Automated Certificate Management Environment
Internet-Draft
Intended status: Standards Track
Expires: 3 September 2026

C. P. Liu
Huawei
M. Ounsworth
Cryptic Forest Software, Ltd
M. Richardson
Sandelman Software Works Inc

2 March 2026

Automated Certificate Management Environment (ACME) Remote Attestation Identifier and Challenge Type
draft-ietf-acme-rats-01

Abstract

This document describes an approach where an ACME Server can challenge an ACME Client to provide Evidence, Endorsements, or Attestation Result according to the Remote ATtestation procedureS (RATS) framework in any format supported by the Conceptual Message Wrapper (CMW).

The ACME Server can optionally challenge the Client for specific claims that it wishes attestation for.

About This Document

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

The latest revision of this draft can be found at https://liuchunchi.github.io/draft-liu-acme-rats/draft-liu-acme-rats.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-acme-rats/.

Discussion of this document takes place on the Automated Certificate Management Environment Working Group mailing list (mailto:acme@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/acme/. Subscribe at https://www.ietf.org/mailman/listinfo/acme/.

Source for this draft and an issue tracker can be found at https://github.com/liuchunchi/draft-liu-acme-rats.

Status of This Memo

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

This Internet-Draft will expire on 3 September 2026.

Copyright Notice

Table of Contents

1. Introduction

ACME [RFC8555] is a standard protocol for issuing and renewing certificates automatically. When an ACME client needs a certificate, it connects to the ACME server, providing a proof of control of a desired identity. Upon success, it then receives a certificate with that identity in it.

These identities become part of the certificate, usually a Fully Qualified Domain Name (FQDN) that goes into the Subject Alt Name (SAN) for a certificate. Prior to ACME, the authorization process of obtaining a certificate from an operator of a (public) certification authority was non-standard and ad-hoc. It ranged from sending faxes on company letterhead to answering an email sent to a well-known email address like hostmaster@example.com, evolving into a process where some randomized nonce could be placed in a particular place on the target web server. The point of this process is to prove that the given DNS FQDN was controlled by the same client system initiating the certificate request.

ACME standardized the process of proving domain control, allowing for full automation of certificate issuance. It has been a massive success: increasing HTTPS usage from 27% in 2013 to over 80% in 2019 [letsencrypt].

While the ACME process supports many kinds of identifiers such as email addresses and DTN node IDs, and other client-type identifiers, ACME for client certificates has not yet become as popular, in part because these types of certificates are usually located on much more general purpose systems such as laptops and computers used by people in which there is no standardized automated way to perform proof of control.

In addition to proving control of an identifier, a major concern that enterprises have with the use of client certificates has been the trustworthiness of the client system itself. Such systems have many more configurations, and are often harder to assess the security posture of as a result. While well managed mutual TLS (client and

server authentication via PKIX certificate) has significant advantages over the more common login via username/password, if the private key associated with a client certificates is disclosed or lost, then the impact can be more significant.

One use case envisioned here is that of client devices within an enterprise. A Network Operations Center (NOC) (which may be internal or an external contracted entity) will issue (client) certificates to devices that can prove via remote attestation that they are running an up-to-date operating system as well as the enterprise-required endpoint security software.

Another use case envisioned is issuance of server certificates, for example TLS certificates or code-signing certificates, where the Certification Authority (CA) requires, in addition to proof of control of the identifiers, some proof about the security posture of the operating environment, such as how the private key is stored, or the running environment of the server application.

This is a place where Remote Attestation Procudures (RATS) [RFC9334] can offer additional assurance. Remote Attestation provides a framework as well as an emerging ecosystem of standardized data formats and tools for a device to provide a remote peer with proof of various system properties in the form of measured evidence and third-party endorsements. ACME servers can leverage the Remote Attestation ecosystem either by directly consuming evidence and endorsements, or by having a third-party verifier assess the evidence and endorsements and produce an attestation result which is then provided to the ACME server.

In this document, we propose an extension to the ACME protocol where ACME Server MAY challenge the ACME Client to produce an Attestation Evidence or Attestation Result in any format that is supported by the RATS Conceptual Message Wrapper [I-D.ietf-rats-msg-wrap], for instance, an EAT (entity attestation token). The ACME Server then verifies the attestation result against an appraisal policy as required by the requested certificate profile. Essentially, this provides a mechanism for the ACME server to challenge for and transport Remote Attestation data, but it leaves out-of-scope how the ACME server verifies the provided data according to its certificate profiles. Remote Attestation can in some cases provide proof of control of an identifier, such as a hardware serial number or a cryptographic public key, but in general Remote Attestation will be at a lower level than the identifier going into the certificate such as a FQDN or an email address. Therefore the design of this extension does not replace the identity challenges in the ACME protocol, but acts supplementally to it.

