CredShields
Blockchain Application Audit

Jan 28th, 2022 • CONFIDENTIAL

Description

This document details the process and result of the Gitopia Web and Blockchain audit performed by CredShields Technologies PTE. LTD. on behalf of Gitopia between Jan 4th, 2022, and Jan 28th, 2023. And a retest was performed on Jan 28th, 2022.

Author

Shashank (Co-founder, CredShields)

shashank@CredShields.com

Reviewers

Aditya Dixit (Research Team Lead)

aditya@CredShields.com

Prepared for

Gitopia

Table of Contents

1. Executive Summary

Gitopia engaged CredShields to perform a Web Application and Blockchain audit from Jan 4th, 2022, to Jan 28th, 2022. During this timeframe, Eighteen (18) vulnerabilities were identified. **A retest was performed on Jan 28th, 2022, and all the bugs have been addressed.**

During the audit, Five (5) vulnerabilities were found with a severity rating of either High or Critical. These vulnerabilities represent the greatest immediate risk to "Gitopia" and should be prioritized for remediation, and fortunately, none were found.

The table below shows the in-scope assets and a breakdown of findings by severity per asset. Section 2.3 contains more information on how severity is calculated.

Assets in Scope	Critical	High	Medium	Low	Info	Σ
Gitopia Blockchain	0	2	4	3	0	**9**
Gitopia Web	0	2	1	3	0	**6**
Git Remote	0	0	1	0	0	**1**
Git Server	0	1	1	0	0	**2**
	0	**5**	**7**	**6**	**0**	**18**

Table: Vulnerabilities Per Asset in Scope

The CredShields team conducted the security audit to focus on identifying vulnerabilities in the Web Application and Blockchain scope during the testing window while abiding by the policies set forth by the Gitopia team.

State of Security

To maintain a robust security posture, it is essential to continuously review and improve upon current security processes. Utilizing CredShields' continuous audit feature allows both Gitopia's internal security and development teams to not only identify specific vulnerabilities but also gain a deeper understanding of the current security threat landscape.

To ensure that vulnerabilities are not introduced when new features are added, or code is refactored, we recommend conducting regular security assessments. Additionally, by analyzing the root cause of resolved vulnerabilities, the internal teams at Gitopia can implement both manual and automated procedures to eliminate entire classes of vulnerabilities in the future. By taking a proactive approach, Gitopia can future-proof its security posture and protect its assets.

2. Methodology

Gitopia engaged CredShields to perform a Gitopia Web Application and Blockchain audit. The following sections cover how the engagement was put together and executed.

2.1 Preparation phase

The CredShields team intercepted all the APIs originating from the web application to understand the application's features and functionalities. They meticulously examined all functions and created a mind map to systematically identify potential security vulnerabilities, prioritizing those that were more critical and business-sensitive for the refactored code. To confirm their findings, the team also worked on the staging version of the Web application and performed verifications and validations during the audit phase. The team also deployed the blockchain locally to perform the testing for the blockchain application. The team also had access to the source code to perform the whitebox audit.

A testing window from Jan 4th, 2022, to Jan 28th, 2022, was agreed upon during the preparation phase.

2.1.1 Scope

During the preparation phase, the following scope for the engagement was agreed-upon:

IN SCOPE ASSETS
- **Git Server Source Code** - **Git Remote Source Code** - **https://gitopia.dev** - **Gitopia Blockchain Source Code**

Table: List of Files in Scope

2.1.2 Documentation

Documentation was provided to the team internally during the audit.

2.1.3 Audit Goals

CredShields uses both in-house tools and manual methods for comprehensive Web Application audits. The majority of the audit is done by manually reviewing the Web Application features and API calls, following OWASP Web application security standards, and an extended industry standard self-developed checklist. The team places emphasis on understanding core concepts, preparing test cases, and evaluating business logic for potential vulnerabilities.

2.2 Retesting phase

Gitopia is actively partnering with CredShields to validate the remediations implemented towards the discovered vulnerabilities.

2.3 Vulnerability classification and severity

CredShields follows OWASP's Risk Rating Methodology to determine the risk associated with discovered vulnerabilities. This approach considers two factors - Likelihood and Impact - which are evaluated with three possible values - **Low**, **Medium**, and **High**, based on factors such as Threat agents, Vulnerability factors, Technical and Business Impacts. The overall severity of the risk is calculated by combining the likelihood and impact estimates.

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

Overall, the categories can be defined as described below -

1. Informational

We prioritize technical excellence and pay attention to detail in our coding practices. Our guidelines, standards, and best practices help ensure software stability and reliability. Informational vulnerabilities are opportunities for improvement and do not pose a direct risk to the organization. Code maintainers should use their own judgment on whether to address them.

2. **Low**

   Low-risk vulnerabilities are those that either have a small impact or can't be exploited repeatedly or those the client considers insignificant based on their specific business circumstances.

3. **Medium**

   Medium-severity vulnerabilities are those caused by weak or flawed logic in the code and can lead to exfiltration or modification of private user information. These vulnerabilities can harm the client's reputation under certain conditions and should be fixed within a specified timeframe.

4. **High**

   High-severity vulnerabilities pose a significant risk to the Web Application and the organization. They can result in the loss of funds for some users, may or may not require specific conditions, and are more complex to exploit. These vulnerabilities can harm the client's reputation and should be fixed immediately.

5. **Critical**

   Critical issues are directly exploitable bugs or security vulnerabilities that do not require specific conditions. They often result in the loss of funds for users or put sensitive user information at risk of compromise or modification. The client's reputation and financial stability will be severely impacted if these issues are not addressed immediately.

2.4 CredShields staff

The following individual at CredShields managed this engagement and produced this report:

- **Shashank, Co-founder CredShields**
    - shashank@CredShields.com

Please feel free to contact this individual with any questions or concerns you have around the engagement or this document.

3. Findings

This chapter contains the results of the security assessment. Findings are sorted by their severity and grouped by the asset and CWE classification. Each asset section will include a summary. The table in the executive summary contains the total number of identified security vulnerabilities per asset per risk indication.

3.1 Findings Overview

3.1.1 Vulnerability Summary

During the security assessment, Eighteen (18) security vulnerabilities were identified in the asset.

VULNERABILITY TITLE	SEVERITY	CWE | Vulnerability Type
Deprecated Function	Medium	Using components with known vulnerabilities

Table: Findings in Git Remote

VULNERABILITY TITLE	SEVERITY	CWE | Vulnerability Type
Unrestricted File Upload	High	Unrestricted Upload of File with Dangerous Type [CWE-434]

Deprecated Function	Medium	Using components with known vulnerabilities

Table: Findings in Git Server

VULNERABILITY TITLE	SEVERITY	CWE | Vulnerability Type
Missing access control on issue deletion	Medium	Access control - [CWE-284]
Race condition in faucet	High	Race condition [CWE-362]
Missing max_credit limit in faucet	Medium	Access control - [CWE-284]
Update-repository breaks ownership	High	Update-repository breaks ownership
Missing Input Validation in the branch leading to Path Manipulation	Low	Missing Input Validation in the branch leading to Path Manipulation
Missing Input Validation in the branch leading to Path Manipulation	Low	Input validation [CWE-20]
Missing Input Validation in “tag” leading to Path Manipulation	Low	Input validation [CWE-20]
Mixed Content in Avatar URL in “update-organization”	Medium	Man in the middle attacks. [CWE-300]
Email leak via DAO endpoint	Medium	Sensitive information disclosure [CWE-200]

Table: Findings in Gitopia Blockchain

VULNERABILITY TITLE	SEVERITY	CWE | Vulnerability Type
Unencrypted Storage of password	High	Plain text storage of password [CWE-256]
Weak password policy	Medium	Weak Password Requirements [CWE-521]
Missing Security Headers	Low	Misconfiguration [CWE-16]
ClickJacking	Low	User Interface (UI) Misrepresentation of Critical Information [CWE-451]
Missing Input Validation in Wallet Name	Low	Improper Input Validation - [CWE-20]
Large File Denial Of Service	High	Denial of service - [CWE-400]

Table: Findings in Gitopia Web Application

4. Remediation Status

Gitopia is actively partnering with CredShields from this engagement to validate the discovered vulnerabilities' remediations. **A retest was performed on Jan 28th, 2022, and all the issues have been addressed.**

Also, the table shows the remediation status of each finding.

