

Linux 2.6 内核 IPSec 支持机构的研究与分析

周莉², 黄宪¹, 陆建德¹

(1. 苏州大学 计算机学院, 江苏 苏州 215006;

2. 苏州市职业大学 计算机工程系, 江苏 苏州 215000)

摘要:新的 Linux 2.6 内核提供了对 IPSec 的支持机构, 文中对 Linux 2.6 内核中新加入的 IPSec 代码进行了深入分析。对比先前不支持 IPSec 的网络协议栈的 Linux 内核, 揭示了 Linux 2.6 内核“无缝”接入 IPSec 处理的方法; 阐述了内核中 IPSec 重要组件——安全关联 SA、安全策略的设计思想以及相关数据库 SAD 和 SPD 的构建方法; 分析了基于 Netlink 套接字通信的内核 IPSec 管理模块、内核加密算法函数库, 总结出一套 Linux 2.6 内核 IPSec 支持机构提供给用户进程的调用方法。

关键词: IPSec; 安全关联; 安全策略; Netlink 套接字

中图分类号: TP311

文献标识码: A

文章编号: 1673-629X(2007)05-0191-04

Research and Analysis of IPSec Support Mechanism in Linux Kernel 2.6

ZHOU Li², HUANG Xian¹, LU Jian-de¹

(1. School of Computer, Suzhou Univ., Suzhou 215006, China;

2. Department of Computer Engineering, Suzhou Vocational College, Suzhou 215000, China)

Abstract: New Linux kernel 2.6 has provided the IPSec support mechanism, and this paper has made thorough analysis to the IPSec code embedded in Linux kernel 2.6. Compares the Linux 2.6 kernel's protocol stack with previous kernels' that don't support IPSec, and unleashes the methods of the IPSec seamless integration in Linux 2.6 kernel. It illustrates the design thoughts of SA and SP structures that are both important IPSec elements, and constructing methods of SAD and SPD. Also analyzed the management module that based on the communication of Netlink sockets and the crypto function bank, and summarized a suit of methods that provided by IPSec support mechanism in Linux kernel 2.6 that can be utilized by user process.

Key words: IPSec; security association; security policy; Netlink socket

0 引言

Internet 要持续发展, 安全是一项重要前提。IPSec 协议在 IP 层提供数据源身份认证、数据完整性检查以及机密性保证, 可以防范数据受到来历不明的攻击。IPSec 协议不仅适用于 IP 目前的版本 IPv4, 也适用于下一代的 IP——IPv6。除此之外, IPSec 协议可为运行于 IP 顶部的任何一种协议提供保护, 如 TCP, UDP 和 ICMP 等等^[1]。作为目前最易扩展、最完整的一种网络安全方案, IPSec 正得到越来越广泛的应用和支持。

作为内核的一个新特性, Linux 2.6 内核提供了全

新的对 IPSec 的支持机构。Linux 2.6 内核中的 IPSec 支持机构主要包括: 内核网络协议栈中 IPSec 处理的“无缝接入”, AH 协议、ESP 协议的处理, IPSec 重要组件安全关联 SA 和安全策略 SP 的数据结构及相关访问算法, 安全关联数据库 SAD 和安全策略数据库 SPD 的构建和访问, IPSec 加密算法函数库的支持, 基于 Netlink 套接字的内核态与用户态的通信支持, 以及内核 IPSec 的管理等。

1 Linux 2.6 内核中 IPSec 支持机构概述

IPSec 主要包括认证头 AH 协议、封装安全载荷 ESP 协议与 Internet 密钥交换 IKE 协议^[2]。AH 协议用于为 IP 提供数据完整性、数据源身份认证和一些可选的、有限的抗重播服务; ESP 协议为 IP 提供机密性、数据源验证以及数据完整性等安全服务; IKE 协议是 IPSec 协议族重要协议之一, 它定义了通信双方进行身

收稿日期: 2006-08-10

基金项目: 江苏省自然科学基金资助项目(BK2004039)

