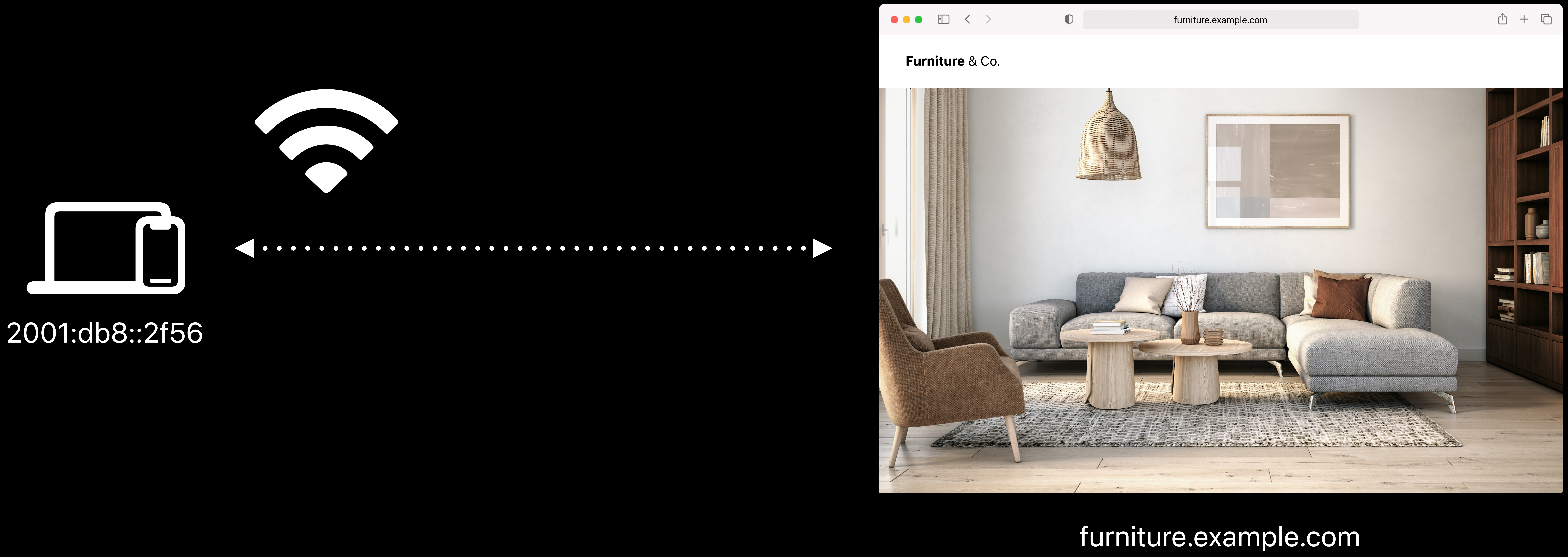
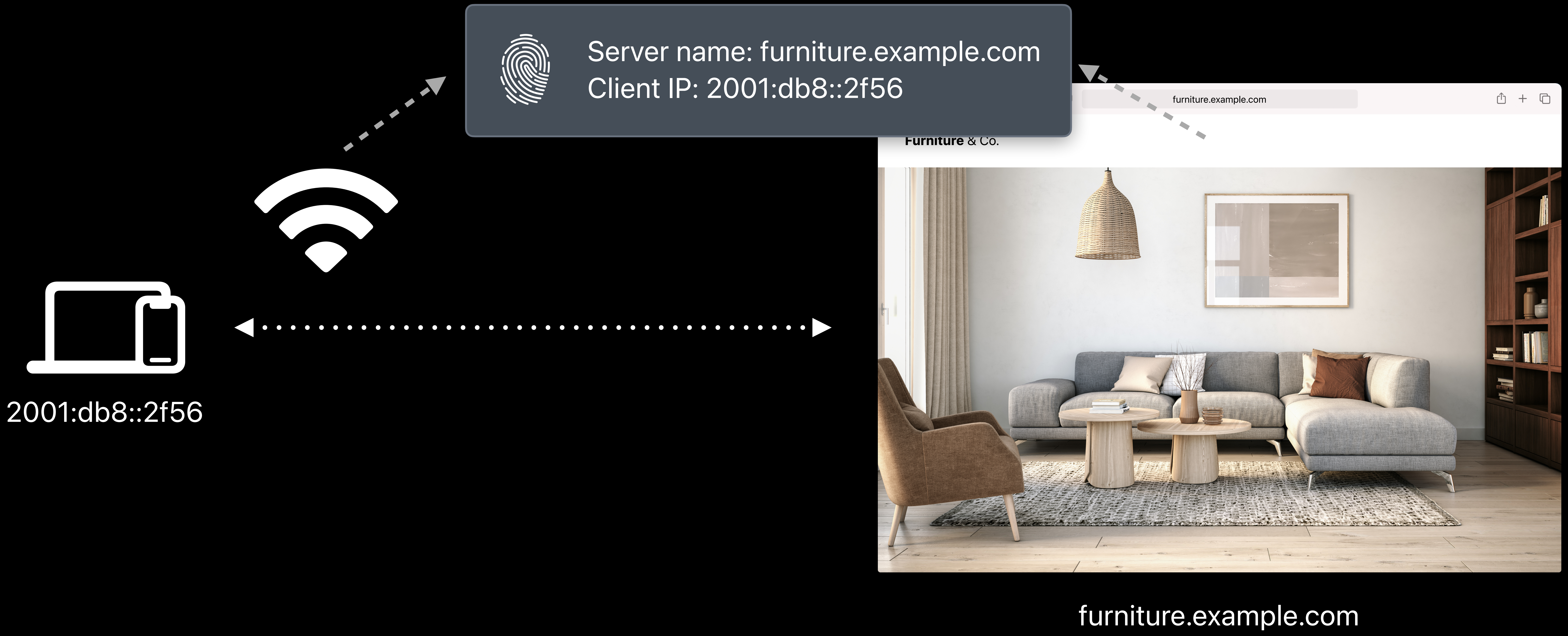
iCloud Private Relay

