TrueNAS - SCALE Create a local Backup and Send to S3 Buckets

To automate backing up your TrueNAS configuration directly to Amazon S3 or any S3-compatible cloud storage, you need to create a **local backup script** and pair it with a **TrueNAS Cloud Sync task**

TrueNAS automatically saves daily configuration databases locally to your system dataset, but exporting it safely off-site requires grabbing the full archive (with the secret seed) via a middleware command (`midclt`).

Step 1: Create a Local Dataset and Script

First, create an isolated directory on your storage pool where TrueNAS will output the configuration files before uploading them. [1]

1. Navigate to **Storage** > **Datasets** and click **Add Dataset**. Name it something distinct like `system-backups`.
2. Open a shell or connection to your TrueNAS system. Create a script named `backup_config.sh` inside that dataset folder:
   bash
   ```
   nano /mnt/YOUR_POOL/system-backups/backup_config.sh
   ```
3. Paste the following script, which requests the configuration archive from the TrueNAS API engine (`midclt`), saves it with a timestamp, and purges backups older than 30 days to avoid clutter:
   bash
   ```
   #!/bin/bash
   BACKUP_DIR="/mnt/YOUR_POOL/system-backups"
   DATE=$(date +%Y%m%d-%H%M%S)
   FILENAME="truenas-config-${DATE}.tar"

   # 1. Request the configuration download job from the SCALE API
   JOB_DATA=$(midclt call core.download "config.save" '[{"secretseed": true}]' "${FILENAME}")
   ```

```
# 2. Extract the download path cleanly using jq (returns: /_download/JOB_ID?auth_token=TOKEN)
DOWNLOAD_PATH=$(echo "$JOB_DATA" | jq -r '.[1]')

# 3. Pull the actual file directly using the exact path provided by the API
curl -s -k --output "${BACKUP_DIR}/${FILENAME}" "http://127.0.0.1${DOWNLOAD_PATH}"

# 4. Delete backups older than 30 days
find "${BACKUP_DIR}" -name "truenas-config-*.tar" -type f -mtime +30 -delete
```

4. Make the script executable:
   bash

```
chmod +x /mnt/YOUR_POOL/system-backups/backup_config.sh
```

Step 2: Automate the Script via Cron Job

Schedule TrueNAS to execute this script automatically every day before sending the files offsite.

1. Navigate to **System** > **Advanced Settings** >**Cron Jobs** and click **Add**.
2. **Description**: `Generate Daily Config Backup Archive`
3. **Command**: `/mnt/YOUR_POOL/system-backups/backup_config.sh`
4. **Run As User**: `root`
5. **Schedule**: Set it to run daily (e.g., Every day at `02:00` AM).
6. Click **Save**.

Step 3: Link Your S3 Bucket Credentials

Now, prepare TrueNAS to communicate securely with your S3 bucket. [1]

1. Go to **Credentials** > **Backup Credentials** > **Cloud Credentials** and click **Add**.
2. Name the credential (e.g., `AWS-S3-Backup`).
3. Under **Provider**, select **Amazon S3** (or generic S3 if using Wasabi, Backblaze B2, or MinIO).
4. Paste your cloud account's **Access Key ID** and **Secret Access Key**.
5. Click **Verify** to confirm the handshake with AWS is successful, then click **Save**

Step 4: Configure the Offsite S3 Cloud Sync Task

The final step uses TrueNAS's built-in replication tools to synchronize your backup folder to the cloud. [1]

1. Navigate to **Data Protection** > **Cloud Sync Tasks** and click **Add**.
2. **Description**: `Sync Config Backups to S3`
3. **Direction**: Select **PUSH** (this uploads data *to* the cloud).

4. **Transfer Mode**: Select **SYNC** (mirrors your local folder; automatically handles remote file deletion when files age out locally).
5. **Directory/Files**: Choose your local path: `/mnt/YOUR_POOL/system-backups`.
6. **Credential**: Select the `AWS-S3-Backup` credential you created in Step 3.
7. **Bucket**: Select your designated target S3 bucket from the auto-populated drop-down menu.
8. **Schedule**: Set this to run daily, at least one hour *after* the local Cron Job (e.g., Every day at `03:00` AM).
9. Click **Save**.

□□ Pro-Tip: S3 Bucket Security
Because TrueNAS configuration files contain your system's password hashes, API tokens, and private networking configurations, it is highly recommended to enable **Server-Side Encryption (SSE-S3)** and **Bucket Versioning** directly inside your AWS S3 Console. This adds a layer of defense against accidental deletion or ransomware targeting your NAS shares.

Revision #1
Created 9 August 2026 20:40:30 by Admin
Updated 24 October 2026 14:12:36 by Admin