分布式VPN网关及其密钥协商的同步

庞 磊,蒋炎河,何德峰
(江南计算技术研究所,江苏 无锡 214083)

摘 要:VPN是目前最流行的安全技术产品,它能够提供非常好的安全通信服务。但是在面临人们日益增长的速度需求时,VPN技术和设备的发展应走向何方?设计了一种基于分布式操作系统的VPN网关,其目的是满足人们对通信业务的传输速度越来越高的要求。采用负载均衡技术的思路,把VPN网关分解为3个部分:IKE服务器、隧道服务器和负载均衡器。最后讨论了这个系统同步的问题,设计出了一种可行的分布式VPN网关的实施方案。分布式VPN网关的提出为VPN技术和设备的发展指出了一个可行的方向。

关键词:IP层的安全;密钥交换;安全关联;负载均衡;网络模拟
中图分类号:TP393.08 **文献标识码**:A **文章编号**:1673-629X(2006)09-0216-03

A VPN Gateway Based on Distributed System and Synchronization of Its Key Exchange

PANG Lei, JIANG Yan-he, HE De-feng
(Jiangnan Institute of Computing Technology, Wuxi 214083, China)

Abstract: VPN can provide a nice security service, but when it faces the needs of higher and higher working speed, what the solution to be? This paper proposed a VPN gateway based on distributed system which satisfy our need on the transfer-speed of the communication work. With the technique of load balance technique, separate the VPN gateway into IKE server, LB server and several tunnel servers. And then designed a synchronizing theory of its key exchange. This distributed VPN-gateway system point out a way to develop the high-speed VPN-gateway technology.

Key words: IPSec; IKE; SA; load balance; NS

0 引 言

随着信息技术的发展,人们对信息传输在安全性和保密性方面的要求越来越细腻。对于某些敏感的信息,人们往往希望能够掌握它的传播和扩散。某些秘密的消息,人们希望能够毫无顾虑、万无一失、秘密和方便快捷地传送到目的地。在这种形势下,许多安全通信的技术和产品就应运而生了。虽然安全通信也涉及存储加密等技术领域,但是一般认为安全通信的主体是通信过程。目前的安全通信产品种类繁多,其中应用最为广泛的是基于IPSec技术的虚拟专用网(Virtual Private Network, VPN)产品,VPN产品的实施主要是以建立VPN安全网关为主。

网络的各个核心部分随着业务量的提高、访问量和数据流量的快速增长,其处理能力和计算强度也相应增大,这使得人们对VPN网关速度的要求越来越高。在此情况下,立竿见影的做法是扔掉现有设备去做大量的硬件升级,但是如果再面临下一次业务量的提升,这又将导致再一次硬件升级的高额成本投入,甚至性能再卓越的设备也不能满足当前业务量需求的时候,应该采用什么解决方案呢?分布式VPN网关是一个比较好的解决方法。

收稿日期:2005-11-16
作者简介:庞 磊(1981-),男,陕西咸阳人,硕士研究生,研究方向为计算机安全。

1 相关的技术

1.1 VPN网关

VPN是目前应用的最多的安全通信产品,它既有软件实现,也有专用设备实现。VPN主要是以IPSec技术为基础开发出来的,从1995年开始,IETF着手研究制定了一套工作在IP层的安全协议,也就是IPSec(IP Security),用于保护IP通信的安全。

IPSec对IP数据包的保护有两种方式:隧道方式和传输方式。其中隧道方式是主要的工作方式。其原理是在隧道一端的安全网关对要保护的数据按照外出SA(Security Association,安全关联)的参数进行加密封装和签名,在隧道另一端的安全网关按照相应的进入SA进行认证解密解封装。这里所说的SA是安全网关之间通过IKE(Internet Key Exchange)程序协商出来的一系列封装数据包的参数,SA的内容见表1。

1.2 负载均衡

负载均衡(Load Balance)提供了一种廉价有效透明的

表1　SA的主要内容

ips_said	SA全局惟一的ID号
ips_life	SA的生存期
ips_authalg	认证算法
ips_encalg	加密算法
ips_flow_d	受保护的目的子网地址段
ips_flow_s	受保护的源子网地址段
ips_key_a	认证密钥
ips_key_e	加密密钥

