An in-depth look at worms, malware, botnets, and

# *Modeling Botnet Propagation Using Time Zones*.

*February 16th, 2006*
*Wyman Park 4th Floor Conference Room*

*Opening Quote Provided by:*

**IRC Operator "Wave" on voltagedrop.redirectme.net**

*Primary Paper Originally by:*

**David Dagon, Cliff Zou, and Wenke Lee**

*Presentation by:*

**Jay Zarfoss**

JOHNS HOPKINS
UNIVERSITY

Information Security
Institute

# Roadmap

1. General overview of a Botnet
2. Bot-infections at the lone PC
3. Botnets Internet-wide
4. How time zones affect Botnets
5. Further extensions and explorations

# What is a botnet?

CERT Definitions:

- Botnet. A collection of computers infected with malicious code that can be controlled remotely through a command and control *(C&C)* infrastructure.

- Bot, or Zombie. An individual computer infected with malicious code that participates in a botnet and carries out the commands of the botnet controller (*botmaster*).

# Why do we care?

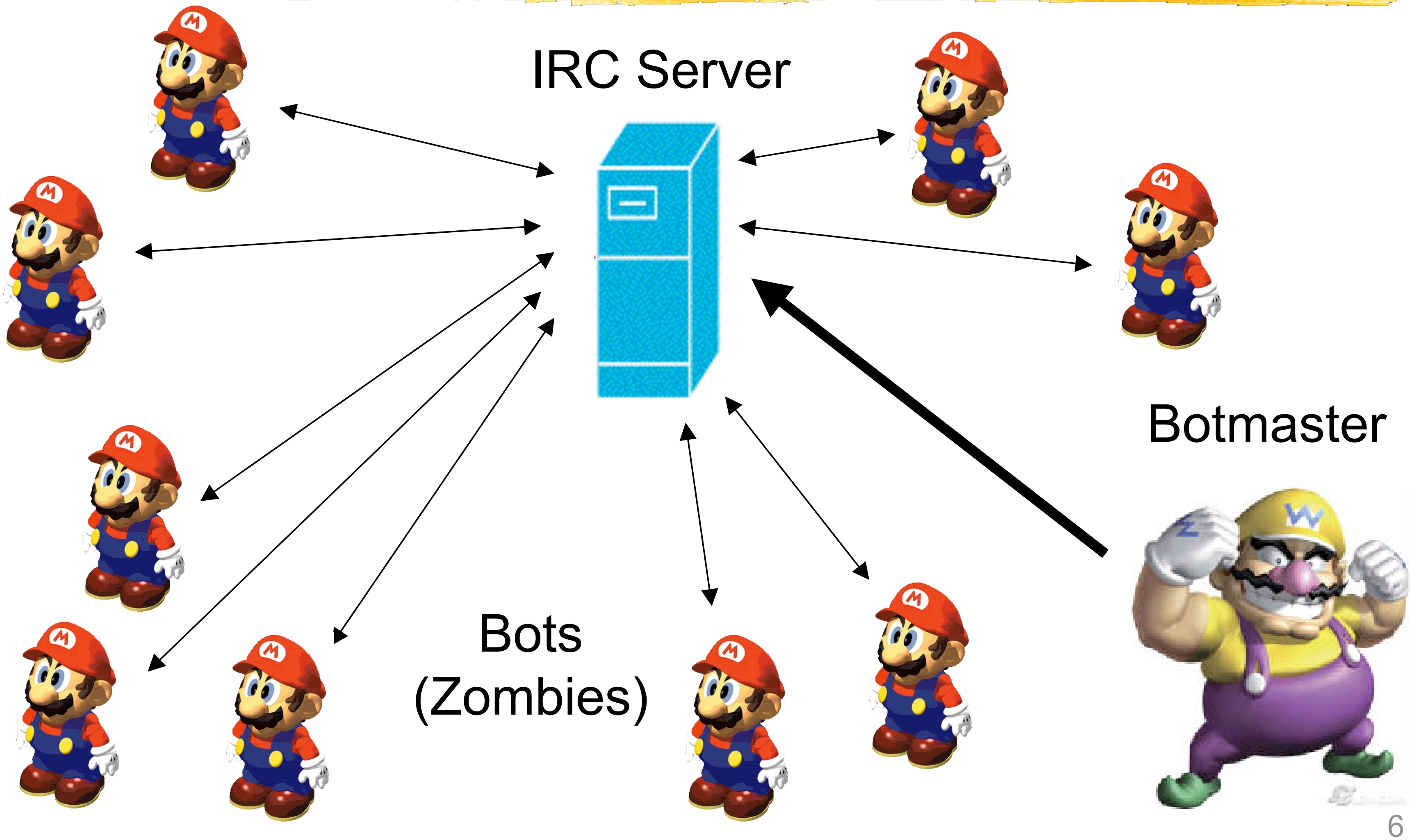
- Generally speaking, an unpatched Windows machine becomes a bot within 10 minutes of joining the Internet.

- *"A botnet is comparable to compulsory military service for windows boxes"*
  - Bjorn Stromberg

# C&C Infrastructure

- How the botmaster issues commands to his army
- Could be implemented with just about any protocol
  - Telnet
  - Instant Messaging Service
  - P2P Network
  - Web Interface
  - **Internet Relay Chat (IRC)**

# IRC-based Botnet



IRC Server

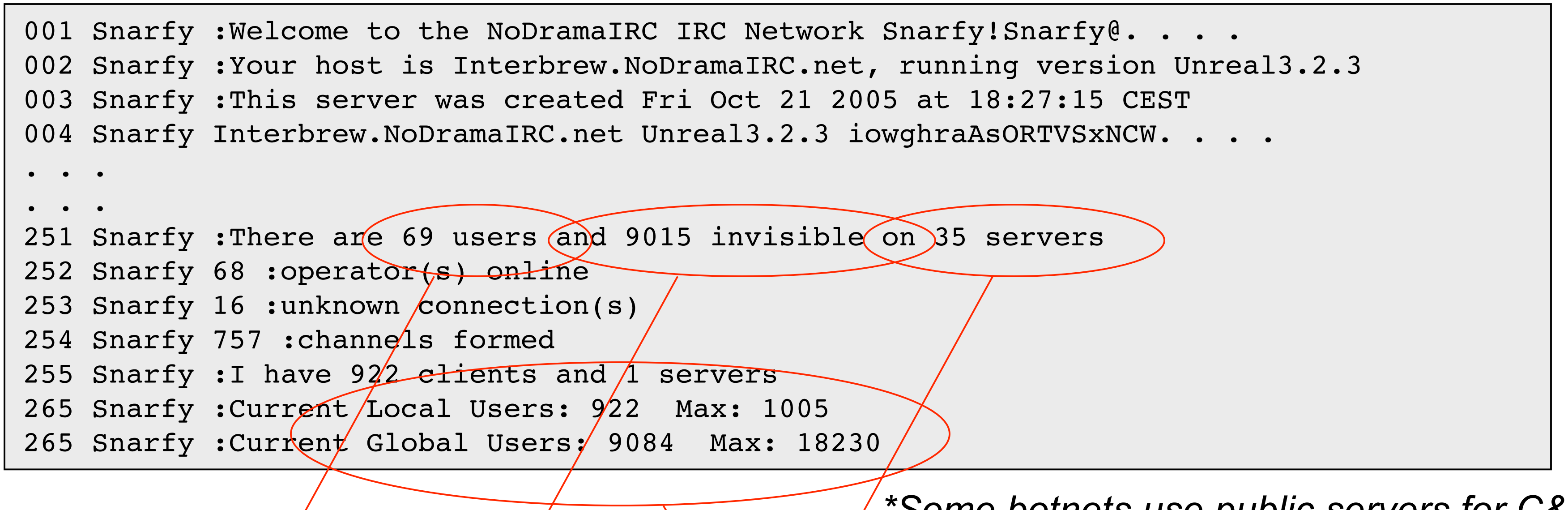
Botmaster

Bots
(Zombies)

# IRC

- Observation: 25 out of 35 botmasters prefer UnrealIRCd available at: www.unrealircd.com

*Sample welcome message from **public** Unreal IRC Server **



```
001 Snarfy :Welcome to the NoDramaIRC IRC Network Snarfy!Snarfy@. . . .
002 Snarfy :Your host is Interbrew.NoDramaIRC.net, running version Unreal3.2.3
003 Snarfy :This server was created Fri Oct 21 2005 at 18:27:15 CEST
004 Snarfy Interbrew.NoDramaIRC.net Unreal3.2.3 iowghraAsORTVSxNCW. . . .
. . .
. . .
251 Snarfy :There are 69 users and 9015 invisible on 35 servers
252 Snarfy 68 :operator(s) online
253 Snarfy 16 :unknown connection(s)
254 Snarfy 757 :channels formed
255 Snarfy :I have 922 clients and 1 servers
265 Snarfy :Current Local Users: 922  Max: 1005
265 Snarfy :Current Global Users: 9084  Max: 18230
```

**Some botnets use public servers for C&C, in this case, bots join a specific channel.*

# What are botnets doing?

- Sending spam
- Stealing passwords
- Extorting online businesses
- Hosting phishing websites
- Click-frauding (5-35% of all clicks)
- Proxying
- Being bought and sold
- Patching themselves
- Recruiting
- Better question: what *aren't* they doing?

# Command the minions

| IRC Command | Bot action |
|---|---|
| `login` | Authenticate botmaster |
| `secure` | Stop vulnerable services |
| `opencmd` | Open a shell to bot |
| `synflood` | Send a SYN flood |
| `update` | Get new version of malware |
| `getclip` | Send clipboard contents |
| **`scanstats`** | **Send bot scanning stats** |
| **`netinfo`** | **Send bot network stats** |
| **`sysinfo`** | **Send bot computer stats** |

Why bother?

Why would the botmaster care?

# Bots used as file servers

- How many movies has YOUR Windows box served lately?

```
#HINDI-FILMZ :#1  294x [698M] [Movie] Dil Bechara Pyar Ka Mara DvD-RiP [ Full / AVI / 2001 ]
#HINDI-FILMZ :#2  126x [141K] [English Subtitles] Dil Bechara Pyar Ka Mara
#HINDI-FILMZ :** 2 packs **  3 of 3 slots open, Record: 45.3KB/s
#HINDI-FILMZ :** Bandwidth Usage ** Current: 0.0KB/s, Record: 304.5KB/s
#HINDI-FILMZ :** To request a file type: /"/msg [HF]-[Street-Hunk]-30 xdcc send #x/" **
#HINDI-FILMZ :** -= #Hindi-Filmz=- **
#HINDI-FILMZ :** I M 100% Desi !! **
#HINDI-FILMZ :Total Offered: 698.5 MB  Total Transferred: 206.57 GB
```

That's a lot of movies served! ( ~ 300)

# Who are the botmasters?

- l33t h4x0rs?
- Graduate students?
- Scam artists?

