# Table of Contents

What is AWS Entity Resolution? ................................................................. 1  
Are you a first-time AWS Entity Resolution user? ........................................ 1  
Features of AWS Entity Resolution ........................................................... 1  
Related services ....................................................................................... 3  
Accessing AWS Entity Resolution ................................................................ 3  
Pricing for AWS Entity Resolution ................................................................ 4  
Setting up AWS Entity Resolution ................................................................ 5  
Sign up for AWS ....................................................................................... 5  
Create an administrator user ...................................................................... 5  
Subscribe to a provider service on AWS Data Exchange .............................. 6  
Prepare data tables .................................................................................. 7  
  Step 1: Prepare your input data ................................................................. 7  
  Step 2: Save your input data table in a supported data format .................... 9  
  Step 3: Upload your input data table to Amazon S3 ................................. 10  
  Step 4: Create an AWS Glue table ............................................................ 10  
Create an IAM role for a console user ....................................................... 11  
Create a workflow job role for AWS Entity Resolution .............................. 11  
Creating a schema mapping ..................................................................... 15  
  Create a schema mapping (pre-populated columns) ............................... 15  
  Create a schema mapping (manually defined columns) ......................... 17  
  Create a schema mapping (JSON editor) ............................................... 18  
Next steps ............................................................................................... 19  
Creating a matching workflow ................................................................. 20  
  Rule-based matching workflow ............................................................... 20  
  Machine learning-based matching workflow ....................................... 24  
  Provider service-based matching workflow ......................................... 27  
  Creating a matching workflow with LiveRamp ..................................... 28  
  Creating a matching workflow with TransUnion .................................. 33  
  Creating a matching workflow with UID 2.0 ....................................... 37  
Create and run an ID mapping workflow ................................................. 41  
Creating an ID mapping workflow ......................................................... 41  
Running an ID mapping workflow ......................................................... 46  
Managing AWS Entity Resolution ............................................................. 48  
  Managing schema mappings ................................................................. 48  
    Clone a schema mapping ................................................................. 48  
    Edit a schema mapping ..................................................................... 48  
    Delete a schema mapping ............................................................... 49  
  Managing matching workflows ............................................................. 49  
    Run a matching workflow ............................................................... 50  
    Edit a matching workflow ............................................................... 50  
    Delete a matching workflow ........................................................... 51  
  Managing ID mapping workflows ......................................................... 51  
    Edit an ID mapping workflow ......................................................... 51  
    Delete an ID mapping workflow ..................................................... 51  
Troubleshooting workflows ...................................................................... 51  
I received an error file. ........................................................................... 52  
Security .................................................................................................. 53  
  Data protection .................................................................................... 53  
    Data encryption at rest for AWS Entity Resolution ............................ 54  
    Key management ............................................................................. 54  
    AWS PrivateLink ............................................................................ 60  
Identity and access management ............................................................. 62  
  Audience ............................................................................................ 62  
  Authenticating with identities ............................................................. 63
Managing access using policies ................................................................. 65
How AWS Entity Resolution works with IAM ................................................ 66
Identity-based policy examples .................................................................... 71
AWS managed policies ..................................................................................... 73
Troubleshooting ............................................................................................... 77
Compliance validation ...................................................................................... 78
Resilience ......................................................................................................... 78
Monitoring ........................................................................................................ 80
CloudTrail logs ................................................................................................ 80
  AWS Entity Resolution information in CloudTrail ............................................. 80
  Understanding AWS Entity Resolution log file entries .................................... 81
AWS CloudFormation resources ....................................................................... 82
AWS Entity Resolution and AWS CloudFormation templates .............................. 82
Learn more about AWS CloudFormation ........................................................ 83
Quotas .............................................................................................................. 84
Document history ............................................................................................. 86
Glossary ........................................................................................................... 88
  Amazon Resource Name (ARN) ....................................................................... 88
  Automatic processing ....................................................................................... 88
  AWS KMS key ARN ......................................................................................... 88
  Cleartext .......................................................................................................... 88
  Confidence level (ConfidenceLevel) ................................................................ 88
  Decryption ....................................................................................................... 88
  Encryption ....................................................................................................... 88
  Group name ..................................................................................................... 89
  Hash ................................................................................................................ 89
  Hash protocol (HashingProtocol) .................................................................... 89
  ID mapping workflow ...................................................................................... 89
  Input field ........................................................................................................ 89
  Input Source ARN (InputSourceARN) ............................................................... 89
  Input type ........................................................................................................ 89
  Machine learning-based matching .................................................................. 90
  Manual processing ........................................................................................... 90
  Many-to-Many matching ................................................................................ 90
  Match ID (MatchID) ....................................................................................... 90
  Match key (MatchKey) .................................................................................... 90
  Match key name ............................................................................................. 91
  Match rule (MatchRule) ................................................................................ 91
  Matching ......................................................................................................... 91
  Matching workflow ........................................................................................ 91
  Matching workflow description ...................................................................... 91
  Matching workflow name ............................................................................... 91
  Matching workflow metadata ........................................................................ 91
  Normalization (ApplyNormalization) .............................................................. 92
    Name ............................................................................................................ 92
    Email ............................................................................................................. 92
    Phone ........................................................................................................... 92
    Address ....................................................................................................... 92
    Hashed ......................................................................................................... 94
    Source_ID .................................................................................................... 94
  One-to-One matching ...................................................................................... 94
  Output ............................................................................................................. 95
  OutputS3Path ................................................................................................. 95
  OutputSourceConfig ...................................................................................... 95
  Provider service-based matching ..................................................................... 95
  Rule-based matching ...................................................................................... 95
  Schema .......................................................................................................... 96
Schema description .................................................................................................................. 96
Schema name .......................................................................................................................... 96
Schema mapping ..................................................................................................................... 96
Schema mapping ARN ............................................................................................................ 96
Unique ID ................................................................................................................................ 96
What is AWS Entity Resolution?

AWS Entity Resolution is a service that helps you match, link, and enhance related records stored across multiple applications, channels, and data stores. You can get started using entity resolution workflows that are flexible, scalable, and can connect to your existing applications and data service providers.

AWS Entity Resolution offers advanced matching techniques, such as rule-based matching, machine learning-powered, and data service provider-led matching, to help you more accurately link and enhance related records of customer information, product codes, or business data codes.

You can use AWS Entity Resolution to create a unified view of customer interactions by linking recent events (such as ad clicks, cart abandonment, and purchases) with pseudonymized signals from with your data service providers into a unique entity ID. You can also better track products that use different codes (for example, SKU, UPC) across your stores. You can use AWS Entity Resolution to control matching accuracy and better protect data security while minimizing data movement.

Topics

- Are you a first-time AWS Entity Resolution user? (p. 1)
- Features of AWS Entity Resolution (p. 1)
- Related services (p. 3)
- Accessing AWS Entity Resolution (p. 3)
- Pricing for AWS Entity Resolution (p. 4)

Are you a first-time AWS Entity Resolution user?

If you're a first-time user of AWS Entity Resolution, we recommend that you begin by reading the following sections:

- Features of AWS Entity Resolution (p. 1)
- Accessing AWS Entity Resolution (p. 3)
- Setting up AWS Entity Resolution (p. 5)

Features of AWS Entity Resolution

AWS Entity Resolution includes the following features:

- Flexible and customizable data preparation

AWS Entity Resolution reads your data from AWS Glue to use as inputs for match processing. You can specify a maximum of 20 data inputs. Each row of the data input table is processed as a record, with a unique entity serving as a primary key. AWS Entity Resolution can operate on encrypted datasets. You must first define the schema mapping (p. 96) for AWS Entity Resolution to understand what input fields you want to use in your matching workflow (p. 91). You can bring your own data schema, or blueprint, from an existing AWS Glue data input, or build your custom schema using an interactive user interface or JSON editor. By default, data inputs are also normalized (p. 92) before matching to improve match processing such as removing special characters and extra spaces, and formatting text to lowercase. You can turn off normalization if your data input has already been normalized. We also provide a GitHub library which you can use to further customize the data normalization process to suit your needs.
• Configurable entity matching workflows

An entity matching workflow (p. 91) is a sequence of steps that you set up to tell AWS Entity Resolution how to match your data input and where to write the consolidated data output. You can set up one or more matching workflows to compare different data inputs and use different matching techniques, such as rule-based matching (p. 95), machine learning matching (p. 90), or data service provider-led matching (p. 95) without entity resolution or machine learning (ML) experience. You can also view the job status of existing matching workflows and metrics, such as resource number, number of records processed, and number of matches found.

• Ready-to-use rule-based matching

This matching technique includes a set of ready-to-use rules in the AWS Management Console or command line interface to find related records, based on your input fields. You can customize the rules, such as adding or removing input fields for each rule, delete rules, rearrange the priority of rules, and create new rules. You can also reset the rules to return them to their original configurations. The data output in your Amazon S3 bucket will have match groups generated by AWS Entity Resolution using the rule-based matching technique (p. 95). Each match group has the rule number used to generate that match associated to it to help you understand the match. For example, the rule number can demonstrate the precision of each match group such that rule one is more precise than rule two.

• Pre-configured machine learning matching

This matching technique includes a pre-configured ML model to find matches across all of your data inputs, especially consumer-based records. The model uses all input fields associated with name, email address, phone number, address, and date of birth data types. The model generates match groups of related records with a confidence score (p. 88) in each group explaining the quality of the match relative to other match groups. The model considers missing input fields and analyzes the entire record together to represent an entity. The data output in your Amazon S3 bucket will have match groups generated by AWS Entity Resolution using the ML matching. This is where each match group has a confidence score associated to it between 0.0–1.0 explaining the precision of the match.

• Matching records with data service providers

With AWS Entity Resolution you can match, link, and enhance your records with leading data service vendors and licensed data sets to expand your ability to understand, reach, and service your customers. For example, you can append attributes to your data to enhance your records, or you can improve the interoperability of systems and platforms you work with to meet your business goals. You can use this matching workflow with a few clicks, removing the need to build and maintain complex proprietary integrations. You must have a license agreement with these data service providers to take advantage of this matching technique.

• Manual bulk processing and automatic incremental processing

You can use data processing to help convert your data input or inputs into a consolidated data output table with similar records that have a common match ID generated using entity matching workflow configurations. Using the API and AWS Management Console or the command line interface, you can run manual bulk processing (p. 90) on demand, based on your existing extract, transform, and load (ETL) data pipeline, which re-processes all data for any new matches and updates to existing matches. Also, for rule-based matching scenarios, you can initiate automatic incremental processing (p. 88) so that as soon as new data is available in your Amazon S3 bucket, the service reads those new records and compares them against existing records. This keeps your matches up to date with any changes in Amazon S3 data.

• Near real-time lookup

Looking up any entity fields through the AWS Entity Resolution GetMatchId API helps you synchronously retrieve an existing match ID. You can call AWS Entity Resolution with personally identifiable information (PII) attributes acquired through different sources and channels. AWS Entity Resolution will hash those attributes for data protection and retrieve the corresponding match ID to link and match the customer. For example, you can get a web signup with an associated name,
email, and mailing address. You use the AWS Entity Resolution GetMatchId API to find out if this customer or entity already exists in your matched results stored in your S3 bucket, along with the corresponding entity match ID associated to it. After you get the entity match ID, you can find the transactional information associated to it in your source applications, such as your customer relationship management (CRM) or customer data platform (CDP) systems.

- **Data protection and regionalization by design**

AWS Entity Resolution offers a default encryption capability that can help you protect your data, and will equip you with an encryption key for every data input into the service. For example, AWS Entity Resolution gives you the flexibility to bring server-side encrypted and hashed data to run rule-based matching workflows. AWS Entity Resolution supports regionalization, which means that your matching workflows run to process your data in the same Region from where you're using the service. You can also encrypt and hash the data output in Amazon S3 before using your resolved data in other applications.

### Related services

The following AWS services are related to AWS Entity Resolution:

- **Amazon S3**
  
  Store data that you bring into AWS Entity Resolution in Amazon S3.
  
  For more information, see the following topics:
  

- **AWS Glue**
  
  Create AWS Glue tables from your data in Amazon S3 for use in AWS Entity Resolution.

  For more information, see the following topics:
  
  [What is AWS Glue?](https://docs.aws.amazon.com/glue/latest/dg/index.html) in the *AWS Glue Developer Guide*

- **AWS CloudTrail**
  
  Use AWS Entity Resolution with CloudTrail logs to enhance your analysis of AWS service activity.

  For more information, see [Logging AWS Entity Resolution API calls using AWS CloudTrail](https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/Logging-API-calls.html) (p. 80).

- **AWS CloudFormation**
  
  Create the following resources in AWS CloudFormation: AWS::EntityResolution::MatchingWorkflow, AWS::EntityResolution::SchemaMapping, and AWS::EntityResolution::IdMappingWorkflow.

  For more information, see [Creating AWS Entity Resolution resources with AWS CloudFormation](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-entityresolution-matchingworkflow.html) (p. 82)

### Accessing AWS Entity Resolution

You can access AWS Entity Resolution through the following options:

- Directly through the AWS Entity Resolution console at [https://console.aws.amazon.com/entityresolution/](https://console.aws.amazon.com/entityresolution/).
- Programmatically through the AWS Entity Resolution API. For more information, see the *AWS Entity Resolution API Reference*. 
Pricing for AWS Entity Resolution

For pricing information, see AWS Entity Resolution Pricing.
Setting up AWS Entity Resolution

Before you use AWS Entity Resolution for the first time, complete the following tasks.

Topics

- Sign up for AWS (p. 5)
- Create an administrator user (p. 5)
- Subscribe to a provider service on AWS Data Exchange (p. 6)
- Prepare data tables (p. 7)
- Create an IAM role for a console user (p. 11)
- Create a workflow job role for AWS Entity Resolution (p. 11)

Sign up for AWS

If you do not have an AWS account, complete the following steps to create one.

To sign up for an AWS account

2. Follow the online instructions.

Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

When you sign up for an AWS account, an AWS account root user is created. The root user has access to all AWS services and resources in the account. As a security best practice, assign administrative access to an administrative user, and use only the root user to perform tasks that require root user access.

Create an administrator user

To create an administrator user, choose one of the following options.

<table>
<thead>
<tr>
<th>Choose one way to manage your administrator</th>
<th>To</th>
<th>By</th>
<th>You can also</th>
</tr>
</thead>
<tbody>
<tr>
<td>In IAM Identity Center (Recommended)</td>
<td>Use short-term credentials to access AWS.</td>
<td>Following the instructions in Getting started in the AWS IAM Identity Center User Guide.</td>
<td>Configure programmatic access by Configuring the AWS CLI to use AWS IAM Identity Center in the AWS Command Line Interface User Guide.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Choose one way to manage your administrator

<table>
<thead>
<tr>
<th>To</th>
<th>By</th>
<th>You can also</th>
</tr>
</thead>
<tbody>
<tr>
<td>In IAM in the IAM User Guide.</td>
<td>Follow the instructions in Creating your first IAM admin user and user group in the IAM User Guide.</td>
<td>Configure programmatic access by Managing access keys for IAM users in the IAM User Guide.</td>
</tr>
<tr>
<td>In IAM (Not recommended)</td>
<td>Use long-term credentials to access AWS.</td>
<td></td>
</tr>
</tbody>
</table>

Subscribe to a provider service on AWS Data Exchange

Complete the following procedure if you are using a provider service-based matching workflow (p. 95) or an ID mapping workflow (p. 89). If you are not using a provider service-based matching workflow or an ID mapping workflow, you can skip this step.

In AWS Entity Resolution, you can choose to run a matching workflow with one of the following provider services if you have a subscription with that provider on AWS Data Exchange. Your data will be matched with a set of inputs defined by your preferred provider.

- LiveRamp
  - LiveRamp Identity Resolution
  - LiveRamp Transcoding
- TransUnion
  - TransUnion TruAudience Transfer-less Identity Resolution & Enrichment
  - TransUnion TruAudience Transfer-less Identity Resolution
- Unified ID 2.0
  - Unified ID 2.0 Operator

In addition, you can run an ID mapping workflow with LiveRamp if you have a subscription with that provider.

- LiveRamp
  - LiveRamp Transcoding

There are two ways to subscribe to a provider service:

- **Private offer** – If you have an existing relationship with a provider, follow the Private products and offers procedure in the AWS Data Exchange User Guide to accept a private offer on AWS Data Exchange.
- **Bring your own subscription** – If you already have an existing data subscription with a provider, follow the Bring Your Own Subscription (BYOS) offers procedure in the AWS Data Exchange User Guide to accept a BYOS offer on AWS Data Exchange.

After you have subscribed to a provider service on AWS Data Exchange, you can then create a matching workflow or an ID mapping workflow with that provider service.
Prepare data tables

In AWS Entity Resolution, each of your input data tables contain source records. These records contain consumer identifiers such as first name, last name, email address, or phone number. These source records can be matched with other source records that you provide within the same or other input data tables. Each record must have a unique Record ID (Unique ID (p. 96)) and you must define it as a primary key while creating a schema mapping within AWS Entity Resolution.

