更新日期：2019年9月，版本：5.0

Pascal：无限可扩展的加密货币

Herman Schoenfeld herman@pascalcoin.org

Albert Molina albert@pascalcoin.org

Pascal是“下一代”加密货币，设计在“无限可伸缩”的架构上。它基于一个简化的帐户模型，而不是utxo模型，并且专为超高事务吞吐量而设计，只需要在节点上保持恒定的存储。它通过引入一个新的加密数据库SafeBox来实现这一点，该数据库与工作证明区块链(可以删除它)协同工作。除了支付用例之外，Pascal还为作为Pascal的主要实际用例的第二层应用程序提供了一个可伸缩的共识和财务层.

介绍

Pascal是下一代加密货币，它解决了现有加密货币的许多限制。它的主要关注点是解决“可伸缩性问题”。testnet 4.1代码基的压力测试结果表明，mainnet的容量可以达到每秒1600个事务。

其次，Pascal通过引入可删除的区块链技术，解决了区块链长期存在的“存储问题”。该技术允许Pascal节点安全地删除超过最后100个块的区块链。通过存储事务流而不是无限的历史，Pascal网络是第一种(也是唯一已知的)加密货币，理论上可以以最大吞吐量运行无限长时间，同时只需要恒定的存储。通过这种方式，Pascal实现了“无限的可伸缩性”。

最后，Pascal的基于账户的模型极大地简化了分类账结构，为用户提供了熟悉的“银行账户”体验，同时保持了像比特币那样的分散化和无权限。这种简化的体系结构还支持将第二层协议封装在Pascal事务中，从而促进了一种新型的分散应用程序体系结构(dApps)。Pascal dApps旨在彼此独立运行，并使用Pascal进行定期财务结算和协商一致。通过这种方式，一个自然切分的第2层dApp运行时出现了，它允许dApp独立地伸缩、存储、执行和处理它们的数据和逻辑，而不影响彼此的性能(不像现有平台在第1层处理所有dApp业务逻辑).

目前，Pascal网络提供以下功能:

- 公平的经济分配(无矿前预挖或ICO)
- 5分钟封堵时间，2年减半时代，4200万枚硬币发行总量，每块封堵尾排1枚硬币
- 即时的0费用交易
- 安全的0确认交易
- 基于帐户的简单模型，使用人类可读写的帐号地址
- 协议内帐户交换和原子交换功能
- 每秒1600个事务吞吐量(TPS)
- 可删除的区块链
- 密码安全的帐户历史记录(一个DAG结构)
- 低内存ASIC和GPU抗工作证明算法
- 第二层扩展点，适用范围广.

在路线图上，Pascal的目标:

- 第1层协议以逐块粒度进行治理
- 第2层证明叠加整合
- 每秒10万 +笔交易（及更多）.

发展历史历程

Pascal最初由Albert Molina [16]于2016年推出 **版本一**.它的发行方式与比特币一样，是一种“公平发行”的货币，没有ICO或pre-mine。这个初始实现引入了一种新的加密数据结构，称为SafeBox。区块链和保险箱协同工作，以保持用户资金和数据的平衡。保险箱的功能是作为一个“分类帐余额”，它维护所有用户帐户的最新状态，而区块链的功能是作为一个“分类帐”，其中包含修改这些用户帐户的交易。虽然Pascal的初始版本允许修剪区块链，但是需要一个节点至少下载并验证一次整个链.

第二版 Herman Schoenfeld加入了这个项目，并提供了一个解决方案，使节点(以及网络本身)能够删除区块链，同时保留其聚合的工作证明安全性[14]，从而克服了这个限制。只有安全盒和最后100个块才需要完全参与网络。版本2还介绍了协议内的帐号交换、帐号名称和难度调整算法的变化。

第三版 引入了一些重要的新变化，包括50%的通货膨胀率降低[1]，20%的开发者奖励[2]来资助开发和基础设施，批量交易[3]和一个新的GPU/ASIC抗验证算法RandomHash[5]。

第四版 引入了第2层数据事务[5]、快速块传播[6]和显著的性能增强。

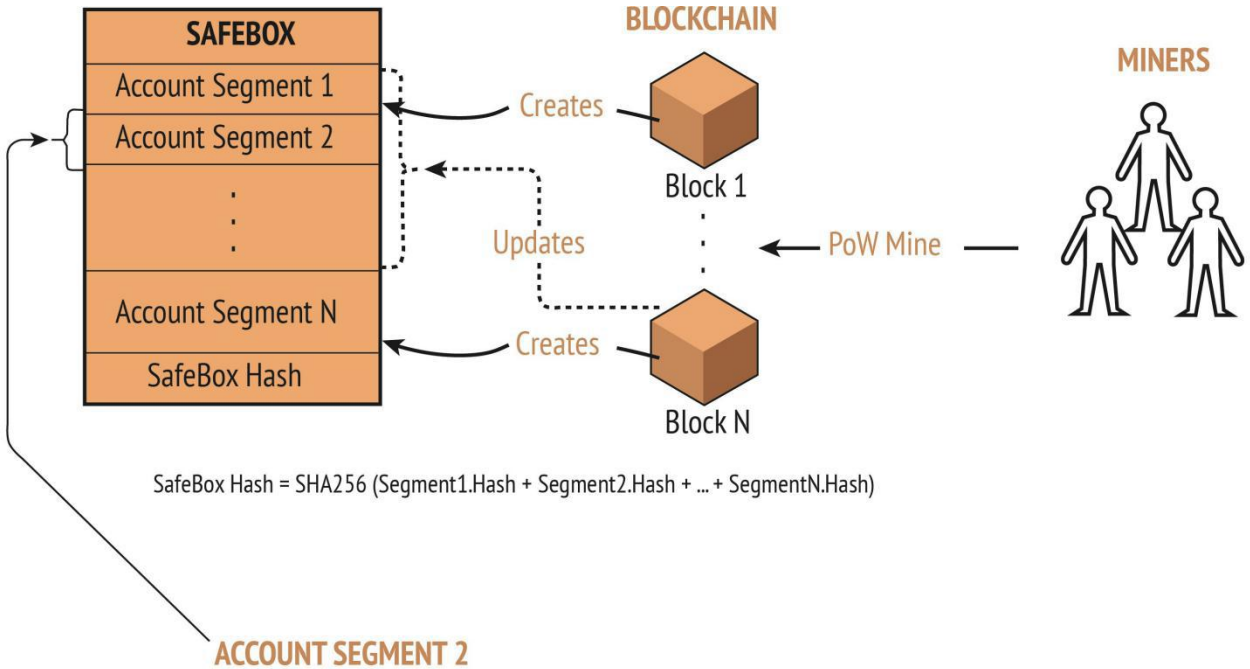
第五版 介绍了加密安全帐户历史[7]，可删除的保险箱和轻节点[8]，第二层寻址格式[9]，新的移动钱包和原子交换。此外，Pascal在这个版本中进行软品牌从PascalCoin改名为“Pascal”.

