SentiSum Secure Software Policy

This Policy Document encompasses all aspects of SentiSum secure software development and must be distributed to all company employees. All company employees must read this document in its entirety. This document will be reviewed and updated by Management on an annual basis or when relevant to include newly developed security standards into the policy and distribute it all employees and contractors as applicable.

Table of Contents

Secure Software Policy

1. Overview

Web application vulnerabilities account for the largest portion of attack vectors outside of malware. It is crucial that any web application be assessed for vulnerabilities and any vulnerabilities be remediated prior to production deployment.

2. Purpose

The purpose of this policy is to define web application security assessments within **SentiSum**. Web application assessments are performed to identify potential or realized weaknesses as a result of inadvertent mis-configuration, weak authentication, insufficient error handling, sensitive information leakage, etc. Discovery and subsequent mitigation of these issues will limit the attack surface of **SentiSum** services available both internally and externally as well as satisfy compliance with any relevant policies in place.

3. Scope

This policy covers all web applications within **SentiSum** in addition to any web application security assessment requested by any individual, group or department for the purposes of maintaining the security posture, compliance, risk management, and change control of technologies in use at **SentiSum**.

All web application security activities will be performed by delegated security personnel either employed or contracted by **SentiSum**. All findings are considered confidential and are to be distributed to persons on a "need to know" basis. Distribution of any findings outside of **SentiSum** is strictly prohibited unless approved by the Chief Information Officer.

Any relationships within multi-tiered applications found during the scoping phase will be included in the assessment unless explicitly limited. Limitations and subsequent justification will be documented prior to the start of the assessment.

4. Policy

4.1 Software Security Assessments

Web applications are subjected to security assessments based on the following criteria:

a) **Vulnerability Management:**

- Risk Rankings - All the vulnerabilities would be assigned a risk ranking such as High, Medium and Low based on industry best practices such as CVSS base score.

- Vulnerability Scanning - As part of the PCI-DSS Compliance requirements, the Company will run internal and external network vulnerability scans at least quarterly and after any significant change in the network (such as new system component installations, changes in network topology, firewall rule modifications, product upgrades).
- Quarterly Internal Vulnerability Scans - Quarterly internal vulnerability scans must be performed by the Company by internal staff or a 3rd party vendor and the scan process has to include that rescans will be done until passing results are obtained, or all High vulnerabilities as defined in PCI DSS Requirement 6.2 are resolved.
- Quarterly External Vulnerability Scans - Quarterly external vulnerability scans must be performed by an Approved Scanning Vendor (ASV) qualified by PCI SSC. Scans conducted after network changes may be performed by the Company's internal staff. The scan process should include re-scans until passing results are obtained.

b) **Patch Management:**

- Updated Security Patches - All Workstations, servers, software, system components etc. must have up-to-date system security patches installed to protect the asset from known vulnerabilities.
- Automatic Updates - All systems must have automatic software updates enabled for system patches released from their respective vendors. Security patches have to be installed within one month of release from the respective vendor and have to follow the process in accordance with change control process.
- Exceptions - Any exceptions to this process have to be documented.

c) **Secure SDLC:** The Secure Application Development policy is a plan of action to guide developers' decisions and actions during the software development lifecycle (SDLC) to ensure software security. This policy aims to be language and platform independent so that it is applicable across all software development projects.

- **Secure Coding:**
  - Development - Development of code shall be checked and validated with the most current versions of the Coding Standards for Secure Application Development. All application developers shall verify that their code is in compliance with the most recent and approved coding standards and guidelines.
  - Code Review - Only validated code shall be implemented into the production environment. A review and validation ensures that code exhibits

fundamental security properties to include correctness, predictability, and attack tolerance.

