Stony Brook University

PragSec Lab

Good Bot, Bad Bot:

Characterizing Automated Browsing Activity

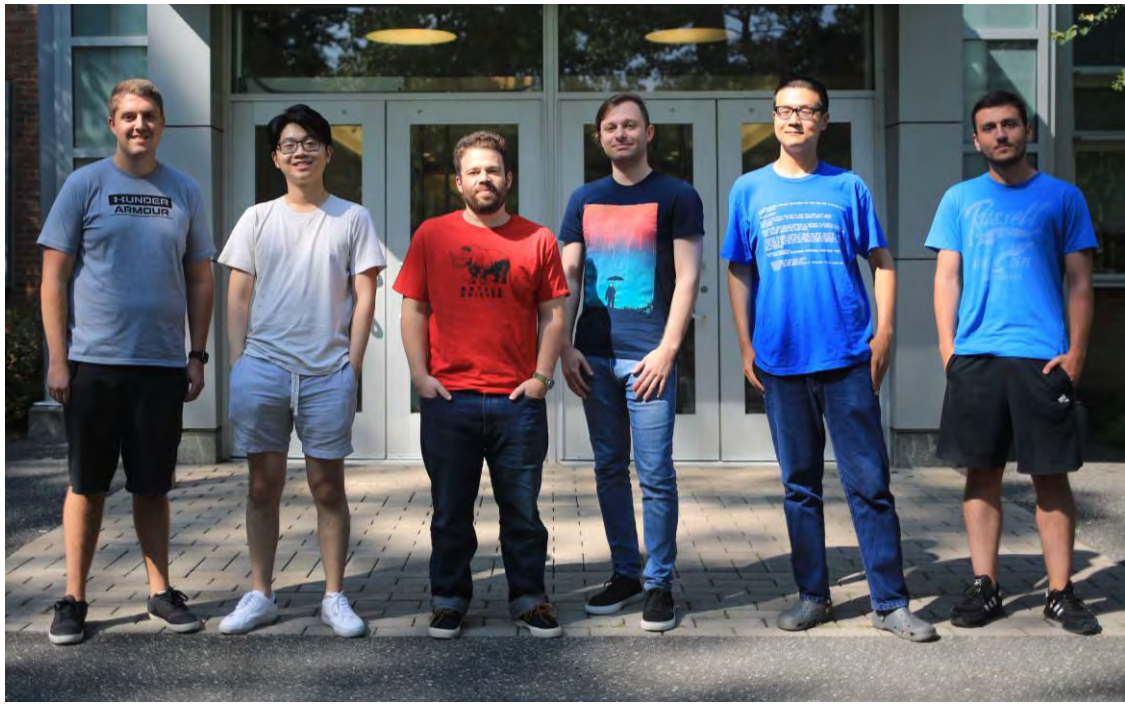
APNIC 52

Nick Nikiforakis

Who am I?

PragSec Lab

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

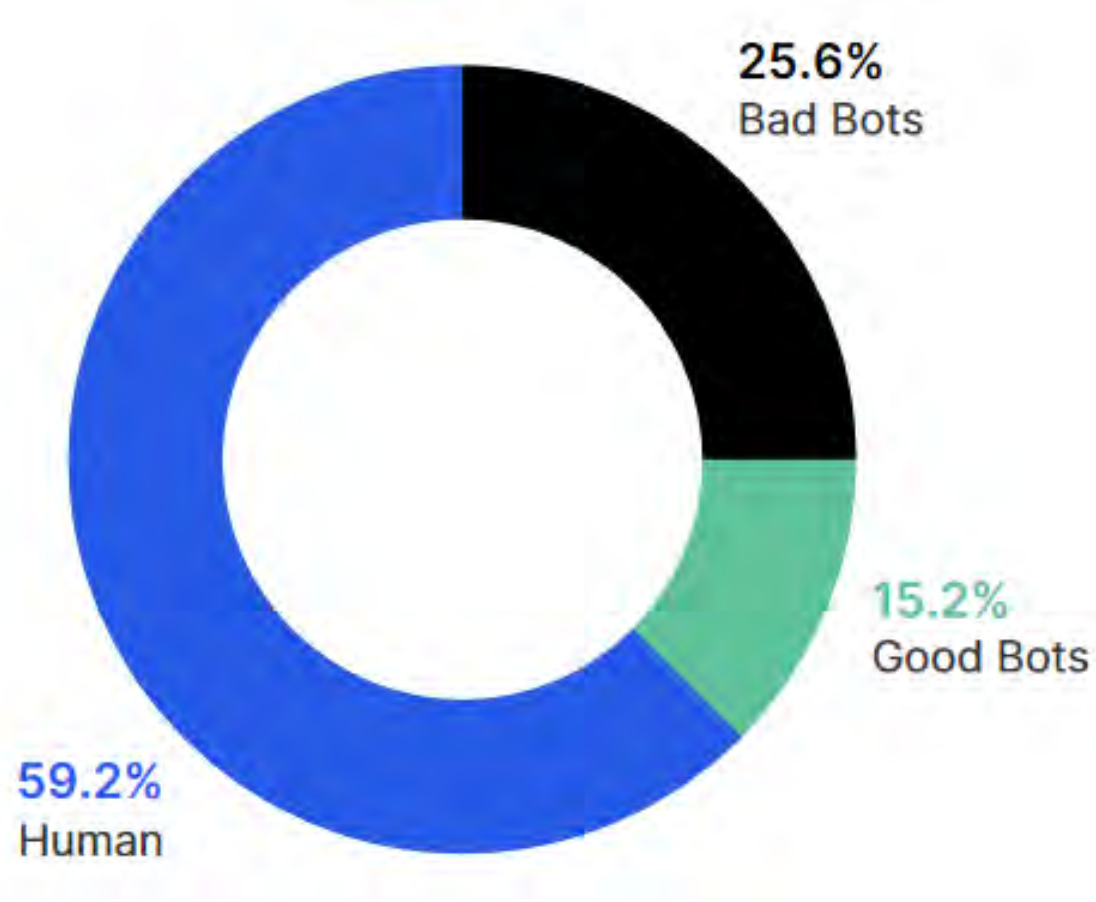


NSF

Office of Naval Research ONR Science & Technology

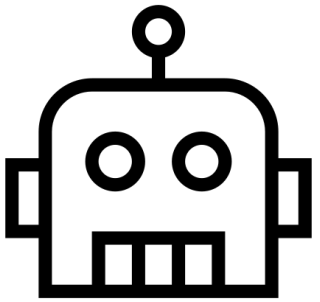
Web bots

- 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, 2021

Detecting benign web bots

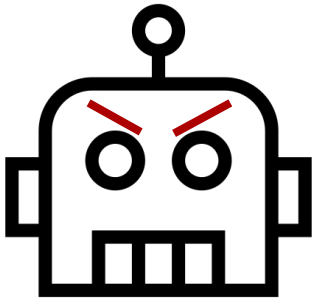


- Benign bots announce themselves

- 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)