- *Maybe…*

# Who are the botmasters?

```
[Diabolic] PRIVMSG #hf-help :enuf for me man
[vtx] PRIVMSG #hf-help :olol
[Diabolic] PRIVMSG #hf-help :hahaha tru
[Diabolic] PRIVMSG #hf-help :i wrote 2 essays 2
[Vtx] PRIVMSG #hf-help :lol
[Diabolic] PRIVMSG #hf-help :1 in class and 1 at home
[Vtx] PRIVMSG #hf-help :thts atleast gud
[Vtx] PRIVMSG #hf-help :i had to write 1 for eng. exam and one for hist. exam
[Vtx] PRIVMSG #hf-help :beat tht
```

High school students?

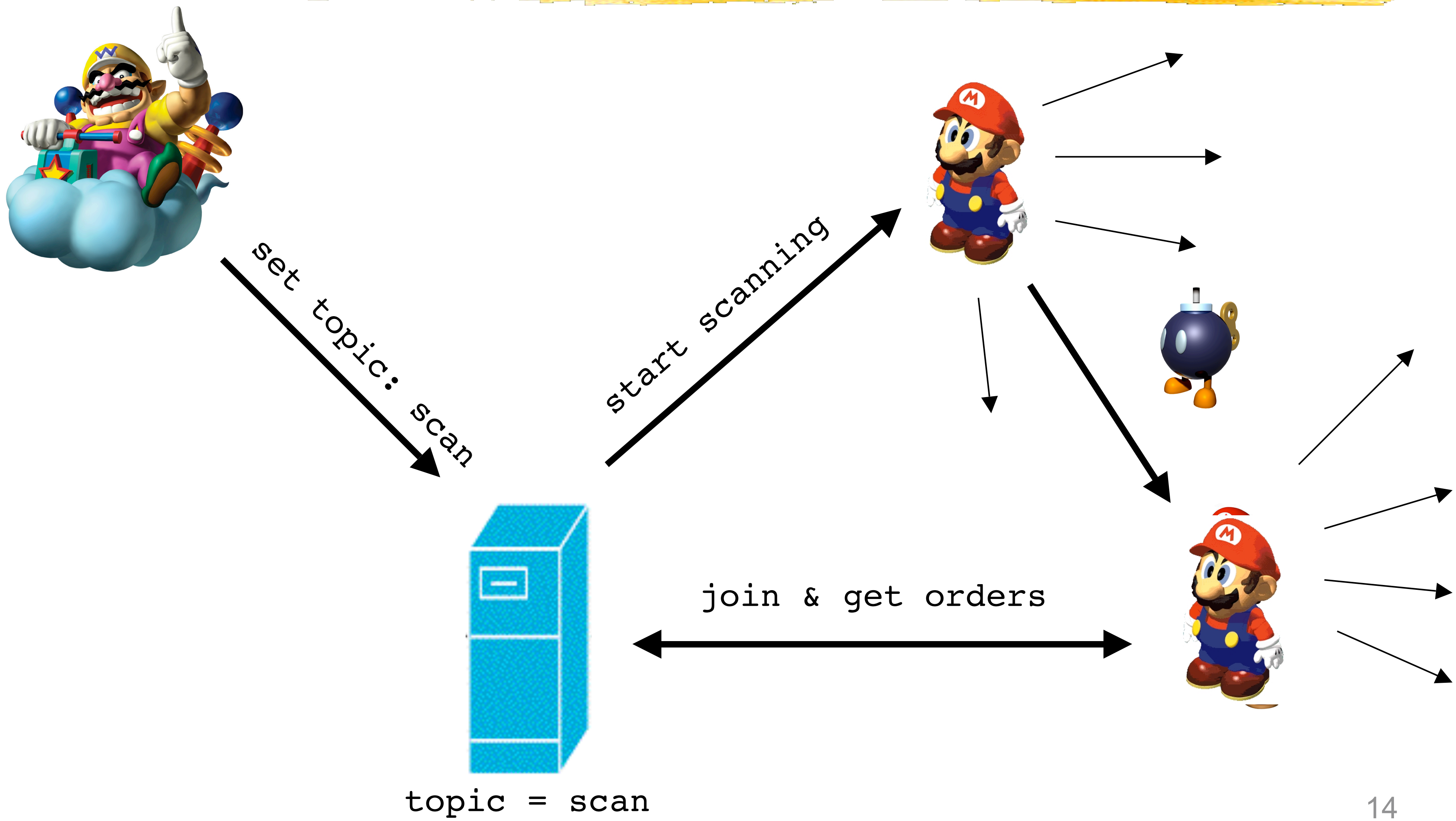
```
[D3si_boi] PRIVMSG #hf-help :man
[D3si_boi] PRIVMSG #hf-help :people are so gay
[D3si_boi] PRIVMSG #hf-help :f**k serioulsy
[D3si_boi] PRIVMSG #hf-help :i had to mop the front lobby at my work
[D3si_boi] PRIVMSG #hf-help :nd f**ked up s**t man
[D3si_boi] PRIVMSG #hf-help :people keep walking over it
[D3si_boi] PRIVMSG #hf-help :over it
[D3si_boi] PRIVMSG #hf-help :dont see it
[D3si_boi] PRIVMSG #hf-help :nd dont even say sorrty
```

Custodians?

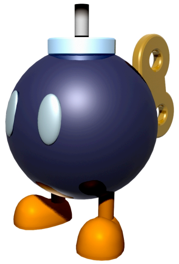
# How is this so easy?

- Botnets use worms to propagate.
  - On many levels, a bot is a worm
    - Like many worms, often spreads with scans
    - Bot can run independent of the C&C
    - Bot can infect (recruit) other machines
  - Botmasters can easily “update” their bots to make sure they have the latest 0day exploits at their disposal
    - *And they do!*

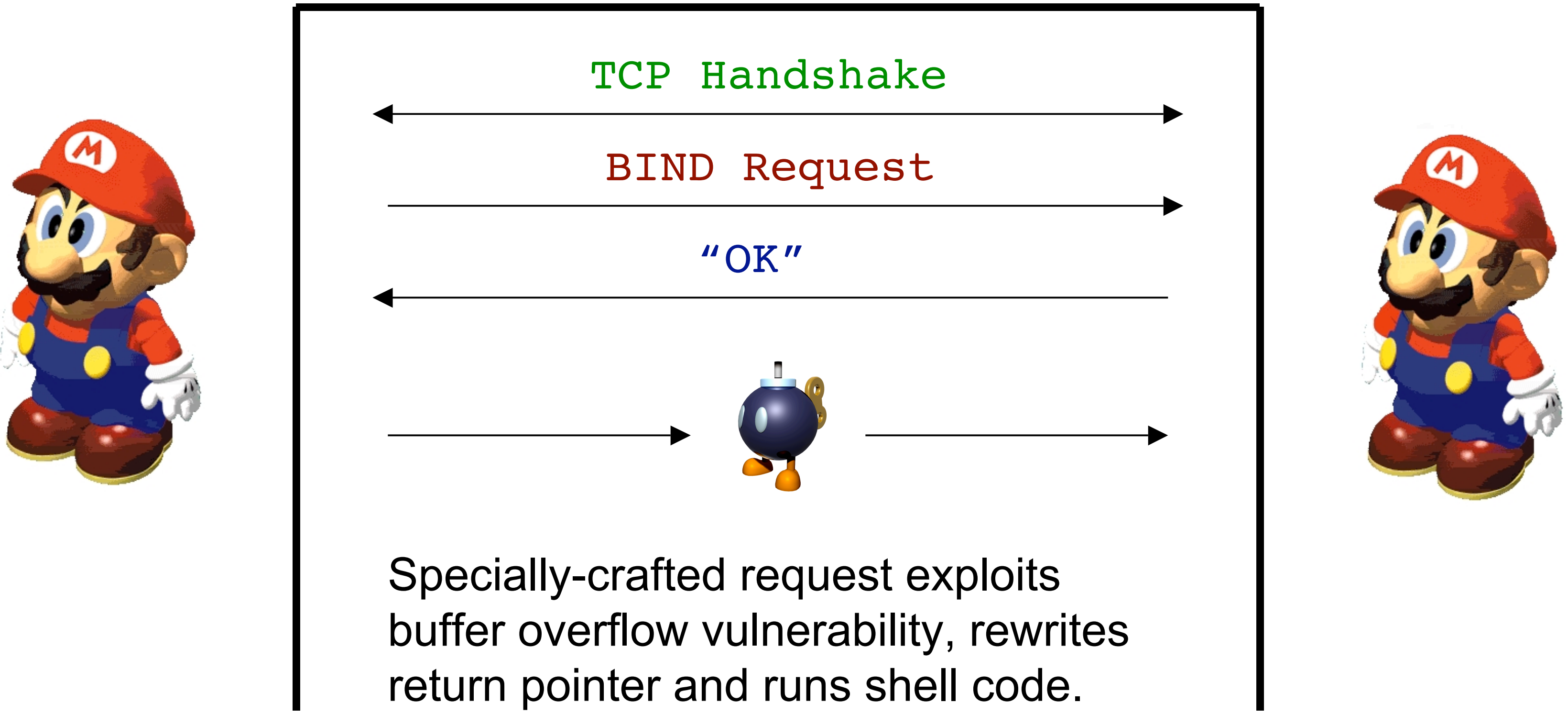
# The single PC recruitment



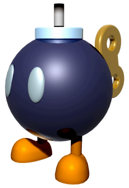
# A Closer Look



A Protocol Perspective, (Simplified) DCOM RPC



Specially-crafted request exploits buffer overflow vulnerability, rewrites return pointer and runs shell code.



# DCOM RPC Exploit Recipe*

```
//Get a funky fresh socket
//Fill in sockaddr and resolve host
//Get shellcode
//Connect to the server
//Send the BIND string
//Read Reply
//Send the evil request
//Read Reply
//Close socket
```

Repeat until botnet is of desired consistency

*Derived verbatim from comments in bot source code*

# Obvious to an IDS

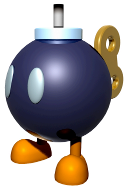
```
char nops[] =
"/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90”
"/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90”
"/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90”
"/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90”
"/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90”
"/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90”
"/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90”
"/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90”
"/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90”
"/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90/x90”
"/x90/x90/x90/x90/x90/x90/x90";

char shellcode_start[]=
"/x46/x00/x58/x00/x4E/x00/x42/x00/x46/x00/x58/x00/x46/x00/x58/x00”
"/x4E/x00/x42/x00/x46/x00/x58/x00/x46/x00/x58/x00/x46/x00/x58/x00”
"/x46/x00/x58/x00”
"/xff/xff/xff/xff”   /* return address */
"/xcc/xe0/xfd/x7f”   /* primary thread data block*/
”/xcc/xe0/xfd/x7f"; /* primary thread data block */
```

VERY Distinct

# Hide from the IDSes

- NOOPS are a dead giveaway
  - Can be replaced by 55 equivalent ops
- Other key strings are distinct too
  - Can obfuscate in very simple ways
    - Send XORed with another string
    - Use very simple encryption schemes
    - Do anything to change the signature!
    - domain.com/phf? == domain.com/./phf?
- IDSes have a lot to look out for!

