Capture the Flag Risk Analysis

Risk ID (CWE ID)	Technical Risk	Technical Risk Indicators	Impact Rating	Impact	Mitigation	Validation Steps
95	Eval Inject	The eval function is used the plupload.silverlight.xap and	VH	Allows arbitrary code execution.	Escape all user input before it reaches the eval function.	Attempt to insert the code: echo(“Eval”); and check if the string Eval occurs on the page.
98	PHP Remote File Inclusion	Require statements in plugins.php, update.php and wp-settings.php do not check user-supplied input before used for required statement.	H	User could require the application to fetch and execute remote code.	Supply a whitelist of locations that required code can come from instead of a blacklist.	Attempt to load a php file with alert(“ not secure”) in it and see if a popup occurs when a page is loaded.
89	SQL Injection	SQL statements in board.php, and scoreboard/ index.php build SQL statements with user-supplied input without escaping special characters.	H	An attacker could execute arbitrary SQL statements allowing access to data and potentially adding incorrect or harmful data to the database.	Use parameterized prepared SQL statements and validate user-input to ensure that it is in the expected format.	Attempt to select * from the users table using a union statement at all paths that use parameterized sql statements and ensure nothing extraneous is printed.
259	Use of Hard-coded Passwords	board.php and scoreboard/ index.php both use hard coded passwords.	M	If the hardcoded password is discovered than all instances of the product are vulnerable to attack.	Store passwords in the environment or configuration files which are not remotely accessible.	Change the passwords and only update the passwords in the configuration file. Test all branches requiring a password as only those using the config passwords should work.
80	Basic XSS	User input displayed an executed without escaping special characters allowing users to execute arbitrary code on the webpage. This occurred on board.php and scoreboard/ index.php	M	Could allow an attacker to steal or manipulate cookies, modify content, or redirect the user to a malicious site.	Escape all user input and validate user input to ensure it fits the expected format.	Attempt to insert the code: <script>alert(“XSS”);</script> in all user inputs and parameters. If a popup occurs then it is not secure.
95	Broken or Risky Cryptographic Algorithm	Files such as bookmark.php use broken cryptographic methods.	M	Could allow the discloser of sensitive information.	Update to use cryptographic methods that are proven secure.	Use grep to ensure that all instances of the old cryptographic method are removed.
73	External Control of File Name or Path	User input is used to construct the filename in files such as plugin-editor.php	M	Could allow access to secure files on the server outside of the webroot.	Escape the user input and ensure that it is in the expected format.	Attempt to access “secure” files through the user input.
209	Information Exposure through an Error Message	On the board.php page more information is given then needed upon incorrect input.	L	It allows the attacker to gain information such as correct username such as to cut down the needed attack space.	Ensure that error messages are generic.	Read all error messages and ensure that the exact issue is not specified (i.e. any login issue should say username or password)