ClarityAlliance

PYTH ORACLE CLIENT SECURITY REVIEW

Conducted by:
KRISTIAN APOSTOLOV, ALIN BARBATEI (ABA)

DECEMBER 8TH, 2024

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
    5.1. Impact 4
    5.2. Likelihood 5
    5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
    [C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
    [H-01] Absence of Pyth Stacks Governance Module 12
    [H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
    [H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
    [M-01] Potential Use of Stale Price When Updating Price 17
    [M-02] Price Update Logic May Cause Denial of Service 19
    [M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
    [M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
    [L-01] Inability to Deactivate Price Update Fee 22
    [L-02] Default Price Update Fee Differs From Documentation 23
    [L-03] Governance Updated Principals Are Not Validated 24
    [L-04] Parallel Governance Proposals Can Be Blocked 25
    [L-05] Incorrect Validation of Guardian Set Index Update 26
    [L-06] Incorrect Validation of Guardian Set ChainId 27
    [L-07] Missing Implicit Stale Price Checking API 28
    [L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
    [L-09] Missing Overlay Checks on V AA Payloads 30
    [L-10] Incorrect Validation of Minor Version When Updating Price 32
    [L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
    [L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
    [L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
    [QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
    [QA-02] Leftover Bitcoin Timestamp Code Usage 38
    [QA-03] Project Call To Action References Outdated Repository 39
    [QA-04] Error Code Inconsistencies 40
    [QA-05] Redeploy Dependency Contracts for Optimization 42
    [QA-06] Eliminate Unused Constants 43
    [QA-07] Redundant Tuple with One Element as Map Key 44
    [QA-08] Incorrect Naming of Update Function Events 45
    [QA-09] Inconsistent Return Values in Governance Update Functions 46
    [QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
    [QA-11] Misleading, Outdated, or Incomplete Comments 48
    [QA-12] Use Constants Where Appropriate 49
    [QA-13] Simplification Opportunities in Code Operations 50
    [QA-14] Typographical Errors 51
    [QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
    [QA-16] AUWV Price Feed Update Length Is Not Validated 53
    [QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

1. About Clarity Alliance

Clarity Alliance is a team of expert whitehat hackers specialising in securing protocols on Stacks.

They have disclosed vulnerabilities that have saved millions in live TVL and conducted thorough reviews for some of the largest projects across the Stacks ecosystem.

Learn more about Clarity Alliance at clarityalliance.org.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

2. Disclaimer

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 Clarity Alliance 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 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.

Blockchain technology and cryptographic assets present a high level of ongoing risk. Clarity Alliance’s position is that each company and individual are responsible for their own due diligence and continuous security. Clarity Alliance’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree
to analyze.

The assessment services provided by Clarity Alliance are subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis.

Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third parties. Notice that smart contracts deployed on the blockchain are not resistant from internal/external exploit. Notice that active smart contract owner privileges constitute an elevated impact to any smart contract’s safety and security. Therefore, Clarity Alliance does not guarantee the explicit security of the audited smart contract, regardless of the verdict.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

3. Introduction

A time-boxed security review of the Pyth Oracle Client implementation, where Clarity Alliance reviewed the scope, whilst simultaneously building out a testing suite for the protocol.

4. About Pyth Oracle

Pyth Network is an oracle that publishes financial market data to multiple blockchains. The market data is contributed by over 80 first-party publishers, including some of the biggest exchanges and market-making firms in the world. Pyth offers price feeds for several asset classes, including US equities, commodities, and cryptocurrencies. Each price feed publishes a robust aggregate of publisher prices that updates multiple times per second. Price feeds are available on multiple blockchains and can be used in off-chain applications.

5. Risk Classification

Severity	Impact: High	Impact: Medium	Impact: Low
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
   5.1. Impact 4
   5.2. Likelihood 5
   5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

5.1 Impact

- High - leads to a significant material loss of assets in the protocol or significantly harms a group of users.
- Medium - only a small amount of funds can be lost (such as leakage of value) or a core functionality of the protocol is affected.
- Low - can lead to any kind of unexpected behavior with some of the protocol's functionalities that's not so critical.

5.2 Likelihood

- High - attack path is possible with reasonable assumptions that mimic on-chain conditions, and the cost of the attack is relatively low compared to the amount of funds that can be stolen or lost.
- Medium - only a conditionally incentivized attack vector, but still relatively likely.
- Low - has too many or too unlikely assumptions or requires a significant stake by the attacker with little or no incentive.

5.3 Action required for severity levels

- Critical - Must fix as soon as possible (if already deployed)
- High - Must fix (before deployment if not already deployed)
- Medium - Should fix
- Low - Could fix

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

6. Security Assessment Summary

Review Commit Hash:
086fff092ef01392a51b96a8972b03b4599945b9

Scope

The following contracts were in the scope of the security review:

- `pyth-governance-v1.clar`
- `pyth-store-v1.clar`
- `pyth-pnau-decoder-v1.clar`
- `pyth-traits-v1.clar`
- `pyth-oracle-v2.clar`
- `pyth-p2wh-decoder-v1.clar`
- `wormhole/wormhole-core-v2.clar`
- `wormhole/wormhole-traits-v1.clar`

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

7. Executive Summary

Over the course of the security review, Kristian Apostolov, Alin Barbatei (ABA) engaged with Trust Machines to review Pyth Oracle. In this period of time a total of **38** issues were uncovered.

Protocol Summary

Protocol Name	Pyth Oracle
Repository	https://github.com/Trust-Machines/stacks-pyth-bridge
Date	December 8th, 2024
Protocol Type	Oracle Client

Findings Count

Severity	Amount
Critical	1
High	3
Medium	4
Low	13
QA	17
Total Findings	**38**

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

Summary of Findings

ID	Title	Severity	Status
[C-01]	Attacker Can Corrupt Guardian Set During Update	Critical	Resolved
[H-01]	Absence of Pyth Stacks Governance Module	High	Acknowledged
[H-02]	Wormhole Contract Vulnerable to Hijacking at Deployment	High	Resolved
[H-03]	Limited Price Updates Due to High Runtime Costs	High	Acknowledged
[M-01]	Potential Use of Stale Price When Updating Price	Medium	Resolved
[M-02]	Price Update Logic May Cause Denial of Service	Medium	Resolved
[M-03]	Changing Governance Data Source May Cause Denial of Service in Operations	Medium	Resolved
[M-04]	Price Cannot Be Updated During Guardian Set Transition Period	Medium	Resolved
[L-01]	Inability to Deactivate Price Update Fee	Low	Resolved
[L-02]	Default Price Update Fee Differs From Documentation	Low	Resolved
[L-03]	Governance Updated Principals Are Not Validated	Low	Resolved
[L-04]	Parallel Governance Proposals Can Be Blocked	Low	Acknowledged
[L-05]	Incorrect Validation of Guardian Set Index Update	Low	Resolved
[L-06]	Incorrect Validation of Guardian Set ChainId	Low	Resolved
[L-07]	Missing Implicit Stale Price Checking API	Low	Resolved
[L-08]	Reconsider Default Fee Receiver and Stale Price Threshold	Low	Acknowledged
[L-09]	Missing Overlay Checks on V AA Payloads	Low	Resolved
[L-10]	Incorrect Validation of Minor Version When Updating Price	Low	Resolved
[L-11]	Wormhole Guardian Set Can Be Updated With An Empty Set	Low	Resolved
[L-12]	Wormhole Guardian Set Can Contain Duplicate Entries	Low	Resolved
[L-13]	PTGM Price Data Sources Length Is Not Validated	Low	Resolved

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

Summary of Findings

ID	Title	Severity	Status
[QA-01]	Remove Outdated pyth-p2wh-decoder-v1 Contract	QA	Resolved
[QA-02]	Leftover Bitcoin Timestamp Code Usage	QA	Resolved
[QA-03]	Project Call To Action References Outdated Repository	QA	Resolved
[QA-04]	Error Code Inconsistencies	QA	Resolved
[QA-05]	Redeploy Dependency Contracts for Optimization	QA	Acknowledged
[QA-06]	Eliminate Unused Constants	QA	Resolved
[QA-07]	Redundant Tuple with One Element as Map Key	QA	Resolved
[QA-08]	Incorrect Naming of Update Function Events	QA	Resolved
[QA-09]	Inconsistent Return Values in Governance Update Functions	QA	Resolved
[QA-10]	Inconsistent Reference to Pyth State Bearing Contract	QA	Resolved
[QA-11]	Misleading, Outdated, or Incomplete Comments	QA	Resolved
[QA-12]	Use Constants Where Appropriate	QA	Resolved
[QA-13]	Simplification Opportunities in Code Operations	QA	Resolved
[QA-14]	Typographical Errors	QA	Resolved
[QA-15]	Merkle Implementation Can Invalidate Correct Price Updates	QA	Acknowledged
[QA-16]	AUWV Price Feed Update Length Is Not Validated	QA	Resolved
[QA-17]	Price Update Can Be From The Future	QA	Acknowledged

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
    5.1. Impact 4
    5.2. Likelihood 5
    5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
  [C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
  [H-01] Absence of Pyth Stacks Governance Module 12
  [H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
  [H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
  [M-01] Potential Use of Stale Price When Updating Price 17
  [M-02] Price Update Logic May Cause Denial of Service 19
  [M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
  [M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
  [L-01] Inability to Deactivate Price Update Fee 22
  [L-02] Default Price Update Fee Differs From Documentation 23
  [L-03] Governance Updated Principals Are Not Validated 24
  [L-04] Parallel Governance Proposals Can Be Blocked 25
  [L-05] Incorrect Validation of Guardian Set Index Update 26
  [L-06] Incorrect Validation of Guardian Set ChainId 27
  [L-07] Missing Implicit Stale Price Checking API 28
  [L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
  [L-09] Missing Overlay Checks on V AA Payloads 30
  [L-10] Incorrect Validation of Minor Version When Updating Price 32
  [L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
  [L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
  [L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
  [QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
  [QA-02] Leftover Bitcoin Timestamp Code Usage 38
  [QA-03] Project Call To Action References Outdated Repository 39
  [QA-04] Error Code Inconsistencies 40
  [QA-05] Redeploy Dependency Contracts for Optimization 42
  [QA-06] Eliminate Unused Constants 43
  [QA-07] Redundant Tuple with One Element as Map Key 44
  [QA-08] Incorrect Naming of Update Function Events 45
  [QA-09] Inconsistent Return Values in Governance Update Functions 46
  [QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
  [QA-11] Misleading, Outdated, or Incomplete Comments 48
  [QA-12] Use Constants Where Appropriate 49
  [QA-13] Simplification Opportunities in Code Operations 50
  [QA-14] Typographical Errors 51
  [QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
  [QA-16] AUWV Price Feed Update Length Is Not Validated 53
  [QA-17] Price Update Can Be From The Future 54

8. Findings

8.1. Critical Findings

[C-01] Attacker Can Corrupt Guardian Set During Update

Description

The Wormhole guardians are responsible for validating any VAA (Verified Action Approval) messages. These messages include both governance updates to the Pyth instance and price updates.