方法,可以扩展网络设备和服务器的带宽、增加吞吐量、加强网络数据处理能力、提高网络的灵活性和可用性。图1是负载均衡示意图。

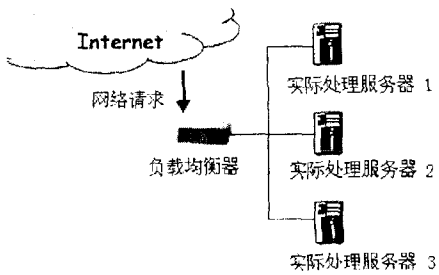

图1　负载均衡器

目前有许多不同的负载均衡技术用以满足不同的应用需求,如软/硬件负载均衡、本地/全局负载均衡、更高网络层负载均衡,以及链路聚合技术。其中,更高网络层负载均衡是设计分布式VPN网关所倚重的思路。

更高网络层负载均衡,通常操作于网络的第四层或第七层。第四层负载均衡将一个IP地址映射为多个内部服务器的IP地址,对每次TCP连接请求动态使用其中一个内部IP地址,达到负载均衡的目的。第七层负载均衡控制应用层服务的内容,通过检查报文HTTP报头,根据报头内的信息来执行负载均衡任务。

2　分布式VPN网关

根据IPSec[1]和负载均衡的技术特点,可以把IKE程序和数据封装与解封装程序分离开,交给IKE服务器和隧道服务器分别完成。再由负载均衡器将它们整合成一个整体,实现分布式VPN网关。这里提出了一种分布式VPN网关[2]的系统框架结构,然后着重讨论了解决系统密钥协商中最主要的同步问题的方案。

2.1　系统框架结构

IKE服务器负责与其他的VPN网关协商密钥和策略,然后把协商的结果即SA分发给各个隧道服务器,更为重要的,IKE服务器还负责系统同步的任务。隧道服务器按照接收到的SA对数据包进行封装和解封装。负载均衡器负责把隧道服务器和IKE服务器整合起来,使系统无论是从受保护的子网还是从信任度低的Internet上看上去都是一个整体。图2是分布式VPN网关系统结构框架。

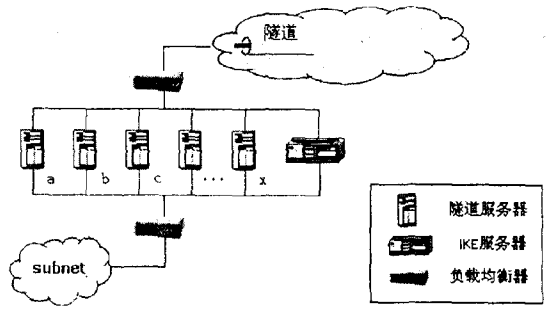

图2　分布式VPN网关系统结构框架

2.2　系统的同步问题

隧道服务器用IKE程序分发协商出来的SA对数据包进行封装和解封装。由于SA中包含对数据包进行加解密和签名以及认证的各种密钥,为了提高系统的安全性,于是要求SA中有一个有限的生存期,在SA的生存期到期之后,应该重新协商并应用新的SA。

一般要求隧道服务器在没有建立起有效的SA时不能够对数据包进行封装和解封装,这包括下列两种情况[3]:

a.在VPN网关初始化的时候,隧道服务器在没有接收到IKE程序发来的最新的SA时,不能够对数据包进行封装和解封装。

b.在VPN网关运行过程中,隧道服务器在SA寿命到期后,不能够对数据包进行封装和解封装。

在这个协议指导下,尽管能够最大可能地保证系统的安全性,但是在重新协商密钥的时候,整个系统必须停止目前的通信,直到新的SA协商完毕。这将造成通信抖动,如果遇到协商消息数据报文的延时甚至丢失或者有些系统使用者希望把SA的有效期设置的尽量短以达到高的安全级别,这个中断和抖动就会变得让人难以忍受。

