CSC 405
Introduction to Computer Security

Botnets and Cybercrime

Alexandros Kapravelos
akaprav@ncsu.edu

(Derived from slides by Chris Kruegel)
Botnets

• Bot
  – autonomous programs performing tasks
  – more recent trend in malicious code development

• Benign bots
  – first bots were programs used for Internet Relay Chat (IRC)
  – react to events in IRC channels
  – typically offer useful services

• Early definition of bot

  An IRC user who is actually a program. On IRC, typically the robot provides some useful service. Examples are NickServ, which tries to prevent random users from adopting nicks already claimed by others.
Botnets

• Eggdrop bot (1993)
  – used to manage IRC chat channels when operator away
    (still maintained, eggheads.org)

• Malicious IRC bots started to evolve
  – takeover wars to control certain IRC channels
  – trash talking (flooding)
  – also involved in denial of service to force IRC netsplit
  – IRC proxies to hide attackers’ origin

• A number of parallel, malicious developments
Botnet History

How did we get here?

• Early 1990s: IRC bots
  – automated management of IRC channels

• 1999 – 2000: Distributed DoS tools (distribution)
  – Trinoo, TFN2k, Stacheldraht

• 1998 – 2000: Trojan Horse (remote control)
  – BackOrifice, BackOrifice2k, SubSeven

• 2001 – 2005: Worms (spreading)
  – Code Red, Blaster, Sasser
Botnets

• Bots today
  – malware (backdoor, Trojan) running on compromised machines
  – incorporates different modules to carry out malicious tasks
    (spamming, DoS, …)
  – remote controlled by criminal entity (called bot master, bot herder)

• Bots are incorporated in network of compromised machines
  – Botnets (sizes up to hundreds of thousands of infected machines)

• Botnets
  – main vehicle for carrying out criminal activities
  – financial motivation
Botnets

• How do botnets get created?
  – infection and spreading

• How are bots (botnets) controlled?
  – command and control channel, robustness features

• What are botnets used for?
  – criminal applications

• How can we mitigate the problem?
  – defense mechanisms
Botnet Creation

• Hosts infected by one of
  – network worm (vulnerabilities)
  – email attachment
  – Trojan version of program (P2P is rife with this)
  – drive-by-downloads (malicious web sites)
  – existing backdoor (from previous infection)
Drive-By Downloads

• Drive-by downloads
  – attacks against web browser and/or vulnerable plugins
  – typically launched via client-side scripts (JavaScript, VBScript)

• Malicious scripts
  – injected into legitimate sites (e.g., via SQL injection)
  – hosted on malicious sites (URLs distributed via spam)
  – embedded into ads

• Redirection
  – landing page redirects to malicious site (e.g., via iframe)
  – makes management easier
  – customize exploits (browser version), serve each IP only once
Drive-By Downloads

- Malicious JavaScript code
  - typically obfuscated and hardened (make analysis more difficult)

function X88MxUL0B(U1TaW1TwV, IyxC82Rbo) {
    var c5kJu150o = 4294967296;
    var s3KRUV5X6 = arguments.callee;
    s3KRUV5X6 = s3KRUV5X6.toString();
    s3KRUV5X6 = s3KRUV5X6 + location.href;
    var s4wL1Rf57 = eval;
    ...
    // LR8yTdO7t holds the decoded code
    try {
        s4wL1Rf57(LR8yTdO7t);
    }
    ...
}
X88MxUL0B('ACada193b99c...76d9A7d6D676279665F5f81');
function Exhne69P() {
  var YuL42y0W = unescape("%u9090%u9090...
  ...%u3030%u3030%u3030%u3030%u3038%u0000");

  ...

