**22 - Sesión**

El servidor hace uso de las sesiones de acuerdo a la especificación "[State Management Mechanism](http://tools.ietf.org/html/rfc6265)” del protocolo HTTP el cual almacena la información de la sesión en formato de cookies cifradas (ver [Plug.Session.COOKIE](http://hexdocs.pm/plug/)) las cuales tienen cada una un periodo máximo de vida de 10 minutos en estado inactivo. Se entiende por estado inactivo de la sesión como el lapso de tiempo en el que fue realizada la ultima petición al servidor y la hora actual del mismo.

Debido al nivel de seguridad y de rastreo de actividad relacionado con el trato de la información ya sea de consulta y/o proecsamiento, todas las sesiones de la aplicación corresponden a usuarios no anónimos con lo cual se asegura que que cada sesión activa en la aplicación corresponde a una entidad válida.

**Configuración de inactividad:**

Rails:

YourApp::Application.config.session\_store :cookie\_store,

:key => ‘psrdd\_session',

:expire\_after => 10.minutes

Elixir:

ok = cowboy\_session\_config:set([

{cookie\_name, <<“psrdd\_session">>},

{expire, 6000}

])

**Diagrama de Inicio de Sesión exitoso:**

**Expiración de sesión por inactividad.**

En cada petición, el servidor realiza una validación de sesión en la que se verifica que ésta se encuentra activa. Si el resultado de tal validación es correcta el servidor de aplicaciones permite el flujo normal al recurso solicitado garantizando así la funcionalidad de la aplicación, en caso contrario el flujo de la petición es redirigido al formulario de autenticación para que se pueda crear nuevamente la sesión si es que ésta no existe o ya se encuentra caducada.

**Diagrama de Petición con sesión no válida:**

**23 - Línea base de seguridad**

**Implementación de autenticación de los usuarios (internos o clientes)**:

Conforme a lo especificado en el punto *28 - Consulta de información*en la autenticación se involucra el uso de la FIEL del usuario.

**Implementación de mecanismo de no repudio de transacciones**:

Relacionado al punto anterior (Autenticación de usuarios) se garantiza que Diverza, no almacena la llave privada de la FIEL de ningún cliente siendo este el factor que garantiza que la entidad que interactua con *la aplicación* es el propietario de la información a la cual se le otorga el acceso.

**Protección contra inyección de código**:

Como se detalla en el punto *26 - Validación de Seguridad del Aplicativo* se garantiza la seguridad de inyección de código mediante mecanismos de validación y escape de la información.

**Inicio de sesión seguro**:

Se garantiza la seguridad en el acceso a la aplicación al proteger los datos enviados/recibidos la aplicación mediante el uso de un canal cifrado basado en HTTP/TLS como se menciona en el punto *26 - Validación de Seguridad del Aplicativo (A8: Insufficient Transport Layer Protection)*

**Validación de datos de entrada / salida para evitar errores en el procesamiento de la información**:

Para garantizar la prevención de errores en el procesamiento de la información se ha unificado la codificación de los datos que la aplicación procesa. La codificación es UTF-8

Los valores de entrada son validados sintáctica y semánticamente de acuerdo al tipo de dato que mejor lo represente, al igual que los datos de salida los valores contenidos son evaluados para que los caracteres especiales o considerado de riesgo sean escapados para evitar la inyección de código ajeno a la aplicación. Existe el término de “Dato Requerido” para referirse a aquellos datos que mas allá de ser informativos, son utilizados por la aplicación.

**Manejo de errores**:

Existen dos formatos de errores:

Servicios: Se refiere a un formato JSON generado como respuesta de una petición errónea mediante un servicio REST.

Web: Se refiere a un mensaje en la sección de notificaciones que pertenece a la estructura del sitio web de la aplicación.

En ambos casos se muestra un código de error de uso interno con el cual clasifica el tipo de error ocurrido seguido de un mensaje con la descripción del fallo en el cual se utiliza un lenguaje poco técnico para facilitar la interpretación del fallo presentado.

Adicionalmente a la notificación al usuario de la aplicación, los errores son escritos en el Log como evidencia del fallo en el que ademas se registra la fecha y hora del servidor.

