AWS re:Invent

ANT281-P

Upgrading AWS Glue to use AWS Lake Formation permissions

Chanakya C. Damarla
Principal Product Manager
Amazon Web Services

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

Why the AWS Lake Formation security model?

Securing and accessing metadata

Securing and accessing data (Amazon Simple Storage Service [Amazon S3] locations)

Upgrading to use Lake Formation permissions

Conclusion

In the past, decision-making …

…revolved around the **enterprise data warehouse**



Data no longer fits



There is **more data** than people think

Data is **more diverse**

Data	Data platforms need to	
grows **>10x** every 5 years	live for **15** years	scale **1,000x**

* IDC, Data Age 2025: The Evolution of Data to Life-Critical: Don't Focus on Big Data, Focus on the Data That's Big, April 2017.

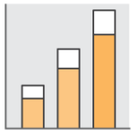
Broader workloads



Data Scientists



Business Users



Analysts



Applications

Machine learning

SQL analytics

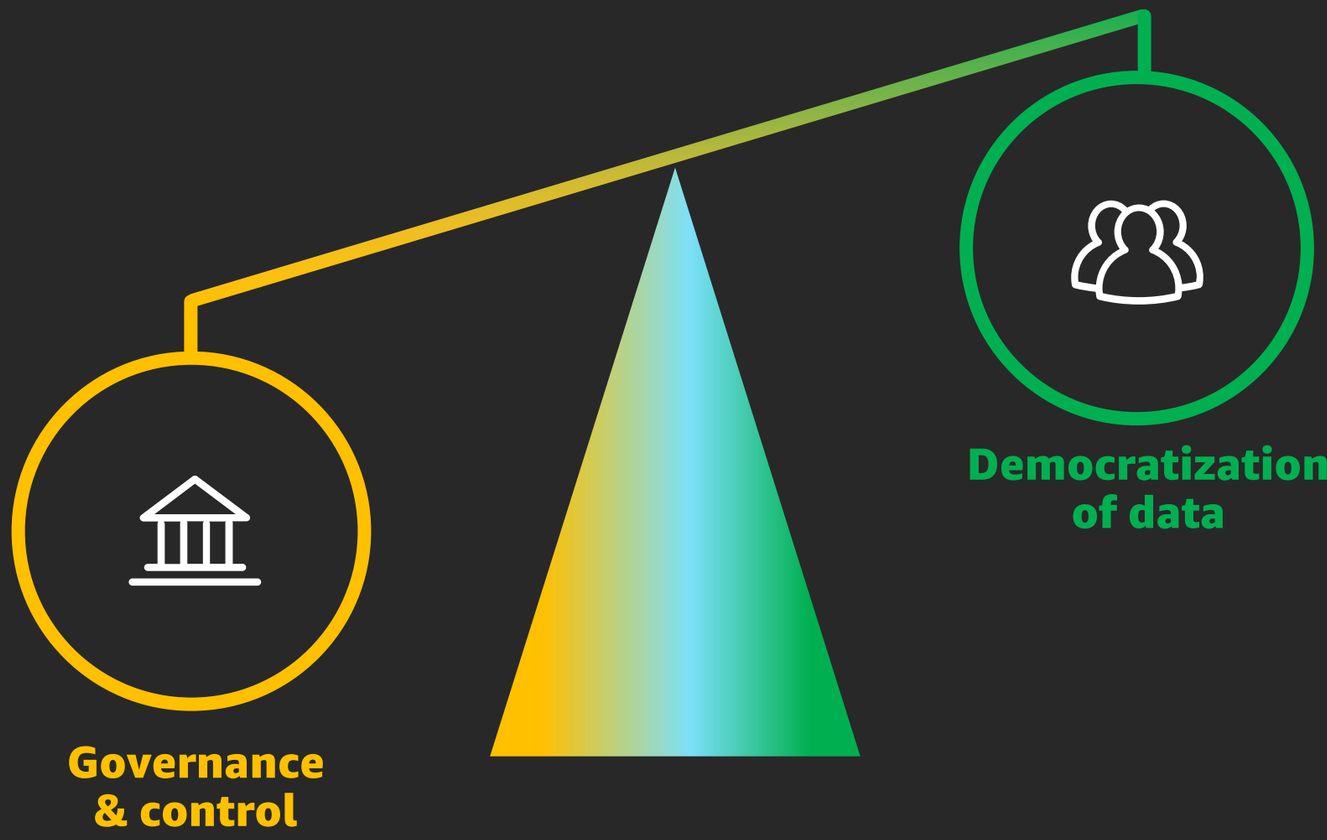
Scientific

Real-time, streaming

There are **more people** accessing data

who want to **analyze it in different ways**

More regulatory pressure



How do you democratize data access and

---

comply with regulations and **governance requirements?**

Data lake architectures are increasingly replacing DWHs





Data catalog



Use the best tool for the job

Pay-as-you-go pricing

Data catalog describes the metadata and provides a mapping:

- Logical model
  - Databases / tables / columns
- Physical model
  - How data is organized in Amazon S3

Data (structured/unstructured) resides in a cost-efficient and durable storage system

Data lake architectures need a central locus of control



To meet regulatory requirements, customers want to be able to set up permissions once and have those permissions apply across all the compute services and tools

Customers' frame of reference: DBMS-style permissions

Intuitive and simple grant/revoke model

```
GRANT { { SELECT | INSERT | UPDATE | DELETE | TRUNCATE | REFERENCES | TRIGGER }
    [, ...] | ALL [ PRIVILEGES ] }
    ON { [ TABLE ] table_name [, ...] | ALL TABLES IN SCHEMA schema_name [, ...] }
    TO { [ GROUP ] role_name | PUBLIC } [, ...] [ WITH GRANT OPTION ]

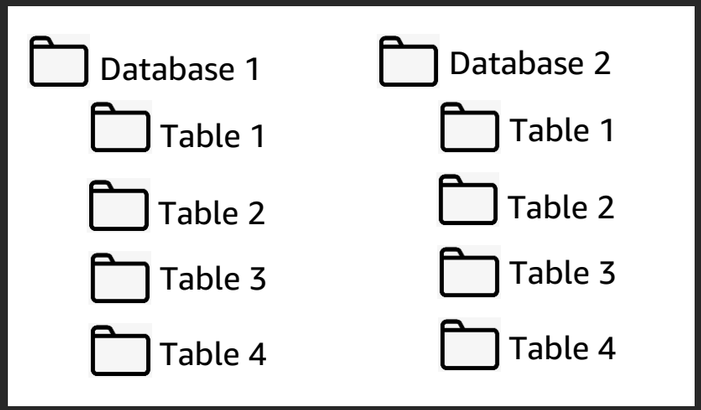
REVOKE [ GRANT OPTION FOR ] { { SELECT | INSERT | UPDATE | DELETE | TRUNCATE | REFERENCES
    | TRIGGER } [, ...] | ALL [ PRIVILEGES ] }
    ON { [ TABLE ] table_name [, ...] | ALL TABLES IN SCHEMA schema_name [, ...] }
    FROM { [ GROUP ] role_name | PUBLIC } [, ...] [ CASCADE | RESTRICT ]
```



Grant/revoke access

- Get necessary permissions
- Metadata | Data & storage

DBMSs manage file system resources



Permissions required across the entire stack





Data catalog



Separate security policies for:

- Compute services
- Data catalog
- Amazon S3 access

Difficult to:

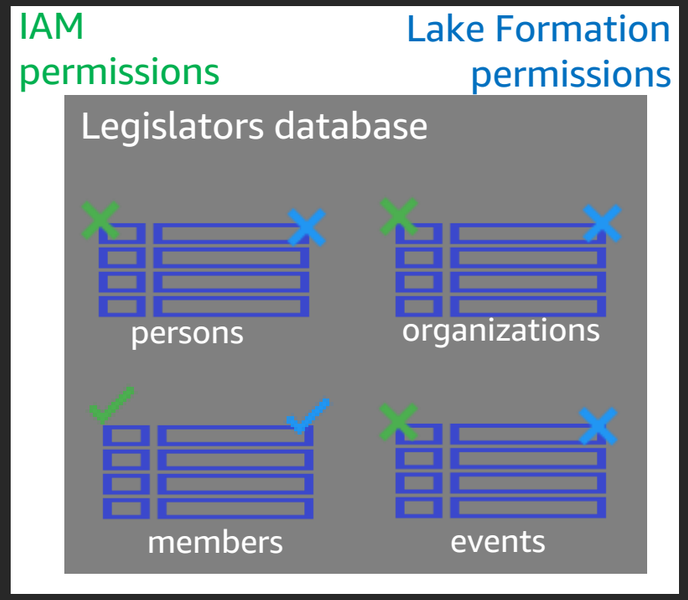
- Set up compliant policies
- Manage change
- Audit data access
- Verify compliance

Simplified permissions model is required

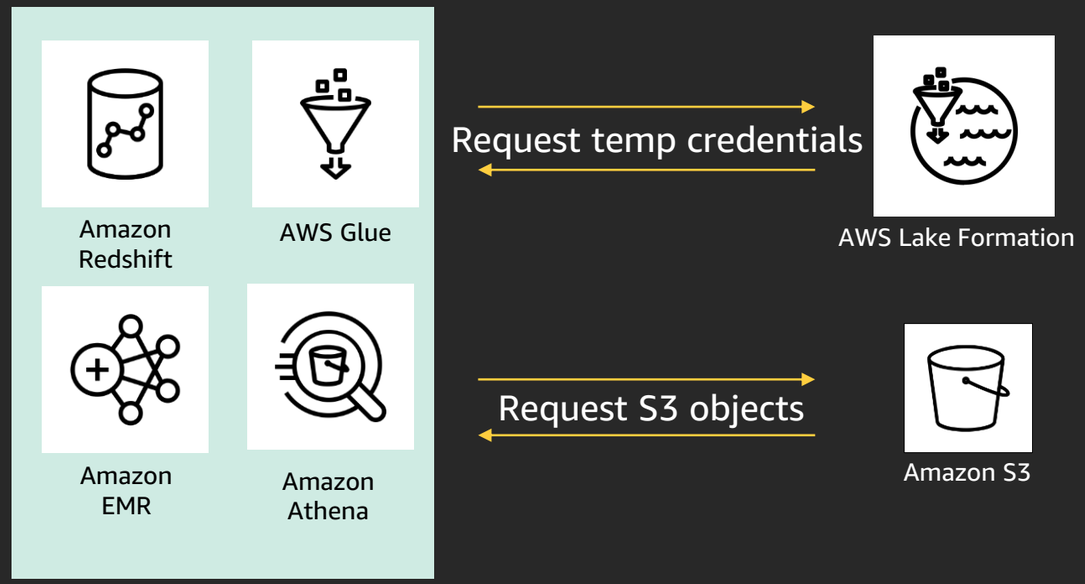
AWS Lake Formation security model

Two primary levers that customers can use to implement DBMS-style permissions on **metadata** and **data**

New permissions



Temp credential vending



AWS Lake Formation security model design goals

Single data catalog

No forced data or metadata state migrations

Incremental adoption/migration

Simple intuitive permissions model

Permissions consistently enforced in all tools

No business disruption: Don’t stop the data catalog

Agenda

Why the AWS Lake Formation security model?

Securing and accessing metadata

Securing and accessing data (Amazon S3 locations)

Upgrading to use Lake Formation permissions

Conclusion

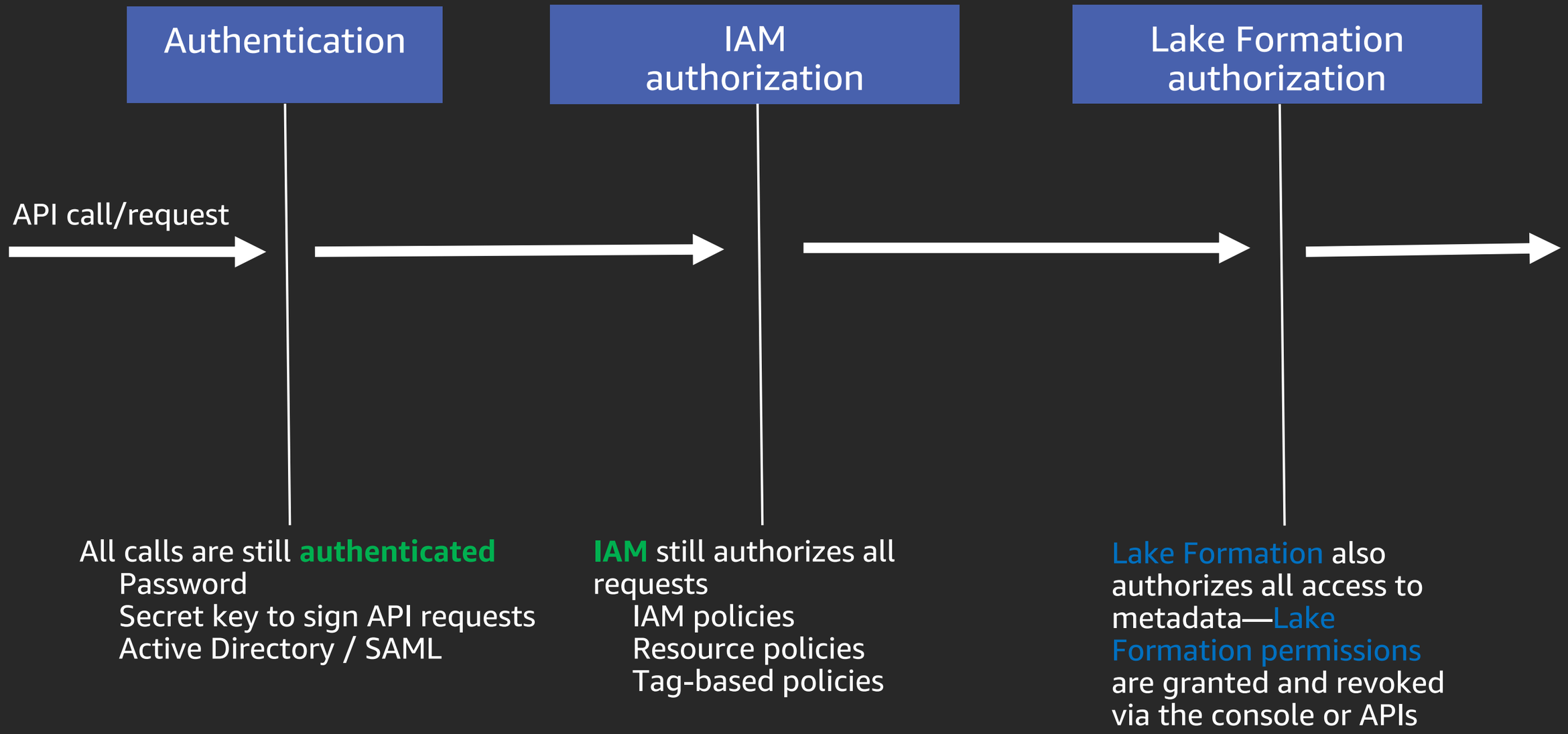
Authentication and authorization with IAM



