SILUBIUM白皮书V3.3

SILUBIUM白皮书

价值传输协议及去中心化应用平台

SILUBIUM WHITE PAPER

Value Transfer Protocol and DAPP Platform

contact : djw@silubium.org

2018年10月15日

目　录

第三部分　SILUBIUM应用

第四部分　SILUBIUM使用情况及里程碑

第五部分　其他

摘　要

经过长达一年的开发，SILUBIUM团队综合研究对比市场其他公链，对SLU进行技术升级，兑变成为崭新的一条集多种开放功能于一体的公链SILUBIUM。SILUBIUM Blockchain (SLU)是全球第四代智能价值公链，并致力于拓展区块链技术的应用边界和技术边界，使普通互联网用户能感受到区块链技术的价值。

在SLU系统中，可以通过价值传输协议(Value Transfer Protocol)来实现点对点的价值转移，并根据此协议，构建一个支持多个行业的(金融、物联网、供应链、社交、游戏、预测等)去中心化的应用开发平台(DAPP Platform)。

SLU系统通过良好的设计原则和设计策略来实现，例如兼容性原则、模块化设计策略、安全性策略和易用性策略。从技术角度分析，SLU致力于实现兼容BI(基于UTXO模型)的POS智能合约平台，将引入Identity，Oracle和Data feeds。在合规性方面，符合不同行业的监管需求。在SLU的公链(Public Blockchain)系统中，在共识机制上，从去中心化程度、实用性、技术可靠性考虑，我们将以 Proof of Stake 为基础，加入节点在线激励因素(Incentive Factor)， 形成IPOS(Incentive POS)的共识协议。在SLU的联盟链中(Permissioned Blockchain)，我们将采用SLU开发出的与Raft融合的Proof of Time 共识协议，使得在联盟链或者私链中，达成共识的时间大大缩短(BlockTime:250ms，Confirmation Time:750ms-3s)。

SLU系统将基于UTXO模型来实现区块链的智能合约，主要考虑以下因素：(1)与比特币生态的兼容性；(2)BIP长期演进协议的兼容性 ；(3)交易的并行处理能力/隐私性/可追溯性。

在SLU系统中，我们把区块链合约(Blockchain Contract)分成智能合约(Smart Contract)和主控合约(Master Contract)，除了支持智能合约外，我们将通过链下因素的引入，形成符合现实世界商业逻辑的区块链主控合约。另外在虚拟机方面，我们兼容EVM，后期通过标记不同的虚拟机类型，可以支持更多的虚拟机，包括LLVM 和 Lua 以及 EVM2.0. 以及为VM开发的更严格的编程语言。

在SLU系统中，我们通过Oracle 和 Data Feed的设计，可以让区块链的智能合约更快落地和更符合商业规则，搭建了现实世界到区块链世界的桥梁。另外SLU系统中，可以通过智能合约来管理参与者的身份信息，将为基于SLU系统的金融服务提供更好的支持。

最后面向移动端策略(Run Mobile)也是SLU特别重视的一个战略，在SILUBIUM的生态系统中，我们将会与第三方开发者，一起从技术架构支持供移动端的服务，包括：移动端钱包、移动端DAPP应用、移动端智能合约服务。我们也鼓励第三方的开发者，加入我们，一起开发区块链的移动端服务，共同推动区块链技术的落地。

第一部分　SILUBIUM设计原则和理念

一、区块链出现的背景和意义

在2008年10月31日，Satoshi Nakamoto <satoshi@vistomail.com> 通过一个密码学小组(gmane.comp.encryption.general)发送了一封邮箱，并第一次公布了比特币的白皮书《Bitcoin: A Peer to Peer Electronic Cash System》，并列出了比特币网络的一些特点:

(1) Double-spending is prevented with a peer-to-peer network 防止双花

(2) No mint or other trusted parties 无铸币厂或其他信任方

(3) Participates can be anonymous 参与者可匿名

(4) New coins are made from Hashcash style Proof-of-work 通过工作量证明方式发行新币

(5) The proof-of-work for new coin generation also powers the network to prevent double-spending 基于工作量证明的新币发行过程中，也同时阻止了双花的发生。

在2009年，比特币的创始区块被挖出，并在第170个区块发生了第一笔比特币的 转账交易(从atoshi到Hal Finney，发生在2009年1月12号)，从此开启了比特币网络作为一种点对点的价值交换网络蓬勃发展的时代，虽然中间经历了各种危机，但是比特币网络的价值从零开始，到今天已经成为一个价值约1100亿美金的点对点支付网络。

点对点价值传输网络的出现有其历史必然性，而Satoshi则是加速这个历史进程的人。从上个世纪80年代，TCP/IP协议的开发，到90年代，网页浏览器的应用和服务器的应用，一直到今天，互联网技术从不同侧面和维度改变了数据交换的模式和人类的生活。互联网技术的发展得益于基础设施的完善，从早期的信息高速公路(Information Super Highway)和各种智能终端的普及，这些也构成了互联网OSI七层模型中，应用层无限拓展的基础。

在互联网的各种协议栈中，我们用的较多有TCP/IP，HTTP，HTTPS，FTP，TELNET，SSH，SMTP，POP3等网络层、传输层、应用层的协议，并且借助这些协议，我们已经比较完美的搭建了各种各样的互联网服务。但是如果我们深思，我们会发现，在比特币网络出现之前，我们一直无法互联网上面，在不借助于第三方的情况下，较好的进行点对点的价值的转移和传输。其实我们并不是缺少一种特定的方法，而是缺少基于信息高速公路(Information Super Highway)的价值高速公路(Value Super Highway)，以及如何实现Value Super Highway的Value Transfer Protocol (VTP协议)，而比特币网络则是运行于信息高速公路上面的第一个VTP协议。在SILUBIUM的白皮书中，我们也第一次归纳和出了互联网应用层Value Transfer Protocol的概念。

随着互联互通技术的发展(互联网、物联网、VR/AR)，人与物体、人与信息的交互方式更加多样化，更多的实体被数字化(Digitalize)和令牌化或者代币化(Tokenize)和符号化(Symbolize) ，一旦实体被数字化或者代币化之后，就完成了实体资产在互联网上面的映射和切分，马上面临的一个问题就是:如何点对点传输这些资产和价值?

因此可以推测，随着互联网服务的进一步深入，实体和虚拟的边界也会开始模糊，点对点价值转移的需求会被凸显出来，因此在互联网上面的Value Super Highway和Value Transfer Protocol必然会出现，而比特币网络加速了这一历史进程。

二、为什么设计SILUBIUM

自从2009年比特币代码开源以来，社区里面出现了很多Altcoin和其他区块链项目，有意义的Altcoin项目成为了区块链技术的试验田(一些毫无意义的Altcoin除外)，对区块链技术的发展和成熟有一定的借鉴意义(例如NameCoin等)，除此之外还有一些从不同角度拓展区块链技术边界的项目，例如ColorCoin协议，NXTCoin，Ripple 和 Stellar，BitShare，Dash，Maidsafe，Factom 等。之后，还有致力于成为通用智能合约平台和去中心化应用平台的Ethereum项目。无数的开发者和社区人员一起参与和见证了区块链技术的快速发展，但是区块链行业不论是从技术角度，还是行业应用角度都还面临着很多挑战。

区块链技术面临的主要问题:

(1) 缺乏新型的智能合约平台，目前现有的智能合约平台主要是基于Proof of Work(POW)，而Proof of Work(POW)的共识机制很难被行业应用大规模部署。

(2) 不同区块链技术之间的兼容性，比如基于UTXO模型的比特币生态和基于Account模型的以太坊生态很难有兼容性。

(3) 共识机制本身缺乏灵活性，因为参与者的不同，在公有链中和联盟链中，对共识机制的要求是不一样的。

(4) 缺乏对行业合规性的考虑，例如在金融行业要求的identity和KYC部分，在现有的区块链系统中，很难保证。

(5) 目前的区块链公链基础架构还在逐步完善，因为受交易速度和安全保障的制约，很难有一个公链能产生较大规模的应用。

现有区块链系统具备很大的封闭性，目前大多数的智能合约的触发条件大多来自于区块链系统本身，很少有来着外界的触发条件，缺乏与现实世界的交互。

针对当前区块链行业的挑战，SILUBIUM在区块链技术和理念上进行了一系列的创新：包括基于UTXO的智能合约模型，面向公有链和联盟链的灵活的共识机制，区块链主控合约的理念和实现， 交易账本和智能合约账本的分离，Oracle和Data Feed的设计和实现等，使得SILUBIUM成为区块链世界与现实商业世界的桥梁。

