Incus Multi-Tenancy

A Beginner's Guide

Ananda Kammampati

“ The Greatest Teacher : One who makes you wanting to learn “

Dedicated to

My classmates Kannan Ramasubramanian and Rajesh Kunnath who are my favorite teachers and my source of inspiration.

Table of Contents

Incus Multi-Tenancy: A Beginner's Guide

Introduction

The primary audience of this book are users who have some working knowledge on incus virtualization platform and would like to understand better on how to configure incus into a multi-tenant platform. The goal is to guide the reader in leveraging incus capabilities and features namely projects, open virtual networking, access control lists and many more to gain an overall understanding in building a minimal multi-tenant system on a single server.

This book explores only a bare minimal design needed for multi-tenancy with regards to storage and networking to gain some insights. This book is not meant to teach the basics of incus virtualization platform. Please refer to the credits section at the end of this book that provide links to the best teachers on incus.

For a complete beginner to incus virtualization platform, I would recommend to start with Trevor Sullivan's YouTube playlist on LXD containers.

The approach taken in this book is very Hands-on. It explains incus multi-tenancy by breaking it down not as chapters but a list of tasks, each task dependent on the completion of its previous task.

The recommended way to learn from this book is to use it as a Cookbook. By having a dedicated server and trying out all the tasks in the same order.

Happy learning :-)

- Ananda Kammampati

Task #01: Hardware Specifications of Incus Server

The mother board is from Supermicro (X10SDV-6C-TLN4F)

CPU	Intel XEON processor D-1528, 6-Core, 12 Threads
Memory	128GB ECC RDIMM DDR4 2133MHz
NICs	2 x 10GbE ports and 2 x 1GbE ports

Refer :
https://www.supermicro.com/en/products/motherboard/x10sdv-tln4f

Task #02: Pre-requisites

Here are the list of things that we are going to accomplish in this task.

1. Checking system details
2. Verifying bootable USB disk partition
3. Installation of ZFS packages
4. Removal of snap package manager
5. Adding two NVMe SSDs
6. Installation of docker (optional)

1. Checking system details

Making a bootable USB thumb drive is beyond the scope of this cookbook (there are tons of awesome online tutorials). As a pre-requisite, we have already made a 512GB USB thumb drive as bootable disk, booted the system and completed the installation.

```
# uname -a
Linux red-handsonbox 6.8.0-55-generic #57-Ubuntu SMP PREEMPT_DYNAMIC Wed Feb 12 23:42:21
UTC 2025 x86_64 x86_64 x86_64 GNU/Linux

# cat /etc/os-release
PRETTY_NAME="Ubuntu 24.04.2 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04.2 LTS (Noble Numbat)"
VERSION_CODENAME=noble
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=noble
```

2. Verifying bootable USB disk partition

Let us make sure that system boots fine from the USB thumb drive and the partitions are intact

```
# lsblk
NAME       MAJ:MIN RM   SIZE     RO TYPE  MOUNTPOINTS
sda              8:0     1     460.3G   0    disk
```

```
├─sda1      8:1     1     1M        0    part
└─sda2      8:2     1     460.3G   0    part   /

# df -h
Filesystem     Size  Used Avail Use% Mounted on
tmpfs            13G  1.6M   13G   1% /run
/dev/sda2      452G   11G  419G   3% /
tmpfs            63G     0   63G   0% /dev/shm
tmpfs            5.0M     0  5.0M   0% /run/lock
tmpfs           13G  8.0K   13G   1% /run/user/0
```

3. Installing ZFS

Compared to other Linux distributions, installation of ZFS seems pretty straight forward on Ubuntu. The commands below are self-explanatory

```
# pwd
/root

# apt update -y

# apt upgrade -y

# apt install zfsutils-linux -y

# modprobe zfs

# zfs --version
zfs-2.2.2-0ubuntu9.1
zfs-kmod-2.2.2-0ubuntu9.1

# zpool --version
zfs-2.2.2-0ubuntu9.1
zfs-kmod-2.2.2-0ubuntu9.1

# modinfo zfs
filename:      /lib/modules/6.8.0-55-generic/kernel/zfs/zfs.ko.zst
version:       2.2.2-0ubuntu9.1
license:       CDDL
license:       Dual BSD/GPL
license:       Dual MIT/GPL
author:        OpenZFS
description:    ZFS
```

```
alias:          zzstd
alias:          zcommon
alias:          zunicode
alias:          znvpair
alias:          zlua
alias:          icp
alias:          zavl
alias:          devname:zfs
alias:          char-major-10-249
srcversion:     307E3A0219DD7E837933412
depends:        spl
retpoline:      Y
name:           zfs
vermagic:       6.8.0-55-generic SMP preempt mod_unload modversions
sig_id:         PKCS#7
signer:         Build time autogenerated kernel key
.....
.....
.....
```

4. Removing snap package manager

I still haven't taken the time to learn. So purging it for now ;-)

```
# apt purge snapd

# apt-mark hold snapd

# rm -rf /var/cache/snapd/

# rm -rf ~/snap

# sync ; reboot
```

5. Adding two NVMe SSD's

The reason for opting 2 NVMe SSDs and the reason behind creating different paritions of equal size needs some detailed explaination, which we will look into in the following tasks. Command 'lsblk' shows the details of 3 storge devices. 'sda' is the bootable USB thumb drive. 'nvme0n1' and 'nvme1n1' are the two NVMe SSD disks of size 2GB and 1GB respectively..

```
# lsblk
NAME       MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda          8:0    1 460.3G  0 disk
├─sda1       8:1    1    1M  0 part
└─sda2       8:2    1 460.3G  0 part /
nvme1n1    259:0    0 931.5G  0 disk
├─nvme1n1p1 259:1    0   155G  0 part
├─nvme1n1p2 259:2    0   155G  0 part
├─nvme1n1p3 259:3    0   155G  0 part
├─nvme1n1p4 259:4    0   155G  0 part
├─nvme1n1p5 259:5    0   155G  0 part
├─nvme1n1p6 259:6    0   155G  0 part
└─nvme1n1p7 259:7    0   1.5G  0 part
nvme0n1    259:8    0   1.9T  0 disk
├─nvme0n1p1 259:9    0   317G  0 part
├─nvme0n1p2 259:10   0   317G  0 part
├─nvme0n1p3 259:11   0   317G  0 part
├─nvme0n1p4 259:12   0   317G  0 part
├─nvme0n1p5 259:13   0   317G  0 part
├─nvme0n1p6 259:14   0   317G  0 part
└─nvme0n1p7 259:15   0   5.7G  0 part
```

Installation of docker

Installation of docker is purely optional at this stage. But since we are with the installation task, we might as well take care of it. The bash shell script shown below has all the required commands to fully install docker engine and start it as a service.

```
# cat install-docker.sh
-----
#!/bin/bash
clear
echo " "
echo "update...."
echo " "
apt -y update ;
echo " "
echo "upgrade...."
echo " "
apt -y upgrade ;
echo " "
echo "install apt-transport-https ca-certificates curl software-properties-common..."
```

```
echo " "
apt install apt-transport-https ca-certificates curl software-properties-common -y ;
echo " "
echo "install key..."
echo " "
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | apt-key add - ;
echo " "
echo "add repository..."
echo " "
add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable" ;
echo " "
echo "install docker-ce..."
echo " "
apt update -y ; apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin
docker-compose-plugin -y ;
echo " "
echo "install docker-compose..."
echo " "
curl -L
https://github.com/docker/compose/releases/download/v2.32.4/docker-compose-$(uname
-s)-$(uname -m) -o /usr/local/bin/docker-compose ;
chmod +x /usr/local/bin/docker-compose ;
echo " "
docker-compose --version ;
echo " "
echo "start docker..."
echo " "
systemctl start docker ;
echo " "
echo "enable docker..."
echo " "
systemctl enable docker ;
echo " "
echo "check docker..."
echo " "
systemctl status -l docker ;
-----

# chmod 755 install-docker.sh

# ./install-docker.sh
```

Task #03: Partitioning of NVMe SSDs

Let us break down this task into following steps:

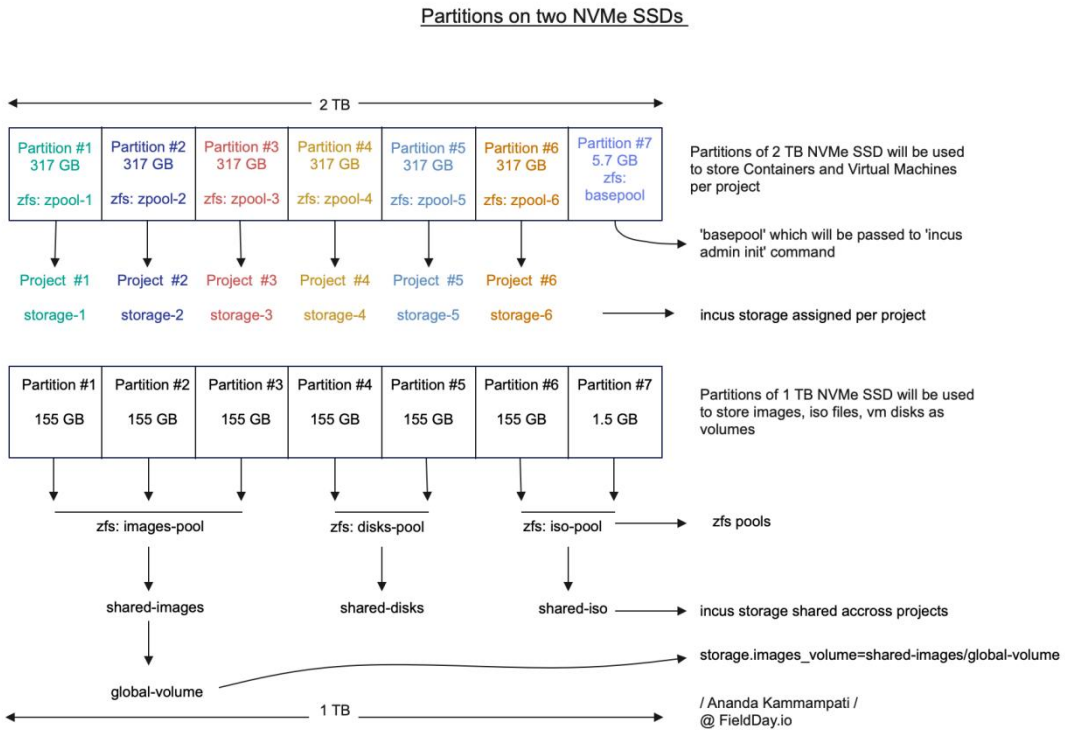
1. Understanding the anotomy of paritions
2. Partitioning of the 2TB SSD
3. Partitioning of the 1TB SSD

1. Understanding the anotomy of Partitions

In this task we will focus on creating multiple partitions of equal sizes on both 1TB and 2TB NVMe SSDs. In incus, Multi-tenancy is achieved by leveraging the contruct/artifact PROJECT. The term 'Project' and 'Tenant' can be used synonymously as they both refer the same idea. In simple terms:

" Project is a method of resource isolation to attain Multi-Tenancy ".

Our idea behind creating multiple partitions is to allocate each partitions to individual Projects. Later we will name the projects as 'Handson-lab-1', 'Handson-lab-2' and so on. Our plan is to have 6 Projects/Tenents on our system. So we will create 6 paritions on each SSDs. Later when we create individual Projects, we will allocate these individual paritions to each Project. For now we will focus on creation of Partitions.



The above diagram shows the anatomy of how we are planning to create 6 partitions on each

SSD. A single partition on the 2TB SSD will be allocated to a single Project. For example Partition #1 on the 2TB SSD will be allocated to Project #1. And this proces will be repeated to all the subsequent Projects.

Note that the Partition #7 on the 2TB SSD is what we use while initializing incus with 'incus admin init' command. This will be clear later as we progress.

We will use the native tool 'gdisk' on Ubuntu24.04 which will be the primary Host OS running on our Supermicro server. To begin with let us focus on the 2TB NVMe SSD which is /dev/nvme0n1 on our system.

Explaning how to use tool gdisk is beyond th scope of this book (refer to online videos that explain better on using the tool).

We will create 6 equal sized partitions (317 GB each) on the 2TB SSD. With the remaining 5.7 GB space, we will create a separate partition. It is this Partition #7 we name it as 'basepool' and will use it to configure incus with the 'incus admin init' command.

2. Partitioning of the 2TB SSD

Let us begin configuring the 2TB SSD with gdisk tool.

```
# gdisk /dev/nvme0n1
GPT fdisk (gdisk) version 1.0.9

Partition table scan:
  MBR: not present
  BSD: not present
  APM: not present
  GPT: not present

Creating new GPT entries in memory.

Command (? for help): n
Partition number (1-128, default 1):
First sector (34-4000797326, default = 2048) or {+-}size{KMGTP}:
Last sector (2048-4000797326, default = 4000796671) or {+-}size{KMGTP}: +317G
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): n
Partition number (2-128, default 2):
First sector (34-4000797326, default = 664799232) or {+-}size{KMGTP}:
```

```
Last sector (664799232-4000797326, default = 4000796671) or {+-}size{KMGTP}: +317G
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): n
Partition number (3-128, default 3):
First sector (34-4000797326, default = 1329596416) or {+-}size{KMGTP}:
Last sector (1329596416-4000797326, default = 4000796671) or {+-}size{KMGTP}: +317G
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): n
Partition number (4-128, default 4):
First sector (34-4000797326, default = 1994393600) or {+-}size{KMGTP}:
Last sector (1994393600-4000797326, default = 4000796671) or {+-}size{KMGTP}: +317G
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): n
Partition number (5-128, default 5):
First sector (34-4000797326, default = 2659190784) or {+-}size{KMGTP}:
Last sector (2659190784-4000797326, default = 4000796671) or {+-}size{KMGTP}: +317G
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): n
Partition number (6-128, default 6):
First sector (34-4000797326, default = 3323987968) or {+-}size{KMGTP}:
Last sector (3323987968-4000797326, default = 4000796671) or {+-}size{KMGTP}: +317G
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): p
Disk /dev/nvme0n1: 4000797360 sectors, 1.9 TiB
Model: TEAM TM8FP6002T
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): A7E37AAC-57CC-4D4F-B995-0919B01794EA
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
```

```
First usable sector is 34, last usable sector is 4000797326
Partitions will be aligned on 2048-sector boundaries
Total free space is 12014189 sectors (5.7 GiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048      664799231   317.0 GiB   8300  Linux filesystem
   2       664799232     1329596415   317.0 GiB   8300  Linux filesystem
   3      1329596416     1994393599   317.0 GiB   8300  Linux filesystem
   4      1994393600     2659190783   317.0 GiB   8300  Linux filesystem
   5      2659190784     3323987967   317.0 GiB   8300  Linux filesystem
   6      3323987968     3988785151   317.0 GiB   8300  Linux filesystem

Command (? for help): n
Partition number (7-128, default 7):
First sector (34-4000797326, default = 3988785152) or {+-}size{KMGTP}:
Last sector (3988785152-4000797326, default = 4000796671) or {+-}size{KMGTP}:
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): p
Disk /dev/nvme0n1: 4000797360 sectors, 1.9 TiB
Model: TEAM TM8FP6002T
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): A7E37AAC-57CC-4D4F-B995-0919B01794EA
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 4000797326
Partitions will be aligned on 2048-sector boundaries
Total free space is 2669 sectors (1.3 MiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048      664799231   317.0 GiB   8300  Linux filesystem
   2       664799232     1329596415   317.0 GiB   8300  Linux filesystem
   3      1329596416     1994393599   317.0 GiB   8300  Linux filesystem
   4      1994393600     2659190783   317.0 GiB   8300  Linux filesystem
   5      2659190784     3323987967   317.0 GiB   8300  Linux filesystem
   6      3323987968     3988785151   317.0 GiB   8300  Linux filesystem
   7      3988785152     4000796671   5.7 GiB     8300  Linux filesystem

Command (? for help): w

Final checks complete. About to write GPT data. THIS WILL OVERWRITE EXISTING
PARTITIONS!!
```

```
Do you want to proceed? (Y/N): Y
OK; writing new GUID partition table (GPT) to /dev/nvme0n1.
The operation has completed successfully.

# gdisk -l /dev/nvme0n1
GPT fdisk (gdisk) version 1.0.9

Partition table scan:
  MBR: protective
  BSD: not present
  APM: not present
  GPT: present

Found valid GPT with protective MBR; using GPT.
Disk /dev/nvme0n1: 4000797360 sectors, 1.9 TiB
Model: TEAM TM8FP6002T
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): A7E37AAC-57CC-4D4F-B995-0919B01794EA
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 4000797326
Partitions will be aligned on 2048-sector boundaries
Total free space is 2669 sectors (1.3 MiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048       664799231   317.0 GiB   8300  Linux filesystem
   2       664799232      1329596415   317.0 GiB   8300  Linux filesystem
   3      1329596416      1994393599   317.0 GiB   8300  Linux filesystem
   4      1994393600      2659190783   317.0 GiB   8300  Linux filesystem
   5      2659190784      3323987967   317.0 GiB   8300  Linux filesystem
   6      3323987968      3988785151   317.0 GiB   8300  Linux filesystem
   7      3988785152      4000796671   5.7 GiB     8300  Linux filesystem

# fdisk -l
Disk /dev/nvme0n1: 1.86 TiB, 2048408248320 bytes, 4000797360 sectors
Disk model: TEAM TM8FP6002T
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: A7E37AAC-57CC-4D4F-B995-0919B01794EA

Device            Start       End   Sectors  Size Type
```

```
/dev/nvme0n1p1      2048  664799231 664797184  317G Linux filesystem
/dev/nvme0n1p2  664799232 1329596415 664797184  317G Linux filesystem
/dev/nvme0n1p3 1329596416 1994393599 664797184  317G Linux filesystem
/dev/nvme0n1p4 1994393600 2659190783 664797184  317G Linux filesystem
/dev/nvme0n1p5 2659190784 3323987967 664797184  317G Linux filesystem
/dev/nvme0n1p6 3323987968 3988785151 664797184  317G Linux filesystem
/dev/nvme0n1p7 3988785152 4000796671  12011520  5.7G Linux filesystem
```

Now that we have completed creating 7 partitions on 2TB SSD, let us move onto the 1TB SSD and repeat the same procedure. The size of each parition will be 155 GB. ie, 1 TB SSD divided into 6 equal partition size. We will create a separate partition with the remaining 1.5 GB space which is Partition #7.

3. Partitioning of the 1TB SSD

```
# gdisk /dev/nvme1n1
GPT fdisk (gdisk) version 1.0.9

Partition table scan:
  MBR: protective
  BSD: not present
  APM: not present
  GPT: present

Found valid GPT with protective MBR; using GPT.

Command (? for help): n
Partition number (1-128, default 1):
First sector (34-1953525134, default = 2048) or {+-}size{KMGTP}:
Last sector (2048-1953525134, default = 1953523711) or {+-}size{KMGTP}: +155G
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): n
Partition number (2-128, default 2):
First sector (34-1953525134, default = 325060608) or {+-}size{KMGTP}:
Last sector (325060608-1953525134, default = 1953523711) or {+-}size{KMGTP}: +155G
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): n
Partition number (3-128, default 3):
```

```
First sector (34-1953525134, default = 650119168) or {+-}size{KMGTP}:
Last sector (650119168-1953525134, default = 1953523711) or {+-}size{KMGTP}: +155G
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): n
Partition number (4-128, default 4):
First sector (34-1953525134, default = 975177728) or {+-}size{KMGTP}:
Last sector (975177728-1953525134, default = 1953523711) or {+-}size{KMGTP}: +155G
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): n
Partition number (5-128, default 5):
First sector (34-1953525134, default = 1300236288) or {+-}size{KMGTP}:
Last sector (1300236288-1953525134, default = 1953523711) or {+-}size{KMGTP}: +155G
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): n
Partition number (6-128, default 6):
First sector (34-1953525134, default = 1625294848) or {+-}size{KMGTP}:
Last sector (1625294848-1953525134, default = 1953523711) or {+-}size{KMGTP}: +155G
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): p
Disk /dev/nvme1n1: 1953525168 sectors, 931.5 GiB
Model: WDC WDS100T2B0C-00PXH0
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): D5557850-4398-F748-8720-008C4186AEAC
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 1953525134
Partitions will be aligned on 2048-sector boundaries
Total free space is 3173741 sectors (1.5 GiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048      325060607   155.0 GiB   8300  Linux filesystem
   2       325060608      650119167   155.0 GiB   8300  Linux filesystem
```

```
   3      650119168      975177727   155.0 GiB   8300  Linux filesystem
   4      975177728     1300236287   155.0 GiB   8300  Linux filesystem
   5     1300236288     1625294847   155.0 GiB   8300  Linux filesystem
   6     1625294848     1950353407   155.0 GiB   8300  Linux filesystem

Command (? for help): n
Partition number (7-128, default 7):
First sector (34-1953525134, default = 1950353408) or {+-}size{KMGTP}:
Last sector (1950353408-1953525134, default = 1953523711) or {+-}size{KMGTP}:
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'

Command (? for help): p
Disk /dev/nvme1n1: 1953525168 sectors, 931.5 GiB
Model: WDC WDS100T2B0C-00PXH0
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): D5557850-4398-F748-8720-008C4186AEAC
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 1953525134
Partitions will be aligned on 2048-sector boundaries
Total free space is 3437 sectors (1.7 MiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048      325060607   155.0 GiB   8300  Linux filesystem
   2      325060608      650119167   155.0 GiB   8300  Linux filesystem
   3      650119168      975177727   155.0 GiB   8300  Linux filesystem
   4      975177728     1300236287   155.0 GiB   8300  Linux filesystem
   5     1300236288     1625294847   155.0 GiB   8300  Linux filesystem
   6     1625294848     1950353407   155.0 GiB   8300  Linux filesystem
   7     1950353408     1953523711   1.5 GiB     8300  Linux filesystem

Command (? for help): w

Final checks complete. About to write GPT data. THIS WILL OVERWRITE EXISTING
PARTITIONS!!

Do you want to proceed? (Y/N): Y
OK; writing new GUID partition table (GPT) to /dev/nvme1n1.
The operation has completed successfully.
root@red-handsonbox:~#
root@red-handsonbox:~# gdisk -l /dev/nvme1n1
GPT fdisk (gdisk) version 1.0.9
```

```
Partition table scan:
  MBR: protective
  BSD: not present
  APM: not present
  GPT: present

Found valid GPT with protective MBR; using GPT.
Disk /dev/nvme1n1: 1953525168 sectors, 931.5 GiB
Model: WDC WDS100T2B0C-00PXH0
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): D5557850-4398-F748-8720-008C4186AEAC
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 1953525134
Partitions will be aligned on 2048-sector boundaries
Total free space is 3437 sectors (1.7 MiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048      325060607   155.0 GiB   8300  Linux filesystem
   2       325060608       650119167   155.0 GiB   8300  Linux filesystem
   3       650119168       975177727   155.0 GiB   8300  Linux filesystem
   4       975177728      1300236287   155.0 GiB   8300  Linux filesystem
   5      1300236288      1625294847   155.0 GiB   8300  Linux filesystem
   6      1625294848      1950353407   155.0 GiB   8300  Linux filesystem
   7      1950353408      1953523711   1.5 GiB     8300  Linux filesystem

# fdisk -l /dev/nvme1n1
Disk /dev/nvme1n1: 931.51 GiB, 1000204886016 bytes, 1953525168 sectors
Disk model: WDC WDS100T2B0C-00PXH0
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: D5557850-4398-F748-8720-008C4186AEAC

Device            Start       End   Sectors  Size Type
/dev/nvme1n1p1      2048  325060607 325058560  155G Linux filesystem
/dev/nvme1n1p2  325060608  650119167 325058560  155G Linux filesystem
/dev/nvme1n1p3  650119168  975177727 325058560  155G Linux filesystem
/dev/nvme1n1p4  975177728 1300236287 325058560  155G Linux filesystem
/dev/nvme1n1p5 1300236288 1625294847 325058560  155G Linux filesystem
/dev/nvme1n1p6 1625294848 1950353407 325058560  155G Linux filesystem
/dev/nvme1n1p7 1950353408 1953523711   3170304  1.5G Linux filesystem
```

Task #04: Creation of ZFS pools

In this task we will focus on creating ZFS pools with the parttions that we have already created with the previous task. Let us break down the procedures into following steps.

1. Verification of zfs
2. Creation of zfs pool per partition on 2TB SSD
3. Creation of zfs pool per partition on 1TB SSD

Please note that the technical know-how on how ZFS filesystem work is beyond the scope of this book. We will primarily focus on using it. ie, creating mutiple ZFS storage pools so that thet each can be assigned to different incus projects

1. Verification of zfs

The commands we use to manage zfs filesystem are 'zpool' and 'zfs'. But first we need to verify that the zfs kernel module is indeed loaded and this step is very crucial. Without this kernel module loaded, zfs will not work, even if zfs related that commands are installed. Luciky, the latest versions of Ubuntu (24.04 as of this writing) come with ZFS support by default which makes life much easier.

If the outout of 'lsmod | grep zfs' doesn't list anything, then we need to resolve loading zfs kernel module first before processing further. If one is not planning on using ZFS, it is still possible to build a Multi-tenancy incus system with other filesystems (example ext4, btrfs). But in our case, we will primariliy focus on using ZFS.

```
# lsmod | grep zfs
zfs  6602752  6
spl 180224  1 zfs

# zfs --version
zfs-2.2.2-0ubuntu9.2
zfs-kmod-2.2.2-0ubuntu9.1

# zpool --version
zfs-2.2.2-0ubuntu9.2
zfs-kmod-2.2.2-0ubuntu9.1
```

2. Creation of zfs pool per partition on 2TB SSD

The command 'lsblk' lists all the partitions that we have created so far on both the SSDs.

Let us take a look at all the paritions of all the disks, including the bootable USD disk

```
# fdisk -l
Disk /dev/nvme0n1: 1.86 TiB, 2048408248320 bytes, 4000797360 sectors
Disk model: TEAM TM8FP6002T
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: A7E37AAC-57CC-4D4F-B995-0919B01794EA

Device            Start       End   Sectors  Size Type
/dev/nvme0n1p1      2048  664799231 664797184  317G Linux filesystem
/dev/nvme0n1p2  664799232 1329596415 664797184  317G Linux filesystem
/dev/nvme0n1p3 1329596416 1994393599 664797184  317G Linux filesystem
/dev/nvme0n1p4 1994393600 2659190783 664797184  317G Linux filesystem
/dev/nvme0n1p5 2659190784 3323987967 664797184  317G Linux filesystem
/dev/nvme0n1p6 3323987968 3988785151 664797184  317G Linux filesystem
/dev/nvme0n1p7 3988785152 4000796671  12011520  5.7G Linux filesystem

Disk /dev/nvme1n1: 931.51 GiB, 1000204886016 bytes, 1953525168 sectors
Disk model: WDC WDS100T2B0C-00PXH0
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: D5557850-4398-F748-8720-008C4186AEAC

Device            Start       End   Sectors  Size Type
/dev/nvme1n1p1      2048  325060607 325058560  155G Linux filesystem
/dev/nvme1n1p2  325060608  650119167 325058560  155G Linux filesystem
/dev/nvme1n1p3  650119168  975177727 325058560  155G Linux filesystem
/dev/nvme1n1p4  975177728 1300236287 325058560  155G Linux filesystem
/dev/nvme1n1p5 1300236288 1625294847 325058560  155G Linux filesystem
/dev/nvme1n1p6 1625294848 1950353407 325058560  155G Linux filesystem
/dev/nvme1n1p7 1950353408 1953523711   3170304  1.5G Linux filesystem

Disk /dev/sda: 460.27 GiB, 494206451712 bytes, 965246976 sectors
Disk model:  SanDisk 3.2Gen1
```

```
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 1DDC2624-2581-4560-AE31-44AAC5E3A796

Device     Start       End   Sectors   Size Type
/dev/sda1   2048      4095      2048    1M BIOS boot
/dev/sda2   4096 965244927 965240832 460.3G Linux filesystem
```

Let us understand how we are planning on creating mutiple zfs pools on each partitions. We will start with the 2 TB SSD first

```
# lsblk
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda           8:0    1 460.3G  0 disk
├─sda1        8:1    1     1M  0 part
└─sda2        8:2    1 460.3G  0 part /  ---> Root on USB thumb drive
nvme0n1     259:0    0   1.9T  0 disk   ----> 2 TB NVMe SSD
├─nvme0n1p1 259:1    0   317G  0 part
├─nvme0n1p2 259:2    0   317G  0 part
├─nvme0n1p3 259:3    0   317G  0 part
├─nvme0n1p4 259:4    0   317G  0 part
├─nvme0n1p5 259:5    0   317G  0 part
├─nvme0n1p6 259:6    0   317G  0 part
└─nvme0n1p7 259:7    0   5.7G  0 part
nvme1n1     259:8    0 931.5G  0 disk   ---> 1 TB NVMe SSD
├─nvme1n1p1 259:9    0   155G  0 part
├─nvme1n1p2 259:10   0   155G  0 part
├─nvme1n1p3 259:11   0   155G  0 part
├─nvme1n1p4 259:12   0   155G  0 part
├─nvme1n1p5 259:13   0   155G  0 part
├─nvme1n1p6 259:14   0   155G  0 part
└─nvme1n1p7 259:15   0   1.5G  0 part
```

```
# lsblk
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda           8:0    1 460.3G  0 disk
├─sda1        8:1    1     1M  0 part
└─sda2        8:2    1 460.3G  0 part /  --> Root partition on USB thumb drive
nvme0n1     259:0    0   1.9T  0 disk   --> 2 TB NVMe SSD
├─nvme0n1p1 259:1    0   317G  0 part   --> Create zfs zpool-1
├─nvme0n1p2 259:2    0   317G  0 part   --> Create zfs zpool-2
├─nvme0n1p3 259:3    0   317G  0 part   --> Create zfs zpool-3
```

```
├─nvme0n1p4 259:4    0   317G  0 part   --> Create zfs zpool-4
├─nvme0n1p5 259:5    0   317G  0 part   --> Create zfs zpool-5
├─nvme0n1p6 259:6    0   317G  0 part   --> Create zfs zpool-6
└─nvme0n1p7 259:7    0   5.7G  0 part   --> Create zfs basebpool
nvme1n1     259:8    0 931.5G  0 disk   --> 1 TB NVMe SSD
├─nvme1n1p1 259:9    0   155G  0 part
├─nvme1n1p2 259:10   0   155G  0 part
├─nvme1n1p3 259:11   0   155G  0 part
├─nvme1n1p4 259:12   0   155G  0 part
├─nvme1n1p5 259:13   0   155G  0 part
├─nvme1n1p6 259:14   0   155G  0 part
└─nvme1n1p7 259:15   0   1.5G  0 part
```

Once again, we will take a look at the output of 'lsblk' command and see how we are planning on creating multiple stoage pools and how each are assigned to/consumed by individual projects.

```
# lsblk
NAME       MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda          8:0    1 460.3G  0 disk
├─sda1       8:1    1     1M  0 part
└─sda2       8:2    1 460.3G  0 part /
nvme0n1    259:0    0   1.9T  0 disk
├─nvme0n1p1 259:1    0   317G  0 part  --> Assign to Project-1
├─nvme0n1p2 259:2    0   317G  0 part  --> Assign to Project-2
├─nvme0n1p3 259:3    0   317G  0 part  --> Assign to Project-3
├─nvme0n1p4 259:4    0   317G  0 part  --> Assign to Project-4
├─nvme0n1p5 259:5    0   317G  0 part  --> Assign to Project-5
├─nvme0n1p6 259:6    0   317G  0 part  --> Assign to Project-6
└─nvme0n1p7 259:7    0   5.7G  0 part  --> Pass to 'incus admin init'
nvme1n1    259:8    0 931.5G  0 disk
├─nvme1n1p1 259:9    0   155G  0 part
├─nvme1n1p2 259:10   0   155G  0 part
├─nvme1n1p3 259:11   0   155G  0 part
├─nvme1n1p4 259:12   0   155G  0 part
├─nvme1n1p5 259:13   0   155G  0 part
├─nvme1n1p6 259:14   0   155G  0 part
└─nvme1n1p7 259:15   0   1.5G  0 part
```

Now that we have a better understanding on what we plan on doing with seperate partitions, let us go ahead and run the zfs commands to create seperate zfs pools on each partition.

```
# zpool create zpool-1 /dev/nvme0n1p1
# zpool create zpool-2 /dev/nvme0n1p2
```

```
# zpool create zpool-3 /dev/nvme0n1p3
# zpool create zpool-4 /dev/nvme0n1p4
# zpool create zpool-5 /dev/nvme0n1p5
# zpool create zpool-6 /dev/nvme0n1p6
# zpool create basepool /dev/nvme0n1p7
```

As simple as that.

```
# zpool list
NAME      SIZE  ALLOC   FREE  CKPOINT  EXPANDSZ   FRAG    CAP  DEDUP    HEALTH  ALTROOT
basepool  5.50G   132K  5.50G        -         -     0%     0%  1.00x    ONLINE  -
zpool-1    316G   110K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-2    316G   108K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-3    316G   110K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-4    316G   110K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-5    316G   110K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-6    316G   110K   316G        -         -     0%     0%  1.00x    ONLINE  -
```

3. Creation of zfs pools on 1TB SSD

Let us repeat a similar exercise but with a slight twist with regards to the 1TB SSD

- Let us create a zfs pool comprising of partition 1, 2, 3 and name it as ‘images-pool’
- Let us create a zfs pool comprising of partitions 4 and 5 and name it as ‘disks-pool’
- Let us create a zfs pool comprising with partitions 6 and 7 and name it as ‘iso-pool’

For clarity, let us mark down the name of the zfs pools that we plan on creating on the existing partitions on the 1TB SSD

```
# lsblk
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda           8:0    1 460.3G  0 disk
├─sda1        8:1    1     1M  0 part
└─sda2        8:2    1 460.3G  0 part /  --> Root partition on USB thumb drive
nvme0n1     259:0    0   1.9T  0 disk   --> 2 TB NVMe SSD
├─nvme0n1p1 259:1    0   317G  0 part
├─nvme0n1p2 259:2    0   317G  0 part
├─nvme0n1p3 259:3    0   317G  0 part
├─nvme0n1p4 259:4    0   317G  0 part
├─nvme0n1p5 259:5    0   317G  0 part
├─nvme0n1p6 259:6    0   317G  0 part
└─nvme0n1p7 259:7    0   5.7G  0 part
nvme1n1     259:8    0 931.5G  0 disk   --> 1 TB NVMe SSD
```

```
├─nvme1n1p1 259:9    0   155G  0 part   --> Part of images-pool
├─nvme1n1p2 259:10   0   155G  0 part   --> Part of images-pool
├─nvme1n1p3 259:11   0   155G  0 part   --> Part of imgpool-pool
├─nvme1n1p4 259:12   0   155G  0 part   --> Part of disks-pool
├─nvme1n1p5 259:13   0   155G  0 part   --> Part of disks-pool
├─nvme1n1p6 259:14   0   155G  0 part   --> Part of iso-pool
└─nvme1n1p7 259:15   0   1.5G  0 part   --> Part of iso-pool
```

Finally let us go ahead and create 3 zfs pools as planned.

```
# zpool create images-pool /dev/nvme1n1p1 /dev/nvme1n1p2 /dev/nvme1n1p3
# zpool create disks-pool /dev/nvme1n1p4 /dev/nvme1n1p5
# zpool create iso-pool /dev/nvme1n1p6 /dev/nvme1n1p7
```

Let us list all the zfs pools that we have created on both the SSDs

```
# zpool list
NAME         SIZE  ALLOC   FREE  CKPOINT  EXPANDSZ   FRAG    CAP  DEDUP    HEALTH  ALTROOT
basepool    5.50G  5.62M  5.49G        -         -     2%     0%  1.00x    ONLINE  -
disks-pool   308G   694K   308G        -         -     0%     0%  1.00x    ONLINE  -
images-pool   462G   797K   462G        -         -     0%     0%  1.00x    ONLINE  -
iso-pool     154G   702K   154G        -         -     0%     0%  1.00x    ONLINE  -
zpool-1      316G   916K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-2      316G   636K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-3      316G   646K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-4      316G   634K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-5      316G   632K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-6      316G   634K   316G        -         -     0%     0%  1.00x    ONLINE  -
```

Task #05: Installation and Configuration of incus

Here are the list of things that we are going to accomplish in this task

1. Installiation of incus
2. Installation verification

1. Installation of incus

The commands to install incus is pretty straight forward. Though there are multiple ways to install incus, let us install from the repository maintained by zabbly.com - the sponsors, developers and maitainers of this open source project incus.

```
# uname -a
Linux red-handsonbox 6.8.0-55-generic #57-Ubuntu SMP PREEMPT_DYNAMIC Wed Feb 12 23:42:21
UTC 2025 x86_64 x86_64 x86_64 GNU/Linux

# cat /etc/os-release
-----
PRETTY_NAME="Ubuntu 24.04.2 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04.2 LTS (Noble Numbat)"
VERSION_CODENAME=noble
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=noble
LOGO=ubuntu-logo
-----

# curl -fsSL https://pkgs.zabbly.com/key.asc | gpg --show-keys --fingerprint
-----
gpg: directory '/root/.gnupg' created
gpg: keybox '/root/.gnupg/pubring.kbx' created
pub   rsa3072 2023-08-23 [SC] [expires: 2025-08-22]
      4EFC 5906 96CB 15B8 7C73  A3AD 82CC 8797 C838 DCFD
uid                   Zabbly Kernel Builds <info@zabbly.com>
sub   rsa3072 2023-08-23 [E] [expires: 2025-08-22]
-----
```

```
# mkdir -p /etc/apt/keyrings/

# curl -fsSL https://pkgs.zabbly.com/key.asc -o /etc/apt/keyrings/zabbly.asc

# sh -c 'cat <<EOF > /etc/apt/sources.list.d/zabbly-incus-stable.sources
Enabled: yes
Types: deb
URIs: https://pkgs.zabbly.com/incus/stable
Suites: $(. /etc/os-release && echo ${VERSION_CODENAME})
Components: main
Architectures: $(dpkg --print-architecture)
Signed-By: /etc/apt/keyrings/zabbly.asc

EOF'

# apt-get update

# apt install net-tools bridge-utils -y

# apt-get install incus -y

# systemctl restart incus.service

# systemctl status -l incus.service

# systemctl enable incus.service
```

The parameters passed to the command 'incus admin init' is important to understand and here is the breakup.

1) We are telling the command/installer not to configure any Clustering setup
2) We are planning to configure a default storage pool
3) We are passing the name 'basepool' to be used as the default storage. As you may recollect we have already created a zfs pool named 'basepool' in the previous chapter where it was created on Partition #7 of the 2 GB SSD
4) We intented to use zfs filesystem instead of other filesystems
5) We are telling the command/installer not to create a new zfs pool but use an existing zfs pool based storage pool named 'basepool' that we already pre-created and is ready for use
6) With regards to network configuration, we are going with the suggested name 'incusbr0' to be used to create a network bridge
7) We are assigning a static ip address of our choice (10.1.1.253/24) to the incus bridge (incusbr0)

The rest of the parameters passed are self explanatory

```
# incus admin init
-----
Would you like to use clustering? (yes/no) [default=no]:
Do you want to configure a new storage pool? (yes/no) [default=yes]:
Name of the new storage pool [default=default]: basepool
Name of the storage backend to use (zfs, btrfs, dir, lvm, lvmcluster) [default=zfs]:
Create a new ZFS pool? (yes/no) [default=yes]: no
Name of the existing ZFS pool or dataset: basepool
Would you like to create a new local network bridge? (yes/no) [default=yes]:
What should the new bridge be called? [default=incusbr0]:
What IPv4 address should be used? (CIDR subnet notation, “auto” or “none”) [default=auto]:
10.1.1.253/24
Would you like to NAT IPv4 traffic on your bridge? [default=yes]:
What IPv6 address should be used? (CIDR subnet notation, “auto” or “none”) [default=auto]:
none
Would you like the server to be available over the network? (yes/no) [default=no]: yes
Address to bind to (not including port) [default=all]:
Port to bind to [default=8443]:
Would you like stale cached images to be updated automatically? (yes/no) [default=yes]:
Would you like a YAML "init" preseed to be printed? (yes/no) [default=no]: yes
config:
  core.https_address: '[::]:8443'
networks:
- config:
    ipv4.address: 10.1.1.253/24
    ipv4.nat: "true"
    ipv6.address: none
  description: ""
  name: incusbr0
  type: ""
  project: default
storage_pools:
- config:
    source: basepool
  description: ""
  name: basepool
  driver: zfs
storage_volumes: []
profiles:
- config: {}
  description: ""
  devices:
```

```
    eth0:
      name: eth0
      network: incusbr0
      type: nic
    root:
      path: /
      pool: basepool
      type: disk
  name: default
  project: default
projects: []
cluster: null
-----
```

2. Installation verification

```
# ps -ef | grep dns
incus     21691   21299  0 04:14 ?       00:00:00 dnsmasq --keep-in-foreground
--strict-order --bind-interfaces --except-interface=lo --pid-file= --no-ping
--interface=incusbr0 --dhcp-rapid-commit --no-negcache --quiet-dhcp --quiet-dhcp6
--quiet-ra --listen-address=10.1.1.253 --dhcp-no-override --dhcp-authoritative
--dhcp-leasefile=/var/lib/incus/networks/incusbr0/dnsmasq.leases
--dhcp-hostsfile=/var/lib/incus/networks/incusbr0/dnsmasq.hosts --dhcp-range
10.1.1.2,10.1.1.254,1h -s incus --interface-name _gateway.incus,incusbr0 -S /incus/
--conf-file=/var/lib/incus/networks/incusbr0/dnsmasq.raw -u incus -g incus

# brctl show incusbr0
bridge name bridge id       STP enabled interfaces
incusbr0        8000.00163e1b7238   no

# incus network show incusbr0
config:
  ipv4.address: 10.1.1.253/24
  ipv4.nat: "true"
  ipv6.address: none
description: ""
name: incusbr0
type: bridge
used_by:
- /1.0/profiles/default
managed: true
status: Created
locations:
```

```
- none
project: default

# incus network list -f compact
    NAME      TYPE    MANAGED     IPV4      IPV6  DESCRIPTION  USED BY   STATE
  br-int     bridge    NO                                    0
  docker0    bridge    NO                                    0
  eno1       physical  NO                                    0
  eno2       physical  NO                                    0
  eno3       physical  NO                                    0
  eno4       physical  NO                                    0
  incusbr0   bridge    YES     10.1.1.253/24  none            1       CREATED
  lo         loopback  NO                                   0
```

Task #06: Incus Storage

Here are the list of things that we are going to accomplish in this task.

1. incus managing storage pools on 2 TB SSD
2. incus managing storage pools on 1 TB SSD

1. incus managing storage pools of 2 TB SSD

Let us take a quick look the output of 'lsblk' command and see how we have partitioned the SSDs and what zfs pools reside on each parition.

```
# lsblk
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda           8:0    1 460.3G  0 disk
├─sda1        8:1    1     1M  0 part
└─sda2        8:2    1 460.3G  0 part / --> Root partition on USB thumb drive
nvme0n1     259:0    0   1.9T  0 disk   --> 2 TB NVMe SSD
├─nvme0n1p1 259:1    0   317G  0 part   --> zfs pool named zpool-1
├─nvme0n1p2 259:2    0   317G  0 part   --> zfs pool named zpool-2
├─nvme0n1p3 259:3    0   317G  0 part   --> zfs pool named zpool-3
├─nvme0n1p4 259:4    0   317G  0 part   --> zfs pool named zpool-4
├─nvme0n1p5 259:5    0   317G  0 part   --> zfs pool named zpool-5
├─nvme0n1p6 259:6    0   317G  0 part   --> zfs pool named zpool-6
└─nvme0n1p7 259:7    0   5.7G  0 part   --> zfs pool named basebpool
nvme1n1     259:8    0 931.5G  0 disk   --> 1 TB NVMe SSD
├─nvme1n1p1 259:9    0   155G  0 part
├─nvme1n1p2 259:10   0   155G  0 part
├─nvme1n1p3 259:11   0   155G  0 part
├─nvme1n1p4 259:12   0   155G  0 part
├─nvme1n1p5 259:13   0   155G  0 part
├─nvme1n1p6 259:14   0   155G  0 part
└─nvme1n1p7 259:15   0   1.5G  0 part
```

Let us list all the zfs pools that we have created so far on each parttion of both the SSDs.

```
# zpool list
NAME         SIZE  ALLOC   FREE  CKPOINT  EXPANDSZ   FRAG    CAP  DEDUP    HEALTH  ALTROOT
basepool     5.50G  6.49M  5.49G       -         -     6%     0%  1.00x    ONLINE  - --> on 2TB SSD
disks-pool    308G   694K   308G       -         -     0%     0%  1.00x    ONLINE  -
images-pool   462G   681K   462G       -         -     0%     0%  1.00x    ONLINE  -
iso-pool      154G   657K   154G       -         -     0%     0%  1.00x    ONLINE  -
zpool-1       316G   746K   316G       -         -     0%     0%  1.00x    ONLINE  - --> on 2TB SSD
```

```
zpool-2      316G   636K   316G        -         -     0%     0%  1.00x    ONLINE  - --> on 2TB SSD
zpool-3      316G   646K   316G        -         -     0%     0%  1.00x    ONLINE  - --> on 2TB SSD
zpool-4      316G   634K   316G        -         -     0%     0%  1.00x    ONLINE  - --> on 2TB SSD
zpool-5      316G   632K   316G        -         -     0%     0%  1.00x    ONLINE  - --> on 2TB SSD
zpool-6      316G   634K   316G        -         -     0%     0%  1.00x    ONLINE  - --> on 2TB SSD
```

We will now go ahead and hand over the zfs pools that we have created on 2TB SSDs to incus for managing storage resources.

Note that the name 'Hands-on-lab' is being used in the description. The reason being, soon we will be creating Projects with the same name (Handson-lab). That way it will be easier to identify which storage pool is assighned to which Project. This will be much clearer with following tasks.

```
# incus storage create storage-1 zfs source=zpool-1 --description="Storage for Containers,
VMs of Handson-lab-1"
# incus storage create storage-2 zfs source=zpool-2 --description="Storage for Containers,
VMs of Handson-lab-2"
# incus storage create storage-3 zfs source=zpool-3 --description="Storage for Containers,
VMs of Handson-lab-3"
# incus storage create storage-4 zfs source=zpool-4 --description="Storage for Containers,
VMs of Handson-lab-4"
# incus storage create storage-5 zfs source=zpool-5 --description="Storage for Containers,
VMs of Handson-lab-5"
# incus storage create storage-6 zfs source=zpool-6 --description="Storage for Containers,
VMs of Handson-lab-6"
```

You may recollect that 'basepool' is already under the control of incus as that is what we passed to 'incus admin init' command at initial setup. Though this step is optional, let us manully edit the configuration of basepool , modify its description so that it will be vertbatim

```
# export EDITOR=vi

# incus storage edit basepool
-----
config:
  source: basepool
  volatile.initial_source: basepool
  zfs.pool_name: basepool
description: passed to incus admin init
name: basepool
driver: zfs
used_by:
- /1.0/profiles/default
```

```
status: Created
locations:
- none
-----
```

Let us now list all the storage pools managed by incus

```
# incus storage list -f compact
NAME    DRIVER               DESCRIPTION                       USED BY   STATE
basepool   zfs    passed to incus admin init                     1       CREATED
storage-1  zfs    Storage for Containers, VMs of Handson-lab-1  0       CREATED
storage-2  zfs    Storage for Containers, VMs of Handson-lab-2  0       CREATED
storage-3  zfs    Storage for Containers, VMs of Handson-lab-3  0       CREATED
storage-4  zfs    Storage for Containers, VMs of Handson-lab-4  0       CREATED
storage-5  zfs    Storage for Containers, VMs of Handson-lab-5  0       CREATED
storage-6  zfs    Storage for Containers, VMs of Handson-lab-6  0       CREATED
```

From now on, we will refer to the storage pools with their assigned names storage-1, storage-2, storage-3 and so on.

2. incus managing storage pools of 1 TB SSD

Here is the current status of the 1TB SSD

```
# lsblk
NAME       MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda          8:0    1 460.3G  0 disk
├─sda1       8:1    1    1M  0 part
└─sda2       8:2    1 460.3G  0 part / --> Root partition on USB thumb drive
nvme0n1    259:0    0   1.9T  0 disk   --> 2 TB NVMe SSD
├─nvme0n1p1 259:1    0   317G  0 part
├─nvme0n1p2 259:2    0   317G  0 part
├─nvme0n1p3 259:3    0   317G  0 part
├─nvme0n1p4 259:4    0   317G  0 part
├─nvme0n1p5 259:5    0   317G  0 part
├─nvme0n1p6 259:6    0   317G  0 part
└─nvme0n1p7 259:7    0   5.7G  0 part
nvme1n1    259:8    0 931.5G  0 disk   --> 1 TB NVMe SSD
├─nvme1n1p1 259:9    0   155G  0 part   --> part of images-pool
├─nvme1n1p2 259:10   0   155G  0 part   --> part of images-pool
├─nvme1n1p3 259:11   0   155G  0 part   --> part of images-pool
├─nvme1n1p4 259:12   0   155G  0 part   --> part of disks-pool
├─nvme1n1p5 259:13   0   155G  0 part   --> part of disks-pool
├─nvme1n1p6 259:14   0   155G  0 part   --> part of iso-pool
```

```
└─nvme1n1p7 259:15   0   1.5G  0 part   --> part of iso-pool
```

Let us list all the storage pools that reside on both the SSDs

```
# zpool list
NAME         SIZE  ALLOC   FREE  CKPOINT  EXPANDSZ   FRAG    CAP  DEDUP    HEALTH  ALTROOT
basepool     5.50G  6.49M  5.49G       -         -     6%     0%  1.00x    ONLINE  -
disks-pool    308G   694K   308G       -         -     0%     0%  1.00x    ONLINE  -  --> On 1TB SSD
images-pool   462G   681K   462G       -         -     0%     0%  1.00x    ONLINE  -  --> On 1TB SSD
iso-pool      154G   657K   154G       -         -     0%     0%  1.00x    ONLINE  -  --> On 1TB SSD
zpool-1       316G   746K   316G       -         -     0%     0%  1.00x    ONLINE  -
zpool-2       316G   636K   316G       -         -     0%     0%  1.00x    ONLINE  -
zpool-3       316G   646K   316G       -         -     0%     0%  1.00x    ONLINE  -
zpool-4       316G   634K   316G       -         -     0%     0%  1.00x    ONLINE  -
zpool-5       316G   632K   316G       -         -     0%     0%  1.00x    ONLINE  -
zpool-6       316G   634K   316G       -         -     0%     0%  1.00x    ONLINE  -
```

Let us now hand over the zfs pools that we have created on 1 TB SSDs to incus for managing storage resources.

```
# incus storage create shared-images zfs source=images-pool --description="shared images accross Projects"

# incus storage create shared-disks zfs source=disks-pool --description="shared disks accross Projects"

# incus storage create shared-iso zfs source=iso-pool --description="shared ISOs accross Projects"
```

Here is the final list of all storage pools managed by incus

```
# incus storage list -f compact
     NAME       DRIVER                DESCRIPTION                      USED BY    STATE
  basepool      zfs     passed to incus admin init                       1        CREATED
  shared-disks   zfs     shared disks accross Projects                   0        CREATED
  shared-images  zfs     shared images accross Projects                  0        CREATED
  shared-iso     zfs     shared ISOs accross Projects                    0        CREATED
  storage-1      zfs     Storage for Containers, VMs of Handson-lab-1    0        CREATED
  storage-2      zfs     Storage for Containers, VMs of Handson-lab-2    0        CREATED
  storage-3      zfs     Storage for Containers, VMs of Handson-lab-3    0        CREATED
  storage-4      zfs     Storage for Containers, VMs of Handson-lab-4    0        CREATED
  storage-5      zfs     Storage for Containers, VMs of Handson-lab-5    0        CREATED
  storage-6      zfs     Storage for Containers, VMs of Handson-lab-6    0        CREATED
```

Task # 07: OVN Installation

OVN stands for Open Virtual Networking. The pimary reason to install OVN networking with incus in our setup is to reuse the same ip addresses and ip address ranges assigned to all the virtual machines and containers across projects - all at the same time without any conflicts. This is an advanced concept and is easy to get confused. So we will take it slowly to understand it better in a way we can build them in a methodical way in the upcoming tasks.

In this task we will fncus on installing ovn networking packages, which is pretty straight forward.

```
# apt install ovn-host ovn-central -y

# ovs-vsctl set open_vswitch . \
   external_ids:ovn-remote=unix:/run/ovn/ovnsb_db.sock \
   external_ids:ovn-encap-type=geneve \
   external_ids:ovn-encap-ip=127.0.0.1

# incus network list -f compact
    NAME      TYPE    MANAGED     IPV4      IPV6  DESCRIPTION  USED BY   STATE
  br-int     bridge    NO                                      0
  docker0    bridge    NO                                      0
  eno1       physical  NO                                      0
  eno2       physical  NO                                      0
  eno3       physical  NO                                      0
  eno4       physical  NO                                      0
  incusbr0   bridge    YES     10.1.1.253/24  none             1       CREATED
  lo         loopback  NO                                     0
  ovs-system  unknown   NO                                     0
```

Here are the details:

a. ‘eno3’ is the physical network interface where host ip address (192.168.1.10) is configured. This is the only uplink on the host
b. eno1, eno2 and eno4 are interfaces where none of them are configured
c. ‘docker0’ is the docker bridge that was created while installing docker
d. ‘incusbr0’ is the incus bridge that was created while installing incus
e. ‘ovn-system’ is what we installed just now

Task #08: Installation of incus-ui-canonical

Apart from the command line, incus platform can be administered from a browser as well. The package providing web UI is straight forward, which is shown below.

