

# Bridging the Transparency Gap: Distributed Remote Attestation

Making selected attestation artefact reusable across verifiers and trust domains

source: [draft-wang-rats-distributed-remote-attestation-02](#)

DongHui Wang

Faye Liu

**Yuning Jiang**

Jun Zhang

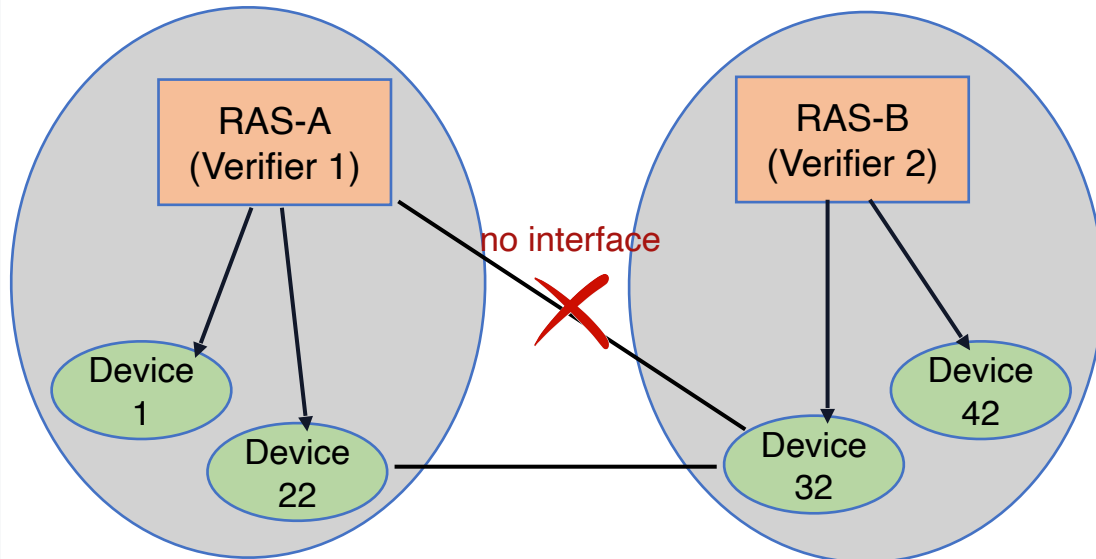
— Huawei

# Why This Matters ?

Problem

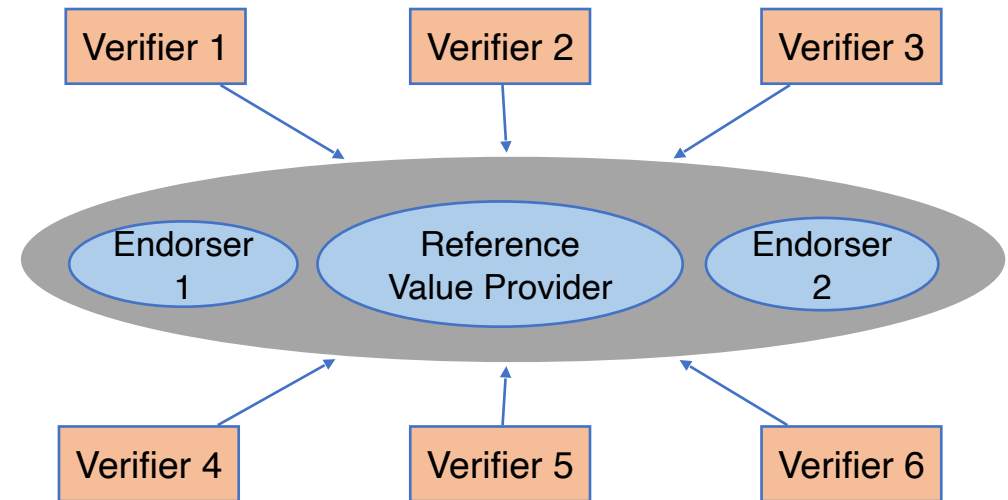
## Cross-domain attestation transparency

A verifier or relying party in domain-A may need endorsement material, reference values, or attestation results that originate in domain-B. Direct point-to-point integration between operational sites does not scale and is often infeasible due to operational and policy constraints.



