Stony Brook University

PragSec Lab

# Bots, Crawlers, and Spiders: Understanding How Automated Web Clients Find and Attack their Victims

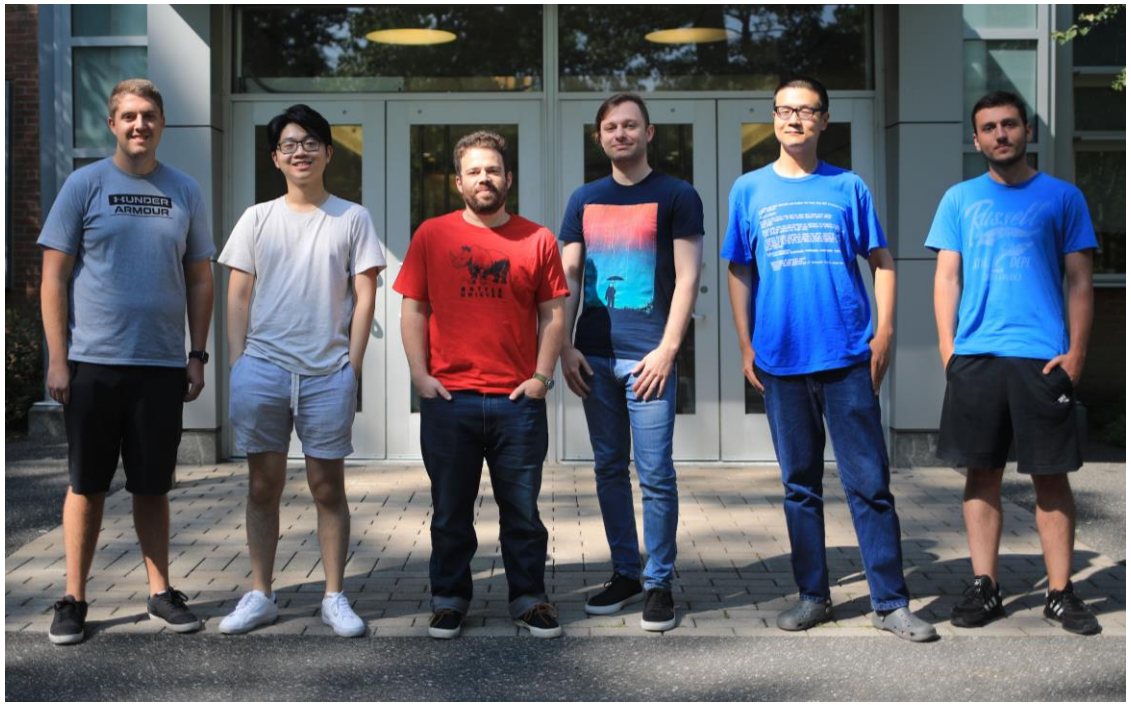
Nick Nikiforakis

**HoTSoS 2024**

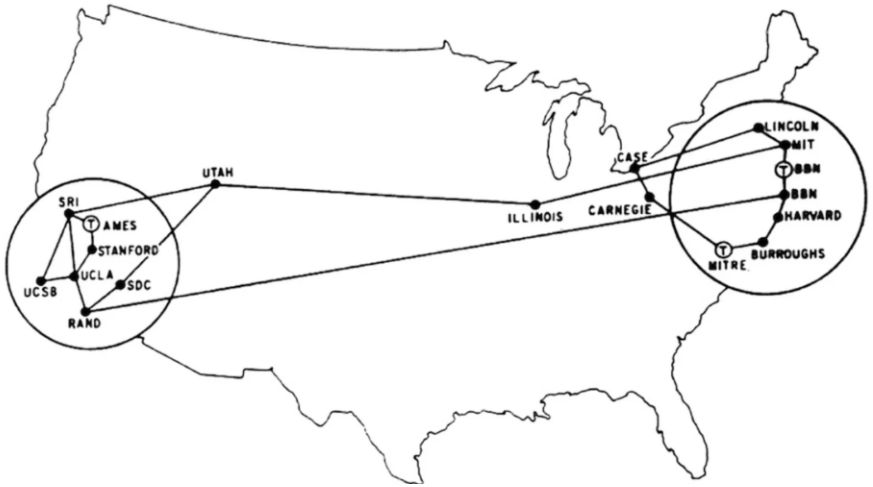
# Who am I?

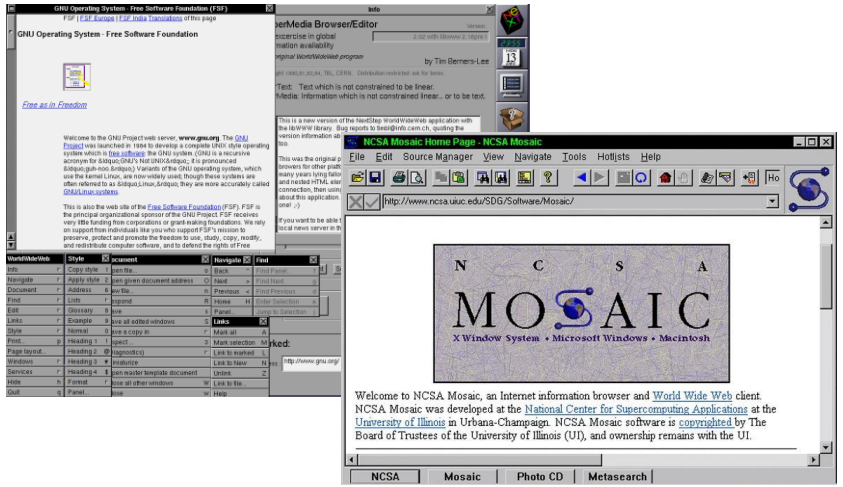
PragSec Lab

- Associate Professor at Stony Brook University
- Areas of research
  - Online tracking
  - DNS security
  - Web application fingerprinting
  - Mobile browser security
  - Attack surface reduction
  - Honeypots and deception
  - Anti-bot technologies

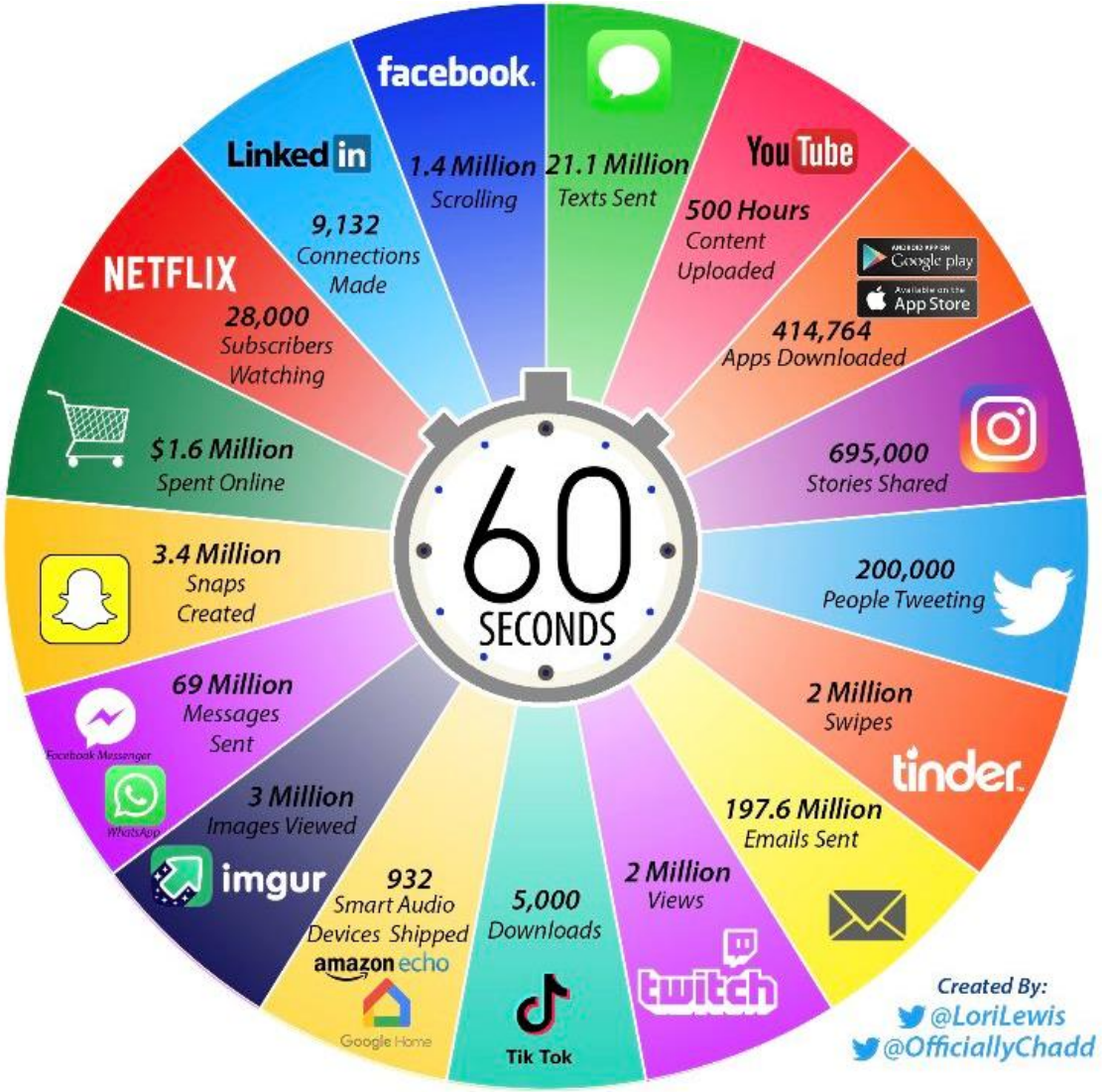


# How it started



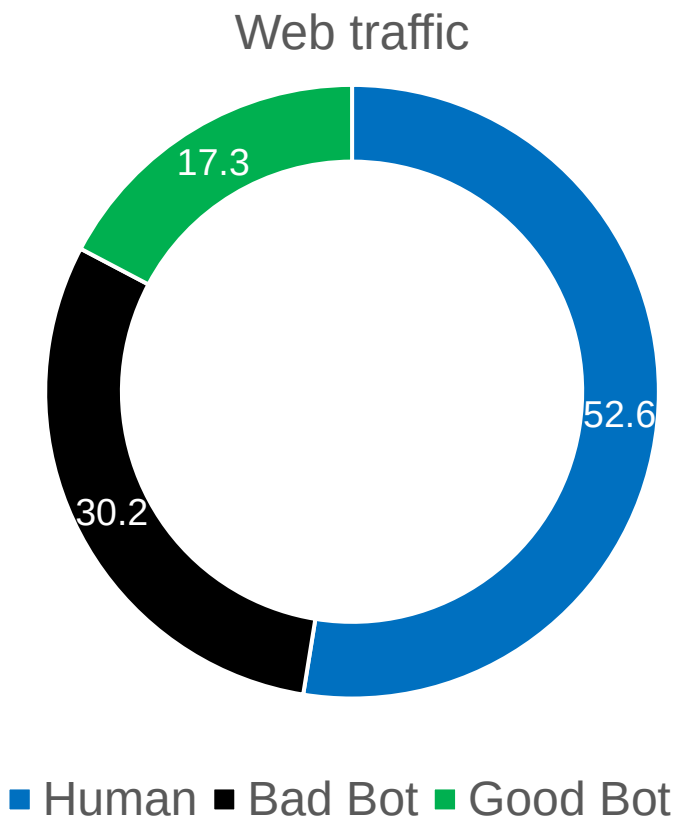




# How it's going



# Web bots

- Web bots are programs that interact with websites in automated ways
  - Benign bots
    - Page indexing, link previews, malware detection
  - Malicious bots
    - Scraping, brute-forcing credentials, stealing backup/configuration files, exploiting vulnerabilities

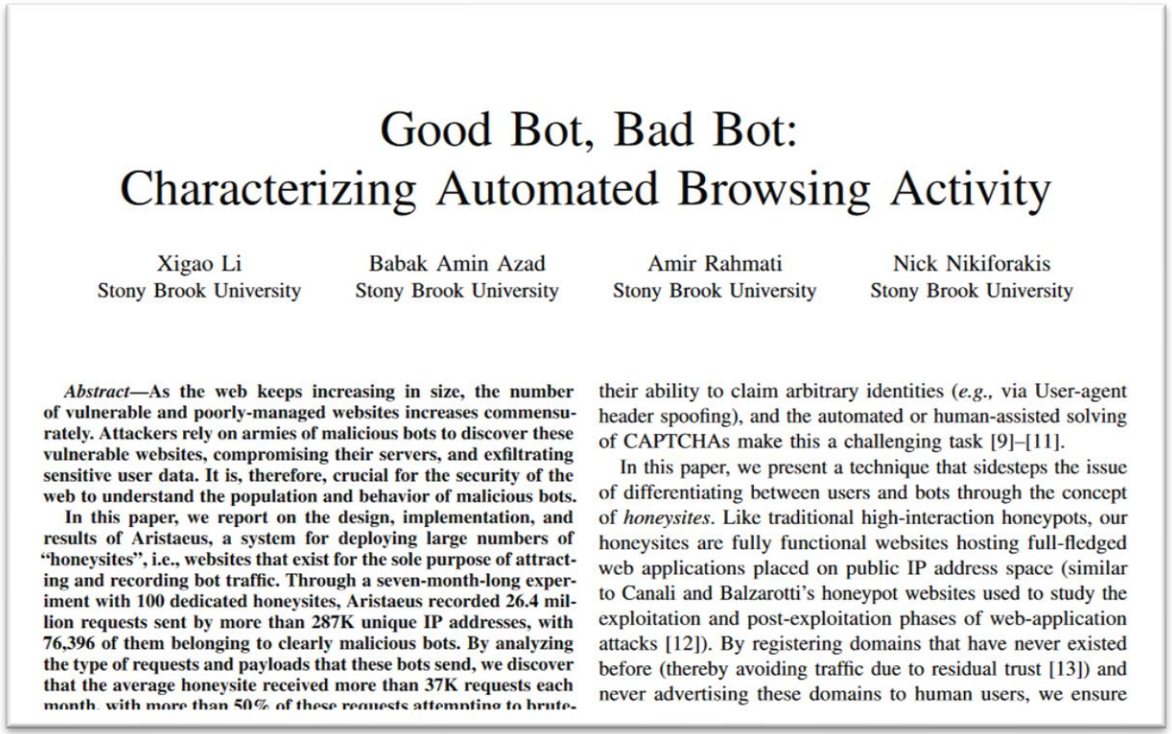



*Source: Imperva Bot Report, 2023*

# Bots and you

- Bots still require mechanisms to procure lists of targets
  - IP-address-based host scanning
  - Crawling popular websites and following links
  - Processing website lists from different application domains
    - Previously-compromised websites
    - Zone files from different authoritative name servers
  - Certificate Transparency?
- What does the average malicious bot do once it discovers a new target website?

