12. Network Attacks

Blase Ur and David Cash
(many slides borrowed from Ben Zhao, Christo Wilson, & others)
February 7\textsuperscript{th}, 2020
CMSC 23200 / 33250
Network threat model

• Network scanning

• Attacks on confidentiality (e.g., eavesdropping)

• Attacks on integrity (e.g., spoofing, packet injection)

• Attacks on availability (e.g., denial of service (DoS))
Scanning and observing networks
Network Scanning: Ping

- Essential, low-level network utility
- Sends a “ping” ICMP message to a host on the internet
  
  ```
  $ ping 66.66.0.255
  PING 66.66.0.255 (66.66.0.255) 56(84) bytes of data.
  64 bytes from 66.66.0.255: icmp_seq=1 ttl=58 time=41.2 ms
  ```

- Destination host is supposed to respond with a “pong”
  - Indicating that it can receive packets

- By default, ping messages are 56 bytes long (+ some header bytes)
  - Maximum size 65535 bytes

- What if you send a ping that is >65535 bytes long?
Ping of Death

- $ ping –s 65535 66.66.0.255
  - Attack identified in 1997
  - IPv6 version identified/fixed in 2013
Network Scanning: Traceroute

- traceroute — hops between me and host
  - Sends repeated ICMP reqs w/ increasing TTL

```
thor Wed Oct 24 (12:51am) [~]: --> traceroute www.slack.com
traceroute to www.slack.com (52.85.115.213), 64 hops max, 52 byte packets
1  v11router (128.135.11.1)  1.265 ms  0.788 ms  0.778 ms
2  a06-021-100-to-d19-07-200.p2p.uchicago.net (10.5.1.186)  1.292 ms  0.749 ms  0.833 ms
3  d19-07-200-to-h01-391-300.p2p.uchicago.net (10.5.1.46)  2.124 ms  2.435 ms  2.072 ms
4  192.170.192.34 (192.170.192.34)  0.755 ms
   192.170.192.32 (192.170.192.32)  0.810 ms  0.701 ms
5  192.170.192.36 (192.170.192.36)  0.887 ms  0.918 ms  0.877 ms
6  r-equinix-isp-ae2-2213.wiscnet.net (216.56.50.45)  1.625 ms  1.803 ms  1.866 ms
7  * * *
8  * * *
9  * * *
10 * * *
11 178.236.3.103 (178.236.3.103)  4.516 ms  4.326 ms  4.320 ms
12 * * *
13 * * *
14 * * *
15  server-52-85-115-213.ind6.r.cloudfront.net (52.85.115.213)  4.554 ms  4.398 ms  4.757 ms
thor Wed Oct 24 (12:52am) [~]: -->
```
Port Scanning

- What services are running on a server? Nmap

```

Starting Nmap 7.01 ( https://nmap.org ) at 2018-10-24 00:55 CDT
Nmap scan report for www.cs.uchicago.edu (34.203.108.171)
Host is up (0.019s latency).
Other addresses for www.cs.uchicago.edu (not scanned): 54.164.17.80 54.85.61.218
rDNS record for 34.203.108.171: ec2-34-203-108-171.compute-1.amazonaws.com
Not shown: 998 filtered ports
PORT    STATE   SERVICE
80/tcp  open    http
443/tcp open    https

Nmap done: 1 IP address (1 host up) scanned in 4.99 seconds
```

- 5 seconds to scan a single machine!!
SYN scan

Only send SYN

Responses:

• SYN-ACK — port open
• RST — port closed
• Nothing — filtered (e.g., firewall)
Port Scanning on Steroids

• How do you speed up scans for all IPv4?
  – Don’t wait for responses; pipeline
  – Parallelize: divide & conquer IPv4 ranges
  – Randomize permutations w/o collisions

• Result: the zmap tool
  – Scan all of IPv4 in 45mins (w/ GigE cxn)
  – IPv4 in 5 mins w/ 10GigE
Eavesdropping

Tools: Wireshark, tcpdump, Bro, …

Steps:

1. Parse data link layer frames
2. Identify network flows
3. Reconstruct IP packet fragments
4. Reconstruct TCP connections
5. Parse app protocol messages
Wireshark, Detailed Protocol Analyzer