ACME can presently offer certificates with multiple identities. Typically, in a server certificate situation, each identity represents a unique FQDN that would be placed into the certificate as distinct Subject Alt Names (SAN). For instance each of the names: example.com, www.example.com, www.example.net and marketing.example.com might be placed in a single certificate for a server that provides web content under those four names.

This document defines a new identity type, trustworthy that the ACME client can ask for. A new attestation-result-01 challenge is defined as a new method by which the ACME Server can challenge the Client to provide Remote Attestation according to the RATS Passport model. The attestation-evidence-02 challenge is also defined, challenging the client to provide direct evidence according to the RATS Background Check model. In this way, the Certification Authority (CA) or Registration Authority (RA) issues certificates only to devices that can provide an appropriate attestation of the required properties / claims, indicating that the device from which the ACME request originates has passed the required security checks.

The term "remote attestation" covers a broad range of standardized and proprietary technologies and data formats and this specification tries to be as technology-agnostic as possible. However, it assumes that evidence could contain sensitive data and therefore needs an optional payload encryption mechanism to keep this information protected as at passes through the various layers of the ACME server and RA to a RATS Verifier, whereas attestation results are assumed to be non-sensitive and therefore do not require this extra protection.

Attested ACME requests can form an essential building block towards the continuous monitoring/validation requirement of Zero-Trust principle when coupled with other operational measures, such as issuing only short-lived certificates.

For ease of denotion, we omit the "ACME" adjective from now on, where Server means ACME Server and Client means ACME Client.

1.1. Design overview

Conceptually, the server is challenging the client to produce remote attestation as part of the certificate enrollment flow. However, this mechanism cannot be defined simply as an ACME challenge for a number of reasons.

First, the ACME protocol [RFC8555] is designed such that a client requests a given identifier to appear in the certificate, and the server provides a list of proof-of-control challenges such as {DNS-01, HTTP-01} to which the client only needs to respond to one. The

design goal here is that the remote attestation challenge supplements the identifier challenges rather than replaces them. Instead we want the behavior that the client must respond to one of {DNS-01, HTTP-01} and also provide remote attestation of the given list of attestable properties.

Second, an ACME client can request multiple identifiers in a single certificate request and so ACME challenges are per identifier. By contrast, remote attestation typically applies to the entire client device or private key and is not coupled to the requested identifiers.

Consider an example where a Client requests a certificate for the DNS name "client01.finance.example" and the Server wants to respond that the client can fulfill either a DNS-01 or an HTTP-01 Challenge to prove ownership of that DNS same, and also the Client must provide two separate types of remote attestation, one measuring the boot stack and device integrity where the application runs, and a separate attestation proving that the subject private key is in an HSM. The desired behavior is {DNS-01 OR HTTP-01} AND measured-boot AND hsm. The only way for the Server to issue four Challenges in ACME that behave this way is to issue the DNS-01 and HTTP-01 under the same Identifier, and the measured-boot and hsm Challenges each under their own Identifier. To achieve this, Section 2.1 introduces a new ACME Identifier type "remote-attestation" which is not really an identifier, but instead conveys the general type of attestation required.

EDNOTE: there will be cases where the attestation does act as proof-of-control of an identifier, such as validating a Serial Number DN component. So do we want to allow attestation-result-01 and attestation-evidence-01 to be used within any identifier so that you can say DNS-01 or HTTP-01 or remote-attestation for cases where that makes sense?

1.2. Related work

TODO: need a compare & contrast with draft-ietf-acme-device-attest. @Ganesh, you’re an author on both, could you write this text?

[CSRATT] define a mechanism for carrying arbitrary remote attestation data within a certificate signing request (CSR) object (PKCS#10 or CRMF). Since ACME internally uses PKCS#10 CSRs, this provides an alternate mechanism for carrying remote attestation within ACME. This specification provides additional functionality that cannot be achieved via attested CSRs, namely giving the ACME server a way to challenge the client not only for attestation, for for attestation of specific properties. It also decouples the attestation from the CSR,

which future-proofs this mechanism in case CSR is removed as a mandatory part of the ACME protocol at some future time. That said, there is no reason that [CSRATT] could not be combined with the mechanism described in this document; for example a client could provide an attested CSR proving protection properties of the private key within an HSM, and also respond to an ACME remote attestation challenge proving the security posture of the application stack.

