Inside the Magic

A Merlin Walkthrough

WHOAMI

- Russel Van Tuyl
- Twitter: @Ne0nd0g / @merlin_c2
- https://www.github.com/Ne0nd0g
- https://www.medium.com/@Ne0nd0g
- Slack: https://bloodhoundgang.herokuapp.com
  - #merlin

Agenda

- Introduction to Merlin
- HTTP Versions
- Merlin Application Concepts
  - Message Types
  - OPAQUE Key Exchange Protocol
  - JSON Web Encryption payloads
  - JSON Web Tokens Authentication
- Merlin Server
  - Main / Listeners / Agents / Modules
- Merlin Agent
  - Customization / Domain Fronting / Evasion / JA3

MERLIN

- A cross-platform post-exploitation Command & Control (C2)
  - 10 different operating systems (i.e. android, dragonfly, or Solaris)
  - 9 different architectures (i.e. arm64, mips64, or ppc64)
- Written in Go programming language
- HTTP/1.1, HTTP/2, and HTTP/3 protocols
- Server Component
- Agent Component
- Documentation: https://merlin-c2.readthedocs.io/en/latest/index.html

HTTP Versions - Overview

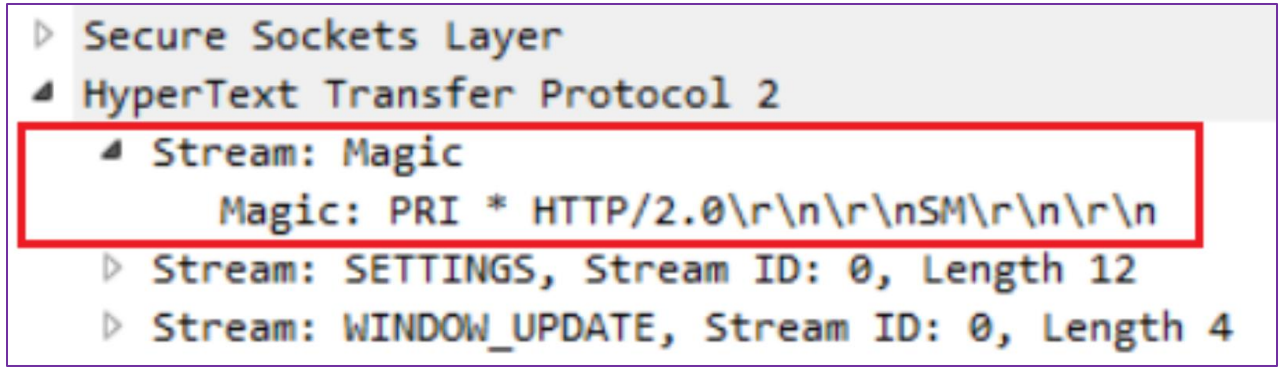
- Hypertext Transfer Protocol (HTTP)
  - 1990 – version 0.9
  - 1996 – version 1.0 RFC 1945
  - 1997 – version 1.1 RFC 2068
  - 2015 – version 2.0 RFC 7540
    - Google's SPDY
  - 2018 – version 3.0 IETF Draft
    - Quick UDP Internet Connections (QUIC) – IETF Draft

HTTP/2

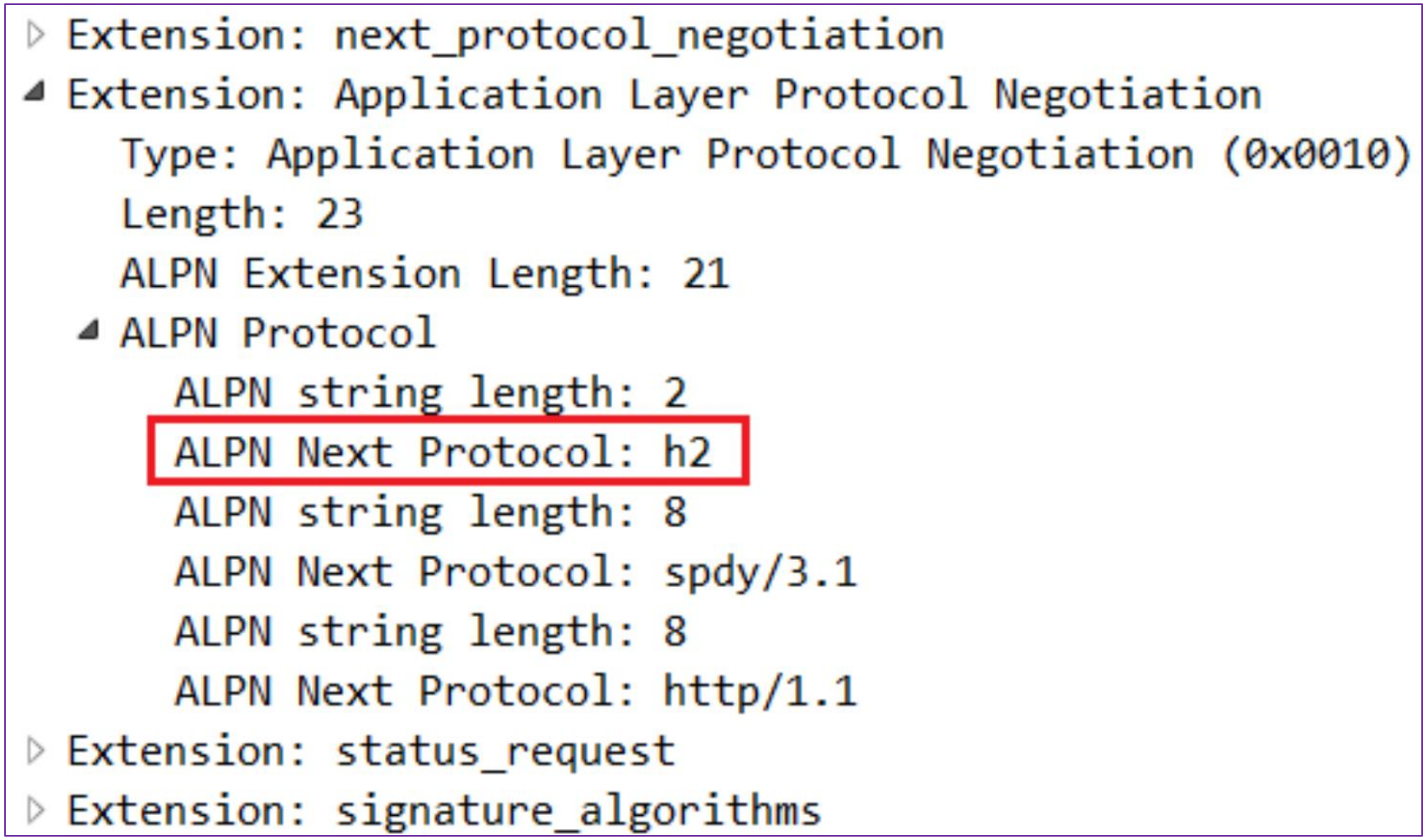
- Binary Protocol
- Multiplexed
- Bidirectional
- Ephemeral and Perfect Forward Secrecy (PFS) Cipher Suites
- PUSH
- Clear-Text version denoted by h2c
- No Prior Knowledge
  - **HTTP**: Upgrade Header
  - **HTTPS**: TLS v1.2+ Application-Layer Protocol Negotiation (ALPN)
- Prior Knowledge
  - Alt-Svc Header

Upgrade Header

```
GET / HTTP/1.1
Host: 127.0.0.1
Connection: Upgrade, HTTP2-Settings
Upgrade: h2c
HTTP2-Settings: AAMAAABkAAQAAP__
Accept: */*
User-Agent: nghttp2/1.3.4

HTTP/1.1 101 Switching Protocols
Upgrade: h2c
Connection: Upgrade

...............d.............PRI * HTTP/2.0

SM
```



