Gestión de Vulnerabilidades

Base de conocimiento que contiene guías, procedimientos, manuales y técnicas para la solución/mitigación de vulnerabilidades.

- Servidores DNS
  - Resolución abierta de DNS
  - Deshabilitar transferencia de zona
- Servidores de correo
  - Deshabilitar Open Relay interno y externo en Postfix
- Misceláneo
  - Mitigar enumeración de usuarios OpenSSH
  - Mitigar HearthBleed en OpenSSL
- Aplicaciones web
  - Inyección SQL
  - Deshabilitar phpinfo
  - Cross-Site Scripting XSS
  - Despliegue de mensajes de error
  - Variables de entorno en PHP
- Sistemas de administración de cotenido (CMS)
  - Exposición del archivo XMLRPC

- Servidores web
  - Listado de directorio (Index Of)
  - Certificados SSL/TLS
  - Eliminar la visualización de archivos de configuración
  - Cross-Site Request Forgery CSRF
  - Copias de seguridad
  - Contenido por defecto
  - Implementación de cabeceras de seguridad
  - Quitar archivos de configuración

Servidores DNS

Capítulo destinado a contramedidas para solucionar vulnerabilidades en servidores de nombres de dominio.

Resolución abierta de DNS

La vulnerabilidad Open Resolver DNS es una vulnerabilidad de seguridad que se refiere a los servidores de nombres de dominio (DNS) que han sido configurados de manera que permiten que cualquier persona en Internet realice consultas a través de ellos sin restricciones. Esto significa que un atacante puede enviar solicitudes de consulta DNS a estos servidores en grandes cantidades y utilizarlos como amplificadores en ataques de denegación de servicio distribuido (DDoS) contra otros sistemas en Internet.

En un ataque de DDoS que utiliza la vulnerabilidad Open Resolver DNS, el atacante envía una gran cantidad de solicitudes de consulta DNS falsas a los servidores Open Resolver, y éstos responden a estas solicitudes enviando grandes cantidades de datos a las direcciones de origen de las solicitudes. Al enviar muchas solicitudes falsas desde diferentes direcciones, el atacante puede hacer que los servidores de destino se vean abrumados por el tráfico de red y se vuelvan inaccesibles.

Para evitar la vulnerabilidad Open Resolver DNS, es importante que los administradores de sistemas configuren sus servidores DNS correctamente. Esto incluye restringir el acceso a los servidores de DNS, implementar listas de control de acceso (ACL), limitar el tamaño de los registros de consulta DNS, y actualizar el software de DNS regularmente para corregir vulnerabilidades conocidas.

Configuración DNS en Bind9

Agregue lo siguiente a las opciones globales:

Las opciones globales de BIND9 se definen en el archivo de configuración principal de BIND9, que normalmente se llama `named.conf` o `named.conf.options`. Este archivo de configuración principal generalmente se encuentra en el directorio `/etc/bind/` o `/etc/named/` en la mayoría de las distribuciones de Linux.

```
options {
    allow-query-cache { none; };
    recursion no;
};
```

Configuración DNS de Microsoft

En la herramienta de consola DNS de Microsoft:

a. Primero, haga clic derecho en el servidor DNS y haga clic en Propiedades.
b. Después de eso, haga clic en la pestaña Avanzado.
c. Finalmente, en las opciones del servidor, seleccione la casilla de verificación "Deshabilitar recursividad" y luego haga clic en Aceptar.

Configuración en postfix 3.x

En el archivo de configuración /etc/postfix/main.cf

En la variable “mydestination” NO deben estar los dominios locales:

```
 localhost. localdomain, localhost
```

Aumentar los parametros (si no existiesen):

```
mynetworks_style = host
relay_domains =
```

Tambien es valido: mynetworks_style = subnet

Sugerencias a considerar

Utilizar cortafuegos compatibles con el DNS y utilizar protección DDoS de terceros.

Deshabilitar transferencia de zona

La vulnerabilidad de transferencia de zona en los servidores DNS es una vulnerabilidad de seguridad que permite a un atacante obtener copias completas de la zona de nombres de un servidor DNS. La zona de nombres de un servidor DNS es un archivo que contiene información sobre los nombres de dominio y sus correspondientes direcciones IP.

La transferencia de zona es un proceso legítimo utilizado por los servidores DNS para compartir información sobre los nombres de dominio entre ellos. Sin embargo, si la configuración del servidor DNS está mal configurada, un atacante puede aprovechar esta función legítima para obtener una copia completa de la zona de nombres, lo que le permitiría realizar ataques de denegación de servicio o identificar otros vectores de ataque.

Los servidores DNS mal configurados pueden permitir a cualquier persona solicitar una transferencia de zona sin autenticación o permitir la transferencia de zona a cualquier dirección IP. Si un atacante identifica que un servidor DNS es vulnerable a la transferencia de zona, puede utilizar herramientas disponibles públicamente para realizar la transferencia de zona y obtener información sobre los nombres de dominio y direcciones IP.

Es importante que los administradores de sistemas configuren correctamente los servidores DNS para evitar la vulnerabilidad de transferencia de zona y proteger la información confidencial. Las medidas de seguridad recomendadas incluyen restringir el acceso a la transferencia de zona solo a direcciones IP específicas y autenticar las solicitudes de transferencia de zona. Además, es importante mantener el software del servidor DNS actualizado con las últimas actualizaciones de seguridad para evitar vulnerabilidades conocidas.

Deshabilitar transferencia de zona (Bind)

En el archivo `/etc/bind/named.conf` adicionar:

```
options {
   allow-transfer {"none";};
};
```

Servidores de correo

Capítulo destinado a configuraciones específicas para mitigar/solucionar vulnerabilidades, errores de configuración en servidores de correo.

Servidores de correo

Deshabilitar Open Relay interno y externo en Postfix

Guía para evitar que personas no autorizadas envíen correos electrónicos a nombre del dominio de una entidad.

Esta solución es para postfix con autenticación mediante dovecot.

En caso de no tener instalado dovecot procedemos a su instalación:

```
apt install dovecot-core dovecot-imapd
```

Instalamos el plugin pcre de postfix:

```
apt install postfix-pcre
```

Configuramos el archivo /etc/postfix/main.cf de la siguiente manera:

```
# See /usr/share/postfix/main.cf.dist for a commented, more complete version


# Debian specific:  Specifying a file name will cause the first
# line of that file to be used as the name.  The Debian default
# is /etc/mailname.
#myorigin = /etc/mailname

smtpd_banner = $myhostname ESMTP $mail_name (Ubuntu)
biff = no

# appending .domain is the MUA's job.
append_dot_mydomain = no
```

```
# Uncomment the next line to generate "delayed mail" warnings
#delay_warning_time = 4h

readme_directory = no

# TLS parameters
smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
smtpd_use_tls=yes
smtpd_tls_session_cache_database = btree:${data_directory}/smtpd_scache
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
smtpd_tls_auth_only = yes


## SASL implementation
smtpd_sasl_type = dovecot
smtpd_sasl_path = private/auth
smtpd_sasl_auth_enable = yes
smtpd_sasl_security_options = noanonymous
smtpd_sasl_tls_security_options = $smtpd_sasl_security_options


# See /usr/share/doc/postfix/TLS_README.gz in the postfix-doc package for
# information on enabling SSL in the smtp client.

myhostname = dominio.gob.bo
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
myorigin = /etc/mailname
mydestination = $myhostname, localhost.dominio.gob.bo, localhost, dominio.gob.bo
relayhost =
recipient_delimiter = +
inet_interfaces = all
inet_protocols = all
mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128 [IP servidor de correo]
```

```
disable_vrfy_command = yes


# Restricciones


smtpd_helo_required = yes
smtpd_sender_login_maps = pcre:/etc/postfix/controlled_envelope_senders.pcre
smtpd_client_restrictions = permit_mynetworks, reject_unknown_client_hostname,
reject_unknown_reverse_client_hostname, permit
smtpd_recipient_restrictions = permit_mynetworks, permit_sasl_authenticated
,reject_unauth_destination,reject_unknown_sender_domain, reject_sender_login_mismatch, permit
```

Los parámetros a considerar son los siguientes:

- **smtpd_sasl_auth_enable = yes** : Habilitar la autenticación SASL en postfix.
- **smtpd_sasl_type = dovecot** : utilizar dovecot para la autenticación SASL.
- **myhostname = dominio.gob.bo** : Declaran los dominios a los que se autorizará mandar mensajes a través del servidor de correo.
- **mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128 [IP servidor de correo]** : Configurar la lista de direcciones IPs que pueden mandar correos. (Típicamente solo la IP interna del servidor de correo).
- **smtpd_helo_required = yes** : Exige que un cliente SMTP remoto se presente con el comando HELO o EHLO antes de enviar el comando MAIL u otros comandos que requieran negociación EHLO.
- **smtpd_sender_login_maps = pcre:/etc/postfix/controlled_envelope_senders.pcre** : Filtro de búsqueda opcional con el dominio de inicio de sesión SASL que poseen las direcciones del remitente (MAIL FROM).
- **smtpd_client_restrictions** : Restricciones de cliente
  - permit_mynetworks .- Permite el envío de correo a clientes que coincidan con las direcciones descritas en mynetworks.
  - reject_unknown_client_hostname .- Hace una comparación con la IP y nombre del cliente para verificar que coincidan, en caso de no coincidir rechaza la solicitud.
  - reject_unknown_reverse_client_hostname .- Rechaza la solicitud cuando la dirección IP del cliente no tiene asignación de dirección-> nombre.
  - permit : Permitir en caso de no ser rechazado en sentencias anteriores.
- smtpd_recipient_restrictions: Restricciones del recipiente (RCPT)
  - permit_mynetworks .- Permite la solicitud a clientes que coincidan con las direcciones descritas en mynetworks.
  - permit_sasl_authenticated .- Permite la solicitud cuando el cliente se haya autenticado correctamente a través del protocolo RFC 4954 (AUTH).
  - reject_unauth_destination .- Rechaza la solicitud si el que envía no es un retransmisor permitido, ó si el receptor del mensaje no es un destino válido.