  var pvOWGrVU = unescape("%0c0c%0c0c");
  pvOWGrVU = BAlrZJkW(pvOWGrVU, Hhvo4b_X);
  for (var cYQZIEiP=0; cYQZIEiP < cFyP_X9B; cYQZIEiP++) {
    RBGvC9bA[cYQZIEiP] = pvOWGrVU + YuL42y0W;
  }
}

function a9_bwCED() {
  try {
    var OBGUiGAa = new ActiveXObject('Sb.SuperBuddy');
    if (OBGUIGAa) {
      Exhne69P();
      dU578_go(9);
      OBGUiGAa.LinkSBIcons(0x0c0c0c0c0c);
    }
  } catch(e) { }
  return 0;
}
Drive-By Download

Latest Web Browsers Versions

% of IPs Per Day

Date

05/07 05/21 06/04 06/18 07/02 07/16 07/30 08/13 08/27

Firefox
IE
Safari
Opera
Chrome
Drive-By Download

![Vulnerable Browser Plugins Graph]

- **All Vulnerabilities**
- **Mebroot Vulnerable**

Flash 10.0.32 Released
Botnet Architectures

• Bot overlay network
  – centralized
    • IRC server (Internet relay chat)
    • web server (HTTP)
    • multiple controllers for robustness
  – peer-to-peer: self organizing
    • each host can be a worker or a proxy; decided dynamically
    • multi-level hierarchies possible

• Push versus pull designs
  – Attacker sends out message to tell bots what to do (push)
  – Worker bots “ask” for work to do (pull)
Centralized Botnet
Example – Agobot

• First discovered in 2002
  – also called Gaobot, Phatbot

• 20,000+ of C++, modular design + open source

• Modules
  – command and control: IRC based
  – protection: encrypted code, polymorphism, anti-disassembly code
  – growth: address scanning w/growing collection of software exploits
    (i.e., to be mounted against other machines under attacker control)
  – DDoS attacks: > 10 different varieties
  – harvesting: send back local PayPal info, …

• 100’s of variants
# Sample Agobot Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>harvest.cdkeys</td>
<td>Return a list of CD keys</td>
<td>pctrl.kill</td>
<td>Kill specified process set from service file</td>
</tr>
<tr>
<td>harvest.emails</td>
<td>Return a list of emails</td>
<td>pctrl.listsvc</td>
<td>Return list of all services that are running</td>
</tr>
<tr>
<td>harvest.emailshttp</td>
<td>Return a list of emails via HTTP</td>
<td>pctrl.killsvc</td>
<td>Delete/stop a specified service</td>
</tr>
<tr>
<td>harvest.aol</td>
<td>Return a list of AOL specific information</td>
<td>pctrl.killpid</td>
<td>Kill specified process</td>
</tr>
<tr>
<td>harvest.registry</td>
<td>Return registry information for specific registry path</td>
<td>inst.asadd</td>
<td>Add an autostart entry</td>
</tr>
<tr>
<td>harvest.windowskeys</td>
<td>Return Windows registry information</td>
<td>inst.asdel</td>
<td>Delete an autostart entry</td>
</tr>
<tr>
<td>pctrl.list</td>
<td>Return list of all processes</td>
<td>inst.svcadd</td>
<td>Adds a service to SCM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>inst.svcdel</td>
<td>Delete a service from SCM</td>
</tr>
</tbody>
</table>
Botnets

[Image: IRC conversation showing botnet commands and attacks]
Botnet Evolution

• Code shared back and forth
  – upgrade with new exploits, new attacks, add BNC, add spam proxy, etc.
  – rootkits and anti-anti-virus to hide from defenders
  – several released under GPL

• All bots today have auto upgrade capability
  – if version of bot < x, then download new version here
Botnet Evolution

• IRC server
  – often easy to take down certain hard-coded IP (dynamic DNS)
  – traffic easier to detect (switch to HTTP)

• HTTP
  – rotating domains (*rendez-vous* points)
    • computation based on current date
    • hard to take down many domains, must also do it quickly
    • reverse engineering domain generation algorithm important
  – Torpig
    • one new domain name per week, multiple TLDs
  – Conficker
    • list of 250 domains, 8 times per day
    • send queries to Google to obtain current time
Botnet Evolution