为了消除中断和抖动,加入了下面的机制。

2.3　同步问题解决方案

为了解决上述两种同步问题,需要让两个负载均衡器了解各个隧道服务器建立SA的状态[4]。考虑到系统的特性,设计出以下解决方案:

首先,引入下面3个参数:

(1)在对数据包进行封装的时候加入ipsec_id标记。

(2)在隧道服务器中引入SA联合体SAU(见表2),存放针对每条隧道最近的两个SA(SA是单向的,如果是双向的隧道,针对这条隧道每一个隧道服务器上就得维护两个联合体)。

表2　SAU

struct ipsec_sa old_SA
struct ipsec_sa new_SA
bool new_in_use

(3)在负载均衡器上维护一个针对每一条隧道的SA的同步状态列表(见表3),存放每条隧道最近的两个SA的ips-said。

表 3 同步状态表

隧道服务器 a	隧道服务器 b	隧道服务器…	隧道服务器 x
ips_said_old	ips_said_old	ips_said_old	ips_said_old
ips_said_new	ips_said_new	ips_said_new	ips_said_new

下面,对同步方案进行描述:

1)系统初始化或者某个 SA 到达软时间的时候,IKE 服务器和另一端的 IKE 服务器进行 SA 的协商。

2)在协商完毕后,IKE 服务器把协商出来的 SA 逐一发送给各个隧道服务器。

3)每一个隧道服务器接收到 IKE 服务器发送来的 SA 后,把 SAU 中存放的 SA 进行更新,如果是系统初始化,SAU 中的 SA 都为空,那么直接把收到的 SA 存放成 new_SA。如果不是系统初始化,就先把 new_SA 复制到 old_SA,然后再把接收到的 SA 存放到 new_SA。最后把 new_in_use 置为 0。

4)隧道服务器更新 SAU 完毕后,发送一个成功回执到 IKE 服务器,IKE 服务器再把此隧道服务器的序号(TSid)附加此次 SA 的 ips_said 发送给两个负载均衡。

5)负载均衡器接收到 IKE 服务器发来的 TSid + ips_said 后,更新同步状态列表。如果是系统初始化,同步状态列表为全零,负载均衡器直接把 ips_said 写入相应的隧道服务器状态下的 ips_said_new,如果不是系统初始化,就先把相应隧道服务器状态下的 ips_said_new 复制到 ips_said_old,然后 ips_said 写入 ips_said_new。

6)SA 到达硬时间的时候,就应该应用新的 SA 了。这时 IKE 服务器向所有的隧道服务器和靠近受保护子网的负载均衡器发送硬时间到达的消息。

7)隧道服务器收到硬时间到达消息后,把 new_in_use 置为 1,靠近受保护的子网的负载均衡器在收到硬时间到达消息后,把相应的 SA 的同步状态表再进行一次更新。即把每一个隧道服务器状态下的 ips_said_new 复制到 ips_said_old,再把 ips_said_new 置为 0。

8)等到 SA 的软时间到达后,重复执行步骤 1)~8)。

在图 2 中设计的分布式网关中加入了上述同步机制后,还需对负载均衡器和隧道服务器作如下规定:

a.子网向隧道发送数据时,要经过隧道服务器的封装。靠近子网的负载均衡器只把数据包转发给同步状态表中 ips_said_old 不为零的隧道服务器。

b.隧道服务器对数据包进行封装的时候,首先根据数据包的目的和源地址查找相应的 SAU,再检查 SAU.new_in_use,如果为 0,就用 SAU.old_SA 对数据包进行封装;如果为 1,就用 SAU.new_SA 对数据包进行封装,封装完毕后发往隧道。

c.子网接收隧道发送来的数据时,要经过隧道服务器的解封装,靠近隧道口的负载均衡器要首先对数据包中的 ipsec_id 标记进行检查,只把数据包转发给同步状态表中包含此 ipsec_id 的隧道服务器。

d.隧道服务器对数据进行解封装的时候,首先根据数据包的目的和源地址查找相应的 SAU,再根据数据包上 ipsec_id 判断是用 SAU.new_SA 对数据解封装还是用 SAU.old_SA 对数据解封装。解封装完毕后,把数据包发送到子网。