- reject_unknown_sender_domain .- Rechaza la solicitud cuando Postfix no es el destino final para la dirección del remitente y el dominio MAIL FROM tiene 1) ningún registro DNS MX ni DNS A, o 2) un registro MX con formato incorrecto, como un registro con un nombre de host MX de longitud cero.
- reject_sender_login_mismatch .- Rechaza la solicitud cuando $ smtpd_sender_login_maps especifica un propietario para la dirección MAIL FROM, pero el cliente no está (SASL) conectado como propietario de la dirección MAIL FROM.
- permit .- Permitir en caso de no ser rechazado en sentencias anteriores.

Adicionalmente es necesario añadir el archivo controlled_envelope_senders.pcre en la dirección establecida, en este caso /etc/postfix/:

```
 #envelop sender        owners (SASL login names)
/^(.*)@dominio\.gob\.bo$/   ${1}
```

Por último es necesario hacer una configuración en dovecot:

Para versiones de dovecot que corresponden a postfix 2.7 se edita el archivo /etc/dovecot/dovecot.conf:

```
client {
      # The client socket is generally safe to export to everyone. Typical use
      # is to export it to your SMTP server so it can do SMTP AUTH lookups
      # using it.
      #path = /var/run/dovecot/auth-client
      path = /var/spool/postfix/private/auth
      mode = 0660
      user = postfix
      group = postfix
    }
```

Para versiones posteriores se edita el archivo /etc/dovecot/conf.d/10-master.conf:

```
service auth {
  # auth_socket_path points to this userdb socket by default. It's typically
  # used by dovecot-lda, doveadm, possibly imap process, etc. Its default
  # permissions make it readable only by root, but you may need to relax these
  # permissions. Users that have access to this socket are able to get a list
  # of all usernames and get results of everyone's userdb lookups.
  unix_listener auth-userdb {
```

```
    #mode = 0600
    #user =
    #group =
  }

  # Postfix smtp-auth
  unix_listener /var/spool/postfix/private/auth {
    mode = 0660
    user = postfix
    group = postfix
  }
```

Misceláneo

Capítulo destinado a la solución, mitigación de software en general.

Misceláneo

Mitigar enumeración de usuarios OpenSSH

Utilizar versiones antiguas de openssh puede permitir la enumeración de usuarios en SSH, por lo cual es recomendable utilizar versiones posteriores a la 7.7 de OPENSSH, a continuación se detallará de cómo realizar la actualización mediante uso de repositorios.

La presente guía fue probada en un servidor Debian 9.7 con versión 7.4 de OpenSSH.

Visualizar la versión de OpenSSH utilizada:

```
$ sshd -V
```

Agregar el repositorio para la última version de openssh:

```
$ nano /etc/apt/source.list
```

Agregar al final:

```
deb http://ftp.de.debian.org/debian sid main
```

Actualizar la lista de paquetes:

```
$ apt update
```

Instalar OpenSSH:

```
$ apt install openssh-server
```

elegir opcion 1 (install the package maintainer’s version)

Verificar la versión instalada:

```
$ telnet localhost 22


SSH-2.0-OpenSSH_8.4p1 Debian-4
```

Si desea puede comentar le repositorio agregado en source.list

```
$ nano /etc/apt/source.list
```

Colocar # al comienzo de la línea añadida:

```
# deb http://ftp.de.debian.org/debian sid main
```

Actualizar la lista de paquetes:

```
$ apt update
```

Misceláneo

Mitigar HearthBleed en OpenSSL

La vulnerabilidad de OpenSSL desactualizado puede permitir a un atacante realizar una variedad de ataques, incluyendo la interceptación de datos cifrados y la inyección de código malicioso. Esta vulnerabilidad puede ser explotada por los atacantes mediante la explotación de una variedad de vulnerabilidades de seguridad conocidas en versiones antiguas de OpenSSL.

Para evitar esta vulnerabilidad, es importante que los sistemas se mantengan actualizados con las últimas versiones de OpenSSL y otros software críticos. También es recomendable realizar auditorías periódicas de seguridad en los sistemas para identificar cualquier vulnerabilidad que pueda estar presente. Además, se deben aplicar medidas de seguridad adicionales, como la utilización de certificados SSL/TLS seguros y la implementación de políticas de autenticación adecuadas, para minimizar el riesgo de ataques.

Actualización OpenSSL en Ubuntu

Esta guía fue probada en un servidor Ubuntu con una versión inicial de OpenSSL de 0.9.8.

Ejecutar el siguiente comando para ver la versión actual de openssl:

```
$ openssl version
```

Y como respuesta nos da la versión y el año de creación del OpenSSL:

```
OpenSSL 0.9.8k 25 mar 2009
```

Descargar desde el repositorio de openssl el archivo al que se actualizará en este caso OpenSSL 1.1.1j:

```
$ https://www.openssl.org/source/openssl-1.1.1j.tar.gz
```

Desempaquetar:

```
$ tar -zxf openssl-1.1.1j.tar.gz
```

Emitir los siguientes comandos para la instalación:

```
$ ./config
$ make // (si el comando make no está instalado ejecutar $ sudo apt install make gcc)
$ make install
```

Creando un enlace desde el binario recién instalado a la ubicación predeterminada:

```
$ sudo ln -s /usr/local/bin/openssl /usr/bin/openssl
$ sudo ldconfig
$ openssl version
```

La respuesta debe ser parecida a la siguiente:

```
OpenSSL 1.1.1j  16 Feb 2021
```

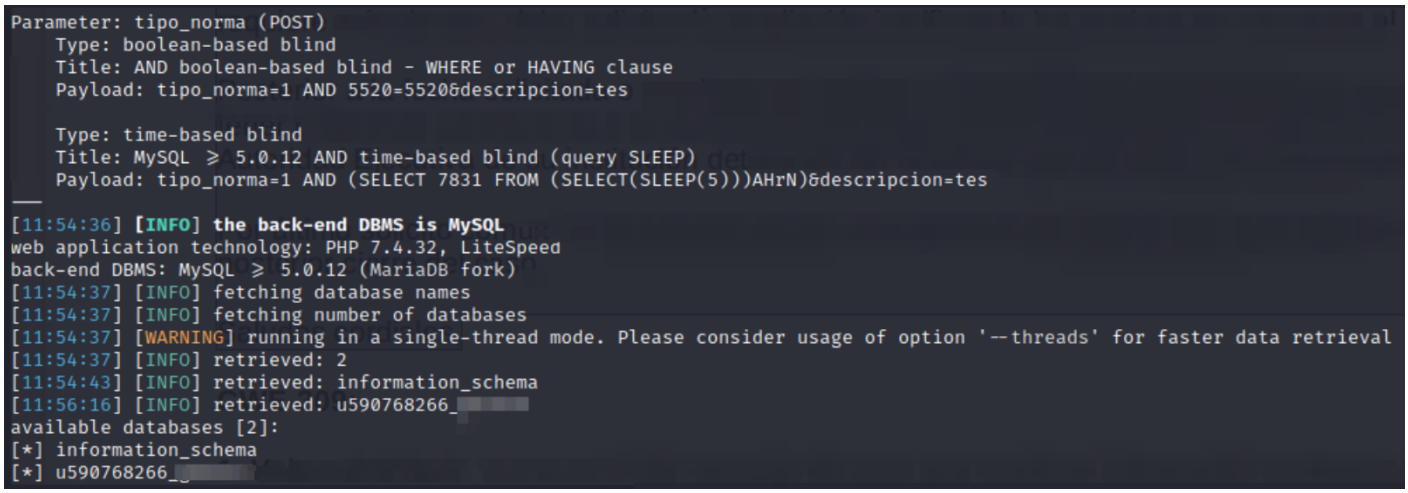
Aplicaciones web

Capítulo destinado a la solución de vulnerabilidades en aplicaciones web.

Inyección SQL

Sin la eliminación o citación suficiente de la sintaxis SQL en las entradas controlables por el usuario, la consulta SQL generada puede hacer que esas entradas se interpreten como SQL en lugar de datos de usuario ordinarios. Esto se puede usar para alterar la lógica de consulta para eludir las comprobaciones de seguridad o para insertar declaraciones adicionales que modifican la base de datos de back-end, posiblemente incluyendo la ejecución de comandos del sistema.

