Authority Tokens for ACME

IETF 101

ACME WG

Jon - London - Mar 2018

STIR and ACME

- What is STIR? Secure Telephone Identity (Revisited)
  - ART Area WG
  - Providing cryptographic authentication for telephone calls
  - Detecting impersonation is crucial to blocking illegal robocalling and other attacks on the telephone network
  - Based on RFC8226 certs
- We currently have two ACME WG documents to support STIR (RFC8226):
  - draft-ietf-acme-telephone
  - draft-ietf-acme-service-provider (based on current ATIS/SIP Forum IPNNI Task group challenge/response mechanism)

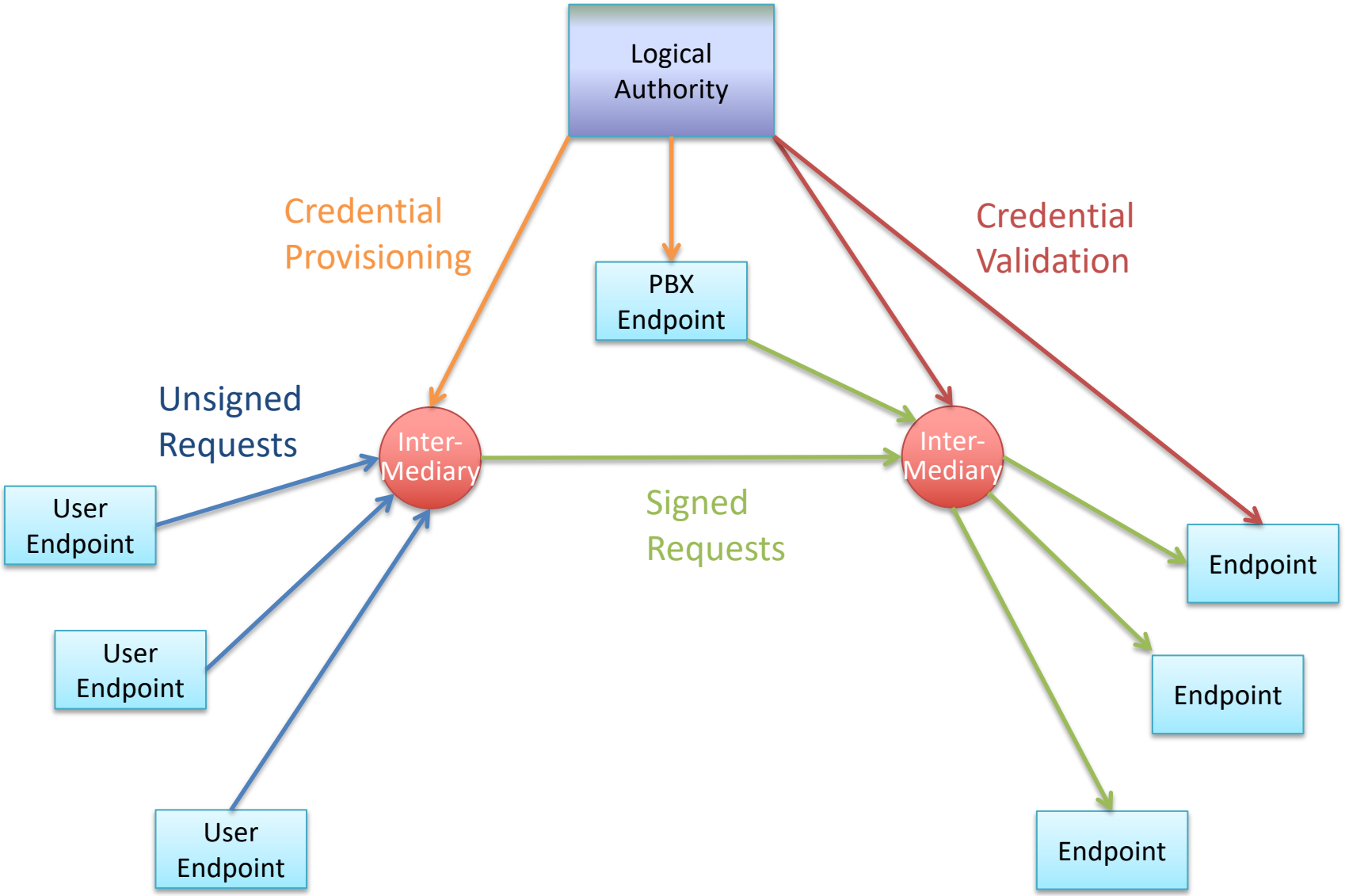
STIR and ACME

- During discussion of draft-ietf-acme-service-provider-02 at IETF-99, WG requested consideration of a generic token mechanism
- Two generic proposals discussed at IETF-100:
  - Abstraction of draft-ietf-acme-service-provider-02 to a very simple token challenge/response mechanism: draft-barnes-acme-token-challenge, with companion service provider code document: draft-barnes-acme-service-provider-code-00
  - Proposal applicable to a broader range of applications: draft-peterson-acme-authority-token-00
