A Formal Cryptanalysis of the Authenticated Encrypted Relay Network Protocol

John G. Underhill

Quantum Resistant Cryptographic Solutions Corporation
contact@qrcscorp.ca

May 22, 2026

Abstract

The Authenticated Encrypted Relay Network (AERN) is a certificate-anchored, post-quantum relay protocol for controlled anonymity and transport domains. The protocol combines an ARS root trust anchor, an ADC topology authority, authenticated post-quantum tunnel establishment, pre-established APS tunnel state, fixed-size relay packets, encrypted routing state, explicit ingress-to-egress relay sessions, encrypted fragmentation metadata, backend-neutral transport callbacks, strict per-hop sequencing, bounded dummy relay traffic, and optional randomized ingress delay. This paper gives a formal cryptanalysis of the protocol as represented by the AERN specification and reference implementation. The analysis develops game-based definitions for certificate and control-message authentication, tunnel-key indistinguishability, weak forward secrecy, conditional post-compromise recovery, authenticated tunnel confidentiality and integrity, strict replay resistance, route-path confidentiality, relay-session correctness, fragment reassembly integrity, backend-boundary separation, and conditional anonymity under partial compromise. The model is deliberately conservative. AERN is analyzed as a low-latency controlled-domain relay system, not as a high-latency mix network. The results show that, under the stated primitive assumptions and implementation constraints, AERN provides authenticated node admission, confidentiality and integrity for per-hop tunnel packets, encrypted relay metadata, strict replay rejection, direction separation for return traffic, and conditional route unlinkability against adversaries that do not simultaneously control or observe the relevant ingress and egress correlation points.

Contents

1 Introduction

1.1 Background and Objective

Anonymity and relay systems are intended to reduce the ability of an observer to associate a client with the destination or service reached through a network. Classical mix and onion-routing systems achieve this objective through relay composition, fixed or padded message formats, and controlled disclosure of path state across relays [11–14]. AERN uses a different deployment model. It is designed for controlled relay domains in which all relay nodes are authenticated, topology membership is administered, certificate status is enforced, and the data phase uses fixed-size authenticated tunnel packets over symmetric APS-to-APS channels.

The purpose of this paper is to analyze the cryptographic and protocol properties of AERN using the AERN technical specification and public source code as the protocol reference [1, 2]. The analysis treats the ARS, ADC, APS, client, and backend transport boundary as distinct protocol actors. It models the relay packet structure directly: a 1500 byte wire packet, a 22 byte authenticated outer header, a 1478 byte ciphertext region, a 1446 byte relay plaintext after authentication and decryption, a 2 byte encrypted length prefix, a 16 byte encrypted route path, a 32 byte encrypted relay payload header, and a variable relay body padded to the fixed packet size.

AERN is not modeled as a public volunteer anonymity network. It is modeled as an authenticated infrastructure protocol in which node admission, topology synchronization, revocation, tunnel establishment, relay forwarding, and backend handoff are controlled by the certificate and topology state distributed by the ARS and ADC. This distinction determines the security target. The protocol aims to provide authenticated participation, encrypted route and session metadata, strict replay rejection, and practical resistance to traffic analysis within a managed low-latency domain. It does not claim to eliminate all timing leakage against a fully global adversary.

1.2 Protocol Summary

The AERN Root Security server, denoted ARS, signs root and child certificates. Each device generates its own signing key pair and certificate request. The secret signing key remains with the originating device. The ARS root signature binds the device verification key, issuer, serial number, expiration interval, designation, algorithm identifier, and version. The AERN Domain Controller, denoted ADC, manages registration, topology, revocation, convergence, and signed administrative messages. APS nodes form the proxy relay mesh. Clients establish an authenticated encrypted tunnel to an entry APS and submit serialized traffic into the relay

system.

APS-to-APS tunnels are derived through an authenticated post-quantum key exchange. The shared secret is expanded through a SHAKE-based key derivation function into directional transmit and receive cipher material. Tunnel packets carry a visible but authenticated outer header. The 22-byte header serializes a one-byte packet flag, a four-byte little-endian ciphertext length, an eight-byte sequence number, an eight-byte UTC timestamp in seconds, and a trailing reserved byte fixed to zero ($1 + 4 + 8 + 8 + 1 = 22$). The entire header is bound as associated data by the tunnel authentication function. The route path, relay payload header, session identifiers, packet identifiers, fragment metadata, payload type, return flag, and backend payload bytes are inside the encrypted relay plaintext.

Relay sessions are explicit. The ingress APS creates a pending relay session when a client flow requires egress delivery. It sends a session-open payload to the selected egress APS and queues non-dummy data until an authenticated session-open acknowledgement is received. The egress creates an active egress-side session entry only after validating the session-open payload. Return traffic uses the same logical session with the return flag set and is delivered through the ingress callback after authentication, session validation, and reassembly when required.

1.3 Contributions

This paper provides the following analysis results.

1. It defines a formal model for AERN entities, certificate state, topology state, APS tunnels, relay packets, route paths, relay sessions, fragmentation, dummy traffic, and backend callbacks.
2. It specifies the active packet layout and all derived capacity values: 1500 byte relay wire packet, 22 byte outer header, 1478 byte ciphertext region, 1446 byte relay plaintext, 16 byte encrypted route path, 1428 byte relay content region, 32 byte encrypted relay payload header, and 1396 byte maximum fragment payload.
3. It models one byte route hints as one-based APS topology ordinals. The value zero is reserved as a route terminator. A single active route domain is therefore limited to 255 route-addressable APS ordinals by this encoding.
4. It gives game-based definitions and reductions for authentication, key indistinguishability, weak forward secrecy, conditional recovery after rekey, per-hop tunnel security, strict replay resistance, session correctness, fragment integrity, route-path confidentiality, and conditional anonymity.
5. It defines the backend transport callback as a proof boundary. The relay core authenticates, decrypts, validates, reassembles, classifies, and delivers

serialized packets to the boundary. The security of the external network device, socket stack, destination, or application transport is outside the cryptographic relay proof.

6. It identifies implementation conformance requirements that must be validated by test vectors and virtual-network execution.

2 Engineering Description of AERN

2.1 Certificate-Anchored Trust Architecture

AERN uses a hierarchical certificate model. The ARS signs child certificates and serves as the trust anchor for the domain. Each child certificate binds a device public verification key to an issuer string, serial number, root serial reference, validity interval, designation, algorithm identifier, and version. Certificate validation requires a valid ARS signature, matching root serial reference, valid expiration interval, accepted algorithm and version fields, and consistency with topology state when a topology node is available.

The ADC is a certified control authority. It validates device certificates, distributes topology lists, signs topology and administrative messages, handles registration and revocation, and may proxy certificate signing requests to an isolated ARS. The ADC signature does not replace the ARS root signature for device certificates. It authenticates control messages and topology state within the AERN domain.

The separation between signing keys and tunnel keys is central. Certificates authenticate identities and messages. They do not contain data-plane encryption keys. Tunnel keys are derived from authenticated KEM exchanges and are scoped to the peer tunnel and direction labels. Compromise of a signing key affects future authentication but does not by itself compute completed tunnel secrets after ephemeral key material and active symmetric state have been erased.

2.2 Topology and Route Hints

The ADC maintains an authenticated topology. A topology node includes address information, certificate hash, issuer, serial number, expiration interval, and role designation. APS nodes use the synchronized topology order to resolve route hints. A route hint is a one byte, one-based APS ordinal. Valid route hints are values in the set $\{1, \ldots, n\}$ for an active route-addressable APS set of size $n \leq 255$. The value zero is reserved and does not identify an APS node.

The one byte hint representation makes route metadata compact and increases usable relay body capacity. It also defines a hard addressability boundary for a single route domain. A deployment with more than 255 physical APS nodes must

either restrict a particular active route domain to 255 route-addressable ordinals or use a separately specified wider hint profile. The security statements in this paper use the effective authenticated and route-addressable APS set as the anonymity set.

2.3 Tunnel Establishment and Key Schedule

AERN tunnel establishment is an authenticated KEM-based exchange bound to device certificates, timestamps, sequence metadata, and role context. The exchange may be instantiated with the post-quantum KEM and signature algorithms configured for the implementation, including the Kyber/ML-KEM and Dilithium/ML-DSA families [7–10]. The analysis assumes IND-CCA2 security for the KEM, EUF-CMA security for the signature scheme, pseudo-randomness of the SHAKE-based KDF, and authenticated encryption security for the tunnel cipher construction [4–6].

Let ss be the shared secret obtained from the authenticated KEM exchange, and let ctx be a transcript context containing local and remote identities, serial numbers, algorithm identifiers, direction labels, and exchange metadata. The derived state is represented as

$$K_{tx} \parallel K_{rx} \parallel N_{tx} \parallel N_{rx} = \mathsf{KDF}(ss, ctx).$$

Direction labels ensure that the transmit state of one endpoint corresponds to the receive state of the peer and that keys are not reused with the same direction semantics. Each tunnel direction maintains independent sequence state.

2.4 Fixed-Size Relay Packet Format

AERN relay packets are fixed at the MTU-sized relay unit of 1500 bytes. The packet is represented as

$$Pkt = Hdr_{out} \parallel C_{rel}.$$

The outer header Hdr_{out} is visible to the adjacent tunnel peer and authenticated as associated data. The ciphertext region C_{rel} contains authenticated encrypted relay plaintext and authentication material. The decrypted relay plaintext is represented as

$$P_{rel} = L_{16} \parallel R_{16} \parallel RHdr_{32} \parallel B \parallel pad.$$

Here L_{16} is a two byte little-endian field encoding the used relay payload length after the route path, R_{16} is the sixteen byte encrypted consumed-path route array, $RHdr_{32}$ is the encrypted relay payload header, B is the relay body, and pad fills the remaining plaintext region.

