CryptoGen Nepal

#MADE4SECURITY

Smart contract Security Review

Final Report

MSSP Alert TOP 250 | ICT AWARD STARTUP CATEGORY

Prepared For:

ComSolBridge

EST. 2019

ISO 27001:2013 COMPLIANT COMPANY

Document Control

Document Name	Final Report of Smart Contract Security Review ComSolBridge
Abstract	This document details the approaches and vulnerabilities identified in the smart contracts of "ComSolBridge" from a security perspective.
Security Classification	Confidential
Location	ComSolBridge, Philadelphia, USA

Authorization

Document Owner	**Reviewed by**	**Authorized by**
CryptoGen Nepal	Aayushman Thapa Magar	Nirmal Dahal

Amendment Log

Version	**Modification Date**	**Section**	**Amendment/ Modification/ Deletion**	**Description of change**
1.0	April 16, 2024	-	-	Final Report
1.1	April 19, 2024	Executive Summary, Vulnerability list, findings	Amendment	Reverification

Distribution list	
Name	**Designation**
Satish Tamrakar	Project Manager, Cryptogen Nepal Pvt. Ltd.
Krishna Dahal	Business Lead, ComSolBridge.

Date: April 16, 2024

To,
Krishna Dahal,
ComSolBridge,
Philadelphia, USA

Dear Sir,

We hereby submit to you the final report of the smart contract security review of ComSolBridge. The security assessment was carried out from **April 14, 2024**, to **April 16, 2024**. The report includes an executive summary, vulnerability summary, and findings with technical details. We believe that the evidence from our analysis provides a reasonable basis for our conclusions and findings regarding the security review objectives and scope. Reverification testing was conducted on **April 19, 2024**. All identified issues were remediated.

We want to express our appreciation to ComSolBridge for being courteous, helpful, and professional, without which completing the Security Assessment would be difficult.

Regards,
Nirmal Dahal
Chief Technology Officer
CryptoGen Nepal Pvt. Ltd.
nirmal.dahal@cryptogennepal.com
+977 9801128467

Table of Contents	Page

Executive Summary

CryptoGen Nepal was engaged by ComSolBridge to security review on organizations' smart contracts. As a result, we found different vulnerabilities in the smart contracts of ComSolBridge. The review was conducted using automated tools and manual code review.

The purpose of this security review was to identify the security vulnerabilities in the smart contracts and suggest the best recommendation for it. We found **One Medium** and **One low vulnerability** in the smart contracts. The identified issues have been remediated successfully.

We conclude the review with security posture as below:

Scope	Overall Security Posture	Comments
Solana Programs (Rust)	**None**	All identified issues have been remediated.

Note: Overall security posture changed from low to none after reverification testing.

Project Overview

The ComSol Bridge smart contracts are meant to act as a bridge program between Commuine AI and Solana. The program is developed in Rust with the anchor framework. The program contains an admin role which is required to perform actions such as miniting, burning, changing the configuration of tokens, and pausing the contract. The admin privileges can be transferred to another account if required by the current admin.

Methodology

A comprehensive examination of the ComSolBridge contract was performed by the CryptoGen team. The team consists of security professionals with extensive experience in offensive security and smart contract security. The following points were paid close attention to:

- Common Solana contract vulnerabilities and anti-patterns, such as:
    - Missing ownership checks
    - Missing signer checks
    - Signed invocation of unverified programs
    - Solana account confusions
    - Re-initiation with cross-instance confusion
    - Missing freeze authority checks
    - Insufficient SPL-Token account verification
    - Missing rent exemption assertion
    - Casting truncation
    - Arithmetic overflows or underflows
    - Numerical precision errors
- Checking for unsafe design which might lead to common vulnerabilities being introduced in the future
- Ensuring that the contract logic correctly implements the project specifications
- Examining the code in detail for contract-specific low-level vulnerabilities
- Ruling out denial of service attacks
- Ruling out economic attacks
- Checking for instructions that allow front-running or sandwiching attacks
- Checking for rug-pull mechanisms or hidden backdoor