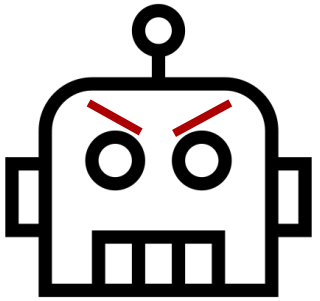
Detecting malicious web bots



- 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

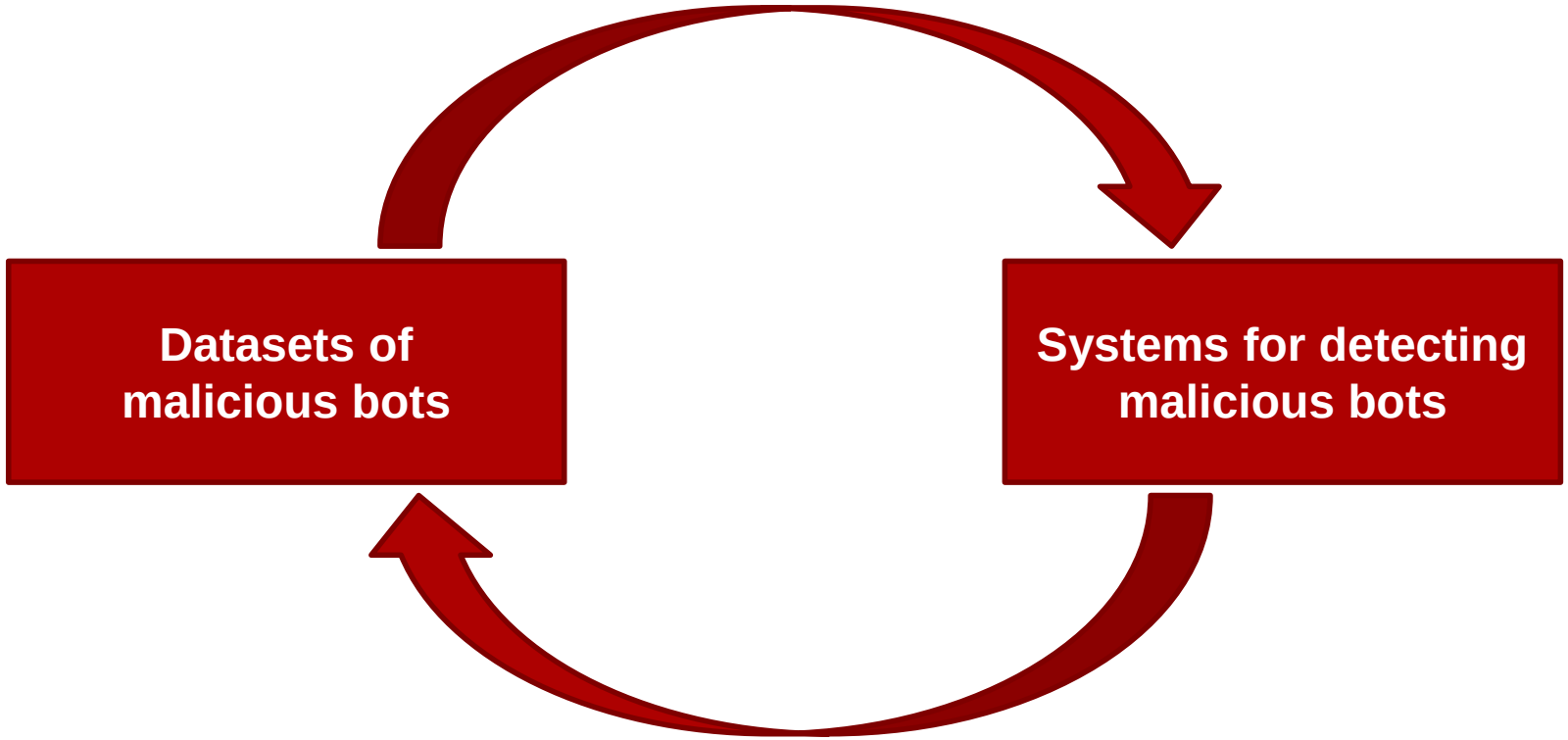
User Agent	IP address	Reverse DNS
Mozilla/5.0 (compatible; Googlebot/2.1…	66.249.66.1	crawl-66-249-66-1.googlebot.com
Mozilla/5.0 (compatible; Googlebot/2.1…	67.245.115.115	cpe-67-245-115-115.nyc.res.rr.com

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
  - ???

Robotic yet circular dependencies



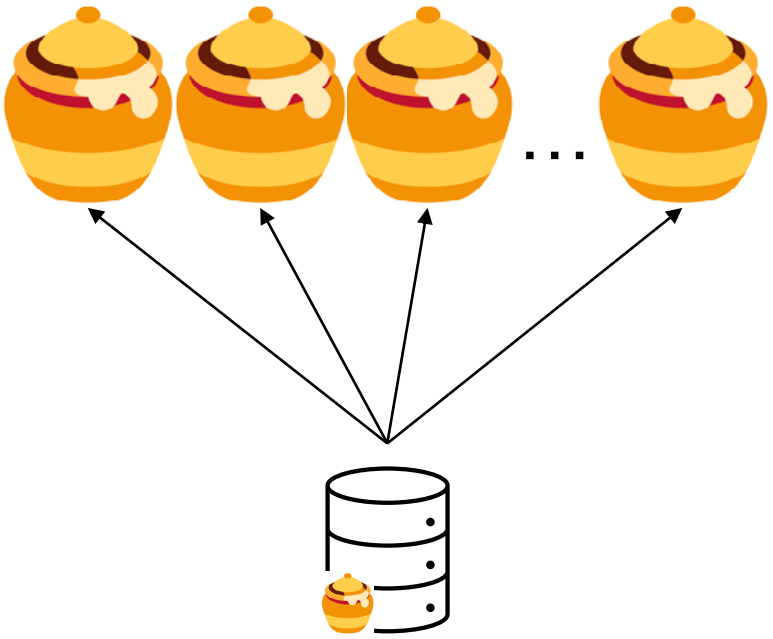
Prior Academic Solutions: Manual filtering of web-server logs

