SELF HOSTED SERVER

Proxmox

- Download
- Bootable USB
- Configure Disk Network Admin Password
- Reboot
- Web Interface

ISOs

- USB Drive ext4 or NTFS
- Identify USB Drive — lsblk
- Mount USB Drive — mkdir /mnt/usb
  mount /dev/***sdb1*** /mnt/usb
- Copy ISO Files — cd /var/lib/vz/template/iso
  cp /mnt/usb/*.iso
  Datacenter > pve > local (pve) > ISO Images
- Upload ISO (GUI)

Windows VM

- Create VM — Create VM name Windows ISO OVMF (UEFI)
- Hard Disk — VirtIO SCSI
- CPU Memory Settings — Cores RAM
- Network — VirtIO (paravirtualized)
- VirtIO Drivers — Second CD/DVD drive VirtIO ISO
- Start — Custom installation load the VirtIO drivers from the second CD/DVD

Helper Scripts LXC

- Visit Website — https://tteck.github.io/Proxmox/
- Proxmox shell — *Bash command*
- Nextcloud, Jellyfin, Post Install, Kernel Cleanup, Host Backup...

Best Practices

- Review Scripts
- Test
- Updates
- Backup Frequently

Bind mount

- Create folder on CT — sudo mkdir -p /mnt/media
- Create folder on host — sudo mkdir -p /mnt/media
- Edit CTs config file (host) — /etc/pve/lxc
  pct set ***container_id*** -mp0 /mnt/media,mp=/mnt/media
- Restart CT — pct restart ***container_id*** on host, or use Proxmox GUI

Samba Shared Folder

- Install Samba — sudo apt-get install samba
- Samba config — sudo nano /etc/samba/smb.conf
- Add a new share — Add these lines at the bottom of the file:

  [*shared_folder*]
  path = */mnt/media*
  available = yes
  valid users = *username*
  read only = no
  browsable = yes
  guest ok = no
- Samba User/password — sudo adduser *new_samba_user* First you need to create the user on host
  sudo smbpasswd -a new_samba_user Then add it to samba database
  If you clone VMs user/password access will be direct. No login with credentials
- Set folder ownership — sudo chown -R *username*:*groupname /mnt/media*
- Set permissions — sudo chmod -R 0755 */mnt/media*
- Restart Samba — sudo systemctl restart smbd
- Access from VM — \\192.168.0.25*shared_folder* Type this on file explorer address bar

Reverse-proxy

Files

- Main Configuration — /etc/nginx/nginx.conf
- Site Configuration — /etc/nginx/sites-available/
- Enabled Site — /etc/nginx/sites-enabled/
- Additional Config — /etc/nginx/conf.d/
- Certificates — /etc/nginx/ssl/
- Log — /var/log/nginx/

Duck DNS

- Create account
- Create Domain — example.duckdns.org
- Token
- Subdomains — example.example.duckdns.org
  No need to create. They´ll work.

Update IP

- Create Directory — mkdir -p ~/duckdns
- Create Script — cd ~/duckdns
  nano duck.sh

```
---------------------------------- duck.sh----------------------------------
#!/bin/bash
DOMAIN="mydomain"
TOKEN="your-duckdns-token"
Echo
url="https://www.duckdns.org/update?domains=$DOMAIN&token=$TOKEN&ip=" | curl -k -o ~/duckdns/duck.log -K -
#^This 3 lines are 1 command^
--------------------------------------------------------------------------------
```

- Make it executable — chmod +x ~/duckdns/duck.sh
- Run regularly
  - Edit Crontab — crontab -e
  - Add Cron Job — */5 * * * * /root/duckdns/duck.sh >/dev/null 2>&1

Reverse proxy LXC NGINX

- Create LXC — Create CT
  Start
- Install NGINX — apt-get update
  apt-get upgrade
  apt-get install nginx
- Configuration File — Nano /etc/nginx/sites-available/reverse-proxy.conf
- Reverse Proxy Config — See *File1* at the bottom
- Enable Configuration — ln -s /etc/nginx/sites-available/reverse-proxy.conf/etc/nginx/sites-enabled/
  #^This 2 lines are 1 command^
  nginx -t
  systemctl restart nginx
- Install Certbot — apt-get install certbot python3-certbot-nginx
- Obtain Certificate — certbot --nginx -d yourdomain.com
- Check domain — https://yourdomain.com

Port forwarding

- Router Rules
  - HTTP (Port 80):
    - External — 80
    - Internal IP — IP address of your reverse proxy
    - Internal — 80
  - HTTPS (Port 443)
    - External — 443
    - Internal IP — IP address of your reverse proxy
    - Internal — 443

Certificates renewal

- Create script — sudo nano /usr/local/bin/renew_certs.sh

```
.....renew_certs.sh.....
#!/bin/bash
# Stop the reverse proxy server (e.g., NGINX)
sudo systemctl stop nginx
# Renew the certificates
sudo certbot renew
# Wait for a specified amount of time (e.g., 60 seconds) to ensure the renewal process completes (This is one line)
sleep 60
# Start the reverse proxy server
sudo systemctl start nginx
.....
```