• Fast flux
  – network of bots with fast changing DNS records
  – many IP addresses for single DNS name (A records)
  – advanced type also change NS records (double flux)
  – used to hide mothership (content) behind proxy network
Botnet Evolution

Single-Flux

“bullet-proof” hosted DNS server

ns.example.com

4) Answer: 10.10.10.10

com

2) Referral: ns.example.com

1) Query: flux.example.com

client

Double-Flux

4) Query redirected & Response returned

53/UDP

“mothership”

ns.example.com

zombie home PC

com

2) Referral: ns.example.com

3) Query: 10.10.10.10

1) Query: flux.example.com

client

DNS Resolution Comparison
Botnet Evolution

dhcp-41-209:~ chris$ dig canadian-pharmacy.com

; <<>> DiG 9.3.5-P2 <<>> canadian-pharmacy.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 688
;; flags: qr rd ra; QUERY: 1, ANSWER: 7, AUTHORITY: 4, ADDITIONAL: 4

;; QUESTION SECTION:
;canadian-pharmacy.com.   IN   A

;; ANSWER SECTION:
canadian-pharmacy.com.   1789  IN   A   69.25.27.170
canadian-pharmacy.com.   1789  IN   A   69.25.27.173
canadian-pharmacy.com.   1789  IN   A   63.251.171.80
canadian-pharmacy.com.   1789  IN   A   63.251.171.81
canadian-pharmacy.com.   1789  IN   A   66.150.161.136
canadian-pharmacy.com.   1789  IN   A   66.150.161.140
canadian-pharmacy.com.   1789  IN   A   66.150.161.141
Example – Storm P2P Botnet

Bot master

HTTP proxies

Proxy bots

Overnet

Worker bots
Botnet Applications

- Entertainment
- Spam
- Proxying
  - for phishing or scam pages
- Denial of service
- Information theft
- Click fraud
Entertainment

• Take over people’s webcams (Bifrost)
Spam

- Use bots
  - to avoid blacklisting (such as Spamhaus DNSBL)
  - in addition to using open proxies
  - not as easy ...
Click Fraud

• Pay-per-click advertising
  – publishers display links from advertisers
  – advertising networks act as middlemen
  – sometimes the same as publishers (e.g., Google)

• Click fraud
  – botnets used to click on pay-per-click ads

• Motivation
  – competition between advertisers
  – revenue generation by bogus content provider
## Botnet Applications

<table>
<thead>
<tr>
<th>Capability</th>
<th>Ago</th>
<th>DSNX</th>
<th>evil</th>
<th>G-SyS</th>
<th>SD</th>
<th>Spy</th>
</tr>
</thead>
<tbody>
<tr>
<td>create port redirect</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>other proxy</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>download file from web</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DNS resolution</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>UDP/ping floods</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>other DDoS floods</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>scan/spread</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>spam</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>visit URL</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
Underground Economy

• Market access to bots
  – bot master collects and manages bots
  – access to proxies sold to spammers, often with commercial-looking web interface

• Rates and payment
  – non-exclusive access to botnet: 10¢ per machine
  – exclusive access: 25¢
  – payment via compromised account or cash out

• Identity theft
  – keystroke logging
  – complete identities available for $25 - $200+
    • Rates depend on financial situation of compromised person
    • Include all info from PC files, plus all websites of interest
Size of the Problem

• Many different opinions and figures
  – one problem is measurement based on unique IPs
  – safe to say that large botnets contain several hundred thousand infected machines
  – of course, many botnets exist at a given time (many smaller)
Mebroot / Torpig

- Take-over of the C&C
Mebroot / Torpig

Statistics (for ~10 days)

- Unique IP Count: 1,148,264
- Unique Torpig keys (machines): 180,835
- 63 GB of PCAP data
- POP accounts: 415,206
- Email addresses: 1,235,122
- Unique credit cards: 875
- Unique ATM pins: 141
- Unique social security numbers: 21
- Passwords: 411,039
Password Analysis

- 297,962 unique credentials

![Graph showing password cracking progress](image)
Botnet Analysis

• Obtain understanding of what a (potentially) malicious binary is doing

• I have already mentioned Anubis
  – other systems exist (CWSandbox, ThreatExpert, …)
Anubis

Anubis: ANalyzing Unknown Binaries

Welcome to Anubis
Anubis is a service for analyzing malware. Submit your Windows executable and receive an analysis report telling you what it does.
This service is still in a testing phase. Please understand that there might be errors and bugs.

