



(12) 发明专利

(10) 授权公告号 CN 101517979 B

(45) 授权公告日 2012. 05. 30

(21) 申请号 200780035490. 7

(51) Int. Cl.

(22) 申请日 2007. 09. 20

H04L 12/28 (2006. 01)

(30) 优先权数据

H04L 29/06 (2006. 01)

2123/DEL/2006 2006. 09. 26 IN  
11/561, 947 2006. 11. 21 US

(56) 对比文件

(85) PCT申请进入国家阶段日

CN 1707997 A, 2005. 12. 14, 全文 .

2009. 03. 25

US 6598167 B2, 2003. 07. 22, 全文 .

审查员 杨凯鹏

(86) PCT申请的申请数据

PCT/US2007/079007 2007. 09. 20

(87) PCT申请的公布数据

W02008/039682 EN 2008. 04. 03

(73) 专利权人 微软公司

地址 美国华盛顿州

(72) 发明人 V·贾殷 M·阿皮亚

K·C·瓦尼娅拉詹 S·贾殷

(74) 专利代理机构 上海专利商标事务所有限公司 31100

代理人 张政权

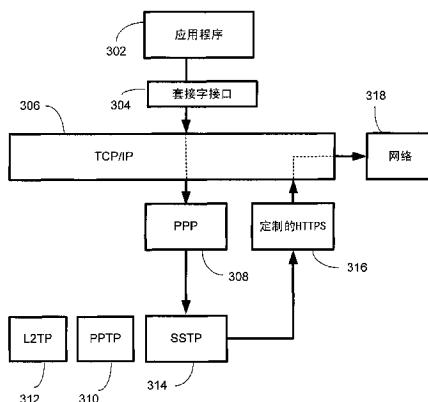
权利要求书 2 页 说明书 11 页 附图 4 页

(54) 发明名称

通过 HTTPS 连接的安全隧道

(57) 摘要

许多安全的隧道要求诸如 L2TP 和 PPTP 等需要特殊处理、授权或安全证书的协议。这通常使它们不能用在企业或机构网络和外面的、公共的网络之间。安全套接字隧道协议 (SSTP) 在内核和用户模式中添加驱动程序, 以便通过常见的 HTTPS 端口来路由诸如 PPP 等标准协议业务。在网络中断的情况下, 交换会话 cookie 允许快速重新连接底层的 HTTPS 连接而不影响更高层的应用程序。



1. 一种支持本地和远程实体之间的安全通信的方法，包括：  
在所述本地和远程实体之间建立会话；  
使用 HTTPS 端口在所述会话上建立加密通信层；  
使用所述 HTTPS 端口在所述本地和远程实体之间发送控制数据；  
使用所述 HTTP 端口在所述会话上创建标准协议管道；  
使用所述标准协议管道通过所述 HTTPS 端口流传输经加密的数据传输。
2. 如权利要求 1 所述的方法，其特征在于，所述会话是 TCP 会话。
3. 如权利要求 1 所述的方法，其特征在于，所述加密通信层遵循安全套接字层和传输层安全中的一个。
4. 如权利要求 1 所述的方法，其特征在于，所述标准协议管道是点对点协议 PPP 连接。
5. 如权利要求 4 所述的方法，其特征在于，所述 PPP 协议是在远程访问连接管理器服务中管理的。
6. 如权利要求 1 所述的方法，其特征在于，还包括使用密钥来加密数据传输，所述密钥是在使用 HTTPS 端口在所述会话上建立所述加密通信层时协商的。
7. 如权利要求 1 所述的方法，其特征在于，在所述本地和远程实体之间发送控制数据包括共享表示客户机和服务器之间的会话连接数据的 cookie。
8. 如权利要求 6 所述的方法，其特征在于，还包括在所述会话断开后请求重新建立所述会话时，在所述本地和远程实体之间发送 cookie。
9. 如权利要求 1 所述的方法，其特征在于，在所述本地和远程实体之间发送控制数据包括支持用于管理以下之一的安全套接字隧道协议 SSTP：呼叫建立、以隧道传送任意协议、在传输中发生中断时的重新连接服务、和缓冲区管理。
10. 如权利要求 9 所述的方法，其特征在于，支持所述 SSTP 协议包括支持控制分组和数据分组共同的且包括控制 / 数据位和长度字段的 SSTP 头部，其中任一分组的净荷都小于 4095 字节。
11. 如权利要求 9 所述的方法，其特征在于，支持 SSTP 控制分组包括支持控制消息，所述控制消息包括连接请求、包括链路 cookie 的连接确认、连接否认、呼叫已连接、快速重新连接请求、快速重新连接确认、回送请求、回送响应、断开连接、断开连接确认、和异常中止呼叫中的至少一个。
12. 一种服务器，支持通过 HTTPS 连接到客户机进程的安全隧道，包括：  
用于支持使用公知协议的服务器侧连接的服务器模块；  
用于支持与所述客户机的 HTTPS 连接的客户机模块；  
用于支持连接到所述客户机进程的安全套接字隧道协议 SSTP 隧道的 SSTP 模块，所述 SSTP 模块提供对快速重新连接的支持，所述快速重新连接用于在维持安全协议连接的同时恢复断开的 PPP 连接。
13. 如权利要求 12 所述的服务器，其特征在于，所述安全协议连接是所述 HTTPS 连接。
14. 如权利要求 12 所述的服务器，其特征在于，所述 SSTP 模块提供控制和数据分组，所述控制分组包括关于以下中的至少一个的消息：连接请求、包括链路 cookie 的连接确认、连接否认、呼叫已连接、快速重新连接请求、快速重新连接确认、回送请求、回送响应、断开连接、断开连接确认、和异常中止呼叫。

15. 如权利要求 12 所述的服务器，其特征在于，所述 SSTP 模块将 cookie 传递给客户机进程以供在恢复所述断开的安全协议连接时使用。

16. 如权利要求 15 所述的服务器，其特征在于，所述 cookie 标识与所述断开的安全协议连接相关联的会话。

17. 一种支持客户机和服务器之间的安全通信的方法，包括：

从内核模式 TCP/IP 模块处的应用程序接收数据；

将所述数据路由到映射模块并确定所述数据的连接映射；

将所述数据传递到网络数据接口 NDIS 模块并使用公知协议准备所述数据；

在用户模式 HTTPS 模块处加密所准备的数据以形成经加密的数据；

使用端口 443 通过 TCP 连接从所述客户机向所述服务器发送所述经加密的数据。

18. 如权利要求 17 所述的方法，其特征在于，还包括：

经由所述内核模式中的、耦合到所述 NDIS 模块的安全协议驱动程序和用户模式中的、耦合到所述 HTTPS 模块的安全协议服务，将所述 NDIS 模块接口到所述 HTTPS 模块。

19. 如权利要求 18 所述的方法，其特征在于，来自所述安全协议服务的数据发送和接收操作使用快速 I/O 来进行与所述安全协议驱动程序的内核模式数据传输，来避免上下文切换和分页。

