InterFi NETWORK

@INTERFINETWORK

SMART CONTRACT SECURITY AUDIT OF

SMART CONTRACT AUDIT | SOLIDITY DEVELOPMENT & TESTING | PROJECT EVALUATION

RELENTLESSLY SECURING THE PUBLIC BLOCKCHAIN

Audit Introduction

Auditing Firm	InterFi Network
Audit Architecture	InterFi Echelon Auditing Standard
Language	Solidity
Client Firm	Space Game
Website	https://spacenft.game/
Twitter	https://twitter.com/spacenftdotgame/
Discord	https://discord.gg/spacegame/
Github	https://github.com/Space-Game-NFT/AuditRef/
Report Date	February 05, 2022

Solidity Source Codes Under Scope

- MnA.sol

  https://github.com/Space-Game-NFT/AuditRef/blob/main/MnA.sol

- MnAGameCR.sol

  https://github.com/Space-Game-NFT/AuditRef/blob/main/MnAGameCR.sol

- ORES.sol

  https://github.com/Space-Game-NFT/AuditRef/blob/main/ORES.sol

- StakingPool.sol

  https://github.com/Space-Game-NFT/AuditRef/blob/main/StakingPool.sol

Audit Summary

InterFi team has performed a line-by-line manual analysis and automated review of smart contracts. Smart contracts were analyzed mainly for common contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

- Space Game's solidity source codes have **LOW RISK SEVERITY**
- Space Game's smart contracts have an **ACTIVE OWNERSHIP**
- Important owner privileges – **NFT MINT, PAUSE, SET/REMOVE ADMIN, AIRDROP, UPGRADE CONTRACTS**
- Space Game's smart contract owner has multiple "Write Contract" privileges. Centralization risk correlated to the active owner is **MEDIUM**
- Space Game's smart contracts are not deployed on any blockchain at the time of the audit, contracts can be modified or altered before blockchain development.

Be aware that smart contracts deployed on the blockchain aren't resistant to internal exploit, external vulnerability, or hack. For a detailed understanding of risk severity, source code vulnerability, functional hack, and audit disclaimer, kindly refer to the audit.

✅ Verify the authenticity of this report on InterFi's GitHub: **https://github.com/interfinetwork**

Table Of Contents

Audit Scope

InterFi was consulted by Space Game to conduct the smart contract security audit of their solidity source code. The audit scope of work is strictly limited to the mentioned solidity file(s) only:

Solidity Source Codes Under Scope

- MnA.sol
- MnAGameCR.sol
- ORES.sol
- StakingPool.sol

Solidity Source Codes NOT Under Scope

- IMnA.sol
- IStakingPool.sol
- ITraits.sol
- IRandomSeedGenerator.sol
- IFounderPass.sol
- IORES.sol
- ISpidox.sol

SHA-1 Hash

Solidity source code is audited at hash #b46005a7541e081eaa451eeb55a2d4844fc064ed

Audit Methodology

The scope of this report is to audit the smart contract source code of Space Game. InterFi has scanned the contract and reviewed the project for common vulnerabilities, exploits, hacks, and back-doors. Below is the list of commonly known smart contract vulnerabilities, exploits, and hacks:

Category	
Smart Contract Vulnerabilities	❖ Re-entrancy ❖ Unhandled Exceptions ❖ Transaction Order Dependency ❖ Integer Overflow ❖ Unrestricted Action ❖ Incorrect Inheritance Order ❖ Typographical Errors ❖ Requirement Violation ❖ Ownership Takeover ❖ Gas Limit and Loops ❖ Deployment Consistency
Source Code Review	❖ Repository Consistency ❖ Data Consistency ❖ Token Supply Manipulation
Functional Assessment	❖ Access Control and Authorization ❖ Operations Trail and Event Generation ❖ Assets Manipulation ❖ Liquidity Access

InterFi's Echelon Audit Standard

The aim of InterFi's "Echelon" standard is to analyze the smart contract and identify the vulnerabilities and the hacks in the smart contract. Mentioned are the steps used by ECHELON-1 to assess the smart contract:

1. Solidity smart contract source code reviewal:
   - ❖ Review of the specifications, sources, and instructions provided to InterFi to make sure we understand the size, scope, and functionality of the smart contract.
   - ❖ Manual review of code, which is the process of reading source code line-by-line to identify potential vulnerabilities.
2. Static, Manual, and Software analysis:
   - ❖ Test coverage analysis is the process of determining whether the test cases are covering the code and how much code is exercised when we run those test cases.
   - ❖ Symbolic execution is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts

Automated 3P frameworks used to assess the smart contract vulnerabilities

- ❖ Slither
- ❖ Consensys MythX, Mythril
- ❖ SWC Registry
- ❖ Solidity Coverage
- ❖ Open Zeppelin Code Analyzer
- ❖ Solidity Code Complier

Risk Classification

Smart contracts are generally designed to manipulate and hold funds denominated in ETH/BNB. This makes them very tempting attack targets, as a successful attack may allow the attacker to directly steal funds from the contract. Below are the typical risk levels of a smart contract:

Vulnerable: A contract is vulnerable if it has been flagged by a static analysis tool as such. As we will see later, this means that some contracts may be vulnerable because of a false positive.

Exploitable: A contract is exploitable if it is vulnerable and the vulnerability could be exploited by an external attacker. For example, if the "vulnerability" flagged by a tool is in a function that requires owning the contract, it would be vulnerable but not exploitable.

Exploited: A contract is exploited if it received a transaction on the main network which triggered one of its vulnerabilities. Therefore, a contract can be vulnerable or even exploitable without having been exploited.

Risk severity	Meaning
! High	This level vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
! Medium	This level vulnerabilities are hard to exploit but very important to fix, they carry an elevated risk of smart contract manipulation, which can lead to high-risk severity
! Low	This level vulnerabilities should be fixed, as they carry an inherent risk of future exploits, and hacks which may or may not impact the smart contract execution.
! Informational	This level vulnerabilities can be ignored. They are code style violations and informational statements in the code. They may not affect the smart contract execution

Static Analysis

Symbol	Meaning
🛑	Function can be modified
💵	Function is payable
🔒	Function is locked
🔐	Function can be accessed
!	Important functionality

MnA	Implementation	IMnA, IERC721Receiver, ERC721Enumerable, Ownable, Pausable		
└	<Constructor>	Public !	🛑	ERC721
└	setContracts	External !	🛑	onlyOwner
└	getTokenWriteBlock	External !		NO!
└	mintForAirdrop	External !	🛑	onlyOwner whenNotPaused
└	claim	External !	🛑	whenNotPaused
└	mint	Public !	🛑	whenNotPaused
└	burn	External !	🛑	whenNotPaused
└	updateOriginAccess	External !	🛑	NO!
└	transferFrom	Public !	🛑	blockIfChangingToken
└	generate	Internal 🔒	🛑	
└	selectTrait	Internal 🔒		
└	selectTraits	Internal 🔒		
└	structToHash	Internal 🔒		
└	isMarine	External !		blockIfChangingToken
└	getMaxTokens	External !		NO!
└	getPaidTokens	External !		NO!
└	setPaidTokens	External !	🛑	onlyOwner
└	setAirdropStage	External !	🛑	onlyOwner
└	setAirdropMintingCount	External !	🛑	onlyOwner
└	setPaused	External !	🛑	requireContractsSet onlyOwner
└	addAdmin	External !	🛑	onlyOwner
└	removeAdmin	External !	🛑	onlyOwner
└	getTokenTraits	External !		blockIfChangingAddress blockIfChangingToken
└	tokenURI	Public !		blockIfChangingAddress blockIfChangingToken
└	tokenOfOwnerByIndex	Public !		blockIfChangingAddress
└	balanceOf	Public !		blockIfChangingAddress
└	ownerOf	Public !		blockIfChangingAddress blockIfChangingToken
└	tokenByIndex	Public !		NO!
└	approve	Public !	🛑	blockIfChangingToken
└	getApproved	Public !		blockIfChangingToken

└	setApprovalForAll	Public ❗	🛑	blockIfChangingAddress
└	isApprovedForAll	Public ❗		blockIfChangingAddress
└	safeTransferFrom	Public ❗	🛑	blockIfChangingToken
└	safeTransferFrom	Public ❗	🛑	blockIfChangingToken
└	onERC721Received	External ❗		NO❗

StakingPool	Implementation	UUPSUpgradeable, OwnableUpgradeable, ReentrancyGuardUpgradeable, IERC721Receiver, PausableUpgradeable, IStakingPool		
└	initialize	Public ❗	🛑	initializer
└	_authorizeUpgrade	Internal 🔒	🛑	onlyOwner
└	setContracts	External ❗	🛑	onlyOwner
└	setTreasureChestId	External ❗	🛑	onlyOwner
└	addManyToMarinePoolAndAlienPool	External ❗	🛑	nonReentrant
└	_addMarineToMarinePool	Internal 🔒	🛑	whenNotPaused _updateEarnings
└	_addAlienToAlienPool	Internal 🔒	🛑	
└	claimManyFromMarinePoolAndAlienPool	External ❗	🛑	whenNotPaused _updateEarnings nonReentrant
└	calculateRewards	External ❗		NO❗
└	_claimMarineFromMarinePool	Internal 🔒	🛑	
└	_claimAlienFromAlienPool	Internal 🔒	🛑	
└	rescue	External ❗	🛑	nonReentrant
└	_payAlienTax	Internal 🔒	🛑	
└	setRescueEnabled	External ❗	🛑	onlyOwner
└	setPaused	External ❗	🛑	requireContractsSet onlyOwner
└	_rankForAlien	Internal 🔒		
└	randomAlienOwner	External ❗		NO❗
└	onERC721Received	External ❗		NO❗

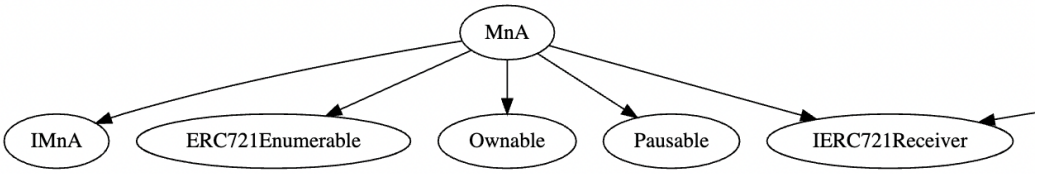
MnAGameCR	Implementation	UUPSUpgradeable, OwnableUpgradeable, ReentrancyGuardUpgradeable, PausableUpgradeable, IMnAGame		
└	initialize	Public ❗	🛑	initializer
└	_authorizeUpgrade	Internal 🔒	🛑	onlyOwner
└	setContracts	External ❗	🛑	onlyOwner
└	getPendingMint	External ❗		NO❗
└	hasMintPending	External ❗		NO❗
└	canMint	External ❗		NO❗
└	addCommitRandom	External ❗	🛑	NO❗
└	deleteCommit	External ❗	🛑	NO❗
└	forceRevealCommit	External ❗	🛑	NO❗
└	mintCommit	External ❗	🛑	whenNotPaused nonReentrant
└	mintReveal	External ❗	🛑	whenNotPaused nonReentrant
└	reveal	Internal 🔒	🛑	
└	mintCost	Public ❗		NO❗
└	selectRecipient	Internal 🔒		
└	setPaused	External ❗	🛑	requireContractsSet onlyOwner
└	setMaxOresCost	External ❗	🛑	requireContractsSet onlyOwner
└	setTreasureChestId	External ❗	🛑	onlyOwner
└	setAllowCommits	External ❗	🛑	onlyOwner

└	setPendingMintAmt	External ❗	🛑	onlyOwner
└	addAdmin	External ❗	🛑	onlyOwner
└	removeAdmin	External ❗	🛑	onlyOwner
└	withdraw	External ❗	🛑	onlyOwner

