Tutorial for installing whook on an Amazon EC2 Linux virtual machine

Explanations

The whook software as it stands creates an unsecured web server (http) capable of managing TradingView alerts and acting accordingly.

To make it more secure, we can insert a “Web proxy” server between it and TradingView, which will accept TradingView connections in secure mode (https) and transmit requests in non-secure mode to whook. We will use nginx as the proxy server.
Whook has been modified to accept being behind a proxy.
As nginx and whook are on the same server, unsecured communication between the two is not a problem, as whook is not accessible from the outside.

To enable nginx to manage https, you need to create certificates using the Letsencrypt tools. It's free and reliable. Letsencrypt is launched on the server to be secured, telling it that you are using nginx (so it can configure it straight away) and the name of the public server.

Unfortunately, AWS machines (***.aws.amazon.com) are not accepted by Letsencrypt because they can disappear and then reappear for another user. We'll get rid of this issue by creating our own public name (hostname) and associating it with the AWS machine.
Letsencrypt will then accept the new public name and create the certificates.
These certificates have a limited lifetime, so they need to be renewed regularly. Fortunately, letsencrypt will add an automatic task that will renew certificates automatically (it will check 2x a day if a certificate is less than 30 days old, and renew it).

To create our own hostname, we'll use an account on noip.com. The free version requires you to log in regularly (at least once every 90 days) to confirm that the name is still in use. The paid version avoids this (a few euros per month).

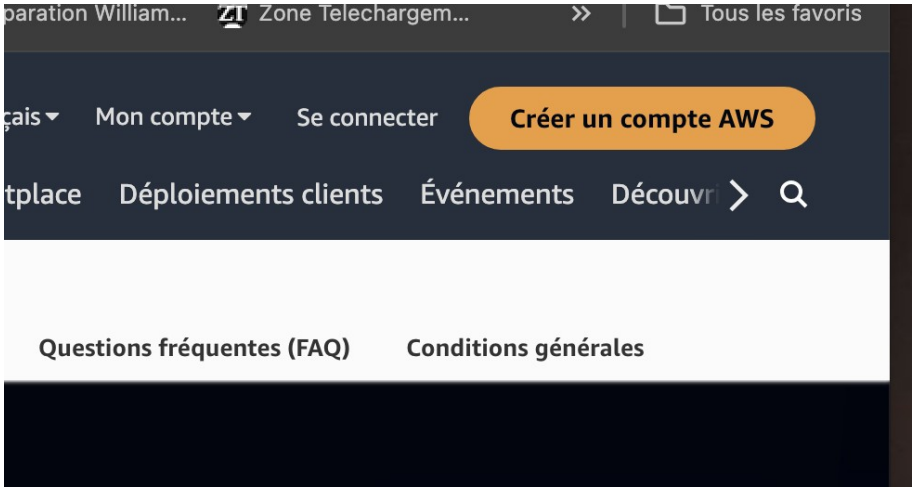
This tutorial explains all the steps to follow, from creating an AWS account to bringing the server online. You will then need to give TradingView the address of the server you have created, in the form https://<hostname>/whook

PS: Most of the screenshots are in French, so there may be a few differences between my description and what you see on your screen. If there are, don't hesitate to let me know. Sorry about that :)

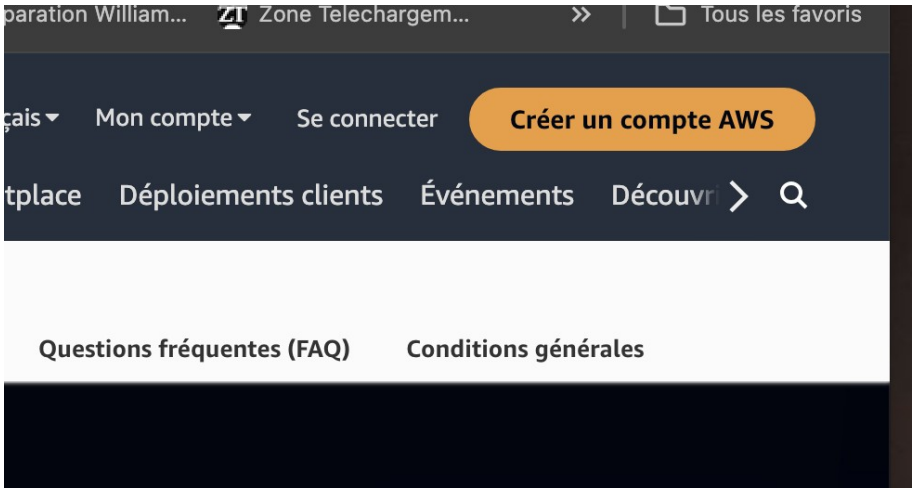
If you have any idea or process that can improve this tutorial, please let me know as well !

Create an Amazon AWS account

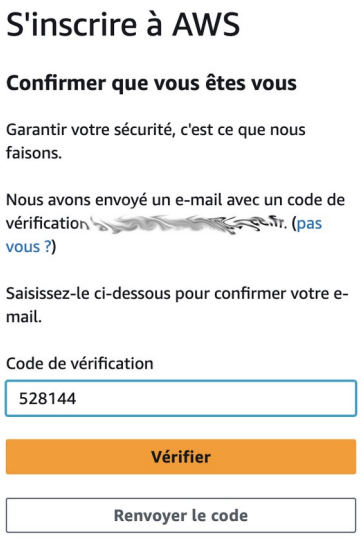
Go to https://aws.amazon.com/ and click on ‘Create an AWS account’.



