Let's Encrypt Made TLS Free. The Certificate Model Is Still the Same.

by Nick Clark | Published March 28, 2026 | PDF

Let's Encrypt transformed web security by providing free, automated TLS certificates through the ACME protocol, removing cost and complexity as barriers to HTTPS adoption. The impact is enormous: hundreds of millions of certificates issued, HTTPS adoption rising from minority to majority. But Let's Encrypt issues the same structural artifact as any other CA: a certificate binding a domain name to a public key, signed by a chain of trust, with a fixed lifetime. Making certificates free did not change what certificates are. The structural gap is between ubiquitous certificate issuance and an identity model that does not require certificates.

Let's Encrypt's contribution to web security through free, automated certificate issuance is one of the most impactful infrastructure projects in recent history. The gap described here is about the certificate model, not about Let's Encrypt's mission.

Automation solved issuance, not the model

ACME automated the certificate lifecycle: domain validation, certificate issuance, installation, and renewal all happen without manual intervention. This eliminated the operational burden that kept many sites on HTTP. But automation made the certificate model easier to use. It did not change the model.

Automated certificates still depend on stored private keys on the server. They still have fixed lifetimes requiring renewal. They still depend on the CA's signing key and the browser's trust store. The operations are automated. The structural dependencies are identical.

Free certificates normalize the credential model

By making certificates free and automated, Let's Encrypt removed the economic incentive to question the certificate model. When certificates were expensive and manual, there was motivation to explore alternatives. With free automation, the certificate model became the assumed infrastructure. The structural properties of certificates, stored keys, hierarchical trust, fixed lifetimes, became invisible because the operational friction disappeared.

What keyless identity addresses

Keyless identity would provide web server identity without certificates, CAs, or stored key material. A server would prove its identity through accumulated behavioral continuity. Browsers would validate identity through trust slope verification rather than certificate chain validation. No CA infrastructure, no certificate lifecycle, no stored private keys.

Let's Encrypt demonstrated that the web benefits from ubiquitous identity verification. Keyless identity would extend that principle by making identity intrinsic to the server rather than dependent on certificates issued by an external authority.

