Risk ID	Technical Risk	Technical Risk Indicators	Impact Rating	Impact	Mitigation	Validation Steps
1, 2	Code Injection (2 flaws)	Improper Neutralization of Directives in Dynamically Evaluated Code ('Eval Injection')	H	User can inject his own code into the server, causing it to do virtually anything within system admin powers.	Validate all user-supplied input to ensure conforming format.	Input benign code into user-supplied input sections, ensure it's filtered out.
42	Code Injection (1 flaw)	Improper control of filename for include/require statement in PHP program ('PHP Remote File Inclusion')	H	User can inject his own code into the server, causing it to do virtually anything within system admin powers.	Validate all user-supplied input to ensure conforming format. Use white lists to specify known safe values.	Input benign code into user-supplied input sections, ensure it's filtered out.
63, 110, 120, 130, 151, 167, 198	SQL Injection (7 flaws)	Improper Neutralization of Special Elements Used in an SQL Command ('SQL Injection')	H	Attacker can execute arbitrary SQL queries against the database, leak private data, delete critical data, etc.	Validate all user-supplied input to ensure conforming format. Avoid dynamically constructing SQL queries.	Input benign SQL injection queries, ensure they are properly formatted.
21, 41, 52, 96, 116, 132, etc.	Credentials Management (9 flaws)	Use of Hard-coded Password	M	Not easily patchable, increases chance of account being compromised, leaving all deployed instances vulnerable.	Store passwords out-of-band from the code. Follow best practices for credentials protection.	Credentials can no longer be found in application code.
6, 8, 17, 23, 91, 136, 154, 180, etc.	Cross-site scripting (75 flaws)	Improper Neutralization of Script-related HTML Tags in a Web Page (Basic XSS)	M	Allows user to embed malicious content/Javascript code, which can be executed in a victim's browser. Cookies can be manipulated or stolen, confidential information stolen, etc.	Contextual escaping on all untrusted data before constructing any HTTP response. Specific escaping methods should be chosen for specific use cases.	Input benign XSS code, ensure it is not executed.
3	Cryptographic Issues (100 flaws)	Cleartext Storage of Sensitive Information in Memory (1 flaw)	M	Sensitive information susceptible to compromise or erroneous exposure. Sensitive data accessible through core dumps or swap files.	Avoid storing sensitive data in plaintext. Clear sensitive data after use. Avoid using immutable types when storing sensitive information.	No plaintext should contain critical data, or at the very least, not older than a specified short span of time.
5		Insufficient Entropy (1 flaw)	M	Attackers can brute force the output of pseudorandom number generators to create a vulnerability.	Use trusted cryptographic random number generator instead.	Less predictable random number generation—random number generation not coming from rand(), but from

						something like CryptoAPI or OpenSSL.
33, 124, 155		Missing Encryption of Sensitive Data (3 flaws)	M	Unencrypted sensitive data passing into functions, and can be erroneously exposed because of it.	Application protects all sensitive data from unnecessary exposure, either by not passing it or by encrypting it when passed through functions.	All sensitive data is encrypted before being copied or passed.
38, 39, 45, 83, 202, 212, etc.		Use of a Broken or Risky CryptoGraphic Algorithm (95 flaws)	M	May result in the disclosure of sensitive information if problems arise in the algorithm.	Switch to a more stable cryptographic algorithm.	A stronger and more stable cryptographic algorithm is chosen for encryption.
4, 29, 92, 123	Directory Traversal (4 flaws)	External Control of File Name or Path	M	An attacker may be able to gain unauthorized access to files on the server, including those outside the webroot.	Validate all user-supplied input to ensure conformation to an expected format. Use centralized data validation routines. Use black lists for certain inputs.	Attempt to access restricted files on the server—it should fail and sanitize your input.
53, 55, 69, 82, 100, 125, 188	Information Leakage (7 flaws)	Information Exposure Through an Error Message	L	Error messages can lead attackers to launch more deadly, precise attacks. If an attack fails, they can focus their attacks further.	Only generic error messages are returned to the end user.	Error messages are only generic that do not reveal any additional details to the end user.
25, 27, 57, 67, 183, 185, 199, 203	Untrusted Initialization (8 flaws)	External Initialization of Trusted Variables or Data Stores	L	If optarg is used in an unbounded string copy, an attacker can specify overly long command line arguments and overflow the destination buffer, potentially resulting in execution of arbitrary code.	Limit size of data copied from the optarg variable.	Overly long string input arguments should be cut off and reduced to only the size limit.