Fast Detection of Scanning Worm Infections

Jaeyeon Jung
Arthur W. Berger
MIT CSAIL

Stuart E. Schechter
~~Harvard DEAS~~
MIT Lincoln Laboratory

This work is sponsored by the Department of Defense under the Air Force Contract F19628-00-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.

Scanning worms are abundant

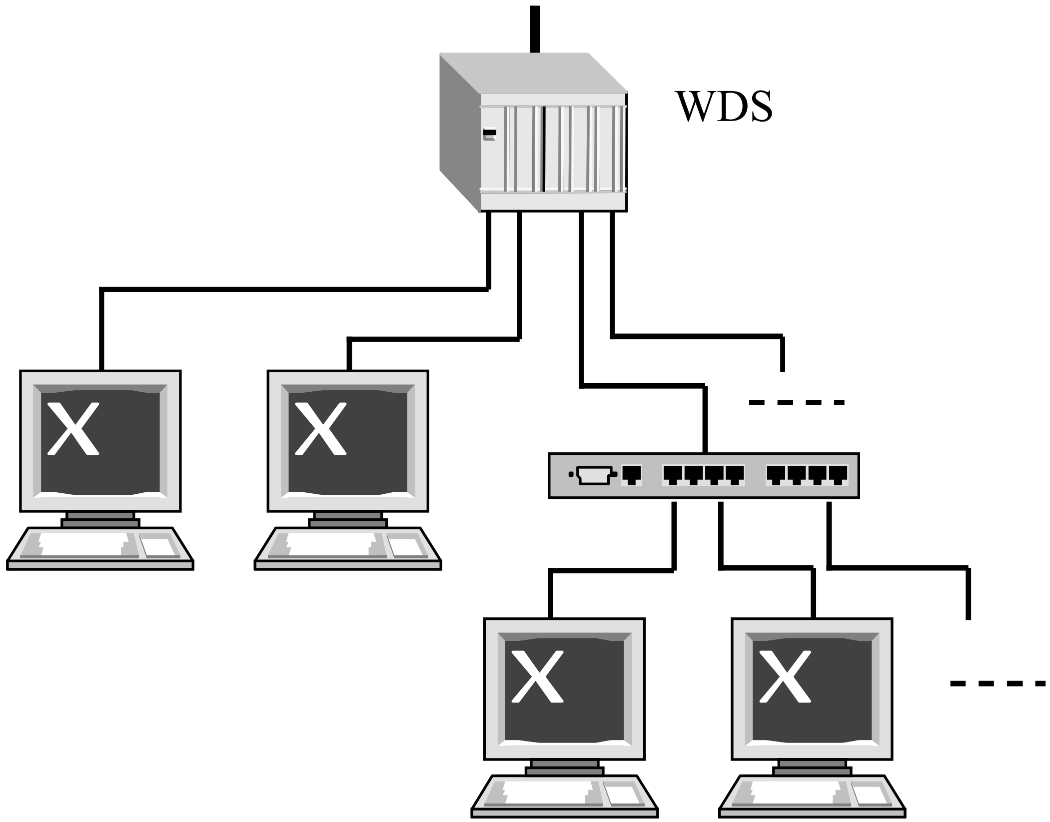
- **Easy to write**
  - **Select target IP is simple…**
  - **Pick at random: (**`Slammer, CodeRed`)
  - **Step through IP space: (**`Blaster`)
  - **Favor local addresses: (**`CodeRed II, Nimda`)
- **Very fast**
  - `Slammer` **– 90% of vulnerable hosts in 10 minutes**
- **Require automated detection/response**

Perimeter defense not enough

- Firewalls are porous
  - Hybrid worms enter as email viruses
  - Portable devices enter/leave network
- Once inside perimeter, worms spread freely
- Infected hosts must be
  - Quarantined…
  - Reliably detected

Stuart E. Schechter
12/17/2004

Worm Detection Systems needed



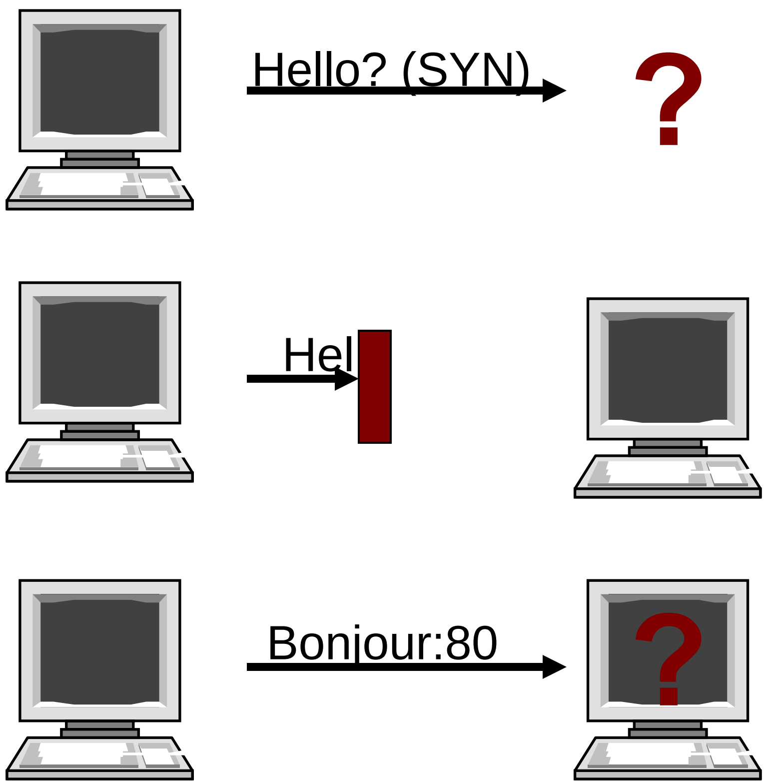
Approaches to detection

- **Worm signatures**
  - Too slow to generate & deploy
- **Fixed connection rate limits** [Williamson *et al.* 03]
  - Worms can scan at rate just below limit
  - False positives from crawlers, mailers
- **Fixed connection failure limits**
  - Require many observations before raising alarms
  - False positives from web crawlers, mailers
- **Connection success/failure ratio** [Jung *et al.* 04]
  - Only applied to detect remote scanners

