Dynamips / Dynagen Exercises

Contents

We are going to get dynamips up so we can have a single cisco router up and reachable from the class network. You should already have dynamps installed if you run the ansible worksheets earlier. In the case you are working with a(n Ubuntu) machine that does not have dynamips installed you'd need to do the following:

We do **not** need to run this on our infrastructure.

```
$ sudo apt-get install dynamips dynagen
```

Concerning sudo: The command above is the only one in the entire lab where you need to use sudo. Run all the commands that follow as the nsrc user.

1 Download pre-requisites

One of you needs to download pre-requiste files to the server that you all will use, so that person should open an SSH session to your workshop server and run the following.

```
$ cd
$ mkdir -p binary-images dynamips/work
$ cd binary-images
$ wget http://kit1.lab.nsrc.org/downloads/binary-images/c3725-12415T14.bin
$ wget http://kit1.lab.nsrc.org/downloads/binary-images/c7200-1514M4.bin
$ cd ~/workshop-kit
$ git pull
```

1.1 A note about IOS images

Note that CISCO does not allow distribution of IOS images - so technically the way someone gets IOS images is say off a router they are running. There are some license issues as far as running IOS on dynamips goes - officially CISCO does not recognise/officially support this so this is something to keep in mind when running your classes.

2 Fire up dynamips

You will all run dynamips for this session so we need to create a directory structure that will have us avoid stomping on each other. For the exercise bellow, assign yourselves within your group a number from 1 to 3 (which we'll reffer to as X) and pick a name (say your first name) that you will use for a directory (which we'll reffer to as *NAME*)

With that in mind, each one of you needs to open two ssh sessions to the mac mini (in different windows) and do the following:

1. In the first SSH session/window to the server (s1.ws.nsrc.org):

```
$ cd
$ mkdir -p dynamips/NAME/work dynamips/NAME/tmp
$ cd dynamips/NAME/tmp
$ dynamips -H 720X
```

2. In a second SSH session/window to the server (s1.ws.nsrc.org):

```
$ cd $HOME/dynamips/NAME
$ vi routers.net
```

3. Create a routers.net looks like the following: substutute *NAME* and *X* respectively.

```
# Using c7200-advipservicesk9-mz.151-4.M4

model = 7200

[s1.ws.nsrc.org:720X]
    workingdir = /home/nsrc/dynamips/NAME/work
    udp = 1X000

    [[7200]]
        image = /home/nsrc/binary-images/c7200-1514M4.bin
        ram = 176
        npe = npe-400
        ghostios = True
        idlepc=0x60608f64
        slot1 = PA-GE
        slot2 = PA-GE
        slot3 = PA-GE
        slot4 = PA-GE
        slot5 = PA-GE
        slot6 = PA-GE

##########################################
# single router tapped into network
    [[router rX]]
        model   = 7200
        console = 201X
        aux     = 301X
        gi1/0   = NIO_tap:tap-lanX
```

4. Save this and use dynagen to load your routers.net

```
$ dynagen routers.net
```

5. You should now be able to see your router listed in dynagen using the `list` command. You can get more details about a router using the `show device rX` command. You can reload a router using the `reload rX` command. You can see what your .net looks like using the `show run` command. You can also use the ? to see some other dynagen commands or completions to the current command. In this case we've substitued the relevant number with *X* and the chosen name with *NAME*. What you type is at the `=>` prompt.

```
Reading configuration file...

Network successfully loaded

Dynagen management console for Dynamips and Pemuwrapper 0.11.0
Copyright (c) 2005-2007 Greg Anuzelli, contributions Pavel Skovajsa

=> list
Name       Type       State      Server          Console
rX         7200       running    s1.ws.nsrc.org:720X 201X

=> show run
[s1.ws.nsrc.org:720X]
    workingdir = /home/nsrc/dynamips/NAME/work
    udp = 1X000
    [[7200]]
        image = /home/nsrc/binary-images/c7200-1514M4.bin
        ram = 176
        idlepc = 0x60608f64
    [[ROUTER rX]]
        slot1 = PA-GE
        g1/0 = nio_tap:tap-lanX
        slot2 = PA-GE
        slot3 = PA-GE
        slot4 = PA-GE
        slot5 = PA-GE
        slot6 = PA-GE

=> show device rX
Router rX is running
  Hardware is dynamips emulated Cisco 7206VXR NPE-400 with 176 MB RAM
  Router's hypervisor runs on s1.ws.nsrc.org:7211, console is on port 201X
  Image is /home/nsrc/binary-images/c7200-1514M4.bin with idle-pc value of 0x60608f64
```

```
  Idle-max value is 1500, idlesleep is 30 ms
  128 KB NVRAM, 64 MB disk0 size, 0 MB disk1 size
   slot 1 hardware is PA-GE with 1 interface
      GigabitEthernet1/0 is connected to real TAP tap-lanX interface
   slot 2 hardware is PA-GE with 1 interface
      GigabitEthernet2/0 is empty
   slot 3 hardware is PA-GE with 1 interface
      GigabitEthernet3/0 is empty
   slot 4 hardware is PA-GE with 1 interface
      GigabitEthernet4/0 is empty
   slot 5 hardware is PA-GE with 1 interface
      GigabitEthernet5/0 is empty
   slot 6 hardware is PA-GE with 1 interface
      GigabitEthernet6/0 is empty

=> ?

Documented commands (type help <topic>):
========================================
capture  confreg  cpuinfo  export  hist    list  py      save   show   suspend
clear    console  end      filter  idlepc  no    reload  send   start  telnet
conf     copy     exit     help    import  push  resume  shell  stop   ver
```

