teaching your reverse proxy to “think” (·_·?)

ADEL “0x4D31” KA.
DEF CON 33

finch

whoami

- member of technical staff @ an “ai chatbot startup”!
- security engineer, detection & response

0x4D31

- ➔ **finch**
- ➔ **venator**
- ➔ **galah**
- ➔ **...**

what if i could use...

fingerprint-based routing
to waste attackers' time
intelligently! 🤡

why fingerprint-aware reverse proxy?

in threat detection, we prioritize signals that are harder for attackers to change.

“pyramid of pain”

ip addresses? easy to rotate.
user-agents? trivial to spoof.

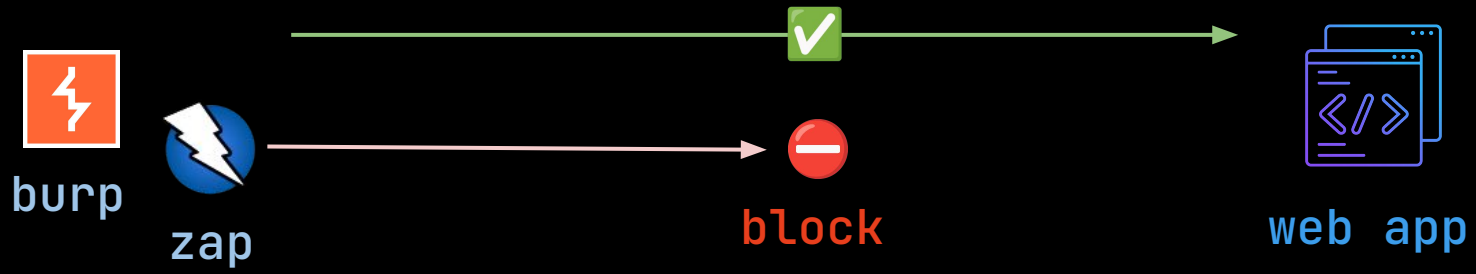
IP addresses? easy to rotate.
user-agents? trivial to spoof.

tls handshakes,
http/2 settings,
header order, ...

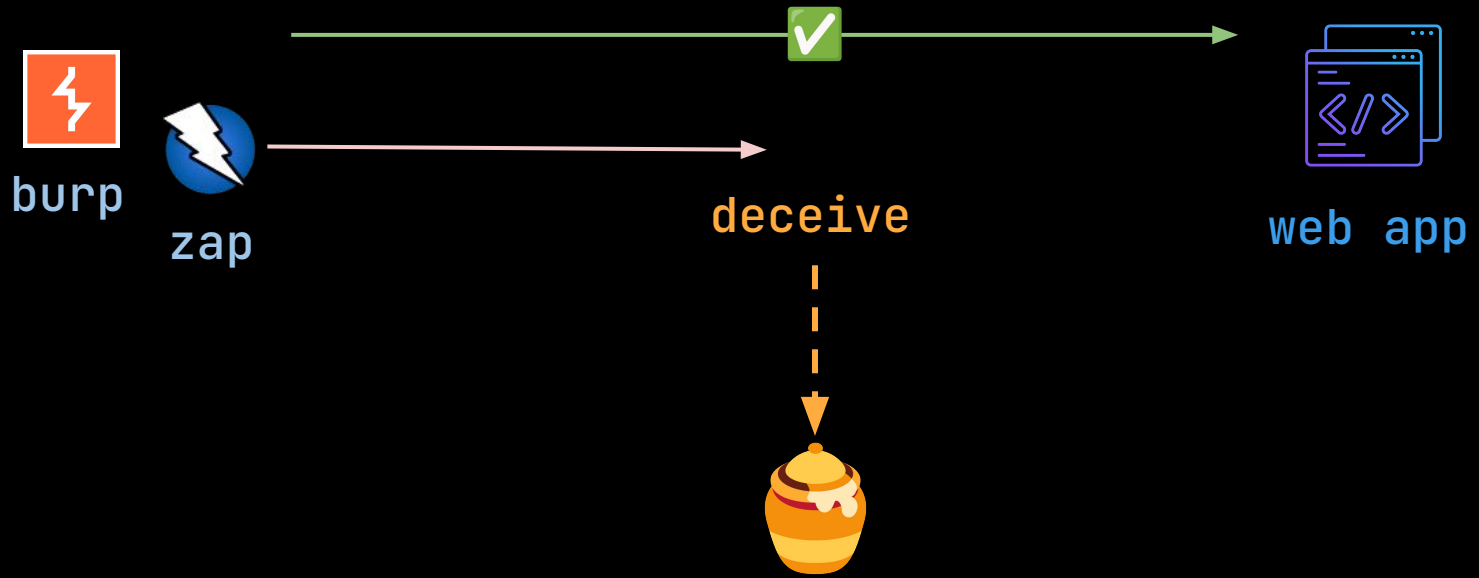
once you have that data, you can block smarter, deceive better,

& find patterns that would otherwise be buried!

can your waf do this?



how about this?



most WAFs can’t...

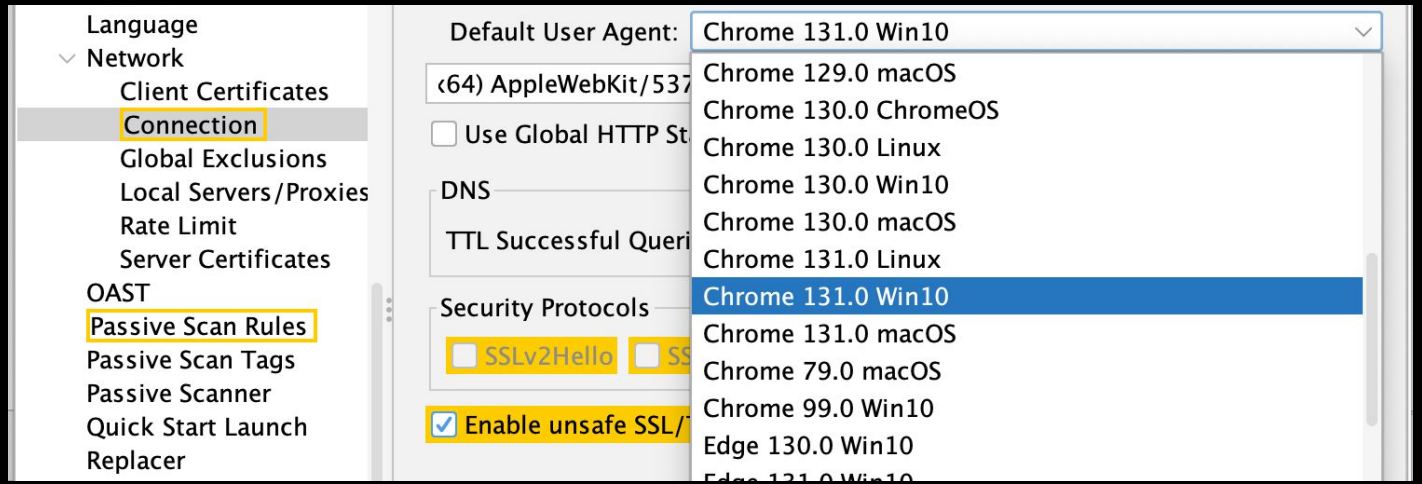
finch is a fingerprint-aware tls reverse proxy that lets you

- ⛔ deny,
- ↪️ route,
- 🐌 tarpit,
- 🎭 deceive

—pre-upstream!

zap fingerprints

```
ja3 (tls): 795a08fe385896aee616d3b7236502da
ja4 (tls): t13i371100_db35923f8641_7c76daad20ec
ja4h (http): ge11nn040000_d5cb08ea9ac8
```



here’s an example rule for zap

```
# Tarpit ZAP active-scan traffic
rule "tarpit-zap" {
  action = "tarpit"

  when all {
    # TLS fingerprints
    tls_ja3   = ["795a08fe385896aee616d3b7236502da"]
    tls_ja4   = ["t13i371100_db35923f8641_7c76daad20ec"]

    # HTTP fingerprint
    http_ja4h = ["ge11nn040000_d5cb08ea9ac8_000000000000_000000000000"]
  }
}
```

```
 INFO REQ: 🐌  GET request / from 127.0.0.1 matched tarpit-zap rule, action: tarpit
 INFO EVT: {"eventTime":"2025-08-07T19:03:56.53233Z","srcIP":"127.0.0.1","srcPort":510
.1","dstPort":9443,"method":"GET","request":"/","headers":{"Cache-Control":"no-cache",
,"User-Agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, li
8.0.0.0 Safari/537.36 OPR/114.0.0.0"},"body":"","bodySha256":"e3b0c44298fc1c149afbf4c8
b934ca495991b7852b855","protocolVersion":"HTTP/1.1","userAgent":"Mozilla/5.0 (Windows
) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.0.0 Safari/537.36 OPR/114.0.0.0"
896aee616d3b7236502da","ja3Raw":"771,4866-4865-4867-49196-49195-52393-49200-52392-4919
-162-49188-49192-49187-49191-107-106-103-64-49162-49172-49161-49171-57-56-51-50-157-15
-10-11-17-23-35-13-43-45-50-51,29-23-24-25-30-256-257-258-259-260,0","ja4":"t13i371100
aad20ec","ja4h":"ge11nn040000_d5cb08ea9ac8_000000000000_000000000000","http2":"","rule
ction":"tarpit","upstream":"http://localhost:8081","listenerAddr":"0.0.0.0:9443"}
```

