OCTUBRE 2022

CLOUD COMPUTING. CLOUD ROBOTICS.

ACTIVIDAD 2 – AWS IoT – Node-Red

PRESENTADO A

Rodríguez Ismael
Rodríguez Uguren Sebastián

PRESENTADO POR

Grupo 4
Juan Cruz Verdolotti
Pablo Gagliardi

Tabla de contenido

página

Consigna..... 3
Resolución punto 1..... 4
Resolución punto 2..... 7
Resolución punto 3..... 10
Resolución punto 4..... 14
Resolución punto 5..... 20
Resolución punto 6..... 21
Resolución punto 7..... 21

Cierre de la actividad..... 30
Información adicional..... 36

ACTIVIDAD 2 – AWS IoT – Node-Red

INFORME DE ACTIVIDAD

1) Se deberá entregar un documento en formato PDF, en el que se expliquen detalladamente, los pasos que se han seguido para resolver las actividades propuestas. Dicho documento, debe incluir las capturas de pantalla necesarias en las que se pueda ver el trabajo del alumno. Además, se deberá entregar un VIDEO MP4 mostrando el funcionamiento del sistema, junto con el flujo de Node-RED con nodos “debug” para mostrar el funcionamiento.

2) Plazo de entrega: Jueves 6/10/2022 hasta las 24hs.

ACTIVIDAD PROPUESTA

1) Desplegar dos instancias t2.micro con Linux Debian 11 en AWS EC2: Instancia A e Instancia B.

2) Instalar “Node-RED” en ambas instancias.

3) Instalar el nodo “node-red-dashboard”.

4) Registrar ambas instancias como objetos/dispositivos en AWS IoT para poder operar mediante el protocolo MQTT.

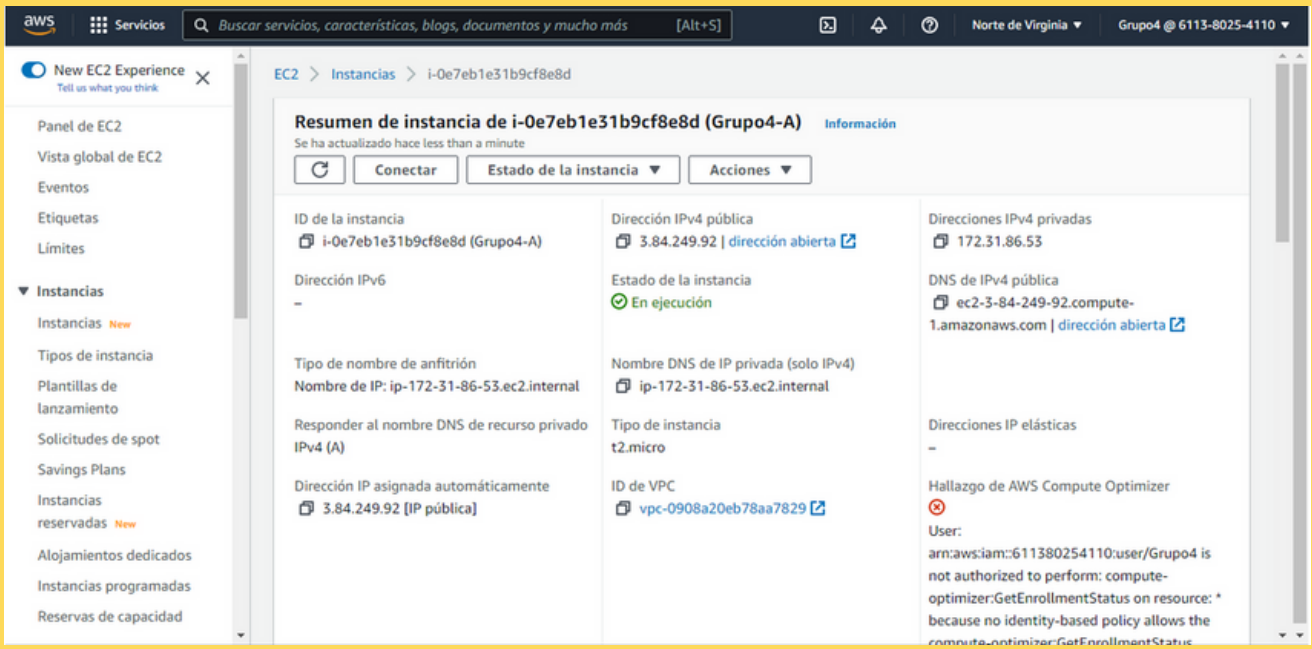
5) Desarrollar, en la instancia A, la solución de un “CONTROL REMOTO” para gestionar la iluminación de un ambiente:´botón “ENCENDIDO/ON”, botón “APAGADO/OFF”, informe de estado de la luminaria y funcionalidad “TIMER de 5s” para que activada dicha funcionalidad, al encender la luminaria se apague a los 5 segundos.

6) Desarrollar, en la instancia B, la solución para controlar la “ILUMINACIÓN” de un ambiente simulando el “ENCENDIDO/APAGADO” de la luminaria, informando el estado de la “LUMINARIA” y el estado del “TIMER”.

7) Las instancias A y B deben comunicarse por medio de MQTT utilizando el bróker de AWS IoT.

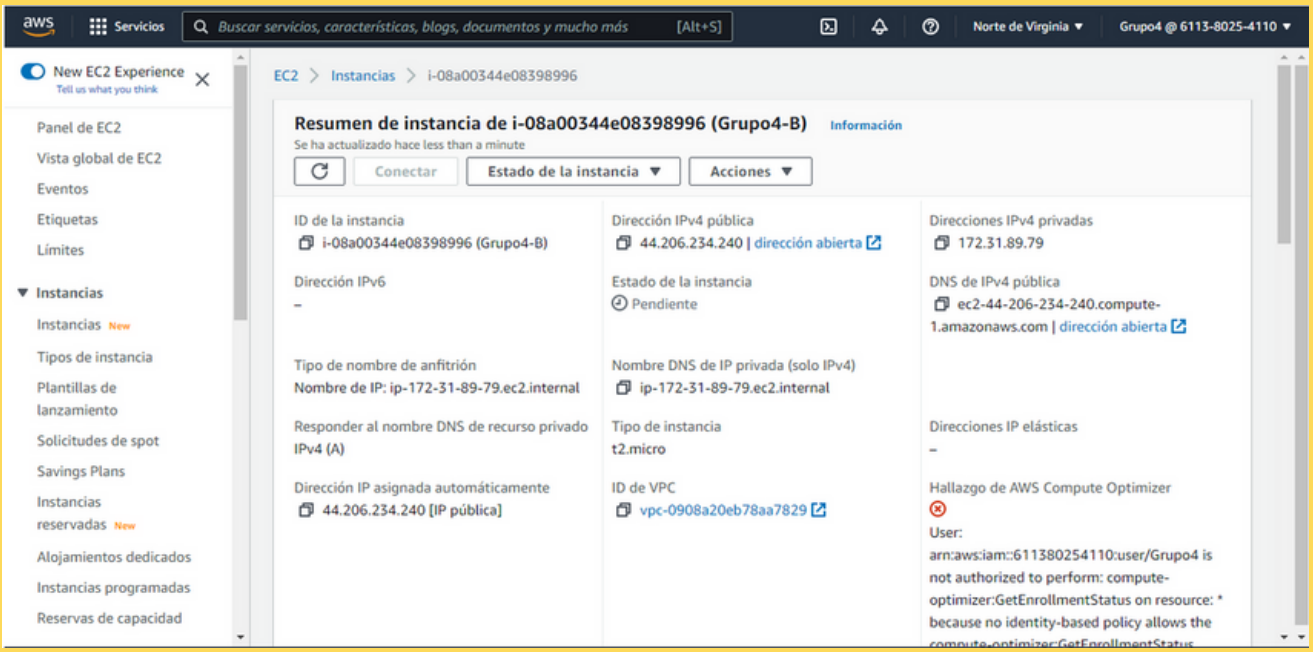
1) Desplegar dos instancias t2.micro con Linux Debian 11 en AWS EC2: Instancia A e Instancia B.

Lanzamos las instancias siguiendo los pasos vistos en la actividad anterior, nombrándolas como "Grupo4-A" y "Grupo4-B"

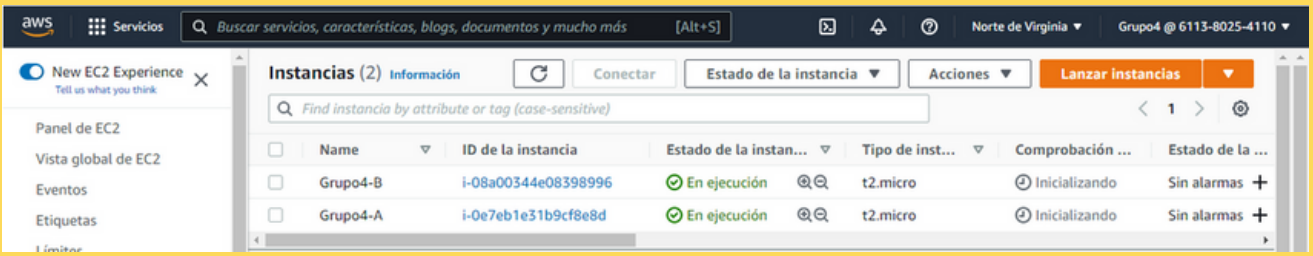


Nota: Por una cuestión de simplicidad y redundancia, se mostrarán las capturas correspondientes a una de las dos instancias, dejando por entendido que repetimos los pasos sobre la otra instancia (siempre que así se requiera).

Se muestra el resumen de la instancia "Grupo4-B", donde todavía se ve pendiente su estado.



Y a continuación, se muestra el estado de las dos instancias "Grupo4-B" y "Grupo4-A", que se encuentran 'En ejecución'.



Realizamos en cada instancia los comandos para luego poder instalar 'node red'.

```
admin@ip-172-31-86-53: ~
login as: admin
Authenticating with public key "Grupo4-1"
Linux ip-172-31-86-53 5.10.0-14-cloud-amd64 #1 SMP Debian 5.10.113-1 (2022-04-29
) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
admin@ip-172-31-86-53:~$ sudo apt update
Get:1 http://security.debian.org/debian-security bullseye-security InRelease [48
.4 kB]
Get:2 http://cdn-aws.deb.debian.org/debian bullseye InRelease [116 kB]
Get:3 http://cdn-aws.deb.debian.org/debian bullseye-updates InRelease [44.1 kB]
Get:4 http://cdn-aws.deb.debian.org/debian bullseye-backports InRelease [49.0 kB
]
Get:5 http://security.debian.org/debian-security bullseye-security/main Sources
[159 kB]
Get:6 http://security.debian.org/debian-security bullseye-security/main amd64 Pa
ckages [186 kB]
Get:7 http://security.debian.org/debian-security bullseye-security/main Translat
ion-en [117 kB]
Get:8 http://cdn-aws.deb.debian.org/debian bullseye/main Sources [8633 kB]
Get:9 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 Packages [8184 kB
]
Get:10 http://cdn-aws.deb.debian.org/debian bullseye/main Translation-en [6239 k
B]
Get:11 http://cdn-aws.deb.debian.org/debian bullseye-updates/main Sources [3588
B]
Get:12 http://cdn-aws.deb.debian.org/debian bullseye-updates/main amd64 Packages
 [6344 B]
Get:13 http://cdn-aws.deb.debian.org/debian bullseye-updates/main Translation-en
 [5890 B]
Get:14 http://cdn-aws.deb.debian.org/debian bullseye-backports/main Sources [297
 kB]
Get:15 http://cdn-aws.deb.debian.org/debian bullseye-backports/main amd64 Packag
es [328 kB]
Get:16 http://cdn-aws.deb.debian.org/debian bullseye-backports/main Translation-
en [267 kB]
Fetched 24.7 MB in 4s (5923 kB/s)
Reading package lists... Done
Building dependency tree... Done
```

```
admin@ip-172-31-86-53:~$ sudo apt update
```

```
admin@ip-172-31-86-53: ~
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
53 packages can be upgraded. Run 'apt list --upgradable' to see them.
admin@ip-172-31-86-53:~$ sudo apt install snapd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  dirmngr gnupg gnupg-l10n gnupg-utils gnupg1 gnupg1-l10n gpg gpg-agent gpg-wks-client gpg-wks-server gpgconf gpgsm gpgv libassuan0 libksba8 liblzo2-2
  libnpth0 pinentry-curses squashfs-tools
Suggested packages:
  dbus-user-session pinentry-gnome3 tor parcimonie xloadimage libpcsclite1 xloadimage | imagemagick | eog scdaemon pinentry-doc zenity | kdialog
The following NEW packages will be installed:
  dirmngr gnupg gnupg-l10n gnupg-utils gnupg1 gnupg1-l10n gpg gpg-agent gpg-wks-client gpg-wks-server gpgconf gpgsm libassuan0 libksba8 liblzo2-2 libnpth0
  pinentry-curses snapd squashfs-tools
The following packages will be upgraded:
  gpgv
1 upgraded, 19 newly installed, 0 to remove and 52 not upgraded.
Need to get 22.9 MB of archives.
After this operation, 77.6 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 gpgv amd64 2.2.27-2+deb11u2 [626 kB]
Get:2 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 libassuan0 amd64 2.5.3-7.1 [50.5 kB]
Get:3 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 gpgconf amd64 2.2.27-2+deb11u2 [548 kB]
Get:4 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 libksba8 amd64 1.5.0-3 [123 kB]
Get:5 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 libnpth0 amd64 1.6-3 [19.0 kB]
Get:6 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 dirmngr amd64 2.2.27-2+deb11u2 [763 kB]
Get:7 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 gnupg-l10n all 2.2.27-2+deb11u2 [1086 kB]
Get:8 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 gnupg-utils amd64 2.2.27-2+deb11u2 [905 kB]
Get:9 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 gpg amd64 2.2.27-2+deb11u2 [928 kB]
Get:10 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 pinentry-curses amd64 1.1.0-4 [64.9 kB]
Get:11 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 gpg-agent amd64 2.2.27-2+deb11u2 [669 kB]
Get:12 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 gpg-wks-client amd64 2.2.27-2+deb11u2 [524 kB]
Get:13 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 gpg-wks-server amd64 2.2.27-2+deb11u2 [516 kB]
Get:14 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 gpgsm amd64 2.2.27-2+deb11u2 [645 kB]
Get:15 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 gnupg all 2.2.27-2+deb11u2 [825 kB]
Get:16 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 gnupg1 amd64 1.4.23-1.1 [602 kB]
Get:17 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 gnupg1-l10n all 1.4.23-1.1 [553 kB]
Get:18 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 liblzo2-2 amd64 2.10-2 [56.9 kB]
Get:19 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 squashfs-tools amd64 1:4.4-2+deb11u2 [135 kB]
Get:20 http://cdn-aws.deb.debian.org/debian bullseye/main amd64 snapd amd64 2.49-1+deb11u1 [13.3 MB]
Fetched 22.9 MB in 0s (53.4 MB/s)
Reading changelogs... Done
```

```
admin@ip-172-31-86-53:~$ sudo apt install snapd
```

```
Do you want to continue? [Y/n] y
```

```
admin@ip-172-31-86-53: ~
Unpacking liblzo2-2:amd64 (2.10-2) ...
Selecting previously unselected package squashfs-tools.
Preparing to unpack .../17-squashfs-tools_1%3a4.4-2+deb11u2_amd64.deb ...
Unpacking squashfs-tools (1:4.4-2+deb11u2) ...
Selecting previously unselected package snapd.
Preparing to unpack .../18-snapd_2.49-1+deb11u1_amd64.deb ...
Unpacking snapd (2.49-1+deb11u1) ...
Setting up libksba8:amd64 (1.5.0-3) ...
Setting up gnupg1 (1.4.23-1.1) ...
Setting up gnupg1-l10n (1.4.23-1.1) ...
Setting up liblzo2-2:amd64 (2.10-2) ...
Setting up libnpth0:amd64 (1.6-3) ...
Setting up libassuan0:amd64 (2.5.3-7.1) ...
Setting up gnupg-l10n (2.2.27-2+deb11u2) ...
Setting up gpgconf (2.2.27-2+deb11u2) ...
Setting up gpg (2.2.27-2+deb11u2) ...
Setting up gnupg-utils (2.2.27-2+deb11u2) ...
Setting up pinentry-curses (1.1.0-4) ...
Setting up gpg-agent (2.2.27-2+deb11u2) ...
Created symlink /etc/systemd/user/sockets.target.wants/gpg-agent-browser.socket → /usr/lib/systemd/user/gpg-agent-browser.socket.
Created symlink /etc/systemd/user/sockets.target.wants/gpg-agent-extra.socket → /usr/lib/systemd/user/gpg-agent-extra.socket.
Created symlink /etc/systemd/user/sockets.target.wants/gpg-agent-ssh.socket → /usr/lib/systemd/user/gpg-agent-ssh.socket.
Created symlink /etc/systemd/user/sockets.target.wants/gpg-agent.socket → /usr/lib/systemd/user/gpg-agent.socket.
Setting up squashfs-tools (1:4.4-2+deb11u2) ...
Setting up gpgsm (2.2.27-2+deb11u2) ...
Setting up dirmngr (2.2.27-2+deb11u2) ...
Created symlink /etc/systemd/user/sockets.target.wants/dirmngr.socket → /usr/lib/systemd/user/dirmngr.socket.
Setting up gpg-wks-server (2.2.27-2+deb11u2) ...
Setting up snapd (2.49-1+deb11u1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/snapd.recovery-chooser-trigger.service → /lib/systemd/system/snapd.recovery-chooser-trigger.servi
ce.
Created symlink /etc/systemd/system/multi-user.target.wants/snapd.seeded.service → /lib/systemd/system/snapd.seeded.service.
Created symlink /etc/systemd/system/cloud-final.service.wants/snapd.seeded.service → /lib/systemd/system/snapd.seeded.service.
Created symlink /etc/systemd/system/multi-user.target.wants/snapd.service → /lib/systemd/system/snapd.service.
Created symlink /etc/systemd/system/sockets.target.wants/snapd.socket → /lib/systemd/system/snapd.socket.
Setting up gpg-wks-client (2.2.27-2+deb11u2) ...
Setting up gnupg (2.2.27-2+deb11u2) ...
Processing triggers for man-db (2.9.4-2) ...
Processing triggers for dbus (1.12.20-2) ...
Processing triggers for libc-bin (2.31-13+deb11u3) ...
admin@ip-172-31-86-53:~$ sudo snap install core
2022-09-28T21:13:39Z INFO Waiting for automatic snapd restart...
core 16-2.57.2 from Canonical✓ installed
admin@ip-172-31-86-53:~$
```

```
admin@ip-172-31-86-53:~$ sudo snap install core
```

2) Instalar “Node-RED” en ambas instancias.

```
admin@ip-172-31-86-53: ~
Preparing to unpack .../17-squashfs-tools_1%3a4.4-2+deb11u2_amd64.deb ...
Unpacking squashfs-tools (1:4.4-2+deb11u2) ...
Selecting previously unselected package snapd.
Preparing to unpack .../18-snapd_2.49-1+deb11u1_amd64.deb ...
Unpacking snapd (2.49-1+deb11u1) ...
Setting up libksba8:amd64 (1.5.0-3) ...
Setting up gnupg1 (1.4.23-1.1) ...
Setting up gnupg1-l10n (1.4.23-1.1) ...
Setting up liblzo2-2:amd64 (2.10-2) ...
Setting up libnpth0:amd64 (1.6-3) ...
Setting up libassuan0:amd64 (2.5.3-7.1) ...
Setting up gnupg-l10n (2.2.27-2+deb11u2) ...
Setting up gpgconf (2.2.27-2+deb11u2) ...
Setting up gpg (2.2.27-2+deb11u2) ...
Setting up gnupg-utils (2.2.27-2+deb11u2) ...
Setting up pinentry-curses (1.1.0-4) ...
Setting up gpg-agent (2.2.27-2+deb11u2) ...
Created symlink /etc/systemd/user/sockets.target.wants/gpg-agent-browser.socket → /usr/lib/systemd/user/gpg-agent-browser.socket.
Created symlink /etc/systemd/user/sockets.target.wants/gpg-agent-extra.socket → /usr/lib/systemd/user/gpg-agent-extra.socket.
Created symlink /etc/systemd/user/sockets.target.wants/gpg-agent-ssh.socket → /usr/lib/systemd/user/gpg-agent-ssh.socket.
Created symlink /etc/systemd/user/sockets.target.wants/gpg-agent.socket → /usr/lib/systemd/user/gpg-agent.socket.
Setting up squashfs-tools (1:4.4-2+deb11u2) ...
Setting up gpgsm (2.2.27-2+deb11u2) ...
Setting up dirmngr (2.2.27-2+deb11u2) ...
Created symlink /etc/systemd/user/sockets.target.wants/dirmngr.socket → /usr/lib/systemd/user/dirmngr.socket.
Setting up gpg-wks-server (2.2.27-2+deb11u2) ...
Setting up snapd (2.49-1+deb11u1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/snapd.recovery-chooser-trigger.service → /lib/systemd/system/snapd.recovery-chooser-trigger.servi
ce.
Created symlink /etc/systemd/system/multi-user.target.wants/snapd.seeded.service → /lib/systemd/system/snapd.seeded.service.
Created symlink /etc/systemd/system/cloud-final.service.wants/snapd.seeded.service → /lib/systemd/system/snapd.seeded.service.
Created symlink /etc/systemd/system/multi-user.target.wants/snapd.service → /lib/systemd/system/snapd.service.
Created symlink /etc/systemd/system/sockets.target.wants/snapd.socket → /lib/systemd/system/snapd.socket.
Setting up gpg-wks-client (2.2.27-2+deb11u2) ...
Setting up gnupg (2.2.27-2+deb11u2) ...
Processing triggers for man-db (2.9.4-2) ...
Processing triggers for dbus (1.12.20-2) ...
Processing triggers for libc-bin (2.31-13+deb11u3) ...
admin@ip-172-31-86-53:~$ sudo snap install core
2022-09-28T21:13:39Z INFO Waiting for automatic snapd restart...
core 16-2.57.2 from Canonical✓ installed
admin@ip-172-31-86-53:~$ sudo snap install node-red
node-red 3.0.2 from Node-RED-Team (noderedteam✓) installed
admin@ip-172-31-86-53:~$
```

```
admin@ip-172-31-86-53:~$ sudo snap install node-red
```

Utilizamos el comando:

~$*netstat -na | more*

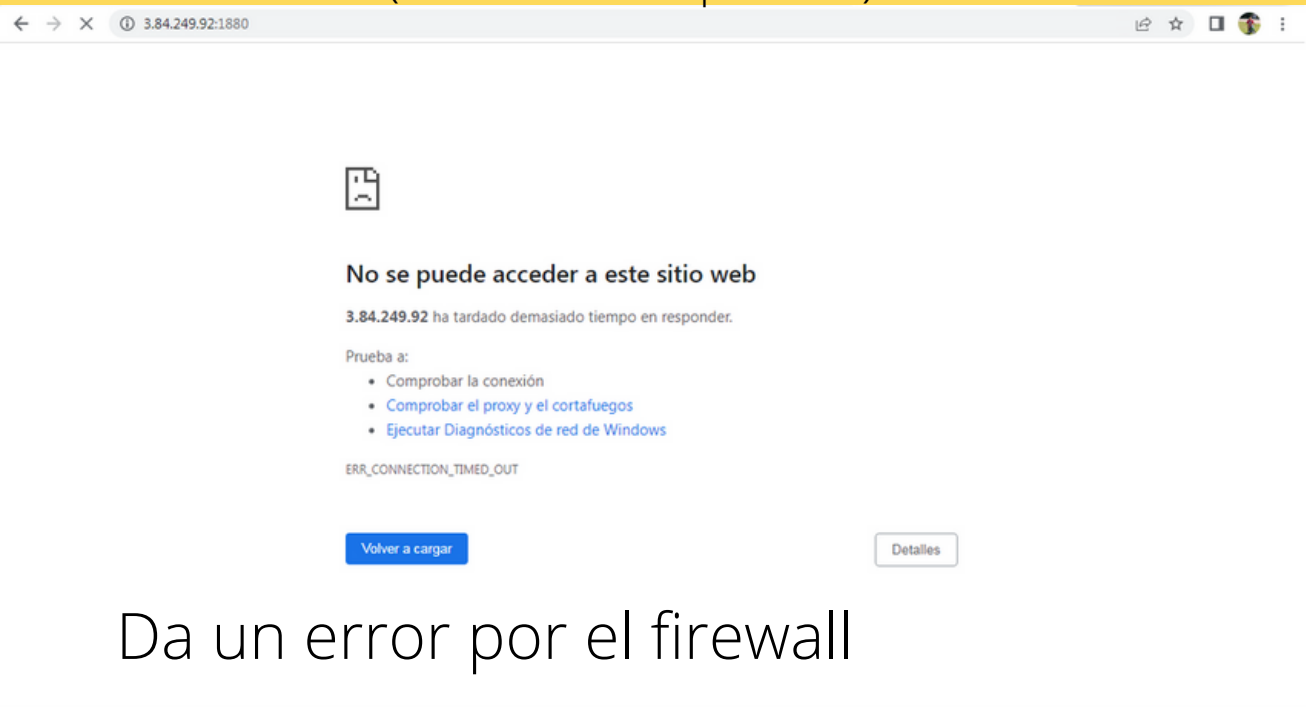
El cual nos muestra los puertos publicados de la instancia

```
admin@ip-172-31-86-53: ~
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:1880            0.0.0.0:*               LISTEN
tcp        0     64 172.31.86.53:22         163.10.33.202:1615      ESTABLISHED
tcp        0      0 172.31.86.53:33352      52.85.132.47:443        TIME_WAIT
tcp        0      0 172.31.86.53:51878      52.85.132.98:443        TIME_WAIT
tcp6       0      0 :::22                   :::*                    LISTEN
udp        0      0 0.0.0.0:68              0.0.0.0:*
udp        0      0 127.0.0.1:323           0.0.0.0:*
udp6       0      0 ::1:323                 :::*
udp6       0      0 fe80::10c8:24ff:fed:546 :::*
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ]         DGRAM                    11843    /run/user/1000/systemd/notify
unix  2      [ ACC ]     STREAM     LISTENING     11846    /run/user/1000/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     11112    /run/dbus/system_bus_socket
unix  2      [ ACC ]     STREAM     LISTENING     11114    /run/uuidd/request
unix  2      [ ]         DGRAM                    11209    /run/chrony/chronyd.sock
unix  3      [ ]         DGRAM                    9624     /run/systemd/notify
unix  2      [ ACC ]     STREAM     LISTENING     9627     /run/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     9629     /run/systemd/userdb/io.systemd.DynamicUser
unix  2      [ ACC ]     STREAM     LISTENING     9630     /run/systemd/io.system.ManagedOOM
unix  2      [ ]         DGRAM                    9639     /run/systemd/journal/syslog
unix  2      [ ACC ]     STREAM     LISTENING     9641     /run/systemd/fsck.progress
unix  10     [ ]         DGRAM                    9645     /run/systemd/journal/dev-log
unix  6      [ ]         DGRAM                    9647     /run/systemd/journal/socket
unix  2      [ ACC ]     STREAM     LISTENING     9649     /run/systemd/journal/stdout
unix  2      [ ACC ]     SEQPACKET  LISTENING     9651     /run/udev/control
unix  2      [ ACC ]     STREAM     LISTENING     9697     /run/systemd/journal/io.systemd.journal
unix  2      [ ACC ]     STREAM     LISTENING     13949    /run/snapd.socket
unix  2      [ ACC ]     STREAM     LISTENING     13950    /run/snapd-snap.socket
unix  3      [ ]         STREAM     CONNECTED     11420    /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     11231
unix  3      [ ]         DGRAM                    9626
unix  2      [ ]         DGRAM                    11170
unix  3      [ ]         STREAM     CONNECTED     11422    /run/systemd/journal/stdout
unix  2      [ ]         DGRAM                    11259
unix  3      [ ]         STREAM     CONNECTED     11272    /run/dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     11176
unix  3      [ ]         DGRAM                    9625
unix  3      [ ]         SEQPACKET  CONNECTED     11211
unix  2      [ ]         DGRAM                    11189
--More--
```

```
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:1880            0.0.0.0:*               LISTEN
```

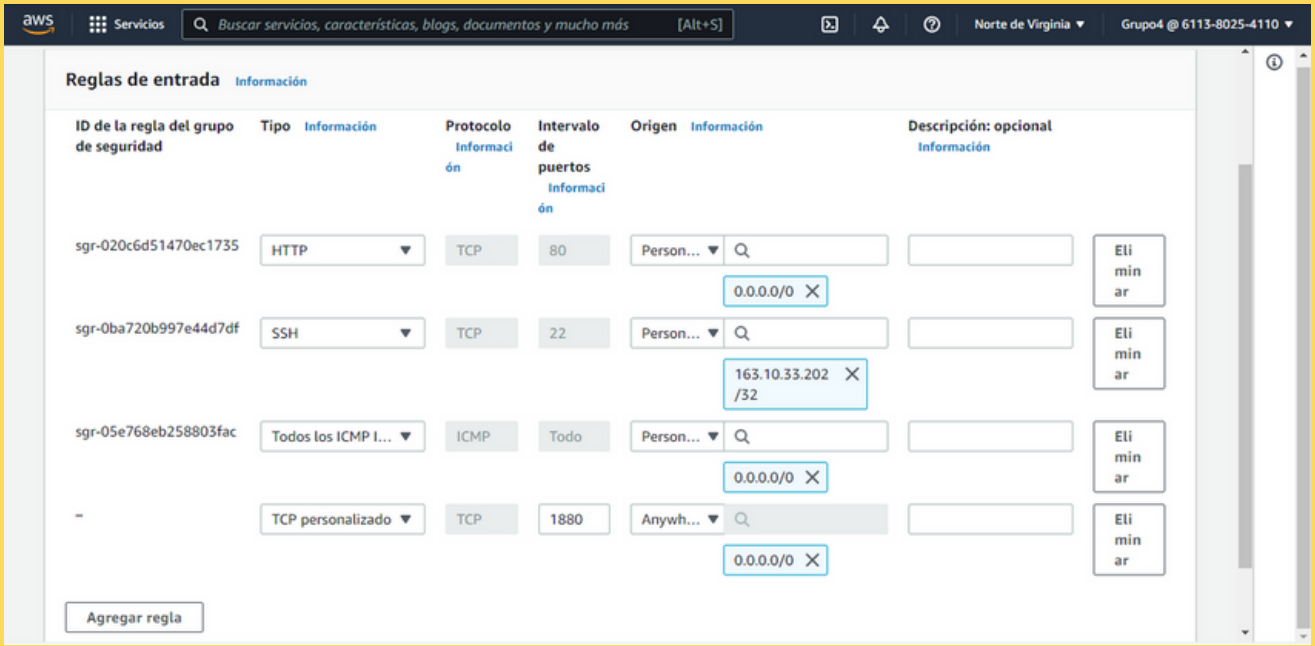
Abrimos el navegador y vamos a:

3.84.249.92:1880 (“IP instancia”:"puerto")

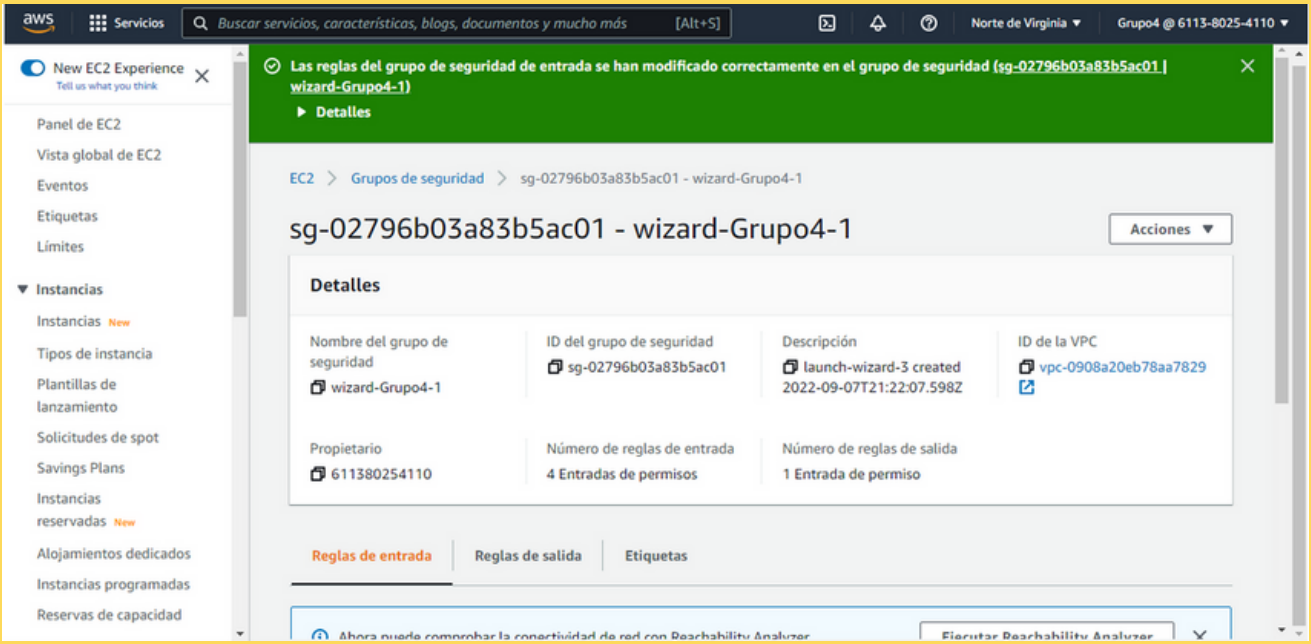


Da un error por el firewall

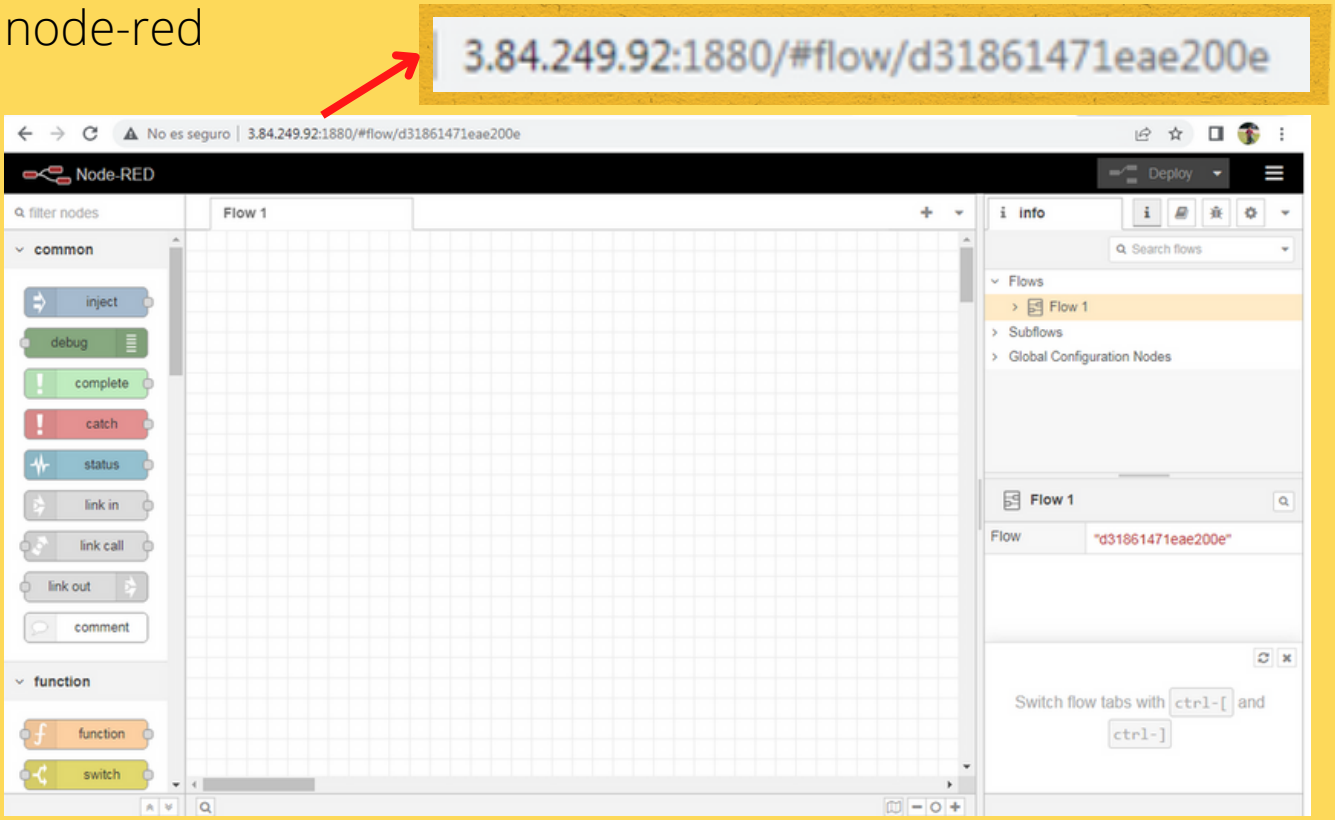
-SOLUCIÓN: agregar la política de seguridad
“TCP PERSONALIZADO PUERTO 1880”



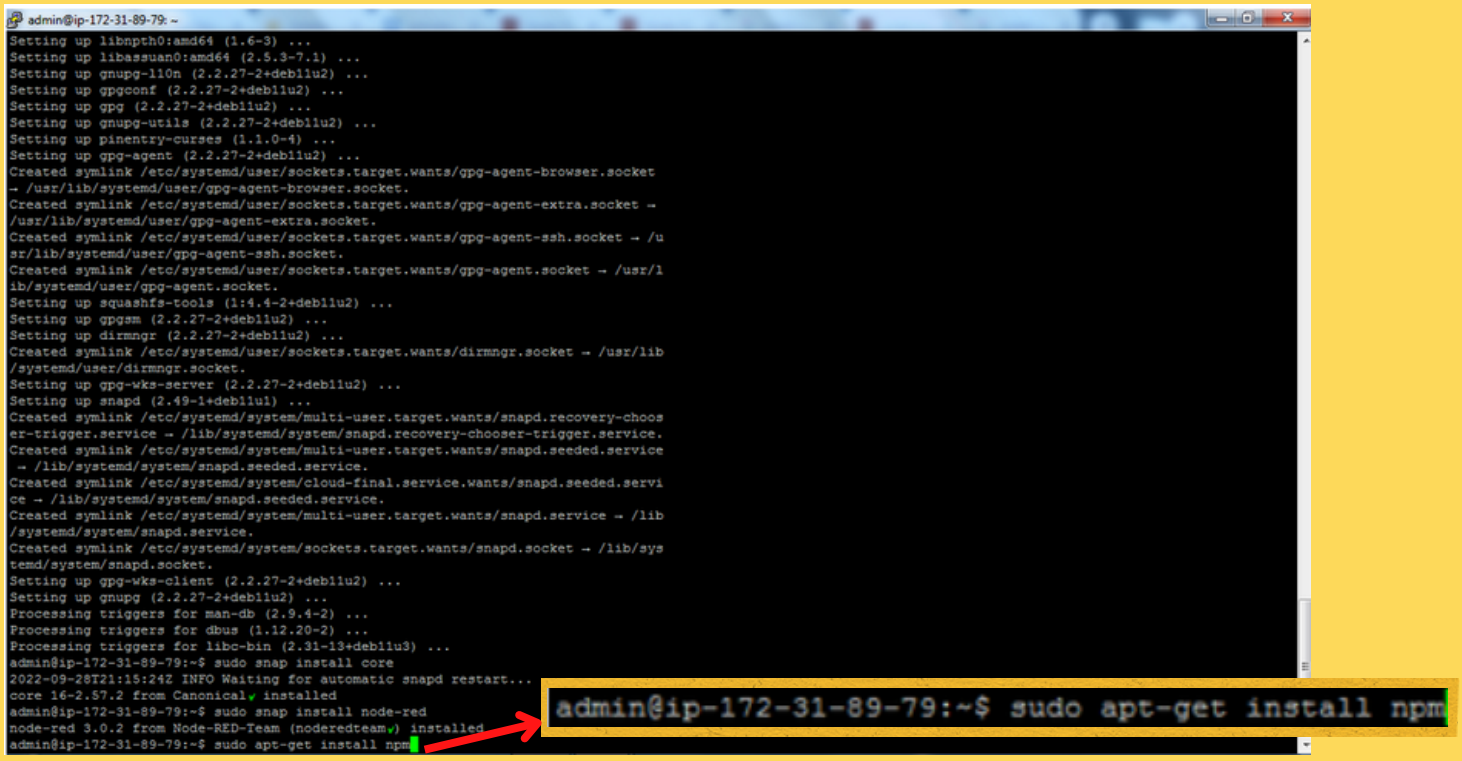
Se puede ver que se cambiaron correctamente las reglas de entrada del grupo de seguridad de la instancia.



Recargamos la página y estamos dentro del dashboard de node-red



3) Instalar el nodo “node-red-dashboard”.



Optamos por la opción de instalación por consola de comandos, por lo que primero debemos instalar npm () con el comando *$sudo apt-get install npm*

Cuando la instalación lo requiera, confirmamos que deseamos continuar

```
admin@ip-172-31-89-79: ~
  libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan6 libatk-bridge2.0-0 libatk1.0-0 libatk1.0-data libatomic1 libatspi2.0-0
  libauthen-sasl-perl libavahi-client3 libavahi-common-data libavahi-common3 libbinutils libc-ares2 libc-dev-bin libc-devtools libc6-dev libcairo-gobject2
  libcairo2 libcc1-0 libclone-perl libcolord2 libcrypt-dev libctf-nobfd0 libctf0 libcups2 libdata-dump-perl libdatrie1 libdconf1 libdeflate0 libdpkg-perl
  libdrm-amdgpu1 libdrm-common libdrm-intel1 libdrm-nouveau2 libdrm-radeon1 libdrm2 libencode-locale-perl libepoxy0 liberror-perl libfakeroot
  libfile-basedir-perl libfile-desktopentry-perl libfile-fcntllock-perl libfile-listing-perl libfile-mimeinfo-perl libfont-afm-perl libfontconfig1
  libfontenc1 libfribidi0 libgcc-10-dev libgd3 libgdbm-compat4 libgdk-pixbuf-2.0-0 libgdk-pixbuf2.0-bin libgdk-pixbuf2.0-common libgl1 libgl1-mesa-dri
  libglapi-mesa libglvnd0 libglx-mesa0 libglx0 libgomp1 libgraphite2-3 libgtk-3-0 libgtk-3-bin libgtk-3-common libharfbuzz0b libhtml-form-perl
  libhtml-format-perl libhtml-parser-perl libhtml-tagset-perl libhtml-tree-perl libhttp-cookies-perl libhttp-daemon-perl libhttp-date-perl
  libhttp-message-perl libhttp-negotiate-perl libio-html-perl libio-socket-ssl-perl libio-stringy-perl libipc-system-simple-perl libisl23 libitm1 libjbig0
  libjpeg62-turbo libjs-highlight.js libjs-inherits libjs-is-typedarray libjs-psl libjs-typedarray-to-buffer libjson-glib-1.0-0 libjson-glib-1.0-common
  liblcms2-2 libllvm11 liblocale-gettext-perl liblsan0 liblua5.3-0 liblwp-mediatypes-perl liblwp-protocol-https-perl libmailtools-perl libmpc3 libmpfr6
  libnet-dbus-perl libnet-http-perl libnet-smtp-ssl-perl libnet-ssleay-perl libnode-dev libnode72 libnsl-dev libpango-1.0-0 libpangocairo-1.0-0
  libpangoft2-1.0-0 libpciaccess0 libperl5.32 libpixman-1-0 libproxylv5 libquadmath0 librest-0.7-0 librsvg2-2 librsvg2-common libsensors-config libsensors5
  libsoup-gnome2.4-1 libsoup2.4-1 libssl-dev libstdc++-10-dev libtext-iconv-perl libthai-data libthai0 libtie-ixhash-perl libtiff5 libtimedate-perl
  libtirpc-dev libtry-tiny-perl libtsan0 libubsan1 liburi-perl libuv1-dev libvte-2.91-0 libvte-2.91-common libvulkan1 libwayland-client0 libwayland-cursor0
  libwayland-egl1 libwebp6 libwww-perl libwww-robotrules-perl libx11-protocol-perl libx11-xcb1 libxcb-dri2-0 libxcb-dri3-0 libxcb-glx0 libxcb-present0
  libxcb-randr0 libxcb-render0 libxcb-shape0 libxcb-shm0 libxcb-sync1 libxcb-xfixes0 libxcomposite1 libxcursor1 libxdamage1 libxfixes3 libxft2 libxi6
  libxinerama1 libxkbcommon0 libxkbfile1 libxml-parser-perl libxml-twig-perl libxml-xpathengine-perl libxmuu1 libxrandr2 libxrender1 libxshmfence1 libxtst6
  libxv1 libxxf86dga1 libxxf86vm1 libz3-4 linux-libc-dev make manpages-dev mesa-vulkan-drivers node-abbrev node-agent-base node-ajv node-ansi
  node-ansi-regex node-ansi-styles node-ansistyles node-aproba node-archy node-are-we-there-yet node-asap node-asn1 node-assert-plus node-asynckit
  node-aws-sign2 node-aws4 node-balanced-match node-bcrypt-pbkdf node-brace-expansion node-builtins node-cacache node-caseless node-chalk node-chownr
  node-clone node-color-convert node-color-name node-colors node-columnify node-combined-stream node-concat-map node-console-control-strings
  node-copy-concurrently node-core-util-is node-dashdash node-debug node-defaults node-delayed-stream node-delegates node-depd node-ecc-jsbn node-encoding
  node-err-code node-escape-string-regexp node-extend node-extsprintf node-fast-deep-equal node-forever-agent node-form-data node-fs-write-stream-atomic
  node-fs.realpath node-function-bind node-gauge node-getpass node-glob node-graceful-fs node-gyp node-har-schema node-har-validator node-has-flag
  node-has-unicode node-hosted-git-info node-http-signature node-https-proxy-agent node-iconv-lite node-iferr node-imurmurhash node-indent-string
  node-inflight node-inherits node-ini node-ip node-ip-regex node-is-typedarray node-isarray node-isexe node-isstream node-jsbn
  node-json-parse-better-errors node-json-schema node-json-schema-traverse node-json-stable-stringify node-json-stringify-safe node-jsonify node-jsonparse
  node-jsonstream node-jsprim node-leven node-lockfile node-lru-cache node-mime node-mime-types node-minimatch node-mkdirp node-move-concurrently node-ms
  node-mute-stream node-nopt node-normalize-package-data node-npm-bundled node-npm-package-arg node-npmlog node-number-is-nan node-oauth-sign
  node-object-assign node-once node-opener node-osenv node-p-map node-path-is-absolute node-performance-now node-process-nextick-args node-promise-inflight
  node-promise-retry node-promzard node-psl node-puka node-punycode node-qs node-read node-read-package-json node-readable-stream node-request node-resolve
  node-resolve-from node-retry node-rimraf node-run-queue node-safe-buffer node-semver node-set-blocking node-signal-exit node-slash node-spdx-correct
  node-spdx-exceptions node-spdx-expression-parse node-spdx-license-ids node-sshpk node-ssri node-string-decoder node-string-width node-strip-ansi
  node-supports-color node-tar node-text-table node-through node-tough-cookie node-tunnel-agent node-tweetnacl node-typedarray-to-buffer
  node-unique-filename node-universalify node-uri-js node-util-deprecate node-uuid node-validate-npm-package-license node-validate-npm-package-name
  node-verror node-wcwidth.js node-which node-wide-align node-wrappy node-write-file-atomic node-yallist nodejs nodejs-doc npm patch perl perl-modules-5.32
  perl-openssl-defaults shared-mime-info termit x11-utils x11-xserver-utils xdg-utils xkb-data
The following packages will be upgraded:
  libc6 libssl1.1 libtirpc-common libtirpc3
4 upgraded, 394 newly installed, 0 to remove and 48 not upgraded.
Need to get 177 MB of archives.
After this operation, 690 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

