SAFINA

Описание системы

2023

office@safina.pro

Содержание

SAFINA

Ноу-хау системы Safina — это быстрая и безопасная технология обращения к блокчейну, расширяющая возможности применения смарт-контрактов, позволяющая создавать новые продукты на блокчейне и управлять операциями в цифровых активах. С Safina владелец активов всегда сможет управлять ими напрямую, а также открываются возможности надстройки большого количества бизнес-процессов, позволяющих соединять блокчейн с традиционными технологиями для расширения бизнеса и повышения эффективности процессов.

Описание

Из чего состоит система Safina:

1. Личный кабинет
   Интерфейс централизует управление всеми кошельками пользователя и подписанием транзакций для тех кошельков, где пользователь указан участником. Благодаря этому пользователь может контролировать подписание созданных им транзакций и получать отчеты и статистические данные о переводах со своих кошельков.

2. Криптокошельки
   Безопасные мультиподписные кошельки для криптоактивов с надежным хранением приватного ключа. Подход основан на протоколе MPC и технологии мультиподписи: для отправки транзакции в сеть требуется единое подтверждение от кворума участников, которое достигается тем, что участник подписывается своей подписью. Процессы создания кошелька, генерации ключей и проверки подписей построены так, что они изолированы от внешних подключений. Клиенты могут использовать такие кошельки для целого ряда операций, например, торговля, холодное хранение, NFT, смарт-контракты и пр.

   Криптокошельки относятся и «привязываются» только к определенной учетной записи (владельца кошелька) без права передачи.

3. Подписание

Подписание транзакций хоть и по факту является частью интерфейса личного кабинета, доступно незарегистрированным в личном кабинете участникам кошелька. В зависимости от вида подписи, выбранной владельцем кошелька, участник вводит в специальную форму либо сгенерированный код, либо подписывает форму своим публичным ключом. Владелец кошелька может видеть прогресс подписания и причины отказа для подписания. Так как участники подписывают транзакцию в личном кабинете, сбор подписей не требует затрат на блокчейн-транзакции и ожидания их валидации.

Условно можно выделить два слоя защиты средств в Safina:

1. Слой подписания

   Как говорилось ранее, подписание основано на протоколе MPC и технологии мультиподписи. После получения нужного количества подписей и их проверки транзакция подписывается единым закрытым ключом кошелька и отправляется в сеть.

   Процедура проверки подписей включает в себя два этапа и проходит через два компонента Safina.

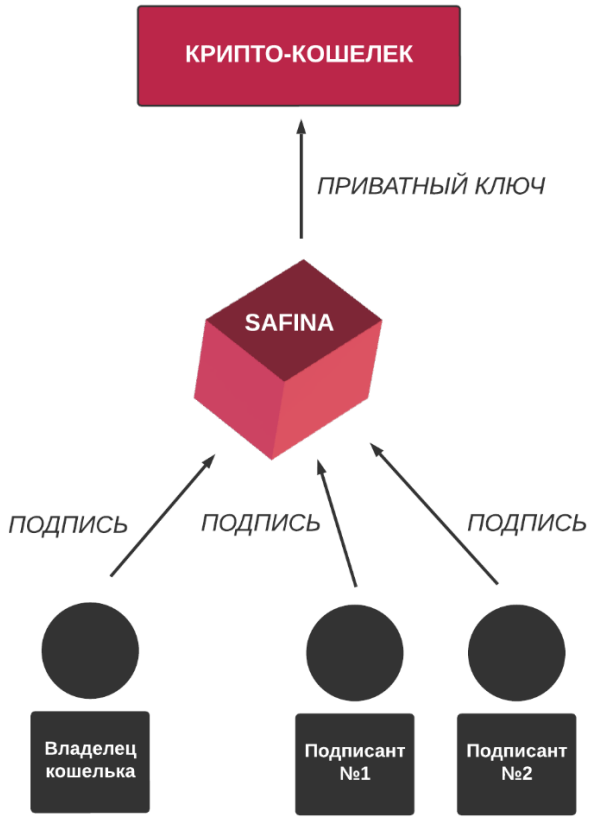


Рис. 1 - Схема подписания

2. Слой хранения ключей
   Сгенерированные ключи хранятся в BlackBox – сервере, расположенном в DMZ. Информация о ключах не может быть получена хакерами, сотрудниками Safina или подписантами, так как данные зашифрованы, а подключение изолировано.

