Instituto Politécnico Nacional
Unidad Profesional Interdisciplinaria
en Ingeniería y Tecnologías Avanzadas

MANUAL DE INSTALACIÓN Y EJECUCIÓN

Seguridad en sistemas de Internet de las cosas usando una técnica MTD de aleatorización de dialectos del protocolo CoAP

Elaboró:
González Durán Edgar
González González Erik Ulises

Índice

Índice de figuras

1. Requerimientos del sistema

Para poder ejecutar el sistema *IoT* que utiliza la técnica *MTD* implementada en este proyecto terminal, se necesitan los siguientes elementos:

Para el cliente:

- Máquina virtual con alguna distribución de Debian.
- Python (versión 8.6 o superior).
- Librerías aiocoap, coapts, coaprcs, coapcpv, coapshow, pickle, numpy, termcolor.

Para el servidor:

- Raspberry Pi 4 modelo B.
- Sistema operativo Raspberry OS.
- Placa Sense Hat para medir temperatura y humedad.
- Python (versión 8.6 o superior).
- Librerías aiocoap, coapts, coaprcs, coapcpv, coapshow, termcolor.

2. Instalación de librerías requeridas

Tanto cliente como servidor requieren algunas librerías de Python para poder ejecutarse correctamente.

2.1 Instalar aiocoap

```
$ git clone https://github.com/chrysn/aiocoap
$ cd aiocoap
```

Con ese comando es posible usar el proyecto desde esa ubicación o se puede instalar con:

```
$ pip install --upgrade "git+https://github.com/chrysn/aiocoap#egg=aiocoap[all]"
```

2.2 Instalar las demás librerías requeridas

En este caso es posible instalar todos los paquetes requeridos usando un solo comando.

Para el cliente:

```
$ pip install coapts coaprcs coapcpv coapshow pickle numpy termcolor
```

Para el servidor:

```
$ pip install coapts coaprcs coapcpv coapshow termcolor
```

3. Descarga del proyecto

Es posible descargar el contenido del proyecto a través del repositorio de *GitHub* donde se encuentra. Esto se hace de la siguiente forma, tanto en el cliente como en el servidor:

```
$ git clone https://github.com/ErikGlz/CoAP-MTD
```

La figura 1 muestra la estructura general del proyecto.

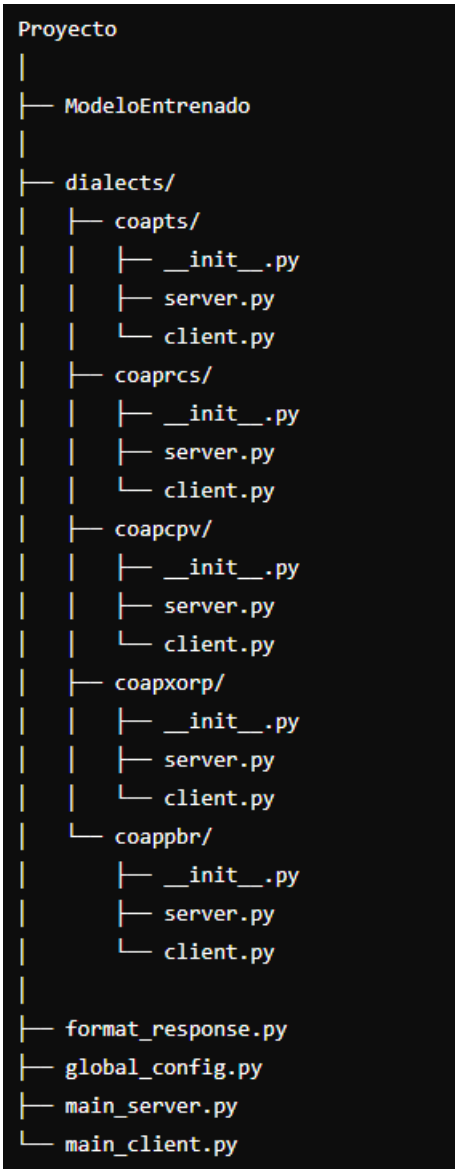

Figura 1: Estructura del sistema *IoT* que utiliza la técnica *MTD*

Como se puede observar, en su estructura se contemplan los códigos de cliente y servidor, por lo que solo se deben usar los que correspondan al lado del sistema que se utilice.

4. Ejecución del proyecto

Una vez que se cuentan con todos los elementos anteriores, es posible ejecutar el sistema *IoT* que utiliza la técnica *MTD*.

4.1 Ejecución del servidor

Para comenzar la ejecución del servidor se usa el siguiente comando:

```
$ python3 main_server.py
```

De esta forma, se inicializará el servidor utilizando el dialecto 1, como se observa en la figura 2.