La inyección SQL se ha convertido en un problema común con los sitios web basados en bases de datos. La falla se detecta y explota fácilmente y, como tal, es probable que cualquier sitio o paquete de software con una base mínima de usuarios esté sujeto a un intento de ataque de este tipo. Esta falla depende del hecho de que SQL no hace una distinción real entre los planos de control y de datos.



Inyección SQL PHP - Solución

La solución aplica para aplicaciones que no usan ningún framework como base de desarrollo.

Opción a)

Usar esta opción en versiones de PHP 4 > = 4.3.0 hasta versiones anteriores a PHP 5.5.0.

```
$id=mysqlrealescapestring($GET["id"]); // sanitizando el parámetro id (ejemplo)
```

Opción b)

Ejemplo, valida el parámetro id

```
$id=$GET["id"];
$id=str_replace("'","",$id);
$id=str_replace('"','',$id);
$id=str_replace(')','',$id);
$id=str_replace('-','',$id);
$id=str_replace('%','',$id);
```

Opción c)

Usar consultas parametrizadas
https://www.php.net/manual/es/mysqli.quickstart.prepared-statements.php

También es importante realizar la verificación y validación en el resto de campos de entrada que existan en formularios o parámetros en URL de su sitio web.

Inyección SQL ASP.net - solución

Comience restringiendo la entrada en el código del lado del servidor para sus páginas web ASP.NET. No confíe en la validación del lado del cliente porque se puede omitir fácilmente. Use la validación del lado del cliente solo para reducir los viajes de ida y vuelta y mejorar la experiencia del usuario.

Si usa controles de servidor, use los controles de validación de ASP.NET, como los controles RegularExpressionValidator y RangeValidator para restringir la entrada. Si usa controles de entrada HTML regulares, use la clase Regex en su código del lado del servidor para restringir la entrada.

Opción a)

Puede restringir su entrada usando un control RegularExpressionValidator como se muestra a continuación:

```
<%@ language="C#" %>
<form id="form1" >
    <asp: TextBox ID="SSN" />
    <asp: RegularExpressionValidator ID="regexpSSN"
                                     ErrorMessage="Incorrect SSN Number"
                                     ControlToValidate="SSN"
                                     ValidationExpression="^\d{3}-\d{2}-\d{4}$" />
```

```
</form>
```

Si la entrada de SSN proviene de otra fuente, como un control HTML, un parámetro de cadena de consulta o una cookie, puede restringirla usando la clase Regex del espacio de nombres System.Text.RegularExpressions . El siguiente ejemplo asume que la entrada se obtiene de una cookie.

Opción b)

Usando System.Text.RegularExpressions;

```
if (Regex.IsMatch(Request.Cookies["SSN"], "^\d{3}-\d{2}-\d{4}$"))
{
    // access the database
}
else
{
    // handle the bad input
}
```

Opción c)

Usar parámetros con procedimientos almacenados:

El uso de procedimientos almacenados no impide necesariamente la inyección de SQL. Lo importante es usar parámetros con procedimientos almacenados. Si no usa parámetros, sus procedimientos almacenados pueden ser susceptibles a la inyección SQL si usan entrada sin filtrar como se describe en la sección "Descripción general" de este documento.

El siguiente código muestra cómo usar SqlParameterCollection al llamar a un procedimiento almacenado.

```
using System.Data;
using System.Data.SqlClient;


using (SqlConnection connection = new SqlConnection(connectionString))
{
  DataSet userDataset = new DataSet();
  SqlDataAdapter myCommand = new SqlDataAdapter(
              "LoginStoredProcedure", connection);
```

```
  myCommand.SelectCommand.CommandType = CommandType.StoredProcedure;
  myCommand.SelectCommand.Parameters.Add("@au_id", SqlDbType.VarChar, 11);
  myCommand.SelectCommand.Parameters["@au_id"].Value = SSN.Text;

  myCommand.Fill(userDataset);
}
```

En este caso, el parámetro @au_id se trata como un valor literal y no como un código ejecutable. Además, se comprueba el tipo y la longitud del parámetro. En el ejemplo de código anterior, el valor de entrada no puede tener más de 11 caracteres. Si los datos no se ajustan al tipo o la longitud definidos por el parámetro, la clase SqlParameter genera una excepción.

Opción d)

Usar parámetros con SQL dinámico:

Si no puede usar procedimientos almacenados, aún debe usar parámetros cuando construya sentencias SQL dinámicas. El siguiente código muestra cómo usar SqlParametersCollection con SQL dinámico.

```
using System.Data;
using System.Data.SqlClient;

using (SqlConnection connection = new SqlConnection(connectionString))
{
  DataSet userDataset = new DataSet();
  SqlDataAdapter myDataAdapter = new SqlDataAdapter(
         "SELECT au_lname, au_fname FROM Authors WHERE au_id = @au_id",
         connection);
  myCommand.SelectCommand.Parameters.Add("@au_id", SqlDbType.VarChar, 11);
  myCommand.SelectCommand.Parameters["@au_id"].Value = SSN.Text;
  myDataAdapter.Fill(userDataset);
}
```

Opción e)

Uso de procesamiento por lotes de parámetros:

Un concepto erróneo común es que si concatena varias sentencias SQL para enviar un lote de sentencias al servidor en un solo viaje de ida y vuelta, no puede usar parámetros. Sin embargo, puede usar esta técnica si se asegura de que los nombres de los parámetros no se repitan. Puede hacer esto fácilmente asegurándose de usar nombres de parámetros únicos durante la concatenación de texto SQL, como se muestra aquí.

```
using System.Data;
using System.Data.SqlClient;
. . .
using (SqlConnection connection = new SqlConnection(connectionString))
{
  SqlDataAdapter dataAdapter = new SqlDataAdapter(
       "SELECT CustomerID INTO #Temp1 FROM Customers " +
       "WHERE CustomerID > @custIDParm; SELECT CompanyName FROM Customers " +
       "WHERE Country = @countryParm and CustomerID IN " +
       "(SELECT CustomerID FROM #Temp1);",
       connection);
  SqlParameter custIDParm = dataAdapter.SelectCommand.Parameters.Add(
                                          "@custIDParm", SqlDbType.NChar, 5);
  custIDParm.Value = customerID.Text;

  SqlParameter countryParm = dataAdapter.SelectCommand.Parameters.Add(
                                      "@countryParm", SqlDbType.NVarChar, 15);
  countryParm.Value = country.Text;

  connection.Open();
  DataSet dataSet = new DataSet();
  dataAdapter.Fill(dataSet);
}
. . .
```

Inyección SQL JAVA - solución

La solución más simple es usar PreparedStatement en lugar de Statement para ejecutar la consulta.

En lugar de concatenar el nombre de usuario y la contraseña en la consulta, los proporcionamos para consultar a través de los métodos de establecimiento de PreparedStatement.

Ahora, el valor del nombre de usuario y la contraseña recibidos de la solicitud se tratan solo como datos, por lo que no se producirá una inyección SQL.

Veamos el código del servlet modificado.

```
String query = "select * from tbluser where username=? and password = ?";
Connection conn = null;
PreparedStatement stmt = null;
//Las credenciales utilizadas son de ejemplo, se recomienda utilizar contraseñas robustas
try {
    conn = DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/user", "root", "root");
    stmt = conn.prepareStatement(query);
    stmt.setString(1, username);
    stmt.setString(2, password);
    ResultSet rs = stmt.executeQuery();
    if (rs.next()) {
        // Login Successful if match is found
        success = true;
    }
    rs.close();
} catch (Exception e) {
    e.printStackTrace();
} finally {
    try {
        stmt.close();
        conn.close();
    } catch (Exception e) {
    }
}
```

Entendamos lo que está pasando en este caso.

Consulta : seleccione * de tbluser donde nombre de usuario =? y contraseña = ?

El signo de interrogación (?) en la consulta anterior se denomina parámetro posicional. Hay 2 parámetros posicionales en la consulta anterior. No concatenamos nombre de usuario y contraseña para consultar. Usamos métodos disponibles en PreparedStatement para proporcionar información de usuario.

Hemos configurado el primer parámetro usando stmt.setString(1, username) y el segundo parámetro usando stmt.setString(2, password). La API de JDBC subyacente se encarga de desinfectar los valores para evitar la inyección de SQL.

Deshabilitar phpinfo

La función phpinfo() en PHP muestra información detallada sobre la configuración de PHP instalada en el servidor, incluyendo la versión de PHP, módulos cargados, variables de entorno y configuración del servidor. Si esta función se deja activa en un servidor web público, puede ser una vulnerabilidad de seguridad.

Esto se debe a que los atacantes pueden utilizar la información expuesta en la función phpinfo() para encontrar vulnerabilidades conocidas en la versión de PHP, los módulos instalados o la configuración del servidor. Los atacantes pueden utilizar esta información para desarrollar ataques específicos que exploten las vulnerabilidades conocidas.

