MinIO: Getting started

How to start using MinIO

> 👋**Welcome on Stackhero's documentation!**
> Stackhero provides MinIO instances that are ready for production in just 2 minutes!
> Including TLS encryption (aka HTTPS), customizable domain name, unlimited messages and size, backups and updates in just a click.
> Try our managed MinIO cloud in just 2 minutes

MinIO is an object storage service compatible with S3 API.

All S3 clients can connect to MinIO and there is a S3 client library for almost every language out there, including Ruby, Node.js, Java, Python, Clojure and Erlang.

You can also use the MinIO SDKs. This is the recommended way to connect to your object storage.
You will find all the available SDKs on MinIO official documentation.

Another way to connect is to use the MinIO CLI interface. This is very helpful when you want to send big files directly from your terminal.
You will find all the informations about it on the MinIO CLI official documentation.

Use the MinIO CLI

The MinIO Console WEB UI is installed out-of-the-box with your **Stackhero for MinIO** instance.
For advanced users and usages, you can use the MinIO CLI in place.

The easiest way to use the MinIO CLI is to run its Docker container.
Start the container: `docker run -it --entrypoint=/bin/sh minio/mc`
Once in the container, add your server: `mc alias set minio https://<STACKHERO_MINIO_HOST> <STACKHERO_MINIO_ROOT_ACCESS_KEY> <STACKHERO_MINIO_ROOT_SECRET_KEY>`

Create a bucket "test" to check if it works: `mc mb minio/test`
Finally list your buckets: `mc ls minio`

To get more help about MinIO CLI commands, you can use the command `mc --help`

Create a user

A user is identified by an access key and a secret key that you will create.

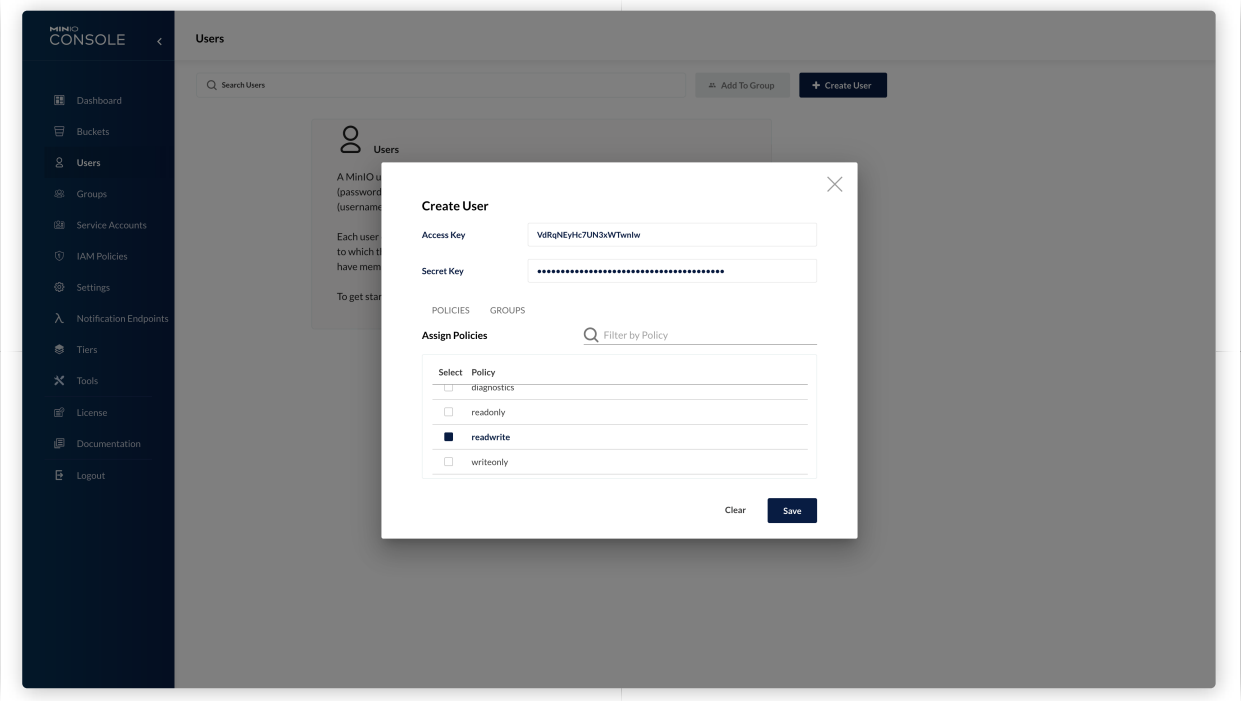
There is two possibilities to create a user: use the MinIO web console or use the MinIO CLI client.

