Chameleon System Security Plan

CHAMELEON

FOR OUR SMARTER WORLD

LEON NETTO

Purpose

The Chameleon System Security plan has been aligned with international security frameworks to ensure the continuous security of the web application while adhering to compliance measures. To determine the robustness of the security protocols in place, the CI/CD Pipeline and Deployment documentation was reviewed, and a system security controls specification was developed. The specification has been developed in line with the NIST and ISO security frameworks and ACSC guidelines.

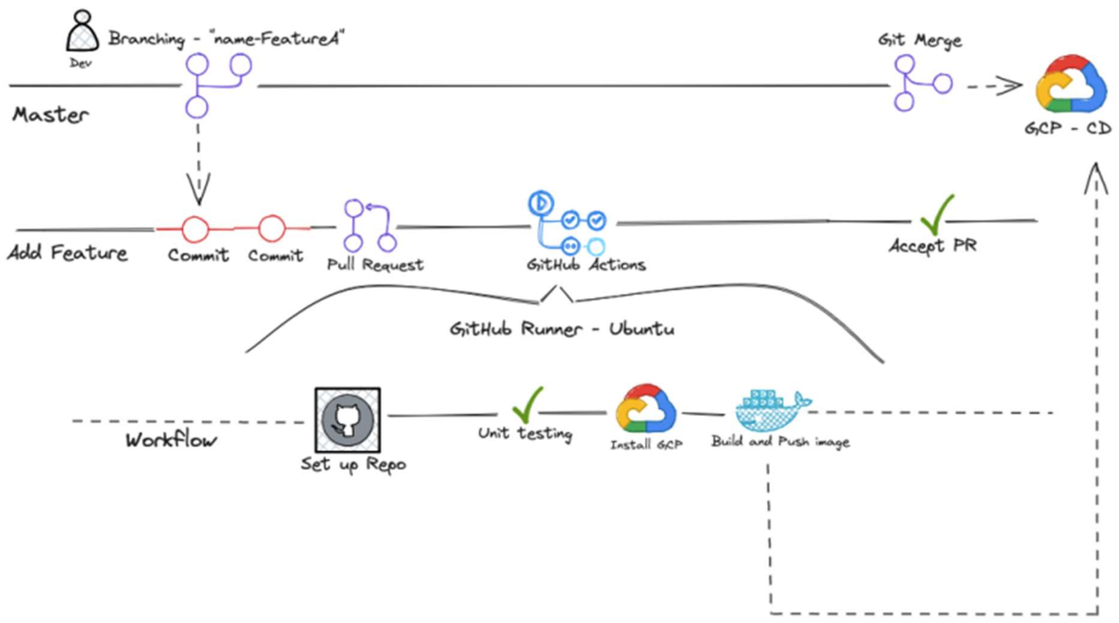
The primary objectives of the security audit encompasses a thorough examination of various domains of the security infrastructure. This includes:

1. Ensure the Chameleon web application's security controls align with established industry standards and best practices.
2. Methods and tools that will be used to identify vulnerabilities, and measures in place to address identified vulnerabilities and implement necessary patches.
3. Employ authentication mechanisms for the Chameleon web application to verify user identities and ensure secure access to sensitive areas of the application.
4. Utilisation of secure coding practices and frameworks in the development process is conducted which includes reviewing coding standards and practices.
5. Implementing access controls within the development pipeline by analysing the mechanisms in place to regulate and restrict access to critical development stages and preventing unauthorised modifications.
6. Developers referencing official secure open community guidelines in the application development process.
7. Measures in place to safeguard sensitive data and ensure the secure transmission of information across the network.

Scope

The scope of the system security plan is limited to the Chameleon website that has been planned for deployment to the Google Cloud Platform. The security controls have been developed in line with national and international standards to address the development and implementation of the cloud-based application. The CI/CD Pipeline and Deployment documentation (documented by Mukund Srinivasan) and the Chameleon Website GitHub repository has also been reviewed and the security controls outlined in the Security Control Specification will be aligned to address the risks identified.

