Modelling and evaluating complex user entitlements in directory services using JSON and REST

Mark Perry, APAC CTO

markperryau
markperry@pingidentity.com
Based in Melbourne, Australia
Nearly 30 years in IT
  • Most in Identity
  • Likes whiskey, good coffee
  • Passionate about Australian Rules Football
  • Plays golf
Career in IAM and Digital Identity
  • IBM
  • /etc/password, YP, NIS
Career in IAM and Digital Identity

• IBM
  • /etc/password, YP, NIS

• Netscape
  • Directory Server v1 & 3
• Career in IAM and Digital Identity
  • IBM
    • /etc/password, YP, NIS
  • Netscape
    • Directory Server v1 & 3
  • Sun Microsystems
    • SunAM, SunIM, Liberty Alliance, SAML
• Career in IAM and Digital Identity
  • IBM
    • /etc/password, YP, NIS
  • Netscape
    • Directory Server v1 & 3
  • Sun Microsystems
    • SunAM, SunIM, Liberty Alliance, SAML
  • Ping Identity
    • OAuth2, OIDC
    • Open Banking
LDAPsql

- Command line tool
- PL/SQL-like
- Use SQL syntax against directories
- Create views, synonyms, etc.
- DISTINCT, COUNT, MAX, MIN, ORDER BY ASC DESC, etc.
- Written in Java
- Pretty output & LDIF
- CRUD
  - INSERT is wizard-driven
## Ping Data Products

pingidentity.com

<table>
<thead>
<tr>
<th>Product</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>PingDirectory</td>
<td>Data Store</td>
</tr>
<tr>
<td></td>
<td>LDAP, REST, JSON</td>
</tr>
<tr>
<td>Ping DataSync</td>
<td>High speed data synchronisation</td>
</tr>
<tr>
<td>Ping Directory Proxy</td>
<td>LDAP proxy</td>
</tr>
<tr>
<td></td>
<td>Load balancing</td>
</tr>
<tr>
<td></td>
<td>Rate limiting</td>
</tr>
<tr>
<td></td>
<td>Operation balancing</td>
</tr>
<tr>
<td>Ping DataGovernance</td>
<td>Data Access Proxy/Service</td>
</tr>
<tr>
<td></td>
<td>Policy Engine</td>
</tr>
<tr>
<td></td>
<td>Data transformation</td>
</tr>
</tbody>
</table>
Entitlements
What can Bob do in this context?

What do you know about Alice? (all authorisations)
Centralisation
  – Single View, Audit
Control
  – Limit rogue admins
Performance
  – vs CRM, MDM, etc.
Access
  – Developer friendly

Why Store Entitlements In The Directory?
Ways of Managing Entitlements

- Group memberships
  - memberOf
- nsRole
  - Static, dynamic
- Attribute-based
  - Delimited attribute
  - JSON
Using JSON Attributes

- Schema Definition
  dn: cn=schema
  objectClass: top
  objectClass: ldapSubentry
  objectClass: subschema
  cn: schema
  attributeTypes: (jsonAttr1-OID NAME 'entitlement' DESC 'entitlement json attribute' EQUALITY jsonObjectExactMatch SYNTAX 1.3.6.1.4.1.30221.2.3.4 USAGE userApplications )
  objectClasses: (jsonObjectClass-OID NAME 'jsonObjectClass' AUXILIARY MAY entitlement )
Modelling JSON Attributes

- **Matching Rules**
  - jsonObjectCaseSensitiveNamesCaseSensitiveValues
  - jsonObjectCaseInsensitiveNamesCaseInsensitiveValues
  - jsonObjectCaseInsensitiveNamesCaseSensitiveValues

- **Constraints**
  - Requiring values of the field to have a specified data type.
  - Indicating whether the field is required or optional.
  - Indicating whether the field can have multiple values in an array. If a field is permitted to have array values, restrictions can also be placed on the number of elements that can be present in the array.
  - Indicating whether the field can have a value that is the null primitive as an alternative to values of the indicated data type.
  - Restricting values of string fields to a predefined set of values, that match a given regular expression, or a specified length.
  - Restricting values of numeric fields with upper and lower bounds.
Using JSON Attributes

- Searches

**Equals field filter type**

```
$ bin/ldapsearch -p 1389 -b dc=example,dc=com -D
"cn=Directory Manager" -w password
'(entitlement:jsonObjectFilterExtensibleMatch:={
"filterType" : "equals", "field" : ["stuff", "onetype", "name"], "value" : "John Doe" })'
```

**Contains field filter type**

```
$ bin/ldapsearch -p 1389 -b dc=example,dc=com -D
"cn=Directory Manager" -w password
'(entitlement:jsonObjectFilterExtensibleMatch:={
"filterType" : "containsField", "field" : "age", "expectedType" : "number" })'
```
Modelling Entitlements

{"entitlement": {
"location": "East",
"role": "Teller"
}}

Nesting is possible!
{"entitlement": {
"location": "East",
"role": {
"desc": "Teller",
"jobCode": "3458645"
... 

}}
}
Examples

- **Employees**
  - Access rights for applications
  - Role, location, time, etc.
  - Relationships
    › Teams
    › Cross-brand

- **Consumers**
  - Access rights for services
  - Cross-product
  - Consent
  - Relationships
    › Families
    › Associations
  - Churn
Evaluating Entitlements

- How much to model and maintain in schema?
- Do different apps need different values?
- Represent in OAuth scopes? Or apps to perform REST calls at runtime?
- Dynamic creation of entitlements in a data access gateway
Ping Data Governance

API
Proxy
Fine-Grained
Access Control
Policy
Engine

Clients

(PFP - Policy Enforcement Point)
(PDP - Policy Decision Point)

PingDataGovernance™

Entitlements

(PIP - Policy Information Points)

μAPI

Sensitive
Data

19 Confidential - Do not distribute. Copyright © 2019, Ping Identity Corporation. All rights reserved.
Enables Flexibility and Ease of Modelling

- Can modify data returned to clients depending on their requirements
- OAuth Token Exchange (soon)
- Drag and drop policy creation tool (no XACML!)
Summary

- Entitlements are growing in importance for customers
  - And they want guidance from us
- Modelling entitlements using JSON
- Evaluating entitlements via REST APIs
- Directories continue to have a huge role to play in securing access