TLS 1.3 Tutorial

...

IETF 100 - Singapore 20171112

Sean Turner | sn3rd
Joe Salowey | Tableau software

Will address TLS 1.3's:

Whats

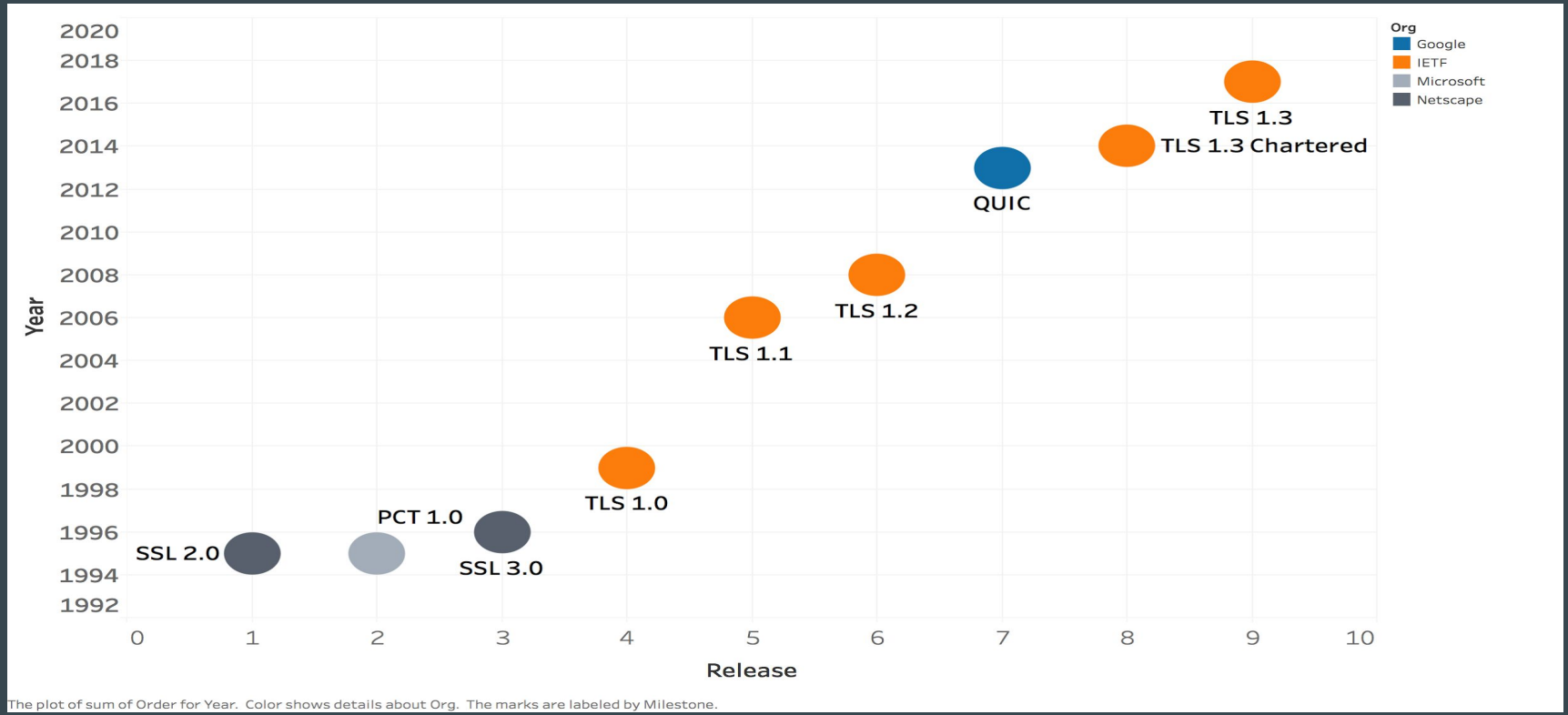
Wheres

Hows

We promise:

Not too Technical
Lots o’ Links
Lame Nerd Humor

Whence does it come?



Who's implementing 1.3?

Open source!

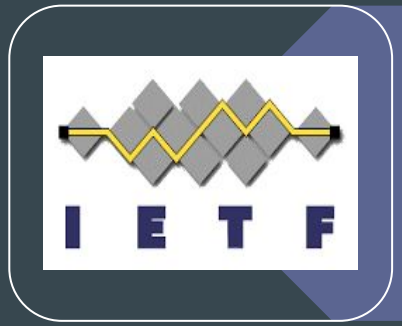
Browsers!