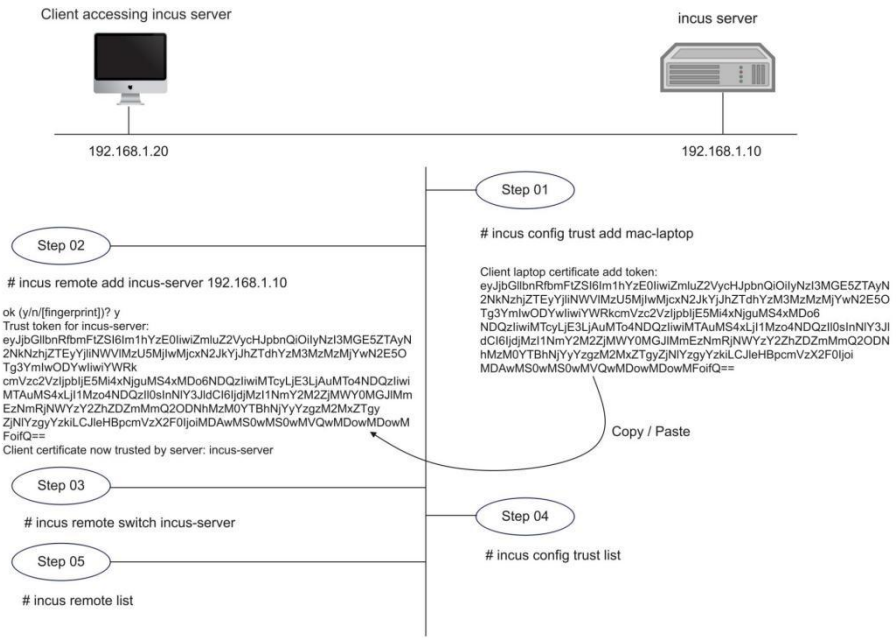
There is yet another open source project named LXConsole to manage incus via brower. For now we will focus and use incus-ui-canonical.

```
# apt install incus-ui-canonical -y
-----
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  incus-ui-canonical
0 upgraded, 1 newly installed, 0 to remove and 2 not upgraded.
Need to get 3,728 kB of archives.
After this operation, 20.1 MB of additional disk space will be used.
Get:1 https://pkgs.zabbly.com/incus/stable noble/main amd64 incus-ui-canonical amd64
1:6.10.1-ubuntu24.04-202503210251 [3,728 kB]
Fetched 3,728 kB in 2s (1,655 kB/s)
Selecting previously unselected package incus-ui-canonical.
(Reading database ... 90200 files and directories currently installed.)
Preparing                                                                    to
unpack .../incus-ui-canonical_1%3a6.10.1-ubuntu24.04-202503210251_amd64.deb ...
Unpacking incus-ui-canonical (1:6.10.1-ubuntu24.04-202503210251) ...
Setting up incus-ui-canonical (1:6.10.1-ubuntu24.04-202503210251) ...
Scanning processes...
Scanning processor microcode...
Scanning linux images...

Running kernel seems to be up-to-date.

The processor microcode seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

```
-----

# incus config set core.https_address :8443

# incus config show
-----
config:
  core.https_address: :8443
-----
```

Task #09: Incus Remote access setup

Once the incus server is up and running, configuring a client machine to remotely access the incus server needs a simple setup. Shown below is a workflow explaining the required steps on both the incus server and the client machine.



In the commands shown below, hostname of the incus server is 'red-handsonbox' and the name of the client machine which will eventually access the incus server remotely is 'mac'.

Step 01:
On the incus server, add the name of the client machine to its trust list. Entering the command shown below will generate a token

```
# incus config trust add mac

Client mac certificate add token:
eyJjbGllbnRfbmFtZSI6Im1hYzE0IiwiZmluZ2VycHJpbnQiOiIyNzI3MGE5ZTAyN2NkNzhjZTEyYjliNWVlMzU
5MjIwMjcxN2JkYjJhZTdhYzM3MzMzMjYwN2E5OTg3YmIwODYwIiwiYWRkcmVzc2VzIjpbIjE5Mi4xNjguMS4xMD
o6NDQzIiwiMTcyLjE3LjAuMTo4NDQzIiwiMTAuMS4xLjI1Mzo4NDQzIl0sInNlY3JldCI6IjdjMzI1NmY2M2ZjM
WY0MGJlMmEzNmRjNWYzY2ZhZDZmMmQ2ODNhMzM0YTBhNjYyYzgzM2MxZTgyZjNlYzgyYzkiLCJleHBpcmVzX2F0
IjoiNDAwMS0wMS0wMVQwMDowMDowMFoifQ==
```

Step 02:
On the client (mac) laptop, run the command shown below, followed by the generated token on the server

```
mac$ incus remote add red-handsonbox 192.168.1.10
ok (y/n/[fingerprint])? y
Trust token for red-handsonbox:
eyJjbGllbnRfbmFtZSI6Im1hYzE0IiwiZmluZ2VycHJpbnQiOiIyNzI3MGE5ZTAyN2NkNzhjZTEyYjliNWVlMzU
5MjIwMjcxN2JkYjJhZTdhYzM3MzMzMjYwN2E5OTg3YmIwODYwIiwiYWRkcmVzc2VzIjpbIjE5Mi4xNjguMS4xMD
o6NDQzIiwiMTcyLjE3LjAuMTo4NDQzIiwiMTAuMS4xLjI1Mzo4NDQzIl0sInNlY3JldCI6IjdjMzI1NmY2M2ZjM
WY0MGJlMmEzNmRjNWYzY2ZhZDZmMmQ2ODNhMzM0YTBhNjYyYzgzM2MxZTgyZjNlYzgyYzkiLCJleHBpcmVzX2F0
IjoiNDAwMS0wMS0wMVQwMDowMDowMFoifQ==
Client certificate now trusted by server: red-handsonbox
```

Step 03:
On the client (mac) laptop, run the command to switch context to the newly added remote incus server

```
mac$ incus remote switch red-handsonbox
```

Step 04:
On the client, verify that incus server is sucessfully added to its remote list

```
mac$ incus remote list -f compact
NAME   URL    PROTOCOL AUTH TYPE   PUBLIC  STATIC  GLOBAL
images  https://images.linuxcontainers.org  simplestreams  none YES NO NO
local     unix://                          incus         file access  NO     YES     NO
red-handsonbox (current)  https://192.168.1.10:8443  incus tls  NO NO NO
```

Step 05:
On the incus serer, verify that the client 'mac' is added successfully to its trust list

```
# incus config trust list -f compact
  NAME      TYPE   DESCRIPTION  FINGERPRINT      EXPIRY DATE
  mac     client             c88b6a7dd7e8  2035/01/04 21:59 UTC
```

Now that the trust for mac laptop is established on the incus server (red-handonbox), we can issue commands directly from a shell terminal on Mac laptop (represented as 'mac%')

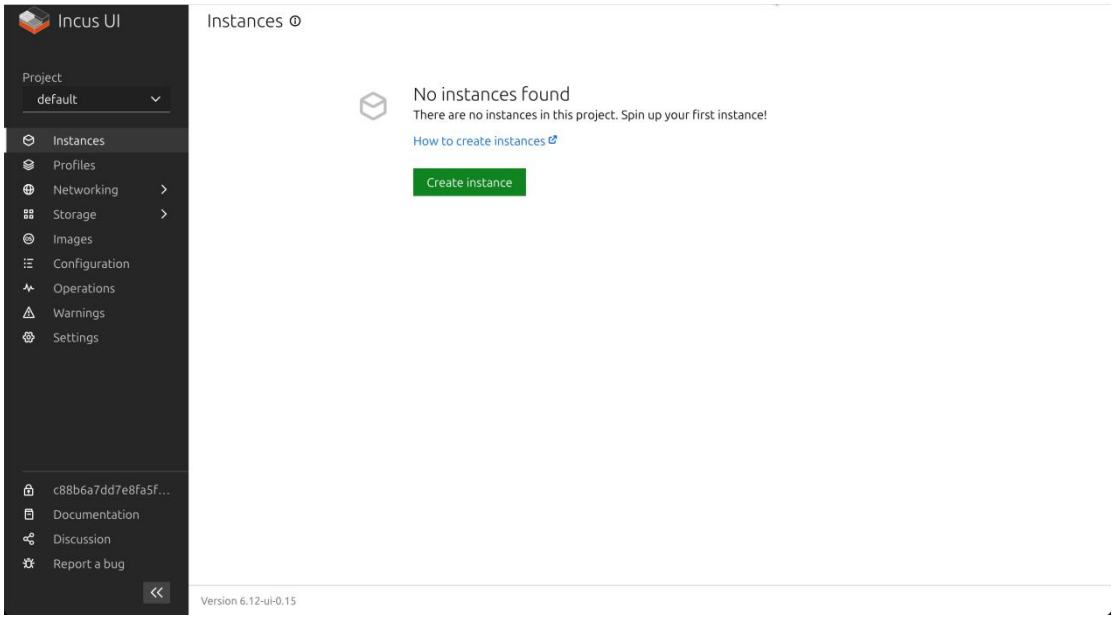
Task #10: Incus web UI

In this task we will take a quick look at incus web UI.

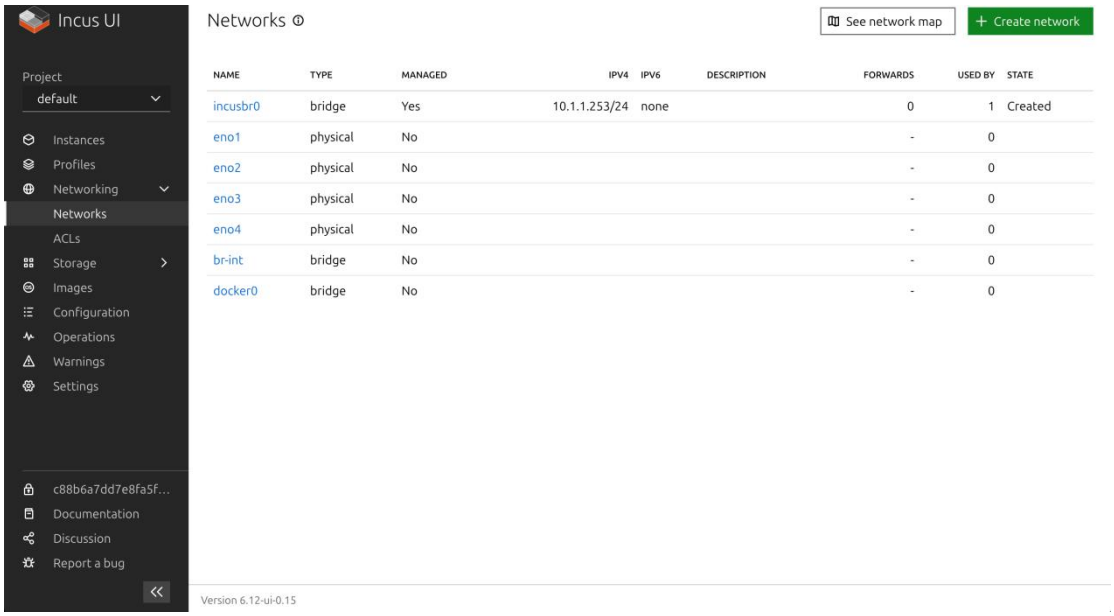
Now that we have established trust between the mac client and the incus server, let us run the following command on the mac machine, which will open up a browser and point to incus web UI

```
mac$ incus webui
Web server running at:
http://127.0.0.1:53100/ui?auth_token=eb79cec0-eef6-480c-b6e2-3178c62e91e5
```

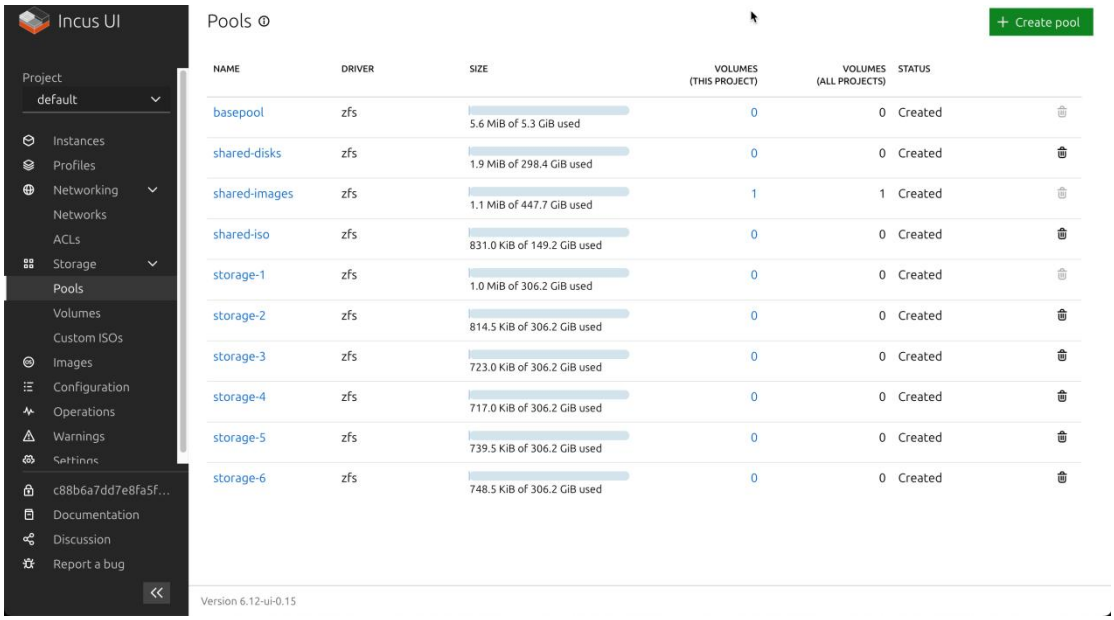
This would be the landing page. It currently shows we do not have any intances of containers or virtual machines running.



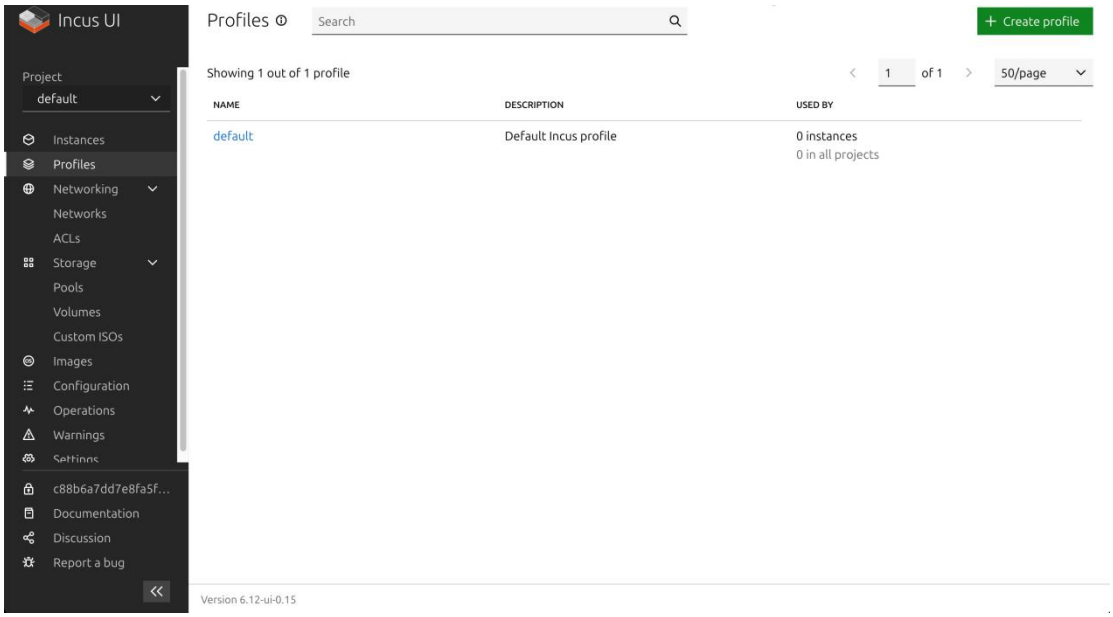
Let us take a look at the Networks

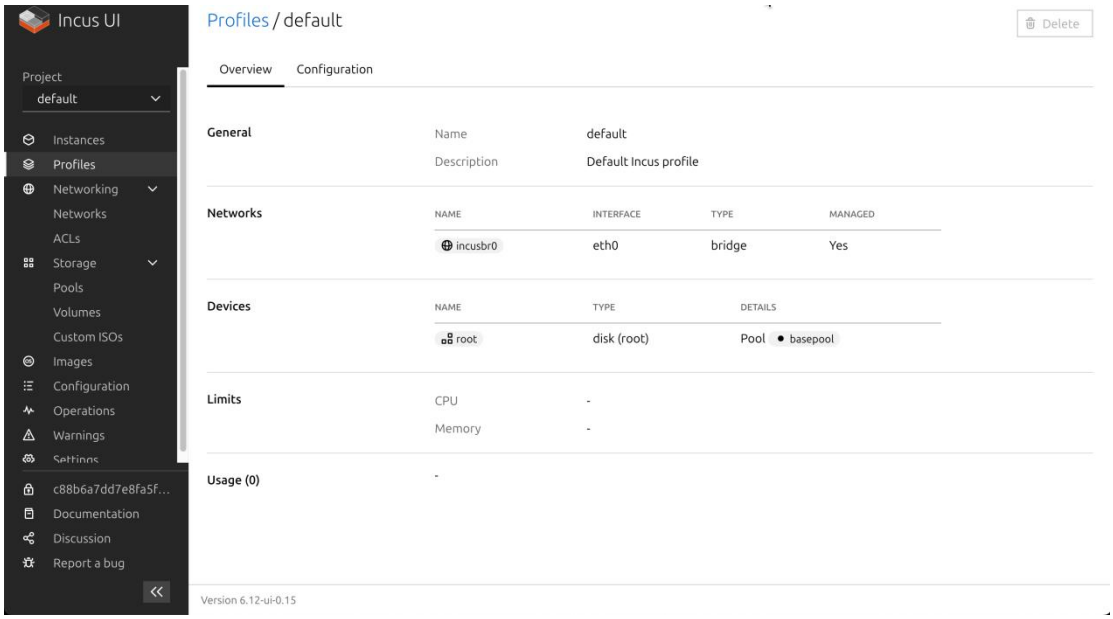


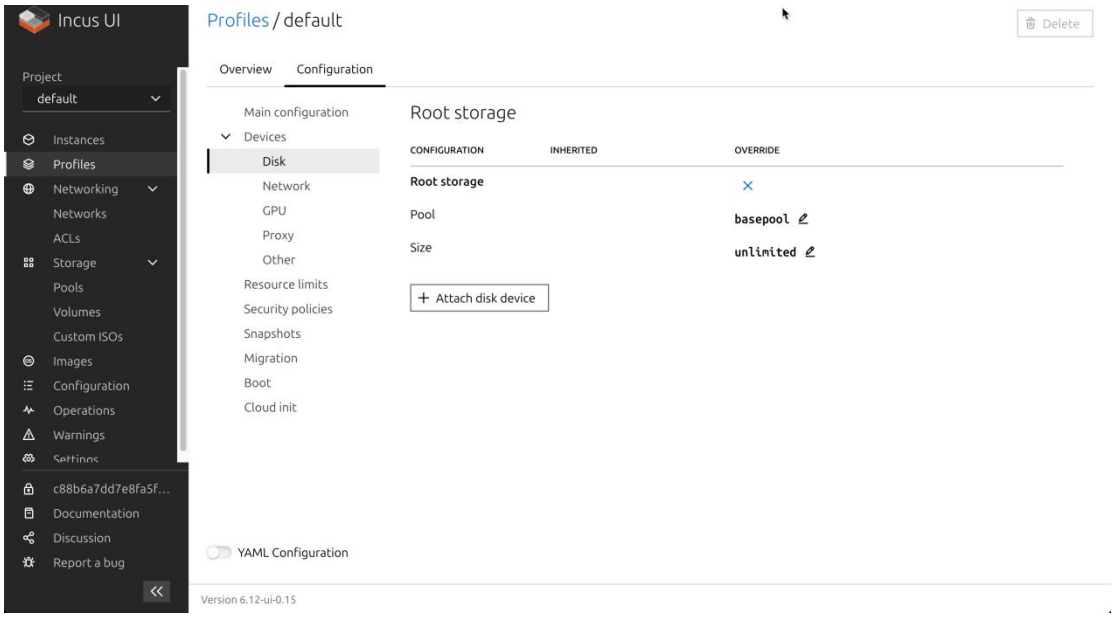
Will go ahead and check storage pools



Finally we will take a look at the details of the default Profile






Incus UI
Project
default
Instances
Profiles
Networking
Networks
ACLs
Storage
Pools
Volumes
Custom ISOs
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Profiles / default
Delete
Overview
Configuration
Main configuration
Devices
Disk
Network
GPU
Proxy
Other
Resource limits
Security policies
Snapshots
Migration
Boot
Cloud init
Root storage
CONFIGURATION
INHERITED
OVERRIDE
Root storage
Pool
basepool
Size
unlimited
+ Attach disk device
YAML Configuration
Version 6.12-ui-0.15

Task #11: Sanity test with a Container

Let us do a sanity testing by launching an ubuntu container. By analyzing the container we see where things stand with regards to what we have built so far.

```
# incus launch images:ubuntu/24.04 c1
```

```
# incus list
+------+---------+-----------------+------+-----------+-----------+
| NAME |  STATE  |      IPV4       | IPV6 |   TYPE    | SNAPSHOTS |
+------+---------+-----------------+------+-----------+-----------+
| c1   | RUNNING | 10.1.1.57 (eth0) |      | CONTAINER | 0         |
+------+---------+-----------------+------+-----------+-----------+
```

Let us get inside the newly spawned container (c1) and ping google dns server, which confirms the network is working

```
# incus shell c1
root@c1:~# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=115 time=22.0 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=115 time=23.1 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=115 time=22.4 ms
```

Let us go ahead and update, upgrade ubuntu distro installed in the container (c1)

```
root@c1:~# apt -y update
Hit:1 http://archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [922 kB]
Fetched 1,174 kB in 1s (876 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
```

```
root@c1:~# apt -y upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

```
root@c1:~#
```

Now let us install package for 'traceroute'

```
root@c1:~# apt install -y traceroute
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  traceroute
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 60.5 kB of archives.
After this operation, 162 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu noble/universe amd64 traceroute amd64 1:2.1.5-1
[60.5 kB]
Fetched 60.5 kB in 1s (107 kB/s)
Selecting previously unselected package traceroute.
(Reading database ... 16181 files and directories currently installed.)
Preparing to unpack .../traceroute_1%3a2.1.5-1_amd64.deb ...
Unpacking traceroute (1:2.1.5-1) ...
Setting up traceroute (1:2.1.5-1) ...
update-alternatives: using /usr/bin/traceroute.db to provide /usr/bin/traceroute
(traceroute) in auto mode
update-alternatives: using /usr/bin/traceroute6.db to provide /usr/bin/traceroute6
(traceroute6) in auto mode
update-alternatives: using /usr/bin/lft.db to provide /usr/bin/lft (lft) in auto mode
update-alternatives: using /usr/bin/traceproto.db to provide /usr/bin/traceproto
(traceproto) in auto mode
update-alternatives: using /usr/sbin/tcptraceroute.db to provide /usr/sbin/tcptraceroute
(tcptraceroute) in auto mode
root@c1:~#
```

Let us confirm that the network traffic does go through newly created incus bridge (incusbr0) and then via its uplink physical nic - en03 in this case.

```
root@c1:~# traceroute zabbly.com
traceroute to zabbly.com (45.45.148.7), 30 hops max, 60 byte packets
 1  _gateway.incus (10.1.1.253)  0.059 ms  0.023 ms  0.017 ms
 2  _gateway (192.168.1.1)  0.336 ms  0.872 ms  0.832 ms
 3  10.61.209.3 (10.61.209.3)  10.462 ms  16.893 ms 10.61.209.2 (10.61.209.2)  16.817 ms
 4  * * *
.....
.....
.....
```

```
root@c1:~# exit
logout
```

Let us take a look at the configuration of the newly spawned container (c1)

```
# incus config show c1
architecture: x86_64
config:
  image.architecture: amd64
  image.description: Ubuntu noble amd64 (20250503_07:42)
  image.os: Ubuntu
  image.release: noble
  image.requirements.cgroup: v2
  image.serial: "20250503_07:42"
  image.type: squashfs
  image.variant: default
  volatile.base_image: b3f31aaf5264c50ba425e769160f586b43a07248c41f1b1680b1b634fd0b1eac
  volatile.cloud-init.instance-id: 1be64a01-047e-43fd-8157-5e72176d5a95
  volatile.eth0.host_name: veth41f59eb7
  volatile.eth0.hwaddr: 10:66:6a:67:88:e0
  volatile.idmap.base: "0"
  volatile.idmap.current:
'[{"Isuid":true,"Isgid":false,"Hostid":1000000,"Nsid":0,"Maprange":1000000000},{"Isuid":
false,"Isgid":true,"Hostid":1000000,"Nsid":0,"Maprange":1000000000}]'
  volatile.idmap.next:
'[{"Isuid":true,"Isgid":false,"Hostid":1000000,"Nsid":0,"Maprange":1000000000},{"Isuid":
false,"Isgid":true,"Hostid":1000000,"Nsid":0,"Maprange":1000000000}]'
  volatile.last_state.idmap: '[]'
  volatile.last_state.power: RUNNING
  volatile.uuid: f7bb3e51-b902-4218-ab45-018f66d2feeb
  volatile.uuid.generation: f7bb3e51-b902-4218-ab45-018f66d2feeb
devices: {}
ephemeral: false
profiles:
- default
stateful: false
description: ""
```

Let us look at the values of 2 configuration parameters namely 'volatile.base_image' and 'profile' and see what they reveal

```
# incus image list -f compact
  ALIAS  FINGERPRINT   PUBLIC           DESCRIPTION           ARCHITECTURE    TYPE
SIZE        UPLOAD DATE
```

```
        b3f31aaf5264  no      Ubuntu noble amd64 (20250503_07:42)  x86_64       CONTAINER
129.19MiB  2025/05/04 05:10 UTC
```

The finger print value reveals from which image the container is spawned (which is the image Id of ubuntu24.04 from the public repository "images"). The following command shows the same:

```
# incus image list images: b3f31aaf5264 -f compact
        ALIAS                  FINGERPRINT     PUBLIC                          DESCRIPTION
ARCHITECTURE    TYPE      SIZE        UPLOAD DATE
  ubuntu/noble (7 more)  b3f31aaf5264  yes     Ubuntu noble amd64 (20250503_07:42)  x86_64
CONTAINER  129.19MiB  2025/05/03 00:00 UTC

# incus image show b3f31aaf5264
auto_update: true
properties:
  architecture: amd64
  description: Ubuntu noble amd64 (20250503_07:42)
  os: Ubuntu
  release: noble
  requirements.cgroup: v2
  serial: "20250503_07:42"
  type: squashfs
  variant: default
public: false
expires_at: 1970-01-01T00:00:00Z
profiles:
- default
```

Now let us look at the value of profile (which is 'default') shown by the following command:

```
# incus profile list
+---------+----------------------+---------+
|  NAME   |     DESCRIPTION      | USED BY |
+---------+----------------------+---------+
| default | Default Incus profile | 1       |
+---------+----------------------+---------+

# incus profile show default
config: {}
description: Default Incus profile
devices:
  eth0:
    name: eth0
    network: incusbr0
```

```
    type: nic
  root:
    path: /
    pool: basepool
    type: disk
name: default
used_by:
- /1.0/instances/c1
project: default
```

The value of 'pool' shows 'basepool' which means the root disk of this container (c1) is located in the storage named basepool. We can display the configurtion of the storage basepool with the command:

```
# incus storage show basepool
config:
  source: basepool
  volatile.initial_source: basepool
  zfs.pool_name: basepool
description: passed to incus admin init
name: basepool
driver: zfs
used_by:
- /1.0/images/b3f31aaf5264c50ba425e769160f586b43a07248c41f1b1680b1b634fd0b1eac
- /1.0/instances/c1
- /1.0/profiles/default
status: Created
locations:
- none
```

The values listed under 'used_by' reveals exactly what we have seen so far:

a) the name of the container (c1)

b) the image id used to spawn the container (b3f31aaf5264)

c) the profile (default) used.

Finally let us do cleanup. Let us stop and delete the container

```
# incus stop c1 ; incus delete c1
```

Task #12: Creating a Volume

In this task we will work on creating a new storage volume. We will create this volume on a specific incus storage pool (shared-images) that we have already created.

This storage volume can be used as a local repository where we can store all the images that can be downloaded from public image servers that are available on the internet. Having this kind of storage volume will be handy when we want to build solutions that can be air-gaped. With this approach, it can save us time and network bandwidth on downloading images from public images servers.

As you may recall, we have already created an incus storage pool named 'shared-images' (backed by zfs pool named 'images-pool'). It is on this storage pool we will create a new volume for the purpose explained above.

```
# incus storage show shared-images
config:
  source: images-pool
  volatile.initial_source: images-pool
  zfs.pool_name: images-pool
description: shared images accross Projects
name: shared-images
driver: zfs
used_by: []
status: Created
locations:
- none
```

Let us create a volume on it with the name 'global-volume'.

```
# incus storage volume create shared-images global-volume
```

Let us take a look at the same storage pool again. Note that the newly created 'global-volume' is now listed under 'used_by:'

```
# incus storage show shared-images
config:
  source: images-pool
  volatile.initial_source: images-pool
  zfs.pool_name: images-pool
description: shared images accross Projects
name: shared-images
driver: zfs
```

```
used_by:
- /1.0/storage-pools/shared-images/volumes/custom/global-volume
status: Created
locations:
- none

# incus storage volume show shared-images global-volume
config: {}
description: ""
name: global-volume
type: custom
used_by: []
location: none
content_type: filesystem
project: default
created_at: 2025-04-30T01:52:52.264481907Z
```

For clarity we will modify the description field to reflect its purpose

```
# incus storage volume edit shared-images global-volume  --> add the description
```

Please note the syntax. We need to supply both the name of the volume as well as the storage pool on which it resides

```
# incus storage volume show shared-images global-volume
config: {}
description: Global Volume to store all images, shared across all projects
name: global-volume
type: custom
used_by: []
location: none
content_type: filesystem
project: default
created_at: 2025-04-30T01:52:52.264481907Z
```

Now that we have created the volume, let us assign it to a gobal key which is primarily meant for the same purpose. ie, defining the location of the volume that contains images.

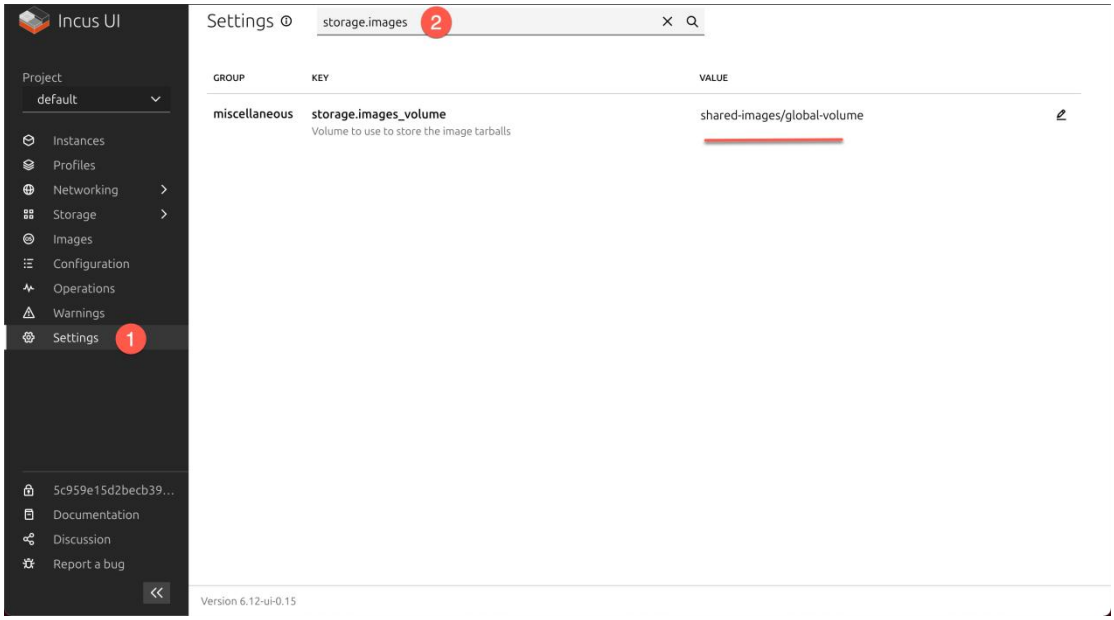
```
# incus config edit
config:
  core.https_address: :8443
  storage.images_volume: shared-images/global-volume <-- add this line
```

```
# incus config show
config:
  core.https_address: :8443
  storage.images_volume: shared-images/global-volume
```

It is also worth knowing the location of this volume in the file system so that we can verify manually where all the images will eventually be stored. And here is that location:

```
# ls -l /var/lib/incus/storage-pools/shared-images/custom/default_global-volume/
```

This setting can also be cross-checked from the web UI by selecting Settings and searching for the key storage.images_volume



Task #13: Sanity testing the volume

In this task let us do a sanity test on the volume that we created to hold all the images, making sure it is working as expected. This is the location where all the images are expected to be stored. Initially the directory will be empty.

```
# ls -l /var/lib/incus/storage-pools/shared-images/custom/default_global-volume/
```

Let us launch a rockylinux 9 container

```
# incus image copy images:rockylinux/9 local:
Image copied successfully!
```

```
# incus image list
+-------+--------------+--------+-------------------------------------+--------------+-----------+-----------+----------------------+
| ALIAS | FINGERPRINT  | PUBLIC |           DESCRIPTION               | ARCHITECTURE |   TYPE    |   SIZE    |    UPLOAD DATE       |
+-------+--------------+--------+-------------------------------------+--------------+-----------+-----------+----------------------+
|       | e80fa8f0662e | no     | Rockylinux 9 amd64 (20250506_02:06) | x86_64       | CONTAINER | 112.16MiB | 2025/05/06 05:50 UTC |
+-------+--------------+--------+-------------------------------------+--------------+-----------+-----------+----------------------+
```

We will take a look at the same directory again.

```
# ls -l /var/lib/incus/storage-pools/shared-images/custom/default_global-volume/
total 114930
-rw-r--r-- 1 root root      932 May  6 05:50 e80fa8f0662efdc1781749cf3cb5f58a95a745327d125108410b0c7ad61ea299
-rw-r--r-- 1 root root 117612544 May  6 05:50 e80fa8f0662efdc1781749cf3cb5f58a95a745327d125108410b0c7ad61ea299.rootfs
```

We can see that the directory is now populated with the image. We can also confirm the name of image based on its fingerprint. Let us launch a new container (c1) using the fingerprint of the image

```
# incus launch local:e80fa8f0662e c1
Launching c1

# incus list
+------+---------+--------------------+------+-----------+-----------+
| NAME |  STATE  |      IPV4          | IPV6 |   TYPE    | SNAPSHOTS |
+------+---------+--------------------+------+-----------+-----------+
| c1   | RUNNING | 10.1.1.151 (eth0)  |      | CONTAINER | 0         |
+------+---------+--------------------+------+-----------+-----------+

# incus shell c1
[root@c1 ~]# ping 8.8.8.8
```

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=115 time=21.5 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=115 time=21.2 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=115 time=19.8 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=115 time=12.6 ms
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 12.647/17.726/21.464/3.855 ms
[root@c1 ~]#
[root@c1 ~]# exit
logout
```

Now that we know the container is working as expected, let us stop it and delete it.

```
# incus stop c1 ; incus delete c1

# incus list
+------+-------+------+------+------+-----------+
| NAME | STATE | IPV4 | IPV6 | TYPE | SNAPSHOTS |
+------+-------+------+------+------+-----------+
```

We will go ahead and remove the image as well

```
# incus image delete e80fa8f0662e

# incus image list

+-------+-------------+--------+-------------+--------------+------+------+-------------+
| ALIAS | FINGERPRINT | PUBLIC | DESCRIPTION | ARCHITECTURE | TYPE | SIZE | UPLOAD DATE |
+-------+-------------+--------+-------------+--------------+------+------+-------------+
```

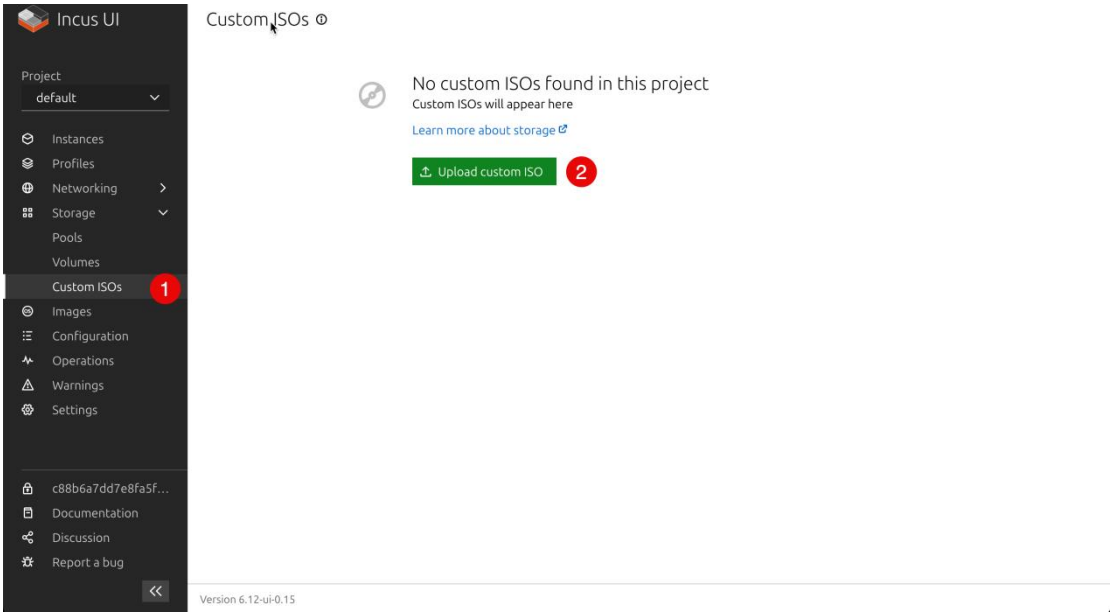
Finally we will take a look at the same directory for one last time and notice that the directory is now empty as we deleted the image.

```
# ls -l /var/lib/incus/storage-pools/shared-images/custom/default_global-volume/
total 0
```

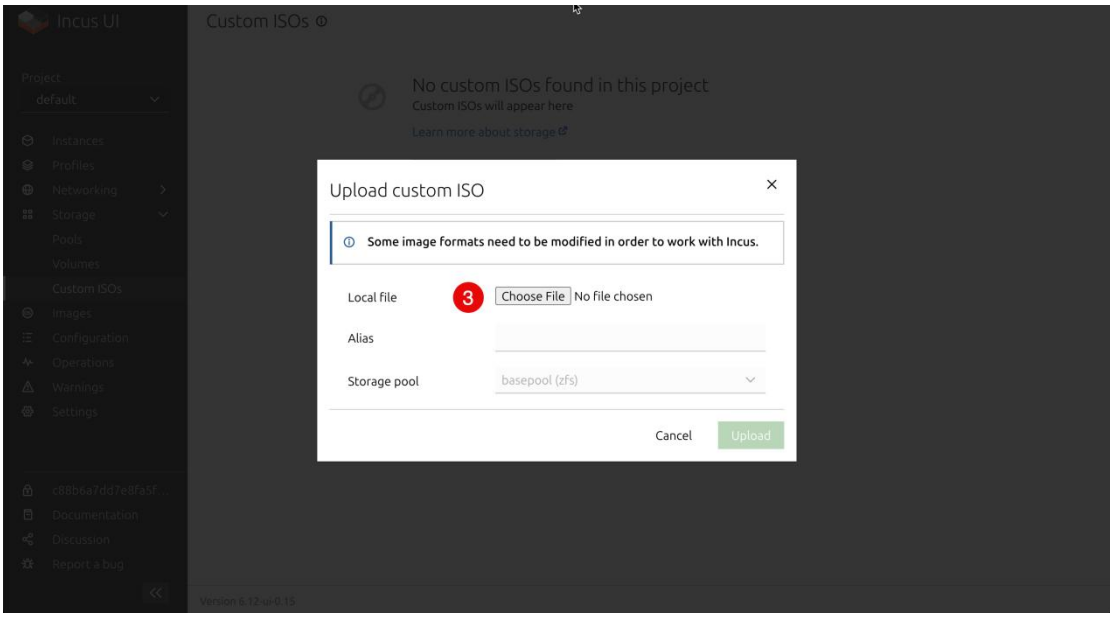
This proves us that the image is stored in the volume (global-volume) that we created before.

Task #14: Uploading an ISO file

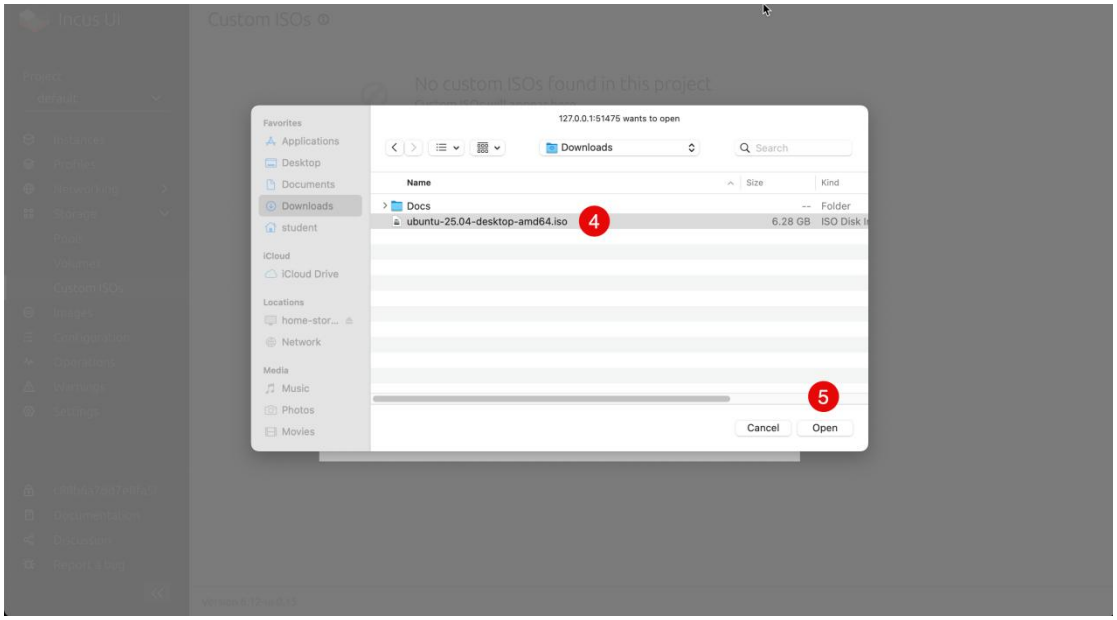
In this task let us look at how to use the incus storage pool 'shared-iso' (backed by zfs pool 'iso-pool') that we created exclusively to store all the ISO files. Let us upload an ISO file to that pool, which is straight forward using web UI.



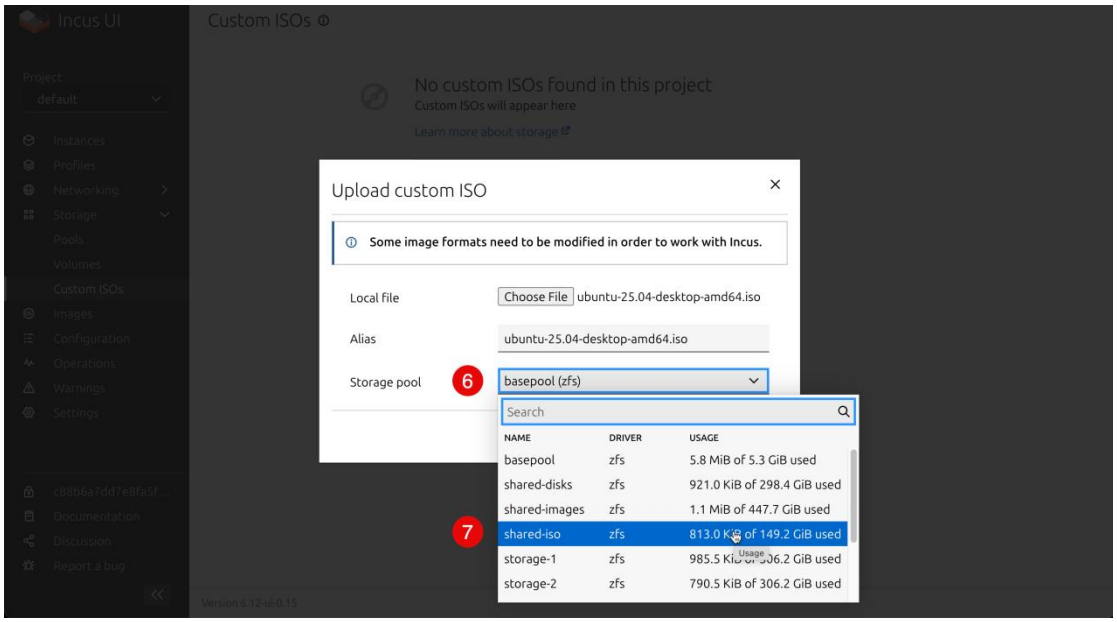
Select the ISO file by clicking on the upload button



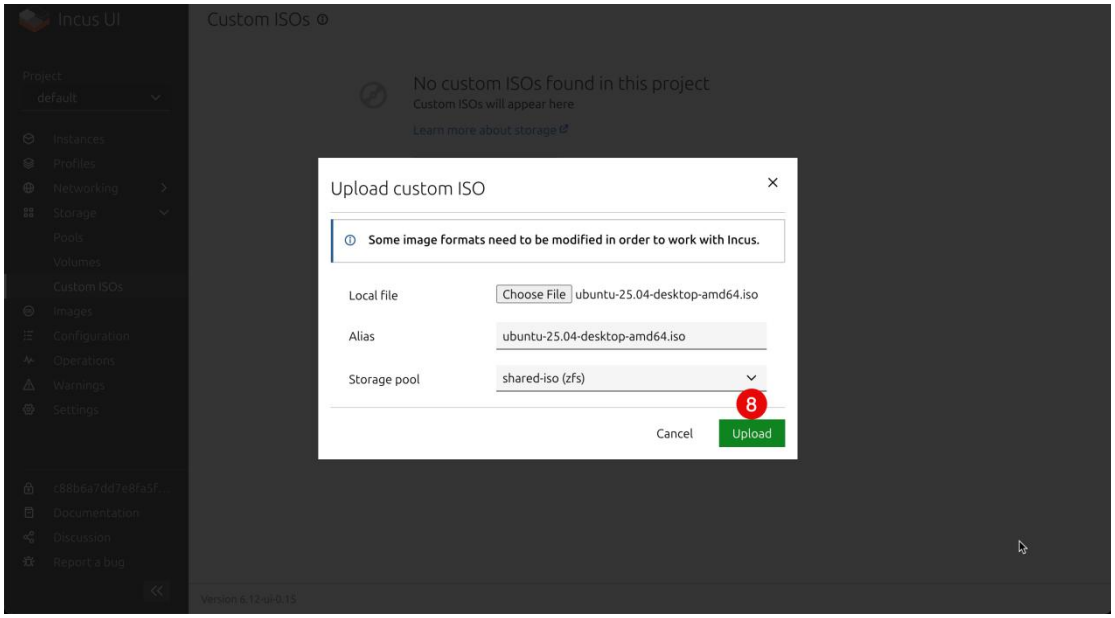
Click Open

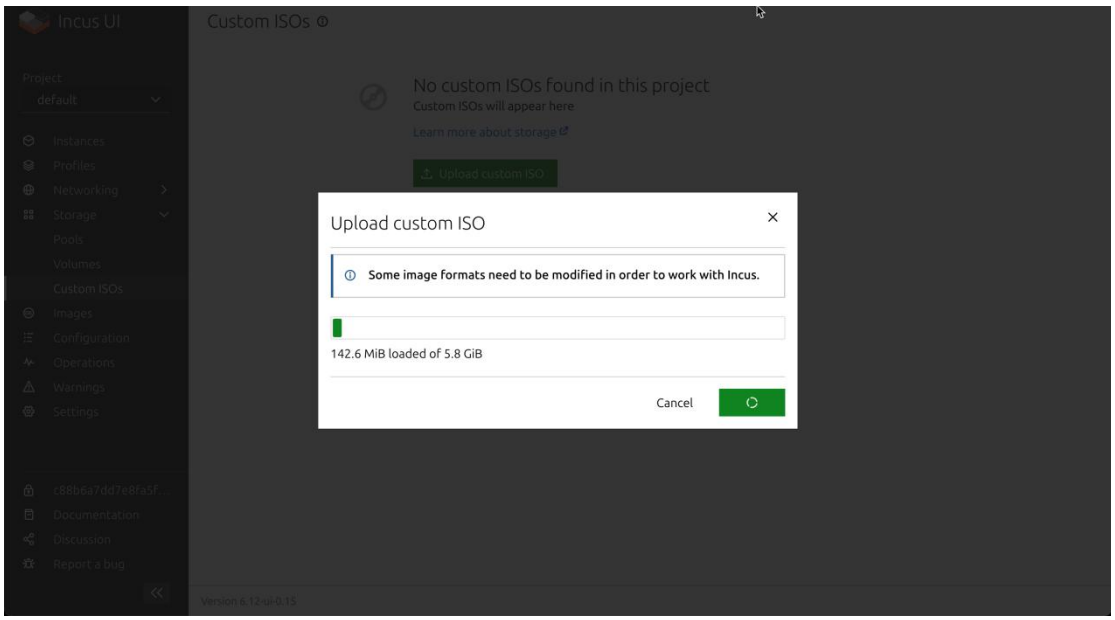


Select ‘shared-iso’ from the dropdown menu under Storage pool

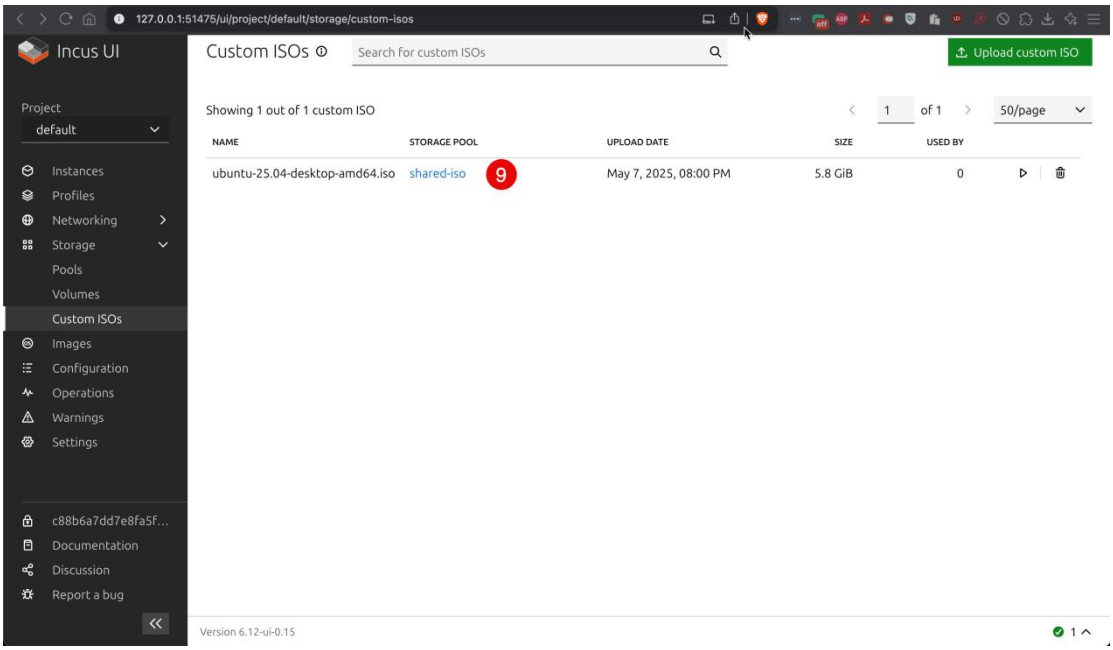


Click Upload

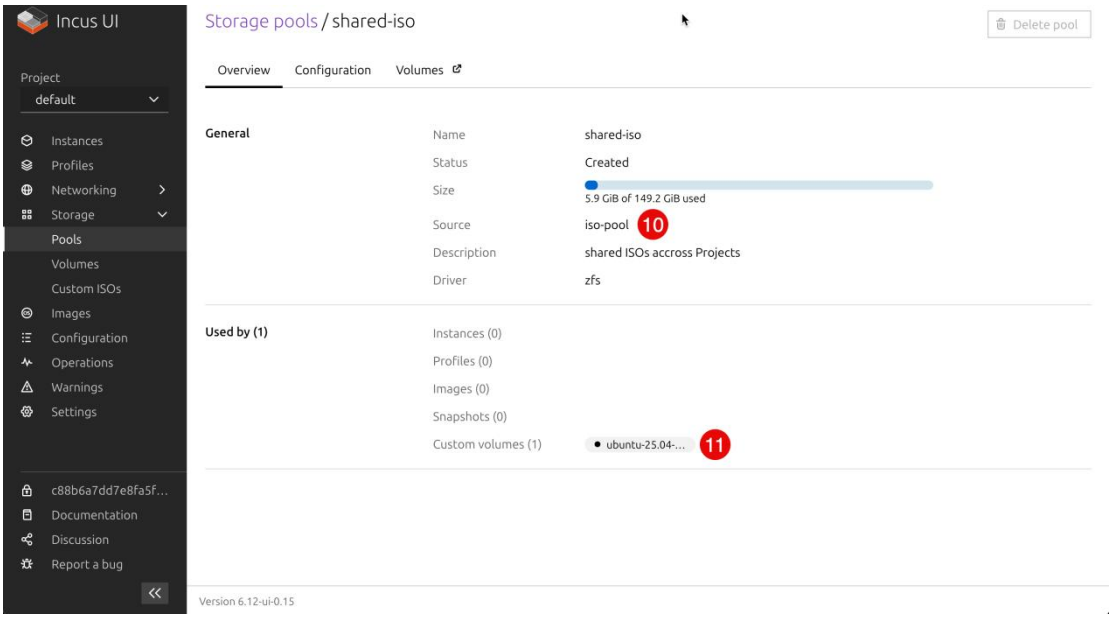




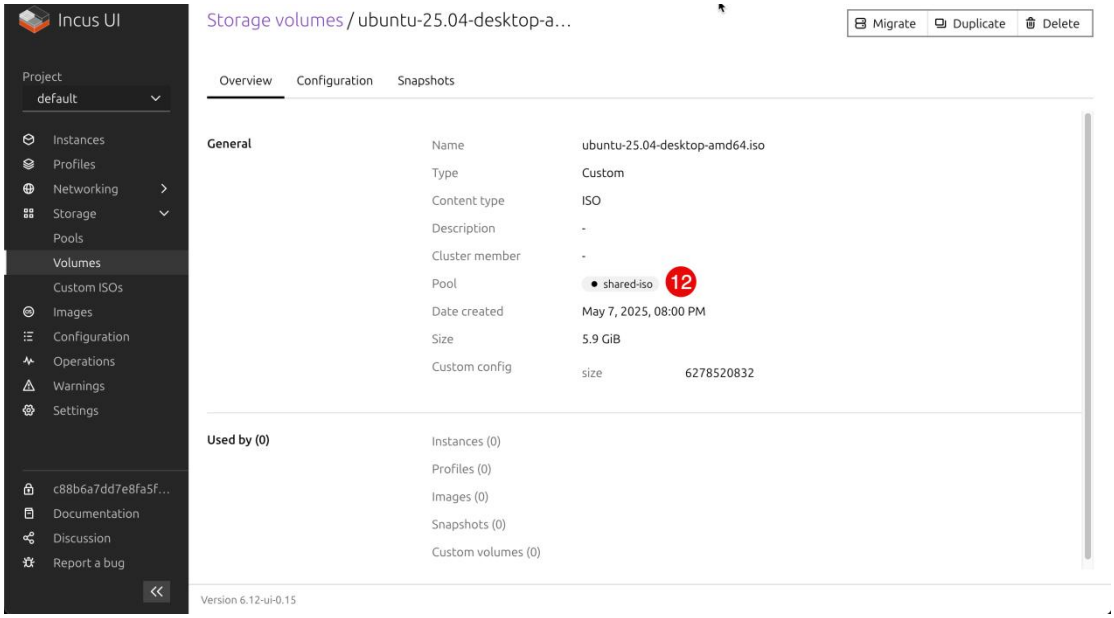
After uploading, the iso file will be listed under ‘shared-iso’ storage pool

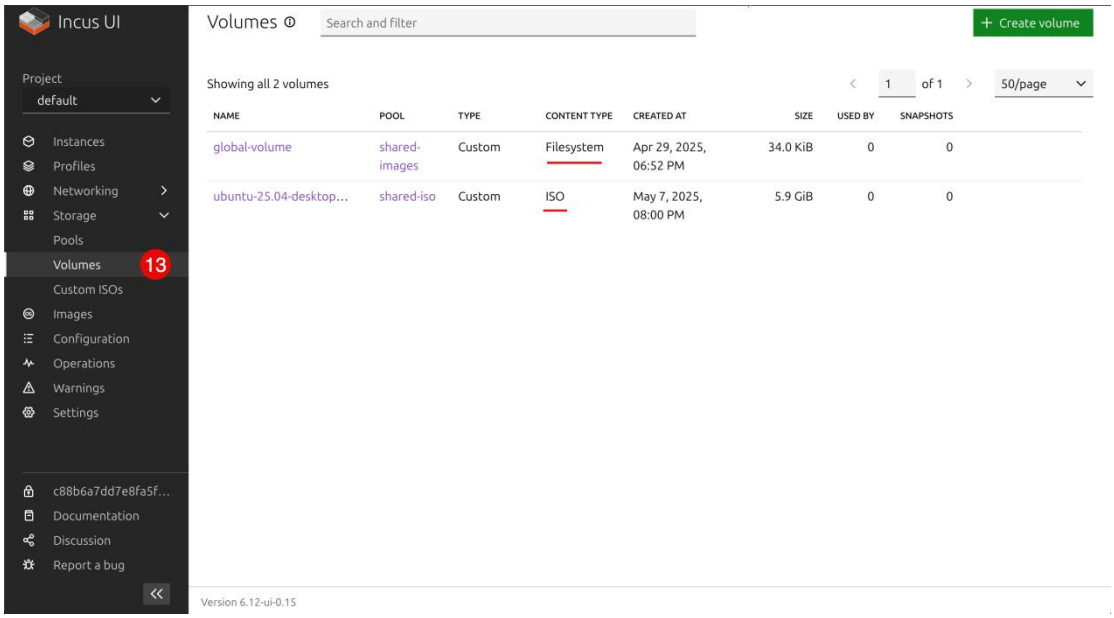


Clicking on ‘shared-iso’ will show where it is located. ie, on the zfs pool named ‘iso-pool’.



It is important to understand that the uloaded iso file (ubuntu25.04-desktop-amd64.iso) file will be listed as a custom volume.


Incus UI
Storage volumes / ubuntu-25.04-desktop-a…
Migrate
Duplicate
Delete
Project
default
Instances
Profiles
Networking
Storage
Pools
Volumes
Custom ISOs
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f…
Documentation
Discussion
Report a bug
Overview
Configuration
Snapshots
General
Name
ubuntu-25.04-desktop-amd64.iso
Type
Custom
Content type
ISO
Description
-
Cluster member
-
Pool
shared-iso
12
Date created
May 7, 2025, 08:00 PM
Size
5.9 GiB
Custom config
size
6278520832
Used by (0)
Instances (0)
Profiles (0)
Images (0)
Snapshots (0)
Custom volumes (0)
Version 6.12-ui-0.15


Incus UI
Volumes
Search and filter
Create volume
Project
default
Showing all 2 volumes
1
of 1
50/page
NAME
POOL
TYPE
CONTENT TYPE
CREATED AT
SIZE
USED BY
SNAPSHOTS
global-volume
shared-images
Custom
Filesystem
Apr 29, 2025, 06:52 PM
34.0 KiB
0
0
ubuntu-25.04-desktop…
shared-iso
Custom
ISO
May 7, 2025, 08:00 PM
5.9 GiB
0
0
Instances
Profiles
Networking
Storage
Pools
Volumes
13
Custom ISOs
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f…
Documentation
Discussion
Report a bug
Version 6.12-ui-0.15

Task #15: ISO file usage

In this task let us work on installing an operating system on a blank virtual machine using the ISO image stored in the storage pool 'shared-iso'.

To begin with, let us create a blank virtual machine and install Ubuntu25.04 on it with the uploaded ISO image.

There are a couple of ways to accomplish this. One way is to use the web UI. The other way is to use a combination of command lines as well as a software named 'virt-viewer'. Both of them can be used from the Mac laptop as we have already taken care of its authentication in task #09.

We have seen using WeUI before. So this time, let us take a look at incus commands and using 'virt-viewer'.

'virt-viewer' is a separate stand alone free software which is a pre-requisite that needs to be installed on (my) Mac laptop. This software is used to acces the console of the virtual machine on which we are planning to install ubuntu25.04 on it

We will begin with using incus command lines. Once the virtual machine is started, we will then go ahead continuing the rest of the installation using virt-viewer.

Let us keep the installation screenshots as minimal as we can or else this task will end up becoming nothing more than a bunch of screenshots.

All these commands are issued on the Mac Laptop.

First step is to create a blank virtual machine with 16GB of RAM, 2 x CPU's, and its root disk of size 10GiB

