教育部

行動通訊主題式跨層次系統整合教學聯盟

次世代超高速傳輸系統整合聯盟中心

本期計畫：自 112 年 10 月 1 日 起 至 114 年 3 月 31 日

可推廣的教材(含教材簡報、實作手冊)
實作平台一、B5G/6G 數位分身網路實作平台

國立虎尾科技大學：蘇暉凱 教授 鄭佳炘 教授

中 華 民 國 1 1 4 年 0 2 月 2 1 日

國立虎尾科技大學
NATIONAL FORMOSA UNIVERSITY

B5G/6G數位分身網路跨層整合實作平台

實驗模組一：B5G/6G行動寬頻網路環境建置

國立虎尾科技大學 電機工程系
蘇暉凱 教授、鄭佳炘 教授

2024年7月

國立虎尾科技大學
NATIONAL FORMOSA UNIVERSITY

Outline

- 實驗目的
- 背景知識
- 實驗環境
- Stage 1: 環境架設
- Stage 2: srsRAN Project、 Free5GC 編譯安裝
  - Step1: 安裝依賴套件
  - Step2: 下載 srsRAN Project原始碼
  - Step3: 編譯 srsRAN Project
  - Step4: 安裝 srsRAN Project 執行檔
  - Step5: 安裝 srsRAN Project預設設定檔
  - Step6: 安裝Free5GC
- Stage 3: srsRAN Project 、 Free5GC
- 執行及測試

國立虎尾科技大學
NATIONAL FORMOSA UNIVERSITY

實驗目的

1. 開發5GC核網中新的NF/AF
2. 使用USIM並透過商用手機連上自建專網

背景知識 – USIM

- USIM（UMTS Subscriber Identity Module)，為用於UMTS中的使用者身分辨識模組
- USIM還可儲存
  - 使用者資料
  - 電話號碼
  - 認證資料
  - 簡訊儲存空間

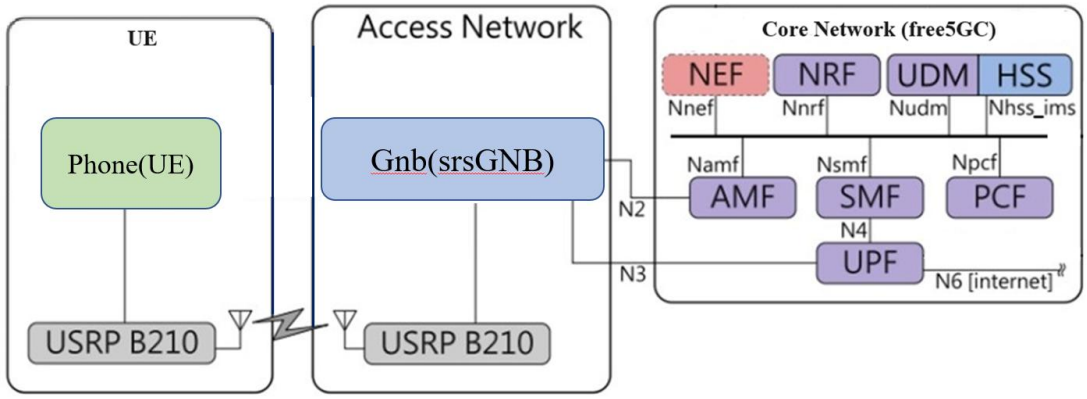
軟硬體環境 – 硬體

名稱	規格	數量	目的
PC	電腦型號： ASUS VivoBook 15	1	啟動5GC與gNB
	USRP B210	1	讓 gNB 透過 USRP 與 UE 溝通
UE	手機型號： ASUS ZenFone 8	1	作為 UE UE1的IMSI：001010000000012
	SIM卡(需有IMSI、MSISDN、K、AMF、USIM Type與Operator Key資訊)	1	使商用手機連上自建專網

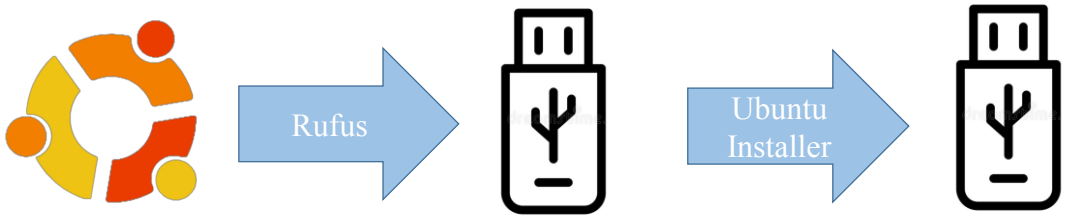
軟硬體環境 – 軟體

名稱	軟體	版本
PC	OS : Ubuntu	Ubuntu 22.04.1
	srsRAN_Project	srsRAN_Project22.04
	free5GC	V3.3.0
	UHD	v4.1.0.7
UE	OS : Android	Android 11
	PingTools Network Utilities	v4.52

實驗架構



隨身系統製作流程



Ubuntu.iso

開機USB
(USB 1)

系統USB
(USB 2)

燒錄軟體安裝

安裝燒錄軟體balenaEtcher

DOWNLOAD

Download Etcher

ASSET	OS	ARCH	
ETCHER FOR WINDOWS (X86|X64) (INSTALLER)	WINDOWS	X86|X64	Download
ETCHER FOR WINDOWS (X86|X64) (PORTABLE)	WINDOWS	X86|X64	Download
ETCHER FOR WINDOWS (LEGACY 32 BIT) (X86|X64) (PORTABLE)	WINDOWS	X86|X64	Download
ETCHER FOR MACOS	MACOS	X64	Download
ETCHER FOR LINUX X64 (64-BIT) (APPIMAGE)	LINUX	X64	Download
ETCHER FOR LINUX (LEGACY 32 BIT) (APPIMAGE)	LINUX	X86	Download

Looking for Debian (.deb) packages or Red Hat (.rpm) packages?

OSS hosting by cloudsmith

Ubuntu 安裝(製作系統USB)

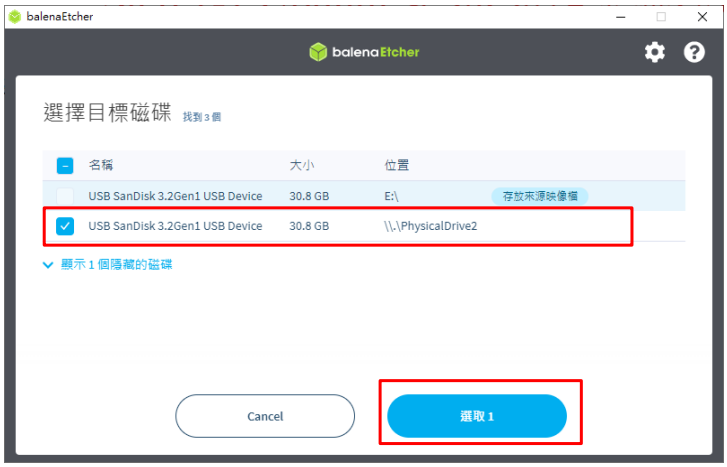
開啟balenaEtcher，選擇再製磁碟

Ubuntu 安裝(製作系統USB)

選擇開機碟(原始系統)，按選取

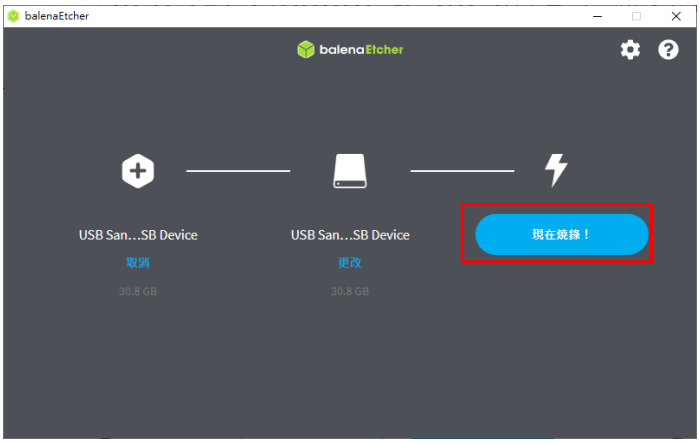
Ubuntu 安裝(製作系統USB)

選擇系統碟(實驗使用的USB)，按選取



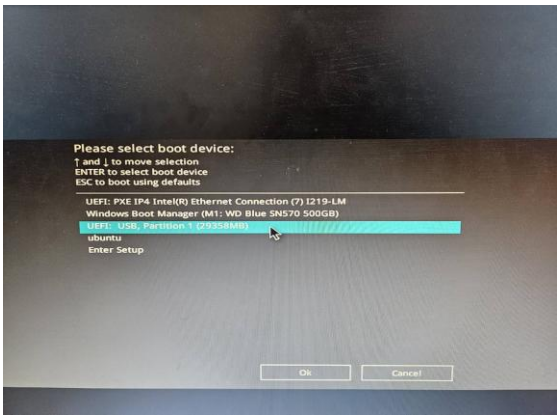
Ubuntu 安裝(製作系統USB)

點選現在燒錄，等待完成



開機方式

開機按F8，進入後，選擇USB(系統碟)開機，即可進入Ubuntu系統



初次使用(終端機)

因為Ubuntu系統的更新較為頻繁，操作實驗前先按”ctrl + alt + T”以開啟Terminal視窗，然後輸入sudo apt update&&sudo apt upgrade 並輸入密碼來升級套件以避免後續安裝產生套件不支援的錯誤

再輸入sudo apt install build-essential vim&&sudo apt-get install git

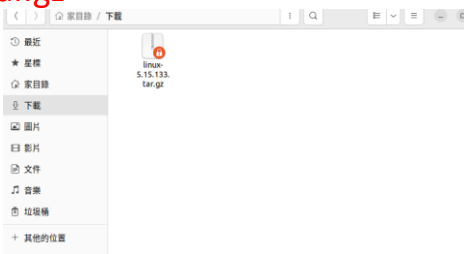
來安裝常用軟體

```
nuk@nuk-UN65H:~$ sudo apt update&&sudo apt upgrade
```

```
nuk@nuk-UN65H:~$ sudo apt-get install git build-essential vim
Reading package lists... Done
Building dependency tree
Reading state information... Done
git is already the newest version (1:2.25.1-1ubuntu3.1).
The following additional packages will be installed:
  dpkg-dev fakeroot g++ g++-9 libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl
  libstdc++-9-dev vim-runtime
Suggested packages:
  debian-keyring g++-multilib g++-9-multilib gcc-9-doc libstdc++-9-doc ctags vim-doc vim-scripts
The following NEW packages will be installed:
  build-essential dpkg-dev fakeroot g++ g++-9 libalgorithm-diff-perl libalgorithm-diff-xs-perl
```


下載安裝kernel

- 輸入:cd Downloads 如果用中文 cd ~/下載
- 輸入:sudo wget https://mirrors.edge.kernel.org/pub/linux/kernel/v5.x/linux-5.15.133.tar.gz && sudo tar -xvf linux-5.15.133.tar.gz -C ~/
- 解壓縮: tar -xzvf linux-5.15.133.tar.gz



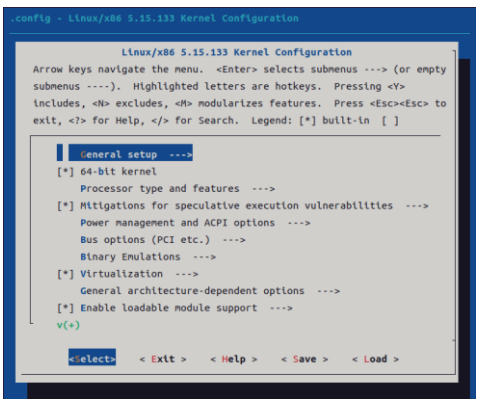
下載安裝kernel

- cd ../linux-5.15.133
- sudo apt install build-essential libncurses-dev libssl-dev libelf-dev bison flex -y

```
bluedog@bluedog-VirtualBox:~/linux-5.15.133$ sudo apt install build-essential li
bncurses-dev libssl-dev libelf-dev bison flex -y
```


下載安裝kernel

- 輸入sudo make menuconfig會看到以下畫面
- 先選擇 Save => Ok =>Exit
- 然後在下面 Exit 退出



開始編譯

- 可以依照自身需求來使多少核心進行編譯 make –j 為全核心，make –j2 為雙核心
- 由於編譯時間需好幾分鐘，需耐心等待
- sudo scripts/config - disable SYSTEM_TRUSTED_KEYS
- sudo scripts/config - disable SYSTEM_REVOCATION_KEYS
- sudo apt-get install dwarves
- sudo make -j2 && sudo make -j2 modules_install && sudo make -j2 install 如果出現選項請輸入3然後enter

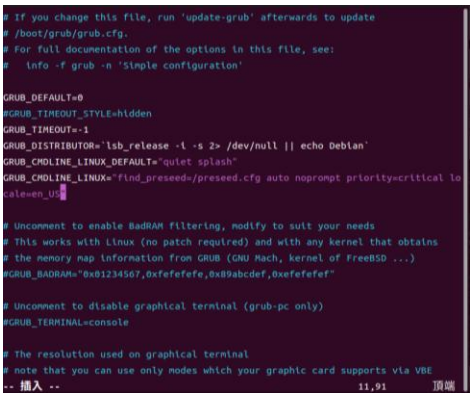
安裝 kernel

- **進入 grub 修改文件**
- sudo vim /etc/default/grub

安裝 kernel

- 這裡需要用到 vim 指令，按 i 進入編輯，修改完成後按esc 跳出編輯模式，在輸入 :wq
- 把看到的文件內容修改成下方文字

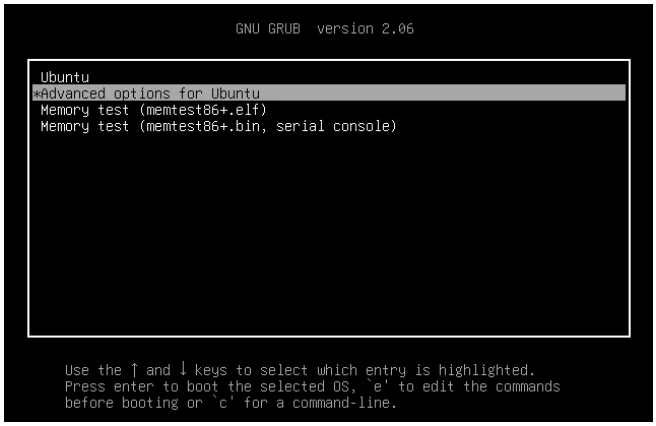
```
GRUB_DEFAULT=0
#GRUB_TIMEOUT_STYLE=hidden
GRUB_TIMEOUT=-1
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
GRUB_CMDLINE_LINUX="find_preseed=/preseed.cfg auto noprompt priority=critical locale=en_us"
```



安裝 kernel

- 修改完成後進入終端機，並輸入
- sudo update-grub
- reboot

選擇 advance option for ubuntu

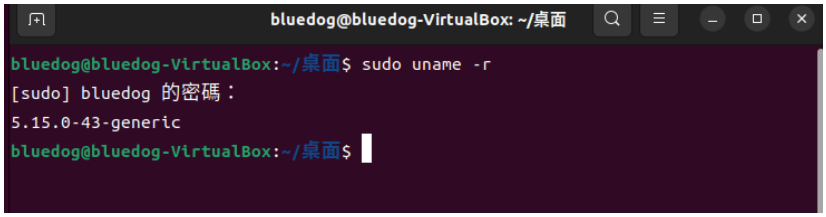


選擇 linux 5.15.xx版本

確認安裝是否成功

確認開機後kernel版本

- sudo uname -r



Stage 1 Check List

	項目	內容
	系統USB	開機按F8，選擇USB(系統碟)作為開機裝置後，順利進入Ubuntu 如未能順利進入，可切換選擇同USB之不同Partition
	作業系統版本	cat /etc/lsb-release 確認是否為22.04.4版本
	更新並升級	sudo apt update&&sudo apt upgrade
	Linux內核版本	uname –r 5.4.0-65-generic

