# IPv6 Testbed Setup at NITK Surathkal, India

Supported by:

## Abstract

This document details the comprehensive setup and configuration of an IPv6-only testbed at the Department of Computer Science and Engineering (CSE), National Institute of Technology Karnataka (NITK), Surathkal, India. The testbed involves the deployment of an OPNsense testbed router, configuring a Generic tunneling InterFace (GIF), switch setup, DNS server configuration using BIND9, IPv6 address assignment with KEA DHCPv6, and end client address configurations. This document also highlights issues encountered during the setup and their corresponding solutions with an aim to serve as a guide for implementing a functional IPv6-only testbed. The documentation prefix 2001:db8::/32 has been used throughout this document as per RFC 3849.

## 1. Testbed Setup

The IPv6 testbed has been set up in the Department of CSE at the NITK Surathkal, India using five desktop computers and one L2 Managed switch. We refer to the desktop computers as Machine 1, 2, 3, 4 and 5. The topology used for setting up the testbed is shown in Figure 1 on the next page.

**ISP Gateway**: This gateway is provided by an Internet Service Provider (in this case, BSNL), and directs network traffic between the internal NITK network and the external Internet.

We have leased a /32 IPv6 address block, and route /36 for NITK's IPv6 Intranet. In accordance with the IPv6 address plan, the subnet 2001:db8:b0::/50 is statically routed to the IPv6 testbed.

**OPNsense router**: A security-centric device that manages firewall rules and routing for the NITK network.

**Core Switch**: A high-capacity switch used to route data between segments, and facilitates connectivity between the distribution switches and the router.

**CSE Switch**: A switch located in the Dept. of CSE that connects the devices in Dept. of CSE subnet.

Machine 1 acts as an OPNsense testbed router, with OPNsense running on bare metal. This machine has two physical interfaces. One interface is used to connect to the L2 Managed switch, while the other is used to connect to the subnet of the Department of CSE.

A GIF Tunnel has also been configured between the OPNsense testbed router and the college's OPNsense router for IPv6 connectivity over IPv4. This is being used as a temporary solution to get the testbed working before routing a VLAN directly.

Ubuntu 22.04.3 has been installed on Machines 2 - 5. Machines 2 and 3 are regular end client machines on separate VLANs. Machine 2 is on VLAN 710, while Machine 3 is on VLAN 720.

**Note:** All commands used in this documentation are specific to the Ubuntu family. While some may be common to Debian, we recommend that the reader verifies the equivalent command for their operating system before usage.

Machine 4 is set up on VLAN 730, and runs a KEA DHCPv6 server. This machine is responsible for IPv6 address allocation to other machines on the testbed.

Machine 5 is also set up on VLAN 730, and runs a BIND9 DNS server. All query resolution requests that come in from Machines 2-4 are forwarded to the DNS server on Machine 5, which either resolves the query or forwards it to an authoritative nameserver through the OPNsense router.

