Defending Computer Networks
Lecture 8: More Port Scanning and Scanning Worms

Stuart Staniford

Adjunct Professor of Computer Science

Logistics

- HW1 back in CMS
- HW2 hopefully out next time
- Quiz next time
  - Half hour
  - No notes/laptops/tablets/phones/etc

New Assigned Reading

- Staniford et al *Practical Automated Detection of Stealthy Portscans* http://webpages.cs.luc.edu/~pld/courses/intrusion/fall05/hoagland_spade.pdf
  - Through section 3.1
- Staniford et al *How to Own the Internet in Your Spare Time*. http://www.icir.org/vern/papers/cdc-usenix-sec02/
- Falliere et al. *W32.Stuxnet Dossier*. http://www2.gwu.edu/~nsarchiv/NSAEBB/NSAEBB424/docs/Cyber-044.pdf

Latest News

SECURITY 9/16/2014 @ 9:26AM | 4,146 views

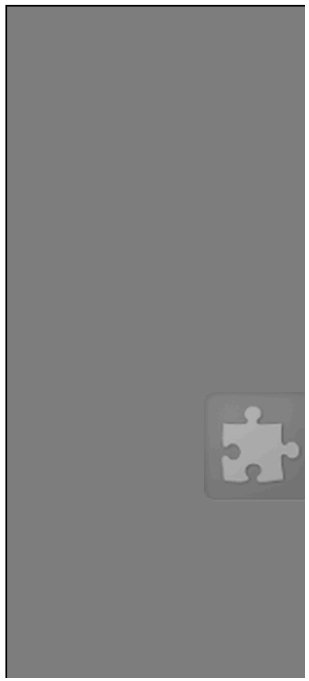
Widespread Android Vulnerability 'A Privacy Disaster', Claim Researchers

+ Comment Now + Follow Comments

Right at the start of September, security researcher Rafay Baloch released details on an Android bug that has now been called a "privacy disaster".

That apparently hyperbolic statement doesn't look too far wide of the mark, given anyone not running the latest release, Android 4.4, is affected. That means as many as 75 per cent of Android devices and millions of users could be open to attack, according to Google's own stats, though not all are likely to be using the affected Android Open Source Platform (AOSP) Browser.

The nature of the bug has worried onlookers too. The flaw could allow a bypass of the Same Origin Policy (SOP) protection used by most modern browsers. Crucially, the SOP protection stops malicious code from spilling over from one site to others open on separate tabs.



http://www.forbes.com/sites/thomasbrewster/2014/09/16/widespread-android-vulnerability-a-privacy-disaster-claim-researchers/

INFOWORLD TECH WATCH
By Woody Leonhard | Follow

Microsoft pulls botched KB 2982385 patch, MS 14-055 for Lync 2010

In stark contrast to the demise of KB 2889866 last week -- a pulled patch that went very well, thanks to Office Sustained Engineering folks who were really on the ball -- the yanking of KB 2982385 was slow and painful. Nearly a week after the problem was first identified, Microsoft finally pulled the patch and posted a note about the problem in a backwater FAQ, leaving the KB 2982385 article hanging with a singularly unhelpful "Oops! The page you are looking for may have a new location, or is no longer available" notification, and with no explanation or mitigation instructions.

The patch itself was a disaster from the get-go.

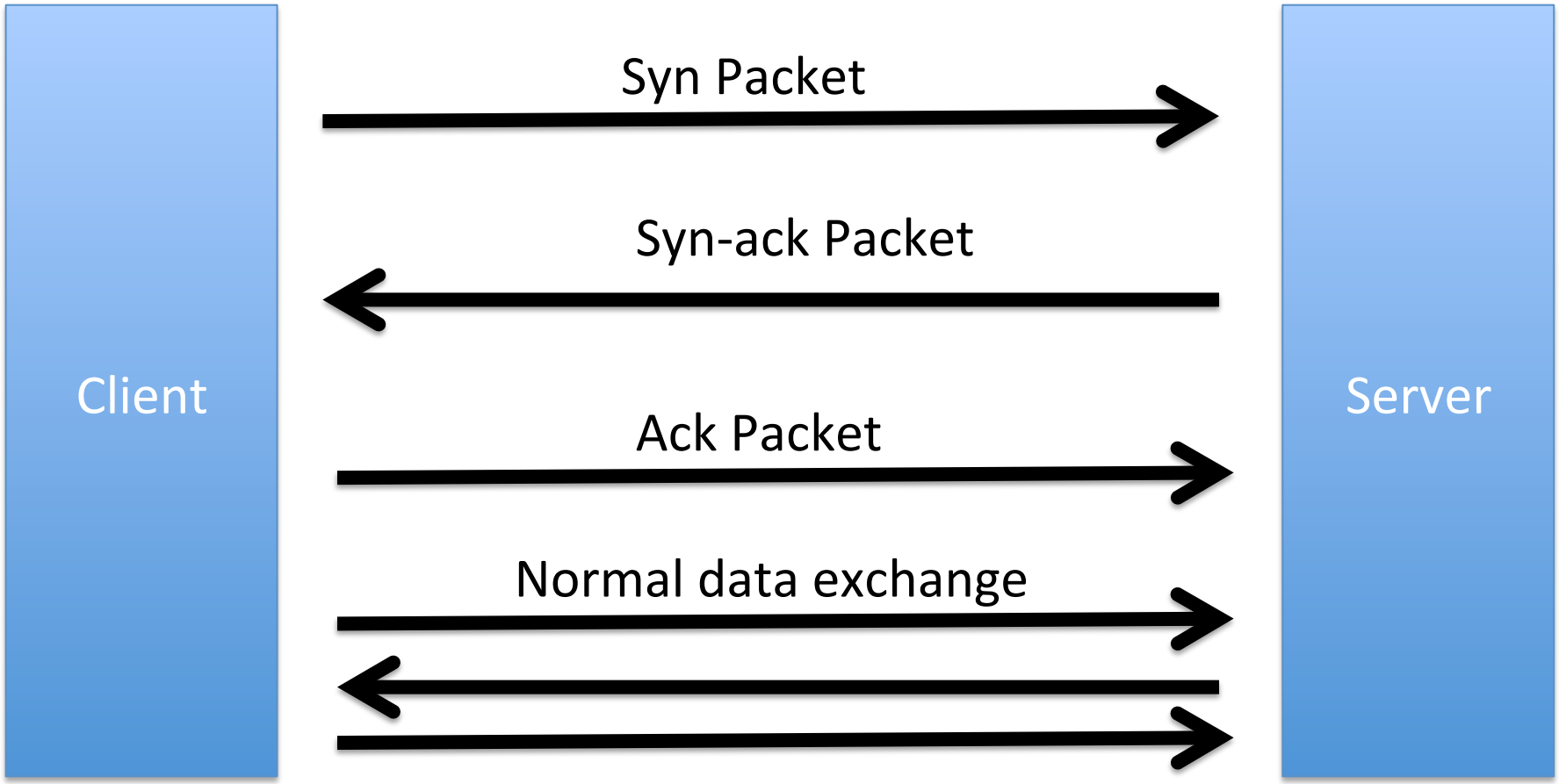
MS 14-055 is an "Important" patch for Lync Server 2010 and 2013 that "resolves three privately reported vulnerabilities in Microsoft Lync Server." In the case of Lync Server 2010, the Response Group Service was exposed to a possible denial-of-service attack. The patch for the server component itself has no listed security impact, "however, as a defense-in-depth measure, Microsoft recommends that customers of this software apply this security update to help protect against any possible new attack vectors identified in the future."