  - WG requested a single proposal

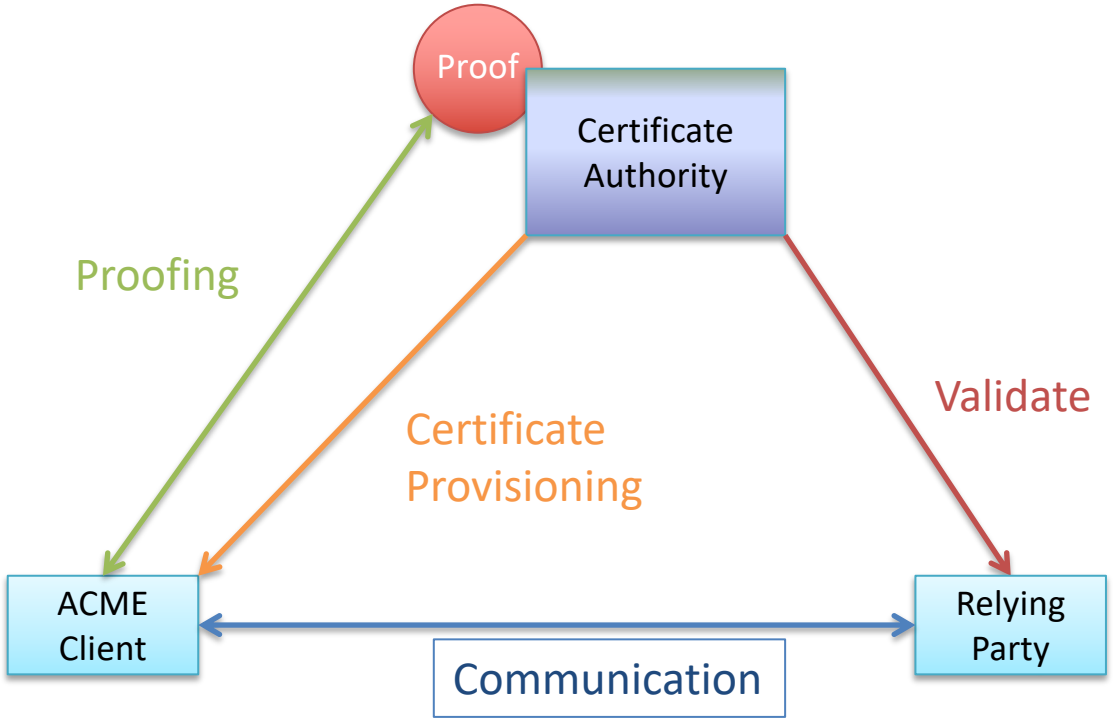
STIR and ACME

- A joint proposal developed in anticipation of IETF-101 comprised of two drafts:
  - Generic “Authority Token” (this presentation)
    - » draft-peterson-acme-authority-token-01
  - Use of Authority Token for TNAuthlist – both TNs and Service Provider Codes (next presentation)
    - » draft-wendt-acme-authority-token-tnauthlist

In-band STIR Logical Architecture



ACME (through a STIR lens)



Authority Token Challenge

- Identified a generic need for authorities to provide tokens to a CA to respond to challenges
  - Surely any number of namespaces have authorities who could generate tokens
    - Inspired by the STIR case, but this could work for domains even
  - Requires the ACME server has some trust relationship with the authority
- draft-peterson-acme-authority-token
  - Framework for tokens that allow authorities trusted by the CA to attest client ownership of names
    - CA can then issue certs via ACME for particular names
  - Need some sort of typing mechanism for tokens, and a means to contact authorities

