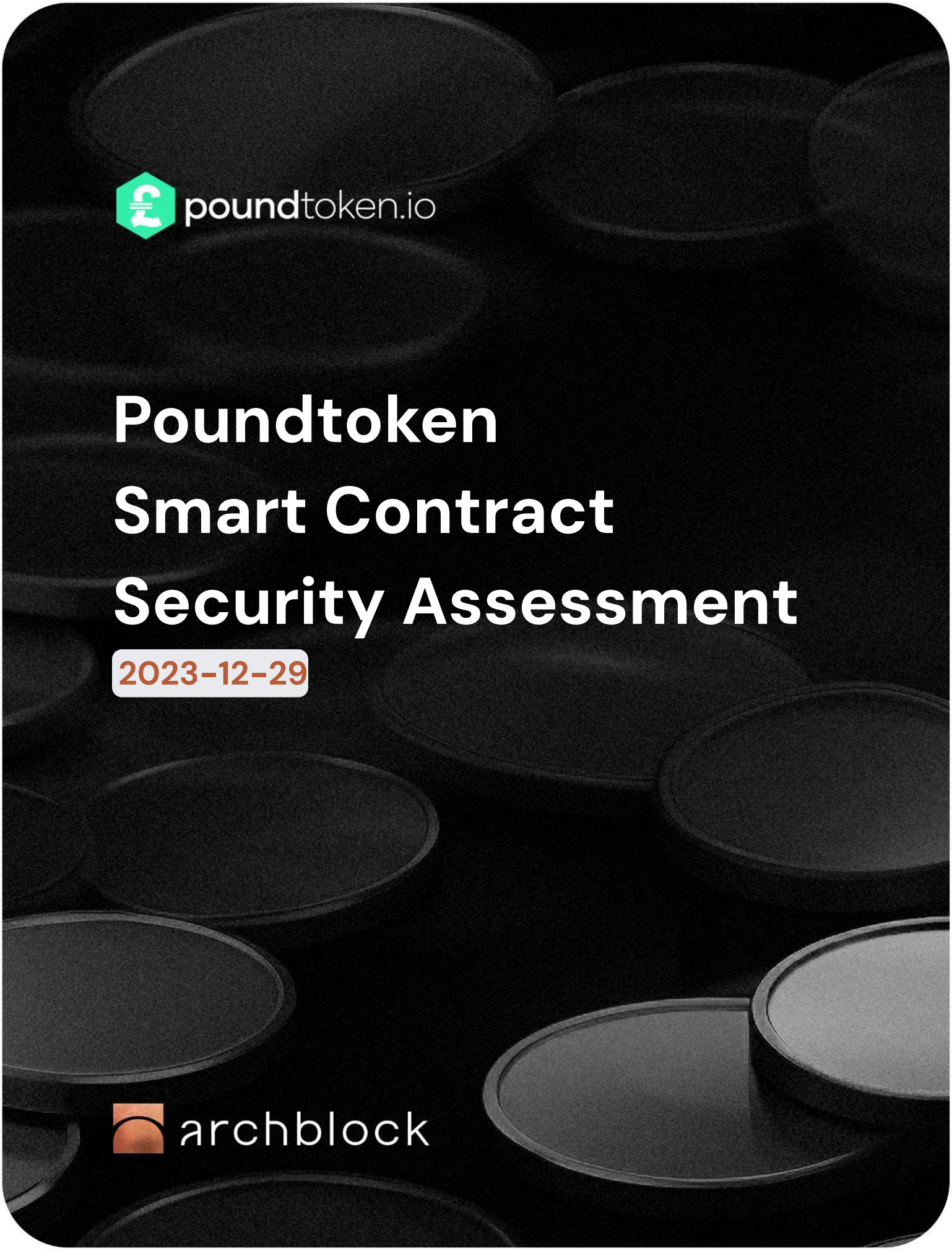
poundtoken.io
Poundtoken
Smart Contract
Security Assessment
2023-12-29
archblock

archblock

1. Executive Summary

Poundtoken (GBPT) is an EVM-based stablecoin with issuance backed by GBP-denominated bank accounts. It has been deployed on Ethereum Mainnet under the following address:

- 0x86B4dBE5D203e634a12364C0e428fa242A3FbA98

Archblock was engaged to perform a security assessment of Poundtoken's currently deployed smart contract code, including the entire token contract but not any governance or configuration mechanisms.

1.1. Findings Overview

Archblock found no changes to the smart contract architecture since the last security audit, dated 2022-05-18:

- https://resources.poundtoken.io/tech-audits/
- https://resources.poundtoken.io/wp-content/uploads/2022/07/Smart-Contract-Source-Code-Review_May22.pdf

The code is clean and sparse. To handle the most critical functionality relating to token transfers and role-based access controls, Poundtoken uses OpenZeppelin, a closely scrutinized and widely adopted smart contract library.

