RCSD ROSEVILLE CITY SCHOOL DISTRICT
Est. 1869

Technology Services

1050 Main Street Roseville, CA 95678
Phone (916) 771-1645 Fax (916) 771-1650

Laura Assem, Executive Director of Technology

Vendor Statement of Compliance
Data Privacy and Protection

This agreement is entered into between the Roseville City School District ("LEA" or "District") and Treering Corporation ("Service Provider") on 08/01/2024 ("Effective Date").

WHEREAS, the LEA and the Service Provider entered into an agreement for Educational Technology services;

WHEREAS, the LEA is a California public entity subject to all state and federal laws governing education, including but not limited to California Assembly Bill 1584 ("AB 1584"), the California Education Code, the Children's Online Privacy and Protection Act ("COPPA"), and the Family Educational Rights and Privacy Act ("FERPA");

WHEREAS, AB 1584 requires, in part, that any agreement entered into, renewed or amended after January 1, 2015, between a local education agency and a third-party service provider must include certain terms;

NOW, THEREFORE, the Parties agree as follows:

Section I: General - All Data

1. **PASSWORD SECURITY.** All passwords are considered secure. Vendors may not disseminate any passwords unless specifically directed by Educational or Technology Services management. Vendors will not provide information concerning Admin accounts (ROOT Admin, container Admin, local NT administrator or Domain administrator) or their equivalent to any persons. District personnel ONLY will disseminate this information. Vendors will never create "back door" or "generic" user accounts on any systems unless specifically directed to do so by LEA management.

   Agree: Yes ◉ No ○

2. **SYSTEM SECURITY.** Unauthorized access to or modification of District systems including file servers, routers, switches, NDS and Internet services is prohibited. Any attempt to bypass or subvert any District security system, both hardware, and software is prohibited.

   Agree: Yes ◉ No ○

3. **PRIVACY**. The vendor will adhere to all provisions of the Federal Family Educational Rights and Privacy Act (FERPA, 20 U.S.C. 123g), California Education Code and district policies regarding the protection and confidentiality of data. At all times, the vendor will consider all data collected in the course of their duties to be protected and confidential. Release of this data can only be authorized by Technology & Information Services management and state and federal law.

   Agree: Yes ◉ No ○

RCSD ROSEVILLE CITY SCHOOL DISTRICT
Est. 1869

Technology Services

1050 Main Street Roseville, CA 95678
Phone (916) 771-1645 Fax (916) 771-1650

Laura Assem, Executive Director of Technology

Section I: General - All Data *(Continued)*

4. **REUSE**: Vendors shall not copy, duplicate, sell, repackage or use for demonstration purposes any Roseville City School District data without the prior, written consent of Educational or Technology Services management.

   Agree: Yes ◉ No ○

5. **TRANSPORT**: Vendor must provide a secure channel (S/FTP, HTTPS, SSH, VPN, etc) for the District to "push" data to the vendor and to extract data as required. Vendors will not have direct access to District systems and will not "pull" data at any time.

   Agree: Yes ◉ No ○

6. **EXTERNAL SECURITY:** Vendor must attach to this document reasonable evidence that their system is secure from external hacking and attacks. Devices such as firewalls and technologies such as NAT are the minimum requirements. Active IDS or similar technology is preferred.

   Agree: Yes ◉ No ○

7. **INTERNAL SECURITY:** Vendors must attach to this document reasonable evidence that their system is secure from internal hacking and attacks. Describe the interactions vendor personal (or their representatives) will have directly with District data. How is uploaded data from the District handled and processed? Who has access to this data? What happens to the data after the upload is complete? What security safeguards are in place to protected unauthorized access to District data? How are backup performed and who has access to and custody of the backup media? How long are backup maintained; what happens to the District data once the backup is "expired"? If any data is printed, what happens to these hard copy records?

   Agree: Yes ◉ No ○

8. **DISTRICT ACCESS:** Vendor must provide a secure means (see Item 5 above) for the District to extract ALL data from the vendor system. This can either be an online extraction tool or a vendor-provided extract as needed by the District (not to exceed quarterly). Describe the means and format of the data (delimited, Excel, MDB, SQL Dump).

   Agree: Yes ◉ No ○

9. **TERMINATION:** Upon termination of this agreement as provided herein, the vendor will permanently delete all customer data from their system as allowed by state and federal law. Vendor may be required to certify the destruction of LEA data within 90 days of contract termination.

   Agree: Yes ◉ No ○

ROSEVILLE CITY SCHOOL DISTRICT
Est. 1869

Technology Services

1050 Main Street Roseville, CA 95678
Phone (916) 771-1645 Fax (916) 771-1650

Laura Assem, Executive Director of Technology

Section II: AB1584 Compliance - Student Information Only

1. Vendor agrees that the Roseville City School District retains ownership and control of all student data.

   Agree: Yes ◉ No ○

2. Vendor must attach to this document a description of how student-created content can be exported and/or transferred to a personal account.

   Agree: Yes ◉ No ○

3. Vendor is prohibited from allowing third-parties access to student information beyond those purposes defined in the contract.

   Agree: Yes ◉ No ○

4. Vendor must attach to this document a description of how parents, legal guardians and students can review and correct their personally identifiable information.

   Agree: Yes ◉ No ○

5. Vendor will attach to this document evidence how student data is kept secure and confidential.

   Agree: Yes ◉ No ○

6. Vendor will attach to this document a description of procedures for notifying affected parents, legal guardians or eligible students when there is an unauthorized disclosure of student records.

   Agree: Yes ◉ No ○

7. Vendor certifies that student records will not be retained or available to a third party once the contract has expired or is canceled (See Page 2, Item 9).