VULNERABILITY TITLE	SEVERITY	REMEDIATION STATUS
Deprecated Function	Medium	**Fixed [28/01/2022]**

Table: Summary of findings and status of remediation (Git Remote)

VULNERABILITY TITLE	SEVERITY	REMEDIATION STATUS
Unrestricted File Upload	High	**Fixed [28/01/2022]**
Deprecated Function	Medium	**Fixed [28/01/2022]**

Table: Summary of findings and status of remediation (Git Server)

VULNERABILITY TITLE	SEVERITY	REMEDIATION STATUS
Missing access control on issue deletion	Medium	**Fixed [28/01/2022]**
Race condition in faucet	High	**Fixed [28/01/2022]**
Missing max_credit limit in faucet	Medium	**Fixed [28/01/2022]**
Update-repository breaks ownership	High	**Fixed [28/01/2022]**
Missing Input Validation in the branch leading to Path Manipulation	Low	**Fixed [28/01/2022]**
Missing Input Validation in the branch leading to Path Manipulation	Low	**Fixed [28/01/2022]**
Missing Input Validation in “tag” leading to Path Manipulation	Low	**Fixed [28/01/2022]**
Mixed Content in Avatar URL in “update-organization”	Medium	**Fixed [28/01/2022]**

Table: Summary of findings and status of remediation (Gitopia Blockchain)

VULNERABILITY TITLE	SEVERITY	REMEDIATION STATUS
Unencrypted Storage of password	High	**Fixed [28/01/2022]**
Weak password policy	Medium	**Fixed [28/01/2022]**

Missing Security Headers	Low	**Fixed [28/01/2022]**
ClickJacking	Low	**Fixed [28/01/2022]**
Missing Input Validation in Wallet Name	Low	**Fixed [28/01/2022]**
Large File Denial Of Service	High	**Fixed [28/01/2022]**

Table: Summary of findings and status of remediation (Gitopia Web)

5. Bug Reports

5. 1 Git Remote

Bug ID#1

Deprecated Function

Vulnerability Type
Using components with known vulnerabilities

Severity
Medium

Description:
Git-remote-gitopia is found to be using a GRPC package provided by golang. This consists of a lot of deprecated functionality and features. One such function is “WithInsecure()” which is deprecated and is not recommended to be used anymore.

PoC:

- The affected code can be seen in the following lines (Line 76) -

```
func (h *GitopiaHandler) Initialize(remote *core.Remote) error {
   var err error

   h.grpcConn, err = grpc.Dial(apiURL,
       grpc.WithInsecure(),
   )
   if err != nil {
       return err
   }...
```

- It can be seen in the package documentation that the function is deprecated - https://pkg.go.dev/google.golang.org/grpc#WithInsecure

Impacts:

Using deprecated functions may cause undesired effects and errors.

Remediation:

It is recommended to use "WithTransportCredentials" and "insecure.NewCredentials()" instead.

Retest:

5.2 Git Server
Bug ID#1

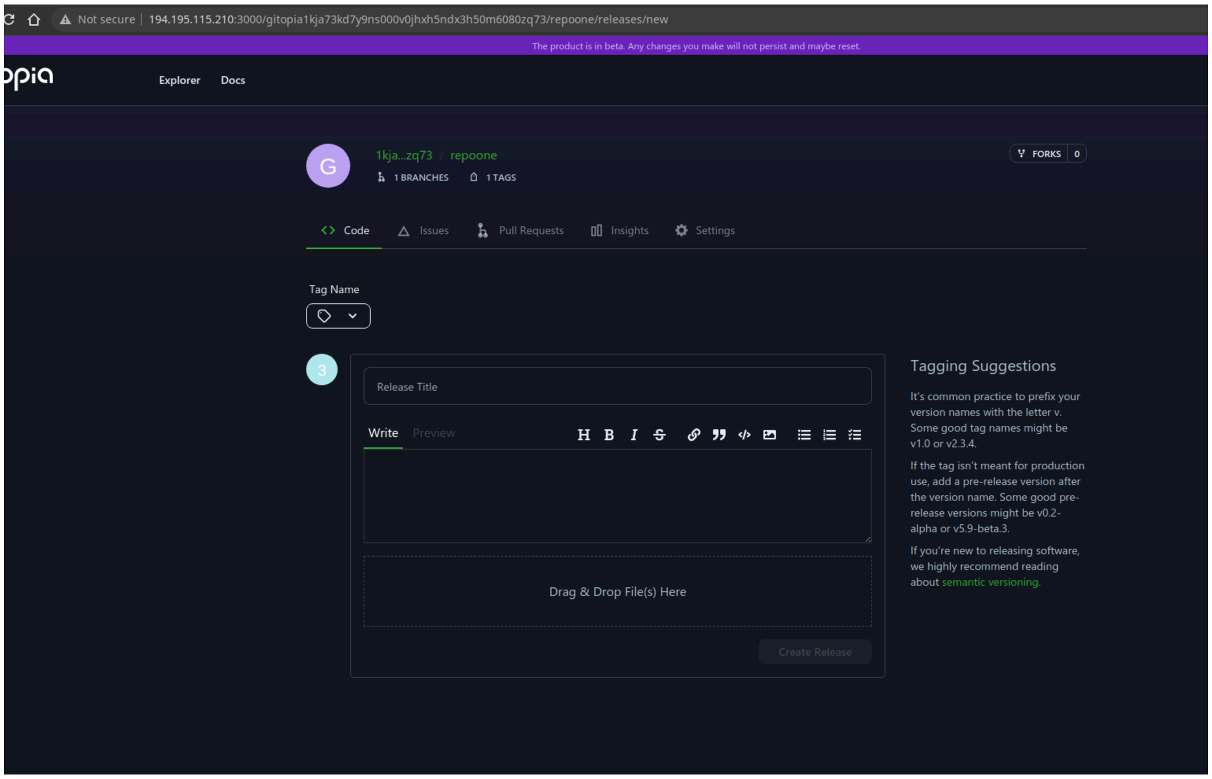
Unrestricted File Upload

Vulnerability Type
Unrestricted Upload of File with Dangerous Type [CWE-434]

Severity
High

Description:
The Releases page does not have any validations on the size of the files being uploaded. This will allow users to upload files with very large sizes consuming all the bandwidth and resources.