Component	Size	Meaning
Relay wire packet	1500 bytes	Fixed relay transmission unit
Outer tunnel header	22 bytes	Visible authenticated header
Ciphertext region	1478 bytes	Authenticated encrypted tunnel payload and tag
Tunnel authentication tag	32 bytes	Authentication material in ciphertext region
Relay plaintext	1446 bytes	Decrypted relay content after authentication
Length prefix	2 bytes	Used relay payload length after route path
Route path	16 bytes	Encrypted consumed-path route array
Relay content region	1428 bytes	Region after length prefix and route path
Relay payload header	32 bytes	Encrypted session and fragment metadata
Data payload per fragment	1396 bytes	Region after relay payload header

Table 1: AERN relay packet size model.

The size model in Table 1 corresponds to the default conformance profile in which the tunnel authentication tag occupies 32 bytes (the RCS authenticated profile with `AERN_CRYPTO_SYMMETRIC_MAC_SIZE`$= 32$). The derived capacities are then $1478 - 32 = 1446$ for the relay plaintext, $1446 - 2 - 16 = 1428$ for the relay content region, and $1428 - 32 = 1396$ for the per-fragment data capacity. These values are not magic constants; they are computed from the tag size by the relations

$$|P_{rel}| = |C_{rel}| - \tau, \quad |B_{max}| = |P_{rel}| - 2 - 16 - 32,$$

where τ is the tunnel tag length. The extended-security profile selects $\tau = 64$, which reduces the relay plaintext, content region, and fragment capacity by 32 bytes each while leaving the wire packet size, outer header, route-path size, and relay-payload-header size unchanged. All numeric results in this paper are stated for $\tau = 32$; the analysis transfers unchanged to $\tau = 64$ with the substituted capacities, because no security argument depends on the specific value of τ beyond the assumption that the tag length is sufficient for the claimed authentication-forgery bound.

The tunnel encryption function is modeled as authenticated encryption with associated data:

$$C_{rel} = \mathsf{Enc}_K(P_{rel}; AAD = Hdr_{out}).$$

The receiver accepts a relay plaintext only if authentication succeeds and the outer header satisfies the flag, ciphertext length, sequence, and timestamp checks. Route state, relay session state, fragment state, and backend state are not modified before these checks pass.

2.5 Encrypted Consumed-Path Route Array

The route path is a sixteen byte encrypted array. Entry zero identifies the route origin or local path start. Entries one through fifteen contain ordered future-hop hints. The first nonzero future-hop hint identifies the next APS peer. A forwarding APS consumes that hop by setting the corresponding array slot to zero, reserializes the route path, and forwards the packet under the tunnel state shared with the next APS. If no future-hop slot is nonzero, the packet is terminal for the traversal direction.

Field	Size	Meaning
path[0]	1 byte	Origin or local path-start hint
path[1..15]	15 bytes	Ordered future-hop hints
hopcount	local state only	Generation-time route count, not serialized

Table 2: Serialized AERN route path carried inside the encrypted relay plaintext.

The route generator requires at least three APS nodes for a complete route. It selects a hop count between the configured minimum and maximum bounds, constrained by the active APS count and the route path capacity. It sets the origin hint in $path[0]$, samples intermediate hints subject to validity constraints, and places the target hint in the final active path position. Interior selection excludes zero, the origin hint, the target hint, and the immediately preceding hint. Consecutive duplicate route hints are invalid. Non-consecutive repetition is not required by the route validation rule, although the generation rule excludes origin and target from the interior positions.

2.6 Encrypted Relay Payload Header

The relay payload header is a 32 byte encrypted structure following the route path. It binds a packet to a relay session and provides the terminal APS with suf-

ficient state to process session messages, data messages, dummy messages, errors, and fragments.

Field	Size	Meaning
sessionid	8 bytes	Logical relay session identifier
packetid	8 bytes	Packet identifier within the session
fragseq	4 bytes	Fragment sequence; zero for unfragmented packet
fragcount	4 bytes	Total fragment count; zero for unfragmented packet
msglen	4 bytes	Valid payload bytes following this header
payloadtype	1 byte	Session, data, dummy, or error payload type
reserved	1 byte	Reserved and set to zero
flags	2 bytes	Relay flags, including the return flag

Table 3: Encrypted AERN relay payload header.

The defined payload classes are none, session-open, session-open acknowledgement, session-close, opaque serialized data, dummy, and error. The relay header does not carry a protocol field. Backend protocol interpretation, when needed, is performed by the backend adapter or by parsing the serialized packet delivered at the callback boundary. A dummy packet uses the dummy payload type, the reserved byte set to zero, and no backend delivery.

2.7 Relay Session-Open Procedure

AERN uses an explicit ingress-to-egress relay session. A session-open payload is 36 bytes and contains the session identifier, destination address, one byte ingress hint, one byte egress hint, destination port, reserved byte, and flags.

The egress validates the destination field, route hints, session identifier, reserved byte, and flags according to local policy. If the request is accepted, the egress creates an active egress-side session entry and returns a 12 byte acknowledgement containing the session identifier, status byte, flags byte, and reserved sixteen-bit field. The ingress releases queued non-dummy data only after accepting a valid acknowledgement bound to the same session identifier and egress context.

Field	Size	Meaning
sessionid	8 bytes	Session identifier chosen by ingress
destination	22 bytes	Serialized destination address
ingresshint	1 byte	One-based ingress APS route hint
egresshint	1 byte	One-based egress APS route hint
port	2 bytes	Destination port
reserved	1 byte	Reserved and set to zero
flags	1 byte	Session-control flags

Table 4: Encrypted AERN session-open payload.

2.8 Fragmentation and Reassembly

AERN fragments serialized data inside the encrypted relay payload layer. Fragment metadata is not a clear outer-header field. A data fragment carries the same session identifier and packet identifier across the fragment set. The maximum data carried in one relay fragment is 1396 bytes. The configured maximum fragment count is 4096. Incomplete fragment sets expire after the configured fragment-cache timeout of 30000 milliseconds.

A terminal APS inserts a fragment into the reassembly cache only after the enclosing tunnel packet has authenticated and decrypted successfully. Reassembly is keyed by session identifier, packet identifier, direction, payload type, flags, and reserved byte. A fragment set is accepted only if every fragment is within bounds, fragment lengths are valid, fragment metadata is consistent, and the reconstructed length matches the accumulated fragment bytes.

2.9 Backend Transport Boundary

The backend transport callback is outside the cryptographic relay core. The relay core authenticates the tunnel packet, decrypts the relay plaintext, validates the route path and relay payload header, checks session state, reassembles fragments when required, and submits the resulting serialized packet to the configured egress or ingress callback. The callback may bind to a TUN or TAP device, raw socket backend, userspace transport stack, packet filter, or application adapter. The formal proof does not assert the security of the external destination, operating-system network device, or application stack.

2.10 Dummy Traffic and Randomized Ingress Delay

Dummy relay traffic uses the same fixed packet size, route-path processing, and tunnel encryption as ordinary relay packets. Dummy packets contain random session and packet identifiers, the dummy payload type, a reserved byte set to zero, and random body bytes. A terminal APS discards authenticated dummy payloads without backend delivery.

Dummy generation is governed by local utilization policy, and is analyzed in relation to established cover-traffic and low-latency anonymity literature [16]. The configured parameters include a 10 percent utilization floor, a 25 percent ceiling, generation intervals between 50 and 250 milliseconds, a 1000 millisecond accounting window, a target based on 128 MTU-sized packets, and a maximum of eight dummy packets per accounting window. Randomized ingress delay may delay outbound packets by zero to 25 milliseconds before mesh injection. These mechanisms are analyzed as traffic-shaping controls, not as complete timing-channel eliminators.

3 Protocol Summary and Notation

3.1 Notation

Symbol	Meaning
λ	Global security parameter
$\mathcal{N}$	Set of certified AERN nodes
$\mathcal{P}$	Set of route-addressable APS relays in the active topology
n	Cardinality of $\mathcal{P}$, with $n \leq 255$ for one byte route hints
C_D^{σ}	Device certificate of D signed by the ARS root
sk_D, pk_D	Device signing key and verification key
$kem.sk, kem.pk$	KEM secret and public key material
(c, ss)	KEM ciphertext and shared secret
Hdr_{out}	22 byte visible authenticated outer tunnel header
C_{rel}	1478 byte authenticated encrypted relay ciphertext region
P_{rel}	1446 byte decrypted relay plaintext
L_{16}	Two byte encrypted used-length prefix
R_{16}	Sixteen byte encrypted consumed-path route array

$RHdr_{32}$	Thirty-two byte encrypted relay payload header
sid	Relay session identifier
pid	Relay packet identifier
seq_{exp}	Next expected receive sequence number
$Q_{pending}$	Pending queue for data awaiting session-open acknowledgement
Q_{delay}	Optional randomized ingress delay queue
B_{egress}	Egress backend transport callback boundary
$B_{ingress}$	Ingress return-delivery callback boundary
Adv	Adversary advantage in a security game

Table 5: Notation used in the AERN analysis.

3.2 Participants and Roles

The ARS signs certificates and defines the certificate trust root. The ADC distributes topology, validates registration, signs topology and administrative messages, processes revocation and convergence, and may authenticate remote signing requests to the ARS. APS nodes are authenticated relays. An APS may act as ingress, interior relay, egress, return-path participant, relay session cache holder, fragment cache holder, and dummy traffic generator. Clients establish tunnels to entry APS nodes and submit serialized traffic. The backend transport boundary receives validated serialized packets from AERN and returns serialized packets into AERN for reverse delivery.