Multi-hop Internet privacy at scale

Real World Crypto 2023

Tommy Pauly, Apple
Christopher A. Wood, Cloudflare
Jana Iyengar, Fastly

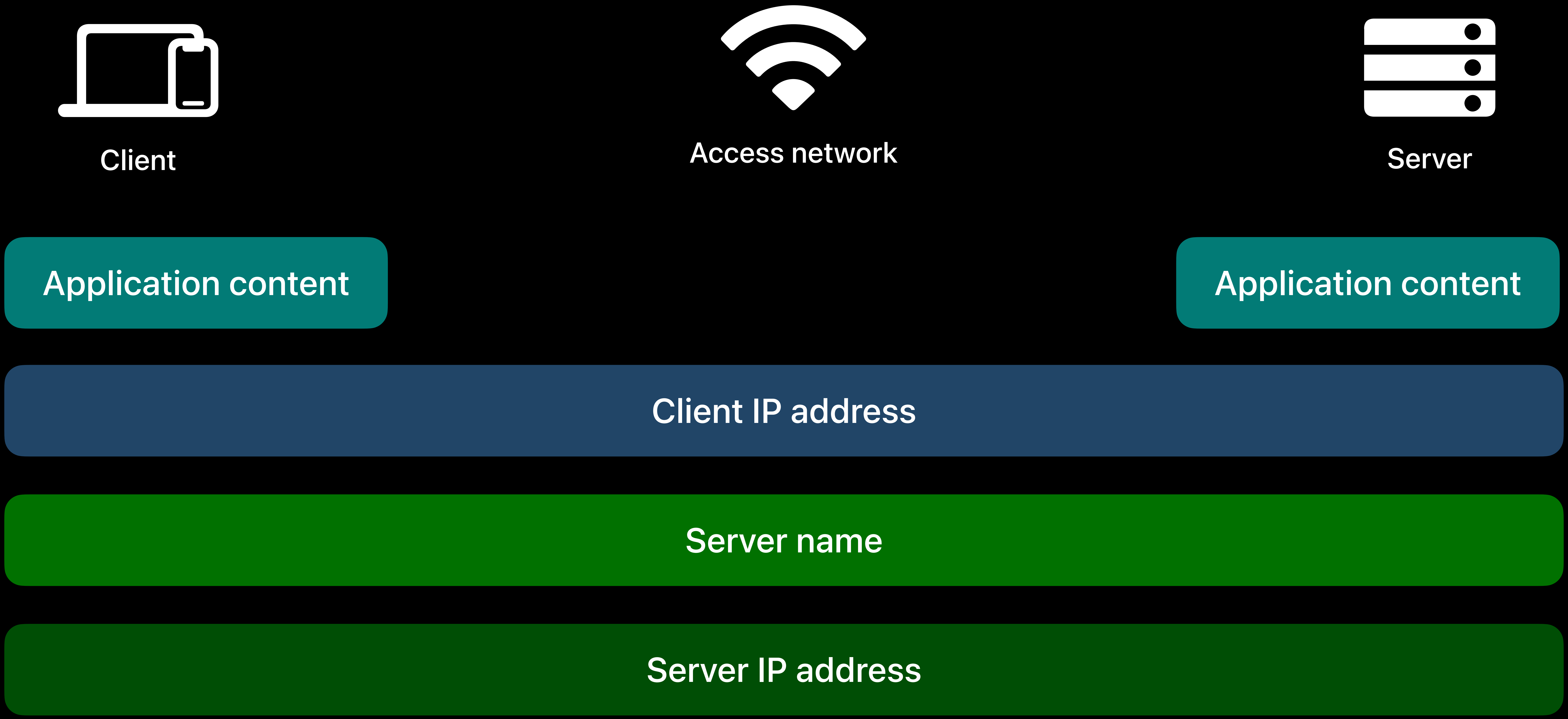

furniture.example.com
Furniture & Co.
2001:db8::2f56
furniture.example.com


Server name: furniture.example.com
Client IP: 2001:db8::2f56
furniture.example.com
Furniture & Co.
2001:db8::2f56
furniture.example.com

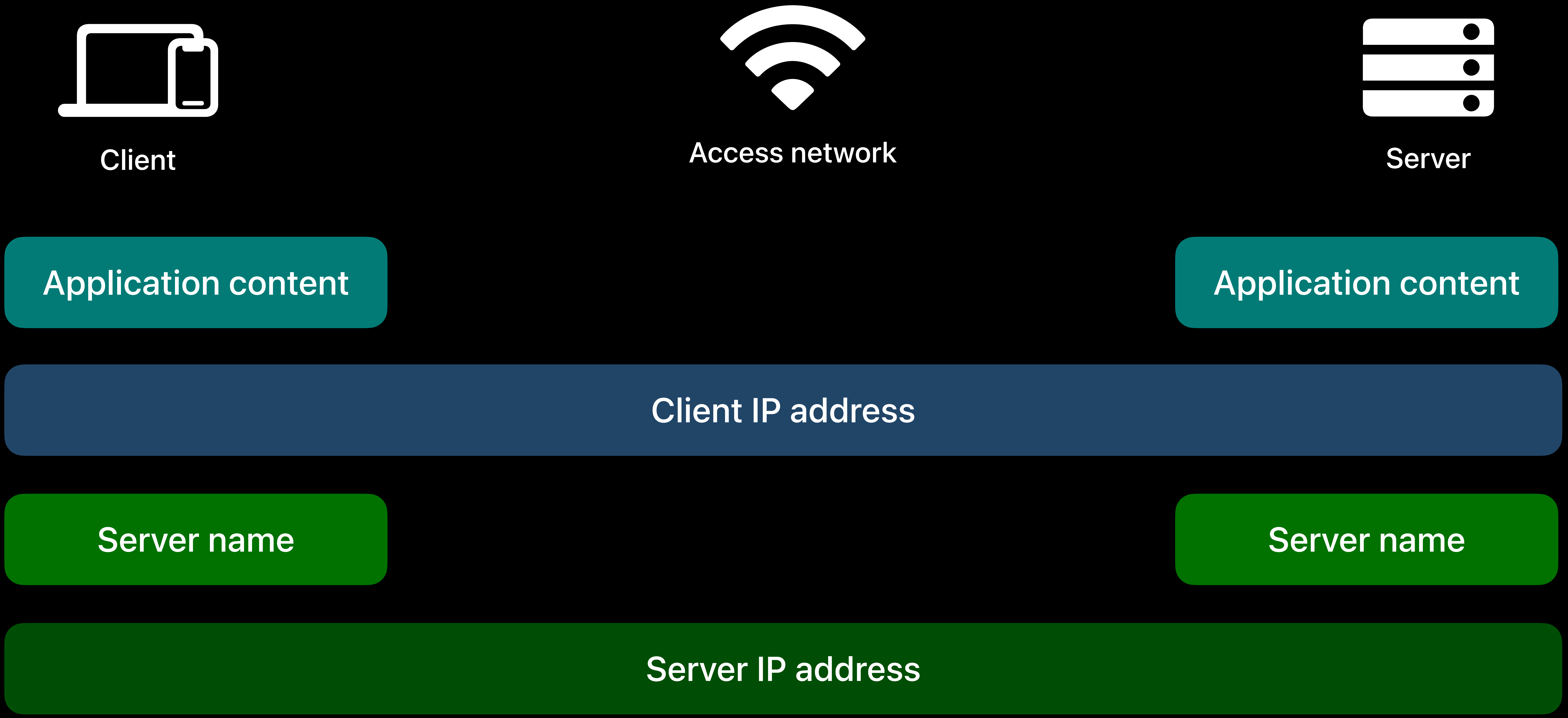
Protect user privacy by ensuring that
when someone uses the Internet,
no single party — not even Apple — can see
both who they are and what servers they access

What architecture achieves this goal?

