Fine penetration tests for fine websites

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Audit-Report Greymass Antelope Snap & Codebase 09.2024

Cure53, Dr.-Ing. M. Heiderich, Dr. D. Bleichenbacher, Dr. N. Kobeissi

Index

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Introduction

“We're Greymass, an imaginative team of developers and researchers who are paving the way to a better decentralized internet. Our guiding vision is an easy Web3 user experience for everyone, whether you're a power user or just starting out. Our Anchor and Unicove products are designed to be both intuitive and safe.”

From https://www.greymass.com/

This report describes the results of a penetration test and source code audit against the Greymass Antelope Snap and its codebase. The inspection was conducted by Cure53 in September 2024.

More precisely, the inspection - registered as *GRY-01* - was requested by Greymass Inc. in August 2024 and carried out by Cure53 in CW39 of 2024. A total of four days were invested to reach the coverage expected for this project. Moreover, it can be mentioned that a team consisting of three senior testers was created and then assigned to this project’s preparation, execution and finalization.

The work was structured using a single work package (WP):

- **WP1**: Penetration tests & code audits against Greymass Antelope Snap & codebase

Cure53 was provided with sources, as well as all further means of access required to complete the tests. The methodology chosen as appropriate for assessing the targets of *GRY-01* should be seen as a white-box approach. All preparations were done in September 2024, namely CW38. This meant that Cure53 could have a smooth start into the testing process scheduled for the following week.

Communications during the test were done using a dedicated shared Slack channel set up to connect the Greymass and Cure53 teams. All involved personnel from both parties could join the discussions on Slack. Not many questions had to be asked; the scope was well prepared and clear. Correspondingly, no noteworthy roadblocks were encountered during the test.

Cure53 gave frequent status updates about the test and the related findings; live-reporting was not specifically requested for this audit. The Cure53 team managed to get good coverage over the scope items. Two security-related findings were observed and classified as security vulnerabilities. One received a *Medium* and the other *Low*-severity score.

Overall, the provided MetaMask Snap code for Antelope integration left a positive impression on the Cure53 team. The deployed security mechanisms are sufficient to showcase a functional way to extend the capabilities of MetaMask.

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Nevertheless, it should be noted that an already good level of security can always be further improved by implementing the recommendations outlined in connection with the two findings. The report will now shed more light on the scope and test setup as well as the available material for testing.

This section will be followed by a chapter that details the test methodology used in this *GRY-01* exercise. This is to show which areas of the software in scope have been covered and what tests have been executed. This might be useful in light of only two actual tickets presented in this report.

After that, the report will list all findings chronologically in the category labeled as *Vulnerabilities* Each finding will be accompanied with a technical description, a PoC where possible as well as mitigation or fix advice.

The report will then close with a conclusion in which Cure53 will elaborate on the general impressions acquired via this *GRY-01* test. The final section also offers some words about the perceived security posture and potential recommendations that the Greymass Antelope Snap and its codebase could benefit from.

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Scope

- **Code audits & security reviews against Greymass Antelope Snap & related codebase**
  - **WP1:** Penetration tests & code audits against Greymass Antelope Snap & codebase
    - **Repository URL:**
      - https://github.com/greymass/antelope-snap
    - **Branch:**
      - Master
    - **Commit:**
      - 6e8d2f190f161157b71c23781cccc10e9c9c0eb9
  - **Test-supporting material was shared with Cure53**
  - **All relevant sources were shared with Cure53**

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Test Methodology

This section documents the testing methodology applied by Cure53 during this project and discusses the resulting coverage, shedding light on how various components were examined. Further clarification concerning areas of investigation subjected to deep-dive assessment is offered, as it was deemed especially necessary in the absence of significant security vulnerabilities on the findings' list.

The source code provided by Greymass was small enough so that it was possible to review it manually line by line. The review focused on several aspects that are detailed below.

One important aspect is the handling of confidential information. Exposure of confidential information should be kept to a minimum. GRY-01-001 points to a case where information is unnecessarily exposed in logs.

The reviewers checked for defensive programming. Unexpected situations should be caught by the code and be handled gracefully. GRY-01-002 documents a case where exceptions are not caught. Such oversights can lead to Denial-of-Service attacks or even data exposure if the code handling exceptions is not aware of the causes of such exceptions.

The code was compared against the underlying protocol. One important aspect that was analyzed encompassed the security requirements of BIP44. In particular, BIP44 requires that the first nodes of a derivation path are hardened. The code that was reviewed does not perform any validity checks on the paths. However, it was confirmed that the underlying library performs checks on the format of the derivation path.

The code was checked against common pitfalls in protocols, such as key reuse for multiple distinct primitives. The implementation restricts the set of primitives and curves, leaving no room for such issues.

A cursory (i.e., interfaces-only) review of the underlying library *metamask* concluded that this library has sufficiently robust interfaces. Any issues stemming from the use of this library would be unexpected.

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Identified Vulnerabilities

The following section lists all vulnerabilities and implementation issues identified during the testing period. Notably, findings are cited in chronological order rather than by degree of impact, with the severity rank offered in brackets following the title heading for each vulnerability. Furthermore, each ticket has been given a unique identifier (e.g., *GRY-01-001*) to facilitate any follow-up correspondence in the future.

GRY-01-001 WP1: Exposure of sensitive data via console logging *(Low)*

Fix Note: *The issue has been addressed by the development team and the fix was verified by Cure53. The problem as described no longer exists.*

The *onRpcRequest* handler includes a *console.log(request);* statement, which logs the entire incoming request object. The logged request may contain sensitive information, such as transaction details in *antelope_signTransaction*. The revealed information could include amounts, recipient addresses and other private data.

Affected file:
src/index.ts

Affected code:

```
export const onRpcRequest: OnRpcRequestHandler = async ({ request }) => {
  console.log(request);
  switch (request.method) {
    case 'antelope_getPublicKey':
      return await getPublicKey(request as AntelopeRequest);

    case 'antelope_signTransaction':
      return String(await signTransaction(request as
AntelopeSignatureRequest));

    default:
      // eslint-disable-next-line @typescript-eslint/no-throw-literal
      throw new MethodNotFoundError(request.method);
  }
};
```

Logging sensitive information violates the principle of least-privilege and can lead to unintended data exposure. It is recommended that the *console.log* statement be eliminated to prevent any and all logging of sensitive data. If logging is necessary for debugging, implementing conditional logging would be a better solution. This method could be toggled during development but remain disabled in production builds.

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

GRY-01-002 WP1: Potential DoS attacks via transaction data *(Medium)*

Fix Note: *The issue has been addressed by the development team and the fix was verified by Cure53. The problem as described no longer exists.*

The Antelope Snap uses *JSON.parse(request.params.transaction)* in order to parse the transaction data without any additional validation. Maliciously crafted transaction data could potentially cause issues during parsing or processing. At the very least, large or malformed transactions could cause the Snap to crash or behave unexpectedly.

Affected file:
src/rpc.ts

Affected code:

```
export async function signTransaction(
  request: AntelopeSignatureRequest,
): Promise<Signature | undefined> {
  // Process incoming transaction
  if (!request.params?.transaction) {
    throw new Error('Missing transaction in request params');
  }
  const transaction =
Transaction.from(JSON.parse(request.params.transaction));

  // Load the appropriate chain definition
  if (!request.params?.chainId) {
    throw new Error('Missing chainId in request params');
  [...]
```

It is recommended to implement robust validation of the transaction data before processing. Try-catch blocks, in addition to length checks on the JSON string prior to attempting parsing, can aid graceful handling of parsing errors or malicious payloads.

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Conclusions

This *GRY-01* audit, performed by Cure53 in September 2024, focused on the MetaMask Snap code for Antelope integration. As the number of findings is limited, the project concludes that the targets can be judged as sound and robust. Nevertheless, certain observations and discoveries have been made and will be detailed next.

GRY-01-001 shows how the Antelope snap RPC logs the entire incoming request object. This potentially contains sensitive information like transaction details, amounts, and recipient addresses. Logging such sensitive data can lead to unintended exposure if an attacker accesses the console logs, violating the principle of least privilege.

It is recommended to remove or limit the processes linked to logging of sensitive data. Proactive actions in this realm could prevent potential leaks and enhance security.

In addition, GRY-01-002 uses *JSON.parse* to parse transaction data without additional validation, making it susceptible to maliciously crafted inputs. Such inputs could cause parsing errors or crashes, leading to DoS or unexpected behavior in the Snap. Implementing robust validation and error handling of the transaction data before processing is recommended to mitigate this risk.

The application builds on a library that implements BIP44 and is - to a large degree - a wrapper around this library. An important aspect in such cases is the robustness of this wrapper. Ideally, such a wrapper catches invalid or unsafe calls.

An example is the expectation that the first nodes of a key derivation path are hardened. These types of requirements are sometimes overlooked. A focus of the audit was to detect such requirements and check that either the provided MetaMask Snap code or the underlying library are robust enough to detect such errors.

Another critical point is to ensure that key material is never reused for multiple purposes. With regard to BIP44, it is for example important that the same private keys have never been used together with both *secp256k1* and *secp256r1.* This type of usage facilitates additional attack vectors. The provided solution simply hardcodes the curve being used. As a result, no key confusion is possible.

All in all, the provided MetaMask Snap code for Antelope integration demonstrates a functional approach to extending the existing capabilities of MetaMask. A small number of security concerns demonstrates this, although the detailed recommendations need to be addressed to ensure safety and trust of end-users.

Cure53 would like to thank Daniel Fugere from the Greymass, Inc. team for their excellent project coordination, support and assistance, both before and during this assignment.