Tobie Fysh at Freebridge Community Housing tweeted about the problem on Sept. 12. Apparently the installer for KB 2982385 throws off a Windows Security message that says, "Windows can't verify the publisher of this driver software .... The driver software you're attempting to install does not have a valid digital signature that verifies who published it, and could potentially be malicious software." Fysh submitted a support case to Microsoft.

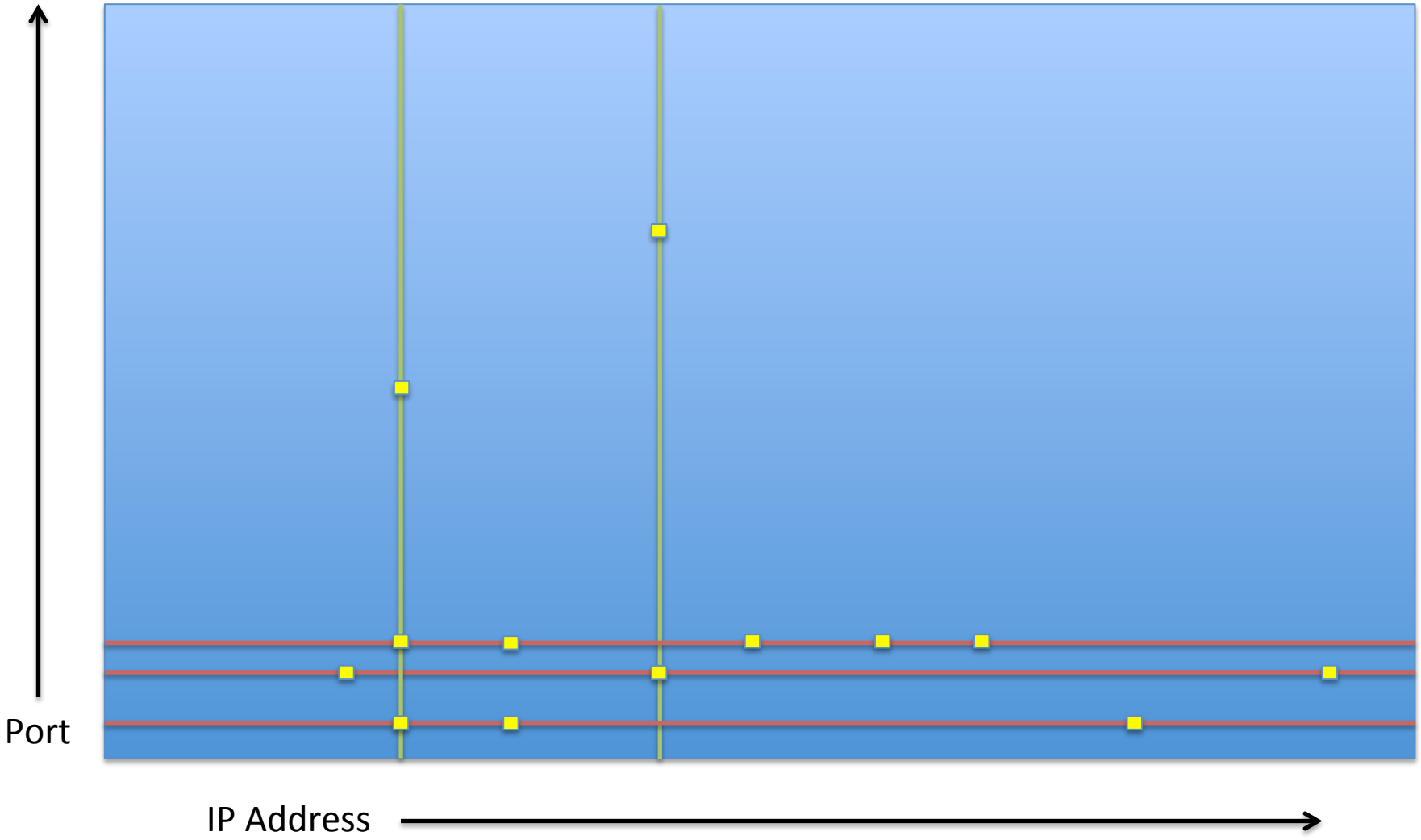
Main Goals for Today

- Finish up portscanning/detection