自治的社区

在第三版，将20%的块奖励添加到协议中，以资助Pascal及其生态系统的进一步发展。这些资金属于社区所有，由社区使用区块链投票方法(和社交媒体平台)拨款进行。

成立了一个临时基金会来管理这些基金，名为PascalCoin foundation Limited，2018年在澳大利亚注册。基金会只是协议产生资金的临时保管者。长期来看，这笔资金将由第二层应用程序管理，该应用程序实现了由利害关系证明覆盖网络保护的第二层协议。该协议将根据Pascal用户自己生成的第一层投票来分配资金.

LAYER-1 ARCHITECTURE



ACCOUNT SEGMENT 2

Acc. Number	Balance	Public Key	Name	Type	N Op	Timestamp	Hash
5	1231.1214	Pubkey	user23@email.com	0 - User Account	4	Unix Time	H2000
6	9.0000	Pubkey	@SlackStyle	1 - PascalChat Room	2	Unix Time	H2001
7	999.0000	PubKey	#SlackStyle	0 - User Account	1	Unix Time	H2002
8	1231.1231	Pubkey	domainstyle.com	2 - PascalShop	3	Unix Time	H2003
9	5555.1111	Pubkey	‹empty›	0 - User Account	1	Unix Time	H2004

Segment Hash = SHA256(AccountNumber ++ Balance ++ rest of fields)

SAFEBOX [保险箱]

SafeBox是由Albert Molina于2016年发明并由Herman Schoenfeld改进的加密数据结构。 SafeBox为分散式共识分类账提供了一种替代方法，可与传统的基于区块链和DAG的方法相媲美。 主要的区别因素是它的简单性.

保险箱的目的是独立于区块链存储用户帐户余额。如果区块链是“分类帐”，那么保险箱就是“分类帐余额”。安全盒的主要创新之处在于它与区块链的加密安全连接。通过单独检查安全盒，用户可以计算构建该安全盒所需的“聚合工作证明”，而不需要验证区块链。这允许网络随着时间的推移安全删除区块链，同时保留该区块链提供的历史工作证明。

从结构上讲，保险箱由一组帐户段组成。每个帐户段都是一组帐户。帐户是维护用户余额、公钥和其他数据的原子记录。当网络挖掘一个新块时，该块提交到当前安全盒并附加一个新帐户段.

区块链

就像比特币一样，金融数据的完整性是通过使用一系列链接在一起的区块来证明的。与比特币一样，这些区块也包含一系列用于改变金融状况的交易。然而，与比特币不同，Pascal块不直接引用前面的块。

相反，它引用前一个安全盒状态，该状态被压缩为一个名为“安全盒散列”的散列。在版本5中，它被称为安全盒根[8]，并被计算为帐户段的merker -Root。当一个块被“应用”到安全盒时，它会附加一个包含新帐户的新帐户段(属于矿工)。.

操作

与比特币和其他加密货币类似，Pascal中的块是用于“操作”的交易的容器。“它们之所以被称为操作，是因为它们是泛化的交易，可以执行的功能远不止在账户之间进行简单的价值转移。例如，有一些操作可以更改帐户的密匙、更改其惟一名称、将其标记为出售、执行原子交换、发送数据和批处理组合。最重要也是最常见的操作是在账户之间转移资金的交易操作.

帐户段

保险箱基本上是所有用户帐户的列表。然而，这些帐户被分组到称为帐户段的段中。每当矿工向安全盒添加一个块时，都会生成一个帐户段。换句话说，每次挖掘一个新块时，安全盒都会自动扩展一个新帐户段。

目前，一个帐户段只包含5个帐户，但在未来的协议升级中，这将成为一个协议变量[10]。SafeBox散列(或SafeBox Root[8])基本上是所有包含帐户段的聚合散列。新块必须通过引用块头中的SafeBox根提交到SafeBox.

账户

账户是保险箱的原子，是用户的数字财产。账户最初授予矿商，然后矿商通过市场机制将账户分发给用户。Pascal帐户也称为PASA。PASA包含各种字段，包括balance (PASC)硬币、公钥、惟一名称、数据字段和其他用于原子交换和协议内交换的内部状态字段.

