文章编号: 1001- 2486(2003) 03- 0079- 05

高效因特网密钥交换协议(IKE) 的研究与实现*

曾迎之, 苏金树, 荣 霓
(国防科技大学计算机学院, 湖南 长沙 410073)

摘 要: IKE 协议是因特网安全协议 IPSec 的重要组件, 它的高效实现是 IPSec 的关键所在。在分析了 IKE 协议的基础上, 给出了一种设计结构, 并进行了性能评价和分析。基于实时嵌入式操作系统 VxWorks 环境的测试表明这是一种高效的实现方案。

关键词: IKE; ISAKMP; 安全联盟; 保护套件; 阶段; 模式
中图分类号: TP393. 08 文献标识码: A

Study and Effective Implementation of the Internet Key Exchange Protocol (IKE)

ZENG Ying-zhi, SU Jin-shu, RONG Ni
(College of Computer, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: The IKE protocol is one of the most important components of the Internet Security Protocol (IPSec), the high-efficient implementation of IKE is a key part of IPSec. After the analysis of the basic mechanism of IKE, this paper put forward a design structure and its realization, also the design performance issue is discussed. Test has proved that it is an effective implementation of IKE.

Key words: IKE; ISAKMP; security association; protection suite; phase; mode

为确保应用数据在公共 IP 网络上通信的可靠性、完整性和隐私性, IETF 因特网工程任务组已提出一整套称之为因特网协议安全(IPSec) 的协议。Internet 安全联盟和密钥管理协议 ISAKMP 为 Internet 团体指定了一个标准协议栈。IPSec 提供的安全服务需要使用共享密钥以执行数据验证以及机密性保证任务。如果采用人工增加密钥的方法, 可实现基本 IPSec 协议间的互通性, 但其难以扩展。因此, 需要定义一种标准的方法, 用以动态地验证 IPSec 参与各方的身份、协商安全服务以及生成共享密钥等等。这种密钥管理协议称为“Internet 密钥交换”(Internet Key Exchange, IKE) 。IETF 制定了 IKE 用于通信双方进行身份认证、协商加密算法和散列算法、生成公钥的一系列规范。

IKE 的实现是 IPSec 系统中的重要部分。一般而言, IKE 实现主要是在应用层产生一个任务, 根据 IPSec的应用唤醒, 整个系统采用顺序执行的机制。一旦两阶段协商开始进行, 则 IKE 进程将根据交换的报文及定时器来进行处理。

本文阐述我们在 VxWorks 上的一种实现结构、设计及各部分组成, 最后对性能进行评价。

1 一种 IKE 的实现方案

根据协议的特点, 在实时嵌入式操作系统 VxWorks 的平台上实现了一个 IKE 协议。在该系统中, IKE 作为一个后台进程运行, 主要承担处理用户的管理配置命令、同协商实体的交互、IKE 数据报的处理以及同内核的安全联盟数据库 SADB 的交互。整个 IKE 系统分成 3 个子系统(见图 1) , 分别是 IKE 管理子系统、IKE 处理子系统和内核子系统。

1.1 IKE 管理子系统

管理子系统下辖 UI 模块, IKE 管理 Server 模块, IKE 配置文件, IKE 状态库, 初始化模块。各模块功

* 收稿日期: 2002- 11- 28
基金项目: 国家自然科学基金重大研究计划资助项目(90104001)
作者简介: 曾迎之(1975—) , 男, 硕士生。

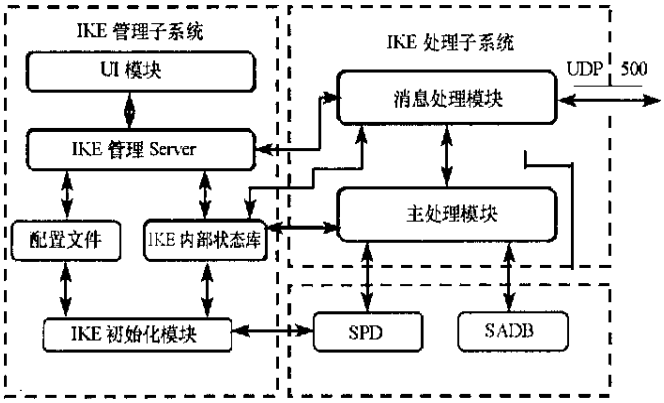

图 1 一种 IKE 实现的软件模块示意图

Fig. 1 Software architecture of an IKE implementation

能介绍如下:

ъ UI 模块: 处理用户输入的命令, 并将结果反馈给用户管理接口。

ъ IKE 管理 Server: 负责监控几个消息队列, 对其中的事件调用注册的处理函数。

ъ 配置文件: 负责记录系统的初始运行参数, IKE 系统根据这些参数进行初始化。

ъ IKE 状态库: 负责记录 IKE 运行期间需要协商的信息和当前的安全联盟 SA 信息。这些数据通过统一组织管理来提供统一接口, 便于各模块共享。在具体的实现中没有设置复杂的操作, 而是把各项数据结构通过指针组织成链表形式。查找、更新、删除、添加等命令是在对应的链表上操作完成。

ъ IKE 初始化模块是在 IPSec 模块初始化时被调用, 其主要功能是设置 IKE 初始化参数、加载安全策略、加载 IKE 两阶段协商提案, 等等。IKE 一旦初始化完毕后, 作为一个独立的任务在嵌入式操作系统 VxWorks 下运行, 通过消息队列与 IPSec 模块通信。

1.2 IKE 处理子系统

IKE 处理子系统负责完成 IKE 协商处理的主要工作, 在阶段一实现的是主模式, 在阶段二为快速模式。主模式交换提供了身份保护机制, 经过 3 个步骤(策略协商交换、Diffie-Hellman 共享值和 nonce 值的交换、对对方身份的验证交换) , 共交换了 6 条消息, 最终建立了 IKE SA。

在建立好 IKE SA 之后, 可用它为其它安全协议生成相应的 SA。这些 SA 是通过第二阶段的快速模式交换来建立的。对一次快速模式交换来说, 它是在以前建立好的 IKE SA 的保护下完成的。快速模式交换通过 3 条消息建立 IPsec SA, 头两条消息协商 IPsec SA 的各项参数值, 并生成 IPsec 使用的密钥; 其中第 2 条消息还为响应方提供在场的证据; 第 3 条消息为发起方提供在场的证据。

如图 1 所示, 按照功能 IKE 处理子系统被划分成 2 个模块: IKE 消息处理模块和 IKE 主处理模块。其中消息处理模块负责各种消息的处理, 包括: 系统运行期间 IKE 管理 Server 要发送的管理消息, 内核发出的 SA 请求, 更新消息, 以及网络上传来的协商消息(以 UDP 的形式走 500 号端口) 。另外, 系统自身由于网络拥塞产生超时消息或设置的 SA 生命过期更新消息等。消息处理模块被设计成服务器模式, 并对每种消息类型设置一条消息队列。模块轮询这几条消息队列, 如果有消息就调用对应的消息处理函数来处理。

在我们实现的 IKE 处理子系统中, 一次典型的 IKE 协商的基本过程如图 2 所示(数字标号表示步骤执行的先后次序) 。其中:

1 ——来自以太网 Ethernet 0 试图发起 IPSec 保护的报文进入 IP 层;

2 ——IP 层收到数据包, 转发给 IPSec 处理模块;

3 ——IPSec 处理模块查看数据包, 到安全策略数据库 SADB 中查询是否有对应到发起 IPSec 安全通信一方的安全策略 SA;

4 ——到安全策略库中检查 IKE 策略(IKE Policy) ;

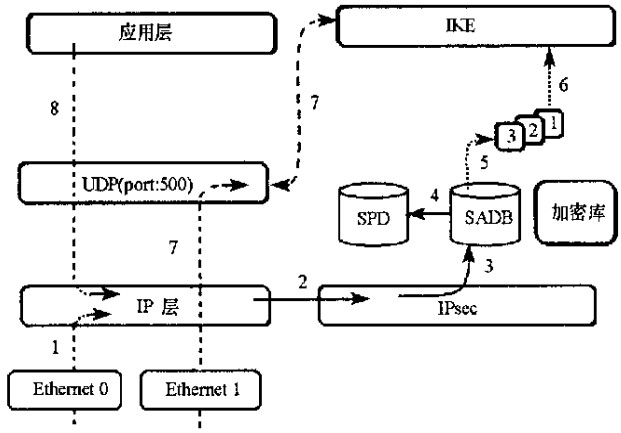

图 2 IKE 协商产生 SA 的步骤示意图

Fig. 2 IKE negotiating SA

5~ 6——如果不存在到发起 IPSec 安全通信一方的 SA, 则需通过 IKE 协商建立 IPSec SA;

7——通过 IKE, 要启用 IPSec 保护协议的两方在 UDP 的 500 号端口上开始协商建立 IPSec SA。协商决定通信双方需要采用的加密算法(如 DES) 、相同的验证算法(如 MD5) 等, 并且同时通过 DH (Diffie–Hellman) 协议建立了共享的会话加密密钥;