3.3 Administrative Message Families

AERN administrative messages include announce broadcast and response, converge request and response, converge update, fragment-key request and response, incremental update request and response, MFK exchange request and response, register request and response, register update and versioned register update response, remote signing request and response, resign request and response, revoke broadcast and response, join request and response, topological query request and response, and topological status request and response.

For a sender S and a control message type t, define

$$M_{admin} = t \parallel payload \parallel seq \parallel utc \parallel version.$$

The sender computes

$$\sigma_S = \mathsf{Sign}_{sk_S}(H(M_{admin})).$$

The receiver accepts only if the sender certificate chains to the ARS root, the sender role is authorized for the message type, the timestamp is fresh, the sequence value is valid under the message policy, the signature verifies, and the payload is consistent with certificate and topology constraints.

4 Adversary and Security Model

4.1 Primary Adversary and Scope

The primary adversary against which AERN is analyzed is a passive, flow-correlating network observer operating against honest, authenticated, owned infrastructure. This adversary can observe packet timing, sizes, and link-level flow direction on network infrastructure it monitors, such as ISP or exchange-point links, and attempts to associate a client with a destination by correlating ingress-side and egress-side traffic. Crucially, this adversary obtains its observations by monitoring links, not by compromising AERN nodes; it is fully active even when every ARS, ADC, and APS node is honest and uncompromised. The data-phase mechanisms analyzed in this paper, namely fixed-size packets, encrypted route and session metadata, per-packet interior route resampling, dummy traffic, and bounded ingress delay, are defenses against this observer rather than against node compromise.

Two further adversary classes are treated as *secondary* and are analyzed only for the structural exposure they create, not as the primary design target. The first is the node-compromise adversary, which may compromise a bounded subset of APS nodes; its effect is bounded by the Partial Compromise result and is positioned as a structural-exposure statement rather than the central anonymity claim. The second is the trust-anchor adversary, namely compromise of the ARS or ADC. Consistent with the deployment model, in which the ARS is maintained offline and the ADC is a hardened in-scope trusted component whose certificate is root-signed and verified by every device, trust-anchor compromise is treated as an operational-security concern outside the core protocol model; the protocol is analyzed under the assumption that the ARS and ADC behave correctly, and results that depend on this are stated as conditional on honest topology distribution. A fully global timing adversary with simultaneous high-precision visibility of all ingress and egress links is explicitly out of primary scope; AERN is a low-latency design and does not claim mixnet-grade resistance against such an adversary.

4.2 Adversary Capabilities

The adversary is a probabilistic polynomial-time algorithm with the ability to observe, delay, drop, reorder, inject, and modify packets on any network link. The

adversary may observe public topology information distributed by the ADC and may compromise selected nodes. A compromise may reveal long-term signing keys, active tunnel state, relay cache state, fragment state, backend-adjacent plaintext delivered at a compromised endpoint, and any unencrypted memory accessible at the time of compromise.

The adversary may attempt to distinguish real and dummy traffic, correlate client ingress timing with backend-facing egress timing, bias scheduling by dropping or delaying packets, replay accepted tunnel packets, modify route paths, mix fragments across packets, forge administrative messages, or cause premature backend delivery. Quantum capability is treated in two models. In the Q1 model, honest parties and the adversary exchange classical protocol messages while the adversary may perform quantum computation against public data, recorded transcripts, public keys, and ciphertexts. In the Q2 model, the adversary is additionally allowed quantum-access interaction with idealized primitive oracles. The concrete protocol messages of AERN are classical; the post-quantum claims in this paper are Q1 claims unless a primitive assumption is explicitly stated in a Q2 oracle model. The primitive assumptions define the boundary of feasible attack.

4.3 Compromise Classes

Long-term signing key compromise gives the adversary the signing key of a device. Ephemeral exchange compromise gives KEM secrets and KDF intermediates present during a tunnel exchange. Active state compromise gives transmit and receive cipher state for a live tunnel. Interior APS compromise reveals the local predecessor and successor relation for packets processed by that APS and the decrypted relay state available at that hop. Ingress compromise reveals client-side timing, entry tunnel state, selected egress context for active sessions, and pending queues. Egress compromise reveals backend-facing payloads after terminal relay processing. ADC compromise affects topology integrity, revocation, availability, and route-set bias. ARS compromise affects future certificate issuance and identity trust.

4.4 Security Targets

AERN is analyzed against the following targets: mutual authentication of devices and administrative messages, tunnel-key indistinguishability, weak forward secrecy for completed exchanges after erasure, conditional recovery after authenticated rekey, confidentiality and integrity of per-hop tunnel packets, route-path confidentiality on the wire, strict replay rejection, relay-session correctness, fragment reassembly integrity, direction separation for return traffic, backend-boundary separation, and conditional anonymity under partial compromise.

5 Game-Based Definitions

Definition 5.1 (Mutual Authentication Game)**.** *The challenger generates an ARS root key pair, issues device certificates, initializes an ADC certificate, and provides public certificates and topology data to the adversary. The adversary may interact with administrative, tunnel-establishment, and relay oracles. The adversary wins if an honest participant accepts a peer identity, topology message, control message, session-open acknowledgement, or tunnel peer as valid for an uncorrupted identity that did not authorize the accepted message.*

Definition 5.2 (Tunnel-Key Indistinguishability)**.** *The challenger executes an authenticated KEM exchange between honest participants. The adversary receives either the real derived tunnel state or random strings of the same length. The adversary wins by distinguishing real state from random subject to freshness restrictions on session reveal and state compromise.*

Definition 5.3 (Weak Forward Secrecy)**.** *The challenger completes a tunnel exchange, erases KEM secret material and KDF intermediates, and then reveals long-term signing keys. The adversary wins if it distinguishes completed session keys from random or decrypts a challenge packet without having obtained active symmetric state for that completed exchange.*

Definition 5.4 (Conditional Recovery after Rekey)**.** *The adversary obtains active tunnel state for a session. The honest peers then perform an authenticated rekey with fresh exchange material and erase the exposed state. The adversary wins if it distinguishes or forges post-rekey traffic without compromising the refreshed state or forging the rekey authentication.*

Definition 5.5 (Per-Hop Tunnel Security)**.** *The challenger provides encryption and decryption oracles for a tunnel direction. The encryption oracle accepts Hdr_{out} and P_{rel} and returns $Hdr_{out} \parallel C_{rel}$. The decryption oracle accepts a packet, validates header policy, verifies authentication, checks freshness and sequencing, and returns P_{rel} or $\perp$. The adversary wins confidentiality by distinguishing encryptions of equal-length relay plaintexts and wins integrity by producing a new accepted packet.*

Definition 5.6 (Strict Replay Resistance)**.** *For each tunnel direction, the challenger maintains seq_{exp}. A packet is accepted only if its authenticated sequence equals seq_{exp}, its timestamp lies within the configured threshold, and authentication succeeds. The adversary wins if a replayed or out-of-sequence packet is accepted without a valid forgery for the next expected sequence.*

Definition 5.7 (Relay Session Correctness)**.** *The adversary may deliver, delay, drop, replay, and modify session-open, session-open acknowledgement, and data*

payloads. It wins if an honest ingress releases queued non-dummy data for a session before accepting a valid session-open acknowledgement bound to the same session identifier and egress context.

Definition 5.8 (Fragment Reassembly Integrity)**.** *The adversary may deliver authenticated and unauthenticated fragments, reorder them, duplicate them, and attempt to mix fragments across sessions or packet identifiers. It wins if an honest terminal endpoint accepts a reassembled message containing any fragment that was not individually authenticated under the corresponding tunnel key and consistent with the same session identifier, packet identifier, fragment count, direction, flags, reserved byte, payload type, and reconstructed length.*

Definition 5.9 (Route-Path Confidentiality)**.** *The adversary submits two equal-length relay plaintexts containing different route paths and relay headers. The challenger encrypts one of them under an honest tunnel key. The adversary wins by identifying which route path and relay header were encrypted, without controlling the relevant decrypting tunnel endpoint.*

Definition 5.10 (Conditional Anonymity)**.** *The adversary selects two client-destination scenarios. The challenger samples a bit b, establishes the client-to-entry tunnel for scenario b, creates an ingress-to-egress relay session, and sends fixed-size relay packets with encrypted route paths and per-packet interior route sampling. The adversary observes links and may compromise an allowed subset of nodes. The adversary wins by identifying b. The theorem excludes adversaries that simultaneously control or observe the relevant ingress-side and egress-side correlation points for the challenge session.*

6 Formal Specification

6.1 Certificate and Topology Acceptance

Let C_D^σ be the ARS-signed certificate of device D. A receiver accepts C_D^σ only if the ARS signature verifies, the root serial reference matches the accepted root, the expiration interval is valid, the designation is authorized for the message or relay role, the algorithm and version are accepted, and any available topology node binds the same certificate hash, issuer, serial number, expiration interval, and designation.

Let $Topo = (v, nodes, H(nodes))$ be a versioned topology state. A topology update is accepted only if it is signed by the ADC, the ADC certificate chains to the ARS root, the version is not stale under local policy, and the serialized topology list satisfies structural validity. Route hint resolution is defined only for APS entries in the active topology.

