Servicio de tokens

Revisión 2

1. Introducción

Se deberá implementar una API REST que permita la gestión básica de tokens de autenticación. El lenguaje de implementación es libre, aunque se recomienda utilizar Python 3. Además, deberá crearse una librería de acceso a la API y un cliente que permita utilizar el servicio sin necesidad de conocer la propia API REST.

2. Implementación del servicio

El servicio no necesitará de capa de persistencia, por lo que sólo necesita dos capas: presentación y negocio. La capa de negocio debe acceder al servicio de autenticación utilizando la librería de acceso proporcionada por dicho servicio. Excepto la revocación de tokens, todas las operaciones son anónimas.

Las operaciones que debe implementar la capa de negocio son:

- Crear un nuevo token.
- Revocar un token.
- Revocar automáticamente los tokens que caduquen.
- Identificar al usuario dueño de un token.
- Si lo desea el usuario, recibir una notificación cuando caduque un token.

La capa de negocio contará con excepciones predefinidas que deberán lanzarse en caso de error, estas excepciones son: TokenNotFound(token) y Forbidden(token). La primera se producirá cuando se intente obtener la información sobre un token que no exista. La segunda cuando el usuario que intenta eliminar un token no sea su dueño.

Por último, se escribirá la capa de presentación, que consistirá en la API REST descrita a continuación.

3. API REST

A continuación, se indican los recursos admitidos por la API:

- API_ROOT/token
    - Método PUT
        - Input: {“username”: <string>, “pass_hash”: <string>, “expiration_cb”: <string>} // expiration_cb es opcional
        - Output:
            - 201 (Created), data: {“token”: <string>, “live_time”: int}
            - 400 (Bad Request)
            - 401 (Unauthorized) // Se puede usar sólo el error 400
- API_ROOT/token/{token}
    - Método DELETE
        - Cabeceras obligatorias: Owner: <string>
        - Output:

                - 204 (No content)
                - 401 (Unauthorized)
                - 404 (Not Found) // Se puede usar sólo el error 401
    - Método GET
        - Output:
            - 200 (OK), data: {“username”: <string>, “roles”: [<string>]}
            - 401 (Unauthorized)
            - 404 (Not Found) // Se puede usar sólo el error 401

La capa de presentación deberá tener en cuenta lo siguiente:

- Cuando al crear un nuevo token, la solicitud incluya “expiration_cb”, el servicio enviará una petición PUT al callback especificado, en el cuerpo del mensaje enviará: {“token”: <string>} con el token que ha caducado. Se esperará una respuesta tipo 2XX, pero en caso negativo, el token caducará igualmente, aunque se deberá informar en los logs del servicio de esta circunstancia.
- Al crear un nuevo token, se devuelve el “live_time” que es el tiempo en segundos que tendrá validez el token antes de ser revocado automáticamente por el servicio.

4. El servidor

El servicio de autenticación se implementará mediante un servidor, que aceptará las siguientes opciones:

- “-p <puerto> o “--port” <puerto>”: establece un puerto de escucha, si no se establece por defecto será el 3002.
- “-l <dirección>” o “--listening <dirección>”: establece una dirección de escucha, por defecto se usará “0.0.0.0”.

5. Pruebas

Se crearán una serie de pruebas unitarias que deberán poder ejecutarse de forma automática dentro del entorno virtual apropiado. Por tanto, el proyecto incluirá un archivo *requirements.txt* con los datos necesarios para crear ese entorno. Las pruebas deberán tener las siguientes características:

- La clase de la capa de negocio deberá probarse con un >90% de cobertura.
- El código del servidor y la capa de presentación en >70%.