When updating the guardian set in the `wormhole-core-v2` contract via the `update-guardians-set` function, the current logic operates as follows:

- The `update-guardians-set` function accepts a VAA containing the new guardian set's Ethereum equivalent addresses (`guardian-set-vaa`) and the public keys corresponding to the new guardian addresses (`uncompressed-public-keys`).
- Calling `update-guardians-set` is permissionless, as it assumes sufficient validation is performed on the VAA itself.
- The VAA is validated to have been issued by the previous guardian set, which is appropriate and standard.
- The resulting Ethereum equivalent addresses are then compared to the provided public keys and input to ensure they match.
- Matched pairs are then directly saved to storage.

The critical issue with this mechanism is that it assumes the resulting matched pairs from the `check-and-consolidate-public-keys` function call are validated, which they are not.

In `check-and-consolidate-public-keys`, if a key pair does not match, an empty entry is added to the result without any validation.

```
(entry (if
  (is-eth-address-matching-public-key uncompressed-public-key eth-address)
        { compressed-public-key: compressed-public-key, uncompressed-public-key: unco
    { compressed-public-key: 0x, uncompressed-public-key: 0x })))
;; ... code ...
result: (unwrap-panic (as-max-len? (append (get result acc) entry) u19)),
```

From a high-level perspective, during an update, a malicious actor can intercept a valid VAA message generated by the Wormhole guardians and submit it with invalid, random public keys. The resulting array of zeroed addresses would then be saved as valid guardians.

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

Consequently, any and all VAA messages would become invalid, permanently disabling the contract. The damage is irreversible, as even a new guardian set cannot be added since they must also be signed by the invalid, zeroed guardians.

Recommendation

In the `wormhole-core-v2` contract, within the `update-guardians-set` function, first filter the resulting `consolidated-public-keys` to remove any zeroed entries.

Secondly, after removing the empty entries, verify that the length of the filtered `consolidated-public-keys` matches the length of the Ethereum addresses extracted from the valid VAA message.

Without this second check, updates with an arbitrary number of valid guardians could still be created.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

8.2. High Findings

[H-01] Absence of Pyth Stacks Governance Module

Description

The Pyth Network has a governance mechanism that enables it to send specific commands to existing implementations on each supported chain. However, this mechanism does not currently support interaction with the Stacks governance contract `pyth-governance-v1`, as it requires the addition of a new, custom Governance Module to the core Pyth codebase.

Pyth developers have allocated IDs for the Stacks ecosystem in PR#1158 and PR#1168, but a governance module implementation is still needed to enable command transmission.

The existing `pyth-governance-v1` contract correctly verifies that the issued Verified Action Approvals messages contain the correct targeted chain and originate from valid emitters:

```
;; Stacks chain id attributed by Pyth
(define-constant EXPECTED_CHAIN_ID (if is-in-mainnet 0xea86 0xc377))

;; ... code ...

;; Check target-chain-id
(asserts! (is-eq
  (get value cursor-target-chain-id) EXPECTED_CHAIN_ID) ERR_INVALID_PTGM)
;; Check module
(asserts! (is-eq (get value cursor-module) EXPECTED_MODULE) ERR_INVALID_PTGM)
```

Currently, the system can be deployed and operate with the existing defaults, but no settings can be updated or changed. For instance, this includes a 10 units fee sent to a hardcoded address, with no means to alter them.

Additionally, due to the absence of a Pyth Stacks governance module, the current governance commands are arbitrary and must align with the future governance implementation.

Recommendation

Collaborate with the Pyth cross-chain developers to integrate the custom Stacks blockchain governance module. Subsequently, ensure that the `pyth-governance-v1` implementation adheres to the required interface.

Given the potential length of this process, a temporary solution could involve using a different, standard governance contract until the Pyth Governance Module is completed.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
   5.1. Impact 4
   5.2. Likelihood 5
   5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

This is a highly sensitive issue, as the community must understand that until Pyth assumes full ownership of the contracts, the current implementation—while correct and valid—is not directly managed by Pyth. It solely utilizes their data feeds for price updates.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment

Description

The Wormhole guardian set is responsible for validating any VAA (Verified Action Approval) messages. These messages include governance updates to the Pyth instance and price updates.

When updating the guardian set in the `wormhole-core-v2` contract via the `update-guardians-set` function for the first time, the initial absence of guardians results in the VAA message guardian validation being bypassed.

```
(let ((vaa (if (var-get guardian-set-initialized)
        (try! (parse-and-verify-vaa guardian-set-vaa))
        (get vaa (try! (parse-vaa guardian-set-vaa)))))
```

This design assumes that the first caller is trustworthy and will correctly set the initial guardians.

However, the issue arises because the `update-guardians-set` function is permissionless. Consequently, after deploying the `wormhole-core-v2` contract, the first caller to execute the update function can set any arbitrary guardians.

A malicious actor could front-run the `update-guardians-set` call and assign their controlled addresses as guardians. By controlling the guardians, they could then generate any price feed equivalent outside the Wormhole ecosystem and manipulate the resulting price.

Recommendation

We propose two potential solutions.

Firstly, when deploying the `wormhole-core-v2` contract, record the deployer's principal. Ensure that the first call to `update-guardians-set` is made by the deployer.

The second solution is to remove the differential behavior in the `update-guardians-set` function and directly hardcode the initial guardian set in the `guardians-sets` map. Although this approach is simpler, it will significantly increase the overall contract size, slightly raising execution fees for each price update call.

CONTENTS

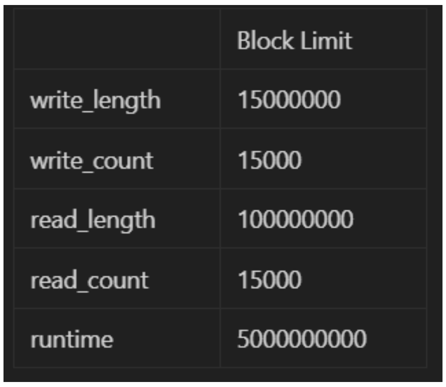
1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
  5.1. Impact 4
  5.2. Likelihood 5
  5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[H-03] Limited Price Updates Due to High Runtime Costs

Description

Pyth operates as a pull-type oracle, allowing anyone to update asset prices by submitting valid price update payloads for each asset. However, the current implementation is highly runtime-intensive, making it challenging to update even a single asset at a time due to the associated costs.

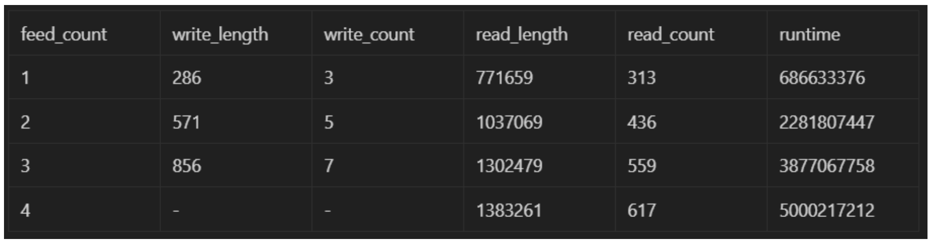
Stacks imposes several execution costs and limits per block. Specifically:



	Block Limit
write_length	15000000
write_count	15000
read_length	100000000
read_count	15000
runtime	5000000000

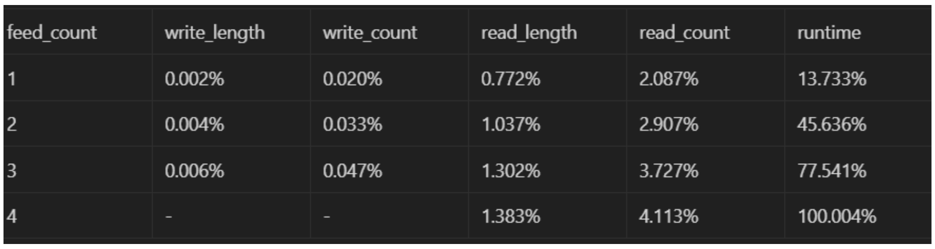
Benchmark results indicate that the runtime execution limit for an entire block is exceeded when updating four different assets (price feeds) simultaneously.

