Matt Brenman
Computer Security
Fall 2013

Capture the Flag: Technical Risk Assessment

Risk ID	Technical Risk	Indicators	Impact Rating	Impact	Mitigation	Validation Steps
1	Code Injection (PHP)	$id = eval($_GET["id"]); in index.php (line 22)	H	User can execute arbitrary PHP code	Validate (and clean) all user input	Removing eval(…) and cleaning input will not allow user to input arbitrary code to execute.
2	SQL Injection	SQL command derived using unvalidated user input in index.php (lines 31, 57, and 63), dblib.php (line 23), ctf_scoreboard.php (line 50, 60), ctf_scoreboard-final.php (line 61, 71)	H	User can execute arbitrary SQL commands, including deleting the database	Use parameterized prepared statements to stop user input from being viewed as part of the query (other than being the variables)	Prepared statements stop user input from being viewed as part of the query, so SQL injection cannot occur
3	Credentials Management (Hard-coded passwords)	Password is stored in plaintext in index.php (line 15, used in line 18), dblib.php (lines 3, 6), ctf_scoreboard.php (31, 34, 105, 108), ctf_scoreboard-final.php (42, 45, 116, 119)	M	Account that uses that password can no longer be trusted, since the credentials are known.	Don't store passwords in plaintext or, if they have to be stored, store them in a different location, such as a properties or config file	Plaintext passwords are not available in source code. This could be checked visually.
4	Cross Site Scripting	HTTP is populated with user input in index.php (lines 44, 45, 51, 59,	M	User can execute arbitrary javascript code,	Neutralize user input so that they cannot insert HTML tags, which include	Users would need to use <> to insert a script, so without these, they cannot insert scripts

		60, 65), ctf_scoreboard.php (113), ctf_scoreboard-final.php (124)		change the appearance of the site, steal cookies, or do other malicious things	javascript tags (get rid of < and >)	
5	Information Leakage	Data that could be sensitive (passwords) or may be useful to an attacker is accessible in dblib.php (lines 8, 27), ctf_scoreboard.php (34, 108) and ctf_scoreboard_final.php (45, 119)	L	Leaking information can allow user to find data that could be useful in an attack. It is not a direct exploit, but it can be used to craft attacks.	Use only generic error messages, so that additional details (that may contain sensitive data) are not sent out to the user.	If messages are generic, there is no chance that unwanted details get sent out (since that would make the messages non-generic.
6	Separation of Concerns	When everything is as separated as possible, access to one piece does not mean access to all.	M	When an attacker breaks into one piece of the program, they have access to everything	Split up modules	When modules are not connected, an attacker must individually break into each one to get access to all instead of breaking into one and getting access to all. Fix break ins with the other risks.
7	Malformed privileges	Root password used to get into mySQL database and PHP	M	Break-ins allow for attacker to have a lot of power	Give accounts the least necessary privilege (such as only INSERT and SELECT), so that attackers have less power even if they break in.	An attacker will only have access to the privilege that the account has, so limiting the account power will limit attacker power