- Application Code Developers shall:
  - Ensure code meets the level of confidence that software is free from exploitable code vulnerabilities, regardless of whether they are already designed into the software or inserted later in its life cycle.
  - Ensure code provides predictable execution or justifiable confidence and that the software, when executed, will provide security functionality as intended.
  - Coding techniques must address injection flaws particularly SQL injection, buffer overflow vulnerabilities, cross site scripting vulnerabilities, improper access control (insecure direct object reference, failure to restrict URL access, directory traversal etc.,), cross site request forgery (CSRF), broken authentication and session management.
  - Never trust incoming data to the system, apply data validation checks.
  - Never rely on the client to store sensitive data no matter how trivial.
  - Disable Error messages that return any information to the user.
  - Use object inheritance, encapsulation, and polymorphism wherever possible.
  - Use environment variables prudently and always check boundaries and buffers.
  - Application must validate input to ensure it is well-formed and meaningful.

- **Application Testing:**
  - Methodology - Application tests must follow the OSSTMM methodology. Tests must be conducted at network, system and application level and must ensure that at least identifies any vulnerabilities documented by OWASP and SANS, as well as those identified in the PCI DSS standard v3.2:
    - Injections: Code, SQL, OS commands, LDAP , XPath , etc.
    - Buffer overflows.
    - Insecure storage of cryptographic keys.
    - Insecure Communications.
    - Improper error handling.
    - Cross - site scripting (XSS).
    - Control of inappropriate access.
    - Cross - site request forgery (CSRF).
    - Broken authentication and incorrectly session management.
    - Any other vulnerability considered High Risk by the organization.

- Documentation - All findings or vulnerabilities identified during the tests carried out will be generated and documented with sufficient evidence to prove the existence of the same. The format of the evidence can be variable in each case, screen capture, raw output of security tools (HTTP headers), photographs, paper documents, etc.
- As a result of the tests performed, generate a document containing at least the following sections:
  - Introduction
  - Executive Summary
  - Methodology
  - Identified vulnerabilities
  - Recommendations for correcting vulnerabilities Conclusions
  - Evidence

d) **Change Management:**

- Process - Changes to information resources shall be managed and executed according to a formal change control process. The process will ensure that changes proposed are reviewed, authorized, tested, implemented, and released in a controlled manner; and that the status of each proposed change is monitored.
- Documentation - The change control process shall be formally defined and documented. A change control process shall be in place to control changes critical to company information resources (such as hardware, software, system documentation and operating procedures). This documented process shall include management responsibilities and procedures. Wherever practicable, operational and application change control procedures should be integrated.
- Audit - All change requests shall be logged whether approved or rejected on a standardized and central system. The approval of all change requests and the results thereof shall be documented. A documented audit trail, maintained at a Business Unit Level, containing relevant information shall be maintained at all times. This should include change request documentation, change authorization and the outcome of the change. No single person should be able to implement changes to the production information systems without the approval of other authorized personnel.
- Risk Assessment - A risk assessment shall be performed for all changes and dependent on the outcome, an impact assessment should be performed.
- Impact Assessment - The impact assessment shall include the potential effect on other information resources and potential cost implications. The impact assessment should, where applicable consider compliance with legislative requirements and standards.