Outline

- Prior work: sequential hypothesis testing
- Two-pronged approach to worm detection
  - Definitively detecting infection events
  - Limiting spread of infection before detection
- Results
- Current limitations & future work

Sequential hypothesis testing:
Scan connections usually fail



Target address may be invalid
(no host at address)

Target may not accept packet from sender (firewall)

Target may not run service
(no listener on port)

Sequential hypothesis testing:
Terminology

- **A first-contact connection (FCC) request is the first packet (TCP or UDP) sent between two distinct hosts**

- ***Y* is a sequence of outgoing first-contact connection observations** $(Y_1, Y_2, \ldots, Y_i, \ldots, Y_n)$

- $$Y_i = \begin{cases} S \quad (0) & \text{if the connection succeeds} \\ F \quad (1) & \text{if the connection fails} \end{cases}$$

- **Example connection sequence (benign host)**

Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7	Y_8
S	S	F	S	S	S	S	S

Sequential hypothesis testing: Key assumption

Worm's scan connections less likely to succeed

$$\Pr\left[S \mid H_{\text{scanning}}\right] < \Pr\left[S \mid H_{\text{benign}}\right]$$

(or worm's scan connections more likely to fail)

$$\Pr\left[F \mid H_{\text{scanning}}\right] > \Pr\left[F \mid H_{\text{benign}}\right]$$

Sequential hypothesis testing:
Event likelihoods compared as ratios

$$\phi(S)=\frac{\Pr\left[S\mid H_{\text{scanning}}\right]}{\Pr\left[S\mid H_{\text{benign}}\right]}<1$$

$$\phi(F)=\frac{\Pr\left[F\mid H_{\text{scanning}}\right]}{\Pr\left[F\mid H_{\text{benign}}\right]}>1$$

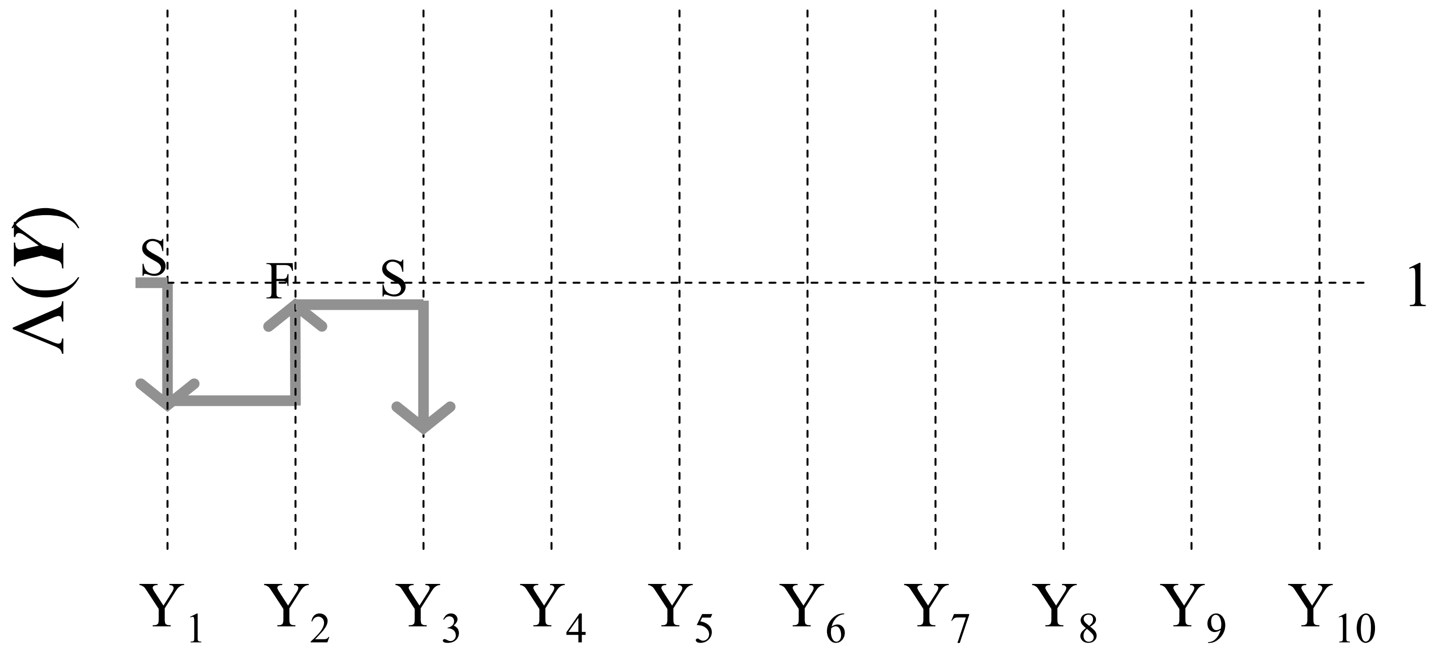
Sequential hypothesis testing:
Sequence likelihood ratios

$$\phi(Y_i) = \frac{\Pr\left[Y_i \mid H_{\text{scanning}}\right]}{\Pr\left[Y_i \mid H_{\text{benign}}\right]}$$

