Techrate1 Techrate

TechRate

AUDIT COMPANY

PresaleMATX
Smart Contract Security Audit

TechRate

June, 2021

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and TechRate and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (TechRate) owe no duty of care towards you or any other person, nor does TechRate make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and TechRate hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, TechRate hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against TechRate, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

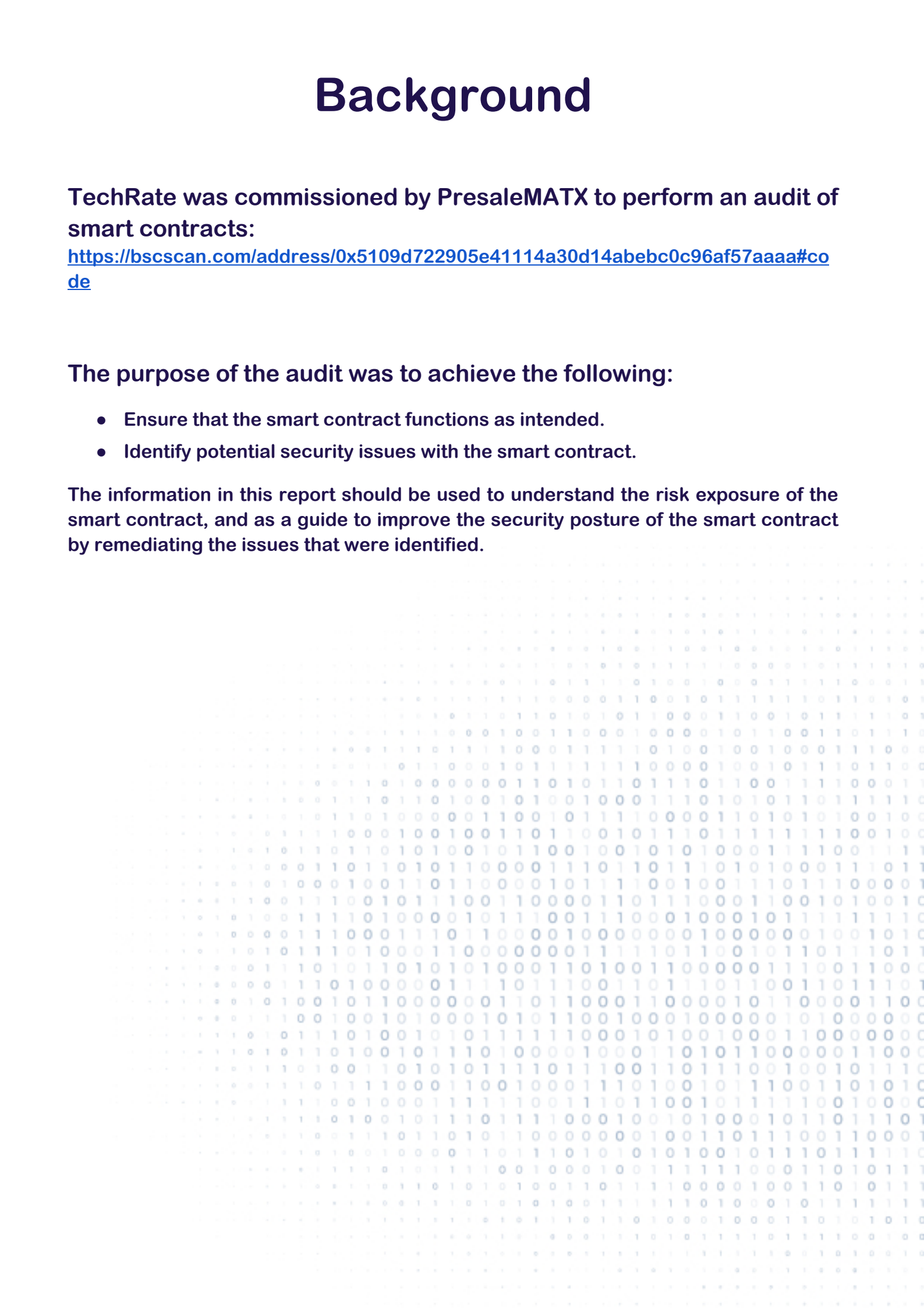
Background

TechRate was commissioned by PresaleMATX to perform an audit of smart contracts:
https://bscscan.com/address/0x5109d722905e41114a30d14abebc0c96af57aaaa#code

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.



Issues Checking Status

	Issue description	Checking status
1.	Compiler errors.	Passed
2.	Race conditions and Reentrancy. Cross-function race conditions.	Passed
3.	Possible delays in data delivery.	Passed
4.	Oracle calls.	Passed
5.	Front running.	Passed
6.	Timestamp dependence.	Passed
7.	Integer Overflow and Underflow.	Passed
8.	DoS with Revert.	Passed
9.	DoS with block gas limit.	Passed
10.	Methods execution permissions.	Passed
11.	Economy model of the contract.	Passed
12.	The impact of the exchange rate on the logic.	Passed
13.	Private user data leaks.	Passed
14.	Malicious Event log.	Passed
15.	Scoping and Declarations.	Passed
16.	Uninitialized storage pointers.	Passed
17.	Arithmetic accuracy.	Passed
18.	Design Logic.	Passed
19.	Cross-function race conditions.	Passed
20.	Safe Open Zeppelin contracts implementation and usage.	Passed
21.	Fallback function security.	Passed

Security Issues

High Severity Issues

No high severity issues found.

Medium Severity Issues

No medium severity issues found.

Low Severity Issues

1. Different types and wrong name

Issue:

- The function updatePrce and updateBNBLimit have uint type parameters but MATXPrice and BNBLimit variables typed as uint256.

```
uint256 public MATXPrice = 20000000e18; // 1 BNB == 20000000 MATX;
uint256 public BNBLimit = 10e18;
ftrace | funcSig
function updatePrce( uint _MATXPrice⬆) external onlyOwner {
    MATXPrice = _MATXPrice⬆;
}

ftrace | funcSig
function updateBNBLimit( uint _BNBLimit⬆) external onlyOwner {
    BNBLimit = _BNBLimit⬆;
}
```

- The name of function updatePrce and variable _minimumInvestment has syntax error.

Recommendation:
Use one type and change the name of the function.

2. Function name

Issue:

- The getPrice function doesn't return a price. It returns amount of token that user can for amount of BNB passed as parameter – amountIn.

Recommendation:
Change the name or logic.

Owner privileges (In the period when the owner is not renounced)

- Owner can change price of MATX, BNBLimit and Minimum investment amount of BNB.

```
function updatePrce( uint _MATXPrice⬆) external onlyOwner {
    MATXPrice = _MATXPrice⬆;
}

function updateBNBLimit( uint _BNBLimit⬆) external onlyOwner {
    BNBLimit = _BNBLimit⬆;
}

function updateMinimumInvestment( uint minimumInvestment⬆) public onlyOwner {
    _minimumIvestment = minimumInvestment⬆;
}
```

- Owner can get tokens and BNB.

```
ftrace | funcSig
function failSafe( uint _amount⬆) public onlyOwner {
    require(MATX.balanceOf(address(this)) >= _amount⬆, "failSafe :: insufficient amount");
    MATX.transfer(_msgSender(),_amount⬆);
}

ftrace | funcSig
function failSafeBNB( uint _amount⬆) public onlyOwner {
    address _contract = address(this);
    require(_contract.balance >= _amount⬆,"insufficient amount");
    msg.sender.transfer(_amount⬆);
}
```

- Owner can transfer and renounce ownership.

```
ftrace | funcSig
function renounceOwnership() public onlyOwner {
    emit OwnershipTransferred(_owner, address(0));
    _owner = address(0);
}

/**
 * @dev Transfers ownership of the contract to a new account (`newOwner`).
 * Can only be called by the current owner.
 */
ftrace | funcSig
function transferOwnership(address newOwner⬆) public onlyOwner {
    _transferOwnership(newOwner⬆);
}
```

Conclusion

Smart contracts contain low severity issues! Liquidity pair contract's security is not checked due to out of scope.

Liquidity locking details NOT provided by the team.

TechRate note:

Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by Owner.