TLS ALPN



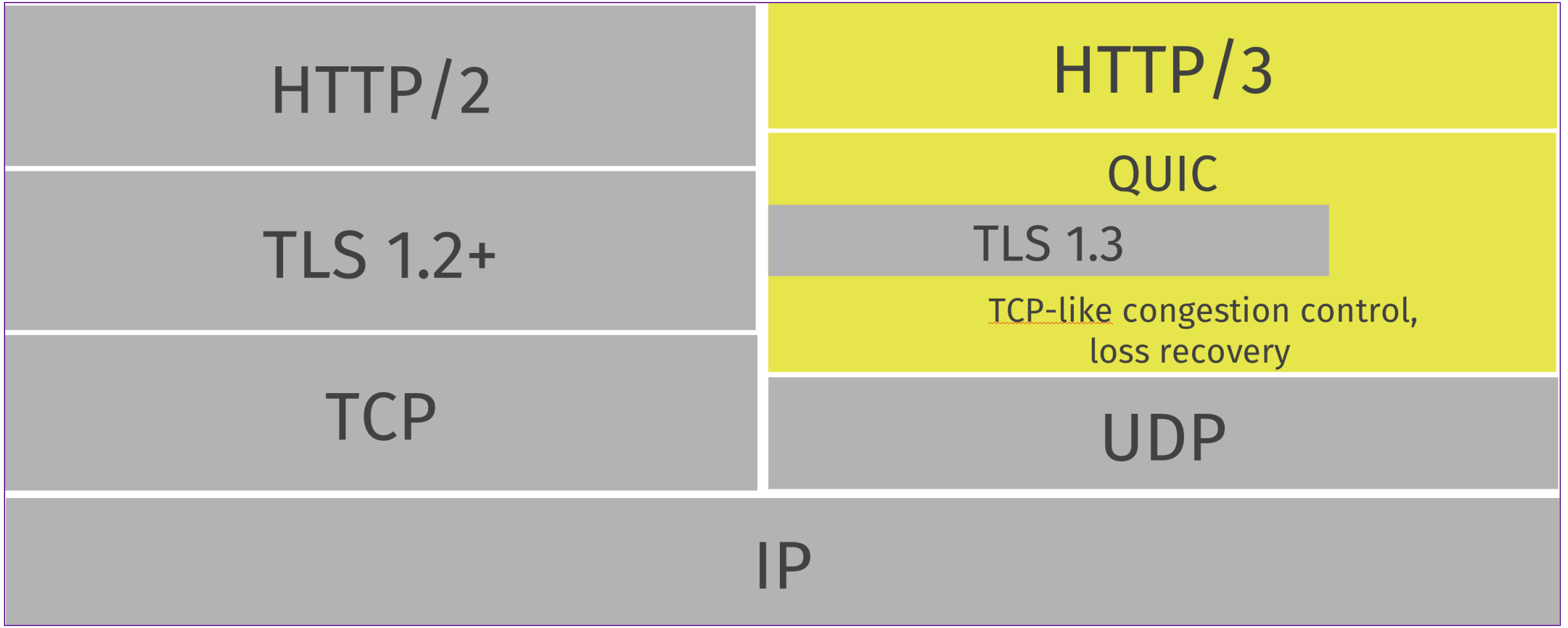
Alt-Svc Header

```
1  HTTP/1.1 200 OK
2  Server: nginx
3  Date: Sat, 31 Oct 2020 14:33:30 GMT
4  Content-Type: text/html; charset=UTF-8
5  Content-Length: 156922
6  Connection: close
7  X-Powered-By: PHP/7.4.12
8  X-Frame-Options: SAMEORIGIN
9  Link: <https://www.hatthieves.es/wp-json/>; rel="https://api.w.org/"
10 Link: <https://www.hatthieves.es/wp-json/wp/v2/pages/1187>; rel="alternate";
    type="application/json"
11 Link: <https://www.hatthieves.es/>; rel=shortlink
12 Vary: Accept-Encoding
13 Alt-Svc: h3-25=":443"; ma=3600, h2=":443"; ma=3600
14 X-XSS-Protection: 1; mode=block
15 X-Permitted-Cross-Domain-Policies: none
16 X-Frame-Options: SAMEORIGIN
17 Content-Security-Policy: frame-ancestors 'self' hatthieves.es
   *.hatthieves.es;
18 X-Content-Type-Options: nosniff
19 Referrer-Policy: same-origin
20 X-Download-Options: noopen
21 Strict-Transport-Security: max-age=31536000; includeSubDomains
22
23 <!DOCTYPE html>
24 <html lang="es" class="no-js no-svg">
```

HTTP/3

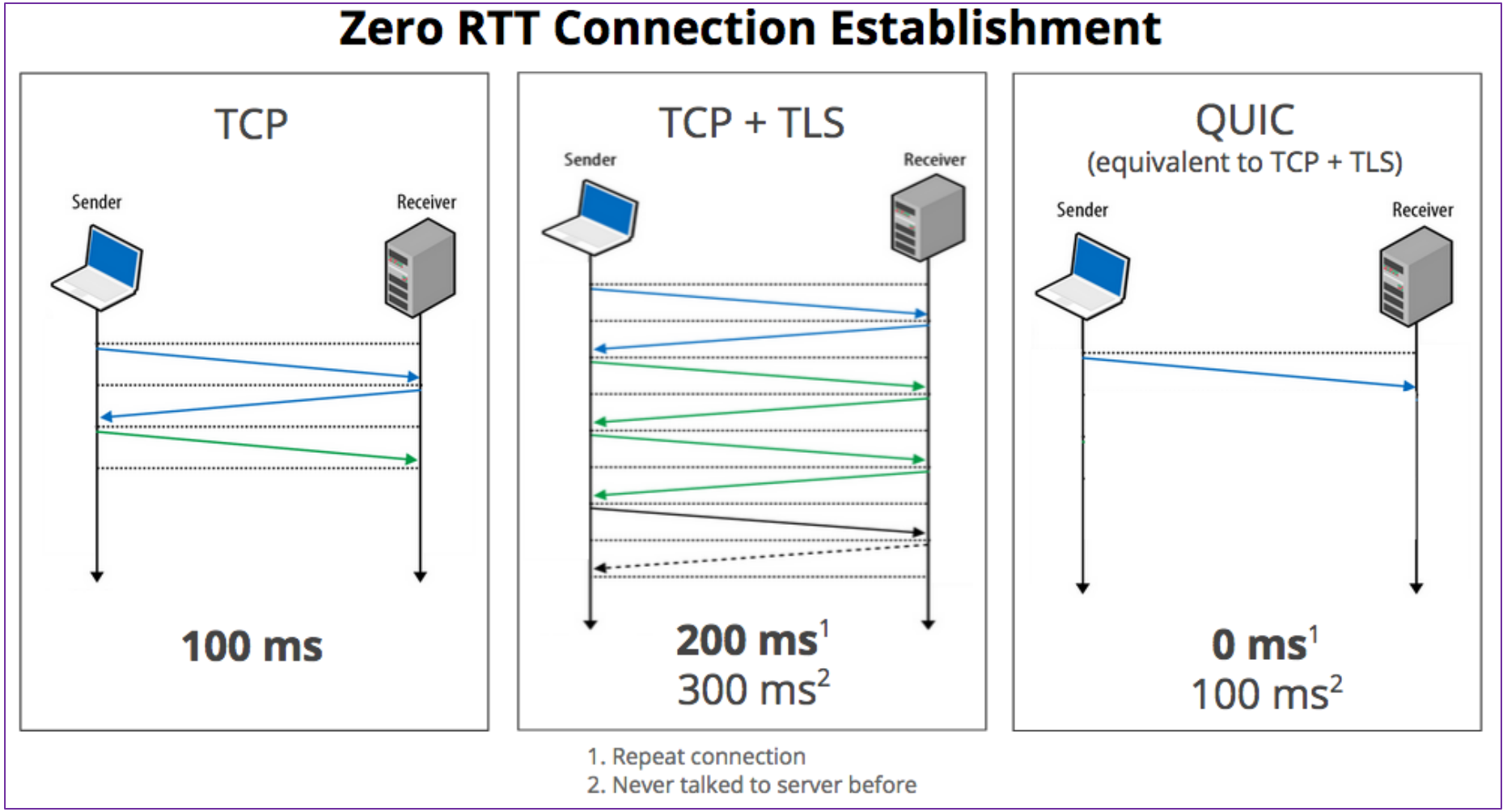
- Quick UDP Internet Connection (QUIC)
  - HTTP/2 over QUIC = HTTP/3
- UDP-based, TCP-behavior
  - User land
- Zero Round Trip
- Congestion Control
- Connection Migration
- TLS 1.3
- Multiplexing without head-of-line blocking
- Alt-svc Header