Test servers available!

name	language	role(s)	version	features/lim
NSS	C	C/S	-18	Almost everything, except post-handshak
Mint	Go	C/S	-18	PSK resumption, 0-RTT, HRR
nqsb	OCaml	C/S	-11	PSK/DHE-PSK, no EC*, no client auth, no tls13test.nqsb.io port 4433, records trace PSK/DHE_PSK token: id: 0x0000 secret: 0x000102030405060708090a0b0c0d0e
ProtoTLS	JavaScript	C/S	-13	EC/DHE/PSK, no HelloRetryRequest
miTLS	F*	C/S	-21	EC/DHE/PSK/0-RTT, no RSA-PSS
Tris	Go	S	-18	ECDHE/PSK/0-RTT, no HelloRetryReques
BoringSSL	C	C/S	-18	P-256, X25519, HelloRetryRequest, resun
Wireshark	C	other	-18 to -21	Full decryption and dissection support fo format proposal) and -18 since 2.4.2. Mis bug.
picotls	C	C/S	-21	P-256, X25519, HelloRetryRequest, resun
rustls	Rust	C/S	-20	P-256/P-384/curve25519, HRR, resumpti
Haskell tls	Haskell	C/S	-21	ECDHE w/ P* and X*, full, HRR, PSK, 0RT
Leto	C#	S	-18	DHE, X25519, AES, no PSK no 0RTT. Test
OpenSSL	C	C/S	-21	P-256, X25519, HelloRetryRequest, resun
wolfSSL	C	C/S	-18 or -20	P-256, P-384, X25519, HelloRetryReques Post-Handshake Auth
tlslite-ng	Python	C/S	-21	ECDHE (all), EdDHE (X25519, X448), FFD HelloRetryRequest, RSA, RSA-PSS keys a extension

Where are the specifications?

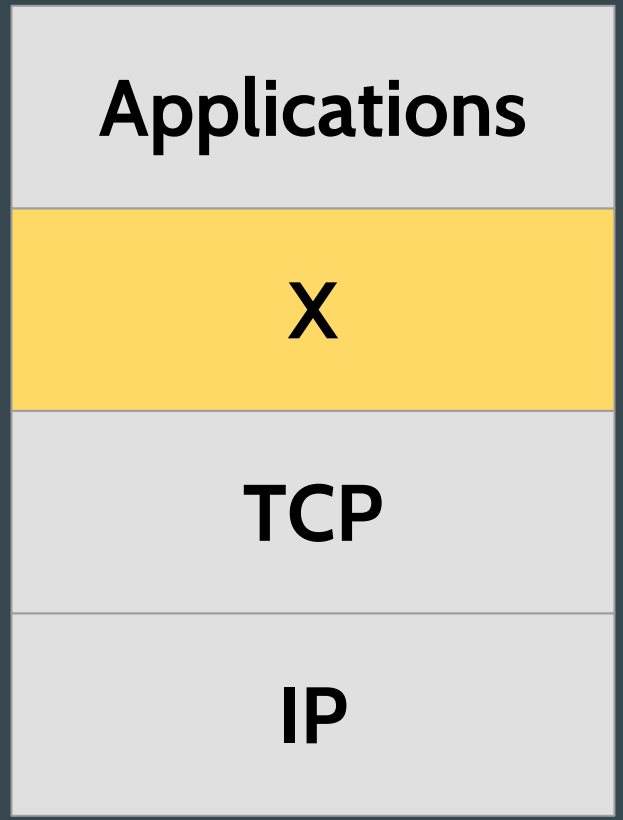
Working copy



Official I-D

Where does it sit?

X marks the spot!



