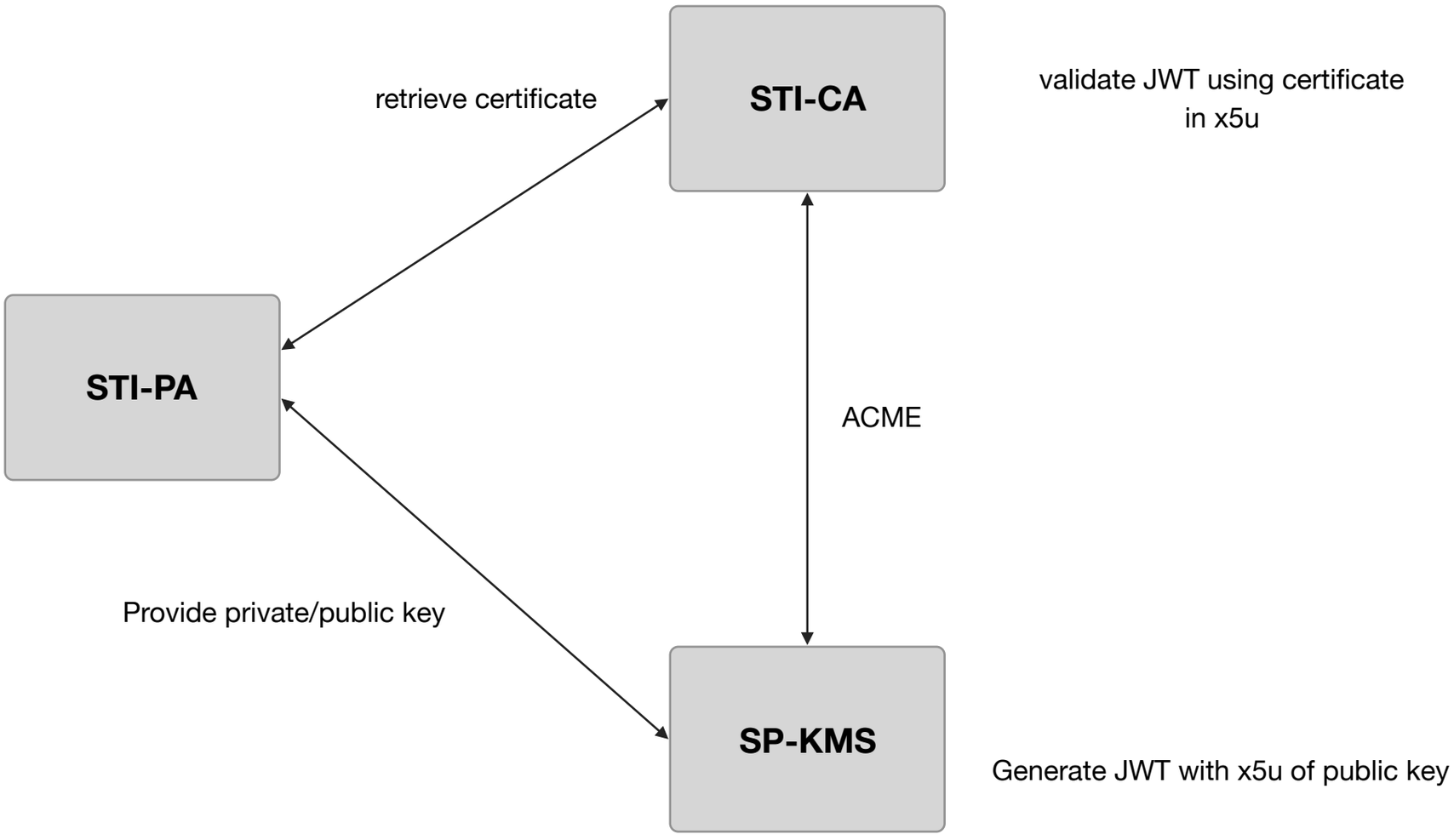
SHAKEN - Certificate Framework Administration using ACME/OAuth

IPNNI Task Force
Chris Wendt - Comcast

Auth framework



ACME - Registration object

- ACME Registration resource represents an account

  - **key** (required, dictionary) - public key encoded as JSON WebKey
    status (required, string) - status of the registration. “valid”, “deactivated”, “revoked” - deactivated represents user invoked deactivation vs. revoked is administrator invoked deactivation.
    contact (optional, array of string) - An array of URIs the ACME server can use to contact the clients about authorization issues (e.g. being revoked)
    terms-of-service-agreed (optional, boolean) - true indicates agreement
    applications (required, string) - URI from which a list of authorization submitted by this account can be fetched