HTTP/3 Stack



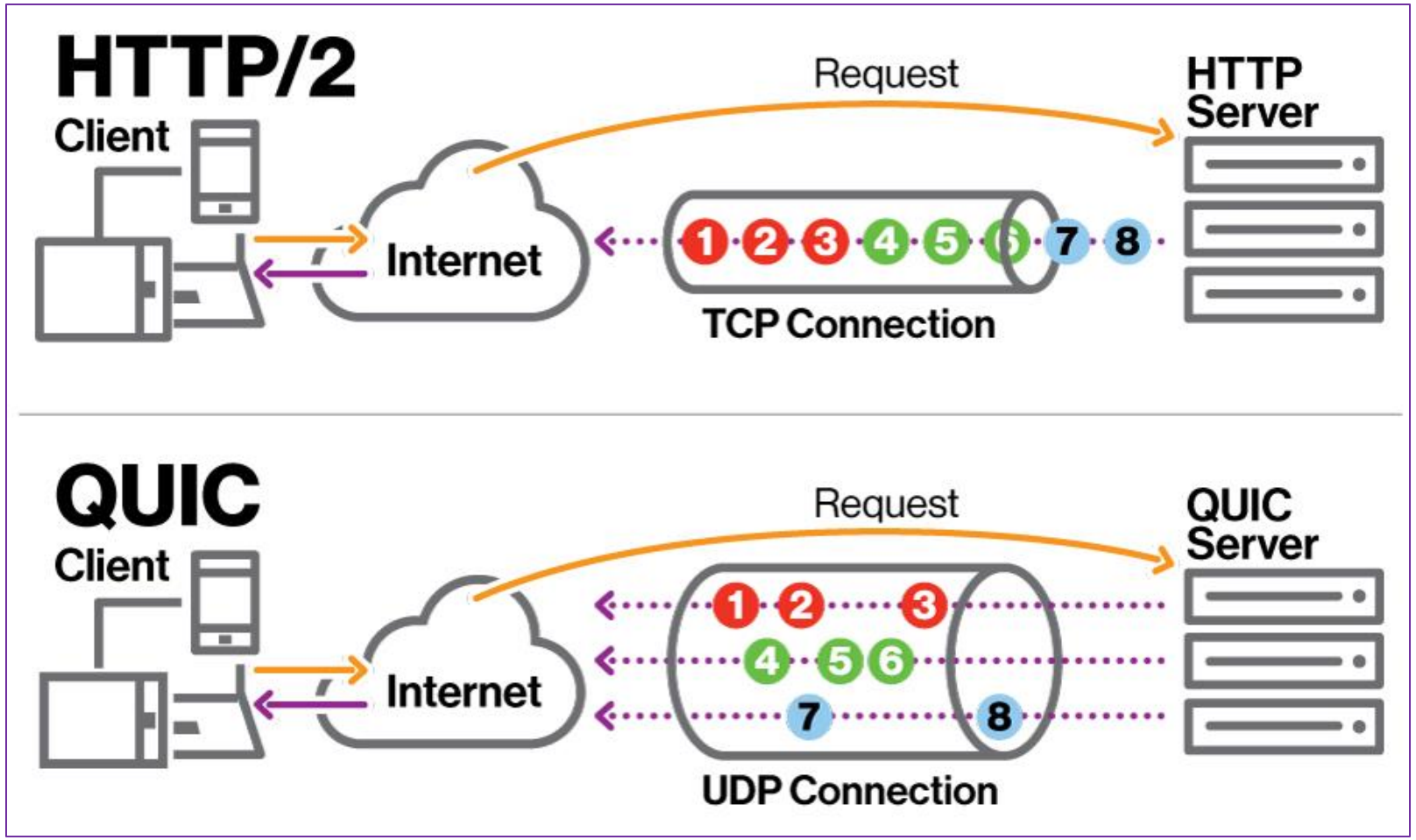
SOURCE: HTTP/3 Explained, https://http3-explained.haxx.se/en/the-protocol

QUIC Trips



SOURCE: https://blog.chromium.org/2015/04/a-quic-update-on-googles-experimental.html

QUIC Connections



Source: https://www.verizondigitalmedia.com/blog/how-quic-speeds-up-all-web-applications/

Merlin Application Concepts

- Message Structure
  - Base message
  - Nested payloads
  - Message padding
- OPAQUE Key Exchange Protocol
- HTTP Payload
  - JSON Web Encryption
  - Golang gob encoded
- JWT Authentication
  - Encrypted

Messages – Base

Version:1

ID:3b882778-17a8-4ec6-8298-f4dca2d928d0

Type: AgentControl

Payload:{

Job:VbdBrHASoF

Command:kill

Args:

Result:

}

Padding:ytSchKUhEEPrEgOdwroplSqGHJYoCTJgne<SNIP>YJnpEkTplOreEEVCBibbjLMsfiyZdZmUdaFXzwYuWXurPWnbkk

Token:eyJhbGciOiJkaXIiLCJjdHkiOiJKV1QiLCJlbmMiOiJBMjU2R0NNIiwidHlwIjoiSldUIn0..DU3W5cRWS1ko3HA1.qnaEKditbRuRVoP4IORtL_LckGe9OX6_F9MUH2F8vslxb5d7pbdJui4qrFuqTQY-8070CyACqRsupbVJGirf0lrPxKCvSZlDl47ZAiFKI-oQ8dE6ApE8zzyj75hbNMyI_qgrL7G6_lqj4nRaAZEHkx2iEh1HFYe1giNme7pyNlGb188oGXgSjyJwvOJwNayhveqVUj1ypOYLcC2z7XZkgE77BefdRi-IJV9AIJkGDy015xhOrLrl3OKCAhvj7a-X5eWQLuGbrn2yjYAdSCsKA7sJBA.jhaDLxfnUohxGZ17JH6U0w

Key Exchange

- Pre-Shared Asymmetric Key
- Password Authenticated Key Exchange (PAKE)
- Encrypted Key Exchange
  - Pre-Shared Symmetric Key
  - Asymmetric Keys
- OPAQUE Key Exchange Protocol
  - IETF Draft
  - Registration
  - Mutual Authentication
  - Secret Salt
  - Encrypted Envelope