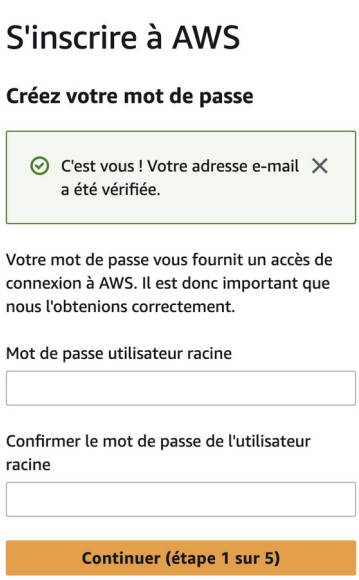
Enter your email address and choose a password, then click on ‘Verify email address’.



Retrieve the code sent by email and enter it in the form, then click on ‘Check’.

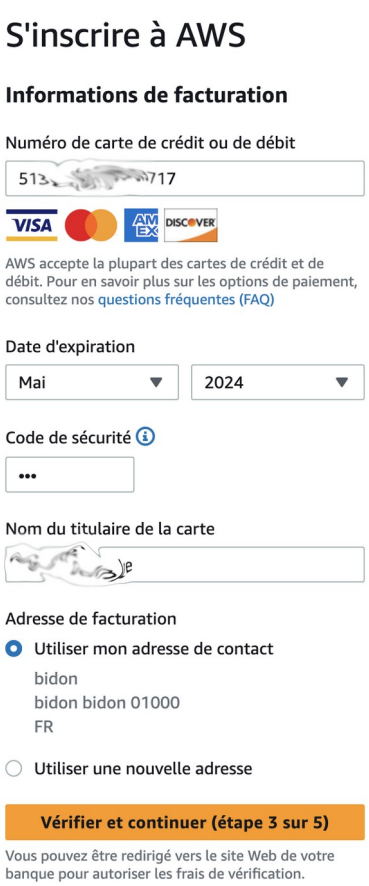


Choose a strong password and enter it 2 times, then click on ‘Continue (step 1 of 5)’.



Fill in the details (name and telephone number should suffice) and click on ‘Continue (step 2 of 5)’.

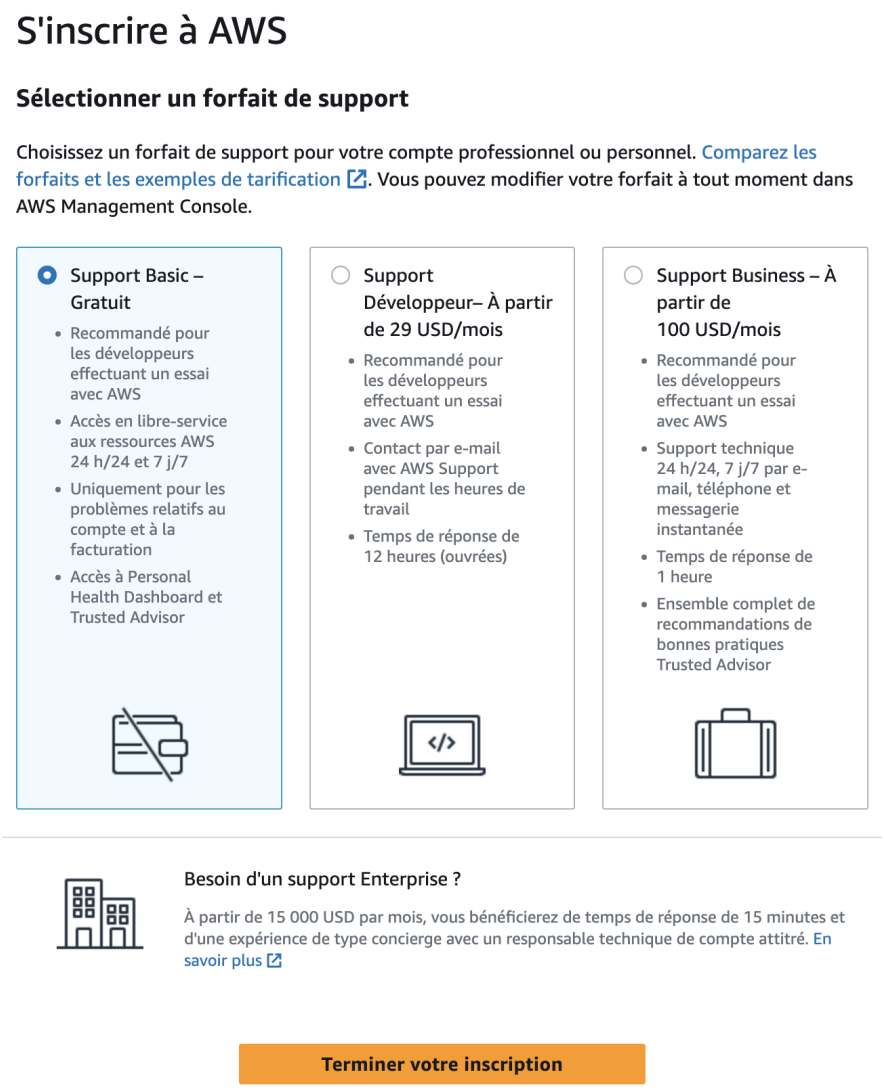
Fill in the billing information and click on ‘Check and continue (step 3 of 5)’.
A bank verification page may be displayed at this point, follow the instructions.



Confirm your identity by entering a mobile phone number. Complete the Captcha and click on ‘Send an SMS (step 4 of 5)’.

Enter the code received by text message and click on ‘Continue (step 4 of 5)’.