无限扩展

在本文档的上下文中，“无限扩展”定义为“网络使用常量存储在最大吞吐量下实现无限运行时间的能力”。换句话说，它是网络永远以最高速度运行的能力，而不需要越来越多的存储设备来支持它。Pascal是唯一(已知的)能够实现无限扩展的加密货币。所有其他加密货币、区块链和dag都依赖于它们完整且永久增长的历史。作者认为，这种依赖关系是这种体系结构的“遗传缺陷”，因为就可用性而言，它给网络施加了最大的可行年龄。

例如，当区块链的大小变得不可管理时，新节点加入网络的速度不能超过

它会增长，现有的节点很难跟上并放弃网络。这导致了…的迅速集中网络对其安全性造成了完全的损害。

这个问题已经出现在其他没有建立在无限可伸缩架构之上的项目中。这样的网络很难支持它们的(相对较低的)需求，常常会出现网络中断、孤儿率升高以及偶尔(无意地)出现的网络拓扑结构分裂。最终，这些网络将完全无法使用。

显然，加密货币网络的长期生存依赖于“无限可伸缩”的体系结构，这是Pascal首创的(专门)。在用户需求比当前加密货币市场高几个数量级的全球采用场景中，将强制使用无限可伸缩的体系结构.

PASCAL如何实现“无限扩展”？

简单地说，由于安全盒包含帐户余额，Pascal中的块在超过100(检查点)的高度时被删除。因为新的块将被添加到链的顶部，而旧的块将从底部删除，所以任何时候都只需要固定数量的块(以及安全盒)。

检查点只是当前提示块的第100个块。如果tip块是200，那么检查点块就是100。如果tip块是201，检查点块就是101。当一个新节点加入网络时，它只需要最新的安全盒和检查点到提示的最后一个块。

通过选择(并验证)具有最聚合的工作证明的安全盒，节点始终可以选择正确的安全盒。能够在不需要区块链的情况下验证整个区块链的聚合工作证明，这是Pascal的主要创新，也是可以删除Pascal区块链的原因.

事务吞吐量

Pascal目前在生产网络上每秒可以完成1600个事务。与比特币的5个TPS和以太坊的15个TPS相比，Pascal已经在这一领域进行了几个数量级的创新。Pascal的最初目标是实现签证级别的可伸缩性。VISA网络平均执行1700个TPS。在高峰时期，这个数字可以达到70000 TPS.

Pascal已经实现了visa级的可伸缩性(平均而言)，这是一个经验事实.

虽然其他项目也做出了类似(甚至更大)的声明，但是他们还没有在他们的生产网络上展示这种能力。此外，他们对LevelDB的依赖也让人对他们的说法产生了怀疑，因为LevelDB只能实现300 TPS。Pascal使用Albert Molina开发的自定义存储数据库，该数据库是为特定目的而构建的，因此性能优越。

展望未来，Pascal路线图包括一些基本增强，可以将吞吐量提高到100,000 TPS或更高。由于Pascal网络的有机需求远远低于它目前所能支持的水平，因此开发人员保留在适当的时候公开该技术的权利，以便将技术泄漏到其他项目的可能性降到最低.

RANDOMHASH2:GPU & 抗ASIC的工作证明

Pascal的区块链由一种低内存、GPU和抗ASIC的哈希算法RandomHash2[11]保护。RandomHash2是一种“高级”哈希算法，它以随机、非确定性的方式组合了77种著名的加密哈希算法。该算法具有内存小、串行性强、递归性强、分支多、执行决策范围广等特点。因此，它对于基于gpu的处理本质上是低效的，并且由于其复杂性，对于ASIC实现来说在经济上是不可行的。

RandomHash2(和RandomHash[4])的主要创新之处在于它在挖掘过程中的使用方式。该算法的设计使得nonce的求值依赖于随机相邻nonces的部分求值。通过在算法中注入这种非强制依赖关系，RandomHash2允许串行挖掘器(CPU)的挖掘速度明显快于并行hasher (GPU)，因为最优非强制集只能在运行时枚举，而不能为并行计算预先排序。因此，并行散列器(GPU)需要为所有轮执行完整的工作负载，而串行散列器(CPU)可以重用前几轮中部分完成的nonce计算。在RandomHash2中，这种“CPU偏向”优化使CPU的哈希率提高了500%.

算法概述:

1. 散列一个nonce需要n个迭代（称为级别），这些迭代是递归计算的;
2. N在MIN_N=2和MAX_N=4之间(含)以不确定的方式每一次变化;
3. (1)中的每一层也依赖于相邻的J个nonces，它们是随机和非确定性确定的;
4. 值J被限制为MIN_J=0和MAX_J=4;
5. 每个级别从77种著名的加密哈希算法中选择一个随机哈希函数;
6. 在一个级别上进行哈希处理的输入摘要哈希是对其所有先验哈希输出的传递闭包的压缩层(1)和它相邻的nonce的先验层(3);
7. 每个级别的输出都扩展为(低)内存硬度;
8. 随机是使用梅森捻线算法生成的;
9. 随机性总是使用加密哈希算法输出的最后一个DWORD进行播种;
10. 第一个输入使用SHA2-256进行预散列;
11. 最后一个输出使用SHA2-256进行后哈希;

地址空间的商品化

在几乎所有其他加密货币中，新用户可以随意为自己创建一个新地址。这创建了一个近乎无限的地址空间，它可以快速地膨胀区块链，而没有相应的用户基础或需求来证明它。无论是恶意的参与者还是良性的参与者都可以利用这种能力实现区块链/DAG的无限制增长.

如果地址空间是有限的，那么它就变成了一种有限的资源，可以被商品化。这就是Pascal Accounts (PASA)的经济学原理。在Pascal语言中，账户数量有限，由矿工铸造而成。然后，矿商通过市场机制将这些账户分发给用户。Pascal包括协议内的PASA市场，允许用户之间买卖帐户。支持私人和公共销售模式。在版本5中，帐户(或硬币)的原子交换可以通过相同的方式执行

商品化的地址空间创建了一种自然的节省空间的机制，因为底层存储没有乱放不需要或使用过的密钥。它还能抑制垃圾邮件发送者、滥用或冷淡地使用安全盒。最重要的是，地址空间的商品化是Pascal无限可伸缩体系结构的一个基本方面.