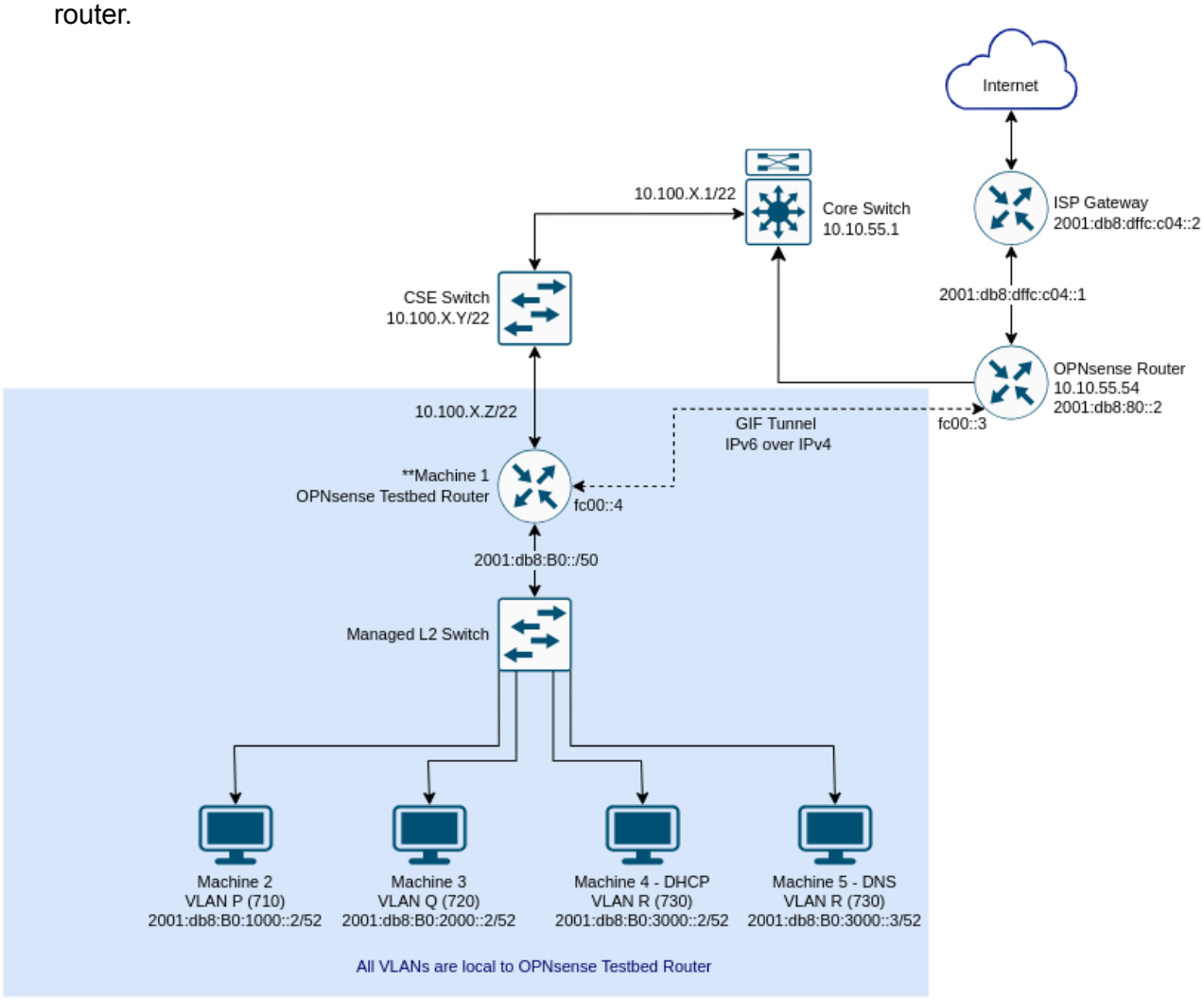



Figure 1: Topology used to set up an IPv6 testbed

**Note:** These IP addresses are specific to the testbed at NITK, and are to be changed as per the configurations available to the reader.

## 2. Switch Configuration

A TP-Link L2 managed switch was used while setting up the testbed.

The CLI Reference Guide[1] and a list of configuration examples[2] were followed to configure the switch.

Port 1: Tagged all VLANs (with native VLAN=1)
Port 2: Untagged VLAN with VLAN ID 710 (P)
Port 3: Untagged VLAN with VLAN ID 720 (Q)
Port 4, 5: Untagged VLAN with VLAN ID 730 (R)

**Note:** These VLAN IDs (710, 720, 730) are specific to the testbed at NITK, and should be changed as per the VLAN IDs used in the readers network.

### 2.1 Command Line Configuration

Access the switch[3] via the console port and login with the default username and password.

```
> enable

View all available ports
config# show interface switchport

View specific port
config# show interface switchport gigabit Ethernet 1/0/1

Configure untagged VLAN with VLAN ID 710 on port 2
config# interface gigabitEthernet 1/0/2
config-if# switchport general allowed vlan 710 untagged
```

Use the above pattern to configure all the required ports.