PoC:
- Go to the Releases section in any repository.
- Try to upload a file with large size (> 2 GB)
- You’ll notice that it does not have any file size validations and will allow uploading files with any sizes


Not secure | 194.195.115.210:3000/gitopia1kja73kd7y9ns000v0jhxh5ndx3h50m6080zq73/repoone/releases/new
The product is in beta. Any changes you make will not persist and maybe reset.
Explorer
Docs
1kja...zq73 / repoone
1 BRANCHES
1 TAGS
FORKS 0
Code
Issues
Pull Requests
Insights
Settings
Tag Name
Release Title
Write
Preview
Drag & Drop File(s) Here
Create Release
Tagging Suggestions
It's common practice to prefix your version names with the letter v. Some good tag names might be v1.0 or v2.3.4.
If the tag isn't meant for production use, add a pre-release version after the version name. Some good pre-release versions might be v0.2-alpha or v5.9-beta.3.
If you're new to releasing software, we highly recommend reading about semantic versioning.

Impacts

Allowing users to upload files with large sizes will lead to resource consumption and bandwidth usage and may lead to potential Denial of Services.

Remediation

Implement a limit on file size to be uploaded through the release. Github uses a 2GB limit and the same is recommended.

Retest:

Bug ID#2

Deprecated Function

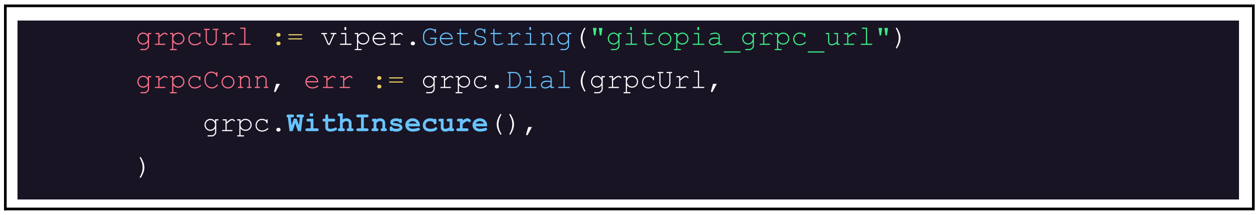
Vulnerability Type
Using components with know vulnerabilities

Severity
Medium

Description:
Git-server is found to be using a GRPC package provided by golang. This consists of a lot of deprecated functionality and features. One such function is “WithInsecure()” which is deprecated and is not recommended to be used anymore.

PoC:

- The affected code can be seen in the following lines (Line 394 - main.go) and (Line 133 - handler.go) -



```
	grpcUrl := viper.GetString("gitopia_grpc_url")
	grpcConn, err := grpc.Dial(grpcUrl,
		grpc.WithInsecure(),
	)
```

- It can be seen in the package documentation that the function is deprecated - https://pkg.go.dev/google.golang.org/grpc#WithInsecure

Impacts:
Using deprecated functions may cause undesired effects and errors.

Remediation:
It is recommended to use “WithTransportCredentials” and “insecure.NewCredentials()” instead.

Retest:

5.3 Gitopia Blockchain
Bug ID#1

Missing access control on issue deletion

Vulnerability Type
Access control - [CWE-284]

Severity
Medium

Description:
It was noticed that users can delete the issues created by them after the repository has been transferred. Although the feature seems to be broken right now still the access control should be in place as after transfer the user cannot add codes but can delete the issues. As per GitHub standards, an issue can be deleted only by the owner of the repository.

PoC:
- We'll be using two wallets to replicate this issue.
- From the first wallet, create a repository, push some code in it and create an issue in that repository.
- Now transfer the ownership of the repository to the other account.
- Try deleting the issue using the first account and it'll still be successful after transferring the ownership.

Impacts:
A malicious user can delete issues created by them after ownership transfer.

Remediation:
Implement the transfer ownership function correctly so that all the access controls are also transferred to the new owner.
Revoke access to the old owner after the transfer. Also, do not allow the issues to be deleted by other users if they're not the owner.

Retest:

Bug ID#2

Race condition in faucet

Vulnerability Type
Race condition [CWE-362]

Severity
High

Description:
During the audit, it was noticed that max_credit could be bypassed allowing a user to add more funds than the limit by sending concurrent requests to the server. This was later found to be a bug in the core starport and the tendermint team was notified.

PoC:
- Go to https://gitopia.dev/
- And we will notice that we can fetch an unlimited amount of tokens from the faucet.

Impacts:
A malicious actor can gain more funds from the faucet than the required limit by sending concurrent requests to the server. This was found to be an issue with the official tendermint code.

Remediation:
Tendermint team was informed and the Gitopia team can just update the latest code to apply the patch once fixed.
https://github.com/tendermint/starport/issues/1965

Retest:

Bug ID#3 [Fixed]

Missing max_credit limit in faucet

Vulnerability Type
Access control - [CWE-284]

Severity
Medium

Description:
It was noticed that the dev endpoint had max_credit limit missing that allowed a user to drain all faucet funds.

PoC:

- Go to https://gitopia.dev/
- And we will notice that we can fetch an unlimited amount of tokens from faucet

Impacts:
A malicious user could fetch unlimited faucet tokens by repeating the request in an automated manner.

Remediation:
Implement max_credit limit in the faucet.

Retest:
A max_credit limit was added to the faucet.

Bug ID#4

Update-repository breaks ownership

Vulnerability Type

Bussiness logic [CWE-840]

Severity

High

Description:

It was noticed that using the gitopiad command “update-repository” to update the repository breaks the functionality of ownership transfer. Hence, once the CLI command is used to update the repository, a user won’t be able to transfer ownership of the repository.

PoC:

- Create a repository from the frontend.
- From the CLI, using gitopiad, run the command to “update-repository” as shown below -

```
gitopiad --home ~/.gitopia/ tx gitopia update-repository 20
"newname" "gitopia123r8sdyhklx68v0ml08j
0e5j3fqnjs9ehzvqrs" "newdesc" 1 "test" "master" --from node0
```

- After editing your new repository, go to “Settings” and transfer the ownership to any other address.
- You’ll notice that it’ll throw an error saying the user is not authorized to transfer the ownership.

Impacts:

The use of CLI, in this case, breaks the functionality of the application.

Remediation:

Fix the root cause that breaks the ownership transfer functionality.

Retest:

Bug ID#5

Missing Input Validation in the branch leading to Path Manipulation

Vulnerability Type
Input validation [CWE-20]

