Jayme Woogerd
Technical Risk Analysis
November 14, 2014

ID	Technical Risk	Risk Indicators	Impact Rating	Impact	Mitigation	Validation Steps
1	SQL injection via improperly neutralized special elements	Evidence in static analysis of source code: board.php line 30; scoreboard/index.php line 50, 60	H	Application data can be read or modified; authentication for restricted data access can be bypassed	Use parameterized prepared statements for all constructed SQL queries; always validate user-supplied input	
2	Use of hard-coded passwords	Evidence in static analysis of source code: board.php line 15, 16; scoreboard/index.php line 31, 34, 106, 109	M	Credentials much more likely to be compromised; password can't be changed without software patch;	Don't hardcode passwords, store them (properly) outside the application; follow best practices for credential storage	
3	Cross-site scripting (XSS) via improperly neutralized HTML tags	Evidence in static analysis of source code: board.php line 43, 44, 50, 58, 59, 64; scoreboard/index.php line 114; many WordPress files	M	Attackers can embed malicious content, e.g. scripts to manipulate cookies, modify (deface) site content, and compromise confidential information	Validate (escape) all user input used to construct any portion of an HTTP response; customize escaping method to specific use case (e.g. HTML, attribute escaping)	
4	Missing encryption of sensitive data	Evidence in static analysis of source code: class-ftp-sockets.php line 138; class-wp-filesytem-ftpext.php line 68, 70	M	Private data such as cryptographic keys or other sensitive information could be exposed	Ensure all sensitive data is properly encrypted	

Jayme Woogerd
Technical Risk Analysis
November 14, 2014

ID	Technical Risk	Risk Indicators	Impact Rating	Impact	Mitigation	Validation Steps
5	Use of a broken or risky cryptographic algorithm	Evidence in static analysis of source code: 95 instances across many WordPress files	M	Private data such as cryptographic keys or other sensitive information could be exposed	Ensure all sensitive data is properly encrypted	
6	External control of file name or path via directory traversal	Evidence in static analysis of source code: class-wp-upgrader.php line 1780 Text/Diff/Engine/ shell.php line 42 Text/Diff/Engine/ shell.php line 43	M	Attacker could gain unauthorized access to files on the server that are normally inaccesible to end-users	Validate user-supplied input; set file permissions appropriately; ensure server software is up to date and patched	
7	Information exposure through error messages	Evidence in static analysis of source code:	L			

Jayme Woogerd
Technical Risk Analysis
November 14, 2014

ID	Technical Risk	Risk Indicators	Impact Rating	Impact	Mitigation	Validation Steps