Building a Data Lake from scratch on AWS using AWS Lake Formation

16 March 2021 - 10 min. read

Amazon S3 | AWS Identity and Access Management (IAM) | AWS Lake Formation | Data and Analytics | Data Lake | Data Security and Governance | MLOps

Introduction

Leveraging available data (Big Data) has become a significant focus for most companies in the last decades. In the last few years, the advent of Cloud Computing has democratized access to more powerful IT resources, thus eliminating the costs and hassles of managing the necessary infrastructure required in an on-premises data center.

Cloud Computing also helps companies use their data efficiently, lowering engineering costs thanks to its managed services' powerfulness.

It also promotes the use of on-demand infrastructures, making it easier to re-think, re-engineer, and re-architect a data lake to explore new use cases.

Being data a focal point for business decisions, managing it efficiently becomes a priority.

Among many ways to do so, the data lake concept, a scalable, low-cost, centralized data repository for storing raw data from various sources, has grown to success. It enables users to store data as-is without structuring it first to run different analytics types, gaining insights, and guiding more accurate strategic business decisions.

Building a data lake is not an easy task: it involves numerous manual steps, making the process complex and, more importantly, very time-consuming. Data usually comes from diverse sources and should be carefully monitored.

Moreover, managing this amount of data requires several procedures to avoid leaks and security breaches, which means you need to set up access management policies, enable encryption of sensitive data and manage keys for it.

Without the right choices about technology, architecture, data quality, and data governance, a data lake can quickly become an isolated mess of difficult-to-use, hard-to-understand, often inaccessible data.

Fortunately, AWS Cloud comes to the rescue with many services designed to manage a data lake, such as AWS Glue and S3.

For this article, we will assume the reader already has some knowledge about AWS Services and understands the concepts behind AWS Glue and S3. If this is not the case, we encourage you to read our latest stories about ingesting data for Machine Learning workloads and managing complex Machine Learning projects via Step Functions.

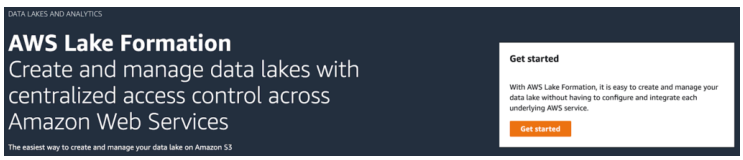
We will explore how to build a very simple data lake using Lake Formation quickly. Then, we will focus on the security and governance advantages that this service offers over plain AWS Glue.

Let us dig into it!

Quick Setup

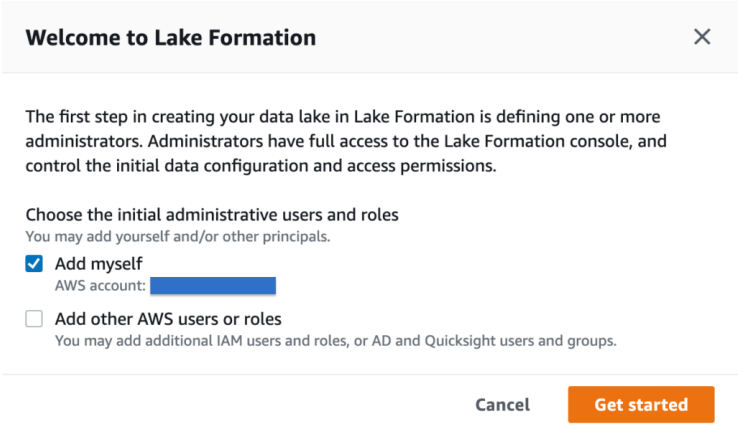
Before focusing on the advantages of managing a data lake through AWS Lake Formation, we first need to create a simple one.

Let us go to the AWS console and choose AWS Lake Formation in the service list or via the search bar. We will find this dashboard:



Welcome screen of Lake Formation

After clicking on “Get started,” we will be asked to set up an administrator for the data lake; it is possible to add AWS users and roles available on the account you are logged into. Select a suitable one, preferably a role, assumable with temporary credentials from humans and services, and go on.