   Agree: Yes ◉ No ○

8. Vendor will attach to this document a description of how they and any third party affiliates comply with FERPA.

   Agree: Yes ◉ No ○

9. Vendor and its agents or third parties are prohibited from using personally identifiable information from student records to target advertising to students

   Agree: Yes ◉ No ○

RCSD ROSEVILLE CITY SCHOOL DISTRICT
Est. 1869

Technology Services

1050 Main Street Roseville, CA 95678
Phone (916) 771-1645 Fax (916) 771-1650

Laura Assem, Executive Director of Technology

Section III: SB 1177 SOPIPA Compliance - Student Information Only

1. Vendors cannot target advertising on their website or any other website using information acquired from students.

   Agree: Yes (●) No ( )

2. Vendors cannot create a profile for a student except for school purposes as defined in the executed contract.

   Agree: Yes (●) No ( )

3. Vendors cannot sell student information.

   Agree: Yes (●) No ( )

4. Vendors cannot disclose student information unless for legal, regulatory, judicial, safety or operational improvement reasons.

   Agree: Yes (●) No ( )

5. Vendors must attach to this document evidence of how student information is protected through reasonable security procedures and practices.

   Agree: Yes (●) No ( )

6. Vendors must delete district-controlled student information when requested by the District.

   Agree: Yes (●) No ( )

7. Vendors must disclose student information when required by law, for legitimate research purposes and for school purposes to educational agencies.

   Agree: Yes (●) No ( )

As an authorized representative of my organization, I accept the conditions listed in this document.

Tevis Diaz, Sales Representative
Print Name

Tevis Diaz
Signature, Date

Laura Assem
Print Name (Roseville City School District)

8/5/2024
Signature, Date (Roseville City School District)

ROSEVILLE CITY SCHOOL DISTRICT
Est. 1869

Technology Services

1050 Main Street Roseville, CA 95678
Phone (916) 771-1645 Fax (916) 771-1650

Laura Assem, Executive Director of Technology

EXHIBITS

Section 1.6: External Security

See attached CASA Certification and Pen Test results

Section 1.7: Internal Security

PII is provided from end-users or the school directly when managing and creating accounts for users in their community. It is not shared with other members of the community.

Automated snapshot backups of our primary production databases are produced on a nightly basis and are encrypted at rest using AWS KMS key stores. These backups are retained for a period of 7 days at which point they are automatically deleted permanently. Snapshots of production data are used for the purpose of restoring data in the case of data loss as the result of a bug or other system incident; upon support ticket escalation for the recovery of data that has been lost as a result of user action in the application that they want to go back in time; or for the purpose of performing a database migration test flow prior to a production upgrade. Access to snapshots are limited to personnel on the engineering team. Access is managed using AWS Identity Center SSO IAM roles requiring MFA.

Data is never printed.

Section II.2: Exporting of Student-Created Content

N/A
Student content is not transferable to personal accounts.

Section II.4: Review and Correcting Personally Identifiable Information (PII)

PPI can be corrected via end-user portal.

ROSEVILLE CITY SCHOOL DISTRICT
Est. 1869

Technology Services

1050 Main Street Roseville, CA 95678
Phone (916) 771-1645 Fax (916) 771-1650

Laura Assem, Executive Director of Technology

EXHIBITS

Section II.5: Securing Student Data

This data is stored in a MySQL database hosted by AWS data center. It operates as an Aurora 3 instance.

All connections and transfers of data use TLS v1.2 with accepted cipher sets: ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES256-GCM-SHA384, ECDHE-RSA-AES256-SHA384. Keys used for the encryption of data at rest are stored using approved service providers like AWS Secrets Manager and KMS. User data is generally stored in a MySQL database running on Aurora 3 using KMS encryption at rest. This is applied for all backups of said database, as well.

Section II.6: Disclosure Notification

The Treering Response Team will notify users either directly or indirectly via public communication.

Section II.8: Family Educational Rights and Privacy Act (FERPA) Compliance

Data Access Controls: We implement appropriate data access controls and authentication mechanisms to safeguard student records.

We update the school community whenever we change our TOS, which include FERPA reminders.

Amendment Requests: We allow parents or eligible students to request amendments to records they believe are inaccurate.

Written Consent: We obtain written permission from parents or eligible students before disclosing any information from a student's education record, except under certain conditions.

Section III.5: How Student Data is Protected:

All data is encrypted at rest and in transit. We have a written company policy regarding Data Access and Use Policies which includes a section on Cryptographic Protection Standards. All connections and transfers of data use TLS v1.2 with accepted cipher sets. Keys used for the encryption of data at rest are stored using approved service providers like AWS Secrets Manager and KMS. User data is generally stored in a MySQL database running on Aurora 3 using KMS encryption at rest. This is applied for all backups of said database, as well.

Reset Form

App Defense Alliance | **CASA**

Assessment Performed by Bishop Fox

Application Name: TreeRing Yearbooks

Certification ID: 2e6a32b4

Assessment Type: Tier 3 (Lab Tested - Lab Verified)

Issue Date: 7/18/2024 12:49:29 PM

Assessment Status: Complete

Expiration Date: 7/19/2025

Statement of Validation

The purpose of this report is to provide users verification that TreeRing has successfully completed a Cloud Application Security Assessment (CASA), validating TreeRing Yearbooks **has satisfied CASA application security requirements.** In meeting these assessment requirements, TreeRing Yearbooks is **verified** to meet the CASA **Tier 3** requirements.

The assessment was conducted by Bishop Fox an independent third party lab, authorized by the App Defence Alliance to conduct CASA security assessments.

About CASA

CASA is based on the industry-recognized Open Web Application Security Project (OWASP) Application Security Verification Standard (ASVS) to provide third-party (3P) application developers with:

A basis for testing technical application security controls

A consistent set of requirements for secure application development

A homogenized coverage and assurance levels for providing security verification using industry-aligned frameworks and open security standards.

More information on CASA, including a complete list of CASA requirements is located on the App Defence Alliance Site.