Aunque los errores generalmente son acuse de un mal funcionamiento y requieren de registro, existen errores de los cuales no es trascendente dejar evidencia, por ejemplo, si un usuario desea modificar su contraseña pueden presentarse estos dos casos en los que se abordan dos tipos de errores:

* *En el formulario, la nueva contraseña y la verificación de contraseña no coinciden*: Se genera un error y se notifica al usuario, como se realiza mediante la aplicación web el mensaje se muestra en la área de notificaciones pero no es necesario dejar evidencia de que el usuario ha tecleado mal su nueva contraseña.
* *Los datos introducidos por el usuario son validos pero la aplicación no puede conectarse con la Base de datos*: Al ocurrir el error se notifica al usuario acerca del fallo y adicionalmente, se registra en el Log de la aplicación ya que afecta a la operación.

**Endurecer el sistema. (Hardening)**:

**25 - Gestión de la integridad de datos**

...

**26 - Validación de seguridad del aplicativo**

**Implementar controles de validación de seguridad de la aplicación.**

**A1: SQL Injection**:

Es garantizada mediante el uso de la biblioteca Ecto la cual funciona como intermediario entre la aplicación y la Base de Datos. El uso de la biblioteca garantiza la validación y procesamiento de las sentencias SQL que se envían al servidor de Base de Datos mediante el uso de un DSL (Domain Specific language por sus siglas en inglés) el cual evalua los valores variables para escapara los caracteres utilizados en el ataque de SQL Injection.

**A2: Cross-site Scripting**:

La información que se procesa en la aplicación se considera segura de este ataque debido a que los datos recolectados por cualquier mecanismo de entrada (a la aplicación) son validados semántica y sintácticamente para reducir los riesgos en su procesamiento. Adicionalmente a lo anteriormente mencionado y para el caso de recibir campos de texto libre en los cuales no pueda establecerse un formato de validación, se aplica santificación de texto en los cuales se escapan los caracteres que podrían servir para realizar la inyección de código. A continuación se muestra una lista de los caracteres que en los que se aplica escape al ser desplegados en cualquier navegador web:

|  |  |
| --- | --- |
| Caracter de riesgo | Valor de reemplazo |
| & | &amp; |
| < | &lt; |
| > | &gt; |
| " | &quot; |
| ' | &apos; ó &#x27; |
| / | &#x2F; |

**A3: Broken Authentication and Session Management**:

Toda funcionalidad de la aplicación que interactuar con datos sensibles está protegida por la política de sesiones (*ver 22 - Sesión*)

**A4: Insecure Direct Object Reference**:

La información solicitada solo podrá ser accesible si se evalúa que:

* *Se realiza mediante una sesión activa*: Se verifica que la petición se realice mediante una sesión válida y activa (*ver 22 - Sesión*).
* *El solicitante es el propietario de la información*: Se verifica que la información solicitada es propiedad de la entidad que representa la sesión con que se realiza la petición.

**A5: Cross-site Request Forgery**:

...

**A6: Security Misconfiguration**:

Las aplicaciones son publicadas mediante diferentes entornos de ejecución que pueden ser Development, Test y Production, este ultimo entorno es el mas estricto y es con el cual se utiliza para publicar las aplicaciones desarrolladas por Diverza hacia la red externa (Internet).

**A7: Failure to Restrict URL Access**:

Todos los recursos dinámicos están protegidos bajo validación de sesión. Para tener accesos a la funcionalidad de cada recurso se valida que se tenga una sesión valida, activa y que quien realice la consulta a tales recursos sea el propietario de la sesión, para tales efectos nos basamos en la especificación "[State Management Mechanism](http://tools.ietf.org/html/rfc6265)” del protocolo HTTP

**A8: Insufficient Transport Layer Protection**:

La comunicación utilizada para la transmisión de datos es mediante el uso de un canal seguro bajo el protocolo HTTPS de la especificación [HTTP/TLS](http://tools.ietf.org/html/rfc2818). Las llaves de cifrado utilizadas en la comunicación entre el cliente-servidor son vigentes y respaldadas por una CA (Certificate Authority).

**A9: Unvalidated Redirects and Forwards**:

El procesamiento de la información y el trafico de esta dentro de la aplicación, está definido por un Flujo que no depende de datos variables provenientes del cliente, es decir, los flujos de navegación y direccionamiento están explícitos dentro de la aplicación por lo que no se consideran los redireccionamientos a recursos tomados como parámetros entre peticiones