Network Working Group
Internet-Draft
Intended status: Informational
Expires: 17 September 2026

B.A. Fisher
DPA R&D Ltd (https://www.dpa-cloud.co.uk)

16 March 2026

TLS-DPA: An Identity-Bound Security Protocol for Traditional, Overlay, and Zero-Port Transports
draft-dpa-tls-dpa-01

Abstract

TLS-DPA is an experimental, identity-bound security protocol inspired by the design of TLS 1.3 ([RFC8446]). It is intended to operate consistently across environments where conventional IP address and port semantics are weak, unstable, or intentionally absent, including zero-port transports such as UZP ([UZP]).

TLS-DPA generalises the handshake so it is not tied to server-side listeners, binds authentication to Service Identities rather than network coordinates, reduces metadata exposure to intermediaries (including rendezvous nodes in UZP fabrics), provides a unified hybrid-KEM post-quantum transition model ([NIST-PQC]), and supports session continuity across overlay path changes (e.g., QUIC Connection IDs; [RFC9000]).

This document is part of an experimental, research-oriented Independent Stream suite. It defines the current normative baseline for trust objects, validation rules, and security semantics within its scope. Hard interoperability is expected for shared object semantics and validation rules. Full wire-level, clustering, and proof-family interoperability is not claimed everywhere yet; the remaining details are intentionally profile-defined or deferred where noted.

Note to Reviewers

This document is part of an experimental, research-oriented suite prepared for the Independent Stream. It is published to enable structured technical review, interoperability discussion, and disciplined specification development, and it remains a work-in-progress research artefact rather than a finished specification.

Within that suite, this revision defines the current normative baseline for trust objects, validation rules, and security semantics within TLS-DPA. Hard interoperability is expected for shared object semantics and validation rules. Full wire-level, clustering, and proof-family interoperability is not claimed everywhere yet; the remaining details are intentionally profile-defined or deferred where noted.

The name TLS-DPA is used to label this research protocol and avoid confusion with the IETF TLS versioning and registry space. It is not presented as a new version of the IETF TLS protocol, and no IANA allocations are requested by this draft.

Where this document provides numeric guidance (for example, replay windows, resumption behaviour, or profile parameters), the intent is to offer recommended bounds suitable for experimentation; profile-based behaviour and implementation discretion are explicitly expected within stated limits.

Reducing metadata exposure in some roles does not imply complete privacy or invisibility, and rendezvous or clustered deployments still require explicit availability assumptions and operational design.

Status of This Memo

Copyright Notice

Table of Contents

1. Scope and Status

This Internet-Draft is part of an experimental, research-oriented suite prepared for the Independent Stream. It specifies TLS-DPA for identity-first and topology-independent deployments, including rendezvous and zero-port fabrics, while remaining open to substantial revision through review and implementation experiments.

Within that suite, this document defines the current normative baseline for trust objects, validation rules, and security semantics within TLS-DPA, especially identity binding, transcript construction, and handshake authorisation. Hard interoperability is expected for shared object semantics and validation rules.

Full wire-level, clustering, and proof-family interoperability is not claimed everywhere yet; the remaining details are intentionally profile-defined or deferred, so this draft should not be read as claiming a fully closed wire image or proof model.

TLS-DPA is designed for environments where conventional port-listening assumptions and IP:port-based identity binding do not hold. It is designed for experimentation and profile-driven deployments within its target environment. Privacy, decentralisation, and availability remain deployment- and profile-dependent properties.

2. Introduction

TLS 1.3 ([RFC8446]) defines the current baseline for transport-layer security on the Internet. However, its usage patterns remain oriented around server-side listeners bound to IP address and port tuples, and many deployments treat these network coordinates as meaningful anchors for authentication and policy.

TLS-DPA extends the design principles of TLS 1.3 to support:

* operation over identity-first, topology-independent transports (for example UZP; [UZP]);

* authentication bound to Service Identities, rather than IP addresses and ports;

* reduced metadata exposure to intermediaries, including rendezvous nodes in UZP fabrics;

* hybrid classical/post-quantum KEM negotiation aligned with the NIST PQC process ([NIST-PQC]);

* session continuity across transport or overlay path changes (for example QUIC Connection IDs; [RFC9000]).

The eventual TLS-DPA wire image is intended to remain close to TLS 1.3, enabling reuse of existing implementation structure while adding explicit identity and transport binding into the handshake transcript and key schedule. This draft does not yet close every byte-level encoding, extension layout, or deployment profile.

TLS-DPA also aligns with zero-trust guidance (NIST SP 800-207 [NIST-SP800-207]) and identity-centric designs such as HIP [RFC7401].

This draft should therefore be read as part of an experimental, research-oriented Independent Stream suite and as the current normative baseline for trust objects, validation rules, and security semantics within TLS-DPA. Hard interoperability is expected for shared object semantics and validation rules. Full wire-level, clustering, and proof-family interoperability is not claimed everywhere yet; the remaining details are intentionally profile-defined or deferred. Reducing metadata exposure in some roles does not imply complete privacy or invisibility, and rendezvous or clustered deployment availability still depends on explicit operational choices.

3. Layering and Interoperability Baseline

This document is organised into four specification strata: core handshake semantics; transport binding rules; UZP / UZPIF applicability and profile rules; and optional or experimental extensions. The intent is to keep the core handshake and trust story stable even where exact extension code points, deployment profiles, or transport-specific optimisations remain open.

Sections Section 6, Section 8, Section 9, Section 10, Section 11, Section 13, Section 14, Section 16, and Section 17 define the core handshake semantics and trust decisions required for baseline

interoperability. Sections Section 7 and Section 12 define the transport binding rules that feed those semantics. Sections Section 18 and Section 19 define the UZP / UZPIF applicability profile. Section Section 20 is outside baseline interoperability unless a later profile explicitly upgrades it.

Baseline interoperable behaviour in this revision requires the core semantics for Service Identity binding, transcript construction, key schedule inputs, minimum revocation processing, Service Identity validation, and alert handling. Baseline interoperable transport binding further requires processing of tlsdpa_service_identity and tlsdpa_transport_binding. The tlsdpa_pq_kem_params extension is required when a negotiated deployment profile enables post-quantum or hybrid KEM operation.

Exact extension code points, final wire encodings, some revocation acquisition paths beyond the baseline processing rules, UZP early-data and rebind policy, and optional attribution metadata remain profile-dependent or experimental in this revision. Optional extensions MUST NOT alter authentication, authorisation, or trust outcomes unless a later profile explicitly upgrades them.

Sections Section 5, Section 15, Section 21, and Section 22 are explanatory or deployment-oriented unless they explicitly restate a normative baseline rule from the sections above.

4. Conventions and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] and [RFC8174] when, and only when, they appear in all capitals, as shown here.

Terminology used throughout this document:

CID Canonical Identity (a long-term public key hash).

EID Ephemeral Identity (a session-level fingerprint).

UZP Zero-port transport as defined by the companion UZP Internet-Draft ([UZP]).

ZPIT Zero-Port Interconnect Tunnel (a UZP fabric channel).

Pantheon A federated or deployment-scoped identity, attestation, and

policy plane whose authorities bind identity, policy, and trust metadata to keys or selectors accepted under local policy, may validate or certify those bindings, and may issue credentials, Grants, and delegations over them.

Service Identity The identity to which TLS-DPA authentication is bound (for example a DNS name, CID, EID, or a UZPIF selector).

Specification-Origin Identifier An OPTIONAL non-authoritative attribution metadata field carried by extension.

5. Design Goals

TLS-DPA is designed to:

1. decouple channel authentication from IP address and port topology;

2. provide identity-first naming independent of network routing;

3. support hybrid classical and post-quantum KEM negotiation aligned with NIST PQC guidance ([NIST-PQC]);

4. reduce metadata in early handshake flights;

5. bind channels to transport-level identifiers (for example UZP SessionIDs or QUIC Connection IDs; [RFC9000]);

6. remain closely aligned with the structure of TLS 1.3 ([RFC8446]);

7. operate efficiently over UZP and UZPIF rendezvous fabrics ([UZP] , [UZPIF]).

Where this document specifies algorithms or parameter sets (for example hybrid KEM combinations), these are intended as recommended profiles and may evolve. Implementations may support additional profiles and apply implementation-defined choices within any explicit limits described in the relevant sections.

6. Core TLS-DPA Semantics

TLS-DPA retains the basic architecture of TLS 1.3 ([RFC8446]) but introduces:

* transport-agnostic channel binding, via a dedicated extension that carries a transport identifier and class;

* Service Identity negotiation and binding into the transcript;

* mandatory transcript binding of identity and transport metadata;

* PQ-ready hybrid KEM negotiation using an explicit parameter extension;

* stable session resumption across topology or path changes.

TLS-DPA defines the handshake over an abstract TLS-DPA Channel. The channel only needs to provide:

* ordered or reliably framed delivery;

* a transport-level identifier (e.g., a TCP 4-tuple, QUIC Connection ID; [RFC9000] , or a UZP SessionID);

* uniqueness sufficient for transcript binding.

Together with Sections Section 8, Section 9, Section 10, Section 11, Section 13, Section 14, Section 16, and Section 17, this section defines the core TLS-DPA handshake semantics for baseline interoperability.

7. Transport Binding Rules

This section and Section Section 12 define the transport binding rules consumed by the core TLS-DPA handshake. Baseline interoperability requires consistent construction and verification of Service Identity and transport-binding inputs; exact code points and final byte-level encodings remain profile-defined in this revision.

TLS-DPA treats the underlying transport as providing one or more channels:

* *TCP* : traditional byte stream;

* *QUIC* : stream over a QUIC connection, identified by QUIC Connection ID ([RFC9000]);

* *UZP* : stream inside a ZPIT, identified by a UZP SessionID ([UZP]).

The handshake binds to this transport using the tlsdpa_transport_binding extension (see Section 12.2).

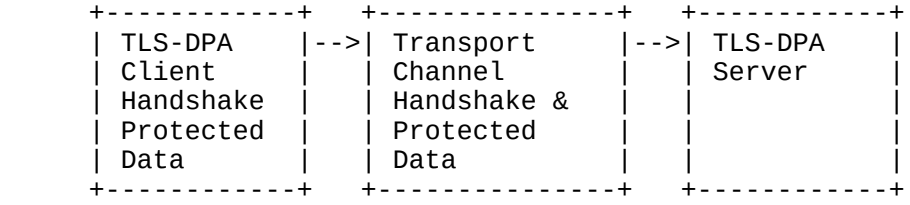

Figure 1: TLS-DPA operating over an abstract transport channel.

This figure places TLS-DPA above a transport channel to highlight the separation from the underlying relay.

8. Identity Binding Model

TLS-DPA authenticates peers using Service Identities, which may be:

* DNS names (validated per RFC 6125).
* UZP CIDs (canonical identities, derived from long-term public keys).
* UZP EIDs (ephemeral, session-level identities).
* UZPIF selectors resolved via Pantheon services, federated identity services, or local trust mappings [UZPIF].

The Service Identity MUST be included in the handshake transcript and validated as described in Section Section 16.

CIDs are intended to be stable over meaningful operational time-scales: changes in CID MUST be treated as key-rotation events and not as transient transport artefacts.

9. Identity Authority and Trust Model

TLS-DPA identity issuance is intentionally decentralisable. Service Identity credentials MAY be self-issued, multi-signed, federation-signed, or organisationally issued. Deployments MAY use one or more issuance models concurrently.

Where Pantheon is used in a UZPIF-aligned deployment, Pantheon authorities bind identity, policy, and trust metadata to keys or selectors accepted under policy; they may validate or certify those bindings and issue credentials, Grants, or delegations over them, but they need not generate or custody the underlying private keys.

TLS-DPA implementations MUST NOT require a single global signing authority. The protocol does not mandate a central certificate authority.

If any root trust model is used by a deployment profile, that model MUST be replaceable without protocol redesign.

9.1. Revocation Model

TLS-DPA revocation is policy-driven and decentralisable. The protocol does not define a mandatory single revocation authority and does not require any global CRL-equivalent service.

Baseline interoperable revocation in this revision is intentionally narrow: signed Revocation Signal objects, optional Threshold-Consensus Evidence, explicit acquisition and freshness rules, and fail-closed handling of unknown status for new admission decisions. Broader federation governance, quorum composition, and alternate revocation transports remain deployment- or profile-defined.

Interoperable decentralised revocation in TLS-DPA is defined here only when revocation is conveyed as explicit Revocation Signal objects (Section 9.1.1), evaluated under a recognised threshold policy and, where required, bound to Threshold-Consensus Evidence (Section 9.1.2), and processed according to the client rules in this section. Deployments that do not exchange such objects remain deployment-local and MUST NOT assume interoperable revocation semantics.

Revocation MAY be federation-scoped, multi-party threshold-based, and client-enforced. For interoperable exchange, Revocation Signals MUST be represented using the common signed artefact envelope defined by UZPIF ([UZPIF]) with "object_type" set to "revocation" and with epoch or sequence values suitable for conflict handling and freshness checks. That object type inherits the UZPIF common envelope unchanged, including canonical serialisation, exact signature coverage, object_id derivation, unknown-extension handling, signature ordering, algorithm identifier matching, epoch-versus-sequence precedence, and the rule that detached signatures are not part of baseline interoperability.

No single entity SHALL possess unilateral global revocation authority.

9.1.1. Revocation Signal Object

To support interoperable client behaviour, TLS-DPA defines a minimal Revocation Signal object. A Revocation Signal MUST use the common signed artefact envelope defined by UZPIF ([UZPIF]) with "object_type" set to "revocation", plus the following minimum revocation-specific body semantics:

* *subject_type:* identifies whether the revocation applies to a service identity, a Grant, an authority key, or a selector.
* *subject_identifier:* identifies the specific service identity, Grant, authority key, or selector being revoked.
* *issuer_authority_id:* identifies the authority context issuing the revocation signal.
* *scope:* defines the operational scope of the revocation, such as a tenant, service class, transport context, or selector namespace.
* *reason_code:* provides a deployment-defined reason such as key compromise, administrative withdrawal, policy violation, or supersession.
* *issue time:* states when the signal was issued.
* *expiry time:* indicates when the revocation signal ceases to apply.
* *optional review_time:* indicates a non-expiry re-evaluation point when deployments require review before the signal lapses.
* *threshold policy identifier:* identifies the quorum, threshold, or federation rule under which the signal is to be evaluated.
* *common-envelope signature set:* one or more signatures sufficient for the relying party to evaluate threshold satisfaction.
* *optional threshold-consensus evidence reference:* a pointer or digest referring to a separate threshold-consensus evidence object when quorum proof is carried out-of-line.
* *optional evidence reference:* a pointer or digest referring to supporting evidence when the revocation is evidence-backed.
* *optional transparency-log checkpoint reference:* a pointer to a transparency checkpoint or append-only log state relevant to the signal.

In terms of the common envelope, "subject_identifier" will normally map to "subject_id", "issue time" to "issued_at", "expiry time" to "not_after", and "threshold policy identifier" to "policy_id". If a separate "review_time" is needed, it belongs in the revocation-specific body and MUST NOT alter envelope validity semantics.

These fields extend only the revocation-specific body. TLS-DPA inherits the UZPIF common envelope unchanged, including canonical serialisation, exact signature coverage, object identifiers, unknown extension handling, signature ordering, algorithm identifier matching, epoch-versus-sequence precedence, and the rule that detached signatures are not part of baseline interoperability.

9.1.2. Threshold-Consensus Evidence Format

When revocation depends on multi-party or threshold consensus and interoperable exchange is required, implementations MUST represent the quorum result as a Threshold-Consensus Evidence object. This object MUST use the common signed artefact envelope defined by UZPIF ([UZPIF]), with "object_type" set to "threshold-consensus-evidence".

A minimal Threshold-Consensus Evidence object MUST carry:

* the referenced Revocation Signal identifier or digest;
* the threshold policy identifier;
* the required threshold or quorum rule;
* the participating authority identifiers or key identifiers;
* the signature set, signature digests, or equivalent verification evidence used to satisfy the threshold;
* the consensus evaluation result, such as satisfied, unsatisfied, or advisory;
* an evaluation timestamp;
* optional supporting evidence references; and
* optional transparency-log or checkpoint references anchoring the threshold evidence.

Threshold-Consensus Evidence MAY be embedded within the Revocation Signal or carried as a separate artefact referenced by it. If it is carried separately, the client MUST bind it to the exact Revocation Signal via digest, object identifier, or another unambiguous reference before using it for enforcement.

These fields populate the threshold-evidence body only and MUST NOT redefine the suite envelope semantics. Threshold-Consensus Evidence inherits the UZPIF common envelope unchanged, including canonical serialisation, exact signature coverage, object identifiers, unknown extension handling, signature ordering, algorithm identifier matching, epoch-versus-sequence precedence, and the rule that detached signatures are not part of baseline interoperability.

9.1.3. Revocation Signal Acquisition, Caching, and Freshness

A baseline interoperable client MUST be able to consume Revocation Signal (Section 9.1.1) and Threshold-Consensus Evidence (Section 9.1.2) objects presented by the peer during handshake or resumption. A client SHOULD also support authenticated out-of-band retrieval from authority, directory, or transparency channels so freshness can be re-established when the peer does not present current objects. Locally retained objects MAY be reused only within the freshness bounds of this section. Regardless of acquisition path, the client MUST authenticate the object under the common envelope rules before using it.

An accepted Revocation Signal, Threshold-Consensus Evidence object, or threshold-policy metadata item MUST be cached no longer than the earliest of: its expiry time; its "review_time" if present; replacement by a newer applicable object under epoch, sequence, or object-identifier precedence; or a stricter local maximum-staleness bound. In the absence of a stricter deployment profile, that local maximum-staleness bound MUST NOT exceed 24 hours from the most recent authenticated acquisition or freshness-confirmation event.

If current revocation state cannot be refreshed or its freshness cannot be re-established within those bounds, the client MUST classify the subject’s revocation status as "unknown" rather than silently treating the subject as clean or not revoked. For new handshakes, session resumption, and any authorisation-expanding transition, unknown status is fail-closed in the baseline model.

9.1.4. Baseline Revocation Processing

A TLS-DPA client or relying service processing a Revocation Signal (Section 9.1.1) or Threshold-Consensus Evidence (Section 9.1.2) MUST verify the object's signatures, algorithm acceptability, issuer authority context, validity interval, declared scope, freshness state, supersession state, and any applicable threshold policy before applying revocation effects. Minimum local processing MUST always determine whether the outcome is enforced revocation, advisory evidence, or unknown revocation state for the decision in question.

* If the locally recognised threshold is satisfied, the client MUST enforce revocation for the stated scope.

* If the signal is authentic but below the locally recognised threshold, it is advisory and MAY influence local risk policy, warnings, or connection decisions.

* Authenticity alone is insufficient: an otherwise valid Revocation Signal or Threshold-Consensus Evidence object MUST NOT be treated as current revocation truth if it is stale, superseded, scope-mismatched, or no longer policy-eligible for the decision being made.

* If the threshold policy is unknown, or if required threshold-consensus evidence is absent or cannot be bound to the Revocation Signal, the client MUST classify the result as unknown revocation state rather than as clean status.

* If a Revocation Signal or Threshold-Consensus Evidence object carries a transparency-log or checkpoint reference, the client SHOULD verify that reference before relying on the signal as quorum-backed evidence.

* For already-established sessions whose revocation freshness later becomes unknown, a deployment profile MAY define bounded fail-soft handling, but that behaviour is outside baseline interoperability. Absent such a profile, the endpoint MUST stop issuing resumption state and MUST block authorisation-expanding actions until freshness is re-established or the session is terminated.

* Clients MUST retain enough cached metadata to determine whether a previously enforced or observed revocation has expired, requires review, has been superseded by a newer signal, or has fallen into unknown freshness state.

* Conflicting revocation signals for the same subject and scope MUST be processed under local policy unless a deployment profile defines a stronger quorum or conflict-resolution rule.

This baseline does not attempt to solve global governance or federation-wide dispute resolution. It defines enough shared behaviour for interoperable handling of signed revocation artefacts, threshold evidence, and checkpoint-anchored revocation claims while leaving authority composition and quorum policy to deployment profiles.

10. Post-Quantum Key Exchange Model

TLS-DPA introduces unified hybrid-KEM negotiation via the tlsdpa_pq_kem_params extension. Supported KEM schemes include:

* X25519 (classical ECDH).

* Kyber768 (PQC KEM candidate).

* A hybrid X25519+Kyber768 mode.

The key schedule incorporates PQ KEM inputs prior to traffic secret derivation, following general design principles for hybrid KEMs in the NIST PQC process [NIST-PQC].

11. Key Schedule Summary

TLS-DPA modifies the TLS 1.3 key derivation [RFC8446] to include:

* Service Identity.

* Transport Binding.

* PQ KEM materials.

Exporter values MUST be bound to both identity and transport (Section Section 14).

At a high level, PQ hybrid KEM inputs augment the TLS 1.3 key schedule:

```
shared_secret = HKDF-Extract(kem_secret || ecdh_secret)
```

Figure 2: Hybrid shared secret extraction.

This equation shows the hybrid extraction step that combines KEM and ECDH inputs.

AEAD algorithms used with TLS-DPA MUST follow their specification-defined tag lengths. Tags MUST NOT be truncated below 96 bits, and 128-bit tags SHOULD be preferred where supported.

12. Transport Binding Extensions

This section carries the handshake extensions that bind the core TLS-DPA semantics into the transcript. Baseline interoperable behaviour requires tlsdpa_service_identity and tlsdpa_transport_binding. The tlsdpa_pq_kem_params extension is required when a negotiated deployment profile enables post-quantum or hybrid KEM operation.

The extension semantics in this section are normative where baseline interoperability requires them, but the example code points and C-like structures are illustrative and intended for experimentation. This draft does not request IANA allocations. Where appropriate, implementations may use private-use ranges or negotiated profiles while preserving the semantics defined here.

12.1. tlsdpa_service_identity

The tlsdpa_service_identity extension carries the Service Identity to which the TLS-DPA handshake is bound. For experimentation, this document uses an example private-use code point value (0xFE01); deployments MAY select alternative values by profile.

```
extension_type = 0xFE01

struct {
    ServiceIdentityType identity_type;
    opaque identity_value<1..2^16-1>;
} ServiceIdentity;

enum {
    dns_name(0),
    uzp_cid(1),
    uzp_eid(2),
    uzpif_selector(3),
    (255)
} ServiceIdentityType;
```

Figure 3: Service Identity extension structure (informative C-like syntax).

This figure shows the fields carried in the Service Identity extension.

The client MUST send exactly one Service Identity. The server MUST validate it according to its type (Section Section 16).

12.2. tlsdpa_transport_binding

The tlsdpa_transport_binding extension binds the handshake transcript to an underlying transport identifier and transport class. For experimentation, this document uses an example private-use code point value (0xFE02); deployments MAY select alternative values by profile.

```
extension_type = 0xFE02

struct {
    opaque transport_id<1..32>;
    uint8  transport_class;   /* 0=TCP, 1=QUIC, 2=UZP */
    opaque transport_params<0..256>;
} TransportBinding;
```

Figure 4: Transport Binding extension structure (informative C-like syntax).

This figure shows the fields used to bind the handshake to a transport identifier and class.

For UZP, transport_id MUST contain the UZP SessionID. For QUIC, it SHOULD contain the QUIC Connection ID [RFC9000].

12.3. tlsdpa_pq_kem_params

The tlsdpa_pq_kem_params extension carries the list of acceptable KEM schemes and related profile parameters. For experimentation, this document uses an example private-use code point value (0xFE03); deployments MAY select alternative values by profile.

```
extension_type = 0xFE03

struct {
    KEMScheme kem_list<2..2^8-1>;
} PQKemParams;

enum {
    x25519(0),
    kyber768(1),
    hybrid_x25519_kyber768(2),
    (255)
} KEMScheme;
```

Figure 5: PQ KEM parameters extension structure (informative C-like syntax).

This figure enumerates the acceptable KEM schemes and profile parameters.

The client proposes a list of acceptable KEM schemes. The selected scheme feeds into the key schedule.

13. Transcript Hashing Rules

New transcript components MUST be inserted as follows:

```
th = Hash(ClientHello
          || ServiceIdentity
          || TransportBinding
          || PQKemParams
          || ServerHello
          || ... )
```

Figure 6: Transcript hashing with identity and transport binding (illustrative).

This figure shows where the new identity and transport inputs are inserted into the transcript hash.

Hash mismatches MUST abort the handshake with illegal_transport_binding or identity_mismatch (Section Section 17).

14. Key Schedule and Exporters

PQ hybrid KEM inputs augment the TLS 1.3 key schedule as:

```
shared_secret = HKDF-Extract(kem_secret || ecdh_secret)
```

Figure 7: Hybrid shared secret extraction (illustrative).

This figure shows the shared secret input used for exporter derivation.

Exporter keys MUST incorporate identity and transport bindings.

14.1. Exporter Binding Requirements

TLS-DPA exporters MUST include:

* Service Identity (SID).

* Transport Binding (TB).

* PQ KEM scheme identifier.

* UZP SessionID (if transport_class = UZP).

14.2. Exporter Label Structure

```
label = "tlsdpa exporter" || 0x00 ||
  identity_type || transport_class
```

Figure 8: Exporter label structure (illustrative).

This figure shows the label composition that binds exporter output to identity and transport.

where identity_type is from ServiceIdentityType, and transport_class is from TransportBinding.

14.3. Exporter Context Structure

```
struct {
    opaque sid_hash[32];      /* BLAKE3-256 of Service Identity */
    opaque tb_hash[32];       /* BLAKE3-256 of TransportBinding */
    opaque kem_id[1];         /* selected KEM scheme */
} ExporterContext;
```

Figure 9: ExporterContext structure (informative C-like syntax).

This figure shows the exporter context fields derived from Service Identity, TransportBinding, and the selected KEM.

14.4. Example Exporter Computation

```
shared = HKDF-Extract(kem_secret || ecdh_secret);
ctx = ExporterContext(sid_hash, tb_hash, kem_id);
key = HKDF-Expand(shared, label, ctx, outlen);
```

Figure 10: Example exporter computation (illustrative).

This figure summarizes the exporter computation flow from shared secret to derived key.

15. Handshake Diagrams

15.1. Full UZP Flight Diagram

Figure 11 provides an illustrative end-to-end view of a TLS-DPA handshake relayed via a rendezvous node (RN) in a UZP fabric. The RN forwards handshake flights without decrypting them. Binding ensures the RN cannot replay or modify flows undetected.

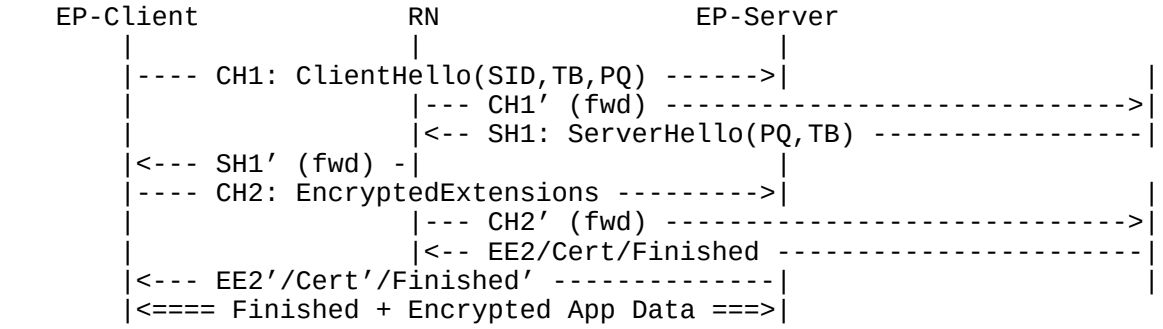

Figure 11: TLS-DPA handshake relayed via an RN, with end-to-end protection over the ZPIT (illustrative).

This figure traces the RN-relayed handshake flights while the endpoints retain end-to-end protection.

Where:

* SID: Service Identity.

* TB: Transport Binding (UZP SessionID mandatory).

* PQ: PQ KEM parameters.

15.2. Generalised TLS-DPA Flow

Figure Figure 12 shows a generalised view of the handshake and the role of a transport layer that relays flights but does not decrypt them.

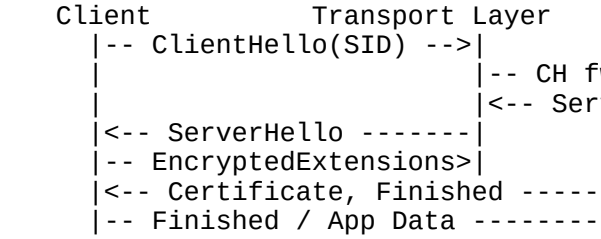

Figure 12: Generalised TLS-DPA handshake layers (illustrative).

This figure shows the transport relay separating TLS-DPA endpoints while preserving end-to-end security.

* SID is carried in the ClientHello.

* The transport layer relays flights but does not decrypt them.

* End-to-end Finished confirms key schedule integrity.

15.3. RN Observation and Channel Truth

Even when TLS-DPA is carried over a rendezvous path or a stitched UZP channel, authenticated channel truth is bound to endpoint identity inputs, transcript hashing rules (Section Section 13), transport binding, and Finished verification rather than to RN observation of packets, flow identifiers, or relay placement.

A rendezvous node or other relay MAY observe encrypted flights, timing, replay-related tuples, or local forwarding metadata, but it MUST NOT be treated as an authoritative source for Service Identity validation, transcript validity, or endpoint authentication state. The only authoritative handshake truth is the endpoint-generated material that validates under the transcript and Service Identity rules in Section Section 16.

Consequently, relay presence, stitched-path continuity, or RN-local telemetry MUST NOT upgrade, override, or substitute for endpoint-authenticated TLS-DPA state. Such observations may be useful for policy enforcement, replay suppression, or audit, but they are not trust anchors.

16. Service Identity Validation

16.1. DNS

DNS-based identities MUST be validated according to [RFC6125].

16.2. UZP CID

The CID MUST equal BLAKE3-256(server_longterm_public_key).

16.3. UZP EID

The EID MUST match the server-presented ephemeral identity for this session.

16.4. UZPIF Selector

UZPIF selectors MUST be resolved via Pantheon services, federated identity services, or local cached mappings consistent with local trust policy, Section Section 9, and [UZPIF].

16.5. Failure Handling

If any validation fails, the implementation MUST abort the handshake with an appropriate alert (Section Section 17):

* identity_mismatch;

* illegal_transport_binding;

* pq_required.

17. New Alerts

TLS-DPA defines the following experimental alert descriptions for use in deployments and interoperability testing. The numeric values shown are illustrative and are not requested for IANA allocation by this draft.

```
enum {
    illegal_transport_binding(200),
    identity_mismatch(201),
    pq_required(202),
    grant_invalid(203),
    grant_expired(204),
    (255)
} AlertDescription;
```

Figure 13: Experimental alert descriptions (illustrative C-like syntax).

This figure lists the experimental alert codes defined by TLS-DPA.

18. UZP / UZPIF Applicability and Profile

This section defines the UZP / UZPIF profile for TLS-DPA. It is required only for deployments that carry TLS-DPA over UZP or consume UZPIF trust artefacts; it does not redefine the core handshake semantics.

TLS-DPA maps naturally to UZP by binding:

* tlsdpa_service_identity -> UZP CID/EID.

* tlsdpa_transport_binding -> UZP SessionID.

* PQ capability and fallback -> Pantheon Grants.

UZP’s multi-step rendezvous and authentication model, together with the UZPIF framework defined in [UZP] and [UZPIF], provides:

* stronger pre-TLS identity establishment;

* reduced man-in-the-middle risk;

* deterministic channel binding for TLS-DPA.

This section and Section Section 19 are normative only for deployments that carry TLS-DPA over UZP or consume UZPIF trust artefacts. They do not redefine the core handshake semantics; they constrain how those semantics are applied in zero-port fabrics.

19. Early Data (0-RTT)

Over UZP:

* early data is transmitted inside a ZPIT;

* replay protection uses CID/EID and Pantheon Grant metadata;

* early data MUST NOT be used if Pantheon Grants specify "no-replay".

19.1. RN Replay Detection

The RN MUST maintain a sliding replay cache keyed on:

grant_nonce || CID || EID || SessionID

Figure 14: Replay cache key tuple (illustrative).

This figure shows the tuple the RN uses to index replay state.

Entries MUST be retained for at least twice the maximum ZPIT propagation delay. Longer retention is permitted. If a duplicate early-flight tuple is observed, the RN MUST drop it silently.

19.2. Endpoint Replay Detection

Endpoints MUST track Grant nonce values associated with early data. For each:

```
(grant_nonce, CID, EID, ticket_age)
```

Figure 15: Endpoint replay tuple (illustrative).

This figure shows the endpoint tuple tracked to detect early data replay.

If an identical tuple is received twice within the resumption window, the endpoint MUST abort with illegal_parameter.

19.3. Grant Nonce Interaction

Pantheon Grant issuers MUST issue or bind a fresh Grant nonce per resumed or 0-RTT-enabled session. The nonce MUST be bound into the handshake transcript.

19.4. 0-RTT over UZP Rebinds

If the UZP SessionID changes during path migration, 0-RTT data MUST be rejected unless the new SessionID is verifiably linked to the previous one via Pantheon metadata.

20. Optional and Experimental Extensions

The extensions in this section are not required for baseline interoperability. They MUST NOT alter authentication, authorisation, transcript truth, or trust-anchor decisions unless a later deployment profile explicitly upgrades them.

20.1. tlsdpa_specification_origin_id

The tlsdpa_specification_origin_id extension carries an OPTIONAL Specification-Origin Identifier for attribution metadata only. For experimentation, this document uses an example private-use code point value (0xFE04); deployments MAY select alternative values by profile.

```
extension_type = 0xFE04

struct {
    opaque origin_id<1..255>;
    opaque origin_uri<0..1024>;
} SpecificationOriginIdentifier;
```

Figure 16: Specification-Origin Identifier extension structure (informative C-like syntax).

This extension is non-authoritative and pure metadata. Endpoints MUST NOT treat it as an authentication, authorisation, or trust-anchor input.

Endpoints MAY log or display this metadata for attribution, but handshake success and identity validation MUST be independent of this extension.

21. Threat Model

TLS-DPA is designed to defend against:

* passive eavesdropping;
* active man-in-the-middle;
* downgrade attacks on both classical and PQ negotiation;
* identity spoofing;
* transport reattachment and rebinding attacks across overlays.

In UZP deployments, RN visibility is expected to be limited primarily to flow identifiers, relay context, and encrypted envelopes. Plaintext application data remains protected, but the exact visibility of Service Identity and related metadata depends on the transport profile and any additional confidentiality mechanisms in use.

22. Operational Considerations

* Middleboxes SHOULD NOT assume fixed IP/port semantics for TLS-DPA channels.
* Monitoring SHOULD use exporter-based identity hooks rather than IP/port heuristics [NIST-SP800-207].
* Session resumption MUST accommodate overlay rebinds (e.g., QUIC Connection IDs, UZP SessionIDs).
* PQ keys and related metadata SHOULD be logged where required for compliance, in line with local policy.

23. Security Considerations

TLS-DPA implementations MUST:

1. ensure identity and transport bindings are transcript-authentic;

2. authenticate PQ hybrid negotiation and detect downgrades;

3. suppress downgrade unless explicitly permitted by policy;

4. minimise metadata exposure, especially in early flights;

5. prevent unauthorised reattachment across transports or overlays.

6. apply revocation policy without assuming a single global revocation authority.

The threat model for TLS-DPA is discussed in Section Section 21.

24. IANA Considerations

This document does not request any IANA actions.

The example code points used for extension_type values and alert descriptions in this document are intended for experimentation (for example in private-use or locally coordinated deployments). Any future request for code point allocation is out of scope for this draft.

25. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <https://www.rfc-editor.org/info/rfc6125>.

26. Informative References

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

[RFC7401] Moskowitz, R., Ed., Heer, T., Jokela, P., and T. Henderson, "Host Identity Protocol Version 2 (HIPv2)", RFC 7401, DOI 10.17487/RFC7401, April 2015, <https://www.rfc-editor.org/info/rfc7401>.

[UZP] Fisher, B. A., "UZP: Universal Zero-Port Transport Protocol", Work in Progress, Internet-Draft, draft-dpa-uzp-transport, <https://datatracker.ietf.org/doc/html/draft-dpa-uzp-transport>.

[UZPIF] Fisher, B. A., "Universal Zero-Port Interconnect Framework (UZPIF)", Work in Progress, Internet-Draft, draft-dpa-uzpif-framework, <https://datatracker.ietf.org/doc/html/draft-dpa-uzpif-framework>.

[NIST-SP800-207] Rose, S., Borchert, O., Mitchell, S., and S. Connelly, "Zero Trust Architecture", NIST SP 800-207, 2019, <https://doi.org/10.6028/NIST.SP.800-207>.

[NIST-PQC] Technology, N. I. O. S. A., "NIST Post-Quantum Cryptography Standardization: Fourth Round Candidate Algorithms", 2022, <https://csrc.nist.gov/Projects/post-quantum-cryptography>.

Acknowledgements

The author thanks colleagues and early reviewers for discussions on identity-first security, transport binding, and post-quantum transition models. Any errors or omissions remain the author’s responsibility.

Author’s Address

Benjamin Anthony Fisher
DPA R&D Ltd (https://www.dpa-cloud.co.uk)
Email: b.fisher@dpa-cloud.co.uk
URI: https://orcid.org/0009-0004-4412-2269