Every input data table is available as an AWS Glue table backed by Amazon S3. You can use your first-party data already within Amazon S3, or import data tables from other SaaS providers into Amazon S3. After the data is uploaded to Amazon S3, you can use an AWS Glue crawler to create a data table in the AWS Glue Data Catalog. You can then use the data table as an input to AWS Entity Resolution.

Preparing your data tables involves the following steps:

Topics
- Step 1: Prepare your input data (p. 7)
- Step 2: Save your input data table in a supported data format (p. 9)
- Step 3: Upload your input data table to Amazon S3 (p. 10)
- Step 4: Create an AWS Glue table (p. 10)

Step 1: Prepare your input data

Complete the following procedure if you are using a matching workflow with a provider service. If you are not using a matching workflow with a provider service, you can skip this step.

For more information, see Subscribe to a provider service on AWS Data Exchange (p. 6).

If you want to run a matching workflow with a provider service-based matching workflow or an ID mapping workflow, consult the following table to prepare your input data:

<table>
<thead>
<tr>
<th>Provider service</th>
<th>Unique ID needed?</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>LiveRamp</td>
<td>Yes</td>
<td>Ensure the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- The <strong>Unique ID (p. 96)</strong> can be either your own pseudonymous identifier or a row ID.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Your data input file format and normalization is aligned with the LiveRamp guidelines.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- For more information about input file formatting guidelines for the matching workflow, see Perform Identity Resolution Through ADX in the LiveRamp documentation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- For more information about input file formatting guidelines for the ID mapping workflow, see Perform Transcoding Through ADX in the LiveRamp documentation.</td>
</tr>
<tr>
<td><strong>Provider service</strong></td>
<td><strong>Unique ID needed?</strong></td>
<td><strong>Actions</strong></td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>TransUnion</td>
<td>Yes</td>
<td>Ensure the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• A <a href="#">Unique ID</a> (p. 96) exists for TransUnion Data Enrichment.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pass along attributes are allowed to persist in input and output to TransUnion. Household E keys and HHID are specific to the client namespace.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Phone number</strong> should be 10 digits, without any special characters such as spaces or hyphens.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Addresses</strong> should be split into</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• a single address line (combine address lines 1 &amp; 2, if present)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• city</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• zip (or zip plus4), without any special characters such as spaces or hyphens</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• state, specified as 2 letter code 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Email addresses</strong> should be in plaintext.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>First Name</strong> can be lower or upper case, nicknames are supported, but titles and suffixes should be excluded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Last Name</strong> can be lower or upper case, middle initials to be excluded.</td>
</tr>
</tbody>
</table>
Step 2: Save your input data table in a supported data format

If you already saved your input data in a supported data format, you can skip this step.

To use AWS Entity Resolution, the input data must be in a format that AWS Entity Resolution supports. AWS Entity Resolution supports the following data formats:

- comma-separated value (CSV)
  
  **Note**
  
  LiveRamp only supports CSV files.

- Parquet
Step 3: Upload your input data table to Amazon S3

If you already have your first-party data table in Amazon S3, you can skip this step.

**Note**
The input data must be stored in Amazon Simple Storage Service (Amazon S3) in the same AWS account and AWS Region in which you want to run the matching workflow.

**To upload your input data table to Amazon S3**

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose **Buckets**, and then choose a bucket to store your data table.
3. Choose **Upload**, and then follow the prompts.
4. Choose the **Objects** tab to view the prefix where your data is stored. Make a note of the name of the folder.
   
   You can select the folder to view the data table.

Step 4: Create an AWS Glue table

The input data in Amazon S3 must be cataloged in AWS Glue and represented as an AWS Glue table. For more information about how to create an AWS Glue table with Amazon S3 as the input, see [Working with crawlers on the AWS Glue console](https://aws.amazon.com/glue/) in the AWS Glue Developer Guide.

**Note**
AWS Entity Resolution doesn't support partitioned tables.

In this step, you set up a crawler in AWS Glue that crawls all the files in your S3 bucket and create an AWS Glue table.

**Note**
AWS Entity Resolution doesn't currently support Amazon S3 locations registered with AWS Lake Formation.

**To create an AWS Glue table**

1. Sign in to the AWS Management Console and open the AWS Glue console at https://console.aws.amazon.com/glue/.
2. From the navigation bar, select **Crawlers**.
3. Select your S3 bucket from the list, and then choose **Add crawler**.
4. On the **Add crawler** page, enter a **Crawler name** and then choose **Next**.
5. Continue through the **Add crawler page**, specifying the details.
6. On the **Choose an IAM role** page, choose **Choose an existing IAM role** and then choose **Next**.
   
   You can also choose **Create an IAM role** or have your administrator create the IAM role if needed.
7. For **Create a schedule for this crawler**, keep the **Frequency** default (Run on demand) and then choose **Next**.
8. For **Configure the crawler's output**, enter the AWS Glue database and then choose **Next**.
9. Review all of the details, and then choose **Finish**.
10. On the **Crawlers** page, select the check box next to your S3 bucket and then choose **Run crawler**.
11. After the crawler is finished running, on the AWS Glue navigation bar, choose **Databases**, and then choose your database name.
12. On the Database page, choose Tables in {your database name}.
   a. View the tables in the AWS Glue database.
   b. To view a table's schema, select a specific table.
13. Make a note of the AWS Glue database name and AWS Glue table name.

Create an IAM role for a console user

To create an IAM role

1. Sign in to the IAM console (https://console.aws.amazon.com/iam/) with your administrator account.
2. Under Access management, choose Roles.

   You can use Roles to create short-term credentials, which is recommended for increased security.
   You can also choose Users to create long-term credentials.
3. Choose Create role.
4. In the Create role wizard, for Trusted entity type, choose AWS account.
5. Keep the option This account selected, and then choose Next.
6. For Add permissions, choose Create Policy.

   A new tab opens.

   a. Select the JSON tab, and then add policies depending on the abilities granted to the console user. AWS Entity Resolution offers the following managed policies based on common use cases:

      • AWS managed policy: AWSEntityResolutionConsoleFullAccess (p. 73)
      • AWS managed policy: AWSEntityResolutionConsoleReadOnlyAccess (p. 76)
    b. Choose Next: Tags, add tags (optional), and then choose Next: Review.
    c. For Review policy, enter a Name and Description, and review the Summary.
    d. Choose Create policy.

   You have created a policy for a collaboration member.

    e. Go back to your original tab and under Add permissions, enter the name of the policy that you just created. (You might need to reload the page.)
    f. Select the check box next to the name of the policy that you created, and then choose Next.
7. For Name, review, and create, enter the Role name and Description.

   a. Review Select trusted entities, enter the AWS account for the person or persons who will assume the role (if necessary).
   b. Review the permissions in Add permissions, and edit if necessary.
   c. Review the Tags, and add tags if necessary.
   d. Choose Create role.

Create a workflow job role for AWS Entity Resolution

AWS Entity Resolution uses a workflow job role to run a workflow. You can create this role using the console if you have the necessary IAM permissions. If you don't have CreateRole permissions, ask your administrator to create the role.
To create a workflow job role for AWS Entity Resolution

2. Under Access management, choose Roles.
   - You can use Roles to create short-term credentials, which is recommended for increased security.
   - You can also choose Users to create long-term credentials.
3. Choose Create role.
4. In the Create role wizard, for Trusted entity type, choose Custom trust policy.
5. Copy and paste the following custom trust policy into the JSON editor.

   ```json
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
           "Service": [
             "entityresolution.amazonaws.com"
           ],
         "Action": "sts:AssumeRole"
       }
     ]
   }
   ```
6. Choose Next.
7. For Add permissions, choose Create Policy.
   A new tab appears.
   a. Copy and paste the following policy into the JSON editor.

   **Note**
   The following example policy supports the permissions needed to read corresponding data resources like Amazon S3 and AWS Glue. However, you might need to modify this policy depending on how you've set up your data sources.
   Your AWS Glue resources and underlying Amazon S3 resources must be in the same AWS Region as AWS Entity Resolution.
   You don't need to grant AWS KMS permissions if your data sources aren't encrypted or decrypted.

   ```json
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Action": [
           "s3:GetObject",
           "s3:ListBucket",
           "s3:GetBucketLocation"
         ],
         "Resource": [
           "arn:aws:s3:::{{input-buckets}}",
           "arn:aws:s3:::{{input-buckets}}/*"
         ],
         "Condition":{
           "StringEquals":{
             "s3:ResourceAccount":[
               "{{account-id}}"
             ]
           },
           "StringLike":{
             "s3:Prefix":[
               "{{prefix}}"
             ]
           },
           "StringEquals":{
             "s3:Delimiter":[
               "/"
             ]
           }
         }
       }
     ]
   }
   ```
b. (Optional) If the input Amazon S3 bucket is encrypted using the customer’s KMS key, add the following:

```
{
  "Effect": "Allow",
  "Action": [
    "kms:Decrypt"
  ],
  "Resource": [
    "arn:aws:kms:{{aws-region}}:{{accountId}}:key/{{inputKeys}}"
  ]
}
```

c. (Optional) If the data being written into the output Amazon S3 bucket needs to be encrypted, add the following:

```
{
  "Effect": "Allow",
  "Action": [
    "s3:PutObject",
    "s3:ListBucket",
    "s3:GetBucketLocation"
  ],
  "Resource": [
    "arn:aws:s3:::{{output-bucket}}",
    "arn:aws:s3:::{{output-bucket}}/*"
  ],
  "Condition":{
    "StringEquals":{
      "s3:ResourceAccount":[
        "{{accountId}}"
      ]
    }
  }
}
```
d. Replace each {{user input placeholder}} with your own information.

| region          | AWS Region of your resources. Your AWS Glue resources, underlying Amazon S3 resources and AWS KMS resources must be in the same AWS Region as the AWS Entity Resolution. |
| accountID       | Your AWS account ID. |
| input-buckets   | Amazon S3 buckets which contains the underlying data objects of AWS Glue where AWS Entity Resolution will read from. |
| output-buckets  | Amazon S3 buckets where AWS Entity Resolution will generate the output data. |
| input-databases | AWS Glue databases where AWS Entity Resolution will read from. |
| input-tables    | AWS Glue tables where AWS Entity Resolution will read from. |
| inputKeys       | Managed keys in AWS Key Management Service. If your input sources are encrypted, AWS Entity Resolution must decrypt your data using your key. |
| outputKeys      | Managed keys in AWS Key Management Service. If you need your output sources to be encrypted, AWS Entity Resolution must encrypt the output data using your key. |

8. Go back to your original tab and under Add permissions, enter the name of the policy that you just created. (You might need to reload the page.)

9. Select the check box next to the name of the policy that you created, and then choose Next.

10. For Name, review, and create, enter the Role name and Description.

   **Note**

   The Role name must match the pattern in the passRole permissions granted to the member who can pass the workflow job role to create a matching workflow. For example, if you’re using the AWSEntityResolutionConsoleFullAccess managed policy, remember to include entityresolution into your role name.

   a. Review Select trusted entities, and edit if necessary.
   b. Review the permissions in Add permissions, and edit if necessary.
   c. Review the Tags, and add tags if necessary.
   d. Choose Create role.

11. The workflow job role for AWS Entity Resolution has been created.
Creating a schema mapping

Creating a schema mapping

To define the input data that you want to resolve, create a schema mapping. The schema mapping process guides you through a set of steps to define the data that you want to resolve by defining your input fields and attribute types, and then defining and grouping your match keys.

There are three ways to create a schema mapping in AWS Entity Resolution:

- Using a guided flow to import existing schema information (p. 15).
- Using a guided flow to manually define input data (p. 17).
- Using the JSON editor to create, paste, or import a schema mapping (p. 18).

Create a schema mapping (pre-populated columns)

This procedure describes the process of creating a schema mapping using the Import from AWS Glue option on the AWS Entity Resolution console. You can use this creation method to define input fields starting with pre-populated columns from an AWS Glue table.

To create schema mapping using pre-populated columns:

1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Data preparation, choose Schema mappings.
3. On the Schema mappings page, in the upper right corner, choose Create schema mapping.
4. For Step 1: Specify schema details, do the following:
   a. For Name and creation method, enter a Schema mapping name and an optional Description.
   b. For Creation method, choose Import from AWS Glue.
   c. Choose the AWS Glue database from the dropdown, and then choose the AWS Glue table from the dropdown.

      To create a new table, go to the AWS Glue console https://console.aws.amazon.com/glue/. For more information, see AWS Glue tables.
   d. For Unique ID, specify the column that can be used to distinctly reference each row of your data.

      Example

      For example: Primary_key, Row_ID, or Record_ID.

      Note

      The Unique ID column is required. The Unique ID must be a unique identifier across all sources. If the Unique ID is not specified, is not unique within the same source or across multiple sources, or overlaps across sources, then the record will be rejected when the matching workflow is run.

   e. For Input fields, choose 1–25 columns to use for matching and for optional pass through.
      i. Select Add columns for pass through if you want to specify the columns that aren't used for matching.
ii. Under **Pass through – optional**, choose the columns to include as pass through columns.

f. (Optional) If you want to enable **Tags** for the resource, choose **Add new tag**, and then enter the **Key** and **Value** pair.

g. Choose **Next**.

5. For **Step 2: Map input fields**:

a. For **Input fields for matching**, specify the **Input type** and **Match key** for each **Input field**.

The **Input type** helps you classify the data. The **Match key** enables input field comparison to your matching workflow.

**Note**

If you are creating a schema mapping to be used with the LiveRamp provider service-based matching technique:

- You can specify the **Input type** as **LiveRamp ID**.
- You can specify the **name** field as either multiple fields (such as **first_name**, **last_name**) or in one field.
- You can specify the **street address** field as either multiple fields (such as **address1**, **address2**) or in one field.

If matching against an address, a zip code is required.
- You can include email or phone with name, and those fields can be matched against the street address.

b. Choose **Next**.

6. For **Step 3: Group data**:

a. Choose the related **Name** fields, and then enter the **Group name** and **Match key**.

**Example**

For example, choose input fields **First name**, **Middle name**, and **Last name**, and then enter a **Group name** called “**Full name**” and a **Match key** called “**Full name**” to enable the comparison.

b. Choose the related **Address** fields, and then enter the **Group name** and **Match key**.

**Example**

For example, choose input fields **Home street address 1**, **Home street address 2**, and **Home city**, and then enter a **Group name** called “**Shipping address**” and a **Match key** called “**Shipping address**” to enable the comparison.

c. Choose the related **Phone number** fields, and then enter the **Group name** and **Match key**.

**Example**

For example, choose input fields **Home phone 1**, **Home phone 2**, and **Cell phone**, and then enter a **Group name** called “**Shipping phone number**” and a **Match key** called “**Shipping phone number**” to enable the comparison.

If you have more than one type of data, you can add more groups.

d. Choose **Next**.

7. For **Step 4: Review and create**:

a. Review the selections that you made for the previous steps and edit if necessary.
Create a schema mapping (manually defined columns)

This procedure describes the process of creating a schema mapping using the Build custom schema option on the AWS Entity Resolution console. Use this creation method to manually define the input fields using a guided flow.

To create schema mapping using manually defined columns

1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Data preparation, choose Schema mappings.
3. On the Schema mappings page, in the upper right corner, choose Create schema mapping.
4. For Step 1: Specify schema details, do the following:
   a. For name and creation method, enter a Schema mapping name and an optional Description.
   b. For Creation method, choose Build custom schema.
   c. For Unique ID, enter a unique ID to identify each row of your data.
      Example
      For example: Primary_key, Row_ID, or Record_ID.
      Note
      The Unique ID column is required. The Unique ID must be a unique identifier across all sources. If the Unique ID is not specified, is not unique within the same source or across multiple sources, or overlaps across sources, then the record will be rejected when the matching workflow is run.
   d. (Optional) If you want to enable Tags for the resource, choose Add new tag, and then enter the Key and Value pair.
   e. Choose Next.
5. For Step 2: Map input fields:
   a. For Input fields for matching, add the Input field, Input type, and Match key.
      You can add up to 25 input fields.
      The Input type helps you classify the data. The Match key enables input field comparison to your matching workflow.
      Note
      If you are creating a schema mapping to be used with the LiveRamp provider service-based matching technique, you can specify the Input type as LiveRamp ID. If you want to include PII data in the output, you must specify the Input type as Custom string.
   b. (Optional) For Input fields for pass through, add the input fields that will not be matched.
   c. Choose Next.
6. For **Step 3: Group data**:

   a. Choose the related **Name** fields, and then enter the **Group name** and **Match key**.

   **Example**

   For example, choose input fields **First name, Middle name, and Last name**, and then enter a **Group name** called “**Full name**” and a **Match key** called “**Full name**” to enable the comparison.

   b. Choose the related **Address** fields, and then enter the **Group name** and **Match key**.

   **Example**

   For example, choose input fields **Home street address 1, Home street address 2, and Home city**, and then enter a **Group name** called “**Shipping address**” and a **Match key** called “**Shipping address**” to enable the comparison.

   c. Choose the related **Phone number** fields, and then enter the **Group name** and **Match key**.

   **Example**

   For example, choose input fields **Home phone 1, Home phone 2, and Cell phone**, and then enter a **Group name** called “**Shipping phone number**” and a **Match key** called “**Shipping phone number**” to enable the comparison.

