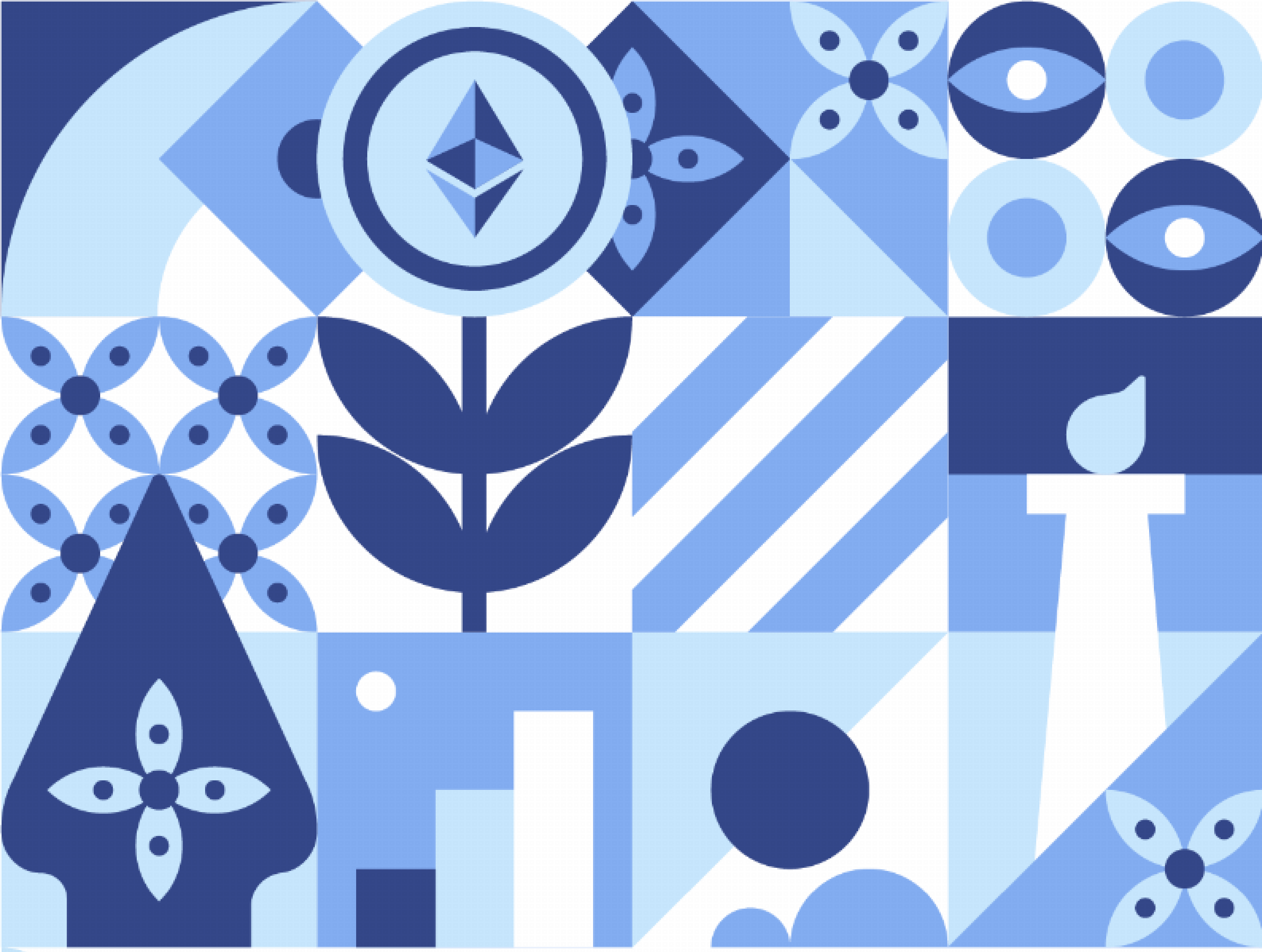

PIRATE TOKEN

Smart Contract Audit Report

SYSFIXED

15 Dec 2022

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
PIRATE TOKEN	PIRATE TOKEN	Ethereum

Addresses

Contract address	0x8d79323d27f3dcc2781fe44192caf1ad7e836787
Contract deployer address	0x83Bf5e55e8aEe4e63210Db612c82dB822f3103fF

Project Website

https://thepiratetoken.com/

Codebase

https://etherscan.io/address/0x8d79323d27f3dcc2781fe44192caf1ad7e836787#code

SUMMARY

We are currently building a P2E pirate game and we are about to release our first tier of NFT's. These NFT's will have huge utility for the game and our MetaVerse.
Our end goal is to build Pirate Ships in the real world for family adventures and fun cruises.

Contract Summary

Documentation Quality

PIRATE TOKEN provides a very good documentation with standard of solidity base code.

- The technical description is provided clearly and structured and also dont have any high risk issue.

Code Quality

The Overall quality of the basecode is standard.

- Standard solidity basecode and rules are already followed by PIRATE TOKEN with the discovery of several low issues.

Test Coverage

Test coverage of the project is 100% (Through Codebase)

Audit Findings Summary

- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 305, 324, 346, 379, 381, 402, 403, 428, 430, 601, 615, 630, 631, 644, 656, 671, 685, 699, 713, 729, 752, 775, 801, 1516, 1535, 1557, 1590, 1592, 1613, 1614, 1639, 1641, 1868, 1872, 1884, 1891, 1900, 2001, 2105, 2140, 2227, 2512, 2522, 2526, 2703, 2811, 3056 and 2001.
- SWC-110 SWC-123 | It is recommended to use of revert(), assert(), and require() in Solidity, and the new REVERT opcode in the EVM on lines 1970, 2002, 2007, 2518, 2687, 2688, 2694, 2698, 2699, 2700, 2709, 2716, 2812, 3128, 3129, 3145, 3146 and 3147.
- SWC-115 | tx.origin should not be used for authorization, use msg.sender instead on lines 2984 and 3084.

CONCLUSION

We have audited the PIRATE TOKEN project released on December 2022 to discover issues and identify potential security vulnerabilities in PIRATE TOKEN Project. This process is used to find technical issues and security loopholes which might be found in the smart contract.

The security audit report provides a satisfactory result with some low-risk issues.

The issues found in the PIRATE TOKEN smart contract code do not pose a considerable risk. The writing of the contract is close to the standard of writing contracts in general. The low-risk issues found are some arithmetic operation issues, tx.origin as a part of authorization control, and out of bounds array access which the index access expression can cause an exception in case of the use of an invalid array index value. We recommend avoiding "tx.origin" issue, using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing.

AUDIT RESULT

Article	Category	Description	Result
Default Visibility	SWC-100 SWC-108	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	PASS
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	ISSUE FOUND
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	PASS
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	PASS
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	PASS
Unprotected Ether Withdrawal	SWC-105	Due to missing or insufficient access controls, malicious parties can withdraw from the contract.	PASS
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	PASS
Reentrancy	SWC-107	Check effect interaction pattern should be followed if the code performs recursive call.	PASS
Uninitialized Storage Pointer	SWC-109	Uninitialized local storage variables can point to unexpected storage locations in the contract.	PASS
Assert Violation	SWC-110 SWC-123	Properly functioning code should never reach a failing assert statement.	ISSUE FOUND
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	PASS
Delegate call to Untrusted Callee	SWC-112	Delegatecalls should only be allowed to trusted addresses.	PASS

DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless required.	PASS
Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	PASS
Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	ISSUE FOUND
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	PASS
Signature Unique ID	SWC-117 SWC-121 SWC-122	Signed messages should always have a unique id. A transaction hash should not be used as a unique id.	PASS
Incorrect Constructor Name	SWC-118	Constructors are special functions that are called only once during the contract creation.	PASS
Shadowing State Variable	SWC-119	State variables should not be shadowed.	PASS
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	PASS
Write to Arbitrary Storage Location	SWC-124	The contract is responsible for ensuring that only authorized user or contract accounts may write to sensitive storage locations.	PASS
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order. The rule of thumb is to inherit contracts from more /general/ to more /specific/.	PASS
Insufficient Gas Griefing	SWC-126	Insufficient gas griefing attacks can be performed on contracts which accept data and use it in a sub-call on another contract.	PASS
Arbitrary Jump Function	SWC-127	As Solidity doesnt support pointer arithmetics, it is impossible to change such variable to an arbitrary value.	PASS

Typographical Error	SWC-129	A typographical error can occur for example when the intent of a defined operation is to sum a number to a variable.	PASS
Override control character	SWC-130	Malicious actors can use the Right-To-Left-Override unicode character to force RTL text rendering and confuse users as to the real intent of a contract.	PASS
Unused variables	SWC-131 SWC-135	Unused variables are allowed in Solidity and they do not pose a direct security issue.	PASS
Unexpected Ether balance	SWC-132	Contracts can behave erroneously when they strictly assume a specific Ether balance.	PASS
Hash Collisions Variable	SWC-133	Using abi.encodePacked() with multiple variable length arguments can, in certain situations, lead to a hash collision.	PASS
Hardcoded gas amount	SWC-134	The transfer() and send() functions forward a fixed amount of 2300 gas.	PASS
Unencrypted Private Data	SWC-136	It is a common misconception that private type variables cannot be read.	PASS

SMART CONTRACT ANALYSIS

Started	Wednesday Dec 14 2022 18:57:54 GMT+0000 (Coordinated Universal Time)
Finished	Thursday Dec 15 2022 14:58:14 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	AntiBotBABYTOKEN.sol

Detected Issues

ID	Title	Severity	Status
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "%" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "%" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "%" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-=" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-115	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	low	acknowledged
SWC-115	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	low	acknowledged

SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 305

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
304   unchecked {
305   _approve(sender, _msgSender(), currentAllowance - amount);
306   }
307
308   return true;
309
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 324

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
323   function increaseAllowance(address spender, uint256 addedValue) public virtual
returns (bool) {
324   _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
325   return true;
326   }
327
328
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 346

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
345   unchecked {
346   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
347   }
348
349   return true;
350
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 379

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
378   unchecked {
379   _balances[sender] = senderBalance - amount;
380   }
381   _balances[recipient] += amount;
382
383
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 381

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
380   }
381   _balances[recipient] += amount;
382
383   emit Transfer(sender, recipient, amount);
384
385
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 402

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
401
402   _totalSupply += amount;
403   _balances[account] += amount;
404   emit Transfer(address(0), account, amount);
405
406
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 403

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
402   _totalSupply += amount;
403   _balances[account] += amount;
404   emit Transfer(address(0), account, amount);
405
406   _afterTokenTransfer(address(0), account, amount);
407
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 428

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
427   unchecked {
428   _balances[account] = accountBalance - amount;
429   }
430   _totalSupply -= amount;
431
432
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 430

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
429   }
430   _totalSupply -= amount;
431
432   emit Transfer(account, address(0), amount);
433
434
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 601

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
600   unchecked {
601   uint256 c = a + b;
602   if (c < a) return (false, 0);
603   return (true, c);
604   }
605
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 615

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
614   if (b > a) return (false, 0);
615   return (true, a - b);
616   }
617   }
618
619
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 630

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
629   if (a == 0) return (true, 0);
630   uint256 c = a * b;
631   if (c / a != b) return (false, 0);
632   return (true, c);
633   }
634
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 631

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
630   uint256 c = a * b;
631   if (c / a != b) return (false, 0);
632   return (true, c);
633   }
634   }
635
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 644

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
643   if (b == 0) return (false, 0);
644   return (true, a / b);
645   }
646   }
647
648
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 656

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
655   if (b == 0) return (false, 0);
656   return (true, a % b);
657   }
658   }
659
660
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 671

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
670   function add(uint256 a, uint256 b) internal pure returns (uint256) {
671   return a + b;
672   }
673
674   /**
675
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 685

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
684   function sub(uint256 a, uint256 b) internal pure returns (uint256) {
685   return a - b;
686   }
687
688   /**
689
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 699

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
698   function mul(uint256 a, uint256 b) internal pure returns (uint256) {
699   return a * b;
700   }
701
702   /**
703
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 713

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
712   function div(uint256 a, uint256 b) internal pure returns (uint256) {
713   return a / b;
714   }
715
716   /**
717
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 729

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
728   function mod(uint256 a, uint256 b) internal pure returns (uint256) {
729   return a % b;
730   }
731
732   /**
733
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 752

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
751   require(b <= a, errorMessage);
752   return a - b;
753   }
754   }
755
756
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 775

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
774   require(b > 0, errorMessage);
775   return a / b;
776   }
777   }
778
779
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 801

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
800   require(b > 0, errorMessage);
801   return a % b;
802   }
803   }
804   }
805
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1516

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1515   unchecked {
1516   _approve(sender, _msgSender(), currentAllowance - amount);
1517   }
1518
1519   return true;
1520
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1535

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1534   function increaseAllowance(address spender, uint256 addedValue) public virtual
returns (bool) {
1535   _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
1536   return true;
1537   }
1538
1539
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1557

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1556   unchecked {
1557   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
1558   }
1559
1560   return true;
1561
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1590

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1589   unchecked {
1590   _balances[sender] = senderBalance - amount;
1591   }
1592   _balances[recipient] += amount;
1593
1594
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1592

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1591   }
1592   _balances[recipient] += amount;
1593
1594   emit Transfer(sender, recipient, amount);
1595
1596
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1613

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1612
1613   _totalSupply += amount;
1614   _balances[account] += amount;
1615   emit Transfer(address(0), account, amount);
1616
1617
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1614

