BG Software

Advanced Security Audit and Threat Assessment Report

Created for GK Amazing App

9/18/19

4401 Hazel Avenue, Suite 225 • Fair Oaks, CA 95628 • p: 916.221.3110 • tf: 833.754.3313
bg.software

BG Software

Advanced Security Audit - GK Amazing App

Pass/Fail	Threat Vector	Location
✗	SQL injection	View/Edit Profile
✓	Cross-site scripting	
✓	Cross-site request forgery	
✓	Session ID Enumeration	
✓	Public Service (Port) Enumeration	
✓	Out-of-date platform (e.g. CMS)	
✓	Password reset username enumeration	
✓	Development errors in production	
✓	Insecure direct object references	
✓	Password sent via email	
✗	Password-only login	Login
✓	Outdated or no transport-layer security	
✓	Improper token or credential storage	
✓	Unchecked redirects	
✓	Language or framework disclosure	
✓	Bad or no password policy	

BG Software

Risk Key
Low
Moderate
High

Threat Descriptions and Proposed Remedies - GK Amazing App

Threat Vector	Issue Found	Description	Severity	Exploit Difficulty	Proposed Remedies
SQL Injection	!	SQL injection is a code injection technique in which malicious SQL statements are inserted into an entry field for execution (e.g. to dump the database contents to the attacker)	High	Low	Use prepared statements with query parameters for all variable data. Avoid using dynamic column names, table names, and SQL keywords.
Cross-site scripting		Cross-site scripting (XSS) is a code injection attack that allows an attacker to execute malicious JavaScript in another user's browser.	Moderate	Moderate	Escape user input, ensuring it is secure before rendering to the end user. Input validation and sanitization is also recommended.
Cross-site request forgery		A CSRF attack allows a hacker to force a logged-in user to perform an important action without their consent or knowledge by leveraging a legitimate user's session cookie.	High	Low	Ensure that GET requests are free of side-effects and only retrieve data. Ensure that requests come from a legitimate host by verifying "origin" and "referrer" request headers. Utilize CSRF tokens classic web applications, and a strong CORS policy on APIs.
Session ID Enumeration		An attack in which websites that use session IDs for authentication can be exploited by incrementing, decrementing, or guessing session identifiers.	Moderate	Low	Use signed session cookies, and use cryptographically random session identifiers.
Public Service (Port) Enumeration		Enumeration is the process of extracting user names, machine names, network resources, shares and services from a system. Attackers can scan public service ports to enumerate compromising information.	Moderate	Moderate	Disable unnecessary network services. Put internal services such as mail and FTP servers behind a firewall and only allow access from whitelisted IP addresses. Consider using IPSEC.
Out-of-date platform (e.g. CMS)		An outdated system can become increasingly susceptible to attacks as more known vulnerabilities are discovered for that system's version.	High	Low	Regular updates to the system should be implemented to ensure highest security standards are met.
Password reset username enumeration		Hackers can exploit differences in password reset error/confirmation messages to enumerate valid usernames and emails.	Moderate	Low	Make login/reset error messages generic enough to describe all error cases whether a valid user is found to reset or not.
Development errors in production		When comprehensive error messages remain visible in production, an attacker can potentially determine language/framework version, file structure, and other potentially hazardous information.	High	Moderate	Development errors should be disabled in production to prevent divulsion of sensitive information.
Insecure direct object references		Insecure Direct Object References occur when an application provides direct access to objects based on user-supplied input, allowing attackers to bypass authorization to access sensitive resources directly.	Moderate	Moderate	Each use of a direct object reference from an un-trusted source must include an access control check to ensure the user is authorized for the requested object.
Password sent via email		Being able to send a user their password via email implies the password is being stored somewhere in a format that can be accessed and decrypted.	Low	Low	Password reset process should involve sending the user an email containing a temporary one-time access token to allow them to reset their password, rather than showing them their password.