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>24.4.97.251</td>
<td>68.87.76.178</td>
<td>DNS</td>
<td>DNS</td>
<td>76</td>
<td>Standard query 0x6bc0 A <a href="http://www.symantec.com">www.symantec.com</a></td>
</tr>
<tr>
<td>2</td>
<td>24.4.97.251</td>
<td>68.87.76.178</td>
<td>DNS</td>
<td>DNS</td>
<td>262</td>
<td>Standard query response 0x6bc0 CNAME <a href="http://www.symantec.d4p.net">www.symantec.d4p.net</a> CNAME s</td>
</tr>
<tr>
<td>3</td>
<td>24.4.97.251</td>
<td>68.87.76.178</td>
<td>DNS</td>
<td>DNS</td>
<td>93</td>
<td>Standard query 0xcdc6 A liveupdate.symantecliveupdate.com</td>
</tr>
<tr>
<td>4</td>
<td>24.4.97.251</td>
<td>68.87.76.178</td>
<td>DNS</td>
<td>DNS</td>
<td>286</td>
<td>Standard query response 0xcdc6 CNAME liveupdate.symantecliveupdate.com</td>
</tr>
<tr>
<td>5</td>
<td>24.4.97.251</td>
<td>80.231.19.118</td>
<td>TCP</td>
<td>TCP</td>
<td>62</td>
<td>trim &gt; http [SYN] Seq=0 Win=65535 Len=0 MSS=1460 SACK_PERM-1</td>
</tr>
<tr>
<td>6</td>
<td>80.231.19.118</td>
<td>24.4.97.251</td>
<td>TCP</td>
<td>TCP</td>
<td>62</td>
<td>http &gt; trim [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SACK_PERM-1</td>
</tr>
<tr>
<td>7</td>
<td>80.231.19.118</td>
<td>24.4.97.251</td>
<td>TCP</td>
<td>TCP</td>
<td>54</td>
<td>trim &gt; http [ACK] Seq=1 Ack=1 Win=65535 Len=0</td>
</tr>
<tr>
<td>8</td>
<td>80.231.19.118</td>
<td>80.231.19.118</td>
<td>TCP</td>
<td>TCP</td>
<td>54</td>
<td>trim &gt; http [ACK] Seq=1 Ack=254 Win=6432 Len=0</td>
</tr>
<tr>
<td>9</td>
<td>80.231.19.118</td>
<td>24.4.97.251</td>
<td>HTTP</td>
<td>HTTP</td>
<td>307</td>
<td>GET /minitri.flg HTTP/1.1</td>
</tr>
<tr>
<td>10</td>
<td>24.4.97.251</td>
<td>80.231.19.118</td>
<td>TCP</td>
<td>TCP</td>
<td>54</td>
<td>trim &gt; http [ACK] Seq=254 Ack=235 Win=65301 Len=0</td>
</tr>
<tr>
<td>11</td>
<td>80.231.19.118</td>
<td>24.4.97.251</td>
<td>TCP</td>
<td>TCP</td>
<td>54</td>
<td>trim &gt; http [ACK] Seq=254 Ack=235 Win=65301 Len=0</td>
</tr>
<tr>
<td>12</td>
<td>80.231.19.118</td>
<td>20.86.310000</td>
<td>HTTP</td>
<td>HTTP</td>
<td>298</td>
<td>GET /automatic20liveupdate_3.0.0.171_english_livetri.zip HTTP/1.1</td>
</tr>
</tbody>
</table>

Boo!
Protocol attacks
Active Attacks: Blind Spoofing

Mallory

src: Alice’s IP, SYN, seq = x

Server

SYN-ACK, ack x+1, seq = y

Alice

src: Alice’s IP, ACK, ack = y+1

Guess y (server’s sequence number) to open forged connection

• Originally: y based on time
• Defense: pseudorandom y
RST Hijacking

If Mallory knows $y$, she has $\frac{1}{2^{32}}$ chance of guessing $p$ & closing connection $\rightarrow$ flood with RSTs

TCP Reset attacks used widely for censorship, e.g., Great Firewall
Inter-domain routing (BGP) attacks and large-scale observation
Recall: BGP: a Path-Vector Protocol

