PARK PLACE TECHNOLOGIES

SOFTWARE INTEGRITY CERTIFICATE

The Bank of India

This is to certify that our product Entuity Software™, Version 22.0 (see Annexure 1) developed by and a copyright of Entuity Limited, follows standard security practices and has been tested and certified for the following checks:

i. That the application has undergone formal unit, system, and stress tests and is free of any obvious bugs that would cause serious impact to functionality or security.

ii. That the software is developed and tested within Entuity's strictly controlled development environment, which meets corporate information security requirements, and is free of any known virus/malwares at the time of sale.

iii. That the application is free of any covert channels in the code being provided and subsequent modifications to be done on them.

iv. We have evaluated the cryptographic implementation and have ensured that only cryptographic modules based on authoritative standards and reputable protocols are used.

v. We confirm that Source code testing is carried out on application source code (to identify and detect security threats and weaknesses in its systems) and there are no open Critical or High vulnerabilities. In turn, Medium and Low vulnerabilities are considered on a case by case basis and either a) resolved before release, b) resolved after release in a future patch, c) not resolved but can be mitigated, d) confirmed as not exposed by Entuity, or e) determined to be not serious enough to fix (before or after release).

vi. We confirm that Application Security testing is carried out for application (to identify and detect security threats and weaknesses in its systems) and there are no open Critical or High vulnerabilities. In turn, Medium and Low vulnerabilities are considered on a case by case basis and either a) resolved before release, b) resolved after release in a future patch, c) not resolved but can be mitigated, d) confirmed as not exposed by Entuity, or e) determined to be not serious enough to fix (before or after release).

vii. We confirm that we are conducting formal security training programs for our software developers/testers on periodical basis, covering OWASP top 10 and more.

PARK PLACE
TECHNOLOGIES

viii. We herewith inform that there are some Open Source libraries/software (with details in Annexure–1) that are used in this product/application. We confirm that all Open Source Libraries/software used in this product/application will be updated for vulnerability mitigation by us on regular basis as per industry best practises.

ix. We confirm that the CERT-In Guidelines for secure Application Design, Development, Implementation and Operations mentioned in the Annexure 2 are followed by us during application development lifecycle as far as is reasonable, and as per applicability and wherever required.

Authorized Signatory .....

Designation: VP Software Engineering, Entuity Ltd

Date: 29 November 2024

PARK PLACE TECHNOLOGIES

Annexure 1

Entuity v22.0 (build 67216) checksums:

Windows filename is 546522_ETSOM_22.0.00.windows.iso.zip

Type	Checksum
MD5	a68c1d83b69673bc7243b82cb69ff3da
SHA1	faeb5f9be2981f0683d292ebf53962fb442b50e6
SHA256	ff628d4ce8341cee833ccf613f11711192f032018605e8d380ce20ec3bb9fe5b

Linux filename is 546523_ETSOM_22.0.00.linux.iso.gz

Type	Checksum
MD5	0948be4b82b6ca15e4b7fdf9eb7bfacb
SHA1	5f4af5911b0eb7a0243f7c81a979e091d58d594c
SHA256	16d4f91e0e8b8b346a9c944c5eb25b2be9b20504db40d197218915e92421364c

Key 3rd party software used:

Product	Version
Apache HTTPD	2.4.62
Apache Tomcat	9.0.90
DWR	3.0.2
Groovy	2.4.13
Guava	20.0
JNA	5.6.0
JRE	21.0.4
Log4j	2.18.0
MariaDB Server	10.6.11
MariaDB Java Client	1.5.9
Quartz	2.3.1
Spring	5.3.39
Highcharts	10.1.0
Mibble	2.9.3
npcap	1.60
Maverick SSH	1.7.57

PARK PLACE
TECHNOLOGIES

Annexure 2

CERT-In Guidelines for Secure Application Design, Development, Implementation and Operations:

1. Establish the Context of the Security in Designing of Application:

   a. Adoption of Secure Software Development Life Cycle such as Microsoft Secure Development Lifecycle (SDL), Open Web Application Security Project (OWASP), Agile Secure Development Lifecycle, NIST Secure Software Development Framework (SSDF).
   b. Engagement of security trained designers and developers in the application development.

2. Implement & Ensure Secure Development Practices:

   a. Incorporate coordinated actions that integrate secure authentication, session management, and access control, all in harmony with the fundamental principles of safeguarding data privacy and protection.
   b. Cryptographic practices to be followed to ensure data is encrypted to make it secure and unreadable (encryption), create distinct signatures to confirm the validity and integrity of data (digital signatures) and produce data representations with defined sizes (hashing).
   c. Proper input validation using regex techniques and built-in security controls provided by programming frameworks in order to validate and sanitize all user inputs should be implemented to prevent common vulnerabilities
   d. Ensure that special characters or sequences in the application code are converted into an alternative, yet equivalent format that poses no threat when processed by the intended interpreter
   e. Proper allocation and deallocation of memory in a program to prevent vulnerabilities like buffer overflows and memory leaks.
   f. The coupling and dependencies should be minimized by including modular architectures, dependency injections, design patterns, building loosely coupled system, divide the application into multiple independent units of execution, namely processes and threads, and then executing them in parallel.
   g. Develop a software technology / language specific secure coding checklist containing defined set of rules and guidelines dealing specifically with the peculiarities, defects and non-standard extensions in respective programming language / technology.
   h. The applications should be developed using Security Test Driven Development (STDD). It involves writing security tests before writing the actual code to ensure that vulnerabilities are identified early and addressed before the code is deployed to production.
   i. It is recommended to utilize environment variables instead of plain text files. Furthermore, it is important to treat input from environment variables as untrusted and validate it accordingly (sensitive information not to be stored in environment variables).
   j. Stored procedures should be used instead of constructing SQL statements to reduce the chances of injection attacks (stored procedures restrict direct user access to the database table).
   k. Prior to deploying an application into a production environment, it is crucial to remove all commented code. Additionally, it is important to ensure that sensitive information, such as system details, network configurations, passwords, server names, IP addresses, or file system paths, is not exposed through error messages or URL contents.

l. Implement proper access control using 'public,' 'private,' or 'protected' access modifiers in the code to specify the location or physical address of the data structures. Avoid multiple inheritances to mitigate potential code vulnerabilities, as declaring essential variables in the public scope can compromise the abstraction principle in object oriented programming.
m. Security wrapper classes should be created for the open-source and third-party libraries used in the code. An organization should integrate third-party components, open-source components and APIs into their development projects only after conducting a thorough evaluation of their credibility and security posture.
n. Data received from client-side validations should be treated as untrusted and undergo validation on the back-end.
o. Employ the principle of least privilege, ensuring that only the minimum necessary privileges are assigned.
p. Threat modelling should be implemented in the application development process by first identifying the boundaries of the application. Thereafter, data flow diagram should be created to visualize the possible access points that attackers may exploit, along with the associated threats. Subsequently, countermeasures should be developed and tested to mitigate these threats.
q. Version control to be maintained to manage and control modifications in application configurations and source code.

3. Provision of Detection of Errors and Vulnerability in Application Design & Development:
   a. Conduct Source Code review to identify security flaws or vulnerabilities.
   b. Conduct Security Vulnerability Assessment.
   c. Conduct penetration testing iteratively with each release or modification of the source code to continually identify and address potential security threats in the application.
   d. Logging and audit trail functionality should be integrated in the application for troubleshooting and compliance requirement.