- IID assumption
- Lambda is likelihood ratio for sequence

$$\Lambda(\boldsymbol{Y}) = \prod_{i=1}^{n} \frac{\Pr\left[Y_i \mid H_{\text{scanning}}\right]}{\Pr\left[Y_i \mid H_{\text{benign}}\right]} = \prod_{i=1}^{n} \phi(Y_i)$$

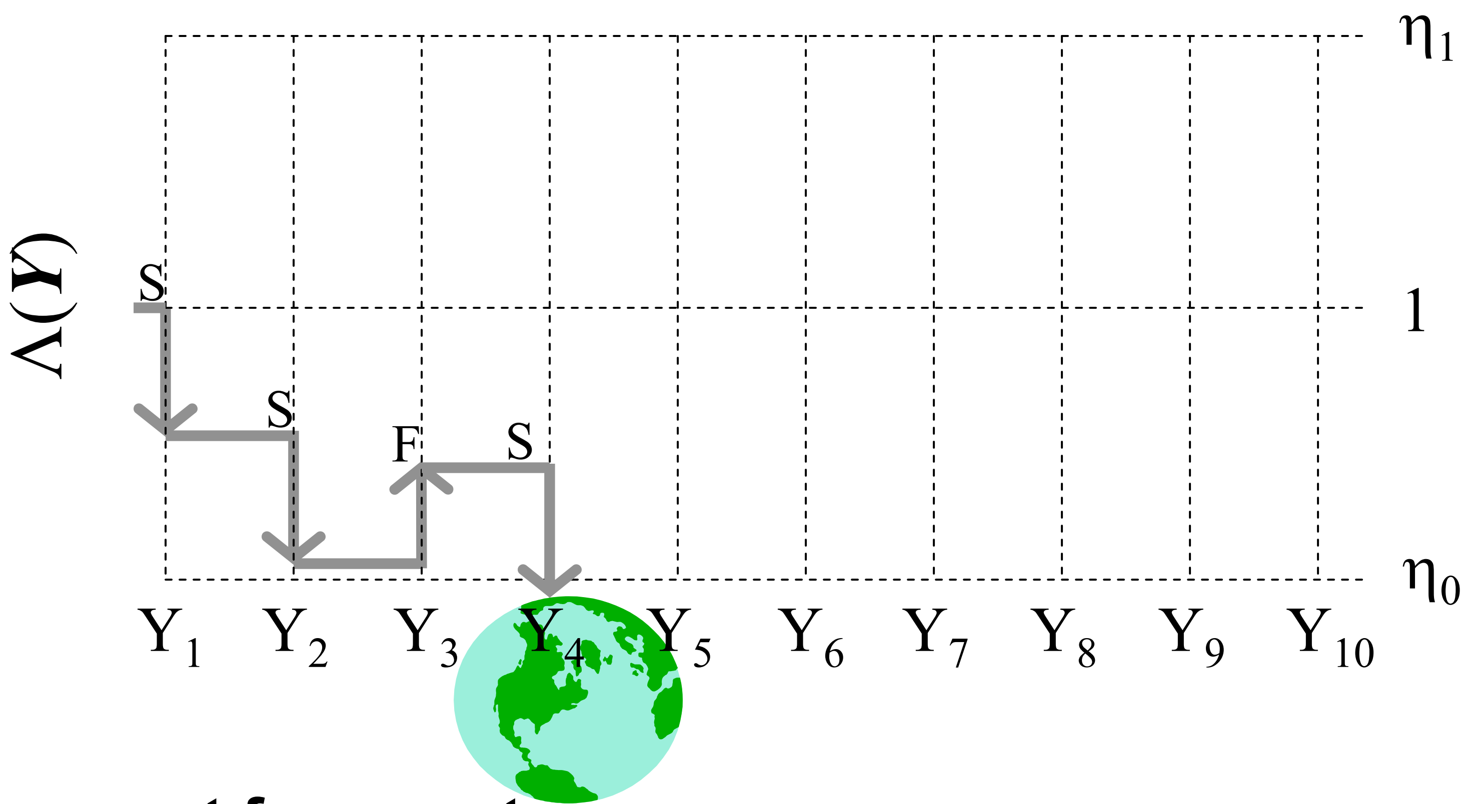
Sequential hypothesis testing:
Graphing the likelihood ratio



$$\Lambda(\boldsymbol{Y}) = \prod_{i=1}^{n} \phi(Y_i) = 1 \times \phi(S) \times \phi(F) \times \phi(S)$$

$$\log \Lambda(\boldsymbol{Y}) = \sum_{i=1}^{n} \log \phi(Y_i) = 0 + \log\ \phi(S) + \log\ \phi(F) + \log\ \phi(S)$$