- Scanning worms

Refresh: 3-way handshake

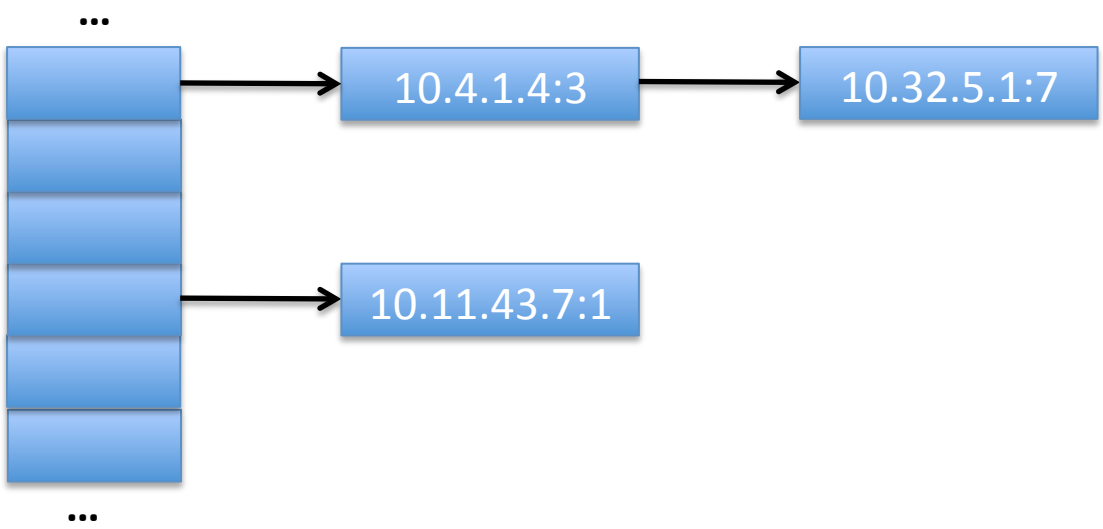


Visualizing Scans



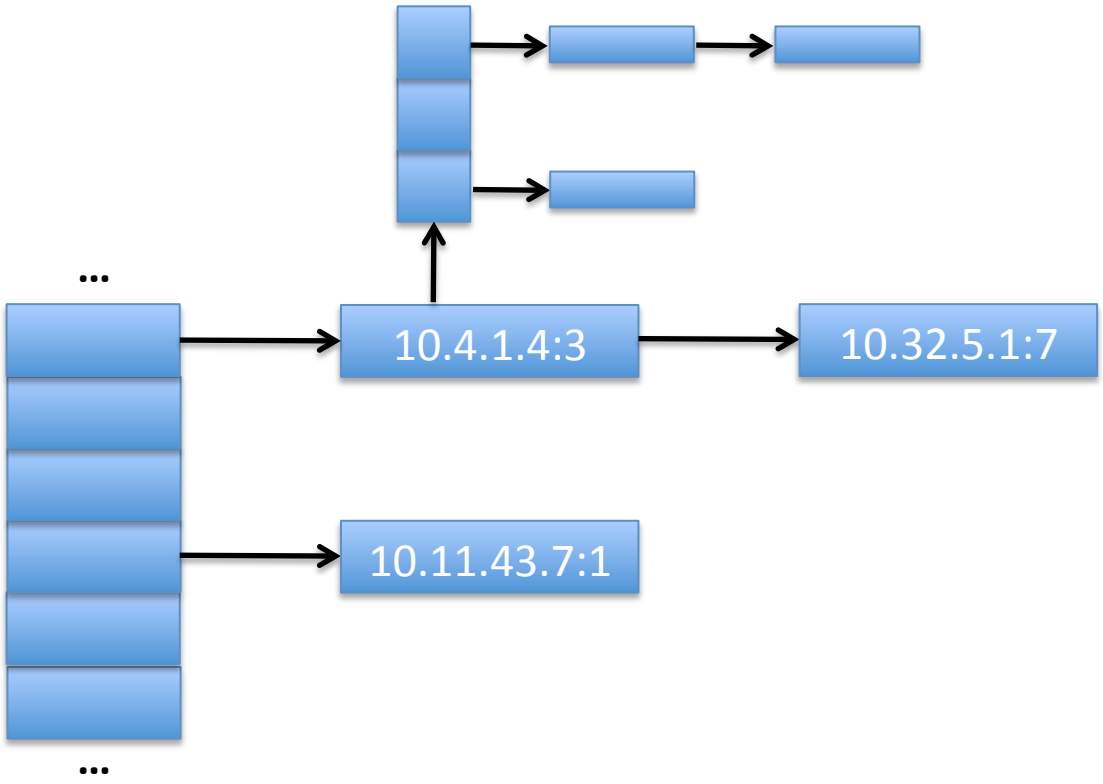
Then we need a data structure

- Simplest possible thing is a hash table
  - keyed on client IP
  - With per-connection counts of relevant stuff
  - Eg just count syns
  - Portscanners will issue more syns than average.
    - Alert when count goes over threshold
    - But what's likely to go wrong?



Keep track of unique dests/src?

- Now have to have a way to know
  - what is a unique dst for that src?