```
mac% incus init vm-ubuntu25 --empty -c limits.memory=16GiB -c limits.cpu=2 -d root,size=10GiB
--vm
```

```
% incus list
+-------------+---------+------+------+-----------------+-----------+
|   NAME      |  STATE  | IPV4 | IPV6 |     TYPE        | SNAPSHOTS |
+-------------+---------+------+------+-----------------+-----------+
| vm-ubuntu25 | STOPPED |      |      | VIRTUAL-MACHINE | 0         |
+-------------+---------+------+------+-----------------+-----------+
```

Let us configure the newly created virtul machine named vm-ubuntu25 with the ISO image

```
mac% incus config device add vm-ubuntu25 iso disk pool=shared-iso
source=ubuntu-25.04-desktop-amd64.iso boot.priority=10
```

The above command line in plain English:

1) Configure the virtual machine by attaching the ISO image located in the storage pool 'shared-iao' as a new device
2) Name of the ISO file being ubuntu-25.04-desktop-amd64.iso
3) Set the boot priority to 10
4) boot priority is about setting the boot order when powered on. It ensures the virtual machine will boot using the ISO image first

Let us go ahead and start the virtual machine

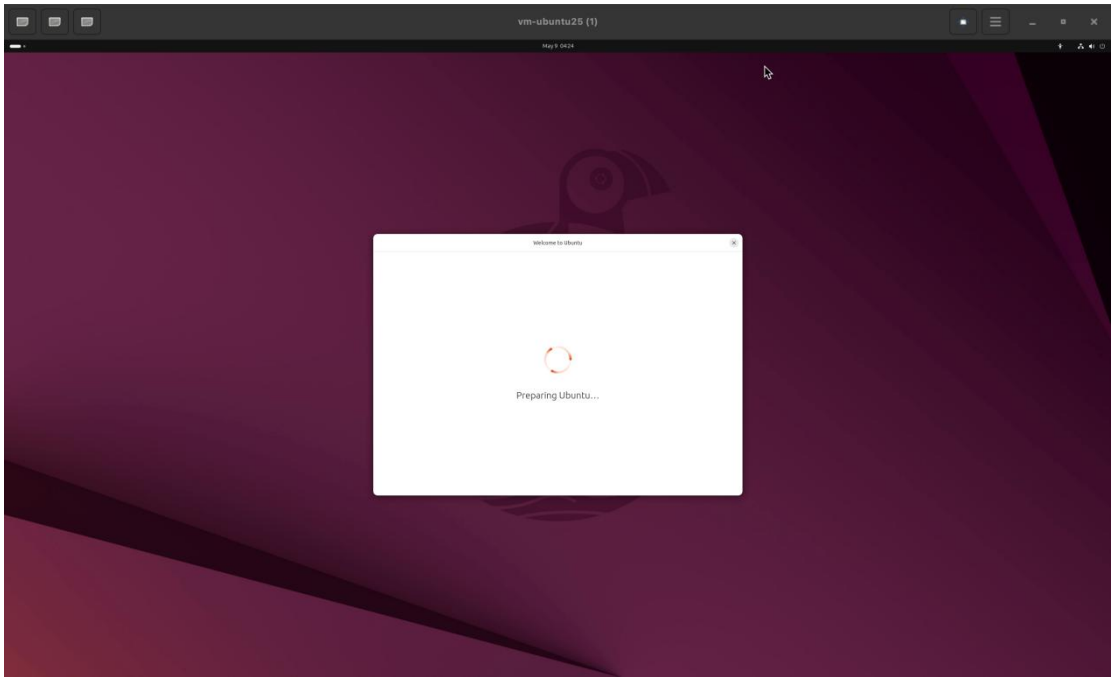
```
mac% incus start vm-ubuntu25
```

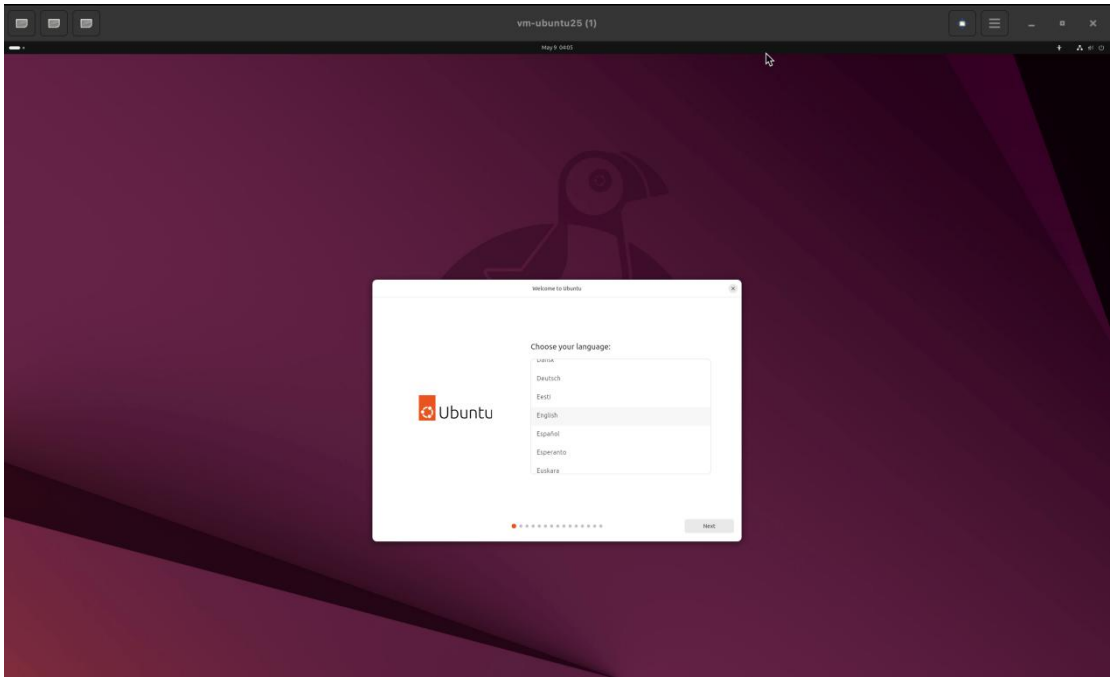
Now that the virtual machine is started, we need to access its console from our Mac Laptop. And here is where the role of virt-viewer comes in. Running the command line below will automatically invoke virt-viewer and gets us the console of the virtual machine (we don't have to run the virt-viewer by ourselves manually)

```
mac% incus console vm-ubuntu25 --type=vga
```

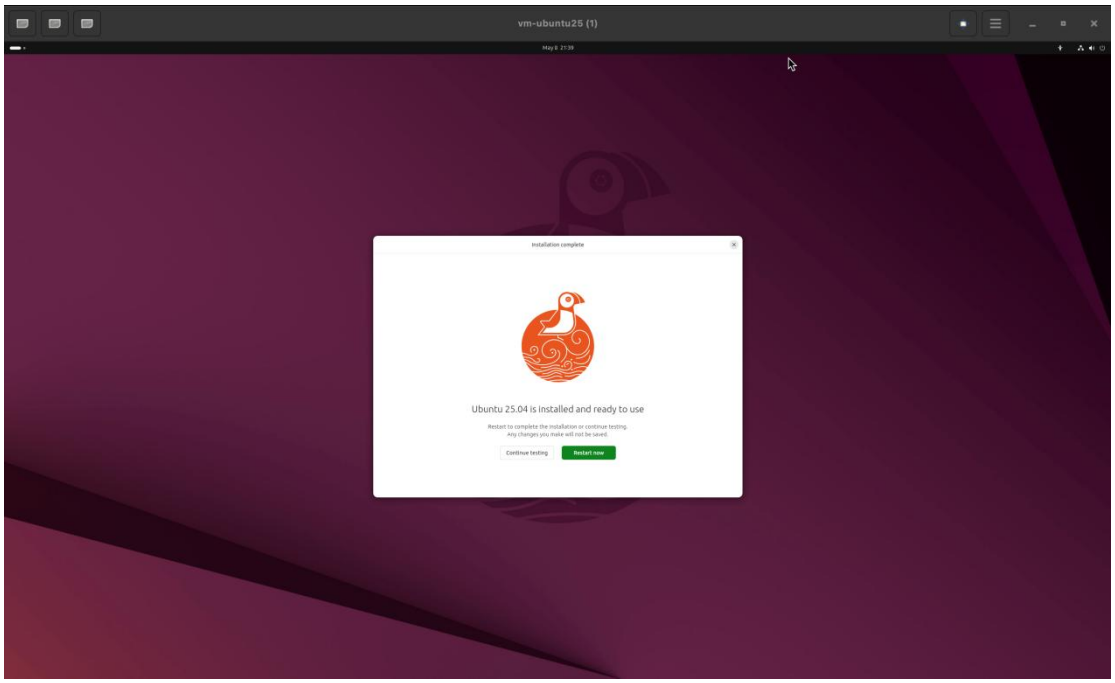
You might see warning messages thrown on the terminal but you will also see virt-viewer instantiated on your laptop.

To begin with you will see the splash screen with the company logo of Zabbly (maintainer of project Incus)

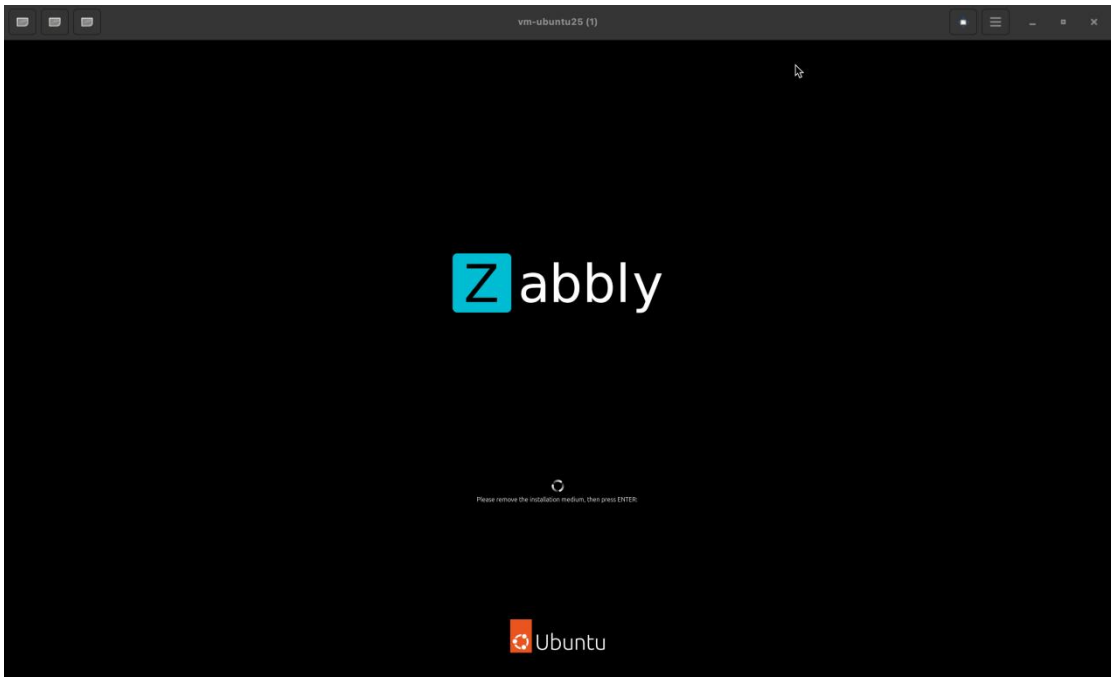




The rest of the screenshots are well known. so let us skip all that and go towards the end



On clicking on the Reboot now button, the virtual machines power cycles and reaches this spot



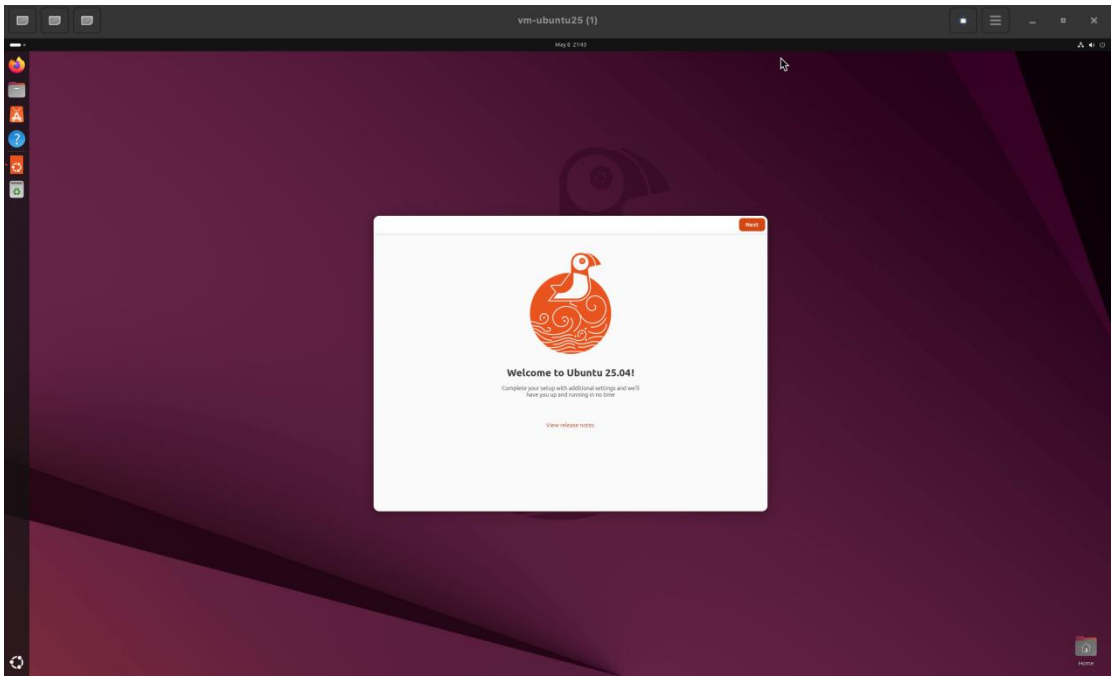
You will notice the message saying “Please remove the installation medium, then press ENTER.

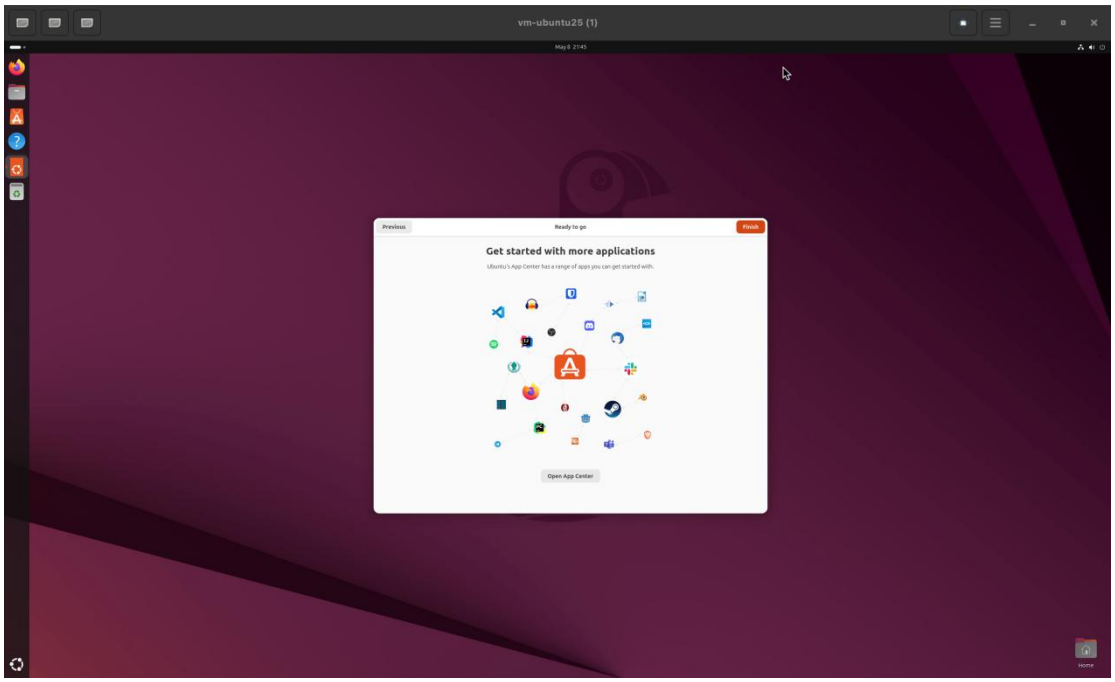
And before pressing the ENTER, run the following incus commands on the Mac laptop

```
mac% incus config device remove vm-ubuntu25 iso
```

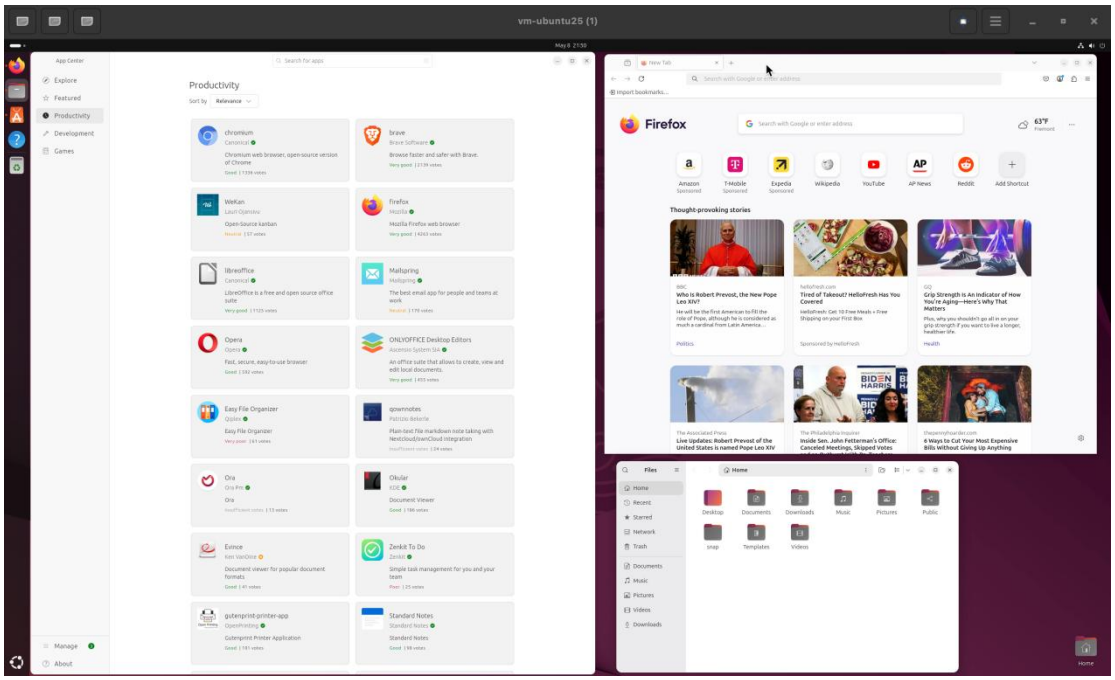
```
mac% incus start vm-ubuntu25
```

You will now see the virtual machine getting power cycled boots from it root disk

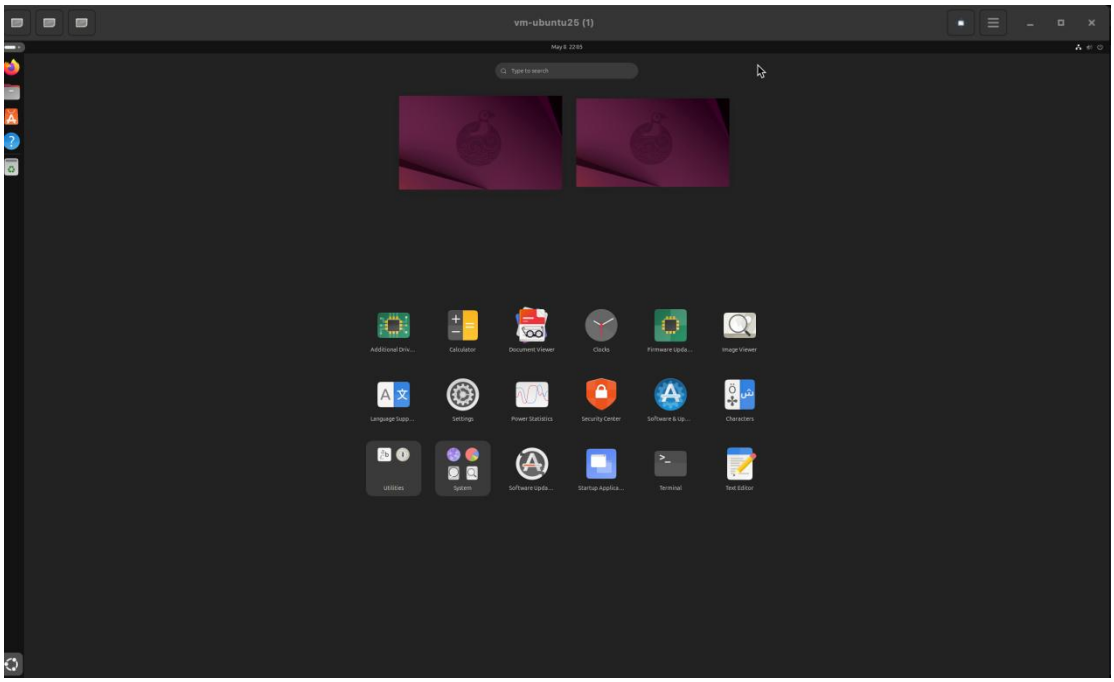




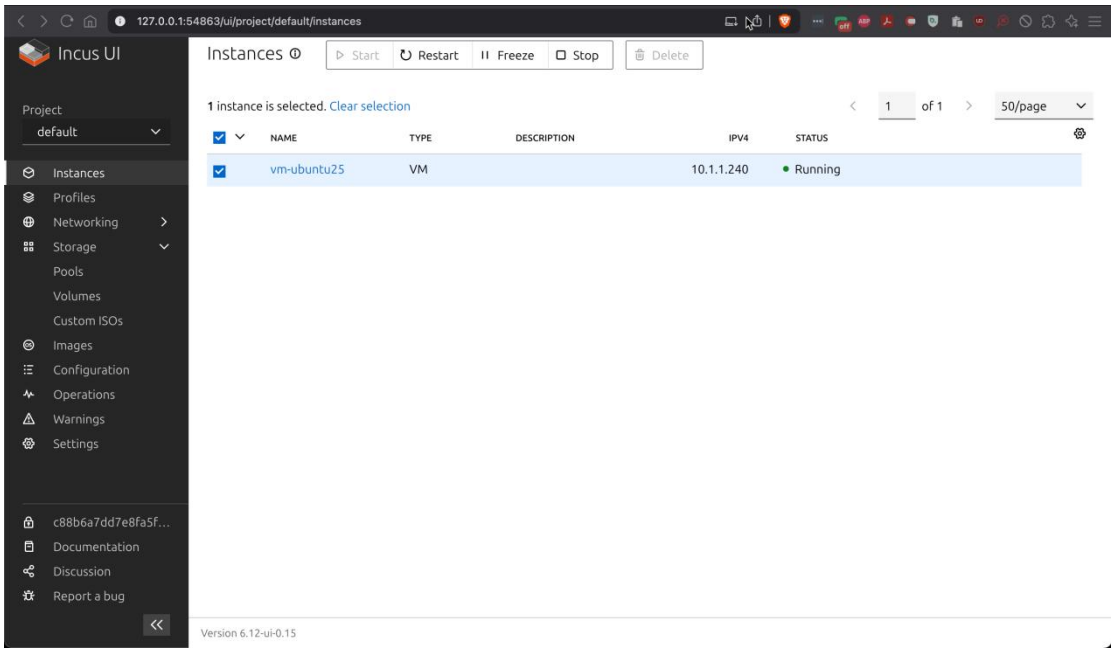
The App store, Firefox browser and File Manager

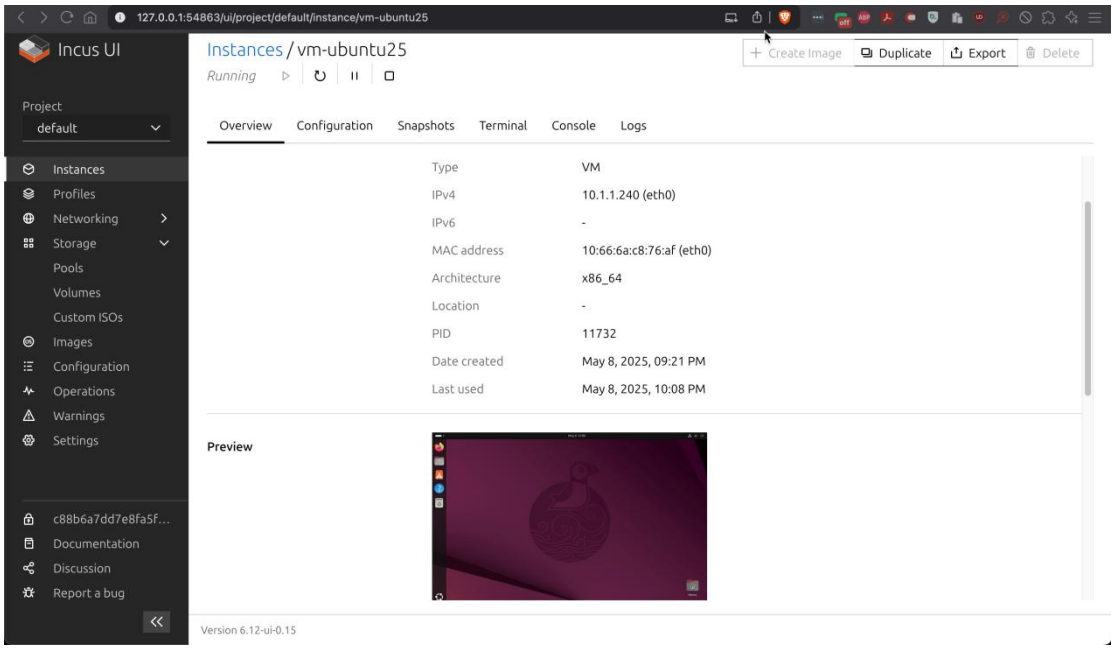


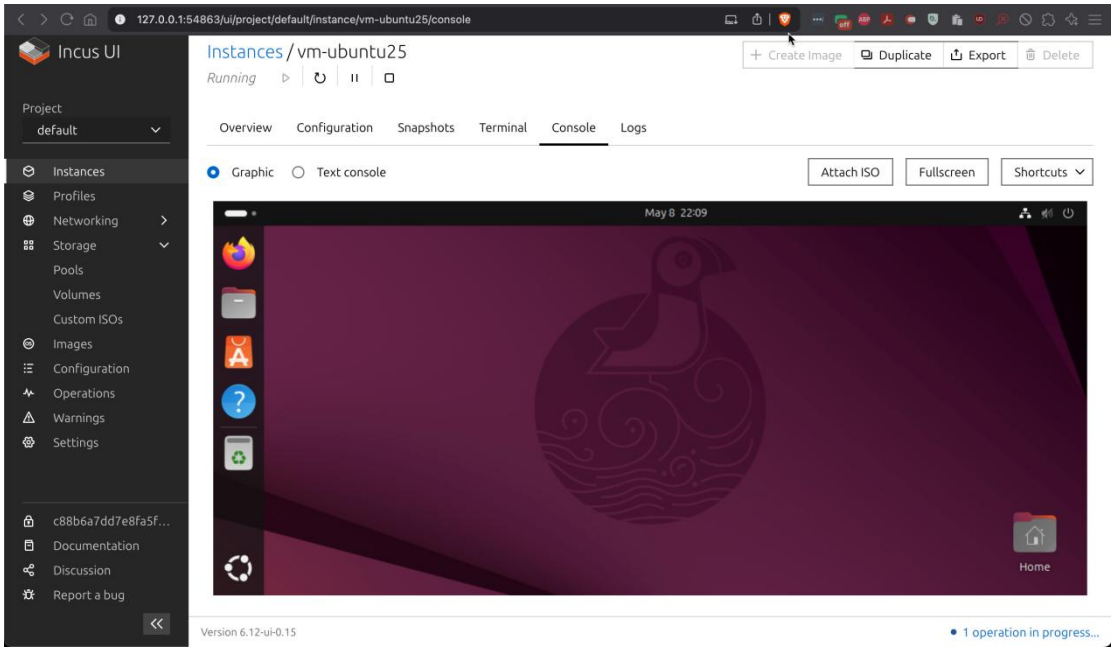
Desktop

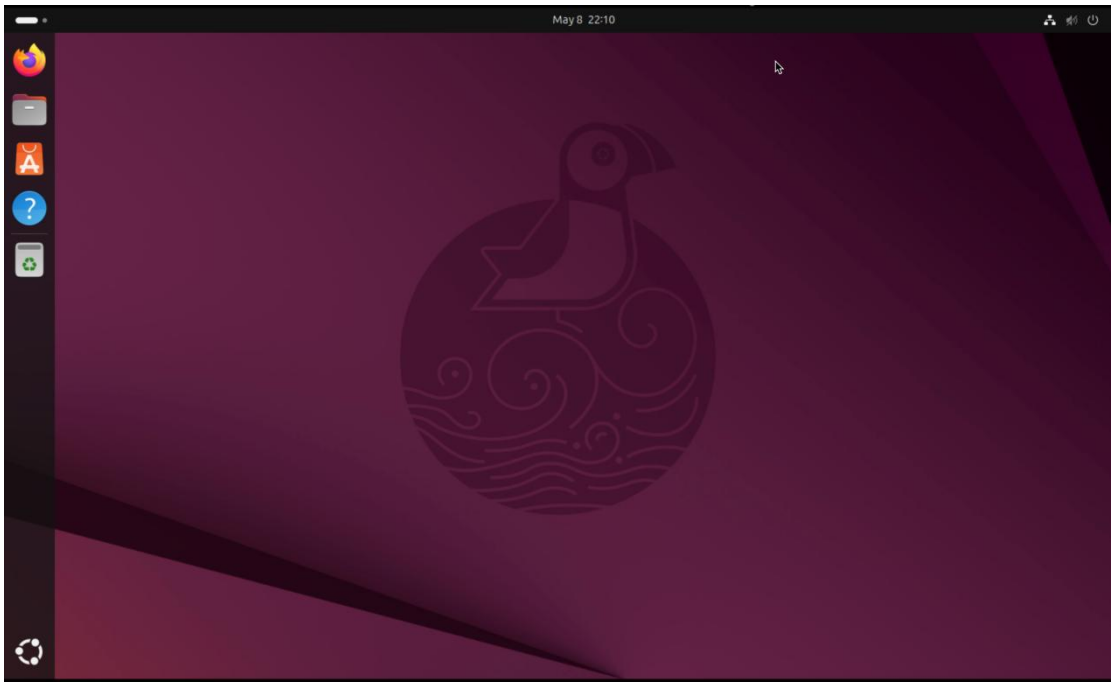


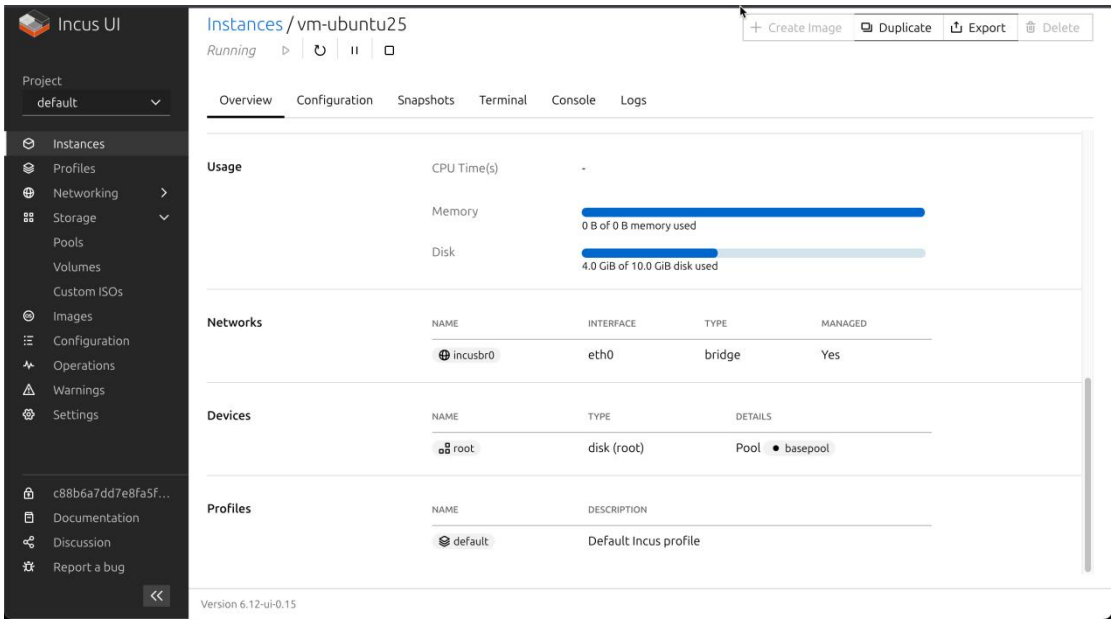
Now that we have access the console of the virual machine via virt-viewer, let us take a quick look at the same virtual machine using web UI




127.0.0.1:54863/ui/project/default/instance/vm-ubuntu25
Incus UI
Instances / vm-ubuntu25
Running
Create Image
Duplicate
Export
Delete
Project
default
Instances
Profiles
Networking
Storage
Pools
Volumes
Custom ISOs
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Overview
Configuration
Snapshots
Terminal
Console
Logs
Type
VM
IPv4
10.1.1.240 (eth0)
IPv6
-
MAC address
10:66:6a:c8:76:af (eth0)
Architecture
x86_64
Location
-
PID
11732
Date created
May 8, 2025, 09:21 PM
Last used
May 8, 2025, 10:08 PM
Preview
Version 6.12-ui-0.15


127.0.0.1:54863/ui/project/default/instance/vm-ubuntu25/console
Incus UI
Instances / vm-ubuntu25
Running
Create Image
Duplicate
Export
Delete
Project
default
Instances
Profiles
Networking
Storage
Pools
Volumes
Custom ISOs
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Overview
Configuration
Snapshots
Terminal
Console
Logs
Graphic
Text console
Attach ISO
Fullscreen
Shortcuts
May 8 22:09
Home
Version 6.12-ui-0.15
1 operation in progress...

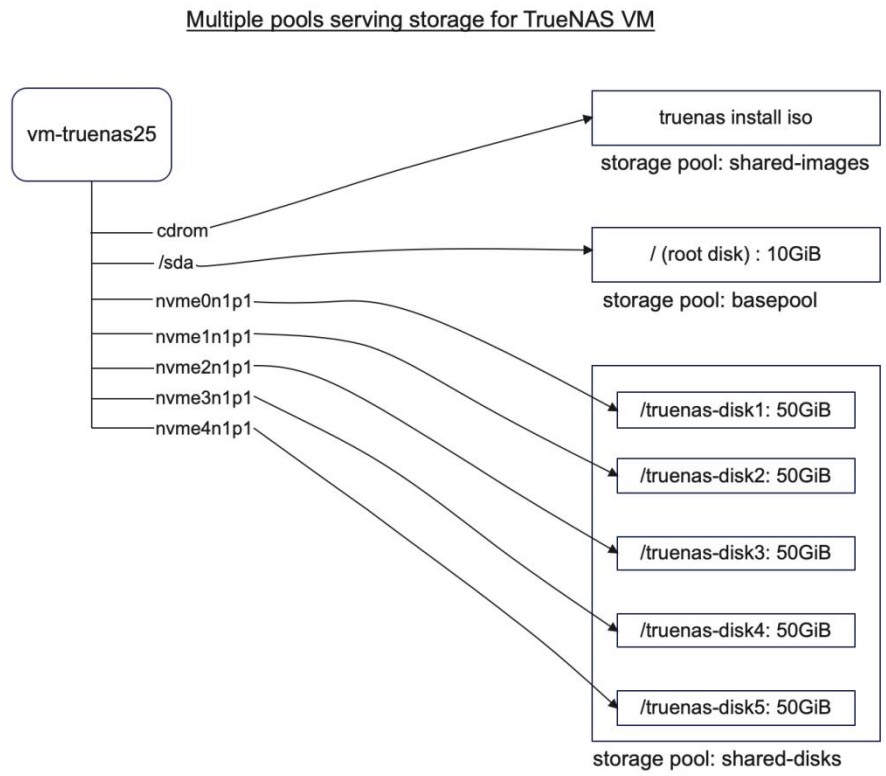

May 8 22:10


Incus UI
Project
default
Instances
Profiles
Networking
Storage
Pools
Volumes
Custom ISOs
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Version 6.12-ui-0.15
Instances / vm-ubuntu25
Running
Create Image
Duplicate
Export
Delete
Overview
Configuration
Snapshots
Terminal
Console
Logs
Usage
CPU Time(s)
-
Memory
0 B of 0 B memory used
Disk
4.0 GiB of 10.0 GiB disk used
Networks
NAME
INTERFACE
TYPE
MANAGED
incusbr0
eth0
bridge
Yes
Devices
NAME
TYPE
DETAILS
root
disk (root)
Pool
basepool
Profiles
NAME
DESCRIPTION
default
Default Incus profile

Task #16: Using custom storage pools and volumes

To understand how to use multiple storage pools, let us create a single virtual machine that spans across multiple storage pools. For this purpose, let us create a blank virtual machine with its root disk stored in one storage pool (basepool), all its data disks gets stored in a second storage pool (shared-disks), and finally installing TrueNAS on it by attaching an ISO file to it, which will be stored in the third storge pool (shared-iso).

This task is based on a video posted by Stephane Graber (incus lead) while evaluating TrueNAS Community Edition.



To begin with, let us import TrueNAS iso file to the storage pool that we specifically created to store all iso fies (ie, shared-iso).

These commands can be carried out from the Mac Laptop

```
mac% incus storage volume import shared-iso ~/Downloads/TrueNAS-SCALE-25.04.0.iso
truenas-install --type=iso
```

Let us check make sure it did land in the storage pool 'shared-iso'. You can see it below listed as truenas-install (under 'used_by')

```
mac% incus storage show shared-iso
config:
  source: iso-pool
  volatile.initial_source: iso-pool
  zfs.pool_name: iso-pool
description: shared ISOs accross Projects
name: shared-iso
driver: zfs
used_by:
- /1.0/storage-pools/shared-iso/volumes/custom/truenas-install
- /1.0/storage-pools/shared-iso/volumes/custom/ubuntu-25.04-desktop-amd64.iso
status: Created
locations:
- none
```

Create an empty virtual machine named vm-truenas25

```
mac% incus create vm-truenas25 --vm --empty -c limits.cpu=4 -c limits.memory=8GiB -c
security.secureboot=false

mac% incus list
+--------------+---------+------+------+-----------------+-----------+
|    NAME      |  STATE  | IPV4 | IPV6 |     TYPE        | SNAPSHOTS |
+--------------+---------+------+------+-----------------+-----------+
| vm-truenas25 | STOPPED |      |      | VIRTUAL-MACHINE | 0         |
+--------------+---------+------+------+-----------------+-----------+
```

The root disk of this newly created virtual machine will be stored in storage pool ‘basepool’

```
mac% incus storage show basepool
config:
  source: basepool
  volatile.initial_source: basepool
  zfs.pool_name: basepool
description: passed to incus admin init
name: basepool
driver: zfs
used_by:
- /1.0/instances/vm-truenas25
- /1.0/profiles/default
status: Created
locations:
- none
```

Let us create 5 volumes (of block type and size 50GiB each) on the incus storage pool 'shared-disks'. These will be attached as data disks to the newly created virtual machine. Later we use all of them to create a RAIDZ1 volume combiing all of them.

```
mac% incus storage show shared-disks
config:
  source: disks-pool
  volatile.initial_source: disks-pool
  zfs.pool_name: disks-pool
description: shared disks accross Projects
name: shared-disks
driver: zfs
used_by: []
status: Created
locations:
- none

mac% incus storage volume create shared-disks truenas-disk1 size=50GiB --type=block

mac% incus storage volume create shared-disks truenas-disk2 size=50GiB --type=block

mac% incus storage volume create shared-disks truenas-disk3 size=50GiB --type=block

mac% incus storage volume create shared-disks truenas-disk4 size=50GiB --type=block

mac% incus storage volume create shared-disks truenas-disk5 size=50GiB --type=block

mac% incus storage show shared-disks
config:
  source: disks-pool
  volatile.initial_source: disks-pool
  zfs.pool_name: disks-pool
description: shared disks accross Projects
name: shared-disks
driver: zfs
used_by:
- /1.0/storage-pools/shared-disks/volumes/custom/truenas-disk1
- /1.0/storage-pools/shared-disks/volumes/custom/truenas-disk2
- /1.0/storage-pools/shared-disks/volumes/custom/truenas-disk3
- /1.0/storage-pools/shared-disks/volumes/custom/truenas-disk4
- /1.0/storage-pools/shared-disks/volumes/custom/truenas-disk5
status: Created
locations:
- none
```

Let us configure the virtual machine vm-truenas25 with all the volumes that we created so far, starting with volume created out of the tuenas iso file that resides in storage pool 'shared-iso'

```
mac% incus config device add vm-truenas25 install disk pool=shared-iso
source=truenas-install boot.priority=10
```

Now let us add all the 5 disks that reside in storage pool 'shared-disks'

```
mac% incus config device add vm-truenas25 disk1 disk pool=shared-disks source=truenas-disk1
io.bus=nvme

mac% incus config device add vm-truenas25 disk2 disk pool=shared-disks source=truenas-disk2
io.bus=nvme

mac% incus config device add vm-truenas25 disk3 disk pool=shared-disks source=truenas-disk3
io.bus=nvme

mac% incus config device add vm-truenas25 disk4 disk pool=shared-disks source=truenas-disk4
io.bus=nvme

mac% incus config device add vm-truenas25 disk5 disk pool=shared-disks source=truenas-disk5
io.bus=nvme

mac% incus config show --expanded vm-truenas25
architecture: x86_64
config:
  limits.cpu: "4"
  limits.memory: 8GiB
  security.secureboot: "false"
  volatile.apply_template: create
  volatile.cloud-init.instance-id: 298a1379-bb19-4687-b96d-c782a5bfafc7
  volatile.eth0.hwaddr: 10:66:6a:bd:0a:cb
  volatile.uuid: 4f4476c1-5c37-4054-a498-7969cc3ea8f1
  volatile.uuid.generation: 4f4476c1-5c37-4054-a498-7969cc3ea8f1
devices:
  disk1:
    io.bus: nvme
    pool: shared-disks
    source: truenas-disk1
    type: disk
  disk2:
    io.bus: nvme
    pool: shared-disks
```

```
    source: truenas-disk2
    type: disk
  disk3:
    io.bus: nvme
    pool: shared-disks
    source: truenas-disk3
    type: disk
  disk4:
    io.bus: nvme
    pool: shared-disks
    source: truenas-disk4
    type: disk
  disk5:
    io.bus: nvme
    pool: shared-disks
    source: truenas-disk5
    type: disk
  eth0:
    name: eth0
    network: incusbr0
    type: nic
  install:
    boot.priority: "10"
    pool: shared-iso
    source: truenas-install
    type: disk
  root:
    path: /
    pool: basepool
    type: disk
ephemeral: false
profiles:
- default
stateful: false
description: ""

mac% incus list
+--------------+---------+---------------------+------+-----------------+-----------+
|    NAME      |  STATE  |        IPV4         | IPV6 |      TYPE       | SNAPSHOTS |
+--------------+---------+---------------------+------+-----------------+-----------+
| vm-truenas25 | RUNNING | 10.1.1.115 (enp5s0) |      | VIRTUAL-MACHINE |  0        |
+--------------+---------+---------------------+------+-----------------+-----------+
```

Now that the virtual machine is up and running, let us access it's shell and run a few commands

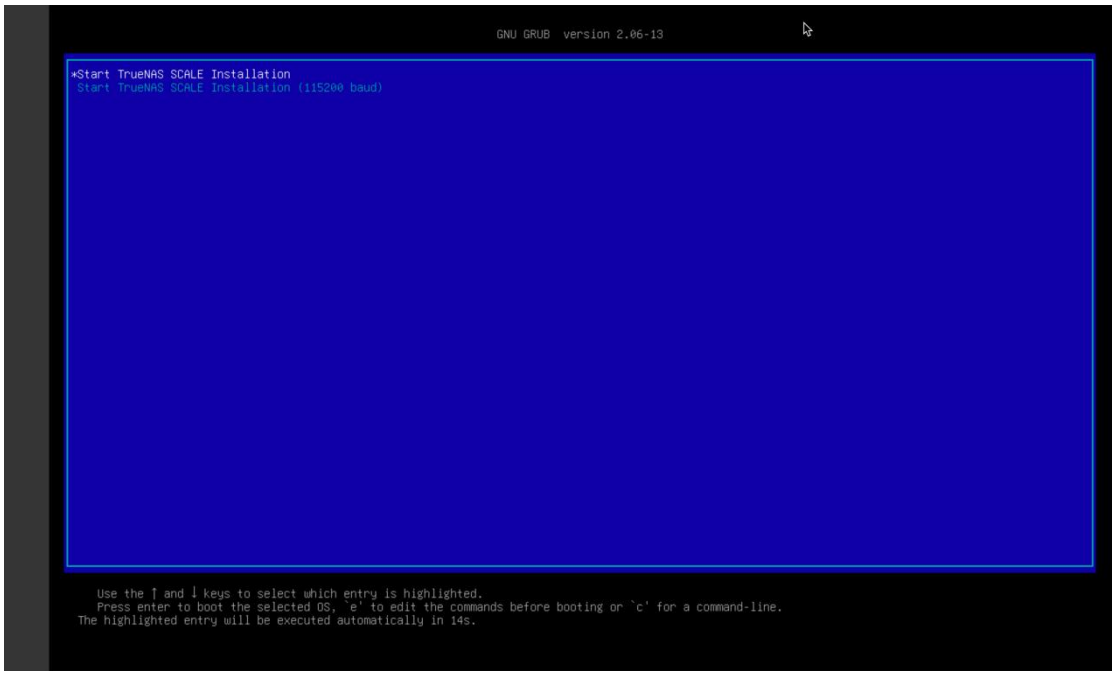
as a sanity check. And also verify the networking setup by installing traceroute command and reach out to zabbly.com

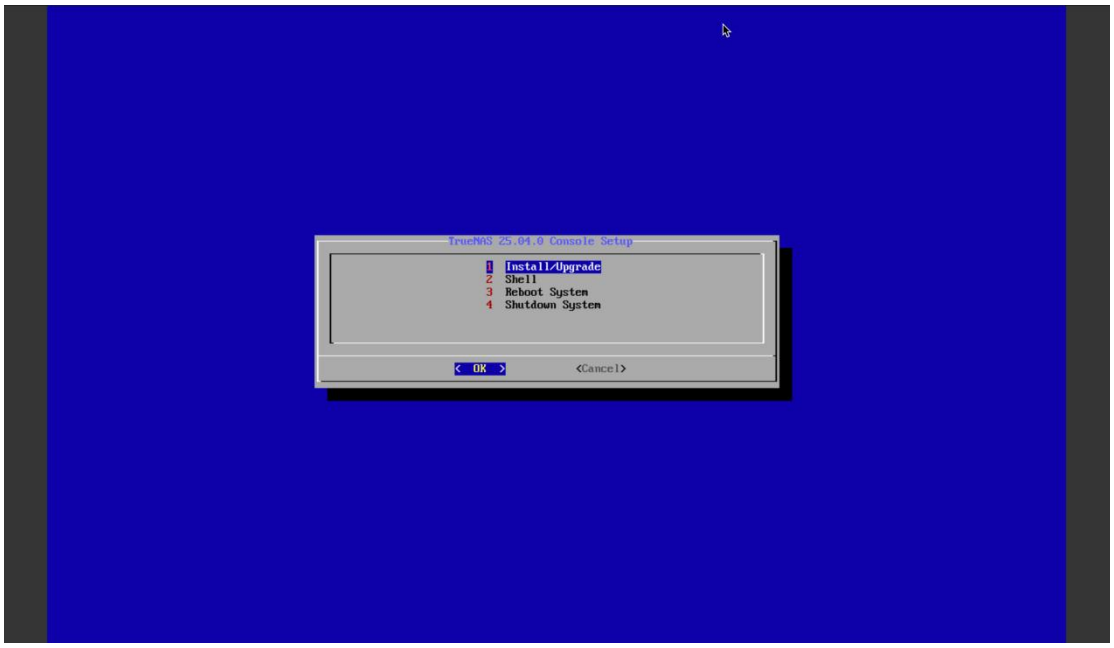
```
mac% incus shell vm-truenas25
root@truenas[~]# uname -a
Linux truenas 6.12.15-production+truenas #1 SMP PREEMPT_DYNAMIC Tue Apr 15 20:07:13 UTC 2025
x86_64 GNU/Linux
root@truenas[~]#
root@truenas[~]# cat /etc/os-release
PRETTY_NAME="Debian GNU/Linux 12 (bookworm)"
NAME="Debian GNU/Linux"
VERSION_ID="12"
VERSION="12 (bookworm)"
VERSION_CODENAME=bookworm
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"

root@truenas[~]# lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda       8:0    0   10G  0 disk
|-sda1    8:1    0    1M  0 part
|-sda2    8:2    0  512M  0 part
`-sda3    8:3    0  9.5G  0 part
nvme0n1 259:0    0   50G  0 disk
nvme1n1 259:1    0   50G  0 disk
nvme2n1 259:2    0   50G  0 disk
nvme3n1 259:3    0   50G  0 disk
nvme4n1 259:4    0   50G  0 disk
root@truenas[~]#
root@truenas[~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp5s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen
1000
    link/ether 10:66:6a:bd:0a:cb brd ff:ff:ff:ff:ff:ff
    inet 10.1.1.115/24 brd 10.1.1.255 scope global dynamic enp5s0
       valid_lft 2859sec preferred_lft 2859sec
    inet6 fe80::1266:6aff:febd:acb/64 scope link
```