Authentication is used to identify a principal
- Password
- Secret key to sign API requests
- Active directory / SAML

IAM authorization is used to determine whether the principal has access to the requested API/resource
- IAM policies
- Resource policies
- Tag-based policies

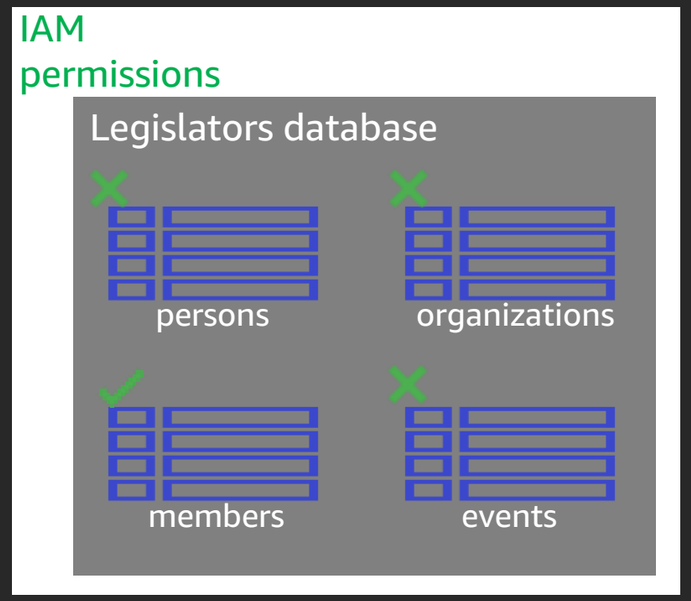
Authentication & authorization with Lake Formation



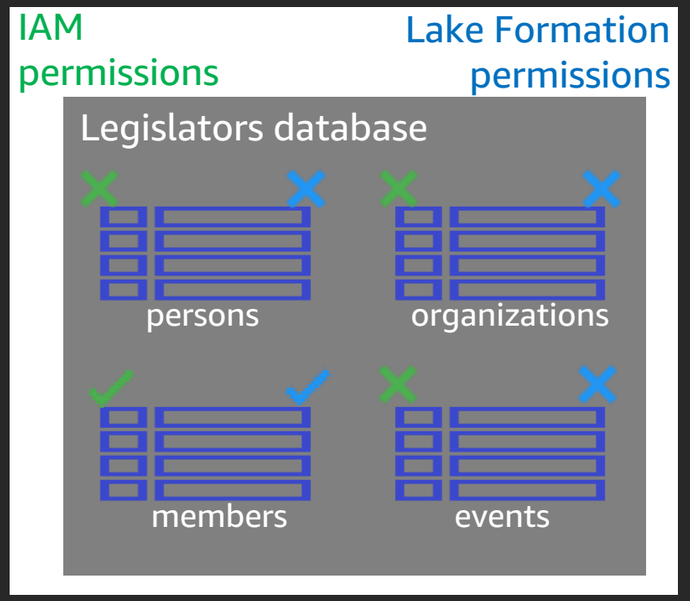
Lake Formation permissions

Lake Formation & **AWS Glue** use the same **data catalog**

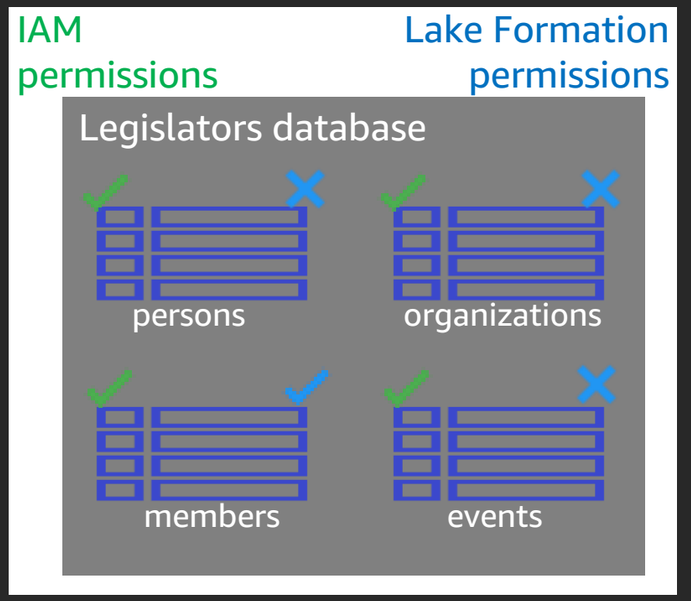
Both **IAM** **and** **Lake Formation** permissions authorize access to metadata



IAM permissions

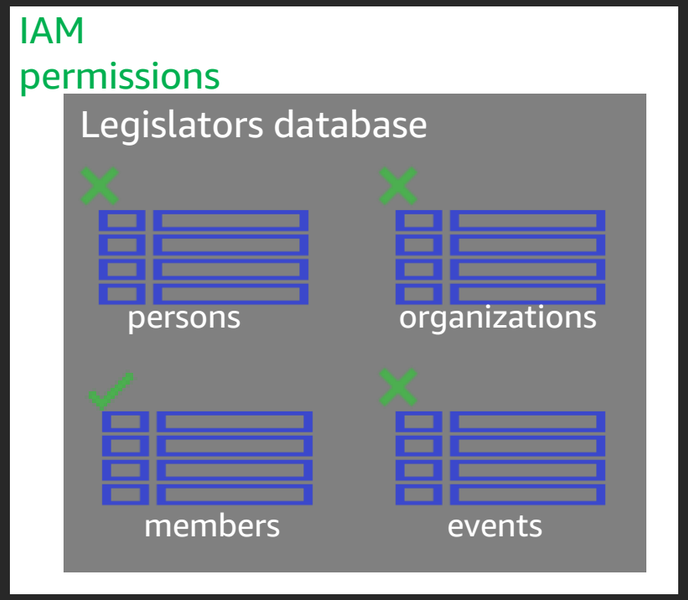


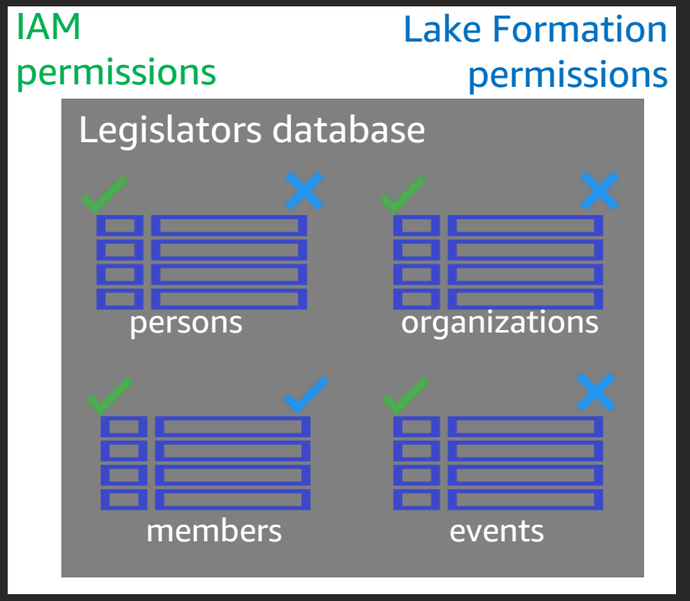
IAM & Lake Formation permissions authorize access



Recommended usage:
Coarse-grained IAM permissions
Fine-grained Lake Formation permissions

What coarse-grained IAM policies look like






```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
         "Effect": "Allow",
         "Action": [
            "glue:GetTables", "glue:SearchTables", "glue:GetTable",
                  "glue:GetDatabase", "glue:GetDatabases"
         ],
         "Resource": [
            "arn:aws:glue:us-east-1:<account>:database/legislators",
            "arn:aws:glue:us-east-1:<account>:table/legislators/*",
            "arn:aws:glue:us-east-1:<account>:catalog"
         ] } ]
}
```