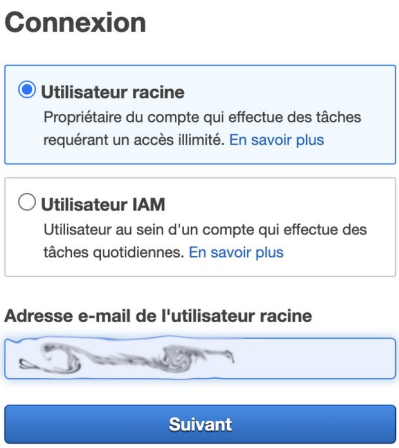
Choose the free package and click on ‘Complete your registration’.



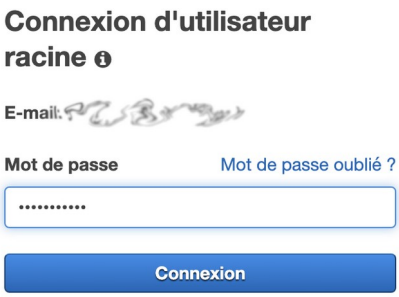
A confirmation message will be displayed. Click on ‘Access the AWS management console’.

Creating the Linux server

Log in as the root user, enter your email address and click ‘Next’.



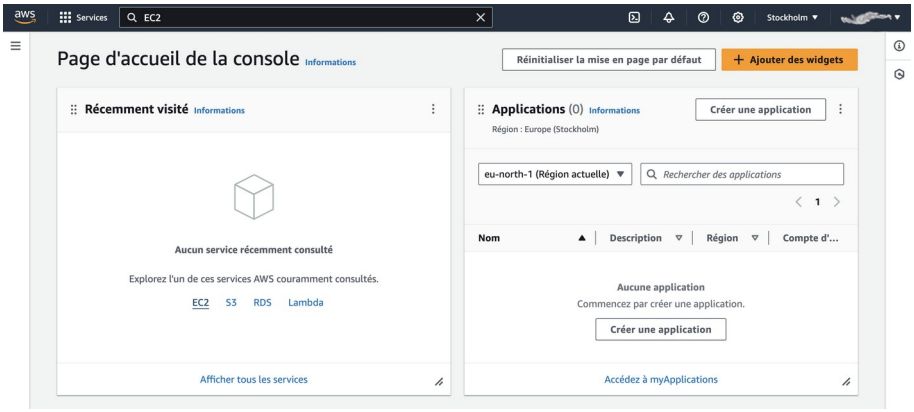
Enter your password and click on ‘Login’.



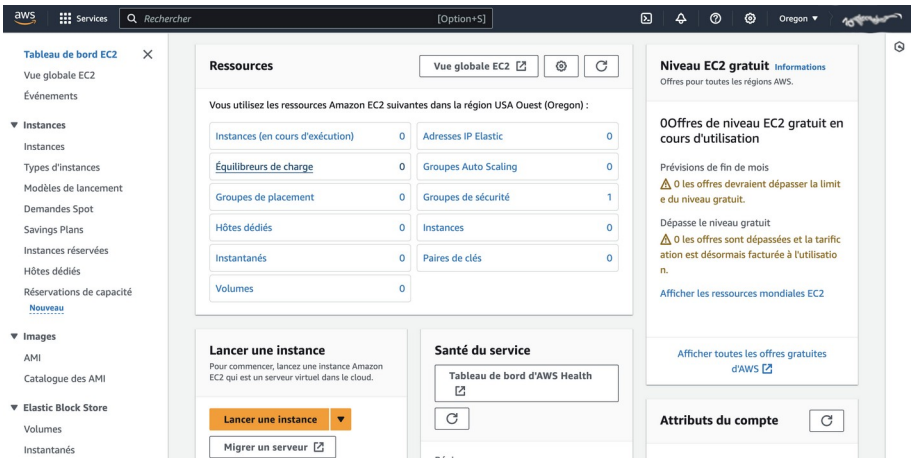
Confirm the Captcha

Enter the code received by email and click on ‘Check and continue’.

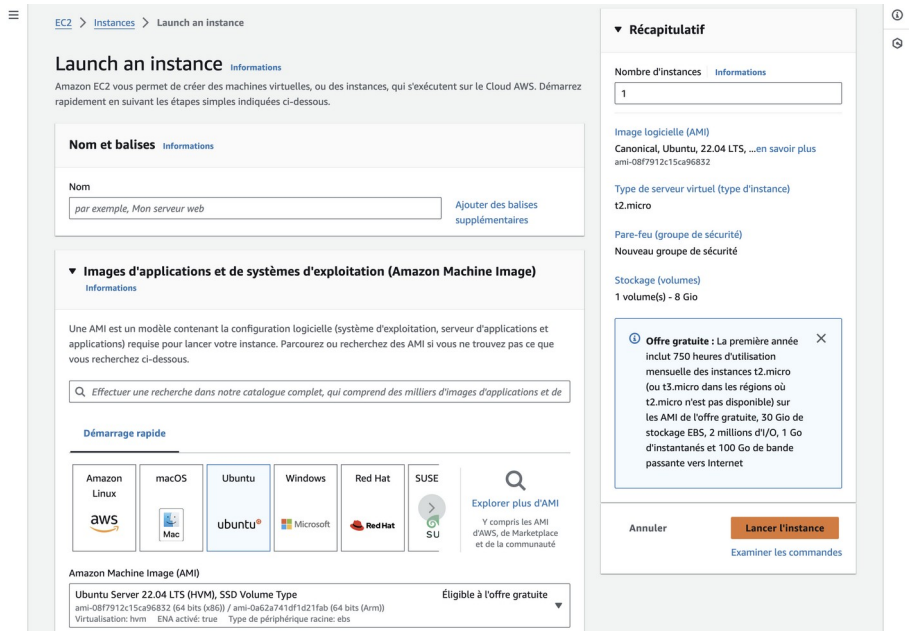
Change the region in the top right-hand corner to the region closest to the trading server, then click on EC2 (or search for ‘EC2’ in the top bar). I personally chose us-west-2b (oregon), because the tradingview server is in Portland.



