



[12] 发明专利申请公布说明书

[21] 申请号 200910008858.1

[43] 公开日 2009 年 11 月 4 日

[11] 公开号 CN 101572700A

[22] 申请日 2009.2.10

[21] 申请号 200910008858.1

[71] 申请人 中科正阳信息安全技术有限公司

地址 100080 北京市海淀区中关村大厦 19 号
新中关大厦 B 座北翼 16 层

[72] 发明人 翟征德 魏冰

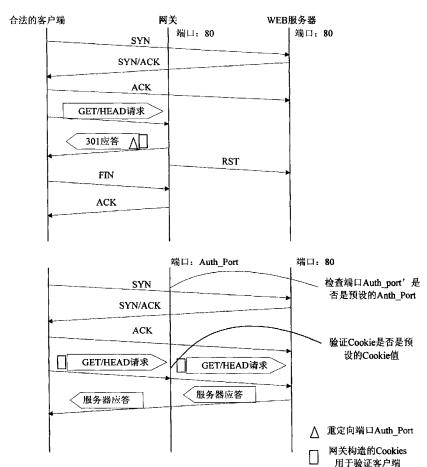
权利要求书 1 页 说明书 6 页 附图 6 页

[54] 发明名称

一种 HTTP Flood 分布式拒绝服务攻击防御方法

[57] 摘要

本发明公开了一种保护 Web 服务器免受 HTTP Flood 分布式拒绝服务攻击的方法和装置。该方法部署在网关设备上，网关设备位于客户端与 Web 服务器之间。方法实现于网络层，对所有进出网络的数据包进行 IP 分片重组和 TCP 流重组，并进行 HTTP 请求的解析。使用基于 Cookie 和端口重定向的机制鉴别合法请求和恶意请求，验证成功的合法请求会自动地经由网关转发至服务器，而验证失败的恶意请求会被网关过滤。该方法实现了对客户端的透明性和服务端低的资源消耗，能有效地识别并过滤恶意请求，达到良好的防御效果。



1. 一种网关保护 Web 服务器免受 HTTP Flood 分布式拒绝服务攻击的防御方法，所述的网关设置在 Web 服务器所在的网络边缘，所述方法包括步骤：

客户端连接 Web 服务器的开放端口，建立三次握手，网关接收客户端向服务器发送的 HTTP 请求。

网关代替 Web 服务器向客户端发送 301 应答。为该客户端生成全局唯一的 Cookie 和端口值；使用新生成的端口号代替开放端口，构造新的 URL，将 Cookie 值和新 URL 在 301 应答头部返回给客户端。301 应答的头部设置 Connection 字段值为 close，通知客户端关闭本次连接。

网关接收客户端发送的带 FIN 标志的 TCP 包，该数据包是客户端用于通知服务器关闭本次连接的，网关作为中间人以服务器名义向客户端发送带 ACK 标志的 TCP 包响应连接的关闭。

网关以客户端名义向服务器发送带 RST 标志的 TCP 包，通知服务器关闭连接。

客户端连接 Web 服务器的非开放端口。网关接收建立三次握手的 SYN 数据包，验证目的端口值是否为预期的值，若是，对数据包进行端口转发，允许 TCP 连接的建立；若不是，丢弃 SYN 数据包。

在非开放端口建立连接后，网关接收客户端向服务器发送的 HTTP 请求，网关验证 HTTP 请求头部是否包含预期的 Cookie 值，若包含，将 HTTP 请求转发给 Web 服务器；若不包含，将 HTTP 请求丢弃，并以客户端名义向服务器发送带 RST 标志的 TCP 包，通知服务器关闭连接。

2. 如权利要求 1 所述的方法，其特征在于网关位于客户端与 Web 服务器之间，包括对所有经过的 IP 报文进行 IP 分片重组的装置。

3. 如权利要求 1 所述的方法，其特征在于网关包括对所有经过的 TCP 数据包进行流重组的装置。

4. 如权利要求 1 所述的方法，其特征在于网关包括识别 HTTP 请求，并对请求头部进行解析的装置。

5. 如权利要求 1 所述的方法，其特征在于网关包括代替 Web 服务器对客户端 HTTP 请求进行应答的装置。

6. 如权利要求 1 所述的方法，其特征在于网关包括获取正确的 TCP 头部序列号、确认号，代替 Web 服务器对客户端发送的带 FIN 标志 TCP 包进行应答的装置。