# Discover the  -payload

Bill Cheswick. **An Evening with Berferd In Which a Cracker is Lured, Endured, and Studied**. In *Proceedings of the Winter Usenix Conference*, San Francisco, CA, 1992.

- Observed hacker attempting to break into a computer through Sendmail bug.
- Cheswick emulated exploitable services **by hand**.
- Sent fake password lists, etc etc.
- Discovered what vulnerabilities the hacker knew about.

- There’s got to be an easier way?

# Virtual Responders

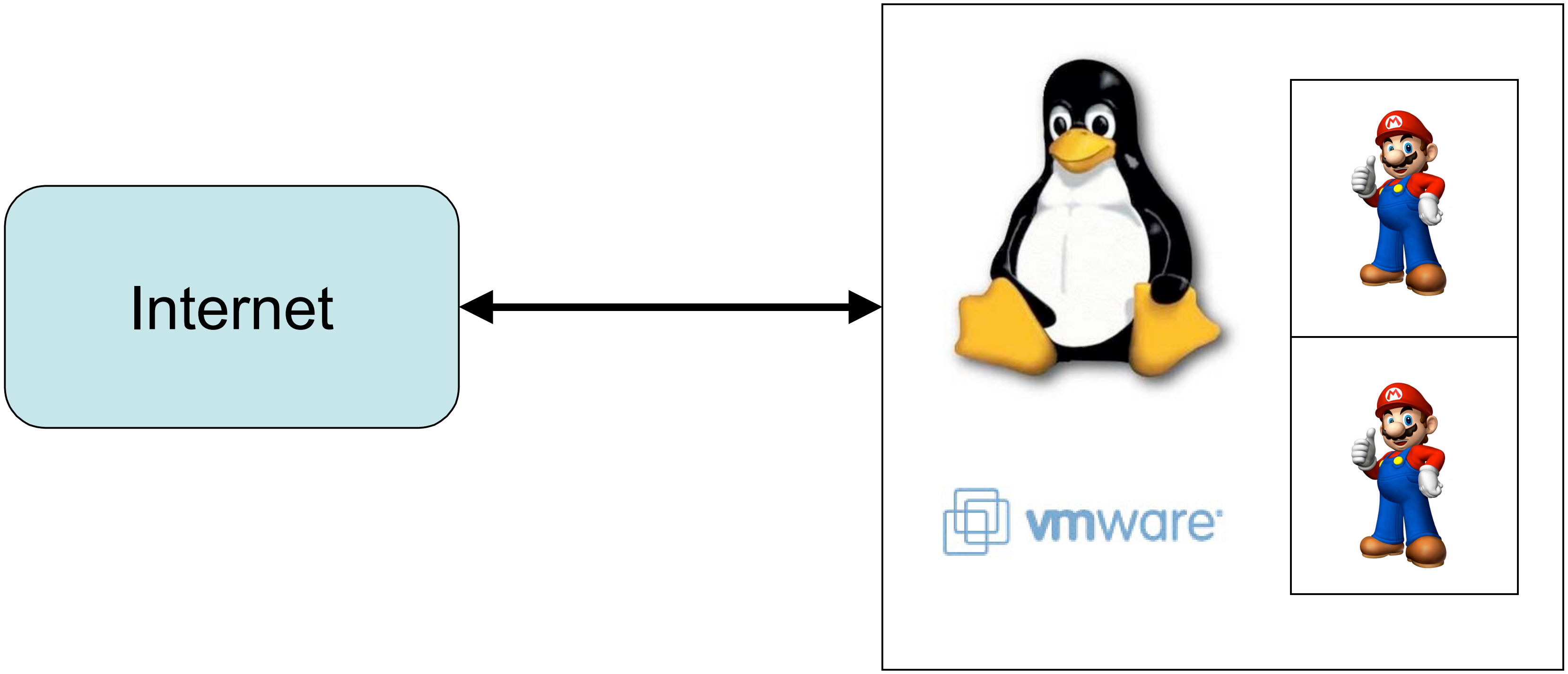
- Runs non-natively (on BSD or Linux)
  1. Simulate known vulnerabilties on well-known ports (DCOM-RPC = port 135)
  2. Analyze incoming shellcode, attempt to extract IPs/URLs from payload
  3. Download from the embedded URL
  4. Submit downloaded file to a database of known malware

# “Traditional” honeypot

- www.honeynet.org
  - When “simulating” a protocol just isn’t gonna cut it, give them the real thing!
  - Monitor traffic to determine behavior
- But…
  - This is really hard to scale up…
  - Honeypots get attacked so violently their stability quickly approaches 0

# Virtual Honeynets

- Run multiple “virtual” instances of vulnerable OS within non-native OS.



# Virtual Honeynets

- Nice!
  - Easy to maintain
  - Cheaper, less hardware to buy
- But…
  - STILL doesn’t scale up very well (we’ve barely been able to run 2 VMs per physical box)
  - Can an attacker somehow tell that he’s talking to a virtual machine and not the real thing?

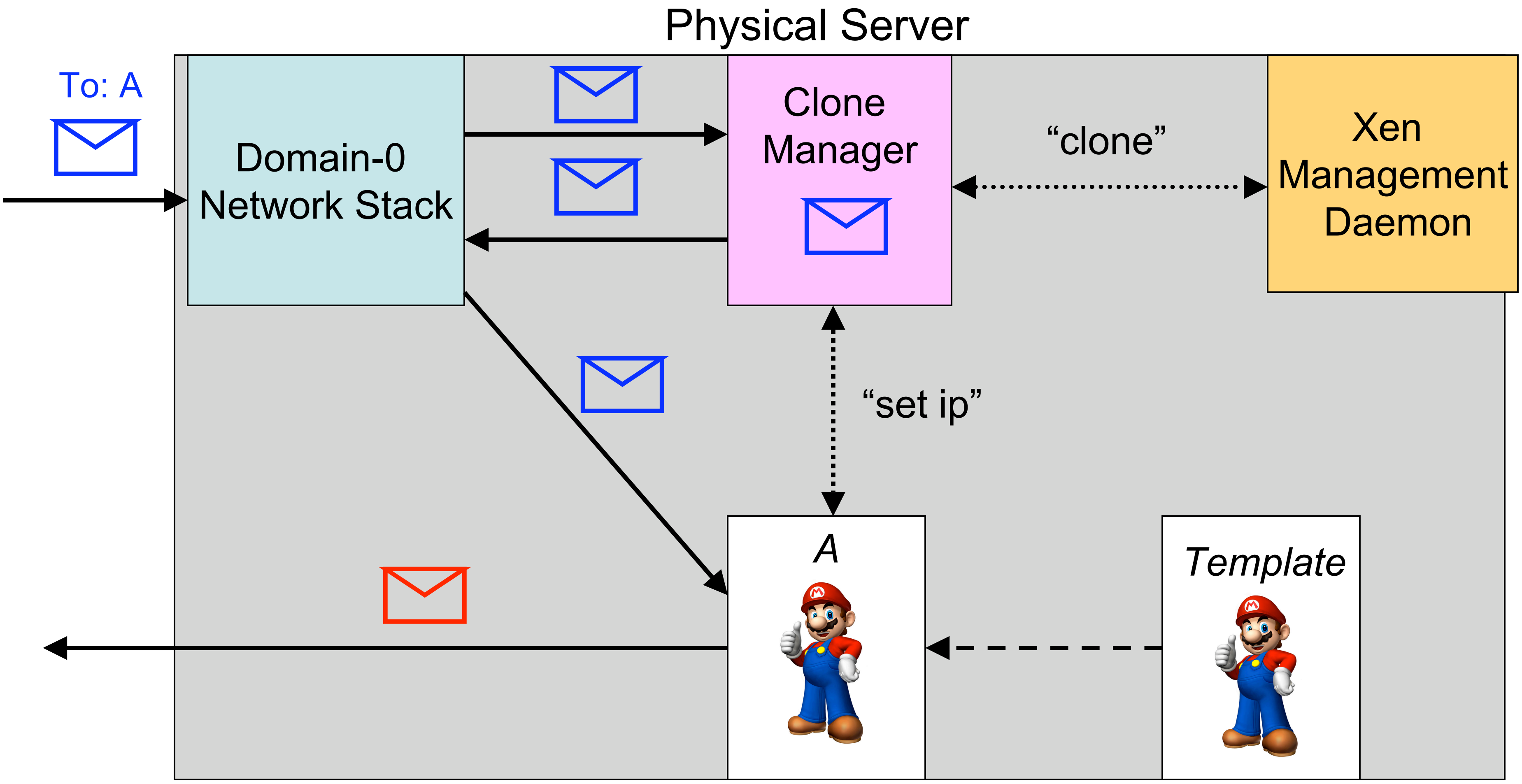
How? And can we prevent this?

# How to scale up

Michael Vrable, et al. **Scalability, Fidelity, and Containment in the Potemkin Virtual Honeyfarm**. In *SOSP 2005*.