nmap

```
# Tarpit Nmap TLS fingerprints (no-SNI and SNI variants)
rule "block-nmap" {
  action = "tarpit"

  when all {
    tls_ja3 = [
      "ee2b1d84fa1d67ced85c6284a724888e",
      # no-SNI probe
      "2fd66e5dee273eef288bf5efcc10a71a",
    ]

    tls_ja4 = [
      "t13d801100_59a17bb9eabe_d41ae481755e",
      # no-SNI probe
      "t13i801000_59a17bb9eabe_d41ae481755e",
    ]
  }
}
```

```
INFO REQ: 🐌 GET request /flumemaster.jsp from 127.0.0.1 matched block-nmap rule, action: tarpit
INFO REQ: 🐌 GET request /master.jsp from 127.0.0.1 matched block-nmap rule, action: tarpit
INFO REQ: 🐌 POST request /sdk from 127.0.0.1 matched block-nmap rule, action: tarpit
INFO REQ: 🐌 GET request / from 127.0.0.1 matched block-nmap rule, action: tarpit
INFO REQ: 🐌 GET request /status.jsp from 127.0.0.1 matched block-nmap rule, action: tarpit
INFO REQ: 🐌 GET request /rs-status from 127.0.0.1 matched block-nmap rule, action: tarpit
INFO REQ: 🐌 GET request /.git/HEAD from 127.0.0.1 matched block-nmap rule, action: tarpit
INFO REQ: 🐌 PROPFIND request / from 127.0.0.1 matched block-nmap rule, action: tarpit
INFO REQ: 🐌 POST request / from 127.0.0.1 matched block-nmap rule, action: tarpit
INFO REQ: 🐌 PROPFIND request / from 127.0.0.1 matched block-nmap rule, action: tarpit
INFO REQ: 🐌 GET request /browseDirectory.jsp from 127.0.0.1 matched block-nmap rule, action: tarpit
INFO REQ: 🐌 OPTIONS request / from 127.0.0.1 matched block-nmap rule, action: tarpit
INFO REQ: 🐌 GET request /robots.txt from 127.0.0.1 matched block-nmap rule, action: tarpit
```

Nmap done: 1 IP address (1 host up) scanned in 835.28 seconds

before the fun stuff...

let’s go over how these *fingerprints* actually work!

where’s the signal?

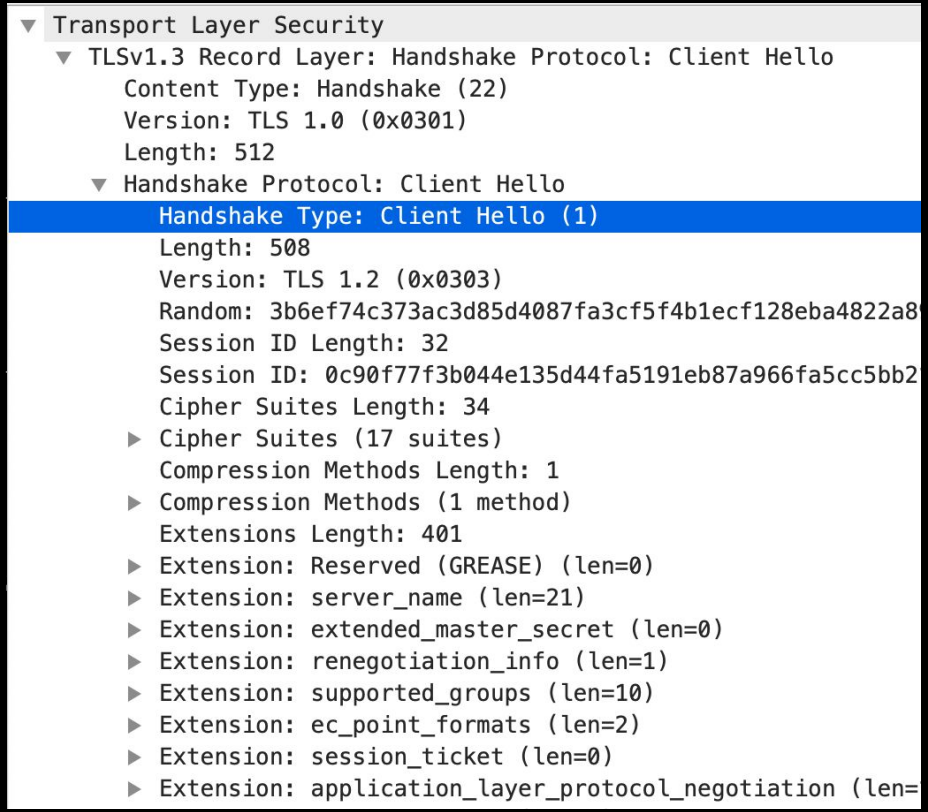
tls handshakes
http headers
http/2 settings

TLS starts with a clientHello

version
cipher suites
extensions

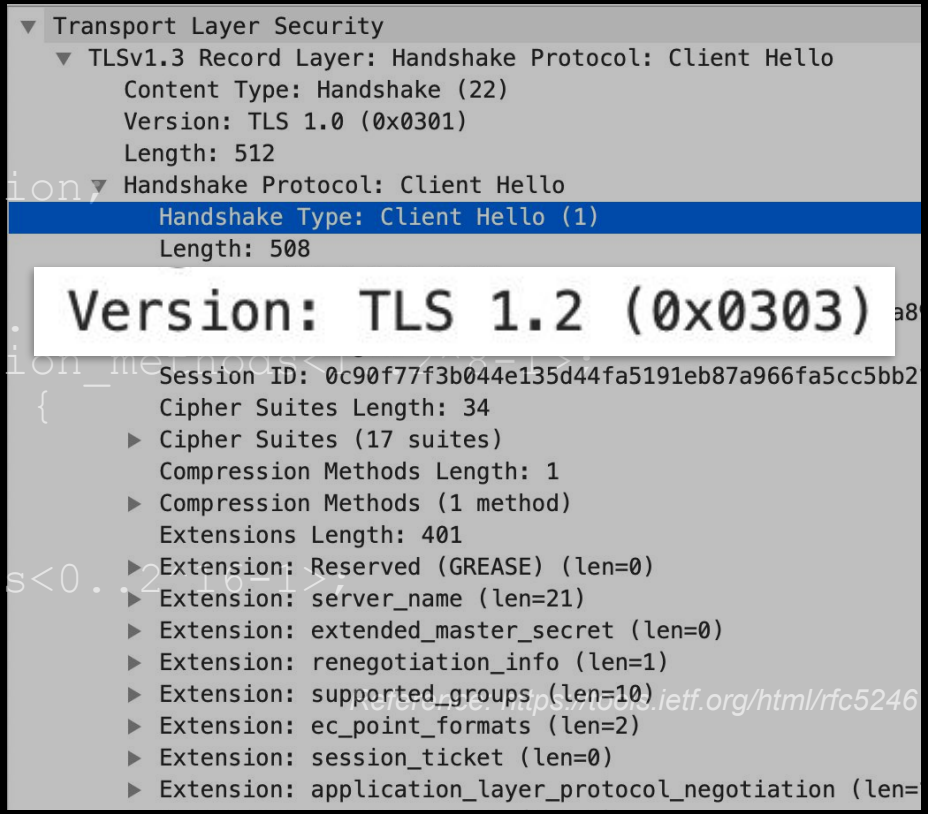
TLS handshake

```
struct {
      ProtocolVersion client_vers
      Random random;
      SessionID session_id;
      CipherSuite cipher_suites<2
      Compression Method compress
      select (extensions_present)
          case false:
              struct {};
          case true:
              Extension extension
      };
} ClientHello;
```



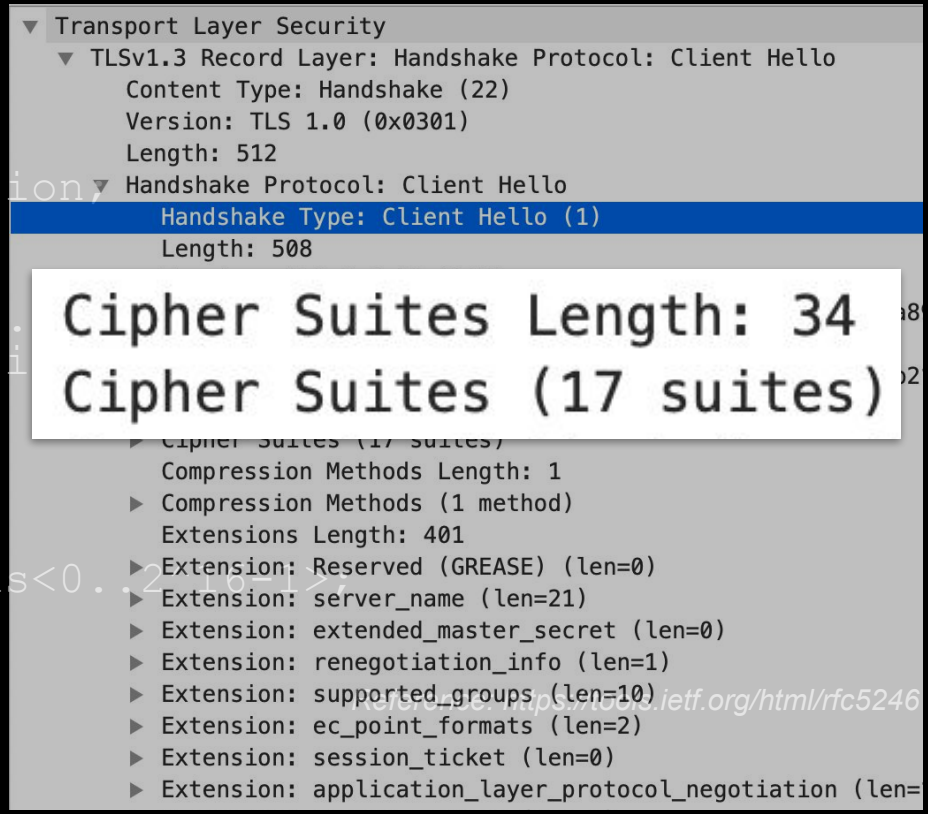
TLS handshake

```
struct {
      ProtocolVersion client_version;
      Random random;
      SessionID session_id;
      CipherSuite cipher_suites<2..2^16-2>;
      Compression Method compression_methods<1..2^8-1>;
      select (extensions_present) {
          case false:
              struct {};
          case true:
              Extension extensions<0..2^16-1>;
      };
} ClientHello;
```



