Custom Roles

XDR IROH

Yann Esposito

[2023-10-03 Tue 15:30]

1 Current state

1.1 Listing Roles (already by org)

`GET /iroh/profile/roles`

Provide a data structure with describing all roles for an Org:

- 3 roles for XDR (admin, user, sat)
- 2 roles for SX (admin, user)

1.2 Role Permissions

The role associated to a user do not necessarily matches the user permission.

The role is only one of the component to use to determine a token or even a user permissions. The permissions are represented by *scopes* which are computed using:

- the user role
- the org properties (activated or not, XDR or not etc…)
- entitlements (not in use but will probably be the case in the future)

1.3 Role Permissions (Tokens)

- the user scopes
- as well as the client scopes
- as well as the scopes requested during the OAuth2 authorization flow

1.4 Current response for an XDR-enabled org

```
GET /iroh/profile/roles
{:admin {:english {:only-role-name "administrator",
                   :adjective "an",
                   :only-role-name-capitalized "Administrator",
                   :english-role-name "an administrator"},
         :role-name "Administrator",
         :role-id "admin",
         :role-description "An admin of users.",
         :visibility "public"},
 :sat {:english {:only-role-name "security analyst",
                 :adjective "a",
                 :only-role-name-capitalized "Security Analyst",
                 :english-role-name "a security analyst"},
       :role-name "Security Analyst",
       :role-id "sat",
       :role-description
       "No account admin. SXO read only + run existing workflows.",
       :visibility "public"},
 :user {:english {:only-role-name "incident responder",
                  :adjective "an",
                  :only-role-name-capitalized "Incident Responder",
                  :english-role-name "an incident responder"},
        :role-name "Incident Responder",
        :role-id "user",
        :role-description
        "This is the closest to current user role:- no account administration- cannot
        :visibility "public"}}
```

1.5 Current response for an SX-only org

```
GET /iroh/profile/roles
{:admin {:english {:only-role-name "admin",
                     :adjective "an",
                     :only-role-name-capitalized "Admin",
                     :english-role-name "an admin"},
           :role-name "Admin",
           :role-id "admin",
           :role-description "An admin of users.",
          :visibility "public"},
   :user {:english {:only-role-name "user",
                    :adjective "a",
                    :only-role-name-capitalized "User",
                    :english-role-name "a user"},
          :role-name "User",
          :role-id "user",
          :role-description "A standard user.",
          :visibility "public"}}
```

1.6 What the API already support

- list all roles for every Org
- change the role of a user
- support roles during invitation and Org access request
- expose a permissions endpoint to check permission access independently of the role
- read/write access restriction
- fine grained *resource* target in the scopes `enrich` → `enrich/observables/observe:write`

1.7 What the API does not support

- No support for create+update but not delete.
- No support for multiple roles (not sure what it means yet)
- No support for custom role creation (obviously)
  - No scopes API for roles

2 Expected Changes

2.1 New API: (exhaustive scopes list)

Exhaustive list of scopes as a forest structure

```
[{:scope "global-intel"
  (optional :description) ,,,
  :accessors ["read"]
  :sub-scopes [{:scope "global-intel/incident"
                 :accessors ["read"]}
                {:scope "global-intel/sighting"
                 :accessors ["read"]}
                ,,,]}
 {:scope "private-intel"
  (optional :description) ,,,
  :accessors ["rw","read","write"]
  :sub-scopes [{,,,}]}]
```

2.2 New API (maybe?)

Expose only a subset of scopes aliases pre-negociated with UX/UI/Doc team:

```
[{:scope-alias "threat-hunt"
  :scopes ["enrich/observables/observe:read","inspect","investigation"]
  :description ,,,,}
 {:scope-alias "incidents"
  :scopes ["private-intel","global-intel:read"]
  :description ,,,}
 ,,, ]
```

2.3 New API: CRUD+Search

API to manage new custom roles

```
(s/defschema NewRole
  {:role-name        s/Str
   :role-description s/Str
   :provided-scopes  Scopes})

(s/defschema Role
  (st/merge NewRole
            {:id s/Str
             :created-at Date
             :updated-at Date}))
```

2.4 Existing APIs

The `GET /iroh/profile/roles` will look like today + added the new custom roles that will look like:

```
{:admin ...
 :sat ...
 :user ...
 :role-d394db9e-613f-11ee-aff9-325096b39f47
 {:role-name "My Company Custom Role"
  :role-description "This is a role that is read only except for workflows"
  :role-id :role-d394db9e-613f-11ee-aff9-325096b39f47
  :visibility "org"
  :associated-scopes #{"inspect:read" "ao" "insights:read" "profile:read"}}

 :role-8891b9f4-6140-11ee-8e1a-325096b39f47
 {:role-name "Manager"
  :role-description "Only for Sam who manage this team but should not directly act"
  :role-id :role-8891b9f4-6140-11ee-8e1a-325096b39f47
  :visibility "org"
  :associated-scopes #{"inspect:read" "ao:read" "insights:read" "profile:read" "users"
```

- `visibility`; `org` for custom, `public` for global.
- `associated-scopes`; only for role management UI

2.5 Introduce sub-accessors (maybe?)

Today: `read`, `write`

```
inspect = inspect:rw
        = inspect:read + inspect:write.
```

Tomorrow: introduce `read:get`, `read:search`, `write:create`, `write:update`, `write:delete`, `write:execute`.

2.5.1 Equivalence of new accessors

```
rw = read + write

read  = read:get      # GET by id
      + read:search   # GET/POST search entities
write = write:create  # POST create new entity
      + write:update  # PUT/PATCH
      + write:delete  # DELETE
      + write:execute # POST to trigger action
```

3 Most important points

- Dynamic role `ids`. **Must use the API**
  - when you call `/iroh/profile/whoami`
  - when you look into the JWT
  - **note**: potentially a list of roles!
- `associated-scopes` field only useful for the Role Management UI.
- Use `/iroh/profile/permissions`
- can also use `scopes` claim if present

3.1 Multiple Roles

- if union of roles for the same user: Expect the role to be a sorted comma separated role ids like; `admin,role-344,sat,user` (which would be equivalent to `admin` here)
- if one role per session, then we will use different `user-id` and thus the role must appear in the UIs (Registration UI, Org switching, etc. . .)