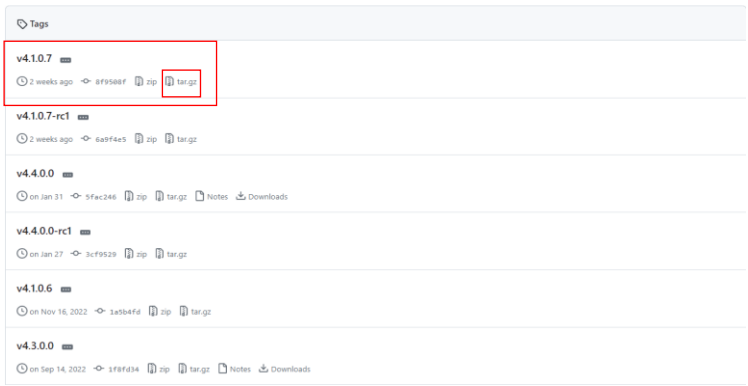
安裝UHD

- 在終端機執行
- sudo apt-get update && sudo apt-get upgrade
- 安裝所需套件
- sudo apt-get install libboost-all-dev libusb-1.0-0-dev doxygen python3-docutils python3-mako python3-numpy python3-requests python3-ruamel.yaml python3-setuptools cmake build-essential

https://blog.csdn.net/qq_41300075/article/details/120914018

下載目標UHD版本

- cd ~/Downloads/
- wget https://github.com/EttusResearch/uhd/archive/refs/tags/v4.1.0.7.tar.gz



https://github.com/EttusResearch/UHD/tags

UHD解壓縮

- tar zxvf v4.1.0.7.tar.gz

UHD編譯

- cd uhd-4.1.0.7/host
- mkdir build
- cd build
- cmake ../

```
--
-- #################################################
-- # UHD enabled components
-- #################################################
--   * LibUHD
--   * LibUHD - C API
--   * LibUHD - Python API
--   * Examples
--   * Utils
--   * Tests
--   * USB
--   * B100
--   * B200
--   * USRP1
--   * USRP2
--   * X300
--   * MPMD
--   * SIM
--   * N300
--   * N320
--   * E320
--   * E300
--   * X400
--   * OctoClock
--   * Manual
--   * API/Doxygen
--   * Man Pages
--
-- #################################################
-- # UHD disabled components
-- #################################################
--   * DPDK
--
-- Building version: 4.1.0.4-0-unknown
-- Using install prefix: /usr/local
-- Configuring done
```

UHD安裝

- 再輸入：
  make
  make時間較長，大約20分鐘，輸入如下指令測試：
- make test
  進行UHD的安裝：
- sudo make install
  注意：安裝更新結束都需執行如下指令，以確保能有效識別！
  sudo ldconfig
  更新環境路徑：
  export LD_LIBRARY_PATH=/usr/local/lib

UHD 測試

再依照訊息安裝python檔，sudo /usr/local/lib/uhd/utils/uhd_images_downloader.py

```
No UHD Devices Found
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ sudo /usr/local/lib/uhd/utils/uhd_images_downloader.py
[INFO] Using base URL: https://files.ettus.com/binaries/cache/
[INFO] Images destination: /usr/local/share/uhd/images
[INFO] No inventory file found at /usr/local/share/uhd/images/inventory.json. Creating an empty one.
09146 kB / 09146 kB (100%) x4xx_x410_fpga_default-g26793b8.zip
21085 kB / 21085 kB (100%) x3xx_x310_fpga_default-g26793b8.zip
19610 kB / 19610 kB (100%) x3xx_x300_fpga_default-g26793b8.zip
01153 kB / 01153 kB (100%) e3xx_e310_sg1_fpga_default-g26793b8.zip
01138 kB / 01138 kB (100%) e3xx_e310_sg3_fpga_default-g26793b8.zip
10156 kB / 10156 kB (100%) e3xx_e320_fpga_default-g26793b8.zip
20731 kB / 20731 kB (100%) n3xx_n310_fpga_default-g26793b8.zip
```

UHD 測試

將USRP接上電腦的USB 3.0
執行測試是否安裝成功 $ sudo uhd_find_devices，再依照訊息安裝python檔

```
a1111@1111:~$ sudo uhd_find_devices
[INFO] [UHD] linux; GNU C++ version 11.4.0; Boost_107400; UHD_4.1.0.7-0-unknown
--------------------------------------------------
-- UHD Device 0
--------------------------------------------------
Device Address:
    serial: 31F66A7
    name: MyB210
    product: B210
    type: b200
```


安裝 srsRAN Project

- sudo apt-get install cmake make gcc g++ pkg-config libfftw3-dev libmbedtls-dev libsctp-dev libyaml-cpp-dev libgtest-dev
- git clone https://github.com/srsRAN/srsRAN_Project.git

```
nfu5g@nfu5g-WS-E500-G5-WS690T: ~
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ sudo apt-get install cmake make gcc g++ pkg-config libfftw3
-dev libmbedtls-dev libsctp-dev libyaml-cpp-dev libgtest-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
g++ is already the newest version (4:11.2.0-1ubuntu1).
g++ set to manually installed.
```

安裝 srsRAN Project

- cd srsRAN_Project
- mkdir build
- cd build
- cmake ../
- make
- make test
- sudo make install

```
nfu5g@nfu5g-WS-E500-G5-WS690T: ~/srsRAN_Project/build
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ cd srsRAN_Project
nfu5g@nfu5g-WS-E500-G5-WS690T:~/srsRAN_Project$ mkdir build
nfu5g@nfu5g-WS-E500-G5-WS690T:~/srsRAN_Project$ cd build
nfu5g@nfu5g-WS-E500-G5-WS690T:~/srsRAN_Project/build$ cmake ../ -DENABLE_EXPORT=ON -DENABLE_
ZEROMQ=ON
-- The C compiler identification is GNU 11.4.0
-- The CXX compiler identification is GNU 11.4.0
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: /usr/bin/cc - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Build type not specified: defaulting to Release.
-- Performing Test HAS_MAYBE_UNINITIALIZED
-- Performing Test HAS_MAYBE_UNINITIALIZED - Success
-- Performing Test HAS_STRINGOP_OVERFLOW
-- Performing Test HAS_STRINGOP_OVERFLOW - Success
-- Performing Test HAVE_NON_VIRTUAL_DTOR
-- Performing Test HAVE_NON_VIRTUAL_DTOR - Success
-- Performing Test HAVE_SUGGEST_OVERRIDE
-- Performing Test HAVE_SUGGEST_OVERRIDE - Success
-- Performing Test HAVE_SHADOW
```


新增gNB設定檔

- cd ~/srsRAN_Project/build/apps/gnb
- vim gnb.yml
- 將右邊程式全選複製貼上

```
# This example configuration outlines how to configure the srsRAN Project gNB to create a single TDD cell
# transmitting in band 78, with 20 MHz bandwidth and 30 kHz sub-carrier-spacing. A USRP B200 is configured
# as the RF frontend using split 8. Note in this example an external clock source is not used, so the sync
# is not defined and the default is used.

cu_cp:
  amf:
    addr: 127.0.1.100
    bind_addr: 127.0.0.1
    supported_tracking_areas:
      - tac: 1
        plmn_list:
          - plmn: "00101"
            tai_slice_support_list:
              - sst: 1

ru_sdr:
  device_driver: uhd
  device_args: type=b200,num_recv_frames=64,num_send_frames=64
  clock: internal
  srate: 23.04
  otw_format: sc12
  tx_gain: 80
  rx_gain: 50

cell_cfg:
  dl_arfcn: 428000
  band: 1
  channel_bandwidth_MHz: 20
  common_scs: 15
  plmn: "00101"
  tac: 1
  pci: 1

log:
  filename: /tmp/gnb.log
  all_level: warning

pcap:
  mac_enable: false
  mac_filename: /tmp/gnb_mac.pcap
  ngap_enable: false
  ngap_filename: /tmp/gnb_ngap.pcap
```

free5gc安裝GO

- wget https://dl.google.com/go/go1.21.8.linux-amd64.tar.gz
- sudo tar -C /usr/local -zxvf go1.21.8.linux-amd64.tar.gz
- mkdir -p ~/go/{bin,pkg,src}
- # The following assume that your shell is bash:
- echo 'export GOPATH=$HOME/go' >> ~/.bashrc
- echo 'export GOROOT=/usr/local/go' >> ~/.bashrc
- echo 'export PATH=$PATH:$GOPATH/bin:$GOROOT/bin' >> ~/.bashrc
- echo 'export GO111MODULE=auto' >> ~/.bashrc
- source ~/.bashrc

如果安裝了其他版本的 Go，請刪除現有版本並安裝 Go 1.21.8：# this assumes your current version of Go is in the default location:
sudo rm -rf /usr/local/go
wget https://dl.google.com/go/go1.21.8.linux-amd64.tar.gz
sudo tar -C /usr/local -zxvf go1.21.8.linux-amd64.tar.gz

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ wget https://dl.google.com/go/go1.21.8.linux-amd64.tar.gz
sudo tar -C /usr/local -zxvf go1.21.8.linux-amd64.tar.gz
mkdir -p ~/go/{bin,pkg,src}
# The following assume that your shell is bash:
echo 'export GOPATH=$HOME/go' >> ~/.bashrc
echo 'export GOROOT=/usr/local/go' >> ~/.bashrc
echo 'export PATH=$PATH:$GOPATH/bin:$GOROOT/bin' >> ~/.bashrc
echo 'export GO111MODULE=auto' >> ~/.bashrc
source ~/.bashrc
--2024-06-04 23:15:58--  https://dl.google.com/go/go1.21.8.linux-amd64.tar.gz
Resolving dl.google.com (dl.google.com)... 142.251.42.238, 2404:6800:4012:3::200e
Connecting to dl.google.com (dl.google.com)|142.251.42.238|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 66647874 (64M) [application/x-gzip]
Saving to: 'go1.21.8.linux-amd64.tar.gz'
```


查看 free5gc安裝GO版本

- go version

Go 1.21.8

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ go version
go version go1.21.8 linux/amd64
```

控制平面支援包

在開始安裝之前，請更新包管理器資料庫並確保已安裝 MongoDB 必備元件
sudo apt update
sudo apt install gnupg curl
添加 MongoDB 公有 GPG 金鑰
curl -fsSL https://pgp.mongodb.com/server-7.0.asc | \
  sudo gpg -o /usr/share/keyrings/mongodb-server-7.0.gpg --dearmor
在 Ubuntu Server 22.04.03 上安裝 MongoDB 7.0.x
echo "deb [arch=amd64,arm64 signed-by=/usr/share/keyrings/mongodb-server-7.0.gpg] https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/7.0 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-7.0.list

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ curl -fsSL https://pgp.mongodb.com/server-7.0.asc | \
   sudo gpg -o /usr/share/keyrings/mongodb-server-7.0.gpg --dearmor
```


啟動mongodb查看狀態

- sudo apt update
- sudo apt install -y mongodb-org
- sudo systemctl start mongod
- sudo systemctl enable mongod
- sudo systemctl status mongod

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ sudo apt update
sudo apt install -y mongodb-org
Hit:1 http://tw.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://tw.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://tw.archive.ubuntu.com/ubuntu jammy-backports InRelease
Ign:4 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/7.0 InRelease
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:6 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/7.0 Release [2090 B]
Get:7 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/7.0 Release.gpg [866 B]
Hit:8 https://ppa.launchpadcontent.net/obsproject/obs-studio/ubuntu jammy InRelease
Get:9 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/7.0/multiverse arm64 Packages [39.8 kB
```

```
Processing triggers for man-db (2.10.2-1) ...
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ sudo systemctl start mongod
```

安裝用戶層面支援包

- sudo apt -y update
- sudo apt -y install git gcc g++ cmake autoconf libtool pkg-config libmnl-dev libyaml-dev

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ sudo apt -y update
sudo apt -y install git gcc g++ cmake autoconf libtool pkg-config libmnl-dev libyaml-dev
Hit:1 http://tw.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://tw.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://tw.archive.ubuntu.com/ubuntu jammy-backports InRelease
Ign:4 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/7.0 InRelease
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:6 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/7.0 Release
Hit:8 https://ppa.launchpadcontent.net/obsproject/obs-studio/ubuntu jammy InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
4 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
g++ is already the newest version (4:11.2.0-1ubuntu1).
gcc is already the newest version (4:11.2.0-1ubuntu1).
pkg-config is already the newest version (0.29.2-1ubuntu3).
cmake is already the newest version (3.22.1-1ubuntu1.22.04.2).
```


查詢網卡名稱

- cd
- sudo ip addr -找尋自己外網網卡名稱

```
nfu5g@nfu5g-ASUS-EXPERTCENTER-D900MC-M900SC:~$ sudo ip addr
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
2: enp0s31f6: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether 50:eb:f6:54:eb:37 brd ff:ff:ff:ff:ff:ff
3: wlx04d9f5c143d0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 04:d9:f5:c1:43:d0 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.106/24 brd 192.168.0.255 scope global dynamic noprefixroute wlx04d9f5c143d0
       valid_lft 587854sec preferred_lft 587854sec
    inet6 fe80::2969:909e:ab89:5a60/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
20: upfgtp: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1464 qdisc noqueue state UNKNOWN group default qlen 1000
    link/none
    inet6 fe80::c1f0:8fa2:fbc2:a080/64 scope link stable-privacy
       valid_lft forever preferred_lft forever
nfu5g@nfu5g-ASUS-EXPERTCENTER-D900MC-M900SC:~$
```

Linux 主機網路設置(關機後須重新設置)

輸入ip addr查詢

- sudo sysctl -w net.ipv4.ip_forward=1
- sudo iptables -t nat -A POSTROUTING -o <dn_interface> -j MASQUERADE
- sudo iptables -A FORWARD -p tcp -m tcp --tcp-flags SYN,RST SYN -j TCPMSS --set-mss 1400
- sudo systemctl stop ufw
- sudo systemctl disable ufw # prevents the firewall to wake up after a OS reboot

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ sudo sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ sudo iptables -t nat -A POSTROUTING -o enp1s0 -j MASQUERADE
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ sudo iptables -A FORWARD -p tcp -m tcp --tcp-flags SYN,RST SYN -j TC
PMSS --set-mss 1400
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ sudo systemctl stop ufw
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ sudo systemctl disable ufw # prevents the firewall to wake up after
a OS reboot
```


安裝核網

- cd ~
- git clone --recursive -b v3.3.0 -j `nproc` https://github.com/free5gc/free5gc.git
- cd free5gc

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ cd ~
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ git clone --recursive -b v3.3.0 -j `nproc` https://github.com/free5g
c/free5gc.git
Cloning into 'free5gc'...
remote: Enumerating objects: 2793, done.
remote: Counting objects: 100% (566/566), done.
remote: Compressing objects: 100% (243/243), done.
remote: Total 2793 (delta 383), reused 469 (delta 320), pack-reused 2227
Receiving objects: 100% (2793/2793), 912.32 KiB | 5.07 MiB/s, done.
Resolving deltas: 100% (1632/1632), done.
Note: switching to '470b8bff3ff2cd4e485bf54cb4c6bfe7e285793e'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -
```

編譯網路功能服務free5gc

- cd ~/free5gc
- make

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~/free5gc$ cd ~/free5gc
make
Start building amf....
cd NFs/amf/cmd && \
CGO_ENABLED=0 go build -gcflags "" -ldflags "-X github.com/free5gc/util/version.VERSION=v3.4.1 -X
github.com/free5gc/util/version.BUILD_TIME=2024-06-04T15:28:43Z -X github.com/free5gc/util/version
.COMMIT_HASH=4d01bec2 -X github.com/free5gc/util/version.COMMIT_TIME=2024-03-27T15:58:48Z" -o /hom
e/nfu5g/free5gc/bin/amf main.go
go: downloading github.com/free5gc/util v1.0.6
go: downloading github.com/urfave/cli v1.22.5
go: downloading github.com/free5gc/openapi v1.0.8
go: downloading github.com/gin-contrib/cors v1.3.1
go: downloading github.com/sirupsen/logrus v1.8.1
go: downloading github.com/asaskevich/govalidator v0.0.0-20210307081110-f21760c49a8d
go: downloading gopkg.in/yaml.v2 v2.4.0
```


檢索 5G GTP-U 內核模組並構建

- cd ~
- git clone -b v0.8.6 https://github.com/free5gc/gtp5g.git
- cd gtp5g
- make
- sudo make install

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~/free5gc$ cd
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ git clone -b v0.8.6 https://github.com/free5gc/gtp5g.git
cd gtp5g
make
sudo make install
Cloning into 'gtp5g'...
remote: Enumerating objects: 902, done.
remote: Counting objects: 100% (423/423), done.
remote: Compressing objects: 100% (180/180), done.
remote: Total 902 (delta 332), reused 282 (delta 243), pack-reused 479
Receiving objects: 100% (902/902), 319.93 KiB | 922.00 KiB/s, done.
Resolving deltas: 100% (577/577), done.
Note: switching to 'd8818ee80a9a004ea0fac3715415395810666921'.
```

安裝 WebConsole

- cd ~
- curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash -
- sudo apt update
- sudo apt install -y nodejs
- sudo corepack enable # setup yarn automatically

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~/gtp5g$ cd
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash -

sudo apt update
sudo apt install -y nodejs
corepack enable # setup yarn automatically
2024-06-04 23:39:19 - Installing pre-requisites
Hit:1 http://tw.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://tw.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://tw.archive.ubuntu.com/ubuntu jammy-backports InRelease
Ign:4 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/7.0 InRelease
Hit:5 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/7.0 Release
Hit:7 https://ppa.launchpadcontent.net/obsproject/obs-studio/ubuntu jammy InRelease
Get:8 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Fetched 129 kB in 4s (35.5 kB/s)
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ sudo corepack enable # setup yarn automatically
```


