Arquitectura de una aplicación distribuida segura (2019)

Karen Paola Duran Vivas. Autor

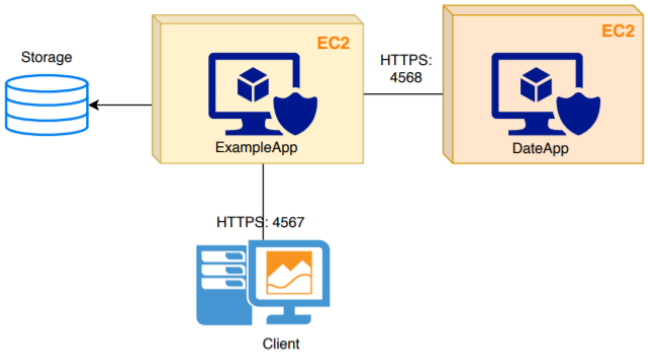
***Resumen*— En el siguiente documento se encuentra la información relacionada con el procedimiento que se llevó a cabo para desarrollar una aplicación en Java con spark y Maven que se conectará a otra aplicación de manera que se garanticen los principios de autenticación, integridad y autorización. El ejercicio trata de desplegar ambas aplicaciones en AWS, y lograr que se comuniquen entre ellas.**

***Palabras clave*—**Seguridad, Certificados, protocolo HTTPS, aplicación distribuida segura.

I. Introducción

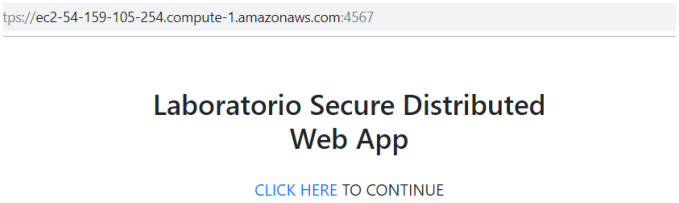
En este documento se explica el desarrollo de una arquitectura para un ejemplo de aplicación segura desarrollada en Java usando spark y maven, en donde habrá un registro para los usuarios, la verificación de credenciales mediante una página login en donde después de haber ingresado a la aplicación con sus respectivas credenciales ésta aplicación se conecta mediante el protocolo HTTPS a otra aplicación que devuelve la hora. Ambas aplicaciones se encuentran desplegadas en AWS, en instancias de EC2 con sus respectivos certificados de seguridad para garantizar autorización, autentificación e integridad.

II. Arquitectura

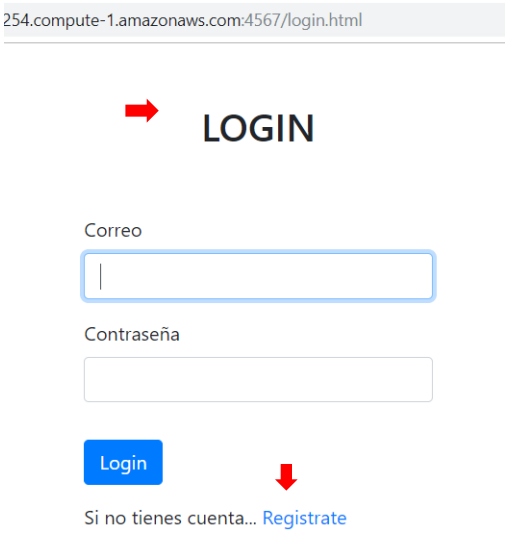


La arquitectura no es tan complicada, tenemos un browser que es el que hace la petición a la aplicación principal, en esta aplicación se encontrarán los servicios de autenticación y registro de usuarios, también se encontrará la persistencia de los usuarios que en este caso se desarrolló como almacenamiento en memoria. Cuando un usuario accede a la aplicación principal, ésta debe comunicarse con la aplicación que devuelve la fecha para mostrársela en pantalla al usuario.