Sequential hypothesis testing:
Testing for scanners

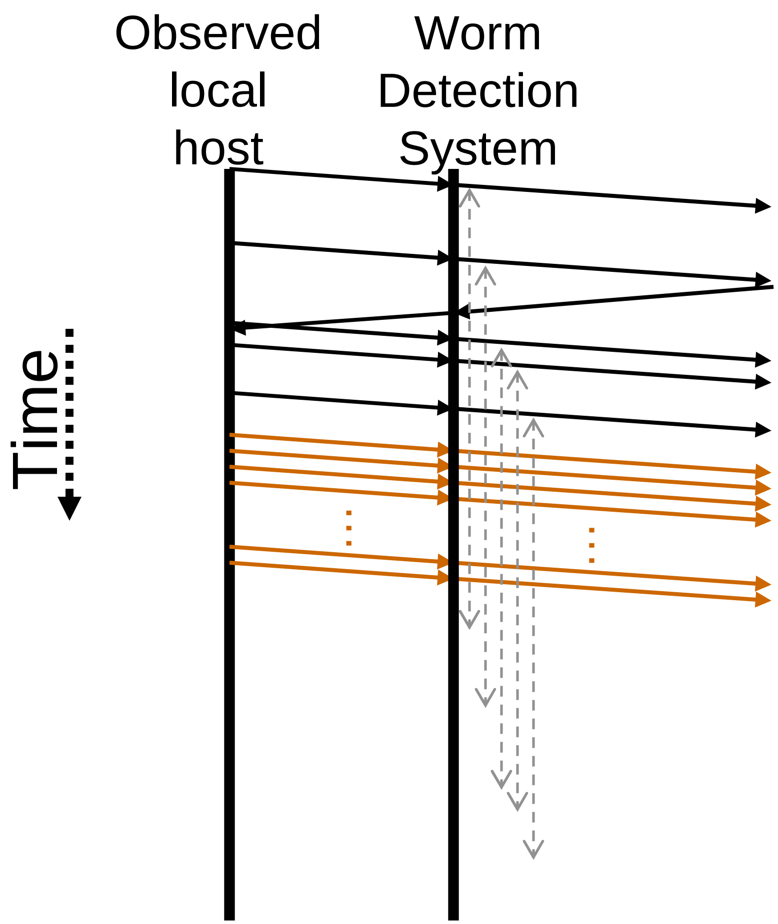


Works great for remote scanners.

Why not for detecting worms on local hosts?

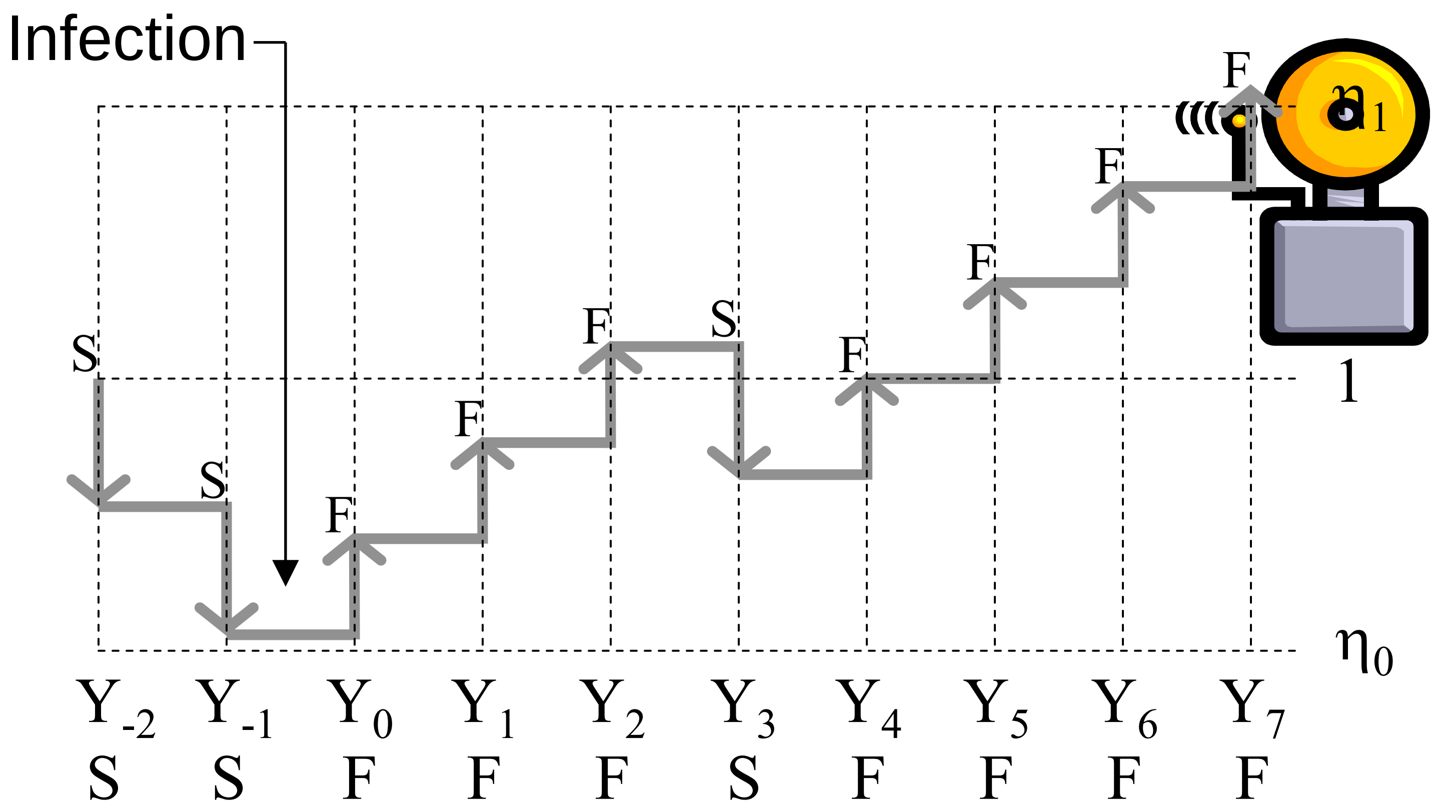
Problems:

Timeout needed to detect failures



Y_1	Y_2	Y_3	Y_4	Y_5
F	S	F	F	F

Problems:
Infections may occur during test



Outline

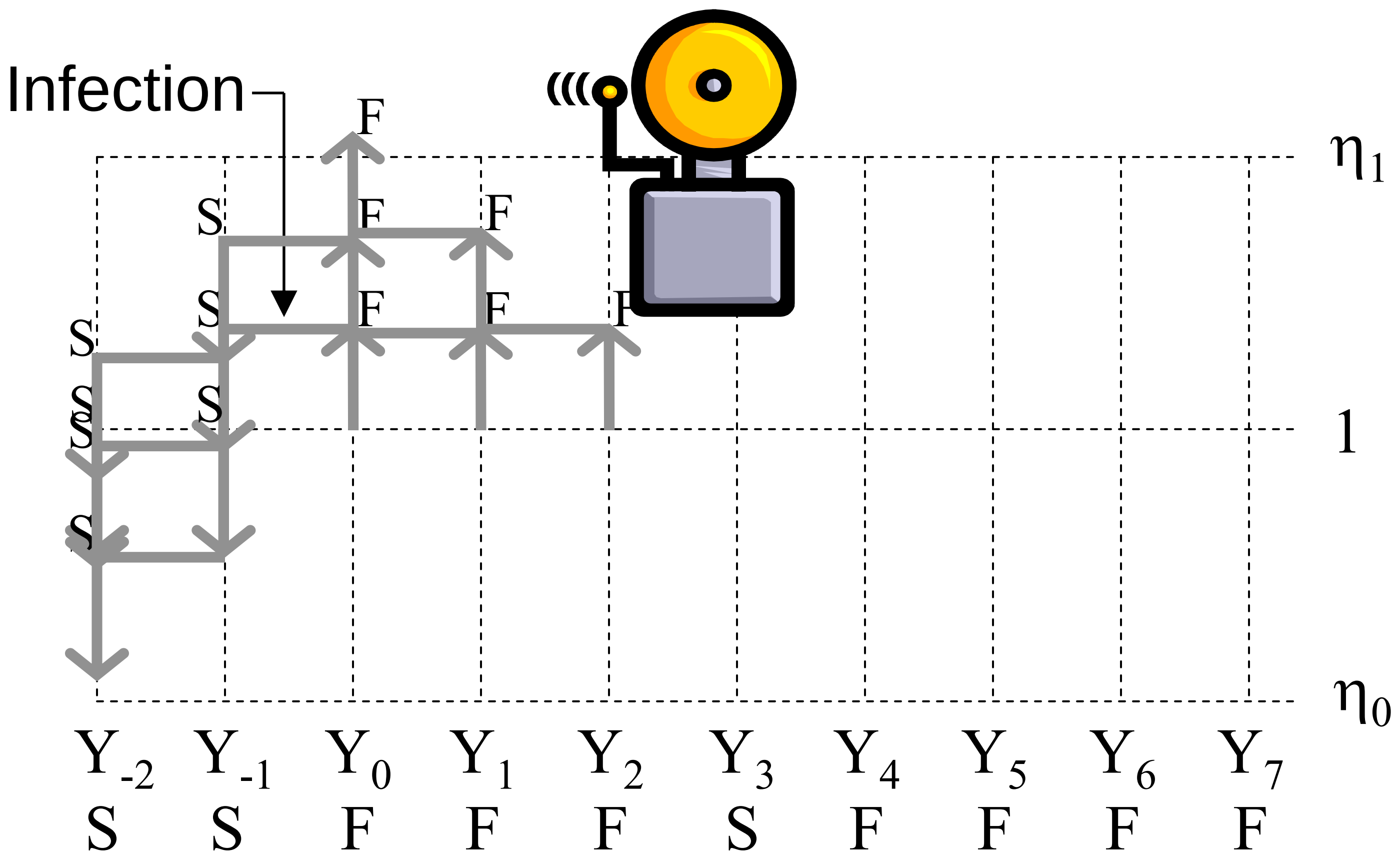
- Prior work: sequential hypothesis testing
- Two-pronged approach to worm detection
  - Definitively detecting infection events
  - Limiting spread of infection before detection
- Results
- Current limitations & future work

Reverse Seq. Hypothesis Testing

As each observation arrives…

- Run test in reverse chronological order
  - Most recent observed connections first
  - Try to conclude before processing pre-infection observations
- Termination conditions:
  - Either threshold exceeded
  - No more observations to process