Защита в Safina

Пользователи могут подключить 2FA Safina для входа в личный кабинет. 2FA Safina – это приложение для двухэтапной аутентификации с использованием одноразового пароля. Пароли генерируются с помощью Time-based One-time Password Algorithm (TOTP) и HMAC-based One-time Password Algorithm (HOTP). Сервис реализует алгоритмы, указанные в RFC 6238 и RFC 4226. Отличием от имеющихся сервисов по двухфакторной аутентификации является то, что одноразовые пароли генерируются на устройстве пользователя и сервере сервиса независимо друг от друга, соответственно, исключается возможность перехвата одноразового пароля третьей стороной.

Одноразовый пароль представляет из себя 6-значное число, которое пользователь должен предоставить в дополнение к имени пользователя и пароля, чтобы войти в личный кабинет Safina или службы иных сервисов.

Компоненты Safina

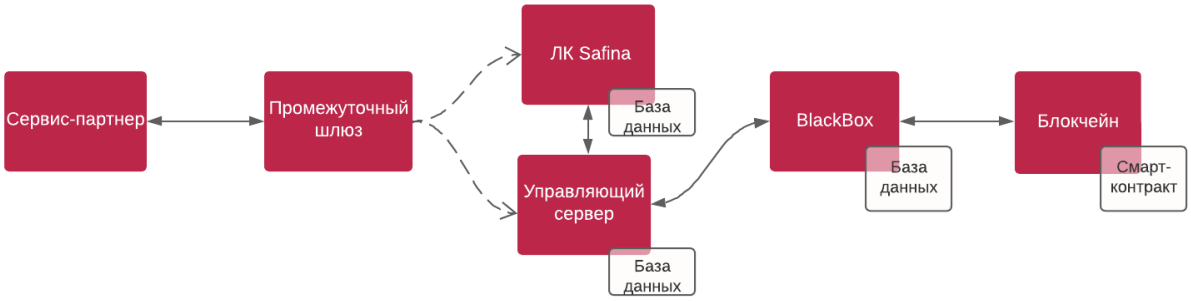
Концептуальная модель в упрощенном виде содержит направление потока информации при подключении системы Safina.

Партнерский-сервис выступает основным пользовательским интерфейсом, который передает запросы и заявки от клиентов промежуточному шлюзу.

Промежуточный шлюз это несколько компонентов системы Safina, между которыми проводится сортировка и распределение данных из заявок клиентов. Это сделано, чтобы отделить информацию, необходимую для проведения блокчейн-операций, от информации, необходимой для внешних бизнес-процессов.

Управляющий сервер BlackBox (далее BB) получает только данные, которые ему нужны для создания кошелька или отправки транзакции

перевода средств в сеть. После выполнения заявки BB уведомляет об этом второй управляющий сервер и запускается обратная цепочка передачи данных между компонентами системы для уведомления клиента о завершении операции.



**ЛК – личный кабинет*

Рис. 2 - Концептуальная модель Safina

BlackBox

Создание BlackBox

- Создание образа на чистой машине[1];
- Перенос образа на рабочую площадку;
- id Domino сервера [2] создаётся на этой же чистой машине;
- Администрирование на стадии начальной настройки Domino осуществляется с соседней чистой машины.

Свойства

- защита от несанкционированного физического доступа к информации;
- привязка к хосту, на котором запущена виртуальная машина;
- внешние исходящие сетевые правила позволяют связь только с 3мя фиксированными нодами (HTTPS), 2мя управляющими серверами[3] (шифрованный трафик между Domino серверами);
- внешние входящие сетевые правила позволяют только SSH подключение к консоли;

---

[1] Чистая машина – без связи с глобальной и офисной сетями, установленная с проверенного образа с подтверждённой контрольной суммой.
[2] HCL Domino Сервер – программное обеспечение компании HCL, серверная часть программного комплекса HCL Notes.
[3] Управляющий сервер – программный компонент системы Safina на базе HCL Notes, выполняющий ряд операций.

- консоль авторизует по Open SSL сертификатам;
- внутренний firewall повторяет внешние сетевые правила.

BB имеет доступ только к трем фиксированным нодам, которые находятся на существующих 2х серверах и собраны с базового исходника нод. Любой человек может настроить ноду под себя в силу общедоступности ее кода, и то, что BB обращается только к ноде Safina, гарантирует, что он не подключится к ноде, которая отличается от основной. Так Safina контролирует сервер, на котором находятся закрытые ключи, и все программы на этом сервере. Несмотря на то, что публичные ноды могут подключиться к одной из трех фиксированных нод, являющейся внешней, на ней ничего не хранится, потому при ее взломе утечки данных не произойдет.

API

Начало работы

Базовым идентификатором пользователя со стороны API является хэш публичного ключа из пары ключей, полученных с помощью алгоритма эллиптических кривых (Elliptic Curve SECP256k1). Хэш публичного ключа это так называемый "адрес", поле x-app-ec-from в запросе, который выступает в роли электронной подписи.

Обращение к API

Для обращения к API обязательно требуется наличие трёх полей электронной подписи запроса (x-app-ec-sign-r, x-app-ec-sign-s, x-app-ec-sign-v) в заголовках запросов. Подписывается URL запроса и поля формы (если это POST запрос).

После вызова функции API, создающей кошелёк, все последующие команды для работы с этим кошельком требуют электронной подписи именно теми ключами, которые были использованы для подписи команды на создание кошелька.

Безопасность

Для одной пары ключей (одного условного пользователя) можно создавать несколько кошельков, каждый кошелёк получает свой уникальный код (получение списка кошельков).

Более правильным является подход, когда для каждого конечного пользователя создается отдельная пара ключей.

Можно использовать подход, когда пара ключей создается сразу для внешней системы, внутри которой уже разделяются кошельки пользователей. В этом случае кошельками всех пользователей можно управлять с помощью этой пары ключей.

Важно помнить, что пара ключей открывает возможность тратить средства, находящиеся на кошельках, которые созданы с помощью этой пары. Поэтому необходимо обеспечить безопасное хранение этой пары.

Существует несколько общедоступных библиотек, которые реализуют функции по созданию ключей, вычислению подписи и проверке подписи.

Для удобного восстановления пары ключей можно воспользоваться механизмом их генерации с использованием SID фразы. В данном случае в надёжном месте сохраняется не сам ключ, а этот набор слов и при восстановлении ключей уменьшается вероятность опечатки. Обычно эта SID фраза записывается человеком от руки на бумагу или сохраняется на физическом носителе, которые впоследствии хранятся в защищённом от уничтожения и несанкционированного доступа месте.

Важно понимать, что данные ключи не используются самой системой для работы с блокчейн сетями. Для каждого создаваемого кошелька система генерирует собственные ключи и уже их использует для работы с блокчейн сетями.

Примерный порядок работы с API

Предварительно необходимо получить справочник блокчейн сетей, чтобы указывать ID сети при создании кошелька. Также доступен справочник поддерживаемых системой токенов с параметрами комиссий системы. Следующий шаг - создание кошелька. При создании кошелька, кроме ID сети блокчейна, нужно указать параметры "мультиподписи": минимальное количество подписантов и описание самих подписантов.

Каждый подписант имеет возможность подписывать условными подписями (ссылкой-кодом, полученным на его email, кодом из смс

уведомления, отправленного на телефон). Дополнительно указывается, нужно ли использовать все указанные у подписанта способы или только один из них.

Время создания кошелька может составлять 10 минут с момента получения последней необходимой подписи.

Для получения информации по созданным кошелькам нужно выполнить операцию получения списка кошельков. Каждый из кошельков в списке будет содержать собственный адрес

При создании кошелька система также добавляет пользователю базовый токен блокчейн сети кошелька с балансом "0". Периодически система проверяет баланс кошелька и обновляет баланс и список токенов которые содержаться на данном кошельке. Получить список токенов и балансы можно командой "Получение списка токенов пользователя с одного кошелька" либо "Получение списка всех токенов пользователя"

Общие для всех запросов параметры

Basic URL

```
https://my.h2k.me/ec/
```

Электронная подпись:

```
HEADRES:
  x-app-ec-from
  x-app-ec-sign-r
  x-app-ec-sign-s
  x-app-ec-sign-v
```

x-app-ec-from : адрес подписавшего (HEX) с префиксом 0x

```
0xa75db3c448bb62e208e4babf500925427837e464
```

x-app-ec-sign-r : R-компонента подписи (HEX) с префиксом 0x

```
0xdb07295a5f780159d51c4872a104e6e486428942db38ea3b7d91433c38658c0b
```

x-app-ec-sign-s : S-компонента подписи (HEX) с префиксом 0x

```
0x3a64d736044d63cff3713b85aa2a2fad902c080c2b64acfcdbee7ce9e20cae0e
```

x-app-ec-sign-v : V-компонента подписи (HEX) с префиксом 0x, флаг восстановления адреса

```
0x1b
```

Подписываемая строка

- "{}" в случае отсутствия полей формы или запроса GET ({});
- json с полями формы в случае запроса POST, PATCH, DELETE.

Пример json

```
{"slist":{"0":{"type":"all","email":"e@mail.com","ecaddres":"0xa12343c448bb62e208e4babf500925427837e464"},"1":{"type":"any","email":"e@mail.ru","sms":"+77777777777","ecaddres":"0x1234567448bb62e208e4babf500925427837e464"}},"network":"1000","info":"My first wallet"}
```

Никаких пробелов в структуре json не допускается, иначе ОШИБКА ПОДПИСИ.

Ответ в случае ошибки

```
{"ERROR":"Ошибка 12313", "LINE":"125"}
```

Описание методов

Получение справочника сетей

Действующие сети

```
GET https://my.h2k.me/ec/netlist/1
```

Отключенные сети

```
GET https://my.h2k.me/ec/netlist/0
```

Ответ

```
[
{"network_id":1000,"network_name":"Bitcoin (BTC)",
"link":"https://www.bitcoin.com",
"address_explorer":"https://www.blockchain.com/btc/address/",
"tx_explorer":"https://www.blockchain.com/btc/tx/",
"block_explorer":"https://www.blockchain.com/btc/block/","info":null,"status":1},

{"network_id":1010,"network_name":"Bitcoin Test (BTC)",
```

"link":"https://www.bitcoin.com",
"address_explorer":"https://www.blockchain.com/btc-testnet/address/",
"tx_explorer":"https://www.blockchain.com/btc-testnet/tx/",
"block_explorer":"https://www.blockchain.com/btc-testnet/block/","info":null,"status":1},

{"network_id":3000,"network_name":"Etherium (ETH)",
"link":"https://ethereum.org",
"address_explorer":"https://etherscan.io/address/",
"tx_explorer":"https://etherscan.io/tx/",
"block_explorer":"https://etherscan.io/block/","info":null,"status":1},

{"network_id":3010,"network_name":"ETH Ropsten Test (ETH)",
"link":"https://ethereum.org",
"address_explorer":"https://ropsten.etherscan.io/address/",
"tx_explorer":"https://ropsten.etherscan.io/tx/",
"block_explorer":"https://ropsten.etherscan.io/block/","info":null,"status":1},
...
]

Создание кошелька

Запрос

```
POST https://my.h2k.me/ec/newWallet
```

Тело запроса

```
{
  "slist": {
        "min_signs":"2",
        "0":{"type":"all", "email":"e@mail.com"},
        "1":{"type":"any", "email":"e@mail.ru", "sms":"+77777777777"},
        "2":{"type":"all", "ecaddress":"0xa75db3c448bb62e208e4babf500925427837e464"}
       }
  "network": "1000",
  "info": "Мой первый кошелёк"
}
```

Если структуры slist нет, то создастся ОБЫЧНЫЙ ОДНОПОДПИСНОЙ КОШЕЛЁК.
Наличие структуры slist означает запрос на МУЛЬТИПОДПИСНОЙ кошелёк. В ней содержатся данные подписантов:

- type = all значит, что для совершения транзакции необходимо подтверждение всеми способами, указанными в slist;
- type = any значит, что для совершения транзакции необходимо подтверждение любым одним способом из указаннымх в slist;

– min_signs – необязательный параметр, определяющий, какое минимальное количество подписей необходимо, чтобы транзакция была подтверждена, значение больше нуля. Если не указан, то подтвердить должны все подписанты;
– email, sms, ecaddress – необязательные параметры, при наличии элемента slist обязательно присутствие одного из них.

Ответ

```
{"myUNID":"BE96CC2F2455C63546258901003B170C"}
```

– myUNID – уникальное имя заявки на создание кошелька.

Получение списка кошельков

```
GET https://my.h2k.me/ec/wallets
```

Запрос GET https://my.h2k.me/ec/wallets_2 возвращает те же самые данные с разделителем ";" вместо ",".
В ответ на запрос придёт JsonArray, каждый элемент которого содержит данные о кошельке. Если кошельков нет, то вернётся просто "[]".


```
[
{"wallet_id":849,"network":5010,"wallet_type":1,"name":"945C6F4C54B3921F4625890300235114","info":"Test
Test","addr":"TXuQVGNiu38V5fcxCgawKucnL1MikefHB5","addr_info":null,"myUNID":"C4834FE85AD1B20E4625890300231CC8","tokenShortNames":"TRX,USDT"},
{"wallet_id":857,"network":5010,"wallet_type":1,"name":"786D9289D827EB5046258903002DE1BC","info":"Test
Test","addr":"TRx6xXChS5sXz3mpvLSNfKuL6w3PBdMZzL","addr_info":null,"myUNID":"F74E34049A12714446258903002DAB04","tokenShortNames":"TRX,USDT"},
{"wallet_id":858,"network":5010,"wallet_type":1,"name":"E256AAB7E8E0B0E646258903002DE217","info":"Test
Test","addr":"TTsoaqmR795wPT1ohVYSJFVXinHjwW6ex8","addr_info":null,"myUNID":"3A8FA1DFAC303D0C46258903002DCA23","tokenShortNames":"TRX,USDT"},
{"wallet_id":861,"network":5010,"wallet_type":1,"name":"0F12E99B06DAFF9E46258903002EAAAE","info":"Test
Test","addr":"TF1WzAQX2hkWnspEgwqyNaQoa3dqCuNT1Z","addr_info":null,"myUNID":"FB48C991F6A78B9A46258903002E6665","tokenShortNames":"TRX,USDT"},
{"wallet_id":862,"network":5010,"wallet_type":1,"name":"5852554B1764A7A946258903002EAB0F","info":"Test
Test","addr":"TY1crvAPB8Z1qpfWTkjjFqZJRztvD9ECoS","addr_info":null,"myUNID":"BB4C9B312887AA4D46258903002E7F6E","tokenShortNames":"TRX,USDT"},
{"wallet_id":863,"network":5010,"wallet_type":1,"name":"C63C1591E72687E1462589030034C9F4","info":"Test
```

Test","addr":"TS5sFz9mo3RS3z2kifw3q2i1ypLbXiCTsQ","addr_info":null,"myUNID":"FF297EC3C1B86300462589030034A4D5","tokenShortNames":"TRX,USDT"},
{"wallet_id":962,"network":5010,"name":"BE96CC2F2455C63546258901003B170C","info":"Test Test Test","addr":""}
]

– "wallet_type":1 значит, что кошелёк горячий.

Если запись кошелька имеет поле пустое addr = "", то она соответствует заявке на создание кошелька и выводится сокращённым набором полей

{"wallet_id":962,"network":5010,"name":"BE96CC2F2455C63546258901003B170C","info":"Test Test Test","addr":""}

– поле name соответствует уникальному номеру заявки на создание кошелька (myUNID) и после создания кошелька будет изменено на новое.

Получение списка всех токенов пользователя

Подписавшему запрос пользователю выдаются все токены со всех его кошельков.
Базовые токены сети с балансом 0 создаются в момент создания кошелька, производные токены появляются после того, как поступят на адрес кошелька.

```
GET https://my.h2k.me/ec/tokens
```

Ответ

[
{
  "id" : "357",
  "wallet_id" : "254",
  "network" : "3010",
  "token" : "TRX",
  "value" : "154,254"
  "decimals" " "6"
  "value_hex" : "0x931BAB0"
},
{
  "id" : "371",

```
  "wallet_id" : "259",
  "network" : "3010",
  "token" : "TRX",
  "value" : "15,0"
  "decimals" " "6"
  "value_hex" : "0xE4E1C0"
},
]
```

Получение справочника токенов

Подписавшему запрос пользователю отдаётся справочник токенов, поддерживаемых системой.

```
GET https://my.h2k.me/ec/tokensinfo
```

Ответ

```
[
 {"token":"1000:::BTC","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"1010:::BTC","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"3000:::ETH","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"3010:::ETH","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"3030:::ETH","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"3040:::ETH","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"3000:::USDT","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"3010:::USDT","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"3030:::USDT","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"3040:::USDT","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"5000:::TRX","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"5010:::TRX","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"5000:::USDT","c":"0.01","cMin":"0.001","cMax":"100"},
 {"token":"5010:::USDT","c":"0.01","cMin":"0.001","cMax":"100"}
]
```

- token – токен;
- c – множитель комиссии (например, 0.01 = 1%);
- cMin – минимальная сумма комиссии;
- cMax – максимальная сумма комиссии.

Получение списка токенов пользователя с одного кошелька

Подписавшему запрос пользователю выдаются токены с одного кошелька.

```
GET https://my.h2k.me/ec/wallet_tokens/:name
```

Пример

```
GET https://my.h2k.me/ec/wallet_tokens/4079BF13E4A8619846258903003556D5
```

Ответ

```
[
{
  "id" : "357",
  "wallet_id" : "254",
  "network" : "3010",
  "token" : "TRX",
  "value" : "154,254"
  "decimals" " "6"
  "value_hex" : "0x931BAB0"
},
{
  "id" : "371",
  "wallet_id" : "259",
  "network" : "3010",
  "token" : "TRX",
  "value" : "15,0"
  "decimals" " "6"
  "value_hex" : "0xE4E1C0"
},
]
```

Отправка транзакции

TODO: добавить ключ "instant":"". Если он есть, то транзакция не требует подтверждения пользователя после сбора подписей.

```
POST https://my.h2k.me/ec/tx
{
  "token":"5010:::TRX###945C6F4C54B3921F4625890300235114",
  "info":"Test Test",
  "value":"1,01",
  "toAddress":"TRx6xXChS5sXz3mpvLSNfKuL6w3PBdMZzL"
}
```

– value – разделитель дробной части всегда запятая

Ответ

```
{"tx_unid":"FA581EE3B5899573462589030071DF52"}
```

Получение списка транзакций

Подписавшему запрос пользователю отдаются все исходящие транзакции со всех его кошельков, инициированные в системе. Внешние транзакции не отображаются

```
GET https://my.h2k.me/ec/tx
```

Ответ


```
[
 {
  "id":470,
  "tx":null,
  "token":"5010:::TRX###9C51E0B331882BCF46258913005338CB",
  "token_name":"TRX",
  "to_addr":"TRx6xXChS5sXz3mpvLSNfKuL6w3PBdMZzL",
  "value":"1,005","value_hex":"0",
  "unid":"2E19FB324B3867C146258915006AC7F0",
  "init_ts":1670786865,
  "min_sign":null,
  "wait":[{"ecaddress":"0xA285990a1Ce696d770d578Cf4473d80e0228DF95"}],
  "signed":null
 },
 {
  "id":469,
  "tx":null,
  "token":"5010:::TRX###9C51E0B331882BCF46258913005338CB",
  "token_name":"TRX",
  "to_addr":"TRx6xXChS5sXz3mpvLSNfKuL6w3PBdMZzL",
  "value":"1,005","value_hex":"0",
  "unid":"C9D0C61AD75DF26F4625891500402B26",
  "init_ts":1670758938,
  "min_sign":null,
  "wait":[{}],
  "signed":[
          {"ecaddress":"0xA285990a1Ce696d770d578Cf4473d80e0228DF95",

"ecsign":"0xb1cb0ac4a67df3dbe4a018bb2f398d98024ca0be12c5c137f980814f8eaf339d0x0ac527b
42e5510646b9f8d4254913ba0c000d454bb2ebbc145cf731d037744950x1c"
          }
         ]
 },
 {
  "id":468,
  "tx":null,
  "token":"5010:::TRX###9C51E0B331882BCF46258913005338CB",
  "token_name":"TRX",
  "to_addr":"TRx6xXChS5sXz3mpvLSNfKuL6w3PBdMZzL",
  "value":"1,005","value_hex":"0",
```

```
  "unid":"D09BBF3B2A77A1B746258915003FB2EC",
  "init_ts":1670758630,
  "min_sign":null,
  "wait":null,
  "signed":null
 }
]
```


– tx – хэш транзакции; заполняется только если транзакция уже размещена в сети, до этого передаётся null.

