



Guide

Installing an Intrusion Detection System on Ubuntu Server 22.04

- Nuno Romão

Objective:

In this lab you'll be able to follow the step-by-step guide to install an Intrusion Detection System on your organization using an Ubuntu Server 22.04. This guide will provide steps to install the server, snort and run a few configurations for your test environment.

We will be installing SNORT as the Intrusion Detection System on an Ubuntu Server 22.04.

Let's install our Ubuntu Server from scratch.

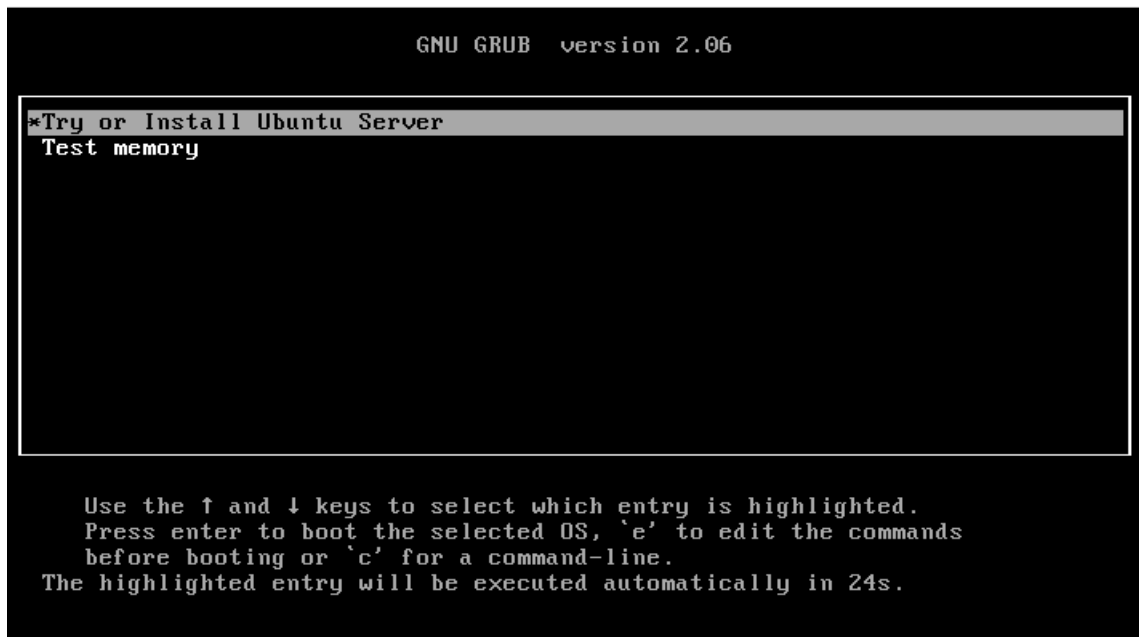


Figure 1 - Ubuntu Server boot

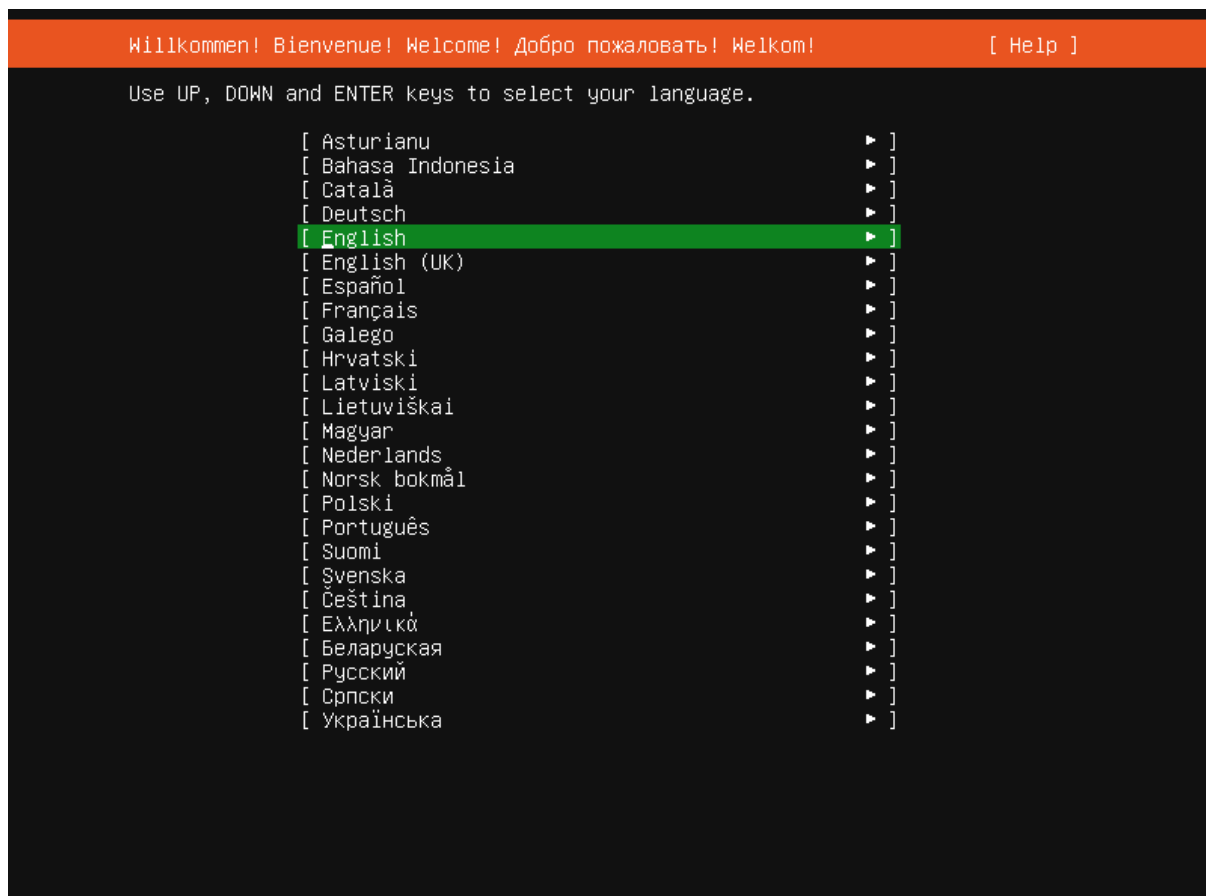


Figure 2 - Select the language for your server

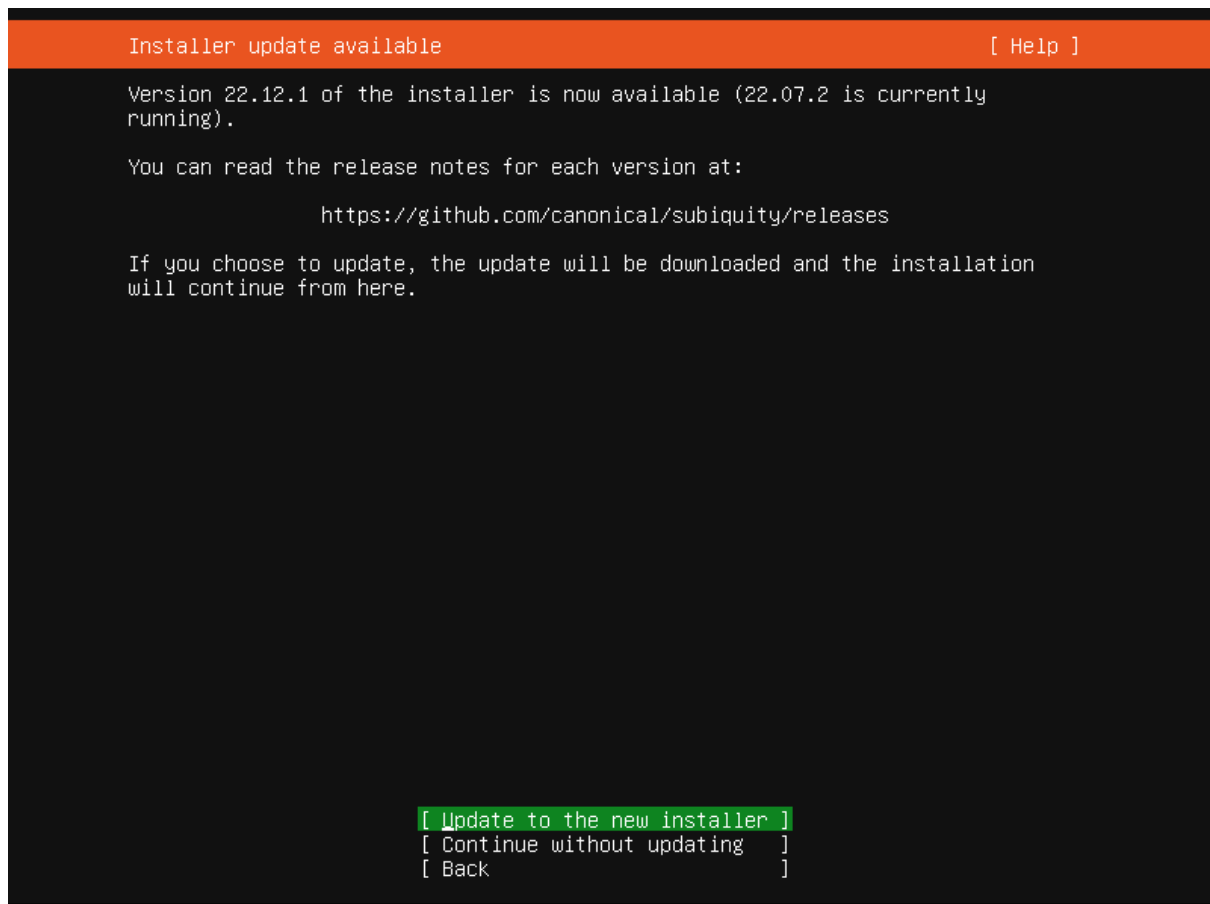


Figure 3 - Update your installer

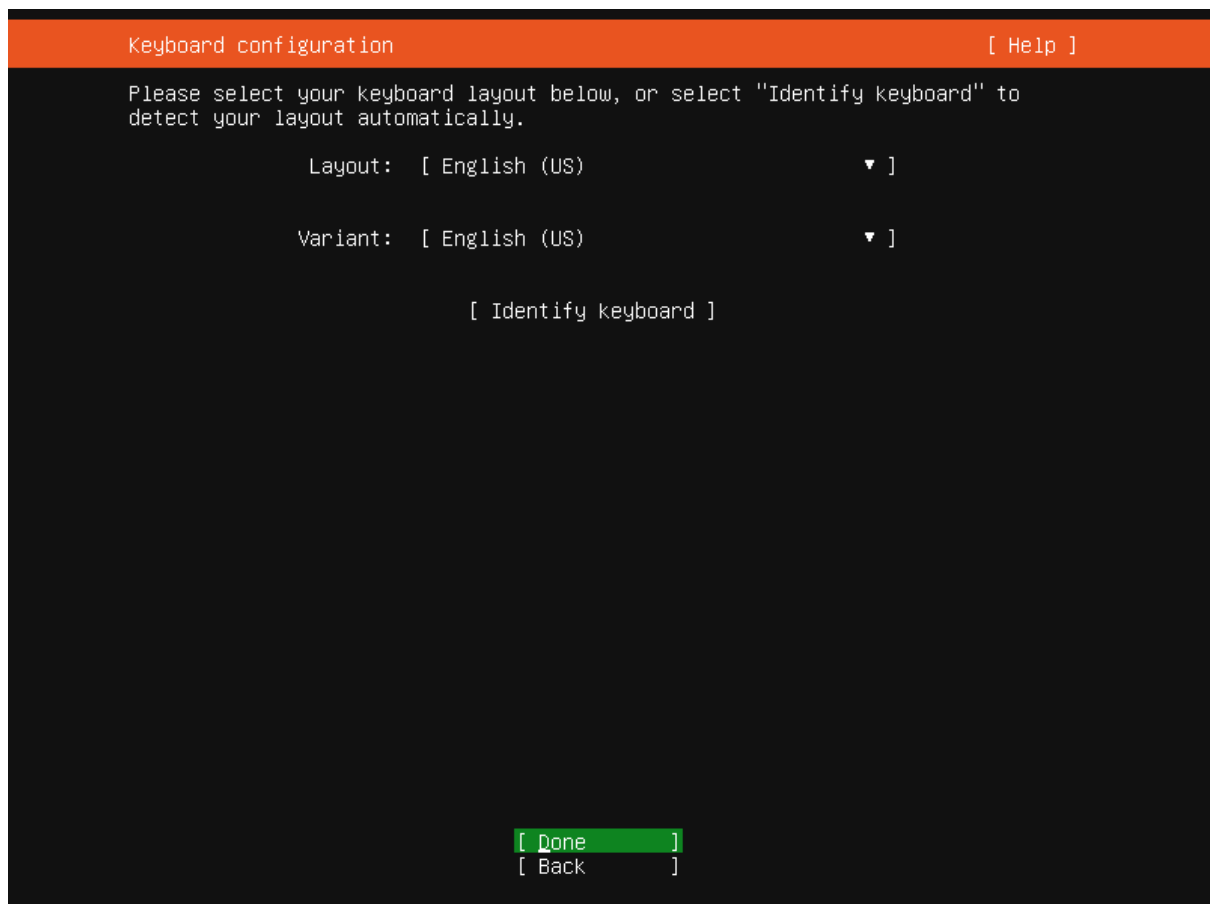


Figure 4 - Keyboard language selection

Choose your keyboard language.

Select the minimized version of Ubuntu, as this server will be used only as a sensor on the network to detect possible attacks.

