Vaultwarden + Tailscale HTTPS + Automated Backup Setup Guide

This document provides a complete workflow for setting up **Vaultwarden** behind **Nginx** with **Tailscale HTTPS**, and implementing an automated encrypted backup system with scheduled FTP uploads and an easy restore process.

---

1. Vaultwarden Docker Setup

Create `/opt/docker/vaultwarden/docker-compose.yml`:

```yaml
services:
  vaultwarden:
    image: vaultwarden/server:latest
    container_name: vaultwarden
    restart: unless-stopped
    environment:
      DOMAIN: "https://ubuntuserver.tailXXXX.ts.net/vaultwarden"
      ROCKET_BASE: /vaultwarden
      ADMIN_TOKEN: admintoken
    volumes:
      - /opt/docker/vaultwarden:/data
    ports:
      - 8000:80
```

Deploy:

```
docker compose up -d
```

---

2. Nginx Reverse Proxy Configuration (with Tailscale HTTPS)

Edit `/etc/nginx/sites-available/vaultwarden.conf`:

```
# Map for websocket upgrade

map $http_upgrade $connection_upgrade {

    default upgrade;

    ''      close;

}

server {
    listen 443 ssl http2;
```

```
    server_name ubuntuserver.tail60802a.ts.net;

    # ---- Tailscale-issued certs ----
    ssl_certificate
/var/lib/tailscale/certs/ubuntuserver.tail60802a.ts.net.crt;
    ssl_certificate_key
/var/lib/tailscale/certs/ubuntuserver.tail60802a.ts.net.key;

    # ---- Optional: allow large uploads ----
    client_max_body_size 256M;

    # ---- Vaultwarden reverse proxy ----
    location /vaultwarden/ {
        proxy_pass http://127.0.0.1:8000;   # Port your Vaultwarden container
exposes
        proxy_http_version 1.1;

        # WebSockets for live sync
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection $connection_upgrade;

        # Standard headers
        proxy_set_header Host $host;
        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;

        # Do not rewrite the /vaultwarden prefix
        # rewrite ^/vaultwarden/(.*)$ /$1 break;
    }
}
```

Restart Nginx:

```
sudo systemctl restart nginx
```

Access your instance:

```
https://ubuntuserver.tailXXXX.ts.net/vaultwarden
```

---

3. Enable the Admin Page

Add to `docker-compose.yml` under `environment:`:

```
ADMIN_TOKEN: admintoken
```

Access admin page:

`https://ubuntuserver.tailXXXX.ts.net/vaultwarden/admin`

---

4. SMTP Email Configuration

In the admin panel → *SMTP Settings*:

Field	Example
Host	smtp.gmail.com
Secure SMTP	starttls
Port	587
From Address	youremail@gmail.com
From Name	Vaultwarden
Username	youremail@gmail.com
Password	app password
SMTP Auth mechanism	plain

---

5. GPG Passphrase File

Create and secure your encryption key:

```
sudo nano /root/.vaultwarden-pass
sudo chmod 600 /root/.vaultwarden-pass
```

---

6. Backup Script

Save to `/usr/local/bin/vaultwarden-backup.sh`:

```
#!/bin/bash
# ========================================
# Vaultwarden Automated Backup & FTP Upload
# ========================================

BACKUP_SRC="/opt/docker/vaultwarden"
BACKUP_DIR="/opt/backups/vaultwarden"
DOCKER_CONTAINER="vaultwarden"
FTP_HOST="100.122.125.15"
FTP_USER="khang"
FTP_PASS="Minhkhang0812@"
FTP_DIR="/backups/vaultwarden"
PASSPHRASE_FILE="/root/.vaultwarden-pass"
DAYS_TO_KEEP=7
```

```
DATE=$(date +%Y-%m-%d_%H-%M-%S)
FILENAME="vaultwarden_backup_${DATE}.tar.gz"
ARCHIVE="${BACKUP_DIR}/${FILENAME}"
ENCRYPTED="${ARCHIVE}.gpg"
LOGFILE="/var/log/vaultwarden-backup.log"

mkdir -p "$(dirname "$LOGFILE")"
exec > >(tee -a "$LOGFILE") 2>&1
echo "======================================"
echo "[+] Vaultwarden Backup Started at $DATE"
echo "======================================"

mkdir -p "$BACKUP_DIR"

echo "[+] Flushing database writes..."
docker exec "$DOCKER_CONTAINER" sqlite3 /data/db.sqlite3 "PRAGMA
wal_checkpoint(TRUNCATE);"

echo "[+] Creating compressed archive..."
tar -czf "$ARCHIVE" -C "$BACKUP_SRC" .

echo "[+] Encrypting backup..."
gpg --batch --yes --passphrase-file "$PASSPHRASE_FILE" -c "$ARCHIVE"
rm "$ARCHIVE"

echo "[+] Uploading to FTP server..."
lftp -u "$FTP_USER","$FTP_PASS" "$FTP_HOST" <<EOF
set ssl:verify-certificate no
mkdir -p "$FTP_DIR"
cd "$FTP_DIR"
put "$ENCRYPTED"
EOF

echo "[+] Removing backups older than $DAYS_TO_KEEP days..."
find "$BACKUP_DIR" -type f -mtime +$DAYS_TO_KEEP -name "*.gpg" -delete

echo "[+] Cleaning up old FTP backups (keep 3 newest)..."
lftp -u "$FTP_USER","$FTP_PASS" "$FTP_HOST" <<EOF
set ssl:verify-certificate no
cd "$FTP_DIR"
cls -1tr *.gpg | head -n -3 | xargs -r rm
bye
EOF

echo "[+] Backup complete: $ENCRYPTED"
echo "======================================"
```