Severity
Low

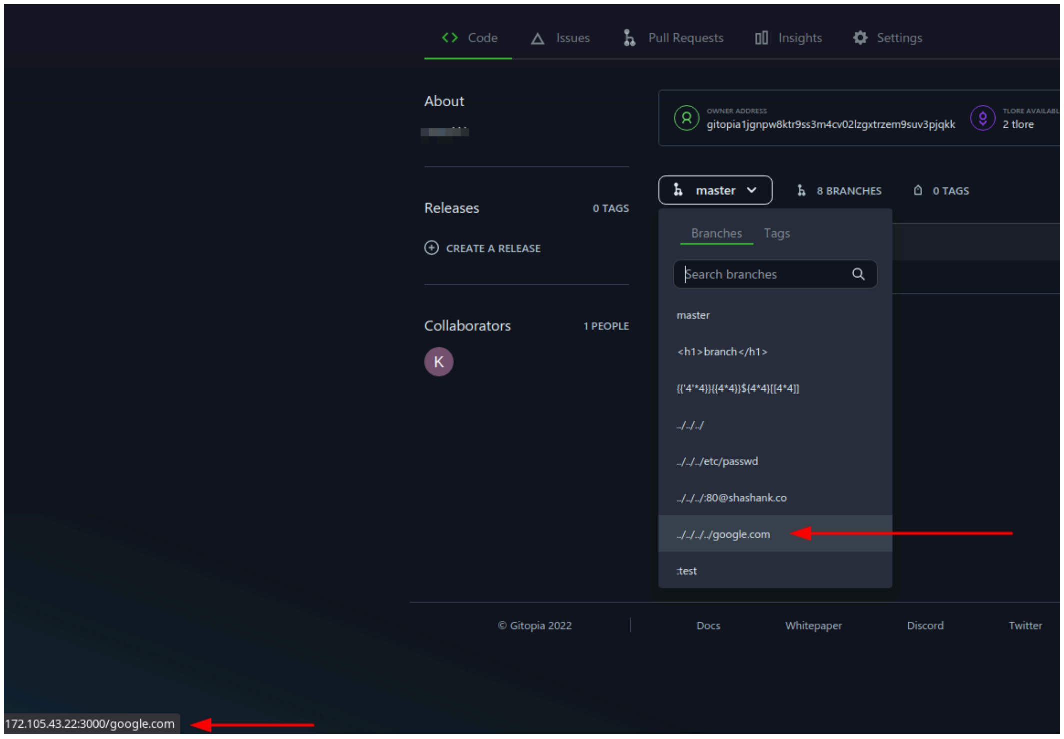
Description:
It was noticed that the branch name was missing input validation that allowed any user to create malicious brach that would redirect users to different web paths.

PoC:

- Use the “gitopiad” CLI to interact with the server. Create a new repository branch using “set-repository-branch”

```
gitopiad tx gitopia set-repository-branch 1 "../../../test" "123"
--from node0
```

- After running the above command, it can be seen that the path for the new branch will be manipulated as soon as someone clicks on it in the UI.



Impacts:

This may be abused in phishing attacks. Adversaries can use this to manipulate their victims into visiting other malicious repositories thinking they're on the same repository but on a different branch.

Remediation:

Implement an input validation on the branch name parameter so that it does not take special characters and path manipulation letters such as "." and "/".

The SHA256 parameter should also have some input validation to check if the user is sending a correct value.

Retest:

Bug ID#6

Missing Input Validation in SHA256 Hash

Vulnerability Type
Input validation [CWE-20]

Severity
Low

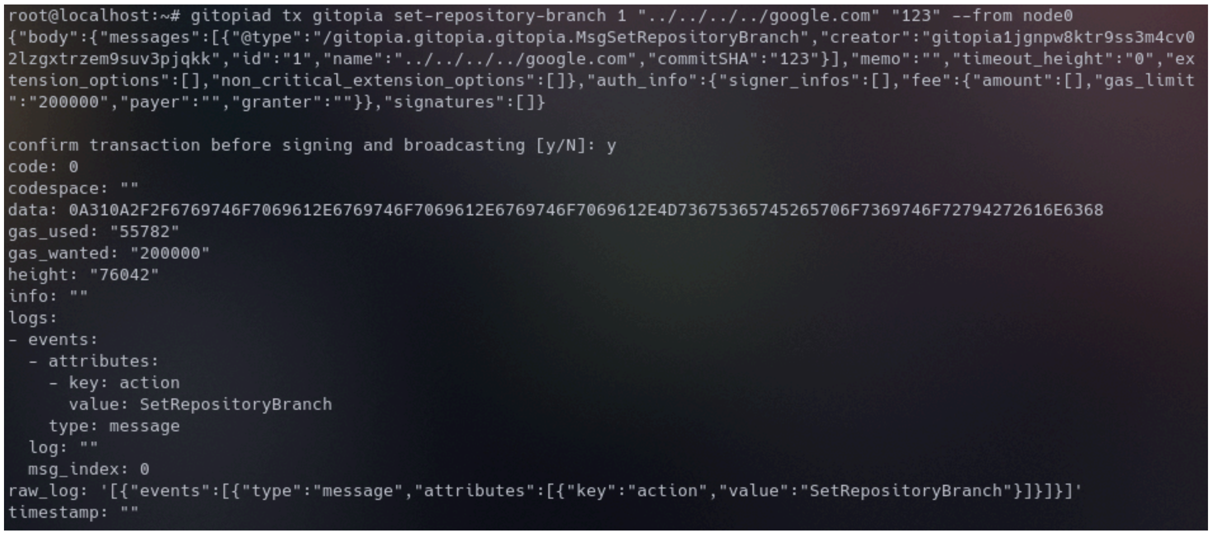
Description:
The SHA256 Hash is not validated when creating a new repository branch using the "gitopiad" CLI. This may be set to any arbitrary value which may lead to hash collisions.

PoC:

- Use the "gitopiad" CLI to interact with the server. Create a new repository branch using "set-repository-branch"

```
gitopiad tx gitopia set-repository-branch 1 "../../../test" "123"
--from node0
```

- After running the above command, it can be seen that the SHA hash of random value will also be accepted.



```
root@localhost:~# gitopiad tx gitopia set-repository-branch 1 "../../../../google.com" "123" --from node0
{"body":{"messages":[{"@type":"/gitopia.gitopia.gitopia.MsgSetRepositoryBranch","creator":"gitopia1jgnpw8ktr9ss3m4cv0
2lzgxtrzem9suv3pjqkk","id":"1","name":"../../../../google.com","commitSHA":"123"}],"memo":"","timeout_height":"0","ex
tension_options":[],"non_critical_extension_options":[]},"auth_info":{"signer_infos":[],"fee":{"amount":[],"gas_limit
":"200000","payer":"","granter":""}},"signatures":[]}

confirm transaction before signing and broadcasting [y/N]: y
code: 0
codespace: ""
data: 0A310A2F2F6769746F7069612E6769746F7069612E6769746F7069612E4D736753657452657706F7369746F72794272616E6368
gas_used: "55782"
gas_wanted: "200000"
height: "76042"
info: ""
logs:
- events:
  - attributes:
    - key: action
      value: SetRepositoryBranch
    type: message
  log: ""
  msg_index: 0
raw_log: '[{"events":[{"type":"message","attributes":[{"key":"action","value":"SetRepositoryBranch"}]}]}]'
timestamp: ""
```

Impacts:

TBD

Remediation:

The SHA256 parameter should have some input validation to check if the user is sending the correct value.