Assessment Scope

The assessment was conducted using CASA requirements. Please note that not all requirements apply to every application. Additionally, some applications may be validated for specific requirements based on pre-existing certifications that have been mapped to the CASA requirements. A list of mapped certifications and corresponding requirements can be found here.

Category	Status

Architecture, Design and Threat Modeling Requirements	Pass
Authentication Verification Requirements	Pass
Session Management Verification Requirements	Pass
Access Control Verification Requirements	Pass
Validation, Sanitization and Encoding Verification Requirements	Pass
Stored Cryptography Verification Requirements	Pass
Error Handling and Logging Verification Requirements	Pass
Data Protection Verification Requirements	Pass
Communications Verification Requirements	Pass
Malicious Code Verification Requirements	Pass
Business Logic Verification Requirements	Pass
File and Resources Verification Requirements	Pass
API and Web Service Verification Requirements	Pass
Configuration Verification Requirements	Pass

Terms Limitations and Disclaimers

TREERING

DECEMBER 2023

CONFIDENTIAL MATERIAL

This document contains confidential and privileged material intended only for the intended recipient. Any unauthorized interception, review, retransmission, dissemination, or use of this information, or any action taken based on this information, is prohibited by law and may result in criminal or civil liability.

Proprietary and Confidential Information includes but is not limited to performance, sales, financial, contractual, and specific marketing information, ideas, technical data, and concepts originated by the disclosing party, its subsidiaries, and affiliates. This information is not previously published or otherwise disclosed to the general public, not available without restriction to the receiving party or others, and not generally furnished to others without compensation. The disclosing party desires to protect this information against public disclosure or competitive use, and it is furnished in accordance with this document and appropriately identified as proprietary when furnished.

INDEPENDENT SECURITY ASSESSOR REPORT

To Whom It May Concern:

Halo Security, LLC (HS), in our role as an independent security assessor, has performed penetration testing of the external facing assets for **TREERING** in accordance with audit procedures and testing methods (Criteria) established with the (Client) such that Criteria is consistent and meet the requirements for network and application penetration testing as established by the Payment Card Industry (PCI) Data Security Standard requirements 11.3.

Our examination was conducted in accordance with information system security assessment best practices as described by the Open Source Security Testing Methodology Manual (OSSTMM) and The National Institute of Standards and Technology (NIST) Special Publication 800-42, Guideline on Network Security Testing and included: (1) obtaining an understanding of Client external network and application systems (2) researching publicly available information sources to identify any sensitive or confidential information about Client systems; (3) attempting to penetrate (exploit) these systems using vulnerabilities discovered; (4) performing such other procedures as we considered necessary in the circumstances. We believe that our examination provides a reasonable basis for our qualified opinion in this report provided in **DECEMBER 2023**.

Because of inherent limitations, security risks, errors, misrepresentations, or changes made to the subject environment during testing, misuse of the Client's systems may occur and not be detected. Even upon completion of our work, we may not identify all security issues or recommend all possible remedial actions. Furthermore, the projection of any conclusions, based on our results, to future periods is subject to the risk that (1) changes made to the system or controls, (2) changes in processing requirements, or (3) a degree of compliance with the policies or procedures may alter the validity of such conclusions.

This report does not represent the quality of Client goods or services nor their suitability for any customer's intended purpose. Further, this letter is intended solely for the benefit of Client in connection with the matters described above and is not to be used for any other purpose without the prior written consent of this firm.

INDEPENDENT SECURITY ASSESSOR

The following assessor(s) worked on aspects of this project, performed remote investigations, reviewed documentation, and contributed to the presentation of this report.

Nick Merritt
VP of Security
https://www.halosecurity.com
Nick Merritt

TABLE OF CONTENTS

ENGAGEMENT

Halo Security used commercially available tools, public domain tools, and customized scripts to conduct the security testing. Halo Security used “attacker” techniques to identify security vulnerabilities and design specific tests to validate the presence of the vulnerabilities in the environment.

PROJECT OBJECTIVES

The objectives of the security testing were to gain control of the target systems or determine if an attack could potentially access sensitive data. Halo Security evaluated the protection of the client's information technology assets (i.e., data, systems, and processes), focusing on the effectiveness of logical access and system software controls. All tests performed by Halo Security aimed to provide value to the client's security efforts by identifying opportunities to improve applicable controls.

PROJECT SCOPE

The "in-scope" assets refer to the systems, networks, and applications that the penetration tester is authorized to test and assess for vulnerabilities. These assets are typically identified in a scope document, which outlines the specific objectives and constraints of the test. On the other hand, "out-of-scope" assets refer to systems, networks, and applications that the penetration tester is not authorized to test and assess. The scope of the test must be clearly defined and agreed upon by all parties involved to ensure that the test is conducted ethically and lawfully and that the test results are accurate and actionable.

TESTING DEPTH

All systems within the target list were in scope for reconnaissance and baseline activities. All systems underwent automated and manual vulnerability scanning to identify as many potential weaknesses as possible. The testing time frame allowed the evaluation of all network services visible to unauthenticated users. All vulnerabilities found were evaluated and reviewed to verify their presence and help quantify the risk to the environment. If vulnerabilities were identified, they were only fully exploited to the degree necessary to evaluate the risk to the systems identified in the target scope with the explicit permission of client personnel.

TESTING METHODOLOGY

Our web application testing methodology is derived from the OSSTMM and OWASP best practices and combined with current threat intelligence and industry experience. Manual testing is performed to look for the following vulnerability categories and others where applicable in areas that may be missed by automated scanning technology.

SQL INJECTION - Penetration testing for SQL injection uncovers vulnerabilities in web app SQL databases. Such exploits inject SQL commands allowing unauthorized access or sensitive data manipulation. Results could be used to fix and secure the database.

AUTHENTICATION FLAWS - Penetration testing attempts to exploit vulnerabilities in a web application's authentication mechanisms to bypass or circumvent the authentication process for gaining unauthorized access. Results can identify and remediate vulnerabilities to prevent unauthorized access to sensitive data.