Por lo tanto, es una buena práctica deshabilitar la función phpinfo() en servidores web públicos o restringir el acceso a esta función solo a usuarios de confianza. Se recomienda también mantener actualizada la versión de PHP y sus módulos instalados para reducir el riesgo de vulnerabilidades conocidas.

phpinfo()

PHP Version 5.5.9-1ubuntu4.29

System	Linux SitioWebBackup20221230 4.15.18-10-pve #1 SMP PVE 4.15.18-32 (Sat, 19 Jan 2019 10:09:37 +0100) x86_64
Build Date	Apr 22 2019 18:33:42
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/apache2
Loaded Configuration File	/etc/php5/apache2/php.ini
Scan this dir for additional .ini files	/etc/php5/apache2/conf.d
Additional .ini files parsed	/etc/php5/apache2/conf.d/05-opcache.ini, /etc/php5/apache2/conf.d/10-pdo.ini, /etc/php5/apache2/conf.d/20-gd.ini, /etc/php5/apache2/conf.d/20-json.ini, /etc/php5/apache2/conf.d/20-mysql.ini, /etc/php5/apache2/conf.d/20-mysqli.ini, /etc/php5/apache2/conf.d/20-pdo_mysql.ini, /etc/php5/apache2/conf.d/20-readline.ini, /etc/php5/apache2/conf.d/mcrypt.ini
PHP API	20121113
PHP Extension	20121212
Zend Extension	220121212
Zend Extension Build	API220121212,NTS
PHP Extension Build	API20121212,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	enabled
Registered PHP Streams	https, ftps, compress.zlib, compress.bzip2, php, file, glob, data, http, ftp, phar, zip
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, sslv3, tls
Registered Stream Filters	zlib.*, bzip2.*, convert.iconv.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, mcrypt.*, mdecrypt.*

Deshabilitar phpinfo en Linux

Abrir el archivo de configuración de PHP (php.ini).

Rutas por defecto en servidores

1. En servidores apache el archivo php.ini se encuentra en la siguiente ruta: `/etc/php/[VERSION]/apache2/`
2. En servidores Nginx el archivo php.ini se encuentra en la siguiente ruta: `/etc/php/[VERSION]/fpm`

Una vez encontrado el archivo php.ini se debe realizar lo siguiente:

Buscar la línea que contiene la directiva `"disable_functions"` y agregue `"phpinfo"` a la lista separada por comas de funciones deshabilitadas.

```
disable_functions = phpinfo
```

Guarde el archivo php.ini y reinicie su servidor web para que los cambios surtan efecto.

Reiniciar Apache:

Abre una línea de comandos en el servidor.
Ejecuta el siguiente comando para reiniciar el servicio de Apache:

```
//para versiones modernas de Linux que usan systemd

sudo systemctl restart apache2

//para versiones antiguas de Linux

sudo service apache2 restart
```

Reiniciar Nginx:

Abre una línea de comandos en el servidor.
Ejecuta el siguiente comando para reiniciar el servicio de Nginx:

```
//para versiones modernas de Linux que usan systemd
```

```
sudo systemctl restart nginx


//para versiones antiguas de Linux


sudo service nginx restart
```

Deshabilitar phpinfo en Windows

Ubicación del archivo php.ini en Windows utilizando XAMPP

1. Abre el cmd.
2. Luego dirigirse a la carpeta donde esta instalada PHP en su unidad c. Por ejemplo:

```
cd c: xamppphp
```

Una vez encontrado el archivo php.ini se debe realizar lo siguiente:

Buscar la línea que contiene la directiva `"disable_functions"` y agregue `"phpinfo"` a la lista separada por comas de funciones deshabilitadas.

```
disable_functions = phpinfo
```

Guarde el archivo php.ini y reinicie su servidor web para que los cambios surtan efecto.

Reiniciar WIndows con XAMPP instlado:

1. Abre la consola de XAMPP desde el menú de inicio o buscando la aplicación en la carpeta de instalación.
2. En la consola de XAMPP, detén los servicios de Apache y MySQL haciendo clic en el botón "Stop" para cada uno de ellos. Asegúrate de que ambos servicios estén detenidos antes de continuar.
3. Una vez que se hayan detenido los servicios, cierra la consola de XAMPP.
4. Abre el menú Inicio de Windows y haz clic en el botón de "Apagar" para mostrar las opciones de apagado.
5. Selecciona "Reiniciar" para reiniciar el servidor.
6. Espera a que el servidor se reinicie completamente y vuelve a abrir la consola de XAMPP.
7. En la consola de XAMPP, inicia los servicios de Apache y MySQL haciendo clic en el botón "Start" para cada uno de ellos. Asegúrate de que ambos servicios estén iniciados antes de continuar.
8. Verifica que tus sitios web o aplicaciones estén funcionando correctamente.

Cross-Site Scripting XSS

XSS Cross-Site Scripting (o "Inyección de scripts entre sitios", en español). Es un tipo de vulnerabilidad de seguridad en aplicaciones web, donde un atacante puede insertar código malicioso (como JavaScript) en una página web, que luego se ejecutará en el navegador de un usuario que visite esa página.

Los ataques XSS ocurren cuando una aplicación web no valida correctamente las entradas de los usuarios, permitiendo que un atacante inserte código malicioso en una página web que se servirá a otros usuarios. El código malicioso puede ser diseñado para robar información personal, redirigir a los usuarios a sitios web maliciosos, mostrar anuncios no deseados, o incluso para tomar control del navegador del usuario.

Existen dos tipos principales de ataques XSS:

Reflejado: El ataque XSS reflejado ocurre cuando el código malicioso se ejecuta en la página web después de que un usuario hace una solicitud a la aplicación web. El código malicioso se "refleja" de vuelta al usuario a través de la respuesta de la aplicación web.

Almacenado: El ataque XSS almacenado ocurre cuando el código malicioso se almacena en la base de datos de la aplicación web y se ejecuta cada vez que se solicita la página web afectada.

Los desarrolladores pueden prevenir ataques XSS mediante la validación y filtrado de las entradas de los usuarios, utilizando bibliotecas de seguridad como Content Security Policy (CSP) y asegurándose de que todas las entradas de los usuarios se escapen de forma adecuada antes de ser utilizadas en una página web.

Sanitización de XSS en PHP

Aquí hay un ejemplo de cómo mitigar el riesgo de XSS en PHP utilizando la función `"htmlspecialchars()"` para escapar las salidas de usuario:

```
<?php
$param = "<a href='test'>Test</a>";
$valid = htmlspecialchars($param, ENT_QUOTES, 'UTF-8');
```

```
echo $valid // <a href='test'>Test</a>
?>
```

En este ejemplo, estamos utilizando la función `"htmlspecialchars()"` para escapar los caracteres especiales HTML en la entrada del usuario. La función toma tres argumentos: la cadena de entrada a escapar, la opción `ENT_QUOTES` para escapar tanto comillas dobles como comillas simples, y la codificación de caracteres 'UTF-8'.

Sanitización XSS en Laravel

Este método fue probado para Laravel 7, y consiste en crear un middleware para sanitizar las entradas.

El middleware proporciona un mecanismo conveniente para inspeccionar y filtrar las solicitudes HTTP que ingresan a su aplicación.

Para crear un middleware se debe aplicar el siguiente comando en la raíz del proyecto de Laravel:

```
php artisan make:middleware XssSanitizer
```

Editar el archivo app/Http/Middleware/XssSanitizer.php para que quede de la siguiente manera:

```
<?php
namespace App\Http\Middleware;
use Closure;
use Illuminate\Http\Request;

class XssSanitizer
{
    /**
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @return mixed
     */
    public function handle(Request $request, Closure $next)
    {
```

```
        $input = $request->all();
        array_walk_recursive($input, function(&$input) {
            $input = strip_tags($input);
        });
        $request->merge($input);
        return $next($request);
    }
}
```

Ahora es necesario agregar la ruta de XssSanitizer.php al vector $routeMiddleware ubicado en app/Http/Kernel.php:

```
protected $routeMiddleware = [

'auth' => \App\Http\Middleware\Authenticate::class,

....

'XssSanitizer' => \App\Http\Middleware\XssSanitizer::class,
];
```

Una vez realizado esto, ya se puede utilizar XssSanitizer middleware en las rutas para realizar la sanitización:

```
Route::group(['middleware' => ['XssSanitizer']], function () {

Route::get('/', function () {
        return view('welcome');
    });
    Route::get('/formulario',function () {
        return view('formulario');
    });

    Route::get('form-get', function (Illuminate\Http\Request $request)
    {
        return  $request->input('buscar');
    })->name('form-get');
```