構建 WebConsole

- cd ~/free5gc
- make webconsole

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~$ cd ~/free5gc
nfu5g@nfu5g-WS-E500-G5-WS690T:~/free5gc$ make webconsole
Start building webconsole....
cd webconsole/frontend && \
yarn install && \
yarn build && \
rm -rf ../public && \
cp -R build ../public
! Corepack is about to download https://registry.yarnpkg.com/yarn/-/yarn-1.22.22.tgz
? Do you want to continue? [Y/n] y

! The local project doesn't define a 'packageManager' field. Corepack will now add one referencing ya
rn@1.22.22+sha512.a6b2f7906b721bba3d67d4aff083df04dad64c399707841b7acf00f6b133b7ac24255f2652fa22ae353
4329dc6180534e98d17432037ff6fd140556e2bb3137e.
! For more details about this field, consult the documentation at https://nodejs.org/api/packages.htm
l#packagemanager

yarn install v1.22.22
warning package-lock.json found. Your project contains lock files generated by tools other than Yarn.
 It is advised not to mix package managers in order to avoid resolution inconsistencies caused by uns
ynchronized lock files. To clear this warning, remove package-lock.json.
[1/4] Resolving packages...
[2/4] Fetching packages...
[################################################################################----------] 1085/1321
```

Test free5GC

- cd ~/free5gc
- chmod +x ./test.sh
- ./test.sh TestRegistration
- ./test.sh TestGUTIRegistration
- ./test.sh TestServiceRequest
- ./test.sh TestXnHandover
- ./test.sh TestDeregistration
- ./test.sh TestPDUSessionReleaseRequest
- ./test.sh TestPaging
- ./test.sh TestN2Handover
- ./test.sh TestNon3GPP
- ./test.sh TestReSynchronization
- ./test_ulcl.sh TestRequestTwoPDUSessions

看到ok pass即可

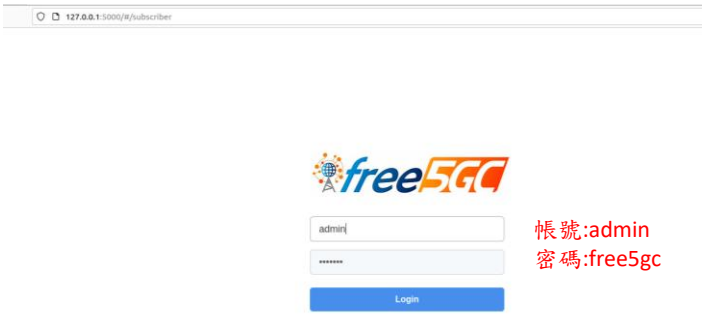
啟動 WebConsole 伺服器

- cd ~/free5gc/webconsole
- ./bin/webconsole

```
nfu5g@nfu5g-WS-E500-G5-WS690T:~/UERANSIM$ cd ~/free5gc/webconsole
./bin/webconsole
2024-06-05T00:09:54.341881674+08:00 [INFO][WEBUI][Main] WEBUI version:
        free5GC version: v3.4.1
        build time:      2024-06-04T15:41:30Z
        commit hash:     3aaa7f34
        commit time:     2024-03-27T16:24:00Z
        go version:      go1.21.8 linux/amd64
2024-06-05T00:09:54.341933736+08:00 [INFO][WEBUI][CFG] Read config from [./confi
g/webuicfg.yaml]
2024-06-05T00:09:54.342103037+08:00 [INFO][WEBUI][Main] Log enable is set to [tr
ue]
2024-06-05T00:09:54.342107568+08:00 [INFO][WEBUI][Main] Log level is set to [inf
o]
2024-06-05T00:09:54.342110546+08:00 [INFO][WEBUI][Main] Report Caller is set to
[false]
2024-06-05T00:09:54.342165122+08:00 [INFO][WEBUI][Init] Server started
[GIN-debug] [WARNING] Running in "debug" mode. Switch to "release" mode in produ
ction.
 - using env:   export GIN_MODE=release
 - using code:  gin.SetMode(gin.ReleaseMode)
```

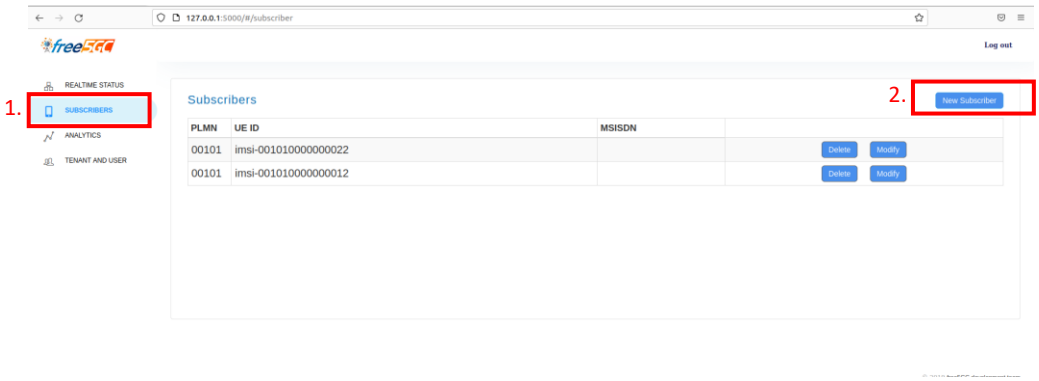
開啟 free5gcUI介面

從主機打開 Web 瀏覽器，然後輸入URL:5000http://127.0.0.1:5000

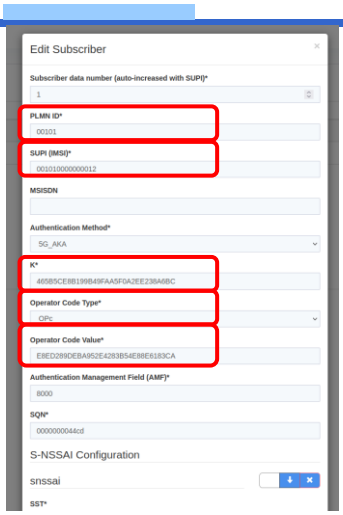


帳號:admin
密碼:free5gc

設定 USER_DB資料



設定 USER_DB資料

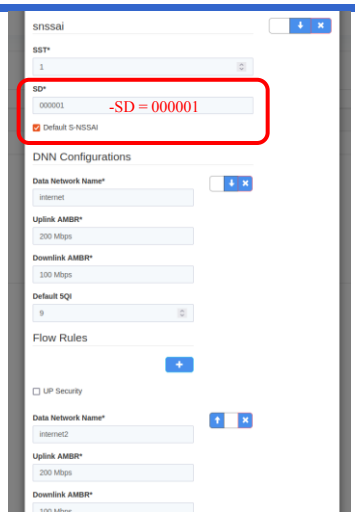


-PLMN ID = 00101

-IMSI = 001010000000012

-K = 465B5CE8B199B49FAA5F0A2EE238A6BC

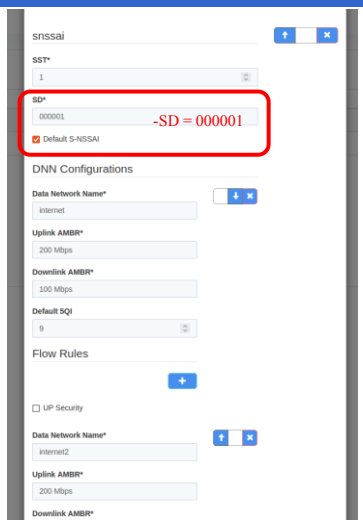
-OPV = E8ED289DEBA952E4283B54E88E6183CA



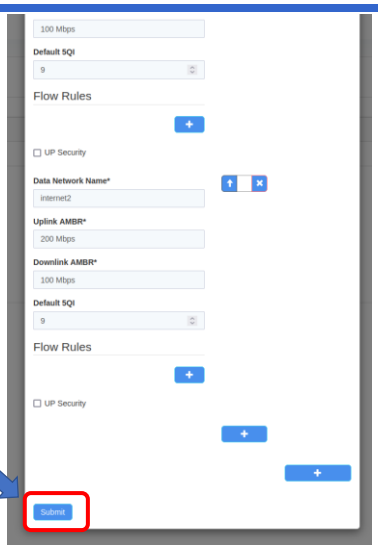
國立虎尾科技大學 NATIONAL FORMOSA UNIVERSITY

53

設定 USER_DB資料



按確認



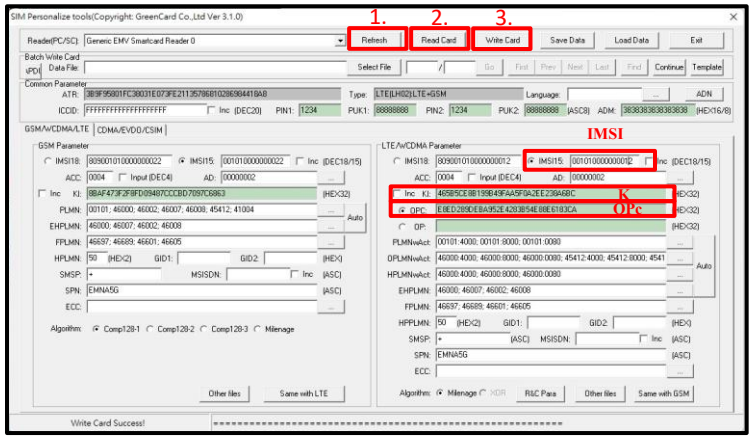
國立虎尾科技大學 NATIONAL FORMOSA UNIVERSITY

54

燒錄USIM

將UE的資料燒進SIM Card

1.尋找讀卡機
2.讀取SIM卡
3.設定完後，再寫入!!!

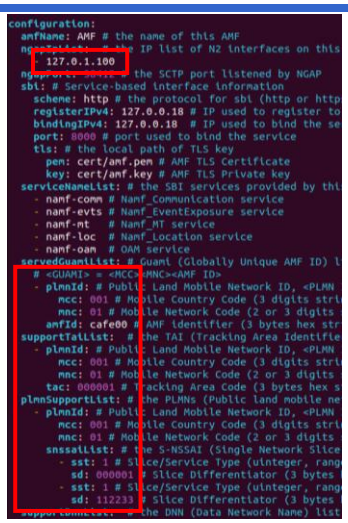


設定核網

- cd ~/free5gc/config
- sudo vim amfcfg.yaml

```
nfu5g@nfu5g-ASUS-EXPERTCENTER-D900MC-M900SC: $ cd free5gc/config
nfu5g@nfu5g-ASUS-EXPERTCENTER-D900MC-M900SC:~/free5gc/config$ sudo vim amfcfg.yaml
```

進入vim依照圖片紅框設定編輯amf設定值



設定核網

- cd ~/free5gc/config
- sudo vim auscfg.yaml

```
nfu5g@nfu5g-ASUS-EXPERTCENTER-D900MC-M900SC:~/free5gc/config$ sudo vim ausfcfg.yaml
```

進入vim依照圖片紅框設定編輯aus設定值

```
info:
  version: 1.0.3
  description: AUSF initial local configuratio

configuration:
  sbi: # Service-based interface information
    scheme: http # the protocol for sbi (http
    registerIPv4: 127.0.0.9 # IP used to regis
    bindingIPv4: 127.0.0.9  # IP used to bind
    port: 8000 # Port used to bind the service
    tls: # the local path of TLS key
      pem: cert/ausf.pem # AUSF TLS Certificat
      key: cert/ausf.key # AUSF TLS Private ke
  serviceNameList: # the SBI services provided
    - nausf-auth # Nausf_UEAuthentication serv
  nrfUri: http://127.0.0.10:8000 # a valid URI
  plmnSupportList: # the PLMNs (Public Land Mo
    - mcc: 001 # Mobile Country Code (3 digits
      mnc: 01  # Mobile Network Code (2 or 3 d
    - mcc: 123 # Mobile Country Code (3 digits
      mnc: 45  # Mobile Network Code (2 or 3 d
  groupId: ausfGroup001 # ID for the group of
  eapAkaSupiImsiPrefix: false # including "ims

logger: # log output setting
  enable: true # true or false
  level: info # how detailed to output, value:
  reportCaller: false # enable the caller repo
~
```


設定核網

- cd ~/free5gc/config
- sudo vim nrfcfg.yaml

```
nfu5g@nfu5g-ASUS-EXPERTCENTER-D900MC-M900SC:~/free5gc/config$ sudo vim nrfcfg.yaml
```

進入vim依照圖片紅框設定編輯nrf設定值

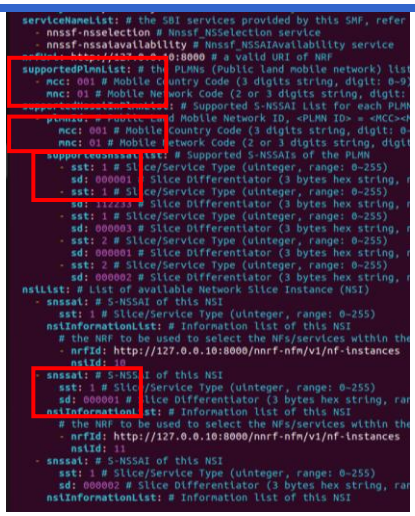
```
info:
  version: 1.0.2
  description: NRF initial local configuration

configuration:
  MongoDBName: free5gc # database name in MongoDB
  MongoDBUrl: mongodb://127.0.0.1:27017 # a valid
  sbi: # Service-based interface information
    scheme: http # the protocol for sbi (http or
    registerIPv4: 127.0.0.10 # IP used to serve N
    bindingIPv4: 127.0.0.10  # IP used to bind th
    port: 8000 # port used to bind the service
    tls: # the local path of TLS key
      pem: cert/nrf.pem # NRF TLS Certificate
      key: cert/nrf.key # NRF TLS Private key
  DefaultPlmnId:
    mcc: 001 # Mobile Country Code (3 digits stri
    mnc: 01 # Mobile Network Code (2 or 3 digits
  serviceNameList: # the SBI services provided by
    - nnrf-nfm # Nnrf_NFManagement service
    - nnrf-disc # Nnrf_NFDiscovery service

logger: # log output setting
  enable: true # true or false
  level: info # how detailed to output, value: tr
  reportCaller: false # enable the caller report
~
~
```

設定核網

- cd ~/free5gc/config
- sudo vim nssfcfg.yaml

```
nfu5g@nfu5g-ASUS-EXPERTCENTER-D900MC-M900SC:~/free5gc/config$ sudo vim nssfcfg.yaml
```

進入vim依照圖片紅框設定編輯nssf設定值



設定核網

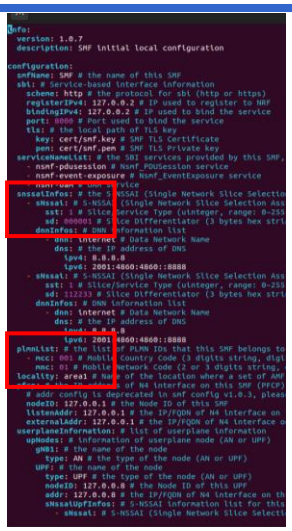
- cd ~/free5gc/config
- sudo vim smfcfg.yaml

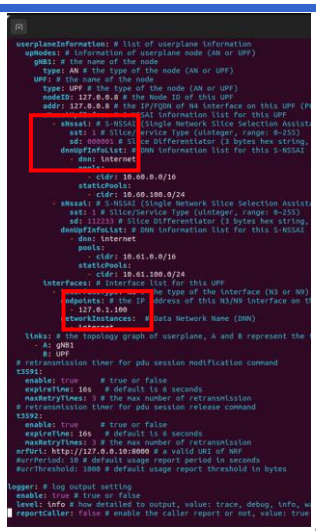
```
nfu5g@nfu5g-ASUS-EXPERTCENTER-D900MC-M900SC:~/free5gc/config$ sudo vim smfcfg.yaml
```

進入vim依照圖片紅框設定編輯smf設定值(下一頁)

設定核網

進入vim依照圖片紅框設定編輯smf設定值





61

設定核網

- cd ~/free5gc/config
- sudo vim upfcfg.yaml

```
nfu5g@nfu5g-ASUS-EXPERTCENTER-D900MC-M900SC:~/free5gc/config$ sudo vim upfcfg.yaml
```

進入vim依照圖片紅框設定編輯upf設定值(下一頁)