The raw benchmark results are as follows:



feed_count	write_length	write_count	read_length	read_count	runtime
1	286	3	771659	313	686633376
2	571	5	1037069	436	2281807447
3	856	7	1302479	559	3877067758
4	-	-	1383261	617	5000217212

Expressing these results as percentages of the block limits, we find (no writes occurred in the last case as it reverted beforehand):



feed_count	write_length	write_count	read_length	read_count	runtime
1	0.002%	0.020%	0.772%	2.087%	13.733%
2	0.004%	0.033%	1.037%	2.907%	45.636%
3	0.006%	0.047%	1.302%	3.727%	77.541%
4	-	-	1.383%	4.113%	100.004%

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
   5.1. Impact 4
   5.2. Likelihood 5
   5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
- **[C-01]** Attacker Can Corrupt Guardian Set During Update 10

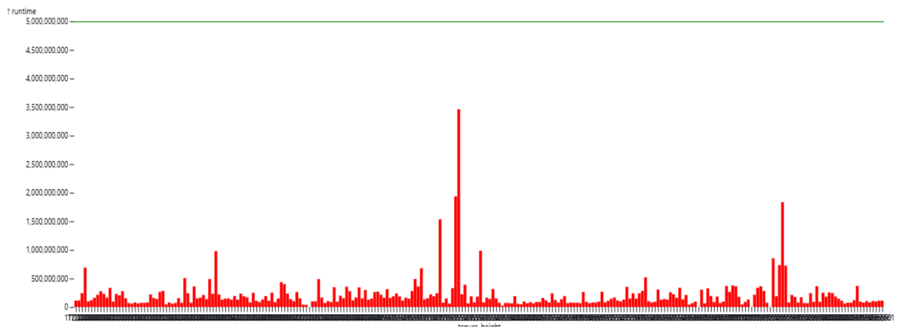
8.2. High Findings 12
- **[H-01]** Absence of Pyth Stacks Governance Module 12
- **[H-02]** Wormhole Contract Vulnerable to Hijacking at Deployment 14
- **[H-03]** Limited Price Updates Due to High Runtime Costs 15

8.3. Medium Findings 17
- **[M-01]** Potential Use of Stale Price When Updating Price 17
- **[M-02]** Price Update Logic May Cause Denial of Service 19
- **[M-03]** Changing Governance Data Source May Cause Denial of Service in Operations 20
- **[M-04]** Price Cannot Be Updated During Guardian Set Transition Period 21

8.4. Low Findings 22
- **[L-01]** Inability to Deactivate Price Update Fee 22
- **[L-02]** Default Price Update Fee Differs From Documentation 23
- **[L-03]** Governance Updated Principals Are Not Validated 24
- **[L-04]** Parallel Governance Proposals Can Be Blocked 25
- **[L-05]** Incorrect Validation of Guardian Set Index Update 26
- **[L-06]** Incorrect Validation of Guardian Set ChainId 27
- **[L-07]** Missing Implicit Stale Price Checking API 28
- **[L-08]** Reconsider Default Fee Receiver and Stale Price Threshold 29
- **[L-09]** Missing Overlay Checks on V AA Payloads 30
- **[L-10]** Incorrect Validation of Minor Version When Updating Price 32
- **[L-11]** Wormhole Guardian Set Can Be Updated With An Empty Set 33
- **[L-12]** Wormhole Guardian Set Can Contain Duplicate Entries 34
- **[L-13]** PTGM Price Data Sources Length Is Not Validated 36

8.5. QA Findings 37
- **[QA-01]** Remove Outdated pyth-p2wh-decoder-v1 Contract 37
- **[QA-02]** Leftover Bitcoin Timestamp Code Usage 38
- **[QA-03]** Project Call To Action References Outdated Repository 39
- **[QA-04]** Error Code Inconsistencies 40
- **[QA-05]** Redeploy Dependency Contracts for Optimization 42
- **[QA-06]** Eliminate Unused Constants 43
- **[QA-07]** Redundant Tuple with One Element as Map Key 44
- **[QA-08]** Incorrect Naming of Update Function Events 45
- **[QA-09]** Inconsistent Return Values in Governance Update Functions 46
- **[QA-10]** Inconsistent Reference to Pyth State Bearing Contract 47
- **[QA-11]** Misleading, Outdated, or Incomplete Comments 48
- **[QA-12]** Use Constants Where Appropriate 49
- **[QA-13]** Simplification Opportunities in Code Operations 50
- **[QA-14]** Typographical Errors 51
- **[QA-15]** Merkle Implementation Can Invalidate Correct Price Updates 52
- **[QA-16]** AUWV Price Feed Update Length Is Not Validated 53
- **[QA-17]** Price Update Can Be From The Future 54

Reviewing current Stacks Space Usage, we observe that runtime usage is generally low, typically under 10%, with a maximum observed around 70%.



In practice, miners can include at most one transaction updating three feeds simultaneously (77% usage) or up to seven individual transactions updating one price each (96% usage) per block. The cost increase per call is exponential as more data is processed when multiple feeds are bundled together.

This limitation restricts Pyth to updating a maximum of seven assets per block. As a general-purpose oracle, this is a significant constraint, as the entire blockchain cannot be limited to such a small number of assets. In practice, more assets can be updated if spread across multiple blocks, considering staleness.

Recommendation

A more runtime-efficient implementation is necessary. While the Stacks blockchain is still expanding, it may be possible to mitigate this issue by developing automated scripts that distribute asset updates over several blocks, leveraging staleness to provide valid but less frequent updates. However, this is not a viable long-term solution.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

8.3. Medium Findings

[M-01] Potential Use of Stale Price When Updating Price

Description

When updating a price in the Pyth storage contract, a staleness check is performed to ensure the updated price is not outdated.

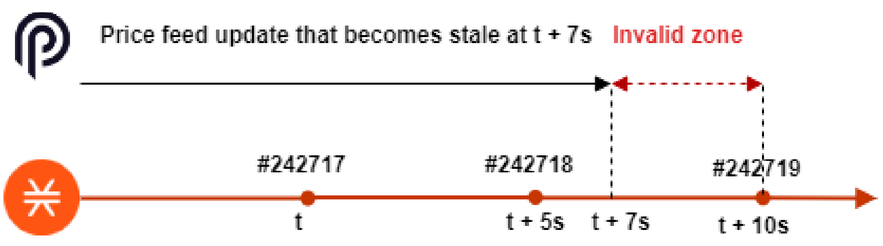
```
(let ((stale-price-threshold
  (contract-call? .pyth-governance-v1 get-stale-price-threshold))
      (latest-bitcoin-timestamp (unwrap! (get-stacks-block-info? time
        (- stacks-block-height u1)) ERR_STALE_PRICE)))
  ;; ... code ...
  ;; Ensure that price is not stale
(asserts! (>= (get publish-time entry)
  (- latest-bitcoin-timestamp stale-price-threshold)) ERR_STALE_PRICE)
```

Note that `latest-bitcoin-timestamp` is actually `latest-stacks-timestamp`, as mentioned in another issue.

The timestamp for the last Stacks block is used instead of the current one because it is not possible to obtain `stacks-block-height` block in which a transaction is currently being processed. The timestamp is only generated after the block is committed to the chain.

This design flaw allows stale prices to be used when updating prices in a block that, once committed to the blockchain, would exceed the stale limit.

Consider the following image and example for theoretical Stacks blocks #242717, #242718, and #242719:



- While execution is in block #242719, the current implementation checks if price feed is stale against the previous block's timestamp
- The previous block, #242718, has the timestamp `t + 5s`, which is within the staleness check limits and passes.
- The chronological staleness check would fail at `t + 7s`, which is in the next block. However, since the current block cannot access this time, the update is considered valid.

CONTENTS
1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

In theory, Stacks blocks are minted every 5 seconds, but real-time data shows variations of up to tens of seconds between blocks.

Another scenario occurs when the Stacks blockchain stops producing blocks and resumes after a significant delay. The price feeds would still be considered valid, as they are compared against the previous block's timestamp. A real-life example of this:

- Stacks block #242879 was mined at `14:22:53 2024.11.21`.
- The next Stacks block #242880 was minted at `14:47:42 2024.11.21`, 25 minutes later.
- Any feed price updates in `#242880` would use the timestamp of block `#242879`, adding 25 minutes to any staleness check.

This delay occurred because Bitcoin block #871339, which anchored the last Stacks block `#242879`, and block #871340 were 25 minutes apart.

Such delays in Bitcoin blocks are not uncommon. Another incident involved Bitcoin blocks #867863 and #867864, which were 30 minutes apart.

The impact is that any price update transaction created between the staleness check limit and the current block commit (the **Invalid zone**) would be considered valid, even if the price is logically stale. This may lead to DeFi protocols using invalid, stale prices, potentially resulting in financial loss for users.

Recommendation

Currently, there is no mechanism to determine time-related information from code running in a transaction being executed in the latest block.

A workaround involves using a variable to denote the Stacks block time and considering it when checking staleness:

```
(define-constant STACKS_BLOCK_TIME u5)

(asserts! (>= (get publish-time entry) (+
  (- latest-bitcoin-timestamp stale-price-threshold) STACKS_BLOCK_TIME)) ERR_STALE_PRI
```

While the example snippet uses a constant 5 seconds to denote Stacks block time, a more robust implementation would involve a variable that can be adjusted by governance to account for different scenarios and blockchain states.

CONTENTS

1. About Clarity Alliance	2
2. Disclaimer	3
3. Introduction	4
4. About Pyth Oracle	4
5. Risk Classification	4
5.1. Impact	4
5.2. Likelihood	5
5.3. Action required for severity levels	5
6. Security Assessment Summary	6
7. Executive Summary	7
8. Summary of Findings	8
8.1. Critical Findings	10
[C-01] Attacker Can Corrupt Guardian Set During Update	10
8.2. High Findings	12
[H-01] Absence of Pyth Stacks Governance Module	12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment	14
[H-03] Limited Price Updates Due to High Runtime Costs	15
8.3. Medium Findings	17
[M-01] Potential Use of Stale Price When Updating Price	17
[M-02] Price Update Logic May Cause Denial of Service	19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations	20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period	21
8.4. Low Findings	22
[L-01] Inability to Deactivate Price Update Fee	22
[L-02] Default Price Update Fee Differs From Documentation	23
[L-03] Governance Updated Principals Are Not Validated	24
[L-04] Parallel Governance Proposals Can Be Blocked	25
[L-05] Incorrect Validation of Guardian Set Index Update	26
[L-06] Incorrect Validation of Guardian Set ChainId	27
[L-07] Missing Implicit Stale Price Checking API	28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold	29
[L-09] Missing Overlay Checks on V AA Payloads	30
[L-10] Incorrect Validation of Minor Version When Updating Price	32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set	33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries	34
[L-13] PTGM Price Data Sources Length Is Not Validated	36
8.5. QA Findings	37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract	37
[QA-02] Leftover Bitcoin Timestamp Code Usage	38
[QA-03] Project Call To Action References Outdated Repository	39
[QA-04] Error Code Inconsistencies	40
[QA-05] Redeploy Dependency Contracts for Optimization	42
[QA-06] Eliminate Unused Constants	43
[QA-07] Redundant Tuple with One Element as Map Key	44
[QA-08] Incorrect Naming of Update Function Events	45
[QA-09] Inconsistent Return Values in Governance Update Functions	46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract	47
[QA-11] Misleading, Outdated, or Incomplete Comments	48
[QA-12] Use Constants Where Appropriate	49
[QA-13] Simplification Opportunities in Code Operations	50
[QA-14] Typographical Errors	51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates	52
[QA-16] AUWV Price Feed Update Length Is Not Validated	53
[QA-17] Price Update Can Be From The Future	54

[M-02] Price Update Logic May Cause Denial of Service

Description

The current implementation of Pyth for the Stacks ecosystem primarily offers two APIs:

- One for retrieving the current price via: `pyth-store-v2::read-price-feed`
- Another for updating the price via: `pyth-oracle-v2::verify-and-update-price-feeds`

The Pyth pull oracle is widely used across numerous blockchains. The price update API is particularly important and is expected to function consistently across all blockchains.