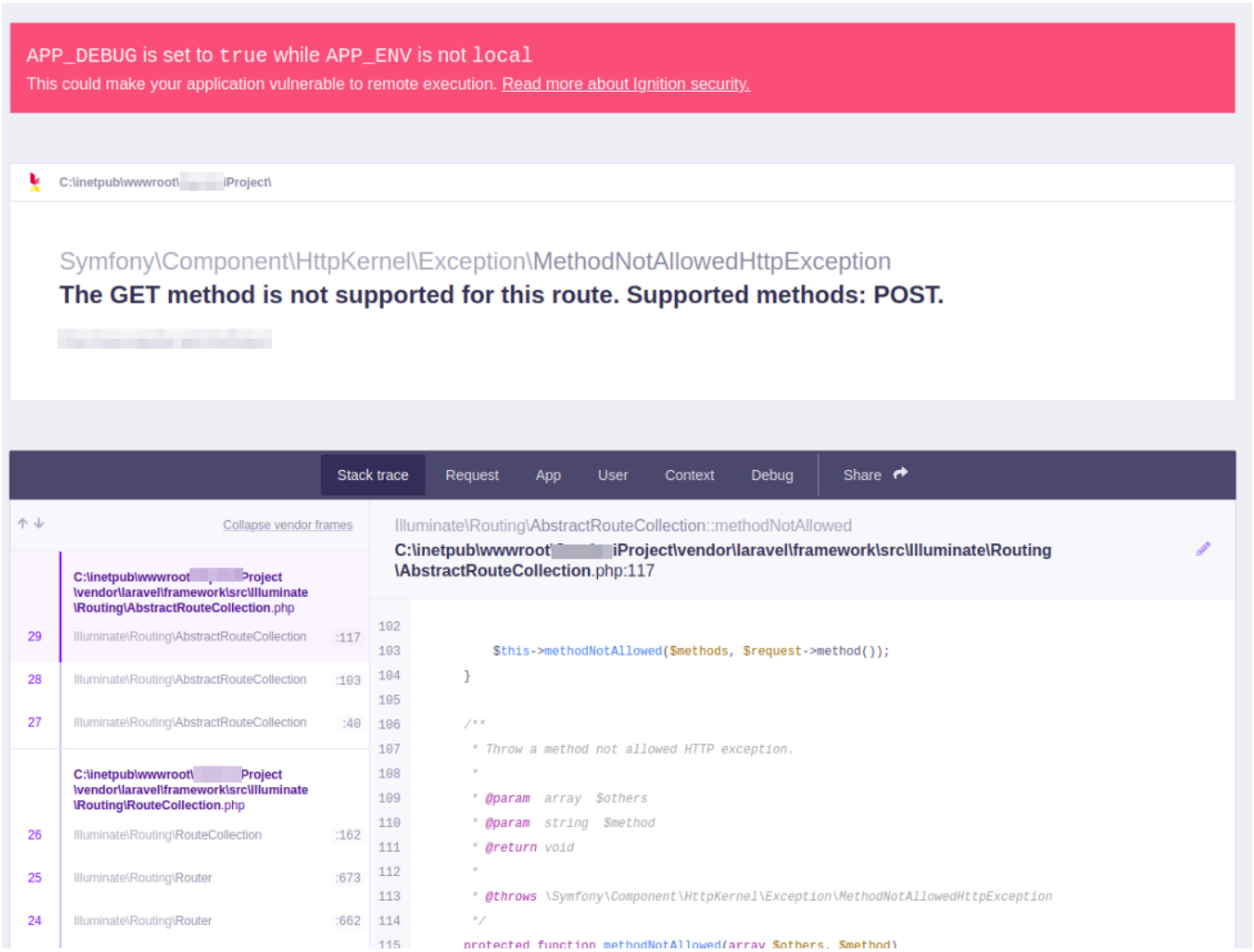
```
});
```

Aplicaciones web

Despliegue de mensajes de error

El software genera un mensaje de error que incluye información confidencial sobre su entorno, usuarios o datos asociados.

La información confidencial puede ser información valiosa por sí misma (como una contraseña, nombres de usuario), o puede ser útil para lanzar ataques dirigidos.



Un atacante puede usar el contenido de los mensajes de error para ayudar a lanzar otro ataque más enfocado.

Deshabilitar el despliegue de errores en PHP

El manual de PHP recomienda deshabilitar "display_errors" en servidores expuestos a Internet.

Para PHP 5.2.4 y superior, la configuración `"display_errors"` en el archivo de configuración `"php.ini"` debe establecerse en `"stderr"` (flujo de salida de error), en lugar de "`stdout`" (flujo de salida enviado a los clientes).

```
display_errors = stderr
```

Para versiones anteriores, `"display_errors"` es un tipo booleano y se puede establecer en `"False"` para desactivarlo. La configuración también se puede deshabilitar en tiempo de ejecución usando `ini_set()` desde dentro de un script PHP.

```
display_error = False
```

Deshabilitar el despliegue de errores en Framework Yii 2.0

Yii incluye embebido un Error Exception Handler que hace del manejo de errores una experiencia mucho más llevadera.

El manejo de excepciones está habilitado por defecto. Se puede deshabilitar definiendo la constante global `"YII_ENABLE_ERROR_HANDLER"` a “`"false"` en el script de entrada (Entry Scripts) de la aplicación.

```
YII_ENABLE_ERROR_HANDLER = false
```

Variables de entorno en PHP

Una variable de entorno es un valor dinámico que se almacena en el sistema operativo y que puede ser utilizado por diferentes aplicaciones y procesos en un sistema informático. Estas variables contienen información que puede ser utilizada por programas y scripts para personalizar su comportamiento y configuración.

Las variables de entorno se establecen y se gestionan a nivel del sistema operativo, y están disponibles para cualquier programa que se ejecute en el sistema. Estas variables pueden contener información como la ubicación de ciertos archivos o directorios, el nombre del usuario actual, la configuración regional o cualquier otra información que se considere útil para el funcionamiento de las aplicaciones.

Las variables de entorno son útiles para personalizar y automatizar el comportamiento de los programas y scripts, y también pueden ser utilizadas para compartir información entre diferentes aplicaciones y procesos en el sistema.

Variables de entorno en Apache

Opción a)

Puede agregar la siguiente instrucción desde el archivo de configuración que se encuentra en `/etc/apache2/sites-available`

```
<Files archivo.extension>
order allow,deny
deny from all
</Files>
```

Opción b)

Puede agregar la siguiente instrucción desde archivos .htaccess:

```
<FilesMatch
"\.(engine|inc|install|make|module|profile|po|sh|.*sql|theme|twig|tpl(\.php)?|xtmpl|yml)(~|\.sw[o
```

```
p]|\.bak|\.orig|\.save)?$|^(\.(?!well-
known).*|Entries.*|Repository|Root|Tag|Template|composer\.(json|lock)|web\.config)$|^#.*#$|\.php(
~|\.sw[op]|\.bak|\.orig|\.save)$">
  <IfModule mod_authz_core.c>
    Require all denied
  </IfModule>
  <IfModule !mod_authz_core.c>
    Order allow,deny
  </IfModule>
</FilesMatch>
```

Sistemas de administración de cotenido (CMS)

Capítulo destinado a la solución/mitigación de vulnerabilidades en sistemas de administración de contenidos: wordpress, joomla, drupla y otros.

Exposición del archivo XMLRPC

En su forma más simple, XML-RPC (llamada a procedimiento remoto) se creó para la comunicación entre plataformas. Este protocolo solía realizar llamadas a procedimientos utilizando HTTP como transporte y XML como codificador. El cliente realiza estas llamadas enviando una solicitud HTTP al servidor y recibe la respuesta HTTP a cambio. XML-RPC invoca funciones a través de una solicitud HTTP y luego estas funciones realizan algunas acciones y envían respuestas codificadas a cambio.

Con el archivo xmlrpc.php habilitado, un actor malintencionado, puede aprovechar las llamadas a procedimeitos remotos (RPC) e invocan funciones para obtener los datos que desean. En la mayoría de los sitios de WordPress, el xmlrpc.php es facilmente rastreable, y con solo enviar datos XML arbitrarios, pueden lograr a controlar el sitio para ejecutar código que han preparado para ejecutar un determinado tipo de ataque.

Solución a la exposición de xmlrpc.php

Puede hacer esto simplemente agregando el bloque de código dentro de su .htaccess. Asegúrese de hacer esto antes de las .htaccess rules que nunca cambian agregadas por WordPress.

```
<Files xmlrpc.php>
order allow,deny
deny from all
</Files>
```

Esto deshabilitará el xmlrpc.php archivo para cada aplicación o servicio que lo use. Puede incluir en la lista blanca una determinada dirección IP en caso de que aún desee acceder a su sitio de WordPress a través de XMLRPC. Para eso, necesitas agregar el siguiente comando:

```
<Files xmlrpc.php>
<RequireAny>
Require ip 1.1.1.2
Require ip 2001:db8::/32
</RequireAny>
```

```
</Files>
```

Servidores web

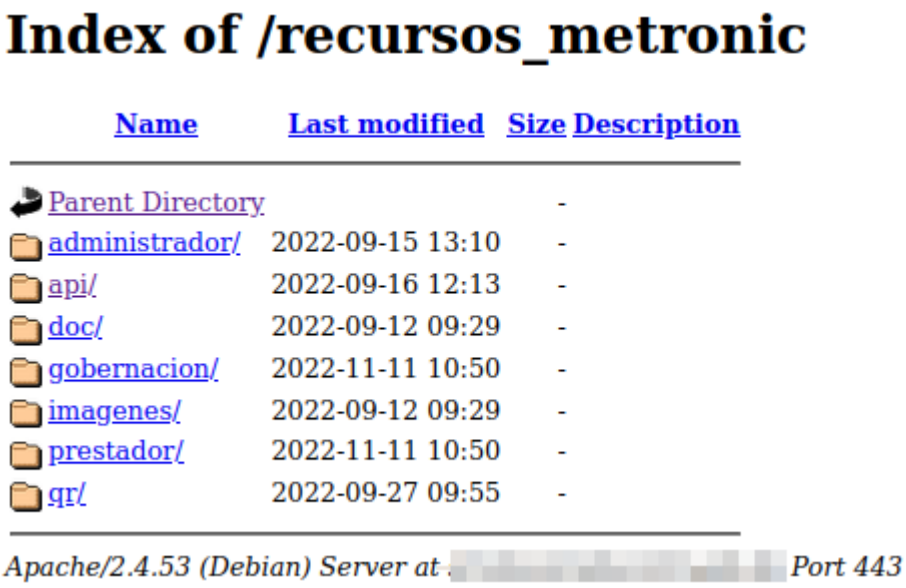
Capítulo destinado a la solución/mitigación de vulnerabilidades en servidores web.