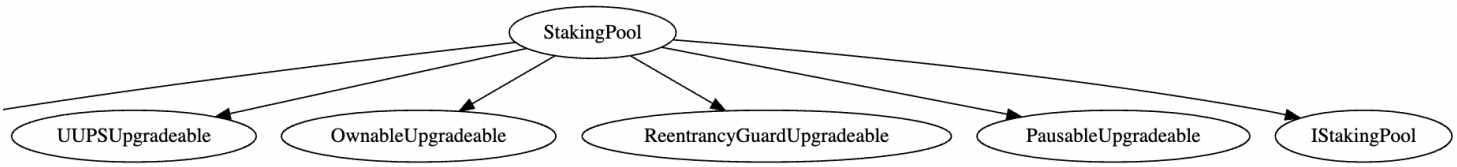
ORES	Implementation	IORES, ERC20, Ownable			
└	<Constructor>	Public ❗	🛑	ERC20
└	addAdmin	External ❗	🛑	onlyOwner
└	removeAdmin	External ❗	🛑	onlyOwner
└	mint	External ❗	🛑	NO❗
└	burn	External ❗	🛑	NO❗
└	transferFrom	Public ❗	🛑	disallowIfStateIsChanging
└	updateOriginAccess	External ❗	🛑	NO❗
└	balanceOf	Public ❗		disallowIfStateIsChanging
└	transfer	Public ❗	🛑	disallowIfStateIsChanging
└	allowance	Public ❗		NO❗
└	approve	Public ❗	🛑	NO❗
└	increaseAllowance	Public ❗	🛑	NO❗
└	decreaseAllowance	Public ❗	🛑	NO❗

Inheritance Graph

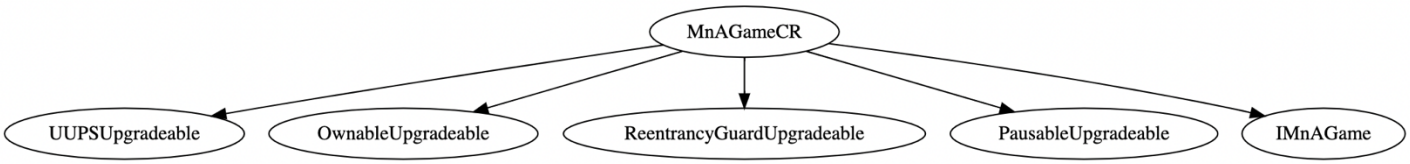
Mna.sol



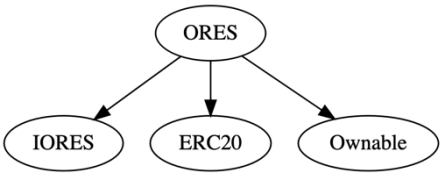
StakingPool.sol



MnaGameCR.sol



ORES.sol



Manual Analysis

Function	Description	Tested	Verdict
Total Supply	provides information about the total token supply	Yes	**Passed**
Balance Of	provides account balance of the owner's account	Yes	**Passed**
Transfer	executes transfers of a specified number of tokens to a specified address	Yes	**Passed**
Approve	allow a spender to withdraw a set number of tokens from a specified account	Yes	**Passed**
Allowance	returns a set number of tokens from a spender to the owner	Yes	**Passed**
Buy Back	is an action in which the project buys back its tokens from the existing holders usually at a market price	NA	NA
Burn	executes transfers of a specified number of tokens to a burn address	Yes	**Passed**
Mint	executes the creation of a specified number of tokens and adds it to the total supply	Yes	**! Low**
Rebase	circulating token supply adjusts (increases or decreases) automatically according to a token's price fluctuations	NA	NA
Blacklist	stops specified wallets from interacting with the smart contract function modules	NA	NA
Pause	stops or locks all function modules of the smart contract	Yes	**! Low**

Function	Description	Tested	Verdict
Dividend	executes transfers of a specified dividend token to a specified address	NA	NA
Airdrop	executes transfers of a specified number of tokens to a specified address	Yes	**Passed**
Max Transaction	a non-whitelisted wallet can only transfer a specified number of tokens	NA	NA
Max Wallet	a non-whitelisted wallet can only hold a specified number of tokens	NA	NA
Cooldown Timer	functionality to limit the number of transactions that a wallet can make within 24-hours	NA	NA
Anti Bot	stops some or all bot wallets from interacting with the smart contract	NA	NA
Anti Snipe	prevents bots from making transaction at "addLiquidity" block	NA	NA
Transfer Ownership	executes transfer of contract ownership to a specified wallet	Yes	**Passed**
Renounce Ownership	executes transfer of contract ownership to a dead address	Yes	**Passed**

Best Practices ✅

- Owner cannot lock or burn the user assets.
- The smart contract utilizes "SafeMath" function to avoid common smart contract vulnerabilities.

Note ⚠️

- ***Be aware that active smart contract owner privileges constitute an elevated impact to smart contract safety and security.***
- Smart contract owner can ***pause*** the smart contract function modules.
- Smart contract owner can ***mint to wallets or mint to airdrop***, this function module is used to mint NFTs.
- Smart contract can ***burn*** tokens from the total supply.
- Smart contract owner can ***explicitly airdrop*** tokens to the specified wallet/wallets.
- Smart contract has a ***low severity issue*** which may or may not create any functional vulnerability.

{

"resource": " /MnAGameCR.sol",

"owner": "_generated_diagnostic_collection_name_#0",

"severity": 8, (! Low Severity)

" Expected token Semicolon got 'LParen'"

"source": "solc",

}

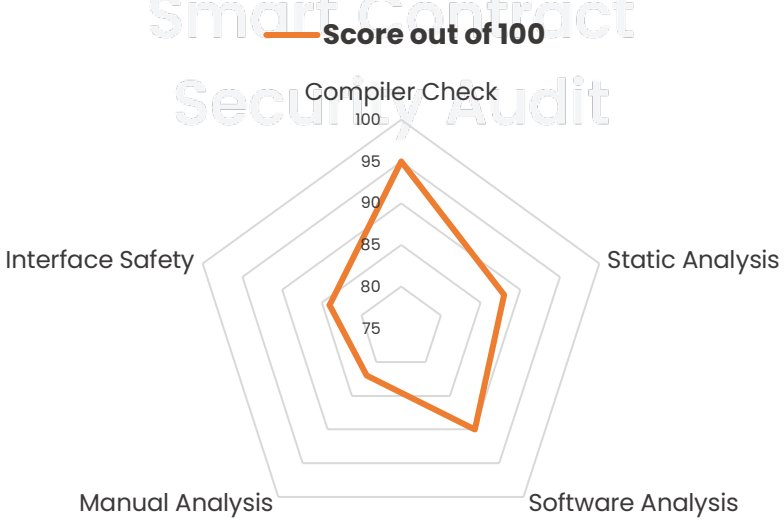
SWC Attacks

SWC ID	Description	Verdict
SWC-101	Integer Overflow and Underflow	**Passed**
SWC-102	Outdated Compiler Version	**! Informational**
SWC-103	Floating Pragma	**Passed**
SWC-104	Unchecked Call Return Value	**Passed**
SWC-105	Unprotected Ether Withdrawal	**Passed**
SWC-106	Unprotected SELF-DESTRUCT Instruction	**Passed**
SWC-107	Re-entrancy	**Passed**
SWC-108	State Variable Default Visibility	**Passed**
SWC-109	Uninitialized Storage Pointer	**Passed**
SWC-110	Assert Violation	**Passed**
SWC-111	Use of Deprecated Solidity Functions	**Passed**
SWC-112	Delegate Call to Untrusted Callee	**Passed**
SWC-113	DoS with Failed Call	**Passed**
SWC-114	Transaction Order Dependence	**Passed**
SWC-115	Authorization through tx.origin	**Passed**
SWC-116	Block values as a proxy for time	**Passed**
SWC-117	Signature Malleability	**Passed**
SWC-118	Incorrect Constructor Name	**Passed**

SWC-119	Shadowing State Variables	**Passed**
SWC-120	Weak Sources of Randomness from Chain Attributes	**Passed**
SWC-121	Missing Protection against Signature Replay Attacks	**Passed**
SWC-122	Lack of Proper Signature Verification	**Passed**
SWC-123	Requirement Violation	**Passed**
SWC-124	Write to Arbitrary Storage Location	**Passed**
SWC-125	Incorrect Inheritance Order	**Passed**
SWC-126	Insufficient Gas Griefing	**Passed**
SWC-127	Arbitrary Jump with Function Type Variable	**Passed**
SWC-128	DoS With Block Gas Limit	**Passed**
SWC-129	Typographical Error	**Passed**
SWC-130	Right-To-Left-Override control character (U+202E)	**Passed**
SWC-131	Presence of unused variables	**! Low**
SWC-132	Unexpected Ether balance	**Passed**
SWC-133	Hash Collisions With Multiple Variable Length Arguments	**Passed**
SWC-134	Message call with the hardcoded gas amount	**Passed**
SWC-135	Code With No Effects (Irrelevant/Dead Code)	**! Low**
SWC-136	Unencrypted Private Data On-Chain	**Passed**

