Sunday at the Village Vanguard

Kévin "Chewie" Sztern

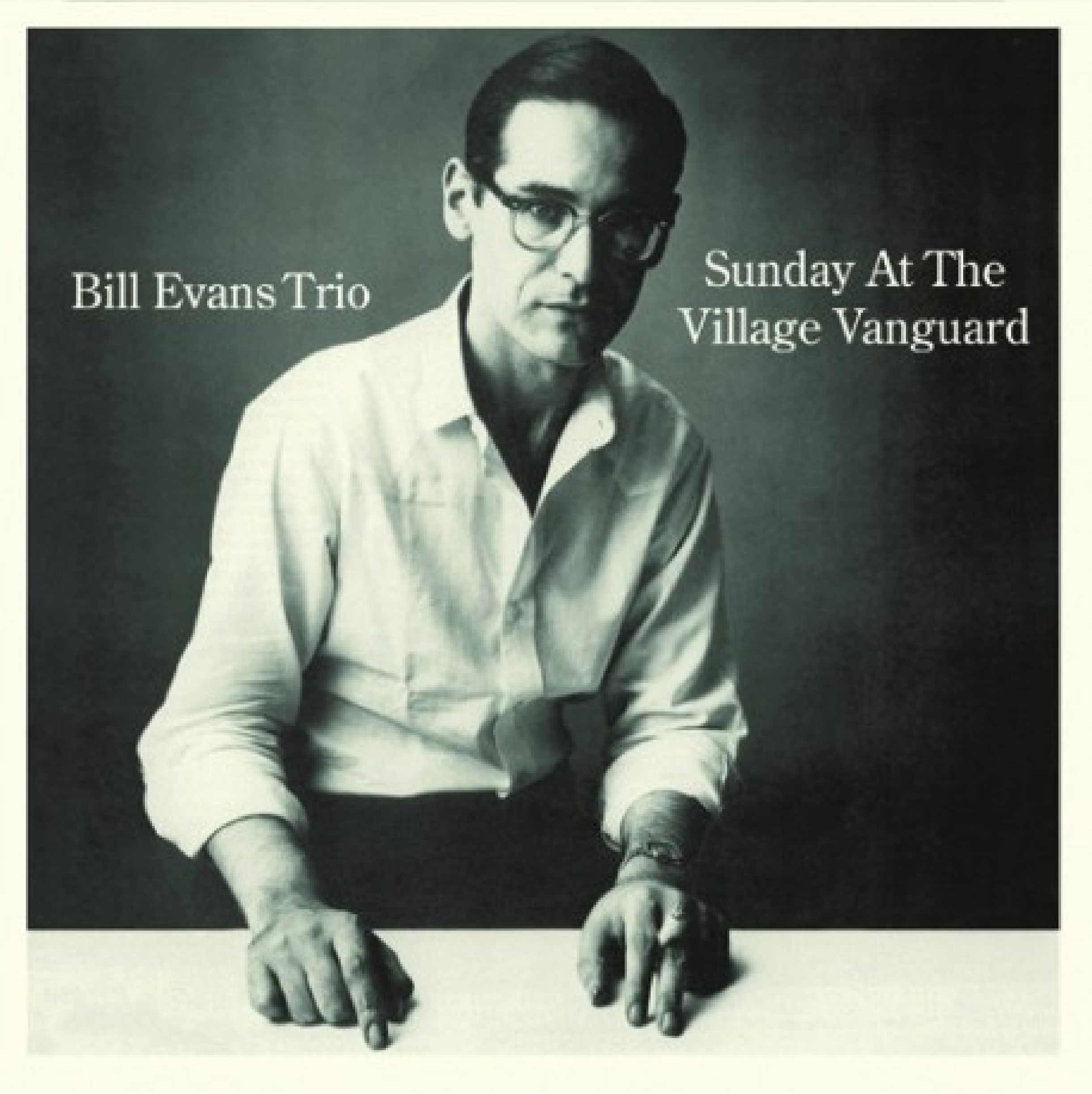
Jan 14, 2021

CONTENTS:

CHAPTER

ONE

ABOUT THIS PROJECT



1.1 *Sunday at the Village Vanguard*

Welcome to the second NET1 project!

Another project, another jazz album. This time, it's *Sunday at the Village Vanguard*, by the Bill Evans trio. Probably the most influential jazz trio of all time, the group was spearheaded by the legendary pianist Bill Evans (you might have already heard him as the Miles Davis' pianist in *Kind of blue*), accompanied by bassist Scott LaFaro and drummer Paul Motian, all of which will have the honor of naming one part of this project.

Sunday at the Village Vanguard can easily be considered the greatest achievement of the trio. Tragically, Scott LaFaro died only ten days after recording.

1.2 What you have to do

This project is all about *iptables*. NAT, firewalls, those two concepts will hold no secret from you afterwards!