为全球采用扩展地址空间

对于商品化的地址空间，一个常见的批评是无法立即搭载“数十亿用户”。虽然从理论上讲，比特币可以在一夜之间在全球人口中传播，但由于网络的交易吞吐量有限，这些用户将无法使用这些地址。这些用户中只有一小部分能够使用网络，如果有的话。

相反，Pascal的网络允许全局事务吞吐量，但矛盾的是，它缺少覆盖全局人口的地址。目前，有超过100万个账户，其中大部分是未使用的。Pascal每分钟向保险箱添加一个新帐户。按照这个速度，需要14000年才能产生足够的地址来覆盖目前的全球人口。显然，需要一个解决方案来根据实际需求扩展地址空间。

解决这个问题的方法是增加每个块创建的帐户数量。块策略[10]提出了一种协议内治理模型，允许现有用户管理每个块创建了多少个帐户，范围从0到4095。这个值可以根据块的不同而变化。因此，在大量采用期间，社区可以将新帐户的创建限制在每天最多110万个帐户。当脉冲减小时，可以将其节流，以减少存储。

这个块策略建议是协议的第6版。应该指出的是，目前，帐户是完全丰富和几乎免费。Pascal开发人员(和社区)致力于确保真正的新用户能够以接近0的成本获得帐户.

可靠的零确认交易

为了方便商家采用，Pascal实现了一种适用于日常商业的可靠的0-confirmation方案。通常，在像比特币这样的加密货币中，接受未经确认(0-confirm)交易的商家面临双重支出攻击的风险。

当一个买家用一笔交易购买了一件商品，但在商家接受付款后，他却偷偷地把这笔钱花了两倍，这时就会发生“双花攻击”。为了实现这一点，支付和双消费事务需要几乎同时发布，但要在地理位置稀少的地方发布。这样，商家节点首先检测“支付”，而矿商节点首先检测“双消费”。商人接受了0确认，但后来才知道，当矿工开采下一个区块时，它被重复使用了。

Pascal通过引入Double-Spend Alerts (DSA)解决了这个问题。DSA是网络消息，警告其他节点有重复开销。它包含提交double-spend的帐号和double-spend事务的副本。当一个节点检测到一个双开销事务时，它向所有已知的对等节点发布一个DSA.

想要接受0-确认交易的商家只需要再等待2-3秒，并收听包含买家账户的DSA。如果在此期间没有检测到DSA，则几乎可以保证在未来的块中生成0确认支付事务。

这是可行的，因为Pascal使用简化的帐户模型，而不是复杂的utxo模型(如比特币)。因此，对于网络中的任何节点，检测重复开销都是一件微不足道的事情。处理DSA的计算成本很低，商家可以通过使用帐号过滤无关的DSAs。

攻击者剩下的唯一选择是与一个采矿者串通，秘密地挖掘双花事务。由于这类攻击属于“工业规模”，它超出了普通小偷的范围，但却在普通商人的风险承受范围之内。因此，使用DSA的商家可以可靠地接受每天的0- validation。然而，与往常一样，涉及大量的事务应该总是等待相当数量的块确认。.

零费用交易

pascal的另一个独特特性是允许用户在每个块上进行一次免费事务（即每5分钟一次）。这给了用户每天288笔免费交易，一个非常合理的数字。一致性规则只是强制一个公钥在任何时候都可以在一个块（或内存池）中拥有最多一个零费用事务。这种简单的方法解决了事务垃圾邮件问题，同时为用户提供了一个非常合理的数量，每天免费交易。

如果一次交易在5分钟内超过一次，则需要非常低的费用（当前设置为0.0001帕斯卡）。零费用交易是基于pascal超高吞吐量的优点实现的，在这种情况下，交易优先级的费用竞价竞争（在其他utxo加密货币中是典型的）是没有意义的。随着当前吞吐量接近最大吞吐量，收费压力才开始上升，这与比特币收费市场的运作方式类似。然而，由于Pascal目前的体系结构实现了Visa规模的吞吐量（平均值），而且路线图打算提供更高的吞吐量数量级，因此对于不经常使用的情况，交易费用始终预计为零。

零费用交易仅适用于“普通”交易，即不用于第二层目的的交易或

批量交易。此类交易将涉及微不足道的费用.

帐号、名称和类型

pascal通过允许用户以与传统银行业务相同的方式往来账户（pasa）来促进用户之间的价值转移。然而，与大多数加密货币不同，Pascal帐户是人类可读写的数字（如1234-56），而不是必须复制粘贴或扫描的复杂字符串。

pascal的一个关键特性是，帐户可以具有公开可见的唯一名称，这与域名系统（dns）的方式非常相似。这允许用户在电子邮件地址、社交媒体名称、业务和品牌名称等之间收发资金。付款仍然是通过他们的号码来指代帐户，但发件人使用该名称以不受信任、不受许可的方式查找收件人，就像n从域名中查找IP地址.

此外，Account有一个Type字段，其数字范围为0到65535，Data字段包含32个字节。 这些字段在第2层用例中起着根本作用。 例如，类型为“12356”的帐户可以（通过社交惯例）同意引用“PascalChat应用程序”。 帐户名称可用作“聊天室名称”。 帐户的交易可以解释为聊天消息及其包含“聊天消息”的有效负载。 更复杂的应用程序可以通过简单地使用帐户数据作为包含聊天消息的第2层数据库的“状态哈希”来完全脱离聊天消息。

这允许帐户充当第2层应用程序的共识机制。 这种共识机制
稍后讨论.

加密的完整性

尽管节点不需要存储完整的区块链，但是安全盒保留了完整区块链的密码完整性。这是因为SafeBox包含用于在该块创建的每个帐户段中构造该SafeBox的所有块头。每个块头都对前一个安全盒状态(即在那个时间点上所有帐户的状态)和前一个块头进行哈希承诺。通过这种方式，通过块标头中的目标难点来保存状态及其演化;因此，用于演化这种状态的总功可以通过检查安全盒本身来计算，而不需要任何块。节点将只采用工作最多的安全盒.

