(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 119645682 A
(43)申请公布日 2025.03.18

(21)申请号 202411708120.7

(22)申请日 2020.04.22

(30)优先权数据
19178579.9 2019.06.05 EP
19178583.1 2019.06.05 EP
19208139.6 2019.11.08 EP

(62)分案原申请数据
202080036913.2 2020.04.22

(71)申请人 万事达卡国际公司
地址 美国

(72)发明人 C·拉杜 M·克林吉
O·拉齐马尼

(74)专利代理机构 中国贸促会专利商标事务所有限公司 11038
专利代理师 冯薇

(51)Int.Cl.
G06F 9/54(2006.01)
H04L 67/1097(2022.01)
H04L 9/40(2022.01)
H04L 9/08(2006.01)
H04L 9/32(2006.01)
G16Y 40/30(2020.01)

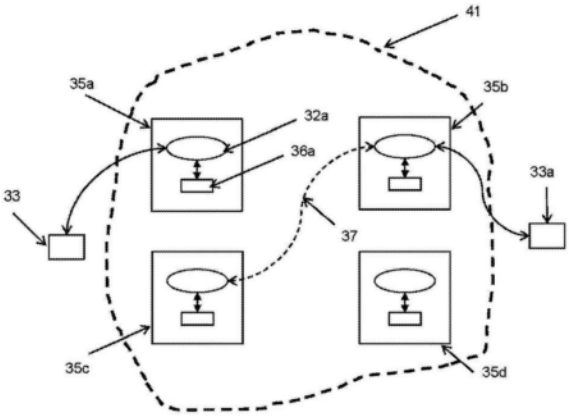
权利要求书2页 说明书24页 附图26页

(54)发明名称

分布式计算系统中的事件管理

(57)摘要

本公开涉及分布式计算系统中的事件管理。描述了一种在分布式计算系统中管理服务事件的方法。分布式计算系统包括能够使用服务过程执行服务的多个计算节点。该方法发生在计算节点中的一个计算节点处。接收或创建服务事件。该服务事件由节点标识符、时间元素和本地计数器值的组合识别。本地计数器值表示自从上次重置以来由服务过程为用户执行的服务事件的数量。然后根据节点标识符和本地计数器值将所识别的服务事件存储在服务过程数据库中。服务过程数据库用于管理分布式系统中的服务事件。当服务事件不再有效时,使用时间元素从服务过程数据库中移除服务事件。还描述了适当的计算节点和适当的分布式系统。



CN 119645682 A

1.一种在分布式计算系统中管理凭证的使用的方法,其中分布式计算系统包括多个计算节点,并且其中第一组计算节点适于提供凭证生成服务,并且其中第二组计算节点适于提供凭证验证服务,其中凭证包括密码证明,其中所述方法包括:

分布式计算系统接收为用户账户生成凭证的请求,其中该请求被定向到作为第一组计算节点中的一个计算节点的第一计算节点;

所述第一计算节点生成所述凭证,并且在与所述第一计算节点处的凭证生成相关联的第一数据库中为所述用户账户创建或增加计数器值条目;

所述第一计算节点响应于所述请求而提供所述凭证以及第一节点的标识、用于生成所述凭证的时间值和所述计数器值条目的计数器值;

分布式计算系统接收验证所述凭证的请求,其中该请求被定向到作为第二组计算节点中的一个计算节点的第二计算节点;

所述第二计算节点验证所述凭证,并且在第二数据库中创建或增加至少一个另外的计数器值条目用于跟踪凭证的验证,其中所述至少一个另外的计数器值条目与第一节点和用户账户相关联。

2.如权利要求1所述的方法,其中至少一个计算节点是第一组计算节点和第二组计算节点两者的成员。

3.如权利要求2所述的方法,其中第一计算节点和第二计算节点是相同的计算节点。

4.如前述权利要求中的任一项所述的方法,其中生成所述凭证需要使用在分布式计算系统中用于凭证生成的多个密码密钥中的一个密码密钥,并且其中验证所述凭证需要访问用于生成所述凭证的密码密钥。

5.如权利要求4所述的方法,其中分布式计算系统具有多个密钥列表,其中每个密钥列表识别一组密码密钥,并且其中密钥列表被分发到用于凭证生成和凭证验证的计算节点中的每个计算节点,其中在第一节点处,在给定时间仅所述密钥列表中的指定的一个密钥列表可用于凭证生成,所述密钥列表具有用于凭证生成的有效期和用于凭证验证的有效期。

6.如权利要求5所述的方法,其中在用于凭证生成的有效期之后,新密钥列表对于凭证生成变得有效并且计数器值被重置。

7.如权利要求5或权利要求6所述的方法,其中所述计数器值具有预定最大值,并且当所述计数器值达到所述预定最大值时,第一计算节点拒绝为所述用户账户生成任何另外的凭证,直到所述计数器值被重置为止。

8.如权利要求5至7中的任一项所述的方法,其中所述计数器值用于从密钥列表中确定用于生成所述凭证的密码密钥。

9.如前述权利要求中的任一项所述的方法,其中所述至少一个另外的计数器中的一个包括重放计数器,用以跟踪对重放由给定节点为给定用户账户生成的凭证的使用的尝试。

10.如前述权利要求中的任一项所述的方法,其中所述至少一个另外的计数器中的一个包括验证失败计数器,用以跟踪由给定节点为给定用户账户生成的凭证的验证的密码失败。

11.如从属于权利要求5的权利要求9或权利要求10所述的方法,其中所述跟踪针对由给定节点为给定用户账户针对给定密钥列表生成的凭证。

12.如从属于权利要求11的前述权利要求中的任一项所述的方法,其中当与第二数据

库中的计数器值相关联的密钥列表对于凭证的验证不再有效时,删除这些计数器值的条目。

13.如前述权利要求中的任一项所述的方法,其中所述另外的计数器值中的每一个具有预定限制,并且如果达到所述预定限制,那么第二计算节点拒绝为所述用户账户验证任何另外的凭证。

14.如前述权利要求中的任一项所述的方法,其中所述请求与交易的执行相关,并且其中执行凭证生成以用于提供要授权的交易,并且其中执行凭证验证以用于对交易进行授权。

15.一种包括多个计算节点的分布式计算系统,其中第一组计算节点适于提供凭证生成服务,并且其中第二组计算节点适于提供凭证验证服务,其中凭证包括密码证明,其中所述分布式计算系统适于执行如权利要求1至14中的至少一项所述的方法。

分布式计算系统中的事件管理

[0001] 本申请是国际申请日为2020年4月22日、国家申请号为202080036913.2、发明名称为“分布式计算系统中的事件管理”的进入中国国家阶段的PCT申请的分案申请。

[0002] 相关申请的交叉引用

[0003] 本申请基于并要求于2019年6月5日提交的序列号为19178579.9的欧洲专利申请;2019年6月5日提交的序列号为19178583.1的欧洲专利申请;以及2019年11月8日提交的序列号为19208139.6的欧洲专利申请的权益和优先权,其内容出于所有目的在此全文并入本文。

技术领域

[0004] 本公开涉及分布式计算系统中的事件管理。

背景技术

[0005] 存在需要中央化(centralized)系统为非常大量的客户端提供服务的许多技术挑战,尤其是当这些客户端在地理上广泛分布时。考虑分布系统使得相关服务能够由一组地理上分布的服务器提供,而不是由一个中央服务器或数据中心提供是合乎逻辑的。

[0006] 在实践中,这种去中央化可能会使用云体系架构,该云体系架构将通常使用多个地理上分布的服务器——或数据中心——来向客户端递送服务。云体系架构可以被认为包括多个节点——当使用云体系架构时,节点可以是多个计算机的聚合,并且可以在给定节点内以“实时”连接和数据共享来覆盖多于一个的数据中心。

[0007] 去中央化本身可能是有问题的,特别是在有必要以这种方式来提供服务的情况下,服务的供给会产生超出提供服务的服务器和接收服务的客户端的后果。例如,如果其它客户端(或其它系统节点)需要回溯到服务提供节点以检查否已提供服务或如何提供服务,或者中央系统是否有必要知道已经如何提供服务或分布式服务器节点的性能,那么新的瓶颈可能会取代以前的瓶颈出现在中央服务器上,系统中的消息传递总量可能增加,并且网络延迟可能成为严重的问题。

[0008] 特别地,事件的管理在分布式系统中变得更加有问题。一个节点可能需要知道在另一个节点发生的事件,并且它可能无法预测在哪里需要该知识。在所有事件都在中央化服务器处发生的情况下,易于按以下方式识别它们:它们可以在整个系统中被一致地引用并在系统管理中使用。在事件仅在系统中的一个点处发生的情况下,在没有过多消息传递的情况下有效地处理事件的识别和管理要困难得多,特别是当可能仅在系统中的有限数量的点处需要事件的知识时,或者除非发生进一步的事件否则可能根本不需要事件的知识时。

发明内容

[0009] 在第一方面,本公开提供了一种在分布式计算系统中管理服务事件的方法,其中分布式计算系统包括能够使用服务过程执行服务的多个计算节点,其中该方法包括在所述

多个计算节点中的一个计算节点处:接收或创建由节点标识符、时间元素和本地计数器值的组合识别的服务事件,其中本地计数器值表示自从上次重置以来由服务过程为用户执行的服务事件的数量;根据节点标识符和本地计数器值将所识别的服务事件存储在服务过程数据库中;使用服务过程数据库来管理分布式系统中的服务事件;以及当服务事件不再有效时,使用时间元素从服务过程数据库中移除服务事件。

[0010] 使用这种方法,可以有效地识别分布式计算系统中出现的服务事件,并且一个计算节点可以建立它们起源于哪个其它计算节点。识别的结构有助于直接用于系统管理并且在事件不再相关时移除它们。

[0011] 在实施例中,服务事件包括密码过程。服务事件可以包括密码证明的生成。然后可以使用本地计数器来识别供密码过程使用的密码密钥。可以根据密码密钥的有效期来管理这些服务事件。

[0012] 此处教导的识别结构对于此目的特别有效,因为时间信息与计数器信息的存在允许与密钥的有效期有效协调。计数器也可以随着时间段的变化而重置,从而允许传送更小的计数器值。

[0013] 在一个示例性上下文中,该方法在与创建服务事件的服务过程相同的节点中执行。这里,该方法还包括确定针对用户的服务过程的服务事件限制、使用本地计数器对服务事件进行计数,以及在超过服务事件限制的情况下针对用户的服务过程的暂停操作。

[0014] 在另一个示例性上下文中,该方法在计算节点中执行,该计算节点包括用于验证服务事件的验证服务过程。在服务事件包括密码证明的生成的情况下,验证服务可以对密码证明进行验证。在这种情况下,可以存在与验证服务相关联的一个或多个附加计数器,存在对该一个或多个附加计数器的附加计数器限制,该方法还包括在超过该附加计数器限制中的一个附加计数器限制的情况下,暂停针对用户的服务过程的操作。在这些附加计数器中可以存在重放计数器来对验证相同服务事件的尝试进行计数。也可以存在密码失败计数器来对验证服务过程的失败进行计数以产生验证结果。

[0015] 在实施例中,分布式计算系统是交易处理系统,并且服务事件包括用于在交易处理系统中进行处理的交易详细信息的生成,并且本地计数器是本地交易计数器。

[0016] 在第二方面,本公开提供了一种分布式计算系统的计算节点,其中该计算节点至少包括编程处理器和存储器,其中该编程处理器适于执行上述第一方面的方法。

[0017] 在第三方面,本公开提供了一种分布式计算系统,包括上述第二方面的多个计算节点。

附图说明

[0018] 现在参考附图以举例的方式描述本公开的具体实施例,其中:

[0019] 图1示出了涉及与中央服务器交互的多个客户端的常规系统;

[0020] 图2示出了提供与图1中的中央服务器相同的服务的分布式计算体系架构交互的多个客户端;

[0021] 图3示出了在图1的常规系统中对事件进行计数的方法;

[0022] 图4示出了在图3的布置中对事件进行计数和管理的方法,该方法应用于本公开的实施例;

[0023] 图5示意性地示出了使用四方模型的分布式交易体系架构;

[0024] 图6图示了适于实现图5的交易体系架构的复杂分布式系统的元素;

[0025] 图7示意性地示出了用于在图5和图6的交易体系架构中实现数字交易的示例性系统;

[0026] 图8示意性地图示了用于交易的数字化实现(enablement)的分布式系统的布置;

[0027] 图9更详细地图示了图8的布置的计算节点;

[0028] 图10图示了图9的计算节点内的元素;

[0029] 图11指示与图9的节点执行的操作相关的交易流程;

[0030] 图12指示在图9到图11的布置的实施例中使用令牌化(tokenization);

[0031] 图13指示可应用于图8的布置的密钥管理的方法;

[0032] 图14图示了与本公开的实施例相关的交易识别的示例性方法;

[0033] 图15图示了可应用于图8的布置的用于数字化交易的一组示例性密码机制;

[0034] 图16图示了具有如图13中所示管理的各个节点的密钥管理的全局模型;

[0035] 图17图示了与图13和图16的密钥管理模型相关联的全局监控模型;

[0036] 图18图示了使用具有图9和图10的节点的传统用例的交易的示例性修改令牌化过程;

[0037] 图19图示了使用传统用例的系统的密钥轮换过程;

[0038] 图20图示了用于使用传统用例的数字化交易的一组示例性密码机制;

[0039] 图21图示了使用适合与图9和图10的节点一起使用的传统用例来传送(carry)本地交易计数器的方法;

[0040] 图22图示了在使用与图9和图10的节点一起使用的卡核实码(CVC)来递送本地交易计数器时使用图21的方法;

[0041] 图23图示了使用适合与图9和图10的节点一起使用的UCAF(通用持卡人认证字段)格式来传送交易凭证信息作为交易的一部分的方法;

[0042] 图24图示了用于使用UCAF格式的数字化交易的一组示例性密码机制;

[0043] 图25图示了使用用于与图9和图10的节点一起使用的DPD(数字支付数据)来传送交易凭证信息的方法;

[0044] 图26更详细地指示了LTC在密钥使用和凭证生成(或验证)方面的作用;

[0045] 图27图示了LTC在解决诸如重放检测、密码失败和重试之类的系统问题中的作用。

具体实施方式

[0046] 一般而言,本公开解决的问题在图1和图2中图示。图1示出了响应于来自非常大量的地理上分布的实体的请求而执行功能的中央系统。这对中央系统在处理能力、存储和消息传递方面提出了强烈需求,并且通常会由于瓶颈和消息传递要求而导致整个系统出现显著延迟。

[0047] 图2示出了替代布置,其中中央系统的角色被分解,使得由一组分布式节点执行相同的功能,每个节点都具有执行中央系统所提供的一些或全部功能的能力。各个节点都应看到比中央系统低得多的需求,并且由于实体应该能够与比中央系统更本地的节点进行交互,因此可以潜在地减少延迟。但是,如上面一般性讨论的,以及下面与交易处理系统具体

相关的,在实现这种好处方面存在重大的技术挑战。

[0048] 图3示出了图1的基于常规中央服务器的系统中的事件处理的典型方法。中央服务器31包含为请求者33提供服务的服务过程(service process)32。如果需要识别服务的每个实例或相关联的服务结果,那么计数过程34与服务过程一起工作以在每次服务过程32为请求者执行服务时分配唯一标识符。该唯一标识符可以用作对该服务实例或该服务实例的结果的后续引用。由于与该服务执行相关的任何进一步交互将涉及返回到服务过程32或中央节点31中的另一个过程,因此可以容易地随时使用该唯一标识符。

[0049] 在诸如图2中所示的分布式系统中,情况更复杂。可以在系统中的多个节点中的一个节点处生成服务,并且其它节点可能需要知道并接合(engage)该服务结果。因此,本公开的实施例提供处理这种更复杂情况的方法。图4图示了如何在包括多个本地节点35a、35b、35c、35d的分布式系统41中实现这一点的一般示例。在这种情况下,请求者33可以从本地节点35a、35b、35c、35d中的一个请求服务。每个本地节点35a在此被示为具有其自己的服务过程32a(在以下讨论的实施例中,本地节点可以具有多个不同的服务,并且可以具有多于一个类型的服务)。

[0050] 该服务过程32a在此与本地计数器36a交互。本地计数器36a对在本地节点35a处执行的服务进行计数并向它们提供对于该本地节点35a唯一的值。如下文进一步描述的,该值在一段时间内可能是唯一的,本地计数器值在该时段之后被重置——在这种情况下,需要本地计数器值以及时间段的指示来实现本地唯一性。该值在分布式系统中通常不是唯一的——两个不同的本地节点35a、35b可以使用相同的本地计数器值(或者甚至相同的本地计数器值和时间段值)识别不同的服务事件。为了用作系统中的标识符,本地唯一值可以与本地节点35a本身的标识组合。这可以用于提供在整个系统中有效唯一的值,并且还允许识别服务的提供商。

[0051] 这种方法具有许多实际用途——例如,请求者33a向本地节点35b做出的新服务请求可能已经用特定标识符识别了先前的服务结果,该标识符指示服务是由本地节点35c提供的。这在第二服务使用第一服务的结果的情况下尤其相关。如下面在特定示例中更详细讨论的那样,这可能适用于当第一服务生成证明并且第二服务验证该证明时——通常这需要第一服务和第二服务有权访问相同或互补的密码资源。这里,这可以导致本地节点35b和本地节点35c之间关于本地节点35b执行第二服务的消息传递37。这可以用于服务管理,例如通过允许确定来自第一服务的原始服务结果是否被滥用。本地节点标识符可以用于建立通信,并且本地计数器可以用于帮助确定服务实例的使用方式。这种使用可以是在特定时间段内——例如,在本地计数器重置之间。下面描述了可以使用这种类型的标识符和特别是本地计数器元素来实现不同功能结果的具体方式。这在交易系统的上下文中详细描述,但应该理解的是,这些可应用于其它功能系统。

[0052] 这个问题与交易处理系统特别相关,特别是与用于处理数字交易的系统相关。在这些情况下,上述本地计数器是本地交易计数器(LTC)。数字交易的数量增长极快,并且需要它们可靠且快速地执行。这些交易的支持可以使用为使用支付卡的基于设备的支付而开发的交易处理系统,并使用此类支付系统的协议,但实际上此类交易与基于设备的交易具有不同的特点。这将在下面讨论,首先参考交易处理系统的一般元素,然后更详细地讨论用于支持数字交易的基础设施。

[0053] 图5是典型的四方模型或四方支付交易方案的框图。该图图示了模型中存在的实体以及在卡方案中操作的实体之间发生的交互。

[0054] 通常,卡方案——与支付卡相关的支付网络——基于以下两个模型中的一个:三方模型或四方模型(由本申请人采用)。出于本文档的目的,下面更详细地描述四方模型。

[0055] 四方模型可以用作交易网络的基础。对于每个交易,该模型包括四个实体类型:持卡人110、商家120、发行方130和收单方140。在该模型中,持卡人110从商家120购买商品或服务。发行方130是向持卡人110发行卡的银行或任何其它金融机构。收单方140向商家120提供用于卡处理的服务。

[0056] 该模型还包括中央交换机150——发行方130和收单方140之间的交互经由交换机150进行路由。交换机150使与一个特定银行收单方140相关联的商家120能够接受来自与不同银行发行方130相关联的持卡人110的支付交易。

[0057] 四方模型中实体之间的典型交易可以被划分为两个主要阶段:授权和结算。持卡人110使用他们的卡发起从商家120购买商品或服务。卡和交易的详细信息经由收单方140和交换机150发送到发行方130以授权交易。持卡人110可能已经在交易中提供了核实信息,并且在一些情况下可能需要经历附加的核实过程来核实他们的身份(诸如,在在线交易情况下的3-D安全)。一旦附加的核实过程完成,交易就被授权。

[0058] 在持卡人110与商家120之间的交易完成后,商家120将交易详细信息提交给收单方140以进行结算。

[0059] 交易详细信息然后由收单方140经由交换机150路由到相关发行方130。在接收到这些交易详细信息后,发行方130将结算资金提供给交换机150,交换机150进而经由收单方140将这些资金转发给商家120。

[0060] 单独地,发行方130和持卡人110在他们之间结算支付金额。作为回报,商家120针对每个交易向收单方140支付服务费,并且收单方140向发行方130支付交换(interchange)费以换取资金结算。

[0061] 在四方系统模型的实际实施方式中,特定方的角色可能涉及共同作用的多个元素。在已经发展超越客户卡和商家终端之间基于接触的交互的实施方式到在诸如智能电话之类的用户计算设备上使用代理或虚拟卡的数字实施方式中,通常是这种情况。

[0062] 图6示出了根据本公开的实施例的适合于持卡人和商家之间的交互的体系架构。该图示出了通用体系架构以供参考,但特别示出了当持卡人与商家服务器进行在线交易时使用的体系架构的特定元素。

[0063] 对于常规的交易,持卡人将使用他们的支付卡6——或诸如适于用作非接触式支付设备的智能电话11之类的移动计算设备——与商家2的POS终端7进行交易。但是,在与本发明相关的实施例中,持卡人将使用他或她的计算设备——其可以是蜂窝电话手机、平板电脑、膝上型电脑、静态个人计算机或任何其它合适的计算设备中的任何或全部计算设备(这里示出了移动电话手机或智能电话11)——并且也可以使用其它计算设备(诸如智能手表或其它可穿戴设备)——用以充当物理支付卡6的代理或充当仅在数字域中操作的虚拟支付卡。如下所述,智能电话11可以通过移动支付应用和数字钱包来实现这一点。智能电话11可以使用它来使用NFC或其它非接触式技术与商家POS终端7进行交易,或者如下面所讨论的与其钱包服务相关联地进行支付。但是,本公开的实施例特别关注的是与商家的在线

交易,而不是与商家POS终端7的接触式或非接触式交易。为了进行在线交易,智能电话11还能够通过任何适当的网络连接(诸如公共互联网)与表示商家2的商家服务器12交互——与商家的连接可以由计算设备上的app或应用提供。

[0064] 交易方案基础设施(交易基础设施)5在此不仅提供操作卡方案所需的计算基础设施以及向诸如收单方3和发行方4的各方提供交易路由和其它消息传递,而且还提供支持持卡人计算设备上的数字钱包的钱包服务17,并且提供互联网网关18以接受基于互联网的交易以供交易基础设施处理。在其它实施例中,钱包服务17可以由与交易方案提供商具有适当信任关系的第三方类似地提供。为了支持令牌化,存在令牌服务提供商19(同样,这被示为交易基础设施5的一部分,但可以由具有适当信任关系的第三方提供),并且交易方案基础设施提供数字实现服务16以支持令牌化数字交易的执行,以及与系统的其它元素交互以使得交易被正确执行——这种数字实现服务可以包括其它元素,诸如令牌服务供给。

[0065] 对于令牌化交易,通过将持卡人令牌映射到他们的卡PAN、检查令牌状态(以确保它处于有效期或以其它方式是有效的)以及使用的任何客户核实方法,在交易方案中对交易进行验证。这允许发行方以普通方式对交易进行授权。

[0066] 图7更详细地示出了支持从移动设备进行数字化支付的交易基础设施的元素。该图示出了作为具体示例的申请人的基于万事达卡云的支付(Mastercard Cloud Based Payment,MCBP)体系架构——这是示例性的而非特定于本发明的,并且图示了该体系架构如何用于支持移动设备(诸如智能电话11)上的移动支付应用215—这里移动支付应用215被显示为包含在钱包应用或数字钱包41内。这样的数字钱包41可以与钱包服务器17通信以允许管理移动支付应用,并且它还可以用于请求由移动设备11使用的支付卡6的数字化。