- An AS-path: sequence of AS’s a route traverses
- Used for loop detection and to apply policy
BGP Prefix Hijacking

• Advertise a more desirable route even if the route isn’t actually more desirable, or even real

• Goal 1: Route traffic through networks you control so that you can observe the traffic

• Goal 2: Send lots of traffic to someone you don’t like (denial of service)
Corrigendum- Most Urgent

GOVERNMENT OF PAKISTAN
PAKISTAN TELECOMMUNICATION AUTHORITY
ZONAL OFFICE PESHAWAR
Plot-11, Sector A-3, Phase-V, Hayatabad, Peshawar.
Ph: 091-9217279- 5829177 Fax: 091-9217254
www.pta.gov.pk

NWFP-33-16 (BW)/06/PTA

Subject: Blocking of Offensive Website

Reference: This office letter of even number dated 22.02.2008.

I am directed to request all ISPs to immediately block access to the following website

URL: http://www.youtube.com/watch?v=o3s8jtvvg00

IPs: 208.65.153.238, 208.65.153.253, 208.65.153.251

Compliance report should reach this office through return fax or at email peshawar@pta.gov.pk today please.

To:
1. M/s Comsats, Peshawar.
2. M/s GOL Internet Services, Peshawar.
3. M/s Cyber Internet, Peshawar.
5. M/s Paknet, Limited, Islamabad
7. M/s Supernet, Peshawar.

Deputy Director
(Enforcement)
BGP Prefix Hijacking

How a Nigerian ISP Accidentally Hijacked the Internet

For 74 minutes, traffic destined for Google and Cloudflare services was routed through Russia and into the largest system of censorship in the world, China’s Great Firewall.

On November 12, 2018, a small ISP in Nigeria made a mistake while updating its network infrastructure that highlights a critical flaw in the fabric of the Internet. The mistake effectively brought down Google — one of the largest tech companies in the world — for 74 minutes.

To understand what happened, we need to cover the basics of how Internet routing works. When I type, for example, HypotheticalDomain.com into my browser and hit enter, my computer creates a web request and sends it to HypotheticalDomain.com servers. These servers likely reside in a different state or country than I do. Therefore, my Internet service provider (ISP) must determine how to route my web browser’s request to the server across the Internet. To maintain their routing tables, ISPs and Internet backbone companies use a protocol called Border Gateway Protocol (BGP).

From Snowden archives, dated April 2013

FAA702 Operations

Two Types of Collection

Upstream

- Collection of communications on fiber cables and infrastructure as data flows past.
  (FAIRVIEW, STORMBREW, BLARNEY, OAKSTAR)

PRISM

- Collection directly from the servers of these U.S. Service Providers: Microsoft, Yahoo, Google, Facebook, PalTalk, AOL, Skype, YouTube, Apple.

You Should Use Both
PRISM Collection Details

What Will You Receive in Collection (Surveillance and Stored Comms)?
It varies by provider. In general:

- E-mail
- Chat – video, voice
- Videos
- Photos
- Stored data
- VoIP
- File transfers
- Video Conferencing
- Notifications of target activity – logins, etc.
- Online Social Networking details
- Special Requests

Current Providers

- Microsoft (Hotmail, etc.)
- Google
- Yahoo!
- Facebook
- PalTalk
- YouTube
- Skype
- AOL
- Apple

Complete list and details on PRISM web page:
Go PRISMFAA

TOP SECRET//SI//ORCON//NOFORN
Dates When PRISM Collection Began For Each Provider

- Microsoft: 9/11/07
- Yahoo: 3/12/08
- Google: 1/14/09
- Facebook: 6/3/09
- PalTalk: 9/24/10
- Skype: 3/31/11
- YouTube: 2/6/11
- AOL: 2012
- Apple: (added Oct 2012)

PRISM Program Cost: ~$20M per year
S-BGP / BGPsec

IP prefix announcements signed

Routes signed
— previous hop authorizes next hop

Higher levels vouch for lower levels
— e.g., ICANN vouches for ARIN, ARIN vouches for AT&T, …

Problem?
Costly and slow adoption
DNS attacks
DNS Cache Poisoning

1. The IP address of example.jp is "172.16.3.2"

2. I’d like to access http://example.jp/. What’s its IP address?

3. The IP address of example.jp is "172.16.3.2"

If the DNS cache server is “poisoned”, the user is directed to a bogus website even if the URL specified is correct.
DNS Cache Poisoning (cont.)

