

# Spark Secure App (MicroSpark)

Julián Eduardo Arias Barrera

Octubre 2020

## 1 Introducción

Esta aplicación permite comunicarse de un servidor a otro a través de https utilizando un certificado.

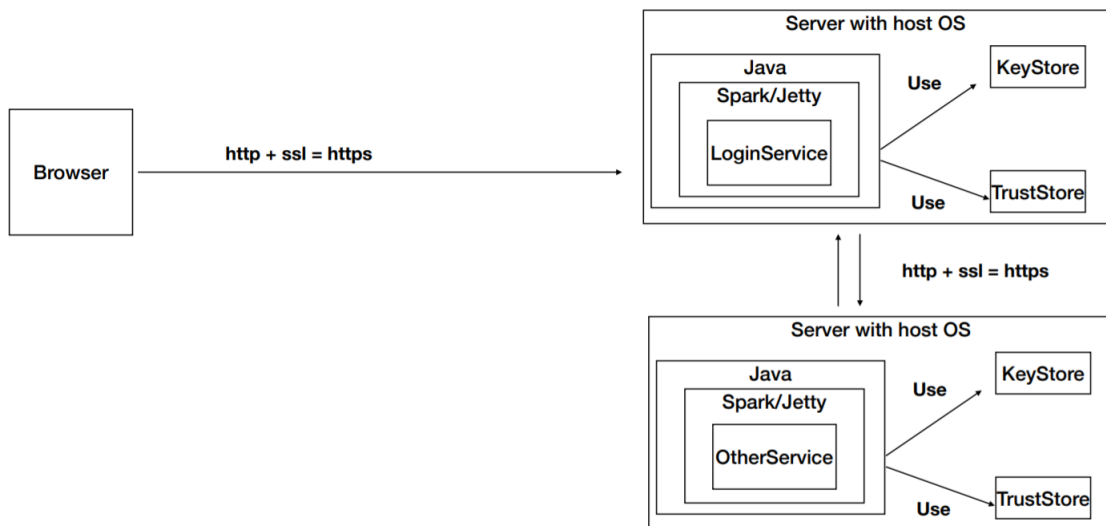


Figure 1: Arquitectura

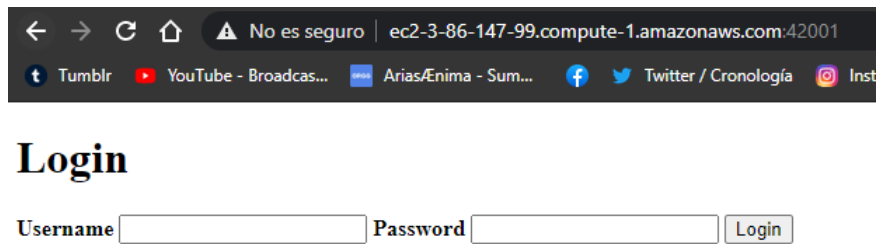
## 2 Ejecución

Con la terminal debemos ejecutar los siguientes comandos:

- Podemos entrar a la aplicación inicialmente en :

```
> https://ec2-3-86-147-99.compute-1.amazonaws.com:42001/
```

- Encontraremos esta vista de login: podremos entrar con el usuario Eduardo y clave :  
miclave



← → ↻ 🏠 No es seguro | ec2-3-86-147-99.compute-1.amazonaws.com:42001

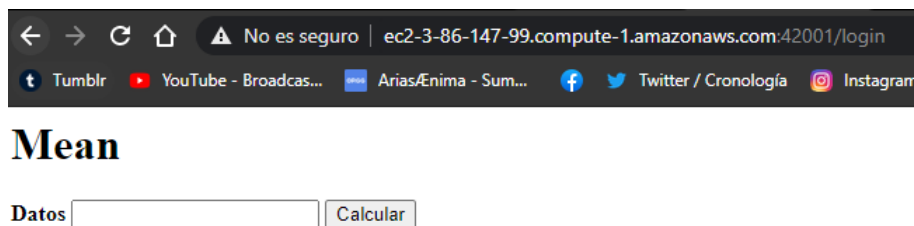
Tumblr YouTube - Broadcas... Arias/Enima - Sum... Twitter / Cronología Instagram

# Login

Username  Password

Figure 2: Login

- Una vez autenticado: (Separado por espacios)



← → ↻ 🏠 No es seguro | ec2-3-86-147-99.compute-1.amazonaws.com:42001/login

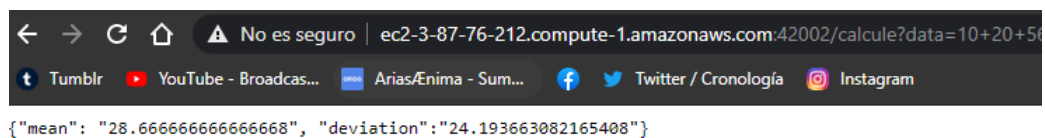
Tumblr YouTube - Broadcas... Arias/Enima - Sum... Twitter / Cronología Instagram

