主动网络试验模型的研究*)

The Research of the Experiment Model of Active Network

王汝传　王绍棣　孙知信　张　莉

(南京邮电学院计算机科学与技术系　南京 210003)

Abstract　Traditional Internet architecture is lack of flexibility and process of protocol standardization and compatibil-ity is time-consuming and costly. Active network (AN) is one of the approaches to solve these technical problems. AN uses open intermediate nodes that make the network to be programmable. Active codes of the message create new network service dynamically and this makes the network architecture more flexible. This paper gives an AN experimental modal and this modal is simulated and analysed systematically.

Keywords　Network architecture, Programmable network, Active network(AN)

1. 引言

当今网络技术正在飞速发展,Internet 取得了巨大成功,近十年来网络新技术虽然层出不穷,但是相应的网络灵活性较差,协议的标准化和兼容性成为网络技术发展的主要障碍。在传统的网络体系结构中,网络层和传输层是整个网络变化的瓶颈部分,这是因为网络层和传输层是不同物理网络之间进行互联的基础,也是不同系统应用层之间进行互通的基础,因此只有保持网络层和传输层相对稳定才能提供一个稳定的网络服务,在网络层和传输层每一点变化都要经历漫长的标准化过程。这个过程往往要持续数年,并且它的改动大部分要经过手工配置才能最终投入运行。另外,对于网络层和传输层的改动往往还带来一些其他问题,例如向下兼容的问题和大量的配置工作。

近年来,通信技术和计算技术的发展,推动着 Internet 规模不断扩大,促进了网络应用发展。例如,从传统传输单一正文数据发展到传输语音和视频数据,从传统的点到点交互发展到单点到多点交互和多点到多点交互等。技术应用发展同时,也暴露出当前网络技术的缺陷,其主要表现为:高速传输的信道和低速的协议处理软件之间的矛盾;多种网络需求与单一网络服务之间的矛盾;动态应用需求与静态网络服务提供之间的矛盾。由于上述矛盾存在,有必要进行新一代网络体系结构的研究,使得网络具有主动性、安全性、实时性和可扩展性。为了解决网络灵活性的问题,在传统网络模型中增加计算结构使得网络具有可编程思想,主动网络就是其中之一。主动网络是由一组称为中间节点的网络节点组成,中间节点可以是路由器或者交换机,这些主动节点共同构成了主动网络的执行环境。

2. 主动网络模型的结构

主动网络是通过主动报文动态扩展了网络服务,使网络更加灵活,网络节点同时支持不同类型的网络协议,主动网络中一个新协议的构建,不需要由统一认证进行标准化,只要与协议相关各方支持即可,主动网络的体系支持协议动态升级,不需要网络部件离线操作,我们提出一个以带内传输和带外传输相结合的机制实现主动网络模型体系结构,如图 1 所示。

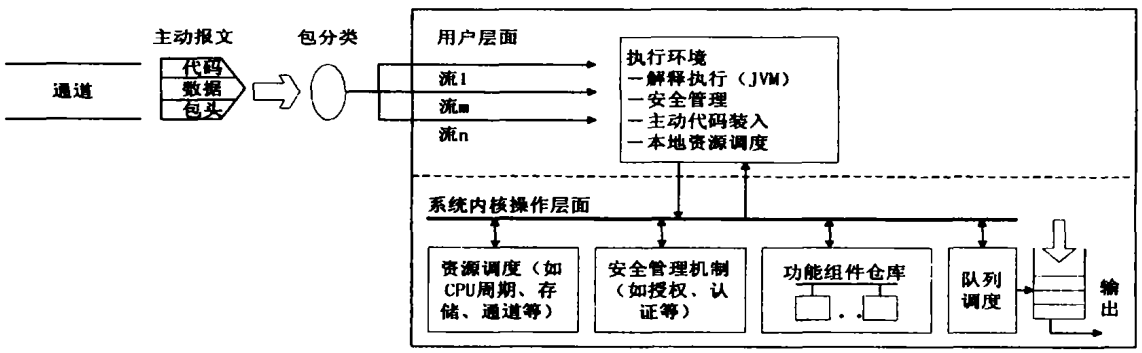

图 1　系统节点体系结构图

该体系结构的主要特点是:

(1)用主动报文代替传统的数据包实现主动网络的可编程性,主动报文可以定制自己的处理程序;

(2)主动节点代替传统的路由器,主动节点执行主动报文的定制程序,进行安全检查、资源分配并维护相关状态;

(3)系统中传统路由器和主动路由器并存,实现向下兼容性;

(4)系统采用带内传输和带外传输相结合的方法,保证了协议动态自动开发和升级;

从系统的角度看,主动网络可以抽象为两个层次,一个是

*)本文得到国家自然科学基金(60173037)和江苏自然科学基金(bk2001123)资助。王汝传　教授,主要研究方向为计算机软件,计算机网络,电子商务及网络安全等。王绍棣　教授,主要研究方向为计算机通信网及信息安全。孙知信　副教授,博士,主要研究方向为计算机软件和计算机通信。张　莉　硕士研究生,主要研究方向为计算机网络安全。

"硬件"组件抽象，包括处理器、存储和通道等硬件资源，它是整个网络运行的基础；另外一个层次是"软件"组件抽象，它提供主动服务，包括资源调度、安全管理和路由选择等，用户定义的主动应用运行在用户层面上，然后由主动节点提供的执行环境解释执行。本系统的设计主要是"软件"组件的抽象。实现上，本系统采用了基于 Red Hat Linux 6.0 的 IP 路由器，在其上附加主动引擎，形成主动节点，主动引擎的结构和功能实现是设计的核心。另外，主动网络可编程性主要是通过主动报文在主动节点的执行实现的，因此设计中主动报文的格式，主动代码的分发机制也是需要考虑的问题。

2.1 主动报文

本系统的主动报文采用 IETF 定义的主动报文封装协议 ANEP 进行封装，通过扩展 Option 域的功能，增加主动网络的安全性能。报文传输通过在 UDP 报文上封装 ANEP 实现主动报文在 UDP 通道上的传输。流经主动节点的报文有普通的数据报文和主动报文两种。根据处理方式的不同，我们将主动报文又分为立即执行报文和主动应用报文两种。主动路由器对数据报文的处理和传统的路由器相同，对主动报文的处理，根据应用的不同，可能为执行定制程序，可能是按报文中定制的方法进行路由转发。各种报文在主动节点的处理过程如图 2 所示。