## 3. OPNsense Testbed Router: Installation and Configuration

OPNsense is an open-source, FreeBSD-based operating system that functions as a Layer 4 router and Stateful Packet Inspection firewall. In our testbed, OPNsense played a vital role with its key features, including a Router Advertisement (RA) daemon for automatic IPv6 address configuration, GIF tunnel capabilities for IPv6 over IPv4, and integrated DHCPv6 and Unbound DNS services that facilitated initial prototyping. While OPNsense addressed early testing requirements, scalability considerations prompted us to deploy separate DHCP and DNS servers. Although OPNsense incorporates the widely used ISC DHCP server and now supports

[1] https://static.tp-link.com/res/down/doc/TL-SG3216(UN)_V2.0_CLI.pdf
[2] https://www.tp-link.com/baltic/support/faq/3534/
[3] https://www.tp-link.com/us/support/faq/291/

KEA DHCPv4, it currently lacks support for KEA DHCPv6. Consequently, the necessity for finer configuration control, such as managing databases and threads, as well as implementing high-availability setups for individual services like DHCP, proved challenging within OPNsense. The platform's user-friendly interface and the above-mentioned features ensured a swift validation of our network topology. Readers are encouraged to evaluate their specific needs and explore alternative tools, such as VyOS[4] and OpenWRT[5], based on their unique feature sets and scalability objectives.

This section describes the steps to configure an OPNsense testbed router on Machine 1. The configuration procedure involves two main steps: firstly, following the initial installation guidelines provided in the OPNsense Documentation[6], and secondly, configuring the interfaces for specific network settings as specified below.

### 3.1 Steps followed for the installation of OPNsense on Machine 1

3.1.1. Download:

- Visit the OPNsense Download Page[7].
- Choose the 64-bit variant (amd64 architecture) appropriate for your system.
- Download the VGA mode package for installation.

3.1.2. Prepare an installation media:

- Use an empty USB stick (>=1 GB) for installation.
- For Windows, use an application like 7zip to extract the downloaded file.
- Remove ".bz2" from the filename after extraction.

3.1.3. Write Image to USB:

- Use tools like dd (for Unix-like OSes), physdiskwrite[8], Etcher[9], or Rufus[10] (for Windows) to write the OPNsense image to the USB flash drive.

3.1.4. System Boot Preparation:

- Ensure access to the console (via keyboard and [virtual] monitor or serial connectivity).
- Identify the key (F#, Del, ESC) to access the boot menu or system BIOS (UEFI) immediately after powering on.

3.1.5. Installation Instructions:

- Boot the system with the OPNsense installation media.
- Press any key when prompted with "Press any key to start the configuration importer."
- If the OPNsense logo appears, restart the system to reach the Importer.
- Type the device name of the existing drive with the configuration and press enter.
- If Importer is successful, boot into the Live environment with the stored configuration.
- If the Importer fails, return to the device selection prompt. Confirm the device name and retry. Check for disk corruption or restore from backup, if needed.

3.1.6. Post-Installation:

---

[4] https://vyos.io/
[5] https://openwrt.org/
[6] https://docs.opnsense.org/manual/install.html#initial-installation-configuration
[7] https://opnsense.org/download/
[8] https://m0n0.ch/wall/physdiskwrite.php
[9] https://etcher.balena.io/#download-etcher
[10] https://rufus.ie/en/

- After booting with OPNsense Full Image, access the Live environment through Local Console, GUI (HTTPS), or SSH.
- Log into the shell using the default credentials: Username: root, Password: opnsense.
- The GUI is accessible at https://192.168.1.1/ with the default credentials (root/opnsense) unless a previous configuration was imported.

### 3.2 Steps followed for the Configuration of VLAN Interface

3.2.1. Interface Assignment:

- Login to the local console and select option '1' to assign interfaces.
- Configure VLANs for VLAN-capable interface em0.
- Avoid VLAN tag '1' to avoid exposing the router to the Dept. of CSE subnet.
- Assign VLAN tag '700' to create em0_vlan700.

3.2.2 Set Interface IP Addresses

- On the local console, select option '2' to set IP addresses for the interface.
- Select the number corresponding to the interface that you wish to configure.
- Assign the IP address via DHCP or enter a static IP address as required.

3.2.3 Network Configuration:

- Enter LAN IPv6 subnet bit count: 52.
- Configure LAN interface details:
    - Assign IPv6 address 2001:db8:b0:1000::2/52 to em0_vlan700.

