第34卷 第14期 计 算 机 工 程 2008年7月
Vol.34 No.14 Computer Engineering July 2008
·网络与通信· 文章编号：1000—3428(2008)14—0129—03 文献标识码：A 中图分类号：TP393.08

基于虚拟以太网的VPN系统

田权斌，李立新，周雁舟
(解放军信息工程大学电子技术学院，郑州 450004)

摘 要：研究基于虚拟以太网的VPN系统，该系统通过在内核构建虚拟网卡实现核心态和用户态的交互，并虚拟了一个以太网的工作环境，在用户态构建安全隧道，实现基于数字证书系统的身份认证、传输加密、数据完整性验证和抗重放攻击等功能。测试表明，在实现VPN功能的基础上，该系统在穿越网络设备、支持多种协议等方面具有良好的性能。

关键词：虚拟以太网；VPN系统；隧道

VPN System Based on Virtual Ethernet

TIAN Quan-bin, LI Li-xin, ZHOU Yan-zhou
(Institute of Electronic Technology, PLA Information Engineering University, Zhengzhou 450004)

【Abstract】This paper studies the virtual Ethernet technology, and develops a VPN system. The system transports packets between kernel space and user space, and simulates a virtual Ethernet environment by constructing virtual NIC in kernel. It realizes the functions of identity authentication based on digital certificate system, traffic encryption, integrality validating and resistance to replay attack by constructing secure tunnel in user space. Test results demonstrate that, besides the primary function of VPN, the system has good performance on passing through net devices and can support multiple net protocols, etc.

【Key words】 virtual Ethernet; VPN system; tunnel

1 概述

在开放式网络中，信息传输的私密性和安全性得不到很好的保障。在这种情况下，虚拟专用网(VPN)方案是一种较好的解决方法。目前市场上主流的VPN是IPSec VPN和SSL VPN, IPSec VPN对处于操作系统核心态的TCP/IP协议栈的IP协议进行了改造，在IP层构造安全隧道。这种模式在穿越网络设备、使用方便性等方面存在一定的局限。近年来，SSL VPN发展非常迅速，SSL VPN对基于Web的应用性能非常优越，但一般的SSL VPN主要使用代理技术，对UDP应用、其他非Web的应用及支持不够。基于虚拟以太网的组网方式采用一种新的隧道构造方式，在操作系统内核完全保留了以太网帧的内容和格式，通过在用户态构建隧道，对内核中的数据包重新封装，避免了IPSec VPN隧道封装的问题，可以很好地支持各种网络协议和应用。本文在介绍虚拟以太网技术的同时，在Linux平台上实现了一个简单的VPN系统。

2 基于虚拟以太网的VPN系统设计

2.1 虚拟以太网原理

虚拟以太网技术通过软件模拟真实网卡，提供数据包在核心态和用户态的传递接口[1]，实现经TCP/IP协议栈封装后的数据包在核心态和用户态之间传递；通过在用户态构建安全隧道的方式模拟一个以太网环境的一种软件VPN组网方式。虚拟以太网技术的核心在于实现一个虚拟网卡，一方面可以与核心态IP协议栈进行交互，对用户表现为具有真实网卡的功能；另一方面，具有核心态和用户态的接口功能，可以将完整的以太包在核心态和用户态之间传递。在构造VPN隧道时，不同于IPSec VPN直接修改IP包，虚拟以太网技术将虚拟隧道的负载由IP包改为完整的以太网帧，避免了改变原协议栈中数据包格式，因此，完全屏蔽了各种复杂网络环境对VPN隧道建立的影响，可以对各种网络设备具有很好的穿透性；同时通信隧道改为应用层建立，完全保留了以太网环境，可以支持以太网所能支持的各种网络协议。

2.2 系统组网结构

系统采用如图1所示拓扑结构，由服务器和客户端组成。服务器用于启动守护进程，等待客户端的连接，并对客户端的数据包进行转发，客户端发起连接，从而建立安全连接。VPN连接建立后，服务器和所有客户端如同工作于一个局域网之中。