OPAQUE Registration – Step 1

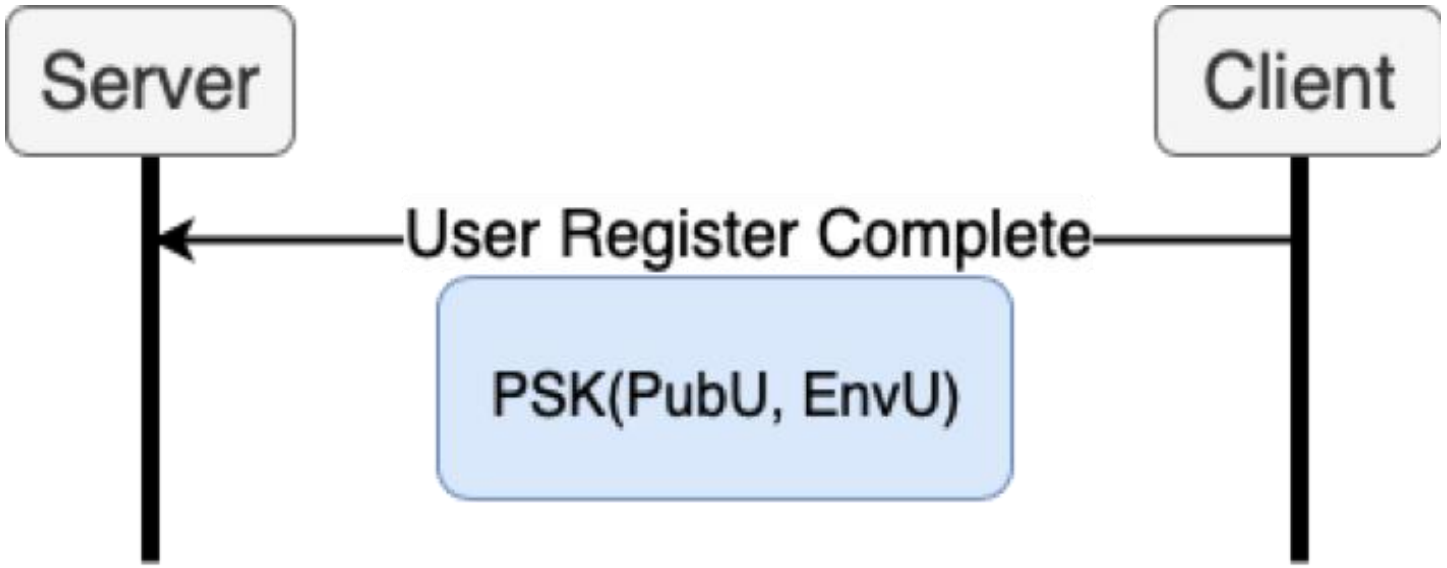
- User Password (PwdU)
  - 30 characters
  - 5,000 PBKDF2 Iterations
  - Unique per Agent
  - Never Transmitted
- Pre-Shared Key (PSK)
  - Default: merlin
  - Not an OPAQUE Requirement
- Alpha
  - Derived from PwdU
  - Can be recovered by attacker

OPAQUE Registration – Step 2

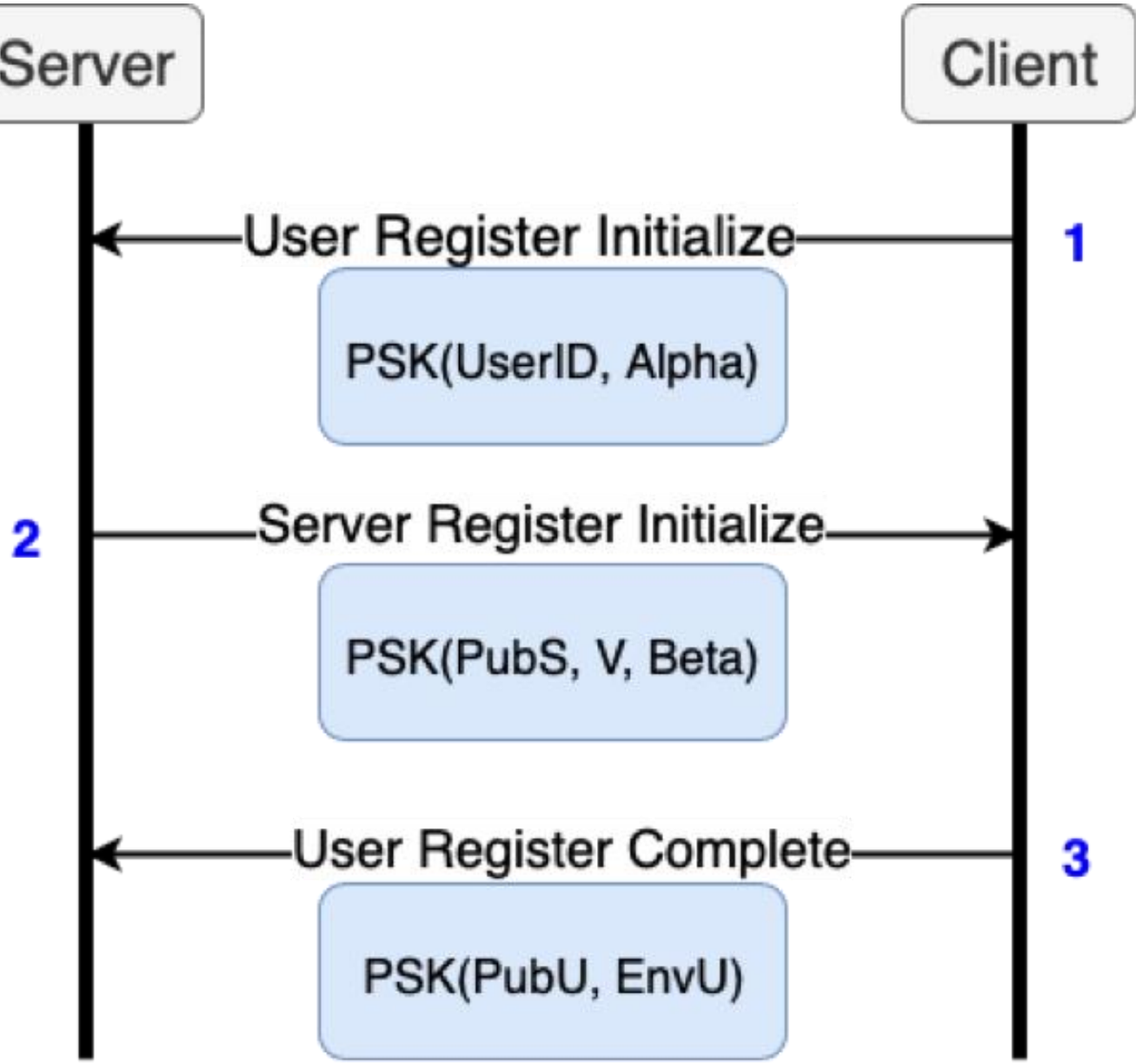
- Server's Public Key (PubS)
- Per-Agent Secret Salt (kU)
  - Never Transmitted
- Computed Second Salt (V)
- Beta
  - Derived from Alpha & kU

OPAQUE Registration – Step 3

- Random Password (RwdU)
  - PwdU + Beta + V
  - Can't be calculated by server
- User's Public Key (PubU)
- User's Private Key (PrivU)
- Encrypted Envelope (EnvU)
  - PubU + PrivU + PubS
  - Encrypted with RwdU
  - Stored on the server

