文章编号:1001 - 9081(2007)05 - 1113 - 03

多操作系统平台可移植 Mobile IPv6 协议栈设计与实现

陈海峰[1,2],张大方[1],李 军[2],蒋 海[2],王 嘉[2,3]
(1. 湖南大学 软件学院,湖南 长沙 410082; 2. 中国科学院 计算技术研究所,北京 100080;
3. 湖南大学 计算机与通信学院,湖南 长沙 410082)
(chenhaifeng@ict.ac.cn)

摘 要:阐述了在多种操作系统平台上可移植的 Mobile IPv6 协议栈设计与实现方法。给出了实现的详细过程,并基于模块划分的策略提出了一种简化的移植方法,将可通用代码和非可通用代码分离并将其模块化,移植过程被简化为非可通用模块的替换。同时简单介绍了该协议栈中网络安全的实现方法。对于其他互联网协议以及移动设备的开发具有一定的参考意义。

关键词:IPv6;移动 IPv6;可移植移动 IPv6 协议栈
中图分类号: TP393.04 **文献标识码**:A

Design and implementation of portable Mobile IPv6 protocol stack

CHEN Hai-feng[1,2], ZHANG Da-fang[1], LI Jun[2], JIANG Hai[2], WANG Jia[2,3]
(1. *School of Software, Hunan University, Changsha Hunan* 410082, *China;*
2. *Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100080, *China;*
3. *School of Computer and Communication, Hunan University, Changsha Hunan* 410082, *China*)

Abstract: Design and implementation of a portable Mobile IPv6 protocol stack (MultiPlat-MIPv6) is presented in detail. A simple transplanting method of Mobile IPv6 protocol stack was brought up, in witch Mobile IPv6 protocol stack includes the universal and the particular modules. Transplanting is simplified to change the particular modules. The security mechanism in this protocol stack is also given simply. It is helpful for the development of network protocol stacks and mobile network devices on different operating system platform.

Key words: IPv6; Mobile IPv6; portable Mobile IPv6 protocol stack

0 引言

随着人们对移动上网的需求增长,Internet 中无线移动设备将会占据越来越重要的地位。IPv6 是下一代互联网核心协议,目前各项基础设施正在大规模建设中,移动 IPv6 协议(Mobile IPv6,MIPv6)基于 IPv6 巨大地址空间,支持任意两点端到端通信,拥有完善的路由优化机制。与移动 IPv4(Mobile IPv4)协议相比,更适应移动 Internet 应用,有利于大规模移动网络部署。移动设备互联网设备品种繁多,所采用的操作系统也千差万别,传统的移动网络协议栈都是根据不同操作系统平台分别设计,这样重复工作巨大,不利于产品的开发以及设备软件的升级维护。因此开发多操作系统平台下可移植的 MIPv6 协议栈是具有现实意义的工作。本文介绍一种采用模块划分策略设计实现多操作系统平台上可移植的 MIPv6 协议栈的方法。

1 MIPv6 协议

MIPv6 协议是在 IPv6 中支持移动性的网络协议,它将参与通信的所有节点划分为 3 种角色:移动节点(Mobile Node,MN),接入点可在不同链路中切换的节点; 家乡代理(Home Agent,HA),MN 注册的网络中的固定节点,它负责记录 MN 当前位置并为不在注册链路的 MN 转发报文; 对端节点(Correspondent Node,CN),与 MN 通信的节点,可以是固定节点,同时也可以是移动节点。