3 Connect to your router

You should now be able to telnet to the console port that you defined earlier using s1 as the host in another terminal. So if you're on windows open putty and make sure that the protocol is *telnet* and the port is *201X*. On a UNIX terminal (OS X or Linux)

```
$ telnet s1.ws.nsrc.org 201X
```

Press enter to get the "Would you like to enter the initial configuration dialog? [yes/no]:" prompt and just hit control+c. Next up is a basic cisco configuration that should look like the following:

Substitute YYYYYYYY with the secret password distributed in class.

Hint: you can use a text editor to prepare this then paste it in a config session.

```
service password-encryption
hostname rX
enable secret YYYYYYYY
!
```

```
aaa new-model
aaa authentication login default local
aaa authentication enable default enable
username nsrc secret YYYYYYYY
!
interface GigabitEthernet1/0
 description link to outside
 ip address 10.10.0.21X 255.255.255.0
 no shutdown
!
ip route 0.0.0.0 0.0.0.0 10.10.0.254
!
ip domain-name ws.nsrc.org
no ip domain-lookup
crypto key generate rsa modulus 2048
ip ssh version 2
```

4 Verify connectivity

1. From the router

   ```
   rX# ping 10.10.0.254
   rX# ping 8.8.8.8
   ```

2. From a pc connected to your wifi

   ```
   ping 10.10.0.21X
   ```

3. You should be able to login to your router using SSH

4.1 How connectivity works:

Remember the ansible scripts created two "bridges" and connected "tap" interfaces to the bridges. To see the bridges you have you can use the `brctl show` command

```
nsrc@kit1:~$ brctl show br-wan
bridge name     bridge id               STP enabled     interfaces
br-wan          8000.685b35841afa       no              eth0
                                                        tap0
                                                        tap1
nsrc@kit1:~$ brctl show br-lan
bridge name     bridge id               STP enabled     interfaces
```

```
br-lan          8000.02c385fcdd6a       no              eth1
                                                        tap-lan0
                                                        tap-lan1
                                                        tap-lan2
                                                        tap-lan3
                                                        tap-lan4
                                                        tap-lan5
                                                        tap-lan6
                                                        tap-lan7
                                                        tap-lan8
                                                        tap-lan9
```

So each of these routers live in a separate dynamips cloud but connect to the br-lan bridge through interfaces tap-lan1, tap-lan2 or tap-lan3 depending on what you put in your routers.net file.

The ip address you set as your default gateway sits on the bridge itself.

```
nsrc@s1:~$ ip addr show br-lan
4: br-lan: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN
    link/ether 06:99:8e:50:dc:16 brd ff:ff:ff:ff:ff:ff
    inet 10.10.0.241/24 brd 10.10.0.255 scope global br-lan
    inet 10.10.0.254/24 brd 10.10.0.255 scope global secondary br-lan:0
nsrc@s1:~$
```

The following diagram illustrates this:

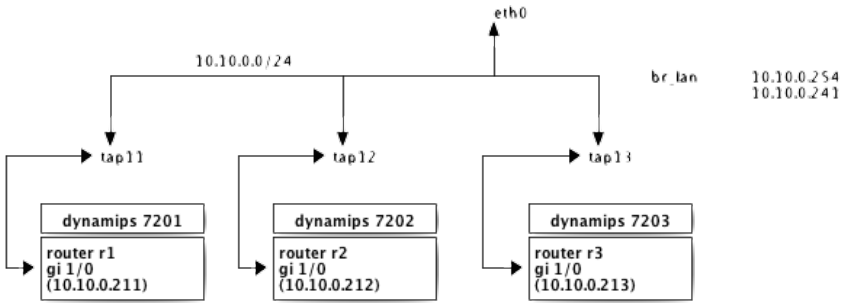

Figure 1: Router connection to LAN

5 Clean up

1. In the second window where you run dynagen ..

```
=> exit
```

2. In the first window where you run Dynamips, use control+c to quit it.
3. You can safely clean out the tmp and work directories

```
cd
rm dynamips/NAME/work/* dynamips/NAME/tmp/*
```