Defense: randomize 16-bit QID

Alice
Q: www.bank.com
QID: x

Local DNS resolver

Mallory
spoof src IP of ns.bank.com
A: 3.3.3.3
guess QID: x

Race

ns.bank.com
A: 2.2.2.2
QID: x
Kaminsky attack (2008)

Alice runs JavaScript from mallory.com

Q: a.bank.com
Q: b.bank.com
Q: c.bank.com

Local DNS resolver

Spoof entire *.bank.com zone

ns.bank.com

Mallory wins if any $r_i = s_j$
DNSSEC

DNS responses signed

Higher levels vouch for lower levels — e.g., root vouches for .edu, .edu vouches for .uchicago, …

Root public key published

Problem?
Costly and slow adoption
The Coffeeshop Attack Scenario

- DNS servers bootstrapped by wireless AP
  - (default setting for WiFi)
- Attacker hosts AP w/ ID (O’Hare Free WiFi)
  - You connect w/ your laptop
  - Your DNS requests go through attacker DNS
    - www.bofa.com → evil bofa.com
  - Password sniffing, malware installs, …

- TLS/SSL certificates to the rescue!
Denial of Service (Attacks on Availability)
Denial of Service (DoS)

- Prevent users from being able to access a specific computer, service, or piece of data
- In essence, an attack on availability
- Possible vectors:
  - Exploit bugs that lead to crashes
  - Exhaust the resources of a target
- Often very easy to perform...
- … and fiendishly difficult to mitigate
DoS Attacker Goals & Threat Model

- Active attacker who may send arbitrary packets
- Goal is to reduce the availability of the victim

I wanna knock those servers offline... but how?
DoS Attack Parameters

• How much bandwidth is available to the attacker?
  – Can be increased by controlling more resources…
  – Or tricking others into participating in the attack

• What kind of packets do you send to victim?
  – Minimize effort and risk of detection for attacker…
  – While also maximizing damage to the victim
Standard DDoS, Revisited

- What kind of packets do you send to the victim?
- Ideally, should be “connectionless”
  - Difficult to spoof TCP connections
- Should maximize the resources used by the victim
TCP SYN Flood

- TCP stack keeps track of connection state in data structures called Transmission Control Blocks (TCBs)
  - New TCB allocated by the kernel whenever a listen socket receives a SYN
  - TCB must persist for at least one RTO

- Attack: flood the victim with SYN packets
  - Exhaust available memory for TCBs, prevent legitimate clients from connecting
  - Crash the server OS by overflowing kernel memory

- Advantages for the attacker
  - No connection – each SYN can be spoofed, no need to hear responses
  - Asymmetry – attacker does not need to allocate TCBs
Exploiting Asymmetry

- Example of a Distributed Denial of Service Attack (DDoS)
- Some DDoS is fueled by volunteers
  - E.g. Anonymous and Low Orbit Ion Canon (LOIC)
- Most DDoS is fueled by botnets
The Smurf Attack

PING Request
Src: 128.91.0.1
Dst: 10.7.0.255

*.*.*.255 is a broadcast packet
Forwarded to all hosts in the /24
Why Does Smurfing Work?

1. ICMP protocol does not include authentication
   - No connections
   - Receivers accept messages without verifying the source
   - Enables attackers to spoof the source of messages

2. Attacker benefits from an amplification factor
   \[
   \text{amp factor} = \frac{\text{total response size}}{\text{request size}}
   \]
Reflection/Amplification Attacks

- Smurfing is an example of a reflection or amplification DDoS attack.
- Fraggle attack similarly uses broadcasts for amplification:
  - Send spoofed UDP packets to IP broadcast addresses on port 7 (echo) and 13 (chargen):
    - echo – 1500 bytes/pkt requests, equal size responses
    - chargen -- 28 bytes/pkt request, 10K-100K bytes of ASCII in response
  - Amp factor:
    - echo – [number of hosts responding to the broadcast]:1
    - chargen – [number of hosts responding to the broadcast]*360:1
DNS Reflection Attack