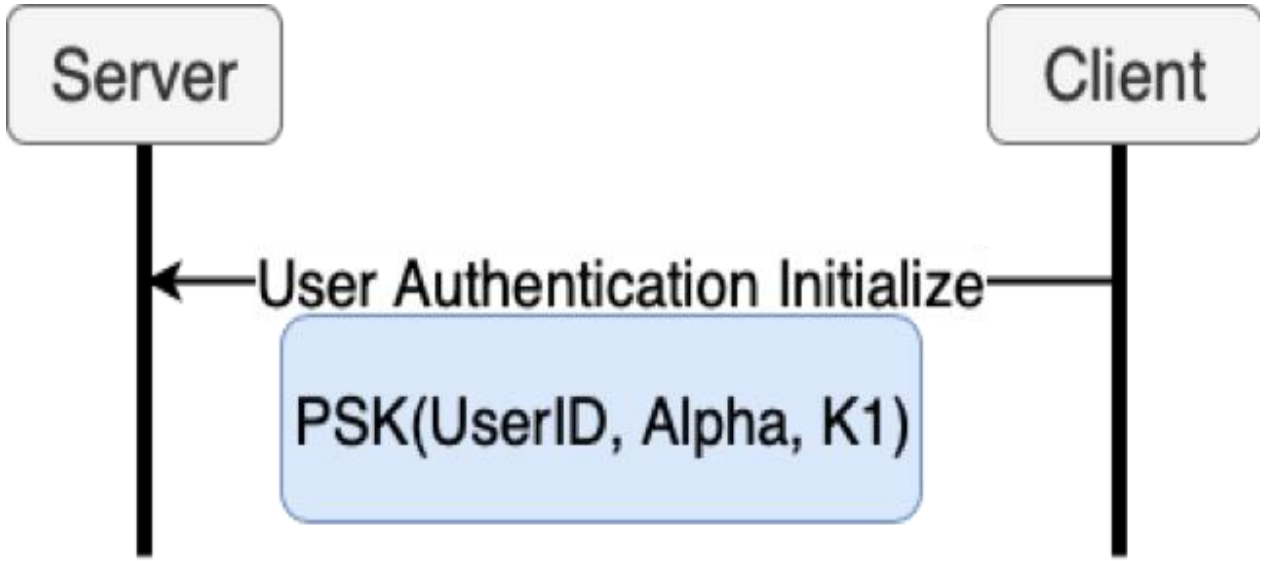


Registration



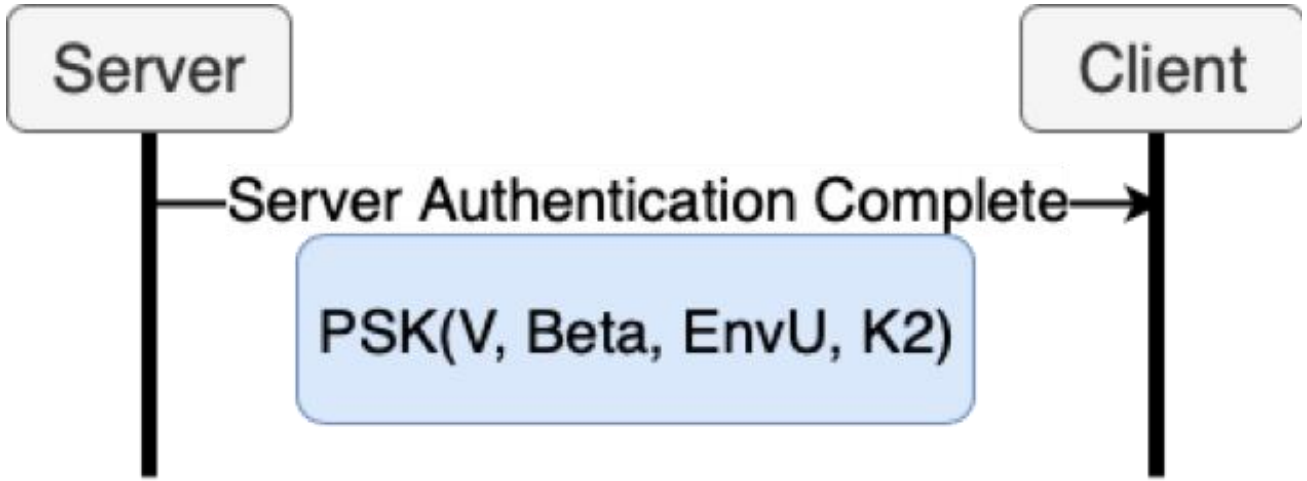
OPAQUE Authentication – Step 1

- Authenticated Diffie-Hellman Key Exchange
  - SIGMA-I protocol
- Key Exchange Message 1 (K1)
- Client Generates Alpha Same as Registration



OPAQUE Authentication – Step 2

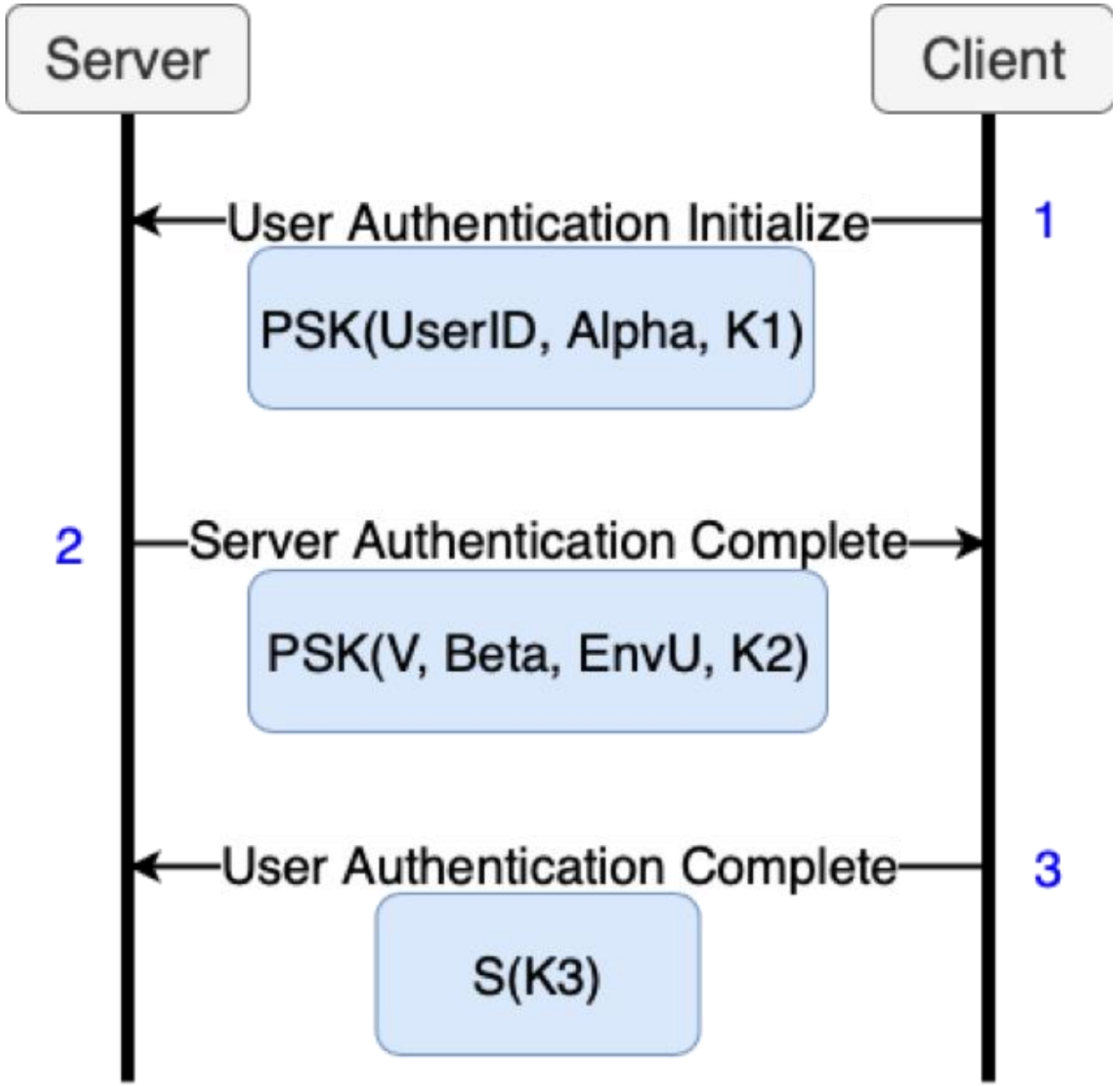
- Generates Beta Same as Registration
- Generate Secret Salt (V) Same as Registration
- Lookup EnvU From Registration
- Key Exchange 2 (K2)
  - Derived from PrivS & K1
- Symmetric Secret (S) Derived From K1 & K2 on Server