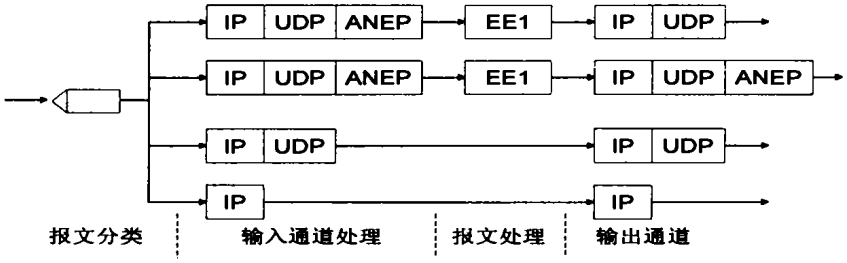

图 2 经过主动节点的数据报文处理图

2.1.1 *主动应用报文* 是为某个应用(协议)定制的主动报文，一个主动应用(协议)的建立，采用的是三层体系结构如图 3 所示。

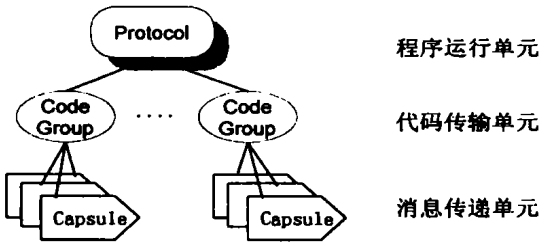

图 3 主动协议的三层体系结构

在三层结构中，主动应用报文(Capsule)是主动应用(协议)的最小组成单位，是与主动节点交互的单元；代码组是一组相关主动报文的集合，在代码分发机制中，它们作为同一个单元来传送；协议是一组相关代码组的集合，这些代码组构成一个完整的应用；在主动节点中协议被看作一个单元来保护和执行。协议由统一的协议号来标识，协议号通过 MD5 加密，大大减少了协议哄骗的发生。主动应用报文的代码采用"按需分配"的分发机制，假设首先一个主动协议的代码存储在源节点上，主动报文本身不携带主动代码，当一个主动报文到达主动节点并调用代码执行协议时，如果发现所需的代码不在该节点上，就调用代码分发机制，向前一个节点申请，装入所需的代码，这时执行被挂起，直到在限定的时间内所需的代码到达该节点。如果代码在限定时间内没有到达，该执行将被放弃。这种方法类似于离散主动网络的代码传输方法。

2.1.2 *立即执行报文* 主动应用报文用协议-组-报文的三层结构定义一个主动应用，对于一些小的应用例如用于网管的某些短程序代码，采用这种机制显得过于复杂，本系统中定义了另外一种主动报文类型：立即执行报文。立即执行报文采用带内传输机制，类似于集成主动网络中采用的方法，即主动报文中封装主动代码，主动代码在目的地址的主动节点上直接运行，完成代码定制的功能。

2.2 报文分发机制

主动应用报文采用一种"按需分配"的主动代码分发机制。采用这种机制，主动应用报文本身不需要携带代码，而是通过这段代码的标识来获得代码，该机制通过代码缓存和报文分发来实现。这种机制一方面可以节约网络资源，如带宽等；另一方面可以提高代码执行的效率。

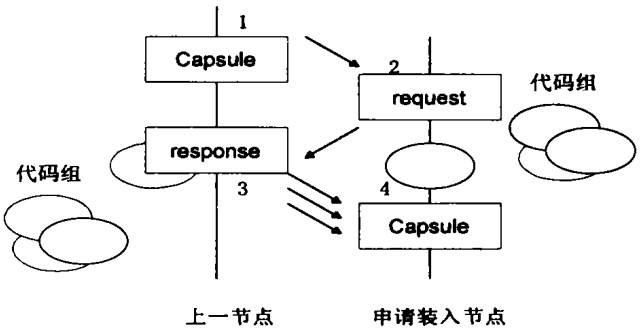

图 4 主动报文分发机制

2.2.1 *代码缓存机制* 为了提高网络的性能，每个主动应用报文所携带的代码可能会在经过的每个主动节点中缓存起来，以后使用相同的主动代码的主动应用报文就不再需要携带这段代码，而可以通过携带这段代码的一个标识就可以达到相同的目的。代码缓存机制正是基于这种情况而设计的。

在主动网络中应该采取一定的策略来缓存代码，同样也要采取某种策略来在网络的主动节点中分发这些代码。本系统使用一种类似于 ANTS 的"按需提取"机制来完成代码分发。在某一种应用的主动报文第一次经过网络中的某一个主动节点的时候，该主动节点会把它的代码缓存起来备以后使用，以后和该应用相关的主动报文就可以只携带这段代码的一个标识，如果某一个主动节点发现它的代码缓存库中没有这段代码，它自动向上级的主动节点索取这段代码，从而形成了一种自动的代码分发机制。代码缓存区采用双链表的数据结构，使用 FIFO 原则管理。

2.2.2 *代码分发机制* 主动网络模型采用软状态存储器来存储主动应用代码，假定一个新的主动应用代码存储在

源节点上，当主动报文在网上传输时，开始在源与目的节点间的每个主动节点中分发主动应用代码，也就是主动节点间的一致性维护。代码分发机制如图 4 所示。

代码分发报文格式：

Source addr	Dest addr	Group ID	Protocol ID	SeqNo	Code

NACK 报文格式：

Source addr	Dest addr	Group ID	Protocol ID	SeqNo

代码的分发过程为：

(1)当一个主动应用被创建时，主动应用的源端节点将代码串行化后装入软状态 Cache，同时通过 UDP Socket 端口向目标节点送出，目标节点是主动报文的目的地址，每一个应用有全局唯一的标识名 Protocol ID，代码分发报文的格式如图 4 所示；