作者简介: 周莉(1980-), 女, 江苏江阴人, 硕士, 助教, 研究方向为计算机网络与网络安全; 陆建德, 教授, 研究方向为计算机网络协议分析设计与网络安全。

份认证、协商加密算法以及生成共享会话密钥的方法。

根据对 Linux 2.6 内核 IPsec 代码的通读与跟踪, 可知, Linux 2.6 内核中的 IPsec 支持机构涵盖并遵循了 RFC2401~RFC2406 中规定的 AH 和 ESP 协议的实现, 在内核算法函数库中提供了对 HMAC-MD5-96、HMAC-SHA1-96、DES-CBC 的支持。由于 IPsec 协议中的 IKE 部分需在应用层实现, 内核中并未涵括, 仅提供了用户空间与内核 IPsec 的 Netlink 套接字通信机构, 为用户态的 IKE 提供了调用接口, 如图 1 所示。

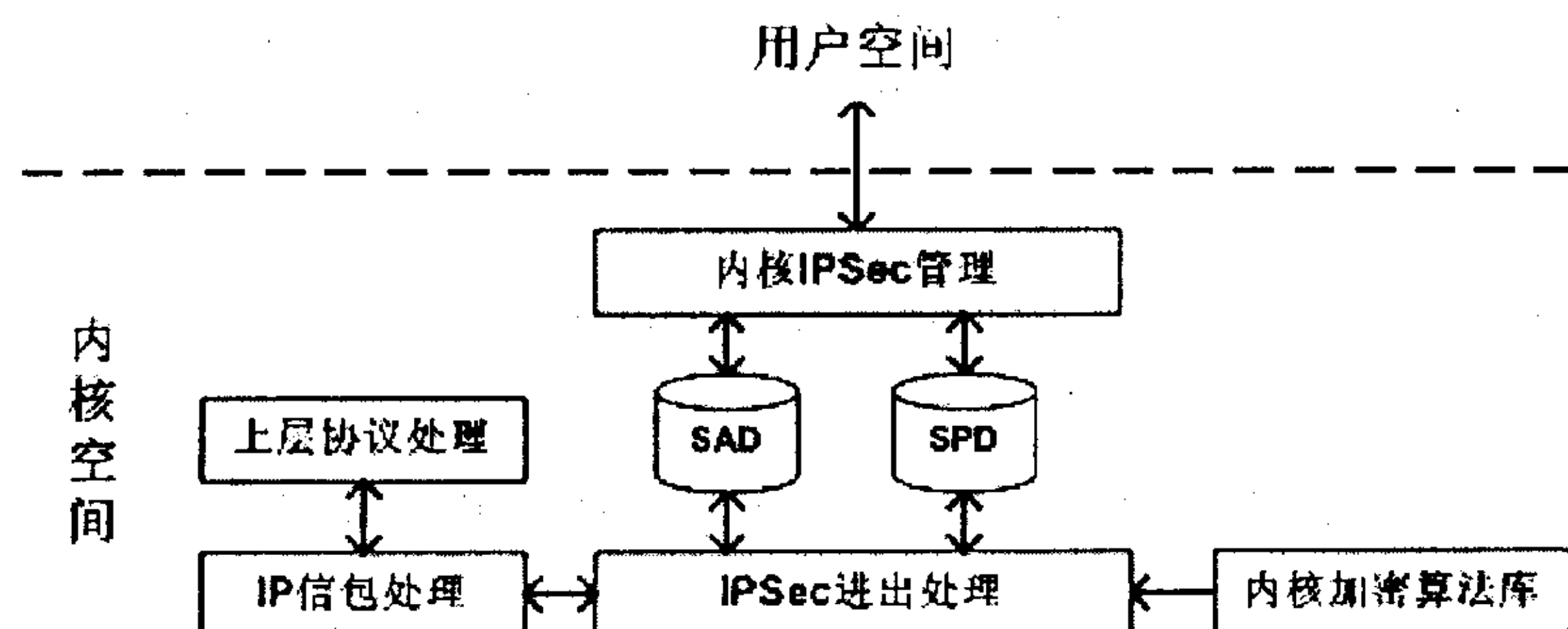


图 1 Linux 2.6 内核中的 IPsec 支持机构

与先前的内核版本相比, Linux 2.6 内核在 IP 层增加了 IPsec 处理的切入点, 经过 IP 层的网络信包如果需要使用 AH 协议或 ESP 协议进行保护, 在相应的切入点都将被重定向到 AH 处理或 ESP 处理模块。

