coinspect

Zcash

Coinspect Source Code Review

Prepared for Zcash • December 2018

v1207

1. Table of Contents

2. Executive Summary

On March 2018, Zcash engaged Coinspect to perform a security audit of the Overwinter Zcash network upgrade source code. The objective of the audit requested by Zcash was to evaluate the impact on consensus and incentives of the Overwinter code changes.

During the assessment, Coinspect identified **2** high-risk issues, **0** medium-risk issues, and **0** low-risk issues. The high-risk issues identified during the assessment are not remotely exploitable by themselves to steal funds or compromise the privacy Zcash users. However, they affect the performance and availability of the p2p network.

On November 2018, Coinspect was asked to review the modifications introduced in the code in order to fix the vulnerabilities reported, and concluded the fixes implemented were correct and as a result, both **findings are now considered resolved**.

3. Introduction

Zcash is an implementation of the Zerocash protocol based on the Bitcoin Core C++ code. It intends to offer a far higher standard of privacy and anonymity through a sophisticated zero-knowledge proving scheme which preserves confidentiality of transaction metadata.

A whitebox security audit was conducted on the Zcash Overwinter network upgrade source code in order to detect security, privacy, and availability related problems.

Overwinter is the first network upgrade for Zcash, it includes: improvements for network upgrades, performance improvements for transparent transactions, and transaction expiry.

The present report was completed on December 7, 2018 by Coinspect.

3.1. Scope

The Zcash auditing strategy tasked experts with different specializations to focus on different aspects of the system.
The objective of the audit requested Coinspect to review **Overwinter** network upgrade changes to the Zcash code, focusing on the **incentives and consensus**. The review was limited to the following code changes in Zcash v1.0.15:

- **Transaction format version 3**
  - Spec: ZIP-202 Code: #2925
- **Network upgrade activation mechanism**
  - Spec: ZIP-200 Code: #2898
- **Transaction Signature Verification**
  - Spec: ZIP-143 Code: #2903
- **Transaction expiry**
  - Spec: ZIP-203 Code: #2874

4. Summary Of Findings

ID[1]	Description	Risk	Fixed
ZEC-012	Transaction Expiry Enables Node Isolation Attack	High	✔
ZEC-013	Transaction Expiry Enables Transaction Flooding at No Cost	High	✔

[1] Findings ZEC-001 to ZEC-011 were part of Coinspect's 2016 report.

5. Findings

ZEC-012 Transaction Expiry Enables Node Isolation Attack

Total Risk	Impact	Location
High	High	src/main.cpp
Fixed ✔	**Likelihood** High	**Category** Denial of Service

Description

Zcash Overwinter specification document ZIP-203 defines a new consensus rule to set an expiration block height after which a transaction cannot be mined. If the transaction is not mined within that time, it will be removed from *mempools* and considered invalid. The transaction format version 3 adds a new field `nExpiryHeight`; if its value is N, then the transaction must be included in block N or earlier.

The expired transaction handling logic enables a new node isolation attack composed of the following steps:

1. Attacker `N0` connects to victim `N1`. Victim is connected to `N2` and `N3`.
2. Attacker prepares more than 10 transactions (`expTXs`) with `txNew.nExpiryHeight` set to `H`.
3. Attacker waits for full block or header of block `B` of height `H`.
4. Attacker sends expTXs to `N1` and then sends block `B`.
5. Victim `N1` processes all the TXs because `N1` height is `H-1`.
6. Victim adds `expTXs` to the `vInventoryToSend` queue of each peer.
7. Victim sends `INVs` to some peer nodes (random process).
8. Victim processes block `B`.
9. Victim's peers (`N2`, `N3`) receive the `INVs` and then the block `B`.
10. Victim's peers ask victim for the `expTXs` transactions.
11. Victim sends `expTXs` transactions because they are in `mapRelay` (they will be deleted from memory pool when block of height `H+1` is connected)
12. **(!) Victim's peers (`N2`,`N3`) score `N1` negatively because victim sent expired transactions.**
13. After 10 transactions victim's peers ban `N1`.
14. Victim tries to connect to new peers.
15. Attacker repeats until there are no more honest peers for victim to connect to and victim is connected only to attacker's nodes.