Create a user using the MinIO web console

To create a user on MinIO, you first have to connect to your Stackhero dashboard.

Select your MinIO service and click on the `Console` link.
Identify yourself with "root" credentials.
You can retrieve them in the Stackhero dashboard, by clicking on the `Configure` button.

Once connected to the MinIO console, click on `Identity`, `Users`, then `Create a user`.
Define an `access key` (username), a `secret key` (password) and select the policies you want (probably "readwrite").



Creating a MinIO user

Once the user is created, you will be able to use these `access key` and `secret key` in your app.

Connect to MinIO from Java

Here is an example of a connection using the MinIO SDK:

```
MinioClient minioClient =
  MinioClient.builder()
    .endpoint("https://XXXXXX.stackhero-network.com")
    .credentials("YOUR_ACCESS_KEY", "YOUR_SECRET_KEY")
    .build();
```

You will find more information on the official MinIO Java SDK documentation.

Connect to MinIO from Ruby

Install the AWS SDK gem:

```
$ gem 'aws-sdk'
$ bundle install
```

Here is a simple example of how to use MinIO via AWS SDK on Ruby:

```
require 'aws-sdk'

Aws.config.update(
  endpoint: 'https://XXXXXX.stackhero-network.com',
  access_key_id: 'YOUR_ACCESS_KEY',
  secret_access_key: 'YOUR_SECRET_KEY',
  force_path_style: true,
  region: 'us-east-1'
)

rubys3_client = Aws::S3::Client.new

# put_object operation

rubys3_client.put_object(
  key: 'testobject',
  body: 'Hello from MinIO!!',
  bucket: 'testbucket',
```

```
  content_type: 'text/plain'
)

# get_object operation

rubys3_client.get_object(
  bucket: 'testbucket',
  key: 'testobject',
  response_target: 'download_testobject'
)

print "Downloaded 'testobject' as  'download_testobject'. "
```

You will find more information on the official MinIO Ruby documentation.

Connect to MinIO from Python

Connect to MinIO with the AWS SDK (boto)

Install the AWS SDK (boto) package:

```
$ pip install boto3
$ pip freeze > requirements.txt
```

Here is an example on how to Connect to MinIO from Python and AWS SDK (boto) package:

```
#!/usr/bin/env/python
import os
import boto3
from botocore.client import Config

s3 = boto3.resource('s3',
                    endpoint_url='https://XXXXXX.stackhero-network.com',
                    aws_access_key_id='YOUR_ACCESS_KEY',
                    aws_secret_access_key='YOUR_SECRET_KEY',
                    config=Config(signature_version='s3v4'),
                    region_name='us-east-1')

# upload a file from local file system '/home/john/piano.mp3' to bucket 'songs'
with 'piano.mp3' as the object name.
s3.Bucket('songs').upload_file('/home/john/piano.mp3','piano.mp3')

# download the object 'piano.mp3' from the bucket 'songs' and save it to local
FS as /tmp/classical.mp3
```

```
s3.Bucket('songs').download_file('piano.mp3', '/tmp/classical.mp3')

print "Downloaded 'piano.mp3' as  'classical.mp3'. "
```

You will find more information on the official MinIO Python documentation.

Connect to MinIO with the MinIO SDK

Install the MinIO package:

```
$ pip install minio
$ pip freeze > requirements.txt
```

Here is an example on how to Connect to MinIO from Python:

```
#!/usr/bin/env/python
from minio import Minio
from minio.error import S3Error

def main():
  # Create a client with the MinIO server playground, its access key
  # and secret key.
  client = Minio(
    endpoint='xxxxxx.stackhero-network.com:443,
    secure=True,
    access_key='YOUR_ACCESS_KEY',
    secret_key='YOUR_SECRET_KEY'
  )

  # Make 'asiatrip' bucket if not exist.
  found = client.bucket_exists("asiatrip")
  if not found:
    client.make_bucket("asiatrip")
  else:
    print("Bucket 'asiatrip' already exists")

  # Upload '/home/user/Photos/asiaphotos.zip' as object name
  # 'asiaphotos-2015.zip' to bucket 'asiatrip'.
  client.fput_object(
    "asiatrip", "asiaphotos-2015.zip", "/home/user/Photos/asiaphotos.zip",
  )
  print(
    "'/home/user/Photos/asiaphotos.zip' is successfully uploaded as "
    "object 'asiaphotos-2015.zip' to bucket 'asiatrip'."
  )
```

```
if __name__ == "__main__":
  try:
    main()
  except S3Error as exc:
    print("error occurred.", exc)
```

You will find more information on the official MinIO Python SDK documentation.