# Basis of today's talk



Good Bot, Bad Bot:
Characterizing Automated Browsing Activity

Xigao Li
Stony Brook University

Babak Amin Azad
Stony Brook University

Amir Rahmati
Stony Brook University

Nick Nikiforakis
Stony Brook University

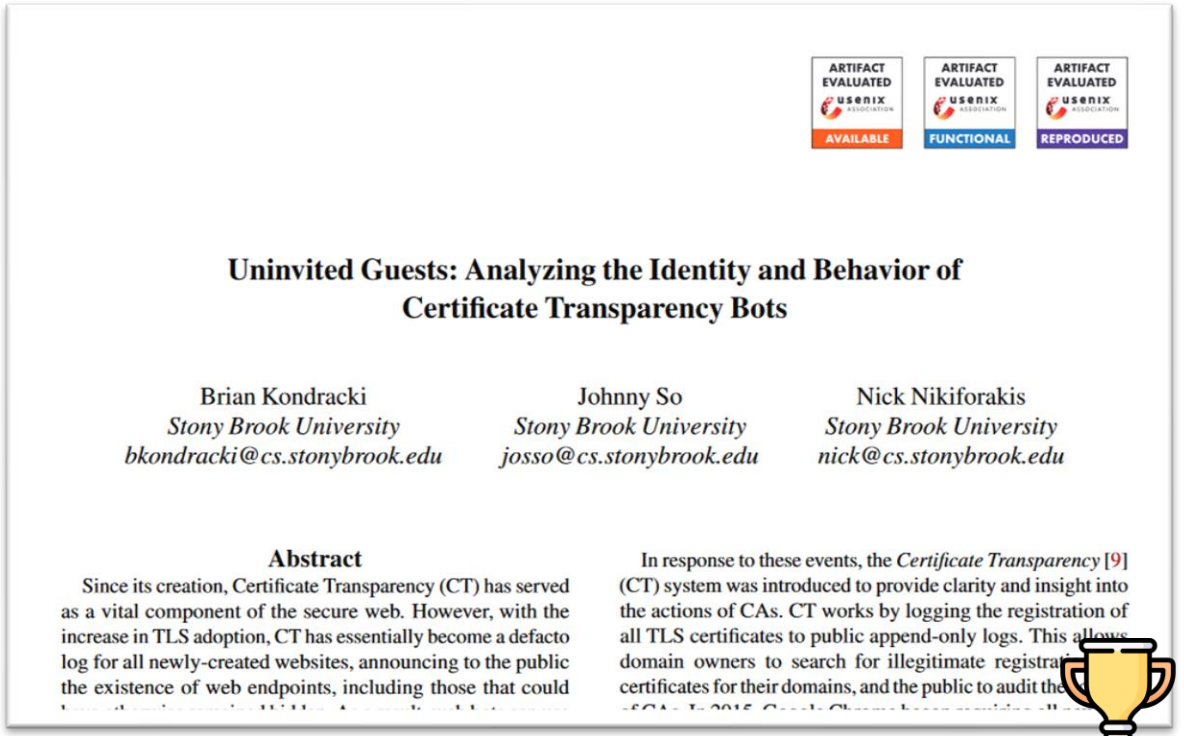
*Abstract*—As the web keeps increasing in size, the number of vulnerable and poorly-managed websites increases commensurately. Attackers rely on armies of malicious bots to discover these vulnerable websites, compromising their servers, and exfiltrating sensitive user data. It is, therefore, crucial for the security of the web to understand the population and behavior of malicious bots.

In this paper, we report on the design, implementation, and results of Aristaeus, a system for deploying large numbers of "honeysites", i.e., websites that exist for the sole purpose of attracting and recording bot traffic. Through a seven-month-long experiment with 100 dedicated honeysites, Aristaeus recorded 26.4 million requests sent by more than 287K unique IP addresses, with 76,396 of them belonging to clearly malicious bots. By analyzing the type of requests and payloads that these bots send, we discover that the average honeysite received more than 37K requests each month, with more than 50% of these requests attempting to brute-

their ability to claim arbitrary identities (*e.g.,* via User-agent header spoofing), and the automated or human-assisted solving of CAPTCHAs make this a challenging task [9]–[11].

In this paper, we present a technique that sidesteps the issue of differentiating between users and bots through the concept of *honeysites*. Like traditional high-interaction honeypots, our honeysites are fully functional websites hosting full-fledged web applications placed on public IP address space (similar to Canali and Balzarotti's honeypot websites used to study the exploitation and post-exploitation phases of web-application attacks [12]). By registering domains that have never existed before (thereby avoiding traffic due to residual trust [13]) and never advertising these domains to human users, we ensure

*IEEE S&P, 2021*






**Uninvited Guests: Analyzing the Identity and Behavior of Certificate Transparency Bots**

Brian Kondracki
*Stony Brook University*
*bkondracki@cs.stonybrook.edu*

Johnny So
*Stony Brook University*
*josso@cs.stonybrook.edu*

Nick Nikiforakis
*Stony Brook University*
*nick@cs.stonybrook.edu*

**Abstract**

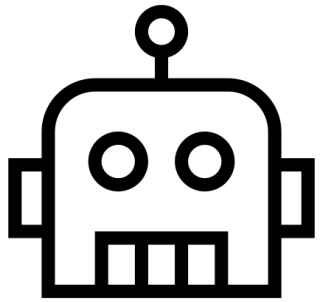
Since its creation, Certificate Transparency (CT) has served as a vital component of the secure web. However, with the increase in TLS adoption, CT has essentially become a defacto log for all newly-created websites, announcing to the public the existence of web endpoints, including those that could

In response to these events, the *Certificate Transparency* [9] (CT) system was introduced to provide clarity and insight into the actions of CAs. CT works by logging the registration of all TLS certificates to public append-only logs. This allows domain owners to search for illegitimate registrations certificates for their domains, and the public to audit the



*USENIX Security, 2022*

# Detecting benign web bots

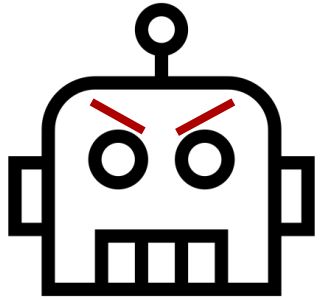


- Benign bots announce themselves

- Google
  - **IP address**: 66.249.66.1
  - **User Agent**: Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)

- Bing
  - **IP address**: 40.77.167.41
  - **User Agent**: Mozilla/5.0 (compatible; bingbot/2.0; +http://www.bing.com/bingbot.htm)

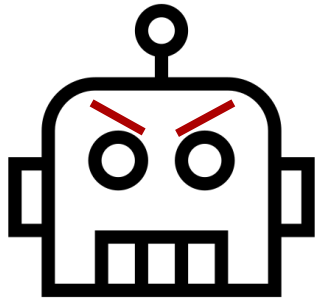
# Detecting malicious web bots



- This is more challenging
- Malicious bot strategy #1
  - Pretend to be a known benign bot (Googlebot/Bingbot/etc.)
  - Scrape/attack with administrators fearing the blocking of a known benign crawler
    - No one wants to block Googlebot
- Defenses
  - Reverse-DNS the IP address claiming to be a bot

| User Agent | IP address | Reverse DNS |
|---|---|---|
| Mozilla/5.0 (compatible; Googlebot/2.1… | 66.249.66.1 | crawl-66-249-66-1.googlebot.com |
| Mozilla/5.0 (compatible; Googlebot/2.1… | 67.245.115.115 | cpe-67-245-115-115.nyc.res.rr.com |

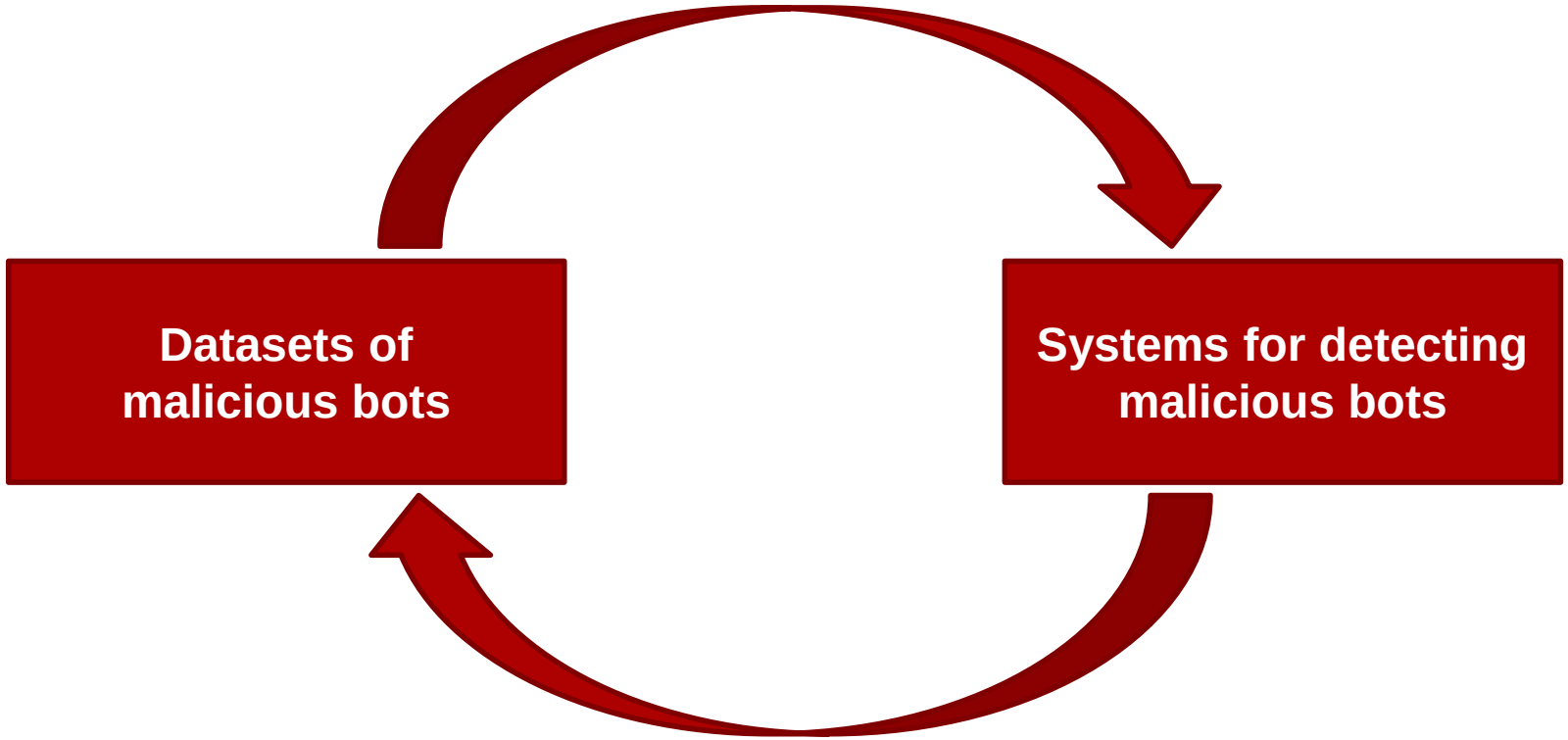
# Detecting malicious web bots



- Malicious bot strategy #2
  - Pretend to be a regular user
- Steps that malicious bots can take
  - Spoof User Agents
  - Simulate user actions
  - Low-and-slow
  - Use proxy servers
- Defenses (open ended)
  - Anomaly detection
    - Timing of requests
    - Types of requests
  - IP address blocklists
  - CAPTCHAs when suspicious
  - ???