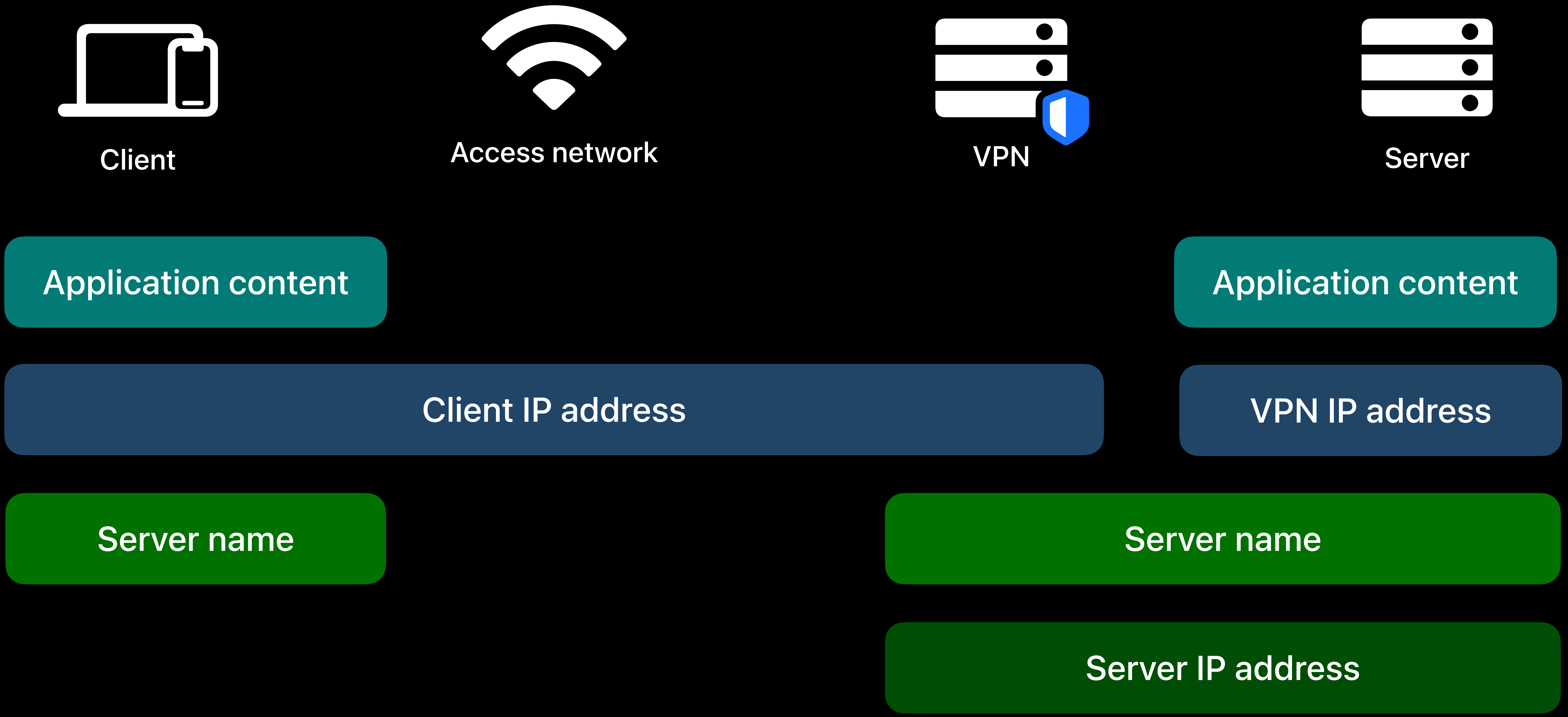
Typical HTTPS



Encrypted DNS, TLS ECH

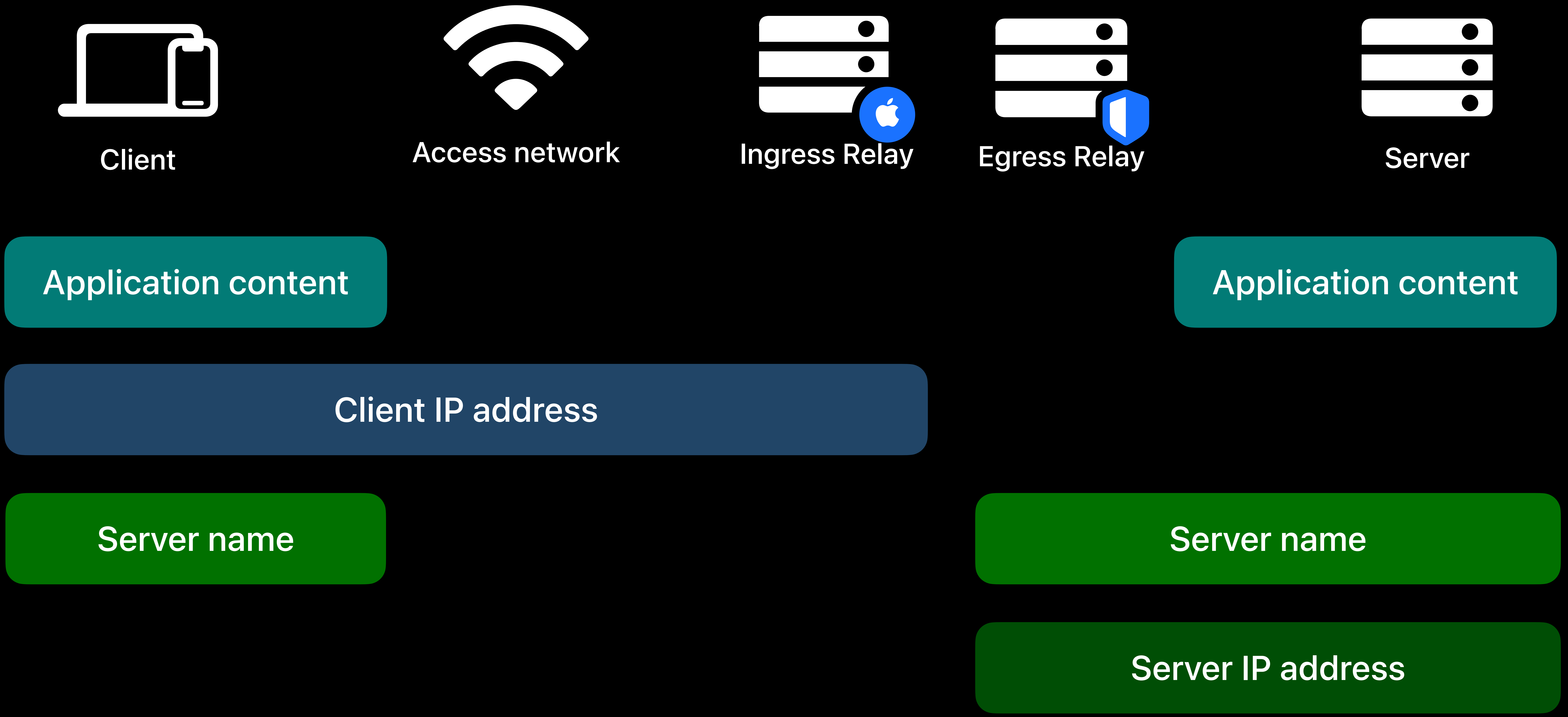


Typical VPN


Client
Access network
VPN
Server
Application content
Application content
Client IP address
VPN IP address
Server name
Server name
Server IP address

At least **two hops** are required to separate client and server identities

Private Relay



iCloud Private Relay

- Double-hop relay network
  - First hops are operated by Apple
  - Second hops are operated by multiple CDN partners
- Maintains general location

Traffic scope

- Relays can transport any TCP/UDP flows
- iCloud Private Relay
  - All Safari browsing traffic
  - All unencrypted HTTP traffic in all apps
  - DNS is protected using Oblivious DoH
- Default traffic in iOS and macOS
  - Third party web trackers in Safari
  - Remote content trackers in Mail

Which protocols best meet this goal?

Protocol requirements

- Efficiency across multiple hops
  - Avoid unnecessary round trips
  - Minimize encapsulation overhead
- Scalability for global traffic
  - Leverage mature stacks in CDNs
  - Widely-supported standardized protocols
- Minimize attack surface
  - Use a lightweight handshake protocol