7. 如权利要求 1 所述的方法，其特征在于网关包括代替客户端对服务器发送带 RST 标志 TCP 包的装置。

8. 如权利要求 1 所述的方法，其特征在于网关包括验证 TCP 连接目的端口是否合法的装置，包含对所有客户端端口值进行生成、维护、更新的装置。

9. 如权利要求 1 所述的方法，其特征在于网关包括对数据包进行端口转发的装置。

10.如权利要求 1 所述的方法，其特征在于网关包括对每个客户端的 Cookie 进行生成、维护、更新的装置；包含检查 Cookie 是否有效的装置。

一种 HTTP Flood 分布式拒绝服务攻击防御方法

技术领域

本发明一般地涉及拒绝服务攻击防御领域。更具体的，本发明涉及一种保护 Web 服务器免受 HTTP Flood 分布式拒绝服务攻击的方法和装置。

背景技术

分布式拒绝服攻击（DDos: Distributed Deny of Service）由于其潜在的破坏性强，而且难于防止和追查，越来越成为常见的攻击方式，严重地威胁着网络的可用性。最常见的 DDos 攻击，就是攻击者在短时间内，通过分布在世界各地的计算机产生大规模的垃圾数据或者非法请求，淹没目标服务器或者目标网络。DDos 为了躲避检测和过滤，自身进行不断地演变，由早期的 SYN Flood 、 UDP Flood、 Ping of Death 等单纯的流量型攻击转变为针对应用层的攻击。应用层的拒绝服务攻击中比较显著的一种是面向 Web 服务器的 HTTP Flood 攻击。攻击者往往针对 Web 服务的昂贵操作，以此大量消耗目标服务器的 CPU、数据库和磁盘资源。例如，访问 Web 服务器的某些特定 URL，服务器为了响应页面请求，必须进行大量的数据库检索、查询操作，以此大量消耗 Web 服务器的主机资源，使得正常用户的请求被拒绝。面向 Web 的 HTTP Flood 攻击不具备明显特征，能很好地伪装成突发流量，利用传统的技术难以防御，并且攻击者往往只需付出相对较小的代价，就能取得明显的攻击效果。

目前，存在一些技术和方法用于 HTTP Flood 攻击的检测和防御，但是其能力还十分有限。麻省理工大学的 Srikanth Kandula 设计和实现了 Kill-Bots，该系统采用图形测试识别非法请求，对其进行过滤，但是 Kill-Bots 是对 Web 服务器内核的扩展，只适用于中小型站点。西班牙 Carlos III 大学的 Juan M.E-T 提出采用 Markovian 模型对 Http 负载进行分析，建立正常流量的 Markovian 模型，并计算实时流量与正常流量的偏离度。Markovian 模型参数的选取会极大地影响检测率和误警率，但在实际应用环境中参数的确定比较困难。

发明内容

本发明提供了一种保护 Web 服务器免受 HTTP Flood 分布式拒绝服务攻击的方法和装置。该方法部署在网关设备上，网关设备位于网络边缘，用于保护网络内所有的 Web 服务器。网关设备位于客户端与 Web 服务器之间，检测 HTTP 的 TCP 连接状态，使用基于 Cookie 和端口重定向的机制鉴别合法请求和非法请求。合法请求会自动地经由网关转发至服务器，而非法请求连接会被网关关闭。

Cookie 是 Web 服务器向用户的浏览器发送的一段 ASCII 码文本。一旦收到 Cookie，浏览器会把 Cookie

请求追踪客户并保持会话状态。Cookie 保存在 HTTP 协议请求或者应答头部在服务器与客户端之间传递。

通用的浏览器均能对 Cookie 进行解析和处理，而大多数攻击工具或者网络蠕虫不具备识别和处理 Cookie 的能力。利用这一点，网关能够鉴别大多数合法请求和恶意请求。

同时，为了增强网关抵抗攻击的鲁棒性，将端口重定向机制与基于 Cookie 的方法结合使用。Web 服务的开放端口为固定端口（通常为 80），网关收到来自客户端未经认证的 HTTP 请求时，会将请求重定向到网关随机生成的端口上。客户端只有正确处理了重定向语义，才会被网关识别为合法用户。

网关监控客户端到服务的所有 TCP 连接，对连接进行 IP 分片重组和流重组。客户端到服务器的请求先经过网关，网关为了鉴别该请求来自合法的用户或是攻击工具会代替服务器对请求进行响应。网关利用 HTTP 协议的 301 响应码，响应内容如附图 1 所示。301 响应码表示请求的数据已永久性地移动到新的位置。网关在 Location 头字段填入新的 URL，改变 URL 中的 Port 值为 Auth_Port，Path 部分保持不变。为了便于处理，Auth_Port 值在 1024 至 65536 之间随机取值，不能等于 80。同时在 Set-Cookie 头字段指定由网关随机生成的 Cookie 值。Connection 头字段赋值为 close，表示关闭本连接，客户端需要重新建立连接发送 HTTP 请求。网关构造的响应包 IP 源地址置为 Web 服务器的地址，所以从客户端的角度，其收到的所有数据包都是来自服务器的，网关对用户来讲是透明的。