```
Do you want to continue? [Y/n] y
```

Una vez instalado npm utilizamos el siguiente comando para instalar node-red-dashboard:

$sudo npm install node-red-dashboard

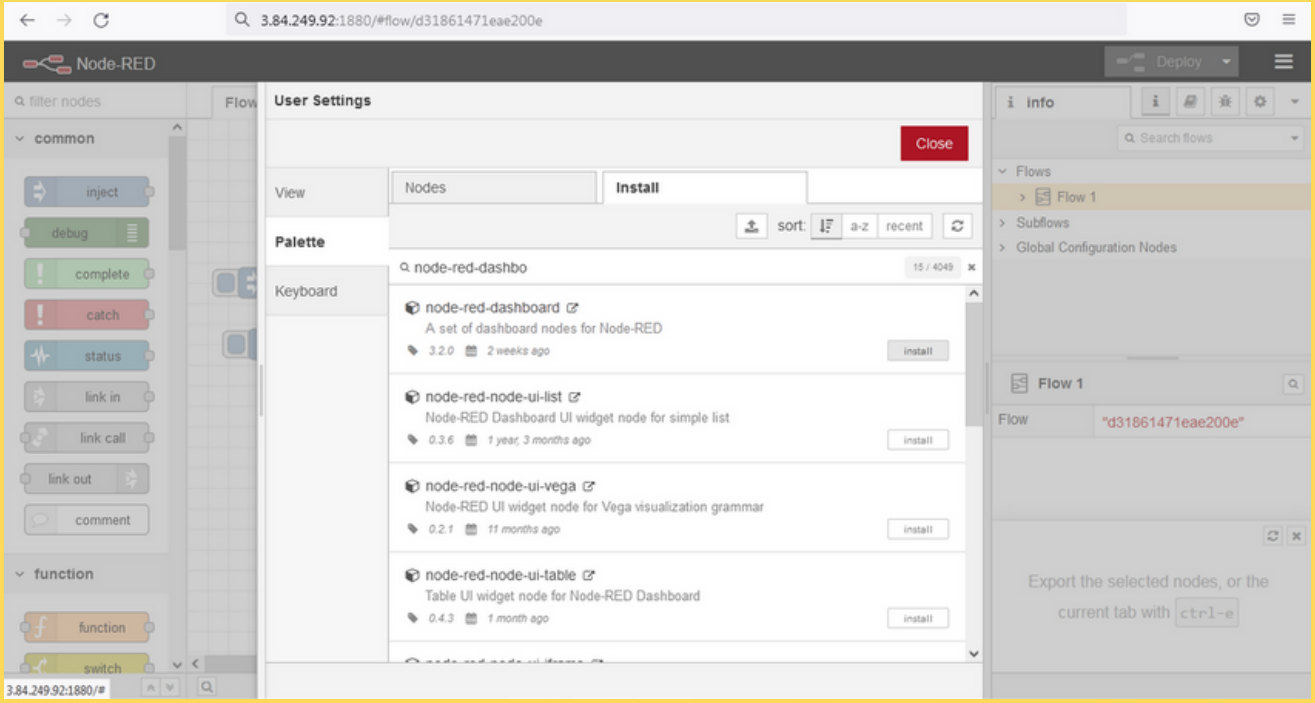
```
admin@ip-172-31-86-53: ~
Setting up node-normalize-package-data (3.0.0+~2.4.0-1) ...
Setting up node-npm-package-arg (8.1.0-1) ...
Setting up node-request (2.88.1-5) ...
Setting up node-npmlog (4.1.2-2) ...
Setting up node-https-proxy-agent (5.0.0-3) ...
Setting up node-cacache (15.0.5+~cs13.9.21-1) ...
Setting up node-read-package-json (3.0.0-1) ...
Setting up node-gyp (7.1.2-4) ...
Setting up npm (7.5.2+ds-2) ...
Setting up liblwp-protocol-https-perl (6.10-1) ...
Setting up libwww-perl (6.52-1) ...
Setting up libxml-parser-perl:amd64 (2.46-2) ...
Setting up libxml-twig-perl (1:3.52-1) ...
Setting up libnet-dbus-perl (1.2.0-1+b1) ...
Processing triggers for man-db (2.9.4-2) ...
Processing triggers for sgml-base (1.30) ...
Setting up libfontconfig1:amd64 (2.13.1-4.2) ...
Setting up fontconfig (2.13.1-4.2) ...
Regenerating fonts cache... done.
Setting up libxft2:amd64 (2.3.2-2) ...
Processing triggers for libglib2.0-0:amd64 (2.66.8-1) ...
Processing triggers for libc-bin (2.31-13+deb11u3) ...
Setting up libpango-1.0-0:amd64 (1.46.2-3) ...
Setting up libcairo2:amd64 (1.16.0-5) ...
Setting up libgd3:amd64 (2.3.0-2) ...
Setting up x11-utils (7.7+5) ...
Setting up libcairo-gobject2:amd64 (1.16.0-5) ...
Setting up glib-networking:amd64 (2.66.0-2) ...
Setting up libpangoft2-1.0-0:amd64 (1.46.2-3) ...
Setting up libpangocairo-1.0-0:amd64 (1.46.2-3) ...
Setting up libc-devtools (2.31-13+deb11u4) ...
Setting up libsoup2.4-1:amd64 (2.72.0-2) ...
Setting up librsvg2-2:amd64 (2.50.3+dfsg-1) ...
Setting up librsvg2-common:amd64 (2.50.3+dfsg-1) ...
Setting up libsoup-gnome2.4-1:amd64 (2.72.0-2) ...
Setting up librest-0.7-0:amd64 (0.8.1-1.1) ...
Setting up libgtk-3-0:amd64 (3.24.24-4+deb11u2) ...
Setting up libgtk-3-bin (3.24.24-4+deb11u2) ...
Setting up libvte-2.91-0:amd64 (0.62.3-1) ...
Setting up termit (3.1-1) ...
update-alternatives: using /usr/bin/termit to provide /usr/bin/x-terminal-emulator (x-terminal-emulator) in auto mode
Processing triggers for libc-bin (2.31-13+deb11u3) ...
Processing triggers for libgdk-pixbuf-2.0-0:amd64 (2.42.2+dfsg-1+deb11u1) ...
admin@ip-172-31-86-53:~$ sudo npm install node-red-dashboard
```

```
npm install node-red-dashboard
```

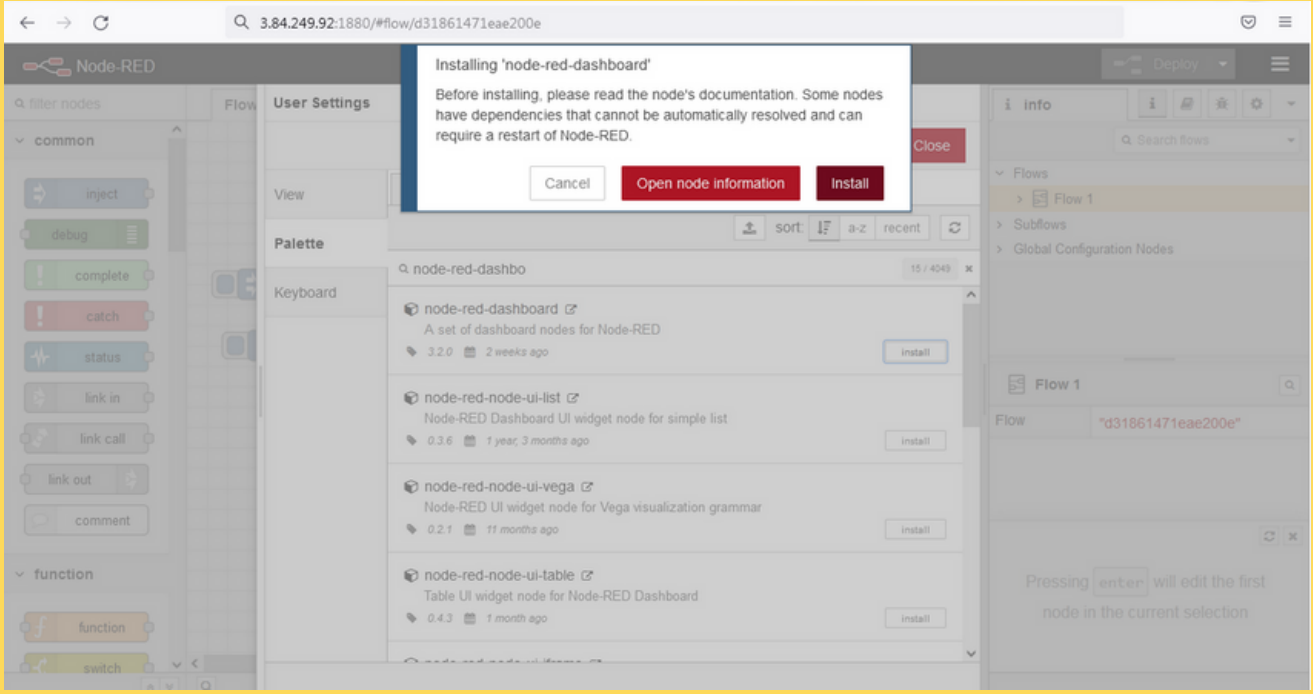
Una vez instalado obtuvimos la siguiente respuesta:

```
admin@ip-172-31-86-53:~$ sudo npm install node-red-dashboard

added 54 packages, and audited 55 packages in 4s

found 0 vulnerabilities
```

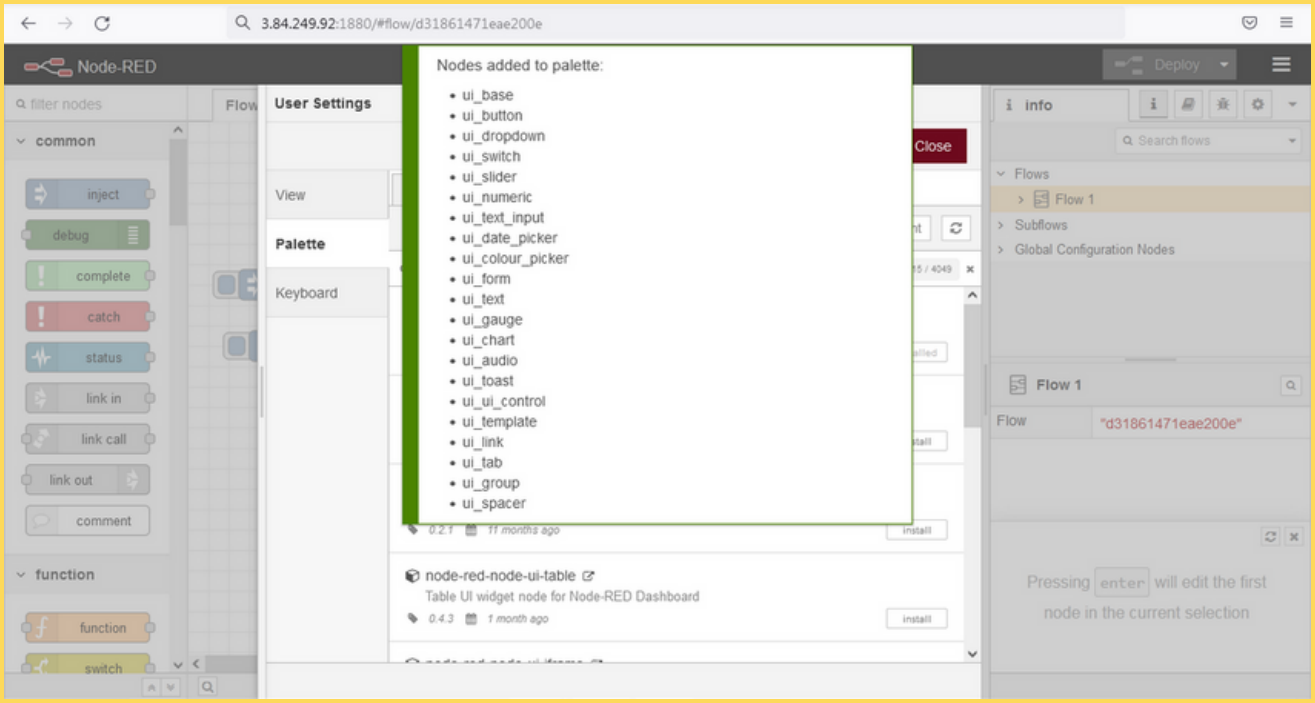
Lo realizado hasta este punto no terminaba de funcionar correctamente, ya que una vez en node-red, no podíamos seleccionar la paleta. Por lo que hicimos la prueba con otro navegador (cambiamos chrome por mozilla), pudiendo encontrar la siguiente solución al problema:



En el dashboard de node-red buscamos la parte de opciones (las tres líneas paralelas horizontales en la esquina superior derecha) y dentro de la configuración de la paleta elegimos "node-red-dashboard" y le damos 'install'.

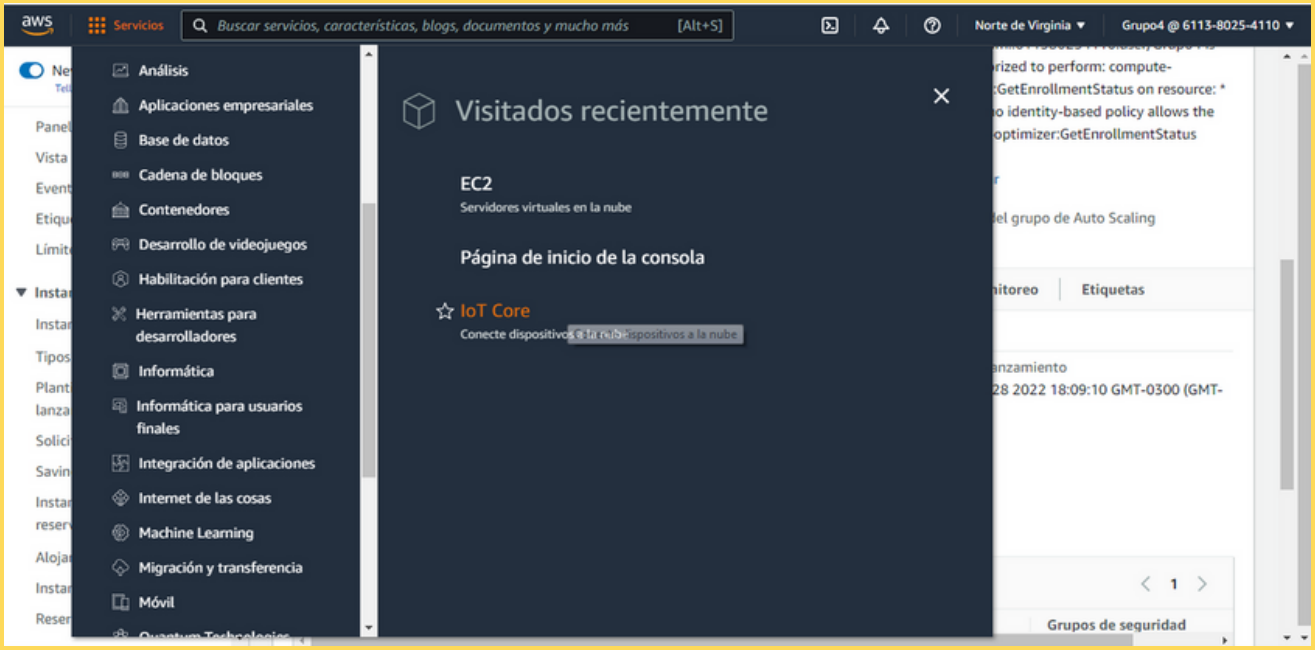


A partir de esta instalación aparecen los nuevos nodos para utilizar.



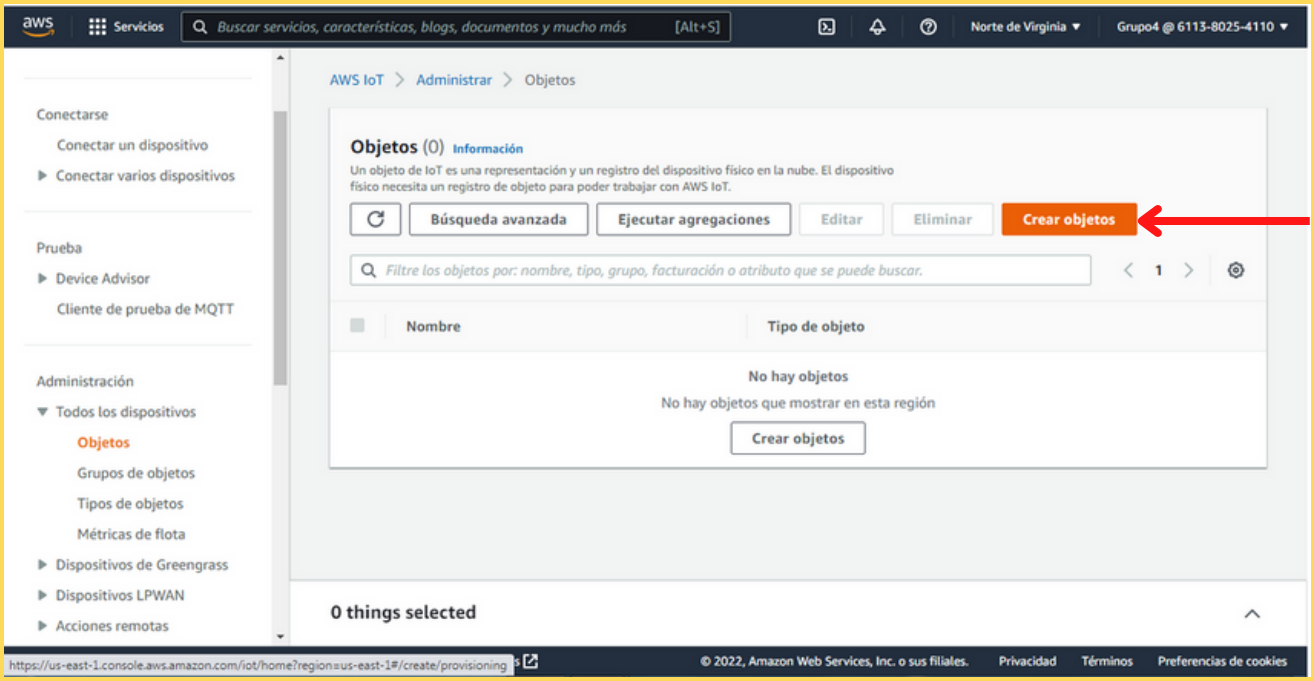
4) Registrar ambas instancias como objetos/dispositivos en AWS IoT para poder operar mediante el protocolo MQTT.

Regresamos al dashboard de Amazon y nos dirigimos a la sección IoT Core

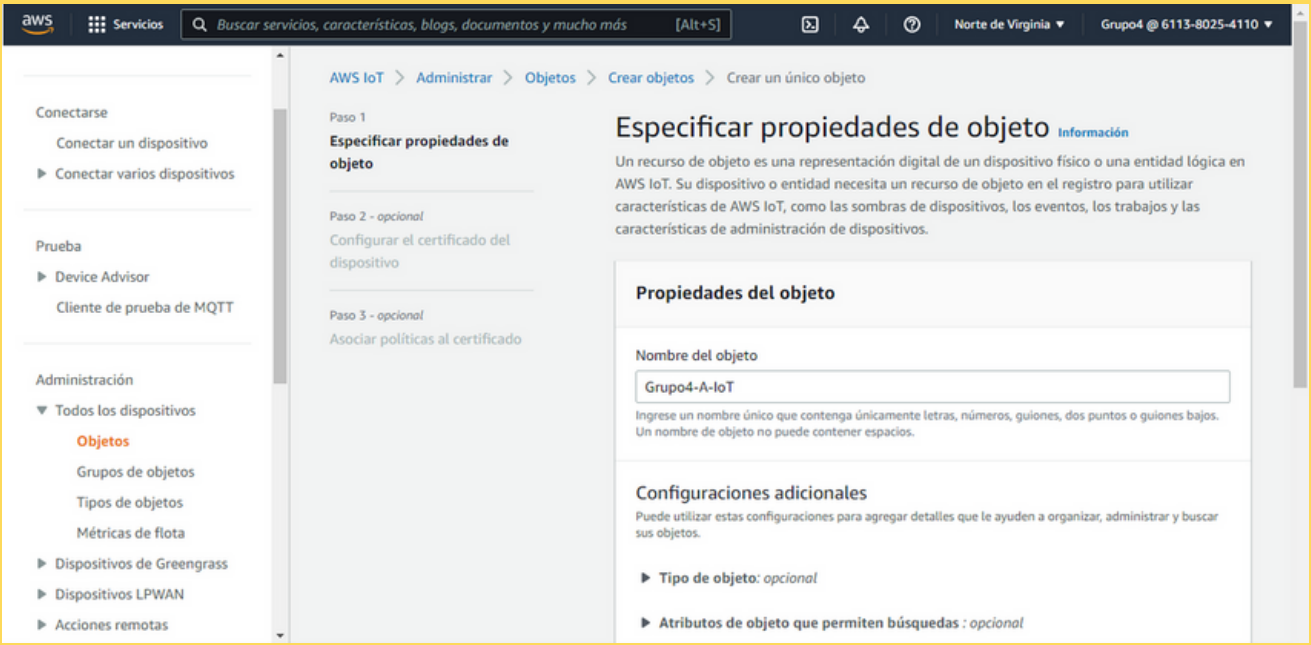


En este caso tuvimos un acceso rápido ya que AWS nos sugirió acceder directamente a IoT Core, sin embargo podemos acceder a ella siguiendo la ruta:
servicios → IOT → IOTCore
-elegimos monitoritorización → todos los dispositivos → objetos → Crear objeto

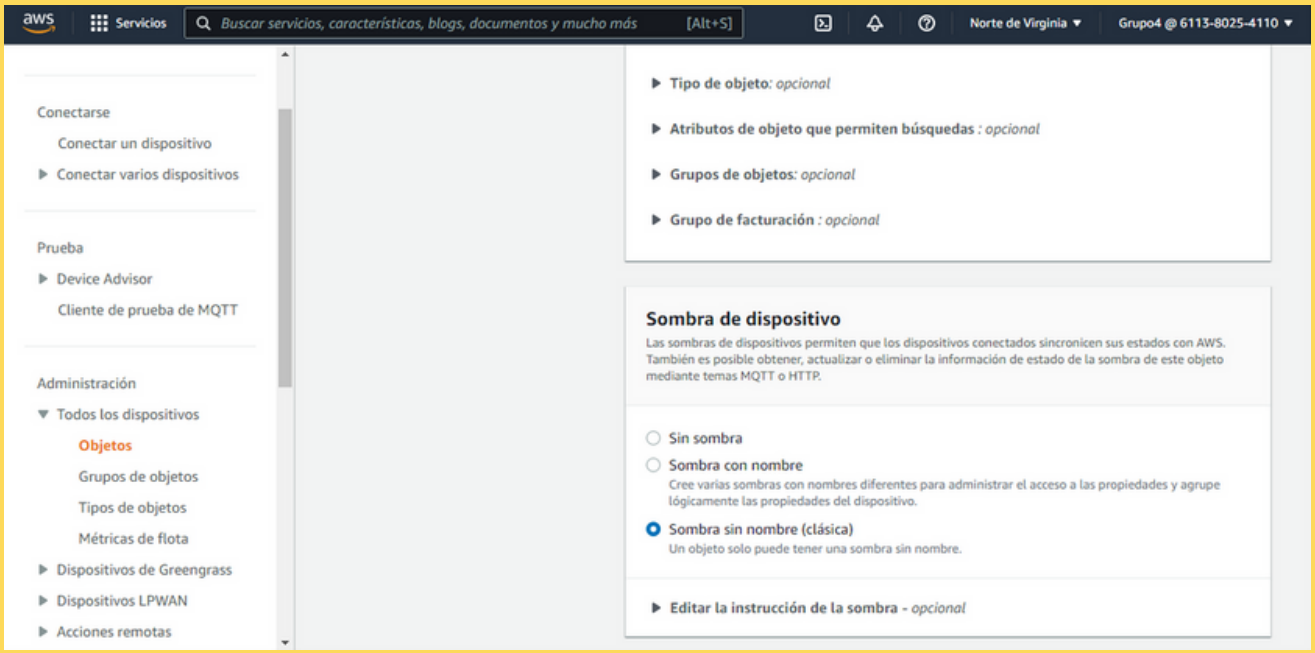
Una vez dentro de AWS IoT, hacemos click sobre el botón "Crear objetos"



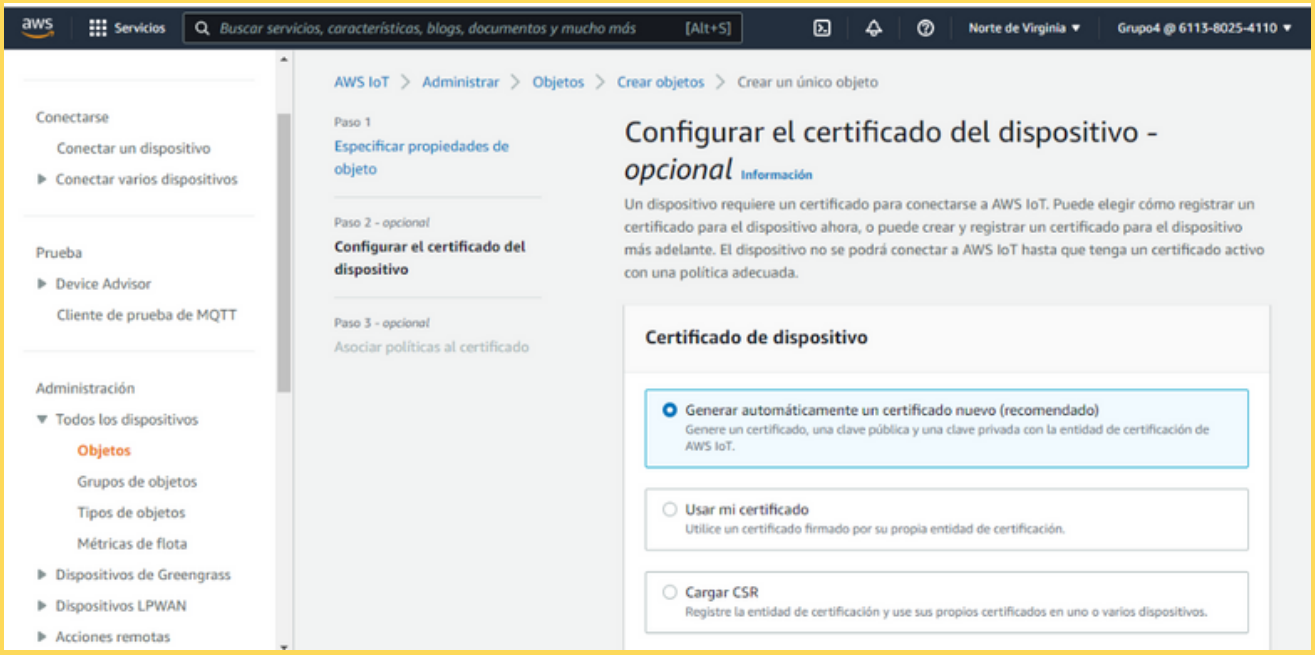
Seleccionamos la opción "crear un único objeto" y lo nombramos como "Grupo4-A-IoT"



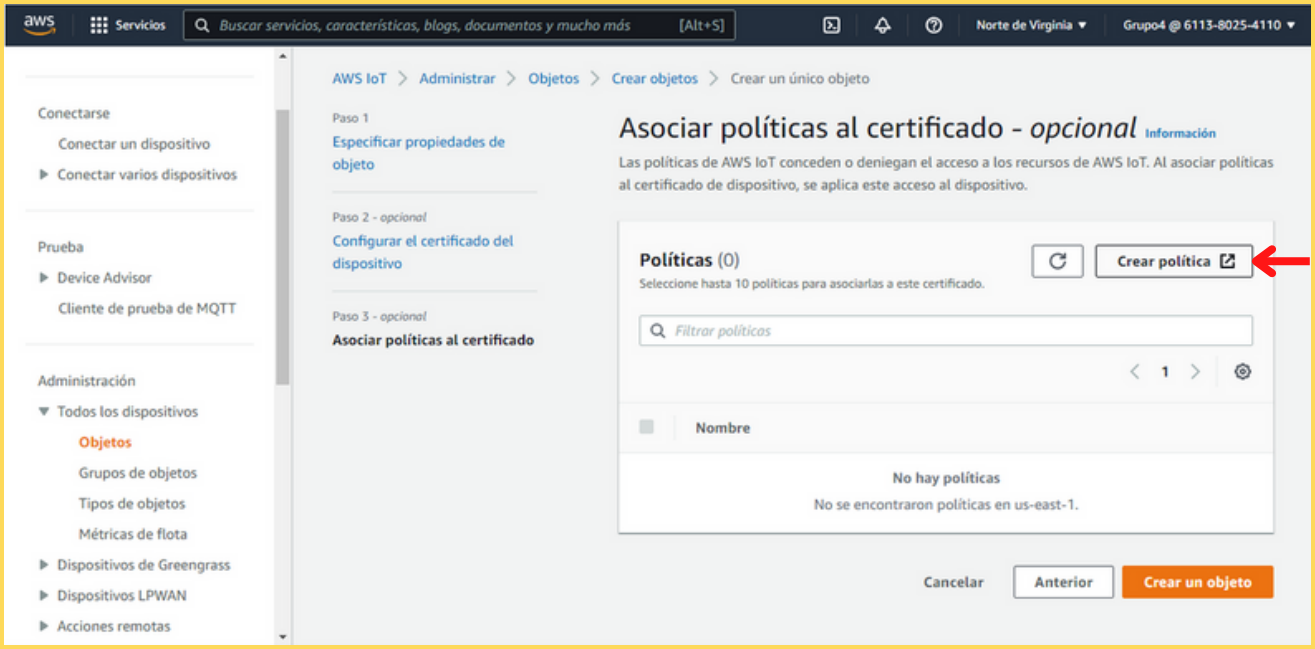
Elegimos el tipo de sombra del dispositivo
(por medio de MQTT tenemos gestión del estado del objeto)
seleccionamos **sombra sin nombre**



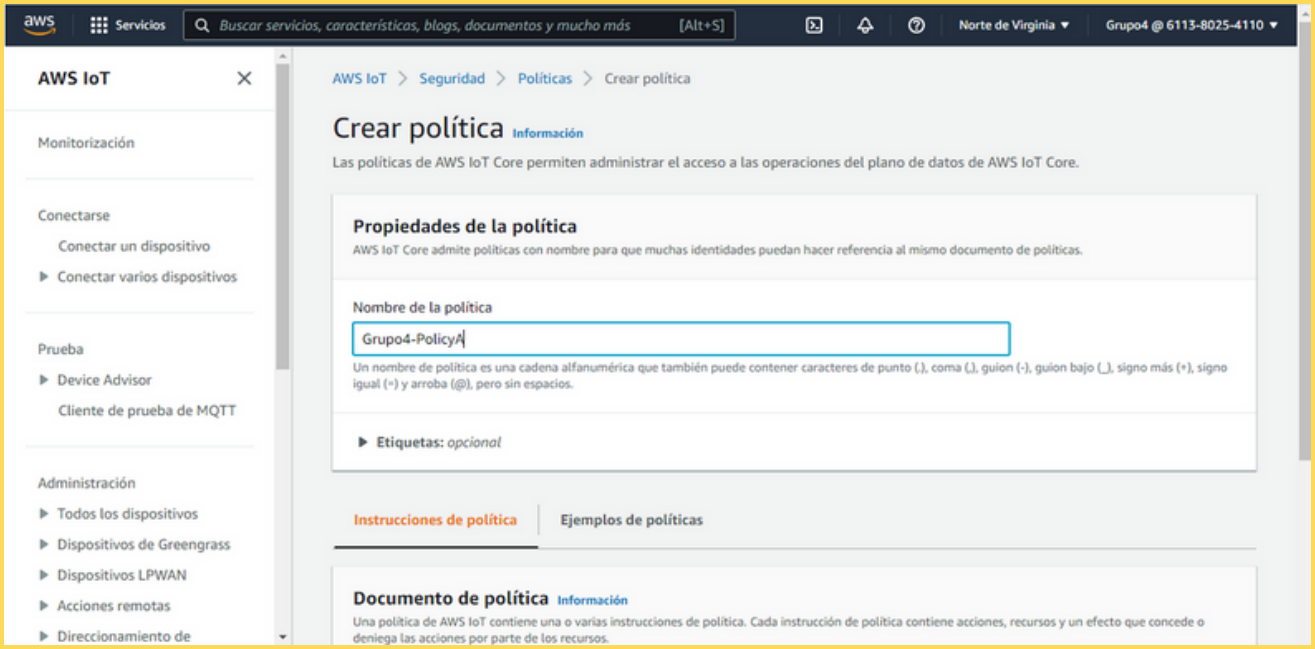
Como paso 2 de la creación del objeto, generamos un certificado para dar identidad al objeto.
Seleccionamos "Generar automáticamente un certificado nuevo"



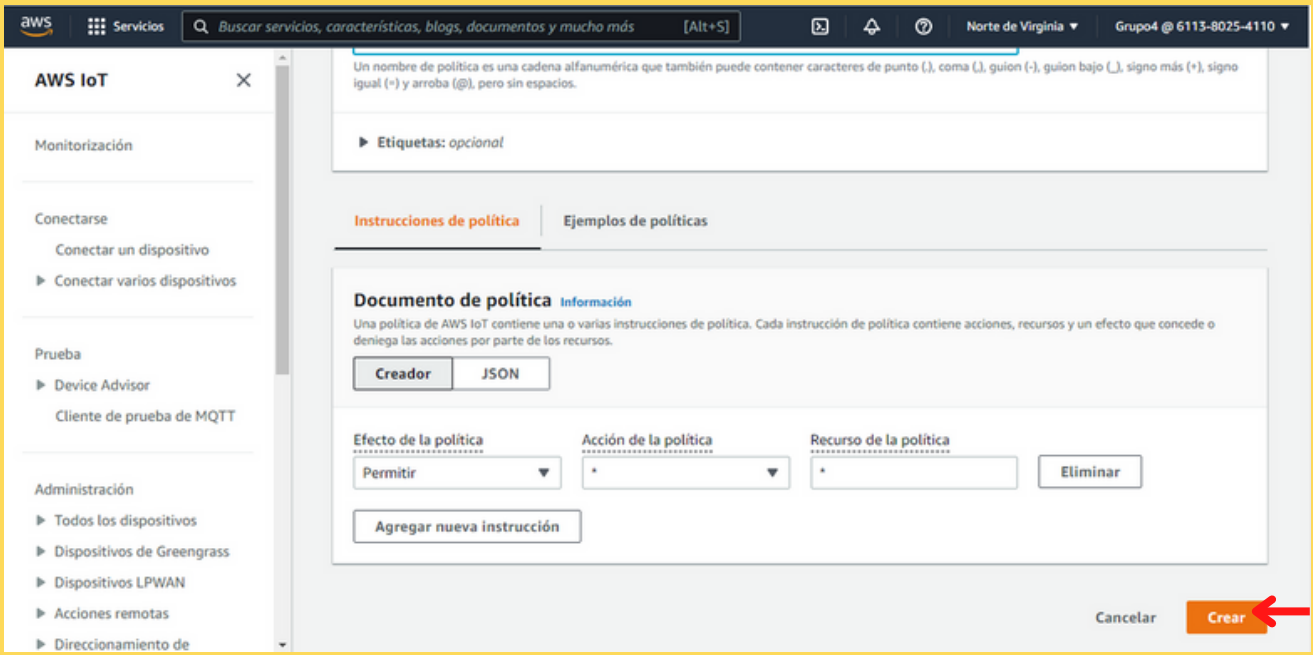
Creamos políticas:



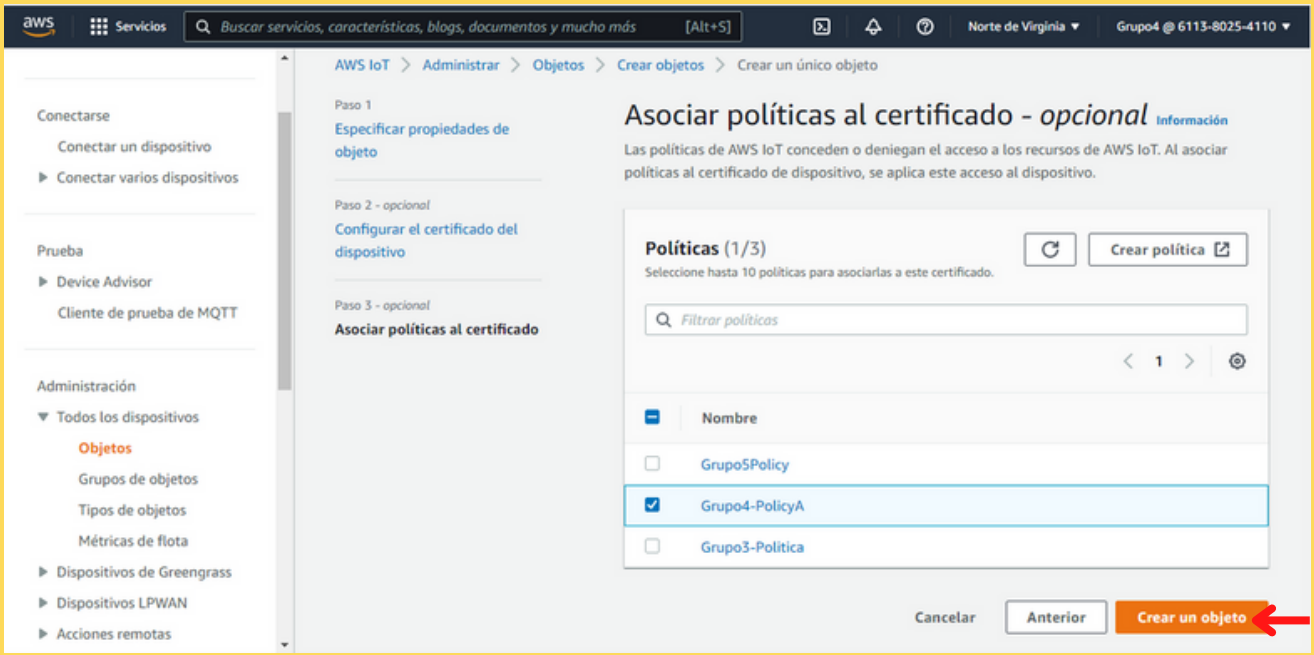
Creamos política “Grupo4-Policy” para dar seguridad al objeto.



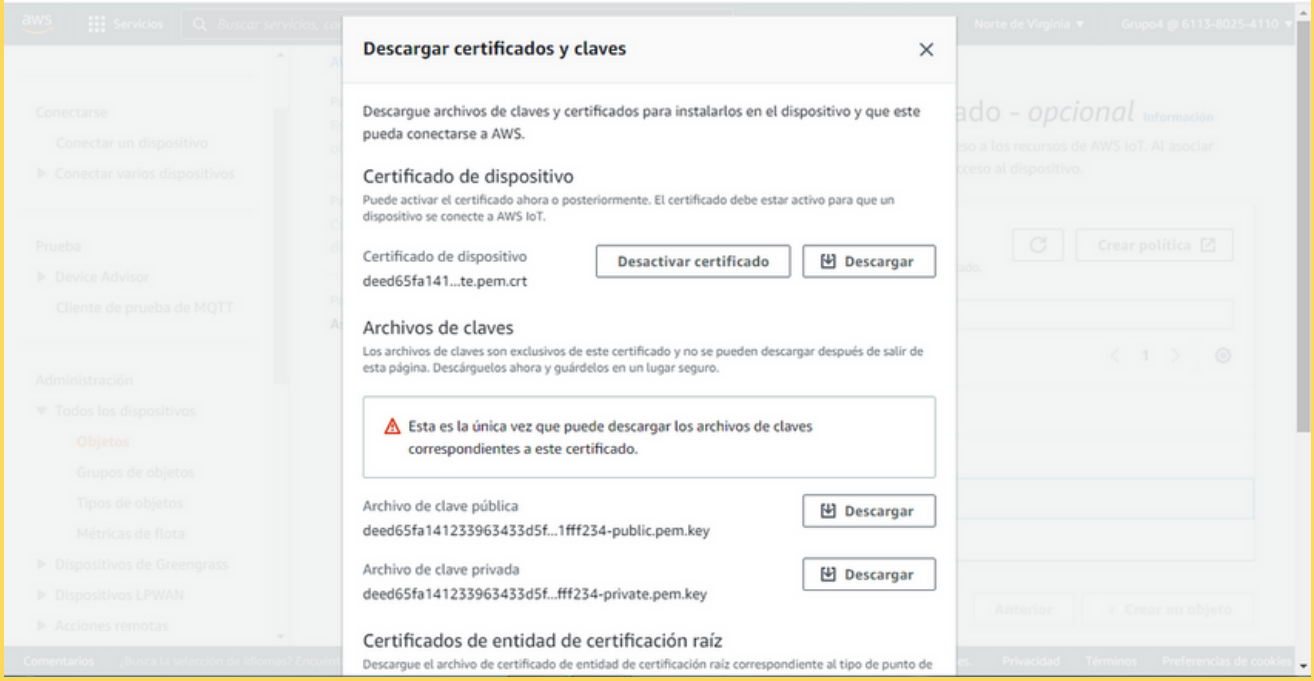
Configuramos el Documento de política de esta forma:
-Efecto de la política: permitir.
-Acción de la política: * (todo lo posible).
-Recurso de la políica: *(todo lo posible)
Hacemos click en Crear

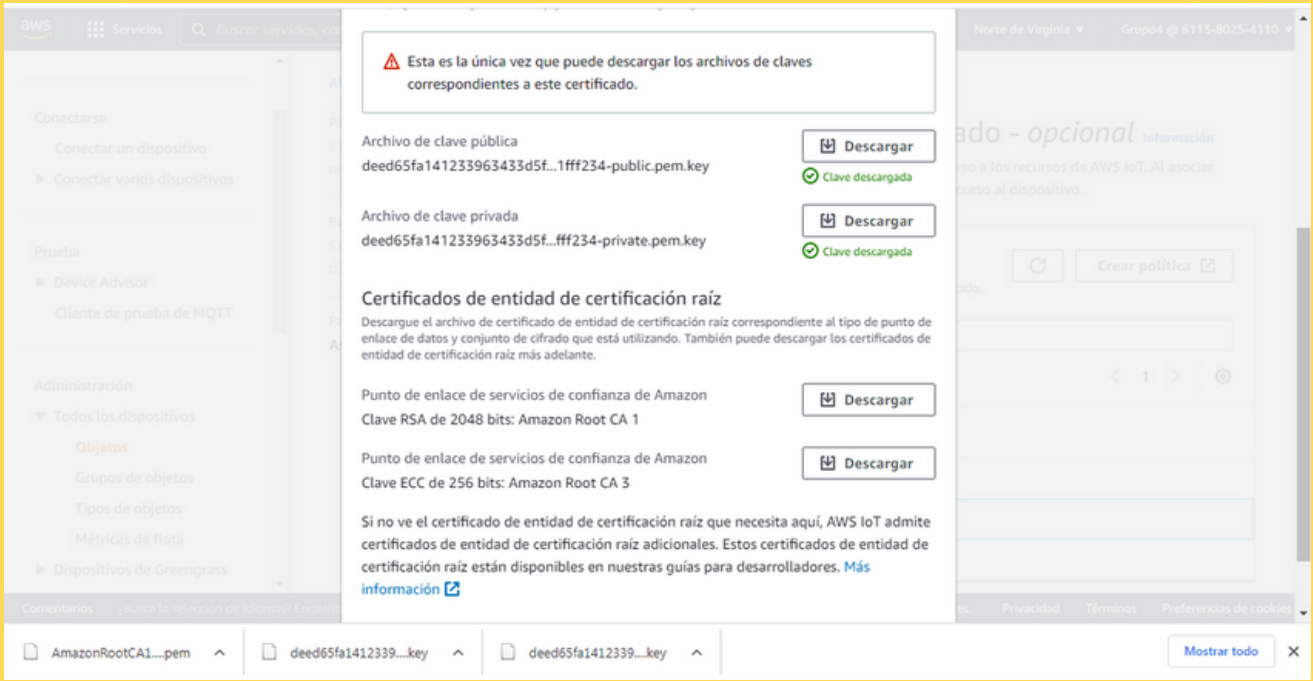


Volvemos a la solapa del objeto ,elegimos nuestra política (la asociamos al certificado) y damos click a Crear un objeto.

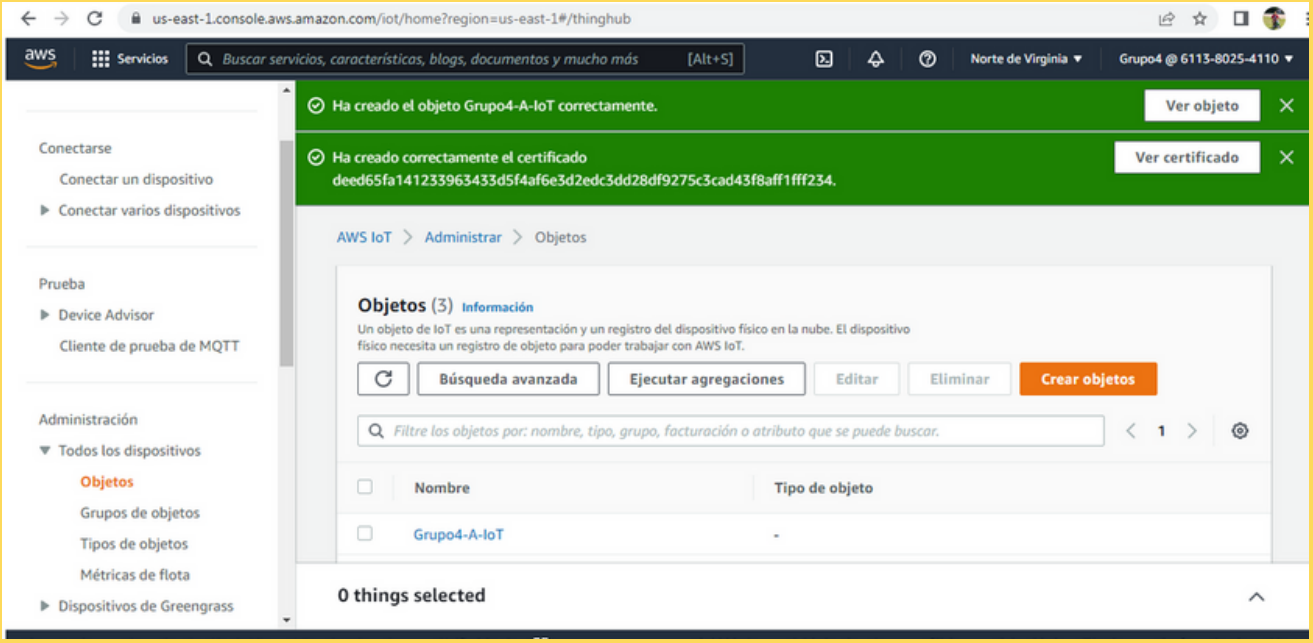


Descargamos el certificado, las claves públicas y privadas, y el certificado raiz (root) CA1:



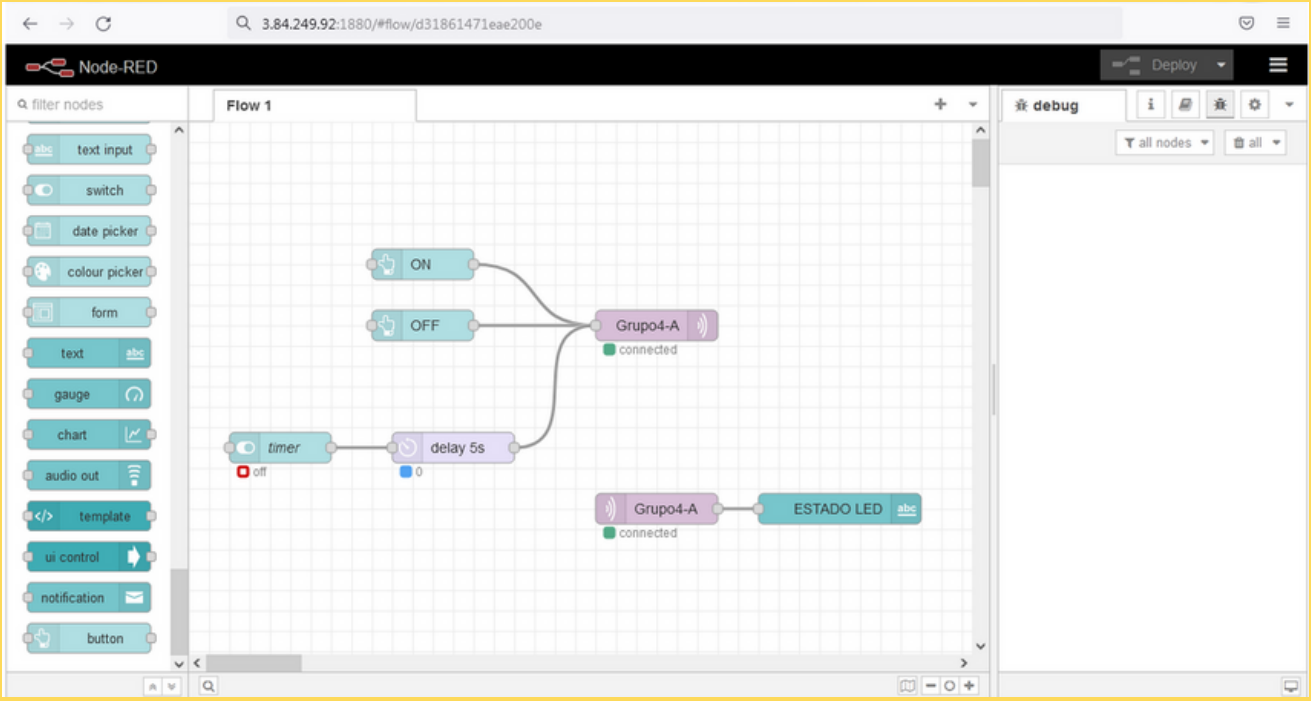


Finalmente hacemos click sobre "Listo" para continuar y obtenemos el siguiente mensaje:

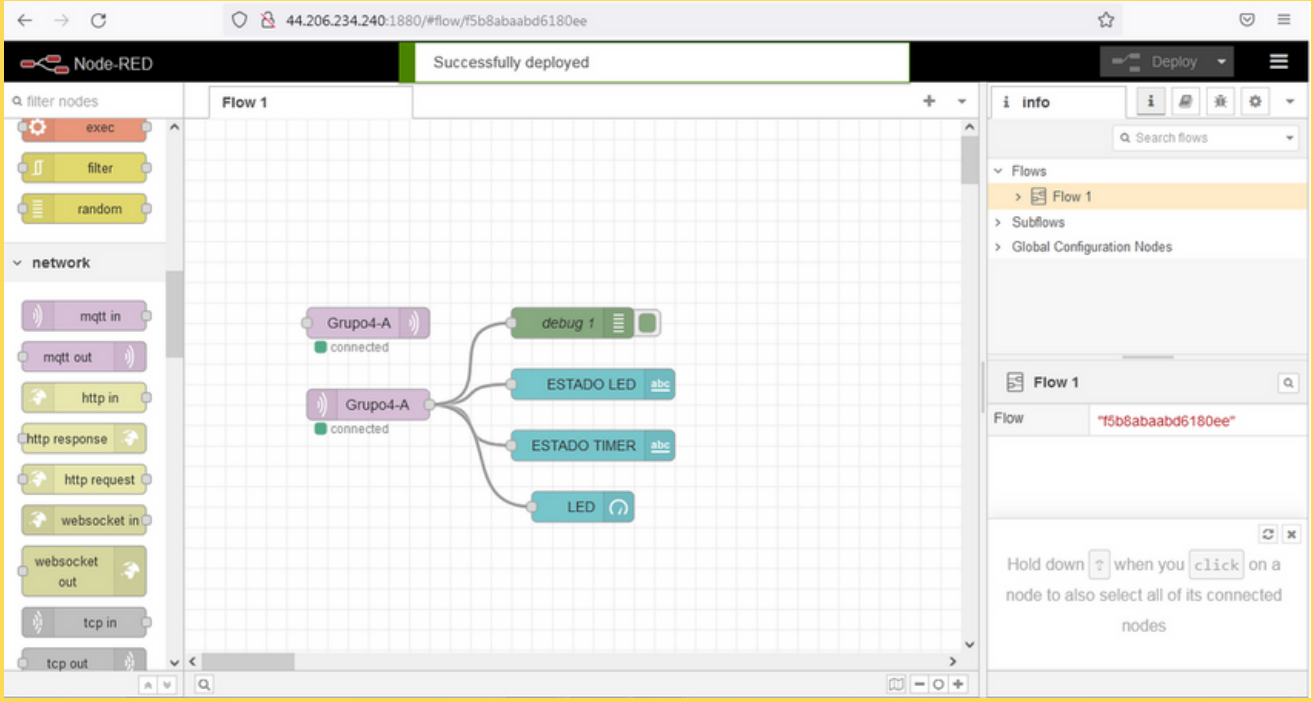


5) Desarrollar, en la instancia A, la solución de un "CONTROL REMOTO" para gestionar la iluminación de un ambiente: botón "ENCENDIDO/ON", botón "APAGADO/OFF", informe de estado de la luminaria y funcionalidad "TIMER de 5s" para que activada dicha funcionalidad, al encender la luminaria se apague a los 5 segundos.

Primero ponemos los nodos en node-red y luego comenzamos a configurarlos

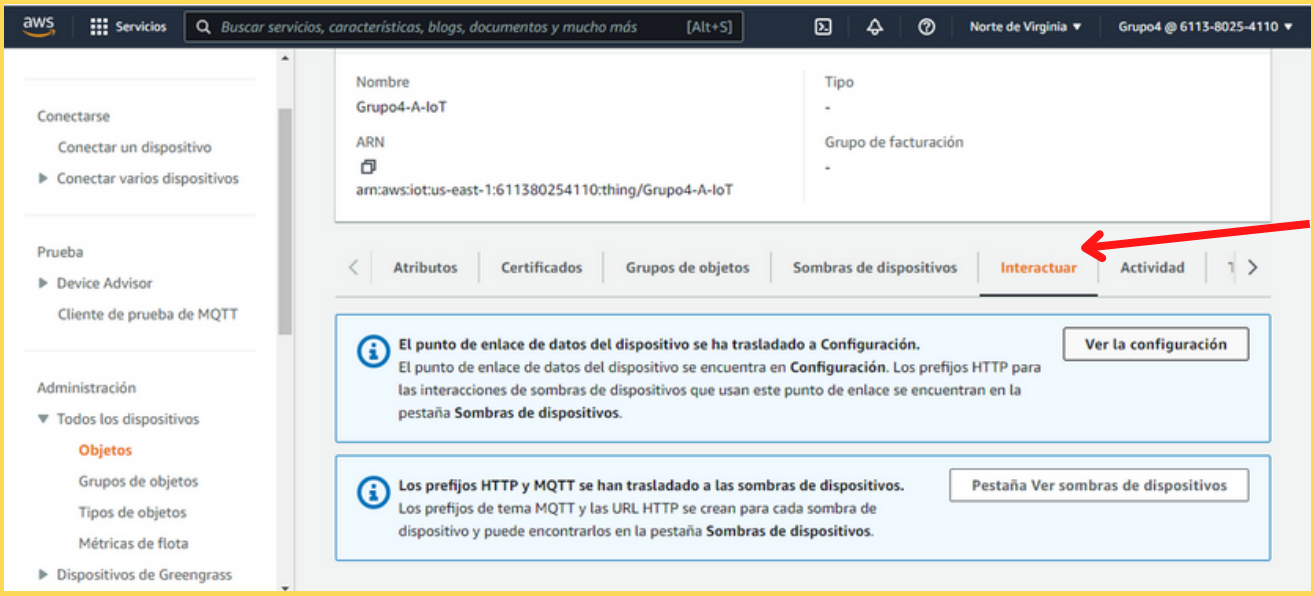


6) Desarrollar, en la instancia B, la solución para controlar la "ILUMINACIÓN" de un ambiente simulando el "ENCENDIDO/APAGADO" de la luminaria, informando el estado de la "LUMINARIA" y el estado del "TIMER".

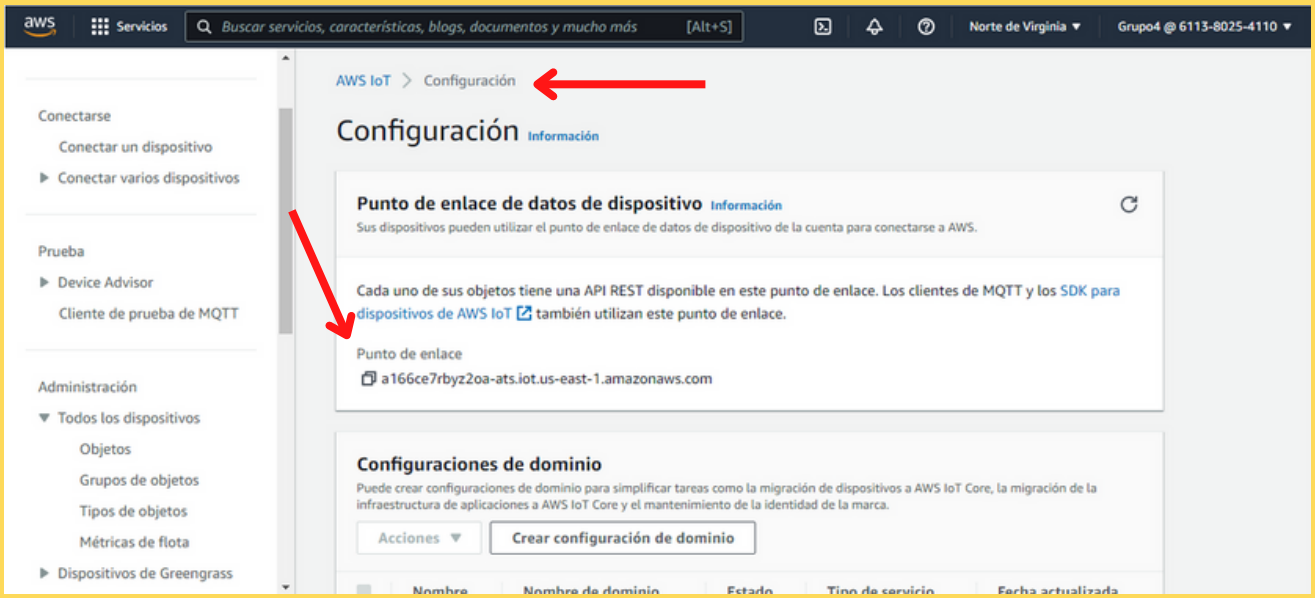


7) Las instancias A y B deben comunicarse por medio de MQTT utilizando el bróker de AWS IoT.

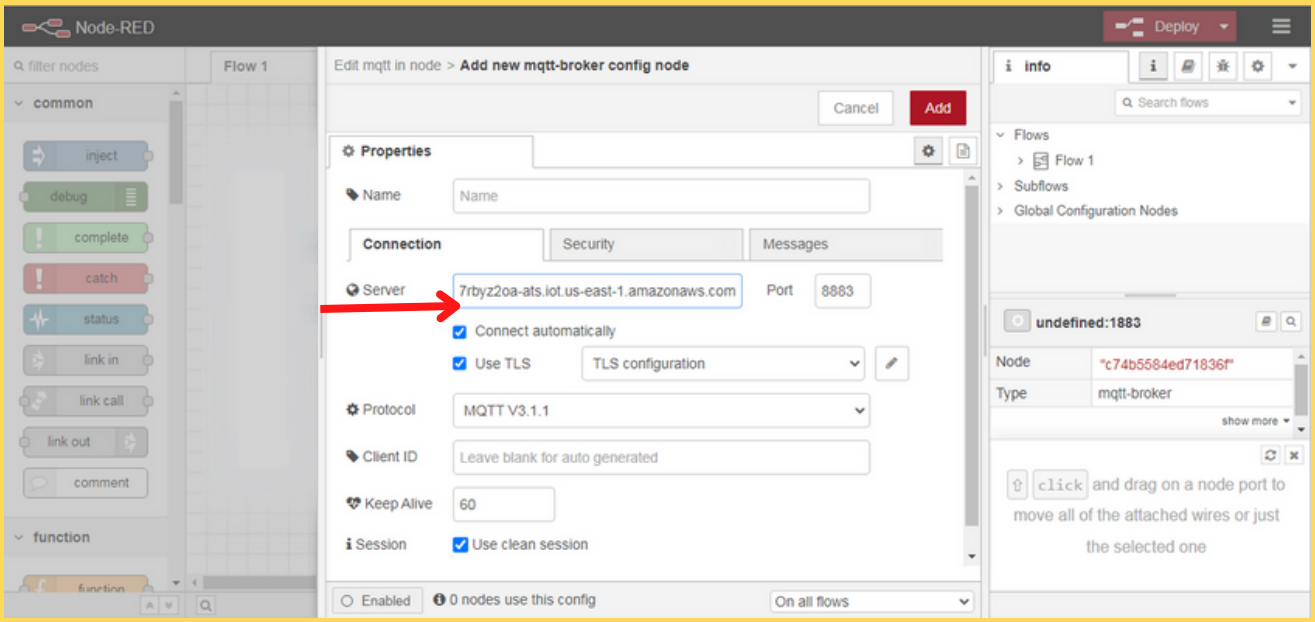
Para comenzar la configuración necesitaremos el punto de enlace de nuestros objetos por lo que vamos AWS y dentro del objeto vamos a la solapa Interactuar:



Donde en la sección de "Configuración" podremos copiar el Punto de enlace.

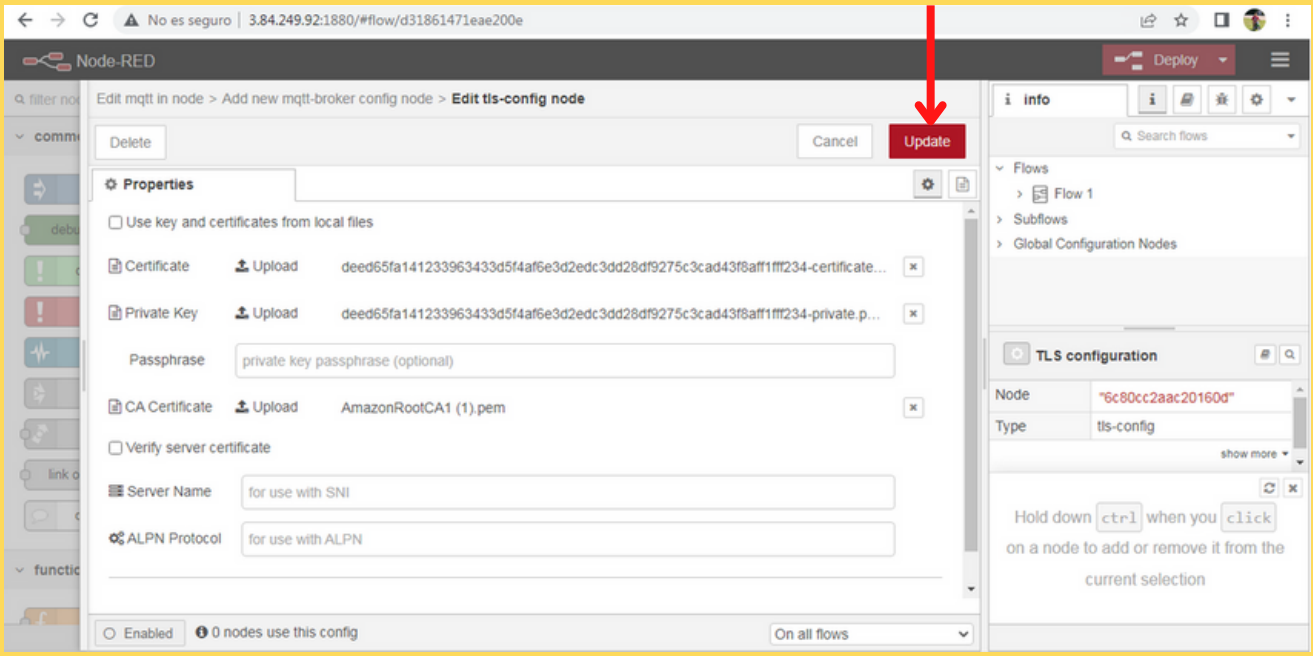


Una vez obtenido el punto de enlace, volvemos al dashboard de node-red para configurar los nodos mqtt

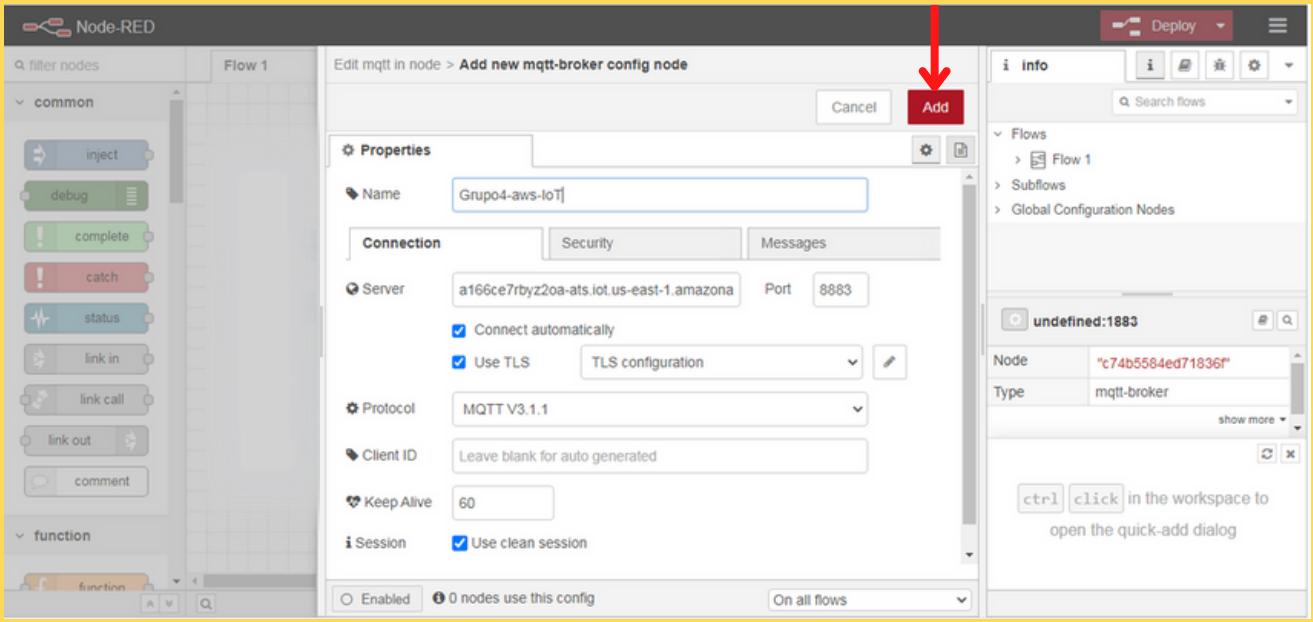


Pegamos el punto de enlace donde dice Server, en PORT ponemos 8883, tildamos 'use TLS' y 'connect automátically'.

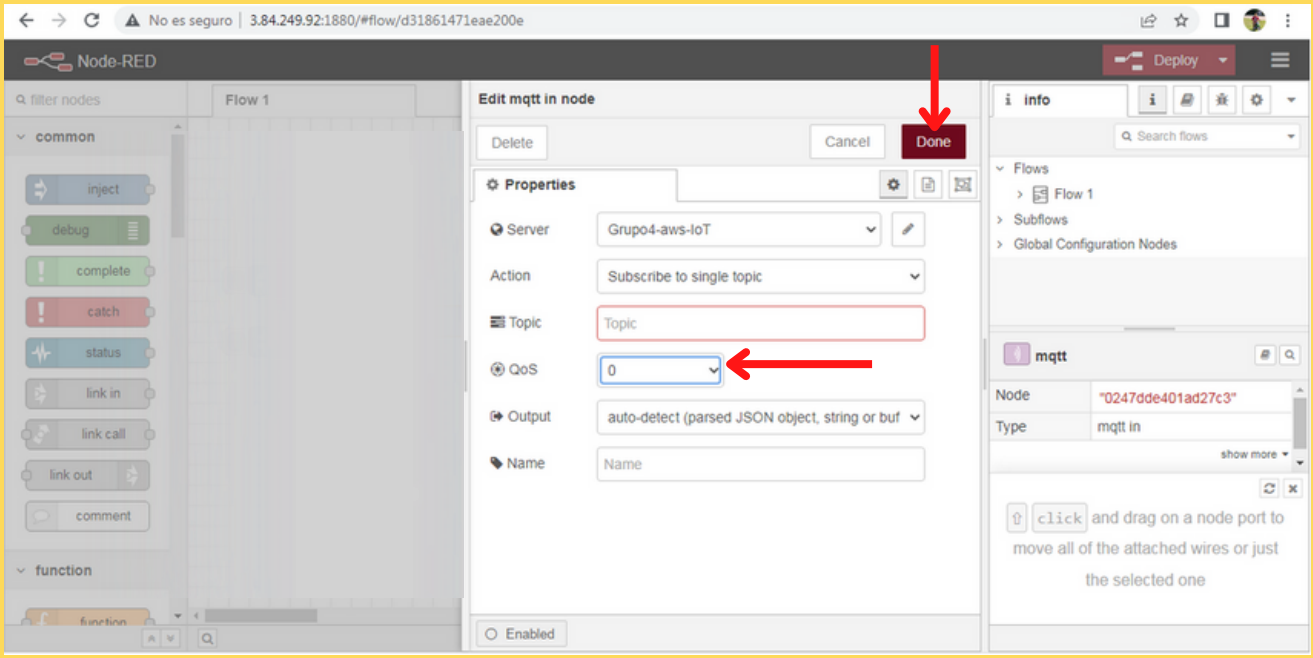
Vamos al lápiz, cargamos el certificado, clave privada y CA certificate y destildamos donde dice 'Verify server certificate'. Por último presionamos Update.



Presionamos en Add:



Configuramos: QOS 0 (cero) y presionamos "Done":

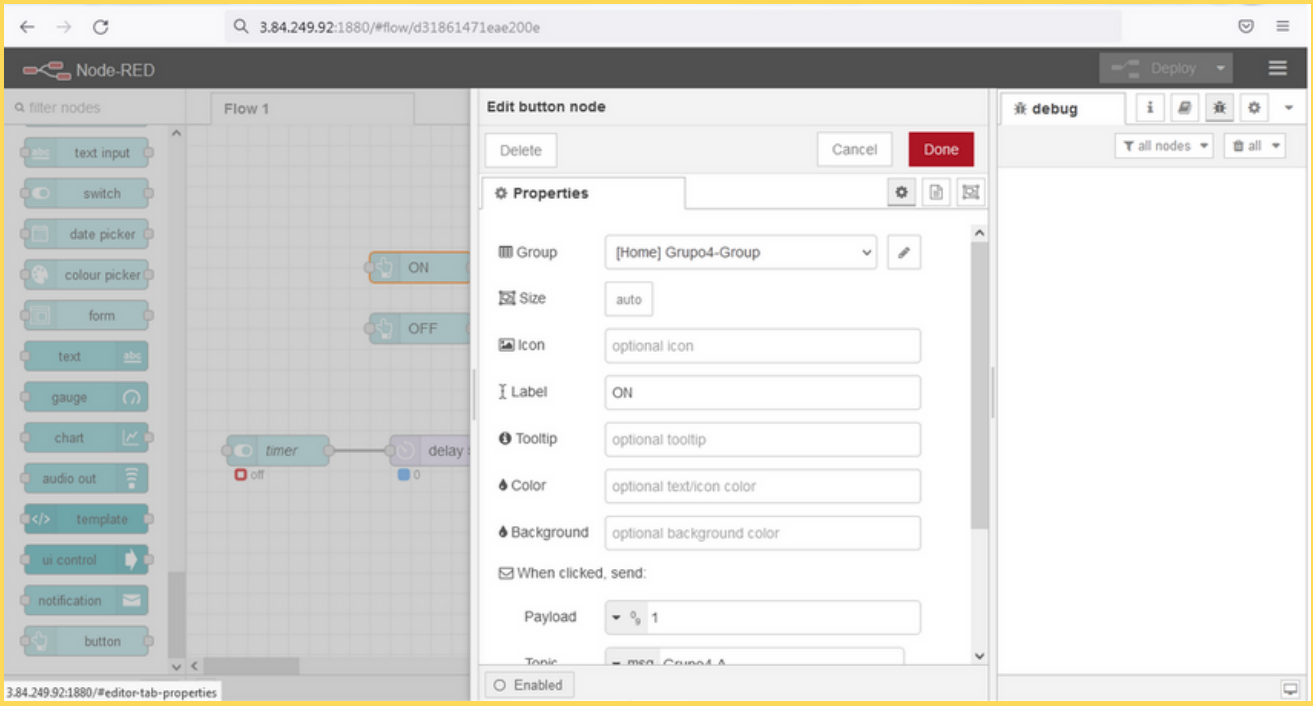


Configuración Instancia A:

Se asigno un 'Group' en común a cada nodo del dashboard.

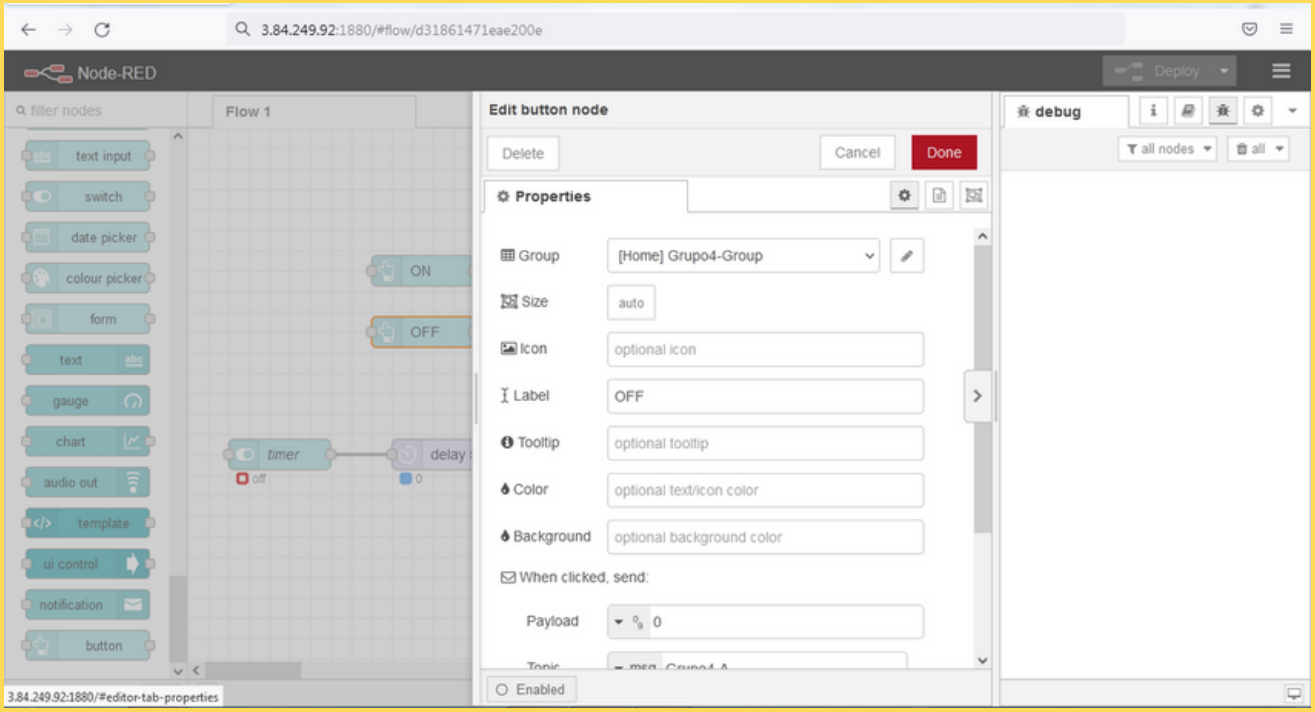
<u>Configuración del botón ON:</u>