DIRECTORY TRAVERSAL - Penetration testing exploits web application vulnerabilities, navigates authorized directories and identifies and remediates vulnerabilities to prevent unauthorized file access.

OS COMMAND INJECTION - Penetration testing involves exploiting web app vulnerabilities to inject and execute malicious commands on the OS command shell. The results help identify and fix security flaws to prevent the unauthorized execution of harmful commands.

BUSINESS LOGIC VULNERABILITIES - Penetration testing identifies logic flaws that can compromise security controls, bypass access restrictions, or access sensitive data in web application workflows.

INFORMATION DISCLOSURE - Penetration testing identifies web application vulnerabilities that may expose sensitive information. Its goal is to access confidential data and system configuration while exploiting weaknesses. The results can be used to prevent unauthorized information disclosure.

ACCESS CONTROL VULNERABILITIES - Penetration testing for access control vulnerabilities exploit weaknesses in a web application's access controls to bypass authentication or authorization mechanisms. The results can help identify and remediate vulnerabilities to prevent unauthorized access and ensure proper access controls.

SERVER-SIDE REQUEST FORGERY (SSRF) - Penetration testing for SSRF involves exploiting a web application's vulnerability to make unauthorized requests from the server, tricking it into accessing potentially malicious external systems, and compromising security.

XML EXTERNAL ENTITY (XXE) INJECTION - Penetration testing for XXE involved trying to exploit web app vulnerabilities by injecting malicious XML entities in the input fields, tricking the server into executing external requests to compromise the server or expose sensitive data.

CROSS-SITE SCRIPTING (XSS) - Penetration testing for XSS involves exploiting vulnerabilities that allow malicious script injection into a web app. The test aims to prevent unauthorized access and misuse of the app's functionality by identifying and repairing vulnerabilities that may lead to data theft or a user's session compromise.

CROSS-SITE REQUEST FORGERY (CSRF) - Penetration testing for CSRF involves exploiting vulnerabilities in web apps that trick users into executing unauthorized actions. Identify and remediate the vulnerabilities to prevent unauthorized access and misuse of sessions.

CROSS-ORIGIN RESOURCE SHARING (CORS) - Penetration testing for CORS involves exploiting a vulnerability in a web app to gain unauthorized access to other domains. This can expose sensitive data or functionality from a different domain. Results help identify and remediate vulnerabilities to prevent unauthorized access, ensuring protection against CORS attacks.

CLICKJACKING - Penetration testing involves exploiting a vulnerability in a web app to manipulate user interactions. The goal is to trick the user into clicking a disguised or hidden element, enabling unauthorized actions. Findings help identify, and fix vulnerabilities, prevent unauthorized access, protect user sessions, and secure against clickjacking attacks.

DOM-BASED VULNERABILITIES - DOM-based penetration testing exploits web app's DOM vulnerabilities to inject code and perform unauthorized actions. Results are used to remediate flaws, ensure security, and prevent attacks.

WEB-SOCKETS VULNERABILITIES - Penetration testing aims to identify and resolve vulnerabilities in the WebSocket protocol used by web applications. The weaknesses tested are gaining unauthorized access, tampering with data, and accessing sensitive information. Results are used to remediate vulnerabilities and safeguard resources from WebSocket-based attacks.

INSECURE DESERIALIZATION -Penetration testing exploits insecure deserialization vulnerabilities in web application data handling by manipulating data to execute unauthorized actions, access sensitive information, or bypass access controls. Tests identify and fix these issues, ensuring the application is not exposed to attacks.

SERVER-SIDE TEMPLATE INJECTION (SSTI) - Penetration testing for SSTI involves exploiting vulnerabilities in a web app's template engine to inject malicious code. Results could prevent misuse, unauthorized access, and SSTI attacks.

WEB CACHE POISONING - Penetration testing aimed to exploit web application cache vulnerabilities by injecting malicious content that compromised users' sessions or stole sensitive information. Such testing could identify and remediate security weaknesses to prevent unauthorized access, misuse of web resources, and web cache poisoning attacks.

HTTP HOST HEADER ATTACKS - Penetration testing for HTTP host header attacks involves exploiting a web app vulnerability by manipulating the HTTP Host header to misdirect requests to another domain/IP. This helps identify and fix the issue to prevent unauthorized data theft, misuse of web app resources, and server compromise.

HTTP REQUEST SMUGGLING - Penetration testing for HTTP request smuggling exploits web app vulnerabilities, manipulating HTTP requests to bypass security controls or gain unauthorized access. This identifies and remediates vulnerabilities, preventing misuse of web app resources or HTTP smuggling attacks.

SSO AUTHENTICATION FLAWS - Penetration testing targeted Single Sign-On vulnerabilities by exploiting flaws in web application's authentication mechanisms. Results were used to address and prevent unauthorized access while ensuring secure and proper access to sensitive data.

SEVERITY RATING

The severity levels of findings can help businesses prioritize their efforts to remediate security issues. Severity levels are assigned to findings, indicating their level of criticality. High-severity findings are typically the most severe and require immediate attention, while medium-severity findings are less severe but still require action.

A **HIGH** severity finding would generally refer to a security issue that poses a significant risk to the confidentiality, integrity, or availability of sensitive data. This could include, for example, a critical vulnerability in a system that could allow an attacker to access or manipulate sensitive data or a failure to properly encrypt sensitive information.

On the other hand, a **MEDIUM** severity finding would generally refer to a security issue that poses a lower risk to the security of sensitive data. This could include, for example, a configuration issue that could potentially allow unauthorized access to systems or a failure to properly implement access controls for sensitive information.

A **LOW** severity finding refers to a security issue that poses a minor risk to the security of sensitive data, such as a missing security control or a minor configuration issue that does not expose sensitive data. Examples might include failing to implement password complexity requirements or automatic logoff. While not as critical as high-severity issues, low-severity findings should be addressed promptly to prevent them from escalating and improve overall security posture. Severity levels may vary depending on the circumstances, and it is the responsibility of the business and security team to prioritize and address findings to maintain security and protect sensitive data.