合法用户收到 301 响应，浏览器会自动对响应进行处理。客户端做两个操作：

1. 向 Web 服务器发送带 FIN 标志位的 TCP 包关闭本次连接。网关收到 FIN 数据包后，代替 Web 服务器给客户端发送 ACK 包。网关将 ACK 包中的源 IP 地址设为服务器地址，并填入合适的 TCP 头字段，包括序列号、确认号、校验和等。
2. 通过三次握手建立新的连接，并对 301 应答中的 URL 发起请求，在请求包中使用 Cookie 头字段，字段的值与应答包中 Set-Cookie 字段的值相同。

由于客户端再次发出请求前会关闭原先的连接，且网关作为中间人代替 Web 服务器响应了连接的关闭，Web 服务器并不知道。因此，网关需要向 Web 服务器发送带 RST 标志的包，通知连接的关闭，该 RST 包的 IP 源地址设为客户端的 IP 地址。

客户端向 Web 服务器建立新的连接，网关收到新建连接的 SYN 包，检查目的端口值是否是之前反馈给客户端的端口值，若不是，直接将该数据包丢弃；若是，将数据包转发至服务器的 80 端口，完成三次握手。此后，该连接上所有的数据包都需要经过网关进行端口转发。

连接建立后，网关收到用户发送的 HTTP 请求，检查 HTTP 包 Cookie 头字段是否包含原先反馈给客户端的 Cookie 值。若 Cookie 值符合要求，表明本次请求是合法的，将请求转发给 Web 服务器。否则，丢弃该请求，并向服务器发送带 RST 标志的包，通知服务器将连接关闭。因网关作为中间人对服务器透明，RST 包 IP 源地址设置为客户端的 IP 地址。

通常攻击工具或者网络蠕虫发起分布式拒绝服务攻击时，只是向 Web 服务器发送大量的 HTTP 请求，并不对服务器的响应进行处理。当网关返回 301 应答并在应答中插入 Cookie 时，攻击工具或网络蠕虫并不能正确地解析应答的语义，体现为不能在预设的端口上进行重定向，并且 HTTP 请求中不会包含预设的 Cookie 值。网关利用这一点能有效地区分合法请求和攻击请求。攻击请求数据包不会被网关转发给 Web 服务器，从而避免了对服务器资源的消耗。

附图 2 表示了合法用户从发出请求到接收到 Web 服务器响应，与网关和 Web 服务器的交互过程。

附图 3 表示了攻击工具或网络蠕虫对 Web 服务器发起的攻击请求被网关截断的交互过程。

网关的实现基于定制的硬件和操作系统，采用专为防火墙设计的硬件平台和裁减的 Liunx 操作系统，上述算法的实现基于 Liunx Netfilter 框架。Netfilter 是一种 Liunx 内核中用于扩展各种网络服务的结构化底层框架，它为 IPv4、IPv6 定义了一套钩子函数，钩子函数在数据报流过协议栈的几个关键点被调用。Netfilter 提供了 5 个钩子函数挂载点，分别是 NF_IP_PRE_ROUTING、NF_IP_FORWARD、NF_IP_LOCAL_IN、NF_IP_LOCAL_OUT、NF_IP_POST_ROUTING。内核模块可以对每种协议的一个或多个钩子进行注册，实现挂接。当某个数据包被传递给 Netfilter 框架时，内核检测是否有模块对该协议的钩子函数进行了注册，若注册了，则调用该模块注册时使用的回调函数。在回调函数中实现用户自定义的功能。

本发明提出的算法通过挂接在 NF_IP_PRE_ROUTING 的钩子函数实现，但不局限于 NF_IP_PRE_ROUTING 挂接点。本算法按照逻辑功能可划分为 4 个模块：流重组模块、端口验证模块、HTTP 请求验证模块、端口转发模块。附图 5 表示了数据包流经网关的过程和各逻辑模块间的关系。

流重组模块使用 Hash 表存储正在建立连接和已经连接的 TCP 流，如附图 4 所示。Hash 表由数组构成，数组中的每一个元素指向 `tcp_stream` 结构的双向链表，一个 `tcp_stream` 结构对应一个 TCP 连接。操作 Hash 表的关键数据结构是 `tuple`，`tuple` 是源地址、目的地址、源端口号、目的端口号的 4 元组。网关接收一个新的数据包时，首先调用 Linux 系统函数 `ip_defrag` 进行 IP 分片重组。重组完成后，对 `tuple` 做 Hash 运算得到对应的 Hash 表数组下标，遍历该数组元素指向的 `tcp_stream` 结构双向链表，查找数据包是否属于网关已经维护的一个 TCP 连接。若没有找到，对 `tuple` 做逆运算，即调换源地址与目的地址、源端口号与目的端口号得到 `reverse_tuple`，使用 `reverse_tuple` 重复上述的 Hash 表查找过程。

