Risk ID	Technical Risk	Technical Risk Indicators	Related CVE, CWE, or OSVDB IDs	Impact Rating	Impact	Mitigation	Validation Steps
1	Remote code execution	Wordpress admin.php and update.php include remote source code based on unsanitized user input. Wordpress plupload module's Silverlight implementation executes user input as code.	CWE IDs 95, 98	H	An attacker can execute arbitrary code remotely.	Update Wordpress, it should validate user input before using it for a php include or eval. If updates do not remove instances of eval and include with user input, stop using Wordpress or rewrite those parts.	Server rejects user input for php include not in whitelist.
2	SQL Injection	Posting on the board (board.php), logging in (dblib.php), submitting flags (index.php), and various wordpress functionality makes SQL queries constructed with unvalidated user input.	CWE ID 89	H	An attacker can execute arbitrary SQL queries, exposing any information in the database, including account credentials and private information such as keys.	Validate user input before constructing a SQL query. Consider using user input to choose one of many predefined queries instead of constructing one dynamically with user input.	User input including symbols meaningful to SQL like quotes is not used in queries. E.g. login attempt with username 1' or '1 = 1 is either sanitized or rejected, not queried.
3	Hard-coded database credentials	Credentials used to access mysql are hard-coded in board.php, dblib.php, index.php, and site-new.php.	CWE ID 259	M	Credentials may be exposed more easily. An attacker who gains access to the source code will gain access to the database account.	Change account credentials and store them in a configuration file (out of source control).	Search through all source files for account credentials in plaintext.
4	Cross-site scripting	Posts on the board do not have script tags sanitized. Posting a script tag results in it being executed on any visitor's browser. The same problem exists in the wordpress admin console.	CWE ID 80	M	An attacker can execute arbitrary javascript on all visitors' browsers.	Sanitize user inputs. Replace < with &lt; and > with &gt; to prevent user input enclosed in <script></script> from being executed by the browser.	Server filters user-inputs with script tags, replacing braces so they are not interpreted as javascript by a browser.
5	Password left in memory	Wordpress plupload module's silverlight implementation doesn't zero memory that held a password.	CWE ID 316	M	An attacker who could get a dump of the memory of the application could retrieve passwords.	Update Wordpress. If this vulnerability is still present in an up-to-date implementation of Wordpress, stop using Wordpress.	Static analysis of the updated application doesn't flag it for CWE 316. A memory dump of the Silverlight application doesn't include any credentials, after they have been entered in the app.
6	Bad random number generation	Wordpress plupload module's silverlight implementation uses an untrusted random number generator to produce GUIDs, so the produced GUIDs will not meet the contract for a GUID.	CWE ID 331	M	An attacker could brute force guess GUIDs from the application. If these are used as session IDs or for any other sensitive purpose, an attacker could use this.	Update Wordpress. If this vulnerability is still present in an up-to-date implementation of Wordpress, stop using Wordpress.	Static analysis of the updated application doesn't flag it for CWE 331 (no use of untrusted random number generators).

7	Unsafe cryptography algorithm	Wordpress uses md5 for cryptographic hashing.	CWE ID 327	M	An attacker could brute force sensitive information hashed with md5. This is particularly risky because md5 is so computationally cheap.	Update Wordpress. If this vulnerability is still present in an up-to-date implementation of Wordpress, stop using Wordpress.	Static analysis of the updated application doesn't flag it for CWE 327 (no use of untrusted md5 or other unsafe cryptographic algorithms).
8	Directory indexing	Wordpress provides directory indexing for content.	CWE ID 548	M	Any uploads are publicly available. Metadata about files such as last modified time is exposed.	At the end of the htaccess file, add “Options -Indexes” to remove directory listings.	Wordpress no longer provides directory indexing at url ctf/wp-content/uploads/
9	Directory traversal	Wordpress uses user input to construct file paths.	CWE ID 73	M	An attacker could construct an input which would cause Wordpress to open files which should be inaccessible. This could leak sensitive information from files on the server.	Update Wordpress. If this vulnerability is still present in an up-to-date implementation of Wordpress, stop using Wordpress.	User input isn't used to construct file paths on the server.
10	Low quality password	User “bobo” on the server has an easy to guess password.	N/A	M	An attacker could easily guess bobo's password (and potentially others) since it is likely to occur in a dictionary. They could use the password to gain remote access via ssh.	Update security policy to mandate more complicated passwords, then require all users to update their passwords. Bobo's bad password is likely symptomatic of a poor security policy, and other users may also have bad passwords.	Dictionary attacks against the new passwords do not crack any of them. It is difficult to verify that this problem is solved, but we can at least check that passwords can't be guessed with common wordlists.
11	Information leakage from error messages	Board, scoreboard, and Wordpress produce unnecessarily informative error messages. E.g. board.php produces a user-visible error message containing the error message from mysql.	CWE ID 209	L	An attacker could more easily learn about the server implementation and configuration in order to construct more focused attacks.	Give users generic error messages that do not include the error messages produced by other components in the system (e.g. mysql).	Error messages visible to the user do not include any information that exposes the configuration of the server.
12	FTP Service running for no reason	Server is running an FTP service which doesn't seem to be needed.	N/A	L	If the FTP service is improperly configured or vulnerabilities are discovered, it could make the server vulnerable. There is no reason to leave an unnecessary vector of attack open.	Stop the FTP service on the server.	Server doesn't respond to netcat on ports 20 or 21.