Retest:

Bug ID#7

Missing Input Validation in “tag” leading to Path Manipulation

Vulnerability Type

Input validation [CWE-20]

Severity

Low

Description:

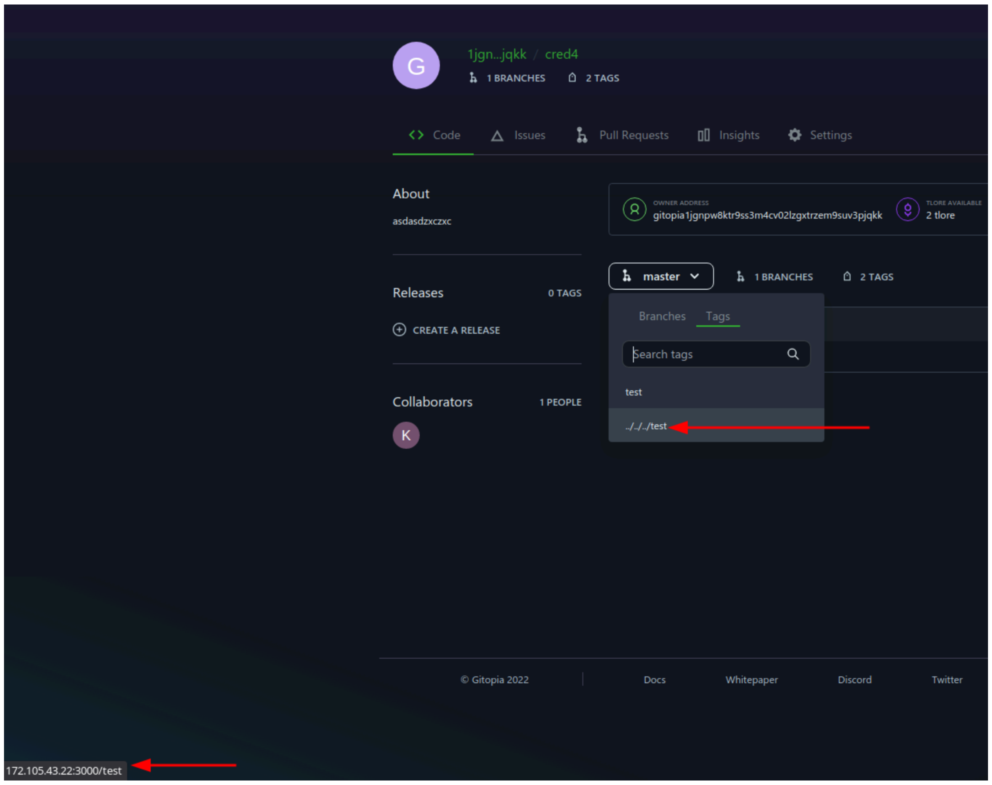
It was noticed that the tag name was missing input validation that allowed any user to create malicious brach that would redirect users to different web paths.

PoC:

- Use the “gitopiad” CLI to interact with the server. Create a new repository branch using “set-repository-tag”

```
gitopiad tx gitopia set-repository-tag 1 "../../../test" "123"
--from node0
```

- After running the above command, it can be seen that the path for the new tag will be manipulated as soon as someone clicks on it in the UI.



Impacts:
This may be abused in phishing attacks. Adversaries can use this to manipulate their victims into visiting other malicious repositories thinking they're on the same repository but on a different URL.

Remediation:
Implement an input validation on the branch name parameter so that it does not take special characters and path manipulation letters such as "." and "/".

Retest:

Bug ID#8

Mixed Content in Avatar URL in “update-organization”

Vulnerability Type

Man in the middle attacks. [CWE-300]

Severity

Medium

Description:

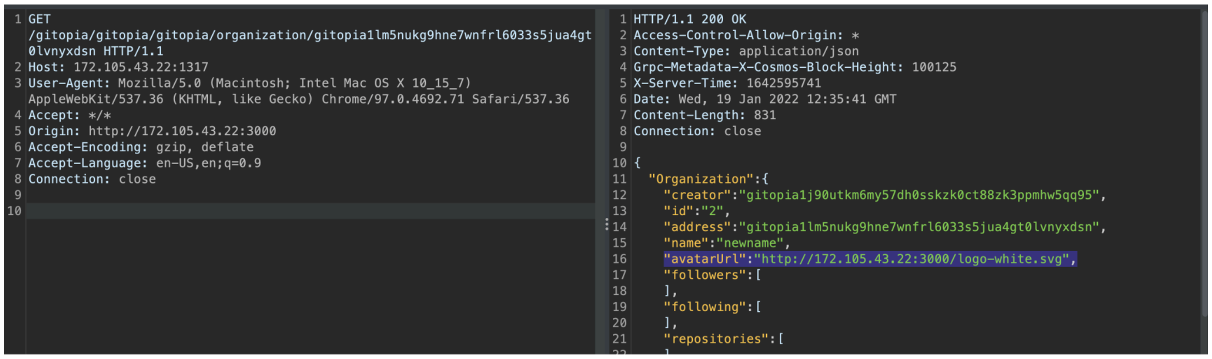
It was noticed that the Avatar URL of the parameters in the CLI command “update-organization” was missing input validation allowing the team to have any URL as the image URL even if it was not an image. This also allowed us to keep an insecure HTTP URL as an image.

PoC:

- Use the “gitopiad” CLI to interact with the server. Create a new organization and use the following command to update the organization.

```
gitopiad tx gitopia update-organization
gitopia1lm5nukg9hne7wnfrl6033s5jua4gt0lvnyxdsn "newname" "h
ttp://172.105.43.22:3000/logo-white.svg" "asd"
"credshields@leak.com" "http://credshields.com" "newdescr" --from
node
3
```

- After running the above command, it can be seen that the path for the image will be changed to an HTTP URL.



```
GET
/gitopia/gitopia/gitopia/organization/gitopia1lm5nukg9hne7wnfrl6033s5jua4gt
0lvnyxdsn HTTP/1.1
Host: 172.105.43.22:1317
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.4692.71 Safari/537.36
Accept: */*
Origin: http://172.105.43.22:3000
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
Grpc-Metadata-X-Cosmos-Block-Height: 100125
X-Server-Time: 1642595741
Date: Wed, 19 Jan 2022 12:35:41 GMT
Content-Length: 831
Connection: close

{
  "Organization":{
    "creator":"gitopia1j90utkm6my57dh0sskzk0ct88zk3ppmhw5qq95",
    "id":"2",
    "address":"gitopia1lm5nukg9hne7wnfrl6033s5jua4gt0lvnyxdsn",
    "name":"newname",
    "avatarUrl":"http://172.105.43.22:3000/logo-white.svg",
    "followers":[
    ],
    "following":[
    ],
    "repositories":[
```

Impacts:

Possible MITM attacks and most updated browsers will flag mixed contents.

Remediation:

Validate that the URL entered by a user is an actual image. Do not allow arbitrary protocols. Use a whitelist approach to only allow "HTTPS" in the image URL.

Retest:

Bug ID#9

Email leak via DAO endpoint

Vulnerability Type

Sensitive information disclosure [CWE-200]

Severity

Medium

Description:

The gitopiad command for "update-organization" accepts the value of email address. This email address gets reflected at the endpoint of the DAO http://172.105.43.22:1317/gitopia/gitopia/gitopia/organization/gitopia1lm5nukg9hne7wnfrl6033s5jua4gt0lvnyxdsn

PoC:

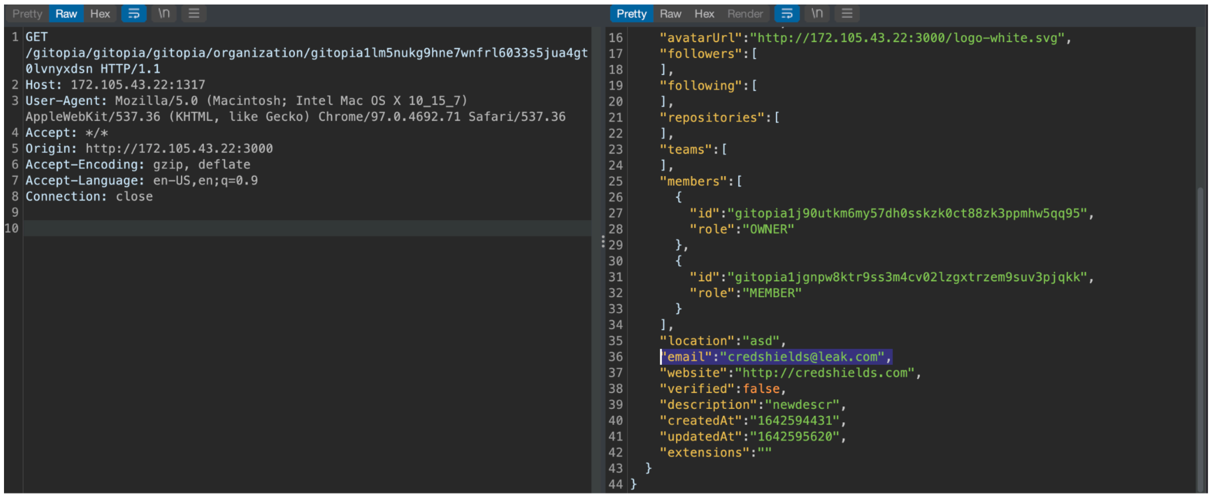
- Use the "gitopiad" CLI to interact with the server. Run the update organization command and set any email address.

```
gitopiad tx gitopia update-organization
gitopia1lm5nukg9hne7wnfrl6033s5jua4gt0lvnyxdsn "newname"
"http://172.105.43.22:3000/logo-white.svg" "asd"
"credshields@leak.com" "http://credshields.com" "newdescr" --from
node3
```

- Call the below API request of the DAO address.

```
http://172.105.43.22:1317/gitopia/gitopia/gitopia/organization/gitopia1lm5nukg9h
ne7wnfrl6033s5jua4gt0lvnyxdsn
```

- We will notice the email publicly disclosed.



```
GET
/gitopia/gitopia/gitopia/organization/gitopia1lm5nukg9hne7wnfrl6033s5jua4gt
0lvnyxdsn HTTP/1.1
Host: 172.105.43.22:1317
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.4692.71 Safari/537.36
Accept: */*
Origin: http://172.105.43.22:3000
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

```
    "avatarUrl":"http://172.105.43.22:3000/logo-white.svg",
    "followers":[
    ],
    "following":[
    ],
    "repositories":[
    ],
    "teams":[
    ],
    "members":[
      {
        "id":"gitopia1j90utkm6my57dh0sskzk0ct88zk3ppmhw5qq95",
        "role":"OWNER"
      },
      {
        "id":"gitopia1jgnpw8ktr9ss3m4cv02lzgxtrzem9suv3pjqkk",
        "role":"MEMBER"
      }
    ],
    "location":"asd",
    "email":"credshields@leak.com",
    "website":"http://credshields.com",
    "verified":false,
    "description":"newdescr",
    "createdAt":"1642594431",
    "updatedAt":"1642595620",
    "extensions":""
  }
}
```

Impacts:

A malicious user can write a script to harvest all the email addresses from the DAO endpoints.

Remediation:

Give an option for users to mask their email ID from public endpoints.
Or remove the email flag if this is not at all required.

Retest:

5.4 Gitopia Web Application

Bug ID#1

Unencrypted Storage of password

Vulnerability Type

Plain text storage of password [CWE-256]

Severity

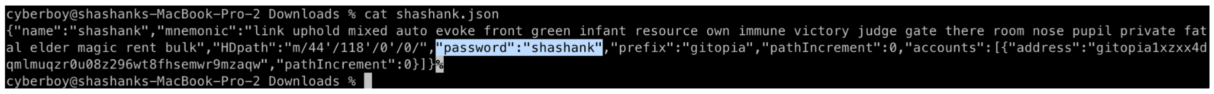
High

Description:

The application was found to be storing passwords and mnemonics in plain texts when imported as a file.

PoC:

- Go to http://localhost:3000/home
- Click on “connect wallet” and then “Create new wallet ”
- Click “Download Backup”
- Now we can notice the text file holds the mnemonic in cleartext.



```
cyberboy@shashanks-MacBook-Pro-2 Downloads % cat shashank.json
{"name":"shashank","mnemonic":"link uphold mixed auto evoke front green infant resource own immune victory judge gate there room nose pupil private fat
al elder magic rent bulk","HDpath":"m/44'/118'/0'/0/","password":"shashank","prefix":"gitopia","pathIncrement":0,"accounts":[{"address":"gitopia1xzxx4d
qmlmuqzr0u08z296wt8fhsemwr9mzaqw","pathIncrement":0}]}%
cyberboy@shashanks-MacBook-Pro-2 Downloads %
```

Impacts:

This vulnerability may be exploited by attackers who manage to get access to the backup wallet either using other attacks or by having physical access to the device.

This will result in loss of funds from the users’ wallets and compromise of the user repositories.

Remediation:

Encrypt password and mnemonics downloaded in the JSON file.

Retest:

Bug ID#2

Weak password policy

Vulnerability Type
Weak Password Requirements [CWE-521]

Severity
Medium

Description:
The application does not require that users should have strong passwords, which makes it easier for attackers to compromise user accounts.
An authentication mechanism is only as strong as its credentials. For this reason, it is important to require users to have strong passwords. Lack of password complexity significantly reduces the search space when trying to guess a user's passwords, making brute-force attacks easier.
https://cwe.mitre.org/data/definitions/521.html

PoC:

- Go to http://localhost:3000/home
- Click on “connect wallet” and then “Create new wallet ”
- Set password value to “a”
- We will notice the weak single-digit password was accepted.

Note: Blockchain RPC calls cannot have a rate limit and hence it is important to have very strong passwords.

Impacts
An attacker can guess weak passwords using dictionary attacks.

Remediation
Implement a strong passwords policy

1. Allow all characters to be used for passwords to avoid shortening the keyspace for brute-force guessing.
2. Do not impose character restrictions such as "must have at least X number of specific character type" in the password. This will shorten the keyspace for brute-force guessing.
3. Disallow short password lengths. 8 characters are generally considered a good minimum password length.
4. Allow for a large maximum password length.
5. Do not advertise the maximum password length as this will shorten the key space for brute-force guessing.

Retest:

Bug ID#3

Missing Security Headers

Vulnerability Type
Misconfiguration [CWE-16]

Severity
Low

Description:
HTTP headers are well known and their implementation can make your application more versatile and secure.
Modern browsers support many HTTP headers that can improve web application security to protect against clickjacking, cross-site scripting, and other common attacks.
The important ones are missing and should be added to the response headers. They are listed below.

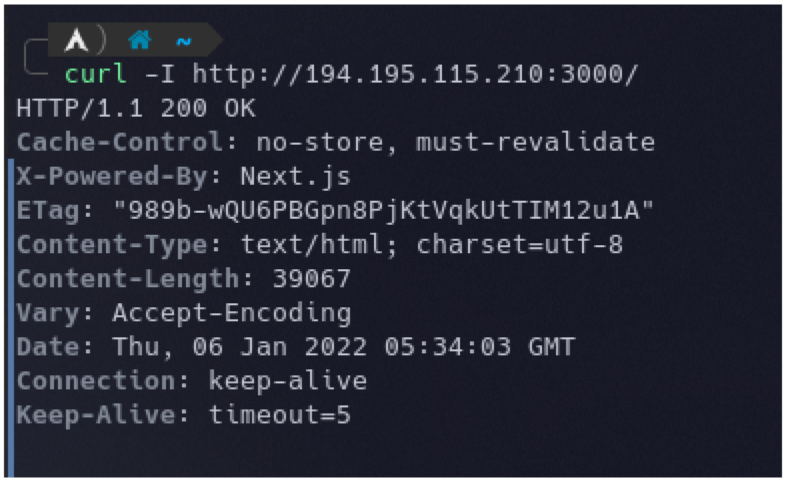
- **Strict-Transport-Security** - HTTP Strict Transport Security is an excellent feature to support your site and strengthens your implementation of TLS by getting the User-Agent to enforce the use of HTTPS. Recommended value "Strict-Transport-Security: max-age=31536000; includeSubDomains".
- **Content-Security-Policy** - Content Security Policy is an effective measure to protect your site from XSS attacks. By whitelisting sources of approved content, you can prevent the browser from loading malicious assets.
- **X-Frame-Options** - X-Frame-Options tell the browser whether you want to allow your site to be framed or not. By preventing a browser from framing your site you can defend against attacks like clickjacking. Recommended value "X-Frame-Options: SAMEORIGIN".
- **X-Content-Type-Options** - X-Content-Type-Options stops a browser from trying to MIME-sniff the content type and forces it to stick with the declared content-type. The only valid value for this header is "X-Content-Type-Options: nosniff".

- **Referrer-Policy** - Referrer Policy is a new header that allows a site to control how much information the browser includes with navigations away from a document and should be set by all sites.
- **Permissions-Policy** - Permissions Policy is a new header that allows a site to control which features and APIs can be used in the browser.

Note that not all headers will be relevant for each endpoint, but security best practices stipulate that these headers should be set for all server responses, regardless of content.

PoC:

1. Send a curl request to view the missing headers