Select a user or a role

When we gain access to the Lake Formation dashboard it’s time to add a Lake Location, which is a valid S3 path to retrieve data from. Data can be obtained by various means, for example with **AWS Glue Jobs**, through the combination of **AWS Kinesis stream** and **Data Firehose**, or by simply uploading data directly to **S3**.

Let’s quickly review all the possibilities to populate our **Glue Catalog** (which defines our data lake under the hood)

First, we’ll add the data lake location by clicking on the “Register location” button in the Register and Ingest section of the service’s dashboard, like in the figure.

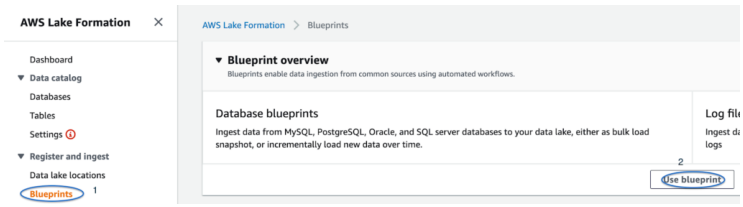
Add a new location for the data lake

We’ll be asked to select an S3 bucket, let’s do so, add a suitable role (or let AWS create one for us), and finally click on “Register location”.

Now we can:

1. simply upload data to S3 before starting the crawling process;
2. use a suitable combination of AWS services to ingest data, like Kinesis Stream and Firehose (see our story for more information);
3. use a **blueprint** from Lake Formation to easily obtain data from various databases or log sources;

Let's review rapidly the third option, which is limited, but still interesting and not yet covered in our previous posts.



Use a blueprint workflow

Clicking on "Use blueprint", we'll be redirected to a form where we can select whether we want to grab data from a database or a log source.

Just follow the instructions to set up a workload, which is basically a Glue ETL Job where all the options for the Extract, Transform, and Load steps are in one place.

For example, for a MySQL, MSSQL, or Oracle database add (or create) an AWS Glue connection, specifying also the source DB and table in this format: **/<_tag_>, encoded_< / strong, encoded_tag_opentable_>tag_closed** Add (or create) the target Glue Catalog, specifying a DB and a table, also browse with the tool provided, for a suitable S3 path to host the catalog data.

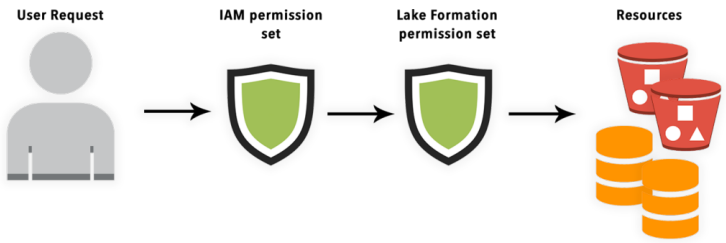
Select a workflow name, decide the job frequency, i.e. "Run on demand", and a table prefix, the other options can be left as defaults.

A couple of notes: always opt for **parquet** format in the target S3 section, as it gives a solid performance boost for dataset operations later on. Also, if you plan to use Athena to query your catalog, please use "_" instead of "-" for database and table names, as the latter character sometimes can lead to unwanted compatibility issues.

Enhanced Security

Once Lake Formation is up and running, we can focus on the details that make it stand out: in primis, a coarse-permission model that works augmenting the one provided by IAM.

This centrally defined model enables fine-grained access to data stored in data lakes with a simple grant/revoke mechanism, like in the figure shown below:



How a request passes through 2 stages of permission before having access to resources

Lake Formation permissions are also enforced at the table and column level and work on all the full stack of AWS services for analytics and machine learning, including, but not limited to, Amazon Athena, Redshift, Sagemaker, and are also mapped with S3 objects under the hood.

Access control in AWS Lake Formation is divided into two distinct areas:

- **Metadata access control** – Permissions on Glue Data Catalog resources that allow **principals** to create, read, update, and delete metadata databases and tables.