Connect to MinIO from PHP

Install the AWS SDK for PHP package:

```
$ composer require aws/aws-sdk-php
```

Here is an example on how to Connect to MinIO from PHP and AWS SDK:

```
<?php

// Include the SDK using the Composer autoloader
date_default_timezone_set('America/Los_Angeles');
require 'vendor/autoload.php';

$s3 = new Aws\S3\S3Client([
  'version' => 'latest',
  'region'  => 'us-east-1',
  'endpoint' => 'https://XXXXXX.stackhero-network.com',
  'use_path_style_endpoint' => true,
  'credentials' => [
    'key'    => 'YOUR_ACCESS_KEY',
    'secret' => 'YOUR_SECRET_KEY'
  ],
]);


// Send a PutObject request and get the result object.
$insert = $s3->putObject([
  'Bucket' => 'testbucket',
  'Key'    => 'testkey',
  'Body'   => 'Hello from MinIO!!'
]);

// Download the contents of the object.
```

```
$retrive = $s3->getObject([
  'Bucket' => 'testbucket',
  'Key'    => 'testkey',
  'SaveAs' => 'testkey_local'
]);

// Print the body of the result by indexing into the result object.
echo $retrive['Body'];
```

You will find more information on the official MinIO PHP documentation.

Connect to MinIO from Go

Connect to MinIO with the AWS SDK

Install the AWS SDK package for GO:

```
$ go get github.com/aws/aws-sdk-go-v2
$ go get github.com/aws/aws-sdk-go-v2/config
```

Here is an example on how to Connect to MinIO from GO and AWS SDK:

```
package main

import (
  "fmt"
  "os"
  "strings"

  "github.com/aws/aws-sdk-go/aws"
  "github.com/aws/aws-sdk-go/aws/credentials"
  "github.com/aws/aws-sdk-go/service/s3/s3manager"
  "github.com/aws/aws-sdk-go/aws/session"
  "github.com/aws/aws-sdk-go/service/s3"
)

func main() {
  bucket := aws.String("newbucket")
  key := aws.String("testobject")

  // Configure to use MinIO Server
  s3Config := &aws.Config{
    Credentials:      credentials.NewStaticCredentials("YOUR_ACCESS_KEY",
"YOUR_SECRET_KEY", ""),
```

```
    Endpoint:         aws.String("https://XXXXXX.stackhero-network.com"),
    Region:           aws.String("us-east-1"),
    DisableSSL:       aws.Bool(false),
    S3ForcePathStyle: aws.Bool(true),
  }
  newSession := session.New(s3Config)

  s3Client := s3.New(newSession)

  cparams := &s3.CreateBucketInput{
    Bucket: bucket, // Required
  }

  // Create a new bucket using the CreateBucket call.
  _, err := s3Client.CreateBucket(cparams)
  if err != nil {
    // Message from an error.
    fmt.Println(err.Error())
    return
  }

  // Upload a new object "testobject" with the string "Hello World!" to our
"newbucket".
  _, err = s3Client.PutObject(&s3.PutObjectInput{
    Body:   strings.NewReader("Hello from MinIO!!"),
    Bucket: bucket,
    Key:    key,
  })
  if err != nil {
    fmt.Printf("Failed to upload data to %s/%s, %s\n", *bucket, *key,
err.Error())
    return
  }
  fmt.Printf("Successfully created bucket %s and uploaded data with key %s\n",
*bucket, *key)

  // Retrieve our "testobject" from our "newbucket" and store it locally in
"testobject_local".
  file, err := os.Create("testobject_local")
  if err != nil {
    fmt.Println("Failed to create file", err)
    return
  }
  defer file.Close()

  downloader := s3manager.NewDownloader(newSession)
  numBytes, err := downloader.Download(file,
  &s3.GetObjectInput{
    Bucket: bucket,
    Key:    key,
  })
```

```
  if err != nil {
    fmt.Println("Failed to download file", err)
    return
  }
  fmt.Println("Downloaded file", file.Name(), numBytes, "bytes")
}
```

You will find more information on the official MinIO GO documentation.