Example challenge

```
"challenges": [
        {
          "type": "tkauth-01",
          "tkauth-type": “ATC",
          "token-authority": "https://authority.example.org/authz",
          "url": "https://boulder.example.com/authz/asdf/0"
          "token": "IlirfxKKXAsHtmzK29Pj8A" }
        ]
```

- The tkauth-type is governed by a registry
  - Specifies the syntax of the token
    - Today we only specify one initial registration, for JWT
  - It is the identifier type in the challenge that tells you what you are asking the authority to attest
- The token-authority supplies an optional URL
  - A hint for where clients can get a token
  - Not mandatory to follow, clients may already know where to get tokens elsewhere

Initial Token Registration

- Based on JWT
  - Used by the TNAuthlist document
- Example ACME response with a JWT
  - The JWT itself is the “ATC” payload in **bold**

```
{ "protected": base64url({
  "alg": "ES256",
  "kid": "https://boulder.example.com/acme/reg/asdf",
  "nonce": "Q_s3MWoqT05TrdkM2MTDcw",
  "url": "https://boulder.example.com/acme/authz/asdf/0" }),
  "payload": base64url({ "ATC": "evaGxfADs...62jcerQ" }),
  "signature": "5wUrDI3eAaV4wl2Rfj3aC0Pp--XB3t4YYuNgacv_D3U" }
```

Open Issue: Fingerprint v. Nonce

- Right now the Token Authority is given the nonce from the Reply-Nonce in the HTTP response
  - That is reflected in the JWT to bind the token to the ACME challenge
- This has some design implications
  - Works per challenge, rather than per ACME account
  - You need a new ATC token for each challenge
    - Could be a lot of work for short-lived certs
  - An alternative: use a fingerprint associated with the ACME account
    - Then a token could be reused for multiple challenges
- Any thoughts?

TNAuthList profile of ACME Authority Token

draft-wendt-acme-authority-token-tnauthlist-00

ACME Working Group
IETF101

Overview

- Profile specification to define the ACME usage RFC8226 certificates and specifically TNAuthList validation/authorization
- Profile of draft-peterson-acme-authority-token
- Specifically needed for cases of telephone service providers based on a national regulator delegated authority

Transactional Overview

- Communications Service provider (CSP) has an authority to represent a set of telephone numbers either explicitly via Telephone Numbers (TNs) or TN ranges or based on a recognized and unique authorized Service Provider Code (SPC) [RFC 8226]
- CSP wants a new certificate and makes a CSR request, gets a challenge with identifier “TNAuthList”
- CSP has a relationship with an authorized service that can provide a valid token representing their association with TNs or SPCs via a TNAuthList representation
- CSP responds to ACME challenge with this token
- Challenge is validated by CA based on token signature signed by known associate authority/certificate
- A RFC 8226 compliant certificate with TNAuthList is created

New Identifier

- type = “TNAuthList”
- value = JSON array of TNAuthList components with associated keys and values

```
POST /acme/new-order HTTP/1.1
Host: example.com
Content-Type: application/jose+json

{
  "protected": base64url({
    "alg": "ES256",
    "kid": "https://example.com/acme/acct/1",
    "nonce": "5XJ1L3lEkMG7tR6pA00clA",
    "url": "https://example.com/acme/new-order"
  }),
  "payload": base64url({
    "identifiers": [{"type:"TNAuthList","value":"["spc":"1234",
      "tn":"2155551212"]"}],
    "notBefore": "2016-01-01T00:00:00Z",
    "notAfter": "2016-01-08T00:00:00Z"
  }),
  "signature": "H6ZXtGjTZyUnPeKn...wEA4TklBdh3e454g"
}
```

Should value be a string?

- seems in acme-acme value is defined as string, should we do stringified JSON, or should ACME consider making value more flexible?

```
POST /acme/new-order HTTP/1.1
Host: example.com
Content-Type: application/jose+json

{
  "protected": base64url({
    "alg": "ES256",
    "kid": "https://example.com/acme/acct/1",
    "nonce": "5XJ1L3lEkMG7tR6pA00clA",
    "url": "https://example.com/acme/new-order"
  }),
  "payload": base64url({
    "identifiers": [{"type:"TNAuthList","value":"["spc":"1234",
      "tn":"2155551212"]"}],
    "notBefore": "2016-01-01T00:00:00Z",
    "notAfter": "2016-01-08T00:00:00Z"
  }),
  "signature": "H6ZXtGjTZyUnPeKn...wEA4TklBdh3e454g"
}
```