- How can we get real honeypot coverage of a large IP space?
- Only create a VM when you NEED to.
- Speed up VM creation with ***flash cloning***
- Share memory between VMs using ***delta virtualization***. (copy-on-write)
- Use faster ***paravirtualization***

# Potemkin Flash Cloning



# Potemkin Δ-Virtualization

- Don’t “copy”, just make a reference
- If you need to write to memory, do a deep copy into a ***shadow pagetable***
  - Simple ping replies don’t need memory
  - Bots may require keeping a lot of state and writing to memory
    - *Those movies take up a lot of space!*

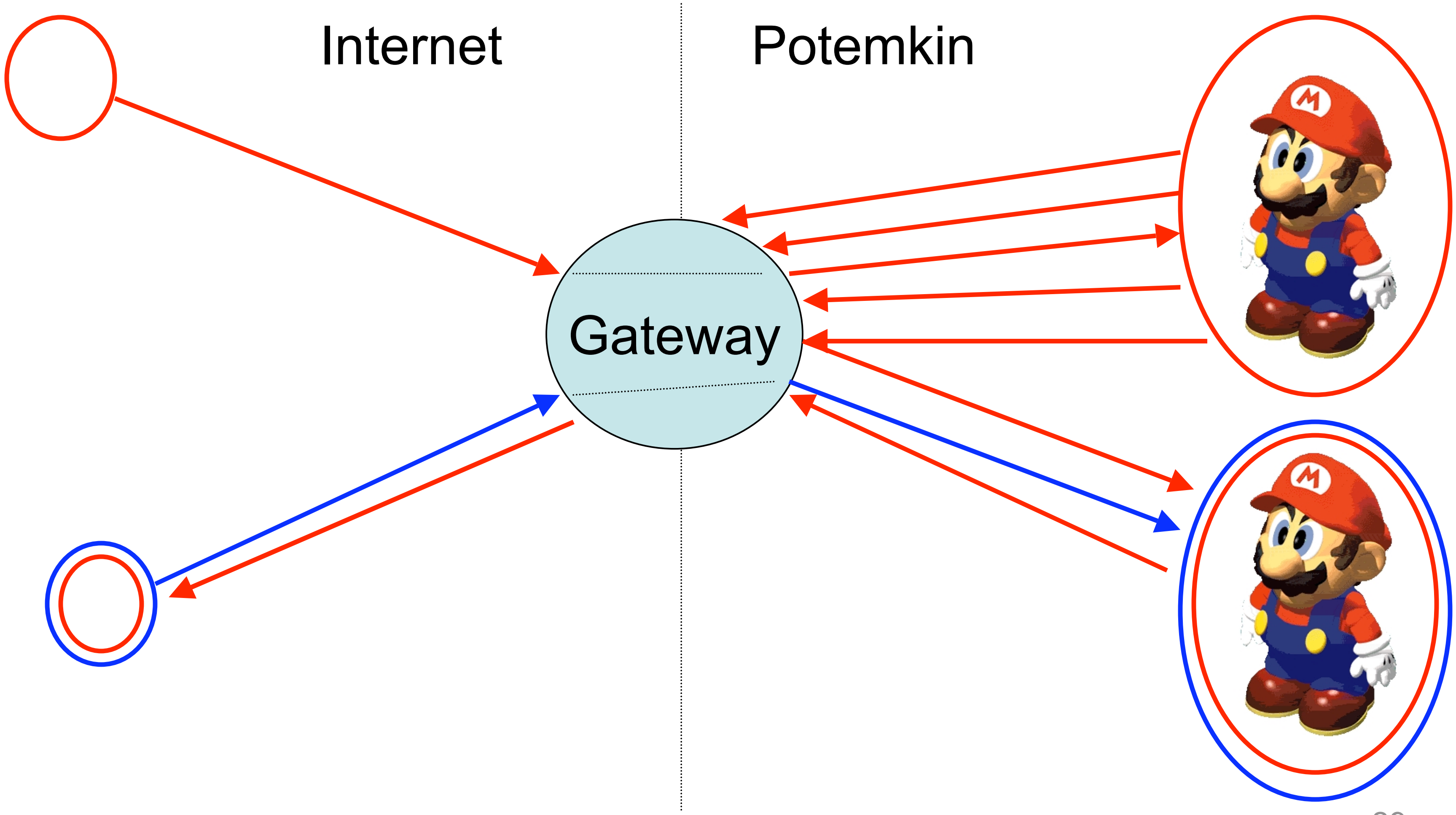
# Potemkin Paravirtualization

- Regular virtualization is too slow!!
  - We pay a huge penalty by simulating the hardware within software
- Solution:
  - Port the virtualized operating system to use the interface provided by the Virtual Machine Monitor (VMM).
  - The virtualized OS is in on the joke!

# The Potemkin Gateway

- Gateway must be VERY smart
  - Manage all inbound traffic to appropriate VM Servers on internal darknet
  - Provide **containment** of outbound traffic
    - Not as simple as keeping outbound traffic limited to the source of initial connection
  - Internal **reflection** between VMs can create cross infections

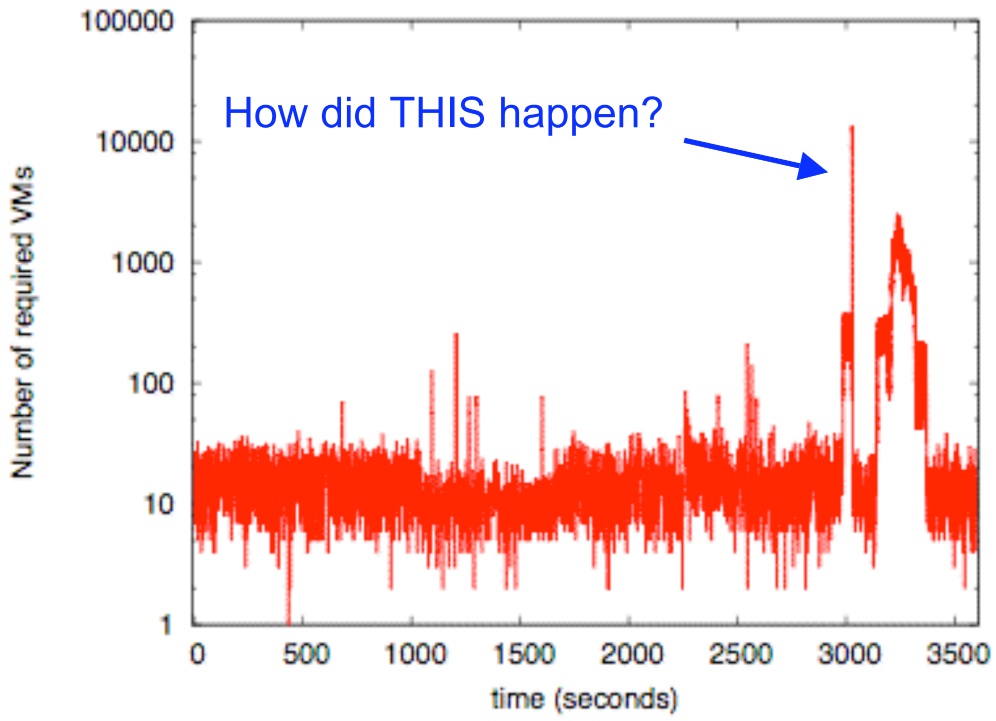
# Cross Infections



# Lots of gotchas

- One virtual machine tries scanning other virtual machines?
  - Sometimes we need to see different worms interacting with each other!
- Handle a single ping to all 64k virtualized hosts? ***All at once?***
  - What about 64k random packets?
- When do we destroy a VM and cannibalize the memory?
  - How do we know the attacks are over?
  - Could we ever reclaim a VM on a botnet?

# Potemkin in the wild



- /16 coverage
- Aggressive VM recycling
- Windows not yet supported (hopefully soon!)

# Filter the “known” traffic

Weidong Cui, Vern Paxson, et al. **Protocol-Independent Adaptive Relay of Application Dialog**. In *NDSS 2006*.

- We are only interested in exploits and malware we haven’t seen before
- Developed *RolePlayer*, a system to mimic most application dialogs
- Honeypots are valuable resources, save them for the malware we don’t already know about!

# Internet-wide scale

How can we track botnets across the entire Internet?

- Bots often behave just like scanning worms.
- Well then how do we track scanning worms over the Internet?
- Possible Answer: **Internet Telescopes**

# Network Telescopes

David Moore, et al. **Network Telescopes: Technical Report**. Cooperative Association for Internet Data Analysis (CAIDA), 2004.

- Monitored portion of IP space where little or no legitimate traffic exists.
- Observes *endemic* attacks
  - Backscatter from SYN floods, DOS attacks
- Observes *pandemic* attacks
  - Scans from an internet-wide worm outbreak

# Network Telescopes*

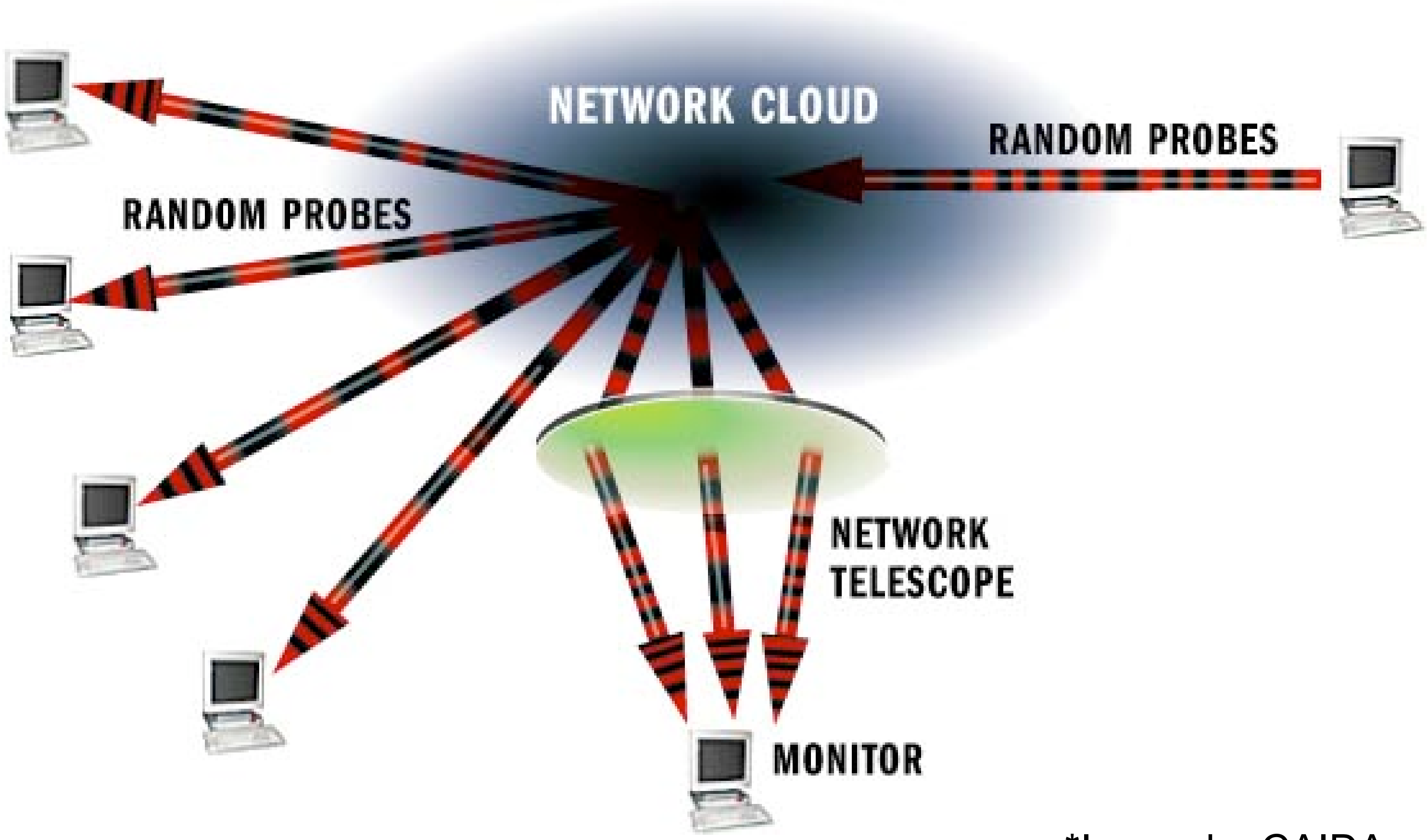
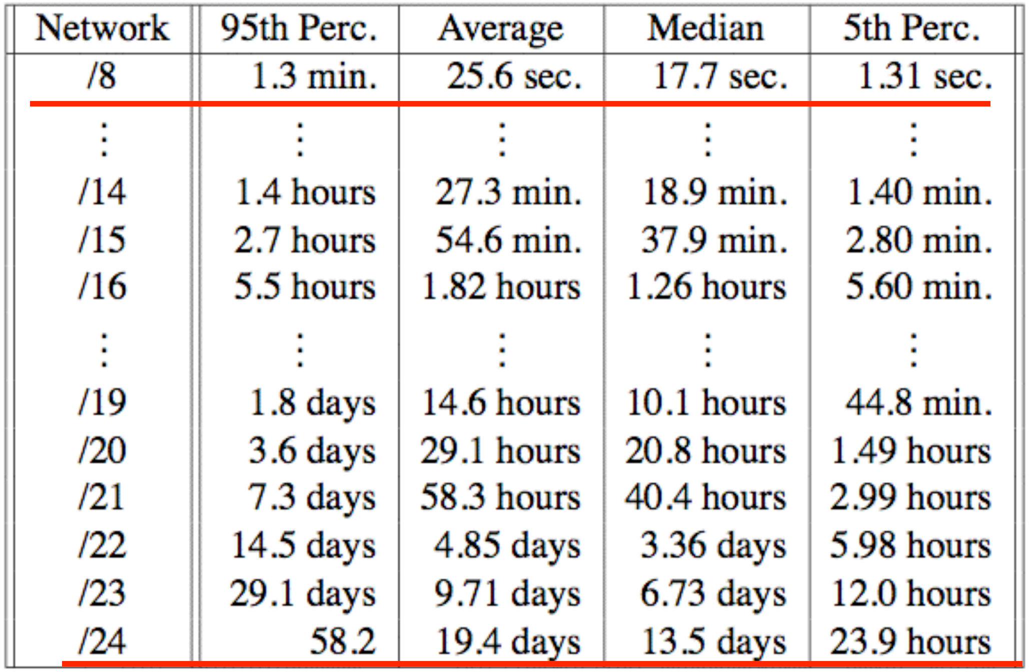