In general, businesses should prioritize remediation of high-severity findings over medium-severity findings, as they pose a greater risk to the security of sensitive data. However, all findings should be addressed in a timely manner to maintain the security of systems and protect sensitive data.

SECURITY TOOLS

During the penetration testing phase, both automated and manual tools are used to identify and confirm vulnerabilities within the system. Automated tools, such as Nessus and OpenVAS, scan the system comprehensively to detect vulnerabilities in the operating system, applications, and network services. These tools have extensive databases of known vulnerabilities and can quickly identify known security holes in the system.

In addition to the automated tools, manual tools like Burp Suite and Nmap are used to conduct more sophisticated tests. Burp Suite is a web application security testing tool that allows penetration testers to intercept, modify, and replay web traffic to test for web application vulnerabilities. Nmap is a network exploration tool used to discover hosts and services on the network, identify open ports, and detect operating systems. These tools require higher expertise and are typically used to test for more complex security issues that automated tools may not detect.

Integrating automated and manual tools allows penetration testers to thoroughly scrutinize the system for various vulnerabilities. This helps identify potential security weaknesses that may have been missed using only automated or manual testing techniques. Overall, this approach helps to ensure that the system is as secure as possible and reduces the risk of a successful cyber attack.

EXECUTIVE

This executive summary presents the issues and recommendations from a comprehensive penetration testing engagement conducted by Halo Security on the in-scope network infrastructure and web applications. The objective of the engagement was to identify vulnerabilities and evaluate the effectiveness of security controls in place. The report thoroughly analyzes the identified vulnerabilities and recommended remediation steps to address the issues and enhance the organization's overall security posture.

SECURITY OVERVIEW

Overall, the engagement **did not identify any significant security vulnerabilities**. The tests conducted were designed to simulate attacks from external threat actors, and no vulnerabilities were found that would allow unauthorized access to sensitive information or disruption of service availability.

The engagement identified **[0] high and [0] medium-severity vulnerabilities**. An attacker could exploit these issues to gain unauthorized access to sensitive information, disrupt service availability, or launch further attacks within the network.

It is important to note that these issues are based on the information and access available during the engagement. A real-world attacker may have additional tools and techniques available to them and may be able to exploit vulnerabilities in ways that were not tested during the engagement.

The most significant issue identified during the engagement were:

- **N/A**

SUMMARY OF FINDINGS

This section summarizes the key findings relevant to the main content of the report. The summary is presented as a table, which assigns a rating to each finding based on the associated exposure level. The rating is determined by calculating each vulnerability's Common Vulnerability Scoring System (CVSS) score. The CVSS is an industry-standard method for evaluating the severity of security vulnerabilities in computer systems. It considers various factors, such as the impact of the vulnerability, the ease of exploiting it, and the availability of mitigations or workarounds.

FINDINGS	
PASS	**OWASP | A01:2021 | NO BROKEN ACCESS CONTROLS** in a web application refer to weaknesses in the control structure that allow users access levels that are not appropriate. This can happen when authentication mechanisms fail or decisions about user privileges are incorrectly configured or neglected. Broken access controls can lead to actions such as data exposure, data manipulation, account hijacking, and privilege escalation.
PASS	**OWASP | A02:2021 | NO CRYPTOGRAPHIC FAILURES** in a web application occur when cryptographic protocols, algorithms, and key management are incorrectly used. Common issues include weak encryption keys, unencrypted data being stored on the server, and inadequate validation of source material. These failures can lead to data exfiltration, sensitive information exploitation, and trust degradation.
PASS	**OWASP | A03:2021 | NO INJECTIONS FLAWS** Injection flaws in a web application are vulnerabilities that occur when untrusted data is sent to an interpreter as part of a command or query. Malicious users can exploit this to execute arbitrary commands on the server and gain access to sensitive data. Common injection flaws include SQL injection, OS command injection, and Cross-Site Scripting (XSS).
PASS	**OWASP | A04:2021 | NO INSECURE DESIGN FLAWS** Insecure design in a web application refers to design flaws or vulnerabilities that can be exploited by malicious users, which can lead to data exfiltration, exploitation of sensitive information, and degradation of trust. Common issues include the lack of input validation, improper authentication/authorization, weak access control, and insecure storage.
PASS	**OWASP | A05:2021 | NO SECURITY MISCONFIGURATION** Security misconfigurations for a web application refer to system settings errors that could expose sensitive data. This can include unpatched applications, exposed services, forgotten administration pages, debugging options left enabled, and vulnerable versions of third-party components. Allowing access to unauthenticated users or using default configurations are common examples.
PASS	**OWASP | A06:2021 | NO OUTDATED COMPONENTS** in a web application refer to outdated software libraries, frameworks, plugins, and applications running on the server. These can often introduce known vulnerabilities that malicious actors can exploit. Keeping all software components up-to-date is an effective way to limit potential security risks to the web application.
PASS	**OWASP | A07:2021 | NO AUTHENTICATION FAILURES** Authentication flaws in a web application allow unauthorized access to the system. This can be due to weak passwords, unsecured login pages, exposed session IDs and cookies, or lack of two-factor authentication. These flaws can give attackers access to sensitive data or allow them to modify content on the website.
PASS	**OWASP | A08:2021 | NO DATA INTEGRITY FAILURES** Data integrity flaws in a web application refer to the alteration or corruption of data without authorization. This can be due to weak encryption protocols, unsanitized user inputs, poor validation techniques, and lack of input validation. Implementing strong authentication measures and proper input validation techniques can help reduce the risk of data integrity flaws.

