(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 110622482 B
(45)授权公告日 2022.02.22

(21)申请号 201880031375.0
(22)申请日 2018.05.31
(65)同一申请的已公布的文献号
申请公布号 CN 110622482 A
(43)申请公布日 2019.12.27
(30)优先权数据
15/611,229 2017.06.01 US
(85)PCT国际申请进入国家阶段日
2019.11.12
(86)PCT国际申请的申请数据
PCT/IB2018/053877 2018.05.31
(87)PCT国际申请的公布数据
WO2018/220570 EN 2018.12.06
(73)专利权人 国际商业机器公司
地址 美国纽约阿芒克
(72)发明人 李承达 熊伟翔 孙维孝 吴明勋
(74)专利代理机构 北京市柳沈律师事务所 11105
代理人 邸万奎
(51)Int.Cl.
H04L 9/40(2022.01)
H04L 67/14(2022.01)
(56)对比文件
US 2016315913 A1,2016.10.27
US 2015288679 A1,2015.10.08
US 2013191631 A1,2013.07.25
CN 104702611 A,2015.06.10
CN 106790285 A,2017.05.31
审查员 高露

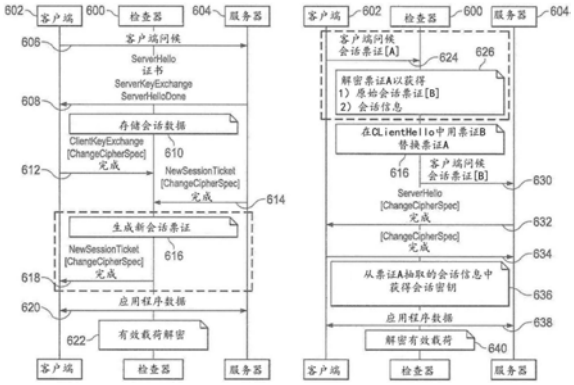
权利要求书2页 说明书13页 附图6页

(54)发明名称

TLS检查中的无高速缓存会话票证支持

(57)摘要

基于网络的设备包括一种可提供具有会话恢复功能、但无需维护会话高速缓存的TLS检查的机制。为此,检查器被配置为使TLS客户端实际上代表TLS检查器参与维护会话上下文。在操作中,检查器首先从TLS服务器接收会话票证,检查器生成并向客户端发出包含原始票证和含有会话密钥的会话上下文信息的复合票证,代替对会话票证进行高速缓存。检查器对复合票证进行加密,以保护会话信息。当TLS客户端提供复合会话票证以恢复TLS连接时,检查器解密该票证并直接从中检索会话上下文。然后,检查器用原始会话票证来恢复TLS会话。



CN 110622482 B

1.一种在位于传输层安全性(TLS)客户端和TLS服务器之间的中介中使用的、在TLS会话期间提供TLS检查功能的方法,包括:

在从TLS服务器接收到原始会话票证后,生成新会话票证,其中,新会话票证是通过将变换函数应用于所述原始会话票证和包括为在会话期间使用而协商的密码套件、主密钥的会话上下文信息而得出的值;

对所述新会话票证应用加密函数;

将加密的新会话票证输出到TLS客户端;

从TLS客户端接收到加密的新会话票证后,解密加密的新会话票证以恢复所述原始会话票证和包括所述主密钥的会话上下文信息;和

使用恢复的原始会话票证和会话上下文信息来恢复所述TLS会话。

2.根据权利要求1所述的方法,其中,所述变换函数将所述原始会话票证和所述会话上下文信息连接在一起。

3.根据权利要求1所述的方法,其中,所述加密的新会话票证是从所述TLS客户端接收的,在Client Hello(客户端问候)中。

4.根据权利要求3所述的方法,其中,通过用原始会话票证替换所述Client Hello中的加密的新会话票证,并将所述Client Hello转发到TLS服务器,来至少部分地恢复所述TLS会话。

5.根据前述权利要求中的任一项所述的方法,其中,使用所述会话上下文信息包括获得会话密钥。

6.根据权利要求5所述的方法,还包括:使用所述会话密钥来解密所述原始会话票证的有效载荷。

7.根据权利要求1-4中的任一项所述的方法,其中,生成所述新会话票证并且输出所述加密的新会话票证以代替在会话高速缓存中缓存所述原始会话票证。

8.根据权利要求5所述的方法,其中,生成所述新会话票证并且输出所述加密的新会话票证以代替在会话高速缓存中缓存所述原始会话票证。

9.一种用于在位于传输层安全性(TLS)客户端和TLS服务器之间的中介中使用的、在TLS会话期间提供TLS检查功能的装置,包括:

处理器;

包含由所述处理器执行的计算机程序指令的计算机存储器,所述计算机程序指令包括程序代码,所述程序代码被配置为:

在从TLS服务器接收到原始会话票证后,生成新会话票证,其中,新会话票证是通过将变换函数应用于所述原始会话票证和包括为在会话期间使用而协商的密码套件、主密钥的会话上下文信息而得出的值;

对所述新会话票证应用加密函数;

将加密的新会话票证输出到TLS客户端;

从TLS客户端接收到加密的新会话票证后,解密加密的新会话票证以恢复所述原始会话票证和包括所述主密钥的会话上下文信息;和

使用恢复的原始会话票证和会话上下文信息来恢复所述TLS会话。

10.根据权利要求9所述的装置,其中,所述变换函数将所述原始会话票证和所述会话

上下文信息连接在一起。

11.根据权利要求9所述的装置,其中,所述加密的新会话票证是从所述TLS客户端接收的,在Client Hello(客户端问候)中。

12.根据权利要求11所述的装置,其中,通过用原始会话票证替换所述Client Hello中的加密的新会话票证,并将所述Client Hello转发到TLS服务器,来至少部分地恢复所述TLS会话。

13.根据权利要求9至12中的任一项所述的装置,其中,使用所述会话上下文信息包括获得会话密钥。

14.根据权利要求13所述的装置,还包括:使用所述会话密钥来解密所述原始会话票证的有效载荷。

15.根据权利要求9至12中的任一项所述的装置,其中,生成所述新会话票证并且输出所述加密的新会话票证以代替在会话高速缓存中缓存所述原始会话票证。

16.根据权利要求13所述的装置,其中,生成所述新会话票证并且输出所述加密的新会话票证以代替在会话高速缓存中缓存所述原始会话票证。

17.一种在TLS会话期间用于传输层安全性(TLS)检查功能的计算机可读存储介质,所述计算机可读存储介质可由处理电路读取,并存储由所述处理电路执行以执行根据权利要求1至8中任一项所述的方法的指令。

TLS检查中的无高速缓存会话票证支持

技术领域

[0001] 本公开总体上涉及联网设备上的信息安全。

背景技术

[0002] 安全威胁在不断演变。随着尖端Web应用程序的快速增长和文件共享的增加,过去可能被认为无害的活动可能会成为攻击者的潜在机会。传统的安全手段,例如反恶意软件和防火墙,已变得更容易被绕开。因此,迫切需要更高级、更主动的威胁防护,以帮助提供针对新出现的威胁的全面安全性。

[0003] 在许多计算环境中,网络连接的非显示设备(“设备”)无处不在。例如,专门为执行传统的面向中间件服务的体系结构(SOA)功能而构建的设备在某些计算机环境中十分普遍。SOA中间件设备可以简化、帮助保护或加速XML和Web服务的部署,同时在企业中扩展现有的SOA基础架构。利用中间件专用硬件和轻量级中间件堆栈可以解决常规软件解决方案所承受的性能负担。此外,设备的外形尺寸为实现中间件SOA功能提供了安全、易用的包装。这些类型的设备提供的一个特殊优势是可以从后端系统卸载处理。为此,使用这样的中间件设备来执行与网络安全有关的计算昂贵的过程是众所周知的。例如,网络入侵防御系统(IPS)设备被设计在位于企业网络的入口点,以保护关键业务资产(例如内部网络、服务器、端点和应用程序)免受恶意威胁。

[0004] 将基于安全套接字层(SSL)和/或基于传输层安全性(TLS)的加密用于网络通信,通常会抑制从网络内部识别和缓解威胁流量的能力。现在,估计所有业务网络流量的三分之二或更多通过SSL/TLS传输。这意味着依赖于网络通信的组织通常无法(从网络)保护企业中可能受到此类威胁的端点。实际上,绝大多数SSL/TLS通信仅使用服务器身份验证,即,服务器通过SSL/TLS协议向客户端进行身份验证,但是客户端未针对服务器进行身份验证。这种身份验证的不对称性为将自身置于客户端和服务器之间以便能够解密通信和检查内容的进程提供了机会。这种“中间人”(MITM)进程可能是恶意的,也可能出于合法原因而使用的,例如数据包检查(用于威胁检测)。

[0005] 因此,已知在客户端和服务器之间提供透明的MITM代理,该代理可以配置为创建和管理两个单独的SSL/TLS会话,一个作为客户端到目标服务器的,另一个作为服务器到启动客户端的。因此,中间代理在服务器上显示为客户端,在客户端上显示为预期的服务器。从客户端发起的通信以及从服务器接收到的任何响应,理论上都可以用于SSL/TLS检查器的检查和后续操作。