OPAQUE Authentication – Step 3

- Generates RwdU Same as Registration
  - PwdU + Beta + V
- Decrypt EnvU
  - PrivU
  - PubU
  - PubS
- Symmetric Secret (S) Derived
- Key Exchange 3 (K3)
- All Traffic Encrypted with S

Authentication



PRISM

- Remember the HTTP/2 Connection Preface?
- OPAQUE Registration Step 1
  - Default Pre-Shared Key: merlin
  - Must know the URL & PSK
- Pre-OPAQUE v0.7.0
  - StatusCheckIn JSON message
- DON'T USE THE DEFAULT PSK!!

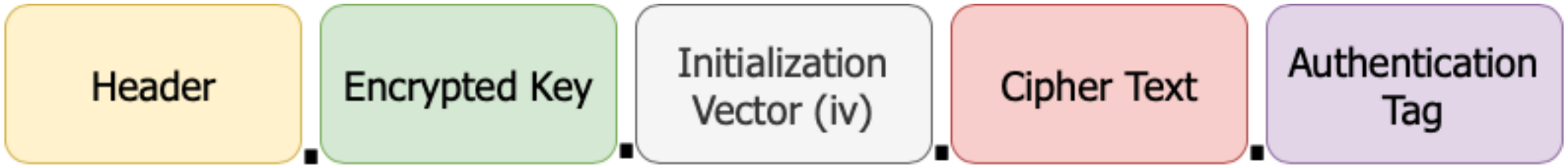
```
kali@kali:~$ ./PRISM-Linux-x64 -proto http -url http://demo.m3rlin.io/SOCON/demo.aspx -psk m3r1nD3m0
[i]Connecting to http://demo.m3rlin.io/SOCON/demo.aspx checking for Merlin server version v0.7.0.BETA or earlier
[-]http://demo.m3rlin.io/SOCON/demo.aspx is not a Merlin server
[i]Connecting to http://demo.m3rlin.io/SOCON/demo.aspx checking for Merlin server version v0.8.0.BETA or greater
[+]Verified Merlin server v0.8.0.BETA or greater instance at http://demo.m3rlin.io/SOCON/demo.aspx
```

JavaScript Object

- Javascript Object Signing and Encryption (JOSE) Working Group
- JavaScript Object Notation (JSON)
- JSON Web Signature (JWS)
  - RFC 7515
- JSON Web Encryption (JWE)
  - RFC 7516
- JSON Web Algorithms (JWA)
  - RFC 7518
- JSON Web Token (JWT)
  - RFC 7519

HTTP Traffic Payload

- JSON Web Encryption (JWE)
  - PBKDF2 w/ HMAC SHA-512 AES 256 GCM Key Wrap
- Per-Message Content Encryption Key (CEK)
  - Initial: PSK
  - Authenticated: OPAQUE Secret
- JSON Compact Serialization Format
- Gob encoded



JWE Key Management

- Key Management Algorithm (alg):
  - 500,000 PBKDF2 Iterations
  - HMAC SHA-512
  - AES 256 key wrap
- Encryption (enc)
  - AES 256 GCM
- PBKDF2 Iterations (p2c)
- Salt (p2s)
  - Random 128-bit

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "PBES2-HS512+A256KW",
  "enc": "A256GCM",
  "p2c": 500000,
  "p2s": "YyOuE7MheftR6QOnny0_Rw"
}
```

Server Response - Encrypted

```
HTTP/1.1 200 OK
Content-Type: application/octet-stream
Date: Sat, 31 Oct 2020 21:07:41 GMT
Transfer-Encoding: chunked

........eyJhbGciOiJQQkVTMi1IUzUxMitBMjU2S1ciLCJlbmMiOiJBMjU2R0NNIiwicDJjIjo1MDAwMDAsInAycyI6ImZYYX
J1bm9fdzBBSTdIUi02QlllMUEifQ.12VdtyIdsnXXTTQ_WZvwH1--BC0wlDISTgKM_c6ypCZeVzlUv4oldw.W-
NUc4x0wQzTlh2Y.sioVBfz-PkzIJD6eOM24PLckIrk9DWIqz3MTK4kktLVI_IU-G5bmKw_JJ-
ZMDdTMCdz3W18da8ACR1g_nGczt6ByjtIoRmwbeLbMGYFVDDfTpHFqnnNdTvRDNN6_X_sDmBW0XQKCxLMnovFpT8zFLigZgard
YFM5mUPv2TH0zAIiLpYEBqtBdQ6esy36XHKIzaEpU4cba9Ej3cr7NszKUpbEK8bNQbsELN6OIXh2iMrl2SJNIUeoA071KKh-
t7OjCU_qk0YOk5cwuSxj7MHZBaIZWcbfMuPxsWN9OMsK_Cd9J8Bl0dKopUe0rIezatH63kNr6j_VzaPgoS8oGsZQYz4zwqIPNq
27pjPw5Vakj6JtLHlk-
iFuiFhUxwW_6xNCvGdPYEAPavZ6wQkiAqvZPFaLZnuSeH8gQpJYd153UhSVRwtOZxvvmHNhtLkGTr_8epwWk5SitfhtsLDNpQp
hhE4l68bFR2q9WZTH7WZ7oJy3MnoY7I5LKO2He3bidZNLJAOe_Ou2vS6LU_Ssul1iuSO1zOolDs6_1PaMDlWgvdJrUJioVC6j6
uN7qy8ZXwyfRY472tFkepLJWKaj8cqECLW7JVqhPgnxsVNa1uX0glnrCyhs2LeWczpZfGlUgXYJDSLZtJ2IelZWQL-DWo-
KDphPPuZA4lDmwomihQjAB_AmsYgMGiY_z8rSBIBTge5ubr1Vrb3F5NGtARw8UpiYRPrOKCZvGGzJ5i5F7KFLUe2W8ZOxEyQQx
KB18okKzq0ZSYLyuO5rKGKiyvZmzEeCs440T2vYqV3ZBwPQAJpluFJAsnmR-qkAxmMuwDUdrfC3Yn-
SpXxKIQVRmwpK7zILXAMgLnITe-K1BHCaDPdTN2KBl-W25-
RiuOSIRsVBoJqaDPnFC0d3MOzahwQWEhQpu8yQI2eoS9tMycGdn0I_c7f0IdvkJ_fEM-T9LUeu2kY851f3MIQdSlceuC-
Pb3GdsDibFHN9HUuCXfcaxixY2O85Ws6Ib_hLed7mzPOvCYWSfsz2ADMiNQKXWZM6LMvKVE-
d829eFGiVYLXd0jBxrWb7h8JXE_P1x2Ax3EJhC5BAqfW4WCrCDh2CpepNYPwB3OgbWx_N69LDK9_LJOVPb1wp-
MxaCAox6U7sK7i-craBkZTaqvHDUDbqrnhidJpUAGCHxy1HMd79d_96uF-
rrugE4FHCskztpq2mNCNOQPxjIZAgcwu_LAnsrHlFkZId34wJ0aIVDn0Icd4KWE1DPjdH5HdPWT2-
Iiisz_Jq11hTHR_VqpefuJ8pUxEJXxfCDY-NpO81Z3CtmsRf5iF7RBmdZXJPp-mY84a3SPtE8n9F-
odcl-8wtNewlvRhUEposNAB--qcEs6VyL-zZX6-
J0jpwJD5lSgs94_gorknhwO9n9mjGJykcYMPbnw19pjbctmiHsTO1Q4A_QuWHAQJxMj8ad1sKikTAXD1U-
```

JSON Web Tokens

SIGNED

ENCRYPTED

COMPACT SERIALIZATION

JWT – Initial Process

Initial JSON Web Token

OPAQUE Only

Encryption

Algorithm: AES 256 GCM
Key Agreement: DIRECT
Key: SHA-256 of PSK

Signature

Algorithm: HMAC + SHA256 (HS256) Algorithm
Key: SHA-256 of PSK
merlin: GCUL3aW8AuLPYcS+xKyL3iS3Kgu6sXSxQTGdS8d2TTQ=

Expires after 10 seconds

JWT – Authenticated Process

Authenticated JSON Web Token

Encrypted & Signed by Merlin Server

Encryption

Algorithm: AES 256 GCM

Key Agreement: DIRECT

Key: Per Listener Random 32-byte Key

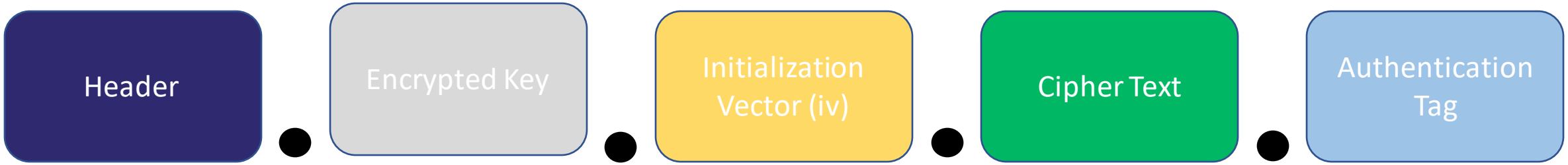
Signature

Algorithm: HMAC + SHA256 (HS256) Algorithm

Key: Per Listener Random 32-byte Key

Expires after (sleep + skew) * MaxRetry

JWT - Encrypted