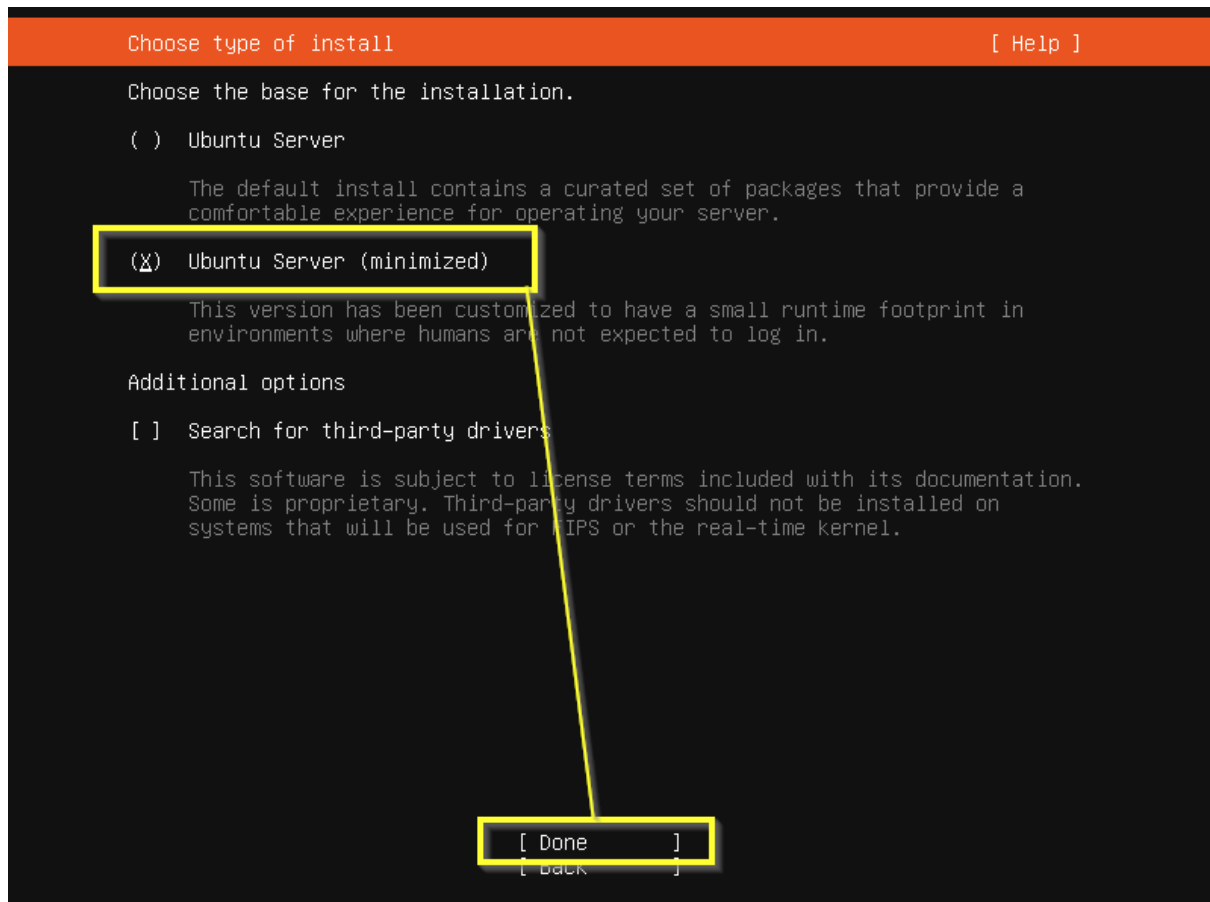


Figure 5 - Ubuntu Server option selection

Leave the network as dhcp for the time being, we'll change the IP address later. (if you want to set the IP now, you can do it, your choice...)

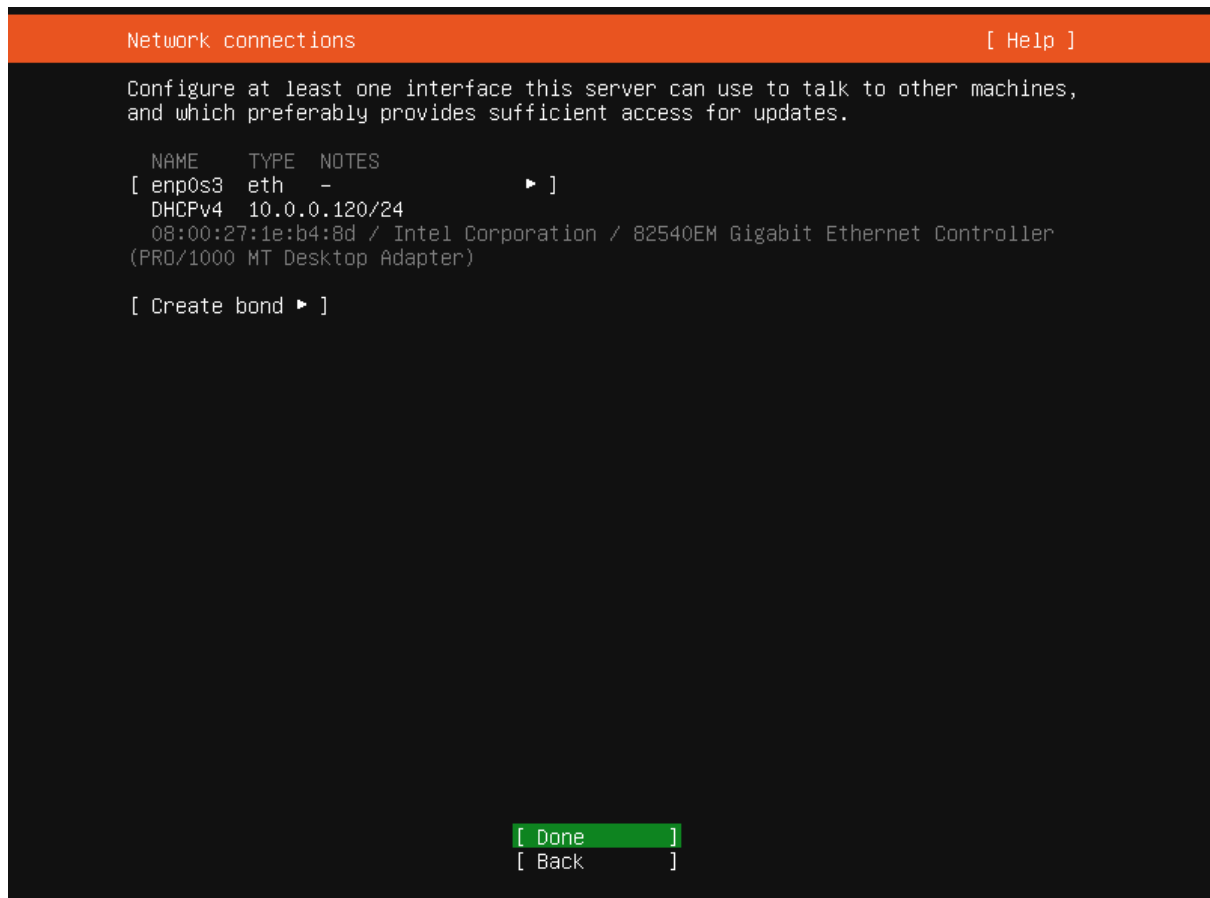


Figure 6 - Server network configuration

Ignore the proxy address, unless you are in a restricted network where it is necessary to configure a proxy address.

Configure proxy [Help]

If this system requires a proxy to connect to the internet, enter its details here.

Proxy address:

If you need to use a HTTP proxy to access the outside world, enter the proxy information here. Otherwise, leave this blank.

The proxy information should be given in the standard form of "http://[[user] [:pass]@]host[:port]/".

[Done]
[Back]

Figure 7 - Ubuntu Server Proxy Address

Leave the default mirror address or use another mirror closer to your location.