Servidores web

Listado de directorio (Index Of)

Una lista de directorios se expone de forma inapropiada, lo que proporciona información potencialmente confidencial a los atacantes.

Una lista de directorio proporciona a un atacante el índice completo de todos los recursos ubicados dentro del directorio. Los riesgos y consecuencias específicos varían según los archivos enumerados y accesibles.



Deshabilitar Index Of en Apache

Las siguientes configuraciones fueron realizadas en un servidor Debian 9, Apache 2.4 y usuario con privilegios sudo. Dependiendo del caso puede elegir una de las siguientes opciones para deshabilitar el listado de directorio.

DEBIAN

1. Deshabilitar el módulo autoindex

Es el método más efectivo, pero se tiene que hacer un análisis previo para aplicarlo, ya que deshabilita esta funcionalidad a nivel global, es decir, si se quiere compartir archivos por HTTP (No recomendado) ya no se podría hacer, y los cambios afectarían a todas las aplicaciones bajo el servidor.

```
$ sudo a2dismod autoindex
```

Después de ejecutar se mostrará el mensaje de advertencia al cual responder con la siguiente frase:

```
To continue type in the phrase 'Yes, do as I say!' or retry by passing '-f': Yes, do as I say!
```

Reiniciar el servidor

Reiniciar el servidor

```
$ sudo systemctl restart apache2.service
```

2. Deshabilitar por archivo de configuración del sitio

Este método es el recomendable, ya que permite deshabilitar esta funcionalidad solo para el sitio en cuestión. Por ejemplo si se tiene el sitio www.sitio-de-prueba.com con el archivo de configuración (virtualhost) `sitio-de-prueba.conf`. Agregar en el archivo la siguiente directiva:

```
<VirtualHost *:80>

......

<Directory /var/www/sitio-de-prueba>
    AllowOverride All
    Options -Indexes
</Directory>

.......

</VirtualHost>
```

Guardar y recargar la configuración.

```
$ sudo systemctl reload apache2.service
```

3. Deshabilitar a través del archivo .htaccess

Es una alternativa similar al archivo de configuración y se tiene modificar o crear dependiendo del caso el archivo .htaccess y agregar:

```
Options -Indexes
```

Guardar el archivo y reiniciar el servidor.

```
$ sudo systemctl restart apache2.service
```

El resultado de aplicar una de las configuraciones anteriores, será la restricción de listar archivos y carpetas.

RHE/CENTOS

1. Deshabilitar por archivo de configuración

Este método es el recomendable, ya que permite deshabilitar esta funcionalidad solo para el sitio en cuestión. Por ejemplo si se tiene el sitio www.sitio-de-prueba.com su correspondiente configuración está en el archivo de configuración `httpd.conf` ubicado en `/etc/httpd/conf/httpd.conf` Agregar en la configuración del virtualhost correspondiente al sitio la siguiente directiva:

```
<VirtualHost *:80>

......

<Directory /var/www/html/sitio-de-prueba>
    AllowOverride All
    Options -Indexes
</Directory>

.......

</VirtualHost>
```

Guardar y reiniciar el servidor.

```
$ sudo /etc/init.d/httpd restart
```

3. Deshabilitar a través del archivo .htaccess

Para esta opción se tiene que habilitar el módulo `rewrite`. Es una alternativa similar al archivo de configuración y se tiene modificar o crear dependiendo del caso el archivo .htaccess y agregar:

```
Options -Indexes
```

Guardar el archivo y probar la configuración.

CPANEL

En ocasiones las páginas web de las entidades están publicadas en un servicio de web hosting al acceden desde el cpanel, y la configuración por defecto mantiene habilitado los índices en los directorios del cpanel y por ende en la capeta `public_html`.

1. Ingresar a cpanel, buscar y seleccionar la opción `Índices`. file 2. Seleccionar la carpeta `public_html`. file 3. Seleccionar la opción `Sin índice`. file

Aplicar la misma configuración en el resto de carpetas del sitio, ya que desde cpanel no es posible aplicar una configuración global que deshabilite el index of.

Tambien se puede aplicar las siguientes opciones:

A. Desde el cPanel

1. Accedemos al panel de control (cPanel) y buscamos el menú de opciones Herramientas Avanzadas.
2. Entramos en la sección Index Manager donde se nos mostrará un árbol con todos los directorios de nuestra web.
3. Navegamos hasta el directorio que deseemos proteger. Para entrar en los directorios hay que hacer click en el icono que está a la izquierda del nombre de la carpeta. Cuando encontremos la carpeta que deseamos proteger hacemos click en el nombre de la misma donde podemos marcar la opción No Indexar.

B. Desde el archivo .htaccess

1. Introduzca la siguiente línea de código `Options -Indexes`
2. También se puede evitar que se listen unos determinados archivos en concreto según su extensión. Por ejemplo, para evitar que se listen archivos de extensión .php y .html escriba: `IndexIgnore *.php *.html`
3. Este método sirve para que se listen los directorios pero haciendo que estos aparezcan en blanco. De esta forma cualquier tipo de fichero no se mostrará a la hora de acceder a su correspondiente carpeta: `IndexIngnore *`

C. Desde un archivo index en blanco

Esta solución es sencilla, solo se debe crear un documento en blanco llamado index.html (o index.php) para evitar el listado de los archivos cuando se acceda al directorio. El problema de este método es que se debe añadir un archivo index en blanco en cada uno de los directorios en los que no se quiere mostrar los archivos.

TOMCAT

https://support.esri.com/en/technical-article/000010708

Certificados SSL/TLS

Let's encrypt es un servicio que ofrece certificados SSL gratuitos a través de una API. Cerbot es un cliente ACME de Let's Encrypt, que tiene varias formas de validar el dominio, busca certificados y configura automáticamente Apache y Nginx.

Certificados Let's Encrypt

Instalación de Cerbot en Debian

Instalación de Cerbot:

```
sudo apt-get update
sudo apt-get install software-properties-common
sudo add-apt repository universe
sudo add-apt repository ppa:cerbot/cerbot
sudo apt-get update
sudo apt-get install cerbot
```

O use aptitude otro administrador de paquetes para su distribución.

Versiones de Ubuntu por debajo de 16.10 yakkety

```
sudo add-apt-repository ppa:certbot/certbot
sudo apt-get update
sudo apt-get install --upgrade letsencrypt
```

Otra distribución

Si tiene alguna otra distribución, hay instrucciones de instalación adicionales en el sitio web oficial de Cerbot. Si prescinde de un administrador de paquetes, generalmente la instalación se reduce a:

```
wget -O /usr/local/bin/certbot-auto https://dl.eff.org/certbot-auto
chmod +x /usr/local/bin/certbot-auto
```

```
ln -s /usr/local/bin/certbot-auto /usr/local/bin/certbot
```

Crear un certificado SSL

Let's Encrypt realiza automáticamente la validción de dominio (DV). La autoridad de certificación (CA) verifica la autenticidad del dominio del servidor. Una vez que haya sido validado, la CA le emitirá certificados SSL.

Ejecutar Let's Encrypt con el siguiente comando:

```
sudo -H ./letsencrypt-auto certonly --standalone -d tudominio.com -d www.tudominio.com
```

Eliminar la visualización de archivos de configuración

Servidores Apache

Servidores Nginx

Cross-Site Request Forgery CSRF

La vulnerabilidad CSRF (Cross-Site Request Forgery) es un tipo de ataque que se produce cuando un usuario autenticado involuntariamente envía una solicitud malintencionada a un sitio web. Este ataque aprovecha la confianza del sitio web en la sesión activa del usuario para enviar una solicitud no autorizada, que puede provocar cambios no deseados en el estado de la cuenta del usuario, como transferencias de dinero, cambios de contraseña, eliminación de datos, entre otros.

Solución

Implementar tokens de seguridad CSRF en formularios HTML que se verifican en el servidor para garantizar que la solicitud sea legítima y no se haya falsificado.