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
         "Effect": "Allow",
         "Action": [
            "glue:GetTables", "glue:SearchTables", "glue:GetTable”,
                  "glue:GetDatabase”, "glue:GetDatabases”
         ],
         "Resource": "arn:aws:glue:us-east-1:<account>:*"
      }
   ]
}
```

Lake Formation security policies

Lake Formation security policies: A triple

Lake Formation principal: IAM principal or Active Directory user/groups

Resource: Locations, catalog, database, table, columns

Permissions: Resource-specific

Locations	Catalog	Database	Table	Column
data_location_access	create_database	create_table	alter	select*
		alter	drop	
		drop	select*	
			insert*	
			delete*	

Inferring required permissions

Lake Formation simplifies permissions management

In some cases, required permissions may be inferred

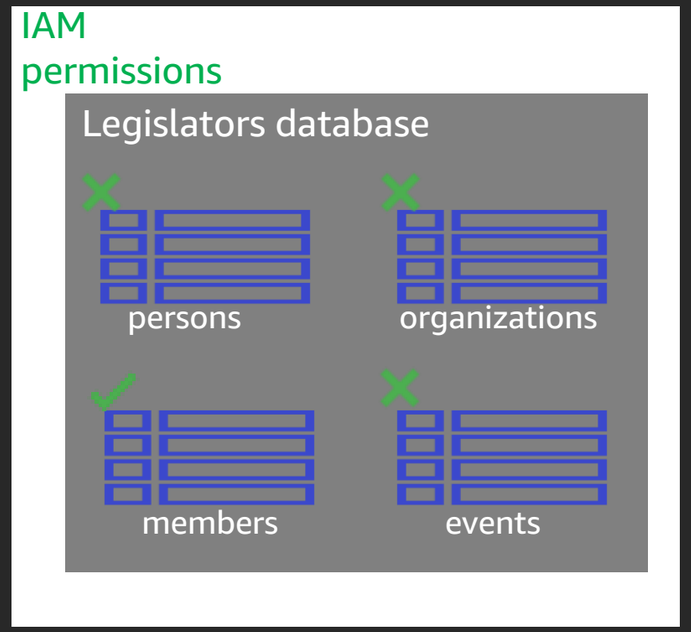
Having access to select from a table => user should be able to view table metadata
=> user should be able to list the database
=> user should be able to read data location

Administrator permissions	Creator permissions	Hierarchy permissions	Location permissions	Scope of operation permissions
Data lake admins can list all locations/ databases/tables.	Creators of databases/ tables have all permissions with grant.	Hierarchical metadata: catalog < db < table	Typically, to associate a table with a database location, permissions are required.	Data access perms mean you should be able to retrieve temporary credentials. select => read insert / delete => write
They can also grant any permission on any location/database/ table.	They can also list created DBs/tables.	Permissions on an object mean you should be able to list the object's parents.	create_table perm on a database with a specified location => you can create a table at any child location.	Some operations, like insert/delete, can only be granted on all columns. You can't insert partial rows.

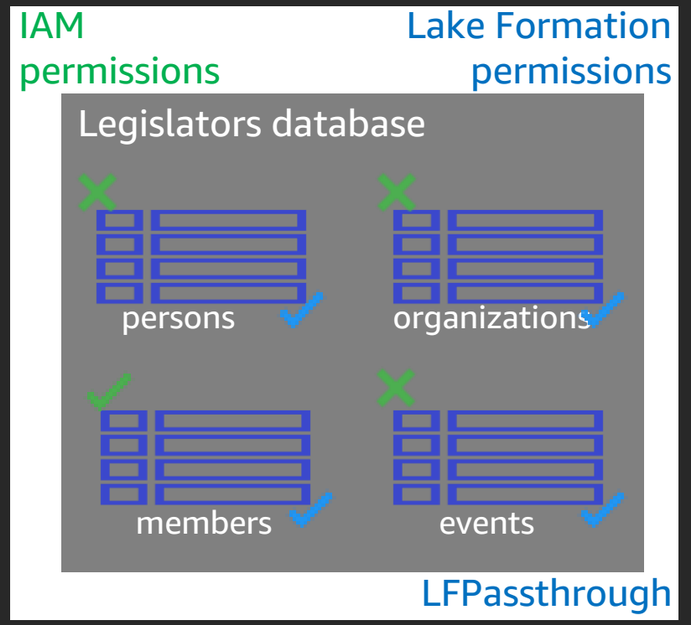
Lake Formation is backward compatible

IAM and **Lake Formation** permissions secure access to the data catalog

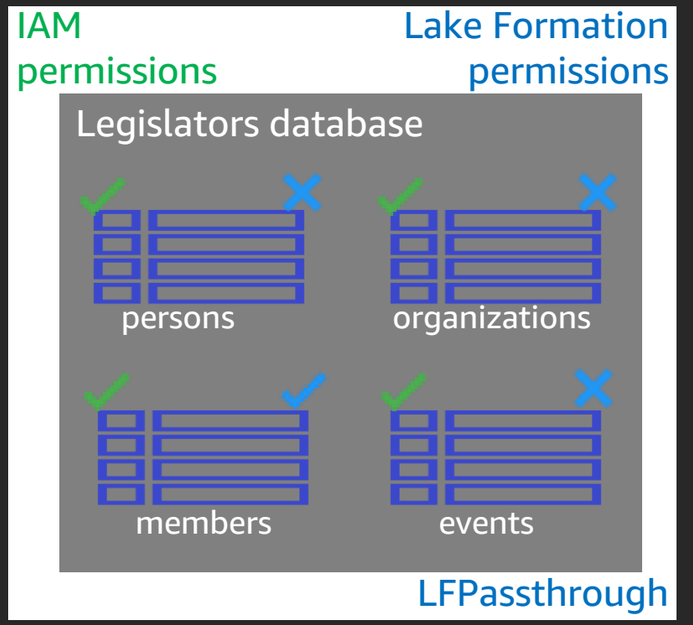
For backward compatibility, **Lake Formation** passes through **IAM** permissions



IAM permissions



LFPassthrough = S2I
"Super" to IAMAllowedPrincipals



Upgrade by
1. Granting Lake Formation permissions
2. Revoking LFPassthrough
3. Utilizing coarse-grained IAM permissions

Agenda

Why the AWS Lake Formation security model?

Securing and accessing metadata

Securing and accessing data (Amazon S3 locations)

Upgrading to use Lake Formation permissions

Conclusion

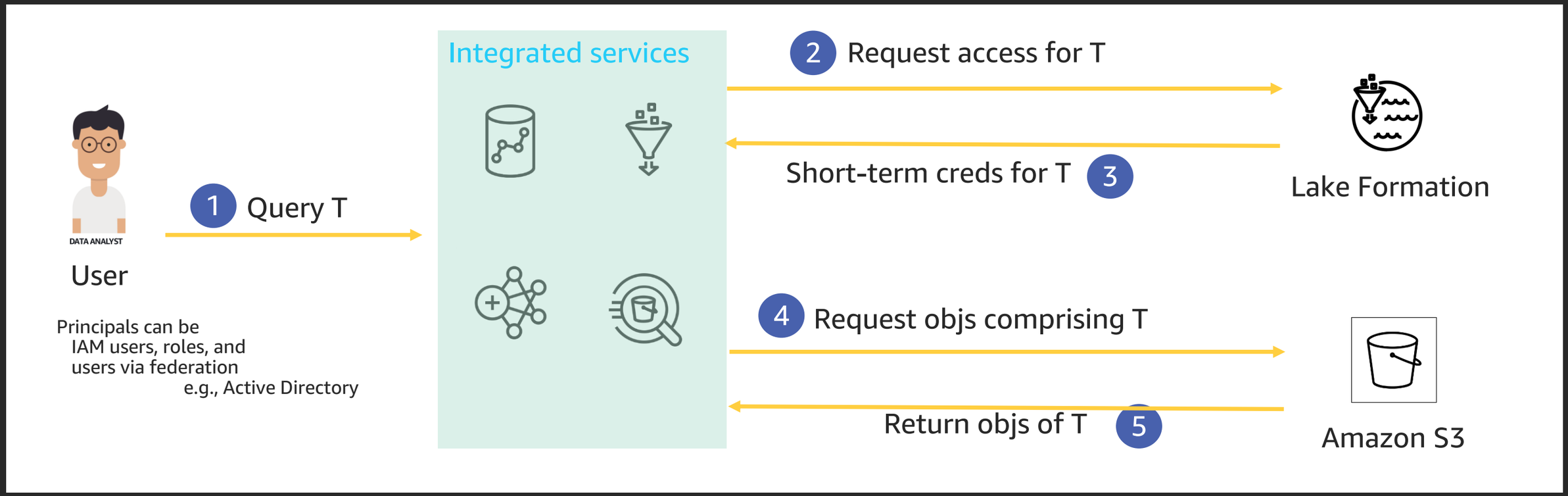
Unify metadata/data access with credential vending

Lake Formation manages access to registered locations

Unifies data access permissions with metadata access permissions

Vends temporary credentials

Integrated services: Request and use these credentials on behalf of principals



Expected usage

IAM permissions

Legislators database



Upgrade by
1. Granting Lake Formation permissions
2. Revoking LFPassthrough
3. Utilizing coarse-grained IAM permissions

IAM permissions

Lake Formation permissions

Legislators database



LFPassthrough

IAM permissions

Amazon S3



4. Enabling GetDataAccess / register location

IAM permissions

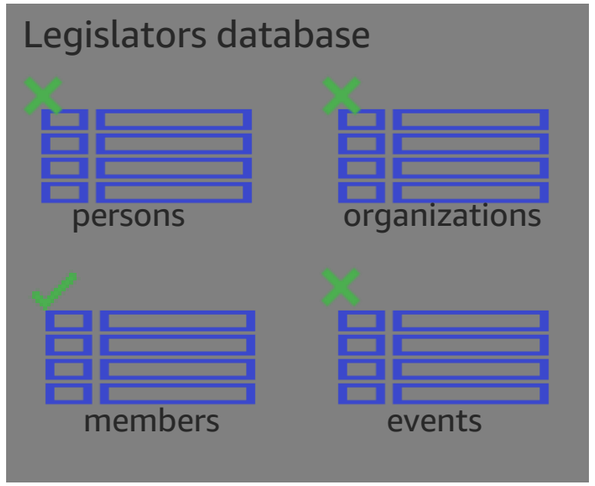
Register location

Amazon S3



Optionally lock down S3 permissions

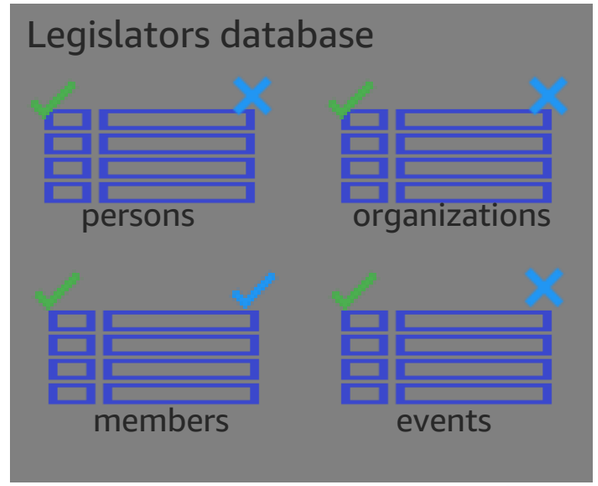
IAM permissions



Upgrade by
1. Granting Lake Formation permissions
2. Revoking LFPassthrough
3. Utilizing coarse-grained IAM permissions