[0067] 万事达卡数字实现服务(MDES)42执行各种功能以支持移动支付和数字化交易。如上所述,MDES 42仅是示例性的——例如,其它实施例可以使用与其它交易处理基础设施相关联的数字化、令牌化和供给服务。钱包服务器17不是MDES 42的一部分——并且在例如移动支付应用215没有嵌入在数字钱包41内的情况下不需要存在——而是充当移动设备11和MDES 42之间的接口。MDES 42还对令牌化交易进行调解,使得它们可以像常规卡交易一样通过交易方案进行处理。MDES 42内显示以下功能元素:账户实现系统(AES)43、凭证管理系统(CMS)44、令牌库45和交易管理系统(TMS)46。这些将在下面简要描述。

[0068] 账户实现系统(AES)43用于卡数字化和用户建立。它将与移动支付应用(这里通过钱包服务器17)交互以进行卡数字化请求,并将在令牌化时填充令牌库45,并将与CMS 44交互以建立具有相关联的密钥的卡简档,用于卡的数字化使用。

[0069] 凭证管理系统(CMS)44支持持卡人凭证的管理,并且是MDES 42内的密钥系统。核心系统441通过与TMS 46的交互来管理与整个交易系统的同步,并管理到AES 43的通道。专用系统442以使用所需的形式向移动支付应用提供必要元素(诸如数字化卡和凭证和密钥)的递送。该系统还可以与钱包服务器17交互以管理移动支付应用。

[0070] 令牌库45——这里显示为在MDES 42内,但它可以是单独控制下的单独元素——是用于令牌信息的储存库,包括令牌和相关联的卡之间的对应关系。在处理令牌化交易时,MDES 42将参考令牌库45,并且卡的令牌化将导致在令牌库45中创建新的条目。

[0071] 交易管理系统(TMS)46在处理令牌化交易时使用。如果交易方案将交易识别为被令牌化,那么交易被路由到TMS 46,TMS 46通过使用令牌库45对交易进行去令牌化。去令牌

化的交易然后被路由到发行方(这里由金融授权系统47表示)以常规方式进行授权。TMS 46还与CMS 44交互以确保与持卡人账户和凭证相关的同步。

[0072] 本公开的实施例针对实现用于执行如图7中所示的数字化交易的系统的各方面——特别是凭证的管理和使用——能够被去中央化。这是通过用一组去中央化的节点替换中央节点来完成的,每个去中央化节点能够进行凭证管理,如图8到图10中所示。

[0073] 图8示出了计算节点Nx的去中央化的系统,每个节点都能够生成G和验证V凭证。这些凭证可以在整个系统中有效(除非由于土壤(on-soil)监管等而仅限于一些节点),并且在这种情况下与一组用户(客户端)的交易相关联,这些用户(客户端)的交易通常通过地理接近度被路由到该节点。节点向客户端提供凭证生成G和凭证验证V作为服务,并且需要能够安全地生成凭证并至少在它们有效时安全地验证它们。在所示的体系架构中,凭证没有被存储——它们是根据请求生成的,并即时进行验证。如图8和图9中所示,除了凭证生成和验证之外,密钥管理K和监控M可以被视为在节点本地和跨系统两者的服务,并且通常需要访问控制AC来允许访问服务。下面将更详细地描述这些方面。

[0074] 合适的计算节点的元素如图10中所示。节点80包括至少一个网络连接81以允许与客户端90和其它节点91以及(在该示例中)协调一个或若干个节点之间的活动的中央节点91a的通信。这里将通信显示为通过单独的网络到每一组其它方——通过用于连接到客户端的第一网络云92和用于连接到分布式系统内的其它节点的第二网络云92a。这反映了这些网络可以在物理上不同,或者可以有不同的安全要求和协议。

[0075] 节点80包含多个常规服务器83(它们将包含它们自己的处理器和存储器——未示出——以及通常在服务器中发现的其它组件)和包含中央数据库的存储器84。节点80内还包括多个硬件安全模块85(HSM),其适于保存密码材料并安全地执行密码函数。这里节点80内的元素被示出借助于总线86进行通信。虽然在这种情况下节点80被表示为单个数据中心,但这不是必需的——“总线”可以例如包括一组相关数据中心之间的专用网络连接,其允许它们提供实时响应,使得它们对于与节点通信的其它实体看起来是集成整体的一部分。

[0076] 支付系统中现有的凭证管理过程是中央化的——创建或验证凭证的任何请求都会导致对中央化系统的查询。对于实现EMV标准的支付系统,凭证是使用根据分层过程导出的密钥生成的。发行方主密钥(IMK)与特定范围的令牌相关联,并且用于凭证的密钥是分层导出的(卡主密钥-CMK-来自IMK,然后是会话密钥-SK-来自CMK)。这种方法用于设备(诸如物理卡),但也用于数字交易。与基于设备的交互相比,数字交易的数量增长非常迅速,其中增长与资源更加一致。

[0077] 在数字生态系统中,虽然存在非常快速的需求增长,但通常也存在更安全的环境,因为交互通常是在商家系统(或支付服务提供商)和交易系统之间通过身份明确的参与者之间的安全途径进行的。因此,当将所有资产安全保持在包括密钥管理和密码操作的受限环境中时,当暴露访问服务的API时,当在服务器上下文中提供服务时,在可以被流线化的设备上下文中存在为了安全性可能要求多个密码操作的交互。

[0078] 虽然通过使用一组分布式服务器来生成和验证凭证来扩展用于执行数字EMV交易的交易系统似乎是可取的,但发现这种方法无法扩展。密钥生成的整体水平不会改变,但系统内的消息传递量会大大增加,因为需要管理和复制非常大量的令牌。由于现有的EMV密钥

生成方法需要定制的代替现成的硬件安全模块(HSM),因此处理要求高且成本高昂,并且数据存储,特别是网络延迟将变得无法管理问题。

[0079] 本公开的实施例通过将令牌的绑定替换为特定分层导出的密钥从而允许将来自密钥堆栈的第一可用密钥改为分配给令牌化交易来支持这种分布式方法。这种方法使用灵活和动态的密钥管理,允许可扩展的解决方案。可以以确保分布式体系架构安全的方式进行监控,而无需传输或复制大量敏感信息。这种方法也可以使用完全符合FIPS的流程在标准HSM中执行——例如,不需要使用DES和3DES。下面更详细地描述这种方法。

[0080] 目前,设备安全模型也用于完全数字交易。这种安全模型涉及存储在交易系统HSM中并用于从相关IMK和卡PAN(主账号)导出卡主密钥(CMK)的发行方主密钥(IMK)。然后这些CMK被存储在设备(通常是安全元件或替代技术)中。当使用基于软件的解决方案使用移动设备生成交易凭证时,使用相关CMK和卡/设备的ATC(应用交易计数器)生成会话密钥(SK)——这目前由如图7中所示的凭证管理系统(CMS)生成。目前,所有的令牌,即使对于完全数字交易,都绑定到这种IMK/CMK/SK推导。这也适用于由服务器通过交易系统暴露的用于远程支付交易的API而生成的交易凭证。

[0081] 虽然术语PAN在下文中一般性地使用,但在数字化交易的上下文中,使用术语TUR(令牌唯一引用)来指代卡或账户的唯一标识符也是合适的。字面上,这些术语应按如下期望的方式使用,以便彼此区分:

[0082] ●PAN是与账户直接相关联的值——这是识别账户的正常(数字)方式——术语FPAN或资金PAN可以用于指示对发行银行账户的引用;

[0083] ●TUR或“令牌唯一引用”是允许在不暴露任何PAN值的情况下识别令牌的值,交易系统中存在确定哪个PAN与TUR相关联的机制。

[0084] 但是,当在下面使用术语PAN时,应该理解这是在广义上使用的标识符,该标识符可以与识别它的账户相关联——并且因此下面使用PAN可以包括TUR。

[0085] 这种方法需要非常繁重的密钥管理负载,这不适用于完全数字交易,如下面参考图11和图12所讨论的。SK的生成以及因此应用密码(AC—EMV交易中的标准机制)需要多个密码操作,并非所有这些操作都可以由常规的现成的HSM执行,因此需要定制的HSM。需要在整个系统中大量分发密钥,使得在发生交易的任何地方交易的执行都能得到支持,并且ATC管理是复杂的。期望使用标准HSM、避免大量密钥复制,同时让密钥直接可用,并且能够提供限制整体HSM数量的解决方案(因为它们通常仅支持几千个密钥)。

[0086] 这种安全性的大部分是即使在系统端点(例如,在持卡人设备处)有可能受到威胁时也提供安全保证。除此之外,安全性的作用有限,如图11中所示。密码函数的主要目的是提供保证——这覆盖了数据的完整性和认证两者。受密码数据保护的交易相关数据包括交易的标识和相关联的令牌,以及所使用的任何密码过程的指示和任何相关的金融数据(以及需要保证的交易的任何其它方面)。这由交易凭证表示——这需要生成G并随后验证V,这些过程被监控M以确保整体系统完整性并由某种密钥管理系统K支持。

[0087] 在完全数字交易的情况下,这些过程发生在受限环境中,其中端点安全性不像设备那样成为问题。如从图12中可以看出,在该域中,令牌不会到达常规交易管理系统的任一端点——持卡人或发行方。替代地,它跨商家系统或支付服务提供商(PSP)和交易方案提供商进行操作。

[0088] 这种方法允许将凭证系统从复杂的中央服务器分散到提供服务的多个节点中。这些节点通常在地理上分布,但可能会扩展到多个数据中心(例如,通过使用云基础设施来实现节点内的数据共享)。这些节点提供与凭证、生成服务G和验证服务V相关的服务,并定义对服务的访问控制的规则。商家或PSP与生成服务G通信以获得凭证,然后在标准授权过程中使用这些凭证,在需要验证凭证时调用验证服务V。这些服务可以访问节点的计算基础设施(HSM、数据库)。还提供监控M和密钥管理K服务——这些服务可以集中组织或包括协调和本地功能的混合。下面将更详细地描述所有这些服务及其相互关系。

[0089] 可以以基本上常规的方式提供对服务的访问控制。可以为节点定义一组通用控制,并可以进行本地修改——例如,以满足本地监管或其它特定安全要求。这种方法可以轻松实现特定于国家的策略,例如通过将特定国家的所有业务限制到一组特定节点。访问控制可以在多于一个的级别(例如,针对各个服务,但也针对节点)执行,并且可以存在针对特定服务类型的特定规则或检查。访问控制潜在地非常颗粒化,并且可以以多种方式提供特定的解决方案——例如,它可以用于允许给定商家在给定令牌的定义时间期间执行最大数量的交易凭证生成操作。

[0090] 图13中所示的密钥管理机制图示了如何能够将有限数量的密钥分配给节点,同时提供确定性过程以选取用于生成凭证的密钥。验证实体可以使用相同的过程来确定生成器使用的密钥,使得它可以验证作为提交验证的凭证的一部分的任何密码材料。

[0091] 对于每个节点,生成G和验证V服务都可以访问HSM池。HSM包含各自由一组密钥标识符(KeyId)唯一识别的密钥。KeyId可以是标签、值、明确的唯一值(诸如UUID)、或具有适当属性的任何其它内容。这些KeyId被存储在唯一识别(标识符)密钥列表中——这些密钥列表提供了标识符(Id)和存储的密钥(KeyId)之间的关系列表。标识符(Id)是由确定性过程确定的,以便建立要使用什么密钥,如将在下面进一步描述的。

[0092] 每个密钥列表的完整性使用密封锁(Seal)来保证——如果密钥列表是从中央位置供给的,那么这可以由与该中央位置相关联的受信任方应用。可以使用例如作为本地功能而不是中央位置的受信任方来支持若干其它分布模型。节点通常有多个可用的密钥列表,但在给定时间仅有用于生成凭证(G)的一个活动密钥列表——但是,验证服务(V)通常需要能够访问可以与仍然有效的凭证相关联的任何密钥列表。这种方法中的密钥轮换非常简单——它可能只涉及用另一个密钥列表替换活动密钥列表。但是,判断需要哪个KeyId来验证凭证非常简单——这将完全由节点标识符和对密钥列表的引用来确定。该信息是凭证的一部分,并用作确定性过程的输入以从密钥列表中选取密钥。

[0093] 图13图示了节点Ni的示例性布置,其具有能够生成与交易相关联的凭证的两个生成服务G。在任何给定的时间点,将需要这些服务G使用给定的密钥列表——比如第一实例中的密钥列表A。这使用黄色和蓝色密钥,因此这些密钥必须加载到生成服务G所使用的HSM中。在一段时间到期后,密钥轮换过程可能例如强制使用密钥列表B——这使用黄色和蓝色密钥,但也使用绿色密钥,因此绿色密钥必须加载到相关HSM中(如果尚未存在)。要使用的特定密钥是通过确定性过程从密钥列表中选择的,如下面将讨论的——这通常会在密钥轮换后给出不同的结果,但情况并非不可避免(例如,Id=3或Id=6将在轮换之前或之后给出蓝色密钥)。虽然生成服务G在密钥轮换后不需要密钥列表A,但验证服务V仍然需要——它们需要访问与潜在有效凭证相关的任何密钥列表。验证服务V必须能够确切地确定生成服

务G使用哪个密钥来生成凭证,以便验证凭证。

[0094] 要密码保护的交易相关数据包括与交易相关联的令牌的标识,但也包括交易本身的标识。为此,需要某种交易标识符。在每个节点,凭证生成和验证服务都可以访问可以用于管理此类数据的本地数据库。为确保跨系统有效管理交易,给定令牌的任何交易凭证生成都应与每个交易的唯一交易标识符相关联。这可以是UUID,但如前所述,在分布式系统中建立UUID具有挑战性,在分布式系统中,可能需要由多个分布式节点中的一个进行交易识别。在本公开的实施例中,可以使用适当的标识符结构(诸如,n比特节点标识符、e比特纪元时间和c比特本地计数器的级联)。

[0095] 在本公开的实施例中,可以通过使用本地交易计数器将交易凭证中传送的数据大小减少到几个数字位(digit)。这可以简单地存储在节点的本地数据库中,并且当本地生成服务G生成新令牌时,本地(而不是全局)值会递增,如图14中的一般项所示的过程。本地交易计数器(LTC)因此可以有助于上述有效唯一的标识符结构,其中节点标识、时间和本地交易计数器的组合用于有效且唯一地识别交易。如下文将进一步讨论的,使用基于时间的过程的密钥列表轮换的使用允许在时间段结束时重置LTC值而不会丢失这些特性,同时限制作为交易流程的一部分传送LTC值所需的数据大小。本地交易计数器的这种和其它用途将在下面进一步详细描述。

[0096] 现在将参考图13描述用于识别用于交易的密钥的示例性过程。如上所述,在任何给定时间,生成服务G可以访问本地HSM中的一组密钥,并根据其当前活动密钥列表来使用密钥。该密钥列表本身是(通过Identifier(标识符))唯一识别出的,并包含条目列表,这些条目对应于标识符(Id)和存储的密钥(由KeyId表示)之间的关系。在密钥列表A的情况下,存在十个条目,并且每个Id是整数。

[0097] 将有与密钥列表相关联的确定性过程来确定哪个密钥将与给定的交易相关联。对于每个密钥列表,不需要是相同的确定性过程,但需要一致地用于该密钥列表,使得生成和验证服务都将获得相同的结果。为了提供这种关联,确定性过程应该对识别交易的信息进行操作,诸如某种交易标识符——在这种情况下,本地交易计数器(LTC)是特别有效的选择,因为它可以方便地获得并且容易处理。这里没有必要使用上面讨论的完整唯一标识符——标识符仅需要能够确定在给定时间段内在给定节点内选择了哪个密钥,因此本地计数器(诸如LTC)适用于该目的。如将在下面看到的,这允许以确定性的方式做出选择,这种方式可以被寻求核实交易的另一方复制。

[0098] 存在可用于函数的很多选择,但最简单的选择是MOD操作——例如,这里,Id=LTC MOD 10将适合提供确定性结果,该结果可以指向任何可用的Id值。有权访问交易数据中的交易计数器值(或从该值导出的任何计数器)的任何验证服务V然后可以确定由生成服务G使用的逻辑密钥标识符,该生成服务G生成凭证并访问正确存储的密钥,而无需任何试错机制。以这种方式将确定性过程函数(以下称为keyList.GetIdFunction)与密钥列表的属性相关联允许可扩展的解决方案,该解决方案可以接受给定密钥列表的任意数量的逻辑密钥标识符。

[0099] 在实施例中使用LTC的密钥选择的整个过程在图14中示出并且还在图26中具有不同的表示。LTC是交易相关数据的一部分,如下面将看到的,它以几种方式用于确定凭证是否可以被生成或是否有效以供使用。LTC还用于计算用于选择密钥标签的值(其可以例如来

自密钥列表内的位置)。此标签与HSM中保持的特定密钥(keyId)相关联,并且HSM在其密码函数中使用此密钥来生成密码结果,该结果随后可以由另一方(同一节点或另一个节点中的验证服务)进行验证。

[0100] HSM密码函数应该适用于通过凭证生成和验证来确保数据完整性和认证。密码函数使用密钥对所选择的交易数据进行操作,并提供不暴露密钥的输出。可以使用各种替代密码函数——HMAC是特别有效的选择,但CMAC、CBC MAC是其中可能的替代方案。使用的密码函数应在密钥列表中指定(如keyList.CryptoFunction),并且还由用于生成和验证的HSM的能力驱动。土壤监管、密码材料出口或其它安全考虑可能导致选择特定的密码函数。

[0101] 在交易数据内,应该有表示在交易过程期间生成的应用密码的信息。这可能是密码的简化形式——例如,在传统交易中,这可以被提供为CVC2字段。这很重要,因为验证服务V必须能够访问生成服务G所使用的所有数据来生成密码——这将包括以下内容:

[0102] 作为交易流程的一部分传送的动态信息;

[0103] 来自以下之一的共享信息:

[0104] 复制过程(诸如密钥列表的管理);

[0105] 特定用例的系统参数。

[0106] 不同用例的标准方法——传统交易、UCAF和DPD字段交易——将在下面进一步讨论。当商家和/或PSP只能将PAN、到期日期和CVC2作为交易流程的一部分进行管理,并且无法访问最近的发展时,传统交易用例提供了解决方案。UCAF用例旨在利用最近引入的通用持卡人认证字段来传送更多数据作为交易流程的一部分。DPD用例涵盖数字支付数据的引入,这是能够传送作为交易流程一部分所需的所有数据的容器。

[0107] 一组完整的密码机制如图15中所示。参考图16讨论密钥管理。该模型中的密钥管理有两个方面:密钥本身的管理,包括它们的生成和递送到与节点相关联的HSM,以及密钥列表的管理,包括它们的生成、分发、激活和停用。密钥列表是敏感资产,而密钥被视为秘密资产——密钥列表定义了用于密码的生成和验证的密钥。在将密钥加载到HSM中时,密钥需要端到端的安全性,其中使用包装/解包技术来安全传输密钥。它们的使用不应受到密钥列表的影响,以防攻击者想要改变密钥列表的内容以更改密钥选择过程。密钥列表的完整性由密封锁保证——通过生成方或相关联的可信方为密钥列表提供密封锁,将涉及合适的密码过程(诸如,具有适当专用密钥或使用例如使用非对称算法(诸如RSA、ECC、SM2...)生成的数字签名的HMAC),并且其效果是系统的任何相关部分都可以确信密钥列表是由适当的一方生成的并且尚未被修改。此外,密钥列表密封锁可以用于密码的生成和验证以保护凭证。

[0108] 不同的控制模型是可能的。可能存在中央控制,其中中央服务生成密钥和密钥列表,并将它们分发到不同的节点。但是,如果在特定节点处需要专用过程,也可能存在本地化控制。如果对于特定国家存在特定要求,那么这可能特别适用——例如,对密码材料出口的土壤监管或限制。如果HSM管理需要专有机制——例如,特定的云服务提供商,那么这也可能适用。这不需要受节点限制——它可以适用于区域内具有中央服务的区域控制(这可能特别适用于特定国家有特定安全模型以满足当地法律要求的情况)。也可以存在混合或复合模型,其中一些密钥和密钥列表供给是中央的,而一些是本地的——也可能存在分布式模型,其中分布式对等点一起承担中央服务的角色。

[0109] 与图17相关讨论的监控可以具有本地和中央两个方面。虽然本地监控和中央化监控都是可能的,但混合方法可能特别有效,以便提供对任何问题的有效检测并产生有效应对以抵消与完全分布式体系架构相关联的风险。

[0110] 要考虑的监控主要有三类:分布式系统的完整性;交易凭证的生成;以及交易凭证的验证。由于交易凭证可以在任何地方生成或验证,因此对整个分布式系统进行有效监控是重要的。要考虑的风险包括攻击者滥用节点中生成服务G所生成的真实交易凭证,特别是尝试在其它节点中的多个验证服务中进行验证——这是一个问题,因为验证服务V通常不会实时查看分布式系统的其它节点中验证服务V采取的动作。

[0111] 虽然监控对于维护系统的完整性是重要的,但限制产生的消息传递量以确保系统可扩展并且不会因监控过程而过载也是重要的。可以进行多个选择来确保有效的监控过程。一个是针对由凭证生成服务G直接馈送的监控过程M。另一个是:由于任何验证服务V都可以验证交易凭证,在任何给定节点处为所有验证服务V提供共同存储。这允许监控服务M本地的相关联数据管理以提供初始防御层,特别是针对重放检测。

[0112] 然后,本地监控服务可以能够经由中央监控服务(中心辐射(hub and spoke))或直接(对等(peer to peer))向其它节点报告,从而向作为第二层防御的其它验证服务V报告,以防止跨节点滥用交易凭证。

[0113] 验证服务本身显然会检查交易凭证是否有效,并且可以级联检测任何问题(诸如验证失败、过度重试以及无效的交易数据或密钥列表引用)——这些可以用于跨节点暂停令牌。还可以执行附加控制,诸如对密钥列表密封锁的验证进行随机检查。另一个监控过程是不同服务类型之间的关联——这里是生成G和验证V——以检测交易凭证是否已经丢失(由于未提交、异常生成或其它原因)。可以跟踪密码材料的使用以跟踪它被适当地使用并且系统被适当地配置——例如,可以跟踪使用给定存储的密钥的密码操作的数量。

[0114] 如上所述,有效监控要遵循的一个原则是提供有效的解决方案,而不会对分布式系统造成过多的负载。这可以通过支持不同级别的信息(例如,在适当的情况下进行本地监控,仅进一步传达本地监控活动的摘要)并通过数据复制和共享减少负载来完成。

[0115] 监控可能因节点而异,或因地区而异,其中存在特定的要求。可以在给定节点或给定地区使用特定的监测过程来解决给定地理区域中的土壤或个人可识别信息(PII)要求。

[0116] 所涉及的挑战是在交易中有效识别凭证是如何生成的,以便实现它们的后续验证——特别是识别哪个节点生成了凭证以及使用了哪个密钥列表来完成它,以及本地交易计数器的状态。这是具有挑战性的,因为交易数据受到高度限制,并且为了提供任何此类信息,有必要更改现有的电子交易协议(诸如ISO 8583)或重新利用现有字段。