Better Idea

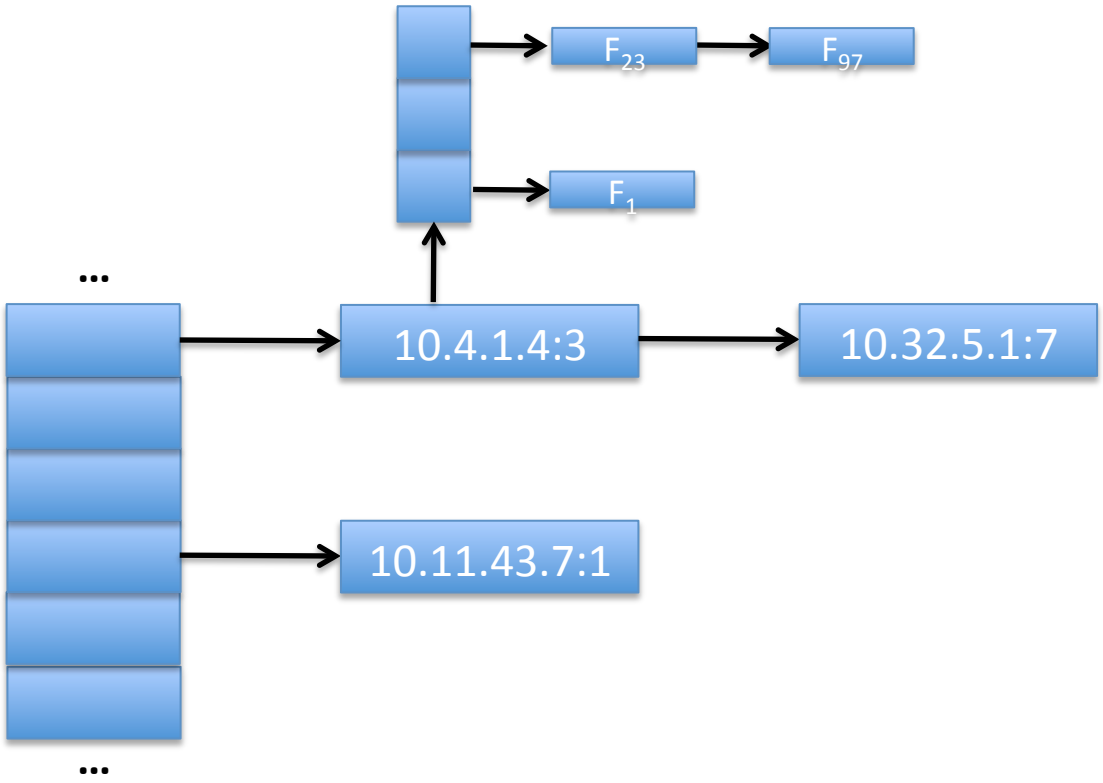
- Key off the idea that port-scanners make a lot of failed connections.
- Legit users make only a few
  - So keep track of "failed-succeed" count
  - Alert when goes over threshold.
- How can the attacker game this?
- Doesn't work in the presence of packet-filter/ firewalls.

Another Idea

- Learn the probability of a syn (say) being to a destination:
  - P(D)
  - Popular servers will have high P(D) (say 5% or 1%)
  - Non-servers will have very low P(D) (1 in 10^6 or 10^9)
  - Take –log(P(D)) and accumulate *that* in hash table
    - Anomaly score
  - Portscanners will accumulate a lot of anomaly score
    - Alert if over a threshold
  - Harder for attackers to game – don't know P(D)
    - Otherwise wouldn't need to portscan

Extending the basic idea

- Keep flow table state
- Know when we see things like unexpected F
- Give that a high anomaly score



Definitions

- Worm: A malicious program capable of infecting and running on other systems across the network via security weaknesses in that system in order to spread.
- Virus: Malicious code capable of infecting and attaching itself to other executables in order to spread.
  - Was more important in the PC/floppy drive era.