MASQUE

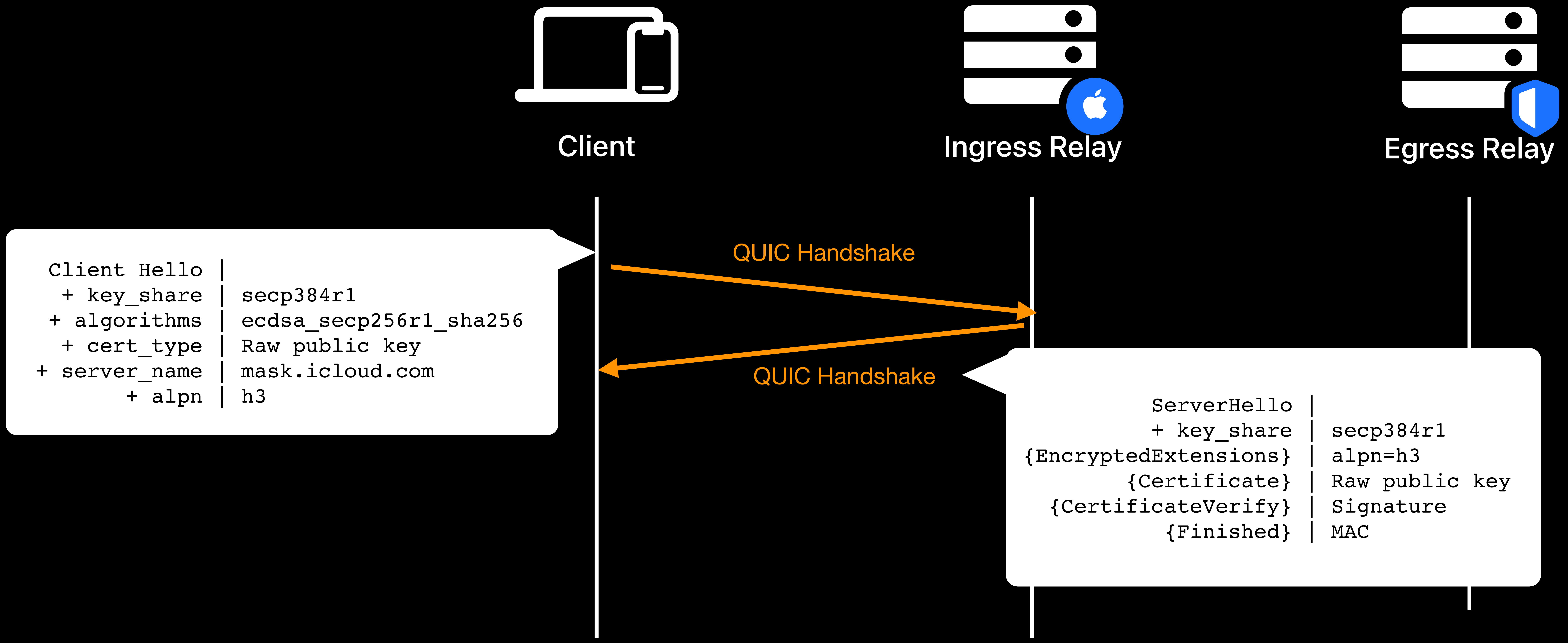
TLS

QUIC

HTTP

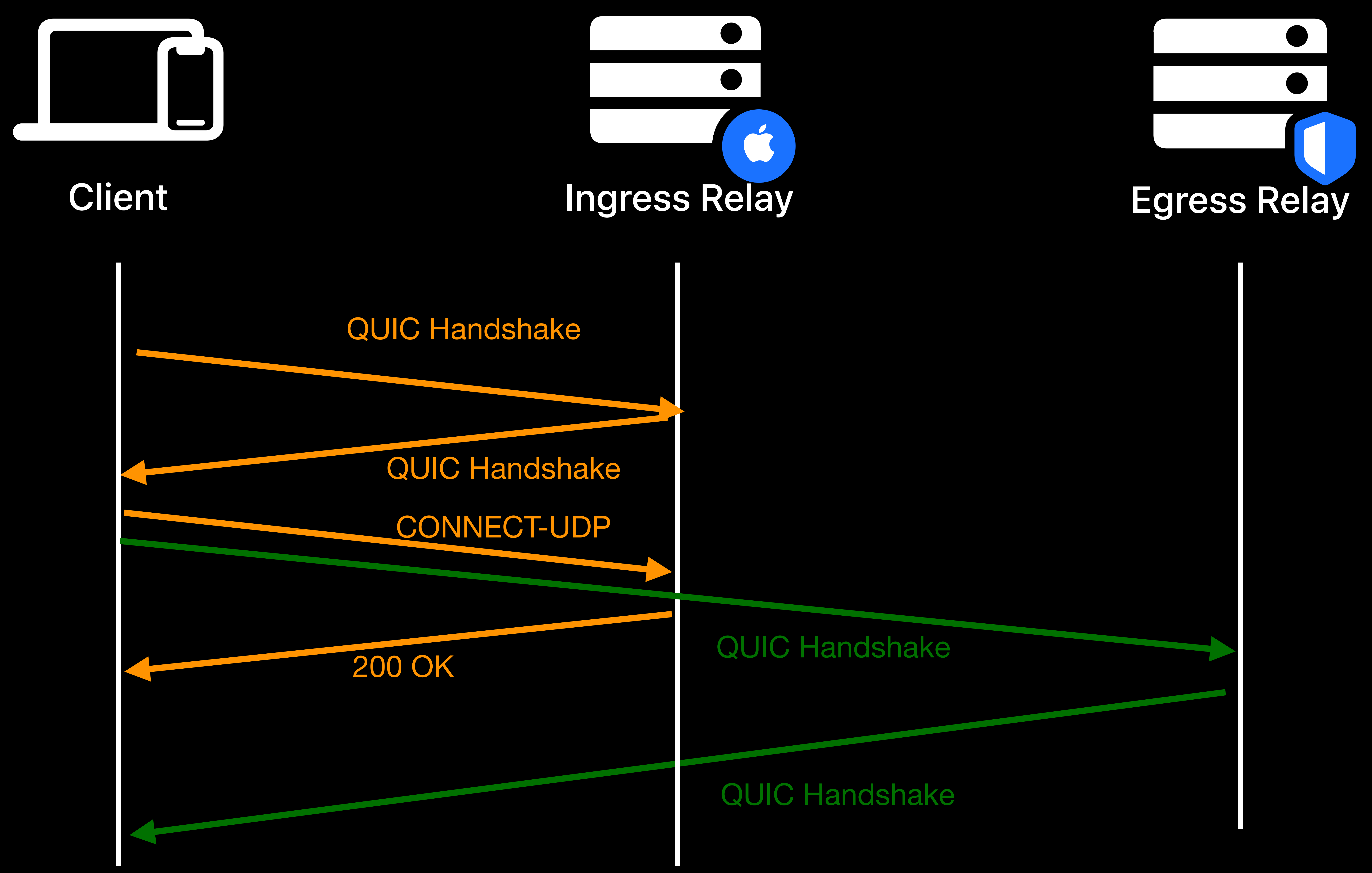
Minimal TLS 1.3

- Memory safe
  - Swift client implementation
- No downgrade allowed
  - Pinned to TLS_AES_256_GCM_SHA384
- Raw public key authentication
  - Minimize error-prone parsing bugs in X.509
  - Pinned keys

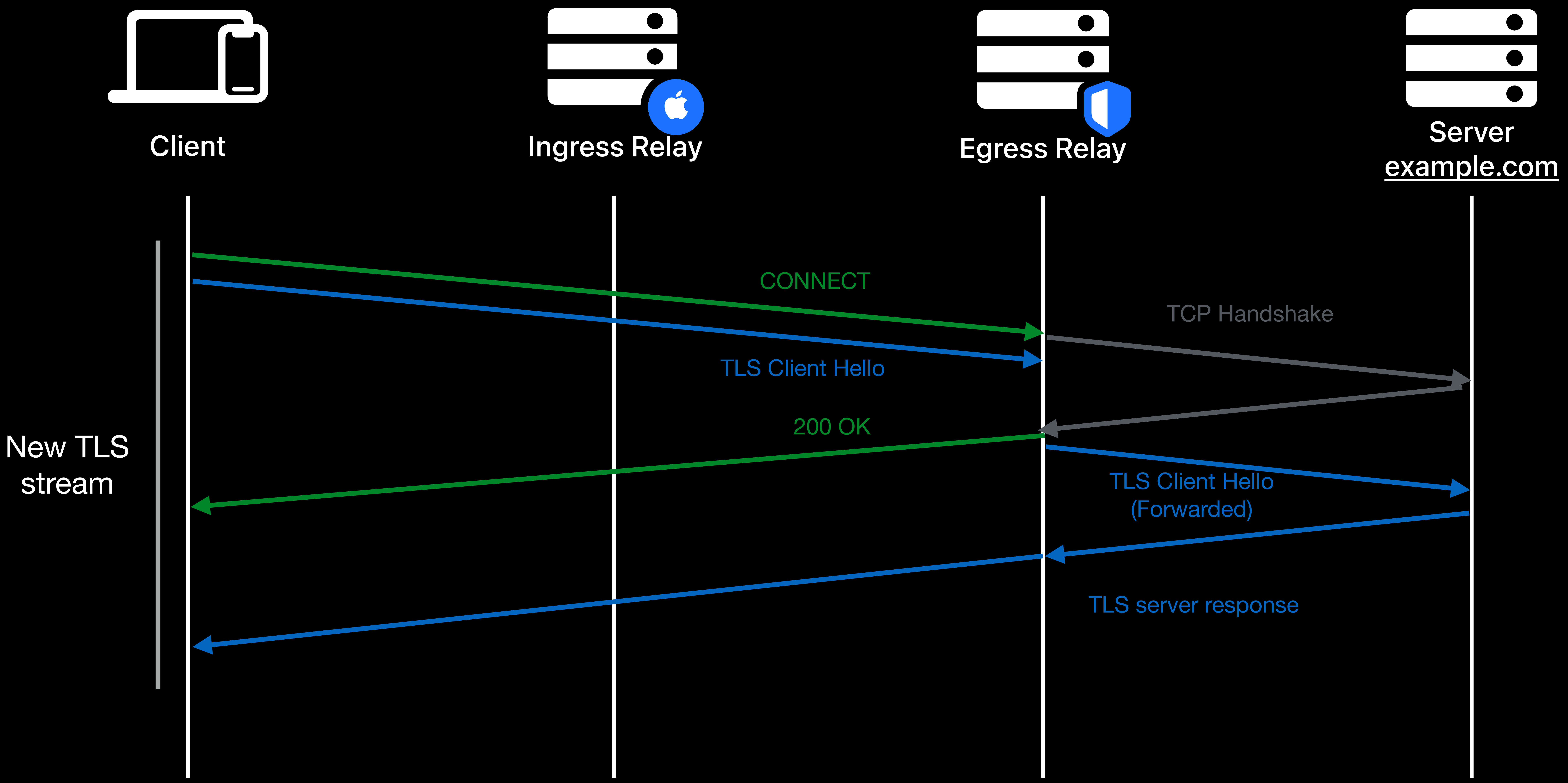

Client
Ingress Relay
Egress Relay
QUIC Handshake
QUIC Handshake
Client Hello |
+ key_share | secp384r1
+ algorithms | ecdsa_secp256r1_sha256
+ cert_type | Raw public key
+ server_name | mask.icloud.com
+ alpn | h3
ServerHello |
+ key_share | secp384r1
{EncryptedExtensions} | alpn=h3
{Certificate} | Raw public key
{CertificateVerify} | Signature
{Finished} | MAC