[0117] 对于传统的电子交易协议,原则上可以被重新利用的字段是主账号(PAN),因为PAN内的一些数字位在此类交易的上下文中可能是隐含的,并且因此可以被重复使用,到期日期,其中一些信息可以以压缩格式以及CVC2传送。可以使用到期日期作为载体直接释放六比特,但这还不够——对于任何扩展系统,节点标识符通常至少需要四比特,并且一比特可能不足以用于密钥列表引用或交易计数器。

[0118] 可以使用的一种方法是使用一组特定的银行信息号码(BIN),其构成PAN中的前六个数字位,以支持上述实施方式——当检测到这些BIN中的一个时,可以采用特殊处理。这可能涉及将令牌与多个PAN值相关联。该模型如图18中所示。FPAN(资金主账号——对应于

实体卡账户)——可以映射到一个或多个令牌,但特定的令牌与特定技术相关联。顶行示出了常规的令牌化过程——FPAN与单个令牌相关联。在使用上述方法的情况下,对于传统接受用例(底线),令牌可以与九个PAN值相关联,但是如下所述,对于某些新格式,仍然可以使用一对一映射。

[0119] 因此,传统情况下交易字段的重用可以如下。对于PAN,可以使用14个数字位来完全识别令牌,其中1个数字位用于与给定数字的令牌相关联的计数器,并且1个数字位用于Luhn数字(其需要被保留为校验和以确保使用有效数字)。到期日期的6比特可以被重新利用,其中x比特用于识别节点,y比特用于引用该节点的相关密钥列表。CVC2提供可以用于密码的三个数字位。

[0120] 为了安全,期望定期更改密钥列表,以确保系统安全免受攻击。能够在凭证创建后的一段时间内允许对其进行验证也是重要的——建议的方法是允许在创建后最多24小时内验证凭证。如果这与每24-36小时操作一次的密钥轮换过程相结合,这意味着虽然生成过程对于给定节点将仅有一个活动密钥列表,但验证过程将仅需要考虑两个密钥列表(一个当前活用于凭证生成,另一个紧接在它之前活动)。使用基于交易计数器建立的确定性过程从而建立要使用的密钥。这种类型的二进制信息(即,一个或另一个)通常可以使用一比特信息进行编码。密码在保护交易完整性方面起着关键作用——成功验证使用正确密钥对给定数据集计算的密码确认最初用于凭证生成的数据是真实的。验证过程中的任何失败都可能来自使用错误的密码材料和/或损坏的交易数据。

[0121] 图19中示出了这种传统布置的示例性密钥轮换过程。首先,根据需要向HSM提供新密钥——这可能是通过从中央源分发,或通过另一个密钥生成过程,例如通过本地密钥生成进行的。然后生成新密钥列表——这可能涉及现有密钥和新密钥——这里,密钥列表中的大多数插槽(slot)涉及密钥列表中新位置中的现有密钥(但是密钥也可能保留在密钥列表中的相同位置——如这里针对密钥列表位置2和6所示),但是在位置3、7和8中也使用了新密钥。同样,这可能是中央过程,一个由分布式网络中的对等点管理,或者一个本地管理。新的密钥列表被分发到有资格进行验证的任何验证服务以及使用它来生成的单个生成服务。新密钥列表然后在验证服务中被激活,并且然后在生成服务中被激活,这自动停用该生成服务中的先前活动密钥列表——此时密钥轮换过程完成。在24小时后,先前密钥列表然后针对验证服务被停用。这种方法适用于传统情况可用的有限空间——只需切换单个比特即可指示使用了哪个密钥列表。

[0122] 一个潜在的问题是,为给定令牌执行的交易数量看起来受到可用于传送交易计数器的空间的限制。一种解决方案是增加交易计数器的可用空间,但这将对应地限制可用令牌的数量(因为这是附加比特可能来自的唯一地方)。另一种可能性将是如果验证最初未能恢复“完整计数器”值,其中该值大于原始可用空间并且仅作为模值存储,那么使用“重试”过程。可以组合这两种方法。还可以为不同的令牌范围引入不同的规则。

[0123] 在每个节点中,每个生成(G)和验证(V)服务都可以访问本地数据库。给定令牌的任何交易凭证的生成都与每个交易的唯一交易标识符相关联。如上所述,对于给定节点中的给定令牌,本地交易计数器(LTC)由“G”使用与给定用例关联的给定密钥列表进行管理。同样的过程适用于由“V”验证时。该信息可以在PAN字段(数字位15,或数字位14和15)中传送,如图21中所示或如图22中使用CV2字段所示,其中到期日期字段中有重试标志,如果LTC

处于较高值,那么在必要时生成“完整计数器”。但是,重要的是要限制对于给定密钥列表对于给定节点对于给定令牌可以由G生成并且由V验证的密码数量,以确保有效的访问控制——该值“MaxTransactionCounter”可以被存储在密钥列表中并由密钥列表密封锁保护。

[0124] 图20中示出了此传统情况的密码过程。在这种情况下,选择HMAC作为密码函数,因为这允许在递送有效功能的同时使用通用HSM。令牌的标识使用PAN值。交易的标识从到期日期(ISO 8583字段DE14)获取信息——特别是节点标识符和引用,可能还带有重试标志——并且来自保持本地交易计数器的PAN字段。密钥的标识和密码方法由本地交易计数器(其确定从密钥列表中选择哪个密钥)以及由密钥管理系统在密钥列表中共享的信息一起提供。定义交易的各种字段可以用作用于生成密码的金融数据(如图18中所示),所有这些字段都用于生成密码,该密码然后被十进制化并在CVC2字段中使用三位最低有效数字。

[0125] 本领域技术人员将认识到,这些协议的一些变体可以优先考虑某些选择或优先级。例如,可以考虑期望找到更有效的方式来传送数据,诸如在交易流程中可以传送更多数据时避免使用重试过程的本地交易计数器——如从图21中可以看出,上面显示的过程允许交易计数器最多使用PAN的两个数字位(并且使用两个数字位限制了可以提供的令牌数量),并在三个CVC2数字位中保持了缩减的密码。一种不同的方法是仅使用密码中的CVC2字段的两个数字位而不是三个数字位,其中另一个数字位用于保持本地交易计数器的最右边的数字位。这可以通过将三个数字位重新布置为不同的顺序以更动态的方式提供——这可以通过将CVC2编码表添加到密钥列表中来完成,使得在使用密钥列表时,编码表——也受密封锁保护——确定为提供CVC2字段而选择的编码。可以通过G和V服务两者都已知的任何值来选择代码——例如,PAN的Luhn数。然后,新密钥列表可以使用完全不同的编码表,从而使该过程具有显著的动态性。

[0126] 这种布置如图22中所示。PAN数字位识别令牌并且还提供Luhn数,并且Luhn数用于确定CVC2字段的数字位顺序——在这种情况下,选择选项3,指示密码在前两个位置的最低有效数字位和次最低有效数字位,计数器的最低有效数字位在第三位置。这导致G和V服务两者都可以导出CVC2输出。

[0127] 如果使用较新版本的电子交易协议,那么可以使用其它字段来传送更多信息。例如,在通用持卡人认证字段(UCAF)可用的情况下(ISO 8583DE48 SE 43),可以使用许多附加字节,从而允许避免在传统情况中使用的妥协。这种方法可以释放另外21个字节的数据来传送数据作为交易流程的一部分。这足以允许传送完整的本地交易计数器值,从而避免需要任何重试机制。可以使用更多的密码材料——8字节的密码,而不是2个或3个数字位。可以使用更多的节点,而不会因为电子交易协议要求中定义的交易数据中的可用空间有限而导致节点识别成为问题。也可以比24小时更频繁地轮换密钥列表,因为存在空间来使用多于一个的比特用于验证服务的密钥列表标识。可以利用交易数据中的可用空间提供附加特征,例如通过支持商家锁定技术(在使用某种形式的商家标识将交易有效绑定到给定商家的情况下)、通过在密码过程中包含附加组件(诸如通过在生成器和验证器之间使用一些随机元素或种子)、或通过采取附加措施来提供符合任何监管要求的完全合规性。

[0128] 如从图23中可以看出,使用UCAF内容的新布局(例如,格式7—这是申请人专有的UCAF格式的标识)有21个字节可用。可以在版本标识符和码本之间拆分一个字节以指定密

码生成中使用的条件数据。完整的字节可以用于保持本地交易计数器——这意味着生成服务G将能够针对给定节点针对给定令牌每个密钥列表生成最多(up to)255个密码,这应该可以防止需要重试计数器并且在激活新密钥列表之前解决交易凭证的需求。另一个字节对于节点标识符数据和密钥列表引用是足够的,这为密码生成和/或验证中使用的条件数据留下完整的10个字节——每个用例都与码本中的值相关联——从而允许使用与授权消息中传送的数据不同的数据(传送的数据可以包括用于交易的不可预测的号码、商家数据,诸如商家类型和卡接受者或收单机构ID代码、金额相关信息…)。8个字节可以用于截断密码,从而显著提高安全性。图24指示密码过程如何与图20中所示的不同——PAN、LTC、节点标识符和引用都可以被轻松地包含在内,并且可以在密码的生成(或验证)中包括附加信息,诸如附加交易字段、码本和其它条件数据。

[0129] 这种方法提供了各种进一步的可能性。为密钥列表引用供给附加比特允许两倍频繁的密钥列表轮换。虽然一些要求仍然存在——诸如需要限制服务G针对给定密钥列表针对给定节点的给定令牌生成的密码数量——但其它要求被免除(完整LTC的存在意味着不需要任何重试过程)。应该注意的是,密钥列表可能仅限于特定用例——传统、UCAF或DPD——并且这可以用于确定指定用例的交易计数器的特定限制。

[0130] 不久将引入称为DPD(数字支付数据)的新格式——这甚至将提供更多的选项,如图25中所示。如上所述,DPD可以将UCAF传送到凭证。在第一个选项中,生成服务可以在交易创建时定义唯一标识符(诸如,UUID),并将其添加到密码过程中使用的数据列表——这允许交易凭证独立于交易详细信息进行端到端跟踪,从而提供监控和欺诈预防优势。在第二个选项中,可以使用大得多的码本,以及节点的扩展标识、更多条件数据的存储和使用如在其它交易流程中使用的标准截断密码——以及用于第一个选项的UUID。

[0131] 现在将更详细地描述本公开的实施例中的本地交易计数器(LTC)的作用。如上所述,LTC有助于为交易供给唯一标识符。LTC本身不是唯一的,但是当与其它值组合时——例如如上所述的节点标识符和时间段标识符,但也可能是其它值,诸如密钥列表标识符和PAN/TUR——它可以提供唯一标识符,特别是使用具有给定PAN/TUR的给定密钥列表的给定节点执行的交易的唯一标识符。还如上所述,LTC还可以用于提供从密钥列表中选择密钥用于密码生成和验证的确定性手段。

[0132] 除了这些功能之外,LTC还可以用于跟踪与交易相关的各种活动,并且可以在交易字段中只能传送有限数据的情况下(诸如上面讨论的传统用例)提供特定的好处。这与重放检测(提交已经用于给定节点和密钥列表的针对给定LTC值的交易凭证的验证)和跟踪失败的密码和重试(其中需要多于一次的验证尝试)相关。

[0133] 下面针对所有用例(传统——L、UCAF——U、DPD选项1——D1;以及DPD选项2——D2)讨论LTC管理。在传统情况下,如以上所讨论的,动态到期日期字段用于传送与LTC相关的附加信息——动态到期日期对验证过程的影响也在下面讨论。

[0134] 首先,将讨论本地交易计数器在生成服务G和验证服务V处的基本操作。

[0135] 生成服务

[0136] LTC在服务性能和在生成服务G的数据库(db_G)中服务操作的记录方面具有关键作用。该数据库用于使用利用keyList.Identifier识别出的给定活动密钥列表来跟踪给定节点(N_i)的各个LTC值。当第一次为给定的PAN或TUR(下文中称为PAN/TUR)生成交易时,在数

据库中创建条目。该条目仅包含一个LTC值,并在该PAN/TUR的任何后续交易凭证生成时使用该给定节点中的该给定密钥列表进行更新。

[0137] 这样做的过程如下。首先,为第一个生成的交易凭证建立LTC的默认值,并在数据库(db_G)中为给定的PAN/TUR创建条目,如所指示的。对于该PAN/TUR的任何后续交易凭证生成,计数器将递增,直到该PAN/TUR达到LTC的限制值为止。这个限制值可以在密钥列表(keyList.Limit.LTC)中定义。然后,交易凭证生成服务G将停止使用该密钥列表为该PAN/TUR生成交易凭证,直到该节点的新密钥列表变为活动为止。

[0138] 验证服务

[0139] 凭证验证服务(V)还使用数据库(db_V),该数据库使用已经过验证的凭证的LTC值。数据库使用利用keyList.Identifier识别出的给定—活动—密钥列表来存储任何给定节点(N_i)的LTC值列表。当使用与给定节点相关联的给定密钥列表首次验证给定PAN/TUR的交易凭证时,将创建条目。数据库(db_v)中的每个条目都与LTC值列表、计数器列表(重放、CryptoFailure和重试——全部默认为0并通过适当的事件递增,如下所述)相关联。使用与给定节点相关联的给定密钥列表对该PAN/TUR的交易凭证进行的任何后续验证将导致更新数据库条目。在停用密钥列表时,当使用该密钥列表生成的凭证无法再合法验证时,用于验证由给定节点(N_i)使用keyList.Identifier为该密钥列表生成的交易凭证的数据库(db_v)的部分内容将被删除。生成器使用的密钥列表的停用与生成的交易凭证的(一个或多个)验证器的该密钥列表的停用之间存在延迟。这种延迟是由业务规则驱动的,从而允许例如在交易凭证的生成和其有效验证之间长达24小时。

[0140] 除了在交易已成功时修改条目外,如下面进一步描述的密码验证过程还将更新数据库(db_V)的内容,以用于附加目的:检测和跟踪重放;跟踪密码失败;以及跟踪重试次数。

[0141] 现在将更详细地描述验证过程,并特别关注LTC管理问题。在这种上下文中,验证过程涵盖以下内容:

[0142] ●服务请求管理

[0143] ●采集和处理交易相关数据,包括:

[0144] ○令牌的标识(PAN/TUR)

[0145] ○交易的标识(使用LTC)

[0146] ○密钥列表的标识(使用节点信息和密钥列表的标识符)

[0147] ○交易密钥和密码函数的标识

[0148] ○任何金融或其它数据的处理

[0149] ●密码的验证

[0150] ●交易凭证的验证

[0151] ●任何用例(L、U、D1和D2)通用的过程

[0152] ●所选择的用例(L、U、D1或D2)的特定过程

[0153] ●向本地监控报告(mV)

[0154] 如以上所指示的,本文档仅关注于LTC使用,因此这里不详细描述验证过程的其它方面。

[0155] 如上所述,密钥列表可以包含LTC的限制值(keyList.Limit.LTC),以便控制凭证生成服务G在给定时间段在给定节点中使用给定密钥列表针对给定PAN/TUR可以生成的交

易凭证的数量。验证过程也可以使用这个限制值。密钥列表还可以包含附加字段以使得能够动态管理其它限制。

[0156] 如果我们考虑由以下使用的密钥列表：

[0157] ●节点N_i,用于生成交易凭证

[0158] ●节点N_i(或任何符合条件的N_j),用于验证交易凭证

[0159] 由Vx完成的验证过程使用密钥列表来支持：

[0160] ●重放管理(keyList.Limit.Replay)

[0161] ○值Limit.Replay用于限制在可以暂停PAN/TUR之前重放尝试的次数

[0162] ○Counter.Replay由节点(N_i或任何符合条件的N_j)用于跟踪由给定节点(N_i)使用与N_i相关联的给定密钥列表为给定PAN/TUR生成的交易凭证的验证的重放尝试的次数。

[0163] ●密码失败管理(keyList.Limit.CryptoFailure和keyList.Limit.CryptoFailureReset)

[0164] ○值Limit.CryptoFailure用于限制在可以暂停PAN/TUR之前密码验证失败的次数。

[0165] ○Counter.CryptoFailure由节点(N_i或任何符合条件的N_j)用于跟踪由给定节点(N_i)使用与N_i相关联的给定密钥列表为给定PAN/TUR生成的交易凭证的密码验证失败的次数。

[0166] ○值Limit.CryptoFailureReset是布尔参数,用于确定是否可以使用有效的密码来重置Counter.CryptoFailure。

[0167] ●重试管理(keyList.Limit.Retry)

[0168] ○值Limit.Retry用于限制在可以暂停PAN/TUR之前的重试次数。

[0169] ○Counter.Retry由节点(N_i或任何符合条件的N_j)用于跟踪在给定节点(N_i)使用与N_i相关联的给定密钥列表为给定PAN/TUR生成的交易凭证的密码验证失败后的重试的次数。

[0170] 如本领域技术人员将认识到的,暂停PAN/TUR是可以用于保护PAN/TUR免受任何欺诈或异常使用的一种对策。

[0171] 重试仅用于传统(L)用例——这在使用允许传送本地交易计数器(LTC)的全部值的格式(诸如UCAF、DPD选项#1或DPD选项#2)时不使用。这些限制和计数器特定于节点——它们不会跨所有节点进行整合。但是,应该注意的是,可以使用协调的监控过程跨所有可能受影响的节点分发与这些计数器相关的信息。

[0172] 每个验证过程(Vj、Vk...)都有其自己的LTC列表和与由给定节点使用给定密钥列表生成的交易凭证对应的每个处理后的PAN/TUR的计数器。图27中描绘了这种布置。

[0173] 图27给出了使用其自己的密钥列表的两个生成节点(Ni和Nx)的示例。

[0174] ●Vj能够验证由Ni生成的交易凭证。

[0175] ●Vk能够验证由Ni和Nx生成的交易凭证。

[0176] ●Vj和Vk跟踪处理后的PAN/TUR,在密钥列表/节点级别管理

[0177] 计数器和LTC的列表：

[0178] ●Vj存储与Ni对应的密钥列表的PAN/TUR相关信息

[0179] ●Vk存储：

[0180] ●与Ni对应的密钥列表的PAN/TUR相关信息

[0181] ●与Nx对应的密钥列表的PAN/TUR相关信息

[0182] 密钥列表的停用触发存储信息的删除。这可以看作是与给定密钥列表相关联的所有计数器的重置。现在将更详细地考虑验证过程动作。

[0183] 验证过程的结果在验证密码时报告以下信息——应注意的是,在传统用例中需要与重试过程和动态到期日期管理相关的附加信息。

[0184]

字段	值	描述	用例			
			L	U	D1	D2
cryptoValidation（密码验证）	SUCCESS	密码验证成功	x	x	x	x
	FAILURE	密码验证失败	x	x	x	x
	FAILURE_EXCEEDED	密码验证失败并超出限制	x	x	x	x
replayDetection（重放检测）	NO_REPLAY	在密码验证期间未检测到重放	x	x	x	x
	REPLAY	在密码验证期间检测到重放	x	x	x	x
	REPLAY_EXCEEDED	在密码验证期间检测到重放并且超出限制	x	x	x	x
retryControl（重试控制）	NO_RETRY	在密码验证期间不使用重试	x	-	-	-
	RETRY	密码验证期间使用重试	x	-	-	-

[0185]

	RETRY_EXCEEDED	在密码验证期间使用重试并且超出限制	x	-	-	-
expiryDate（到期日期）	STANDARD	使用标准参考时间处理到期日期来完成验证	x	-	-	-
	SPECIAL	使用特殊机制处理到期日期来完成验证	x	-	-	-

[0186] 表1——验证结果

[0187] 验证过程可以使用函数SetValidationOutcome()来采集关于交易凭证的验证的信息。该信息将由验证服务V使用,但也由监控服务(其可能是节点本地的、跨多个节点或全局协调的,或两者)使用。重放和密码失败的限制委托以及重试的管理限制将在下面进一步讨论。

[0188] 当超出相关联的限制(Limit.Replay、Limit.CryptoFailure或Limit.Retry)时使用通知过程。这可能导致暂停PAN/TUR。

[0189] 下面在与动态到期日期对验证过程的影响相关的部分中介绍了使用(STANDARD(标准)或SPECIAL(特殊))到期日期的概念——标准是到期日期默认的值。

[0190] 现在将讨论无需重试即可管理与LTC相关的验证过程——这与所有用例相关。

[0191] 重放和密码失败的限制按照如下进行管理。如果PAN/TUR未暂停,那么以下过程适用于节点N_i生成的交易凭证的验证。PAN/TUR的暂停将被传达给验证服务。

[0192] ●使用“该”LTC值

[0193] ●SetValidationOutcome(replayDetection=NO_REPLAY,retryControl=NO_

RETRY)

[0194] ●密码验证

[0195] ○密码(成功)

[0196] ■SetValidationOutcome(cryptoValidation＝SUCCESS)

[0197] ■LTC不在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0198] -将LTC添加到数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0199] -如果Limit.CryptoFailureReset,重置Counter.CryptoFailure[N_i,keyList.Identifier,PAN/TUR]

[0200] ■LTC已经在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0201] -递增(或创建)Counter.Replay[N_i,keyList.Identifier,PAN/TUR]

[0202] -Counter.Replay<Limit.Replay

[0203] ●SetValidationOutcome(replayDetection＝REPLAY)

[0204] -Counter.Replay≥Limit.Replay

[0205] ●SetValidationOutcome(replayDetection＝REPLAY_EXCEEDED)

[0206] ●通知欺诈管理服务(可能导致暂停suspending PAN/TUR)

[0207] ○密码(失败)

[0208] ■SetValidationOutcome(cryptoValidation＝FAILURE)

[0209] ■递增(或创建)Counter.CryptoFailure[N_i,keyList.Identifier,PAN/TUR]

[0210] ■Counter.CryptoFailure≥Limit.CryptoFailure

[0211] -SetValidationOutcome(cryptoValidation＝FAILURE_EXCEEDED)

[0212] -通知欺诈管理服务(可能导致暂停PAN/TUR)

[0213] ■LTC不在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0214] -将LTCM添加到数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0215] ■LTC已经在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0216] -递增(或创建)Counter.Replay[N_i,keyList.Identifier,PAN/TUR]

[0217] -Counter.Replay<Limit.Replay

[0218] ●SetValidationOutcome(replayDetection＝REPLAY)

[0219] -Counter.Replay≥Limit.Replay

[0220] ●SetValidationOutcome(replayDetection＝REPLAY_EXCEEDED)

[0221] ●通知欺诈管理服务(可能导致暂停PAN/TUR)

[0222] 这个过程允许有效地建立限制,并在超出限制时通知欺诈管理服务。

[0223] 可能的选项是已经在密码成功(成功的密码验证)之后重置用于跟踪密码失败的计数器。密钥列表中可以存在值-Limit.CryptoFailureReset用于此目的。可能的替代方案是:

[0224] ●考虑为给定密钥列表的生命周期定义的计数器及其相关联的限制,无需任何重置[假]。

[0225] ●放宽这些规则,并将有效的密码视为重置的充分条件[真]。

[0226] 可以如下实现重放和密码失败的限制委托。同样,如果PAN/TUR未暂停,那么以下过程适用于验证为节点N_i生成的交易凭证。在这种情况下,验证服务V不会根据相关联限制

本身来执行重放或密码失败的检查,但V正在委托它。该过程如下:

[0227] ●使用“该”LTC值

[0228] ●SetValidationOutcome(replayDetection＝NO_REPLAY,retryControl＝NO_RETRY)

[0229] ●密码验证

[0230] ○密码(成功)

[0231] ■SetValidationOutcome(cryptoValidation＝SUCCESS)

[0232] ■LTC不在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0233] -将LTC添加到数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0234] -如果Limit.CryptoFailureReset,重置Counter.CryptoFailure[N_i,keyList.Identifier,PAN/TUR]

[0235] ■LTC已经在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0236] -递增(或创建)Counter.Replay[N_i,keyList.Identifier,PAN/TUR]

[0237] -SetValidationOutcome(replayDetection＝REPLAY)

[0238] ○密码(失败)

[0239] ■SetValidationOutcome(cryptoValidation＝FAILURE)

[0240] ■递增(或创建)Counter.CryptoFailure[N_i,keyList.Identifier,PAN/TUR]

[0241] ■LTC不在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0242] -将LTC添加到数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0243] ■LTC已经在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0244] -递增(或创建)Counter.Replay[N_i,keyList.Identifier,PAN/TUR]

[0245] -SetValidationOutcome(replayDetection＝REPLAY)

[0246] 用于在成功的密码验证后跟踪密码失败的计数器的重置是使用来自密钥列表中的Limit.CryptoFailureReset值定义的选项,如上面的部分所述。

[0247] 另一个实体将使用以下过程管理对于Replay和CryptoFailure的限制,该过程可以覆盖由V设置的验证结果:

[0248] ●Counter.Replay≥Limit.Replay

[0249] ○SetValidationOutcome(replayDetection＝REPLAY_EXCEEDED)

[0250] ○通知欺诈管理服务(可能导致暂停PAN/TUR)

[0251] ●Counter.CryptoFailure≥Limit.CryptoFailure

[0252] ○SetValidationOutcome(cryptoValidation＝FAILURE_EXCEEDED)

[0253] ○通知欺诈管理服务(可能导致暂停PAN/TUR)

[0254] 这允许V将这些检查有效委托给另一个实体,诸如欺诈管理服务或将执行这些检查以减少验证器(V)上的负载的任何专用服务。

[0255] 现在将讨论传统用例的LTC管理的特殊考虑。由于捕获LTC和其它数据的可用空间有限所导致的问题,传统用例(L)明显更复杂,如上文参考图19至图22所示。UCAF(U)和数字支付数据选项#1(D1)和选项#2(D2)用例有更多的可用空间——例如,在UCAF中,LTC可以作为交易数据的一部分(例如,作为格式7中的1个字节,其中条件数据可用于进一步空间)传送。这避免了在传统用例中需要解决的各种问题(诸如,恢复本地交易计数器的全部值,并

在密码验证失败的情况下重试)。在DPD中,还有甚至更多可用空间(2个字节加上条件数据)。因此,下面的内容通常与传统用例(L)相关,但是下面肯定地指出它也适用于UCAF和DPD用例(诸如验证而无需重试)。

[0256] 如前所述,传统用例(L)对可以作为交易数据的一部分传送的数据有严格的大小限制,并且只能传送LTC值的一部分——通常是一个数字位(C)。这意味着必须采用恢复过程来有效且可靠地恢复LTC数据。对于传统用例,验证过程有两种风格——“无重试过程”,如上所述并用于所有其它用例,以及特定于传统用例的“重试过程”。从一个值C开始,对每个的需求如下识别:

[0257] 如果我们基于密钥列表(keyList.Limit.LTC)中定义的LTC限制将KLmax视为LTC的最大值,并将RF作为使用到期日期传送的重试标志,我们有:

[0258] ●KLmax≤19:C导致一个LTC值[无重试-以上阐述的过程]

[0259] ■RF=0

[0260] ●LTC=0$\underline{C}$

[0261] ■RF=1

[0262] ●LTC=1$\underline{C}$

[0263] ●KLmax>19

[0264] ■RF=0:C导致一个LTC值[无重试-以上阐述的过程]

[0265] ●LTC=0$\underline{C}$

[0266] ■RF=1:C导致LTC值的一个或多个候选[重试-下面描述的过程]

[0267] ●LTC候选=(1$\underline{C}$,2$\underline{C}$,…,n$\underline{C}$)with n$\underline{C}$≤KLmax

[0268] 上面的逻辑假设我们可以信任节点的标识、密钥列表的引用和使用到期日期传送的重试标志的值。

[0269] 对于V来说,恢复该信息可能具有挑战性——下面将进一步描述动态到期日期对验证过程的影响。

[0270] 其它考虑因素通常与传统(L)用例相关,该用例由于缺乏可用于在交易中传送LTC数据的空间而变得复杂。存在两个主要考虑的领域。一个是限制管理——可以管理重试限制,而重放限制可以在替代选项中管理或委托。另一个考虑的领域是动态到期日期的使用及其后果,特别是管理月末的困难,其中可以在其它时间有效重用的某些字段变得重要。

[0271] 首先,考虑重试过程,其中验证节点V管理重放和重试两者的限制。如果PAN/TUR未暂停,那么以下过程适用于验证由节点N_i生成的交易凭证。V使用以下过程来管理重放和重试的限制:

[0272] ●SetValidationOutcome(replayDetection=NO_REPLAY,retryControl=NO_RETRY)

[0273] ●对候选列表进行排序,使得我们优先考虑尚未在存储在数据库(dbV)[N_i,keyList.Identifier,PAN/TUR]中的LTC列表中的(一个或多个)候选。

[0274] 对值进行排序旨在解决候选列表之间的冲突风险(即,具有相同的密码值)。使用此优先级过程,我们尝试避免在另一个有效候选尚未用于密码验证时报告重放(其通常指示交易已经被拒绝)。

[0275] ●使用从LTC_1开始的(LTC_1,LTC_2,…,LTC_n)的LTC进行循环

[0276] ○密码验证

[0277] ■密码(成功)

[0278] -SetValidationOutcome(cryptoValidation＝SUCCESS)

[0279] -LTC不在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0280] ●将LTC添加到数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0281] ●如果Limit.CryptoFailureReset,重置

[0282] Counter.CryptoFailure[N_i,keyList.Identifier,PAN/TUR]

[0283] -LTC已经在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0284] ●递增或(创建)Counter.Replay[N_i,keyList.Identifier,PAN/TUR]

[0285] ●Counter.Replay<Limit.Replay

[0286] ■SetValidationOutcome(replayDetection＝REPLAY)

[0287] ●Counter.Replay≥Limit.Replay

[0288] ■SetValidationOutcome(replayDetection＝REPLAY_EXCEEDED)

[0289] ■通知欺诈管理服务(可能导致暂停PAN/TUR)

[0290] -退出循环

[0291] ■密码(失败)

[0292] -SetValidationOutcome(cryptoValidation＝FAILURE)

[0293] -递增或(创建)Counter.Retry[N_i,keyList.Identifier,PAN/TUR]

[0294] -Counter.Retry<Limit.Retry

[0295] ●SetValidationOutcome(retryControl＝RETRY)

[0296] -Counter.Retry≥Limit.Retry

[0297] ●SetValidationOutcome(retryControl＝RETRY_EXCEEDED)

[0298] ●通知欺诈管理服务(可能导致暂停PAN/TUR)

[0299] ●退出循环

[0300] -任何剩余候选LTC_i?

[0301] ●可用

[0302] ■利用LTC_i继续循环(密码验证)

[0303] ●不可用

[0304] ■退出循环

[0305] 这种方法允许对重放和重试两者进行有效管理——具有重置重放的选项——在超过限制时启用升级。应该注意的是,“重放”不适用于密码失败——引入Counter_Retry以限制节点使用与该节点相关联的给定密钥列表处理给定PAN/TUR的重试次数。如前所述,在成功密码验证之后重置密码失败计数器是可以使用的选项,并且可以使用Limit.CryptoFailureReset在密钥列表中定义。

[0306] 委托重放限制的替代过程如下:

[0307] ●SetValidationOutcome(replayDetection＝NO_REPLAY,retryControl＝NO_RETRY)

[0308] ●对候选列表进行排序,使得我们优先考虑尚未在存储在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中的LTC列表中的(一个或多个)候选。

[0309] 对值进行排序旨在解决候选列表之间的冲突风险(即,具有相同的密码值)。使用此优先级过程,我们尝试避免在另一个有效候选尚未用于密码验证时报告重放(与交易的拒绝相关联)。

[0310] ●使用从LTC_1开始的(LTC_1,LTC_2,…,LTC_n)的LTC进行循环

[0311] ○密码验证

[0312] ■密码(成功)

[0313] -SetValidationOutcome(cryptoValidation=SUCCESS)

[0314] -LTC不在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0315] ●将LTC添加到数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0316] ●如果Limit.CryptoFailureReset,重置

[0317] Counter.CryptoFailure[N_i,keyList.Identifier,PAN/TUR]

[0318] -LTC已经在数据库(db_V)[N_i,keyList.Identifier,PAN/TUR]中

[0319] ●递增或(创建)Counter.Replay[N_i,keyList.Identifier,PAN/TUR]

[0320] ●SetValidationOutcome(replayDetection=REPLAY)

[0321] -退出循环

[0322] ■密码(失败)

[0323] -SetValidationOutcome(cryptoValidation=FAILURE)

[0324] -递增或(创建)Counter.Retry[N_i,keyList.Identifier,PAN/TUR]

[0325] -Counter.Retry<Limit.Retry

[0326] ●SetValidationOutcome(retryControl=RETRY)

[0327] -Counter.Retry≥Limit.Retry

[0328] ●SetValidationOutcome(retryControl=RETRY_EXCEEDED)

[0329] ●通知欺诈管理服务(可能导致暂停PAN/TUR)

[0330] ●退出循环

[0331] -任何剩余候选LTC_i?

[0332] ●可用

[0333] ■利用LTC_i继续循环(密码验证)

[0334] ●不可用

[0335] ■退出循环

[0336] 其它方面的考虑与初始过程中的一样。在委托的情况下,另一个实体将使用以下过程管理Replay的限制,该过程可以覆盖由V设置的验证结果:

[0337] ●Counter.Replay≥Limit.Replay

[0338] ○SetValidationOutcome(replayDetection=REPLAY_EXCEEDED)

[0339] ○通知欺诈管理服务(可能导致暂停PAN/TUR)

[0340] 以下讨论涉及在传统(L)用例中使用“动态到期日期”来传送信息。到期日期字段用于通过将exp月份添加到使用t_x(UTC)作为参考计算的下个月值(YYMM)来传送6比特值(exp)。

[0341]

b6	b5	b4	b3	b2	b1	描述
x						密钥列表引用

	x					重试标志
		x	x	x	x	节点标识符

[0342] 工作示例如下：

[0343]

密钥列表引用	0 （0b）	010011b = 19
重试标志	1 （1b）	
节点标识符	3 （0011b）	

[0344] ●t_X(10:30:00AM CST,星期三,2019年6月26日)＝03:30:00PM UTC,星期三,2019年6月26日

[0345] ●基于t_X的下个月(YYMM)(UTC)＝1907

[0346] ●动态到期日期＝1907“+”19＝2102

[0347] 动态到期日期由G计算作为对于“L”用例的交易凭证生成的一部分——这里使用它是因为PAN、到期日期和CVC2是商家/PSP及其收单方可以处理的最小数据集,因此需要一些机制来传送附加的必要信息。

[0348] 在大多数情况下,可以使用简单的确定性过程来可靠地提取信息。G知道与生成交易凭证对应的时间t_G。

[0349] ●时间t_G可以被转换成UTC时区,其可以用作整个系统的参考

[0350] ●这个转换后的值可以明确地确定“下个月”的值

[0351] ●动态到期日期是“下个月”和与上述6比特信息对应的值的组合。

[0352] 简而言之,对于给定的交易凭证生成,我们可以为使用确定性过程建立的动态到期日期设置一个值。验证服务V遵循相同的逻辑,但用作与交易凭证的验证对应的参考时间t_V。