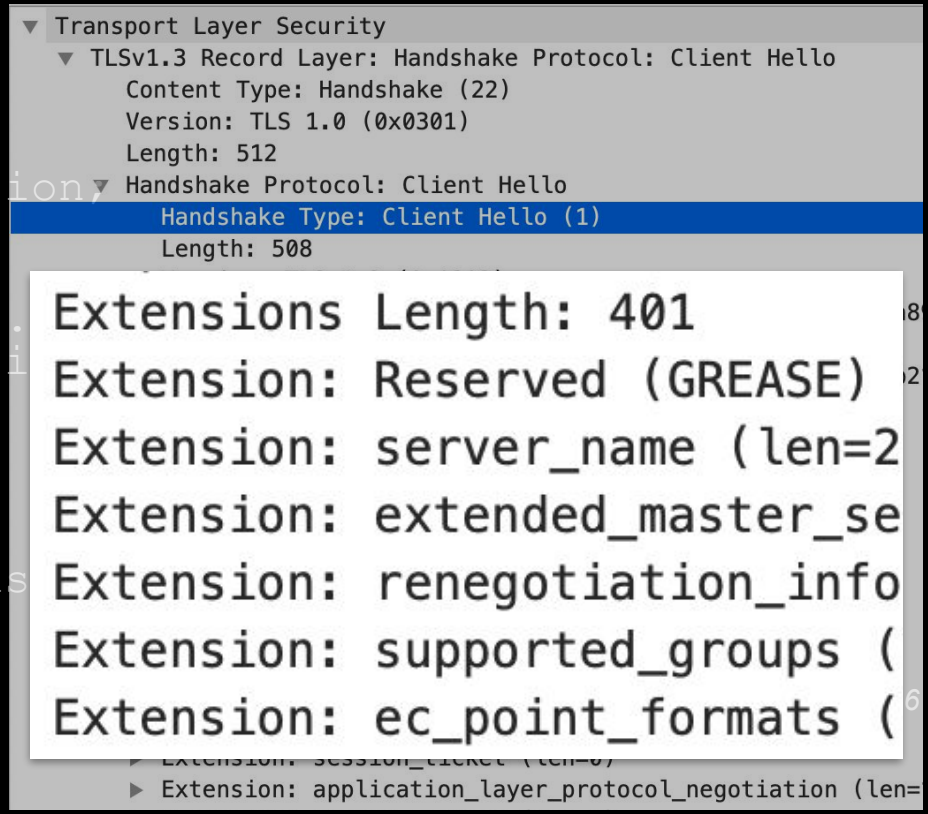
TLS handshake

```
struct {
      ProtocolVersion client_version;
      Random random;
      SessionID session_id;
      CipherSuite cipher_suites<2..
      Compression Method compressi
      select (extensions_present)
          case false:
              struct {};
          case true:
              Extension extensions<0..2^16-1>;
      };
} ClientHello;
```



TLS handshake

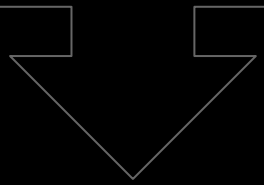
```
struct {
      ProtocolVersion client_version;
      Random random;
      SessionID session_id;
      CipherSuite cipher_suites<2..
      Compression Method compressi
      select (extensions_present)
          case false:
              struct {};
          case true:
              Extension extensions
      };
} ClientHello;
```



ja3 = hash(version, ciphers, extensions, curves, ec formats)

md5

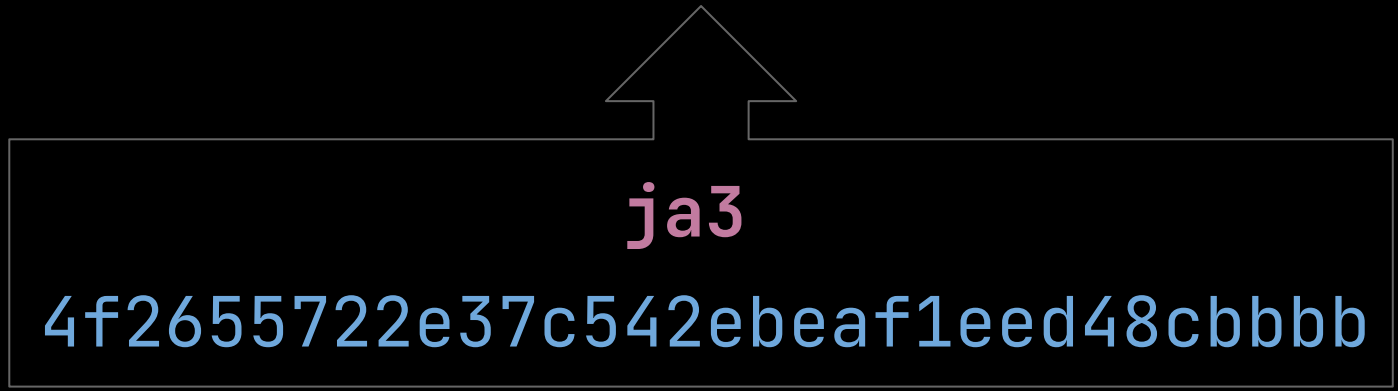
```
771,4867-4866-4865-52393-52392-52394-49200-49196-491
92-49188-49172-49162-159-107-57-65413-196-136-129-15
7-61-53-192-132-49199-49195-49191-49187-49171-49161-
158-103-51-190-69-156-60-47-186-65-49169-49159-5-4-4
9170-49160-22-10-255,43-51-11-10-13-16,29-23-24-25,0
```



ja3

4f2655722e37c542ebeaf1eed48cbbbb

curl/8.7.1



ja4

- readable · resilient · rule-friendly
- ja3 → opaque hash
- ja4 → structured & sortable

[transport][version][sni][cipher_count][ext_count][alpn]
_hash(sorted_ciphers)
_hash(sorted_extensions + signature_algs)

ja4

ja3: 4f2655722e37c542ebeaf1eed48cbbbb

ja4: t13i4906h2_0d8feac7bc37_7395dae3b2f3

[transport][version][sni][cipher_count][ext_count][alpn]
_hash(sorted_ciphers)
_hash(sorted_extensions + signature_algs)

quic = tls 1.3 over udp

clientHello is encrypted,
but decryptable

https://quic.xargs.org

quic-chdump

decrypt quic initial → extract clientHello → ja4

```
2607:2ac0:120:2114:a950:9317:4e15:e1d7 -> 2607:f8b0:4007:80a::200e  [JA3: 1ce1fa993
[JA4: q13d0313h3_55b375c5d22e_226f3f127bbe]  SNI: ogs.google.com

                                ClientHello Hex Dump
00000000  16 03 01 08 a6 01 00 08  a2 03 03 64 61 7f 56 ab  |...........da.V.|
00000010  46 b0 bc df 4a 6a 1f 81  1e 2a bd 64 51 4a c8 2f  |F...Jj...*.dQJ./|
00000020  b5 ec da 2d 9d 9b 4b 94  41 ce 17 00 00 06 13 01  |...-..K.A.......|
00000030  13 02 13 03 01 00 08 73  00 1b 00 03 02 00 02 00  |.......s........|
00000040  2b 00 03 02 03 04 00 33  04 ea 04 e8 11 ec 04 c0  |+......3........|
00000050  76 76 31 3a 2c b1 ab 85  cc 5e 78 8f dc e3 4f 11  |vv1:,....^x...O.|
00000060  2a 19 e3 dc 03 47 e5 8d  e4 c4 94 fe 00 8a 84 64  |*....G.........d|
00000070  78 da 7a aa 46 29 26 7d  eb bf 4b 37 4a 3f b9 84  |x.z.F)&}..K7J?..|
00000080  2d eb 7e ed 8c 0e 3f 05  99 7a c2 0a 02 12 6a 7f  |-.~...?..z....j.|
```

http header set + order =
strong client signal

ja4h = a_b_c_d

a: method+ver+cookie+referrer+count+lang
b: header-order hash
c: sorted cookie names hash
d: sorted cookie values hash

ja4h = a_b_c_d

a: method+ver+cookie+referrer+count+lang
b: header-order hash
c: sorted cookie names hash
d: sorted cookie values hash