重新组织检查点

Pascal节点只需要保留最后100个块。这意味着在100个块之前的SafeBox状态无法在事务级别进行验证。假设由于SafeBox是“最有效”的，因此最后100个块之前的帐户状态是有效的。仅必须验证最后100个块内的操作（ 事务）。

这引入了一种“状态攻击”场景，其中恶意玩家在较长时间内使用多数哈希值秘密地挖掘损坏的SafeBox。在100个区块之后，恶意矿工将损坏的SafeBox作为真正正确的SafeBox发布到网络。由于损坏的SafeBox比当前诚实的SafeBox具有更多聚合的工作证明，并且损坏的SafeBox的最后100个块内的所有事务都是有效的，诚实的网络被欺骗接受损坏的SafeBox。

要解决此问题，Pascal会对网络施加100块重组组织限制。这意味着节点将不接受新的SafeBox，除非它从最后100个块中的当前SafeBox分叉。以这种方式，状态攻击情形被阻止，因为恶意矿工将无法将其损坏的SafeBox传播到诚实网络。

其他加密货币也成功地采用了类似的方法。例如，比特币现金采用10块重组组织限额。在许多人看来，超过10个街区的重组是其他严重问题的原因。因此，虽然这种做法在某些人看来可能会引起争议，但对其他人来说却是无关紧要的。但是，Pascal开发人员还在研究使用涉及加密安全帐户历史的欺诈证据来解决状态攻击的替代方法。如果通过这种方法出现解决方案，可以在以后安全地更换重新组织检查点.

私人交易

由于Pascal的设计，往来账户的交易很容易被任何人审计。对于许多用例来说，这是一个很好的特性，因为它使得审计、调节和会计非常简单。然而，pascal也致力于为那些隐私非常重要的用例提供同样的匿名特性。因此，pascal有一个强大的隐私路线图，已经部分推出。

在pascal版本1中，用户可以使用pasa交换方法私下传输pasc。该帐户本身将交换所有权，而不是将pasc发送到彼此的帐户。因此，只要密钥交换是通过安全通道传输的，就不可能跟踪pasc的支付。

pascal的版本3增加了协议pasa和pasc的混合功能。这种能力将允许未来推出类似于比特币现金开创的coinshuffle[12]/cashfusion[13]的协议内翻滚工作流。通过这种方法，选择进行私人交易的用户只需支付一笔边际费用，而在幕后，钱包将使他们的交易变得模糊，无法重建。它通过在许多节点之间启动一个网络范围的工作流来实现这一点，这些节点共同创建一个大型事务，发送PASC并在各种帐户和密钥之间交换PASA。由于产生的事务涉及许多参与者，因此

谁知道对方的任何信息，原来的发送者、接收者和金额都是不可逆转的模糊。通过快速连续地将这些事务链接在一起，基于模糊处理的匿名性以很小的额外成本增加了几个数量级。

此外，节点将能够通过提供其潜在的pasc和pasa用于这种混合事务来赚取费用。这使得他们可以赚取剩余收入，同时为网络提供一组丰富的pasc/pasa来参与非对称性。这将为pascal用户带来快速、廉价和无缝的匿名性。路线图还包括1层零知识证明和2层dapp的研发建议，以实现匿名性的其他途径.

原子互换

原子交换是一种智能合约技术，可以在不使用中间人（如第三方交换）的情况下，将一种加密货币交换给另一种加密货币。原子交换以加密安全的方式在各自的区块链内作为智能合约事件发生。原子交换是dex能力的基石。

pascal版本5通过协议内帐户交换机制引入原子交换功能[15]。一个账户可以被时间锁定和散列锁定给一个交易对手，只要在时间锁定期内解除散列锁定，该交易对手就可以选择取得该账户（或其中的硬币）的所有权。.

COINROT AND 帐户恢复

硬币腐烂是由于丢失/损坏的密钥和/或自然所有者死亡而导致硬币永久丢失的现象，并且已经成为困扰加密货币的一个值得注意的问题。例如，据估计，大约20%的BTC已经“腐烂”。Pascal解决了这个问题，它允许矿工在4年后收回一个不活跃的帐户。如果帐户在4年内没有发生任何状态更改，则视为不活动.