`tcp_stream` 结构描述了一个 TCP 连接的所有信息，关键的变量包括 2 个 `half_stream` 结构分别描述客户端、服务器信息；一个 `verifier` 结构封装 HTTP 请求验证函数；一个 `reassem_stop` 标志位标识 TCP 重组是否继续进行；一个 `data` 缓冲区保存已排序的应用层数据。当系统收到一个新的 SYN 数据包，且其不属于系统维护的任何一个 TCP 连接时，表明它为 TCP 三次握手的 SYN 包。调用端口验证模块，检查其目的端口否为保护服务的开放端口或是重定向 `Auth_Port` 端口，若是，则为该 TCP 连接初始化一个新的 `tcp_stream` 结构加入 Hash 表，否则，直接丢弃 SYN 包。若目的端口是预期的 `Auth_Port` 端口，则调用端口转发模块

将其转发至服务器的 80 端口。初始化 `tcp_stream` 结构时注册 `verifier` 结构，TCP 重组的适当时候会根据 `verifier` 调用相应的 HTTP 请求验证函数。使用 `verifier` 结构的好处是使应用层的检查模块化，当需要添加新的验证逻辑时，只需在 `verifier` 结构中注册新的验证函数。

本算法只关心客户端发往服务器的应用层数据，`tcp_stream` 结构的 `data` 缓冲区中存放已排序的应用层数据，在客户端的 `half_stream` 结构里维护失序的 TCP 包链表。当有新的 TCP 包到来，判断 TCP 重组条件：若未完整获取第一个 HTTP 请求的头部数据，并且已经重组的数据包长度或数据包个数未达到上限，则进行 TCP 重组；否则，停止重组，直接进行数据包转发。进行 TCP 重组时，检查序列号，将循序到达的包加入 `data` 缓冲区，作为 HTTP 请求验证函数的输入，失序的数据包存入 `half_stream` 的 TCP 包链表，供后续 TCP 重组使用。

每当有新的数据加入 `data` 缓冲区，会调用 HTTP 请求验证模块。验证模块判断 `data` 缓冲区是否已包含完整的 HTTP 头部数据，若是，进行 `Cookie` 值合法性检查；否则，终止本次检查，等待完整的 HTTP 头部数据到达。HTTP 协议 1.1 版本默认采取持久连接，即在一个 TCP 连接中可进行多次 HTTP 请求和应答。持久连接中只需对第一个 HTTP 请求进行检查。那么完整获取第一个 HTTP 请求的头字段后，将 `reassem_top` 标志位置位，表示该 TCP 连接的后续数据无需重组，后续的数据包根据检查结果直接转发或丢弃。HTTP 协议 1.0 及之前版本不支持持久连接，一个 TCP 连接中只可发送一次 HTTP 请求。网关对其的处理与 1.1 版本类似，不再累述。

端口验证模块在网关收到 TCP 三次握手的 SYN 包时被调用。若目的端口为 80，表示即将建立的连接不是重定向连接，为该连接初始化 `tcp_stream` 结构。若目的端口非 80，表明即将建立的连接是经过网关重定向的连接，检查端口号是否与源 IP 地址相对应，若对应，为该连接初始化 `tcp_stream` 结构，将该数据包交由端口转发模块进行转发；若不对应，将该数据包丢弃。

HTTP 请求验证模块在获取 TCP 连接中第一个完整的 HTTP 请求头部时被调用。该模块检查请求头中的 `Cookie` 字段，按照数据包内容的不同，可能出现以下 4 种情况：

1. HTTP 请求头中没有 `Cookie` 字段，且 TCP 目的端口为 80，则网关给客户端发送 301 应答，在 301 应答中填入新的 URL 地址，新 URL 中填入为该客户端随机生成的端口值 `Auth_Port`。在 301 应答头部加入 `Set-Cookie` 字段，填入随机生成的 `Cookie` 值。端口号、`Cookie` 值、及对应的客户端 IP 地址会被记录下来，供之后的验证使用。同时，301 应答头部 `Connection` 字段设为 `close`，表示关闭本连接。因为网关对客户端透明，301 应答的源 IP 地址设为 Web 服务器的地址，从客户的角度，该应答是从 Web 服务器发出的。同时，网关需要给 Web 服务器发送带 RST 标志的 TCP 包，通知服务器将本连接关闭。同样 RST 包的源地址设为客户端的 IP 地址。