```
curl -I http://194.195.115.210:3000/
HTTP/1.1 200 OK
Cache-Control: no-store, must-revalidate
X-Powered-By: Next.js
ETag: "989b-wQU6PBGpn8PjKtVqkUtTIM12u1A"
Content-Type: text/html; charset=utf-8
Content-Length: 39067
Vary: Accept-Encoding
Date: Thu, 06 Jan 2022 05:34:03 GMT
Connection: keep-alive
Keep-Alive: timeout=5
```

It can be seen that all the necessary security headers noted in the Description are missing.

Impacts

The lack of these headers do not impact the application in a direct way but acts as an additional level of protection against other attacks such as Clickjacking, XSS, MITM, etc.
Taking advantage of the missing headers, most of the time requires the presence of another vulnerability.
The finding presents a low risk and demonstrates that the security posture of the tested applications is not in line with the best practices.

Remediation
Implement all the missing security headers mentioned in the description and as per the requirement of the application.

Retest:

Bug ID#4

ClickJacking

Vulnerability Type
User Interface (UI) Misrepresentation of Critical Information [CWE-451]

Severity
Low

Description:
Clickjacking, also known as a "UI redress attack", is when an attacker uses multiple transparent or opaque layers to trick a user into clicking on a button or link on another page when they were intending to click on the top-level page. Thus, the attacker is "hijacking" clicks meant for their page and routing them to another page, most likely owned by another application, domain, or both. Using a similar technique, keystrokes can also be hijacked. With a carefully crafted combination of stylesheets, iframes, and text boxes, a user can be led to believe they are typing in the password to their email or bank account, but are instead typing into an invisible frame controlled by the attacker.
https://cwe.mitre.org/data/definitions/451.html

PoC:

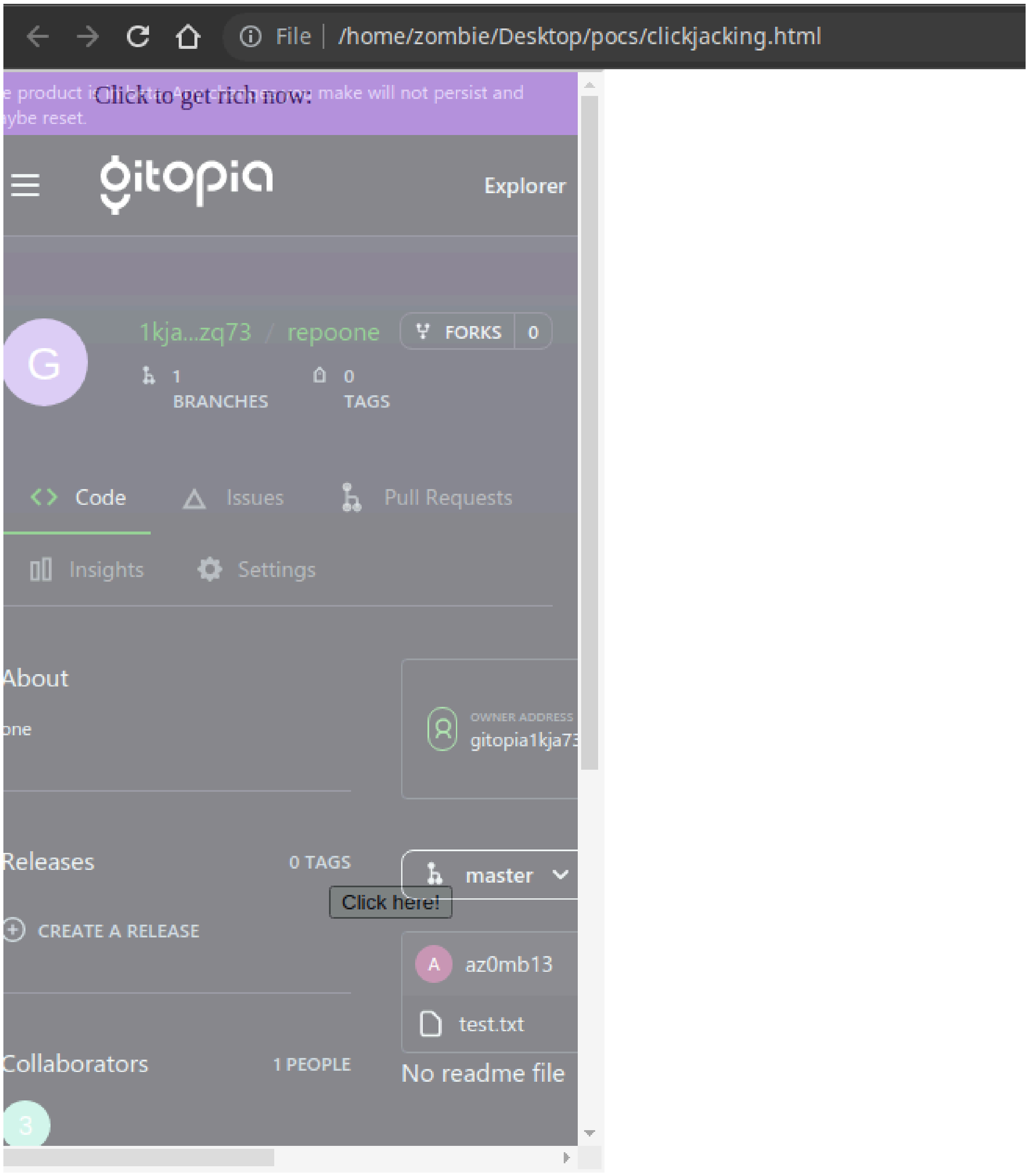
1. To replicate the issue, make sure you're logged in to the application in the same browser session.
2. Save the below PoC code as an HTML file and open it in the browser.