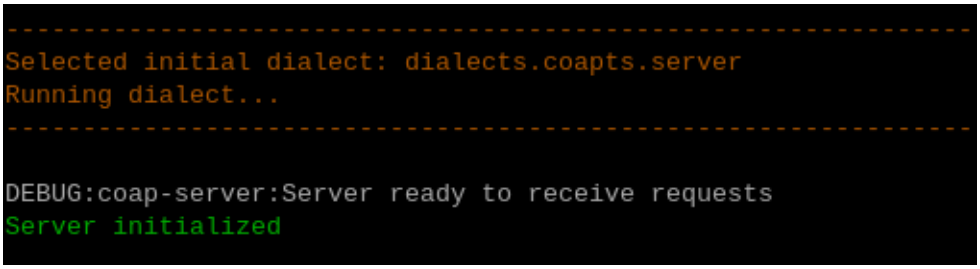

Figura 2: Inicialización del servidor

Debido a que el servidor hace todo el procesamiento de las peticiones de forma automática, no es necesario realizar ninguna acción una vez que se ejecuta.

Una vez que se hace el cambio de dialecto, el servidor indica que se recibió una *seed* y que se hará el cambio al dialecto seleccionado, para posteriormente cerrar el contexto del dialecto actual e iniciar la ejecución del nuevo dialecto, como se oberva en la figura 3.

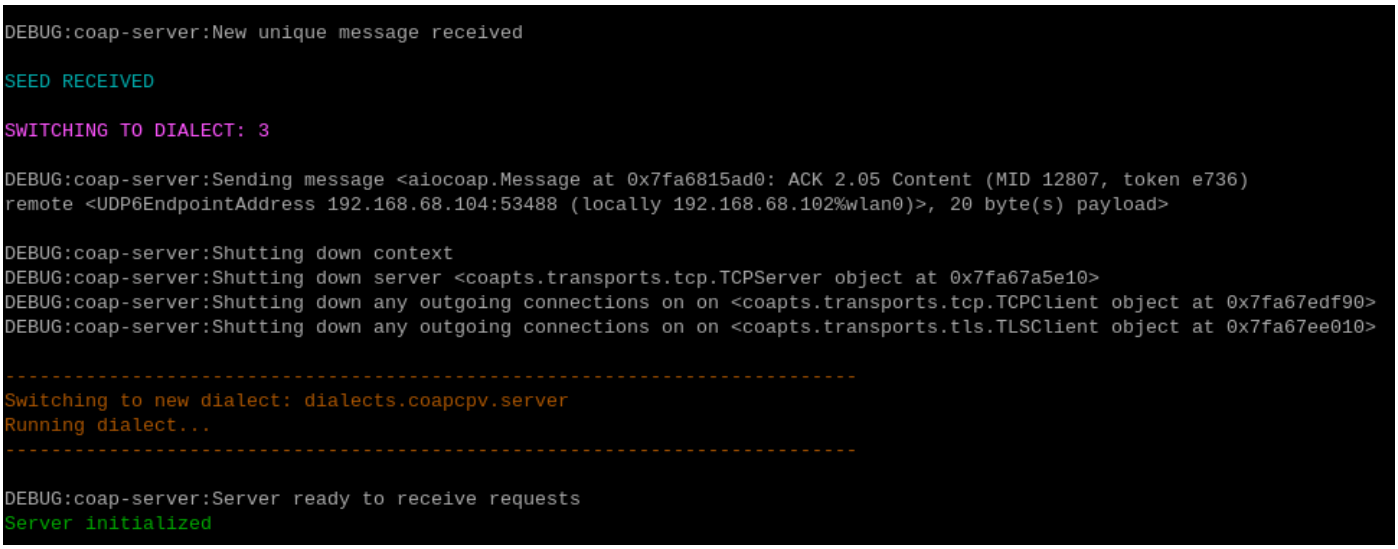

Figura 3: Cambio de dialecto en el servidor

4.2 Ejecución del cliente

Para comenzar la ejecución del cliente se usa el siguiente comando:

```
$ python3 main_client.py
```

Al igual que en el servidor, el cliente se inicializa utilizando el dialecto 1, como se puede ver en la figura 4.

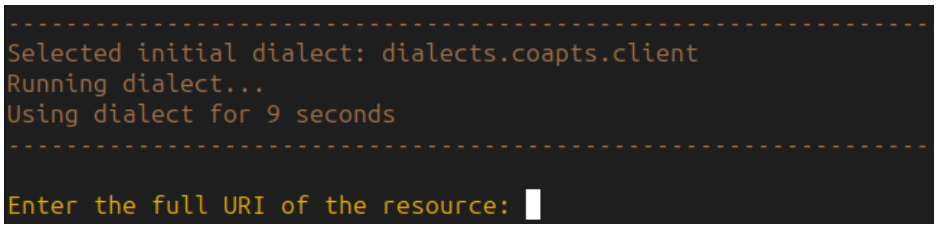

Figura 4: Inicialización del cliente

Como se puede observar, además de mostrar la información de la inicialización, también es posible visualizar el tiempo establecido para utilizar dicho dialecto antes de que el sistema se reconfigure y comience a utilizar un nuevo dialecto. También es posible introducir la *URI* del recurso que se desea obtener del servidor, el cual debe tener la siguiente estructura:

```
coap://[IP del servidor]/[recurso]
```

Una vez que se conoce la dirección *IP* y el recurso que se desea obtener, es posible hacer peticiones al servidor y recibir la correspondiente respuesta con la información requerida, como se observa en la figura 5.