Lemma 6.1 (Administrative Message Integrity)**.** *Assume the signature scheme is EUF-CMA secure and hash collisions are negligible. No polynomial-time adversary can cause an honest node to accept a modified administrative message from an uncorrupted sender except with negligible probability.*

Proof. Acceptance requires a valid signature over $H(M_{admin})$. A modification to the message type, payload, sequence, timestamp, or version changes the hash except with collision probability. If the sender is uncorrupted, a valid signature on the modified hash is an EUF-CMA forgery. The adversary therefore succeeds only by finding a collision or forging a signature. □

6.2 Tunnel Acceptance Procedure

The receive-side tunnel procedure is modeled as follows:

$\mathsf{Accept}(Hdr_{out}, C_{rel})$:
  parse Hdr_{out};
  reject if the packet flag or ciphertext length is invalid;
  reject if the timestamp is outside the threshold;
  reject if $seq \neq seq_{exp}$;
  verify authentication over Hdr_{out}, C_{rel};
  reject if authentication fails;
  $P_{rel} \leftarrow \mathsf{Dec}_K(Hdr_{out}, C_{rel})$;
  $seq_{exp} \leftarrow seq_{exp} + 1$;
  parse $L_{16}, R_{16}, RHdr_{32}, B$.

The order is security-critical. Relay state, route state, session state, fragment state, and backend state are not advanced before timestamp, sequence, and authentication checks succeed.

6.3 Route Generation and Forwarding

Let $\mathcal{P} = \{P_1, \dots, P_n\}$ be the active route-addressable APS topology, with $3 \leq n \leq 255$, indexed in canonical topology-sorted order. The implementation stores topology nodes in a zero-based array, while route hints are one-based ordinals into that order. For the node at zero-based array index $j \in \{0, \dots, n-1\}$ the route hint is $hint = j+1$, so that the valid hint range is $\{1, \dots, n\}$ and the value 0 is reserved as the route terminator and as the empty value for unused future-hop slots. Equivalently, under the one-based labelling $P_1, \dots, P_n$ used here, $hint(P_i) = i$. The implementation realizes this with the bijection `route_hint_from_index`$(j) = j+1$

and its inverse `route_aps_ordinal_from_hint`$(h) = h - 1$, which is defined only for $h \neq 0$. A generated route is represented in R_{16} as

$$R_{16} = [h_0, h_1, \ldots, h_{15}],$$

where h_0 is the origin hint, h_j for $j \geq 1$ are future-hop hints, and zero terminates unused future slots. For a valid generated route, all nonzero hints are within $\{1, \ldots, n\}$, adjacent nonzero hints are distinct, interior hints exclude the origin and target, and the final active hint identifies the selected target APS.

A forwarding APS decrypts the tunnel packet, validates R_{16}, locates the first nonzero future-hop slot $j \geq 1$, resolves h_j against topology, sets $h_j = 0$, reserializes R_{16}, encrypts the resulting relay plaintext under the tunnel shared with the next APS, and transmits a fixed-size packet. If no nonzero future-hop slot exists, the packet is terminal for the current traversal direction.

Lemma 6.2 (Route Validity Preservation)**.** *If a route path R_{16} generated by the AERN route generator is processed by an honest forwarding APS and the selected next-hop hint resolves to a valid APS peer, then the reserialized route path remains structurally valid for the next hop.*

Proof. The route-validation predicate `route_map_audit_valid` requires (i) $path[0] \neq 0$ and $path[0] \in \{1, \ldots, n\}$, and (ii) for the subsequence of nonzero entries read left to right, every adjacent pair is in range and distinct; zero entries are skipped. The forwarding operation sets exactly one nonzero future-hop slot h_j to zero, introduces no new hint value, does not modify $path[0]$, and does not reorder the remaining nonzero entries. Range validity of every surviving nonzero entry is therefore unchanged.

The only nontrivial obligation is the adjacency check, because zeroing h_j can make two entries that were previously separated by h_j newly adjacent in the nonzero subsequence. Because consumption always removes the *first* nonzero future-hop, the only adjacent pair that can be newly exposed at a single forwarding step is $(path[0], b)$, where b is the new first future-hop, if one exists. For routes generated by the implementation, interior and target hints exclude the origin value $path[0]$, so $b \neq path[0]$. Hence the newly exposed pair is distinct. All other adjacent nonzero pairs are inherited unchanged from a valid array. The reserialized route path therefore satisfies (i) and (ii) and remains structurally valid for the next hop. □

6.4 Relay Session State Machine

A relay session is identified by sid and by its ingress and egress context. The possible status values are none, pending, active, closing, expired, and failed.

On the first outbound serialized packet for a destination, the ingress creates $pending(sid)$, transmits a session-open payload, and stores non-dummy packets in $Q_{pending}$. The egress creates $active(sid)$ only after accepting the session-open payload. The ingress changes $pending(sid)$ to $active(sid)$ only after accepting a session-open acknowledgement with the same sid and egress context.

Theorem 6.3 (Relay Session Correctness)**.** *An honest ingress APS does not release queued non-dummy data for a relay session unless it has accepted a valid session-open acknowledgement bound to the same session identifier and egress context.*

Proof. By the session state machine, data for a new session is placed in $Q_{pending}$ while the session is pending. The only transition from pending to active is acceptance of a session-open acknowledgement. Acceptance requires authenticated tunnel delivery, valid relay payload parsing, matching session identifier, matching egress context, and an acceptable status value. A modified acknowledgement fails tunnel authentication or payload validation except with tunnel-forgery probability. A replayed acknowledgement fails strict sequencing or session-context checks. Therefore queued non-dummy data cannot be released before a valid acknowledgement for the same context is accepted. □

6.5 Fragmentation Model

Let M be a serialized backend payload. If $|M| \leq 1396$, it may be carried as a single data payload with $fragseq = 0$ and $fragcount = 0$. If $|M| > 1396$, the sender partitions M into fragments $M_1, \ldots, M_k$, where $k \leq 4096$ and $|M_i| \leq 1396$ for all i. Each fragment carries the same sid and pid in the encrypted relay payload header.

Theorem 6.4 (Fragment Reassembly Integrity)**.** *A message accepted by an honest terminal APS or client return path consists only of fragments that were individually authenticated under the corresponding tunnel key and whose encrypted relay headers agree on session identifier, packet identifier, fragment count, direction, payload type, flags, reserved byte, and reconstructed length.*

Proof. A fragment is inserted into the fragment cache only after the enclosing tunnel packet passes timestamp validation, strict sequence validation, and authentication. Fragment metadata is carried inside $RHdr_{32}$ and is therefore integrity-protected by the tunnel authentication. The fragment cache binds the first valid metadata values for the set and rejects fragments that disagree with the bound payload type, reserved byte, flags, direction, session identifier, packet identifier, or size policy. Reassembly requires all fragments before timeout and length consistency. Mixing a fragment from another set therefore requires either metadata equality under a valid authenticated packet or a successful tunnel forgery. □

7 Cryptographic Assumptions

7.1 KEM Security

The configured post-quantum KEM is assumed IND-CCA2 secure against Q1 adversaries attacking classical public keys, ciphertexts, and transcripts. A stronger Q2 treatment would require the selected KEM and its proof model to support quantum-access decapsulation-oracle formulations; that stronger model is not needed for the classical AERN message interface analyzed here. For any polynomial-time adversary $\mathcal{A}$,

$$\mathsf{Adv}^{\text{IND-CCA2}}_{\text{KEM},\mathcal{A}}(\lambda) \leq \epsilon_{kem}(\lambda).$$

This assumption supports indistinguishability of the shared secret used to derive tunnel state.

7.2 Signature Security

The signature scheme used for certificates and control messages is assumed existentially unforgeable under chosen-message attack against Q1 adversaries. A Q2 signing-oracle model would be a stronger primitive assumption and is not claimed by the protocol analysis unless supplied by the instantiated signature scheme. For any polynomial-time adversary $\mathcal{A}$,

$$\mathsf{Adv}^{\text{EUF-CMA}}_{\text{Sig},\mathcal{A}}(\lambda) \leq \epsilon_{sig}(\lambda).$$

This assumption supports certificate authenticity, administrative message integrity, and authenticated binding of exchange material to certified identities.

7.3 KDF Security

The SHAKE-based KDF is modeled as a pseudo-random function or random oracle over the shared secret and transcript context. The default security statement is a Q1 statement over classical inputs and outputs. A Q2 statement requires the corresponding quantum random-oracle or quantum-access PRF assumption for the instantiated proof. If ss is unknown to the adversary, then $\mathsf{KDF}(ss, ctx)$ is computationally indistinguishable from random output of the same length except with advantage $\epsilon_{kdf}(\lambda)$.

7.4 Tunnel Cipher and Authentication Security

The tunnel cipher is assumed to provide confidentiality and ciphertext integrity under synchronized state and valid nonce discipline against classical packet-query

adversaries, with quantum resistance inherited from the underlying primitive assumptions in the Q1 model. For an RCS/KMAC instantiation this is represented by pseudo-randomness of the cipher output and unforgeability of the authentication mechanism [3]. For an approved AEAD profile, the corresponding AEAD security advantage replaces these terms.