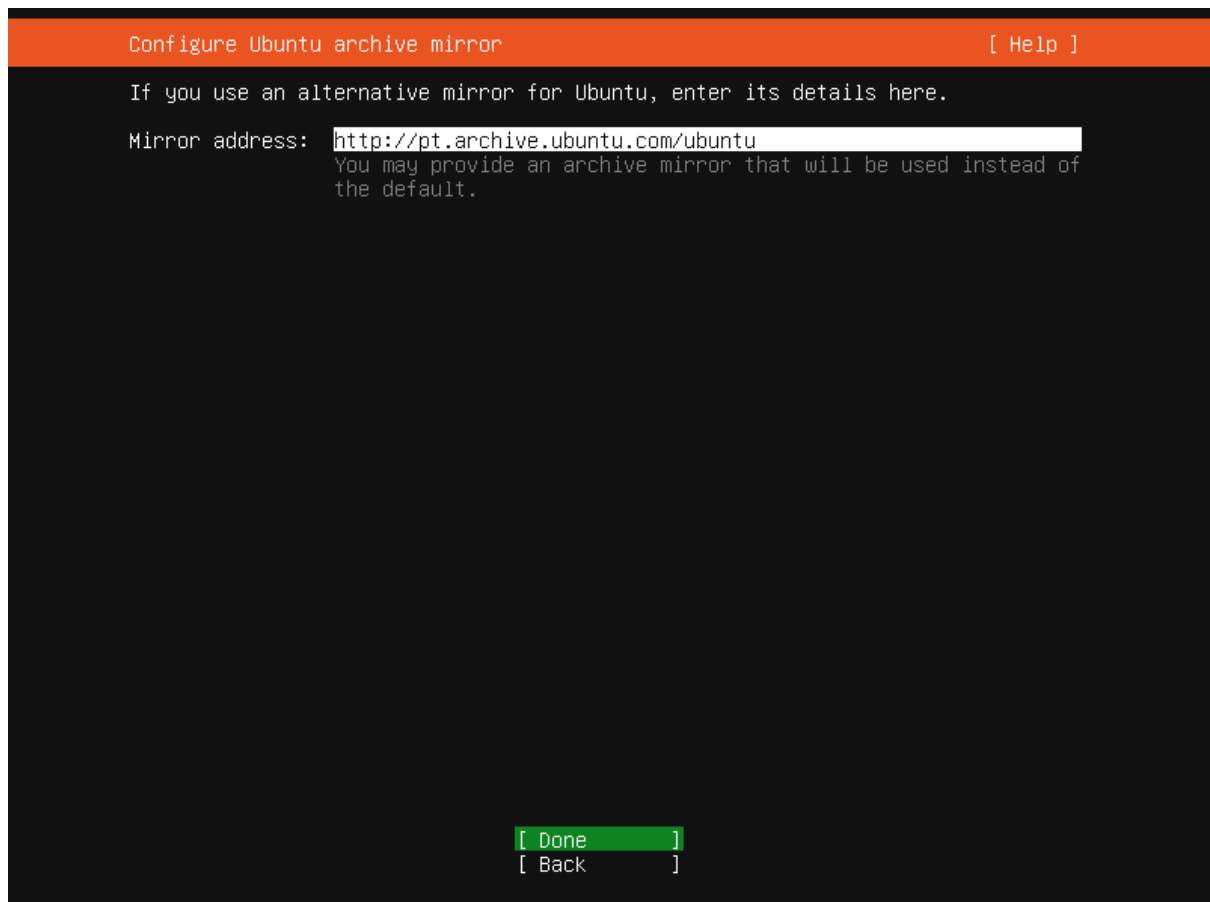


Figure 8 - Ubuntu Mirror Address configuration

If you want to save space, remove the LVM option and continue.
(it doesn't affect what we'll be doing)

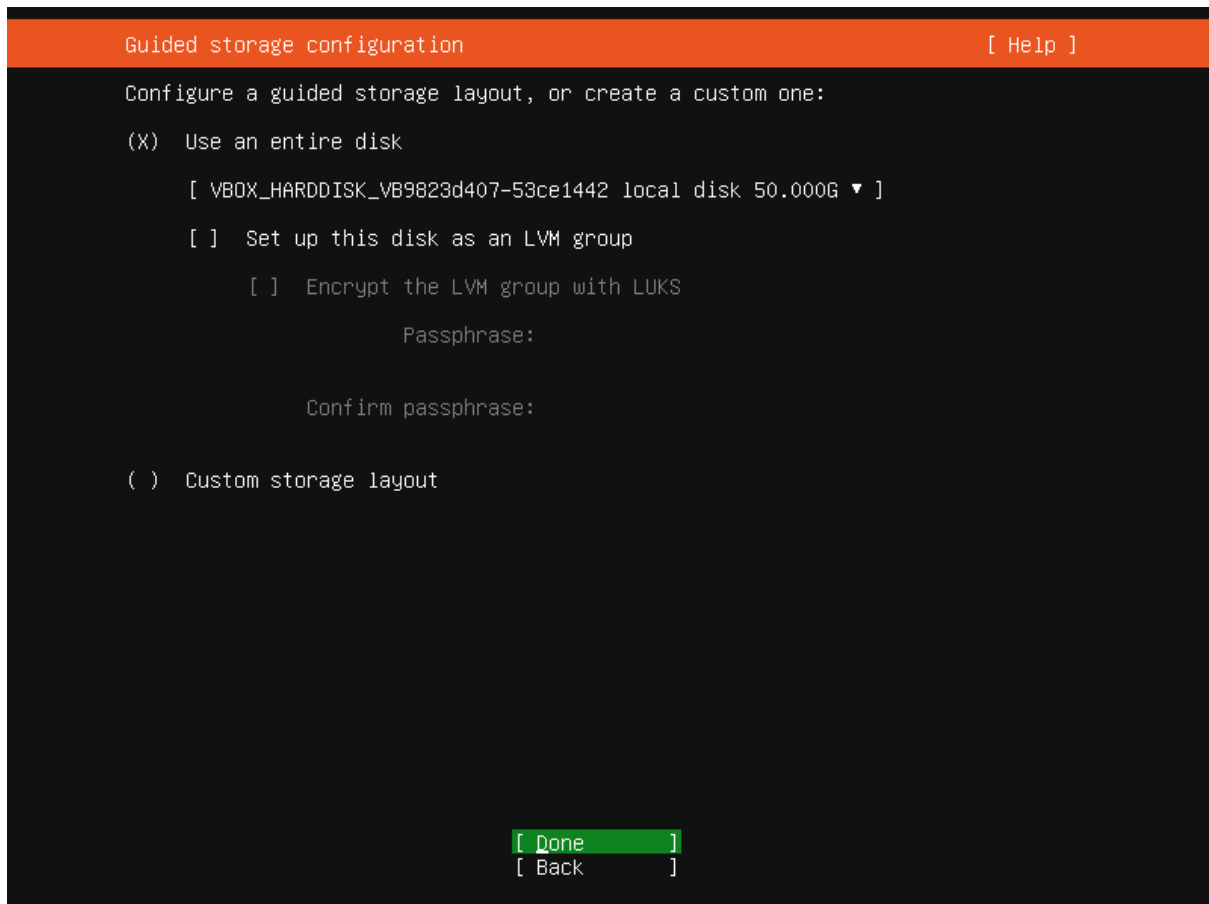


Figure 9 - Ubuntu LVM setup

Lets proceed.

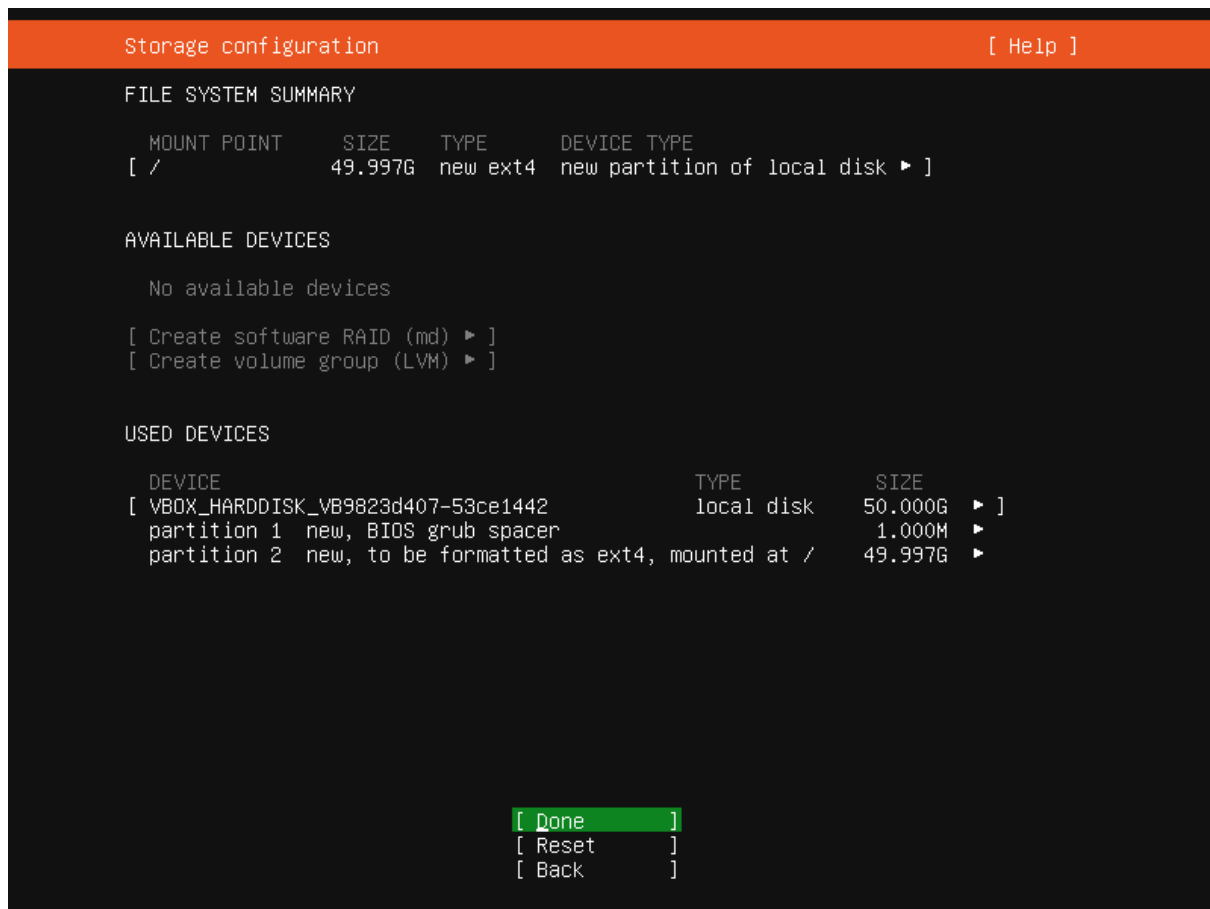


Figure 10 - Storage Configuration

Continue with the installation process.