Research questions

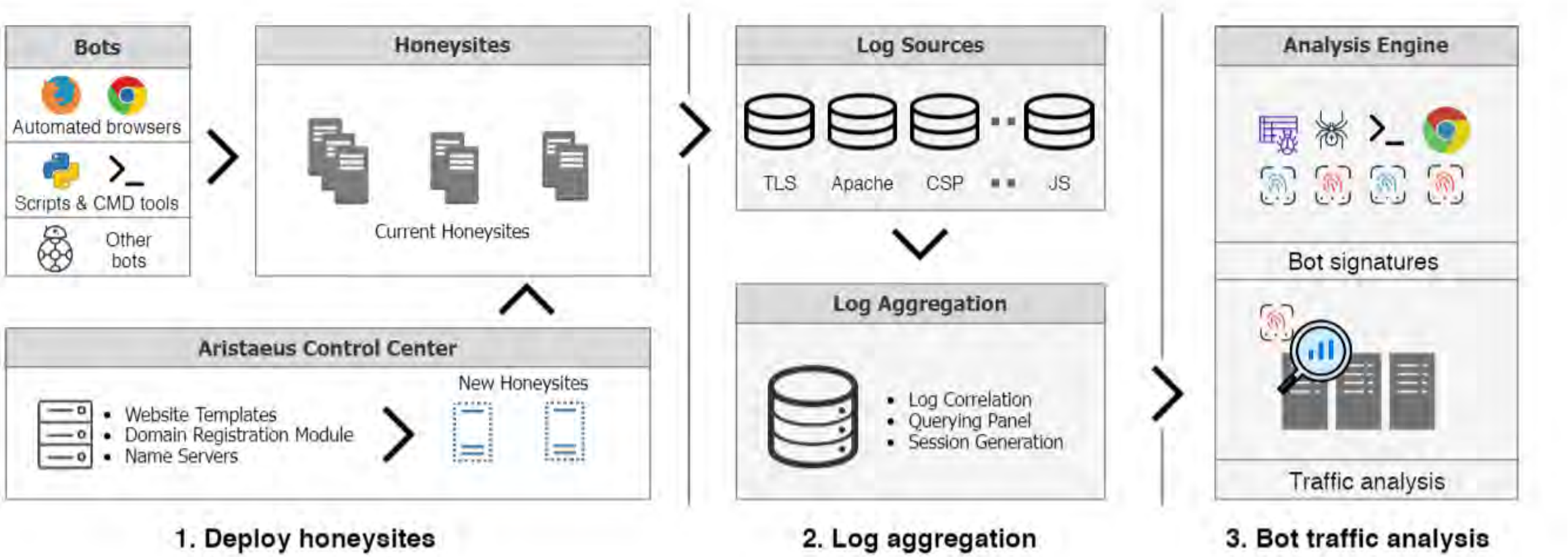
- Can we curate a bot-only dataset in a way that doesn't depend on our manual-analysis prowess?
  - Benign vs. malicious bots
  - Activities of malicious bots
  - Claimed vs. actual identity of malicious bots
  - Trends of bot-activity over time

Network of honeysites

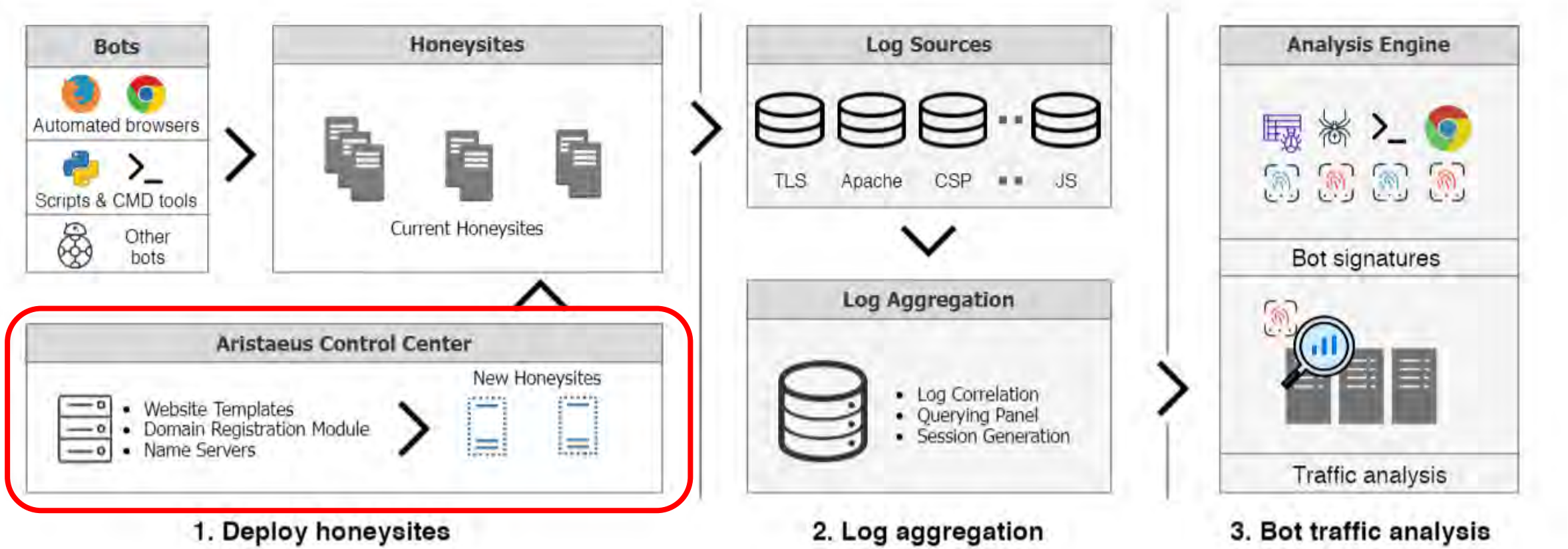
- Aristaeus
  - A system that provides flexible remote deployment and management of honeysites
  - Honeysites:
    - Fully-functional web applications, augmented with state-of-the-art fingerprinting techniques
  - A centralized log server pulls logs from each honeysite on a daily basis
    - Injected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



