H2020 iP Over IcN- the betTer IP (POINT)

Examples

Deploy an ICN-only POINT Network and Run ICN Applications

Author: Mays AL-Naday

1. Overview

The objective of this example is to demonstrate how a basic Information-centric network can be setup using the core ICN of our POINT platform (aka Blackadder). For further details about our platform, refer to POINT D3.1.

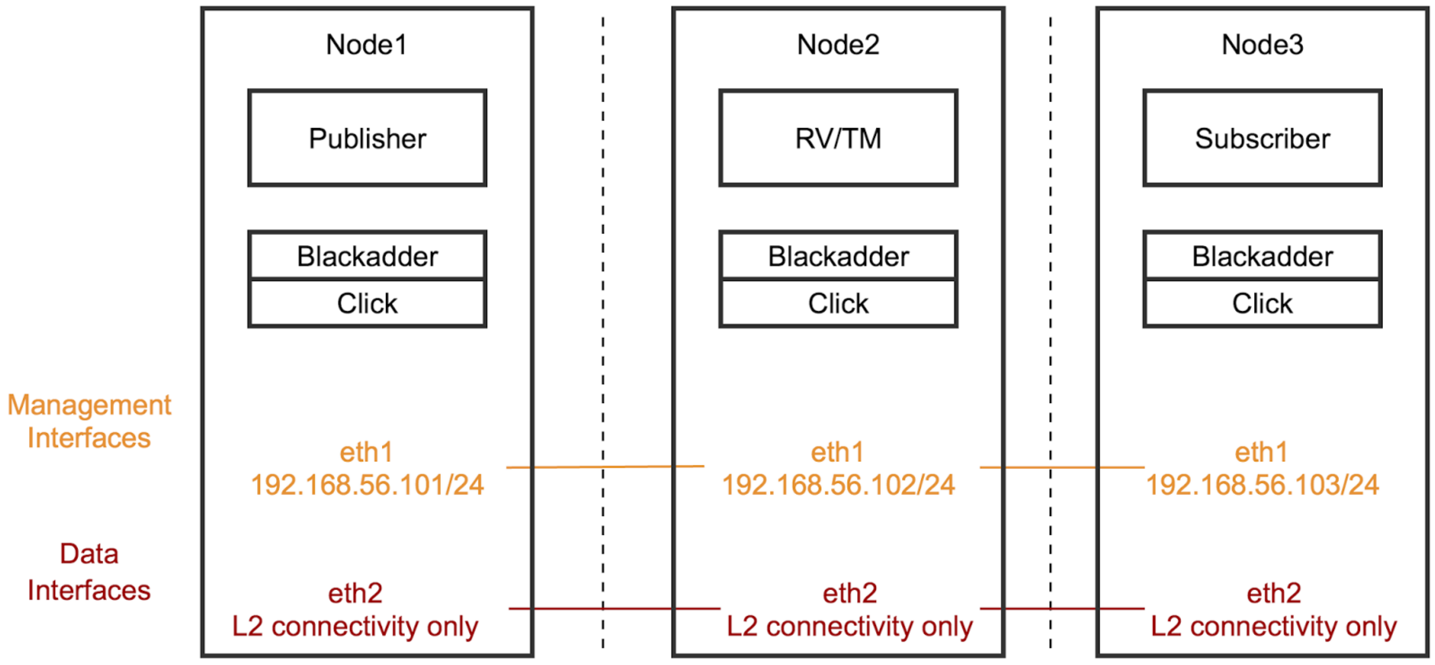

Figure 1: Example Basic ICN Setup

The setup consists of three nodes:

- **Publisher**: Provider of Information
- **RV/TM**: Management node operating both the RV/TM core ICN functions. Notice that these function can be operated from two separate nodes; however, for simplicity they have been placed in the same node.
- **Subscriber**: Consumer of Information

Two network interfaces (eth1 & eth2) are required in each node:

- Management interfaces (eth1): these are required for to facilitate the deployment of the ICN topology, including remote login and the transmission of click configuration files to each node, using the deployment tool provided with our platform. For details of how to install the deployment tool, please refer to Section 2.1.3 of the HowTo.

    - Each interface **must be associated with an IP address that provides accessibility to the node**.
    - In this example we associated all interfaces with IPs from the subnet 192.168.56.0/24; however, IP addresses from different subnets can be used as long as they are accessable by the node where you are deploying
- Data Interfaces (eth2): these are required to provide the data connectivity between the ICN nodes. Notice that these interfaces are also used to construct ICN control connectivity, represented by the control path from the RV and the TM to every node in the network. These RV/TM paths are used to communicate control information (e.g. provide a publisher with the FId to transmit information) from the RV/TM to every node.
    - These interfaces don't require IP addresses. If they are assigned IP addresses no error will be produced in this example, but the IP addresses will not be used.
    - In this example, it has been advised to not assign IP addresses to the data interfaces, in order to familiarize the user with this practice as it will be required in other examples.