# Mean

Datos

Figure 3: Vista de escritura de datos

- La otra maquina:



← → ↻ 🏠 No es seguro | ec2-3-87-76-212.compute-1.amazonaws.com:42002/calcul?data=10+20+50

Tumblr YouTube - Broadcas... Arias/Enima - Sum... Twitter / Cronología Instagram

```
{ "mean": "28.666666666666668", "deviation": "24.193663082165408" }
```

Figure 4: Vista de escritura de datos

### 3 Diseño

Hay un servicio de login que verifica que la clave encriptada coincida con el usuario guardado, si es así guarda atributos en la sesión y devuelve un formulario para intentar utilizar el servicio de la otra maquina.

```

try {
    MessageDigest md5 = MessageDigest.getInstance("MD5");
    md5.update(StandardCharsets.UTF_8.encode(req.queryParams("password")));
    enc=String.format("%032x", new BigInteger(1, md5.digest()));

} catch (NoSuchAlgorithmException ex) {
    Logger.getLogger(LoginSparkService.class.getName()).log(Level.SEVERE, null, ex);
}
if(req.queryParams("username").equals(username) && enc.equals(password) ){
    req.session().attribute("username",req.queryParams("username"));
    req.session().attribute("encpassword",enc);
    req.session().attribute("auth",true);
    return "<html>\n" +
        "    <head>\n" +
        "        <title>Mean service</title>\n" +
        "        <meta charset=\"UTF-8\">\n" +
        "        <meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0\">\n" +
        "    </head>\n" +
        "    \n" +
        "    <h1>Mean</h1>\n" +
        "    <form method=\"get\" action=\"/calcule\">\n" +
        "        <label for=\"data\"><b>Datos</b></label>\n" +
        "        <input type=\"text\" name=\"data\"/> \n" +
        "        <button type=\"submit\">Calcular</button>\n" +
        "    </form>\n" +
        "    \n" +
        "</html>";
}else{
    return "{\"message\":\"Fail! \"}";
}

```

Figure 5: Servicio Login

Si el login falla mostrara un json con error.

En el servicio calcule verifica que exista un usuario autenticado en la sesión, si es así , se loggea en la segunda maquina enviando la contraseña encriptada y finalmente pide el servicio al a segunda maquina.

```

get("/hello", (req, res) -> "Hello Web services");
post("/login", (req, res) -> { return login(req); });
get("/calcule", (req, res) -> {
    if(req.session().attribute("auth")){
        String url="https://ec2-3-87-76-212.compute-1.amazonaws.com:42002/login?username="+req.session().attribute("username")+"&password="+req.session().attribute("encpassword");
        URLReader.read(url); // login
        return URLReader.read("https://ec2-3-87-76-212.compute-1.amazonaws.com:42002/calcule?" + req.queryString()); // el servicio
    }else{
        return "No tiene permiso a este servicio";
    }
});
}

public static String login(Request req) {
    String enc="";
    try {
        MessageDigest md5 = MessageDigest.getInstance("MD5");

```

Figure 6: Servicio Calcule en la primera maquina

En la segunda maquina existe un login simple que responde si se pudo loggear y guardar atributos en la sesión.

Solo corre el servicio de calcular si hay una sesión autenticada (tuvo que loggearse anteriormente la primera maquina)

```

private static void configureCalculator() {
    get("/calcular", (req, res)
        -> {
            res.status(200);
            res.type("application/json");
            List<Double> respondedata=calculateAInput(req.queryParams("data"),new Calculator.DoubleMath[]{MEAN,DEVIATION});
            if(respondedata!=null && req.session().attribute("auth")!=null){
                return "{\"mean\": \"\"+respondedata.get(0)+"\", \"deviation\": \"\"+respondedata.get(1)+"\"}";
            }else{
                return "{\"message\": \"no autorizado\"}";
            }
        });
}

public static String login(Request req) {

    if (req.queryParams("username").equals(username) && req.queryParams("password").equals(password)) {
        req.session().attribute("username", req.queryParams("username"));
        req.session().attribute("auth", true);
        return "{\"message\": \"Sucess login! \"}";
    } else {
        return "{\"message\": \"Fail! \"}";
    }
}
}

```

Figure 7: Servicio de dev. Std.

## 4 Conclusiones

Las certificaciones resultan ser utiles para comprobar que los dominios son seguros y no vienen de un DNS "suplantado".