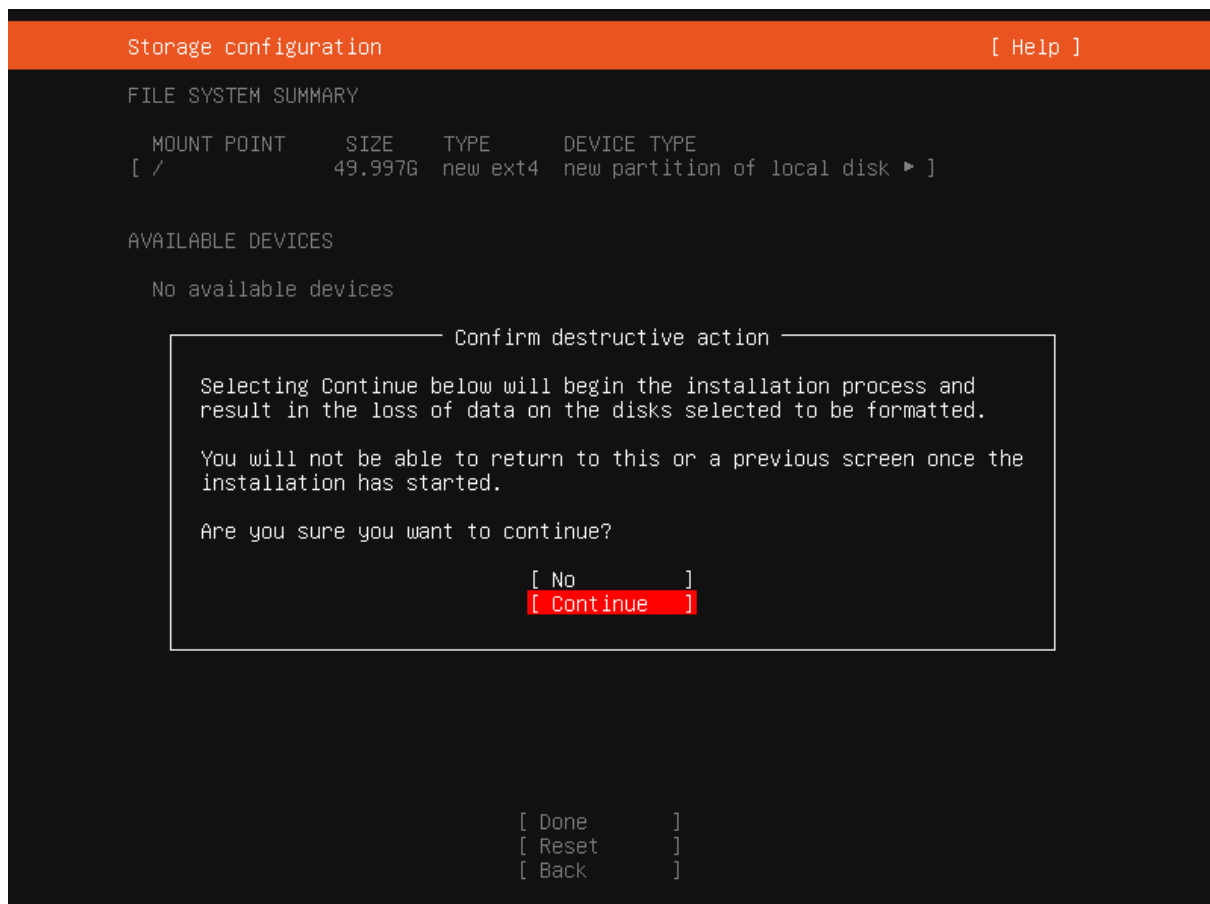
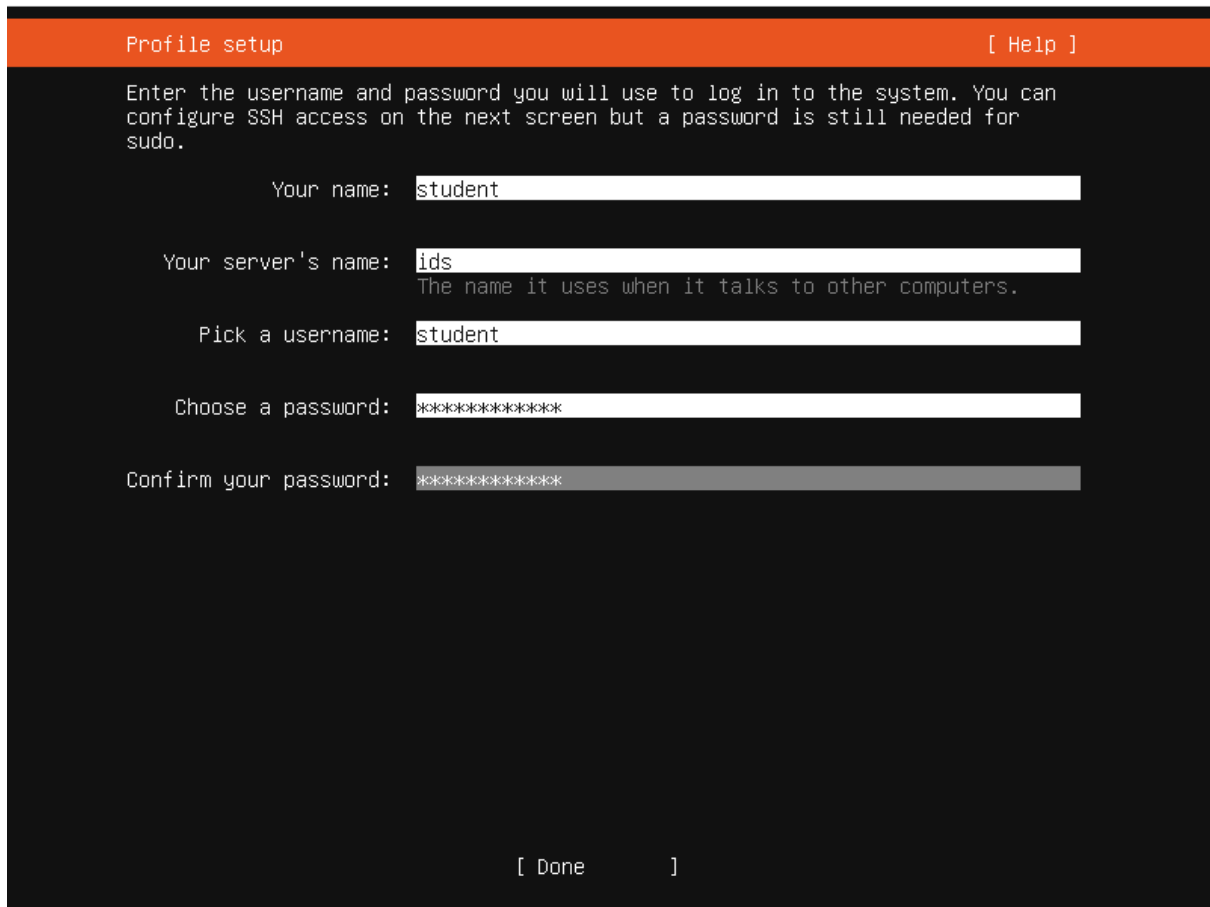


Figure 11 - Installation confirmation

Setup an account for your server.



The screenshot shows a terminal window titled "Profile setup" with a "[Help]" link in the top right corner. The instructions state: "Enter the username and password you will use to log in to the system. You can configure SSH access on the next screen but a password is still needed for sudo." The form contains five input fields: "Your name:" with the value "student", "Your server's name:" with the value "ids" and a subtext "The name it uses when it talks to other computers.", "Pick a username:" with the value "student", "Choose a password:" with masked characters "*****", and "Confirm your password:" with masked characters "*****". At the bottom, there is a "[Done]" button.

```
Profile setup [ Help ]

Enter the username and password you will use to log in to the system. You can
configure SSH access on the next screen but a password is still needed for
sudo.

Your name: student

Your server's name: ids
The name it uses when it talks to other computers.

Pick a username: student

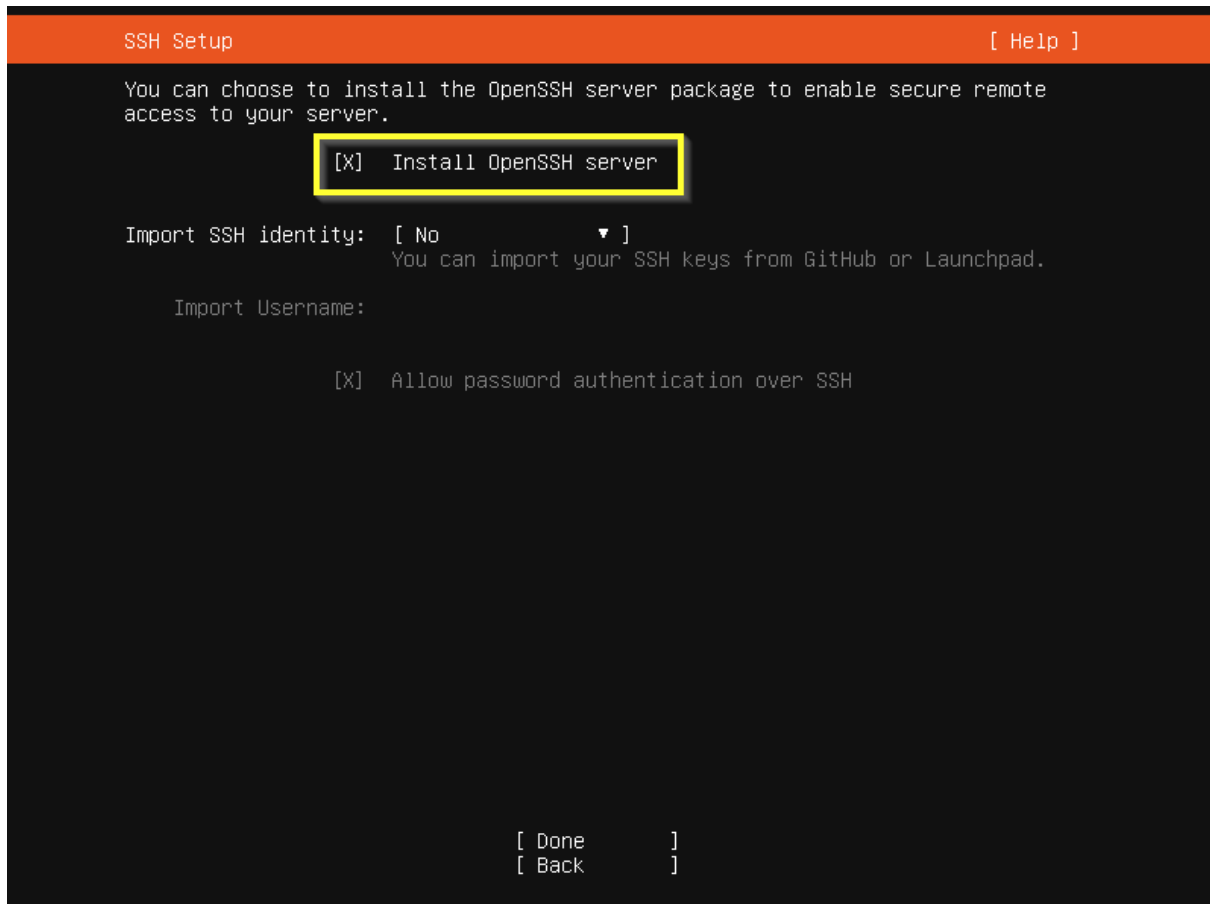
Choose a password: *****

Confirm your password: *****

[ Done ]
```

Figure 12 - User account setup

Install OpenSSH server to login remotely later.