3.2.4 Verification and Access:

- The web GUI for OPNsense can be accessed through the following URLs:
    - Default IP address: https://192.168.1.1/.
    - Statically assigned IPv4 address: http://10.100.15.71.

**Note:** The interface names and IP addresses are specific to the testbed at NITK, and are to be changed as per the configurations available to the reader.

## 4. GIF Tunnel Configuration

### 4.1 Creation of GIF Tunnel

- Start with adding a new gif interface. Access the web GUI of the OPNsense router. Navigate to Interfaces -> Other Types -> GIF and click on Add.
- Configure the tunnel as per your requirements. In this case, we have used the following configuration:
    - Parent interface: LAN
    - GIF remote address: 10.10.55.54
    - GIF tunnel local address: fc00::4/64
    - GIF tunnel remote address: fc00::3/64
- This GIF tunnel must now be assigned as a new interface. Navigate to Interfaces -> Assignments. Select the GIF tunnel as the Device, and add it.
- Under Interface -> [OPTX] (the new interface), select Enable Interface and save it.

### 4.2 Rules and Routes for the GIF Tunnel

- Create a GIF tunnel at the NITK OPNsense router (NITKO) [fc00::3] and the OPNsense Testbed router (CSO) [fc00::4].
- Add a firewall rule on NITKO to allow IPv6 traffic from CSO to the BSNL gateway. This is for outgoing traffic to the Internet from CSO.
- Add a firewall rule to allow all IPv4 traffic to and from NITKO to and from CSO for all protocols.
- Add 2001:db8:b0::/50 (CSO prefix) with a static route to [fc00::4] on NITKO.
- Add fc00::3 as the default gateway on CSO.
- Add rule on CSO to allow incoming traffic through 2001:db8:b0::/50.

**Note:**

- The GIF Tunnel is being used as a temporary solution to get the testbed working before routing a VLAN directly. Users may implement it if required, but it is not a mandatory step.
- These IPv6 addresses are specific to the testbed at NITK and are to be changed as per the configurations available to the reader.

## 5. DNS Server Configuration

This section outlines the steps taken to configure a DNS server on the testbed. The configuration guide[11] was used to set up a BIND9 recursive server with only an IPv6 interface. Note that the configuration detailed in this section does not create an authoritative nameserver to serve AAAA records.

### 5.1 General Information

- Installation:

```
$ sudo apt update
$ sudo apt install bind9
```

- Check BIND Version:

```
$ named -v
```

Output:
*BIND 9.18.12-1ubuntu1.2-Ubuntu (Extended Support Version) <id:>*

Note that the exact output may vary depending on the version installed.

### 5.2 Initial Setup

- Check that the BIND9 server is up and running:

```
$ sudo systemctl status named
```

[11] https://www.lacnic.net/innovaportal/file/6687/1/lacnic-dns-ipv6only_en.pdf

- Check network interface configuration and IPv6 addresses:

  ```
  $ ip a
  ```

  The interface and IP address information obtained as output will be used to configure the DNS server.

### 5.3 Server Configuration

The DNS server configured on the testbed is a recursive resolver. When it does not find an answer in its local database, it queries other DNS servers to resolve the query.

The main configuration file is '**/etc/bind/named.conf**', which contains the following lines:

```
// This is the primary configuration file for the BIND DNS server named.
// Please read /usr/share/doc/bind9/README.Debian for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
// If you are just adding zones, please do that in /etc/bind/named.conf.local

// include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";

acl my_ipv6_net {
    2001:db8:b0:1000::/52;
};

zone "example.com" {
    type master;
    file "/var/lib/bind/example.com";
    allow-query { any; };
    allow-transfer { any; };
    dnssec-policy default;
    inline-signing yes;
};


options {
    directory "/var/cache/bind";
    listen-on-v6 {2001:db8:b0:3000::3;};
    query-source-v6 address 2001:db8:b0:3000::4;
    recursion yes;
    forwarders { 2001:4860:4860::8888; };
    forward only;
    allow-recursion { my_ipv6_net;};
    allow-query { any; };
};
```

**Note:** The IPv6 addresses in the */etc/bind/named.conf* file are specific to the testbed at NITK, and are to be changed as per the configurations available to the reader.

