Swarm Storage HowTo!

Πίνακας περιεχομένων

1. Install Minio

1.1. Create Docker secrets for MinIO

create secrets

```
KEY=$(od -vN 32 -An -tx1 /dev/urandom | tr -d ' \n' ; echo)
SECRET=$(od -vN 32 -An -tx1 /dev/urandom | tr -d ' \n' ; echo)
echo $KEY > key
echo $SECRET > secret
echo $KEY | docker secret create access_key -
echo $SECRET | docker secret create secret_key -
```

1.2. Create node labels

create labels

```
docker node update --label-add minio1=true [node-name] ①
docker node update --label-add minio2=true [node-name]
docker node update --label-add minio3=true [node-name]
docker node update --label-add minio4=true [node-name]

docker node update --label-add group=minio [node-name] ②
docker node update --label-add group=minio [node-name]
```

① node name from command: docker node ls e.g. **snf-12118** (minio)

② node name from command: docker node ls e.g. **snf-12118** (proxy)

1.3. Generate a Certificate

Create a configuration file (openssl.conf)

```
[req]
distinguished_name = req_distinguished_name
x509_extensions = v3_req
prompt = no

[req_distinguished_name]
C = US ①
ST = VA ①
L = Somewhere ①
O = MyOrg ①
OU = MyOU ①
CN = MyServerName ①

[v3_req]
subjectAltName = @alt_names

[alt_names]
IP.1 = 127.0.0.1 ②
```

① change to the correct values

② change to the correct IP address

Run openssl and specify the configuration file

```
openssl req -x509 -nodes -days 730 -newkey rsa:2048 -keyout private.key -out
public.crt -config openssl.conf
```

1.4. Create Yaml file

docker-compose

```yaml
services:
  minio1: ①
    image: minio/minio:RELEASE.2020-04-10T03-34-42Z ②
    hostname: minio1
    volumes:
      - minio1-data:/export ③
    ports:
      - "9001:9000" ④
    networks:
      - minio_distributed ⑤
    deploy:
      restart_policy:
        delay: 10s
        max_attempts: 10
        window: 60s
      placement:
        constraints:
          - node.labels.minio1==true ⑥
    command: server http://minio{1...4}/export ⑦
    secrets: ⑧
      - secret_key
      - access_key
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:9000/minio/health/live"] ⑨
      interval: 30s
      timeout: 20s
      retries: 3

  minio2: ⑩
    image: minio/minio:RELEASE.2020-04-10T03-34-42Z
    hostname: minio2 ⑩
    volumes:
      - minio2-data:/export ⑪
    ports:
      - "9002:9000" ⑫
    networks:
      - minio_distributed ⑤
    deploy:
      restart_policy:
        delay: 10s
        max_attempts: 10
        window: 60s
      placement:
        constraints:
          - node.labels.minio2==true ⑬
    command: server http://minio{1...4}/export
    secrets:
      - secret_key
      - access_key
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:9000/minio/health/live"]
```

```
      interval: 30s
      timeout: 20s
      retries: 3

  minio3:
    image: minio/minio:RELEASE.2020-04-10T03-34-42Z
    hostname: minio3
    volumes:
      - minio3-data:/export
    ports:
      - "9003:9000"
    networks:
      - minio_distributed ⑤
    deploy:
      restart_policy:
        delay: 10s
        max_attempts: 10
        window: 60s
      placement:
        constraints:
          - node.labels.minio3==true
    command: server http://minio{1...4}/export
    secrets:
      - secret_key
      - access_key
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:9000/minio/health/live"]
      interval: 30s
      timeout: 20s
      retries: 3

  minio4:
    image: minio/minio:RELEASE.2020-04-10T03-34-42Z
    hostname: minio4
    volumes:
      - minio4-data:/export
    ports:
      - "9004:9000"
    networks:
      - minio_distributed ⑤
    deploy:
      restart_policy:
        delay: 10s
        max_attempts: 10
        window: 60s
      placement:
        constraints:
          - node.labels.minio4==true
    command: server http://minio{1...4}/export
    secrets:
      - secret_key
```

```
      - access_key
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:9000/minio/health/live"]
      interval: 30s
      timeout: 20s
      retries: 3

  web:
    image: nginx:1.17.9-alpine
    deploy:
      mode: replicated
      restart_policy:
        delay: 10s
        max_attempts: 10
        window: 60s
      replicas: 2
      placement:
        max_replicas_per_node: 1
        constraints:
          - node.labels.group==minio ⑭
    ports:
      - "8080:80"
      - "9443:443"
    volumes: ⑮
      - /PATH_to_FILE/minio.conf:/etc/nginx/conf.d/default.conf ⑯
      - /PATH_to_FILE/public.crt:/etc/nginx/public.crt ⑰
      - /PATH_to_FILE/private.key:/etc/nginx/private.key ⑰
    networks:
      - minio_distributed ⑤


volumes:
  minio1-data:

  minio2-data:

  minio3-data:

  minio4-data:


networks:
  minio_distributed: ⑤
    driver: overlay

secrets:
  secret_key:
    external: true
  access_key:
    external: true
```