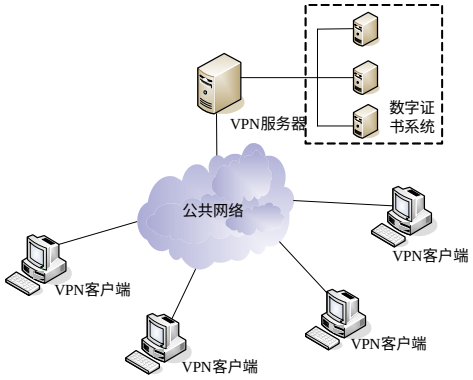

图1 虚拟以太网VPN网络拓扑结构

2.3 系统结构设计

服务器和客户端的系统结构相类似，主要包括虚拟网卡模块和安全隧道模块。其中虚拟网卡模块工作在核心态，包括网卡驱动子模块和字符设备[2]子模块；安全隧道模块工作

作者简介：田权斌(1979－)，男，硕士研究生，主研方向：信息安全，计算机网络；李立新，副教授、博士；周雁舟，副研究员、硕士
收稿日期：2007-09-27 **E-mail：**tian_quanbin@126.com

在用户态，包括身份认证子模块、传输加密子模块和通信封装子模块。其主要模块组成和对数据处理的流程如图 2 所示。服务器和客户端的区别主要在以下几个模块：(1)通信封装子模块，服务器端作为守护程序等待客户端的连接，客户端主动发起连接；(2)身份认证子模块，客户端主要进行数字签名工作，服务器端主要进行产生随机数和验证签名工作；(3)服务器端需要维护一张VPN地址表，记录客户端VPN地址和真实地址的对应关系，用于转发客户端之间的数据包。