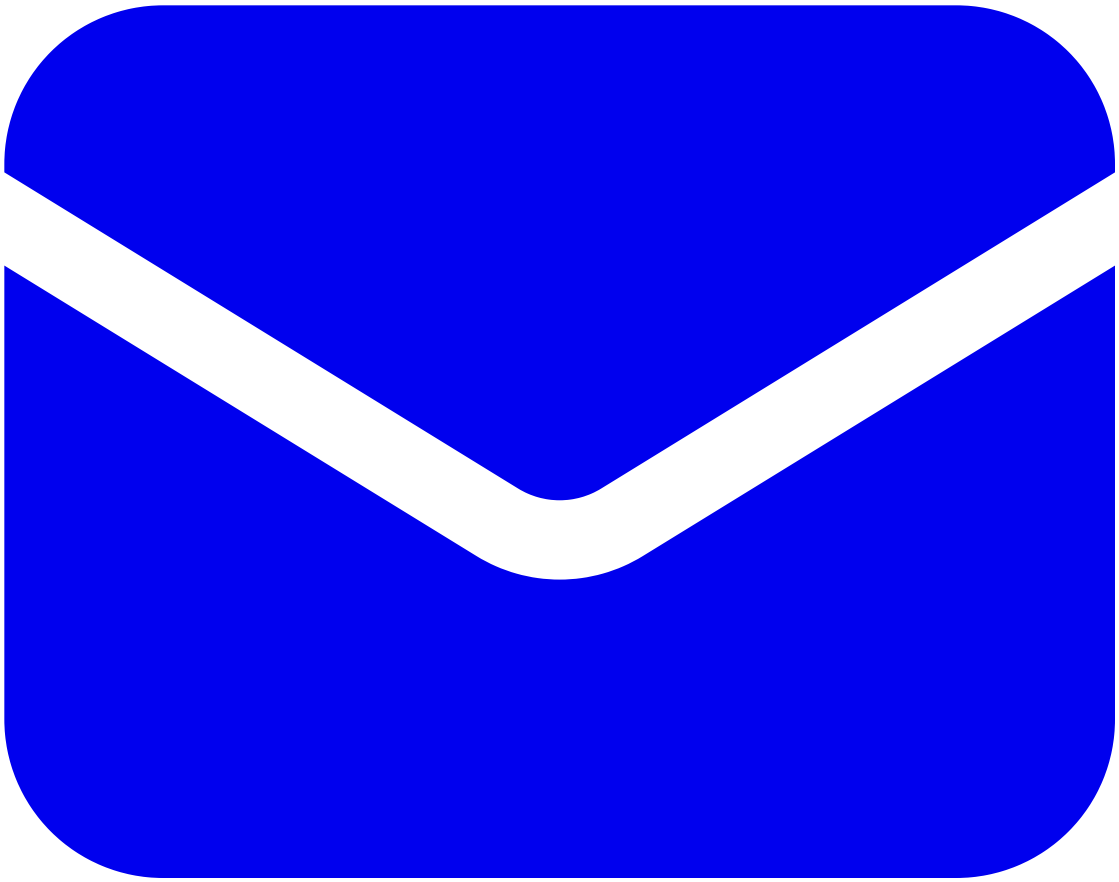
Keyless Identity All 21 steps →

Identity from accumulated continuity. Post-quantum by construction.

Patent
US 19/388,580 · published
Primary Technical Disclosure
○ Stateless Device Pseudonymity and Secure Messaging in Cognition-Native Systems
Secondary Technical
○ Continuity-Based Biological Identity Using Trust-Slope Validation○ Trust Slope as Identity Primitive: Cumulative Hash Chains Replace Static Credentials○ Dual-Source Identity Derivation: Hardware Anchors and Local State Vectors Combined Per Epoch○ Stateless Symmetric Encryption: Session Keys Derived From Current Identity State○ Two-Stage Message Authentication: Transport Continuity Screening Before Semantic Validation○ Agent-Substrate Slope Entanglement: Binding Every Mutation Step to Its Execution Host○ Append-Only Mutation Lineage Log: Forward-Secure Identity Transition Chains○ Cumulative Slope Validation Across Substrates: Multi-Node Provenance Verification○ Quorum-Based Identity Recovery: Peer Attestation After Memory Loss○ Entropy Anchor Rotation: Proactive Identity Reseeding With Forward Links○ Biometric-Assisted Reseeding: Privacy-Preserving Fuzzy Extractors for Anchor Rotation○ Delayed Slope Validation: Bounded Proof Windows for Disconnected Environments○ Sparse Trust Slope Recovery: Compact Checkpoints for Resource-Constrained Devices○ Predictive Identity Validation: Drift Detection Before Full Discontinuity○ Legacy PKI Fallback: Session-Scoped Adapters With Strict Isolation Boundaries○ Post-Quantum Alignment: Hash-Based Security Without Vulnerable Hardness Assumptions
Applications (General)
○ Trust Slope Entanglement: Cryptographic Lineage for Semantic Agents○ Post-Quantum Enterprise Identity Migration○ Billions of IoT Devices Need Authentication Without Keys○ Financial Identity Without Credential Databases○ Patient Identity Through Behavioral Continuity○ Supply Chain Authentication Without PKI○ Smart Building Access Through Continuity○ Vehicle Operator Identity Binding○ Displaced Person Identity Without Documents
Applications (Specific)
○ Okta Centralized Enterprise Identity. The Keys That Prove It Are Still Stored Somewhere.○ Auth0 Made Developer Identity Easy. The Credential Model Underneath Did Not Change.○ YubiKey Made Hardware Authentication Practical. The Key Is Still the Vulnerability.○ CLEAR Made Airport Identity Fast. It Built a Biometric Database to Do It.○ Worldcoin Scans Irises to Prove Humanity. The Proof Depends on a Central Enrollment System.○ Jumio Automated ID Verification. The Verification Still Depends on Documents.○ Microsoft Entra Unified Cloud Identity. Identity Still Depends on Stored Credentials.○ Ping Identity Built Enterprise Federation. The Federation Depends on Shared Secrets.○ OneLogin Simplified Enterprise SSO. The SSO Token Is Still a Credential.○ Duo Security Made MFA Ubiquitous. The Second Factor Is Still a Credential.○ Thales HSMs Protect Key Material. The Keys Still Exist.○ Entrust Issues Digital Certificates. The Certificate Is a Stored Credential.○ DigiCert Secures the Web With TLS Certificates. The Certificate Model Has Structural Limits.● Let's Encrypt Made TLS Free. The Certificate Model Is Still the Same.
Keyless Identity overview →
AQ
deterministic
autonomy

Legal

Last updated: 2026-03-03



-
- nick@qu3ry.net
- 72 28 14 36 01

Invented by Nick Clark | Founding Investors: Devin Wilkie