Cloud Storage
Pluggable Access Control

David Slik
NetApp, Inc.
Agenda

- Access Control
  - The classic models: DAC, MAC & RBAC
  - Emerging access control methodologies
- Access Control in Cloud Storage Systems
  - What’s wrong with where we are?
  - Access Control in CDMI
- CDMI ACL Delegation
- Demo
What is Access Control?

- A subset of Authorization
- Determines WHAT the WHO can do to THIS
  - WHAT – Identity of the action (read, write, execute, delete, etc)
  - WHO – Identity of the subject (process, computer, human, etc)
  - THIS – Identity of the object (file, program, sensor, actuator, etc)
- Distinct from, but usually coupled with, authentication
  - Authentication is the means by which to determine the WHO
- Storage system provides the WHAT (actions) and the THIS (objects)
What is Access Control?

- A client issues a request
- A protocol encapsulates the client’s credentials and the operation that is to be performed
- A server verifies client’s identity from the presented credentials, and determines the permissions for the referenced object
What is Access Control?

- Resolution of credentials to identity is often delegated.
  - E.g. Active Directory
- Permissions have often been tightly coupled with the objects and actions themselves.
Access Control: The Classic Models

- **DAC: Discretionary Access Control**
  - Control strictly defined by administrator-defined policy
  - E.g. Classification (Secret, Top Secret, etc)

- **MAC: Mandatory Access Control**
  - Control defined by object’s owner (or member of a group)
  - E.g. **storage ACLs**

- **RBAC: Role-Based Access Control**
  - Control defined by organizational role (identity -> role -> access)
  - ACL “groups” are often (ab)used for this purpose
Access Control: Emerging Methodologies

- **ABAC: Attribute Based Access Control**
  - Refinement to RBAC to take into account attributes (e.g. XACML)

- **GBAC: Graph Based Access Control**
  - Primarily found on social networks, e.g. “Friend of a friend”
  - Declarative access rights based on semantic graphs

- **Biometric**
  - Mixing of Identity and Access Control

- **Location Based Access Control**
  - Geofencing, etc.
Access Control: Emerging Methodologies

- CBAC: Context Based Access Control
  - Access allowed or disallowed based on context of surrounding operations
  - E.g. disallowing whole file replacement with high Shannon Entropy (ransomware protection)

- Capability Based Access Control
  - Operations allowed when an access token (capability) is possessed.
  - E.g. S3 signed URLs

- Risk Based Access Control
  - Takes into account risk of granting access. Deep learning, AI-based?
Today’s Cloud Storage Systems are built around the traditional storage model:
- A series of operations (Create, Read, Update, Delete, Copy, etc)
- Performed on objects
- With identity determined by verifying credentials included in the request (typically as HTTP headers)

All major cloud storage systems support authentication delegation
- Active Directory, MFA, Social Networks, Customer systems, etc.

No cloud storage system currently supports Access Control delegation
Access Control in Cloud Storage Systems

- Why is this a problem?
  - What if the provided access control system isn’t suitable or sufficiently expressive or extensible? (e.g. DoD use cases)
  - What if the provided access control system isn’t consistent across clouds? (e.g. Cross-cloud portability)
  - What if the provided access control system isn’t sufficiently unchanging? (e.g. Unexpected software upgrades)
  - What if the provided access control system isn’t functionality that should be in the cloud? (In-house decision making)
Access Control in Cloud Storage Systems

- NetApp has had access control delegation for file system operations since 2002:
  - create, open, rename, delete, read, write, create_dir, rename_dir, delete_dir, getattr, setattr
  - Sends additional details (file info, user info, etc)
  - Support for synchronous blocking calls and asynchronous notifications
- Fpolicy has enabled many different third-party integrations:
  - File screening, File access reporting, User and directory quotas, HSM, Archiving, File replication, Data governance
CDMI Access Control

CDMI Client

CDMI GET
+ Credentials
+ Object

GET Response

CDMI Server

AD Lookup
+ Credentials

AD Result
+ Principle

Active Directory

ACL Evaluation
CDMI Access Control Delegation

- CDMI introduced a form of Delegated Access Control in 2015/2016, known as the Delegated Access Control (DAC) Extension
  - Primarily intended to be used with encrypted objects
  - Specifies metadata indicating how to submit a DAC request to a DAC provider, which returns a DAC response
  - Assumes asynchronous responses
  - Assumes moderate to high latency
  - Complements ACLs
- More details can be found by searching for “CDMI DAC Extension”
- Need a simpler approach that replaces ACLs altogether
CDMI Access Control Delegation