```
      valid_lft forever preferred_lft forever

root@truenas[~]# ping zabbly.com
PING zabbly.com (45.45.148.7) 56(84) bytes of data.
64 bytes from rproxy.dcmtl.stgraber.org (45.45.148.7): icmp_seq=1 ttl=49 time=84.4 ms
64 bytes from rproxy.dcmtl.stgraber.org (45.45.148.7): icmp_seq=2 ttl=49 time=85.6 ms
64 bytes from rproxy.dcmtl.stgraber.org (45.45.148.7): icmp_seq=3 ttl=49 time=80.4 ms
^C
--- zabbly.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 80.444/83.474/85.604/2.200 ms
root@truenas[~]# exit
```

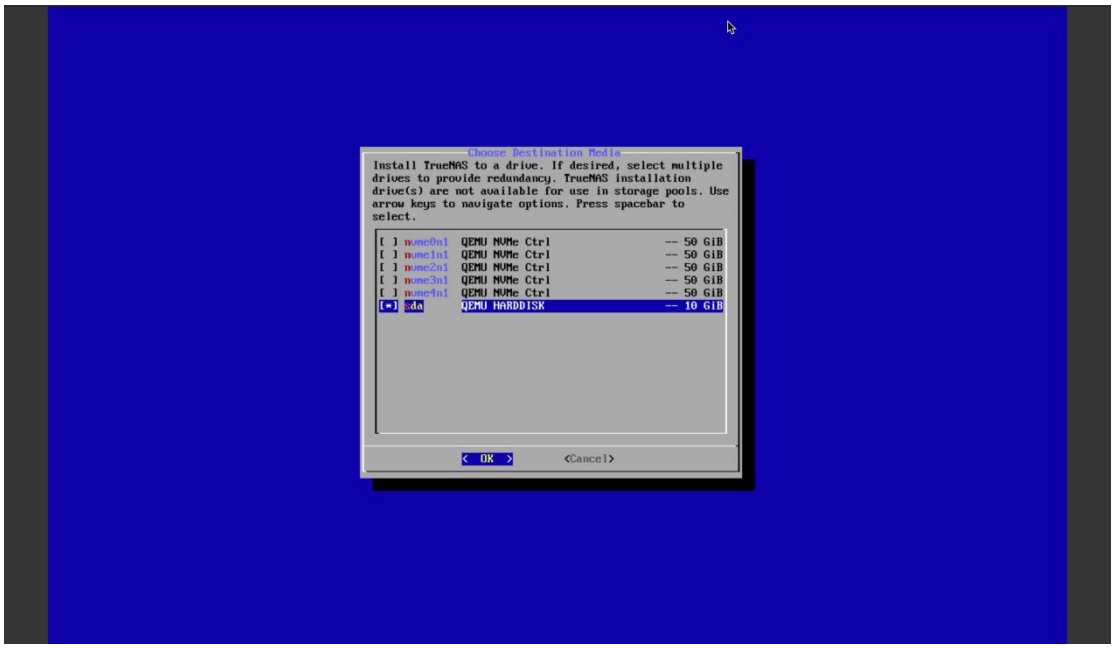
let us connect to the console of the virtual machine vm-truenas25 and continue with the installation of TrueNAS Community Edition.

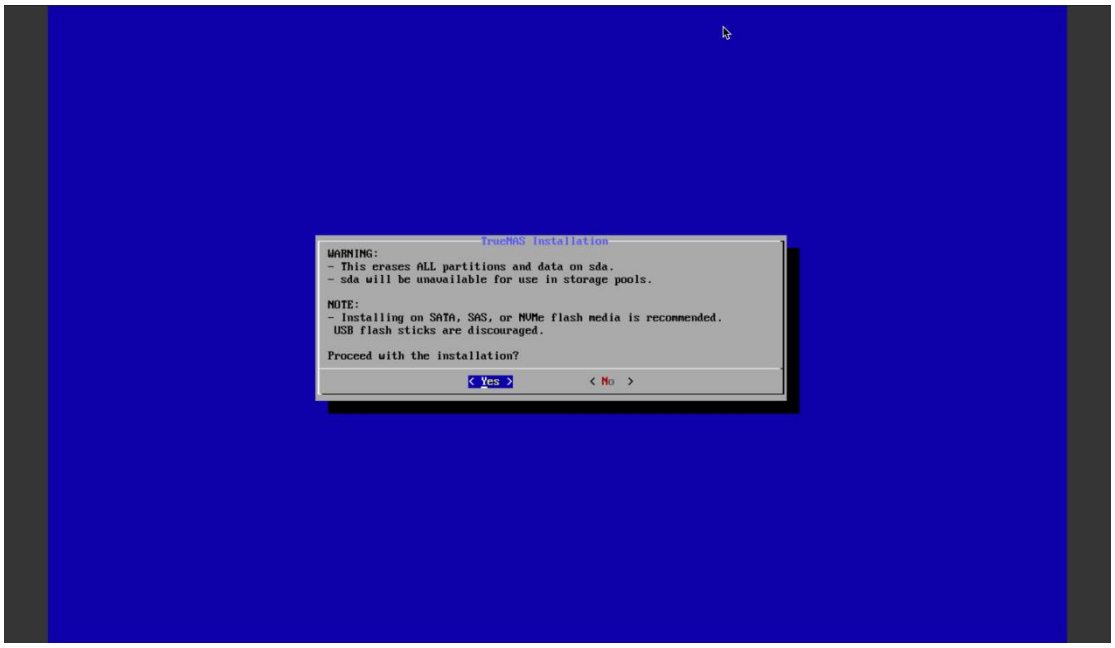
```
mac% incus start vm-truenas25 --console=vga
```

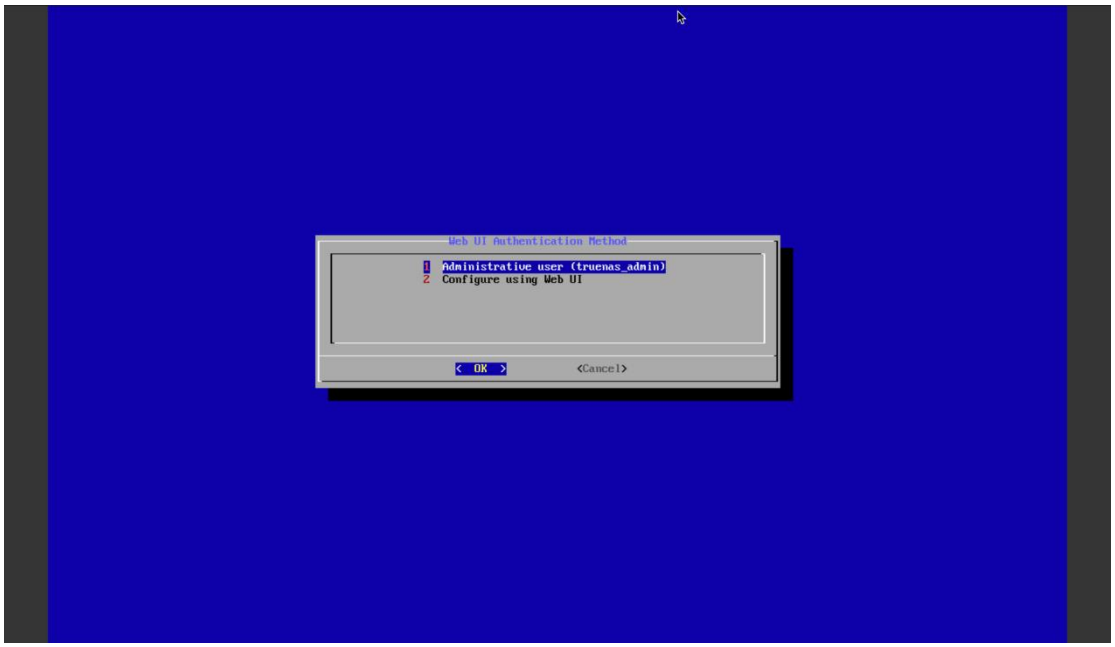


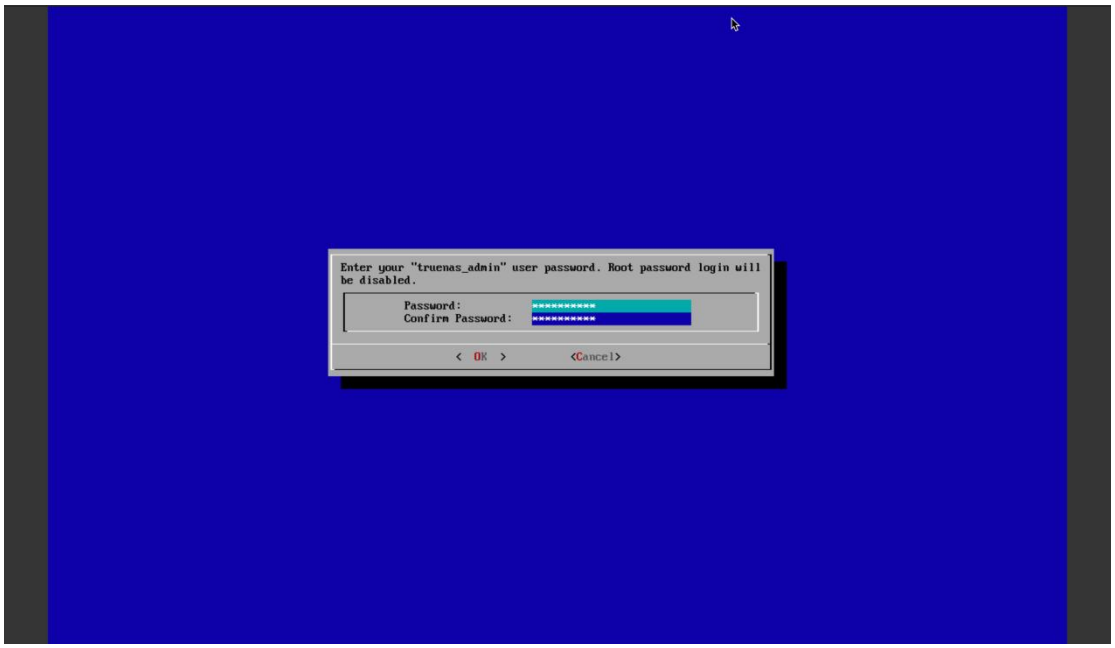


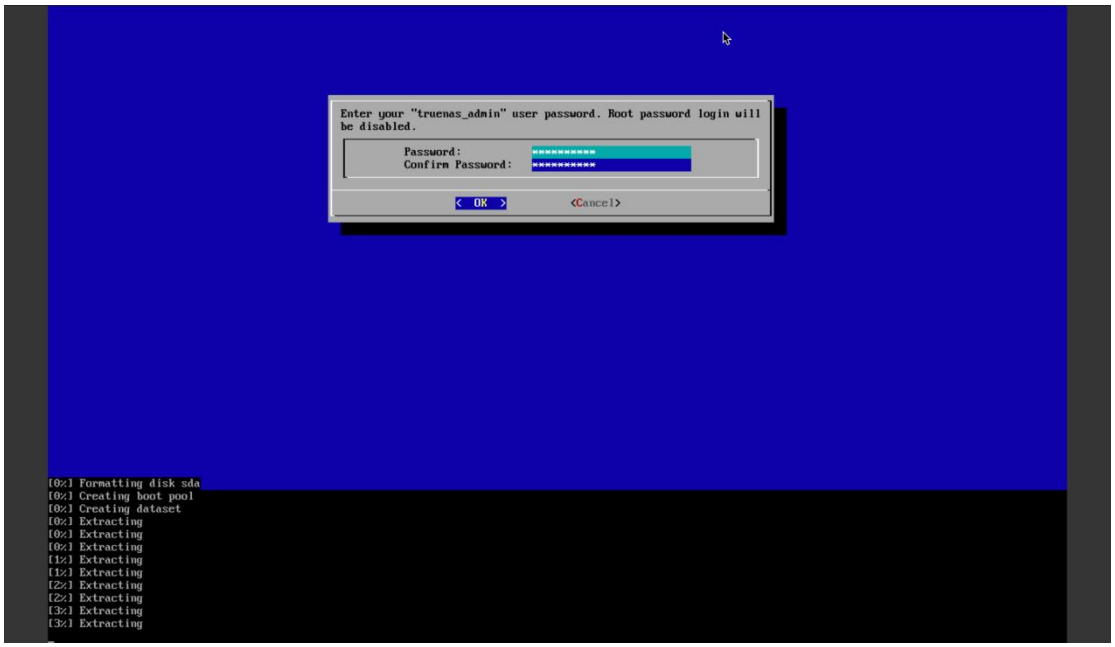
Let us select ‘sda’ as the root disk

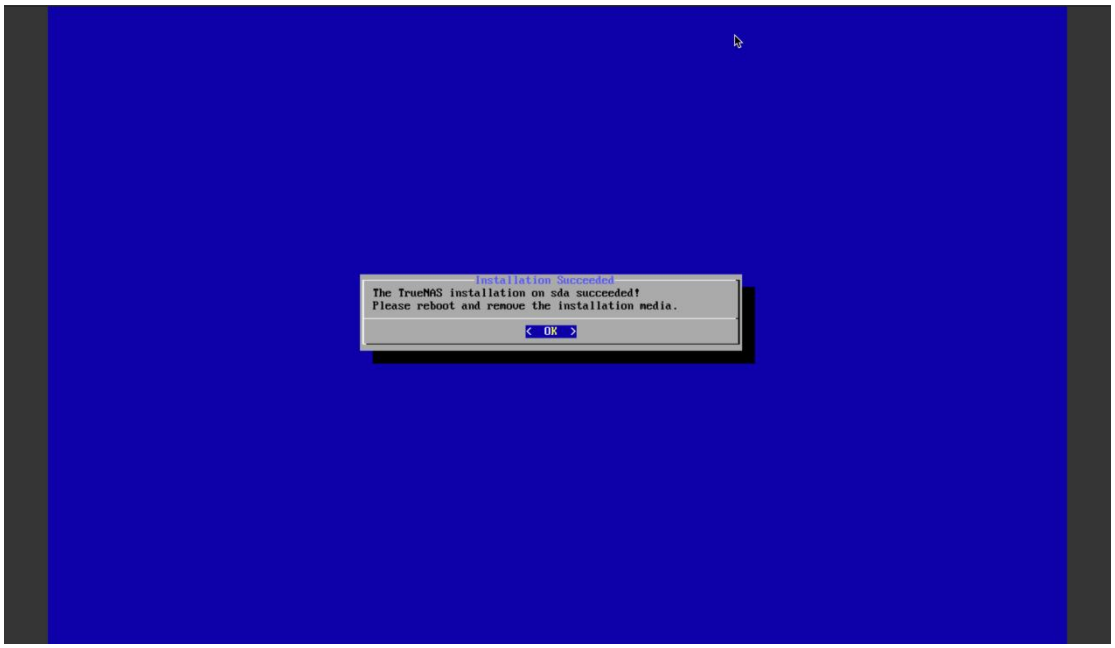



TrueNAS Installation
WARNING:
- This erases ALL partitions and data on sda.
- sda will be unavailable for use in storage pools.
NOTE:
- Installing on SATA, SAS, or NVMe flash media is recommended.
USB flash sticks are discouraged.
Proceed with the installation?
< Yes >
< No >


Web UI Authentication Method
1 Administrative user (truenas_admin)
2 Configure using Web UI
< OK >
<Cancel>

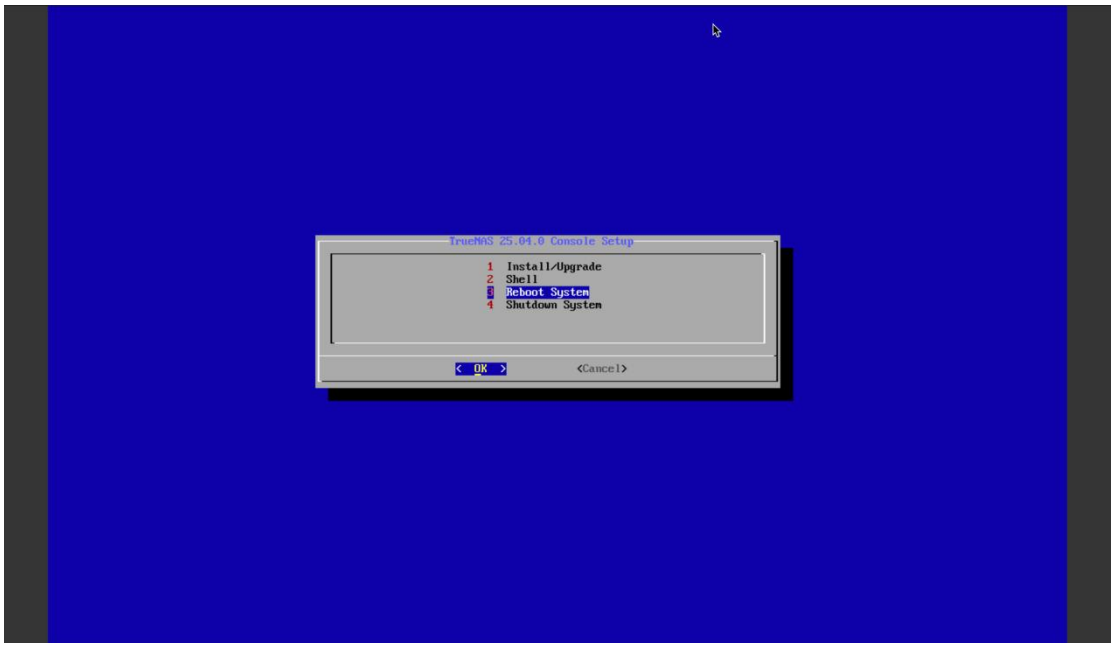

Enter your "truenas_admin" user password. Root password login will be disabled.
Password:
Confirm Password:
< OK >
<Cancel>


Enter your "truenas_admin" user password. Root password login will be disabled.
Password:
Confirm Password:
< OK >
<Cancel>
[0%] Formatting disk sda
[0%] Creating boot pool
[0%] Creating dataset
[0%] Extracting
[0%] Extracting
[0%] Extracting
[1%] Extracting
[1%] Extracting
[2%] Extracting
[2%] Extracting
[3%] Extracting
[3%] Extracting

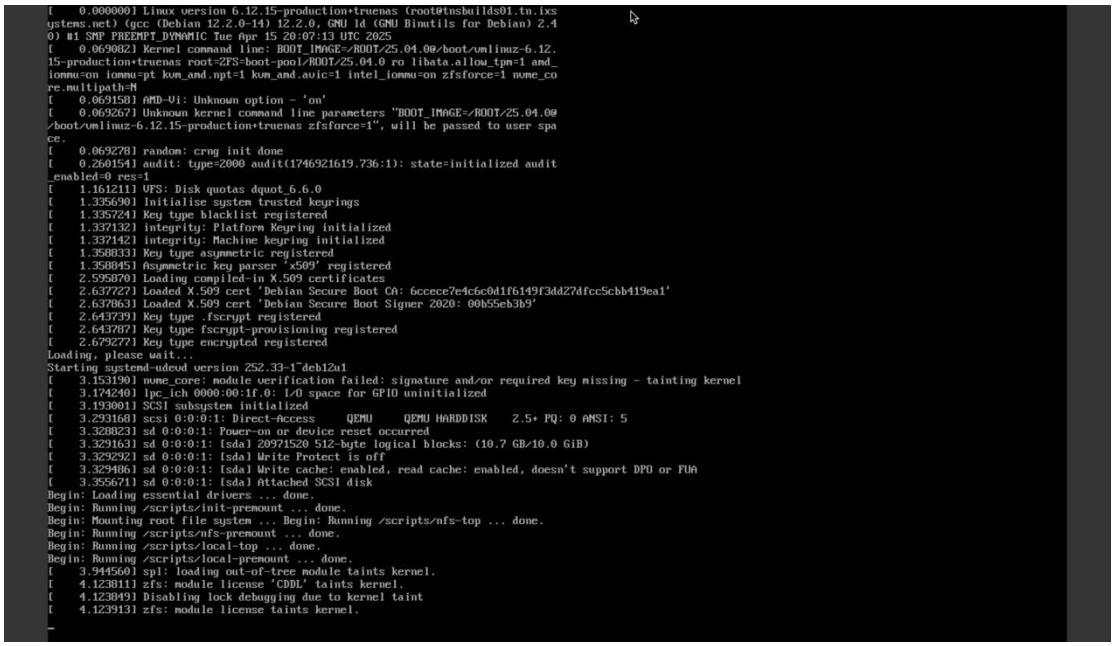


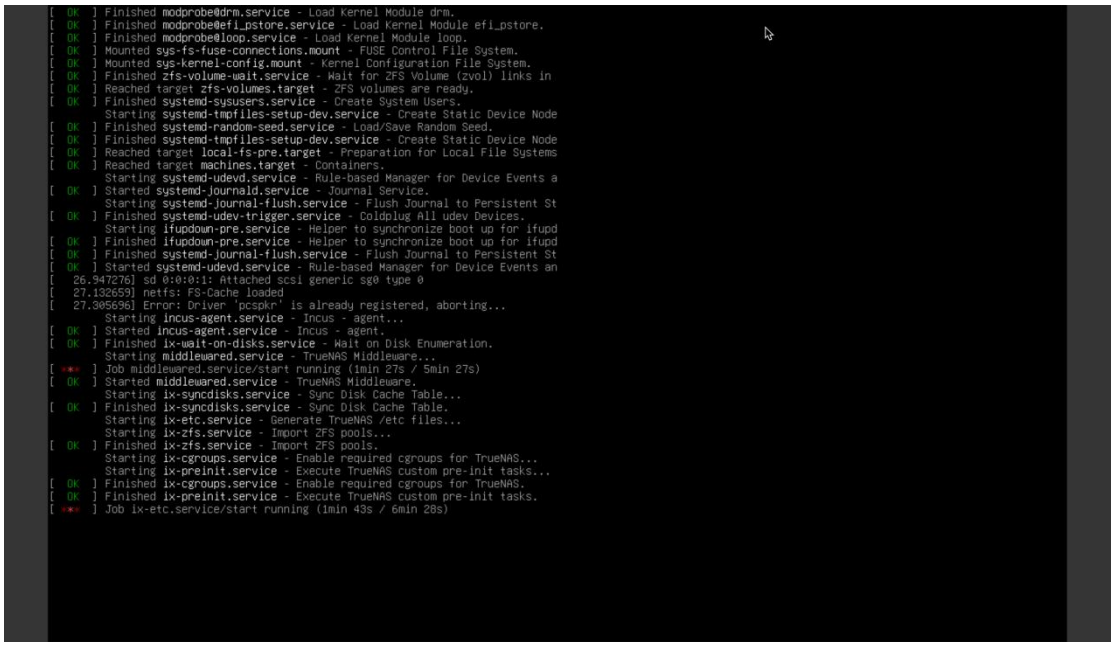
Now that the installation is completed, let us remove the attached iso file (volume) from virtual machine with the following command:

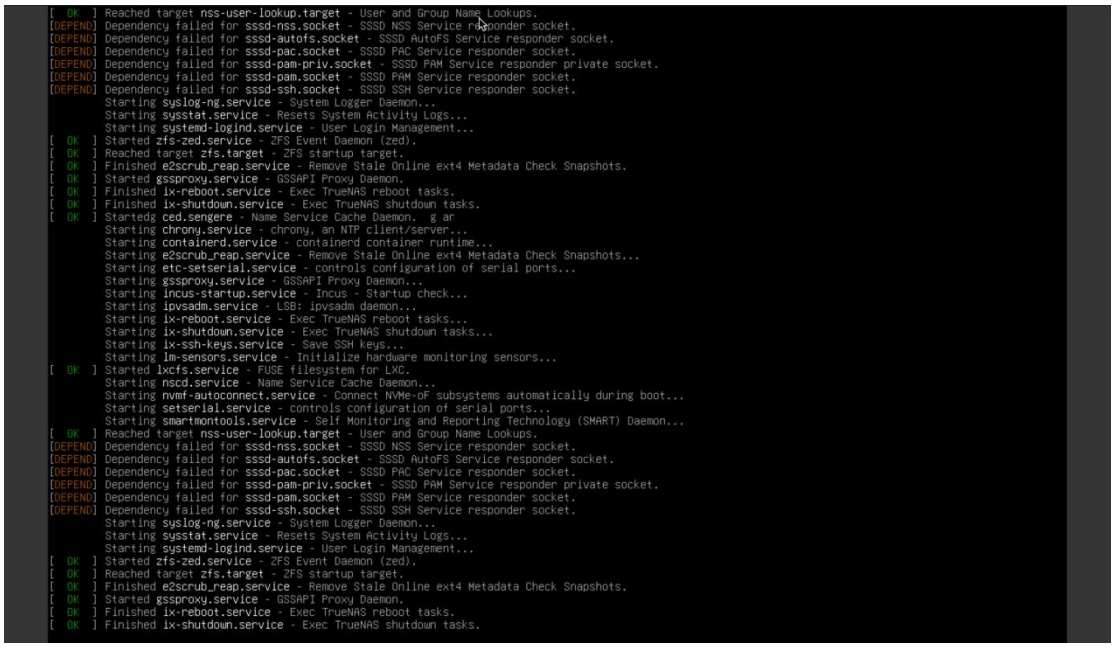
```
mac% incus config device remove vm-truenas25 install # 'install' being the install disk
```



Let us get connected back to console ones again and make sure the vitual machine boots up fine

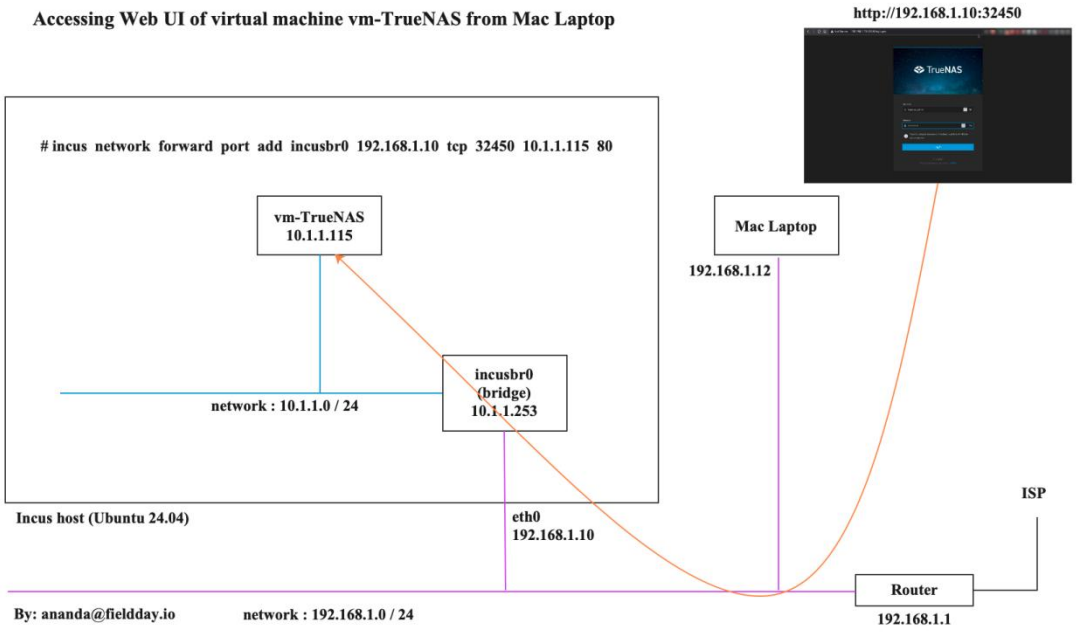

[0.000000] Linux version 6.12.15-production+truenas (root@tnsbuilds01.tn.ixsystems.net) (gcc (Debian 12.2.0-14) 12.2.0, GNU ld (GNU Binutils for Debian) 2.40) #1 SMP PREEMPT_DYNAMIC Tue Apr 15 20:07:13 UTC 2025
[0.069082] Kernel command line: BOOT_IMAGE=/ROOT/25.04.0@/boot/vmlinuz-6.12.15-production+truenas root=ZFS=boot-pool/ROOT/25.04.0 ro libata.allow_tpm=1 amd_iommu=on iommu=pt kvm_amd.npt=1 kvm_amd.avic=1 intel_iommu=on zfsforce=1 nvme_core.multipath=N
[0.069158] AMD-Vi: Unknown option - 'on'
[0.069267] Unknown kernel command line parameters "BOOT_IMAGE=/ROOT/25.04.0@/boot/vmlinuz-6.12.15-production+truenas zfsforce=1", will be passed to user space.
[0.069278] random: crng init done
[0.260154] audit: type=2000 audit(1746921619.736:1): state=initialized audit_enabled=0 res=1
[1.161211] VFS: Disk quotas dquot_6.6.0
[1.335690] Initialise system trusted keyrings
[1.335724] Key type blacklist registered
[1.337132] integrity: Platform Keyring initialized
[1.337142] integrity: Machine keyring initialized
[1.358833] Key type asymmetric registered
[1.358845] Asymmetric key parser 'x509' registered
[2.595870] Loading compiled-in X.509 certificates
[2.637727] Loaded X.509 cert 'Debian Secure Boot CA: 6ccece7e4c6c0d1f6149f3dd27dfcc5cbb419ea1'
[2.637863] Loaded X.509 cert 'Debian Secure Boot Signer 2020: 00b55eb3b9'
[2.643739] Key type .fscrypt registered
[2.643787] Key type fscrypt-provisioning registered
[2.679277] Key type encrypted registered
Loading, please wait...
Starting systemd-udevd version 252.33-1~deb12u1
[3.153190] nvme_core: module verification failed: signature and/or required key missing - tainting kernel
[3.174240] lpc_ich 0000:00:1f.0: I/O space for GPIO uninitialized
[3.193001] SCSI subsystem initialized
[3.293168] scsi 0:0:0:1: Direct-Access QEMU QEMU HARDDISK 2.5+ PQ: 0 ANSI: 5
[3.328023] sd 0:0:0:1: Power-on or device reset occurred
[3.329163] sd 0:0:0:1: [sda] 20971520 512-byte logical blocks: (10.7 GB/10.0 GiB)
[3.329292] sd 0:0:0:1: [sda] Write Protect is off
[3.329486] sd 0:0:0:1: [sda] Write cache: enabled, read cache: enabled, doesn't support DPO or FUA
[3.355671] sd 0:0:0:1: [sda] Attached SCSI disk
Begin: Loading essential drivers ... done.
Begin: Running /scripts/init-premount ... done.
Begin: Mounting root file system ... Begin: Running /scripts/nfs-top ... done.
Begin: Running /scripts/nfs-premount ... done.
Begin: Running /scripts/local-top ... done.
Begin: Running /scripts/local-premount ... done.
[3.944560] spl: loading out-of-tree module taints kernel.
[4.123811] zfs: module license 'CDDL' taints kernel.
[4.123849] Disabling lock debugging due to kernel taint
[4.123913] zfs: module license taints kernel.


[OK] Finished modprobe@drm.service - Load Kernel Module drm.
[OK] Finished modprobe@efi_pstore.service - Load Kernel Module efi_pstore.
[OK] Finished modprobe@loop.service - Load Kernel Module loop.
[OK] Mounted sys-fs-fuse-connections.mount - FUSE Control File System.
[OK] Mounted sys-kernel-config.mount - Kernel Configuration File System.
[OK] Finished zfs-volume-wait.service - Wait for ZFS Volume (zvol) links in
[OK] Reached target zfs-volumes.target - ZFS volumes are ready.
[OK] Finished systemd-sysusers.service - Create System Users.
Starting systemd-tmpfiles-setup-dev.service - Create Static Device Node
[OK] Finished systemd-random-seed.service - Load/Save Random Seed.
[OK] Finished systemd-tmpfiles-setup-dev.service - Create Static Device Node
[OK] Reached target local-fs-pre.target - Preparation for Local File Systems
[OK] Reached target machines.target - Containers.
Starting systemd-udevd.service - Rule-based Manager for Device Events a
[OK] Started systemd-journald.service - Journal Service.
Starting systemd-journal-flush.service - Flush Journal to Persistent St
[OK] Finished systemd-udev-trigger.service - Coldplug All udev Devices.
Starting ifupdown-pre.service - Helper to synchronize boot up for ifupd
[OK] Finished ifupdown-pre.service - Helper to synchronize boot up for ifupd
[OK] Finished systemd-journal-flush.service - Flush Journal to Persistent St
[OK] Started systemd-udevd.service - Rule-based Manager for Device Events an
[26.947276] sd 0:0:0:1: Attached scsi generic sg0 type 0
[27.132659] netfs: FS-Cache loaded
[27.305696] Error: Driver 'pcspkr' is already registered, aborting...
Starting incus-agent.service - Incus - agent...
[OK] Started incus-agent.service - Incus - agent.
[OK] Finished ix-wait-on-disks.service - Wait on Disk Enumeration.
Starting middlewared.service - TrueNAS Middleware...
[***] Job middlewared.service/start running (1min 27s / 5min 27s)
[OK] Started middlewared.service - TrueNAS Middleware.
Starting ix-syncdisks.service - Sync Disk Cache Table...
[OK] Finished ix-syncdisks.service - Sync Disk Cache Table.
Starting ix-etc.service - Generate TrueNAS /etc files...
Starting ix-zfs.service - Import ZFS pools...
[OK] Finished ix-zfs.service - Import ZFS pools.
Starting ix-cgroups.service - Enable required cgroups for TrueNAS...
Starting ix-preinit.service - Execute TrueNAS custom pre-init tasks...
[OK] Finished ix-cgroups.service - Enable required cgroups for TrueNAS.
[OK] Finished ix-preinit.service - Execute TrueNAS custom pre-init tasks.
[***] Job ix-etc.service/start running (1min 43s / 6min 28s)



We have reached a milestore where the installation is completed. Let us now configure TrueNAS from our Mac Laptop using the browser. But to accomplish that, we need to understand what it takes to reach out the administrative web UI running on vm-truenas25, which is on a different network segment (10.1.1.115), from the host os (192.1168.1.10). which is the same network as our mac laptop.

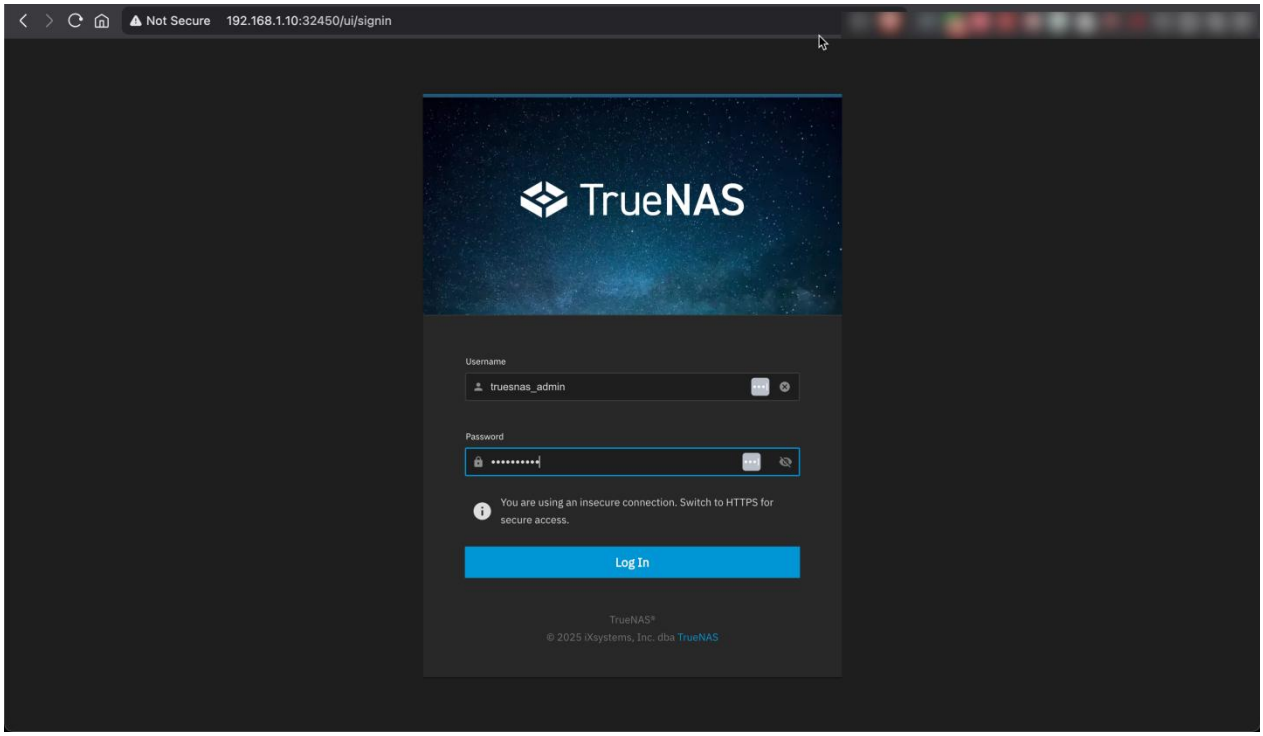
Let us understand our topology clearly so that we know what is takes to reach the incus admin web UI from mac laptop.



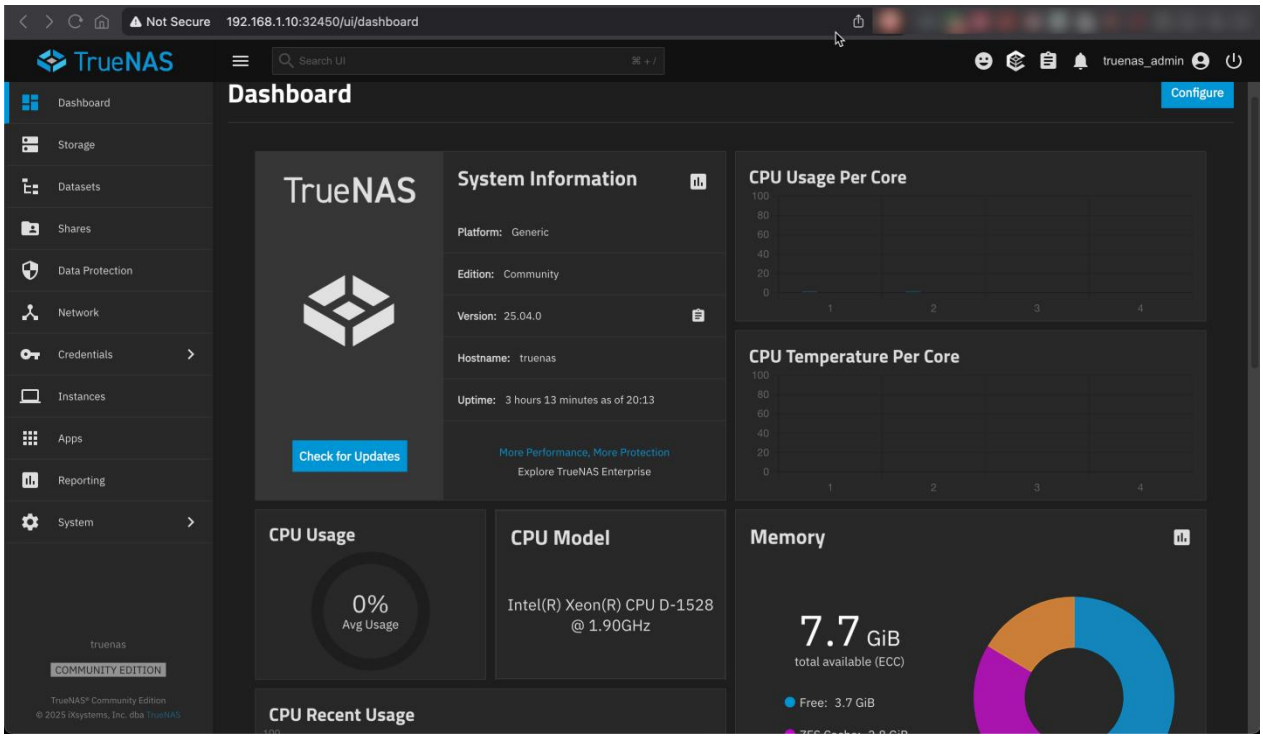
Issuing the network forward rule show below on the incus server at ip address 192.168.1.10, port 32450 will be forwarded to ip address 10.1.1.115, port number 80, via incusbr0

