Applying COSE Signatures for YANG Data Provenance

draft-lopez-opsawg-yang-provenance-02

D. López, A. Pastor (*Telefónica)*

A. Huang Feng (*INSA-Lyon*)

H. Birkholz (Fraunhofer SIT)

IETF#119, Brisbane (AU), March 2024

Prov·e·nance | ˈprävən(ə)ns |

- More specifically, *data provenance*
  - A documented trail accounting for the origin of a piece of data and where it has moved from to where it is presently
- Assurance of the origin and integrity of YANG datasets
  - Motivated by the discussion on metadata manifests
    - draft-ietf-opsawg- collected-data-manifest
  - Whenever the dataset is used beyond an original online flow
    - Use of data intermediaries, such as data lakes
    - AI/ML training and validation
    - Audit trails, including forensics evidence

The Foundations

- Current practice relies on the transport protocol
  - Identiy and crypto material in TLS, SSH…
  - Suitable for online flows
  - Contentious if used offline
- This proposal implies native support
  - Avoiding transitive trust
  - Very low impact on models using it
  - Recursion
- Based on COSE
  - Concise
  - Detached payload

Applying Provenance

- Convened by a leaf element
  - Containing a COSE signature bitstring
  - Of type `provenance-signature`, defined as:

```
typedef provenance-signature {
      type binary;
      description
        "The provenance-signature type represents a digital signature
         associated to the enclosing element. The signature is based
         on COSE and generated using a canonicalized version of the
         enclosing element.";
      reference "draft-lopez-opsawg-yang-provenance";
}
```

Provenance Signatures

- COSE single signature string with *[nil]* payload
  - Algorithm-identifier, following COSE conventions and registries.
  - KID (Key ID), locally used and interpreted by the signer and the validator
  - The serialization method:
    - xml, json, cbor
  - Algorithm-parameters, following the COSE conventions
  - The signature, using as external supplied data
    - The whole element enclosing the signature leaf
    - Without the signature leaf element
    - Applying the corresponding canonicalization method

```
COSE_Sign1 = [
protected /algorithm-identifier, kid, serialization-method/
unprotected /algorithm-parameters/
signature /using as external data the content
           of the (meta-)data without the signature leaf/
]
```

Enclosing Methods – Provenance Elements

1. Add a leaf element containing a provenance signature
   - One and only one in the enclosing element
   - Anywhere

2. Include a provenance signature in NETCONF Event Notifications and YANG-Push Notifications
   - `ietf-notification-provenance` augmentation within the `ietf-notification module`

```
module: ietf-notification-provenance
    augment-structure /inotif:notification:
      +-- notification-provenance?   iyangprov:provenance-signature
```

```
module: ietf-notification
  structure notification:
    +-- eventTime                              yang:date-and-time
    +-- inotifprov:notification-provenance?    iyangprov:provenance-signature
```

```
module: ietf-platform-manifest
  +--ro platforms
  +--ro platform* [id]
  +--ro platform-provenance? provenance-signature
  +--ro id string
  +--ro name? string
  +--ro vendor? string
  +--ro vendor-pen? uint32
  +--ro software-version? string
  +--ro software-flavor? string
  +--ro os-version? string
  +--ro os-type? string
  +--ro yang-push-streams
  |  +--ro stream* [name]
  |     +--ro name
  |     +--ro description?
  +--ro yang-library
  + . . .
  .
  .
  .
```

Enclosing Methods – Metadata

3. Include a provenance signature as metadata in YANG instance data
   - In YANG instance data files, for data at rest.

```
module: ietf-yang-instance-data-provenance
  augment-structure instance-data-set:
    +--provenance-string?   provenance-signature
```

4. Include provenance signatures as YANG annotations
   - Not requiring modification of existing YANG schemas

```
md:annotation provenance-string {
          type provenance-signature;
          description
            "This annotation contains a digital signature corresponding
             to the YANG element in which it appears.";
        }
```

The Recursion Issue

- The draft only allows a provenance signature for a given enclosure
- But they can be recursive
  - Inner non-leaf elements in method 1
  - Within the `notificationContent` in method 2
  - Within each `content-data` in the `instance-data-set` in method 3
  - As part of the element the annotation applies to in method 4
  - Even combined at different recursion levels
- The rules for (detached) signature generation and verification are intended to support this
- Making recursive provenance validation
  - Data aggregation
  - Specific validation of relevant children

What Comes Next

- Previous versions were presented, discussed and refined in OPSAWG
- AD suggested to discuss this proposal in NETMOD
- Open to
  - Comments and suggestions
  - Moving the draft if the group thinks it is adequate
- Immediate work to do
  - Address sections TBD
  - Continue gathering implementation experience
- Propose adoption as soon as an operational demonstrator is available
  - Promising early PoC results