CS259D: Data Mining for CyberSecurity
Botnet

- Networks of machines compromised by malware
- Estimated 16-25% of computers on Internet part of a botnet
  - Botnet Rustock has over 1 million bots
  - Botnet Storm one of “world’s top super computers”
- Applications
  - Information and identity theft
  - Distributed denial of service (DDoS)
  - Software piracy
  - Spaming/Phishing
    - Almost 80% of all email traffic
    - Example: Grum, Cutwail, Rustock
  - Underground economy
    - 10,000 bots for $15
- Scale of damage (cf. International Telecommunication Union)
  - $13.2B direct damages to global economy in 2006
  - $67.2B in direct and indirect damages to US businesses in 2005
  - Global cost of spam in 2007: $100B global, $35B in US
Starting point

- **Internet Relay Chat (IRC)**
  - Text-based chat system
  - Organize communications in channels
  - Botnets to control interactions in IRC chat rooms
    - Interpret simple commands
    - Provide administration support
    - Offer simple games/services
    - Retrieve information: OS, logins, emails, etc.
  - First IRC bot: Eggdrop, 1993
Botnet components

- Zombies
  - High transmission rates
  - Low levels of security
  - Distant locations
  - Mostly MS Windows
Botnet lifecycle

- Initial infection
  - infected websites, email attachments, removable media, etc.

- Secondary injection
  - Host downloads & runs binaries, becomes a bot
  - FTP, HTTP, or P2P

- Connection or Rally
  - process of establishing connection with C&C
  - Happens every time the host is restarted

- Malicious activities
  - More intense message exchange between bot and C&C

- Maintenance and upgrading
Botnet C&C topologies

- Star
- Multi-server
- Hierarchical
- Random
Star C&C topology

- Centralized C&C communicate with all bots

Protocols used:

- IRC
  - C&C functionality of SDBot, GTBot, Agobot still in use
    - Source code published by author
- HTTP
  - Blend in with normal user traffic
  - Do-it-yourself kits
- Instant-Messaging (IM) protocols
  - ICQ, AIM, MSN Messenger
  - Needs creating one account per bot

Pro:
- Speed of Control

Con:
- Single point of failure
Multi-Server C&C topology

- Extension of Star topology
- C&C servers communicate among themselves

Pros:
- No single point of failure
- Geographical optimization

Cons:
- Requires more planning/effort from the operator
Hierarchical C&C topology

- One group of bots acting as servants
  - Static routable IP addresses
  - Proxy C&C instructions to client bots
- Variant: Hierarchical Kademlia
  - Set of clusters or islands of bots
  - P2P for intra-cluster communication
  - Inter-cluster communication: super bot peers
- Pros:
  - Botnet awareness: Interception of botnet won’t enumerate all members, unlikely to reveal C&C
  - Ease of resale
    - Lease/resale sections of botnet to other operators
- Cons:
  - Command latency: Not suitable for real-time activities
Random topology

- **No centralized C&C**
  - Commands injected by botmaster via any bot by sharing/publishing command files
  - Commands signed as authoritative to avoid takeover

- **Future: Skype-based botnets**
  - Better blend in with other P2P traffic

- **Pros:**
  - Highly resilient

- **Cons:**
  - Command latency
  - Botnet enumeration
Rallying mechanisms

- C&C location resolution
- Static Lists
  - Hard-coded list of IP addresses
  - Can be detected via a feed of botnet IPs
- Fluxing
  - Add resilience
  - Types
    - IP flux
    - Domain flux
IP flux

- Constant changing of IP address information

Single flux
- Multiple (100s-1000s) IP addresses associated with a domain name
- IP addresses registered and de-registered rapidly
  - Round-robin allocation
  - Short Time to Live (TTL) for DNS A records

Double flux
- Flux IP address of fully-qualified domain name
- Flux IP address of DNS server (NS records) used to look up IP address
Domain flux

