Securing HCL ZIETrans Web Application (Using HTTP Header)

What are HTTP Security Headers?

HTTP Security Headers are a collection of key-value pairs included in the headers of a HTTP response to strengthen the security of web applications and websites. These are a specialized subset of HTTP headers designed to address security concerns. By dictating how browsers and clients handle web pages and resources, they safeguard against various security threats, such as cross-site scripting (XSS), clickjacking, and protocol downgrades.

Why do we need HTTP Security Headers?

HTTP security headers are essential for safeguarding sensitive data, such as user login credentials, and preventing unauthorized access, data theft, or manipulation of web pages. They mitigate some of the most common cyber threats, including:

- Hacker attacks
- Malware injections
- Clickjacking
- Malicious script injections

These headers add an extra layer of protection by restricting interactions between the server and browser during a web application's operation. They also assist in meeting regulatory compliance requirements and adhering to industry best practices.

Important HTTP Security Headers to implement

To enhance the security of your web applications, consider implementing the following crucial HTTP security headers:

- X-XSS-Protection
- Content-Security-Policy
- X-Content-Type-Options
- Strict-Transport-Security

For other security-related considerations:

- Other Security Considerations
- Enabling CSRF and XSS protections in ZIETrans

HTTP Security Headers in ZIETrans

HCL ZIETrans provides provisions to protect against 'X-XSS-Protection, X-Content-Type-Options, and Content-Security-Policy.' In the ZIETrans **web.xml** file, update the *'param-value'* under the filter *'HatsHeaderSecurityFilter'* section from **"NO"** to **"YES"**. Additionally, you can add URLs that should be secured using these features.

```
<filter-mapping>
    <filter-name>HatsHeaderSecurityFilter</filter-name>
    <url-pattern>
    </url-pattern>
    <url-pattern>/</url-pattern>
    <url-pattern>/entry</url-pattern>
</filter-mapping>
```

- **X-XSS-Protection**

The X-XSS-Protection header is a response header designed to enable or configure built-in anti-XSS (Cross-Site Scripting) protection in web browsers like Internet Explorer, Chrome, and Safari. It helps mitigate XSS attacks by controlling how browsers respond to potential XSS threats.

Common values for X-XSS-Protection

- **" X-XSS-Protection: 1; mode=block"**
  Enables XSS protection and blocks the page if a potential XSS attack is detected.

- **“X-XSS-Protection 0"**
  Disables XSS protection.
- **“X-XSS-Protection: 1; report=<reporting-uri>"**
  Enables XSS protection and will send a report if a potential XSS attack is detected. The <reporting-uri> value should be replaced with the URI that the report should be sent to.

To enable X-XSS-Protection in the ZIETrans project, change the param-value from ‘**NO**’ to ‘**YES’** in the **web.xml** file. The default X-XSS-Protection value will be ‘**1; mode=block**’.

```
<init-param>
    <param-name>X-XSS-Protection</param-name>
    <param-value>NO</param-value>
    <param-value>YES</param-value>
</init-param>
```

For more information, refer to https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protection

NOTE: This header is deprecated, and support has been removed from most browsers. To overcome this challenge, ZIETrans provides a custom filter that sanitizes the malicious script. To enable this filter, add the filter below to web.xml.

```
<filter-name>XSS-Filter</filter-name>
<filter-class>
    com.ibm.hats.runtime.filters.HatsXSSFilter
</filter-class>
</filter>
<filter-mapping>
    <filter-name>XSS-Filter</filter-name>
      <url-pattern>/*</url-pattern>
</filter-mapping>
```

- **Content Security Policy (CSP)**

CSP is a security feature that helps to prevent **cross-site scripting (XSS)**, **clickjacking**, and other code injection attacks by giving website owners control over the resources a web page can load and execute. For example, a website owner can use CSP to specify that a web page can only load scripts from the website's server, not other third-party

servers.

To enable the ZIETrans project's CSP feature, change the param-value from '**NO**' to '**YES'** in the **web.xml** file**.** The default CSP value will be "**frame-ancestors 'self'**".

```
<init-param>
    <param-name>Content-Security-Policy</param-name>
    <param-value>NO</param-value>
    <param-value>YES</param-value>
</init-param>
```

The frame-ancestors directive in the Content-Security-Policy (CSP) HTTP response header specifies whether a browser is permitted to render a page within a <frame> or <iframe>. By restricting content embedding into other sites, this directive helps protect against clickjacking attacks. Its functionality is similar to that of the '**X-FRAME-OPTIONS'** header.

Common uses of the frame-ancestors directive in CSP:

- Content-Security-Policy: frame-ancestors **'none';**
  Prevents any domain from embedding the content in a frame. It's recommended unless there's a specific requirement for framing.

- Content-Security-Policy: frame-ancestors **'self'**
  Allows only the current site to embed its content in a frame.

- Content-Security-Policy: frame-ancestors **'self' *.somesite.com;**
  Permits the current site, all pages on somesite.com (regardless of protocol), and the specific page myfriend.site.com to embed the content, provided it uses HTTPS and the default port (443).

For more information, refer to https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP.

- **X-Content-Type-Options**

The X-Content-Type-Options is an HTTP response header designed to prevent browsers, such as Internet Explorer and Google Chrome, from "mime-sniffing" a response's content type to something different from what the server declares. When this header is set to 'nosniff', the browser strictly adheres to the declared content type, helping to protect against attacks like cross-site scripting (XSS).

