Fine penetration tests for fine websites

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Audit-Report Fysical Smart Contract 01.-02.2018

Cure53, Dr.-Ing. M. Heiderich, MSc. N. Kobeissi, BSc. F. Fäßler

Index

Introduction

“Fysical is a decentralized location data market. It helps businesses and machines make better decisions by providing them with high quality data on how humans move through the physical world.”

From https://fysical.com/

This report documents the findings of a security audit targeting a smart contract provided by Fysical Technologies Pte. Ltd. The project was carried out by Cure53 in late January and early February 2018. It yielded four findings and led to a formulation of various recommendations.

As for the resources, approaches and scope, it should be noted that the main focus of this assessment was placed on the Fysical’s smart contract feature which will be used in a decentralized data market location. The audit, which was completed by three members of the Cure53 team, relied on the so-called white-box methodology. This signifies that the testers had full access to the relatively complex but easy to read smart contract sources. These also included extensive comments composed in the Solidity language. The time budget allocated to this project entailed five days for testing, communications and reporting.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Before the investigations of the smart contract in scope began, the in-house team at Fysical and the Cure53 testing team held a dedicated kick-off meeting. This meeting served as means to clarify the purpose of the product and communicate the risks observed by the internal developers. It was agreed that the Cure53's tasks would be to audit the sources for security issues, logical bugs and other problems that might put the Fysical business model in danger. What is more, it was to be examined whether any potential existed for threatening the safety of the Fysical clients' and users' assets. Formal correctness and other technical issues were also added to the scope. Throughout the duration of the project, the Fysical and Cure53 teams remained in close contact. A dedicated, private Slack channel was used to discuss arising questions. Later on, an extensive discussion was held on Skype.

In regard to the outcomes, the auditors managed to find four general weaknesses in the smart contract sources. More importantly, the testers issued a small set of general recommendations for the Fysical team to consider. Both the findings and the broader action items are listed in this report, which will now quickly shed light on the scope. It then documents the results on the case-by-case basis, with further recommendations ensuing. The report closes with a conclusion about the state of security matters on the Fysical's scope in light of this Cure53 assessment.

Note: *This issues documented in this report were reviewed by Fysical and the subsequent changes and comments were discussed with Cure53. Each documented issue was added a note about the current status. The original report texts were left untouched.*

Scope

- **Fysical Smart Contracts**
  - Audited commit ID: 3d57d5c46ebfa3f0f416ce0569d598a78c441738

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Miscellaneous Issues

This section covers those noteworthy findings that did not lead to an exploit but might aid an attacker in achieving their malicious goals in the future. Most of these results are vulnerable code snippets that did not provide an easy way to be called. Conclusively, while a vulnerability is present, an exploit might not always be possible.

FYS-01-001 Restructure Meta-Resources to avoid infinite Ref. Chain *(Medium)*

Note: *This issue was acknowledged by the Fysical team and better documentation was added to advise about potential risks.*

Fysical currently allows specifying *ResourceSets* with the logic that each of the given sets holds a list of *Resources* and a pointer to a *metaResourceSet*. Each *Resource*, which can be linked either via a regular *ResourceSet* or with a *metaResourceSet*, can also have an additional link to the *metaResourceSet*. This setup creates a potential for an infinite reference chain. In effect, it comprise a flaw that can be abused to create unmanageable data structures in the Fysical's ledger.

Cure53 recommends that a *ResourceSet* should hold a list of *Resources* instead. These listed *Resources* should only be able to point to a single *metaResourceSet* per each item. Correspondingly, a *metaResourceSet* should only contain *metaResources* and exclude *Resources*, while *metaResources* should not be able to point to further *Resources*, *metaResources*, or anything else.

FYS-01-002 ResourceSet should check Resource Creator *(Medium)*

Note: *The client reviewed the proposal and decided to accept the risk.*

Sharing some similarity with FYS-01-001, there is another *Resource* data structure-related issue. In order to create a set of *Resources* to be offered and sold through the Fysical smart contract, multiple *Resources* have to be combined. These individual *Resources* do not record or check the *Creator*, thus allowing anybody to create a fake *ResourceSet* for sale. This *Set* can contain trusted *Resources* from other brokers.

Affected Data Structures:
A *Resource* stores a reference to a URI along with all information that should convey trust in the *Resource*, such as the *meta resource* information. However, it does not specify the *Creator* of the *Resource*.

```
struct Resource {
        uint256 uriSetId;
        uint256 encryptionAlgorithmId;
        uint256 metaResourceSetId;
```

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

```
}
```

Conversely, the set of *Resources* combining multiple *Resources* together for a sale proposal stores the *Creator* information.

```
struct ResourceSet {
        address creator;
        uint256 creatorPublicKeyId;
        uint256 proposalEncryptionAlgorithmId;
        uint256[] uniqueResourceIdsSortedAscending;
        uint256 metaResourceSetId;
}
```

The smart contract could only allow linking *Resources* from the same *Creator*, so that resellers would have to create an entirely new *Resource,* even though a necessary given *Resource* had been already well-described in the past. Therefore, by simply recording the *Creator,* it would be possible to improve accountability. In other words, clients would be able to interact with the data to identify mismatches with *Creators*. Depending on the situation, this could help the identification of malicious deception or it could generally provide a better understanding of the original source of a *Resource*.

FYS-01-003 Escrow Namespace mixed with Balance Namespace *(Low)*

***Note:** The client reviewed the proposal and decided to accept the risk.*

In many instances of the Fysical smart contract's code, a shorthand involving the Ethereum address "0" is used in order to transfer tokens into an "*Escrow*" address space.

Example code:
Fysical.sol line 948:

```
balances[address(0)] = balances[address(0)].add(tokenTransfer.tokenCount);
```

The reason behind deploying this shortcut was linked to proving that the *escrow* address was not within the control of any Fysical's protocol principals. While this is indeed accomplished by this setting, it also unnecessarily overloads the existing functions. Especially affected are events such as *Transfer*, which are strictly meant to track balance transfers between the actual Fysical's principals. At the same time, the aforementioned "0" Ethereum address refers to a *null* address index.

It is currently unclear whether this approach should be seen as unsafe, but Cure53 nevertheless recommends specifying a parallel *namespace* for *escrow* transactions and

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

balance. Specifying an *escrowBalances* mapping, separated from the currently employed *balances* mapping, would also allow tracking *escrow* transactions per user per transaction. It would replace present handling which relies on lumping all *escrow* sums into a single balance mapping.

Furthermore, specifying an independent set of *EscrowTransfer* events would make it possible to differentiate the *escrow* transfers from the regular transfers that involve actual users. The *ERC20* standard[1] specifies a *Transfer* event which must be mandatorily called for all transfers. This item presumably includes *escrow*-type transfers. However, double-registering *escrow* transfers via the *EscrowTransfer* event only would then mean that the *escrow* events can be told apart from the non-escrow events in the future.

FYS-01-004 No guarantees on URI Security or Correctness *(High)*

Note: *This issue was acknowledged by the Fysical team and better documentation was added to advise about potential risks.*

In the present deployment, *Resource* providers are allowed to set any URI on the Fysical public ledger as a pointer to a particular *Resource*. There are various potential problems with URIs that are dependent on how brokers will interact at the application layer within the smart contract. For example:

- **Buyer de-anonymization:** A provider can point to a *Resource* on a web server they own. This approach means tracking the IP address (or other information) of the buyer as they access the URI to obtain the encrypted *Resource*.
- **Malformed URIs:** A URI could potentially be submitted as a data URI[2]. Depending on the application layer - which Cure53 had notably no access to during this audit - this could present security issues or mislead users. Overall, it is certainly an unintended effect that can be ruled out with simple URI validation.
- **Resource Unavailability:** In simple terms, a buyer could simply make a *Resource* unavailable right after a payment has been concluded.

The current proposal for a “checksum” appears to be one of a list of checksum algorithms. This is unnecessary as checksums are meant for error correction, which is not something that needs to be mitigated for transfers over protocols such as HTTPS. Cure53 suggests scrapping checksums for a design where both the file seller and buyer provide keyed hashes, thus proving the integrity of the encrypted data in the process.

[1] https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md
[2] https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Data_URIs

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

This design could be as follows:

1. Before posting the file for sale on the Fysical ledger, the seller calculates the output of the following HKDF function[3]: *HKDF(f, k, 2) = (h1, h2)* where *f* is a SHA-2 hash of the encrypted file, *k* is a hashing key, and *2* is the number of desired outputs.
2. Instead of posting a checksum, the seller posts *(k, h1)* as the verifying data.
3. The client downloads the encrypted file before sending a Proposal. When sending their Proposal, the client is obliged to also provide *h2*, which they can only do if they have correctly obtained a local copy of the file.
4. Within the Proposal, the client includes *h2*. This allows anyone on the ledger to verify whether the client has indeed obtained the correct encrypted data without compromising the integrity of the transfer.

Important security note

The above design is only secure for verifying the *integrity* of the data transfer. Crucially, it does not verify *authenticity* of the data. This must be done by additionally deploying an AEAD cipher or encrypt-then-HMAC construction for file encryption. What is more, strategies might be put in place to ensure that the secret keys are communicated over an authenticated channel. To clarify, an authenticated channel is a channel where both principals are mutually certain of their identity.

Ideally, the Fysical compound could require more restrictions with reference to how URIs are supposed to be used. Additional verification should be considered, for example in a sense that only a HTTP or FTP URI is approved. However, note that this approach will restrict the smart contract's flexibility.

[3] https://tools.ietf.org/html/rfc5869

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

General Recommendations

In this section, Cure53 lists potential future directions in the Fysical's development. These could which could be beneficial from a security standpoint as well as more broadly. Two ideas have already been discussed internally during the test and deserve a mention in this report.

Buyer Reputation Metrics Integrated into the Public Ledger

According to the Fysical team, it is currently the case that sellers are pre-audited via a network of trusted brokers. This process is to determine their trustworthiness before they are allowed to trade on the Fysical platform. Trusted sellers are integral to the security and efficacy of the platform, especially given that all data exchanged on the platform is totally encrypted until after payment is made. This means that no guaranteed method for detecting deceptive sales exists.

In the future, it would be a good idea for Fysical to implement a revised smart contract logic, specifically by asking buyers to rate sellers. If these ratings were recorded and verified on the Fysical's public ledger, it would allow for the integration of a wider market of sellers. In addition, it would mean that buyers have a stake in determining the trust of a seller by themselves instead of relying on brokers. Though a revised model would have to account for the potential problem of fake reviews, the trust in the brokers could be alleviated and not taken for granted.

Proving Information About Dataset Using Zero-Knowledge Constructions

In zero-knowledge arguments, a *verifier* wants to check that a *prover* possesses secret knowledge satisfying some stated conditions[4]. Recent work by researchers at UCL was able to describe a zero-knowledge system that can verify the simulation of gas molecules in under 7kb and with new speed records[5].

A future endeavor could apply this research outputs to the Fysical's smart contracts, allowing buyers to provide proof about dataset contents without actually revealing it. For example, an encrypted dataset can come with a zero-knowledge proof that it contains information about the population movements in the city of New York.

[4] https://www.benthamsgaze.org/2016/10/25/how-to-do-zero-knowledge-from-discrete-logs-in-under-7kb/
[5] https://eprint.iacr.org/2016/263

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Conclusions

The results of this Cure53 security audit of the smart contract operated by Fysical are positive. Despite the relatively complex data structures and functions, the smart contract appears very strong in terms of security properties. The overall small number of findings is an outstandingly positive sign, attesting to the fact that the Fysical team is developing their smart contract product with security in mind.

It should be noted that this Cure53-Fysical cooperation enabled in-depth discussions about the broader notions around the smart contract project's objectives and security premises. This enabled Cure53 to review design decisions made by Fysical in the context of potential security threats. While it has been understood that Fysical could justify various choices, Cure53 nevertheless decided to issue some recommendations for security improvements. In particular, Cure53 identified that the URI field, *Resources* and links between *Resource*s can become very complex. Although Fysical deliberately chose and intended for the design to remain flexible, the auditors concurred that the flexibility of the smart contract signifies an observable security downgrade. It increases the risk of deception and may even enable attacks on the application layer through, for instance, malicious URIs. As a consequence, a well-designed app is necessary to aid brokers in understanding these highly complex links and build the necessary trust. Because the application was not in scope of this audit, it is impossible to make determinations about its soundness. Detailed inspection of the app is advised as security properties depend on both the - already proven robust - smart contract *and* the connected application as well.

More specific points for consideration stemming from the audit concerned compliance with the *EC20* and using the balance mapping for *escrow.* In this realm, Cure53 firmly believes that overloading the namespace here is not a good solution. A trade-off between the flexibility in favor of the lower levels of complexity would help Fysical to create a stronger application interfacing with the contract. Finally, the biggest concern from the Cure53's side is linked to the integrity of the off-chain *Resources* referenced by URIs. Cure53 recommends to strengthen this design by including both the file seller and the buyer through a protocol based on HKDF. To conclude, Fysical smart contract project is secure and devised in a way that protects its users. In order to make the security of the project even better, Cure53 advises that additional solutions and alterations delineated above are taken into consideration.

Cure53 would like to thank Justin Mann John Foley and Rich Garfield from the Fysical Technologies Pte. Ltd. team for their excellent project coordination, support and assistance, both before and during this assignment.