						Connor Blanck
Risk ID	Technical Risk	Technical Risk Indicators	Impact Rating	Impact	Mitigation	Validation Steps
1	Using PHP "eval"	line 22 of index.php	H	Adversary can run arbitrary PHP commands, including system calls.	Remove this line.	The line doesn't exist anymore, thus it cannot be exploited.
2	Allowing unrestricted system calls from PHP	line 22 of index.php	H	Adversary can run arbitrary system calls with all the permissions of the web server. This can allow them to traverse directories, execute binaries, and view sensitive information.	Limit system calls with PHP configuration, or remove the PHP eval to avoid system calls from adversaries all together.	Ensure a changed PHP config file or that line 22 of index.php is removed.
3	Improper sanitation of user input	lines 31,57,63 of index.php and line 23 of dblib.php	H	Adversary can run arbitrary SQL commands with root access.	Sanitize all user input with mysqli_real_escape or use prepared statements.	All SQL queries in user input will be escaped, thus there cannot be SQL injection.
4	Hard coded password	line 15 of index.php	M	Adversary can obtain root credentials for the MySQL database by extracting this file.	Store the password in a protected and encrypted configuration file so that it cannot be used without proper permissions.	Once the hard coded credentials are removed, establish the database connection with the proper protected configuration file to confirm that it works.
5	Cross site scripting (XSS)	lines 44, 45, 51, 59, 60, 65	M	Adversary can use embed arbitrary HTML within the page, possibly using malicious Javascript	Sanitize all user input and output by HTML escaping it with htmlspecialchars in PHP	After escaping all input and output with htmlspecialchars, attempt to input html tags as input and observe how they are output as <, > etc...
6	Buffer Overflow	line 10 of namegame.c	H	Adversary can execute malicious code by exploiting the buffer overflow.	Use a function like strncpy that allows you to specify the length of the buffer, eliminating any possibility of overflowing a buffer	Without any functions that provide an opportunity for buffer overflow, it is impossible.
7	Broken Authentication	line 23 of main.php	M	Adversary can fake a login by manually setting cookies	Don't rely on cookies for sensitive information, as they can easily be faked. Use more reliable session authentication.	Removing all checks to cookies on important conditions should fix this problem.