However, the current Stacks implementation deviates from the known behavior when updating price feeds if no valid feeds are provided.

In the EVM API, the price update function call will succeed even if the update is ignored. In contrast, in Stacks, the call reverts with `ERR-INVALID-UPDATES` if there isn't at least one valid price update.

```
;; Ensure we have at least one entry
(asserts! (> (len successful-updates) u0) ERR_INVALID_UPDATES)
```

This significant deviation from the known behavior means that all integrating protocols that choose to automatically submit price feeds may be front-run by other submitters, causing their own update call to revert.

This issue can arise either naturally, as users update their prices, or maliciously, through a third party actively targeting a protocol.

Recommendation

Remove the check for valid updates on line L64.

Additionally, a change must be made in the oracle contract, as the fee will be 0 in such cases, resulting in the `stx-transfer?` call to revert. Charge the fee only if it is greater than 0.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

[M-03] Changing Governance Data Source May Cause Denial of Service in Operations

Description

The Pyth Stacks governance contract is designed to accept updates only from an initial, trusted emitter. However, this emitter can be changed through a call to `pyth-governance-v1::update-governance-data-source`.

An update initiated by the trusted emitter is encapsulated within a Verified Action Approval (VAA) message. Within the `pyth-governance-v1` contract, it is always verified that any message received from the trusted emitter is newer than the last message received.

```
;; Check Sequence
(asserts! (> sequence (var-get last-sequence-processed)) ERR_OUTDATED)
;; Update Sequence
(var-set last-sequence-processed sequence)
```

This check uses the sequence number, which is:

> sequence u64 - the auto-incrementing integer that represents the number of messages published by this emitter

However, the current mechanism for updating the trusted governance data source does not consider this sequence number. After changing the governance emitter, the `last-sequence-processed` variable is not reset to match the new emitter.

If the new emitter has published fewer messages than the previous trusted emitter, subsequent updates will be considered outdated and will revert with `ERR_OUTDATED` .

Consider the following scenario:

- The Pyth governance emitter reaches 30,000 messages.
- Governance decides to switch to a newly created emitter.
- The new emitter has 0 messages.
- Consequently, all further governance updates will revert wit `ERR_OUTDATED` .

The new emitter must then send a large number of messages to reach the sequence number of the previous emitter. Until that point, no updates can be made to the Pyth Stacks contract.

Recommendation

When updating the governance data source using the `update-governance-data-source` function, include the new emitter's sequence number in the VAA message and update `last-sequence-processed` accordingly.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

[M-04] Price Cannot Be Updated During Guardian Set Transition Period

Description

The Wormhole guardian set update procedure consists of two phases:

1. The current guardian set emits messages containing the new guardian set to all targeted chains.
2. The current guardian set replaces itself with the new guardian set.

Between the emission of update messages to all chains `(1)` and the actual change of the set `(2)` , up to 24 hours can elapse. During this period, price feed updates continue to be emitted using the old guardian set and must be accounted for.

To address this, Pyth oracle implementations are required to accept price update messages emitted from the old guardian set for an additional 24 hours after the update, even though the guardian set has changed.

This expiration pattern is evident in existing Pyth implementations. For example, in Ethereum, when a new guardian set is submitted, the old guardian set is set to expire. The old guardian set will expire in 24 hours, meaning that messages sent with it remain valid until that time.

The current Pyth Stacks implementation does not incorporate any expiration logic and only verifies that received messages belong to the currently active guardian set.

```
;; Ensure that the guardian-set-id is the active one
(asserts! (is-eq (get guardian-set-id (get vaa message))
  (var-get active-guardian-set-id))
```

As a result, during any future guardian set update, price updates may become unavailable for up to 24 hours.

Recommendation

When invoking the `ormhole-core-v2::update-guardians-set` function, also set an expiration time for the soon-to-be-old guardian set. In the `parse-and-verify-vaa` function, modify the set verification to allow the Verified Action Approval (VAA) message guardian set as valid if it originates from a known guardian set and is within the expiration time frame.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

8.4. Low Findings

[L-01] Inability to Deactivate Price Update Fee

Description

The current system for updating feed prices does not permit setting a fee to 0.

Although fees can technically be set to 0 through a call to `pyth-governance-v1::update-fee-value`, the functions `verify-and-update-price-feeds` and `decode-price-feeds` from the `pyth-oracle-v2` contract will revert when a fee of 0 is encountered. This is due to the `stx-transfer?` function attempting to transfer 0 tokens.

This restriction hinders the protocol's ability to offer promotional periods or to initially promote the use of the oracle without charging users any fees.

Recommendation

Adjust the `verify-and-update-price-feeds` and `decode-price-feeds` functions to ensure that STX is only transferred if the fee is greater than 0.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[L-02] Default Price Update Fee Differs From Documentation

Description

According to the Pyth documentation regarding fees:

> until governance is live, the fee will be 1 of the smallest denomination of the blockchain's native token (e.g., 1 wei on Ethereum)

This is not correctly implemented on the Stacks blockchain, as the default fee is set to 10 units of the smallest denomination, instead of one.

In `pyth-governance-v1` , the fee exponent and mantissa are set to a default of 1- 1:

```
(define-data-var fee-value
  { mantissa: uint, exponent: uint }
  { mantissa: u1, exponent: u1 })
```

This means that when the actual fee amount is calculated, it results in `10` units.

```
(fee-amount (* (len updated-prices) (* (get mantissa fee-info) (pow u10
  (get exponent fee-info)))))
```

While not a major issue, this implementation detail differs from how Pyth is intended to operate.

Recommendation

Change the default fee exponent to `u0` instead of `u1`.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[L-03] Governance Updated Principals Are Not Validated

Description

When updating the principals in the Pyth governance contract, there is no verification to ensure that the resulting principal is valid for the current network.

For instance, if a contract from the testnet is mistakenly set instead of one from the mainnet, the entire Pyth oracle would cease to function until another governance update corrects the error.

Recommendation

In the `pyth-governance-v1::parse-principal` function, validate the `new-principal` variable using the `is-standard` Clarity function.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[L-04] Parallel Governance Proposals Can Be Blocked

Description

The current `pyth-governance-v1` governance implementation is permissionless, allowing anyone to call the update configuration functions, provided the Verified Action Approval (VAA) message is valid. The VAA message can only be issued by the authorized Pyth governance module. Once created on Pythnet, it can be forwarded to the Stacks contract by anyone.

To prevent replay attacks or the reuse of old VAAs, the Stacks contract checks that the VAA sequence number is newer than any previously seen.

```
;; Check Sequence
(asserts! (> sequence (var-get last-sequence-processed)) ERR_OUTDATED)
;; Update Sequence
(var-set last-sequence-processed sequence)
```

However, this mechanism allows for a specific denial-of-service (DOS) scenario if there is more than one active proposal. By intentionally submitting a newer proposal before an older one is processed, the older proposal can be invalidated.

Example:

- Governance needs to update the Stacks decoder and fee.
- Two proposals are created: Proposal A to update the decoder and Proposal B to increase the fee. Proposal B will have a higher sequence number than A.
- Before governance bots can automatically call Stacks contracts with the updated values, a malicious user can submit Proposal B's VAA first, causing Proposal A to revert.

The overall impact is that when multiple proposals are in parallel, an attacker can delay the full system update by front-running all updates with the most recent proposal.

Recommendation

As this scenario is extremely rare and would only cause a slight delay in updates, the recommendation is to be aware of this possibility and ensure Pyth governance creates a new proposal only after the previous one has been updated.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

[L-05] Incorrect Validation of Guardian Set Index Update

Description

When updating the existing guardian set in the `wormhole-core-v2` contract, several validations are performed on the new set provided.

The guardian set index check, conducted in the `parse-and-verify-guardians-set` function, is incorrect.

Official implementations must always ensure that the next index set increases by exactly 1 step, as specified here. However, the `parse guardians-set` implementation only checks for an increase in value.

```
;; Ensure that next index > current index
(asserts! (> (get value cursor-new-index) (var-get active-guardian-set-id))
  ERR_GSU_CHECK_INDEX)
```

This deviates from the general Pyth procedure and may lead to unexpected situations when upgrading sets.

Recommendation

Modify the next index check to ensure it always increases by exactly one step from the previous set.

With this modification, attention must be given to the `wormhole-core-v2.active-guardian-set-id` variable, which is set to 0.

As it stands, the wormhole governance contract can only be initialized with the guardian set index starting at 1. However, if the set reaches higher values before the `wormhole-core-v2` contract is deployed, it will revert if attempting to update it directly to the required set.

In such cases, either change the default `active-guardian-set-id` value to `<existing-set-id> - 1` or discard any guardian set index checks when the contract is first initialized.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[L-06] Incorrect Validation of Guardian Set ChainId

Description

When updating the existing guardian set in the `wormhole-core-v2` contract, several validations are performed on the newly provided set. However, the validation of the guardian set chainId in the `parse-and-verify-guardians-set` function is incorrect.

Official implementations verify that the chainID is either the current Pyth designated chainID for the network or 0.

The current Stacks implementation incorrectly mandates that the chainId value must always be 0, instead of also accepting the current chainId as valid.

```
(asserts! (is-eq (get value cursor-chain) u0)
```

This deviation from established implementations may result in governance guardian update messages being incorrectly invalidated.

Recommendation

Modify the check on line L349 to also recognize the Stacks Pyth chain ID (`60038` for mainnet or `50039` for testnet) as valid.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[L-07] Missing Implicit Stale Price Checking API

Description

The current Stacks Pyth implementation only offers an API that does not validate price staleness when retrieving the price, but does so only when updating it.