- Apply changes:

```
# systemctl reload named
# systemctl restart named
```

- Validate that BIND9 is listening on IPv6:

```
# netstat -puan | grep named
```

### 5.4 Configuration Walkthrough

This section provides a brief explanation of the IPv6 configuration parameters used in the *'options{ …};'* section in the */etc/bind/named.conf* file outlined in Section 5.3.

5.4.1 Access Control Lists

- We create an Access Control List (ACL) called *my_ipv6_net* to match clients to the specified IPv6 addresses.
- We have added the IPv6 prefix corresponding to only one VLAN for the purpose of testing.

5.4.2 Options

- listen-on-v6
  - `listen-on-v6 {2001:db8:b0:3000::3;};` tells the BIND9 server which IPv6 addresses it should listen on for IPv6 DNS requests.
  - We may also specify that the server should listen on multiple IPv6 addresses: `listen-on-v6 {::1; 2001:db8:b0:3000::3;}`
- query-source-v6 address
  - `query-source-v6 address 2001:db8:b0:3000::4;` specifies the IPv6 address on which the DNS server establishes any outgoing connections.
- forwarders
  - `forwarders { 2001:4860:4860::8888; };` is optional and is used when we wish to forward all DNS queries to another DNS server.

**Note:** In the above configuration, the DNS server listens on one IPv6 address and establishes outgoing connections on another IPv6 address.

### 5.5 Modifications on OPNsense Router

To forward DNS queries from VLAN P and Q to the DNS server on VLAN R (see the VLANs in Figure 1), a port forwarding rule is set up in the *Firewall -> NAT -> Port Forward* section.

The redirect target IP is set to the IPv6 address on which the DNS server listens, and the redirect port is set to 53.

# 6. DHCP Installation and Configuration

This section provides the steps necessary to deploy IPv6 with KEA DHCP on a testbed. KEA DHCP, an open-source DHCP server developed by the Internet Systems Consortium (ISC), supports both IPv4 and IPv6 addressing. It supports both DHCPv4 and DHCPv6 protocols along with their extensions, e.g. prefix delegation.

## 6.1. KEA DHCP Installation

6.1.1 Version Information

- KEA Version: 1.6.1
- Documentation: https://kea.readthedocs.io/en/kea-1.6.1/
- The Ubuntu Server documentation[12] and APNIC Academy Lab[13] were used as guidance for the installation and configuration of the DHCPv6 server.

6.1.2 Installation Steps

- Elevate Permissions:

```
$ sudo su -
```

- Install Package:

```
# apt install kea
```

## 6.2. Configuration of KEA DHCP for IPv6

6.2.1 Steps for configuring KEA DHCPv6

1: Navigate to Kea Configuration Directory

```
# cd /etc/kea/
```

2: Create a Configuration Backup

Before making changes, create a backup of the existing kea-dhcp6.conf file.

```
# cp kea-dhcp6.conf kea-dhcp6.conf.bak
```

3: Edit the KEA DHCPv6 Configuration

Open the kea-dhcp6.conf file for editing using your preferred text editor (e.g., nano).

```
# nano kea-dhcp6.conf
```

[12] https://ubuntu.com/server/docs/how-to-install-and-configure-isc-kea
[13] https://academy.apnic.net/en/virtual-labs/deploying-dhcpv6-with-kea-61398

4: Configure KEA DHCPv6

Add the following configuration to the file:

```
{
  "Dhcp6": {
    "valid-lifetime": 4000,
    "renew-timer": 1000,
    "rebind-timer": 2000,
    "preferred-lifetime": 3000,
    "interfaces-config": {
      "interfaces": [
        "eth0/2001:db8:B0:2000::3"
      ]
    },
    "lease-database": {
      "type": "memfile",
      "persist": true,
      "name": "/var/lib/kea/dhcp6.leases"
    },
    "subnet6": [
      {
        "subnet": "2001:db8:B0:1000::/52",
        "pools": [
          {
            "pool": "2001:db8:B0:1000::2/64"
          }
        ]
      }
    ],
    "loggers": [
      {
        "name": "kea-dhcp6",
        "output_options": [
          {
            "output": "/var/log/kea-dhcp6.log",
            "maxsize": 1048576,
            "maxver": 10
          }
        ],
        "severity": "INFO",
        "debuglevel": 0
      }
    ]
  }
}
```