- Prioritization - All change requests shall be prioritized in terms of benefits, urgency, effort required and potential impact on operations.
- Testing - Changes shall be tested in an isolated, controlled, and representative environment (where such an environment is feasible) prior to implementation to minimize the effect on the relevant business process, to assess its impact on operations and security and to verify that only intended and approved changes were made.
- Version Control - Any software change and/or update shall be controlled with version control. Older versions shall be retained in accordance with corporate retention and storage management policies.
- Approval - All changes shall be approved prior to implementation. Approval of changes shall be based on formal acceptance criteria i.e. the change request was done by an authorized user, the impact assessment was performed and proposed changes were tested.
- Notification - All users, significantly affected by a change, shall be notified of the change. The user representative shall sign-off on the change. Users shall be required to make submissions and comment prior to the acceptance of the change.
- Production Deployment - Implementation will only be undertaken after appropriate testing and approval by the stakeholders. All major changes shall be treated as new system implementation and shall be established as a project. Major changes will be classified according to effort required to develop and implement said changes.
- Recovery Process - Procedures for aborting and recovering from unsuccessful changes shall be documented. Should the outcome of a change be different to the expected result (as identified in the testing of the change), procedures and responsibilities shall be noted for the recovery and continuity of the affected areas. Fallback procedures will be in place to ensure systems can revert back to what they were prior to implementation of changes.
- Data Retention & Archival - Information resources documentation shall be updated on the completion of each change and old documentation shall be archived or disposed of as per the documentation and data retention policies.
- Hot Fixes - Specific procedures to ensure the proper control, authorization, and documentation of emergency changes shall be in place. Specific parameters will be defined as a standard for classifying changes as Emergency changes.
- Post Production Monitoring - All changes will be monitored once they have been rolled-out to the production environment. Deviations from design specifications and test results will be documented and escalated to the solution

owner for ratification.

e) **Awareness Testing:**

- Meetings - Review handling procedures for sensitive information and hold periodic security awareness meetings to incorporate these procedures into day-to-day practice.
- Acknowledgement - Distribute this security policy document to all company employees to read. It is required that all employees confirm that they understand the content of this security policy document by signing an acknowledgement form.
- Security Clearance - All employees that handle sensitive information will undergo background checks (such as criminal and credit record checks, within the limits of the local law) before they commence their employment.
- Contractual binding for third party access - All third parties with access to credit card account numbers are contractually obligated to comply with card association security standards (**PCI/DSS**).
- Policy Audit - Company security policies must be reviewed annually and updated as needed.

f) **Protect Stored Data**:

- Secure Storage - All sensitive cardholder data stored and handled by the Company and its employees must be securely protected against unauthorized use at all times. Any sensitive card data that is no longer required by the Company for business reasons must be discarded in a secure and irrecoverable manner.
- Mask PAN - If there is no specific need to see the full PAN (Permanent Account Number), it has to be masked when displayed.
- No Sharing of PAN’s - PAN's which are not protected as stated above should not be sent to the outside network via end user messaging technologies like chats, ICQ messenger etc.
- It is strictly prohibited to store:
  - The contents of the payment card magnetic stripe (track data) on any media whatsoever.
  - The CVV/CVC (the 3 or 4 digit number on the signature panel on the reverse of the payment card) on any media whatsoever.
  - The PIN or the encrypted PIN Block under any circumstance.

g) **Protect Data in Transit:**

- No clear text transmission - Cardholder data (PAN, track data etc.) must never be sent over the Internet via email, instant chat or any other end user technologies.

- Use Encryption for sending card data over wire - If there is a business justification to send cardholder data via email or via the Internet or any other modes then it should be done after authorization and by using a strong encryption mechanism (i.e. – AES encryption, RSA, PGP encryption, TLS 1.2, IPSEC).
- End to End tracking - The transportation of media containing sensitive cardholder data to another location must be authorized by management, logged and inventoried before leaving the premises. Only secure courier services may be used for the transportation of such media. The status of the shipment should be monitored until it has been delivered to its new location.

h) **Disposal of Stored Data:**

- All data must be securely disposed of when no longer required by the Company, regardless of the media or application type on which it is stored.
- An automatic process must exist to permanently delete on-line data, when no longer required.
- All hard copies of cardholder data must be manually destroyed when no longer required for valid and justified business reasons. A quarterly process must be in place to confirm that all non-electronic cardholder data has been appropriately disposed of in a timely manner.
- All cardholder data on the electronic media must be rendered unrecoverable when deleted e.g. through degaussing or electronically wiped using military grade secure deletion processes or the physical destruction of the media.
- If secure wipe programs are used, the process must define the industry accepted standards followed for secure deletion.
- All cardholder information awaiting destruction must be held in lockable storage containers clearly marked “To Be Shredded” and access to these containers must be restricted.