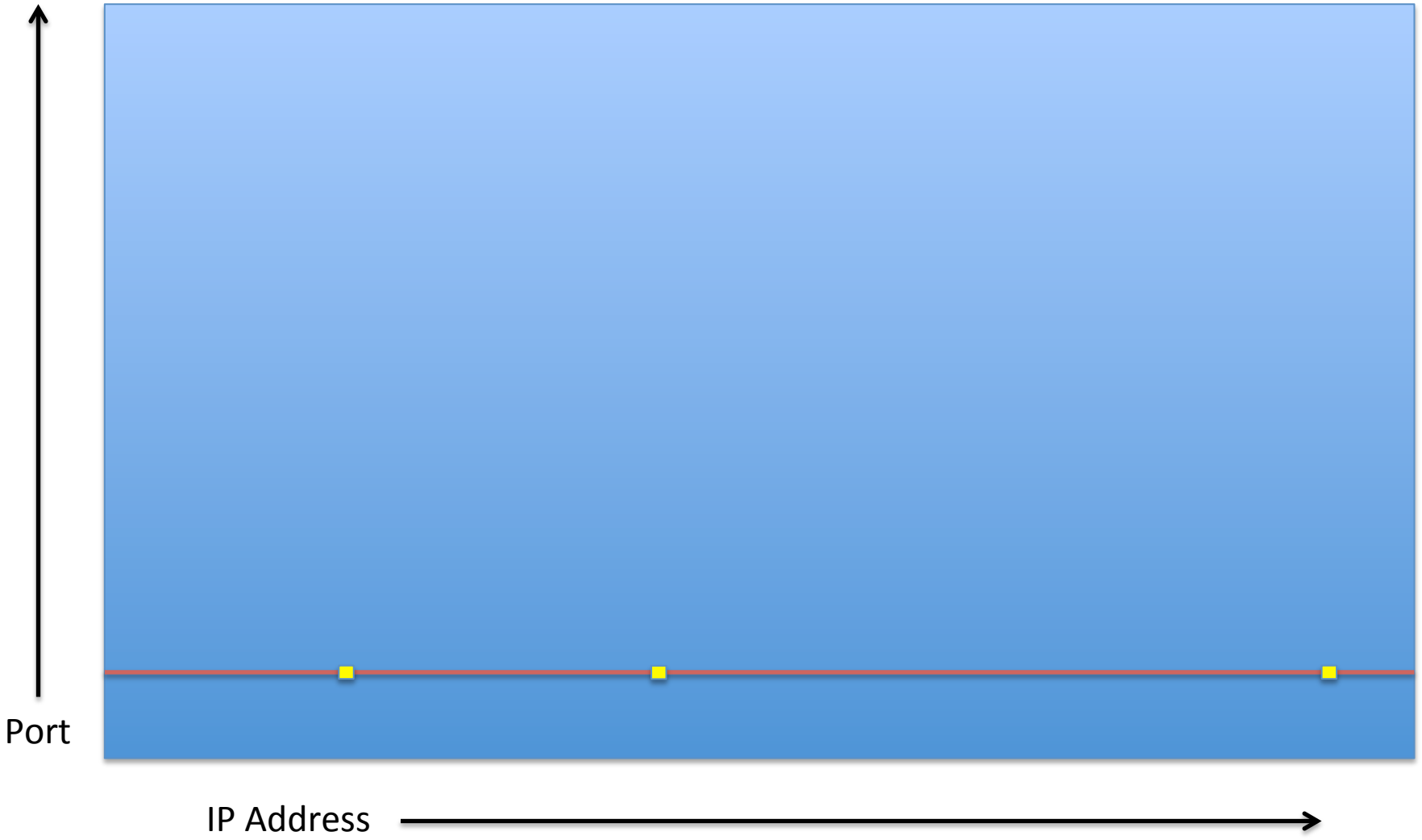
Ancient History

- 1988 Internet Worm (first serious worm problem)
  - Infected BSD Unix systems
  - Robert Morris Jr – Cornell student at the time.
  - Exploited vulnerabilities
    - Stack overflow in finger
    - Unauthenticated debug functionality in sendmail
    - guessing weak passwords.
  - Looked on host for info about other hosts.
    - Topological worm.
  - "disrupted normal activities and Internet connectivity for many days"
    - http://spaf.cerias.purdue.edu/tech-reps/823.pdf

Worms

- If we can scan,…
- And we have an exploit for a server port
- We have the basis for a worm!

Random Scanning Worms



Scanning Worms Big in the early 2000s

- 2001 Code Red (I and II), Nimda
- 2003 Slammer and Blaster/Welchia
- 2004 Sasser
- 2008 Conficker

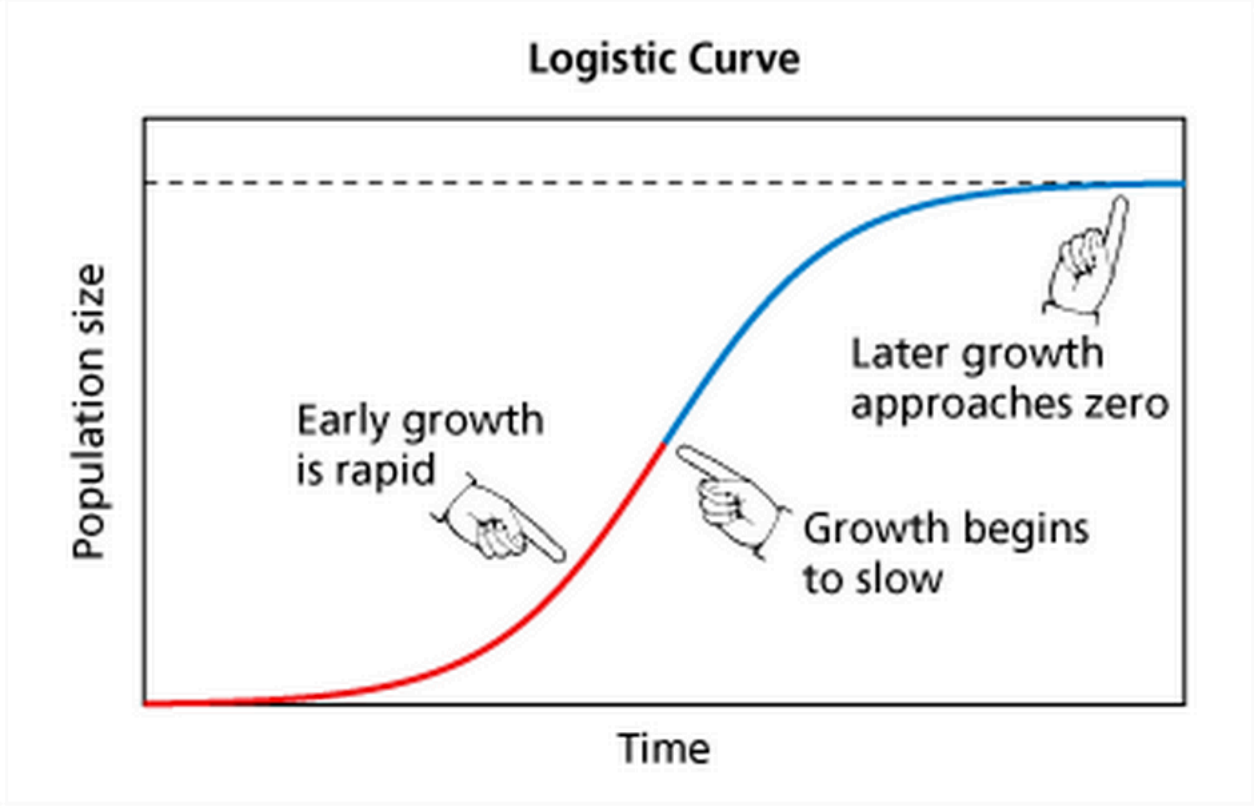
Let's Do Some Calculus

- For a whole-Internet random scanning worm
  - Let v be fraction of addresses that are vulnerable to worm exploit
  - Let S be average scanning rate (syns/hr).
  - Then K = Sv is number of new compromises/hr/ already infected machine.
  - $N = 2^{32}v$ is the total number of machines
  - Fraction of machines comprised at time t is a(t)