Example Registration object:

```
{
  "contact": [
    “mailto:cert-admin@example.com",
    “tel:+12025551212”
    ],
  "terms-of-service-agreed": true,
  "applications": "https://example.com/acme/reg/
  1/apps"
}
```

Example response to applications URL:

```
   {
     "applications": [
       "https://example.com/acme/reg/1/apps/1",
       "https://example.com/acme/reg/1/apps/2",
       /* 47 more URLs not shown for example brevity */
       "https://example.com/acme/reg/1/apps/50"
     ]
   }
```

ACME - Application object

- ACME Application object represents a client's request for a certificate, and it's lifecycle through to issuance.

  - **status** (required, string) - status of the application. "pending", "valid", "invalid"
    expires (optional, string) - timestamp of when the server will no longer consider the application valid
    csr (required, string) - CSR encoding for the requested certificate
    notBefore (optional, string) - requested notBefore field in the certificate
    notAfter (optional, string) - requested notAfter field in the certificate
    requirements (required, array) - requirements client needs to fulfill before granting certificate
    certificate (optional, string) - URL for the certificate issued for this application

ACME - Application object

```
Example Application object:
  {
     "status": "pending",
     "expires": "2015-03-01T14:09:00Z",

     "csr": "jcRf4uXra7FGYW5ZMewvV...rhlnznwy8YbpMGqwidEXfE",
     "notBefore": "2016-01-01T00:00:00Z",
     "notAfter": "2016-01-08T00:00:00Z",

     "requirements": [
       {
         "type": "authorization",
         "status": "valid",
         "url": "https://example.com/acme/authz/1234"
       }
     ]

     "certificate": "https://example.com/acme/cert/1234"
  }
```

- ACME Application object represents a client's request for a certificate, and it's lifecycle through to issuance.

  - **type** (required, string) - "authorization" or "out-of-band"
    status (required, string) - "pending", "valid", "invalid"
    url (required for "authorization", string) - URL of the authorization resource

ACME - Authorization object

- ACME authorization object represents a server's authorization for an account to represent an identifier.

  - **identifier** (required, dictionary of string) - The identifier that the account is authorized to represent
    - **type** (required, string) - type of identifier (e.g. "dns" or "spid")
    - **value** (required, string) - identifier itself

