Annexure 4

Application Security Audit

Web Application Security Audit

IT Auditor shall perform Security Audit of the Web application and provide “Safe to Host” certificate for the Application to go live. Also, Security Audit needs to be performed on the application before release of any new patch or version. SI should take into consideration the timelines for the Security Audit to ensure that application can go live within said time frame.

Web Application Security Audit should be performed taking into consideration the latest OWASP guidelines and consider the following:

- SQL Injection
- Broken Authentication and Session Management
- Cross-Site Scripting (XSS)
- Insecure Direct Object References
- Security misconfiguration
- Insecure Cryptographic Storage
- Sensitive Data Exposure
- Missing Function Level Access Control
- Cross-Site Request Forgery (CSRF)
- Using Known Vulnerable Components
- Un-validated Redirects and Forwards
- Failure to Restrict URL Access
- Insufficient Transport Layer Protection
- Any other attacks, which are vulnerable to web sites and web applications

The IT Auditor should also perform the following activities to assess the web applications:

- **Re-Engineering**
  - § Decompose or deconstruct the binary codes, if accessible;
  - § Determine the protocol specification of the server/client application;
  - § Guess program logic from the error/debug messages in the application outputs and program behaviour/performance;

- **Authentication**
  - § Find possible brute force password guessing access points in the applications;
  - § Find valid login credentials with password grinding, if possible;
  - § Bypass authentication system with spoofed tokens;
  - § Bypass authentication system using Injection attacks;

- § Bypass authentication system with replay authentication information;
- § Determine the application logic to maintain the authentication sessions - number of (consecutive) failure logins allowed, login timeout, etc.;
- § Determine the limitations of access control in the applications - access permissions, login session duration, idle duration;
- § Determine the transmission of authentication credentials in clear text/ encrypted/ hash form;

- **Session Management**
  - § Determine the session management information - number of concurrent sessions, IP-based authentication, role-based authentication, identity-based authentication, cookie usage, session ID in URL encoding string, session ID in hidden HTML field variables, etc.;
  - § Guess the session ID sequence and format;
  - § Determine if session ID is maintained with IP address information; check if the same session information can be retried and reused in another machine;
  - § Determine the session management limitations - bandwidth usages, file download/upload limitations, transaction limitations, etc.;
  - § Gather excessive information with direct URL, direct instruction, action sequence jumping and/or pages skipping;
  - § Gather sensitive information with Man-In-the-Middle attacks;
  - § Inject excess/bogus information with Session-Hijacking techniques;
  - § Replay gathered information to bypass session authentication;

- **Input Manipulation**
  - § Verify if input validation is happening at client or server or at both ends;
  - § Find the limitations of the defined variables and protocol payload - data length, data type, construct format, etc.;
  - § Use exceptionally long character-strings to find buffer overflows vulnerability in the applications;
  - § Concatenate commands in the input strings of the applications;
  - § Inject SQL language in the input strings of database-tiered web applications;
  - § Examine "Cross-Site Scripting" in the web applications of the system;
  - § Examine unauthorized directory/file access with path/directory traversal in the input strings of the applications;
  - § Use specific URL-encoded strings and/or Unicode-encoded strings to bypass input validation mechanisms of the applications;
  - § Execute remote commands through "Server Side Include";
  - § Manipulate the session/persistent cookies to modify the logic in the server-side web applications;
  - § Manipulate the (hidden) field variable in the HTML forms to modify the logic in the server-side web applications;
  - § Manipulate the "Referrer", "Host", etc. HTTP Protocol variables to modify the logic in the server-side web applications;

- Use illogical/illegal input to test the application error-handling routines and to find useful debug/error messages from the applications;

- **Output Manipulation**
  - Retrieve valuable information stored in the cookies;
  - Retrieve valuable information from the client application cache;
  - Retrieve valuable information stored in the serialized objects;
  - Retrieve valuable information stored in the temporary files and objects;
  - Retrieve bulk information/ multiple rows from database;

- **Information Leakage**
  - Find useful information in hidden field variables of the HTML forms and comments in the HTML documents;
  - Find valuable information stored in the HTML source code on browser like Unencrypted View State;
  - Examine the information contained in the application banners, usage instructions, welcome messages, log-out messages, application help messages, debug/error messages, etc.;

Penetration Testing

To identify ways to exploit vulnerabilities to circumvent or defeat the security features of system components through manual process that may include the use of vulnerability scanning or other automated tools.

Standards

In addition to the internal IT policies, the IT Auditor should adhere to all the applicable laws of land and rules, regulations and guidelines prescribed by various regulatory, statutory and Government authorities, multiple industry-accepted methodologies during the execution such as the following:

- National Critical Information Infrastructure Protection Center (NCIIPC)
- The National Institute of Standards and Technology (“NIST”) Special Publication 800-115
- Cert-In Guidelines
- Latest ISO27001
- Open Source Security Testing Methodology Manual (“OSSTMM”)
- OWASP Testing Guide
- Penetration Testing Execution Standard
- Penetration Testing Framework

Security Audit Documentation

The Security Audit report should contain details of all steps, test vectors, and exploited vulnerabilities that lead to positive and /or false positive penetration during testing for which remediation and retesting are required. It is also important to identify vulnerabilities that are not always exploitable but may pose a potential risk to the environment.

The report shall have the following sections:

a. **Executive Summary**: Brief high-level summary of the penetration test scope and major findings with overall severity graph;

b. **Methodology:** Details on the methodologies used to complete the testing;

c. **Constraints:** Document any restrictions imposed on testing such as designated testing hours, bandwidth restrictions, special testing requirements for legacy systems, etc.;

d. **Summary of test results:** Detailed results for vulnerabilities discovered, exploited vulnerabilities and proof of concepts/screenshots, detailed explanations of the implications of findings, business impacts, and risks for each of the identified vulnerabilities;

e. **Recommendations:** Remediation recommendations to close the deficiencies identified. Detailed steps (wherever/whenever applicable) to be followed while mitigating the reported deficiencies. Security issues that pose an imminent threat to the system are to be reported immediately;

f. **Tools:** Details of all the tools used, purpose of each tool and the impact of each tool on the testing;

g. **Clean up:** After testing, there may be tasks to be performed to restore the target environment (e.g. update/removal of test accounts or database entries added or modified during testing, uninstall of test tools or other artifacts, restoring active protection-system settings, and/or other activities the tester may not have permissions to perform, etc.). Provide directions on how clean up should be performed and how to verify that security controls have been restored.

Retest (if required)

If significant vulnerabilities are identified from the Security Audit, the SI will be required to fix the identified vulnerabilities within a time period, as mutually agreed with WDRA. Thereafter, IT Auditor shall perform a retest to validate if the changes mitigate the original risk. The scope of a retest should consider whether any changes occurring as a result of remediation identified from the test are classified as significant. A Security Audit report, as per the above specifications, should be prepared after the retest.