
# Other Security Updates

Michael McCool

Oct 22, 2020

# Outline

- TD PRs, Merged
  - Combination Scheme (PR 944)
  - OAuth2.0 Flows (PR 927)
- TD PRs, Deferred
  - Simplified Inline Security Definitions (PR 945)
  - Signed TDs with LD-Proofs (PR 943)
- Security and Privacy Guidelines
  - End-to-End Security Discussion
- Use Cases
  - Review of Use Cases (eg Retail, based on Conexxus Security Checklist)

# Combo Scheme: Situation and Problems

Situation in TD 1.0:

- Array of security definition names are allowed in the "security" field
- Multiple names are combined with "AND"
- "OR" must be done by having multiple "forms"

- Problems:
  - "AND" interpretation is inconsistent with OpenAPI
  - "OR" leads to redundant "forms" with duplicate non-security information

Use Case for AND:

- Proxies having one scheme, endpoint having another

Use Case for OR:

- Endpoints and/or proxies can allow different authentication/authorization options

- Use Case for AND/OR combination:
  - Proxy allowing multiple schemes and endpoint also allowing multiple (potentially different) schemes

# Combo Scheme

- Allows AND or OR combinations of schemes
- Can be nested (probably we should add an assertion to limit nesting depth to 2) to give AND/OR combination for proxy/endpoint multiple schemes use case
- Syntax follows that used by JSON Schema
- Examples in TD 1.1 spec (right).

## AND (allOf) Combination

```
"securityDefinitions": {
  "proxy_sc": {
    "scheme": "digest",
    "proxy": "https://portal.example.com/"
  },
  "bearer_sc": {
    "scheme": "bearer",
    "in":"header",
    "format": "jwt",
    "alg": "ES256",
    "authorization":
      "https://servient.example.com:8443/"
  },
  "combo_sc": {
    "scheme": "combo",
    "allOf": ["proxy_sc", "bearer_sc"]
  }
},
"security": "combo_sc",
```

# Combo Scheme

- Allows AND or OR combinations of schemes
- Can be nested (probably we should add an assertion to limit nesting depth to 2) to give AND/OR combination for proxy/endpoint multiple schemes use case
- Syntax follows that used by JSON Schema
- Examples in TD 1.1 spec (right).

## OR (oneOf) Combination

```
"securityDefinitions": {
  "basic_sc": {
    "scheme": "basic"
  },
  "digest_sc": {
    "scheme": "digest"
  },
  "bearer_sc": {
    "scheme": "bearer"
  },
  "combo_sc": {
    "scheme": "combo",
    "oneOf": [
      "basic_sc",
      "digest_sc",
      "bearer_sc"
    ]
  }
},
"security": "combo_sc",
```

# Simplified Inline Security Definitions

- In simple use cases the need for separate "securityDefinitions" and "security" activations is annoying
- Making up a name just to use it once is annoying
- This PR allows a SecurityScheme object to be given instead of a string in a "security" field
- Unfortunately, breaks compatibility, so deferred to 2.0

Currently, must do:

```
"securityDefinitions": {
  "basic_sc": {
    "scheme": "basic"
},
"security": "basic_sc",
```

This PR would *also* allow:

```
"security": {
  "scheme": "basic"
}
```

# Use of Combo with Inline Definitions

- If "combo" is made the default for "scheme",
- *and* we also extend the arguments to "allOf" and "oneOf" in the combo scheme to also allows objects consistently,
- *then* a simple syntax is enabled for inline AND/OR Combinations

```
"security": {
  "allOf": [
    {
      "scheme": "digest",
      "proxy": "https://portal.example.com/"
    },
    {
      "scheme": "bearer",
      ...
    }
  ]
}
```

```
"security": {
   "oneOf": [
      { "scheme": "basic_sc" },
      { "scheme": "digest_sc" },
      { "scheme": "bearer_sc" }
    ]
}
```

# Signing TDs and LD-PROOFS

- Signing TDs to prevent/detect tampering would be useful
- DID previously included a "proof" and "proofChain"
- LD-PROOF needs canonicalization of JSON-LD for RDF round-tripping
  - Unfortunately method not finalized
  - But it sounds like it might be soon
  - Currently deferred...
- Could use"plain" JWS signing instead
  - but it would be a nuisance for RDF-based directories

```
"proof": {
  "type": "Ed25519Signature2018",
  "proofPurpose": "assertionMethod",
  "created": "2017-09-23T20:21:34Z",
  "verificationMethod":
    "did:example:123456#key1",
  "challenge":
    "2bbgh3dgjg2302d-d2b3gi423d42",
  "domain": "example.org",
  "jws": "eyJ0eXAiOiJK...gFWFOEjXk"
}
```