8. Configuration Management Policy

Ceresti uses standard configuration management practices to manage all changes to Production systems.

8.1 Applicable Standards

8.1.1 Applicable Standards from the HITRUST Common Security Framework

- 05 - Configuration Management

8.1.2 Applicable Standards from the HIPAA Security Rule

- 164.310(a)(2)(iii) Access Control & Validation Procedures

8.2 Configuration Management Policies

1. No systems are deployed into Ceresti environments without approval of the Ceresti CTO.
2. All changes to production systems, network devices, and firewalls are approved by the Ceresti CTO before they are implemented to assure they comply with business and security requirements.
3. All changes to production systems are tested before they are implemented in production.
4. Implementation of approved changes are only performed by authorized personnel.
5. All frontend functionality (developer dashboards and portals) is separated from backend (database and app servers) systems by being deployed on separate containers.
6. All software and systems are tested using unit tests and end to end tests.
7. All committed code is reviewed using pull requests to assure software code quality and proactively detect potential security issues in development.
8. Ceresti utilizes development and staging environments that mirror production to assure proper function.
9. All formal change requests require unique ID and authentication and are managed using YouTrack - our software development ticketing system.

8.3 Provisioning Production Systems

1. Before provisioning any systems, dev team members must file a request in the

Ceresti Quality Management System.
    - Quality Management System access requires authenticated users.
    - The CTO grants access to the Quality Management System following the procedures covered in the Access Establishment and Modification section.
2. The CTO, or an authorized delegate of the CTO, must approve the provisioning request before any new system can be provisioned.
3. Once provisioning has been approved, the dev team member must configure the new system according to the standard baseline chosen for the system's role.
4. If the system will be used to house production data (ePHI), the system must be provisioned within the secure Aptible infrastructure including the use of Postgres and / or Redis on a secure secondary network. All data records must be encrypted in transit and at rest.
5. Once the system has been provisioned, the dev team member must contact the Security Officer to inspect the new system.
6. The new system may be rotated into production once the CTO verifies all the provisioning steps listed above have been correctly followed and has marked the Issue with the `Approved` state.

8.4 Changing Existing Systems

1. Subsequent changes to already-provisioned systems are handled on a case by case basis by initiating a change request through the Quality Management System.
2. The change request will be reviewed by the Security Officer to ensure it complies will all Ceresti Policies.
3. Before rolling out the change to production, the change must be tested and validated in the Staging environment. The results of the testing must be reviewed and approved by the CTO.
4. Once the request has been approved by the CTO, the dev team member may roll out the change into production environments.

8.6 Software Development Procedures

1. All development uses feature branches based on the main branch used for the current release. Any changes required for a new feature or defect fix are committed to that feature branch.
    - These changes must be covered under 1) a unit test where possible, or 2) integration tests.
    - Integration tests are required if unit tests cannot reliably exercise all facets of the change.
2. Developers are strongly encouraged to follow the commit message conventions

suggested by GitHub.
    - Commit messages should be wrapped to 72 characters.
    - Commit messages should be written in the present tense. This convention matches up with commit messages generated by commands like git merge and git revert.
3. Once the feature and corresponding tests are complete, a pull request will be created. The pull request should indicate which feature or defect is being addressed and should provide a high-level description of the changes made.
4. Code reviews are performed as part of the pull request procedure. Once a change is ready for review, the author(s) will notify other engineers using an appropriate mechanism, typically via a `@channel` message in Slack.
    - Other engineers will review the changes, using the guidelines above.
    - Engineers should note all potential issues with the code; it is the responsibility of the author(s) to address those issues or explain why they are not applicable.
5. If the feature or defect interacts with ePHI, or controls access to data potentially containing ePHI, the code changes must be reviewed by Ceresti's Security Officer before the feature is marked as complete.
    - The Security Officer will provide a security analysis of features to ensure they satisfy Ceresti's compliance and security commitments.
    - This review must include a security analysis for potential vulnerabilities such as those listed in the OWASP Top 10 or the CWE top 25.
    - This review must also verify that any actions performed by authenticated users will generate appropriate audit log entries.

8.7 Software Release Procedures

1. Software releases are treated as changes to existing systems and thus follow the procedure described in §8.4.