Notice: It is possible to use a single interface for both management and data connectivity. In this example they have been separated to illustrate a good practice and also to prepare the user for later examples, where this separation is a requirement.

2. Configuration

2.1 Management Network

In each Nodex, configure the management interfaces in your network configuration file (don't forget to replace 'x' in the IP address below with the node number):

```
~$ sudo nano /etc/network/interfaces
```

```
auto eth1
iface eth1 inet static
      address 192.168.56.x
      netmask 255.255.255.0
      broadcast 192.168.56.255
```

2.2 Credentials

To allow the deployment tool to use the configuration file for constructing the ICN setup and access all three nodes , the latter need to have the same credentials (i.e. username and password) for remote login. To do so, create on each node an account with a fixed username and password (e.g. username = `point`, password = `point`). **This must be the same for all three nodes**.

2.3 Passwordless SSH

In order to allow the deployment tool fast access to each node, without having to enter the node password for each remote login process, which can be a very time consuming process, create (if not existent already) a ssh key of the deploying node and copy it to each Nodex in the setup:

```
~$ ssh-keygen -t rsa -b 2048
~$ ssh-copy-id point@192.168.56.x
```

2.4 ICN Configuration

To construct an ICN setup, we have defined a libconfig template that can be used to represent various topologies. for detailed description of this template, refer to Section 3.1 of the HowTo.

For this example setup, the network configuration can be found in `~/blackadder/deployment/examples/basic_3node_icn.cfg`

Notice that, for each node in the configuration file, the IP address of the management interface is used to identify the node for the deployment tool in the field "`tesbed_ip`", while the data interface is used to represent the node connectivity in the "`connections{}`".

3. Network Deployment

Now that interfaces have been configured, the ICN setup can be deployed from any of the nodes in the setup; or, it can also be deployed from a different, fourth, node as long as it has accessibility to all three nodes. To deploy the network, do (assuming blackadder is in your home directory):

```
~$ cd ~/blackadder/deployment/
~$ ./deploy -c examples/basic_3node_icn.cfg -l
```

To check if the core ICN platform is running in each node, grap the process ID of click modular router that should be running the ICN configuration file:

```
~$ pgrep click
```

if a process number is returned then the node is running

3.1 Running the Topology Manager (TM)

If the name of POINT root directory is 'blackadder' and you only need the default TM functionality, then you don't need to go through this section, as the deployment tool will find the path to the TM and run the application as part of deploying the ICN network. However, if the name is otherwise different, or you would like to use the TM with one of its extensions, then you would need to run the TM explicitly as described below.
The TM app is responsible for constructing delivery paths between publishers and subscribers; without having it running no pub/sub communication can happen. The TM uses a topology map of the network, represented in by a graphml file named '`topology.graphml`'. The topology map is created by the deployment tool during the network construction phase, for details of `topolgoy.graphml` see Section 3.1.6 of the HowTo.

Now, in this example we have elected Node2 to be the RV/TM of this network and, therefore, Node2 is where you should run the TM app as follows:

```
Node2$ cd ~/blackadder/TopologyManager
Node2$ sudo ./tm /tmp/topology.graphml
```

Notice that the TM may run with various extensions that provide traffic engineering functionalities. such as:

- `-p, path management`
- `-r, resiliency`
- `-t, traffic_engineering`
- `-q, QoS`

`-p` and `-r,` require also running the Link-state Monitoring application as described in section 3.2.1. `-r` also requires running the Resiliency Manager (RM) Application.
Now the setup is ready to run publish/subscribe applications as described by some of our examples below.

3.2 Example Applications

3.2.1 Video Streaming

This is a simple, vlc-based, video streaming application, having a video publisher that stores content in a predefined directory being able to publish a video catalogue, which an interested subscriber can subscribed for to view the catalogue and pick a video to be streamed. To run the app, whereby you can stream a video from the publisher of this example to the subscriber, do the following:

- Place a video file, (not too high quality) in the directory (this is where the publisher will look for stored videos) : `/home/point/Videos/`
- In the publisher, cd to `/home/point/blackadder/examples/video_streaming/`
- Allow vlc to run with sudo permissions - many solutions are available on the web for how to enable vlc with sudo (**NOTICE: running vlc with sudo permissions may impose security risks - so if you choose to do this step, you do it at your own risk)**
- In the publisher, run the video publisher: sudo ./video_publisher
- Write the name of the video file (e.g. video file name: `video_sample.mp4`) in the command line window of the publisher
- After that , type "." to indicate the end of the video catalogue
- In the subscriber, run the video subscriber: sudo ./video_subscriber (remember to allow vlc with sudo permissions on the subscriber as well)
- You should receive the video file name on the subscriber command line window, if so then just write the name back in the window and vlc should start now.!