This behavior is completely different from standard Pyth APIs, which include at least one function that implicitly checks for staleness. For example, EVM:getPrice reverts with:

> StalePrice: The on-chain price has not been updated within the last getValidTimePeriod() seconds.

Recommendation

Introduce a `get-price` equivalent function in the `pyth-oracle-v2` contract that checks for price freshness using the default price staleness threshold.

Do not modify the `read-price-feed` function. A pure, unchecked version of price retrieval must also exist so that protocols can determine their own threshold, if necessary.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

[L-08] Reconsider Default Fee Receiver and Stale Price Threshold

Description

The current deployment of Pyth governance utilizes the following default settings for the fee receiver and stale duration checks:

- Fee receiver principal: `'SP3CRXBDXQ2N5P7E25Q39MEX1HSMRDSEAP3CFK2Z3` (mainnet) or `'ST3CRXBDXQ2N5P7E25Q39MEX1HSMRDSEAP1JST19D` (testnet)
- Stale price threshold: 2 hours for mainnet and 5 years for testnet

If governance will not be accessible for some time post-deployment, these values become crucial as they will remain in effect until governance is available. Therefore, they must be selected with care.

Recommendation

Select an appropriate, team-controlled principal for receiving fees, which will then be forwarded to the Pyth team.

Reevaluate the 2-hour price validity on the mainnet, as it is excessively long for the current version of Nakamoto Stacks.

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[L-09] Missing Overlay Checks on V AA Payloads

Description

Throughout the codebase, there are several key areas where Verified Action Approval (VAA) messages are parsed. In none of these cases are there checks to ensure that overlay data is not attached to the end of a valid message.

Overlay refers to binary data that exists in a buffer after all parsing has been completed but is not referenced by any code logic. Generally, this results in increased resource consumption when parsing data buffers, but it may also lead to unexpected behavior at times.

Pyth EVM standard implementations ensure that no overlay exists when parsing messages. Examples include setting.guardians or changing fees.

However, the Stacks Pyth implementation does not perform such validation. Instances where it should be added include parsing the last ending cursor in a VAA:

- In `'wormhole-core-v2::parse-and-verify-guardians-set`, there should be no data after reading the guardian set keys.
- In `pyth-governance-v1`, for all function updates, such as in `parse-and-verify-fee-value`, no further data should exist after the fee exponent.
- In `pyth-governance-v1`, within `parse-and-verify-prices-updates`.

Recommendation

In all cases where the end of a VAA buffer is parsed, ensure that the entire payload is accounted for and no extra overlay data exists.

Example of adding checks in `wormhole-core-v2::parse-and-verify-guardians-set`:

```
(asserts! (is-eq (get pos (get next guardians-bytes))
  (len bytes)) ERR_GSU_CHECK_OVERLAY)
```

Another example of adding checks in `pyth-governance-v1::parse-and-verify-fee-value`:

```
(asserts! (is-eq (get pos (get next cursor-exponent))
  (len ptgm-body)) ERR_INVALID_ACTION_PAYLOAD)
```

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
   5.1. Impact 4
   5.2. Likelihood 5
   5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
- **[C-01]** Attacker Can Corrupt Guardian Set During Update 10

8.2. High Findings 12
- **[H-01]** Absence of Pyth Stacks Governance Module 12
- **[H-02]** Wormhole Contract Vulnerable to Hijacking at Deployment 14
- **[H-03]** Limited Price Updates Due to High Runtime Costs 15

8.3. Medium Findings 17
- **[M-01]** Potential Use of Stale Price When Updating Price 17
- **[M-02]** Price Update Logic May Cause Denial of Service 19
- **[M-03]** Changing Governance Data Source May Cause Denial of Service in Operations 20
- **[M-04]** Price Cannot Be Updated During Guardian Set Transition Period 21

8.4. Low Findings 22
- **[L-01]** Inability to Deactivate Price Update Fee 22
- **[L-02]** Default Price Update Fee Differs From Documentation 23
- **[L-03]** Governance Updated Principals Are Not Validated 24
- **[L-04]** Parallel Governance Proposals Can Be Blocked 25
- **[L-05]** Incorrect Validation of Guardian Set Index Update 26
- **[L-06]** Incorrect Validation of Guardian Set ChainId 27
- **[L-07]** Missing Implicit Stale Price Checking API 28
- **[L-08]** Reconsider Default Fee Receiver and Stale Price Threshold 29
- **[L-09]** Missing Overlay Checks on V AA Payloads 30
- **[L-10]** Incorrect Validation of Minor Version When Updating Price 32
- **[L-11]** Wormhole Guardian Set Can Be Updated With An Empty Set 33
- **[L-12]** Wormhole Guardian Set Can Contain Duplicate Entries 34
- **[L-13]** PTGM Price Data Sources Length Is Not Validated 36

8.5. QA Findings 37
- **[QA-01]** Remove Outdated pyth-p2wh-decoder-v1 Contract 37
- **[QA-02]** Leftover Bitcoin Timestamp Code Usage 38
- **[QA-03]** Project Call To Action References Outdated Repository 39
- **[QA-04]** Error Code Inconsistencies 40
- **[QA-05]** Redeploy Dependency Contracts for Optimization 42
- **[QA-06]** Eliminate Unused Constants 43
- **[QA-07]** Redundant Tuple with One Element as Map Key 44
- **[QA-08]** Incorrect Naming of Update Function Events 45
- **[QA-09]** Inconsistent Return Values in Governance Update Functions 46
- **[QA-10]** Inconsistent Reference to Pyth State Bearing Contract 47
- **[QA-11]** Misleading, Outdated, or Incomplete Comments 48
- **[QA-12]** Use Constants Where Appropriate 49
- **[QA-13]** Simplification Opportunities in Code Operations 50
- **[QA-14]** Typographical Errors 51
- **[QA-15]** Merkle Implementation Can Invalidate Correct Price Updates 52
- **[QA-16]** AUWV Price Feed Update Length Is Not Validated 53
- **[QA-17]** Price Update Can Be From The Future 54

Final example of adding an overlay check in `pyth-pnau-decoder-v1::and-verify-prices-updates`, which requires several changes:

```diff
@@ -42,6 +42,8 @@
 (define-constant ERR_UNAUTHORIZED_FLOW (err u2404))
 ;; Price update not signed by an authorized source
 (define-constant ERR_UNAUTHORIZED_PRICE_UPDATE (err u2401))
+;; VAA buffer has unused, extra leading bytes (overlay)
+(define-constant ERR_OVERLAY_PRESENT (err u2402))

 ;;;; Public functions
 (define-public (decode-and-verify-price-feeds (pnau-bytes (buff 8192))
   (wormhole-core-address <wormhole-core-trait>))
@@ -132,7 +134,7 @@
 (define-private (parse-and-verify-prices-updates (bytes (buff 8192))
   (merkle-root-hash (buff 20)))
   (let ((cursor-num-updates (try! (
     let
   )
        (cursor-updates-bytes
          (contract-call? ‘SP2J933XB2CP2JQ1A4FGN8JA968BBG3NK3EKZ7Q9F.hk cursor-v2 sli
-       (updates (get result
- (fold parse-price-info-and-proof cursor-updates-bytes {
+        (updates-data (fold parse-price-info-and-proof cursor-updates-bytes {
          result: (list),
          cursor: {
            index: u0,
@@ -140,12 +142,15 @@
          },
          bytes: cursor-updates-bytes,
          limit: (get value cursor-num-updates)
-        })))
+        }))
+        (updates (get result updates-data))
        (merkle-proof-checks-success (get result
          (fold check-merkle-proof updates {
          result: true,
          merkle-root-hash: merkle-root-hash
        }))))
     (asserts! merkle-proof-checks-success MERKLE_ROOT_MISMATCH)
+
+      ;; Overlay check; 1 is added because 1 byte is used to store “cursor-num-updates
+     (asserts! (is-eq (+ u1 (get next-update-index (get cursor updates-data)))
+ (len bytes)) ERR_OVERLAY_PRESENT)
    (ok updates)))
 (define-private (check-merkle-proof
```

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[L-10] Incorrect Validation of Minor Version When Updating Price

Description

When updating the price of an asset, the provided price data payload is verified to comply with the Pyth Network Accumulator Update (PNAU) standard.

One aspect of this validation involves checking the minimum allowed version, which is currently done incorrectly.

The Stacks Pyth implementation mistakenly enforces that the minor version must be equal to 0 (`PYTHNET_MINOR_VERSION`).

```
;; Check minor version
(asserts! (is-eq
  (get value cursor-version-min) PYTHNET_MINOR_VERSION) ERR_VERSION_MIN)
```

In contrast, official implementations recognize minor versions as forward compatible and ensure no downgrades occur.

This approach deviates from the general Pyth procedure and may lead to unexpected situations where price updates are incorrectly invalidated.

Recommendation

Modify the minimum version check in `pyth-pnau-decoder-v1::parse-pnau-header` to ensure the minimum version is not less than `PYTHNET_MINOR_VERSION` .

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set

Description

When updating the guardian set in the `wormhole-core-v2` contract via the `update-guardians-set` function, the current logic does not check if the newly validated set is empty.

If the guardian set is empty, no new messages can be validated, effectively rendering the entire functionality inoperative.

Although this scenario is unlikely, it could occur if an empty update message payload, correctly signed, is mistakenly provided by the current guardian set.

Recommendation

In the `update-guardians-set` function, ensure that the length of the guardian set to be saved is greater than zero.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

[L-12] Wormhole Guardian Set Can Contain Duplicate Entries

Description

When updating the guardian set in the `wormhole-core-v2` contract via the `update-guardians-set` function, the current logic does not check if the new set contains duplicate addresses.

In addition to being logically incorrect, duplicate addresses are not counted when validating future VAA (Verified Action Approval) messages and may even prevent reaching the minimum quorum because of it.

This situation can only occur if, by mistake, a correctly signed update message payload is provided by the current guardian set.

Recommendation

Check for duplicate Ethereum addresses in the V AA payload when the `update-guardians-set` is called.