| | | | | |
|---|---|---|---|---|
| Page Views: | | 1 | Total Visits: | 1 |
| Exit Time: | | Apr 28 2021 21:03:07 | Location: | Los Angeles, California, United States |
| Resolution: | | 800x600 | IP Address: | Multacom Corporation (173.82.104.167) |
| System: | | Chrome 87.0 Win10 | Referring URL: | www.isvoc.com/ |
| | | | Visit Page: | https://securitee.org/ |
| Page Views: | | 1 | Total Visits: | 1 |
| Exit Time: | | Apr 28 2021 21:03:07 | Location: | Los Angeles, California, United States |
| Resolution: | | 800x600 | IP Address: | Multacom Corporation (173.82.104.167) |
| System: | | Chrome 86.0 Win10 | Referring URL: | www.isvoc.com/ |
| | | | Visit Page: | https://securitee.org/ |
| Page Views: | | 1 | Total Visits: | 1 |
| Exit Time: | | Apr 28 2021 21:03:07 | Location: | Los Angeles, California, United States |
| Resolution: | | 800x600 | IP Address: | Multacom Corporation (173.82.104.167) |
| System: | | Chrome 87.0 Win10 | Referring URL: | www.securemymind.com/ |
| | | | Visit Page: | https://securitee.org/ |

# Robotic yet circular dependencies



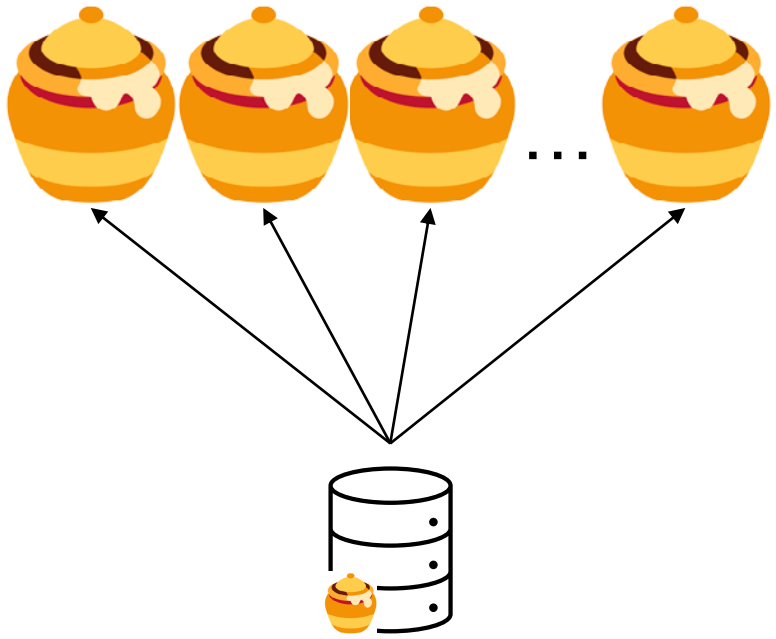

*Prior Academic Solutions: Manual filtering of web-server logs*

# Research questions

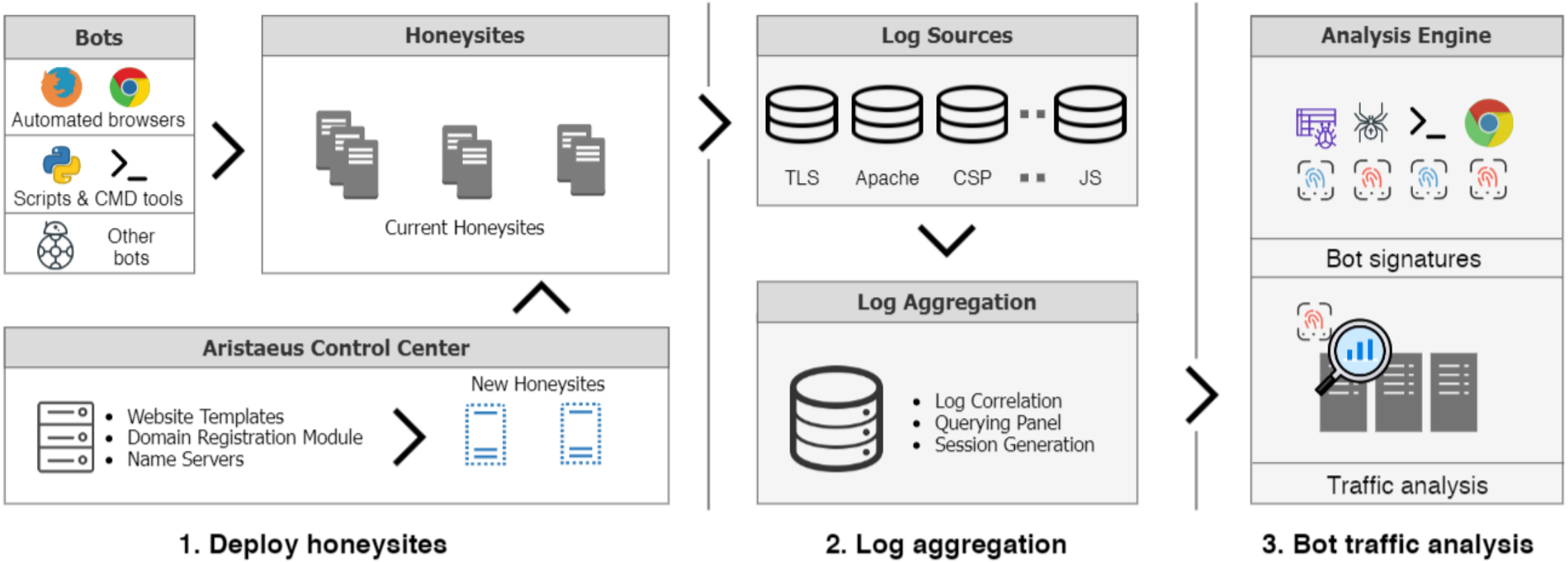
- Can we curate a bot-only dataset in a way that doesn't depend on our manual-analysis prowess?
  - Benign vs. malicious bots
  - Activities of malicious bots
  - Claimed vs. actual identity of malicious bots
  - Trends of bot-activity over time

# Network of honeysites

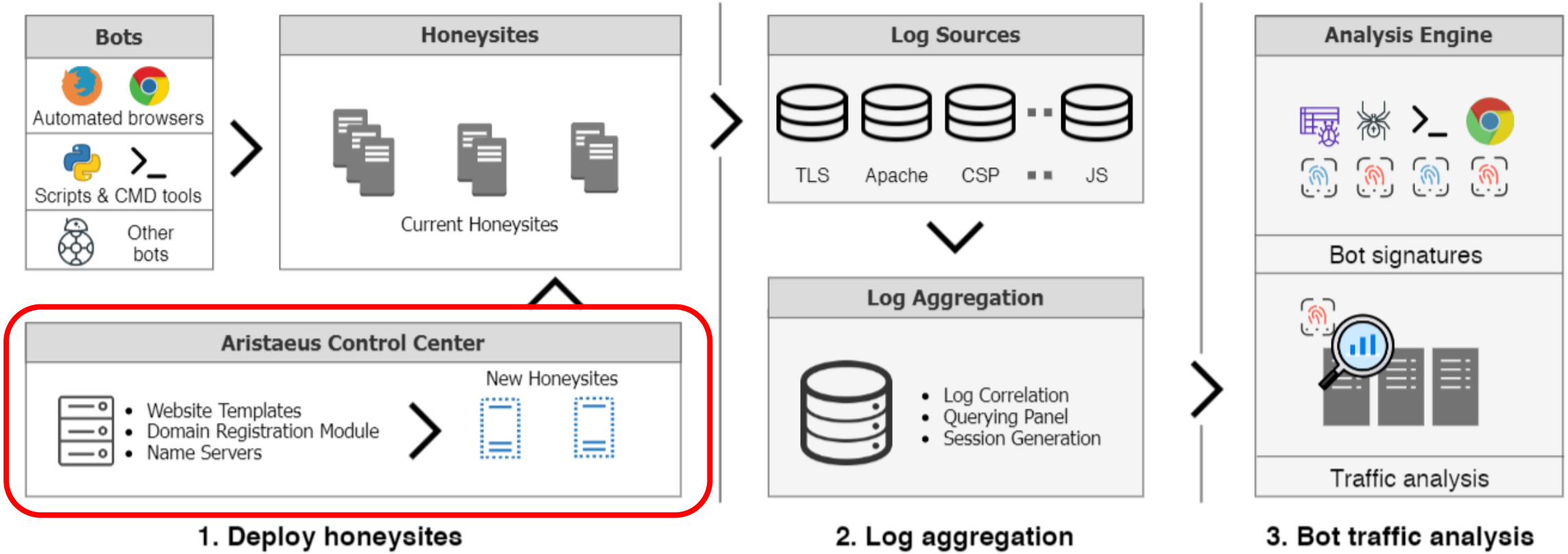
- Aristaeus
  - A system that provides flexible remote deployment and management of honeysites
  - Honeysites:
    - Fully-functional web applications, augmented with state-of-the-art fingerprinting techniques
  - A centralized log server pulls logs from each honeysite on a daily basis
    - Injected in a distributed database (Elastic Search)