20. 如权利要求 17 所述的方法，其特征在于，所述公知协议是点对点协议连接。

## 通过 HTTPS 连接的安全隧道

[0001] 背景

[0002] 计算机之间的通信造成了关于数据业务安全，在某些情况下甚至关于端点计算机本身的漏洞点。多种技术可用于解决网络安全。存在用于保护单个逻辑连接上的应用程序到应用程序通信的技术，如通常用于 web 浏览器到 web 服务器超文本 HTTP 业务的安全套接字层（SSL）。存在通过保护网络连接本身而非两应用程序之间的业务来保护端点之间的所有数据业务的其它技术。其示例是若干形式的虚拟专用网络，如点对点隧道协议（PPTP）和第二层隧道协议（L2TP）/ 网际协议安全（IPSec）。虚拟专用网络在一端加密并在另一端解密来保护端点之间的所有业务免遭窃听和中间人攻击。

[0003] 然而，这些安全协议通常要求诸如 IPSec 证书等特殊设置，或使用往往被企业级防火墙阻塞的不标准的端口。隧道业务可在阻塞通用路由封装（GRE 阻塞）的因特网服务提供商（ISP）处被阻塞。而隧道协议的另一不方便之处是使用具有本地分配的 IP 地址的网络地址转换（NAT）。

[0004] 另外，即使创建了安全信道，或例程超时或网络问题引起的较低级连接中的中断也可以打断较高级应用程序连接。从这种服务中断中恢复通常要求在重新连接网络后的应用程序到应用程序恢复。

[0005] 概述

[0006] 被设计为对 NAT 和现有网络安全措施友好的隧道协议使用通过超文本传输协议安全（HTTPS）连接的安全隧道协议。与正常的 HTTPS 连接形成对比，被指定为安全套接字隧道协议即 SSTP 的安全隧道协议以例如点对点协议（PPP）等标准协议的方式来支持计算机和相关联的服务器之间的所有网络业务。HTTPS 实际上被所有防火墙和 ISP 接纳，并与 NAT 兼容。以下描述的 SSTP 协议使用若干技术来向应用程序呈现全面的、标准的连接协议，以便在无需修改或知晓底层连接的情况下使用。SSTP 甚至用内核和用户模式之间的若干跨越来维护其它常见协议的性能特性。SSTP 的另一方面支持在客户机和服务器之间的 cookie 交换，从而在连接丢失的情况下允许快速重新连接。该快速重新连接能力允许在应用程序知道任何中断之前重新建立连接。

[0007] 附图简述

[0008] 图 1 是适用于实现本发明的计算机的简化的和代表性框图；

[0009] 图 2 是经由 SSTP 连接来连接的两个计算机的简化的和代表性框图；

[0010] 图 3 是支持 SSTP 连接的功能块的简化的和代表性框图；以及

[0011] 图 4 是支持 SSTP 连接的一实施例的功能块的简化的和代表性框图。

[0012] 详细描述

[0013] 尽管下文阐明了众多不同实施例的详细描述，但是应当理解，该描述的法律范围由本发明所附的权利要求书的言辞来限定。该详细描述应被解释为仅是示例性的，且不描述每一可能的实施例，因为描述每一可能的实施例即使不是不可能的也是不切实际的。可使用现有技术或在本申请提交日之后开发的技术来实现众多替换实施例，而这仍落入权利要求书的范围之内。

[0014] 还应该理解，在本专利中，除非使用句子“如此处所用，术语‘\_\_\_\_\_’特此被定义为意指……”或者类似句子来明确地定义一个术语，否则不管是明确地还是含蓄地，都没有限制该术语意义超出其平常或普通意义的意图，并且，这一术语不应该被解释为被限制在基于本专利的任何部分中（除了权利要求书的语言之外）所做的任何陈述的范围内。就本专利所附的权利要求书中所述的任何术语在本专利中以与单数意义相一致的方式来引用而言，这是为简明起见而如此做的，仅仅是为了不使读者感到混淆，且这类权利要求术语并不旨在隐含地或以其它方式限于该单数意义。最后，除非一权利要求要素是通过叙述单词“装置”和功能而没有叙述任何结构来定义的，否则任何权利要求要素的范围并不旨在基于 35U.S.C. § 12 第 6 段的应用来解释。

[0015] 许多发明性功能和许多发明性原理最佳地使用或利用软件程序或指令以及诸如专用 IC 等集成电路 (IC) 来实现。期望本领域的普通技术人员虽然可能要进行大量的工作和由例如可用时间、现有技术以及经济问题促动的许多设计选择，但是当受到此处所公开的概念和原理的指引时仍能够容易地以最小的实验来生成这些软件指令和程序以及 IC。因此，为了简明以及最小化使根据本发明的原理和概念晦涩的任何风险，对这些软件和 IC（如果有的话）的进一步讨论将限于对于较佳实施例的原理和概念所必需的那些讨论。

[0016] 图 1 示出可以主存本发明各实施例的一个或多个的计算机 110 形式的计算设备，且被详细讨论以提供后续讨论的上下文。

[0017] 计算机 110 的组件可包括但不限于，处理单元 120、系统存储器 130 以及将包括系统存储器的各类系统组件耦合至处理单元 120 的系统总线 121。系统总线 121 可以是几种类型的总线结构中的任何一种，包括存储器总线或存储控制器、外围总线、以及使用各种总线体系结构中的任一种的局部总线。

[0018] 计算机 110 通常包括各种计算机可读介质。计算机可读介质可以是能由计算机 110 访问的任何可用介质，而且包含易失性、非易失性介质以及可移动和不可移动介质。作为示例而非局限，计算机可读介质可以包括计算机存储介质和通信介质。计算机存储介质包括以用于存储诸如计算机可读指令、数据结构、程序模块或其它数据等信息的任何方法或技术来实现的易失性和非易失性、可移动和不可移动介质。计算机存储介质包括，但不限于，RAM、ROM、EEPROM、闪存或其它存储器技术、CD-ROM、数字多功能盘 (DVD) 或其它光盘存储、磁带盒、磁带、磁盘存储或其它磁性存储设备、或能用于存储所需信息且可以由计算机 110 访问的任何其它介质。上述中任一组合也应包括在计算机可读介质的范围之内。

[0019] 系统存储器 130 包括易失性和 / 或非易失性存储器形式的计算机存储介质，如只读存储器 (ROM) 131 和随机存取存储器 (RAM) 132。基本输入 / 输出系统 133(BIOS) 包括如在启动时帮助在计算机 110 内的元件之间传输信息的基本例程，它通常储存在 ROM 131 中。RAM 132 通常包含处理单元 120 可以立即访问和 / 或目前正在其上操作的数据和 / 或程序模块。作为示例，而非限制，图 1 示出了操作系统 134、应用程序 135、其它程序模块 136 和程序数据 137。

[0020] 计算机 110 也可以包括其它可移动 / 不可移动、易失性 / 非易失性计算机存储介质。仅作为示例，图 1 示出了从不可移动、非易失性磁介质中读取或向其写入的硬盘驱动器 140，从可移动、非易失性磁盘 152 中读取或向其写入的磁盘驱动器 151，以及从诸如 CD ROM 或其它光学介质等可移动、非易失性光盘 156 中读取或向其写入的光盘驱动器 155。可以在

示例性操作环境中使用的其它可移动 / 不可移动、易失性 / 非易失性计算机存储介质包括但不限于, 磁带盒、闪存卡、数字多功能盘、数字录像带、固态 RAM、固态 ROM 等等。硬盘驱动器 141 通常由不可移动存储器接口, 诸如接口 140 连接至系统总线 121, 磁盘驱动器 151 和光盘驱动器 155 通常由可移动存储器接口, 诸如接口 150 连接至系统总线 121。