8 Handshake Security

Theorem 8.1 (Handshake Correctness)**.** *If two honest parties complete the authenticated KEM exchange, validate certificates, verify signatures, use the same accepted transcript context, and KEM decapsulation succeeds, then the transmit state of each party matches the receive state of the peer for the corresponding tunnel direction.*

Proof. KEM correctness gives both parties the same shared secret ss. Both parties verify the same certificate identities, algorithm identifiers, role context, and exchange material before deriving state. The KDF is deterministic. Direction labels assign one derived state to the $A \to B$ direction and the complementary state to the $B \to A$ direction. The corresponding transmit and receive states therefore match by construction. □

Theorem 8.2 (Tunnel-Key Indistinguishability)**.** *Let $\mathcal{A}$ attack a fresh AERN tunnel exchange. If the KEM is IND-CCA2 secure, the KDF is pseudo-random, the signature scheme is EUF-CMA secure, and hash collisions are negligible, then*

$$\mathsf{Adv}_{\mathcal{A}}^{Key} \leq \epsilon_{kem} + \epsilon_{kdf} + \epsilon_{sig} + \epsilon_{coll}.$$

Proof. We proceed by a sequence of games, writing Adv_i for the adversary's advantage in Game i.

Game 0 is the real Tunnel-Key Indistinguishability experiment (the Tunnel-Key Indistinguishability game), in which the challenger runs an honest authenticated MFK/MEK exchange and returns either the real derived state $K_{tx} \parallel K_{rx} \parallel N_{tx} \parallel N_{rx}$ or random strings of equal length.

Game 1 aborts if the adversary causes either honest party to accept a key-exchange message (the signed public encapsulation key, the signed ciphertext, or the certificate transcript) that the certified peer did not produce. Each such message is authenticated by a signature over a hash that binds the message body, the peer identity through its certificate, and the freshness fields (sequence number and UTC timestamp). By Lemma 6.1, any accepted but unauthorized message yields an EUF-CMA forgery or a hash collision, so $|\mathsf{Adv}_0 - \mathsf{Adv}_1| \leq \epsilon_{sig} + \epsilon_{coll}$. After Game 1 the transcript context ctx seen by the two honest parties is identical and is bound to the certified identities; in particular the adversary cannot have injected a chosen encapsulation key or ciphertext.

Game 2 replaces the KEM shared secret ss delivered to the honest endpoints with a uniformly random secret $\widetilde{ss}$ of the same length. Because the responder's KEM key pair is generated freshly for this exchange and the ciphertext was produced honestly under it (guaranteed by Game 1), a distinguisher between Game 1 and Game 2 is an IND-CCA2 distinguisher against the KEM, using the decapsulation oracle to answer any non-challenge tunnel queries. Hence $|\mathsf{Adv}_1 - \mathsf{Adv}_2| \leq \epsilon_{kem}$.

Game 3 replaces $\mathsf{KDF}(\widetilde{ss}, ctx)$ with a uniformly random string of the same length. Since $\widetilde{ss}$ is uniform and independent of the adversary's view, this step is a PRF/random-oracle distinguishing step for the KDF keyed on $\widetilde{ss}$, so $|\mathsf{Adv}_2 - \mathsf{Adv}_3| \leq \epsilon_{kdf}$. In Game 3 the challenge real-or-random values are identically distributed, so $\mathsf{Adv}_3 = 0$.

Summing the differences gives $\mathsf{Adv}^{Key}_{\mathcal{A}} \leq \epsilon_{kem} + \epsilon_{kdf} + \epsilon_{sig} + \epsilon_{coll}$. The directional labels in ctx ensure K_{tx}, N_{tx} for one endpoint equal K_{rx}, N_{rx} for the peer and that the two directions receive independent key material from the single random KDF output, so the bound applies jointly to both tunnel directions. □

Theorem 8.3 (Mutual Authentication)**.** *An adversary that causes an honest AERN participant to accept an unauthorized peer identity, topology control message, administrative message, or authenticated exchange transcript can be converted into an adversary that forges the signature scheme or finds a hash collision.*

Proof. Let $\mathcal{A}$ be an adversary that, with non-negligible probability, makes some honest party accept one of the listed objects as originating from an uncorrupted identity I that did not authorize it. We build $\mathcal{B}$ against EUF-CMA. $\mathcal{B}$ receives the challenge verification key, embeds it as the verification key of I in a guessed certified identity (guessing among the polynomially many identities loses only a polynomial factor), generates all other keys honestly, and answers $\mathcal{A}$'s oracle queries by signing through its EUF-CMA signing oracle for messages legitimately produced by I and signing locally for all other parties. Certificate validation, topology distribution, control messaging, session-open acknowledgement, and tunnel-establishment transcripts are each accepted only after verifying a signature by the claimed sender over a hash $H(\cdot)$ that includes the message body together with the binding fields: the issuer and serial number of the sender certificate, the message type, and the freshness fields (sequence number and UTC timestamp).

When $\mathcal{A}$ succeeds, the accepting honest party has verified a signature σ^* on a hash $H(m^*)$ under I's verification key. Two cases arise. If m^* was never signed by I (the message content, type, identity binding, or freshness field differs from anything I produced), then $(H(m^*), \sigma^*)$ is a valid EUF-CMA forgery, which $\mathcal{B}$ outputs. If m^* collides with some message m that I did sign, i.e. $H(m^*) = H(m)$ with $m^* \neq m$, then $\mathcal{A}$ exhibits a hash collision. Because the freshness fields and the certificate identity are inside the hashed image, replays across sessions or

identities fall into the first case (a different binding field) rather than constituting authorized acceptance. Therefore $\mathsf{Adv}^{auth}_{\mathcal{A}} \leq q_I \cdot \epsilon_{sig} + \epsilon_{coll}$, where q_I bounds the number of certified identities, and mutual authentication reduces to signature unforgeability and collision resistance. □

Theorem 8.4 (Weak Forward Secrecy)**.** *Completed AERN tunnel exchanges remain confidential after later compromise of long-term signing keys, provided ephemeral KEM secrets, KDF intermediate values, and active symmetric state for the completed exchange were erased before compromise.*

Proof. The argument rests on the separation between the long-term signing keys and the data-plane key material, and on the exchange producing fresh secrets per session. In the MFK/MEK exchange the responder generates a KEM key pair freshly for each exchange; the shared secret ss is obtained by encapsulation under that ephemeral public key, and the directional tunnel state is derived as $\mathsf{KDF}(ss, ctx)$. The long-term signing keys are used only to authenticate the exchange messages (the public key, the ciphertext, and the certificate transcript); they never key the KDF and never serve as encryption or authentication keys for tunnel packets.

Consider an adversary that obtains the long-term signing keys of both endpoints *after* a target exchange has completed, provided the ephemeral KEM secret key, the shared secret ss, the KDF intermediates, and the active symmetric tunnel state for that exchange were erased before the compromise. The post-compromise view contains the public transcript and the signing keys, but not ss and not the derived state. Recovering the challenge plaintext, or distinguishing the completed session keys from random, therefore requires computing ss from the public transcript (an IND-CCA2 / one-wayness break of the KEM against an honestly generated ephemeral public key) or distinguishing $\mathsf{KDF}(ss, ctx)$ from random (a KDF break). Possession of the signing keys does not assist either task: signing keys let the adversary authenticate *future* messages but do not invert the ephemeral encapsulation of a past exchange. Hence the advantage against erased completed exchanges is bounded by $\epsilon_{kem} + \epsilon_{kdf}$, and weak forward secrecy holds. The qualifier "weak" reflects that the guarantee covers only exchanges that completed and whose ephemeral and active state were erased prior to compromise; it does not cover traffic protected by state that is still live at the moment of compromise, which is addressed separately by the Active State Exposure theorem. □

Theorem 8.5 (Conditional Recovery after Rekey)**.** *Assume an adversary obtained active tunnel state for a session but does not obtain refreshed state. If peers complete an authenticated rekey with fresh KEM material and erase the exposed state, then the adversary's advantage against post-rekey traffic is bounded by the primitive and tunnel-security advantages for the refreshed state.*

Proof. After rekey, encryption and authentication use state derived from fresh exchange material. The exposed state is not used. Distinguishing or forging post-rekey traffic therefore requires distinguishing the fresh KEM/KDF output, forging rekey authentication, or breaking the refreshed tunnel security. The exposed state does not compute the refreshed state under the stated assumptions. □

9 Channel Security and Replay Resistance

Theorem 9.1 (Per-Hop Confidentiality and Integrity)**.** *Let $\mathcal{A}$ make polynomially many encryption and decryption queries against an honest tunnel direction. The advantage of $\mathcal{A}$ against AERN per-hop tunnel confidentiality and integrity is bounded by the tunnel cipher pseudo-randomness advantage, authentication-forgery advantage, state-misuse probability under implementation constraints, and collision probability. Under an AEAD profile the first two terms are replaced by the AEAD security advantage.*

Proof. Confidentiality follows by replacing the tunnel cipher output with an ideal encryption or random pad in the usual hybrid. Integrity follows because an accepted new packet must carry a valid authentication result over Hdr_{out} and C_{rel}. Producing such a packet without active tunnel state is an authentication forgery except with collision or state-misuse probability. Since the route path, relay payload header, and body are inside C_{rel}, the same bound applies to route and relay metadata. □

Theorem 9.2 (Route-Path and Relay Metadata Confidentiality)**.** *If the tunnel encryption is confidential for equal-length relay plaintexts, then a wire observer cannot distinguish route paths, session identifiers, packet identifiers, payload types, fragment metadata, reserved byte values, or return flags for equal-length relay plaintexts except with the tunnel confidentiality advantage.*

Proof. All listed values are contained in R_{16} or $RHdr_{32}$, which are subfields of P_{rel}. The adversary observes Hdr_{out} and C_{rel} but not P_{rel}. Distinguishing which route path or relay header was encrypted would distinguish the encryption of two equal-length relay plaintexts, contradicting tunnel confidentiality except with the stated advantage. □

Theorem 9.3 (Strict Replay Resistance)**.** *A replayed AERN tunnel packet is rejected unless the adversary can produce a valid authenticated packet with the next expected receive sequence number and a fresh timestamp, or has compromised active tunnel state.*

Proof. Each tunnel direction maintains a 64-bit receive counter seq_{exp}, initialized at tunnel establishment and never decremented. The acceptance procedure validates the authenticated outer header and verifies the tunnel tag over $Hdr_{out} \parallel C_{rel}$

before advancing any state, and accepts a packet only when its authenticated sequence s satisfies $s = seq_{exp}$ and its timestamp lies within the configured threshold; seq_{exp} is incremented only on acceptance and is left unchanged on any rejection.