# Overview of Aristaeus

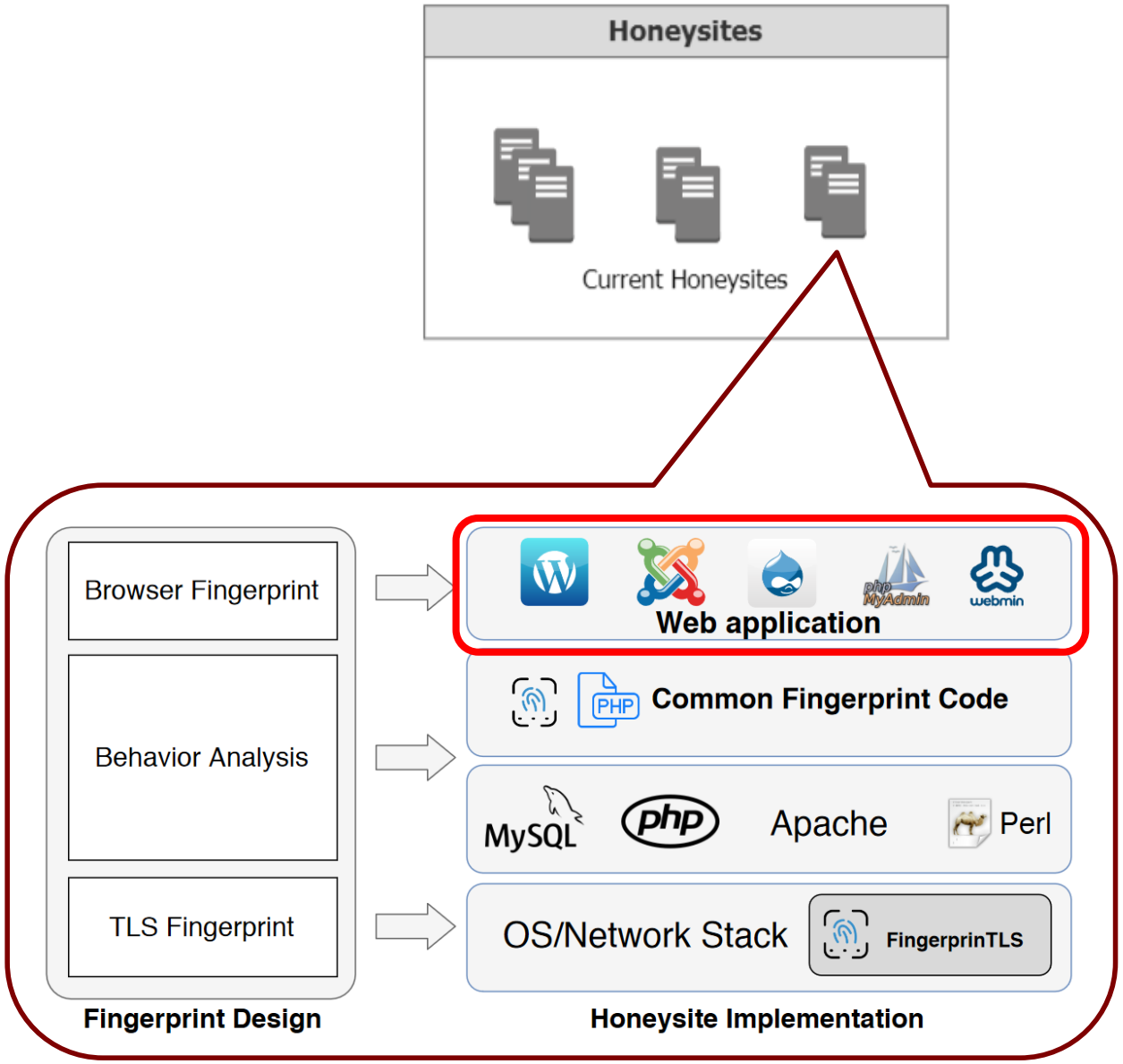


# Overview of Aristaeus



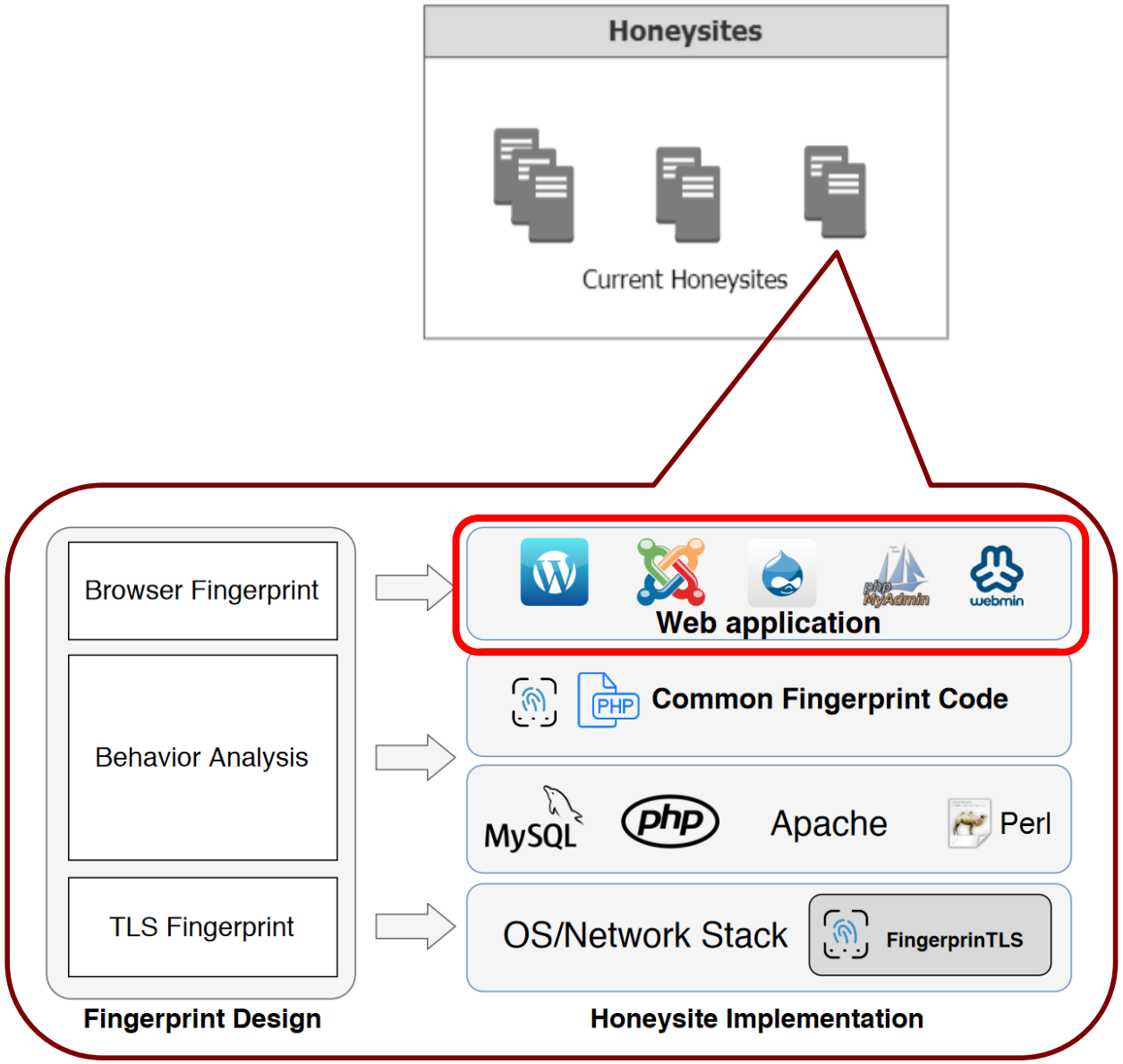
# What's the best bait?

- Deployed web applications
  - WordPress, Joomla, Drupal, PHPMyAdmin, and Webmin
    - Tens of years of development
    - Hundreds of vulnerabilities
    - Millions of installations
  - Content Management Systems and System Administration tools
    - Promise of data **and** Remote Code Execution



# Client fingerprinting

- Javascript API support
  - Basic support test
    - document.write(), var img …
  - Ajax support
- Browser fingerprinting
  - What information can we gather from common JS APIs?
- Support for security policies
  - CSP, X-Frame-Options, Mixed Content (HTTP/HTTPS) ,etc.



# One slide primer on TLS handshakes

- In TLS ClientHello, Clients inform Servers of their TLS capabilities
  - TLS versions
  - Ciphersuites

# Everyone's different

- Different TLS Clients implement things slightly differently
  - Chrome/Chromium support GREASE, a mechanism for catching interoperability issues between clients and servers
  - Firefox and Safari do not support GREASE
  - Command-line tools built using Python, curl, Perl, will have different TLS libraries than both Chrome and Firefox

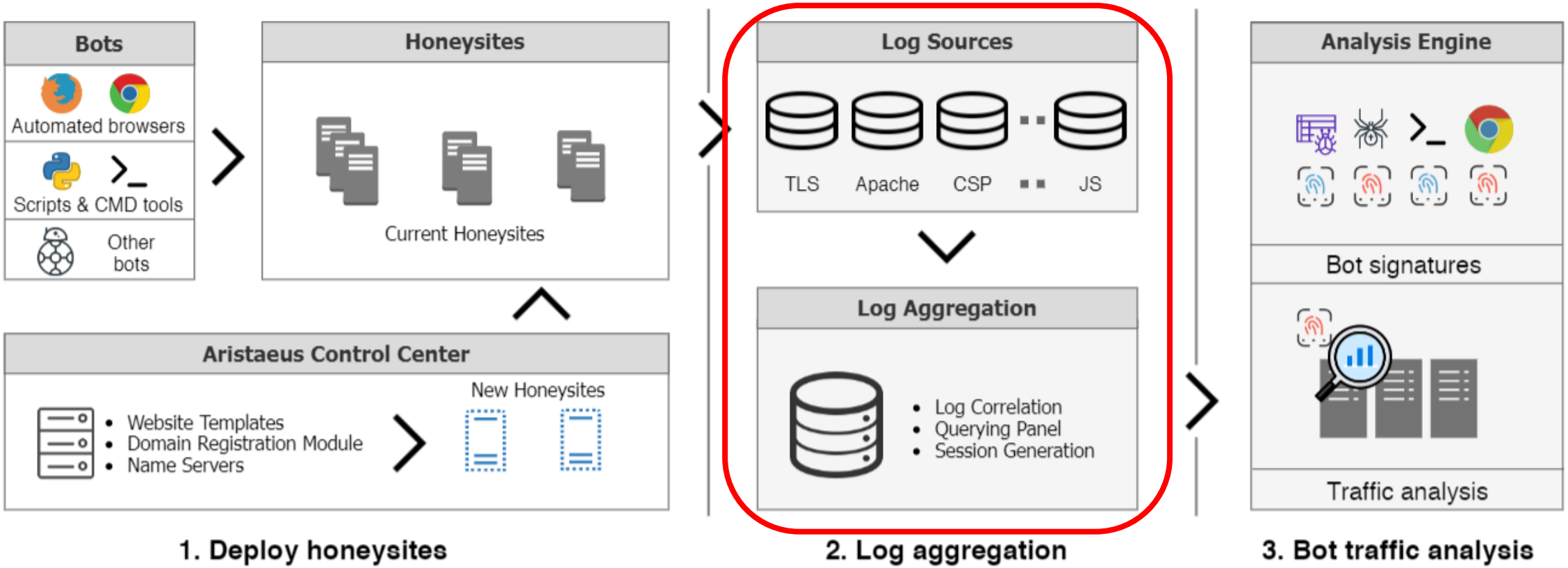
```
import "net/http"

resp, err := http.Get("https://example.com/")
```

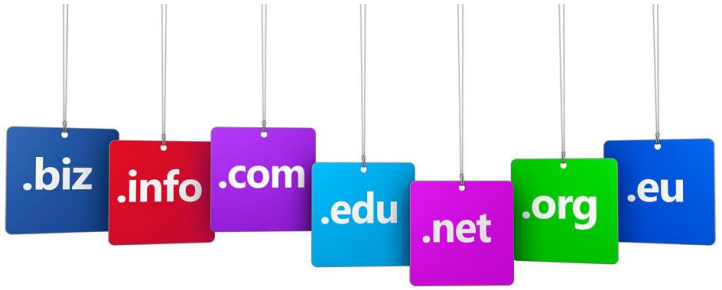

```
"tlsfp": {
    "ciphersuite": "0xC02F 0xC030 0xC02B 0xC02C 0xCCA8 0xCCA9 0xC013
                    0xC009 0xC014 0xC00A 0x009C 0x009D 0x002F 0x0035 0xC012 0x000A",
    "tls_version": "0x0303",
    "sig_alg": "0x0401 0x0403 0x0501 0x0503 0x0601 0x0603 0x0201 0x0203 ",
    "src_port": 22260,
    "record_tls_version": "0x0301",
    "timestamp": "2020-04-25 03:55:59",
    "server_name": "www.historytenantfile.com",
    "ipv4_src": "167.71.193.105",
    "e_curves": "0x001D 0x0017 0x0018 0x0019 ",
    "extensions": "0x0000 0x0005 0x000A 0x000B 0x000D 0xFF01 0x0012 ",
    "ciphersuite_length": "0x0020",
}
```


Go-http-client

# Overview of Aristaeus



# Deployment of Aristaeus

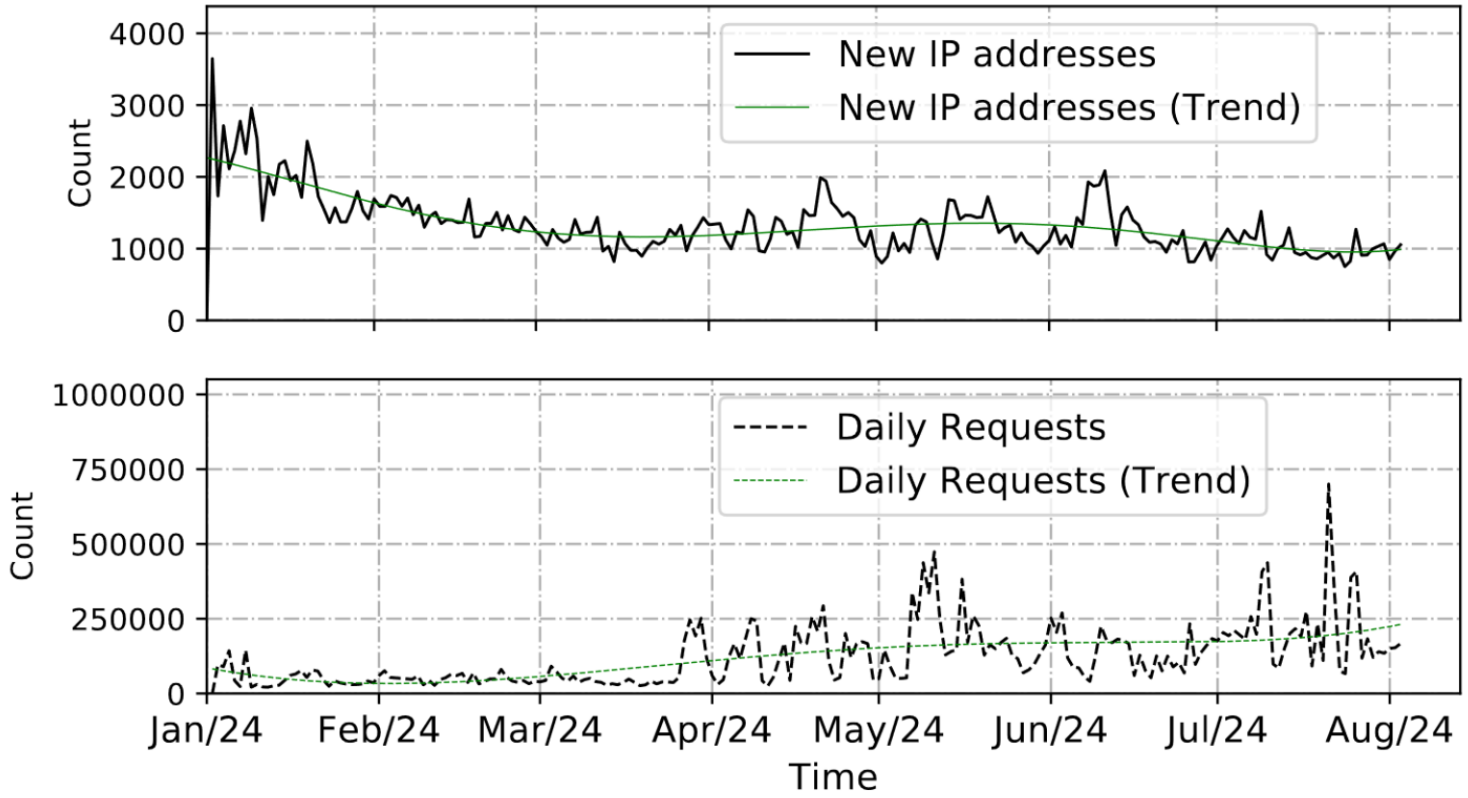



- Register 100 domains
  - One condition: Domains should have never been registered before
  - Avoid residual-trust traffic from old sites and buggy systems
  - No public advertisement of these domains
- Spawn one honeysite for each domain
  - 100 VMs in AWS
    - North America, Europe, and Asia
  - Let's Encrypt automatically used to get valid TLS certificates
- 7-month long experiment recording everything and anything

# By the numbers

# Daily traffic

- We keep observing new sources, for the entire 7 months
- Average of 1,235 requests per day



# Site discovery

- Since we never advertised our domains, how do bots find us?