   If you have more than one type of data, you can add more groups.

    d. Choose **Next**.

7. For **Step 4: Review and create**:

   a. Review the selections that you made for the previous steps and edit if necessary.

   b. Choose **Create schema mapping**.

   **Note**

   You can’t modify a schema mapping after you associate it to a workflow. You can clone a schema mapping if you want to use an existing configuration to create a new schema mapping.

Create a schema mapping (JSON editor)

This procedure describes the process of creating a schema mapping using the **Use JSON editor** option on the **AWS Entity Resolution console**. Use this creation method to use a JSON editor to create, paste, or import a schema mapping. The **Unique ID** and **Input fields** are not available with this option.

**To create schema mapping using the JSON editor**

1. Sign in to the AWS Management Console and open the **AWS Entity Resolution console** with your AWS account (if you have not yet done so).

2. In the left navigation pane, under **Data preparation**, choose **Schema mappings**.

3. On the **Schema mappings** page, in the upper right corner, choose **Create schema mapping**.

4. For **Step 1: Specify schema details**, do the following:

   a. For name and creation method, enter a **Schema mapping name** and an optional **Description**.

   b. For **Creation method**, choose **Use JSON editor**.

   c. (Optional) If you want to enable **Tags** for the resource, choose **Add new tag**, and then enter the **Key** and **Value** pair.
d. Choose Next.

5. For Step 2: Specify mapping:
   a. Start building the schema in the JSON editor or choose one of the following options:

<table>
<thead>
<tr>
<th>If you want to ...</th>
<th>Then choose...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start building your schema mapping</td>
<td>Insert sample JSON and then edit the information as necessary.</td>
</tr>
<tr>
<td>Use an existing JSON file</td>
<td>Import from file</td>
</tr>
</tbody>
</table>

b. Choose Next.

6. For Step 3: Review and create:
   a. Review the selections that you made for the previous steps and edit if necessary.
   b. Choose Create schema mapping.

   Note
   You can't modify a schema mapping after you associate it to a workflow. You can clone a schema mapping if you want to use an existing configuration to create a new schema mapping.

Next steps

You are now ready to:

- Create a matching workflow (p. 20)
- Create an ID mapping workflow (p. 41)
Creating a matching workflow

After you create a schema mapping, you can create one or more matching workflows to specify data inputs, normalization steps, and choose your desired matching techniques. There are three matching techniques:

- **Rule-based matching** (p. 95) is a hierarchical set of waterfall matching rules, suggested by AWS Entity Resolution, based upon the data that you input and is completely configurable by you.
- **Machine learning-based matching** (p. 90) is a preset process that will attempt to match records across all of the data that you input.
- **Provider services** (p. 95) enables you to match your known identifiers with your preferred data service provider.

AWS Entity Resolution currently integrates with the following data service providers: LiveRamp, TransUnion, and UID 2.0. You can use a public subscription for these providers on AWS Data Exchange or negotiate a private offer directly with the data provider. For more information, see **Subscribe to a provider service on AWS Data Exchange** (p. 6).

AWS Entity Resolution reads your data from the location(s) specified by you and writes results to a location that you choose. You can use AWS Entity Resolution to hash output data if desired – helping you maintain control over your data.

You can also use the output of rule-based or ML matching as an input to provider service-based matching or the other way around to meet your business needs. For example, you can run rule-based matching to find matches on your data first and then send a subset of unmatched records to provider service-based matching to save provider subscription costs.

**Topics**
- Create a rule-based matching workflow (p. 20)
- Create a machine learning-based matching workflow (p. 24)
- Create a provider service-based matching workflow (p. 27)

Create a rule-based matching workflow

The rule-based matching workflow enables you to compare cleartext or hashed data to find exact matches based on criteria that you customize.

When AWS Entity Resolution finds a match between two or more records in your data, it assigns a **Match ID** (p. 90) to the records in the matched set of data.

For rule-based matching, it applies the **rule number** (p. 91) that generated the match.

**To create a rule-based matching workflow:**

1. Sign in to the AWS Management Console and open the **AWS Entity Resolution console** with your AWS account (if you have not yet done so).
2. In the left navigation pane, under **Workflows**, choose **Matching**.
3. On the Matching workflows page, in the upper right corner, choose Create matching workflow.

4. For Step 1: Specify matching workflow details, do the following:
   a. Enter a Matching workflow name and an optional Description.
   b. For Data input, choose an AWS Glue database from the dropdown, select the AWS Glue table, and then the corresponding Schema mapping.

      You can add up to 19 data inputs.
   c. The Normalize data option is selected by default, so that data inputs are normalized before matching. If you don't want to normalize data, deselect the Normalize data option.
   d. Specify the Service access permissions by selecting either Create and use a new service role or Use an existing service role.

<table>
<thead>
<tr>
<th>If you choose...</th>
<th>Then...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create and use a new service role</td>
<td>AWS Entity Resolution creates a service role with the required policy for this table.</td>
</tr>
<tr>
<td></td>
<td>The default Service role name is entityresolution-matching-workflow-&lt;timestamp&gt;</td>
</tr>
<tr>
<td></td>
<td>You must have permissions to create roles and attach policies.</td>
</tr>
<tr>
<td></td>
<td>If your input data is encrypted, you can choose the My data input is encrypted by a KMS key option and then enter an AWS KMS key that will be used to decrypt your data input.</td>
</tr>
<tr>
<td>Use an existing service role</td>
<td>Choose an Existing service role name from the dropdown list.</td>
</tr>
<tr>
<td></td>
<td>The list of roles are displayed if you have permissions to list roles.</td>
</tr>
<tr>
<td></td>
<td>If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use.</td>
</tr>
<tr>
<td></td>
<td>If there are no existing service roles, the option to Use an existing service role is unavailable.</td>
</tr>
<tr>
<td></td>
<td>By default, AWS Entity Resolution does not attempt to update the existing role policy to add necessary permissions.</td>
</tr>
</tbody>
</table>

   e. (Optional) To enable Tags for the resource, choose Add new tag, and then enter the Key and Value pair.
   f. Choose Next.

5. For Step 2: Choose matching technique:
   a. For Matching method, choose Rule-based matching.
b. For **Processing cadence**, choose one of the following.

<table>
<thead>
<tr>
<th>If you want to...</th>
<th>Then choose...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run a workflow on demand for a bulk update</td>
<td>Manual</td>
</tr>
<tr>
<td>Run a workflow as soon as new data is in your S3 bucket</td>
<td>Automatic</td>
</tr>
</tbody>
</table>

c. For **Matching rules**, enter a **Rule name** and then choose the **Match keys** for that rule.

You can apply up to 15 different match keys across your rules to define match criteria.

You can create up to 15 rules.
d. For **Comparison type**, choose one of the following.

<table>
<thead>
<tr>
<th>If you want to...</th>
<th>Then choose...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find any combination of matches across data stored in multiple input fields</td>
<td><strong>Multiple input field comparison</strong></td>
</tr>
<tr>
<td>Limit comparison to a single input field</td>
<td><strong>Single input field comparison</strong></td>
</tr>
</tbody>
</table>

![Comparison type](image)

Choose Next.

6. For **Step 3: Specify data output and format**:
   a. For **Data output destination and format**, choose the Amazon S3 location for the data output and whether the **Data format** will be **Normalized data** or **Original data**.
   b. For **Encryption**, if you choose to **Customize encryption settings**, enter the **AWS KMS key ARN**.
   c. View the **System generated output**.
   d. For **Data output**, view all of the fields that are included.
   e. Determine if you want to include, hide, or mask fields.

<table>
<thead>
<tr>
<th>If you want to...</th>
<th>Then choose...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include fields</td>
<td>Keep the output state as <strong>Included</strong>.</td>
</tr>
<tr>
<td>Hide fields (exclude from output)</td>
<td>Choose the <strong>Output field</strong>, and then choose <strong>Hide</strong>.</td>
</tr>
<tr>
<td>Mask fields</td>
<td>Choose the <strong>Output field</strong>, and then choose <strong>Hash output</strong>.</td>
</tr>
<tr>
<td>Reset the previous settings</td>
<td>Choose <strong>Reset</strong>.</td>
</tr>
</tbody>
</table>

Choose Next.

7. For **Step 4: Review and create**:
   a. Review the selections that you made for the previous steps and edit if necessary.
   b. Choose **Create and run**.
A message appears, indicating that the matching workflow has been created and that the job has started.

8. On the matching workflow details page, on the **Metrics** tab, view the following under **Last job metrics**:

   - The **Job ID**.
   - The **Status** of the matching workflow job: Queued, In progress, Completed, Failed
   - The **Time completed** for the workflow job.
   - The number of **Records processed**.
   - The number of **Records not processed**.
   - The **Unique match IDs generated**.
   - The number of **Input records**.

   You can also view the job metrics for matching workflow jobs that have been previously run under the **Job history**.

9. After the matching workflow job completes (**Status** is Completed), you can go to the **Data output** tab and then select your **Amazon S3 location** to view the results.

You are now ready to:

- **Edit a matching workflow** (p. 50)
- **Delete a matching workflow** (p. 51)
- **Run a matching workflow** (p. 50)

**Create a machine learning-based matching workflow**

The machine learning-based matching workflow enables you to compare cleartext data to find a broad range of matches using a machine learning model.

**Note**

The machine learning model does not support the comparison of hashed data.

When AWS Entity Resolution finds a match between two or more records in your data, it assigns a **Match ID** (p. 90) to the records in the matched set of data.

For machine learning-based matching, it applies the match **confidence level** (p. 88) percentage.

**To create a ML-based matching workflow:**

1. Sign in to the AWS Management Console and open the **AWS Entity Resolution console** with your AWS account (if you have not yet done so).
2. In the left navigation pane, under **Workflows**, choose **Matching**.
3. On the **Matching workflows** page, in the upper right corner, choose **Create matching workflow**.
4. For **Step 1: Specify matching workflow details**, do the following:
   a. Enter a **Matching workflow name** and an optional **Description**.
   b. For **Data input**, choose an **AWS Glue database** from the dropdown, select the **AWS Glue table**, and then the corresponding **Schema mapping**.
You can add up to 20 data inputs.

c. The **Normalize data** option is selected by default, so that data inputs are normalized before matching. If you don't want to normalize data, deselect the **Normalize data** option.

d. Specify the **Service access** permissions by selecting either **Create and use a new service role** or **Use an existing service role**.

<table>
<thead>
<tr>
<th>If you choose...</th>
<th>Then...</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Create and use a new service role</strong></td>
<td>AWS Entity Resolution creates a service role with the required policy for this table. The default <strong>Service role name</strong> is <code>entityresolution-matching-workflow-&lt;timestamp&gt;</code> You must have permissions to create roles and attach policies. If your input data is encrypted, you can choose the <strong>My data input is encrypted by a KMS key</strong> option and then enter an <strong>AWS KMS key</strong> that will be used to decrypt your data input.</td>
</tr>
<tr>
<td><strong>Use an existing service role</strong></td>
<td>Choose an <strong>Existing service role name</strong> from the dropdown list. The list of roles are displayed if you have permissions to list roles. If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use. If there are no existing service roles, the option to <strong>Use an existing service role</strong> is unavailable. By default, AWS Entity Resolution does not attempt to update the existing role policy to add necessary permissions.</td>
</tr>
</tbody>
</table>

e. (Optional) To enable **Tags** for the resource, choose **Add new tag**, and then enter the **Key** and **Value** pair.

f. Choose **Next**.

5. For **Step 2: Choose matching technique**:

a. For **Matching method**, choose **Machine learning-based matching**.
b. For Processing cadence, the Manual option is selected. This option enables you to run a workflow on demand for a bulk update.

c. Choose Next.

6. For Step 3: Specify data output and format:

a. For Data output destination and format, choose the Amazon S3 location for the data output and whether the Data format will be Normalized data or Original data.

b. For Encryption, if you choose to Customize encryption settings, enter the AWS KMS key ARN.

c. View the System generated output.

d. For Data output, view all of the fields that are included.

e. Determine if you want to include, hide, or mask fields.

<table>
<thead>
<tr>
<th>If you want to...</th>
<th>Then choose...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include fields</td>
<td>Keep the output state as Included.</td>
</tr>
<tr>
<td>Hide fields (exclude from output)</td>
<td>Choose the Output field, and then choose Hide.</td>
</tr>
<tr>
<td>Mask fields</td>
<td>Choose the Output field, and then choose Hash output.</td>
</tr>
<tr>
<td>Reset the previous settings</td>
<td>Choose Reset.</td>
</tr>
</tbody>
</table>
f. Choose **Next**.

7. For **Step 4: Review and create**:
   a. Review the selections that you made for the previous steps and edit if necessary.
   b. Choose **Create and run**.

   A message appears, indicating that the matching workflow has been created and that the job has started.

8. On the matching workflow details page, on the **Metrics** tab, view the following under **Last job metrics**:
   - The **Job ID**.
   - The **Status** of the matching workflow job: **Queued**, **In progress**, **Completed**, **Failed**
   - The **Time completed** for the workflow job.
   - The number of **Records processed**.
   - The number of **Records not processed**.
   - The **Unique match IDs generated**.
   - The number of **Input records**.

   You can also view the job metrics for matching workflow jobs that have been previously run under the **Job history**.

9. After the matching workflow job completes (**Status is Completed**), you can go to the **Data output** tab and then select your **Amazon S3 location** to view the results.

You are now ready to:

- **Edit a matching workflow** (p. 50)
- **Delete a matching workflow** (p. 51)
- **Run a matching workflow** (p. 50)

---

### Create a provider service-based matching workflow

If you have a subscription with a provider service through AWS Data Exchange, you can match your known identifiers with your preferred provider. AWS Entity Resolution currently supports the following data provider services:

- LiveRamp
- TransUnion
- Unified ID 2.0

For more information about creating a new subscription or reusing an existing subscription to a provider service, see **Subscribe to a provider service on AWS Data Exchange** (p. 6).

The following sections describe how to create a provider-based matching workflow.

**Topics**

- **Creating a matching workflow with LiveRamp** (p. 28)
- **Creating a matching workflow with TransUnion** (p. 33)
Creating a matching workflow with LiveRamp

If you have a subscription to the LiveRamp service, you can create a matching workflow with the LiveRamp service to perform identity resolution.

The LiveRamp service provides a identifier called the Ramp ID (RAMPID). The RAMPID is one of the most commonly used IDs in demand-side platforms to create an audience for an advertising campaign. Using a matching workflow with LiveRamp, you can resolve hashed email addresses to RAMPIDs.