对比互联网技术的发展路径，我们发现不论是区块链技术本身，还是基于区块链技术的应用，都处于行业发展早期，有很多值得探索的方向。

因此我们希望可以构建一个全新的区块链生态系统，作为未来世界可选的互联网价值传输协议的可选项，并把整个区块链行业的易用性向前推进一步，这也是我们设计SILUBIUM的原因。

SILUBIUM致力于拓展区块链技术的应用边界和技术边界，使普通互联网用户能感受到区块链技术的价值，并构建一个全新的基于区块链技术的开发者和用户的生态系统。

三、SILUBIUM的设计原则

3.1 SILUBIUM兼容性设计

与比特币网络和以太坊网络的兼容性(Compatible with Bitcoin and Ethereum network)比特币网络的生态系统是目前最大的一个区块链技术的生态系统，根据网络效应和马太效应(Matthew Effect)的影响，我们有理由相信比特币生态系统会进一步扩大和完善，也意味着更高的代码成熟度和更多的开

发者。SILUBIUM在设计之初，就尽量保持与比特币系统的兼容性，例如同样使用UTXO的交易模型和相同的交易数据结构等，这也为以后借力比特币的BIP协议，提供了技术上的可能性。因此SILUBIUM后面可以兼容大部分的BIP协议，例如闪电网络(lighting network)和侧链(sidechain)和驱动链技术(drivechain)和基于零知识证明的Zcash协议等。

Ethereum第一次将智能合约的概念从理论变成实际，从而拓展了区块链技术的边界，其中的Ethereum Virtual Machine (EVM)虽然有很多值得改进的地方(例如: Transaction-ordering Dependence attack/Timestamp Dependence Attack / Mishandled Exceptions等)，但是EVM是目前为止唯一一个经过测试的智能合约虚拟机，因此保持和EVM的兼容性，就显得非常重要，因此SILUBIUM的虚拟机将保持和EVM的兼容性，所有在以太坊平台上面开发的智能合约，也可以在SILUBIUM平台上面运行。

SILUBIUM本身的向下兼容性(DownwardCompatibility)。

软件的向下兼容性也是一个非常重要的问题，使用旧版本创建的文件和智能合约，将能持续在新版本上面运行，而不用用户强制升级，这将会给用户带来很多便利。因为智能合约的特殊性和一次性部署，如果不能实现向下兼容性，将会给已经执行过的智能合约带来很大问题，也造成后期软件无法迭代和升级，也会出现EVM2.0和EVM1.0无法兼容的问题，这也是区块链系统软件设计者需要注意的问题。

3.2 SILUBIUM模块化设计思想

模块化的设计更利于软件的开发和维护，因此在SLU中，我们分为以下3个大的模块 ：SILUBIUM技术模块SLU Tech :SLU Core、SLU VM、SLU identity、SLU Oracle and DataFeed、SLU Storage等;

SILUBIUM用户交互模块 SLU UI :SLU IDE、SLU Mobile、SLU Web 、SLU Node 等;

SILUBIUM商业路径模块:SLU Business:SLU Financial、SLU legal and risk、SLU Industry、SLU Competitor 等。

3.3 SILUBIUM安全性策略

A. SLU基础平台所用技术的可靠性

SILUBIUM的第一阶段致力于提供一个面向不同行业的高可用的兼容UTXO模型的POS机制的智能合约平台。UTXO模型是比特币网络非常核心的一部分，其中的代码具备比较高的成熟度，兼容UTXO模型，不但可以吸纳比特币生态系统的其他开发者和现有开发工具，还可以具有比较好的安全性。

关于共识协议部分，除了中本聪在比特币网络中采用的Proof of Work(包括Sha256/Sha3/scrypt/X11/X13/ 等其他变种)，经过比特币网络8年的大规模测试以外，在公链服务中，另外一个广泛运用和被测试过的共识协议就是Proof of Stake (POS/Dpos/POS 后续演进协议等)，并且在POS后续演进协议中(POS2.0/POS3.0)已经逐步消除了针对POS协议的各种潜在攻击(例如:币龄攻击、POS节点全部离线， 预先计算Hash值攻击等)。

因此在SLU系统中，我们第一阶段采取的共识协议的基础为POS协议，并添加激励措施，鼓励节点在线，我们可以称之为IPOS(Incentive Proof of Stake)。目前在我们的SLU测试网络中，所采取的依然是POS3.0的协议，后续的开发中，会逐渐转移到IPOS协议。关于共识协议的长期演进，为了取得更高的可拓展性，以及未来支持基于SLU系统的私链服务，在共识协议的第二阶段，我们将采用基于时间序列(POT:Proof of Time)和Raft协议融合的共识协议，期望取得金融级别的数据处理能力。在我们的第二阶段的共识协议的模拟中，设计目标为:区块时间250毫秒，确认时间:750毫秒-3秒。

因为融合 Raft 协议，更适合在有限的网络节点中，随机选取 Leader 来进行周期性的记账。第二阶段的共识协议的设计，更多是为了以后基于SLU系统的私链网络或者联盟链网络而设计，是否可以用到SLU的后续共识协议演进中，我们会做进一步探索。

关于智能合约的虚拟机部分，SLU目前测试网络中支持EVM，EVM也是目前为止唯一 一个被测试过的智能合约虚拟机，后期重点包括(1)开发一种更严格的智能合约编程语言；(2)支持EVM2.0(Wasm)；(3)探讨在其他虚拟机平台上(LLVM、Lua、NodeJS) 移植的可能性。

关于Identity 和 Oracle(Data Feed)部分，详见后续章节。

B. SLU平台发行的安全性策略

SLU 平台在发行前将会经过一系列严格测试，其中包括软件功能性测试、P2P网络性能测试、潜在攻击向量测试、可靠性测试，安全审计和代码审核，Alpha 版本测试，Bata 版本测试，通过完善的软件测试流程，来控制软件质量。

3.4 SILUBIUM易用性策略

针对节点的易用性，我们将为用户钱包提供两种或者三种不同操作模式，包括Simple Mode、Professional Mode、Expert Mode，分别对应不同的用户操作系统和开发需求，并且op_return加载数据大小从40bytes升至1024bytes。

除此之外，我们将供SLU系统的完善API服务，将基于Standard JSON-RPC来供远程的和本地的API调用服务，并考虑版本之间的兼容性。另外我们还将供SLU的IDE服务，可以开发和调试相关代码和服务。

此外，用户可以通过浏览器(Chrome or Firefox 等)访问DAPP服务，例如用户可以在浏览器中输入SLU://DappName 的形式，来访问SLU系统中的去中心化应用程序，例如在SLU系统的中一个去中心化拼车服务“车来了”，用户可以通过在浏览器中，输入:SLU://chelaile，就可以访问相应的DAPP服务。

第二部分 SILUBIUM实现方案

一、 SILUBIUM公链

SILUBIUM公链致力于开发除了比特币和以太坊之外的新的生态系统，通过完善的设计，来实现和比特币长期技术演进和以太坊虚拟机相兼容的特性;并且以行业应用为导向，通过移动端DAPP开发策略，把区块链的技术优势带给不同行业的应用者和普通互联网用户。另外SLU的公链系统注重智能合约的实际应用，将通过完善的Oracle和Identity部分的设计，给传统互联网企业(金融、物联网等)提供一个合规性的开放的区块链技术试验田。除此之外，SLU系统注重去中心化应用的开发，通过吸引第三方开发者加入，一起为普通用户供移动端的去中心化应用，所有根据SLU系统开发的第三方应用，SLU将通过完善的评价体系，给予开发者奖励。

二、UTXO vs Account Model

2.1 UTXO 模型剖析

在比特币的网络中，UTXO(Unspent Transaction Output未花费交易输出)是比特币交易的基本单位，通过交易的输入和输出，比特币网络将金钱变化成一段数据结构，区别于信用卡支付必须在加密安全网络中传输，比特币的数据可以在任何不一定安全的网络中传输(WiFi、蓝牙，NFC，表格等)。UTXO可以是“一聪”(1×10^{-8} BTC)的任意整数倍。尽管UTXO可以是小于2100万的任意数值，但是一旦UTXO被创造出来，只能作为一个整体被花掉。如果一笔交易需要的BTC小于某一个UTXO的值，那么该UTXO依然会被当做一个整体花费掉，并形成一个找零的UTXO。