MIPv6 首先保证 MN 发生移动后 MN 和 CN 之间的通信连接的延续性,即当 MN 移动后 MN 和 CN 之间的通信不被中断,在此基础上提高两者之间的通信效率。MN 在家乡链路时使用 HoA(家乡地址)与所有的 CN 通信,这种方式与传统的 IP 通信方式没有区别;MN 移动到外地链路获得 CoA(转交地址),通过家乡注册通知 HA 自己当前的位置,这时 MN 与 CN 之间所有通信使用 HA 中转,HA 与 MN 之间采用隧道/反向隧道传递之间的 IP 报文。为了提高 MN 和 CN 之间的通信效率,MN 通过 CN 注册过程通知 CN 自己的当前位置,并与之建立路由优化关系。一旦路由优化关系建立,CN 与 MN 之间采取直接方式通信。为实现 CN 与 MN 之间直接通信,MIPv6 定义了新的路由扩展头(Type2 路由扩展头)以及家乡地址选项(Home Address Option, HAO):Type2 路由扩展头携带在 CN 发向 MN 的 IP 包文中,MN 根据 Type2 路由扩展头中携带的自己的 HoA 替换 IP 包目的地址还原 IP 报文;家乡地址选项由 MN 发向 CN 的 IP 报文的目的地选项头携带,CN 根据该头中携带的 MN 的 HoA 替换 IP 包中源地址还原 IP 报文。MIPv6 以上的网络协议始终采用 MN 的 HoA 作为节点地址与其他节点通信,IP 及以上协议层感觉不到 MN 的移动过程以及 MIPv6 对 IP 包的处理过程,由此保持了 MN 移动环境下与 CN 之间网络传输的延续性。

总的来说,MIPv6 协议完成两个主要功能:

收稿日期:2006 - 12 - 06;**修订日期**:2007 - 02 - 09 **基金项目**:中国科学院计算技术研究所知识创新资金资助项目

作者简介:陈海峰(1975 -),男,硕士研究生,主要研究方向:下一代互联网技术; 张大方,男,教授,博士生导师,主要研究方向:网络测试技术; 李军,男,博士,主要研究方向:下一代互联网技术; 蒋海,男,博士研究生,主要研究方向 计算机网络技术; 王嘉,女,硕士研究生,主要研究方向:计算机通信通信.

1)移动相关功能,MN 发现自己的移动并获得新地址通知 HA 和 CN,设定好 MN 和 CN 之间的通信状态数据,并定期地维护这些数据;

2)数据接收和发送功能,当 MN 移动状态已经确定,MN 和 CN 之间接收和发送数据的处理,即根据目前移动状态接收和发送携带移动相关信息(Type2 路由扩展头,家乡地址选项)的 IP 报文。

2 MultiPlat-MIPv6 结构设计

多平台可移植 MIPv6 协议栈(MultiPlat-MIPv6)协议栈设计的主要目标是无论操作系统平台如何变化,MultiPlat-MIPv6 协议栈尽可能少地修改代码,并且采用模块化替换方式简化平台移植过程。为此,需要区分开 MultiPlat-MIPv6 协议栈所包含的平台相关与平台无关两类代码:平台相关代码与具体的操作系统或硬件相关联,不能脱离操作系统具体特性;平台无关代码不直接与操作系统或硬件相关联,不依赖于具体的操作系统。区分出这两部分代码才能在设计中确定不同平台下保持通用的代码(平台无关代码),和必须重新编写的代码(平台相关代码); MultiPlat-MIPv6 协议中哪些功能可用平台无关代码实现,哪些功能必须采用平台相关代码实现是设计的重点工作,需要根据操作系统以及 MIPv6 相关特点综合考虑。与 MIPv6 协议的功能相对应,MultiPlat-MIPv6 协议栈包括两个主要的功能模块:

1) 移动相关功能模块识别 MN 当前所在网络,完成网络切换,通知 HA、CN,MN 的当前位置,建立 MN 和 CN 之间的路由优化关系以及维护移动相关状态信息,它是一些逻辑过程的集合,与操作系统以及硬件无直接联系,通过合适的设计方法,可作为平台无关部分在用户态以服务程序的方式实现;

2) 数据接收和发送功能模块分别处理路由优化前后的数据接收与发送,由于这部分直接相联系的网络协议程序是在操作系统内核运行,并需要和硬件驱动程序相关接口通信,因此这部分必须运行在内核态必定是平台相关的。