[RATSPA] defines a summary of a local assessment of posture for managed systems and across various layers. The claims and mechanisms defined in [RATSPA] are a good basis for the assessment that will need to be done in order to satisfy the trustworthiness challenge detailed in this document.

2. Overview

2.1. 'remote-attestation' identifier

A new identifier type to indicate client support or server request for remote attestation. This is a "dummy" identifier in that the value does not contain an actual identifier, but instead a property that is the be remotely attested. The value MAY be left empty, or contain a property hint as per Section 4.

type (required, string): The string "remote-attestation".

value (required, string): A string from the ACME Attest Claims Hint Registry defined in Section 4, which could be the empty string.

A Client MAY advertize support for multiple "remote-attestation" identifiers in the same ACME protocol flow.

A Server MAY issue challenges for multiple "remote-attestation" identifiers in the same ACME protocol flow. As the Server is authoritative for the requirements to be fulfilled for the given certificate request, the Server MAY choose not to issue a Challenge for every "remote-attestation" identifier type that the Client advertized support for, and conversely, the Server MAY issue a challenge for "remote-attestation" identifiers that the Client did not advertize support for.

EDNOTE: Is there any reason for the client to include "remote-attestation" identifiers in the newOrder at all?

2.2. remote-attest-01 Challenge

A remote-attest-01 challenge type asks the Client to provide Evidence appropriate for making a trustworthiness decision. The Client SHOULD use the provided freshness_nonce as an attestation freshness nonce, if the Client's underlying attestation technology supports freshness nonces.

The Server MAY include a attestClaimsHint containing a list of claims or specific properties that it would like to see attested.

The Client MUST complete the challenge by returning a CMW [I-D.ietf-rats-msg-wrap] which MAY contain remote attestation data in any defined CMW format, including: EAT [RFC9711], WebAuthn (cite), TPM attest_certify (?cite), PKIXKeyAttesation [RATSKA]. It may contain other RATS conceptual message types such as evidence, endorsement, or attestation result as appropriate for the mode. Since this specification allows wide flexibility on the contents of the remote attestation data, this document does not remove the need for vendors to perform interoperability testing with CAs to ensure compatibility.

This section describes the challenge/response extensions and procedures to use them.

2.2.1. remote-attest-01 Challenge Object

The remote-attest-01 Challenge Object is:

The basic fields type (which MUST be "remote-attest-01"), url, status, validated and error are preserved from Section 8 of [RFC8555] un-modified. The following new fields are added:

freshness_nonce (required, string): A randomly created nonce provided by the server which MUST be included in the Attestation Results to provide freshness.

EDNOTE TODO: we should decide whether this nonce MAY / SHOULD / SHOULD NOT / MUST NOT be the same as the ACME nonce or the ACME Challenge URL.

attestClaimsHint (optional, list of string) If the Server requires attestation of specific claims or properties in order to issue the requested certificate profile, then it MAY list one or more types of claims from the newly-defined ACME Attest Claims Hints registry defined in Section 4.

verifierEncryptionCredential (optional, JWK) A URL where the Client

can fetch the encryption public key that it can use for encrypting the remote-attestation challenge response.

The attestClaimsHint SHOULD contain values from the "JSON Web Token Claims" registry created by [RFC7519], in particular claims related to remote attestation as registered in [RFC9711] and related documents, but MAY contain non-registered values. The Client SHOULD attempt to collect evidence, endorsements, or attestation results from its local environment that satisfies these claims, either directly if the local environment supports EAT [RFC7519], or mapped to the closest equivalent claims in the supported format. The Client MAY ignore any claims that it does not recognize or that it is unable to collect remote attestation for. In other words, the Client SHOULD return what it has rather than failing, and allow the Server to decide if it is acceptable for the requested certificate profile.