i) **Network Security & Monitoring:**

- Audit Network Security - Firewalls must be implemented at each Internet connection and any demilitarized zone and the internal company network. A network diagram detailing all the inbound and outbound connections must be maintained and reviewed every 6 months.
- Secure Perimeter - Firewall and router configurations must restrict connections between untrusted networks and any systems in the cardholder data environment.
- Define Rules for Data Traffic - All inbound and outbound traffic must be restricted to that which is required for the cardholder data environment. All inbound network traffic is blocked by default, unless explicitly allowed and the

restrictions have to be documented. All outbound traffic has to be authorized by management (i.e. what are the whitelisted category of sites that can be visited by the employees) and the restrictions have to be documented.

- Secure Wireless (WiFi) Perimeter - Establish firewalls between any wireless networks and the cardholder data environment. Quarantine wireless users into a DMZ, where they will be authenticated and firewalled as if they were coming in from the Internet.
- Review Topology - Disclosure of private IP addresses to external entities must be authorized. A topology of the firewall environment has to be documented and has to be updated in accordance to the changes in the network. The firewall rules will be reviewed on a six months basis to ensure validity and the firewall has to have clean up rule at the bottom of the rule base.
- No Loose connections - No direct connections from Internet to cardholder data environment will be permitted. All traffic has to traverse through a firewall.

4.2 Software Security Risks

All security issues that are discovered during testing must be mitigated based upon the following risk levels. The Risk Levels are based on the OWASP Risk Rating Methodology. Remediation validation testing will be required to validate fix and/or mitigation strategies for any discovered issues of Medium risk level or greater.

a) **High:** Any high risk issue must be fixed immediately or other mitigation strategies must be put in place to limit exposure before deployment. Applications with high risk issues are subject to being taken off-line or denied release into the live environment.

b) **Medium:** Medium risk issues should be reviewed to determine what is required to mitigate and scheduled accordingly. Applications with medium risk issues may be taken off-line or denied release into the live environment based on the number of issues and if multiple issues increase the risk to an unacceptable level. Issues should be fixed in a patch/point release unless other mitigation strategies will limit exposure.

c) **Low:** Issue should be reviewed to determine what is required to correct the issue and scheduled accordingly.

4.3 Software Security Assessments Levels

a) **Full:** A full assessment is comprised of tests for all known web application vulnerabilities using both automated and manual tools based on the OWASP Testing Guide. A full assessment will use manual penetration testing techniques to validate discovered vulnerabilities to determine the overall risk of any and all discovered.

b) **Quick:** A quick assessment will consist of a (typically) automated scan of an application for the OWASP Top Ten web application security risks at a minimum.

c) **Targeted:** A targeted assessment is performed to verify vulnerability remediation changes or new application functionality.

4.4 Software Security Assessment Tools

The current approved web application security assessment tools that will be used for the testing are:

a) **KALI Linux:** Kali Linux is a Debian-based Linux distribution aimed at advanced Penetration Testing and Security Auditing. Kali contains several hundred tools which are geared towards various information security tasks, such as Penetration Testing, Security research, Computer Forensics and Reverse Engineering.

b) **OWASP ZAP:** It is an open source web application security scanner. When used as a proxy server it allows the user to manipulate all of the traffic that passes through it, including traffic using https. It can also run in a 'daemon' mode which is then controlled via a REST Application programming interface. Some of the built in features include: Intercepting proxy server, Traditional and AJAX Web crawlers, Automated scanner, Passive scanner, Forced browsing, Fuzzer, WebSocket support, Scripting languages, and Plug-n-Hack support. It has a plugin-based architecture and an online 'marketplace' which allows new or updated features to be added.