- Spoof DNS requests to many open DNS resolvers
  - DNS is a UDP-based protocol, no authentication of requests
  - Open resolvers accept requests from any client
    - E.g. 8.8.8.8, 8.8.4.4, 1.1.1.1, 1.0.0.1
    - February 2014 – 25 million open DNS resolvers on the internet
- 64 byte DNS queries generate large responses
  - Old-school “A” record query → maximum 512 byte response
  - EDNS0 extension “ANY” record query → 1000-6000 byte response
    - E.g. $ dig ANY isc.org
  - Amp factor – 180:1
- Attackers have been known to register their own domains and install very large records just to enable reflection attacks!
Reflection Example

DNS Request
Src: 128.91.0.1
Dst: whatever
NTP Reflection Attack

• Spoof requests to open Network Time Protocol (NTP) servers
  – NTP is a UDP-based protocol, no authentication of requests
  – May 2014 – 2.2 million open NTP servers on the internet

• 234 byte queries generate large responses
  – *monlist* query: server returns a list of all recent connections
  – Other queries are possible, i.e. *version* and *showpeers*
  – Amp factor – from 10:1 to 560:1
memcached Reflection Attack

• Spoof requests to open memcached servers
  – Popular <key:value> server used to cache web objects
  – memcached uses a UDP-based protocol, no authentication of requests
  – February 2018 – 50k open memcached servers on the internet

• 1460 byte queries generate large responses
  – A single query can request multiple 1MB <key:value> pairs from the database
  – Amp factor – up to 50000:1
# Infamous DDoS Attacks

<table>
<thead>
<tr>
<th>When</th>
<th>Against Who</th>
<th>Size</th>
<th>How</th>
</tr>
</thead>
<tbody>
<tr>
<td>March 2013</td>
<td>Spamhaus</td>
<td>120 Gbps</td>
<td>Botnet + DNS reflection</td>
</tr>
<tr>
<td>February 2014</td>
<td>Cloudflare</td>
<td>400 Gbps</td>
<td>Botnet + NTP reflection</td>
</tr>
<tr>
<td>September 2016</td>
<td>Krebs</td>
<td>620 Gbps</td>
<td>Mirai</td>
</tr>
<tr>
<td>October 2016</td>
<td>Dyn (major DNS provider)</td>
<td>1.2 Tbps</td>
<td>Mirai</td>
</tr>
<tr>
<td>March 2018</td>
<td>Github</td>
<td>1.35 Tbps</td>
<td>Botnet + memcached reflection</td>
</tr>
</tbody>
</table>
Content Delivery Networks (CDNs)

- CDNs help companies scale-up their websites
  - Cache customer content on many replica servers
  - Users access the website via the replicas

- Examples: Akamai, Cloudflare, Rackspace, Amazon Cloudfront, etc.

- Side-benefit: DDoS protection
  - CDNs have many servers, and a huge amount of bandwidth
  - Difficult to knock all the replicas offline
  - Difficult to saturate all available bandwidth
  - No direct access to the master server

- Cloudflare: 15 Tbps of bandwidth over 149 data centers
CDN Basics

Website content and database is here

- Users requests all go through the replicas
- Most served from cache

Content is cached in the replicas
DDoS Defense via CDNs

- What if you DDoS the master replica?
  - Cached copies in the CDN still available
  - Easy to do ingress filtering at the master

- What if you DDoS the replicas?
  - Difficult to kill them all
  - Dynamic DNS can redirect users to live replicas
Internet Crime as a Financial Ecosystem

As the Internet evolved, so did cybercrime...

- Stolen Account Credentials
- Credit Card and Bank Account Theft
- DDoS and Ransomware Extortion
- Blackhat SEO
- Spam
- Click Fraud and Ad Injection
- Bitcoin Mining
- Carders, Cashiers, and Money Mules
- Phishing
- Pharma
- Counterfeit Goods
- Fake Anti-virus
- Malware Attachments

- Pay-per-Install and Exploit-as-a-Service
- Botnets
- Zero-day Development
- Crimeware Development