6: Start KEA DHCPv6

Run the following command to start KEA DHCPv6 using the configured file:

```
# kea-dhcp6 -c /etc/kea/kea-dhcp6.conf
```

7: Enable KEA DHCPv6 Service

If you want KEA DHCPv6 to start automatically on system boot, enable the service:

```
$ sudo systemctl enable kea-dhcp6
```

8: On Client Side:

Edit the network interface configuration file on the client to use DHCPv6:

```
$ sudo nano /etc/network/interfaces
$ iface eth0 inet6 dhcp
```

This completes the configuration of KEA DHCPv6 on your system. Ensure that the client's network interface configuration is set to obtain IPv6 addresses via DHCPv6.

6.2.2 Configuration Guide

This section provides a brief explanation of the KEA DHCPv6 server configuration file. The configuration file defines parameters for DHCPv6 server operation, including lease lifetimes, interface settings, lease database specifications, subnet configurations, and logging details.

A. Global Parameters

- valid-lifetime: Specifies the default valid lifetime for addresses.
- renew-timer: Specifies the T1 timer for address renewal.
- rebind-timer: Specifies the T2 timer for address rebind.
- preferred-lifetime: Specifies the default preferred lifetime for addresses.

B. Interfaces Configuration

- interfaces-config: Specifies the network interfaces on which the server should listen to DHCP messages.
- interfaces: Specifies a list of network interfaces on which the server should listen - DHCP server listens on "eth0" for messages with the specified IPv6 address "2001:db8:B0:2000::3".

C. Lease Database

- lease-database: Configures the storage for lease information.
    - type: "memfile" - Specifies the use of an in-memory database with on-disk storage.
    - persist: true - Indicates that leases are stored persistently.
    - name: "/var/lib/kea/dhcp6.leases" - Specifies the path for the lease database.

6.2.3. Subnet Configuration

A. IPv6 Subnet

- subnet6: Defines the IPv6 subnets the server should handle.
    - subnet: "2001:db8:B0:1000::/52" - Specifies the subnet range.
    - pools: Defines address pools within the subnet.
        - pool: "2001:db8:B0:1000::2/64" - Specifies the address pool range.

B. Logging Configuration

- loggers: Configures logging settings for the DHCPv6 server.
    - name: "kea-dhcp6" - Logger name.
    - output_options: Specifies output options for logs.
        - output: "/var/log/kea-dhcp6.log" - Log file path.
        - maxsize: 1048576 - Maximum log file size in bytes.
        - maxver: 10 - Maximum number of log files.
    - severity: "INFO" - Log severity level.
    - debuglevel: 0 - Debugging level.

**Note:** Please customize configuration files and commands based on your specific network requirements. Refer to the official KEA DHCP documentation[14] for detailed information.

## 6.3. KEA Control and KEA Control Agent Configuration

The 'keactrl' utility is a shell script designed to facilitate the control, initiation, and reconfiguration processes for various KEA servers, including *kea-dhcp6* and *kea-ctrl-agent*. It also offers functionalities for checking the current operational status of these servers and identifying the configuration files in use.

Notably, *keactrl* is exclusively available when KEA is constructed from source code. In instances where KEA is installed using native packages, *systemd* scripts are typically provided for management purposes.