The probability of success of the attack can be made much higher by pre-sending one hundred orphan transactions that will expire on the same deadline `N`, all depending on a single output O. This output O is part of a missing tiny (i.e. low cost) valid transaction, which doesn't need to expire. The attacker pre-sends the orphan transactions, and sends the

transaction containing O just before sending block `N+1`. The victim's node would trigger up to one hundred `INVentory` notifications to its peers on the reception the triggering transaction. Even if the transaction broadcasting process is randomized by the Trickle method (`fTrickleWait`), the probability that victim's peers receiving expired transactions is risen while performing a single attack.

Recommendations

There are several possibilities to prevent this attack:

1. Protect by design. INV messages are extended to contain expiry information. Before nodes request an object advertised by an INV, they can check whether the object has expired or when the object will expire. When nodes receive a transaction requested by an INV, they can discard the transaction if it has expired between request and reception. If nodes receive a transaction and it does not match the INV's expiry value, nodes punish the sender of the INV. The drawback of this defense is that it requires modifying the network protocol.

2. Do not ban in case of a recent expired transactions. When a expired transaction arrives to a node, if it has recently expired (e.g. in the last block), the originator is not banned. This protection measure may open a theoretical transaction spamming attack, just before a block is propagated, that may be used by a miner to delay the propagation of competing miner's block.

Since Zcash will undergo a hard-fork, we recommend modifying the INV command.

Attack Details

Coinspect reproduced the attack in a test environment, the Zcash debug.log snippets below illustrate the consequences of the attack for the victim node.

Debug.log from victim

```
2018-03-23 02:03:35 UpdateTip: new
best=01edb3c477e7a77719a1fa39d78e7d61686c45bdd7eb4200d7dd973c92cb205a  height=204  log2_work=11.766943
tx=205  date=2018-03-23 02:03:35 progress=1.000000  cache=0.0MiB(30tx)
2018-03-23 02:03:35   - Connect postprocess: 0.30ms [0.00s]
2018-03-23 02:03:35 - Connect block: 0.64ms [0.00s]
2018-03-23 02:03:35   nActualTimespan = 10200  before dampening
2018-03-23 02:03:35   nActualTimespan = 4462  before bounds
2018-03-23 02:03:35 GetNextWorkRequired RETARGET
2018-03-23 02:03:35 params.AveragingWindowTimespan() = 2550    nActualTimespan = 2550
2018-03-23 02:03:35 Current average: 200f0f04
0f0f040000000000000000000000000000000000000000000000000000000000
2018-03-23 02:03:35 After:  200f0f03  0f0f03ffffffffffffffffffffffffffffffffffffffffffffffffffff9e4
2018-03-23 02:03:35 sending: inv (1117 bytes) peer=1
2018-03-23 02:03:35 sending: inv (253 bytes) peer=2
2018-03-23 02:03:35 received: getheaders (645 bytes) peer=1
2018-03-23 02:03:35 getheaders 204 to 01edb3c477e7a77719a1fa39d78e7d61686c45bdd7eb4200d7dd973c92cb205a
from peer=1
2018-03-23 02:03:35 sending: headers (179 bytes) peer=1
2018-03-23 02:03:35 sending: inv (865 bytes) peer=2
```

```
2018-03-23 02:03:35 received: getdata (37 bytes) peer=1
2018-03-23 02:03:35 received getdata (1 invsz) peer=1
2018-03-23 02:03:35 received getdata for: block
01edb3c477e7a77719a1fa39d78e7d61686c45bdd7eb4200d7dd973c92cb205a peer=1
2018-03-23 02:03:35 sending: block (277 bytes) peer=1
2018-03-23 02:03:35 received: getdata (1081 bytes) peer=1
2018-03-23 02:03:35 received getdata (30 invsz) peer=1
2018-03-23 02:03:35 received getdata for: tx
185fc0698ff549a2587ab9868f79324d44990a31413bf521b713d95f3b8e6aa4 peer=1
2018-03-23 02:03:35 sending: tx (234 bytes) peer=1
2018-03-23 02:03:35 sending: tx (234 bytes) peer=1
2018-03-23 02:03:35 sending: tx (234 bytes) peer=1
2018-03-23 02:03:35 sending: tx (234 bytes) peer=1
2018-03-23 02:03:35 sending: tx (234 bytes) peer=1
[ last line repeated for 25 additional transactions ]
2018-03-23 02:03:36 received: reject (58 bytes) peer=1
2018-03-23 02:03:36 Reject tx code 16: txoverwinterexpired: hash
185fc0698ff549a2587ab9868f79324d44990a31413bf521b713d95f3b8e6aa4
2018-03-23 02:03:36 received: reject (58 bytes) peer=1
2018-03-23 02:03:36 Reject tx code 16: txoverwinterexpired: hash
556a672cad8c1fad5f58d7bee103adc1cec98ff77c0af7ee2b9b6d7622eb5f1f
2018-03-23 02:03:36 received: reject (58 bytes) peer=1
2018-03-23 02:03:36 Reject tx code 16: txoverwinterexpired: hash
b523ca2550aed2a35d42e33e43d4b11545cbf434e6acad929a079ee5444b0893
2018-03-23 02:03:36 received: reject (58 bytes) peer=1
2018-03-23 02:03:36 Reject tx code 16: txoverwinterexpired: hash
1deb7bfae84d56c70bc7357c205954fe23dee87f0f415dcd69f7d701f2b50f5c
2018-03-23 02:03:36 received: reject (58 bytes) peer=1
2018-03-23 02:03:36 Reject tx code 16: txoverwinterexpired: hash
8675921a2e8d17abd1df035bb0211e89376e822ae1149c7acfba8657df59ff8e
2018-03-23 02:03:36 received: reject (58 bytes) peer=1
2018-03-23 02:03:36 Reject tx code 16: txoverwinterexpired: hash
3a3d6554a5aceb85ddbdd6d7ae093cffa62a4d474bced13ba4b553926b894130
2018-03-23 02:03:36 received: reject (58 bytes) peer=1
2018-03-23 02:03:36 Reject tx code 16: txoverwinterexpired: hash
b9a3434ba0d4fbae46a4a33d5728a7f321c7288cd555a425e2993b1438397d7f
2018-03-23 02:03:36 received: reject (58 bytes) peer=1
2018-03-23 02:03:36 Reject tx code 16: txoverwinterexpired: hash
6438a490b0564adbbd7caf5115013877e5d69e3d7f0ae45c79c6c2cb7627e3ce
2018-03-23 02:03:36 received: reject (58 bytes) peer=1
2018-03-23 02:03:36 Reject tx code 16: txoverwinterexpired: hash
139419cb29736af3cc42f84ce20dc368b44369573c4a260a77a2c500bf3f2170
2018-03-23 02:03:36 received: reject (58 bytes) peer=1
2018-03-23 02:03:36 Reject tx code 16: txoverwinterexpired: hash
29652e3323abcba8284c8906152ff12fc289ccdbecc9237842b75969237b54f1
2018-03-23 02:03:36 socket closed
2018-03-23 02:03:36 disconnecting peer=1
```