[0021] 上文讨论并在图 1 中示出的驱动器及其关联的计算机存储介质为计算机系统 110 提供了计算机可读指令、数据结构、程序模块和其它数据的存储。例如, 在图 1 中, 硬盘驱动器 141 被示为存储操作系统 144、应用程序 145、其它程序模块 146 和程序数据 147。注意, 这些组件可以与操作系统 134、应用程序 135、其它程序模块 136 和程序数据 137 相同, 也可以与它们不同。操作系统 144、应用程序 145、其它程序模块 146 和程序数据 147 在这里被标注了不同的标号是为了说明至少它们是不同的副本。

[0022] 计算机 110 可使用至一个或多个远程计算机, 如远程计算机 180 的逻辑连接在网络化环境中操作。远程计算机 180 可以是个人计算机、服务器、路由器、网络 PC、对等设备或其它常见网络节点, 且通常包括上文相对于计算机 110 描述的许多或所有元件, 尽管在图 1 中只示出存储器存储设备 181。图 1 中所示的逻辑连接包括局域网 (LAN) 171 和广域网 (WAN) 173, 但也可以包括其它网络。这样的联网环境在办公室、企业范围计算机网络、内联网和因特网中是常见的。

[0023] 当在 LAN 联网环境中使用时, 计算机 110 通过网络接口或适配器 170 连接至 LAN 171。当在 WAN 联网环境中使用时, 计算机 110 通常包括调制解调器 172 (电话、电缆、DSL 等) 或用于通过诸如因特网等 WAN 173 建立通信的其它装置。调制解调器 172 可以是内置或外置的, 它可以连接至系统总线 121、网络接口 170 或其它适当的机制。在网络化环境中, 相对于计算机 110 所描述的程序模块或其部分可被储存在远程存储器存储设备中。作为示例而非局限, 图 1 示出了远程应用程序 185 驻留在存储器设备 181 上。可以理解, 所示的网络连接是示例性的, 且可以使用在计算机之间建立通信链路的其它手段。

[0024] 图 2 是经由 SSTP 连接来连接的两个计算机的简化的和代表性框图。第一计算机 202, 例如支持诸如邮件、web 浏览、数据库访问等用户应用程序的客户机计算机, 可以耦合到第二计算机 204。在一示例性实施例中, 第一计算机 202 在企业或机构防火墙或安全区的外面。第二计算机 204 可以是支持第一计算机 202 上的应用程序的客户机 - 服务器通信的服务器。然而, 在许多实施例中, 第二计算机 204 可以是专用于支持来自企业或机构服务器或安全区外面的计算机的业务的远程访问服务器。第二计算机 204 可以在所谓的‘非军事化区’中, 其用来帮助受保护网络和外面的尝试访问受保护网络的实体之间的安全接口。两个计算机 202、204 可以经由网络 206 连接。应用程序 208 和由应用程序 n 210 表示的其它应用程序可以使用客户机网络接口 212 来发送和接收数据。客户机网络接口 212 可以向应用程序 208、210 呈现通信 API 214。通信 API 的一个示例是点对点协议 (PPP)。可以支持任何协议, 只要两侧都同意该协议。客户机网络接口 212 还包括用于耦合到网络 206 的 HTTPS 模块 216。

[0025] 在服务器侧, 对应于客户机网络接口 212 的服务器网络接口 218 可以包括耦合到网络 206 的 HTTPS 模块 220, 且还可以包括对应于客户机网络接口通信 API 214 的通信 API 224。通信 API 224 可以附加到主存应用程序 226、228、230 的一个或多个服务器上。在一实施例中, 服务器应用程序之一可以包括认证服务器 230。认证服务器 230 可被用来在会

话开始期间认证客户机凭证,且还可以包括对作为建立 HTTPS 会话的一部分的 SSL 密钥交换的支持。第二计算机 204 和各应用程序服务器 226、228、230 之间的业务可以使用正常的 IP/IPv6 路由协议来路由。

[0026] 并且,客户机上的一示例性实施例和应用程序,如 web 浏览器,可以开始并与诸如 ISP 等网络连接。可以建立从客户机网络接口 212 到服务器网络接口 218 的连接,以建立 SSTP 隧道。SSTP 隧道将在以下更详细讨论。在建立安全的 SSTP 隧道后,服务器网络接口 218 可以使用例如 PPP 等同意的协议,来向一个或多个服务器应用程序 226、228、230 转发业务。在一示例性实施例中,例如在企业环境中,认证服务器 230 可被用来建立第一计算机 202 处的用户的身份。一旦认证了用户,则用户可被授权访问一个或多个企业应用程序,如电子邮件、数据库访问、企业公告板等。

[0027] 图 3 是支持 SSTP 连接的功能块的示例性框图,其示出了来自图 2 的第一或第二计算机 202、204 的外出业务。通过 SSTP 协议的数据传输可以遵循三阶段过程:安全会话建立、SSTP 控制业务和 SSTP 数据业务。在安全会话建立期间,可以在客户机和服务器之间进行 TCP 连接,随后进行标准的 SSL 握手,包括 Diffie-Hellman 密钥交换。这建立了 HTTPS 会话。

[0028] 一旦 HTTPS 会话就绪,则 SSTP 驱动程序 (SSTPDVR) 可以激活管理 SSTP 协议的状态机。PPP 会话协商随后可以通过 SSTP 连接进行。在 PPP 会话就绪后,该信道为经由 PPP 协议的隧道应用程序业务做好准备。

[0029] 在初始会话设置和安全协商完成后,应用程序 302 可以向诸如公知的 Winsock 接口等套接字接口 304 发送数据。套接字接口 304 可以将数据沿着协议栈向下传递到 TCP/IP 接口 306。TCP/IP 接口随后可以确定该分组是去往 STP 隧道的,并将该数据路由到适当的协议层 - 在该实施例中是 PPP 模块 308。SSTP 协议与诸如 PPTP 310 或 L2TP 312 等其它安全协议存在于同一层。PPP 模块 308 执行 PPP 成帧,且封装将该数据传递到专用 SSTP 模块 314。SSTP 模块 314 处理内核和用户模式之间的交互,执行专门的缓冲,并支持 SSTP 命令集。经处理的数据被发送到 HTTPS 模块 316 以供使用 SSL 来加密,并被发送回 TCP/IP 接口 306。但是这一次, TCP/IP 接口 306 识别出该业务是标准 HTTPS 业务并将其路由到网络 318。HTTPS 业务被广泛用于诸如因特网商务等事物,且一般不被 ISP 或防火墙阻塞。在与 web 代理一起使用时,HTTPS 业务将被转发到适当的端口,如标准 HTTPS 端口 443。

[0030] 通过使用 SSTP 协议的安全隧道的数据可以包括控制业务和数据业务。以下是用于控制和数据业务的示例性命令集及其对应的分组格式。

[0031] SSTP 协议包括 2 类分组

[0032] - 控制分组 (SCP-SSTP 控制分组)

[0033] - 数据分组 (SDP-SSTP 数据分组)

[0034] 顾名思义,控制分组将是某种信道专用控制消息,而数据分组承载来自更高层的数据。

[0035] SSTP 协议具有基本头部,该头部是跨控制和数据消息共同的。

[0036] `typedef BYTE SSTP_PACKET_TYPE, *PSSTP_PACKET_TYPE;`

[0037] `#define SSTP_PACKET_TYPE_CONTROL ((BYTE) 0)`

[0038] `#define SSTP_PACKET_TYPE_DATA ((BYTE) 1)`

```

[0039] #define SSTP_VERSION_1((BYTE)0x00010000)
[0040] typedef struct_SSTP_LENGTH
[0041] {
[0042]     USHORT Reserved :4 ;
[0043]     USHORT Length :12 ;
[0044] } SSTP_LENGTH, *PSSTP_LENGTH ;
[0045] typedef struct_SSTP HEADER
[0046] {
[0047]     BYTE Version ;
[0048]     BYTE Reserved :7 ;
[0049]     BYTE ControlMessage :1 ;
[0050]     SSTP_LENGTH Length ;
[0051]     union
[0052]     {
[0053]         SSTP_CONTROL_MESSAGE ControlMessage ;
[0054]         BYTE Payload[0]
[0055]     } ;
[0056] } SSTP_HEADER, *PSSTP_HEADER ;
[0057] Version(版本)-1 字节
[0058] Control(控制)/Data(数据)-1 字节, 只有其最低有效位被使用。其余是保留
(Reserved) 位
[0059] Length(长度)-2 字节 - 限为 12 位

```

[0060]	0	8	16	20	24	32
+-----+-----+-----+-----+-----+-----+						
	版本	保留	C X X X X	长度	...	
+-----+-----+-----+-----+-----+-----+						

[0064] Length 字段是该 SSTP 分组的不包括 SSTP\_HEADER 的长度。其不能超过 4095 字节。SSTP 协议不应接受超过该限制的传输请求 (来自更高层 - 在此情况下是 PPP), 因为否则 SSTP 协议将必须处理分段。

[0065] 控制消息格式

[0066] 如上所述, SSTP 控制消息将在 SSTP\_HEADER 之后呈现, 假定 PacketType (分组类型) 是 SSTP\_PACKET\_TYPE\_CONTROL。控制消息将包括 ControlMessageType (控制消息类型) 和形成完整的控制消息的多个属性 - 长度 - 值字段。控制消息类型定义如下 :

```

[0067]     typedef enum_SSTP_CONTROL_MESSAGE_TYPE
[0068]     {
[0069]         SSTP_MESSAGE_CONNECT_REQUEST,
[0070]         SSTP_MESSAGE_CONNECT_ACK,
[0071]         SSTP_MESSAGE_CONNECT_NACK,
[0072]         SSTP_MESSAGE_CALL_CONNECTED,

```

```

[0073]      SSTP_MESSAGE_FAST_RECONNECT_REQUEST,
[0074]      SSTP_MESSAGE_FAST_RECONNECT_ACK,
[0075]      SSTP_MESSAGE_ECHO_REQUEST,
[0076]      SSTP_MESSAGE_ECHO_RESPONSE,
[0077]      SSTP_MESSAGE_DISCONNECT,
[0078]      SSTP_MESSAGE_DISCONNECT_ACK,
[0079]      SSTP_MESSAGE_ABORT_CALL
[0080] } SSTP_CONTROL_MESSAGE_TYPE, *PSSTP_CONTROL_MESSAGE
[0081] _TYPE ;
[0082]     typedef struct_SSTP_CONTROL_MESSAGE
[0083]     {
[0084]         USHORT MessageType ;
[0085]         USHORT NumAttributes ;
[0086]         BYTE Attributes[0] ;
[0087]     } SSTP_CONTROL_MESSAGE, *PSSTP_CONTROL_MESSAGE ;
[0088] 0          8          16         24         32
[0089] +---+---+---+---+---+---+---+
[0090] |    消息类型    |    属性数量    |...
[0091] +---+---+---+---+---+---+---+
[0092] typedef struct_SSTP_CONTROL_ATTRIBUTE
[0093] {
[0094]     BYTE Reserved ;// 可被用于关于属性的元数据
[0095]     BYTE AttributeId ;
[0096]     SSTP_LENGTH AttributeLength ;
[0097]     BYTE Value[0] ;// 接下来的大小为 AttributeLength 的字节
[0098] } SSTP_CONTROL_ATTRIBUTE, *PSSTP_CONTROL_ATTRIBUTE ;
[0099] 0          8          16         24         32
[0100] +---+---+---+---+---+---+---+
[0101] |    保留    |    属性 ID    |    保留    |    属性长度    |    值
[0102] +---+---+---+---+---+---+---+
[0103] typedef enum_SSTP_ATTRIBUTE
[0104] {
[0105]     SSTP_ATTRIBUTE_SUPPORT_FAST_RECONNECT,
[0106]     SSTP_ATTRIBUTE_LINK_COOKIE,
[0107]     SSTP_ATTRIBUTE_COMPLETION_STATUS,
[0108]     SSTP_ATTRIBUTE_ENCAPSULATED_PROTOCOL_ID
[0109] } SSTP_CTRL_MSG_ATTRIBUTE, *PSSTP_CTRL_MSG_ATTRIBUTE ;
[0110] SSTP_ATTRIBUTE_SUPPORT_FAST_RECONNECT
[0111] 这是无值属性且它的存在意味着客户机正在使用不可靠（如无线）介质或代理在

```

连接上有时间限制并将要求对快速重新连接 (Fast-Reconnect) 的支持。在就这一点达成一致后，在 HTTPS 介质由于某种原因而停机时，客户机和服务器都将不清除该连接上下文并向更上层指示。连接信息在被清除之前将被持久保存某一预定持续时间。如果未就这一点达成一致，则在从底层介质检测到连接丢失时，该连接上下文将被立即清除。注意，只有在底层介质经历 TCP 会话异常中止时，该连接上下文才将被维护以用于快速重新连接。

[0112] 该属性还在其中快速重新连接是不可能的实现之间提供兼容性。这将防止在不支持的场景中不必要的持久保持连接上下文。

[0113] SSTP\_ATTRIBUTE\_LINK\_COOKIE

[0114] 该属性是将被用于快速重新连接场景的可选属性。在客户机和服务器都支持快速重新连接时，服务器将向客户机提供链路 cookie 作为 SSTP\_MESSAGE\_CONNECT\_ACK 的一部分。这是客户机将需要将其作为向服务器的 SSTP\_MESSAGE\_FAST\_RECONNECT\_REQUEST 的一部分来传递的 cookie，来标识该呼叫需要相关联的上下文。

[0115] SSTP\_ATTRIBUTE\_COMPLETION\_STATUS

[0116] 这被用来指示请求的完成状态。在控制消息中这可以出现超过一次。值是 8 字节大小并具有以下结构：

```
[0117]     typedef struct_SSTP_ATTRIB_VALUE_COMPLETION_STATUS
[0118]     {
[0119]         BYTE     Reserved[3] ;
[0120]         BYTE     AttribId ;
[0121]         DWORD    Status ;
[0122]         BYTE     AttrivValue[0] ;
[0123]     }             SSTP_ATTRIB_VALUE_COMPLETION_STATUS,
[0124]     *PSSTP_ATTRIB_VALUE_COMPLETION_STATUS ;
```

[0125] 在 NAK 消息中，该属性将提供关于特定属性为何被拒绝的更多信息。

[0126] 例如，服务器可以用 AttribId(属性 Id)SSTP\_ATTRIBUTE\_SUPPORT\_FAST\_RECONNECT 和 Status(状态)ERROR\_NOT\_SUPPORTED 来响应，以指示该特征不被服务器支持。

[0127] 如果原始属性具有服务器不遵守的某一特定值，该属性将具有被拒绝的属性专用的、以 AttrivValue(属性值)开始的某一值。例如，如果客户机正在协商具有值 A、B 和 C 的 SSTP\_ATTRIBUTE\_ENCAPSULATED\_PROTOCOL\_ID，如果服务器不接受 B&C，则其将发送 2 个具有保持未被接受的协议 ID 的 USHORT 的 AttrivValue 的 COMPLETION\_STATUS 属性。如果万一被拒绝的属性值超过 64 字节，则在 NAK 消息中，值大小将被截断到 64 字节。

[0128] SSTP\_ATTRIBUTE\_ENCAPSULATED\_PROTOCOL\_ID

[0129] 这指定将通过 SSTP 封装发送的协议 id。在一给定消息中，可以有关于要支持的所有各种协议 ID 的多个这种属性。

```
[0130]     typedef enum_SSTP_ENCAPSULATED_PROTOCOL_ID
[0131]     {
[0132]         SSTP_PROTOCOL_ID_PPP = 1
[0133]     }             SSTP_ENCAPSULATED_PROTOCOL_ID,
[0134]     *PSSTP_ENCAPSULATED_PROTOCOL_ID ;
```

- [0135] SSTP\_MESSAGE\_CONNECT\_REQUEST
- [0136] 在客户机尝试与服务器建立 SSTP 会话时,这将是被发出的第一个消息。这具有以下属性 :
- [0137] SSTP\_ATTRIBUTE\_SUPPORT\_FAST\_RECONNECT[ 可任选 ]
- [0138] SSTP\_ATTRIBUTE\_ENCAPSULATED\_PROTOCOL\_ID
- [0139] 基于较早请求的结果,客户机能以关于各种属性(或不同属性集)的不同值来重新发送该信息。存在预定义数量的参数重新协商,其后连接将被异常中止。
- [0140] SSTP\_MESSAGE\_CONNECT\_ACK
- [0141] 这响应于连接请求来发送,且在服务器分配 FAST\_RECONNECT 的情况下,其将具有链路 cookie。否则,该消息将不具有任何 SSTP\_ATTRIBUTE\_LINK\_COOKIE 属性。
- [0142] SSTP\_MESSAGE\_CONNECT\_NAK
- [0143] 这响应于连接请求来发送,且其将具有服务器不接受的各属性的列表。响应于 NAK,客户机必须发出具有它想要的所有属性及它们的值的新的 CONNECT\_REQUEST。它不能只提供经调整的值。除非服务器正在确认,否则其将不存储客户机所传递的属性值。
- [0144] SSTP\_MESSAGE\_CALL\_CONNECTED
- [0145] 这将由客户机响应于 SSTP\_MESSAGE\_CONNECT\_ACK 来发送以完成与服务器的握手。这不具有与其相关联的任何属性。
- [0146] SSTP\_MESSAGE\_FAST\_RECONNECT\_REQUEST
- [0147] 这将由客户机用来进行快速重新连接(如果其已经协商过了)。这将具有表示现有链路 cookie 值的 SSTP\_ATTRIBUTE\_LINK\_COOKIE。
- [0148] SSTP\_MESSAGE\_FAST\_RECONNECT\_ACK
- [0149] 这将由服务器在成功的快速重新连接的情况下向客户机发送。如果快速重新连接未成功,则服务器将用 ABORT(异常中止) 请求使该连接异常中止。
- [0150] SSTP\_MESSAGE\_ECHO\_REQUEST
- [0151] 这是保活消息且其不具有任何相关联的属性。
- [0152] SSTP\_MESSAGE\_ECHO\_RESPONSE
- [0153] 这是响应于回送(echo)请求所发送的保活消息且其具有相关联的任何属性。如果未从远程站点接收到该响应达 3 次重复,并且没有数据业务流,则该连接将被异常中止。
- [0154] SSTP\_MESSAGE\_DISCONNECT
- [0155] 这将由客户机 / 服务器的任一个来发送以启动断开连接。在发出断开连接请求之后从服务器接收到的所有数据分组将被丢弃。这可任选地具有 SSTP\_ATTRIBUTE\_COMPLETION\_STATUS。在向远程站点发出断开连接请求后,本地站点应当等待断开连接超时或直到接收到断开连接 ACK 为止。将不进行任何重新传输。
- [0156] SSTP\_MESSAGE\_DISCONNECT\_ACK
- [0157] 在从远程站点接收到 SSTP\_MESSAGE\_DISCONNECT 后,这将由客户机或服务器发出。这将不具有任何属性。
- [0158] SSTP\_MESSAGE\_ABORT\_CALL
- [0159] 这将在基本 SSTP 协商中存在失败时发出。其可能是无法找到连接请求参数,或其可能是因为不能将快速重新连接 cookie 匹配到连接上下文而引起的。这将具有 SSTP\_

ATTRIBUTE\_COMPLETION\_STATUS 来指示失败原因。

[0160] 数据消息格式

[0161] 在 ControlMessage 位是 OFF(关闭) 时,净荷将表示所协商的协议数据。注意,如上所述,一个实施例只支持一个协议的净荷。然而,在另一实施例中,SSTP 信道协议可被用来路由不同种类协议的分组。

[0162] 图 4 是支持 SSTP 连接的一个实施例的功能块的简化的和代表性框图,其示出了关于操作的用户和内核模式的功能块的关系。该图被用来更详细地示出与 SSTP 协议相关联的控制和数据业务。

[0163] 用户模式模块 402 支持所有用户应用程序且被限制为不能直接访问硬件。内核模式模块 404 维持对所有硬件资源的控制,且是可以直接访问诸如网络接口等硬件的唯一模块。在该说明性附图中,用户模式模块是应用程序 / 套接字接口 406、远程访问连接管理器和 PPP 引擎 408(RASMAN)、SSTP 服务 410(SSTPSVC) 和 HTTP/WinHTTP 模块 412。

[0164] 内核模式模块包括作为应用程序到硬件网络协议的定义的网络驱动程序接口规范 414(NDIS) 和 HTTP/HTTPS 系统文件 416。NDIS 包括 TCP/IP 模块 418、广域网成帧模块 420、NDIS 广域网模块 422 和 SSTP 驱动程序 424(SSTPDRV)。图 4 的虚线指示跨模式连接,而实线指示模式内连接。

[0165] 在操作中,在成功建立 HTTPS 会话(例如,TCP 连接 +SSL 握手)后,诸如图 2 的第一计算机 202 等第一计算机处的 SSTPSVC 410 将与诸如图 2 的第二计算机 204 等远程站点设置会话上下文。即,在 SSL 握手完成后,SSTPSVC 410 将在 HTTPS 模块内触发上下文设置活动。在这完成后,SSTPDRV 424 将随后通过 HTTPS 会话开始 SSTP 有限状态机。在该阶段期间,只有 SSTPDRV/SSTPSVC 410 424 和 HTTPS 416 模块在交互。一旦该设置完成,将在 NDISWAN 422 和 SSTP 会话之间创建绑定。远程访问连接管理器(RASMAN)408 由 NDISWAN 通知 SSTP 会话,并通过 SSTP 连接启动 PPP 协商。PPP 有限状态机在 RASMAN 408 中(在已加载的 PPP 模块内)。PPP 控制分组将直接从 RASMAN 408 传递到 NDISWAN 422。NDISWAN 422 将将其传递到 SSTPDRV 424。SSTP 驱动程序将该分组移交给 SSTPSVC 410,而 SSTPSVC 410 将其传递到 HTTPS 模块 412。以正常的方式,HTTPS 模块 412 将该数据传递到 TCP/IP 模块 418 以通过网络路由。只具有初始头部的未完成的 PUT 请求将被发送给远程服务器。服务器将立即用 PUT 响应回复。PUT 请求延续(作为实体本体)将形成客户机到服务器数据业务且响应实体本体将是服务器到客户机数据业务。在交换头部之后,则 SSTP 响应可供使用。只有在 PPP 协商完成之后,信道才为隧道应用程序业务做好准备。

[0166] 在信道隧道(即,SSTP 会话)准备好时,数据业务可以通过该链路来承载。内核模式 TCP/IP 模块 418 接受来自用户模式中的应用程序和套接字接口 406 的数据分组形式的业务。TCP/IP 模块 418 标识分组是通过 SSTP 隧道来路由的,并将其移交给 WAN 映射模块 420。WAN 映射模块 420 将使该连接(SSTP)映射到正确的接口并将其传递到 NDISWAN 模块 422。这大致是图 3 的 PPP 模块 308 的等效方案。NDISWAN 负责 PPP 成帧和压缩。可以在该层的 PPP 模块处进行的任何加密都被关闭,因为其将是 SSL 加密的。从该点处开始,操作序列将与以上控制业务相同。即,到 RASMAN408、SSTPDRV 414、SSTPSVC 410 和 HTTPS 模块 416。一旦 SSTP/PPP 封装的数据字节到达 HTTPS 模块,则 HTTPS 模块在进行 SSL 加密之后通过 TCP 连接(默认端口 443)来发送该数据字节。所以,该分组再次从用户模式 HTTPS

模块来到 TCP/IP 模块 418,但路由将确定该业务要通过以太网接口(未示出)送出,而非像原始应用程序数据一样被送到 WAN 成帧模块 420。

[0167] SSTP 达到诸如连接数量、连接时间和带宽等与其它隧道协议相似的性能特性的目标可能要求优化来克服用户和内核模式之间的若干跨越。具体地,在穿过 SSTP 协议的若干附加模块时,可以通过避免不必要的缓冲区复制、分页、上下文切换和无序发送 / 接收来增加性能。另外,可以通过保证其它协议有公平带宽可用,即 SSTP 隧道不失控来增加性能。

[0168] 由于 SSTPSVC 410 负责发送 / 接收,所以其用于发送 / 接收的缓冲区在用户模式中也是可用的。为避免缓冲区复制以及避免与用户模式缓冲区相关联的任何分页操作,发送操作将是来自 SSTPSVC 的拉 (PULL) 操作,而接收操作将是来自 SSTPSVC 的推 (PUSH)。这将保证缓冲区可用于内核模式 SSTPDRV 424 而不引起附加的上下文切换或分页操作。上下文切换将通过使用将在用户模式线程的上下文中发生的快速 I/O 来减少。(快速 I/O 涉及绕开 I/O 子系统并直接将数据复制到输出寄存器。) 此外,为避免给定隧道的无序发送 / 接收,发送和接收可被串行化。

[0169] 在要将分组发送到远程端点时,在到达 SSTPDRV 424 时将采取以下动作。

[0170] • NDIS 分组将如其在 SSTPDRV 424 时一样排队

[0171] • 如果 SSTPDRV 424 未在进行快速 I/O(FastIO) (由发送触发器 (SendTrigger) IRP 的存在 / 缺失来表示),其只是继续对分组进行排队

[0172] • 如果发送触发器 IRP 已在 SSTPDRV 424 中,则完成 IRP 以向用户模式指示有分组要发送

[0173] • 用户模式将向其线程池排队工作项来从内核模式中拉出各字节。这有 2 个优点 - 串行化,因为每一隧道只有一个工作项,并且第二 - 随处理器的数目缩放(因为来自不同隧道的工作项可以跨处理器来最大可能限度地利用 CPU 资源)

[0174] • 在工作项执行时,其将进行快速 I/O 来取出各字节以发送,且对 HTTP 层进行缓冲区排队来异步地进行发送操作,并用新缓冲区进行取出直到快速 I/O 返回 0 字节时为止。这减少了与使用 I/O 完成端口的典型的异步完成相关联的上下文切换,并且内核处的缓冲区要求(对于异步缓冲区)不再是必要条件。

[0175] • 如果达到 MaxNumWorkerIterations(最大量工作重复),即使快速 I/O 正在进行,也要重新调度工作项以便下一可用工作项被执行。这将允许公平的带宽可用性。

[0176] • 如果快速 I/O 返回要发送 0 字节,则发送触发器 IRP 将被再次排队。

[0177] 在接收数据时,HTTP 层将向 SSTPSRV 410 指示所接收到的字节(通过 I/O 完成端口或其它异步回叫机制)。以下操作序列可以发生:

[0178] • 在从 HTTP 层 412 获得分组后,所接收到的缓冲区被在服务处排队

[0179] • 如果要处理接收到的字节的工作项已在进行,则不需做什么事。否则,将该工作项排队。

[0180] • 在该工作项中,继续进行快速 I/O 请求来指示所接收到的字节。SSTPDRV 424 将自己应付对从快速 I/O 例程接收到的字节的处理。这提供串行化以及减少了上下文切换。

[0181] • 所接收到的帧将由 NDIS 层描绘并复制以呈现给更上层驱动程序层

[0182] • 如果指示给小型端口(miniport)的缓冲区的数量超过 MaxNumWorkerIterations,则重新排队该工作项。这又是为了避免失控隧道,在失控隧道

中,一个隧道上的高数据传输降低带宽可用性或另一隧道可用的处理时间。

[0183] 另外,TCP Nagle,即低级分组缓冲将被关闭,以允许立即发送和接收操作。缓冲区大小可被选为足够大,以便TCP连接被利用到可能的最高速率。

[0184] 尽管上文阐明了众多不同实施例的详细描述,但是应当理解,本专利的法律范围由本专利所附的权利要求书的言辞来限定。该详细描述应被解释为仅是示例性的,且不描述本发明的每一可能的实施例,因为描述每一可能的实施例即使不是不可能的也是不切实际的。可使用现有技术或在本专利提交日之后开发的技术来实现众多替换实施例,这仍将落入定义本发明的权利要求书的范围之内。

[0185] 由此,可在此处所描述和示出的技术和结构上作出许多修改和变化而不脱离本发明的精神和范围。因此,应当理解,此处所描述的方法和装置仅是说明性的,且不限制本发明的范围。

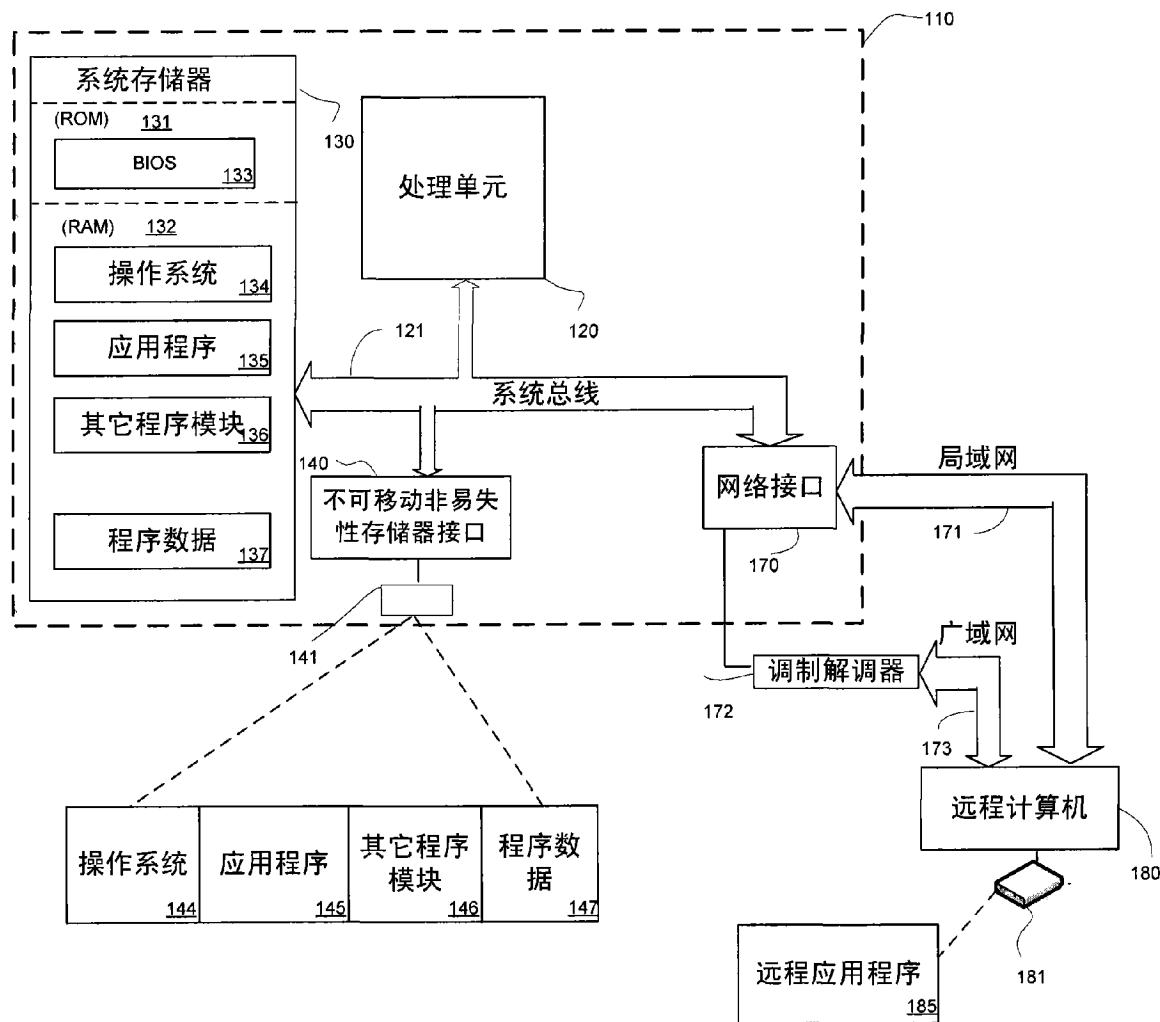


图 1

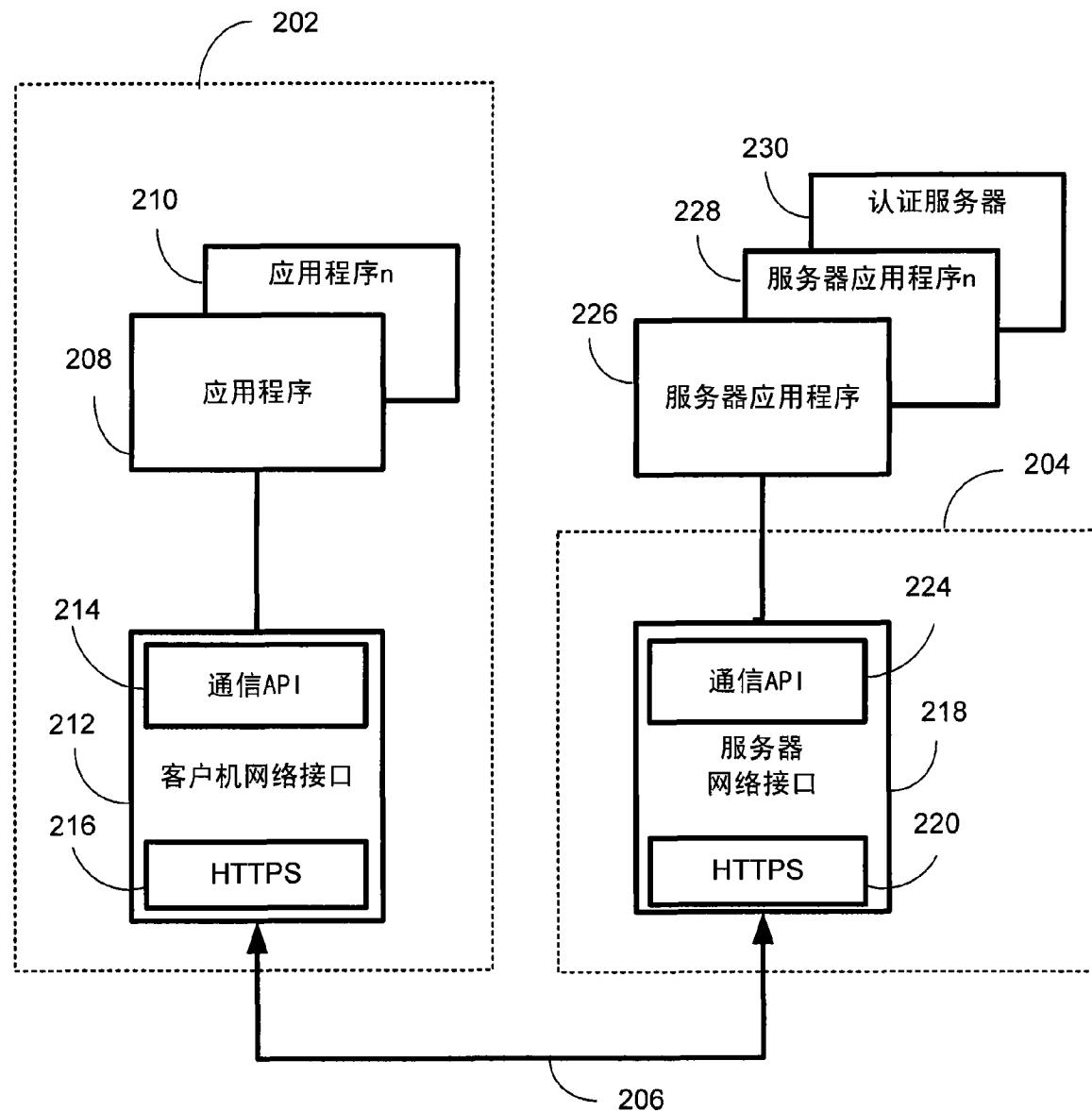


图 2