c) **NMAP:** Nmap ("Network Mapper") is a free and open source (license) utility for network discovery and security auditing. **SentiSum's** IT security personal can use it for tasks such as network inventory, managing service upgrade schedules, and monitoring host or service uptime. Nmap uses raw IP packets in novel ways to determine what hosts are available on the network, what services (application name and version) those hosts are offering, what operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, and dozens of other characteristics. It was designed to rapidly scan large networks, but works fine against single hosts. In addition to the classic command-line Nmap executable, the Nmap suite includes an advanced GUI and results viewer (Zenmap), a flexible data transfer, redirection, and debugging tool (Ncat), a utility for comparing scan results (Ndiff), and a packet generation and response analysis tool (Nping).

d) **OWASP WebGoat:** WebGoat is a deliberately insecure web application maintained by OWASP designed to teach or practice web application security lessons. You can Application developers at **SentiSum** can use it to practice vulnerabilities such as Cross-site scripting, SQL Injections, Buffer Overflow. CSRF, Access Control etc.

e) **Static Code Analysis Tools:** Static Code Analysis (also known as Source Code Analysis) is usually performed as part of a Code Review (also known as white-box testing) and is carried out at the Implementation phase of a Security Development

Lifecycle (SDL). Static Code Analysis commonly refers to the running of Static Code Analysis tools that attempt to highlight possible vulnerabilities within 'static' (non-running) source code by using techniques such as Taint Analysis and Data Flow Analysis.

Other tools and/or techniques may be used depending upon what is found in the default assessment and the need to determine validity and risk are subject to the discretion of the Security Engineering team.

5. Policy Compliance

5.1 Compliance Measurement

The **SentiSum** Infosec team will verify compliance to this policy through various methods, including but not limited to, periodic walk-through, video monitoring, business tool reports, internal and external audits, and feedback to the policy owner.

5.2 Exceptions

Any exception to the policy must be approved by the **SentiSum** Infosec team in the advance.

5.3 Non-Compliance

An employee found to have violated this policy may be subjected to the disciplinary action, up to and including termination of employment. Web application assessments are a requirement of the change control process and are required to adhere to this policy unless found to be exempt. All application releases must pass through the change control process. Any web applications that do not adhere to this policy may be taken offline until such time that a formal assessment can be performed at the discretion of the Chief Information Officer.

6. Related Standards, Policies, and Processes

OWASP Top Ten Project

OWASP Testing Guide

OWASP Risk Rating Methodology

Strong Cryptography - NIST

Secure Protocols – NIST

Top 10 Secure Coding Practices - CERT

7. Web Application Security Best Practices

7.1 Create a Web Application Security Blueprint

You can't hope to stay on top of web application security best practices without having a plan in place for doing so. All too often, companies take a disorganized approach to the situation and end up accomplishing next to nothing. Sit down with your IT security team to develop a detailed, actionable web application security plan. It should outline your organization's goals.

For example, perhaps you want to enhance your overall compliance, or maybe you need to protect your brand more carefully. It should also prioritize which applications should be secured first and how they will be tested. Whether you choose to do so manually, through a cloud solution, through software that you have on site, through a managed service provider or through some other means.

A fairly detailed 6-step sample web application security checklist can be found here.

Additionally, if your organization is large enough, your blueprint should name the individuals within the organization who should be involved in maintaining web application security best practices on an ongoing basis. Finally, be sure to factor in the costs that your organization will incur by engaging in these activities.

7.2 Perform an Inventory of Your Web Applications

Organized as though you think your company may be, you probably don't have a very clear idea about which applications it relies on, on a daily basis. In fact, most organizations have many rogue applications running at any given time and never notice them until something goes wrong. You can't hope to maintain effective web application security without knowing **precisely which applications your company uses**.

How many are there? Where are they located? Performing such an inventory can be a big undertaking, and it is likely to take some time to complete. While performing it, make a note of the purpose of each application. Chances are that when it is all said and done, there will be many applications that are either redundant or completely pointless. This inventory will come in handy for the steps that are to follow too, so take your time and make sure to get every single application.