To enable X-Content-Type-Options in a ZIETrans project, change the param-value from '**NO**' to '**YES**' in the **web.xml** file. The default value for this header is set to **'nosniff'**.

```
<init-param>
    <param-name>X-Content-Type-Options</param-name>
    <param-value>NO</param-value>
    <param-value>YES</param-value>
</init-param>
```

For more information, refer to https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Content-Type-Options

- **HTTP Strict Transport Security (Only for HTTPS-enabled sites)**

HTTP Strict Transport Security (HSTS) is a web security mechanism designed to protect websites from protocol downgrade attacks and cookie hijacking. It instructs a web server to only allow access via a secure HTTPS connection, blocking insecure HTTP access. This protects against man-in-the-middle attacks, ensuring sensitive data is not intercepted by attackers. The HSTS policy is declared using the Strict-Transport-Security HTTP response header.

- **max-age=<expire-time>**: Specifies the duration (in seconds) the browser should remember to access the site using HTTPS only.

- **includeSubDomains *(Optional)***: This rule applies to all subdomains of the site if specified.
- **preload *(Optional, Non-standard)***: When this directive is used, the max-age value must be at least 31536000 (1 year), and the includeSubDomains directive must also be present. Note this is not part of the official specification.

To implement HSTS, application servers need to configure it either in the web.xml or server configuration file. For example, if you're using IBM Websphere, here's how you can enable HSTS using **web.xml:**

```
        <context-param>
                <description>Enable HTTP Strict Transport Security (HSTS) at
the web application level max-age is 1 year.
                Add includeSubDomains to apply this rule to all of the site's
subdomains as well</description>
           <param-name>com.ibm.ws.webcontainer.ADD_STS_HEADER_WEBAPP</param-n
ame>
           <param-value>max-age=31536000; includeSubDomains</param-value>
        </context-param>
```

Note: The browser ignores the Strict-Transport-Security header when your site has only been accessed using HTTP. Only when the site is accessed over HTTPS will the browser follow the Strict-Transport-Security header.

For more information, refer to https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security

Enable HTTP security headers using the Java filter class

To add new security headers or modify the default values of the headers mentioned above, you can create a filter class that intercepts HTTP requests before they access ZIETrans resources.

Note: If changes are done to existing headers, the new changes overwrite the default value set by **web.xml**.

```
public class SecureHeadersFilter implements Filter{

    private static final String HSTS = "Strict-Transport-Security";
    private static final String AGE = "max-age=31536000;";
    private static final String REFERRERPOLICY = "Referrer-Policy";
    private static final String REFERRERPOLICY_VALUE = "no-referrer";

    @Override
    public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
            throws IOException, ServletException {


        HttpServletResponse res = (HttpServletResponse) response;

        //Lets website tell the browser that they should be only accessed via HTTPS
        res.setHeader(HSTS, AGE);

        //control how much of the referrer information should be revealed
        res.setHeader(REFERRERPOLICY, REFERRERPOLICY_VALUE);

        chain.doFilter(request, response);
    }
```

In **web.xml,** add *'<filter>'*, *'<filter-mapping>'*, and *'<url-pattern>'* to the newly created filter. In *'url-pattern',* define application URLs on which the filter should be applied.

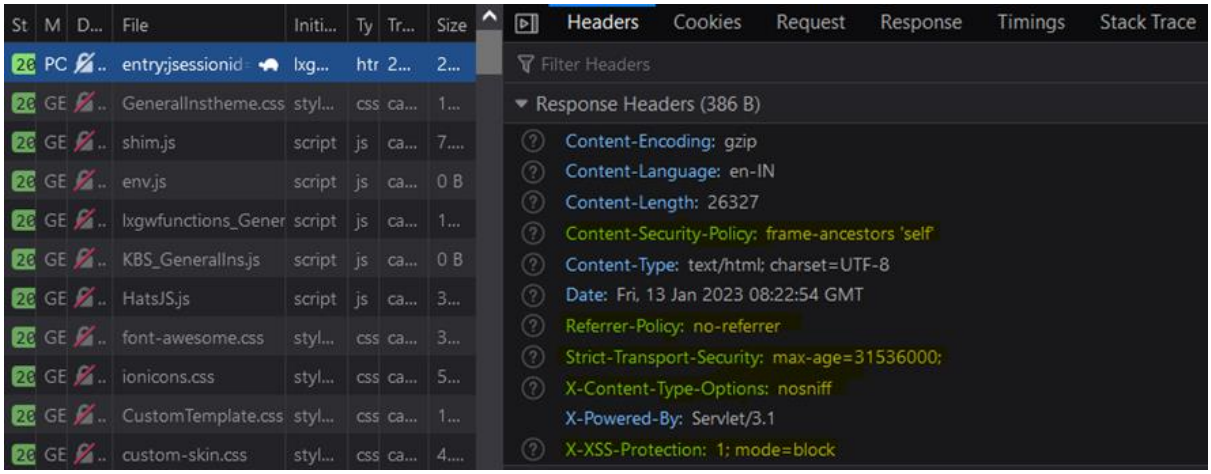
```
<filter>
        <description>This filter will invoke SecureHeadersFilter methods</description>
        <display-name>SecureHeadersFilter</display-name>
        <filter-name>SecureHeadersFilter</filter-name>
        <filter-class>filters.SecureHeadersFilter</filter-class>
</filter>
```

```
<filter-mapping>
    <filter-name>SecureHeadersFilter</filter-name>
    <url-pattern>/</url-pattern>
    <url-pattern>/entry</url-pattern>
</filter-mapping>
```

Final output

When you enable the security mentioned above in the ZIETrans application, you'll find the relevant values in the HTTP calls.

Here's a sample output captured from the browser network console.



Author Profile:

Supreeth Roy A

Software Engineer, Lab Services

Supreeth Roy has a strong background in designing and developing web applications. Currently involved in HCL ZIE products as part of the Mainframe Lab Services Team, leveraging technical skills to drive innovation and deliver high-quality solutions.