(2)源和目的端间的中间节点若收到一个代码段报文，中间节点的软状态管理机制负责代码 Cache；

(3)如果目标节点没有收到序列号为 SeqNo 的代码报文时，向上级节点发送 NACK 报文，其格式如上。上级节点收到 NACK 报文将重传 SeqNo 代码段。目标节点每收到一个代码段将其存入软状态 Cache 中；

(4)一个主动报文在执行前，首先检查需要的主动应用代码是否在主动应用 Cache 中，如果不在，等待一定时间后再次查询代码分发机制是否已将其装入，每次查询失败触发其向上级节点发送 NACK 报文，如果主动报文等待超过一定的时间(如 30 秒)，表示这次装入失败，执行被终止；

(5)在没有主动节点访问软状态 Cache 的闲置(Idle)期，软状态管理器周期性地访问上级节点软状态 Cache，维护源和目的节点间的 Cache 一致性。

2.3 主动节点

主动网络中的每一个主动应用是通过主动报文与主动节点的交互来实现的。前面的章节中，我们提到主动节点由执行环境和节点操作系统(NodeOS)组成，本系统的主动节点结构如图 5 所示。

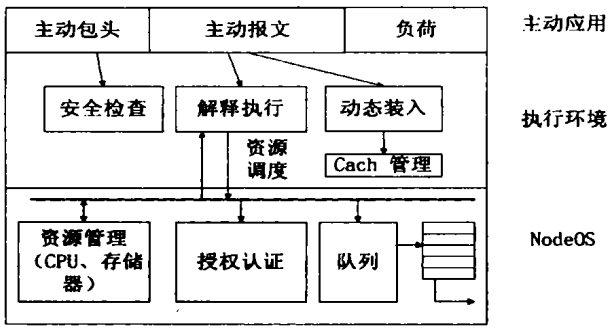

图 5 主动节点结构图

经过主动节点的主动报文可能很多，基于安全和效率的考虑，主动节点不一定对每个主动报文中的主动代码都执行。在本系统中，我们设定每个主动节点根据主动报文的源和目的地址决定是否执行该报文，即在每个主动节点中定义一个列表存储它可执行的主动代码所属的网络范围。

主动节点的主要工作流程是：当一个主动报文经由通道到达主动节点后，首先进行报文分类，节点检查数据包的源和目的地址，若在执行范围外，则直接进行转发。然后检查报文类型，若为传统数据报文，则向基础网络服务提出请求，调用相应的协议处理程序对数据包进行处理。若数据包是执行范围内的主动报文，首先对报文进行完整性和安全检查，然后备份。节点地址是目的地址：首先判断主动数据包的类型。如果是立即执行主动报文则从报文的代码域抽取相应的主动代码与相关数据，并将其装载执行，完成代码定制的功能；如果应用生成了需要转发的结果数据，主动节点要将其封装成数据包，发送给目的节点。如果主动报文的类型为主动应用包，节点首先检查软状态 Cache 中是否有所需的代码段，如果存在则执行，不存在就调用上节论述的代码转发机制进行装载，结果的处理同立即执行报文相同。如果节点地址不是目的地址：除了对主动报文的执行外，还要根据主动报文的执行情况确定按默认路由转发主动报文或者按主动报文定制的转发算法转发主动报文。