A terminal window titled "SSH Setup" with a red header bar. The header bar contains the title "SSH Setup" on the left and "[Help]" on the right. The main content area is black with white text. It starts with the instruction "You can choose to install the OpenSSH server package to enable secure remote access to your server." Below this is a yellow-bordered box containing the text "[X] Install OpenSSH server". Underneath, it says "Import SSH identity: [No ▼]" followed by the subtext "You can import your SSH keys from GitHub or Launchpad." Then, it asks "Import Username:" followed by a blank line. Below that is "[X] Allow password authentication over SSH". At the bottom right, there are two options: "[Done]" and "[Back]".

```
SSH Setup [ Help ]

You can choose to install the OpenSSH server package to enable secure remote
access to your server.

[X] Install OpenSSH server

Import SSH identity: [ No ▼ ]
You can import your SSH keys from GitHub or Launchpad.

Import Username:

[X] Allow password authentication over SSH

[ Done ]
[ Back ]
```

Figure 13 - Setup OpenSSH to connect remotely

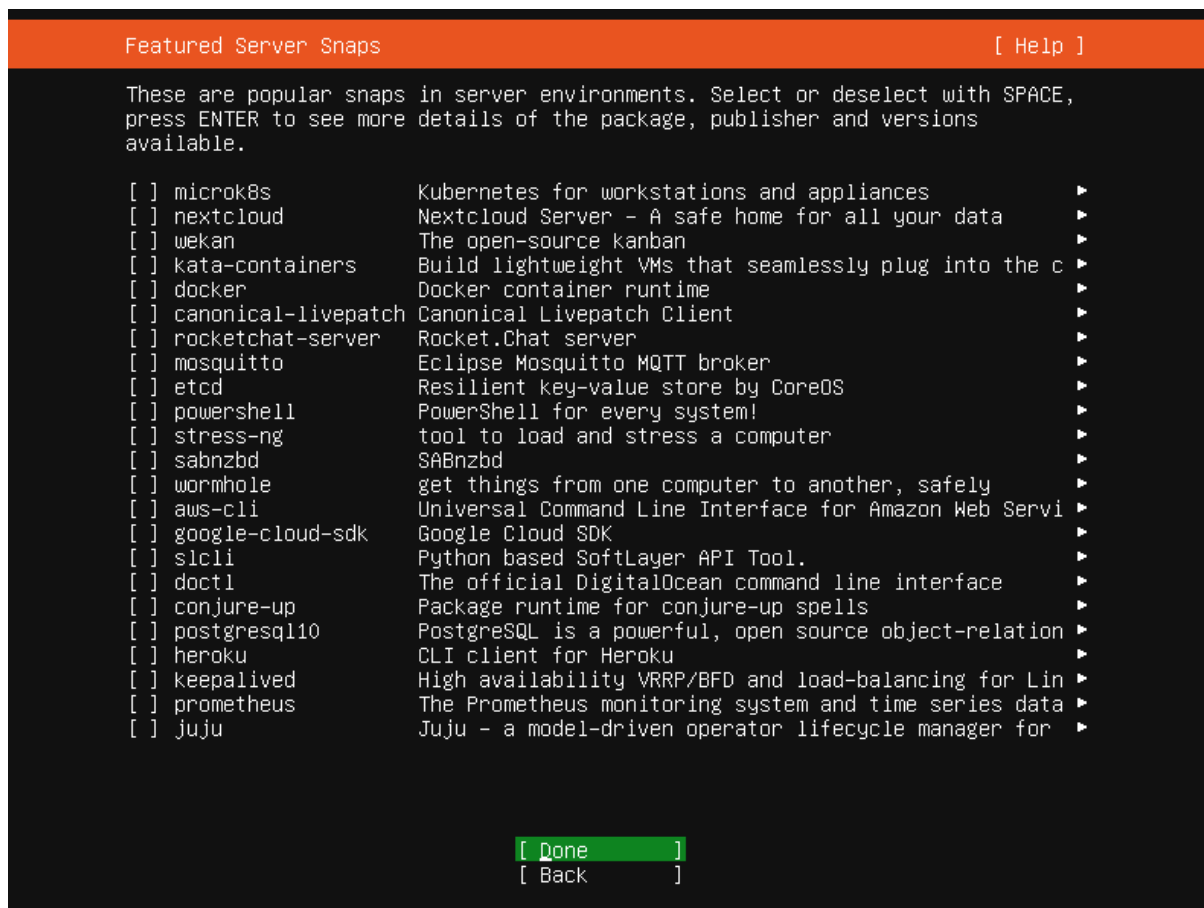
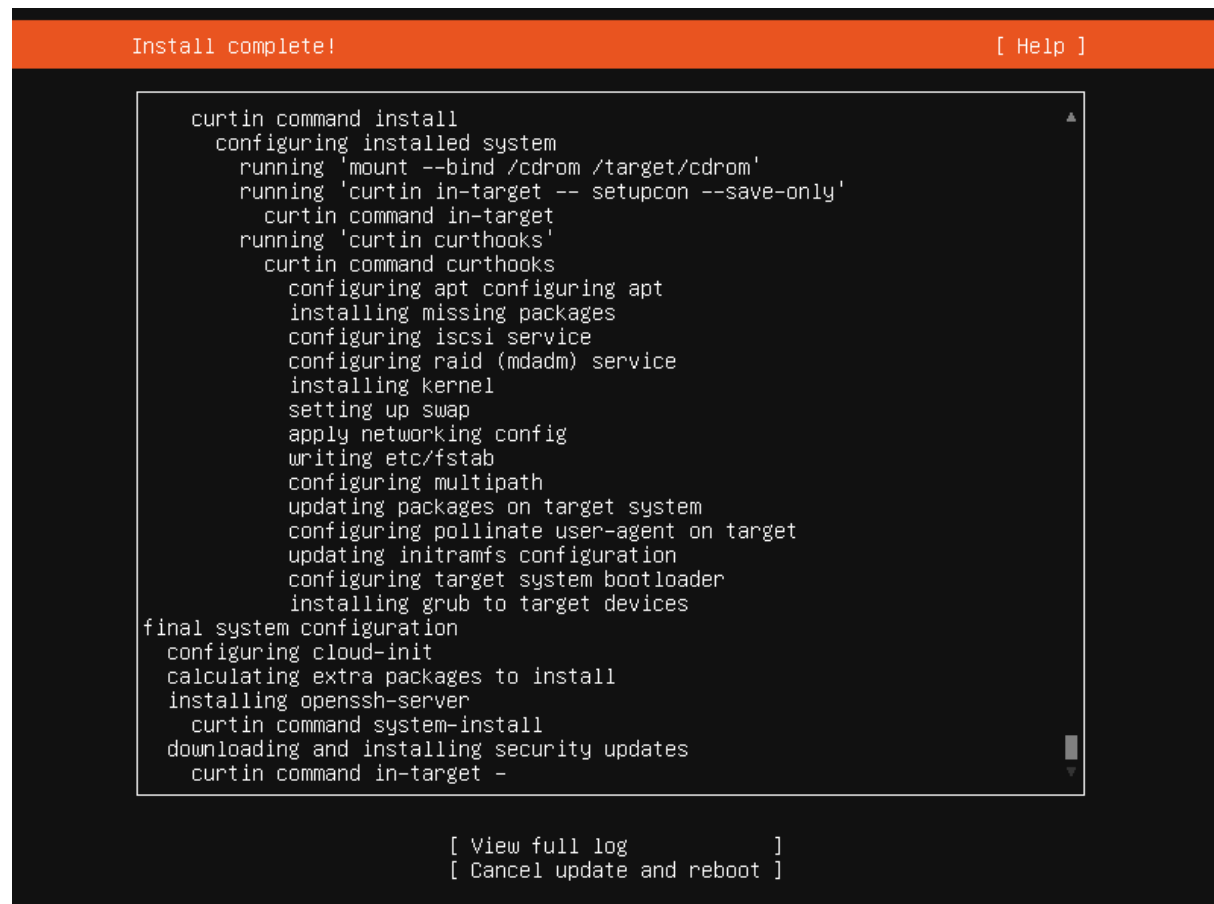


Figure 14 - Ignore additional packages and continue

Let the installation and update process finish.

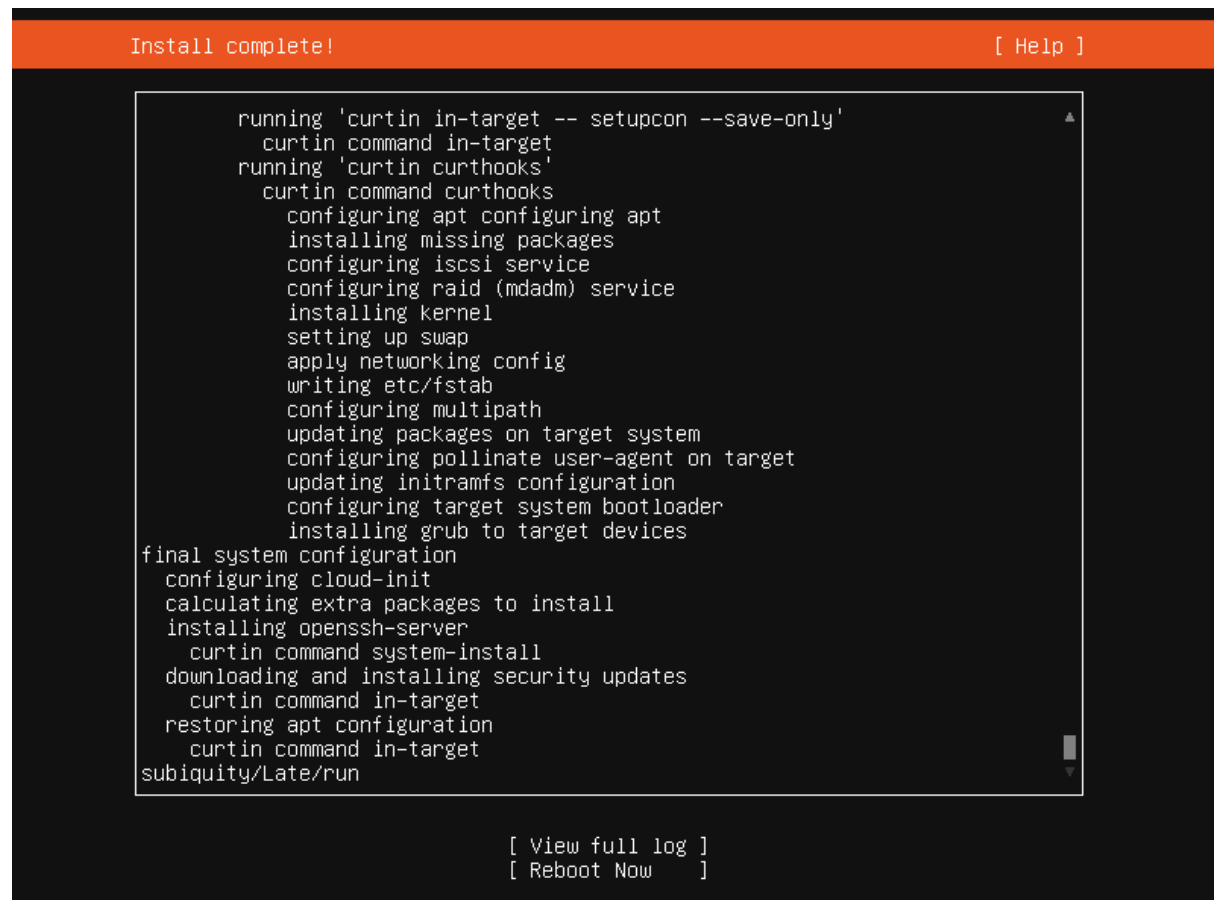


```
Install complete! [ Help ]

curtin command install
  configuring installed system
    running 'mount --bind /cdrom /target/cdrom'
    running 'curtin in-target -- setupcon --save-only'
  curtin command in-target
    running 'curtin curthooks'
  curtin command curthooks
    configuring apt configuring apt
    installing missing packages
    configuring iscsi service
    configuring raid (mdadm) service
    installing kernel
    setting up swap
    apply networking config
    writing etc/fstab
    configuring multipath
    updating packages on target system
    configuring pollinate user-agent on target
    updating initramfs configuration
    configuring target system bootloader
    installing grub to target devices
final system configuration
  configuring cloud-init
  calculating extra packages to install
  installing openssh-server
    curtin command system-install
  downloading and installing security updates
    curtin command in-target -

[ View full log ]
[ Cancel update and reboot ]
```