Fix any packet the adversary delivers. If it carries an authenticated sequence $s < seq_{exp}$ (a replay or a stale packet) the equality test fails and it is rejected with no state change. If $s > seq_{exp}$ (a reordered or future packet) the equality test also fails under the strict zero-width window and it is rejected. The remaining possibility is $s = seq_{exp}$: a packet the adversary did not legitimately obtain for the current expected sequence is accepted only if it carries a valid tag over a header bearing seq_{exp} and a fresh timestamp. Since the sequence and timestamp are inside the authenticated header, producing such a packet without the active tunnel state is an INT-CTXT forgery, bounded by the tunnel authentication advantage. The only non-forgery route to acceptance at $s = seq_{exp}$ is the unique honest packet the legitimate sender emitted for that sequence, which is by definition not a replay. Counter wraparound is out of scope: it would require 2^{64} accepted packets on a single direction, beyond which the rekey policy mandates fresh tunnel state. Hence a replayed or out-of-sequence packet is rejected except with tunnel-forgery probability or under active-state compromise. □

Lemma 9.4 (Packet Size Uniformity)**.** *All AERN relay wire packets have fixed size 1500 bytes. Packet length therefore does not reveal route length, payload type, fragment state, or application message length within a single relay packet.*

Proof. Each relay packet is serialized into a 1500 byte wire buffer. Variable length body data is represented inside the encrypted relay plaintext and padded or fragmented into fixed-size packets. A wire observer may still observe packet counts and timing, but packet size is constant. □

10 Anonymity and Traffic Analysis

10.1 Conditional Anonymity Model

AERN's anonymity claim is stated against the primary adversary of Section 4.1: a passive, flow-correlating link observer facing honest, authenticated infrastructure. Against this adversary the protocol hides the route path, relay payload header, session identifiers, fragment metadata, and return flags inside the encrypted relay plaintext, presents only fixed-size packets and authenticated outer headers on the wire, resamples the interior route on every packet, and emits optional dummy traffic under bounded local policy.

The anonymity objective in this model is not perfect unlinkability but *endpoint non-attribution*: the observer should be unable to assert, with confidence above

what the cover process permits, that a given observed flow traverses a specific ingress-to-egress APS pair. Because per-packet payloads are size-uniform and encrypted, interior addressing is encrypted, the interior route varies per packet, and real flows are mixed with dummy traffic and bounded jitter on a loaded mesh, the mapping from an observed ingress event to an observed egress event becomes a probabilistic inference rather than a deterministic linkage. The strength of this property is a function of network saturation, the cover-traffic budget, and flow shape, and is quantified in Section 10.4. It does not hold against a fully global observer with high-precision visibility of all ingress and egress links, and it does not protect the client-destination relation when the relevant ingress-side and egress-side correlation positions are simultaneously node-compromised; that residual exposure is bounded separately below.

10.2 Endpoint Compromise Bound

The following bound addresses the *secondary* node-compromise adversary, not the primary link observer of Section 4.1. It does not describe the anonymity the protocol provides against passive flow correlation, which is treated in Section 10.4; it instead quantifies the structural exposure that arises if the adversary additionally compromises APS nodes and happens to occupy both endpoint correlation positions for a session. Let f be the fraction of route-addressable APS nodes compromised by the adversary, and suppose ingress-side and egress-side endpoint selection is independent over the effective route domain without ADC bias. The probability that the adversary controls both endpoint correlation positions is approximately f^2 under this simplified independent model. This is not a complete anonymity proof, and the value of f^2 is meaningful only at the route-set size n of an actual deployment: at small n the route-addressable set is itself a small anonymity set, independently of f.

Theorem 10.1 (Partial Compromise Bound)**.** *Conditioned on honest topology distribution and independent endpoint selection from a route-addressable APS set with compromised fraction f, the probability that the adversary controls both endpoint relay correlation positions is approximately f^2. Interior-only compromise reveals local forwarding observations but not the complete client-destination relation by itself.*

Proof. Let $\mathcal{P}$ be the route-addressable APS set with $|\mathcal{P}| = n$ and let $\mathcal{C} \subseteq \mathcal{P}$ be the compromised subset with $|\mathcal{C}| = fn$. The ingress is the client's entry APS, selected by the client; the egress is selected by the ingress when it creates the relay session. Model both selections as drawn from the topology-distributed set $\mathcal{P}$. Under the stated conditions of honest topology distribution (the ADC does not bias either selection toward $\mathcal{C}$) and selection independence, the event that the ingress correlation

position is compromised has probability $\Pr[I \in \mathcal{C}] = f$, the event that the egress correlation position is compromised has probability $\Pr[E \in \mathcal{C}] = f$, and the two events are independent, so $\Pr[I \in \mathcal{C} \wedge E \in \mathcal{C}] = f^2$. If selection is uniform without replacement over distinct ingress and egress, the joint probability is the slightly smaller $\frac{fn}{n} \cdot \frac{fn-1}{n-1} \leq f^2$, so f^2 is a conservative upper bound. An interior-only relay decrypts only packets for which it is an adjacent tunnel endpoint and may read the consumed-path route array for those packets; it does not simultaneously hold the client-facing entry tunnel state and the backend-facing egress state unless it also occupies an endpoint correlation position or obtains additional timing observations, so interior compromise alone does not yield the client–destination relation.

The bound is conditional. If the ADC is compromised or biased, it can steer entry or egress selection toward $\mathcal{C}$, in which case the joint compromise probability is governed by the induced (non-uniform) selection distribution rather than by f^2; this dependence is exactly why the anonymity statements are conditioned on honest topology distribution. □

10.3 Interior Route Sampling

Interior routes are sampled from the effective APS topology subject to validity constraints. The generation rule excludes zero, the origin hint, the target hint, and the immediately preceding hint. The route generator uses bounded rejection sampling. For route domains with very small eligible interior sets, the generator can fail rather than emit an invalid route.

Lemma 10.2 (Conditional Interior Route Uniformity)**.** *For a fixed origin, target, route length, and eligible set, if the random source is uniform and the rejection predicate is deterministic, then each valid interior sequence satisfying the predicate is selected with equal probability conditioned on success.*

Proof. The generator samples each candidate from the same uniform source and accepts only candidates satisfying the deterministic predicate. For any two accepted interior sequences of the same length, the probability of producing the sequence is the product of equal per-position conditional probabilities over the accepted candidate set. Conditioning on success removes failed rejection paths. The distribution over valid accepted sequences is therefore uniform within the defined constraints. □

10.4 Traffic Shaping Limits

Dummy traffic and randomized ingress delay reduce deterministic timing linkage and sparse-flow observability. Let D_{dummy} be the local dummy process and

let δ be the bounded ingress delay. The observable process can be represented as

$$Obs = RealTraffic_{\delta} \cup D_{dummy}.$$

A local observer faces additional uncertainty because some packets may be dummy and some real packets may be delayed.

We now make the endpoint non-attribution property of Section 10.1 quantitative, since its strength is entirely a function of the cover-traffic budget and network saturation rather than a property that holds unconditionally. Fix an accounting window W (the implementation default is $1000\,\text{ms}$). Let ρ be the real-traffic rate offered to an ingress APS over W, let d be the number of dummy packets emitted in W, and let $\delta \in [0, \delta_{max}]$ be the per-packet ingress delay. The implementation bounds these parameters: dummy emission is capped at $d \leq 8$ packets per window against a target window of 128 MTU-sized packets, dummy generation is gated by a utilization floor of 10% and ceiling of 25% at randomized 50–$250\,\text{ms}$ intervals, and $\delta_{max} = 25\,\text{ms}$. A flow-correlating observer attempting to match an ingress event to an egress event within timing tolerance Δ must distinguish the real flow from the background of concurrent real and dummy flows that fall within Δ. The number of confusable candidates grows with the per-window packet population $\rho W + d$ relative to the matching tolerance, so the observer's per-event attribution advantage decreases as saturation ρW and the cover count d increase and as δ_{max} approaches Δ.

Two consequences follow directly from the bounded parameters. First, because d is capped at 8 per window against a 128-packet target, the dummy contribution is a meaningful share of the population only when real utilization is low; this is by design, since dummy generation is suppressed above the 25% ceiling, so cover traffic raises the floor of background activity during sparse periods but does not scale with a saturated link. Second, because $\delta_{max} = 25\,\text{ms}$ is small relative to the inter-arrival spacing of a sustained high-rate flow, endpoint non-attribution is strong for bursty, low-rate, interactive flows on a loaded mesh and weak for large continuous streams on a sparse mesh, where ρW is dominated by a single identifiable flow and the cover budget cannot mask it. A fully global observer with ingress and egress timing visibility may still correlate aggregate counts and timing, especially against sustained streams. AERN therefore provides quantified traffic-analysis mitigation whose strength is set by the deployment's saturation and the parameters above, rather than complete timing-channel elimination. This is the universal operating envelope of low-latency cover-traffic systems and is not specific to AERN.