An implementation example could involve adding a deduplication step in `parse-guardian` , followed by a check to ensure the length of `eth-addresses` matches the noted guardian count:

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
 5.1. Impact 4
 5.2. Likelihood 5
 5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
 [C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
 [H-01] Absence of Pyth Stacks Governance Module 12
 [H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
 [H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
 [M-01] Potential Use of Stale Price When Updating Price 17
 [M-02] Price Update Logic May Cause Denial of Service 19
 [M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
 [M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
 [L-01] Inability to Deactivate Price Update Fee 22
 [L-02] Default Price Update Fee Differs From Documentation 23
 [L-03] Governance Updated Principals Are Not Validated 24
 [L-04] Parallel Governance Proposals Can Be Blocked 25
 [L-05] Incorrect Validation of Guardian Set Index Update 26
 [L-06] Incorrect Validation of Guardian Set ChainId 27
 [L-07] Missing Implicit Stale Price Checking API 28
 [L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
 [L-09] Missing Overlay Checks on V AA Payloads 30
 [L-10] Incorrect Validation of Minor Version When Updating Price 32
 [L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
 [L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
 [L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
 [QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
 [QA-02] Leftover Bitcoin Timestamp Code Usage 38
 [QA-03] Project Call To Action References Outdated Repository 39
 [QA-04] Error Code Inconsistencies 40
 [QA-05] Redeploy Dependency Contracts for Optimization 42
 [QA-06] Eliminate Unused Constants 43
 [QA-07] Redundant Tuple with One Element as Map Key 44
 [QA-08] Incorrect Naming of Update Function Events 45
 [QA-09] Inconsistent Return Values in Governance Update Functions 46
 [QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
 [QA-11] Misleading, Outdated, or Incomplete Comments 48
 [QA-12] Use Constants Where Appropriate 49
 [QA-13] Simplification Opportunities in Code Operations 50
 [QA-14] Typographical Errors 51
 [QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
 [QA-16] AUWV Price Feed Update Length Is Not Validated 53
 [QA-17] Price Update Can Be From The Future 54

```diff
@@ -68,6 +68,9 @@
 (define-constant ERR_GSU_CHECK_INDEX (err u1304))
 ;; Guardian Set Update emission payload unauthorized
 (define-constant ERR_GSU_CHECK_EMITTER (err u1305))
+;; Duplicate guardian addresses found
+(define-constant ERR_DUPLICATED_GUARDIAN_ADDRESSES (err u1306))
+
 ;; Guardian set upgrade emitting address

   (define-constant GSU-EMITTING-ADDRESS 0x0000000000000000000000000000000000000000000
@@ -317,10 +320,13 @@
   (let (
     (cursor-address-bytes (unwrap-panic (
      cursor-address-bytes
    ), pos: cue-position }
   )
-  {
-    bytes: (get bytes acc),
-    result: (unwrap-panic (as-max-len? (append (get result acc)
- (get value cursor-address-bytes)) u19))
-  }))
+  (if (is-none (index-of? (get result acc) (get value cursor-address-bytes)))
+    {
+      bytes: (get bytes acc),
+      result: (unwrap-panic (as-max-len? (append (get result acc)
+ (get value cursor-address-bytes)) u19))
+    }
+    acc
+  )))
 ;; @desc Parse and verify payload's VAA
 (define-private (parse-and-verify-guardians-set (bytes (buff 8192)))
@@ -339,6 +345,8 @@
           ERR_GSU_PARSING_GUARDIANS_BYTES))
        (guardians-cues (get result (fold is-guardian-cue
         (get value guardians-bytes) { cursor: u0, result: (list) })))
       (eth-addresses (get result (fold parse-guardian guardians-cues { bytes:
        (get value guardians-bytes), result: (list) }))))
+   ;; Ensure there are no duplicate addresses
+   (asserts! (is-eq (len eth-addresses)
+ (get value cursor-guardians-count)) ERR_DUPLICATED_GUARDIAN_ADDRESSES)
    ;; Ensure that this message was emitted from an authorized module
    (asserts! (is-eq
      (get value cursor-module) 0x000000000000000000000000000000000000000000000000000
    ERR_GSU_CHECK_MODULE)
```

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
  5.1. Impact 4
  5.2. Likelihood 5
  5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
  [C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
  [H-01] Absence of Pyth Stacks Governance Module 12
  [H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
  [H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
  [M-01] Potential Use of Stale Price When Updating Price 17
  [M-02] Price Update Logic May Cause Denial of Service 19
  [M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
  [M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
  [L-01] Inability to Deactivate Price Update Fee 22
  [L-02] Default Price Update Fee Differs From Documentation 23
  [L-03] Governance Updated Principals Are Not Validated 24
  [L-04] Parallel Governance Proposals Can Be Blocked 25
  [L-05] Incorrect Validation of Guardian Set Index Update 26
  [L-06] Incorrect Validation of Guardian Set ChainId 27
  [L-07] Missing Implicit Stale Price Checking API 28
  [L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
  [L-09] Missing Overlay Checks on V AA Payloads 30
  [L-10] Incorrect Validation of Minor Version When Updating Price 32
  [L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
  [L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
  [L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
  [QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
  [QA-02] Leftover Bitcoin Timestamp Code Usage 38
  [QA-03] Project Call To Action References Outdated Repository 39
  [QA-04] Error Code Inconsistencies 40
  [QA-05] Redeploy Dependency Contracts for Optimization 42
  [QA-06] Eliminate Unused Constants 43
  [QA-07] Redundant Tuple with One Element as Map Key 44
  [QA-08] Incorrect Naming of Update Function Events 45
  [QA-09] Inconsistent Return Values in Governance Update Functions 46
  [QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
  [QA-11] Misleading, Outdated, or Incomplete Comments 48
  [QA-12] Use Constants Where Appropriate 49
  [QA-13] Simplification Opportunities in Code Operations 50
  [QA-14] Typographical Errors 51
  [QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
  [QA-16] AUWV Price Feed Update Length Is Not Validated 53
  [QA-17] Price Update Can Be From The Future 54

[L-13] PTGM Price Data Sources Length Is Not Validated

Description

The function `pyth-governance-v1::update-prices-data-sources` is responsible for updating price data sources. During this process, a malformed payload may occur where the PTGM (Pyth Governance Message) specifies a certain number of sources, but fewer are actually provided.

This scenario is not currently checked, and in the unlikely event that such a payload is sent, fewer sources than intended may be added. This could potentially lead to a lack of price validation during updates.

Recommendation

In the `parse-and-verify-prices-data-sources` function of the `pyth-governance-v1` contract, implement a check to ensure that the number of data sources indicated by the payload matches the number of sources parsed.

Example implementation:

```
+++ b/contracts/pyth-governance-v1.clar
@@ -48,6 +48,8 @@
 (define-constant ERR_UNAUTHORIZED_UPDATE (err u4006))
 ;; Error parsing PTGM
 (define-constant ERR_INVALID_PTGM (err u4007))
+;; Error invalid price data source
+(define-constant ERR_INVALID_PRICE_DATA_SOURCES (err u4008))

 (define-data-var governance-data-source
   { emitter-chain: uint, emitter-address: (buff 32) }
@@ -425,6 +427,7 @@
           bytes: cursor-data-sources-bytes,
           limit: (get value cursor-num-data-sources)
         }))))
+    (asserts! (is-eq (get value cursor-num-data-sources)
+ (len data-sources)) ERR_INVALID_PRICE_DATA_SOURCES)
    (ok data-sources)))

(define-private (parse-data-source
```

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

8.5. QA Findings

[QA-01] Remove Outdated `pyth-p2wh-decoder-v1` Contract

Description

The `pyth-p2wh-decoder-v1` contract present in the codebase is neither utilized nor compatible with other contracts. Developers included it for posterity as it served as the initial building blocks and a developer playground.

> chore: re-introduce p2wh decoder (for posterity)

This contract is incompatible with existing contracts in several ways:

- It does not adhere to the `pyth-traits-v1.decoder-trait`, lacking the `decode-and-verify-price-feeds` function.
- It maintains state.
- It lacks necessary checks and validations.
- It expects to interact with a nonexistent `wormhole-core-dev-preview-1` contract.
- It contains TODOs and remnants of development.

Recommendation

Remove the contract entirely from the codebase. For posterity, the contract is already preserved in the hirosystems repository., so duplicating it in the new codebase is unnecessary.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[QA-02] Leftover Bitcoin Timestamp Code Usage

Description

Following the recent Pyth changes related to the Nakamoto upgrade, the Stacks block timestamp is now used instead of the burnchain (Bitcoin) block time to determine staleness validation.

This change was implemented in the `pyth-store-v2` by switching to `get-stacks-block-info?` and `stacks-block-height`.

However, some additional changes were not made in the code to fully reflect this update:

1. The `latest-bitcoin-timestamp` variable in the `pyth-store-v2` `batch-entry` function should be renamed to `latest-stacks-timestamp` as the current name is misleading.

```
(latest-bitcoin-timestamp (unwrap! (get-stacks-block-info? time
  (- stacks-block-height u1)) ERR_STALE_PRICE)))
```

2. The current version of the `pyth-store-v2` contract is v2, but the internal version comment still incorrectly states: `;; Version: v1`

Recommendation

Rename the `latest-bitcoin-timestamp` variable to `latest-stacks-timestamp` in the `pyth-store-v2::write-batch-entry` function. Update the version comment to v2 in `pyth-store-v2`.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[QA-03] Project Call To Action References Outdated Repository

Description

All project contracts include the `Check for latest version` and `Report an issue` call-to-action (CTA) links that currently direct users to the outdated `hirosystems` repository, rather than the new Trust Machines repository.

```
;; Check for latest version:
// <https://github.com/hirosystems/stacks-pyth-bridge#latest-version>
;; Report an issue: <https://github.com/hirosystems/stacks-pyth-bridge/
issues>
```

In the unlikely event of any system issues, users would mistakenly submit their reports to the incorrect repository.