Detecting infection events: Reverse Seq. Hypothesis Testing



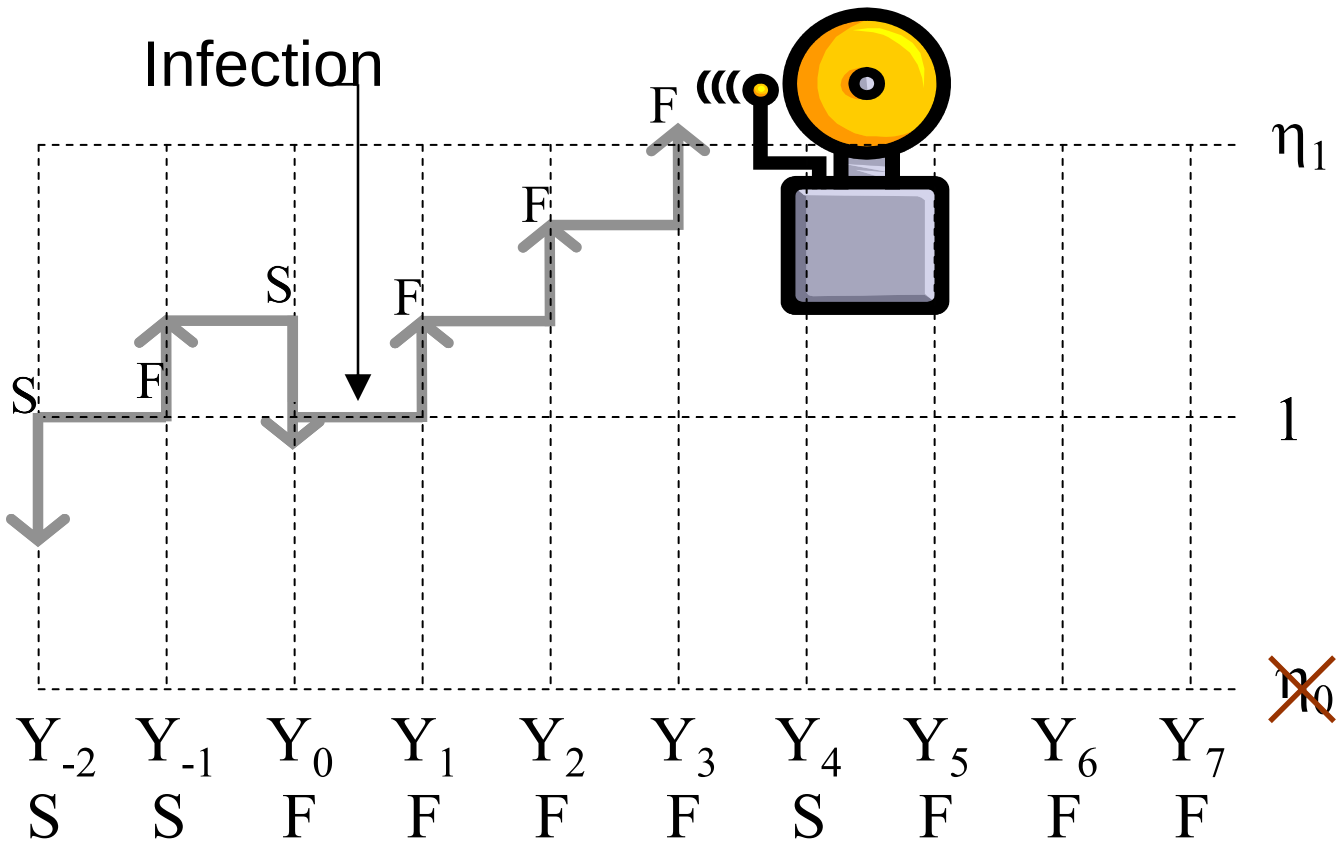
Detecting infection events: Cost for naïve implementation

As described, algorithm requires:

- One test per observation
- Multiple iterations per test
- Must keep history of past observations

Detecting infection events:
An optimization

$$\overline{\Lambda}(\mathbf{Y}_n) = \max\left(1, \overline{\Lambda}(\mathbf{Y}_{n-1}) \times \phi(Y_i)\right)$$



Stuart E. Schechter
12/17/2004

Implementation

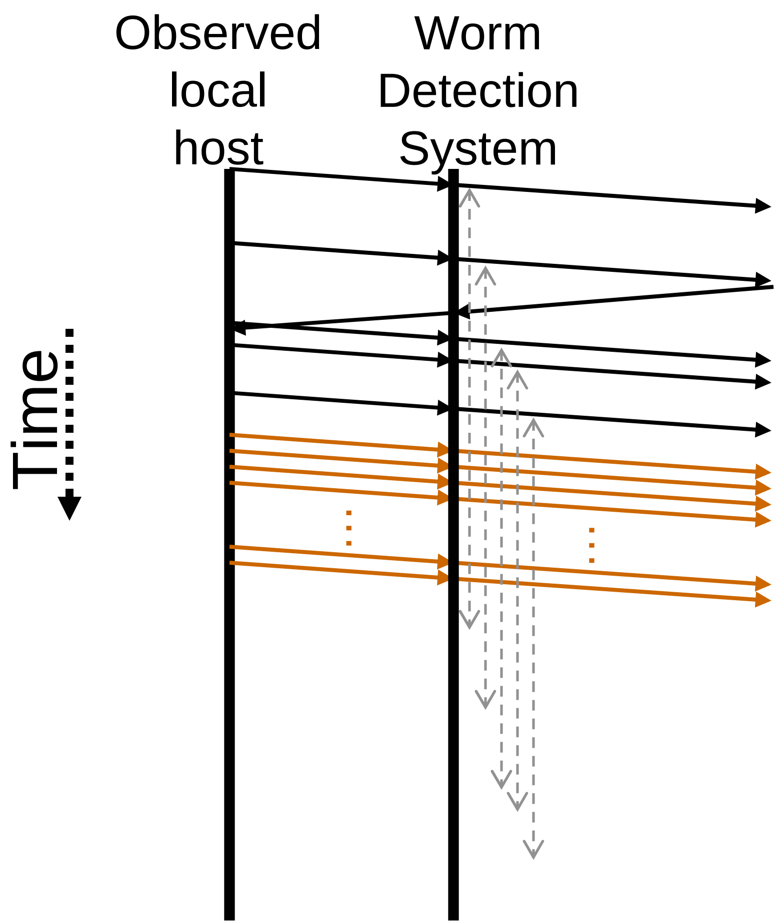
- **First-contact connection approximation**
  - Kept list of 64 most recently contacted hosts
  - FCC is any packet sent to host not on list
- **FCC success rate constants**
  - Scanners = 10%, Benign = 70%
- **Hypothesis test constraints**
  - 0.00005 false positives per FCC (per test)
  - 0.99 chance of detection if infected (per test)

