Network Working Group
Internet-Draft
Intended status: Informational
Expires: 8 November 2026

J-I. Kadjo
Zenith Intelligence Technologies Inc.

8 May 2026

ZI2 Certified Email Server Attestation Protocol
draft-kadjo-zi2cert-01

Abstract

This document specifies the ZI2 Certified Email Server Attestation Protocol, a mechanism for cryptographic attestation of email origin at the mail server level. Unlike existing email authentication protocols (SPF, DKIM, DMARC), which operate at the domain level, ZI2 Certified operates at the server-instance level, enabling definitive identification of the specific server that dispatched a given email message. The protocol introduces two custom mail headers, a DNS TXT record publication format, a graduated trust model with four levels, and an integration interface for AI-based email classification systems. The protocol operates complementarily to existing email authentication infrastructure and introduces an ARC-Delegated Trust mechanism to preserve origin attestation across legitimate transit modifications.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 8 November 2026.

Table of Contents

1. Introduction

1.1. Problem Statement

The current email authentication ecosystem relies on three principal protocols: SPF [RFC7208], DKIM [RFC6376], and DMARC [RFC7489]. Each of these protocols operates at the domain level, answering questions about whether a domain authorized the sending of a message. None of these protocols, individually or in combination, answer a more fundamental question: "Did this specific server actually send this email?"

This distinction is critical in multi-tenant mail server environments, where a single server hosts multiple domains under a

shared IP address and MTA process. In such environments, domain-level authentication cannot distinguish between legitimate email from one co-hosted domain and spoofed email purporting to originate from another co-hosted domain. An attacker who compromises one domain's DNS records or DKIM keys can forge emails that pass SPF, DKIM, and DMARC validation for any domain on the shared server.

1.2. Scope

This document specifies a server-level cryptographic attestation protocol that:

1. Generates a unique Ed25519 [RFC8032] key pair per server instance.
2. Signs outbound email with a server-specific attestation header.
3. Publishes the server's public key via DNS TXT records.
4. Verifies inbound email bearing the attestation header against the published public key.
5. Assigns graduated trust levels reflecting the strength of the verification method.
6. Integrates with ARC to preserve trust across forwarding gateways.

1.3. Relationship to Existing Protocols

ZI2 Certified does not modify, replace, or interfere with SPF, DKIM, DMARC, ARC [RFC8617], BIMI, MTA-STS, or DANE [RFC7671]. It operates as a complementary attestation layer that addresses the server-level authentication gap left by existing domain-level protocols.

Specifically, ZI2 Certified addresses the structural limitation of cryptographic body hashes breaking during transit modifications (such as mailing list subject tagging or gateway footer injections). By natively integrating with the Authenticated Received Chain (ARC) [RFC8617], ZI2 Certified acts as the mathematically certain origin anchor for ARC. A trusted intermediary verifies the ZI2 signature before modifying the message and seals the result within the ARC chain, allowing final recipients to authenticate the origin despite transit alterations.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] and [RFC8174].

Certification: The process by which a sending mail server generates a cryptographic attestation of email origin and injects the attestation into the email as a structured header.

Verification: The process by which a receiving mail server validates a certification header by retrieving the sender server’s public key and verifying the cryptographic signature.

Certifying Server: A mail server that generates and injects ZI2 Certified headers on outbound email originating from its hosted domains.

Trust Level: A graduated integer value (0 through 3) representing the confidence in the verification result based on the method of public key retrieval.

Server Identifier (sid): The fully-qualified domain name (FQDN) of the certifying server.

3. Protocol Specification

3.1. Cryptographic Primitives & Canonicalization

The protocol uses Ed25519 [RFC8032] for signing, providing approximately 128 bits of security, deterministic signatures, 32-byte public keys, and 64-byte signatures. SHA-256 [RFC6234] is used for message body hash computation, and Base64 [RFC4648] is used for encoding.

MIME Decoding and Canonicalization: To ensure the SHA-256 hash survives routine MTA transfer-encoding conversions (e.g., 8BITMIME downgrades to 7-bit quoted-printable), implementations MUST decode any Content-Transfer-Encoding prior to hashing. Furthermore, implementations MUST apply RFC 6376 (DKIM) Relaxed Body Canonicalization to the decoded body content before computing the hash.

3.2. Key Generation

To provide true server-level attestation, the Certifying Server operates using a single cryptographic identity.

Upon initial deployment, the Certifying Server MUST generate a single Ed25519 key pair (the "Server DNA"). The private key MUST be stored securely within the MTA’s restricted filesystem and MUST NOT be accessible to individual tenants or the unprivileged SMTP process.