2. HTTP 请求头中没有 `Cookie` 字段，且 TCP 目的端口非 80。因为只有经过网关重定向的数据包才

会发往非 80 端口，但此时请求头中没有 Cookie 字段，表明该请求为非法请求。网关将数据包直接丢弃，并向 Web 服务器发送带 RST 标志位的包，通知 Web 服务器将已建立的连接关闭，减小 Web 服务器的资源消耗。

3. HTTP 请求头中包含 Cookie 字段，且 TCP 目的端口为 80。因目的端口为 80，表明该连接未经过网关重定向，请求头中的 Cookie 为服务器与客户之间的约定，并不是由网关指定。类似于 1 中的情况，网关代替 Web 服务器给客户端发送 301 应答，应答中填入新的端口号和 Cookie 值。此时由网关随机生成的 Cookie 附在源 Cookie 值的后面，不会破坏原 Cookie 的语义。同时，网关需要给 Web 服务器发送带 RST 标志的 TCP 包，通知服务器将本连接关闭。

4. HTTP 请求头中包含 Cookie 字段，且 TCP 目的端口非 80。因目的端口非 80，表明该连接经过了网关重定向，此时需进一步检查 Cookie 值的合法性。根据源 IP 地址查找网关为该客户端生成的 Cookie 值，检查存储的 Cookie 值是否是当前请求包中 Cookie 值的子串，若是，表明 HTTP 请求合法，调用端口转发模块进行转发；否则，将该请求丢弃，并向 Web 服务器发送带 RST 标志位的包，通知 Web 服务器将已建立的连接关闭。

上述 1 和 3 的情况下，网关在 301 应答中置 Connection 头字段值为 close，通知客户端关闭连接。随后，网关收到来自客户端的带 FIN 标志的 TCP 数据包，网关需要代替 Web 服务器对该 FIN 包进行 ACK 回应。

网关对端口号和 Cookie 值的生成、维护、更新描述如下。端口号和 Cookie 值根据客户端源 IP 地址组织为一棵二叉排序树，如附图 6 所示。树中每个结点保存了 IP 源地址及该 IP 地址对应的存储结构。每个存储结构包括端口号、Cookie 值、时间信息。端口号和 Cookie 值均具有时效性，存在时间超过一定值的端口号和 Cookie 值会失效。二叉排序树中，若根结点的左子树非空，则左子树中所有结点的 IP 源地址小于根结点的 IP 源地址值；若根结点的右子树非空，则右子树中所有结点的 IP 源地址大于根结点的 IP 源地址值。左、右子树又各是一棵二叉排序树。网关生成重定向应答时，随机生成端口号和 Cookie 值，并记录下生成时间，根据客户端的源 IP 地址，在二叉树的适当位置插入存储结构。网关收到新建连接的 SYN 包或者获取一个完整的 HTTP 头部数据时，查找二叉排序树，判断目的端口号或者 Cookie 值是否符合，且生成时间未超过时延，根据查找的结果进行后续的操作。TCP 连接关闭时，删除源 IP 地址对应的结点。

附图说明

图 1 是网关利用 HTTP 协议 301 响应码进行应答的示意图。

图 2 是合法用户从发出请求到接收 Web 服务器响应与网关、Web 服务器交互过程的示意图。

图 3 是攻击工具或网络蠕虫对 Web 服务器发起攻击被网关截断的交互过程示意图。

图 4 是实现 TCP 连接跟踪的数据结构示意图。

图 5 是网络数据包流经网关防御逻辑模块的示意图。

图 6 是网关维护端口号和 Cookies 值的数据结构示意图。

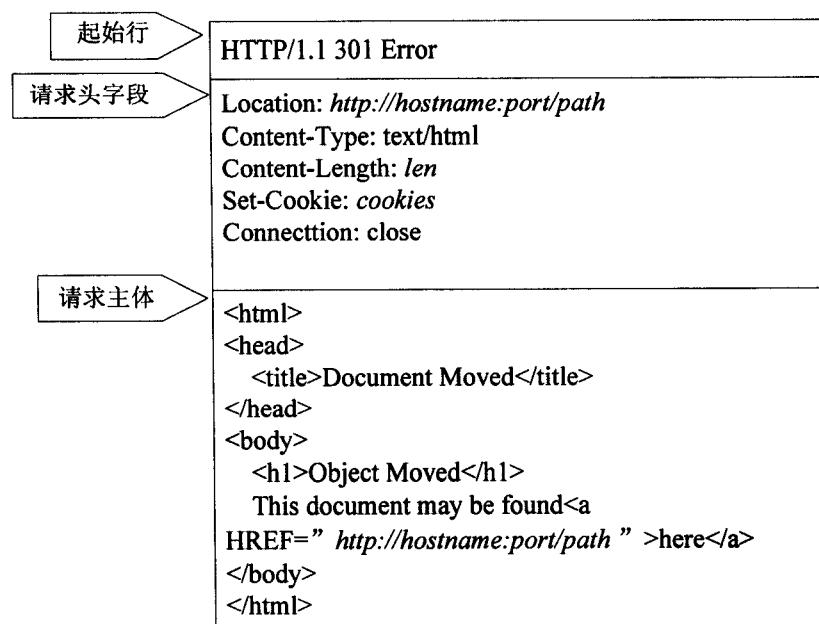


图 1

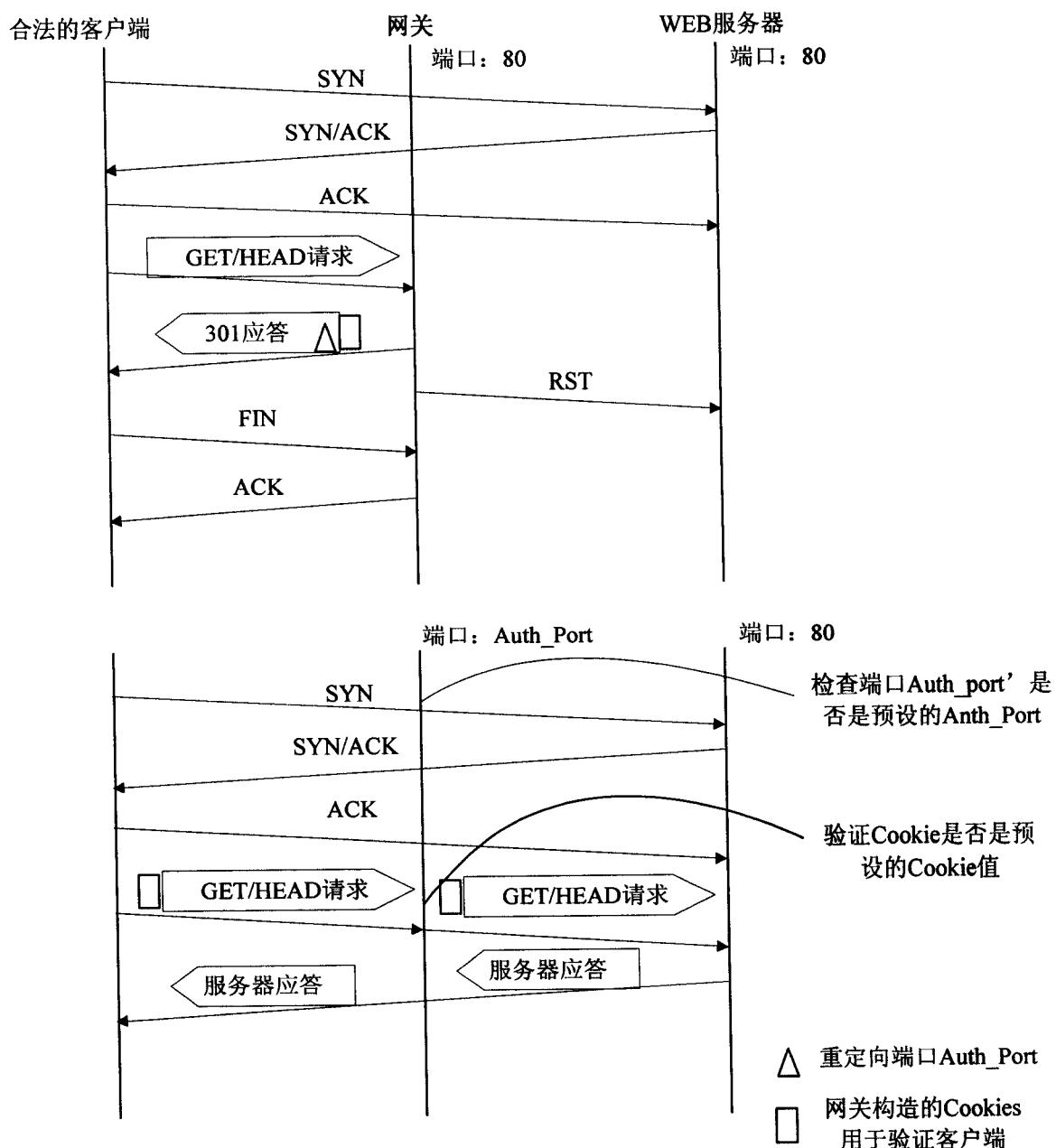


图 2