This workflow requires an Amazon S3 data staging bucket where you want the matching workflow output to be temporarily written. Before you create a ID mapping workflow with LiveRamp, add the following permissions to the data staging bucket.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::715724997226:root"
      },
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3::<staging-bucket>",
        "arn:aws:s3::<staging-bucket>/*"
      ]
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::715724997226:root"
      },
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation",
        "s3:GetBucketPolicy",
        "s3:ListBucketVersions",
        "s3:GetBucketAcl"
      ],
      "Resource": [
        "arn:aws:s3::<staging-bucket>",
        "arn:aws:s3::<staging-bucket>/*"
      ]
    }
  ]
}
```

Replace each `<user input placeholder>` with your own information.

`staging-bucket` Amazon S3 bucket that temporarily stores your data while running a provider service-based workflow.
To create a matching workflow with LiveRamp:

1. Sign in to the AWS Management Console and open the [AWS Entity Resolution console](https://aws.amazon.com/console/) with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Workflows, choose Matching.
3. On the Matching workflows page, in the upper right corner, choose Create matching workflow.
4. For **Step 1: Specify matching workflow details**, do the following:
   a. Enter a **Matching workflow name** and an optional **Description**.
   b. For **Data input**, choose an **AWS Glue database** from the dropdown, select the **AWS Glue table**, and then select the corresponding **Schema mapping**.

You can add up to 20 data inputs.

c. The **Normalize data** option is selected by default, so that data inputs are normalized before matching.

If you are using the email-only resolution process, deselect the **Normalize data** option, because only hashed emails are used for input data.

d. Specify the **Service access** permissions by selecting either **Create and use a new service role** or **Use an existing service role**.

<table>
<thead>
<tr>
<th>If you choose...</th>
<th>Then...</th>
</tr>
</thead>
</table>
| **Create and use a new service role** | AWS Entity Resolution creates a service role with the required policy for this table.  

The default **Service role name** is `entityresolution-matching-workflow-<timestamp>`  

You must have permissions to create roles and attach policies.  

If your input data is encrypted, you can choose the **My data inputs are encrypted by a KMS key** option and then enter an **AWS KMS key** that will be used to decrypt your data input. |
| **Use an existing service role**   | Choose an **Existing service role name** from the dropdown list.  

The list of roles are displayed if you have permissions to list roles.  

If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use.  

If there are no existing service roles, the option to **Use an existing service role** is unavailable.  

By default, AWS Entity Resolution does not attempt to update the existing role policy to add necessary permissions. |
e. (Optional) To enable Tags for the resource, choose Add new tag, and then enter the Key and Value pair.

f. Choose Next.

5. For Step 2: Choose matching technique:

a. For Matching method, choose Provider services.

b. Choose LiveRamp.
Note
Ensure that your data input file format and normalization is aligned with the provider service's guidelines.
For more information about input file formatting guidelines for the matching workflow, see Perform Identity Resolution Through ADX in the LiveRamp documentation.

c. For LiveRamp configuration, enter a Client ID manager ARN and a Client secret manager ARN.
d. For Data staging, choose the Amazon S3 location for the temporary storage of your data while it processes.
You must have permission to the data staging Amazon S3 location. For more information, see the section called “Create a workflow job role for AWS Entity Resolution” (p. 11).
e. Choose Next.

6. For Step 3: Specify data output:

a. For Data output destination and format, choose the Amazon S3 location for the data output and whether the Data format will be Normalized data or Original data.
b. For Encryption, if you choose to Customize encryption settings, enter the AWS KMS key ARN.
c. View the LiveRamp generated output.
This is the additional information generated by LiveRamp.
d. For Data output, view all of the fields that are included and determine if you want to include, hide, or mask fields.

Note
If you have chosen LiveRamp, due to LiveRamp privacy filters that remove PII, some fields will display an Output state of Unavailable.

<table>
<thead>
<tr>
<th>If you want to...</th>
<th>Then choose...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include fields</td>
<td>Keep the output state as Included.</td>
</tr>
<tr>
<td>Hide fields (exclude from output)</td>
<td>Choose the Output field, and then choose Hide.</td>
</tr>
<tr>
<td>Mask fields</td>
<td>Choose the Output field, and then choose Hash output.</td>
</tr>
<tr>
<td>Reset the previous settings</td>
<td>Choose Reset.</td>
</tr>
</tbody>
</table>
7. For Step 4: Review and create:
   a. Review the selections that you made for the previous steps and edit if necessary.
   b. Choose Create and run.

   A message appears, indicating that the matching workflow has been created and that the job has started.

8. On the matching workflow details page, on the Metrics tab, view the following under Last job metrics:

   - The Job ID.
   - The Status of the matching workflow job: Queued, In progress, Completed, Failed
   - The Time completed for the workflow job.
   - The number of Records processed.
   - The number of Records not processed.
   - The Unique match IDs generated.
   - The number of Input records.

   You can also view the job metrics for matching workflow jobs that have been previously run under the Job history.
9. After the matching workflow job completes (Status is Completed), you can go to the Data output tab and then select your Amazon S3 location to view the results.

You are now ready to:

- Edit a matching workflow (p. 50)
- Delete a matching workflow (p. 51)
- Run a matching workflow (p. 50)

Creating a matching workflow with TransUnion

If you have a subscription to the TransUnion service, you can improve customer understanding by linking, matching, and enhancing customer-related records stored across disparate channels with TransUnion Person and Household E Keys and over 200 data attributes.

The TransUnion service provides identifiers known as the TransUnion Individual and Household IDs. TransUnion provides ID assignment (also known as encoding) of known identifiers such as name address, phone number, and email address.

This workflow requires an Amazon S3 data staging bucket where you want the matching workflow output to be temporarily written. Before you create a matching workflow with TransUnion, add the following permissions to the data staging bucket.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::103054336026:root"
         },
         "Action": [
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject"
         ],
         "Resource": [
            "arn:aws:s3:::<staging-bucket>",
            "arn:aws:s3:::<staging-bucket>/*"
         ]
      },
      {
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::103054336026:root"
         },
         "Action": [
            "s3:ListBucket",
            "s3:GetBucketLocation",
            "s3:GetBucketPolicy",
            "s3:ListBucketVersions",
            "s3:GetBucketAcl"
         ],
         "Resource": [
            "arn:aws:s3:::<staging-bucket>",
            "arn:aws:s3:::<staging-bucket>/*"
         ]
      }
   ]
}
```
To create a matching workflow with TransUnion:

1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Workflows, choose Matching.
3. On the Matching workflows page, in the upper right corner, choose Create matching workflow.
4. For Step 1: Specify matching workflow details, do the following:
   a. Enter a Matching workflow name and an optional Description.
   b. For Data input, choose an AWS Glue database from the dropdown, select the AWS Glue table, and then select the corresponding Schema mapping. You can add up to 20 data inputs.
   c. The Normalize data option is selected by default, so that data inputs are normalized before matching. If you don't want to normalize data, deselect the Normalize data option.
   d. Specify the Service access permissions by selecting either Create and use a new service role or Use an existing service role.

<table>
<thead>
<tr>
<th>If you choose...</th>
<th>Then...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create and use a new service role</td>
<td>AWS Entity Resolution creates a service role with the required policy for this table. The default Service role name is entityresolution-matching-workflow-&lt;timestamp&gt; You must have permissions to create roles and attach policies. If your input data is encrypted, you can choose the My data inputs are encrypted by a KMS key option and then enter an AWS KMS key that will be used to decrypt your data input.</td>
</tr>
<tr>
<td>Use an existing service role</td>
<td>Choose an Existing service role name from the dropdown list. The list of roles are displayed if you have permissions to list roles. If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use.</td>
</tr>
<tr>
<td>If you choose...</td>
<td>Then...</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------</td>
</tr>
<tr>
<td>If there are no existing service roles, the option to <strong>Use an existing service role</strong> is unavailable.</td>
<td></td>
</tr>
<tr>
<td>By default, AWS Entity Resolution does not attempt to update the existing role policy to add necessary permissions.</td>
<td></td>
</tr>
</tbody>
</table>

**e.** (Optional) To enable **Tags** for the resource, choose **Add new tag**, and then enter the **Key** and **Value** pair.

**f.** Choose **Next**.

5. **For Step 2: Choose matching technique:**
   a. For **Matching method**, choose **Provider services**.
   b. Choose **TransUnion**.

**Note**
Ensure that your data input file format and normalization is aligned with the provider service’s guidelines.

**c.** For **TransUnion products**, select a product from the dropdown list.
For Data staging, choose the Amazon S3 location for the temporary storage of your data while it processes.

You must have permission to the data staging Amazon S3 location. For more information, see the section called “Create a workflow job role for AWS Entity Resolution” (p. 11).

6. Choose Next.

7. For Step 3: Specify data output:
   a. For Data output destination and format, choose the Amazon S3 location for the data output and whether the Data format will be Normalized data or Original data.
   b. For Encryption, if you choose to Customize encryption settings, enter the AWS KMS key ARN.
   c. View the TransUnion generated output.

   This is the additional information generated by TransUnion.

   d. For Data output, view all of the fields that are included and determine if you want to include, hide, or mask fields.

<table>
<thead>
<tr>
<th>If you want to...</th>
<th>Then choose...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include fields</td>
<td>Keep the output state as Included.</td>
</tr>
<tr>
<td>Hide fields (exclude from output)</td>
<td>Choose the Output field, and then choose Hide.</td>
</tr>
<tr>
<td>Mask fields</td>
<td>Choose the Output field, and then choose Hash output.</td>
</tr>
<tr>
<td>Reset the previous settings</td>
<td>Choose Reset.</td>
</tr>
</tbody>
</table>

   e. For System generated output, view all of the fields that are included.
   f. Choose Next.

8. For Step 4: Review and create:
   a. Review the selections that you made for the previous steps and edit if necessary.
   b. Choose Create and run.

   A message appears, indicating that the matching workflow has been created and that the job has started.

9. On the matching workflow details page, on the Metrics tab, view the following under Last job metrics:
   - The Job ID.
   - The Status of the matching workflow job: Queued, In progress, Completed, Failed
   - The Time completed for the workflow job.
   - The number of Records processed.
   - The number of Records not processed.
   - The Unique match IDs generated.
   - The number of Input records.

   You can also view the job metrics for matching workflow jobs that have been previously run under the Job history.

10. After the matching workflow job completes (Status is Completed), you can go to the Data output tab and then select your Amazon S3 location to view the results.
Creating a matching workflow with UID 2.0

If you have a subscription to the Unified ID 2.0 service, you can activate advertising campaigns with deterministic identity and lean on interoperability with many UID2-enabled participants across the advertising ecosystem. For more information, see Unified ID 2.0 Overview.

The Unified ID 2.0 service provides raw UID 2, which is used for building advertising campaigns in The Trade Desk platform. UID 2.0 is generated using an open source framework.

In one workflow you can use either Email Address or Phone number for raw UID2 generation but not both. If both are present in the schema mapping, then the workflow will pick the Email Address and the Phone number will be a pass-through field. To support both, create a new schema mapping where Phone number is mapped but Email Address is not. Then, create a second workflow using this new schema mapping.

**Note**

Raw UID2s are created by adding salts from salt buckets which are rotated approximately once a year, causing the raw UID2 to also be rotated with it, so it is recommended that you refresh the raw UID2s daily. For more information, see https://unifiedid.com/docs/getting-started/gs-faqs#how-often-should-uid2s-be-refreshed-for-incremental-updates

To create a matching workflow with UID 2.0:

1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Workflows, choose Matching.
3. On the Matching workflows page, in the upper right corner, choose Create matching workflow.
4. For Step 1: Specify matching workflow details, do the following:
   a. Enter a Matching workflow name and an optional Description.
   b. For Data input, choose an AWS Glue database from the dropdown, select the AWS Glue table, and then select the corresponding Schema mapping.

   You can add up to 20 data inputs.
   c. Leave the Normalize data option is selected, so that data inputs (Email Address or Phone number) are normalized before matching.

   For more information about Email Address normalization, see Email Address Normalization in the UID 2.0 documentation.

   For more information about Phone number normalization, see Phone Number Normalization in the UID 2.0 documentation.
   d. Specify the Service access permissions by selecting either Create and use a new service role or Use an existing service role.

<table>
<thead>
<tr>
<th>If you choose...</th>
<th>Then...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create and use a new service role</td>
<td>AWS Entity Resolution creates a service role with the required policy for this table.</td>
</tr>
<tr>
<td>If you choose…</td>
<td>Then…</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>The default Service role name is entityresolution-matching-workflow-&lt;timestamp&gt;</td>
</tr>
<tr>
<td></td>
<td>You must have permissions to create roles and attach policies.</td>
</tr>
<tr>
<td></td>
<td>If your input data is encrypted, you can choose the My data inputs are encrypted by a KMS key option and then enter an AWS KMS key that will be used to decrypt your data input.</td>
</tr>
<tr>
<td>Use an existing service role</td>
<td>Choose an Existing service role name from the dropdown list.</td>
</tr>
<tr>
<td></td>
<td>The list of roles are displayed if you have permissions to list roles.</td>
</tr>
<tr>
<td></td>
<td>If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use.</td>
</tr>
<tr>
<td></td>
<td>If there are no existing service roles, the option to Use an existing service role is unavailable.</td>
</tr>
<tr>
<td></td>
<td>By default, AWS Entity Resolution does not attempt to update the existing role policy to add necessary permissions.</td>
</tr>
</tbody>
</table>

e. (Optional) To enable Tags for the resource, choose Add new tag, and then enter the Key and Value pair.

f. Choose Next.

5. For Step 2: Choose matching technique:
   a. For Matching method, choose Provider services.
   b. Choose Unified ID 2.0.
c. Choose Next.

6. For **Step 3: Specify data output**:
   
a. For **Data output destination and format**, choose the Amazon S3 location for the data output and whether the **Data format** will be **Normalized data** or **Original data**.

b. For **Encryption**, if you choose to **Customize encryption settings**, enter the AWS KMS key ARN.

c. View the **Unified ID 2.0 generated output**.

   This is a list of all of the additional information generated by UID 2.0

d. For **Data output**, view all of the fields that are included and determine if you want to include, hide, or mask fields.

<table>
<thead>
<tr>
<th>If you want to...</th>
<th>Then choose...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include fields</td>
<td>Keep the output state as <strong>Included</strong>.</td>
</tr>
<tr>
<td>Hide fields (exclude from output)</td>
<td>Choose the <strong>Output field</strong>, and then choose <strong>Hide</strong>.</td>
</tr>
<tr>
<td>Mask fields</td>
<td>Choose the <strong>Output field</strong>, and then choose <strong>Hash output</strong>.</td>
</tr>
</tbody>
</table>
If you want to... | Then choose...
---|---
Reset the previous settings | Choose Reset.

e. For **System generated output**, view all of the fields that are included.
f. Choose **Next**.

7. For **Step 4: Review and create**:
   - Review the selections that you made for the previous steps and edit if necessary.
   - Choose **Create and run**.
     
     A message appears, indicating that the matching workflow has been created and that the job has started.

8. On the matching workflow details page, on the **Metrics** tab, view the following under **Last job metrics**:
   - The **Job ID**.
   - The **Status** of the matching workflow job: **Queued, In progress, Completed, Failed**
   - The **Time completed** for the workflow job.
   - The number of **Records processed**.
   - The number of **Records not processed**.
   - The **Unique match IDs generated**.
   - The number of **Input records**.

   You can also view the job metrics for matching workflow jobs that have been previously run under the **Job history**.

9. After the matching workflow job completes (**Status is Completed**), you can go to the **Data output** tab and then select your **Amazon S3 location** to view the results.

You are now ready to:

- **Edit a matching workflow** (p. 50)
- **Delete a matching workflow** (p. 51)
- **Run a matching workflow** (p. 50)
Create and run an ID mapping workflow

The ID mapping workflow in AWS Entity Resolution is currently integrated with LiveRamp. If you have a subscription to the LiveRamp service, you can create an ID mapping workflow with the LiveRamp service to perform transcoding. With LiveRamp transcoding, you can translate a set of source Ramp IDs into any destination Ramp ID. This removes the need for you to share customer data directly with advertising platforms by using the Ramp ID as a token to represent your customers.

For more information, see Perform Transcoding Through ADX.

Topics
- Creating an ID mapping workflow (p. 41)
- Running an ID mapping workflow (p. 46)

Creating an ID mapping workflow

After you complete the set up steps (p. 5) and create a schema mapping (p. 15), you can create one or more ID mapping workflows to translate a set of source Ramp IDs to another using either maintained or derived Ramp IDs.