根据对以上分析我们将 MultiPlat-MIPv6 协议栈设计成为 3 个主要的结构部分:移动相关功能模块与平台无关称之为通用功能体工作在用户态;数据接收和发送功能模块与平台相关称之为内核功能体工作在内核态;另外需要在内核态与用户态程序之间建立一个中间层,该层与移动协议规定的功能没有直接的关系,但包括一系列内核态和用户态之间通信接口函数、通用线程管理函数、定时函数。这些函数为通用功能体对不同操作系统函数的调用保持统一函数名和参数格式,用以屏蔽平台差异,称之为适配层。图 1 是以 Linux、Windows 为例给出 MultiPlat-MIPv6 体系结构。

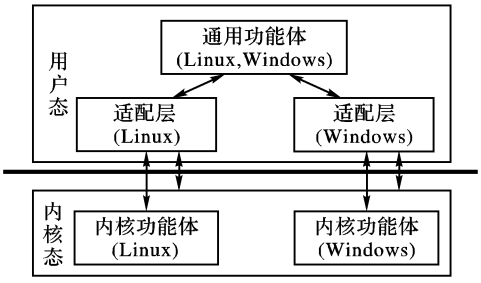

图 1 MultPlat-MIPv6 体系结构

如图 1 所示通用功能体在不同的操作系统下保持了通用性(源代码通用性),其他两部分需要根据操作系统的不同而做适应性修改。该体系结构划分出通用代码和不通用代码,并将其模块化,操作系统平台间移植过程被简化为部分模块替换。这是 MultiPlat-MIPv6 的主要特点。

3 MultiPlat-MIPv6 实现

MultiPlat-MIPv6 三个部分的实现都采用大多数操作系统编译器支持的标准 C 语言编写,这样可以保持语言的统一性。

通用功能体是协议栈的主要部分,它包含协议栈中绝大多数的代码,该部分源代码禁止直接调用基于平台的系统函数,采用统一的适配层接口函数调用操作系统相关功能。它是在 MultiPlat-MIPv6 平台移植过程中保持不变的部分。

内核功能体是移植过程的重要部分,移植的主要工作集中在重新编写这一部分的代码模块,它的实现需要根据实际情况选择具体的实现方式,我们选择模块安装的方法插入操作系统核心。不建议修改操作系统内核源代码,如修改操作系统源代码,操作系统升级可能产生新的移植工作,而且大多数的操作系统为非开源系统,修改源代码并不可行。

适配层起到连接通用功能体与内核功能体以及操作系统的桥梁作用,它直接关系到通用功能体通用性。适配层向通用功能体提供统一的通信接口函数、线程管理函数以及定时函数,避免通用功能体直接调用形式各异的操作系统 API 函数。

3.1 通用功能体实现

通用功能体集中了 MIPv6 协议大多数逻辑处理过程,根据移动相关控制命令报文,触发操作过程,并维护移动相关数据,是移动切换,路由优化,移动状态数据产生和维护的管理过程。通用功能体是 MultiPlat-MIPv6 的核心部分,主要完成以下三个方面的功能:移动设备发生移动时根据相关网络信息发现新的链路,在新的链路中注册得到 CoA 地址,通知 HA 和 CN 并进行绑定,在 MN 和 CN 之间建立路由优化关系;产生并定期维护移动状态信息(BC 和 BUL 表);将最新的 BC 和 BUL 表信息通知内核功能体,保持准确的接收和发送状态。

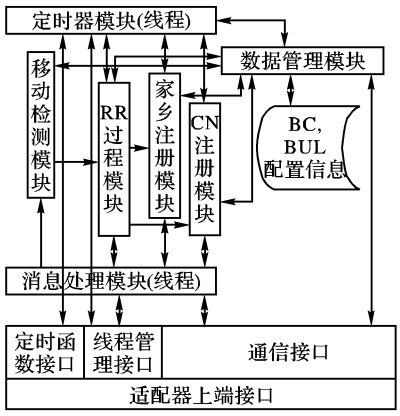

图 2 通用功能体结构

通用功能体启动消息处理线程和定时器线程两个关键线程。消息处理线程是一个循环监听过程,通过适配层的通信接口得到 MIPv6 需要处理的 MH 报文(MH 报文是 MIPv6 定义的移动相关命令报文)以及 ICMP 报文,根据具体报文类型和参数分别启动移动切换、RR 过程、家乡注册、CN 注册等过程、创建并管理 BC、BUL 表项;定时器线程根据定时器链表定期维护 BC、BUL 表,调度其它需要定时操作的过程。