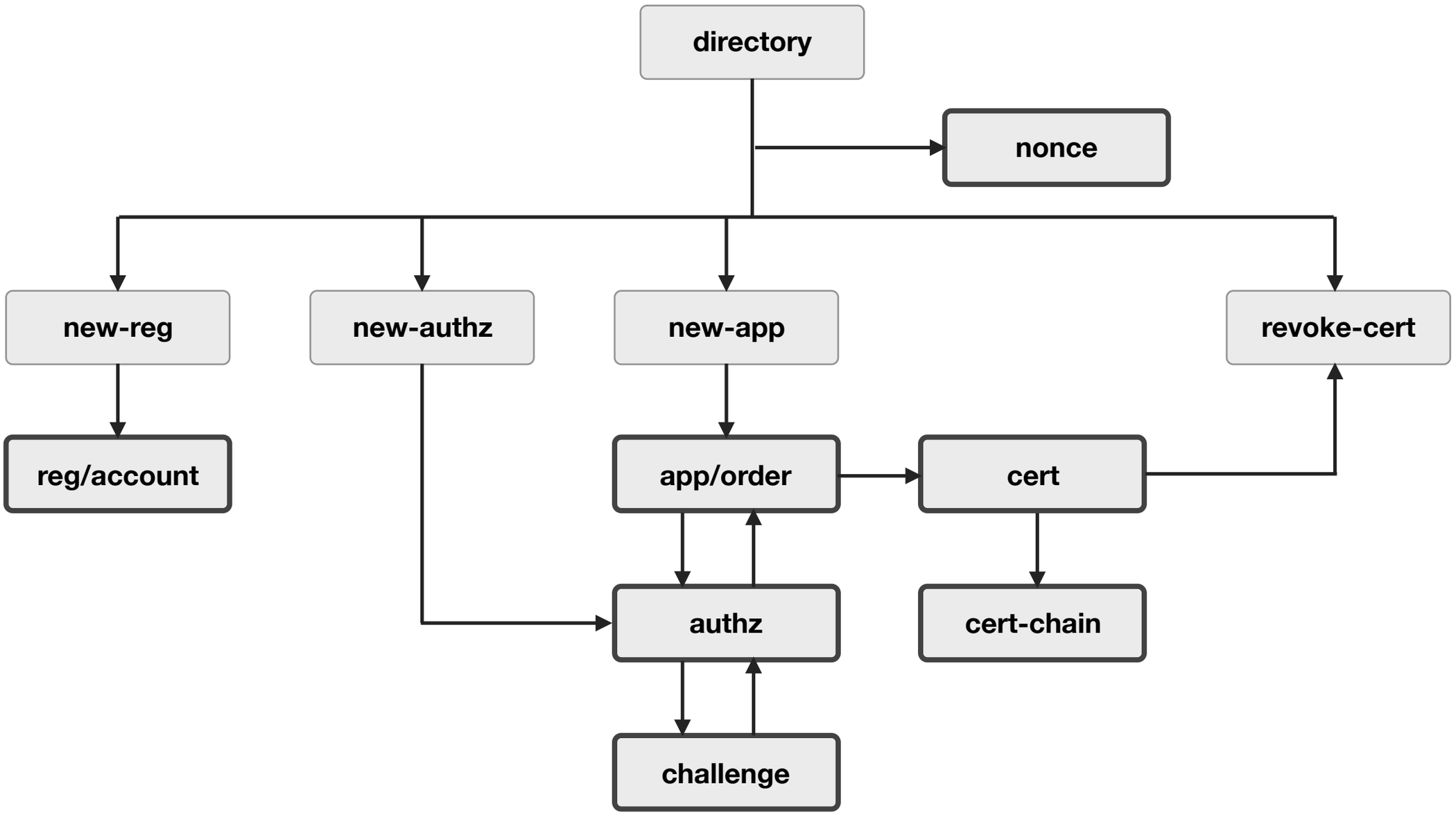
    status (required, string) - status of the authorization. "pending", "processing", "valid", "invalid", "revoked" - default value is "pending".
    expires (optional, string) - if present, contains URI for an application resource, if absent, then CA MUST consider authorization valid for all applications
    challenges (required, array) - an array of challenges used for authorization

```
{
  "status": "valid",
  "expires": "2015-03-01T14:09:00Z",

  "identifier": {
    "type": "dns",
    "value": "example.org"
  },

  "challenges": [
    {
      "type": "http-01",
      "status": "valid",
      "validated": "2014-12-01T12:05:00Z",
      "keyAuthorization": "SXQe-2XODaDxNR...vb29HhjjLPSggwiE"
    }
  ]
}
```