MASQUE relays

- HTTP/3 over QUIC forward proxies
  - Shared infrastructure and wire format with common web traffic
- Supported modes
  - CONNECT, for TCP next-hops
  - "CONNECT UDP", for QUIC and UDP next-hops (RFC 9298)
  - Oblivious HTTP Relay, for supported gateways

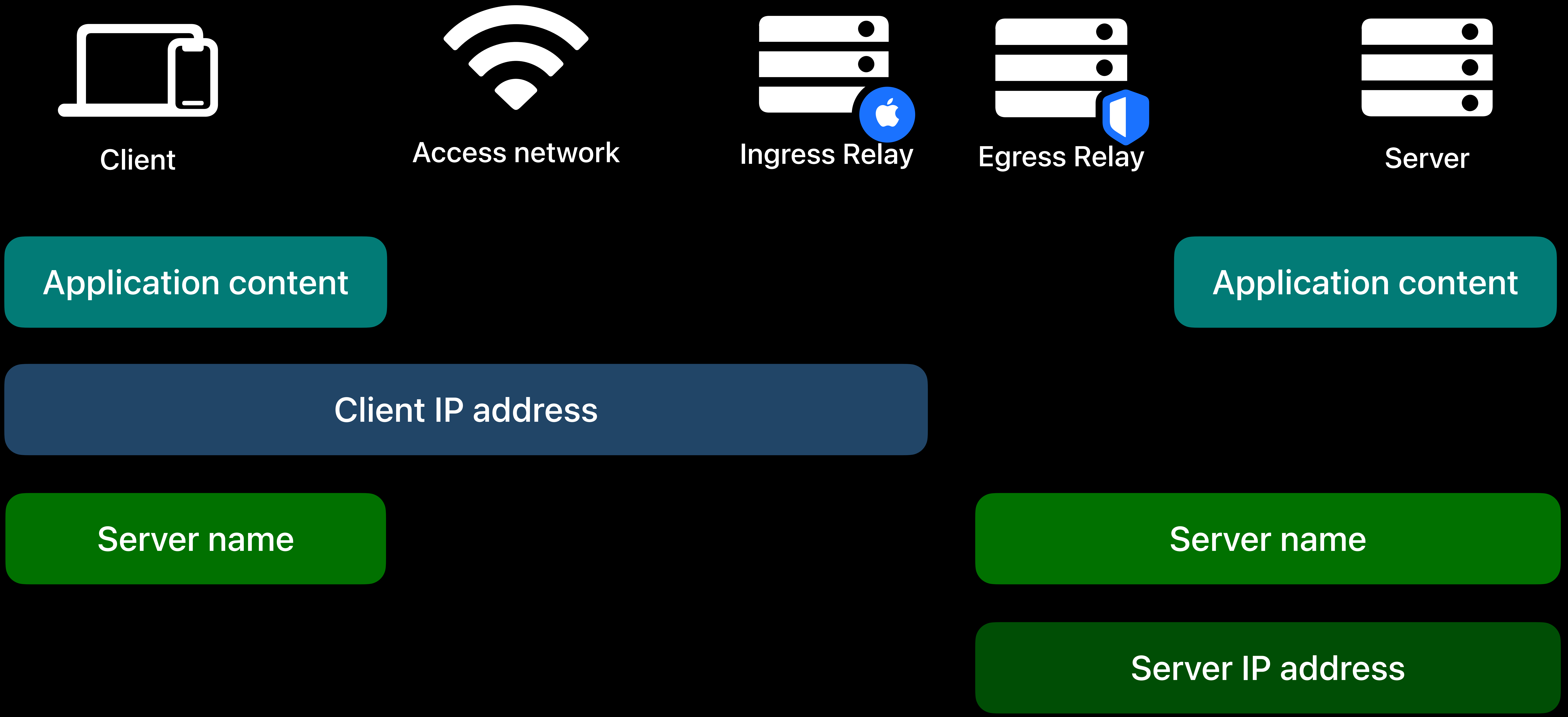


Client
Ingress Relay
Egress Relay
QUIC Handshake
QUIC Handshake
CONNECT-UDP
QUIC Handshake
200 OK
QUIC Handshake

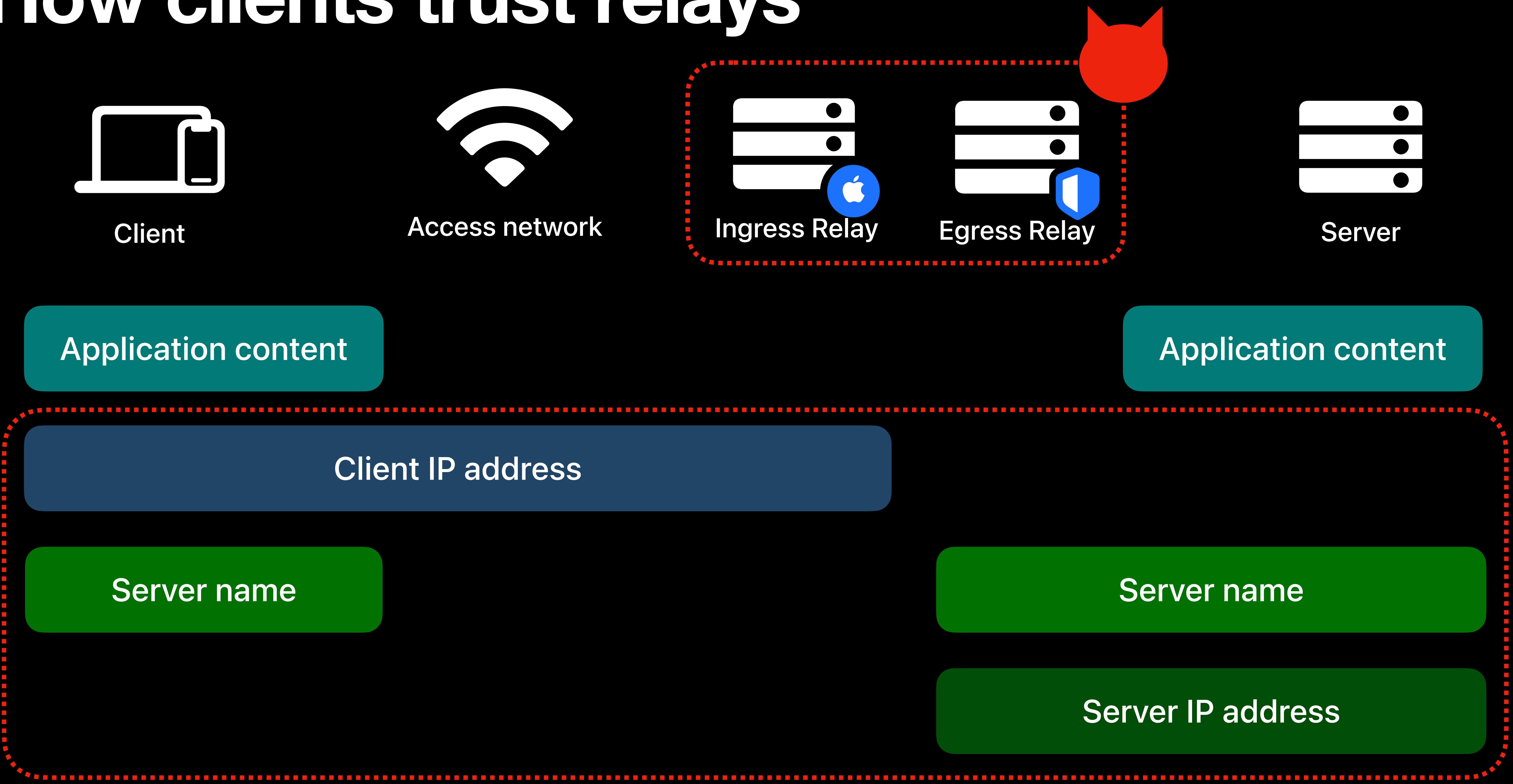

Client
Ingress Relay
Egress Relay
Server
example.com
New TLS stream
CONNECT
TLS Client Hello
TCP Handshake
200 OK
TLS Client Hello (Forwarded)
TLS server response

How do we prevent abuse?