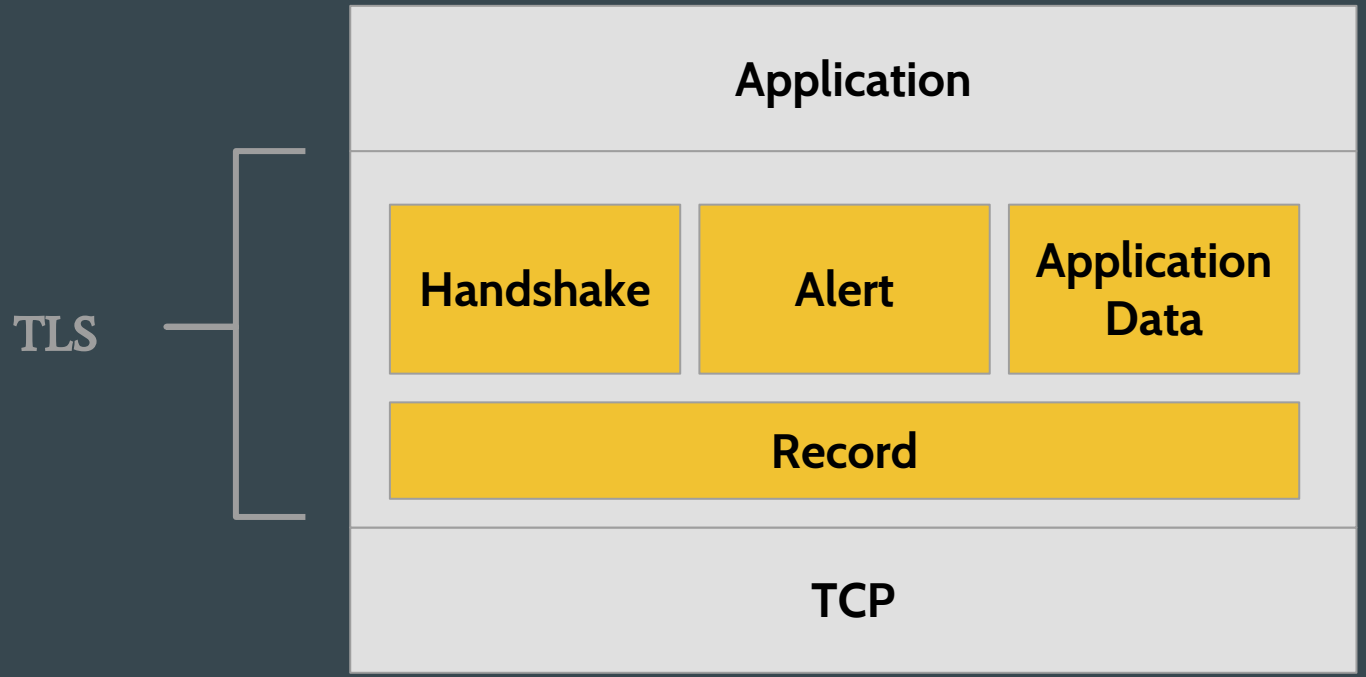
What does it do?

Begone Eve!

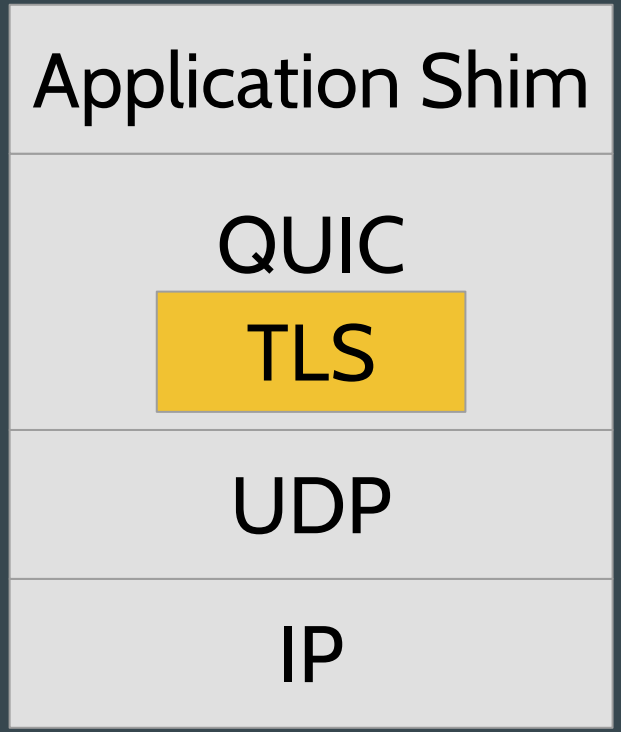
Mallory No More!