PASS	**OWASP | A09:2021 | NO MONITORING FAILURES** refer to the lack of proper logging and security monitoring of web application activities. This can lead to malicious actors taking advantage of authentication and data integrity flaws without any records of their activities being captured. To help mitigate such issues, it's important to perform comprehensive logging and regularly review logs for suspicious activity.
PASS	**OWASP | A10:2021 | NO SERVER-SIDE REQUEST FORGERY** (SSRF) is a form of attack that allows an attacker to make requests from a server to internal, external, and unsecured resources without authorization. SSRF can exploit local and remote file systems, databases, web applications, and protocols like HTTP/HTTPS, DNS, and SMTP. To protect against SSRF attacks, it's important to implement proper authentication measures and secure input validation techniques.

RECOMMENDATIONS

LONG-TERM INITIATIVES

These initiatives should be performed to maintain the overall security posture.

- Expand the scope of your annual penetration testing program to all public-facing services and applications, especially custom-built web applications with complex business logic.

COMPARATIVE ANALYSIS

Based on the issues, Halo Security compared the client's security posture to similar-sized organizations. Security classified the client's perimeter security posture as **ABOVE AVERAGE**. This classification was based on the following strengths and weaknesses:

STRENGTHS

- No vulnerabilities were identified on the networks tested.

WEAKNESSES

- N/A

SCOPE

The customer has identified certain areas or components of a product or system that they want to be evaluated as part of the testing process. These specific parts are considered "in-scope" for testing, and any areas not specified as in-scope will not be included. The customer has defined the scope of the testing by specifying which targets need to be tested.

TARGETS

1. treering.com
2. web.treering.com
3. review.treering.com
4. api.treering.com
5. api-render.treering.com
6. upload.treering.com
7. lock.treering.com
8. qa.treering.com
9. auth.treering.com
10. ops.treering.com
11. docs.treering.com
12. ws.treering.com
13. book.treering.com
14. asset.treering.com
15. full.treering.com
16. thumb.treering.com
17. static.treering.com
18. print.treering.com

FINDINGS

PASS | A01:2021 | NO BROKEN ACCESS CONTROLS OWASP

OWASP **A01:2021 – Broken Access Control**

POINTER **https://owasp.org/Top10/A01_2021-Broken_Access_Control**

DESCRIPTION:

Broken access controls in a web application refer to weaknesses or failures in the system that allows unauthorized users to access restricted areas or sensitive information. There are two main types of access control flaws: vertical and horizontal.

Vertical access control flaws are security errors that can occur when users can increase their privileges beyond what they should be able to. For example, if a guest user is suddenly granted administrator privileges instead of the intended limited access, this could be considered a vertical access control flaw. These flaws can occur when proper authentication or authorization checks are not implemented, allowing users to bypass any restrictions and gain higher-level access. These vulnerabilities can have serious consequences, resulting in data theft, fraud, and other malicious activities. Therefore, organizations should ensure all authentication and authorization measures are appropriately implemented and regularly monitored to prevent such issues.

Horizontal access control flaws refer to specific security issues that arise when users can gain access to certain information or functions within an organization that they should not have access to. This could be due to a lack of adequate authorization for the user group or simply because the user has been given too much access. Examples of horizontal access control flaws include allowing a low-level user to view or edit sensitive files belonging to a higher-level user or providing a guest with full account privileges when they only need limited access. If these flaws go unchecked, it can lead to data theft, fraud, and other malicious activities.

RESULT:

We discovered no vulnerabilities related to this OWASP category. However, it's essential to consider that the information and access available during the engagement may differ from what a real-world attacker can access. As such, they may have additional tools and techniques which could be used to exploit vulnerabilities not tested in our engagement.

PASS | A02:2021 | NO CRYPTOGRAPHIC FAILURES

OWASP

OWASP	**A02:2021 – Cryptographic Failures**
POINTER	**https://owasp.org/Top10/A02_2021-Cryptographic_Failures**

DESCRIPTION:

HTTPS certificates are digital certificates used to secure website communication. They are used to establish an encrypted connection between a website and a user's browser, ensuring that third parties cannot intercept sensitive information, such as passwords and credit card numbers, during transmission.

Weak encryption algorithms: Some certificates may use encryption algorithms no longer considered secure, such as SHA-1, which can be vulnerable to attacks.

Weak Key Strength: Certificates with weak key strength can be vulnerable to brute force attacks and compromise the website's security.

Outdated Protocols: Certificates that use outdated protocols, such as SSL, can be vulnerable to attacks and should be updated to more secure protocols like TLS.

Self-signed certificates: Self-signed certificates are not issued by a trusted certificate authority and, therefore, cannot be trusted to secure the website.

Domain name mismatch: If the certificate does not match the website's domain name, it can indicate that the website is not secure, leading to a security warning for users.

Certificate expiration: If the certificate has expired, it can no longer be trusted to secure the website.

RESULT:

We discovered no vulnerabilities related to this OWASP category. However, it's essential to consider that the information and access available during the engagement may differ from what a real-world attacker can access. As such, they may have additional tools and techniques which could be used to exploit vulnerabilities not tested in our engagement.

PASS | A03:2021 | NO INJECTIONS FLAWS OWASP

OWASP	**A03_2021-Injection**
POINTER	**https://owasp.org/Top10/A03_2021-Injection**

DESCRIPTION:

SQL injection is a type of security vulnerability in which an attacker inserts malicious code into an SQL statement, via user input, to gain unauthorized access to data stored in a database. This can be done to steal sensitive information, modify or delete data, or execute system-level commands on the host operating system. By exploiting vulnerabilities in the SQL code of a website or application, an attacker can bypass security measures and gain unauthorized access to sensitive information.

Cross-site scripting (XSS) is a web application security vulnerability that enables malicious actors to inject code into a website to gain unauthorized access or execute malicious actions. XSS attacks typically steal user information, launch phishing campaigns, and modify web page content. To prevent XSS attacks, developers should take strict measures such as sanitizing input data, using output encoding techniques, and validating data types on the server side.