Debug.log from a peer of the victim

```
2018-03-23 02:03:35 getheaders 203 to 0000000000000000000000000000000000000000000000000000000000000000
from peer=4
2018-03-23 02:03:35 sending: headers (179 bytes) peer=4
2018-03-23 02:03:35 received: pong (8 bytes) peer=4
2018-03-23 02:03:35 received: headers (179 bytes) peer=4
2018-03-23 02:03:35   nActualTimespan = 10200  before dampening
2018-03-23 02:03:35   nActualTimespan = 4462  before bounds
2018-03-23 02:03:35 GetNextWorkRequired RETARGET
2018-03-23 02:03:35 params.AveragingWindowTimespan() = 2550    nActualTimespan = 2550
2018-03-23 02:03:35 Current average: 200f0f04
0f0f040f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f
```

```
2018-03-23 02:03:35 After:  200f0f04  0f0f040f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0740
2018-03-23 02:03:35 received: inv (1117 bytes) peer=1
2018-03-23 02:03:35 got inv: tx 185fc0698ff549a2587ab9868f79324d44990a31413bf521b713d95f3b8e6aa4  new
peer=1
2018-03-23 02:03:35 askfor tx 185fc0698ff549a2587ab9868f79324d44990a31413bf521b713d95f3b8e6aa4  0
(00:00:00) peer=1
2018-03-23 02:03:35 got inv: tx 556a672cad8c1fad5f58d7bee103adc1cec98ff77c0af7ee2b9b6d7622eb5f1f  new
peer=1
2018-03-23 02:03:35 askfor tx 556a672cad8c1fad5f58d7bee103adc1cec98ff77c0af7ee2b9b6d7622eb5f1f  0
(00:00:00) peer=1
2018-03-23 02:03:35 got inv: tx b523ca2550aed2a35d42e33e43d4b11545cbf434e6acad929a079ee5444b0893  new
peer=1
2018-03-23 02:03:35 askfor tx b523ca2550aed2a35d42e33e43d4b11545cbf434e6acad929a079ee5444b0893  0
(00:00:00) peer=1
2018-03-23 02:03:35 got inv: tx 1deb7bfae84d56c70bc7357c205954fe23dee87f0f415dcd69f7d701f2b50f5c  new
peer=1
[ … got inv / askfor repeated 25 times … ]
2018-03-23 02:03:35 sending: getheaders (645 bytes) peer=1
2018-03-23 02:03:35 getheaders (203) 01edb3c477e7a77719a1fa39d78e7d61686c45bdd7eb4200d7dd973c92cb205a
to peer=1
2018-03-23 02:03:35 sending: getdata (37 bytes) peer=1
2018-03-23 02:03:35 Requesting tx 185fc0698ff549a2587ab9868f79324d44990a31413bf521b713d95f3b8e6aa4
peer=1
2018-03-23 02:03:35 Requesting tx 556a672cad8c1fad5f58d7bee103adc1cec98ff77c0af7ee2b9b6d7622eb5f1f
peer=1
2018-03-23 02:03:35 Requesting tx b523ca2550aed2a35d42e33e43d4b11545cbf434e6acad929a079ee5444b0893
peer=1
2018-03-23 02:03:35 Requesting tx 1deb7bfae84d56c70bc7357c205954fe23dee87f0f415dcd69f7d701f2b50f5c
peer=1
2018-03-23 02:03:35 Requesting tx 8675921a2e8d17abd1df035bb0211e89376e822ae1149c7acfba8657df59ff8e
peer=1
[... Requesting tx repeated for additional 25 transactions … ]
2018-03-23 02:03:35 sending: getdata (1081 bytes) peer=1
2018-03-23 02:03:36 received: inv (253 bytes) peer=2
2018-03-23 02:03:36 got inv: tx 139419cb29736af3cc42f84ce20dc368b44369573c4a260a77a2c500bf3f2170  new
peer=2
2018-03-23 02:03:36 askfor tx 139419cb29736af3cc42f84ce20dc368b44369573c4a260a77a2c500bf3f2170
1521770614997414 (02:03:34) peer=2
[... more peer2 INVs ... ]
2018-03-23 02:03:36 received: headers (179 bytes) peer=1
2018-03-23 02:03:36   nActualTimespan = 10200  before dampening
2018-03-23 02:03:36   nActualTimespan = 4462  before bounds
2018-03-23 02:03:36 GetNextWorkRequired RETARGET
2018-03-23 02:03:36 params.AveragingWindowTimespan() = 2550    nActualTimespan = 2550
2018-03-23 02:03:36 Current average: 200f0f04
0f0f040f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f
2018-03-23 02:03:36 After:  200f0f04  0f0f040f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0740
2018-03-23 02:03:36 received: block (277 bytes) peer=1
2018-03-23 02:03:36 received block 01edb3c477e7a77719a1fa39d78e7d61686c45bdd7eb4200d7dd973c92cb205a
peer=1
2018-03-23 02:03:36   - Load block from disk: 0.00ms [0.00s]
2018-03-23 02:03:36       - Connect 1 transactions: 0.12ms (0.122ms/tx, 0.000ms/txin) [0.00s]
2018-03-23 02:03:36     - Verify 0 txins: 0.20ms (0.000ms/txin) [0.00s]
2018-03-23 02:03:36     - Index writing: 0.10ms [0.00s]
2018-03-23 02:03:36     - Callbacks: 0.05ms [0.00s]
2018-03-23 02:03:36   - Connect total: 0.56ms [0.00s]
2018-03-23 02:03:36   - Flush: 0.03ms [0.00s]
2018-03-23 02:03:36   - Writing chainstate: 0.03ms [0.00s]
2018-03-23 02:03:36 Blockpolicy recalculating dynamic cutoffs:
2018-03-23 02:03:36   2: For conf success > 0.85 need Priority >:           -1 from buckets      inf -
inf  Cur Bucket stats   -nan%       0.0/(0.0+0 mempool)
2018-03-23 02:03:36   2: For conf success > 0.85 need FeeRate >:           -1 from buckets      inf -
inf  Cur Bucket stats   -nan%       0.0/(0.0+0 mempool)
2018-03-23 02:03:36  10: For conf success < 0.50 need Priority <:           -1 from buckets 5.76e+07 -
5.76e+07  Cur Bucket stats   -nan%       0.0/(0.0+0 mempool)
```

```
2018-03-23 02:03:36  10: For conf success < 0.50 need FeeRate <:           -1 from buckets      100 -
100  Cur Bucket stats   -nan%       0.0/(0.0+0 mempool)
2018-03-23 02:03:36 Blockpolicy after updating estimates for 0 confirmed entries, new mempool map size
0
2018-03-23 02:03:36 UpdateTip: new
best=01edb3c477e7a77719a1fa39d78e7d61686c45bdd7eb4200d7dd973c92cb205a  height=204  log2_work=11.766943
tx=205  date=2018-03-23 02:03:35 progress=1.000000  cache=0.0MiB(1tx)
2018-03-23 02:03:36   - Connect postprocess: 0.38ms [0.00s]
2018-03-23 02:03:36 - Connect block: 0.99ms [0.00s]
2018-03-23 02:03:36 sending: inv (37 bytes) peer=3
2018-03-23 02:03:36 sending: inv (37 bytes) peer=4
2018-03-23 02:03:36 received: tx (234 bytes) peer=1
2018-03-23 02:03:36 ERROR: ContextualCheckTransaction(): transaction is expired
2018-03-23 02:03:36 ERROR: AcceptToMemoryPool: ContextualCheckTransaction failed
2018-03-23 02:03:36 185fc0698ff549a2587ab9868f79324d44990a31413bf521b713d95f3b8e6aa4 from peer=1
/MagicBean:1.0.15/ was not accepted into the memory pool: tx-overwinter-expired
2018-03-23 02:03:36 sending: reject (58 bytes) peer=1
2018-03-23 02:03:36 Misbehaving: 127.0.0.1:55090 (0 -> 10)
2018-03-23 02:03:36 received: getheaders (645 bytes) peer=3
2018-03-23 02:03:36 getheaders 204 to 01edb3c477e7a77719a1fa39d78e7d61686c45bdd7eb4200d7dd973c92cb205a
from peer=3
2018-03-23 02:03:36 sending: headers (179 bytes) peer=3
2018-03-23 02:03:36 received: tx (234 bytes) peer=1
2018-03-23 02:03:36 ERROR: ContextualCheckTransaction(): transaction is expired
2018-03-23 02:03:36 ERROR: AcceptToMemoryPool: ContextualCheckTransaction failed
2018-03-23 02:03:36 556a672cad8c1fad5f58d7bee103adc1cec98ff77c0af7ee2b9b6d7622eb5f1f from peer=1
/MagicBean:1.0.15/ was not accepted into the memory pool: tx-overwinter-expired
2018-03-23 02:03:36 sending: reject (58 bytes) peer=1
2018-03-23 02:03:36 Misbehaving: 127.0.0.1:55090 (10 -> 20)
2018-03-23 02:03:36 received: getdata (37 bytes) peer=3
2018-03-23 02:03:36 received getdata (1 invsz) peer=3
2018-03-23 02:03:36 received getdata for: block
01edb3c477e7a77719a1fa39d78e7d61686c45bdd7eb4200d7dd973c92cb205a peer=3
2018-03-23 02:03:36 sending: block (277 bytes) peer=3
2018-03-23 02:03:36 received: tx (234 bytes) peer=1
2018-03-23 02:03:36 ERROR: ContextualCheckTransaction(): transaction is expired
2018-03-23 02:03:36 ERROR: AcceptToMemoryPool: ContextualCheckTransaction failed
2018-03-23 02:03:36 b523ca2550aed2a35d42e33e43d4b11545cbf434e6acad929a079ee5444b0893 from peer=1
/MagicBean:1.0.15/ was not accepted into the memory pool: tx-overwinter-expired
2018-03-23 02:03:36 sending: reject (58 bytes) peer=1
2018-03-23 02:03:36 Misbehaving: 127.0.0.1:55090 (20 -> 30)
2018-03-23 02:03:36 received: tx (234 bytes) peer=1
2018-03-23 02:03:36 ERROR: ContextualCheckTransaction(): transaction is expired
2018-03-23 02:03:36 ERROR: AcceptToMemoryPool: ContextualCheckTransaction failed
2018-03-23 02:03:36 1deb7bfae84d56c70bc7357c205954fe23dee87f0f415dcd69f7d701f2b50f5c from peer=1
/MagicBean:1.0.15/ was not accepted into the memory pool: tx-overwinter-expired
2018-03-23 02:03:36 sending: reject (58 bytes) peer=1
2018-03-23 02:03:36 Misbehaving: 127.0.0.1:55090 (30 -> 40)
2018-03-23 02:03:36 received: tx (234 bytes) peer=1
2018-03-23 02:03:36 ERROR: ContextualCheckTransaction(): transaction is expired
2018-03-23 02:03:36 ERROR: AcceptToMemoryPool: ContextualCheckTransaction failed
2018-03-23 02:03:36 8675921a2e8d17abd1df035bb0211e89376e822ae1149c7acfba8657df59ff8e from peer=1
/MagicBean:1.0.15/ was not accepted into the memory pool: tx-overwinter-expired
2018-03-23 02:03:36 sending: reject (58 bytes) peer=1
2018-03-23 02:03:36 Misbehaving: 127.0.0.1:55090 (40 -> 50)
2018-03-23 02:03:36 received: tx (234 bytes) peer=1
2018-03-23 02:03:36 ERROR: ContextualCheckTransaction(): transaction is expired
2018-03-23 02:03:36 ERROR: AcceptToMemoryPool: ContextualCheckTransaction failed
2018-03-23 02:03:36 3a3d6554a5aceb85ddbdd6d7ae093cffa62a4d474bced13ba4b553926b894130 from peer=1
/MagicBean:1.0.15/ was not accepted into the memory pool: tx-overwinter-expired
2018-03-23 02:03:36 sending: reject (58 bytes) peer=1
2018-03-23 02:03:36 Misbehaving: 127.0.0.1:55090 (50 -> 60)
2018-03-23 02:03:36 received: tx (234 bytes) peer=1
2018-03-23 02:03:36 ERROR: ContextualCheckTransaction(): transaction is expired
2018-03-23 02:03:36 ERROR: AcceptToMemoryPool: ContextualCheckTransaction failed
```

```
2018-03-23 02:03:36 b9a3434ba0d4fbae46a4a33d5728a7f321c7288cd555a425e2993b1438397d7f from peer=1
/MagicBean:1.0.15/ was not accepted into the memory pool: tx-overwinter-expired
2018-03-23 02:03:36 sending: reject (58 bytes) peer=1
2018-03-23 02:03:36 Misbehaving: 127.0.0.1:55090 (60 -> 70)
2018-03-23 02:03:36 received: tx (235 bytes) peer=1
2018-03-23 02:03:36 ERROR: ContextualCheckTransaction(): transaction is expired
2018-03-23 02:03:36 ERROR: AcceptToMemoryPool: ContextualCheckTransaction failed
2018-03-23 02:03:36 6438a490b0564adbbd7caf5115013877e5d69e3d7f0ae45c79c6c2cb7627e3ce from peer=1
/MagicBean:1.0.15/ was not accepted into the memory pool: tx-overwinter-expired
2018-03-23 02:03:36 sending: reject (58 bytes) peer=1
2018-03-23 02:03:36 Misbehaving: 127.0.0.1:55090 (70 -> 80)
2018-03-23 02:03:36 received: tx (234 bytes) peer=1
2018-03-23 02:03:36 ERROR: ContextualCheckTransaction(): transaction is expired
2018-03-23 02:03:36 ERROR: AcceptToMemoryPool: ContextualCheckTransaction failed
2018-03-23 02:03:36 139419cb29736af3cc42f84ce20dc368b44369573c4a260a77a2c500bf3f2170 from peer=1
/MagicBean:1.0.15/ was not accepted into the memory pool: tx-overwinter-expired
2018-03-23 02:03:36 sending: reject (58 bytes) peer=1
2018-03-23 02:03:36 Misbehaving: 127.0.0.1:55090 (80 -> 90)
2018-03-23 02:03:36 received: tx (234 bytes) peer=1
2018-03-23 02:03:36 ERROR: ContextualCheckTransaction(): transaction is expired
2018-03-23 02:03:36 ERROR: AcceptToMemoryPool: ContextualCheckTransaction failed
2018-03-23 02:03:36 29652e3323abcba8284c8906152ff12fc289ccdbecc9237842b75969237b54f1 from peer=1
/MagicBean:1.0.15/ was not accepted into the memory pool: tx-overwinter-expired
2018-03-23 02:03:36 sending: reject (58 bytes) peer=1
2018-03-23 02:03:36 Misbehaving: 127.0.0.1:55090 (90 -> 100) BAN THRESHOLD EXCEEDED
```