This ID mapping workflow requires an Amazon S3 data staging bucket where you want the ID mapping workflow output to be temporarily written. Before you create a ID mapping workflow with LiveRamp, add the following permissions to the data staging bucket.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::715724997226:root"
      },
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3::<staging-bucket>",
        "arn:aws:s3::<staging-bucket>/*"
      ]
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::715724997226:root"
      },
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3::<staging-bucket> /*"
      ]
    }
  ]
}
```
To create an ID mapping workflow:

1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Workflows, choose ID mapping.
3. On the ID mapping workflows page, in the upper right corner, choose Create ID mapping workflow.
4. For Step 1: Specify ID mapping workflow details, do the following:
   a. Enter an ID mapping workflow name and an optional Description.
   b. View the ID mapping method.

   AWS Entity Resolution currently offers the LiveRamp provider service as an ID mapping method.

   If you have a subscription to LiveRamp, the status is listed as Subscribed.

   For more information on how to subscribe to LiveRamp, see Subscribe to a provider service on AWS Data Exchange (p. 6).
Note

Ensure that your data input file format is aligned with the provider service's guidelines. For more information about LiveRamp's input file formatting guidelines, see Perform Transcoding Through ADX in the LiveRamp documentation.

c. For LiveRamp configuration, enter the following ids, provided by LiveRamp:

- Client ID manager ARN
- Client secret manager ARN
- Target domain

d. For Data staging, select the Amazon S3 location where you want to ID mapping workflow output to be written temporarily.

e. (Optional) To enable Tags for the resource, choose Add new tag, and then enter the Key and Value pair.

f. Choose Next.
5. For **Step 2: Specify data input:**

   a. For **Data input**, choose an **AWS Glue** database from the dropdown, select the **AWS Glue table**, and then select the corresponding **Schema mapping**.

      You can add up to 19 data inputs.

   b. Specify the **Service access** permissions by selecting either **Create and use a new service role** or **Use an existing service role**.

<table>
<thead>
<tr>
<th>If you choose...</th>
<th>Then...</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Create and use a new service role</strong></td>
<td>AWS Entity Resolution creates a service role with the required policy for this table.</td>
</tr>
<tr>
<td></td>
<td>The default Service role name is <code>entityresolution-id-mapping-workflow-&lt;timestamp&gt;</code></td>
</tr>
<tr>
<td></td>
<td>You must have permissions to create roles and attach policies.</td>
</tr>
<tr>
<td></td>
<td>If your input data is encrypted, you can choose the <strong>My data inputs are encrypted by a KMS key</strong> option and then enter an <strong>AWS KMS key</strong> that will be used to decrypt your data input.</td>
</tr>
<tr>
<td><strong>Use an existing service role</strong></td>
<td>Choose an <strong>Existing service role name</strong> from the dropdown list.</td>
</tr>
<tr>
<td></td>
<td>The list of roles are displayed if you have permissions to list roles.</td>
</tr>
<tr>
<td></td>
<td>If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use.</td>
</tr>
<tr>
<td></td>
<td>If there are no existing service roles, the option to <strong>Use an existing service role</strong> is unavailable.</td>
</tr>
</tbody>
</table>
By default, AWS Entity Resolution does not attempt to update the existing role policy to add necessary permissions.

6. Choose **Next**.

7. **For Step 3: Specify data output:**
   a. For **Data output destination**, choose the **Amazon S3 location** for the data output.
   b. For **Encryption**, if you choose to **Customize encryption settings**, enter the **AWS KMS key ARN**.
   c. View the **LiveRamp generated output**.
   d. Choose **Next**.
8. **For Step 4: Review and create:**
   a. Review the selections that you made for the previous steps and edit if necessary.
   b. Choose **Create**.
   c. A message appears, indicating that the ID mapping workflow has been created.

You are now ready to:
- **Run an ID mapping workflow** (p. 46)
- **Edit an ID mapping workflow** (p. 51)
- **Delete an ID mapping workflow** (p. 51)

## Running an ID mapping workflow

**To run an ID mapping workflow:**

1. Sign in to the AWS Management Console and open the [AWS Entity Resolution console](https://aws.amazon.com/entity-resolution/) with your AWS account (if you have not yet done so).
2. In the left navigation pane, under **Workflows**, choose **ID mapping**.
3. Choose the ID mapping workflow.
4. On the ID mapping workflow details page, in the upper right corner, choose **Run**.
5. On the matching workflow details page, on the **Metrics** tab, view the following under **Last job metrics**:
   - The **Job ID**.
   - The **Status** of the matching workflow job: **Queued**, **In progress**, **Completed**, **Failed**
6. After the ID mapping workflow job completes (status is Completed), you can go to the Data output tab and then select your Amazon S3 location to view the results.

After you get your CSV file, you can join the RAMPID with the TRANSCODED_ID.
Managing AWS Entity Resolution

The following topics explain how to run and manage workflows using the AWS Entity Resolution console.

For information about how to manage AWS Entity Resolution using the AWS SDKs, see the AWS Entity Resolution API Reference.

Topics
- Managing schema mappings (p. 48)
- Managing matching workflows (p. 49)
- Managing ID mapping workflows (p. 51)
- Troubleshooting workflows (p. 51)

Managing schema mappings

Topics
- Clone a schema mapping (p. 48)
- Edit a schema mapping (p. 48)
- Delete a schema mapping (p. 49)

Clone a schema mapping

You can clone a schema mapping if you want to use an existing configuration to create a new schema mapping.

To clone a schema mapping:

1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Data preparation, choose Schema mappings.
3. Choose the schema mapping.
4. Choose Clone.
5. On the Specify schema details page, make any necessary changes and then choose Next.
6. On the Choose matching technique page, make any necessary changes and then choose Next.
7. On the Map input fields page, make any necessary changes and then choose Next.
8. On the Group data page, make any necessary changes and then choose Next.
9. On the Review and save page, make any necessary changes and then choose Clone schema mapping.

Edit a schema mapping

You can only edit a schema mapping before you associate it to a workflow. After you've associated a schema mapping to a workflow, you can't edit it. You can clone a schema mapping if you want to use an existing configuration to create a new schema mapping.
To edit a schema mapping:
1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Data preparation, choose Schema mappings.
3. Choose the schema mapping.
4. Choose Edit.
5. On the Specify schema details page, make any necessary changes and then choose Next.
6. On the Choose matching technique page, make any necessary changes and then choose Next.
7. On the Map input fields page, make any necessary changes and then choose Next.
8. On the Group data page, make any necessary changes and then choose Next.
9. On the Review and save page, make any necessary changes and then choose Edit schema mapping.

Delete a schema mapping

You can't delete a schema mappings when it's associated to a matching workflow. You must first remove the schema mapping from all associated matching workflows before the schema mapping you can delete it.

To delete a schema mapping:
1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Data preparation, choose Schema mappings.
3. Choose the schema mapping.
4. Choose Delete.
5. Confirm the deletion and then choose Delete.

Managing matching workflows

After you create a Rule-based matching or Machine learning-based matching workflow with the Manual processing type, you can run a matching workflow job.

Note
If you create a matching workflow with the Automatic processing type, your matching workflow jobs will run each time a data input is updated.

AWS Entity Resolution reads your data from your specified location or locations and finds a match between two or more records in your data. It then assigns a match ID to the records in the matched set of data.

- If you specified the Rule-based matching technique, AWS Entity Resolution will also assign the rule number applied that generated the match.
- If you specified the Machine learning-based matching technique, AWS Entity Resolution will also assign the match confidence level percentage.

AWS Entity Resolution then writes data output files to a location that you choose.

A workflow can have multiple runs and the results (successes or errors) are written to a folder with the jobId as the name.
Run a matching workflow

To run a matching workflow:

1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Workflows, choose Matching.
3. Choose the matching workflow.
4. On the matching workflow details page, in the upper right corner, choose Run.

A message appears, indicating that the job has started.
5. On the Metrics tab, under Job history, view the following:

   - The Status of the matching workflow job: In progress, Completed, Failed
   - The number of Records processed.
   - The number of Matches found.
   - The number of Unique records.
   - The Duration of the job.
   - The Job ID.
6. After the matching workflow job completes (Status is Completed), you can go to the Data output tab and then select your Amazon S3 location to view the results.

Edit a matching workflow

To edit a matching workflow:

1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Workflows, choose Matching.
3. Choose the matching workflow.
4. On the matching workflow details page, in the upper right corner, choose Edit.
5. On the Specify matching workflow details page, make any necessary changes and then choose Next.
6. On the Choose matching technique page, make any necessary changes and then choose Next.
7. On the Specify data output page, make any necessary changes and then choose Next.
8. On the Review and save page, make any necessary changes and then choose Save.
Delete a matching workflow

To delete a matching workflow:

1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Workflows, choose Matching.
3. Choose the matching workflow.
4. On the matching workflow details page, in the upper right corner, choose Delete.
5. Confirm the deletion and then choose Delete.

Managing ID mapping workflows

Topics
- Edit an ID mapping workflow (p. 51)
- Delete an ID mapping workflow (p. 51)

Edit an ID mapping workflow

To edit an ID mapping workflow:

1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Workflows, choose ID mapping.
3. Choose the ID mapping workflow.
4. On the ID mapping workflow details page, in the upper right corner, choose Edit.
5. On the Specify ID mapping workflow details page, make any necessary changes and then choose Next.
6. On the Specify data output page, make any necessary changes and then choose Next.
7. On the Review and save page, make any necessary changes and then choose Save.

Delete an ID mapping workflow

To delete an ID mapping workflow:

1. Sign in to the AWS Management Console and open the AWS Entity Resolution console with your AWS account (if you have not yet done so).
2. In the left navigation pane, under Workflows, choose ID mapping.
3. Choose the ID mapping workflow.
4. On the ID mapping workflow details page, in the upper right corner, choose Delete.
5. Confirm the deletion and then choose Delete.

Troubleshooting workflows

Use the following information to help you diagnose and fix common issues that you might encounter when running workflows.
I received an error file.

The records in the error file can be created for the following reasons:

- The Unique ID (p. 96) is:
  - null
  - missing in a row of data
  - missing in a record in the data table
  - repeated in another row of data in the data table
  - not specified
  - not unique within the same source
  - not unique across multiple sources
  - overlaps across sources

- One of the fields in the schema mapping (p. 96) includes a reserved name:
  - EmailAddress
  - InputSourceARN
  - MatchRule
  - MatchID
  - HashingProtocol
  - ConfidenceLevel
  - Source

If the record in the error file is created due to the reasons listed previously, you are charged, because it incurs processing cost for the service. If the record in the error file is because of an internal server error, you are not charged.
Security in AWS Entity Resolution

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from data centers and network architectures that are built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security of the cloud and security in the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to AWS Entity Resolution, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company’s requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using AWS Entity Resolution. The following topics show you how to configure AWS Entity Resolution to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your AWS Entity Resolution resources.

**Topics**
- Data protection in AWS Entity Resolution (p. 53)
- Identity and access management for AWS Entity Resolution (p. 62)
- Compliance validation for AWS Entity Resolution (p. 78)
- Resilience in AWS Entity Resolution (p. 78)

Data protection in AWS Entity Resolution

The AWS shared responsibility model applies to data protection in AWS Entity Resolution. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. This content includes the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the AWS Security Blog.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
• If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers’ email addresses, into tags or free-form text fields such as a Name field. This includes when you work with AWS Entity Resolution or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Data encryption at rest for AWS Entity Resolution

AWS Entity Resolution provides encryption by default to protect sensitive customer data at rest using AWS owned encryption keys.

**AWS owned keys** – AWS Entity Resolution uses these keys by default to automatically encrypt personally identifiable data. You can't view, manage, or use AWS owned keys, or audit their use. However, you don't have to take any action to protect the keys that encrypt your data. For more information, see [AWS owned keys](https://docs.aws.amazon.com/kms/latest/developerguide/aws-owned-keys.html) in the [AWS Key Management Service Developer Guide](https://docs.aws.amazon.com/kms/latest/developerguide/). Encryption of data at rest by default helps reduce the operational overhead and complexity involved in protecting sensitive data. At the same time, you can use it to build secure applications that meet strict encryption compliance and regulatory requirements.

Alternatively, you can also provide a customer managed KMS key for encryption when you create your matching workflow resource.

**Customer managed keys** – AWS Entity Resolution supports the use of a symmetric customer managed KMS key that you create, own, and manage to allow encryption of your sensitive data. Because you have full control of this layer of encryption, you can perform such tasks as:

• Establishing and maintaining key policies  
• Establishing and maintaining IAM policies and grants  
• Enabling and disabling key policies  
• Rotating key cryptographic material  
• Adding tags  
• Creating key aliases  
• Scheduling keys for deletion

For more information, see [customer managed key](https://docs.aws.amazon.com/kms/latest/developerguide/aws-owned-keys.html) in the [AWS Key Management Service Developer Guide](https://docs.aws.amazon.com/kms/latest/developerguide/). For more information about AWS KMS, see [What is AWS Key Management Service?](https://docs.aws.amazon.com/kms/latest/developerguide/)

Key management

How AWS Entity Resolution uses grants in AWS KMS

AWS Entity Resolution requires a grant to use your customer managed key. When you create a matching workflow encrypted with a customer managed key, AWS Entity Resolution creates a grant on your behalf by sending a `CreateGrant` request to AWS KMS. Grants in AWS KMS are used to give AWS Entity Resolution access to a KMS key in a customer account. AWS Entity Resolution requires the grant to use your customer managed key for the following internal operations:
• Send `GenerateDataKey` requests to AWS KMS to generate data keys encrypted by your customer managed key.
• Send `Decrypt` requests to AWS KMS to decrypt the encrypted data keys so that they can be used to encrypt your data.

You can revoke access to the grant, or remove the service's access to the customer managed key at any time. If you do, AWS Entity Resolution won't be able to access any of the data encrypted by the customer managed key, which affects operations that are dependent on that data. For example, if you remove the service access to your key through the grant and attempt to start a job for a matching workflow encrypted with a customer key, then the operation would return an `AccessDeniedException` error.

### Creating a customer managed key

You can create a symmetric customer managed key by using the AWS Management Console, or the AWS KMS APIs.

**To create a symmetric customer managed key**

AWS Entity Resolution supports encryption using Symmetric encryption KMS keys. Follow the steps for **Creating symmetric customer managed key** in the AWS Key Management Service Developer Guide.

**Key policy statement**

Key policies control access to your customer managed key. Every customer managed key must have exactly one key policy, which contains statements that determine who can use the key and how they can use it. When you create your customer managed key, you can specify a key policy. For more information, see **Managing access to customer managed keys** in the AWS Key Management Service Developer Guide.

To use your customer managed key with your AWS Entity Resolution resources, the following API operations must be permitted in the key policy:

• **kms:DescribeKey** – Provides information such as the key ARN, creation date (and deletion date, if applicable), the key state, and the origin and expiration date (if any) of the key material. It includes fields, like KeySpec, that help you distinguish different types of KMS keys. It also displays the key usage (encryption, signing, or generating and verifying MACs) and the algorithms that the KMS key supports. AWS Entity Resolution validates that the KeySpec is SYMMETRIC_DEFAULT and KeyUsage is ENCRYPT_DECRYPT.

• **kms:CreateGrant** – Adds a grant to a customer managed key. Grants control access to a specified KMS key, which allows access to grant operations AWS Entity Resolution requires. For more information about **Using Grants**, see the AWS Key Management Service Developer Guide.

This allows AWS Entity Resolution to do the following:

• Call `GenerateDataKey` to generate an encrypted data key and store it, because the data key isn't immediately used to encrypt.
• Call `Decrypt` to use the stored encrypted data key to access encrypted data.
• Set up a retiring principal to allow the service to `RetireGrant`.

The following are policy statement examples you can add for AWS Entity Resolution:

```json
{
    "Sid" : "Allow access to principals authorized to use AWS Entity Resolution",
    "Effect" : "Allow",
    "Principal" : {
        "AWS" : "*
    },
```
"Action" : ["kms:DescribeKey","kms:CreateGrant"],
"Resource" : "*",
"Condition" : {
  "StringEquals" : {
    "kms:ViaService" : "entityresolution.region.amazonaws.com",
    "kms:CallerAccount" : "111122223333"
  }
}

Permissions for users

When you configure a KMS key as the default key for encryption, the default KMS key policy allows any user with access to the required KMS actions to use this KMS key to encrypt or decrypt resources. You must grant users permission to call the following actions in order to use customer managed KMS key encryption:

• kms:CreateGrant
• kms:Decrypt
• kms:DescribeKey
• kms:GenerateDataKey*
• kms:ReEncrypt

During a CreateMatchingWorkflow request, AWS Entity Resolution will send a DescribeKey request to AWS KMS on your behalf. This will require the IAM entity making the CreateMatchingWorkflow request with a customer managed KMS key to have the kms:DescribeKey permissions on the KMS key policy.

Similarly, the IAM entity invoking the StartMatchingJob API must have kms:Decrypt and kms:GenerateDataKey permissions on the customer managed KMS key provided in the matching workflow.

For more information about specifying permissions in a policy, see the AWS Key Management Service Developer Guide.

For more information about troubleshooting key access, see the AWS Key Management Service Developer Guide.

Specifying a customer managed key for AWS Entity Resolution

You can specify a customer managed key as a second layer encryption for the following resources:

Matching workflow – When you create a matching workflow resource, you can specify the data key by entering a KMSArn, which AWS Entity Resolution uses to encrypt the identifiable personal data stored by the resource.

KMSArn – Enter a key ARN, which is a key identifier for an AWS KMS customer managed key.

Monitoring your encryption keys for AWS Entity Resolution Service

When you use an AWS KMS customer managed key with your AWS Entity Resolution Service resources, you can use AWS CloudTrail or Amazon CloudWatch Logs to track requests that AWS Entity Resolution sends to AWS KMS.

The following examples are AWS CloudTrail events for CreateGrant, GenerateDataKey, Decrypt, and DescribeKey to monitor AWS KMS operations called by AWS Entity Resolution to access data encrypted by your customer managed key:
CreateGrant

When you use an AWS KMS customer managed key to encrypt your matching workflow resource, AWS Entity Resolution sends a CreateGrant request on your behalf to access the KMS key in your AWS account. The grant that AWS Entity Resolution creates are specific to the resource associated with the AWS KMS customer managed key. In addition, AWS Entity Resolution uses the RetireGrant operation to remove a grant when you delete a resource.

The following example event records the CreateGrant operation:

```json
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
    "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-04-22T17:02:00Z"
      }
    },
    "invokedBy": "entityresolution.amazonaws.com"
  },
  "eventTime": "2021-04-22T17:07:02Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "CreateGrant",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "172.12.34.56",
  "userAgent": "ExampleDesktop/1.0 (V1; OS)",
  "requestParameters": {
    "retiringPrincipal": "entityresolution.region.amazonaws.com",
    "operations": [
      "GenerateDataKey",
      "Decrypt"
    ],
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
    "granteePrincipal": "entityresolution.region.amazonaws.com"
  },
  "responseElements": {
    "grantId": "0ab0ac0db700f00e000cc0a000f0000000c0bc0a0000aaafSAMPLE",
```
AWS Entity Resolution uses the DescribeKey operation to verify if the AWS KMS customer managed key associated with your matching resource exists in the account and Region.

The following example event records the DescribeKey operation.