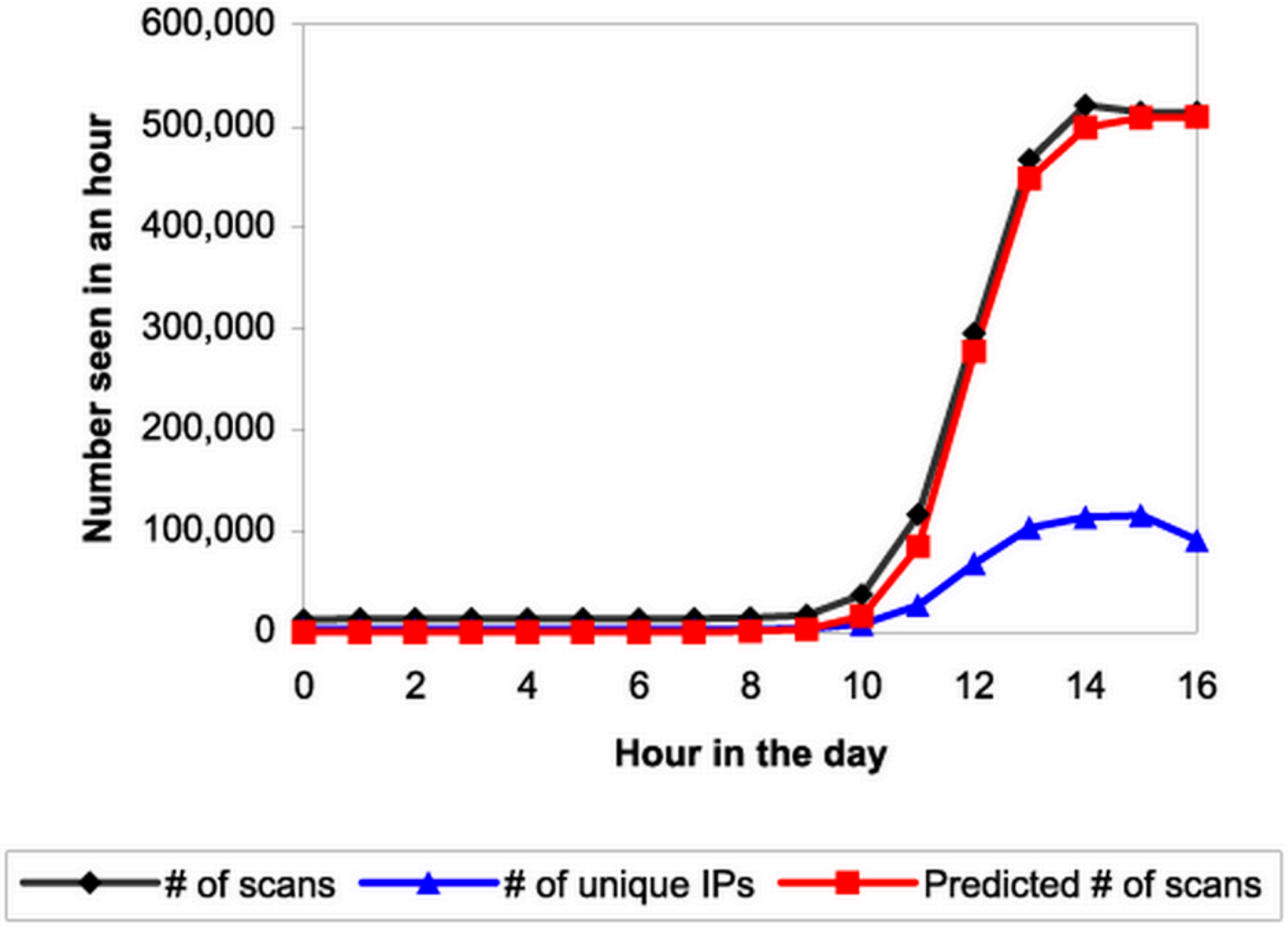
Then…

- How many more machines will be compromised in next dt interval?
  - Nda = NaK(1-a)dt
    - Assumes only infect once ("SI model" of epidemiology)
  - da/dt = Ka(1-a)
  - $a(t) = e^{K(t-T)}/(1+e^{K(t-T)})$
  - Sigmoid or Logistic curve
    - Pierre Verhulst, 19th C.