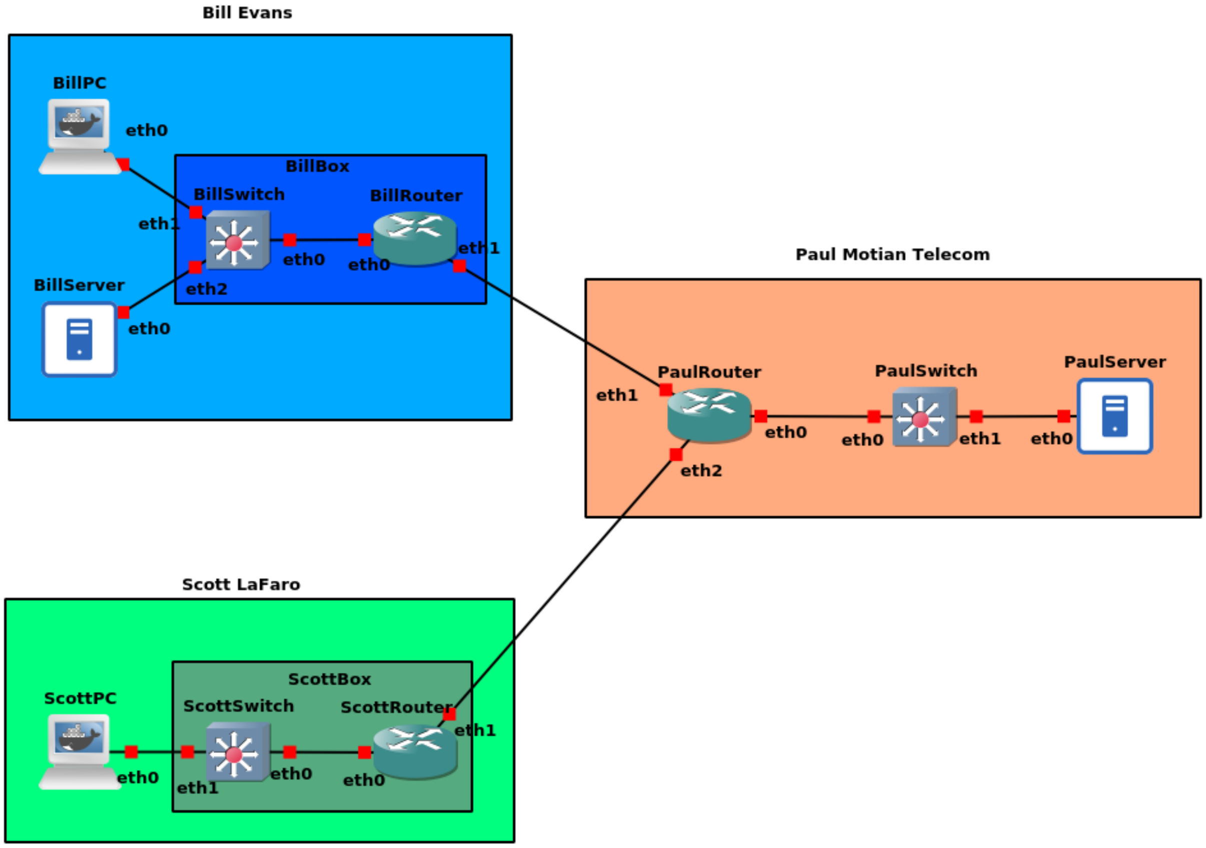
iptables is a complicated beast, and you will probably need to do some research on your own in order to use it properly. Some starting points are available on Moodle.

Note: A successor?

Recently, there's been a new kid on the block, *nftables*, which aims to replace *iptables* as the new tool of choice for packet filtering. Since this is still relatively recent, we'll stick to *iptables* as it is the most likely to encounter in the wild.

Since creating all the machines is laborious and not very interesting, you will find on the assignment page a base project that you can import in GNS3. In this base project, all the machines are present and correctly plugged, but no configuration is present. All you have to do is go to each machine and configure it as instructed!

The topology looks like this:



There are three main areas of interest in this topology, each named after a member of the Bill Evans trio. In particular, Bill and Scott act as *clients*, using the services of Paul, who plays the role of the Internet Service Provider.

Note the appearance of a new node type : the webserver. This appliance works exactly like a hosts, except it contains an HTTP server that can be reached on port 80. In addition, this server is configured to return its hostname and the IP of the client when contacted. Very useful for debugging! You don't have to add the appliance, GNS3 should take care of everything when importing the base project (you won't see it in your toolbox, but it's ok).

1.3 Exporting your project

Once again, I ask you to submit an export in the portable GNS3 format, using the "Export portable project" entry in the "File" menu.

Name the file *<login>-sunday.gns3project*, where *<login>* is your login. This is the file you will submit through Moodle. Be careful to input your login **exactly** as found on the intranet, and keep it all lowercase.

Before submitting it, I strongly advise you to try importing it first and testing that everything works correctly, you can never be too sure! I'm not responsible for errors on your parts and won't help you if you forgot to test your project before submitting it.

Now, let's go through through each zone one by one!

CHAPTER

TWO

BILL

Bill is a customer of Paul, the ISP (*Internet Service Provider*). As such, he was given a box to use at home, made up of a switch and a router. On this box, he has plugged two machines: *BillPC* and *BillServer*.