The verifierEncryptionCredential is an encryption key in JSON Web Key (JWK) {!RFC7517} format that the Client can use to encrypt the attestation data that it will return. It is intended for cases where the evidence contains sensitive data the Client wishes to protect it against accidental logging, for example by HTTP proxies, as it passes through the Server’s application stack. This could for example include data such as identifiers that could be linkable to a person and therefore qualify as Personally Identifiable Information, or any other detailed type of system measurement that the Client deems sensitive. The credential SHOULD be in the JSON Web Key (JWK) [RFC7517] format, but MAY be in other reasonable formats such as a DER or PEM encoded X.509 certificate.

EDNOTE: in the name of simplicity, make this "MUST JWK"?

EDNOTE: We need to think carefully about whether this step needs to behave differently depending on whether the client will provide Evidence vs Attestation Result; particularly the nonces work a bit differently in the two cases. ... is it enough for the Server to always provide a nonce and just ignore it if the thing it gets back is an AR? Or maybe go even more hands-off and say "Here’s a nonce field, whether and how you use it is up to the RA and its attestation Verifier" ?

2.2.2. remote-attest-01 Response

The HTTP POST body to the challenge URL MUST be a raw JSON or CBOR CMW [I-D.ietf-rats-msg-wrap], Section 5.2, base64 encoded as necessary.

If the Server provided a verifierEncryptionCredential and the Client wishes to make use of it, then the entire CWM payload MUST be placed inside a JSON Web Encryption (JWE) envelope [RFC7516].

3. Example Protocol Flow

3.1. Step 1: newOrder Request Object

During the certificate order creation step, the Client sends a /newOrder JWS request (Section 7.4 of [RFC8555]) whose payload contains an array of identifiers. To indicate support for remote attestation, the client adds one or more remote-attestation identifiers to the array of identifiers. This is entirely optional and the server MAY choose to challenge for attestation or not according to its configuration for the requested certificate profile irrespective of whether the client indicated that it is attestation-capable.

The "remote-attestation" identifier MAY appear as the only identifier type in the array, but only if the supported remote attestation type is capable of proving control of the identifier(s) that will go into the certificate’s CN or SANs. For example, a measured-boot style remote attestation MAY be capable of proving ownership of a hardware serial number, or a WebAuthn / FIDO / Passkey style remote attestation MAY be capable of proving control of a FIDO token belonging to a given username or email address. However, more typically the "remote-attestation" identifier serves to provide supplemental trustworthiness next to a direct proof-of-control identity challenge such as DNS-01 or HTTP-01.

In this example, a client is requesting a certificate for a dns identity client01.finance.example and offers that it can provide remote attestation of the measured-boot stack of the application server via the remote-attestation identifier "measured-boot", as well as provide remote attestation from the underlying cryptographic hardware holding the private key via the remote-attestation identifier "hsm". These values refer to the ACME Attest Properties Hint Registry defined in Section 4.

An example extended newOrder JWS request:

```
POST /acme/new-order HTTP/1.1
Content-Type: application/json
{
  "protected": base64url({
    "alg": "ES256",
  }),
  "payload": base64url({
    "identifiers": [
      { "type": "dns", "value": "client01.finance.example" },
      { "type": "remote-attestation", "value": "measured-boot" },
      { "type": "remote-attestation", "value": "hsm" },
    ],
  }),
  "signature": "H6ZXtGjTZyUnPeKn...wEA4TklBdh3e454g"
}
```

3.2. Step 2: Order Object

As explained in [RFC8555], Section 7.1.3, the server returns an Order Object.

An example extended Order Object that includes confirmation that this order will include remote attestation:

```
POST /acme/new-order HTTP/1.1
...

HTTP/1.1 200 OK
Content-Type: application/json
{
  "status": "pending",

  "identifiers": [
      { "type": "dns", "value": "client01.finance.example" },
      { "type": "remote-attestation", "value": "measured-boot" },
      { "type": "remote-attestation", "value": "hsm" },
  ],

  "authorizations": [
    "https://example.com/acme/authz/PAniVnsZcis",
    "https://example.com/acme/authz/C1uq5Dr+x8GSEJTSKW5B",
    "https://example.com/acme/authz/01jcCDfT0iJBmUlaueum",
  ],

  "finalize": "https://example.com/acme/order/T..fgo/finalize",
}
```

The server is not required to match the list of remote-attestation identifiers provided by the client. The server MAY challenge for a subset of the remote-attestation identifiers offered by the client, or it MAY challenge for remote-attestation identifiers that were not offered by the client. The server has full discretion for deciding what properties are required to be attested for the requested certificate profile.

3.3. Step 3: Authorization Object

The Client MUST complete the authorizations provided by the server, but it MAY do so in any order [RFC8555].