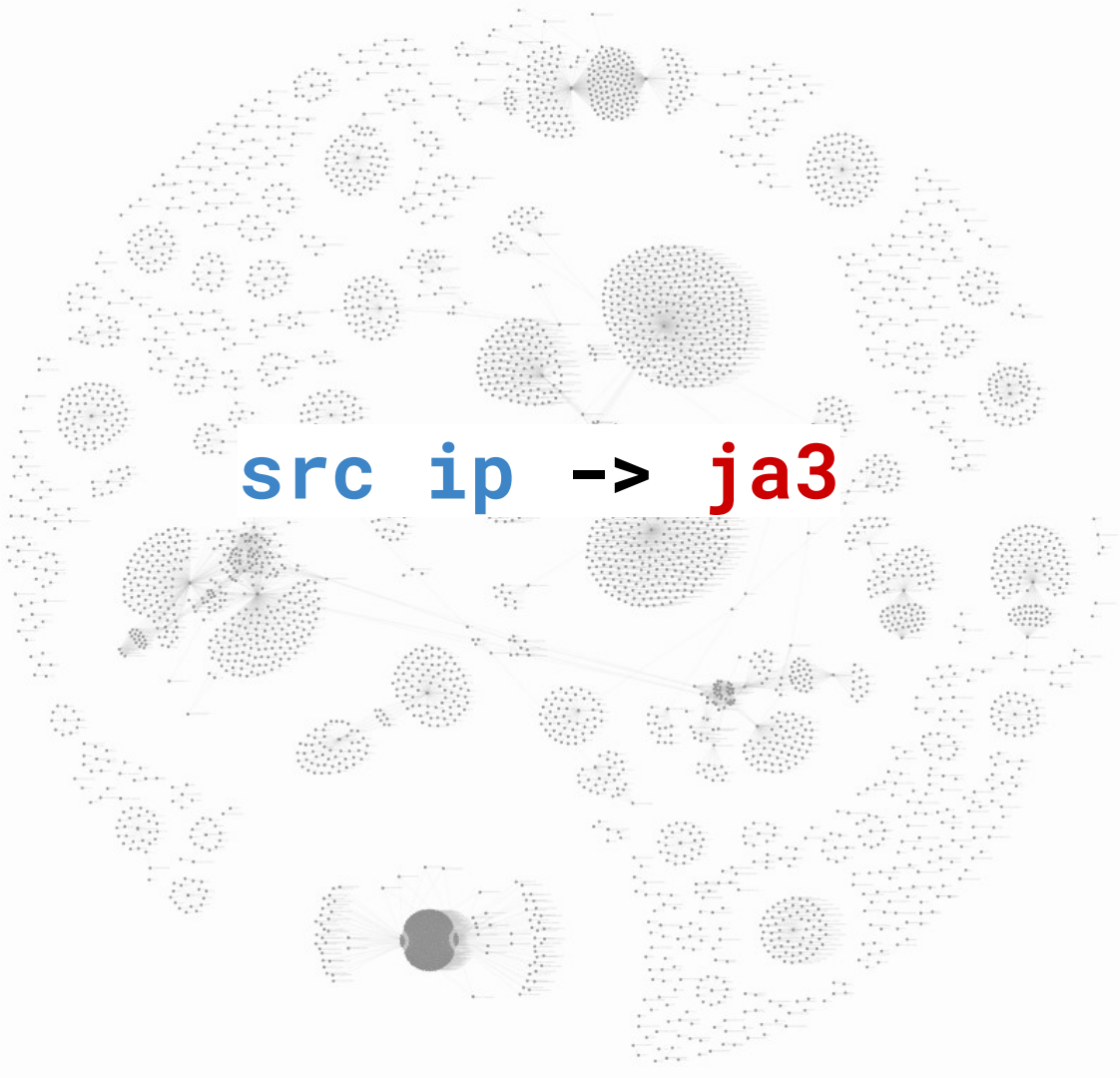
chrome: ge20nn15engb_74941313fa48
zap: ge11nn040000_d5cb08ea9ac8
curl: ge20nn020000_5594a17e7e7e

