

# Disciplina: Data Disclosure Protocol

TeachMePlease, <https://teachmeplease.com>

Serokell, <https://serokell.io>

Version 0.1  
December 8, 2017

## Abstract

This document describes the two-party protocol of fair data trade used in the Disciplina blockchain platform.

## 1 Notation

Throughout this document we use the following notations:

Notation	Description
$\mathbf{A}$	Party that takes part in the protocol
$H(m)$	Result of applying a collision-resistant hash-function $H$ to a message $m$
$\text{mtree}(a)$	Merkle tree of the data array $a$
$\text{root}(M)$	Root element of the Merkle tree $M$
$\text{path}(e, M)$	Path of the element $e$ in the Merkle tree $M$
$k$	Symmetric key
$pk_{\mathbf{A}}, sk_{\mathbf{A}}$	Public and secret keys of a party $\mathbf{A}$
$E_k(m)$	Symmetric encryption with the key $k$
$E_{\mathbf{A}}(m)$	Asymmetric encryption with the key $pk_{\mathbf{A}}$ *
$\text{Sig}_{\mathbf{A}}(m)$	Tuple $(\mathbf{A}, m, \text{sig}(sk_{\mathbf{A}}, H(m)))$ , where $\text{sig}$ is a digital signature algorithm*
$\text{sizeof}(m)$	Size of $m$ in bytes
$\oplus$	Binary string concatenation

## 2 Prerequisites

Suppose the seller  $\mathbf{S}$  has some data  $D$ . Before the deal  $\mathbf{S}$  ought to perform some preparation steps.  $\mathbf{S}$  should:

1. Divide  $D$  into  $N$  chunks of size no more than 1 KiB:

$$D = \bigoplus_{i=1}^N d_i, \quad \text{sizeof}(d_i) \leq 1 \text{ KiB} \quad (1)$$

2. Generate a symmetric key  $k$
3. Encrypt each  $d_i$  with  $k$  and make an array of encrypted chunks:

$$D_{\mathbf{A}} = \{E_k(d_1), E_k(d_2), \dots, E_k(d_N)\} \quad (2)$$

---

\*The particular keys  $pk_{\mathbf{A}}$  and  $sk_{\mathbf{A}}$  belonging to the party  $\mathbf{A}$  are generally deducible from the context

4. Compute a Merkle root of the encrypted chunks:

$$R = \text{root}(\text{mtree}(D_{\mathbf{B}})) \quad (3)$$

On this stage  $R$  is a public knowledge, while  $k, D_{\mathbf{B}}$  and all of the  $d_i$  are kept hidden.

### 3 Protocol description

The protocol fairness is guaranteed by a contract on the public chain. The contract is able to hold money and is stateful: it is capable of storing a log  $L$  with data. All the data that parties send to the contract are appended to  $L$ .

1. The buyer generates a new keypair  $(pk_{\mathbf{B}}, sk_{\mathbf{B}})$ , creates the contract and sends the money to the contract address. Along with the money,  $\mathbf{B}$  sends the public key  $pk_{\mathbf{B}}$  of the newly generated keypair.
2. If  $\mathbf{S}$  agrees to proceed, she also sends a predefined amount of money to the contract address.
3.  $\mathbf{S}$  transfers the encrypted data chunks  $D_{\mathbf{B}}$  to the buyer.  $\mathbf{B}$  computes the Merkle root  $R'$  of the received data  $D_{\mathbf{B}}'$ :

$$R' = \text{root}(\text{mtree}(D_{\mathbf{B}}')) \quad (4)$$

4.  $\mathbf{B}$  makes a transaction with a receipt  $\text{Sig}_{\mathbf{B}}(R')$  to the contract address.
5.  $\mathbf{S}$  sends  $\text{Sig}_{\mathbf{S}}(\{\text{E}_{\mathbf{B}}(k), R\})$  to the contract. The contract accepts it iff  $R = R'$  (this implies that  $D_{\mathbf{B}} = D_{\mathbf{B}}'$ ).
6.  $\mathbf{B}$  decyphers and checks the received data. In case some data chunk  $e_i \in D_{\mathbf{B}}$  is invalid,  $\mathbf{B}$  sends a transaction with  $\{sk_{\mathbf{B}}, e_i, \text{path}(e_i, \text{mtree}(D_{\mathbf{B}}))\}$  to the contract. By doing so,  $\mathbf{B}$  reveals the data chunk  $d_i$  corresponding to the encrypted chunk  $e_i$ . She also shares proof that  $e_i$  was indeed part of Mekle tree with root  $R$ . The contract checks the validity of  $d_i$  and decides whether  $\mathbf{B}$  has rightfully accused  $\mathbf{S}$  of cheating.

The on-chain communications of the parties (steps 2, 4, 5, 6) are bounded by a time frame  $\tau$ . In order for the transaction to be valid, the time  $t$  passed since the previous step have to be less than or equal to  $\tau$ . In case  $t > \tau$  the communication between the parties is considered over, and one of the protocol exit points (Sec. 4) is automatically triggered.

### 4 Protocol exit points

To decide on whether the communication is over, the protocol utilizes some timeout  $\tau$  (e.g. 1 hour) which bounds all communications that should happen between  $\mathbf{B}$  and  $\mathbf{S}$ .

**Before step 4 :**  $\mathbf{B}, \mathbf{S}$  get their money back

Interpretation:  $\mathbf{S}$  wasn't able to transfer the data to  $\mathbf{B}$ .

**After step 4:**  $\mathbf{B}, \mathbf{S}$  get their money back

Interpretation:  $\mathbf{B}$  received the data, but  $\mathbf{S}$  wasn't able to share the key for it.

**After step 5:**  $\mathbf{S}$  gets all money

Interpretation:  $\mathbf{S}$  correctly shared data to  $\mathbf{B}$

**After step 6:**  $\mathbf{B}$  gets all money

Interpretation:  $\mathbf{B}$  proofs  $\mathbf{S}$  cheated,  $\mathbf{S}$  loses all money