- Se asigno el 'Label' para que se vea en el nodo.
- Y se asigno el 'Payload' con el valor 1, para que se vea en el otro nodo el 1 cuando se presiona el botón ON.



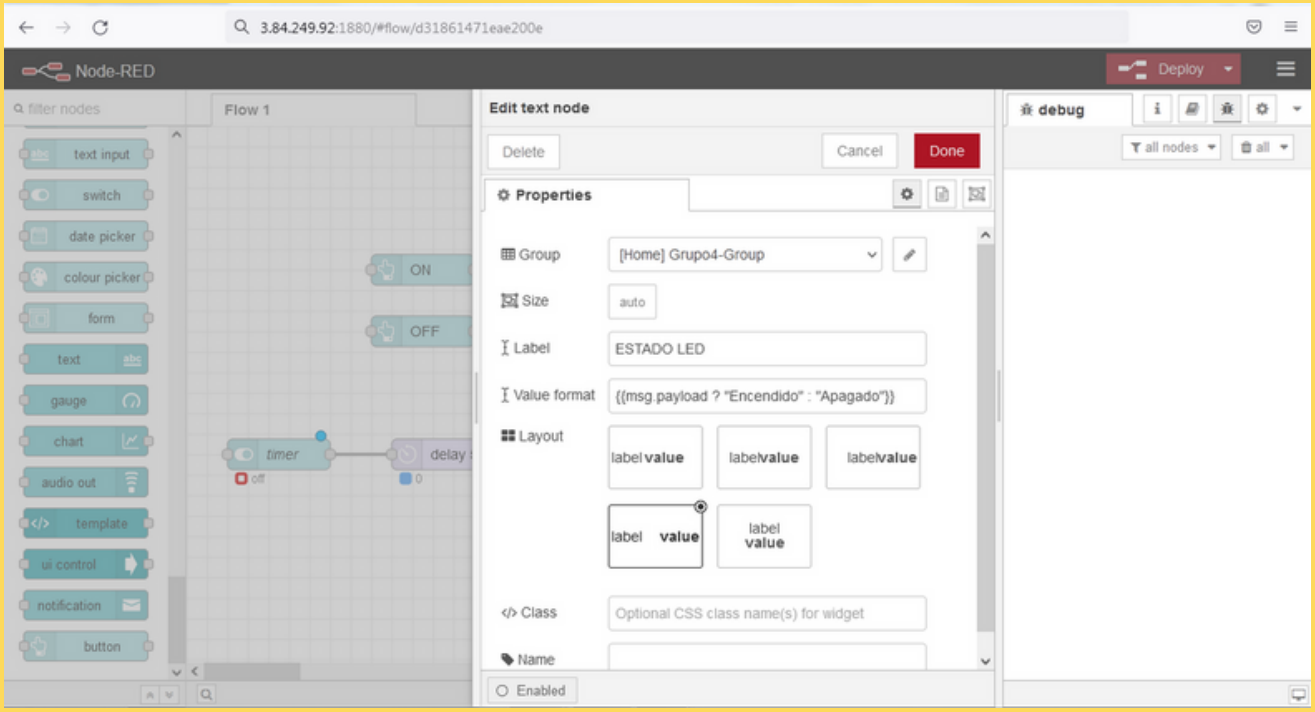
Configuración del botón OFF:

- Al igual que en el boton ON, se le asigno el mismo 'Group', el valor del 'Label' para que se vea en el nodo, y el 'Payload' con valor 0, para que se vea en el dashboard.



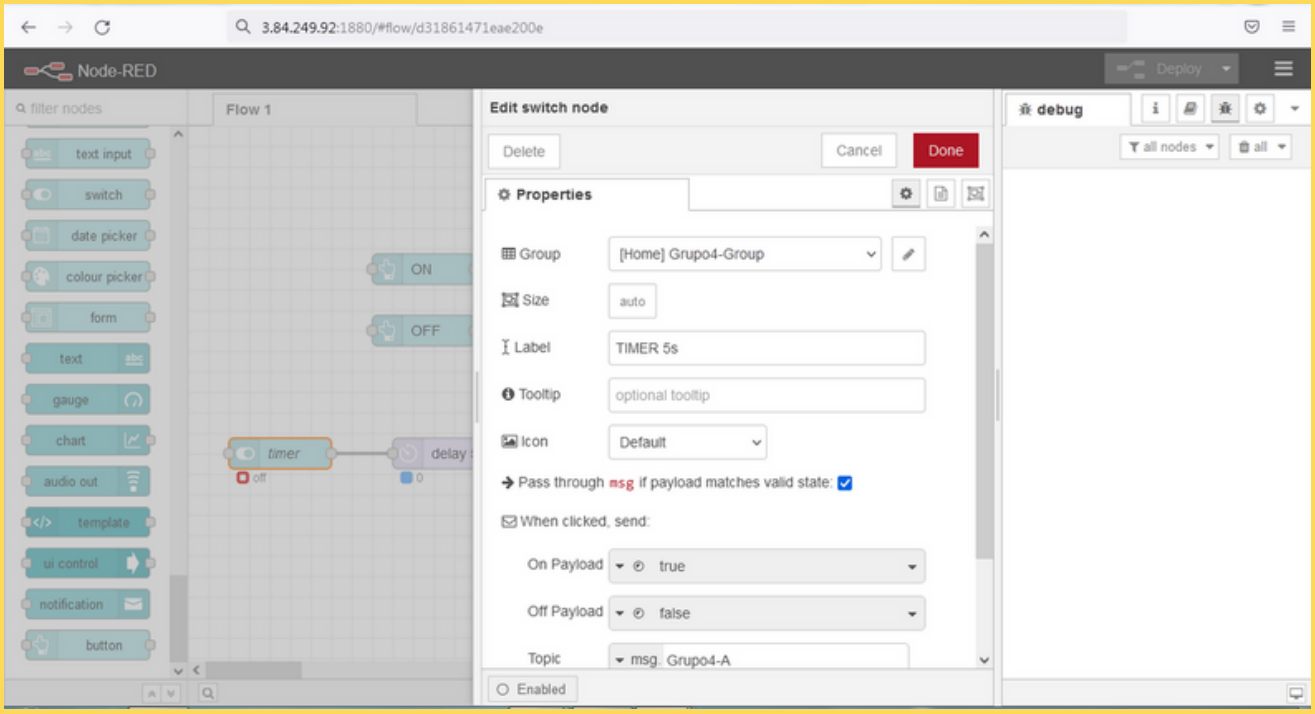
Configuración nodo estado del led:

- Se asigno el 'Group', el 'Label', y en el valor que toma utilizamos un codigo para que cuando el valor sea 1, aparezca la palabra "Encendido" en el dashboard, y "Apagado" cuando el valor es 0.



Configuración del switch del Timer:

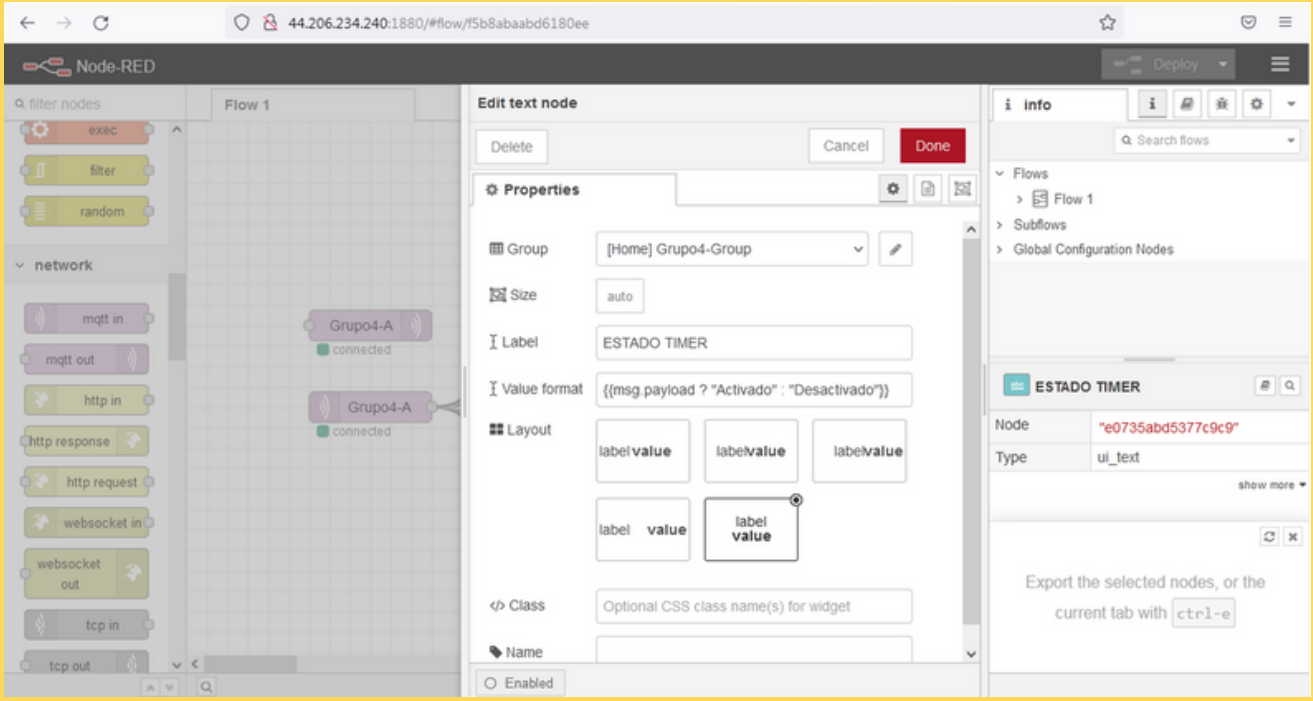
- es un timer de 5 segundos, por defecto, y no pudimosconfigurar bien los valores de 'On Payload' para que haga lo que deberia.



Configuración Instancia B:

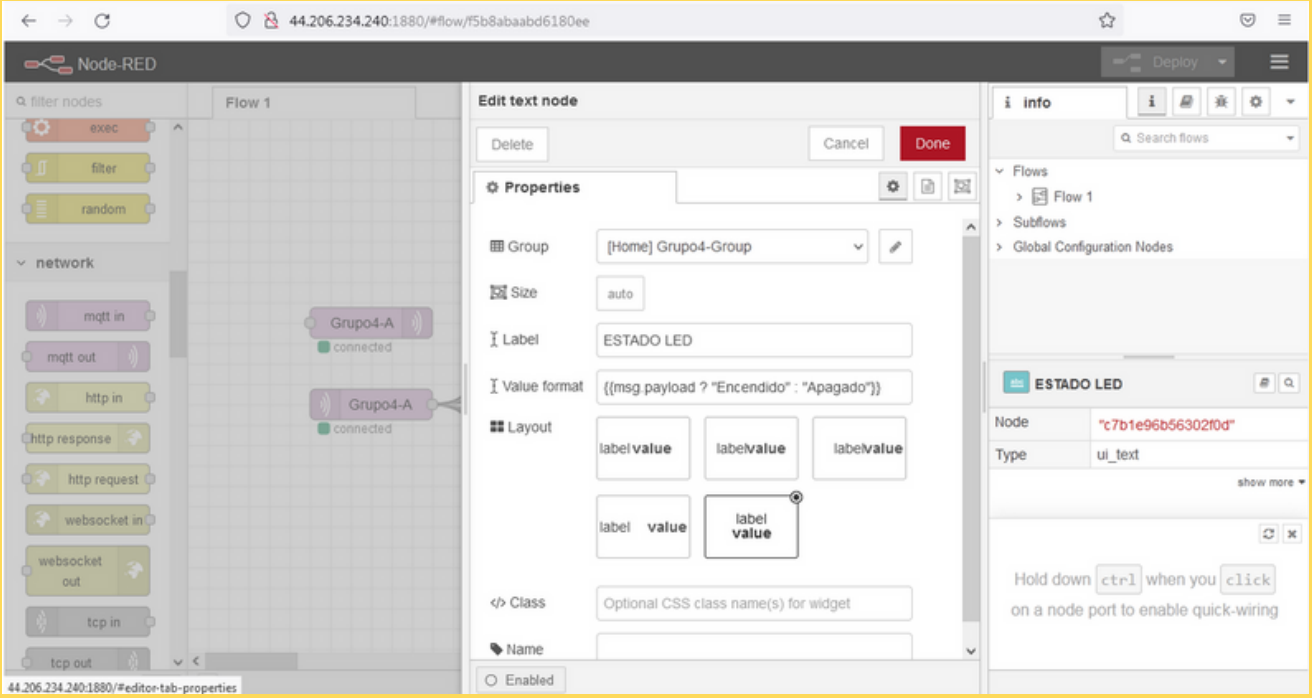
Configuración nodo estado del timer:

- No pudimos configurarlo de manera correcta, ya que deberia mostrar "Activado", cuando esta activo el switch, y "Desactivado" cuando se desactiva, y en lugar de eso muestra dependiendo de la variable payload, que es la del ON y el OFF.



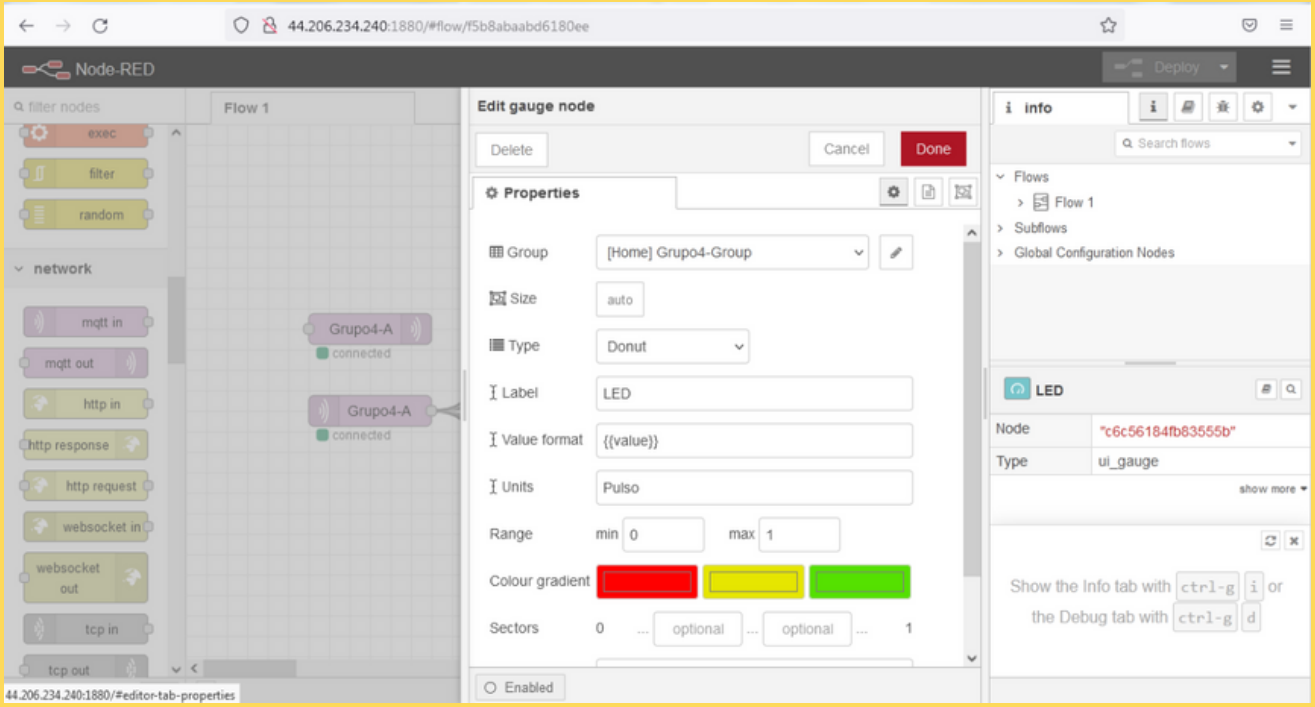
Configuración nodo estado del led:

- Se configuro de la misma forma que el nodo estado del led de la otra instancia.

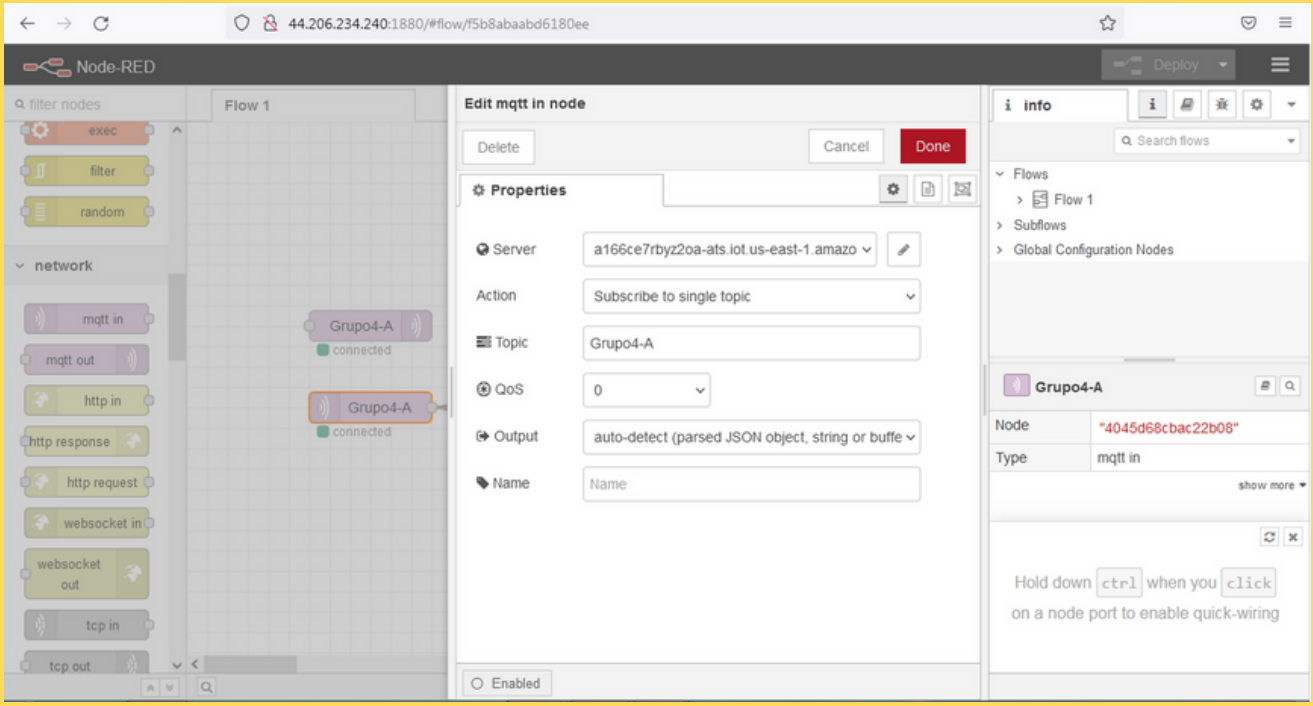


Configuración nodo que representa al led:

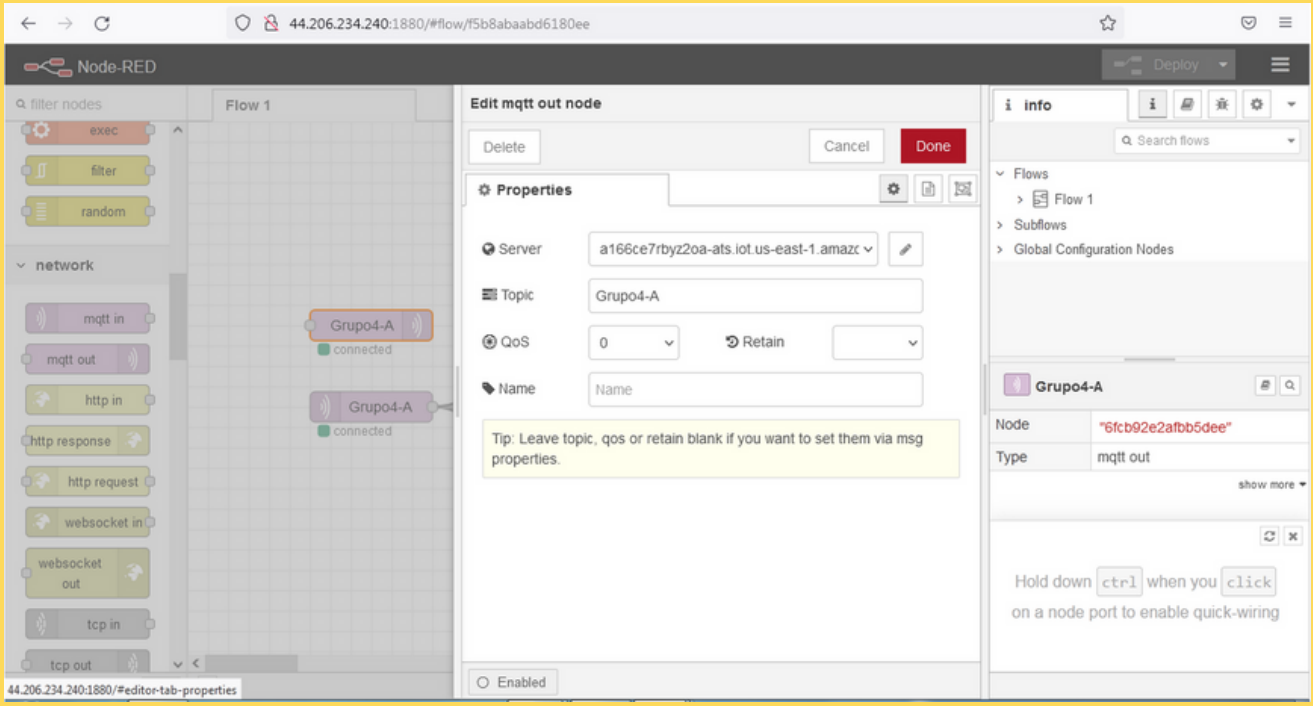
- Se asigno de tipo 'Donut', para que se vea de la forma que aparecía en los ejemplos.
- Se configuro para que aparezca la palabra "Pulso" en el dashboard.
- Y se asigno el color verde para cuando el valor es 1.



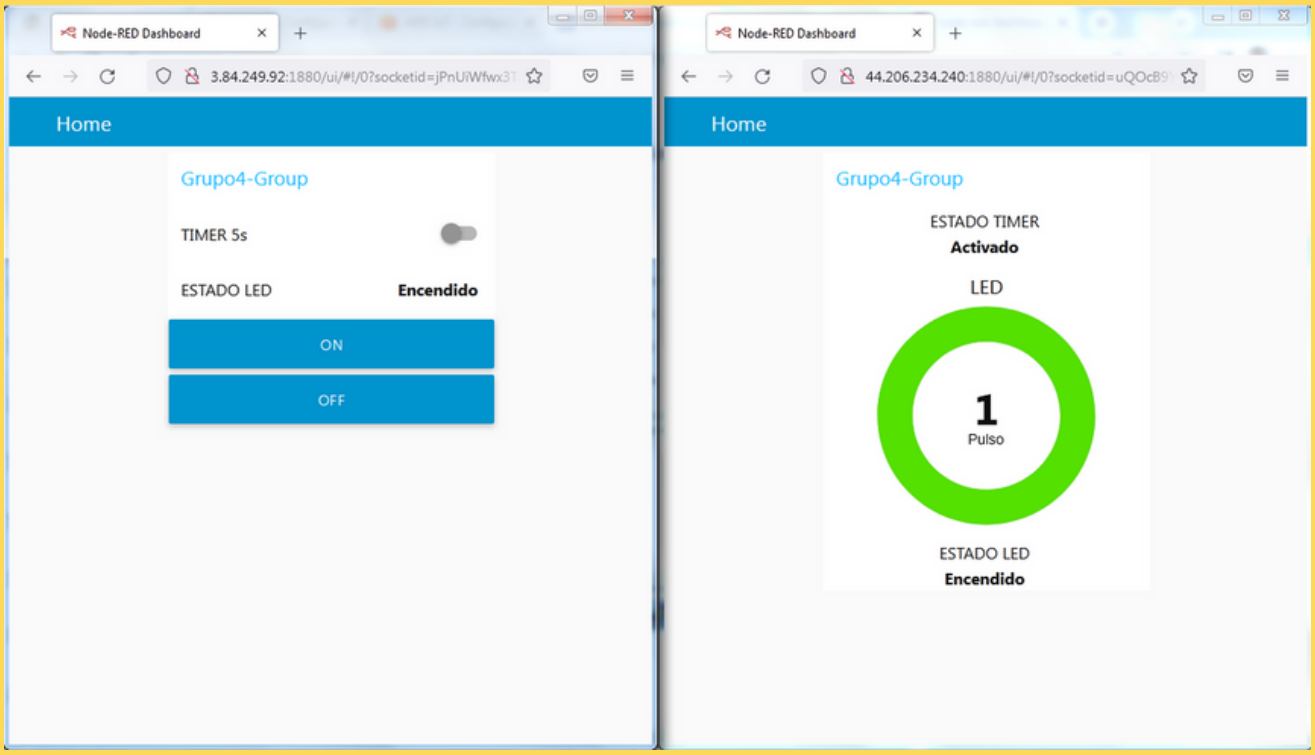
Configuración nodo MQTT IN:

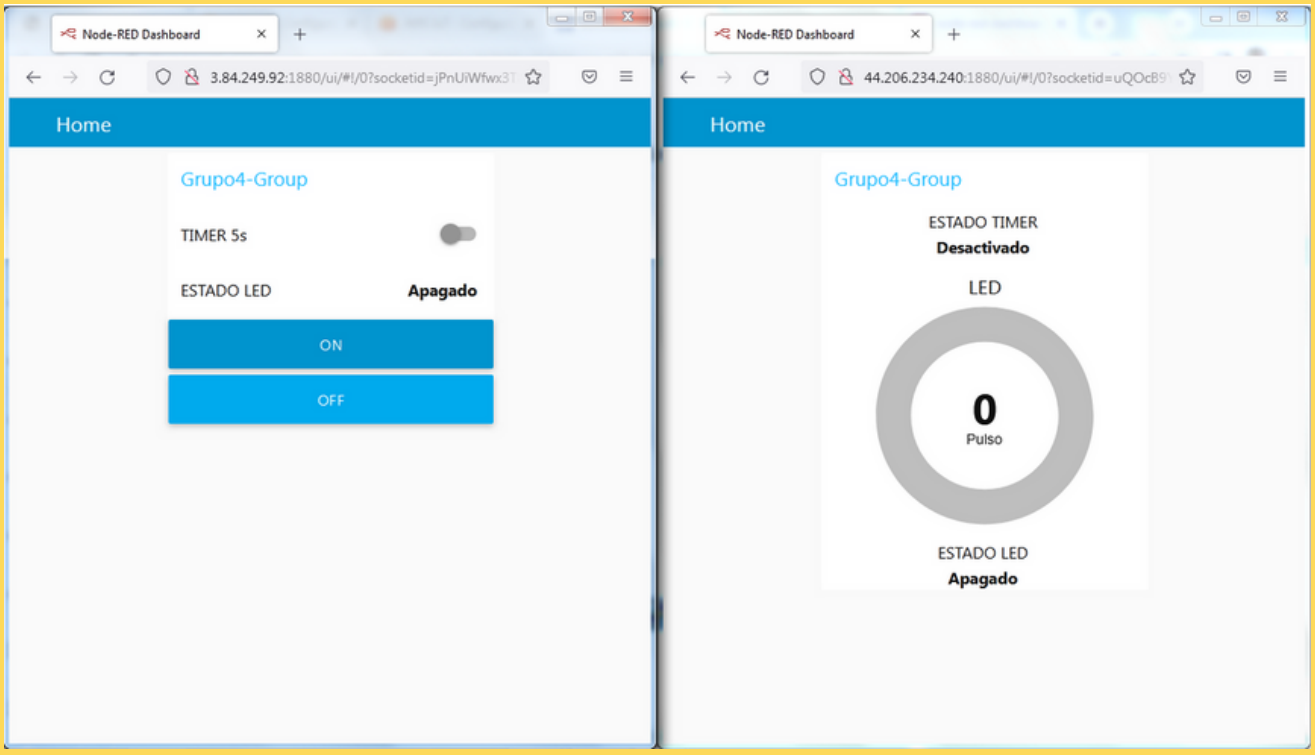


Configuración nodo MQTT OUT:



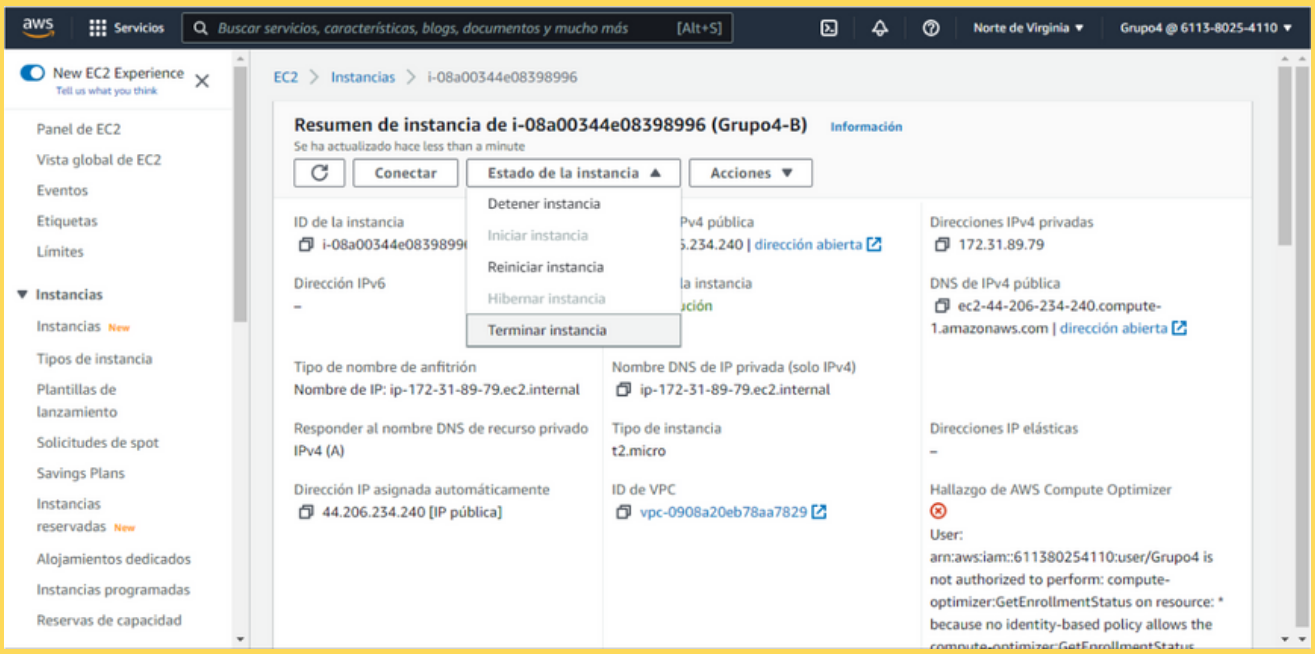
Para ver los dashboards, se ingreso la siguiente dirección:
ip-publica:1880/ui
A continuacion, se ven los resultados:



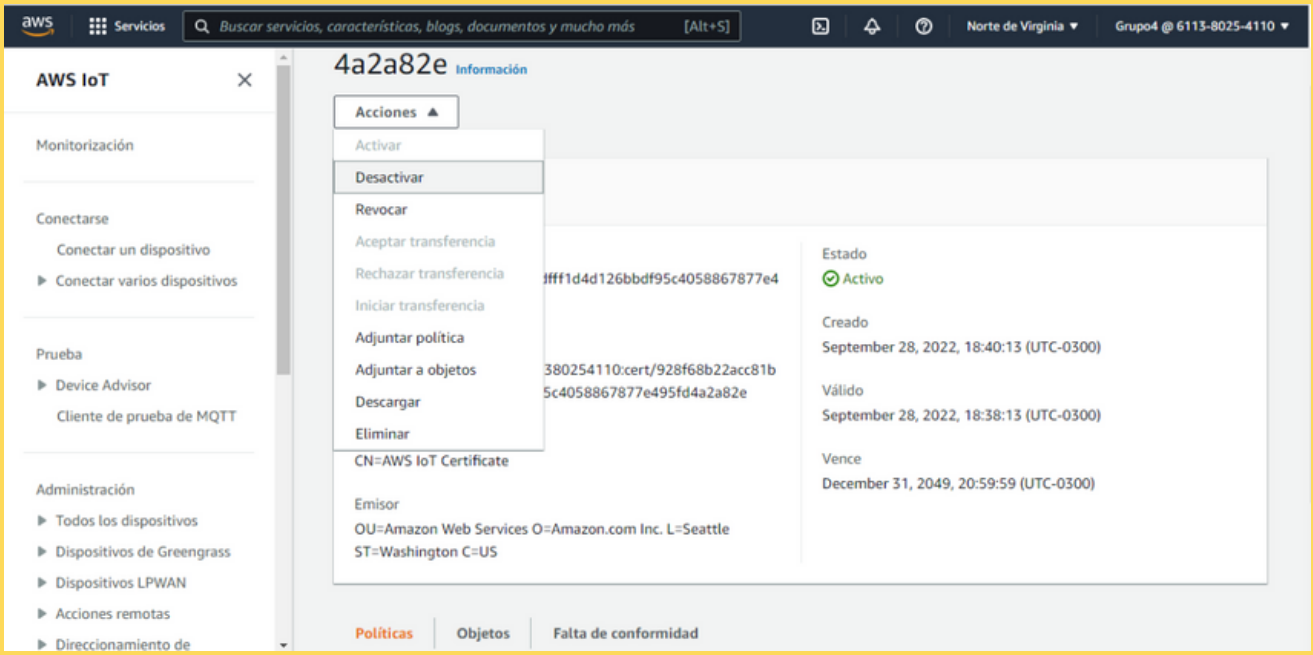


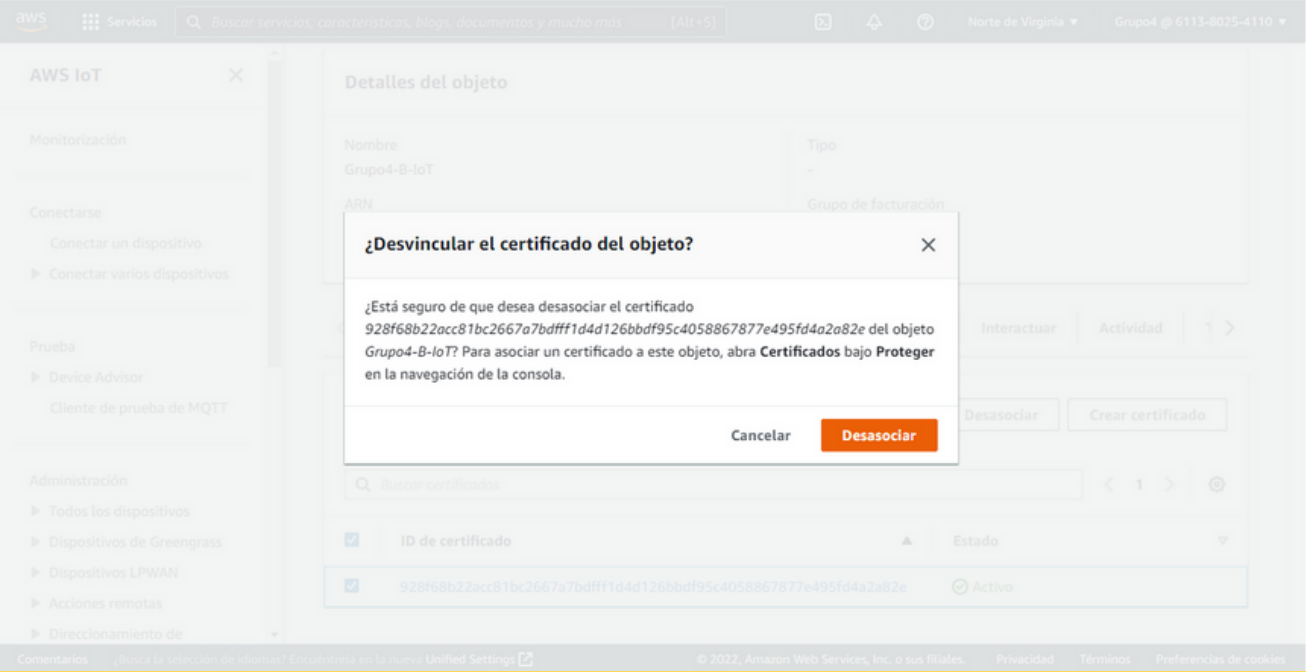
Para finalizar, como siempre, terminamos las instancias y liberamos los recursos ocupados:

Primero, como vimos en la actividad anterior, terminamos las instancias.

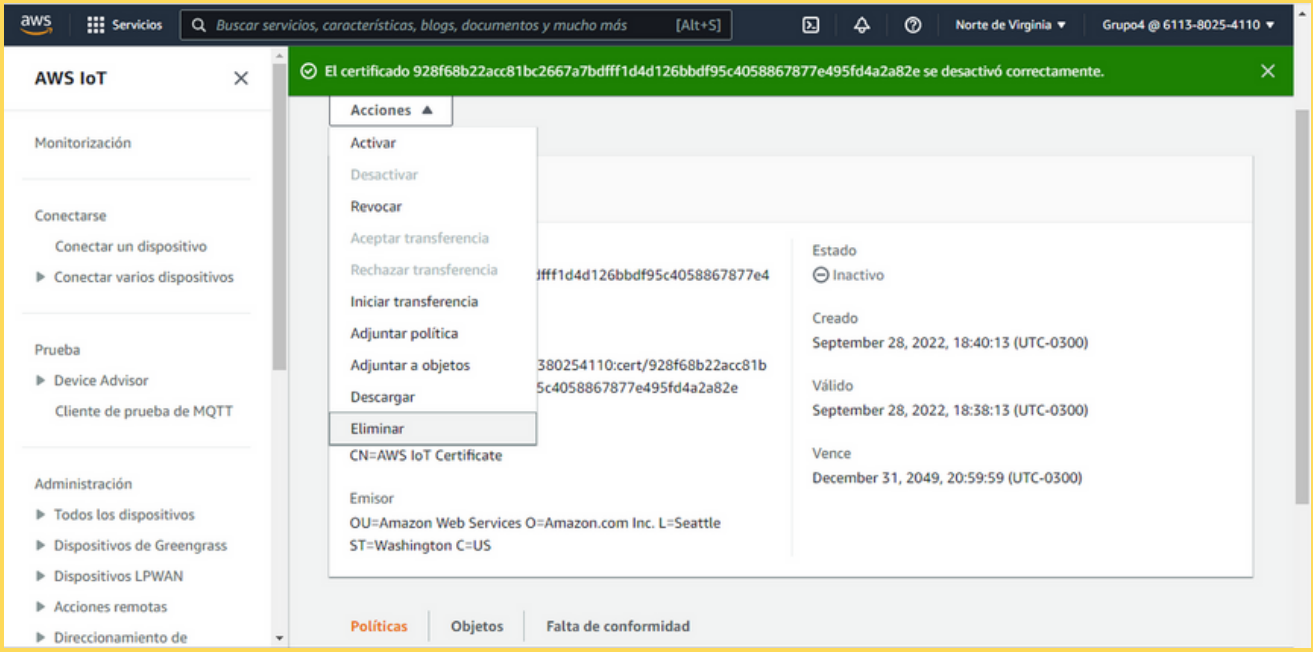


Luego procedemos a la eliminamos el certificado, el cual primero requerirá ser desactivado

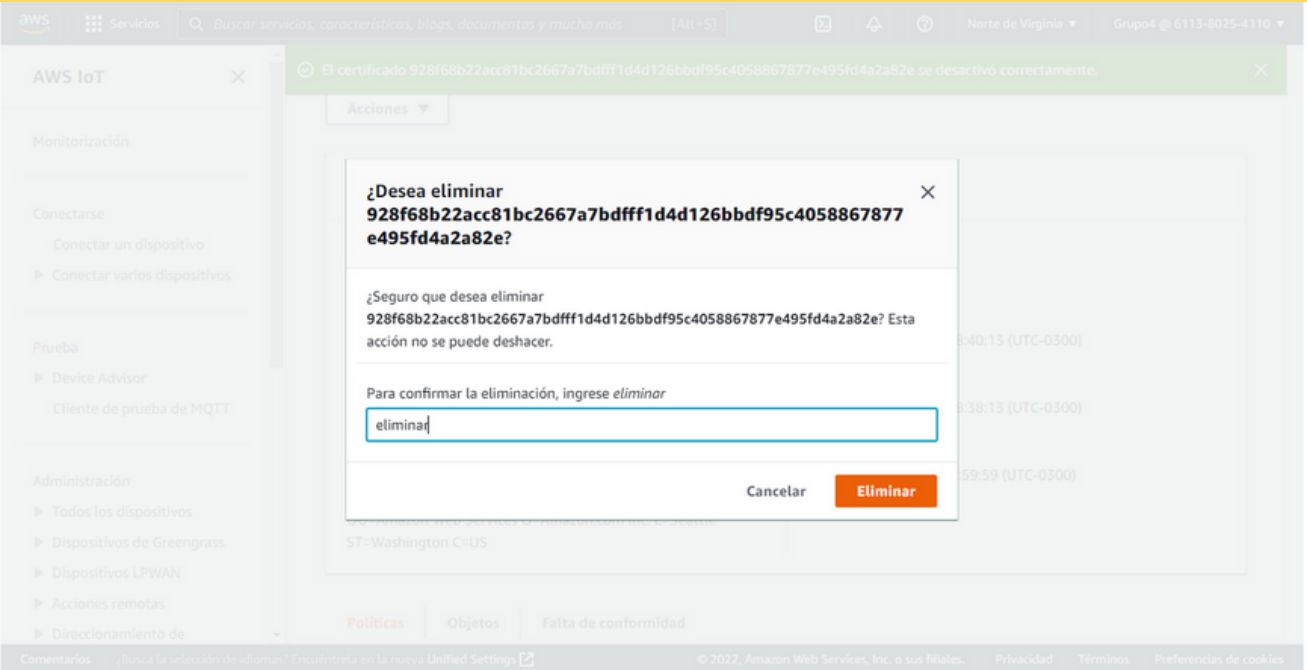




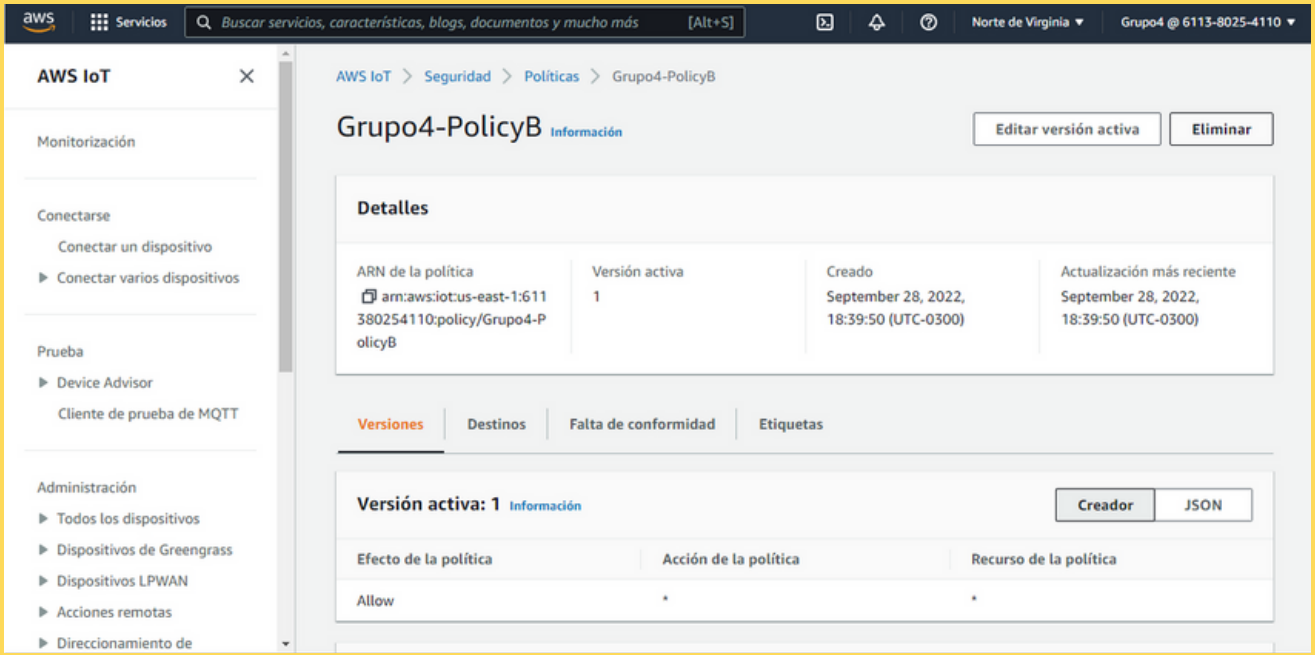
Una vez desasociado podrá ser eliminado.

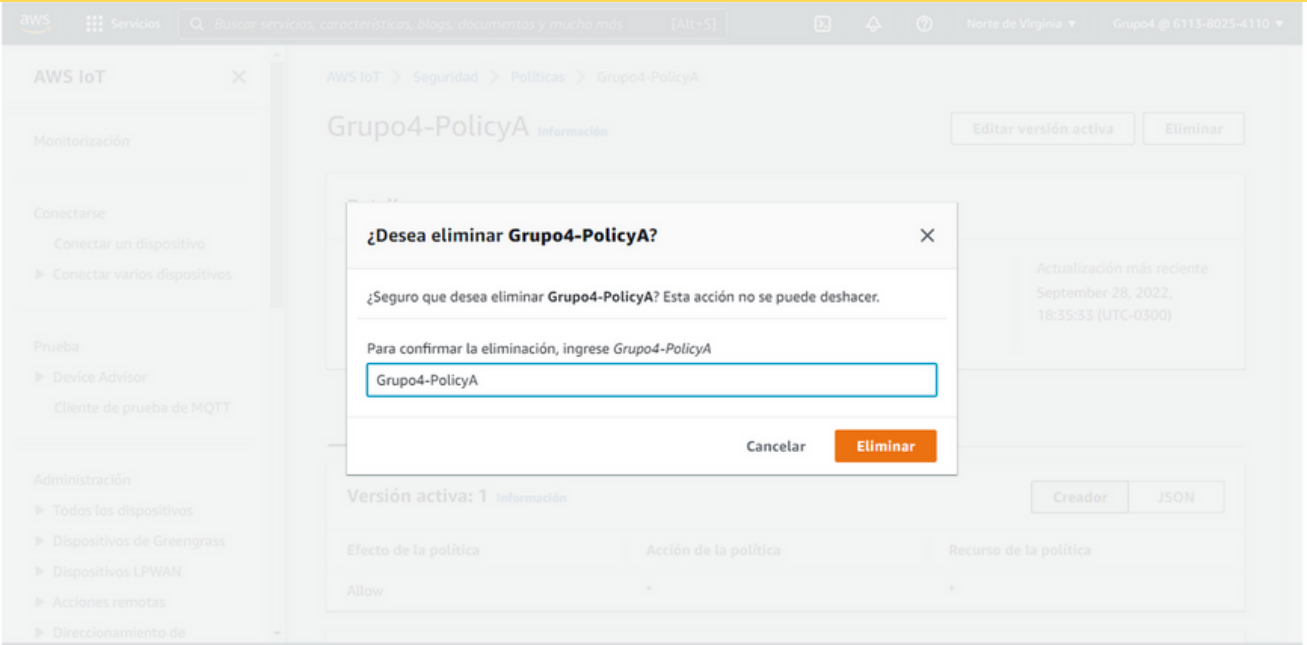


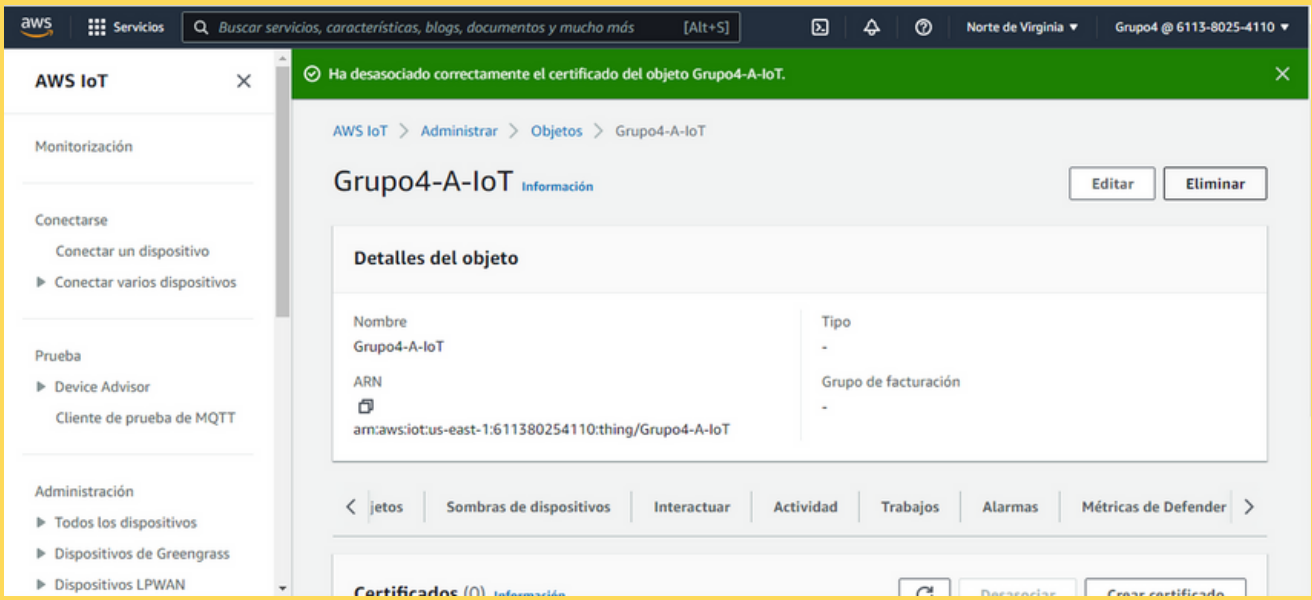
Confirmamos que queremos eliminarlo.

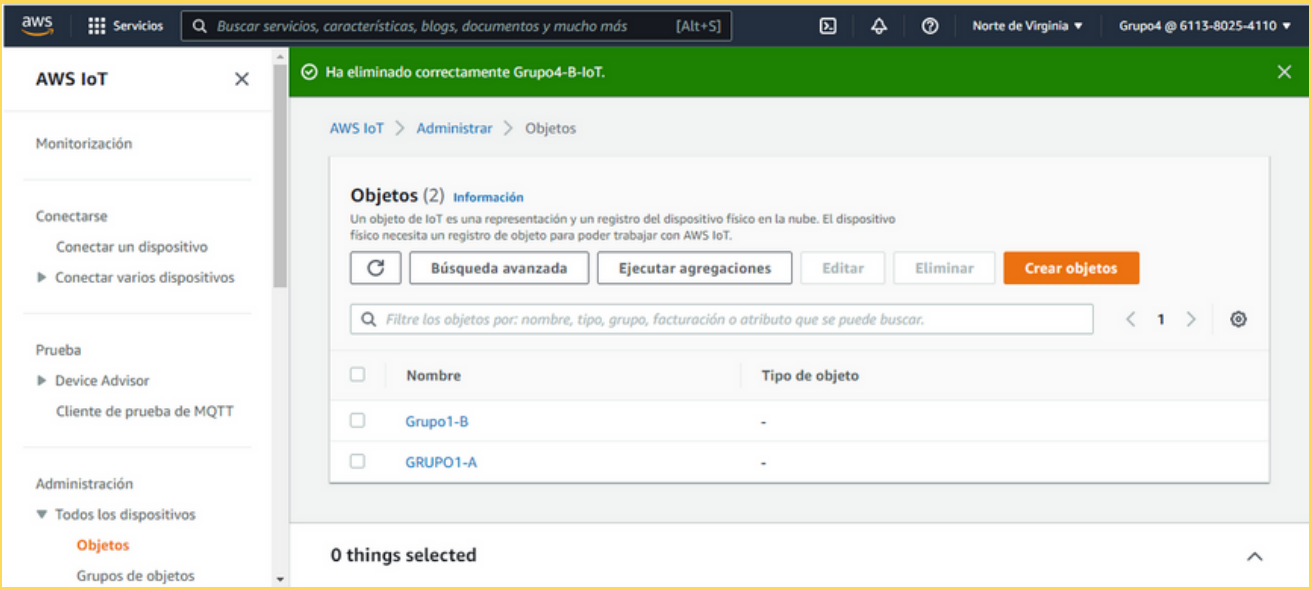


Luego eliminamos las políticas de seguridad.

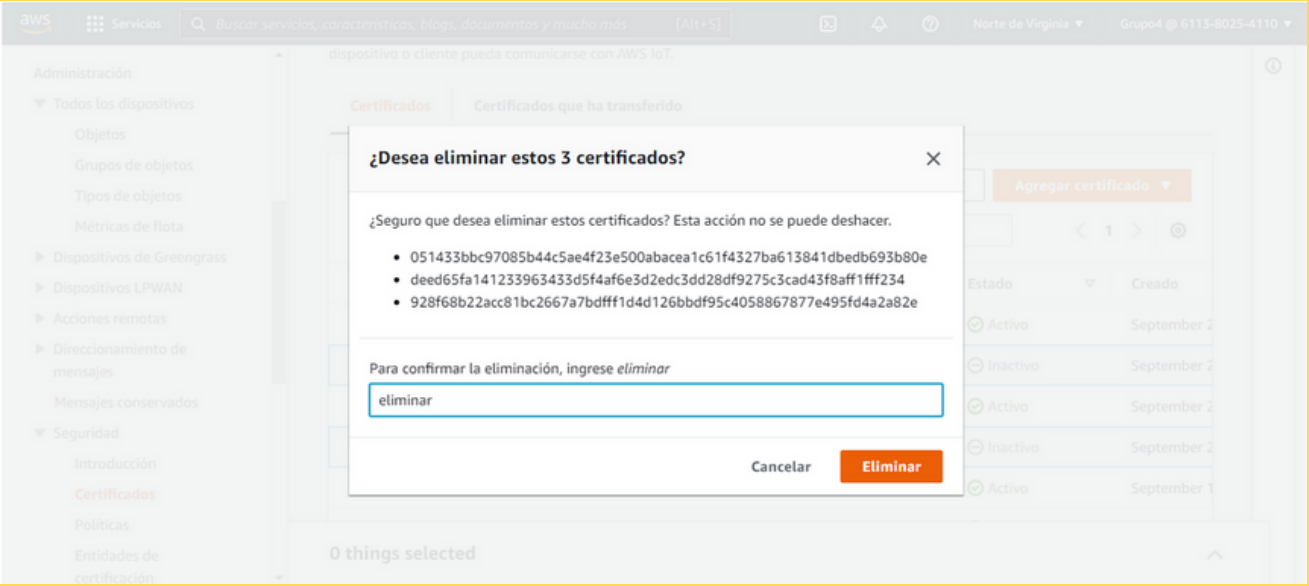


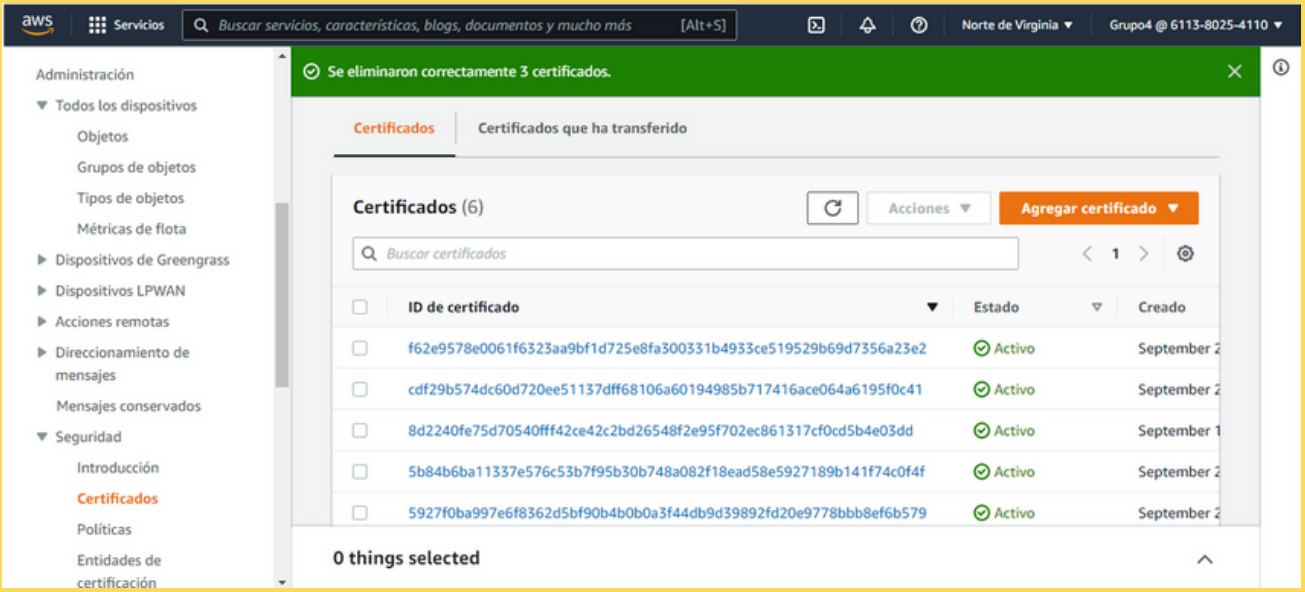

¿Desea eliminar Grupo4-PolicyA?
¿Seguro que desea eliminar Grupo4-PolicyA? Esta acción no se puede deshacer.
Para confirmar la eliminación, ingrese Grupo4-PolicyA
Grupo4-PolicyA
Cancelar
Eliminar


Ha desasociado correctamente el certificado del objeto Grupo4-A-IoT.
AWS IoT > Administrar > Objetos > Grupo4-A-IoT
Grupo4-A-IoT Información
Editar
Eliminar
Detalles del objeto
Nombre
Grupo4-A-IoT
Tipo
-
ARN
arn:aws:iot:us-east-1:611380254110:thing/Grupo4-A-IoT
Grupo de facturación
-
Sombras de dispositivos
Interactuar
Actividad
Trabajos
Alarmas
Métricas de Defender



Eliminamos los certificados





Aclaración: Para la resolución de la actividad solo fueron requeridos 2 certificados. En la captura se observan 3 debido a que uno fue el resultado de una prueba.

Información Adicional

IoT VS M2M

M2M (machine to machine, 'máquina a máquina') es un concepto genérico que se refiere al intercambio de información o comunicación en formato de datos entre dos máquinas remotas. El desarrollo de la tecnología M2M está relacionado con el Internet de las Cosas y otras tecnologías avanzadas. Los principales campos de aplicación son la automoción, la electrónica, la logística y las empresas de la energía. Las aplicaciones se extenderán en el futuro a otros sectores.
Tecnológicamente, el desarrollo de la tecnología hace prever un uso creciente. Estos son algunos de las causas:

- Comunicación por satélite. La mejora de comunicaciones satelitales permitirá una comunicación global entre máquinas, refiriéndonos en este caso, por ejemplo, a vehículos marítimos, aéreos o terrestres.
- Incremento del número de dispositivos conectados. También de la complejidad creciente de los dispositivos y de las transacciones de negocio.
- Incremento de las transacciones debido a la aparición de tecnologías transversales tales como el blockchain, que permitirá seguridad y auditoría, o la inteligencia artificial.

