SHAKEN - Certificate Framework
Administration using ACME/OAuth

IPNNI Task Force
Chris Wendt – Comcast
(revisions on 8/02/17 by Mary Barnes)

Based on draft-ietf-acme-acme-07
Auth framework

- **STI-PA**
  - Provide private/public key

- **STI-CA**
  - Retrieve certificate
  - Validate JWT using certificate in x5u

- **ACME**

- **SP-KMS**
  - Generate JWT with x5u of public key
ACME - Account object

- ACME account 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
  - **orders** (required, string) - URI from which a list of orders submitted by this account can be fetched via a GET

Example Registration object:
```json
{
    "contact": [
        "mailto:cert-admin@example.com",
        "tel:+12025551212"
    ],
    "terms-of-service-agreed": true,
    "orders": "https://example.com/acme/reg/1/apps"
}
```

Example response to applications URL:
```json
{
    "orders": [
        "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 - Order object

- ACME Order 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
  - **authorizations** (required, array) – authorizations the client needs to fulfill before granting certificate
  - **certificate** (optional, string) - URL for the certificate issued for this application
ACME – Order 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",
    "authorizations": [
        {
            "url": "https://example.com/acme/authz/1234"
        }
    ],
    "certificate": "https://example.com/acme/cert/1234"
}

- ACME Order object represents a client’s request for a certificate, and it’s lifecycle through to issuance.
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 "TNAuthList")
  - **value** (required, string) - identifier itself
**status** (required, string) - status of the authorization. “pending”, “processing”, “valid”, “invalid”, “revoked” - default value is “pending”.

**scope** (optional, string) – if present must contain a URL for an order, such that this authorization is only valid for the specific order.

**expires** (optional, string) - if present, contains URI for an order resource, if absent, then CA MUST consider authorization valid for all orders.

**challenges** (required, array) - an array of challenges used for authorization:

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

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

  "challenges": [
    {
      "type": "spc-token-01",
      "status": "valid",
      "validated": "2014-12-01T12:05:00Z",
      "spcAuthorization": "SXQe-2XODaDxNR...vb29HhjjLPSggwiE"
    }
  ]
}
```
ACME object directory

- directory
  - new-nonce
  - new-acct
    - reg/account
  - new-authz
  - new-order
    - order
    - authz
      - challenge
    - cert
    - cert-chain
  - revoke-cert
ACME - high level flow

- ACME high level request flow:
  - Get a nonce - HEAD new-nonce
  - Create Account - POST new-account
  - Apply for a cert (order) - POST new-order
  - 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

- Request CA token: GET /sti-pa/account/:id/token
  - 200 OK
- Establish account with Administrator with secure credentials
- Process New SPC and add account
- Provide SP account access through secure means
- Request nonce: HEAD /acme/new-nonce
  - 200 OK
- Registration request: POST /acme/new-order
  - 201 Created
- Create order
SHAKEN ACME/Auth call flow - certificate request

- **Apply for certificate** - POST /acme/new-order
  - 201 Created
  - Create new order and authz object

- **Get Authz** - GET /acme/authz/1234
  - 200 OK
  - Provide URL for auth challenge

- Check for fresh token, if expired request new token from STI-PA

- **Set token to respond to challenge** - POST /acme/authz/1234/0
  - 200 OK with updated challenge in body

- Request public key to validate signature of token is administrator signed - GET /sti-pa/cert.crt
  - 200 OK

- Validate token in challenge with admin cert, and set authz status to “valid” for success

- **Check that authz status is “valid”** - GET /acme/authz/1234
  - 200 OK - with valid then continue, if “pending”, try authz until “valid”

- Once authz is “valid” STI-CA will process CSR and create certificate

- **Download the certificate** - GET /acme/cert/1234
  - 200 OK - with certificate in body
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: oFvlFP1whRIYS2jTaXbA
  Cache-Control: no-store

- **Step 2 – New Account Request - likely done per SP-KMS node**

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

  ```
  {
  "protected": base64url({
    "alg": "ES256",
    "jwk": {...},
    "nonce": "6S8lqOGY7eL2lsGoTZYifg",
    "url": "https://sti-ca.com/acme/new-acct"
  }),
  "payload": base64url({
    "terms-of-service-agreed": true,
    "contact": [
      "mailto:cert-admin-sp-kms01@sp.com",
      "tel:+12155551212"
    ]
  })),
  "signature": "RZPOnYoPs1PhjszF...-nhX1qtOFPB519l"
  }
  ``
SHAKEN flow

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

```plaintext
HTTP/1.1 201 Created
Content-Type: application/json
Replay-Nonce: D8s4D2mLs8Vn-goWuPQeKA
Location: https://sti-ca.com/acme/acct/1
Link: <https://sti-ca.com/acme/some-directory>;rel="directory"

{
    "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 */,
      "url": "https://sti-ca.com/acme/key-change"
    ),
    "payload": base64url(
      "account": "https://sti-ca.com/acme/reg/asdf",
      "newKey": /* new key */
    )
  ),
  "signature": "Xe8B94RD30Azj2ea...8BmZlRtcSKPSd8gU"
},
"signature": "5TWiqfYQflDfALQv...x9C2mg8JGPxl5bI4"
}
```
SHAKEN flow

- Step 4 - apply for new certificate with a CSR

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

{
  "protected": base64url({
    "alg": "ES256",
    "kid": "https://sti-ca.com/acme/acct/1",
    "nonce": "5XJ1L3lEkMG7tR6pA00clA",
    "url": "https://sti-ca.com/acme/new-order"
  })
  "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: MYAuvOpaoliywTezizk5vw
Location: https://sti-ca.com/acme/order/asdf

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

{
  "status": "pending",
  "identifier": {
    "type": "TNAuthList",
    "value": ["X0001", "X0002", "X0003"]
  },
  "challenges": [
    {
      "type": "spc-token-01",
      "url": "https://sti-ca.com/authz/1234/0",
      "token": "DGyRejmcovFefeNFDGkA"
    }
  ]
}
SHAKEN flow

- Step 7 - respond to the challenge

```plaintext
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/acct/1",
    "nonce": "Q_s3MWoqT05TrdkM2MTDcw",
    "url": "https://sti-ca.com/acme/authz/asdf/0"
  }),
  "payload": base64url({
    "type": "spc-token-01",
    "spcAuthorization": "DGyRejmCovFefeNfDGDkfAllir...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": "TNAuthList",
    "value": ["X0001", "X0002", "X0003"]
  },

  "challenges": [?
    {?
      "type": "spc-token"
      "status": "valid",
      "validated": "2014-12-01T12:05:00Z"
      "spcAuthorization": "DGyRejmCovFefeNfDGDKfAllir...vb29HhjjLPSggwiE"
    }
  ]
}
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/pem-certificate-chain

Link: <https://sti-ca.com/acme/some-directory>;rel="index"

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