From xkcd.com

Wat, Wat! There's how many protocols!?

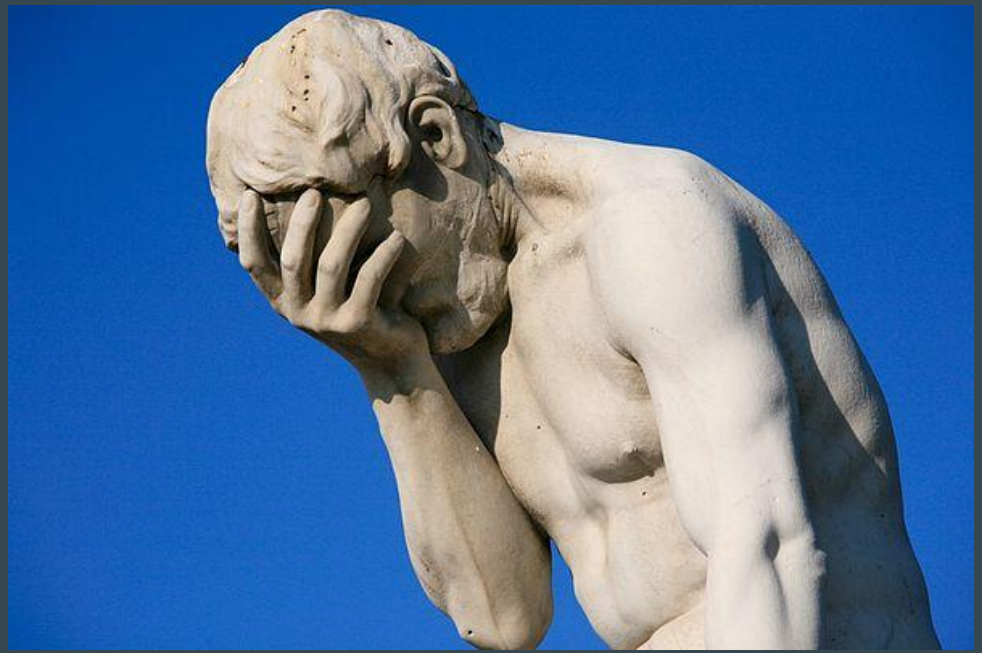


Wat! Wat! You don’t need to use all the protocols?



QUIC does not use
TLS’ Application Data

What was wrong with the previous versions?



Lucky 13

BEAST

Freak

Logjam

Drown

Crime

Breach

Triple Handshake

Poodle

Sweet32

...

What were the design goals?

PRIVATE

Why is it more secure?

What did you remove to make it more secure?

SHA-1

Compression

Stream Ciphers

Static RSA Key Exchange

Renegotiation

Block Ciphers

Why is it more secure?

Record Payload Algorithms: AEAD-only

Key Establishment Algorithms: (EC)DHE or PSK

Convergence of PSK, Session Resumption, Session Tickets and 0-RTT

TLS1.2

>100

Cipher Suites

TLS1.3

005

What algorithms are supported?

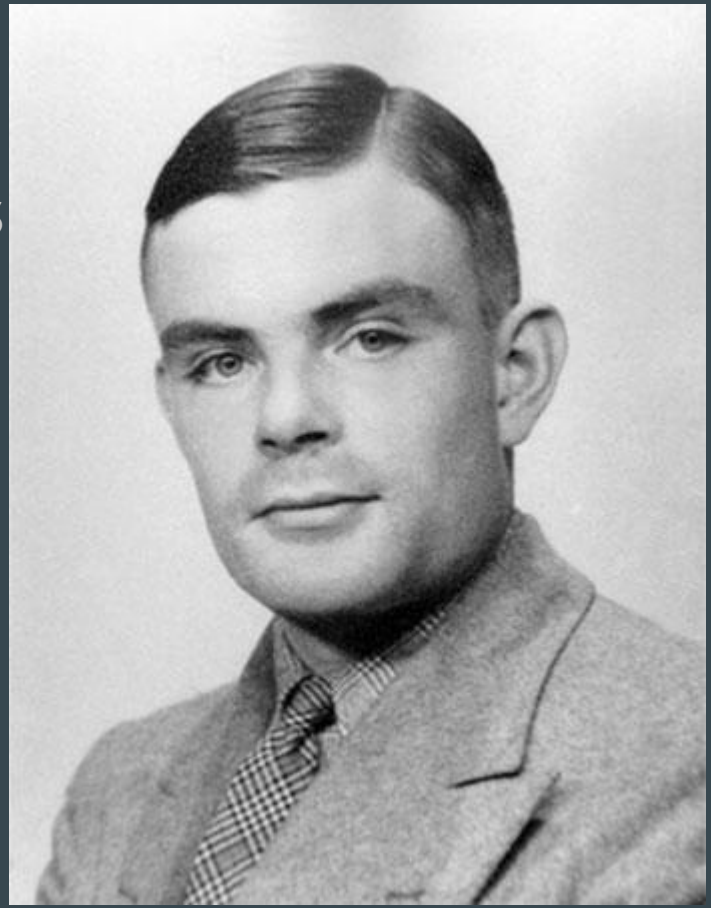
AEAD: AES-GCM, AES-CCM, CHACHA20-Poly1305