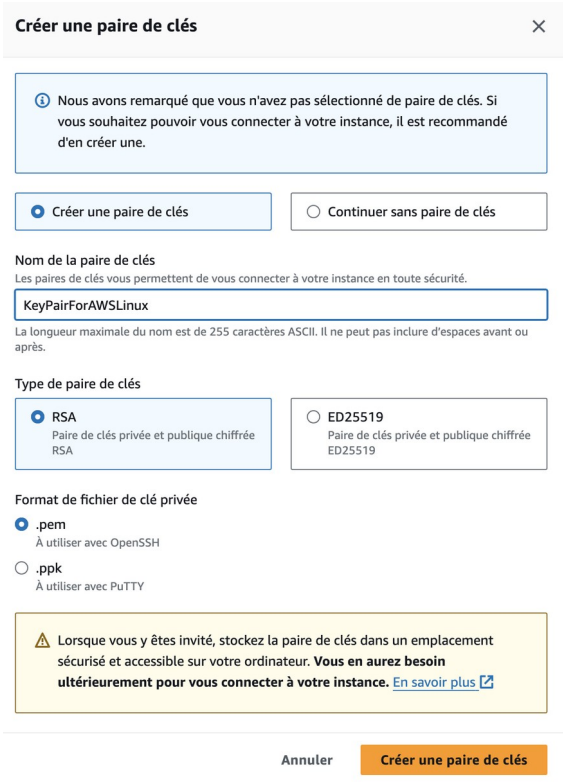
Click on “Start an instance”



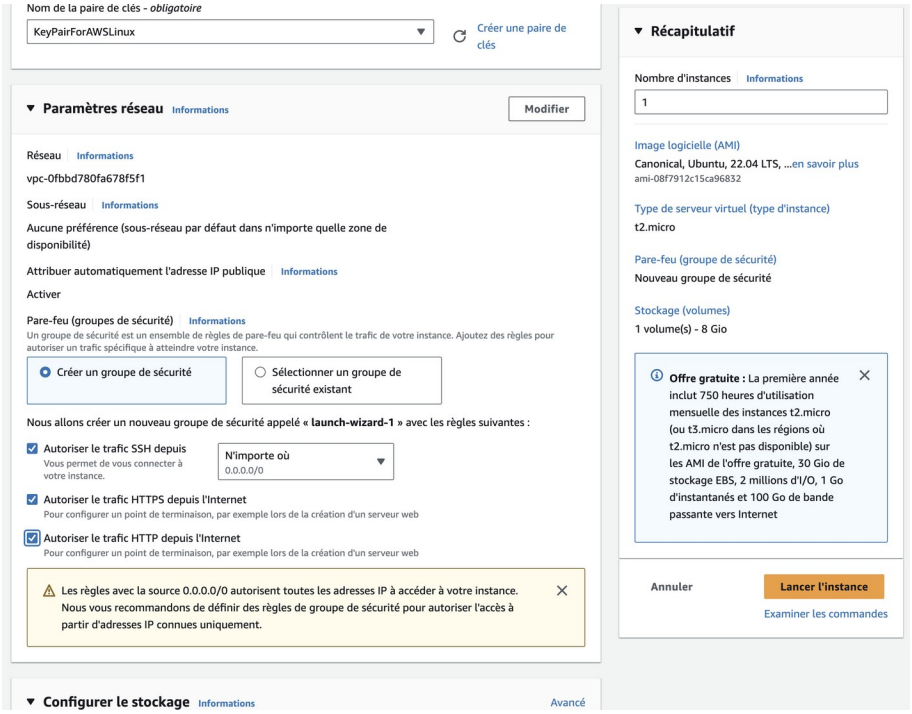
Choose Ubuntu in the “Quick start” category, then click on “Start instance”.



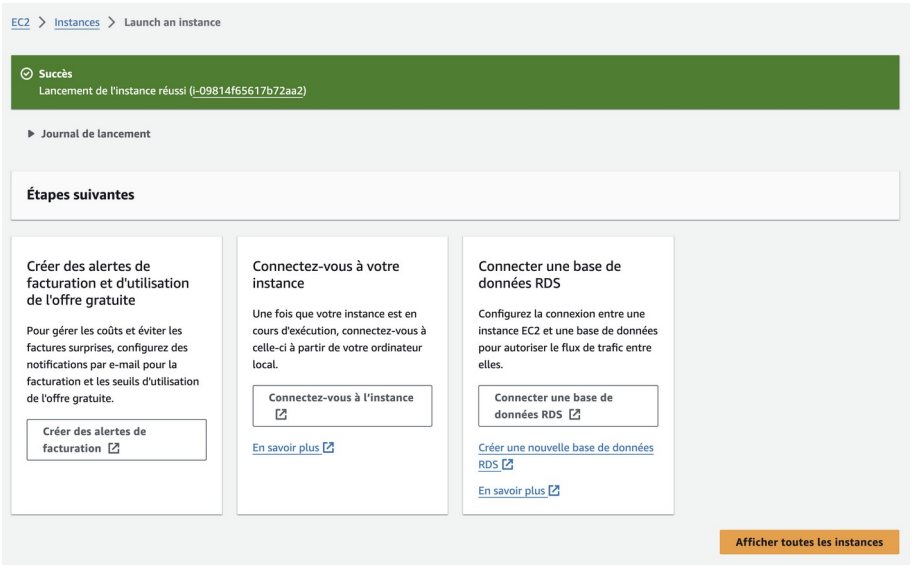
Choose a name in ‘Key pair name’ and click on ‘Create a key pair’. Save the file in a secure location (this will only be useful if you want to connect to the server using SSH, but we'll use another method).