这样用户通过主动报文可以将网络协议注入到主动节点中，从而实现对主动网络的编程。

2.4 安全问题

由于主动代码是不可信任对象，对于主动网络的安全产生了很大的威胁，设计一个主动网络必须考虑安全主动环境的问题。主动网络的安全可以从以下三个方面来考虑：

(1)安全访问机制：防止主动服务对象对未授权资源的使用；

(2)资源消耗控制机制：保护主动网络中的资源如主动节点的 CPU、内存和带宽资源不被任何一个主动服务对象过度消耗；

(3)审计和监控机制：保护主动节点和主动报文不受到破坏和攻击；

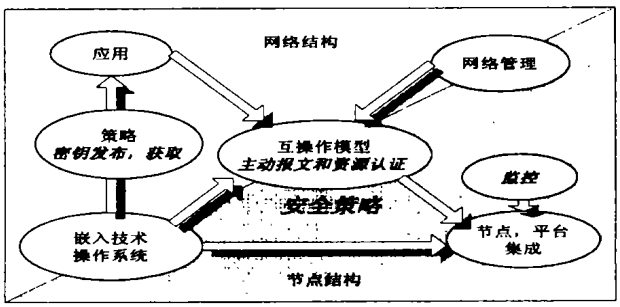

图 6 主动网络的安全渗透模型

图 6 是主动网络的安全渗透模型，安全策略渗透在网络和节点的结构当中。本系统模型实现的安全策略主要是：审计和本地资源保护。

2.4.1 报文审计 其目的是确定报文发送者的身份和权限。每个主动报文携带一个安全证书(policy)，类似于 Ipv6 扩展头部中身份验证的设置。安全证书代表了报文的发送者或者作者。异构环境中证书必须加密，防止欺骗的发生。接口为：checkPolicy(Capsule C)，验证安全证书的正确性。

此外，因为主动应用报文的报文标识能在网络范围内唯一标识报文，可以用作报文的审计，防止未经审计的报文被执行。

2.4.2 资源管理 对主动网络的资源管理，不同的系统有不同的解决方法，例如 SwitchWare 中构建了 SANE 的网络安全中间层；使用主动网络专用的编程语言 PLAN，由于语言本身的安全特性如：无循环结构来实现主动网络的资源管理。

(1)网络资源管理。无论是传统网络或者是主动网络，报文的带宽资源消耗都很难准确限定，在本系统的报文格式中设计借鉴了 Ipv6 中站点限制(hop limit)的方法，也就是在报

文中加入资源限制(Resource Limit)域。该域的值每经过一个站点衰减一次，当值为 0 时，报文被丢弃，保证了报文不会在网络中永存。

(2)节点本地资源管理。设计的主动网络以 Java 为编程语言，同时 Java 也是主动代码语言，每个主动节点上运行 JVM，因此资源管理通过调用 JVM 的资源管理策略来实现。在 JDK1.2 后 Java 虚拟机提供了 JVMPI 接口，可以实现统计每个线程有关的资源消耗。以该接口为基础的资源监控器收集 JVMPI 中每个活动对象的线程起停、对象分配和 I/O 事件。资源监控器为每个应用统计 CPU、内存的消耗情况，然后决策资源控制。通过抛出异常来阻止安全管理器授权之外的操作。