*Image by CAIDA

# Network Telescopes

- The seismographs of the Internet: can detect even single source of random scans or attacks



| Network | 95th Perc. | Average | Median | 5th Perc. |
|---|---|---|---|---|
| /8 | 1.3 min. | 25.6 sec. | 17.7 sec. | 1.31 sec. |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| /14 | 1.4 hours | 27.3 min. | 18.9 min. | 1.40 min. |
| /15 | 2.7 hours | 54.6 min. | 37.9 min. | 2.80 min. |
| /16 | 5.5 hours | 1.82 hours | 1.26 hours | 5.60 min. |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| /19 | 1.8 days | 14.6 hours | 10.1 hours | 44.8 min. |
| /20 | 3.6 days | 29.1 hours | 20.8 hours | 1.49 hours |
| /21 | 7.3 days | 58.3 hours | 40.4 hours | 2.99 hours |
| /22 | 14.5 days | 4.85 days | 3.36 days | 5.98 hours |
| /23 | 29.1 days | 9.71 days | 6.73 days | 12.0 hours |
| /24 | 58.2 | 19.4 days | 13.5 days | 23.9 hours |

On a /8, will detect 10 scan/sec random scan within seconds! :-)

Assumes uniform, random scans !

On a /24, you're not going to see anything for days… :-(

# What can telescopes do for us?

- Witness global worm outbreaks…
- Witness the spread of large botnets?
  - After all, bots act like worms!
  - Right?
- Can data collected by telescopes help us build a model describing the spread of worms and botnets?

# Model scanning worm propagation

- Epidemiological Model
  - First attempt to model worm and virus propagation through the internet.

$$\underbrace{\frac{dn}{dt}}_{\text{Change in infected ratio}} = \underbrace{\beta}_{\text{“Birth rate” of new infections}} n(1-n) - (\underbrace{d}_{\text{“Death Rate”}})(n)$$

# Epidemiological Model

- No consideration of patching rate
- Considers infections continuously
- In reality, infections follow a more discrete timeflow
  - Units of time to get scan results
  - Units of time to interact with a vulnerable process
  - Units of time to send worm copy

# The AAWP Model (2003)

- Analytical Active Worm Propagation
  - Consider time discretely, time in “units”
  - Add a patching rate, *p*.

$$\underbrace{n_{i+1}}_{\text{Number of infections at time i+1}} = \underbrace{(1-d-p)}_{\text{Decrease of infections due to patching \& deaths}}n_i + \underbrace{[(1-p)^i N - n_i]}_{\text{Still vulnerable after patching}}\underbrace{[1-(1-\frac{1}{2^{32}})^{\overset{\text{Scanning rate}}{\overset{\uparrow}{sn_i}}}]}_{\text{Chance of getting scanned}}$$

# What about non-uniform scanning?

- In reality, many worms use *non-uniform scanning*, eg (Nimba):
  - 50% of the time, scan within same /16
  - 25% of the time, scan within same /8
  - 25% of the time, scan space randomly

# Botnet scans

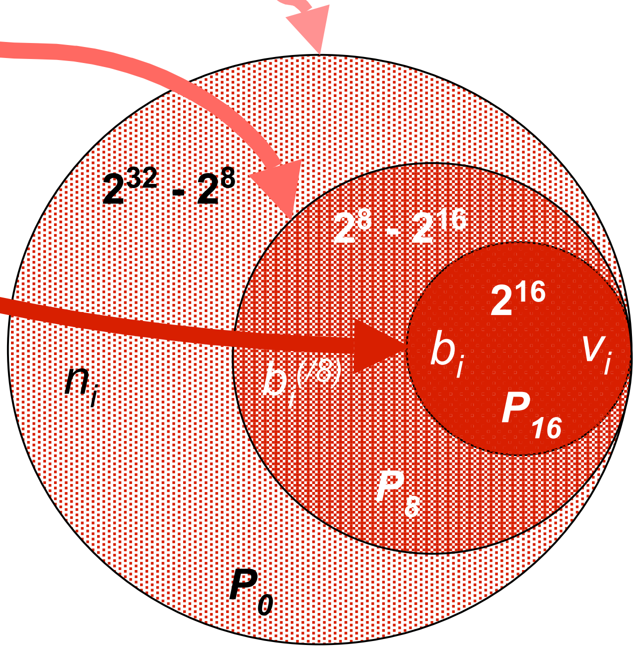
- Our data shows that bot scans are overwhelmingly **NON-UNIFORM**
- Observed 1040 commands to scan
- 511/1040 scans within a /8
- 492/1040 scans within a /16
- We observed 37 orders (~3.5%) to scan uniformly and randomly within the entire IP space

# Model with non-uniform scans

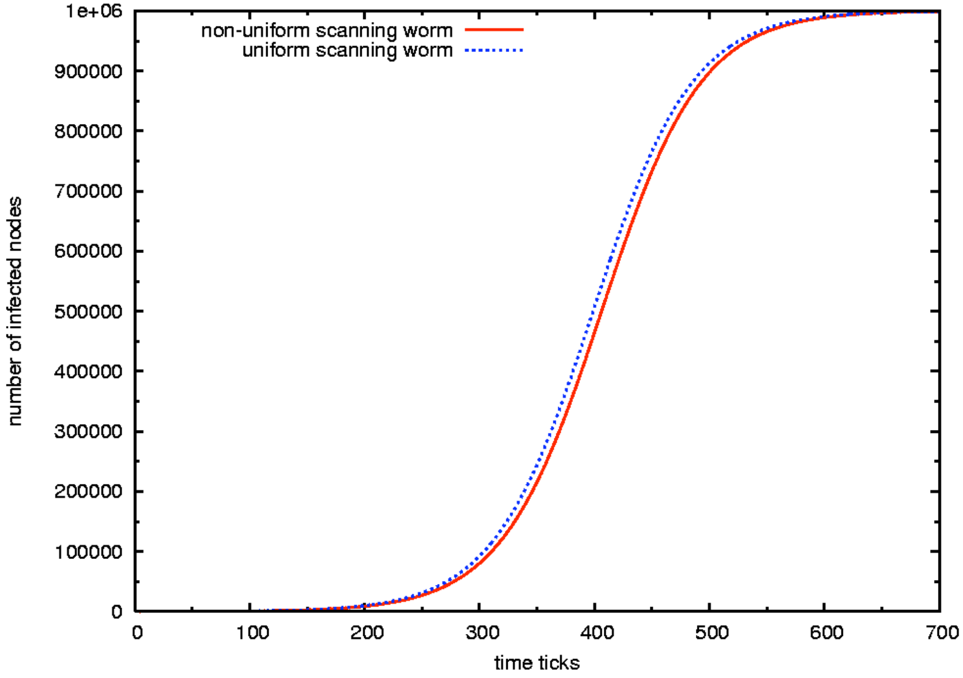
Effective scanning rate

$$k_i^j = \underbrace{p_{16}\, s\, b_i^j}_{} + \underbrace{p_8\, s\, b_i^{(/8)} \frac{2^{16}}{2^{24}}}_{} + \underbrace{p_0\, s\, n_i \frac{2^{16}}{2^{32}}}_{}$$

Number of scans coming into the jth /16 subnet



# Why do the botmasters care?



AAWP by itself seems to imply that **uniform** scanning worms propagate faster!

Expected infection speed for a uniformly scanning vs. a non-uniformly scanning worm with same other parameters

# Uniform vs. non-uniform

Does uniform scanning REALLY create faster propagation??

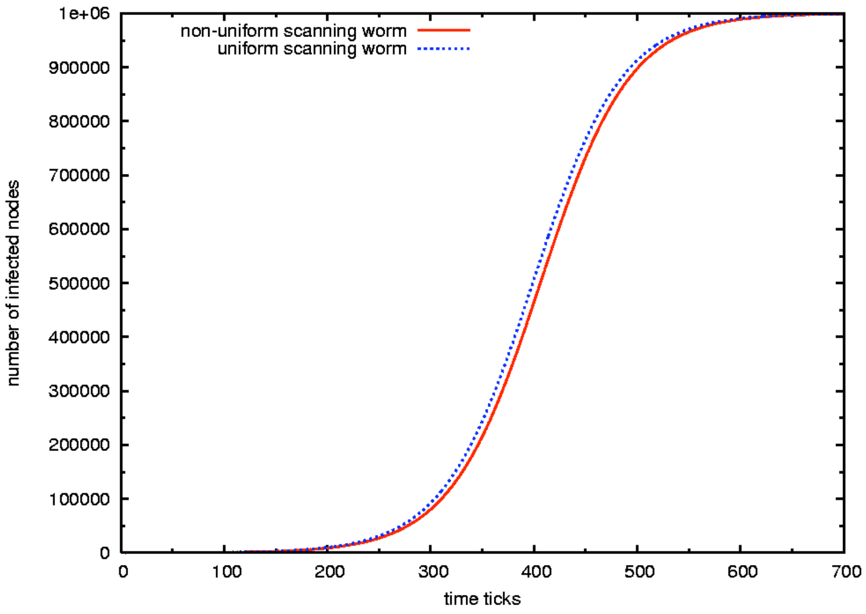
- Code Red (uniform scanning)
  - 10,000 infected in 14 hours
- Code Red II (non-uniform scanning)
  - 359,000 infected in 14 hours
- Hmm… what are we doing wrong?