MultiPlat-MIPv6 不采用 RAW SOCKET 机制截获 MIPv6 需要处理的 MH 报文以及 ICMP 报文,而是采用内核功能体从 NIC 驱动程序上端直接截获将其存入缓冲区,并通过适配层函数向通用功能体传送的方法实现,这样做的原因是不同

操作系统 RAW SOKET 机制存在很多差异，不利于通用功能体代码通用性，采取自己截获的方法易于统一接口。

通用功能体代码在 MultiPlat-MIPv6 中占绝大多数，是协议栈中最复杂的处理过程的集合。由于通用功能体需要启动消息处理线程和定时线程进行管理，并由内核截获的网络控制报文驱动相关处理。这些过程必定调用操作系统相关函数（包括线程管理函数、定时函数以及内核通信函数），这些函数因操作系统不同而不同，为了使通用功能体程序代码保持通用性，我们不直接调用系统相关函数。适配层为其定义一套类似的调用函数，这些函数对通用功能体保持统一的函数名称和参数格式，以此屏蔽平台差异。

3.2　内核功能体实现

内核功能体与操作系统直接相关，需要根据操作系统具体情况选择合适的实现方式，内核功能体截获上下行的 IP 数据报文，为通用功能体提供网络控制消息，并转发通用功能体发出的控制报文；同时也是数据报文处理者，负责添加、删除报文的移动信息（Type2 路由扩展头，家乡地址选项），封装解封装 IP 隧道，向 IPv6 协议栈提交还原的普通 IP 数据报文、发送添加了移动信息的 IP 数据报文。

图 3 是内核功能体结构。

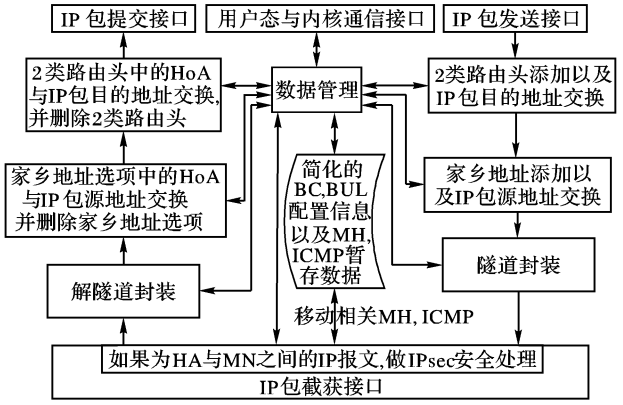

图 3　内核功能体结构

内核功能体与操作系统平台紧密相关。它的实现需要解决两个问题：（1）IP 报文截获机制；（2）内核态与用户态程序之间通信机制。这里分别以 Linux 和 Windows 操作系统平台为例介绍内核功能体的实现：Linux 平台内核功能体实现采用 Net_filter、NetLink 两种机制，前者是网卡与网络驱动程序之间编程接口、后者是用户态程序与核心态程序通信的编程接口。利用 Net_filter 机制直接截获 IPv6 网络协议程序与网卡驱动之间的上下行 IP 报文，实现 IP 报文截获。在 Net_filter 内部使用 NetLink 的通信接口与通用功能体之间进行数据通信；Windows 平台内核功能体的实现类似于 Linux，Windows 的 NDIS（Network Driver Interface Specification）机制[2]（网络驱动程序接口规范）和 WDM（WIN32 DRIVER MODEL）机制[3]（设备驱动编程模型）分别提供了 Windows 平台下截获 IPv6 网络协议层与网卡驱动之间上下行 IP 报文以及内核功能体与通用功能体之间的数据通信机制，在 NDIS 中嵌入 WDM 结构完成内核功能体功能。其他操作系统，内核功能体的实现需要根据实际情况选择合适的机制。