In this example, the Server has created an Authorization Object for the "remote-attestation" and "dns" identifiers. The client accesses each authorization object from the URLs given in the Order Object. In this example, the PAniVnsZcis authorization relates to the dns identifier, and it is not changed from [RFC8555], Section 8. The C1uq5Dr+x8GSEJTSKW5B authorization corresponds to the remote-attestation "measured-boot" identifier.

Here is an example remote-attest-01 challenge:

```
GET https://example.com/acme/authz/C1uq5Dr+x8GSEJTSKW5B HTTP/1.1
..

HTTP/1.1 200 OK
Content-Type: application/json
{
  "status": "pending",
  "expires": "2025-09-30T14:09:07.99Z",

  "identifier": {
    "type": "remote-attestation",
    "value": "measured-boot"
  },

  "challenges": [
    {
      "type": "remote-attest-01",
      "url": "https://example.com/acme/chall/prV_8235AD9d",
      "status": "pending",
      "freshness_nonce": "yoW1RL2zPBzYEHBQ06Jy",
      "attestClaimsHint": ["hwmodel", "swversion", "submods", "manifests",],
      "verifierEncryptionCredential": "https://example.com/acme/ra-encr-keys/_fyg_W85y
Tul8zDPygcIgKmj-xA",
    }
  ],
}
```

EDNOTE: TODO: check 8555 if "token" is mandatory, and if so, use that to carry the attestation freshness nonce.

In this example, the Server is indicating that it wants a remote attestation result including the following claims:

```
"attestClaimsHint": ["hwmodel", "swversion", "submods", "manifests",],
```

meaning that it is interesting in the type of device and what software is running on it. This field is called a "hint" because the ACME Client might not be capable of obtaining remote attestation evidence, endorsements, or attestation results that directly map to these claims. Servers SHOULD NOT we written to expect exactly these claims back. This mechanism does not remove the need for vendors to perform interop testing against CAs.

EDNOTE: is the attestClaimsHint actually adding anything useful on top of the identifier values of "measured-boot", "hsm", "passkey", etc?

3.4. Step 4: Respond to Remote Attestation Challenge

The client now queries its local environment to obtain evidence, endorsements and/or attestation results to satisfy the Remote Attestation challenge.

If the underlying remote attestation technology supports a freshness nonce, then the Client passes the (example) token yoW1RL2zPBzYEHBQ06Jy into the underlying attestation service as the nonce.

The payload of the Remote Attestation Challenge response MUST be a CMW [I-D.ietf-rats-msg-wrap], but the underlying payload type within the CMW is left unconstrained and therefore the details of collecting the remote attestation data for this this step are not in scope for this document. As an example, it might use EAT [RFC9711], TPM-CHARRA [RFC9684], or X, or Y (XXX: insert more options)

Assume the following binary blob is Remote Attestation evidence collected by the Client:

```
yePAuQj5xXAnz87/7ItOkDTk5Y4syoW1RL2zPBzYEHBQ06JyUvZDYPYjeTqwlPszb9Grbxw0UAEFx5DxObV1
```

This result is sent as a POST to https://example.com/acme/chall/prV_8235AD9d

```
POST https://example.com/acme/chall/prV_8235AD9d HTTP/1.1
..
Content-Type: application/cmw+cbor

yePAuQj5xXAnz87/7ItOkDTk5Y4syoW1RL2zPBzYEHBQ06JyUvZDYPYjeTqwlPszb9Grbxw0UAEFx5DxObV1

(EDIT: change to cwm+jws example that wraps the example binary blob)
```

Alternatively, the Client MAY use the JWE format [RFC7516] to encrypt the remote attestation data for the Server's verifierEncryptionCredential and instead send a POST like this:

```
POST https://example.com/acme/chall/prV_8235AD9d HTTP/1.1
..
Content-Type: application/jwt

eyJhbGciOiJSU0EtT0FFUCIsImVuYyI6IkEyNTZHQ00ifQ.RYSEVOaD[snip]wwEbtY96_8P3a_A.5QLBf5hAX
CkP7CdS.XBiai[snip]RWz1D3f.kCIIMWMoObyekuQ33u0oYQ
```

('[snip]' indicates that the JWE has been truncated for publication.)

The Server decodes the provided CMW [I-D.ietf-rats-msg-wrap].