- Bulletproof Hosting
Drive-by Exploits

• Browsers are extremely complex
  – Millions of lines of source code
  – Rely on equally complex plugins from 3rd party developers
    • *e.g.* Adobe Flash, Microsoft Silverlight, Java
• Must deal with untrusted, complex inputs
  – Network packets from arbitrary servers
  – HTML/XML, JavaScript, stylesheets, images, video, audio, etc.
• Recipe for disaster
  – Attacker directs victim to website containing malicious content
  – Leverage exploits in browser to attack OS and gain persistence
Modern Browser Architecture

- Browsers handle many types of complex input
  - HTML/XML
  - JavaScript
  - Stylesheets
  - Images/video/audio
  - Java and Flash bytecode
- Parsing bugs may be exploitable
- JavaScript gives attackers the ability to stage exploits
New HTML page with some JavaScript inside

Shellcode

Target address

Malformed XML data that triggers a buffer overflow

Heap spraying: fill memory with copies of the shellcode to increase chances of successful exploitation

Trigger the overflow by injecting the bugged XML into the HTML page
Executing a Drive-by

• Host exploits on a *bulletproof host*
  – No need to distribute (expensive) exploit code to other websites
  – Resist law enforcement takedowns

• Victim acquisition
  – Spam containing links (email, SMS, messenger)
  – Compromise legitimate websites & add booby-traps (*e.g.* via XSS)
    • Hidden *iframes* that load exploit website
    • Force a redirect to the exploit website
Blackhole malware kit, released in 2010, dominated market in 2012-2013

Annual license of $1500, or $200/week, targeted Java, Flash, Windows, PDFs

Suspect arrested in Oct 2013
# Exploits Used by Blackhole

<table>
<thead>
<tr>
<th>CVE</th>
<th>Target</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVE-2011-2110</td>
<td>Flash</td>
<td>Adobe Flash Player unspecified code execution</td>
</tr>
<tr>
<td>CVE-2011-0611</td>
<td>Flash</td>
<td>Adobe Flash Player unspecified code execution</td>
</tr>
<tr>
<td>CVE-2010-3552</td>
<td>Java</td>
<td>Skyline</td>
</tr>
<tr>
<td>CVE-2010-1885</td>
<td>Windows</td>
<td>Microsoft Windows Help and Support Center</td>
</tr>
<tr>
<td>CVE-2010-1423</td>
<td>Java</td>
<td>Java Development Toolkit insufficient argument validation</td>
</tr>
<tr>
<td>CVE-2010-0886</td>
<td>Java</td>
<td>Unspecified vulnerability</td>
</tr>
<tr>
<td>CVE-2010-0842</td>
<td>Java</td>
<td>JRE MixerSequencer invalid array index</td>
</tr>
<tr>
<td>CVE-2010-0840</td>
<td>Java</td>
<td>Java trusted methods chaining</td>
</tr>
<tr>
<td>CVE-2010-0188</td>
<td>Adobe Acrobat</td>
<td>LibTIFF integer overflow</td>
</tr>
<tr>
<td>CVE-2010-4324</td>
<td>Adobe Acrobat</td>
<td>Use after free vulnerability in doc.media.newPlayer</td>
</tr>
</tbody>
</table>
The backbone of the underground

BOTNETS
From Crimeware to Botnets

• Infected machines are a fundamentally valuable resource
  – Unique IP addresses for spamming
  – Bandwidth for DDoS
  – CPU cycles for bitcoin mining
  – Credentials

• Early malware monetized these resources directly
  – Infection and monetization were tightly coupled

• Botnets allow criminals to rent access to infected hosts
  – Infrastructure as a service, i.e. the cloud for criminals
  – Command and Control (C&C) infrastructure for controlling bots
  – Enables huge-scale criminal campaigns
Old-School C&C: IRC Channels

- Problem: single point of failure
- Easy to locate and take down

 snd spam: `<subject> <msg>`

 snd spam: `<subject> <msg>`

 snd spam: `<subject> <msg>`
Fast Flux DNS

But: ISPs can blacklist the rendezvous domain

Change DNS → IP mapping every 10 seconds
Domain Name Generation (DGA)

Bots generate many possible domains each day

HTTP Servers

Botmaster

...But the Botmaster only needs to register a few

www.sb39fwn.com  www.17-cjbq0n.com  www.xx8h4d9n.com

Can be combined with fast flux
“Your Botnet is My Botnet”