    Detection threshold will be hit before benign threshold

Outline

- Prior work: sequential hypothesis testing
- Two-pronged approach to worm detection
  - Detecting infection events
  - Limiting spread of infection before detection
- Results
- Current limitations & future work

Timeout needed to detect failures



Stuart E. Schechter
12/17/2004

Limiting infection spread before detection:

Credit-based connection rate limiting

Each local host i given starting balance (C_i = ~~10~~)

Issuing an FCC costs i a credit

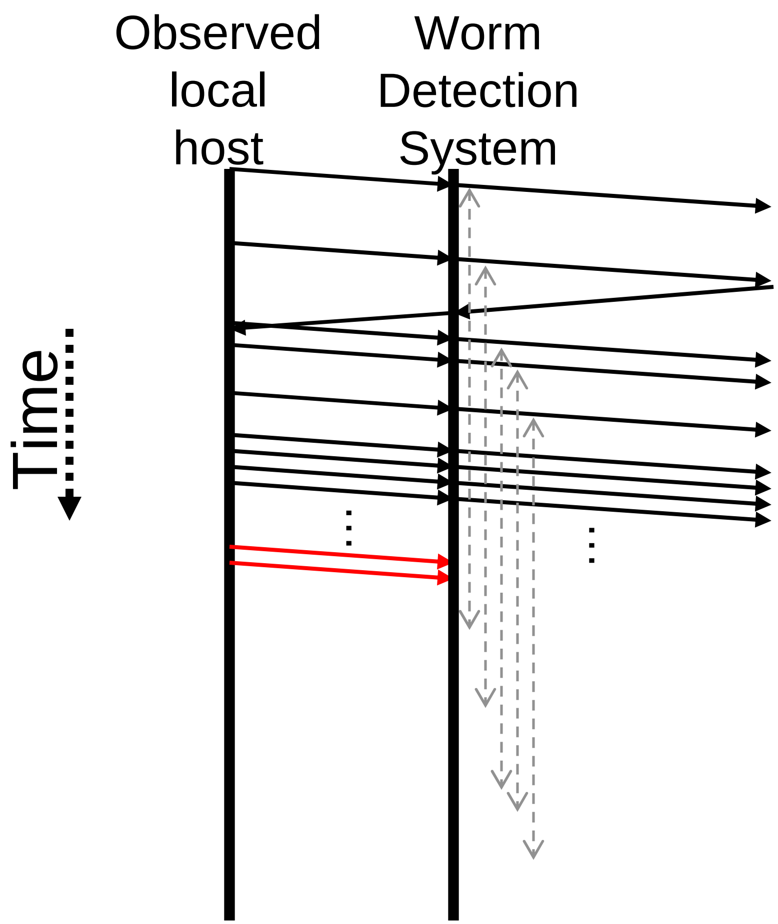
$$C_i = C_i + \log \phi(F)$$

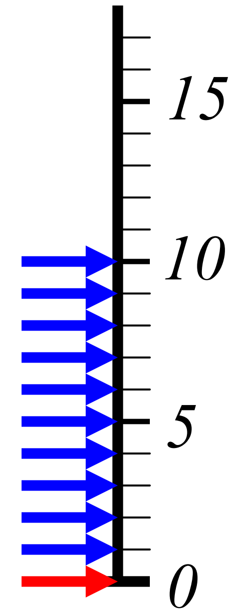
Drop request if $C_i \leq$ ~~$0, C_i = C_i - 1$~~ otherwise

When FCC succeeds i gets two credits ~~($C_i = C_i + 2$)~~

$$C_i = C_i - \log \phi(F) + \log \phi(S)$$

Limiting infection spread before detection:
CBCRL in action





Credit-based connection rate limiting

- To prevent build-up of large credit balances
  - Simulate inflation each second

$$C_i = \max\left(10, \frac{2}{3} C_i\right) \text{ if } C_i > 10$$

  - Hosts with perfect success rate will have twice as many credits as they needed in previous second.
- To prevent starvation
  - Hosts bankrupt for four seconds receive one credit

Outline

- Prior work: sequential hypothesis testing
- Two-pronged approach to worm detection
  - Definitively detecting infection events
  - Limiting spread of infection before detection
- Results
- Current limitations & future work

Results:

Data sets

	isp-03	isp-04
When collected	1:14 PM April 10, 2003	1:36 PM January 28, 2004
Duration	627 minutes	66 minutes
Total outbound connection attempts	1,402,178	178,518
Total active local hosts	404	451

Results:
Reverse seq. hypothesis testing

	isp-03	isp-04
Worms/Scanners detected	**5**	**6**
`CodeRed II`	2	0
`Blaster`	0	1
`MyDoom*`	0	3
`Minmail.j*`	0	1
`HTTP (other)`	3	1
False alarms	**0**	**6**
`HTTP`	0	3
`SMTP`	0	3
P2P	**6**	**11**
Total	**11**	**23**

Results:
Credit–based connection rate limiting

- **No unnecessary rate limiting**
  - Dropped only connections from hosts later deemed to be scanners by hypothesis test
  - Didn't allow any connections to escape reverse sequential hypothesis testing

Why not just use CBCRL alone?

