How to store Denodo logs in Amazon S3

Revision 20230324

NOTE
This document is confidential and proprietary of **Denodo Technologies**.
No part of this document may be reproduced in any form by any means without prior written authorization of **Denodo Technologies**.

CONTENTS

1 INTRODUCTION

By default, the Denodo components store the log files on the local filesystem. However, you can configure them to store the log files on Amazon AWS S3 as well. This is useful for Denodo deployments that run on AWS and that you plan on switching off when you no longer need them. With this feature, you make sure that the logging information is saved even if the instance is deleted.

This feature was added in Denodo 7.0 update 20190903 and it has been added for the Solution Manager in Denodo 8.0 update 20220815.

2 CONFIGURATION

2.1 AUTHENTICATION

You have to provide credentials to interact with S3 in order to allow you to send logs to the bucket.

The recommended ways are:

a. Use instance profiles with an IAM role (when working with EC2 instances).

b. Configure AWS credentials in the operating system.

2.1.1 Create IAM role and associate it to instance at launch time

The AWS documentation about instance profile roles is available here.

In order to give the instances the needed permissions you have to:

1. Create a security policy to access the bucket.

2. Create an IAM role.

3. Launch the instance with the desired role / update instance to work with that role.

2.1.1.1 Create security policy

Follow these steps to create security policy:

1. Open the policies panel in IAM console and click **Create policy**.

2. In the **JSON** tab, paste and edit the following JSON updating the bucket ARN with your bucket:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:CreateBucket",
                "s3:HeadBucket",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::your-bucket-name/*"
            ]
        }
    ]
}
```

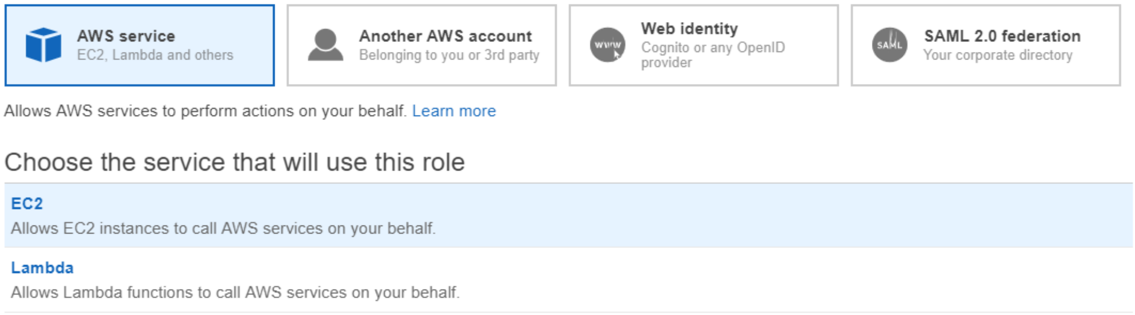
3. Click **Review policy**, enter the name and description and create the policy.

2.1.1.2 Create IAM role

Follow these steps to create an IAM role:

1. Open the roles panel in the IAM console, on the left hand side click on **Roles** and click **Create role**.

2. Select **EC2** and click **Next: permissions**.



Allows AWS services to perform actions on your behalf. Learn more

Choose the service that will use this role

EC2
Allows EC2 instances to call AWS services on your behalf.

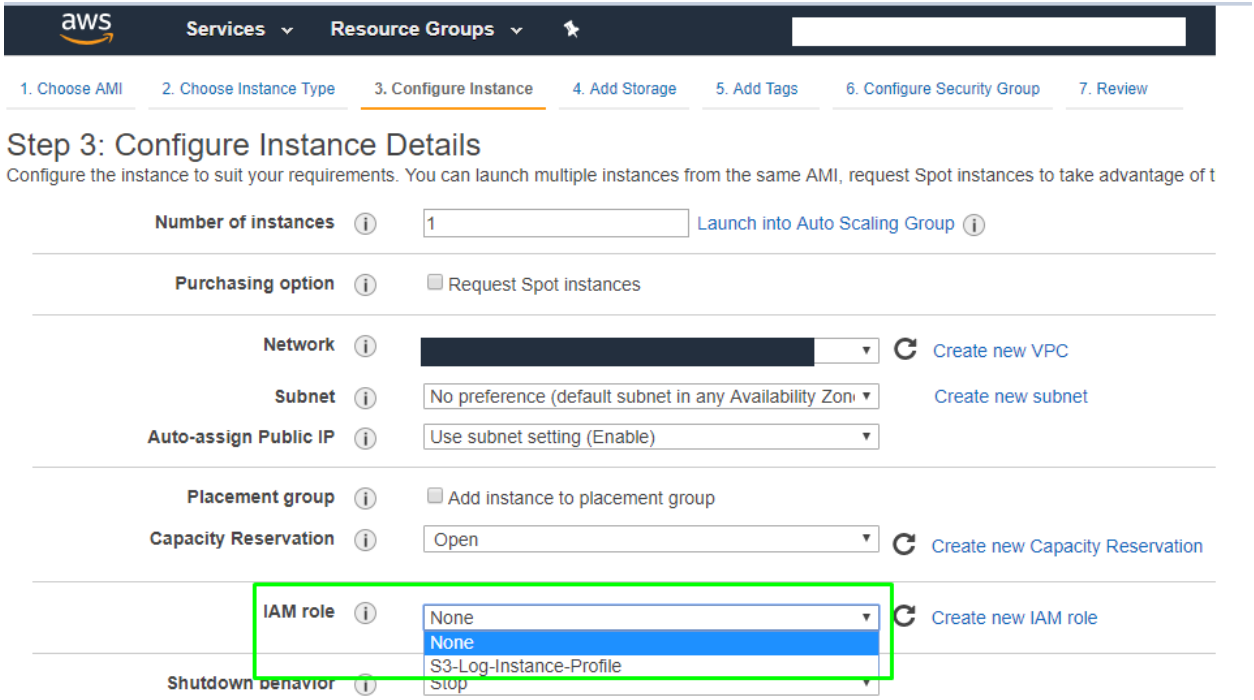
Lambda
Allows Lambda functions to call AWS services on your behalf.

3. Select the policy created in the previous step.

4. In the following dialogs, enter the tags information if you want, provide the Role name and description and create the role.

2.1.1.3 Launch instance using IAM role

When you launch an instance normally, in order to launch the new instances with permissions, select the corresponding role in the step “Configure instance”.



Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of t

Number of instances: 1 — Launch into Auto Scaling Group

Purchasing option: □ Request Spot instances

Network: [redacted] — Create new VPC

Subnet: No preference (default subnet in any Availability Zon — Create new subnet

Auto-assign Public IP: Use subnet setting (Enable)

Placement group: □ Add instance to placement group

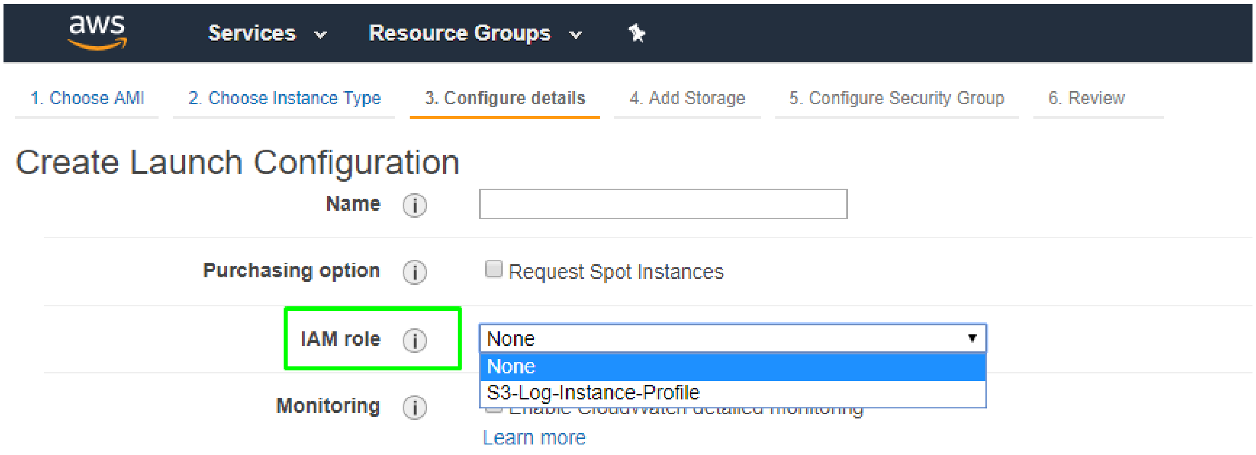
Capacity Reservation: Open — Create new Capacity Reservation

IAM role: None (None / S3-Log-Instance-Profile) — Create new IAM role

Shutdown behavior: Stop

If you use autoscaling, you can specify the IAM role:

- Selecting the “IAM role” if you use a launch configuration:



Create Launch Configuration

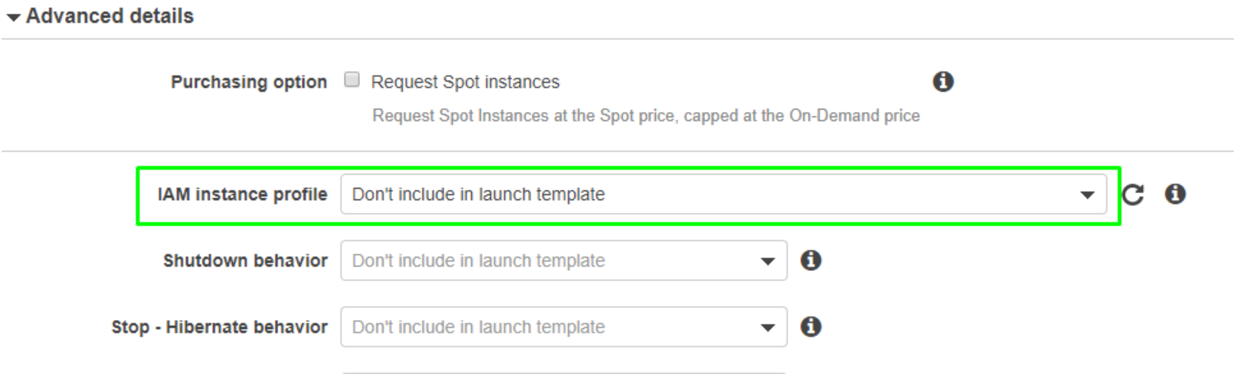
Name:

Purchasing option: □ Request Spot Instances

IAM role: None (None / S3-Log-Instance-Profile)

Monitoring: □ Enable CloudWatch detailed monitoring — Learn more

- Selecting the “IAM instance profile” in “Advanced details” section if you use a launch template:



Purchasing option □ Request Spot instances

Request Spot Instances at the Spot price, capped at the On-Demand price

IAM instance profile: Don't include in launch template

Shutdown behavior: Don't include in launch template

Stop - Hibernate behavior: Don't include in launch template

2.1.2 AWS credentials

The appender also allows to configure the access and secret keys with the properties previously described inside the appender section:

a. **s3AwsKey**: access key.

b. **s3AwsSecret**: secret key.

When s3AwsKey and s3AwsSecret are present, **they have precedence over other possible authentication methods** defined in the credentials chain.

This method is available, but **it is recommended to use the instance profile role option**.

2.2 LOG4J2.XML FILE CONFIGURATION

To store the log files of a Denodo server on an S3 bucket you need to modify the $Denodo_Home/conf/vdp/log4j2.xml file of the Denodo component.

Add the S3Appender inside the “Appenders” section:

```
<S3Appender                                                name="S3Appender">
       <PatternLayout  pattern="%-4r  [%t]  %-5p  %d{yyyy-MM-dd'T'HH:mm:ss.SSS}
%c             %x             -             %m                         %n"/>
      <s3Bucket>bucket-name</s3Bucket>
      <s3Path>%instanceId/desiredPath/</s3Path>
                                 <stagingBufferSize>2500</stagingBufferSize>
                                     <stagingBufferAge>60</stagingBufferAge>
      <s3Region>eu-central-1</s3Region>
</S3Appender>
```

Add the appender to the Root logger:

```
<Root                                                         level="error">
                                     <AppenderRef      ref="FILEOUT"       />
```

```
                                   <AppenderRef      ref="S3Appender"      />
</Root>
```

The following properties control how the logs are stored in S3:

- **s3Bucket**: S3 bucket to use. The appender will attempt to create the bucket if it does not exist.
- **s3Path**: path where each log file will be stored inside the S3 bucket. You can specify the *%instanceId* keyword in the path. The appender will replace this value with the instance id of the instance in AWS.

With these properties you can control when the appender will send the logs to the S3 bucket:

- **stagingBufferSize**: Number of log messages after which the content is written to the S3 bucket.
- **stagingBufferAge**: Number of minutes after which the content is written to the S3 bucket. If this property is defined, stagingBufferSize property value is ignored.

There is also the following connection property:

- **s3Region**: identifier of the region. The specified region will be used to interact with S3.

The following access properties are optional:

a. **s3AwsKey**: AWS access key to access AWS services / perform some API operations.
b. **s3AwsSecret**: AWS secret key to access AWS services / perform some API operations.

We recommend using instance profile roles instead of configuring AWS credentials directly in the log4j2.xml file.

NOTE: to log the server start / stop messages, you will also need to add these loggers categories in the `Loggers` section:

```
<Logger            name="server.start"            level="info"            />
<Logger name="server.stop" level="info" />
```

The log4j2.xml files are available in the following paths in a Denodo Platform installation for each type of server:

- Virtual DataPort server: `<DENODO_HOME>/conf/vdp/log4j2.xml`
- Scheduler server: `<DENODO_HOME>/conf/scheduler/log4j2.xml`
- Scheduler Index: `<DENODO_HOME>/conf/arn-index/log4j2.xml`

In each case, you will need to stop the corresponding server, make the changes and start the server with the new appender configured.