同时, Linux 2.6 内核设计了安全关联 SA 和安全策略 SP 数据结构 (/usr/src/linux/include/net/xfrm.h), 构建了安全关联数据库 (/usr/src/linux/net/xfrm/xfrm_state.c) 和安全策略数据库 (/usr/src/linux/include/net/xfrm_policy.c)。安全策略指示对 IP 信包提供何种保护, 并以何种方式实施保护。安全关联是两个 IPsec 应用实体间的一个单向逻辑链接, 决定保护什么、如何保护以及由谁来保护通信数据。安全关联 SA 是安全策略的具体化和实例化。

为了支持并增强 IPsec 处理所需的加密计算, 内核构建了一个加密算法函数库 (/usr/src/linux/crypto/), 不仅支持 RFC 所要求的 IPsec 加密算法, 还提供了更多安全性更高的加密算法, 如 SHA256、SHA512 等, 这在当前 SHA1 已被破解的情况下尤为重要, 为用户提供了安全强度更大的加密选择。

Linux 2.6 内核在总体构架中还提供了一个内核 IPsec 管理模块 (/usr/src/linux/net/xfrm/xfrm_user.c), 该模块提供了 Netlink 套接字的通信接口, 用于完成 IPsec 处理中用户态进程与内核的通信。通过该接口, 用户空间进程可以实现对内核中的 SAD 和 SPD 的配置, 内核也可以通过该接口触发用户进程与另一个 VPN 实体进行 IKE 交互。

2 内核网络协议栈 IPsec“无缝接入”的分析

相对于原有不提供 IPsec 支持机构的 Linux IP 协议栈, Linux 2.6 内核对 IP 协议栈中一些重要的网络数据结构作出了修改, 在 IP 层增加了 IPsec 处理的切入点, 实现了 IPsec 的“无缝”接入。

2.1 套接字缓冲区(sk_buff)对 IPsec 的支持

sk_buff 是网络处理中一个重要的数据结构, 在协议栈处理中作为收发数据的载体, 协议栈各层次都和该数据结构密切相关^[3]。

当 sk_buff 在协议层流动过程中, 每个协议都对 sk_buff 数据区内容进行修改, 在发送数据时向缓冲区添加自己的协议头和协议尾, 而在接收数据时去掉这些协议头和协议尾^[3]。Linux 2.6 内核中 dst_entry 结构(“目的入口”, 用于记录 IP 信包的路由处理), 提供对 IPsec 处理的支持——dst_entry 不仅可以记录 IP 路由信息, 还可记录查询安全策略数据库 (SPD)

的结果, 文中, 起这样作用的 dst_entry 称为“安全处理目的入口”。“安全处理目的入口”起处理重定向作用。信包外出路由处理时根据 SPD 为信包创建“安全处理目的入口”并将其插在 sk_buff“目的入口”队列中 IP 路由形成的“目的入口”之前。在对 sk_buff 中的“目的入口”队列每个结点依次处理时, 安全处理很自然地被首先执行。“安全处理目的入口”dst_entry 中成员函数 output 指明了具体安全协议处理函数, 成员变量 xfrm 记录了安全处理中将要使用到的安全关联。

一个“安全处理目的入口”可以表达为使用一个安全关联对信包进行一次安全处理。如果保护同一数据流的安全关联不止一个, 即形成一个 SA 束(如先用 ESP 处理, 再用 AH 处理), 就需要多个 dst_entry 类型的“安全处理目的入口”来描述。在内核中, 一个 SA 束表达 dst_entry 结构的链表, 链表中成员通过 dst_entry.child 链接在一起, 如图 2 所示。

```

dst->xfrm->xfrm_state #1
|---. child->dst->xfrm->xfrm_state #2
|---. child->dst->xfrm->xfrm_state #3
|---. child->NULL
  
