BOTSWANA COMMUNICATIONS REGULATORY AUTHORITY

WEBSITE APPLICATION SECURITY GUIDELINES

Revision Date: 29 March 2021

DRAFT

Reviewed By

Position	**Chief Technology Officer**
Name	**Mr. Tshoganetso Kepaletswe**
Signature	
Date	

Approved

Position	**Chief Executive**
Name	**Mr. Martin Mokgware**
Signature	
Date	

Contents

1 Introduction

1.1 Website applications are gateway to accessing online service, and deals security surrounding websites, such as Application Programming Interface (APIs). Firewalls and Secure Socket Layer (SSL) do not provide protection against a web application attack, because access to the website must be made public.

1.2 All modern database systems are accessed through specific ports (e.g., port 80 and 443), and attackers can directly connect to the databases bypassing the security mechanisms of the operating system. These ports remain open to allow communication with legitimate traffic and therefore constitute a major vulnerability.

1.3 Web applications have direct access to backend data such as customer databases which has valuable data and are much more difficult to secure. Those that do not have access will have some form of script that allows data capture and transmission. If an attacker becomes aware of weaknesses in such a script, they may easily reroute unwitting traffic to another location and illegitimately steal personal details.

1.4 Web applications are a gateway to databases especially custom applications which are not developed with security best practices and do not undergo regular security audits. Perpetrators consider web applications high-priority targets due to unattended vulnerabilities, malicious code manipulation, sensitive private data collected from successful source code manipulation.

1.5 Most attacks can be easily automated and launched indiscriminately against thousands, or even hundreds of thousands of targets at a time. Websites designed and hosted without security in mind exposes the end users to exploitation and other high-risk threats. Among other consequences, this can result in information theft, damaged client relationships, revoked licenses, and legal proceedings.

1.6 Hosting companies that accept client websites which do not conform to minimum security guidelines and those that fail to secure client websites applications run the risk of attackers.

1.7 Given the aforesaid, wholistic guidelines that guard against insecure web applications online are necessary

2. TERMS AND DEFINITIONS

The following terms and definitions apply for this document:

API: means the Application Programming Interface

Cyber crime : criminal activities carried out by means of computers or the Internet;

Information: data or other knowledge that has value to the Authority.

Information Security: preservation of confidentiality, integrity and availability of information, including authenticity, accountability, relevance, non-repudiation, and reliability;

Multi-factor Authentication (MFA): Authentication method that requires the user to provide two or more verification factors to gain access to a resource such as an application, online account, or a VPN.

Threat: means any deliberate form of unjustified act or attempt to expose, alter, disable, destroy, steal or gain unauthorized access to or make unauthorized use of to an asset.

Vulnerability: weakness of a web application that can be exploited by one or more threats.

Website: client-server program run on the browser to access online services such as banking, retail, webmail, auction etc;

Website Security: the process of protecting websites and online services against different security threats that exploit vulnerabilities in an application's code.

3. PURPOSE AND OBJECTIVE

3.1 The purpose and objective of these Guidelines is to set important checklist and steps in protecting websites from exploitation; including but not limited to having good software development hygiene, continuous patching of discovered vulnerabilities, using up-to-date encryption, and requiring proper authentication.

4. SCOPE

4.1 This document outlines the security Guidelines to be implemented in the website architecture design and finished products of web applications. It will apply to all entities/Registrars hosting .bw domains and Registrants owning the websites.

4.2 The Guidelines apply to all forms of websites that belong to the .bw domain namespace with records in the registry. This includes all websites that are redirected from a .bw domain name.

5. Common website application attack vectors and common vulnerabilities

5.1 Serious weaknesses or vulnerabilities allow criminals to gain direct and public access to databases to steal sensitive data. Attackers prefer gaining access to valuable (e.g., personal data and financial details) and sensitive data residing on the database server because of the immense payoffs in selling data breaches (e.g. to dark-web). If web applications are not secure, and vulnerable to at least one of the various forms of hacking techniques, then the entire database of sensitive information is at serious risk of a web application attack.

5.2 ***Denial of Service & Distributed Denial of Service****:* overloading a targeted server and/or its infrastructure with different types of attack traffic such that the server is unable to process effectively incoming requests, and begins to slow down, eventually denying service to incoming requests from legitimate users.

5.3 ***Injection***: untrusted data sent to a code interpreter through a form input or some other data submission to a web application. Injection attacks exploit a variety of vulnerabilities to supply untrusted user input which the application then executes. Some of the most common are :_

5.3.1 **SQL Injection** : is an attack that makes it possible to execute malicious SQL statements. Attackers can use SQL Injection vulnerabilities to bypass application security measures. Attackers can authenticate and authorize the web page or web application and retrieve the content of the entire SQL database, and add, modify, and delete records in the database.

5.4 **Cross-Site Scripting (XSS)** : XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user and inject malicious code using the user input of vulnerable web applications to trick users and redirect them towards phishing sites.

5.4.1 **Operating system (OS) Command Injection:** (shell injection) is a web security vulnerability that allows an attacker to execute operating system (OS) commands on the server running an application, and fully compromise the application and all its data.

5.5 **Code injection (Remote Code Execution (RCE),** occurs when an attacker exploits an input validation flaw in software to introduce and execute malicious code. Any application that directly uses unvalidated input is vulnerable to code injection and are prime target for attackers.

5.6 ***Broken Authentication and session Management:*** *W*ebsites creates a session cookie and session ID for each valid session, and these cookies contain sensitive data like username, password, etc. A check should be done to find the strength of the authentication and session management. Keys, session tokens, cookies should be implemented properly without compromising passwords.