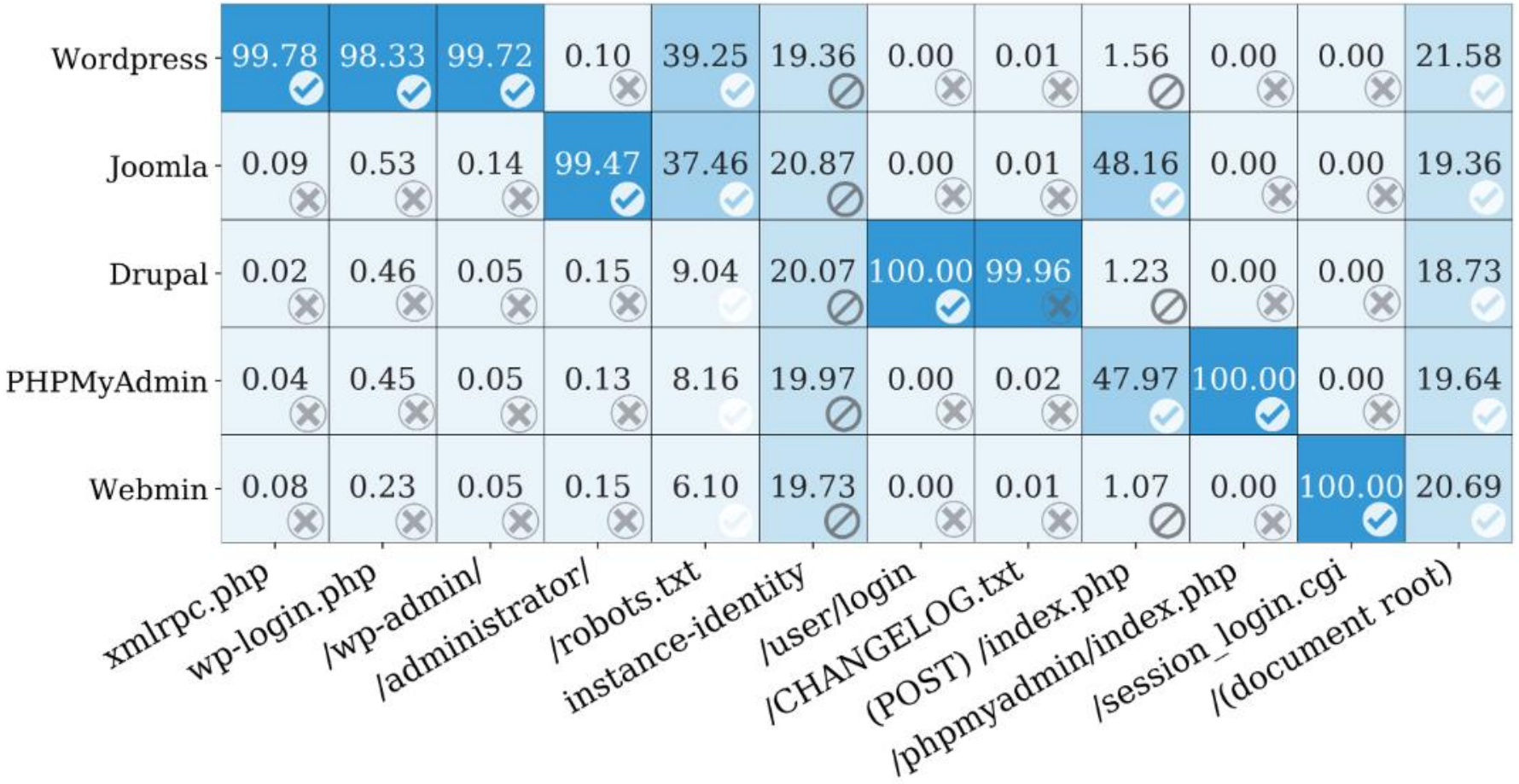
- Inspect the Host header of client-side HTTP headers:
  - 44% of bots visit through the IP address
  - 30% present no Host header
  - 26% explicitly ask for our domains
    - Certificate transparency
    - Zone files
    - Prior crawls


```
"hasforensic": true,
"flog": {
  "headersText": [
    "Host:52.3.222.202",
    "User-Agent:Mozilla/5.0 (Windows NT 10.
    "Accept:*/*\n"
  ],
  "headersKV": {
    "Nonce": "ap",
    "Host": "52.3.222.202",
    "Accept": "*/*\n",
    "User-Agent": "Mozilla/5.0 (Windows NT
  },
  "request": "GET / HTTP/1.1",
  "fid": "XqOyiz0QYdqDTO9GefocHgAAAAI"
```

```
  ],
  "headersKV": {
    "Nonce": "ap",
    "Accept-Encoding": "gzip",
    "Connection": "close\n",
    "User-Agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64;
    "Host": "www.objectivecurtainbook.com",
    "Cookie": "csessid=5ea3edf88e02d; wordpress_test_cookie
    "Referer": "https%3a//www.objectivecurtainbook.com/wp-l
    "Content-Type": "application/x-www-form-urlencoded"
  },
  "request": "GET /wp-admin/ HTTP/1.1",
  "fid": "XqPt-yv0Qhj0lfbRhAU6sQAAAAU"
```

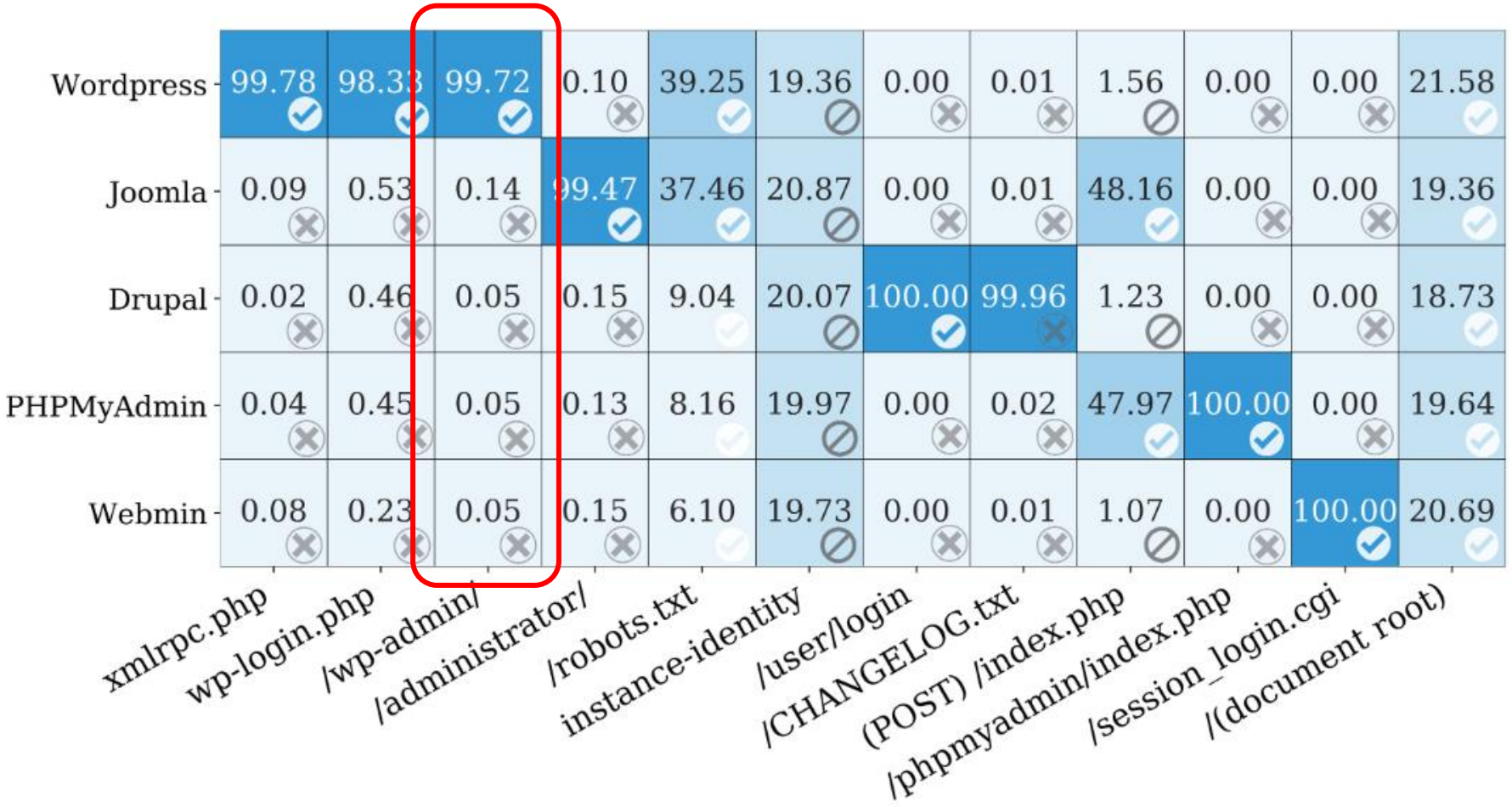
# Popular endpoints

**✓=exists, ✗=does not exist, 🚫=not accessible**



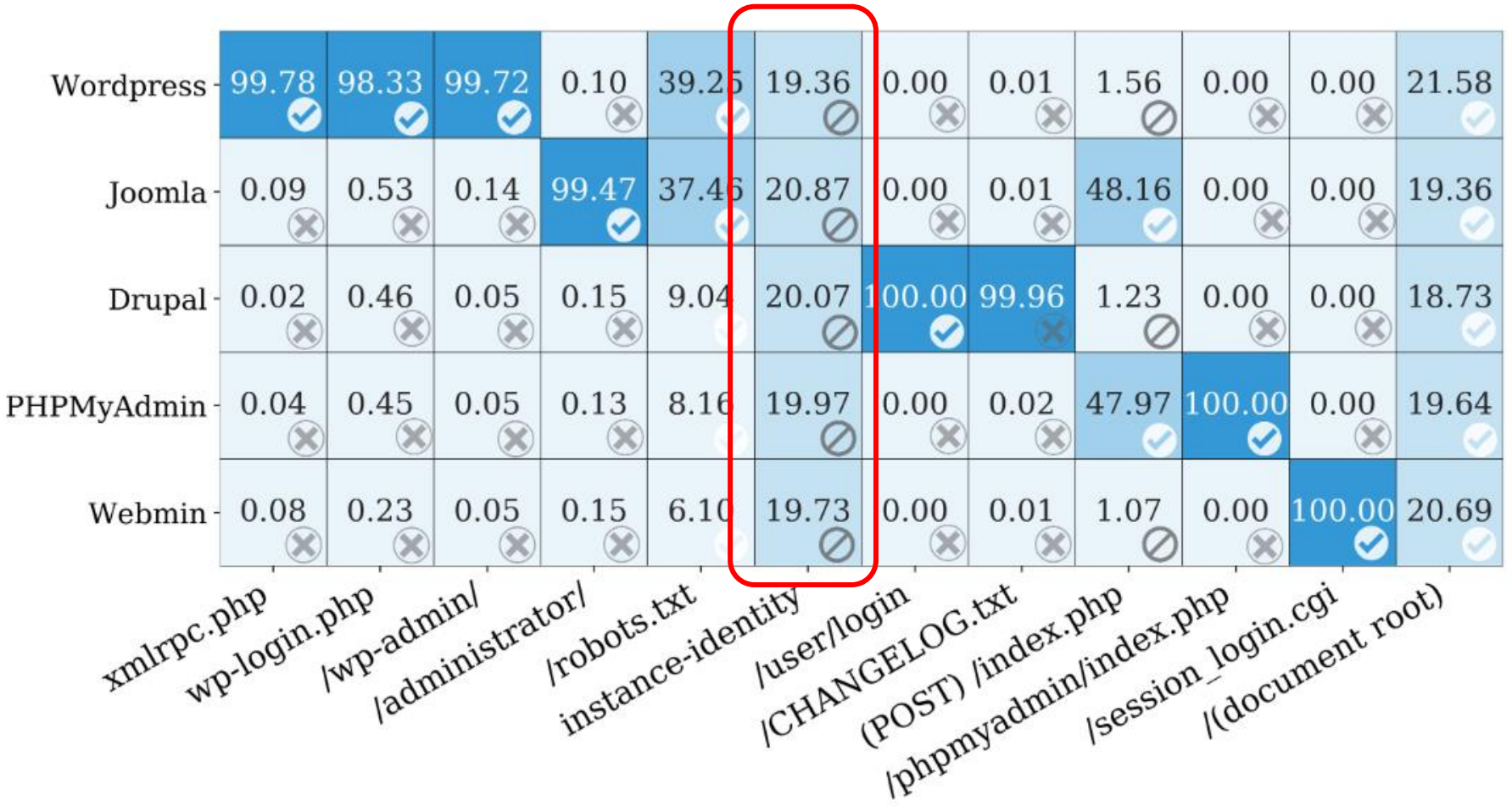
# Popular endpoints

✓=exists, ✗=does not exist, 🚫=not accessible



# Popular endpoints

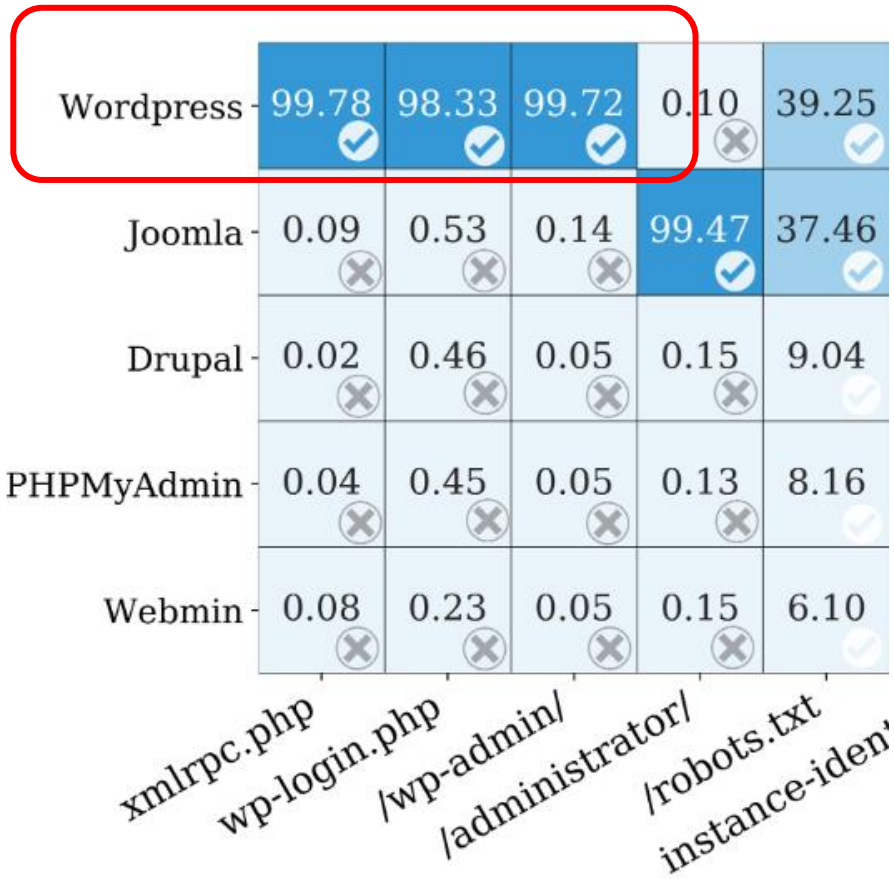
✓=exists, ✗=does not exist, 🚫=not accessible



