is impossible for the network to preemptively block unknown malicious communication.

Using the network to secure against worms is analogous to cutting off a limb after it has become gangrenous.