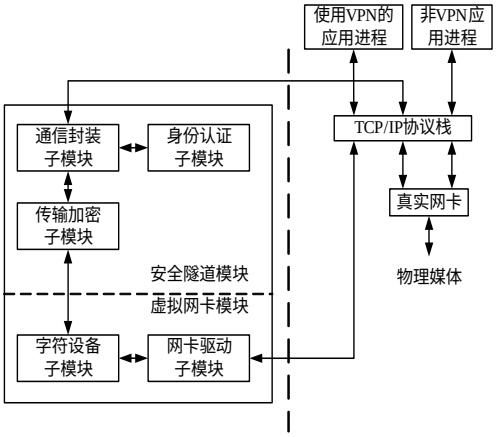

图 2　虚拟以太网 VPN 系统模块结构及数据处理流程

2.4　系统工作流程

VPN 的建立首先需要客户端和服务器建立安全连接，系统建立安全连接的主要流程如下：

(1)服务器端通信封装子模块进行初始化；

(2)客户端安全隧道模块的通信封装子模块向服务器端发出连接请求；

(3)服务器端通信封装子模块接受请求，并建立普通连接，调用身份认证子模块产生随机数发往客户端；

(4)客户端用私钥对随机数进行签名；

(5)客户端将自己的证书和签名结果发送到服务器端；

(6)服务器从签名内容中取出证书序列号，进而查询客户端相应证书的有效性；

(7)证书有效服务器调用身份认证子模块对客户端的签名结果进行解签；

(8)服务器比较解签结果和产生的随机数，符合则初始化虚拟网卡，增加客户端的 VPN 地址和真实地址关系表项，否则返回。

安全连接的流程如图 3 所示。

与服务器建立了安全连接后的客户端就可以和服务器或其他与服务器，建立安全连接的客户端，并进行安全通信。通信时系统对数据的处理流程如下：

(1)数据发送端：

1)使用 VPN 的应用数据经过 TCP/IP 协议栈的封装形成以太网数据包；

2)虚拟网卡模块的网卡驱动子模块接收 TCP/IP 协议栈的数据包发往字符设备子模块；

3)工作在用户态的安全隧道模块检测字符设备子模块中是否有数据可读，当有数据时，从字符设备子模块中读取数据包；

4)传输加密子模块对数据包进行添加 ID 号、计算哈希值[3]和加密操作后送往通信封装子模块；

5)通信封装子模块将数据包再发往 TCP/IP 协议栈，最后经由真实网卡发送出去。

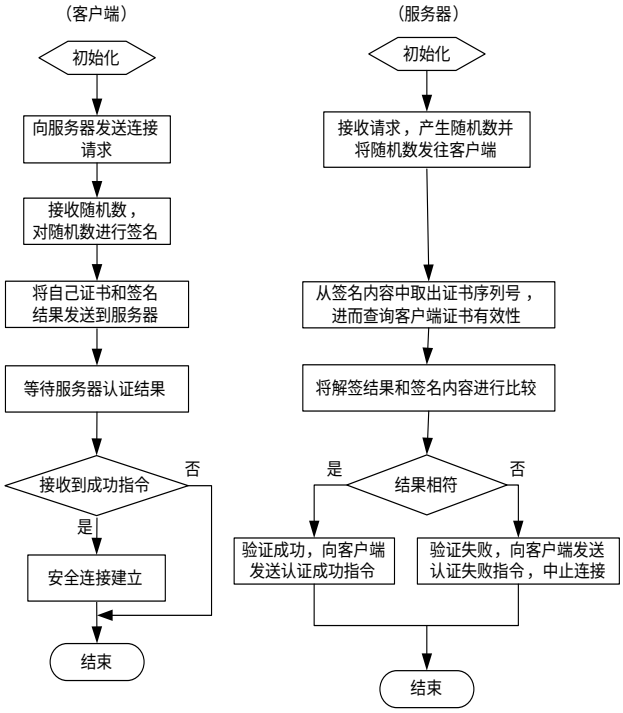

图 3　安全连接建立流程

(2)接收端是服务器时，操作如下：

1)真实网卡接收到发送端发来的数据包，并送往 TCP/IP 协议栈解封装；

2)TCP/IP 协议栈将解封装后的数据发送到通信封装子模块中；

3)通信封装子模块将数据解封装后送往传输加密子模块，传输加密子模块对数据进行解密、验证完整性和进行 ID 号判断，如果错误则丢弃数据包；

4)如果验证正确则解析出数据包中的目的 VPN 地址，如果是服务器本身 VPN 地址，数据的处理方式和接收端是客户端的情况相同；

5)如果不是服务器本身 VPN 地址，则查询 VPN 地址表得到目的 VPN 地址对应客户端的真实地址；

6)将接收到的数据包转发到相应的客户端。

(3)接收端是客户端时，操作如下：

1)真实网卡接收到发送端发来的数据包交 TCP/IP 协议栈解封装；

2)TCP/IP 协议栈将解封装后的数据发送到通信封装子模块中；

3)通信封装子模块将数据发送到传输加密子模块，传输加密子模块对数据进行解密、验证完整性和 ID 号判断，如果错误则丢弃数据包，否则将解析后的数据帧发往字符设备子模块；

4)字符设备子模块检测到传输加密子模块发送来的数据后接收数据传送给网卡驱动子模块；

5)网卡驱动子模块再将数据经 TCP/IP 协议栈交给使用 VPN 的应用进程。

3　VPN 系统的实现

3.1　虚拟网卡模块

虚拟网卡模块工作在核心态，一方面模拟真实网卡驱动的工作过程，和操作系统内核的TCP/IP协议栈交互；另一方

面向用户态提供一个字符设备接口，实现核心态和用户态的数据交互，两方面的功能分别由网卡驱动子模块和字符设备子模块实现。虚拟网卡模块基于开源项目Universal TUN/TAP device driver[4](以下简称TUN/TAP)开发。TUN/TAP有两种工作模式：虚拟点到点网络设备TUN模式，虚拟以太网设备TAP模式。本文采用虚拟以太网设备TAP模式。在TAP模式下，TUN/TAP提供两个应用接口——/dev/tap*X*(字符设备接口)和/tap*X*(虚拟以太网接口)，其中，*X*代表第*X*个设备。用户态的应用可以将以太帧写往字符设备接口/dev/tap*X*，Linux内核会从虚拟以太网接口/tap*X*接收该帧，同时，内核写往虚拟以太网接口/tap*X*的以太网帧可以被用户从字符设备接口/dev/tap*X*接收到。TUN/TAP设备用以下结构描述：

```
struct tun_struct {
char    name[8];    //设备名
unsigned long    flags;    //设备类型
struct fasync_struct    fasync;    //文件异步通知结构
struct wait_queue    *read_wait;    //等待队列
struct device    dev;    //网络设备结构
struct sk_buff_head   txq;    //网络缓冲区队列
struct enet_statistics          stats;    //虚拟网卡状态结构
#ifdef TUN_DEBUG
int debug;    //调试变量
#endif    };
```

3.1.1　网卡驱动子模块

网卡驱动子模块主要实现和 TCP/IP 协议栈交互的作用。需要实现打开网卡设备、初始化网络设备、处理 TCP/IP 协议栈数据包队列、处理多点传送、给出网卡设备相关状态及关闭网卡设备功能。

3.1.2　字符设备子模块

字符设备子模块为用户态提供一个字符设备接口，供用户态调用，从而实现核心态和用户态的交互。字符设备子模块主要需要实现以下功能：打开字符设备，注册字符设备，从字符设备读，写字符设备，关闭字符设备。