```

图 2 dst_entry 中记录 SA 束的方法

进入信包的 sk_buff 中 sec_path * 类型成员变量 sp(包含指向 SA 的指针数组)记录了在 IPsec 进入处理过程中, 依次应用于信包处理的 SA 及其相关信息, 用于以后跟策略相比较, 验证已执行的 IPsec 进入处理是否正确。sec_path 及相关结构在 /usr/src/linux/include/net/xfrm.h 中定义。

2.2 外出、转发、进入信包的 IPsec 处理

本机外出数据从应用层流经传输层,到达网络层,在网络层对信包进行外出路由处理时加入了策略查询,并在需要时为信包创建“安全处理目的入口”。

路由处理结束之后,信包经过 NF_IP_LOCAL_OUT,然后调用 dst_output()函数,依次处理 sk_buff.dst 中所有的目的入口。当 dst_output()处理到“安全处理目的入口”时,执行 IPsec 处理、应用安全关联,使信包得到安全协议的保护。

转发的信包是先经过进入处理的外出信包,转发的信包经过进入处理之后,在 ip_rcv_finish()函数中执行路由判断,在经过过滤点 NF_IP_FORWARD,后转给 ip_forward()函数,该函数调用 xfrm4_policy_check()函数检查进入处理中的 IPsec 处理是否正确,然后调用 xfrm4_route_forward()函数查询策略表并将结果记录于 sk_buff 中,在 NF_IP_FORWARD 过滤点,由 dst_output()函数来对信包执行 IPsec 外出保护。

IPsec 对于进入信包处理,主要有两部分工作:一是处理信包中安全协议部分的数据,完成对信包的安全性检查;二是根据策略验证之前的 IPsec 进入处理是否正确。

xfrm4_rcv()函数是 IPsec 进入处理的总控函数。该函数根据具体的安全协议,调用相应的协议处理函数,完成对信包中安全协议数据的处理。如果 IP 信包有多个安全协议共同保护,xfrm4_rcv()函数将多次调用具体安全协议的处理函数,完成对信包中所有安全协议的处理。验证通过,则表明这个信包所接受的安全保护就是安全策略中指定的安全保护。

3 IPsec 组件设计

3.1 安全关联 SA 的设计

内核中 SA 的结构 xfrm_state 定义于 /usr/src/linux/include/net/xfrm.h,主要的成员域包括:安全关联的索引(<安全参数索引,IP 目的地址,安全协议>三元组^[4])、源地址、选择符 AH 认证算法和所需要的密钥、ESP 认证算法及密钥、加密算法及密钥、生存期的设定以及生存期的使用情况、安全协议使用模式(传输模式或通道模式)、抗重播服务组件(序列号计数器和抗重播窗口)、安全协议的私有信息。

选择符是通信的一部分参数组成的一个结构,用以提供安全策略数据库中的索引^[5]。SA 中记录的选择符,用于对 IPsec 进入处理校验部分。在 IPsec 进入处理中,首先调用 SA 对信包进行 IPsec 处理,然后根据 SA 的选择符查询安全策略数据库,找到对应安全

策略,根据策略判断之前对信包执行的 IPsec 进入处理是否正确。

安全关联用 xfrm_algo 类型结构来记录安全处理所需的算法和密钥,xfrm_algo 结构在 /usr/src/linux/include/linux/xfrm.h 中定义。

生存期是 SA 能够存在的时间。超过这个时间,那个 SA 就不可再继续使用。为避免在 SA 过期后造成通信停顿,采用两种类型生存期——软生存期和硬生存期。所谓“软生存期”,是指用它来警告 SA 马上就要到期了;所谓“硬生存期”,是指用它来标识 SA 已经过期,不可再用。生存期有了软硬之分,可利用软生存期和硬生存期之间的时间差来及时协商好一个新 SA。