CI/CD Pipeline



Reference: CI/CD Pipeline and Deployment documentation

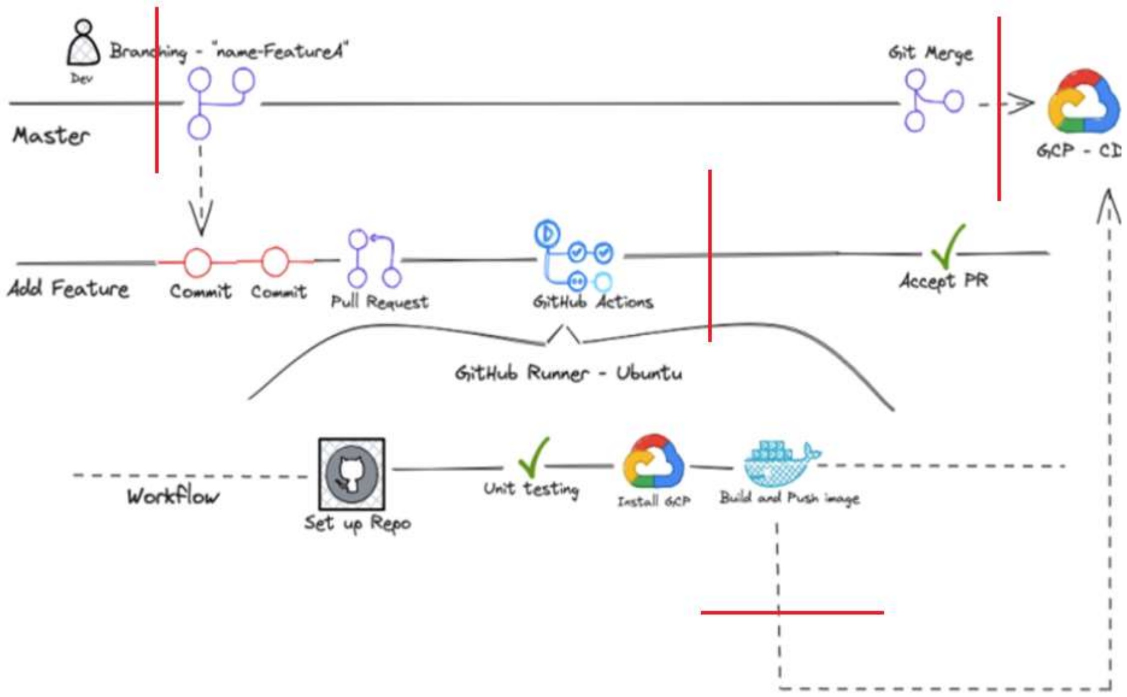
Security Control Specification

Access Controls
Unauthorised staff must not be able to modify or access the web application in any way.
Unauthorised staff must not be able to modify or access the application's code or repository in any way.
Secure role-based access controls should be applied to the GitHub Actions, Google Cloud Platform (GCP) and the other tools in the pipeline.
API keys or any authentication credentials must be stored and managed in a securely.
MFA must be applied and enforced to login to the application
Secure Design
Development, testing and production environments must be segregated.
Secure-by-design principles are used.
Validation and/or sanitisation must be performed on all input handled by the web application.
Content-Security-Policy, HSTS and X-Frame-Options must be implemented via security policy in the response headers.
Vulnerability and Patch Management
Static application security testing (SAST) and dynamic application security testing (DAST) must be applied to check for vulnerabilities in the web application.
Docker base images must be updated regularly to ensure the most up-to-date security patches are applied.
Vulnerability scans should be performed on Docker images to check for vulnerabilities.
Vulnerability scans should be performed for Dependencies.
Network Security
HTTPS must be used to deliver the web application content and a digital certification must issued by a Trusted Certificate Authority.
Most current TLS version must be used.
A Web Application Firewall should be used.
Log Management
The web application must have the following events logged: authentication, attempted access that is denied, crashes and error messages.
Logged events must be stored in a centrally.