[0353] 如本领域技术人员将认识到的,上述实施例是示例性的,并且本领域技术人员根据以上阐述的原理和示例工作可以开发落入本公开的精神和范围内的其它实施例。

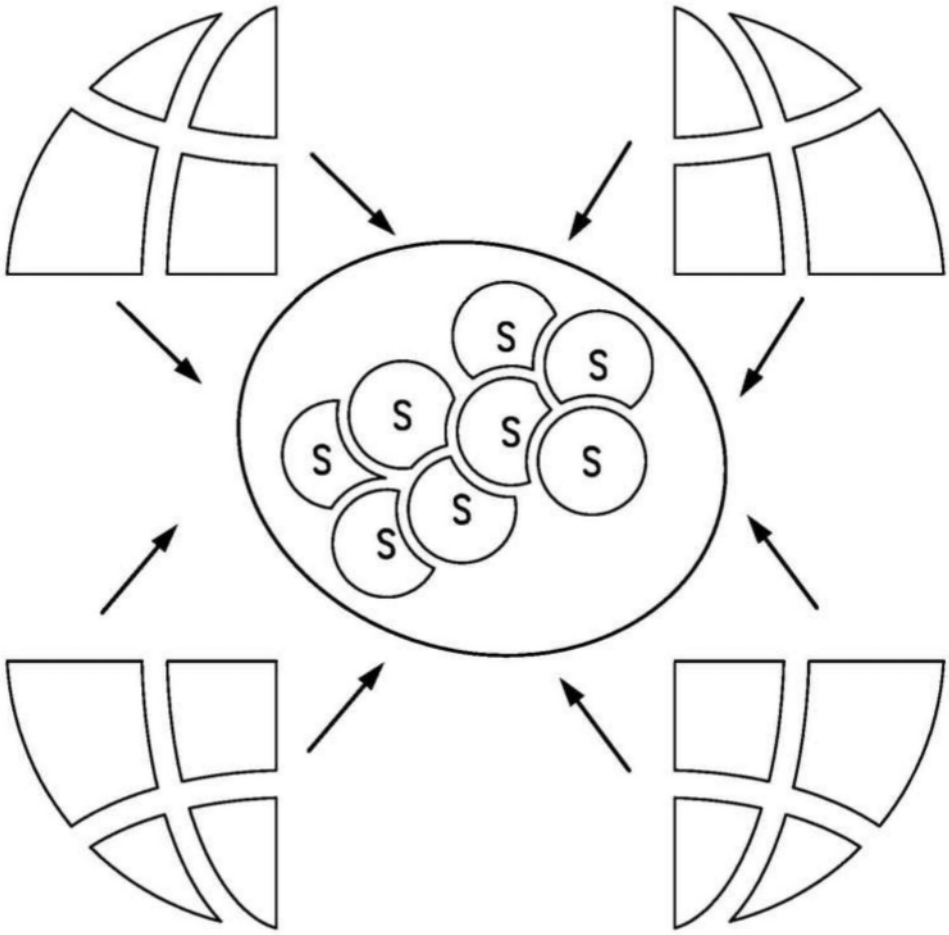


图1

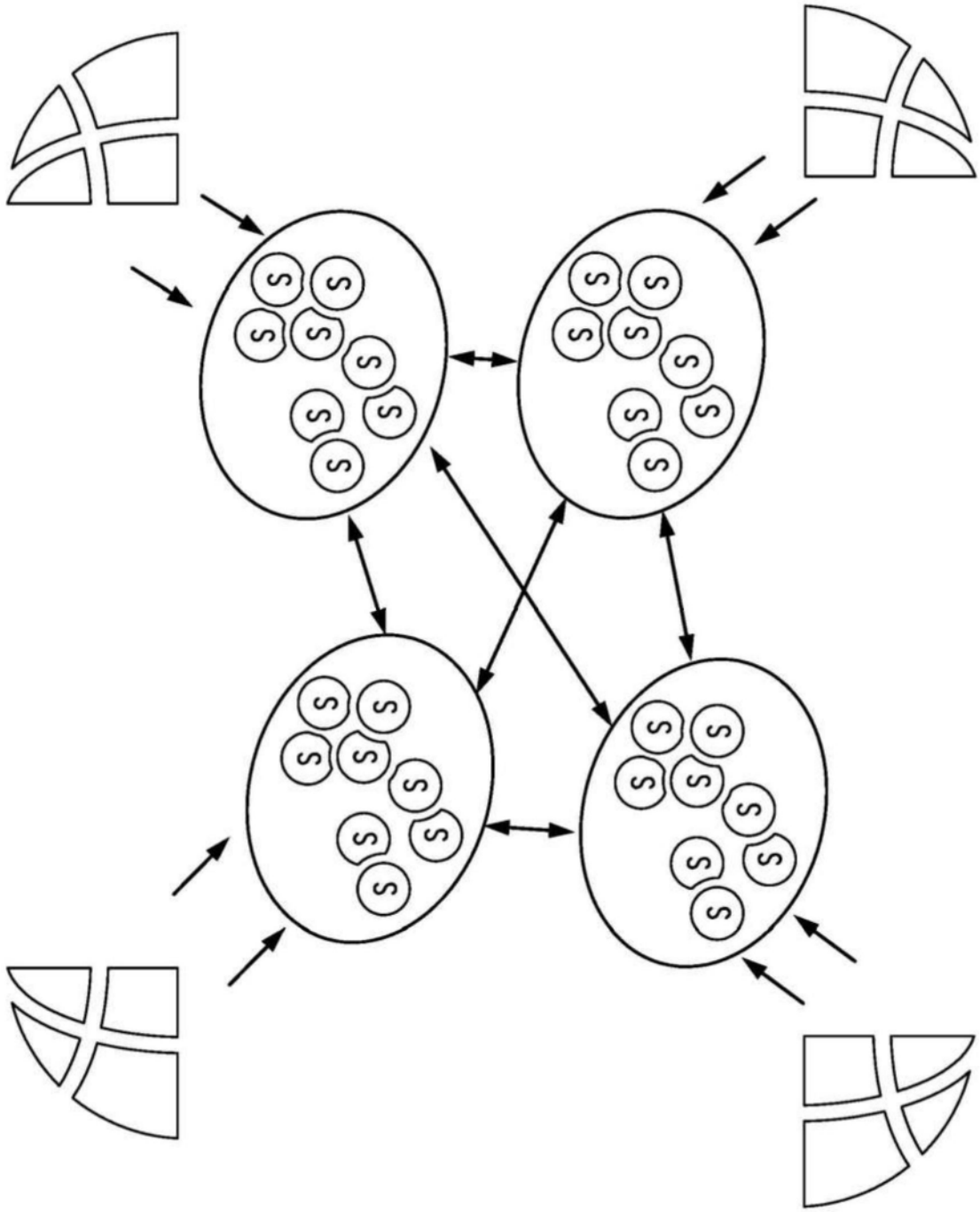


图2

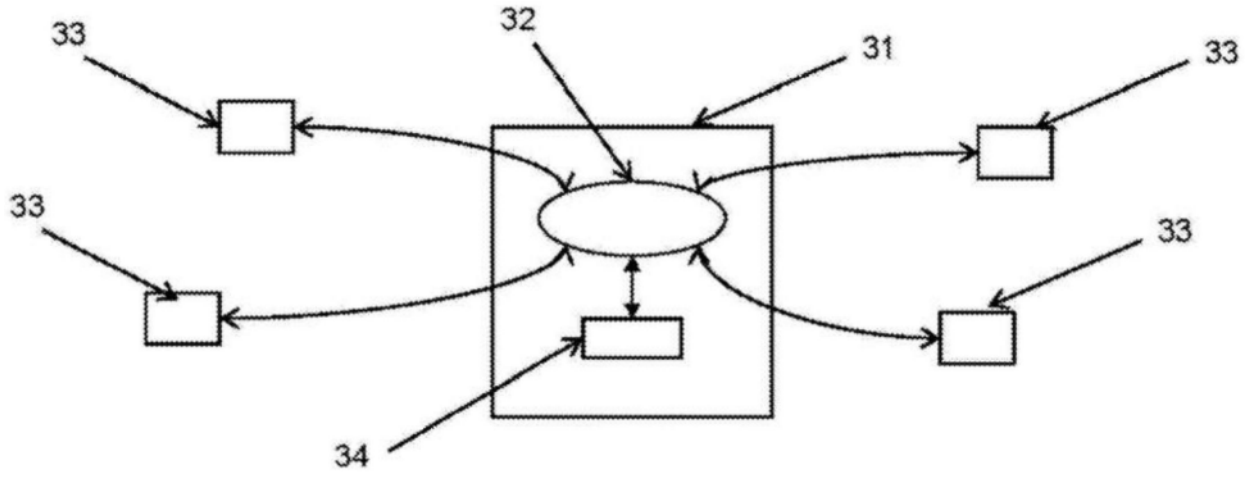


图3

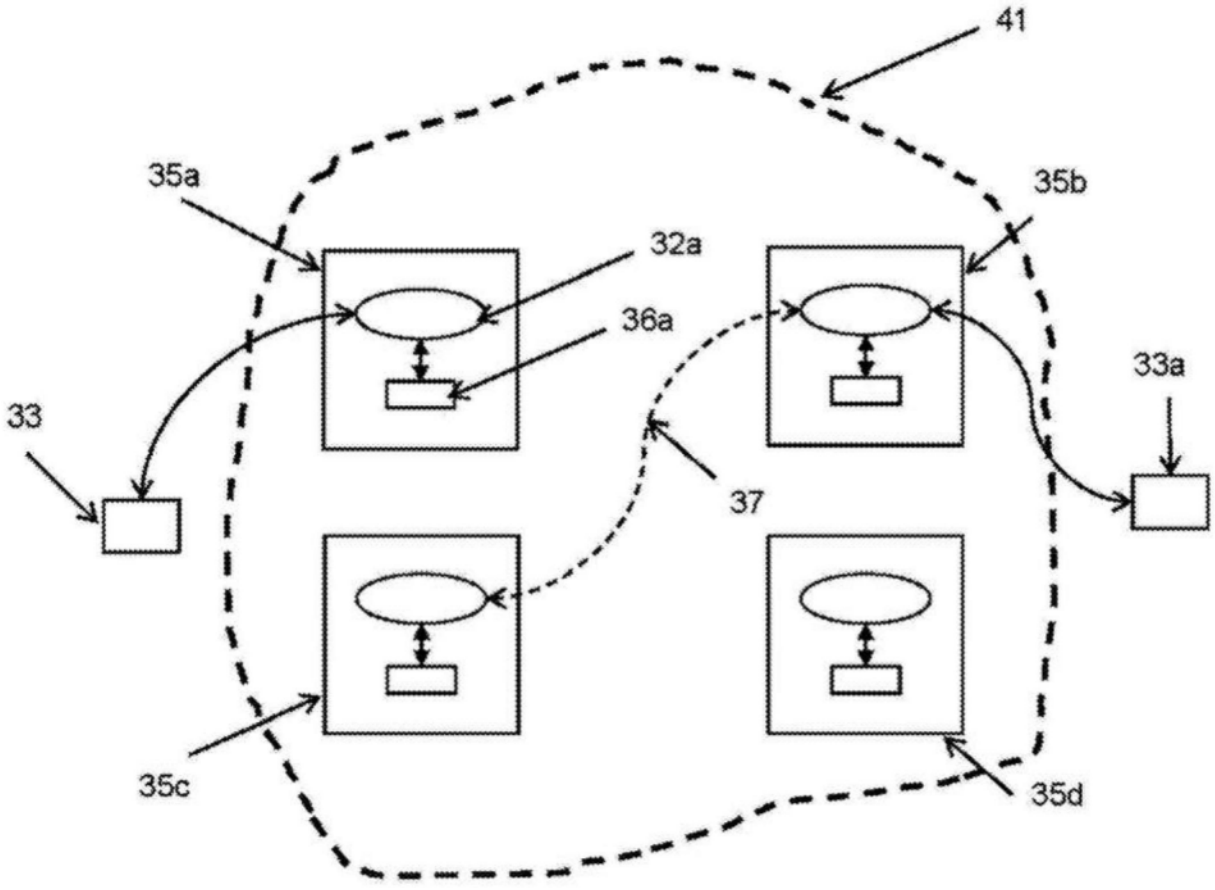


图4

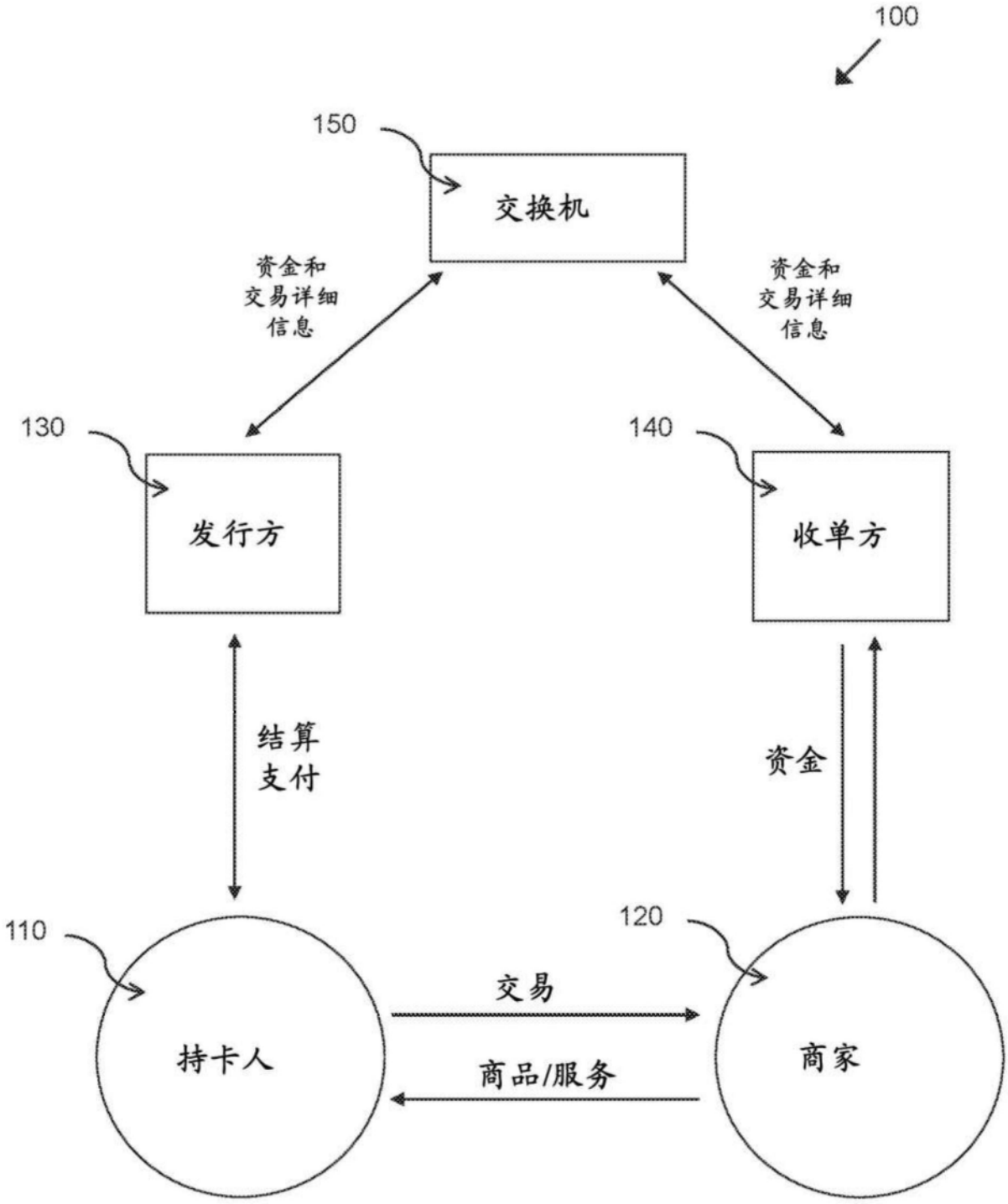

100
150
交换机
资金和
交易详细
信息
资金和
交易详细
信息
130
发行方
140
收单方
结算
支付
资金
110
持卡人
120
交易
商品/服务
商家

图5

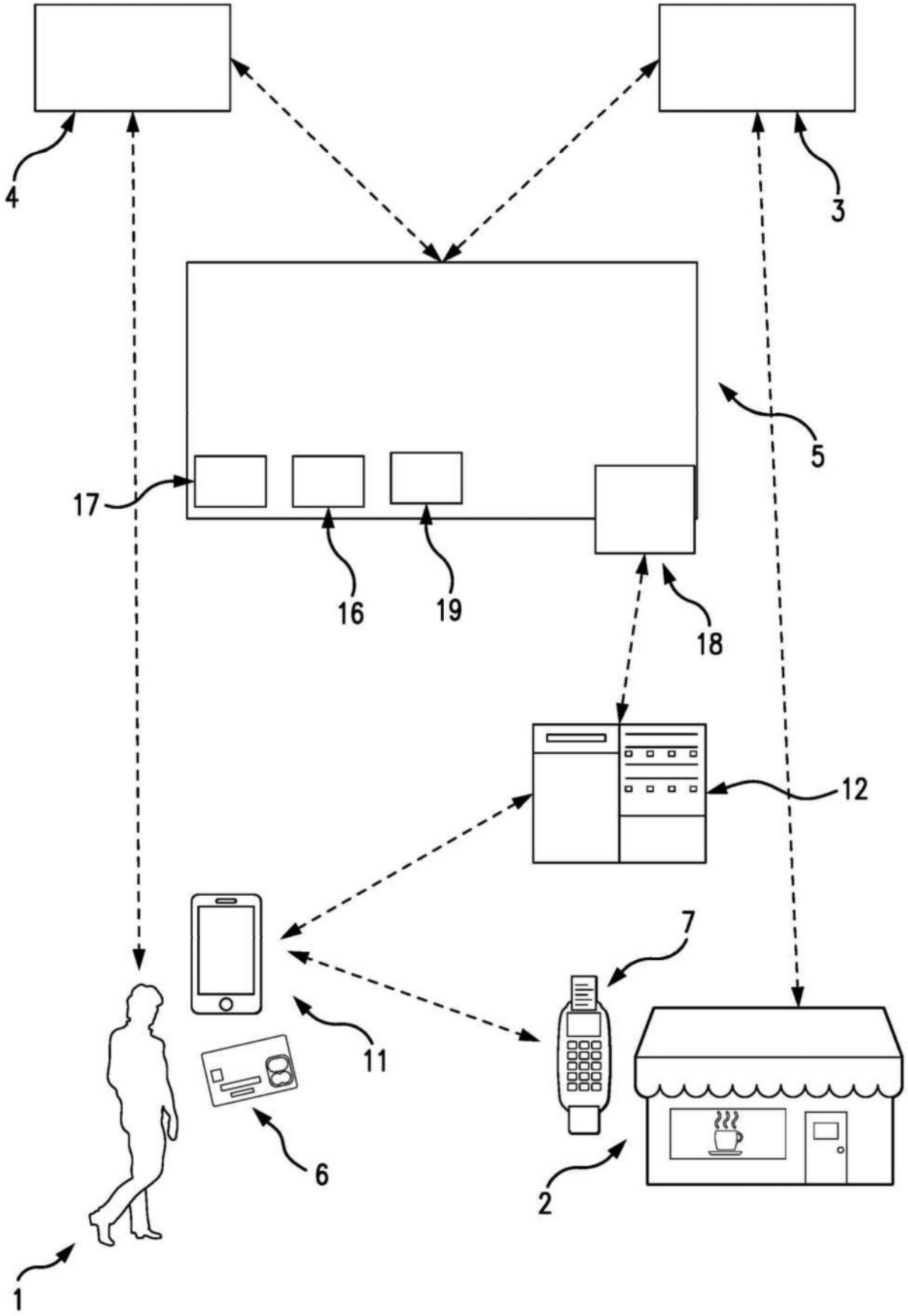


图6

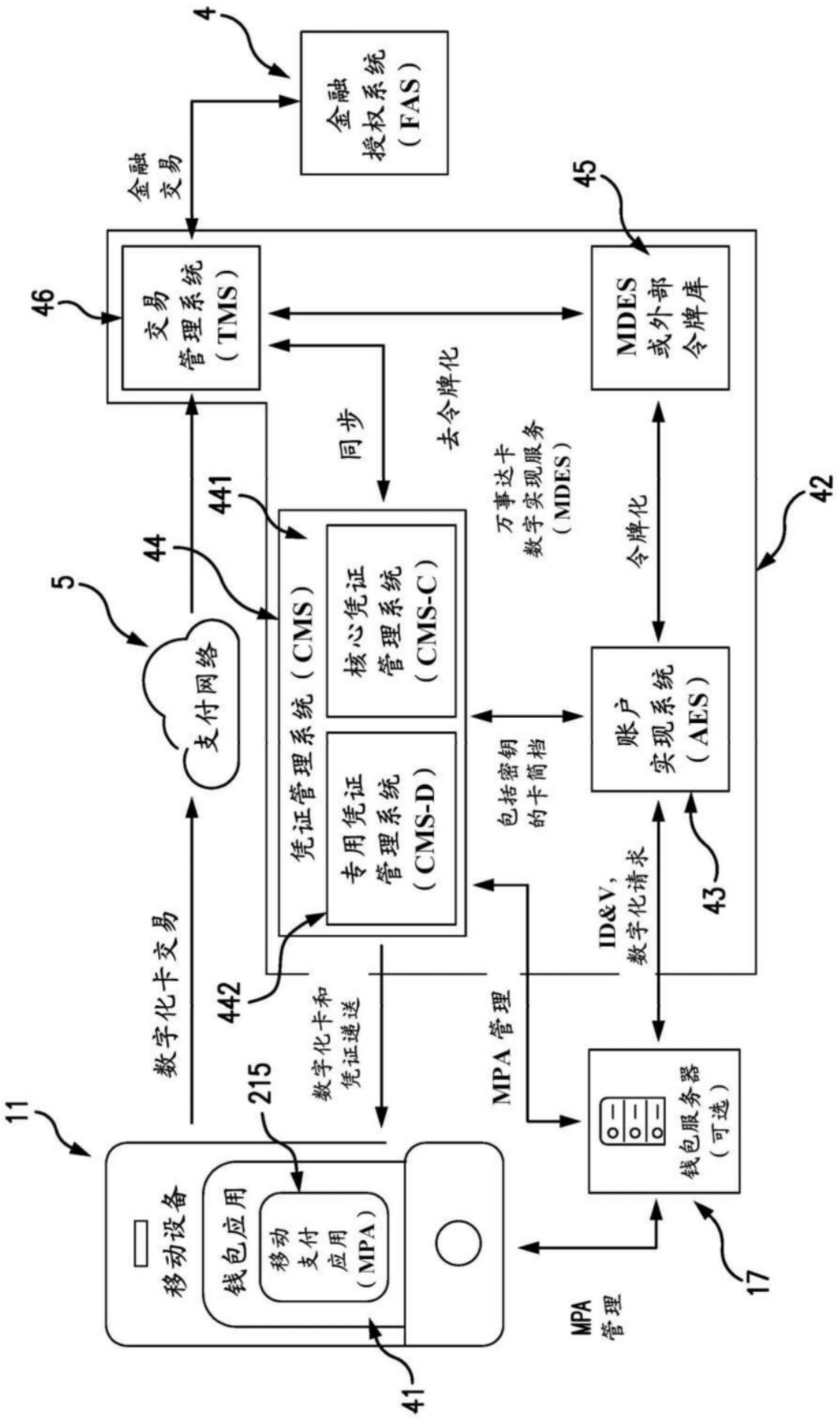

4
金融授权系统（FAS）
金融交易
45
46
交易管理系统（TMS）
MDES或外部令牌库
同步
去令牌化
万事达卡数字实现服务（MDES）
令牌化
42
441
44
5
核心凭证管理系统（CMS-C）
凭证管理系统（CMS）
支付网络
专用凭证管理系统（CMS-D）
账户实现系统（AES）
包括密钥的卡简档
ID&V，数字化请求
43
数字化卡交易
442
数字化卡和凭证递送
MPA 管理
215
11
移动设备
钱包应用
移动支付应用（MPA）
钱包服务器（可选）
MPA管理
17
41