El Internet de las cosas (**IoT**, IdC) describe objetos físicos (o grupos de estos) con sensores, capacidad de procesamiento, software y otras tecnologías que se conectan e intercambian datos con otros dispositivos y sistemas a través de internet u otras redes de comunicación.El Internet de las cosas se ha considerado un término erróneo porque los dispositivos no necesitan estar conectados a la Internet pública. Sólo necesitan estar conectadas a una red y ser direccionables individualmente.

Modelos de comunicación de IoT

Desde un punto de vista operativo, tiene sentido pensar en cómo se conectan y comunican dispositivos del IoT desde la perspectiva del modelo de comunicación. En 2015, la Junta de Arquitectura de Internet (IAB) publicó un documento orientación para la creación de redes de objetos inteligentes que describe marcos de cuatro modelos de comunicación comunes utilizados en dispositivos de comunicación al Internet de las Cosas.

- Comunicaciones ‘dispositivo a dispositivo’
- Modelo ‘dispositivo a puerta de enlace’
- Modelo de intercambio de datos a través del back-end
- Comunicaciones ‘dispositivo a la nube'

En un modelo de comunicación de dispositivo a la nube, el dispositivo de la IoT se conecta directamente a un servicio en la nube, como por ejemplo un proveedor de servicios de aplicaciones para intercambiar datos y controlar el tráfico de mensajes. Este enfoque suele aprovechar los mecanismos de comunicación existentes (por ejemplo, las conexiones Wi-Fi o Ethernet cableadas tradicionales) para establecer una conexión entre el dispositivo y la red IP, que luego se conecta con el servicio en la nube.
https://es.wikipedia.org/wiki/M2M
https://es.wikipedia.org/wiki/Internet_de_las_cosas

Dispositivos : (Sensores y actuadores)
Los sensores y actuadores son elementos esenciales de los sistemas embebidos. Se utilizan en varias aplicaciones de la vida real, como el sistema de control de vuelo en un avión, los sistemas de control de procesos en los reactores nucleares, las centrales eléctricas que requieren ser operados en un control automatizado. Los sensores y los actuadores se diferencian principalmente por el propósito que ambos proporcionan, el sensor se utiliza para monitorear los cambios en el medio ambiente mediante el uso de measurands mientras que el actuador se utiliza cuando junto con el monitoreo también se aplica el control como para controlar el cambio físico.
Estos dispositivos actúan como el mediador entre el entorno físico y el sistema electrónico donde el sensor y el actuador están incrustados.

Un **sensor** es un instrumento electrónico que es capaz de medir la cantidad física y generar una salida considerada. Esta salida de los sensores suele ser en forma de señales eléctricas. Los sensores se colocan de forma que puedan interactuar directamente con el entorno para percibir la energía de entrada con la ayuda de un elemento sensor. Esta energía detectada se convierte en una forma más adecuada por un elemento de transducción.
Hay varios tipos de sensores como los de posición, temperatura, presión, velocidad, pero fundamentalmente hay dos tipos – analógico y digital.

Un **actuador** es un dispositivo que altera la cantidad física ya que puede hacer que un componente mecánico se mueva después de obtener alguna entrada del sensor. En otras palabras, recibe la entrada de control (generalmente en forma de señal eléctrica) y genera un cambio en el sistema físico
Conclusión
Los sensores presentan al ordenador información sobre el estado del sistema. Por otro lado, los actuadores aceptan órdenes para realizar una función.

https://localizadorgpscoche.com/diferencia-entre-sensores-y-actuadores/
https://es.wikipedia.org/wiki/Sensor

Smart Cities

Hoy en día, las ciudades se están volviendo "más inteligentes" para brindar una mejor prestación y calidad de los servicios a través del monitoreo continuo de los residentes y la infraestructura, y la comunicación relativamente instantánea del desempeño subóptimo. En gran medida, esto requiere una gran dependencia de la automatización, la conectividad a Internet y lo que se conoce como "Internet de las cosas (IoT)". (Internet de las cosas se refiere a la conexión de dispositivos a Internet que se pueden controlar o se pueden usar para enviar información de control). Entonces, una ciudad inteligente es una ciudad en la que se implementa un conjunto de sensores (generalmente cientos o miles) para recopilar datos electrónicos de y sobre las personas y la infraestructura para mejorar la eficiencia y la calidad de vida. Los residentes y los trabajadores de la ciudad, a su vez, pueden recibir aplicaciones que les permitan acceder a los servicios de la ciudad, recibir y emitir informes de apagones, accidentes y delitos, pagar impuestos, tarifas y similares. En la ciudad inteligente se enfatiza la eficiencia energética y la sustentabilidad.