[0006] 当执行SSL/TLS连接的中间人检查时,SSL/TLS检查器支持TLS会话恢复是重要的要求。常规上,众所周知,TLS服务器根据两(2)种不同的方法之一,即会话ID(RFC 4507)和会话票证(RFC 5077),向客户端(不考虑MITM)提供会话恢复。会话ID是被发明用来加快SSL/TLS握手并赋能会话恢复的第一个机制。在会话ID中,所有会话信息都存储在服务器端。会话ID会话恢复有几个缺点,最主要的是,由于存在大量缓存的会话,因此需要花费时间和空间来查找给定的会话ID,这是性能瓶颈。会话ID的另一个主要缺点是,一个会话ID只

能在一个服务器上工作,这使得部署很难扩展,除非会话高速缓存(session caches)在服务器之间同步。为了解决会话ID的不足,开发了会话票证。会话票证是一种使TLS服务器能够恢复会话并避免保持每个客户端会话状态的机制。为此,TLS服务器将会话状态封装到一个票证中并将其转发给客户端。该票证由服务提供商签名。客户端随后可以使用获取的票证恢复会话。当服务器稍后收到一个票证并确定它已由服务提供商签名时,它将接受该票证中存储的所有设置。会话票证由于其可扩展性和服务器端较少的资源开销而在Web服务器中广泛使用。

[0007] 为了在TLS检查中支持会话票证(例如,通过透明的MITM代理),必须进行高速缓存以维护票证和会话密钥之间的映射。这是因为没有实用的方法来解密会话票证,因为加密机制是由应用程序/服务控制的,而不是由SSL/TLS规范定义的。但是,在检查器中维护会话高速缓存有几个缺点,包括由于难以分配会话高速缓存而导致的可扩展性差,由于可能导致代理用尽高速缓存的存储和内存限制而导致的会话高速缓存大小的限制,使高速缓存循环时间变得复杂并且限制可用于映射会话票证的哈希算法的类型的CPU边界,以及容易受到可能故意冲刷会话高速缓存中条目并由此绕过检查的攻击者的拒绝服务攻击。

[0008] 仍然需要在提供会话票证支持、但可以克服现有方法中的这些和相关缺陷的TLS检查器中提供TLS会话恢复。

[0009] 因此,在本领域中需要解决前述问题。

发明内容

[0010] 从第一方面来看,本发明提供了一种在位于传输层安全性(TLS)客户端和TLS服务器之间的中介中操作并且在TLS会话期间提供TLS检查功能的方法,该方法包括:在从TLS服务器接收到原始会话票证时,生成新会话票证;对新会话票证应用加密函数;将加密的新会话票证输出到TLS客户端;从TLS客户端接收到加密的新会话票证后,解密加密的新会话票证以恢复原始会话票证和会话上下文信息;使用恢复的原始会话票证和会话上下文信息来恢复TLS会话。

[0011] 从另一方面来看,本发明提供一种设备,包括:处理器;包含由处理器执行的计算机程序指令的计算机存储器,该计算机程序指令包括程序代码,该程序代码被配置为:在从TLS服务器接收到原始会话票证时,生成新会话票证;对新会话票证应用加密函数;将加密的新会话票证输出到TLS客户端;在从TLS客户端接收到加密的新会话票证后,解密加密的新会话票证以恢复原始会话票证和会话上下文信息;使用恢复的原始会话票证和会话上下文信息来恢复TLS会话。

[0012] 从另一方面来看,本发明提供一种用于在TLS会话期间进行传输层安全性(TLS)检查功能的计算机程序产品,该计算机程序产品包括可由处理电路读取并存储指令的计算机可读存储介质,所述指令供处理电路执行以执行用于执行本发明的步骤的方法。

[0013] 从另一方面来看,本发明提供一种存储在计算机可读介质上并且可加载到数字计算机的内部存储器中的计算机程序,该计算机程序包括当所述程序在计算机上运行时用于执行本发明的步骤的软件代码部分。

[0014] 从另一方面来看,本发明提供一种位于传输层安全性(TLS)客户端和TLS服务器之间的设备,包括:硬件处理器,以及保存被配置为无高速缓存TLS检查器机制的计算机程序

指令的计算机存储器，所述计算机程序指令用于执行以下操作：从TLS服务器接收会话票证，并作为响应：(a)生成复合会话票证，和(b)将复合会话票证输出到TLS客户端以代替缓存会话票证，并从TLS客户端发送的票证接收复合会话，作为响应：(c)恢复会话票证，和(d)使用恢复的会话票证来恢复与TLS服务器的会话。

[0015] 基于网络的设备包括使该设备能够提供具有会话恢复的TLS检查的机制，而无需在检查器中维护会话缓存。为此，检查器被配置得没有会话缓存，因此不再维护(从TLS服务器接收的)会话票证与会话上下文之间的映射。相反，检查器的配置方式是使TLS客户端实际上代表TLS检查器来参与维护会话上下文。在操作中，当检查器首先从TLS服务器接收会话票证时，代之以并代替对其进行缓存时，检查器生成并向客户端发出包括原始票证和含有一个或多个会话密钥的会话上下文信息的复合票证，而不是缓存该会话票证。通常，有两个会话密钥，一个用于客户端，一个用于服务器端。优选地，检查器对复合票证(或复合会话票证)进行加密以保护会话信息。当TLS客户端(反过来向检查器)提供复合会话票证以恢复TLS连接时，检查器解密该票证并直接从中检索会话上下文。然后，检查器使用原始会话票证恢复到TLS服务器的TLS会话。这种方法消除了任何缓存查找，甚至消除了在TLS检查器中维护本地会话缓存的要求。相反，实际上，复合票证成为会话票证本身的缓存。

[0016] 概括来说，根据本公开的第一方面，一种方法在位于客户端和服务器之间的中介中操作并且提供TLS检查功能。从TLS服务器接收到原始会话票证后，生成新会话票证，而不是将原始会话票证高速缓存在会话缓存中。新会话票证包括通过将变换函数应用于原始会话票证和会话上下文信息(即，为在会话期间使用而协商的密码套件、主密钥等)而得出的值。变换函数可能会有所不同。代表性的变换函数将会话上下文信息连接到原始会话票证。然后将加密函数应用于新会话票证(有时在本文中称为复合会话票证)以保护会话信息。然后将加密的变换后的会话票证传递到发出请求的TLS客户端。稍后从TLS客户端收到该票证后，将对其解密以恢复原始会话票证和会话上下文信息。然后使用所恢复的信息，恢复与TLS服务器的TLS会话。

[0017] 根据本公开的第二方面，一种装置位于传输层安全性(TLS)客户端和TLS服务器之间，以在TLS会话期间提供TLS检查功能。该设备包括一组一个或多个硬件处理器，以及计算机存储器，该计算机存储器保持由硬件处理器执行的计算机程序指令，以执行诸如上述方法步骤的一组操作。

[0018] 根据本公开的第三方面，描述了一种用于数据处理系统的非暂时性计算机可读介质中的计算机程序产品。数据处理系统位于传输层安全性(TLS)客户端和TLS服务器之间，以在TLS会话期间提供TLS检查功能。该计算机程序产品保存在数据处理系统中执行并被配置为执行诸如上述方法步骤的操作的计算机程序指令。

[0019] 前述内容概述了所公开主题的一些更相关的特征。这些特征应被解释为仅仅是说明性的。如将要描述的，通过以不同的方式应用所公开的主题或通过修改主题，可以获得许多其他有益的结果。

附图说明

[0020] 为了更全面地理解本主题及其优点，现在参考以下结合附图进行的描述，其中：

[0021] 图1描绘了其中可以实现说明性实施例的示例性方面的分布式数据处理环境的示

例性框图；

[0022] 图2是其中可以实现说明性实施例的示例性方面的数据处理系统的示例性框图；

[0023] 图3示出了其中可以实现所公开的主题的示例性的基于网络的安全设备；

[0024] 图4示出了如何在中间人设备中处理传统的SSL/TLS通信以促进安全流量的检查；

[0025] 图5示出了使用结合有会话高速缓存的TLS检查器的TLS会话恢复；和

[0026] 图6描绘了根据本公开的主题的TLS会话恢复，其中TLS检查器提供无高速缓存会话票证支持。

具体实施方式

[0027] 现在参考附图，特别是参考图1和2，提供了可以在其中实现本公开的说明性实施例的数据处理环境的示例性图。应当理解，图1-2仅是示例性的，无意于主张或暗示对可以在其中实现所公开的主题的方面或实施例的环境的任何限制。在不脱离本发明的范围的情况下，可以对所描绘的环境进行许多修改。

[0028] 客户端-服务器技术

[0029] 现在参考附图，图1描绘了其中可以实现说明性实施例的各方面的示例性分布式数据处理系统的图示。分布式数据处理系统100可以包括其中可以实现说明性实施例的各方面的计算机网络。分布式数据处理系统100包含至少一个网络102，该网络是用于在分布式数据处理系统100中连接在一起的各种设备和计算机之间提供通信链接的介质。网络102可以包括诸如有线、无线通信链接或光缆之类的连接。