① Service name

② Image name

③ Volume to Use

④ Expose port

⑤ Network to Use

⑥ Node Placement

⑦ Start server

⑧ insert secrets

⑨ health check command

⑩ **NEW** Service name

⑪ **NEW** Volume

⑫ **NEW** Port

⑬ **NEW** Label

⑭ Node Placement (Proxy)

⑮ Bind mount config files

⑯ Nginx config file

⑰ ssl keys

1.5. Create config file (proxy)

nginx config

```
upstream minio_servers {
    server minio1:9000;  ①
    server minio2:9000;  ①
    server minio3:9000;  ①
    server minio4:9000;  ①
}
proxy_cache_path /var/tmp levels=1:2 keys_zone=my_cache:10m max_size=10g inactive=60m
use_temp_path=off;
server {
    listen 80;
    server_name name.example.org; ②
    return 301 https://name.example.org$request_uri;  ③
}
server {
 listen  443 ssl;
 server_name name.example.org;

 # To allow special characters in headers
 ignore_invalid_headers off;
 # Allow any size file to be uploaded.
 # Set to a value such as 1000m; to restrict file size to a specific value
```

```
  client_max_body_size 0;
  # To disable buffering
  proxy_buffering off;

   ssl_certificate    /etc/nginx/public.crt; ④
     ssl_certificate_key /etc/nginx/private.key; ④
     ssl_protocols       TLSv1 TLSv1.1 TLSv1.2;
     ssl_ciphers         HIGH:!aNULL:!MD5;

  location / {
  proxy_cache      my_cache;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header Host $http_host;

     proxy_set_header X-NginX-Proxy true;
     proxy_ssl_session_reuse off;
     proxy_redirect off;


    proxy_connect_timeout 300;
    # Default is HTTP/1, keepalive is only enabled in HTTP/1.1
    proxy_http_version 1.1;
    proxy_set_header Connection "";
    chunked_transfer_encoding off;

    #proxy_pass http://minio1:9000; # If you are using docker-compose this would be the
hostname i.e. minio
   proxy_pass http://minio_servers; ⑤
    # Health Check endpoint might go here. See
https://www.nginx.com/resources/wiki/modules/healthcheck/
    # /minio/health/live;
  }
}
```

① Service names from yaml

② Server name or IP

③ Redirect to https

④ keys

⑤ pass to servers

1.6. Copy files to nodes

cp files

```
# copy files to proxy server
scp minio.conf user@IP:/PATH_to_FILE/minio.conf ①
scp private.key user@IP:/PATH_to_FILE/private.key ①
scp public.crt user@IP:/PATH_to_FILE/public.crt ①
```

① change **ip** (see <2> in create_node_labels) and **PATH_to_FILE** (see <16> in create_yaml_file)