Overview of Aristaeus

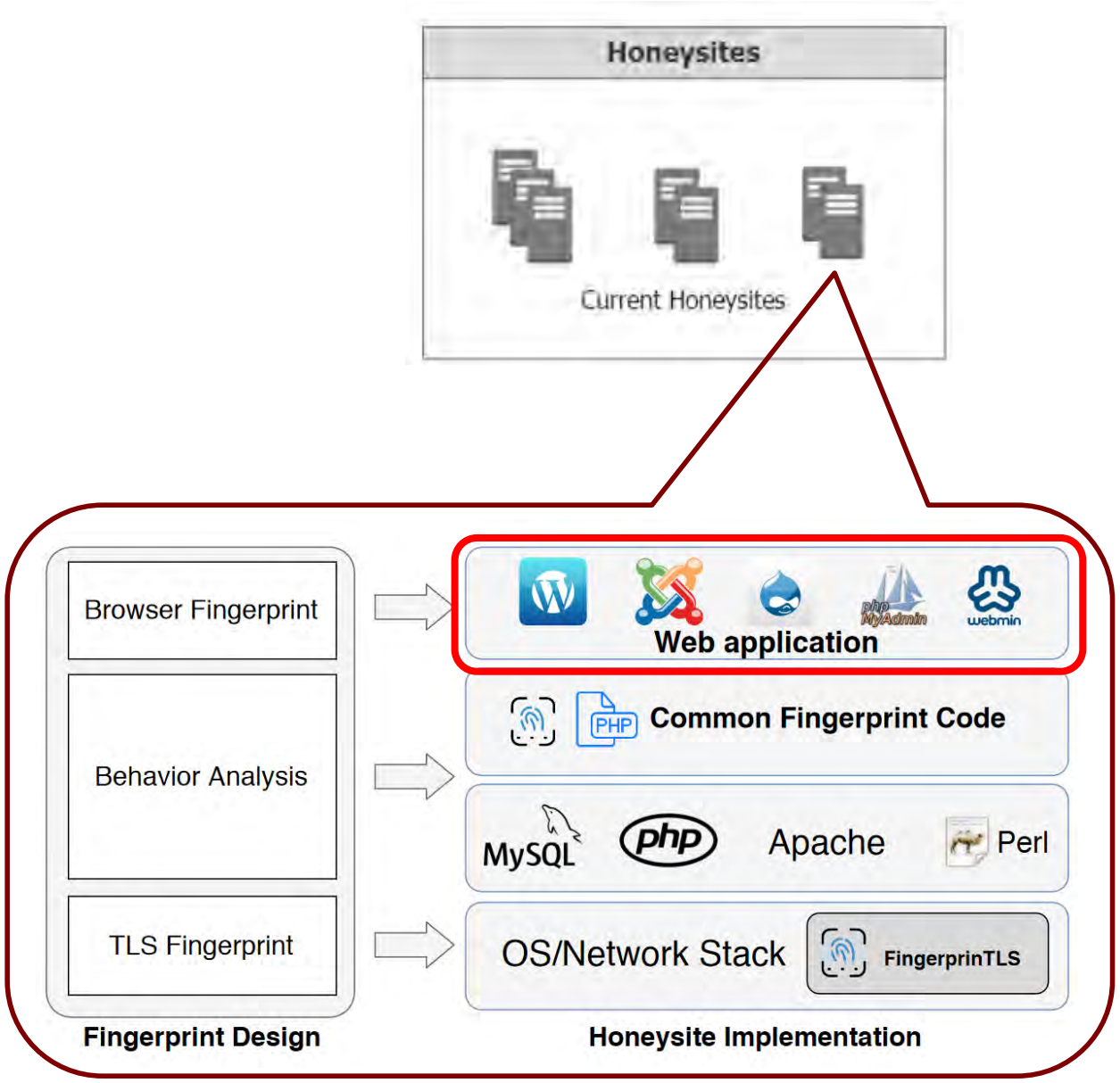


Overview of Aristaeus



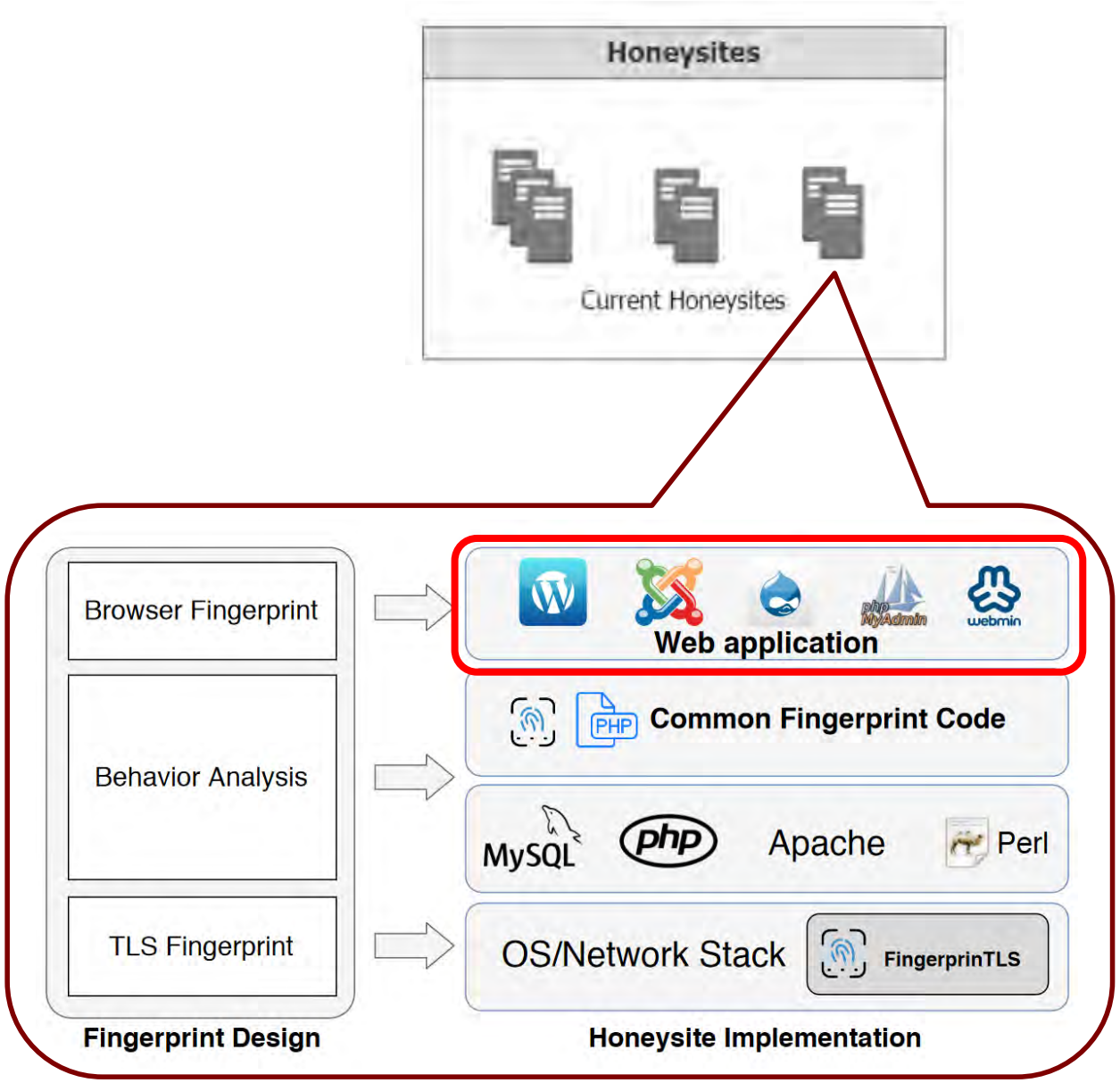
What's the best bait?