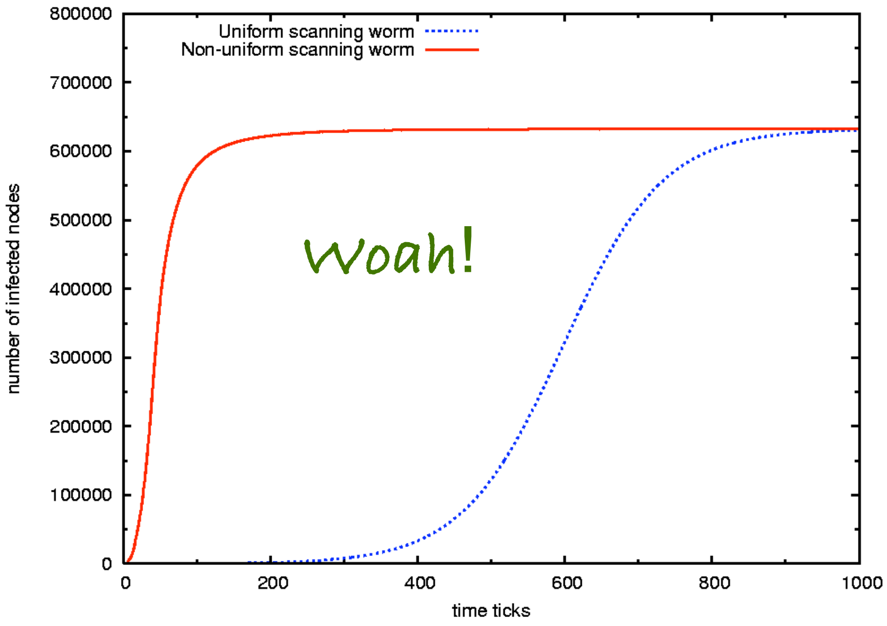
# Improving the Model

Moheeb Abu Rajab, et al. **On the Effectiveness of Distributed Worm Monitoring**. Proceeding of the *14th Usenix Security Symposium*, 2005.

- The vulnerable population isn’t spread evenly over the entire IP space.
- What happens to our worm propagation models when we use real-world victim-distribution data?

# Vulnerable Population





Run the same experiment, but this time consider the vulnerable population to be non-uniformly distributed among IP space.

# Worm -> Botnets Model Recap

- Continuous Model
- Discrete Model
- Add Non-Uniform Scanning
- Add Distribution of Vulnerable Pop.
- Add Homogeneity of scanning rate

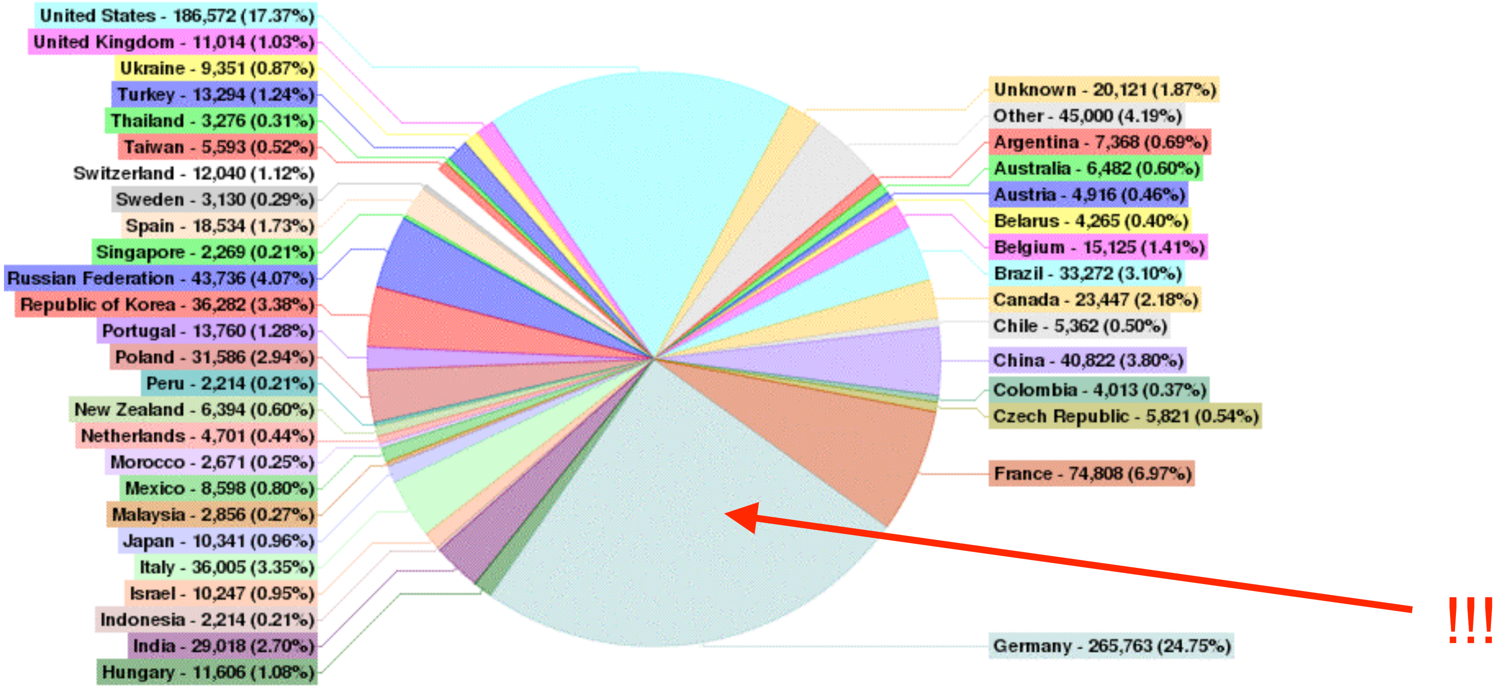
- *Is there more?*

# Time Zones

David Dagon, et al. **Modeling Botnet Propagation Using Time Zones.** Proceedings of *ISOC NDSS 2006.*

- People turn their computers off at night
- Create diurnal pattern in infections
- A bot can’t follow orders if he’s not turned on!

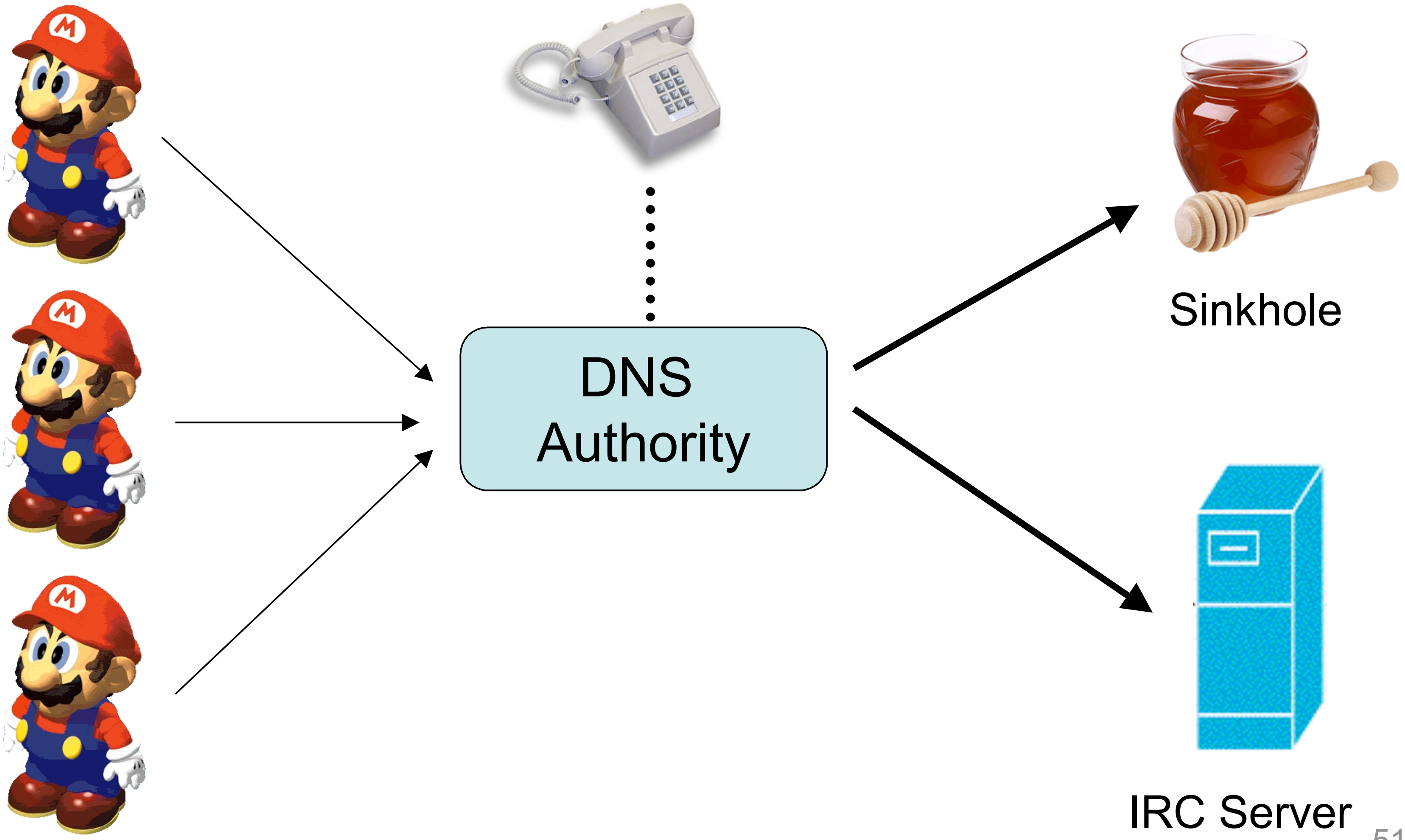
# Time Zone Motivation



!!!