How clients trust relays



How clients trust relays

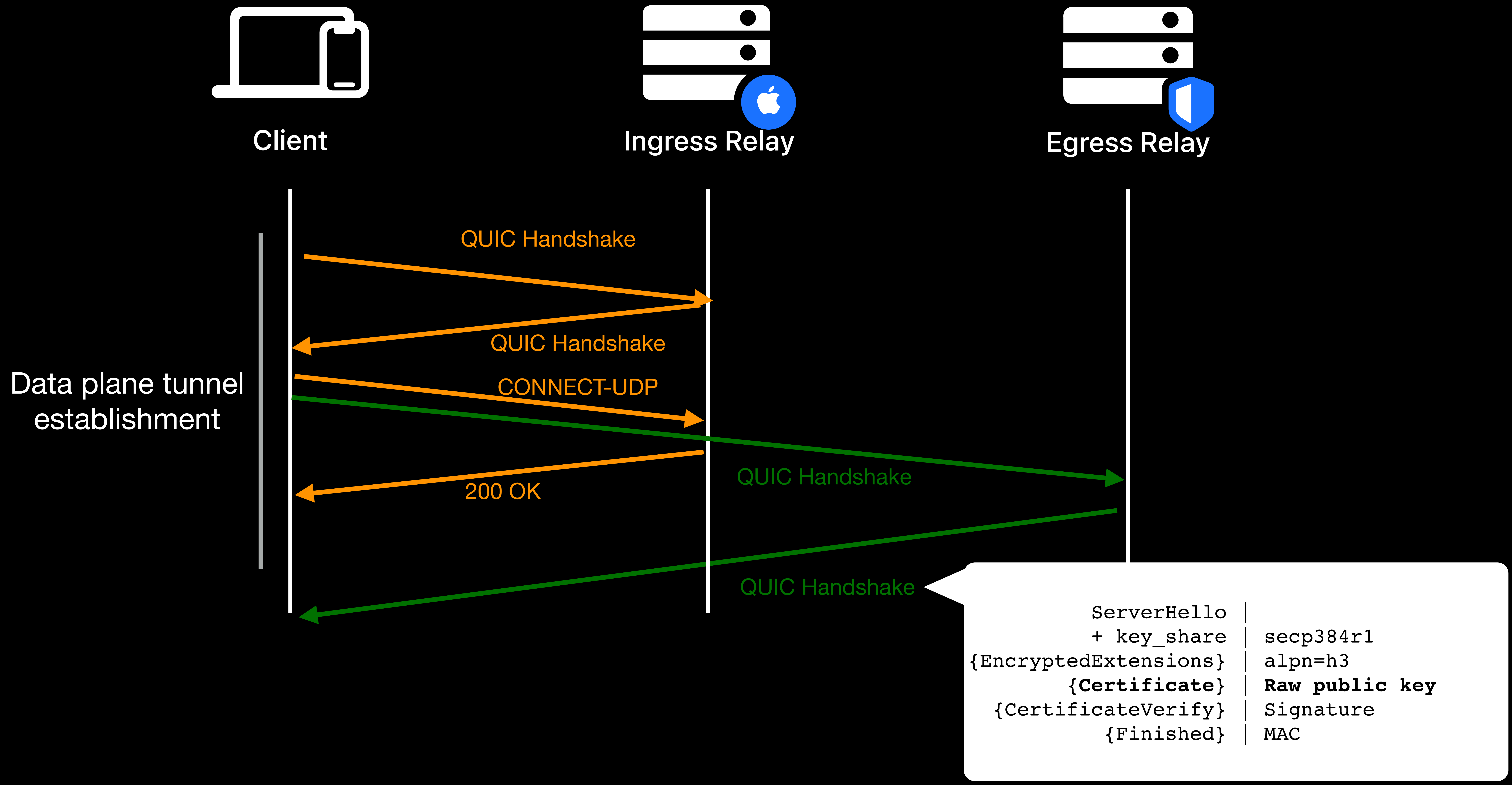


Relay authentication

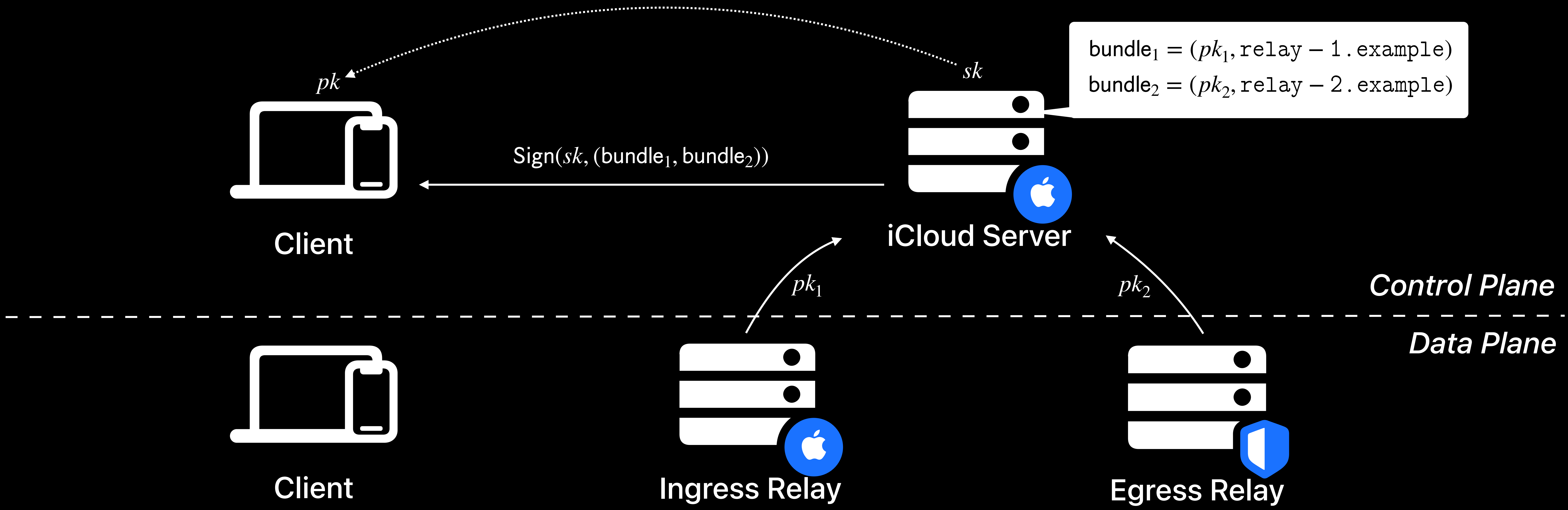
Security goal: Minimize X.509 dependencies in the data plane

Privacy goal: Ensure all clients get consistent authentication material for relays (so they can't be tagged and tracked)

