Report for:

EXAMPLE LTD

June 2021

Version: 2.0

Prepared By: Extropy.IO
Email: info@extropy.io
Telephone: +44 1865261424

Executive Summary

This report presents the findings of the smart contract security assessment conducted on behalf of Example Ltd. The assessment was conducted between 01/05/21 and 08/06/2021 and was authorised by Example Ltd.
I

Assessment Summary

Following the first report, we were supplied with the latest deployed contracts are were able to compile them.

All of the issues raised in the initial report have been resolved

We were not supplied with tests and so are unable to give an opinion on the test coverage or quality of testing.

The following table breaks down the issues which were identified by phase and severity of risk.

Phase	Description	Critical	High	Medium	Low	Info	Total
1	Smart Contracts Audit	0	0	1	5	4	10
2	Final Report	0	0	0	0	0	0

Using This Report

To facilitate the dissemination of the information within this report throughout your organisation, this document has been divided into the following clearly marked and separable sections.

Document Breakdown		
0	Executive Summary	Management level, strategic overview of the assessment and the risks posed to the business
1	Technical Summary	An overview of the assessment from a more technical perspective, including a defined scope and any caveats which may apply
2	Technical Details	Detailed discussion (including evidence and recommendations) for each individual security issue which was identified
3	Methodologies	Audit process and tools used
4	Client Questions	

Document Control

Client Confidentiality

This document contains Client Confidential information and may not be copied without written permission.

Proprietary Information

The content of this document should be considered proprietary information and should not be disclosed outside of Example Ltd.
Extropy gives permission to copy this report for the purposes of disseminating information within your organisation or any regulatory agency.

Document Version Control			
Data Classification	Client Confidential		
Client Name	Example Ltd.		
Document Title	Example Smart Contracts Audit		
Author	Extropy Audit Team		

Document History			
Issue No.	**Issue Date**	**Issued By**	**Change Description**
1.0	01/06/2021	Laurence Kirk	Released to client
2.0	21/06/2021	Laurence Kirk	Released to client

Document Distribution List	
Example Person	CTO Example Ltd.
Laurence Kirk	CEO, Extropy

1. Technical Summary

Extropy was contracted by Example Ltd to conduct a code review and smart contracts vulnerability assessment in order to identify security issues that could negatively affect Example Ltd's business or reputation if they led to the compromise or abuse of systems.

1.1 Scope

Source Units in Scope

Source Units Analyzed: 1
Source Units in Scope: 1 (100%)

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
📝	blockchain/contracts /SPLDistribution.sol	1	———	188	183	147	7	68	📩🌀
📝	**Totals**	**1**	———	**188**	**183**	**147**	**7**	**68**	📩🌀

Design

Contracts Overview

Components

📝Contracts	📚Libraries	🔍Interfaces	🎨Abstract
1	0	0	0

Exposed Functions

This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.