- **Underlying data access control** – Permissions for S3 which include data access and data location permissions. Data access permissions enable **principals** to read and write S3 Objects. Data location permissions enable the creation of metadata databases and tables that point to specific S3 locations.

Using these elements to centralize the data access policies is simple: first shut down any direct access to required buckets in S3, so Lake Formation manages all data access.

Next, configure data protection and **access policies** to enforce those policies across all the AWS services accessing data in the data lake. By leveraging Metadata and Object permissions, we can configure users and roles to only access specific data down to the table and column level.

Before assigning policies to users and resources, Lake Formation needs some required personas to work correctly, and those are also needed to grant backward compatibility with previous created IAM permissions for S3, AWS Glue, Athena, and more:

IAM administrator

Users who can create IAM users and roles. Has the AdministratorAccess AWS managed policy. It can also be designated as a data lake administrator.

Data lake administrator

Users who can register Amazon S3 locations, access the Data Catalog, create databases, create and run workflows, grant Lake Formation permissions to other users, and view AWS CloudTrail logs.

Workflow role

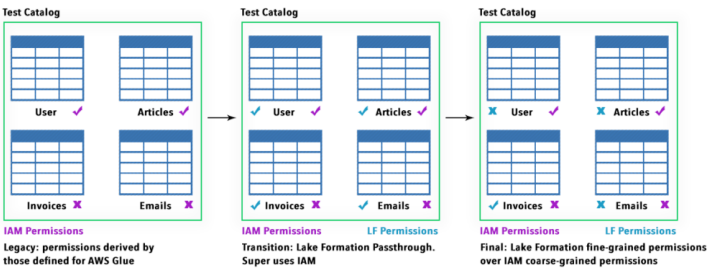
Role that runs a workflow on behalf of a user. You specify this role when you create a workflow from a blueprint. You specify this role when you create a workflow from a blueprint.

The first two personas are **IAMAllowedPrincipals** and have granted "Super" permission and "Use only IAM access control" enabled by default grating backward compatibility with pre-existing workloads with Glue, S3, and Athena.

How permissions work

Fine-grained Permissions are organized in a way that Lake formation can use to substitute coarse-grained IAM permissions. This helps for more seamless transitions from the old permission set to the one managed by Lake Formation.

This is a simple schema illustrating the possible choices:



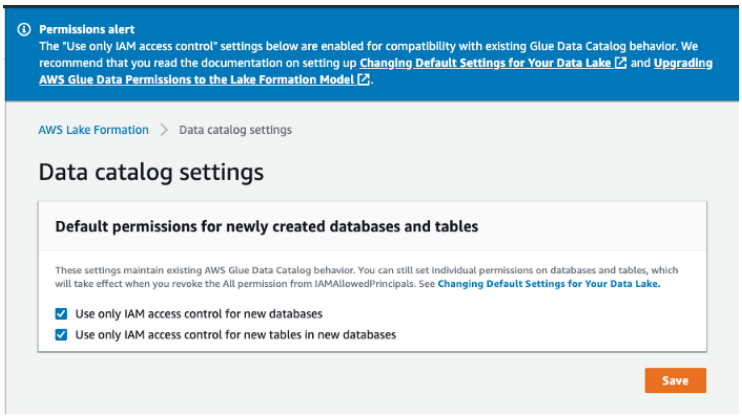
How transition from IAM to Lake Formation permission set works

To view a list of permissions available via Lake Formation follow the official AWS documentation.

Lake Formation also currently supports Server-Side-Encryption on S3, as well as private endpoints for VPC.

It also records all activity in AWS CloudTrail (which can also be a supported dataset), giving an upper hand on network isolation and auditability.

Lake Formation permissions **apply only in the Region in which they were granted.** For backward compatibility, Lake Formation **passes through IAM permissions for new resources until directed differently**.



Warning on the use of legacy permissions

Enhanced Governance

What really helps maintain control over your data lake is that with Lake Formation, we finally have a centralized dashboard to control your S3 Locations, ETL Jobs, Crawlers, Glue Catalogs, and moreover permissions.

