title: Log Archiving description: Archiving logs to an S3-compatible object store

You can configure each Logs App to compress and store all logs it receives to an S3-compatible object store.

### How do I configure archiving for my Logs App?

In Sematext, go to `Logs` view and choose `Archive Logs` in the App context menu (*three-dots* icon) of the App whose logs you want to ship to an S3-compatible object store:

For Amazon S3, all you have to provide are credentials and a bucket name:

For any other S3-compatible object store you'll also have to provide a service endpoint. Cloud object store providers, like *DigitalOcean Spaces* or *IBM Cloud Object Storage* usually refer to it as a `Public Service Endpoint` in bucket configuration. For Minio users this would be your Minio public URL.
Note that this is available in Sematext Enterprise, too.

Paste `Access key ID` and `Secret access key` in the corresponding fields.

Enter `Bucket name` (just the simple name, not fully qualified ARN) and choose `Compression` (read on for more details about compression) and confirm with `Verify and save`.

At this point, your Logs App is going to check whether the information is valid using the AWS S3 API.

After the check is done you'll see a feedback message confirming information validity or an error message.

### How to obtain credentials from AWS?

For our Logs App AWS S3 Settings, besides S3 bucket name, you'll need `Access Key ID` and `Secret Access Key`.

Log in to your AWS account, go to `IAM > Users` and open (or create) a user that you want to use for S3 uploads:

Click on `Create Access Key`:

Note down `Access Key ID` and `Secret Access Key` (you can `Download Credentials` to a safe place if you like, but it's not necessary).

### Which credentials are required when using AWS S3 Bucket Access Policy?

In order to verify access to your S3 bucket, the Logs App will first use the credentials to log in and, if successful, proceed to create a dummy file inside the bucket.

If the file creation was successful Logs App will attempt to delete it.

For these reasons, the following credentials must be given to the bucket when saving AWS S3 settings:

- `s3:GetObject`
- `s3:PutObject`
- `s3:DeleteObject`

After the verification is done you can remove the `s3:DeleteObject` permission from the bucket policy.

**NOTE:** if deletion of the dummy object fails the Logs App will ignore it and conclude that it can start shipping logs.

### How are logs compressed in S3?

You have the option of choosing between two modern, lossless compression codecs from the `LZ77` family, with excellent speed/compression ratio, `LZ4` and `LZF`.

If you choose `No compression` option, logs will be stored in raw, uncompressed format, as JSON files.

### How can I decompress logs archived in S3?

You can decompress logs by installing these command line programs (then use`man lz4`or`man lzf` for further instructions):

### Ubuntu/Debian

```
sudo apt-get install liblz4-tool
sudo apt-get install libcompress-lzf-java (landed in Ubuntu 15.04)
```

### OSX

```
brew install lz4
brew install liblzf
```

### Which folder structure is used when uploading logs to S3?

Inside a bucket that you specify in settings, the following folder hierarchy is created:

`sematext_[app-token-start]/[year]/[month]/[day]/[hour]`

Where `[app-token-start]` is the first sequence of your Logs App's token.

For example, an App whose token begins with `f333a7d7` will have a folder with the following path on `May 01, 2021 at 11:20PM UTC`:

`sematext_f333a7d7/2021/05/01/23/`

(warning) **Note:**

Before **May 01, 2017** the folder hierarchy was more flat:

`/<tokenMD5HexHash>/logsene_<date>/<hour>`

For example: `856f4f9c3c084da08ec7ea9ad5d4cadf/logsene_2016-07-20/18`