UTXO可以看做被私钥的拥有者锁定的、并被整个比特币网络识别的比特币货币单位。

在UTXO模型中，被某一个交易消耗的UTXO被称为交易输入，由交易创建的UTXO被称为交易输出。通过这种方式，一定量的比特币在不同的私钥所有者之间转移，并在交易链条中不断消耗和创建新的UTXO。一笔比特币交易通过所有者的私钥签名来解锁UTXO，并通过使用新的所有者的比特币地址来锁定并创建UTXO。在比特币网络的起始的阶段，矿工通过一种特殊的交易类型，Coinbase交易创造的交易的输出(该交易没有输入)所产生的比特币，可以用于创建其他的UTXO。

UTXO被每一个全节点(Full Node)比特币客户端在一个储存于内存中的数据库所追踪， 该数据库也被称为“UTXO 集”或者“UTXO 池”，新的交易构建时从UTXO池中消耗一个或多个输出，而比特币网络监测着以百万为单位的所有可用的UTXO，世界上在比特币网络中并不存在“比特币余额”的概念，因为比特币网络上只会记录所有未花费的UTXO，比特币的余额的概念更多是通过比特币钱包客户端派生出来的产物，比特币钱包通过扫区块链并聚合所有属于该用户的UTXO来计算该用户的余额。

另外关于交易费用的问题，我们可以通过计算输入和输出的差额，来计算一笔交易的交易费用。

由于每一个比特币的全客户端都会对每一笔交易按照一系列的规则，进行独立校验，一笔比特币交易所有的交易信息都包含在脚本中，如果任何一个节点按照脚本执行，并对结果的有效性进行了校验，那么其他所有节点必将得到一致性的校验结果，这也意味着一笔有效的交易对所有人都是有效的。

比特币网络中的每一笔交易的执行依赖于解锁脚本和锁定脚本。解锁脚本可以解决锁定脚本对某一输出值的阻碍，锁定脚本会在某一笔输出值上设置花费的条件 。解锁脚本通常包含私钥的一个签名，也被称为ScriptSig，锁定脚本通常会把一个交易输出锁定到一个比特币地址上(公钥的哈希Hash值)。

比特币全节点客户端会同时执行锁定脚本和解锁脚本来验证某一笔交易的合法性。客户端会先检索输入所指向的UTXO，这个UTXO包含一个定义了花费条件的锁定脚本，然后客户端会读取试图花费这个UTXO的由客户端构造的输入中所包含的解锁脚本，并执行这两个脚本。如果从解锁脚本处复制好堆栈数据之后，再执行锁定脚本的结果为真，那么说明解锁脚本有权使用该UTXO，并发起新的交易。

比特币在交易中使用脚本系统，与FORTH(一种编译语言)一样，脚本是简单的、基于堆栈的、并且

从左向右处理，它特意被设计成非图灵完备的，没有循环(LOOP)语句的一种系统。在比特币网络中，脚本系统对数据的操作都通过堆栈完成(主堆栈和副堆栈)，堆栈是一个常用的抽象数据类型，最主要的特点就是后进先出(LIFO:Last In First Out)。

在比特币的客户端中，开发者把比特币客户端支持的脚本类型通过(Standard)函数做了一个总结，在(Standard)函数中包含5种类型的脚本：P2PKH (Pay to publickey hash)、P2PK(Pay to publickey)、MultiSignature(限15个私钥签名)、P2SH(Pay to Script hash)和OP_Return。这5种标准的脚本类型实现了，通过公钥哈希支付、通过公钥支付、多重签名、通过脚本哈希支付、以及数据输出的功能。通过这5种标准的脚本类型，比特币客户端可以实现较复杂的支付逻辑。另外，一个非标准化的脚本类型也有可能被创建，但是必须找到一个愿意打包该非标准化的交易的矿工，该非标准化的脚本才会被执行。

我们以P2PKH (Pay to publickey hash)为例，来说明脚本的产生和执行过程。假设我们需要向某一个面包店支付0.01BTC 来购买面包，面包店的地址为：Bread Address。

则该交易的输出为：
OP_DUP OP_HASH160 <Bread Public Key Hash> OP_EQUAL OP_CHECKSIG

锁定脚本对应的解锁脚本为：
<Bread Signature> <Bread Public Key>

将两个脚本结合起来可以形成如下组合脚本:
<Bread Signature> <Bread Public Key> OP_DUP OP_HASH160 <Bread Public Key Hash> OP_EQUAL OP_CHECKSIG

只有当解锁版脚本与锁定版脚本的设定条件相符合的时候，执行组合脚本时才会显示结果为真(Ture)。要想执行组合脚本的结果为真，也就意味着，Bread Signature的签名是有Bread Address对应的私钥所签名，是一个Bread Address的有效签名，只有这样交易执行结果才会通过(结果为真)。

虽然比特币的脚本语言包含很多的操作符，但是需要注意的是比特币脚本语言是非图灵完备的。在该脚本语言中，是没有循环功能的，这也意味着交易的复杂性有限，交易的可执行次数有限。脚本并不是一种通用的编程语言，这些限制也避免了潜在创造无限循环或者其他复杂逻辑漏洞的支付条件，从而对比特币网络的安全性留下隐患。

在UTXO模型中，我们可以通过公开的账本清晰的追溯每一笔交易的历史记录，并且可以做到完全透明，另外UTXO模型也带来了一定的并行处理能力，可以发起多地址到多地址的交易，为可拓展性带来一定借鉴。除此之外UTXO模型也带来了一定的隐私性，用户可以通过Change address，来作为UTXO的输出地址。但是UTXO本身是无状态的，我们将通过一系列的创新的设计，实现基于UTXO类型的智能合约。

2.2 Account 模型剖析

与UTXO模型不同的是，以太坊是有账户体系的，在以太坊的白皮书中，我们可以看到以太坊的账户体系:

在以太坊系统中，状态是由被称为“账户”(每个账户由一个20字节的地址)的对象和在两个账户之间转移价值和信息的状态转换构成的。以太坊的账户包含四个部分: 随机数，用于确定每笔交易只能被处理

一次的计数器，账户目前的以太币余额，账户的合约代码(如果有的话,账户的存储默认为空)。

以太币(Ether)是以太坊内部的主要加密燃料，用于支付交易费用。一般而言，以太坊有两种类型的账户：外部所有的账户(由私钥控制的)和合约账户(由合约代码控制)。外部所有的账户没有代码，人们可以通过创建和签名一笔交易从一个外部账户发送消息。每当合约账户收到一条消息，合约内部的代码就会被激活，允许它对内部存储进行读取和写入，和发送其它消息或者创建合约。

在以太坊系统中，通过一个有状态的账户系统来记录账户余额，每个账户余额的增加/减少更现实世界中的银行记账方式，每产生一个新的区块，都会可能对全局状态造成影响。每个账户都有自己的余额、存储和代码区域。这样合约就可以调用账户或者地址，并且把相应的执行结果在存储区域进行存储。

在目前以太坊的账户系统中，通过client/rpc，只能进行一对一的转账，也就意味着每次只能从一个账户转移到另外一个账户。尽管通过智能合约可以发送到更多的账户，但是这些内部交易只能在用户的账户余额上显示，却很难在以太坊的公开账本上追踪。

比特币网络的UTXO模型，保证了比特币交易的连续性和可追溯性，也是比特币架构的核心设计，考虑到比特币的网络效应和UTXO模型的优点，在SLU的公链系统中，我们第一步决定采取基于UTXO模型。

三、SILUBIUM 第二代SLU共识机制Silkworm

3.1 Pos1.0、Pos2.0、Pos3.0

自从中本聪在2008年创造出比特币以来，工作量证明（Proof of Work，以下简称 PoW）的设计理念已成为P2P电子货币的主流思想。在中本聪的设计中，PoW是保证采矿（特殊交易：coinbase）及BTC安全的支柱，本质意味着BTC需要消耗能源来维护运行，维护这样一个网络的运转需要消耗大量的成本。

2011年创造的PPC是从BTC衍生出来的一种P2P的电子密码货币，以权益证明（Proof of Stake，以下简称 PoS）取代 PoW 来维护网络安全。PoS 是一种利息币(特殊交易：coinstake)，区块持有人可以消耗他的币龄（coinage）获得利息，同时获得为网络产生一个区块和用PoS造币的优先权。