Recommendation

Update all CTAs to direct users to the current repository.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

[QA-04] Error Code Inconsistencies

Description

In the codebase, each contract should have a unique error code range to easily identify the originating contract of an error. The current error code ranges implemented are:

```
wormhole-core-v2          1000 - 1999 -> start 1001
pyth-governance-v1        4000 - 4999 -> start 4001
pyth-oracle-v2             400 -  499 -> start  402
pyth-pnau-decoder-v1      2000 - 2999 -> start 2001
pyth-store-v2             5000 - 5999 -> start 5000
```

Although the ranges are unique, several minor issues exist:

- There is no range for 3000-3999.
- In `wormhole-core-v2` :
  - ◊ The error code `u1104` is missing between `ERR_VAA_CHECKS_REDUNDANT_SIGNATURE` and `ERR_VAA_CHECKS_GUARDIAN_SET_CONSISTENCY`.
  - ◊ `ERR_VAA_CHECKS_REDUNDANT_SIGNATURE` is unused.
- In `pyth-governance-v1` :
  - ◊ The `ERR_UNAUTHORIZED_ACCESS` error (`u4004`) is placed before error `u4001` instead of between `u4003` and `u4005`.
  - ◊ `ERR_UNEXPECTED_ACTION_PAYLOAD` is redundant, as `ERR_INVALID_ACTION_PAYLOAD` already exists with the same logic. `ERR_UNEXPECTED_ACTION_PAYLOAD` is also used only once.
- In `pyth-oracle-v2` :
  - ◊ Error codes are in the hundreds range, while others are in the thousands range.
  - ◊ `ERR_BALANCE_INSUFFICIENT` starts at `u402` instead `u401`
- In `pyth-pnau-decoder-v1` :
  - ◊ `ERR_NOT_FOUND` is unused and has an out-of-range value (`u0`).
  - ◊ `ERR_UNAUTHORIZED_FLOW` is unused with an incorrect comment.
  - ◊ `MERKLE_ROOT_MISMATCH` is an error but lacks the `ERR_` prefix.
- In `pyth-store-v2` :
  - ◊ The first error, `ERR_NEWER_PRICE_AVAILABLE` starts at base 0 (`u5000`) unlike other contracts.
  - ◊ The `read` and `get-price` functions both have a hardcoded `(err u404)` error with an incorrect range.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

- The `ERR_PANIC` constant is present in several contracts but is unused.

These issues complicate debugging for third-party integrators.

Recommendation

Implement the following changes:

1. Remove `ERR_PANIC` from all contracts.
2. In `wormhole-core-v2`, remove `ERR_VAA_CHECKS_REDUNDANT_SIGNATURE` and rebase `ERR_VAA_CHECKS_GUARDIAN_SET_CONSISTENCY` to `(err u1103)`.
3. In `pyth-governance-v1`, move the `ERR_UNAUTHORIZED_ACCESS` declaration between `ERR_INVALID_ACTION_PAYLOAD` and `ERR_OUTDATED`. Also, remove `ERR_UNEXPECTED_ACTION_PAYLOAD` and use `ERR_INVALID_ACTION_PAYLOAD` in its place. Ensure all other errors are correctly rebased afterward.
4. Change the `pyth-oracle-v2` error code range to 3000-3999 and start `ERR_BALANCE_INSUFFICIENT` at `u3001`.
5. In `pyth-pnau-decoder-v1`, remove the `ERR_NOT_FOUND` and `ERR_UNAUTHORIZED_FLOW` error codes. Rename `MERKLE_ROOT_MISMATCH` to `ERR_MERKLE_ROOT_MISMATCH`.
6. In `pyth-store-v2`, rebase `ERR_NEWER_PRICE_AVAILABLE` to `u5005`. Create an `ERR_PRICE_FEED_NOT_FOUND` error, set it to `(err u5006)` and use it in the `read` and `get-price`

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[QA-05] Redeploy Dependency Contracts for Optimization

Description

In Clarity, there are two methods for calling functions from another smart contract:

- Using only the contract name, without the deployment standard principal (e.g., `.my-contract`)
- Using the full contract principal (e.g., `ST1PQHQKV0RJXZFY1DGX8MNSNYVE3VGZJSRTPGZGM.my-contract`)

When the second method is used, the total read costs of the transaction increase due to the additional data that needs to be copied between function contract calls.

Throughout the codebase, dependencies are called using their full contract principal. Examples include: `'SP2J933XB2CP2JQ1A4FGN8JA968BBG3NK3EKZ7Q9F.hk-cursor-v2` or `'SP2J933XB2CP2JQ1A4FGN8JA968BBG3NK3EKZ7Q9F.hk-merkle-tree-keccak160-v1'`.

This practice increases the overall cost of executing calls through the oracle contract.

Recommendation

Redeploy all the used dependencies locally under the same deployer address to enable the use of the short contract principal calling convention.

CONTENTS
1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[QA-06] Eliminate Unused Constants

Description

The codebase contains several instances of unused constants:

- In the `wormhole-core-v2` contract, the constant `hk-cursor-v2` is declared but never utilized.
- In `pyth-pnau-decoder-v1`, the constants `STX_USD` and `BTC_USD` are also not used.

Recommendation

Remove these constants to enhance code readability, minimize clutter, and slightly decrease runtime read counts and costs.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[QA-07] Redundant Tuple with One Element as Map Key

Description

In the `wormhole-core-v2` contract, there is a map of guardian sets defined as follows:

```
;; Map tracking guardian sets
(define-map guardian-sets
  { set-id: uint }
  (list 19 { compressed-public-key: (buff 33), uncompressed-public-key:
    (buff 64) }))
```

This map unnecessarily uses a tuple containing only a single element, a `set-id` uint, instead of using the uint directly. This approach increases the overall operational cost and reduces code readability.

Recommendation

Modify the map to use a `uint` as the key instead of a tuple.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[QA-08] Incorrect Naming of Update Function Events

Description

In the `pyth-governance-v1` contract, each update function emits an event where the `type` is named after the function, omitting the `update-*` prefix.

Examples:

- `update-fee-value` → `type: "fee-value"`
- `update-stale-price-threshold` → `type: "stale-price-threshold"`
- `update-wormhole-core-contract` → `type: "wormhole-core-contract"`

However, there are two exceptions to this pattern:

- `update-fee-recipient-address` → `type: "fee-recipient"`
- `update-pyth-store-contract` → `type: "pyth-storage-contract"`

This inconsistency is also found in the `wormhole-core-v2` contract:

- `update-guardians-set` → `type: "guardian-set"`

Recommendation

To enhance code consistency, modify the `type` message in the `print` command for `update-fee-recipient-address` to `"fee-recipient-address"` and for `update-pyth-store-contract` to `"pyth-store-contract"`.

Similarly, for `update-guardians-set`, change the print message to `"guardians-set"`.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[QA-09] Inconsistent Return Values in Governance Update Functions

Description

In the `pyth-governance-v1` contract, each update function typically returns the specific data that was updated. However, this consistency is broken for updates targeting execution plan contracts (whitelisted contracts). In these instances, the update functions return the entire update execution plan, which includes four contracts, rather than just the single updated contract.

Recommendation

If this behavior is intentional, it should be acknowledged. Otherwise, modify the return values of the `update-wormhole-core-contract`, `update-pyth-oracle-contract`, `update-pyth-decoder-contract`, and `update-pyth-store-contract` functions to `(ok updated-data)`.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
    5.1. Impact 4
    5.2. Likelihood 5
    5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[QA-10] Inconsistent Reference to Pyth State Bearing Contract

Description

Within the codebase, the Pyth contract responsible for holding the prices of each token is referred to as the `storage` contract, although its actual name is `pyth-store-v2`.

This inconsistency in naming conventions reduces the overall readability of the code.

Recommendation

Rename the `pyth-store-v2` contract to `pyth-storage-v2`.

CONTENTS
1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

[QA-11] Misleading, Outdated, or Incomplete Comments

Description

The codebase contains comments that are either misleading or outdated.

Instances:

- `pyth-governance-v1`
    - ◊ In `update-stale-price-threshold`, the comment `;; Update fee-value` is a leftover from a copy-paste. It should reference updating the stale price threshold.
    - ◊ In `update-stale-price-threshold`, the comment `;; Update fee-recipient address` should be `;; Update fee-recipient-address` to align with existing commenting patterns.
    - ◊ In `update-governance-data-source`, the comment `;; Update prices-data-sources` is a copy-paste remnant. It should reference updating governance data sources instead.
    - ◊ In `parse-data-source`, the comment `;; Perform assertions` is outdated, as no assertions are performed afterward. It should be removed.

- `wormhole-core-v2`
    - ◊ The function description comment for `batch-recover-public-keys` is duplicated from `check-and-consolidate-public-keys`. The comment for `batch-recover-public-keys` should be rewritten.
    - ◊ In `parse-and-verify-guardians-set`, the comment `;; Ensure that this message is matching the adequate action` is duplicated. For the second check, it should indicate that the chain is adequate, not the action.

- `pyth-pnau-decoder-v1`
    - ◊ In `parse-proof`, the comment `;; Perform assertions` is outdated, as no assertions are performed afterward. It should be removed.

Recommendation

Address the mentioned instances as recommended above.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

[QA-12] Use Constants Where Appropriate

Description

To enhance code readability, it is recommended to use meaningful constants where applicable. Below are instances in the current codebase where constants can be utilized, along with suggested names:

- In `wormhole-core-v2`:
  - ◊ At L343, replace `0x00000000000000000000000000000000000000000000000000000000436f7265` with `CORE_STRING_MODULE`.
  - ◊ At L346, replace `u2` with `ACTION_GUARDIAN_SET_UPDATE`.
  - ◊ At L349, replace `u0` with `CORE_CHAIN_ID`.
  - ◊ Replace all instances of `u20` with `GUARDIAN_ETH_ADDRESS_SIZE`.

- In `pyth-governance-v1`
  - ◊ At L452, replace `u34` with `SIZEOF_EMITTER_DATA`.

- In `pyth-pnau-decoder-v1`
  - ◊ At L89 and L120, replace `u0` with `UPDATE_TYPE_WORMHOLE_MERKLE`.
  - ◊ At L230, replace `u0` with `MESSAGE_TYPE_PRICE_FEED`.
  - ◊ At L218, L240, and L289, replace `u20` with `MERKLE_PROOF_HASH_SIZE`.

Recommendation

Implement the suggested changes.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

[QA-13] Simplification Opportunities in Code Operations

Description

There are several instances within the codebase where minor simplifications can be made to improve code readability and, in some cases, reduce code size.