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1613   _totalSupply += amount;
1614   _balances[account] += amount;
1615   emit Transfer(address(0), account, amount);
1616
1617   _afterTokenTransfer(address(0), account, amount);
1618
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1639

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1638   unchecked {
1639   _balances[account] = accountBalance - amount;
1640   }
1641   _totalSupply -= amount;
1642
1643
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 1641

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1640   }
1641   _totalSupply -= amount;
1642
1643   emit Transfer(account, address(0), amount);
1644
1645
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1868

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1867   function mul(int256 a, int256 b) internal pure returns (int256) {
1868   int256 c = a * b;
1869
1870   // Detect overflow when multiplying MIN_INT256 with -1
1871   require(c != MIN_INT256 || (a & MIN_INT256) != (b & MIN_INT256));
1872
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1872

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1871   require(c != MIN_INT256 || (a & MIN_INT256) != (b & MIN_INT256));
1872   require((b == 0) || (c / b == a));
1873   return c;
1874   }
1875
1876
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1884

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1883   // Solidity already throws when dividing by 0.
1884   return a / b;
1885   }
1886
1887   /**
1888
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1891

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1890   function sub(int256 a, int256 b) internal pure returns (int256) {
1891   int256 c = a - b;
1892   require((b >= 0 && c <= a) || (b < 0 && c > a));
1893   return c;
1894   }
1895
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1900

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1899   function add(int256 a, int256 b) internal pure returns (int256) {
1900   int256 c = a + b;
1901   require((b >= 0 && c >= a) || (b < 0 && c < a));
1902   return c;
1903   }
1904
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 2001

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2000   uint256 index = map.indexOf[key];
2001   uint256 lastIndex = map.keys.length - 1;
2002   address lastKey = map.keys[lastIndex];
2003
2004   map.indexOf[lastKey] = index;
2005
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 2105

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2104   //  see https://github.com/ethereum/EIPs/issues/1726#issuecomment-472352728
2105   uint256 internal constant magnitude = 2**128;
2106
2107   uint256 internal magnifiedDividendPerShare;
2108
2109
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 2140

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2139   magnifiedDividendPerShare = magnifiedDividendPerShare.add(
2140   (amount).mul(magnitude) / totalSupply()
2141   );
2142   emit DividendsDistributed(msg.sender, amount);
2143
2144
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 2227

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2226   return
2227   magnifiedDividendPerShare
2228   .mul(balanceOf(_owner))
2229   .toInt256Safe()
2230   .add(magnifiedDividendCorrections[_owner])
2231
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 2512

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2511   while (gasUsed < gas && iterations < numberOfTokenHolders) {
2512   _lastProcessedIndex++;
2513
2514   if (_lastProcessedIndex >= tokenHoldersMap.keys.length) {
2515   _lastProcessedIndex = 0;
2516
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 2522

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2521   if (processAccount(payable(account), true)) {
2522   claims++;
2523   }
2524   }
2525
2526
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 2526

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2525
2526   iterations++;
2527
2528   uint256 newGasLeft = gasleft();
2529
2530
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 2703

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2702    require(totalFees <= 25, "Total fee is over 25%");
2703    swapTokensAtAmount = totalSupply_.mul(2).div(10**6); // 0.002%
2704
2705    // use by default 300,000 gas to process auto-claiming dividends
2706    gasForProcessing = 300000;
2707
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 2811

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2810   ) public onlyOwner {
2811   for (uint256 i = 0; i < accounts.length; i++) {
2812   _isExcludedFromFees[accounts[i]] = excluded;
2813   }
2814
2815
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 3056

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
3055   if (automatedMarketMakerPairs[to]) {
3056   fees += amount.mul(1).div(100);
3057   }
3058   amount = amount.sub(fees);
3059
3060
```

SWC-101 | COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED

LINE 2001

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2000   uint256 index = map.indexOf[key];
2001   uint256 lastIndex = map.keys.length - 1;
2002   address lastKey = map.keys[lastIndex];
2003
2004   map.indexOf[lastKey] = index;
2005
```

SWC-115 | USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.

LINE 2984

low SEVERITY

Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing.