Challenge/challenge response per ATC

```
GET /acme/authz/1234 HTTP/1.1
Host: example.com
HTTP/1.1 200 OK
Content-Type: application/json
Link: <https://example.com/acme/some-directory>;rel="index"

{
  "status": "pending",
  "expires": "2018-03-03T14:09:00Z",

  "identifier": {
    "type:"TNAuthList",
    "value":"["spc":"1234","tn":"2155551212"]"
  },

  "challenges": [
    {
      "type": "tkauth-01",
      "tkauth-type": "ATC",
      "token-authority": "https://authority.example.org/authz",
      "url": "https://boulder.example.com/authz/asdf/0"
      "token": "IlirfxKKXAsHtmzK29Pj8A"
    }
  ]
}
```

```
POST /acme/authz/asdf/0 HTTP/1.1
Host: sti-ca.com
Content-Type: application/jose+json

{
 "protected": base64url({
 "alg": "ES256",
 "kid": "https://sti-ca.com/acme/reg/asdf",
 "nonce": "Q_s3MWoqT05TrdkM2MTDcw",
 "url": "https://sti-ca.com/acme/authz/asdf/0"
}),
 "payload": base64url({
 "ATC": "DGyRejmCefe7v4N...vb29HhjjLPSggwiE"
}),
 "signature": "9cbg5JO1Gf5YLjjz...SpkUfcdPai9uVYYQ"
}
```

ATC token/“atc” claim

- all claims are per ATC, except “atc”
- ATC claim contains key of “TNAuthList” and value of JSON array of TNAuthList components
- Similar question of value should be “string” or not, probably would like to keep it consistent with Identifier rules

```
{ "typ":"JWT",
  "alg":"ES256",
  "x5u":https://authority.example.org/cert
}

{
 “iss":"https://authority.example.org/authz",
 "exp":1300819380,
 "jti":"id6098364921",
 "atc":["TnAuthList","["spc":"1234","tn":"2155551212"]",
   "Q_s3MWoqT05TrdkM2MTDcw"]
}
```

Example 1

- TNAuthList Authority Token authorizing a TNAuthList containing a single SPC value

```
{
  "typ":"JWT",
  "alg":"ES256",
  "x5u":https://authority.example.org/cert
}

{
  "iss":"https://authority.example.org/authz",
  "exp":1300819380,
  "jti":"id6098364921",
  "atc":["TnAuthList","["spc":"1234"]","Q_s3MWoqT05TrdkM2MTDcw"]
}
```

Example 2

- TNAuthList Authority Token authorizing a TNAuthList identifier containing an SPC value plus a range of TNs

```
  {
    "typ":"JWT",
    "alg":"ES256",
    “x5u":https://authority.example.org/cert
  }

  {
    "iss":"https://authority.example.org/authz",
    "exp":1300819380,
    "jti":"id6098364921",
    "atc":["TnAuthList",
         ["spc":"1234","tn-
range":{"start":"12155551212","count":"50"}],
         "Q_s3MWoqT05TrdkM2MTDcw"]
  }
```

Example 3

- TNAuthList Authority Token authorizing a TNAuthList identifier containing a single TN

```
{
  "typ":"JWT",
  "alg":"ES256",
  "x5u":https://authority.example.org/cert
}

{
  "iss":"https://authority.example.org/authz",
  "exp":1300819380,
  "jti":"id6098364921",
  "atc":["TnAuthList",
       ["tn":"12155551212"],
       "Q_s3MWoqT05TrdkM2MTDcw"]
}
```

Next Steps

- Since last meeting we went back and aligned on a plan that incorporated a generic token mechanism for authority specific use cases and split off the STIR specific parts into a profile document
- This is fairly straight forward
- Industry is working via STIR/SHAKEN in North America for finalizing solution for call identity validation
- Would like to move forward fairly quickly

Next Steps

- Working Group adoption?

  1. Generic ACME “Token Authority” mechanism: draft-peterson-acme-authority-token-01

  2. TNAuthlist for TNs and Service Provider codes:

     - draft-wendt-acme-authority-token-tnauthlist

- Replaces both:

  – draft-ietf-acme-telephone

  – draft-ietf-acme-service-provider