虽然内核功能体根据不同的操作系统需要分别编写，但它内部的基本功能模块相同，将其再次划分，区分出逻辑处理模块和非逻辑处理模块（内核结构性模块），将逻辑处理模块统一编写成为一个函数调用库，采用比较好的封装方法使得这些小的功能模块在不同的操作系统内核程序中可以被调用，不需要再重新修改。内核功能体的编写工作关键是根据操作系统平台的不同将这些小逻辑模块依据具体操作系统内核结构特点合理的组合，使其共同完成内核功能体功能。

3.3　适配层实现

通用功能体是否能保持源代码通用性，关键在于适配层的封装质量，适配层向通用功能体提供统一接口的相关函数。该层中包括内核态与用户态之间的通信接口函数、线程管理函数、定时函数：

1）通用功能体调用适配层通信接口监听并读取 MH、ICMP 报文，其反馈的 MH 和 ICMP 报文通过通信接口通知内核功能体转发到移动网络。通用功能体管理的相关状态数据需要依靠通信接口即时地通知内核功能体数据管理模块，保持状态数据信息同步；

2）通用功能体调用线程管理函数管理控制线程，不同系统的线程管理函数各不相同，但线程管理函数的功能基本类似，包括线程产生，关闭，互斥等基本操作。为保持通用功能体代码的通用性，适配层为其提供统一函数名称和参数格式的线程管理函数，内部使用预编译方法区别不同操作系统下的具体的函数实现；

3）通用功能体需要定期的维护移动相关信息，微秒到毫秒级的定时函数由操作系统提供，适配层同样采用预编译的方法统一封装操作系统定时函数。

4　安全机制实现

移动 IPv6 安全[4]包括三个方面：HA 注册过程及 CN 注册过程的安全保障；MN 与 CN 之间的安全保障；HA 与 MN 之间的安全保障。

1）根据 RFC3775 规定 HA 注册过程及 CN 注册过程的安全保障由 RR 过程来完成，在通用功能体中得以实现；

2）MN 与 CN 之间的安全保障，可以采用通用 IPsec 机制实现，这种机制实现在 MIPv6 以上的网络层次，这里不做特殊处理，值得注意的是在 IPsec 配置时需要针对 MH 的 HoA 地址配置；

3）HA 与 MN 之间的安全是必须得到保障的，选择在内核功能体中实现，采用手动方式配置安全策略，在图 2 的 IP 包截获接口，监督上下行的数据包，发现与 HA 通信的 IP 报文，根据 HA 和 MN 之间的 IPsec 策略配置调用安全模块封装解封装 IP 数据来确保安全传输。

5　结语

本文阐述了一种多平台可移植的移动 IPv6 协议栈（MN、CN）的设计和实现，采用平台无关和平台相关代码分离方法，将协议栈划分为通用功能体、内核功能体、适配层三个主要部分，这样的体系结构划分出平台移植过程中通用代码和不通用代码，并将其模块化，移植过程被简化为替换相关结构模块，这样大大减少了移植工作量，保证了协议栈大部分代码的通用性。

参考文献：

[1] JOHNSO D, PERKINS C, ARKKO J. IETFRFC3775, Mobility Support in IPv6[S].

[2] 朱雁辉. Windows 防火墙与网络封包截获技术[M]. 北京：电子工业出版社，2002.

[3] CANT C. Writing Windows WDM Device Drivers[M]. 北京：机械工业出版社，2001.

[4] ARKKO J, DEVARAPALLI V. IETF RFC 3776. Using IPsec to Protect Mobile IPv6 Signaling Between Mobile Nodes and Home Agents[S].