Theorem 10.3 (Conditional Route Unlinkability)**.** *Against an adversary that does not simultaneously control or observe the relevant ingress-side and egress-side correlation points, encrypted route metadata, fixed-size packets, and per-packet*

interior route sampling prevent intermediate relay observations from directly revealing the client-destination relationship, except through residual timing and volume leakage.

Proof. A passive wire observer sees fixed-size packets and authenticated outer headers, but not R_{16} or $RHdr_{32}$. An interior relay observes the local tunnel predecessor and successor relation and the decrypted relay state for packets it processes. It does not observe the client-side tunnel endpoint and backend-facing endpoint simultaneously unless it is also positioned at those correlation points or can infer them through timing. Per-packet interior route sampling prevents a stable interior route from defining the session. Thus direct metadata linkage is unavailable under the stated compromise restriction, leaving timing and volume leakage as the residual channel. □

11 Systemic Risks and Trust Anchors

11.1 Centralized Trust

The ARS and ADC are high-value trust components. The ARS controls certificate issuance. The ADC controls topology distribution, registration, revocation, and administrative message signing. This centralized model prevents uncontrolled relay admission and simplifies validation, but it concentrates availability and trust assumptions. Formal anonymity claims are conditioned on honest topology distribution or on an explicit model of topology bias.

Theorem 11.1 (Offline Trust Anchor Safety)**.** *Temporary unavailability of the ARS or ADC does not break confidentiality or integrity of established tunnel traffic, provided nodes continue using authenticated certificates and topology state and do not accept unauthenticated replacements.*

Proof. Established tunnel traffic is protected by symmetric state derived from authenticated KEM exchanges. The ARS and ADC are not decryption or authentication-tag oracles for those packets. If nodes do not replace certificate or topology state with unauthenticated material, offline trust anchors do not alter established tunnel keys or packet authentication checks. □

Theorem 11.2 (Trust Anchor Compromise Boundary)**.** *ARS compromise affects future identity trust. ADC compromise affects topology integrity, revocation, route-set bias, and availability. Neither compromise reveals completed tunnel keys by itself after ephemeral exchange material and active symmetric state have been erased.*

Proof. Completed tunnel keys are derived from KEM shared secrets and KDF context, not from ARS or ADC signing keys. Obtaining a trust-anchor signing

key does not compute erased KEM secrets or erased symmetric state. For future executions, a compromised ARS can authorize malicious identities and a compromised ADC can bias topology or suppress revocation. Those actions affect future authentication and anonymity assumptions without directly decrypting completed tunnel traffic. □

11.2 Forward-Secrecy Boundary

AERN provides weak forward secrecy for completed exchanges after erasure. It does not protect active traffic if the adversary obtains active symmetric state. Active state compromise exposes traffic protected by that state until teardown or authenticated rekey replaces it.

Theorem 11.3 (Active State Exposure)**.** *If a node is compromised during an active tunnel and the adversary obtains current symmetric cipher state, then traffic under that state is exposed until successful rekey or teardown. Traffic protected by erased states remains protected under the KEM, KDF, and tunnel-security assumptions.*

Proof. Active symmetric state is sufficient to decrypt or forge packets protected by that state. When the state is replaced by authenticated rekey and the exposed state is erased, subsequent packets use different derived material. Traffic protected by erased states requires the erased state or the underlying KEM secret. Under the stated assumptions, those values cannot be reconstructed from signing keys alone. □

12 Implementation Conformance Requirements

12.1 Packet and Route Validation

A conforming implementation must serialize the outer tunnel header as 22 bytes, authenticate that header as associated data, use a 1478 byte ciphertext region, derive a 1446 byte decrypted relay plaintext, encode a 2 byte used-length prefix, encode the route path as a 16 byte encrypted array, and encode the relay payload header as a 32 byte encrypted structure. Route hints must be one byte one-based APS ordinals. Zero must not identify an APS node. A route domain using this encoding must not assign more than 255 route-addressable APS hints.

Route processing must authenticate the packet before parsing the route path. A forwarding APS must not modify route state before successful authentication, timestamp validation, and strict sequence validation. The forwarding operation must consume the first nonzero future-hop slot by setting it to zero. Terminal status must be defined as absence of future nonzero route hints.

12.2 Session, Fragment, and Backend Validation

A conforming implementation must hold non-dummy session data in a pending queue until a valid session-open acknowledgement is accepted. It must reject acknowledgement messages with mismatched session identifier, invalid egress context, invalid status, invalid reserved field, stale timestamp, failed authentication, or invalid sequence.

Fragment processing must occur after successful tunnel authentication. The maximum per-fragment payload under the described packet layout is 1396 bytes. The maximum configured fragment count is 4096. Fragment sets must be cleared on timeout, session teardown, tunnel failure, or inconsistent metadata. Backend callbacks must receive only authenticated, session-valid, and reassembled serialized packets. Dummy payloads must not invoke backend delivery.

12.3 Logging and Metadata Discipline

AERN relay implementations must avoid logging client content, backend destination payloads, route paths, relay session identifiers in ordinary operational logs, packet identifiers, and fragment identifiers. Administrative logs may record device identity, event time, error class, certificate state, and topology state needed for operations. Backend adapters must preserve the same metadata discipline at the relay boundary.

12.4 Validation Requirements

A complete validation suite should include packet header serialization tests, tunnel authentication failure tests, strict replay tests, route-path generation tests, route-path consumption tests, terminal detection tests, session-open acknowledgement tests, pending queue release tests, fragment reassembly tests, dummy discard tests, backend callback isolation tests, topology revocation tests, route-domain boundary tests at 254, 255, and invalid 256 route-addressable APS entries, and full in-memory virtual network execution with ARS, ADC, multiple APS nodes, a client, egress backend delivery, and return traffic.

13 AERN Reference Architecture

13.1 Reference Architecture and Module Boundaries

The analysis treats AERN as a compact single-domain relay profile with explicit protocol-module boundaries. Global constants and protocol-size definitions are associated with `aern.h`; certificate processing with `certificate.c`;

topology ordering, hashing, and version handling with `topology.c`; administrative message families and tunnel-establishment messages with `network.c`; mesh encryption key state with `mek.c`; relay packet, route, session, dummy, delay, and backend-boundary functions with `route.c`; relay session state with `relaysession.c`; ingress queues with `relayqueue.c`; and fragment processing with `fragment.c`. The role modules `ars.c`, `adc.c`, `aps.c`, and `client.c` orchestrate the protocol functions for the four domain roles.

The topology state stores the monotonic topology version directly in the topology-list state structure. Version zero denotes uninitialized or disposed state. Mutating operations increment the version. Incoming topology versions used for replacement or versioned register-update processing are required to be fresh under the local policy. This structure is modeled directly in the topology acceptance and stale-update rejection analysis.

13.2 Topology Monotonicity

Definition 13.1 (Fresh Topology Update)**.** *Let $T = (v, L)$ be a local topology state with version v and serialized list L. Let $T' = (v', L')$ be an incoming signed topology update. T' is fresh for T if $v' > v$, the ADC signature over the update verifies, the ADC certificate chains to the ARS root, and L' is structurally valid.*

Theorem 13.2 (Stale Topology Rejection)**.** *Assume the ADC signature scheme is EUF-CMA secure and the implementation applies strict monotonic version checking before topology replacement. An adversary cannot cause an honest node to replace a nonzero topology state with an older authenticated topology state except by forging an ADC signature, compromising the ADC signing key, or exploiting an implementation error outside the modeled verification procedure.*

Proof. A replacement requires acceptance of an incoming topology update. Acceptance requires a valid ADC signature and a version greater than the local version. A stale update has version $v' \leq v$ once the local node has advanced beyond it, and is therefore rejected before replacement. To make a stale serialized list acceptable with a fresh version, the adversary must produce a valid ADC signature over the modified version and list, which is an EUF-CMA forgery unless the ADC signing key is compromised. The theorem follows from the monotonic check and signature unforgeability. □

13.3 Mesh Peer State

The mesh encryption key layer maintains peer state values representing absence of state, synchronization in progress, synchronized state, failed state, and expired state. A peer is route-usable only in synchronized state with installed transmit and

receive cipher state. This state machine is an implementation-level guard around the tunnel-security model. The formal tunnel games assume the tunnel has reached accepted synchronized state. Packets received under a failed, expired, or absent peer state are outside the accepted tunnel relation and must be rejected before relay processing.

Lemma 13.3 (Peer-State Admission)**.** *If route forwarding is restricted to synchronized peers and peer synchronization can be reached only after authenticated certificate validation and KEM-derived cipher-state installation, then a failed, expired, or uninitialized APS peer cannot be selected as a valid next hop by an honest forwarding implementation.*

Proof. The forwarding procedure resolves the next route hint against local topology and connection state. The conformance rule requires the resolved connection to be synchronized before forwarding. Failed, expired, and uninitialized states are not synchronized. Therefore, no such state satisfies the forwarding precondition. A selection would require bypassing the state check or corrupting local state, both of which are outside the honest forwarding procedure. □

13.4 Queue and Fragment State Isolation

AERN separates pending-session queues, optional randomized ingress-delay queues, relay session cache state, and fragment cache state. This separation is necessary for two reasons. First, non-dummy application data must not cross the backend boundary before the egress has explicitly accepted the session. Second, fragment reassembly must not allow unauthenticated, stale, direction-mismatched, or metadata-inconsistent fragments to influence backend delivery.