```
<style>
iframe { /* iframe from the victim site */
 width: 400px;
 height: 700px;
 position: absolute;
 top:0; left:-20px;
 opacity: 0.5; /* in real opacity:0 */
 z-index: 1;
}
</style>

<div style="margin-left: 50px">Click to get rich now:</div>

<!-- The url from the victim site -->
<iframe
src="http://194.195.115.210:3000/gitopia1kja73kd7y9ns000v0jhxh5ndx3h50m6
080zq73/repoone"></iframe>

<button style="margin-left:200px; margin-top:495px;">Click
here!</button>
```


File | /home/zombie/Desktop/pocs/clickjacking.html
e product is ... make will not persist and
ybe reset.
Click to get rich now!
gitopia
Explorer
1kja...zq73 / repoone
FORKS 0
1 BRANCHES
0 TAGS
Code
Issues
Pull Requests
Insights
Settings
About
one
OWNER ADDRESS
gitopia1kja73
Releases
0 TAGS
CREATE A RELEASE
master
Click here!
az0mb13
test.txt
No readme file
Collaborators
1 PEOPLE

Impacts

An attacker may trick a victim user to perform sensitive actions present in the application and the actual application being invisible due to `z-index` will make users think that they're performing actions on a completely different domain/app.

Remediation

You may use several techniques to secure against Clickjacking. You can also use several of them together to ensure full coverage. Here are the techniques in order of preference.

1. **Content-Security-Policy: frame-ancestors**

Content-Security-Policy (CSP) is an HTTP response header. It was designed primarily to protect against XSS. Currently, it also includes an anti-clickjacking frame-ancestors directive. This directive controls how the page can be embedded by different sites by specifying parent pages that may embed the page. Embedding control covers the following tags: <frame>, <iframe>, <embed>, <object>, and <applet>. Eg:

```
Content-Security-Policy: frame-ancestors 'self' '\*.website.com';
```

2. **X-Frame-Options**

X-Frame-Options (XFO) is an HTTP response header. This response header is a simple predecessor of the frame-ancestors directive. Formally, CSP frame-ancestors obsoletes the X-Frame-Options header. Eg:

```
X-Frame-Options: allow-from https://www.website.com/
```

3. **Framebusting**

Framebusting (also known as framebreaking or framekilling) is a client-side technique. It does not require any modifications to HTTP headers. All you need to do is modify your web page HTML code. It is the most generic method to protect against clickjacking and works even in legacy browsers (such as IE6). However, it is not as reliable as HTTP header options and in some cases may be circumvented. Eg:

```
<style id="antiClickjack">body{display:none !important;}</style>
<script type="text/javascript">
if (self === top) {
var antiClickjack = document.getElementById("antiClickjack");
antiClickjack.parentNode.removeChild(antiClickjack);
} else {
top.location = self.location;
}
</script>
```

Retest:

Bug ID#5

Missing Input Validation in Wallet Name

Vulnerability Type

Improper Input Validation - [CWE-20]

Severity

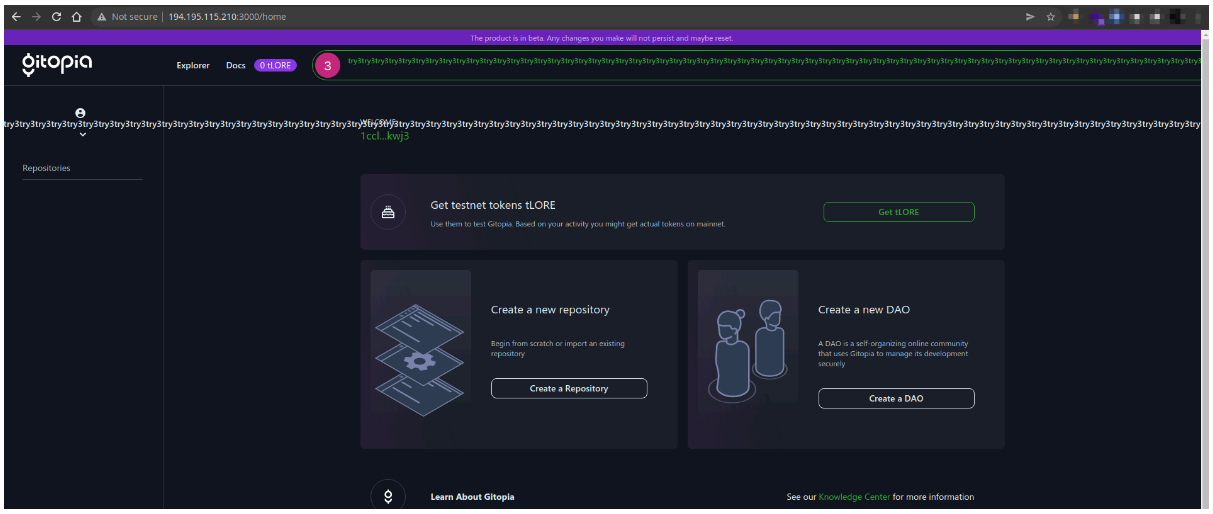
Low

Description:

Gitopia web does not have any input validations on the Wallet name when creating wallets. This may be abused to have wallet names with unlimited characters leading to deformation of the web page after wallet creation.

PoC:

1. Go to the Website and create a wallet.
2. Keep a long wallet name as shown below -


Not secure | 194.195.115.210:3000/home
The product is in beta. Any changes you make will not persist and maybe reset.
gitopia
Explorer
Docs
0 tLORE
3
1ccl...kwj3
Repositories
Get testnet tokens tLORE
Use them to test Gitopia. Based on your activity you might get actual tokens on mainnet.
Get tLORE
Create a new repository
Begin from scratch or import an existing repository
Create a Repository
Create a new DAO
A DAO is a self-organizing online community that uses Gitopia to manage its development securely
Create a DAO
Learn About Gitopia
See our Knowledge Center for more information

Impacts

Missing input validation may lead to multiple issues such as Client-side Denial of Services and unexpected results on the pages.

Remediation
Have a client and a server-side input validation on all the input parameters.

Retest:

Bug ID#6

Large File Denial Of Service

Vulnerability Type
Denial of service - [CWE-400]

Severity
High

Description:
The website is not truncating very large files when viewing them. This leads to the page being unresponsive and denial of service. This also leads the server to crash in some cases exhausting the javascript heap memory.

PoC:

- Upload a very large file containing multiple lines. We uploaded a file of size 135 MB containing 9544235 lines of text.
- After the code is pushed, open the file in the web browser and you'll notice that the website will become unresponsive and the server will crash.

Impacts:
This vulnerability will affect the availability of the website and will lead to downtime if a large file is uploaded. This can also be abused to take down other users by sending pull requests to their repositories.

Remediation:
Truncate the number of lines when viewing a large file as done by GitHub so as to not show everything directly.

Retest:

6. Disclosure

The Reports provided by CredShields is not an endorsement or condemnation of any specific project or team and do not guarantee the security of any specific project. The contents of this report are not intended to be used to make decisions about buying or selling tokens, products, services, or any other assets and should not be interpreted as such.

Emerging technologies such as Web3 carry a high level of technical risk and uncertainty. CredShields does not provide any warranty or representation about the quality of code, the business model or the proprietors of any such business model, or the legal compliance of any business. The report is not intended to be used as investment advice and should not be relied upon as such.

CredShields Audit team is not responsible for any decisions or actions taken by any third party based on the report.