```json
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
    "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-04-22T17:02:00Z"
      }
    },
    "invokedBy": "entityresolution.amazonaws.com"
  },
  "eventTime": "2021-04-22T17:07:02Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DescribeKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "172.12.34.56",
  "userAgent": "ExampleDesktop/1.0 (V1; OS)",
  "requestParameters": {
    "keyId": "00dd0db0-0000-0000-ac00-b0c000SAMPLE"
  },
  "responseElements": null,
  "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fbaSAMPLE",
  "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fbaSAMPLE",
  "readOnly": true,
}
```
GenerateDataKey

When you enable an AWS KMS customer managed key for your matching workflow resource, AWS Entity Resolution sends a GenerateDataKey request through Amazon Simple Storage Service (Amazon S3) to AWS KMS that specifies the AWS KMS customer managed key for the resource.

The following example event records the GenerateDataKey operation.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AWSService",
    "invokedBy": "s3.amazonaws.com"
  },
  "eventTime": "2021-04-22T17:07:02Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateDataKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "172.12.34.56",
  "userAgent": "ExampleDesktop/1.0 (V1; OS)",
  "requestParameters": {
    "keySpec": "AES_256",
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
  },
  "responseElements": null,
  "requestID": "ff000af-00eb-00ce-0e00-ea000f0f0b05SAMPLE",
  "eventID": "ff000af-00eb-00ce-0e00-ea000f0f0b05SAMPLE",
  "readOnly": true,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "111122223333",
  "sharedEventID": "57f5dbee-16da-413e-979f-2c4c6663475e"
}
```

Decrypt

When you enable an AWS KMS customer managed key for your matching workflow resource, AWS Entity Resolution sends a Decrypt request through Amazon Simple Storage Service (Amazon S3) to AWS KMS that specifies the AWS KMS customer managed key for the resource.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AWSService",
    "invokedBy": "s3.amazonaws.com"
  },
  "eventTime": "2021-04-22T17:07:02Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateDataKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "172.12.34.56",
  "userAgent": "ExampleDesktop/1.0 (V1; OS)",
  "requestParameters": {
    "keySpec": "AES_256",
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
  },
  "responseElements": null,
  "requestID": "ff000af-00eb-00ce-0e00-ea000f0f0b05SAMPLE",
  "eventID": "ff000af-00eb-00ce-0e00-ea000f0f0b05SAMPLE",
  "readOnly": true,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "111122223333",
  "sharedEventID": "57f5dbee-16da-413e-979f-2c4c6663475e"
}
```
The following example event records the Decrypt operation.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "s3.amazonaws.com"
    },
    "eventTime": "2021-04-22T17:10:51Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "111122223333",
    "sharedEventID": "dc129381-1d94-49bd-b522-f56a3482d088"
}
```

Considerations

AWS Entity Resolution doesn't support updating a matching workflow with a new customer managed KMS key. In such cases, you can create a new workflow with the customer managed KMS key.

Learn more

The following resources provide more information about data encryption at rest.

For more information about [AWS Key Management Service basic concepts](https://docs.aws.amazon.com/kms/latest/devguide/), see the [AWS Key Management Service Developer Guide](https://docs.aws.amazon.com/kms/latest/devguide/).

For more information about [Security best practices for AWS Key Management Service](https://docs.aws.amazon.com/kms/latest/devguide/), see the [AWS Key Management Service Developer Guide](https://docs.aws.amazon.com/kms/latest/devguide/).

Access AWS Entity Resolution using an interface endpoint (AWS PrivateLink)

You can use AWS PrivateLink to create a private connection between your VPC and AWS Entity Resolution. You can access AWS Entity Resolution as if it were in your VPC, without the use of an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC don't need public IP addresses to access AWS Entity Resolution.
You establish this private connection by creating an *interface endpoint*, powered by AWS PrivateLink. We create an endpoint network interface in each subnet that you enable for the interface endpoint. These are requester-managed network interfaces that serve as the entry point for traffic destined for AWS Entity Resolution.

For more information, see [Access AWS services through AWS PrivateLink](https://docs.aws.amazon.com/privatelink/latest/userguide/private-link-access.html) in the *AWS PrivateLink Guide*.

### Considerations for AWS Entity Resolution

Before you set up an interface endpoint for AWS Entity Resolution, review [Considerations](https://docs.aws.amazon.com/privatelink/latest/userguide/private-link-access.html) in the *AWS PrivateLink Guide*.

AWS Entity Resolution supports making calls to all of its API actions through the interface endpoint.

VPC endpoint policies are not supported for AWS Entity Resolution. By default, full access to AWS Entity Resolution is allowed through the interface endpoint. Alternatively, you can associate a security group with the endpoint network interfaces to control traffic to AWS Entity Resolution through the interface endpoint.

### Create an interface endpoint for AWS Entity Resolution

You can create an interface endpoint for AWS Entity Resolution using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see [Create an interface endpoint](https://docs.aws.amazon.com/privatelink/latest/userguide/private-link-access.html#create-interface-endpoint) in the *AWS PrivateLink Guide*.

Create an interface endpoint for AWS Entity Resolution using the following service name:

```plaintext
com.amazonaws.region.entityresolution
```

If you enable private DNS for the interface endpoint, you can make API requests to AWS Entity Resolution using its default Regional DNS name. For example, `entityresolution.us-east-1.amazonaws.com`.

### Create an endpoint policy for your interface endpoint

An endpoint policy is an IAM resource that you can attach to an interface endpoint. The default endpoint policy allows full access to AWS Entity Resolution through the interface endpoint. To control the access allowed to AWS Entity Resolution from your VPC, attach a custom endpoint policy to the interface endpoint.

An endpoint policy specifies the following information:

- The principals that can perform actions (AWS accounts, IAM users, and IAM roles).
- The actions that can be performed.
- The resources on which the actions can be performed.

For more information, see [Control access to services using endpoint policies](https://docs.aws.amazon.com/privatelink/latest/userguide/private-link-access.html#create-endpoint-policy) in the *AWS PrivateLink Guide*.

#### Example: VPC endpoint policy for AWS Entity Resolution actions

The following is an example of a custom endpoint policy. When you attach this policy to your interface endpoint, it grants access to the listed AWS Entity Resolution actions for all principals on all resources.

```json
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "AWS": ["*"]
    },
    "Action": ["entityresolution:*"],
    "Resource": ["*"]
  }]
}
```
Identity and access management for AWS Entity Resolution

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be authenticated (signed in) and authorized (have permissions) to use AWS Entity Resolution resources. IAM is an AWS service that you can use with no additional charge.

Topics

- Audience (p. 62)
- Authenticating with identities (p. 63)
- Managing access using policies (p. 65)
- How AWS Entity Resolution works with IAM (p. 66)
- Identity-based policy examples for AWS Entity Resolution (p. 71)
- AWS managed policies for AWS Entity Resolution (p. 73)
- Troubleshooting AWS Entity Resolution identity and access (p. 77)

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in AWS Entity Resolution.

Service user – If you use the AWS Entity Resolution service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more AWS Entity Resolution features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in AWS Entity Resolution, see Troubleshooting AWS Entity Resolution identity and access (p. 77).

Service administrator – If you're in charge of AWS Entity Resolution resources at your company, you probably have full access to AWS Entity Resolution. It's your job to determine which AWS Entity Resolution features and resources your service users should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with AWS Entity Resolution, see How AWS Entity Resolution works with IAM (p. 66).

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to AWS Entity Resolution. To view example AWS Entity Resolution
identity-based policies that you can use in IAM, see Identity-based policy examples for AWS Entity Resolution (p. 71).

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be authenticated (signed in to AWS) as the AWS account root user, as an IAM user, or by assuming an IAM role.

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users, your company's single sign-on authentication, and your Google or Facebook credentials are examples of federated identities. When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see How to sign in to your AWS account in the AWS Sign-In User Guide.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests by using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see Signing AWS API requests in the IAM User Guide.

Regardless of the authentication method that you use, you might be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Multi-factor authentication in the AWS IAM Identity Center User Guide and Using multi-factor authentication (MFA) in AWS in the IAM User Guide.

AWS account root user

When you create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account root user and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the IAM User Guide.

Federated identity

As a best practice, require human users, including users that require administrator access, to use federation with an identity provider to access AWS services by using temporary credentials.

A federated identity is a user from your enterprise user directory, a web identity provider, the AWS Directory Service, the Identity Center directory, or any user that accesses AWS services by using credentials provided through an identity source. When federated identities access AWS accounts, they assume roles, and the roles provide temporary credentials.

For centralized access management, we recommend that you use AWS IAM Identity Center. You can create users and groups in IAM Identity Center, or you can connect and synchronize to a set of users and groups in your own identity source for use across all your AWS accounts and applications. For information about IAM Identity Center, see What is IAM Identity Center? in the AWS IAM Identity Center User Guide.

IAM users and groups

An IAM user is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM
users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see Rotate access keys regularly for use cases that require long-term credentials in the IAM User Guide.

An IAM group is an identity that specifies a collection of IAM users. You can’t sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named IAMAdmins and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see When to create an IAM user (instead of a role) in the IAM User Guide.

IAM roles

An IAM role is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by switching roles. You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see Using IAM roles in the IAM User Guide.

IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see Creating a role for a third-party Identity Provider in the IAM User Guide. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permissions sets, see Permission sets in the AWS IAM Identity Center User Guide.

- **Temporary IAM user permissions** – An IAM user or role can assume an IAM role to temporarily take on different permissions for a specific task.

- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the IAM User Guide.

- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.

- **Principal permissions** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. Policies grant permissions to a principal. When you use some services, you might perform an action that then triggers another action in a different service. In this case, you must have permissions to perform both actions. To see whether an action requires additional dependent actions in a policy, see in the Service Authorization Reference.

- **Service role** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the IAM User Guide.

- **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the IAM User Guide.

Administrators can use AWS JSON policies to specify who has access to what. That is, which principal can perform actions on what resources, and under what conditions.

By default, users and roles have no permissions. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the \texttt{iam:GetRole} action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Creating IAM policies in the IAM User Guide.

Identity-based policies can be further categorized as inline policies or managed policies. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choosing between managed policies and inline policies in the IAM User Guide.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM role trust policies and Amazon S3 bucket policies. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.
Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see Access control list (ACL) overview in the Amazon Simple Storage Service Developer Guide.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of an entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the Principal field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the IAM User Guide.

- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see How SCPs work in the AWS Organizations User Guide.

- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the IAM User Guide.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the IAM User Guide.

How AWS Entity Resolution works with IAM

Before you use IAM to manage access to AWS Entity Resolution, learn what IAM features are available to use with AWS Entity Resolution.

IAM features you can use with AWS Entity Resolution

<table>
<thead>
<tr>
<th>IAM feature</th>
<th>AWS Entity Resolution support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identity-based policies (p. 67)</td>
<td>Yes</td>
</tr>
<tr>
<td>Resource-based policies (p. 67)</td>
<td>No</td>
</tr>
<tr>
<td>IAM feature</td>
<td>AWS Entity Resolution support</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>Policy actions (p. 68)</td>
<td>Yes</td>
</tr>
<tr>
<td>Policy resources (p. 68)</td>
<td>Yes</td>
</tr>
<tr>
<td>Policy condition keys (p. 69)</td>
<td>Yes</td>
</tr>
<tr>
<td>ACLs (p. 69)</td>
<td>No</td>
</tr>
<tr>
<td>ABAC (tags in policies) (p. 70)</td>
<td>Partial</td>
</tr>
<tr>
<td>Temporary credentials (p. 70)</td>
<td>Yes</td>
</tr>
<tr>
<td>Principal permissions (p. 70)</td>
<td>Yes</td>
</tr>
<tr>
<td>Service roles (p. 71)</td>
<td>Yes</td>
</tr>
<tr>
<td>Service-linked roles (p. 71)</td>
<td>No</td>
</tr>
</tbody>
</table>

To get a high-level view of how AWS Entity Resolution and other AWS services work with most IAM features, see [AWS services that work with IAM](https://docs.aws.amazon.com/IAM/latest/UserGuide/what-is-iam.html) in the IAM User Guide.

**Identity-based policies for AWS Entity Resolution**

| Supports identity-based policies | Yes |

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see [Creating IAM policies](https://docs.aws.amazon.com/IAM/latest/UserGuide/what-is-iam.html) in the IAM User Guide.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. You can't specify the principal in an identity-based policy because it applies to the user or role to which it is attached. To learn about all of the elements that you can use in a JSON policy, see [IAM JSON policy elements reference](https://docs.aws.amazon.com/IAM/latest/UserGuide/reference-policies-elements.html) in the IAM User Guide.

**Identity-based policy examples for AWS Entity Resolution**


**Resource-based policies within AWS Entity Resolution**

| Supports resource-based policies | No |

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM role trust policies and Amazon S3 bucket policies. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.
To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. Adding a cross-account principal to a resource-based policy is only half of establishing the trust relationship. When the principal and the resource are in different AWS accounts, an IAM administrator in the trusted account must also grant the principal entity (user or role) permission to access the resource. They grant permission by attaching an identity-based policy to the entity. However, if a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required. For more information, see How IAM roles differ from resource-based policies in the IAM User Guide.

Policy actions for AWS Entity Resolution

| Supports policy actions | Yes |

Administrators can use AWS JSON policies to specify who has access to what. That is, which principal can perform actions on what resources, and under what conditions.

The Action element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as permission-only actions that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called dependent actions.

Include actions in a policy to grant permissions to perform the associated operation.

To see a list of AWS Entity Resolution actions, see Actions Defined by AWS Entity Resolution in the Service Authorization Reference.

Policy actions in AWS Entity Resolution use the following prefix before the action:

entityresolution

To specify multiple actions in a single statement, separate them with commas.

"Action": [
   "entityresolution:action1",
   "entityresolution:action2"
]

To view examples of AWS Entity Resolution identity-based policies, see Identity-based policy examples for AWS Entity Resolution (p. 71).

Policy resources for AWS Entity Resolution

| Supports policy resources | Yes |

Administrators can use AWS JSON policies to specify who has access to what. That is, which principal can perform actions on what resources, and under what conditions.

The Resource JSON policy element specifies the object or objects to which the action applies. Statements must include either a Resource or a NotResource element. As a best practice, specify
a resource using its Amazon Resource Name (ARN). You can do this for actions that support a specific resource type, known as resource-level permissions.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

"Resource": "*"

To see a list of AWS Entity Resolution resource types and their ARNs, see Resources Defined by AWS Entity Resolution in the Service Authorization Reference. To learn with which actions you can specify the ARN of each resource, see Actions Defined by AWS Entity Resolution.

To view examples of AWS Entity Resolution identity-based policies, see Identity-based policy examples for AWS Entity Resolution (p. 71).

Policy condition keys for AWS Entity Resolution

Supports service-specific policy condition keys Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which principal can perform actions on what resources, and under what conditions.

The Condition element (or Condition block) lets you specify conditions in which a statement is in effect. The Condition element is optional. You can create condition expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple Condition elements in a statement, or multiple keys in a single Condition element, AWS evaluates them using a logical AND operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical OR operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see IAM policy elements: variables and tags in the IAM User Guide.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see AWS global condition context keys in the IAM User Guide.

To see a list of AWS Entity Resolution condition keys, see Condition Keys for AWS Entity Resolution in the Service Authorization Reference. To learn with which actions and resources you can use a condition key, see Actions Defined by AWS Entity Resolution.

To view examples of AWS Entity Resolution identity-based policies, see Identity-based policy examples for AWS Entity Resolution (p. 71).

ACLs in AWS Entity Resolution

Supports ACLs No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.
ABAC with AWS Entity Resolution

| Supports ABAC (tags in policies) | Partial |

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called *tags*. You can attach tags to IAM entities (users or roles) and to many AWS resources. Tagging entities and resources is the first step of ABAC. Then you design ABAC policies to allow operations when the principal's tag matches the tag on the resource that they are trying to access.

ABAC is helpful in environments that are growing rapidly and helps with situations where policy management becomes cumbersome.

To control access based on tags, you provide tag information in the *condition element* of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

If a service supports all three condition keys for every resource type, then the value is *Yes* for the service. If a service supports all three condition keys for only some resource types, then the value is *Partial*.

For more information about ABAC, see *What is ABAC?* in the IAM User Guide. To view a tutorial with steps for setting up ABAC, see *Use attribute-based access control (ABAC)* in the IAM User Guide.

Using temporary credentials with AWS Entity Resolution

| Supports temporary credentials | Yes |

Some AWS services don't work when you sign in using temporary credentials. For additional information, including which AWS services work with temporary credentials, see *AWS services that work with IAM* in the IAM User Guide.

You are using temporary credentials if you sign in to the AWS Management Console using any method except a user name and password. For example, when you access AWS using your company's single sign-on (SSO) link, that process automatically creates temporary credentials. You also automatically create temporary credentials when you sign in to the console as a user and then switch roles. For more information about switching roles, see *Switching to a role (console)* in the IAM User Guide.

You can manually create temporary credentials using the AWS CLI or AWS API. You can then use those temporary credentials to access AWS. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see *Temporary security credentials in IAM*.

Cross-service principal permissions for AWS Entity Resolution

| Supports principal permissions | Yes |

When you use an IAM user or role to perform actions in AWS, you are considered a principal. Policies grant permissions to a principal. When you use some services, you might perform an action that then triggers another action in a different service. In this case, you must have permissions to perform both actions. To see whether an action requires additional dependent actions in a policy, see in the Service Authorization Reference.
Service roles for AWS Entity Resolution

<table>
<thead>
<tr>
<th>Supports service roles</th>
<th>Yes</th>
</tr>
</thead>
</table>

A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the IAM User Guide.

**Warning**
Changing the permissions for a service role might break AWS Entity Resolution functionality. Edit service roles only when AWS Entity Resolution provides guidance to do so.

Service-linked roles for AWS Entity Resolution

<table>
<thead>
<tr>
<th>Supports service-linked roles</th>
<th>No</th>
</tr>
</thead>
</table>

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

For details about creating or managing service-linked roles, see AWS services that work with IAM. Find a service in the table that includes a Yes in the Service-linked role column. Choose the Yes link to view the service-linked role documentation for that service.