[0030] 在所示示例中，服务器104和服务器106与存储器108一起连接到网络102。此外，客户端110、112和114也连接到网络102。这些客户端110、112和114可以是例如个人计算机、网络计算机等。在所示示例中，服务器104向客户端110、112和114提供诸如引导文件、操作系统映像和应用程序的数据。在所示示例中，客户端110、112和114是服务器104的客户端。分布式数据处理系统100可以包括其他服务器、客户端和其他未显示的设备。

[0031] 在所示示例中，分布式数据处理系统100是具有网络102的因特网，该网络102代表使用协议的传输控制协议/互联网协议(TCP/IP)套件进行相互通信的网络和网关的全球集合。因特网的核心是主要节点或主机之间的高速数据通信线路的骨干，该主机由成千上万个路由数据和消息的商业、政府、教育和其他计算机系统组成。当然，分布式数据处理系统100也可以被实现为包括许多不同类型的网络，例如内联网、局域网(LAN)、广域网(WAN)等。如上所述，图1旨在作为示例，而不作为对所公开主题的不同实施例的体系结构限制，因此，图1中所示的特定元素不应被认为是对其中可以实现本发明的说明性实施例的环境的限制。

[0032] 现在参考图2，示出了在其中可以实现说明性实施例的各方面的示例性数据处理系统的框图。数据处理系统200是诸如图1中的客户端110的计算机的示例，其中设置有实现本公开的说明性实施例的过程的计算机可用代码或指令。

[0033] 参考图2，示出了其中可以实现说明性实施例的数据处理系统的框图。数据处理系统200是诸如图1中的服务器104或客户端110的计算机的示例，其中可以定位实现说明性实施例的过程的计算机可用程序代码或指令。在该说明性示例中，数据处理系统200包括通信结构202，该通信结构在处理器单元204、内存206、持久性存储器208、通信单元210、输入/输

出(I/O)单元212和显示器214之间提供通信。

[0034] 处理器单元204用于执行可以被加载到内存206中的软件的指令。处理器单元204可以是一个或多个处理器的集合,或者可以是多处理器核心,这取决于特定的实现。此外,可以使用一个或多个异构处理器系统来实现处理器单元204,其中一个主处理器与一个次级处理器一起存在于一个芯片上。作为另一个说明性示例,处理器单元204可以是包含多个相同类型的处理器的对称多处理器(SMP)系统。

[0035] 内存206和持久性存储器208是存储设备的示例。存储设备是能够临时地和/或永久地存储信息的任何硬件。在这些示例中,内存206可以是例如随机存取存储器或任何其他合适的易失性或非易失性存储设备。持久性存储器208可以取决于特定的实现而采取各种形式。例如,持久性存储器208可以包含一个或多个组件或设备。例如,持久性存储器208可以是硬盘驱动器、闪存、可重写光盘、可重写磁带或上述的某种组合。持久性存储器208所使用的介质也可以是可移动的。例如,可移动硬盘驱动器可用于持久性存储器208。

[0036] 在这些示例中,通信单元210提供与其他数据处理系统或设备的通信。在这些示例中,通信单元210是网络接口卡。通信单元210可以通过使用物理和无线通信链路之一或二者来提供通信。

[0037] 输入/输出单元212允许与可以连接到数据处理系统200的其他设备进行数据的输入和输出。例如,输入/输出单元212可以为用户通过键盘和鼠标输入提供连接。此外,输入/输出单元212可以将输出发送到打印机。显示器214提供向用户显示信息的机制。

[0038] 操作系统和应用程序或程序的指令位于持久性存储器208上。这些指令可以被加载到内存206中以由处理器单元204执行。不同实施例的过程可以由处理器单元204使用可以位于诸如内存206之类的存储器中的计算机实现的指令来执行。这些指令被称为程序代码、计算机可用程序代码或计算机可读程序代码,可由处理器单元204中的处理器读取和执行。不同实施例中的程序代码可以体现在不同的物理或有形计算机可读介质上,例如内存206或持久性存储器208上。

[0039] 程序代码216以功能形式位于可选择性地移除的计算机可读介质218上,并且可以被加载到数据处理系统200或传送到数据处理系统200以便由处理器单元204执行。在这些示例中,程序代码216和计算机可读介质218构成计算机程序产品220。在一个示例中,计算机可读介质218可以是有形的形式,例如光盘或磁盘,该光盘或磁盘被插入或放置在作为持久性存储器208的一部分的驱动器或其他设备中,以转移到诸如是持久性存储器208的一部分的硬盘驱动器的存储设备。以有形的形式,计算机可读介质218也可以采用连接到数据处理系统200的诸如硬盘驱动器、拇指驱动器或闪存之类的持久性存储器的形式。有形形式的计算机可读介质218也称为计算机可记录存储介质。在某些情况下,计算机可记录介质218可能不可移动。

[0040] 备选地,可以通过到通信单元210的通信链路和/或到输入/输出单元212的连接,将程序代码216从计算机可读介质218传输到数据处理系统200。在说明性示例中,该通信链路或连接可以是物理的或无线的。计算机可读介质还可以采取非有形介质的形式,例如包含程序代码的通信链路或无线传输。针对数据处理系统200示出的不同组件并不意味着对可以实现不同实施例的方式提供架构上的限制。可以在数据处理系统中实现不同的说明性实施例,该数据处理系统包括除针对数据处理系统200而示出的那些组件之外或代替其的

组件。图2的中所示的其他组件可以与所示的示例不同。作为一个示例，数据处理系统200中的存储设备是可以存储数据的任何硬件设备。内存206、持久性存储器208和计算机可读介质218是有形形式的存储设备的示例。

[0041] 在另一示例中，总线系统可以用于实现通信结构202，并且可以由一个或多个总线组成，例如系统总线或输入/输出总线。当然，可以使用提供在附接到总线系统的不同组件或设备之间的数据传输的任何适当类型的架构来实现总线系统。另外，通信单元可以包括一个或多个用于发送和接收数据的设备，例如调制解调器或网络适配器。另外，存储器可以是例如内存206或诸如在接口和存储器控制器集线器中发现的高速缓存，其可以存在于通信结构202中。

[0042] 可以用一种或多种编程语言的任何组合来编写用于执行本发明的操作的计算机程序代码，所述编程语言包括诸如Java™、Smalltalk、C++、C#、Objective-C之类的编程语言和传统的过程编程语言。程序代码可以完全在用户计算机上执行，部分在用户计算机上执行，作为独立软件包执行，部分在用户计算机上并且部分在远程计算机上执行，或者完全在远程计算机或服务器上执行。在后一种情况下，远程计算机可以通过任何类型的网络(包括局域网(LAN)或广域网(WAN))连接到用户的计算机，或者可以与外部计算机建立连接(例如通过使用因特网服务提供商的因特网)。Java和所有基于Java的商标和徽标是Oracle和/或其分支机构的商标或注册商标。

[0043] 本领域普通技术人员将认识到，图1和图2中的硬件可能取决于实施方式而有所不同。除了或代替图1-2所示的硬件，可以使用其他内部硬件或外围设备，例如闪存、等效的非易失性存储器或光盘驱动器等。而且，在不脱离所公开的主题的范围的情况下，说明性实施例的处理可以应用于除前述的SMP系统之外的多处理器数据处理系统。

[0044] 如将看到的，本文描述的技术可以结合诸如图1所示的标准客户端-服务器范例而操作，其中客户端计算机与在一组一个或多个计算机上执行的可访问因特网的基于Web的门户进行通信。最终用户操作能够访问门户并与门户交互的可连接因特网的设备(例如台式计算机、笔记本计算机、支持因特网的移动设备等)。通常，每个客户端或服务器机器是诸如图2所示的数据处理系统，包括硬件和软件，这些实体通过诸如因特网、内联网、外联网、专用网络或任何其他通信介质或链路的网络彼此通信。数据处理系统通常包括一个或多个处理器、一个操作系统、一个或多个应用程序以及一个或多个实用程序。数据处理系统上的应用程序提供对Web服务的本地支持，包括但不限于对HTTP、SOAP、XML、WSDL、UDDI和WSFL的支持。有关SOAP、WSDL、UDDI和WSFL的信息可从万维网联盟(W3C)获得，该联盟负责制定和维护这些标准。有关HTTP和XML的更多信息可从因特网工程任务组(IETF)获得。假定熟悉这些标准。

[0045] 作为进一步的背景，安全套接字层/传输层安全性(SSL/TLS)是一种众所周知的加密协议，用于保护网络(如因特网)上的通信安全。诸如SSL/TLS之类的密码协议通常基于公共密钥密码系统，例如RSA(Rivest、Shamir和Adelman)加密算法。对于传统的基于RSA的SSL会话，连接的双方约定了一个“主控密码”(pre-master secret)，用于为会话的其余部分生成参数。通常，双方使用RSA非对称加密来建立主控密码，而无需以明文形式交换实际值。在操作中，SSL客户端生成主控密码，并使用SSL服务器的公开RSA密钥对其进行加密。这将生成一个加密的主控密码(ePMS)，然后将其提供给SSL服务器。SSL服务器具有专用解密密钥，