Access Controls

Trusted Boundaries

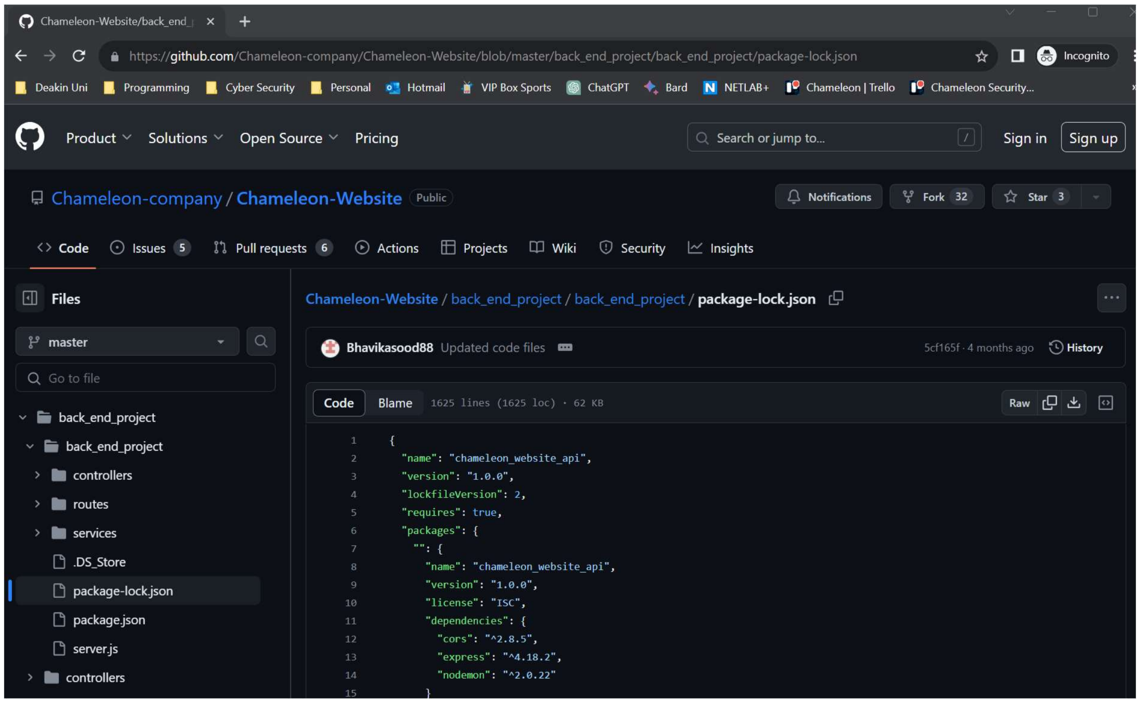
Security boundaries are services within a workflow that exchange data or trust. As a result, the permissions and access within these services should be controlled to ensure the confidentiality, availability and integrity of data as it flows from one service, or trusted boundary, to another within the CI/CD pipeline.



Security Trusted Boundaries depicted in red

Within the CI/CD pipeline the following trusted boundaries were identified and access controls should be implemented:

1. **GitHub Repository** - Access to the repository should be restricted to authorised users and permissions and authentication controls should be implemented for only authorised users to create pull requests, create branches and access and make changes to the application code. It is recommended that the Chameleon-Website repository be changed to private as sensitive information such as the back end and front end architecture and build of the application can be viewed by unauthorised personnels.



```
{
  "name": "chameleon_website_api",
  "version": "1.0.0",
  "lockfileVersion": 2,
  "requires": true,
  "packages": {
    "": {
      "name": "chameleon_website_api",
      "version": "1.0.0",
      "license": "ISC",
      "dependencies": {
        "cors": "^2.8.5",
        "express": "^4.18.2",
        "nodemon": "^2.0.22"
      }
```

2. **GitHub Actions -** GitHub actions tends to store sensitive information such as access tokens, API keys and other secrets such as login credentials. It is highly recommended that GitHub Action Secrets is used for storing and managing secrets securely. For more information on implementing this visit: https://docs.github.com/en/rest/actions/secrets?apiVersion=2022-11-28

3. **Docker -** Dependencies and various command executions are used when building docker images. It is important that that the Docker build image is secure to only the authorised staff and the Docker image does not contain any vulnerabilities.