Identity-based policy examples for AWS Entity Resolution

By default, users and roles don’t have permission to create or modify AWS Entity Resolution resources. They also can’t perform tasks by using the AWS Management Console, AWS Command Line Interface (AWS CLI), or AWS API. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see Creating IAM policies in the IAM User Guide.

For details about actions and resource types defined by AWS Entity Resolution, including the format of the ARNs for each of the resource types, see Actions, Resources, and Condition Keys for AWS Entity Resolution in the Service Authorization Reference.

Topics
- Policy best practices (p. 71)
- Using the AWS Entity Resolution console (p. 72)
- Allow users to view their own permissions (p. 72)

Policy best practices

Identity-based policies determine whether someone can create, access, or delete AWS Entity Resolution resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:
• **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the AWS managed policies that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see [AWS managed policies](http://aws.amazon.com) or [AWS managed policies for job functions](http://aws.amazon.com) in the IAM User Guide.

• **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as least-privilege permissions. For more information about using IAM to apply permissions, see [Policies and permissions in IAM](http://aws.amazon.com) in the IAM User Guide.

• **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as AWS CloudFormation. For more information, see [IAM JSON policy elements: Condition](http://aws.amazon.com) in the IAM User Guide.

• **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see [IAM Access Analyzer policy validation](http://aws.amazon.com) in the IAM User Guide.

• **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see [Configuring MFA-protected API access](http://aws.amazon.com) in the IAM User Guide.

For more information about best practices in IAM, see [Security best practices in IAM](http://aws.amazon.com) in the IAM User Guide.

### Using the AWS Entity Resolution console

To access the AWS Entity Resolution console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the AWS Entity Resolution resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won’t function as intended for entities (users or roles) with that policy.

You don’t need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that they’re trying to perform.

To ensure that users and roles can still use the AWS Entity Resolution console, also attach the AWS Entity Resolution **ConsoleAccess** or **ReadOnly** AWS managed policy to the entities. For more information, see [Adding permissions to a user](http://aws.amazon.com) in the IAM User Guide.

### Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
```
AWS managed policies for AWS Entity Resolution

An AWS managed policy is a standalone policy that is created and administered by AWS. AWS managed policies are designed to provide permissions for many common use cases so that you can start assigning permissions to users, groups, and roles.

Keep in mind that AWS managed policies might not grant least-privilege permissions for your specific use cases because they're available for all AWS customers to use. We recommend that you reduce permissions further by defining customer managed policies that are specific to your use cases.

You cannot change the permissions defined in AWS managed policies. If AWS updates the permissions defined in an AWS managed policy, the update affects all principal identities (users, groups, and roles) that the policy is attached to. AWS is most likely to update an AWS managed policy when a new AWS service is launched or new API operations become available for existing services.

For more information, see AWS managed policies in the IAM User Guide.

AWS managed policy: AWSEntityResolutionConsoleFullAccess

You can attach the AWSEntityResolutionConsoleFullAccess policy to your IAM identities.

This policy grants full access to AWS Entity Resolution endpoints and resources.

This policy also allows certain read access to related AWS services like S3, AWS Glue, Tagging and AWS KMS so that the console can display choices and use the selected ones to perform entity resolution actions. Some resources are narrowed down to contain the service name entityresolution.

Because AWS Entity Resolution relies on a passed role to perform actions on related AWS resources, this policy also grants the permissions to select and pass a desired role.
Permissions details

This policy includes the following permissions.

- **EntityResolutionAccess** – Allows principals full access to AWS Entity Resolution endpoints and resources.
- **GlueSourcesConsoleDisplay** – Grants the access to list AWS Glue tables as data source options and import table schema of a data source for user experience.
- **S3BucketsConsoleDisplay** – Grants the access to list all S3 buckets as data source options.
- **S3SourcesConsoleDisplay** – Grants the access to display S3 buckets as data source options.
- **TaggingConsoleDisplay** – Grants the access to read tagging keys and values.
- **KMSConsoleDisplay** – Grants the access to describe keys and list aliases in AWS Key Management Service to decrypt and encrypt data sources.
- **ListRolesToPickForPassing** – Grants the access to list all roles so that the user can pick the role to be passed.
- **PassRoleToEntityResolutionService** – Grants the access to pass a narrowed down role to the AWS Entity Resolution service.
- **ManageEventBridgeRules** – Grants the access to create, update, and delete the Amazon EventBridge rule for getting S3 notifications.
- **ADXReadAccess** – Grants the access to AWS Data Exchange to verify if the customer has an entitlement or a subscription.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "EntityResolutionAccess",
      "Effect": "Allow",
      "Action": [
        "entityresolution:*"
      ],
      "Resource": "*"
    },
    {
      "Sid": "GlueSourcesConsoleDisplay",
      "Effect": "Allow",
      "Action": [
        "glue:GetSchema",
        "glue:SearchTables",
        "glue:GetSchemaByDefinition",
        "glue:GetSchemaVersion",
        "glue:GetSchemaVersionsDiff",
        "glue:GetDatabase",
        "glue:GetDatabases",
        "glue:GetTable",
        "glue:GetTables",
        "glue:GetTableVersion",
        "glue:GetTableVersions"
      ],
      "Resource": "*"
    },
    {
      "Sid": "S3BucketsConsoleDisplay",
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets"
      ],
      "Resource": "*"
    }
  ]
}
```


>`{ 
  "Sid": "S3SourcesConsoleDisplay",
  "Effect": "Allow",
  "Action": [
    "s3:ListBucket",
    "s3:GetBucketLocation",
    "s3:ListBucketVersions",
    "s3:GetBucketVersioning"
  ],
  "Resource": "*"
},

{ 
  "Sid": "TaggingConsoleDisplay",
  "Effect": "Allow",
  "Action": [
    "tag:GetTagKeys",
    "tag:GetTagValues"
  ],
  "Resource": "*"
},

{ 
  "Sid": "KMSConsoleDisplay",
  "Effect": "Allow",
  "Action": [
    "kms:DescribeKey",
    "kms:ListAliases"
  ],
  "Resource": "*"
},

{ 
  "Sid": "ListRolesToPickRoleForPassing",
  "Effect": "Allow",
  "Action": [
    "iam:ListRoles"
  ],
  "Resource": "*"
},

{ 
  "Sid": "PassRoleToEntityResolutionService",
  "Effect": "Allow",
  "Action": [
    "iam:PassRole"
  ],
  "Resource": "arn:aws:iam::*:role/*entityresolution*",
  "Condition": {
    "StringEquals": {
      "iam:PassedToService": [
        "entityresolution.amazonaws.com"
      ]
    }
  }
},

{ 
  "Sid": "ManageEventBridgeRules",
  "Effect": "Allow",
  "Action": [
    "events:PutRule",
    "events:DeleteRule",
    "events:PutTargets"
  ],
  "Resource": [
    "arn:aws:events:*:*:rule/entity-resolution-automatic*"
  ]
},

{ 

}
You can attach `AWSEntityResolutionConsoleReadOnlyAccess` to your IAM entities.

This policy grants read-only access to AWS Entity Resolution endpoints and resources.

**Permissions details**

This policy includes the following permissions.

- **EntityResolutionRead** – Allows principals read-only access to AWS Entity Resolution endpoints and resources.

**AWS Entity Resolution updates to AWS managed policies**

View details about updates to AWS managed policies for AWS Entity Resolution since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the AWS Entity Resolution Document history page.

<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWSEntityResolutionConsoleFullAccess</td>
<td>– Update to existing policy</td>
<td>October 16, 2023</td>
</tr>
<tr>
<td></td>
<td>Added <code>ADXReadAccess</code> and <code>ManageEventBridgeRules</code> to enable the provider services option in the matching workflow.</td>
<td></td>
</tr>
<tr>
<td>AWSEntityResolutionConsoleFullAccess</td>
<td>AWS Entity Resolution started tracking changes</td>
<td>August 18, 2023</td>
</tr>
</tbody>
</table>
Troubleshooting AWS Entity Resolution identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with AWS Entity Resolution and IAM.

Topics
- I am not authorized to perform an action in AWS Entity Resolution (p. 77)
- I am not authorized to perform iam:PassRole (p. 77)
- I want to allow people outside of my AWS account to access my AWS Entity Resolution resources (p. 77)

I am not authorized to perform an action in AWS Entity Resolution

If the AWS Management Console tells you that you're not authorized to perform an action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password.