1. In `wormhole-core-v2` , the `parse-and-verify-guardians-set` function returns a tuple containing both `value` and `next` elements. Since the `next` element is never used, the content of `value` can be returned directly. This change alters the function from returning a cursor to returning direct data.
2. In `pyth-pnau-decoder-v1::decode-and-verify-price-feeds` the `prices-updates` variable is declared only to be immediately returned. Avoid this redundancy by directly returning the result of the `decode-pnau-price-update` call.
3. In `pyth-store-v2::write-batch-entry` , the `(get publish-time entry)` value is retrieved three times. Store it as a variable within the existing `let` to avoid repetition.

Recommendation

Implement the suggested changes. .

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
   5.1. Impact 4
   5.2. Likelihood 5
   5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

[QA-14] Typographical Errors

Description

There are several typographical errors throughout the codebase:

- `malformatted` should be corrected to `malformed`
- `informations` should be corrected to `information`
- `byts` should be corrected to `bytes`
- `lastest` should be corrected to `latest`
- `cursor-udpate-type` should be corrected to `cursor-update-type`
- `Verficiation` should be corrected to `Verification`
- `expansive` should be corrected to `expensive`
- `Ensure action's expectation` should be corrected to `Ensure action's expected`

Recommendation

Correct all the identified typographical errors.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
   5.1. Impact 4
   5.2. Likelihood 5
   5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

[QA-15] Merkle Implementation Can Invalidate Correct Price Updates

Description

When a price update is performed, Merkle validation is conducted on the resulting data using the `hk-merkle-tree-keccak160-v1` contract.

In this implementation, when comparing nodes that are each 20 bytes in length, the value is trimmed to a 16-byte unsigned integer.

```
(define-read-only (buff-20-to-uint (bytes (buff 20)))
    (buff-to-uint-be (unwrap-panic (as-max-len? (unwrap-panic
      (slice? bytes u0 u15)) u16))))
```

This trimming is necessary because Stacks does not support integers larger than 128 bits.

Due to this trimming, theoretically, a valid Merkle root-proofs pair could be mistakenly identified as invalid by the Stacks implementation. This occurs because the node traversal may shift in the wrong direction, resulting in a different ending hash. Consequently, a valid price update payload could be discarded.

Recommendation

Given the extreme unlikelihood of this scenario occurring, and considering that an alternative price update will be readily available, we recommend being aware of this situation and acknowledging it.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
  5.1. Impact 4
  5.2. Likelihood 5
  5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
  [C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
  [H-01] Absence of Pyth Stacks Governance Module 12
  [H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
  [H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
  [M-01] Potential Use of Stale Price When Updating Price 17
  [M-02] Price Update Logic May Cause Denial of Service 19
  [M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
  [M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
  [L-01] Inability to Deactivate Price Update Fee 22
  [L-02] Default Price Update Fee Differs From Documentation 23
  [L-03] Governance Updated Principals Are Not Validated 24
  [L-04] Parallel Governance Proposals Can Be Blocked 25
  [L-05] Incorrect Validation of Guardian Set Index Update 26
  [L-06] Incorrect Validation of Guardian Set ChainId 27
  [L-07] Missing Implicit Stale Price Checking API 28
  [L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
  [L-09] Missing Overlay Checks on V AA Payloads 30
  [L-10] Incorrect Validation of Minor Version When Updating Price 32
  [L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
  [L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
  [L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
  [QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
  [QA-02] Leftover Bitcoin Timestamp Code Usage 38
  [QA-03] Project Call To Action References Outdated Repository 39
  [QA-04] Error Code Inconsistencies 40
  [QA-05] Redeploy Dependency Contracts for Optimization 42
  [QA-06] Eliminate Unused Constants 43
  [QA-07] Redundant Tuple with One Element as Map Key 44
  [QA-08] Incorrect Naming of Update Function Events 45
  [QA-09] Inconsistent Return Values in Governance Update Functions 46
  [QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
  [QA-11] Misleading, Outdated, or Incomplete Comments 48
  [QA-12] Use Constants Where Appropriate 49
  [QA-13] Simplification Opportunities in Code Operations 50
  [QA-14] Typographical Errors 51
  [QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
  [QA-16] AUWV Price Feed Update Length Is Not Validated 53
  [QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

[QA-16] AUWV Price Feed Update Length Is Not Validated

Description

When updating a price feed within the `pyth-pnau-decoder-v1` several checks are performed on the AUWV (Accumulator Update Wormhole Verification) message. However, one check that is missing is to verify that the number of feeds encoded within the payload matches the number of updates actually encoded. I

In theory, a corrupt message could be constructed with a lower actual feed count embedded. Such a message might have been initially intended to contain several updates, but by omitting them, it could cause an upstream revert.

Recommendation

Given the very low likelihood of this scenario occurring, and considering the already runtime-intensive implementation, this check can be acknowledged but not necessarily implemented.

If a check is desired, consider the following example:

```
@@ -36,6 +36,8 @@
 (define-constant ERR_INVALID_AUWV (err u2007))
 ;; Merkle root mismatch
 (define-constant MERKLE_ROOT_MISMATCH (err u2008))
+;; Incorrect AUWV payload
+(define-constant ERR_INCORRECT_AUWV_PAYLOAD (err u2009))
 ;; Price not found
 (define-constant ERR_NOT_FOUND (err u0))
 ;; Price not found
@@ -146,6 +148,7 @@
           merkle-root-hash: merkle-root-hash
         }))))
     (asserts! merkle-proof-checks-success MERKLE_ROOT_MISMATCH)
+    (asserts! (is-eq (get value cursor-num-updates)
+ (len updates)) ERR_INCORRECT_AUWV_PAYLOAD)
    (ok updates)))
(define-private (check-merkle-proof
```

Note: If the check is added, two tests will fail: `should fail if the price price is below stale threshold` and `should fail if PNAU include mismatches`. These tests fail because a different error is raised than expected, specifically the one addressed in this issue. Both tests exhibit the issue, as they encode only one price update but note that they will encode 3 updates.

CONTENTS

1. About Clarity Alliance 2
2. Disclaimer 3
3. Introduction 4
4. About Pyth Oracle 4
5. Risk Classification 4
5.1. Impact 4
5.2. Likelihood 5
5.3. Action required for severity levels 5
6. Security Assessment Summary 6
7. Executive Summary 7
8. Summary of Findings 8
8.1. Critical Findings 10
[C-01] Attacker Can Corrupt Guardian Set During Update 10
8.2. High Findings 12
[H-01] Absence of Pyth Stacks Governance Module 12
[H-02] Wormhole Contract Vulnerable to Hijacking at Deployment 14
[H-03] Limited Price Updates Due to High Runtime Costs 15
8.3. Medium Findings 17
[M-01] Potential Use of Stale Price When Updating Price 17
[M-02] Price Update Logic May Cause Denial of Service 19
[M-03] Changing Governance Data Source May Cause Denial of Service in Operations 20
[M-04] Price Cannot Be Updated During Guardian Set Transition Period 21
8.4. Low Findings 22
[L-01] Inability to Deactivate Price Update Fee 22
[L-02] Default Price Update Fee Differs From Documentation 23
[L-03] Governance Updated Principals Are Not Validated 24
[L-04] Parallel Governance Proposals Can Be Blocked 25
[L-05] Incorrect Validation of Guardian Set Index Update 26
[L-06] Incorrect Validation of Guardian Set ChainId 27
[L-07] Missing Implicit Stale Price Checking API 28
[L-08] Reconsider Default Fee Receiver and Stale Price Threshold 29
[L-09] Missing Overlay Checks on V AA Payloads 30
[L-10] Incorrect Validation of Minor Version When Updating Price 32
[L-11] Wormhole Guardian Set Can Be Updated With An Empty Set 33
[L-12] Wormhole Guardian Set Can Contain Duplicate Entries 34
[L-13] PTGM Price Data Sources Length Is Not Validated 36
8.5. QA Findings 37
[QA-01] Remove Outdated pyth-p2wh-decoder-v1 Contract 37
[QA-02] Leftover Bitcoin Timestamp Code Usage 38
[QA-03] Project Call To Action References Outdated Repository 39
[QA-04] Error Code Inconsistencies 40
[QA-05] Redeploy Dependency Contracts for Optimization 42
[QA-06] Eliminate Unused Constants 43
[QA-07] Redundant Tuple with One Element as Map Key 44
[QA-08] Incorrect Naming of Update Function Events 45
[QA-09] Inconsistent Return Values in Governance Update Functions 46
[QA-10] Inconsistent Reference to Pyth State Bearing Contract 47
[QA-11] Misleading, Outdated, or Incomplete Comments 48
[QA-12] Use Constants Where Appropriate 49
[QA-13] Simplification Opportunities in Code Operations 50
[QA-14] Typographical Errors 51
[QA-15] Merkle Implementation Can Invalidate Correct Price Updates 52
[QA-16] AUWV Price Feed Update Length Is Not Validated 53
[QA-17] Price Update Can Be From The Future 54

ClarityAlliance
Security Review

Pyth Oracle Client

[QA-17] Price Update Can Be From The Future

Description

When updating the price of an asset, several checks are performed concerning the feed's publish time. However, one scenario that is not addressed is when a price marked "from the future" is provided. If such a price feed, with a publish time set in the future, is used, any new price updates until that time is reached will be discarded.

This check is not consistently implemented across different Pyth crosschain versions. For instance, the Ethereum Pyth contracts do not implement this check, whereas the Fuel blockchain Pyth implementation does include this check.

For the Stacks blockchain, this check cannot be easily performed because there is no mechanism to retrieve the timestamp of the currently executing block, only the timestamp of the last executed block. In theory, Stacks blocks are minted at a frequency of 5 seconds. However, real-time data shows variations of up to tens of seconds between blocks.

There have also been instances where the Stacks blockchain stops producing blocks and resumes after a significant delay. A real-life example is Stacks block #242879, which was mined at `14:22:53 2024.11.21`, followed by the next Stacks block #242880 minted at `14:47:42 2024.11.21` —25 minutes later.

Implementing a mechanism to validate that the price publish time is not from the future will likely result in normal price update invalidations, as only the previous block's timestamp can be used as a reference point, not the current one.

Recommendation

Acknowledge the issue as it is.