3.2.2 ICN Ping

This app allows ICN nodes to ping each other, similar to conventional IP ping. The app consists of two parts:

- **ping_publisher**: comparable to a 'client' in the IP world and runs on the node that want to issue the ping requests.
- **ping_subscriber**: comparable to a 'server' in the IP world and runs on the node that is to be pinged by other nodes. it basically allows the node to respond to ping requests by issued by different ping_publishers.

The app runs with the following configuration parameters, which can be set through command line arguments:

- `implicit_rv` (`boolean 0|1`): determines whether each ping is a full pub/sub operation involving RV match pub/sub and TM path formation, or only the first ping involves RV/TM while the rest is a communicated directly using the FIDs created and cached from the first ping. The purpose of this variation is to evaluate the RV delay, compared to direct communication. **Notice:** that this is highly dependent on the size of the information data structure maintained by the RV. When only this app is running and nothing else - i.e. the data structure is very small - the difference in delay should be negligible

- `ping_subscriber_NodeID` : The NodeID of the ping_subscriber. This can be the full NodeID (e.g. `00000101`) or only the varying part of the ID (e.g. `101` instead of `00000101`)
- `ping_publisher_NodeID` : The NodeID of the ping_publisher, which should be the same as the NodeID stated in the network configuration file, used with the deployment tool. This can be the full NodeID (e.g. `00000101`) or only the varying part of the ID (e.g. `101` instead of `00000101`). Notice this argument is only set with the `ping_publisher`, not the `ping_subscriber`
- `number_of_ping_requests` : This sets the number of requests to be issued from the `ping_publisher`, default to `1000` if no value is provided for this argument. This argument is only configurable with the `ping_publisher`.

To run the app between two ICN nodes (Node1 and Node3 in the topology of figure 2.1) with RV/TM only involved in the first ping (i.e. `implicit_rv = 0`):

- Consider a scenario where Node1 `(101)` wants to check if Node3 `(103)` is active or not. In this case `Node1` will run the `ping_publisher` and Node3 will run the `ping_subscriber`. Lets also consider to set the number of ping requests to `5`
- In each node, do: `cd ~/blackadder/examples/traffic_engineering/`
- in Node1 do:
  - `sudo ./ping_publisher  0 103 101 5`
- in Node3 do:
  - `sudo ./ping_subscriber 0 103`
- Now, you should start observing ping responses on the command line window, with the first ping response having larger delay than subsequent responses.

3.2.3 Link-state Monitoring

This is a control application that allows the ICN node to periodically announce its presence to its neighbours (i.e. immediately adjacent nodes) and at the same time learns about its neighbours and therefore its connectivity to them, when receiving similar advertisements from each neighbour. The objective of the app is to provide a detection mechanism of link failure scenarios for resilience and path management purposes.

The app operates by sending a ‘live’ message periodically using the BROADCAST dissemination strategy, which is also link local (i.e., the packet does not traverse more than one link). At the same time, the app maintains a state of all neighbours, from which it received a similar ‘live’ message, indicating for each neighbor the node ID and a timestamp of the last received ‘live’ from this neighbour. Timestamps have a global lifetime setting in the app which is a configuration parameter that can be set for each node through a command line argument. **If no setting is provided by the user, the default lifetime is set to 3 seconds**.

When the state of a neighbour exceeds its lifetime without receiving a new message (i.e., the time since the last received ‘live’ is larger than the set lifetime), the node considers the link between itself and the respective neighbour to have failed and accordingly sends a Link State Notification to the TM, indicating the link that has failed. It is then up to the TM to take respective actions, depending on whether or not it is operating with the extension for resilience or path management.

To run the app, access each node and do:

- `cd ~/blackadder/examples/traffic_engineering`
- `sudo ./linkstate_monitor`
- When the app runs on all three nodes, Node2 should be receiving ‘live’ messages from both Node1 (`00000101`) and Node3 (`00000103`)

Notice: the this app can be used for basic debugging of ICN network connectivity, confirming whether or not an ICN network setup is configured correctly and each node can see the neighbours it is suppose to connect to, according to the network topology configuration file.