ECs:

- Sig: p256, p384, p512, EdDSA (25519 and 448)
- KE Groups: p256, p384, p512, 25519, 448

Named FFDHE Groups

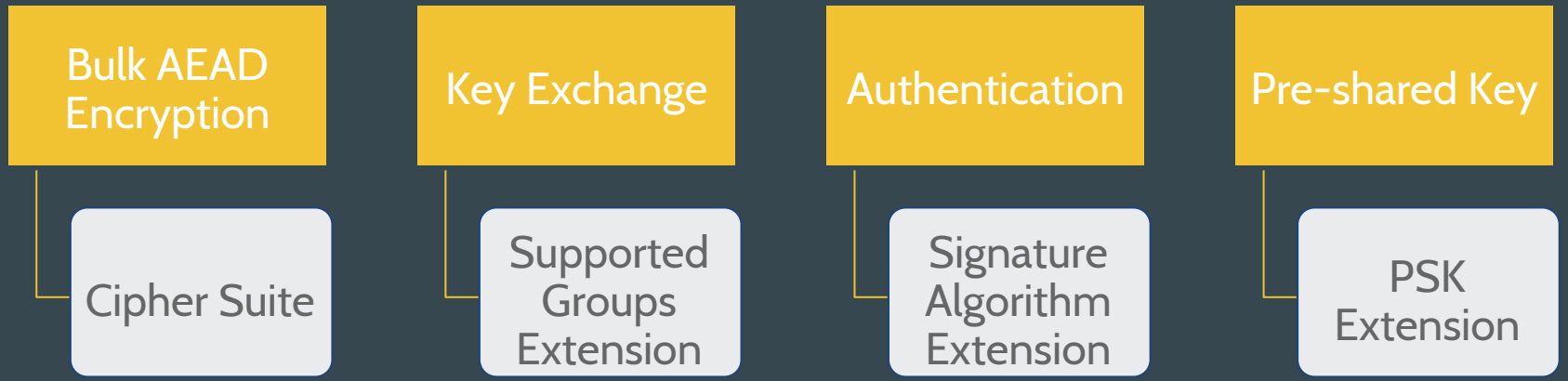
RSA-PSS Signatures



How do you specify ciphers?