8——启用 IPSec 保护协议的双方开始 IPSec 保护下的安全通信。

IKE 主处理模块负责对 IKE 状态库的查询, IKE 协商消息的验证, 以及与内核子系统的通信。

为了保证各个模块能对协商的数据进行共享, 设计了供主处理模块查询的 IKE 状态库模块, 在统一的接口上实现查询、更新、删除、添加等操作。

根据 IKE 协议对应的 RFC 文档, 每种模式的协商过程都有固定的消息条数, 且每条消息的内容都做了明确的规定。不同的认证方式下消息载荷的加密/解密方式有所不同, 以此来区分不同模式下的不同消息。我们通过设置状态标识参量, 有效地解决了协商状态的划分问题。每种状态都有对应的状态标识, 当协商过程中下一条消息到来时, 主处理模块就调用此时状态所对应的消息处理函数进行分析处理, 验证通过就更改状态标识参量到下一个状态。消息处理函数的条件分支语句分别对应不同数值的状态标识参量。当需要对某条消息进行处理时, 以不同的状态参数值作索引, 找到不同消息的状态函数处理入口进行下一步的处理。这样的处理, 既能够提高查询和判断的效率, 又可以明确状态的变迁。下面列出了我们在具体实现中采用的分支函数和状态标识参量的对照表, 如表 1 所示。

1.3 内核子系统

IKE 作为一个应用层协议实现在应用层, 同内核进行 SA 消息的传递是必须实现的。我们设计的内核子系统包括安全策略数据库 SPD 和 SADB 两个子模块。SPD 是 IPSec 在系统内核中维护的系统安全策略的副本。在包处理的过程中, 对于每一个入栈或者是出栈的数据包都要检索 SPD, 查找可能的安全应用。SA 是构成 IPSec 的基础, 它规定了用来保护数据报安全的 IPSec 协议、加密算法、验证算法、密钥以及有效时间等, 两个通信实体间为进行安全通信而协商建立起来的一种约定。SADB 存储 SA 的各项安全联盟参数。

我们通过接口函数实现内核子系统和 IKE 的通信。IKE 和 SPD 之间的接口函数有两个: ike _ spd _ get _ outbound_ policy 负责 IKE 到安全策略数据库 SPD 中相应出栈安全策略的查询; ike _ spd _ get _ security _ policy _ indicator在 IPSec 出栈安全策略检查中被用来获取安全策略执行指示。IKE 同 SADB 通信的接口也是双向的。IKE 负责动态填充 SADB, 要向 SADB 发送消息, 也要接收从 SADB 返回的消息。在接口实现中我们主要设置了下面三个接口处理函数: ike _ control _ dispatcher _ process _ packet _ from _ ipsec 用来处理被 IPSec 模块处理过的与 SADB 相关的 IP _ MESSAGE 报文信息; protection _ suite _ manager _ renew _ protection_ suite 负责更新 SA 对应保护套件的参数; protection _ suite _ manager _ delete _ sa _bundle负责从

SADB里删除SA,清除作废的保护套件及其管理器。

表1 IKE协商在不同状态参数值下的对应分支执行函数表

Tab.1 Offset implement functions during the IKE negotiation

主模式(第一阶段)		
状态标识参量 main_mode_state	发起方/响应方 / 消息序号	分支执行函数
MM_NO_STATE	发起方发出第1条消息	ike_channel_builder_send_sa_message 发送SA消息
	响应方发出第2条消息	ike_channel_builder_send_sa_message 回送SA消息 同时响应方修改channel builder下的状态标识参量值: main_mode_state= MM_SA_STATE
MM_KE_STATE	发起方发出第3条消息	ike_channel_builder_send_ke_message 发送密钥交换消息
	响应方发出第4条消息	ike_channel_builder_send_ke_message 回送密钥交换消息 同时响应方修改channel builder下的状态标识参量值: main_mode_state= MM_KE_STATE
MM_ID_STATE	发起方发出第5条消息	ike_channel_builder_send_id_message 发送身份ID消息
	响应方发出第6条消息	ike_channel_builder_send_id_message 回送身份ID消息 同时响应方修改channel builder下的状态标识参量值: main_mode_state= MM_ID_STATE
快速模式(第二阶段)		
状态标识参量 quick_mode_state	发起方/响应方 / 消息序号	分支执行函数
QUICK_MODE_NO_STATE	发起方发出第7条消息	ipsec_negotiator_send_qm_sa_message 到此,阶段一结束,协商发起方同时创建阶段二的协商对象,发出第二阶段的第一条消息
	响应方发出第8条消息	ipsec_negotiator_receive_qm_initial_message 然后 ipsec_negotiator_send_qm_sa__message
QUICK_MODE_SA_STATE	发起方发出第9条消息	ipsec_negotiator_send_qm_final_message
QUICK_MODE_ESTABLISHED_STATE	响应方	ipsec_negotiator_receive_qm_initial_message 接收消息
		ipsec_negotiator_send_qm_message (空操作)
		ipsec_negotiator_add_bundle_to_sadb 在SADB中添加新协商出来的SA参数