Para conocer más a cerca de la vulnerabilidad y formas de mitigación consultar Cross-Site Request Forgery Prevention Cheat Sheet CSRF

Copias de seguridad

Una copia de seguridad, también conocida como respaldo o backup, es una copia de los datos y archivos importantes de un sistema, como documentos, bases de datos, configuraciones y otros elementos críticos. El propósito de una copia de seguridad es permitir la recuperación de los datos en caso de pérdida, daño o corrupción de los originales.

El almacenamiento de las copias de seguridad es una consideración crucial para garantizar su efectividad. Algunas opciones comunes para almacenar copias de seguridad son:

1. Dispositivos de almacenamiento local: Puedes guardar las copias de seguridad en dispositivos de almacenamiento físico conectados directamente a tu sistema, como discos duros externos, NAS (Network Attached Storage) u otros medios extraibles.
2. Servidores de respaldo en la red local: Si se cuenta con una red local, se puede configurar un servidor dedicado donde almacenar las copias de seguridad de varios sistemas.
3. Servidores FTP o SFTP remotos: Puedes alamacenar copias de seguridad en servidores remotos utilizando protocolos de tranasferencia de archivos como FTP (File Transfer Protocol) o SFTP (Secure File Transfer Protocol).

Copia de seguridad en Nginx

Para realizar copias de seguridad en Ngnix se pueden utilizar los siguientes comandos:

1. Utilizar el comando cp para copiar archivos y directorios, por ejemplo:

```
cp -r /ruta/directorio_origen /ruta/directorio_destino

En Nginx los archivos de configuración se encuentran en la ruta:

/etc/nginx/

Los archivos del DocumentRoot en la ruta:

/var/www/ o /usr/share/nginx/html/
```

2. Utilizar el comando tar para crear archivos de respaldo comprimidos, por ejemplo:

```
tar -czvf archivo.tar.gz /ruta/directorio
```

Copia de seguridad Apache

Para realizar copias de seguridad en apache se pueden utilizar los siguientes comandos:

1. Utilizar el comando cp para copiar archivos y directorios, por ejemplo:

```
cp -r /ruta/directorio_origen /ruta/directorio_destino

En Apache los archivos de configuración se encuentran en la ruta:

/etc/apache2/

Los archivos del DocumentRoot en la ruta:

/var/www/html/
```

2. Utilizar el comando tar para crear archivos de respaldo comprimidos, por ejemplo:

```
tar -czvf archivo.tar.gz /ruta/directorio
```

Es importante mencionar que las rutas especificas pueden variar según la configuración del sistema.

Contenido por defecto

El contenido por defecto en servidores se refiere al conjunto de archivos y directorios que se proporcionan automáticamente al instalar un servidor web en un sistema operativo. Estos archivos y directorios son parte de la configuración predeterminada y su propósito es ofrecer una página o sitio web básico cuando se accede al servidor sin haber configurado o desplegado ningún sitio específico.

En muchos casos, es deseable eliminar o modificar el contenido por defecto para evitar la exposición de información innecesaria o para personalizar la experiencia del usuario al acceder al servidor. Esto se puede lograr eliminando o reemplazando los archivos y directorios por defecto con el contenido adecuado para el sitio web.

Antes de remover los directorios y archivos de instalación por defecto se recomienda realizar una copia de seguridad de los mismos.

Remover directorios y archivos de instalación (manuales) en Apache2

Apache generalmente incluye un directorio de documentos por defecto, `/var/www/html/` en sistemas basados en Debian/Ubuntu.

Puedes eliminar los archivos y directorios por defecto en esa ubicación par evitar que sean accesibles. Por ejemplo, en linux puedes usar el siguiente comando para eliminar el contenido por defecto:

```
sudo rm -r /var/www/html/*
```

Finalmente reiniciar el servidor Apache.

Remover directorios y archivos de instalación (manuales) en Tomcat

Los archivos de instalación por defecto en tomcat se encuentran en `/opt/tomcat/` o en alguna otra ubicación del sistema.

Dentro del directorio de Tomcat, busca el directorio `/webapps/`. Este directorio contiene las aplicaciones web desplegadas en Tomcat.

Eliminar el directorio `/docs/`, este directorio contiene la documentación y los manuales de Tomcat, puede utilizar el siguiente comando en linux para eliminar el directorio:

```
sudo rm -r /opt/tomcat/webapps/docs
```

Finalmente reiniciar el servidor Tomcat.

Implementación de cabeceras de seguridad

Las cabeceras de seguridad son una forma de agregar capas adicionales de seguridad a los sitios web y aplicaciones web. Son metadatos enviados por el servidor web junto con las respuestas a las solicitudes del cliente, y que proporcionan información adicional sobre cómo se debe manejar la respuesta y cómo se debe proteger la sesión del usuario.

Las cabeceras de seguridad se utilizan para implementar políticas de seguridad adicionales en la comunicación entre el cliente y el servidor web, y se pueden utilizar para mitigar una variedad de vulnerabilidades de seguridad, como ataques de inyección de código, cross-site scripting (XSS), clickjacking, sniffing de paquetes y otros.

Entre las cabeceras de seguridad más comunes se incluyen:

- Content-Security-Policy (CSP): Esta cabecera permite a los sitios web especificar qué tipos de contenido (como scripts, fuentes, imágenes, etc.) se pueden cargar y desde qué orígenes se pueden cargar. Esto puede prevenir los ataques XSS, ya que limita el alcance de los scripts maliciosos.
- Strict-Transport-Security (HSTS): Esta cabecera obliga a los navegadores web a utilizar HTTPS para todas las solicitudes al servidor. Esto protege contra ataques de sniffing de paquetes y garantiza que la comunicación entre el cliente y el servidor se realice de forma segura.
- X-Frame-Options: Esta cabecera permite a los sitios web controlar si su contenido se puede cargar en un iframe de otro dominio, lo que ayuda a prevenir ataques de clickjacking.
- X-XSS-Protection: Esta cabecera activa el filtro anti-XSS del navegador y previene ataques XSS.
- X-Content-Type-Options: Esta cabecera ayuda a prevenir ataques de sniffing de contenido, especificando el tipo de contenido que el servidor web debe enviar.

Implementación de cabeceras de seguridad en servidores Apache

Implementación realizada en apache 2.4 y Debian se requieren que el usuario tenga permisos de administrador (sudo):

Habilitar las cabeceras de seguridad:

```
# a2enmod headers
```

Configurar del archivo security.conf:

```
# nano /etc/apache2/conf-enabled/security.conf
```

Modificar los siguientes valores:

```
Header set X-Content-Type-Options: "nosniff"

Header always set X-Frame-Options: "SAMEORIGIN"

Header always set Referrer-Policy: "no-referrer"

Header set X-XSS-Protection: "1; mode=block"
```

Finalmente reiniciar el servidor:

```
systemctl restart apache2
```

Implementación de cabeceras de seguridad en servidores Nginx

Implementación realizada en Nginx y Debian

Ir al directorio `/etc/nginx/conf.d/` e ingresar al archivo de configuración `.conf` que este utilizando.

Abrir el archivo de configuración y agregar las siguientes líneas al bloque `server`:

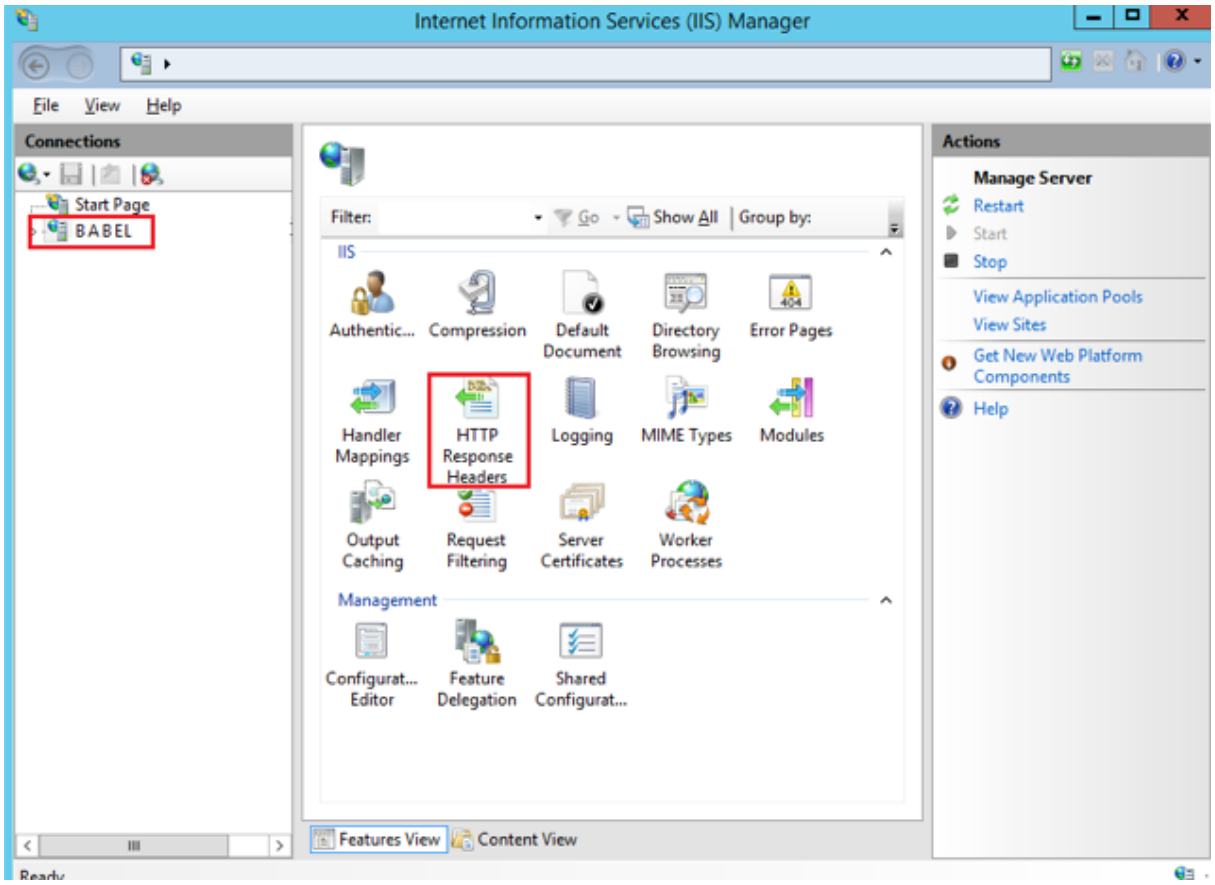
```
# Configuración de cabeceras de seguridad
add_header X-XSS-Protection "1; mode=block";
add_header X-Content-Type-Options "nosniff";
add_header X-Frame-Options "SAMEORIGIN";
add_header Referrer-Policy "strict-origin-when-cross-origin";
add_header Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-inline'; style-
src 'self' 'unsafe-inline'";
```