IAM permissions

Lake Formation permissions



LFPassthrough

IAM permissions

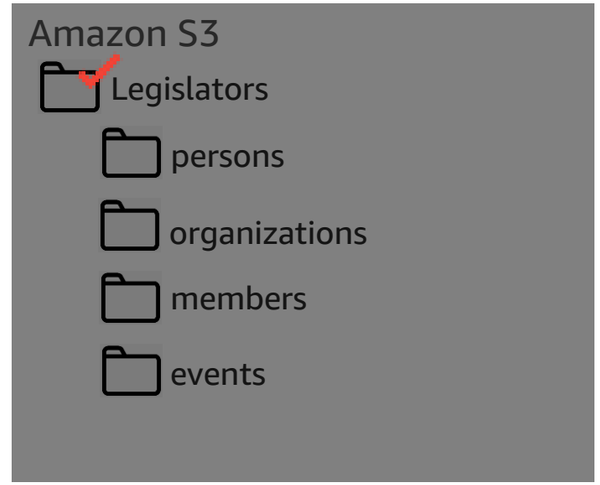


4. Enabling GetDataAccess / register location

5. Locking down S3 permissions
Optional, as it may break console/service access

IAM permissions

Register location



Data access APIs…

IAM permissions

Legislators database

persons

organizations

members

events

IAM permissions

Register location

Amazon S3

Legislators

persons

organizations

members

events



User



Amazon Athena

1 select * from legislators.members limit 10;

2 GetTable(legislators.members)

Schema is returned
Includes a bool if the location is registered
List of authorized data access columns



Lake Formation

3 GetTemporaryTableCredentials(table)

Short-term creds for table locs

(User has access to database1.table3)
(select | insert | delete **not** alter | drop)

4 Request objs from loc

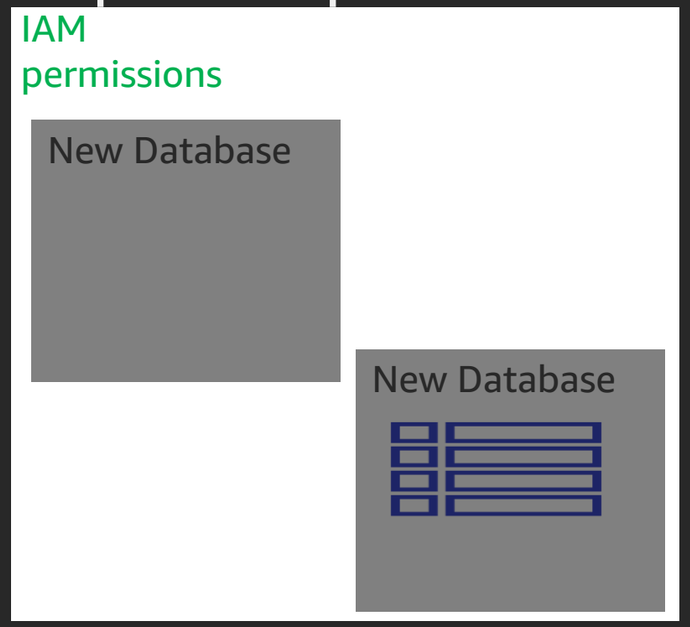
Return objs from loc

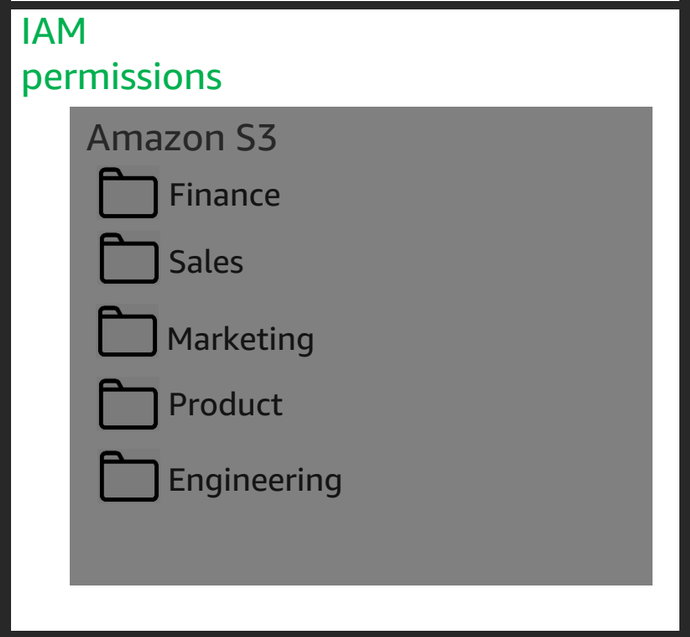


Amazon S3

5 Run query / filter columns / return results

Explicit permissions for database/table locations





When creating databases/tables, it's possible to associate locations

- The location is where the underlying data associated with the table resides
- Required for tables
- Optional for databases

Until now, the discussed permissions were on metadata

- create_database / create_table lets you create databases/table
- But where can you point them?
- Can anyone create a table in the Finance bucket?

Location permissions are explicitly specified

Agenda

Why the AWS Lake Formation security model?

Securing and accessing metadata

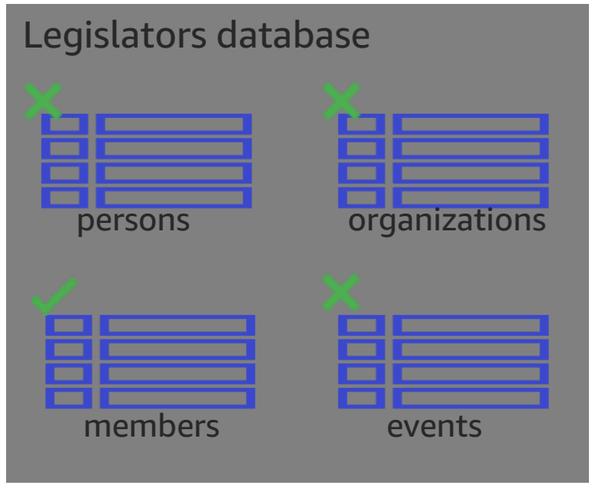
Securing and accessing data (Amazon S3 locations)

Upgrading to use Lake Formation permissions

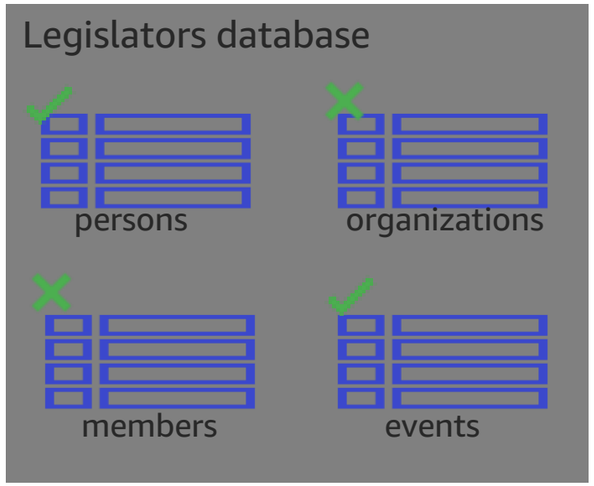
Conclusion

Starting point

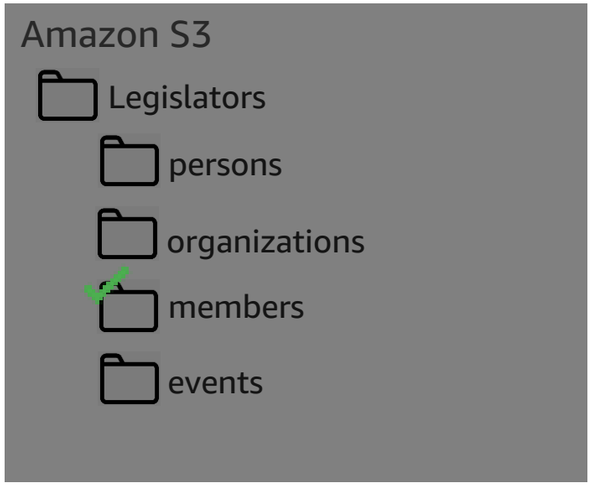
Jack: IAM permissions



Jill: IAM permissions



Jack: IAM permissions



Jill: IAM permissions



This will be the starting point for this demonstration.

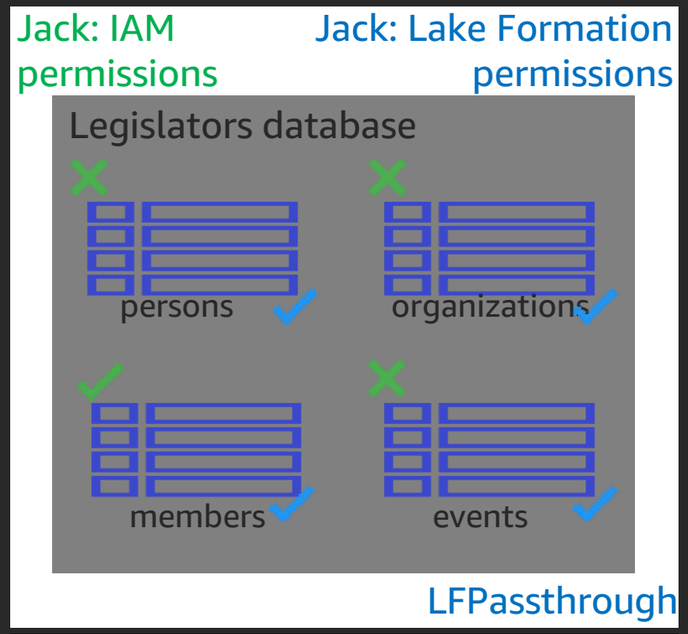
Jack has access to:

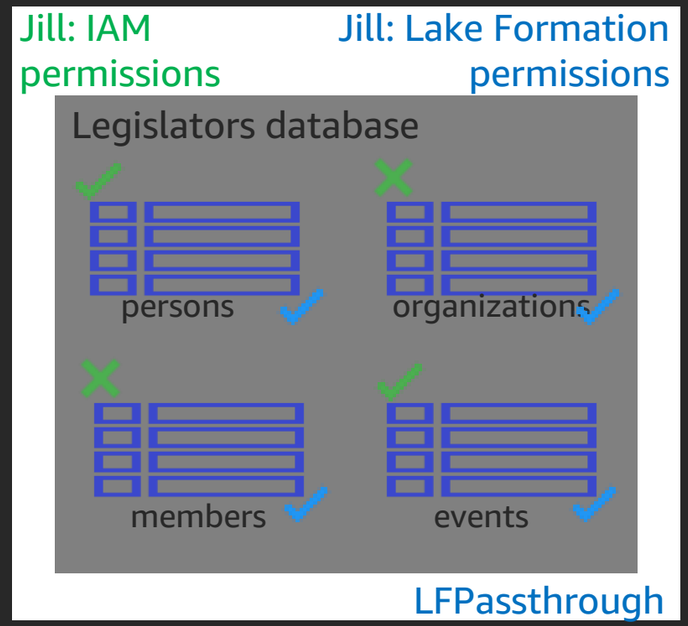
- MD: Legislators/members
- S3: members_json/*

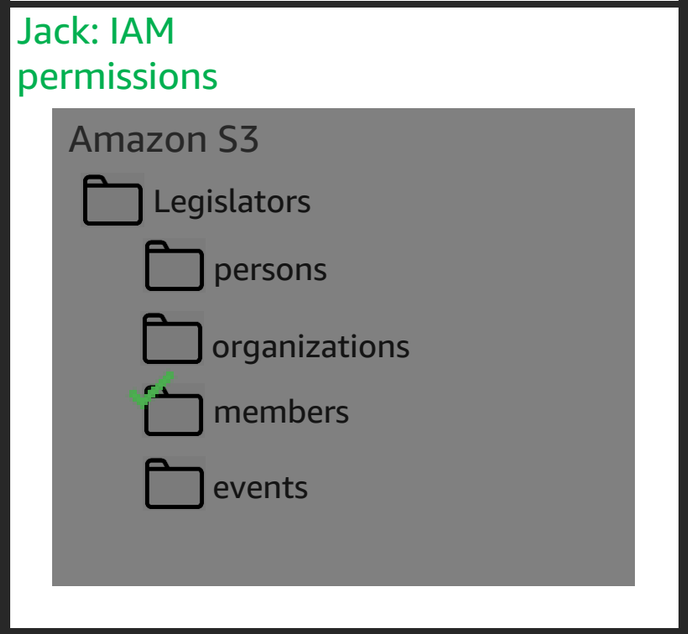
Jill has access to:

- MD: Legislators/persons & /events
- S3: persons_json/* & events_json/*

Lake Formation launch introduced new permissions









When Lake Formation launched, for backward compatibility, Lake Formation implemented LFPassthrough.

Super to IAMAllowedPrincipals (S2I) on all existing metadata objects.

The expression granting access is:
IAMPerms **and** (S2I **or** LFPerms)

always true

Jack still has access to:
MD: Legislators/members
S3: members_json/*

Jill still has access to:
MD: Legislators/persons & /events
S3: persons_json/* & events_json/*

Upgrading to use Lake Formation permissions

1. Grant Lake Formation permissions

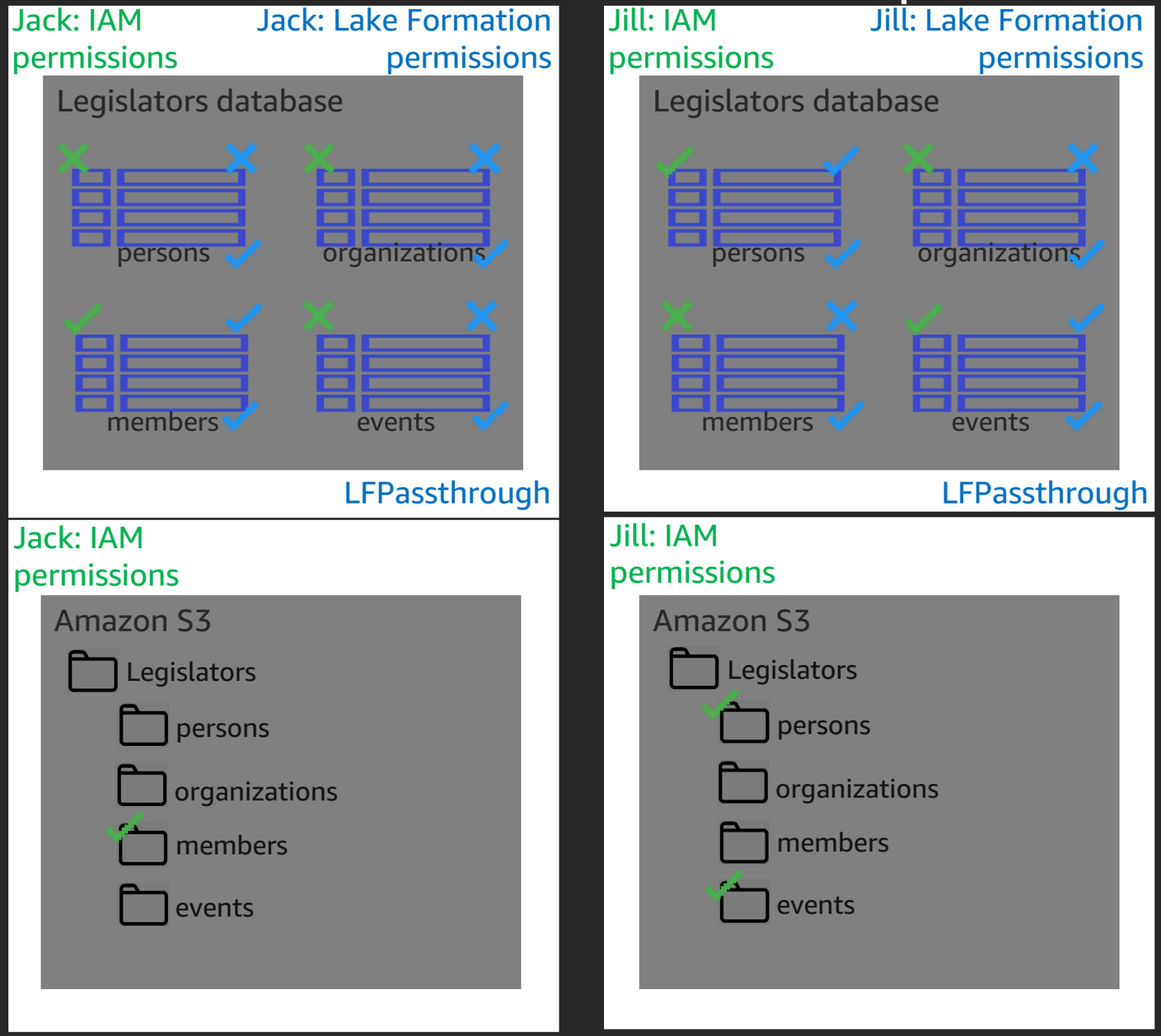
2. Revoke LFPassthrough

3. Utilize coarse-grained IAM permissions

4. Enable GetDataAccess / register location

5. Optionally lock down S3 permissions

1. Grant Lake Formation permissions



Expected usage is to have:
- Coarse-grained IAM permissions
- Fine-grained Lake Formation permissions

Understand existing permissions:
- Look at IAM policies
- Review InsufficientLFPermissions

Set up fine-grained Lake Formation permissions:

The expression granting access is:

IAMPerms **and** (S2I **or** LFPerms)

always true

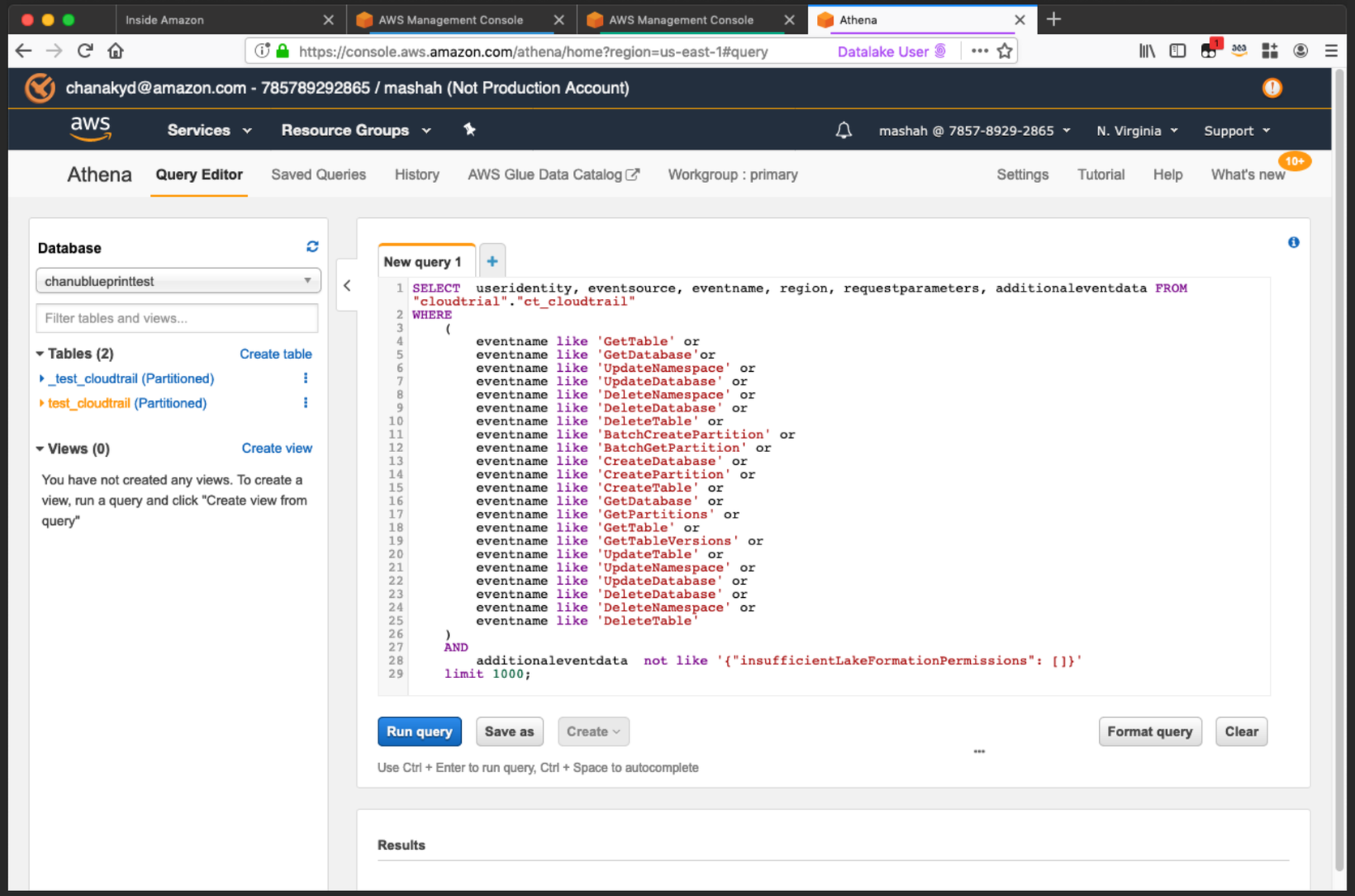
Jack still has access to:
- MD: Legislators/members
- S3: members_json/*