When finished, reboot your server.



```
Install complete! [ Help ]

running 'curtin in-target -- setupcon --save-only'
  curtin command in-target
running 'curtin curthooks'
  curtin command curthooks
    configuring apt
    installing missing packages
    configuring iscsi service
    configuring raid (mdadm) service
    installing kernel
    setting up swap
    apply networking config
    writing etc/fstab
    configuring multipath
    updating packages on target system
    configuring pollinate user-agent on target
    updating initramfs configuration
    configuring target system bootloader
    installing grub to target devices
final system configuration
  configuring cloud-init
  calculating extra packages to install
  installing openssh-server
  curtin command system-install
  downloading and installing security updates
  curtin command in-target
  restoring apt configuration
  curtin command in-target
subiquity/Late/run

[ View full log ]
[ Reboot Now ]
```

When the system boots for the first time, it will generate the SSH keys, you can ignore it and login to the system to setup your IP address.

```
Ubuntu 22.04.1 LTS ids tty1
ids login: [ 19.564498] cloud-init[746]: Cloud-init v. 22.2-0ubuntu1~22.04.3 running 'modules:conf
ig' at Mon, 13 Feb 2023 10:53:22 +0000. Up 19.47 seconds.
[ 20.460111] cloud-init[762]: Cloud-init v. 22.2-0ubuntu1~22.04.3 running 'modules:final' at Mon,
13 Feb 2023 10:53:23 +0000. Up 20.36 seconds.
ci-info: no authorized SSH keys fingerprints found for user student.
<14>Feb 13 10:53:23 cloud-init: #####
<14>Feb 13 10:53:23 cloud-init: -----BEGIN SSH HOST KEY FINGERPRINTS-----
<14>Feb 13 10:53:23 cloud-init: 1024 SHA256:Qa0PYGQqjijhALxRzBwP9C8a2W/Bn4mSN+zpnXm9gcE root@ids (DS
SA)
<14>Feb 13 10:53:23 cloud-init: 256 SHA256:pQbsN80YTDNWvshubDmDriQkD50lcmYNAPG9rWJp690 root@ids (ECD
SA)
<14>Feb 13 10:53:23 cloud-init: 256 SHA256:u1r2TyZ93qpcXI5bilVkJhMjThqJd0nCy/wEFLtkmXs root@ids (ED2
5519)
<14>Feb 13 10:53:23 cloud-init: 3072 SHA256:0vv8TXZKwFvUEM0od7ZGb4d1XzLaefp4d3nc2UQ2rC4 root@ids (RS
SA)
<14>Feb 13 10:53:23 cloud-init: -----END SSH HOST KEY FINGERPRINTS-----
<14>Feb 13 10:53:23 cloud-init: #####
-----BEGIN SSH HOST KEY KEYS-----
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAuNTYAAAAIbmlzdHAuNTYAAABBBB5E0JnKr1+RbYnXbXU/zVxte0z3
+Xzu1BPI7ewYdtXp6QwbHqU2TYQtrmFbCHdsEqmNSqHOUT/BbvqniH4w5U= root@ids
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIMJ3eixChDXEEhWMYuoNh8WRE0n9NfjZDxMm4oW4ig+M root@ids
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCg4iG5tLI8EchWdswymAS0VQD34vg+eibSOX0WRfIOKJRso7DoS4FIX2ZTGaEo
J2nEDI6PXqCpChM1Ipy9EMN2sK3nmie2Uibj3lX/Gx5NiBItY9ppli4TnsLGXIP6Ir6Ydi+eosW0HIxC6l6uyQb+y65H2oz84yX2
Jhi1/Dv+lmnv25keYr9KtPq3H9bQemjjbwJxqQoJnbk0K50THXGdI2nMRq8oxL3k5pGDe77sB/qU+GcJwT9QczizQCEK+biuAe
JbosFatSQG1TPsNnNIG268n674X+J+RTHX03uxnC2nHrWfV22km1Mo88ms01ODPHeSUXNR0JcofDb4a8UB7cWo6hh0TpDNjbAut
eT2KSejBvWIDquu6gD13HsP1brVCEfiFpH9p56TNVsCq2Wqh1M3XL1AGnscdmzb3rrCN/Gfseictxdt6bo762lmg7bgfX/efpX1D
YEapoaXSkLEagwuMowINzCY1I1rqUtR6GkCs9rF4VEe59NBENE= root@ids
-----END SSH HOST KEY KEYS-----
[ 20.626068] cloud-init[762]: Cloud-init v. 22.2-0ubuntu1~22.04.3 finished at Mon, 13 Feb 2023 10:
53:23 +0000. DataSource DataSourceNone. Up 20.61 seconds
[ 20.626673] cloud-init[762]: 2023-02-13 10:53:23,518 - cc_final_message.py[WARNING]: Used fallback
datasource
-
```

Figure 15 - Ubuntu Server first boot

Login and update your system.

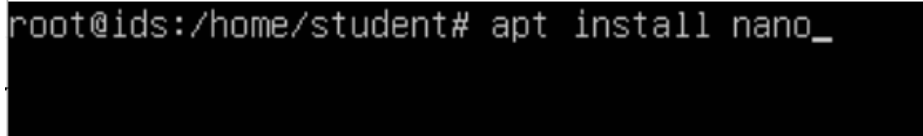
apt update

```
root@ids:/home/student# apt update
Hit:1 http://pt.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://pt.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://pt.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://pt.archive.ubuntu.com/ubuntu jammy-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
50 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ids:/home/student#
```

Figure 16 - Ubuntu server update

Install an editor.

apt install nano

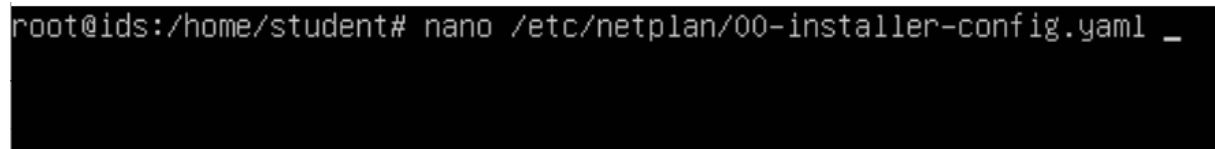


```
root@ids:/home/student# apt install nano_
```

Figure 17 - nano installation process

Edit your netplan network configuration.

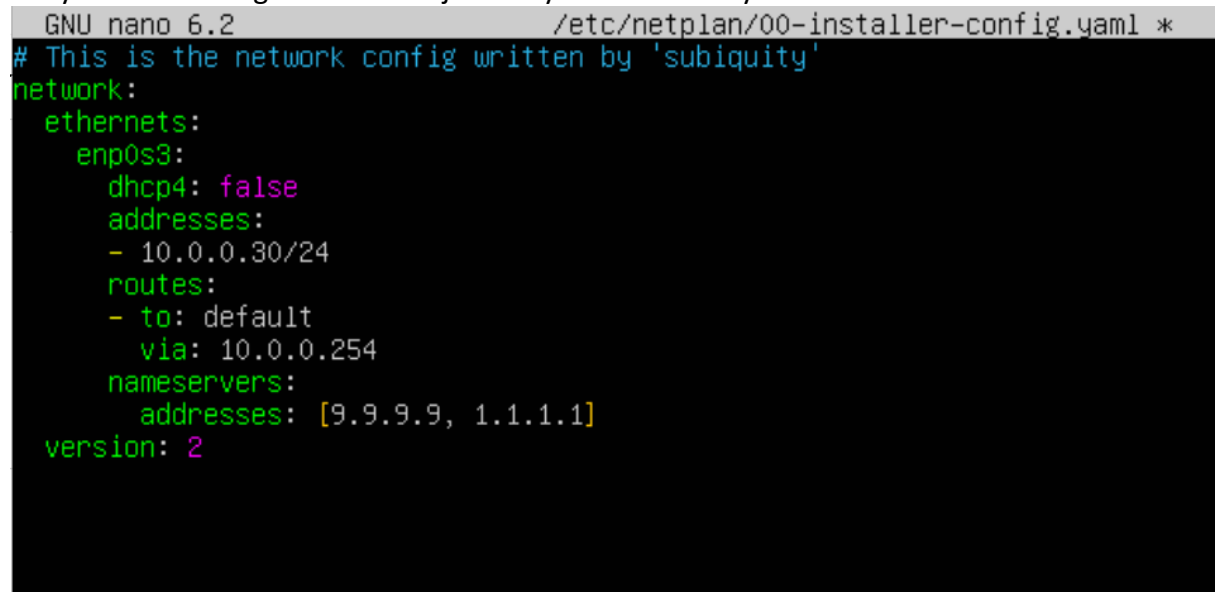
nano /etc/netplan/00-installer-config.yaml



```
root@ids:/home/student# nano /etc/netplan/00-installer-config.yaml _
```

Figure 18 - Edit network configuration file

Set your own configuration and adjust for your own reality.



```
GNU nano 6.2 /etc/netplan/00-installer-config.yaml *
# This is the network config written by 'subiquity'
network:
  ethernets:
    enp0s3:
      dhcp4: false
      addresses:
        - 10.0.0.30/24
      routes:
        - to: default
          via: 10.0.0.254
      nameservers:
        addresses: [9.9.9.9, 1.1.1.1]
  version: 2
```

Figure 19 - Network configuration for the current server

Save the document, exit and apply your configurations.

netplan try

netplan apply

```
root@ids:/home/student# netplan try
Do you want to keep these settings?

Press ENTER before the timeout to accept the new configuration

Changes will revert in 120 seconds
Configuration accepted.
root@ids:/home/student# netplan apply
root@ids:/home/student# _
```

Figure 20 - Apply network configurations

Using Putty from Windows, we'll configure the sensor for our network.

