Comentario técnico: CTC-091
Componente: **Autorización en servidor web con ESP32 y Mongoose-OS**
Autor: Sergio R. Caprile, Senior R&D Engineer

Revisiones	Fecha	Comentarios
0	23/04/20	

Mongoose-OS incorpora un servidor HTTP con Digest Authentication que podemos aprovechar para limitar quiénes pueden acceder al mismo. La identificación es simplemente por una pareja usuario/password, pero al menos es un modo simple y rápido (particularmente en un microcontrolador) de proveer una seguridad mínima. Esto no evita que lo que se envía pueda ser observado con un sniffer, tema que abordaremos en otra oportunidad; sin embargo el password no es revelado.

HTTP Digest

Es una de las formas posibles de autenticación en un servidor web. Posee, frente al método básico original, la ventaja de no revelar la clave del usuario, la cual en vez de ser enviada es comprobada mediante una operación de hashing realizada por ambos interlocutores. El servidor puede, de este modo, validar al usuario y permitirle acceso a los contenidos protegidos. El usuario, sin embargo, no tiene forma de saber si el servidor es realmente quien dice ser, dado que para esto se requiere un esquema de autenticación más complicado, con un ente certificador.

Configuración

A continuación, configuramos el ESP32 con Mongoose-OS para operar como un Access Point con un web server. Si bien el funcionamiento como Access Point ya de por sí posee un password, asumimos que el servidor web en sí requiere acceso restringido. La operación como AP nos resulta la forma tal vez más rápida y simple de realizar las pruebas y explicar la operación. La configuración puede realizarse manualmente mediante RPC, en un archivo de configuración JSON; o definirla en el archivo YAML que describe el proyecto. Para las pruebas elegimos esta última opción.

```
libs:
  - origin: https://github.com/mongoose-os-libs/http-server # incorpora el servidor web
config_schema:
  - ["http.auth_domain", "myESP"]         # Nombre de dominio (realm) que muestra el navegador
  - ["http.auth_file", "myauth.txt"]      # Archivo conteniendo los pares usuario/password
```

Como vemos, no es algo que resulte muy amigable para modificaciones dinámicas; sino que debe preverse o al menos idear un mecanismo apropiado. La autorización es para todo el contenido estático del servidor[1], es decir, las páginas HTML que colocaremos en el directorio *fs* del proyecto; no así los RPC que sean accedidos por HTTP, lo cual abordaremos en otra oportunidad.
El archivo de configuración tiene el mismo formato utilizado por los servidores HTTP como Apache, por lo que lo generamos mediante *htdigest*. Finalmente, deberemos ubicar dicho archivo en el directorio *fs* del proyecto; será empaquetado al compilar y colocado dentro del dispositivo al grabar el firmware.
Si realizamos cambios mientras estamos desarrollando, podemos enviar la nueva versión del archivo mediante `mos put fs/myauth.txt .` Lo mismo podríamos realizar más adelante mediante alguna de las opciones de

1 Otras configuraciones son posibles, aunque tienen requerimientos de desarrollo más elevados.

transporte de RPC, (aunque en este caso sólo hemos compilado el soporte elemental). En ese caso utilizaremos el método `FS.Put` . Esto no es algo trivial, por lo que lo recomendable es hacerlo mediante *mos tool*[2].

htdigest

En general, para crear un nuevo archivo usaremos la opción *-c* (*create*), la cual deberemos omitir para agregar un usuario a un archivo existente:

```
htdigest -c <nuevoarchivo> <realm> <nuevousuario>
```

```
htdigest  <archivo> <realm> <nuevousuario>
```

El archivo es del tipo:

```
<usuario>:<realm>:<hash>
```

por ejemplo, el que proveemos contiene:

```
bob:myESP:6e34a8e3f1a6a0ca3d3d9401ba03145a
```

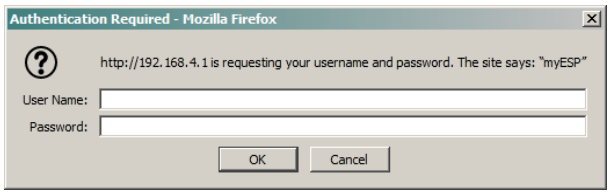
que corresponde al mismo usuario *bob* con password *hello* que hemos utilizado en otros Comentarios Técnicos.

Operación

Luego de compilado el código (`mos build`) y grabado el microcontrolador (`mos flash`) mediante *mos tool*, observaremos en el log la dirección del Access Point y el nombre.

```
[Apr 21 18:39:50.935] esp32_wifi.c:450        WiFi AP: SSID myESP_807A98, channel 6
[Apr 21 18:39:51.802] esp32_wifi.c:507        WiFi AP IP: 192.168.4.1/255.255.255.0 gw [...]
[Apr 21 18:39:51.813] mgos_http_server.c:343  HTTP server started on [80]
[Apr 21 18:39:51.880] init.js:6               ### init script started ###
```

Con cualquier dispositivo con capacidad de conexión a una red WiFi y un navegador nos conectaremos a esa red y luego pedimos al navegador la página principal de esa dirección: *http://192.168.4.1/*, y observaremos el clásico pedido de autorización.



2 https://mongoose-os.com/docs/mongoose-os/quickstart/setup.md