1-25-06 to 2-07-06 Bagle Infections by Country

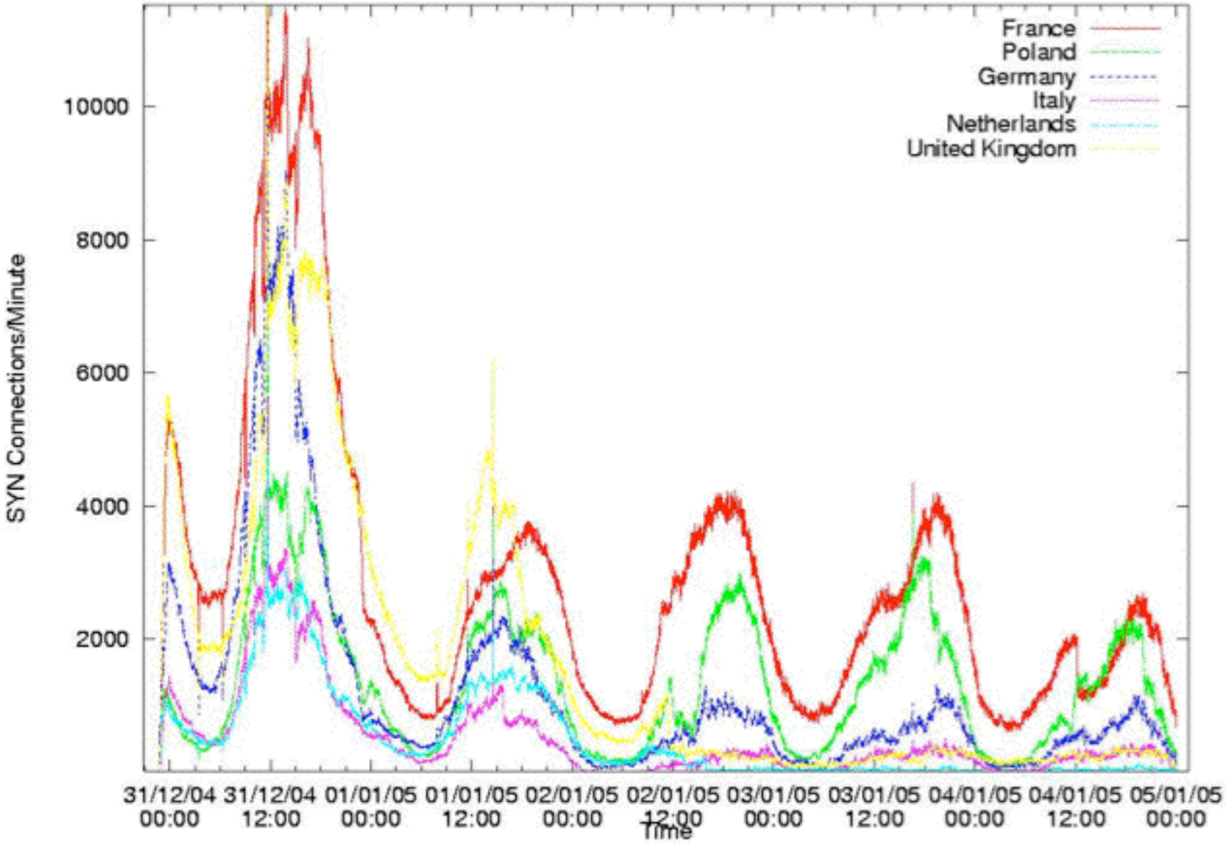
# Data Collection Diagram



# Data Collection Problems

- Forensics and hand analysis of malware binaries hard to automate
- Assumes cooperative DNS owners
- Claim all 50/50 of bots used DNS
  - We have observed 9/35 with no DNS
  - Other sample of botnets shows over 40/300 with no DNS.

# Measuring Botnet Size



Count SYNs coming into the sinkhole.

Claim is that these SYNs are the result of bots trying to connect.

# True size?

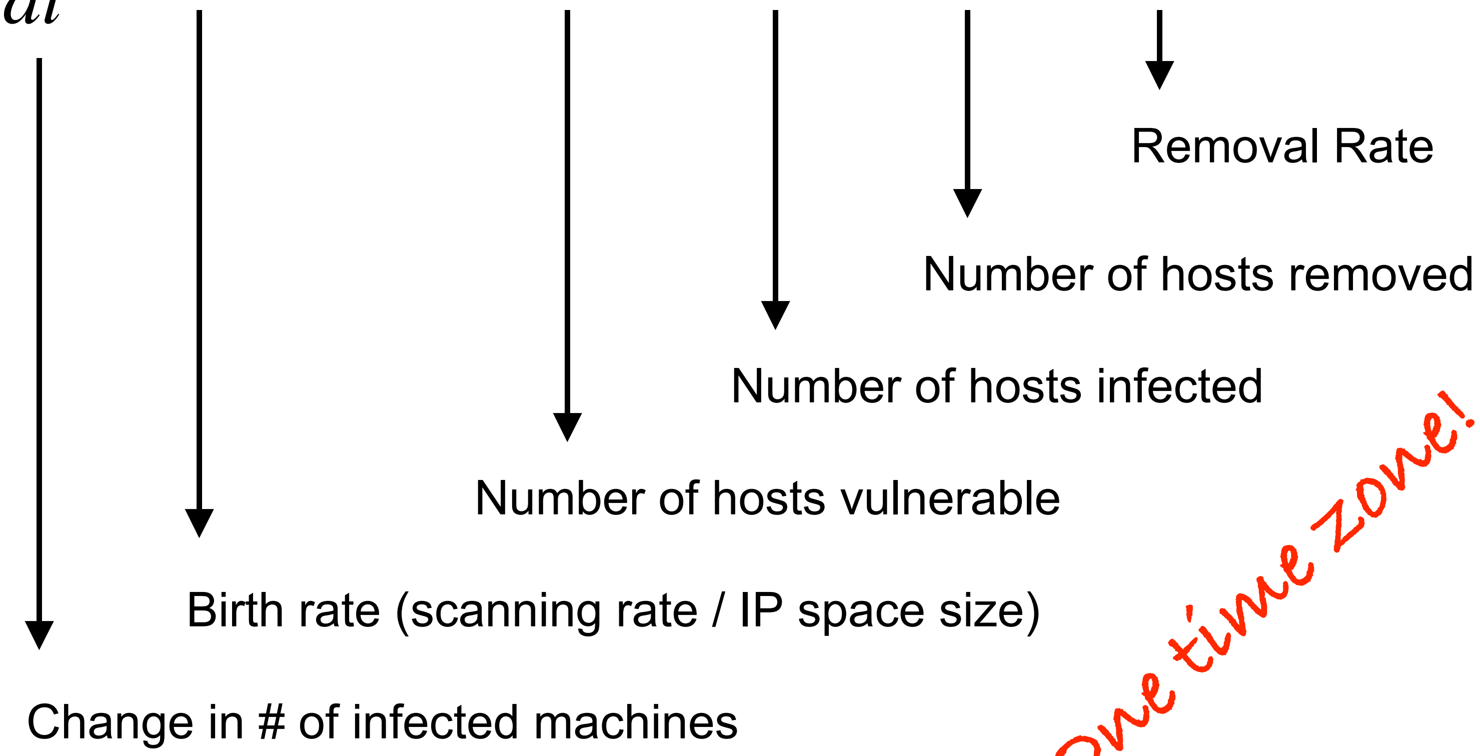
- Need to look at the application (IRC) layer to be sure of the actual size.
- Would require creating an IRC-like server at the end of the sinkhole.
- Is this really representative of the true botnet size??

# The diurnal model

- Start with Epidemiological Model
- Add $\alpha(t)$ function
  - "*diurnal shaping function*"
  - Fraction of vulnerable computers in time zone *t*, due to powered off PCs
  - $\alpha(t)$ peak at midday, valley at night
  - Use observed traffic to calculate $\alpha(t)$

# The Diurnal Model

$$\frac{dI(t)}{dt} = \beta\alpha^2(t)I(t)[N(t) - I(t) - R(t)] - \gamma\alpha(t)I(t)$$



Removal Rate

Number of hosts removed

Number of hosts infected

Number of hosts vulnerable

Birth rate (scanning rate / IP space size)

Change in # of infected machines

One time zone!

# Multiple Time Zones

Consider how all time zones *j* affect **one** time zone *i*.

$$\frac{dI_i(t)}{dt} = \alpha_i(t)[N(t) - I(t) - R(t)]\underbrace{\sum_{j=1}^{K} \beta_{ji}\alpha_j(t) I_j(t)} - \gamma_i \alpha_i I_i(t)$$

Birth rate has to consider that scans may be coming from different time zones, sum all possibilities.

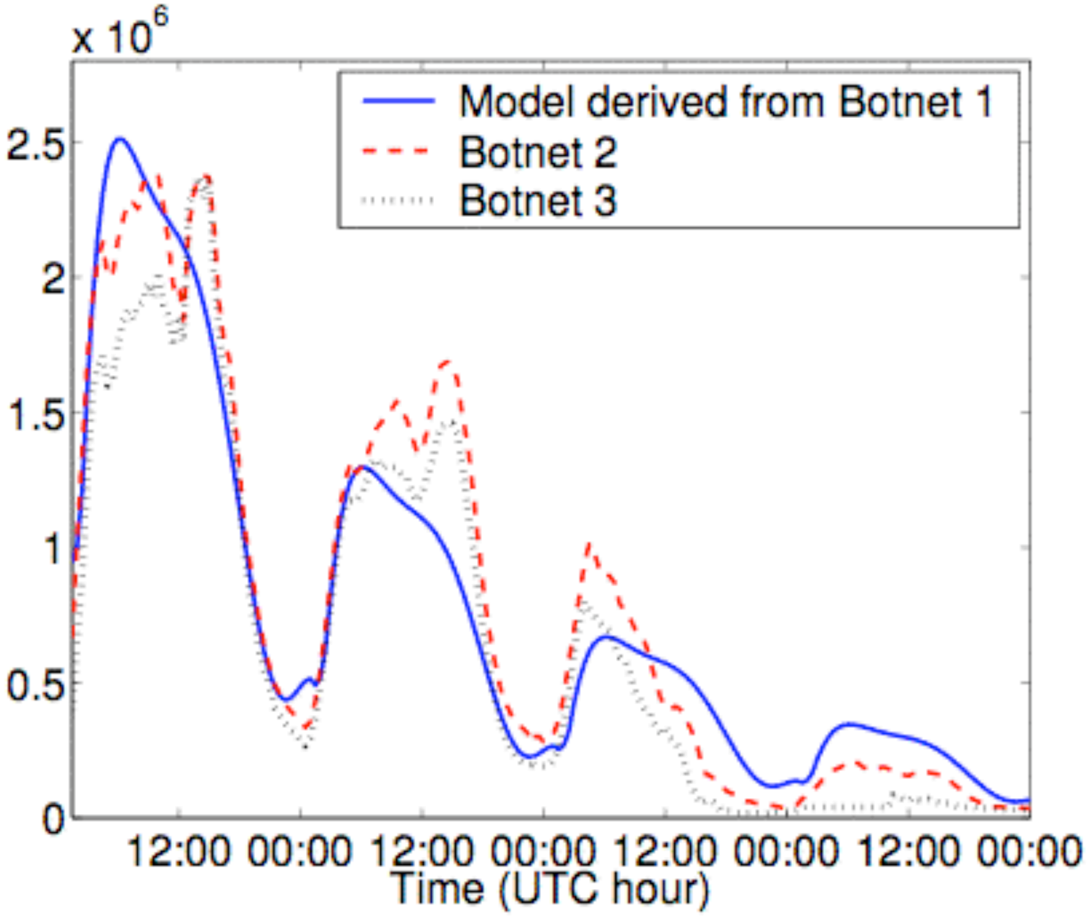
# Diurnal Model Motivation

- Is an updated model necessary?
  - Authors claim we may better choose which outbreak to focus on
- Does this really apply to botnets?
  - Botnets can change activities on the fly!