```
version: 1.0.3
description: UPF initial local configuration

# The listen IP and nodeID of the N4 interface on t
pfcp:
  addr: 127.0.0.8   # IP addr for listening
  nodeID: 127.0.0.8 # External IP or FQDN can be re
  retransTimeout: 1s # retransmission timeout
  maxRetrans: 3 # the max number of retransmission

gtpu:
  forwarder: gtp5g
  # The IP list of the N3/N9 interfaces on this UPF
  # If there are multiple connection, set addr to 0
  ifList:
    - addr: 127.0.1.100
      type: N3
      # name: upf.5gc.nctu.me
      # ifname: gtpif
      # mtu: 1400

# The DNN list supported by UPF
dnnList:
  - dnn: internet # Data Network Name
    cidr: 10.60.0.0/24 # Classless Inter-Domain Rou
    # natifname: eth0

logger: # log output setting
  enable: true # true or false
  level: info # how detailed to output, value: trac
  reportCaller: false # enable the caller report or
```

62

創建核網腳本

- cd ~/free5gc
- sudo vim 001.sh

```
nfu5g@nfu5g-ASUS-EXPERTCENTER-D900MC-M900SC:~$ cd free5gc
nfu5g@nfu5g-ASUS-EXPERTCENTER-D900MC-M900SC:~/free5gc$ sudo vim 001.sh
```


創建核網腳本

```
#!/usr/bin/env bash

PID_LIST=()

NF_LIST="nrf amf smf udr pcf udm nssf ausf chf"

export GIN_MODE=release

for NF in ${NF_LIST}; do
    ./bin/${NF} &
    PID_LIST+=($!)
    sleep 1
done

function terminate()
{
    sudo kill -SIGTERM ${PID_LIST[${#PID_LIST[@]}-2]} ${PID_LIST[${#PID_LIST[@]}-1]}
    sleep 2
}

trap terminate SIGINT
wait ${PID_LIST}
```

```
#!/usr/bin/env bash
PID_LIST=()
NF_LIST="nrf amf smf udr pcf udm nssf ausf chf"
export GIN_MODE=release
for NF in ${NF_LIST}; do
  ./bin/${NF} &
  PID_LIST+=($!)
  sleep 1
done
function terminate()
{
  sudo kill -SIGTERM ${PID_LIST[${#PID_LIST[@]}-2]} ${PID_LIST[${#PID_LIST[@]}-1]}
  sleep 2
}
trap terminate SIGINT
wait ${PID_LIST}
```

修改完輸入以下程式來給予權限

- sudo chmod +x ./001.sh

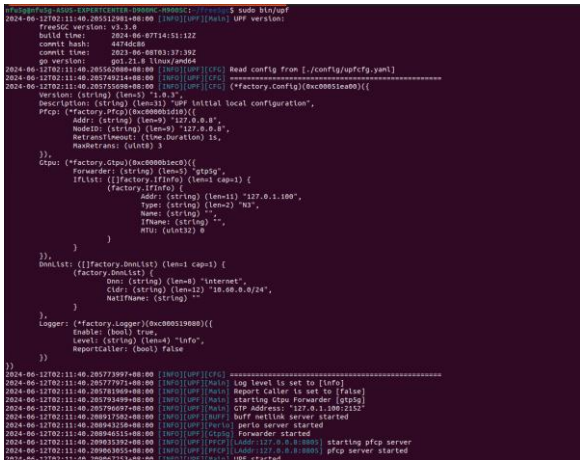
手機設定

插入SIM卡後
手機進入設定>網路和網際網路>網際網路>SIM旁的齒輪>存取點名稱>右上角+>設定完後點右上角圖案(三個點) 選儲存

如右圖設定APN、MCC、MNC

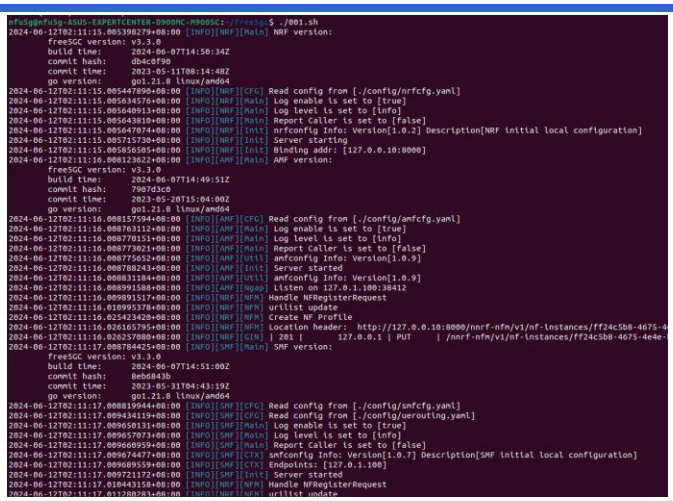
開啟UPF

- cd free5gc
- sudo bin/upf



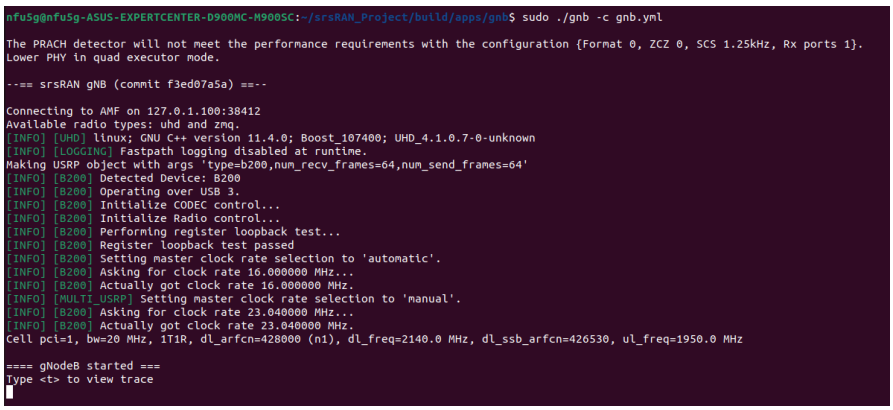
開啟核網

- cd free5gc
- ./001.sh



開啟 gNB

- cd srsRAN_Project/build/apps/gnb
- sudo ./gnb -c gnb.yml



USRP設置

手機連線

1.開飛航再關閉
2.點自動選取網路
3.選 Test PLMN 1-1(00101)(沒看到就重複上面步驟)
4.開啟漫遊

查看手機是否能上網

71

國立虎尾科技大學 NATIONAL FORMOSA UNIVERSITY

B5G/6G數位分身網路跨層整合實作平台

實驗模組二：數位分身網路平台建置

國立虎尾科技大學 電機工程系
蘇暉凱 教授、鄭佳炘 教授

2024年7月

實驗目的

1. 架設開源數位分身平台Open Twin
2. 建立數位分身範例

Opentwins

OpenTwins 是一個專門用來創建數位分身體（Digital Twins）的開源框架。其主要目的是透過模擬物理世界中的實體來幫助用戶進行數據收集、分析和預測。數位分身體技術可以應用在許多領域，如智慧城市、工業製造、智慧建築等，透過虛擬模型來優化實際操作、減少成本和提高效率。

核心功能包括：

1.數據整合：支持多種數據來源和設備。

2.實時監控：提供實時數據處理和分析能力。

3.模型構建：允許用戶構建和管理數字孿生體模型。

4.開放性：開源並且支持擴展，社區活躍。

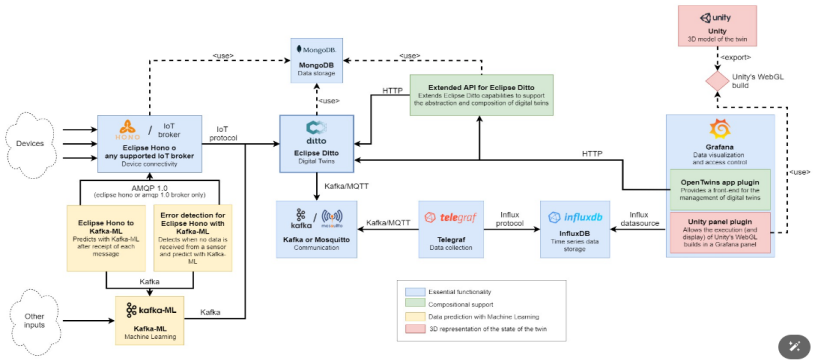
軟硬體環境 – 硬體

名稱	規格	數量	目的
PC	建議CPU:6核、16G RAM、64GB ROM及以上	1	建置Open Twin平台

軟硬體環境 – 軟體

名稱	軟體	版本
PC	OS : Ubuntu	Ubuntu 22.04.1
	Docker	v26.1.4
	Kubernetes	v1.29.6
	Helm v3	v3.15.2
	Minikube	v1.33.1

Opentwins



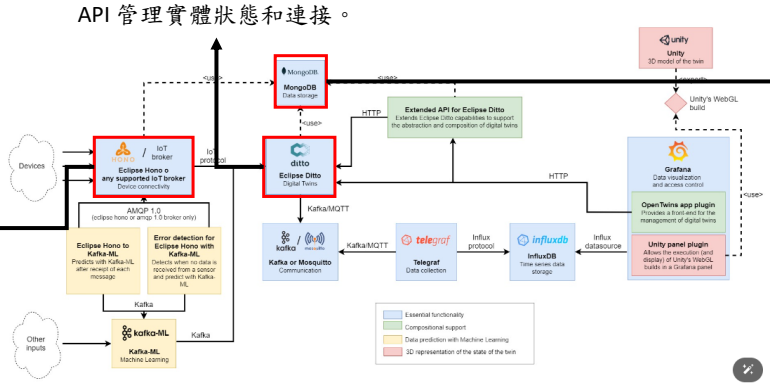
藍色部分為OpenTwins 的核心，提供了數位分身開發平台的基本功能：數位分身的定義、與物聯網設備的連接、資訊儲存以及用戶友好的數據視覺化。

國立虎尾科技大學 NATIONAL FORMOSA UNIVERSITY

Opentwins

Eclipse Ditto：OpenTwins 的核心組件，用於創建和管理數字孿生體。透過 JSON 架構描述實體，並透過 API 管理實體狀態和連接。

Eclipse Hono：支援多種物聯網協議，集中接收數據，直接連接到 Eclipse Ditto。



MongoDB：存儲數字孿生體的狀態、策略、連接信息及近期事件。

國立虎尾科技大學 NATIONAL FORMOSA UNIVERSITY

安裝VirtualBox7.0

- https://www.virtualbox.org/wiki/Downloads

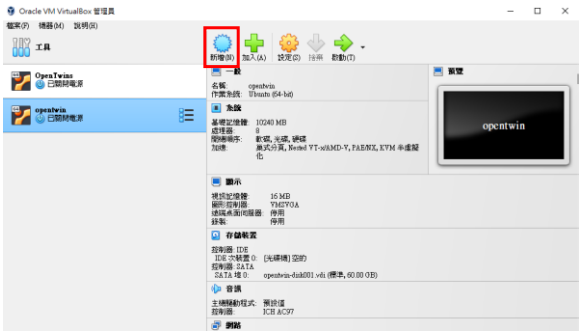
安裝VirtualBox7.0

- 下載完後開始安裝
- 基本上一直按下一步就好

安裝虛擬機器軟體

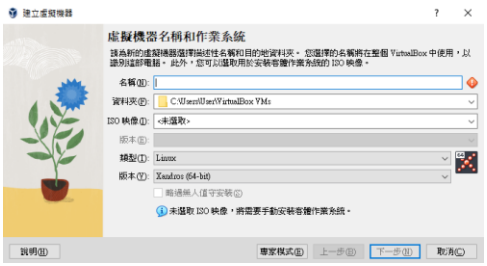
Oracle VM VirtualBox環境設定，下載完Oracle VM VirtualBox後，打開應用程式設定基本環境設定名稱、類型及版本

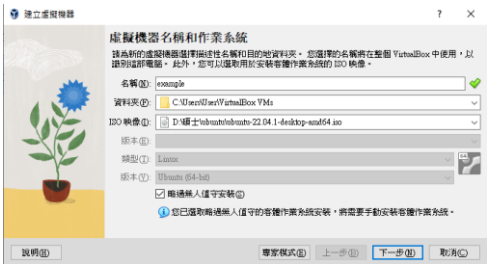
Oracle VM VirtualBox



安裝虛擬機器軟體

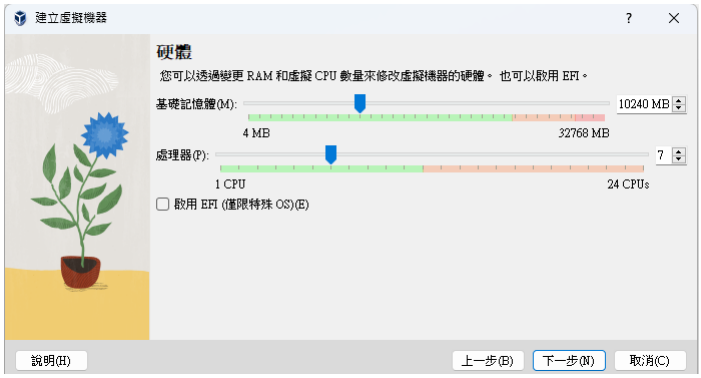
建立名稱、選擇ISO檔、勾選略過無人值守安裝





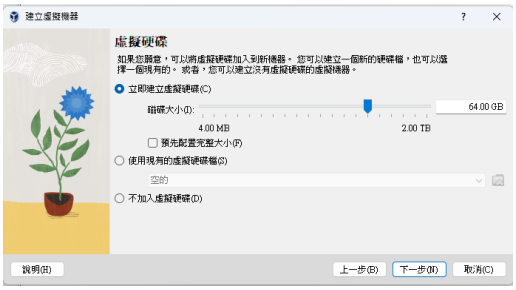
安裝虛擬機器軟體

輸入記憶體及處理器



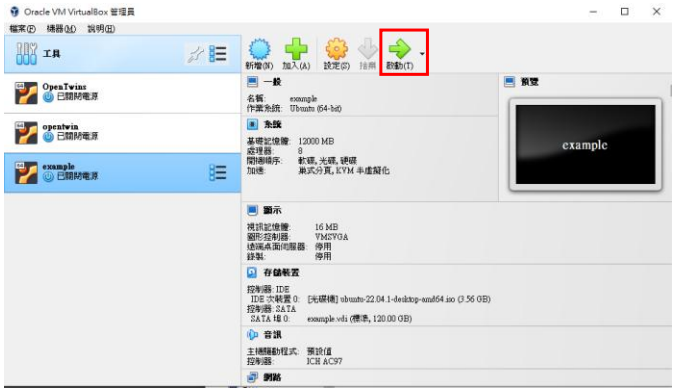
安裝虛擬機器軟體

建立虛擬硬碟



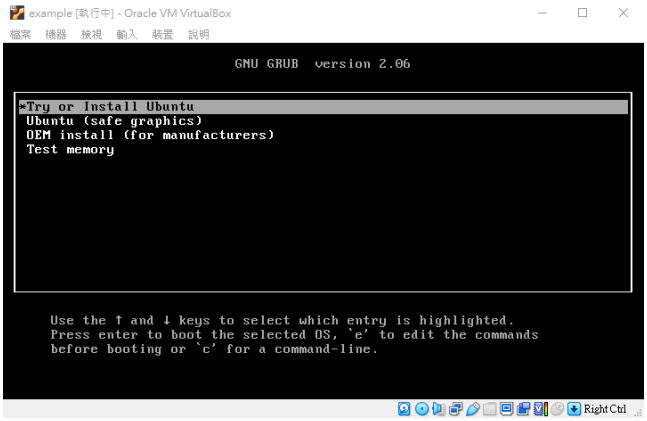
安裝虛擬機器軟體

啟動虛擬環境



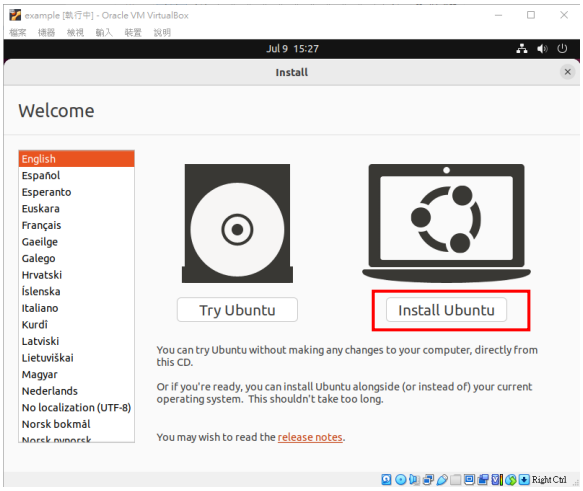
安裝虛擬機器軟體

選擇Try or Install Ubuntu



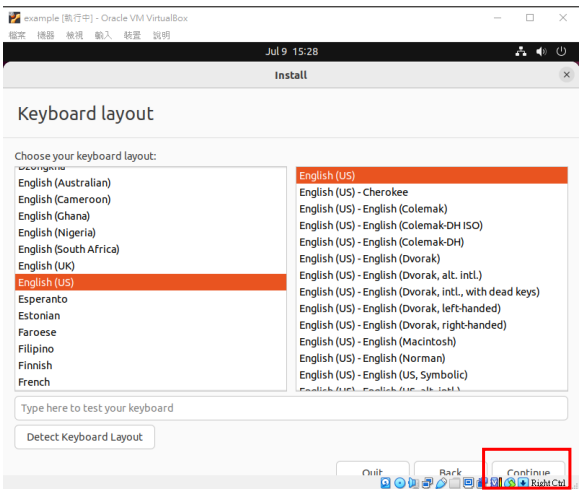
安裝虛擬機器軟體

左側選擇英文 按下Install Ubuntu



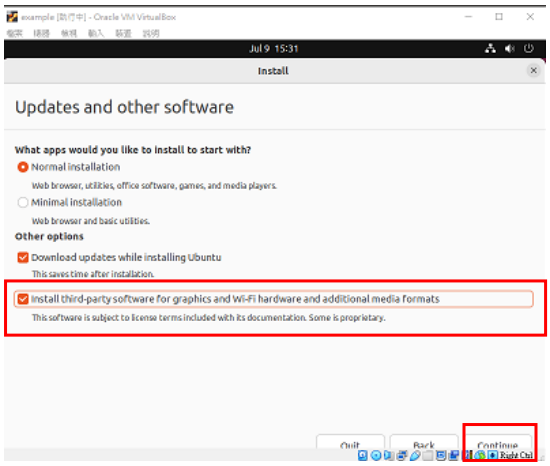
安裝虛擬機器軟體

選擇continue



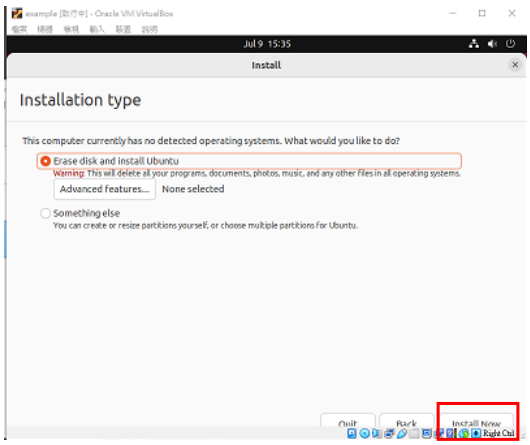
安裝虛擬機器軟體

選擇下方選項後按Continue



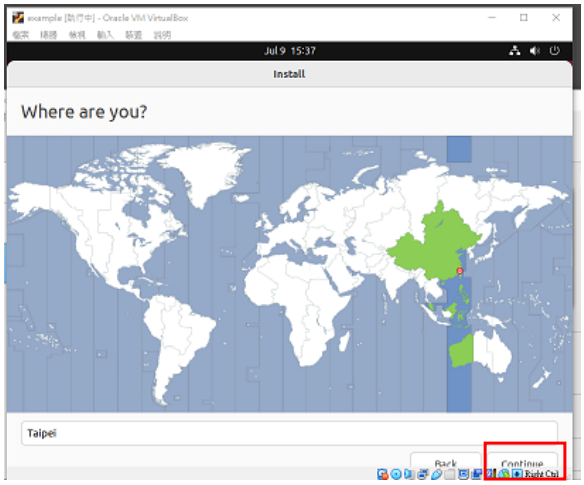
安裝虛擬機器軟體

選擇Install Now



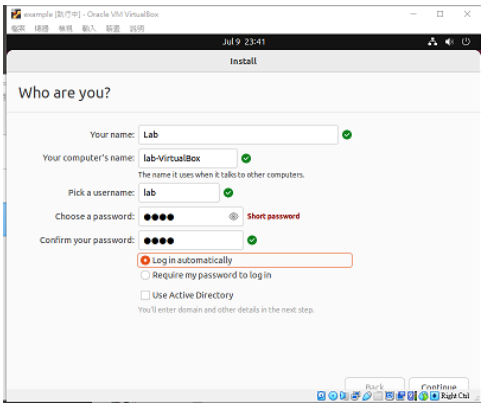
安裝虛擬機器軟體

選擇Install Now



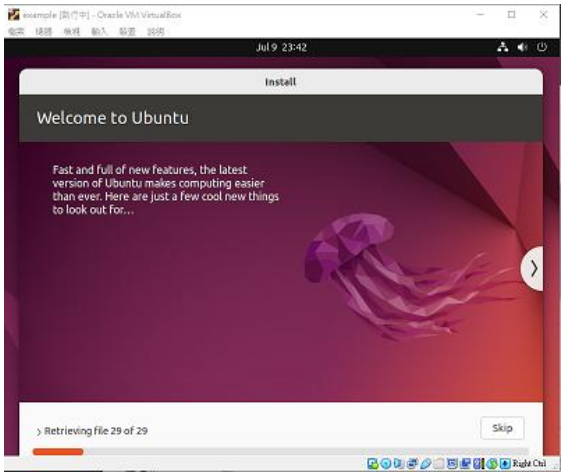
安裝虛擬機器軟體

輸入名稱密碼(可自訂)、選擇自動登入後Continue



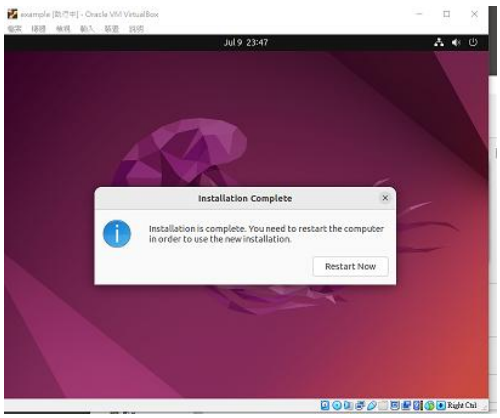
安裝虛擬機器軟體

隨即進入安裝階段



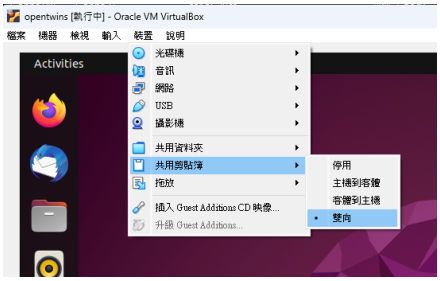
安裝虛擬機器軟體

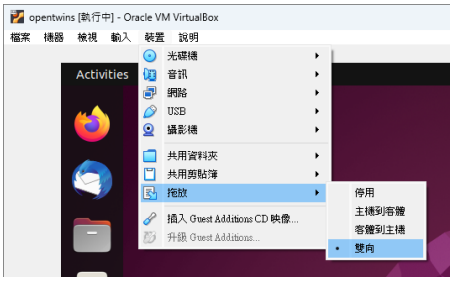
Restart Now



設定虛擬機

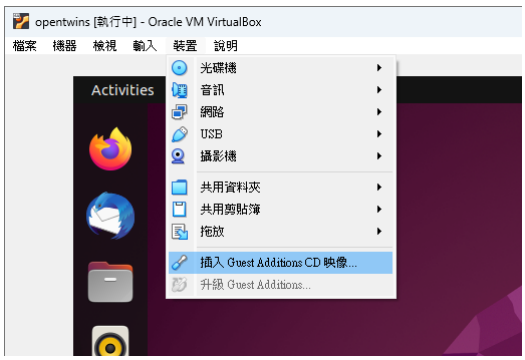
- 從新開機後，將點選裝置將剪貼簿及托放改為雙向





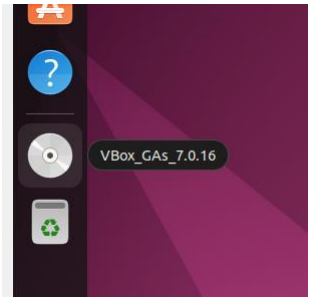
設定虛擬機

- 點選裝置，點擊插入CD映像



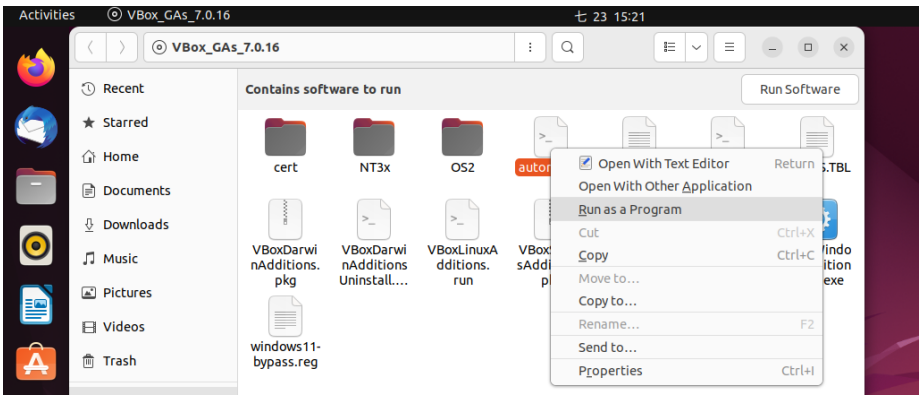
設定虛擬機

- 點選此光碟



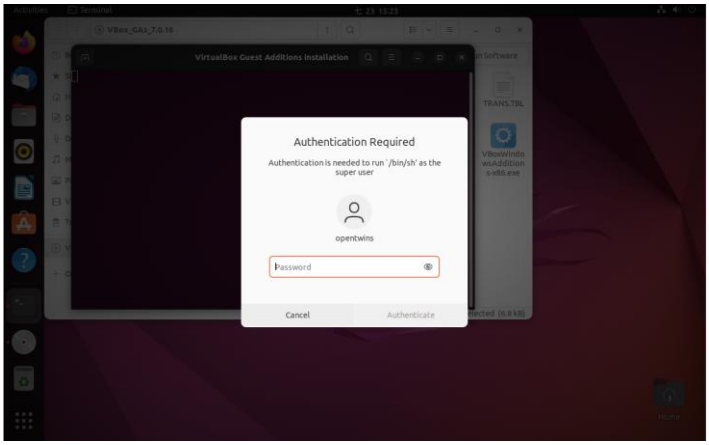
設定虛擬機

- 打開後找到autorun.sh，按右鍵點擊Run as a Program



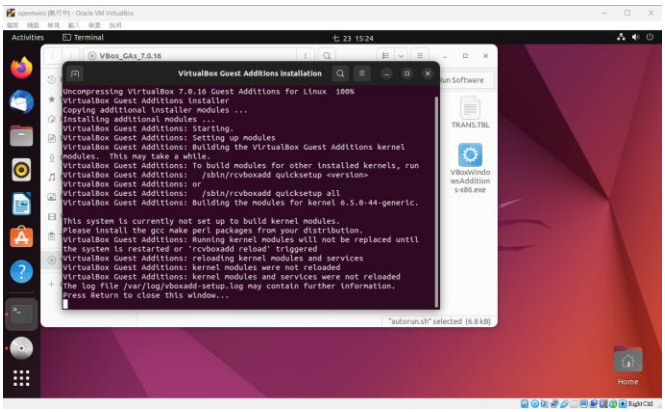
設定虛擬機

- 輸入密碼



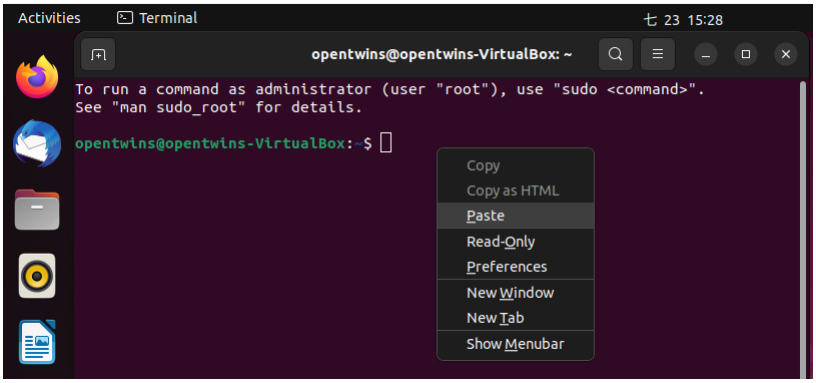
設定虛擬機

- 看到Press Return to close this window… 後，關閉視窗並將虛擬機重新開機



設定虛擬機

- 重新開機後，此時就可以啟用電腦與虛擬機的雙向黏貼了



更新系統

因為Ubuntu系統的更新較為頻繁，操作實驗前先按”ctrl＋alt＋T” 以開啟Terminal視窗，然後輸入sudo apt update&&sudo apt upgrade 並輸入密碼來升級套件以避免後續安裝產生套件不支援的錯誤

