Guidelines:CodingSecure_es

Recomendaciones generales (Planteamiento)

- Adoptar el principio de privilegios mínimos
- No confiar en los datos introducidos por el usuario
- Utilizar opciones predeterminadas seguras
- No depender de la seguridad por medio de la oscuridad
- Validar todo acceso al sistema
- Asumir que los sistemas externos no son seguros
- Reducir el área de exposición
- Cometer errores de forma segura (gestion de excepciones)
- No olvidar que el alcance de la seguridad lo define su punto más débil
- Si no se utiliza, deshabilitarlo
- Programación y seguimiento de los ciclos de pruebas

Guías de desarrollo seguro

Definir el proceso y las necesidades de seguridad

Como primer paso, los desarrolladores han de definir el proceso que utilizarán en el desarrollo y alcance de la seguridad en la aplicación que están diseñando.

Se han de determinar que puntos de control seguridad se van a emplear en el código:

1. Acceso. Que usuario es, y que permisos tiene.
2. Funcionalidades. Conocer que permisos se deben tener para ejecutar segun que funciones de nuestro código.
3. Acceso externo. Asegurarnos de que el código no es accesible fuera de esos ámbitos de seguridad externos.
4. Trazas de control. Verificar que todos los accesos indebidos son debidamente registrados y todos los errores en la verificación de seguridad son registrados.

Métodos de entrada / salida

En el framework que se emplee, deben existir funciones para obtener información del usuario o mostrársela. Habra que usar _SIEMPRE_ estas funciones en vez de las funciones nativas del lenguage, generalmente estas funciones se pueden clasificar como:

1. Entrada de informacion del usuario (safe_input, get_parameter)....
2. Salida de información hacia el usuario (safe_output, render_data, ascii_output...)
3. Registros de información a ficheros log (xxx_log, xxx_audit)
4. Creación de enlaces HTTP o a recursos de imagenes (print_link, print_image).

El uso de estos métodos evita problemas de SQL injection, Cross Site Scripting, Parameter Bashing y

otros problemas habituales en entornos WEB y son de forzado cumplimiento en el 99% de los casos.

Métodos de acceso a base de datos

En el framework que se emplee, deben existir funciones para acceder a la información de la Base de Datos, puede que incluso permitan una capa de abstracción de acceso a la base de datos (Oracle, PostgreSQL, MySQL). Es obligatorio usar _SIEMPRE_ estas funciones en vez de las funciones nativas del lenguage, generalmente estas funciones se pueden clasificar como:

1. Acceso a la informacion (get_db_sql_..., get_db_row...).
2. Inserción o modificación (db_insert, db_update).

El uso de estos métodos evita problemas de SQL injection, Cross Site Scripting, Parameter Bashing y otros problemas habituales en entornos WEB y son de forzado cumplimiento en el 100% de los casos. Esto también permite utilizar técnicas de optimización, que no entran dentro del apartado de seguridad.

Recursos compartidos

En determinadas ocasiones es posible que tengamos que hacer uso de sistemas, tales como ficheros temporales, semáforos, tablas de datos compartidas, u otros mecanismos. Deberemos ser conscientes de que esa información no es segura, y no almacenar ahi ningún dato sensible, como ID de sesión, passwords, o cualquier información que pueda servir a un atacante para escalar privilegios.

Fase de pruebas

Es imprescindible, antes de dar por cerrado un desarrollo, hacer unas pruebas de acceso, donde se probará con un usuario sin privilegios a las zonas de código, para verificar si el sistema de protección funciona.

From:
https://pandorafms.com/manual/ - **Pandora FMS Documentation**

Permanent link:
https://pandorafms.com/manual/guidelines/codingsecure_es

Last update: **2021/11/05 12:05**