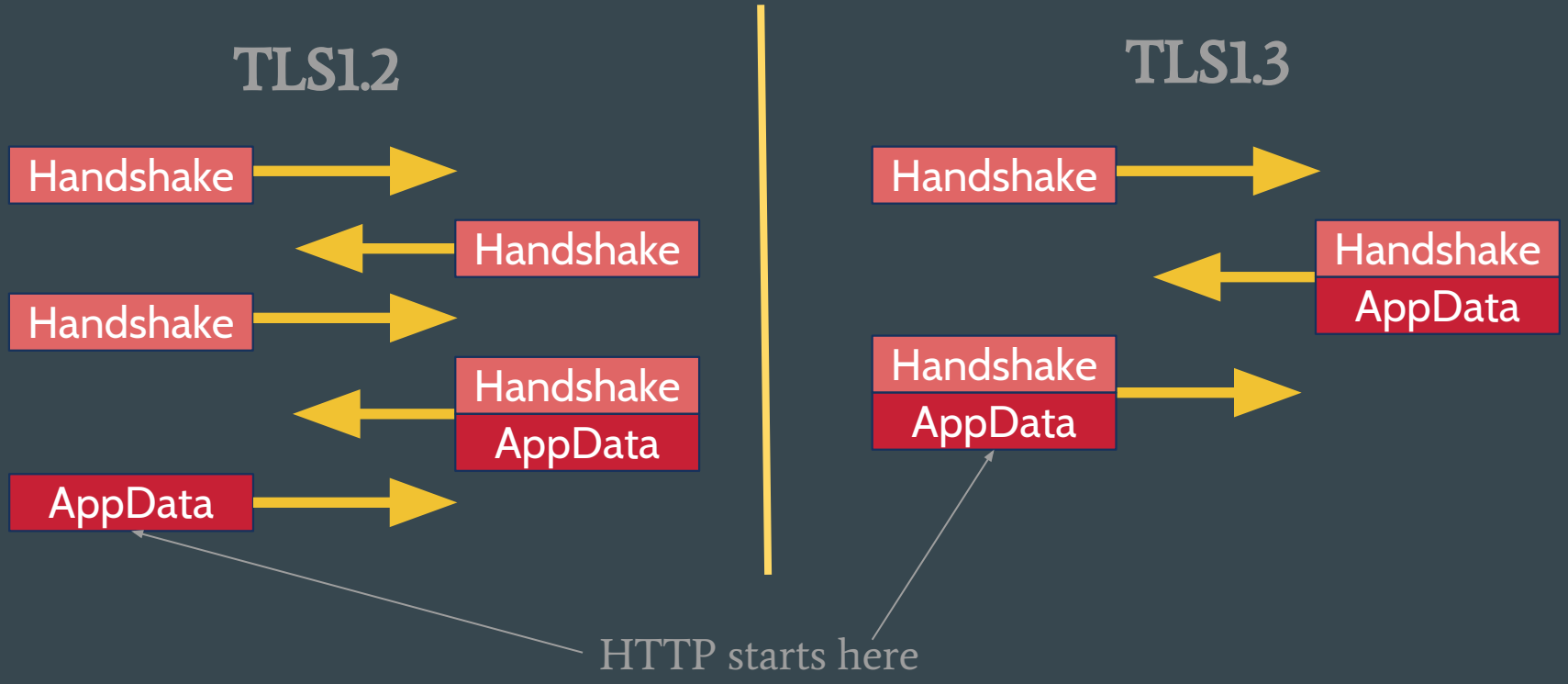
OLD: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA34

NEW: a la carte



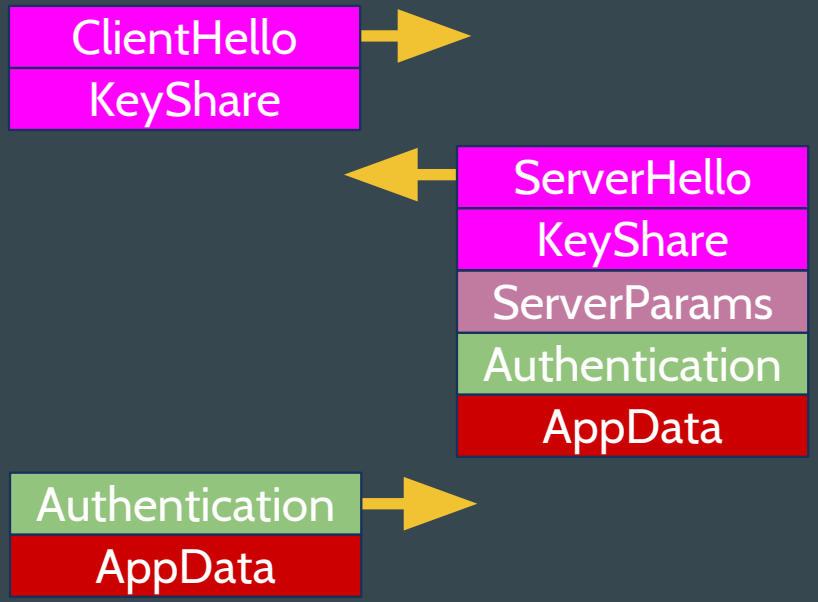
IANA Registry will include Recommended column

Come again - it's faster?