Activate the 3 checkboxes and click on ‘start the instance’.

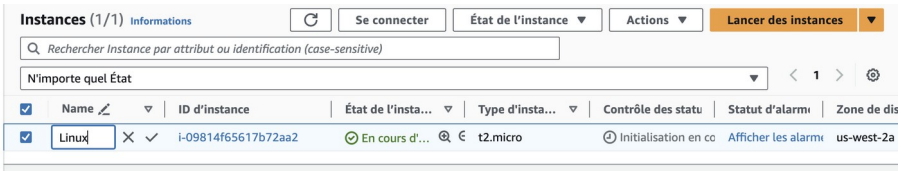


Click on “Display all instances”

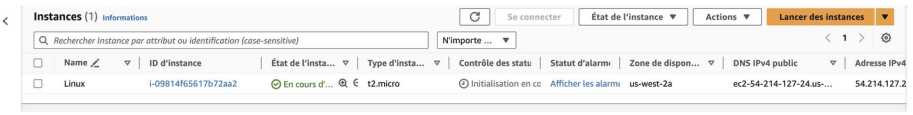


Setting up the Linux server (part 1)

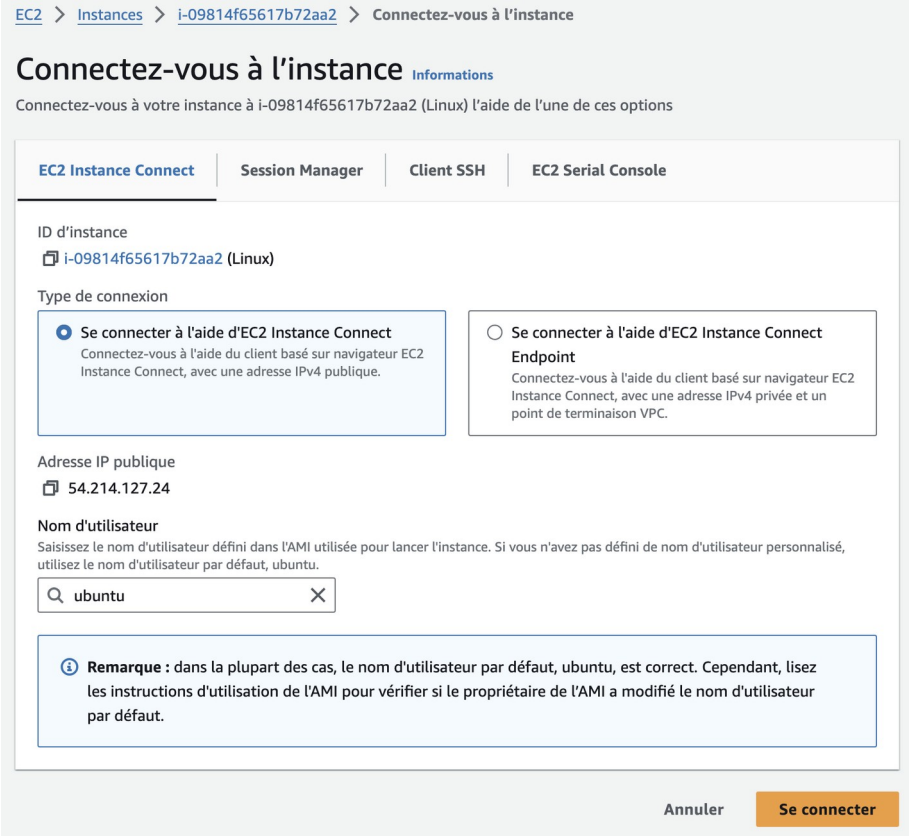
Enter a name for the instance and click on ‘Connect’.