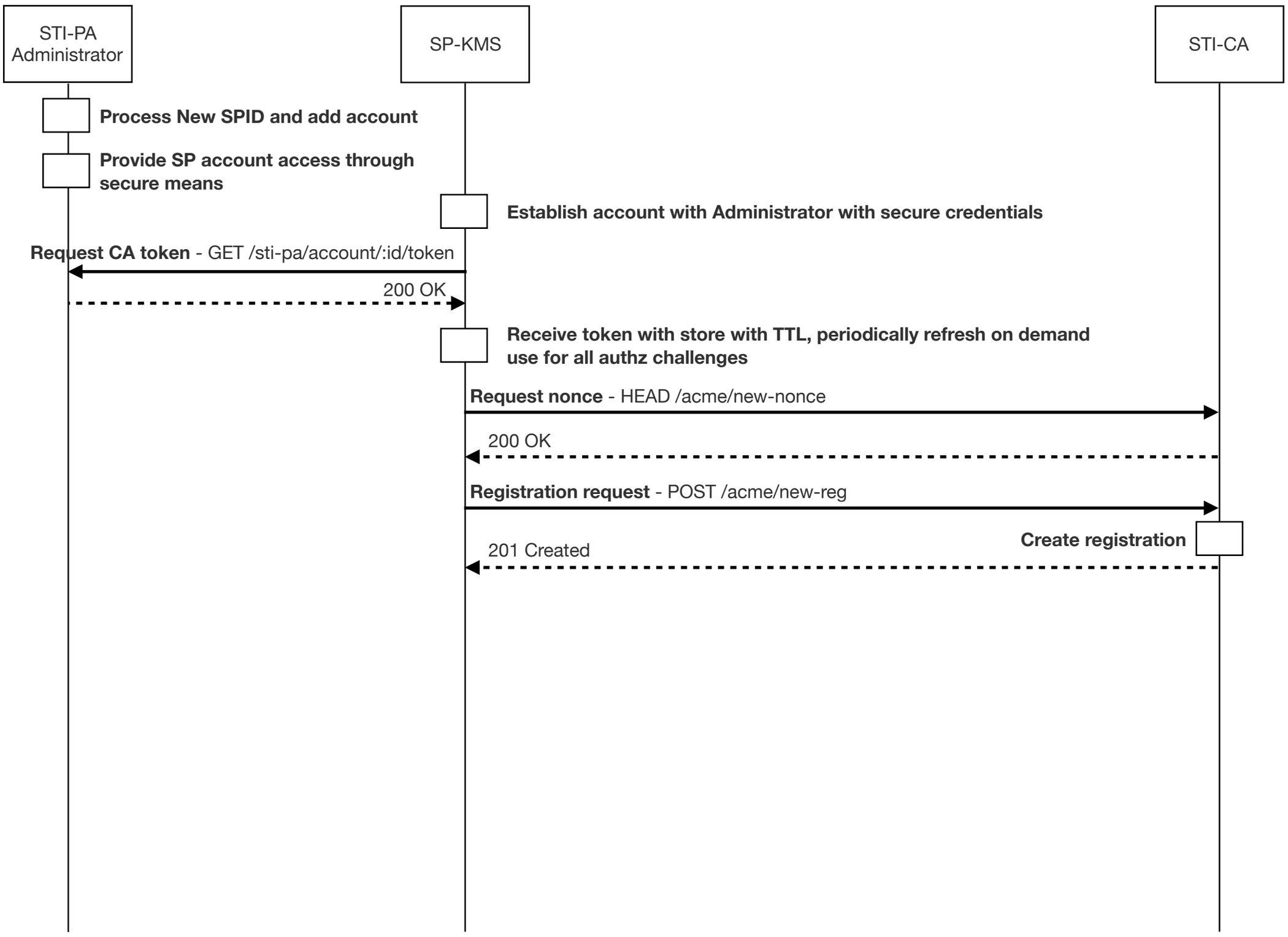
ACME object directory



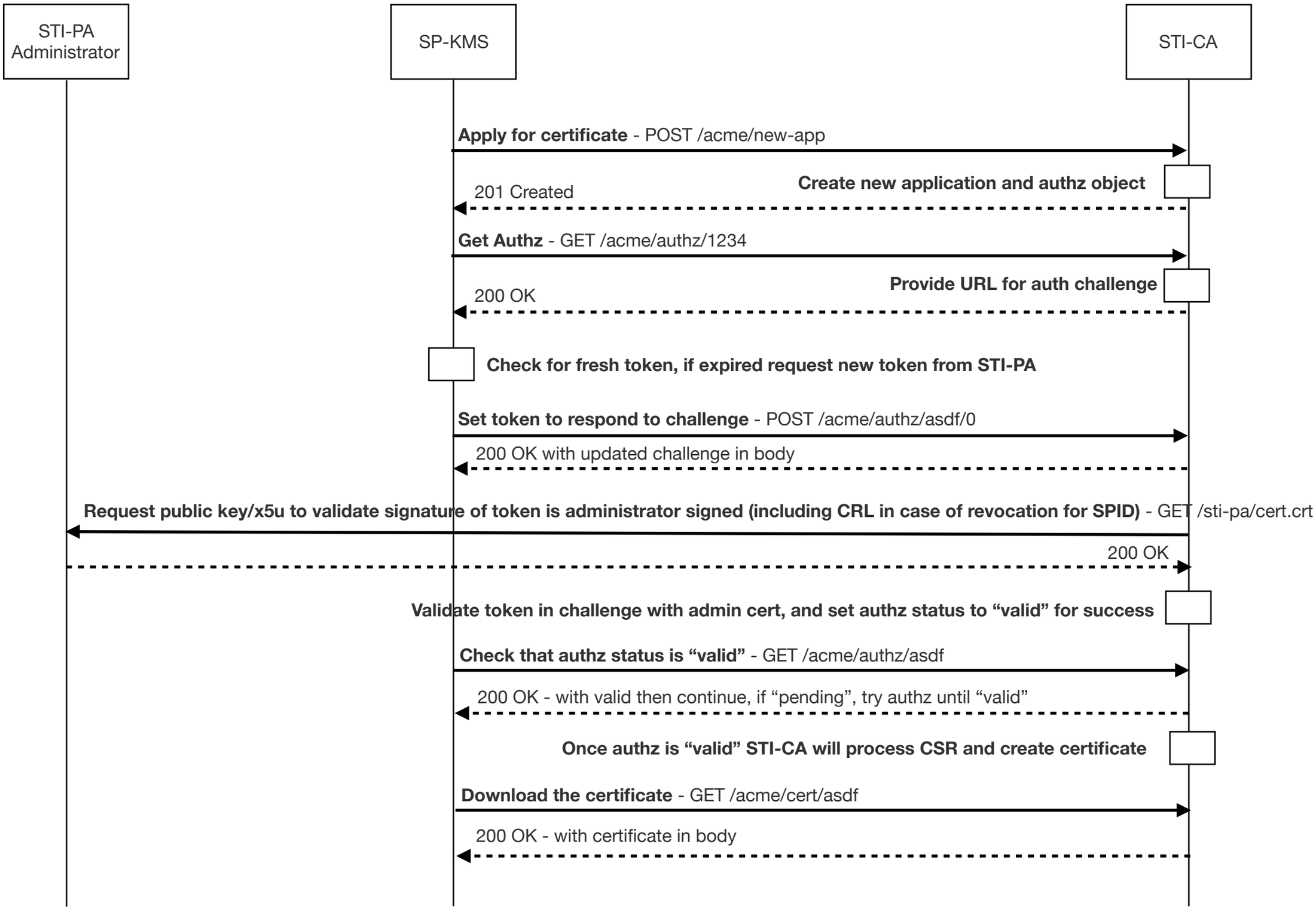
ACME - high level flow

- ACME high level request flow:
  - **Get a nonce** - HEAD new-nonce
  - **Register/Account** - POST new-reg
  - **Apply for a cert (app/order)** - POST new-app
  - **Fetch challenges** - GET authz
  - **Answer challenges** - POST challenge
  - **Poll for status** - GET authz
  - **Check for new cert** - GET cert

SHAKEN ACME/Auth call flow - account setup/registration



SHAKEN ACME/Auth call flow - certificate request



SHAKEN ACME flow

- Step 1 - Request nonce

```
HEAD /acme/new-nonce HTTP/1.1
 Host: sti-ca.com

 HTTP/1.1 200 OK
 Replay-Nonce: oFvnlFP1wIhRlYS2jTaXbA
 Cache-Control: no-store
```

- Step 2 - Registration Request - likely done per SP-KMS node

```
POST /acme/new-reg HTTP/1.1
Host: sti-ca.com
Content-Type: application/jose+json

{
  "protected": base64url({
    "alg": "ES256",
    "jwk": {...},
    "nonce": "6S8IqOGY7eL2lsGoTZYifg",