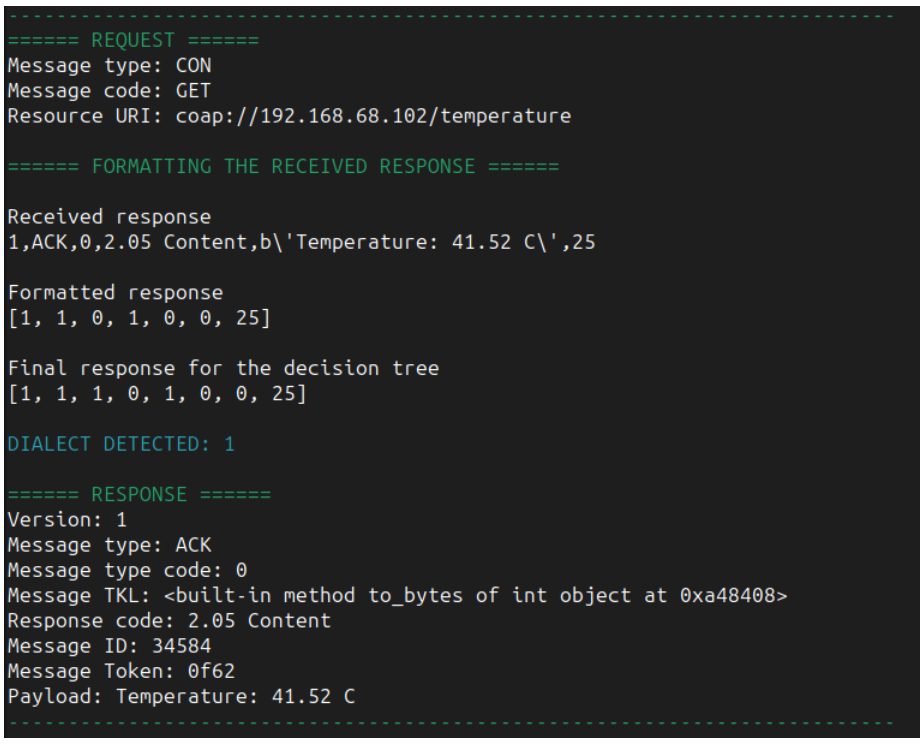

Figura 5: Información obtenida al hacer una petición y recibir la respuesta

Una vez que termina el tiempo de uso del dialecto actual, el cliente indica que se está haciendo el cambio al nuevo dialecto seleccionado, para posteriormente cerrar el contexto del dialecto actual e inicializar el nuevo dialecto, permitiendo de nuevo introducir la *URI* del recurso deseado. Como se observa en la figura 6.

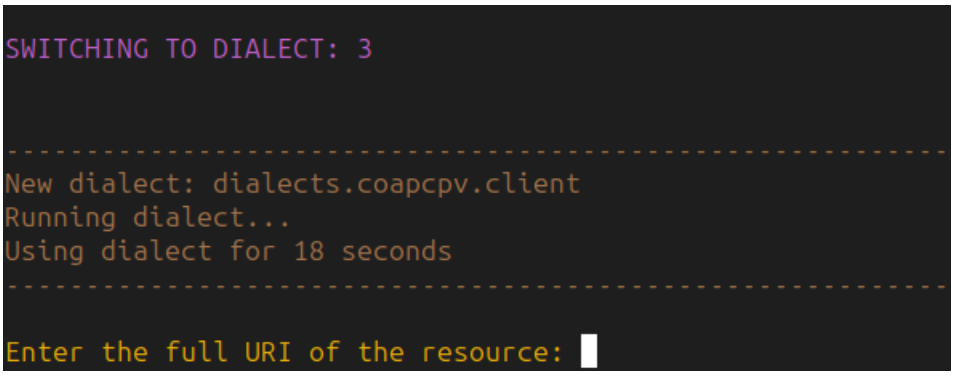

Figura 6: Cambio de dialecto en el cliente

Este proceso se realiza de manera continua e indeterminada, ya que el sistema se reconfigura cada 5 a 20 segundos para cambiar de dialecto. De esta forma, se añade seguridad al protocolo al incrementar la dificultad y el tiempo requerido para que un ataque tenga éxito.