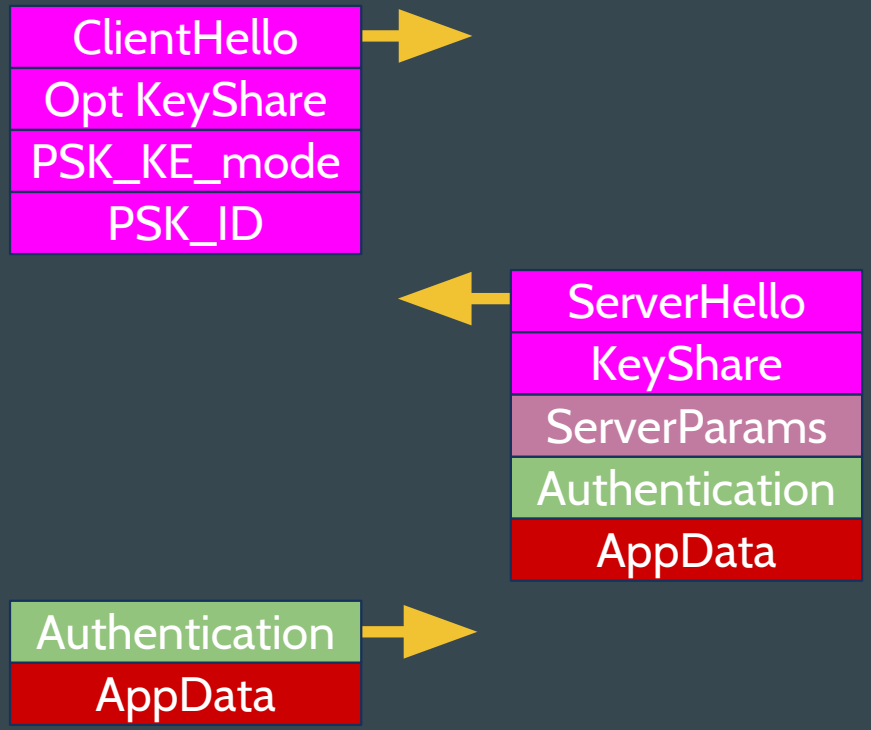


What are the normal modes?

1-RTT

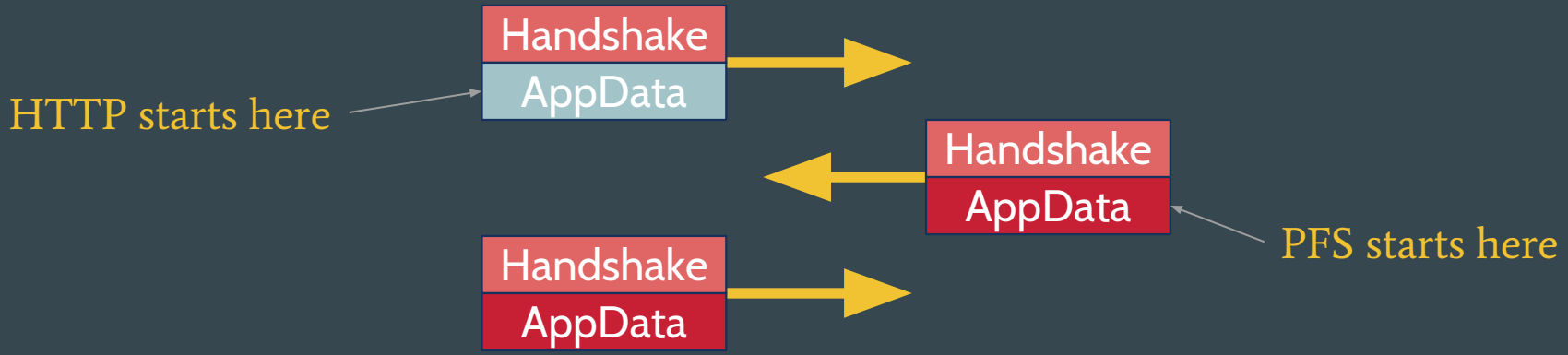


Resumption (PSK)



Is that *all* you got?

TLS1.3 0-RTT Data



WARNING: 0-RTT Data is replayable and not PFS!