```
# incus network forward port add incusbr0 192.168.1.10 tcp 32450 10.1.1.115 80
```

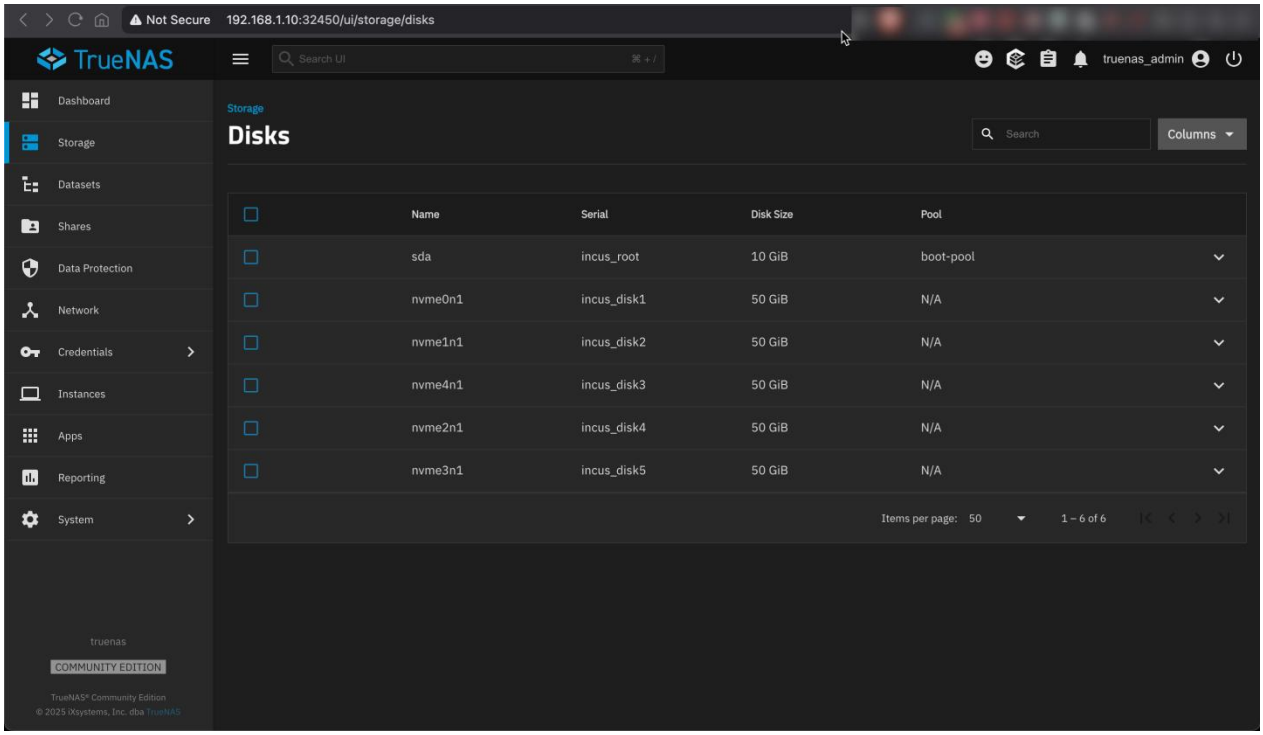
Now pointing the browser on mac laptop to http://192.168.1.10:32450 will get us the admin portal of TrueNAS running at 10.1.1.115



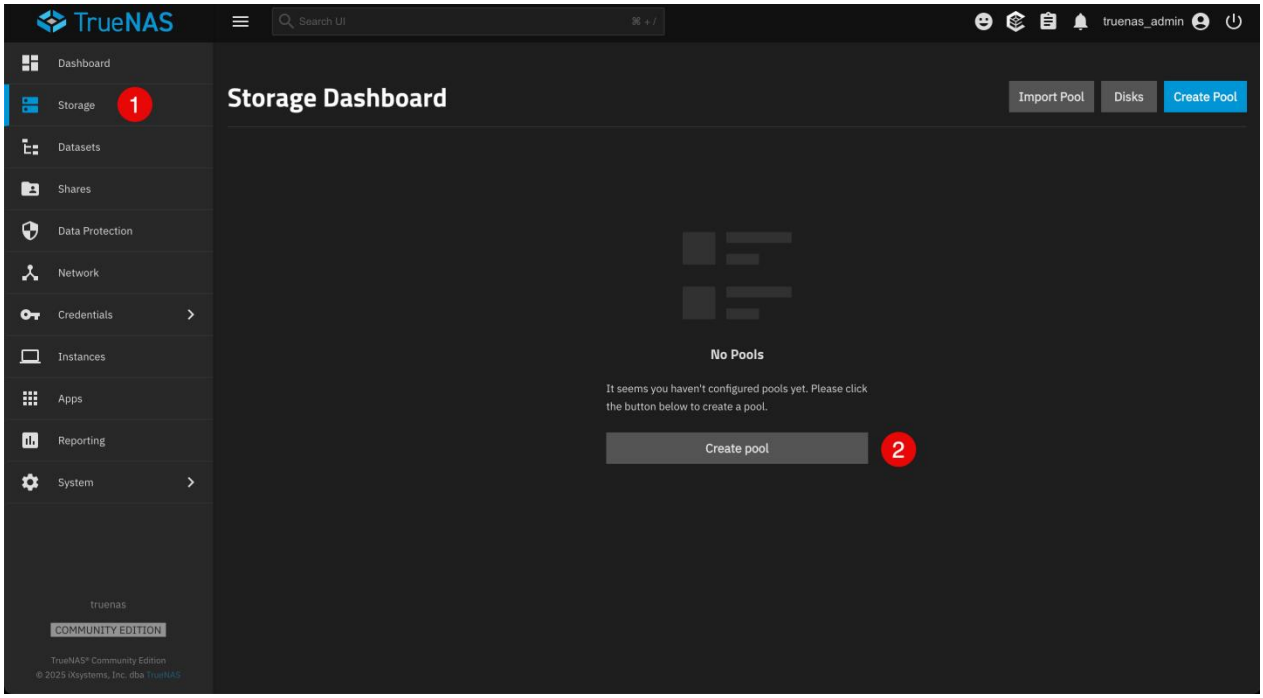
The Main Dashboard

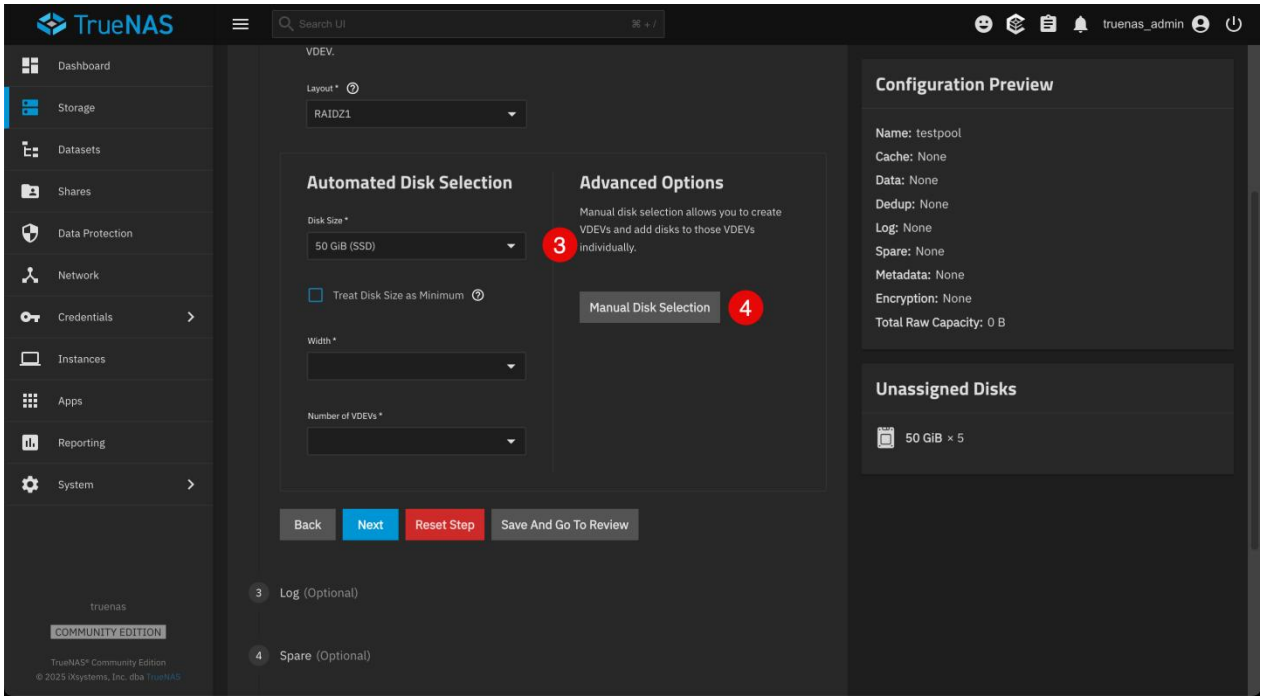


All 5 disks will be listed

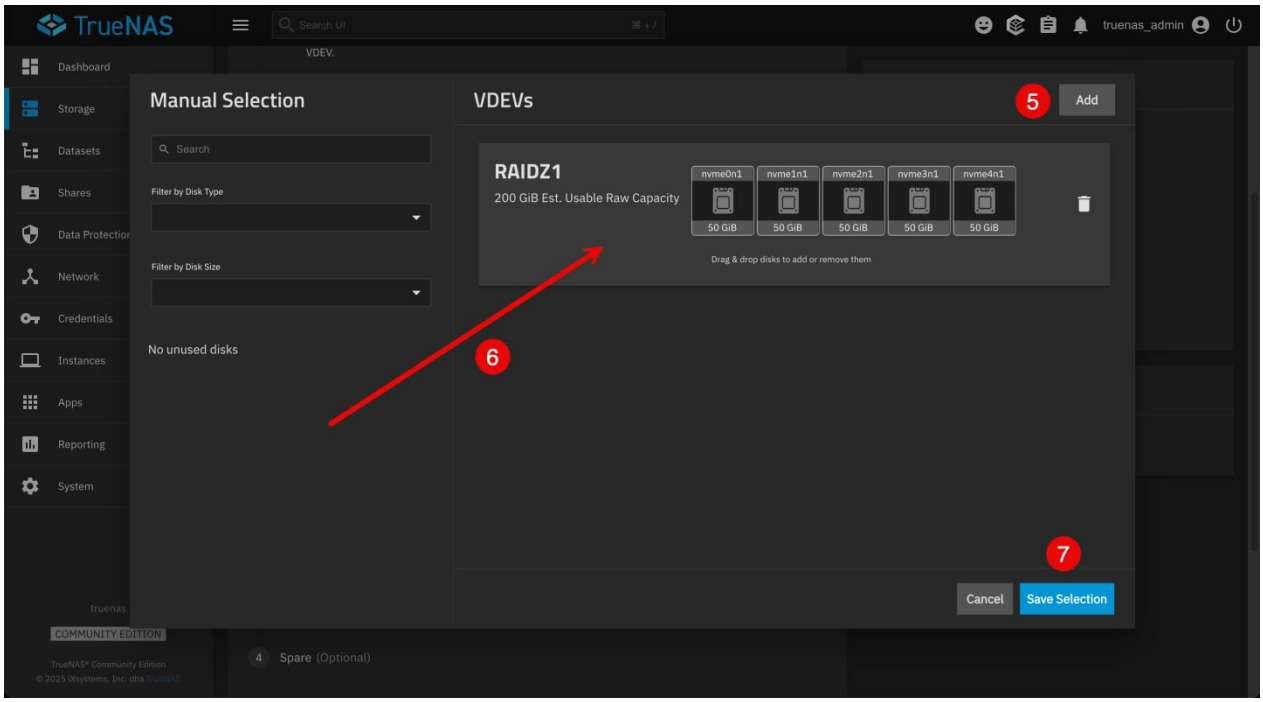


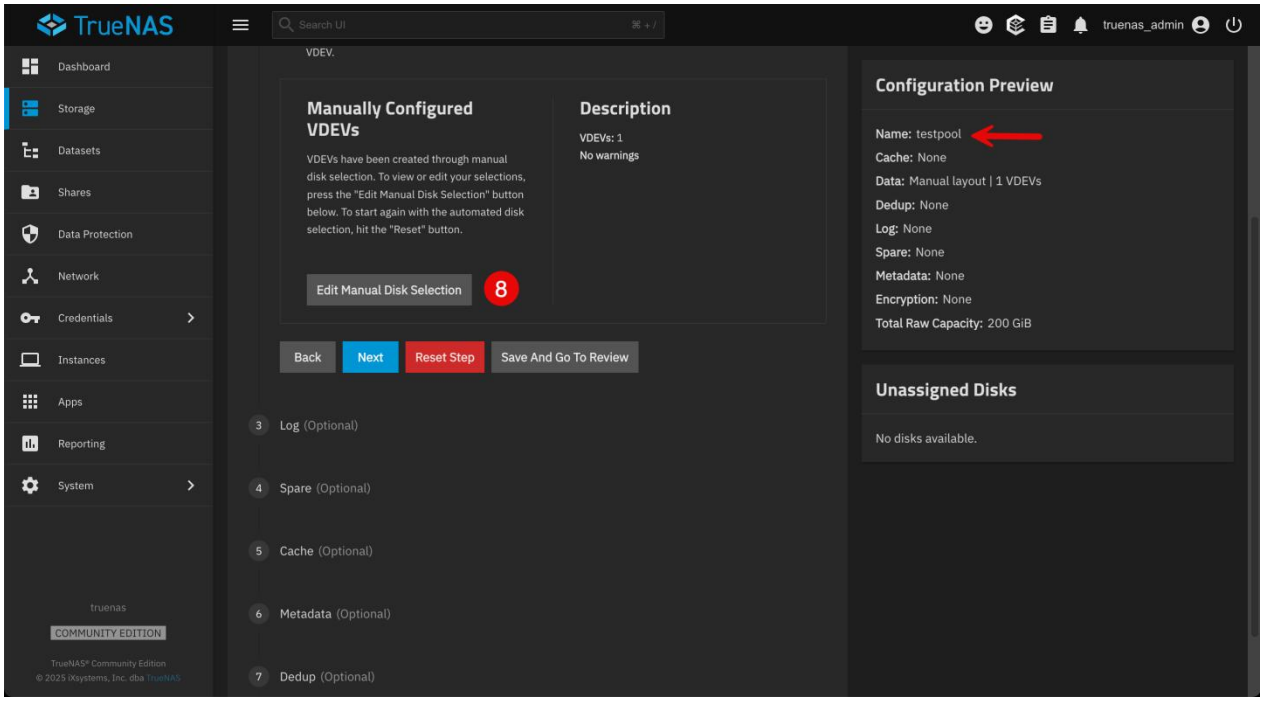
Create a new storage pool

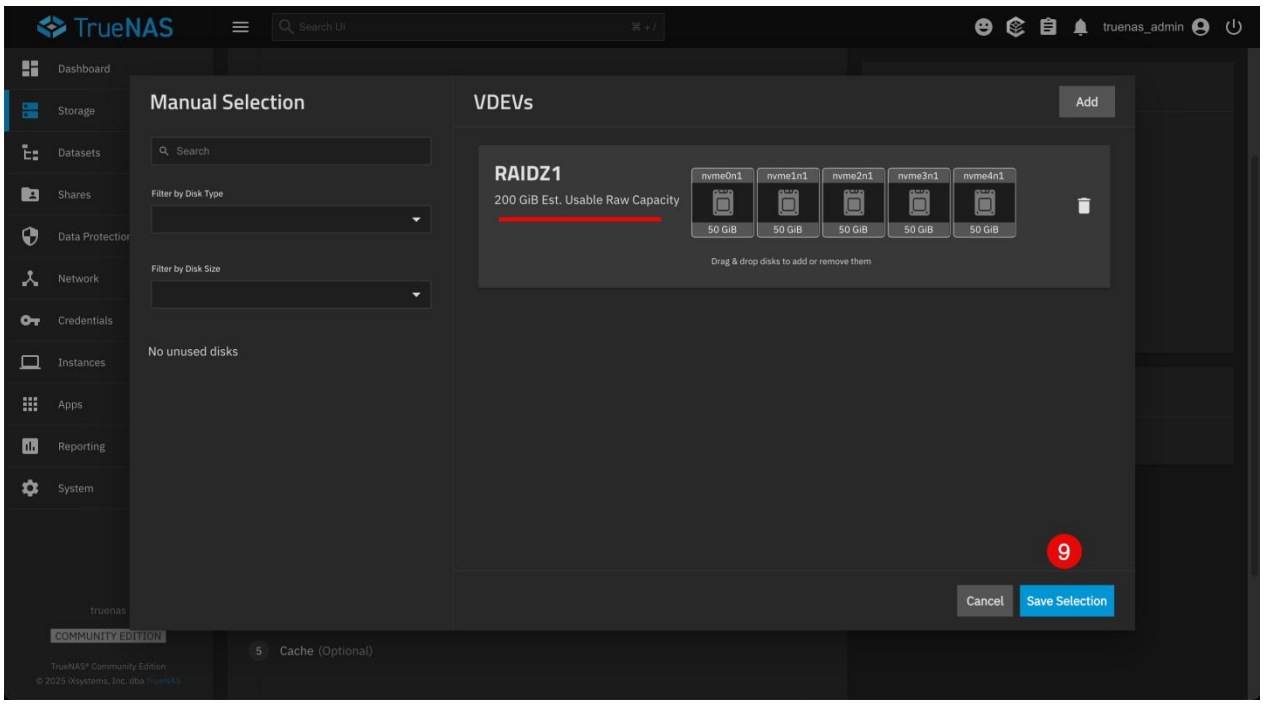


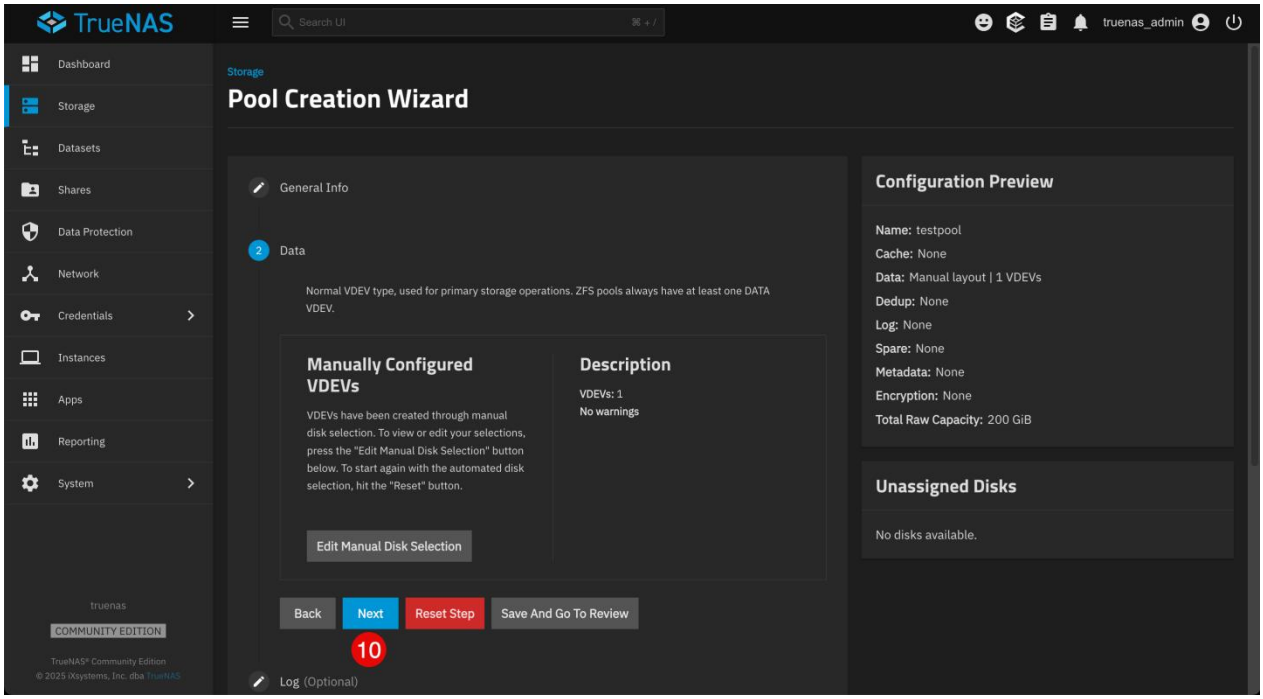


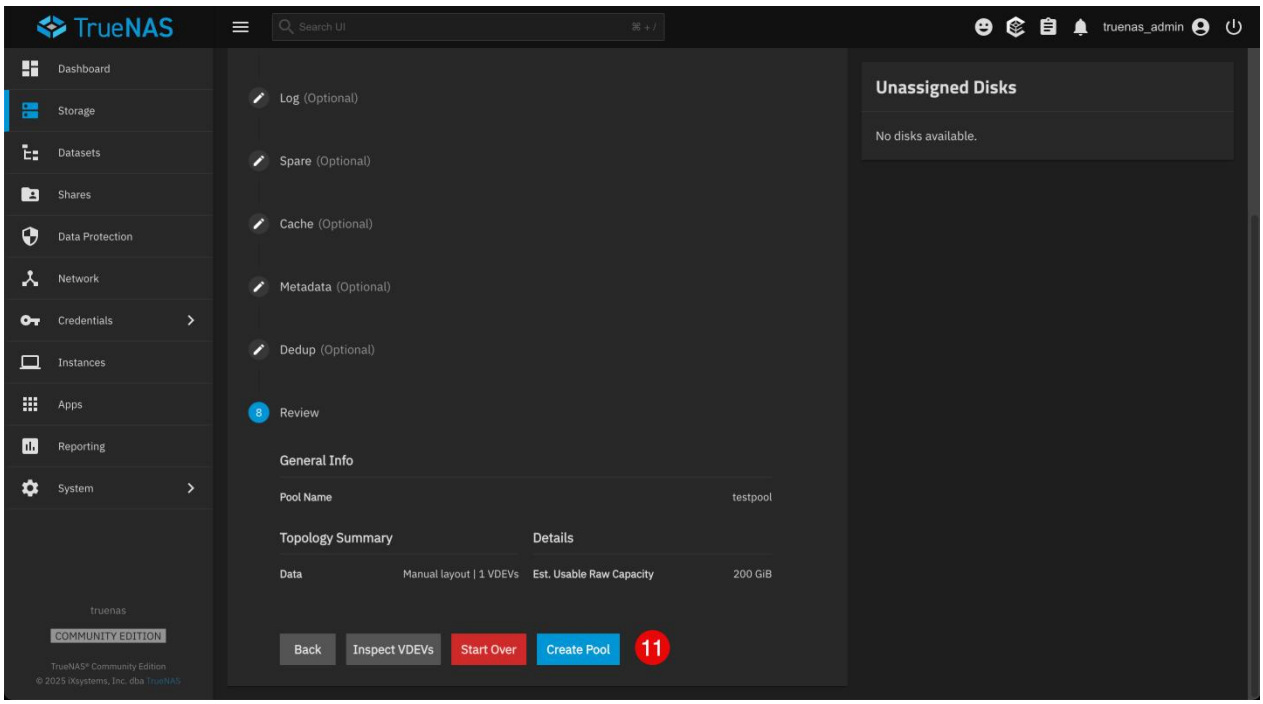
Select all 5 disks to create a RAIDZ1

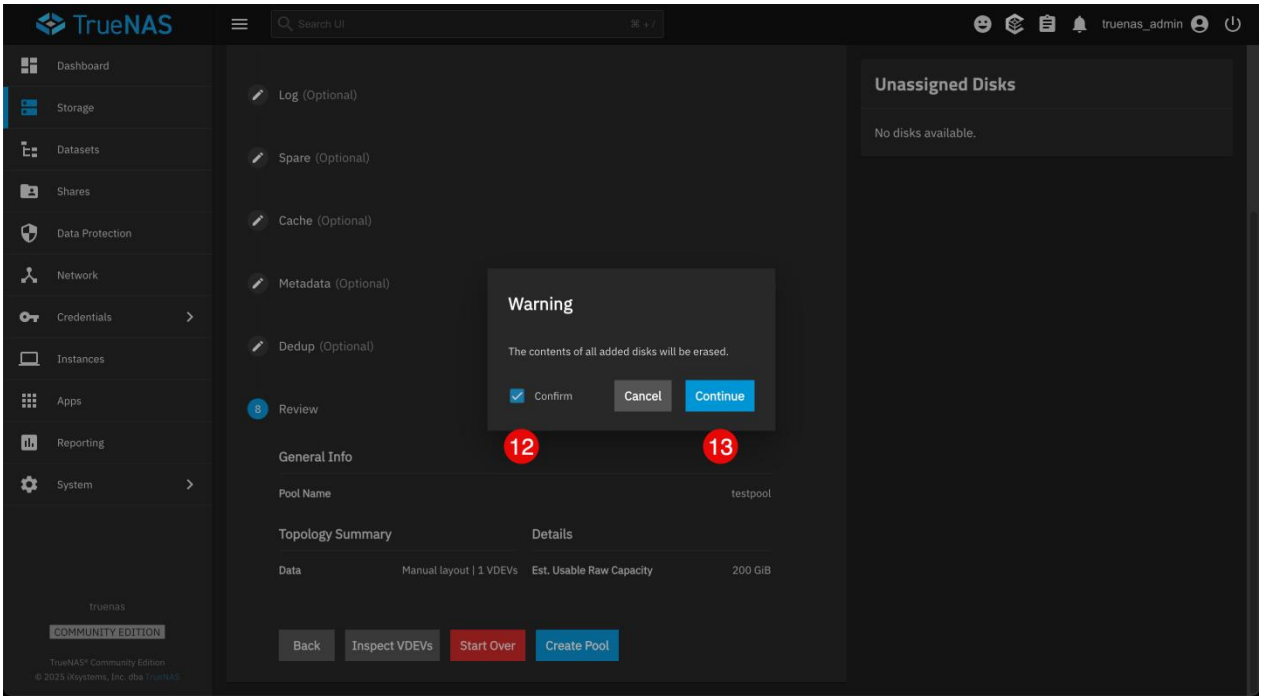


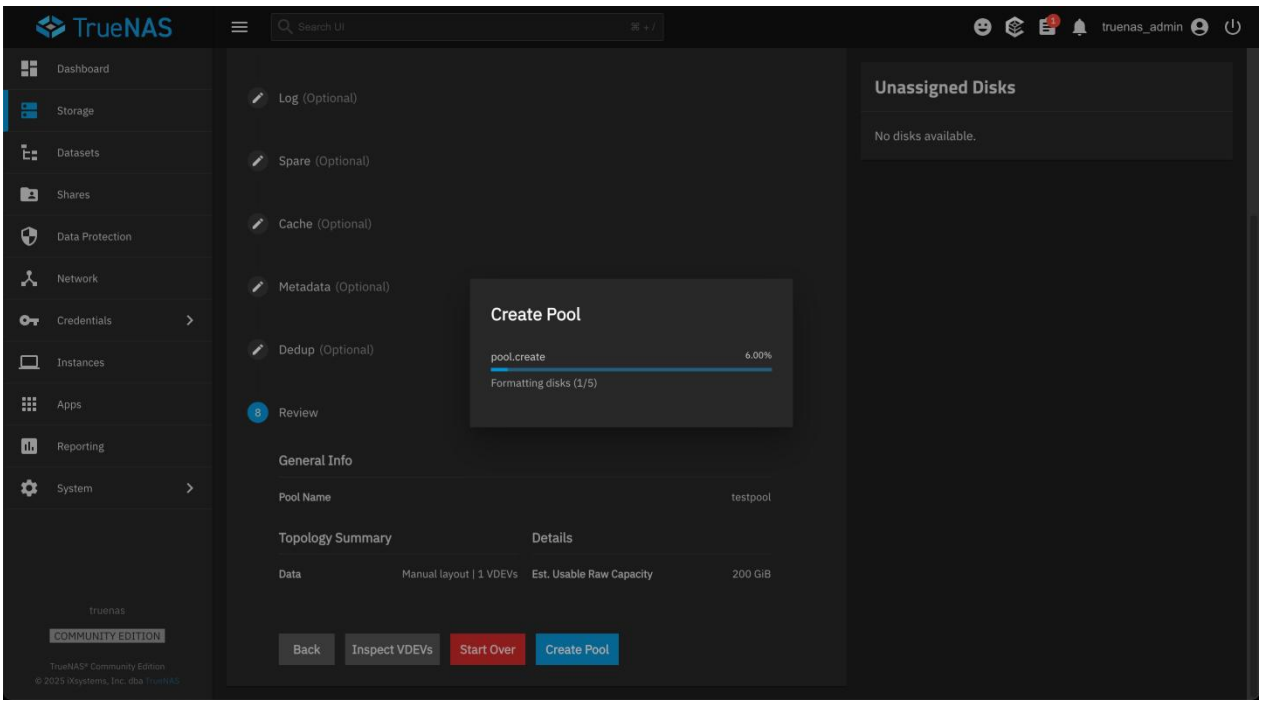

TrueNAS
Search UI
truenas_admin
Dashboard
Storage
Datasets
Shares
Data Protection
Network
Credentials
Instances
Apps
Reporting
System
VDEV.
Manually Configured VDEVs
VDEVs have been created through manual disk selection. To view or edit your selections, press the "Edit Manual Disk Selection" button below. To start again with the automated disk selection, hit the "Reset" button.
Edit Manual Disk Selection
8
Description
VDEVs: 1
No warnings
Back
Next
Reset Step
Save And Go To Review
3 Log (Optional)
4 Spare (Optional)
5 Cache (Optional)
6 Metadata (Optional)
7 Dedup (Optional)
Configuration Preview
Name: testpool
Cache: None
Data: Manual layout | 1 VDEVs
Dedup: None
Log: None
Spare: None
Metadata: None
Encryption: None
Total Raw Capacity: 200 GiB
Unassigned Disks
No disks available.
truenas
COMMUNITY EDITION
TrueNAS® Community Edition
© 2025 iXsystems, Inc. dba TrueNAS

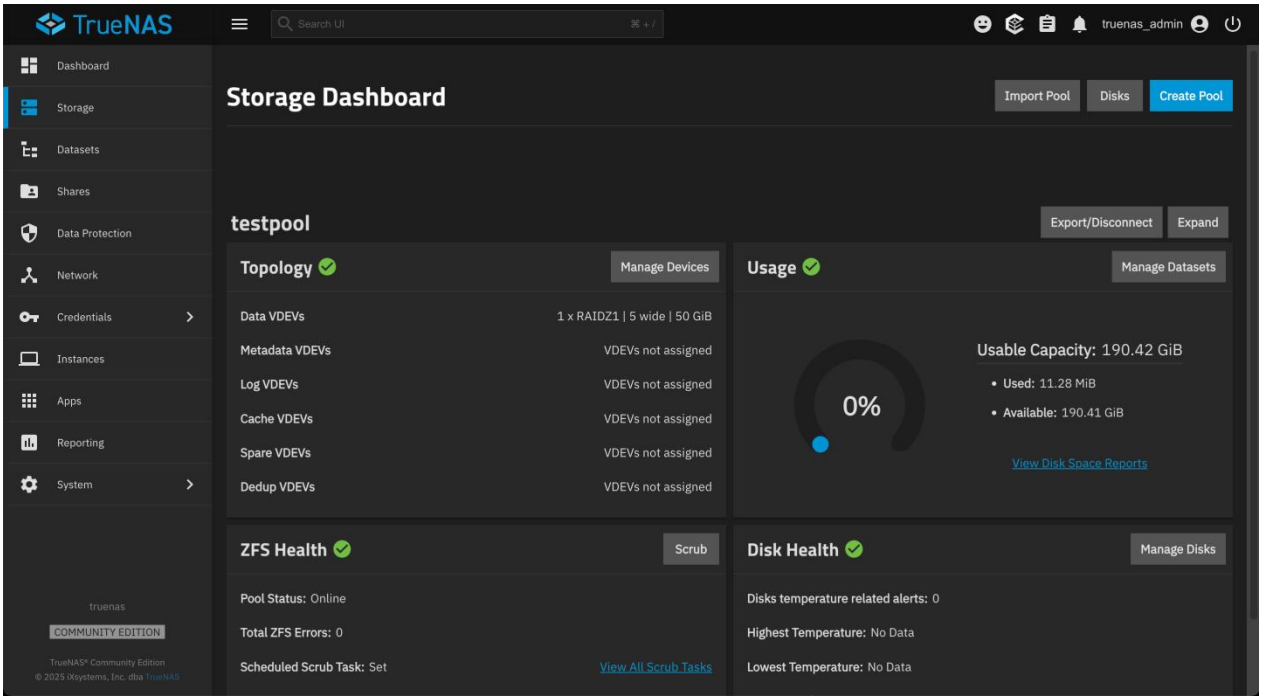

TrueNAS
Search UI
truenas_admin
Dashboard
Storage
Datasets
Shares
Data Protection
Network
Credentials
Instances
Apps
Reporting
System
Manual Selection
Search
Filter by Disk Type
Filter by Disk Size
No unused disks
VDEVs
Add
RAIDZ1
200 GiB Est. Usable Raw Capacity
nvme0n1
nvme1n1
nvme2n1
nvme3n1
nvme4n1
50 GiB
Drag & drop disks to add or remove them
9
Cancel
Save Selection
5 Cache (Optional)
truenas
COMMUNITY EDITION
TrueNAS® Community Edition
© 2025 iXsystems, Inc. dba TrueNAS


TrueNAS
Search UI
truenas_admin
Dashboard
Storage
Datasets
Shares
Data Protection
Network
Credentials
Instances
Apps
Reporting
System
truenas
COMMUNITY EDITION
TrueNAS® Community Edition
© 2025 iXsystems, Inc. dba TrueNAS
Storage
Pool Creation Wizard
General Info
2 Data
Normal VDEV type, used for primary storage operations. ZFS pools always have at least one DATA VDEV.
Manually Configured VDEVs
VDEVs have been created through manual disk selection. To view or edit your selections, press the "Edit Manual Disk Selection" button below. To start again with the automated disk selection, hit the "Reset" button.
Edit Manual Disk Selection
Description
VDEVs: 1
No warnings
Back
Next
Reset Step
Save And Go To Review
10
Log (Optional)
Configuration Preview
Name: testpool
Cache: None
Data: Manual layout | 1 VDEVs
Dedup: None
Log: None
Spare: None
Metadata: None
Encryption: None
Total Raw Capacity: 200 GiB
Unassigned Disks
No disks available.


TrueNAS
Search UI
truenas_admin
Dashboard
Storage
Datasets
Shares
Data Protection
Network
Credentials
Instances
Apps
Reporting
System
truenas
COMMUNITY EDITION
TrueNAS® Community Edition
© 2025 iXsystems, Inc. dba TrueNAS
Log (Optional)
Spare (Optional)
Cache (Optional)
Metadata (Optional)
Dedup (Optional)
8 Review
General Info
Pool Name
testpool
Topology Summary
Data
Manual layout | 1 VDEVs
Details
Est. Usable Raw Capacity
200 GiB
Back
Inspect VDEVs
Start Over
Create Pool
11
Unassigned Disks
No disks available.


TrueNAS
Search UI
truenas_admin
Dashboard
Storage
Datasets
Shares
Data Protection
Network
Credentials
Instances
Apps
Reporting
System
Log (Optional)
Spare (Optional)
Cache (Optional)
Metadata (Optional)
Dedup (Optional)
8 Review
Warning
The contents of all added disks will be erased.
Confirm
Cancel
Continue
12
13
General Info
Pool Name
testpool
Topology Summary
Details
Data
Manual layout | 1 VDEVs
Est. Usable Raw Capacity
200 GiB
Back
Inspect VDEVs
Start Over
Create Pool
Unassigned Disks
No disks available.
truenas
COMMUNITY EDITION
TrueNAS® Community Edition
© 2025 iXsystems, Inc. dba TrueNAS


TrueNAS
Search UI
truenas_admin
Dashboard
Storage
Datasets
Shares
Data Protection
Network
Credentials
Instances
Apps
Reporting
System
Log (Optional)
Spare (Optional)
Cache (Optional)
Metadata (Optional)
Dedup (Optional)
8 Review
Create Pool
pool.create
6.00%
Formatting disks (1/5)
General Info
Pool Name
testpool
Topology Summary
Details
Data
Manual layout | 1 VDEVs
Est. Usable Raw Capacity
200 GiB
Back
Inspect VDEVs
Start Over
Create Pool
Unassigned Disks
No disks available.
truenas
COMMUNITY EDITION
TrueNAS® Community Edition
© 2025 iXsystems, Inc. dba TrueNAS

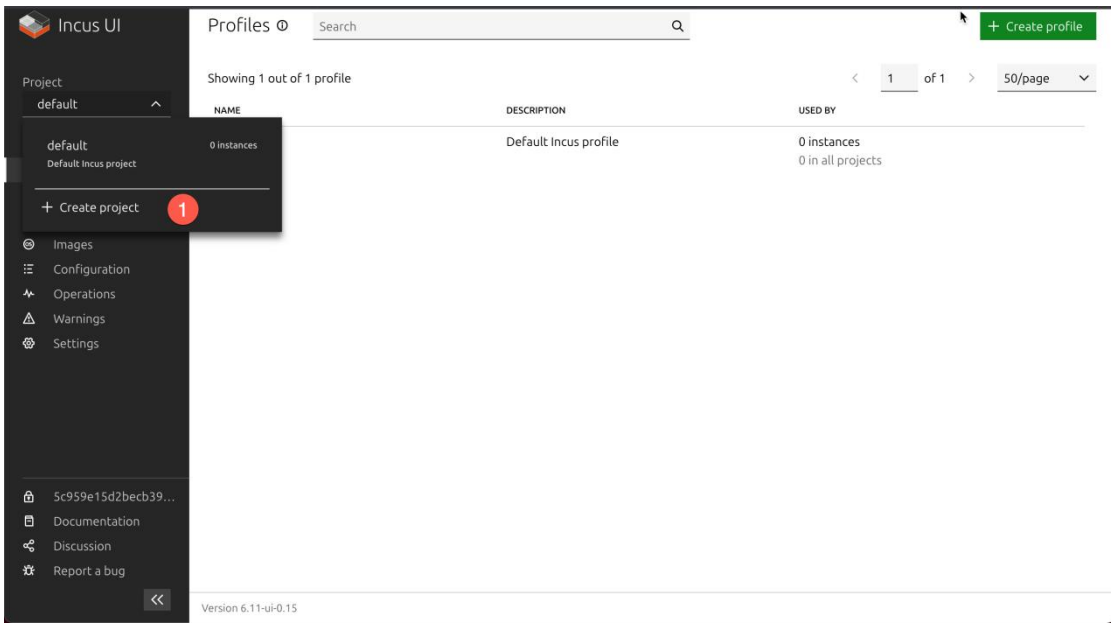

TrueNAS
Search UI
⌘ + /
truenas_admin
Dashboard
Storage
Datasets
Shares
Data Protection
Network
Credentials
Instances
Apps
Reporting
System
truenas
COMMUNITY EDITION
TrueNAS® Community Edition
© 2025 iXsystems, Inc. dba TrueNAS
Storage Dashboard
Import Pool
Disks
Create Pool
testpool
Export/Disconnect
Expand
Topology
Manage Devices
Data VDEVs
1 x RAIDZ1 | 5 wide | 50 GiB
Metadata VDEVs
VDEVs not assigned
Log VDEVs
VDEVs not assigned
Cache VDEVs
VDEVs not assigned
Spare VDEVs
VDEVs not assigned
Dedup VDEVs
VDEVs not assigned
Usage
Manage Datasets
0%
Usable Capacity: 190.42 GiB
Used: 11.28 MiB
Available: 190.41 GiB
View Disk Space Reports
ZFS Health
Scrub
Pool Status: Online
Total ZFS Errors: 0
Scheduled Scrub Task: Set
View All Scrub Tasks
Disk Health
Manage Disks
Disks temperature related alerts: 0
Highest Temperature: No Data
Lowest Temperature: No Data

Task #17: Creating Projects using web UI

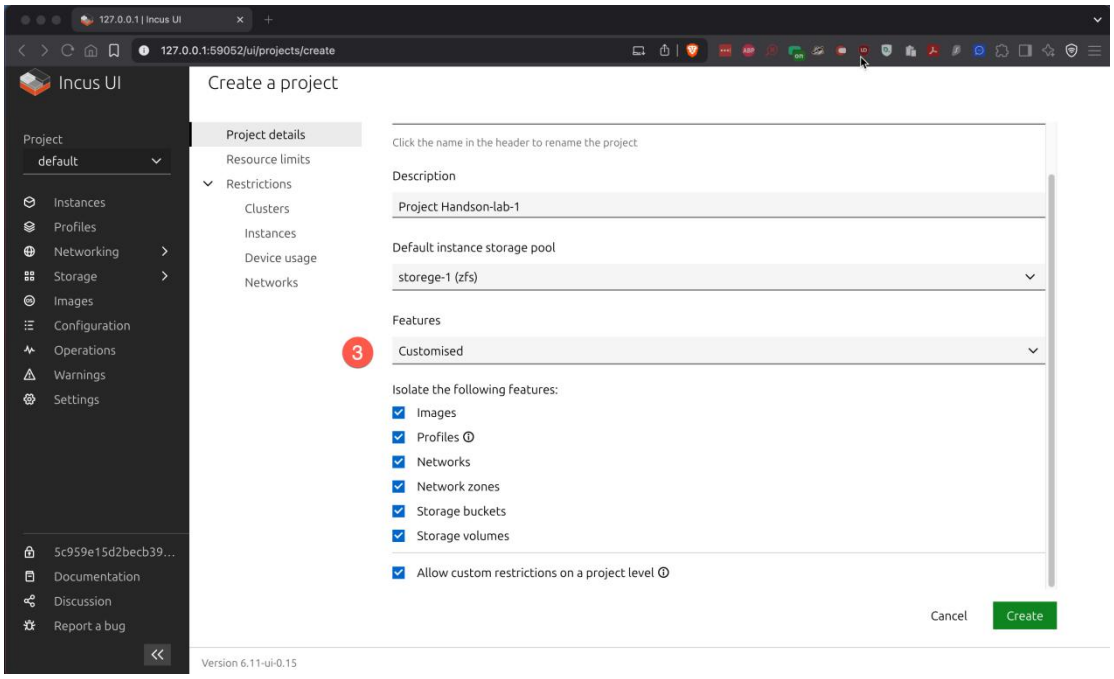
In this task we will create projects using web UI. We will create 6 Projects in total. We well then assign each Project with it's own storage, the ones we created in previous tasks.

Running the command 'incus webui' on our Mac laptop will open up a browser session to our incus server (recollect that we have already establised trust with the incus server).

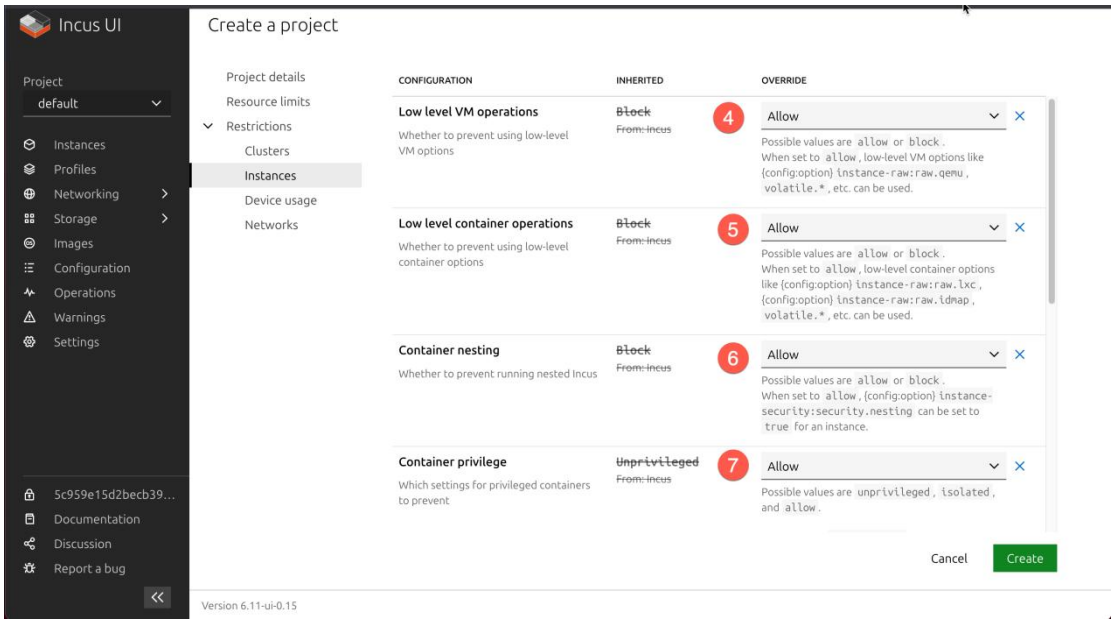
Let us navigate the left panel and create a new project as shown above.



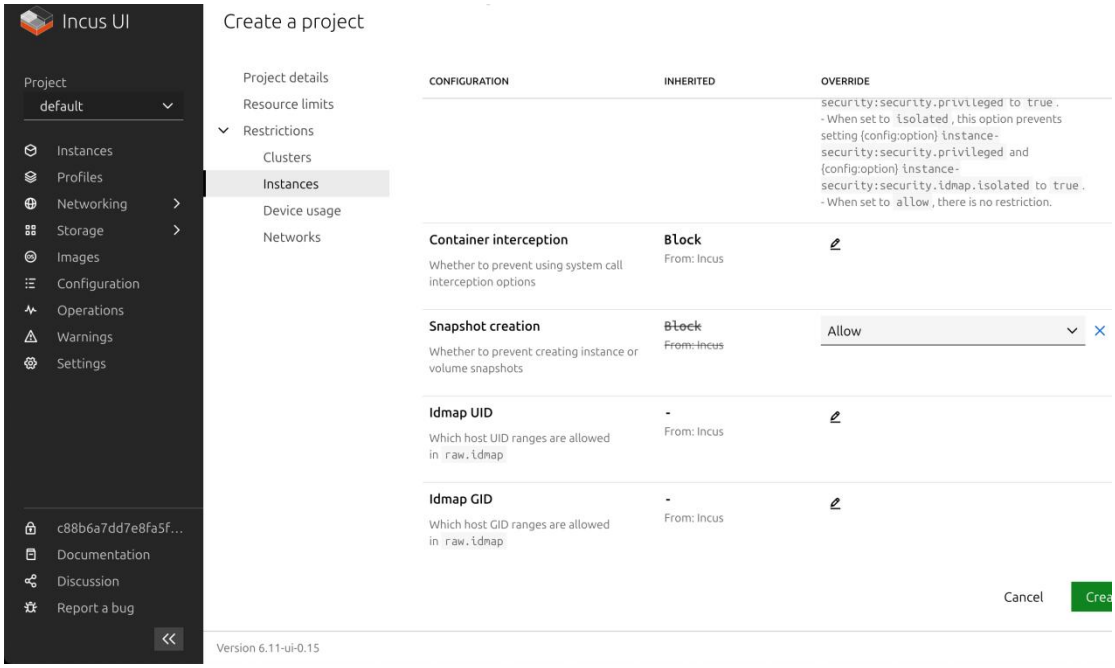
1. Let us give our first project a name 'Handson-lab-1'.
2. Enter a relavant text under Description
3. From the dropdown menu we will select and assign 'storage-1' to this Project
4. Select 'Customized' under Features dropdown menu

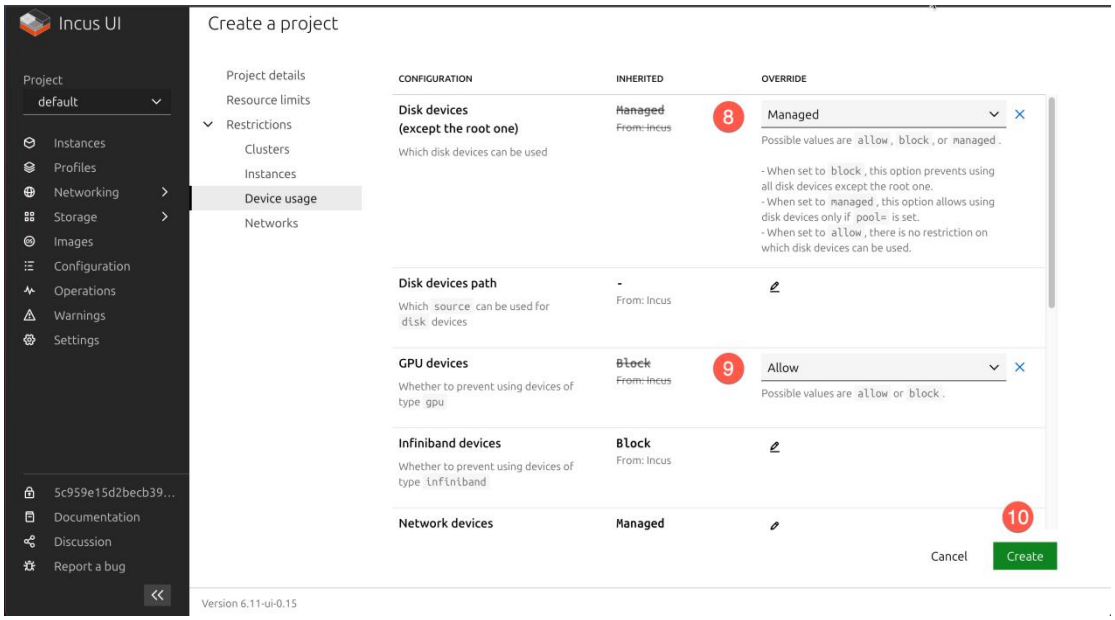


5. Under Restrictions, let us select ‘Instances’ and ‘Allow’ previleges to all instances

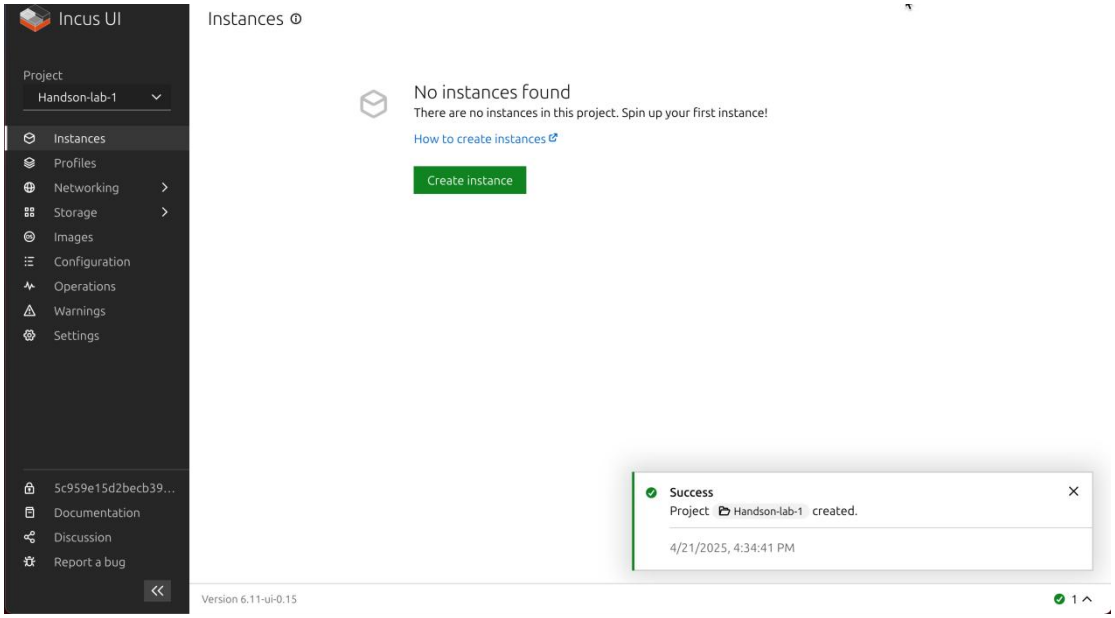


6. Under Restrictions, let us give privileges to disk devices and GPU devices
7. Allow Snapshot creation features as well





8. We will get a confirmation on sucessful creation of the Project - Handson-lab-1
9. Note that on confirmation, the UI lists Handson-lab-1 under Project in the left panel
10. We need to keep in mind that any changes made in the UI will reflect on that particular project alone, until the user selects a different from project from the dropdown menu



Let us run the following command to see the current configuration of the project - handson-lab-1

```
mac$ incus project list -f compact
       NAME        IMAGES  PROFILES  STORAGE VOLUMES  STORAGE BUCKETS  NETWORKS  NETWORK ZONES      DESCRIPTION      USED BY
  Handson-lab-1     YES     YES       YES              YES              YES       YES              Project Handson-lab-1  1
  default (current) YES     YES       YES              YES              YES       YES              Default Incus project  2
```

```
mac$ incus project show Handson-lab-1
config:
  features.images: "true"
  features.networks: "false"
  features.networks.zones: "false"
  features.profiles: "true"
  features.storage.buckets: "true"
  features.storage.volumes: "true"
  restricted: "true"
  restricted.containers.lowlevel: allow
  restricted.containers.nesting: allow
  restricted.containers.privilege: allow
  restricted.devices.disk: managed
  restricted.devices.gpu: allow
  restricted.snapshots: allow
  restricted.virtual-machines.lowlevel: allow
description: Project Handson-lab-1
name: Handson-lab-1
used_by:
- /1.0/profiles/default?project=Handson-lab-1
```

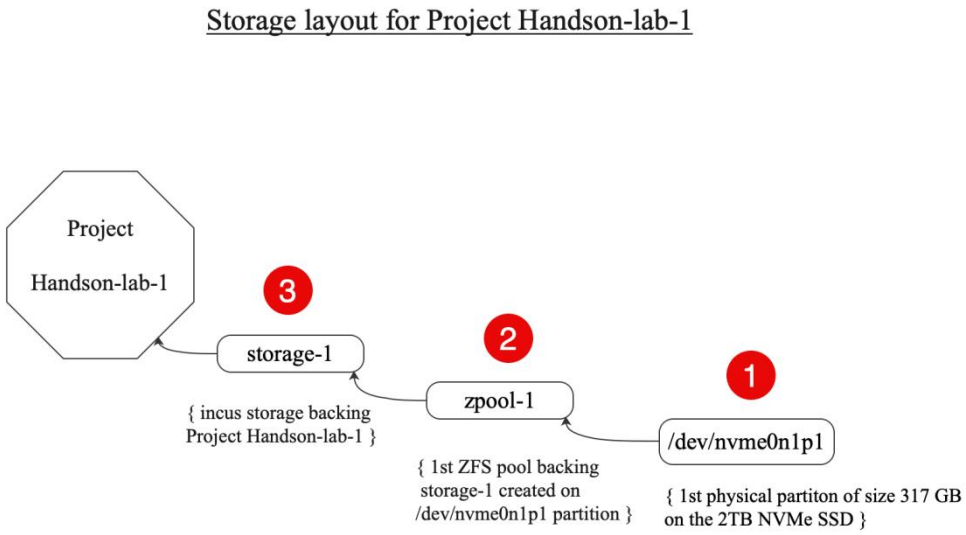
From the storage side as well we can confirm that project Handson-lab-1 is indeed backed by storage-1

```
mac% incus storage show storage-1
config:
  source: zpool-1
  volatile.initial_source: zpool-1
  zfs.pool_name: zpool-1
description: Storage for Containers, VMs of Handson-lab-1
name: storage-1
driver: zfs
used_by:
- /1.0/profiles/default?project=Handson-lab-1
status: Created
locations:
- none
```

And we know storage-1 is indeed backed by the zfs pool zpool-1 , which is shown above defined in the key zfs.pool_name: zpool-1

Task #18: Verifying storage for project: Handson-lab-1

In this task we will take a quick look at how the storage layout is being assigned to the project 'Handson-lab-1' (We are not planning on configuring anything in this task).



A zfs pool named 'zool-1' was created on the first physical partition on the 2TB NVMe SSD /dev/nvme0p1. incus storage pool 'storage-1' was created on zpool-1, which is now managed by incus.

```
# lsblk
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda           8:0    1 460.3G  0 disk
├─sda1        8:1    1     1M  0 part
└─sda2        8:2    1 460.3G  0 part /
zd0         230:0    0   5.8G  0 disk
nvme0n1     259:0    0   1.9T  0 disk
├─nvme0n1p1 259:1    0   317G  0 part ---> zpool-1
├─nvme0n1p2 259:2    0   317G  0 part
├─nvme0n1p3 259:3    0   317G  0 part
├─nvme0n1p4 259:4    0   317G  0 part
├─nvme0n1p5 259:5    0   317G  0 part
├─nvme0n1p6 259:6    0   317G  0 part
└─nvme0n1p7 259:7    0   5.7G  0 part
nvme1n1     259:8    0 931.5G  0 disk
├─nvme1n1p1 259:9    0   155G  0 part
├─nvme1n1p2 259:10   0   155G  0 part
├─nvme1n1p3 259:11   0   155G  0 part
├─nvme1n1p4 259:12   0   155G  0 part
```

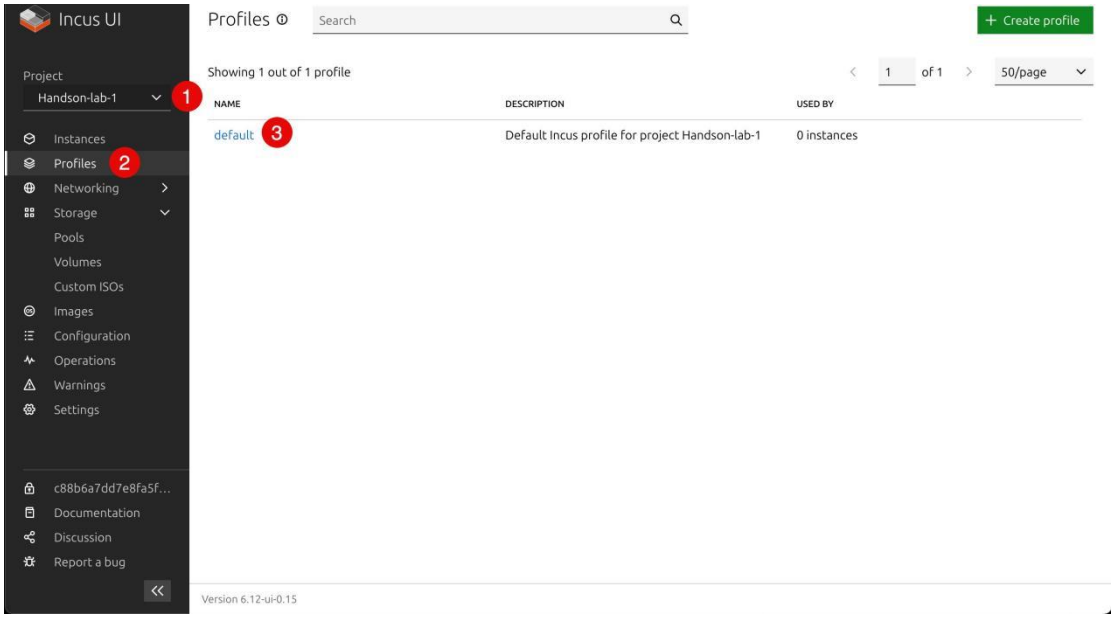
```
├─nvme1n1p5 259:13   0   155G  0 part
├─nvme1n1p6 259:14   0   155G  0 part
└─nvme1n1p7 259:15   0   1.5G  0 part

# zpool list
NAME         SIZE  ALLOC   FREE  CKPOINT  EXPANDSZ   FRAG    CAP  DEDUP    HEALTH  ALTROOT
basepool     5.50G  7.13M  5.49G        -         -    22%     0%  1.00x    ONLINE  -
disks-pool    308G  2.92M   308G        -         -     0%     0%  1.00x    ONLINE  -
images-pool   462G  1.15M   462G        -         -     0%     0%  1.00x    ONLINE  -
iso-pool      154G  5.86G   148G        -         -     0%     3%  1.00x    ONLINE  -
zpool-1       316G  1002K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-2       316G   804K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-3       316G   741K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-4       316G   724K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-5       316G   747K   316G        -         -     0%     0%  1.00x    ONLINE  -
zpool-6       316G   747K   316G        -         -     0%     0%  1.00x    ONLINE  -

# incus storage list -f compact
     NAME       DRIVER                DESCRIPTION                    USED BY   STATE
  basepool       zfs     passed to incus admin init                     1       CREATED
  shared-disks    zfs     shared disks accross Projects                 0       CREATED
  shared-images   zfs     shared images accross Projects                1       CREATED
  shared-iso      zfs     shared ISOs accross Projects                  1       CREATED
  storage-1       zfs     Storage for Containers, VMs of Handson-lab-1  1       CREATED
  storage-2       zfs     Storage for Containers, VMs of Handson-lab-2  0       CREATED
  storage-3       zfs     Storage for Containers, VMs of Handson-lab-3  0       CREATED
  storage-4       zfs     Storage for Containers, VMs of Handson-lab-4  0       CREATED
  storage-5       zfs     Storage for Containers, VMs of Handson-lab-5  0       CREATED
  storage-6       zfs     Storage for Containers, VMs of Handson-lab-6  0       CREATED
```

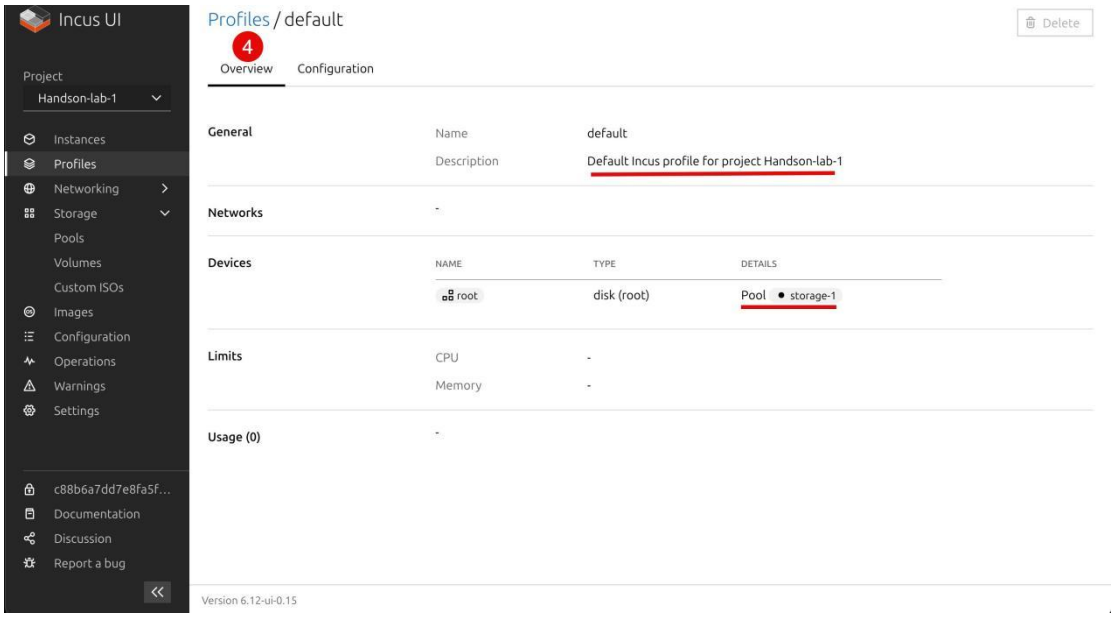
Task #19: default profile for project: Handon-lab-1

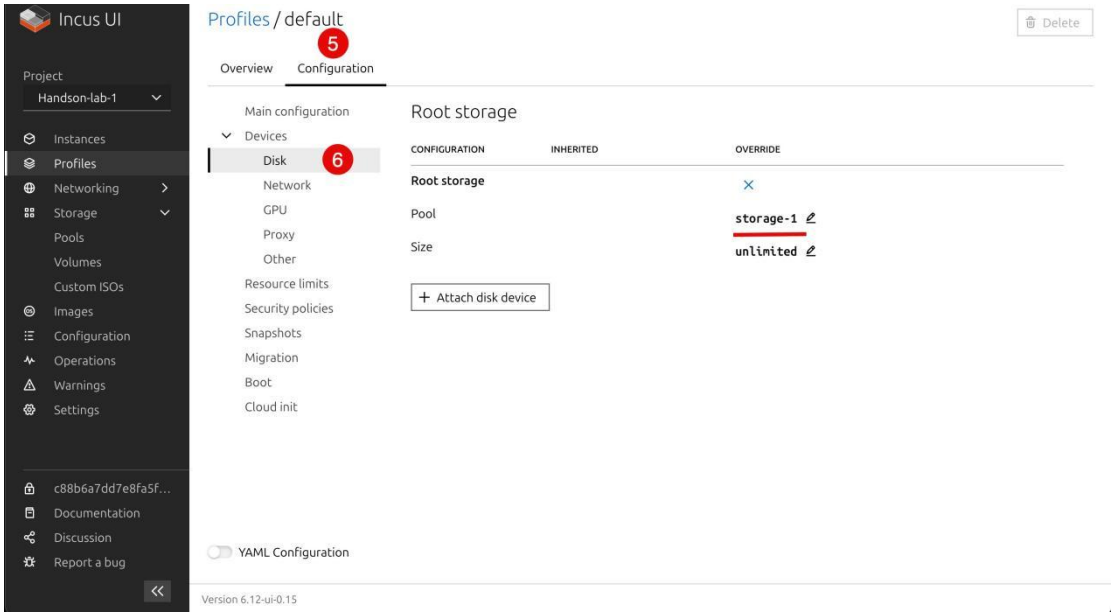
In this task we will look at the 'default profile' created for Project Handson-lab-1.

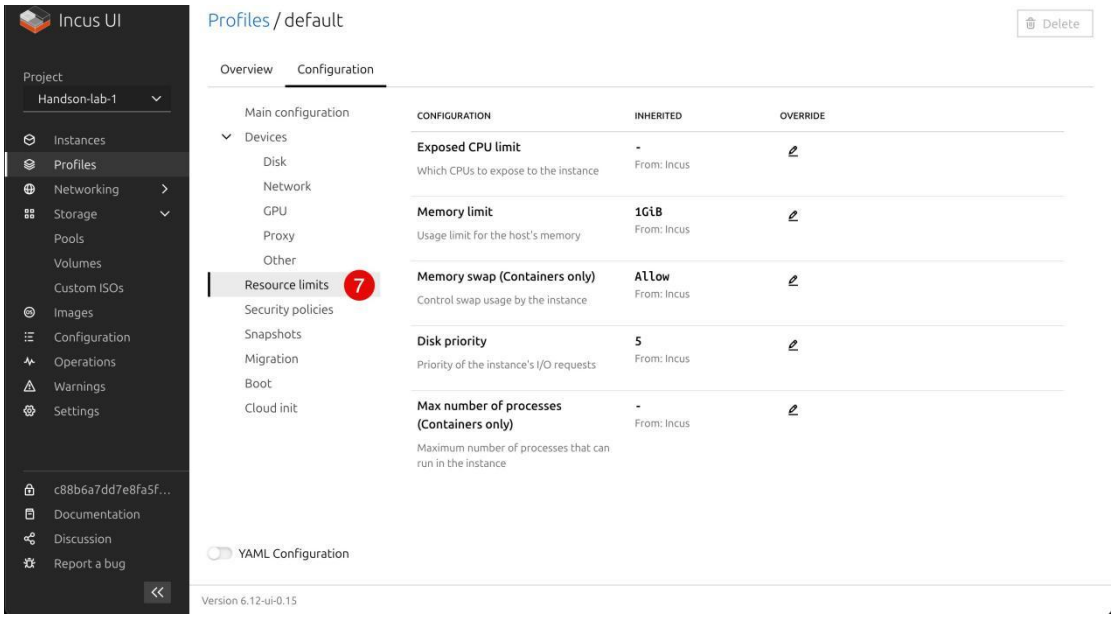
In Incus, a profile is a named set of configuration settings that can be applied to one or more instances (containers or VMs). Profiles allow you to define reusable configurations, making it easier to manage multiple instances with similar requirements.

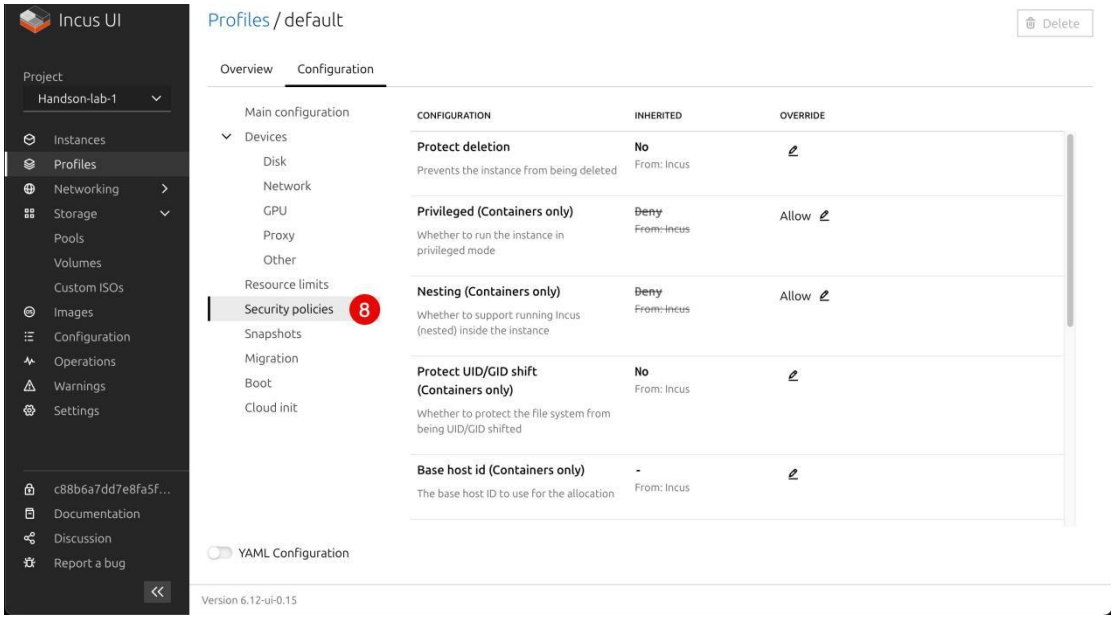
We can also create profiles at various levels namely - storage, network and projects. As part of creating a new project, incus will create a default profile. Let us take a look at it via web UI and command line.






Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Storage
Pools
Volumes
Custom ISOs
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Profiles / default
5
Delete
Overview
Configuration
Main configuration
Devices
Disk
6
Network
GPU
Proxy
Other
Resource limits
Security policies
Snapshots
Migration
Boot
Cloud init
Root storage
CONFIGURATION
INHERITED
OVERRIDE
Root storage
Pool
storage-1
Size
unlimited
+ Attach disk device
YAML Configuration
Version 6.12-ui-0.15


Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Storage
Pools
Volumes
Custom ISOs
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Profiles / default
Delete
Overview
Configuration
Main configuration
Devices
Disk
Network
GPU
Proxy
Other
Resource limits
7
Security policies
Snapshots
Migration
Boot
Cloud init
CONFIGURATION
INHERITED
OVERRIDE
Exposed CPU limit
Which CPUs to expose to the instance
-
From: Incus
Memory limit
Usage limit for the host's memory
1GiB
From: Incus
Memory swap (Containers only)
Control swap usage by the instance
Allow
From: Incus
Disk priority
Priority of the instance's I/O requests
5
From: Incus
Max number of processes (Containers only)
Maximum number of processes that can run in the instance
-
From: Incus
YAML Configuration
Version 6.12-ui-0.15

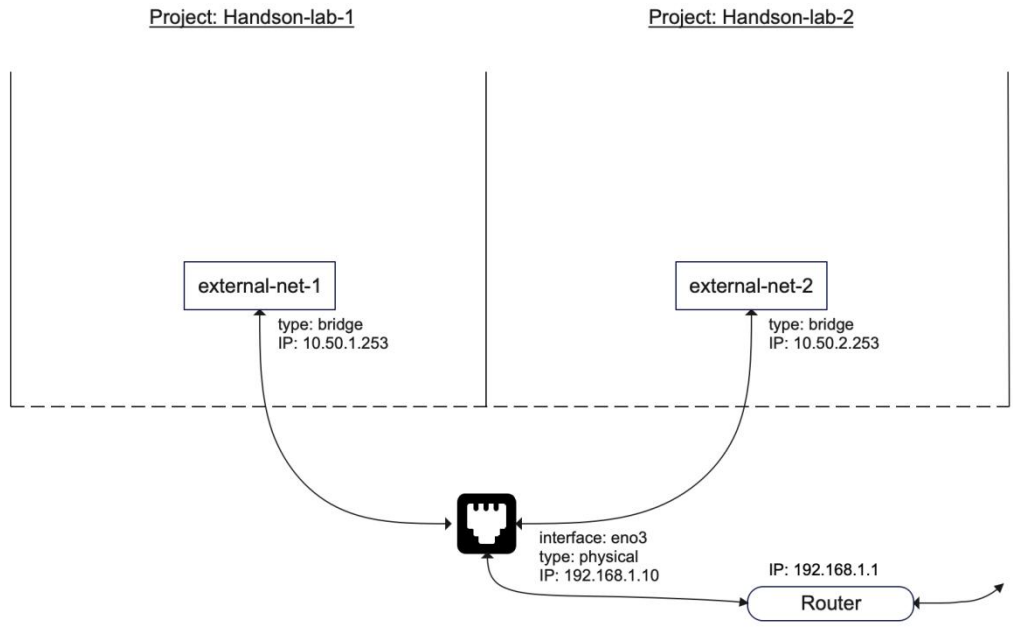


The same defaults can also be listed via command lines

```
mac% incus profile list --project Handson-lab-1
+---------+-------------------------------------------------+---------+
|  NAME   |                   DESCRIPTION                   | USED BY |
+---------+-------------------------------------------------+---------+
| default | Default Incus profile for project Handson-lab-1 | 0       |
+---------+-------------------------------------------------+---------+

mac% incus profile show default --project Handson-lab-1
config:
  security.nesting: "true"
  security.privileged: "true"
description: Default Incus profile for project Handson-lab-1
devices:
  root:
    path: /
    pool: storage-1
    type: disk
name: default
used_by: []
project: Handson-lab-1
```

Task #20: Creating external network for project: Handson-lab-1

In this task (and the ones following) we will focus primarily on the networking aspects of incus. To begin with, we will create an external network for project Handson-lab-1. Let us take a look at its network topology.



To get an overall idea of what we eventually wanted to build, the above diagram showcases the networking topology of two different projects. For now, we will focus on just the first project alone (Handson-lab-1).

Let us first switch to the project Handson-lab-1

```
mac% incus project list
+-------------------+--------+----------+-----------------+-----------------+----------+--------------+-----------------------+---------+
|      NAME         | IMAGES | PROFILES | STORAGE VOLUMES | STORAGE BUCKETS | NETWORKS | NETWORK ZONES |     DESCRIPTION       | USED BY |
+-------------------+--------+----------+-----------------+-----------------+----------+--------------+-----------------------+---------+
| Handson-lab-1     | YES    | YES      | YES             | YES             | YES      | YES          | Project Handson-lab-1 | 1       |
+-------------------+--------+----------+-----------------+-----------------+----------+--------------+-----------------------+---------+
| default (current) | YES    | YES      | YES             | YES             | YES      | YES          | Default Incus project | 4       |
+-------------------+--------+----------+-----------------+-----------------+----------+--------------+-----------------------+---------+
```

```
mac% incus project switch Handson-lab-1

mac% incus project show Handson-lab-1
config:
  features.images: "true"
```

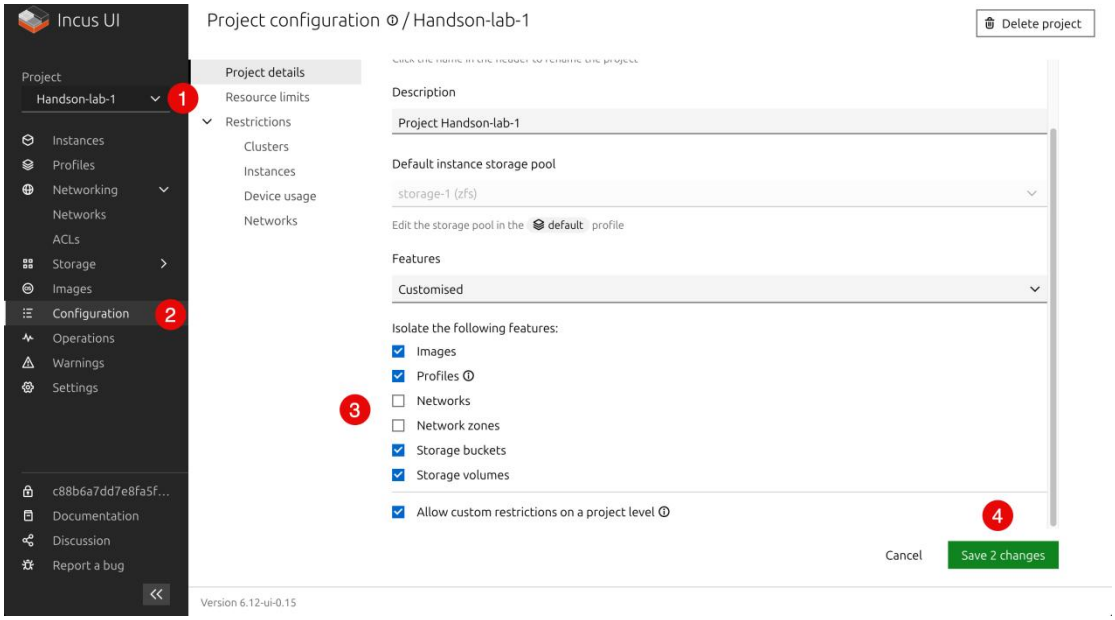
```
  features.networks: "true"
  features.networks.zones: "true"
  features.profiles: "true"
  features.storage.buckets: "true"
  features.storage.volumes: "true"
  restricted: "true"
  restricted.containers.lowlevel: allow
  restricted.containers.nesting: allow
  restricted.containers.privilege: allow
  restricted.devices.disk: managed
  restricted.devices.gpu: allow
  restricted.snapshots: allow
  restricted.virtual-machines.lowlevel: allow
description: Project Handson-lab-1
name: Handson-lab-1
used_by:
- /1.0/profiles/default?project=Handson-lab-1
```

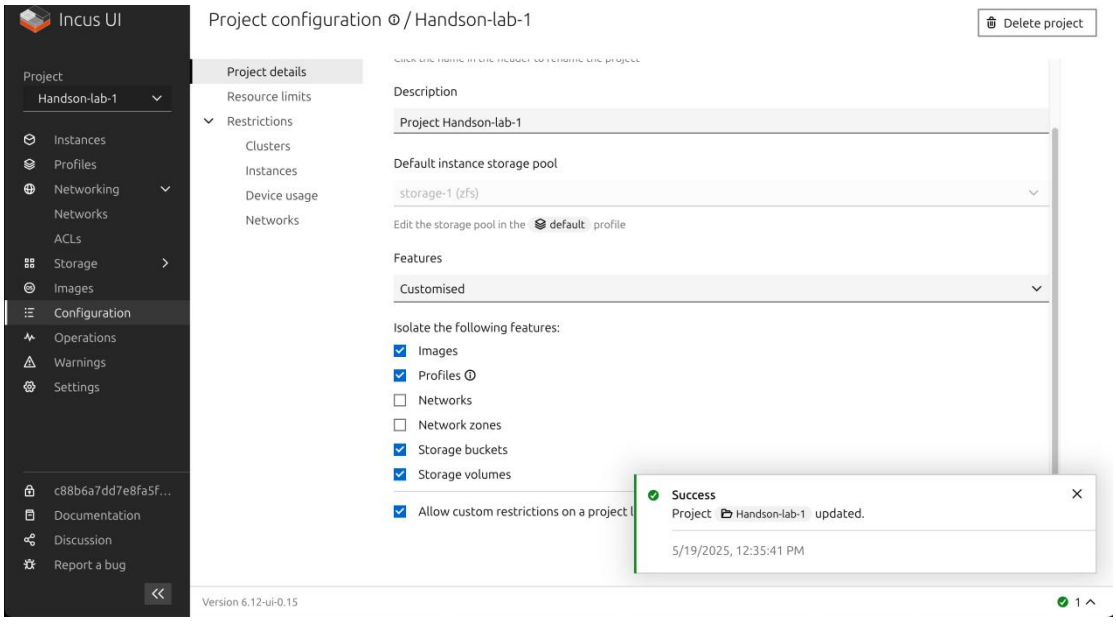
Let us take a look at the current network setup

```
mac% incus network list
+------------+----------+---------+---------------+------+-------------+---------+---------+
|   NAME     |   TYPE   | MANAGED |     IPV4      | IPV6 | DESCRIPTION | USED BY |  STATE  |
+------------+----------+---------+---------------+------+-------------+---------+---------+
| br-int     | bridge   | NO      |               |      |             | 0       |         |
+------------+----------+---------+---------------+------+-------------+---------+---------+
| docker0    | bridge   | NO      |               |      |             | 0       |         |
+------------+----------+---------+---------------+------+-------------+---------+---------+
| eno1       | physical | NO      |               |      |             | 0       |         |
+------------+----------+---------+---------------+------+-------------+---------+---------+
| eno2       | physical | NO      |               |      |             | 0       |         |
+------------+----------+---------+---------------+------+-------------+---------+---------+
| eno3       | physical | NO      |               |      |             | 0       |         |
+------------+----------+---------+---------------+------+-------------+---------+---------+
| eno4       | physical | NO      |               |      |             | 0       |         |
+------------+----------+---------+---------------+------+-------------+---------+---------+
| incusbr0   | bridge   | YES     | 10.1.1.253/24 | none |             | 1       | CREATED |
+------------+----------+---------+---------------+------+-------------+---------+---------+
| lo         | loopback | NO      |               |      |             | 0       |         |
+------------+----------+---------+---------------+------+-------------+---------+---------+
| ovs-system | unknown  | NO      |               |      |             | 0       |         |
+------------+----------+---------+---------------+------+-------------+---------+---------+
```

Let us create an external network. Note that we are NOT using the incus network bridge (incusbr0) in our setup. It means any container or a virtual machine that connects to the external network, will not go though the incusbr0 bridge but it's network traffic will directly go to the physical network interface 'eno3'. We may want to have the network topology this way to get some clarity on incus networking features and concepts.

Let us also have some flexibility in the network access among projects. To have network isolation in place we will create network (access control lists) ACL rules, which we will work on a later task (#28). For now let us de-select networks and network zones for project Handson-lab-1 using web UI.





In the following command we pass all the required parameters to create an external network (named 'external-net-1') - exclusively dedicated to Project Handson-lab-1.