Jill still has access to:
- MD: Legislators/persons & /events
- S3: persons_json/* & events_json/*

Find insufficient Lake Formation permissions

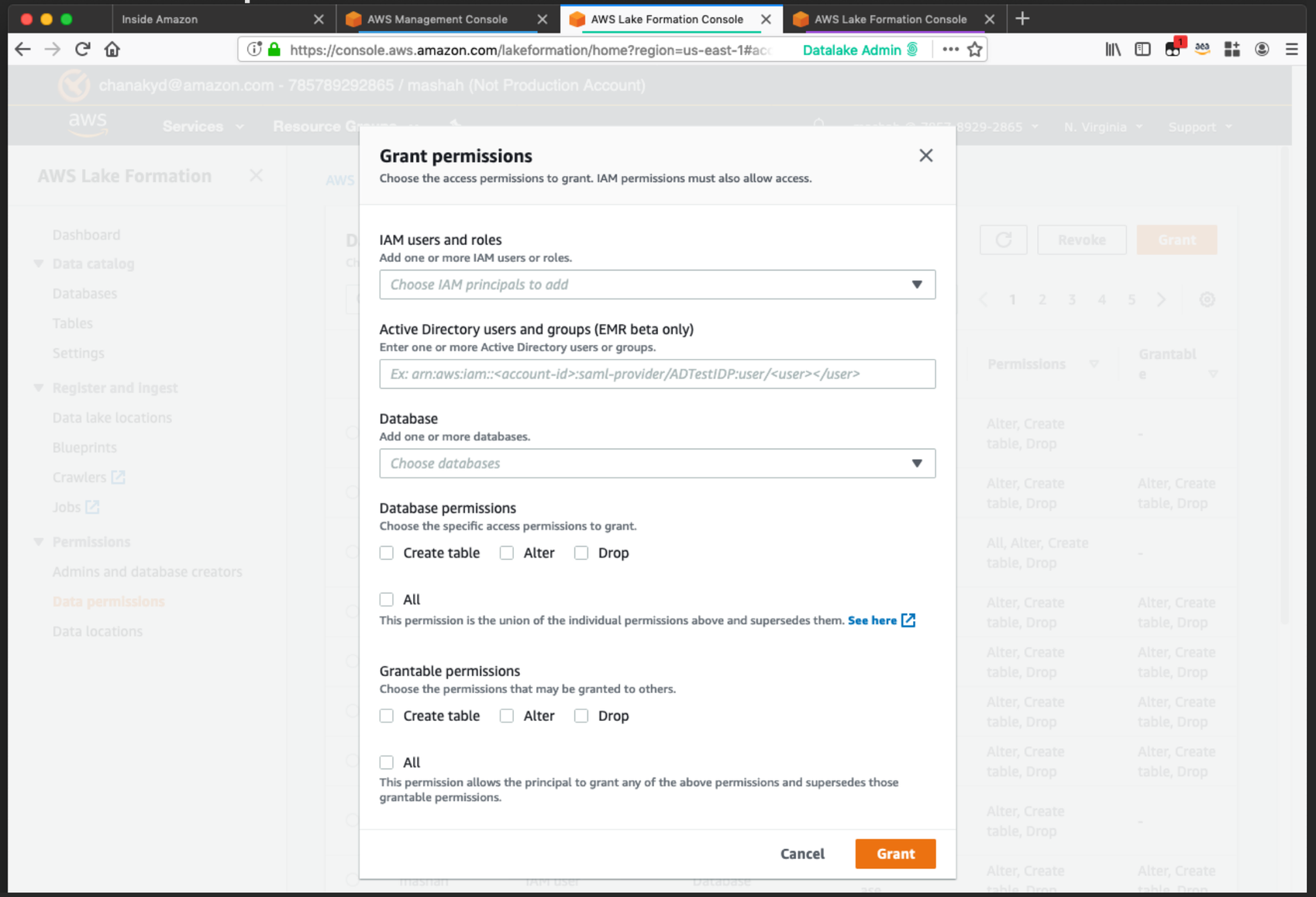
Query AWS CloudTrail logs to identify which metadata request APIs would have failed because of insufficient Lake Formation permissions.

Use a sufficiently long timeframe to capture all/most usage.



Grant Lake Formation permissions

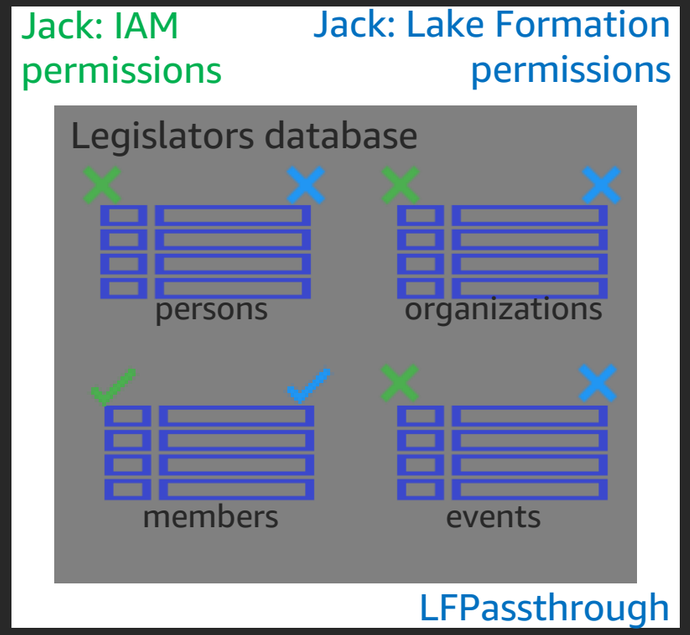
Grant Lake Formation permissions using the Lake Formation console or APIs.

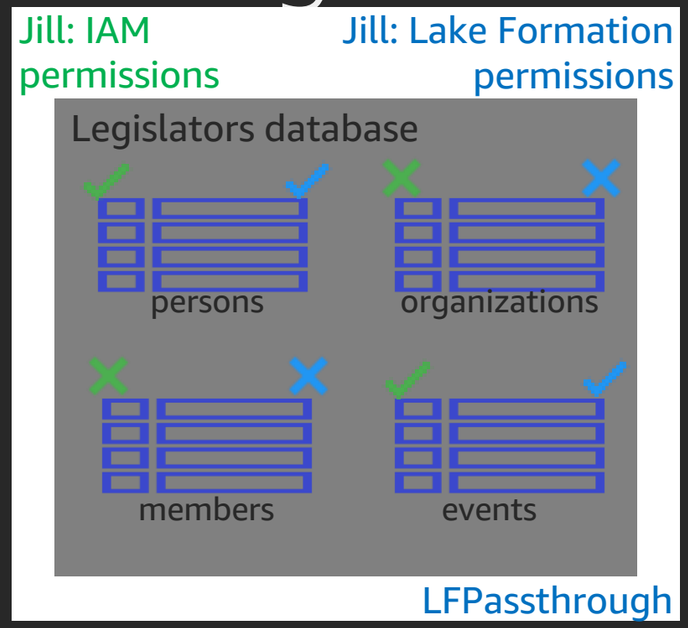


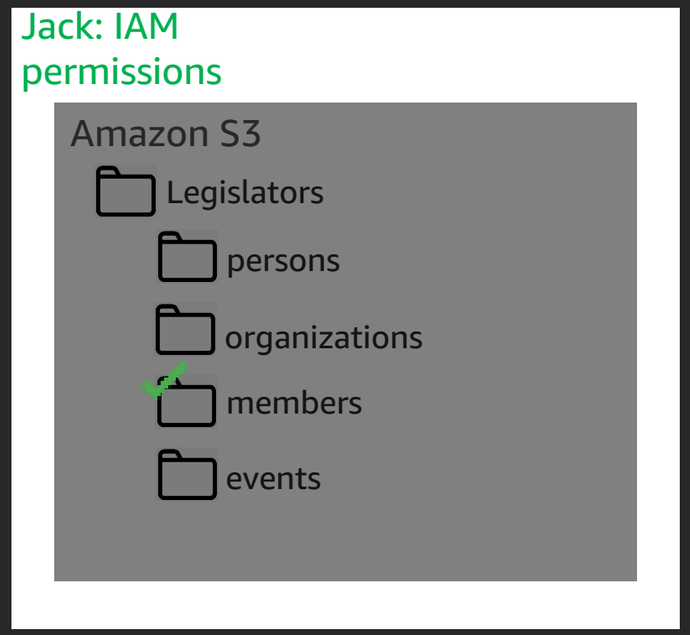
Upgrading to use Lake Formation permissions

1. Grant Lake Formation permissions
2. Revoke LFPassthrough
3. Utilize coarse-grained IAM permissions
4. Enable GetDataAccess / register location
5. Optionally lock down S3 permissions

2. Revoke LFPassthrough









Fine-grained Lake Formation permissions set up, but they are not being enforced, because S2I is on.

Verify no InsufficientLFPermissions, or else you may break existing access.

Revoke LFPassthrough (S2I)

The expression granting access is:
IAMPerms **and** (~~S2I~~ **or** LFPerms)

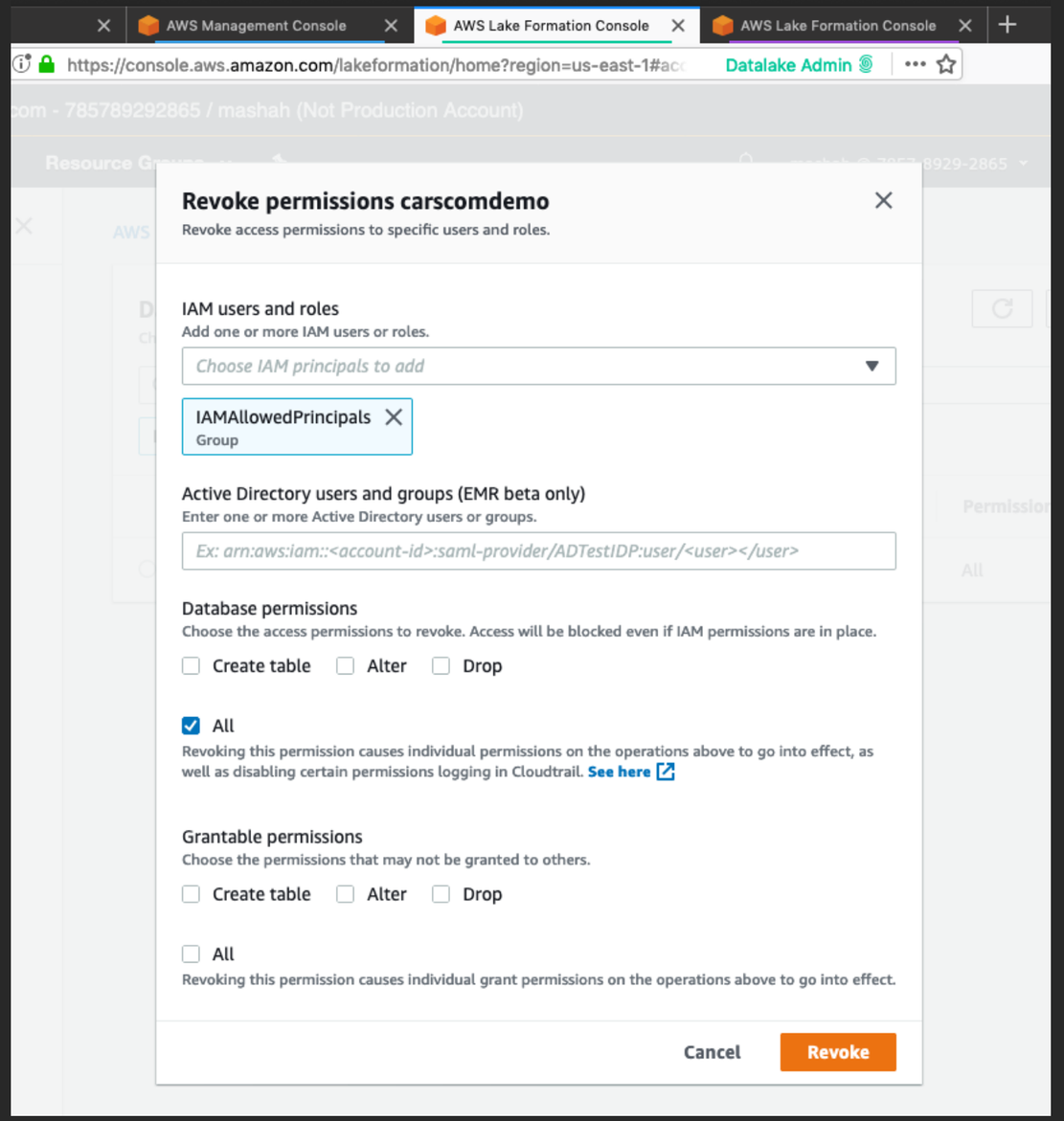
not always true

User1 still has access to:
MD: Legislators/members
S3: members_json/*

User2 still has access to:
MD: Legislators/persons & /events
S3: persons_json/* & events_json/*

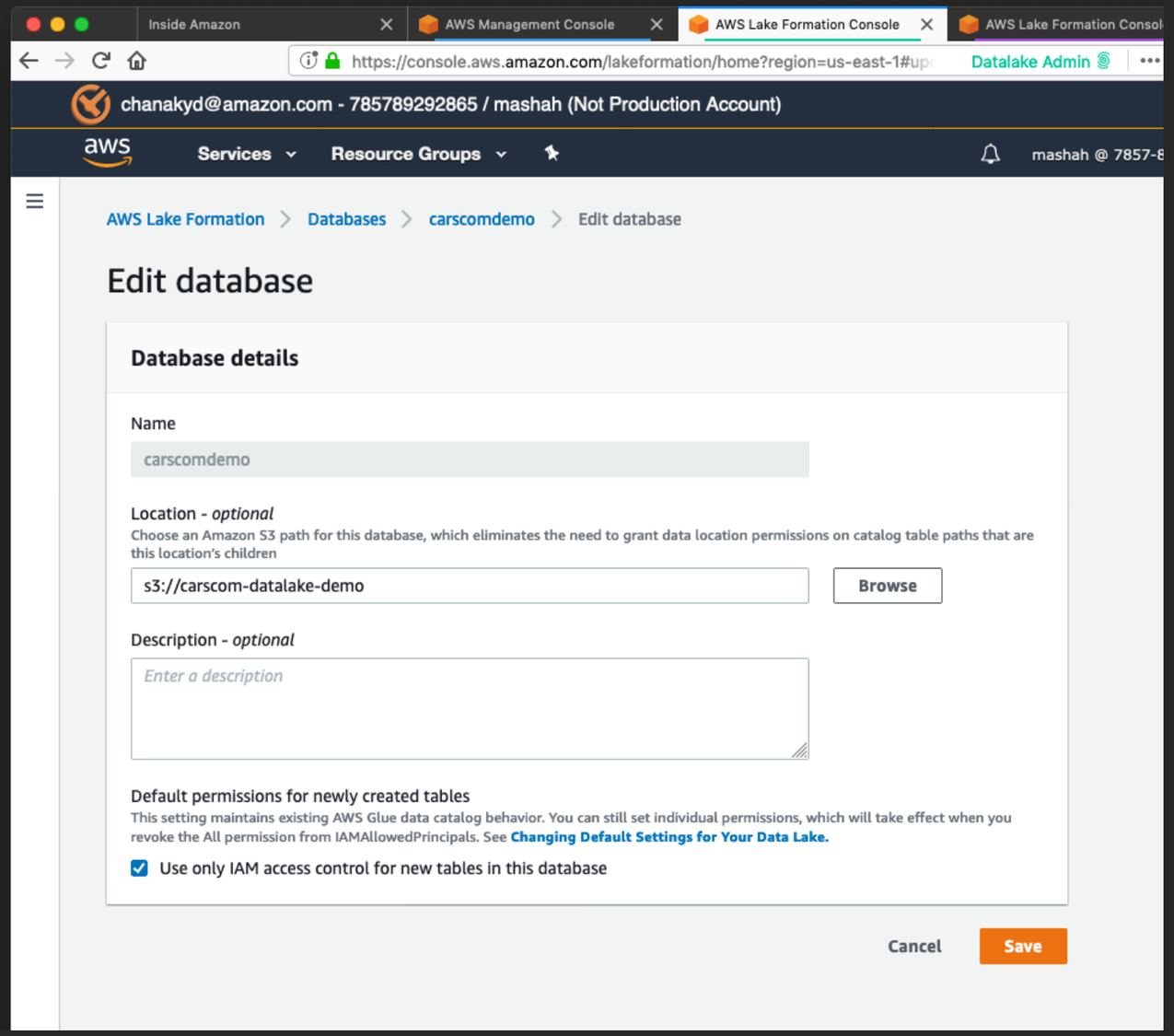
Revoke LFPassthrough on databases & tables

Revoke the “Super” permission from IAMAllowedPrincipals.



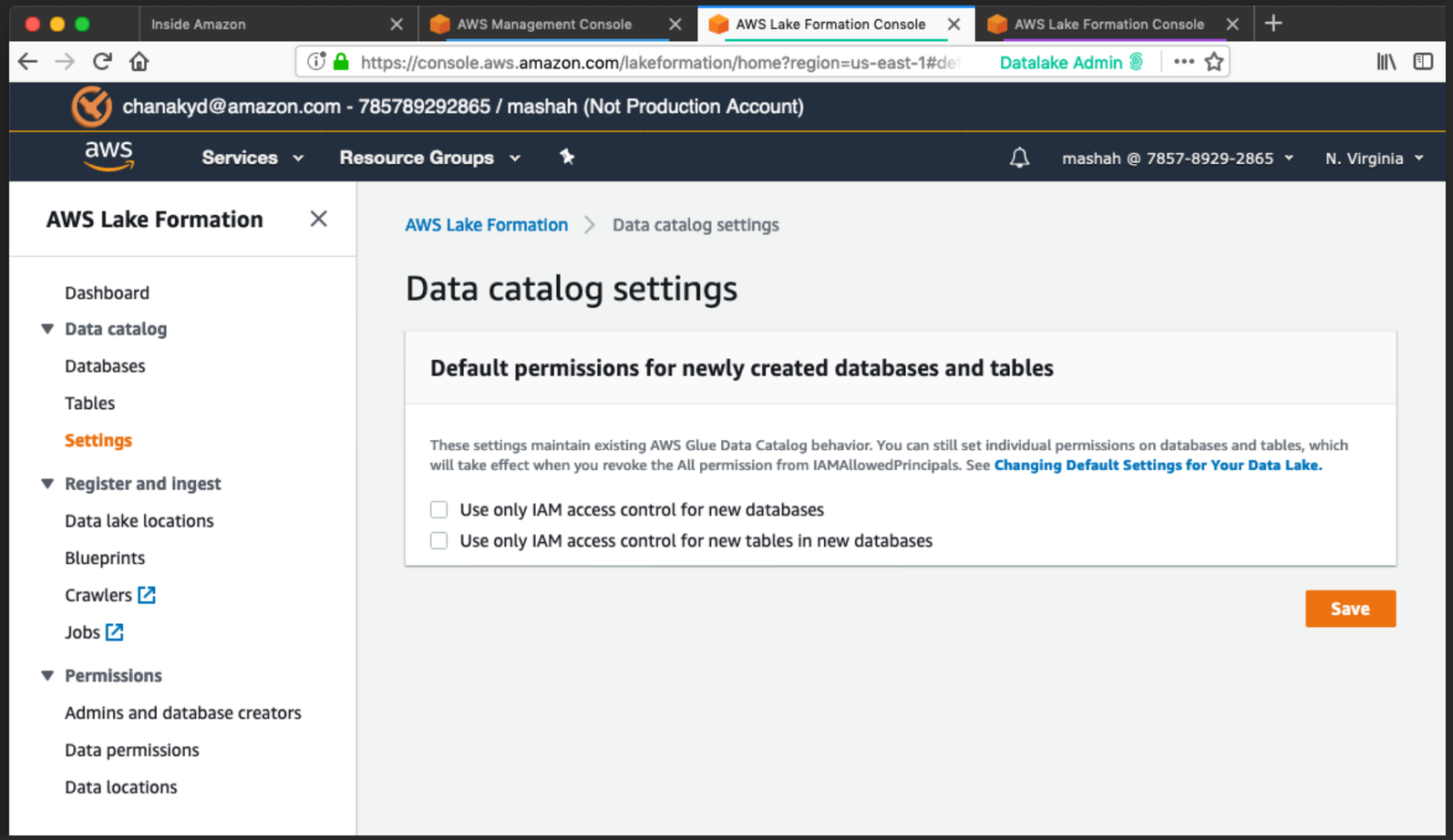
Revoke LFPassthrough on new tables

Remove setting so new tables will not have S2I enabled on them.



Revoke LFPassthrough on catalog

Change the default settings on the data catalog.



Upgrading to use Lake Formation permissions

1. Grant Lake Formation permissions

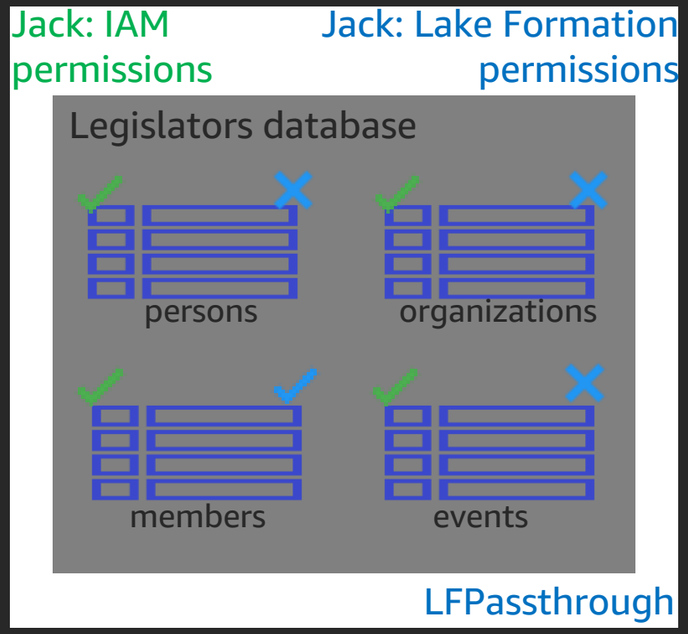
2. Revoke LFPassthrough

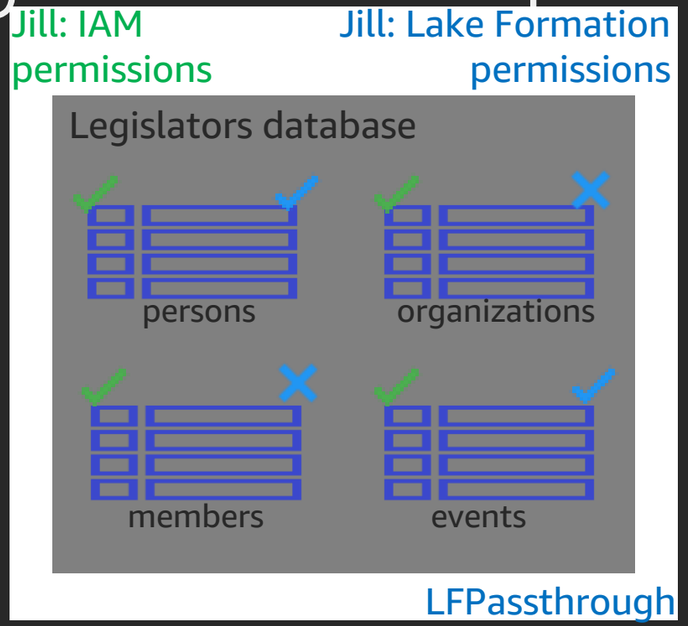
3. Utilize coarse-grained IAM permissions

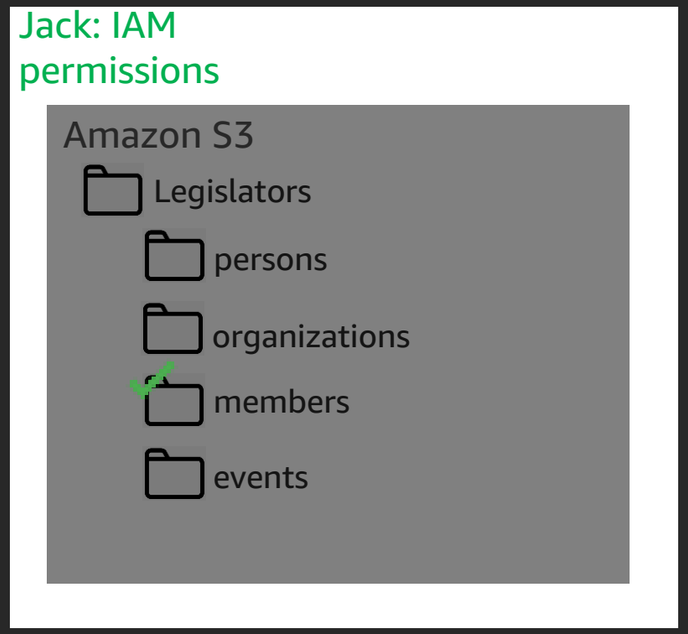
4. Enable GetDataAccess / register location

5. Optionally lock down S3 permissions

3. Utilize coarse-grained IAM permissions









Fine-grained Lake Formation permissions are set up, so we can now use coarse-grained IAM permissions.

The expression granting access is:

IAMPerms **and** (~~S2I~~ **or** LFPerms)

always true

Jack still has access to:

MD: Legislators/members

S3: members_json/*

Jill still has access to:

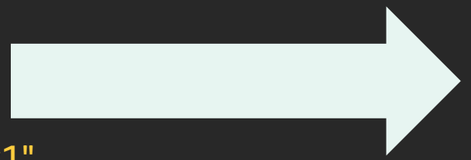
MD: Legislators/persons & /events

S3: persons_json/* & events_json/*

Utilize coarse-grained IAM permissions

Replace metadata access policies with this policy.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
         "Effect": "Allow",
         "Action": [
            "glue:GetTables", "glue:SearchTables", "glue:GetTable"
         ],
         "Resource": [
            "arn:aws:glue:us-west-2:acct:database/legislators_target_v1",
            "arn:aws:glue:us-west-2:acct:table/legislators_target_v1/organizations_json",
            "arn:aws:glue:us-west-2:acct:table/legislators_target_v1/memberships_json",
            "arn:aws:glue:us-west-2:acct:catalog"
         ]
      }
   ]
}
```



```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
         "Effect": "Allow",
         "Action": [
            "glue:GetTables",
            "glue:SearchTables",
            "glue:GetTable",
            "glue:GetDatabases",
            "glue:GetDatabase"
         ],
         "Resource": [
            "arn:aws:glue:us-west-2:785789292865:*"
         ]
      }
   ]
}
```