It supports record protection?

```
struct {
    opaque content[TLSPlaintext.length];
    ContentType type;
    uint8 zeros[length_of_padding];
} TLSInnerPlaintext;

struct {
    ContentType opaque_type = 23; /* application_data */
    ProtocolVersion legacy_record_version = 0x0301; /* TLS v1.x */
    uint16 length;
    opaque encrypted_record[length];
} TLSCiphertext;
```

Padding for Length Hiding

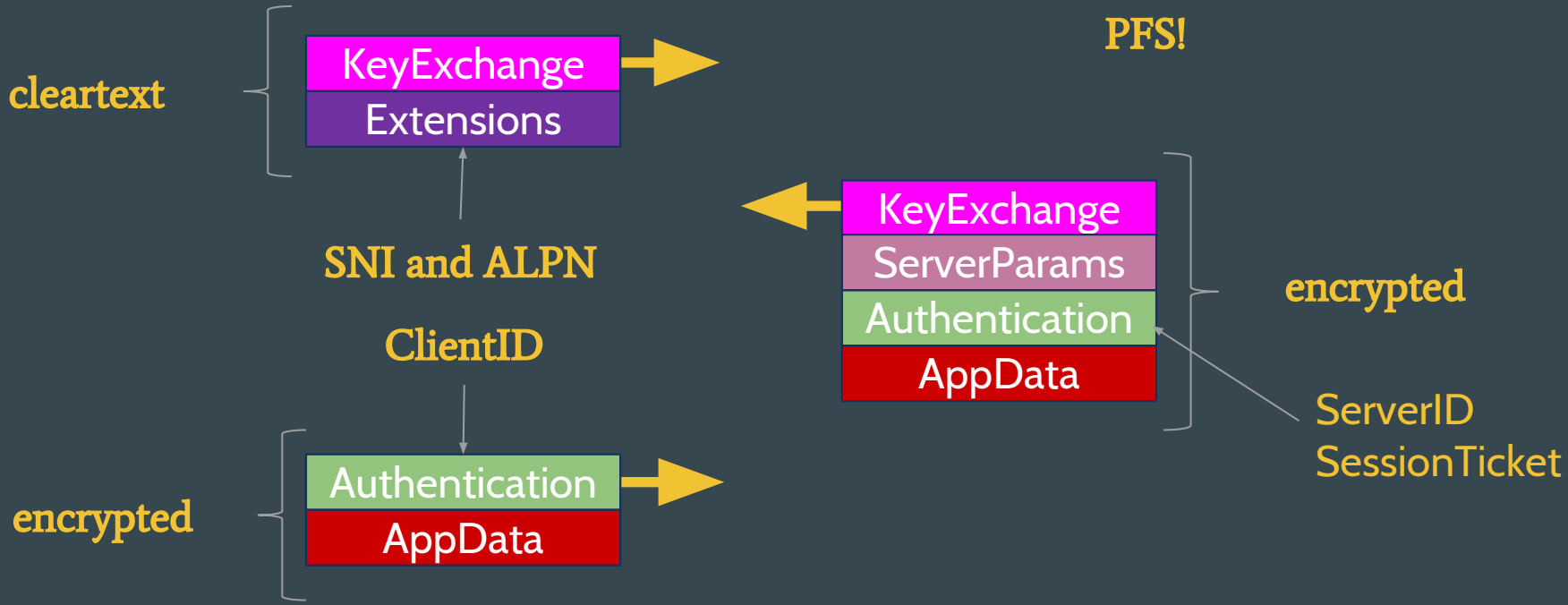
Unencrypted ContentType and Version no longer meaningful

You turned PFS on!?

Perfect Forward Secrecy is the default.

Also available with PSK modes.

You're encrypting more early though, right!?

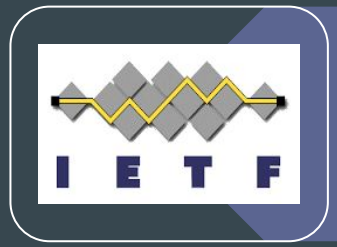


What’s not to like!?

TLS1.3-related drafts

TLS1.3 Test Vectors

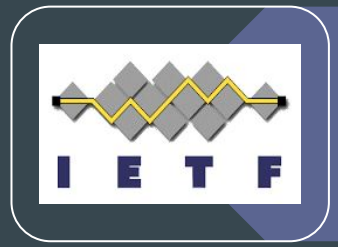
Working copy



Official I-D

DTLS1.3

Working copy



Official I-D