- Make it executable — sudo chmod +x /usr/local/bin/renew_certs.sh
- Edit cron job — crontab -e
  - Add this line — 0 2 * * * /usr/local/bin/renew_certs.sh
- Test the script — sudo /usr/local/bin/renew_certs.sh
  This will restart reverse-proxy and connection via domain will be lost until nginx restarts. Local access to services will be available.
- Check logs — sudo cat /var/log/letsencrypt/letsencrypt.log
- Verify server status — sudo systemctl status nginx

---------- File1----------

```
# Main domain
# This block redirects HTTP requests to HTTPS for the main domain example.duckdns.org (Proxmox)
server {
listen 80;
server_name example.duckdns.org;
if ($host = example.duckdns.org) {
return 301 https://$host$request_uri;  # Redirect HTTP to HTTPS
}                                                                    # managed by Certbot
return 404;                                                          # managed by Certbot
}
# This block handles HTTPS requests for the main domain example.duckdns.org (Proxmox)
server {
listen 443 ssl;                                                      # managed by Certbot
server_name example.duckdns.org;                                     # Replace with your domain or IP address
                                                                     # SSL configuration
ssl_certificate /etc/letsencrypt/live/example.duckdns.org/fullchain.pem;     # managed by Certbot
ssl_certificate_key /etc/letsencrypt/live/example.duckdns.org/privkey.pem;   # managed by Certbot
include /etc/letsencrypt/options-ssl-nginx.conf;                     # managed by Certbot
ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;                       # managed by Certbot
  location / {
  proxy_pass https://192.168.0.25:8006;                              # Replace with your Proxmox host IP and port
  proxy_ssl_verify off;                                              # Disable SSL verification if Proxmox uses self-signed certificates
                                                                     # Proxy settings
  proxy_set_header Host $host;                                       # Pass the original Host header
  proxy_set_header X-Real-IP $remote_addr;                           # Pass the real client IP
  proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;       # Pass the client IP through proxies
  proxy_set_header X-Forwarded-Proto $scheme;                        # Pass the original protocol (http or https)
  proxy_set_header Upgrade $http_upgrade;                            # Support WebSocket connections
  proxy_set_header Connection "upgrade";                             # Support WebSocket connections
  }
}
# Jellyfin subdomain
# This block handles HTTPS requests for the Jellyfin server
server {
listen 443 ssl;
server_name jellyfin.example.duckdns.org;
                                                                     # SSL configuration
ssl_certificate /etc/letsencrypt/live/jellyfin.example.duckdns.org/fullchain.pem;
ssl_certificate_key /etc/letsencrypt/live/jellyfin.example.duckdns.org/privkey.pem;
ssl_protocols TLSv1.2 TLSv1.3;                                       # Use secure TLS protocols
ssl_ciphers 'ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-
GCM-SHA384';                                                         # Use secure ciphers. These 2 lines are 1 command
ssl_prefer_server_ciphers on;                                        # Prefer server ciphers
ssl_dhparam /etc/ssl/certs/dhparam.pem;                              # Optional: Generate for stronger DH keys
error_page 497 https://$host$request_uri;
  location / {
  proxy_pass http://192.168.0.3:8096/;                               # Jellyfin's IP and port (8096 by default)
                                                                     # Proxy settings
  proxy_set_header Host $host;                                       # Pass the original Host header
  proxy_set_header X-Real-IP $remote_addr;                           # Pass the real client IP
  proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;       # Pass the client IP through proxies
  proxy_set_header X-Forwarded-Proto $scheme;  # Pass the original protocol (http or https)
  proxy_http_version 1.1;                                            # Use HTTP/1.1 for WebSocket support
  proxy_set_header Upgrade $http_upgrade;                            # Support WebSocket connections
  proxy_set_header Connection 'upgrade';                             # Support WebSocket connections
  }
}
```

```
                                                                                # Nextcloud subdomain
                                                    # This block handles HTTPS requests for the Nextcloud server
server {
listen 443 ssl;
server_name nextcloud.example.duckdns.org;
                                                                                # SSL configuration
ssl_certificate /etc/letsencrypt/live/nextcloud.jongfad.duckdns.org/fullchain.pem;      # not placed by Certbot
      Both are possible. Certbot always places certs in correct folder but only add lines to config file if specified.
ssl_certificate_key /etc/letsencrypt/live/nextcloud.example.duckdns.org/privkey.pem;
ssl_protocols TLSv1.2 TLSv1.3;                                  # Use secure TLS protocols
ssl_ciphers 'TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-RSA-AES256-GCM-SHA384';          # Use secure ciphers. These 2 lines are 1 command
ssl_prefer_server_ciphers on;                                   # Prefer server ciphers
  location / {
  proxy_pass https://192.168.0.32:443;          # Replace with the internal IP of your Nextcloud container
                                                                                # Proxy settings
  proxy_set_header Host $host;                                  # Pass the original Host header
  proxy_set_header X-Real-IP $remote_addr;                      # Pass the real client IP
  proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;  # Pass the client IP through proxies
  proxy_set_header X-Forwarded-Proto https;                     # Pass the original protocol (https)
  }
}
```

----------**End Of File**----------