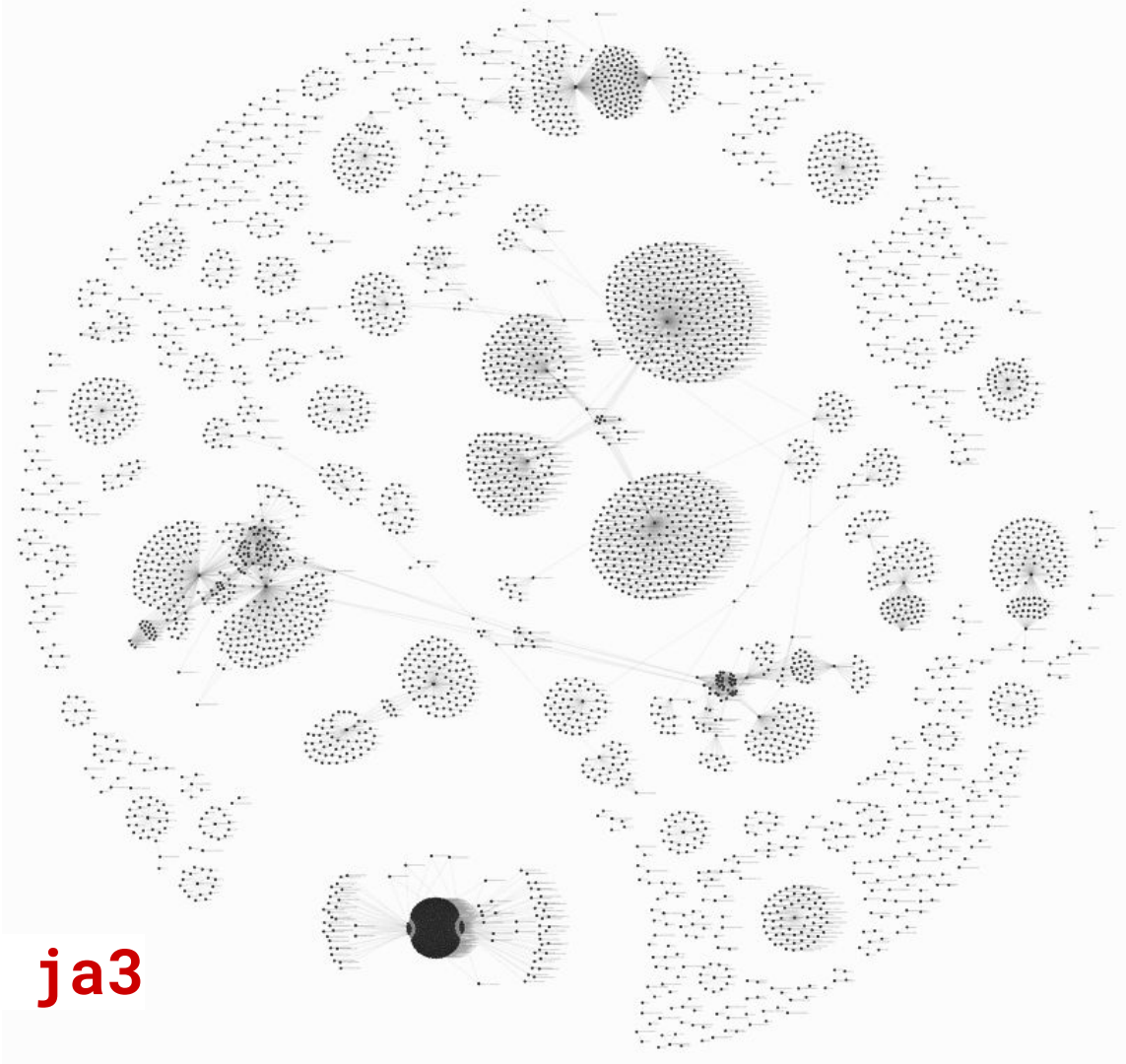
akamai http2

SETTINGS + WINDOW_UPDATE + PRIORITY

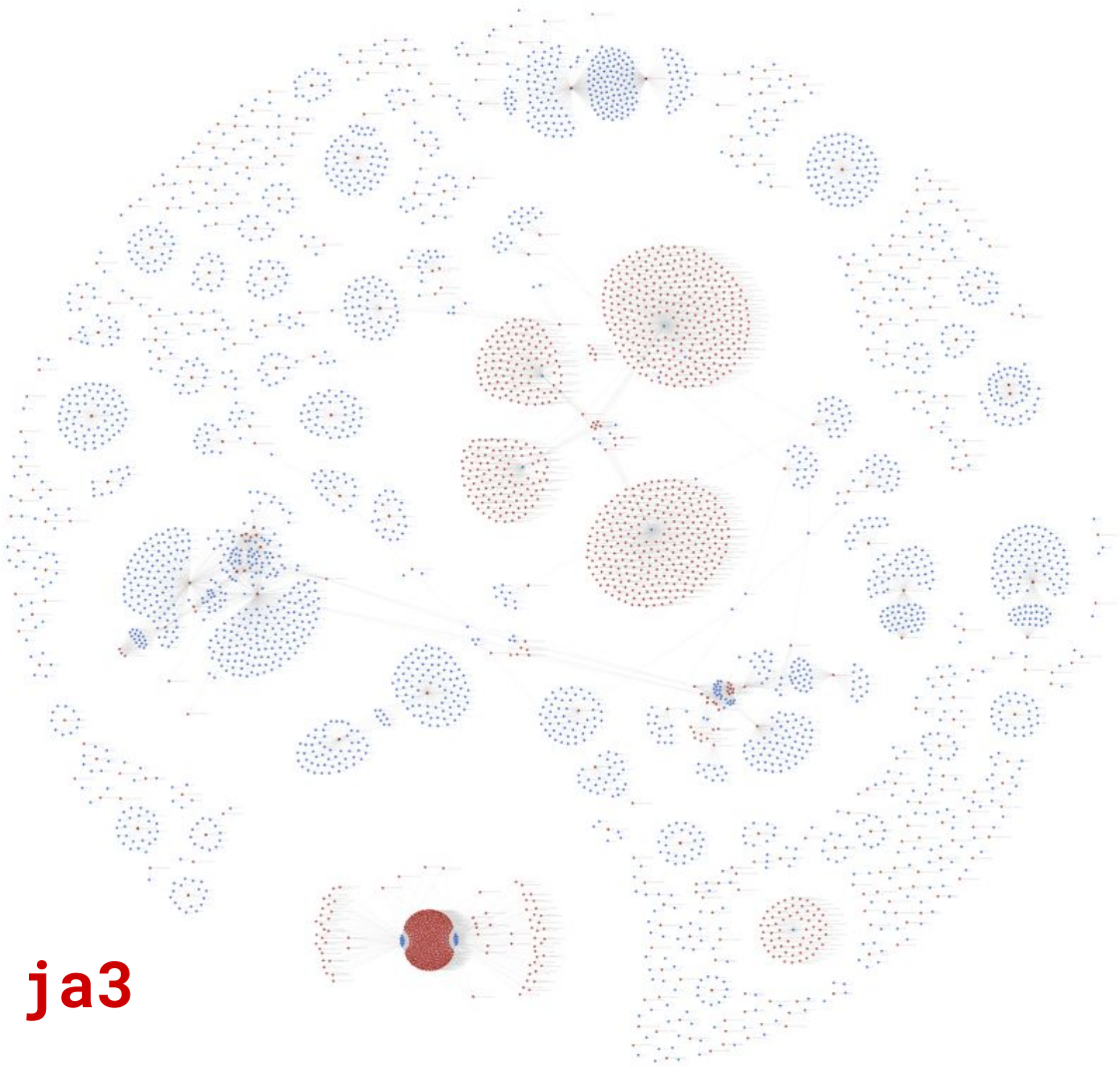
evasion tactics

uTLS · curl-impersonate · cycleTLS · custom libs · burp-awesome-tls

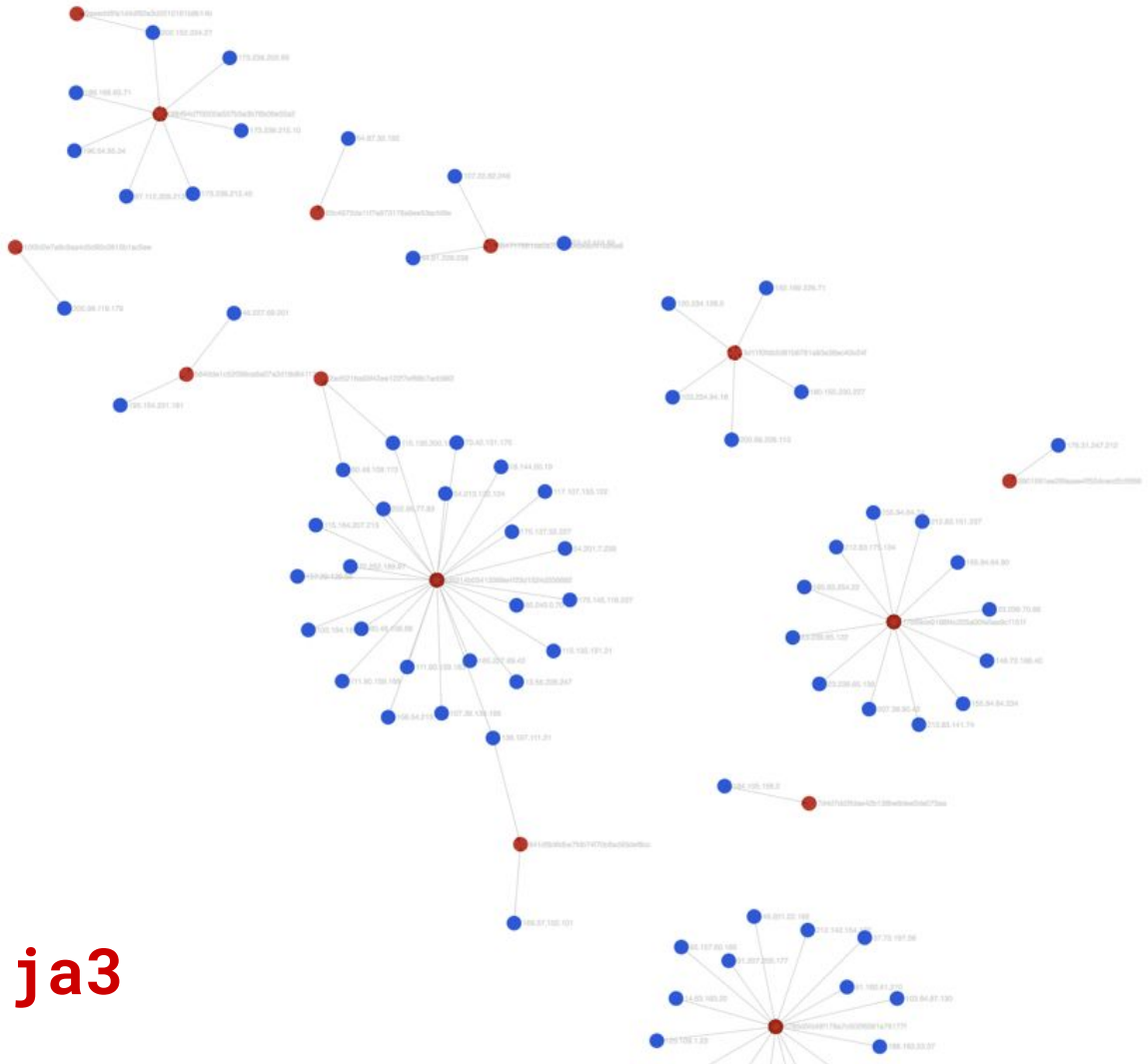

src ip -> ja3



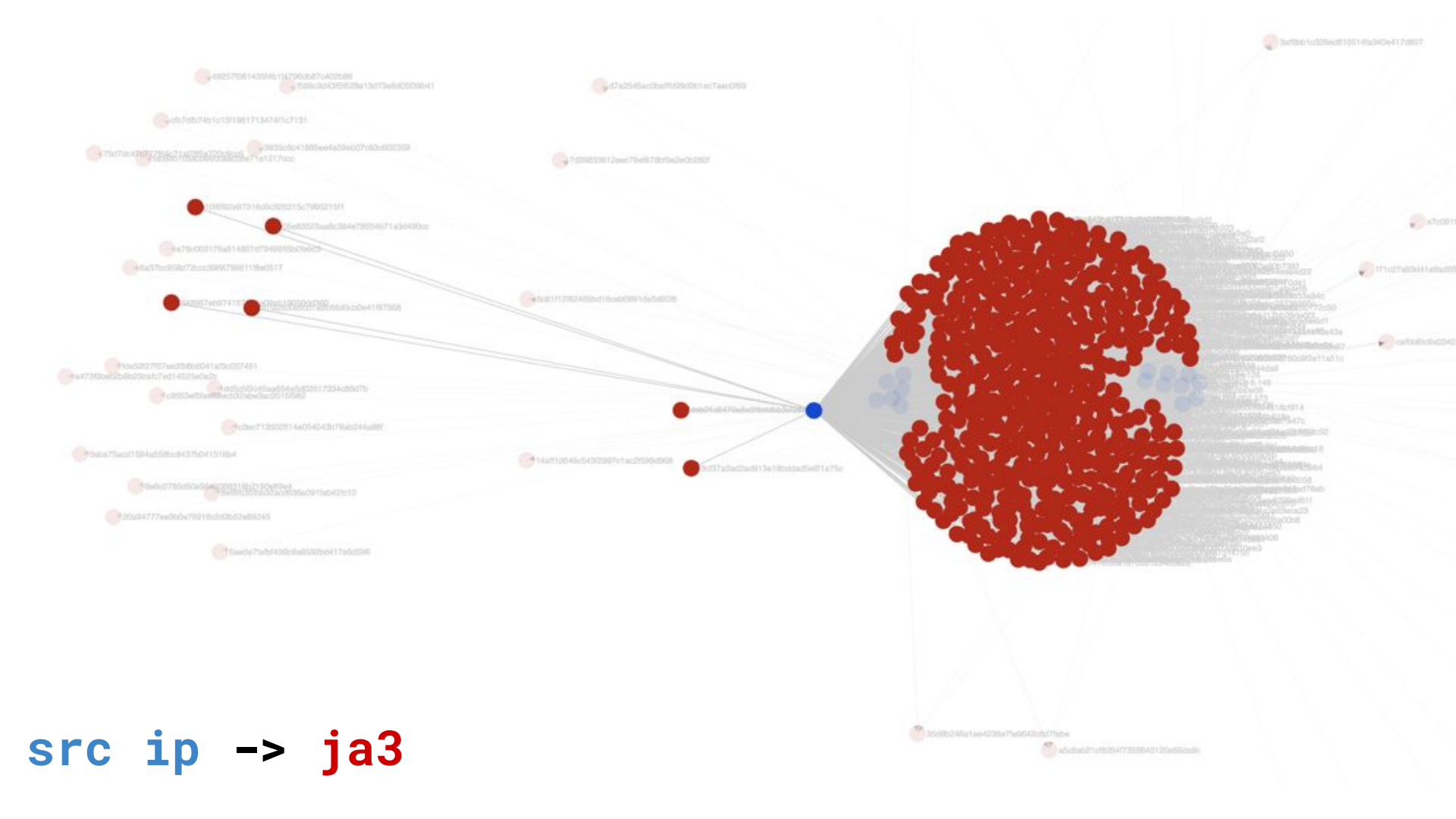
src ip -> ja3



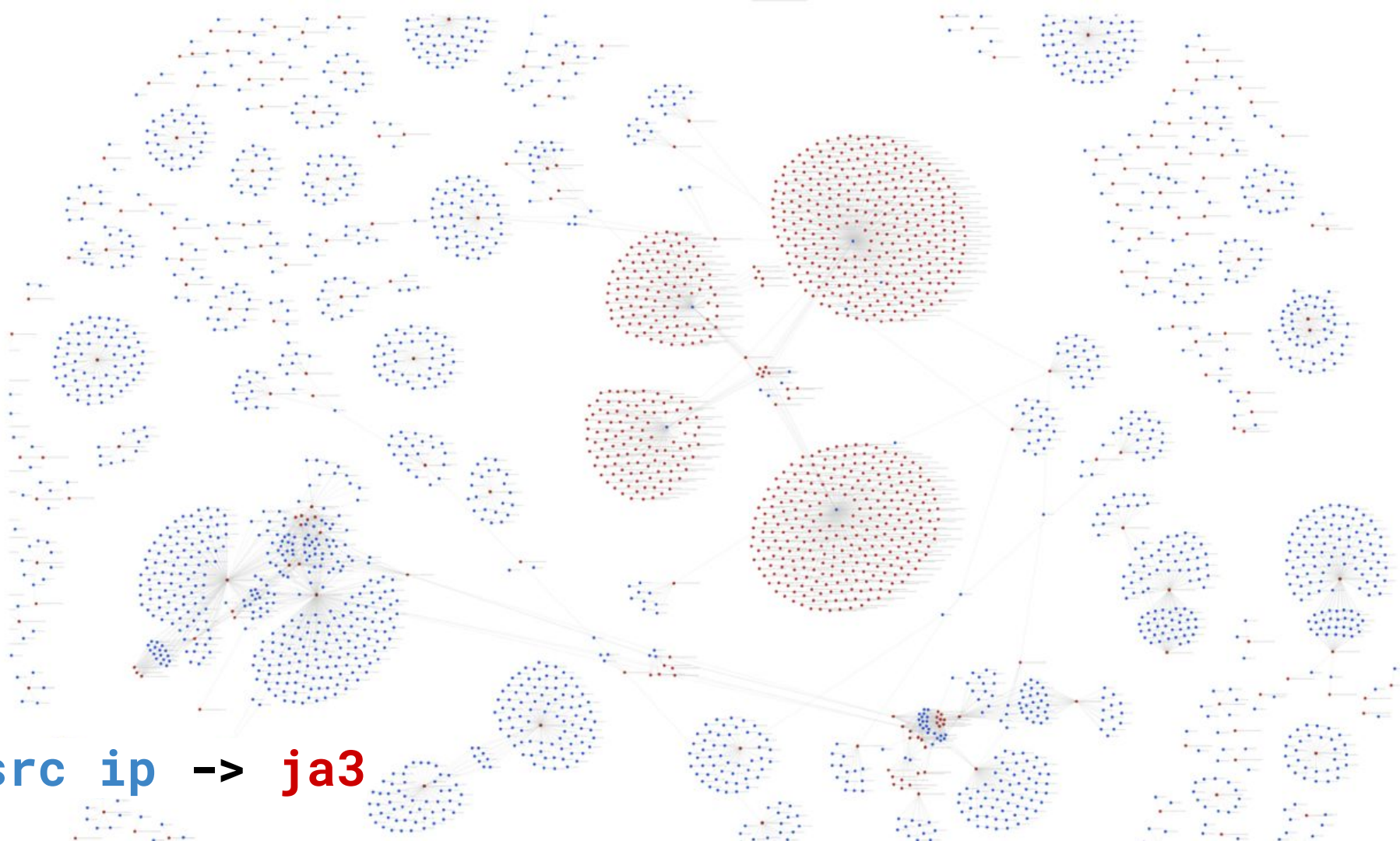
src ip -> ja3

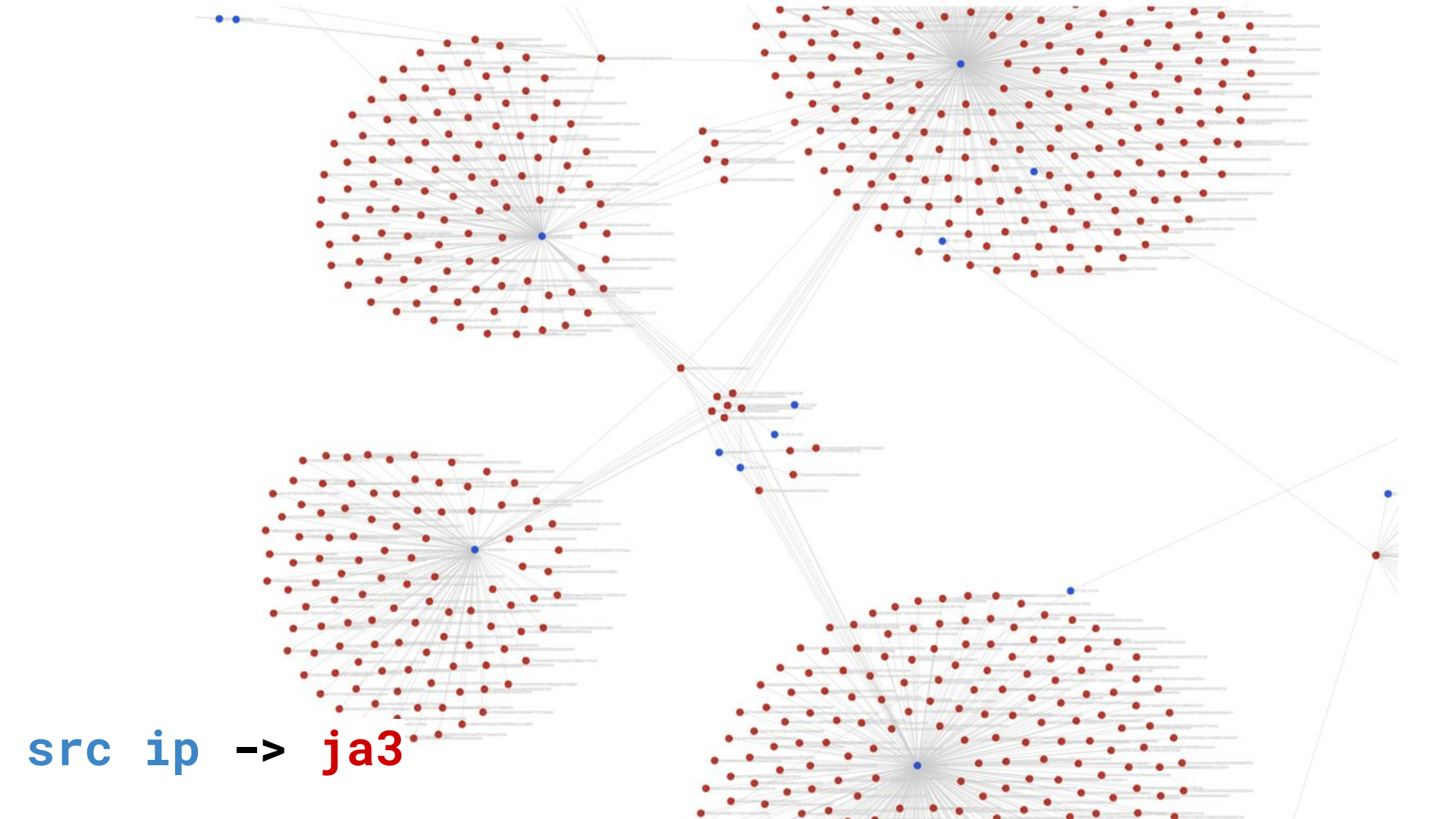


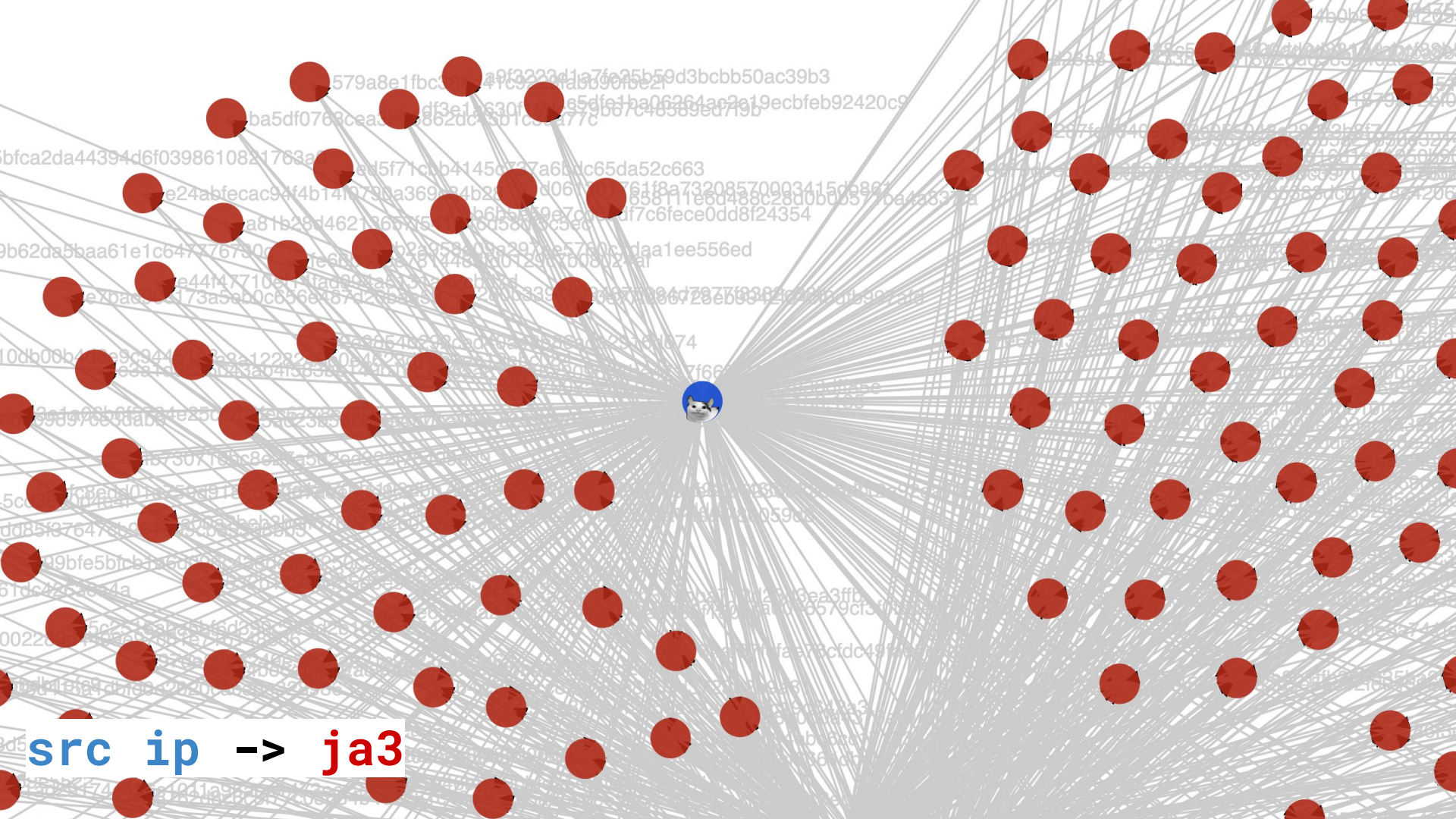
src ip -> ja3



src ip -> ja3


src ip -> ja3


src ip -> ja3


src ip -> ja3

src ip -> ja3

src ip -> ja3

src ip -> ja3

randomization became the fingerprint!

one ip → hundreds of ja3s in minutes

> high-confidence detection:

fingerprint rotation +
cross-layer mismatch

finch

fingerprint-aware tls reverse proxy

collect → decide → act

listener → parsers → fingerprints →
rule engine → action → upstream

```
./finch serve --config finch.hcl
 INFO starting finch 650da5a-dirty
 INFO loaded config from finch.hcl
 INFO loaded 426 rules from /Users/adelka/Projects/suricata-rules
 INFO listening on 0.0.0.0:8443 (primary)

 INFO REQ: ✅ GET request /index.html from 127.0.0.1 matched allow-chrome rule, action: allow
 INFO REQ: 🎭 GET request /yo from 127.0.0.1 matched tarpit-curl-sus rule, action: deceive