Risk Status & Radar Chart

Risk Severity	Status
! High	No high severity issues identified
! Medium	No medium severity issues identified
! Low	4 low severity issues identified ❖ Please Review Report
! Informational	2 informational severity issues identified ❖ Active Ownership ❖ Outdated Compiler
Verified	54 functions and instances verified and checked



Auditor's Verdict

InterFi team has performed a line-by-line manual analysis and automated review of smart contracts. Smart contracts were analyzed mainly for common contract vulnerabilities, exploits, and manipulation hacks.

- Space Game's solidity source codes have **LOW RISK SEVERITY**
- Space Game's smart contracts have an **ACTIVE OWNERSHIP**
- Space Game's smart contract owner has multiple "Write Contract" privileges. Centralization risk correlated to the active owner is **MEDIUM**

Note for stakeholders

- Be aware that active smart contract owner privileges constitute an elevated impact on smart contract safety and security.
- If the smart contract is not deployed on any blockchain at the time of the audit, the contract can be modified or altered before blockchain development. Verify contract's deployment status in the audit report.
- Make sure that the project team's KYC/identity is verified by an independent firm.
- Always check if the contract's liquidity is locked. A longer liquidity lock plays an important role in the project's longevity. It is recommended to have multiple liquidity providers.
- Examine the unlocked token supply in the owner, developer, or team's private wallets. Understand the project's tokenomics, and make sure the tokens outside of the LP Pair are vested or locked for a longer period.
- Ensure that the project's official website is hosted on a trusted platform, and is using an active SSL certificate. The website's domain should be registered for a longer period.

Important Disclaimer

InterFi Network provides contract development, testing, auditing and project evaluation services for blockchain projects. The purpose of the audit is to analyze the on-chain smart contract source code and to provide a basic overview of the project. **This report should not be transmitted, disclosed, referred to, or relied upon by any person for any purpose without InterFi's prior written consent.**

InterFi provides the easy-to-understand assessment of the project, and the smart contract (otherwise known as the source code). The audit makes no statements or warranties on the security of the code. It also cannot be considered as enough assessment regarding the utility and safety of the code, bug-free status, or any other statements of the contract. While we have used all the data at our disposal to provide the transparent analysis, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts. **Be aware that smart contracts deployed on a blockchain aren't resistant to external vulnerability, or a hack. Be aware that active smart contract owner privileges constitute an elevated impact on smart contract safety and security. Therefore, InterFi does not guarantee the explicit security of the audited smart contract.**

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

This report should not be considered as an endorsement or disapproval of any project or team. The information provided in this report does not constitute investment advice, financial advice, trading advice, or any other sort of advice and you should not treat any of the report's content as such. Do conduct your due diligence and consult your financial advisor before making any investment decisions.

About InterFi Network

InterFi Network provides intelligent blockchain solutions. InterFi is developing an ecosystem that is seamless and responsive. Some of our services: Blockchain Security, Token Launchpad, NFT Marketplace, etc. **InterFi's mission is to interconnect multiple services like Blockchain Security, DeFi, Gaming, and Marketplace under one ecosystem that is seamless, multi-chain compatible, scalable, secure, fast, responsive, and easy to use.**

InterFi is built by a decentralized team of UI experts, contributors, engineers, and enthusiasts from all over the world. Our team currently consists of 6+ core team members, and 10+ casual contributors. **InterFi provides manual, static, and automatic smart contract analysis, to ensure that project is checked against known attacks and potential vulnerabilities.**

To learn more, visit **https://interfi.network**

To view our audit portfolio, visit **https://github.com/interfinetwork**

To book an audit, message **https://t.me/interfiaudits**

@INTERFINETWORK

RELENTLESSLY SECURING THE PUBLIC BLOCKCHAIN | MADE IN CANADA