```
opentwin@opentwin-VirtualBox:~$ sudo apt-get update && sudo apt-get upgrade
Hit:1 http://tw.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://tw.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:4 http://tw.archive.ubuntu.com/ubuntu jammy-backports InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi i965-va-driver
  intel-media-va-driver libaacs0 libaom3 libass9 libavcodec58 libavformat58
  libavutil56 libbdplus0 libblas3 libbluray2 libbs2b0 libchromaprint1
  libcodec2-1.0 libdav1d5 libflashrom1 libflite1 libftdi1-2 libgme0 libgsm1
  libgstreamer-plugins-bad1.0-0 libigdgmm12 liblilv-0-0 libllvm13 libmfx1
  libmysofa1 libnorm1 libopenmpt0 libpgm-5.3-0 libpostproc55 librabbitmq4
  librubberband2 libserd-0-0 libshine3 libsnappy1v5 libsord-0-0 libsratom-0-0
  libsrt1.4-gnutls libssh-gcrypt-4 libswresample3 libswscale5 libudfread0
  libva-drm2 libva-wayland2 libva-x11-2 libva2 libvdpau1 libvidstab1.1
  libx265-199 libxvidcore4 libzimg2 libzmq5 libzvbi-common libzvbi0
  mesa-va-drivers mesa-vdpau-drivers pocketsphinx-en-us va-driver-all
  vdpau-driver-all
Use 'sudo apt autoremove' to remove them.
The following packages have been kept back:
  gir1.2-javascriptcoregtk-4.0 gir1.2-webkit2-4.0 libjavascriptcoregtk-4.0-18
  libwebkit2gtk-4.0-37 python3-software-properties python3-update-manager
```

更新套件索引

sudo apt-get update

```
opentwin@opentwin-VirtualBox:~$ sudo apt-get update
Hit:1 http://tw.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://tw.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://tw.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Reading package lists... Done
```


安裝所需的套件

sudo apt-get install \
ca-certificates \
curl \
gnupg \
lsb-release

```
opentwin@opentwin-VirtualBox:~$ sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
lsb-release is already the newest version (11.1.0ubuntu4).
lsb-release set to manually installed.
ca-certificates is already the newest version (20230311ubuntu0.22.04.1).
ca-certificates set to manually installed.
gnupg is already the newest version (2.2.27-3ubuntu2.1).
gnupg set to manually installed.
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi i965-va-driver intel-media-va-driver libaacs0 libaom3 libass9 libavcodec58
  libavformat58 libavutil56 libbdplus0 libblas3 libbluray2 libbs2b0 libchromaprint1 libcodec2-1.0 libdav1d5 libflashrom1 libflite1
  libftdi1-2 libgme0 libgsm1 libgstreamer-plugins-bad1.0-0 libigdgmm12 liblilv-0-0 libllvm13 libmfx1 libmysofa1 libnorm1 libopenmpt0
  libpgm-5.3-0 libpostproc55 librabbitmq4 librubberband2 libserd-0-0 libshine3 libsnappy1v5 libsord-0-0 libsratom-0-0 libsrt1.4-gnutls
  libssh-gcrypt-4 libswresample3 libswscale5 libudfread0 libva-drm2 libva-wayland2 libva-x11-2 libva2 libvdpau1 libvidstab1.1 libx265-199
```

添加 Docker 的官方 GPG 密鑰

sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

```
opentwin@opentwin-VirtualBox:~$ sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```


設置 Docker 的穩定版儲存庫

echo \
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

```
opentwin@opentwin-VirtualBox:~$ echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

更新套件索引並安裝 Docker引擎

- sudo apt-get update
- sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