Logistic Curve



Code Red

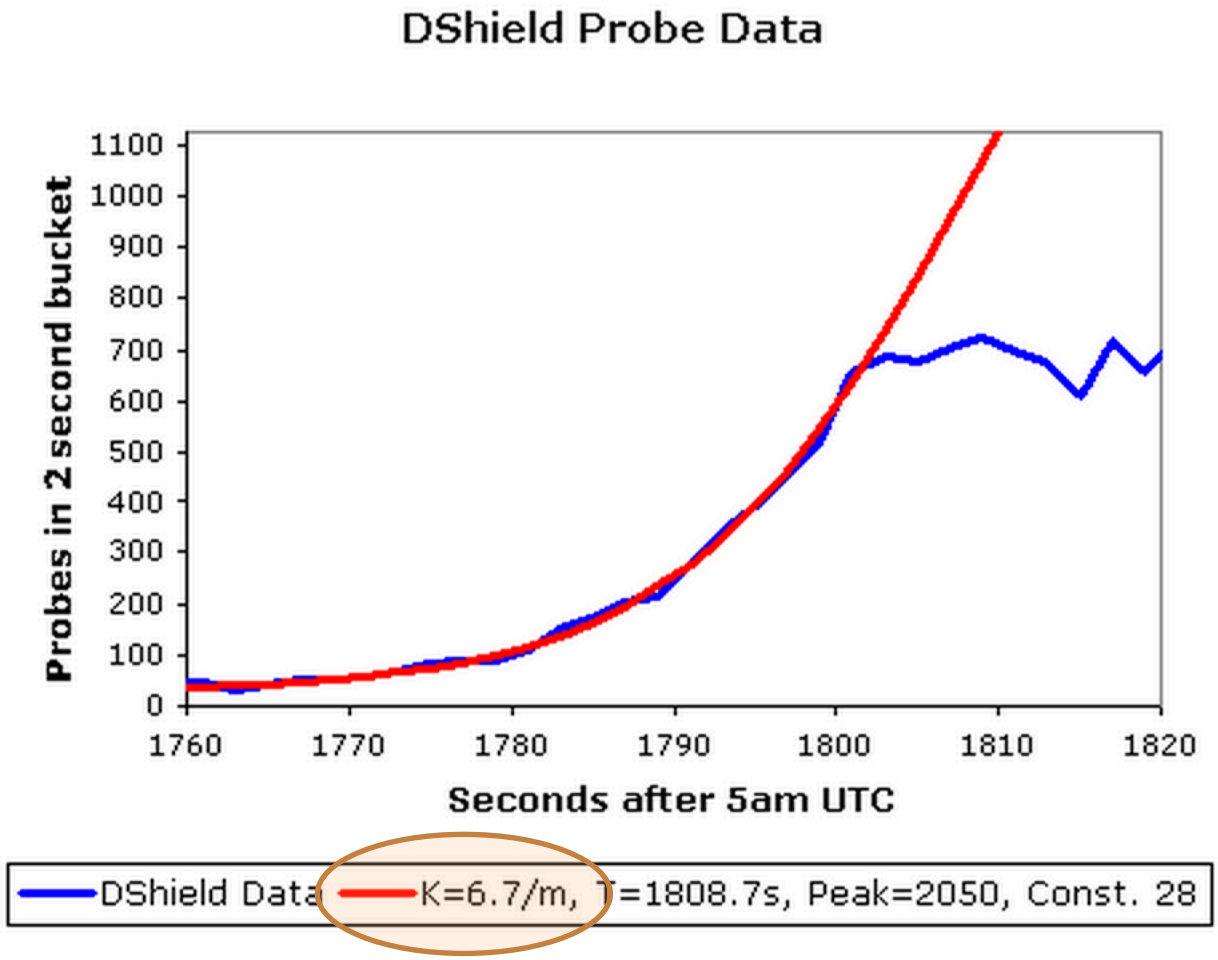


K = 1.8/hr

Code Red Spread

- Vulnerability in IIS Web Server
  - http://www.youtube.com/watch?v=v6GnX3ZhuAg

We thought Code Red was fast, until…



http://www.caida.org/publications/papers/2003/sapphire/sapphire.html

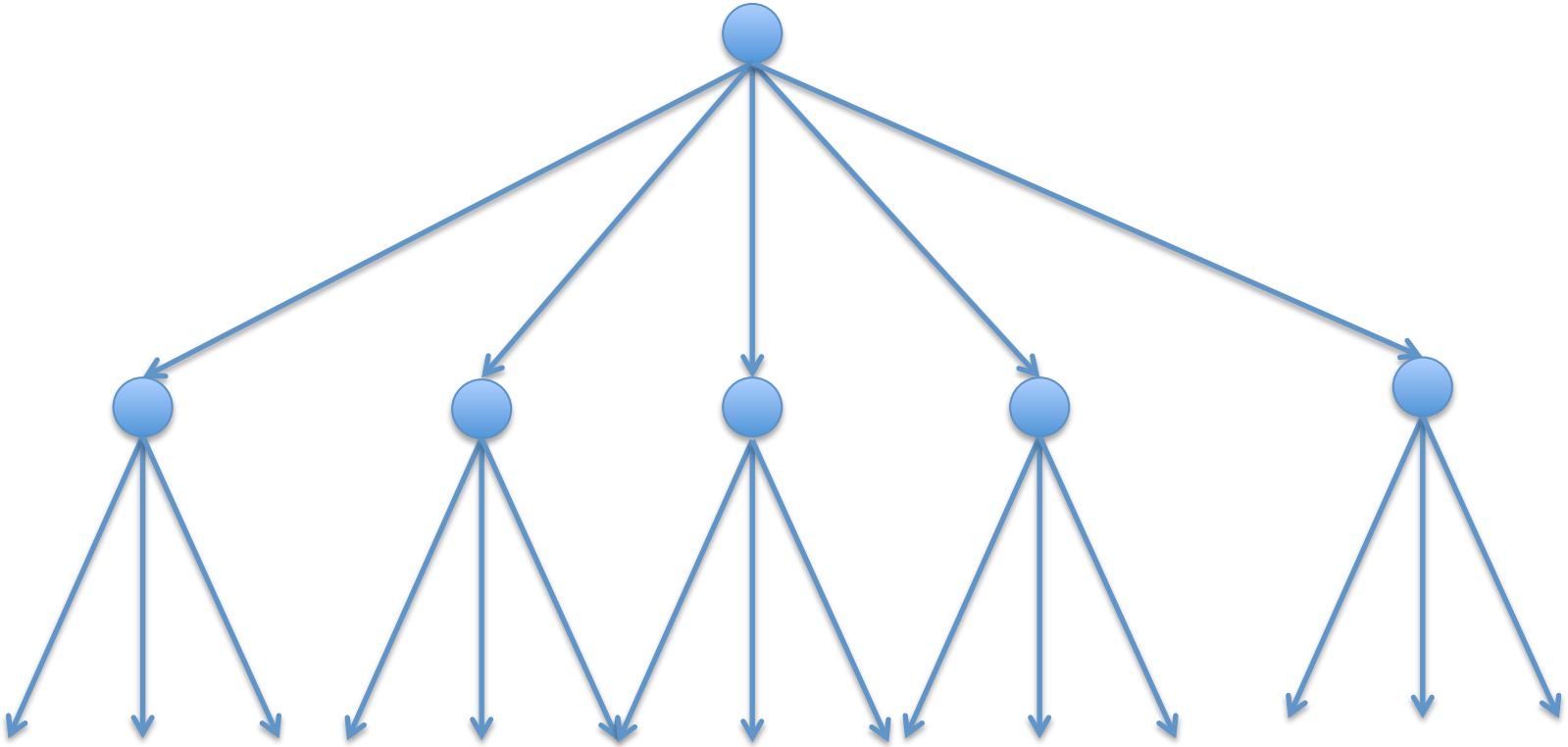
How Slammer Did It

- Occurred in 2003.
- Exploited a vulnerability in MS SQL Server.
- Worm data was only 376 bytes! Fit in one packet.
- Handcrafted machine code contained:
  - Data to overflow buffer and gain control
  - Code to find the addresses of needed functions.
  - Code to initialize a UDP socket
  - Code to seed the pseudo-random number generator
  - Code to generate a random address
  - Code to copy the worm to the address via the socket
- Could spew out hundreds or thousands of worms per second from each infected machine.

