Security report

|  |  |
| --- | --- |
| Security breach | Covered? |
| Protection against malicious file uploads | Yes |
| Protection against Man-in-the-middle attacks | No |
| Protection against Link Injection Protection | Yes |
| Protection against Attribute autocomplete | Yes |
| Click hijacking protection | Yes |

# Protection against malicious file uploads

We restrict uploaded **file type** by only accepting “.940” file extension on frontend. Moreover, to instruct browsers to consider file types as defined and disallow content sniffing, we add X-Content-Type-Options header with parameter "nosniff" to our web page’s HTTP response. We also use blacklist to prevent **file names** from containing special characters: ' " - ; | # = . Before inserting **file content** into database, we check whether uploaded mt940 file can be parsed correctly. If it is, the parsing result will be inserted into database; otherwise, the file is malicious and rejected.

Protection against Man-in-the-middle attacks

All our communications are sent over HTTP because of server configuration; thus, it is unlikely for us to protect against man-in-the-middle attacks.

# Protection against Link Injection Protection

We do not display any sensitive user input in address bar. To prevent the attacker from manipulating the web page URL, we use servlet filter to ensure only the authenticated user can access all web pages. It will redirect the user to landing page if he/she has not logged in. Also, we add "X-XSS-Protection" header with value "1; mode=block" to tell the browser that it should not render the webpage in case that it detects an attack. We use blacklist on user input to prevent attackers from injecting special script or HTML brackets.

# Protection against Attribute autocomplete

We disable autocompletion for form fields by setting autocomplete=*"off"* in .jsp file. However, we cannot prevent auto-filling if the user enables ‘save password’ function in his/her browser configuration. We add Cache-Control header with value "no-cache, must-revalidate, max-age=0, no-store" to define the cache must revalidate the state of the resource before using it, and send the request even if the cache has stored the resource.

# Click hijacking protection

To ensure our site doesn’t get used in clickjacking attack, we add X-Frame-Options header with values "DENY", "SAMEORIGIN" and "ALLOW-FROM \*url\*" to tell browsers not to render a page in a <frame>, <iframe> or <object> tag.