**Universidad de Jaén.**

**Escuela Politécnica Superior de Linares.**

# PRÁCTICA 4.

# IMPLEMENTACIÓN DE UN SERVICIO BÁSICO DE AUTENTICACIÓN BASADO EN MAC

***Fecha de inicio: domingo, 20 de mayo de 2018***

***Asignatura:***

- ***Aplicaciones Telemáticas para la Administración.***
  ***(Grado en Ingeniería Telemática)***

***Profesor:***

- ***Raquel Viciana Abad (Departamento Ingeniería Telemática)***

***Integrantes:***

- ***Juan Núñez Lerma.***
- ***Fernando Cabrera Caballero.***

En esta práctica se nos pide introducir un mecanismo de autentificación al servicio desarrollado en la práctica 3, que incorpore un Código de Autenticación del Mensaje (MAC). Esta petición deberá incluir información suficiente para comprobar la identidad del usuario y para guardar un registro temporal de cuando tuvo lugar el acceso.

En la Aplicación cliente se realiza una petición HTTP (GET) al servidor en la que se adjunta: Nombre de usuario, DNIe, Fecha y hora de la petición en formato de texto, Clave pública del certificado de autenticación., Código hash generado con los datos anteriores y la clave del servicio (clave secreta compartida con el servidor). Tanto la clave pública como el código hash deben de codificarse en base64. A continuación vemos el código donde obtenemos los datos y codificamos en base 64:

```
//Obtenemos la fecha y la pasamos a B64
Date fecha = new Date();
System.out.println(fecha);
String fechaString1 = fecha.toString();
String fechaString=Base64.getEncoder().encodeToString(fechaString1.getBytes());

String hash=nick+dni+fechaString/*+clavePublica*/+claveServicio;
System.out.println("cadena a hacer hash: " + hash);
MessageDigest sha256=MessageDigest.getInstance("SHA-256");
sha256.update(hash.getBytes("UTF-8"));
String hashB64=Base64.getEncoder().encodeToString(sha256.digest()); //2bb80d5...527a25b
System.out.println(hashB64);
```

Para realizar la petición GET debemos mandar en la url los datos anteriores (usuario,hash…). Empleamos la clase URLConnection para hacer la conexión con el servidor. Veamos el código implementado:

```
    URL url = new URL("http://localhost:8081/p4/autenticar?nick="+nick+"&dni="+dni+"&fechaString="+fechaString+"&hashB64
                    //+"&clavePublicaB64="+clavePublicaB64+"&hashB64="+hashB64);
URLConnection con = url.openConnection();
```

*Debido a los problemas con la librería del certificado no podemos obtener ni la firma ni la clave pública.*

Una vez que los datos han sido enviados en la url, el cliente recibe la respuesta del servidor, la procesa al igual que en la práctica 3 y nos aparece una ventana emergente dependiendo de la respuesta.

En cuanto al servidor, implementamos el servicio para que obtenga los parámetros enviados desde el cliente:

```
@RequestMapping(value = "/autenticar", method = {RequestMethod.POST,RequestMethod.GET})
public String login(HttpServletRequest request,Model model){
String respuestaServidor=null;
String response=null;
     //URL
     String url="";
    //Variables donde vamos a guardar los atributos introducidos en la url
    String nick,dni,fechaString,hashB64;

    String nick1 = request.getParameter("nick");
    dni = request.getParameter("dni");
    String fechaString1=request.getParameter("fechaString");
    hashB64=request.getParameter("hashB64");
    System.out.println(hashB64);
```

Debido a los problemas generados al enviar los datos en la url tenemos que enviar desde el cliente usuario y fecha codificados también en base64. Por tanto, en el servidor para poder buscar la clave secreta asociada al usuario y al dni debemos decodificar los datos.

```
Base64.Decoder decoder = Base64.getDecoder();
byte[] decodedByteArray = decoder.decode(nick);

String nick1 = new String(decodedByteArray);

decodedByteArray = decoder.decode(fechaString);
String fechaString1 = new String(decodedByteArray);
```

Generamos en el DAOUsuarios un método para obtener la clave privada a partir del dni y usuario. Para esto en la tabla usuarios de la práctica 3 añadimos una nueva columna en la cual se encuentra la clave privada de cada usuario registrado.

```
//Obtener clave secreta
public String obtenerclave(String nick,String dni){
        String sql = "select * from usuarios where Usuario = ? AND DNI=?";
        Object[ ] parametros = {nick,dni}; //Array de objetos
        Mapper mapper = new Mapper();
        List <Usuario> usuarios =  this.jdbcTemplate.query(sql, parametros, mapper);
        if (usuarios.isEmpty()) return null;
        else  return usuarios.get(0).getClaveSecreta();

}
```

Una vez comprobado que usuario y dni se encuentran en BBDD y obtenida la clave pública, pasamos a generar el hash con los datos recibidos en la url y comprobamos si este coincide con

el hash recibido. En este caso mandamos la respuesta al cliente 200 OK, si no mandamos otros códigos de error dependiendo del error.

```
if(dao.buscaUsuario(nick,dni) !=null ){
    String clavesecreta=dao.obtenerclave(nick,dni);

   //Falta clavepublica

   //Hacemos el hash
   String hashobtenido=nick1+dni+fechaString1+clavesecreta;
   MessageDigest sha256 = null;
   try {
       sha256 = MessageDigest.getInstance("SHA-256");
   } catch (NoSuchAlgorithmException e) {
       // TODO Auto-generated catch block
       e.printStackTrace();
   }
    try {
       sha256.update(hashobtenido.getBytes("UTF-8"));
   } catch (UnsupportedEncodingException e) {
       // TODO Auto-generated catch block
       e.printStackTrace();
   }
    String hashserv=Base64.getEncoder().encodeToString(sha256.digest()); //2bb80d5...527a25b


       //Caso que el hash sea distinto
         //Caso que el hash sea distinto
         if(hashB64==null || !hashB64.equals(hashserv) ){
          respuestaServidor="401 UNAUTHORIZED";
          model.addAttribute("respuestaServidor",respuestaServidor); //Enviamos la respuesta al jsp.

          response="Error hash ";//Mensaje
          url="autentication";//Nos vamos al jsp Autentication
      }else{
              //si coincide usuario y password
              //Muestro el jsp con la info de bddd
              //Por tanto hay que recorrer la lista
              respuestaServidor="200 OK";
              model.addAttribute("respuestaServidor",respuestaServidor); //Enviamos la respuesta al jsp.
              response="Autenticado con exito";//Mensage
              url="autentication";//Nos vamos al jsp Autentication
      }
   }  else{
              //Caso que no coincidan
      respuestaServidor="400 BAD REQUEST"; //solicitud incorrecta
      model.addAttribute("respuestaServidor",respuestaServidor); //Enviamos la respuesta al jsp.
      response="Usuario no registrados en BBDD ";//Mensage
      url="autentication";//Nos vamos al jsp Autentication
         }

      //logger.info((response+" Respuesta "+respuestaServidor); //Informamos del suceso.
      return url;

  }
```

Repositorio Servidor:

https://github.com/FernandoCabrera/AATTpADMON_1718_P4_SERVER

Repositorio Cliente: https://github.com/JuanNunezLerma/AATTpADMON_1718_P04