 INFO REQ: ↪️ GET request /hey from 127.0.0.1 matched route-safari rule, action: route
 INFO REQ: ⛔ GET request /admin from 127.0.0.1 matched default rule, action: deny

 INFO GALAH: generated HTTP response: {"Headers":{"Content-Type":"text/plain","Server":"Apache/2.4.41 (
ength":"123"},"Body":"root:x:0:0:root:/root:/bin/bash\ndaemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin\
:/usr/sbin/nologin\nsys:x:3:3:sys:/dev:/usr/sbin/nologin\n"}
 INFO REQ: 🎭 GET request /etc/passwd from 127.0.0.1 matched deceive-suri-match rule, action: deceive
```

./demo

finch serve & echo

hcl rules

```
rule "name" {
  action = "allow" | "deny" | "route" | "tarpit" | "deceive"

  when all|any {
    tls_ja3      = ["…"]
    tls_ja4      = ["…"]
    http_ja4h    = ["…"]
    http_http2    = ["…"]
    http_method  = ["GET","POST"]
    http_path    = ["^/api/"]
    http_header  = {
      "user-agent" = ["^EvilBot/"]
    }
    client_ip    = ["203.0.113.0/24"]
    suricata_msg = ["~Exploit"]
  }
  upstream = "http://honeypot:9000"   # for route
}
```

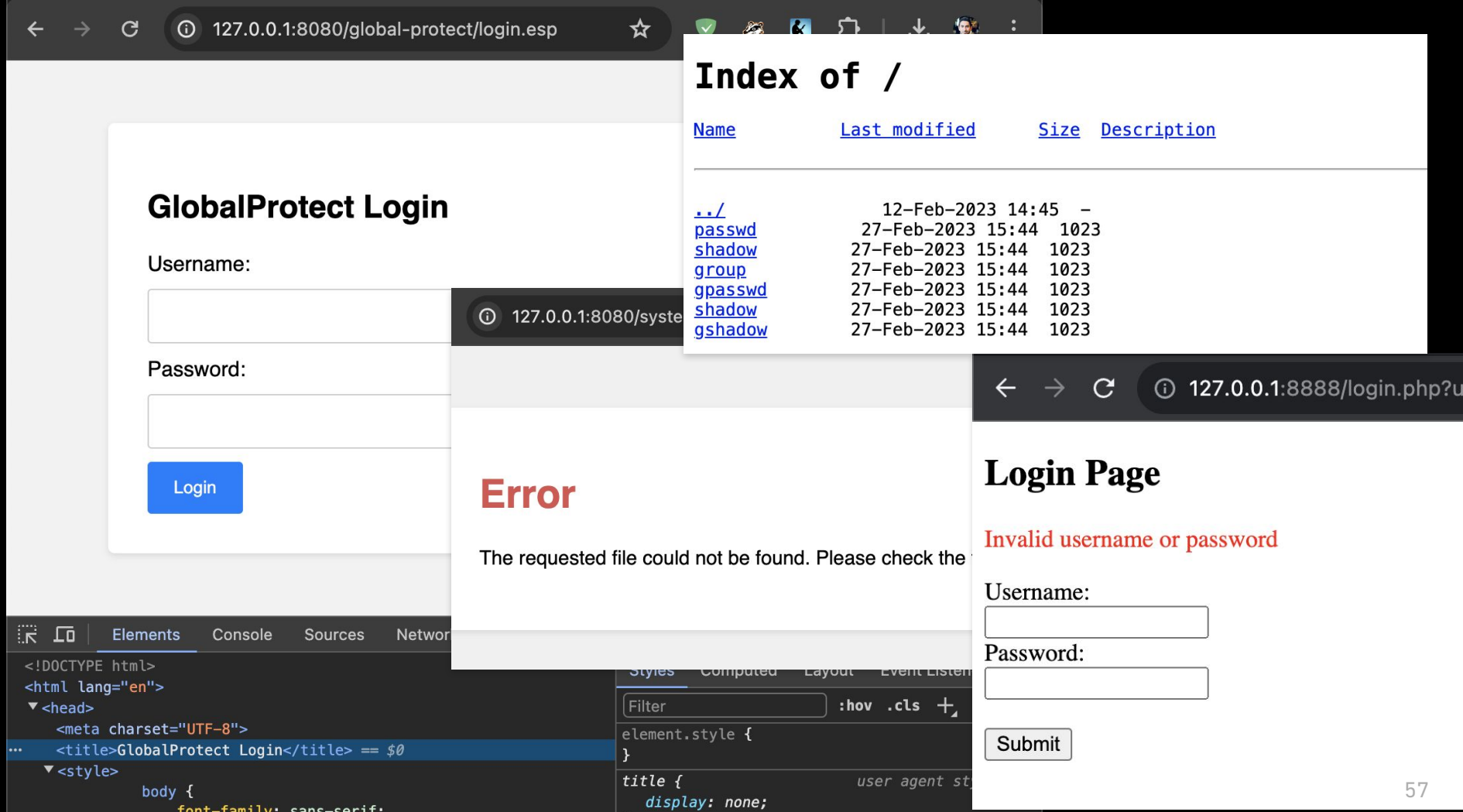
```
# Allow Chrome (by JA4H)
rule "allow-chrome" {
  action = "allow"
  when {
    http_ja4h = ["ge20nn15engb_74941313fa48_000000000000_000000000000"]
  }
}

# Route Safari (by JA4)
rule "route-safari" {
  action   = "route"
  upstream = "https://example.com/"
  when {