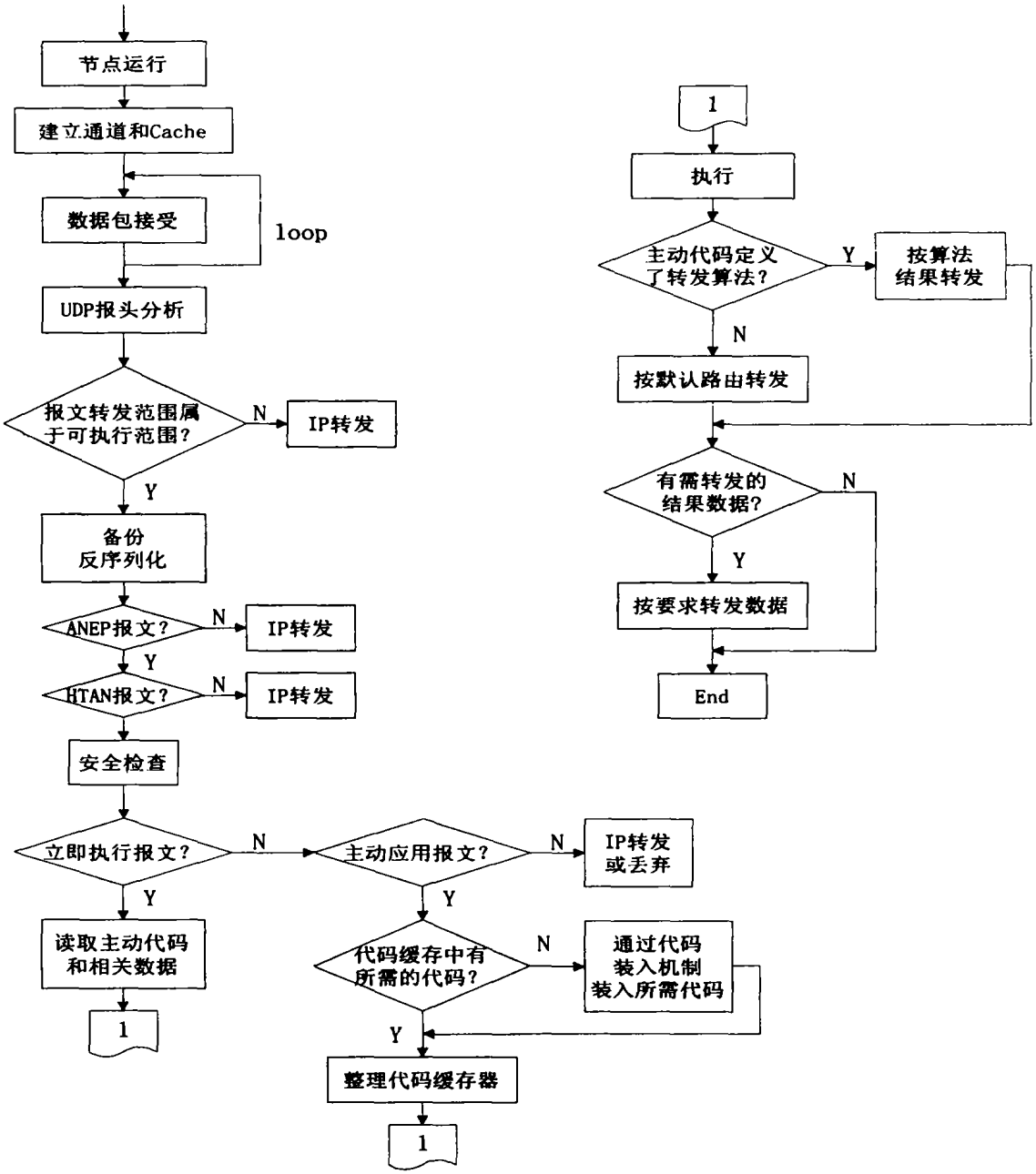

图 7　系统模拟流程图

3. 系统模拟与分析

3.1　系统模拟

本系统的模拟试验环境的主动路由器由一台 K6/450，128M 内存，两块 10/100M 自适应网卡，Red hat Linux6.0 系统，设置了 IP 路由转发功能的计算机模拟。与另外两台计算机通过 UDP 通道连接成主动网络。三台计算机上运行 JDK 和 HTAN 模拟程序，实现主动报文在主动节点上的执行和转发。主动网络模型的主要流程图如图 7 所示。模拟系统的结构图如图 8 所示。