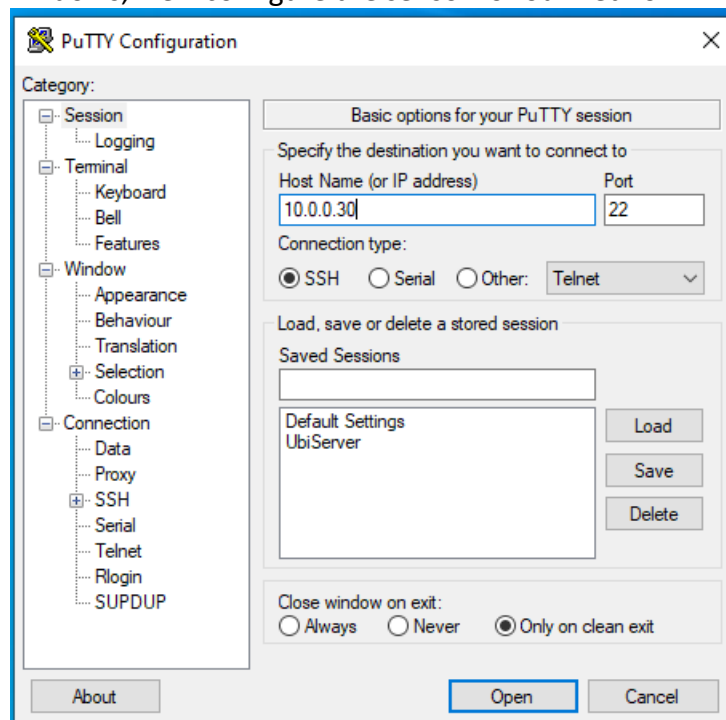


Figure 21 - Connect remotely to the Ubuntu Server

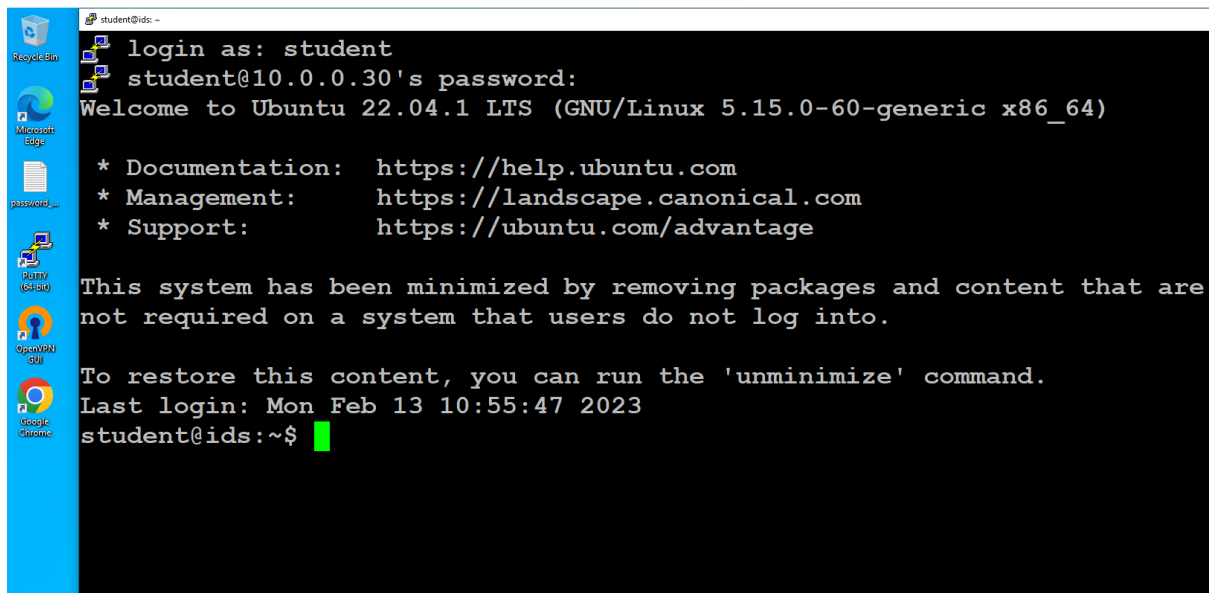


Figure 22 - Connected to Ubuntu Server

Lets install SNORT.

apt install snort

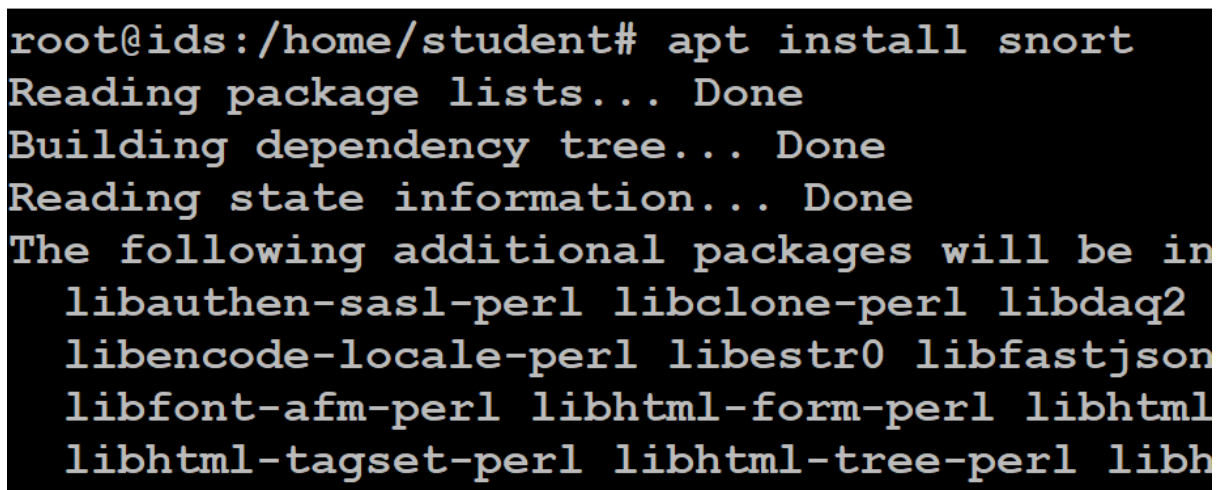


Figure 23 - Snort installation process

Setup the network CIDR you'll be monitoring.
You can setup multiple networks, by separating then using a comma.

```
Setting up snort (2.9.15.1-6build1) ...
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend cannot
be used. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 78.)
debconf: falling back to frontend: Readline
Snort configuration: interface default not set, using 'enp0s3'
Configuring snort
-----

Please use the CIDR form - for example, 192.168.1.0/24 for a block of 256
addresses or 192.168.1.42/32 for just one. Multiple values should be
comma-separated (without spaces).

You can leave this value empty and configure HOME_NET in /etc/snort/snort.conf
instead. This is useful if you are using Snort in a system which frequently
changes network and does not have a static IP address assigned.

Please note that if Snort is configured to use multiple interfaces, it will use
this value as the HOME_NET definition for all of them.

Address range for the local network: 10.0.0.0/24
Progress: [ 89%] [#####.....]
```

Figure 24 - Setting "HOME_NET" as 10.0.0.0 network

Press enter and let the installation finish.

You can validate your snort installation by using the test configuration.

snort -T -i enp0s3 -c /etc/snort/snort.conf

-T is used to test the configuration file

-i is used to define the network interface you want to use

-c is used to specify the configuration file

```
root@ids:/home/student# snort -T -i enp0s3 -c /etc/snort/snort.conf
```

Figure 25 - Test snort installation

By running this command, the output will provide information on how many rules have been imported, and if the configuration is OK or if it has any errors.

```

4057 Snort rules read
    3383 detection rules
    0 decoder rules
    0 preprocessor rules
3383 Option Chains linked into 932 Chain Headers
+++++

```

```

+-----[Rule Port Counts]-----+
|          tcp      udp      icmp      ip
|      src      151      18          0      0
|      dst     3306     126          0      0
|      any      383      48         52     22
|      nc        27        8         15     20
|      s+d       12        5          0      0
+-----+

```

```

+-----[detection-filter-config]-----+
| memory-cap : 1048576 bytes
+-----[detection-filter-rules]-----+
| none
+-----+

```

Figure 26 - Snort imported rules

```

Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>
Preprocessor Object: SF_SDF Version 1.1 <Build 1>
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>

Snort successfully validated the configuration!
Snort exiting
root@ids:/home/student# █

```

Figure 27 - Snort configuration status

If the configurations are OK, you are good to go.

Let's create a sample rule to detect ICMP on the network and see if we get alerts from snort.

To create a rule, you can edit the following file.

```

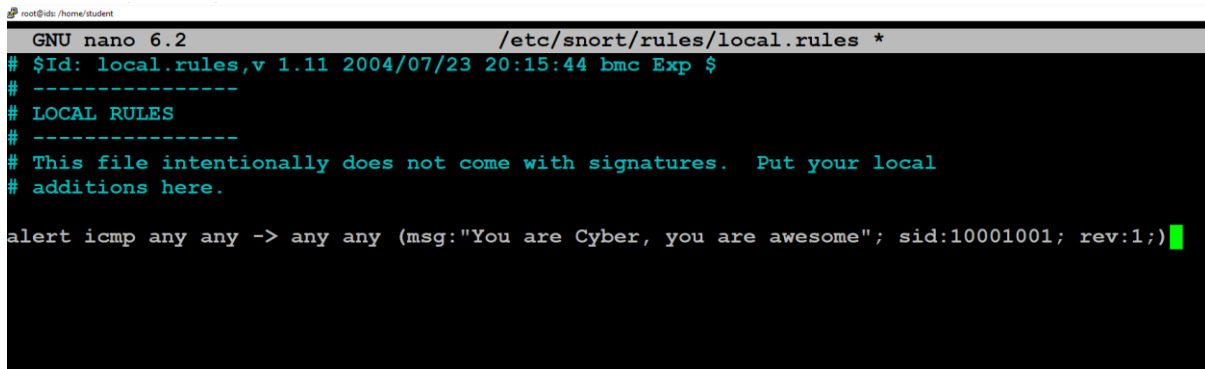
root@ids:/home/student# nano /etc/snort/rules/local.rules █