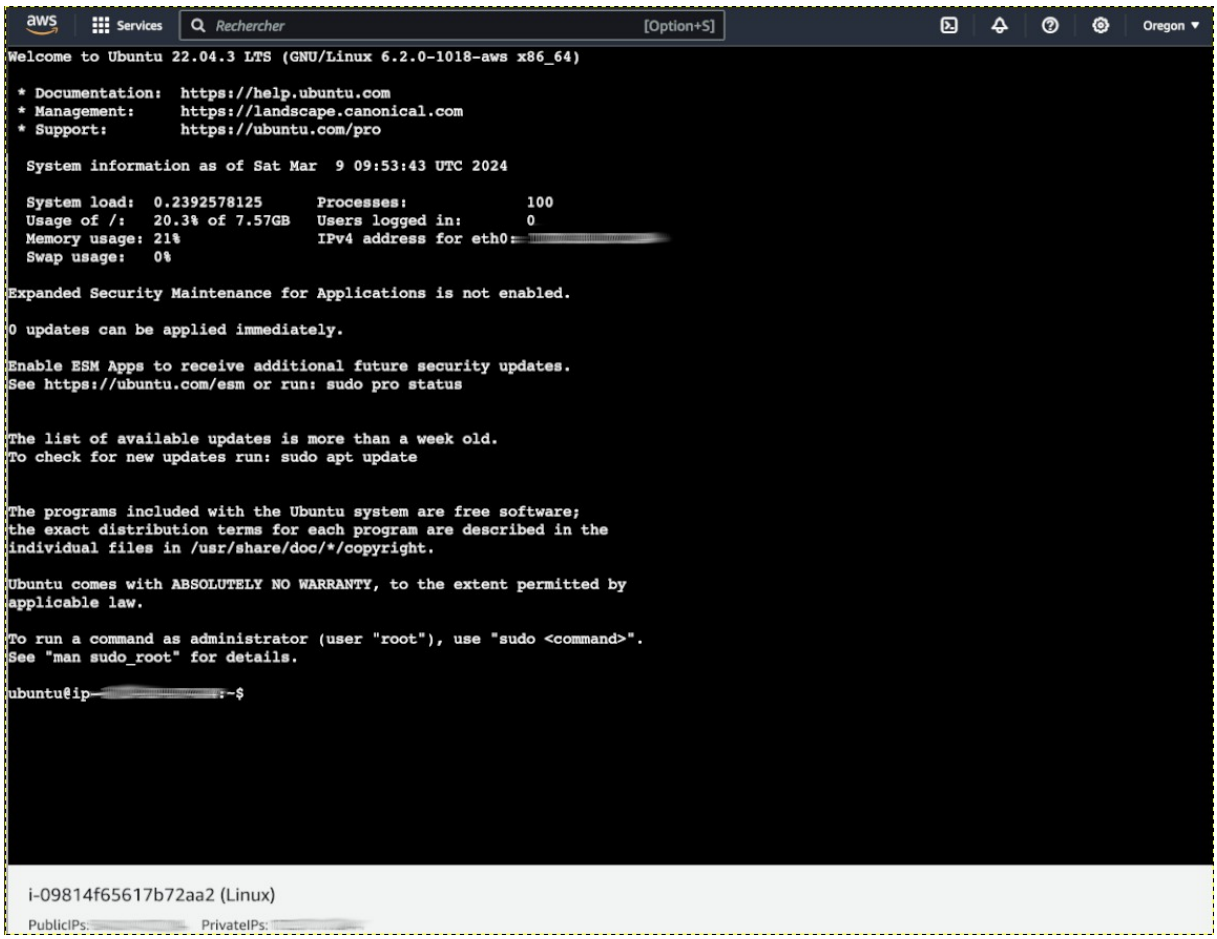
Note the public IPv4 address on the right



Click on ‘Connect’.



A shell opens in a new tab. Click on it to activate the focus.



Enter the commands :

```
sudo apt update && sudo apt install -y python3-pip letsencrypt nginx python3-certbot-nginx
```

```
sudo pip install ccxt
```

```
sudo pip install flask
```

```
git clone https://github.com/gdumgdum/whook_behind_proxy.git
```

```
cd whook_behind_proxy
```

```
chmod +x main.py
```

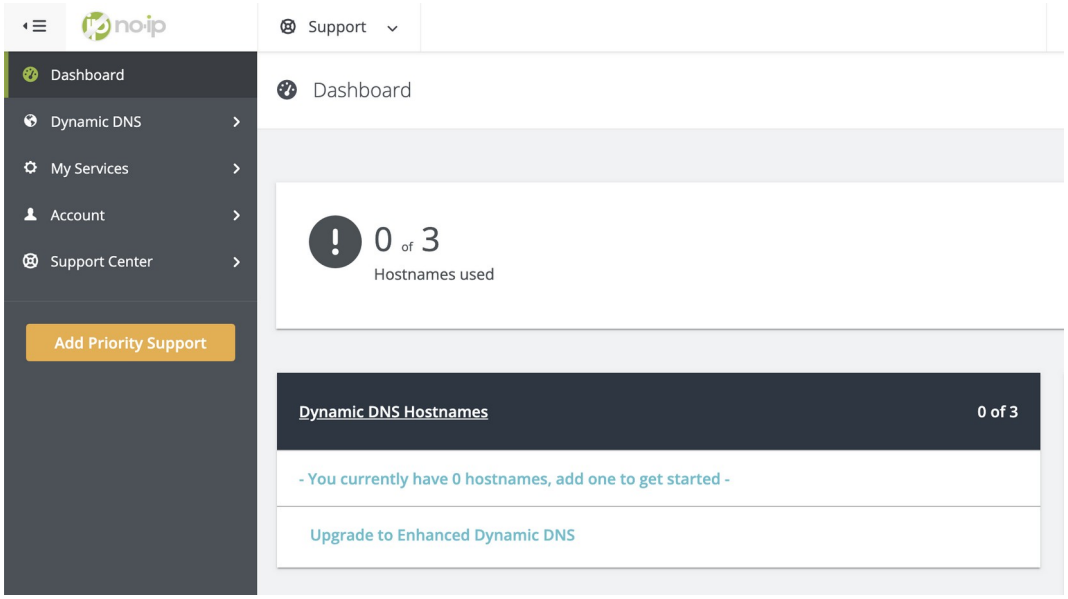
```
./main.py
```
which generates an error (necessary to create the accounts.json file)