Code injection is a cyber attack in which malicious code is injected into an application or system to gain unauthorized access or execute malicious actions. Code injection attacks typically steal user information, launch phishing campaigns, and modify database content. To protect against code injection, developers should take strict precautions such as validating input data, using language-specific APIs, and sanitizing output data.

RESULT:

We discovered no vulnerabilities related to this OWASP category. However, it's essential to consider that the information and access available during the engagement may differ from what a real-world attacker can access. As such, they may have additional tools and techniques which could be used to exploit vulnerabilities not tested in our engagement.

PASS | A04:2021 | NO INSECURE DESIGN FLAWS

OWASP

OWASP	**A04:2021 – Insecure Design**
POINTER	**https://owasp.org/Top10/A04_2021-Insecure_Design**

DESCRIPTION:

Business logic flaws refer to errors in the implementation of rules and processes that govern a web application's behavior, such as inconsistent data validation, broken access control, insufficient logging, unhandled exceptions and errors, inadequate data protection, incorrect calculation, and improper use of cryptographic functions, which can lead to unexpected behavior and security vulnerabilities. It's important to thoroughly test and validate the implementation of business rules and processes to ensure their correctness and security.

Upload functionality in web applications must **prevent the upload of dangerous files** such as executables, scripts, archive files, and malicious documents to maintain the security and stability of the web server and protect users. These dangerous files can run malicious code, inject malicious code into the website, contain harmful files, or contain embedded malicious scripts or macros. Web applications can implement file type restrictions and virus scanning of uploaded files to ensure that only safe and acceptable files are uploaded.

Error messages may contain sensitive information valuable to attackers in launching further attacks. Error messages can be self-generated by the source code or externally generated by the environment, such as a language interpreter. The contents of error messages can reveal information such as full pathnames, query logic, and passwords, making it easier for attackers to launch targeted attacks.

RESULT:

We discovered no vulnerabilities related to this OWASP category. However, it's essential to consider that the information and access available during the engagement may differ from what a real-world attacker can access. As such, they may have additional tools and techniques which could be used to exploit vulnerabilities not tested in our engagement.

PASS | A05:2021 | NO SECURITY MISCONFIGURATION

OWASP

OWASP **A05:2021 – Security Misconfiguration**

POINTER **https://owasp.org/Top10/A05_2021-Security_Misconfiguration**

DESCRIPTION:

Web server misconfiguration is an error or oversight in the setup or management of a web server that affects its security, functionality, or performance. Examples include incorrect file permissions, inadequate security settings, misconfigured network settings, insufficient error handling, and missing updates or patches. Misconfigurations can lead to vulnerabilities that attackers, degraded performance, or unexpected behavior for users can exploit.

Sample applications included with the application server have not been removed from the production environment, leaving the server vulnerable to security exploits. One such example is the admin console, which has known security flaws that can be exploited if the default accounts are not changed. An attacker can gain unauthorized access by logging in with the default password and taking control of the server.

Directory listing on the server has not been disabled, allowing an attacker to easily list its directories. The attacker gains access to compiled Java classes, which they decompile and reverse engineer to uncover the code. As a result, the attacker discovers a critical flaw in the application's access control.

RESULT:

We discovered no vulnerabilities related to this OWASP category. However, it's essential to consider that the information and access available during the engagement may differ from what a real-world attacker can access. As such, they may have additional tools and techniques which could be used to exploit vulnerabilities not tested in our engagement.

PASS | A06:2021 | NO OUTDATED COMPONENTS

OWASP

OWASP **A06:2021 – Vulnerable and Outdated Components**

POINTER **https://owasp.org/Top10/A06_2021-Vulnerable_and_Outdated_Components**

DESCRIPTION:

Outdated web server components include using an old version of a programming language, such as PHP 5.1 instead of the latest version of PHP 7.4; using an outdated database system such as MySQL 4, instead of the latest version of MySQL 8; and running an archaic web server platform such as IIS 6, instead of the current IIS 10. Outdated hardware components can also lead to security risks, as they may not have had the latest updates applied to them. For example, a web server still utilizing old hard drives could be vulnerable to data breaches due to its lack of encryption protection. Web servers must keep their components up-to-date to remain secure and perform optimally.

Outdated JavaScript components on web applications include using outdated libraries such as jQuery 1. x instead of the latest version of jQuery 3.5 or using an unsupported browser such as Internet Explorer 6 instead of a current version like Chrome or Firefox. Deprecated JavaScript functions such as “document.write()” can also lead to potential security risks. Web applications must stay up-to-date with the most secure and supported versions of JavaScript and its components to ensure they remain safe and secure from malicious attacks.

Common outdated network protocols on the internet today include IP version 4 (IPv4), TELNET, FTP, and HTTP 1.0. These protocols are no longer considered secure and have been supplanted by newer, more secure options such as IPv6, SSH/SFTP, HTTPS 1.2, and beyond. Outdated protocols can lead to various security issues, such as denial-of-service and man-in-the-middle attacks.

RESULT:

We discovered no vulnerabilities related to this OWASP category. However, it's essential to consider that the information and access available during the engagement may differ from what a real-world attacker can access. As such, they may have additional tools and techniques which could be used to exploit vulnerabilities not tested in our engagement.

PASS | A07:2021 | NO AUTHENTICATION FAILURES

OWASP

OWASP **A07:2021 – Identification and Authentication Failures**

POINTER **https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures**

DESCRIPTION:

The "forgot password" feature is that it often fails to properly authenticate a user's identity. This could be due to weak security measures, a lack of user verification processes such as two-factor authentication, or vulnerabilities in the software code. In addition, many forget password forms do not provide adequate recovery mechanisms for users with multiple accounts with different or long and complex passwords. As a result, users cannot reclaim access to their accounts, leading to lost data and significant downtime. Finally, some web applications may even have built-in tracking mechanisms which malicious attackers can exploit to gain control of user data and account information.

