ZeroXPad

Smart Contract Audit Report

SYSFIXED

27 Jan 2023

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
ZeroXPad	ZXP	Binance Smart Chain

Addresses

Contract address	0xBBB603Da8A209188B1d083a6f7a6f66D4992a5f4
Contract deployer address	0x488A8CA56f29BFbe28e6f4cf898D5c3C1455deDa

Project Website

https://www.0xpad.app/

Codebase

https://bscscan.com/address/0xBBB603Da8A209188B1d083a6f7a6f66D4992a5f4#code

SUMMARY

0xpad is a hybrid fundraiser fusing marketing along an innovative borrow mechanism to raise funds for startups, guarantee returns for investors, in addition to introducing the CBC standard.

Contract Summary

Documentation Quality

ZeroXPad 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 ZeroXPad 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 283, 301, 320, 321, 338, 354, 369, 383, 397, 411, 427, 450, 473, 499, 943, 973, 1009, 1012, 1034, 1037, 1063, 1065, 1118, 1235, 1379 and 1395.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 9, 172, 223, 259, 508, 534, 623, 714, 742, 1168 and 1206.

CONCLUSION

We have audited the ZeroXPad project released on January 2023 to discover issues and identify potential security vulnerabilities in ZeroXPad Project. This process is used to find technical issues and security loopholes which might be found in the smart contract.

The security audit report yielded satisfactory results with some low-risk issues.

The issues found in the code on ZeroXPad smart contract 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 and floating pragmas set on multiple lines.

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.	ISSUE FOUND
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	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
Assert Violation	SWC-110	Properly functioning code should never reach a failing assert statement.	PASS
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	PASS
Delegate call to Untrusted Caller	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.	PASS
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
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
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

SMART CONTRACT ANALYSIS

Started	Thursday Jan 26 2023 18:54:50 GMT+0000 (Coordinated Universal Time)
Finished	Friday Jan 27 2023 19:43:57 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	ZeroXPad.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-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 283

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
282   unchecked {
283   uint256 c = a + b;
284   if (c < a) return (false, 0);
285   return (true, c);
286   }
287
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 301

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
300   if (b > a) return (false, 0);
301   return (true, a - b);
302   }
303   }
304
305
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 320

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
319   if (a == 0) return (true, 0);
320   uint256 c = a * b;
321   if (c / a != b) return (false, 0);
322   return (true, c);
323   }
324
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 321

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
320   uint256 c = a * b;
321   if (c / a != b) return (false, 0);
322   return (true, c);
323   }
324   }
325
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 338

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
337   if (b == 0) return (false, 0);
338   return (true, a / b);
339   }
340   }
341
342
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 354

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
353   if (b == 0) return (false, 0);
354   return (true, a % b);
355   }
356   }
357
358
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 369

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
368   function add(uint256 a, uint256 b) internal pure returns (uint256) {
369   return a + b;
370   }
371
372   /**
373
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 383

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
382   function sub(uint256 a, uint256 b) internal pure returns (uint256) {
383   return a - b;
384   }
385
386   /**
387
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 397

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
396   function mul(uint256 a, uint256 b) internal pure returns (uint256) {
397   return a * b;
398   }
399
400   /**
401
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 411

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
410   function div(uint256 a, uint256 b) internal pure returns (uint256) {
411   return a / b;
412   }
413
414   /**
415
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 427

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
426   function mod(uint256 a, uint256 b) internal pure returns (uint256) {
427   return a % b;
428   }
429
430   /**
431
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 450

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
449   require(b <= a, errorMessage);
450   return a - b;
451   }
452   }
453
454
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 473

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
472   require(b > 0, errorMessage);
473   return a / b;
474   }
475   }
476
477
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 499

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
498   require(b > 0, errorMessage);
499   return a % b;
500   }
501   }
502   }
503
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 943

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
942   address owner = _msgSender();
943   _approve(owner, spender, allowance(owner, spender) + addedValue);
944   return true;
945   }
946
947
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 973

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
972   unchecked {
973   _approve(owner, spender, currentAllowance - subtractedValue);
974   }
975
976   return true;
977
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1009

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
1008   unchecked {
1009   _balances[from] = fromBalance - amount;
1010   // Overflow not possible: the sum of all balances is capped by totalSupply, and
the sum is preserved by
1011   // decrementing then incrementing.
1012   _balances[to] += amount;
1013
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1012

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
1011   // decrementing then incrementing.
1012   _balances[to] += amount;
1013   }
1014
1015   emit Transfer(from, to, amount);
1016
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1034

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
1033
1034   _totalSupply += amount;
1035   unchecked {
1036   // Overflow not possible: balance + amount is at most totalSupply + amount, which
is checked above.
1037   _balances[account] += amount;
1038
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1037

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
1036   // Overflow not possible: balance + amount is at most totalSupply + amount, which
is checked above.
1037   _balances[account] += amount;
1038   }
1039   emit Transfer(address(0), account, amount);
1040
1041
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1063

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
1062   unchecked {
1063   _balances[account] = accountBalance - amount;
1064   // Overflow not possible: amount <= accountBalance <= totalSupply.
1065   _totalSupply -= amount;
1066   }
1067
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 1065

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
1064   // Overflow not possible: amount <= accountBalance <= totalSupply.
1065   _totalSupply -= amount;
1066   }
1067
1068   emit Transfer(account, address(0), amount);
1069
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1118

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
1117   unchecked {
1118   _approve(owner, spender, currentAllowance - amount);
1119   }
1120   }
1121   }
1122
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1235

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
1234   require(
1235   ERC20.totalSupply() + amount <= cap(),
1236   "ERC20Capped: cap exceeded"
1237   );
1238   super._mint(account, amount);
1239
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1379

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
1378   uint256 fees = amount.mul(buyFee_).div(10000);
1379   uint256 rest = amount - fees;
1380
1381   super._transfer(from, treasury_, fees);
1382   super._transfer(from, to, rest);
1383
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1395

low SEVERITY

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

Source File

- ZeroXPad.sol

Locations

```
1394
1395   uint256 rest = amount - fees;
1396   super._transfer(from, treasury_, fees);
1397   super._transfer(from, to, rest);
1398   } else {
1399
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 9

low SEVERITY

The current pragma Solidity directive is ""=0.6.2"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- ZeroXPad.sol

Locations

```
8
9   pragma solidity >=0.6.2;
10
11   interface IUniswapV2Router01 {
12   function factory() external pure returns (address);
13
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 172

low SEVERITY

The current pragma Solidity directive is ""^0.6.2"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- ZeroXPad.sol

Locations

```
171
172   pragma solidity ^0.6.2;
173
174   interface IUniswapV2Router02 is IUniswapV2Router01 {
175   function removeLiquidityETHSupportingFeeOnTransferTokens(
176
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 223

low SEVERITY

The current pragma Solidity directive is ""^0.5.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- ZeroXPad.sol

Locations

```
222
223   pragma solidity ^0.5.0;
224
225   interface IUniswapV2Factory {
226   event PairCreated(
227
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 259

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- ZeroXPad.sol

Locations

```
258
259   pragma solidity ^0.8.0;
260
261   // CAUTION
262   // This version of SafeMath should only be used with Solidity 0.8 or later,
263
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 508

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- ZeroXPad.sol

Locations

```
507
508   pragma solidity ^0.8.0;
509
510   /**
511   * @dev Provides information about the current execution context, including the
512
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 534

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- ZeroXPad.sol

Locations

```
533
534   pragma solidity ^0.8.0;
535
536   /**
537   * @dev Contract module which provides a basic access control mechanism, where
538
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 623

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- ZeroXPad.sol

Locations

```
622
623   pragma solidity ^0.8.0;
624
625   /**
626   * @dev Interface of the ERC20 standard as defined in the EIP.
627
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 714

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- ZeroXPad.sol

Locations

```
713
714   pragma solidity ^0.8.0;
715
716   /**
717   * @dev Interface for the optional metadata functions from the ERC20 standard.
718
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 742

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- ZeroXPad.sol

Locations

```
741
742   pragma solidity ^0.8.0;
743
744   /**
745   * @dev Implementation of the {IERC20} interface.
746
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 1168

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- ZeroXPad.sol

Locations

```
1167
1168   pragma solidity ^0.8.0;
1169
1170   /**
1171   * @dev Extension of {ERC20} that allows token holders to destroy both their own
1172
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 1206

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- ZeroXPad.sol

Locations

```
1205
1206   pragma solidity ^0.8.0;
1207
1208   /**
1209   * @dev Extension of {ERC20} that adds a cap to the supply of tokens.
1210
```

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.