4. **Google Cloud Platform -** When the images have been built in Docker and need to be pushed to the GCP, this should be secured with strong credentials and restricted to authorised personnel. The principal of least privilege should be applied to ensure only selected authorised staff can make the necessary changes.

Multifactor Authentication

Multi factor authentication must also be applied and enforced to the Chameleon Website. The current authorisation tool is Firebase and Google MFA can be setup on each individual account, however this is not enforced and can be challenging to manage for multiple users.

The recommendation is to use an Identity and Access Management solution such as Keycloak. Keycloak is an opensource IAM solution that provides secure authentication, authorisation and user management services for various applications and APIs. It provides MFA and user profiles can be created with the appropriate access levels.

https://www.keycloak.org/

Secure Design

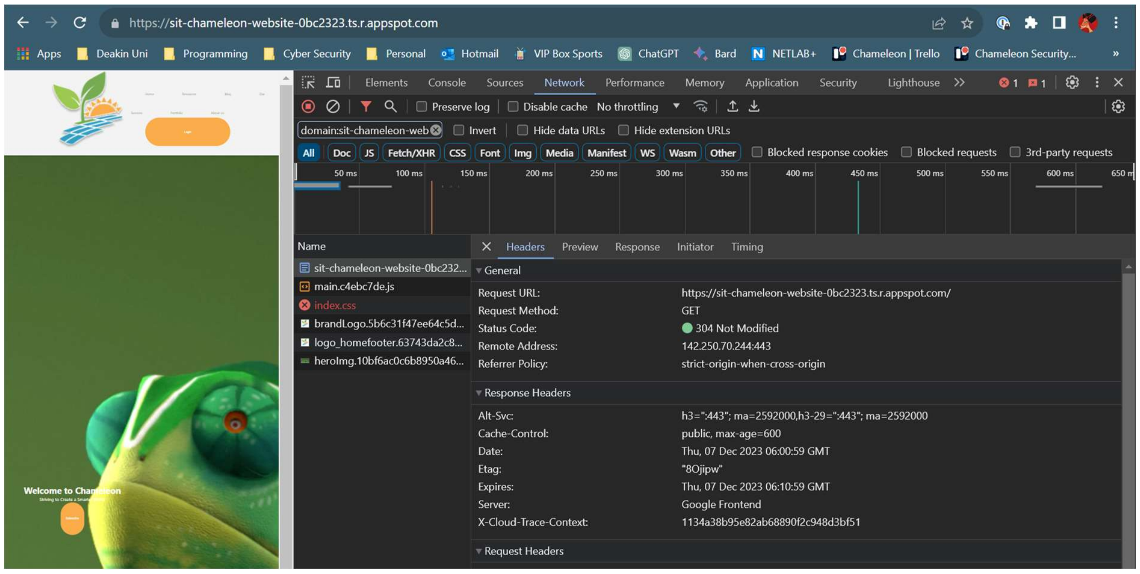
Secure design principles throughout the development process are required to ensure the Chameleon website remains resilient against threats and vulnerabilities. The multi layered security approach focuses on the application maintaining a secure posture at multi levels which includes environments the application is deployed, adhering to recommended design protocols for a secure architecture and implementing specific controls to reduce vulnerabilities.