Upgrading to use Lake Formation permissions

1. Grant Lake Formation permissions

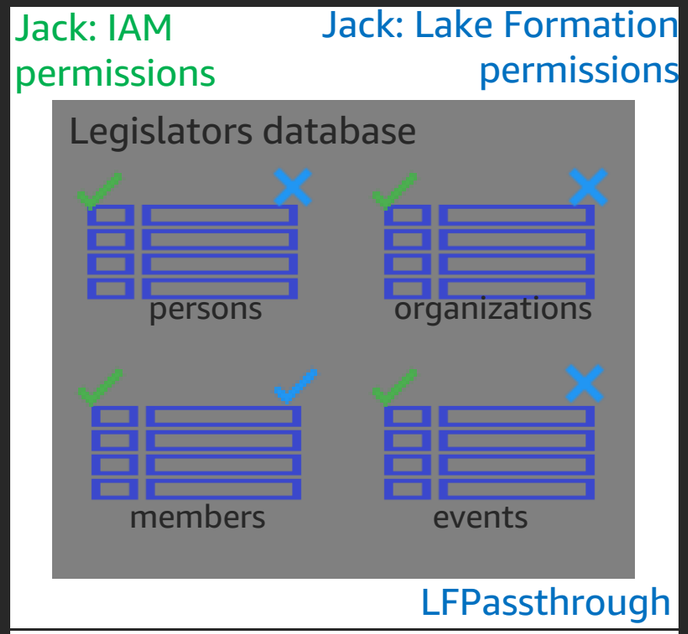
2. Revoke LFPassthrough

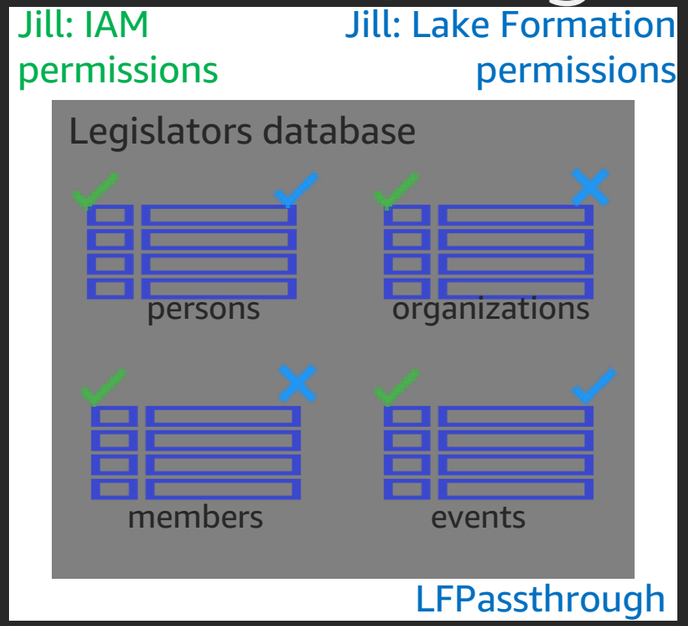
3. Utilize coarse-grained IAM permissions