6. ***Insecure Deserialization:*** is a vulnerability which occurs when untrusted data is used to abuse the logic of an application, inflict a denial of service (DoS) attack, or even execute arbitrary code upon it being deserialized.
7. ***Data Exposure Vulnerability***: Sensitive data such as credit cards and cardholder Information, access credentials, health records, personal Information, finance data, or any other data the business may deem relevant can be inadeptly exposed by web applications. Sensitive data should be encrypted or kept hidden.
8. ***Memory Corruption (Buffer Overflow):*** is a condition when a program attempts to put more data in a buffer than it can hold or when a program attempts to put data in a memory area past a buffer. A buffer is a sequential section of memory allocated to contain anything from a

character string to an array of integers. Writing outside the bounds of a block of allocated memory can corrupt data, crash the program, or cause the execution of malicious code***.***

8.1.1 ***Clickjacing: i***s an attack that tricks a user into clicking a webpage element which is invisible or disguised as another element. This can cause users to download malware, visit malicious web pages, provide credentials or sensitive information, transfer money, or purchase products online.

8.1.2 ***Using Components with Known Vulnerabilities:*** This are gaps in security that have been identified, either by the developer/vendor of the products used, by the user/developer, or by the hacker/intruder. To exploit known security vulnerabilities, hackers identify a weak component in the system by scanning the system using automated tools or manually in the libraries of the application.

8.1.3 ***Security misconfigurations Vulnerabilities:*** Are configuration weaknesses that exist in software components or subsystems. For example, web server software may ship with default user accounts that an attacker can use to access the system, or the software may contain sample files, such as configuration files, and scripts that an attacker can exploit. In addition, software may have unneeded services enabled, such as remote administration functionality.

9. GUIDELINES

9.1 These guidelines are intended to ensure that all software developers and hosting companies understand the dangers and possible remediation measures of insecure website applications.

9.1.1 Implement strong password policies;

9.1.2 Require Multifactor Authentication (MFA) for login;

9.1.3 Enforce encryption and use proper key management and standard algorithms. Use SSL Certificate with at least 2048 bit and SHA 256 encryption or higher. Ensure that the SSL Certificate is valid and keep track of the certificate expiry date and take necessary action to renew/replace the certificate before expiry.

9.1.4 Disable support for SSL 2.0, SSL3.0, TLS 1.0 at the server level and use TLS 1.2 ; Disable weak ciphers like DES, 3DES, RC4. Use Strong Ciphers like AES, GCM;

9.1.5 Any “non-https” requests received on the website/applications, should be forcefully re directed to “https”;

9.1.6 Provide sufficient logging and monitoring for suspicious activities or security incidents occurring on your web application;

9.1.7 review all permissions, update configurations, and install patches and upgrades;

9.1.8 Implement secure installation processes and a system hardening process, ensuring that unnecessary, unused features or frameworks are removed or not installed at all;

9.1.9 Use Anycast services to properly route legitimate requests without a loss of service Dos and DDOS;

9.1.10 Implement Domain Name System Security;

9.1.11 Use development framework which filter XSS by Design

9.1.12 Implement Content Security Policy (CSP) as a standard way to selectively specify which content should be loaded in web applications;

9.1.13 Implement integrity checks and enforce strict type constraints during deserialization;

9.1.14 Classify application data being processed, stored or transmitted by level of sensitivity, and apply controls accordingly.

9.1.15 Only accept serialized objects from trusted sources.

9.1.16 Implement server-side input validation, sanitization checks, e to prevent hostile data within XML documents;

9.1.17 Use timeouts, and test any place where uploads are made;

9.1.18 Disable caching for responses containing sensitive data and avoid storing any data unnecessarily.

9.1.19 Use less complicated formats like JSON, avoid serialization of sensitive data;

9.1.20 Ensure that all Websites and Applications and their respective CMS (Content Management System), 3rd party plugins, codes…etc., are updated to the latest versions.

9.1.21 All exceptions should be handled appropriately. Custom error pages should be displayed for any errors/exceptions. At no point of time, a portion of source code should be displayed on the page in case of an error or exception.

9.1.22 http Only Cookies should be enabled, to restrict access to cookies.

9.1.23 All default user names and IIS/apache pages (like admin, default.aspx, index.aspx...etc) should be renamed. The access url for admin panel/CMS, should also be renamed.

9.1.24 The Web Server processes should not be running under Administrator or Root user Account.

9.1.25 A dedicated User account with limited privileges should be used for the Web Server Processes.

10. Compliance

8.1 Any website that does not conform to these guidelines may lead to website taken down, and maybe a source of malicious activities on the .bw domain name space, in accordance with the Acceptable Use Policy and the Registration Terms and Conditions found at nic.net.bw/legal-policies.

11. GUIDELINES REVIEW

11.1 These guidelines shall be reviewed by the Authority from time to time and in consultation with stakeholders.

10 GUIDELINES ENFORCEMENT

10.1 However, these guidelines are deemed appropriate to question the negligence or exercise of the duty of care of the owner of a website or any hosting company through which any suspected malicious activities may be deposited through a .bw domain.

10.2 Any Authorised Registrar or Registrant found to be in violation of the ccTLD policy will be subject to disciplinary procedures and/or legal action deemed appropriate by the Authority.

11 FURTHER INFORMATION

11.1 Further information, clarification and advice on these guidelines can be obtained from the ccTLD Unit email (registry@bocra.org.bw) or security@cirt.org.bw