Client	(a)	Server (b)
$egin{aligned} r_a & \stackrel{random}{\longleftarrow} \{0,1\}^{256} \ & Ephermal\ key:\ Q_a & \leftarrow d_aG \end{aligned}$	ClientHello: $r_{\rm a}$ KeyShare: $Q_{\rm a}$ ServerHello: $r_{\rm b}$ KeyShare: $Q_{\rm b}$	$S_{\text{early}} \leftarrow \text{Extract}(0,0)$ $r_{\text{b}} \xleftarrow{\text{random}} \{0,1\}^{256}$ $\text{Ephermal key: } Q_{\text{b}} \leftarrow d_{\text{b}}G$ $\text{Key exchanged via ECDHE: } x \leftarrow (x,y) = d_{\text{a}}Q_{\text{b}}$
		$\begin{split} S_{handshake} &\leftarrow Extract(Derive(S_{early}, 'derived', \emptyset), x) \\ S_{master} &\leftarrow Extract(Derive(S_{handshake}, 'derived', \emptyset), 0) \end{split}$
$\mbox{Handshake traffic key: } K_{\mbox{\scriptsize handshake}_a} \leftarrow \mbox{\sf Derive}(S_{\mbox{\scriptsize handshake}}, \mbox{\sf 'c hs traffic'}, \mbox{\sf transcript}) \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \$	$\{Certificate\colon Public\ key\ with\ CA\ signature\}_{K_{handshake_b}}$	—— Handshake traffic key: $K_{handshake_b} \leftarrow Derive(S_{handshake}, 's\ hs\ traffic', transcript)$
	$\{ CertificateVerify: Transcript \ with \ ECDSA \ signature \}_{K_{handshake_b}} $ $\{ Finished: \ HMAC(K_{finished}, \ transcript) \}_{K_{handshake_b}}$	
$Finished \ key \colon \ K_{finished} \leftarrow Derive(K_{handshake_a}, 'finished', transcript) \ $	$\{ {\sf Finished} \colon {\sf HMAC}(K_{\sf finished}, {\sf transcript}) \}_{K_{\sf handshake}_a}$	$\longrightarrow S_{resumption} \leftarrow Derive(S_{master}, 'res\ master', transcript)$
$\textbf{Application traffic key: } K_{a_0} \leftarrow Derive(S_{master}, 'c ap traffic', transcript) $	$\{Application\ Data\}_{K_{a_0}}$	Application traffic key: $K_{\rm b_0} \leftarrow {\rm Derive}(S_{\rm master}, {\rm 's\ ap\ traffic'}, {\rm transcript})$ \longrightarrow
	NewSessionTicket: {session key ID, IV, encrypted state, HMAC()} $_{K_{b_0}}$ (Connections terminated. That triggers session resumption with 0-RTT)	Creates a pre-shared key (PSK) binding to enable session resumption
$S_{early} \leftarrow Extract(0, S_{resumption})$ $Binder\ key\colon K_{binder} \leftarrow Derive(S_{early}, 'res\ binder', \emptyset)$ $Early\ Traffic\ Key\colon K_{early} \leftarrow Derive(S_{early}, 'c\ e\ traffic', transcript)$ $Finished\ key\colon K_{finished} \leftarrow Derive(K_{binder}, 'finished', transcript)$	ClientHello: KeyShare: PskKeyExchangeModes: 'psk_dhe_ke' EarlyDataIndication PreSharedKey: $\{session\ key\ ID,\ HMAC(K_{finished},\ transcript)\}$ $\{Application\ Data\}_{K_{early}}$	
	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	$\underline{\phantom{AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA$
•	$\{ Finished \colon HMAC(K_{finished}, transcript) \}_{K_{handshake_b}}$	$S_{master} \leftarrow Extract(Derive(S_{handshake}, 'derived', \emptyset), 0)$ $$
	{Application Data} _{Kb0}	—— Application traffic key: $K_{\mathrm{b_0}} \leftarrow \mathrm{Derive}(S_{\mathrm{master}}, \mathrm{'s\ ap\ traffic'}, \mathrm{transcript})$
$Finished \ key: \ K_{finished} \leftarrow Derive(K_{handshake_a}, 'finished', transcript) \ -$	$\{{\sf EndOfEarlyData}\}_{K_{\sf early}}$ $\{{\sf Finished}\colon {\sf HMAC}(K_{\sf finished}, {\sf transcript})\}_{K_{\sf handshake}_a}$	→
$Application\ traffic\ key:\ K_{a_0} \leftarrow Derive(S_{master}, 'c\ ap\ traffic', transcript) \ -$	$\{ {\sf Application \ Data} \}_{K_{{\sf b}_0}}$ $\{ {\sf Application \ Data} \}_{K_{{\sf a}_0}}$	
Disclaimer: this diagram is a rough sketch of the TLS 1.3 handshake and record protocol. It serves as a quickstarter to understand the protocol flows. It may contain inaccurate or oversimplified representations. 1) TLS Settings Cipher Suite: TLS_AES_128_GCM_SHA256 Digital Signature: ecdsa_secp256r1_sha256 Key Exchange: (Q, d)		

Digital Signature: ecdsa_secp256r1_sha256

Key Exchange: secp256r1 (NIST P-256) with (G, n) as part of domain parameters, with public and private key in the form of (Q, d)Pre-Shared Key Cipher: TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA384

2) Protocol Notations

Key Extraction Function: Extract(salt, keying material)

Key Derive Function: Derive(secret, label, transcript), where transcript is the concatenation of each included handshake message.

Encryption: $\{\text{plaintext}\}_{\text{key}}$, which denotes an AEAD-Encrypt operation with write key and IV generated from key.