6.3.1. Configuration Steps for keactrl.conf and kea-ctrl-agent.conf

keactrl.conf Configuration:

1. Access the keactrl.conf file for editing:

```
sudo nano /etc/kea/keactrl.conf
```

2. Apply the following configurations:

```
dhcp4=no # Disable DHCPv4 server
dhcp6=yes # Enable DHCPv6 server
```

[14] https://kea.readthedocs.io/en/kea-1.6.1/

```
dhcp_ddns=no # Disable DHCP DDNS server
ctrl_agent=yes # Enable Control Agent
```

kea-ctrl-agent.conf Configuration:

1. Open the kea-ctrl-agent.conf file for editing:

```
nano /etc/kea/kea-ctrl-agent.conf
```

2. Find the part with HTTP settings and update accordingly:

```
"Control-agent": {
"http-host": "192.168.2.100", # Set the HTTP host IP
"http-port": 8080 # Set the HTTP port
```

6.3.2. Initiate KEA Servers and Control Agent

1. Explicitly start DHCPv6 server:

```
keactrl start -c /etc/kea/keactrl.conf -s dhcp6
```

2. Commence all servers, including DHCPv6, using default parameters:

```
keactrl start
```

3. Verify the status of KEA servers:

```
keactrl status
```

# 7. End Client Address Configuration

## 7.1. Initial Configuration with Static Addresses

The following steps were followed to configure the addresses on end client devices:

1. Open a terminal on the end client machine.

2. Execute the following command to configure a static IPv6 address:

```
sudo nmcli connection modify "<interface-name>" ipv6.method
manual ipv6.address "2001:db8:b0:1000::2/64" ipv6.gateway
"2001:db8:b0:1000::1"
```

3. To apply the changes, restart the network manager:

```
sudo systemctl restart NetworkManager
```

4. Verify the configuration:

```
nmcli connection show "<interface-name>"
```

**Note:** The provided commands are specific to Ubuntu and rely on installing the Network Manager CLI tool. It is important to acknowledge that these commands may not be universally available across all Linux distributions.

### 7.2. SLAAC Configuration with OPNsense Router Advertisements

Configuration Steps:

1. Activate Router Advertisements on the OPNsense router, providing end users with the flexibility to configure the Unmanaged mode, which is defined using the A (AdvAutonomous) and L (AdvOnLink) flags. The A (AdvAutonomous) flag is set to enable SLAAC, allowing end clients to autoconfigure IPv6 addresses. The L (AdvOnLink) flag is set to ensure that the advertised prefixes are considered on-link for local communication.
2. End client machines configured with SLAAC will automatically obtain IPv6 addresses from the advertised prefixes.
3. Verify the assigned addresses on the end client using:

```
ip -6 address show
```

### 7.3. DHCPv6 Configuration with KEA DHCPv6 Server

The following steps were followed to configure addresses on end client devices using DHCPv6:

1. Open a terminal on the end client machine.
2. Execute the following command to configure the network interface for DHCPv6:

```
sudo dhclient -6
```

3. The DHCPv6 client will request and receive an IPv6 address from the KEA DHCPv6 server.
4. Verify the assigned address on the end client using:

```
ip -6 address show
```

## 8. Debugging Guide

This section is a comprehensive log of issues faced and solutions found during the setup of the testbed.

### 8.1 Static IP Address Assignment

**Problem 1:**

- Initially, the OPNsense router interface on VLAN P was assigned a static address of 2001:db8:b0:0000::1/52
- Two results were observed on a (seemingly) random basis:

- The IP address was assigned, but removed from the interface within about 15 minutes.
- Error message that the IP was already assigned to another interface.

**Solution:**

- The entire 2001:db8:b0::/50 is statically routed to the WAN interface on the testbed's OPNsense router, and the loopback address is 2001:db8:b0::1/128.
- To avoid conflict, the IP addresses in 2001:db8:b0:1000::/52 were used for machines in VLAN P.

### 8.2 OPNsense Services

**Problem 2:**

- Sometimes, changes to the Router Advertisement (RA) and DHCPv6 Relay services were not reflected on using the 'restart service' option.
- It was also observed that the services would sometimes stop after any changes were made.

**Solution:**

- After each configuration change, check the services being used on OPNsense (in this case, Router Advertisements and DHCPv6 relay).
- Reload the service and ensure that all required services are running.