The Server MUST perform a full verification of the remote attestation data, including verification of the signature against a pre-configured trust anchor, and appraisal according to a suitable appraisal policy. The CA's certificate policy is the final authority for what trust anchors and appraisal policies will apply and the details are not in scope for this document.

At this point, if the client were to re-retrieve the authorization object from step 3, it would observe (if everything was accepted and successfully verified) that the status for this challenge would now be marked as valid.

3.5. Step 5: Perform other challenges

The client SHOULD now perform any other Remote Attestation or non-Remote Attestation challenges that were listed in the Order Object from step 2. ACME provides no ordering constraint on the challenges, so the Client MAY process them in any order, including concurrently.

3.6. Step 6: Finalize Order, retrieve certificate

At this point, the process continues as described in Section 7.4 of [RFC8555]. This means that the finalize action is used, which includes a CSR. If all is well, it will result in a certificate being issued.

4. ACME Attest Properties Hint Registry

In order for the client to communicate in the newOrder request what types of attestation it is capable of producing, and for the server to indicate in the newOrder response what properties it requires attestation of, this specification creates a new IANA registry called "ACME Attest Properties Hint Registry. The hint is used as the value of the "remote-attestation" identifier type, as described in {#new-order-req} and {#new-order-resp}. In order to preserve vendor flexibility, the initial values in the ACME Attest Properties Hint Registry are intended to be generic in nature, and decoupled from the RATS conceptual message type (evidence, endorsement, or attestation result) or attestation data format (EAT (cite), WebAuthn (cite), TPM attest_certify (cite), PKIXKeyAttestation (cite), or device-proprietary). This model expects CAs to publish documentation about what specific data formats they support, and for vendors to perform interoperability testing with CAs to ensure compatibility. Ultimately, the CA’s certificate policies will be the authority on what evidence or attestation results it will accept.

The ACME Attest Claims Hint Registry is intended to help clients to collect evidence or attestation results that are most likely to be acceptable to the server, but are not a guaranteed replacement for performing interoperability testing between a given attesting device and a given CA. Similarly, an ACME attestation hint may not map one-to-one with attestation functionality exposed by the underlying attesting device, so ACME clients might need to act as intermediaries mapping ACME hints to vendor-specific functionality on a per-hardware-vendor basis.

See Section 7.1 for the initial contents of this new registry.

5. Example use cases

5.1. Conflicting duties

(EDIT: This text might be stale)

1. Integration/compatibility difficulty: Integrating SOC and NOC requires plenty of customized, case-by-case developing work. Especially considering different system vendors, system versions, different data models and formats due to different client needs... Let alone possible updates.

2. Conflict of duties: NOC people do not want SOC people to interfere with their daily work, and so do SOC people. Also, NOC people may have limited security knowledge, and SOC people vice versa. Where to draw the line and what is the best tool to help them collaborate is a question.

5.2. Enterprise WiFi Access

In enterprise access cases, security administrators wish to check the security status of an accessing end device before it connects to the internal network. Endpoint Detection and Response (EDR) softwares can check the security/trustworthiness statuses of the device and produce an Attestation Result (AR) if the check passes. ACME-RA procedures can then be used to redeem a certificate using the AR.

With that being said, a more specific use case is as follows: an enterprise employee visits multiple campuses, and connects to each one’s WiFi. For example, an inspector visits many (tens of) power substations a day, connects to the local WiFi, download log data, proceed to the next and repeat the process.

Current access solution include: 1. The inspector remembers the password for each WiFi, and conduct the 802.1X EAP password-based (PAP/CHAP/MS-CHAPv2) authentication. or 2. an enterprise MDM receives the passwords and usernames over application layer connection from the MDM server, and enter them on user’s behalf. While Solution 1 obviously suffer from management burdens induced by massive number of password pairs, and password rotation requirements, the drawback of Solution 2 is more obsecure, which include:

a. Bring Your Own Device (BYOD) situation and MDM is not available.
b. Password could risk leakage due to APP compromise, or during Internet transmission. Anyone with leaked password can access, without binding of trusted/usual devices. c. The RADIUS Client/ Access Point/Switch is not aware of the identity of the accessing device, therefore cannot enforce more fine-grained access policies.

An ideal user story is: 1. When the inspector is at base (or whenever the Remote Attestation-based check is available), he get his device inspected and redeem a certificate using ACME-RA. 2. When at substation, the inspector authenticate to the WiFi using EAP-TLS, where all the substations have the company root CA installed. 2*. Alternatively, the Step 2 can use EAP-repeater mode, where the RADIUS Client redirects the request back to the RADIUS Server for more advanced checks.