Lack of two-factor authentication: It adds an extra layer of security by requiring users to provide passwords and other information, such as a code from a physical token or their fingerprint. Without this additional step, it is easier for attackers to gain access to systems and data.

A web application's lack of brute force protection can create a serious security vulnerability. Without this type of security, malicious actors could use automated tools and scripts to rapidly guess credentials, potentially gaining access to an application or system. In addition, brute force attacks could cause serious performance problems for the application, as servers and resources would be tied up attempting to handle the malicious requests. These attacks can also result in data leakage, as passwords and information are exposed during the attack attempt. Allowing these types of attacks to occur unchecked can have serious consequences for an organization's security posture and should be guarded against robust authentication measures such as two-factor authentication or rate-limiting login attempts.

RESULT:

We discovered no vulnerabilities related to this OWASP category. However, it's essential to consider that the information and access available during the engagement may differ from what a real-world attacker can access. As such, they may have additional tools and techniques which could be used to exploit vulnerabilities not tested in our engagement.

PASS | A08:2021 | NO DATA INTEGRITY FAILURES

OWASP

OWASP **A08:2021 – Software and Data Integrity Failures**

POINTER **https://owasp.org/Top10/A08_2021-Software_and_Data_Integrity_Failures**

DESCRIPTION:

Unsigned firmware presents a major security risk to home routers, set-top boxes, device firmware, and other connected devices as it allows malicious actors to manipulate the code without any verification. This can lead to data theft, malware insertion, exploitation of vulnerabilities, and many other malicious activities. To mitigate such risks, it's important to use signed firmware that enforces authentication measures and allows for proper remediation if an attack is detected. It's also important to regularly update the firmware so that outdated versions do not remain vulnerable for longer periods.

Nation-states have been known to attack update mechanisms with malicious intent. This was the case with a notable incident where a software firm's secure build and update integrity processes were subverted, allowing for a highly targeted malicious update to be distributed to more than 18,000 organizations. Around 100 of these organizations were affected by the attack, making it one of the most far-reaching and significant breaches in history. It's important to take the necessary steps to **ensure the integrity of update mechanisms** to prevent such incidents.

Insecure deserialization can pose a major threat to applications. In the case of a React application calling a set of Spring Boot microservices, serializing the user state and passing it back and forth with each request may seem an effective way to ensure code immutability. However, an attacker can notice the "rO0" Java object signature (in base64) and exploit this vulnerability by using the Java Serial Killer tool to gain control of the application server and execute malicious code. To protect against such threats, proper security measures must be implemented to ensure data transmission integrity within each request.

RESULT:

NOTICE: The security audit, performed from the standpoint of an attacker, **was unable to determine compliance with OWASP requirements**, but no evidence of violations was observed.

PASS | A09:2021 | NO MONITORING FAILURES

OWASP

OWASP **A09:2021 – Security Logging and Monitoring Failures**

POINTER **https://owasp.org/Top10/A09_2021-Security_Logging_and_Monitoring_Failures**

DESCRIPTION:

Having **proper logging and monitoring in place** is crucial for ensuring the security of a system. Logging all login, access control, and server-side input validation failures provides valuable information for identifying potentially malicious activity. This information can be used to detect security incidents and respond to them quickly, which is crucial for minimizing the damage caused by an attack.

To effectively use log information, the logs should be generated in a format easily consumable by log management solutions. The logs should also be encoded securely to prevent attackers from tampering with or destroying them. For high-value transactions, it's important to have an audit trail that can't be tampered with or deleted. This can be achieved using append-only database tables or similar methods.

Effective monitoring and alerting should be established to ensure quick detection and response to suspicious activity. This can be done using DevSecOps teams responsible for integrating security into the development and operations process. Having dedicated teams for this purpose makes it possible to detect security incidents more quickly and respond to them more effectively.

Finally, it's important to have an **incident response and recovery plan in place**. This plan guides responding to and recovering from security incidents. A commonly used incident response and recovery plan is NIST 800-61r2 or a later version. This plan helps organizations be prepared for security incidents and respond to them organizationally and effectively.

RESULT:

NOTICE: The security audit, performed from the standpoint of an attacker, **was unable to determine compliance with OWASP requirements**, but no evidence of violations was observed.

PASS | A10:2021 | NO SERVER-SIDE REQUEST FORGERY

OWASP

OWASP **A10:2021 – Server-Side Request Forgery (SSRF)**

POINTER **https://owasp.org/Top10/A10_2021-Server-Side_Request_Forgery_%28SSRF%29**

DESCRIPTION:

At the application layer, it is important to properly secure your systems to **prevent attacks such as Server Side Request Forgery (SSRF)**. To start, sanitizing and validating all input data supplied by clients is crucial. You should also enforce the URL schema, port, and destination with a positive allow list and disable HTTP redirections. Be mindful of URL consistency to avoid attacks like DNS rebinding and "time of check, time of use" (TOCTOU) race conditions. However, do not rely on mitigating SSRF through a deny list or regular expression, as attackers have the tools and skills to bypass them.

In addition to these measures, consider not deploying other security-related services on front systems and controlling local traffic on these systems. If your frontends have dedicated and manageable user groups, it is advisable to use network encryption, such as VPNs, to provide extra protection.

Regarding SSRF attacks, **attackers can use scenarios such as port scanning internal servers, exposing sensitive data, accessing cloud service metadata storage, and compromising internal services**. For example, an attacker can map out internal networks and determine the openness of ports on internal servers by using SSRF payload connections. They can also access local files or internal services to gain sensitive information or abuse internal services to conduct further attacks, such as Remote Code Execution or Denial of Service.

RESULT:

We discovered no vulnerabilities related to this OWASP category. However, it's essential to consider that the information and access available during the engagement may differ from what a real-world attacker can access. As such, they may have additional tools and techniques which could be used to exploit vulnerabilities not tested in our engagement.