然后将其用于解密加密的主控密码。此时，客户端和服务器都有原始的主控密码，可以使用它来生成用于实际的加密和安全数据交换的对称密钥。

[0046] 网络上的加密流量是通过一个信任链发生的。每个Web服务器都有一个证书，该证书会出示给每个客户端(通常是Web浏览器)，以表明他们就是他们所说的身份。Web服务器通常从可以证明Web服务器合法性的授权机构(证书颁发机构或CA)获取这些证书。服务器的证书指示从其获得证书的授权机构(“颁发者”)。Web浏览器通常具有他们信任的颁发者列表。当向Web浏览器提供来自Web服务器的证书时，浏览器将检查颁发者并将其与它的受信任列表进行匹配。如果找到匹配项，则连接将继续。如果找不到匹配项，浏览器通常会显示警告，并可能拒绝连接。除了受信任的事实以外，CA不一定是一个特殊的实体。任何实体都可以将自己设置为信任或签署证书。证书可以信任自己，这称为自签名证书。要使用SSL/TLS与客户端进行互操作，必须创建客户端将隐式信任的证书。关于网络设备(如下所述)，假定管理员可以配置企业客户端以信任该设备来签署证书。实际上，设备的颁发者便位于浏览器的受信任颁发者列表中。

[0047] 网络连接的安全设备

[0048] 网络设备通常是机架安装的设备。该设备包括使设备可以用作敏感信息的安全保险库的物理安全性。通常，该设备被制造，预装软件，然后在企业或其他网络操作环境中或与之结合部署;可替代地，可以将盒子定位在本地，然后配备以可以在例如私有或本地云计算环境内被安全地部署和管理的标准或定制的中间件虚拟映像。该设备可能包括硬件和固件加密支持，可能用于加密硬盘上的数据。没有用户(包括管理用户)可以访问物理磁盘上的任何数据。特别地，优选地，操作系统(例如Linux®)锁定根帐户并且不提供命令外壳，并且用户没有文件系统访问权限。通常，设备不包括显示设备、CD或其他光盘驱动器或任何USB、火线(Firewire)或其他使设备能连接到该设备的端口。它被设计为一个封闭且安全的环境，具有有限的可访问性，然后仅由经过身份验证和授权的个人使用。Linux是Linus Torvalds在美国和/或其他国家的注册商标。

[0049] 参看图3，代表性的操作环境包括物理设备300，其连接到网络302。该设备可以使用诸如以上关于图2所描述的数据处理系统来实现，其可以代表图1中所示的服务器(或客户端)之一。典型地，设备300包括基于Web 2.0的用户界面(UI)、命令行界面(CLI)和基于REST的应用程序编程界面(API)。在该示例中，已经向该设备提供了包括操作系统304、应用服务器306、HTTP服务器308和其他应用程序310的映像。该映像中可以包括其他软件解决方案(未显示)。这些软件元素可以预装在该设备上，其中可以包括其他数据(例如模板、脚本、文件等)。当然，特定的软件配置将取决于对设备的使用。该设备包括一个或多个存储设备(例如磁盘315)。存储设备的类型和数量可以有所不同。

[0050] 安全网络通信的拦截、解密和检查

[0051] 作为进一步的背景，图4示出了根据已知技术的用于拦截、解密和检查安全网络通信的已知中间人(MITM)设备400的基本操作。该设备在如上所述并且在图2中示出的安全网络设备中实现。更一般地，该设备是诸如图1所示的计算系统。

[0052] 如图所示，设备400连接在客户端402和服务器404之间。客户端和服务器被配置为使用SSL或TLS保护通信。假定您熟悉SSL/TLS。在该实施例中，设备400通过创建和管理两(2)个单独的SSL/TLS会话(其中一个作为到目标服务器404的客户端进程X_{SS} 406，另一个作

为到发起客户端402的服务器进程X_{cs} 408)，在客户端402和服务器404之间提供透明的(或中间人)代理。X_{ss}和X_{cs}组件有时在本文中称为SSL实例，其中SSL实例通常是组成SSL会话的一段代码。SSL会话(或会话上下文)是在两个端点之间发生的通信本身。中间代理400因此在服务器404上作为客户端出现，并且在客户端402上作为预期服务器出现。从客户端402发起的通信以及从服务器404接收的任何响应，然后可用于检查(或其他处理，例如重写)和随后的动作。为此，设备400可以包括协议分析模块(例如，安全网络保护PAM)，其提供分组检查功能以识别并可能减轻网络威胁。该模块(或可能受支持其他分组检查应用程序)的特定细节不是本公开的一方面。

[0053] 在操作中，如图4所示，在初始TCP握手(未示出)之后，客户端402生成SSL/TLS会话发起请求消息(以下称为“Client Hello”)，以开始到服务器的SSL/TLS握手。这是步骤1。代理拦截此连接，并将其定向到面向客户端的服务器组件X_{cs}408。在步骤2，X_{cs}组件读取该Client Hello，解释数据，并通常以Server Hello消息、证书(Certificate)和服务器完成(Done)消息来响应客户端402。在步骤3中，将在设备内部配置和设置全新的SSL连接。这是由X_{ss}发起的面向服务器的连接。X_{ss}然后生成一个新的Client Hello(此处称为ClientHello2，以使其与步骤1中的ClientHello区别开)，并将该(新的)Client Hello发送到服务器。在步骤4中，服务器404读取该新的Client Hello，并响应以ServerHello2、Certificate2和ServerDone2。这些消息又与X_{cs}在步骤2中发布给客户端的消息不同。结果，存在两(2)个不同的连接，一个在客户端402和X_{cs} 408之间，另一个在X_{ss} 406和服务器404之间。在这一点上，如果(例如通过PAM或其他应用程序的)MITM处理确定这不是希望检查的(客户端-服务器)连接，则系统要么必须继续检查该连接(也许忽略结果)，要么将其完全关闭。该确定可以以任何方便的方式进行，例如通过针对从服务器接收到的证书中的信息(上面的证书2)执行基于策略的规则匹配来进行。

[0054] TLS检查中的无高速缓存会话票证支持

[0055] 以上述为背景，现在描述本公开的TLS检查中的无高速缓存会话票证支持。假定熟悉按照Internet RFC 5077的TLS会话票证。可实施该技术的代表性商业产品是下一代入侵防御系统(IPS)IBM®QRadar®网络安全(XGS)(以前称为IBM安全性网络保护(XGS))。当然，该商业产品的标识并不旨在进行限制，因为该方法可以在任何中间设备、设备、产品或系统中执行。IBM和QRadar是国际商业机器公司在全球许多司法管辖区注册的商标。

[0056] 如上所述，并且根据本公开，诸如所描述的基于网络的设备包括使设备能够提供具有会话恢复的TLS检查、但是不需要在检查器中保持会话高速缓存的机制。为此，将检查器配置为不具有会话高速缓存(或者可替代地，具有不使用的高速缓存)，因此不再维护(从TLS服务器接收的)会话票证与会话上下文之间的映射。相反，检查器的配置方式是使TLS客户端实际上代表TLS检查器来参与维护会话上下文。在操作中，当检查器首先从TLS服务器接收会话票证时，检查器不是对其进行高速缓存，而是会生成包括该原始票证和包含会话密钥的会话上下文信息的复合票证并将其发送给客户端。该复合票证(或复合会话票证)最好由检查器进行加密以保护会话信息。当TLS客户端将复合会话票证返回给检查器以恢复TLS连接时，检查器解密该票证并直接从中检索会话上下文。然后，检查器使用原始会话票证恢复TLS会话。这种方法消除了任何高速缓存查找，甚至消除了在TLS检查器中维护本地会话高速缓存的要求。相反，实际上，复合票证变成会话票证本身的高速缓存。

[0057] 众所周知,假定设备包括使(无论基于代理还是其他代理的)MITM进程能够例如在针对服务器证书或其他的规则匹配时恢复原始连接并重新连接原始端点(无需检查)的机制。TLS支持中的会话票证用于此目的。

[0058] 图5描绘了一个流程图,其示出了TLS中的常规会话票证,即具有包括会话高速缓存的检查器500。假定熟悉标准TLS握手语义。如图所示,图5中的流程图示出了位于TLS客户端502和TLS服务器504中间的检查器。在常规的TLS握手流程中,客户端502向服务器504发出Client Hello,这是步骤506。在步骤508,服务器504响应以各种消息,即Server Hello、证书、ServerKeyExchange和ServerHelloDone。会话上下文信息(密码套件、主控密钥等)在步骤510由检查器500存储。在步骤512,客户端502向服务器返回各种消息,即ClientKeyExchange、可选的ChangeCipherSpec和完成(Finished)。然后,服务器504响应以另一组消息,即NewSessionTicket(新会话票证)、对(如果由客户端发送的)ChangeCipherSpec消息的任何响应和完成。这样就完成了初始的TLS握手。在步骤516,检查器500将新票证存储在其本地缓存中。在使用中,并且如步骤518所示,应用程序数据在客户端502和服务器504之间流动。检查器500使用票证中的会话信息来解密有效载荷—如步骤520所示,以执行一个或多个检查器功能。检查器还将会话票证传递回客户端。