SA 中协议私有数据包括安全协议头长度、安全协议尾长度、安全协议的处理函数、安全协议等数据。在安全关联中使用 xfrm_type * 类型的成员 type 来指向具体安全协议的处理函数,使用 void * 类型 data 来记录具体安全协议的私有数据,SA 中的 header_len 表示安全协议头长度,trailer_len 表示安全协议尾空间长度。这些私有数据都在 SA 建立初始化时被赋值。

3.2 安全关联数据库 SAD 的构建

Linux 2.6 内核用 SPI 哈希表和目的地址哈希表管理所有 SA 记录,组成安全关联数据库 SAD,这两张哈希表定义在 /usr/src/linux/net/xfrm/xfrm_state.c 中,都是 list_head 类型的静态数组,数组中的每一个成员对应一个内核链表的表头。数组大小 XFRM_DST_HSIZE 在 /usr/src/linux/include/net/xfrm.h 被定义为 1024。数组定义如下:

```
static struct list_head xfrm_state_bydst[XFRM_DST_HSIZE];
static struct list_head xfrm_state_byspi[XFRM_DST_HSIZE];
```

将 SA 加入到 SAD 中可以通过对 <SPI,安全协议,目的地址>进行哈希运算,也可以仅通过对目的地址做哈希计算,得到相应哈希值,将该哈希值作为相对于以上相应数组首地址的偏移,根据偏移可以决定将这个 SA 组织到相应哈希表的哪一个链表中。

3.3 安全策略 SP 的设计

Linux 2.6 内核中安全策略 xfrm_policy 定义于 /usr/src/linux/include/net/xfrm.h,安全策略主要的成员包括:选择符、策略行为、生存期设置及其使用情况、信包应用了安全策略之后的“目的入口”、安全关联模板(用于记录安全策略使用到的安全关联的信息)。

通信处理到 SPD 中策略的映射关系是由“选择符 Selector”来建立的,选择符由通信的一部分参数来标识,根据选择符来匹配某一策略用于当前通信处理。

Linux 2.6 内核中安全策略选择符的数据结构为 xfrm_selector (定义于 /usr/src/linux/include/linux/

xfrm.h), 主要包括策略的目的地址、策略的源地址、策略的目的端口、策略的源端口、地址域、目的 IP 地址前缀、源 IP 地址前缀、上层协议值、网络接口索引号、用户 ID。

在 Linux 2.6 内核中, IPSec 处理为安全策略定义了三种行为:应用 IPSec 安全服务、丢弃和绕过。

3.4 安全策略库 SPD 的构建

Linux 2.6 内核中定义了一组双向链表作为 SPD 来统一管理安全策略, 链表表头放在 xfrm_policy_list 数组 (/usr/src/linux/net/xfrm/xfrm_policy.c) 中。所有的安全策略都通过 xfrm_policy.next 链接到 SPD, 应用方向相同的安全策略链接在同一个链表中。内核中负责进入处理安全策略链接到 xfrm_policy_list[XFRM_POLICY_IN], 负责外出处理的安全策略链接到 xfrm_policy_list[XFRM_POLICY_OUT]。

4 内核 IPSec 管理模块

4.1 通信接口——Netlink 套接字

Linux 2.6 内核 IPSec 管理模块使用 Netlink 套接字来实现用户空间与内核空间的交互。内核的 Netlink 套接字相当于服务端, 套接字建立后处于等待状态, 等待用户空间 Netlink 套接字的连接请求; 用户进程通过用户 Netlink 套接字与内核 Netlink 套接字交互信息。

Linux 2.6 内核 Netlink 套接字由 netlink_kernel_create() 函数 (/usr/src/linux/net/netlink/af_netlink.c) 创建, 创建同时需要指明数据接收处理函数。

netlink_kernel_create(套接字协议, 消息处理函数);

用户 Netlink 套接字的创建方法和一般套接字一样, 创建方法如下:

```
int sockfd = socket(PF_NETLINK, SOCK_RAW, protocol); 或
int sockfd = socket(PF_NETLINK, SOCK_DGRAM, protocol);
```

Netlink 套接字可接收的协议在 /usr/src/linux/include/linux/netlink.h 中有定义。

内核态和用户态的 Netlink 套接字创建完成, 并且用户态 Netlink 套接字绑定地址之后, 内核空间和用户空间就可通过这两个套接字进行通信。

内核 IPSec 管理模块在模块初始化时 (/usr/src/net/xfrm/xfrm_user.c 中的 xfrm_user_init() 函数) 创建的 Netlink 套接字的协议类型为 XFRM_NETLIKE, 使用 xfrm_netlink_rcv() 函数来处理接收到的用户空间消息。

xfrm_netlink_rcv() 函数不断地检查内核态 IPSec Netlink 套接字接收队列, 如果有消息待处理, 则调用 xfrm_user_rcv_skb() 函数对接收到消息的长度进行合法性检查; 通过合法性检查后, 调用 xfrm_user_rcv_msg() 函数来判断消息的类型, 调用相应的函数完成处理工作。

Netlink 套接字之间通信的消息, 必须以固定格式的消息头 (nlmsghdr 结构, /usr/src/linux/include/linux/netlink.h) 开始, 消息头的数据成员主要包括: 消息总长度、消息类型, 标识, 发送消息的进程 ID、序列号。其中消息类型是和 Netlink 套接字的具体协议相关的, 不同协议定义有不同的消息类型。

内核 IPSec 管理模块接收的, 由用户进程发起的消息有: 新建 SA、删除 SA、查询 SA、新建安全策略、删除安全策略、查询安全策略、获取 SPI 更新安全策略、更新 SA。

内核 IPSec 管理模块发现 SA 过期、安全策略过期这些信息, 也会及时地向用户空间发送相应的消息, 以说明有问题待处理, 由内核 IPSec 管理模块主动发起的消息如下: 要求建立 SA、SA 超期、安全策略超期。

4.2 消息格式

用户进程发送向内核 IPSec 管理模块交互的消息及消息格式如表 1 所示, 其中各结构体的详细定义见 /usr/src/linux/include/net/xfrm.h, 消息的类型由 Netlink 套接字消息头中的消息类型变量指定。

表 1 交互消息格式

消息发起方	消息类型	消息格式
用户进程	新建 SA 更新 SA	Netlink 套接字消息头 (struct nlmsghdr) + 安全关联信息 (struct xfrm_sa_info) + 算法信息 (struct rtattr, struct xfrm_algo)
	删除 SA 查询 SA	Netlink 套接字消息头 (struct nlmsghdr) + SA 索引信息 (struct xfrm_usersa_id)
	获取 SPI	Netlink 套接字消息头 (struct nlmsghdr) + SPI 信息 (struct xfrm_userspi_info)
	导出 SAD	Netlink 套接字消息头 (struct nlmsghdr)
	新建 SP 更新 SP	Netlink 套接字消息头 (struct nlmsghdr) + 安全策略信息 (struct xfrm_userpolicy_info) + SA 模板信息 (struct rtattr + struct xfrm_user_tmpl)
	删除 SP 查询 SP	Netlink 套接字消息头 (struct nlmsghdr) + SA 索引信息 (struct xfrm_userpolicy_id)
	导出 SAD	Netlink 套接字消息头 (struct nlmsghdr)
内核 IPSec 管理进程	新建 SA	Netlink 套接字消息头 (struct nlmsghdr) + 内核要求建立的 SA 信息 (struct xfrm_user_acquire)
	SA 超期消息	Netlink 套接字消息头 (struct nlmsghdr) + SA 超期消息 (struct xfrm_user_expire)
	SP 超期消息	Netlink 套接字消息头 (struct nlmsghdr) + SP 超期消息 (struct xfrm_user_polexpire)

(下转第 198 页)