```
opentwin@opentwin-VirtualBox:~$ sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
Hit:1 http://tw.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://tw.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://tw.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:4 https://download.docker.com/linux/ubuntu jammy InRelease [48.8 kB]
Get:5 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [35.6 kB]
Hit:6 http://security.ubuntu.com/ubuntu jammy-security InRelease
Fetched 84.4 kB in 5s (15.4 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi i965-va-driver intel-media-va-driver libaacs0 libaom3 libass9 libavcodec58
  libavformat58 libavutil56 libbdplus0 libblas3 libbluray2 libbs2b0 libchromaprint1 libcodec2-1.0 libdav1d5 libflashrom1 libflite1
  libftdi1-2 libgme0 libgsm1 libgstreamer-plugins-bad1.0-0 libigdgmm12 liblilv-0-0 libllvm13 libmfx1 libmysofa1 libnorm1 libopenmpt0
  libpgm-5.3-0 libpostproc55 librabbitmq4 librubberband2 libserd-0-0 libshine3 libsnappy1v5 libsord-0-0 libsratom-0-0 libsrt1.4-gnutls
  libssh-gcrypt-4 libswresample3 libswscale5 libudfread0 libva-drm2 libva-wayland2 libva-x11-2 libva2 libvdpau1 libvidstab1.1 libx265-199
  libxvidcore4 libzimg2 libzmq5 libzvbi-common libzvbi0 mesa-va-drivers mesa-vdpau-drivers pocketsphinx-en-us va-driver-all
  vdpau-driver-all
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  docker-ce-rootless-extras git git-man liberror-perl libslirp0 pigz slirp4netns
Suggested packages:
```


啟動並驗證 Docker

- sudo systemctl start docker
- sudo systemctl enable docker
- sudo docker run hello-world

```
opentwin@opentwin-VirtualBox:~$ sudo systemctl start docker
sudo systemctl enable docker
sudo docker run hello-world
Synchronizing state of docker.service with SysV service script with /lib/systemd/s
Executing: /lib/systemd/systemd-sysv-install enable docker
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:94323f3e5e09a8b9515d74337010375a456c909543e1ff1538f5116d38ab3989
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

啟動並驗證 Docker

如果看到類似於 "Hello from Docker!" 的信息，則說明 Docker 已經正確安裝並運行。
非 root 用戶運行 Docker
sudo usermod -aG docker $USER

重新重新啟動Docker
newgrp docker

```
opentwin@opentwin-VirtualBox:~$ sudo usermod -aG docker $USER
opentwin@opentwin-VirtualBox:~$ newgrp docker
```


啟動並驗證 Docker

驗證 Docker 設置
docker run hello-world

```
opentwin@opentwin-VirtualBox:~$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

安裝 kubectl

sudo snap install kubectl --classic

```
opentwin@opentwin-VirtualBox:~$ sudo snap install kubectl --classic
kubectl 1.29.6 from Canonical✓ installed
```


確認 kubectl 安裝成功

kubectl version --client

```
opentwin@opentwin-VirtualBox:~$ kubectl version --client
Client Version: v1.29.6
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
```

如果看到 kubectl 的版本號，說明安裝成功

安裝 Helm v3

curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash

```
opentwin@opentwin-VirtualBox:~$ curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 11694  100 11694    0     0  22454      0 --:--:-- --:--:-- --:--:-- 22445
Downloading https://get.helm.sh/helm-v3.15.2-linux-amd64.tar.gz
Verifying checksum... Done.
Preparing to install helm into /usr/local/bin
helm installed into /usr/local/bin/helm
```


確認 Helm 安裝成功

helm version

```
opentwins@opentwins-E500-G9-WS760T:~$ helm version
version.BuildInfo{Version:"v3.15.2", GitCommit:"1a500d5625419a524fdae4b33de351cc4f58ec35", GitTreeSta
te:"clean", GoVersion:"go1.22.4"}
```

如果看到 Helm 的版本號，說明安裝成功

使用 Minikube 設置本地 Kubernetes 集群

下載 Minikube 安裝腳本
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

```
opentwins@opentwins-E500-G9-WS760T:~$ curl -LO https://storage.googleapis.com/minikube/releases/lates
t/minikube-linux-amd64
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  3 91.1M    3 3202k    0     0   536k      0  0:02:54  0:00:05  0:02:49  584k
```


使用 Minikube 設置本地 Kubernetes 集群

安裝 Minikube
sudo install minikube-linux-amd64 /usr/local/bin/minikube
驗證 Minikube 安裝成功
minikube version

```
opentwin@opentwin-VirtualBox:~$ sudo install minikube-linux-amd64 /usr/local/bin/minikube
opentwin@opentwin-VirtualBox:~$ minikube version
minikube version: v1.33.1
commit: 5883c09216182566a63dff4c326a6fc9ed2982ff
opentwin@opentwin-VirtualBox:~$ minikube version
minikube version: v1.33.1
commit: 5883c09216182566a63dff4c326a6fc9ed2982ff
```

設置 Minikube 集群

minikube start --cpus 4 --disk-size 40gb --memory 8192

```
opentwin@opentwin-VirtualBox:~$ minikube start --cpus 4 --disk-size 40gb --memory 8192
😄  minikube v1.33.1 on Ubuntu 22.04 (vbox/amd64)
✨  Automatically selected the docker driver. Other choices: none, ssh
📌  Using Docker driver with root privileges
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🚜  Pulling base image v0.0.44 ...
💾  Downloading Kubernetes v1.30.0 preload ...
    > preloaded-images-k8s-v18-v1...:  342.90 MiB / 342.90 MiB  100.00% 5.25 Mi
    > gcr.io/k8s-minikube/kicbase...:  481.58 MiB / 481.58 MiB  100.00% 6.45 Mi
🔥  Creating docker container (CPUs=4, Memory=8192MB) ...
🐳  Preparing Kubernetes v1.30.0 on Docker 26.1.1 ...
    ▪ Generating certificates and keys ...
    ▪ Booting up control plane ...
    ▪ Configuring RBAC rules ...
🔗  Configuring bridge CNI (Container Networking Interface) ...
🔎  Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Enabled addons: default-storageclass, storage-provisioner
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

這會啟動一個包含 4 個 CPU、40GB硬碟和 8GB 記憶體的本地 Kubernetes 集群，請確保你的硬體規個在這之上

設置 Minikube 集群

設置 kubectl 使用 Minikube 的上下文

kubectl config use-context minikube

```
opentwin@opentwin-VirtualBox:~$ kubectl config use-context minikube
Switched to context "minikube".
```

設置 Minikube 集群

新增 OpenTwins helm 圖表所在的 ERTIS 儲存庫

helm repo add ertis https://ertis-research.github.io/Helm-charts/

```
opentwin@opentwin-VirtualBox:~$ helm repo add ertis https://ertis-research.github.io/Helm-charts/
"ertis" has been added to your repositories
```


設置 Minikube 集群

創建命名空間

kubectl create namespace opentwins

```
opentwin@opentwin-VirtualBox:~$ kubectl create namespace opentwins
namespace/opentwins created
```

執行 Helm 的OpenTwins部屬命令

(可能需要等待久一點時間)
helm upgrade --install opentwins ertis/OpenTwins -n opentwins --wait --dependency-update --timeout 60m

```
opentwins@opentwins-E500-G9-WS760T:~$ helm upgrade --install opentwins ertis/OpenTwins -n opentwins -
-wait --dependency-update --timeout 40m
Release "opentwins" has been upgraded. Happy Helming!
NAME: opentwins
LAST DEPLOYED: Fri Jul  5 17:25:13 2024
NAMESPACE: opentwins
STATUS: deployed
REVISION: 4
```


執行 Helm 的OpenTwins部屬命令

如果 helm upgrade --install 命令完成後沒有報錯，則安裝過程應該是成功的。您可以使用以下命令來檢查 Helm 發行版的狀態
helm list -n opentwins

```
opentwins@opentwins-E500-G9-WS760T:~$ helm list -n opentwins
NAME            NAMESPACE       REVISION        UPDATED                                 STATUS      C
HART            APP VERSION
opentwins       opentwins       4               2024-07-05 17:25:13.993990941 +0800 CST deployed    O
penTwins-0.5.17 0.5.0
```

執行 Helm 的OpenTwins部屬 命令

檢查 opentwins 命名空間中所有 Pod 的狀態，以確保它們都處於 Running 或 Completed 狀態
kubectl get pods -n opentwins

```
opentwin@opentwin-VirtualBox:~$ kubectl get pods -n opentwins
NAME                                        READY   STATUS    RESTARTS        AGE
opentwins-ditto-connectivity-85896d76b6-6sfq7   1/1     Running   0               7m58s
opentwins-ditto-extended-api-65b8586b49-m62lz   1/1     Running   0               7m58s
opentwins-ditto-gateway-86dd95d46c-65ztw        1/1     Running   0               7m58s
opentwins-ditto-nginx-856bbd86b7-pdxfm          1/1     Running   0               7m58s
opentwins-ditto-policies-6fd5d86476-2sccc       1/1     Running   0               7m58s
opentwins-ditto-things-7c559474cd-n4n47         1/1     Running   0               7m58s
opentwins-ditto-thingssearch-8fcb67669-lz267    1/1     Running   0               7m58s
opentwins-grafana-7d4cfbd857-q6g4n              2/2     Running   0               7m58s
opentwins-influxdb2-0                           1/1     Running   0               7m58s
opentwins-mongodb-657dfd5646-sz5vs              1/1     Running   0               7m58s
opentwins-mosquitto-66d8f6955c-f64n9            1/1     Running   0               7m58s
opentwins-telegraf-c6b9bddc-zbsb7               1/1     Running   3 (4m50s ago)   7m58s
```


執行 Helm 的OpenTwins部屬 命令

檢查所有服務的狀態，以確保它們都已正確啟動
kubectl get services -n opentwins

```
opentwin@opentwin-VirtualBox:~$ kubectl get services -n opentwins
NAME                        TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)                       AGE
opentwins-ditto-extended-api   NodePort    10.100.84.155    <none>        8080:30526/TCP                8h
opentwins-ditto-gateway        ClusterIP   10.110.78.45     <none>        8080/TCP                      8h
opentwins-ditto-nginx          NodePort    10.103.42.251    <none>        8080:30525/TCP                8h
opentwins-grafana              NodePort    10.109.126.54    <none>        80:30718/TCP                  8h
opentwins-influxdb2            NodePort    10.103.196.43    <none>        80:30716/TCP                  8h
opentwins-mongodb              NodePort    10.108.214.181   <none>        27017:30717/TCP               8h
opentwins-mosquitto            NodePort    10.101.31.38     <none>        1883:30511/TCP,9001:32463/TCP   8h
```

尋找Minikube IP 地址

minikube ip

```
opentwin@opentwin-VirtualBox:~$ minikube ip
192.168.49.2
```

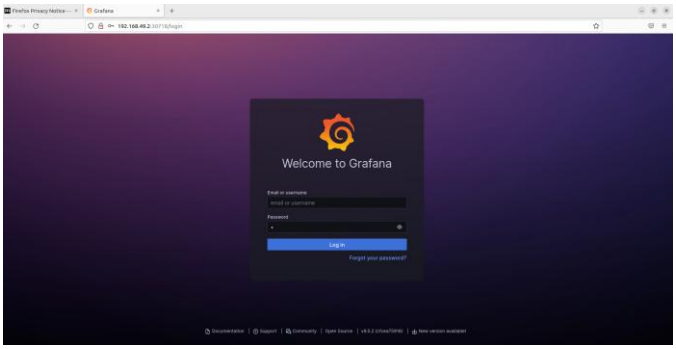
假設你的輸出為192.168.49.2
外部 URL
根據您的服務輸出，這些是每個服務的外部 URL：
Eclipse Ditto 擴充 API
URL: http://192.168.49.2:30526

Eclipse Ditto Nginx
URL: http://192.168.49.2:30525

Grafana
URL: http://192.168.49.2:30718
您可以在瀏覽器中訪問這些 URL 來確認服務是否正常運行。

Grafana介面

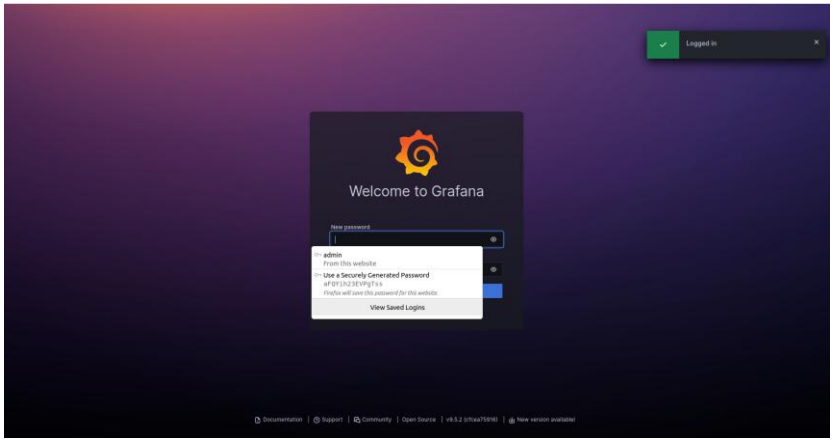
網頁輸入: 192.168.49.2:30718
可以進入Grafana介面



預設為使用者admin和密碼admin

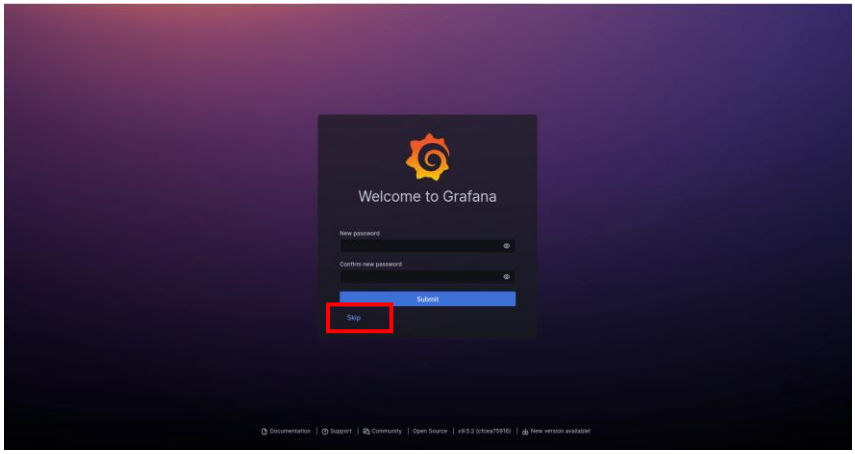
Grafana介面

不用修改密碼，鼠標直接在空白處點一下



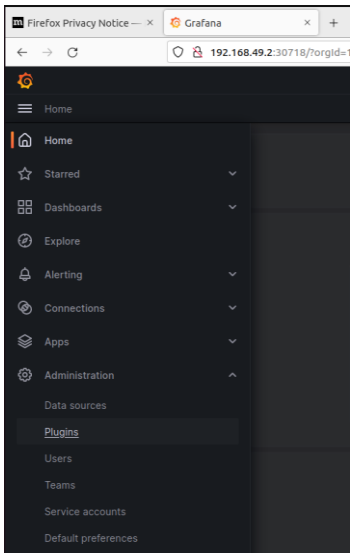
Grafana介面

按下Skip登入



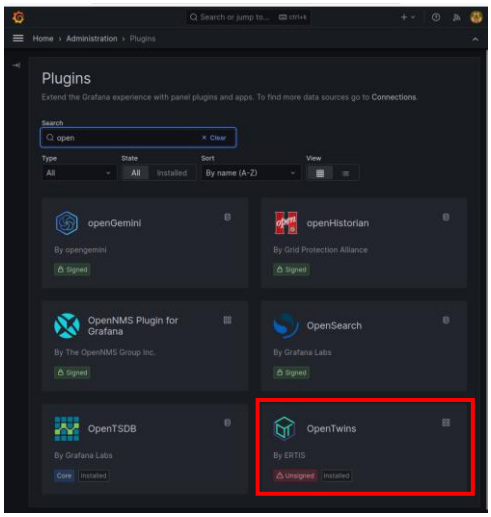
Grafana介面

訪問左側下拉式選單並選擇Administration > Plugins。到達那裡後，找到OpenTwins插件並透過點擊啟用將其啟用。



Grafana介面

搜尋open twins，選取右下角的opentwins



Grafana介面

填入紅框中的資訊:

Eclipse Ditto URL:

http://192.168.49.2:30525

Eclipse Ditto Extended API URL:

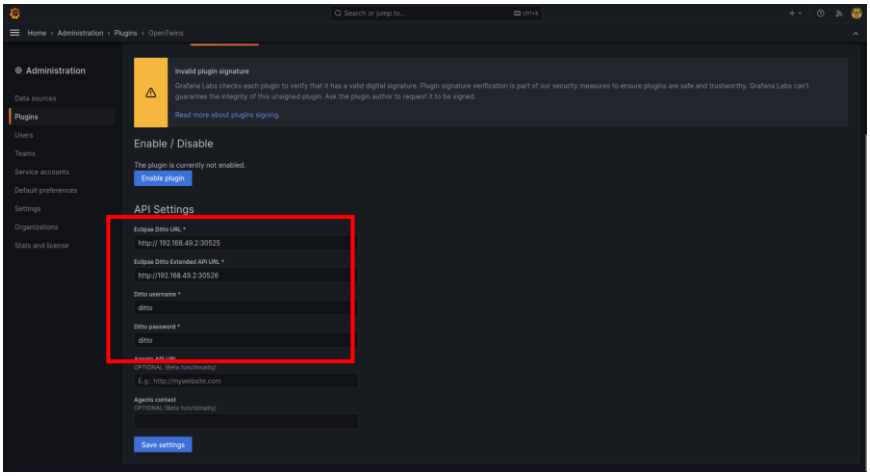
http://192.168.49.2:30526

Ditto username:

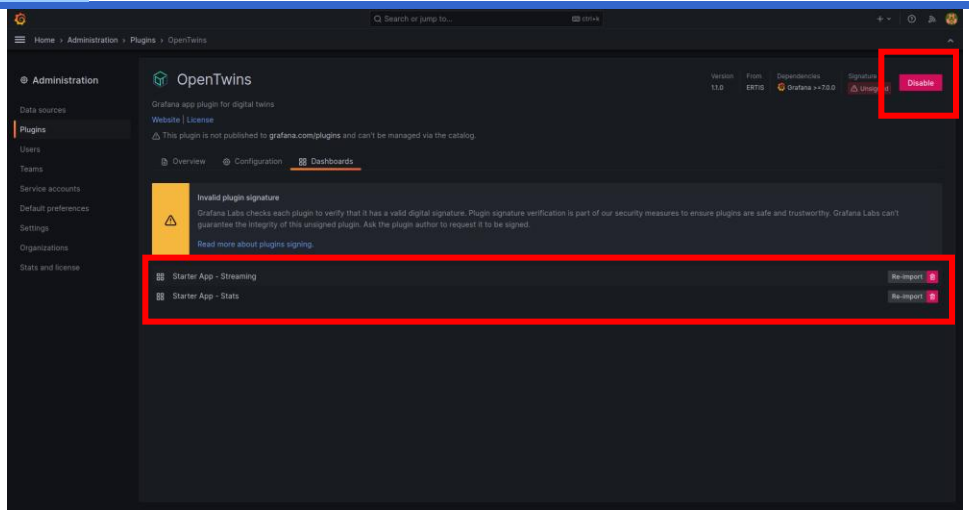
ditto

Ditto passward:

ditto

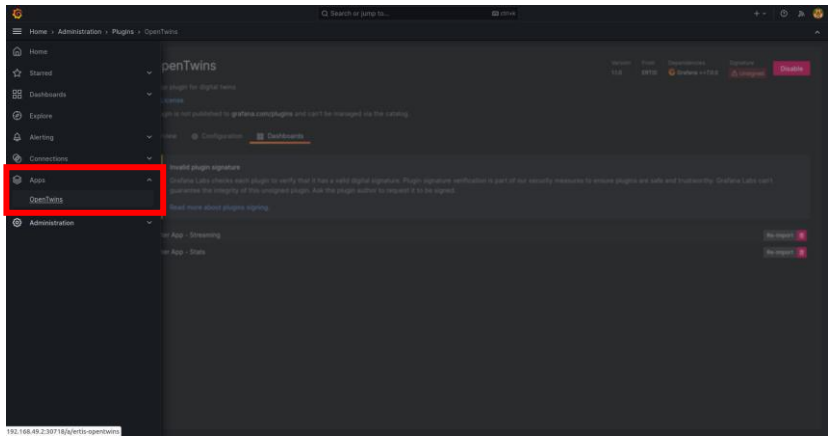


Grafana介面

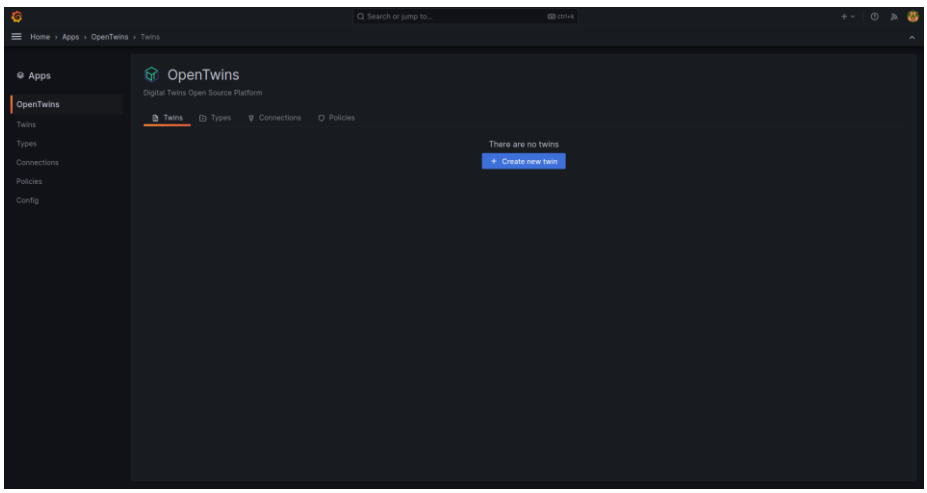


Grafana介面

App > OpenTwins在左側下拉式選單部分找到可用的應用程式。
現在您可以開始使用 OpenTwins

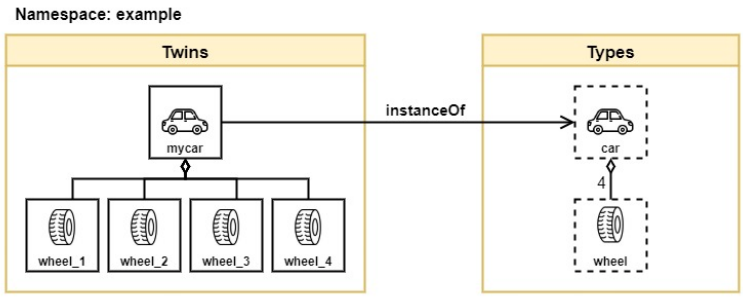


Grafana介面



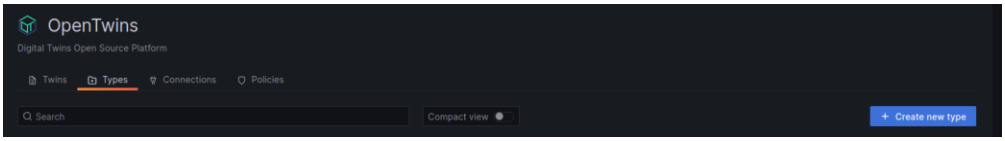
範例應用

利用該平台的功能，我們將創建一個複合數位分身。為此，我們將定義類型“car”和“wheel”，它們將分別抽象化汽車和車輪的資訊。這些類型將透過組合關係連結起來，這意味著汽車由四個輪子組成。一旦所有這些都設定完畢，將汽車實例化為數位雙胞胎將自動為所有四個車輪產生雙胞胎。透過這種方式，我們可以獨立存取每個車輪的數據，並輕鬆地為其他汽車或其他環境添加數位分身。



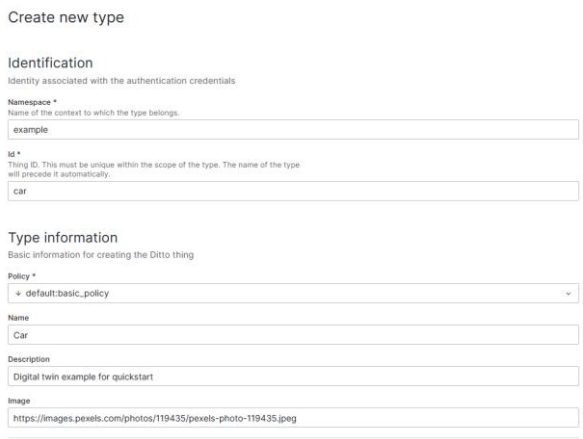
範例應用

進入opentwin介面
進入Type



點擊藍色的 + Create new type

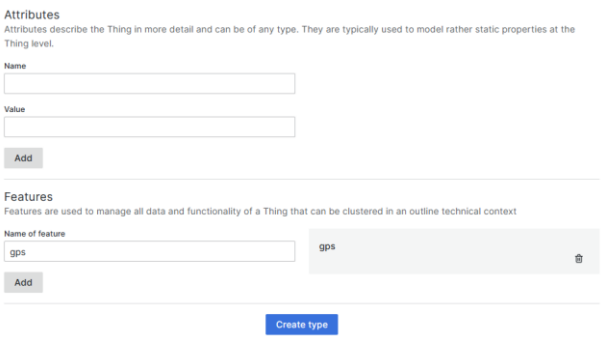
範例應用



example
car
default:basic_policy
Car
Digital twin example for quickstart
https://images.pexels.com/photos/119435/pexels-photo-119435.jpeg

範例應用

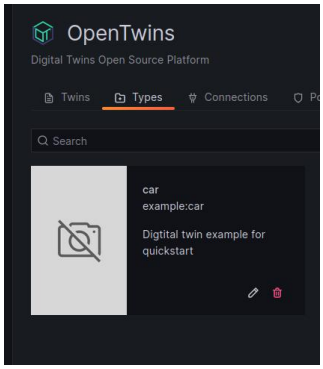
填寫gps，並按Add加入



輸入完後按Create type按鈕

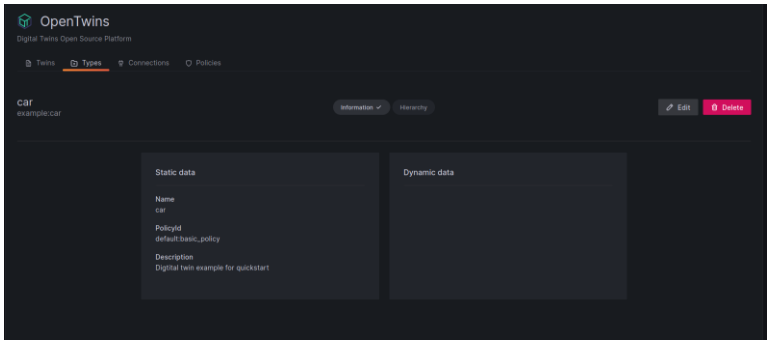
範例應用

回到Type介面



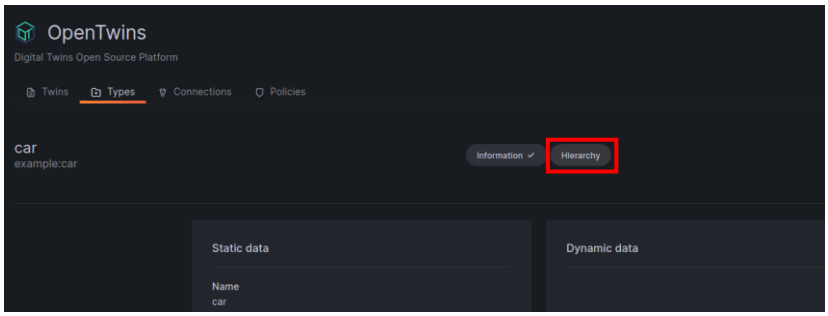
範例應用

點擊並進入剛剛創建的car



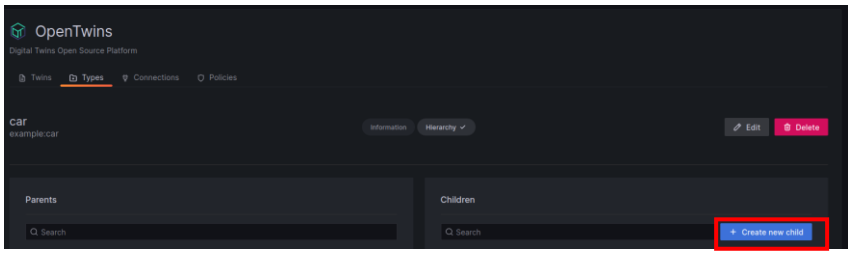
範例應用

點擊Hierarchy



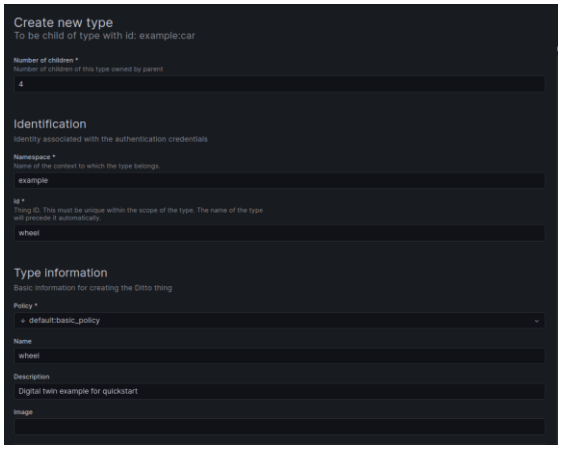
範例應用

進入Hierarchy後 點擊藍色的Create new child 添加子型態(4個輪子)



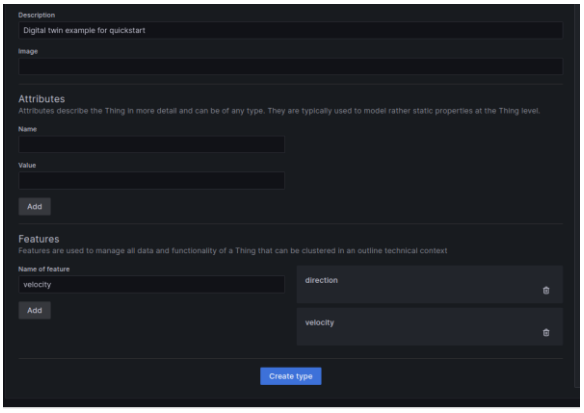
範例應用

輸入以下資訊(其中的4是因為車子有四個輪胎)



範例應用

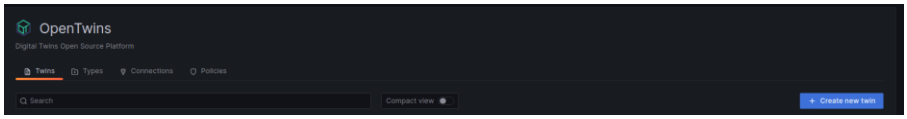
輸入以下資訊(其中的4是因為車子有四個輪胎)



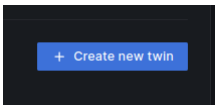
輸入完後按Create type按鈕

範例應用

創建twin:
進入Twin

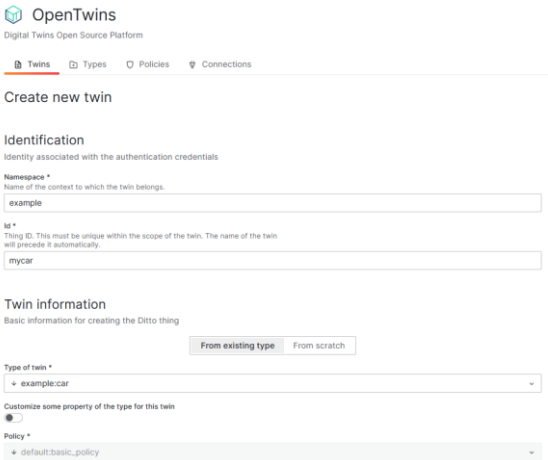


按下藍色Create new Twin

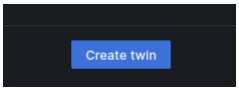


範例應用

按下圖輸入

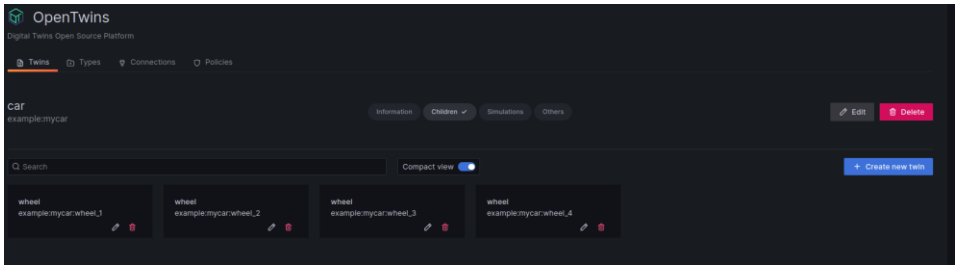


輸入後按下Create twin



範例應用

點擊建立學生後，將自動產生 5 個數位分身。過程完成後，將出現一條成功訊息。如果我們檢查雙胞胎列表，我們將看到我們的雙胞胎範例：汽車。透過點擊它並存取「子項目」選項卡，我們將找到與其輪子相對應的四個雙胞胎，每個雙胞胎都具有各自類型中指定的功能。