BG Software

Risk Key
Low
Moderate
High

Threat Descriptions and Proposed Remedies - GK Amazing App

Threat Vector	Issue Found	Description	Severity	Exploit Difficulty	Proposed Remedies
Password-only login	!	Apps without additional login metrics such as captcha or two-factor authentication are vulnerable to brute force attacks.			Additional login metrics need to be enforced.
Outdated or no transport-layer security		Deprecated or outdated security protocols that are in use will always be more vulnerable to exploit because they will be better understood and more exploits will exist for older versions.			Security protocols should be kept up to date to prevent exposure to known exploits.
Improper token or credential storage		Storing access tokens locally makes an application vulnerable to XSS, over-the-shoulder attacks, man-in-the-middle attacks, and more.			Credentials and access tokens should not be stored in the browser.
Unchecked redirects		An Open Redirection is when a web application or server uses a user-submitted link to redirect the user to a given website or page. Allowing open redirection creates opportunities for malicious users to direct other users to unsafe sites using your URL.			Do not allow user-submitted input for redirect destination. If unavoidable, validate input against whitelisted destinations.
Language or framework disclosure		Determining the framework used by a web application is the first step for an attacker to exploit known issues within a framework. These can be exposed upon inspecting the source code, or divulged in response headers.			Ensure no response headers or background information is being sent out by servers, and that all services running on open ports do not disclose this information either.
Bad or no password policy		Without a robust password policy that is enforced properly, users will be given the opportunity to provide weak, easily exploitable passwords.			Create and enforce a strict password policy, ensuring users can only provide sufficiently strong passwords.

Threat Assessment Report

Login

Password-Only Login

Affected Areas and Vulnerability Demonstration:

The login page uses a username and password without a captcha or two-factor authentication, which makes the `login.php` script susceptible to brute-force attacks. A tool such as Hydra or patator can be used to find valid login credentials.

Risk Severity and Vulnerability Impact:

Coupled with the application's lack of a password policy, the risk is high. Our security analysts quickly found and compromised an account which used an easily guessable username-password combination. They were then able to retrieve this user's personal information, and to modify information on her behalf.

Remedies:

At a minimum use a captcha on the login form so that manual interaction is required on the part of the user. Also add a delayed response when a login attempt is unsuccessful, which will further help to mitigate brute-force attacks.

- Consider offering two-factor authentication.
- Consider moving to a well-known identity provider, such as Auth0, or moving to social sign-on (e.g. log in with Facebook).

Service We Offer:

BGI can assist in integrating the login form with Google Captcha. We can also help to move the software to a cloud-based identity provider, which would remove the burden of password storage altogether.

View/Edit Profile

SQL Injection

Affected Areas and Vulnerability Demonstration:

When viewing or modifying a profile, the `userID` parameter in the URL/request body is passed to a SQL query without proper sanitization. Malicious SQL can be injected by, for example, using a `UNION` statement in place of a `userID`.

Table and column names can be enumerated by inspecting the `INFORMATION_SCHEMA` views, as demonstrated by the following code snippet:

```
1' UNION SELECT COLUMN_NAME AS first_name, TABLE_NAME AS last_name FROM
INFORMATION_SCHEMA.COLUMNS WHERE 'a' = 'a
```

The above allows an attacker to determine every table name and column name in the database, thereby making it trivial to retrieve all data. For example, usernames and password hashes can be stolen as follows:

```
1' UNION SELECT user AS first_name, password AS last_name FROM users WHERE
'a' = 'a
```

Risk Severity and Vulnerability Impact:

This is a high-risk vulnerability because it allows for the retrieval of all application data including users' credentials. Because the system's password storage mechanism does not meet modern security best practices (see below), a nefarious user could quickly garner all users' plain-text passwords.

Remedies:

A system-wide audit of all SQL statements should be undertaken. All parameters passed to SQL statements should be paremeterized using prepared statements. Variable keywords and expressions, if present, should be coded such that they are not user-controllable (for example, a user-controller ORDER BY column would be vulnerable.)

Service We Offer:

Our engineers can audit the source code, find all SQL that is susceptible to code injection, and offer consultation to your software developers on how specifically to combat each threat. We can also make a fork of the source code, correct the SQL injection vulnerabilities, and then issue a pull request for your engineers. Your team would then review the changes and reintegrate them back into the master branch of the code.

Service We Offer:

Our team can work with your developers to locate and implement a password hashing library. We can also offer consultation in the area of migrating from MD5 storage to a more modern and secure hashing algorithm. Alternatively, we can help to migrate to a cloud-based identity provider, which eliminates the need to store passwords and lowers liability.