The following design practices will assist Chameleon developers to enhance the security of the Chameleon website:

1. **Segregate environments -** The development, testing and production environments must be segregated. This is crucial to prevent security risks and issues. This practice has not been implemented within the Chameleon Website team. It is recommended that separate environments are used within each stage of the development lifecycle, and that appropriate controls are set for each environment. For example, the sandbox or test environment should be securely isolated from the production environment and data.
2. **Input validation and Sanitization -** Proper input and validation is vital to prevent Cross Site Scripting and SQL injection Attacks. Inputs should be first checked if is correct and

then sanitised accordingly to discard any malicious data. Proper validation checks should be applied, and malicious data should be encoded or escaped as part of the sanitisation. It is recommended that developer refer to the OWASP Top Ten Proactive Security Controls. Developers should refer to the secure-by-design techniques and have them incorporated from the start of the development process, ensuring that security is at the forefront of the development lifecycle.

3. **Content-Security-Policy (CSP), HTTP Strict Transport Security (HSTS) and X-Frame-Options -** The CSP, HSTS and X-Frame-Options are security headers in the HTTP response headers. These specific HTTP response headers provide additional security for the Chameleon Web Application. The CSP header mitigates Cross Site Scripting attacks, X-Frame-Options mitigates clickjacking attacks and HSTS ensures a secure connection is made. It is recommended the Chameleon Website application and webservers enforce these security headers. It is worth nothing that the Chameleon test website does not have the headers implemented.



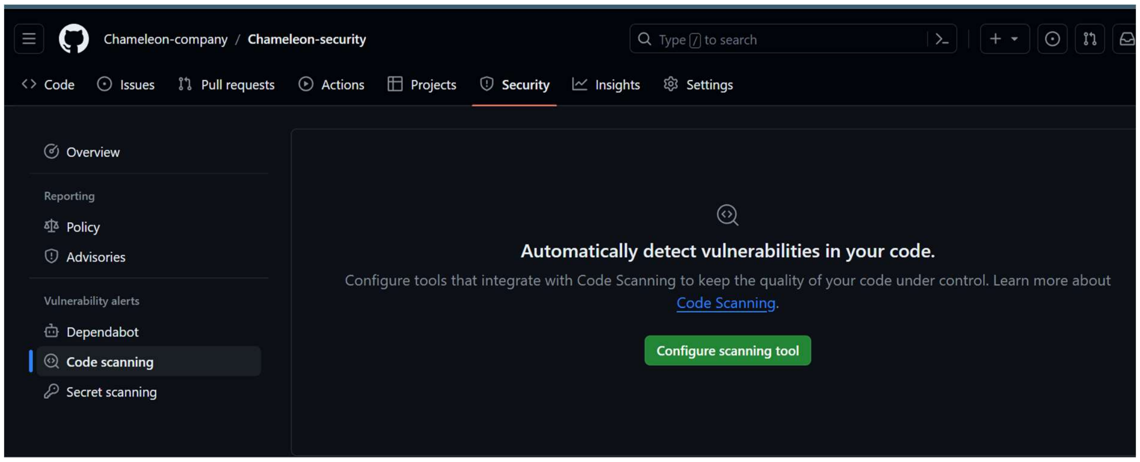
```
▾ General
Request URL:            https://sit-chameleon-website-0bc2323.ts.r.appspot.com/
Request Method:         GET
Status Code:            ● 304 Not Modified
Remote Address:         142.250.70.244:443
Referrer Policy:        strict-origin-when-cross-origin
▾ Response Headers
Alt-Svc:                h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
Cache-Control:          public, max-age=600
Date:                   Thu, 07 Dec 2023 06:00:59 GMT
Etag:                   "8Ojipw"
Expires:                Thu, 07 Dec 2023 06:10:59 GMT
Server:                 Google Frontend
X-Cloud-Trace-Context:  1134a38b95e82ab68890f2c948d3bf51
▾ Request Headers
```

Vulnerability and Patch Management

Vulnerability and patch management is crucial in fortifying the Chameleon web application to maintain a resilient and secure platform. It assists in identifying weaknesses and security that could be exploited by malicious actors. Timely vulnerability and patch management reduces these risks, making the system less susceptible to cyberattacks. It also ensures compliance measures are addressed in line with industry regulations and data protection standards.

