Argent Audit Technical Report

High Severity Vulnerability

OPENZEPPELIN | JUNE 18, 2020

Security Audits

Summary

A high-severity vulnerability in the Argent wallet would have allowed attackers to take over wallets with no guardians. User action would have been needed to prevent the takeover attack in less than 36 hours, which then would have opened an alternative Denial of Service (DoS) attack vector with potential to indefinitely freeze their funds.

Our initial analysis reported 329 wallets at immediate risk in mainnet, with nearly 162 ETH in total holdings, plus additional quantities of tokens and DeFi holdings. Additionally we identified 5513 wallets with no guardians which would become vulnerable as soon as they upgraded to the latest version of Argent's contracts, although Argent reports the majority of these are inactive and should not be considered Argent users.

Upon our private disclosure of the vulnerability to Argent, immediate action from their team and affected users was required to keep funds safe.

The vulnerability has been assigned CVE 2020-15302.

Timeline of Events

- June 19th: Argent fixed and updated the vulnerable smart contract in mainnet and released a new version of mobile application. Both teams publicly disclosed the issue.

The Vulnerability

All code in this article references commit f874e4c8 in the `master` branch of the https://github.com/argentlabs/argent-contracts repository, the latest commit as of June 12 2020 (day on which the vulnerability was reported to Argent).

Argent users can add “guardians” to their wallets. These are trusted accounts (externally-owned or contracts) that are given permission to execute specific actions on the wallet. Today, all Argent wallets created in the official mobile application include a default guardian (controlled by Argent), added during creation of the wallet, using Argent’s latest `WalletFactory` contract (deployed on March 30th 2020). However, all wallets created *before* March 30th 2020 were initially created without a guardian, since they were created using Argent’s old `WalletFactory` contract, which did not allow adding a guardian during creation.

Furthermore, the contracts impose no restrictions on the number of guardians a wallet may have, allowing wallets without guardians. Users may decide to revoke the default guardian added by Argent, add a user-controlled guardian (such as a hardware wallet), or add a trusted-party’s Argent wallet as a guardian.

One of the several powers guardians are granted is the ability to trigger the recovery of a wallet – a process that, when successfully executed, sets a new account as the wallet’s owner. Wallet recovery is a two-step process: first it is expected to be triggered by a guardian, and then after a 36-hour window it can be effectively finalized.

The wallet recovery process is triggered via the `executeRecovery` function of the `RecoveryManager` contract, providing as parameters the target wallet and its new owner. This function can only be called by the `execute` function (inherited from the `RelayerModuleV2` contract), which applies a number of validations before forwarding the call. Among the validations, the `execute` function uses the `getRequiredSignatures` function to determine how many signatures are required to begin the wallet recovery process.

```
if (methodId == EXECUTE_RECOVERY_PREFIX) {
    return SafeMath.ceil(guardianStorage.guardianCount(_wallet)
}
```

The `guardianCount` function of the `GuardianStorage` contract, in charge of keeping track of how many guardians wallets have, returns zero for wallets with no guardians. In such a scenario, the snippet above would end up calculating `ceil(0/2)`, which results in 0. As a consequence, the `executeRecovery` function can be called without signatures for a wallet with zero guardians – and therefore be triggered by any attacker wishing to take control of the wallet and steal its funds.

After calling the `executeRecovery` function, the attacker must wait for the duration of the `recoveryPeriod` (which defaults to 36 hours), and will then be able to execute `finalizeRecovery` to complete the recovery process of the target wallet. During the 36-hour window the wallet is locked by the `LockManager` contract.

The owner of the wallet has the opportunity to prevent the wallet takeover by calling the `cancelRecovery` function before the end of the `recoveryPeriod`. Argent reports that their monitoring system would send an SMS and email to the user notifying them of the recovery attempt, prompting them to cancel it if they did not lose their wallet.

Even in the case where the user successfully cancels the takeover, the attacker can immediately trigger the attack again targeting the same wallet. By repeating this process, the attacker can perform a Denial of Service (DoS) attack forcing the victim wallet to remain in the `recoveryPeriod` and thereby becoming permanently locked as long as the attack continues. Maintaining a locked state means that the owner cannot execute actions such as:

- Adding a guardian to the wallet (which would halt the attack).
- Transferring any funds out of the wallet.
- Closing their positions in Compound or Maker, potentially putting the wallet's loans at risk of being liquidated.

execute the necessary actions to do it in less than 36 hours, while all their funds are at risk.

Furthermore, sensible attackers will likely carry out this attack in times of high network congestion, when it would be particularly difficult and expensive for a user to submit the transaction to cancel the recovery and add a guardian in the short time window.

Wallets at Risk

We provided Argent with proof-of-concept exploits to reproduce the vulnerability, together with all on-chain analysis scripts we used to rapidly identify users at risk. These were fundamental to understanding the severe impact of the vulnerability were it to be exploited in the wild.

Wallets had to meet 2 criteria to be at immediate risk of attack. They must have had 0 guardians, and have upgraded from the old `RecoveryManager` module to the new one which was deployed on May 7th. At the time of reporting the vulnerability, there were 329 wallets that met such criteria, 74 of which had positive ETH or token holdings. Only 13 of the 329 wallets had added a guardian after 3 days of our initial report.

Moreover, at the time of our analysis we found there were a further 5513 wallets that had 0 guardians and had not *yet* upgraded to use the new `RecoveryManager`. The next time that these users opened their Argent app, they would need to upgrade to the flawed `RecoveryManager` to continue using the mobile application, and become vulnerable to attack. According to Argent, the majority of these wallets were created during Argent's beta release, never used the wallet, and should therefore not be considered Argent users. Nonetheless, we found out that in the 3 days after reporting the vulnerability to Argent, 22 such users upgraded their wallet and opened themselves up for attack.

The Fix

Since it was unclear from Argent's documentation whether a wallet should be allowed to have zero guardians, in our initial report we proposed different approaches to fix the vulnerability.

a wallet had no guardians, the function did not return 0.

Argent's team arrived at the same conclusions, and reported they would be implementing our suggested fix. We advised Argent that the fix should be reviewed by Argent's trusted security auditors. In the meantime, Argent reported that they were already contacting affected users encouraging them to add at least one guardian to their wallets as soon as possible. Finally, Argent deployed the fixed version of the `RecoveryManager` contract to mainnet, which can be found at address 0xdc350d09f71c48c5d22fbe2741e4d6a03970e192.

As part of Argent's public bug bounty program, both teams agreed on classifying the vulnerability as High severity.

We must highlight that OpenZeppelin and Argent collaborated throughout the entire responsible disclosure process to prevent Argent affected users losing funds.

Related Posts

Beefy Zap Audit

BeefyZapRouter serves as a versatile intermediary designed to execute users' orders through routes...

OpenBrush Contracts Library Security Review

OpenBrush is an open-source smart contract library written in the Rust programming language and the...

Linea Bridge Audit

Linea is a ZK-rollup deployed on top of Ethereum. It is designed to be EVM-compatible and aims to...