Reiniciar el servidor nginx:

```
sudo systemctl restart nginx
```

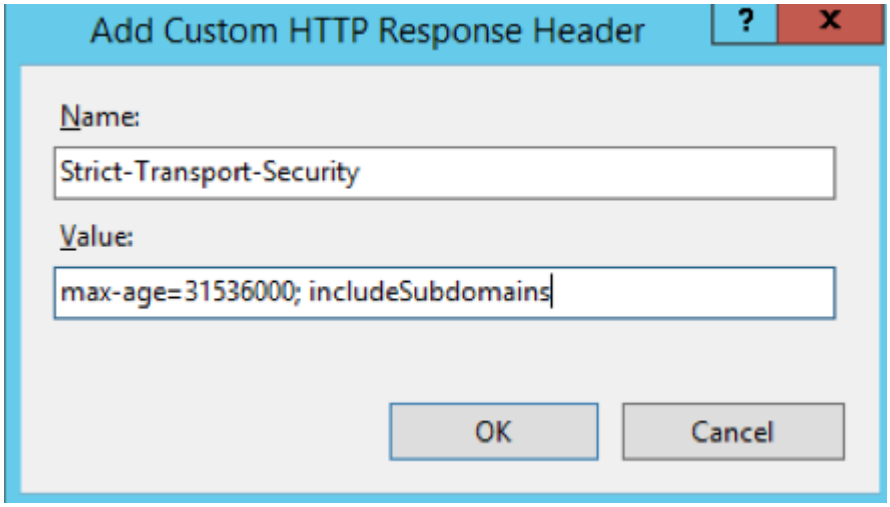
Implementación de cabeceras de seguridad en servidores IIS

Configuración en el Servidor IIS, para su configuración, en la ventana encabezados de respuesta HTTP, haga clic en agregar en el panel derecho de acciones y luego ingrese los detalles del encabezado como se muestra a continuación.

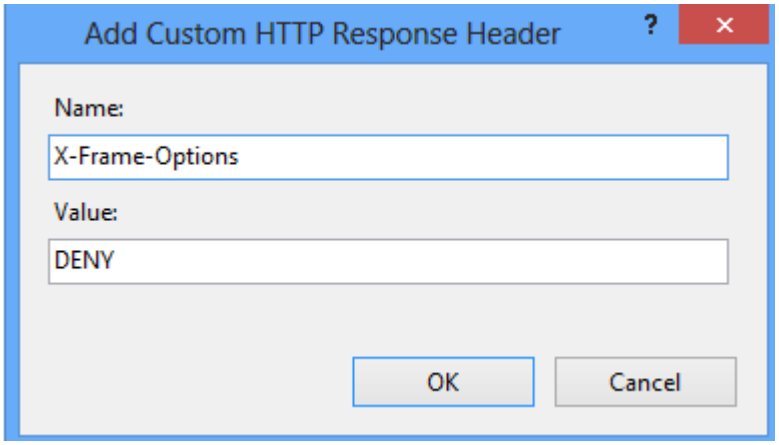


Strict-Transport-Security

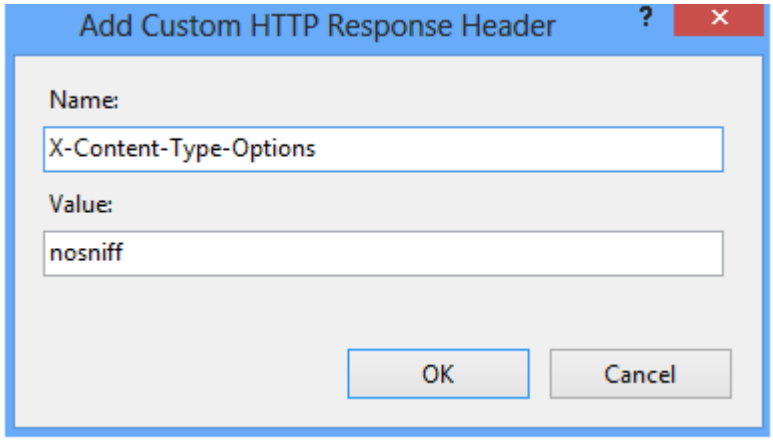
El valor “max-age=63072000” es el número de segundos que se establece para que la navegación haga uso del encabezado.



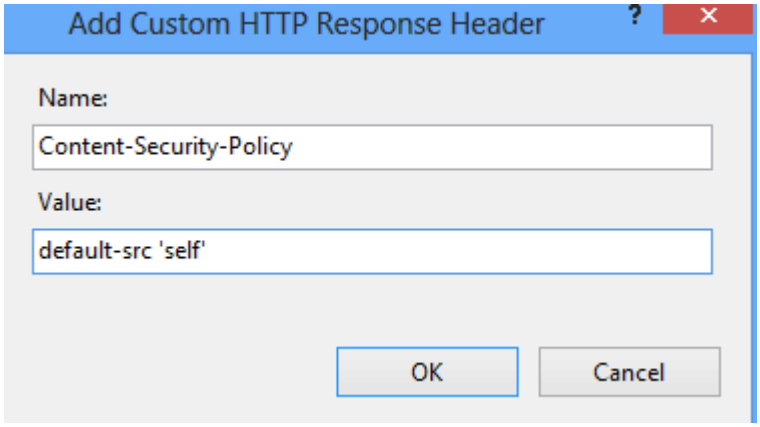
X-Frame-Options



X-Content-Type-Options



Content-Security-Policiy


Add Custom HTTP Response Header
Name:
Content-Security-Policy
Value:
default-src 'self'
OK
Cancel

Quitar archivos de configuración

La exposición de archivos de configuración se refiere al riesgo de que los archivos de configuración de un servidor o una aplicación sean accesibles públicamente a través de Internet. Estos archivos contienen información sensible, como contraseñas, claves de API u otros datos confidenciales que podrían ser utilizados por un atacante para comprometer la seguridad del sistema.

La exposición de archivos de configuración puede ocurrir debido a una configuración incorrecta del servidor web o a la falta de medidas de seguridad adecuadas. Algunos ejemplos de archivos de configuración comunes que deben mantenerse privados son los archivos .env, .config, web.config, php.ini, entre otros.

El impacto de la exposición de archivos de configuración puede ser significativo, ya que permite a los atacantes obtener información sensible y utilizarla para realizar acciones maliciosas, como el acceso no autorizado a sistemas, la modificación de la configuración, el robo de datos o la ejecución de ataques más avanzados.

Quitar archivos de configuración en Apache

1. Abre el archivo de configuración principal de Apache, generalmente ubicado en `/etc/httpd/httpd.conf` o `/etc/apache2/apache2.conf.`

2. Dentro del bloque Directory, puedes establecer reglas de acceso y restricciones para los archivos y directorios específicos.

```
<FilesMatch "^(.*\.pl|.*\.env|.*\.config|.*\.yml)$">
    Require all denied
</FilesMatch>
```

Quitar archivos de configuración en Nginx

1. Abre el archivo de configuración principal de Nginx, generalmente ubicado en `/etc/nginx/nginx.conf`.

2. Dentro del bloque http, puedes agregar una directiva para denegar el acceso a los archivos específicos que deseas proteger.

```
location ~* /\.(pl|env|config|yml)$ {
    deny all;
}
```