PoS1.0中的币龄可能会被恶意的节点滥用以获得更高的网络权重并成功实施双花（doublespend），诚实的节点可以通过定期开启钱包进行权益累积（staking）而滥用这一系统，另权重修正因子没有对哈希功能进行足够的模糊处理以防止攻击者提前计算出未来的权益累积证明，因此恶意的攻击者将能够计算出权益累积证明的解答的下一次间隔，从而能够连续生成多个区块并实施能够危害到整个网络的恶意攻击。

2014年成立的Blackcoin，对 PoS1.0进行优化到PoS2.0，去掉了币龄，就需要所有节点必须更多的保持在线以进行权益累积，权重修正因子在每一次修正因子间歇时都会改变，对区块时间戳

做了适当的改变，使其在PoS机制下更有效的工作，将区块哈希算法从Scrypt改回到SHA256d。PoS3.0 进行了一些更新，最显著的改变是从1%的年度PoS奖励变为静态1.5BLK（黑币）。

3.2 MPoS

2017年QTUM的MPoS共识机制是基于Blackcoin PoS3.0改进而来。MPoS机制中没有限制通证的最小抵押数额，在抵押通证参与挖矿过程中加入了别的限制因素。首先，QTUM通证在完成一次转账或者完成一次抵押之后，新收到的或解除抵押通证需要等待500个区块的生成时间。其次，每当一个新区块生成，它的生成者只能立即收到总收益的十分之一，余下的收益将分给五百个区块高度之前的九个区块生成者，这将减少垃圾交易攻击的风险，同时也为用户参与生成区块提供了相对公平的机会。

3.3 Silkworm

2018年SILUBIUM从第一代SLB升级到第二代SLU，创造了基于MpoS改进的共识机制Silkworm。首先，新收到的或解除抵押通证只需要等待100个区块确认，加快流转过程。其次，新区块生产者收到总收益的五分之一，余下的收益将分给一百个区块高度之前的四个区块生产者。再次，区块奖励减小为1SLU，同时若生产区块的UTXO产生时间超过30天，则可获得0.58%的月利息。Silkworm鼓励节点在线，获得区块生产权力的同时获得利息，在线节点越多SILUBIUM越安全稳定。

四、合约和虚拟机 Contract and VM

在本小节，我们主要讨论合约，以及合约的执行环境虚拟机，以及SILUBIUM系统的出的主控合约(Master Contract)的理念。这里的Contract我们指代Blockchain Related Contract，并把区块链合约分成Smart Contract 和 Master Contract。

智能合约 Smart Contract:区块链合约代码通过虚拟机执行，并且不侧重链下数据的输入，借助于区块链网络本身 供合约触发条件，完成合约的执行。

主控合约Master Contract:区块合约代码通过虚拟机执行，侧重链下数据的输入(Oracle和DataFeeds)，通过链下数据和区块链网络的共同输入作为触发条件，完成合约的执行。在SLU的系统中，考虑到合约的实际用途，我们会设计完善的Oracle和DataFeed服务，把区块链的合约推向实际的商业应用场景。

我们来看一下比特币网络中的最简单的合约类型：多重签名。

在比特币的2 of 3的多重签名合约中，参与方同意把比特币资金放到一个交易的输入中，通过

一个多重签名的交易，花费这币资金通常需要至少2方的同意才能花费这笔资金。这种合约需要一个仲裁者(Mediator)作为第三方参与，因为多重签名合约会出现两种结果。第一种同意合约的执行结果，并且都发布自己的签名。第二种结果是其中一方不同意发布签名，造成这笔资金无法使用，这个时候就需要仲裁者参与，并且把资金给释放给他认为正确的一方。在这种合约过程中，我们可以把Mediator设计成自动供可信数据源的Oracle，并且从外部实际获取相应的数据，而不是从区块链网络本身获取合约执行的输入数据。

得益于以太坊的网络的合约执行环境EVM，以太坊中的合约的编写和执行变得非常简单。但是现阶段以太坊上面的合约执行没有过多引入外面数据的干预，也就造成了在显示商业场景中的局限性。在SLU的合约平台上，我们将把外部数据和干预措施抽象为Oracle和DataFeed，期望通过主控合约形式，把区块链的合约带到具体的商业场景中。

我们来看一个简单的合约逻辑(ST为SLU系统的Token):

```
function unlock(oracles, inputs) {
var better1Amount
var better2Amount
if (oracle.name.win == “barcelona” ) {
better1Amount = “1-ST”
better2Amount = “0-ST” } else {
better1Amount = “0-ST”
better2Amount = “1-ST”
}
return {
 “outputs” : [
{ recipient : “1MyuemkT4raRPPjhDwd9MyzbBNt9QAwKHc” ,
value : better1Amount },
{ recipient : “1MdnpBCnFnjNFUaChHUMjfdW6bvhpR8WQi” ,
 value : better2Amount }, ],
 “fee” : [ “0.5-ST” , ] }
}
```

如果在一场Barcelona对阵Real Madrid的比赛中，Barcelona赢得了比赛(作为Oracle的输入)，合约参与者1赢得比赛，否则输掉比赛。我们将在后面的章节中，具体探讨Oracle的创建和格式。

Ethereum中的DAO事件暴露了智能合约设计中可能存在的潜在安全因素，因为在以太坊上面的智能合约一旦部署，EVM就会通过预先定义的寻址指针和OP_CODES，一步一步执行合约代码，并且根据合约的执行结果对全局状态进行转换。现实世界中的软件开发一般通过多次迭代来完成，但是智能合约一旦执行无法通过迭代改进，这个特性虽然符合区块链的准则，但是与现实世界的社会准则和商业准则有很大不同。因此在SLU系统中，除了支持以太坊的智能合约，我们将建立现实世界与区块链的桥梁，通过链下规则的引入，把最简单的和最小集的合约需求写在区块链上面，例如合约的参与方和授权方等，并通过链下数据的输入来触发合约的执行结果，以及通过链下的仲裁

来升级合约代码等。主控合约(Master Contract)的初衷在于引入社会规则和商业规则到区块链上面，使得区块链技术更容易对接现实世界的需求。

五、VM on SILUBIUM UTXO区块链

5.1 SLU 标准交易类型和合约交易类型

在UTXO模型的区块链系统中，VM部分会独立于原有的脚本语言单独存在。在SLU系统中，存在两种交易结构，分别是Standard Transaction和VM Transaction。

StandardTransaction：标准交易类型，我们标记为V1类型；VMTransaction：合约交易类型，我们标记为V2类型。

正常的类似比特币的交易类型我们标记为Version1，Version1的transaction保持与比特币网络标准交易的相似性，对于用到虚拟机的部分我们创造一个新的交易类型，我们标记为Version2。

在Version2的交易类型中，我们会设计以下区域:

输入和输出有一个“vm”区块，1代表标准的交易类型，2代表需要通过VM执行的交易类型。

这样新的VM合约可以通过创建一个新的输出来构建，并且随后可以通过一个标准交易来把SLU Token发送到一个合约地址。其中Token可以通过新的操作符来分配，例如通过一个assign-to的opcode来分配。

5.2 SLU 系统存储合约代码

通过拓展比特币网络的脚本语言，添加新的操作符，比如 OP_VM ，合约的bytecode将会被编码到交易输出中，合约字节数取决于系统设计的MAX_SCRIPT_SIZE。

5.3 SLU 系统中合约的创建

在合约创建的时候，通过创建人来建立合约

```
contract Escow { address owner Escow() {
owner = msg.sender }
}
```

通过现有的脚本语言的解释器来执行OP_VM 操作符，并把控制权转移给VM来执行相应的合约。

5.4 SLU 合约花费SILUBIUM

合约作为某一笔交易的输出的时候，该输出又可以作为 ‘send’ opcode 的输入，Sendopcode可以把V1和V2类型的输出和一定量的SILUBIUM发送到另外一个输出或者地址。

我们来看一个向某一个SLU合约发送SILUBIUM的过程。

假设在第100个block，我们有这样一个合约，合约的TXID为:1234，输出为0 contract MoneySender{
function sendmoney(outputScript){ send(outputScript, this.balance / 2)
} }
有两笔SLU的标准交易被发送到这个合约地址value of 100 coins at block 150, tx id 1 assign-to 100.0
Value of 50 coins at block 200, tx id 2
assign-to 100.0
稍后我们在第300个区块的时候，调用该合约call(contract100.0.sendmoney, myBitcoinAddressScript) //

最后我们将调用 Sendmoney 和 Send opcode. 并且该区块一笔有效的交易会包含所有的合约执行完成后的所有的send opcode.这样在该区块的一笔交易中将会有tx1和tx2的2个输入，并且包含myBitcoinAddressscript和change address的两个输出。

5.5 SLU 系统中合约状态的保存

每个合约脚本都有自己的状态，合约状态被存储在Statedb中，Statedb可以通过区块链中个合约重建(reconstruction)，我们把它叫做重建过程(reconstruction process)。状态总是可以重建/创建re(constructed)从现有区块中的合约。

其中Statedb应该可以重回的具体的交易中来处理一些冲突交易或者区块链中的短期的分叉。比特币网络所采用的Berkley DB并不是最优选择。

Statedb的状态只会收到确认的区块的影响，未确认的交易和合约本身都不会影响Statedb的状态。

5.6 SLU 系统中的密码学公钥方案

目前在以太坊中的公钥是通过预编译的合约来实现的，在SLU系统中，我们将通过VM的opcode来实现。

5.7 Gas 机制的变化

EVM系统中Gas的概念将会在SLU系统中被移除，合约本身和合约调用合约都会自己来设定相应的费用，这个费用需要包含整个交易过程的费用(包括递归调用recursive call)。为了防止P2P spam，会对每一笔交易收取一笔小的费用。除了这个费用，记账节点和网络将会选择某一费率下他们可以执行的opcode的数量。

记账节点在处理交易的时候会遵守以下的流程:

1. 拒绝不包含最低费用的交易
2. 如果交易字节数太大，并且费用太低，这笔交易也不会被执行
3. 如果交易字节数很小，并且费用适中/偏低，记账节点会执行交易，但是如果对应的opcode太多，也可能中途被终止
4. 高的交易费用的交易会被优先执行，除们将根据实际的需求，会设计一个opcode的执行次数的上限，比如限制在100k以内的operations。

5.8 合约账本(Contract Ledger)和合约的可读性(Contract Readability)

在SLU系统中，除了基于UTXO模型的可追溯的Transaction Ledger，我们还将构建一个合约内容的Contract Ledger，方便大家的审计和阅读智能合约。

在以太坊系统中，智能合约的编写者，可以选择不发布合约明文内容和合约意图。在SLU系统中，我们将构建一个Contract Ledger来存储所有的SLU明文可读性强的合约内容，用户可以选择性的把自己感兴趣的合约代码和合约解释通过P2P的形式下载到自己的SLU客户端。

Contract Ledger的构建，可以给SLU系统中的合约带来更多的透明性和可读性，以及可审计性。

5.9 SLU 系统合约地址

在SLU系统中，我们会根据以下规则构建新的合约地址类型，该地址区别于Standard Transaction Address

SLU Contract Address=

version + hash(spending_vout_txids[] + spending_vout_numbers[] + contract_bytes + contract_vout_number) + checksum

其中的Version number的不同，后续可能对应不同的VM执行环境。比如Version=1，代表兼容EVM的合约类型，Version=2代表兼容Lua VM的合约类型，这也给SLU合约带来的后续的可拓展性。

六、Oracle和DataFeed

本小节论述SLU系统中Oracle和Data Feed的理念和实现方法。

在SLU系统中的区块链合约注重合约在商业社会中的实用性。SLU系统中的合约可能会需要和现实社会中的数据来做交互。比如SLU系统中合约的执行有时候需要查询链外的数据(比如汇率、GDP、某个城市的温度、比赛结果等)。在SLU系统中，Data Feed代表任何可以用来从链外取得数据，并把数据供给区块链合约(或去中心化应用)的系统、程序、技术机制。

另外所有基于区块链的智能合约都需要所有节点执行相同的智能合约代码，也造成了智能合

约会花费巨大计算资源，也变得异常昂贵。考虑到实际商业中用途，在SLU系统中，针对联盟链，我们将创建一种链下合约的执行机制。正如我们在2.4小节中所述的，我们把这种引入链下触发条件的区块链合约，叫做主控合约。除此之外，链下合约还可以给参与方更多的隐私保护，这一点更符合金融机构的需求。

另外，很多智能合约的应用(选举、彩票、cut and choose protocol)等都需要随机数的产生，我们会在本小节最后探讨随机数的产生方式。

6.1 通过 Oracle 实现链下合约的执行

在SLU系统中，Oracle代表可信的特定的机构、实体、节点、公钥地址。Oracle通过data feed的工作模式，来实现自己的职责。

关于区块链系统中最小信任的数据源之前有一些探索，从“Schellingcoin”到“Truthcoin”通过两阶段高数据，并给接近平均值的数据源奖励，到通过参与者的权重来决定数据源的最终取值。也有一些项目尝试通过Intel最新的Software Guard Extensions来保证数据执行的可考虑，来最终供数据源。因为Oracle的复杂性，在智能合约领域，目前还没有一个可以激励Oracle供商的方式从而形成大规模的Oracle服务。

6.2 SLU 系统中 DataFeed 实现思路

通常可信的数据源来源于被社会公认的机构，例如社会的消费者价格指数(CPI)和某一个重大比赛的结果。我们可以通过这些机构的网络服务来使用相应API接口，获取相应数据。比如通过HTTP的请求。

在SLU系统中的Data Feed，我们将引入博弈论的设计理念，通过不同数据源的Deposit来作为担保条件，并对诚实可信的数据源进行 reward。当有多个数据源的引入的时候，将通过对数据预先设置好的数据共识规则，来进行数据的处理。

在SLU系统中，Oracle data可以是Json file的形式在创建和存储。

Oracle data file 的一个例子: {
rate : {
“USD/EURO” : “1.10351”, “USD/GBP” : “1.31930”, “USD/JPY” : “0.00954”
} }

比如这是一个智能合约引用的利率的例子。在智能合约中可以被引用:

```
function unlock(oracles, inputs) {
if (oracles.currency.rate[ “USD/EURO” ] > 1.10) { // make decision #1
} else {
// make decision #2
}
}
```

考虑到安全因素，智能合约不会直接调用外部的数据，在智能合约读取数据之前，Oracle需要把相应数据写入到公开账本中。

Oracle的创建

我们可以通过Oracle的“创建交易”，来创建Oracle，在SLU系统中分为以下两个步骤

1. 通过OracleCreat Transaction，发起一笔交易;
2. 把数据写入到公开账本。

Oracle 创建交易

```
{
 “id” : “hash of transaction, excluding signatures”,
 “transactionType” : “oracleCreateTransaction”,
 “oracleName” : “nameOfOracle”,
 “oracleProviderKey” : “MyuemkT4raRPPjhDwd9MyzbBNt9QAwKHc”,
 “oracleId” : “oracle_MyuemkT4raRPPjhDwd9MyzbBNt9QAwKHc”
 “creation-payment” : “transactionId”, “name-payment” : “transactionId”,
 “fee-collection-recipient” : “@oracleProviderAccount”, “dataHash” :
 “MyuemkT4raRPPjhDwd9MyzbBNt9QAwKHc”,
}
```

在Oracle的创建交易中，我们会制定一些参数，包括TransactionType和Oracle ID和DataHash等等。

6.3 随机数方案

公平的区块链合约需要良好的随机数，之前社区有考虑过使用比特币的每个区块的块头和区块头部的Hash值来构建随机数，但是当利益足够大的时候，矿工可以选择欺骗。例如当一个竞猜合约的结果取决于随机数的时候，并且如果竞猜合约的价值大于12.5个Btc，那么矿工可以选择不报块，使得竞猜结果有利于自己。另外比特币网络的出块时间的不稳定性，也造成了构建随机数过程中的困难性。

其他的随机数方案包括两阶段公布随机数结果。比如在一个选举过程中，需要随机抽取审计员，第一阶段:竞选委员会可以随机选取一个号码，并公布相应的Hash值。第二阶段:他们公布相应的随机数，并把该区块的区块数和该随机数利用，并产生新的随机数。但是如果竞选委员会不公布随机数，就会造成结果的无效性。

在RanDAO中，可以通过两阶段的交互产生较好的随机数，并且引入的激励机制，激励随机源成为诚实节点。

通过利用承诺协议(Commitment Protocol)和多阶段交和博弈机制的引入，我们可以产生较好的随机数，来保证合约的公平性。

七、Identity and Privacy

SLU系统将通过智能合约管理SILUBIUM平台的上的用户SLU系统将供可选的身份识别模块，Identity是区块链系统可以对接金融系统的前条件。

在SLU系统中，我们将区分Identity客户和非Identity客户。

SLU系统开发者将开发基于相应的Identity智能合约代码，并把代码开源给第三方。通过第三方征信机构的引入，在SLU系统中，通过Identity智能合约验证的客户将会拥有更多的优先级。