## Many-to-many distribution and reuse of security artefact

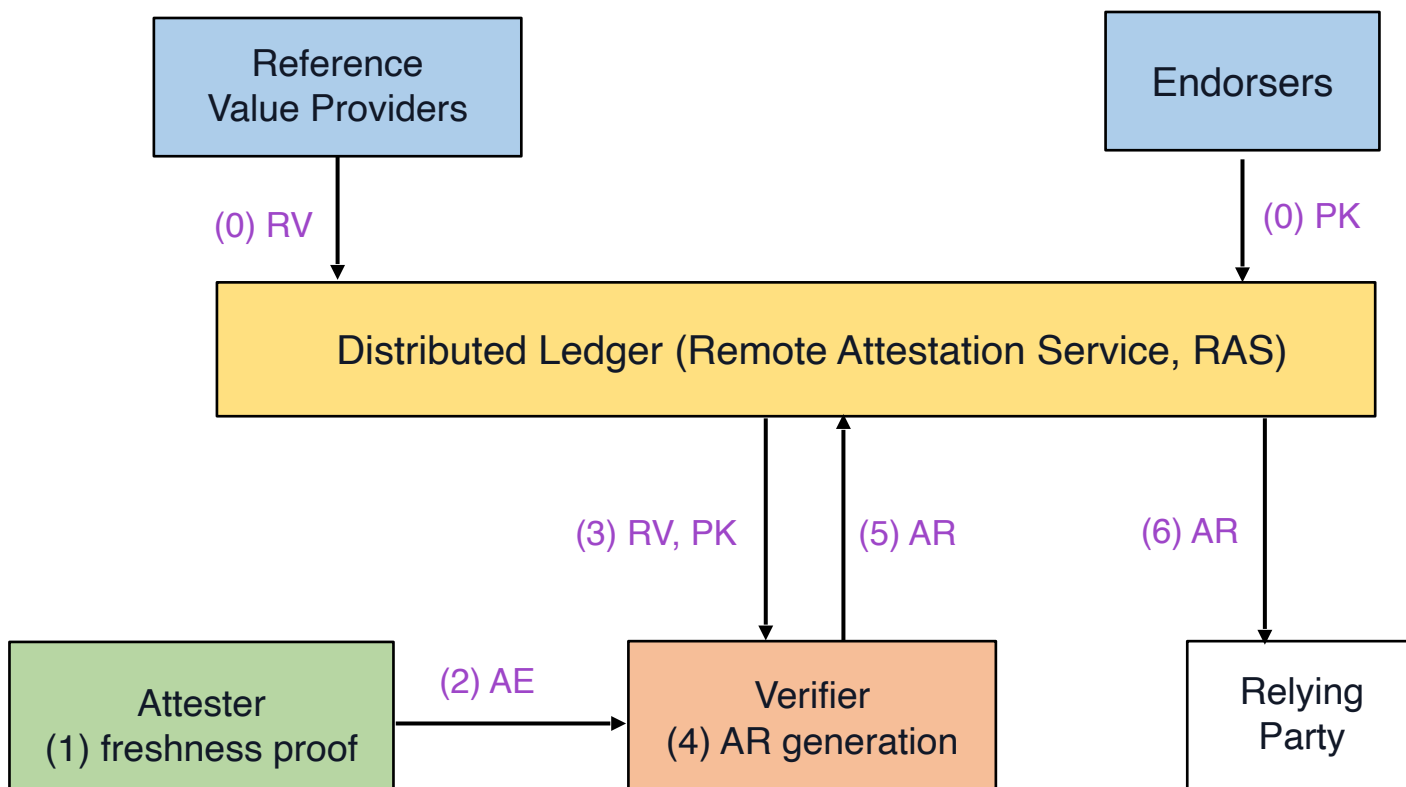
Verifiers need endorser public keys, endorsements, and reference values from multiple providers. Providers need to distribute artefact to multiple verifiers across domains. Similar many-to-many scaling issues apply when attestation results are shared for reuse.



**Without a shared publication channel, each integration becomes a bespoke, brittle dependency.**

# Pattern 1: DL Publication with Off-Chain Attestation and Verification

Pattern A

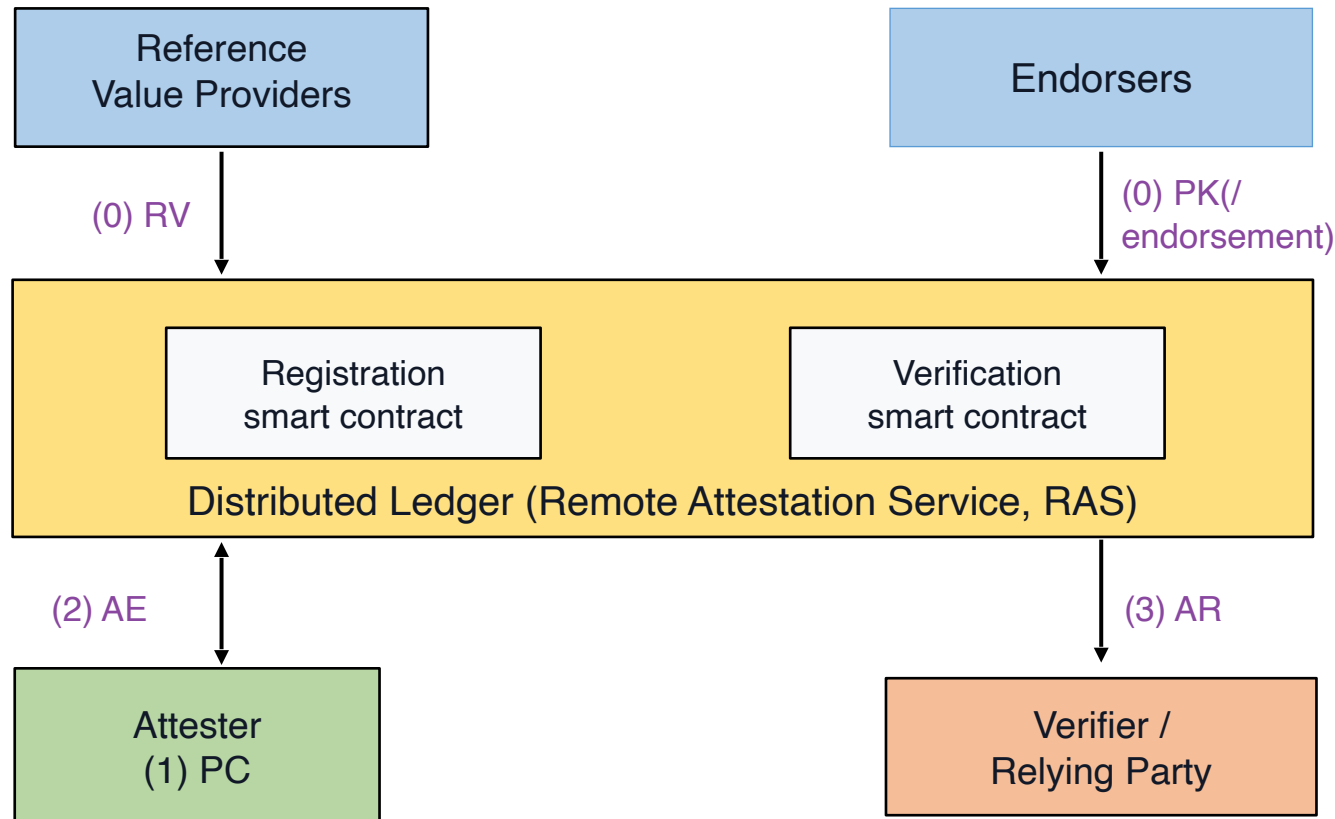


Attestation evidence exchange and appraisal follow existing RATS flows, while the DL is used to publish and retrieve verifier inputs and verifier outputs.

- (0) Publish inputs – RV providers and Endorsers publish RVs, PKs, and endorsements to the DL/RAS.
- (1) Issue challenge – Verifier or authorised entity issues a freshness challenge.
- (2) Send evidence – Attester generates AE with freshness proof and sends it to the Verifier (off-chain).
- (3) Fetch inputs – Verifier retrieves RVs and Endorser artefacts from the DL/RAS.
- (4) Appraise – Verifier evaluates AE and produces an Attestation Result (AR).
- (5/6) Publish & reuse – AR (or its digest/pointer) is published to the DL/RAS for RP reuse under policy

## Pattern 2: DL Publication with Optional On-Chain Verification

Pattern B



In this pattern, the DL is used for publication as in the previous pattern, and hosts verification logic (e.g., smart contracts) or record verifiers' appraisal outcomes.

(0) Publish inputs – RVs and Endorser artefacts (PKs, endorsements) are registered on the DL/RAS via the Registration smart contract.

(1) Publish evidence – Attester submits AE (with freshness proof and optional public challenge) to the DL/RAS.

(2) Verify on-chain – Verification smart contract validates AE using RVs/endorsements.

(3) Retrieve result – Verifier or RP queries the DL/RAS to obtain AR for decision or reuse.

# Implementation: TLS + Hyper-ledger Fabric Remote Attestation

Demo

The entire remote attestation process consists of 4 steps: Attester builds evidence —> RP forwards —> Verifier evaluates and writes to chain —> RP queries aggregated result for decision.

tpm-tls13-fabric-ra-demo  
Landscape flow console

ATTESTER · RP · VERIFIER · FABRIC

### TLS-bound attestation, one-screen view.

Built for demos: the full verification path stays visible in one landscape frame.

BINDING POINT  
TLS 1.3 transcript + CertificateVerify

EVIDENCE MODEL  
EAT, Signed AR, aggregated verdict

Artifacts + run locally

TLS CA

AK chain

ClientHelloRandom

Evidence / descriptor

Signed AR

Aggregated verdict

cd demo-web && python3 server.py

Open `http://localhost:8000` and trigger `One-click verify`. Use `LOG_DIR= ./demo-web/logs` for log parsing.

One-click verify

End & shutdown components

Flow board

Live logs

Refresh logs

End-to-end attestation flow  
12 animated checkpoints compressed into a landscape matrix.

Run state: idle

Auto-play enabled

RP LANE

ATTESTER LANE

VERIFIER LANE

FABRIC LANE

01 RP HANDSHAKE  
ClientHello  
RP opens TLS 1.3 and advertises the attestation-aware path.

02 ATTESTER TLS  
ServerHello + certs  
Attester replies with its TLS 1.3 handshake and certificate path.

03 ATTESTER OXFFA5  
Attach evidence  
Evidence is carried in the private-use CertificateEntry extension.

04 RP NONCE BIND  
Parse + bind nonce  
RP reads PeerAttestationData and binds it to ClientHelloRandom.

05 RP VERIFYREQUEST  
Send to Verifier  
RP forwards evidence, descriptor data, and the challenge context.

06 VERIFIER POLICY  
Validate evidence  
Verifier checks AK chain, PCR/eventlog, TCB, revocation, and profile.

07 FABRIC OPS  
Record nonce  
Replay protection is enforced with CheckAndRecordNonceV2.

08 FABRIC COMMIT  
Commit AR + verdict  
Evidence commitment, AR submission, and aggregation verdict are written.

09 VERIFIER RETURN  
Return Signed AR  
Verifier sends back the signed attestation result and evidence id.

10 RP CROSS-CHECK  
Verify on ledger  
RP re-checks signatures, commitment hash, AR set, and verdict policy.

11 RP DECISION  
Send decision  
Allowed / denied and evidence id are returned over TLS application data.

12 ATTESTER GATE  
Gate app traffic  
Post-handshake data is released only after the attestation decision.

Fabric transaction path

CheckAndRecordNonceV2(attester\_id, nonce\_hash, verifier\_id, ttl)

RecordEvidenceCommitmentV3(rp\_id, attester\_id, nonce\_hash, evidence\_digest, uri, format)

SubmitARV3(evidence\_id, verifier\_id, signed\_ar, policy\_digest, trusted)

TryAggregateWithPolicy(evidence\_id) -> AggregatedVerdictRecord

Channel split

channel-trust

trust anchors, participants, policies

channel-private-rv

reference values and rollout profiles

channel-ops

nonce records, commitments, AR set, aggregated verdict

LIVE RUN

LAST DECISION

Decision: allowed=true

evidence\_id:  
76540c70c511baba93a97cea8828a0bae1d23e9506a4cf78401ad3ffb3e3a013

time: 2026/02/25 11:54:04

SOURCE  
./demo-web/logs

MODE  
On-chain aggregation

Live stream preview

Start stream

Stop

Canvas capture feeds a compact video monitor.

Streaming (waiting for One-click verify)

VIDEO PREVIEW

Attestation Live Stream  
Time: 03:12  
01: ClientHello  
02: ServerHello + certs  
03: Attach evidence  
04: Parse + bind nonce  
05: Send to Verifier  
06: Validate evidence  
07: Record nonce  
08: Commit AR + verdict  
09: Return Signed AR  
10: Verify on ledger

Live execution log

Local replay and backend SSE both land here.

[system] waiting for run...

[system] API connected

logs/rp.log

logs/attester.log

logs/verifier.log

IETF125 - Hackathon - Talk

# Continue the Conversation

Next

## Feedback is more than welcome!

Whether the draft should stay DL-focused or be framed more generally as a shared tamper-evident publication channel; how to align DRA with existing RATS information models; and what trust, access-control, provenance, and freshness assumptions should be explicit.

## Where to read more

[draft-wang-rats-distributed-remote-attestation](#)

The current draft already covers two DRA patterns and a discussion of freshness, access control, and provenance.

Thank you