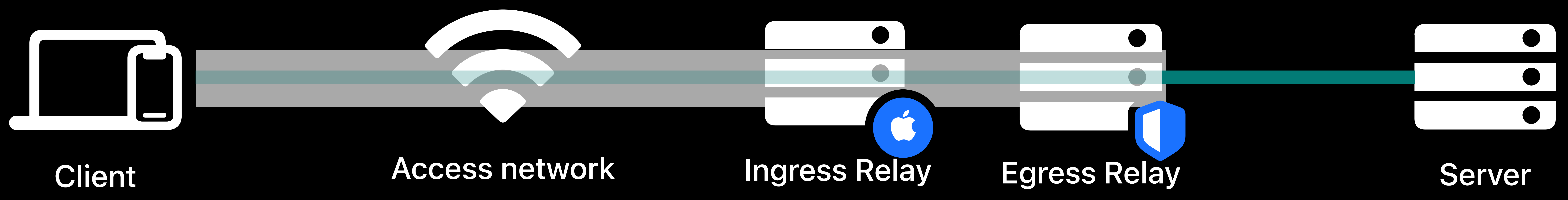
Tunnel establishment



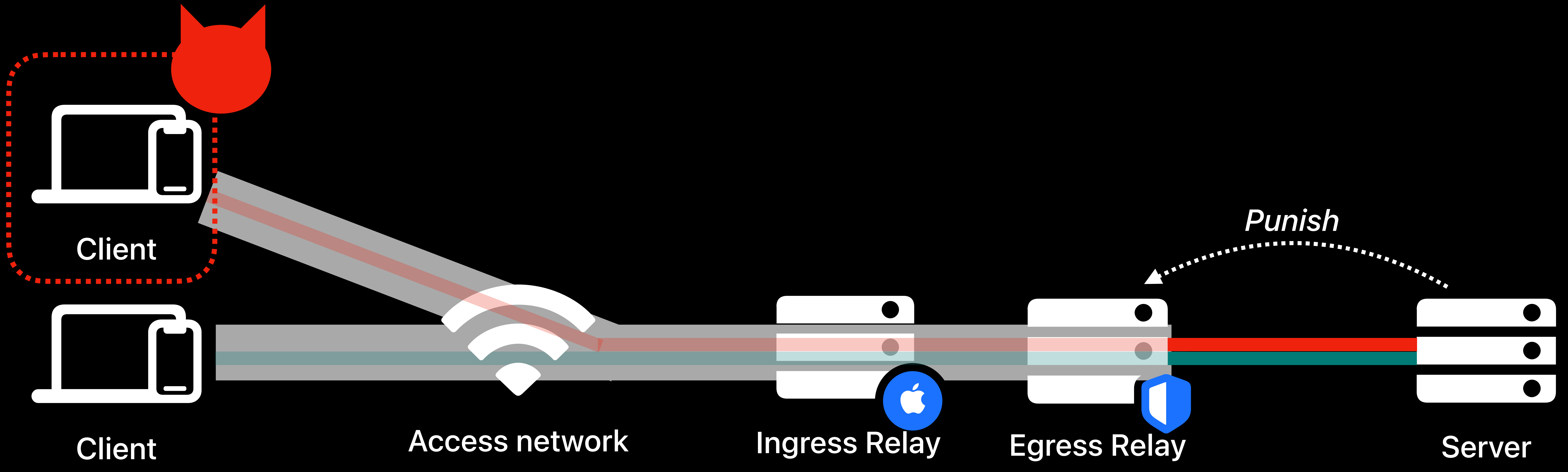
Authenticated key distribution



How relays trust clients



How relays trust clients

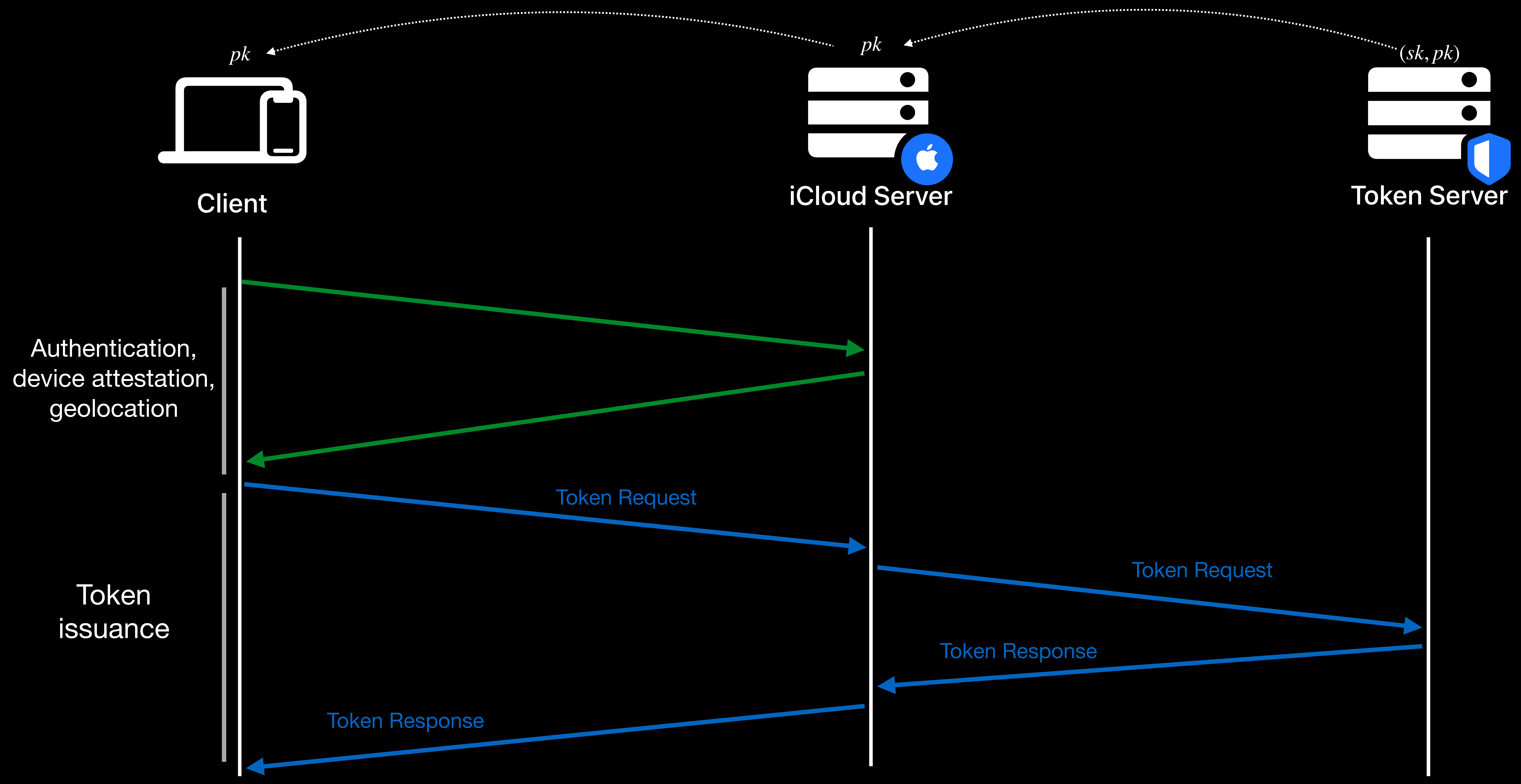


Client authentication

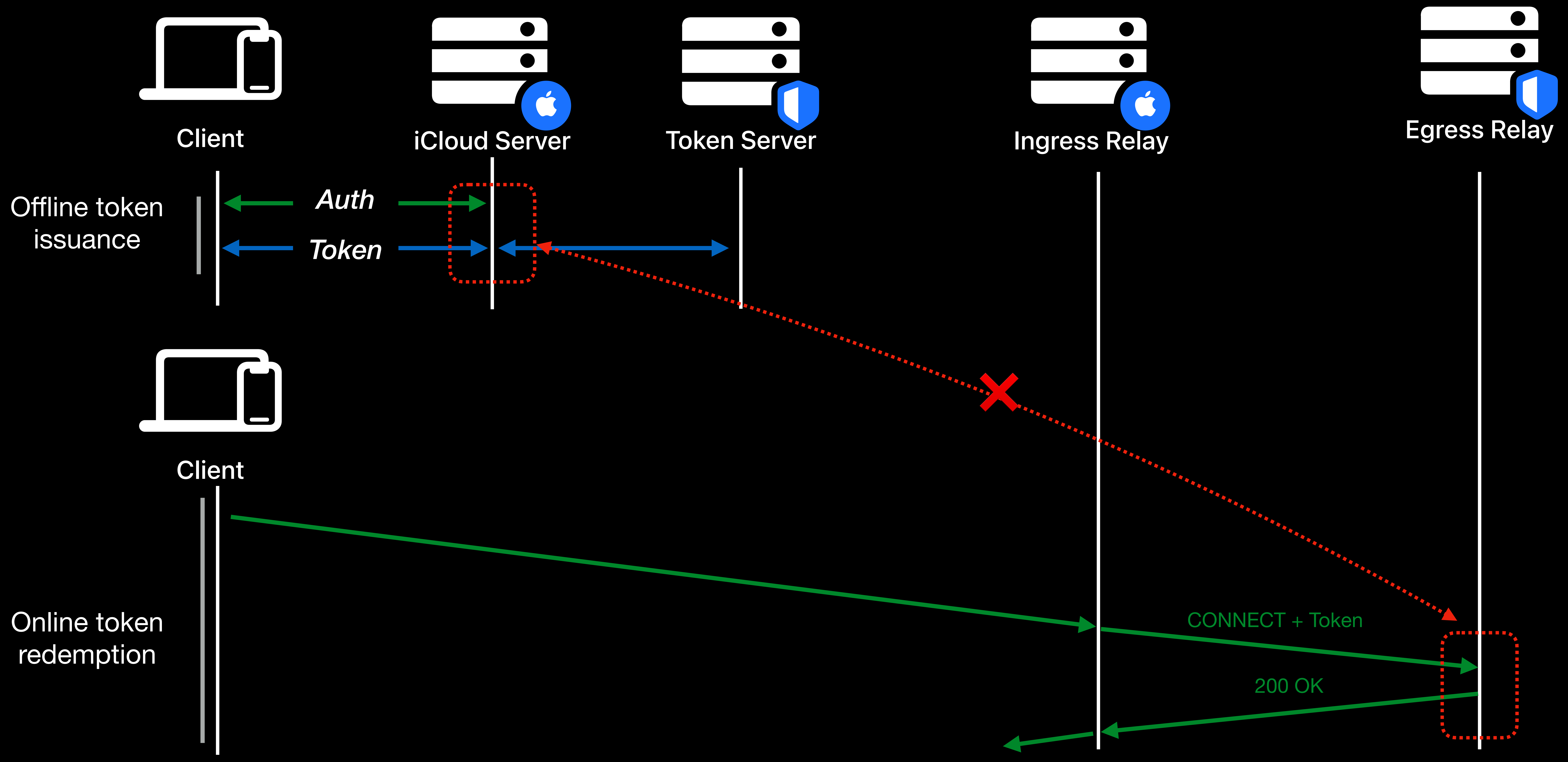
Security goal: ensure only *trusted* users can use the system