- 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



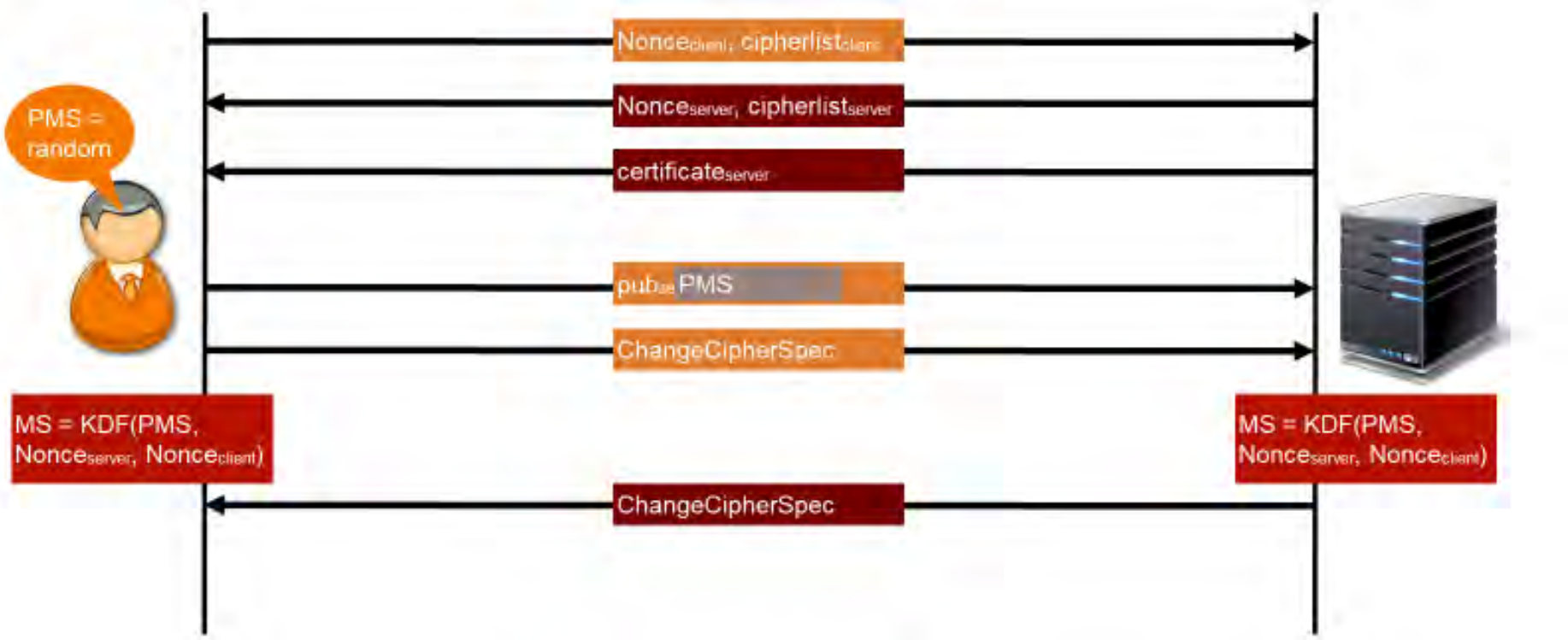
Client fingerprinting

- 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

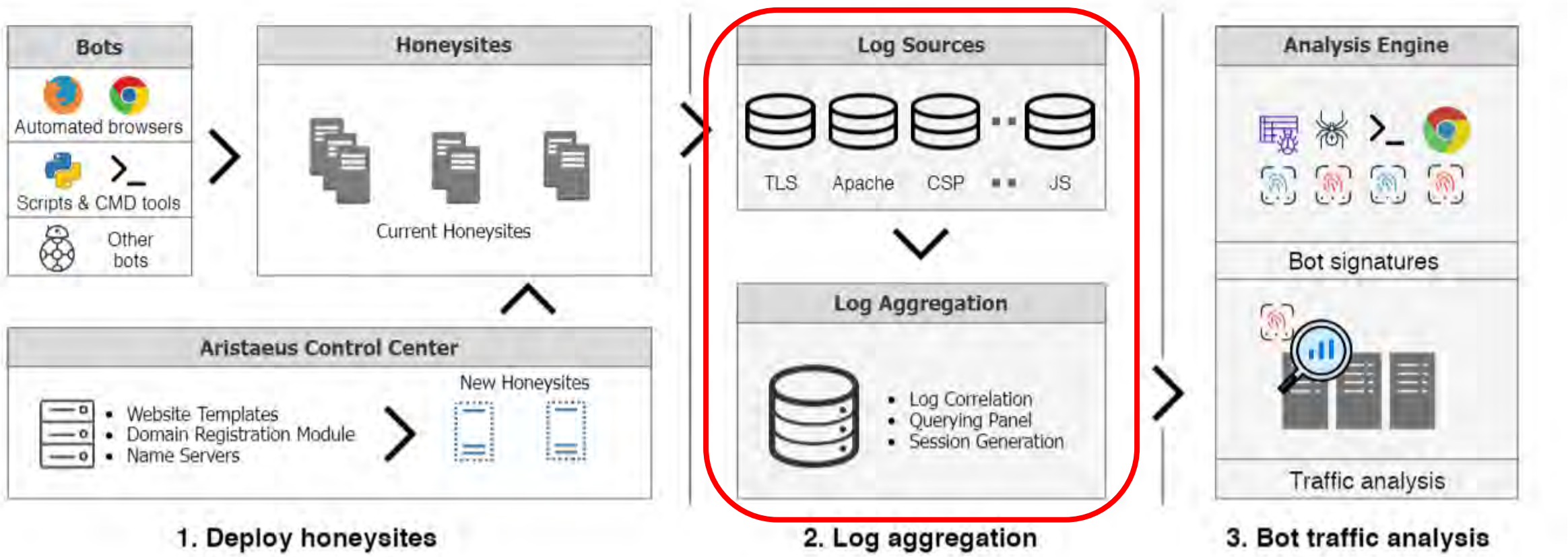
```
import "net/http"

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



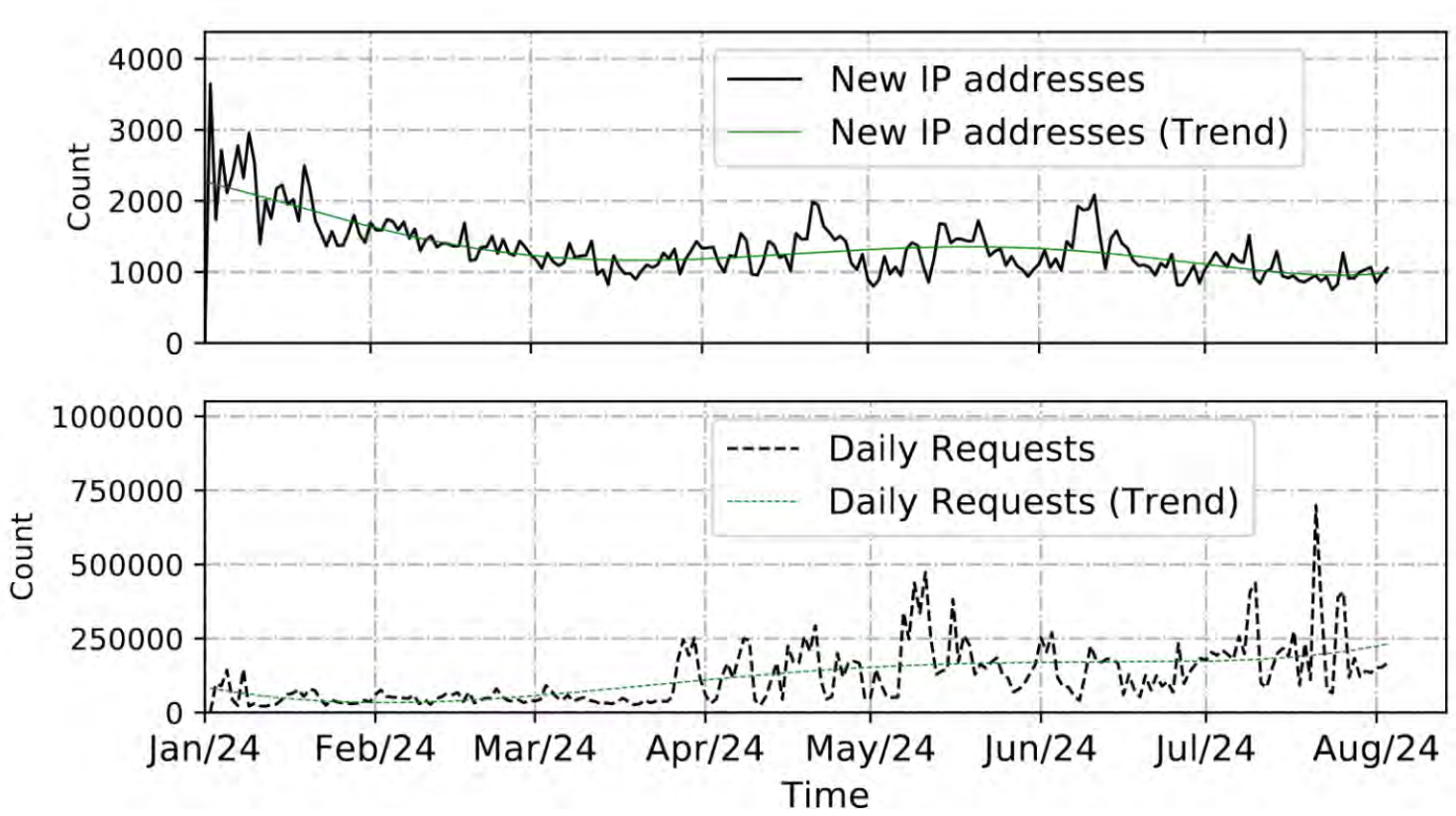
