Safe Edges × griffy

SMART CONTRACT AUDIT REPORT FOR Griffy Market

CONTENTS

ABOUT SAFE EDGES

Safe Edges is a cybersecurity company that strives to elevate the standards of security in Web 3.0, create a safer environment for users, and ensure mass Web 3.0 adoption.

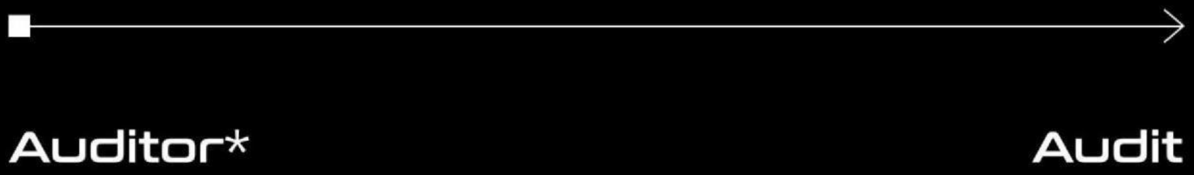
Safe Edges has multiple top-notch auditing teams specialized in different fields of information security, showing extreme performance in the most challenging and technically complex tasks, including but not limited to: Infrastructure Audits, Zero Knowledge Proofs / Novel Cryptography, DeFi and NFTs. Safe Edges not only uses widely known methodologies and flows, but focuses on discovering and introducing new ones on a day-to-day basis.

Since Safe Edges was founded in 2023, it has had an impressive track record and recognition in the industry: Piyush shukla - CISO

METHODOLOGY

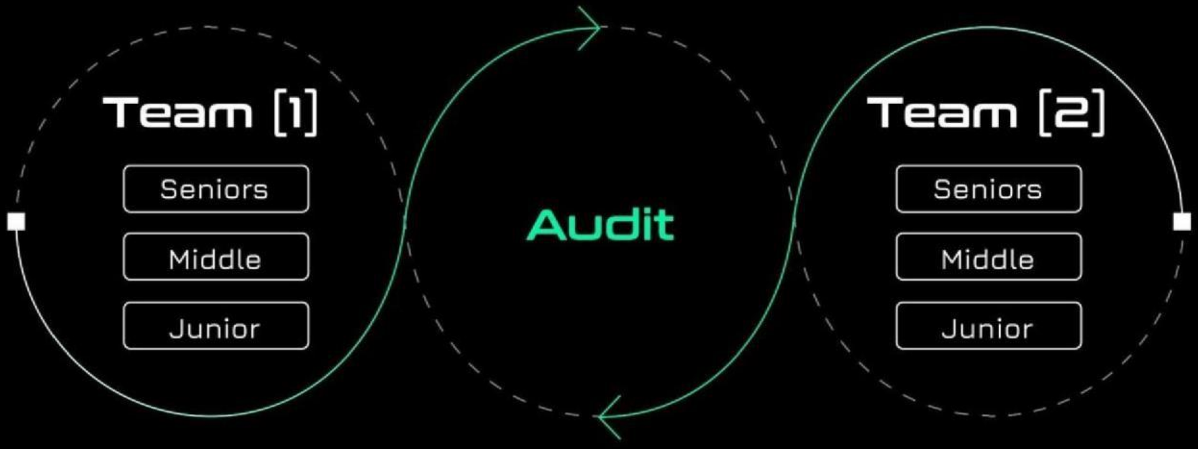
COMMON AUDIT PROCESS

Companies often assign just one engineer to one security assessment with no specified level. Despite the possible impeccable skills of the assigned engineer, it carries risks of the human factor that can affect the product's lifecycle.



SAFE EDGES METHODOLOGY

Safe Edges methodology involves 2 teams, including multiple auditors of different seniority, with at least 5 security engineers. This unique cross-checking mechanism helps us provide the best quality in the market.