例如在SLU系统中的面向金融服务的DAPP中，Identity验证的客户，将获得更多的权限。

关于Privacy，因为SLU系统兼容UTXO模型，ZeroCoin为基于UTXO模型的加密传输协议，目前Zcash已经处于公开版，我们将通过与Zcash协议的融合，给SLU系统中的智能合约和交易供更多的隐私保护。

八、SLU系统的扩容方案

在比特币诞生之初，比特币的创始人中本聪并没有特意限制区块的大小，区块最大可以达到32MB。当时，平均每个区块大小为1-2KB，有人认为区块链上限过高容易造成计算资源的浪费，还容易发生DDOS攻击。因此，为了保证比特币系统的安全和稳定，中本聪决定临时将区块大小限制在1MB。那时比特币的用户数量少，交易量也没有那么大，并不会造成区块拥堵。2013年至今，比特币价格直线飙升，用户越来越多，比特币网络拥堵、交易费用上升的问题逐渐涌现出来。BCH团队已经给比特币“扩容”，即通过修改比特币底层代码，从而达到提高交易处理能力的目的。SLU经过优化比特币的代码，选择更大的区块容量32M的区块，在保证交易安全性的同时，提高SLU交易的及时性。

第三部分　SILUBIUM应用 Applications

一、去中心化应用(DAPP)

SLU系统致力从技术层面全面支持去中心化应用，尤其是通过移动端策略的引入，将不同的DAPP想法产品化，使普通互联网用户可以真正感受到区块链技术带来的价值。

面向不同行业的DAPP应用，可以把区块链技术带给更多的用户和行业。例如去中心化社交、去中心化的存储和去中心化的域名服务、去中心化的计算服务等，通过激励机制的引入，将更深层次利用共享经济的理念，改变现有的APP市场和商业模式。

区块链技术为搭建去中心化应用(Decentralized Applications) 提供基础架构。在SILUBIUM中，通过完善的SLU API的设计和Docker的分发，简化开发者的准备工作，使开发者可以快速上手相

应的开发工作。并将通过SLU系统内部的Token激励开发者开发出高质量的DAPP。

二、多个行业的支持(IndustryOriented)

在SLU系统中，通过不同共识机制的引入和监管的需求，可以为行业发展需求也提供支持。

例如SLU系统中，提供的基于 Proof of Time 和 Raft 协议融合的共识机制，可以满足可信网络中，对区块链速度和容量的要求，通过基于区块链技术的主控合约和Oracle和Data Feeds的引入，也可以引入更多线下的因素。通过Identity和Privacy的设计，可以符合金融行业的监管需求。

在SLU系统中，可以支持多个行业的应用需求:例如金融业、物联网、供应链、社交和游戏、慈善、数字资产和股权等。另外基于SLU的智能合约和主控合约，通过图灵完备的编程语言，可以实现更复杂商业逻辑的支持，并将支持更多的行业。

三、移动端策略(Run Mobile)

“Run Mobile”是SILUBIUM开发团队的一个时刻谨记的原则，面向移动端策略也是推动区块链技术落地的一个重要环节，在SILUBIUM的生态系统中，我们不仅全面支持并推动移动应用战略，而且我们将会与第三方开发者，一起为用户供移动端的服务，包括:移动端钱包、移动端Dapp应用、移动端智能合约应用等服务。我们也鼓励第三方的开发者，加入我们，一起推动区块链技术的落地，开发出普通互联网用户可以使用的区块链移动端服务。

四、SILUBIUM & Silktrader的行为挖矿与销毁

4.1行为挖矿

前两周会将平台总收益的100%作为分红奖励（后续会调整为80-90%），奖励给持有SLU的用户。全天24小时，每小时整点对持SLU账户进行快照，统计每账户应得分红额度，并于次日15:00（新加坡时间）陆续将分红奖励发放至SLU持有者账户，发放时间持续约2~4个小时。

分红规则：

1. 分红机制为SLU持有者权益，仅SLU持有者可得分红
2. 每自然天为一个分红周期，平台按周期发放分红
3. 每自然天24小时，每小时整点快照一次，计算应发分红，累计24小时，次日一次性发放
4. 分红所得币种以平台实际获得手续费为准

   *仅限流通的SLU方可参与分红，未解锁部分不参与

计算公式：

每百万份SLU分红=（平台当日总收益*100%/SLU总流通量）*1,000,000

矿工当日分红收入=SLU持有量/SLU总流通量*平台当日总收益*100%

挖矿详细规则

在Silktrader进行交易即被视作挖矿，交易用户即被视为“矿工”，挖矿产出物为平台币SLU。

您仅需在Silktrader交易，产生的交易手续费会折算为SLU，于次日0：00按小时陆续发放前一日同时段收益至您账户。

挖矿细则：

1. 每小时整点统计用户产生的交易手续费
2. 按统计时段的SLU均价，将交易手续费100%折算为SLU数量并累计
3. 于次日0：00按小时陆续发放前一日同时段收益至用户Silktrader账户
4. 其他在Silktrader操作的挖矿奖励（如邀请、签到等）

SLU均价计算公式：

SLU均价=统计时段内平台SLU总成交额/统计时段内平台SLU总成交量

4.2 SLU销毁

SLU被用作SilkTrader的平台币。在提币、用户发布临时广告时，都需要使用一定价值的SLU；SilkTrader平台游戏最终盈利的15%将用于回购并销毁SLU，销毁地址为：SLUNTjasQ36opbVgh1vMdUGTAuv7dF9rfB5c，用户可随时查询此公开地址，以接受全社区成员的监督。

五、资产交易结算

SILUBIUM是一种服务于“一带一路”的国际型数字资产，是一种前瞻性的金融投资全球化解决方案，主要面向全球进行数字资产配置交易和基建、民生领域的跨境跨资产投资结算。SILUBIUM主要涵盖了数字资产基础设施建设、数字金融平台、区块链应用生态三大产业布局，并通过“安全+、收益+、自由+”三大价值体系，实现数字资产的良性配置，旨在让数字资产从线上走到线下，为人类社会无障碍交流和经济全球化建设做出应有的贡献。

在交易过程中，数字签名是解决方案的一部分，但倘若仍然需要一个受信任的第三方来阻止重复支付问题，其主要优势将丧失。问题与比特币及其他加密货币一样，人们通过点对点网络解决重复支付问题。对于SILUBIUM而言，网络“时间戳”是通过基础区块链技术完成的，并不完全需要挖矿作业的工作量证明（POW），其结果是与成为另一种提供工作量证明或权益证明的试验性交换系统相反，对消费者而言，SILUBIUM是一种实用且可靠的价值交换手段。这项技术正在逐步成熟，SILUBIUM将使用闪电网络实现极高的交易速度。SILUBIUM系统还计划使用SILUBIUMATM机、刷卡机以及硬件钱包作为用户工具并在全球推广。SILUBIUM系统还会在全世界各地广泛联系开发者共同完成未来发展中出现的各种需求，为全世界的贸易结算及跨境资金流动提供优异的解决方案。

解决方案：微信支付和支付宝之类的支付平台尽管使用广泛，但仍然完全依赖金融机构作为受信任的第三方处理电子支付。此类系统，诸如通过完全受控的银行系统进行法定货币的数字交易，作为仅有的选项已被广泛接受，但仍然饱受基于信任模型的天生缺陷的困扰。即便是向境外以及向其他地方转移现金的过程中所使用的非法的“地下通道”，信任仍然是这类交易能够实现的根本性

因素。此外，此类系统仍然需要金融机构作为调解方。这种调解通常并不值得大费周章，对于小额交易以及临时交易尤其如此。在交易可能撤销的情况下，双方对于信任的要求提高，用户不得不对其客户加以提防，从而要求客户比其他情况下提供更多的信息。假设一定比例的欺诈是不可避免的，尽管可以通过交换实物货币避免此类损失及支付的不确定性，在没有受信任方的情况下，借助通讯手段完成支付的机制并不存在。电子支付系统基于密码证据而非信任，因此允许任何两个自愿的当事方在没有可信任的第三方的情况下进行直接交易。SILUBIUM的运行方式与比特币以及其他优秀的币种类似相同，但有自己足够多的优点。计算上不可能逆转的交易方式可以保护卖方免受欺诈，例行的托管机制则可以方便　地应用以保护买家。同样地，重复支付问题则可通过当下流行的点对点分布式时间戳服务器生成按时间顺序排列交易的计算证据来解决。

交易机制一个单位的SILUBIUM可以理解为一个数字签名链。所有者通过数字签署前一个交易的哈希和下一个所有者的公钥将此类单元交易至下一个所有者。这些签名将被附加在该单元之后，可由接收者验证所有权。交易公开发布, 参与者通过一个系统能够就某个接收顺序的历史达成一致。发起交易哈希签署过程下一个所有人的公共密匙交易历史验证过程接收人。

附加值直接定位为“结算币”，除了通过相关市场推动全球的贸易结算使用以外，还兼顾消费功能，支持个人用以消费以及提供资产在全球的安全自由流动解决方案。鼓励市场将其作为现金使用。SILUBIUM用户可享受极低的交易费用，大额交易还可按升序额外享受百分比折扣。SILUBIUM的交易速度极快，通常情况下2-60秒内即可完成，完全满足跨境消费结算的需要，闪电网络技术的成熟运用，会使交易结算速度做到完全及时。在交易手续费上公平低价设计，尽量少的减少交易成本。

第四部分　SILUBIUM使用情况及里程碑

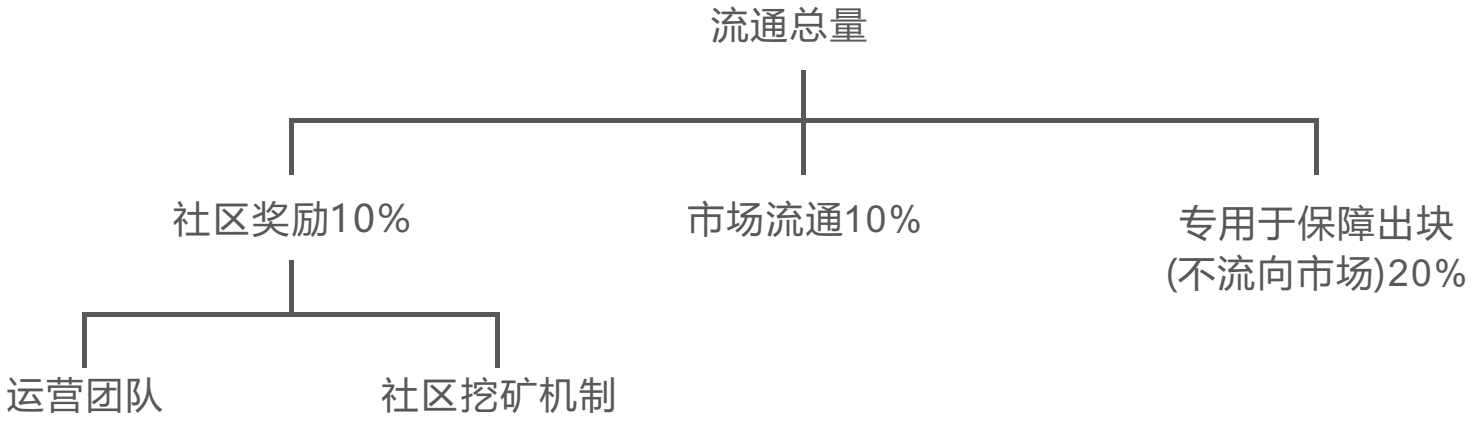
一、关于SILUBIUM的发行

鉴于SILUBIUM具有成熟系统的技术支持能力以及遍布全球的业务落地资源 ，在项目推广的早期阶段，SILUBIUM就吸引了人们巨大的兴趣。SILUBIUM应用合作伙伴不断增多，对持有SLU的刚性需求不断增大。

二、目前SLU最新使用情况如下：

(1) SLU总量：1亿枚（永不增发）

(2) 流通总量：40%



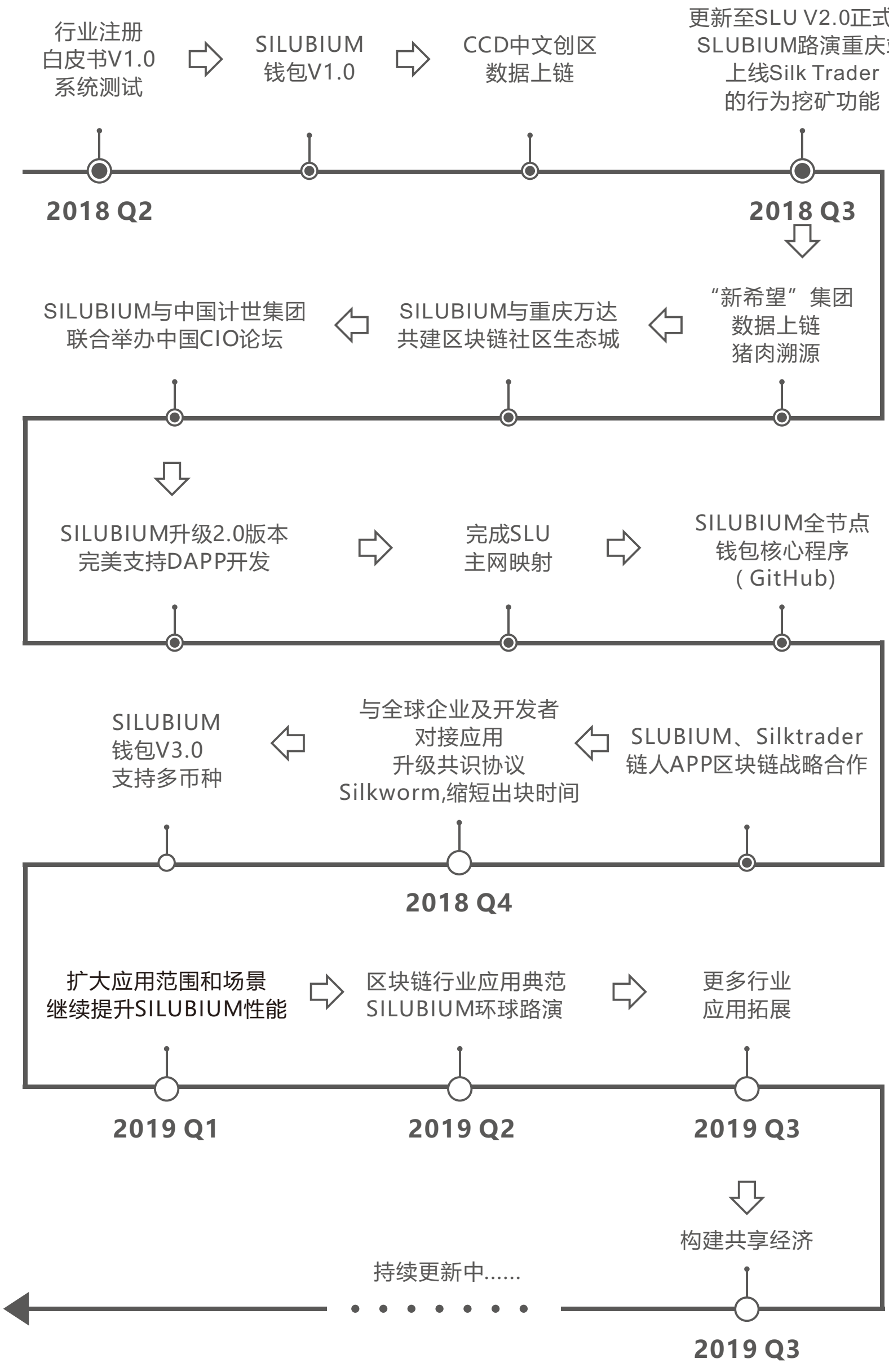
(3) 冷钱包锁仓：60%

锁仓期自2017年1月至2020年。从2020年开始，每年按SLU总量的5%逐步释放。锁仓情况可通过SLU官方浏览器查询（https://silkchain2.silubium.org),
锁仓地址如下：