How Fast Could a Worm Be?

- Flash worm strategy
  - Do the scanning ahead of time
    - Map entire network/Internet for vulnerabilities
    - (We know intelligence agencies do this)
  - Then precompute spread tree.
    - Each worm instance carries part of address list with it
    - Takes on infecting its part

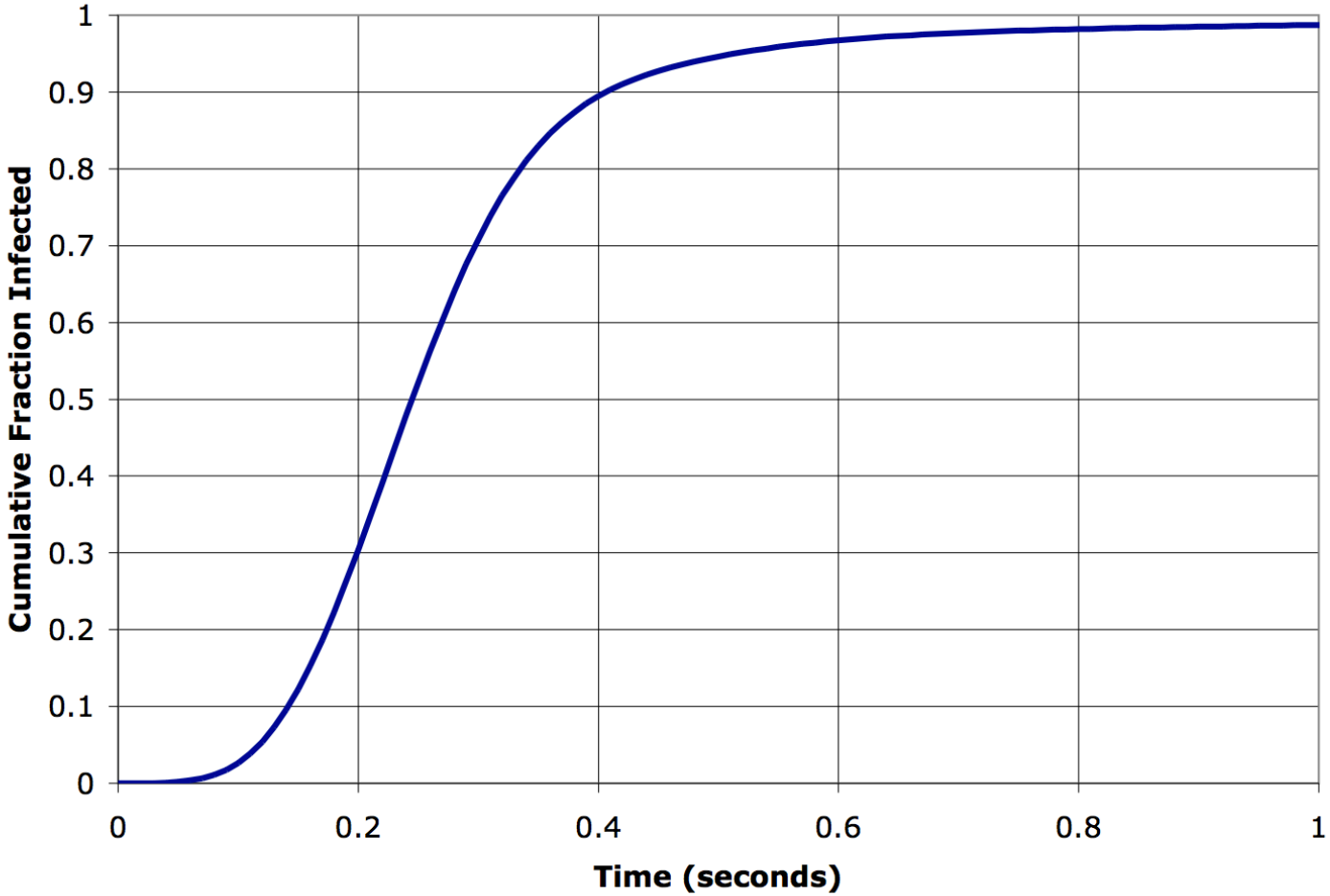
Flash Worm Spread Tree



Strategy

- Pick a very fast node to start on
  - Then a shallow tree
  - Secondary node address lists can fit in one pkt
- Simulation used observed Slammer packet delivery speed distribution
- Observed Internet latency distribution
- Optimum was 9260x107 to infect 1m hosts
- http://www.caida.org/publications/papers/2004/topspeedworms/topspeed-worm04.pdf

Single Packet Flash Simulation



Fraction of a million Internet hosts infected

http://www.caida.org/publications/papers/2004/topspeedworms/topspeed-worm04.pdf

Ultimate constraint

- Speed of light in fiber
  - To go half way around world as crow flies:
    - 6370000*3.14159/3x10^8/(2/3)
    - about 100ms
- Plus time to infect each host

Defenses for Scanning Worms

- Host-level:
  - ASLR/DEP/Canaries/etc
  - Limit outbound connections
- Network level
  - Detect/block scanning
    - Firewalls
    - Packet filtering in routers
    - Intrusion prevention systems
    - In-switch security measures

Overall Dynamic

- Suppose each worm finds r children to infect
  - Total before containment/remediation.
- Successive generations:
  - 1, r, r^2, r^3,…
  - $1 + r + r^2 + r^3 + \ldots = 1/(1-r)$ if $r<1$
  - Eg if r = 0.9, total is 1/(1-0.9) = 10
  - If r > 1, series diverges.
- So must ensure each worm instance finds on average less than 1 child
  - Epidemic peters out
  - Known as “epidemic threshold”
  - Similar to critical mass in nuclear explosions