# dnsmasq and tftp-hpa (solo arch no seguir del todo)

The command you provided and the subsequent configuration steps are related to setting up a **network boot (netboot)** environment on an Arch Linux system.

## Understanding the Command

```
sudo pacman -S dnsmasq tftp-hpa
```

- **`sudo`** : Executes the command with administrative (root) privileges.
- **`pacman`** : The package manager for Arch Linux.
- **`-S`** : Syncs and installs the specified packages.
- **`dnsmasq`** : A lightweight DNS, DHCP, and TFTP server.
- **`tftp-hpa`** : A TFTP (Trivial File Transfer Protocol) server implementation.

**Purpose**: This command installs both `dnsmasq` and `tftp-hpa` on your Arch Linux system. `dnsmasq` will handle DHCP (assigning IP addresses) and DNS services, while `tftp-hpa` will manage the transfer of boot files necessary for netbooting.

## Configuring Arch Linux for Netbooting

Setting up a netboot server involves several steps: installing necessary packages, configuring services, preparing boot files, and ensuring proper network settings. Below is a comprehensive guide to configuring your Arch Linux system for netbooting.

### 1. Install Required Packages

You've already started this with:

```
sudo pacman -S dnsmasq tftp-hpa
```

Additionally, you might need other packages depending on your specific netboot requirements, such as `pxelinux` for PXE booting or `syslinux` for boot loaders.

```
sudo pacman -S syslinux
```

## 2. Configure `dnsmasq`

`dnsmasq` will serve both DHCP and TFTP requests. Here's how to configure it:

1. **Backup the Default Configuration**

   ```
   sudo cp /etc/dnsmasq.conf /etc/dnsmasq.conf.backup
   ```

2. **Edit the Configuration File**

   Open the `dnsmasq` configuration file in your preferred text editor:

   ```
   sudo nano /etc/dnsmasq.conf
   ```

3. **Configure DHCP Settings**

   Add or modify the following settings to define your DHCP range and boot options:

```
# Enable DHCP
dhcp-range=192.168.1.100,192.168.1.200,12h

# Specify the boot file and server
dhcp-boot=pxelinux.0,arch-netboot,192.168.1.1

# Enable TFTP
enable-tftp
tftp-root=/srv/tftp
```

**Explanation**:

- **`dhcp-range`** : Defines the range of IP addresses to assign to clients (e.g., from `192.168.1.100` to `192.168.1.200` ) with a lease time of 12 hours.
- **`dhcp-boot`** : Specifies the boot file ( `pxelinux.0` ), the hostname ( `arch-netboot` ), and the TFTP server's IP address ( `192.168.1.1` ).
- **`enable-tftp`** : Activates the TFTP server.
- **`tftp-root`** : Sets the directory from which TFTP will serve files.

4. **Create the TFTP Root Directory**

```
sudo mkdir -p /srv/tftp
sudo chmod -R 755 /srv/tftp
```

## 3. Configure `tftp-hpa`

1. **Edit the TFTP Configuration**

   The configuration file for `tftp-hpa` is typically located at `/etc/conf.d/tftpd` . Create or edit this file:

```
sudo nano /etc/conf.d/tftpd
```

2. **Add the Following Configuration**

```
# /etc/conf.d/tftpd
TFTP_USERNAME="tftp"
TFTP_DIRECTORY="/srv/tftp"
TFTP_ADDRESS="0.0.0.0:69"
TFTP_OPTIONS="--secure --create"
```

**Explanation**:

- `TFTP_USERNAME` : Runs the TFTP server under the `tftp` user for security.
- `TFTP_DIRECTORY` : Specifies the root directory for TFTP.
- `TFTP_ADDRESS` : Binds the TFTP server to all network interfaces on port 69.
- `TFTP_OPTIONS` : Enables secure mode and allows the creation of new files.