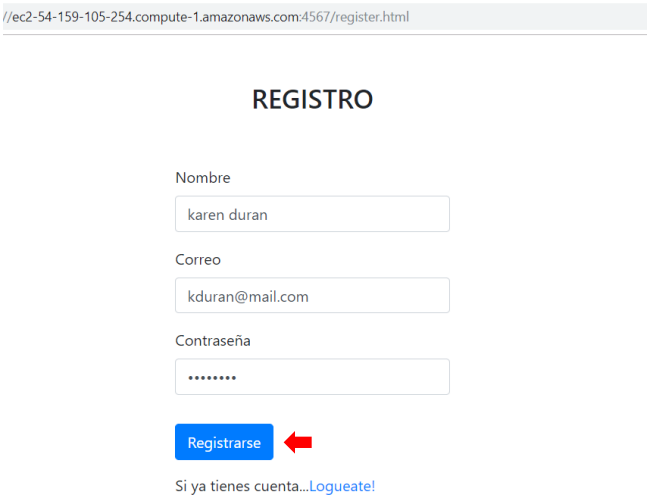
A continuación, ingresaremos a la aplicación principal.



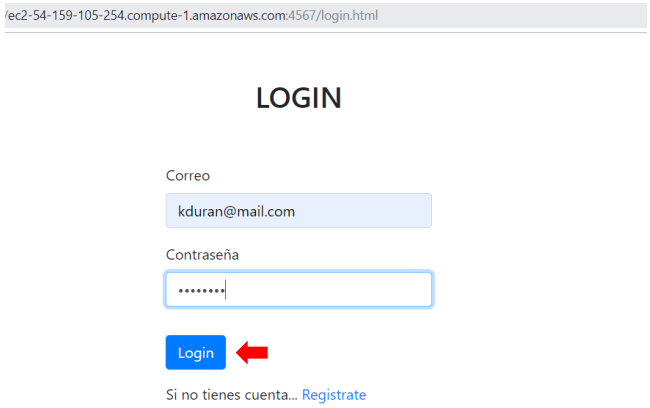
El siguiente paso es dar click en “CLICK HERE” para poder continuar, la aplicación nos redigirá al login.



Como no tenemos ningún usuario, entonces vamos a crear uno en registro.

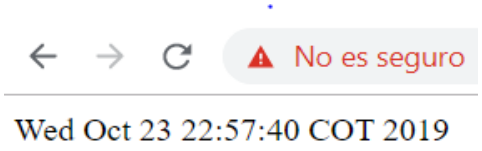


Cuando el usuario se registra, la aplicación cifra la contraseña y la almacena cifrada, porque de esta forma se puede garantizar confidencialidad e integridad debido a que la contraseña original en texto plano no podrá ser visible tan fácilmente.
Después de haberlo creado, la aplicación nos redireccionará a la página de login, otra vez.



Y ahora sí, nos logueamos en la página. Al momento de ingresar a la aplicación, como la contraseña está cifrada, lo que se hace es comparar la contraseña que acaba de ingresar el usuario con la que está almacenada, si son diferentes rechaza el ingreso porque esto significaría que los hash son distintos y por tanto las contraseñas también. Con esta situación se está garantizando el principio de autorización, ya que sólo aquellos usuarios que se hayan registrado anteriormente podrán tener acceso a la aplicación. Después de haber puesto las credenciales correctamente, debería aparecernos la fecha.

Con el fin de garantizar integridad, autorización y autenticación se generaron certificados para cada aplicación, esto se hizo mediante una keystore que almacena certificados propios y también se generó para cada aplicación un truststore como indicador de confiabilidad en cada servicio.



Wed Oct 23 22:57:40 COT 2019

III. Conclusión

Este tipo de implementaciones pueden verse algo complicadas al principio, sobre todo porque hay que garantizar el hecho de que sean seguras, sin embargo lo principalmente importante es el uso de los certificados y entender como funcionan, ya con este laboratorio se entiende que para lograr que dos aplicaciones se comuniquen de una forma segura y lograr que se reconozcan los certificados se deben intercambiar los truststore, de este modo cuando se quiera verificar si es quien dice ser, este proceso se haga sin mayor problema. El protocolo HTTPS nos garantiza una conexión segura entre los servidores y entre el cliente-servidor, cifrando la comunicación entre estos, del mismo modo el cifrado de la contraseña en la aplicación principal también es muy importante porque estamos tratando de garantizar integridad, autorización y autenticación, entonces entre más medidas de seguridad se tomen mucho mejor.

Referencias

- https://github.com/tipsy/spark-ssl
- https://docs.oracle.com/cd/E19798-01/821-1841/gjrgy/
- https://docs.oracle.com/cd/E19509-01/820-3503/ggfen/index.htmlhttps://aws.amazon.com/es/serverless/build-a-web-app/