The following recommendations have been provided to ensure that vulnerabilities in the Chameleon Website and the software services within the CI/CD pipeline are identified and addressed before being exploited:

1. **SAST Testing -** Implement a Static Application Security Testing (SAST) tool to analyse the code of the application to identify vulnerabilities before the program is executed. This will help identify risks early in the development process. GitHub Code Scanning can be integrated with various SAST tools such as SonarQube Community Edition which support multiple programming languages such as Java, C#, JavaScript etc.



Automatically detect vulnerabilities in your code.

Configure tools that integrate with Code Scanning to keep the quality of your code under control. Learn more about Code Scanning.

GitHub Code Scanning

SonarQube Community Edition

2. **DAST Testing -** Implement a Dynamic Application Security Testing (tool) to analyse the runtime of the application and identify how the Chameleon web application behave while in production. This form of testing if useful towards during development phase and when migrated to production. The tool will continually search for weaknesses in application, so developers can patch the vulnerabilities before being exploited. The OWASP ZAP (Zed Attack Proxy) is an open-source DAST tool that can be used to security test Chameleon website's runtime.

OWASP ZAP

3. **Snyk -** The open source Snyk tool is a security vulnerability scanning tool that can be integrated with Docker. It is used to identify vulnerabilities within Docker images, software packages and dependencies. Testing should be used to perform vulnerability scanning for containerised applications such as the Chameleon web application as it will identify potential vulnerabilities in the software components that have been bundled with the Chameleon Docker image.

Snyk

Network Security

Network security, in particular transport security is an important to secure the web application's communication over various networks. It is imperative that the communication is encrypted between the Chameleon application and other clients and hosts it interacts with within the network. This is to safeguard the integrity of the data during transmission and prevent malicious actors from tampering or manipulating the data.

The following recommendations have been provided to secure the communication with the Chameleon we application:

1. **HTTPS (TLS) –** It is crucial that that data is transmitted using HTTPS to ensure that the communication is encrypted and secure. Transport Layer Security is the protocol that ensures the privacy of communication between the web application and the client it is connected to. It is recommended that version TLS 1.2 and above is used due the minimal security risks.

2. **Digital certificate -** For authentication measures, a Digital Certificate must be issued by a Trusted Certificate Authority to ensure the legitimacy of the domain and the digital certificate. Users and other systems will automatically trust a digital certificate signed by a Trusted CA due to it being from a trusted source for secure online communication.

3. **Web Application Firewall (WAF) –** A WAF must be configured for the Chameleon Website to ensure the application is protected from Cross Site Scripting, DDoS and SQL Injection attacks. It will monitor, filter and block any malicious HTTP traffic to the Chameleon application. It is recommended that GCP's Cloud Armor WAF is applied to the application to provided an extra layer of defence against such attacks.

Google Cloud Armor WAF

Log Management

Log management is an important aspect of cyber security. It provides security and integrity of information and can be used for multiple purposes such as detecting and responding to security incidents, forensic analysis, compliance and auditing and activity monitoring of users and systems.

The following log management recommendations have been provided to help secure the integrity of the Chameleon application:

1. **Logged Events –** The logs events below need to be collected:
    - **a.** Authentication**:** To identify who has accessed the system to identify threat, unauthorised attempts and user activity.
    - **b.** Denied Access – Helps understand the source of unauthorised attempts made to access the application the types of attempts made.
    - **c.** Crashes – Identify bugs or vulnerabilities within the application.
    - **d.** Error Messages – Assists in troubleshooting issues with the application so that the issues or vulnerabilities can be addressed quickly.

2. **Centralised Storage -** All logs must be stored in a centralised service to ensure all events stored are easily accessible. It also simplifies the process for security professionals or sys admins to analyse the information in the event of a security incident. It is recommended that GCP's Google Cloud Logging is used to store the logs given that the Chameleon application runs on the Google Cloud Platform and can be integrated with other services such as GitHub and Docker.

Google Cloud Logging