https://education.nationalgeographic.org/resource/smart-cities

Smart logistics

Al hablar de smart logistics se hace referencia a un tipo de logística concreta que está caracterizada por una mejora considerable de la eficiencia en todos los procesos integrados en el propio desarrollo de la actividad logística. Es decir, no se trata de un único sistema de logística, sino que más bien hace referencia a que, dicha logística, se ejecute desde una perspectiva lo más eficiente, sostenible y rentable posible.
En este sentido, aunque no existe un único aspecto que deba ser tenido en cuenta a la hora de identificar las smart logistics, sí que existen ciertos elementos comunes y muy presentes en ella:

- Logística sostenible
- Logística social
- Uso de nuevas tecnologías aplicadas a la trazabilidad
- Automatización de procesos

Estos cuatro elementos son clave a la hora de comprender qué es smart logistics, ya que son aspectos que aumentan la eficiencia de las actividades logísticas en su conjunto y, al mismo tiempo, responden a las demandas de los consumidores que esperan que sus decisiones de compra tengan un impacto no solo en su inmediatez económica, sino en la sociedad en su conjunto.

https://www.beetrack.com/es/blog/smart-logistics-que-es

Tecnologías de comunicación :

NFC: NFC (Near Field Communication), en castellano Comunicación de Campo Cercano, es una tecnología que permite la comunicación de corto alcance entre dos dispositivos inalámbricos de manera cómoda y rápida.
La tecnología NFC se utiliza para intercambiar información entre dos dispositivos, los cuales deben situarse a una corta distancia (no más de 15 centímetros). Cuando dos dispositivos que cuentan con este sistema se acercan, la tecnología NFC comienza a funcionar. Es una evolución de la tecnología contactless que no solo sirve para realizar pagos sino también intercambiar todo tipo de información.
¿Cómo funciona NFC?
Esta tecnología puede aplicar dos modos de funcionamiento: activo y pasivo. En el modo activo se produce un flujo de información en ambas direcciones, ya que ambos dispositivos son capaces de generar su propio campo electromagnético y utilizarlo para intercambiar información.
Por otro lado, en el modo pasivo uno de los dispositivos no cuenta con fuente de energía propia y necesita que el segundo dispositivo genere el campo electromagnético en el que se van a intercambiar los datos. Es decir, el dispositivo activo actúa como un lector de la información que contiene el dispositivo pasivo.

https://www.bbva.com.ar/economia-para-tu-dia-a-dia/tecnologia/banca-digital/que-es-nfc-como-funciona-y-para-que-usarlo.html

RFID: RFID o identificación por radiofrecuencia (del inglés Radio Frequency Identification) es un sistema de almacenamiento y recuperación de datos remotos que usa dispositivos denominados etiquetas, tarjetas o transpondedores RFID. El propósito fundamental de la tecnología RFID es transmitir la identidad de un objeto (similar a un número de serie único) mediante ondas de radio. Las tecnologías RFID se agrupan dentro de las denominadas Auto ID (automatic identification, o identificación automática).
Las etiquetas RFID (RFID tag en inglés) son unos dispositivos pequeños, similares a una calcomanía, que pueden ser adheridas o incorporadas a un producto, un animal o una persona. Contienen antenas para permitirles recibir y responder a peticiones por radiofrecuencia desde un emisor-receptor RFID. Las etiquetas pasivas no necesitan alimentación eléctrica interna, mientras que las activas sí lo requieren.
Una de las ventajas del uso de radiofrecuencia (en lugar, por ejemplo, de infrarrojos como muchas alarmas) es que no se requiere visión directa entre emisor y receptor.

https://es.wikipedia.org/wiki/RFID
Bluetooth 4.0, WiFi: https://www.youtube.com/watch?v=tYwHyzZ7IJY

Protocolo MQTT

MQTT son las siglas de **Message Queuing Telemetry Transport**. Se trata de un protocolo de mensajería ligero para usar en casos de clientes que necesitan una huella de código pequeña, que están conectados a redes no fiables o con recursos limitados en cuanto al ancho de banda. Se utiliza principalmente para comunicaciones de máquina a máquina (M2M) o conexiones del tipo de Internet de las cosas.