Connect to MinIO with the MinIO SDK

Install the MinIO SDK package for GO:

```
$ GO111MODULE=on go get github.com/minio/minio-go/v7
```

Here is an example on how to Connect to MinIO from GO and MinIO SDK:

```
package main

import (
  "log"

  "github.com/minio/minio-go/v7"
  "github.com/minio/minio-go/v7/pkg/credentials"
)

func main() {
  endpoint := "xxxxxx.stackhero-network.com"
  accessKeyID := "YOUR_ACCESS_KEY"
  secretAccessKey := "YOUR_SECRET_KEY"
  useSSL := true


  // Initialize minio client object.
  minioClient, err := minio.New(endpoint, &minio.Options{
    Creds:  credentials.NewStaticV4(accessKeyID, secretAccessKey, ""),
    Secure: useSSL,
  })
  if err != nil {
    log.Fatalln(err)
  }

  log.Printf("%#v\n", minioClient) // minioClient is now set up
}
```

You will find more information on the official MinIO GO SDK documentation.

Connect to MinIO from Node.js

Connect to MinIO with the AWS SDK

Install the AWS SDK package for Node.js:

```
$ npm install --save aws-sdk
```

Here is an example on how to Connect to MinIO from Node.js and MinIO SDK:

```
const process = require('process');
const AWS = require('aws-sdk');

const s3  = new AWS.S3({
  accessKeyId: 'YOUR_ACCESS_KEY',
  secretAccessKey: 'YOUR_SECRET_KEY',
  endpoint: 'https://XXXXXX.stackhero-network.com',
  s3ForcePathStyle: true, // needed with minio?
  signatureVersion: 'v4'
});

// putObject operation.
s3.putObject(
  { Bucket: 'testbucket', Key: 'testobject', Body: 'Hello from MinIO!!' },
  (err, data) => {
    if (err)
      console.log(err)
    else
      console.log('Successfully uploaded data to testbucket/testobject');
  }
);

// getObject operation.
const file = require('fs').createWriteStream('/tmp/mykey');
s3.getObject({ Bucket: 'testbucket', Key: 'testobject' })
  .on('httpData', chunk => file.write(chunk))
  .on('httpDone', () => file.end())
  .send();
```

You will find more information on the official MinIO Node.js documentation.

Connect to MinIO with the MinIO SDK

Install the MinIO SDK package for Node.js:

```
$ npm install --save minio
```

Here is an example on how to Connect to MinIO from Node.js and MinIO SDK:

```
const process = require('process');
const Minio = require('minio');

// Instantiate the minio client with the endpoint
// and access keys as shown below.
const minioClient = new Minio.Client({
  endpoint: 'xxxxxx.stackhero-network.com',
  port: 443,
  useSSL: true,
  accessKeyId: 'YOUR_ACCESS_KEY',
  secretAccessKey: 'YOUR_SECRET_KEY',
});

// File that needs to be uploaded.
const file = '/tmp/photos-europe.tar'

// Make a bucket called europetrip.
minioClient.makeBucket(
  'europetrip',
  'us-east-1',
  err => {
    if (err) return console.log(err)

    console.log('Bucket created successfully in "us-east-1".')

    const metaData = {
      'Content-Type': 'application/octet-stream',
      'X-Amz-Meta-Testing': 1234,
      'example': 5678
    }
    // Using fPutObject API upload your file to the bucket europetrip.
    minioClient.fPutObject(
      'europetrip',
      'photos-europe.tar',
      file,
      metaData,
      (err, etag) => {
        if (err) return console.log(err)
```

```
        console.log('File uploaded successfully.')
      }
    );
  }
);
```

Our

- Directus
- Docker
- Elasticsearch
- GitLab
- Grafana
- Graylog
- InfluxDB
- Kibana
- MariaDB
- Matomo
- Mattermost
- Memcached
- Mercure-Hub
- MinIO
- Mosquitto
- MySQL
- Nextcloud
- Node-RED
- Node.js
- PHP
- Postfix
- PostgreSQL
- Prometheus
- Python
- RabbitMQ
- Redis®*
- RethinkDB