範例應用

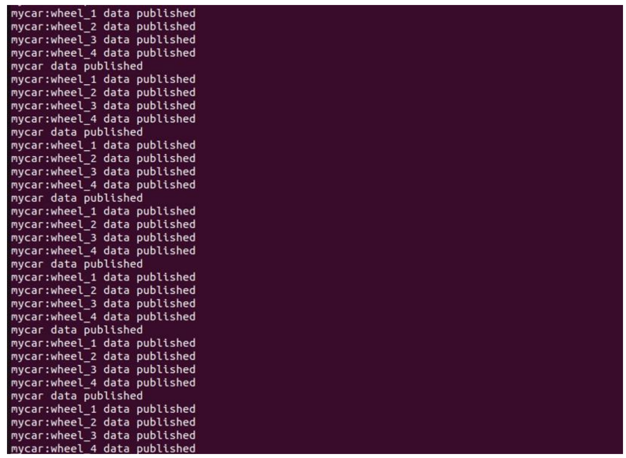
安裝pip:
sudo apt-get install python3-pip

由於我們沒有真實的數據，我們將創建一個 Python 腳本，該腳本每 5 秒從汽車及其車輪生成一次隨機數據，並以 Ditto 協定將其發送到 Mosquitto。 要運行腳本，我們需要安裝 MQTT 的 Paho 庫
pip install paho-mqtt
pip install typing_extensions
sudo apt install build-essential vim

輸入sudo vim car.py
輸入程式如下(在以下文稿中，您必須將 MQTT 代理地址和埠更改為您自己的地址和埠)

範例應用

輸入後開啟程式
python3 car.py
開始發送虛擬數據

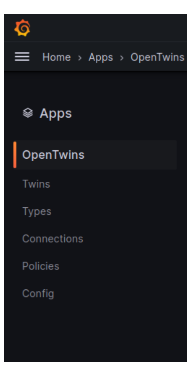


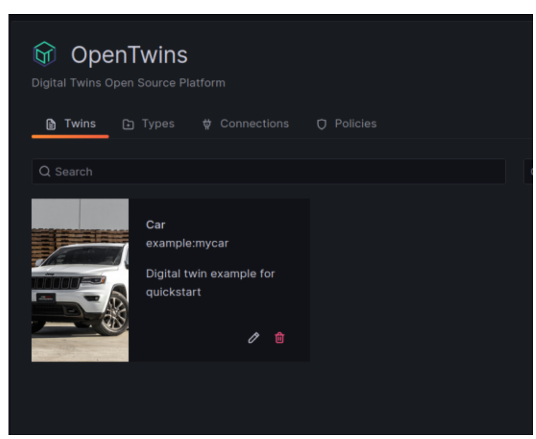
國立虎尾科技大學
NATIONAL FORMOSA UNIVERSITY

149

149

範例應用



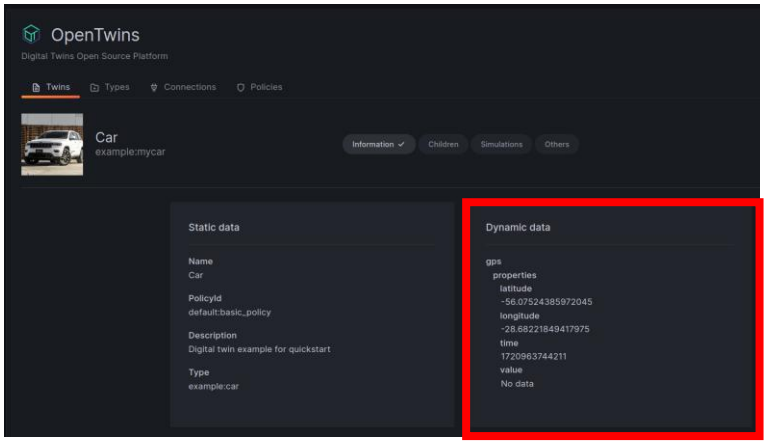


國立虎尾科技大學
NATIONAL FORMOSA UNIVERSITY

150

150

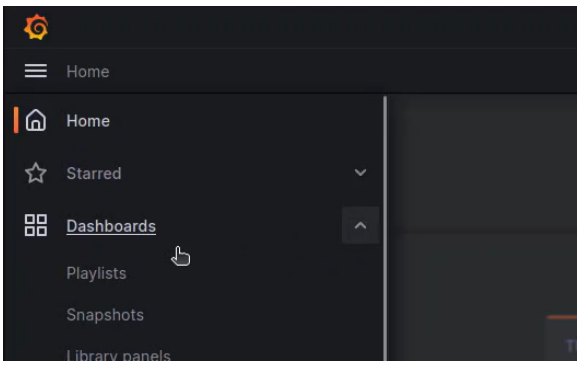
範例應用



範例應用

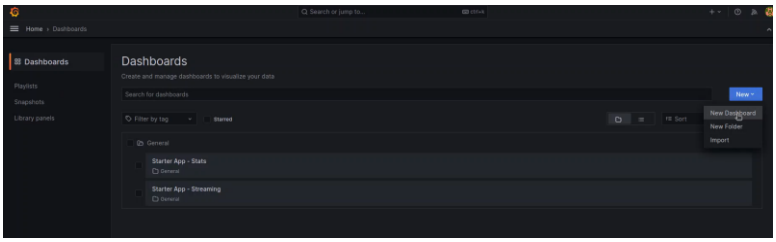
程式開始且opentwin 開始有數值後

在選單中找到Dashboards



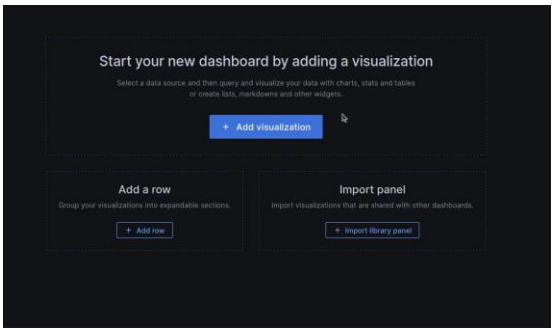
範例應用

進入後點選New 然後點New Dashboard



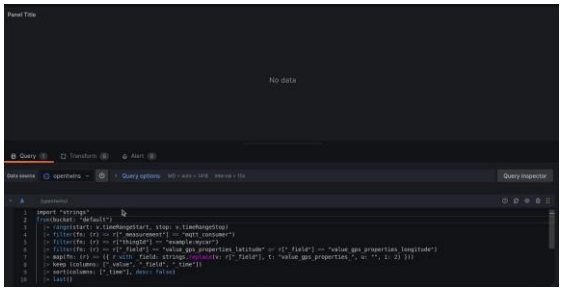
範例應用

點選中間Add visualization



範例應用

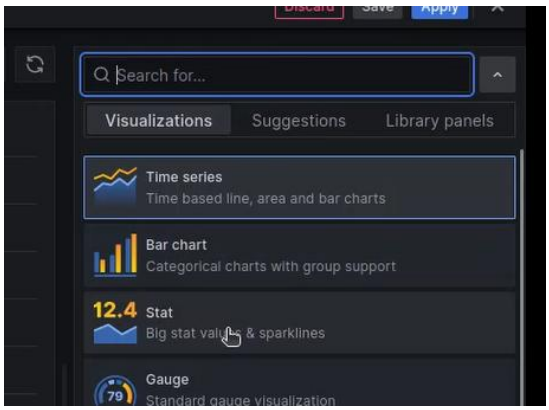
Data source 選擇openywins
然後在下面填入



```
import "strings"
from(bucket: "default")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "mqtt_consumer")
  |> filter(fn: (r) => r["thingId"] == "example:mycar")
  |> filter(fn: (r) => r["_field"] ==
"value_gps_properties_latitude" or r["_field"] ==
"value_gps_properties_longitude")
  |> map(fn: (r) => ({ r with _field: strings.replace(v: r["_field"], t:
"value_gps_properties_", u: "", i: 2) }))
  |> keep (columns: ["_value", "_field", "_time"])
  |> sort(columns: ["_time"], desc: false)
  |> last()
```

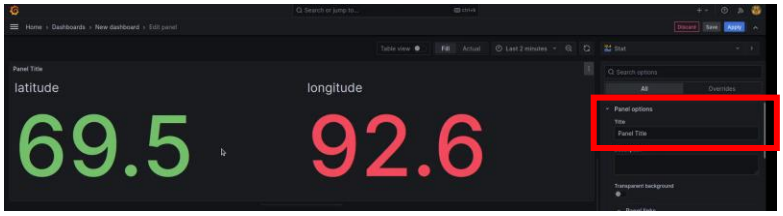

範例應用

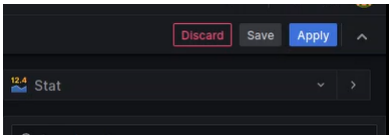
然後在右上角選擇圖表為stat格式



範例應用

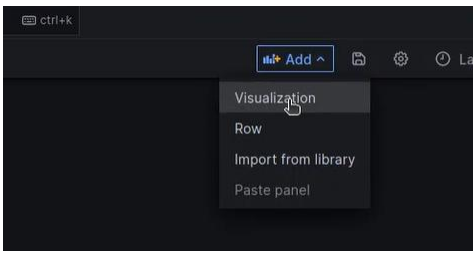
修改title為Current GPS
然後按Apply

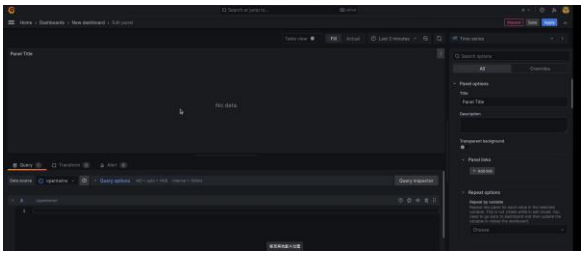




範例應用

添加新表格

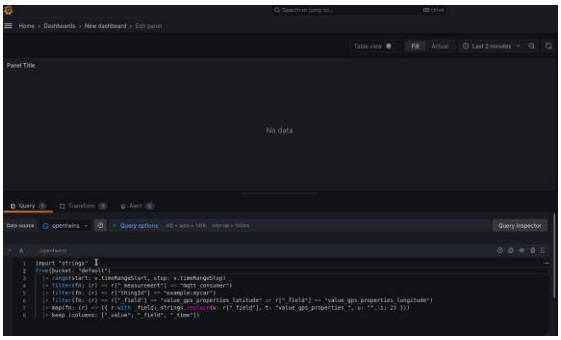




範例應用

Data source 選擇openywins
然後在下面填入

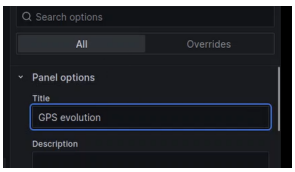
```
import "strings"
from(bucket: "default")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "mqtt_consumer")
  |> filter(fn: (r) => r["thingId"] == "example:mycar")
  |> filter(fn: (r) => r["_field"] ==
"value_gps_properties_latitude" or r["_field"] ==
"value_gps_properties_longitude")
  |> map(fn: (r) => ({ r with _field: strings.replace(v: r["_field"], t:
"value_gps_properties_", u: "", i: 2) }))
  |> keep (columns: ["_value", "_field", "_time"])
```



範例應用

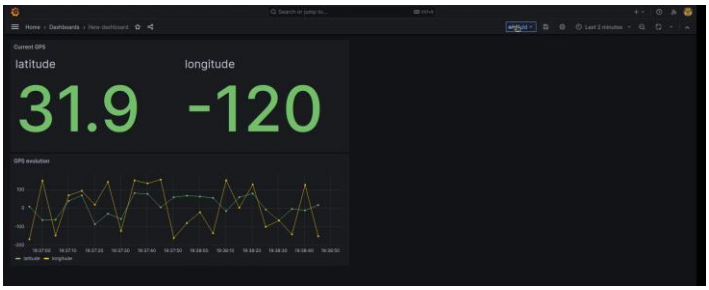
圖表格式為Time series

修改title為GPS evolution



範例應用

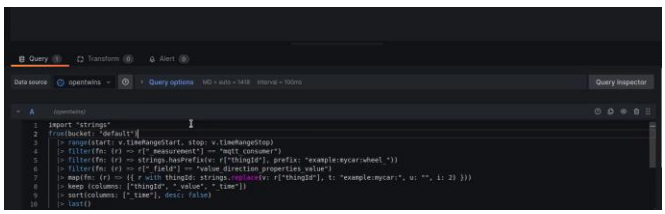
儲存 然後添加新表格



範例應用

Data source 選擇openywins
然後在下面填入

```
import "strings"
from(bucket: "default")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "mqtt_consumer")
  |> filter(fn: (r) => strings.hasPrefix(v: r["thingId"], prefix:
"example:mycar:wheel_"))
  |> filter(fn: (r) => r["_field"] ==
"value_direction_properties_value")
  |> map(fn: (r) => ({ r with thingId: strings.replace(v:
r["thingId"], t: "example:mycar:", u: "", i: 2) }))
  |> keep (columns: ["thingId", "_value", "_time"])
  |> sort(columns: ["_time"], desc: false)
  |> last()
```

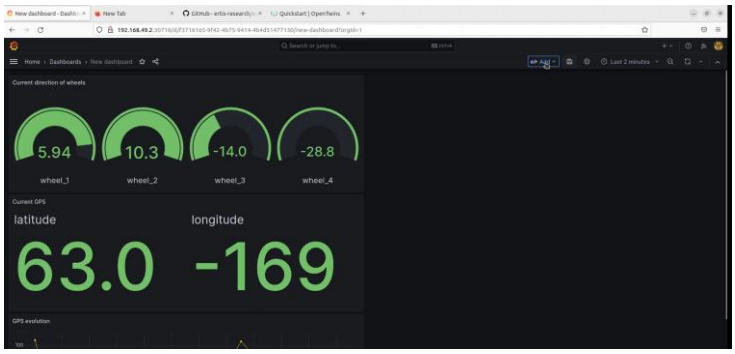


範例應用

表格格式選Gauge 修改title為Current direction of wheels

範例應用

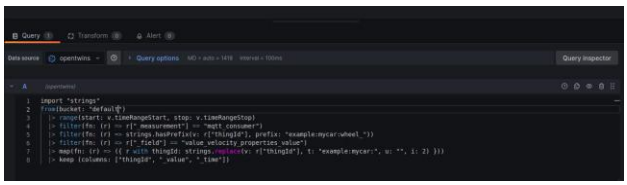
儲存，添加新表格



範例應用

Data source 選擇openywins
然後在下面填入

```
import "strings"
from(bucket: "default")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "mqtt_consumer")
  |> filter(fn: (r) => strings.hasPrefix(v: r["thingId"], prefix:
"example:mycar:wheel_"))
  |> filter(fn: (r) => r["_field"] ==
"value_velocity_properties_value")
  |> map(fn: (r) => ({ r with thingId: strings.replace(v:
r["thingId"], t: "example:mycar:", u: "", i: 2) }))
  |> keep (columns: ["thingId", "_value", "_time"])
```



範例應用

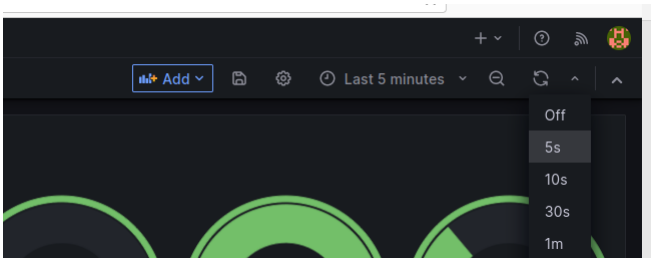
表格格式選Time series 修改title為Wheels velocity comparison

範例應用

儲存
將圖表排版

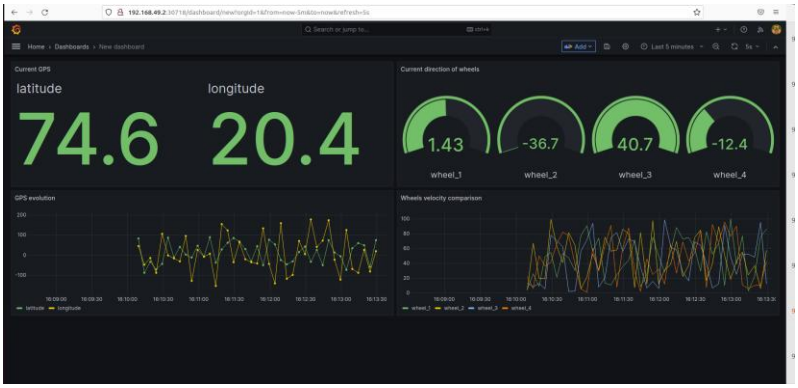
範例應用

- 將圖表調整成五秒更新一次



範例應用

- 完成圖



附件_opentwin重新開機後error解決方法

重新開機後 開啟mimikube

- minikube start

```
opentwin@opentwin-VirtualBox: $ minikube start
😄  minikube v1.33.1 on Ubuntu 22.04
✨  Using the docker driver based on existing profile
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🚜  Pulling base image v0.0.44 ...
🔄  Restarting existing docker container for "minikube" ...
🐳  Preparing Kubernetes v1.30.0 on Docker 26.1.1 ...
🔎  Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Enabled addons: storage-provisioner, default-storageclass
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```