```
mac% incus network create external-net-1 ipv4.address=10.50.1.253/24 ipv4.nat=true
ipv6.address=none ipv4.dhcp.ranges=10.50.1.101-10.50.1.200 --project=Handson-lab-1
```

```
mac% incus network list
+----------------+----------+---------+----------------+------+-------------+---------+---------+
|     NAME       |   TYPE   | MANAGED |      IPV4      | IPV6 | DESCRIPTION | USED BY |  STATE  |
+----------------+----------+---------+----------------+------+-------------+---------+---------+
| br-int         | bridge   | NO      |                |      |             | 0       |         |
+----------------+----------+---------+----------------+------+-------------+---------+---------+
| docker0        | bridge   | NO      |                |      |             | 0       |         |
+----------------+----------+---------+----------------+------+-------------+---------+---------+
| eno1           | physical | NO      |                |      |             | 0       |         |
+----------------+----------+---------+----------------+------+-------------+---------+---------+
| eno2           | physical | NO      |                |      |             | 0       |         |
+----------------+----------+---------+----------------+------+-------------+---------+---------+
| eno3           | physical | NO      |                |      |             | 0       |         |
+----------------+----------+---------+----------------+------+-------------+---------+---------+
| eno4           | physical | NO      |                |      |             | 0       |         |
+----------------+----------+---------+----------------+------+-------------+---------+---------+
| external-net-1 | bridge   | YES     | 10.50.1.253/24 | none |             | 0       | CREATED |
+----------------+----------+---------+----------------+------+-------------+---------+---------+
| incusbr0       | bridge   | YES     | 10.1.1.253/24  | none |             | 1       | CREATED |
+----------------+----------+---------+----------------+------+-------------+---------+---------+
| lo             | loopback | NO      |                |      |             | 0       |         |
+----------------+----------+---------+----------------+------+-------------+---------+---------+
| ovs-system     | unknown  | NO      |                |      |             | 0       |         |
+----------------+----------+---------+----------------+------+-------------+---------+---------+
```

```
mac$ incus network show external-net-1
config:
  ipv4.address: 10.50.1.253/24
  ipv4.dhcp.ranges: 10.50.1.101-10.50.1.200
  ipv4.nat: "true"
  ipv6.address: none
description: ""
name: external-net-1
type: bridge
used_by: []
managed: true
status: Created
locations:
```

```
- none
project: default
```

Let us go ahead and edit the Description field of the newly created external-net-1 to make it verbatim

```
mac$ incus network edit external-net-1
config:
  ipv4.address: 10.50.1.253/24
  ipv4.dhcp.ranges: 10.50.1.101-10.50.1.200
  ipv4.nat: "true"
  ipv6.address: none
description: "uplink for Handson-lab-1"
name: external-net-1
type: bridge
used_by: []
managed: true
status: Created
locations:
- none
project: default
```

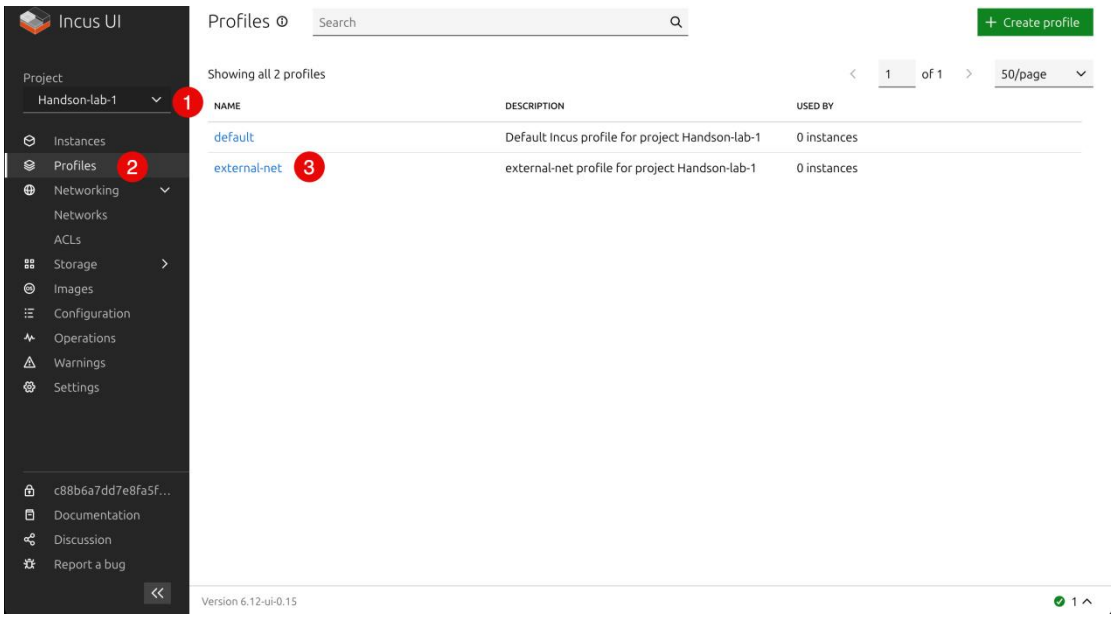
```
mac$ incus network list
+---------------+----------+---------+---------------+------+-------------------------+---------+---------+
|     NAME      |   TYPE   | MANAGED |     IPV4      | IPV6 |      DESCRIPTION        | USED BY |  STATE  |
+---------------+----------+---------+---------------+------+-------------------------+---------+---------+
| br-int        | bridge   | NO      |               |      |                         | 0       |         |
+---------------+----------+---------+---------------+------+-------------------------+---------+---------+
| docker0       | bridge   | NO      |               |      |                         | 0       |         |
+---------------+----------+---------+---------------+------+-------------------------+---------+---------+
| eno1          | physical | NO      |               |      |                         | 0       |         |
+---------------+----------+---------+---------------+------+-------------------------+---------+---------+
| eno2          | physical | NO      |               |      |                         | 0       |         |
+---------------+----------+---------+---------------+------+-------------------------+---------+---------+
| eno3          | physical | NO      |               |      |                         | 0       |         |
+---------------+----------+---------+---------------+------+-------------------------+---------+---------+
| eno4          | physical | NO      |               |      |                         | 0       |         |
+---------------+----------+---------+---------------+------+-------------------------+---------+---------+
| external-net-1 | bridge   | YES     | 10.50.1.253/24  | none | uplink for Handson-lab-1 | 0       | CREATED |
+---------------+----------+---------+---------------+------+-------------------------+---------+---------+
| incusbr0      | bridge   | YES     | 10.1.1.253/24  | none |                         | 1       | CREATED |
+---------------+----------+---------+---------------+------+-------------------------+---------+---------+
| lo            | loopback | NO      |               |      |                         | 0       |         |
+---------------+----------+---------+---------------+------+-------------------------+---------+---------+
```

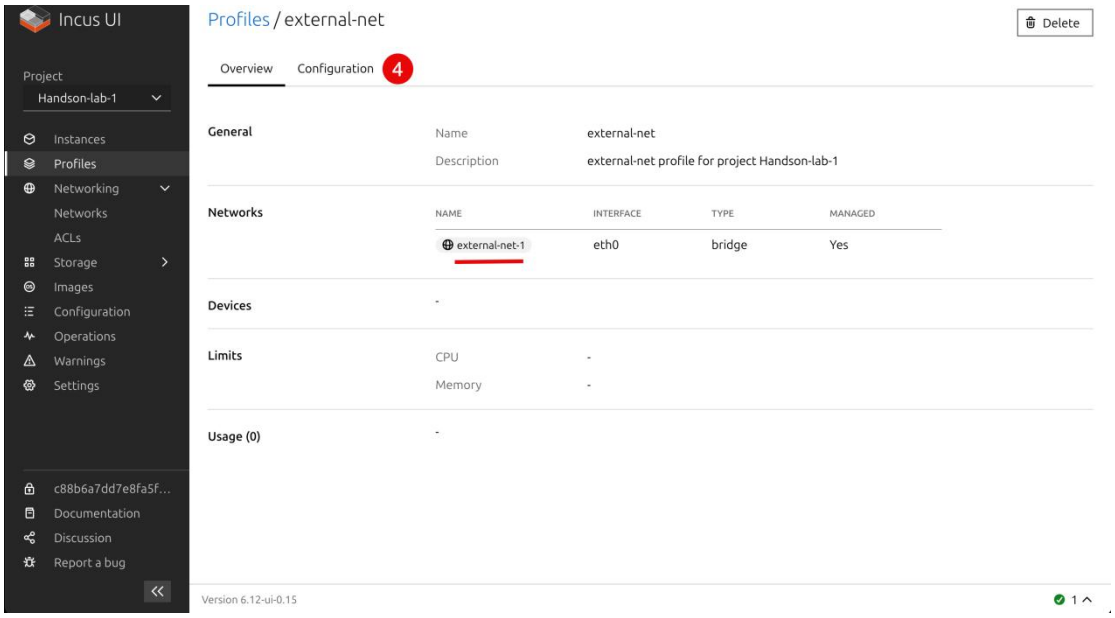
```
| ovs-system     | unknown  | NO      |               |      |                          | 0      |         |
+----------------+----------+---------+---------------+------+--------------------------+--------+---------+
```

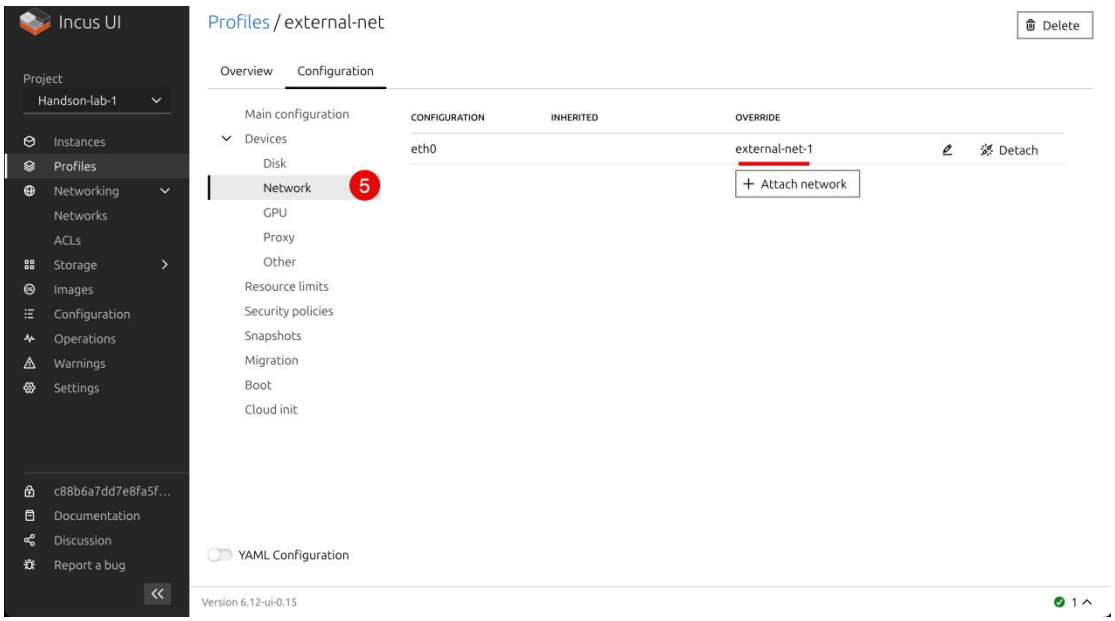
Now that we have created the externet-net-1, let us create a network profile for it. By passing this network profile while launching a container or a virtual machine, will make that instance connect to the external-net-1. This concept will be clear once we see an example (which we will carry it out as a separate task).

```
mac$ incus profile create external-net
mac$ incus profile edit external-net
config: {}
description: external-net profile for project Handson-lab-1
devices:
  eth0:
    name: eth0
    network: external-net-1
    type: nic
name: external-net
used_by: []
project: Handson-lab-1
```

The network profile can also be viewed via web UI





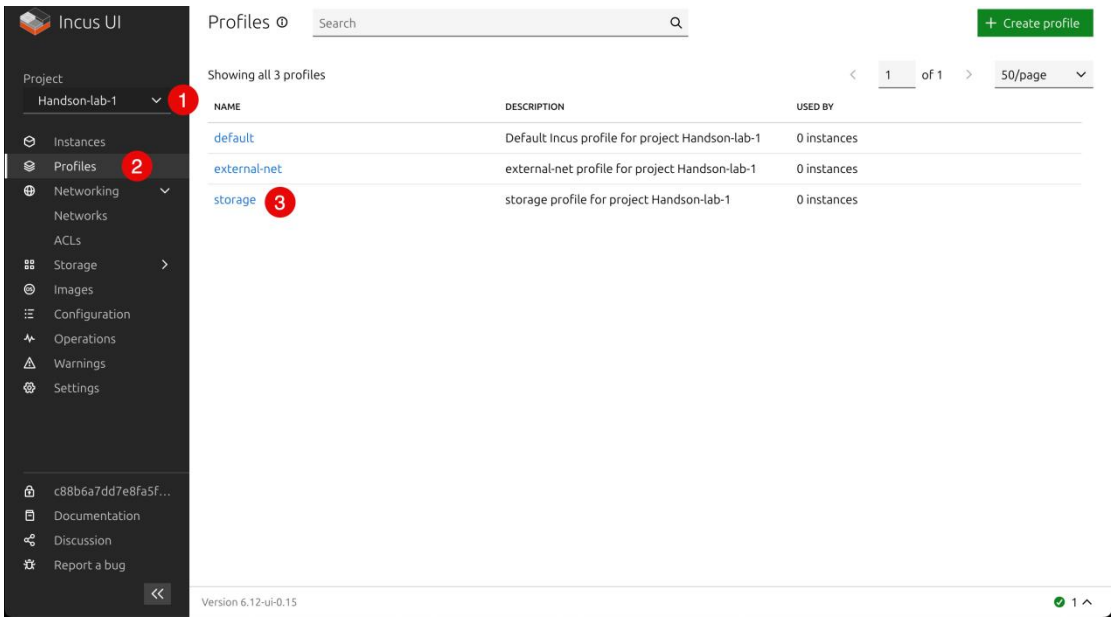


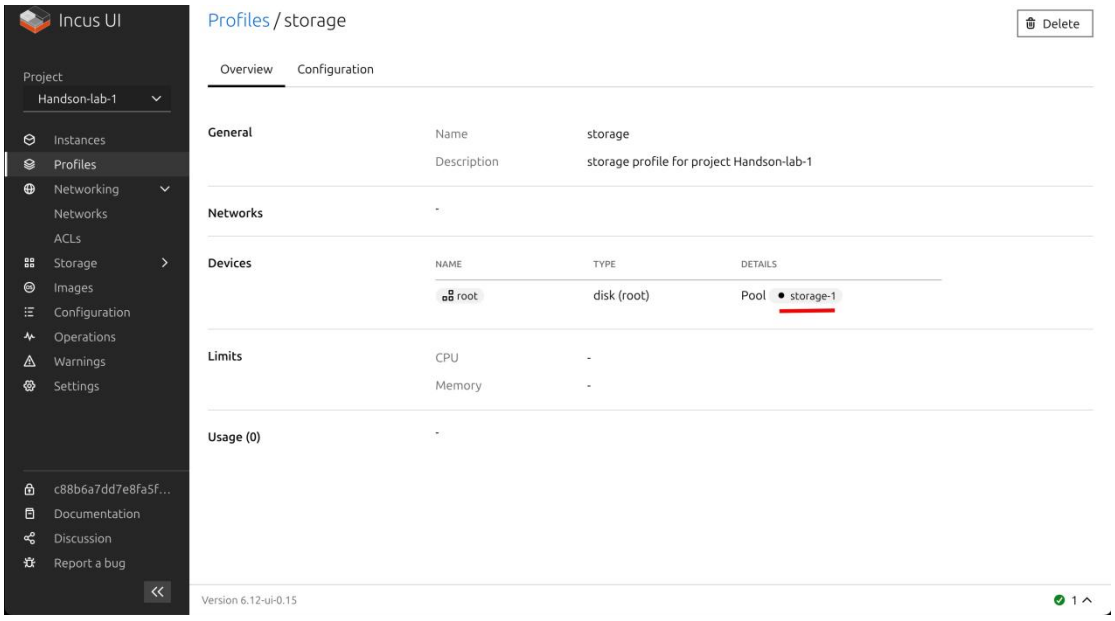
Similar to creating a network profile, let us create a storage profile where we define the storage pool dedicated to project Handson-lab-1 alone (not sharing with any other projects, delivers storage isolation and multi-tenancy)

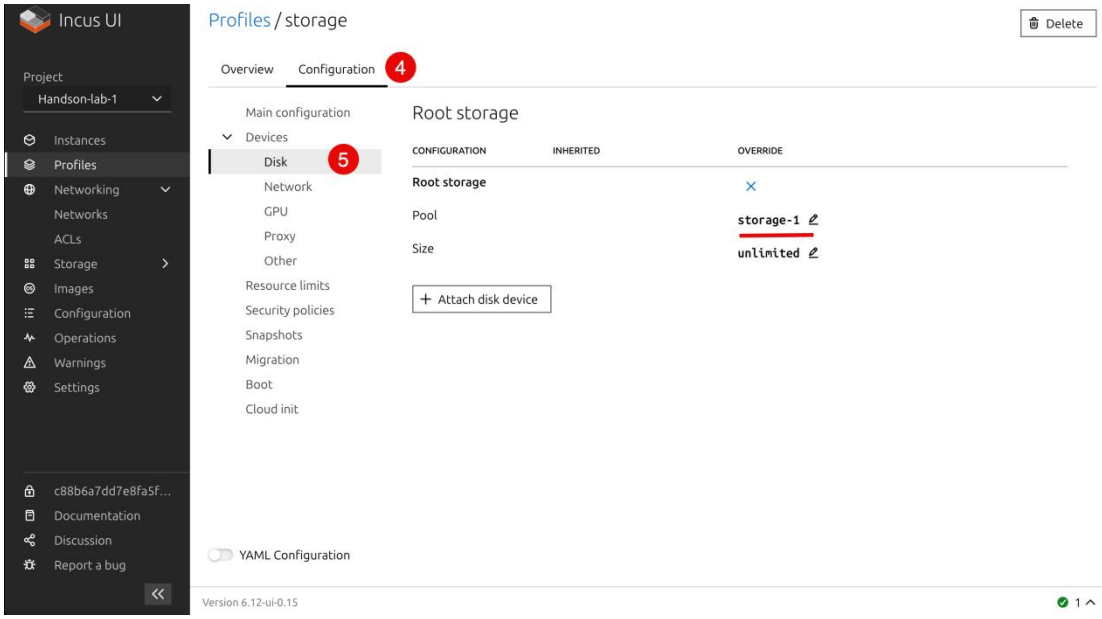
```
mac$ incus profile create storage
mac$ incus profile edit storage
config: {}
description: storage profile for project Handson-lab-1
devices:
  root:
    path: /
    pool: storage-1
```

```
    type: disk
name: storage
used_by: []
project: Handson-lab-1
```

Let us again verify this via web UI






Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Networks
ACLs
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Profiles / storage
Delete
Overview
Configuration
4
Main configuration
Devices
Disk
5
Network
GPU
Proxy
Other
Resource limits
Security policies
Snapshots
Migration
Boot
Cloud init
Root storage
CONFIGURATION
INHERITED
OVERRIDE
Root storage
Pool
storage-1
Size
unlimited
+ Attach disk device
YAML Configuration
Version 6.12-ui-0.15
1

Task #21: Sanity testing external network of project: Handson-lab-1

Now that we have completed creating an external network for project Handson-lab-1, let us launch a container instance (c1), passing the profiles that we have created and see how the container consume resources.

```
mac% incus launch images:ubuntu/24.04 c1 --profile=external-net --profile=storage -d
root,size=5GiB

# incus list
+------+---------+--------------------+------+-----------+-----------+
| NAME |  STATE  |       IPV4         | IPV6 |   TYPE    | SNAPSHOTS |
+------+---------+--------------------+------+-----------+-----------+
| c1   | RUNNING | 10.50.1.181 (eth0) |      | CONTAINER | 0         |
+------+---------+--------------------+------+-----------+-----------+
```

The first thing to notice is the IP address assigned to the container c1 (10.50.1.181). You may recollect that when we created the external network (external-net-1) in the previous task, we passed a parameter 'ipv4.dhcp.ranges=10.50.1.101-10.50.1.200' to the 'incus network create' command. We can now confirm that the ip address assigned to the container (c1) is provided from that range.

```
mac% incus info c1
Name: c1
Description:
Status: RUNNING
Type: container
Architecture: x86_64
PID: 84472
Created: 2025/05/19 21:46 UTC
Last Used: 2025/05/19 21:46 UTC
Started: 2025/05/19 21:46 UTC

Resources:
  Processes: 12
  Disk usage:
    root: 1.06MiB
  CPU usage:
    CPU usage (in seconds): 2
  Memory usage:
    Memory (current): 44.16MiB
  Network usage:
    eth0:
```

```
    Type: broadcast
    State: UP
    Host interface: veth4639fcf4
    MAC address: 10:66:6a:dd:75:84
    MTU: 1500
    Bytes received: 389B
    Bytes sent: 1.44kB
    Packets received: 2
    Packets sent: 16
    IP addresses:
      inet:  10.50.1.181/24 (global)
      inet6: fe80::1266:6aff:fedd:7584/64 (link)
  lo:
    Type: loopback
    State: UP
    MTU: 65536
    Bytes received: 0B
    Bytes sent: 0B
    Packets received: 0
    Packets sent: 0
    IP addresses:
      inet:  127.0.0.1/8 (local)
      inet6: ::1/128 (local)
```

Let us get inside the container and run a few commands.

```
mac% incus shell c1
root@c1:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
21: eth0@if22: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group
default qlen 1000
    link/ether 10:66:6a:dd:75:84 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.50.1.181/24 metric 100 brd 10.50.1.255 scope global dynamic eth0
       valid_lft 2262sec preferred_lft 2262sec
    inet6 fe80::1266:6aff:fedd:7584/64 scope link
       valid_lft forever preferred_lft forever

root@c1:~# apt update -y
root@c1:~# apt upgrade -y
```

```
root@c1:~# apt install -y traceroute
```

Let us run the traceroute command reaching out to website zabbly.com (maintainers of project incus).

```
root@c1:~# traceroute zabbly.com
traceroute to zabbly.com (45.45.148.7), 30 hops max, 60 byte packets
 1  _gateway.incus (10.50.1.253)  0.066 ms  0.025 ms  0.020 ms
 2  _gateway (192.168.1.1)  0.333 ms  0.874 ms  0.828 ms
 3  10.61.209.3 (10.61.209.3)  19.560 ms  19.519 ms  19.478 ms
 4  po-323-339-rur201.sanjose.ca.sfba.comcast.net (96.216.8.237)  19.359 ms * *
[ removed the rest of the output ]
```

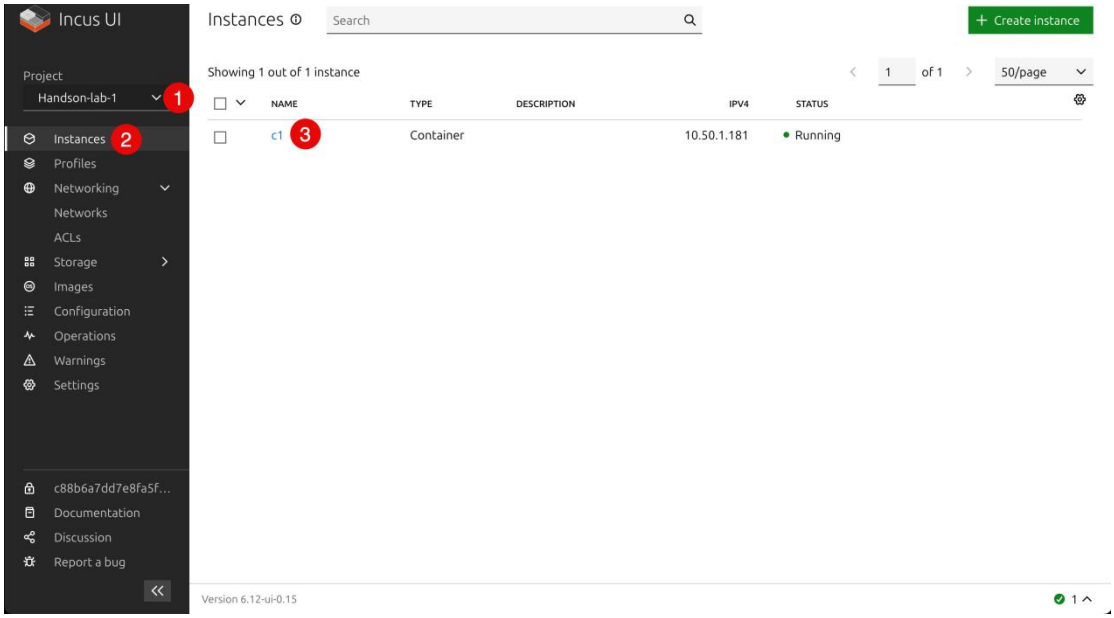
To reach zabbly.com (45.45.148.7) the network traffic goes to external-net-1 first (10.50.1.253). From there it goes to 192.168.1.1 which is the ip address of the router on the physical network.

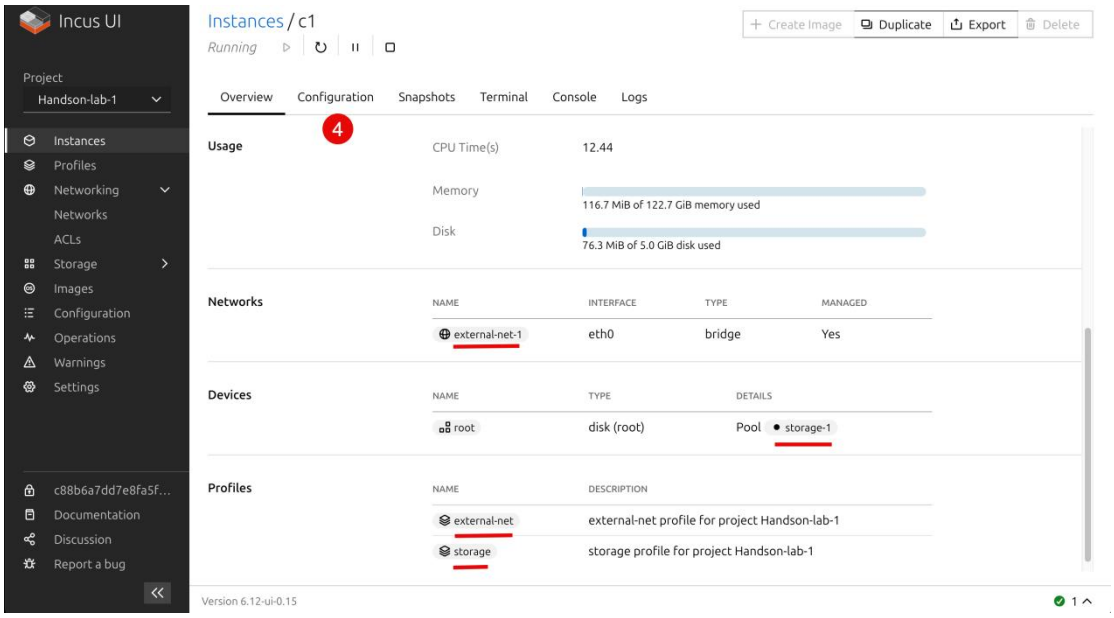
It is important to note that the network traffic did NOT go through incus network bridge (incusbr0), which was our intention when we designed the network topology in the previous task.

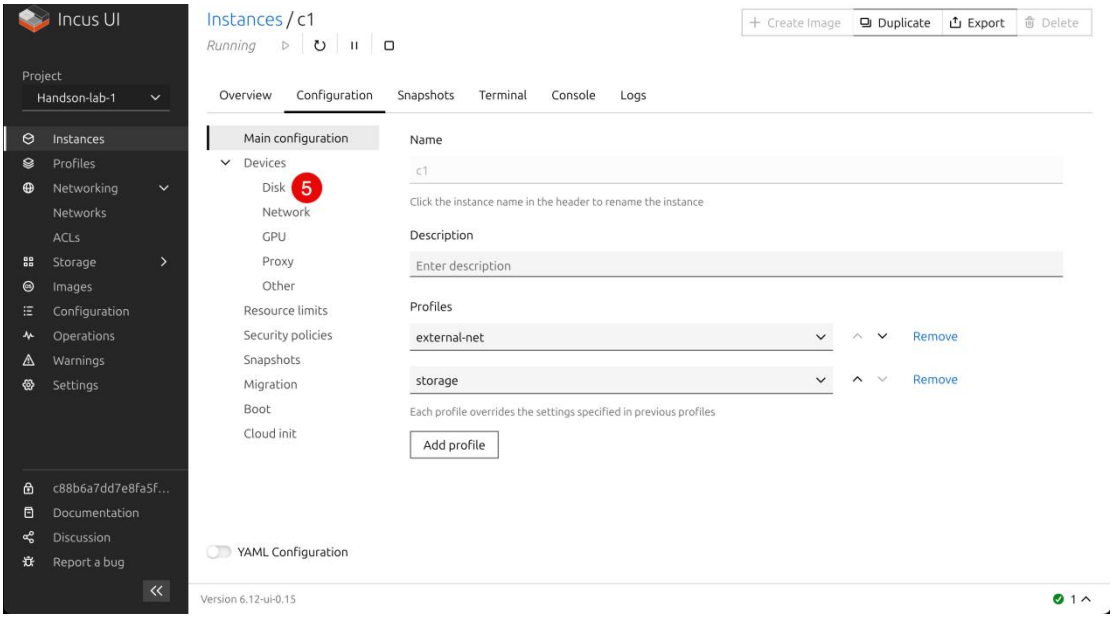
Let us take a look at how the storage is spread cross for the container c1.

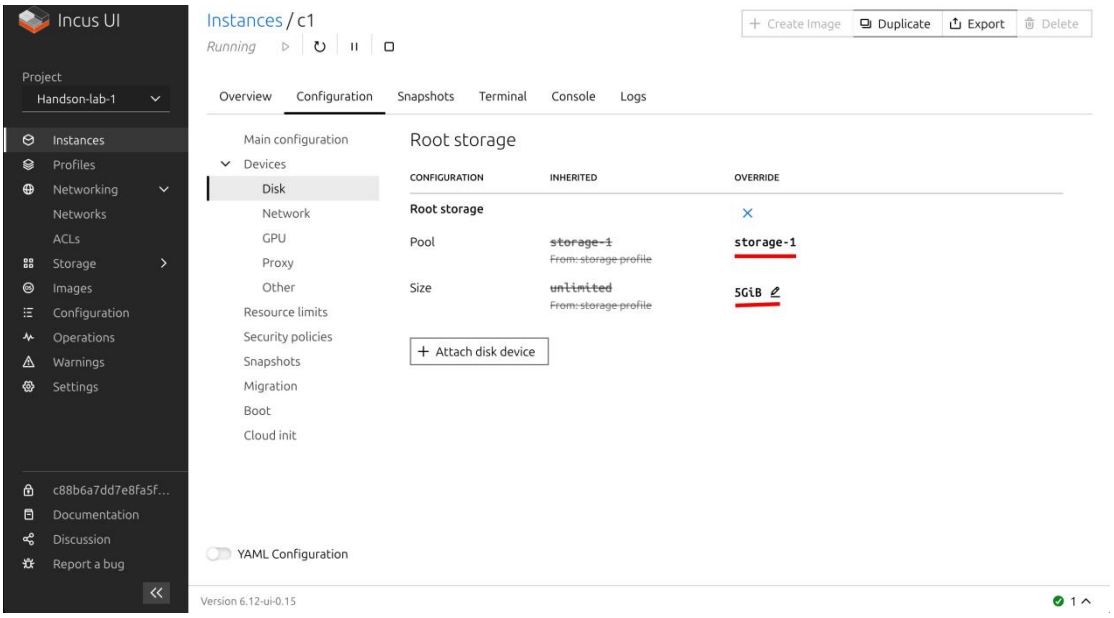
```
mac% incus storage show storage-1
config:
  source: zpool-1
  volatile.initial_source: zpool-1
  zfs.pool_name: zpool-1
description: Storage for Containers, VMs of Handson-lab-1
name: storage-1
driver: zfs
used_by:
-
/1.0/images/13fbccd9a71b3110757f537ed43f3bd23d56f655c9915a654baf25cb6f1da80e?project=Ha
ndson-lab-1
- /1.0/instances/c1?project=Handson-lab-1
- /1.0/profiles/default?project=Handson-lab-1
- /1.0/profiles/storage?project=Handson-lab-1
status: Created
locations:
- none
```

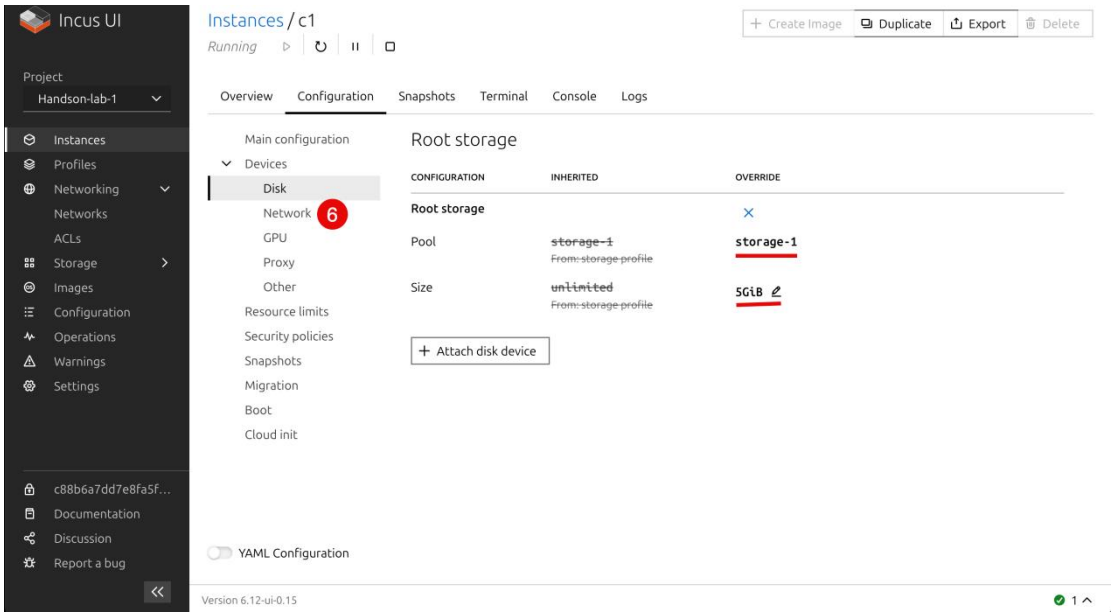
And finally let us look at the same container via web UI

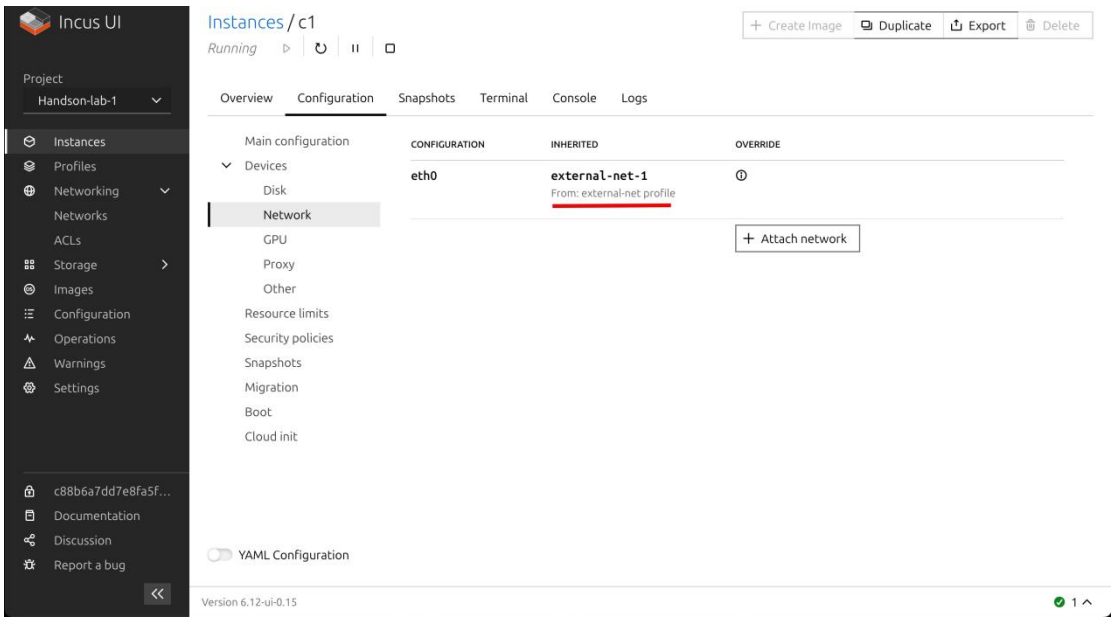

Incus UI
Project
Handson-lab-1
1
Instances
2
Profiles
Networking
Networks
ACLs
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Instances
Search
Create instance
Showing 1 out of 1 instance
1
of 1
50/page
NAME
TYPE
DESCRIPTION
IPV4
STATUS
c1
3
Container
10.50.1.181
Running
Version 6.12-ui-0.15
1

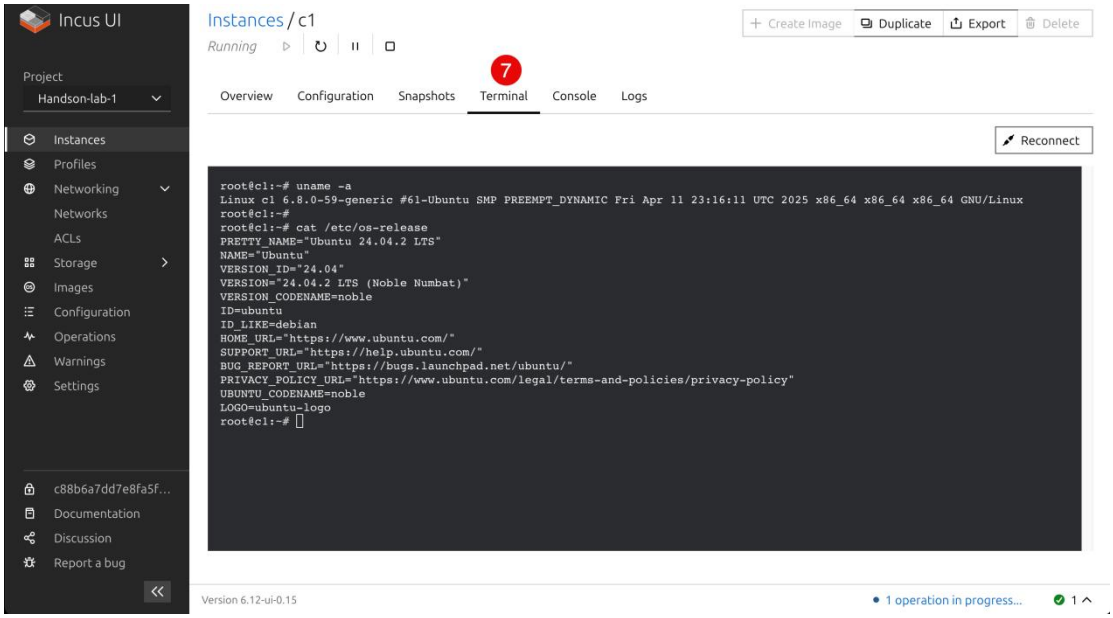

Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Networks
ACLs
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Instances / c1
Running
Create Image
Duplicate
Export
Delete
Overview
Configuration
4
Snapshots
Terminal
Console
Logs
Usage
CPU Time(s)
12.44
Memory
116.7 MiB of 122.7 GiB memory used
Disk
76.3 MiB of 5.0 GiB disk used
Networks
NAME
INTERFACE
TYPE
MANAGED
external-net-1
eth0
bridge
Yes
Devices
NAME
TYPE
DETAILS
root
disk (root)
Pool
storage-1
Profiles
NAME
DESCRIPTION
external-net
external-net profile for project Handson-lab-1
storage
storage profile for project Handson-lab-1
Version 6.12-ui-0.15
1

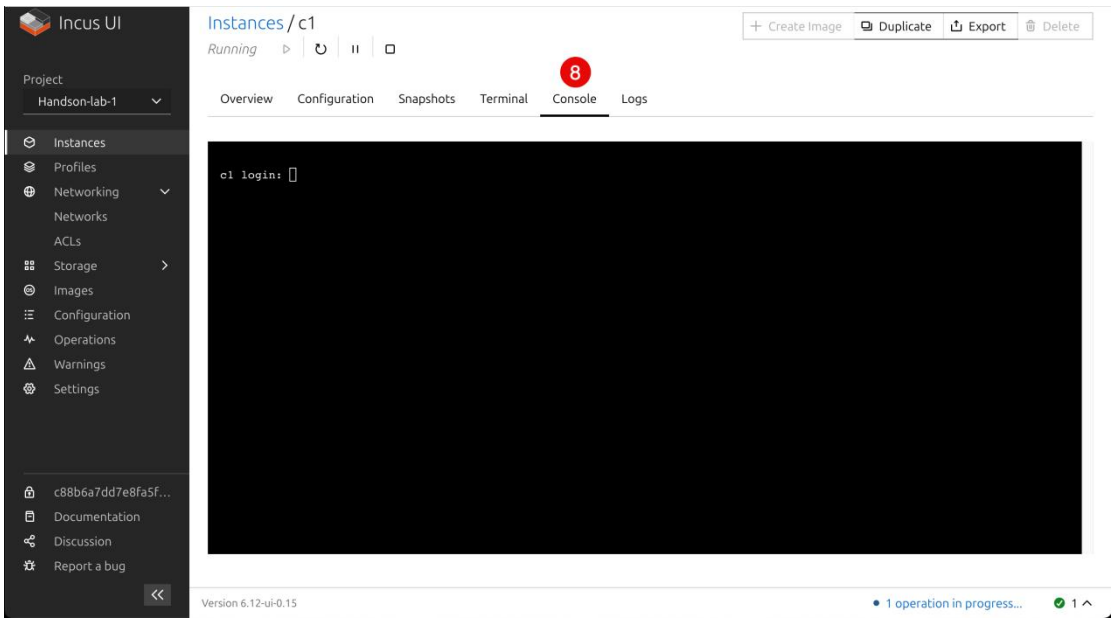

Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Networks
ACLs
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Instances / c1
Running
Create Image
Duplicate
Export
Delete
Overview
Configuration
Snapshots
Terminal
Console
Logs
Main configuration
Devices
Disk
5
Network
GPU
Proxy
Other
Resource limits
Security policies
Snapshots
Migration
Boot
Cloud init
Name
c1
Click the instance name in the header to rename the instance
Description
Enter description
Profiles
external-net
Remove
storage
Remove
Each profile overrides the settings specified in previous profiles
Add profile
YAML Configuration
Version 6.12-ui-0.15
1


Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Networks
ACLs
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Instances / c1
Running
Create Image
Duplicate
Export
Delete
Overview
Configuration
Snapshots
Terminal
Console
Logs
Main configuration
Devices
Disk
Network
GPU
Proxy
Other
Resource limits
Security policies
Snapshots
Migration
Boot
Cloud init
Root storage
CONFIGURATION
INHERITED
OVERRIDE
Root storage
Pool
storage-1
From: storage profile
storage-1
Size
unlimited
From: storage profile
5GiB
Attach disk device
YAML Configuration
Version 6.12-ui-0.15
1


Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Networks
ACLs
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Instances / c1
Running
Create Image
Duplicate
Export
Delete
Overview
Configuration
Snapshots
Terminal
Console
Logs
Main configuration
Devices
Disk
Network
6
GPU
Proxy
Other
Resource limits
Security policies
Snapshots
Migration
Boot
Cloud init
Root storage
CONFIGURATION
INHERITED
OVERRIDE
Root storage
Pool
storage-1
From: storage profile
storage-1
Size
unlimited
From: storage profile
5GiB
Attach disk device
YAML Configuration
Version 6.12-ui-0.15
1


Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Networks
ACLs
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Instances / c1
Running
Create Image
Duplicate
Export
Delete
Overview
Configuration
Snapshots
Terminal
Console
Logs
Main configuration
Devices
Disk
Network
GPU
Proxy
Other
Resource limits
Security policies
Snapshots
Migration
Boot
Cloud init
CONFIGURATION
INHERITED
OVERRIDE
eth0
external-net-1
From: external-net profile
Attach network
YAML Configuration
Version 6.12-ui-0.15
1


Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Networks
ACLs
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Instances / c1
Running
Create Image
Duplicate
Export
Delete
7
Overview
Configuration
Snapshots
Terminal
Console
Logs
Reconnect
root@c1:~# uname -a
Linux c1 6.8.0-59-generic #61-Ubuntu SMP PREEMPT_DYNAMIC Fri Apr 11 23:16:11 UTC 2025 x86_64 x86_64 x86_64 GNU/Linux
root@c1:~#
root@c1:~# cat /etc/os-release
PRETTY_NAME="Ubuntu 24.04.2 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04.2 LTS (Noble Numbat)"
VERSION_CODENAME=noble
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=noble
LOGO=ubuntu-logo
root@c1:~#
Version 6.12-ui-0.15
1 operation in progress...
1

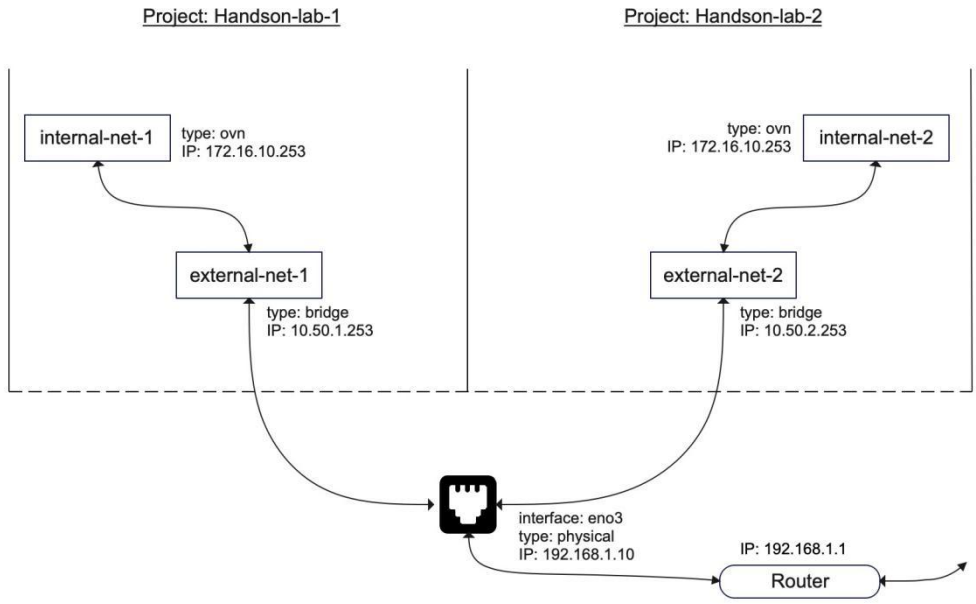

Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Networks
ACLs
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Instances / c1
Running
Create Image
Duplicate
Export
Delete
8
Overview
Configuration
Snapshots
Terminal
Console
Logs
c1 login:
Version 6.12-ui-0.15
1 operation in progress...
1

Task #22: Creating an Internal network for project: Handson-lab-1

Let us now move onto creating a second network dedicate to the project Handson-lab-1. Let us call it 'internal-net-1'. The reason we want to name it 'internal network' is because we want to limit the scope of this network, exclusively for all the containers and virtual machines that run within project Handson-lab-1.

The second important concept we want to understand in this task is about OVN networks (Open Virtual Networking). By using the OVN network type for internal-net-1, we can repurpose the same IP address range across Projects. The importance of this feature may not stand out right away immediately. But as we make progress, we will see its power and advantage down the line when we build more usecases/tasks aound it.

Let us take a look at its network topology where we are introducing internal network and its dependency on the external network that we already have in place.



Let us set an ipv4 ovn range on the external network

```
mac% incus network set external-net-1 ipv4.ovn.ranges=10.50.1.1-10.50.1.25
```

Let us enable ipv4.routing on the external network

```
mac% incus network set external-net-1 ipv4.routing=true
```

Let us go ahead and create the internal network of type OVN

```
mac% incus network create internal-net-1 --type=ovn ipv4.address=172.16.10.253/24
```

```
ipv4.nat=true ipv6.address=none network=external-net-1
```

We will set a dhcp range as well for the internal network

Let us edit the description of internal network to make it verbatim

```
mac% incus network edit internal-net-1
config:
  bridge.mtu: "1500"
  ipv4.address: 172.16.10.253/24
  ipv4.dhcp.ranges: 172.16.10.101-172.16.10.200
  ipv4.nat: "true"
  ipv6.address: none
  network: external-net-1
  volatile.network.ipv4.address: 10.50.1.1
description: "routes via external-net-1"
name: internal-net-1
type: ovn
used_by: []
managed: true
status: Created
locations:
- none
project: default
```

```
mac% incus network list
+---------------+--------+---------+-----------------+------+--------------------------+---------+---------+
|     NAME      |  TYPE  | MANAGED |      IPV4       | IPV6 |       DESCRIPTION        | USED BY |  STATE  |
+---------------+--------+---------+-----------------+------+--------------------------+---------+---------+
| external-net-1 | bridge | YES    | 10.50.1.253/24   | none | uplink for Handson-lab-1  | 3       | CREATED |
+---------------+--------+---------+-----------------+------+--------------------------+---------+---------+
| incusbr0      | bridge | YES     | 10.1.1.253/24    | none |                           | 1       | CREATED |
+---------------+--------+---------+-----------------+------+--------------------------+---------+---------+
| internal-net-1 | ovn    | YES     | 172.16.10.253/24  | none | routes via external-net-1 | 0       | CREATED |
+---------------+--------+---------+-----------------+------+--------------------------+---------+---------+
```

While we are at it, we will go ahead and make changes to the description of incus bridge as well (incusbr0)

```
mac% incus network edit incusbr0
config:
  ipv4.address: 10.1.1.253/24
  ipv4.nat: "true"
  ipv6.address: none
```

```
description: “routes traffic via eno3”
name: incusbr0
type: bridge
used_by:
- /1.0/profiles/default
managed: true
status: Created
locations:
- none
project: default
```

```
mac% incus network list
+----------------+--------+---------+-----------------+------+---------------------------+---------+---------+
|     NAME       |  TYPE  | MANAGED |      IPV4       | IPV6 |       DESCRIPTION         | USED BY |  STATE  |
+----------------+--------+---------+-----------------+------+---------------------------+---------+---------+
| external-net-1 | bridge | YES     | 10.50.1.253/24  | none | uplink for Handson-lab-1  | 3       | CREATED |
+----------------+--------+---------+-----------------+------+---------------------------+---------+---------+
| incusbr0       | bridge | YES     | 10.1.1.253/24   | none | routes traffic via eno3   | 1       | CREATED |
+----------------+--------+---------+-----------------+------+---------------------------+---------+---------+
| internal-net-1 | ovn    | YES     | 172.16.10.253/24 | none | routes via external-net-1 | 0       | CREATED |
+----------------+--------+---------+-----------------+------+---------------------------+---------+---------+
```

Similar to external network, let us create a profile for the internal network as well.