# Is the model right?

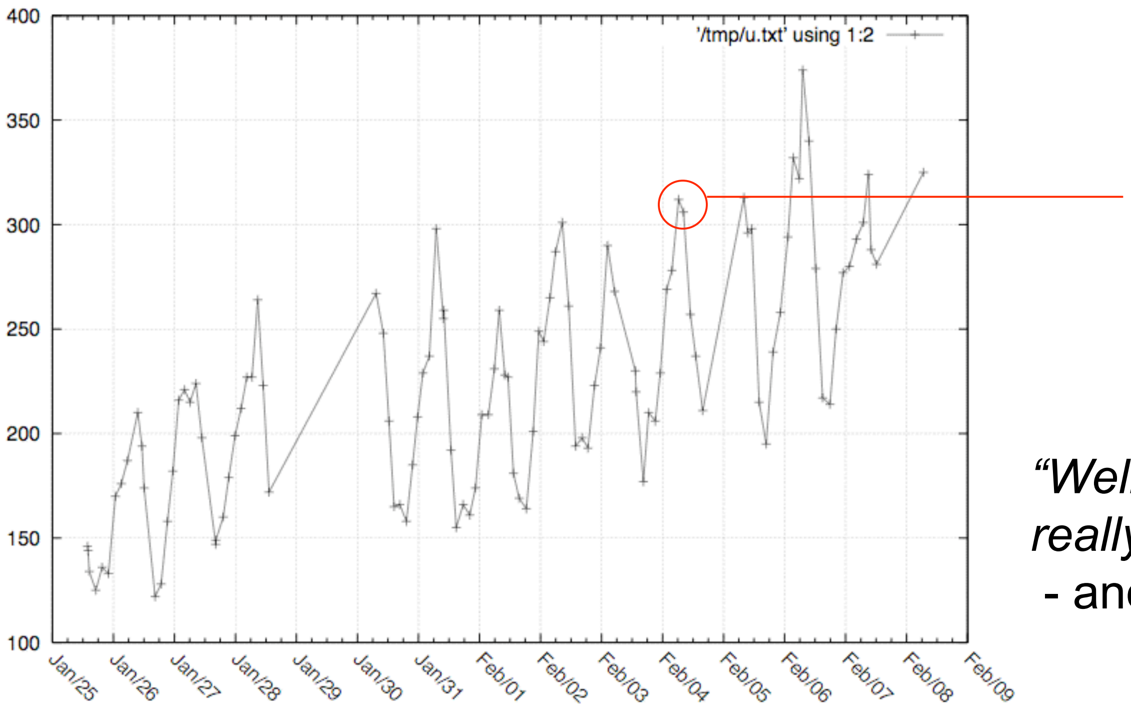
Well, the graphs look good...

Data seems to fit relatively well...



# Alternative method

We joined a botnet server, let it tell us how many online users (infected bots) via its 'welcome' message.

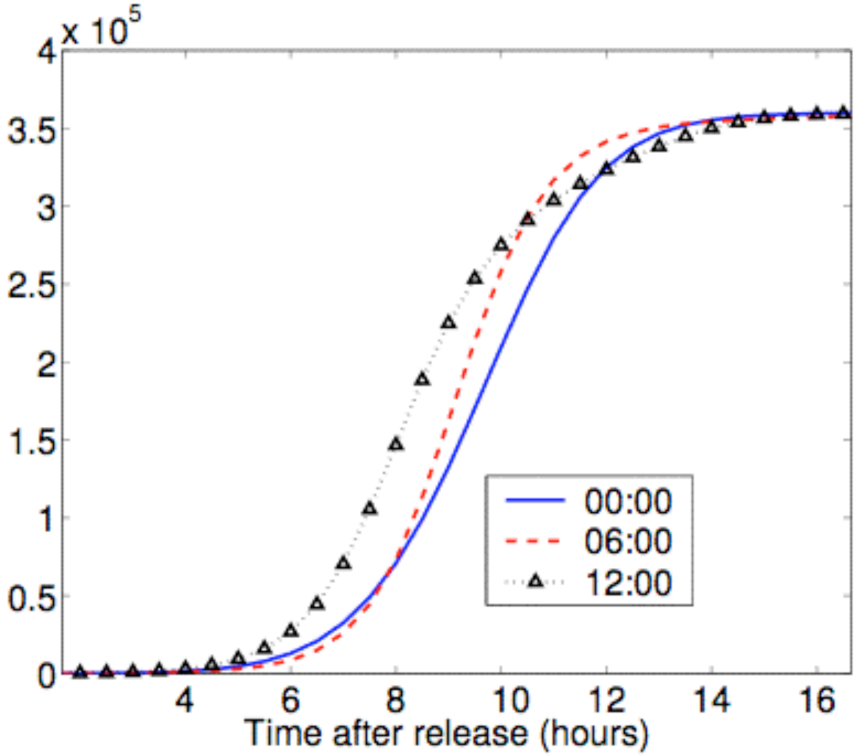


Definitely one peak and one valley per day.

*"Well I'll be damned. There really IS a diurnal pattern."*
- anonymous

# What does model say?

Optimal time to release worm (launch bot scans):



If we believe this, then release time doesn’t matter all that much.

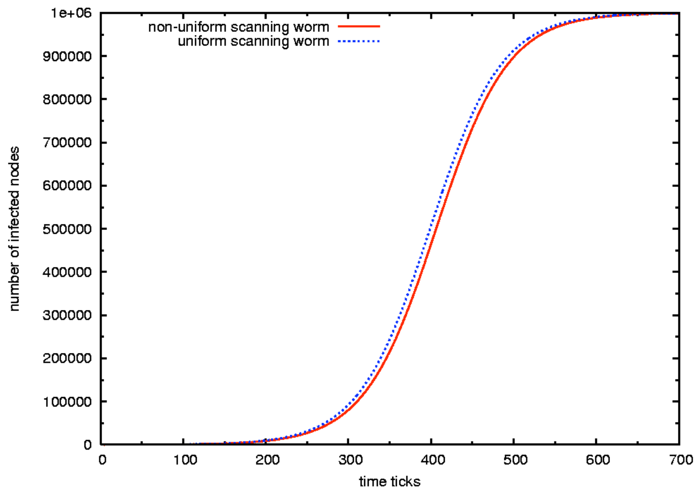
# Is this model complete?

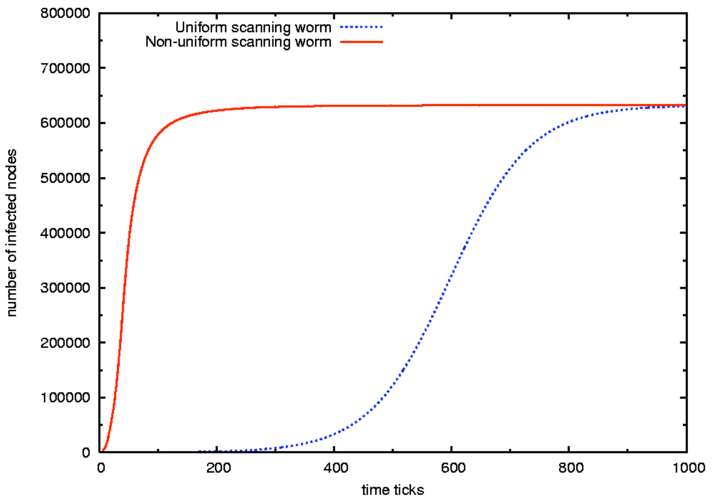
- Current Diurnal Model
  - Is continuous, not discrete
  - Assumes uniform distribution of vulnerable hosts (we know this is false)
  - **Assumes that bot-related scans are performed uniformly (also false!)**

# Assumptions Matter!!!

Remember what happened when we changed one “little” assumption about the distribution of the vulnerable population?

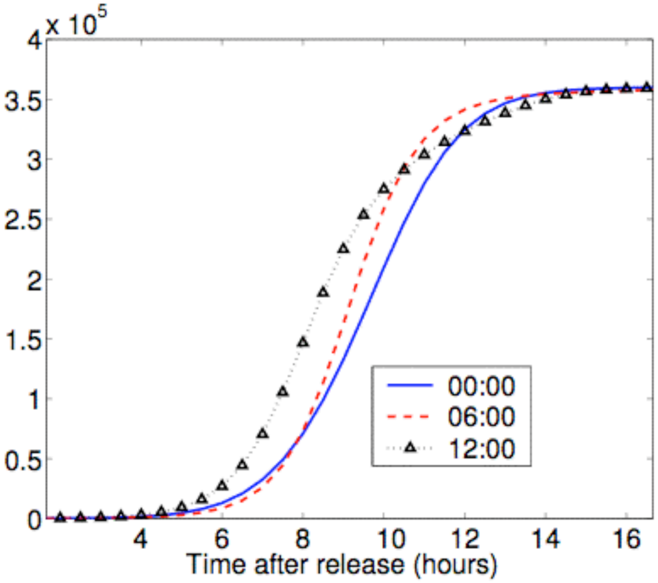
Propagation rates for non-uniform scanning worms changed **drastically**.





# Assumptions Matter!!

“Best” release time



Assume non-uniform scanning

?

Assume non-uniform victim population distribution

?

# Making better assumptions

- We saw last week the importance of assumptions about our adversary
- This week, we see the importance of assumptions in bot behavior.
- How important would time zones be if we changed our assumptions?

# Other good resources

- The Honeynet Project
  - www.honeynet.org
- Know Your Enemy: Tracking Botnets
  - www.honeynet.org/papers/bots/
- Botnets as a Vehicle for Online Crime
  - www.cert.org/archive/pdf/Botnets.pdf
- Moheeb
  - Down the hall

# Papers, papers, papers

Evan Cooke, et al. **The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets.** Proceedings of *SRUTI 2005.*

Felix C. Freiling, et al**. Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks.** *ESORICS 2005.*

Barford, Paul and Yegneswaran, Vinod. **A Look Inside Botnets.** To appear in *Advances in Information Security, Springer, 2006*.