MQTT se ejecuta sobre TCP/IP utilizando una topología PUSH/SUBSCRIBE. En la arquitectura MQTT, existen dos tipos de sistemas: clientes y brókeres. **Un bróker es el servidor con el que se comunican los clientes**: recibe comunicaciones de unos y se las envía a otros. Los clientes no se comunican directamente entre sí, sino que se conectan con el bróker. Cada cliente puede ser un editor, un suscriptor o ambos.

MQTT es un protocolo controlado por eventos, donde no hay transmisión de datos periódica o continua. Así se mantiene el volumen de transmisión al mínimo. Un cliente sólo publica cuando hay información para enviar, y un bróker sólo envía información a los suscriptores cuando llegan nuevos datos.

Otra forma en que MQTT minimiza sus transmisiones es con un tamaño de mensaje pequeño y bien definido. Cada mensaje tiene un encabezado fijo de apenas 2 bytes. Se puede utilizar un encabezado opcional, pero eso incrementa el tamaño del mensaje. La carga útil del mensaje está limitada a únicamente 256 MB. **Tres niveles diferentes de calidad de servicio (QoS)** permiten a los diseñadores de redes elegir entre minimizar la transmisión de datos y maximizar la fiabilidad.

QoS 0: ofrece la cantidad mínima de transmisión de datos. Con este nivel, cada mensaje se entrega a un suscriptor una vez, sin confirmación, por lo que no hay forma de saber si los suscriptores recibieron el mensaje.

QoS 1: el bróker intenta entregar el mensaje y, luego, espera una respuesta de confirmación del suscriptor. Si no se recibe una confirmación dentro de un período de tiempo especificado, el mensaje se envía de nuevo.

QoS 2: el cliente y el bróker utilizan un protocolo de enlace de cuatro pasos para garantizar no sólo que el mensaje se reciba, sino que lo haga una única vez.

QoS 0 puede ser la mejor opción para situaciones donde las comunicaciones son fiables pero limitadas. En aquellos casos en que las comunicaciones no sean fiables, pero donde las conexiones tampoco gozan de recursos limitados, QoS 2 sería la mejor opción. QoS 1 proporciona una solución a caballo entre ambos mundos, pero requiere que la aplicación que recibe los datos sepa cómo gestionar los duplicados.

https://www.paessler.com/es/it-explained/mqtt

AWS IoT – Node-Red

Node-RED es una herramienta de programación visual que se implementa en dispositivos controladores de hardware. Trabaja mostrando de manera visual las relaciones y funciones de manera que se pueda programar sin escribir. Es un panel de flow al que se pueden incorporar nodos que se comuniquen entre ellos y puede instalarse en equipos como ordenadores Windows, Linux, o en servidores en la nube.

Este sistema de representaciónpuede ayudarte a visualizar gráficamente el flujo de la información y consiste en su representación gráfica.

Es una herramienta diseñada para comunicar hardware, que conecta varios dispositivos a la vez, tanto de hardware como de servicios de Internet. Es perfecto para los equipos dedicados a trabajo industrial o como prueba de soluciones para equipos de planta que necesiten comunicarse entre sí. Se trata de un motor de flujos que cuenta con un enfoque IoT, y esta condición permite definir flujos de servicios a través de protocolos como el **MQTT.**

Facilita por sus características el flujo de información, que distribuye hacia el hardware que, a su vez, generará un servicio.

https://blog.gruposinelec.com/actualidad/que-es-node-red-y-para-que-sirve/

Cloud Robotics

La robótica en la nube es un campo de la robótica que intenta invocar tecnologías en la nube, como computación en la nube, almacenamiento en la nube y otras tecnologías de Internet centradas en los beneficios de la infraestructura convergente y los servicios compartidos para la robótica. Cuando están conectados a la nube, los robots pueden beneficiarse de los poderosos recursos de cómputo, almacenamiento y comunicación de los centros de datos modernos en la nube, que pueden procesar y compartir información de varios robots o agentes (otras máquinas, objetos inteligentes, humanos, etc.) . Los humanos también pueden delegar tareas a robots de forma remota a través de redes. Las tecnologías de computación en la nube permiten dotar a los sistemas robóticos de una capacidad poderosa al mismo tiempo que reducen los costos a través de las tecnologías en la nube. Por lo tanto, es posible construir robots más inteligentes, livianos y de bajo costo con un "cerebro" inteligente en la nube. El "cerebro" consta de centro de datos, base de conocimiento, planificadores de tareas, aprendizaje profundo, procesamiento de información, modelos ambientales, soporte de comunicación, etc.

https://en.wikipedia.org/wiki/Cloud_robotics
https://docs.aws.amazon.com/es_es/iot/latest/developerguide/aws-iot-how-it-works.html
https://aws.amazon.com/es/what-is/iot/
https://docs.aws.amazon.com/es_es/iot/latest/developerguide/what-is-aws-iot.html