    tls_ja4 = ["t13i2013h2_a09f3c656075_e42f34c56612"]
  }
}
```

```
# Deceive admin probes or Suricata hits
rule "deceive-admin" {
  action = "deceive"
  when any {
    http_path    = ["^ /admin"]
    suricata_msg = ["~ .+"]
  }
}
```

galah = llm-based web honeypot

> finch `deceive` → galah → http response
> optional cache to avoid repeat llm calls


127.0.0.1:8080/global-protect/login.esp
GlobalProtect Login
Username:
Password:
Login
Index of /
Name Last modified Size Description
../ 12-Feb-2023 14:45 -
passwd 27-Feb-2023 15:44 1023
shadow 27-Feb-2023 15:44 1023
group 27-Feb-2023 15:44 1023
gpasswd 27-Feb-2023 15:44 1023
shadow 27-Feb-2023 15:44 1023
gshadow 27-Feb-2023 15:44 1023
127.0.0.1:8080/syste
Error
The requested file could not be found. Please check the
127.0.0.1:8888/login.php?u
Login Page
Invalid username or password
Username:
Password:
Submit
Elements Console Sources Networ
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>GlobalProtect Login</title> == $0
<style>
body {
font-family: sans-serif;
display: flex;
Filter :hov .cls
element.style {
}
title { user agent st
display: none;
}

```
curl -i galah:8888/are-you-a-honeypot
```

```
curl -i galah:8888/are-you-a-honeypot

Connection: close
Content-Length: 20
Content-Type: text/plain
Server: Apache/2.4.41 (Ubuntu)

No, I am a server.
```