- Domain wildcarding
- Domain generation algorithms
Domain wildcarding

- Use wildcarding in DNS records
  - Example: *.domain.com
- Useful for spamming/phishing; wildcard information used to
  - Identify victim (e.g., rjhgbrwh.domain.com)
  - Track success
Domain generation algorithms

- Create a dynamic list of FQDN’s every day
  - Cryptographic domain names
- Generated FQDN’s polled by bot to find C&C
- Example: the worm Conficker.C
  - Generates 50,000 domain names every day
  - Attempts to contact 500
  - 1% chance of update every day if operator registers only 1 domain per day
  - Preventing update requires registering 50,000 new domains every day
- Benefit
  - Domain names generated in volume, with short (typically 1-day) life span
  - Very difficult to investigate/block all possible domain names
Blind proxy redirection

- Add an extra layer of resiliency
- Proxy IP/domain lookup and C&C traffic
Botnet detection: challenges

- Botnet traffic similar to normal traffic
  - Likely encrypted as well
- Botnets evolve rapidly
  - New bots constantly getting added
  - Changing protocols
  - Changing architectures
  - Changing infection models
  - Fast flux hosting
Intrinsic properties of botnets:
- Bots communicate with C&C servers/peers
  - Centralized, Decentralized, etc.
- Bots do malicious activities
  - IRC-based botnets
    - 53% of botnet activity related to scan
      - For DDoS or spreading
    - 14.4% related to binary downloading
  - HTTP/P2P-based botnets
    - Mostly for sending spam
- Bots act a similar/correlated way
  - Otherwise, just a group of unrelated/isolated infections
  - Bots are non-human driven, programmed to perform C&C logic/communications
Botnet detection: BotMiner

- **Detection method:**
  - Cluster similar communication traffic
    - Who is talking to whom
    - C-plane (C&C communication traffic)
  - Cluster similar malicious traffic
    - Who is doing what
    - A-plane (Activity traffic)
  - Perform cross-cluster correlation
    - Find a coordinated group pattern

- **Assumes no a priori knowledge of**
  - Botnet’s protocol
  - Captured bot binaries (botnet signatures)
  - C&C server names/addresses
  - Content of the C&C communication
BotMiner Architecture

Network Traffic

A-Plane Monitor
- Scan
- Spam
- Binary Downloading
- Exploit
- ...

Activity Log

A-Plane Clustering

C-Plane Monitor
- Flow Log

C-Plane Clustering

Cross-Plane Correlation

Reports
BotMiner: Traffic monitor

- **C-plane**: captures network flows
  - Who is talking to whom
  - Each record contains the following info:
    Time, duration, source IP & port, destination IP & port, number of packets & bytes transferred in each direction

- **A-plane**:
  - Who is doing what
  - Analyzes outbound traffic
  - Detects several types of malicious activities
    - Scanning:
    - Spamming:
    - Binary downloads
    - Exploit attempts
  - Based on Snort, with some modifications
BotMiner: C-plane clustering
BotMiner: C-plane clustering

- Find machines with similar communication patterns
- Steps:
  - First two steps not critical but help efficiency
    - Filter out irrelevant traffic flows, filter out flows that are not completely established, filter out flows with well-known destinations
  - Third step: Given an epoch, aggregate into communication flows (C-flows)
    - C-flow = \{F_i\} where F_i have same protocol (TCP/UDP), source IP, destination IP & port
C-flow feature extraction

- For each C-flow:
  - Temporal
    - Number of flows per hour (fph)
    - Number of bytes per second (bps)
  - Spatial
    - Number of packets per flow (ppf)
    - Number of bytes per packets (bpp)
C-plane clustering

- Performed in two steps using a variant of k-means
  - Coarse grained clustering on entire dataset
  - Fine-grained clustering on multiple smaller clusters using all features
- Reduced feature set:
  - Avg, Std-dev of each feature
- Full feature set:
  - 13 bins per feature to approximate their distribution
A-plane Clustering

- Scan activity features
  - Scanning ports
  - Target subnet
- Spam activity features
  - SMTP connection destinations
- Binary download
  - First/random portion/packet of the binary
# BotMiner: Results