The key identifier MUST follow the format "zi2-cert-YYYY-MM".

3.3. Outbound Certification

Prior to certification, the ZI2 engine MUST verify two conditions:

1. The message contains exactly one RFC5322 "From:" header. If multiple "From:" headers are present, the engine MUST reject certification to prevent header smuggling.
2. The SMTP authenticated user (SASL login) or local envelope sender is explicitly authorized by local policy to send on behalf of the domain in the "From:" header.

Upon successful authorization, the Certifying Server MUST construct a structured canonical certification payload string. The payload string MUST be formatted as semi-colon delimited tag-value pairs in the following strict order:

```
v=1;
from=<lowercase value of the RFC5322 From: header>;
to=<lowercase value of the RFC5322 To: header, or empty if absent>;
cc=<lowercase value of the RFC5322 Cc: header, or empty if absent>;
rt=<lowercase value of the RFC5322 Reply-To header, or empty if
    absent>;
subj=<Base64-encoded value of the RFC5322 Subject header, or empty
      if absent>;
ts=<Unix timestamp>;
mid=<Message-ID value>;
sid=<server FQDN>;
kid=<Key identifier>;
bh=<Base64-encoded SHA-256 hash of the decoded and canonicalized
    message body>;
n=<Base64-encoded 128-bit cryptographically secure random nonce>;
```

Recipient Binding & Header Injection Rules: The engine MUST bind ONLY to the visible RFC5322 "To:" and "Cc:" headers. It MUST NOT bind to or inject envelope recipients (RCPT TO) into the header. Furthermore, all tags in the payload string are REQUIRED. If an optional header is absent in the outbound message, its corresponding tag MUST be included with an empty value (e.g., "subj=;"). During Inbound Verification (Section 3.4), if a tag is empty but the verifier detects that the corresponding header is present in the received message, the verification MUST fail.

The payload string MUST be signed with the Certifying Server’s Ed25519 private key. The resulting signature and payload components are injected into the outbound message as a single X-ZI2-Certified header.

3.4. Inbound Verification

The verifying server MUST check for the X-ZI2-Certified header, retrieve the public key, reconstruct the canonical structured payload, and verify the Ed25519 signature.

To retrieve the public key, the verifying server MUST extract the kid= tag from the X-ZI2-Certified header, and the domain from the RFC5322 "From:" header.

The verifier MUST perform a DNS TXT query for:
<kid>._zi2cert.<From-Domain>
(For example, if the header contains kid=zi2-srv1, the query is zi2-srv1._zi2cert.example.com)

The verifier MUST NOT retrieve the key based solely on the "sid=" tag, as the sid is an unverified claim until the signature is validated against the domain’s published key.

The verification engine MUST also validate:

1. Timestamp freshness: The timestamp MUST be within 24 hours of the verification time and MUST NOT be more than 5 minutes in the future.
2. Body hash integrity: The reconstructed SHA-256 hash of the local decoded body (after DKIM Relaxed Canonicalization) MUST match the "bh=" tag.
3. Header integrity: The values used to reconstruct the payload MUST match the visible headers of the received message.

If the signature verifies but the body hash integrity check fails, the message is presumed to have been modified in transit. In this scenario, the verifier SHOULD attempt ARC-Delegated Trust Verification as defined in Section 3.5.

3.5. ARC-Delegated Trust Verification

To gracefully support legitimate forwarding that breaks the full-body hash, verifying servers MUST support ARC-Delegated Trust:

1. The verifier SHALL evaluate the message for a valid Authenticated Received Chain (ARC) [RFC8617].
2. If a valid ARC chain exists, the verifier SHALL inspect the ARC-Authentication-Results (AAR) headers to locate the specific hop (i=N) that recorded the "zi2=pass" method result.
3. The verifier SHALL extract the ARC Sealer domain (the "d=" tag in the ARC-Message-Signature) for that exact hop (i=N).
4. The verifier SHALL evaluate that specific ARC Sealer domain against a locally configured Trusted Sealer Registry. It is explicitly insufficient for only the final hop of the ARC chain to be trusted; the hop asserting the ZI2 verification MUST be trusted.
5. If the ARC Sealer for hop i=N is trusted, and the chain validates, the verifier SHALL accept the delegated attestation. The final

verification result MUST be recorded as "arc_delegated_pass", and the Trust Level MUST be assigned as Level 1 (DNS).

If the ARC chain is invalid, the specific hop asserting the ZI2 result is untrusted, or the server identities do not align, the delegation MUST be rejected and the result recorded as "tampered".

4. Header Format

The X-ZI2-Certified header uses tag=value format separated by semicolons. The header MUST NOT contain redundant hashes of message fields that can be reconstructed directly from the email meta-data. All tags are REQUIRED:

- v: Protocol version (currently 1)
- t: Signing timestamp (Unix epoch seconds)
- sid: Server identifier (FQDN)
- bh: Base64-encoded SHA-256 hash of the decoded and canonicalized body
- n: Base64-encoded 16-byte random nonce
- sig: Base64 Ed25519 signature
- kid: Key identifier string

The X-ZI2-Verified header uses tag=value format:

- result: pass / spoofed / tampered / arc_delegated_pass / none
- level: 0, 1, 2, or 3
- reason: Machine-readable explanation string
- sid: Parsed server identifier
- ts: Verification timestamp

Base64 encoding MUST comply with [RFC4648] Section 4. Because Mail Transfer Agents (MTAs) frequently insert Folding White Space (FWS) to adhere to line length limits, verifiers MUST tolerate and strip whitespace, line-breaks, and carriage returns within Base64 strings prior to decoding.

5. DNS Publication

A certifying server MUST instruct each hosted domain to publish the server’s public key as a DNS TXT record at the subdomain "_zi2cert".

Format: "v=ZI2CERT1; k=ed25519; p=<base64_server_public_key>; kid=<key_id>"

The explicit version string "v=ZI2CERT1;" ensures strict segregation from DKIM and DMARC parsers.

Because multiple domains co-hosted on the same server publish the exact same server public key, the DNS record provides cryptographic proof that the domain owner explicitly authorizes this specific server instance’s identity to dispatch its mail.

5.1. Key Rollover and Rotation

To maintain cryptographic hygiene or recover from a suspected compromise, the Certifying Server MUST support non-disruptive key rotation of the Server Key. Servers MUST implement a transitional overlap window to prevent transit validation failures caused by DNS propagation delays.

1. The Certifying Server generates the new Ed25519 Server Key.
2. The server instructs all hosted domains to publish the new public key as a second, concurrent DNS TXT record alongside the existing record (using a new "kid=" identifier, e.g., kid=zi2-cert-2026-11).
3. The server waits a sufficient period (RECOMMENDED 48 hours) to ensure global DNS propagation of the new records.
4. The server switches the ZI2 engine to sign new outbound messages using the new key, injecting the new "kid=" value into the header.
5. After the 24-hour timestamp freshness window expires for the last messages signed by the old key, the server instructs tenants to safely remove the old DNS TXT records.

Verifying servers easily accommodate this by strictly using the "kid=" tag in the incoming header to query the correct corresponding DNS record, ensuring in-flight emails are not invalidated during the transition.

6. Trust Levels

- Level 3 (ABSOLUTE): Intra-server verification. The signing and verifying keys are mathematically linked locally. No network query required. Provides mathematical certainty.
- Level 2 (VERIFIED): Key obtained via direct bilateral exchange.
- Level 1 (DNS): Key obtained via DNS TXT record or HTTPS endpoint.
- Level 0 (UNKNOWN): No certification present or verification failed.

6.1. AI Systems Integration Interface

Modern email defense systems rely heavily on Machine Learning (ML) and Large Language Models (LLMs) to detect Business Email Compromise (BEC) and spear-phishing. These AI engines traditionally evaluate probabilistic signals, such as stylometry, semantic urgency, and domain reputation.

The ZI2 protocol provides a deterministic, cryptographic origin signal that allows AI systems to anchor their analysis in mathematical fact. When an AI classification engine or downstream filter ingests an email bearing an X-ZI2-Verified header, it SHOULD map the results to its weighting engine as follows:

1. Levels 3 (ABSOLUTE) and 2 (VERIFIED): The origin is mathematically certain. The AI engine SHOULD bypass probabilistic sender-identity heuristics (e.g., "Does this sound like the CEO?") and rely on the verified server identity.
2. Level 1 (DNS) with "pass" or "arc_delegated_pass": The engine SHOULD positively weight the server-origin identity in its anomaly detection models, utilizing the "sid=" tag as a stable reputation anchor.
3. Result "spoofed" or "tampered": The engine MUST flag the message as a high-confidence attack, explicitly overriding any positive semantic scores or domain-reputation weights. Cryptographic failure supersedes linguistic legitimacy.

7. Security Considerations

DNS Poisoning: Affects Trust Level 1 only. DNSSEC [RFC4033] SHOULD be deployed to secure the "_zi2cert" TXT records.

Replay Attacks: Mitigated by the 128-bit nonce, the strict 24-hour timestamp freshness window, and the cryptographically bound Message-ID and Recipient fields. Verifiers MAY implement a stateful nonce cache to detect short-term replays within the 24-hour window, though the timestamp and Message-ID binding provide significant mitigation in stateless environments.

Forwarding Abuse: An attacker cannot forge an ARC chain to bypass a broken ZI2 signature because the verifier strictly requires the ARC sealing domain to reside in a Local Policy Trusted Sealer Registry, and further enforces sid-to-domain alignment to prevent cross-domain delegation spoofing.

Cross-Tenant Spoofing: Strictly prevented by the mandatory explicit authorization check required in Section 3.3. The ZI2 engine securely attests to the server’s origin, but it fundamentally relies on the integrity of the MTA’s local SASL or envelope authentication boundary to authorize the use of the server’s signing key for a specific domain.

8. IANA Considerations

This document requests the following actions from IANA.

8.1. Email Authentication Methods Registry

This document requests that IANA add a new method to the "Email Authentication Methods" registry (created by [RFC7601] and updated by [RFC8601]).

- Method: zi2
- Defined In: [RFC-TBD]
- ptype: header
- property: sid
- Value: The fully-qualified domain name (FQDN) of the certifying server, extracted from the "sid=" tag of the X-ZI2-Certified header.

8.2. Email Authentication Result Names Registry

This document requests that IANA add the following result codes to the "Email Authentication Result Names" registry for the "zi2" method:

- Code: pass
- Description: The ZI2 Certified signature and body hash were successfully verified directly by the receiving MTA.

- Code: spoofed
- Description: A mandatory ZI2 Certified signature was missing from an email asserting to originate from a known local domain.

- Code: tampered
- Description: The ZI2 Certified signature or body hash failed verification, and no trusted ARC delegation was present.

- Code: arc_delegated_pass
- Description: The direct ZI2 signature failed due to transit modification, but a "pass" result was successfully delegated via a valid Authenticated Received Chain (ARC) from a trusted sealer.

8.3. Message Header Field Registry

This document requests that IANA register the following new header fields in the "Message Header Field" registry:

- Header Field Name: X-ZI2-Certified
- Protocol: mail
- Status: informational
- Reference: [RFC-TBD]

- Header Field Name: X-ZI2-Verified
- Protocol: mail
- Status: informational
- Reference: [RFC-TBD]

9. References

9.1. 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>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/info/rfc4648>.

[RFC6234] Eastlake 3rd, D. and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", RFC 6234, DOI 10.17487/RFC6234, May 2011, <https://www.rfc-editor.org/info/rfc6234>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/info/rfc8032>.

[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>.

[RFC8617] Andersen, K., Long, B., Ed., Blank, S., Ed., and M. Kucherawy, Ed., "The Authenticated Received Chain (ARC) Protocol", RFC 8617, DOI 10.17487/RFC8617, July 2019, <https://www.rfc-editor.org/info/rfc8617>.

9.2. Informative References

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <https://www.rfc-editor.org/info/rfc4033>.

[RFC6376] Crocker, D., Ed., Hansen, T., Ed., and M. Kucherawy, Ed., "DomainKeys Identified Mail (DKIM) Signatures", STD 76, RFC 6376, DOI 10.17487/RFC6376, September 2011, <https://www.rfc-editor.org/info/rfc6376>.

[RFC7208] Kitterman, S., "Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1", RFC 7208, DOI 10.17487/RFC7208, April 2014, <https://www.rfc-editor.org/info/rfc7208>.

[RFC7489] Kucherawy, M., Ed. and E. Zwicky, Ed., "Domain-based Message Authentication, Reporting, and Conformance (DMARC)", RFC 7489, DOI 10.17487/RFC7489, March 2015, <https://www.rfc-editor.org/info/rfc7489>.

[RFC7601] Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", RFC 7601, DOI 10.17487/RFC7601, August 2015, <https://www.rfc-editor.org/info/rfc7601>.

[RFC7671] Dukhovni, V. and W. Hardaker, "The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance", RFC 7671, DOI 10.17487/RFC7671, October 2015, <https://www.rfc-editor.org/info/rfc7671>.

[RFC8601] Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", RFC 8601, DOI 10.17487/RFC8601, May 2019, <https://www.rfc-editor.org/info/rfc8601>.

Author’s Address

Joseph-Israel Kadjo
Zenith Intelligence Technologies Inc.
Hagerstown, MD 21740
United States of America
Email: dev@zi2.app
URI: https://zi2.app