SEVERITY STRUCTURE

The vulnerability severity is calculated based on two components

- Impact of the vulnerability
- Probability of the vulnerability

IMPACT	PROBABILITY			
	Rare	Unlikely	Likely	Very Likely
Low / Info	Low / Info	Low / Info	Medium	Medium
Medium	Low / Info	Medium	Medium	High
High	Medium	Medium	High	Critical
Critical	Medium	High	Critical	Critical

SEVERITY CHARACTERISTICS

Smart contract vulnerabilities can range in severity and impact, and it's important to understand their level of severity in order to prioritize their resolution. Here are the different types of severity levels of smart contract vulnerabilities:

CRITICAL

Vulnerabilities with this level of severity can result in significant financial losses or reputational damage. They often allow an attacker to gain complete control of a contract, directly steal or freeze funds from the contract or users, or permanently block the functionality of a protocol. Examples include infinite mints and governance manipulation.

HIGH

Vulnerabilities with this level of severity can result in some financial losses or reputational damage. They often allow an attacker to directly steal yield from the contract or users, or temporarily freeze funds. Examples include inadequate access control integer overflow/underflow, or logic bugs.

MEDIUM

Vulnerabilities with this level of severity can result in some damage to the protocol or users, without profit for the attacker. They often allow an attacker to exploit a contract to cause harm, but the impact may be limited, such as temporarily blocking the functionality of the protocol. Examples include uninitialized storage pointers and failure to check external calls.

LOW

Vulnerabilities with this level of severity may not result in financial losses or significant harm. They may, however, impact the usability or reliability of a contract. Examples include slippage and front-running, or minor logic bugs.

INFORMATIONAL

Vulnerabilities with this level of severity are regarding gas optimizations and code style. They often involve issues with documentation, incorrect usage of EIP standards, best practices for saving gas, or the overall design of a contract. Examples include not conforming to ERC20, or disagreement between documentation and code.

It's important to consider all types of vulnerabilities, including informational ones, when assessing the security of smart contracts. A comprehensive security audit should consider all types of vulnerabilities to ensure the highest level of security and reliability.

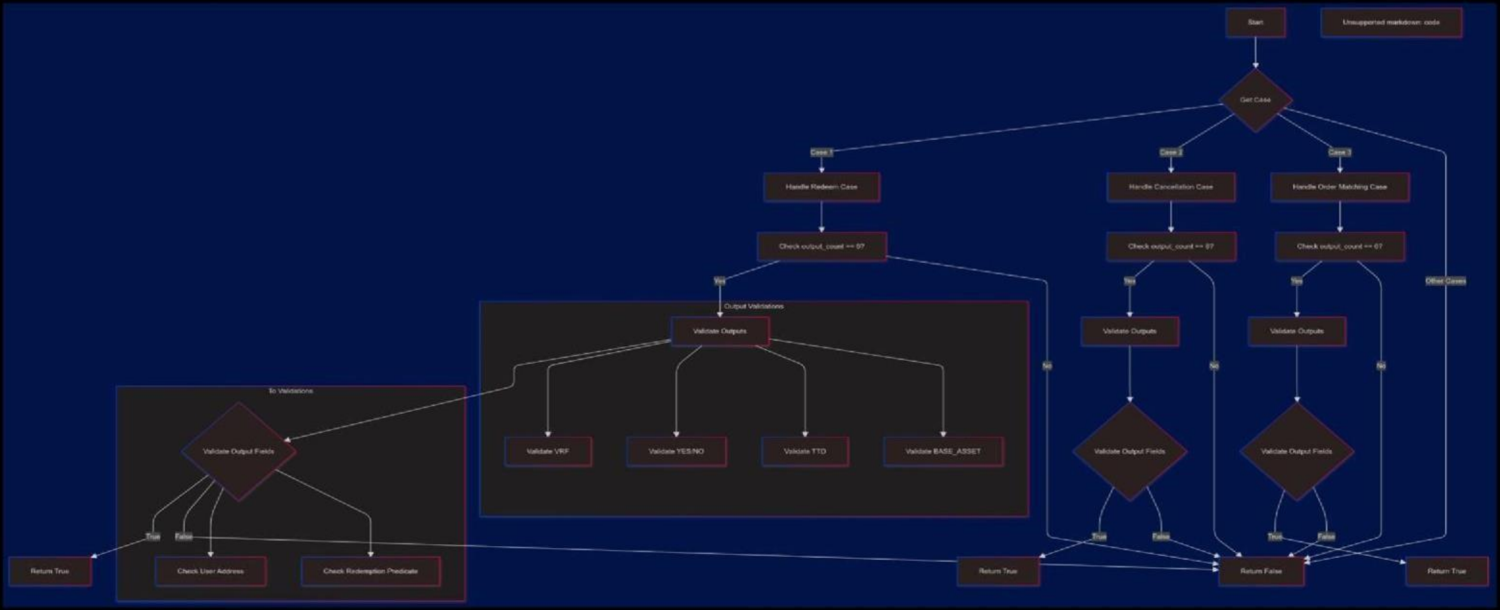
EXECUTIVE SUMMARY

OVERVIEW

Griffy is a decentralized prediction market platform built on the Fuel chain, designed to harness the power of blockchain technology for innovative forecasting. In decentralized prediction markets like Griffy, users can buy and sell prediction shares on the outcomes of various events—ranging from elections and sports to market trends. This is akin to trading in a stock market, where the share price reflects the collective probability of an event's outcome.

By leveraging the scalability and high performance of the Fuel chain, Griffy ensures fast and efficient transactions, providing a seamless experience for users. The platform's decentralized nature, secured through smart contracts on the blockchain, guarantees transparency, trust, and censorship resistance, eliminating the need for a central authority.

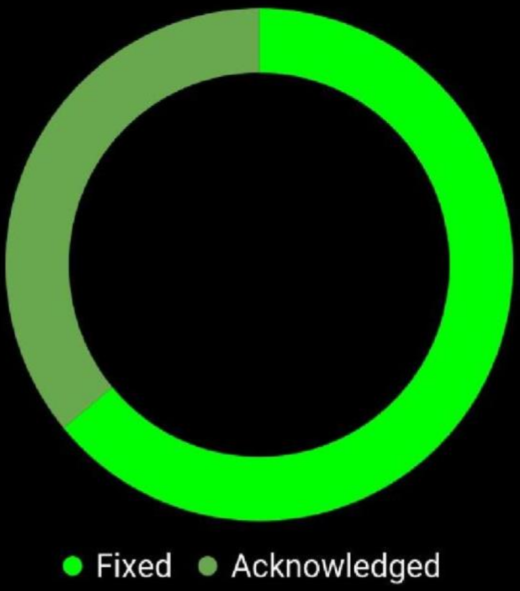
Griffy rewards participants for accurate predictions, creating financial incentives that align with honest and well-informed forecasting. As decentralized prediction markets evolve, and with the Fuel chain's advanced capabilities, Griffy aims to set a new standard for harnessing collective intelligence and offering a fair, open, and efficient platform for predictive insights.



SUMMARY

SEVERITY	NUMBER OF FINDINGS
CRITICAL	0
HIGH	0
MEDIUM	2
LOW	0
INFORMATIONAL	1

STATUS



WEAKNESSES

This section contains the list of discovered weaknesses.

F-1. Missing validation in Commission Calculation

SEVERITY: Medium

REMEDIATION: See DESCRIPTION

STATUS: fixed, commits

DESCRIPTION:

The variable COMMISSION_PERCENTAGE is currently defined as u64 = 0. This likely implies that it will be set later:

```
COMMISSION_PERCENTAGE: u64 = 0,
```

There is no explicit maximum limit set for the COMMISSION_PERCENTAGE. This can lead to potential issues where, if not properly validated, the commission could reach unreasonable or unintended levels, such as 100%.

Recommendation:

Add a Maximum Commission Check: Implement a validation that ensures COMMISSION_PERCENTAGE does not exceed a reasonable threshold, such as 20% to 30%, depending on the business logic

F-2. Missing validation in FEES Calculation

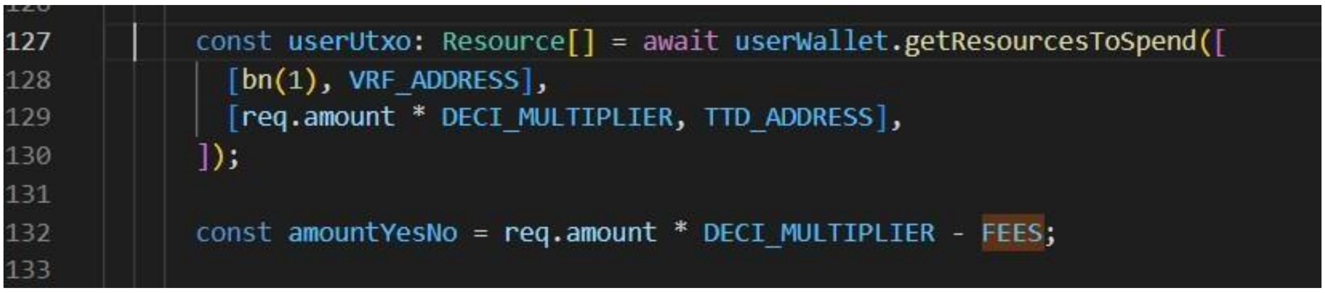
SEVERITY: Medium

REMEDIATION: see description

STATUS: fixed

DESCRIPTION:

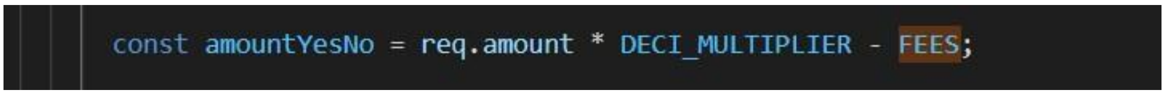
There is a missing validation check in the code for scenarios where the entire requested amount is consumed by the fees. Specifically, the current logic includes a check for:



```
const userUtxo: Resource[] = await userWallet.getResourcesToSpend([
  [bn(1), VRF_ADDRESS],
  [req.amount * DECI_MULTIPLIER, TTD_ADDRESS],
]);

const amountYesNo = req.amount * DECI_MULTIPLIER - FEES;
```

However, no additional check exists for cases where the result of:



```
const amountYesNo = req.amount * DECI_MULTIPLIER - FEES;
```

becomes zero. In this case, if req.amount * DECI_MULTIPLIER equals FEES, the calculated amountYesNo would become 0. This suggests that the entire requested amount is consumed by the fees, effectively leaving no remaining amount for the operation.

F-3. ZERO_B256 deprecated

SEVERITY: Informative

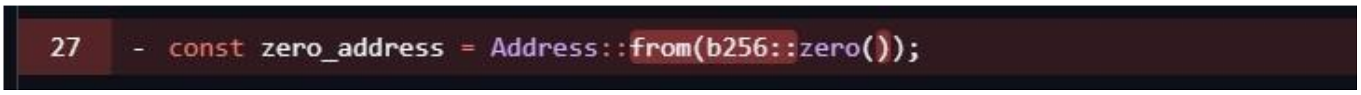
REMEDIATION: see description

STATUS: fixed

DESCRIPTION:

Use of the ZERO_B256 is deprecated as per the updated Sway language standards.

https://fuellabs.github.io/sway/master/std/constants/constant.ZERO_B256.html



```
-  const zero_address = Address::from(b256::zero());
```

Recommendation:

It is advised to use b256::zero() instead

Safe Edges