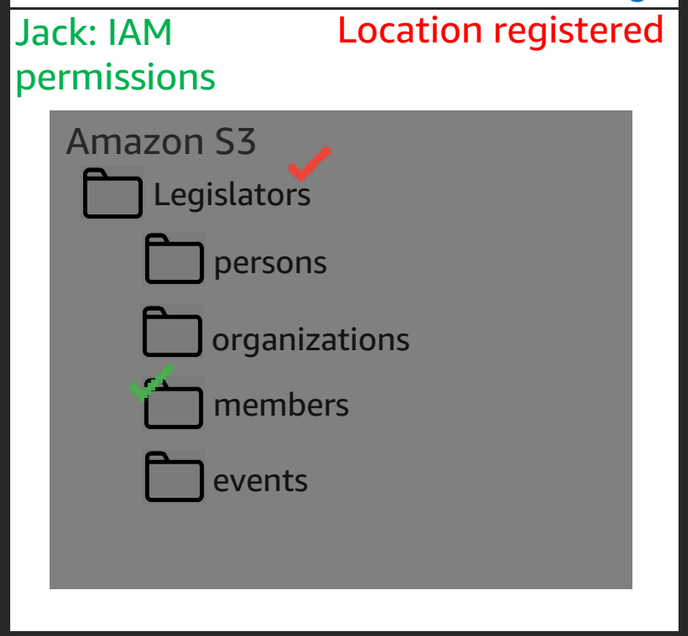
4. Enable GetDataAccess / register location

5. Optionally lock down S3 permissions

4. Enable GetDataAccess / register location









Unify metadata access with data access by registering locations so Lake Formation can vend credentials.

Audit resources that point to locations you are registering.

When you register a location, integrated services will use Lake Formation credentials to access location. So it is possible to break existing usage.

Users must have **GetDataAccess** permissions to use the temporary credentials.

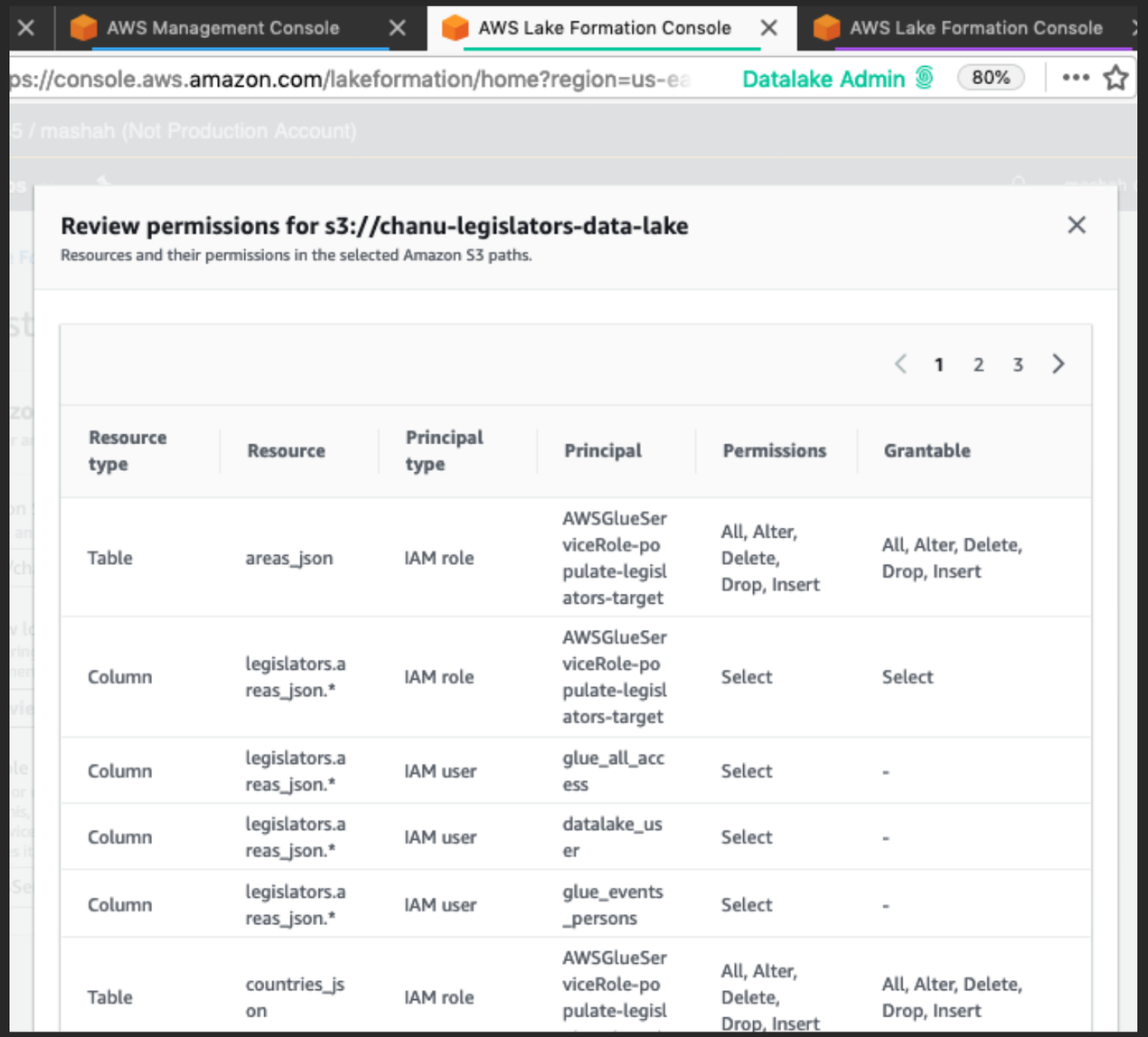
Review location permissions

Audit resources that point to locations you will register.

Once a location is registered:

Integrated services will use Lake Formation creds

This may break existing usage



Resource type	Resource	Principal type	Principal	Permissions	Grantable
Table	areas_json	IAM role	AWSGlueServiceRole-populate-legislators-target	All, Alter, Delete, Drop, Insert	All, Alter, Delete, Drop, Insert
Column	legislators.areas_json.*	IAM role	AWSGlueServiceRole-populate-legislators-target	Select	Select
Column	legislators.areas_json.*	IAM user	glue_all_access	Select	-
Column	legislators.areas_json.*	IAM user	datalake_user	Select	-
Column	legislators.areas_json.*	IAM user	glue_events_persons	Select	-
Table	countries_json	IAM role	AWSGlueServiceRole-populate-legisl	All, Alter, Delete, Drop, Insert	All, Alter, Delete, Drop, Insert

Enable GetDataAccess

Add this IAM policy to all users

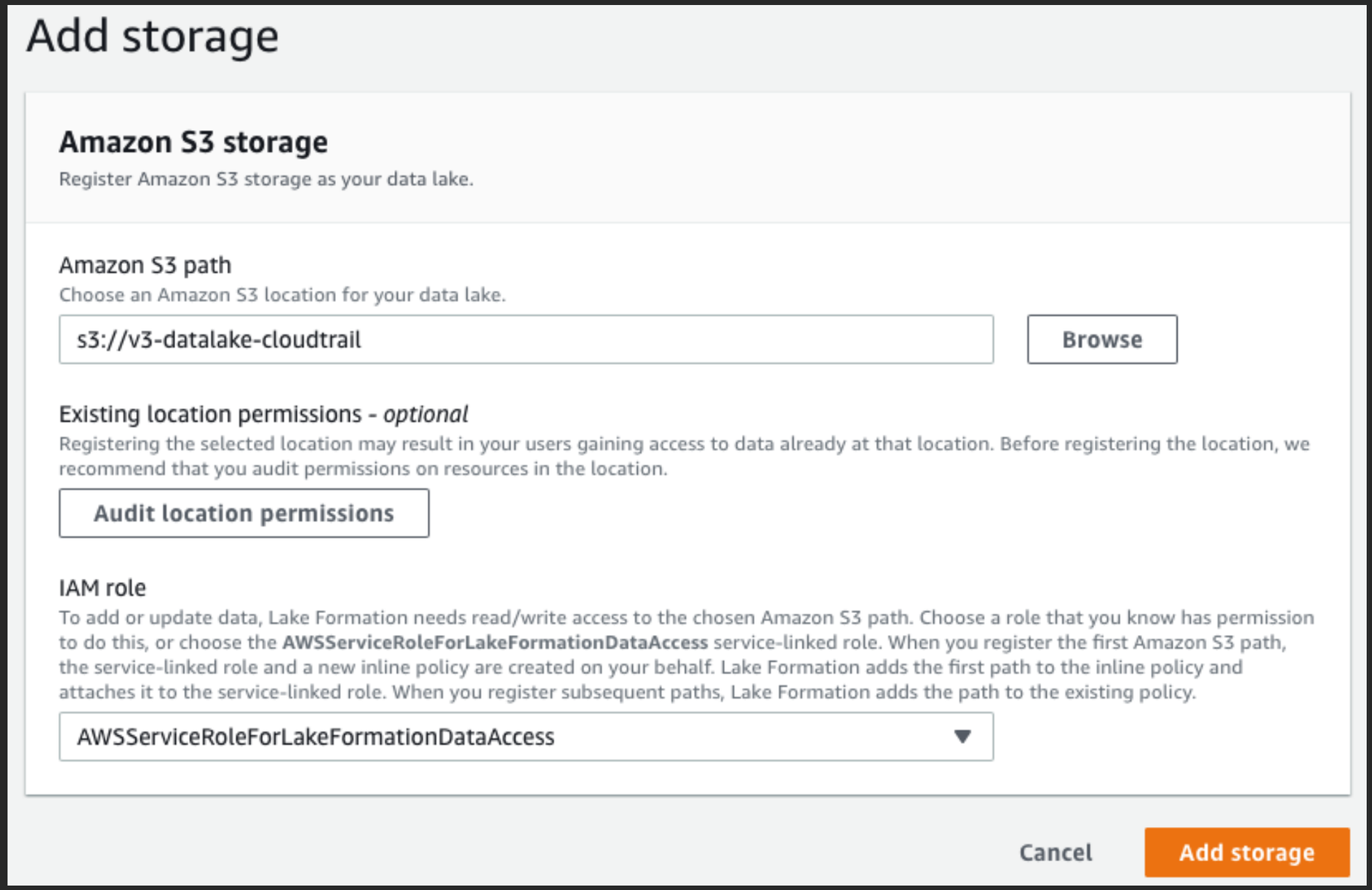
GetDataAccess is a virtual API

It is used to authorize the following APIs:
- GetTemporaryTableCredentials
- GetTemporaryPartitionCredentials