```
mac% incus profile create internal-net
```

We will make the description field verbatim

```
mac% incus profile edit internal-net
config:
description: "internal network profile for project Handson-lab-1"
devices:
  eth0:
    name: eth0
    network: internal-net-1
    type: nic
name: internal-net
used_by: []
project: Handson-lab-1
```

Task #23: Sanity testing internal network of project: Handson-lab-1

Now that we have completed creating an internal network for project Handson-lab-1, let to launch a container instance, pass the profiles that we have created for internal network and see how the container consumes resources.

```
mac% incus project switch Handson-lab-1

mac% incus launch images:ubuntu/24.04 c2 --profile=internal-net --profile=storage -d
root,size=5GiB

mac% incus list
+------+---------+---------------------+------+-----------+-----------+
| NAME |  STATE  |        IPV4         | IPV6 |   TYPE    | SNAPSHOTS |
+------+---------+---------------------+------+-----------+-----------+
| c1   | RUNNING | 10.50.1.181 (eth0)  |      | CONTAINER | 0         |
+------+---------+---------------------+------+-----------+-----------+
| c2   | RUNNING | 172.16.10.2 (eth0)  |      | CONTAINER | 0         |
+------+---------+---------------------+------+-----------+-----------+
```

Note the IP address assigned to the container c2 (172.16.10.2). You may recollect that when we created the internal network (internal-net-1) in the previous task, we passed a parameter 'ipv4.dhcp.ranges=172.16.10.100-172.16.10.200' to the 'incus network create' command. We can now confirm that the ip address assigned to the container (c2) is provided from that range.

```
mac% incus info c2
Name: c2
Description:
Status: RUNNING
Type: container
Architecture: x86_64
PID: 88988
Created: 2025/05/20 15:58 UTC
Last Used: 2025/05/20 15:58 UTC
Started: 2025/05/20 15:58 UTC

Resources:
  Processes: 12
  Disk usage:
    root: 1.05MiB
  CPU usage:
    CPU usage (in seconds): 2
  Memory usage:
```

```
    Memory (current): 43.95MiB
  Network usage:
    eth0:
      Type: broadcast
      State: UP
      Host interface: veth4c60ac20
      MAC address: 10:66:6a:ca:9e:f1
      MTU: 1500
      Bytes received: 688B
      Bytes sent: 1.58kB
      Packets received: 2
      Packets sent: 14
      IP addresses:
        inet:  172.16.10.2/24 (global)
        inet6: fe80::1266:6aff:feca:9ef1/64 (link)
    lo:
      Type: loopback
      State: UP
      MTU: 65536
      Bytes received: 0B
      Bytes sent: 0B
      Packets received: 0
      Packets sent: 0
      IP addresses:
        inet:  127.0.0.1/8 (local)
        inet6: ::1/128 (local)
```

Let us get inside the container and run a few commands.

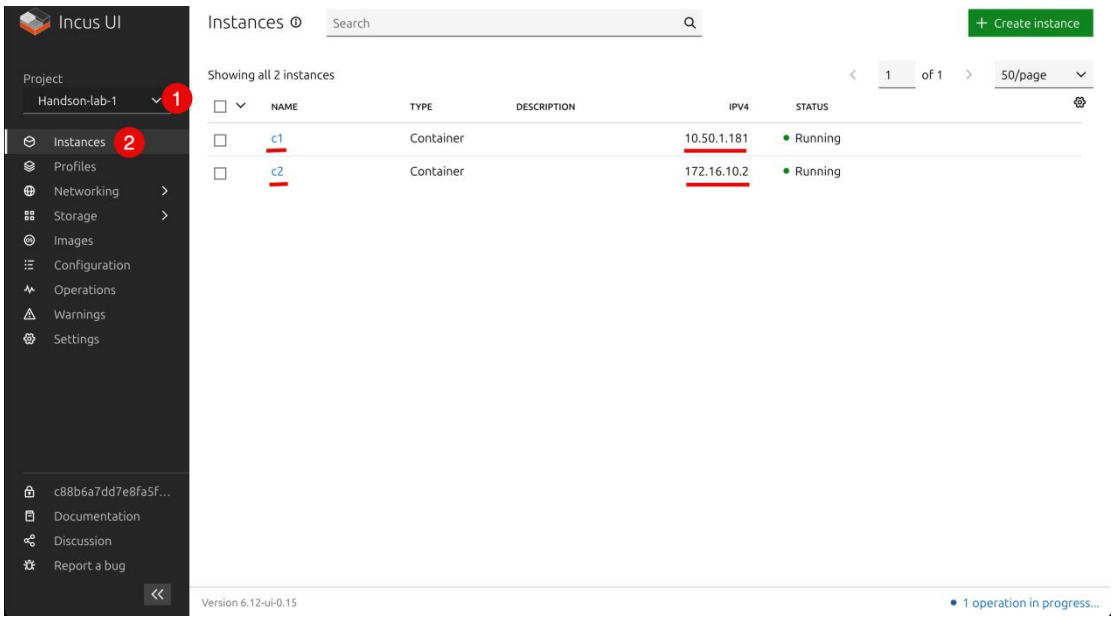
```
mac% incus shell c2
root@c2:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
26: eth0@if27: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group
default qlen 1000
    link/ether 10:66:6a:ca:9e:f1 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.16.10.2/24 metric 100 brd 172.16.10.255 scope global dynamic eth0
       valid_lft 3243sec preferred_lft 3243sec
    inet6 fe80::1266:6aff:feca:9ef1/64 scope link
       valid_lft forever preferred_lft forever
```

```
root@c2:~# apt update -y
root@c2:~# apt upgrade -y
root@c2:~# apt install -y traceroute

root@c2:~# traceroute zabbly.com
traceroute to zabbly.com (45.45.148.7), 30 hops max, 60 byte packets
 1  _gateway (172.16.10.253)  4.353 ms  4.587 ms  4.812 ms
 2  _gateway.incus (10.50.1.253)  6.278 ms  6.290 ms  6.267 ms
 3  _gateway (192.168.1.1)  6.150 ms  5.777 ms  6.046 ms
 4  10.61.209.2 (10.61.209.2)  17.201 ms 10.61.209.3 (10.61.209.3)  17.086 ms 10.61.209.2
(10.61.209.2)  23.465 ms
..
[ removed the rest of the output ]
```

To reach zabbly.com (45.45.148.7) the network traffic goes to internal-net-1 first (172.16.10.253). The next router it reaches is the external-net-1 (10.50.1.253). Then finally from there it goes to 192.168.1.1 which is the ip address of the router on the physical network.

It is important to note that the network traffic did NOT go through incus network bridge (incusbr0), which was our intention when we designed the network topology in the previous task.

We can see from the web UI that currently we have 2 containers c1 and c2 running inside the project Handson-lab-1. container c1 is connected to the external-net-1 and c2 is connected to internal-net-1

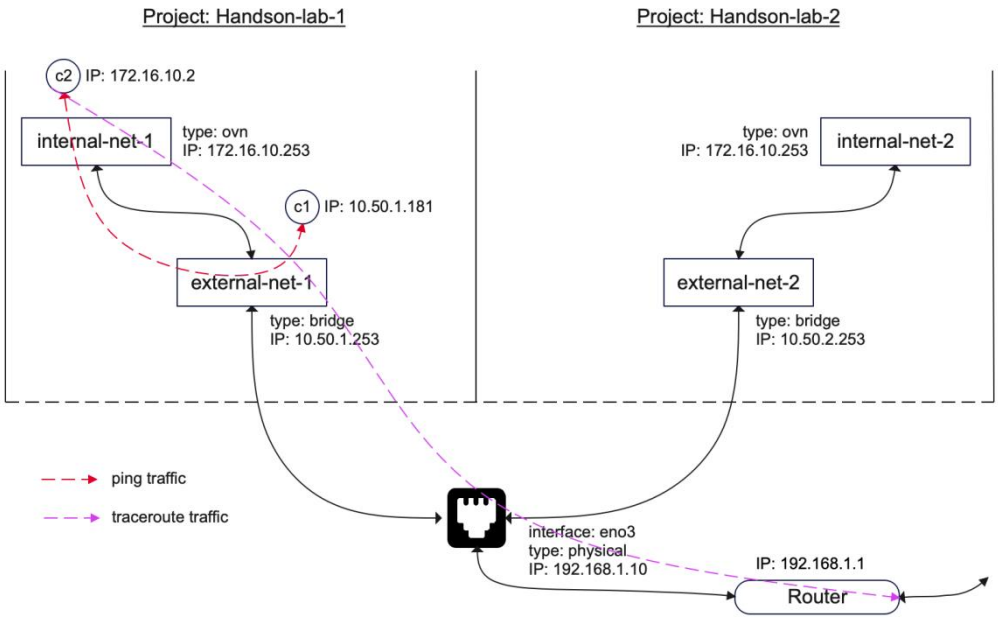


As we can see, containers c1 and c2 are on different networks. Now let us test if we can ping from c2 to c1.

```
root@c2:~# ping c1
PING c1.incus (10.50.1.181) 56(84) bytes of data.
64 bytes from c1.incus (10.50.1.181): icmp_seq=1 ttl=63 time=6.48 ms
64 bytes from c1.incus (10.50.1.181): icmp_seq=2 ttl=63 time=0.634 ms
64 bytes from c1.incus (10.50.1.181): icmp_seq=3 ttl=63 time=0.113 ms
^C
--- c1.incus ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.113/2.407/6.476/2.884 ms
root@c2:~#
```

This proves network traffic works fine as expected between the internal-net-1 and external-net-1.

And here is a diagrammatic representation of the network traffic flow of both the tests that we have conducted so far - ping and traceroute test.



The red colored arrow shows the network traffic flow of the ping test. And the purple colored arrow shows the network traffic flow of the traceroute test.

Task #24: Creating of a profile for a desktop virtual machine

In this task, let us create or profile for a desktop virtual machine defining its configurations. In the next task we will look at how to use this profile. Run the following command to create a new profile named 'desktop'.

```
mac% incus project switch Handson-lab-1
mac% incus profile create desktop
mac% incus profile edit desktop
config:
  limits.memory: 8GiB
  limits.cpu: 2
  security.nesting: "true"
description: desktop profile or project Handson-lab-1
devices:
  eth0:
    name: eth0
    network: external-net-1
    type: nic
  eth1:
    name: eth1
    network: internal-net-1
    type: nic
name: desktop
used_by: []
project: Handson-lab-1
```

The profile is self-explanatory. It defines the memory (8GiB), CPU count=2, suports virtualization nesting, defines 2 network interfaces. The first network interface (eth0) will be connected to the external network (external-net-1) and the second network interface (eth1) will be connected to internal network (internal-net-1).

Note that we did not define anything with regards to the storage. As you may recall we have already created a profile for storage before (aptly named 'storage') which defines which storage pool needs to be used. Here it is for a quick reference:

```
mac$ incus profile show storage
config: {}
description: storage profile for project Handson-lab-1
devices:
  root:
    path: /
    pool: storage-1
```

```
   type: disk
name: storage
used_by:
- /1.0/instances/c1?project=Handson-lab-1
- /1.0/instances/c2?project=Handson-lab-1
project: Handson-lab-1
```

Also note that 'storage' profile only defines the pool (storage-1) where the root disk will be stored. But it does not define the size of the root disk the virtual machine that will eventually be allocated to the virtual machine. We will see in the next task as where and how the size of the root disk will be defined.

Task #25: Creation of desktop virtual machine

In this task, let us get into the action of launching an ubuntu desktop leveraging the profiles that we have created before - namely 'storage' and 'desktop' profiles.

Let us launch an ubuntu desktop passing the profiles as parameters to the incus launch command line..

```
mac% incus project switch Handson-lab-1
```

```
mac% incus launch images:ubuntu/plucky/desktop ubuntu-desktop-vm --profile=storage
--profile=desktop -d root,size=20GiB --vm
```

```
mac% incus list
+-------------------+---------+----------------------+------+-----------------+-----------+
|      NAME         |  STATE  |         IPV4         | IPV6 |      TYPE       | SNAPSHOTS |
+-------------------+---------+----------------------+------+-----------------+-----------+
| c1                | RUNNING | 10.50.1.181 (eth0)   |      | CONTAINER       | 0         |
+-------------------+---------+----------------------+------+-----------------+-----------+
| c2                | RUNNING | 172.16.10.2 (eth0)   |      | CONTAINER       | 0         |
+-------------------+---------+----------------------+------+-----------------+-----------+
| ubuntu-desktop-vm | RUNNING | 172.16.10.3 (enp6s0) |      | VIRTUAL-MACHINE | 0         |
|                   |         | 10.50.1.110 (enp5s0) |      |                 |           |
+-------------------+---------+----------------------+------+-----------------+-----------+
```

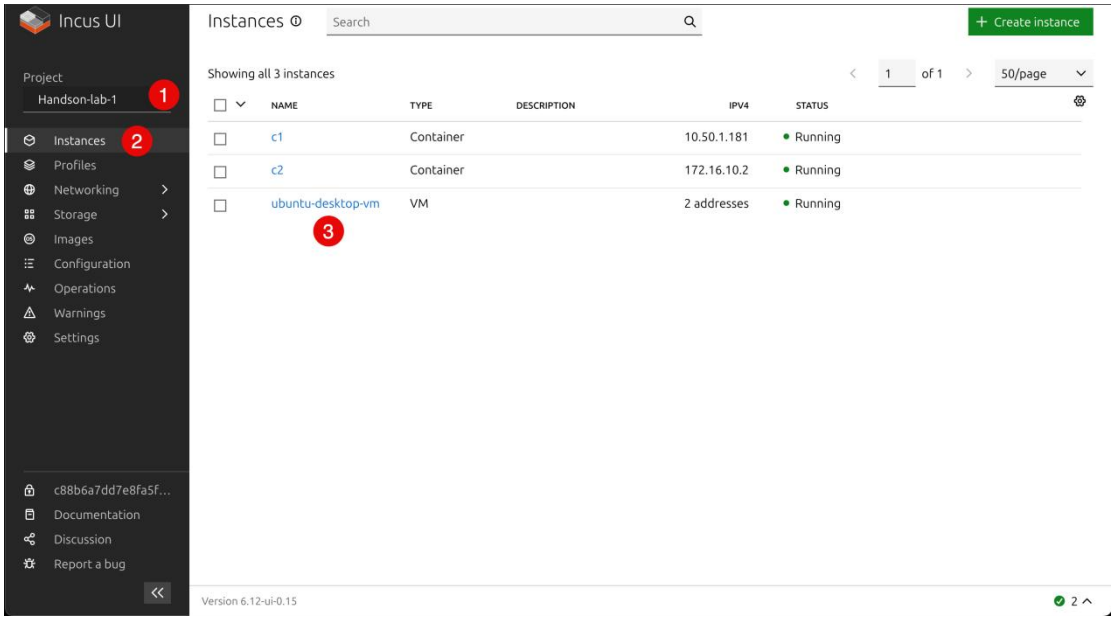
The first thing to notice are the ip addresses of the two network interfaces and the networks to which they are connected to.

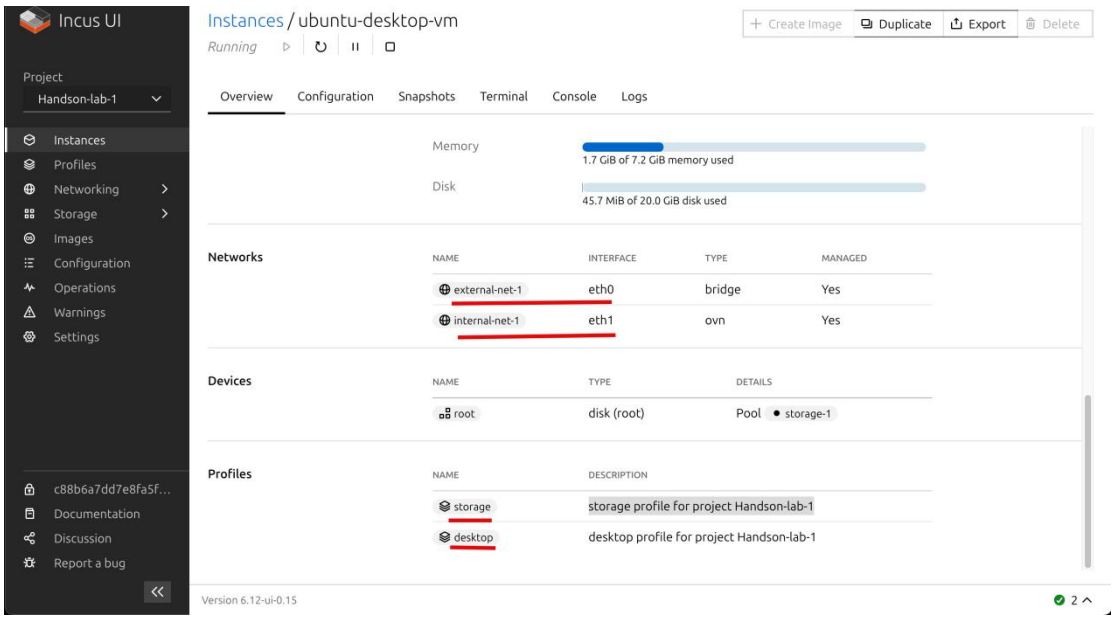
- enp5s0 is the first network interface connected to the external-net-1
- enp6s0 is the second network interface connected to the internal-net-1

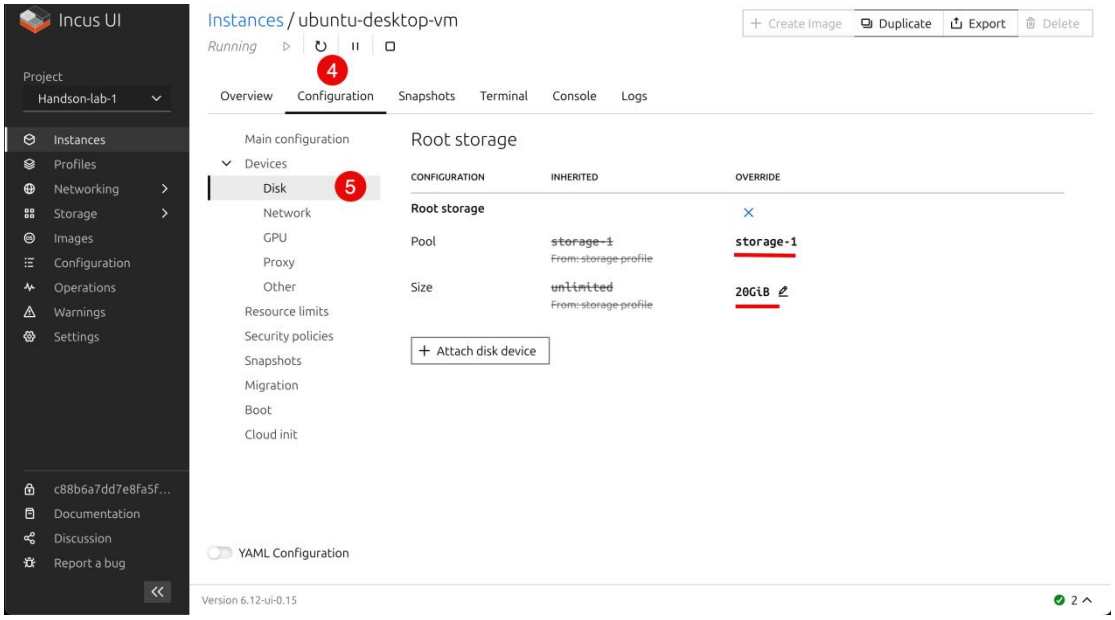
The reason being , the profile 'desktop' which is passed as the parameter, defines to which networks the two network interfaces should be connected to.

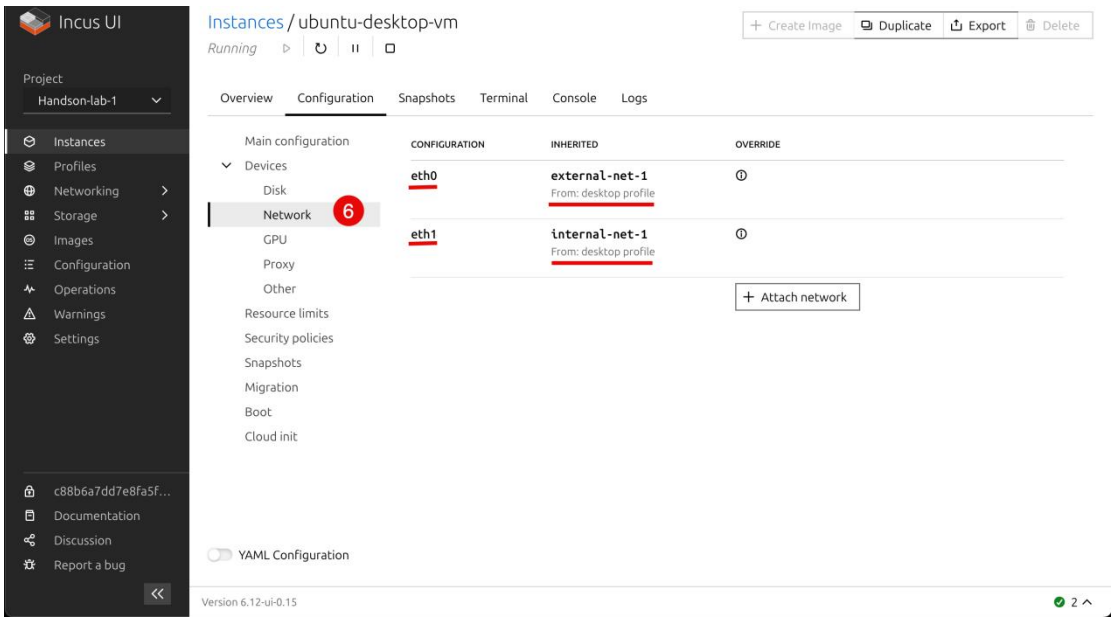
Similarly, the profile 'storage' defines to which storage pool (storage-1) the root disk of the desktop virtual machine should be stored.

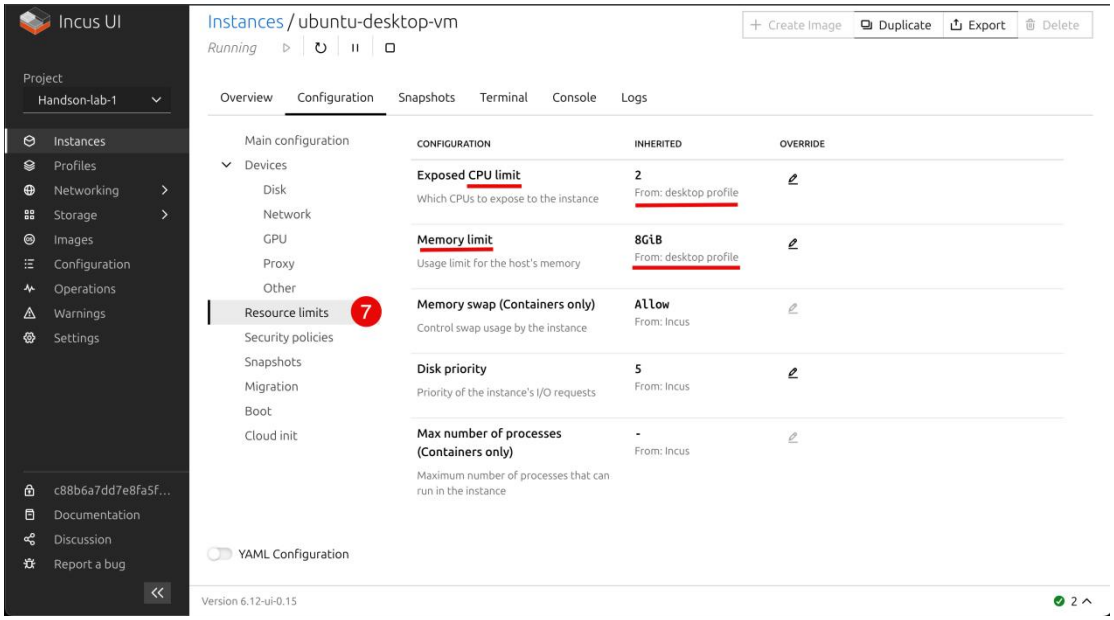
The same configuration can be seen using web UI

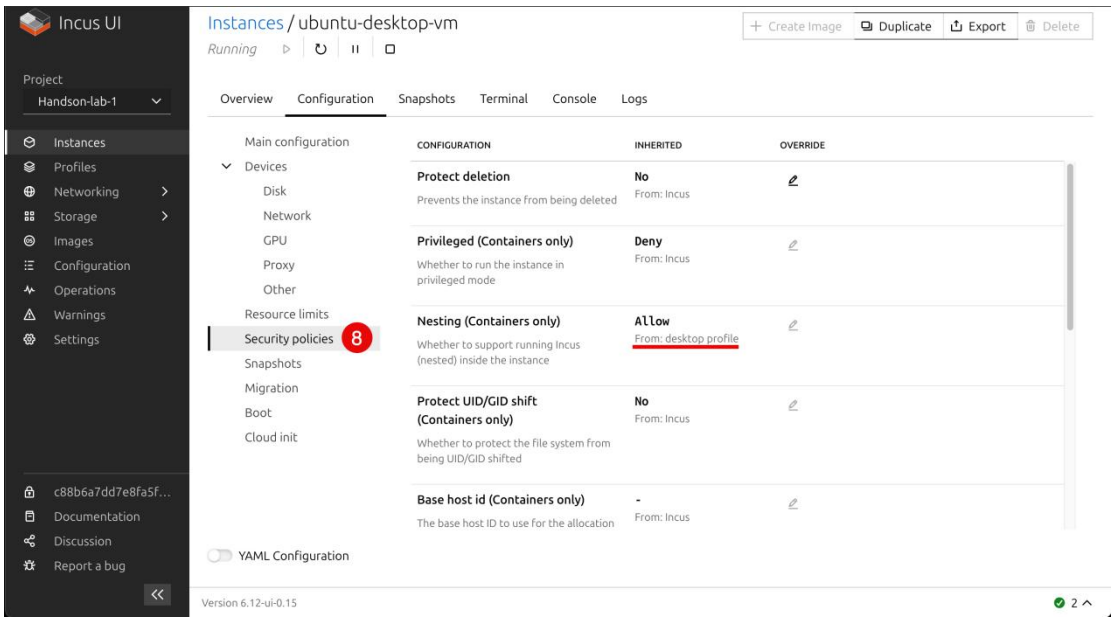

Incus UI
Project
Handson-lab-1
1
Instances
2
Profiles
Networking
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Instances
Search
Create instance
Showing all 3 instances
1 of 1
50/page
NAME
TYPE
DESCRIPTION
IPV4
STATUS
c1
Container
10.50.1.181
Running
c2
Container
172.16.10.2
Running
ubuntu-desktop-vm
VM
2 addresses
Running
3
Version 6.12-ui-0.15
2


Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Instances / ubuntu-desktop-vm
Running
Create Image
Duplicate
Export
Delete
Overview
Configuration
Snapshots
Terminal
Console
Logs
Memory
1.7 GiB of 7.2 GiB memory used
Disk
45.7 MiB of 20.0 GiB disk used
Networks
NAME
INTERFACE
TYPE
MANAGED
external-net-1
eth0
bridge
Yes
internal-net-1
eth1
ovn
Yes
Devices
NAME
TYPE
DETAILS
root
disk (root)
Pool
storage-1
Profiles
NAME
DESCRIPTION
storage
storage profile for project Handson-lab-1
desktop
desktop profile for project Handson-lab-1
Version 6.12-ui-0.15
2


Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Instances / ubuntu-desktop-vm
Running
4
Create Image
Duplicate
Export
Delete
Overview
Configuration
Snapshots
Terminal
Console
Logs
Main configuration
Devices
Disk
5
Network
GPU
Proxy
Other
Resource limits
Security policies
Snapshots
Migration
Boot
Cloud init
Root storage
CONFIGURATION
INHERITED
OVERRIDE
Root storage
Pool
storage-1
From: storage profile
storage-1
Size
unlimited
From: storage profile
20GiB
Attach disk device
YAML Configuration
Version 6.12-ui-0.15
2


Incus UI
Project
Handson-lab-1
Instances
Profiles
Networking
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Instances / ubuntu-desktop-vm
Running
Create Image
Duplicate
Export
Delete
Overview
Configuration
Snapshots
Terminal
Console
Logs
Main configuration
Devices
Disk
Network
6
GPU
Proxy
Other
Resource limits
Security policies
Snapshots
Migration
Boot
Cloud init
CONFIGURATION
INHERITED
OVERRIDE
eth0
external-net-1
From: desktop profile
eth1
internal-net-1
From: desktop profile
Attach network
YAML Configuration
Version 6.12-ui-0.15
2



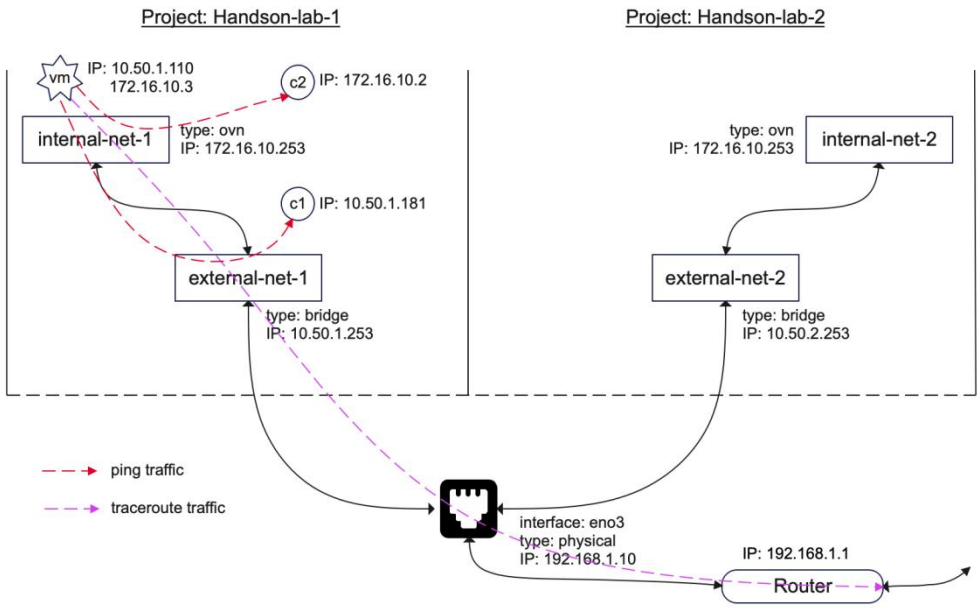


Let us run the same sanity tests that we have done before.

```
mac% incus shell ubuntu-desktop-vm
root@ubuntu-desktop-vm:~# apt update -y
root@ubuntu-desktop-vm:~# apt upgrade -y
root@ubuntu-desktop-vm:~# apt install traceroute -y

root@ubuntu-desktop-vm:~# ping c1
PING c1.incus (10.50.1.181) 56(84) bytes of data.
64 bytes from c1.incus (10.50.1.181): icmp_seq=1 ttl=64 time=0.403 ms
64 bytes from c1.incus (10.50.1.181): icmp_seq=2 ttl=64 time=0.314 ms
64 bytes from c1.incus (10.50.1.181): icmp_seq=3 ttl=64 time=0.319 ms
```

```
64 bytes from c1.incus (10.50.1.181): icmp_seq=4 ttl=64 time=0.315 ms
^C
--- c1.incus ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3062ms
rtt min/avg/max/mdev = 0.314/0.337/0.403/0.037 ms

root@ubuntu-desktop-vm:~# ping c2
PING c2.incus (172.16.10.2) 56(84) bytes of data.
64 bytes from 172.16.10.2: icmp_seq=1 ttl=64 time=1.95 ms
64 bytes from 172.16.10.2: icmp_seq=2 ttl=64 time=0.247 ms
64 bytes from 172.16.10.2: icmp_seq=3 ttl=64 time=0.227 ms
64 bytes from 172.16.10.2: icmp_seq=4 ttl=64 time=0.281 ms
^C
--- c2.incus ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3037ms
rtt min/avg/max/mdev = 0.227/0.676/1.952/0.736 ms

root@ubuntu-desktop-vm:~# traceroute zabbly.com
traceroute to zabbly.com (45.45.148.7), 30 hops max, 60 byte packets
 1  _gateway.incus (10.50.1.253)  1.139 ms  1.038 ms  1.008 ms
 2  _gateway (192.168.1.1)  0.973 ms  32.912 ms  32.905 ms
 3  10.61.209.2 (10.61.209.2)  12.192 ms  21.238 ms  21.104 ms
 ...
[ removed the rest of the output ]
```

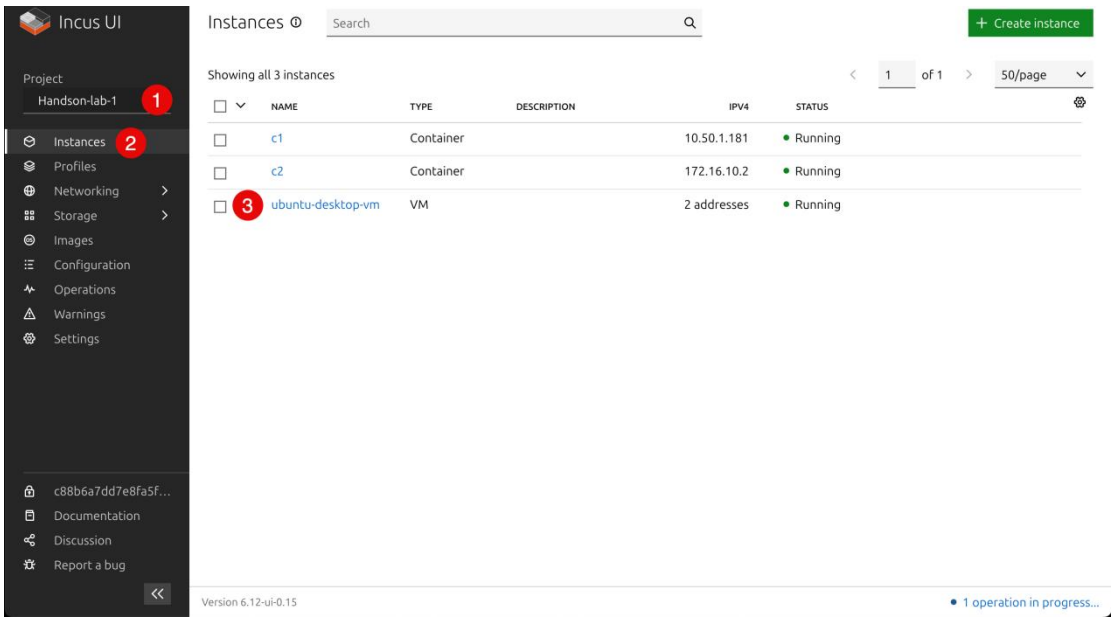
Here is diagramatic respresentation of the ping tests

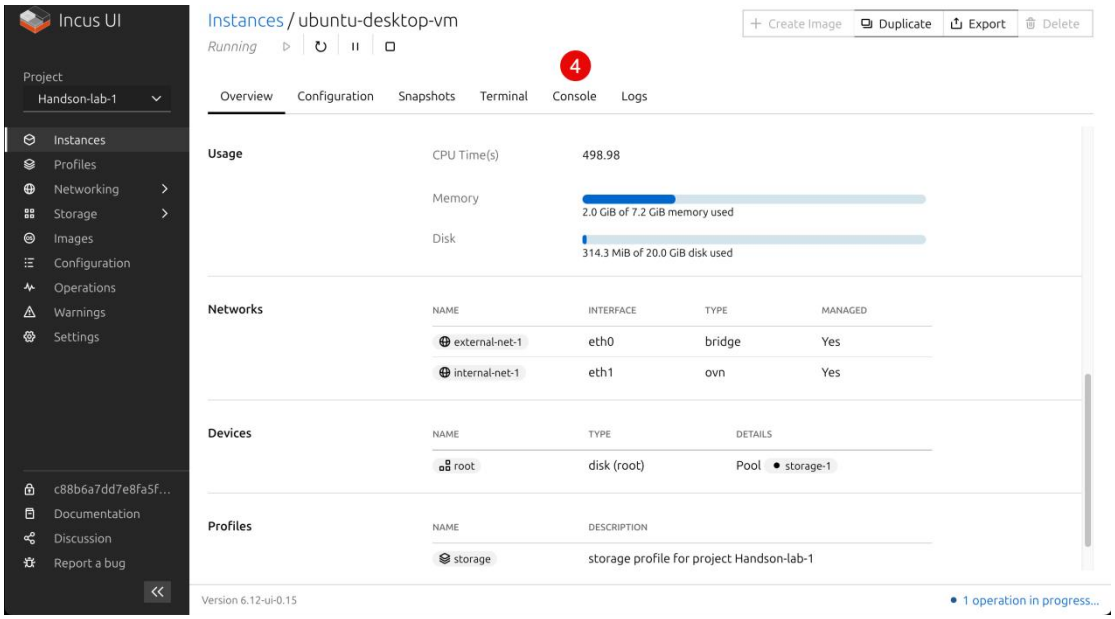


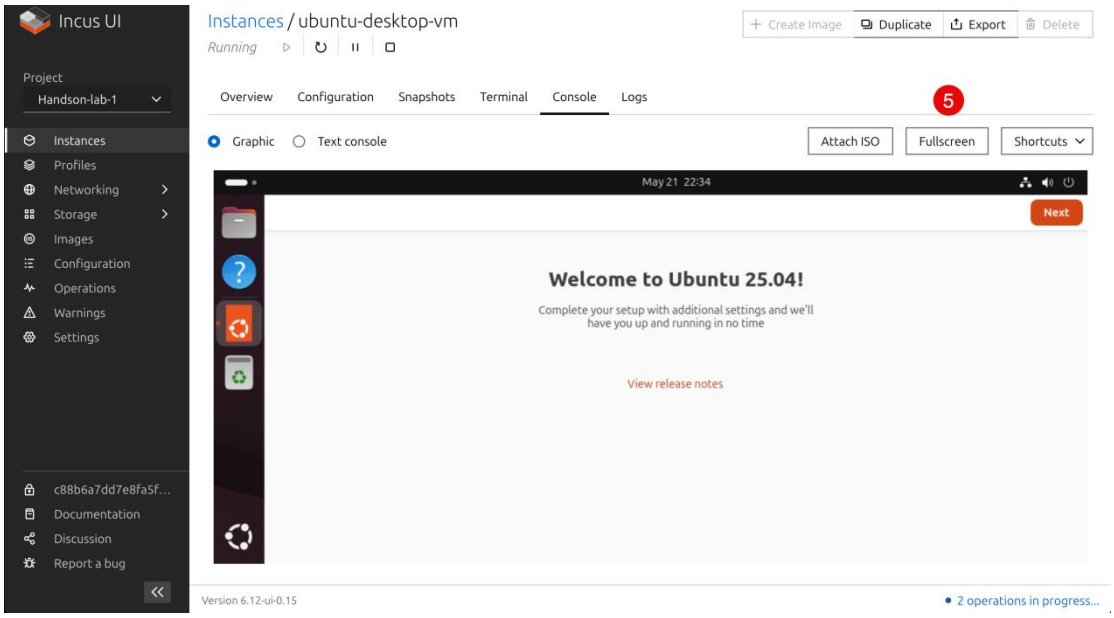
There are two ways we can access the console of this virtual machine

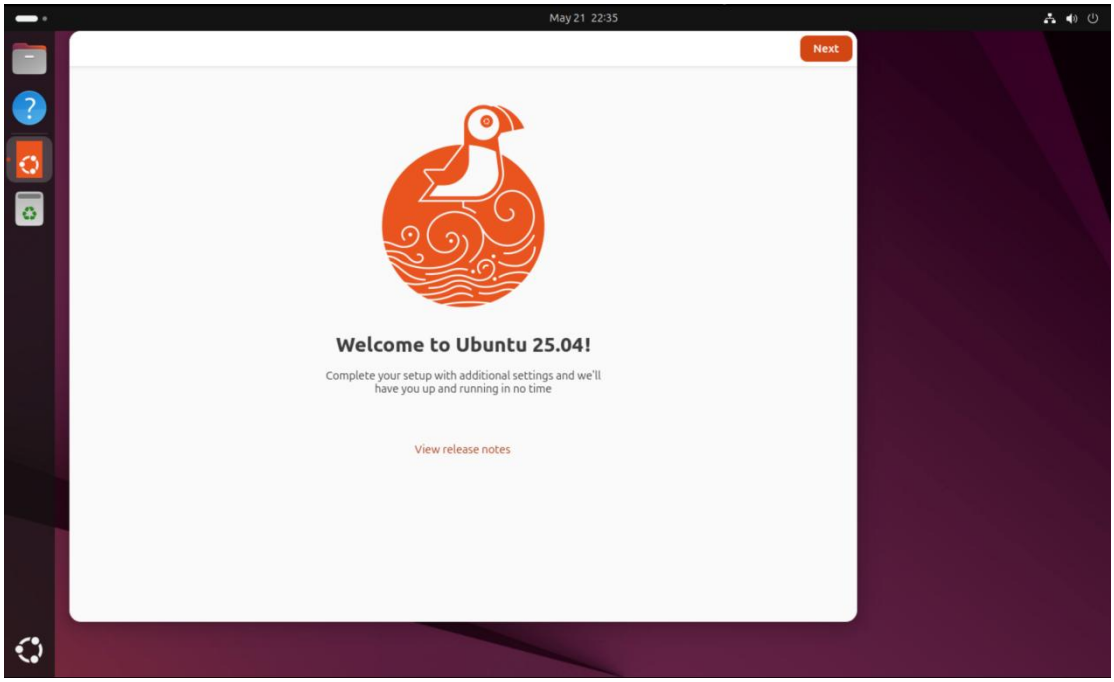
- Method #1: web UI
- Method #2: From Mac Laptop using command line

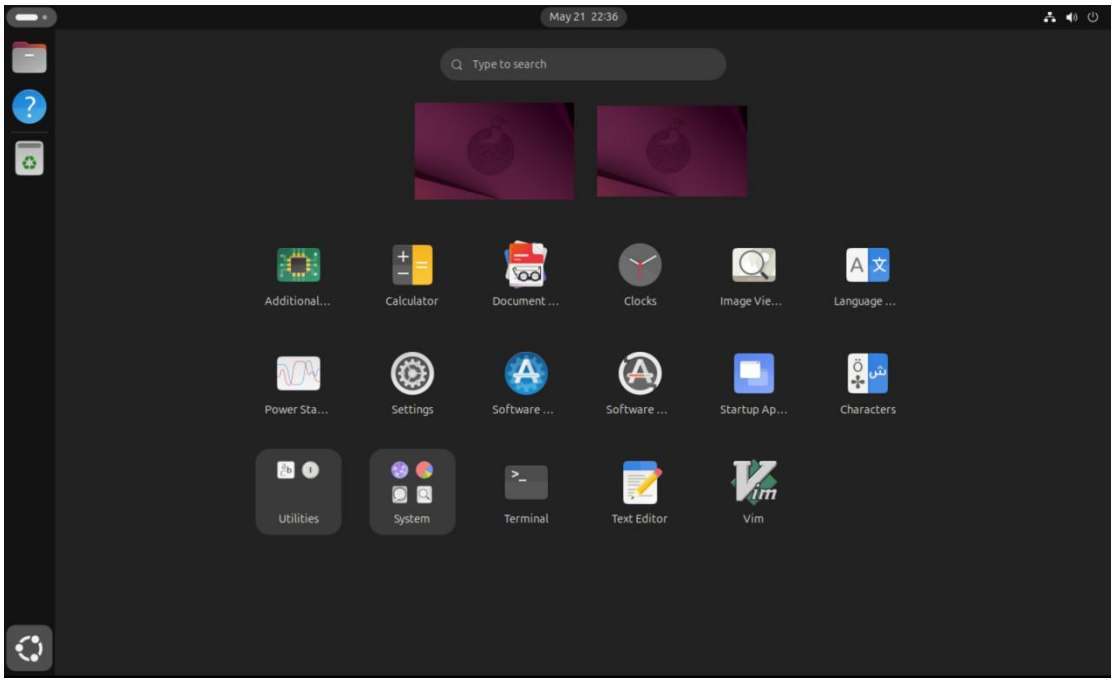
Method #1: web UI





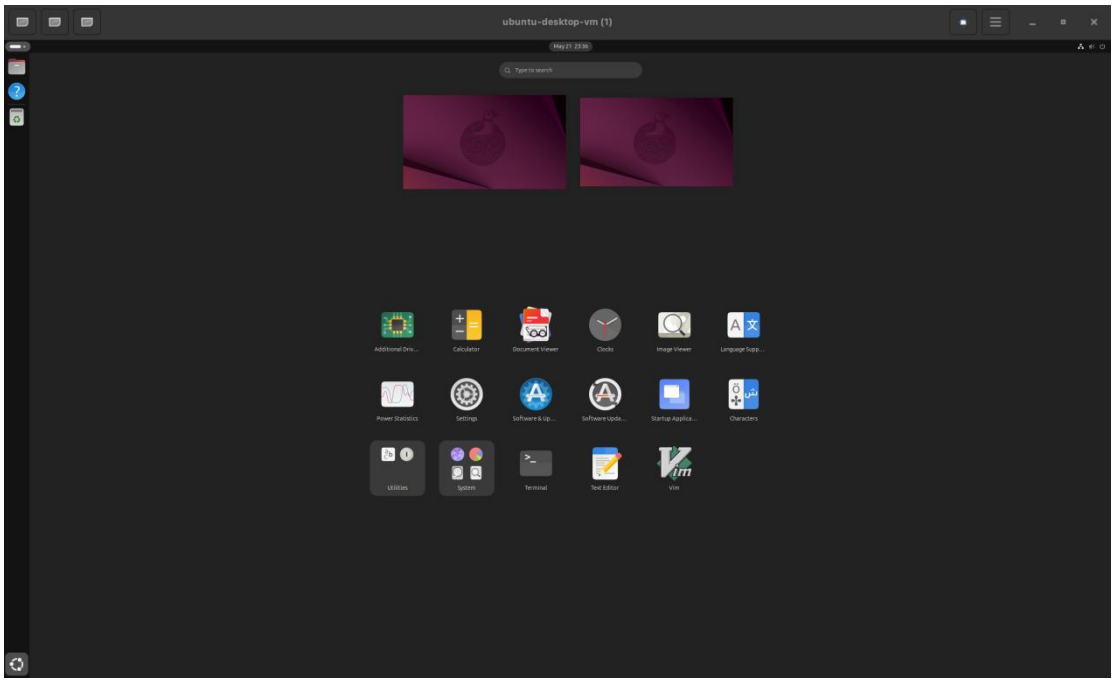

Incus UI
Instances / ubuntu-desktop-vm
Running
Create Image
Duplicate
Export
Delete
Project
Handson-lab-1
Instances
Profiles
Networking
Storage
Images
Configuration
Operations
Warnings
Settings
Overview
Configuration
Snapshots
Terminal
Console
Logs
5
Graphic
Text console
Attach ISO
Fullscreen
Shortcuts
May 21 22:34
Next
Welcome to Ubuntu 25.04!
Complete your setup with additional settings and we'll have you up and running in no time
View release notes
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Version 6.12-ui-0.15
2 operations in progress...


May 21 22:35
Next
Welcome to Ubuntu 25.04!
Complete your setup with additional settings and we'll have you up and running in no time
View release notes



Method #2: From Mac Laptop using command line

```
mac% incus project switch Handson-lab-1
mac% incus console ubuntu-desktop-vm --type=vga
```



Task #26: Creation of project: Handson-lab-2

Let us take a look at how project Handson-lab-1 has evolved so far with regards to all the networks, profile, storage pool and all the virtual machine and container instances that we have launched so far.

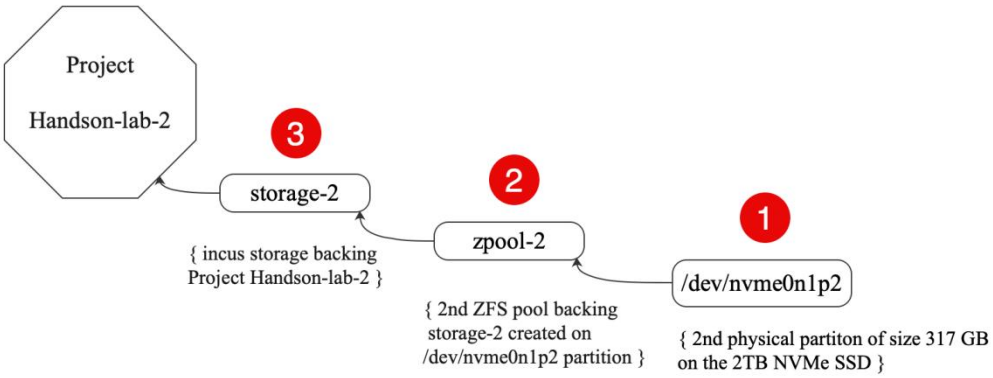
We are going to follow the same procedures to create a second project Handson-lab-2. Once we have it in place, we will understand how to segregate the networks among projects, which is a crucial concept to understand to attain multi-tenancy.