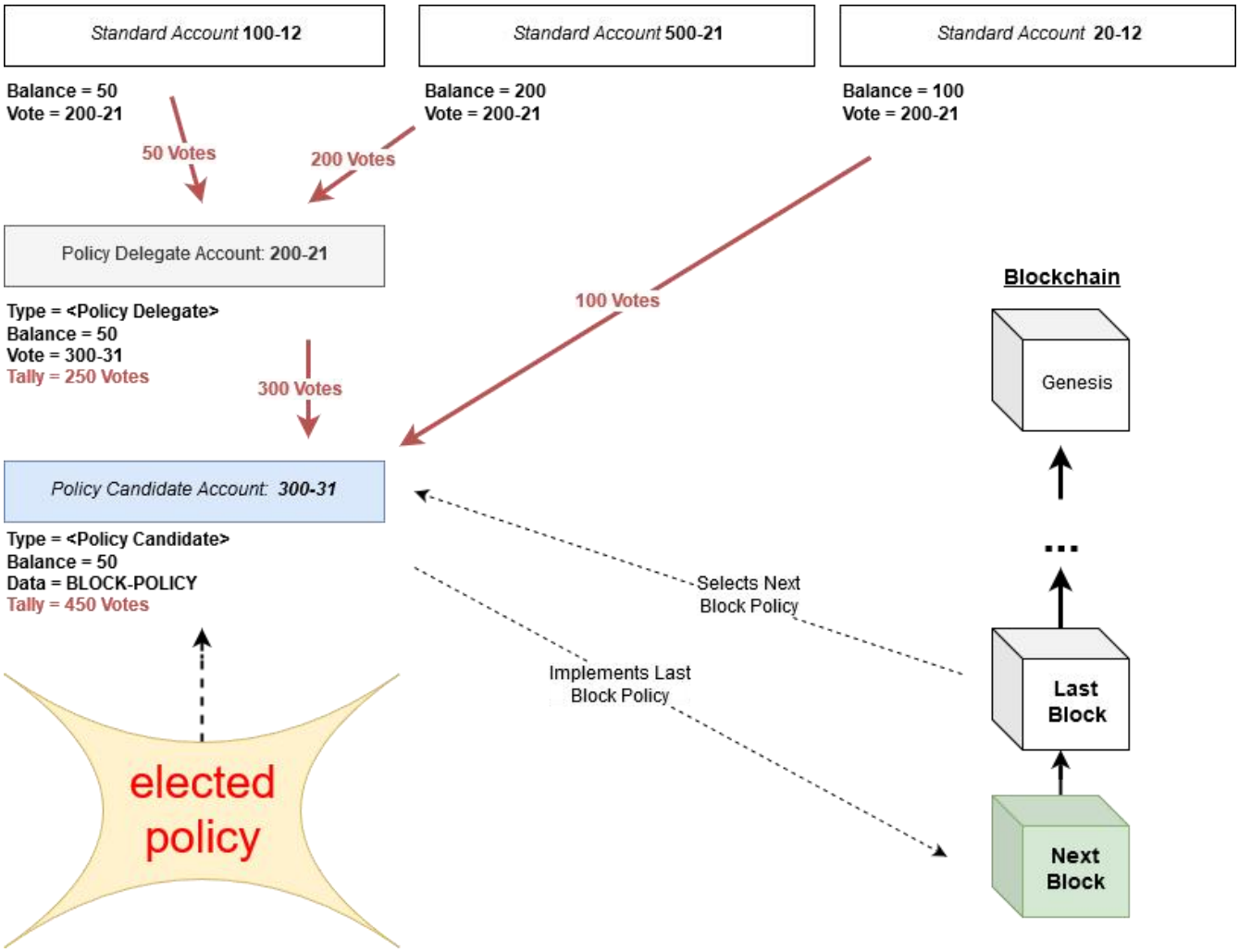
区块策略和协议内治理

为版本6提出了一种简单有效的第1层治理模型，称为“块策略”[10]。 虽然只是一个相对较小的技术变革，但Block Policy提供了一项重大创新，解决了当今分散的共识系统所面临的许多问题.

区块政策建议：

- 将现有的技术和经济协议常量重新实现为动态变量.
- 允许帐户使用安全盒信号机制对这些变量进行投票.
- 允许逐个块地更改这些变量，从而支持实时治理.
- 不再授予矿商新账户，而是在没有所有权的情况下发行，并以浮动价格公开出售.

该方案在第一层引入协议内实时治理，并建立了一种加密的安全投票机制，任何人、任何时间、任何权限都可以立即对其进行审计。通过将经济常量从源代码转移到块策略中，第一层协议现在从根本上增强了“外部世界反馈”。这对项目的每个层次都有深远的影响，包括安全性、可用性、采用、治理和市场价格.