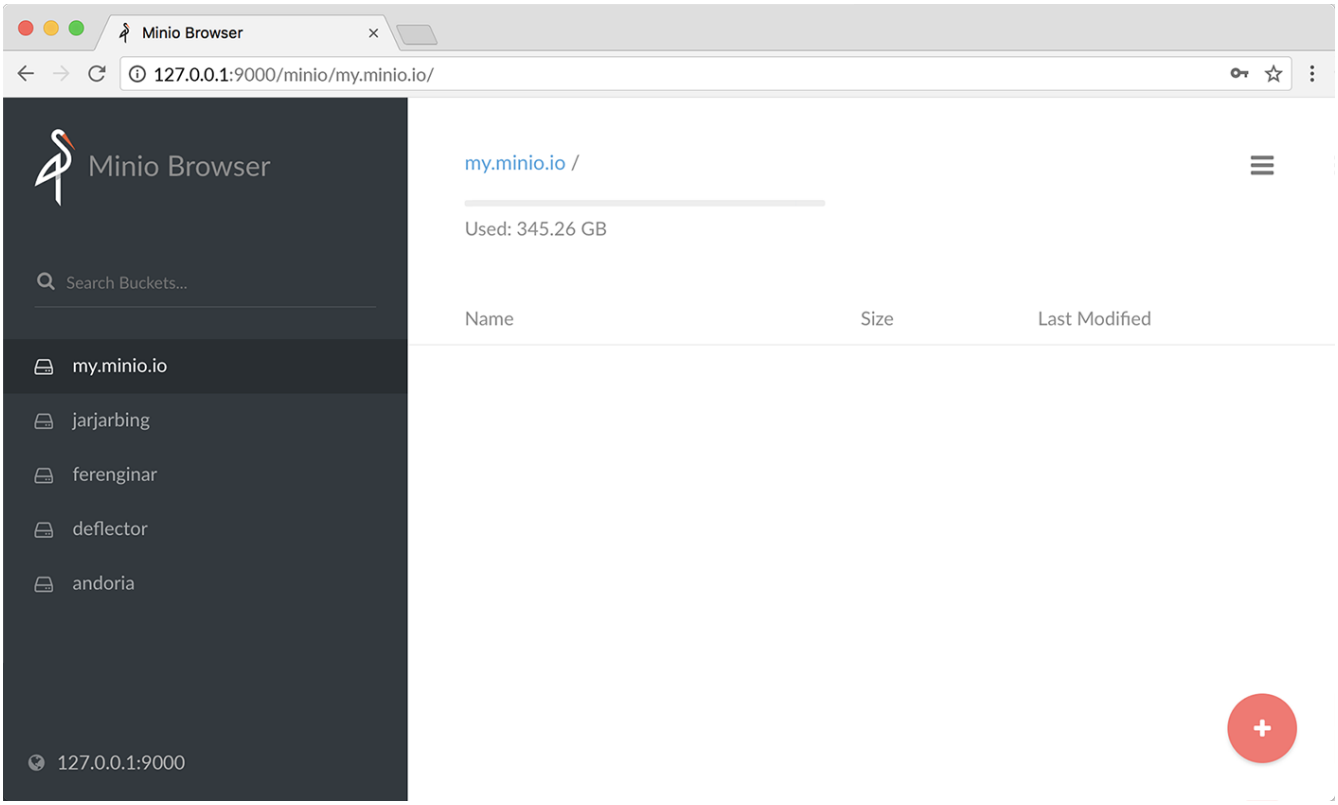
1.7. deploy

stack deploy

```
docker stack deploy --compose-file=docker-compose.yaml minio_stack
```

1.8. Test MinIO in Browser

Point your web browser to http://ip:9443



2. Install tools

2.1. Install AWS CLI

Universal Command Line Interface for Amazon Web Services

aws cli

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

The AWS Command Line Interface (AWS CLI) is an open source tool that enables you to interact with AWS services using commands in your command-line shell

create file /home/user/.aws/credentials

```
[default]
aws_secret_access_key = key ①
aws_access_key_id = secret  ①
```

① see

create file /home/user/.aws/config

```
[default]
s3 =
    signature_version = s3v4
region = us-east-1
```

2.2. Install mc client

MinIO Client (mc) provides a modern alternative to UNIX commands like ls, cat, cp, mirror, diff, find etc. It supports filesystems and Amazon S3 compatible cloud storage service (AWS Signature v2 and v4).

mc

```
wget https://dl.min.io/client/mc/release/linux-amd64/mc
chmod +x mc
./mc --help
```