Notification
Choose how you want to receive the analysis report.

- Web browser: The analysis result will be displayed in your browser as soon as it is ready.
- Email: We email you a link to the analysis result. For this type of notification, please enter your email-address here:

Choose the executable file
Choose the file that you want to analyze. The file must be a Windows executable. (details)
File (max. 8MB): no file selected

Get a priority boost
Enter the code that you see in the image on the left and your submission will be analyzed before all automatic submissions.

Submit for Analysis

Anubis Version: 1.63.0
## Malware Activity

<table>
<thead>
<tr>
<th>Observed Behavior</th>
<th>Percentage of Samples</th>
<th>Percentage of Clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installation of a Windows kernel driver:</td>
<td>3.34%</td>
<td>1.57%</td>
</tr>
<tr>
<td>Installation of a Windows service:</td>
<td>12.12%</td>
<td>7.96%</td>
</tr>
<tr>
<td>Modifying the hosts file:</td>
<td>1.97%</td>
<td>2.47%</td>
</tr>
<tr>
<td>Creating a file:</td>
<td>70.78%</td>
<td>69.90%</td>
</tr>
<tr>
<td>Deleting a file:</td>
<td>42.57%</td>
<td>43.43%</td>
</tr>
<tr>
<td>Modifying a file:</td>
<td>79.87%</td>
<td>75.62%</td>
</tr>
<tr>
<td>Installation of an IE BHO:</td>
<td>1.72%</td>
<td>1.75%</td>
</tr>
<tr>
<td>Installation of an IE Toolbar:</td>
<td>0.07%</td>
<td>0.18%</td>
</tr>
<tr>
<td>Display a GUI window:</td>
<td>33.26%</td>
<td>42.54%</td>
</tr>
<tr>
<td>Network Traffic:</td>
<td>55.18%</td>
<td>45.12%</td>
</tr>
<tr>
<td>Writing to stderr:</td>
<td>0.78%</td>
<td>0.37%</td>
</tr>
<tr>
<td>Writing to stdout:</td>
<td>1.09%</td>
<td>1.04%</td>
</tr>
<tr>
<td>Modifying a registry value:</td>
<td>74.59%</td>
<td>69.92%</td>
</tr>
<tr>
<td>Creating a registry key:</td>
<td>62.71%</td>
<td>52.25%</td>
</tr>
<tr>
<td>Creating a process:</td>
<td>52.19%</td>
<td>50.64%</td>
</tr>
</tbody>
</table>

Table 2: Overview of observed behavior.
Malware Activity

**Executables**
62% - Windows (or subfolder)
15% - Document and Settings

**Temporary files**
21% - Internet Explorer Temp

**Interesting registry keys**
36% [ Autostart related keys ]
SystemCertificates\TrustedPublisher\Certificates
Windows\CurrentVersion\Policies\System
(prevent TaskManager invocation)
MSWindows\Security settings
Evasion
Combating Evasion

- Malware can perform two kinds of checks
  - those based on system calls and environment values (user Andy)
  - those based on system (CPU) features and timing

- First check can be handled by multipath execution; second is more problematic

- Idea
  - execute malware on real host and record interactions
    - in particular, we need to recall system call return values
  - replay malware on Anubis, providing recorded system call results
  - assumption: program execution is deterministic
  - thus, when we see a deviation between the execution traces, the malware attempts to evade Anubis
Botnet Defense

• Signature-based (most AV products)

• Rule-based
  – monitor outbound network connections
    block certain ports (25, 6667, ...)