Edit the file by typing:
```
nano accounts.json
```

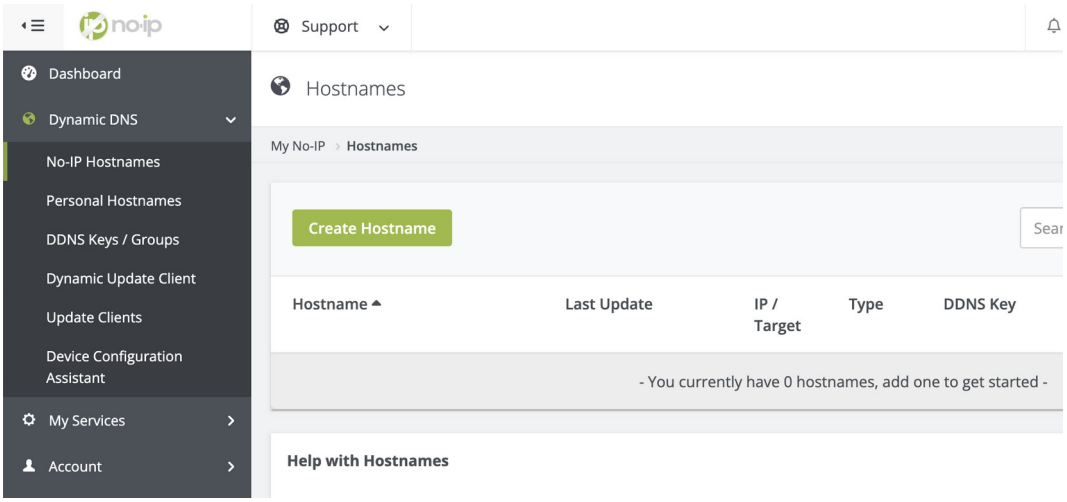
Modify the file by entering personal information, then save/quit by using the *Ctrl+X* command, then Y, then Enter

Creating a noip.com account

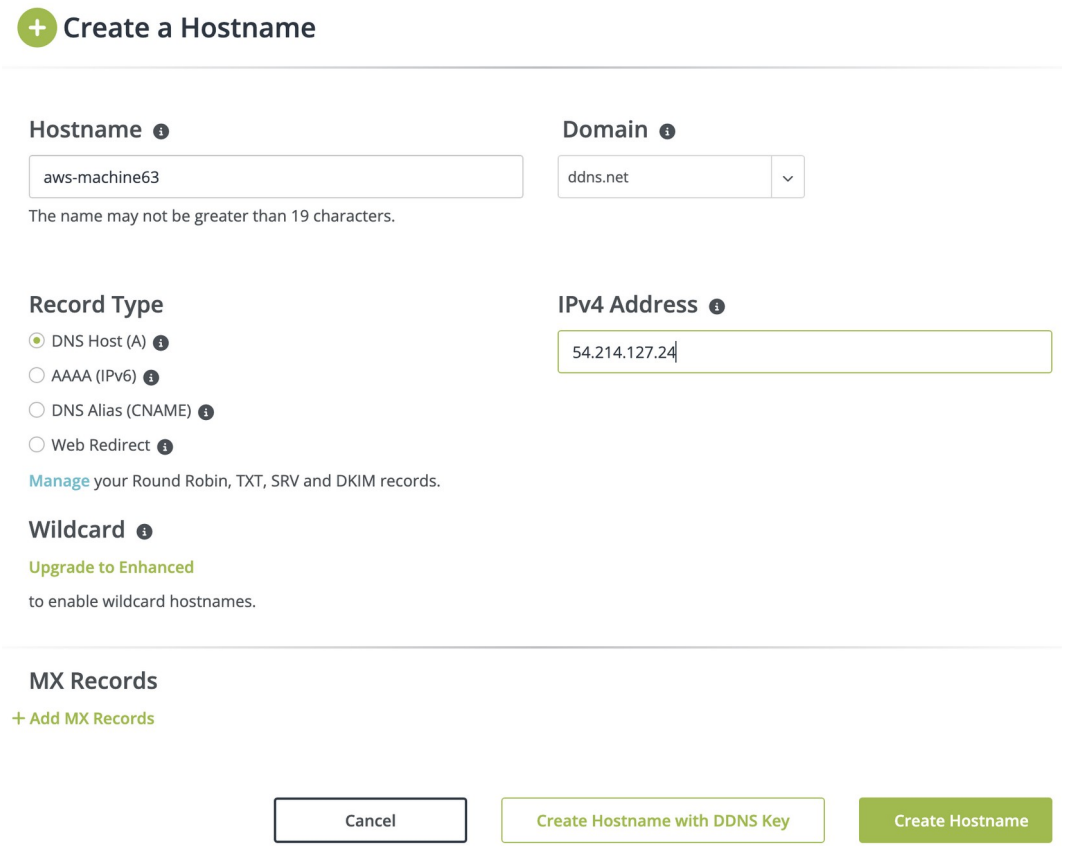
Create an account at https://www.noip.com/ then click on ‘Dynamic DNS’.



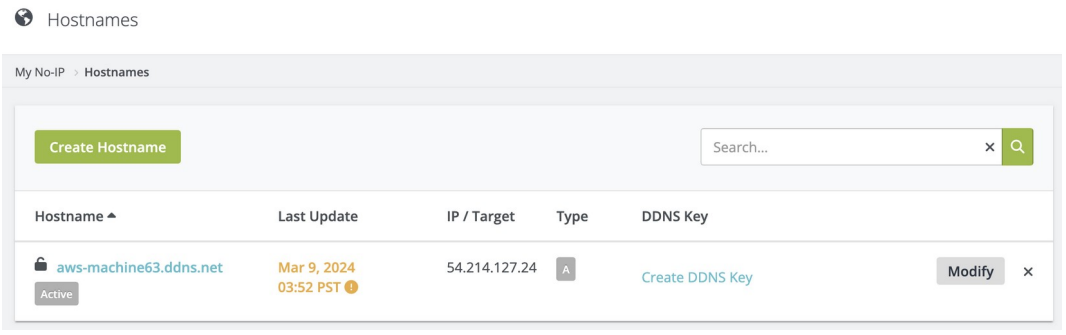
Click on ‘Create Hostname



Choose a name and a domain, fill in ‘IPv4 Address’ with the address noted above, then click on ‘Create Hostname’.