[0059] 检查器进行的一项或多项操作会中断会话,因此需要检查器具有进行恢复会话的能力。为此,并且如步骤522所示,假定客户端发出另一个客户端问候(Client Hello),这次传递会话票证(Session Ticket)。在步骤524,检查器在其本地缓存中执行对会话票证的查找。如果票证匹配,则检查器500向服务器发出新的客户端问候(Client Hello),传递会话票证。这是步骤526。在步骤528,服务器504响应以各种消息,即Server Hello(服务器问候)、证书(the Certificate)、可选的ChangeCipherSpec和完成(Finished)。在步骤530,客户端502然后响应以一组消息,即对(如果由服务器发送的)ChangeCipherSpec消息的任何响应和完成(Finished)。这样就完成了会话恢复所需的后续TLS握手。在步骤532,检查器500从其高速缓存中获得(在步骤510中存储的)一个或多个会话密钥。通常,有两个会话密钥,每个会话一个(客户端↔检查器,以及检查器↔服务器)。然后,当应用程序数据在客户端和服务器之间传递时(步骤532),检查器根据需要用会话密钥来解密有效负载。这是步骤534。

[0060] 图6类似于图5,但是该图描绘了根据本公开的技术对常规会话高速缓存方法的改变。如已描述的,修改后的技术在TLS检查中提供了“无高速缓存的”会话票证支持。为此,将检查器配置为不再维护(从TLS服务器接收的)会话票证与会话上下文之间的映射。相反,将检查器配置为使TLS客户端实际上代表TLS检查器来参与维护会话上下文。这种方法消除了任何高速缓存查找,甚至消除了在TLS检查器中维护本地会话高速缓存的要求。相反,实际上,检查器与客户端之间的交互为检查器提供了一种将高速缓存从其自己的会话高速缓存卸载到请求客户端上的方式,这最终会(尽管以更改的形式)存储会话票证。实际上,客户端变成会话票证本身的高速缓存。

[0061] 在操作中,当检查器首先从TLS服务器接收会话票证时,检查器不是对其进行高速缓存,而是生成并向客户端发出包括原始票证和含有会话密钥的会话上下文信息的复合票证。优选地,检查器对该复合票证(或复合会话票证)进行加密以保护会话信息。当TLS客户端将复合会话票证返回给检查器以恢复TLS连接时,检查器解密该票证并直接从其中检索

会话上下文。然后，检查器用原始会话票证恢复到TLS服务器的TLS会话。

[0062] 图6中描绘了无高速缓存会话票证支持。再次，检查器600被示为位于TLS客户端602和TLS服务器604之间的中间。与图5不同的是，检查器600不需要合并会话高速缓存。与传统的TLS握手流程一样，客户端602向服务器604发出客户端问候(Client Hello)。这是步骤606。在步骤608，服务器404响应以各种消息，即服务器问候、证书、ServerKeyExchange(服务器密钥交换)和ServerHelloDone(服务器问候完成)。再次，在步骤610，由检查器600存储会话上下文信息(密码套件、主控密钥等)。在步骤612，客户端602返回各种消息，即，ClientKeyExchange(客户端密钥交换)，可选的ChangeCipherSpec和完成(Finished)。但是，这次，这些消息被检查器拦截，而不是(如图5所示的那样)传递给服务器。在步骤614，服务器604提供NewSessionTicket(新会话票证)消息，可选的ChangeCipherSpec消息和完成(Finished)消息。NewSessionTicket包括原始会话票证。在步骤616，检查器600生成新会话票证，称为[A]，而不是存储由服务器604提供的原始会话票证(如图5中的步骤516)。下面描述用于生成新会话票证的优选技术的更多细节。然后，检查器600向客户端602输出一组消息，即NewSessionTicket[A](新会话票证[A])、可选的ChangeCipherSpec和完成(Finished)。这是步骤618。NewSessionTicket[A]消息包括新会话票证[A]。在步骤620，应用程序数据在客户端602和服务器604之间流动。如步骤622所示，检查器600用票证中的会话信息来解密有效载荷，以执行一个或多个检查器功能。

[0063] 如前所述，检查器进行的一个或多个操作中断会话，因此需要检查器600具有进行会话恢复的能力。为此，并且如步骤624所示，假定客户端发出了另一个客户端问候，这一次将新会话票证[A]传回到检查器。在步骤626，检查器600解密新会话票证[A]以恢复原始会话票证(称为[B])，并恢复会话信息。在步骤628，检查器600在Client Hello中用票证[B]替换票证[A]，并且在步骤630，检查器600将Client Hello(带有会话票[B])发送到服务器604。在步骤632，服务器604响应以各种消息，即Server Hello、证书、ChangeCipherSpec和完成(Finished)。在步骤634，客户端502然后响应以一组消息，即ChangeCipherSpec和完成(Finished)。这就完成了会话恢复所需的后续TSL握手。在步骤636，检查器560从在步骤626从票证[A]提取的会话信息中获取会话密钥。然后，在客户端和服务器之间传递应用程序数据(步骤638)时，检查器600根据需要使用该会话密钥解密有效载荷，这是步骤640。

[0064] 总结一下，通过比较图5中的步骤514、516和524与图6中的步骤616、618和626可以看出，这里的方法避免了将会话票证高速缓存在检查器本身中。相反，当检查器首先从TLS服务器接收会话票证时，检查器不是对其进行高速缓存，而是会生成复合票证并将其发送给客户端，该复合票证包括原始票证和含有会话密钥的会话上下文信息。优选地，检查器对复合票证(或复合会话票证)进行加密以保护会话信息。当TLS客户端向检查器返回复合会话票证以恢复TLS连接时，检查器解密该票证并直接从中检索会话上下文。然后，检查器用原始会话票证来恢复到TLS服务器的TLS会话。

[0065] 以下描述用于生成新会话票证的优选技术。这是图6中的步骤616。优选地，如下地产生新的会话票证：

[0066] Encrypt(Transform(原始会话票证、会话信息)、加密密钥)，其中会话信息包括密码信息，诸如密码套件、主控密钥等，Transform(变换)是将所标识的数据—即原始会话票证、会话信息—组合在一起的任何方法或计算，而Encrypt(加密)则是指能保护新会话票证

的任何加密算法。典型的Transform可以是连接操作，但这不是限制，因为更复杂的计算方法可以应用于该变换。当然，变换必须具有关联的逆(inverse)，以便可以恢复原始数据。对于加密包装器，典型的解决方案可以利用公共密钥密码术，以便加密密钥是具有关联的私钥或密钥(private or secret key)的公钥，当以后从客户端接收到新会话票证时，该私钥或密钥将用于解密。因此，如上所述，通过首先将变换函数应用于与TLS会话相关联的原始会话票证和会话上下文信息来生成新会话票证，然后将加密函数应用于该变换的结果。应用该变换的结果有时在本文中称为“复合”会话票证或新会话票证。检查器将加密的新会话票证输出到客户端。

[0067] 当检查器在后续Client Hello中从客户端接收到加密的新会话票证时，由检查器通过解密该票证而获得(恢复)原始会话票证、会话信息，然后用原始会话票证替换Client Hello中的新会话票证，来恢复会话。如上所述，优选地用在检查器处维护的公钥对的私钥来执行解密。最好有两个解密操作，即：Get_Ticket(解密(新会话票证))和Get_SessionInfo(解密(新会话票证))，如果解密成功，则解密后的有效载荷应包含原始会话票证和会话信息。Get_Ticket函数获取/恢复原始会话票证，而Get_SessionInfo函数则获取解密有效负载所需的会话信息，诸如主控TLS密钥。

[0068] 上述方法具有许多优点。当前提供TLS会话票证支持的TLS检查器机制存在一些缺点，这些缺点可以通过所描述的方法来克服。首先，不需要在检查器中维护一个会话高速缓存。结果，克服了已知方法的缺点(可扩展性差、处理和内存效率低下以及拒绝服务(DoS)利用的可能性)。该方法易于实现，因为可以轻松地以计算有效的方式实现附加功能。该方法是高度可靠和安全的。通过卸载会话票证本身的高速缓存，该方法可提高检查器机制的性能，从而增强中介程序的整体操作。

[0069] 本文的技术还促进了TLS支持基础结构的扩展。因此，例如，当实施多个检查器时，只要优选地预先在检查器之间同步变换函数(例如，通过共享解密复合票证所需的密钥)，每个检查器都能理解复合会话票证并且然后恢复检查。

[0070] 尽管已经在代理的上下文中描述了这些技术，但这不是限制性的。概括而言，本文所述的处理可以在位于客户端和服务器之间的任何中介中进行。在一个这样的实施例中，中介提供透明的内联内容检查和修改。客户端和服务器是计算实体(端点)。中介可以被配置为物理设备、虚拟设备或其某种组合。它可以用于许多不同的应用程序，包括但不限于加密(SSL/TLS)会话的解密，以便可以按照前面描述的方式执行安全检查。

[0071] 尽管已经描述了优选的操作环境和用例(安全设备)，但是本文的技术可以在其中期望截取、解密、检查和/或修改(重写)往返计算系统或设备的网络流量的任何其他操作环境中使用。

[0072] 如上所述，上述功能可以以独立的方式实现，例如由处理器执行的基于软件的功能，或者可以将其用作服务(包括通过SOAP/XML接口作为Web服务)。本文描述的特定硬件和软件实现细节仅出于说明目的，并不意味着限制所描述主题的范围。