2.1 IP addressing

Paul has only given one address to Bill: *198.51.100.11*, part of a */31* subnet. The other machine of the subnet, unsurprisingly, lives at the other end of the box, inside the ISP's premises, with the IP *198.51.100.10*. As such, *198.51.100.11* is the IP that should be configured for *eth1* on *BillRouter*. Of course, *198.51.100.10* acts as the default gateway for *BillRouter*.

Since only one address was given, Bill uses a private IP subnet, *192.168.1.0/24*, for each of his machines. *192.168.1.1* lives on the internal interface of *BillRouter*, *BillPC* has *192.168.1.2*, and finally *BillServer* has *192.168.1.3*.

2.2 NAT

Since those are private addresses, Bill needs to configure SNAT (also known as masquerading) on the router, to hide those addresses with the public one.

To make things more complicated, Bill runs a webserver on *BillServer*. Thus, he also needs to configure *DNAT* on the router to forward incoming connections on port *80* to *BillServer* on port *80*.

For complicated reasons (look up NAT hairpinning for more info), using this DNAT rule from inside (as in, contacting *198.51.100.11:80* from *BillPC*) is actually quite tricky, so this scenario won't be tested.

Warning: Don't forget to save the *iptables* configuration with *iptables-save*! Otherwise, all your precious configuration will be lost on reboot.

Also, do remember that *iptables-save* only prints the configuration on stdout, you need to redirect the output to */etc/sysconfig/iptables* for the save to take effect.

2.3 Recap

To recap:

Name	IP Addresses
BillPC	• 192.168.1.2/24 (eth0)
BillServer	• 192.168.1.3/24 (eth0)
BillRouter	• 192.168.1.1/24 (eth0) • 198.51.100.11/31 (eth1)

In real life…

The connection to the ISP is of course not done with an ethernet cable, but with a dedicated link like DSL, cable or fiber. The box probably doesn't connect directly to a router, but instead goes through various systems like a DSLAM and a BRAS.

CHAPTER THREE

SCOTT

Scott's situation is very similar to Bill, with a very simple twist: a bit of an airhead, Scott went too quickly in his installation and did not configure his network appropriately. As such, while his IP addressing is correct, there is **NO** NAT configuration and the private addresses leak to the ISP. Those private addresses, as you will see on the next area, will be dropped by the ISP firewall, and Scott will have no outside connectivity. This scenario is very useful to verify that Paul's configuration was done correctly.

3.1 IP addressing

Paul has given one address to Scott: *198.51.100.69*. Just like Bill, this IP is part of a */31* block, and the ISP holds the corresponding end, *198.51.100.68*. The ISP end acts as the default gateway for *ScottRouter*.

On the internal side, *ScottRouter* has *192.168.1.1/24*, while *ScottPC* has *192.168.1.2/24*.

3.2 Recap

To recap:

Name	IP Addresses
ScottPC	• 192.168.1.2/24 (eth0)
ScottRouter	• 192.168.1.1/24 (eth0) • 198.51.100.69/31 (eth1)

CHAPTER

FOUR

PAUL

Paul (or rather his very successful company, *Paul Motian Telecom*) is the local ISP (*Internet Service Provider*). Additionally, he also acts as a *Content Service Provider*, by offering an innovative social network for jazz enthusiasts, running on *PaulServer*.

To do this job, Paul was assigned the *198.51.100.0/24* block from his RIR, which he separated in two subnets:

- *198.51.100.0/25* is used to offer connectivity to his clients
- *198.51.100.128/25* is used for internal services and infrastructure

For each new customer, Paul carves out a new */31* block from *198.51.100.0/25*, and gives one end to the customer, keeping the other end on its router for connectivity.

In particular, Paul has two customers, Bill and Scott, each with a part of their own */31* subnet. The *PaulRouter* router thus holds the other ends on *eth1* and *eth2*.

4.1 The webserver

Paul also offers a web application running on *PaulServer*. Just like *BillServer*, the HTTP server comes pre-configured and enabled, all you need is the IP configuration!

The IP to use is *198.51.100.142/25*. I'll let you deduce what should the gateway be.

4.2 Firewalling

One of the jobs of the ISP is making sure that no private addresses leak out from its customers. To do this, the *PaulRouter* router is configured to drop all packets containing those addresses. In our case and to save some time, dropping forwarded packets with a source IP in *192.168.0.0/16* will be enough.

Warning: Just like Bill, don't forget *iptables-save*! Once again, don't just type it, redirect it appropriately!

4.3 Recap

To recap:

Name	IP Addresses
PaulRouter	• 198.51.100.129/25 (eth0) • 198.51.100.10/31 (eth1) • 198.51.100.68/31 (eth2)
PaulServer	• 198.51.100.142/25 (eth0)

CHAPTER

FIVE

CONCLUSION

Now that you have discovered the joy of configuring (and debugging!) *iptables*, please join me in the traditional network engineer prayer:

```
NAT is shit
Who the fuck thought this was a good idea
Why are we still not using IPv6
I want to punch a puppy
```

Go in peace, my brothers and sisters.