Check that everything is fine



Setting up the Linux server (part 2)

Return to the server console and type:

```
sudo nano /etc/nginx/sites-available/default
```

Modify the file to indicate the server name chosen on noip.com

```
#
# include snippets/snakeoil.conf;

root /var/www/html;

# Add index.php to the list if you are using PHP
index index.html index.htm index.nginx-debian.html;

server_name aws-machine63.ddns.net;

location / {
        # First attempt to serve request as file, then
        # as directory, then fall back to displaying a 4
        try_files $uri $uri/ =404;
}
```

Save and exit using *Ctrl+X*, Y, Enter

Check that the configuration is correct by typing:

```
sudo nginx -t
```

```
ubuntu@ip-[redacted]:/etc/nginx/sites-available$ sudo nano /etc/nginx/sites-available/default
ubuntu@ip-[redacted]:/etc/nginx/sites-available$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
ubuntu@ip-[redacted]:/etc/nginx/sites-available$
```

Restart Nginx

```
sudo systemctl reload nginx
```

Create HTTPS certificates by typing:

```
sudo certbot --nginx -d <hostname de noip.com>
```

Replacing the end with the name chosen on noip.com
Warning: it is better to answer N to the second question to avoid spam.

example :

```
ubuntu@ip-XXX-XXX-XXX-XXX:/etc/nginx/sites-available$ sudo certbot --nginx -d aws-
machine63.ddns.net
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Enter email address (used for urgent renewal and security notices)
 (Enter 'c' to cancel): ga******@******.fr

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.3-September-21-2022.pdf. You must
agree in order to register with the ACME server. Do you agree?
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
(Y)es/(N)o: Y
```

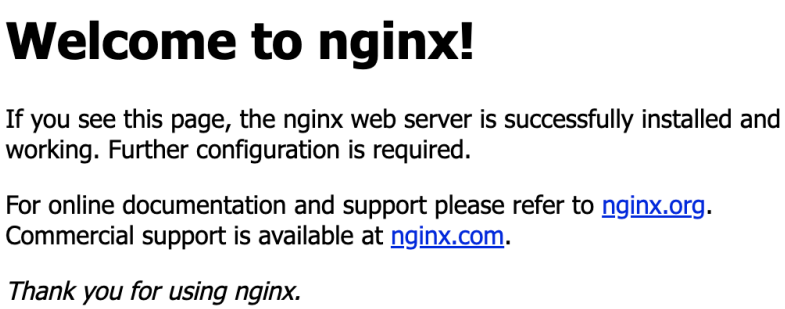
```
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Would you be willing, once your first certificate is successfully issued, to
share your email address with the Electronic Frontier Foundation, a founding
partner of the Let's Encrypt project and the non-profit organization that
develops Certbot? We'd like to send you email about our work encrypting the web,
EFF news, campaigns, and ways to support digital freedom.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
(Y)es/(N)o: Y
Account registered.
Requesting a certificate for aws-machine63.ddns.net

Successfully received certificate.
Certificate is saved at: /etc/letsencrypt/live/aws-machine63.ddns.net/fullchain.pem
Key is saved at:         /etc/letsencrypt/live/aws-machine63.ddns.net/privkey.pem
This certificate expires on 2024-06-07.
These files will be updated when the certificate renews.
Certbot has set up a scheduled task to automatically renew this certificate in the background.

Deploying certificate
Successfully deployed certificate for aws-machine63.ddns.net to /etc/nginx/sites-enabled/default
Congratulations! You have successfully enabled HTTPS on https://aws-machine63.ddns.net

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
If you like Certbot, please consider supporting our work by:
 * Donating to ISRG / Let's Encrypt:   https://letsencrypt.org/donate
 * Donating to EFF:                    https://eff.org/donate-le
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Open a new browser tab and go to https://aws-machine63.ddns.net/, replacing the name with the hostname chosen on noip.com. The result should be as follows:



Then type the command:

```
sudo nano /etc/nginx/sites-available/default
```

Modify the file by replacing the location section

```
        server_name aws-machine63.ddns.net;

        # location / {
        #       # First attempt to serve request as file, then
        #       # as directory, then fall back to displaying a 404.
        #       try_files $uri $uri/ =404;
        # }
        location / {
            proxy_pass http://127.0.0.1:50000/;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;
            proxy_set_header X-Forwarded-Host $host;
            proxy_set_header X-Forwarded-Prefix /;
        }

        # pass PHP scripts to FastCGI server
```

Save and exit (*Ctrl+X*, Y, Enter)

Restart Nginx:

```
sudo systemctl reload nginx
```

Creation of the whook service

In the console:

```
sudo nano /etc/systemd/system/whook.service
```

And copy/paste this content:

```
[Unit]
Description=Whook service
After=multi-user.target

[Service]
Type=simple
Restart=always
ExecStart=/usr/bin/python3 /home/ubuntu/whook_behind_proxy/main.py
WorkingDirectory=/home/ubuntu/whook_behind_proxy

[Install]
WantedBy=multi-user.target
```

Save and exit (*Ctrl+X*, Y, Enter)

Activate/start the service:

```
sudo systemctl enable whook
sudo systemctl start whook
```

Check service start-up:

```
sudo systemctl status whook
```

You can view the service logs:

```
tail -100 /var/log/syslog
```

Whook update

In your AWS server console, go into your whook repository:

```
cp /home/ubuntu/yourwhookrep
```

then use this git command :

```
git pull
```

If it’s already up to date il will returns :

```
Already up to date.
```

If some modification has been done to the accounts.json file, don’t forget to update it manually with nano.