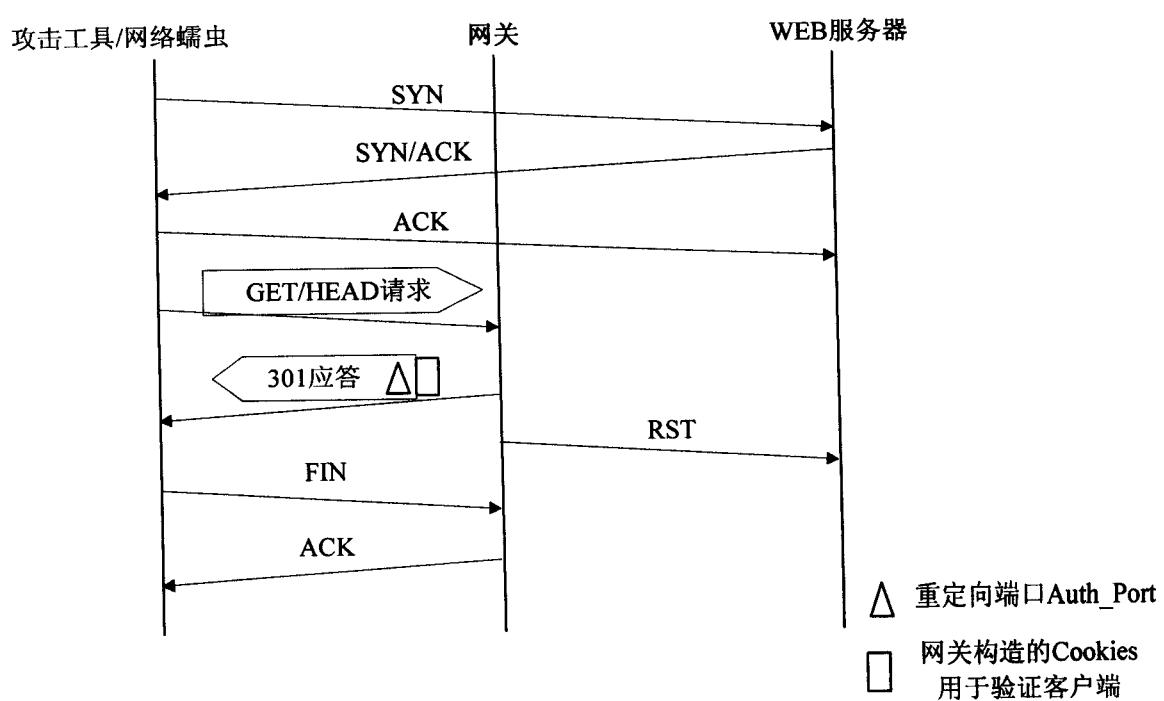


图 3

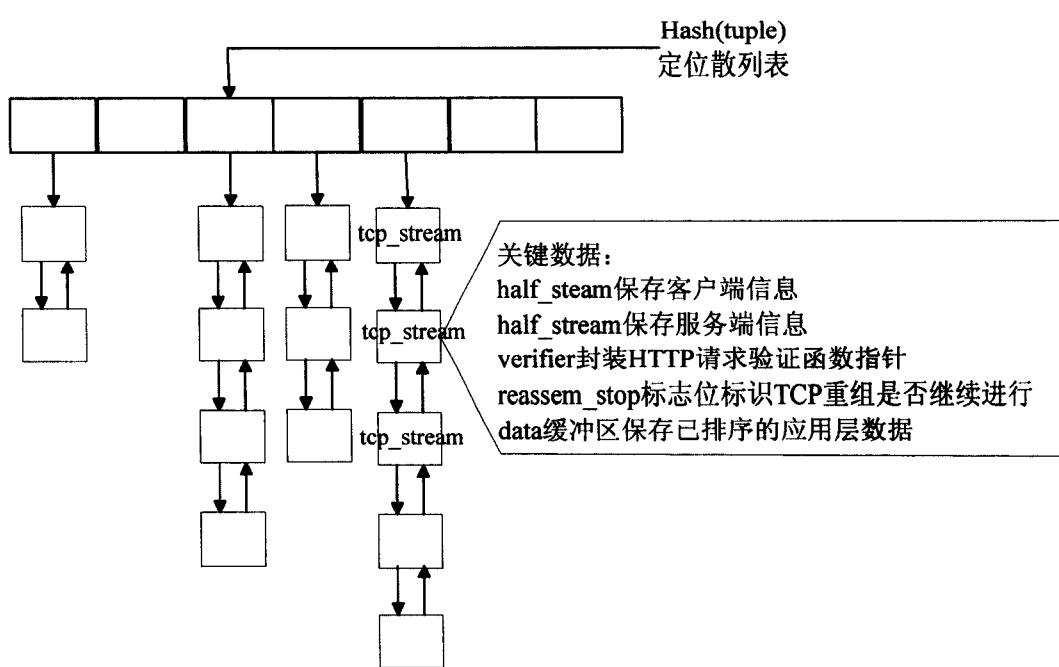


图 4

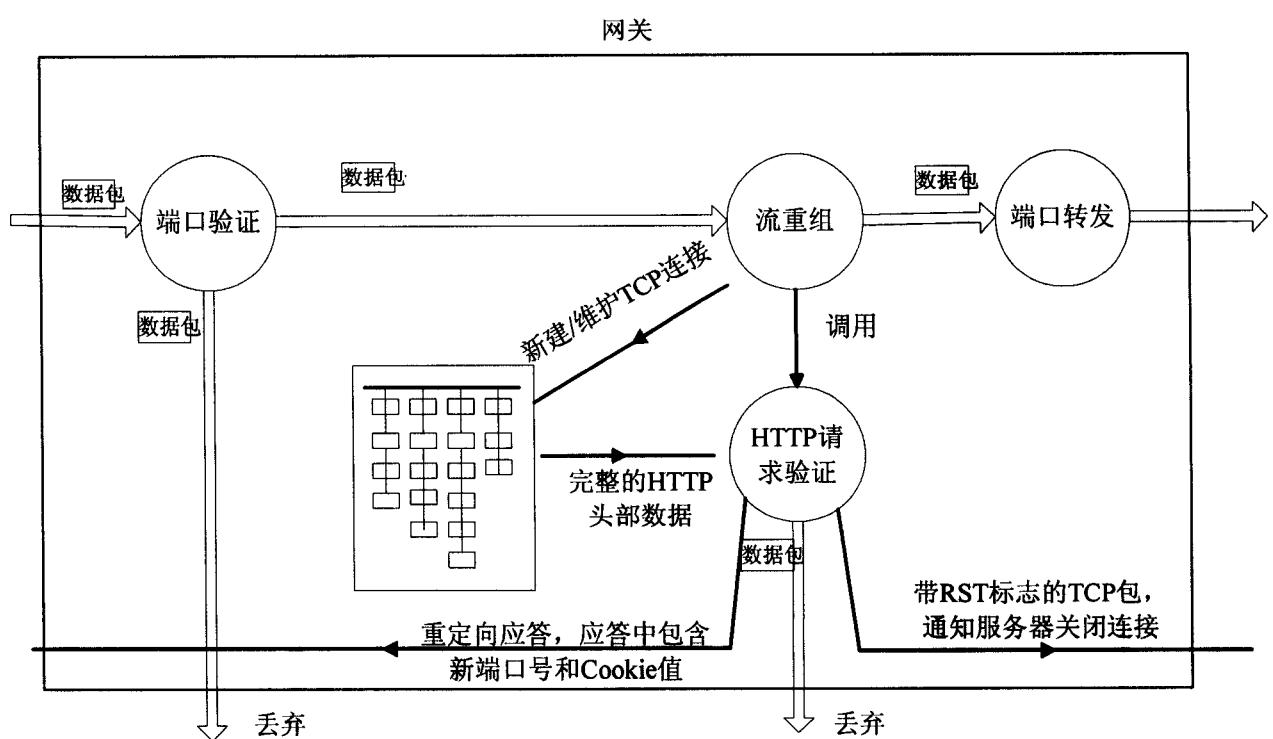


图 5

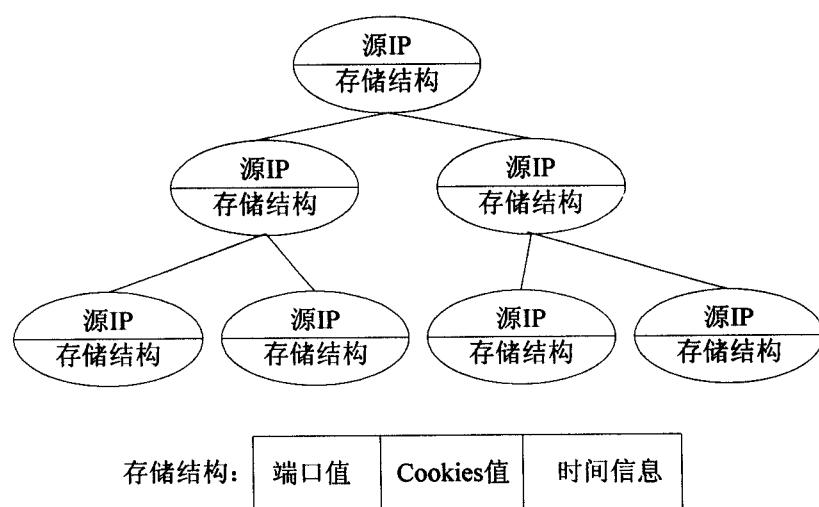


图 6