False negatives…

Connection issued before infection received after infection and scan begins could delay detection

Outline

- Prior work: sequential hypothesis testing
- Two-pronged approach to worm detection
  - Definitively detecting infection events
  - Limiting spread of infection before detection
- Results
- Current limitations & future work

Limitations and next steps

- Denial of service attack
  - Create web page with 500 image references to random addresses
  - Host that browses page will be quarantined (perhaps for good reason)
- Enable user to deactivate HTTP quarantine (reverse Turing test)

Limitations and next steps

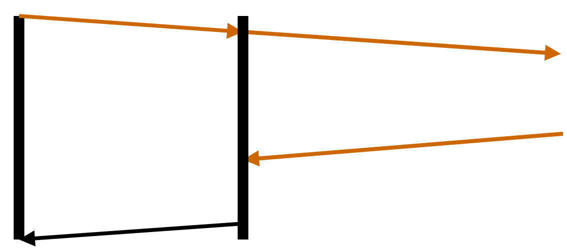
- Known-replier attack
  - Worms interleave lists of known hosts with scans
  - Attack is easier if list of previously known host list stored in limited buffer
  - May interleave requests to commonly used ports
- Forged response attack
  - Partner on outside forges responses to hide failures
- Run two tests, (local->local, local->remote)
  - Use sparse IP space internally (NAT)

Test on host/service pairs

- Perform separate tests for each unique local host/destination port pair
  - Enables different thresholds for different services
  - Prevents known-replier attack using services not targeted by the worm
- Integrate new host event observations
  - Connection rate increases
  - New services contacted (e.g. SMTP)
  - Recently contact by host now deemed infected

Bringing approaches together

- Merge rate limiting approach into rev. sequential hypothesis testing
  - Assume connections failed until proven otherwise, remove quarantine if proven innocent *(similar to Weaver, Staniford, Paxson @ USENIX Sec)*
  - Allow bankrupt host to send TCP SYNs…



Conclusion

- **Reverse seq. hypothesis test detects infection events**
  - **Number of observations required to reach conclusion is adjusted with strength of evidence**
- **CBCRL eliminates risk of infection while waiting for connections to fail (time-out)**
- **Worms contained within network**

Acknowledgements

- Dave Anderson
- Hari Balakrishnan
- Kim Hazelwood Cettei
- Rob Cunningham
- Glenn Holloway
- Vern Paxson
- Mike Smith

Limitations & next steps

- Not all first-contact connections requests independent
  – Many may contact the same network
  – Networks may go down
- Remove IID assumption
  – Likelihood of failure greater if connection sent to network where last connection failed
  – Hypothesis test should account for this

Future work:
Detecting topological worms

- Topological worms
  - Worm uses info on host to locate targets
  - May search cache, history, configuration files
  - E.g. SSH known_hosts

Limiting infection spread before detection:
Previous Approaches

- Virus throttle [Twycross & Williamson ’03]
  - Working set of up to 5 destination addresses
  - Queue new connection requests if
    1. working set is full
    2. destination address not in working set
  - Each second

    remove LRU destination address from working set

    add first destination address in queue to working set

    send all pending connection requests to that address

- Limits FCC rate to one request/second

Limiting infection spread before detection:
Previous Approaches

- **Limitations of virus throttles**
  - Legitimate high rate FCC traffic throttled
    - Web crawlers
    - Mailers
  - Rate limits should automatically adapt to needs of legitimate traffic
- **Virus throttle reports infection when queue length ≥ 100**
  - Low scanning rate worms never detected

Results:
Comparison to virus throttling

	isp-03	isp-04
Worms/Scanners detected	**3**	**2**
`CodeRed II`	2	0
`Blaster`	0	~~1~~ 0
`MyDoom*`	0	~~3~~ 1
`Minmail.j*`	0	~~1~~ 0
`HTTP (other)`	~~3~~ 1	1
False alarms	**0**	**0**
`HTTP`	0	~~3~~ 0
`SMTP`	0	~~3~~ 0
P2P	**~~6~~ 2**	**~~11~~ 3**
Total	**5**	**5**

Results:
Credit–based connection rate limiting

- **No unnecessary rate limiting**
  - CBCRL only dropped connections from hosts later deemed to be scanners by hypothesis test
- **In contrast, virus throttling**
  - Rate limited 84 of 404 hosts in isp-03
  - Rate limited 59 of 451 hosts in isp-04
  - Performed poorly despite generous definition of rate limiting (queue length > 5)

Sequential hypothesis testing:
Reaching a conclusion

Conclusion reached when threshold exceeded

- **Scanning:** $\Lambda(Y) > \eta_1$

$$\eta_1 = \frac{\text{minimum desired detection rate}}{\text{maximum desired false positive rate}}$$

- **Benign:** $\Lambda(Y) < \eta_0$

$$\eta_0 = \frac{1 - (\text{minimum desired detection rate})}{1 - (\text{maximum desired false positive rate})}$$

Algorithmic cost: Optimized

- **New function run in forward sequence**

$$\overline{\Lambda}(\mathbf{Y}_n) = \max\left(1, \overline{\Lambda}(\mathbf{Y}_{n-1}) \times \phi(Y_i)\right)$$

- **Exceeds infection threshold if and only if reverse sequential hypothesis would**
- **Observations processed in forward order, then thrown out**
- **One calculation per observation**
  - **Three operations (1 addition, 2 comparisons)**

Stuart E. Schechter
12/17/2004

Limiting infection spread before detection: Credit-based connection rate limiting

- Each local host i given starting balance
  - $C_i = 10$
- Issuing an FCC costs i credit
  - Drop request if $C_i \leq 0$
  - $C_i = C_i - 1$ otherwise
- When FCC succeeds i gets two credits
  - $C_i = C_i + 2$