上述同步机制,对分布式 VPN 网关内隧道服务器的同步主要靠 IKE 服务器,不同 VPN 网关之间同步是靠负载均衡器对数据包的转发实现的。

3 建模和分析

VPN 网关内的不同隧道服务器的同步依靠 IKE 服务器和负载均衡器的策略控制来实现。假设某一隧道服务器出现异常,没有正确得到最近的 SA,则它就不会发送成功回执,负载均衡器就不会更新它的状态,用新的 SA 封装的数据包就不会转发到这个隧道服务器上来。

VPN 网关之间的同步用实验来模拟。NS(网络仿真器[5])是网络模拟的强大工具,它对网络上的离散事件模拟得尤为出色,笔者在 NS 的环境下设计一个模拟分布式 VPN 网关的同步情况的简化模型,只考虑两个 VPN 网关之间负载进衡器的同步,如图 3 所示。

图 3 中模拟一个单向的隧道,其中节点 1 和节点 2 分别是隧道两端的负载均衡器,节点 0 和节点 3 都是隧道服务器加密受保护子网发向隧道的数据包,并在数据包上打上一个从 0 开始递增的标识,这个标识用于模拟 SA 的更新。当硬时间到达的时候,标识加 1 并重新计时。节点 1 和节点 2 的缓存无限大,节点 1 把经过封装的数据包发往节点 2,节点 2 判断该数据包是否丢弃。程序设计了几个事件时间,分别为软时间 Tr、硬时间 Ty 和节点 n 第 x 次应用新的 SA 时间也就是第 x 次硬时间到达的时间 Tx_n。节点同步所需的最大时间是 Tt,各节点随机在 3 个时间点后加上一个小于 Tt 的时间按照上述同步方案做出相应的状态转换。规定节点 2 在 Tx_2 到 $Tx\_2 + Tr + Tt\_x2$ 内可以接受标识为 x 和 $x-1$ 的包,在 $Tx + Tr + Tt\_x2$ 到 $Tx + Ty(= Tx + 1)$ 内只能接受标识为 x 和 $x+1$ 的数据包。Tt_x2 是节点 2 在第 x 次循环中到达软时间到接收到隧道服务器发来的第一个同步完毕信号的时间间隔。假设 Tt_x2 最大为 Ts,即 $0 < Tt\_x2 < Ts$。

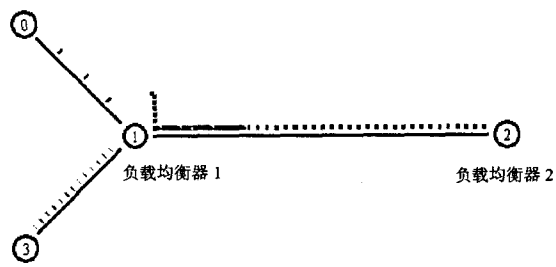

图 3 VPN 网管之间单向隧道的模拟

笔者分析只要满足以下条件,就可以实现无抖动:

(1)软时间大于最大同步时间 Tr > Tt;

(下转第 221 页)

为此,提出了RUDP(Reliable UDP)协议来满足可靠通信的要求[3]。

4 RUDP协议简介

从计算机网络层次体系的角度来看,RUDP的层次结构如图1所示。可见,RUDP就是在原协议的传输层的TCP/IP协议和应用层之间加入了一层为保证可靠数据传送而实现UDP的软件模块而形成RUDP的一个五层体系结构[3]。这样就可以利用TCP/IP的UDP协议实现一种基于消息的面向连接的可靠数据传递机制[4]。

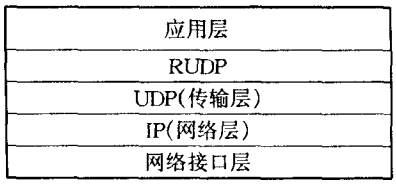

图1　RUDP层次体系结构

RUDP协议实现的基本原理:

作为可靠的面向连接的数据传递机制,双方在通信之前必须要建立连接,RUDP采用了和TCP类似的3次握手协议[5]。在连接建立过程中,考虑双方同时发起建链的情况,按照请求的选择原则对双方的IP地址进行比较,拒绝IP地址小的一方发起的连接请求。