Получение списка транзакций определённого токена

Подписавшему запрос пользователю выдаются все исходящие транзакции определённого токена в кошельке, инициированные внутри системы. Внешние транзакции не отображаются.

```
GET https://my.h2k.me/ec/tx:token
```

Ответ


```
[
 {
  "id":470,
  "tx":null,
  "to_addr":"TRx6xXChS5sXz3mpvLSNfKuL6w3PBdMZzL",
  "value":"1,005",
  "value_hex":"0",
  "unid":"2E19FB324B3867C146258915006AC7F0",
  "init_ts":1670786865
 },
 {
  "id":469,
  "tx":null,
  "to_addr":"TRx6xXChS5sXz3mpvLSNfKuL6w3PBdMZzL",
  "value":"1,005",
  "value_hex":"0",
  "unid":"C9D0C61AD75DF26F4625891500402B26",
  "init_ts":1670758938
 }
]
```


– tx – хэш транзакции; заполняется только если транзакция уже размещена в сети, до этого передаётся null.

Получение списка ожидаемых подписей для транзакции

```
GET https://my.h2k.me/ec/tx_sign_wait/:tx_unid
```

Пример

```
GET https://my.h2k.me/ec/tx_sign_wait/AD9EAFA6DADCC3BE4625890E007795B3
```

Ответ


```
[{"ecaddress":"0x534dfB93c5e19974C16c380e9B822CD80Cc3a825"},{"ecaddress":"0x534dfB93c5e19974C16c380e9B822CD80Cc3a825"}]
```


Получение списка полученных подписей для транзакции

```
GET https://my.h2k.me/ec/tx_sign_signed/:tx_unid
```

Пример

```
GET https://my.h2k.me/ec/tx_sign_signed/AD9EAFA6DADCC3BE4625890E007795B3
```

Ответ


```
[{"ecaddress":"0x534dfB93c5e19974C16c380e9B822CD80Cc3a825"},{}]
```


Получение списка полученных и ожидаемых подписей для транзакции

```
GET https://my.h2k.me/ec/tx_sign/:tx_unid
```

Пример

```
GET https://my.h2k.me/ec/tx_sign/AD9EAFA6DADCC3BE4625890E007795B3
```

Ответ


```
[{"signed":{"ecaddress":"0x534dfB93c5e19974C16c380e9B822CD80Cc3a825"}},{"wait":{"ecaddress":"0x534dfB93c5e19974C16c380e9B822CD80Cc3a825"}}]
```


Получение массива tx_unid (транзакций)

Запрос возвращает массив информации по транзакциям, которые должен утвердить (подписать) отправивший запрос пользователь (владелец ECAddress подписи запроса)

```
GET https://my.h2k.me/ec/tx_by_ec
```

Ответ

```
[{"token":"5010:::TRX###6F8B336576989FA246258909002D76E3","to_addr":"TRx6xXChS5sXz3mpvLSNfKuL6w3PBdMZzL","tx_value":"1,105","init_ts":1670190446,"unid":"AD9EAFA6DADCC3BE4625890E007795B3"}]
```

Подписание транзакции

```
POST https://my.h2k.me/ec/tx_sign/:tx_unid
```

Пример

```
POST https://my.h2k.me/ec/tx_sign/AD9EAFA6DADCC3BE4625890E007795B3
```

Ответ

```
{}
```

Отклонение транзакции

```
POST https://my.h2k.me/ec/tx_reject/:tx_unid
```

Пример

```
POST https://my.h2k.me/ec/tx_reject/AD9EAFA6DADCC3BE4625890E007795B3
```

– {"ec_reject":"Это текст причины отклонения транзакции",}.

Ответ

```
{}
```

Получение массива токенов пользователя и общего баланса по ним

Запрос возвращает массив с информацией обо всех токенах пользователя и общий баланс по ним. Базовые токены возвращаются даже если баланс 0, производные только те, по которым были изменения баланса.

```
GET https://my.h2k.me/ec/user_tokens/
```

Ответ

```
 [
  {"network_name":"Tron Nile TestNet (TRX)","token":"TRX","value":100},
  {"network_name":"Tron Nile TestNet (TRX)","token":"USDT","value":10}
]
```

Получение последних 50-ти транзакций, которые подписал пользователь

Запрос возвращает массив с информацией.

```
GET https://my.h2k.me/ec/tx_sign_signed/
```

Ответ

```
[

{"to_addr":"TRx6xXChS5sXz3mpvLSNfKuL6w3PBdMZzL","tx_value":"1,005","tx":null,"init_ts":1670758938,"info":null,"token":"5010:::TRX"},

{"to_addr":"TRx6xXChS5sXz3mpvLSNfKuL6w3PBdMZzL","tx_value":"1,005","tx":null,"init_ts":1670758133,"info":null,"token":"5010:::TRX"}
]
```