1. Create Object (with DAC Metadata)
2. Encrypt Object value
3. CDMI PUT Request
4. Federate Object
5. CDMI PUT Response
6. Store Keys
7. Store Keys
8. Determine destination based on DAC Request URI
9. PUT DAC Request
10. Decrypt and Verify DAC Request
11. Make access control decision
12. Get Key
13. Receive Key
14. Create DAC Response
15. Determine if Intermediary
16. PUT DAC Response
17. Determine destination based on DAC Response URI
18. PUT DAC Response
19. Decrypt and Verify DAC Response
20. Extract Key
21. GET Response (Decrypt, send plaintext)

1. Request Object Contents (plaintext)
2. GET Request
3. Check for DAC Metadata
4. Create DAC Request
5. Extract Request URI
6. Determine if intermediary
7. PUT DAC Request
8. Determine destination based on DAC Request URI
9. PUT DAC Request
10. Decrypt and Verify DAC Request
11. Make access control decision
12. Get Key
13. Receive Key
14. Create DAC Response
15. Determine if Intermediary
16. PUT DAC Response
17. Determine destination based on DAC Response URI
18. PUT DAC Response
19. Decrypt and Verify DAC Response
20. Extract Key
21. GET Response (Decrypt, send plaintext)
CDMI ACL Delegation

CDMI Client → CDMI Server
CDMI GET + Credentials + Object

GET Response

CDMI Server → ACL Provider
AD Lookup + Credentials

AD Result + Principle

ACLs

ACL Provider → Active Directory
Access Control Request + Principle + Operation + Identifier

Access Control Response
CDMI ACL Delegation

- With CDMI Authentication Delegation, delegation is controlled by the CDMI domain
  - Each CDMI Object belongs to a domain
  - Domain configuration indicates if, and to which external system, authentication is delegated
- This approach can also work for Access Control Delegation
  - But Access Control Delegation and Authentication Delegation can be orthogonal, and can require different granularities
Option 1: Domain specifies an ACL delegation provider, similar to how an authentication delegation provider is configured:

<table>
<thead>
<tr>
<th>Metadata name</th>
<th>Type</th>
<th>Description</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdm_i_member_enabled</td>
<td>JSON string</td>
<td>If true, this field indicates that requests associated with this domain member are allowed. If false, all requests performed by this domain member shall result in an HTTP status code of 403 Forbidden.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>cdm_i_member_type</td>
<td>JSON string</td>
<td>This field indicates the type of member record. Values include &quot;user&quot; and &quot;delegation&quot;.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>cdm_i_delegation_URI</td>
<td>JSON string</td>
<td>This field contains the URI of an external identity resolution provider (such as LDAP or Active Directory) or the URI of a domain membership container object. External delegations are expressed in the form of ldap:// or ad://.</td>
<td>Mandatory</td>
</tr>
</tbody>
</table>
Option 2: A new ACE type field “DELEGATE”, which instructs the server to delegate to the system specified in the ACE identifier field.

This makes the delegation “sticky”, as it survives data movement and archiving, even if the domain changes.

This also allows fallback permissions when the delegation is unavailable or unsupported.

```json
"cdmi_acl": [{
    "acetype": "DELEGATE",
    "identifier": "https://127.0.0.1/delegate/",
    "aceflags": "OBJECT_INHERIT, CONTAINER_INHERIT",
    "acemask": "ALL_PERMS"
}, {
    "acetype": "DENY",
    "identifier": "EVERYONE@",
    "aceflags": "OBJECT_INHERIT, CONTAINER_INHERIT",
    "acemask": "ALL_PERMS"
}]
```
CDMI Access Control Delegation

- Simple proof of concept was implemented in a CDMI testing server to demonstrates that these concepts work.
- The following gaps were identified that would need to be addressed if standardized:
  - Support for fallback and load balancing across multiple delegation servers.
  - Using CDMI scopes for restricting delegations when specified in domains.
Live Demo

SNIA

CDMI

HTML 5

Open Policy Agent
This is a simple approach
This can be standardized into any system that uses ACLs
- S3 ACLs
- Google Cloud storage
- NFSv4, CIFS, etc
Provides portable Access control delegation

* Azure blobs only support ACLs at a container granularity
Questions?

- Contact Details:
  David Slik
d slik@netapp.com