[0073] 更一般地，在所公开主题的上下文内的计算设备各自是包括硬件和软件的数据处理系统(诸如图2中所示)，并且这些实体通过诸如因特网、内联网、外联网、专用网络或任何其他通信介质或链接互相通信。数据处理系统上的应用程序为Web和其他已知服务和协议提供本机支持，包括但不限于对HTTP、FTP、SMTP、SOAP、XML、WSDL、UDDI和WSFL等的支持。有

关SOAP、WSDL、UDDI和WSFL的信息可从万维网联盟(W3C)获得，该联盟负责制定和维护这些标准。有关HTTP、FTP、SMTP和XML的更多信息，可以从因特网工程任务组(IETF)获得。假定熟悉这些已知的标准和协议。

[0074] 本文描述的技术可以在各种客户端侧架构(例如、防火墙、NAT设备)中或与之结合，并且可以包括简单n层架构、Web门户、联合系统等。可以在松耦合服务器(包括基于“云”的)环境中实践本文的技术。

[0075] 更一般地，本文描述的主题可以采取完全硬件实施例、完全软件实施例或既包含硬件又包含软件元素的实施例的形式。在优选实施例中，可信平台模块功能在软件中实现，该软件包括但不限于固件、驻留软件、微代码等。此外，下载和删除界面和功能可以采取可从计算机可用或计算机可读介质访问的计算机程序产品的形式，该计算机可用或计算机可读介质提供程序代码以供计算机或任何指令执行系统使用或与其结合使用。为了本说明的目的，计算机可用或计算机可读介质可以是任何能包含或存储由指令执行系统、装置或设备使用或与其结合使用的程序的设备。介质可以是电子、磁性、光学、电磁、红外或半导体系统(或装置或设备)。计算机可读介质的示例包括半导体或固态存储器、磁带、可移动计算机磁盘、随机存取存储器(RAM)、只读存储器(ROM)、刚性磁盘和光盘。光盘的当前示例包括光盘-只读存储器(CD-ROM)，光盘-读/写(CD-R/W)和DVD。该计算机可读介质是有形的非暂时性物品。

[0076] 计算机程序产品可以是具有用于实现一个或多个所描述的功能的程序指令(或程序代码)的产品。那些指令或代码可以在通过网络从远程数据处理系统上下载之后存储在数据处理系统中的非暂时性计算机可读存储介质中。或者，那些指令或代码可以存储在服务器数据处理系统中的计算机可读存储介质中，并且适合于通过网络下载到远程数据处理系统，以供在远程系统内的计算机可读存储介质中使用。

[0077] 在代表性实施例中，界面和实用程序在专用计算平台中实现，优选在一个或多个处理器执行的软件中实现。该软件被维护在与一个或多个处理器相关联的一个或多个数据存储或存储器中，并且该软件可以被实现为一个或多个计算机程序。这些专用硬件和软件共同包括上述功能。

[0078] 尽管以上描述了由本发明的某些实施例执行的操作的特定顺序，但是应当理解，这样的顺序是示例性的，因为替代实施例可以以不同的顺序执行操作，组合某些操作，重叠某些操作，如此等等。说明书中对给定实施例的引用指示所描述的实施例可以包括特定的特征、结构或特性，但是每个实施例可以不必包括该特定的特征、结构或特性。

[0079] 最后，尽管已经分别描述了系统的给定组件，但是本领域的普通技术人员将理解，一些功能可以在给定指令、程序序列、代码部分等中组合或共享。

[0080] 设备不限于任何特定类型。同样，上述操作也可以与任何已知的技术或机制结合使用，这些技术或机制本身就是用来拦截、解密、检查、修改、重写和重新加密来自任何计算机的数据，而与计算机的物理配置无关。

[0081] 本文的技术总体提供对技术或技术领域的上述改进，以及对如上所述的网络连接的安全设备的特定技术改进。

[0082] 提供不具有会话高速缓存的TLS检查器的概念(该技术的“无高速缓存”性质)不一定要求在物理上将TLS检查器配置得没有这种高速缓存，尽管这将是通常的配置。具有未被

利用或以其他方式绕过的会话高速缓存的TLS检查器在所公开的要求保护的主题的范围之内。因此，如本文所用的“无高速缓存”，指的是会话高速缓存被省略或者如果存在则未被利用。

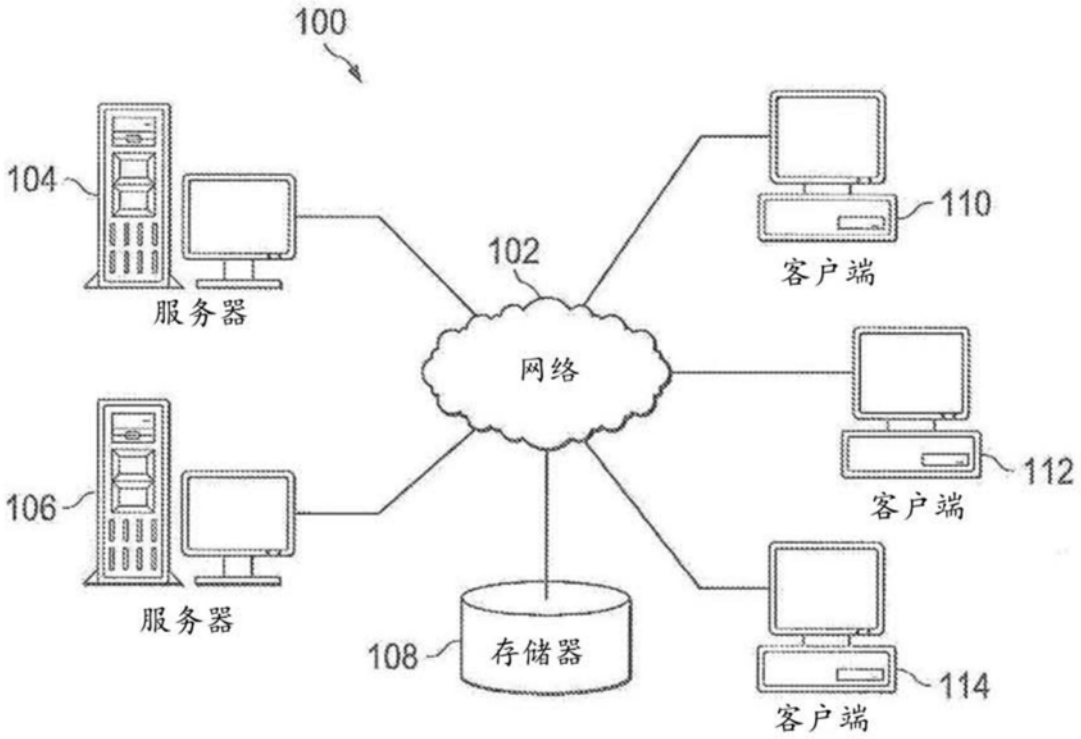


图1

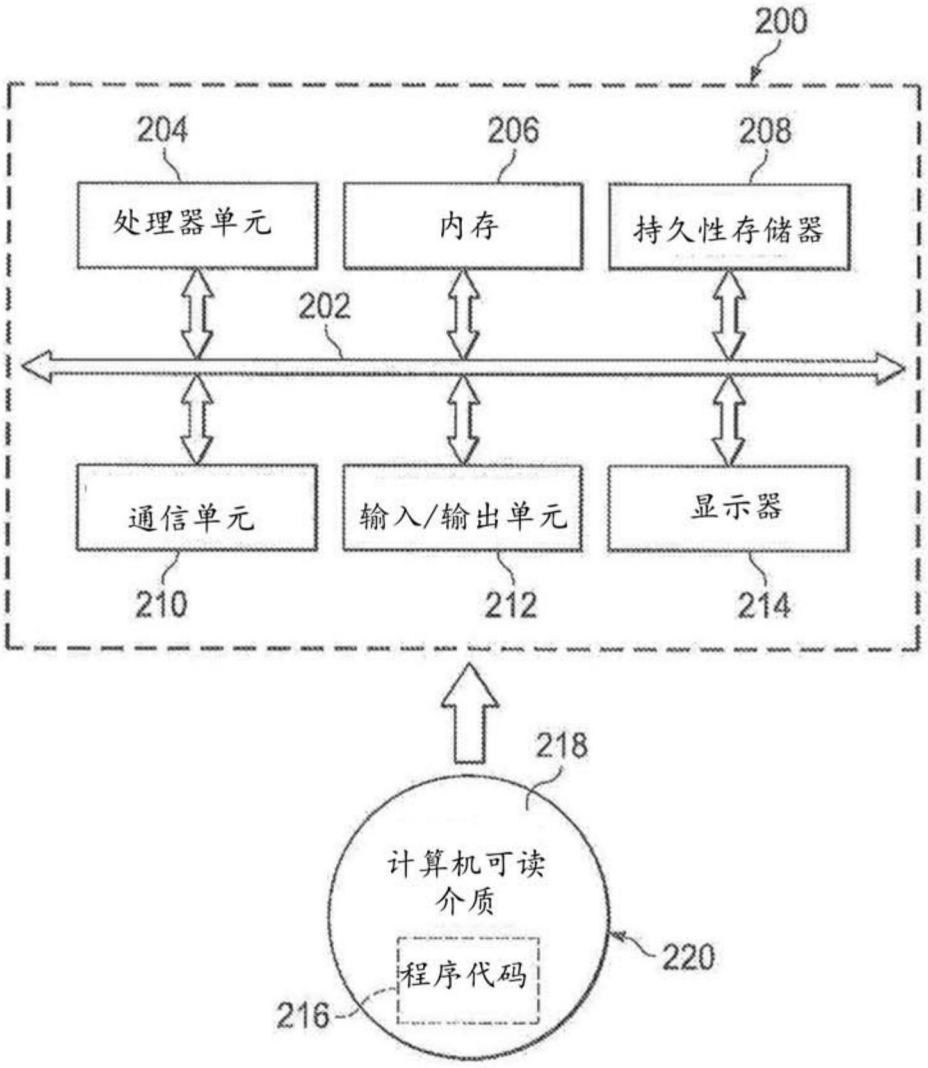

200
204
206
208
处理器单元
内存
持久性存储器
202
通信单元
输入/输出单元
显示器
210
212
214
218
计算机可读介质
程序代码
216
220