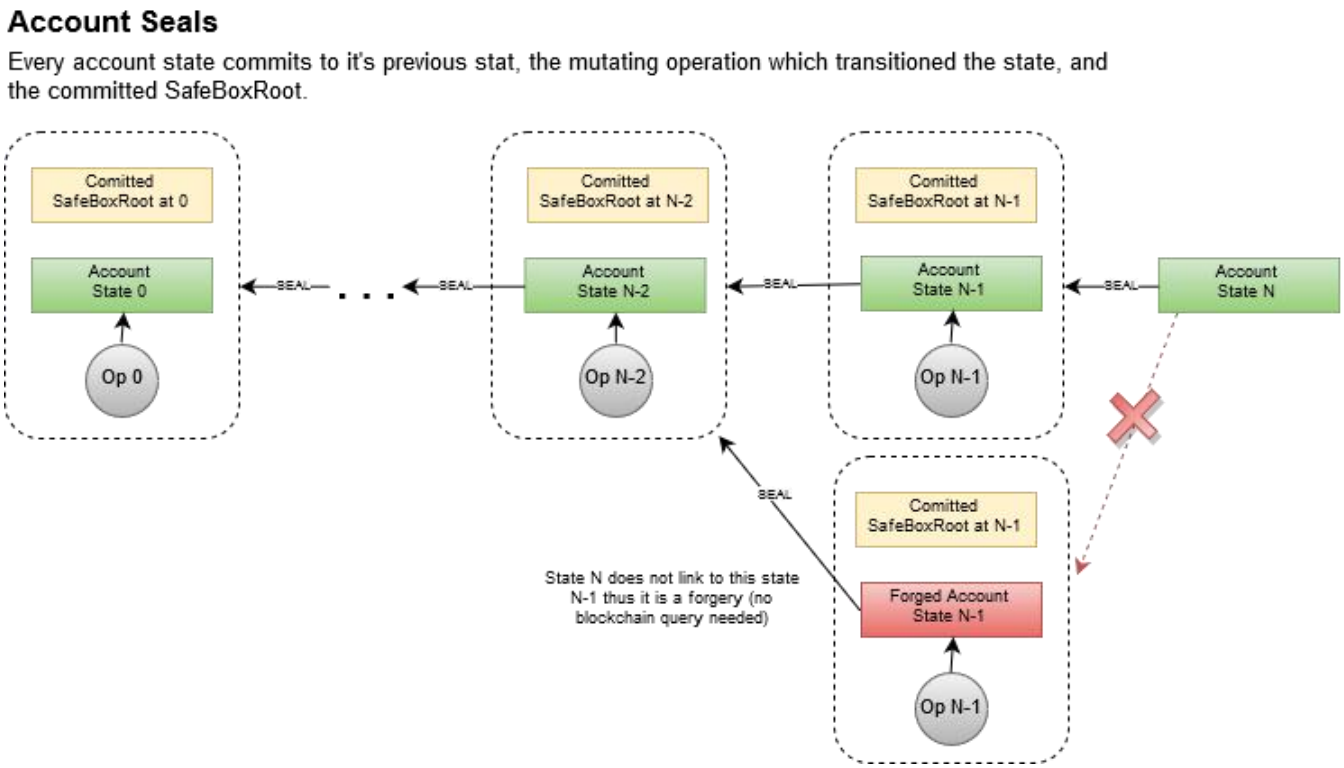
安全的帐户历史

Pascal中的事务以各种方式改变帐户状态(例如余额、名称、密钥、数据等)，并遵循网络上所有节点强制执行的严格一致规则。每次帐户状态更改时，它的先前状态都会从保险箱中丢失。由于Pascal网络只需要维护最后的100个块，所以帐户历史记录在100块之后就会从网络中丢失。

为了了解超过100个历史块，用户必须手动跟踪这些操作，或者从存档节点请求它们。这里出现了一个难题，如果没有完整的区块链，用户就无法确定帐户历史记录是伪造的、损坏的、无效的，甚至是有效的。用户只剩下一个信任模型。在无信任的情况下，这样的历史本质上是无用的，因为它无法从本质上验证。

Pascal通过使用“帐户印章”[7]解决了这个问题。每个帐户都包含一个名为“Seal”的20字节字段，该字段是对其先前状态的散列承诺，以及改变该状态的事务。这将导致一个密码可验证的帐户状态链。

由于每个状态都链接到前一个状态，而该前一个状态又链接到其前一个状态，因此帐户印章将提交该帐户的整个前一个历史记录。因此，拥有帐户历史记录的用户可以通过按时间顺序重新计算印章来验证历史记录，以确保最终印章与当前保险箱印章匹配.



LAYER-2 结构

Pascal layer -1被设计为一个可伸缩的金融层，可以在帐户之间传输值。这里的主要用例是支付。Pascal的技术在支付方面做得非常好，可以说比大多数其他加密货币都要好，因为它没有任何脚本复杂性的负担。

然而，安全盒设计允许新的、引人注目的用例，这些用例本质上将Pascal重新用作外部第二层协议的共识层和财务层。在这些用例中，Pascal充当第1层协议，它支持第2层协议，就像TCP/IP是HTTP的基本协议一样。这些第二层协议、应用程序和网络统称为“dApps”。

dApps独立于Pascal运行，使用自己的协议、网络连接和存储。它们可以以多种方式集成到Pascal中。虽然完整的技术堆栈不会在本白皮书中披露，但将提供一些讨论.

数据操作

Pascal提供一个名为OP_DATA的操作，该操作允许一个帐户向另一个帐户发送一个数据包。这些包可以是公开的，也可以是加密的。加密模式包括使用发送方或接收方密钥的ECIES加密，或使用共享密钥的AES加密。这些数据包的长度为255字节，包括一个16字节的GUID键和一个16位序列字段，可用于将数据包分组到一个更大的逻辑数据流中。这种方法为第二层协议提供了清晰的封装功能，与网络中TCP/IP封装HTTP的方式非常相似.

分散共识分类账

数据操作与加密的安全帐户历史记录相结合，可以用于为各种用例实现分散的一致账本。按照这种解释，帐户历史成为第二层节点所作的按时间顺序排列的报表的分类账，这些节点不能伪造、修改或篡改。这些数据消息嵌入符合第二层协议的第二层消息。如果这些第2层节点之间发生任何争议，此第1层帐户历史记录将作为所有第2层事务的不可变审计日志和权限。第二层的共识仅仅是通过检查第一级帐户的历史而产生的。第2层节点之间不需要任何信任。

由于帐户历史在密码上是安全的，并且本质上是可验证的，因此第二层节点可以在不使用第1层区块链或安全盒的情况下验证它们的一致分类账。他们只需要确保最新的帐户状态与大多数工作证明保险箱中的帐户状态匹配。这可以通过请求来自第一层网络的父帐户段的merke -proof来实现，这实际上是一个即时操作.

证明安全重叠网络

目前，第二层网络可以使用第1层Pascal作为仅供协商一致使用的层，但不能作为财务层。为了完成第二层的集成，Pascal需要一种机制来启用第二层网络来授权资金

从第1层Pascal帐户开始，第1层矿工不需要参与第2层业务逻辑。允许第二层网络实现这一目标的技术已经完全设计好了。在原型制作完成并得到验证之前，该技术的细节目前仍未公开.