    "url": “https://sti-ca.com/acme/new-reg”
  })
  "payload": base64url({
    "terms-of-service-agreed": true,
    "contact": [
      “mailto:cert-admin-sp-kms01@sp.com”,
      "tel:+12155551212"
    ]
  }),
  "signature": "RZPOnYoPs1PhjszF...-nh6X1qtOFPB519I"
}
```

```
Example jwk public key - EC256
{“kty":"EC",
 "crv":"P-256",
 "x":"f83OJ3D2xF1Bg8vub9tLe1gHMzV76e8Tus9uPHvRVEU",
 "y":"x_FEzRu9m36HLN_tue659LNpXW6pCyStikYjKIWI5a0",
 "kid":"Public key used in JWS spec Appendix A.3 example"
}
```

SHAKEN flow

- Step 3 - If registration exists return 200 OK response, otherwise if registration is created return 201

```
HTTP/1.1 201 Created
Content-Type: application/json
Replay-Nonce: D8s4D2mLs8Vn-goWuPQeKA
Location: https://sti-ca.com/acme/reg/asdf
Link: <https://sti-ca.com/acme/some-directory>;rel=“directory"

{
  "key": { /* JWK from JWS header */ },
  "status": "valid",

  "contact": [
    “mailto:cert-admin-sp-kms01@sp.com”,
    "tel:+12155551212"
  ]
}
```