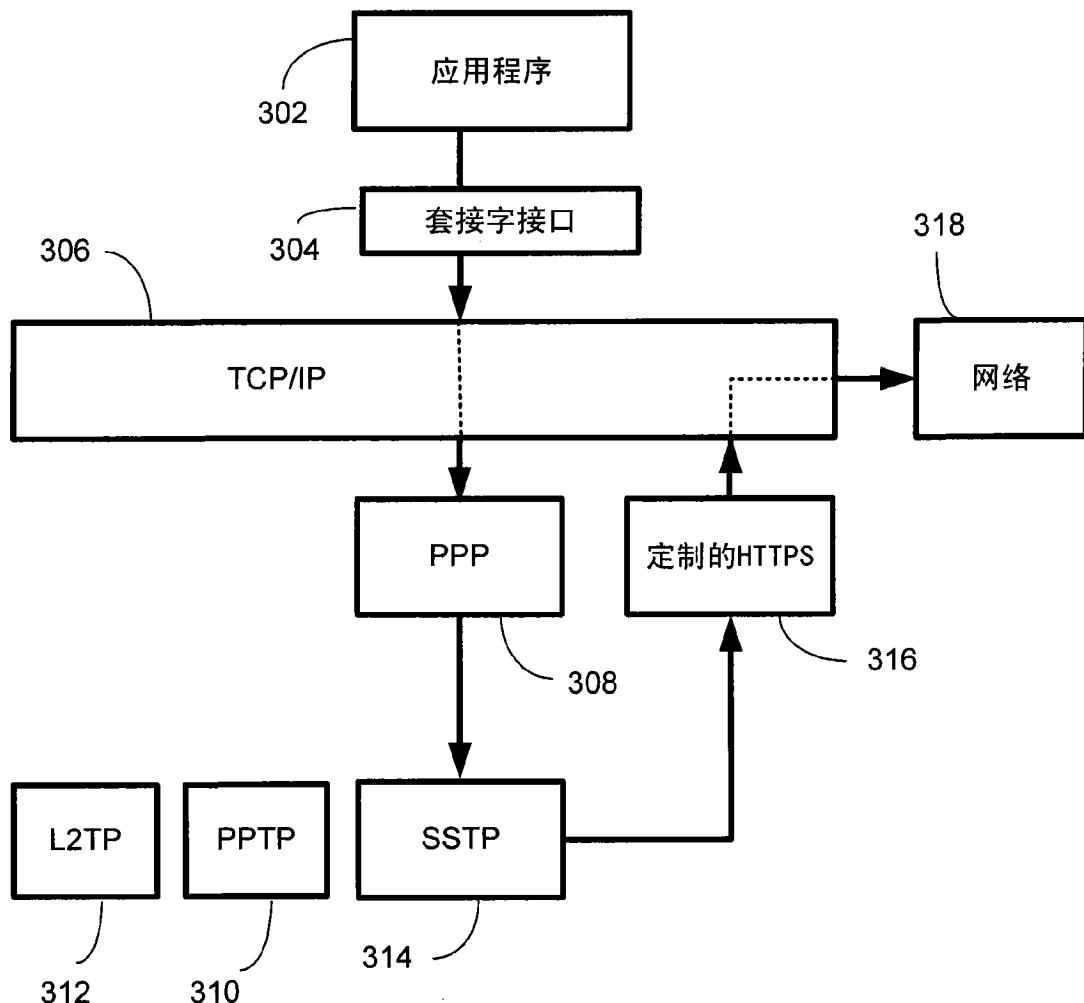


图 3

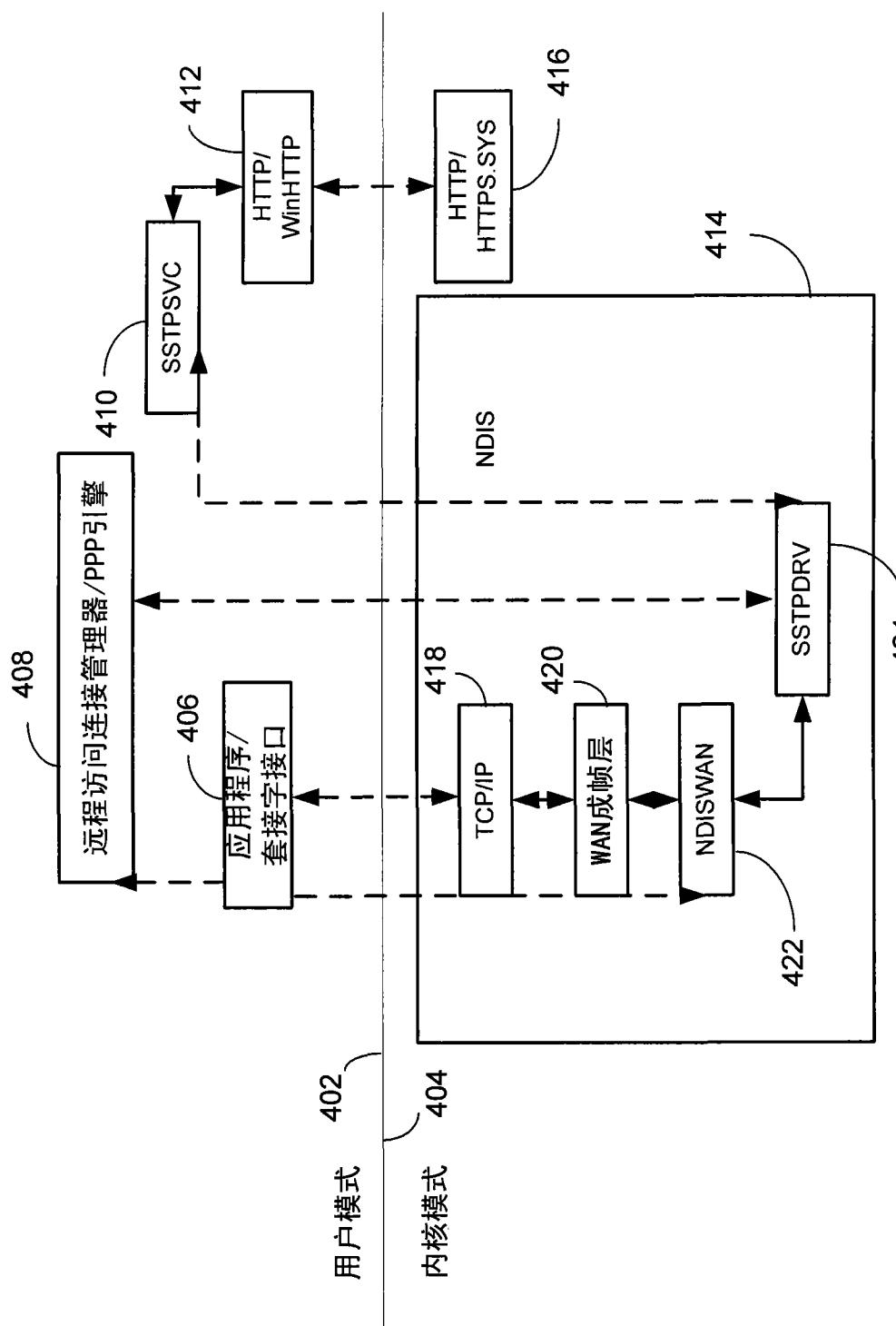


图 4