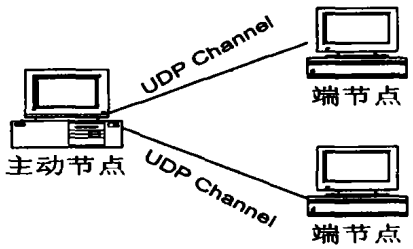

图 8　模拟系统结构

3.2 系统分析

对主动网络系统的测试主要应该针对以下几个方面：新协议安装的灵活性、代码和节点的安全性、健壮性以及系统处理性能。据此本系统设计了两个测试试验验证系统的性能。

3.2.1 *效率测试* 其目的是测试主动节点对主动报文处理能力和效率问题，如图 9 所示。

设计不同长度的立即执行报文，测试它们从通道读取、报文分类、安全验证、执行到转发的全过程所需的时间。

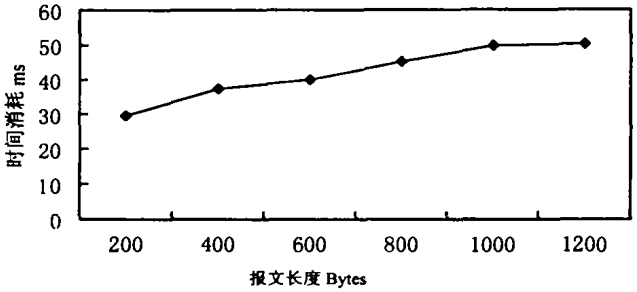

图 9 效率测试图

3.2.2 *健壮性分析* 健壮性和安全性：设计集中有带有自身缺陷的主动报文，这些主动报文包含以下情况：操作受安全机制保护的本地资源；代码中含无界循环，不能正常结束；主动报文的安全证书错误和主动报文转发路由错误。以上报文可以验证系统的健壮性和抵御拒绝服务攻击的能力。经过验证，系统可以正确处理以下的异常类型：

异常类型	描述
ResourceException	主动代码对未授权资源进行了操作
TimeException	一个主动进程的运行时间超长
PolicyException	安全证书认证失败
NoRouteException	转发路由不存在

上述异常产生后，异常消息被打包发送给该主动代码的发送端。

结束语 主动网络作为较新的研究领域，仍有许多问题有待进一步研究。目前的很多主动网络安全机制解决仅是单个主动节点的资源保护和主动代码的审计机制，如何保护主动代码对网络资源全局消耗和分配还是个困难问题。由于主动网络允许应用定制其在网络中的执行过程，如果主动网络设计不当，就可能造成难以预料的应用进程间相互影响，从而影响网络的透明性。

参 考 文 献

1 Wetherall D, Guttag J, Tennenhouse D L. ANTS: A tool kit for building and dynamically deploying network protocols. In: IEEE OPENARCH'98. San Francisco, 1998. 50～55

2 Biswas J. The IEEE P1520 Standards Initiative for Programmable Network Interfaces. IEEE Communications, Sepcial Issue on Programmable Network, 1998, 36(10)

3 Vanet G. A Self-Configuring Data Caching Architecture Based on Active Network Techniques. First International Working Conference, 1999. 32～37

4 Calderon M. Active Networks Support for Multicast Applications. IEEE Network Magazine, 1998, (5): 46～52

5 许强，温涛，等. 主动网络技术—新一代网络体系结构的发展趋势. 计算机工程，2000，26(2)：1～3

6 何丹，谢力. 一种新型的网络体系结构—主动网络. 计算机研究与发展，1999，36(1)：27～30

7 顾冠群，等. 高性能网络体系结构及其关键技术. 计算机世界，2000. 11

(上接第 61 页)

大文件需要更多的 RAM 缓存导致的。但是从图 6 中也可以看到，本文设计的消除同步写机制在各种文件尺寸下都是有效的，其速度都比普通 NFS 高。

结束语 本文提出了一种新型的 NAS 协作与缓存机制—ACAWM。ACAWM 在 NAS 自动组合的基础上消除了文件同步写，显著地提高了 NAS 的文件写性能。我们将 ACAWM 应用到 NFS 实现中，可以在完全不影响 NAS 易用性的情况下大幅度地提高 NAS 系统的性能，因此 ACAWM 有很大的实用价值。使用 ACAWM 可以在不添加任何硬件设备的基础上，实现高性能的异步写，其综合性价比高于使用 NVRAM 的系统。

参 考 文 献

1 Juszczak C. Improving the Write Performance of an NFS Serevr. USENIX Conference Proceedings, San Francisco, CA, 1994. 247～259

2 Ruemmler C, Wilkes J. Unix disk access patterns. In: Proc. of the Winter'93 USENIX Conf. 1993. 405～420

3 Baker M, Asami S, Deprit E, Ousterhout J, Seltzer M. Non-volatile Memory for Fast, Reliable File Systems. In: Proc. of the 5-th Intl. Conf. on Architectural Support for Programming Languages and Operating Systems, Boston, MA, Oct. 1992. 10～22

4 Ioannidis S, Markatos E, Sevaslidou J. Using Network Memory to Improve the Performance of Transaction-Based Systems. In: Proc. of Intl. Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA), 1998

5 Pnevmatikatos D, Markatos E P, Maglis G, Ioannidis S. On using Network RAM as a non-volatile Buffer. Cluster Computing, 1999, 2(4): 295～303