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2983   gas,
2984   tx.origin
2985   );
2986   }
2987
2988
```

SWC-115 | USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.

LINE 3084

low SEVERITY

Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing.

Source File

- AntiBotBABYTOKEN.sol

Locations

```
3083   gas,
3084   tx.origin
3085   );
3086   } catch {}
3087   }
3088
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1970

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1969   {
1970   return map.keys[index];
1971   }
1972
1973   function size(Map storage map) public view returns (uint256) {
1974
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2002

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2001   uint256 lastIndex = map.keys.length - 1;
2002   address lastKey = map.keys[lastIndex];
2003
2004   map.indexOf[lastKey] = index;
2005   delete map.indexOf[key];
2006
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2007

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2006
2007    map.keys[index] = lastKey;
2008    map.keys.pop();
2009    }
2010    }
2011
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2518

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2517
2518    address account = tokenHoldersMap.keys[_lastProcessedIndex];
2519
2520    if (canAutoClaim(lastClaimTimes[account])) {
2521    if (processAccount(payable(account), true)) {
2522
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2687

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2686   ) payable ERC20(name_, symbol_) {
2687   rewardToken = addrs[0];
2688   _marketingWalletAddress = addrs[2];
2689   require(
2690   msg.sender != _marketingWalletAddress,
2691
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2688

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2687   rewardToken = addrs[0];
2688   _marketingWalletAddress = addrs[2];
2689   require(
2690   msg.sender != _marketingWalletAddress,
2691   "Owner and marketing wallet cannot be the same"
2692
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2694

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2693
2694   pinkAntiBot = IPinkAntiBot(addrs[4]);
2695   pinkAntiBot.setTokenOwner(owner());
2696   enableAntiBot = true;
2697
2698
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2698

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2697
2698   tokenRewardsFee = feeSettings[0];
2699   liquidityFee = feeSettings[1];
2700   marketingFee = feeSettings[2];
2701   totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
2702
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2699

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2698   tokenRewardsFee = feeSettings[0];
2699   liquidityFee = feeSettings[1];
2700   marketingFee = feeSettings[2];
2701   totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
2702   require(totalFees <= 25, "Total fee is over 25%");
2703
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2700

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2699   liquidityFee = feeSettings[1];
2700   marketingFee = feeSettings[2];
2701   totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
2702   require(totalFees <= 25, "Total fee is over 25%");
2703   swapTokensAtAmount = totalSupply_.mul(2).div(10**6); // 0.002%
2704
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2709

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2708   dividendTracker = BABYTOKENDividendTracker(
2709   payable(Clones.clone(addrs[3]))
2710   );
2711   dividendTracker.initialize(
2712   rewardToken,
2713
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2716

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2715
2716   IUniswapV2Router02 _uniswapV2Router = IUniswapV2Router02(addrs[1]);
2717   // Create a uniswap pair for this new token
2718   address _uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory())
2719   .createPair(address(this), _uniswapV2Router.WETH());
2720
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2812

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2811   for (uint256 i = 0; i < accounts.length; i++) {
2812   _isExcludedFromFees[accounts[i]] = excluded;
2813   }
2814
2815   emit ExcludeMultipleAccountsFromFees(accounts, excluded);
2816
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 3128

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
3127   address[] memory path = new address[](2);
3128   path[0] = address(this);
3129   path[1] = uniswapV2Router.WETH();
3130
3131   _approve(address(this), address(uniswapV2Router), tokenAmount);
3132
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 3129

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
3128   path[0] = address(this);
3129   path[1] = uniswapV2Router.WETH();
3130
3131   _approve(address(this), address(uniswapV2Router), tokenAmount);
3132
3133
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 3145

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
3144   address[] memory path = new address[](3);
3145   path[0] = address(this);
3146   path[1] = uniswapV2Router.WETH();
3147   path[2] = rewardToken;
3148
3149
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 3146

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
3145    path[0] = address(this);
3146    path[1] = uniswapV2Router.WETH();
3147    path[2] = rewardToken;
3148
3149    _approve(address(this), address(uniswapV2Router), tokenAmount);
3150
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 3147

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
3146   path[1] = uniswapV2Router.WETH();
3147   path[2] = rewardToken;
3148
3149   _approve(address(this), address(uniswapV2Router), tokenAmount);
3150
3151
```

DISCLAIMER

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 Sysfixed's 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 Sysfixed 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 is a limited report on our findings based on our analysis, in accordance with good industry practice as of 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.

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.

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 Sysfixed and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (Sysfixed) owe no duty of care.

ABOUT US

Sysfixed is a blockchain security certification organization established in 2021 with the objective to provide smart contract security services and verify their correctness in blockchain-based protocols. Sysfixed automatically scans for security vulnerabilities in Ethereum and other EVM-based blockchain smart contracts. Sysfixed a comprehensive range of analysis techniques—including static analysis, dynamic analysis, and symbolic execution—can accurately detect security vulnerabilities to provide an in-depth analysis report. With a vibrant ecosystem of world-class integration partners that amplify developer productivity, Sysfixed can be utilized in all phases of your project's lifecycle. Our team of security experts is dedicated to the research and improvement of our tools and techniques used to fortify your code.