Critical	High	Medium	Low	Informational
0	0	0	0	3

Apart from a small number of Informational comments, Archblock has not found any security issues of significance related to the smart contracts under scope.

2. Scope

Archblock assessed the following targets over two engineer-days, primarily via a manual source code review and static analysis, but supplemented by transaction simulation of key functions.

AdminUpgradeabilityProxy

https://etherscan.io/token/0x86B4dBE5D203e634a12364C0e428fa242A3FbA98

BFTokenUpgradeable

https://etherscan.io/address/0xf08421CE7c9e57aCBd183A92e295809Fcb6325c3

2.1. Excluded from Scope

IssueToAddress: https://etherscan.io/address/0x621f8F8727E87d6F05fd9f03A60b484926FB105A

config:
https://etherscan.io/address/0x483FCcBbEA20FA888e8d97CD27c2BfFd5F237CDa

3. Limitations and Use

Note that this audit is not designed to replace functional tests required before any software release, and does not give any warranties on finding all possible security issues of the given smart contract(s) or blockchain software, i.e., the evaluation result does not guarantee the nonexistence of any further findings of security issues. One audit cannot be considered enough. We always recommend proceeding with several independent audits and a public bug bounty program to improve the security of the code. This report should not be used as investment advice.

Security assessments cannot uncover all existing vulnerabilities; even an assessment in which no vulnerabilities are found is not a guarantee of a secure system. However, code assessments enable the discovery of vulnerabilities that were overlooked during development and areas where additional security measures are necessary. In most cases, applications are either fully protected against a certain type of attack, or they are completely unprotected against it. Some of the issues may affect the entire application, while some lack protection only in certain areas. This is why we carry out a source code assessment aimed at determining all locations that need to be fixed. Within the customer-determined time frame, Archblock has performed an assessment in order to discover as many vulnerabilities as possible.

The focus of our assessment was limited to the code parts defined in the Master Services Agreement entered into by and between Archblock and Blackfridge on 2023-12-29. We assessed whether the project follows the provided specifications. These assessments are based on the provided threat model and trust assumptions. We draw attention to the fact that due to inherent limitations in any software development process and software product, an inherent risk exists that even major failures or malfunctions can remain undetected. Further uncertainties exist in any software product or application used during the development, which itself cannot be free from any error or failures. These preconditions can have an impact on the system's code and/or functions and/or operation. We did not assess the underlying third-party infrastructure which adds further inherent risks as we rely on the correct execution of the included third-party technology stack itself. Report readers should also take into account that over the life cycle of any software, changes to the product itself or to the environment in which it is operated can have an impact leading to operational behaviors other than those initially determined in the business specification.

4. Terminology

Archblock uses CVSS terminology to classify the Severity of a finding, which is based on

- Likelihood: how likely this finding will be exploited
- Impact: the consequences of an exploit of this finding

SEVERITY		Impact		
		High	Medium	Low
Likelihood	High	Critical	High	Medium
	Medium	High	Medium	Low
	Low	Medium	Low	Low

Archblock also uses an additional classification for Informational findings, which pose no known security vulnerabilities but represent improvements that can be made to security posture.

5. Critical Findings

6. High Findings

7. Medium Findings

8. Low Findings

9. Informational Findings

9.1. No Etherscan-verified source code for IssueToAddress or config

Poundtoken's IssueToAddress and config addresses do not have their source code verified on Etherscan. For any deployed smart contracts, we highly recommend publishing source code with corresponding compiler settings. Verifying on Etherscan ensures that the smart contract bytecode deployed to a particular address matches the expected source code in the repository.

This is considered a standard practice adopted across the EVM-based smart contract industry, to prevent supply-chain or insider attacks and allow security assessors to make responsible vulnerability disclosures on smart contracts they come across.

9.2. No public source code repository with commit hash

Poundtoken does not publish a public source code repository, with a specified commit hash for each release, to clarify what exact version of code a security assessment has been performed on. This could have allowed either a rogue insider or compromised development machine to deploy a maliciously crafted smart contract on-chain that resembles the assessed code, except with nearly imperceptible changes that make it vulnerable to exploit.

This is considered a standard practice adopted across the EVM-based smart contract industry, to allow anyone to confirm that the assessed code matches what has been deployed on-chain.

9.3. Documentation is sparse and does not follow NatSpec formatting

BFTokenUpgradeable and its imports are fairly sparsely documented. Code documentation should primarily explain the code's intent, and, in particular, why certain design decisions have been made. This assists in determining whether the code itself works as intended.

In addition, it is a standard practice to use NatSpec formatting to annotate Solidity functions.