🌐Public	💰Payable
6	0

External	Internal	Private	Pure	View
2	6	0	1	1

StateVariables

Total	🌐Public
11	4

Capabilities

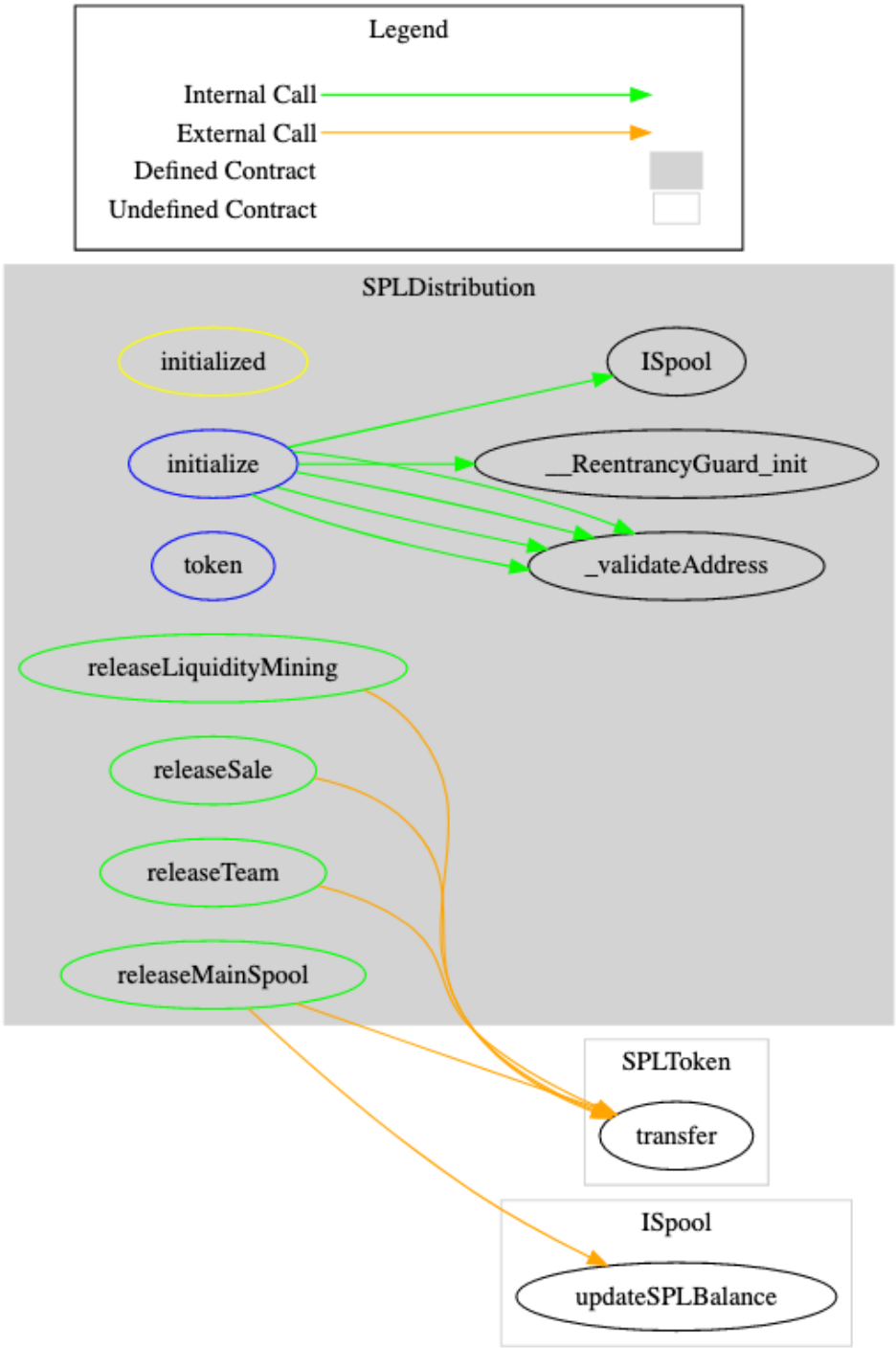
Solidity Versions observed	🧪 Experimental Features	💰 Can Receive Funds	🖥️ Uses Assembly	💣 Has Destroyable Contracts
`0.8.0`		———	**** (0 asm blocks)	———

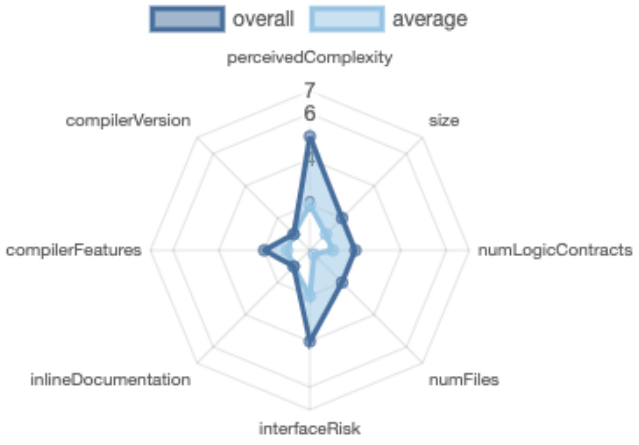
📩 Transfers ETH	⚡ Low-Level Calls	👥 DelegateCall	🧮 Uses Hash Functions	🧪 ECRecover	🌀 New/Create/Create2
`yes`	———	———	———	———	`yes` → `NewContract:SPLToken`

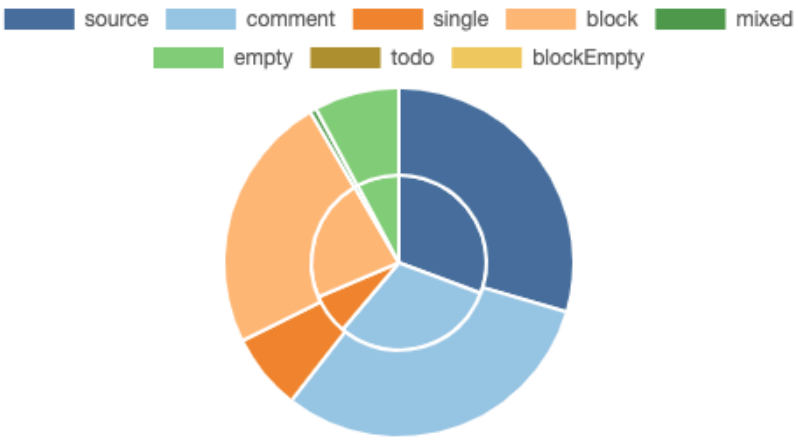
Dependencies / External Imports

Dependency / Import Path	Count
@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol	1

Call Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
SPLDistribution
initialized
ISpool
initialize
__ReentrancyGuard_init
token
_validateAddress
releaseLiquidityMining
releaseSale
releaseTeam
releaseMainSpool
SPLToken
transfer
ISpool
updateSPLBalance


overall
average
perceivedComplexity
7
6
size
numLogicContracts
numFiles
interfaceRisk
inlineDocumentation
compilerFeatures
compilerVersion


source
comment
single
block
mixed
empty
todo
blockEmpty

Contracts Description Table

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
SPLDistribution	Implementation	ReentrancyGuardUpgradeable		
└	initialize	External ❗	🛑	initializer
└	token	External ❗		NO ❗
└	releaseMainSpool	Public ❗	🛑	initialized nonReentrant
└	releaseLiquidityMining	Public ❗	🛑	initialized nonReentrant
└	releaseSale	Public ❗	🛑	initialized nonReentrant
└	releaseTeam	Public ❗	🛑	initialized nonReentrant
└	_validateAddress	Internal 🔒		

1.2 Disclaimer

The audit makes no statements or warranty about utility of the code, safety of the code, suitability of the business model, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only.

2 Technical Findings – Smart Contracts Audit

The remainder of this document is technical in nature and provides additional detail about the items already discussed, for the purposes of remediation and risk assessment.

2.1 External Calls

Risk Rating	Medium

Description:

External function calls may fail or return an error, this should be properly handled in the calling contract

```
Contract1 (lines 97,132,139)
```

Resolved : In this instance, this behaviour is by design.

2.2 Missing function implementations

Risk Rating	Low

Description:

In the following functions implementations were not provided in the supplied code.

```
function 1
function 2
```

Resolved **:** The functions are available in the base contract or UpgradeableProxy.

2.3 Use a fixed and consistent pragma version

Risk Rating	Low

Description:

Different versions of solidity were specified

Resolved **:** A consistent version of Solidity is used throughout

2.4 Prevent Underflow / Overflow

Risk Rating	Low

Description:

Addition of integers can result in overflow

```
adminBalance(addr1, currentBalance(addr2)+amount);
```

Recommendation:

Use Open Zeppelin safe math library or upgrade to solidity version 8

Affects:

Smart Contract
Contract3

2. Tool List

The following tools were used during the assessment:

Tools Used	Description	Resources
Solidity Metrics	Static analysis	https://github.com/ConsenSys/solidity-metrics
Surya	Code visualizer	https://github.com/ConsenSys/surya
SWC Registry	Vulnerability database	https://swcregistry.io/

3 Tailored Methodologies

3.1 Smart Contracts Audit

3.1.1 Audit Goals

We will audit the code in accordance with the following criteria:

Sound Architecture

This audit includes assessments of the overall architecture and design choices. Given the subjective nature of these assessments, it will be up to the development team to determine whether any changes should be made.

Smart Contract Best Practices

This audit will evaluate whether the codebase follows the current established best practices for smart contract development.

Code Correctness

This audit will evaluate whether the code does what it is intended to do.

Code Quality

This audit will evaluate whether the code has been written in a way that ensures readability and maintainability.

Security

This audit will look for any exploitable security vulnerabilities, or other potential threats to the users.

Testing and testability

This audit will examine how easily tested the code is, and review how thoroughly tested the code is.

Although we have commented on the application design, issues of crypto-economics, game theory and suitability for business purposes as they relate to this project are beyond the scope of this audit.

3.2 Test Methodology

The security audit is performed in two phases:

Independent Code Review

The code is inspected separately by four team members checking for software errors and known vulnerabilities.

Static Analysis

The code is subject to static analysis using Slither and Solidity Metrics

Further Questions supplied by Example

Section 1: The smart contract upgradeability mechanism

Evaluate, clarify and provide feedback on the chosen eternal storage upgradeability mechanism:

- **Constants in smart contract etc.**
  - Smart contract constants

    From Solidity Read The Docs

    "The compiler does not reserve a storage slot for these variables, and every occurrence is replaced by the respective value."

- **Clarify that an upgrade of either the contract a or contract b contracts through the upgradeable proxy can only be performed by the admin account**

  - UpgradeableProxy1.sol lines 84-93, require statement makes sure only Admin can call function setImplementation()

```
function seMethod () external {
    require(msg.sender ==admin(), 'only admin ');
```

In order to change admin of the proxy, the current admin has to call etc.