Another exciting feature is that Lake Formation comes with Cloud Trail enabled, so every action done by users or services via IAM roles is checked and logged directly in the dashboard.

Cloud Trail access activity directly from Lake Formation dashboard

Another matter that we must manage when dealing with data lakes is data deduplication and data cleaning, which, if passed over, leads to inconsistent, inefficient, often inaccessible data.

By encapsulating AWS Glue capabilities, Lake Formation offers **FindMatches ML transform**: a Glue Job used to remove duplicated data by leveraging Machine Learning algorithms. We can select a threshold for Accuracy to indicate how much precision the algorithm must use to identify potentially duplicate data (more precision, more costs).

To check it in detail follow this tutorial from AWS.

Preview capabilities

Here is a quick preview of the features currently in closed beta. You will have to sign in with AWS to request early access. Transactions - Insert, delete, and modify rows concurrently

To manage the evolution of data in time, it is essential to define procedures capable of keeping the data lake always up-to-date. This is crucial as access to data must be granted to different users at any time, and we also need to ensure data integrity. Data is also often organized in both structured and unstructured ways.

Implementing real-time updates is complex and challenging to scale. AWS Lake Formation introduces, in preview, new APIs that support ACID transactions using a new data lake table type, called a Governed table.

It allows multiple users to concurrently insert, delete, and modify rows across tables while still allowing other users to run queries and machine learning models on the same data sets, with the assurance of data being actual.

Row-level security

Making sure users have access to only the correct data in a data lake is difficult. Data lake administrators often maintain multiple copies of data to apply different security policies for different users. This adds complexity, operational overhead, and extra storage costs.

AWS Lake Formation already allows setting access policies to hide data, even on a per-column basis, i.e., social security numbers.

With row-level security, we can give special per-row permissions to users and roles, i.e., access to specific regional data or data related to a specific bank account.

Acceleration

Better performance with filtering, aggregations, and automatic file compaction, thanks to a new storage optimizer that automatically combines small files into larger files to speed up queries by up to 7x. The process is performed in the background with no performance impact on production workloads.

References

https://pages.awscloud.com/Lake_Formation_Feature_Preview.html

https://aws.amazon.com/lake-formation/faqs/

https://docs.aws.amazon.com/lake-formation/latest/dg/access-control-overview.html

https://docs.aws.amazon.com/lake-formation/latest/dg/permissions-reference.html

Takeaways

We have seen all the features that make Lake Formation a befitting choice for managing data lakes on AWS.

We focused mainly on this service's security and governance aspects, showing how managing permissions on an Object-level in S3 can be a hectic process, simplified by Lake Formation permissions.

We showed how it enables to grant/revoke permissions to users or roles on a table/column level.

We have seen that AWS Lake Formation Permissions are better suited than IAM permissions to secure a data lake because they are enforced on logical objects like a database, table, or column instead of files and directories; they provide granular control for column-level access.

We have also seen that these permissions are internally mapped to underlying objects sitting in S3.

Thanks to a straightforward UI, we do not need to keep multiple tabs open to keep track of ETL Jobs, S3 locations, and Data Catalogs for our workloads. All this

information resides in a single dashboard where we can directly revoke/grant permissions of the objects residing there.

We took a quick look also at the new features available in early access. In particular, a new type of table, the **governed** table, allowing for seamless transactions to keep data always up-to-date. The possibility of using a per-row access policy and a new storage optimizer to increase performance on managing large quantities of small files.

Though managing permissions, data ingestion workflow is made easy, but most of the Glue processes like ETL, Crawler, ML specific transformations have to be set up manually.

So, this is it! As always, feel free to comment in the section below, and reach us for any doubt, question or idea!

See you in a couple of weeks for another exciting story! Stay safe and **#proud2becloud.**

Alessandro Gaggia

Head of software development at beSharp and Full-Stack Developer, I keep all our codebases up-to-date. I write code in almost any language, but Typescript is my favorite. I live for IT, Game design, Cinema, Comics, and... good food. Drawing is my passion!