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "lakeformation:getDataAccess"
         ],
         "Resource": "*"
      }
   ]
}
```

Register location

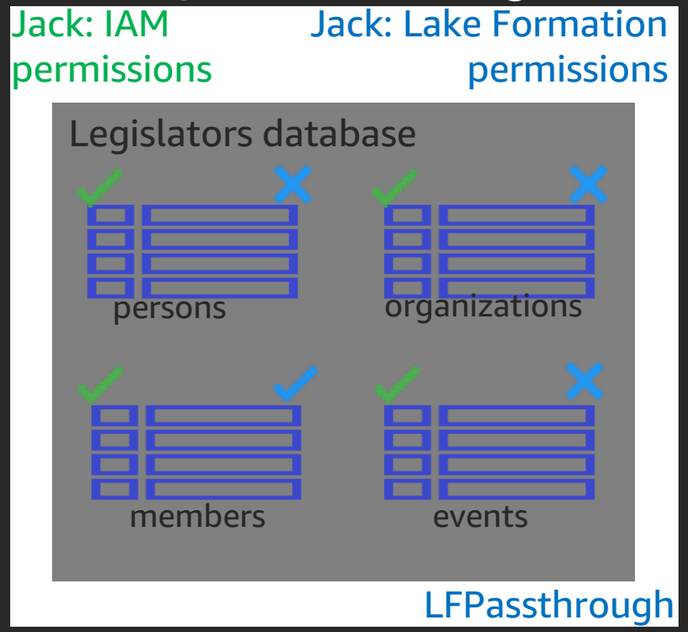
Register a location using the Lake Formation console or APIs.

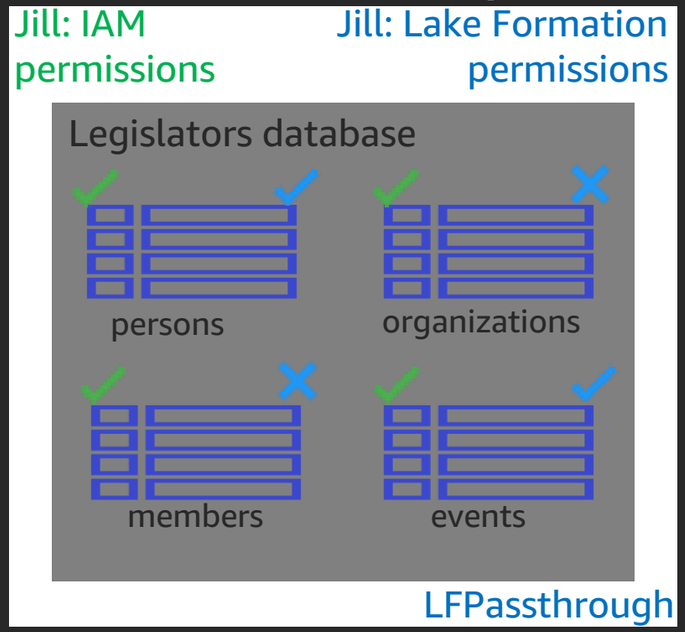


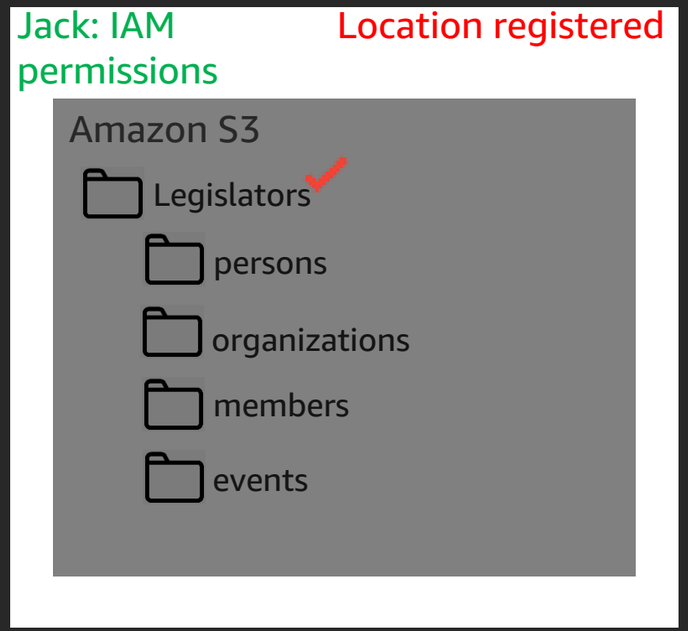
Upgrading to use Lake Formation permissions

1. Grant Lake Formation permissions
2. Revoke LFPassthrough
3. Utilize coarse-grained IAM permissions
4. Enable GetDataAccess / register location
5. Optionally lock down S3 permissions

5. Optionally lock down S3 permissions







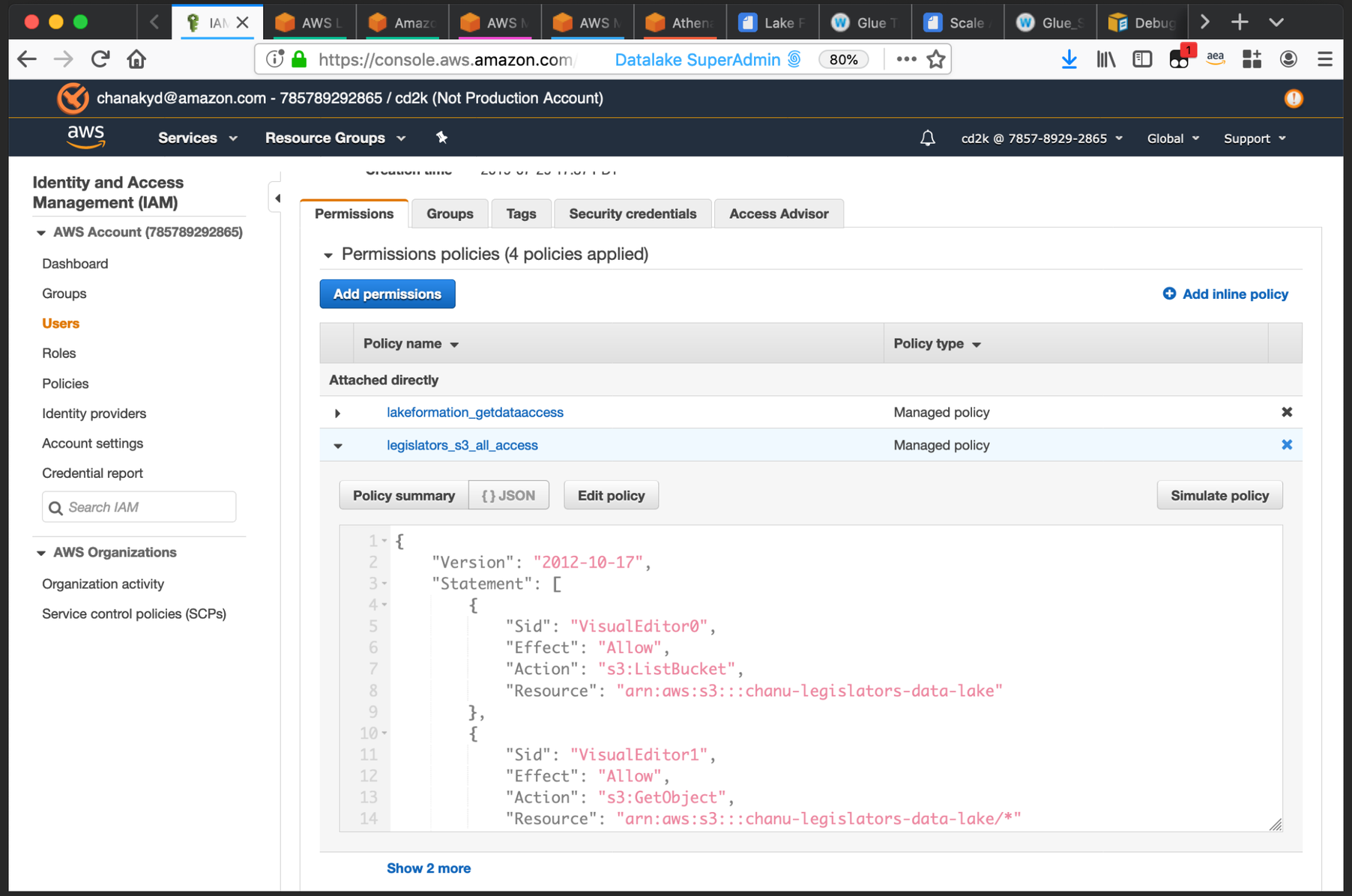


Because Lake Formation can vend credentials, it is now possible to lock down S3 access policies. This step is optional.

Remember that Lake Formation only vends credentials to integrated services. So some users may still require S3 permissions outside of Lake Formation, in order to use the console, or services not integrated with Lake Formation.

Lock down S3 permissions

Remove S3 access IAM policies.



Agenda

Why the AWS Lake Formation security model?

Securing and accessing metadata

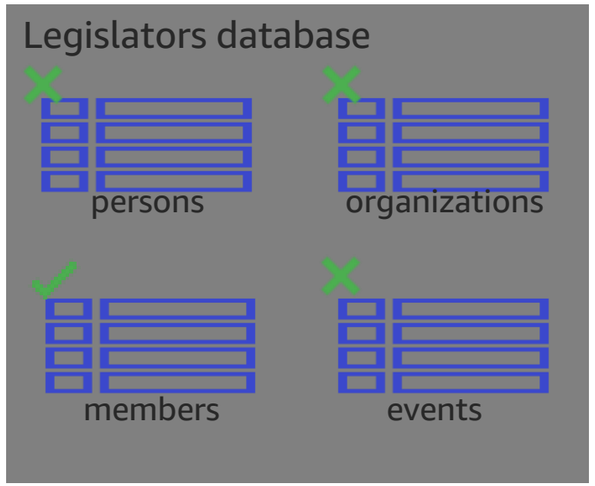
Securing and accessing data (Amazon S3 locations)

Upgrading to use Lake Formation permissions

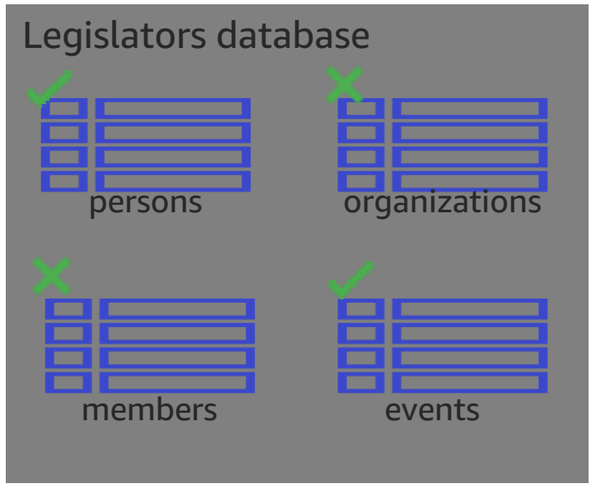
Conclusion

We started here

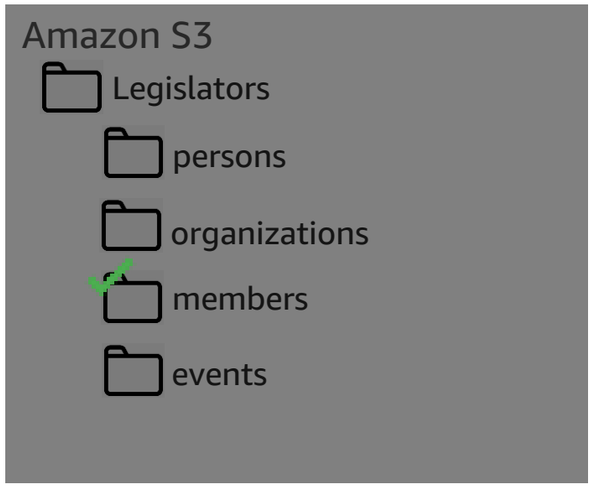
Jack: IAM permissions



Jill: IAM permissions



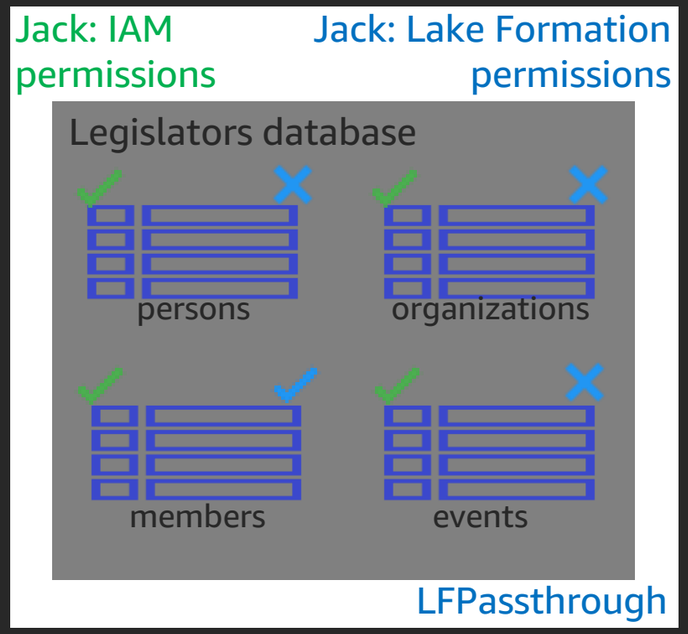
Jack: IAM permissions

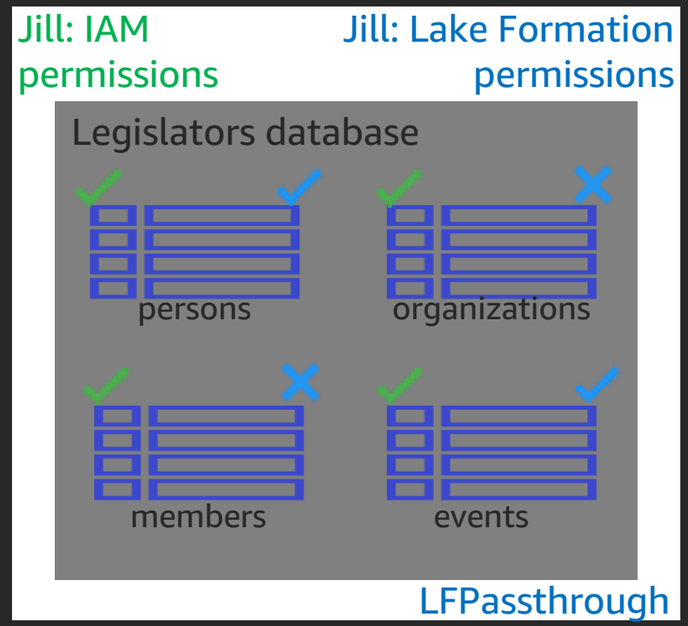


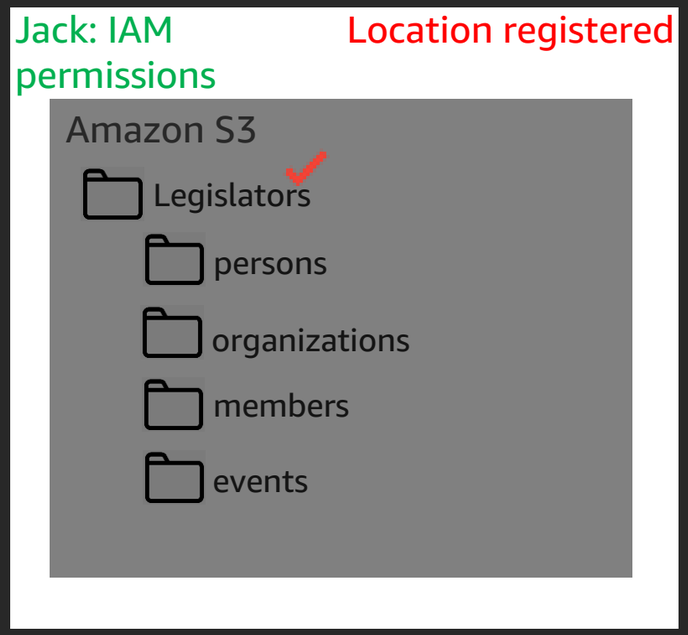
Jill: IAM permissions



We finished here









Conclusion

Worked at every step in the upgrade process!

No data migration

No metadata state migration

You just migrated the permissions you wanted

Data catalog was running the entire time—no disruptions!

Thank you!

Chanakya C. Damarla

chanakyd@amazon.com

AWS re:Invent

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws