MythX

REPORT 617109BC83789C00189043B3

Created	Thu Oct 21 2021 06:33:32 GMT+0000 (Coordinated Universal Time)
Number of analyses	1
User	614184718bfa12ce53f29d58

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
bc53519a-c4d3-4b10-a5ff-baf24006dd11	/flattenedcontracts/gausscrowdsale_flattened.sol	0

Analysis bc53519a-c4d3-4b10-a5ff-baf24006dd11 MythX

Started	Thu Oct 21 2021 06:33:33 GMT+0000 (Coordinated Universal Time)
Finished	Thu Oct 21 2021 06:33:38 GMT+0000 (Coordinated Universal Time)
Mode	Deep
Client Tool	Mythx-Vscode-Extension
Main Source File	/Flattenedcontracts/Gausscrowdsale_flattened.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	0	0

ISSUES

UNKNOWN Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
300 __Ownable_init();
301 startTime = _startTime;
302 endTime = startTime + (42 days);
303 crowdsaleWallet = _crowdsaleWallet;
304 _token = IBEP20(_gaussAddress);
```

UNKNOWN Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
303 crowdsaleWallet = _crowdsaleWallet;
304 _token = IBEP20(_gaussAddress);
305 purchaseCap = (100 * 10**18);
306 jagerRaised = 0;
307 gaussSold = 0;
```

UNKNOWN Arithmetic operation "**" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
303 crowdsaleWallet = _crowdsaleWallet;
304 _token = IBEP20(_gaussAddress);
305 purchaseCap = (100 * 10**18);
306 jagerRaised = 0;
307 gaussSold = 0;
```

UNKNOWN Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
339 require(_jagerAmount != 0, "Crowdsale: amount of BNB must be greater than 0.");
340 require(_jagerAmount <= purchaseCap, "Crowdsale: amount of BNB sent must lower than 100");
341 require((purchaseTotals[_beneficiary] + _jagerAmount) <= purchaseCap, "Crowdsale: amount of BNB entered exceeds buyers purchase cap.");
342 }
343
```

UNKNOWN Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
347
348 // Calculates the token amount using the "jagerAmount" and the rate at the current stage.
349 uint256 tokenAmount = ((_jagerAmount * rates[currentStage])/(10**18));
350 require((gaussSold + tokenAmount) <= stages[15], "Crowdsale: token amount can not be more that total amount alloted to Crowdsale.");
351
```

UNKNOWN Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
347
348 // Calculates the token amount using the "jagerAmount" and the rate at the current stage.
349 uint256 tokenAmount = ((_jagerAmount * rates[currentStage])/(10**18));
350 require((gaussSold + tokenAmount) <= stages[15], "Crowdsale: token amount can not be more that total amount alloted to Crowdsale.");
351
```

UNKNOWN Arithmetic operation "**" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
347
348 // Calculates the token amount using the "jagerAmount" and the rate at the current stage.
349 uint256 tokenAmount = ((_jagerAmount * rates[currentStage])/(10**18));
350 require((gaussSold + tokenAmount) <= stages[15], "Crowdsale: token amount can not be more that total amount alloted to Crowdsale.");
351
```

UNKNOWN Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
348 // Calculates the token amount using the "jagerAmount" and the rate at the current stage.
349 uint256 tokenAmount = ((_jagerAmount * rates[currentStage])/(10**18));
350 require((gaussSold + tokenAmount) <= stages[15], "Crowdsale: token amount can not be more that total amount alloted to Crowdsale.");
351
352 // Adds the wallet address and "tokenAmount" to the beneficiary's balance.
```

UNKNOWN Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
354
355 // Adds the "jagerAmount" to the purchaseTotal of the buyer.
356 purchaseTotals[_beneficiary] += _jagerAmount;
357
358 // Tranfers the BNB recieved in purchase to the Crowdsale Wallet.
```

UNKNOWN Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
366 // Updates the amount of tokens left in the Crowdsale; may change the stage if conditions are met.
367 function _updatePurchasingState(uint256 _tokenAmount, uint256 _jagerAmount) internal {
368 gaussSold += _tokenAmount;
369 jagerRaised += _jagerAmount;
370
```

UNKNOWN Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
367 function _updatePurchasingState(uint256 _tokenAmount, uint256 _jagerAmount) internal {
368 gaussSold += _tokenAmount;
369 jagerRaised += _jagerAmount;
370
371 if (gaussSold >= stages[currentStage]) {
```

UNKNOWN Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
371 if (gaussSold >= stages[currentStage]) {
372 if (currentStage < stages.length) {
373 currentStage += 1;
374 }
375 }
```

UNKNOWN Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
399 uint256[] memory gaussBought = new uint256[](buyers.length);
400
401 for (uint256 i = 0; i < buyers.length; i++) {
402 wallets[i] = buyers[i].wallet;
403 bnbSpent[i] = purchaseTotals[buyers[i].wallet];
```

UNKNOWN Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
415 require(wallets.length == tokenAmounts.length);
416
417 for (uint256 i = 0; i < wallets.length; i++) {
418 balances[i] = (Buyer(wallets[i], tokenAmounts[i]));
419 }
```

UNKNOWN Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
419 }
420
421 for (uint256 i = 0; i < balances.length; i++) {
422 if (i >= wallets.length) {
423 balances.pop();
```

UNKNOWN Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
435 Buyer[] memory buyers = balances;
436
437 for (uint256 i = 0; i < buyers.length; i++) {
438 require(_token.transfer(buyers[i].wallet, buyers[i].tokenAmount));
439 }
```

UNKNOWN Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
449
450 uint256 amount;
451 for (uint256 i = 0; i < balances.length; i++) {
452 if (balances[i].wallet == msg.sender) {
453 amount = balances[i].tokenAmount;
```

UNKNOWN

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
347
348 // Calculates the token amount using the "jagerAmount" and the rate at the current stage.
349 uint256 tokenAmount = ((_jagerAmount * rates[currentStage])/(10**18));
350 require((gaussSold + tokenAmount) <= stages[15], "Crowdsale: token amount can not be more that total amount alloted to Crowdsale.");
351
```

UNKNOWN

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
348 // Calculates the token amount using the "jagerAmount" and the rate at the current stage.
349 uint256 tokenAmount = ((_jagerAmount * rates[currentStage])/(10**18));
350 require((gaussSold + tokenAmount) <= stages[15], "Crowdsale: token amount can not be more that total amount alloted to Crowdsale.");
351
352 // Adds the wallet address and "tokenAmount" to the beneficiary's balance.
```

UNKNOWN

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
369 jagerRaised += _jagerAmount;
370
371 if (gaussSold >= stages[currentStage]) {
372 if (currentStage < stages.length) {
373 currentStage += 1;
```

UNKNOWN Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
400
401 for (uint256 i = 0; i < buyers.length; i++) {
402 wallets[i] = buyers[i].wallet;
403 bnbSpent[i] = purchaseTotals[buyers[i].wallet];
404 gaussBought[i] = buyers[i].tokenAmount;
```

UNKNOWN Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
400
401 for (uint256 i = 0; i < buyers.length; i++) {
402 wallets[i] = buyers[i].wallet;
403 bnbSpent[i] = purchaseTotals[buyers[i].wallet];
404 gaussBought[i] = buyers[i].tokenAmount;
```

UNKNOWN Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
401 for (uint256 i = 0; i < buyers.length; i++) {
402 wallets[i] = buyers[i].wallet;
403 bnbSpent[i] = purchaseTotals[buyers[i].wallet];
404 gaussBought[i] = buyers[i].tokenAmount;
405 }
```

UNKNOWN Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
401 for (uint256 i = 0; i < buyers.length; i++) {
402 wallets[i] = buyers[i].wallet;
403 bnbSpent[i] = purchaseTotals[buyers[i].wallet];
404 gaussBought[i] = buyers[i].tokenAmount;
405 }
```

UNKNOWN Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
402 wallets[i] = buyers[i].wallet;
403 bnbSpent[i] = purchaseTotals[buyers[i].wallet];
404 gaussBought[i] = buyers[i].tokenAmount;
405 }
406
```

UNKNOWN Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
402 wallets[i] = buyers[i].wallet;
403 bnbSpent[i] = purchaseTotals[buyers[i].wallet];
404 gaussBought[i] = buyers[i].tokenAmount;
405 }
406
```

UNKNOWN Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
416
417 for (uint256 i = 0; i < wallets.length; i++) {
418 balances[i] = (Buyer(wallets[i], tokenAmounts[i]));
419 }
420
```

UNKNOWN Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
416
417 for (uint256 i = 0; i < wallets.length; i++) {
418 balances[i] = (Buyer(wallets[i], tokenAmounts[i]));
419 }
420
```

UNKNOWN Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
416
417 for (uint256 i = 0; i < wallets.length; i++) {
418 balances[i] = (Buyer(wallets[i], tokenAmounts[i]));
419 }
420
```

UNKNOWN Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
436
437 for (uint256 i = 0; i < buyers.length; i++) {
438 require(_token.transfer(buyers[i].wallet, buyers[i].tokenAmount));
439 }
440 }
```

UNKNOWN Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
436
437 for (uint256 i = 0; i < buyers.length; i++) {
438 require(_token.transfer(buyers[i].wallet, buyers[i].tokenAmount));
439 }
440 }
```

UNKNOWN Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
450 uint256 amount;
451 for (uint256 i = 0; i < balances.length; i++) {
452 if (balances[i].wallet == msg.sender) {
453 amount = balances[i].tokenAmount;
454
```

UNKNOWN

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
451 for (uint256 i = 0; i < balances.length; i++) {
452 if (balances[i].wallet == msg.sender) {
453 amount = balances[i].tokenAmount;
454
455 require(amount > 0, "Crowdsale: can not withdrawl 0 amount.");
```

UNKNOWN

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

SWC-110

Source file

/flattenedcontracts/gausscrowdsale_flattened.sol

Locations

```
455 require(amount > 0, "Crowdsale: can not withdrawl 0 amount.");
456 _token.transfer(msg.sender, amount);
457 balances[i].tokenAmount = 0;
458 }
459 }
```