5.3. BYOD devices

Another example is issuing S/MIME certificates to BYOD devices only if they can prove via attestation that they are registered to a corporate MDM and the user they are registered to matches the user for which a certificate has been requested.

In this case, the Server might challenge the client to prove that it is properly-registered to the enterprise to the same user as the subject of the requested S/MIME certificate, and that the device is running the corporate-approved security agents.

5.4. Private key in hardware

In some scenarios the CA might require that the private key corresponding to the certificate request is stored in cryptographic hardware and non-extractable. For example, the certificate profile for some types of administrative credentials may be required to be stored in a token or smartcard. Or the CA might be required to enforce that the private key is stored in a FIPS-certified HSM running in a configuration compliant with its FIPS certificate -- this is the case, for example, with CA/Browser Forum Code Signing certificates [CABF-CSBRs] which can be attested for example via [RATSKA].

It could also be possible that the requested certificate profile does not require the requested key to be hardware-backed, but that the CA will issue the certificate with extra assurance, for example an extra policy OID or a longer expiry period, if attestation of hardware can be provided.

6. Security Considerations

The attestation-result-01 challenge (the Passport Model) is the mandatory to implement. The encrypted-evidence-01 challenge (the background-check model) is optional.

In all cases the Server has to be able to verify Attestation Results from the Verifier. To do that it requires appropriate trust anchors.

In the Passport model, Evidence -- which may contain personally identifiable information (PII)) -- is never seen by the ACME Server. Additionally, there is no need for the Verifier to accept connections from ACME Server(s). The Attester/Verifier relationship used in the Passport Model leverages a pre-existing relationship. For instance if the Verifier is operated by the manufacturer of the Attester (or their designate), then this is the same relationship that would be used to obtain updated software/firmware. In this case, the trust anchors may also be publically available, but the Server does not need any further relationship with the Verifier.

In the background-check model, Evidence is sent from the Attester to the ACME Server. The ACME Server then relays this Evidence to a Verifier. The Evidence is encrypted so that the Server it never able to see any PII which might be included. The choice of Verifier is more complex in the background-check model. Not only does ACME Server have to have the correct trust anchors to verify the resulting Attestation Results, but the ACME Server will need some kind of business relationship with the Verifier in order for the Verifier to be willing to appraise Evidence.

The trustworthy identifier is not an actual identifier. It does not result in any specific contents to the certificate Subject or SubjectAltName.

7. IANA Considerations

7.1. ACME Attest Properties Hint Registry

IANA is requested to open a new registry, XXXXXXXX

Type: designated expert

The registry has the following columns:

* Property Hint: the string value to be placed within an ACME identifier of type "remote-attestation".

* Description: a description of the general property which the client is expected to attest.

The initial registry contents is shown in the table below.

Property Hint	Description
""	Empty string. Indicates client support for, or a server request for, attestation without being specific about what type. Typically this means the client will produce whatever remote attestation data it is capable of.
"hsm"	Attestation that the private key associated with this certificate request is stored in cryptographic hardware such as a TPM or PKCS#11 HSM. In the case of PKCS#11 HSMs, the attestation SHOULD contain the PKCs#11 properties of the private key storage, as well as an indication of whether the cryptographic module is operating in FIPS mode.
"measured_boot"	Attestation from the device's onboard measured-boot stack of the device running the application.
"os_patch_level"	Attestation to the version or patch level of the device's operating system.
"sw_manifest"	A manifest list of all software currently running on the device.
"fido2"	A request for FIDO2-based user authentication; maybe this is to validate a user identifier directly against the FIDO2 data, or maybe this serves as a second-factor to the CA's certificate pickup flow.

Table 1

In general, the target environment that the Server wants to be remotely attested is the environment where the application is running. The "hsm" property is an exception to this because this wants attestation from the cryptographic module instead.

In cases where the ACME client is running on a different host from the application, remote attestation always refers to the application host. In other words, a centralized ACME client MUST fulfill the

ACME-RA challenge by getting the application server that will ultimately use the certificate to query its local environment for remote attestation, and the ACME client MUST NOT present remote attestation from the host where it is running.

EDNOTE: I am somewhat surprised that a registry like this does not already exist associated with CMW.