```
eyJhbGciOiJkaXIiLCJjdHkiOiJKV1QiLCJlbmMiOiJBMjU2R0NNIiwidH
lwIjoiSldUIn0..C9pb7qubWMdrUGJE.j-oy65QIBBcpogWTjhEznW9-
aTyezB4M6rStz_Oq81oTLSJTrTAIovDcmadzP3I_lAxAnb7DEROFc6wi3P
oeCFedJwx3qNGJmljsM0NiVN8Vihe9MoE7WDBI-
ZvzBAM9r0paQlzF9GRKG53d4lsEWCx_kfA1pAlpR65g6DXqeiTX8mUEsK2
XXgMjQPh2wMmxrX4Cb5koqoqMMTJuzNOkmtMhvAJMRzQOlDzWyYBGd7Aal
40beBa6EGsnR0w.JPj2N77DOAdmCrZ4QYKu8A
```

JWT – HTTP Traffic

```
POST /SOCON/demo.aspx HTTP/1.1
Host: demo.m3rlin.io
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/40.0.2214.85 Safari/537.36
Content-Length: 5957
Authorization: Bearer
eyJhbGciOiJkaXIiLCJjdHkiOiJKV1QiLCJlbmMiOiJBMjU2R0NNIiwidHlwIjoiSldUIn0..C9pb7qubWMdrUGJE.j-
oy65QIBBcpogWTjhEznW9-
aTyezB4M6rStz_Oq81oTLSJTrTAIovDcmadzP3I_lAxAnb7DEROFc6wi3PoeCFedJwx3qNGJmljsM0NiVN8Vihe9MoE7WDBI-
ZvzBAM9r0paQlzF9GRKG53d4lsEWCx_kfA1pAlpR65g6DXqeiTX8mUEsK2XXgMjQPh2wMmxrX4Cb5koqoqMMTJuzNOkmtMhvAJ
MRzQQlDzWyYBGd7Aal40beBa6FGsnR0w.JPj2N77DOAdmCrZ4QYKu8A
Content-Type: application/octet-stream; charset=utf-8
Accept-Encoding: gzip