<table>
<thead>
<tr>
<th>Botnet</th>
<th>Number of Bots</th>
<th>Detected?</th>
<th>Clustered Bots</th>
<th>Detection Rate</th>
<th>False Positive Clusters/Hosts</th>
<th>FP Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRC-rbot</td>
<td>4</td>
<td>YES</td>
<td>4</td>
<td>100%</td>
<td>1/2</td>
<td>0.003</td>
</tr>
<tr>
<td>IRC-sdbot</td>
<td>4</td>
<td>YES</td>
<td>4</td>
<td>100%</td>
<td>1/2</td>
<td>0.003</td>
</tr>
<tr>
<td>IRC-spybot</td>
<td>4</td>
<td>YES</td>
<td>3</td>
<td>75%</td>
<td>1/2</td>
<td>0.003</td>
</tr>
<tr>
<td>IRC-N</td>
<td>259</td>
<td>YES</td>
<td>258</td>
<td>99.6%</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>HTTP-1</td>
<td>4</td>
<td>YES</td>
<td>4</td>
<td>100%</td>
<td>1/2</td>
<td>0.003</td>
</tr>
<tr>
<td>HTTP-2</td>
<td>4</td>
<td>YES</td>
<td>4</td>
<td>100%</td>
<td>1/2</td>
<td>0.003</td>
</tr>
<tr>
<td>P2P-Storm</td>
<td>13</td>
<td>YES</td>
<td>13</td>
<td>100%</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>P2P-Nugache</td>
<td>82</td>
<td>YES</td>
<td>82</td>
<td>100%</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
BotMiner: Limitations

- Evading C-plane monitoring/clustering
  - Randomize individual communication patterns
    - Example: randomize number of packets per flow, number of bytes per packet

- Evading A-plane monitoring/clustering
  - Stealthy malicious activities
    - Scan slowly

- Evading cross-plane analysis
  - Delay the malicious activities (give commands a few days in advance)

- Offline system
  - Prolonged data collection
Botnet detection: BotFinder (2012)

- **Goals**
  - Detect individual bot infections
  - Only rely on network flow
    - Resilient to encryption or obfuscation
    - No need for deep packet inspection
  - Detect stealthy bots stealing data but not spamming

- **Observation**
  - C&C connections follow regular patterns
  - Run bot binaries in a controlled environment, learn patterns
Botnet detection: BotFinder (2012)

- Observation
  - C&C connections follow regular patterns
  - Bots send similar traffic to C&C
  - Upload information to C&C in similar way
  - Timing patterns of communications with C&C
BotFinder: System

1a) Training Malware Traffic

1b) Traffic for Investigation

2) Flow Reassembly

3) Trace Extraction

4) Statistical Feature Analysis

5a) Model Creation

5b) Malware Detection
BotFinder: Features

- Average time between the start times of two subsequent flows in the trace
- Average duration of a connection
- Number of bytes transferred on average to the source
- Number of bytes transferred on average to the destination
- Fourier Transform over the flow start times
BotFinder: Model Creation and Matching

- Cluster each feature separately
  - Malware features uncorrelated

- Matching: Match each feature of the trace against the corresponding model’s cluster
## Results

<table>
<thead>
<tr>
<th>Malware Family</th>
<th>BotFinder Detection</th>
<th>BotFinder False Positives</th>
<th>BotHunter Detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Banbra</td>
<td>100%</td>
<td>0</td>
<td>24%</td>
</tr>
<tr>
<td>Bifrose</td>
<td>49%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>Blackenergy</td>
<td>85%</td>
<td>2</td>
<td>21%</td>
</tr>
<tr>
<td>Dedler</td>
<td>63%</td>
<td>0</td>
<td>n/a</td>
</tr>
<tr>
<td>Pushdo</td>
<td>81%</td>
<td>0</td>
<td>11%</td>
</tr>
<tr>
<td>Sasfis</td>
<td>87%</td>
<td>1</td>
<td>0%</td>
</tr>
</tbody>
</table>
References

- Botnet Communication Topologies

- BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection