Rapport de travail préparatoire au mémoire de Master

Thomas Suau

1er juin 2024

Résumé

La forme de ce rapport a beaucoup été amenée à changer et j'ai réécris plusieurs fois ce travail que je préfère vous proposer en français. Une version en anglais a été réalisée initialement dont je reporte le principal travail sur le script Bitcoin et les transactions en annexe 4.
L'objectif principal de mon Mémoire est de tenter un questionnement et une recherche académique sur les protocoles basés sur Bitcoin utilisant l'enveloppe `OP_FALSE OP_IF`*. Ce travail préparatoire à pour but de donner un premier aperçu de l'état de l'art à ce sujet et proposer une direction pour une étude systématique de ces protocoles et de leurs applications. Étant des technologies en constante évolution ce travail est fait dans le contexte du mai 2024 où la dernière bip proposée est la bip 389.*

Table des matières

Préliminaires

Ce travail se place dans le cadre d'une étude sur la technologie Blockchain spécialisée sur Bitcoin. Pr. Etienne Riviere à réalisé un cours sur le script Bitcoin sur lequel je me suis appuyé lors de la première version de ce mémoire : preparatory work for master thesis, en partie joint en annexe.
Ce travail s'adresse aux chercheurs en Blockchain ayant effectué des premiers travaux sur Bitcoin. Dans cette version en français, je suppose connu les mécanismes de transactions, d'utxo et les principaux `OP_CODE` du script Bitcoin.
Les termes techniques en anglais sont donnés sans traduction ainsi que les définitions et citations considérées.

1 Introduction

On appelle enveloppe tout script Bitcoin cette suite d'`OP_CODE` à la fin des script de vérifications : `OP_FALSE OP_IF OP_PUSH [...] OP_ENDIF` détaillé en figure 1. On remarquera que cela permet d'inscrire de la donnée sur Bitcoin mais qu'il ne s'agit pas de l'unique méthode. L'utilisation de l'`OP_RETURN` [4] ou plus récemment du protocole BTC Stamps [21] permettent également d'inscrire et de manipuler de la donnée sur Bitcoin mais ce n'est pas l'objet de ce travail. De plus, le travail de Robin Linus [19] BitVM ne sera pas abordé, bien qu'il apporte un cadre effectif de calcul sur Bitcoin. On se concentrera sur les deux protocoles Ordinals [15] et Atomicals [9].

```
OP_FALSE
OP_IF
  OP_PUSH "ord"
  OP_PUSH 1
  OP_PUSH "text/plain;charset=utf-8"
  OP_PUSH 0
  OP_PUSH "Hello, world!"
OP_ENDIF
```

```
OP_FALSE
OP_IF
 0x0461746F6D // Push "atom" 4 bytes
 <Operation>  // Followed by a single push to denote the operation type
 <Payload>    // Payload (CBOR encoded) for the operation
OP_ENDIF
```

FIGURE 1 – Enveloppe Ordinals (gauche) Enveloppe Atomicals (right)

Ces protocoles utilisent du Script Bitcoin [13] et le modèle UTXO [20] (Unspent Transaction Output) de Bitcoin.
Arrivée très récemment, cela peut changer l'utilisation faites de la blockchain Bitcoin et éventuellement exécuter des actions auparavant limitée à Ethereum [26]. Les notions de virtualisation, de machine virtuelle et d'*account based model*[1] émergent en utilisant cette notion d'enveloppe.

Les protocoles Ordinals et Atomicals proposent deux approches très différentes de la notion d'objets sur Bitcoin. La section 2, reprend les concepts associés à ces deux protocoles et la section 3 propose un ensemble de questionnements, de pistes de résolutions ainsi que des propositions de tests sur ces différents objets.

1. *Account based model* [18] fait référence aux blockchains n'utilisant pas d'UTXO mais plutôt des comptes individuel.

2 État de l'art

Comme vu précédemment sur Atomicals nous avons :

```
OP_FALSE
OP_IF
        hex('atom')
        <OP>: Operation in the Protocol
        <Payload>: CBOR
OP_ENDIF
```

Payload : The part of transmitted data that is the actual intended message.

CBOR : Concise Binary Object Representation "Data format whose design goals include the possibility of extremely small code size, fairly small message size and extensibility without the need for version negotiation. These design goals make it different from earlier binary serialization such as ASN. 1 and message pack." [6]

BLOB : Binary Large OBject Used for different purposes. One use case is database large objects. "[Can] refer to a group of connected pixels in a binary image" [25] "BLOB extraction and the latter as BLOB classification"

Une question qui apparait est : y a-t'il une conversion équivalente entre CBOR et BLOB ?

Objet d'une db : une ou plusieurs lignes dans une ou plusieurs tables (ensemble de lignes et de colonnes) ayant une propriété commune (Car 'select'). Les relations au sein d'une db apporte la complexité qui se distingue d'un fichier plat (fichier texte non-relationnel) comme json.
Peut-on extraire un modèle relationnel d'un fichier json ?
Un fichier json ne requièrent aucun autre prérequis qu'être de la forme 2.

```
{
}
```

On remarque qu'Ethereum Mainnet [2] utilise le format BLOB pour communiquer l'évolution de l'état machine (EVM).
Plutôt utilisé pour gérer les objets de base de données rel.

Le protocole Atomicals propose des standards json définissant et permettant l'interaction avec les atoms. Un atom est un objet du protocole : Realm, Container, NFT, FT (possibles évolutions) On peut interagir avec ces objets via les Opérations.
Les opérations intégrées et actuellement utilisées sont : atomicals-js/<op>
"We can define any kind of objects but if the operation is not implemented in one client it will execute nothing."

Conversion de base de données relationnelle en document non relationnel sans perte de la configuration des données relationnelles. [24]

2. Discussions Pr. Etienne Riviere.

3 Objectifs

Une des premières interrogations pouvant avoir un impact conséquent dans la recherche est : Est-il possible d'exécuter du code contenu dans des BLOB (Binary Large OBject) ?
Cela permettrait d'exécuter une forme de *Smart Contracts*, actuellement propre à Ethereum.

Est-il possible d'utiliser des objets issus d'un de ces protocoles afin de *bridger* en Layer 1 (nativement) des actifs présent sur des blockchains UTXO-based ?
Cela rentre dans le cadre des recherches faites par Chainlink sur le protocole CCIP (Cross-Chain Interoperability Protocol) [8]. On pourrait étudier les méthodes utilisées par AtomicSwap [3] pour étudier la faisabilité et la sécurité de potentielles solutions.

Peut-on créer une classification de ces protocoles ?
Cela afin d'ouvrir les possibilités de recherches sur les protocoles basé sur l'enveloppe

4 Annexe

Initial work for the preparatory work for Master Thesis'.

4.1 Bitcoin transaction

Before to dive into Bitcoin script usage for Lightning Network (LN) [16] and messaging protocols a quick overview about Bitcoin transaction is necessary.

There is a complete tutorial about Bitcoin transaction by chaincodelabs [7]. This GitHub repository is explaining in details how a Bitcoin transaction is built and signed.

I just want to highlight some important points about Bitcoin transaction. Everything sent on the Bitcoin network is a transaction. Every transactions are sent into hexadecimal format, interpreted by the network as a combination of operations, addresses, values and inert data.
We can highlight that Bitcoin is an UTXO-based model which stands for Unspent Transaction Output. Every transactions are built from such unspent output.

FIGURE 2 – Hexadecimal of the first bitcoin transaction

Build an algorithm to colour transactions should be a first step for the research process about Bitcoin script usage. Indeed, it will allow to better understand the transaction structure and how to manipulate it.

To analyse a transaction at the lowest level we need to decode the hexadecimal format. The first transaction ever made on the Bitcoin network is the one from Satoshi Nakamoto Hal Finney. The hexadecimal format of this transaction is shown in the figure 2. In this transaction scripts are respectively in green and in orange. The green script is the scriptSig and the orange script is the scriptPubKey. We detail further this two kind of script.

4.2 Bitcoin script

Bitcoin script is a stack-based language. An UTXO has a lock script and transaction has an unlock script.
The most common type of transactions called standard transaction are payment "From Alice to Bob", we called them Pay to-Public Key Hash (P2PKH). Such transactions contain the scriptSig that unlocks the funds and the scriptPubKey that locks the output.
There are other type of transactions called non-standard transactions [5]. As we want to study protocols which are using Bitcoin Script we would focus on non-standard transactions built with custom scriptPubKey.
Validating a transaction requires executing both script and the Stack at the end must be empty. The Bitcoin script is composed of more than +200 operations called `OP_CODES`. The most used operations are :

— `OP_DUP` : Which are duplicate data in the current stack ;

— `OP_HASH160` : Make a double SHA256 ;

— `OP_EQUALVERIFY` : Verify and remove stack if data are correct ;

— `OP_CHECKSIG` : Check signature of address owner ;

A very basic scriptPubKey is the following :

```
OP_DUP OP_HASH160 <PubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
```

This script is used to send funds to a Bitcoin address (P2PKH) [5]. The scriptPubKey is composed of 5 operations. The first operation is `OP_DUP` which duplicates the top stack element. The second operation is `OP_HASH160` which hashes twice the top stack element : first by SHA-256 and then with RIPEMD-160. The third operation is `OP_EQUALVERIFY` which compares the top two stack elements. The fourth operation is `OP_CHECKSIG` which checks the signature.

In order to simplify usage of Bitcoin Script, Pieter Wuille, Andrew Poelstra, and Sanket Kanjalkar at Blockstream Research have developed a new language called Miniscript [29]. Miniscript is a subset of Bitcoin Script. It is a high-level language that allows to write Bitcoin Script in a more readable way. Miniscript is a very powerful

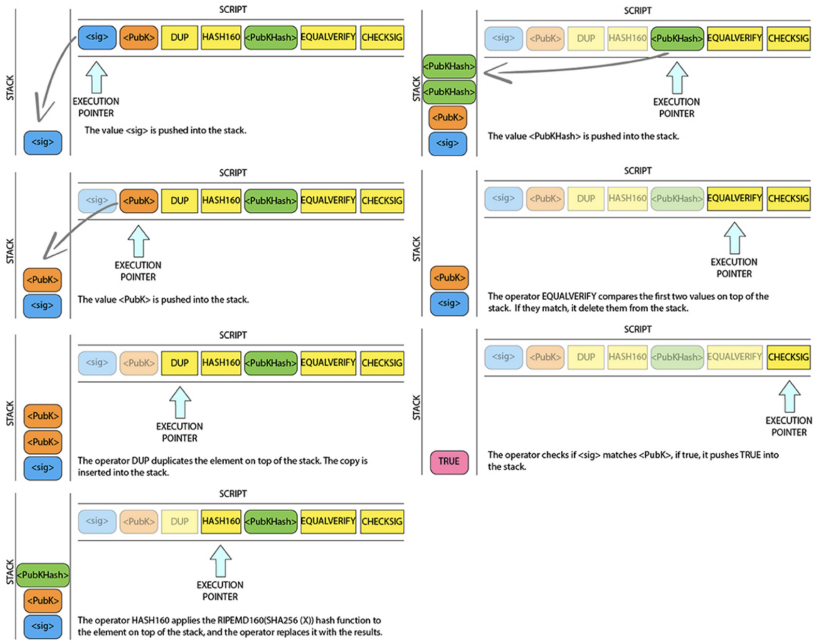

FIGURE 3 – Execution by the Stack [5]

tool to write Bitcoin Script.

One `OP_CODE` very used to define arbitrary protocols on Bitcoin is : `OP_RETURN`. Introduced in Bitcoin core as a standard operation in 2014 [4], it allows user to push arbitrary data with a length of 83 bytes maximum. Data pushed by user is evaluated to false by the Bitcoin protocol.
In Bartoletti and Pompianu [4] p.7, we have a tab to summarise some usage of `OP_RETURN` in Bitcoin transactions. We can notice that the most used protocol was the `Colu` (CC) protocol. This protocol was used to create colored coins on the Bitcoin network. Colu was apparently a first initiative from Bankbox to allow Central Bank to issue digital currencies, where Colored Coins was a first experimentation on Bitcoin blockchain Colu Open-Sources Protocol to Help Central Banks Issue Digital Currencies.

4.3 State of the art

Blockchain is today a research topic for some Universities and Research Center in the world. But we can notice that a lot of Bitcoin research and Blockchain research articles are more focused on the price than on the technology itself.

The Lightning Network (LN) is a second layer protocol that allows to make instant and low fees transactions. This protocol is more complex and involves a lot of Bitcoin Script. The protocol is based on the concept of payment channels. The payment

channels are a way to make off-chain transactions. They are leveraging unsigned Funding Bitcoin transactions (PSBT) and Commitment transactions, with some specific `OP_CODE`.
The LN produced a lot of academic research over the last years. In an attack perspective [1, 2, 10], to study network scalability [12] or to study the protocol itself [11].

Since many years, the concept of Vault and permissionned addresses is very discussed into the developper community. It's well know as Bitcoin *Covenants* [22, 23].
The problem can be described as followed : we want to enforced on the protocol side some addresses to make specific actions only. For example a company would want to enable its employee to use the bitcoin (BTC) of the company without allowing them to spend the Bitcoin for themselves. They may want to enforce on the protocol side rules for address to spend the BTC. Other uses cases can be to enforce an address to spend funds only on specific whitelisted address and avoid the loss of BTC by malicious attackers.
The concept of Vault is pretty similar and wants for addresses to play the role of locker for a certain given time. Those approach can be very useful for professional and securities purpose. In another hand, it will create permissionned addresses and some actors are strongly against this upgrade. The exact implementation is still very discussed by Bitcoin Core developers, it's an active research field.
Some new computing power should be available with such upgrades and our work could help the research to better understand improvements and limitations of such implementations.

More recently, with Taprscipt [28] new kind of covenants where purposed as the use of `OP_CAT` [14] or `OP_CHECKTEMPLATEVERIFY` [27]. The Bitcoin Covenants are a very discussed topic by Bitcoin core developper and it should be considered as a research work about Bitcoin script [17].

Références

[1] Ali Abdullah and A. M. Mutawa. An invitation model protocol (imp) for the bitcoin asymmetric lightning network. *Computer Science and Symmetry/Asymmetry : Feature Papers*, 2023.

[2] Riard Antoine, Naumenko Gleb, and Reuben Youngblom. Time-dilation attacks on the lightning network. *Computer Science and Symmetry/Asymmetry : Feature Papers*, 2021.

[3] Deshpande Apoorvaa and Herlihy Maurice. Privacy-preserving cross-chain atomic swap. *International Conference on Financial Cryptography and Data Security*, pages 540–549, 2020.

[4] Massimo Bartoletti and Livio Pompianu. An analysis of bitcoin op_return metadata. *Financial Cryptography and Data Security*, 2017.

[5] Stefano Bistarelli, Ivan Mercanti, and Francesco Santini. An analysis of non-standard bitcoin transactions. In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pages 93–96, 2018.

[6] C. Bormann and P. Hoffman. Rfc 8949 concise binary object (cbor). *Internet Engineering Task Force (IETF)*, 2020.

[7] Chaincodelabs. Bitcoin transaction tutorial. Bitcoin Transaction Tutorial, 2022.

[8] Chainlink. Ccip architecture. Official Documentation, 2024.

[9] Atom community. Atomicals documentation. Community Guide, 2024.

[10] Rohrer Elias, Malliaris Julian, and Florian Tschorsch. Discharged payment channels : Quantifying the lightning network's resilience to topology-based attacks. *IEEE Security & Privacy on the Blockchain 2019*, 2019.

[11] Rohrer Elias, Malliaris Julian, and Florian Tschorsch. On the impact of attachment strategies for payment channel networks. *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 2021.

[12] Yuwei Guo, Jinfeng Tong, and Chen Feng. A measurement study of bitcoin lightning network. *2020 IEEE International Conference on Blockchain (Blockchain)*, 2020.

[13] Brakmić Harris. *Bitcoin Script. in Bitcoin and Lightning Network on Raspberry Pi : Running Nodes on Pi3, Pi4 and Pi Zero.* Apress, Berkeley, CA, November 2019.

[14] Ethan Heilman and Armin Sabouri. Op_cat. BIP-OPCAT.

[15] Open Ordinals Institute. Ordinals handbook. Official Documentation, January 2024.

[16] Dryja Joseph, Poon ; Thaddeus. The bitcoin lightning network : Scalable off-chain instant payments. January 2016.

[17] Swambo K., Spencer Hommel, and alt. Bitcoin covenants : Three ways to control the future. *arXiv*, 2020.

[18] Brünjes Lars and Gabbay Murdoch J. Utxo- vs account-based smart contract blockchain programming paradigms. In *Leveraging Applications of Formal Methods, Verification and Validation : Applications*, volume 12478, pages 73–88. Spinger, Cham, October 2020.

[19] Robin Linus. Bitvm : Compute anything on bitcoin. Whitepaper, December 12 2023.

[20] Chakravaty Manuel, M.T., Chapman James, Wadler Philip, and alt. The extended utxo model. In *Financial Cryptography and Data Security*, pages 525–539. Springer, Cham, August 2020.

[21] Mikeinspace. Btc stamps protocol. Official Documentation, February 2023.

[22] Malte Möser, Ittay Eyal, and Emin Gün Sirer. Bitcoin covenants. In *Financial Cryptography and Data Security*. Springer Berlin Heidelberg, 2016.

[23] Russel O'Connor and Marta Piekarska. Enhancing bitcoin transactions with covenants. 2017.

[24] K. Revathi, T. Tamilselvi, B. Dhanwanth, and alt. Auto json : An automatic transformation model for converting relationnal database to non-relational documents. *researchgate*, 2023.

[25] Moeslund Thomas B. and Moeslund Thomas B. Blob analysis. *Introduction to video and image processing Build Real Systems and Applications*, pages 103–115, 2012.

[26] Butherin Vitalik. Ethereum whitepaper. Official link, 2014.

[27] Pieter Wuille, Jonas Nick, and Anthony Towns. Taproot : Segwit version 1 spending rules. bip-0341, January 2020.

[28] Pieter Wuille, Jonas Nick, and Anthony Towns. Validation of taproot scripts. bip-0342, January 2020.

[29] Pieter Wuille, Andrew Poelstra, and Sanket Kanjalkar. Bitcoin miniscript. Miniscript, 2019.