及如何通过 Internet 传输消息等内容;UDDI 建立在 WSDL 基

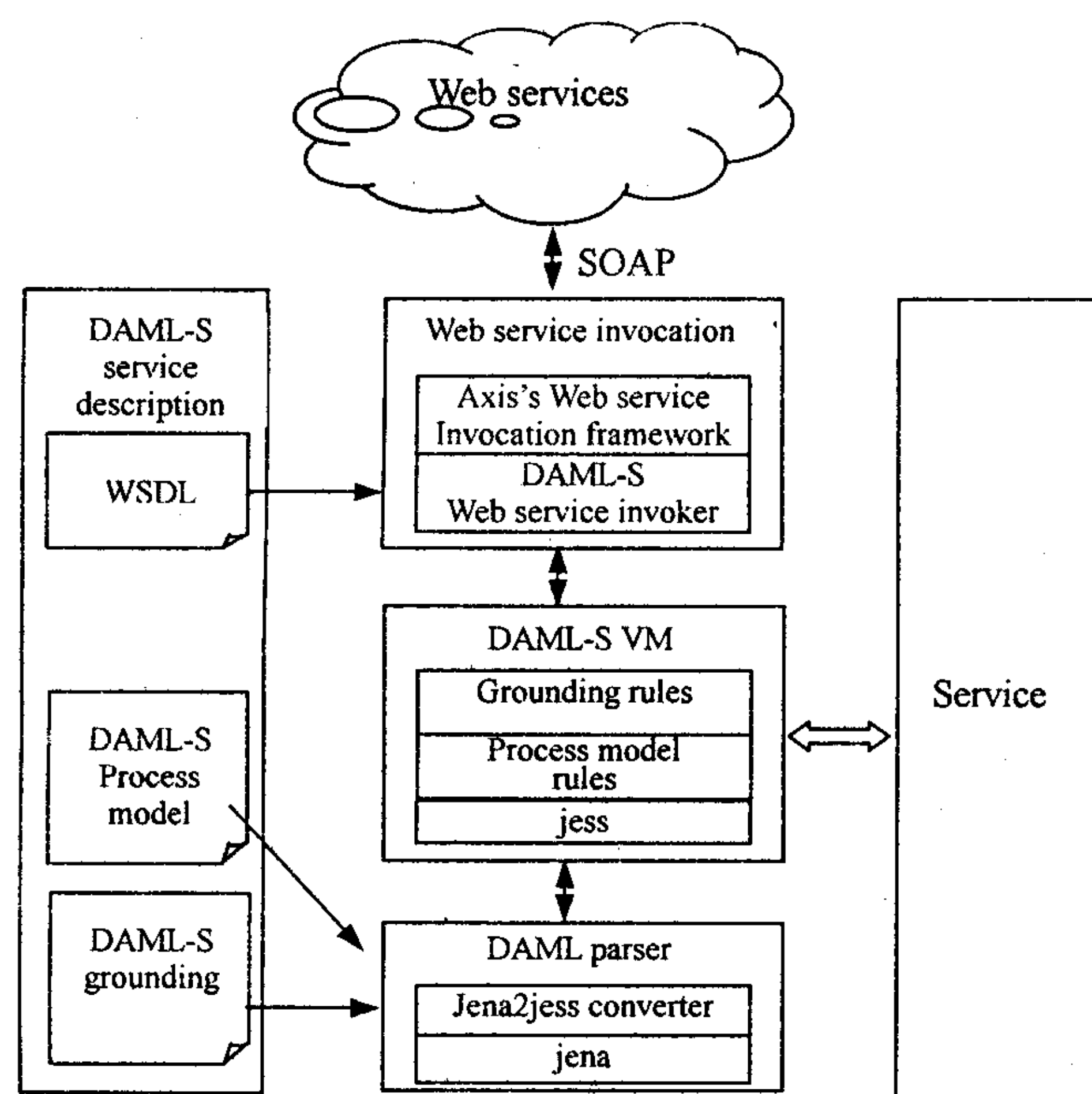


图 4 基于 OWL-S 的 Web 服务调用框架

础之上,把 WSDL 描述的信息通过一个注册中心保存起来,为服务提供者和服务请求者提供中介。这些标准以 XML 为基础,解决了 Internet 环境中信息交互的异构问题。但是目前这种交互模式多多少少需要人的参与,未来的 Web 发展方向是服务交互的自动化和智能化,OWL-S 的引入是实现 Web 服务的自动发现、动态调用和组装的重要尝试,但同时也给已获得广泛应用的传统的 Web 服务体系结构带来冲击,其中表现在如何对 WSDL 进行语义扩展,增强语义的 WSDL 到 UDDI 的转换,基于服务功能描述而非关键字的 Web 服务查找技术、服务的动态组装等问题。文中给出了