Make it executable:

```
sudo chmod +x /usr/local/bin/vaultwarden-backup.sh
```

---

7. Restore Script

Save to `/usr/local/bin/vaultwarden-restore.sh`:

```
#!/bin/bash
# ========================================
# Vaultwarden Restore Script
# ========================================

BACKUP_DIR="/opt/backups/vaultwarden"
RESTORE_DIR="/opt/docker/vaultwarden"
PASSPHRASE_FILE="/root/.vaultwarden-pass"
BACKUP_FILE="$1"  # Encrypted .gpg backup file path

if [ -z "$BACKUP_FILE" ]; then
  echo "Usage: sudo vaultwarden-restore.sh /path/to/backup.tar.gz.gpg"
  exit 1
fi

TEMP_DIR="/tmp/vaultwarden_restore"
mkdir -p "$TEMP_DIR"

echo "[+] Decrypting backup..."
gpg --batch --yes --passphrase-file "$PASSPHRASE_FILE" -o
"$TEMP_DIR/backup.tar.gz" -d "$BACKUP_FILE"

echo "[+] Extracting backup contents..."
tar -xzf "$TEMP_DIR/backup.tar.gz" -C "$TEMP_DIR"

echo "[+] Stopping Vaultwarden container..."
docker stop vaultwarden

echo "[+] Restoring files to $RESTORE_DIR..."
rm -rf "$RESTORE_DIR"/*
cp -r "$TEMP_DIR"/* "$RESTORE_DIR"/


echo "[+] Restarting Vaultwarden..."
docker start vaultwarden

echo "[+] Restore completed successfully."
rm -rf "$TEMP_DIR"
```

Make it executable:

```
sudo chmod +x /usr/local/bin/vaultwarden-restore.sh
```

To restore:

```
sudo /usr/local/bin/vaultwarden-restore.sh
/opt/backups/vaultwarden/vaultwarden_backup_2025-11-05_12-00-00.tar.gz.gpg
```

---

8. Schedule Automatic Backups

Edit crontab:

```
sudo crontab -e
```

Add:

```
0 2 */5 * * /usr/local/bin/vaultwarden-backup.sh >> /var/log/vaultwarden-
backup.log 2>&1
```

Runs every **5 days at 2:00 AM**.

---

9. Log Rotation

Create `/etc/logrotate.d/vaultwarden-backup`:

```
/var/log/vaultwarden-backup.log {
    weekly
    rotate 4
    compress
    missingok
    notifempty
    create 640 root adm
}
```

☑ Final System Summary

Component	Description
Vaultwarden	Running behind Nginx with Tailscale HTTPS
Admin Panel	Accessible via `/vaultwarden/admin`
SMTP Email	Configured for email notifications
Automated Backup	Runs every 5 days at 2:00 AM
Encryption	GPG with secure stored passphrase
FTP Uploads	Automatically uploaded and oldest files cleaned

Component	Description
Local Retention	Deletes backups older than 7 days
Restore Script	Decrypts and restores Vaultwarden data automatically
Logs	Saved in `/var/log/vaultwarden-backup.log` with rotation

This configuration ensures a fully automated, encrypted, and redundant Vaultwarden deployment — easy to restore and portable across systems.