Elastos DID Specification（draft）

create, read and write a did

Song SJun

2018

目录

[概述 2](#_Toc526645767)

[Elastos DID 2](#_Toc526645768)

[DID的生成方法 2](#_Toc526645769)

[DID的命名规则 2](#_Toc526645770)

[验证DID与公钥的对应关系 2](#_Toc526645771)

[验证是否是DID的身份持有人 3](#_Toc526645772)

[Elastos DID的费用 3](#_Toc526645773)

[Elastos DID Side-Chain接口方法 3](#_Toc526645774)

[DID属性的格式 3](#_Toc526645775)

[DID的Write方法 3](#_Toc526645776)

[DID的Read方法 4](#_Toc526645777)

[DID的Remove方法 4](#_Toc526645778)

# 概述

DID (Decentralized Identifier) 是指可以由用户自主发行、自主证明拥有权的数字身份。

在传统互联网的技术实现里，通常需要由中心发行数字身份。这样即可以避免命名冲突;也可以完成身份验证。再通过这个中心完成陌生人之间的身份验证。

受比特币的去中心化钱包的启发，我们可以使用钱包地址作为用户的ID，通过地址对应的公钥进行验签，完成身份验证。陌生人之间不再需要第三方来确认身份。从而为互联网的独立、自主运行提供了一种数字身份的解决方案。

Elastos在上述基础之上，实现了Elastos DID Side-Chain，为互联网世界提供去中心数字身份的解决方案。让每个人都可以免费拥有属于自己的DID。通过拥有DID，每个人都可以在互联网世界里自我证明 (self-sovereign)；每个人都可以为自己的数字资产确权；每个人都可以基于可信身份安全通信；

Elastos DID基于Elastos DID Side-Chain生成、验证和存证。本文档将对Elastos DID进行描述和定义。

# Elastos DID

我们在这里将这个Elastos DID Side-Chain私钥对应的地址定义为Elastos DID。

## DID的生成方法

Elastos基于BIP44实现多侧链私钥方案，DID Side-Chain具体算法和参数如下：

生成私钥的路径为：” m/44'/1'/0'/change/address\_index”。

生成公钥的算法和参数：基于ECC结合HMAC-SHA512算法和secp256r1参数生成对应公钥。

生成DID的算法和参数：基于SHA256算法采用P2SH方式生成对应地址。

签名的算法和参数：采用ECDSA算法和secp256r1参数实现签名。

## DID的命名规则

DID的首字母为“I”。

## 验证DID与公钥的对应关系

通过根据公钥生成DID 的算法再次生成DID，可以用来检验所得到的公钥和DID是否匹配。

## 验证是否是DID的身份持有人

用公钥生成DID，验证两个DID是否匹配，用来验证DID与公钥是否匹配。

通过公钥对签名进行验签，用来验证是否是该公钥对应的私钥的签名。

## Elastos DID的费用

创建DID不需要手续费。但任何向Elastos DID Side-Chain的写入操作都需要手续费。

一次交易写入小数据量的收费比较低，收费标准是：xxxxxxx（待补充）

一次交易写入大量数据的收费比较高，收费标准是：xxxxxxx

# Elastos DID Side-Chain接口方法

Elastos DID Side-Chain拥有标准区块链访问接口，可以通过这些标准接口查询区块信息、交易信息和原始内容。另外也可以通过Elastos DID 相关接口完成对DID属性的写入和读取。

## DID属性的格式

DID的属性路径兼容标准URI (<https://tools.ietf.org/html/rfc3986>) 中path-absolute的定义。它的根是DID，从次一级开始作为属性的路径。ABNF定义如下：

DID-Property-Path = DID [ \*(did-path) ] ”/” did-property [ "#" did-fragment ]

DID = Elastos DID side-chain private key address

did-path = "/" [ 1\*pchar \*( "/" 1\*pchar) ]

did-property = 1\*pchar

did-fragment = 1\*pchar

pchar = unreserved / pct-encoded / sub-delims / ":" / "@"

unreserved = ALPHA / DIGIT / "-" / "." / "\_" / "~"

sub-delims = "!" / "$" / "&" / "'" / "(" / ")" / "\*" / "+" / "," / ";" / "="

pct-encoded = "%" HEXDIG HEXDIG

## DID的Write方法

Write方法用于写入DID的属性。

参数：

Path：DID Property path，用于标识目标操作的DID和相应的属性。

Value：写入属性的具体内容，采用固定的JSON格式 [ “value”, “value”,… ]

Sign：使用DID的私钥对Value进行签名的结果。

返回：

Result：执行结果，成功或失败。

TXID：对应此次写入的交易Hash值。

可能失败的原因：

签名不正确。

公钥与DID不匹配。

举例：

为一个DID写入名字属性

Path：”IHLhCEbwViWBPwh1VhpECzYEA7jQHZ4zLv/name”

Value： [ “Alice” ]

Sign：”E6BB279CBD4727B41F2AA8B18E99B3F99DECBB8737D284FFDD408B356C912EE21AD478BCC0ABD65246938F17DDE64258FD8A9684C0649B23AE1318F7B9CEEEC7”

## DID的Read方法

Read方法用于读取DID的属性，如果有多条记录，默认只返回最新的。

如果传入DID对应的公钥，将使用它验证Value的签名是否匹配。如果没有传入公钥参数，则不做验证。

参数：

Path：DID Property path，用于标识目标操作的DID和相应的属性。

TXID：写入时返回的TXID。如果提供TXID，则返回它对应的Value，否则返回最新记录。

Public Key：对应DID的公钥，用于验证返回内容，可选。

返回：

Result：执行结果，成功或失败。

Value：要读取的属性的内容。

TXID：该记录所在的交易Hash。

Verified：标识所返回的结果是否验证过签名。

可能失败的原因：

没有找到对应的属性。

查找到的属性已被删除。

签名不正确。

公钥与DID不匹配。

## DID的Remove方法

Remove用于删除DID的属性。当一个DID的属性被Remove以后，Read将不会再返回它的Value。

参数：

Path：DID Property path，用于标识目标操作的DID和相应的属性。

TXID：最近一次写入该属性时返回的TXID

Sign：使用DID的私钥对TXID参数进行签名的结果。

返回：

Result，执行结果，成功或失败。

TXID：该记录所在的交易Hash。

可能失败的原因：

没有找到对应的属性。

查找到的属性已被删除。

签名不正确。

公钥与DID不匹配。