3. **Enable and Start the TFTP Service**

```
sudo systemctl enable tftpd
sudo systemctl start tftpd
```

### 4. Prepare Boot Files

1. **Install Syslinux**

   Syslinux provides the `pxelinux.0` bootloader necessary for PXE booting.

```
sudo pacman -S syslinux
```

2. **Copy Bootloader Files to TFTP Directory**

```
sudo cp /usr/lib/syslinux/pxelinux.0 /srv/tftp/
sudo mkdir /srv/tftp/pxelinux.cfg
```

3. **Create a PXE Configuration File**

   Create a default configuration file to specify boot parameters.

```
sudo nano /srv/tftp/pxelinux.cfg/default
```

**Add the Following Content**:

```
DEFAULT menu.c32
PROMPT 0
TIMEOUT 100
MENU TITLE Arch Linux Netboot

LABEL arch
    MENU LABEL Install Arch Linux
    KERNEL vmlinuz-linux
    APPEND initrd=initramfs-linux.img archisobasedir=arch archisolabel=ARCH_202309
```

**Explanation**:

- `DEFAULT menu.c32` : Sets the default menu.
- `PROMPT` : Disables the prompt.
- `TIMEOUT` : Sets a timeout before default selection.
- `MENU TITLE` : Titles the boot menu.
- `LABEL` : Defines a boot option.
- `KERNEL` : Specifies the kernel to boot.
- `APPEND` : Passes parameters to the kernel.

4. **Copy Kernel and Initramfs**

   Ensure that the kernel ( `vmlinuz-linux` ) and initramfs ( `initramfs-linux.img` ) are available in the TFTP directory.

```
sudo cp /boot/vmlinuz-linux /srv/tftp/
sudo cp /boot/initramfs-linux.img /srv/tftp/
```

*If you're using a different kernel or initramfs, adjust the filenames accordingly.*

## 5. Configure and Start `dnsmasq`

1. **Enable and Start `dnsmasq` Service**

   ```
   sudo systemctl enable dnsmasq
   sudo systemctl start dnsmasq
   ```

2. **Verify `dnsmasq` Status**

   Ensure that `dnsmasq` is running without errors.

   ```
   sudo systemctl status dnsmasq
   ```

## 6. Network Configuration

Ensure that your network interface is correctly configured to allow DHCP and TFTP traffic. This typically involves:

- **Static IP Assignment**: Assign a static IP to the server if it's acting as the DHCP server.

  Edit your network configuration (e.g., using `systemd-networkd`, `netctl`, or another network manager) to set a static IP like `192.168.1.1`.

- **Firewall Settings**: Allow traffic on ports **67** (DHCP), **68** (DHCP), **69** (TFTP), and any other necessary ports.

  Example using `ufw`:

  ```
  sudo ufw allow 67/udp
  sudo ufw allow 68/udp
  sudo ufw allow 69/udp
  sudo ufw reload
  ```

## 7. Testing the Setup

1. **Boot a Client Machine via Network**

   Configure a client computer's BIOS/UEFI to boot from the network (PXE). Upon booting, it should receive an IP address from `dnsmasq`, retrieve the bootloader via TFTP, and proceed with the installation or desired operation.

2. **Troubleshooting**

   - **Check Logs**: Review `dnsmasq` and `tftpd` logs for any errors.

     ```
     sudo journalctl -u dnsmasq
     sudo journalctl -u tftpd
     ```

   - **Verify TFTP Access**: Use a TFTP client to ensure that files are accessible.

     ```
     tftp 192.168.1.1
     tftp> get pxelinux.0
     ```

   - **Ensure Correct File Paths**: Double-check that all necessary boot files are in the `/srv/tftp` directory.

## Additional Considerations

- **Secure Your Services**: Ensure that only authorized devices can access your netboot server to prevent unauthorized usage.
- **Automate with Scripts**: For large-scale deployments, consider automating the setup using scripts or configuration management tools like Ansible.
- **Update Boot Files**: Keep your kernel and initramfs updated to ensure security and compatibility.
- **Monitor Network Traffic**: Use tools like `tcpdump` or `wireshark` to monitor DHCP and TFTP traffic if you encounter issues.

## Resources

- **Arch Wiki**:

  - PXE
  - Dnsmasq
  - TFTP
- **Syslinux Documentation**: Syslinux Wiki

By following these steps, you should be able to set up your Arch Linux system as a netboot server, allowing client machines to boot and install or run systems over the network.