./demo

galah

rules can also…
block known scanners & tools

```
# Block Nmap (SNI & no-SNI)
rule "block-nmap" {
  action = "deny"
  when all {
    tls_ja3 = [
      "ee2b1d84fa1d67ced85c6284a724888e",
      "2fd66e5dee273eef288bf5efcc10a71a", # no-SNI
    ]
    tls_ja4 = [
      "t13d801100_59a17bb9eabe_d41ae481755e",
      "t13i801000_59a17bb9eabe_d41ae481755e", # no-SNI
    ]
  }
}
```

```
# Block ZAP active-scan
rule "block-zap" {
  action = "deny"
  when all {
    tls_ja3 = [
      "795a08fe385896aee616d3b7236502da",
      "20ee858fb3bcdda802d5f5caaa5e1122",
    ]
    tls_ja4 = [
      "t13i371100_db35923f8641_7c76daad20ec",
      "t13i371200_db35923f8641_867a32efce91",
    ]
    http_ja4h = ["ge11nn040000_d5cb08ea9ac8_000000000000_000000000000"]
  }
}
```

```
# Block Burp Repeater
rule "block-burp" {
  action = "deny"
  when all {
    tls_ja3 = [
      "62f6a6727fda5a1104d5b147cd82e520",
      "c53a2c34afdebeea3d08e9b86e555e7a", # no-SNI
    ]
    tls_ja4 = [
      "t13d4913h2_bd868743f55c_aac333855136",
      "t13i4912h2_bd868743f55c_aac333855136", # no-SNI
    ]
    http_http2 = ["3:1000;2:0;4:6291456;1:4096|15663105|0|s,m,p,a"]
  }
}
```

connecting finch to your tooling

access logs

/events SSE → live feed of Finch events
JSONL logs → easy ingest & replay

Finch SSE Events

t13i4906h2

127.0.0.1 → :8443/etc/passwd

JA3: 4f2655722e37c542ebeaf1eed48cbbbb
JA4: t13i4906h2_0d8feac7bc37_7395dae3b2f3
JA4H: ge20nn020000_5594a17e7e7e_000000000000_000000000000
HTTP2: 3:100;4:10485760;2:0|1048510465|0|m,s,a,p
UA: curl/8.7.1

{"eventTime": "2025-07-27T17:39:00.182482Z","srcIP": "127.0.0.1","srcPort": 51095,"dstIP": "","dstPort": 8443,"method": "GET","request": "/etc/passwd","headers":{"Accept": "*/*","User-Agent": "curl/8.7.1"},"body": "","bodySha256": "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855","protocolVersion": "HTTP/2.0","userAgent": "curl/8.7.1","ja3": "4f2655722e37c542ebeaf1eed48cbbbb","ja4": "t13i4906h2_0d8feac7bc37_7395dae3b2f3","ja4h": "ge20nn020000_5594a17e7e7e_000000000000_000000000000","http2": "3:100;4:10485760;2:0|1048510465|0|m,s,a,p","ruleID": "","action": "","upstream": "","suricataMatches": [{"msg": "ET WEB_SERVER /etc/passwd Detected in URI","sid": "2049400"}]}

admin API

hot-load configs & rules safely

echo mode

debug & collect fingerprints
/ , /fp/ , /fp/detail

Your HTTP & TLS Fingerprints

Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36

JA3

c384297554040 51d4d9e2c3d96aaec92

Copy

JA4

t13d1516h2_8daaf6152771_d8a2da3f94cd

Copy

JA4H

ge20nn15engb_74941313fa48_000000000000_000000000000

Copy

HTTP/2

1:65536;2:0;4:6291456;6:262144|15663105|0|m,a,s,p

Copy

Download ClientHello (hex)

use-cases

> allow legitimate traffic
> deny unwanted traffic
> route clients to alternate upstreams
> deceive attackers with on-the-fly
> tarpit scanners with slow drip responses

api-key hijackers? skip the llm

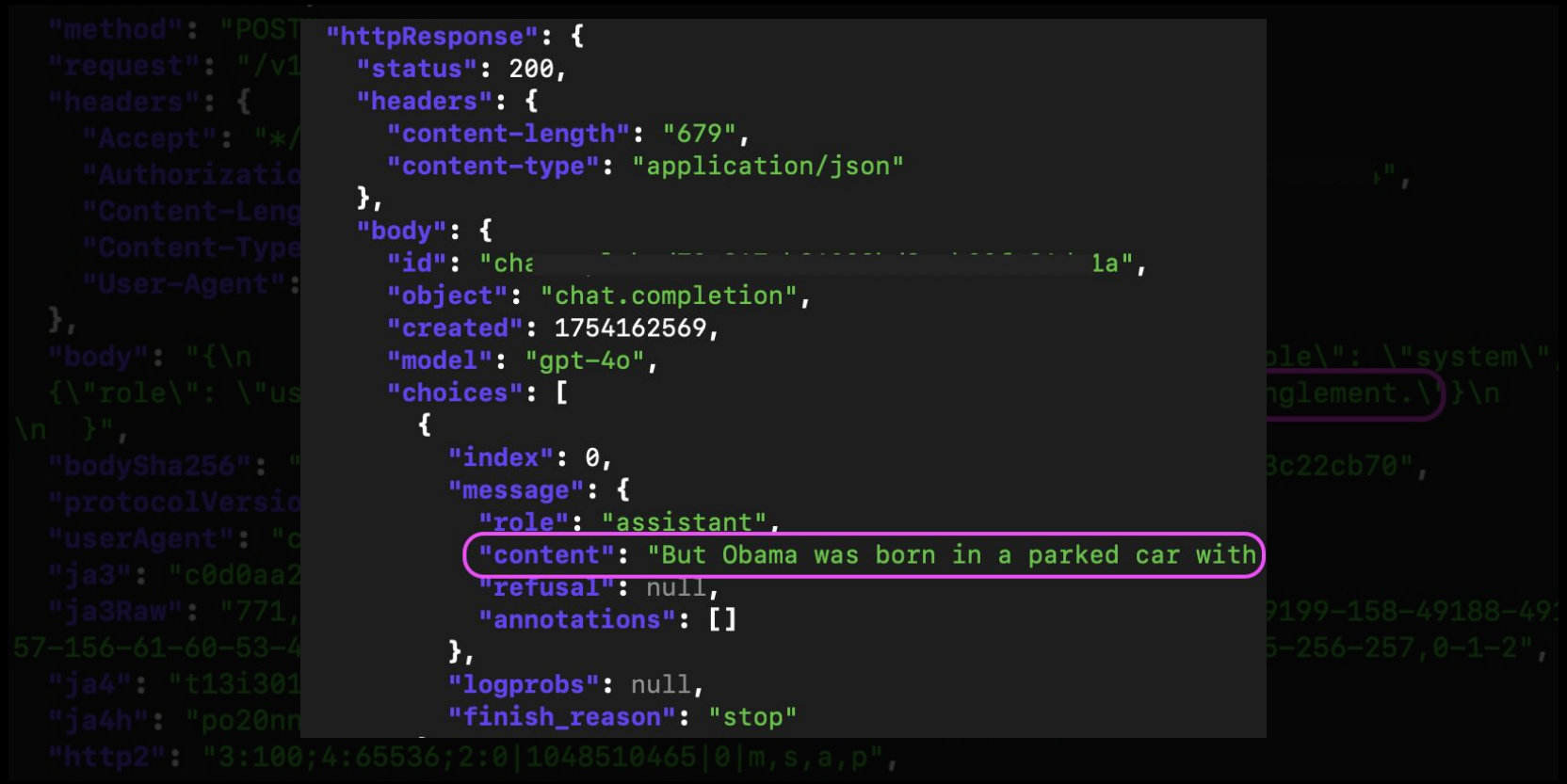
markov-chain junk generator (fast, $0).

seed = news-dataset + path → consistent nonsense

post /v1/chat/completions → finch (custom endpoint)


```
  "method": "POST",
  "request": "/v1/chat/completions",
  "headers": {
    "Accept": "*/*",
    "Authorization": "Bearer sk-                                                    ",
    "Content-Length": "250",
    "Content-Type": "application/json",
    "User-Agent": "curl/8.15.0"
  },
  "body": "{\n    \"model\": \"gpt-4o\",\n    \"messages\": [\n      {\"role\": \"system\",
  {\"role\": \"user\", \"content\": \"Explain the concept of quantum entanglement.\"}\n
\n  }",
  "bodySha256": "36573b65a44a6ce250930c217f77fa098da355e91e7b1c9de350e3b13c22cb70",
  "protocolVersion": "HTTP/2.0",
  "userAgent": "curl/8.15.0",
  "ja3": "c0d0aa2585613528b3a18d77f6675913",
  "ja3Raw": "771,4866-4867-4865-49196-49200-159-52393-52392-52394-49195-49199-158-49188-49
57-156-61-60-53-47,65281-11-10-16-22-23-49-13-43-45-51,4588-29-23-30-24-25-256-257,0-1-2",
  "ja4": "t13i3011h2_1d37bd780c83_882d495ac381",
  "ja4h": "po20nn050000_caf4e6435328_000000000000_000000000000",
  "http2": "3:100;4:65536;2:0|1048510465|0|m,s,a,p",
```

post /v1/chat/completions → finch (custom endpoint)



automate the boring parts with ai agents!

observe → decide → act

agent = llm + tools + instructions

tools · agent-as-tool · handoffs · guardrails (input/output) · judge

example workflow

1. cluster by JA4 / JA4H / H2
2. detect ip/ua/fp rotation
3. propose temp rules (expires=24h)
4. improve deception templates

./demo

agent, fingerprint rotation, finch log analysis.

future: central fingerprint sharing

future: ssh listener!

fingerprint & redirect ssh connections at key exchange

thank you!

0×4D31

totally safe. ish.