The following example error occurs when the mateojackson IAM user tries to use the console to view details about a fictional my-example-widget resource but does not have the fictional entityresolution:GetWidget permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform: entityresolution:GetWidget on resource: my-example-widget
```

In this case, Mateo asks his administrator to update his policies to allow him to access the my-example-widget resource using the entityresolution:GetWidget action.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the iam:PassRole action, your policies must be updated to allow you to pass a role to AWS Entity Resolution.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named marymajor tries to use the console to perform an action in AWS Entity Resolution. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform: iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the iam:PassRole action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I want to allow people outside of my AWS account to access my AWS Entity Resolution resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support
Compliance validation

Compliance validation for AWS Entity Resolution

To learn whether an AWS service is within the scope of specific compliance programs, see AWS services in Scope by Compliance Program and choose the compliance program that you are interested in. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- **Security and Compliance Quick Start Guides** – These deployment guides discuss architectural considerations and provide steps for deploying baseline environments on AWS that are security and compliance focused.
- **Architecting for HIPAA Security and Compliance on Amazon Web Services** – This whitepaper describes how companies can use AWS to create HIPAA-eligible applications.
  
  **Note**
  Not all AWS services are HIPAA eligible. For more information, see the HIPAA Eligible Services Reference.

- **AWS Compliance Resources** – This collection of workbooks and guides might apply to your industry and location.
- **Evaluating Resources with Rules** in the AWS Config Developer Guide – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- **AWS Security Hub** – This AWS service provides a comprehensive view of your security state within AWS. Security Hub uses security controls to evaluate your AWS resources and to check your compliance against security industry standards and best practices. For a list of supported services and controls, see Security Hub controls reference.
- **AWS Audit Manager** – This AWS service helps you continuously audit your AWS usage to simplify how you manage risk and compliance with regulations and industry standards.

Resilience in AWS Entity Resolution

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency,
high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

In addition to the AWS global infrastructure, AWS Entity Resolution offers several features to help support your data resiliency and backup needs.
Monitoring AWS Entity Resolution

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS Entity Resolution and your other AWS solutions. AWS provides the following monitoring tools to watch AWS Entity Resolution, report when something is wrong, and take automatic actions when appropriate:

- **AWS CloudTrail** captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon S3 bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. For more information, see the [AWS CloudTrail User Guide](https://docs.aws.amazon.com/AmazonCloudTrail/latest/userguide/). The calls captured include calls from the AWS Entity Resolution console and code calls to the AWS Entity Resolution API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AWS Entity Resolution. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to AWS Entity Resolution, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the [AWS CloudTrail User Guide](https://docs.aws.amazon.com/AmazonCloudTrail/latest/userguide/).

AWS Entity Resolution information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS Entity Resolution, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see [Viewing events with CloudTrail Event history](https://docs.aws.amazon.com/AmazonCloudTrail/latest/userguide/). For an ongoing record of events in your AWS account, including events for AWS Entity Resolution, create a trail. A trail enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for creating a trail
- CloudTrail supported services and integrations
- Configuring Amazon SNS notifications for CloudTrail
- Receiving CloudTrail log files from multiple regions and Receiving CloudTrail log files from multiple accounts

All AWS Entity Resolution actions are logged by CloudTrail and are documented in the [AWS Entity Resolution API Reference](https://docs.aws.amazon.com/aws-sdk-reference/).
Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail `userIdentity` element.

**Understanding AWS Entity Resolution log file entries**

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.
Creating AWS Entity Resolution resources with AWS CloudFormation

AWS Entity Resolution is integrated with AWS CloudFormation, a service that helps you to model and set up your AWS resources so that you can spend less time creating and managing your resources and infrastructure. You create a template that describes all the AWS resources that you want (such as AWS::EntityResolution::MatchingWorkflow, AWS::EntityResolution::SchemaMapping and AWS::EntityResolution::IdMappingWorkflow), and AWS CloudFormation provisions and configures those resources for you.

When you use AWS CloudFormation, you can reuse your template to set up your AWS Entity Resolution resources consistently and repeatedly. Describe your resources once, and then provision the same resources over and over in multiple AWS accounts and Regions.

AWS Entity Resolution and AWS CloudFormation templates

To provision and configure resources for AWS Entity Resolution and related services, you must understand AWS CloudFormation templates. Templates are formatted text files in JSON or YAML. These templates describe the resources that you want to provision in your AWS CloudFormation stacks. If you're unfamiliar with JSON or YAML, you can use AWS CloudFormation Designer to help you get started with AWS CloudFormation templates. For more information, see What is AWS CloudFormation Designer? in the AWS CloudFormation User Guide.

AWS Entity Resolution supports creating AWS::EntityResolution::MatchingWorkflow, AWS::EntityResolution::SchemaMapping and AWS::EntityResolution::IdMappingWorkflow in AWS CloudFormation. For more information, including examples of JSON and YAML templates for AWS::EntityResolution::MatchingWorkflow, AWS::EntityResolution::SchemaMapping and AWS::EntityResolution::IdMappingWorkflow, see the AWS Entity Resolution resource type reference in the AWS CloudFormation User Guide.

The following templates are available:

- **Matching workflow**

  Create a MatchingWorkflow object which stores the configuration of the data processing job to be run.

  For more information, see the following topics:

  - [CreateMatchingWorkflow](https://docs.aws.amazon.com/aws-sdk-common参考资料.md) in the AWS Entity Resolution API Reference

- **Schema mapping**

  Create a schema mapping, which defines the schema of the input customer records table.

  For more information, see the following topics:

CreateSchemaMapping in the AWS Entity Resolution API Reference

• **ID mapping workflow**

Create an `IdMappingWorkflow` object which stores the configuration of the data processing job to be run.

For more information, see the following topics:

- [AWS::EntityResolution::IdMappingWorkflow](#) in the AWS CloudFormation User Guide
- [CreateIdMappingWorkflow](#) in the AWS Entity Resolution API Reference

Learn more about AWS CloudFormation

To learn more about AWS CloudFormation, see the following resources:

- [AWS CloudFormation](#)
- [AWS CloudFormation User Guide](#)
- [AWS CloudFormation API Reference](#)
- [AWS CloudFormation Command Line Interface User Guide](#)
Quotas for AWS Entity Resolution

Your AWS account has default quotas, formerly referred to as limits, for each AWS service. Unless otherwise noted, each quota is Region-specific. You can request increases for some quotas, but other quotas can't be increased.

To view the quotas for AWS Entity Resolution, open the Service Quotas console. In the navigation pane, choose AWS services and select AWS Entity Resolution.

To request a quota increase, see Requesting a Quota Increase in the Service Quotas User Guide. If the quota is not yet available in Service Quotas, use the limit increase form.

Your AWS account has the following quotas related to AWS Entity Resolution.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default</th>
<th>Adjustable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concurrent ID mapping jobs</td>
<td>1</td>
<td>No</td>
<td>The maximum number of ID mapping jobs that can be processed concurrently in the current AWS Region.</td>
</tr>
<tr>
<td>Concurrent matching jobs</td>
<td>1</td>
<td>No</td>
<td>The maximum number of matching jobs that can be processed concurrently in the current AWS Region.</td>
</tr>
<tr>
<td>Concurrent provider service matching jobs</td>
<td>1</td>
<td>No</td>
<td>The maximum number of provider service matching jobs that can be processed concurrently in the current AWS Region.</td>
</tr>
<tr>
<td>Data input</td>
<td>20</td>
<td>No</td>
<td>This is the list of input tables that you want to use in a matching workflow. Each input corresponds to a column in your AWS Glue input data table, which contains the column name and additional information that AWS Entity Resolution uses for matching purposes. Inputs must contain a Unique ID plus at least one additional input field.</td>
</tr>
<tr>
<td>Data output</td>
<td>750</td>
<td>No</td>
<td>This is a list of OutputAttribute objects, each of which have the fields Name and Hashed. Each of these objects represent a column to be included in the AWS Glue output table and whether you want the values in the column to be hashed.</td>
</tr>
<tr>
<td>Data schema</td>
<td>25</td>
<td>No</td>
<td>The maximum number of data schema input fields.</td>
</tr>
<tr>
<td>ID mapping workflows</td>
<td>10</td>
<td>Yes</td>
<td>The maximum number of ID mapping workflows that you can create in this AWS account in the current AWS Region.</td>
</tr>
</tbody>
</table>
### Name

<table>
<thead>
<tr>
<th>Name</th>
<th>Default</th>
<th>Adjustable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Match rule</td>
<td>15</td>
<td>No</td>
<td>For rule-based matching, this is the rule number applied that generated a matched record set. This is part of Matching Workflow Metadata that will be included in output.</td>
</tr>
<tr>
<td>Matching workflows</td>
<td>10</td>
<td>Yes</td>
<td>The maximum number of matching workflows.</td>
</tr>
<tr>
<td>Number of rules per workflow</td>
<td>15</td>
<td>No</td>
<td>The maximum number of rules per matching workflow.</td>
</tr>
<tr>
<td>Rate of GetMatchId API requests</td>
<td>50</td>
<td>Yes</td>
<td>The maximum number of GetCustomerId API requests per second.</td>
</tr>
<tr>
<td>Schema mappings</td>
<td>50</td>
<td>Yes</td>
<td>The maximum number of schema mappings that you can create in this account in the current AWS Region.</td>
</tr>
<tr>
<td>Unique match keys per across ruleset</td>
<td>15</td>
<td>No</td>
<td>The maximum number of unique match keys per rule set. A match key instructs AWS Entity Resolution which input fields are to be considered as similar data and which are to be considered as different data. This helps AWS Entity Resolution automatically configure rule-based matching rules and compare similar data stored in different input fields.</td>
</tr>
</tbody>
</table>

### API throttling quotas

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rate of GetMatchId requests</td>
<td>50 TPS</td>
<td>Maximum number of GetMatchId API calls per second.</td>
</tr>
</tbody>
</table>
# Document history for the AWS Entity Resolution User Guide

The following table describes the documentation releases for AWS Entity Resolution.

For notification about updates to this documentation, you can subscribe to the RSS feed. To subscribe to RSS updates, you must have an RSS plug-in enabled for the browser you are using.

<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AWS PrivateLink</strong></td>
<td>AWS Entity Resolution now supports additional data security with AWS PrivateLink that helps customers to privately access services hosted on AWS.</td>
<td>October 20, 2023</td>
</tr>
</tbody>
</table>
| **AWS CloudFormation Resources**             | **– New and updated resources**
AWS Entity Resolution has added the following resource:
AWS::EntityResolution::IdMappingWorkflow  
and updated the following resources:
AWS::EntityResolution::MatchingWorkflow  
and
AWS::EntityResolution::Schemamapping.                                                                                     | October 19, 2023 |
| **Update to existing policy**                | The following new permissions have been added to the AWSEntityResolutionConsoleFullAccess managed policy: ADXReadAccess and ManageEventBridgeRules.                                                         | October 16, 2023 |
| **Schema mapping – update**                  | Customers now have the ability to edit and update an existing data schema.                                                                                                                                     | October 16, 2023 |
| **Matching workflow – update**               | Customers can now select a preferred data provider service to help match and link their data.                                                                                                                | October 16, 2023 |
| **ID mapping workflow**                      | Customers can use this new workflow to specify ID mapping details, choose your desired ID mapping method, and specify data input and output fields.                                                        | October 16, 2023 |
| **AWS CloudFormation integration**           | AWS Entity Resolution now integrates with AWS CloudFormation.                                                                                                                                                 | August 24, 2023  |
| **AWS managed policy update - New policies** | AWS Entity Resolution added two new managed policies.                                                                                                                                                         | August 18, 2023  |
| Initial release (p. 86) | Initial release of the AWS Entity Resolution User Guide | July 26, 2023 |
AWS Entity Resolution Glossary

Amazon Resource Name (ARN)

A unique identifier for AWS resources. ARNs are required when you need to specify a resource unambiguously across all of AWS Entity Resolution, such as in AWS Entity Resolution policies, Amazon Relational Database Service (Amazon RDS) tags, and API calls.

Automatic processing

A processing cadence option for a matching workflow job that enables it to be run on automatically when your data input changes.

This option is available for rule-based matching (p. 95) only.

By default, the processing cadence for a matching workflow job is set to Manual (p. 90), which enables it to be run on demand. You can set up Automatic processing to run your matching workflow job automatically when your data input changes. This keeps your matching workflow output up-to-date.

AWS KMS key ARN

This is your AWS KMS Amazon Resource Name (ARN) for encryption at rest. If not provided, system will use an AWS Entity Resolution managed KMS key.

Cleartext

Data that isn't cryptographically protected.

Confidence level (ConfidenceLevel)

For ML matching, this is the confidence level applied by AWS Entity Resolution when ML identifies a matched record set. This is part of the matching workflow metadata (p. 91) that will be included in output.

Decryption

The process of transforming encrypted data back to its original form. Decryption can only be performed if you have access to the secret key.

Encryption

The process of encoding data into a form that appears random using a secret value called a key. It's impossible to determine the original plaintext without access to the key.
Group name

The **Group name** references the entire group of input fields and can help you to group parsed data together for matching purposes.

For example, if there are three input fields: **first_name**, **middle_name**, and **last_name**, you can group them together by entering in the **Group name** as **full_name** for matching and output.

Hash

Hashing means applying a cryptographic algorithm that produces an irreversible and unique string of characters of a fixed size—called a hash. AWS Entity Resolution uses Secure Hash Algorithm 256-bit (SHA256) hash protocol and will output a 32-byte character string. In AWS Entity Resolution, you can choose whether to hash data values in your output.

Hash protocol (HashingProtocol)

AWS Entity Resolution uses Secure Hash Algorithm 256-bit (SHA256) hash protocol and will output a 32-byte character string. This is part of the [matching workflow metadata](p. 91) that will be included in output.

ID mapping workflow

The process that you set up to specify the input data to translate your IDs and how you want the ID mapping to be performed.

AWS Entity Resolution currently supports LiveRamp as the ID mapping method. You must have a subscription to LiveRamp through AWS Data Exchange to use the ID mapping workflow.

For more information, see [Subscribe to a provider service on AWS Data Exchange](p. 6).

Input field

An input field corresponds to a column name from your AWS Glue input data table.

Input Source ARN (InputSourceARN)

The Amazon Resource Name (ARN) that was generated for an AWS Glue table input. This is part of [matching workflow metadata](p. 91) that will be included in output.

Input type

The type of input data. You select it from a pre-configured list of values such as name, address, phone number, or email address. Input type tells AWS Entity Resolution what kind of data that you're presenting it, allowing it to be classified and normalized properly.
Machine learning-based matching

Machine learning-based matching (ML matching) finds matches across your data that might be incomplete or might not look exactly the same. ML matching is a preset process that will attempt to match records across all of the data you input. ML matching returns a match ID (p. 90) and a confidence level (p. 88) for each matched set of data.

Manual processing

A processing cadence option for a matching workflow job that enables it to be run on demand.

This option is set by default and is available for both rule-based matching (p. 95) and machine learning-based matching (p. 90).

Many-to-Many matching

Many-to-many matching compares multiple instances of similar data. Values in input fields that have been assigned the same match key will be matched against each other, regardless of whether they are in the same input field or different input fields.

For example, you might have multiple phone number input fields like mobile_phone and home_phone that have the same match key “Phone”. Use many-to-many matching to compare data in the mobile_phone input field with data in the mobile_phone input field and data in the home_phone input field.

Matching rules evaluate data in multiple input fields with the same match key with an (or) operation, and one-to-many matching compares values across multiple input fields. This means that if any combination of mobile_phone or home_phone matches between two records, the “Phone” match key will return a match. For match key “Phone” to find a match, Record One mobile_phone = Record Two mobile_phone OR Record One mobile_phone = Record Two home_phone OR Record One home_phone = Record Two home_phone OR Record One home_phone = Record Two mobile_phone.

Match ID (MatchID)

For rule-based matching and ML matching, this is the ID generated by AWS Entity Resolution and applied to each matched record set. This is part of the matching workflow metadata (p. 91) that will be included in output.

Match key (MatchKey)

Match key instructs AWS Entity Resolution which input fields to consider as similar data and which to consider as different data. This helps AWS Entity Resolution automatically configure rule-based matching rules and compare similar data stored in different input fields.

If there are multiple types of phone number information like a mobile_phone input field and a home_phone input field in your data that you would like compared together, you could give them both the match key “Phone”. Then rule-based matching can be configured to compare data using “or” statements in all input fields with the “Phone” match key (see One-to-One Matching (p. 94) and Many-to-Many Matching (p. 90) definitions in Matching Workflow section).
If you want rule-based matching to consider different types of phone number information completely separately, you can create more specific match keys like "Mobile_Phone" and "Home_Phone". Then, when setting up a matching workflow, you can specify how each phone match key will be used in rule-based matching.

If no MatchKey is specified for a particular input field, it can't be used in matching but can be carried through the matching workflow process and can be output if desired.

**Match key name**

The name assigned to a **Match Key**.

**Match rule (MatchRule)**

For rule-based matching, this is the rule number applied that generated a matched record set. This is part of the **matching workflow metadata (p. 91)** that will be included in output.

**Matching**

The process of combining and comparing data from different input fields, tables, or databases and determining which of it is alike – or “matches” – based upon satisfying certain matching criteria (for example, either through matching rules or models).

**Matching workflow**

The process that you set up to specify the input data to match together and how the matching should be performed.

**Matching workflow description**

An optional description of the matching workflow that you might choose to enter. Descriptions help you differentiate between matching workflows if you create more than one.

**Matching workflow name**

The name for the matching workflow that you specify.

**Note**

Matching workflow names must be unique. They can't have the same name or an error will be returned.

**Matching workflow metadata**

Information generated and output by AWS Entity Resolution during a matching workflow job. This information is required on output.
Normalization (ApplyNormalization)

Choose whether to normalize input data as defined in the schema. Normalization standardizes data by removing extra spaces and special characters and standardizing to lowercase format.

For example, if an input field has an input type of PHONE_NUMBER, and the values in the input table are formatted as (123) 456-7890, AWS Entity Resolution will normalize the values to 1234567890.

The following sections describe the normalization rules.

Topics
- Name (p. 92)
- Email (p. 92)
- Phone (p. 92)
- Address (p. 92)
- Hashed (p. 94)
- Source_ID (p. 94)

Name
- TRIM = Trims leading and trailing whitespace
- LOWERCASE = Lowercases all alpha characters
- CONVERT_ACCENT = Covert accented letter to regular letter
- REMOVE_ALL_NON_ALPHA = Removes all non-alpha characters [a-zA-Z]

Email
- TRIM = Trims leading and trailing whitespace
- LOWERCASE = Lowercases all alpha characters
- CONVERT_ACCENT = Covert accented letter to regular letter
- REMOVE_ALL_NON_EMAIL_CHARS = Removes all non-alpha-numeric characters [a-zA-Z0-9] and [.@-]

Phone
- TRIM = Trims leading and trailing whitespace
- REMOVE_ALL_NON_NUMERIC = Removes all non-numeric characters [0-9]
- REMOVE_ALL_LEADING_ZEROES = Removes all leading zeroes

Address
- TRIM = Trims leading and trailing whitespace
- LOWERCASE = Lowercases all alpha characters
- CONVERT_ACCENT = Covert accented letter to regular letter
• REMOVE_ALL_NON_ALPHA = Removes all non-alpha characters [a-zA-Z]
• RENAME_WORDS using ADDRESS_RENAME_WORD_MAP = replace words in Address string with words from ADDRESS_RENAME_WORD_MAP (p. 93)
• RENAME_DELIMITERS using ADDRESS_RENAME_DELIMITER_MAP = replace delimiters in Address string with string from ADDRESS_RENAME_DELIMITER_MAP (p. 93)
• RENAME_DIRECTIONS using ADDRESS_RENAME_DIRECTION_MAP = replace delimiters in Address string with string from ADDRESS_RENAME_DIRECTION_MAP (p. 94)
• RENAME_NUMBERS using ADDRESS_RENAME_NUMBER_MAP = replace numbers in Address string with string from ADDRESS_RENAME_NUMBER_MAP (p. 93)
• RENAME_SPECIAL_CHARS using ADDRESS_RENAME_SPECIAL_CHAR_MAP = replace special characters in Address string with string from ADDRESS_RENAME_SPECIAL_CHAR_MAP (p. 93)

ADDRESS_RENAME_WORD_MAP

These are the words that will be renamed when normalizing the address string.

```
"avenue": "ave",
"bouled": "blvd",
"circle": "cir",
"circles": "cirs",
"court": "ct",
"centre": "ctr",
"center": "ctr",
"drive": "dr",
"freeway": "fwy",
"frwy": "fwy",
"highway": "hwy",
"lane": "ln",
"parks": "park",
"parkways": "pkwy",
"pky": "pkwy",
"pkway": "pkwy",
"pkwy": "pkwy",
"parkway": "pkwy",
"parkwy": "pkwy",
"place": "pl",
"plaza": "plz",
"plza": "plz",
"road": "rd",
"square": "sq",
"squa": "sq",
"sqr": "sq",
"street": "st",
"str": "st",
"str.": "strasse"
```

ADDRESS_RENAME_DELIMITER_MAP

These are the delimiters that will be renamed when normalizing the address string.

```
": " 
": " 
": " 
": " 
": " 
": " 
": "number "
```
ADDRESS_RENAME_DIRECTION_MAP

These are the direction identifiers that will be renamed when normalizing the address string.

```
"east": "e",
"north": "n",
"south": "s",
"west": "w",
"northeast": "ne",
"northwest": "nw",
"southeast": "se",
"southwest": "sw"
```

ADDRESS_RENAME_NUMBER_MAP

These are the number strings that will be renamed when normalizing the address string.

```
"número": "number",
"numero": "number",
"no": "number",
"núm": "number",
"num": "number"
```

ADDRESS_RENAME_SPECIAL_CHAR_MAP

These are the special characters string that will be renamed when normalizing the address string.

```
"ß": "ss",
"ä": "ae",
"ö": "oe",
"ü": "ue",
"ø": "o",
"æ": "ae"
```

Hashed

- **TRIM** = Trims leading and trailing whitespace

Source_ID

- **TRIM** = Trims leading and trailing whitespace

One-to-One matching

One-to-one matching compares single instances of similar data. Input fields with the same match key and values in the same input field will be matched against each other.

For example, you might have multiple phone number input fields like `mobile_phone` and `home_phone` that have the same match key "Phone". Use one-to-one matching to compare data in the `mobile_phone` input field with data in the `mobile_phone` input field and to compare data in the `home_phone` input field with data in the `home_phone` input field. Data in the `mobile_phone` input field won't be compared with data in the `home_phone` input field.
Matching rules evaluate data in multiple input fields with the same match key with an (or) operation, and one-to-many matching compares values within a single input field. This means that if `mobile_phone` or `home_phone` matches between two records, the “Phone” match key will return a match. For match key “Phone” to find a match, `Record One mobile_phone = Record Two mobile_phone OR Record One home_phone = Record Two home_phone`.

Matching rules evaluate data in input fields with different match keys with an (and) operation. If you want rule-based matching to consider different types of phone number information completely separately, you can create more specific match keys like “mobile_phone” and “home_phone”. If you want to use both match keys in a rule to find matches, `Record One mobile_phone = Record Two mobile_phone AND Record One home_phone = Record Two home_phone`.

Output

A list of `OutputAttribute` objects, each of which have the fields `Name` and `Hashed`. Each of these objects represent a column to be included in the AWS Glue output table and whether you want the values in the column to be hashed.

OutputS3Path

The S3 destination to which AWS Entity Resolution will write the output table.

OutputSourceConfig

A list of `OutputSource` objects, each of which have the fields `OutputS3Path`, `ApplyNormalization`, and `Output`.

Provider service-based matching

Provider service-based matching is process designed to match, link, and enhance your records with preferred data service providers and licensed data sets. You must have a subscription through AWS Data Exchange with the provider service to use this matching technique.

AWS Entity Resolution currently integrates with the following data service providers:

- LiveRamp
- TransUnion
- UID 2.0

Rule-based matching

Rule-based matching is process designed to find exact matches. Rule-based matching is a hierarchical set of waterfall matching rules, suggested by AWS Entity Resolution, based upon the data that you input and completely configurable by you. All match keys provided within rule criteria must match exactly for compared data to be declared a match and for associated metadata to be output. Rule-based matching returns a `Match ID (p. 90)` and a rule number for each matched set of data.

We recommend defining rules that can uniquely identify an entity. Order your rules to find more precise matches first.
For example, let’s say you have two rules, Rule 1 and Rule 2.

These rules have the following match keys:

- **Rule 1** includes Full name and Address
- **Rule 2** includes Full name, Address, and Phone

Because **Rule 1** runs first, no matches will be found by **Rule 2** because they would have all been found by **Rule 1**.

To find matches that are differentiated by Phone, reorder the rules, like this:

- **Rule 2** includes Full name, Address, and Phone
- **Rule 1** includes Full name and Address

### Schema

The term used for a structure or layout defining how a set of data is organized and connected.

### Schema description

An optional description of the schema that you can choose to enter. Descriptions help you differentiate between schema mappings if you create more than one.

### Schema name

The name of the schema.

**Note**

Schema names must be unique. They can't have the same name or an error will be returned.

### Schema mapping

Schema mapping in AWS Entity Resolution is the process by which you tell AWS Entity Resolution how to interpret your data for matching. You define the schema of the input data table that you want AWS Entity Resolution to read into a matching workflow.

### Schema mapping ARN

The Amazon Resource Name (ARN) generated for the [schema mapping](#).

### Unique ID

A unique identifier that you designate and that must be assigned to each row of input data that AWS Entity Resolution reads.
Example

For example: **Primary_key**, **Row_ID**, or **Record_ID**.

The **Unique ID** column is required and must be a unique identifier across all sources.

When the [matching workflow](p. 91) is run, the record will be rejected if the **Unique ID** is:

- not specified
- is not unique within the same source or across multiple sources
- overlaps across sources.