8. References

8.1. Normative References

[RFC8555] Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", RFC 8555, DOI 10.17487/RFC8555, March 2019, <https://www.rfc-editor.org/rfc/rfc8555>.

[RFC9334] Birkholz, H., Thaler, D., Richardson, M., Smith, N., and W. Pan, "Remote ATtestation procedureS (RATS) Architecture", RFC 9334, DOI 10.17487/RFC9334, January 2023, <https://www.rfc-editor.org/rfc/rfc9334>.

[I-D.ietf-rats-msg-wrap] Birkholz, H., Smith, N., Fossati, T., Tschofenig, H., and D. Glaze, "RATS Conceptual Messages Wrapper (CMW)", Work in Progress, Internet-Draft, draft-ietf-rats-msg-wrap-23, 11 December 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-msg-wrap-23>.

[I-D.ietf-rats-ar4si] Voit, E., Birkholz, H., Hardjono, T., Fossati, T., and V. Scarlata, "Attestation Results for Secure Interactions", Work in Progress, Internet-Draft, draft-ietf-rats-ar4si-09, 15 August 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-ar4si-09>.

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <https://www.rfc-editor.org/rfc/rfc7519>.

[RFC9711] Lundblade, L., Mandyam, G., O'Donoghue, J., and C. Wallace, "The Entity Attestation Token (EAT)", RFC 9711, DOI 10.17487/RFC9711, April 2025, <https://www.rfc-editor.org/rfc/rfc9711>.

[RFC7517] Jones, M., "JSON Web Key (JWK)", RFC 7517, DOI 10.17487/RFC7517, May 2015, <https://www.rfc-editor.org/rfc/rfc7517>.

[RFC7516] Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)", RFC 7516, DOI 10.17487/RFC7516, May 2015, <https://www.rfc-editor.org/rfc/rfc7516>.

8.2. Informative References

[CSRATT] Ounsworth, M., Tschofenig, H., Birkholz, H., Wiseman, M., and N. Smith, "Use of Remote Attestation with Certification Signing Requests", Work in Progress, Internet-Draft, draft-ietf-lamps-csr-attestation-23, 1 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-lamps-csr-attestation-23>.

[RATSPA] Moriarty, K., Wiseman, M., Stein, A. J., and C. Nelogal, "Remote Posture Assessment for Systems, Containers, and Applications at Scale", Work in Progress, Internet-Draft, draft-ietf-rats-posture-assessment-03, 7 July 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-posture-assessment-03>.

[RATSKA] Ounsworth, M., Fiset, J., Tschofenig, H., Birkholz, H., Wiseman, M., and N. Smith, "Evidence Encoding for Hardware Security Modules", Work in Progress, Internet-Draft, draft-ietf-rats-pkix-key-attestation-03, 1 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-pkix-key-attestation-03>.

[I-D.draft-bweeks-acme-device-attest-01] Weeks, B., "Automated Certificate Management Environment (ACME) Device Attestation Extension", Work in Progress, Internet-Draft, draft-bweeks-acme-device-attest-01, 7 August 2022, <https://datatracker.ietf.org/doc/html/draft-bweeks-acme-device-attest-01>.

[letsencrypt] Electronic Frontier Foundation, "Celebrating Ten Years of Encrypting the Web with Let’s Encrypt", 20 August 2025, <https://www.eff.org/deeplinks/2023/08/celebrating-ten-years-encrypting-web-lets-encrypt>.

[CABF-CSBRs] CA/BROWSER FORUM, "Baseline Requirements for Code-Signing Certificates", n.d., <https://cabforum.org/working-groups/code-signing/documents/>.

[RFC9684] Birkholz, H., Eckel, M., Bhandari, S., Voit, E., Sulzen, B., Xia, L., Laffey, T., and G. C. Fedorkow, "A YANG Data Model for Challenge-Response-Based Remote Attestation (CHARRA) Procedures Using Trusted Platform Modules (TPMs)", RFC 9684, DOI 10.17487/RFC9684, December 2024, <https://www.rfc-editor.org/rfc/rfc9684>.

Acknowledgments

TODO acknowledge.

Authors' Addresses

Chunchi Peter Liu
Huawei
Email: liuchunchi@huawei.com

Mike Ounsworth
Cryptic Forest Software, Ltd
Email: mike@ounsworth.ca

Michael Richardson
Sandelman Software Works Inc
Email: mcr+ietf@sandelman.ca