..B....=eyJhbGciOiJQQkVTMi1IUzUxMitBMjU2S1ciLCJlbmMiOiJBMjU2R0NNIiwicDJjIjo1MDAwMDAsInAycyI6IlZOY1
lnUGtGelBZLUY4QjdZcC1rVEEifQ.3xmm2LaO1R1XRP9lMthmtGSNuE20PW-ckRYkYc3bADd4h88j1Hne2A.mes6sU-
```

Merlin Server

- Tab Completion
- Help Menus
- Command Aliases
- Module Support
- Server Logs
- UTC Timestamps
- Orphaned Agent Handling
- Host System Command Execution
- Self-Signed TLS Certificate Generation

Merlin Server – Menus

- Main
- Listener
  - Configure listeners to receive agent traffic
- Agent
  - Interact with and control agents
  - Per agent log file in ./data/agents/<ID>
- Modules
  - Located in ./data/modules
  - JSON File
  - Standard or Extended

Demo – Menus Demo

Merlin Agent

- Configure at execution time or compile time
- Native Commands
- Verbose and Debug Output
- DLL Agent
- Kill Date
- Detailed Agent Logs
- Max Retry
- Dynamic JA3 Hash Modification
- HTTP Host Header (Domain Fronting)

Building an Agent

- Perquisites
  - go
  - git
  - mingw-w64
- Makefile Parameters
  - URL
  - PSK
  - Proxy
  - Host
  - Proto
  - JA3
- Output: `data/temp/<version>/build/`

Demo – Custom Agent

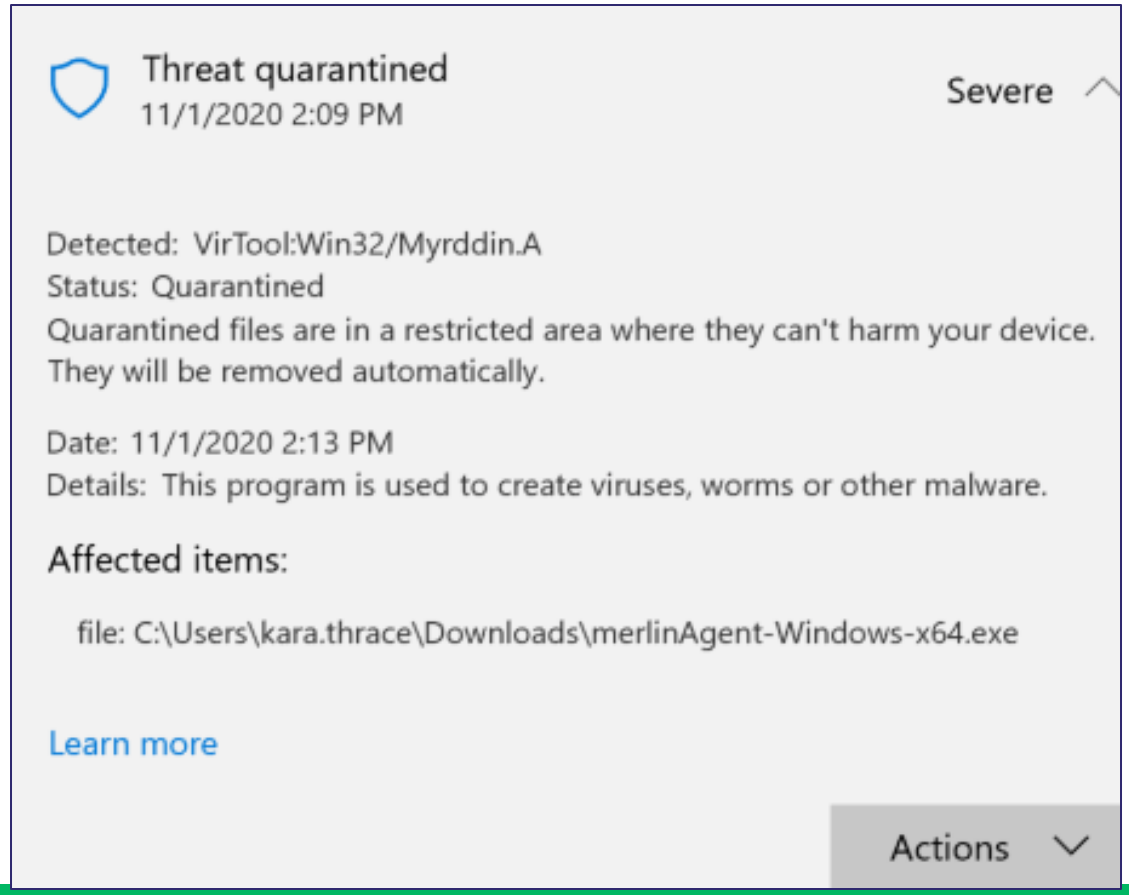
Agent – Domain Fronting

- CDN HTTP/2 Support
  - Support to entry point
  - No support to origin server
- v0.8.0
  - HTTP/1.1 Support
  - HTTP Host Header
  - HTTP Proxy
- AWS
  - Plain-text HTTP

Demo – Domain Fronting

Windows Defender Threat

- Pre-compiled Merlin agent detected by Windows Defender

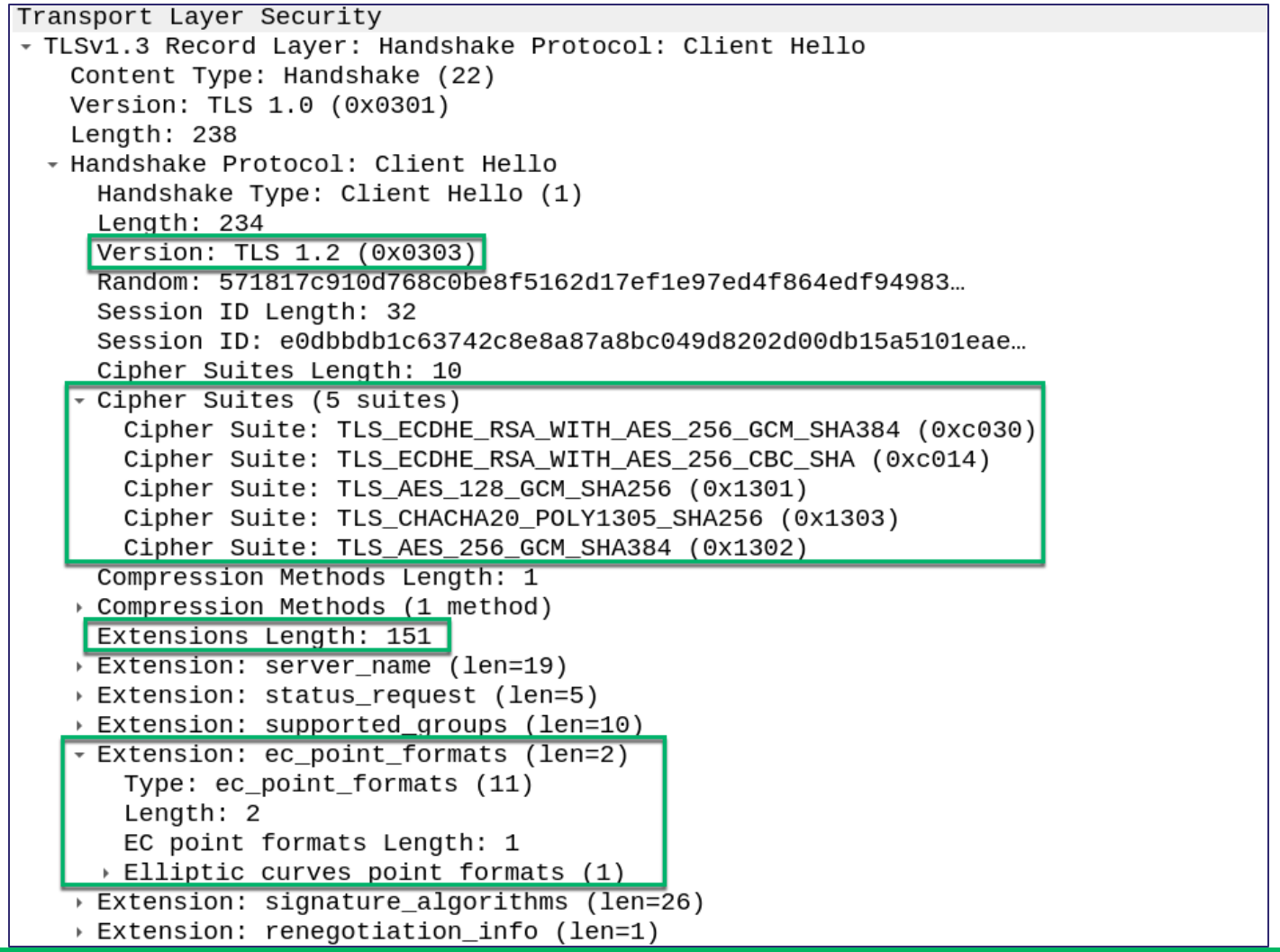


Demo – Defender Bypass

JA3

- JA3: https://github.com/salesforce/ja3
- Hashed Fields (MD5)
  - SSL/TLS Version
  - Ciphers Suites
  - TLS Extensions
  - Elliptic Curves
  - Elliptic Curve Point Formats
- JA3 Fingerprint Database: https://ja3er.com
- Merlin Client 771,49200-49172-4865-4867-4866,0-5-10-11-13-65281-16-18-43-51,29-23-24-25,0 --> f57430d0bbb4a97b82945f83e6bab359
- ja3transport: https://github.com/CUCyber/ja3transport

TLS Client Hello Record



Demo – JA3

Modules

- Standard Modules
- macOS
  - HealthInspector
  - Orchard
  - SwiftBelt
  - Bifrost
- Extended Modules
- Windows
  - Minidump
  - Shellcode
  - sRDI

Module - Shellcode

- Methods
  - Self
  - CreateRemoteThread
  - RtlCreateuserThread
  - QueueUserAPC*
- Format
  - Hex (i.e. 5051525356)
  - 0x50, 0x51, 0x52 … with or without spaces and commas
  - \x50\x51\x52\x53
  - Base64 encoded version of above formats
  - Read from file

Demo – Shellcode Injection

Demo - Minidump

Conclusion

- Cross-Platform Server and Agent
- HTTP 1, 2, and 3!
- OPAQUE Authenticated Key Exchange
- JWT Authorization
- JWE Encrypted Payload
- Multiple Listeners
- Customized Agents
- Agent Evasion Tips

MERLIN

www.specterops.io
@specterops
info@specterops.io