一种利用 OWL-S 新的 Web 服务体系结构,讨论了利用 OWL-S 的 Web 服务发现和调用机制。今后将在基于服务语义的动态查找、基于本体的 Web 服务动态合成方面做进一步研究。

参考文献:

- [1] Christensen E, Curbera F, Meredith G, et al. Web Services Description Language (WSDL) 1.1 [EB/OL]. 2001-03-15. <http://www.w3.org/TR/wsdl>.
- [2] Gudgin M, Hadley M, Mendelsohn N, et al. Simple Object Access Protocol (SOAP) 1.2 [EB/OL]. 2003-06-24. <http://www.w3.org/TR/SOAP12/>.
- [3] Clement L, Hately A, Riegen C V, et al. UDDI Version 3.0 [EB/OL]. 2004-10. <http://www.uddi.org/>.
- [4] Borst W N. Construction of Engineering Ontologies for Knowledge Sharing and Reuse [D]. Enschede: University of Twente, 1997.
- [5] Fensel D. Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce [M]. [s.l.]: Springer, 2001.
- [6] Burstein M H, Hobbs J R, Lassila O, et al. DAML-S: Web Service Description for the Semantic Web [C] // Proc. of the Int'l Semantic Web Conf. Sardinia: Springer-Verlag, 2002: 348-363.
- [7] Martin D, Burstein M, Denker G, et al. DAML Services [EB/OL]. 2006-03. <http://www.daml.org/services/owl-s/>.
- [8] Paolucci M, Kawamura T, Payne T, et al. Importing the Semantic Web in UDDI [C] // Proc. E-Services and the Semantic Web, LNCS 2512. [s.l.]: Springer-Verlag, 2002.
- [9] Sycara K, Paolucci M, Ankolekar A, et al. Automated Discovery, Interaction and Composition of Semantic Web Services [J]. Journal of Web Semantics, 2003, 1(1): 27-46.

(上接第 194 页)

5 结束语

随着网络通信安全化的推进,凭借其自身的优势,IPSec 受到越来越多的关注、实施和推广,最新的 Linux 2.6 内核也对此提供了支持。文中着重分析在最新的 Linux 2.6 中的 IPSec 支持机构,对比以往不支持 IPSec 的网络协议栈,分析出 Linux 2.6 内核“无缝”接入 IPSec 处理的方法;阐述了 IPSec 重要组件——安全关联 SA、安全策略的设计思想以及安全关联数据库 SAD 和安全策略数据库 SPD 的构建方法;分析了内核对 IPSec 进行管理的模块,并总结了 Linux 2.6 内核 IPSec 支持机构提供给用户进程可被调用的管理功能。文中的分析工作都是基于研读内核源代码,结合 IPSec 原理,整理并总结出来的,对于利用 Linux 内核

IPSec 作为基础设施欲作进一步开发的 VPN 开发者具有参考价值。

参考文献:

- [1] Davis C R. IPSec: VPN 的安全实施 [M]. 北京:清华大学出版社, 2002.
- [2] Kent S, Atkinson R. Security Architecture for the Internet Protocol [S]. RFC 2401. 1998.
- [3] 陈莉君. Linux 操作系统内核分析 [M]. 北京:人民邮电出版社, 2000.
- [4] Harkins D, Carrel D. The Internet Key Exchange (IKE) [S]. RFC 2409. 1998.
- [5] Doraswamy N, Harkins D. IPSec 新一代因特网安全标准 [M]. 北京:机械工业出版社, 2000.