图2

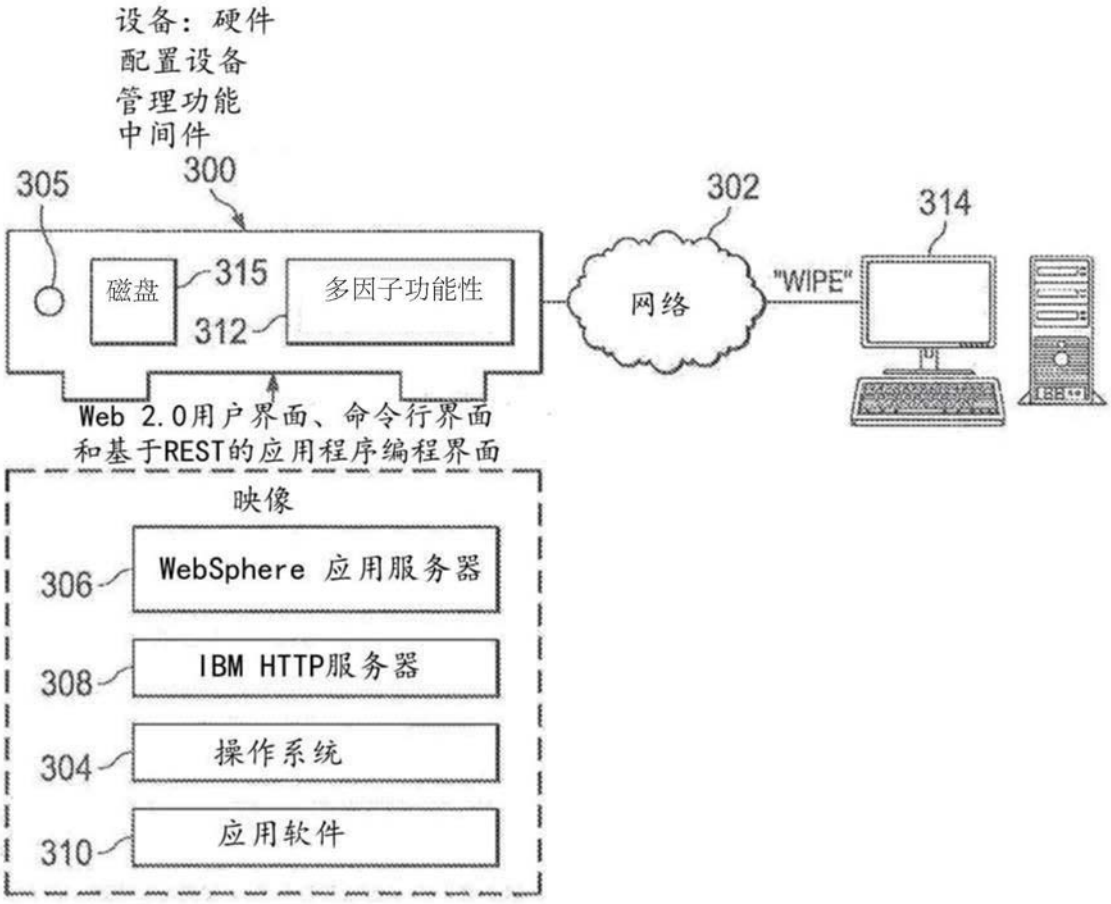

设备：硬件
配置设备
管理功能
中间件
300
305
302
314
磁盘
315
多因子功能性
312
网络
"WIPE"
Web 2.0用户界面、命令行界面
和基于REST的应用程序编程界面
映像
306
WebSphere 应用服务器
308
IBM HTTP服务器
304
操作系统
310
应用软件

图3

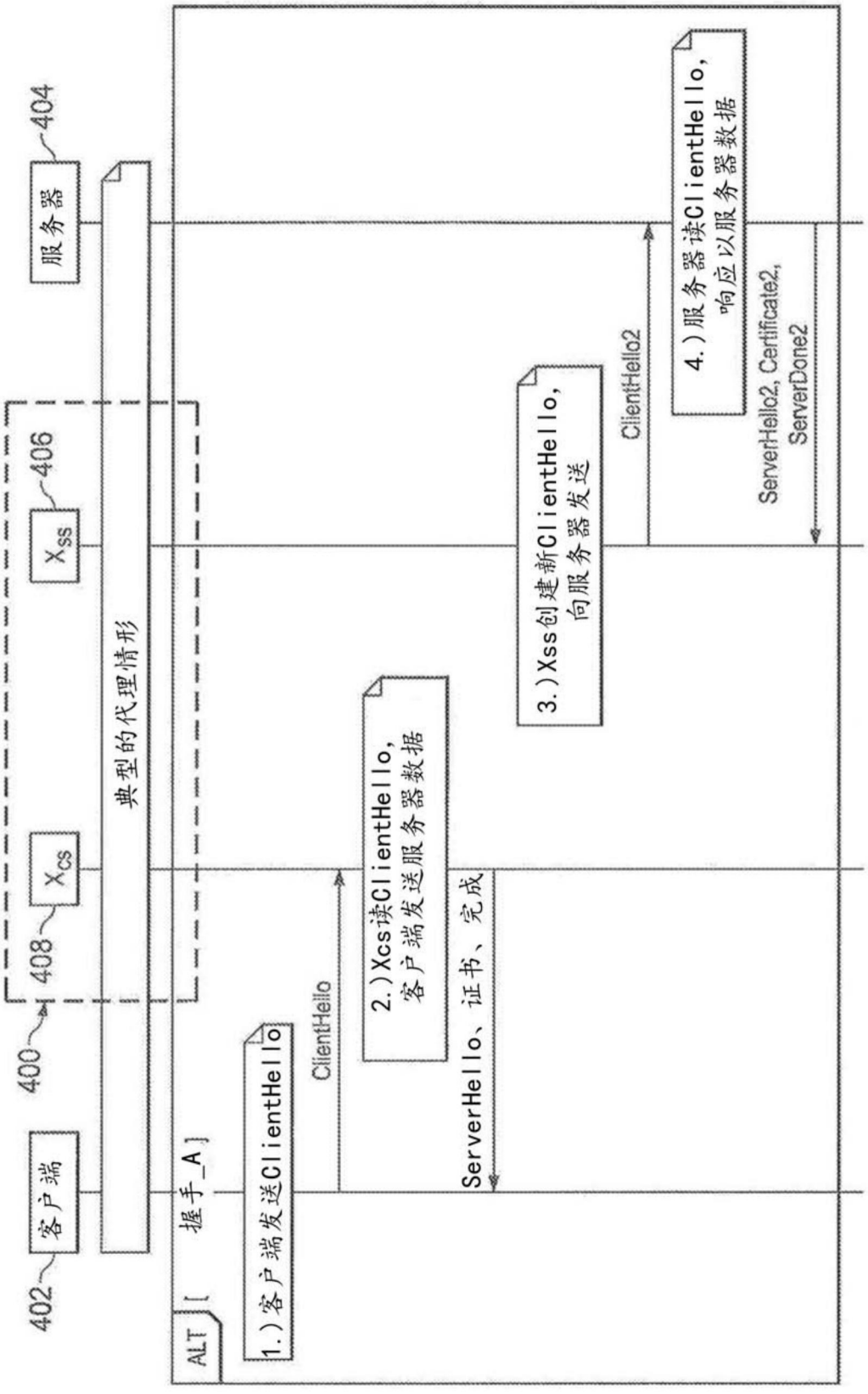

402
客户端
400
408
X_{CS}
406
X_{SS}
404
服务器
典型的代理情形
ALT
[握手_A]
1.)客户端发送ClientHello
ClientHello
2.)Xcs读ClientHello,
客户端发送服务器数据
ServerHello、证书、完成
3.)Xss创建新ClientHello,
向服务器发送
ClientHello2
4.)服务器读ClientHello,
响应以服务器数据
ServerHello2, Certificate2,
ServerDone2

图4

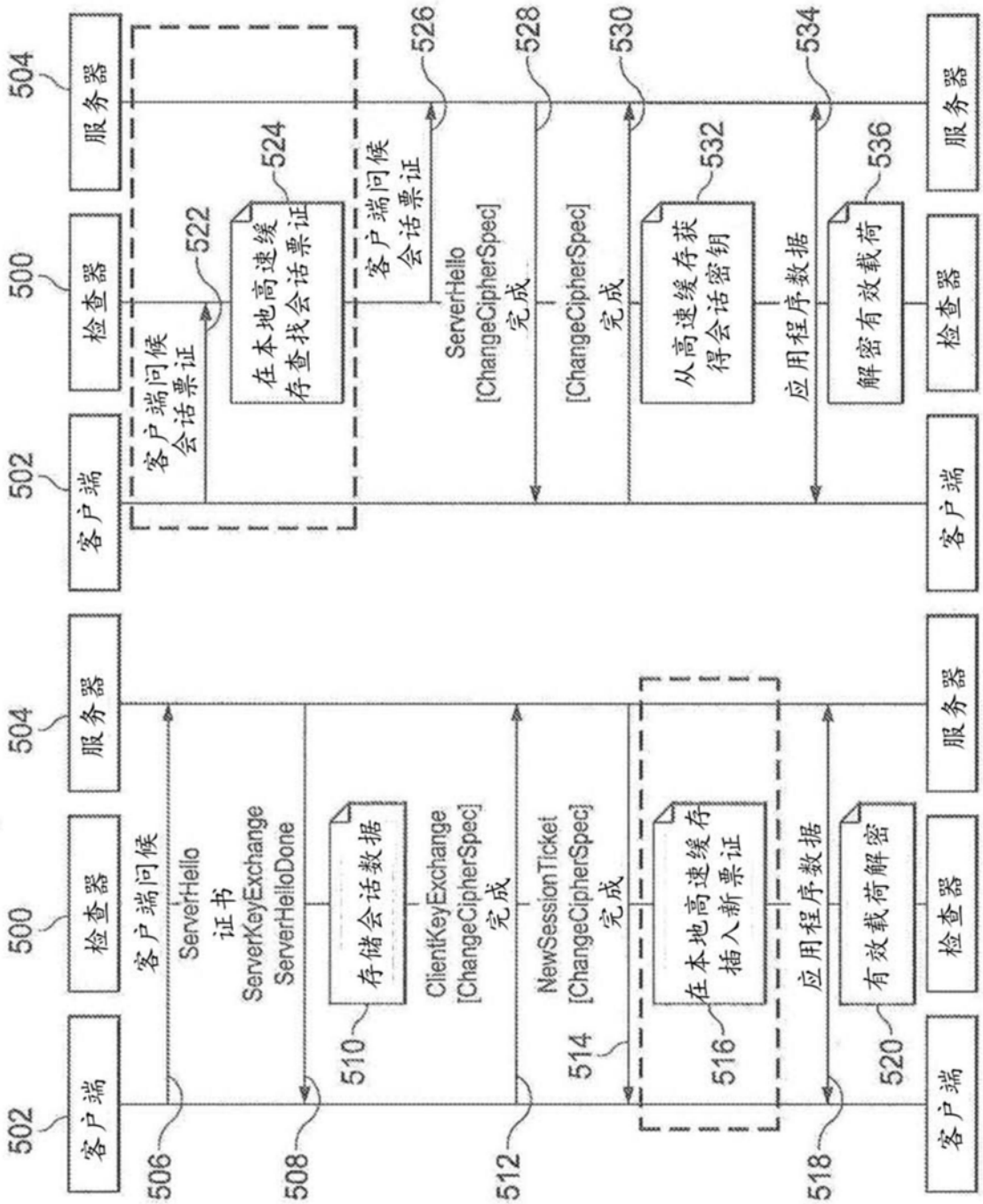

502
客户端
500
检查器
504
服务器
客户端问候
ServerHello
证书
ServerKeyExchange
ServerHelloDone
506
存储会话数据
510
508
ClientKeyExchange
[ChangeCipherSpec]
完成
512
NewSessionTicket
[ChangeCipherSpec]
完成
514
在本地高速缓存
插入新票证
516
应用程序数据
518
有效载荷解密
520
客户端问候
会话票证
522
在本地高速缓
存查找会话票证
524
客户端问候
会话票证
526
ServerHello
[ChangeCipherSpec]
完成
528
[ChangeCipherSpec]
完成
530
从高速缓存获
得会话密钥
532
应用程序数据
534
解密有效载荷
536

图5

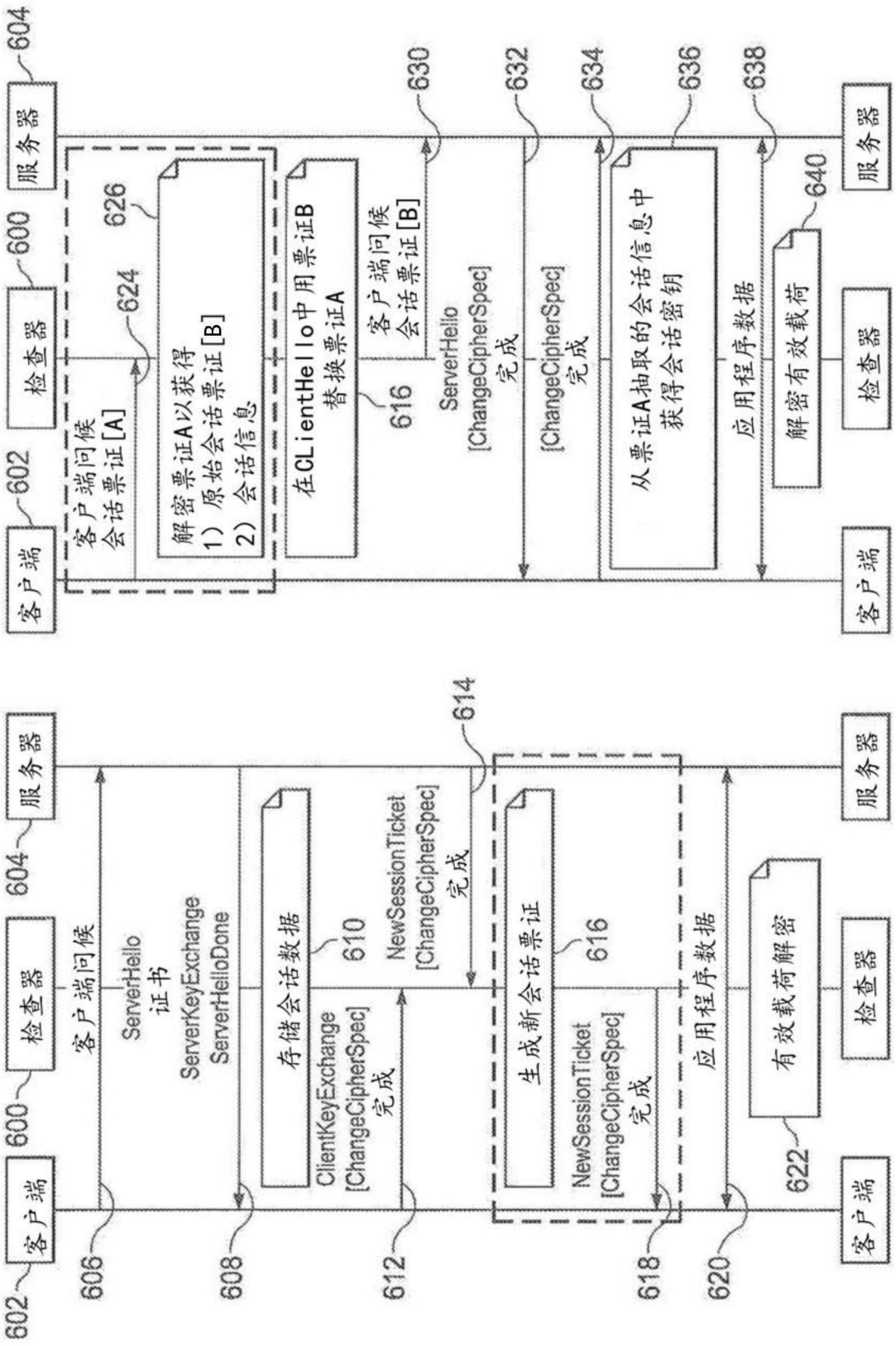

客户端
602
检查器
600
服务器
604
客户端问候
606
ServerHello
证书
ServerKeyExchange
ServerHelloDone
608
存储会话数据
ClientKeyExchange
[ChangeCipherSpec]
完成
610
612
NewSessionTicket
[ChangeCipherSpec]
完成
614
生成新会话票证
NewSessionTicket
[ChangeCipherSpec]
完成
616
618
应用程序数据
620
有效载荷解密
622
客户端问候
会话票证[A]
624
解密票证A以获得
1）原始会话票证[B]
2）会话信息
626
在ClientHello中用票证B
替换票证A
616
客户端问候
会话票证[B]
630
ServerHello
[ChangeCipherSpec]
完成
632
[ChangeCipherSpec]
完成
634
从票证A抽取的会话信息中
获得会话密钥
636
应用程序数据
638
解密有效载荷
640

图6