```
mac% incus project show Handson-lab-1
config:
  features.images: "true"
  features.networks: "false"
  features.networks.zones: "false"
  features.profiles: "true"
  features.storage.buckets: "true"
  features.storage.volumes: "true"
  restricted: "true"
  restricted.containers.lowlevel: allow
  restricted.containers.nesting: allow
  restricted.containers.privilege: allow
  restricted.devices.disk: managed
  restricted.devices.gpu: allow
  restricted.snapshots: allow
  restricted.virtual-machines.lowlevel: allow
description: Project Handson-lab-1
name: Handson-lab-1
used_by:
- /1.0/instances/c1?project=Handson-lab-1
- /1.0/instances/c2?project=Handson-lab-1
- /1.0/instances/ubuntu-desktop-vm?project=Handson-lab-1
-
/1.0/images/6b4ffc77179ffc19b47d4fe9044d3e6b81408ddb8ec849fdc2c2b256c5a55a0d?project=Ha
ndson-lab-1
-
/1.0/images/c9679e4c68822195ef69182e8b97261ef534a18f48e72d93ba15a0ffb3a0f835?project=Ha
ndson-lab-1
- /1.0/profiles/default?project=Handson-lab-1
- /1.0/profiles/desktop?project=Handson-lab-1
- /1.0/profiles/external-net?project=Handson-lab-1
- /1.0/profiles/internal-net?project=Handson-lab-1
- /1.0/profiles/storage?project=Handson-lab-1
```

To create a new project named Handson-lab-2, we need to repeat from task #17 to task #23. (I am skipping to showcase the same document contents once again as it will be redundent).

Here is the overall design for project Handson-lab-2, which is almost similar to project Handson-lab-1

Storage layout for Project Handson-lab-2



Here are the details after creation of project Handson-lab-2

```
mac% incus project switch Handson-lab-2
```

Project Handson-lab-2

```
mac% incus project show Handson-lab-2
config:
  features.images: "true"
  features.networks: "false"
  features.networks.zones: "false"
  features.profiles: "true"
  features.storage.buckets: "true"
  features.storage.volumes: "true"
  restricted: "true"
  restricted.containers.lowlevel: allow
  restricted.containers.nesting: allow
  restricted.containers.privilege: allow
  restricted.devices.disk: managed
  restricted.devices.gpu: allow
  restricted.snapshots: allow
```

```
  restricted.virtual-machines.lowlevel: allow
description: Project Handson-lab-2
name: Handson-lab-2
used_by:
- /1.0/profiles/default?project=Handson-lab-2
```

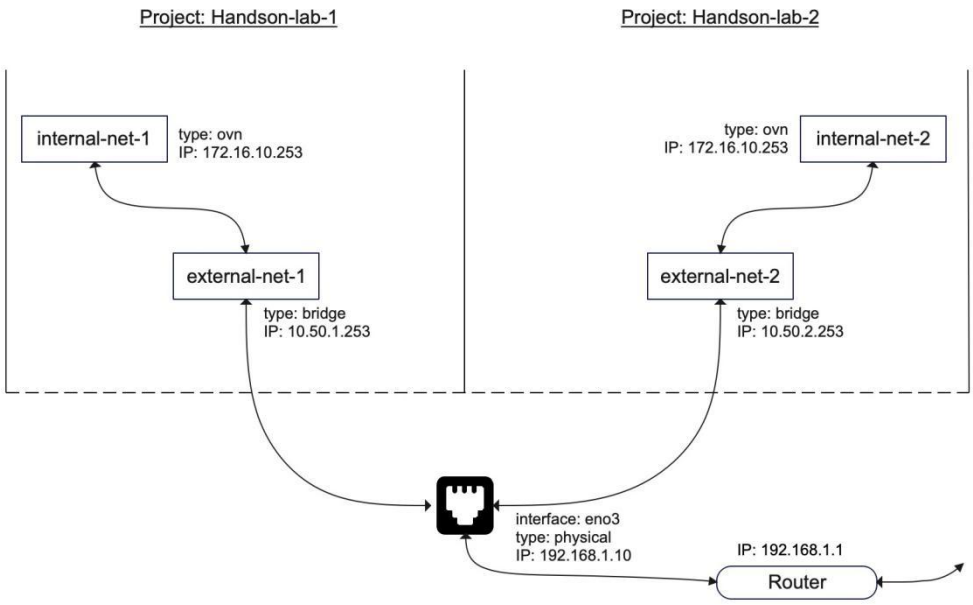
default profile

```
mac% incus profile show default
config:
  security.nesting: "true"
  security.privileged: "true"
description: Default Incus profile for project Handson-lab-2
devices:
  root:
    path: /
    pool: storage-2
    type: disk
name: default
used_by: []
project: Handson-lab-2
```

Storage profile

```
mac% incus storage show storage
config: {}
description: storage profile for project Handson-lab-2
devices:
  root:
    path: /
    pool: storage-2
    type: disk
name: storage
used_by: []
project: Handson-lab-2
```

Network Architecture for project Handson-lab-2



External network - external-net-2

```
mac% incus network show external-net-2
config:
  ipv4.address: 10.50.2.253/24
  ipv4.dhcp.ranges: 10.50.2.101-10.50.2.200
  ipv4.nat: "true"
  ipv6.address: none
description: ""
name: external-net-2
type: bridge
used_by: []
managed: true
status: Created
locations:
- none
project: default
```

Profile for external-net-2

```
mac% incus profile show external-net
config: {}
description: external-net profile for project Handson-lab-2
devices:
```

```
  eth0:
    name: eth0
    network: external-net-2
    type: nic
name: external-net
used_by: []
project: Handson-lab-2
```

Internal network - internal-net-2

```
mac% incus network show internal-net-2
config:
  bridge.mtu: "1500"
  ipv4.address: 172.16.10.253/24
  ipv4.nat: "true"
  ipv6.address: none
  network: external-net-2
  volatile.network.ipv4.address: 10.50.2.1
description: routes via external-net-2
name: internal-net-2
type: ovn
used_by:
- /1.0/profiles/internal-net?project=Handson-lab-2
managed: true
status: Created
locations:
- none
project: default
```

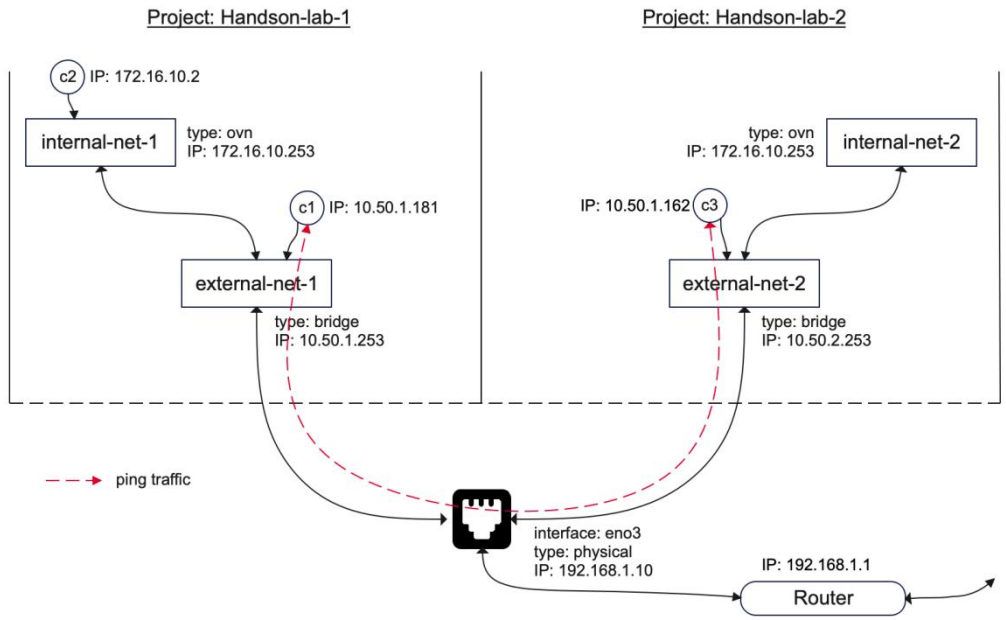
Profile for internal-net-2

```
mac% incus profile show internal-net
config: {}
description: internal network profile for project Handson-lab-2
devices:
  eth0:
    name: eth0
    network: internal-net-2
    type: nic
name: internal-net
used_by: []
project: Handson-lab-2
```

```
mac% incus profile list
+--------------+-----------------------------------------------------+---------+
|     NAME     |                     DESCRIPTION                     | USED BY |
+--------------+-----------------------------------------------------+---------+
| default      | Default Incus profile for project Handson-lab-2     | 0       |
+--------------+-----------------------------------------------------+---------+
| external-net | external-net profile for project Handson-lab-2      | 0       |
+--------------+-----------------------------------------------------+---------+
| internal-net | internal network profile for project Handson-lab-2  | 0       |
+--------------+-----------------------------------------------------+---------+
| storage      | storage profile for project Handson-lab-2           | 0       |
+--------------+-----------------------------------------------------+---------+

mac% incus network list
+---------------+--------+---------+-----------------+------+--------------------------+---------+---------+
|     NAME      |  TYPE  | MANAGED |      IPV4       | IPV6 |       DESCRIPTION        | USED BY |  STATE  |
+---------------+--------+---------+-----------------+------+--------------------------+---------+---------+
| external-net-1 | bridge | YES     | 10.50.1.253/24  | none | uplink for Handson-lab-1  | 5       | CREATED |
+---------------+--------+---------+-----------------+------+--------------------------+---------+---------+
| external-net-2 | bridge | YES     | 10.50.2.253/24  | none | uplink for Handson-lab-2  | 2       | CREATED |
+---------------+--------+---------+-----------------+------+--------------------------+---------+---------+
| incusbr0      | bridge | YES     | 10.1.1.253/24   | none | routes traffic via eno3   | 1       | CREATED |
+---------------+--------+---------+-----------------+------+--------------------------+---------+---------+
| internal-net-1 | ovn    | YES     | 172.16.10.253/24 | none | routes via external-net-1 | 4       | CREATED |
+---------------+--------+---------+-----------------+------+--------------------------+---------+---------+
| internal-net-2 | ovn    | YES     | 172.16.10.253/24 | none | routes via external-net-2 | 1       | CREATED |
+---------------+--------+---------+-----------------+------+--------------------------+---------+---------+
```

Task #27: Network test between projects - Take #1

Now that we have two projects in place namely Handson-lab-1 and Handson-lab-2, let us try to do a simple ping test across projects. We already have a few instances running in project Handson-lab-1. Let us launch a container instance in project Handson-lab-2 and see if we can ping a container instance in project Handson-lab-1.



Let us launch an ubuntu container (c3) connecting it to external-net-2

```
mac% incus launch images:ubuntu/24.04 c3 --profile=external-net --profile=storage -d
root,size=5GiB

% incus list
+------+---------+--------------------+------+-----------+-----------+
| NAME |  STATE  |       IPV4         | IPV6 |   TYPE    | SNAPSHOTS |
+------+---------+--------------------+------+-----------+-----------+
| c3   | RUNNING | 10.50.2.162 (eth0) |      | CONTAINER | 0         |
+------+---------+--------------------+------+-----------+-----------+
```

Let us get inside the contiainer c3 and see if we can reach a host on the internet by pinging zabbly.com

```
mac% incus shell c3
root@c3:~# ping zabbly.com
PING zabbly.com (45.45.148.7) 56(84) bytes of data.
64 bytes from rproxy.dcmtl.stgraber.org (45.45.148.7): icmp_seq=1 ttl=49 time=84.3 ms
```

```
64 bytes from rproxy.dcmtl.stgraber.org (45.45.148.7): icmp_seq=2 ttl=49 time=85.1 ms
64 bytes from rproxy.dcmtl.stgraber.org (45.45.148.7): icmp_seq=3 ttl=49 time=81.6 ms
^C
--- zabbly.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 81.600/83.682/85.147/1.512 ms
root@c3:~#
```

We will ping from the container (c3, IP: 10.50.2.162) running inside project Handson-lab-2 to the container (c1, IP: 10.50.1.181) running inside project Handson-lab-1

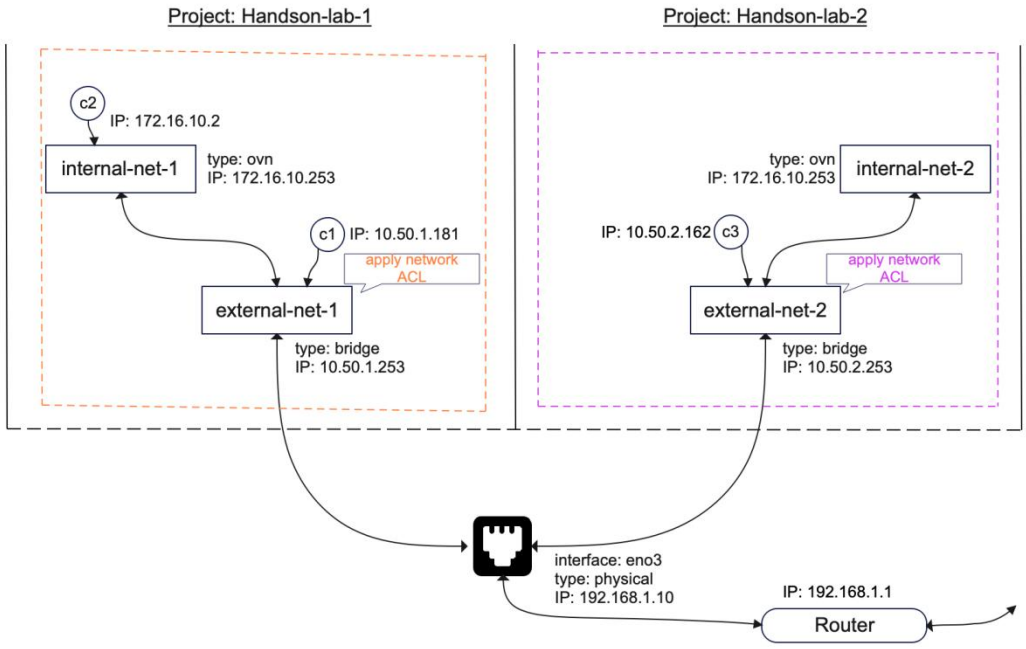
```
root@c3:~# ping 10.50.1.181
PING 10.50.1.181 (10.50.1.181) 56(84) bytes of data.
64 bytes from 10.50.1.181: icmp_seq=1 ttl=63 time=0.158 ms
64 bytes from 10.50.1.181: icmp_seq=2 ttl=63 time=0.100 ms
64 bytes from 10.50.1.181: icmp_seq=3 ttl=63 time=0.097 ms
^C
--- 10.50.1.181 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2045ms
rtt min/avg/max/mdev = 0.097/0.118/0.158/0.028 ms
root@c3:~#
```

Well, what it means is that there is NO network isolation among the Projects in our setup, which is a crucial aspect for multi-tenancy. So let us working on building network isolation in our next task.

Task #28: Setting up network isolation among projects

As we have seen before network traffic is not isolated between our projects, which is needed for a multi-tenancy setup. To accomplish this, we need to write network ACL (access control lists) rules and apply them onto appropriate networks based on the network architecture design. In our case, we need to apply ACL rules to the external networks of both the projects.

The color coding shown below definites the perimeter/boundary of network isolation per project defined by separate netork ACL per project.



ACL rule for external-net-2 of project Handson-lab-2

Let us create the ACL rule

```
mac% incus project switch Handson-lab-2

mac% incus network acl list
+------+-------------+---------+
| NAME | DESCRIPTION | USED BY |
+------+-------------+---------+

mac% incus list
+------+---------+--------------------+------+-----------+-----------+
| NAME |  STATE  |        IPV4        | IPV6 |   TYPE    | SNAPSHOTS |
+------+---------+--------------------+------+-----------+-----------+
```

```
| c3   | RUNNING  | 10.50.2.162 (eth0) |      | CONTAINER |   0       |
+------+---------+--------------------+------+-----------+-----------+

mac% incus network acl create isolate-lab-2
mac% incus network acl edit isolate-lab-2
name: isolate-lab-2
description: allow external-net-2 only. reject other external-nets
egress:
- action: allow
  state: enabled
- action: reject
  destination: 10.50.1.0/24
  state: enabled
- action: reject
  destination: 10.50.1.253/24
  state: enabled
ingress:
- action: allow
  state: enabled
- action: reject
  source: 10.50.1.0/24
  state: enabled
config: {}
used_by: []
project: default
```

We are defining the ingress and egress rules. For this project Handson-lab-2, we are mentioning that the external ip address range of project Handson-lab-1 to be excplicitly rejected (10.50.1.0/24) and the rest are accepted. And we will do the reverse for project Handson-lab-1.

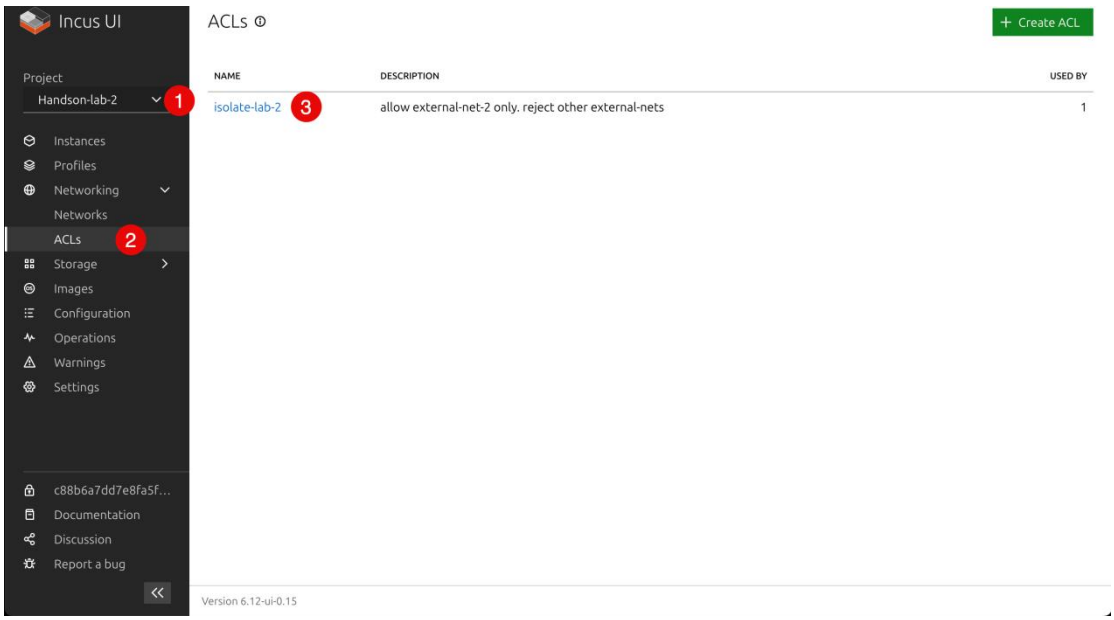
In this way, the external-net-2 that is assigned to project Handson-lab-2 will reject any traffic from external-net-1 assigned to Handson-lab-1.

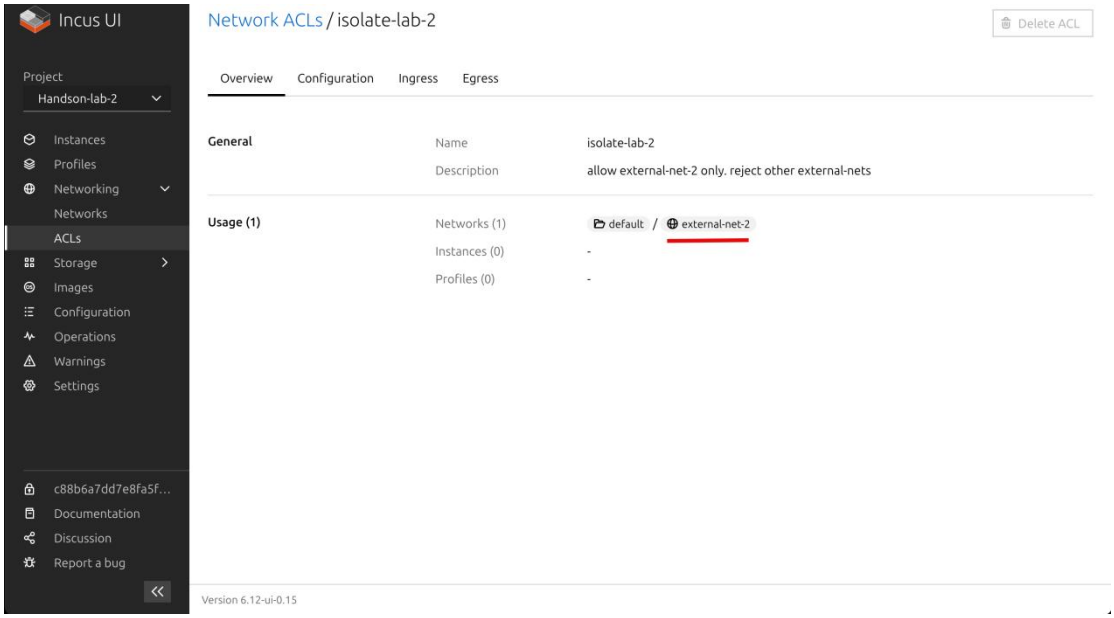
So far we have only created the ACL rule. The next step is to apply this rule to the external-net-2.

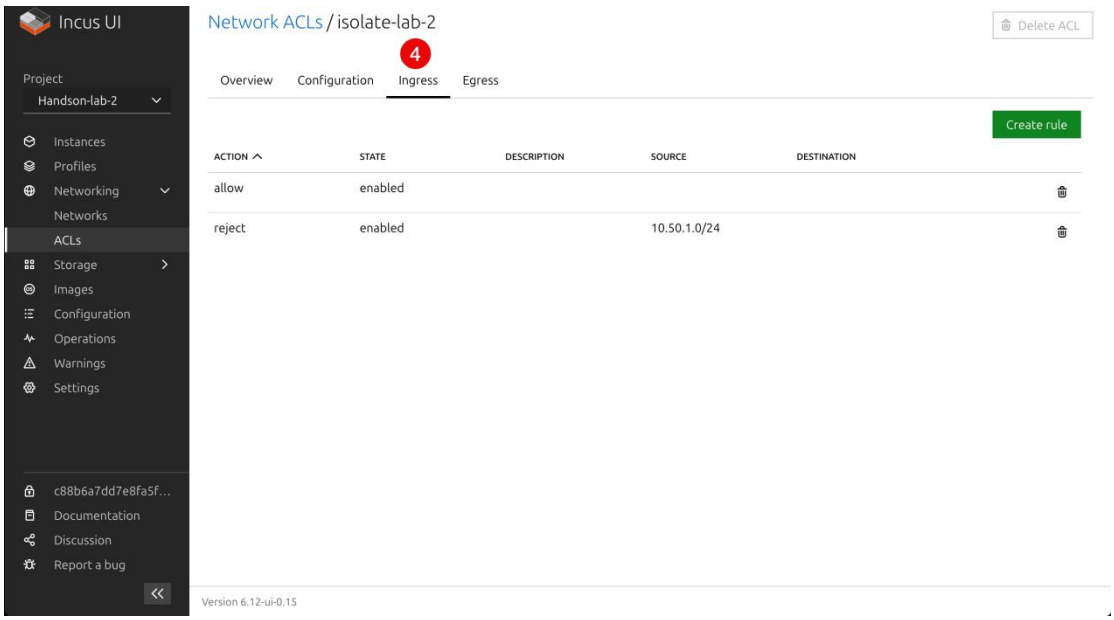
```
mac% incus network edit external-net-2
config:
  ipv4.address: 10.50.2.253/24
  ipv4.dhcp.ranges: 10.50.2.101-10.50.2.200
  ipv4.nat: "true"
  ipv4.ovn.ranges: 10.50.2.1-10.50.2.25
  ipv4.routing: "true"
  ipv6.address: none
```

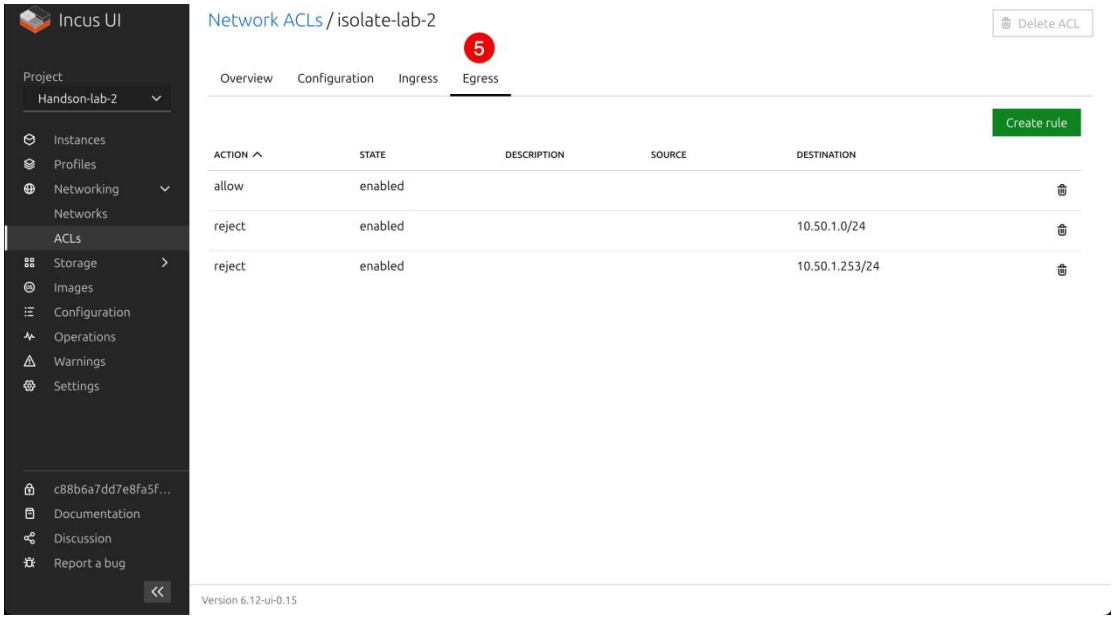
```
  security.acls: isolate-lab-2 --> Added this line
description: uplink for Handson-lab-2
name: external-net-2
type: bridge
used_by:
- /1.0/instances/c3?project=Handson-lab-2
- /1.0/networks/internal-net-2
- /1.0/profiles/external-net?project=Handson-lab-2
managed: true
status: Created
locations:
- none
project: default
```

Let us take a look at how the newly created ACL rule shows up in web UI.




Incus UI
Project
Handson-lab-2
Instances
Profiles
Networking
Networks
ACLs
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Network ACLs / isolate-lab-2
Delete ACL
Overview
Configuration
Ingress
Egress
General
Name
isolate-lab-2
Description
allow external-net-2 only. reject other external-nets
Usage (1)
Networks (1)
default / external-net-2
Instances (0)
-
Profiles (0)
-
Version 6.12-ui-0.15


Incus UI
Project
Handson-lab-2
Instances
Profiles
Networking
Networks
ACLs
Storage
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Network ACLs / isolate-lab-2
Delete ACL
4
Overview
Configuration
Ingress
Egress
Create rule
ACTION
STATE
DESCRIPTION
SOURCE
DESTINATION
allow
enabled
reject
enabled
10.50.1.0/24
Version 6.12-ui-0.15



Now that we have completed applying a network ACL rule for external-net-2 that is backing project Handson-lab-2, let us do the same for external-net-1 that is backing project Handson-lab-1.

ACL rule for external-net-1 of project Handson-lab-1

Let us create the ACL rule

```
mac% incus project switch Handson-lab-1

mac% incus network acl list
+---------------+-----------------------------------------------------------+---------+
|     NAME      |                    DESCRIPTION                            | USED BY |
+---------------+-----------------------------------------------------------+---------+
| isolate-lab-2 | allow external-net-2 only. reject other external-nets | 1       |
+---------------+-----------------------------------------------------------+---------+

mac% incus network acl edit isolate-lab-1
name: isolate-lab-1
description: allow external-net-1 only. reject other external-nets
egress:
- action: allow
  state: enabled
- action: reject
  destination: 10.50.2.0/24
  state: enabled
- action: reject
  destination: 10.50.2.253/24
```

```
  state: enabled
ingress:
- action: allow
  state: enabled
- action: reject
  source: 10.50.2.0/24
  state: enabled
config: {}
used_by: []
project: default
```

Let us apply this ACL onto exernal-net-1 which serves project Haandson-lab-1

```
mac% incus network edit external-net-1
config:
  ipv4.address: 10.50.1.253/24
  ipv4.dhcp.ranges: 10.50.1.101-10.50.1.200
  ipv4.nat: "true"
  ipv4.ovn.ranges: 10.50.1.1-10.50.1.25
  ipv4.routing: "true"
  ipv6.address: none
  security.acls: isolate-lab-1 --> Added this line
description: uplink for Handson-lab-1
name: external-net-1
type: bridge
used_by:
- /1.0/instances/c1?project=Handson-lab-1
- /1.0/instances/ubuntu-desktop-vm?project=Handson-lab-1
- /1.0/networks/internal-net-1
- /1.0/profiles/desktop?project=Handson-lab-1
- /1.0/profiles/external-net?project=Handson-lab-1
managed: true
status: Created
locations:
- none
project: default

mac% incus network acl list
+---------------+-----------------------------------------------------------+---------+
|    NAME       |                    DESCRIPTION                            | USED BY |
+---------------+-----------------------------------------------------------+---------+
| isolate-lab-1 | allow external-net-1 only. reject other external-nets | 0       |
+---------------+-----------------------------------------------------------+---------+
| isolate-lab-2 | allow external-net-2 only. reject other external-nets | 1       |
```

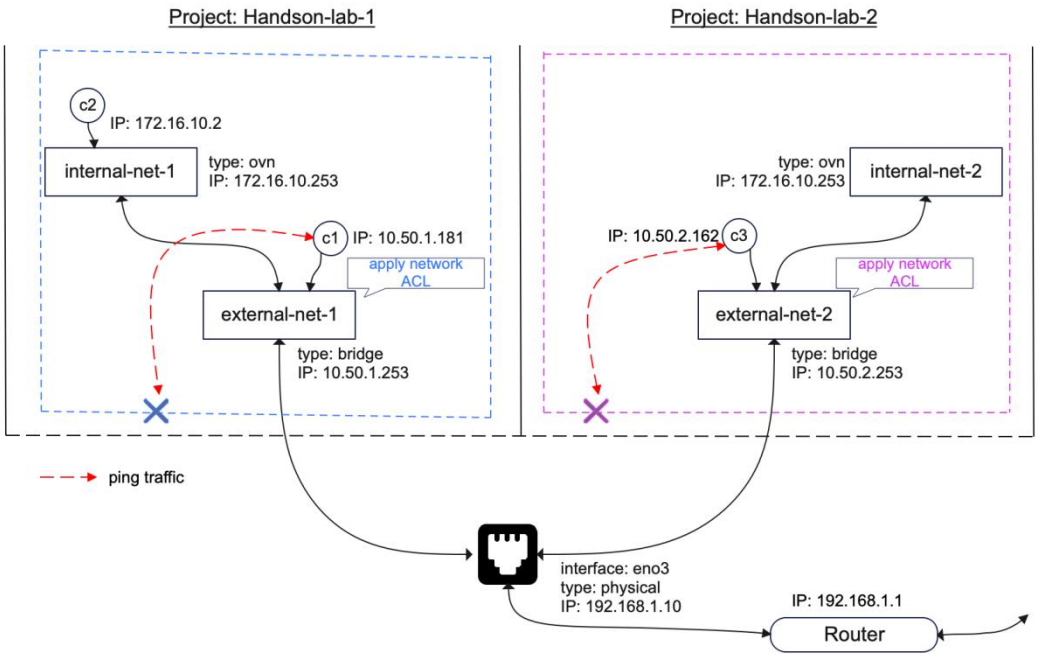
```
+---------------+-------------------------------------------------------------+---------+
```

Now that we have two network ACL rules in place, one (isolate-lab-1) is applied to external-net-1 and the other (isolate-lab-2) is to external-net-2.

Let us repeat the ping test again and see system's behavior.

Task #29: Network test between projects - Take #2

Let us see the outcome of the ping test once again, now that we have network ACLs in each project in place.



Let us access project Handson-lab-2 first.

```
mac% incus project switch Handson-lab-2

mac% incus list
+------+---------+--------------------+------+-----------+-----------+
| NAME |  STATE  |        IPV4        | IPV6 |   TYPE    | SNAPSHOTS |
+------+---------+--------------------+------+-----------+-----------+
| c3   | RUNNING | 10.50.2.162 (eth0) |      | CONTAINER | 0         |
+------+---------+--------------------+------+-----------+-----------+
```

Let us get into the container (c3) and try to ping container (c1) running inside the project Handson-lab-1.

```
mac% incus shell c3
root@c3:~# ping 10.50.1.181
PING 10.50.1.181 (10.50.1.181) 56(84) bytes of data.
From 10.50.2.253 icmp_seq=1 Destination Port Unreachable
From 10.50.2.253 icmp_seq=2 Destination Port Unreachable
```

```
From 10.50.2.253 icmp_seq=3 Destination Port Unreachable
^C
--- 10.50.1.181 ping statistics ---
3 packets transmitted, 0 received, +3 errors, 100% packet loss, time 2027ms
```

We can clearly see the ping fails because of the network ACL in place in the project Handson-lab-1. Let us see if we can ping a host on the internet.

```
root@c3:~# ping zabbly.com
PING zabbly.com (45.45.148.7) 56(84) bytes of data.
64 bytes from rproxy.dcmtl.stgraber.org (45.45.148.7): icmp_seq=1 ttl=49 time=93.0 ms
64 bytes from rproxy.dcmtl.stgraber.org (45.45.148.7): icmp_seq=2 ttl=49 time=93.7 ms
64 bytes from rproxy.dcmtl.stgraber.org (45.45.148.7): icmp_seq=3 ttl=49 time=94.3 ms
^C
--- zabbly.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 93.036/93.665/94.273/0.505 ms
root@c3:~#
```

This proves that the network isolation is only the context of reaching out any container or virtual machine inside Handson-lab-1.

Let us do the other way around as well. Let us access project Handson-lab-1

```
mac% incus project switch Handson-lab-1
```

```
mac% incus list
```

```
+------------------+---------+----------------------+------+-----------------+-----------+
|      NAME        |  STATE  |         IPV4         | IPV6 |       TYPE      | SNAPSHOTS |
+------------------+---------+----------------------+------+-----------------+-----------+
| c1               | RUNNING | 10.50.1.181 (eth0)   |      | CONTAINER       | 0         |
+------------------+---------+----------------------+------+-----------------+-----------+
| c2               | RUNNING | 172.16.10.2 (eth0)   |      | CONTAINER       | 0         |
+------------------+---------+----------------------+------+-----------------+-----------+
| ubuntu-desktop-vm | RUNNING | 172.16.10.3 (enp6s0) |      | VIRTUAL-MACHINE | 0         |
|                  |         | 10.50.1.110 (enp5s0) |      |                 |           |
+------------------+---------+----------------------+------+-----------------+-----------+
```

Let us get inside the container (c2) and try to ping container (c3) running inside project Handson-lab-2

```
% incus shell c2
root@c2:~# ping 10.50.2.162
```

```
PING 10.50.2.162 (10.50.2.162) 56(84) bytes of data.
From 10.50.1.253 icmp_seq=1 Destination Port Unreachable
From 10.50.1.253 icmp_seq=2 Destination Port Unreachable
From 10.50.1.253 icmp_seq=3 Destination Port Unreachable
^C
--- 10.50.2.162 ping statistics ---
3 packets transmitted, 0 received, +3 errors, 100% packet loss, time 2052ms
```

We see that the network traffic is blocked by the network ACL applied on external-net-2 (of project Handson-lab-2)

But if we try to ping another virtual machine running inside project Handson-lab-1 that is connected to the same external-net-1, the ping is successful.

```
root@c2:~# ping 10.50.1.110
PING 10.50.1.110 (10.50.1.110) 56(84) bytes of data.
64 bytes from 10.50.1.110: icmp_seq=1 ttl=63 time=7.22 ms
64 bytes from 10.50.1.110: icmp_seq=2 ttl=63 time=0.880 ms
64 bytes from 10.50.1.110: icmp_seq=3 ttl=63 time=0.311 ms
^C
--- 10.50.1.110 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2012ms
rtt min/avg/max/mdev = 0.311/2.803/7.218/3.130 ms
```

Finally let us make sure that we can ping a host on the internet, to prove that the network ACLs are applied only to deny each other's network traffic (from the individual projects point of view) and not to the outside world.

```
root@c2:~# ping zabbly.com
PING zabbly.com (45.45.148.7) 56(84) bytes of data.
64 bytes from rproxy.dcmtl.stgraber.org (45.45.148.7): icmp_seq=1 ttl=48 time=95.9 ms
64 bytes from rproxy.dcmtl.stgraber.org (45.45.148.7): icmp_seq=2 ttl=48 time=91.8 ms
64 bytes from rproxy.dcmtl.stgraber.org (45.45.148.7): icmp_seq=3 ttl=48 time=90.5 ms
^C
--- zabbly.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 90.502/92.744/95.911/2.302 ms
```

Task #30: A fully built multi-tenant system with 6 tenants

In this task we will look at a fully built multi-tenant system with 6 tenants (Handson-lab-1 to Handson-lab-6), with an internal and external network each. Each having it's own storage pool (storage-1 to storage-6).

The way we need to build projects Handson-lab-1 to Handson-lab-6 is exactly how we built Handson-lab-1 and Handson-lab-2. So there is no value in repeating the same documenation again. Instead, we will take a look at how all the 6 project looks like at the end. This output of this can be used as a reference.

```
mac% incus project show Handson-lab-1
config:
  features.images: "true"
  features.networks: "false"
  features.networks.zones: "false"
  features.profiles: "true"
  features.storage.buckets: "true"
  features.storage.volumes: "true"
  restricted: "true"
  restricted.containers.lowlevel: allow
  restricted.containers.nesting: allow
  restricted.containers.privilege: allow
  restricted.devices.disk: managed
  restricted.devices.gpu: allow
  restricted.snapshots: allow
  restricted.virtual-machines.lowlevel: allow
description: Project Handson-lab-1
name: Handson-lab-1
used_by:
- /1.0/instances/c1?project=Handson-lab-1
- /1.0/instances/c2?project=Handson-lab-1
- /1.0/instances/ubuntu-desktop-vm?project=Handson-lab-1
-
/1.0/images/1684815eba8b8f9d41293b3784bd4b480b432c56cfab9436cdfdc82ce8ded254?project=Ha
ndson-lab-1
-
/1.0/images/4ead11cd2bb106abfea928d8d5d84cdd05f127fa5b3e657acfe1ec9e737d9093?project=Ha
ndson-lab-1
-
/1.0/images/7485692e4b5f05a3b15cef15a9db5e238e269bc9f1ed7fea01a873f479029443?project=Ha
ndson-lab-1
- /1.0/profiles/default?project=Handson-lab-1
```

```
- /1.0/profiles/desktop?project=Handson-lab-1
- /1.0/profiles/external-net?project=Handson-lab-1
- /1.0/profiles/internal-net?project=Handson-lab-1
- /1.0/profiles/storage?project=Handson-lab-1

mac% incus project show Handson-lab-2
config:
  features.images: "true"
  features.networks: "false"
  features.networks.zones: "false"
  features.profiles: "true"
  features.storage.buckets: "true"
  features.storage.volumes: "true"
  restricted: "true"
  restricted.containers.lowlevel: allow
  restricted.containers.nesting: allow
  restricted.containers.privilege: allow
  restricted.devices.disk: managed
  restricted.devices.gpu: allow
  restricted.snapshots: allow
  restricted.virtual-machines.lowlevel: allow
description: Project Handson-lab-2
name: Handson-lab-2
used_by:
- /1.0/instances/c3?project=Handson-lab-2
-
/1.0/images/7485692e4b5f05a3b15cef15a9db5e238e269bc9f1ed7fea01a873f479029443?project=Ha
ndson-lab-2
- /1.0/profiles/default?project=Handson-lab-2
- /1.0/profiles/external-net?project=Handson-lab-2
- /1.0/profiles/internal-net?project=Handson-lab-2
- /1.0/profiles/storage?project=Handson-lab-2

mac% incus project show Handson-lab-3
config:
  features.images: "true"
  features.networks: "false"
  features.networks.zones: "false"
  features.profiles: "true"
  features.storage.buckets: "true"
  features.storage.volumes: "true"
  restricted: "true"
  restricted.containers.lowlevel: allow
  restricted.containers.nesting: allow
```

```
  restricted.containers.privilege: allow
  restricted.devices.disk: managed
  restricted.devices.gpu: allow
  restricted.snapshots: allow
  restricted.virtual-machines.lowlevel: allow
description: Project Handson-lab-3
name: Handson-lab-3
used_by:
- /1.0/profiles/default?project=Handson-lab-3
- /1.0/profiles/external-net?project=Handson-lab-3
- /1.0/profiles/internal-net?project=Handson-lab-3
- /1.0/profiles/storage?project=Handson-lab-3

mac% incus project show Handson-lab-4
config:
  features.images: "true"
  features.networks: "false"
  features.networks.zones: "false"
  features.profiles: "true"
  features.storage.buckets: "true"
  features.storage.volumes: "true"
  restricted: "true"
  restricted.containers.lowlevel: allow
  restricted.containers.nesting: allow
  restricted.containers.privilege: allow
  restricted.devices.disk: managed
  restricted.devices.gpu: allow
  restricted.snapshots: allow
  restricted.virtual-machines.lowlevel: allow
description: Project Handson-lab-4
name: Handson-lab-4
used_by:
- /1.0/profiles/default?project=Handson-lab-4
- /1.0/profiles/external-net?project=Handson-lab-4
- /1.0/profiles/internal-net?project=Handson-lab-4
- /1.0/profiles/storage?project=Handson-lab-4

mac% incus project show Handson-lab-5
config:
  features.images: "true"
  features.networks: "false"
  features.networks.zones: "false"
  features.profiles: "true"
  features.storage.buckets: "true"
```

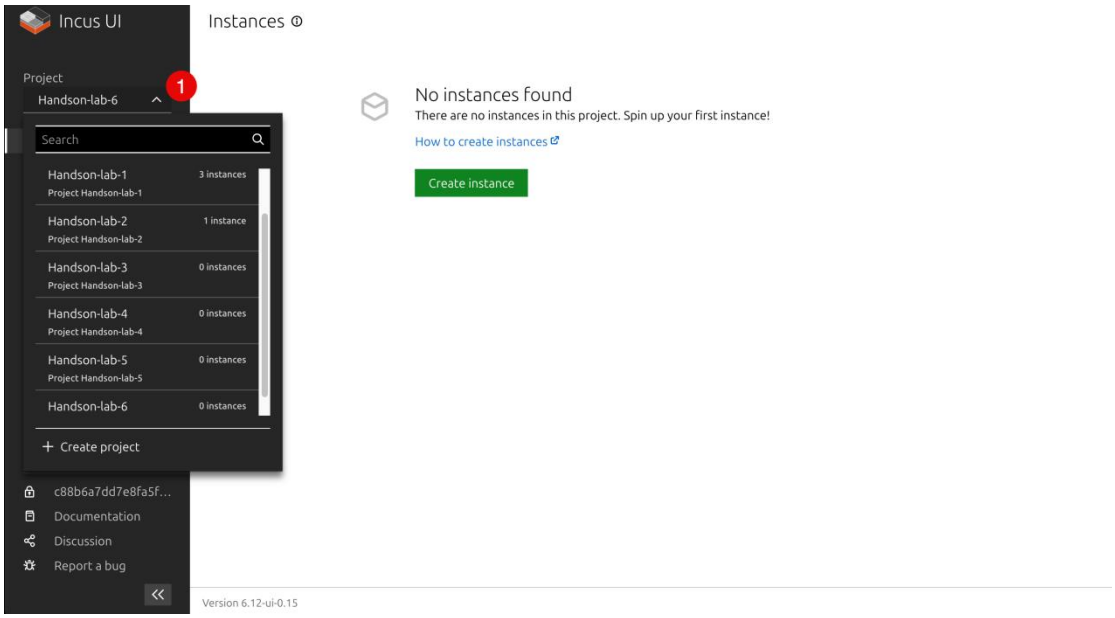
```
  features.storage.volumes: "true"
  restricted: "true"
  restricted.containers.lowlevel: allow
  restricted.containers.nesting: allow
  restricted.containers.privilege: allow
  restricted.devices.disk: managed
  restricted.devices.gpu: allow
  restricted.snapshots: allow
  restricted.virtual-machines.lowlevel: allow
description: Project Handson-lab-5
name: Handson-lab-5
used_by:
- /1.0/profiles/default?project=Handson-lab-5
- /1.0/profiles/external-net?project=Handson-lab-5
- /1.0/profiles/internal-net?project=Handson-lab-5
- /1.0/profiles/storage?project=Handson-lab-5

mac% incus project show Handson-lab-6
config:
  features.images: "true"
  features.networks: "false"
  features.networks.zones: "false"
  features.profiles: "true"
  features.storage.buckets: "true"
  features.storage.volumes: "true"
  restricted: "true"
  restricted.containers.lowlevel: allow
  restricted.containers.nesting: allow
  restricted.containers.privilege: allow
  restricted.devices.disk: allow
  restricted.devices.gpu: allow
  restricted.snapshots: allow
  restricted.virtual-machines.lowlevel: allow
description: Project Handson-lab-6
name: Handson-lab-6
used_by:
- /1.0/profiles/default?project=Handson-lab-6
- /1.0/profiles/external-net?project=Handson-lab-6
- /1.0/profiles/internal-net?project=Handson-lab-6
- /1.0/profiles/storage?project=Handson-lab-6

mac% incus storage list -f compact
     NAME      DRIVER               DESCRIPTION               USED BY   STATE
  basepool      zfs    passed to incus admin init                1       CREATED
```

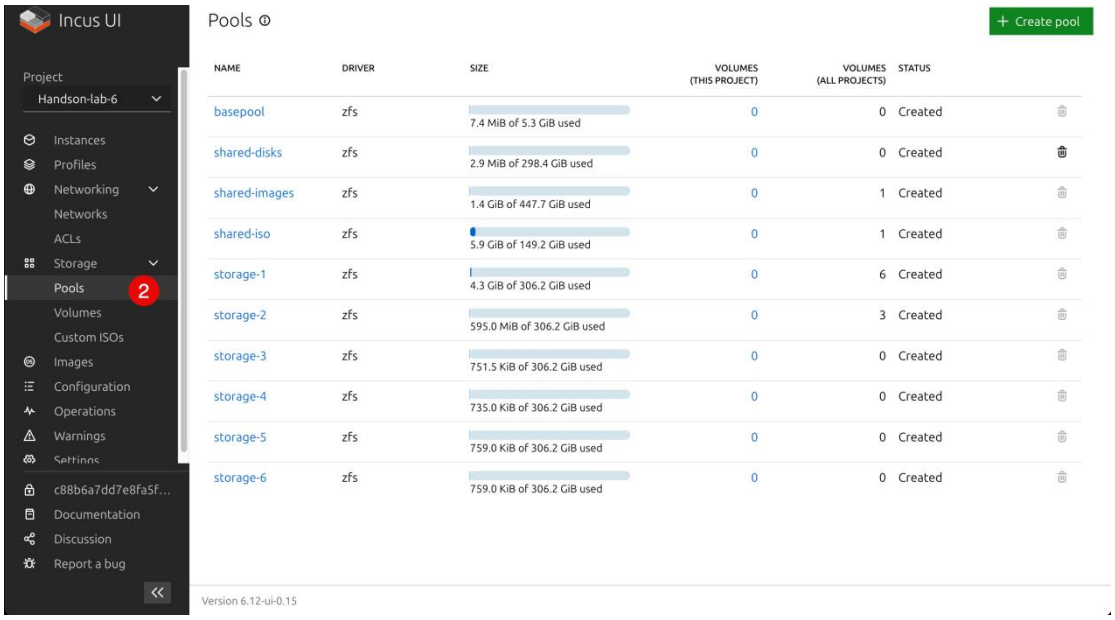
```
  shared-disks   zfs     shared disks accross Projects                0       CREATED
  shared-images  zfs     shared images accross Projects               1       CREATED
  shared-iso     zfs     shared ISOs accross Projects                 1       CREATED
  storage-1      zfs     Storage for Containers, VMs of Handson-lab-1  9       CREATED
  storage-2      zfs     Storage for Containers, VMs of Handson-lab-2  5       CREATED
  storage-3      zfs     Storage for Containers, VMs of Handson-lab-3  2       CREATED
  storage-4      zfs     Storage for Containers, VMs of Handson-lab-4  2       CREATED
  storage-5      zfs     Storage for Containers, VMs of Handson-lab-5  2       CREATED
  storage-6      zfs     Storage for Containers, VMs of Handson-lab-6  1       CREATED
```

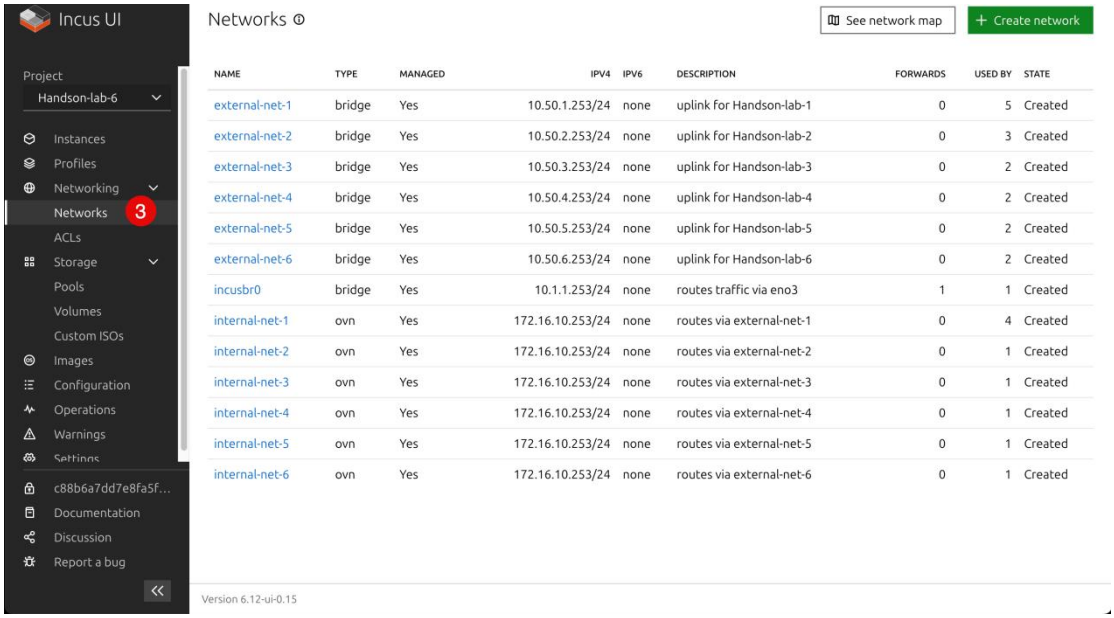
```
mac% incus network list -f compact
      NAME        TYPE   MANAGED       IPV4        IPV6        DESCRIPTION        USED BY   STATE
  external-net-1  bridge  YES     10.50.1.253/24    none  uplink for Handson-lab-1   5       CREATED
  external-net-2  bridge  YES     10.50.2.253/24    none  uplink for Handson-lab-2   3       CREATED
  external-net-3  bridge  YES     10.50.3.253/24    none  uplink for Handson-lab-3   2       CREATED
  external-net-4  bridge  YES     10.50.4.253/24    none  uplink for Handson-lab-4   2       CREATED
  external-net-5  bridge  YES     10.50.5.253/24    none  uplink for Handson-lab-5   2       CREATED
  external-net-6  bridge  YES     10.50.6.253/24    none  uplink for Handson-lab-6   2       CREATED
  incusbr0        bridge  YES     10.1.1.253/24     none  routes traffic via eno3    1       CREATED
  internal-net-1  ovn     YES     172.16.10.253/24  none  routes via external-net-1  4       CREATED
  internal-net-2  ovn     YES     172.16.10.253/24  none  routes via external-net-2  1       CREATED
  internal-net-3  ovn     YES     172.16.10.253/24  none  routes via external-net-3  1       CREATED
  internal-net-4  ovn     YES     172.16.10.253/24  none  routes via external-net-4  1       CREATED
  internal-net-5  ovn     YES     172.16.10.253/24  none  routes via external-net-5  1       CREATED
  internal-net-6  ovn     YES     172.16.10.253/24  none  routes via external-net-6  1       CREATED
```

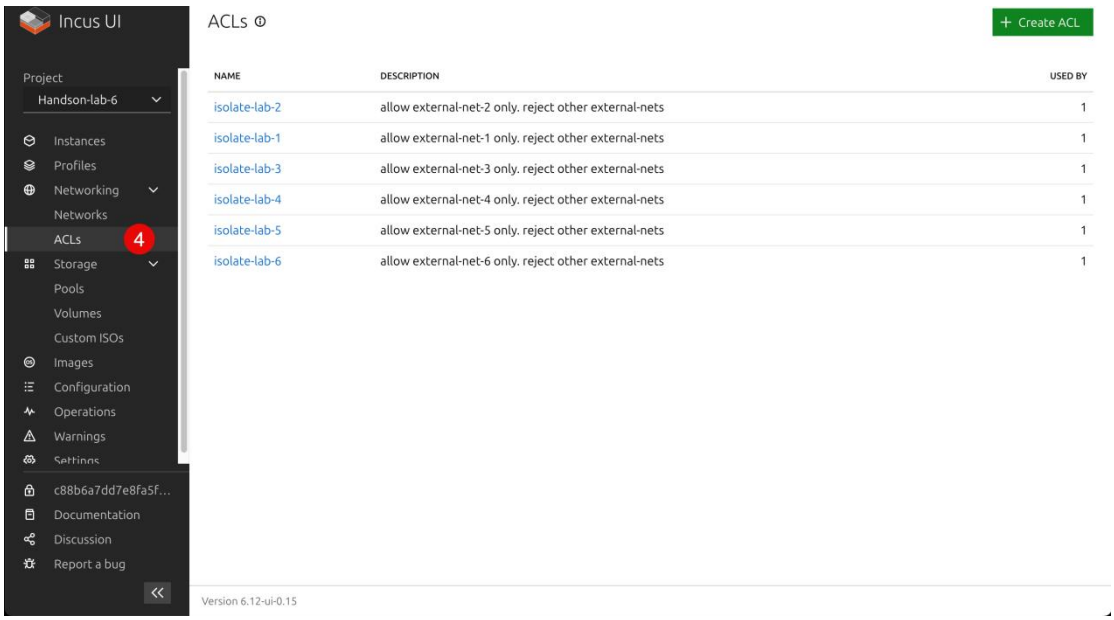
```
mac % incus network acl list -f compact
  NAME                        DESCRIPTION                      USED BY
  isolate-lab-1  allow external-net-1 only. reject other external-nets  1
  isolate-lab-2  allow external-net-2 only. reject other external-nets  1
  isolate-lab-3  allow external-net-3 only. reject other external-nets  1
  isolate-lab-4  allow external-net-4 only. reject other external-nets  1
  isolate-lab-5  allow external-net-5 only. reject other external-nets  1
  isolate-lab-6  allow external-net-6 only. reject other external-nets  1
```

With web UI


Incus UI
Instances
Project
Handson-lab-6
1
Search
Handson-lab-1
Project Handson-lab-1
3 instances
Handson-lab-2
Project Handson-lab-2
1 instance
Handson-lab-3
Project Handson-lab-3
0 instances
Handson-lab-4
Project Handson-lab-4
0 instances
Handson-lab-5
Project Handson-lab-5
0 instances
Handson-lab-6
0 instances
+ Create project
No instances found
There are no instances in this project. Spin up your first instance!
How to create instances
Create instance
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Version 6.12-ui-0.15


Incus UI
Pools
+ Create pool
Project
Handson-lab-6
Instances
Profiles
Networking
Networks
ACLs
Storage
Pools
2
Volumes
Custom ISOs
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
NAME
DRIVER
SIZE
VOLUMES (THIS PROJECT)
VOLUMES (ALL PROJECTS)
STATUS
basepool zfs 7.4 MiB of 5.3 GiB used 0 0 Created
shared-disks zfs 2.9 MiB of 298.4 GiB used 0 0 Created
shared-images zfs 1.4 GiB of 447.7 GiB used 0 1 Created
shared-iso zfs 5.9 GiB of 149.2 GiB used 0 1 Created
storage-1 zfs 4.3 GiB of 306.2 GiB used 0 6 Created
storage-2 zfs 595.0 MiB of 306.2 GiB used 0 3 Created
storage-3 zfs 751.5 KiB of 306.2 GiB used 0 0 Created
storage-4 zfs 735.0 KiB of 306.2 GiB used 0 0 Created
storage-5 zfs 759.0 KiB of 306.2 GiB used 0 0 Created
storage-6 zfs 759.0 KiB of 306.2 GiB used 0 0 Created
Version 6.12-ui-0.15


Incus UI
Project
Handson-lab-6
Instances
Profiles
Networking
Networks
3
ACLs
Storage
Pools
Volumes
Custom ISOs
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
Networks
See network map
Create network
NAME TYPE MANAGED IPV4 IPV6 DESCRIPTION FORWARDS USED BY STATE
external-net-1 bridge Yes 10.50.1.253/24 none uplink for Handson-lab-1 0 5 Created
external-net-2 bridge Yes 10.50.2.253/24 none uplink for Handson-lab-2 0 3 Created
external-net-3 bridge Yes 10.50.3.253/24 none uplink for Handson-lab-3 0 2 Created
external-net-4 bridge Yes 10.50.4.253/24 none uplink for Handson-lab-4 0 2 Created
external-net-5 bridge Yes 10.50.5.253/24 none uplink for Handson-lab-5 0 2 Created
external-net-6 bridge Yes 10.50.6.253/24 none uplink for Handson-lab-6 0 2 Created
incusbr0 bridge Yes 10.1.1.253/24 none routes traffic via eno3 1 1 Created
internal-net-1 ovn Yes 172.16.10.253/24 none routes via external-net-1 0 4 Created
internal-net-2 ovn Yes 172.16.10.253/24 none routes via external-net-2 0 1 Created
internal-net-3 ovn Yes 172.16.10.253/24 none routes via external-net-3 0 1 Created
internal-net-4 ovn Yes 172.16.10.253/24 none routes via external-net-4 0 1 Created
internal-net-5 ovn Yes 172.16.10.253/24 none routes via external-net-5 0 1 Created
internal-net-6 ovn Yes 172.16.10.253/24 none routes via external-net-6 0 1 Created
Version 6.12-ui-0.15


Incus UI
Project
Handson-lab-6
Instances
Profiles
Networking
Networks
ACLs
4
Storage
Pools
Volumes
Custom ISOs
Images
Configuration
Operations
Warnings
Settings
c88b6a7dd7e8fa5f...
Documentation
Discussion
Report a bug
ACLs
Create ACL
NAME DESCRIPTION USED BY
isolate-lab-2 allow external-net-2 only. reject other external-nets 1
isolate-lab-1 allow external-net-1 only. reject other external-nets 1
isolate-lab-3 allow external-net-3 only. reject other external-nets 1
isolate-lab-4 allow external-net-4 only. reject other external-nets 1
isolate-lab-5 allow external-net-5 only. reject other external-nets 1
isolate-lab-6 allow external-net-6 only. reject other external-nets 1
Version 6.12-ui-0.15

Credits and Thanks to my incus Gurus

Stephane Graber, Zabbly
https://www.youtube.com/@TheZabbly/videos

Trevor Sullivan
https://www.youtube.com/@TrevorSullivan/videos

Simos Xenitellis
https://blog.simos.info/

Scott at Scotti-BYTE
https://www.youtube.com/@scottibyte/videos

Incus community
https://discuss.linuxcontainers.org

Deepseek.com
https://www.deepseek.com/en

About the Author

Coordinates:

Name : Ananda Kammampati
Contact : ananda at fieldday dot io
Profile : https://www.linkedin.com/in/anandakammampati

Humble Brag:

1. Founder www.fieldday.io
2. Creator of StudentOS: http://fieldday.io/StudentOS
3. Conceptualized and designed 'Hands-on-Box' , a unique portable server appliance for the needs of field and training (http://www.fieldday.io/hands-on-box/)
4. Published articles:
   https://www.linkedin.com/in/anandakammampati/recent-activity/articles/
5. More articles in fieldddday.io website
6. Repository:
   https://github.com/fieldday-io/books

- Education:

Bachelors in Computer Technology, Government College of Technology, Coimbatore, TN, India

- Many hats:

Technical Product Manager, Senior Solutions Architect, BizDev Manager, Technical Lead, R & D Engineer, Computer Operator Trainee

- Learned at:

1. VMWare (CA, USA)
2. Sun Microsystems (CA, USA)
3. HCL Consulting (CA, USA)
4. BFL Software Ltd (Bengaluru)
5. Tata Elxsi India Ltd (Bengaluru)
6. Supercomputer Education & Research Center (IISc, Bengaluru)