# Popular endpoints

- Clear evidence of tailored attacks
  - Bots first identify that a site is WordPress-powered
  - Then, they start bruteforcing credentials

- **Implication**: If you don't run multiple types of applications, you won't see a malicious bot



# JavaScript and Bot Behaviors

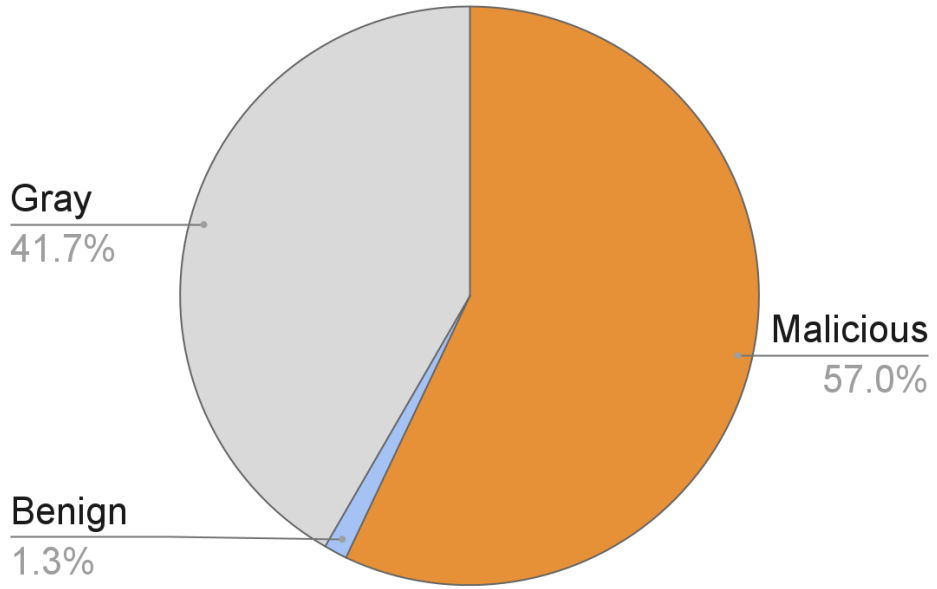

- Out of 1.7M sessions, only 11K (0.63%) supported JavaScript
  - No JavaScript, no JavaScript-based fingerprinting
  - Fingerprints submitted on only 0.59% of sessions
- Honoring of robots.txt
  - We did not observe **any** violations of robots.txt
  - Popularity of fake disallow entries?
- Shared/Distributed crawling
  - 42.8% of requests originated from different IP addresses than anticipated
    - Widely observed in Google bots (19.6% of all reuse)
  - No evidence of distributed crawling in malicious bots

# Good bot or bad bot?

- We classify the connecting bots as follows:

- Benign
  - Verified search-engine bots
  - Bots by security researchers and companies
- Malicious
  - Sending unsolicited POST requests towards auth endpoints
  - Send fingerprinting-related, vulnerability-related requests
- Other
  - Remainder… we don't know much about those

| Type | Total SEBot Requests | Verified Requests |
|---|---|---|
| Googlebot | 233,024 | 210,917 (90.5%) |
| Bingbot | 77,618 | 77,574 (99.9%) |
| Baidubot | 2,284 | 61 (0.026%) |
| Yandexbot | 4,894 | 4,785 (97.8%) |
| **Total** | 317,820 | 293,337 (**92.3%**) |

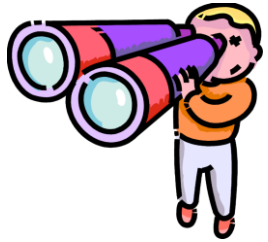


# Bad Bots Brute-forcing

- Credential brute-forcing attempts
  - 50.8% of total requests
  - 47,667 unique IP addresses
  - Trying common passwords as well as the domain itself
    - www.example.com as a password for admin panel of example.com
  - 99.6% of bots issued fewer than 10 attempts
    - "Spray and pray"
    - We had observed the same phenomenon on SSH honeypots, in 2017 [A]

*[A] Barron et al. "Picky Attackers: Quantifying the Role of System Properties on Intruder Behavior", ACSAC 2017*

# Bad bots: Reconnaissance

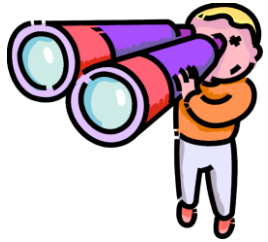


- Application fingerprinting
  - Attempting to infer the version of a web application or its plugins
  - Matched requests against signatures of WhatWeb and BlindElephant
  - 223K requests, 12K bot IP addresses
- Exploitation attempts
  - We focused on server-side exploits from exploit-db (593 signatures)
  - 238K requests, 10K bot IP addresses

| Path | # requests | Unique IPs | Target applications |
|---|---|---|---|
| /CHANGELOG.txt | 116,513 | 97 | Drupal, Joomla, Moodle and spip |
| /(thinkphp\|TP)/(public\|index) | 55,144 | 3,608 | ThinkPHP |
| /wp-content/plugins | 32,917 | 2,416 | WordPress |
| /solr/ | 23,307 | 919 | Apache Solr |
| /manager/html | 10,615 | 1,557 | Tomcat Manager |

| Path | # requests | Unique IPs | CVE/EDB-ID |
|---|---|---|---|
| /vendor/phpunit/.../eval-stdin.php | 70,875 | 346 | CVE-2017-9841 |
| /scripts/setup.php | 67,417 | 1,567 | CVE-2009-1151 |
| /?XDEBUG_SESSION_START=phpstorm | 23,447 | 7 | EDB-44568 |
| /?a=fetch&content=<php>die(@md5(HelloThinkCMF))</php> | 21,819 | 953 | CVE-2019-7580 |
| /cgi-bin/mainfunction.cgi | 20,105 | 2,055 | CVE-2020-8515 |

# Bad bots: Reconnaissance



- Searching for backdoors
  - shell.php, cmd.php, up.php
  - 144K requests, 6.7K unique IP addresses
- Searching for unprotected files
  - .old, .sql, .php~, .zip, .bak, .env
  - 52K requests, 5.8K unique IP addresses

- 929 bots did all of the above
  - Minority of bots willing to keep attacking until they are either blocked or they run out of vectors

# Bots and TLS fingerprinting

- Unlike JS fingerprinting, TLS fingerprinting worked really well
  - 558 unique fingerprints shared over 10M requests
    - Small number of tools and libraries
- 86.2% of bots claiming Firefox/Chrome were fake
  - Matching signatures of curl, libwww-perl, Go, and Python
- Exploitation attempts do not match real browser fingerprints

| Tools | Unique FPs | IP Count | Total Requests |
|---|---|---|---|
| Go-http-client | 28 | 15,862 | 8,708,876 |
| Libwww-perl or wget | 17 | 6,102 | 120,423 |
| PycURL/curl | 26 | 3,942 | 80,374 |
| Python-urllib 3 | 8 | 2,858 | 22,885 |
| NetcraftSurveyAgent | 2 | 2,381 | 14,464 |
| msnbot/bingbot | 4 | 1,995 | 44,437 |
| Chrome-1(Googlebot) | 1 | 1,836 | 28,082 |
| Python-requests 2.x | 11 | 1,063 | 754,711 |
| commix/v2.9-stable | 3 | 1,029 | 5,738 |
| Java/1.8.0 | 8 | 308 | 1,710 |
| MJ12Bot | 2 | 289 | 28,065 |
| Chrome-2(Chrome, Opera) | 1 | 490 | 66,631 |
| Chrome-3(Headless Chrome) | 1 | 80 | 2,829 |
| Chrome-4(coc_coc_browser) | 1 | 4 | 101 |
| **Total** | **113** | 38,239 | 9,879,326 |

***TABLE V:*** *TLS fingerprint of malicious requests*

| Type | Python | Golang | libwww / wget | Chrome / Firefox | Unknown | Total |
|---|---|---|---|---|---|---|
| Backdoor | 231 | 1,718 | 349 | 3 | 482 | 2,783 |
| Backup File | 411 | 171 | 84 | 0 | 1,803 | 2,469 |
| Exploits | 275 | 18,283 | 607 | 0 | 390 | 19,555 |
| Fingerprinting | 1,524 | 3,670 | 630 | 139 | 7,226 | 13,189 |

# Case study

- Time to weaponize
  - 5 RCE vulnerabilities got discovered during our 7-month study
  - Aristaeus could now observe how fast attackers weaponize a new exploit

| Software/Firmware | CVE | Time to weaponize |
|---|---|---|
| MSSQL Reporting Servers | CVE-2020-0618 | 4 days |
| Liferay Portal | CVE-2020-7961 | 4 days |
| DrayTech modems | CVE-2020-8585 | 2 days |
| Netgear GPON router | EDB-48225 | **Same day** |
| F5 Traffic Management UI | CVE-2020-5902 | **Same day** |

# Basis of today's talk

Good Bot, Bad Bot:
Characterizing Automated Browsing Activity

Xigao Li
Stony Brook University

Babak Amin Azad
Stony Brook University

Amir Rahmati
Stony Brook University

Nick Nikiforakis
Stony Brook University

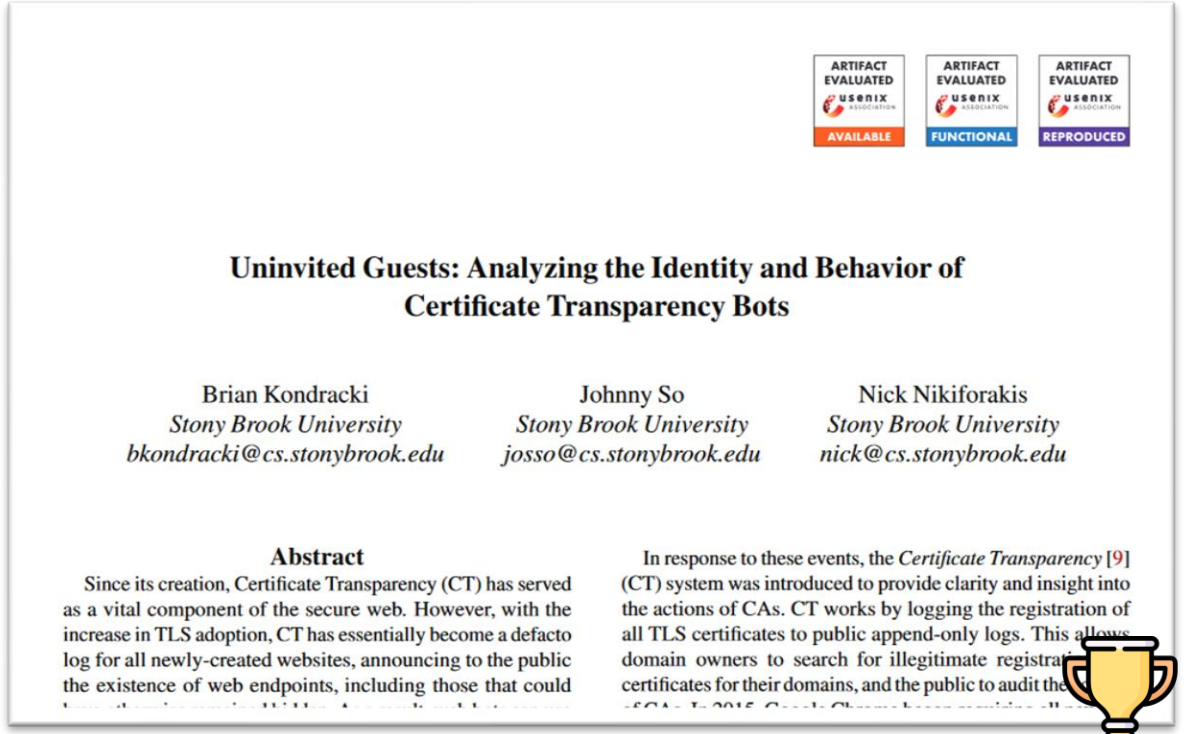
*Abstract*—As the web keeps increasing in size, the number of vulnerable and poorly-managed websites increases commensurately. Attackers rely on armies of malicious bots to discover these vulnerable websites, compromising their servers, and exfiltrating sensitive user data. It is, therefore, crucial for the security of the web to understand the population and behavior of malicious bots.

