Xarpent

Smart Contract Audit Report

SYSFIXED

15 Jan 2023

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Xarpent	XPT	BSC

Addresses

Contract address	0x970A68775186509749561F863a0E228ca8c43D0c
Contract deployer address	0xD3f5E23BC65FEa6a34bd06E4cb3E07cd60004fF1

Project Website

https://xarpent.finance/

Codebase

https://bscscan.com/address/0x970A68775186509749561F863a0E228ca8c43D0c#code

SUMMARY

Trade assets across the supported blockchains with one click. Partner projects can include our free tool on their website to make it simple for their community to exchange tokens between chains. The project has a cross-chain aggregator, a free widget tool, monthly integration of new chains, always online support bot, Reps, staking rounds, partnerships, and collaborations.

Contract Summary

Documentation Quality

Xarpent provides a document with a good standard and stable solidity basecode.

- The technical description is provided clearly and structured.

Code Quality

The Overall quality of the basecode is GOOD with only 3 low-risk issues

- Standart solidity basecode and rules are already followed with Xarpent Project but there is still has potential-randomness (SWC-120).

Test Coverage

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

Audit Findings Summary

- SWC-101 | Arithmetic operation Issues discovered on lines 194, 216, 241, 270, 271, 400, 435, 468, 478, 489, 517, 526, 534, 545, 552, 556, 576, 577, 579, 585, 586, 587, 594, 599, 604, 653, 690, and 1001.
- SWC-103 | A floating pragma is set on lines 7.
- SWC-110 | Out of bounds array access on lines 615 and 616.
- SWC-120 | Potential use of "block.number" as a source of randomness on lines 517 and 660.

CONCLUSION

We have audited the Xarpent Coin, which has been released to discover issues and identify potential security vulnerabilities in LaunchVerseProject. This process is used to find bugs, technical issues, and security loopholes that find some common issues in the code.

The security audit report produced satisfactory results with three low-risk issues on the contract project.

The most common issue in writing code on contracts that do not pose a big risk is that writing on contracts is close to the standard of writing contracts in general. Some of the common issues that were found were asserted violations, a floating pragma set Potential use of "block.number" as a source of randomness. We recommend not using any of those environment variables as sources of randomness and being aware that using these variables introduces a certain level of trust in miners.

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
Check-Effect Interaction	SWC-107	Check-Effect-Interaction pattern should be followed if the code performs ANY external call.	PASS
Assert Violation	SWC-110	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 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.	ISSUE FOUND
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	Sat Jan 14 2023 05:57:30 GMT+0000 (Coordinated Universal Time)
Finished	Sun Jan 15 2023 06:57:30 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	Xarpent.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-103	A FLOATING PRAGMA IS SET.	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-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 194

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
193   require(currentAllowance >= amount, "BEP20: transfer amount exceeds allowance");
194   _approve(sender, _msgSender(), currentAllowance - amount);
195
196   return true;
197   }
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 216

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
215   {
216   _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
217   return true;
218   }
219
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 241

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
240   require(currentAllowance >= subtractedValue, "BEP20: decreased allowance below
zero");
241   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
242
243   return true;
244   }
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 270

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
269   require(senderBalance >= amount, "BEP20: transfer amount exceeds balance");
270   _balances[sender] = senderBalance - amount;
271   _balances[recipient] += amount;
272
273   emit Transfer(sender, recipient, amount);
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 271

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
270   _balances[sender] = senderBalance - amount;
271   _balances[recipient] += amount;
272
273   emit Transfer(sender, recipient, amount);
274   }
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 400

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
399
400   uint256 public tokenLiquidityThreshold = 5e4 * 10**18;
401
402   uint256 public genesis_block;
403   uint256 private deadline = 3;
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 435

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
434   constructor() BEP20("Xarpent", "XPT") {
435   _tokengeneration(msg.sender, 5e7 * 10**decimals());
436   exemptFee[msg.sender] = true;
437
438   IRouter _router = IRouter(0x10ED43C718714eb63d5aA57B78B54704E256024E);
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 468

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
467   require(currentAllowance >= amount, "BEP20: transfer amount exceeds allowance");
468   _approve(sender, _msgSender(), currentAllowance - amount);
469
470   return true;
471   }
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 478

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
477   {
478   _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
479   return true;
480   }
481
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 489

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
488   require(currentAllowance >= subtractedValue, "BEP20: decreased allowance below
zero");
489   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
490
491   return true;
492   }
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 517

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
516   !exemptFee[recipient] &&
517   block.number < genesis_block + deadline;
518
519   //set fee to zero if fees in contract are handled or exempted
520   if (_interlock || exemptFee[sender] || exemptFee[recipient])
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 526

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
525   feeswap =
526   sellTaxes.liquidity +
527   sellTaxes.marketing +
528   sellTaxes.ops +
529   sellTaxes.dev;
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 534

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
533   feeswap =
534   taxes.liquidity +
535   taxes.marketing +
536   taxes.ops +
537   taxes.dev;
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 545

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
544
545   fee = (amount * feesum) / 100;
546
547   //send fees if threshold has been reached
548   //don't do this on buys, breaks swap
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 552

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
551   //rest to recipient
552   super._transfer(sender, recipient, amount - fee);
553   if (fee > 0) {
554   //send the fee to the contract
555   if (feeswap > 0) {
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 556

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
555   if (feeswap > 0) {
556   uint256 feeAmount = (amount * feeswap) / 100;
557   super._transfer(sender, address(this), feeAmount);
558   }
559
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 576

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
575   // Split the contract balance into halves
576   uint256 denominator = feeswap * 2;
577   uint256 tokensToAddLiquidityWith = (contractBalance * swapTaxes.liquidity) /
578   denominator;
579   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 577

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
576   uint256 denominator = feeswap * 2;
577   uint256 tokensToAddLiquidityWith = (contractBalance * swapTaxes.liquidity) /
578   denominator;
579   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
580
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 579

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
578   denominator;
579   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
580
581   uint256 initialBalance = address(this).balance;
582
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 585

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
584
585   uint256 deltaBalance = address(this).balance - initialBalance;
586   uint256 unitBalance = deltaBalance / (denominator - swapTaxes.liquidity);
587   uint256 ethToAddLiquidityWith = unitBalance * swapTaxes.liquidity;
588
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 586

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
585   uint256 deltaBalance = address(this).balance - initialBalance;
586   uint256 unitBalance = deltaBalance / (denominator - swapTaxes.liquidity);
587   uint256 ethToAddLiquidityWith = unitBalance * swapTaxes.liquidity;
588
589   if (ethToAddLiquidityWith > 0) {
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 587

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
586   uint256 unitBalance = deltaBalance / (denominator - swapTaxes.liquidity);
587   uint256 ethToAddLiquidityWith = unitBalance * swapTaxes.liquidity;
588
589   if (ethToAddLiquidityWith > 0) {
590   // Add liquidity to pancake
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 594

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
593
594   uint256 marketingAmt = unitBalance * 2 * swapTaxes.marketing;
595   if (marketingAmt > 0) {
596   payable(marketingWallet).sendValue(marketingAmt);
597   }
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 599

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
598
599   uint256 opsAmt = unitBalance * 2 * swapTaxes.ops;
600   if (opsAmt > 0) {
601   payable(opsWallet).sendValue(opsAmt);
602   }
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 604

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
603
604   uint256 devAmt = unitBalance * 2 * swapTaxes.dev;
605   if (devAmt > 0) {
606   payable(devWallet).sendValue(devAmt);
607   }
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 653

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
652   require(new_amount <= 5e5, "Swap threshold amount should be lower or equal to 1% of
tokens");
653   tokenLiquidityThreshold = new_amount * 10**decimals();
654   }
655
656   function EnableTrading() external onlyOwner {
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 690

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
689   function bulkExemptFee(address[] memory accounts, bool state) external onlyOwner {
690   for (uint256 i = 0; i < accounts.length; i++) {
691   exemptFee[accounts[i]] = state;
692   }
693   }
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 7

low SEVERITY

The current pragma Solidity directive is ""^0.8.17"". 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

- Xarpent.Sol

Locations

```
6
7   pragma solidity ^0.8.17;
8
9   abstract contract Context {
10   function _msgSender() internal view virtual returns (address) {
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 615

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
614   address[] memory path = new address[](2);
615   path[0] = address(this);
616   path[1] = router.WETH();
617
618   _approve(address(this), address(router), tokenAmount);
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 616

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
615   path[0] = address(this);
616   path[1] = router.WETH();
617
618   _approve(address(this), address(router), tokenAmount);
619
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 691

low SEVERITY

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

Source File

- Xarpent.Sol

Locations

```
690   for (uint256 i = 0; i < accounts.length; i++) {
691   exemptFee[accounts[i]] = state;
692   }
693   }
694
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 517

low SEVERITY

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source File

- Xarpent.Sol

Locations

```
516   !exemptFee[recipient] &&
517   block.number < genesis_block + deadline;
518
519   //set fee to zero if fees in contract are handled or exempted
520   if (_interlock || exemptFee[sender] || exemptFee[recipient])
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 660

low SEVERITY

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source File

- Xarpent.Sol

Locations

```
659   providingLiquidity = true;
660   genesis_block = block.number;
661   }
662
663   function updatedeadline(uint256 _deadline) external onlyOwner {
```

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.