VIBRANIUM AUDITS

Security Assessment

GIdt

Verified by Vibranium Audits on 14 October 2024

Vibranium Audits Verified on October 14th, 2024

Gldt

The security assessment was prepared by Vibranium Audits.

Executive Summary

TYPES	ECOSYSTEM	METHODS
DEFI/NFT	ICP	Manual Review, penetration testing
LANGUAGE	**TIMELINE**	**KEY COMPONENTS**
Rust	Delivered on 14/10/2024	N/A
CODEBASE		**COMMITS**
N/A		N/A

2

Vulnerability Summary

9	0	0	0	9	0	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined	Unresolved

0	Critical		Critical vulnerabilities are usually straightforward to exploit and can lead to the loss of user funds or contract state manipulation by external or internal actors.
1	High	0 Resolved	High vulnerabilities are usually harder to exploit, requiring specific conditions, or have a more limited scope, but can still lead to the loss of user funds or contract state manipulation by external or internal actors.
2	Medium	0 Resolved	Medium vulnerabilities are usually limited to state manipulations, but cannot lead to assets loss. Major deviations from best practices are also in this category.
2	Low	0 Resolved	Low vulnerabilities are related to outdated and unused code or minor gas optimization. These issues won't have a significant impact on code execution, but affect the code quality.
1	Informational	0 Resolved	Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

CODEBASE | GLDT

Repository

N/A

Commits

N/A

AUDIT SCOPE | GLDT

1 repo audited ● 1 repo with Acknowledged findings ● 0 files with Resolved findings

ID			Branch	Commit Hash
● VGF			N/A	N/A

APPROACH & METHODS | GLDT

This report has been prepared for GLDT(2024) to discover issues and vulnerabilities in the source code of the GLDT project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Manual Review, rigorous Penetration Testing and Static Analysis techniques.

The auditing process pays special attention to the following considerations:

- Pen-Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the code base to ensure compliance with current best practices and industry standards
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire code base by industry experts.

The security assessment resulted in findings that ranged from critical to informational We recommend addressing these findings to ensure a high level of security standards and industry practices.
We suggest recommendations that could better serve the project from the security perspective:

- Testing the smart contracts against both common and uncommon attack vectors.
- Enhance general coding practices for better structures of source codes.
- Review unit tests to cover the possible use cases.
- Review functions for readability, especially for future development work.

TABLE OF CONTENTS | GLDT

FINDINGS | GLDT

9	1	5	2	1
Total Findings	High	Medium	Low	Informational

This report has been prepared to discover issues and vulnerabilities for GLDT.
Through this audit, we have uncovered 9 issues ranging from different severity levels.
Utilizing the techniques of Manual Review & Penetration Testing, we discovered the following findings:

ID	Title	Category	Severity	Status
VGF-01	Unbounded Task Retry	Logical Issue	High	Acknowledged
VGF-02	Unbounded Input Size for Swap Details	Logical Issue	Medium	Acknowledged
VGF-03	Inefficient Use of unwrap_or	Best Practices	Informational	Acknowledged
VGF-04	Panic on unwrap()	Best Practices	Low	Acknowledged
VGF-05	Missing Error Handling	Logical Issue	Low	Acknowledged
VGF-06	Unbounded Vector for Errors	Logical Issue	Medium	Acknowledged
VGF-07	Lack of Error Handling for insufficient Cycle Balance	Logical Issue	Medium	Acknowledged
VGF-08	Lack of Error Logging for Failed retry_async Operations	Logical Issue	Medium	Acknowledged
VGF-09	Unwrap Risk	Logical Issue	Medium	Acknowledged

VGF-01 | Unbounded Task Retry with retry_async

Category	Severity	Location	Status
Logical Issue	HIGH	/impl/src/updates/swap_tokens_for_nft.rs	Acknowledged

Description

The retry_async function is used to retry operations like icrc1_transfer or sale_nft_origyn. While retrying operations is useful, there is no backoff mechanism or limit to the number of retries, which can lead to excessive load on the system if the error is persistent. Location:

- retry_async(|| icrc1_transfer(gldt_canister_id, &args), 3).await

Recommendation

Implement an exponential backoff strategy and limit the number of retries to prevent potential system overload:

```
use tokio::time::sleep;
use std::time::Duration;

for i in 0..3 {
    match icrc1_transfer(gldt_canister_id, &args).await {
        Ok(result) => return Ok(result),
        Err(e) => {
            sleep(Duration::from_secs(2u64.pow(i))).await; // Exponential backoff
        }
    }
}
Err("Retry failed")
```

VGF-02 | Unbounded Input Size for Swap Details

Category	Severity	Location	Status
Logical Issue	MEDIUM	common/src/types/swap.rs	Acknowledged

Description

Swap detail structures like SwapDetailForward and SwapDetailReverse have no constraints on the size of their fields such as nft_id_string and sale_id. This could lead to unbounded input, resulting in large memory consumption or even resource exhaustion attacks.
Location:

- SwapDetailForward::nft_id_string
- SwapDetailReverse::nft_id_string

Recommendation

Limit the length of strings to a reasonable size:

```
fn update_sale_id(&mut self, sale_id: String) {
    if sale_id.len() > MAX_SALE_ID_LENGTH {
        panic!("Sale ID exceeds maximum length");
    }
    self.sale_id = sale_id;
}
```

VGF-03 | Inefficient Use of unwrap_or in add_swap_index_for_user

Category	Severity	Location	Status
Logical Issue	INFORMATIONAL	model/archive.rs	Acknowledged

Description

The unwrap_or(VecNat::default()) in add_swap_index_for_user could be simplified by using unwrap_or_default(), which directly calls the Default implementation.
Location:

- add_swap_index_for_user: let mut indexes = self.user_swap_id_map.get(&user).unwrap_or(VecNat::default());

Recommendation

Simplify the code by using unwrap_or_default():

```
let mut indexes = self.user_swap_id_map.get(&user).unwrap_or_default();
```

VGF-04 | Panic on unwrap() in VecNat::to_bytes and VecNat::from_bytes

Category	Severity	Location	Status
Logical Issue	LOW	model/archive.rs common/src/types/swap.rs	Acknowledged

Description

he use of unwrap() in the to_bytes and from_bytes methods introduces the possibility of runtime panics. If encoding or decoding fails, the program will panic and crash. While this is unlikely in normal circumstances, it becomes a risk if the data gets corrupted or if a malicious user manipulates it.
Location:

- to_bytes: Cow::Owned(Encode!(self).unwrap())
- from_bytes: Decode!(&bytes, Self).unwrap()
- SwapId::to_bytes: Cow::Owned(Encode!(self).unwrap())
- SwapId::from_bytes: Decode!(&bytes, Self).unwrap()

Recommendation

Use error handling with Result instead of unwrap(). This will allow the program to handle failures gracefully without panicking:

```
fn to_bytes(&self) -> Result<Cow<[u8]>, EncodeError> {
    Encode!(self).map(Cow::Owned)
}

fn from_bytes(bytes: Cow<[u8]>) -> Result<Self, DecodeError> {
    Decode!(&bytes, Self)
}
```

VGF-05 | Missing Error Handling for stable_size

Category	Severity	Location	Status
Logical Issue	low	model/archive.rs	Acknowledged

Description

In the get_archive_size_bytes method, the stable_size() function could potentially return an error (although rare). There's no error handling for this case.
Location:

- get_archive_size_bytes: let num_pages = stable_size();

Recommendation

Check for errors when calling stable_size() and handle them appropriately:

```
let num_pages = stable_size().unwrap_or(0);
```

VGF-06 | Unbounded Vector for Errors

Category	Severity	Location	Status
Logical Issue	MEDIUM	common/src/types/swap.rs	Acknowledged

Description

In the SwapErrorReverse::NftValidationFailed, the vector Vec<NftValidationError> could grow without bounds, leading to memory exhaustion if a large number of errors are recorded.
Location:

- SwapErrorReverse::NftValidationFailed(Vec<NftValidationError>)

Recommendation

Limit the size of the vector to prevent resource exhaustion:

```
if errors.len() > MAX_VALIDATION_ERRORS {
    panic!("Too many validation errors");
}
```

VGF-07 | Lack of Error Handling for Insufficient Cycle Balance

Categor y	Severity	Location	Status
Logical Issue	MEDIUM	/impl/src/jobs/manage_stale_swaps.rs	Acknowledged

Description

In the function handle_archive_canister_cycles, if the canister's cycle balance is less than the required base (swap_canister_required_base), the function returns silently without logging or any indication of why it failed to proceed. This makes debugging and monitoring difficult since no feedback is provided about why the transfer did not occur.
Location:

- handle_archive_canister_cycles: if this_canister_cycle_balance < swap_canister_required_base { return (); }

Recommendation

Log an informative message when the canister's cycle balance is insufficient:

```
if this_canister_cycle_balance < swap_canister_required_base {
    debug!("Insufficient cycles: {this_canister_cycle_balance}, required:
{swap_canister_required_base}");
    return ();
}
```

VGF-08 | Lack of Error Logging for Failed retry_async Operations

Category	Severity	Location	Status
Logical Issue	MEDIUM	/impl/src/updates/swap_tokens_for_nft.rs	Acknowledged

Description

In retry_async, if all retries fail, the error is logged at the debug level. Given the critical nature of transferring tokens and minting NFTs, this should be logged at a higher severity level (e.g., error or warn) to ensure that these failures are visible in production.
Location:

- debug!("FORWARD SWAP :: mint :: error :: {msg}");

Recommendation

Change the log level to warn or error when a critical operation fails after retries:

```
error!("FORWARD SWAP :: mint :: error :: {msg}");
```

VGF-09 | Unwrap Risk in validate_nft_escrow_subaccount

Category	Severity	Location	Status
Logical Issue	MEDIUM	/impl/src/updates/swap_tokens_for_nft.rs	Acknowledged

Description

The validate_nft_escrow_subaccount function relies on try_into() to convert a slice into a fixed-size array. If this conversion fails (e.g., the slice is not of the expected size), it returns an error. However, the error handling could be more explicit, and the use of unwrap() in other parts of the system could lead to a panic if misused similarly.
Location:

- validate_nft_escrow_subaccount: args.escrow_info.account.sub_account.as_slice().try_into()

Recommendation

Ensure that the slice size is validated before calling try_into(). This prevents unexpected panics:

```
let sub_account_slice = args.escrow_info.account.sub_account.as_slice();
if sub_account_slice.len() == 32 {
    Ok(sub_account_slice.try_into().unwrap())
} else {
    Err(NotificationError::InvalidEscrowSubaccount("Invalid subaccount size".to_string()))
}
```

DISCLAIMER | VIBRANIUM AUDITS

This report is subject to the terms and conditions (including without limitation description of services confidentiality disclaimer and limitation of liability) set forth in the Services Agreement or the scope of services and terms and conditions provided to you (“Customer” or the “Company”) in connection with the Agreement This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement This report may not be transmitted disclosed referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Vibranium Audits prior written consent in each instance

This report is not nor should be considered an “endorsement” or “disapproval” of any particular project or team This report is not nor should be considered an indication of the economics or value of any “product” or “asset”created by any team or project that contracts Vibranium Audits to perform a security assessment This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed nor do they provide any indication of the technologies proprietors business business model or legal compliance

This report should not be used in any way to make decisions around investment or involvement with any particular project This report in no way provides investment advice nor should be leveraged as investment advice of any sort This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology

Blockchain technology and cryptographic assets present a high level of ongoing risk Vibranium Audits position is that each company and individual are responsible for their own due diligence and continuous security Vibranium Audits goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze

The assessment services provided by Vibranium Audits is subject to dependencies and under continuing development You Agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty The assessment reports could include false positives false negatives and other unpredictable results The services may access and depend upon multiple layers of third-parties

ALL SERVICES THE LABELS THE ASSESSMENT REPORT WORK PRODUCT OR OTHER MATERIALS OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED “AS IS” AND “AS AVAILABLE” AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW VIBRANIUM AUDITS HEREBY DISCLAIMS ALL WARRANTIES WHETHER EXPRESS IMPLIED STATUTORY OR OTHERWISE WITH RESPECT TO THE SERVICES ASSESSMENT REPORT OR OTHER MATERIALS WITHOUT LIMITING THE FOREGOING VIBRANIUM AUDITS SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY FITNESS FOR A PARTICULAR PURPOSE TITLE AND NON-INFRINGEMENT AND ALL WARRANTIES ARISING FROM COURSE OF DEALING USAGE OR TRADE PRACTICE WITHOUT LIMITING THE FOREGOING VIBRANIUM AUDITS MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES THE LABELS THE ASSESSMENT REPORT WORK PRODUCT OR OTHER MATERIALS OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF WILL MEET CUSTOMER’S OR ANOTHER PERSON’S REQUIREMENTS ACHIEVE ANY INTENDED RESULT BE COMPATIBLE OR WORK WITH ANY SOFTWARE SYSTEM OR OTHER SERVICES OR BE SECURE ACCURATE COMPLETE FREE OF HARMFUL CODE

OR ERROR-FREE WITHOUT LIMITATION TO THE FORGOING, VIBRANIUM AUDITS PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KINDTHAT WILL MEET CUSTOMER' S REQUIREMENTS ACHIEVE ANY INTENDED RESULTS BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE APPLICATIONS SYSTEMS OR SERVICES OPERATE WITHOUT INTERRUPTION MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED

WITHOUT LIMITING THE FOREGOING NEITHER VIBRANIUM AUDITS NOR ANY OF VIBRANIUM AUDITS AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND EXPRESS OR IMPLIED AS TO THE ACCURACY RELIABILITY ORCURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE VIBRANIUM AUDITS WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS MISTAKES OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE OF ANY NATURE WHATSOEVER RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES ASSESSMENT REPORT OR OTHER MATERIALS

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNEROR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS

THE SERVICES ASSESSMENT REPORT AND ANY OTHER MATERIALS HERE UNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT NOR MAY COPIES BE DELIVERED TO ANY OTHER PERSON WITHOUT VIBRANIUM AUDITS PRIOR WRITTEN CONSENT IN EACH INSTANCE

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES ASSESSMENT REPORT AND ANY ACCOMPANYING MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST VIBRANIUM AUDITS WITH RESPECT TO SUCH SERVICES ASSESSMENT REPORT AND ANY ACCOMPANYING MATERIALS

THE REPRESENTATIONS AND WARRANTIES OF VIBRANIUM AUDITS CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER ACCORDINGLY NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST VIBRANIUM AUDITS WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE

FOR AVOIDANCE OF DOUBT THE SERVICES INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS ORMATERIALS SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

Vibranium | Securing the Web3 World

Vibranium Audits is a blockchain security company that was founded in 2021 by professors from the University of Greenwich and cyber-security engineers from ITI Capital. As pioneers in the field, Vibranium Audits utilizes best-in-class Formal Verification and AI technology to secure and monitor blockchains, smart contracts, and Web3 apps.