In this paper, we report on the design, implementation, and results of Aristaeus, a system for deploying large numbers of "honeysites", i.e., websites that exist for the sole purpose of attracting and recording bot traffic. Through a seven-month-long experiment with 100 dedicated honeysites, Aristaeus recorded 26.4 million requests sent by more than 287K unique IP addresses, with 76,396 of them belonging to clearly malicious bots. By analyzing the type of requests and payloads that these bots send, we discover that the average honeysite received more than 37K requests each month, with more than 50% of these requests attempting to brute-

their ability to claim arbitrary identities (*e.g.,* via User-agent header spoofing), and the automated or human-assisted solving of CAPTCHAs make this a challenging task [9]–[11].

In this paper, we present a technique that sidesteps the issue of differentiating between users and bots through the concept of *honeysites*. Like traditional high-interaction honeypots, our honeysites are fully functional websites hosting full-fledged web applications placed on public IP address space (similar to Canali and Balzarotti's honeypot websites used to study the exploitation and post-exploitation phases of web-application attacks [12]). By registering domains that have never existed before (thereby avoiding traffic due to residual trust [13]) and never advertising these domains to human users, we ensure

IEEE S&P, 2021



ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

**Uninvited Guests: Analyzing the Identity and Behavior of Certificate Transparency Bots**

Brian Kondracki
*Stony Brook University*
*bkondracki@cs.stonybrook.edu*

Johnny So
*Stony Brook University*
*josso@cs.stonybrook.edu*

Nick Nikiforakis
*Stony Brook University*
*nick@cs.stonybrook.edu*

**Abstract**

Since its creation, Certificate Transparency (CT) has served as a vital component of the secure web. However, with the increase in TLS adoption, CT has essentially become a defacto log for all newly-created websites, announcing to the public the existence of web endpoints, including those that could

In response to these events, the *Certificate Transparency* [9] (CT) system was introduced to provide clarity and insight into the actions of CAs. CT works by logging the registration of all TLS certificates to public append-only logs. This allows domain owners to search for illegitimate registrations certificates for their domains, and the public to audit the



USENIX Security, 2022

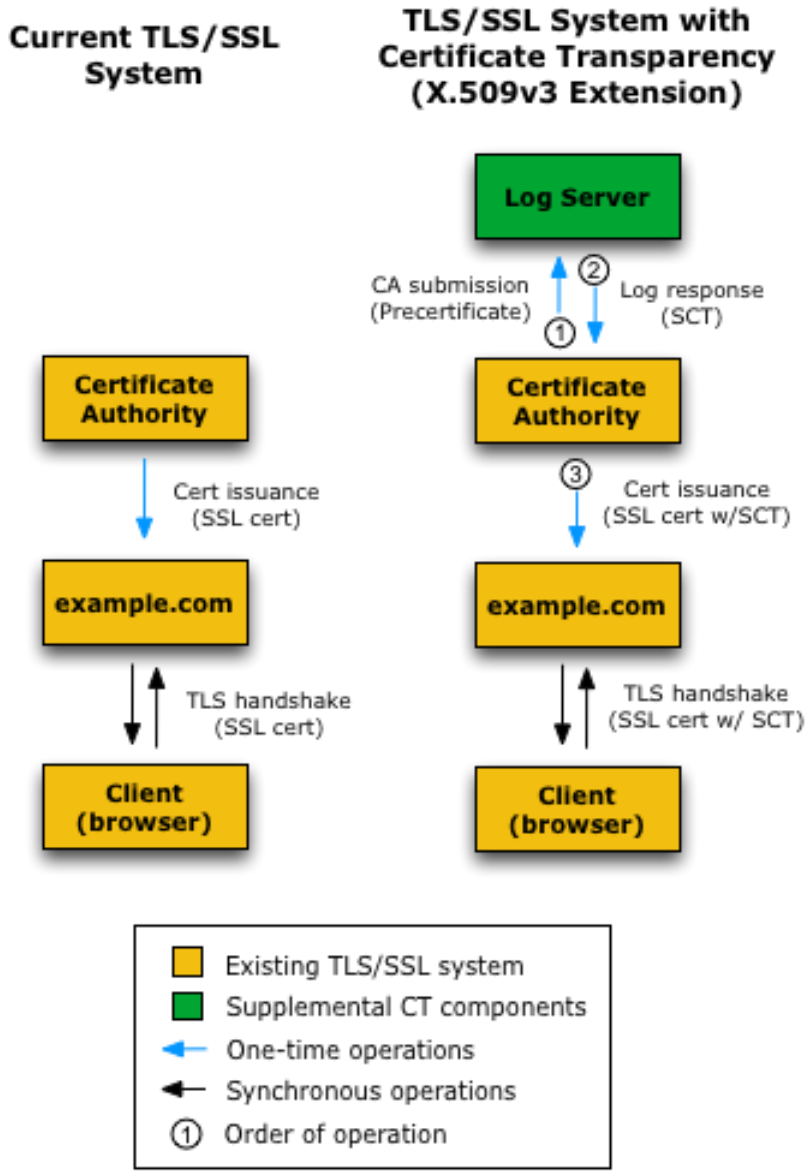
# How Certificate Transparency came to be

- High-profile root Certificate Authority security incidents in 2010s
  - Compromised
    - Comodo
    - DigiNotar
  - Misbehaving/Misconfigured
    - TrustWave
    - TurkTrust
- Unwanted activity would be discovered much later
  - E.g. when the attacker successfully deployed the illicit certificates



# What is Certificate Transparency?

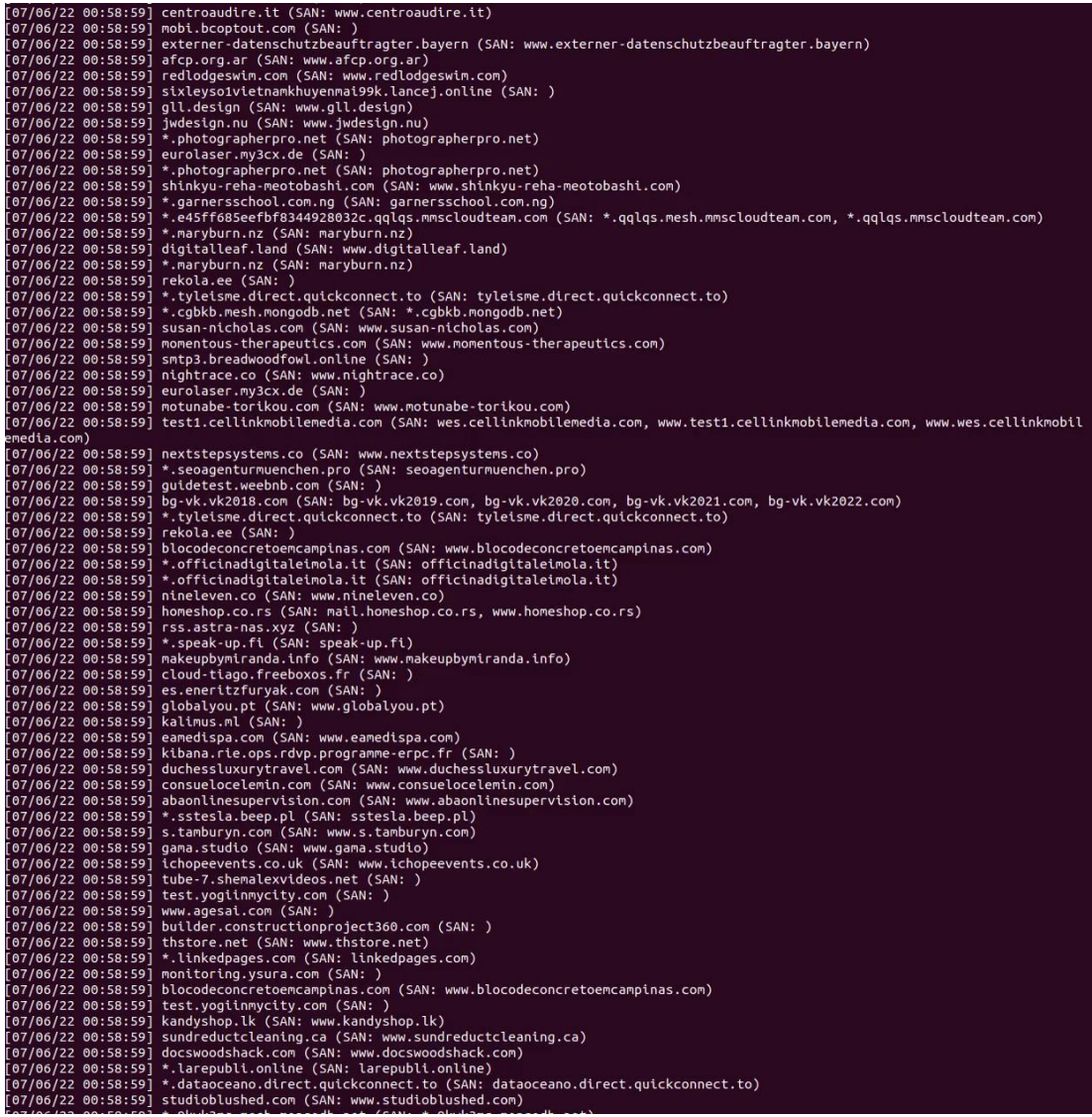
- Issuing certificates should not be a secret
  - Make it fully transparent which CA is issuing which certificates for which domains
- Proposal: use third party for append-only log
  - after (pre-)certificate submission, log issues Signed Certificate Timestamp (SCT)
  - CA adds SCT to certificate, signs it, hands out
- Chrome only allows Symantec certificate with EV if they are in CT logs
  - enforced since June 2016
- Since April 2018, all new certificates must have an SCT



# Uses of certificate transparency

- It is straightforward for people to get access to certificate transparency logs
  - Search for specific domains
  - Get alerts when specific domains/variations of domains are issued certificates
  - Get access to the constant stream of issued certificates

- Alternative uses
  - Identify phishing sites and sites that abuse trademarks
  - **Identify new targets for attacks**
    - **Endpoints that could have otherwise remained hidden**

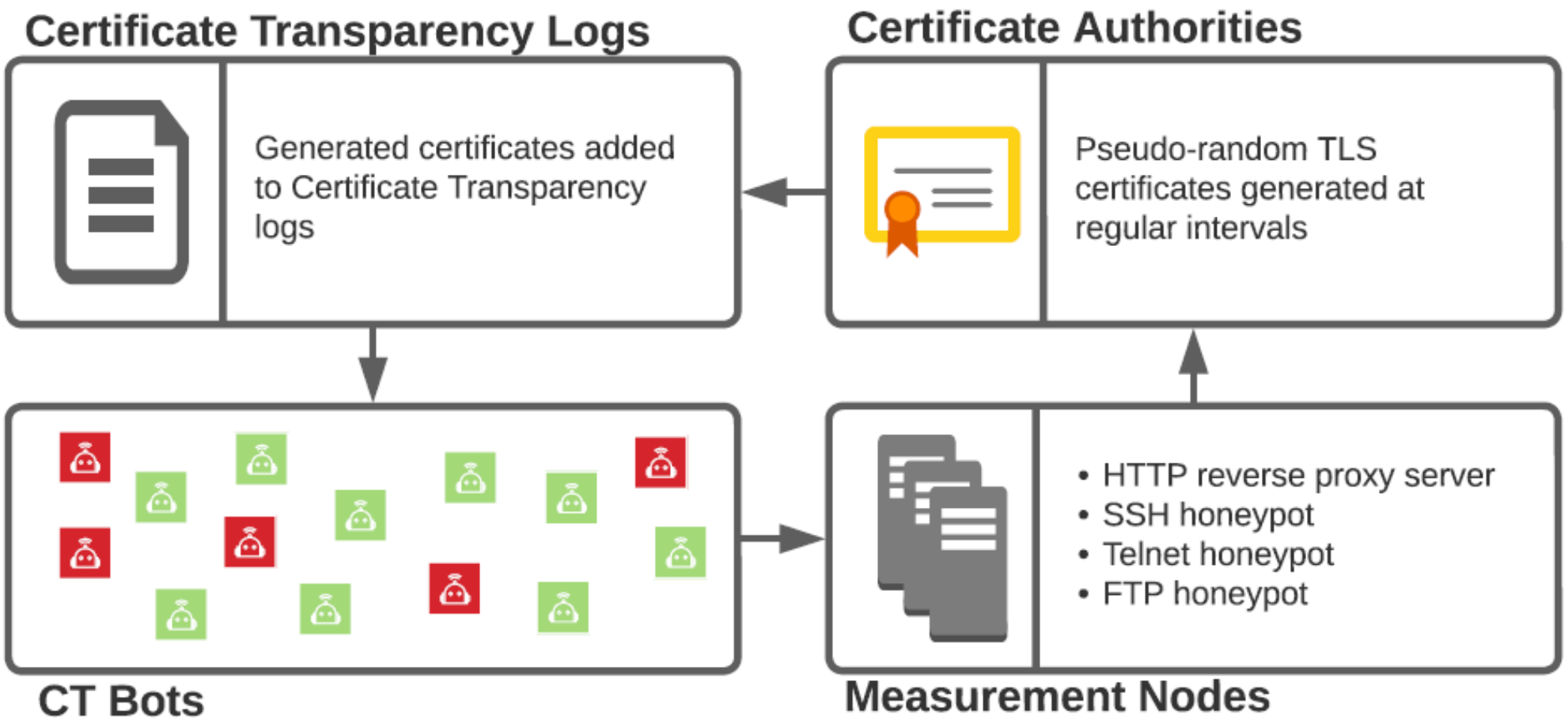


# Research questions

- Do web bots monitor CT logs for targets?
- Is the targeting based on the makeup of each domain?
- What is the overall behavior of CT bots?