Deployment of Aristaeus

- 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?

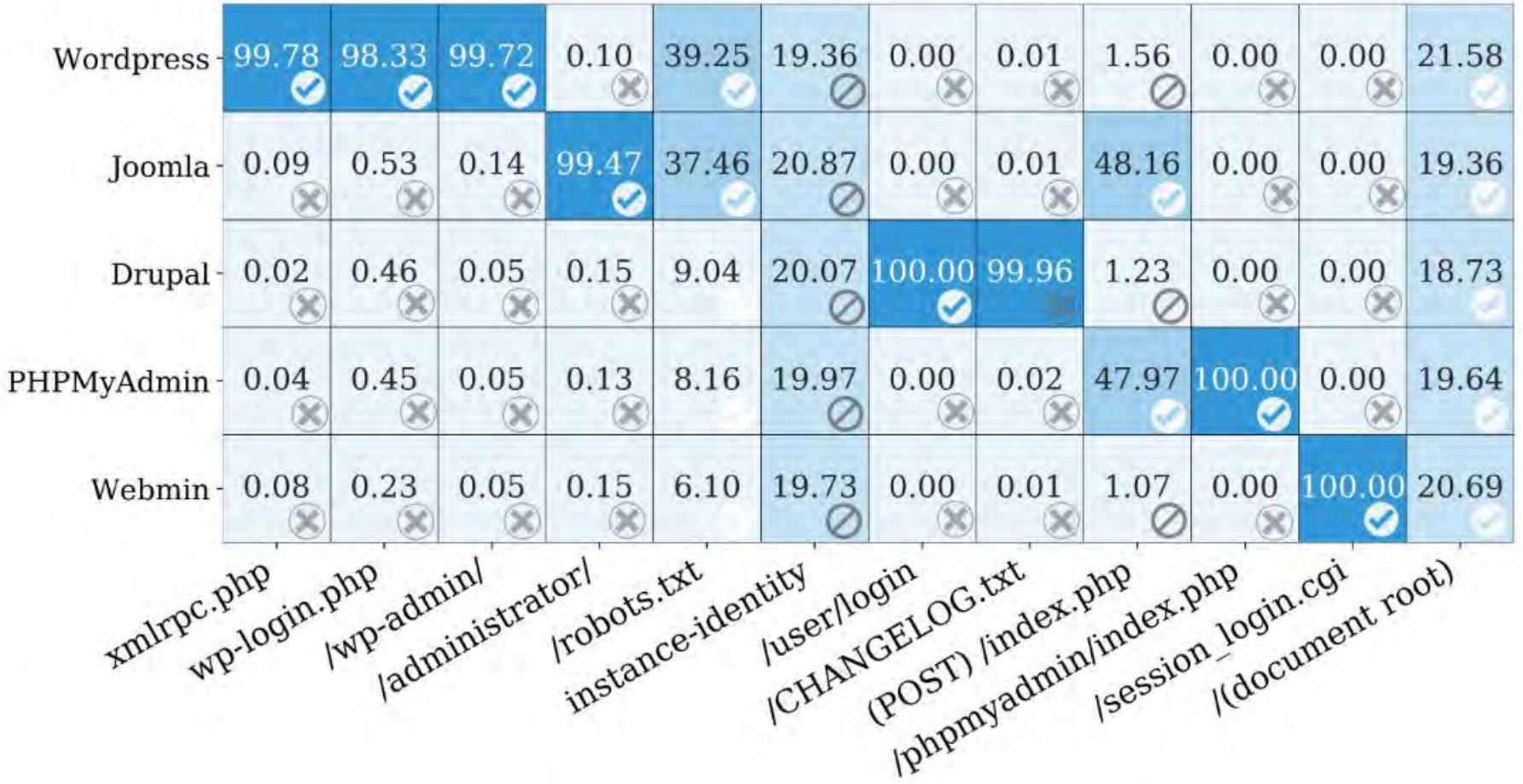
- Inspect the Host header of client-side HTTP headers:
  - 44% of bots visit through the IP address
  - 30% present no Host header
  - 26% explicitly ask for our domains
    - Certificate transparency
    - Zone files
    - Prior crawls