Scope

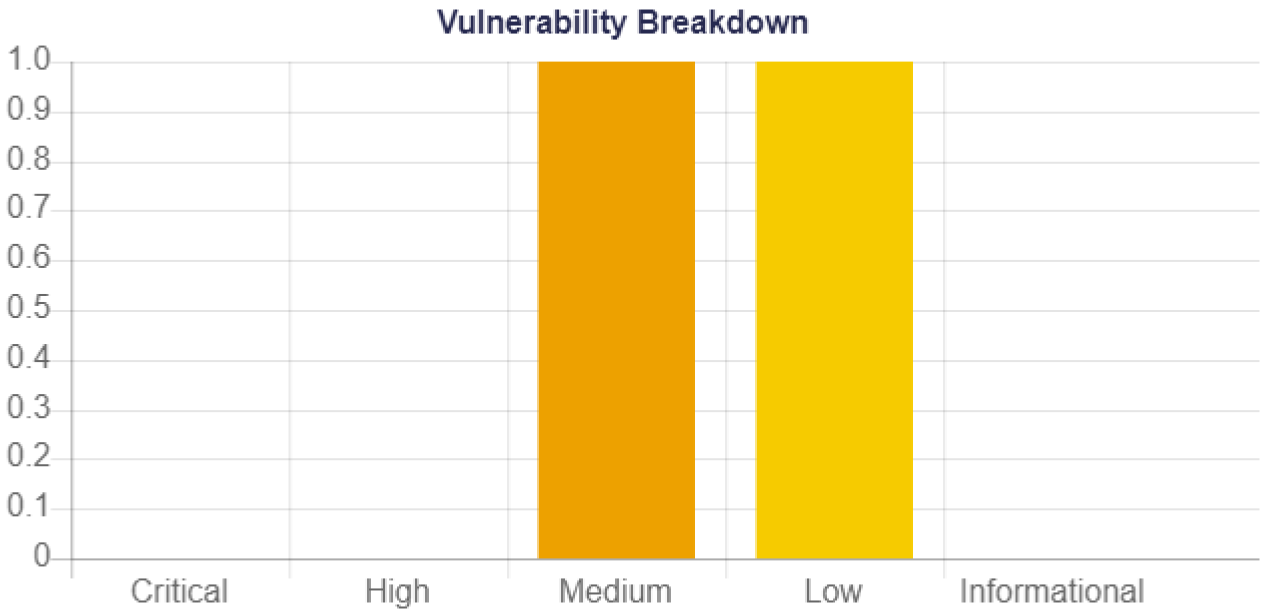
- Smart contracts (Solana – Rust)
    - https://github.com/shrestha-roshan/comsol-bridge
    - ec2a2c3f9f1ffcdb904b95a38c4073cd4de41f69

Sn.	File	LoC
1.	programs\bridge-solana\src\instructions\burn_token.rs	58
2.	programs\bridge-solana\src\instructions\emergency_pause.rs	23
3.	programs\bridge-solana\src\instructions\init_config.rs	41
4.	programs\bridge-solana\src\instructions\mint_token.rs	95
5.	programs\bridge-solana\src\instructions\mod.rs	12
6.	programs\bridge-solana\src\instructions\update_admin.rs	23
7.	programs\bridge-solana\src\instructions\update_token_config.rs	35
8.	programs\bridge-solana\src\state\bridge.rs	12
9.	programs\bridge-solana\src\state\mod.rs	2
10.	programs\bridge-solana\src\error.rs	14
11.	programs\bridge-solana\src\lib.rs	35
Total Lines of Code		350

Note: Comments and blank lines excluded.

Vulnerability Summary

The charts below are designed to provide a quick snapshot of the assessment. For information regarding risk ratings. Please refer to the Findings section for more details.



Vulnerability List

Vulnerability	Severity	Identified Date	Status
No transfer ownership pattern when updating admin	Medium	April 14, 2024	Remediated
Program can be initialized by anyone	Low	April 15, 2024	Remediated

Findings

No transfer ownership pattern when updating admin	Remediated

Description

Currently, the admin role transfer process involves the current admin calling ***update_admin()*** function. This function checks if the caller is the current admin and sets the provided account as the new admin. No confirmation from the new admin is required.

Impact

Admin privileges for the current admin are immediately revoked when the new admin is selected. If the new admin is an invalid account, then administrative privileges are lost.

Recommendation

A two-step process is recommended:

- Current admin provides the new account as a “candidate” for new admin.
- The “candidate“ calls the function again to accept the role.

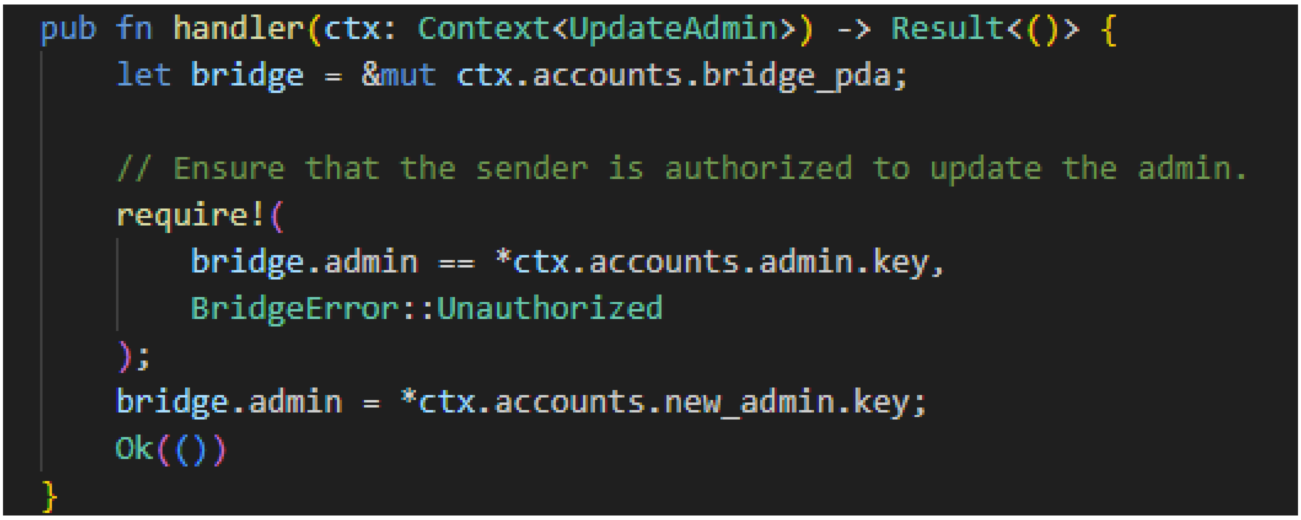
Only if the caller is the candidate, then administrative privileges are transferred.

Status

Remediated (Previously medium)

Code Snip

- programs\bridge-solana\src\instructions\update_admin.rs#L18-L28



```
pub fn handler(ctx: Context<UpdateAdmin>) -> Result<()> {
    let bridge = &mut ctx.accounts.bridge_pda;

    // Ensure that the sender is authorized to update the admin.
    require!(
        bridge.admin == *ctx.accounts.admin.key,
        BridgeError::Unauthorized
    );
    bridge.admin = *ctx.accounts.new_admin.key;
    Ok(())
}
```

References

- [A more secure ownership transfer pattern - HackMD]

Client remarks
Updating the program now requires both the old_admin and the new_admin to sign a transaction, adding an extra layer of security o prevent the use of invalid accounts.

Program can be initialized by anyone	**Remediated**

Description

The ***init_config()*** function is used to set the initialization values such as admin, fee, etc. After deployment, anyone can call this function with appropriate parameters to initialize the program.

Impact

If unauthorized accounts initialize the program, then the program may be unusable.

Recommendation

It is recommended to:

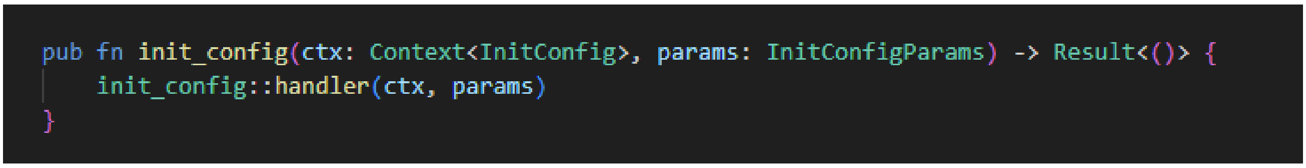
- Deploy and initialize the program in a single transaction.
- Use a PDA to set the initialization parameters.
- Implement access control mechanism to ensure only authorized accounts can call the ***init_config()*** function.

Status

Remediated (Previously low)

Code Snip

- programs\bridge-solana\src\lib.rs#L24-L26



```rust
pub fn init_config(ctx: Context<InitConfig>, params: InitConfigParams) -> Result<()> {
    init_config::handler(ctx, params)
}
```

References

- [Initializing Accounts in Solana and Anchor (rareskills.io)]

Client remarks
The initialization process is a one-time operation that will be executed in a single transaction during deployment.

Appendix

OWASP Risk Rating Methodology

The OWASP Risk Rating Methodology considers several risk factors when assessing the security posture of a web application. These factors include the likelihood of an attack occurring, the impact or potential harm resulting from a successful attack, and the prevalence or exposure of the application.

- Likelihood: Likelihood refers to the probability that a specific threat will exploit a vulnerability. It considers factors such as the existence of mitigating controls and the motivation and capability of potential attackers.
- Impact: Impact assesses the potential damage that would result from the successful exploitation of a vulnerability. This can include financial losses, damage to reputation, loss of sensitive data, and regulatory penalties.

OVERALL RISK SEVERITY				
IMPACT	High	Medium	High	Critical
	Medium	Low	Medium	High
	Low	Info	Low	Medium
		Low	**Medium**	**High**
	Likelihood			

OUR SERVICES

Our Services As Information Security Company Includes:

- SECURITY OPERATIONS CENTER
- INFORMATION SECURITY AUDIT
- SWIFT CSP ASSESSMENT
- DARKWEB MONITORING & BRAND PROTECTION
- VULNERABILITY MANAGEMENT
- PENETRATION TESTING
- INCIDENT RESPONSE
- THREAT ANALYSIS
- SERVER CONFIGURATION ASSESSMENT
- CYBER SECURITY CONSULTANT
- INFORMATION SECURITY TRAINING

CryptoGen Nepal

/cryptogennepal
www.cryptogennepal.com
+977-1-4528928
whois@cryptogennepal.com