Theorem 13.4 (Pending Queue Safety)**.** *If the ingress implementation releases pending non-dummy data only in response to a valid session-open acknowledgement bound to the same session identifier and egress context, then packet delay, replay, or reordering cannot by itself cause early backend delivery.*

Proof. Delay and reordering do not change the session identifier or acknowledgement authentication result. Replay of an old acknowledgement is rejected by tunnel sequence and freshness checks unless the adversary produces a new accepted tunnel packet. A mismatched acknowledgement fails the session binding test. Consequently, the pending-to-active transition is reachable only through a valid acknowledgement for the same session and egress context, and early backend delivery cannot occur through scheduling manipulation alone. □

Theorem 13.5 (Fragment Cache Noninterference)**.** *Assume fragment insertion occurs only after tunnel authentication and that reassembly is keyed by session identifier, packet identifier, direction, payload type, flags, reserved byte, and fragment*

count. Then fragments from different sessions, packets, directions, or payload classes cannot be combined into one accepted backend message by an honest terminal APS.

Proof. The fragment cache key and consistency rules partition fragments by the listed fields. A fragment with a different session identifier, packet identifier, direction, payload type, flags, reserved byte, or count maps to a different cache entry or fails the consistency check for the existing entry. Because backend delivery requires a complete and internally consistent set, cross-session or cross-direction mixing cannot produce an accepted reconstructed message. □

13.5 Backend Adapter Security Boundary

AERN exposes backend-neutral callbacks as the transport boundary. The proof boundary is therefore the callback invocation carrying an authenticated, session-valid, and reassembled serialized packet. The external interface, operating-system network stack, socket adapter, TUN or TAP device, raw socket, userspace transport stack, or application adapter is not proven by the AERN relay proof. A backend that logs route paths, session identifiers, packet identifiers, fragment identifiers, or payload contents can defeat the operational privacy objective even though the relay cryptography remains intact.

14 Expanded Conformance and Test Obligations

A security proof for AERN depends on implementation conformance to the modeled state transitions. The following test obligations are therefore part of the scientific validation target rather than optional engineering hygiene.

14.1 Certificate and Control-Plane Tests

A complete conformance suite should verify root and child certificate serialization, certificate text encoding, root signature validation, expiration rejection, algorithm and version rejection, designation rejection, malformed certificate rejection, ADC-signed topology transfer, stale topology rejection, revocation effects, and every administrative message family under valid, malformed, stale, corrupted-signature, wrong-designation, and replayed inputs. Rejection tests should verify that topology, certificate cache, peer state, relay session state, queues, and fragment caches are not mutated after failed authentication or failed freshness validation.

14.2 Tunnel and Replay Tests

Tunnel tests should verify directional key separation, initialized transmit and receive sequence values, exact next-sequence acceptance, duplicate rejection, future-sequence rejection, stale timestamp rejection, modified outer-header rejection, modified ciphertext rejection, authentication-failure accounting, and no receive-sequence advancement on rejection. The strict replay profile uses a replay-window size of zero. Any out-of-order transport profile would require a separate theorem and separate tests.

14.3 Route, Session, Fragment, and Backend Tests

Route tests should cover route generation at the minimum hop count, maximum hop count, three-node boundary, invalid two-node domain, one-byte hint limits, terminal detection, next-hop consumption, and failure when the resolved next-hop peer is missing or not synchronized. Session tests should cover session-open acceptance, acknowledgement binding, pending queue release, timeout cleanup, return-flag handling, close, failed session cleanup, and backend callback isolation. Fragment tests should cover unfragmented delivery, maximum single-fragment payload, multi-fragment delivery, out-of-order completion after authentication, duplicate fragments, inconsistent fragment count, inconsistent length, wrong direction, wrong payload type, wrong reserved byte, timeout, and exactly-once backend delivery.

14.4 Virtual-Network Execution

A peer-reviewable validation suite should instantiate an ARS, an ADC, at least four APS nodes, a client, and a backend callback endpoint. It should execute certificate creation, ADC certification, remote signing when enabled, APS registration, topology convergence, peer certificate retrieval, MFK exchange and MEK state installation, APS synchronization, client registration or join, entry tunnel establishment, session-open, opaque outbound delivery, return delivery, fragmentation, dummy packet discard, ingress delay expiration, replay rejection, stale topology rejection, APS revocation, and teardown. A four-APS configuration is preferable to the mathematical minimum of three because it exercises interior route variation and avoids reducing the route sampler to a degenerate path set.

15 Conclusion

15.1 Formal Results

This paper presents a formal cryptanalysis of AERN as a controlled-domain authenticated relay protocol. Mutual authentication reduces to ARS certificate validation, ADC control-message authorization, signature unforgeability, and hash collision resistance. Tunnel-key indistinguishability reduces to KEM and KDF security. Weak forward secrecy holds for completed exchanges after erasure of ephemeral and active symmetric state. Per-hop confidentiality and integrity reduce to the tunnel cipher and authentication assumptions. Strict replay resistance follows from exact next-sequence validation and authenticated timestamps. Session correctness follows from the requirement that queued non-dummy data is released only after an authenticated session-open acknowledgement is accepted. Fragment integrity follows from authenticated encrypted fragment metadata and consistency checks.

15.2 Security Posture

AERN provides strong authenticated relay transport within a centrally administered topology. Its main cryptographic strengths are certificate-bound node admission, authenticated control messages, post-quantum tunnel establishment, fixed-size tunnel packets, encrypted route paths, encrypted session and fragment metadata, explicit session-open state, strict sequencing, and backend-boundary separation. Its main limitations are trust concentration in the ARS and ADC, the one byte route-hint limit of 255 route-addressable APS entries per active route domain, endpoint-correlation exposure under simultaneous ingress and egress compromise, and residual timing leakage inherent in low-latency transport.

15.3 Further Work

Further work should include public test vectors for the packet layout, independent verification of the route-path generation and consumption rules, a conformance suite for session and fragment state transitions, formal review of the RCS authenticated tunnel profile, operational guidance for ARS and ADC hardening, and a separately specified large-domain profile if deployments require more than 255 route-addressable APS ordinals in a single route context.

References

[1] John G. Underhill, *Authenticated Encrypted Relay Network - AERN: Technical Specification*, Quantum Resistant Cryptographic Solutions Corporation,

2026. Available: https://qrcscorp.ca/documents/aern_specification.pdf.

[2] Quantum Resistant Cryptographic Solutions Corporation, *Authenticated Encrypted Relay Network (AERN) source repository*, 2026. Available: https://github.com/QRCS-CORP/AERN.

[3] John G. Underhill, *The Design and Formal Analysis of RCS: A Quantum-Resilient AEAD Scheme*, Quantum Resistant Cryptographic Solutions Corporation, 2025. Available: https://qrcscorp.ca/documents/rcs_formal.pdf.

[4] Guido Bertoni, Joan Daemen, Michael Peeters, and Gilles Van Assche, "Keccak," in *Advances in Cryptology - EUROCRYPT 2013*, Lecture Notes in Computer Science, vol. 7881, Springer, 2013. DOI: https://doi.org/10.1007/978-3-642-38348-9_19.

[5] National Institute of Standards and Technology, *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*, FIPS PUB 202, August 2015. DOI: https://doi.org/10.6028/NIST.FIPS.202.

[6] John M. Kelsey, Shu-jen Chang, and Ray Perlner, *SHA-3 Derived Functions: cSHAKE, KMAC, TupleHash, and ParallelHash*, NIST Special Publication 800-185, December 2016. DOI: https://doi.org/10.6028/NIST.SP.800-185.

[7] Roberto Avanzi, Joppe Bos, Leo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehle, *CRYSTALS-Kyber Algorithm Specifications and Supporting Documentation*, version 3.02, 2021. Available: https://pq-crystals.org/kyber/data/kyber-specification-round3-20210804.pdf.

[8] National Institute of Standards and Technology, *Module-Lattice-Based Key-Encapsulation Mechanism Standard*, FIPS PUB 203, August 2024. DOI: https://doi.org/10.6028/NIST.FIPS.203.

[9] Leo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehle, *CRYSTALS-Dilithium: Algorithm Specifications and Supporting Documentation*, version 3.1, 2021. Available: https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf.

[10] National Institute of Standards and Technology, *Module-Lattice-Based Digital Signature Standard*, FIPS PUB 204, August 2024. DOI: https://doi.org/10.6028/NIST.FIPS.204.

[11] David L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84-90, February 1981. DOI: https://doi.org/10.1145/358549.358563.

[12] David M. Goldschlag, Michael G. Reed, and Paul F. Syverson, "Onion Routing for Anonymous and Private Internet Connections," *Communications of the ACM*, vol. 42, no. 2, pp. 39-41, February 1999. DOI: https://doi.org/10.1145/293411.293443.

[13] Roger Dingledine, Nick Mathewson, and Paul Syverson, "Tor: The Second-Generation Onion Router," in *Proceedings of the 13th USENIX Security Symposium*, 2004. Available: https://www.usenix.org/conference/13th-usenix-security-symposium/tor-second-generation-onion-router.

[14] George Danezis and Ian Goldberg, "Sphinx: A Compact and Provably Secure Mix Format," in *Proceedings of the 2009 IEEE Symposium on Security and Privacy*, 2009. Available: https://eprint.iacr.org/2008/475.

[15] George Danezis, Roger Dingledine, and Nick Mathewson, "Mixminion: Design of a Type III Anonymous Remailer Protocol," in *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, 2003. DOI: https://doi.org/10.1109/SECPRI.2003.1199323.

[16] Ania M. Piotrowska, Jamie Hayes, Tariq Elahi, Sebastian Meiser, and George Danezis, "The Loopix Anonymity System," in *Proceedings of the 26th USENIX Security Symposium*, 2017. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/piotrowska.