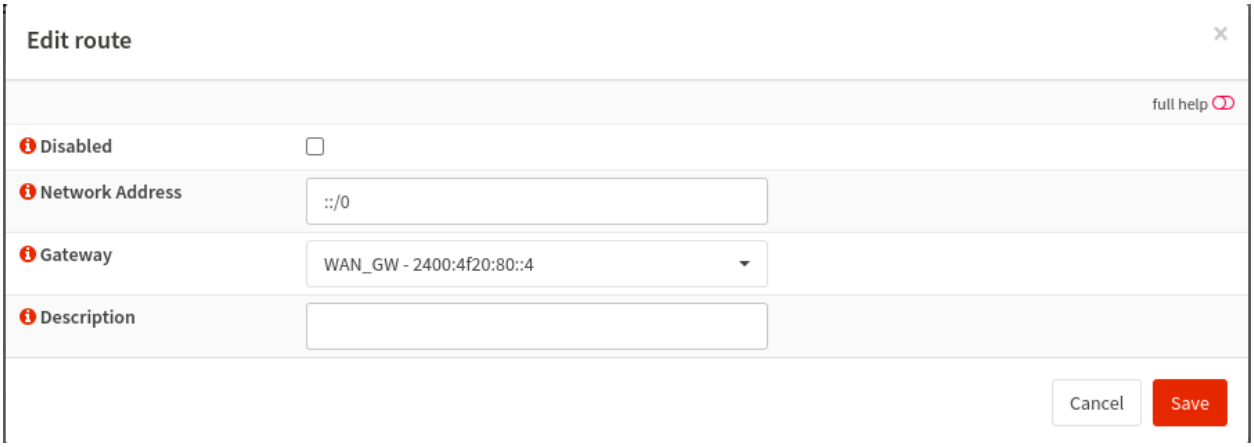
### 8.3 Internet Access

**Problem 3:**

- Pinging an external IP address (say 2001:4860:4860::8888) from the OPNsense router resulted in failure.

**Solution:**

- There was no route that allowed traffic to go through the WAN gateway. Adding a route in System -> Routes -> Configuration resolved this issue.



**Problem 4:**

- Pinging from Machine 2 on VLAN P (2001:db8:b0:1000::2) to an external IPv6 address resulted in failure.

**Solution:**

- Floating Rules contained a rule for incoming IPv6 traffic that specified the gateway as the GIF tunnel connected to CSO (the upstream gateway).
- Due to this firewall rule, the responses to the outgoing packets were being routed through the upstream gateway. Thus, the response packets were sent back to the Internet and unable to reach the end clients.
- Disabling this rule resolved the issue and ensured that the testbed had access to the Internet.

**Note:** The IPv6 addresses in the debugging guide are specific to the testbed at NITK, and are to be interpreted as per the configurations available to the reader.

## Conclusions

This documentation outlined the setup of the L2 switch, OPNsense router, and services such as DNS and DHCPv6 that have been set up on the IPv6 testbed. The future work on the testbed involves capturing packet traces to validate the functioning of the testbed. Subsequently, the tasks involve configuration and testing of services such as Wireguard VPN, OpenVPN and Docker that are frequently used at NITK. This testbed serves as a playground to test different aspects of IPv6 deployment before migrating the NITK campus network to IPv6.

## References


### Switch Configuration

- CLI Reference Guide: https://static.tp-link.com/res/down/doc/TL-SG3216(UN)_V2.0_CLI.pdf
- Configuration Examples: https://www.tp-link.com/baltic/support/faq/3534/
- Accessing the switch: https://www.tp-link.com/us/support/faq/291/

### Router Configuration

- OPNsense Installation Documentation: https://docs.opnsense.org/manual/install.html#initial-installation-configuration
- OPNsense Download Page: https://opnsense.org/download/

### DNS Server Configuration

- https://www.lacnic.net/innovaportal/file/6687/1/lacnic-dns-ipv6only_en.pdf

**DHCP Server Configuration**

- Download Link: https://ftp.isc.org/isc/kea/1.6.1/
- KEA DHCP Documentation: https://kea.readthedocs.io/en/kea-1.6.1/
- Ubuntu Server Documentation to configure ISC Kea: https://ubuntu.com/server/docs/how-to-install-and-configure-isc-kea
- APNIC Academy Virtual Lab (Deploying DHCPv6 with KEA): https://academy.apnic.net/en/virtual-labs/deploying-dhcpv6-with-kea-61398