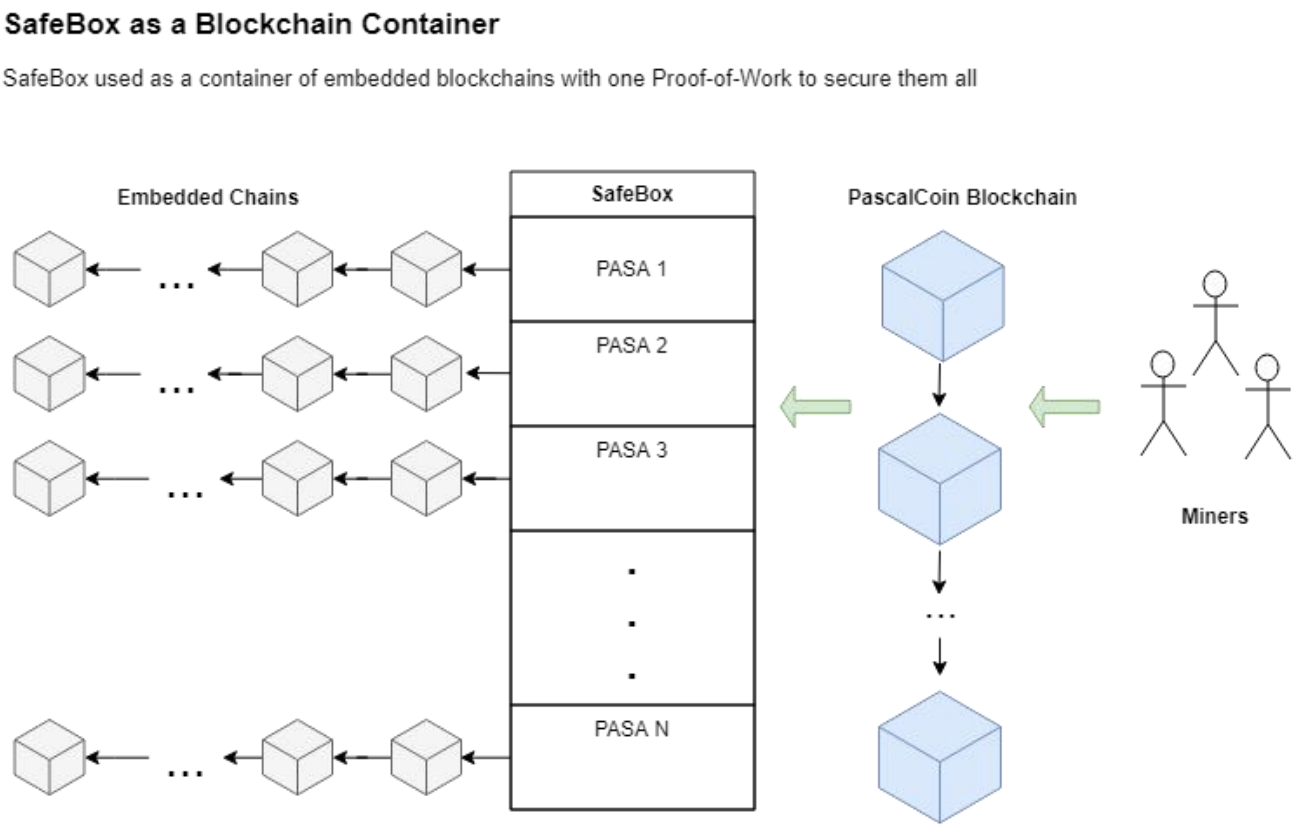
货币化 APIs

由于Pascal的基于帐户的模型，启用了一种新的dApp形式，称为货币化API。货币化API是将帐户重新概念化为消息队列。进入消息队列的数据操作充当“请求”，类似地，来自消息队列的数据操作充当“响应”。由于请求和响应包含数据和货币价值，因此创建了货币化的API系统。

货币化的api已经在部署中，比如GetPasa.com，它接收包含公钥的请求并通过向该密钥发送帐户来响应。类似地，Pascal的安全盒体系结构目前支持一整类“智能代理”应用程序.

嵌入链

上面描述的分散一致分类账方案也可以用来维护侧链的一组块头，称为“嵌入链”。来自嵌入链的块的内容不一定要包含，只需要包含头。在这种解释下，安全盒变成了一个“区块链容器”，能够保护大量区块链(每个帐户一个)，其中只有一层-1工作证明可以保护所有区块链.



致谢

作者感谢Tyler Swob撰写的这份白皮书的建议和编辑.

参考

[1] Schoenfeld, H (2018). PIP-0010 - 50% Inflation Reduction. *URL https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0010.md*

[2] Schoenfeld, H (2018). PIP-0011 - 20% Developer Reward. *URL https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0011.md*

[3] Schoenfeld, H (2018). PIP-0017 - MULTI-OP: atomic batch operations. *URL https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0017.md*

[4] Schoenfeld, H (2017). PIP-0009 - RandomHash: GPU & ASIC Resistant Hash Algorithm. *URL https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0009.md*

[5] Schoenfeld, H (2018). PIP-0016 - DATA-OP: In-Protocol Data Exchange (Layer-2 support). URL *https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0016.md*

[6] Schoenfeld, H (2018). PIP-0015 - Fast Block Propagation. *URL https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0015.md*

[7] Schoenfeld, H (2019). PIP-0029 - Account Seals: Cryptographically Secure Account Histories. *URL https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0029.md*

[8] Schoenfeld, H (2019). PIP-0030 - SafeBoxRoot: Deletable SafeBox and Light-Nodes. *URL https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0030.md*

[9] Schoenfeld, H (2019). PIP-0027 - E-PASA: Infinite Address-Space (via Layer-2). *URL https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0027.md*

[10] Schoenfeld, H (2019). PIP-0035 - Block Policy: Layer-1 Governance. *URL https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0035.md*

[11] Schoenfeld, H (2019). PIP-0036 - RandomHash2: Enhanced GPU & ASIC Resistant Hash Algorithm. *URL https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0035.md*

[12] Tim Ruffing, Pedro Moreno-Sanchez, Aniket Kate. CoinShuffle: Practical Decentralized Coin Mixing for Bitcoin. *URL https://crypsys.mmci.uni-saarland.de/projects/CoinShuffle/coinshuffle.pdf*

[13] Jonald Fyookball, Mark B. Lundeberg. CashFusion. *URL https://github.com/cashshuffle/spec/blob/master/CASHFUSION.md*

[14] Schoenfeld, H (2017). PIP-0003 - Infinite Scaling via Deletable Blockchain. *URL https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0003.md*

[15] Schoenfeld, H (2019). PIP-0032A - Atomic Swaps via Hash-Locked Accounts. *URL https://github.com/PascalCoin/PascalCoin/blob/master/PIP/PIP-0032A.md*

[16] Molina, A (2016). Pascal Coin: Crypto currency without need of historical operations. *URL https://github.com/PascalCoin/PascalCoin/blob/master/PascalCoin%20White%20Paper%20-%20EN.pdf*