ACME Identifiers and Challenges for VoIP Service Providers

draft-ietf-acme-service-provider-01

mary.ietf.barnes@gmail.com
chris-ietf@chriswendt.net

July 21, 2017
Changes since last version

- Added details about the Service Provider Code Token including function in SHAKEN and format
- Changed format of challenge type from:
  - keyAuthorization – Token |.| base64url(JWK_Thumbprint(accountKey))
  to:
  - spcAuthorization: Token |.| spcAuthzToken (which contains the “fingerprint”)
- Described processing of challenge response by the ACME server
SHAKEN Certificate Management Architecture

- **STI-PA**: Policy Administrator
- **SP-KMS**: Service Provider Key Management Server
- **SKS**: Secure Key Store

Diagram:

- STI-PA
  - HTTPS
  - Service Provider Code Token
- STI-CA
  - HTTPS
  - ACME
- SP-KMS
  - Private Key
  - HTTPS
  - Public Key Certificate
- SKS
  - HTTPS
- STI-AS
  - HTTPS
- STI-CR
  - HTTPS
- STI-VS

Legend:

- HTTPS
- STI-AS: Service Identity Authority Server
- STI-CA: Service Identity Certificate Authority
- STI-CR: Service Identity Certificate Revocation
- STI-PA: Service Identity Policy Administrator
STI-PA Account Setup, SPC Token Acquisition, ACME Acct Registration

STI-PA Administrator

- Process New SPC and add account
- Provide SP account access through secure means

SP-KMS

- Establish account with Administrator with secure credentials

STI-CA

Request CA token - GET /sti-pa/account/id/token

200 OK

Receive token with store with TTL, periodically refresh on demand use for all authz challenges

Request nonce - HEAD /acme/new-nonce

200 OK

Registration request - POST /acme/new-account

201 Created

Create registration
Certificate Acquisition

1. **Apply for certificate** - POST /acme/new-order
2. **201 Created**
3. **Create new application and authz object**
4. **Get Authz** - GET /acme/authz/1234
5. **200 OK**
6. **Provide URL for auth challenge**
7. **Check for fresh token, if expired request new token from STI-PA**
8. **Set token to respond to challenge** - POST /acme/authz/1234/0
9. **200 OK with updated challenge in body**
10. **Request public key to validate signature of token is administrator signed** - GET /sti-pa/cert.crt
11. **200 OK**
12. **Validate token in challenge with admin cert, and set authz status to “valid” for success**
13. **Check that authz status is “valid”** - GET /acme/authz/1234
14. **200 OK - with valid then continue, if “pending”, try authz until “valid”**
15. **Once authz is “valid” STI-CA will process CSR and create certificate**
16. **Download the certificate** - GET /acme/cert/1234
17. **200 OK - with certificate in body**
Service Provider Code Token

JWT Header:
- alg: Defines the algorithm used in the signature of the token. For Service Provider Code tokens, the algorithm MUST be "ES256".
- typ: Set to standard "JWT" value.
- x5u: Defines the URL of the certificate of the STI-PA validating the Service Provider Code.

JWT Payload:
- sub (*): Service Provider Code value being validated in the form of a JSON array of ASCII strings.
- iat: DateTime value of the time and date the token was issued.
- nbf: DateTime value of the starting time and date that the token is valid.
- exp: DateTime value of the ending time and date that the token expires.
- fingerprint: (Certificate) key fingerprint of the ACME credentials the Service Provider used to create an account with the CA.

“fingerprint” is of the form:
base64url(JWK_Thumbprint(accountKey))

* For ATIS-1000080, only a single Service Provider Code is required in the “sub” field.
Discussion points

1. “Identifier” is specific to STIR TNAuthList (includes both TNs and Service Provider Codes)
   • Draft-ietf-acme-telephone defines a “tn” identifier
     • Could this re-use TNAuthList and define a new challenge type (tn-01)?
       • Depends upon the answer to 2.

2. Challenge type is specific to Service Provider Code Tokens
   • Could we use this as a generic challenge type for tokens of the same form?
     • Possibly *but* could slow down progression of this document