• Network content
  – Match network packet contents to known command strings (keywords)
    e.g., DoS command – .ddos.httpflood
  – suspicious IRC nicknames (Rishi)

• Network traffic monitoring
  – IP addresses (blacklists)
  – connection patterns
  – DNS queries

• Network monitoring (Rogue networks)
Botnet Defense

• Attack command and control infrastructure
  – take IRC channel offline
  – when dynamic DNS is used for central command server, route traffic to black hole
  – unregister malicious domains
  – Sybil attacks in P2P networks

• Honeypots
  – vulnerable computer that serves no purpose other than to attract attackers and study their behavior in controlled environments
  – when honeypot is compromised, bot logs into botnet
  – allows defender to study actions of botnet owners
Network Content – BotHunter

- Snort-based sensor suite for malware event detection
  - inbound scan detection
  - remote to local exploit detection
  - anomaly detection system for exploits over key TCP protocols
  - Botnet specific egg download banners,
  - Victim-to-C&C-based communications exchanges
    - particularly for IRC bot protocols

- Event correlation
  - combines information from sensors to recognize bots that infect and coordinate with your internal network assets
Network Traffic Patterns

• Unique characteristic: “Rallying”
  – bots spread like worms and Trojan horses
  – payloads may be common backdoors
  – (centralized) control of botnet is characteristic feature

• DNS-based monitoring
  – bots installed at network edge
  – IP addresses may vary, use Dynamic DNS (DDNS)
  – bots talk to controller, make DDNS lookup
    • pattern of DDNS lookup is easy to spot
Network Traffic Patterns

• Correlation of network traffic
  – detect similar connection patterns between hosts
  – similar command and control traffic (C-plane)
  – similar malicious activity (A-plane)
  – correlation between C-plane and A-plane for detection

• Properties
  – no a priori knowledge of C&C traffic required
  – require multiple infected machines in monitored network
Rogue Networks

• Networks persistently hosting malicious content for an extended period of time

• Legitimate networks will respond to abuse complaints
  – remove offending content

• Examples of rogue networks
  – Russian Business Network (RBN)
  – Atrivo/Intercage
  – McColo
  – Triple Fiber Network (3FN)
Rogue Networks

Comcast

Infected Machines (bots, websites)

AT&T

Rogue ASN

C&C / exploit servers
Objectives

• Systematically identify networks that are acting maliciously
• Notify legitimate networks to remediate malicious activity
• Assist legitimate ISPs de-peer (disconnect) from rogue networks
• Make it difficult for cybercriminals to find safe havens
Identifying Malicious Networks

• How to identify malicious content?
  – botnet C&C found by Anubis
  – exploit servers found by Wepawet

• When to consider a host malicious?
  – longevity!

• How to account for size?
  – larger networks will have more malicious content

• Computing a malscore for each autonomous system
Your Security Zen

SHA\text{ttered}

The first concrete collision attack against SHA-1

https://shattered.io

\begin{itemize}
\item CWI: Marc Stevens, Pierre Karpman
\item Google: Elie Bursztein, Ange Albertini, Yarik Markov
\end{itemize}

\begin{itemize}
\item sha1sum *\.pdf
38762cf7f55934b34d179ae6a4c80cadccbb7f0a 1.pdf
38762cf7f55934b34d179ae6a4c80cadccbb7f0a 2.pdf
\item sha256sum *\.pdf
2bb787a73e37352f92383abe7e2902936d1059ad9f1ba6daa9c1e58ee6970d0 1.pdf
d4488775d29bdef7993367d541064dbdda50d383f89f0aa13a6ff2e0894ba5ff 2.pdf
\end{itemize}
SHA-1 Collision

9,223,372,036,854,775,808 SHA1 computations

6,500 years of single-CPU computations
110 years of single-GPU computations