7.3 Prioritize Your Web Applications

After completing the inventory of your existing web applications, sorting them in order of priority is the logical next step. You may doubt it now, but your list is likely to be very long. Without prioritizing which applications to focus on first, you will struggle to make any meaningful progress.

Sort the applications into three categories: Critical, Serious, and Normal.

Critical applications are primarily those that are externally facing and contain customer information. These are the applications that should be managed first, as they are the most likely to be targeted and exploited by the hackers. Serious applications may be internal or external and may contain some sensitive information. Normal applications have far less exposure, but they should be included in tests down the road.

By categorizing your applications like this, you can reserve extensive testing for critical ones and use less intensive testing for less critical ones. This allows you to make the most effective use of your company's resources and will help you achieve progress more quickly

7.3 Run Applications Using the Fewest Privileges Possible

Even after all of your web applications have been assessed, tested and purged of the most problematic vulnerabilities, you aren't in the clear. Every web application has specific privileges on both local and remote computers. These privileges can and should be adjusted to enhance security.

Always use the least permissive settings for all web applications. This means that applications should be buttoned down. Only highly authorized people should be able to make system changes and the like. You might consider including this in your initial assessment. Otherwise, you will have to go back down the entire list adjusting settings again. For the vast majority of applications, only system administrators need complete access. Most other users can accomplish what they need with minimally permissive settings.

In the unlikely event that privileges are adjusted incorrectly for an application and certain users can't access the features that they need, the problem can be handled when it occurs. It is far better to be too restrictive in this situation than to be too permissive.

7.4 Use Cookies Securely

Another area that many organizations don't think about when addressing web application security best practices is the use of cookies. Cookies are incredibly convenient for businesses and users alike. They allow users to be remembered by sites that they visit so that **future visits are faster and, in many cases, more personalized**. However, the cookies can also be manipulated by hackers to gain access to the protected areas.

While you certainly don't have to stop using cookies – indeed, to do so would be a major step backward in many ways – you should adjust the settings for yours to minimize the risk of attacks.

a) First, **never use cookies to store highly sensitive or critical information**. For example, don't use cookies to remember users' passwords, as this makes it incredibly easy for hackers to gain unauthorized access.
b) You should also be **conservative when setting expiration dates** for cookies. Sure, it's nice to know that a cookie will remain valid for a user for months on end, but the reality is that each one presents a security risk.
c) Finally, consider **encrypting the information** that is stored in the cookies that you use.

7.5 Advance Web Security Suggestions

a) Implement HTTPS and redirect all HTTP traffic to HTTPS.

b) Help prevent cross-site scripting attacks by implementing the x-xss-protection security header.
c) Implement a content security policy.
d) Help prevent man in the middle attacks by enabling public key pins.
e) Apply sub resource integrity to your resource's <script> or <link> elements.
f) Use an updated version of TLS. To learn more, read TLS 1.2 vs TLS 1.1 article and avoid using SSL completely.
g) This goes without saying, use **strong passwords** that employ a combination of lowercase and uppercase letters, numbers, special symbols, etc.

7.6 Conduct Web Application Security Awareness Training

If you run a company, chances are that only certain people within your organization have a decent grasp of the importance of web application security and how it works. The majority of users have only the most basic understanding of the issue, and this can make them careless. This is also problematic because uneducated users fail to identify security risks.

By educating employees, they will more readily spot vulnerabilities themselves. In essence, bringing everyone up to speed about web application security is a terrific way to get everyone in on the act of finding and eliminating vulnerabilities. With this in mind, consider bringing in a web application security specialist to conduct awareness training for your employees.

By bringing everyone on board and making sure that they know what to do if they encounter vulnerability or other issue, you can strengthen your overall web application security process and maintain the best possible web application security best practices.

8. Definitions and Terms

None.