ZEC-013 Transaction Expiry Enables Transaction Flooding at No Cost

Total Risk	Impact	Location
High	High	src/main.cpp
Fixed ✔	Likelihood: High	Category: Denial of Service

Description

The new expiry mechanism allows an attacker to flood the network with transactions without paying any cost if the attacker sends transactions close to expiry. This flooding could be used maliciously for several attacks. One of which is the miner to miner attack that follows: When a malicious miner Mallory detects a new block B on the network created by another miner (say Alice), Mallory can broadcast thousands of transactions that expire in the same block height, and therefore cannot be included in any following block. By doing so, Mallory can delay the propagation of the block B. The more connected Mallory is, the higher the chances she has to delay the block B. At the same time Mallory will mine on top of B as normal. The remaining miners (say Bob) will not get notified of the new block B on time, and therefore Bob will be mining a competing block. It's highly probable that any competing block created during this interval loses the race to be considered the best block. This is because both Mallory and Alice are already mining on top of B, and B is propagating first. The higher the mining concentration is, the higher the effect of this attack. For example, having a 10 minute block interval, delaying propagation of B by 30 seconds will make Bob waste approximately 5% of the his hashing time.
The attack can be applied both to Alice and Bob, given Mallory a sustained advantage.

Recommendations

It is recommended that transactions close to expiry (e.g. only one block left) are not propagated by the network. This shouldn't bring new problems since there is no known use case where transactions should be created with such a short lifespan.

Alternatively, UTXOs leading to expiring transactions could be temporarily banned so that transactions that use those UTXOs are temporarily not propagated by the network. The banning interval can be doubled each time a valid transaction consuming those UTXOs expires.

6. Fix for ZEC-013

Zcash developed the first fix suggested by the Coinspect team in the git pull request #3689 found at https://github.com/zcash/zcash/pull/3689. Coinspect consultants reviewed the new code and verified this fix was correctly implemented and the reported vulnerability does not exist anymore.

The new code introduces a check to guarantee transactions which expire soon are not accepted into the mempool. The criteria to consider a transaction to be expiring soon is determined by the `TX_EXPIRING_SOON_THRESHOLD` constant, which is set to 3 blocks. Additionally, those transactions are prevented from being relayed to other peers.

As a consequence, it is not longer possible to spam the network with transactions with no cost, because this means the attacker would need to increase the expiration block number, and can not be certain his transactions will not be included in a block, thus paying for their cost. In a scenario where blocks are continuously full, an attacker could try to game the fee market to prevent his transactions from being mined before expiry. Then, a bigger expiry height threshold from the current block should be enforced to accept and/or relay a transaction.