- Valid and up-to-date device
- Geo-based egress restrictions

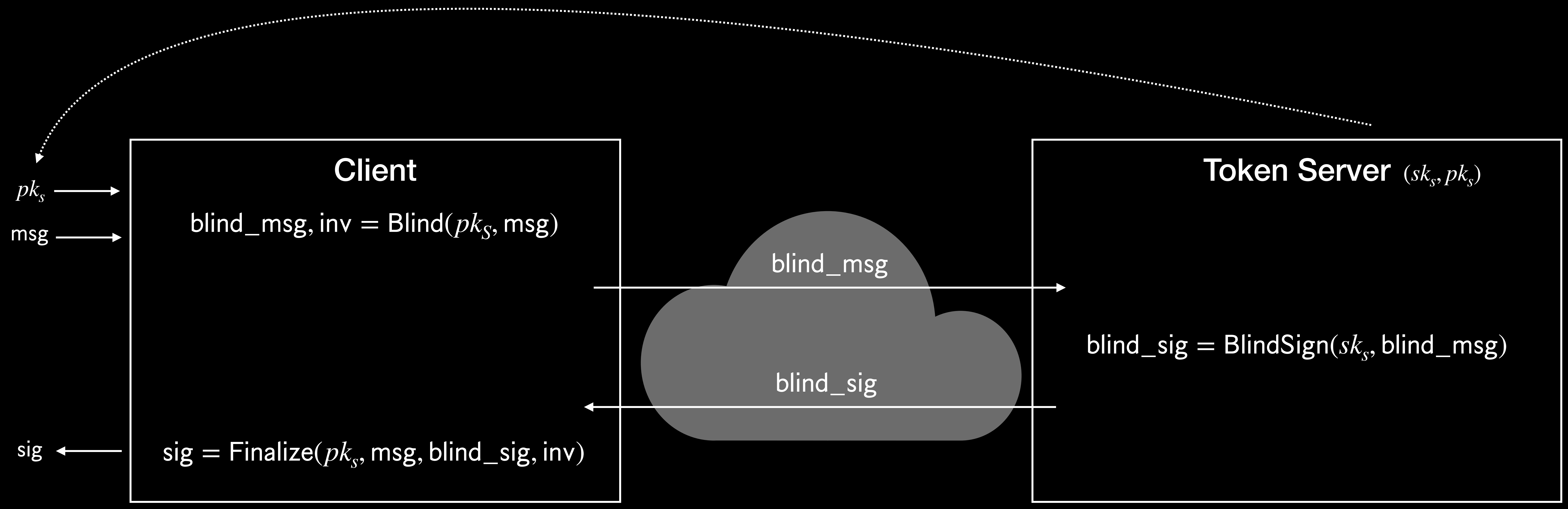
Privacy goal: Authentication material not tied to any individual client identifying information



pk
pk
(sk, pk)
Client
iCloud Server
Token Server
Authentication,
device attestation,
geolocation
Token
issuance
Token Request
Token Request
Token Response
Token Response


Client
iCloud Server
Token Server
Ingress Relay
Egress Relay
Offline token issuance
Auth
Token
Client
Online token redemption
CONNECT + Token
200 OK

Blind RSA (RSA-BSSA)



RSA-BSSA selection

- Explored elliptic-curve based blind signature protocols
  - Known protocols either required pairings (BLS) or involved signer state (Schnorr)
  - ROS assumption for Schnorr-based protocols was broken (2020/495)
- Blind RSA is comparatively robust, stateless, and widely understood
- PSS encoding lowered barrier to adoption but required additional analysis
  - Existing analysis gave confidence in FDH variants
- New analysis (2022/895) demonstrated RSA-BSSA with PSS was secure for Private Relay
  - Also highlighted sharp edges for blind RSA with malicious signers, but these do not apply to Private Relay

What is the impact?

IP address privacy is changing
the Internet ecosystem

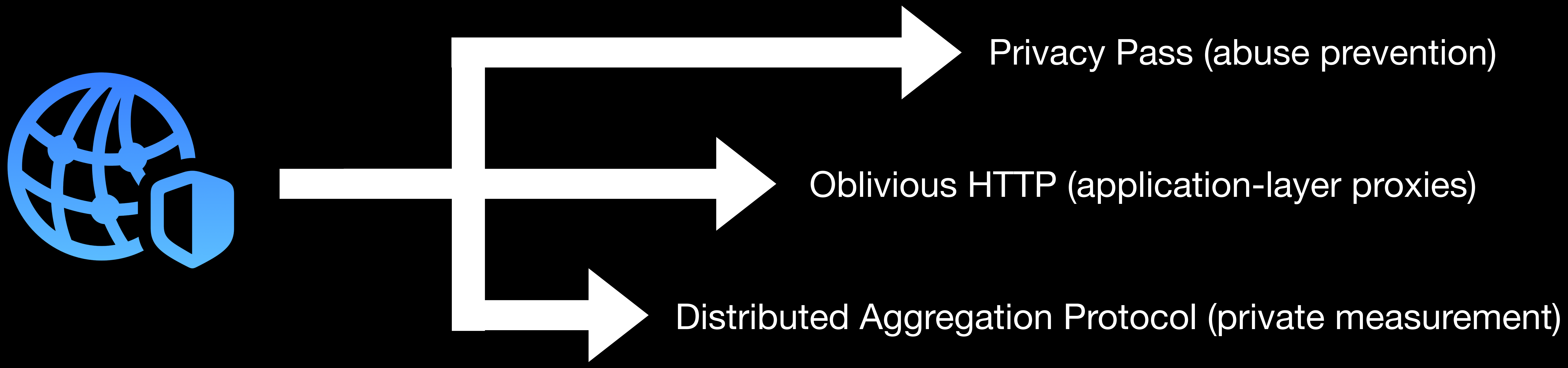
Solutions that were based on IP address tracking and state management need to adapt

A new generation of privacy-enhancing protocols are replacing previous mechanisms that relied on IP addresses

Ecosystem adaptation

	Status-quo	Mitigations with IP Privacy
Anti-abuse	IP address used as input to abuse detection	Origins use mechanisms like Privacy Pass
Geolocation	GeoIP databases identity locations	Relay egress IPs registered for regions globally, based on a rough location of the original client IP
State management	Some websites use IP addresses as state, instead of cookies	Relays maintain a (shared) egress IP for a browsing session

Emergent technologies



Discussion venues

Architecture and Data Plane Protocols

MASQUE
IETF

HTTP

QUIC

Authentication and Control Plane Protocols

TLS

PRIVACY PASS
IETF

Cryptographic Protocols, Analyses, and Verification

Crypto Forum Research Group

RWC / HACS