- CTPOT
  - The first distributed honeypot system built specifically for Certificate Transparency
    - Lure bots to our honeypots
    - Fingerprint them and study their behavior

# Architecture of CTPOT



# Building attractive domains

*bwr11215lkj013247.wp-admin.elmlilydove.xyz*

- Timestamp encoded in first-level subdomain
  - These subdomains are entirely invisible to everyone outside of CT

# Building attractive domains

*bwr11215lkj013247.wp-admin.elmlilydove.xyz*

- Target encoded in second-level subdomain
  - Three types of targets: impersonating, sensitive, baseline

| Impersonating |
|---|
| google |
| facebook |
| twitter |
| paypal |
| etc. |

| Sensitive |
|---|
| wp-admin |
| sql |
| demo |
| mail |
| etc. |

| Baseline |
|---|
| banana |
| pear |
| apple |
| carrot |
| etc. |

# Building attractive domains

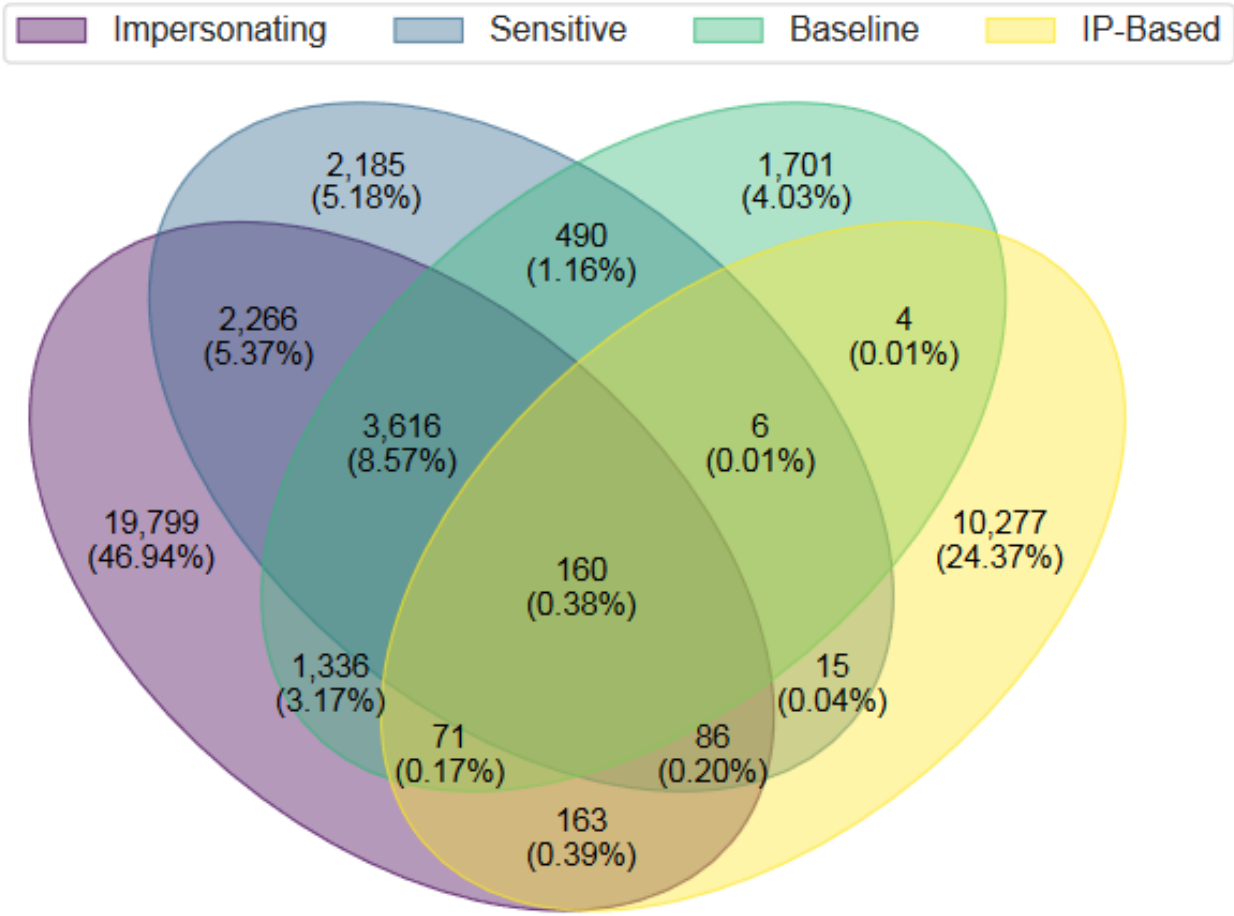
*bwr11215lkj013247.wp-admin.elmlilydove.xyz*

- Primary domain composed of appending benign words together (trees, flowers and birds)
  - Benign and uninteresting
  - All possible trademarks removed

- **Goal**: Force CT bots to make a decision based on the presented subdomain

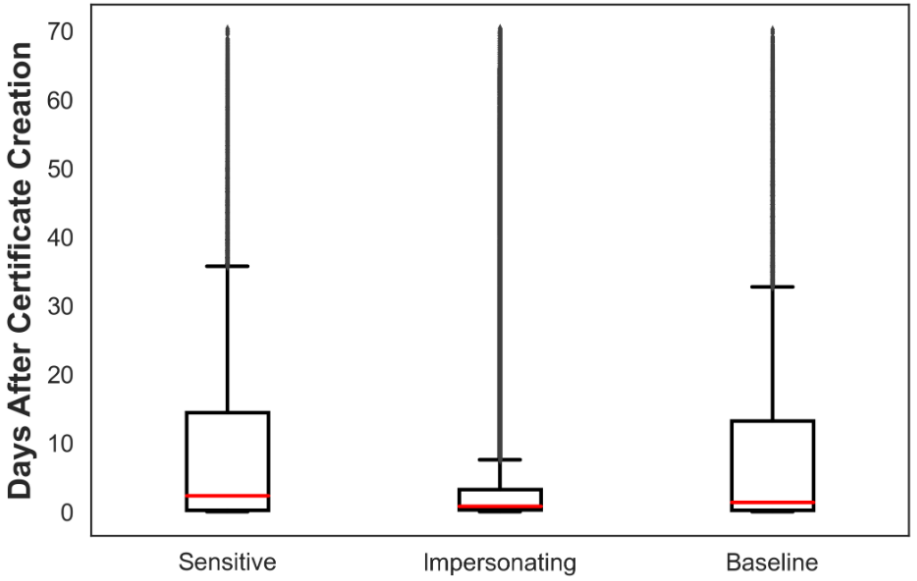
# CTPOT Deployment

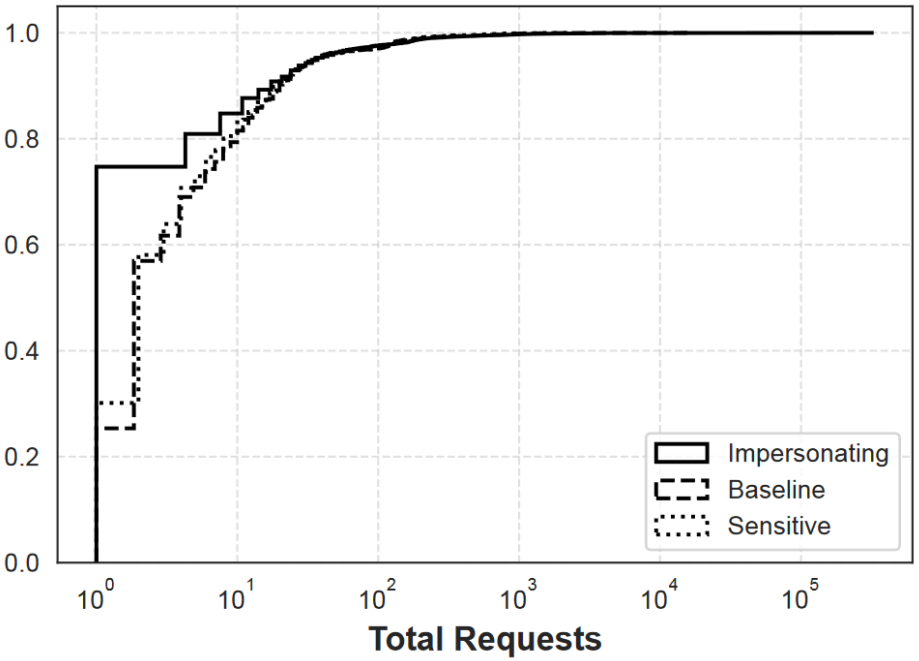
- Deployed CTPOT for 12 weeks
  - 4,657 TLS certificates requested
- Results
  - 1.5 million requests from 31,898 IP addresses
  - Distinct bots, compared to IP-address-based server discovery



# CT bot request statistics

- New domains receive requests as fast as 12 seconds after certificate creation
  - No time to spin up an outdated/vulnerable server and secure it online
- Diverging behavior among bots targeting different domains
  - CT bots targeting impersonating websites are much less persistent



# CT Bots self-identification

*HTTP User Agent*

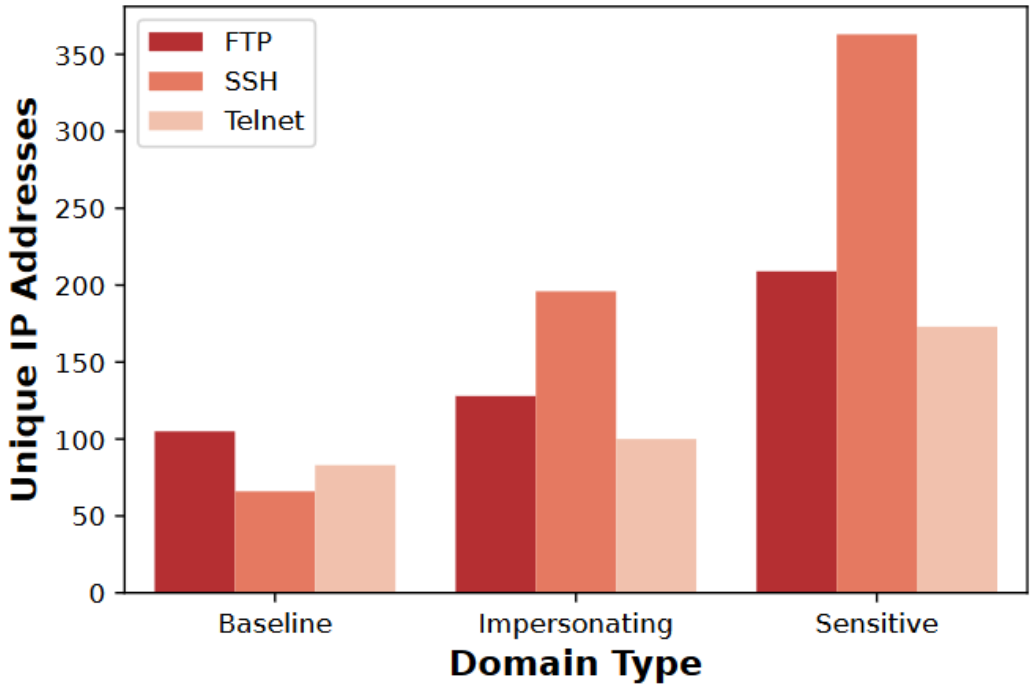
| User-agent Type | Impersonating | Sensitive | Baseline |
|---|---|---|---|
| Browser | 84.71% | 78.38% | 76.11% |
| Academic/ Industry | 5.64% | 13.44% | 15.10% |
| Library | 3.90% | 1.79% | 4.31% |
| Scanning Tool | 3.01% | 3.50% | 3.22% |
| Other | 2.73% | 2.88% | 1.23% |

*TLS Fingerprint*

| Fingerprint Type | Impersonating | Sensitive | Baseline |
|---|---|---|---|
| Library | 40.94% | 17.58% | 12.07% |
| Academic/ Industry | 20.37% | 30.63% | 33.45% |
| Unknown | 14.46% | 20.46% | 25.70% |
| Scanning Tool | 12.52% | 28.22% | 26.59% |
| Browser | 11.59% | 2.94% | 1.98% |

# Malicious behavior from CT Bots

- Some of the CT Bots do not stop at HTTP requests
  - 90.5% of network-probing bots attempted to authenticate
- Less than 5% of the bot IP addresses were present in blocklists
  - Highlighting the completeness issues of blocklists
  - Method for possible blocklist augmentation



# Conclusion

Stony Brook University

PragSec Lab

- As the web keeps growing, so does the volume of attacks against web applications
  - Attackers are automating both the discovery and the exploitation of vulnerable hosts and services
    - Traditional and zero-day attacks launched against public websites
  - Manual hardening of hosts and networks is not fast enough
- Certificate Transparency allows everyone to audit certificates
  - Including attackers who can abuse it to identify new targets as soon as they get online
- CTPOT allows defenders to study these attackers
  - Protect production systems from the same attackers
  - Engage them in deception


nick@cs.stonybrook.edu
www.securitee.org