Remote ATtestation procedureS
Internet-Draft
Intended status: Standards Track
Expires: 11 August 2026

D. Condrey
Writerslogic Inc

7 February 2026

Trust Anchor Bootstrap Protocol for Proof of Process
draft-condrey-rats-witnessd-enrollment-00

Abstract

This document specifies a trust anchor bootstrap protocol for the witnessd Proof of Process framework. The protocol defines how new devices enter the witnessd ecosystem, how device keys are provisioned and verified, and how Verifiers discover and validate device trust anchors.

Three enrollment modes are defined: self-sovereign mode for individual users, organizational mode for enterprise deployments, and public registry mode for federated trust ecosystems.

About This Document

This note is to be removed before publishing as an RFC.

This is a companion document to draft-condrey-rats-pop, which defines the core Proof of Process evidence framework. This document addresses the bootstrapping problem: how do Verifiers know which device keys to trust?

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 11 August 2026.

Table of Contents

1. Introduction

The Proof of Process (PoP) specification [I-D.condrey-rats-pop] defines how Attesters generate Evidence during document authorship. Evidence packets are signed with device keys, but the core specification does not define how these keys are provisioned or how Verifiers establish trust in them.

This document addresses the trust anchor bootstrap problem by defining:

* Device enrollment protocols for provisioning signing keys
* Trust anchor discovery mechanisms for Verifiers
* Multiple enrollment modes for different deployment scenarios
* Key lifecycle management including rotation and migration

1.1. Requirements Notation

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Enrollment Modes

2.1. Self-Sovereign Mode

In self-sovereign mode, users generate their own device keys and manually provide trust anchors to Verifiers they wish to work with. This mode provides maximum privacy but requires out-of-band trust establishment.

Workflow:

1. Device generates key pair in secure hardware (if available)
2. Device exports public key and self-signed certificate
3. User manually provides public key to Verifier (via secure channel)
4. Verifier stores trust anchor associated with user identity

Use cases: Individual authors, privacy-conscious users, testing environments.

2.2. Organizational Mode

In organizational mode, device keys are signed by an organization's Certificate Authority (CA). The organization publishes its CA certificate, and Verifiers trust devices whose keys chain to the organization's CA.

Workflow:

1. Organization establishes CA infrastructure
2. Organization publishes CA certificate at well-known location
3. Device generates key pair and creates certificate signing request
4. Organization signs device certificate (may require proof of device identity)
5. Device stores signed certificate
6. Verifiers validate device certificates against organization CA

Use cases: Enterprises, academic institutions, publishing houses.

2.2.1. Well-Known URI for Device Discovery

Organizations SHOULD publish device trust information at:

/.well-known/witnessd-devices

This endpoint returns a signed JSON document listing:

* Organization CA certificate chain
* Supported enrollment methods
* Device certificate validation policy
* Contact information for enrollment requests

2.3. Public Registry Mode

In public registry mode, devices register with a public discovery service, similar to ACME for TLS certificates. This enables federated trust without requiring prior relationship establishment.

Workflow:

1. Device generates key pair

2. Device proves control of user identifier (email, domain, etc.)

3. Registry issues device certificate bound to user identifier

4. Registry publishes certificate in transparency log

5. Verifiers query registry to validate device certificates

Use cases: Open ecosystems, cross-organization verification, consumer applications.

3. Device Enrollment Protocol

The device enrollment protocol establishes trust between a new device and the enrollment authority. The protocol uses a challenge-response flow to verify device identity and key possession.

3.1. Enrollment Request

Devices initiate enrollment by submitting a request containing:

```
enrollment-request = {
  device-public-key: bstr,      ; Device's public key (COSE_Key)
  user-identifier: tstr,        ; Email, domain, or opaque ID
  enrollment-mode: mode-type,
  ? device-attestation: bstr,   ; TPM/Secure Enclave attestation
  ? proof-of-possession: bstr,  ; Signature over challenge
  nonce: bstr,                  ; Freshness nonce
  timestamp: time
}

mode-type = &(
  self-sovereign: 0,
  organizational: 1,
  public-registry: 2
)
```

3.2. Challenge-Response Flow

For organizational and public registry modes, the enrollment authority issues a challenge:

1. Device submits initial enrollment request

2. Authority returns challenge (random nonce + user verification requirement)

3. Device signs challenge with private key (proof of possession)

4. User completes verification (email link, domain TXT record, etc.)

5. Device submits signed challenge and verification proof

6. Authority validates and issues device certificate

```
enrollment-challenge = {
  challenge-nonce: bstr,
  verification-method: verification-type,
  ? verification-uri: tstr,    ; For email/web verification
  expires: time
}

verification-type = &(
  email-link: 0,
  domain-txt: 1,
  organizational-approval: 2,
  device-attestation-only: 3
)

enrollment-response = {
  challenge-nonce: bstr,
  challenge-signature: bstr,   ; Signed by device private key
  verification-token: tstr,    ; From email/domain verification
  timestamp: time
}
```

3.3. Certificate Issuance

Upon successful verification, the authority issues a device certificate:

```
device-certificate = {
  device-id: bstr,              ; Unique device identifier
  public-key: bstr,             ; COSE_Key
  user-identifier: tstr,
  issuer: tstr,                 ; Authority identifier
  issued-at: time,
  expires-at: time,
  ? constraints: [* constraint],
  signature: bstr               ; Authority signature
}

constraint = &(
  max-evidence-per-day: uint,
  allowed-document-types: [* tstr],
  geographic-restriction: tstr
)
```

Certificates MUST be valid for no more than 1 year. Devices MUST re-enroll before certificate expiration.

4. Trust Anchor Discovery

Verifiers must discover and maintain trust anchors for device certificate validation. This section defines discovery mechanisms and caching policies.

4.1. Discovery Mechanisms

Verifiers discover trust anchors through:

Well-known URI: Query /.well-known/witnessd-devices at the organization's domain to retrieve CA certificates and enrollment policies.

Public Registry: Query the public registry service for device certificates bound to user identifiers. The registry provides certificate chains and revocation status.

Out-of-band provisioning: For self-sovereign mode, trust anchors are manually configured by the Verifier operator.

4.2. Caching Policy

Verifiers SHOULD cache trust anchors to reduce latency and network load. Caching policies:

* CA certificates: Cache for up to 24 hours or until the next-update field indicates refresh is needed

* Device certificates: Cache for the certificate validity period minus a safety margin (RECOMMENDED: 1 hour)

* Revocation status: Cache according to revocation list TTL (see draft-condrey-rats-witnessd-revocation)

Verifiers MUST refresh cached trust anchors before expiration. Stale cache entries MUST NOT be used for verification.

4.3. Revocation Checking Integration

Trust anchor discovery integrates with revocation checking:

1. Verifier receives Evidence packet

2. Verifier extracts device certificate from Evidence

3. Verifier validates certificate chain to trusted CA

4. Verifier checks certificate revocation status

5. Verifier checks Evidence revocation status

6. If all checks pass, Verifier proceeds with appraisal

Certificate revocation and Evidence revocation are independent. A revoked certificate invalidates all Evidence signed by that device. Evidence may be revoked even if the certificate remains valid.

5. Key Lifecycle Management

Device keys have a lifecycle from generation through retirement. This section defines key management operations.

5.1. Key Rotation

Devices SHOULD rotate signing keys periodically to limit exposure from potential compromise. Key rotation workflow:

1. Device generates new key pair

2. Device requests certificate for new key (standard enrollment)

3. Device publishes key rotation notice signed by old key

4. Rotation notice links old key to new key

5. Old key enters grace period (RECOMMENDED: 30 days)

6. After grace period, old key is retired

```
key-rotation-notice = {
  old-device-id: bstr,
  new-device-id: bstr,
  old-public-key: bstr,
  new-public-key: bstr,
  rotation-time: time,
  grace-period-end: time,
  old-key-signature: bstr,    ; Proves control of old key
  new-key-signature: bstr     ; Proves control of new key
}
```

Evidence signed during the grace period MAY use either key. After the grace period, only the new key is valid.

5.2. Device Migration

When users migrate to a new device, Evidence continuity must be maintained. Migration is treated as key rotation with additional verification:

* User initiates migration from old device (if available)

* Old device signs migration authorization for new device

* New device completes enrollment with migration authorization

* If old device is unavailable, user completes out-of-band verification (email, organizational approval, etc.)

Migration without old device access requires stronger verification to prevent unauthorized device takeover.

5.3. Key Compromise Response

When a device key is compromised:

1. User reports compromise to enrollment authority

2. Authority revokes device certificate immediately

3. User enrolls new device with fresh key

4. User may re-endorse historical Evidence with new key (see draft-condrey-rats-witnessd-revocation)

Hardware-backed keys (TPM, Secure Enclave) reduce compromise risk and are RECOMMENDED for high-security deployments.

5.4. Certificate Renewal

Devices MUST renew certificates before expiration. Renewal may use the existing key (if not compromised) or coincide with key rotation. Renewal workflow:

1. Device submits renewal request with current certificate
2. Authority validates current certificate is still valid
3. Authority issues new certificate (same or new key)
4. New certificate validity period begins from issuance

Devices SHOULD initiate renewal at least 7 days before certificate expiration to allow for processing delays.

6. Security Considerations

6.1. Threat Model

The enrollment protocol considers the following threats:

Unauthorized enrollment: Attacker attempts to enroll a device under another user's identity. Mitigated by user identifier verification (email, domain, organizational approval).

Key extraction: Attacker attempts to extract device private key. Mitigated by hardware-backed key storage (TPM, Secure Enclave) where available.

Enrollment authority compromise: Attacker compromises the enrollment authority and issues fraudulent certificates. Mitigated by certificate transparency logs and short certificate validity periods.

Man-in-the-middle: Attacker intercepts enrollment messages. Mitigated by TLS for all enrollment communications and challenge-response binding.

Replay attacks: Attacker replays old enrollment requests. Mitigated by nonces and timestamps in all protocol messages.

6.2. Compromise Impact Analysis

The impact of various compromises:

* Single device key compromise: Only Evidence from that device is affected. Other devices and users are unaffected.

* Organizational CA compromise: All devices enrolled under that CA are potentially affected. Requires CA re-keying and mass device re-enrollment.

* Public registry compromise: All devices using that registry are potentially affected. Requires registry recovery and certificate reissuance.

Organizational deployments SHOULD use Hardware Security Modules (HSMs) for CA key protection. Public registries MUST implement multi-party controls for certificate issuance.

6.3. Mitigation Requirements

Implementations MUST:

* Use TLS 1.3 or later for all enrollment communications

* Validate certificate chains completely before trusting

* Check certificate revocation status before accepting Evidence

* Implement rate limiting to prevent enrollment abuse

* Log enrollment events for audit purposes

Implementations SHOULD:

* Use hardware-backed key storage where available

* Implement certificate transparency monitoring

* Support multiple enrollment authorities for redundancy

* Provide key compromise notification mechanisms

7. Privacy Considerations

7.1. Device Identifier Privacy

Device identifiers enable tracking across Evidence packets. To protect user privacy:

* Device identifiers SHOULD be opaque random values, not derived from hardware identifiers or user information

* Users MAY request new device identifiers during key rotation to break tracking linkage

* Verifiers MUST NOT share device verification patterns with third parties

Self-sovereign mode provides the strongest privacy as no central authority learns about device enrollment. Public registry mode has the weakest privacy as the registry sees all enrolled devices.

7.2. Enrollment Metadata Minimization

Enrollment authorities collect metadata during device registration. To minimize privacy impact:

* Collect only the minimum information necessary for enrollment

* Do not retain IP addresses or location data beyond enrollment completion

* Do not share enrollment metadata with third parties

* Provide data deletion upon user request (subject to legal retention requirements)

Enrollment authorities MUST publish a privacy policy describing data collection, retention, and sharing practices.

7.3. Registry Privacy Modes

Public registries MAY offer privacy modes:

Public mode: Device certificates are publicly discoverable. Anyone can verify Evidence from the device. Suitable for public authors.

Unlisted mode: Device certificates are not publicly listed but can be retrieved by packet-id lookup. Provides some obscurity.

Private mode: Device certificates are only returned to authenticated

Verifiers with a legitimate need. Suitable for sensitive contexts.

Users SHOULD select the privacy mode appropriate for their use case. The default mode is deployment-specific.

8. IANA Considerations

This document requests registration of the following well-known URI:

* URI suffix: witnessd-devices
* Change controller: IETF
* Specification document: [this document]

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, 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, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

[I-D.condrey-rats-pop] Condrey, D., "Proof of Process: An Evidence Framework for Digital Authorship Attestation", Work in Progress, Internet-Draft, draft-condrey-rats-pop, <https://datatracker.ietf.org/doc/html/draft-condrey-rats-pop>.

Author's Address

David Condrey
Writerslogic Inc
United States
Email: david@writerslogic.com