RUDP通过滑动窗口协议来实现收发同步和流量控制。在接收任务收到消息之后,接收方窗口缩小,当调度任务将消息处理完之后,将窗口扩大。如果调度任务扩大窗口时,发现窗口从0扩大到X则向通信任务发送一条消息,通知窗口变化,经过协议处理任务处理之后向对端发送窗口扩大通知。调度任务处理完一条消息,使得接收窗口扩大到1,如果立即向通信任务发送窗口扩大消息,导致内部和外部消息流量都增加,所以此处X的值一般要大于1。为每个连接维护两个定时器,一个是发送定时器,一个是接收定时器。两个定时器被连接的各个状态进行复用,在不同状态有不同的含义。

① 保活定时器,在发送状态迁移并执行链路保活时设置;

② 窗口探测定时器,在发送状态迁移并执行窗口探测时设置;

③ 重传定时器,在发送状态迁移并执行重传的时候设置;

④ 接收定时器只作为链路检测,定时检测对端是否有消息或者保活消息过来。

RUDP协议充分吸收了TCP协议设计上的优点,在UDP协议的基础上提供了面向连接的可靠通信机制,使其更加符合以太网的特点,可以充分利用以太网的优势来服务电信网络。

5 总　结

通过对TCP,UDP协议的详细对比分析,论证了在以太网络中UDP协议的优势,同时针对UDP在可靠性方面的不足进行了改进,简单介绍了RUDP协议的原理。通过分析可以看出采用RUDP的效率在以太网络中要优于TCP协议,因此可以作为电信设备模块之间的通信协议。

参考文献:

[1] Postel J. Transmission Control Protocol[S]. RFC 793, DARPA. 1981.

[2] Postel J. User Datagram Protocol[S]. RFC768. 1980.

[3] Velten D. Reliable Data Protocol[S]. RFC908. 1984.

[4] Comer D E. 用TCP/IP进行网络互联(卷2)[M]. 张　娟,王　海,译. 北京:电子工业出版社,1998.

[5] Wright G R, Stevens W R. TCP/IP详解卷2:实现[M]. 陆雪莹,蒋　慧,等译. 北京:机械工业出版社,1999.

(上接第218页)

(2)软时间到硬时间的间隔大于最大的Tt_x2,即Ty - Tr > Ts。

Tt和Ts是不可控制的,但是Tr和Ty是我们设置的。在实验中设置Tr和Ty满足上述条件的时候,没有发生丢包现象。由实验证明了结论。

4 结束语

在信息技术飞速发展的今天,人们对网络通信的要求主要分有效传输速度和通信安全两个方面,通常来说,传输速度和安全是存在冲突的,二者都需要耗费大量的计算资源。传统的单一计算机系统资源无法继续发掘的时候,分布式操作系统必将是拓展计算资源的惟一出路。文中设计了一种分布式VPN网关,它利用了分布式操作系统的资源可扩展性强的优势,提高了网络通信中速度和安全两方面性能。所以这是一个实施高速VPN网关的可行方案。

参考文献:

[1] 胡金柱,王　锐,张昭理. 一种分布式信息系统的构建模型及其应用研究[J]. 计算机科学,2003,30(10):96-98.

[2] Davis C R. IPSec VPN的安全实施[M]. 周永斌,冯登国,徐　霞,等译. 北京:清华大学出版社,2002.

[3] Steiner M, Tsudik G, Waidner M. Key Agreement in Dynamic Peer Groups[J]. IEEE Trans Parallel and Distrib Sys, 2000, 11:69-80.

[4] Dondeti L, Mukherjee S, Samal A. Disec: A distributed framework for scalable secure many-to-many communication [A]. Fifth IEEE Symposium on Computers and Communications (ISCC 2000)[C]. New York, USA: IEEE Computer Society Press, 2000. 693-705.

[5] 徐雷鸣,庞　博,赵　耀. NS与网络模拟[M]. 北京:人民邮电出版社,2003.