图7

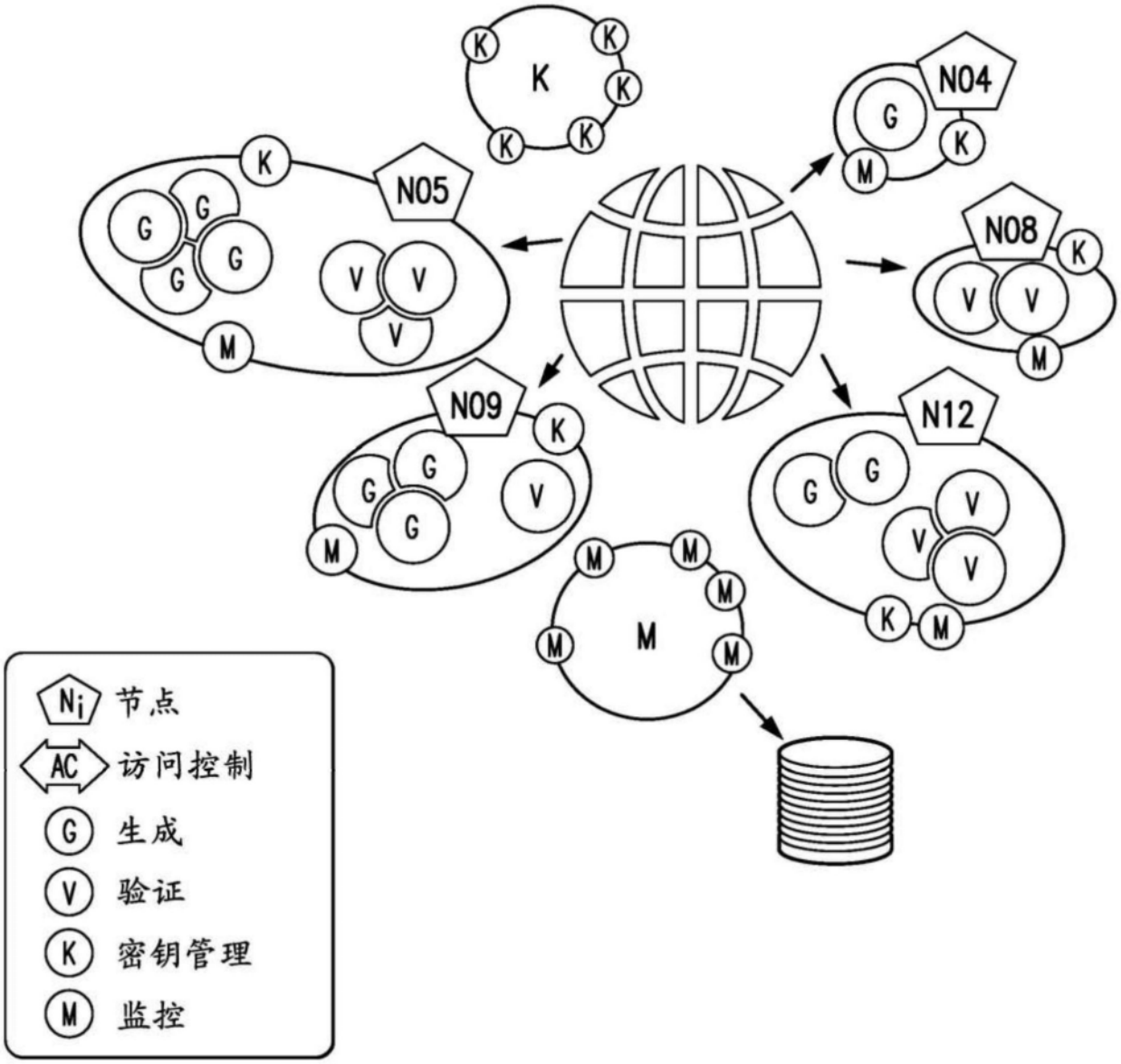


图8

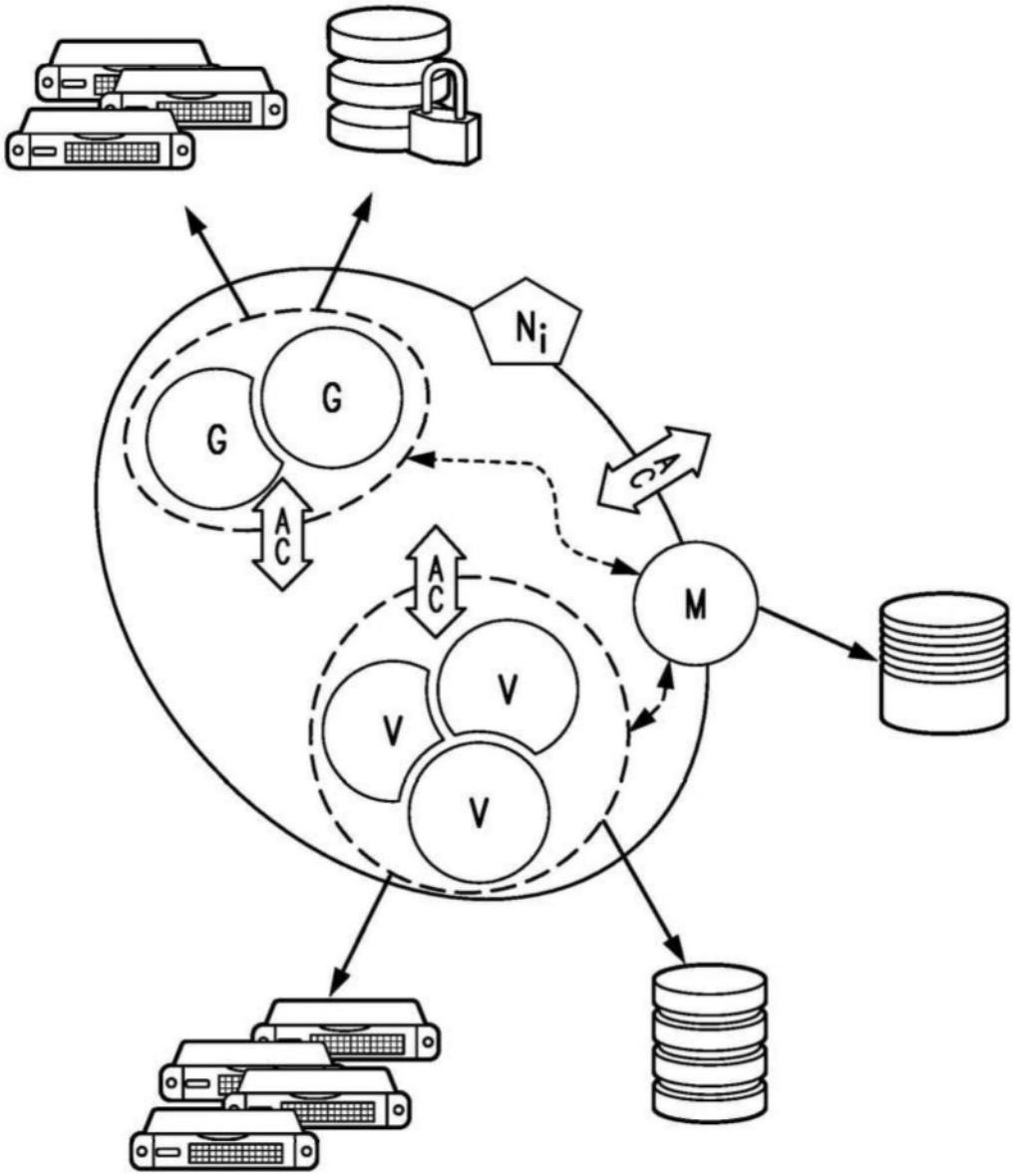


图9

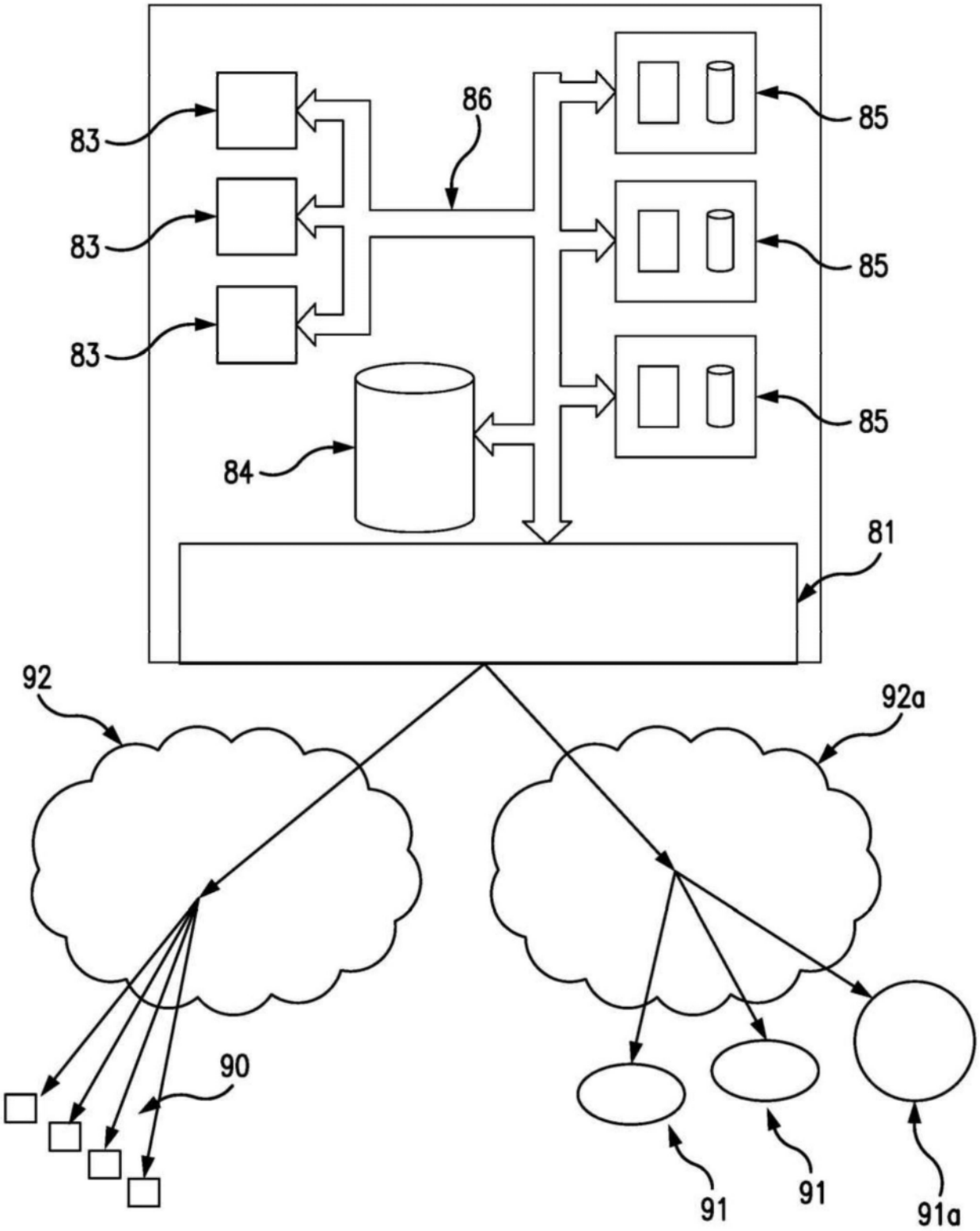


图10

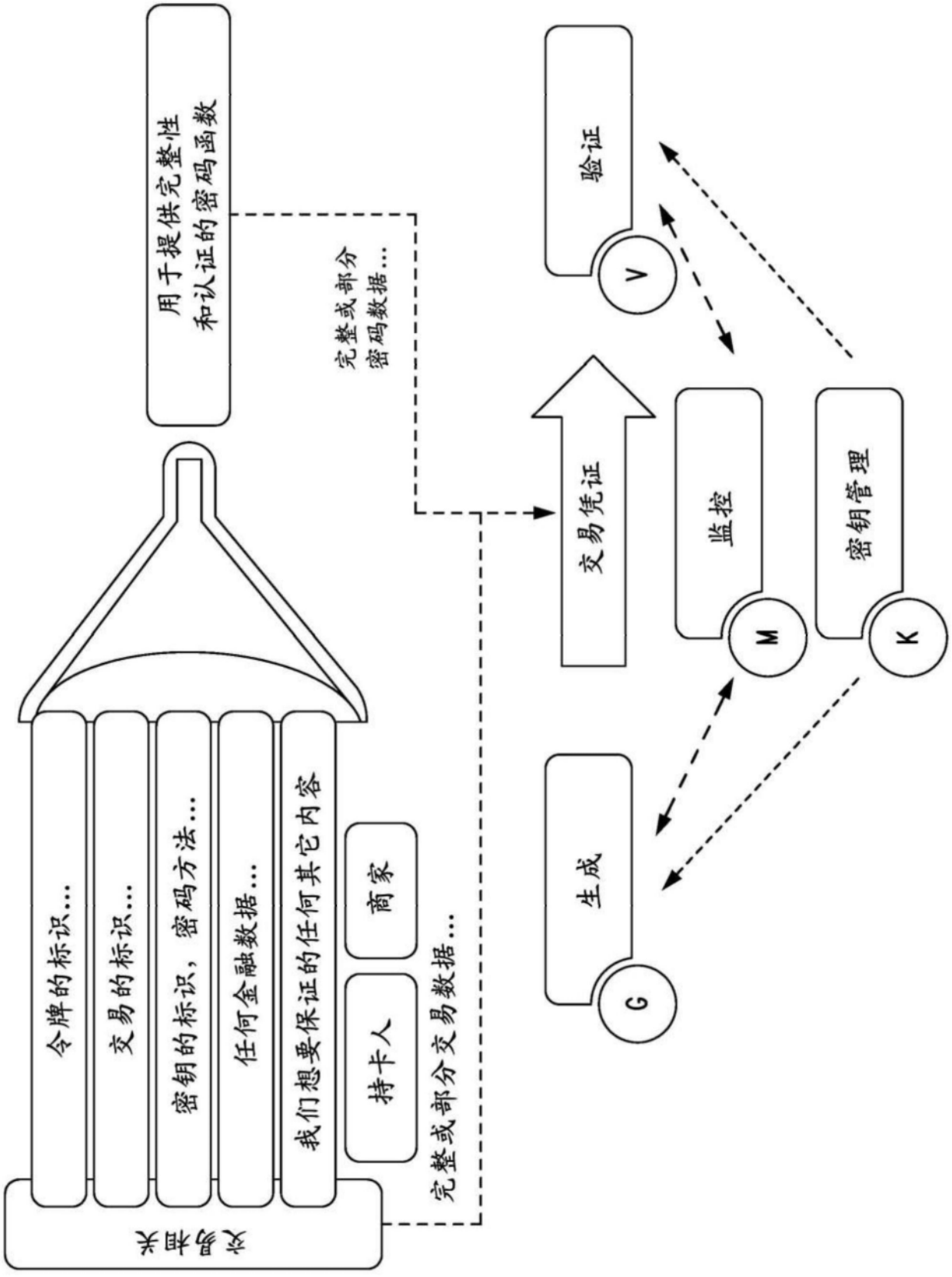

用于提供完整性
和认证的密码函数
完整或部分
密码数据...
验证
V
交易凭证
监控
M
密钥管理
K
生成
G
令牌的标识...
交易的标识...
密钥的标识，密码方法...
任何金融数据...
我们想要保证的任何其它内容
持卡人
商家
完整或部分交易数据...

图11

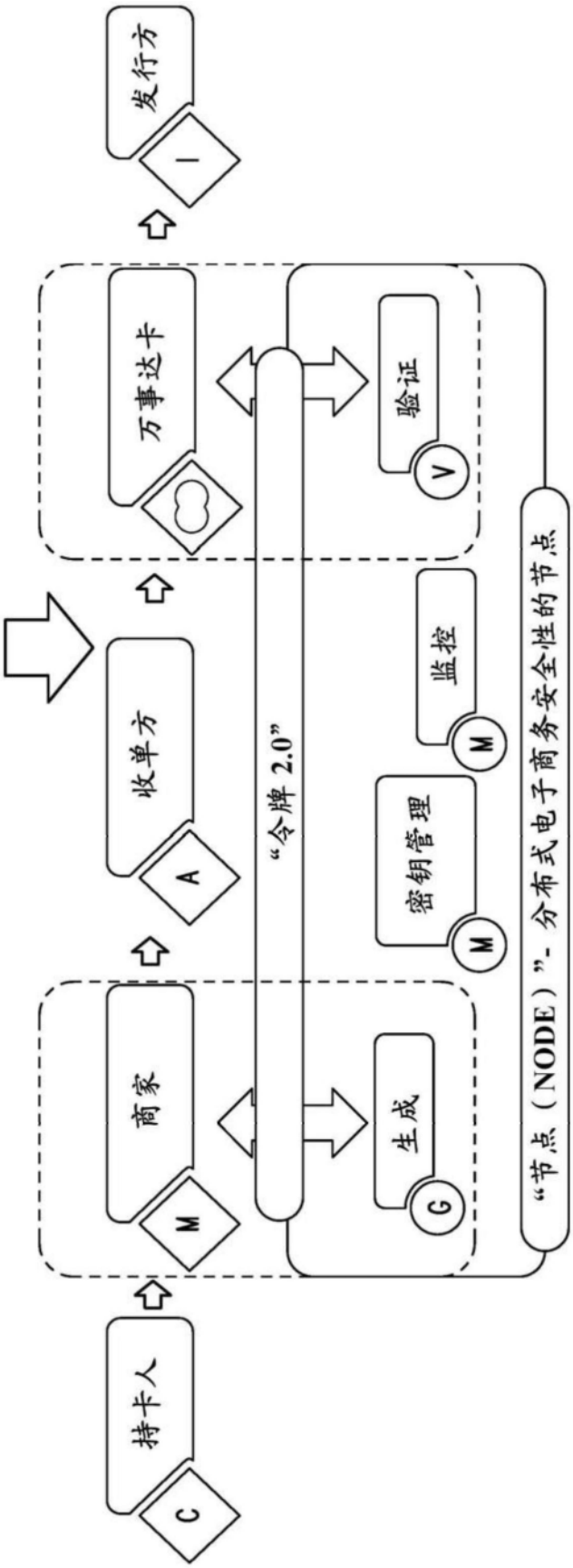

发行方
I
万事达卡
验证
V
收单方
A
“令牌 2.0”
监控
M
密钥管理
M
商家
M
生成
G
持卡人
C
“节点（NODE）”- 分布式电子商务安全性的节点

图12

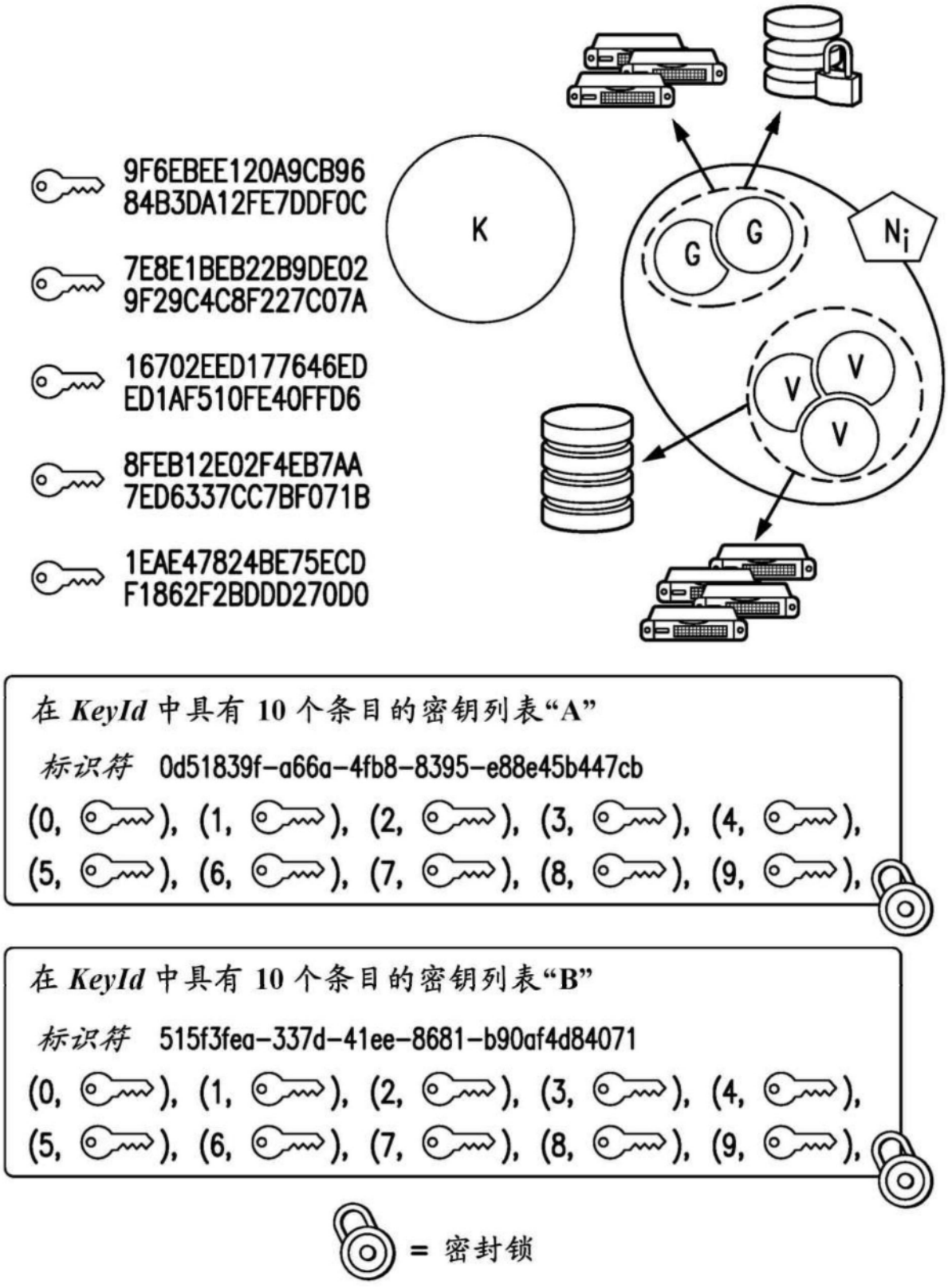

9F6EBEE120A9CB96
84B3DA12FE7DDF0C
7E8E1BEB22B9DE02
9F29C4C8F227C07A
16702EED177646ED
ED1AF510FE40FFD6
8FEB12E02F4EB7AA
7ED6337CC7BF071B
1EAE47824BE75ECD
F1862F2BDDD270D0
K
G
G
Ni
V
V
V
在 KeyId 中具有 10 个条目的密钥列表“A”
标识符 0d51839f-a66a-4fb8-8395-e88e45b447cb
(0,), (1,), (2,), (3,), (4,),
(5,), (6,), (7,), (8,), (9,),
在 KeyId 中具有 10 个条目的密钥列表“B”
标识符 515f3fea-337d-41ee-8681-b90af4d84071
(0,), (1,), (2,), (3,), (4,),
(5,), (6,), (7,), (8,), (9,),
= 密封锁

图13

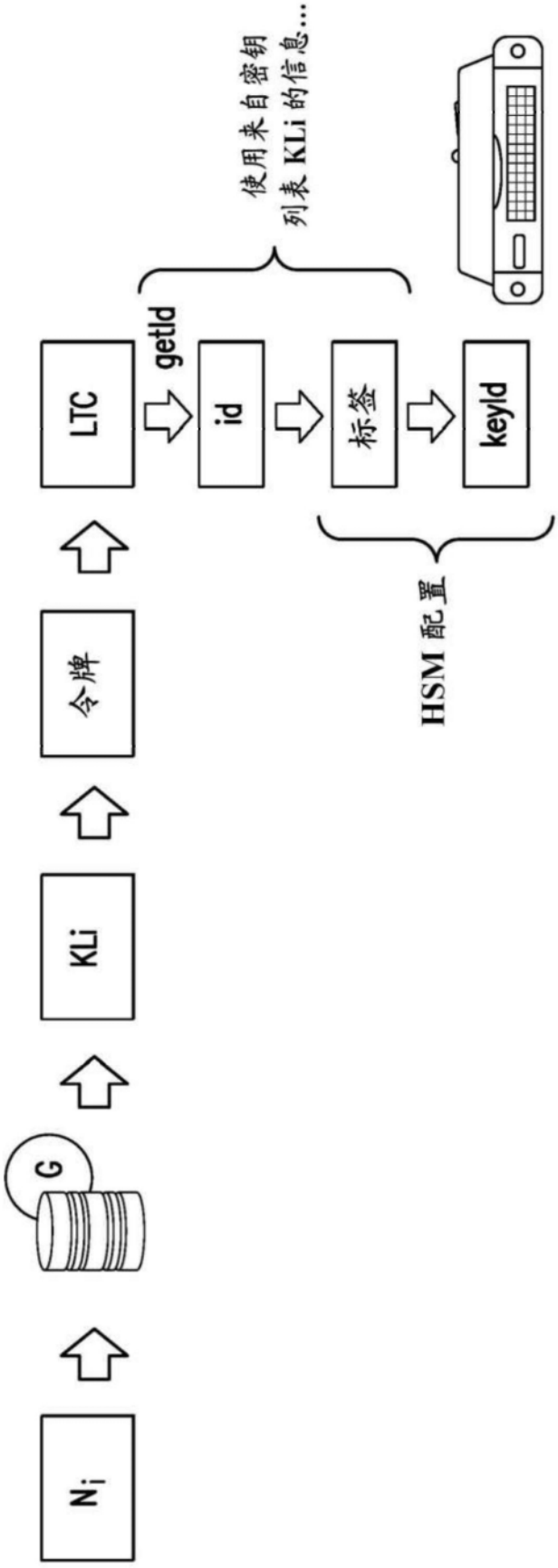


图14

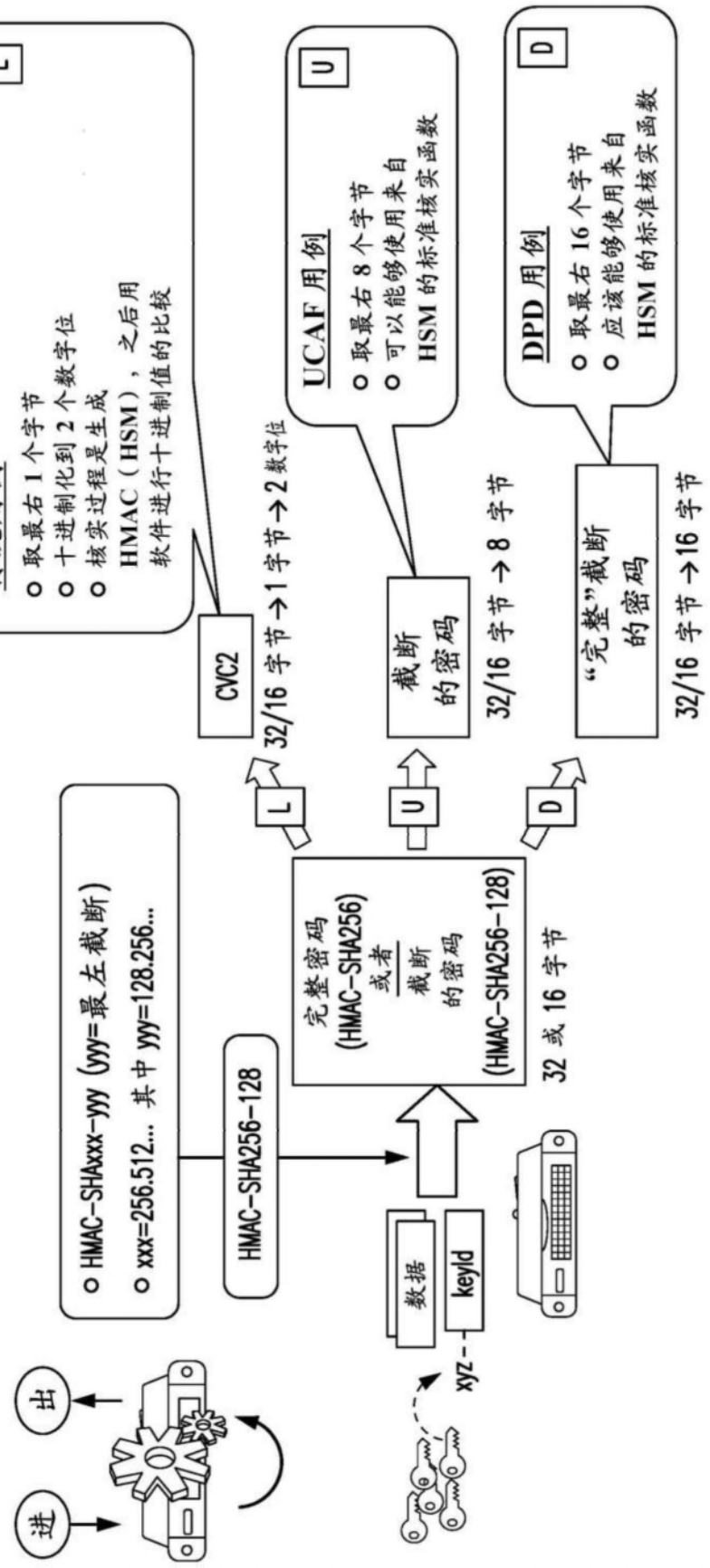

传统用例
o 取最右 1 个字节
o 十进制化到 2 个数字位
o 核实过程是生成
HMAC（HSM），之后用
软件进行十进制值的比较
L
CVC2
32/16 字节→1 字节→2 数字位
UCAF 用例
o 取最右 8 个字节
o 可以能够使用来自
HSM 的标准核实函数
U
截断
的密码
32/16 字节→8 字节
DPD 用例
o 取最右 16 个字节
o 应该能够使用来自
HSM 的标准核实函数
D
“完整”截断
的密码
32/16 字节→16 字节
o HMAC-SHAxxx-yyy（yyy=最左截断）
o xxx=256.512... 其中 yyy=128.256...
HMAC-SHA256-128
完整密码
(HMAC-SHA256)
或者
截断
的密码
(HMAC-SHA256-128)
32 或 16 字节
数据
keyId
xyz-
进
出

图15

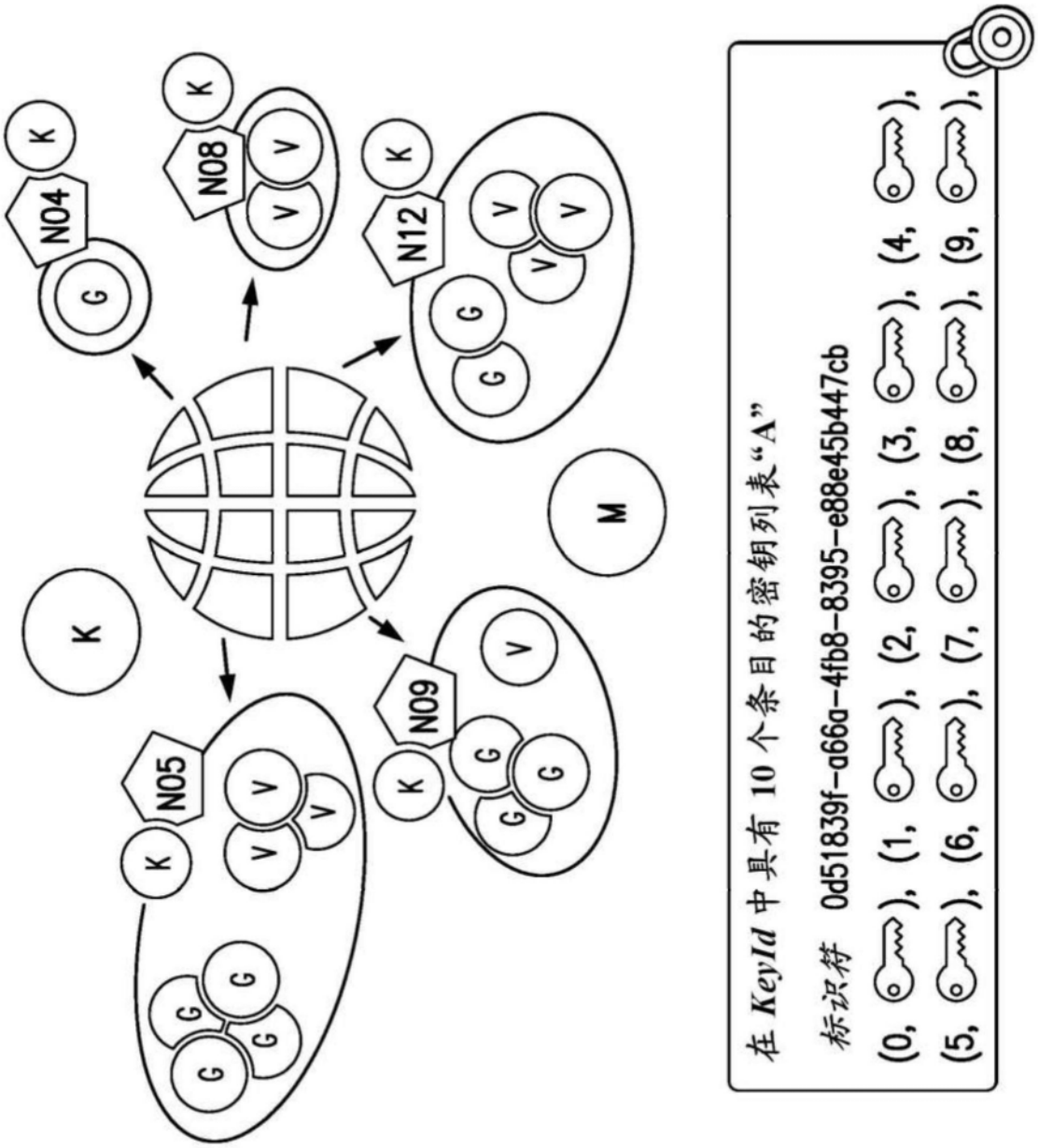

K
N04
G
K
N08
V
V
V
K
N12
G
G
V
V
V
M
K
N05
K
V
V
V
G
G
G
G
K
N09
G
G
G
V
在 KeyId 中具有 10 个条目的密钥列表“A”
标识符 0d51839f-a66a-4fb8-8395-e88e45b447cb
(0,), (1,), (2,), (3,), (4,),
(5,), (6,), (7,), (8,), (9,),

图16

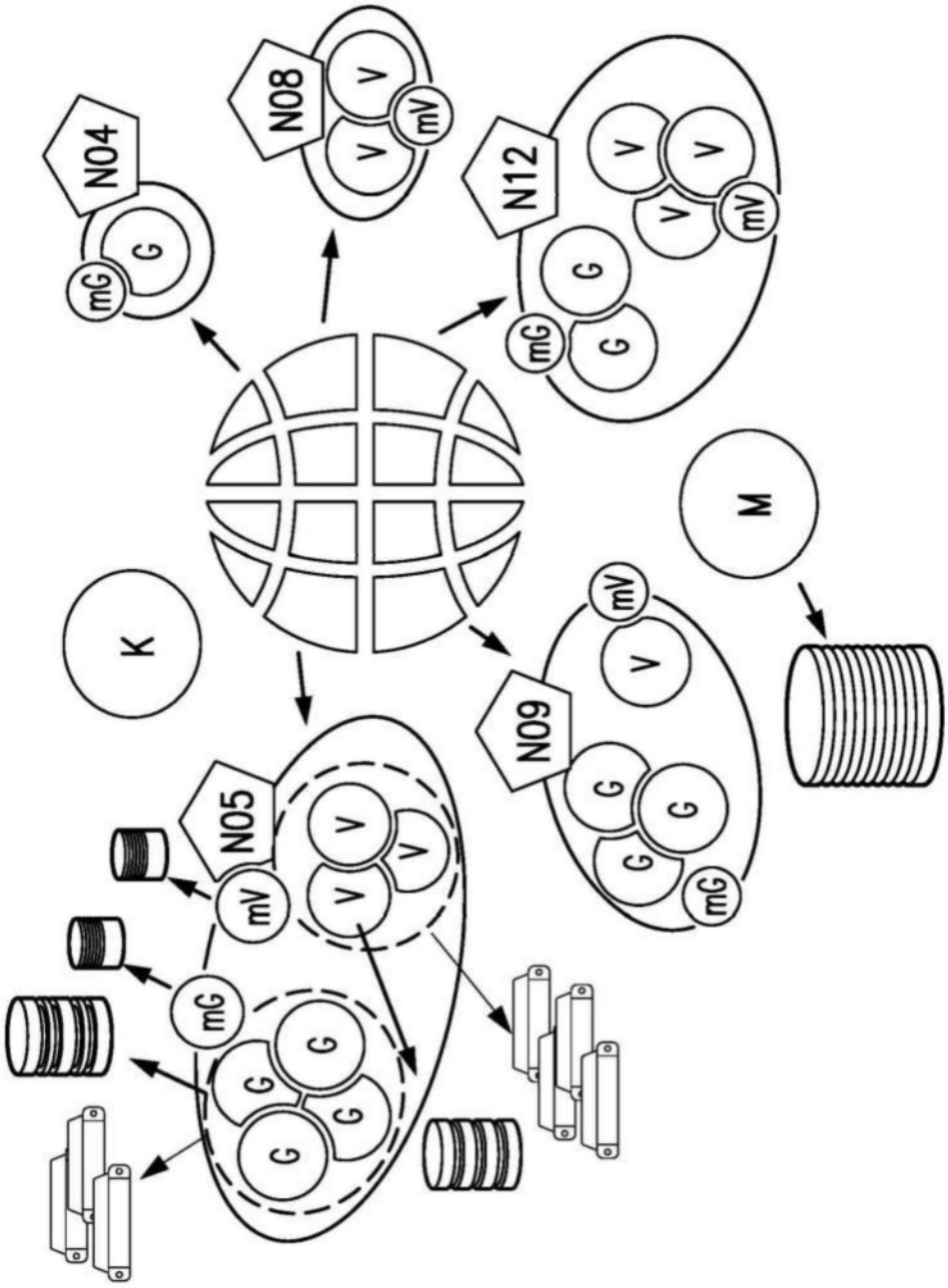

N04
mG
G
N08
V
V
mV
N12
V
V
V
mV
mG
G
M
K
N09
mV
V
G
G
G
mG
N05
mV
V
V
mG
G
G
G
G

图17

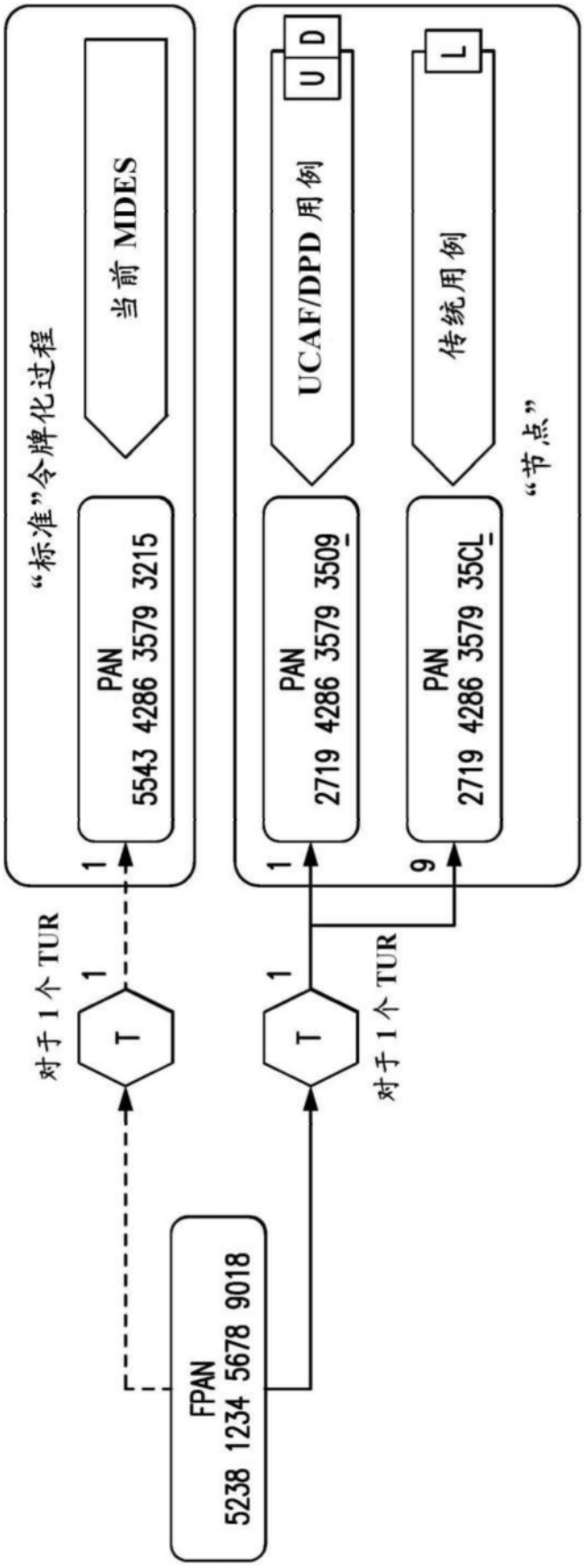

“标准”令牌化过程
当前 MDES
PAN
5543 4286 3579 3215
1
对于 1 个 TUR
1
T
FPAN
5238 1234 5678 9018
“节点”
UCAF/DPD 用例
U D
PAN
2719 4286 3579 3509_
1
传统用例
L
PAN
2719 4286 3579 35CL_
9
对于 1 个 TUR
1
T

图18

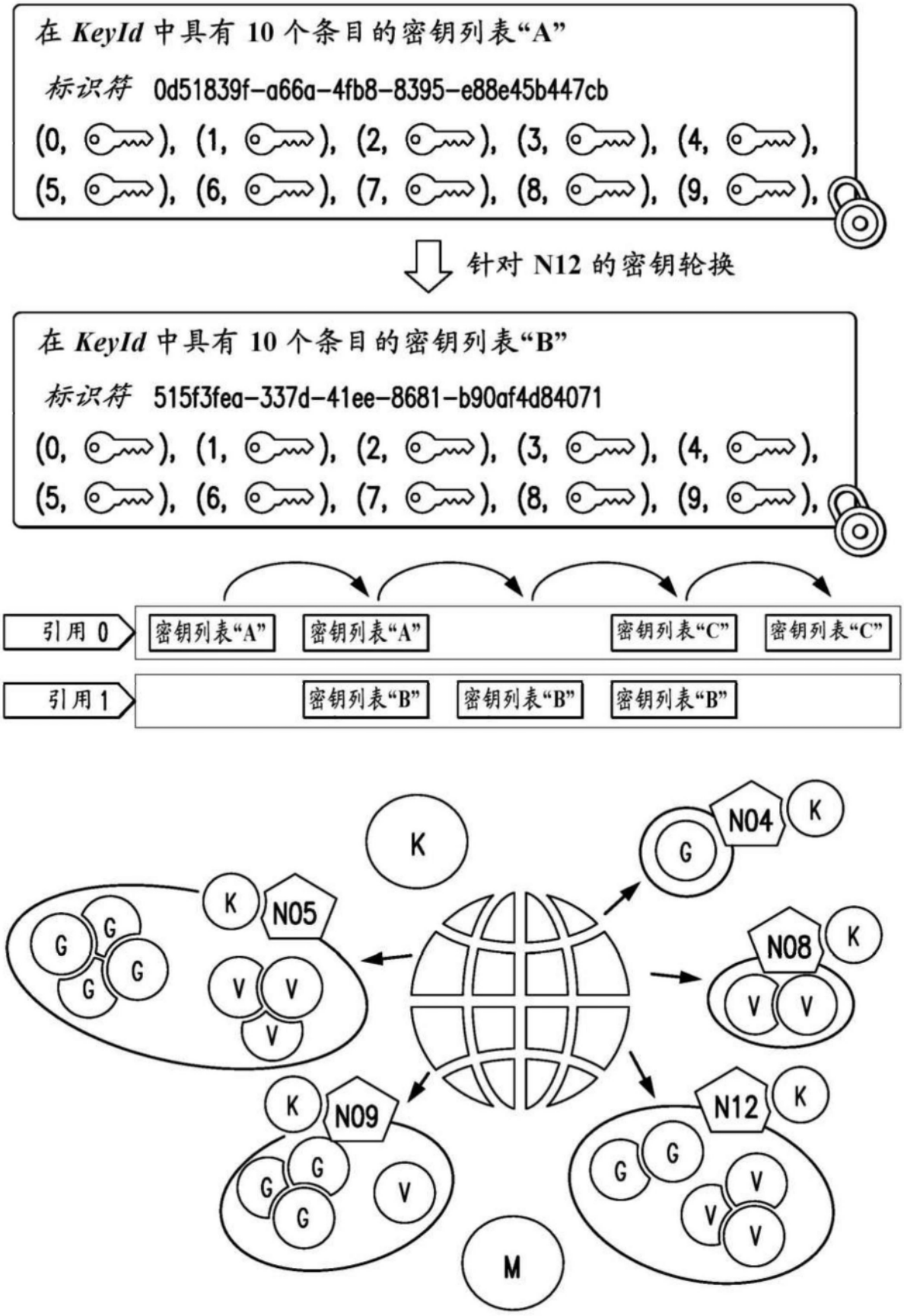

在 KeyId 中具有 10 个条目的密钥列表“A”
标识符 0d51839f-a66a-4fb8-8395-e88e45b447cb
(0,), (1,), (2,), (3,), (4,),
(5,), (6,), (7,), (8,), (9,),
针对 N12 的密钥轮换
在 KeyId 中具有 10 个条目的密钥列表“B”
标识符 515f3fea-337d-41ee-8681-b90af4d84071
(0,), (1,), (2,), (3,), (4,),
(5,), (6,), (7,), (8,), (9,),
引用 0
密钥列表“A”
密钥列表“A”
密钥列表“C”
密钥列表“C”
引用 1
密钥列表“B”
密钥列表“B”
密钥列表“B”
K
N04
K
G
N05
K
G
G
G
G
V
V
V
N08
K
V
V
K
N09
G
G
G
V
N12
K
G
G
V
V
V
M

图19

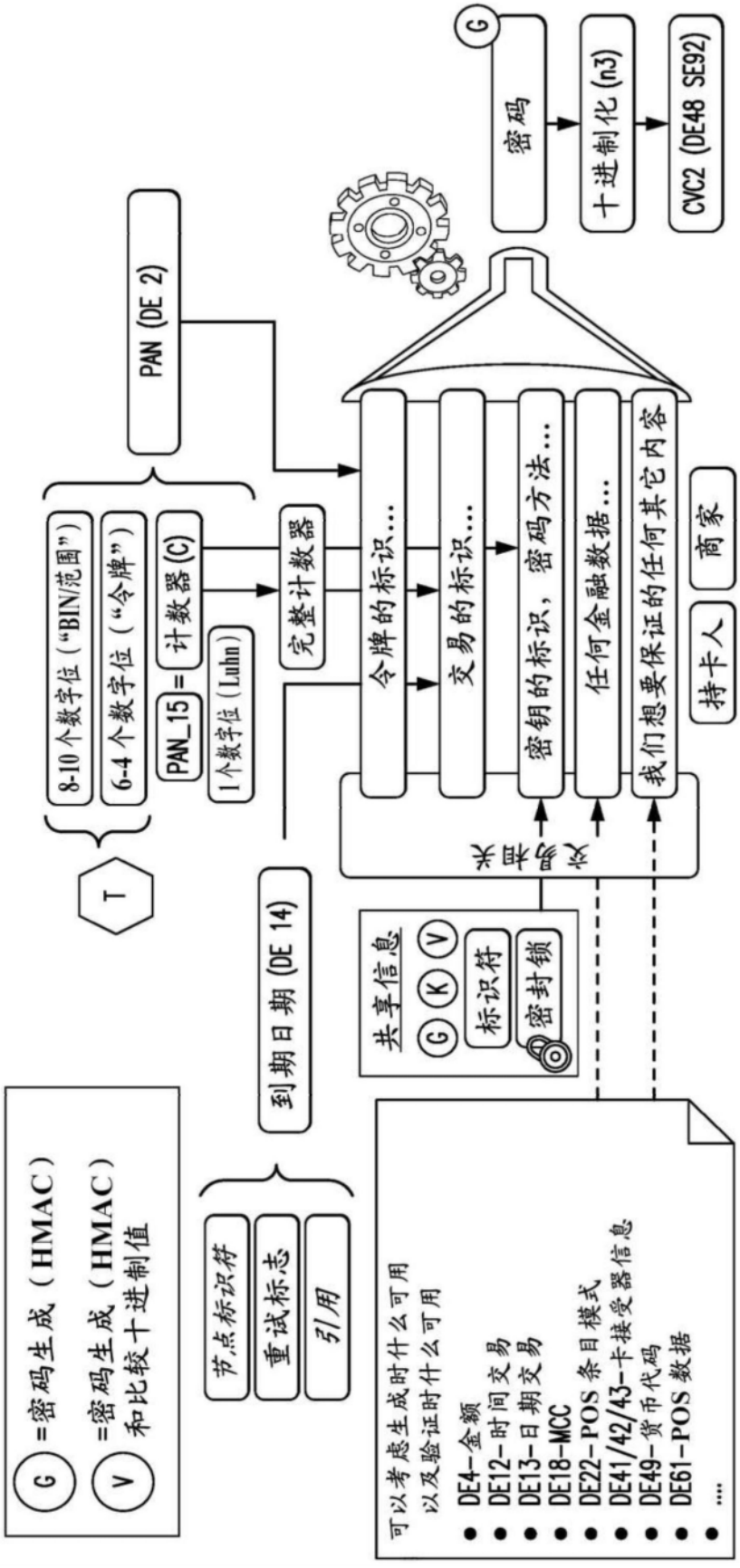

G =密码生成（HMAC）
V =密码生成（HMAC）和比较十进制值
节点标识符
重试标志
引用
到期日期 (DE 14)
共享信息
G K V
标识符
密封锁
可以考虑生成时什么可用
以及验证时什么可用
DE4-金额
DE12-时间交易
DE13-日期交易
DE18-MCC
DE22-POS 条目模式
DE41/42/43-卡接受器信息
DE49-货币代码
DE61-POS 数据
....
T
8-10 个数字位（"BIN/范围"）
6-4 个数字位（"令牌"）
PAN_15
计数器 (C)
1 个数字位 (Luhn)
PAN (DE 2)
完整计数器
令牌的标识...
交易的标识...
密钥的标识，密码方法...
任何金融数据...
我们想要保证的任何其它内容
持卡人
商家
密码
十进制化 (n3)
CVC2 (DE48 SE92)

图20

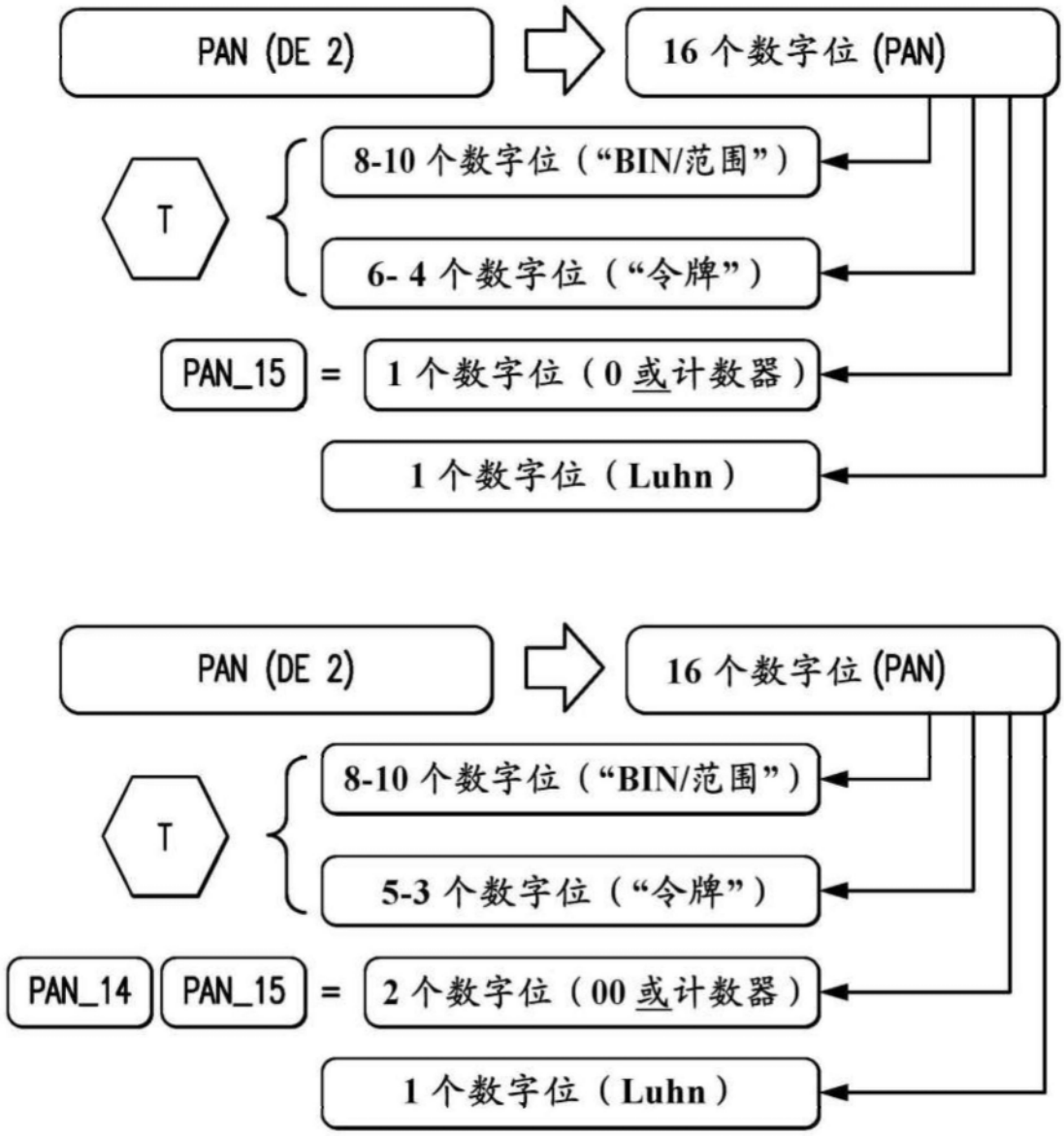

PAN (DE 2)
16 个数字位 (PAN)
T
8-10 个数字位（“BIN/范围”）
6- 4 个数字位（“令牌”）
PAN_15 =
1 个数字位（0 或计数器）
1 个数字位（Luhn）
PAN (DE 2)
16 个数字位 (PAN)
T
8-10 个数字位（“BIN/范围”）
5-3 个数字位（“令牌”）
PAN_14
PAN_15 =
2 个数字位（00 或计数器）
1 个数字位（Luhn）

图21

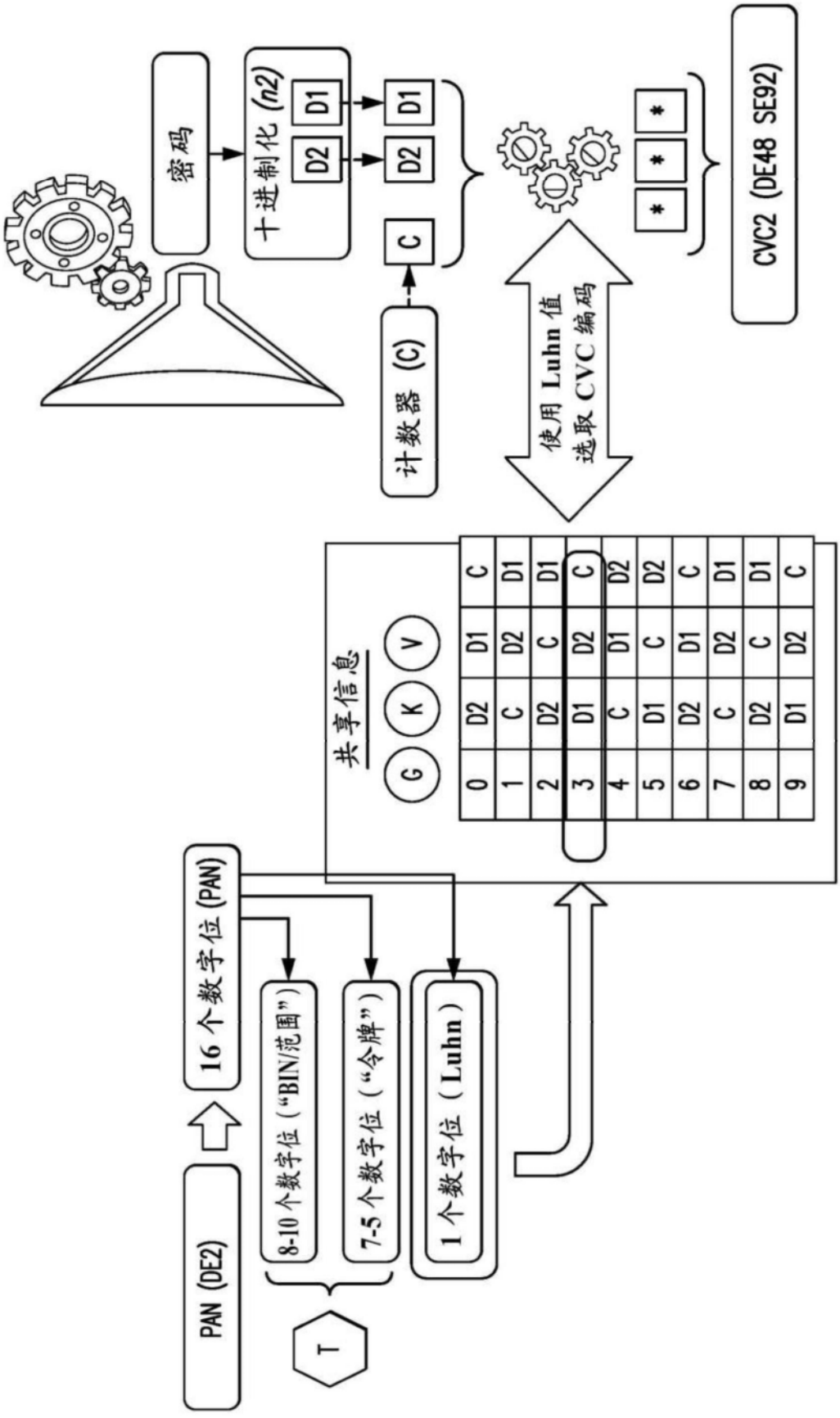

密码
十进制化 (n2)
D2
D1
C
计数器 (C)
使用 Luhn 值
选取 CVC 编码
CVC2 (DE48 SE92)
共享信息
G
K
V
0 D2 D1 C
1 C D2 D1
2 D2 C D1
3 D1 D2 C
4 C D1 D2
5 D1 C D2
6 D2 D1 C
7 C D2 D1
8 D2 C D1
9 D1 D2 C
16 个数字位 (PAN)
PAN (DE2)
T
8-10 个数字位 ("BIN/范围")
7-5 个数字位 ("令牌")
1 个数字位 (Luhn)

图22

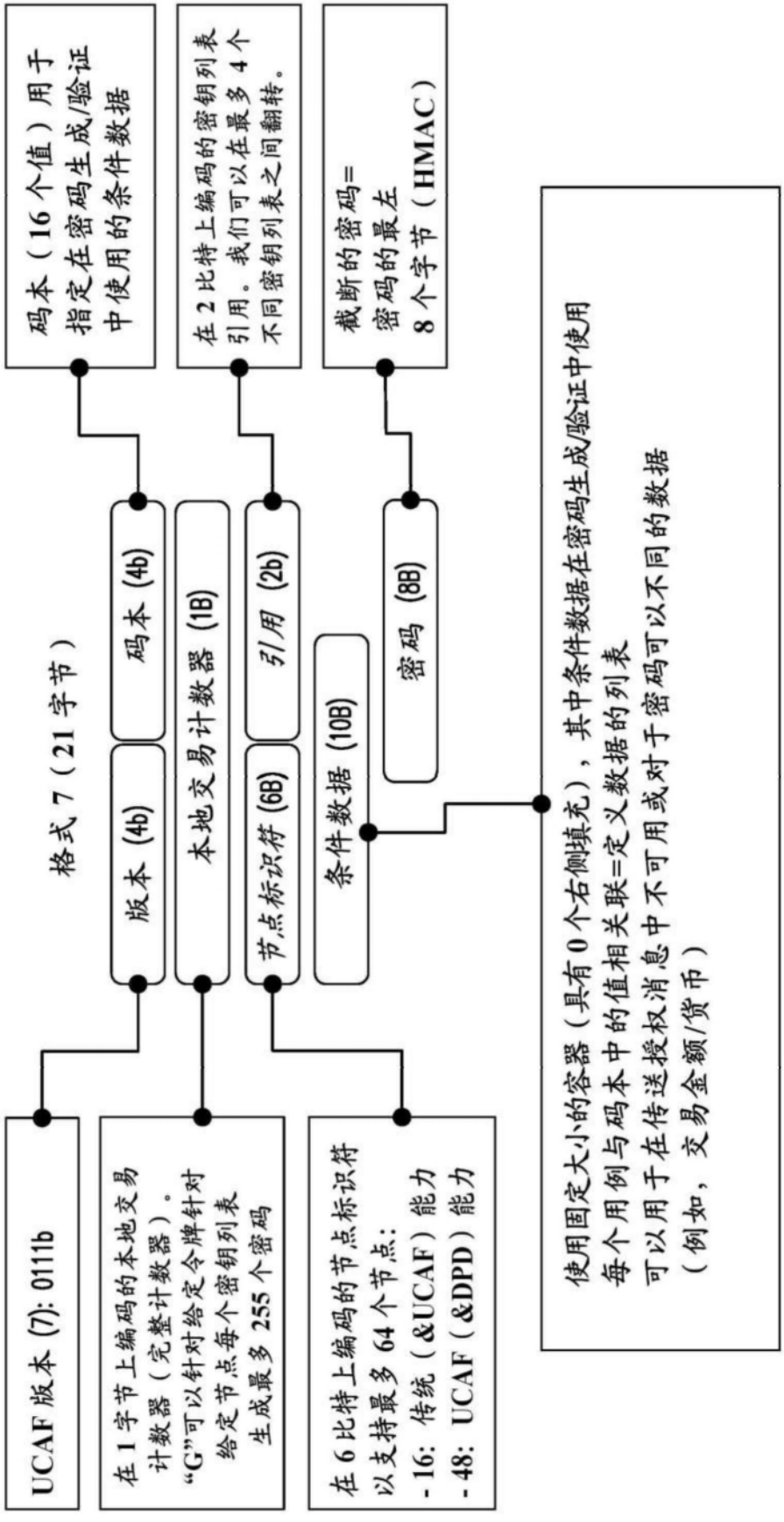

格式 7（21 字节）
UCAF 版本 (7): 0111b
在 1 字节上编码的本地交易计数器（完整计数器）。
"G"可以针对给定令牌针对给定节点每个密钥列表生成最多 255 个密码
在 6 比特上编码的节点标识符以支持最多 64 个节点：
- 16：传统（&UCAF）能力
- 48：UCAF（&DPD）能力
版本 (4b)
本地交易计数器 (1B)
节点标识符 (6B)
条件数据 (10B)
码本 (4b)
引用 (2b)
密码 (8B)
码本（16 个值）用于指定在密码生成/验证中使用的条件数据
在 2 比特上编码的密钥列表引用。我们可以在最多 4 个不同密钥列表之间翻转。
截断的密码=密码的最左 8 个字节（HMAC）
使用固定大小的容器（具有 0 个右侧填充），其中条件数据在密码生成/验证中使用
每个用例与码本中的值相关联=定义数据的列表
可以用于在传送授权消息中不可用或对于密码可以不同的数据
（例如，交易金额/货币）

图23

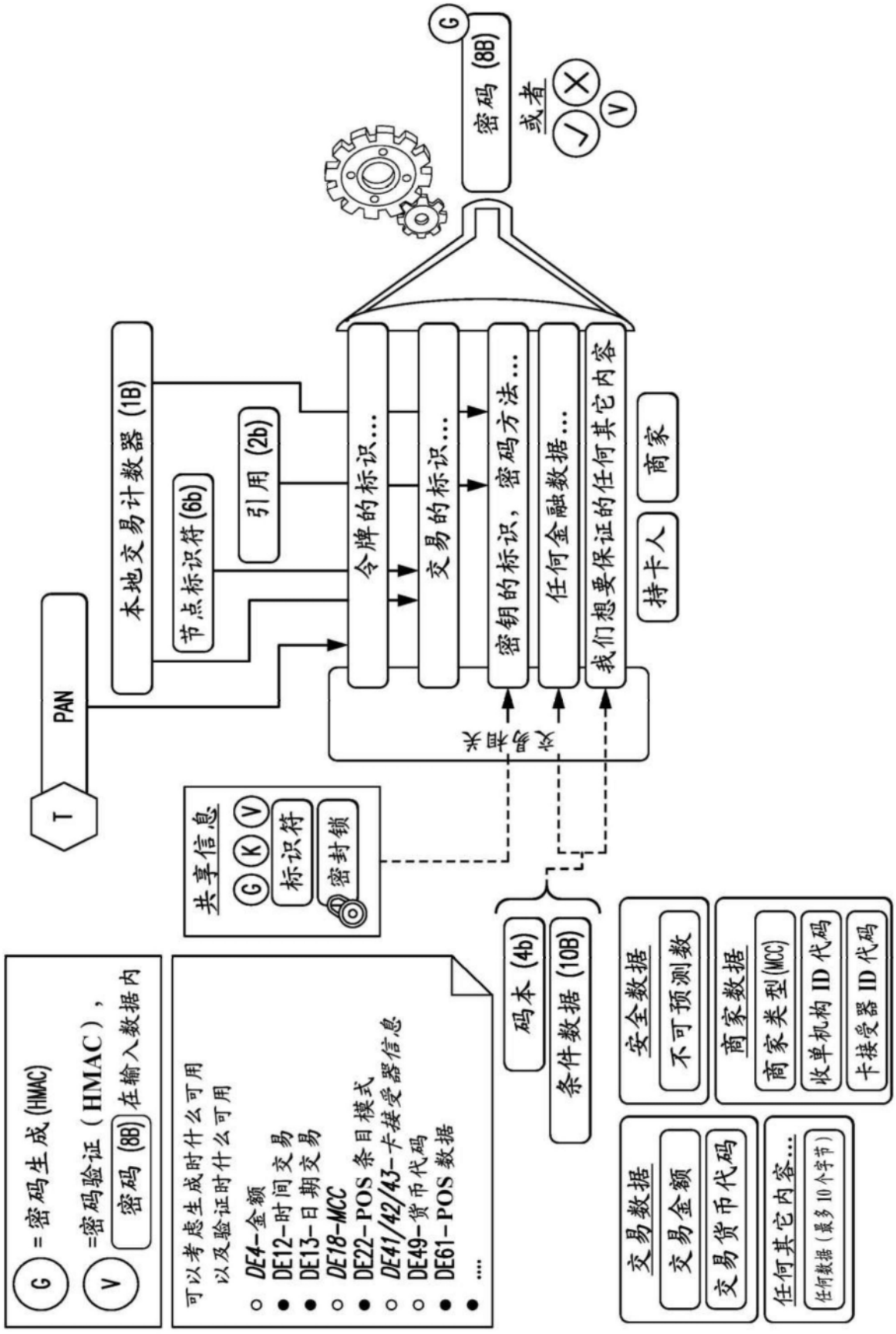

G =密码生成 (HMAC)
V =密码验证（HMAC），
密码 (8B) 在输入数据内
可以考虑生成时什么可用
以及验证时什么可用
DE4-金额
DE12-时间交易
DE13-日期交易
DE18-MCC
DE22-POS 条目模式
DE41/42/43-卡接受器信息
DE49-货币代码
DE61-POS 数据
....
T
PAN
本地交易计数器 (1B)
节点标识符(6b)
引用 (2b)
令牌的标识...
交易的标识...
密钥的标识，密码方法...
任何金融数据...
我们想要保证的任何其它内容
持卡人
商家
共享信息
标识符
密封锁
码本 (4b)
条件数据 (10B)
安全数据
不可预测数
商家数据
商家类型(MCC)
收单机构 ID 代码
卡接受器 ID 代码
交易数据
交易金额
交易货币代码
任何其它内容...
任何数据（最多 10 个字节）
密码 (8B)
或者

图24

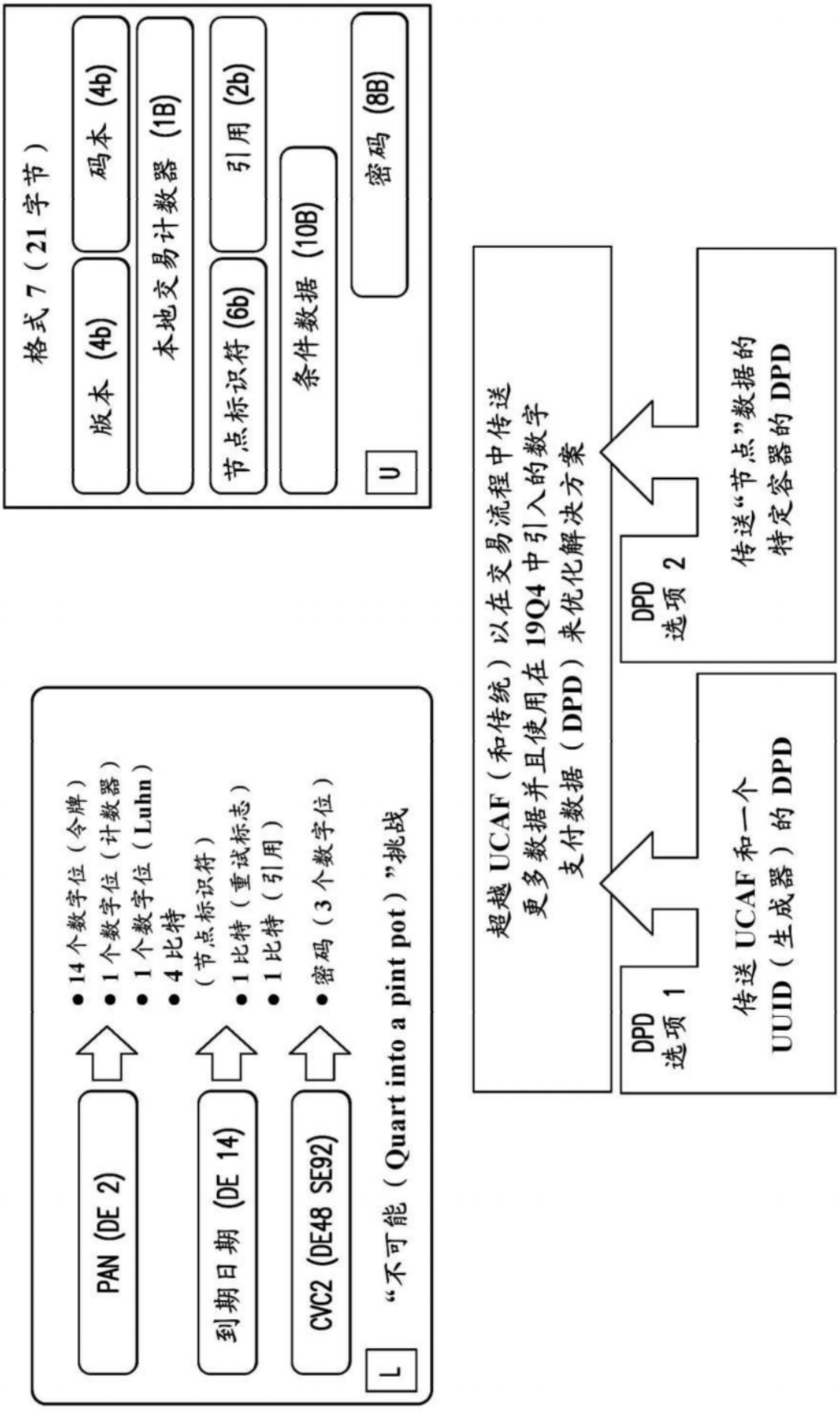

格式 7（21 字节）
版本 (4b)
码本 (4b)
本地交易计数器 (1B)
节点标识符 (6b)
引用 (2b)
条件数据 (10B)
U
密码 (8B)
PAN (DE 2)
到期日期 (DE 14)
CVC2 (DE48 SE92)
L
• 14 个数字位（令牌）
• 1 个数字位（计数器）
• 1 个数字位（Luhn）
• 4 比特
（节点标识符）
• 1 比特（重试标志）
• 1 比特（引用）
• 密码（3 个数字位）
“不可能（Quart into a pint pot）”挑战
超越 UCAF（和传统）以在交易流程中传送
更多数据并且使用在 19Q4 中引入的数字
支付数据（DPD）来优化解决方案
DPD
选项 1
传送 UCAF 和一个
UUID（生成器）的 DPD
DPD
选项 2
传送“节点”数据的
特定容器的 DPD

图25

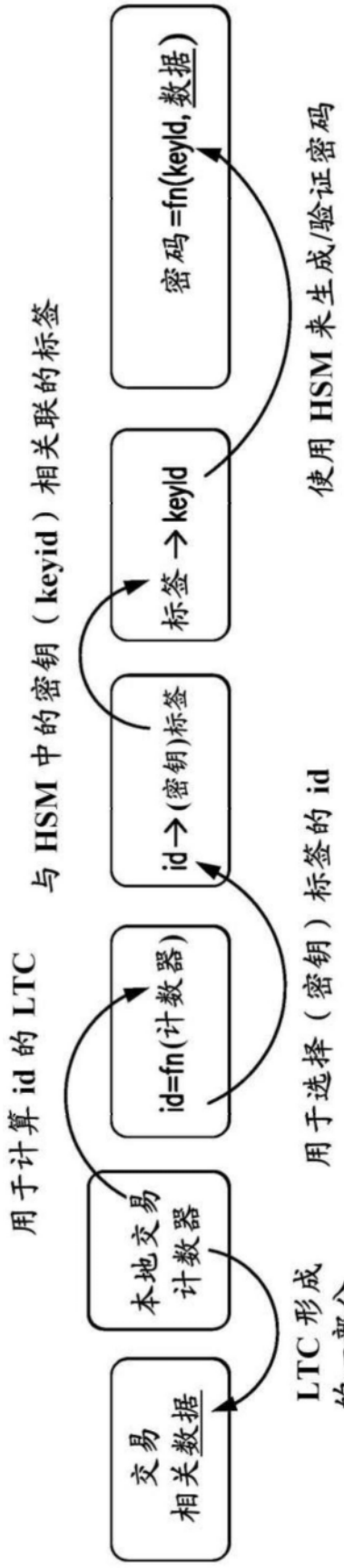

交易
相关数据
本地交易
计数器
id=fn(计数器)
id→(密钥)标签
标签→keyId
密码=fn(keyId,数据)
LTC 形成
的一部分...
用于选择（密钥）标签的 id
使用 HSM 来生成/验证密码
用于计算 id 的 LTC
与 HSM 中的密钥（keyid）相关联的标签

图26

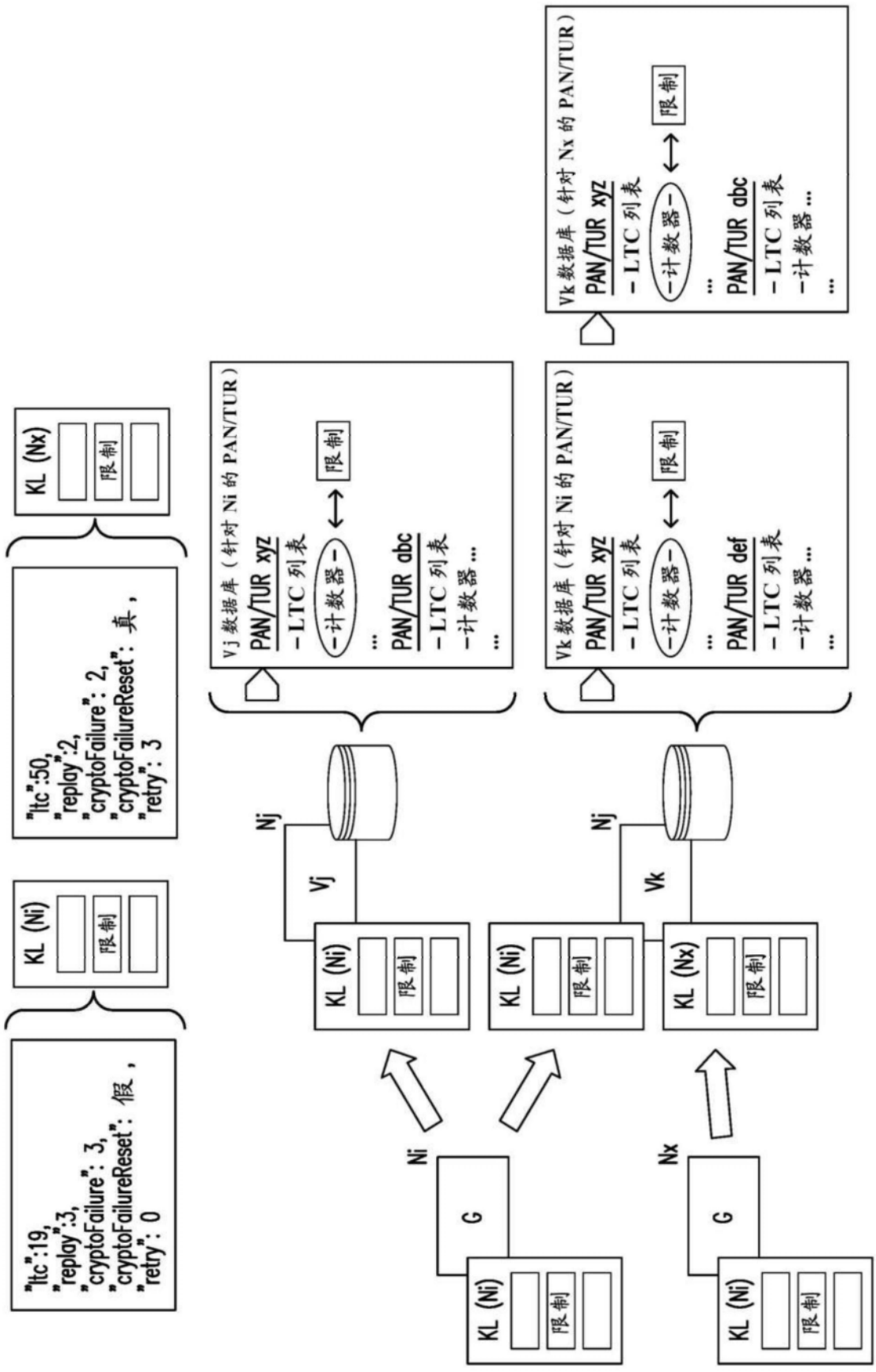

"ltc":19,
"replay":3,
"cryptoFailure": 3,
"cryptoFailureReset": 假 ,
"retry": 0
KL (Ni)
限制
"ltc":50,
"replay":2,
"cryptoFailure": 2,
"cryptoFailureReset": 真 ,
"retry": 3
KL (Nx)
Ni
G
KL (Ni)
限制
Nx
G
KL (Ni)
限制
Nj
Vj
KL (Ni)
限制
Nj
Vk
KL (Ni)
限制
KL (Nx)
限制
Vj 数据库（针对 Ni 的 PAN/TUR）
PAN/TUR xyz
-LTC 列表
-计数器-
限制
...
PAN/TUR abc
-LTC 列表
-计数器...
...
Vk 数据库（针对 Ni 的 PAN/TUR）
PAN/TUR xyz
-LTC 列表
-计数器-
限制
...
PAN/TUR def
-LTC 列表
-计数器...
...
Vk 数据库（针对 Nx 的 PAN/TUR）
PAN/TUR xyz
-LTC 列表
-计数器-
限制
...
PAN/TUR abc
-LTC 列表
-计数器...
...

图27