锁仓地址	锁仓金额（万个）	释放时间
SLUaAVmdobxB6uZ54cos34b7oaWxDbWgTM71	300	2020年
SLUbrHNARrb5TRaVAdy1wcFWnBR6fgMGoQET	200	2020年
SLUbW9vt8FxQ3Sa1FXNkP2hLZ65ohJsZxfvb	300	2021年
SLUcVtPxkuAacRcXhZZZvzNLYYyBQSEV8tRG	200	2021年
SLUcy7NmehL12cjv6E2CR9qVkLNG8nQDPcYQ	300	2022年
SLUcZAZqAyiWm2k2c7XbWgYzwsv4iTm7EAtC	200	2022年
SLUdHfmhYvekf9uqRgMvndM6pA34KYgrWkNh	300	2023年
SLUdN6LktL4Vfv3FTntGeMXqQhqjAv7TkGwN	200	2023年
SLUfBaM2d4e4aHowkWzNamQewAqpsJVKFqSk	300	2024年
SLUfidpAkzcEoB5TYYRf9Ui2eCK63Ux7C7F1	200	2024年
SLUhFd7fdQj5Wx29h3okCbEAiThXFhmyaNnQ	300	2025年
SLUhQ9fASLavrA84y6v4x4kthMKzU5w7cV2B	200	2025年
SLUj3bhFimMunFWhfeRMW4Gp6ePouRbYy2HQ	300	2026年
SLUjgz6B3eioMFQWG645GV4iCA85dmAKci4d	200	2026年
SLUk3yMp41nXXrXAK95stkXRu9RJR3dCyCnF	300	2027年
SLUNbt4hN3zwGzsVGGLypkmfeDxWcKQU2xzs	200	2027年
SLUQxeFzzZ7U4y6JFT9t4hDeCyMTkWvAeYeu	300	2028年
SLURbascjJpQ75fgsYny2FHC5PPL4i3S3PQM	200	2028年
SLUSdK76w3AZVysoJoSBU7VnGXFSsN9xniGk	300	2029年
SLUUXV1ahes5wcfpB79GmqornmRuh4Pmzowb	200	2029年
SLUW28Pbfqckh1xU7ANMshZZ4A7os3N3P6iX	300	2030年
SLUXJyGTCTZS66nWqRyWQgDcsY9CU4Z5Yh73	200	2030年
SLUZmvfiwEy24MrWP4sbEEHGTwuQ9VN2YYS7	300	2031年
SLUZoyZHz2ENAuTF42AQbkmJ6nNAf9mJgDYS	200	2031年

合计：6000万SLU

三、SLU里程碑


行业注册
白皮书V1.0
系统测试
SILUBIUM
钱包V1.0
CCD中文创区
数据上链
更新至SLU V2.0正式版
SLUBIUM路演重庆站
上线Silk Trader
的行为挖矿功能
2018 Q2
2018 Q3
“新希望”集团
数据上链
猪肉溯源
SILUBIUM与重庆万达
共建区块链社区生态城
SILUBIUM与中国计世集团
联合举办中国CIO论坛
SILUBIUM全节点
钱包核心程序
(GitHub)
完成SLU
主网映射
SILUBIUM升级2.0版本
完美支持DAPP开发
SLUBIUM、Silktrader
链人APP区块链战略合作
与全球企业及开发者
对接应用
升级共识协议
Silkworm,缩短出块时间
SILUBIUM
钱包V3.0
支持多币种
2018 Q4
构建共享经济
更多行业
应用拓展
区块链行业应用典范
SILUBIUM环球路演
扩大应用范围和场景
继续提升SILUBIUM性能
2019 Q3
2019 Q2
2019 Q1
持续更新中.....

四、 部分合作伙伴

五、 其他

5.1 参考文献:

(1) https://en.bitcoin.it/wiki/Category:History

(2) https://panteracapital.com/wp-content/uploads/The-Final-Piece-of-the-Protocol-Puzzle.pdf

(3) https://github.com/bitcoinbook/bitcoinbook

(4) https://github.com/ethereum/wiki/wiki/White-Paper

(5) S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system, 2009, https://www.bitcoin.org/bitcoin.pdf

(6) N. Szabo, Smart contracts, 1994, http://szabo.best.vwh.net/smart.contracts.html

(7) N. Szabo, The idea of smart contracts, 1997, http://szabo.best.vwh.net/idea.html

(8) Bruce Schneier, Applied Cryptography (digital cash objectives are on pg. 123)
Crypto and Eurocrypt conference proceedings, 1982-1994
David Johnston et al., The General Theory of Decentralized Applications, Dapps, 2015, https://github.com/DavidJohnstonCEO/DecentralizedApplications

(9) Vitalik Buterin, Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform, 2013, http://ethereum.org/ethereum.html

(10) Paul Sztorc, Peer-to-Peer Oracle System and Prediction Marketplace, 2015, http://bitcoinhivemind.com/papers/truthcoin-whitepaper.pdf

(11) PriceFeed Smart Contract, 2016, http://feed.ether.camp/
(12) V. Costan and S. Devadas, Intel SGX Explained, 2016, https://eprint.iacr.org/2016/086.pdf
(13) E. Shi. Trusted Hardware: Life, the Composable Universe, and Everything. Talk at the DIMACS Workshop of Cryptography and Big Data, 2015Ahmed Kosba et al., Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts, 2016, https://www.weusecoins.com/assets/pdf/library/Hawk%20-%20The%20Blockchain%20Model%2 0of%20Cryptography%20and%20Privacy-Preserving%20Smart%20Contracts.pdf
(14) Iddo Bentov and Ranjit Kumaresan, How to Use Bitcoin to Design Fair Protocols, 2014, https://eprint.iacr.org/2014/129.pdf
(15) Marcin Andrychowicz et al., Secure Multiparty Computations on Bitcoin, 2013, https://eprint.iacr.org/2013/784.pdf
(16) Ranjit Kumaresan and Iddo Bentov, How to Use Bitcoin to Incentivize Correct Computations, 2014, https://people.csail.mit.edu/ranjit/papers/incentives.pdf
(17) Aggelos Kiayias, Hong-Sheng Zhou, and Vassilis Zikas, Fair and Robust Multi-party Computation Using a Global Transaction Ledger, 2016, http://link.springer.com/chapter/10.1007%2F978-3-662-49896-5_25
(18) Guy Zyskind, Oz Nathan, Alex Pentland, Enigma: Decentralized Computation Platform with Guaranteed Privacy, 2015, http://enigma.media.mit.edu/enigma_full.pdf
(19) Joseph Bonneau et al., On Bitcoin as a public randomness source, 2015, https://eprint.iacr.org/2015/1015.pdf
(20) Dennis Mckinnon et al., RANDAO, 2014, https://github.com/dennismckinnon/Ethereum-Contracts/tree/master/RANDAO
(21) Dennis Mckinnon et al., RANDAO, 2015, https://github.com/randao/randao/blob/master/README.en.md
(22) Marcin Andrychowicz et al., Secure multiparty computations on Bitcoin, 2014, https://eprint.iacr.org/2013/784.pdf
(23) Arvind Narayanan et al., Bitcoin and Cryptocurrency Technologies, 2016, http://www.the-blockchain.com/docs/Princeton%20Bitcoin%20and%20Cryptocurrency%20Tech nologies%20Course.pdf
(24) Matt Springer, Is Bitcoin Currently Experiencing a Selfish Miner Attack?, 2014, http://scienceblogs.com/builtonfacts/2014/01/11/is-bitcoin-currently-experiencing-a-selfish-miner -attack/
(25) Edward Felten, Game Theory and Bitcoin, 2013, https://freedom-to-tinker.com/blog/felten/game-theory-and-bitcoin/
(26) Litecoin, 2011, https://litecoin.info/

(27) Evan Duffield and Kyle Hagan, Darkcoin: Peer-To-Peer Crypto-currency with anonymous Blockchain Transactions and Improved Proof-of-Work System, 2014, https://www.dash.org/wp-content/uploads/2014/09/DarkcoinWhitepaper.pdf
(28) Evan Duffield and Daniel Diaz, Dash: A Privacy Centric Crypto Currency, 2015, https://www.dash.org/wp-content/uploads/2015/04/Dash-WhitepaperV1.pdf
(29) Sunny King and Scott Nadal, PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake, 2012, http://web.archive.org/web/20131228174819/ http://peercoin.net/bin/peercoin-paper.pdf
(30) Novacoin, 2013, http://coinwiki.info/en/Novacoin
(31) Nxt, 2013, http://wiki.nxtcrypto.org/wiki/Whitepaper:Nxt

5.2 版本变更记录

- SILUBIUM白皮书 V3.3版本 2018年10月15日
- SILUBIUM白皮书 V3.2版本 2018年7月15日
- SILUBIUM白皮书 V3.1版本 2018年6月20日
- SILUBIUM白皮书 V2.1版本 2018年2月1日