```

Figure 28 - Edit snort rules on local.rules files

This rule is ridiculous, but we'll make sure it is working:

```
alert icmp any any -> any any (msg:"You are Cyber, you are awesome"; sid:10001001; rev:1;)
```



```
GNU nano 6.2 /etc/snort/rules/local.rules *
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# -----
# LOCAL RULES
# -----
# This file intentionally does not come with signatures.  Put your local
# additions here.

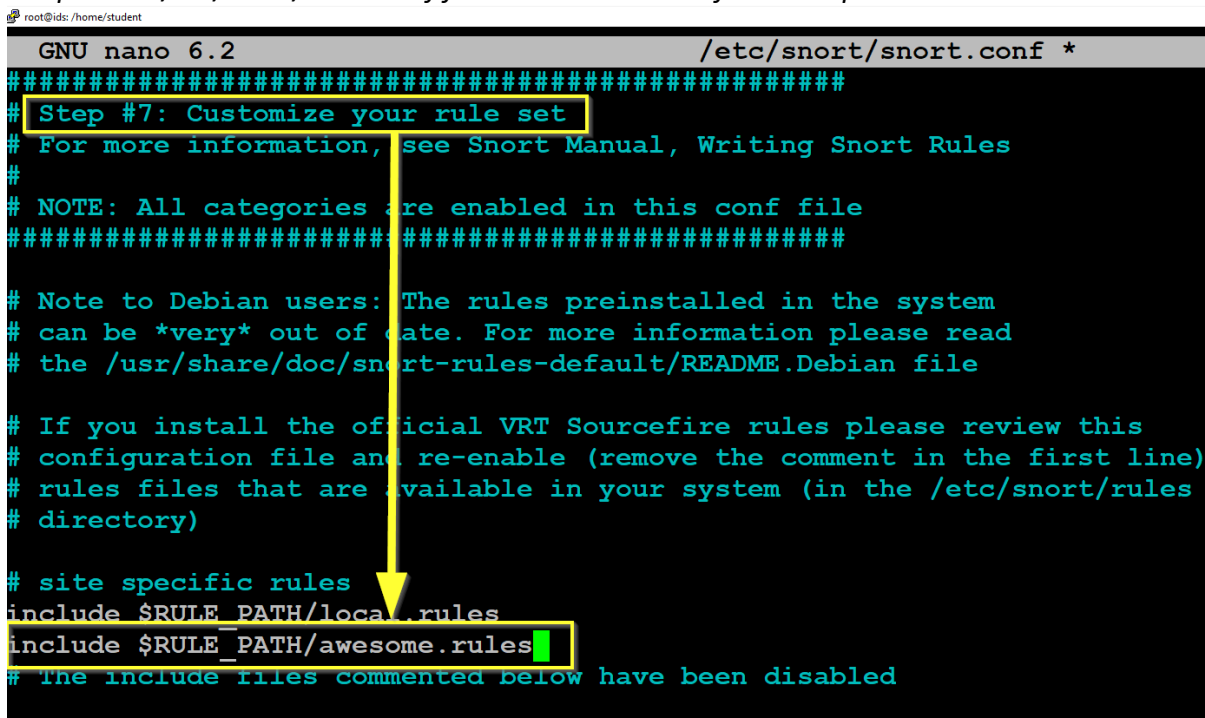
alert icmp any any -> any any (msg:"You are Cyber, you are awesome"; sid:10001001; rev:1;)
```

Figure 29 - Sample snort rules

Save the file and exit.

NOTE: it is possible to create other files to set your rules, but you'll need to import it in snort's configuration.

For example, if we create a file named "awesome.rules" at /etc/snort/rules/, we would need to open the /etc/snort/snort.conf file and add the new file on step 7.



```
GNU nano 6.2 /etc/snort/snort.conf *
#####
# Step #7: Customize your rule set
# For more information, see Snort Manual, Writing Snort Rules
#
# NOTE: All categories are enabled in this conf file
#####

# Note to Debian users: The rules preinstalled in the system
# can be *very* out of date. For more information please read
# the /usr/share/doc/snort-rules-default/README.Debian file

# If you install the official VRT Sourcefire rules please review this
# configuration file and re-enable (remove the comment in the first line)
# rules files that are available in your system (in the /etc/snort/rules
# directory)

# site specific rules
include $RULE_PATH/local.rules
include $RULE_PATH/awesome.rules
# The include files commented below have been disabled
```

Figure 30 – Line sample we needed to add if we had such a file with snort rules

Once you edit new rules, it is a good idea to test the configuration before running snort in a production detection mode.

```
Snort successfully validated the configuration!
Snort exiting
root@ids:/home/student#
```

Figure 31 - Snort rules validated

If everything is ok, we are ready to validate if the server detects our pings.
If you want to see the alerts in the terminal, let's run the alert mode as "console".
snort -A console -q -i enp0s3 -c /etc/snort/snort.conf

```
root@ids:/home/student# snort -A console -q -i enp0s3 -c /etc/snort/snort.conf
```

Let it run.

Now open a command line and try to ping any machine inside or outside your network.

NOTE: both machines are working with promiscuous network interfaces

```
root@ids:/home/student# snort -A console -q -i enp0s3 -c /etc/snort/snort.conf
02/13-11:34:43.523385  [**] [1:10001001:1] You are Cyber, you are awesome [**] [Priority: 0] {ICMP}
10.0.0.107 -> 8.8.8.8
02/13-11:34:43.538989  [**] [1:10001001:1] You are Cyber, you are awesome [**] [Priority: 0] {ICMP}
8.8.8.8 -> 10.0.0.107
02/13-11:34:44.560356  [**] [1:10001001:1] You are Cyber, you are awesome [**] [Priority: 0] {ICMP}
10.0.0.107 -> 8.8.8.8
02/13-11:34:44.592700  [**] [1:10001001:1] You are Cyber, you are awesome [**] [Priority: 0] {ICMP}
8.8.8.8 -> 10.0.0.107
02/13-11:34:45.598172  [**] [1:10001001:1] You are Cyber, you are awesome [**] [Priority: 0] {ICMP}
10.0.0.107 -> 8.8.8.8
02/13-11:34:45.613216  [**] [1:10001001:1] You are Cyber, you are awesome [**] [Priority: 0] {ICMP}
8.8.8.8 -> 10.0.0.107
02/13-11:34:46.610275  [**] [1:10001001:1] You are Cyber, you are awesome [**] [Priority: 0] {ICMP}
10.0.0.107 -> 8.8.8.8
02/13-11:34:46.625188  [**] [1:10001001:1] You are Cyber, you are awesome [**] [Priority: 0] {ICMP}
8.8.8.8 -> 10.0.0.107
```

```
C:\Users\admin>ping 8.8.8.8
Pinging 8.8.8.8 with 32 bytes of data:
Reply from 8.8.8.8: bytes=32 time=16ms TTL=57
Reply from 8.8.8.8: bytes=32 time=32ms TTL=57
Reply from 8.8.8.8: bytes=32 time=15ms TTL=57
Reply from 8.8.8.8: bytes=32 time=15ms TTL=57

Ping statistics for 8.8.8.8:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 15ms, Maximum = 32ms, Average = 19ms

C:\Users\admin>
```

Figure 32 - ICMP packets detected by snort

It is working, but if you want to leave it running and generating alerts to a log file, we need to leave snort setup in "Full" mode.

Run the command with the following changes. You can use "&" at the end of the command if you want to decouple the command process from your terminal and leave it running even if you close the terminal.

snort -A full -q -i enp0s3 -c /etc/snort/snort.conf &

```
root@ids:/home/student# snort -A full -q -i enp0s3 -c /etc/snort/snort.conf &
```

Figure 33 - Snort command

The alerts will not be shown in the terminal.

```
root@ids:/home/student# snort -A full -q -i enp0s3 -c /etc/snort/snort.conf &
[2] 800
root@ids:/home/student#
```

```
C:\Users\admin>ping 9.9.9.9

Pinging 9.9.9.9 with 32 bytes of data:
Reply from 9.9.9.9: bytes=32 time=21ms TTL=57
Reply from 9.9.9.9: bytes=32 time=8ms TTL=57
Reply from 9.9.9.9: bytes=32 time=11ms TTL=57
Reply from 9.9.9.9: bytes=32 time=8ms TTL=57

Ping statistics for 9.9.9.9:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 8ms, Maximum = 21ms, Average = 12ms

C:\Users\admin>
```

Figure 34 - Snort running in the background

We can now refer the the alerts log file.

tail -f /var/log/snort/alert

```
root@ids:/home/student# tail -f /var/log/snort/alert
02/13-11:41:37.111579 10.0.0.107 -> 9.9.9.9
ICMP TTL:128 TOS:0x0 ID:597 IpLen:20 DgmLen:60
Type:8 Code:0 ID:1 Seq:20 ECHO

[**] [1:10001001:1] You are Cyber, you are awesome [**]
[Priority: 0]
02/13-11:41:37.119655 9.9.9.9 -> 10.0.0.107
ICMP TTL:57 TOS:0x8 ID:44829 IpLen:20 DgmLen:60
Type:0 Code:0 ID:1 Seq:20 ECHO REPLY

[**] [1:10001001:1] You are Cyber, you are awesome [**]
[Priority: 0]
02/13-11:41:51.437783 10.0.0.107 -> 9.9.9.9
ICMP TTL:128 TOS:0x0 ID:598 IpLen:20 DgmLen:60
Type:8 Code:0 ID:1 Seq:21 ECHO

[**] [1:10001001:1] You are Cyber, you are awesome [**]
[Priority: 0]
02/13-11:41:51.454824 9.9.9.9 -> 10.0.0.107
ICMP TTL:57 TOS:0x8 ID:44916 IpLen:20 DgmLen:60
Type:0 Code:0 ID:1 Seq:21 ECHO REPLY

[**] [1:10001001:1] You are Cyber, you are awesome [**]
```

```
C:\Users\admin>ping 9.9.9.9

Pinging 9.9.9.9 with 32 bytes of data:
Reply from 9.9.9.9: bytes=32 time=9ms TTL=57
Reply from 9.9.9.9: bytes=32 time=17ms TTL=57
Reply from 9.9.9.9: bytes=32 time=8ms TTL=57
Reply from 9.9.9.9: bytes=32 time=8ms TTL=57

Ping statistics for 9.9.9.9:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 8ms, Maximum = 17ms, Average = 10ms

C:\Users\admin>ping 9.9.9.9

Pinging 9.9.9.9 with 32 bytes of data:
Reply from 9.9.9.9: bytes=32 time=17ms TTL=57
Reply from 9.9.9.9: bytes=32 time=10ms TTL=57
Reply from 9.9.9.9: bytes=32 time=9ms TTL=57
Reply from 9.9.9.9: bytes=32 time=33ms TTL=57

Ping statistics for 9.9.9.9:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 9ms, Maximum = 33ms, Average = 17ms

C:\Users\admin>
```

Figure 35 - Snort alerts in log file

Since you now have an alert log file, it'll be possible to send that information to a SIEM.

NOTE: Sometimes you'll be able to see files like the following picture. They are packet captures that can be opened in wireshark to analyse packets that triggered an alert.

```
root@ids:/home/student# cd /var/log/snort/
root@ids:/var/log/snort# ls
alert          snort.alert.fast  snort.log.1676287959  snort.log.1676288346
snort.alert    snort.log         snort.log.1676288079
root@ids:/var/log/snort#
```

Figure 36 - Snort packet captures from generated alerts