```
"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"
```

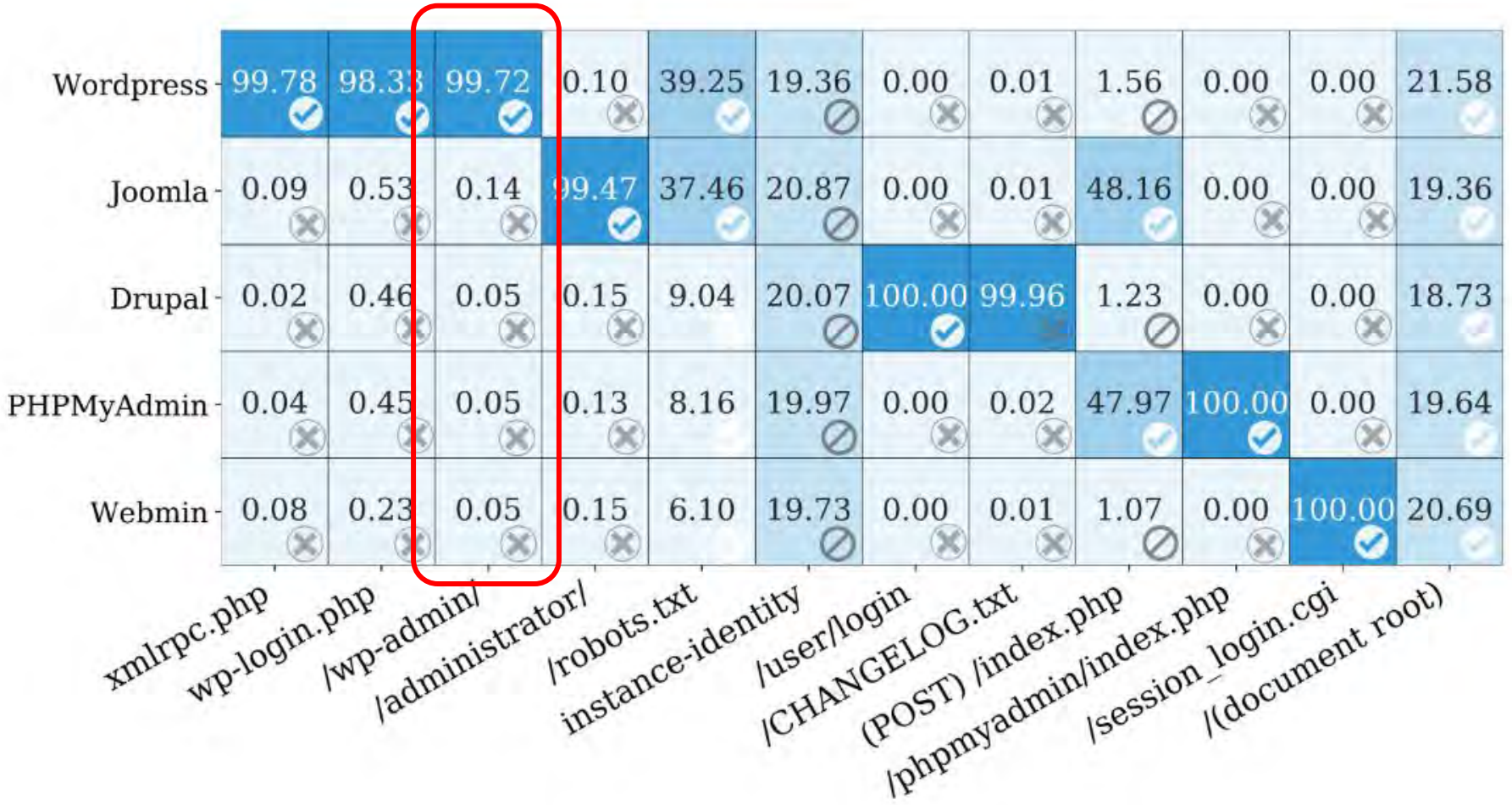
Popular endpoints

✔=exists, ✗=does not exist, ⃠=not accessible



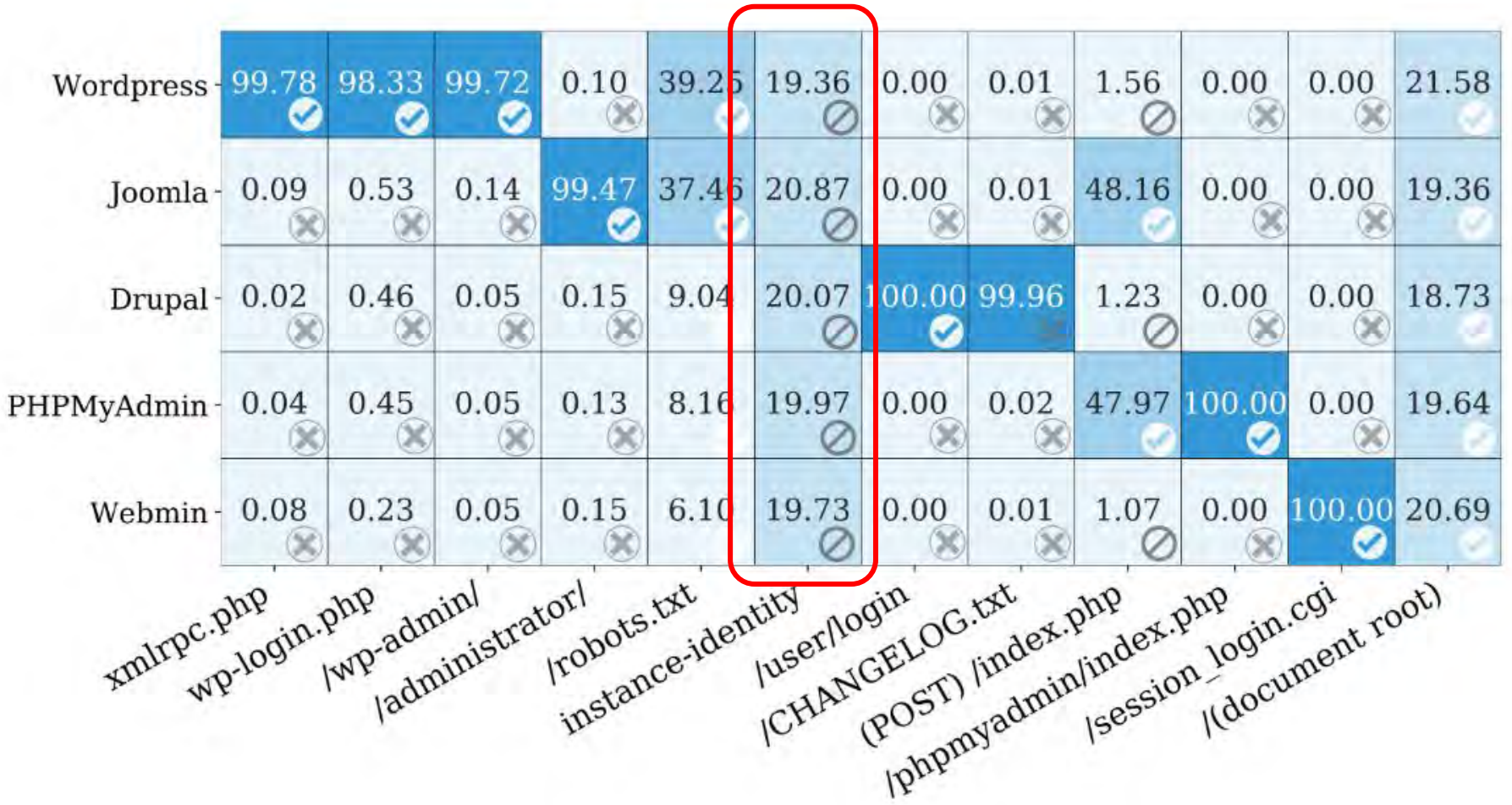
Popular endpoints

✔=exists, ✗=does not exist, ⃠=not accessible



Popular endpoints

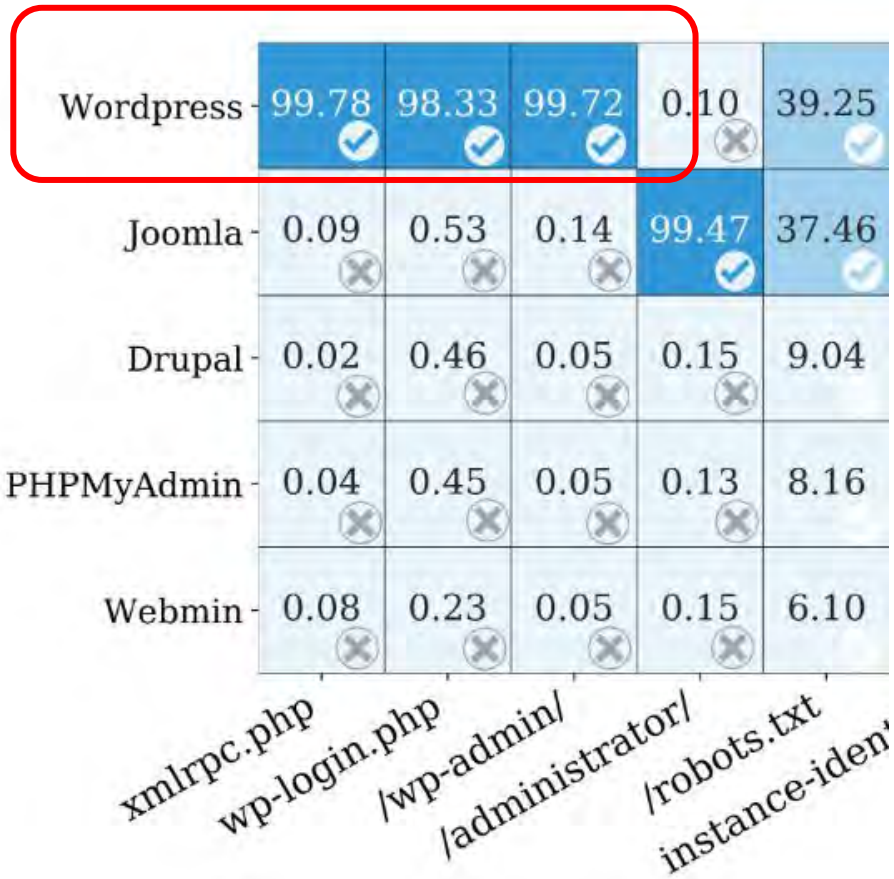
✔=exists, ✗=does not exist, ⃠=not accessible



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

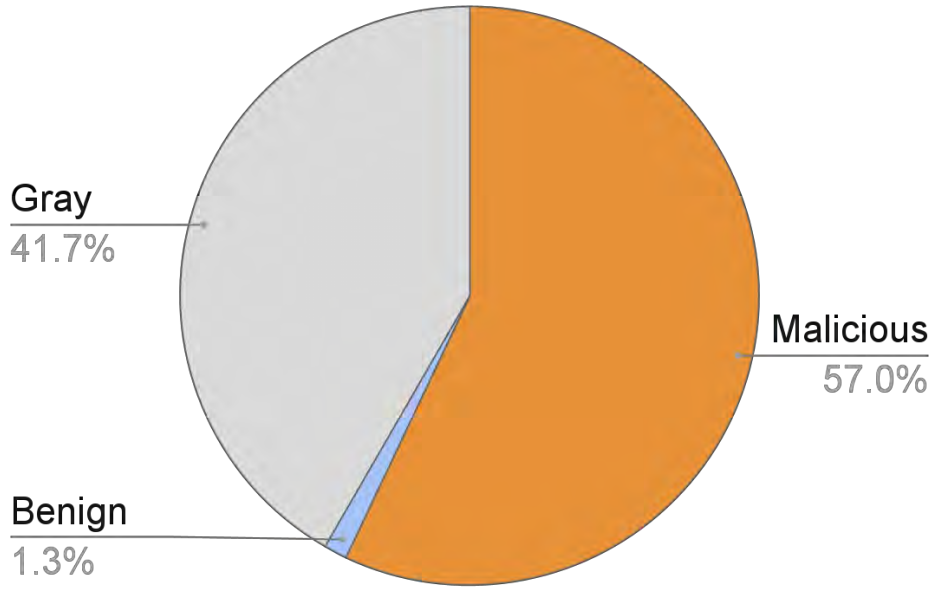
- Out of 1.7M sessions, only 11K (0.63%) supported JavaScript
  - No JavaScript, no JavaScript-based fingerprinting
  - Fingerprints submitted on only 0.59% of sessions
- Honoring of robots.txt
  - We did not observe **any** violations of robots.txt
  - Popularity of fake disallow entries?
- Shared/Distributed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

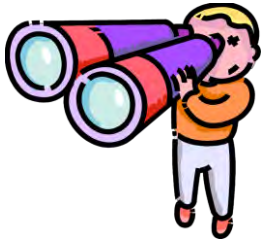
Bad bots: Reconnaissance

- 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

Case studies

- Failed cloaking attempts
  - Bots sending two user agents
    - "User-Agent" **and** "userAgent"
  - Host-header weirdness
    - HOST, hoSt
- 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**

Conclusion

- As more software moves to the web, so do attackers
  - Even unpopular sites are scanned thousands of times a month by malicious bots
- Honeypots and deception technology can help us attract them and fingerprint them
  - For modest operational costs, Aristaeus outperforms popular OSINT blocklists
  - Identify trends in attacker techniques, tools, and sp exploit weaponization
- Communication details can betray a client' identity


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

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