SHAKEN flow

- Step 3a - if key is compromised, a key-change can be initiated

```
POST /acme/key-change HTTP/1.1
Host: sti-ca.com
Content-Type: application/jose+json

{
  "protected": base64url({
    "alg": "ES256",
    "jwk": /* old key */,
    "nonce": "K60BWPrMQG9SDxBDS_xtSw",
    "url": “https://sti-ca.com/acme/key-change"
  }),
  "payload": base64url({
    "protected": base64url({
      "alg": "ES256",
      "jwk": /* new key */,
    }),
    "payload": base64url({
      "account": “https://sti-ca.com/acme/reg/asdf",
      "newKey": /* new key */
    })
    "signature": "Xe8B94RD30Azj2ea...8BmZIRtcSKPSd8gU"
  }),
  "signature": "5TWiqIYQfIDfALQv...x9C2mg8JGPxl5bI4"
}
```

SHAKEN flow

- Step 4 - apply for new certificate with a CSR

```
POST /acme/new-app 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": "5XJ1L3lEkMG7tR6pA00clA",
    "url": “https://sti-ca.com/acme/new-app"
  })
  "payload": base64url({
    "csr": "5jNudRx6Ye4HzKEqT5...FS6aKdZeGsysoCo4H9P",
    "notBefore": "2016-01-01T00:00:00Z",
    "notAfter": "2016-01-08T00:00:00Z"
  }),
  "signature": "H6ZXtGjTZyUnPeKn...wEA4TklBdh3e454g"
}
```

SHAKEN flow

- Step 5 - server provides required challenge authorization with URL, client must respond with authorization before “expires” time

```
HTTP/1.1 201 Created
Replay-Nonce: MYAuvOpaoIiywTezizk5vw
Location: https://sti-ca.com/acme/app/asdf

{
  "status": "pending",
  "expires": "2015-03-01T14:09:00Z",
  "csr": "jcRf4uXra7FGYW5ZMewvV...rhlnznwy8YbpMGqwidEXfE",
  "notBefore": "2016-01-01T00:00:00Z",
  "notAfter": "2016-01-08T00:00:00Z",

  "requirements": [
    {
      "type": "authorization",
      "status": "valid",
      "url": “https://sti-ca.com/acme/authz/1234"
    }
  ]
}
```

SHAKEN flow

- Step 6 - get authorization

```
GET /acme/authz/1234 HTTP/1.1
Host: sti-ca.com

HTTP/1.1 200 OK
Content-Type: application/json
Link: <https://sti-ca.com/acme/some-directory>;rel=“directory"

{
  "status": "pending",

  "identifier": {
    "type": "spid",
    "value": "123"
  },

  "challenges": [
    {
      "type": "token",
      "url": “https://sti-ca.com/authz/asdf/0"
    }
  ],
}
```

SHAKEN flow

- Step 7 - respond to the challenge

```
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({
    "type": "token",
    "keyAuthorization": "IlirfxKKXA...vb29HhjjLPSggwiE"
  }),
  "signature": "9cbg5JO1Gf5YLjjz...SpkUfcdPai9uVYYQ"
}
```

SHAKEN flow

- Step 8 - check on status of authorization

```
GET /acme/authz/asdf HTTP/1.1
Host: sti-ca.com

HTTP/1.1 200 OK

{
  "status": "valid",
  "expires": "2015-03-01T14:09:00Z",
  "identifier": {
    "type": "spid",
    "value": "123"
  },

  "challenges": [
    {
      "type": "token"
      "status": "valid",
      "validated": "2014-12-01T12:05:00Z"
    }
  ]
}
```

SHAKEN flow

- Step 9 - download the certificate

```
GET /acme/cert/asdf HTTP/1.1
Host: sti-ca.com
Accept: application/pkix-cert

HTTP/1.1 200 OK
Content-Type: application/pkix-cert
Link: <https://sti-ca.com/acme/ca-cert>;rel=“up";title="issuer"
Link: <https://sti-ca.com/acme/revoke-cert>;rel="revoke"
Link: <https://sti-ca.com/acme/app/asdf>;rel="author"
Link: <https://sti-ca.com/acme/sct/asdf>;rel="ct-sct"
Link: <https://sti-ca.com/acme/some-directory>;rel="directory"

-----BEGIN CERTIFICATE-----
[End-entity certificate contents]
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
[Issuer certificate contents]
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
[Other certificate contents]
-----END CERTIFICATE-----
```