3.2　安全隧道模块

安全隧道模块的作用是在用户态构建一个安全的通信隧道。安全隧道模块由通信封装子模块、身份认证子模块和传输加密子模块组成。在安全隧道模块中，采用 SOCKET 通信方式建立通信链路；通过证书认证、数据加密和产生 20 B HMAC 值来保证数据的可靠性、机密性和完整性；通过给数据包添加 4 B ID 号实现抗重放攻击。经安全隧道模块封装后的数据包格式如图 4 所示。

图 4　虚拟以太网隧道数据封装格式

3.2.1　通信封装子模块

通信封装子模块主要用于建立通信链路，发送和接收数据。该模块通过SOCKET通信机制实现，采用TCP协议建立面向连接的通信，通过多进程方式实现多客户端的并发连接[5]。首先服务器端启动侦听套接字守护进程，等待客户端的连接，当检测到客户端的连接时启动一个子进程建立普通连接，接着服务器调用身份认证子模块对客户端进行身份认证，当身份认证通过后就可以进行数据传输。当某一客户端退出时终止子进程，关闭该安全隧道。

3.2.2　身份认证子模块

身份认证子模块是基于数字证书系统体系结构的。采用挑战应答方式对客户端进行单方向认证。客户端主要完成对随机数进行签名，将数字证书内容读入缓存，其中签名采用 ECC 数字签名算法，服务器端主要实现产生随机数和对客户端签名值进行验证工作。

3.2.3　传输加密子模块

传输加密子模块是保证数据机密性、完整性及提供抗重放功能的实现模块。该模块需要提供产生和添加 ID 号、计算和添加 HMAC 值、数据加密和数据解密的功能。其中，ID 号是基于时间产生，HMAC 值的计算采用数字证书系统提供的算法接口实现，加密函数使用通用对称加密算法。在传输加密子模块还封装了加解密函数接口，可支持其他对称密码算法。

4　性能测试

为了进一步检验系统的功能和安全性，对系统进行了以下内容的测试。

(1)网络协议支持测试：通过对 HTTP, FTP, TELNET, SMTP 等应用的测试，表明系统可以很好地支持 FTP、TELNET、SMTP 等应用协议。

(2)穿越网络设备测试：通过在两个网段分别运行系统服务器端和客户端，开放系统所使用端口，在两个网段中间设置有防火墙和 NAT 的情况下，测试系统的连通性。测试表明，系统在穿越 NAT、防火墙等网络设备时，具有很好的穿透性。

(3)传输速率测试：在 100 Mb/s 的两个网段上分别建立服务器和客户端，通过在一端建立 FTP 服务器，在另一端下载固定大小的文件来测试系统的传输速率，通过 10 次测试，平均速率为 1 017.672 KB/s，可以满足普通的网络应用需求。

(4)身份认证测试：首先通过系统建立安全连接确保系统的连通性，然后断开连接，分别测试在客户端修改、删除证书文件或证书对应的私钥文件的情况下系统的连通性，通过测试，在客户端修改、删除证书文件或证书对应的私钥文件的情况下系统都无法建立连接，从而验证了基于数字证书认证的有效性。

(5)加密测试：通过 Sniffer 抓包工具分析物理网卡和虚拟网卡的数据包，可以看出经系统加解密前后的数据包及 HMAC 值和数据包的 ID 号，从而验证了数据加密及完整性和抗重放攻击的有效性。

5　结束语

本文给出了 Linux 下一种简单的基于虚拟以太网的 VPN 的设计实现方法，实现了一个 VPN 原型系统。系统采用的基于虚拟以太网的方法在应用层构建隧道建立 VPN，其优点在于避免了在核心态改变 IP 数据包格式带来的问题，从而可以很好地穿越各种网络设备；支持多种网络接入方式和多种网络协议；设计了灵活的加密接口，可以支持多种加密算法。目前的不足之处在于：(1)系统认证方式只对客户端进行单向认证，安全性强度较弱，下一步应加入对服务器端的认证；(2)系统在处理数据包时两次经过 TCP/IP 协议栈，处理速度较慢，不能满足高速网络应用的需求。总的来说，本文基于虚拟以太网的 VPN 实现方法及系统，实现了 VPN 的安全功能，是一种灵活的 VPN 实现方法，在实现远程安全传输和构建虚拟专用网方面有着很好的应用前景。　(下转第 134 页)