• Takeover of the Torpig botnet
  – Random domain generation + fast flux
  – Team reverse engineered domain generation algorithm
  – Registered 30 days of domains before the botmaster!
  – Full control of the botnet for 10 days

• Goal of botnet: credential theft and phishing spam
  – Steals credit card numbers, bank accounts, etc.
  – Researchers gathered all this data

• Other novel point: accurate estimation of botnet size
Torpig Architecture

Attacker places a redirect on the vulnerable server

Vulnerable web server

<iframe>

GET / (1)

GET / (2)

Victim client

Rootkit installation

Mebroot drive-by-download server

GET/?gnh5 (3)

gnh5.exe (4)

(becomes a bot)

Trojan installation

Mebroot C&C server

Collect stolen data

Torpig C&C server

Collect stolen data

Stolen data (5)

Config

Torpig DLLs

Phishing HTML

Injection server

Capture banking passwords

Researchers Infiltrated Here
Stopping Botnets

• Individual perspective: ridding your network of bots
  – Anti-virus and anti-malware
  – Intrusion and anomaly detection to identify infections, block traffic

• Global perspective: takedowns and arrests
  – Create a sinkhole (fake C&C server)
  – Track down and arrest the perpetrators
<table>
<thead>
<tr>
<th>Botnet Name</th>
<th>Timeframe</th>
<th>Estimated Size</th>
<th>Taken Down by...</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNS Changer</td>
<td>2006-2011</td>
<td>4M</td>
<td>FBI, Trend Micro</td>
</tr>
<tr>
<td>Rustock</td>
<td>2006-2011</td>
<td>150K-2.4M</td>
<td>FBI, Microsoft, Fireeye, Univ. of Washington</td>
</tr>
<tr>
<td>Grum</td>
<td>2008-2012</td>
<td>560K-840K</td>
<td>Fireeye, Spamhaus</td>
</tr>
<tr>
<td>Conficker</td>
<td>2008-2009</td>
<td>4M-13M</td>
<td>FBI, Microsoft, Symantec, ICANN</td>
</tr>
<tr>
<td>Citadel</td>
<td>2011-2013</td>
<td></td>
<td>FBI, Microsoft</td>
</tr>
<tr>
<td>Gameover Zeus/Cryptolocker</td>
<td>2012-2014</td>
<td></td>
<td>DoJ, FBI, Europol, Dell, Microsoft, Level3, McAfee, Symantec, Sophos, Trend Micro, Carnegie Mellon, Georgia Tech, etc.</td>
</tr>
<tr>
<td>SIMDA</td>
<td>2011-2015</td>
<td>770K</td>
<td>INTERPOL, Trend Micro, Microsoft, Kaspersky Lab</td>
</tr>
<tr>
<td>DRIDEX</td>
<td>2014-2015</td>
<td></td>
<td>FBI, Trend Micro</td>
</tr>
<tr>
<td>Avalanche</td>
<td>2009-2016</td>
<td>500K</td>
<td>FBI, Symantec, Fraunhofer</td>
</tr>
</tbody>
</table>
Scratching the Surface of the Underground

- **Zero-days**
  - The competitive market for fresh exploits

- **Search Engine Optimization (SEO)**
  - Attempt to push garbage results to the top of Google search

- **Click fraud and ad injection**
  - Steal money from legitimate advertisers

- **Bitcoin mining (Botcoin)**
  - Steal CPU cycles from infected hosts to mint currency

- **CATPCHA-solving services**
  - Employ real people to solve CAPTCHAs for a small fee

- **Crowdturfing**
  - Employ real people to create fake accounts (*Sybils* or *sock puppets*)
  - Perform phone and email verification so accounts look legitimate
A Pragmatic Perspective

• Evidence shows cybercrime market large & profitable

• But not as bad as some commentators claim
  – The cybercrime underground not a billion dollar industry
  – Botnets almost never control tens of millions of hosts

• Cybercrime huge problem due to asymmetry
  – Example: spam
    • Criminals may spend millions of dollars sending spam per year
    • Industry spends billions of dollars / year on spam defense
  – An attacker can strike anywhere around the globe at any time
  – Barriers to entry are low, costs are easily offset by profits
  – Arrests are uncommon