2 评价与分析

为检验本系统的协商效率,我们对其进行了协商测试。主要测试协商的完整性和完成一次协商所需的时间。测试平台:2台运行嵌入式操作系统VxWorks的具有IPSec功能的路由器担任安全网关;PC1和PC2分别是这2个安全网关下的用户端机;在PC3上运行抓包软件snifferPro V4.60.01; PC1和PC2轮流负责发起IKE协商。

通过查看安全网关控制台显示和sniffer抓获报文的分析,5次协商的成功率是100%。

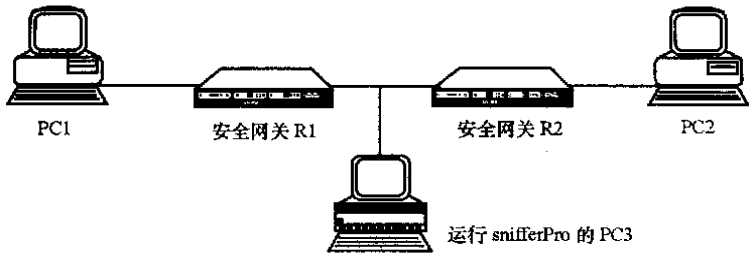

图3 IKE 协商测试平台示意图

Fig. 3 Test environment of IKE negotiation

对 10 次协商完成时间取平均值, 得到了本系统平均完成一次协商所需的时间: 2. 773s。表 2 是其中一次的数据。

表 **2** **sniffer** 抓获的一次完整协商数据包参数表

Tab. 2 The negotiation data packets captured by sniffer

序号	数据包长度(B)	相对时间	时间间隔	抓包绝对时间
1	146	0 00: 00. 000	0. 0000	2002- 08- 29 17: 01: 40
2	122	0 00: 00. 344	0. 344941	2002- 08- 29 17: 01: 40
3	222	0 00: 00. 650	0. 305802	2002- 08- 29 17: 01: 40
4	222	0 00: 01. 006	0. 355729	2002- 08- 29 17: 01: 41
5	110	0 00: 02. 051	1. 045125	2002- 08- 29 17: 01: 42
6	110	0 00: 02. 117	0. 066187	2002- 08- 29 17: 01: 42
7	198	0 00: 02. 333	0. 216710	2002- 08- 29 17: 01: 42
8	198	0 00: 02. 667	0. 333949	2002- 08- 29 17: 01: 42
9	94	0 00: 02. 772	0. 105188	2002- 08- 29 17: 01: 42

从表 2 中可以看到, 在协商过程总共 2. 7s 的过程中, 协商发起方发送序号为 5 的第 3 条消息的时间间隔最长, 达 1. 045s。10 次协商中该项的平均时长为 1. 051s。这是因为这一步执行了证明前 4 条消息完整性的哈希算法。我们考虑在下一步的系统改进工作中, 散列摘要部分的哈希计算完全可以考虑采用更为高效的硬件(DSP 或者 FPGA) 来实现, 这样就可以显著缩短第 3 条消息的时间间隔, 从而进一步提高 IKE 系统的整体性能。

3 结束语

本文论述了 IKE 在嵌入式操作系统环境中的实现, 能较为快速地完成 SA 协商。但参照完整规范的 RFC 相关标准, 还有一定的差距, 如还没有实现野蛮模式等。将系统以专用芯片实现并提供向上的用户接口, 将是我们下一步工作的方向。

参 考 文 献:

[1] Maughan D, Schertler M, Schneider M, Turner J. Internet Security Association and Key Management Protocol (ISAKMP)[S]. RFC 2408, November 1998.

[2] Harkins D, Carrel D. The Internet Key Exchange(IKE)[S]. RFC 2409, November 1998.

[3] Piper D. The Internet IP Security Domain of Interpretation for ISAKMP[S]. RFC 2407, November 1998.