COMP116 Lab 9 Technical Risk Analysis

Adon Shapiro

Risk ID	1	2	3	4	5	6	7	8
Technical Risk	login.php can be easily bypassed without actual credentials	Sensitive info encoded with base64	Login.php authenticates on the client-side	SQLmap is able to retrieve sensitive info from the database with no credentials	Sensitive info in plain text (html comments, stream from port 7777, jpg file)	Improper permissions in accessible directories(FLAG.txt and .git)	A key is displayed when logging out of the application	Port 7777 is left open with sensitive info acessible
Technical Risk Indicators	SQL control sequences in logs	self-evident	Difficult to detect	Database can be accessed without authorization	self-evident	Private directories appear in web browser	self-evident	Unnecessarily open ports
Related CWE or CVE IDs	CWE-89: https://cwe.mitre.org/data/definitions/89.html	CWE-261: https://cwe.mitre.org/data/definitions/261.html	CWE-602: https://cwe.mitre.org/data/definitions/602.html, CWE-603: https://cwe.mitre.org/data/definitions/603.html	CWE-89: https://cwe.mitre.org/data/definitions/89.html	CWE-312,: https://cwe.mitre.org/data/definitions/312.html, CWE-319: https://cwe.mitre.org/data/definitions/319.html	CWE-118: https://cwe.mitre.org/data/definitions/118.html	CWE-200: https://cwe.mitre.org/data/definitions/200.html	CWE-99: https://cwe.mitre.org/data/definitions/99.html
Impact Rating	M	M	H	H	M	M	L	L

Impact	Unauthorized access to content which is not meant to be accessed	The sensitive info can be decoded by anyone and is all but completely visible	Anyone with a proxy can alter outgoing packets to make themselves look like an admin	The database is completely insecure and accessible to anyone	Since the information is visible to all, the impact depends on the sensitivity of the information. Potentially disastrous.	Files and directories that are not meant to be served to the end-user are visible and accessible to all	Some clues may be given as to the integrity or security of the system.	Access to the machine is not granted but some potentially sensitive info is revealed
Mitigation	Never trust user input! Sanitize entry fields, do not allow SQL commands.	Use strong encryption rather than simple encoding	Authenticate on the server where users cannot tamper with credentials	Require better (any) authentication to query the database	Either use encryption or place the sensitive information somewhere less visible	Use proper file permissions or disable all directory traversal	Do not give out unnecessary information to any user, especially un-privileged ones	Close all ports that are not in use by some secure service on the machine (e.g. 22 for ssh)
Validation Steps	Ensure no SQL commands will be accepted as auth attempts	Info cannot be decoded without extra info (public key, password, etc)	All authentication is done on server-side	The database rejects queries without credentials	All sensitive info is hidden or encrypted	The files or directories in question return 403 errors when access is requested	No information should be given when logging in or out	Only necessary and secure ports are open