基于 Hash 函数的分布式路由算法*)

余盛季　李毅超　王　涛

(电子科技大学计算机科学与工程学院　成都 610054)

摘　要　随着 Internet 的发展,路由器成为了网络性能的瓶颈。路由算法的效率和鲁棒性成为网络研究领域的热点之一。分布式系统采用并行运行,避免了单点故障。本文提出的分布式路由器使用 IP 做为任务分配粒度,利用 Hash 算法实现负载均衡。改进了基于心跳和检测点的故障检测机制,在较低的系统开销下缩短了系统检测的响应时间。仿真试验的结果表明,该机制可行且高效。

关键词　分布式,路由,负载均衡,容错

A Novel Mechanism of Distributed Routing Algorithm Based on Hash

YU Sheng-Ji　LI Yi-Chao　WANG Tao

(College of Computer Science and Engineering, UESTC, Chengdu 610054)

Abstract　With the development of Internet, routers are becoming the bottleneck of networks. The performance and robustness of routers are important research areas in network. Distributed system can provide the parallel execution and avoid the single point of failure. In this paper, Distributed Routers (DR) uses IP as the task assignment granularity and Hash algorithm to implement the load balance. The fault detection based on the heartbeat mechanism and checkpoint is improved to decrease the detection response time under the low system expense. The results of simulation show that DR is an effective parallel router scheme in distributed environment.

Keywords　Distributed system, Router, Load balance, Fault tolerance

1　引言

随着千兆以太网和光纤网络的普及,路由器日益成为了网络流量的瓶颈。针对路由器进行改进成为了计算机网路研究的热点之一。此外,作为 Internet 上的核心设备,路由器的可靠性不应被忽视。基于上述问题,我们设计并实现了分布式路由器 Distributed Routers(DR)。利用分布式框架,路由器的并行处理能力[1]得到了提高,并解决了单点故障问题。

在分布式路由器中,负载均衡和故障检测是两个关键的问题。它们直接影响到整个系统的执行效率。

本文第 2 节给出了路由节点上的负载均衡算法,第 3 节提供了故障检测机制的设计,第 4 节给出仿真试验,最后是总结。

2　负载均衡算法

在分布式系统中,路由节点间相互协作,利用负载均衡算法[2]共同处理网络数据包。分布式系统中没有中心节点,所以,全部的数据包将会发送到每一个节点上。每个节点只处理属于它的数据。

在 DR 系统中,客户机设置的网关 IP 用同一个虚拟 IP 表示,同时静态设置其虚拟的 MAC 地址。路由器上的网卡将被配置成 promisc 模式。

当数据提交给负载均衡算法之前,数据包需要经过过滤处理,以确保是单 IP 包。处理流程如图 1 所示。

各个路由节点上的负载均衡算法同时执行,进行任务分配,所以当故障节点被检测出时,系统将进行一些调整。首先,屏蔽故障节点;其次,将在活跃节点上重新分配任务,实现任务迁移。

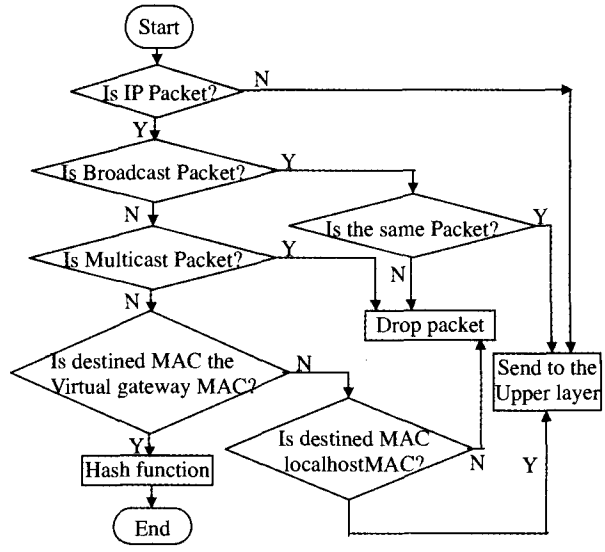

图 1　路由算法数据流程图

负载均衡算法必须满足下列条件。

1. 算法不能采用中心节点模式[3],因为如果任务分配由单个节点负责,很容易成为整个系统的瓶颈。

2. 算法应该和节点的个数无关。这样才能保证整个分布式路由器系统的可扩展性。

3. 算法应该是高速的,系统分配数据包而产生的系统开销应该被最小化[4]。

为了满足上述需求,Hash 函数设计如下。

Input:ipaddr(source IP address), id(id of router nodes), N(the num-

*)四川省青年软件创新工程资助项目,编号:2004AA0354。**余盛季**　硕士,助教,主要研究方向:计算机网络与网络安全。**李毅超**　硕士,副教授,主要研究方向:计算机网络与网络安全。**王　涛**　博士研究生,主要研究方向:分布式文件系统,P2P 计算。

```
    ber of routers);
hash ( ipaddr, id, N)
   {
     h =ipaddr % N ;
     if(h= =id)accept ( ) ; /* dealing with the packet in this node
      */
         else reject ( ) ; /* dropping the packet */
}
```

3 故障检测的实现

如果系统中有一些路由节点不能提供路由服务，故障检测机制需要及时地发现问题。

目前的故障检测机制主要是基于检测点[5]和基于心跳检测[6]。我们对此进行改进，结合了上述两种机制。考虑到计时器需要占用大量的系统资源[7]，尽量避免计时器中断的产生。另外，计时器心跳信号的间隔将依据系统检测来设定。

假设 T_1 表示系统检测的时间间隔，T_2 表示服务请求的时间。心跳检测的时间间隔则应该是 $min(T_1, T_2)$。因为路由器的服务请求一般都是短任务，所以当请求频繁到达时，$T_1 > T_2$，这时计时器不会激活。而当请求比较少时，计时器的激活不会影响到系统的性能。

通过分析现有的故障检测机制，对检查点机制而言，一旦检测开始执行，则每检测一次就需要节点间交互一次，开销较大。而对心跳检测机制而言，当节点在一个周期内没有收到心跳信号时，节点将被认为发生故障。有的时候，会存在一些误判而导致系统切换抖动。为了避免误判和减少系统开销，我们把心跳检测与检查点检测结合起来。

节点间我们采用心跳检测机制，每个节点主动发送它们自己的心跳信号。同时，设置检查点。假设 T_3 表示检查点检测的周期，则 $T_3 > T_1$，即检查点的检测周期应该大于心跳检测的周期。检查点可以利用计时器计时，如果在一个周期 T_1 内可以接收到全部节点的心跳信号，则计时器将会进行下一个计时周期。如果在周期 T_3 内没有收到一些节点的心跳信号，那些节点将会被判定为失效节点。

心跳信号可以用于标示一个节点。在我们的系统中，使用数据结构（site state）来记录。

```
struct site state{
    int fwid; /* id of node */;
    unsigned long ipaddr;/* IP address of node */;
    int fwnum;/* number of node */;
    };
```

为了检查在一个周期 T_3 内节点是否收到系统中各个节点的心跳信号，在节点上设置一个记录全部节点心跳信号的状态数组。如图 2 所示。

object statue	node1	node2	...	noden
	0	0		0

图 2 心跳机制的状态数组

初始状态下，每个节点的状态都为 0，如果接到某个节点发来的心跳信号则相应的 statue 置 1。一个时间周期 T_3 到达时，节点通过查看状态数组的状态决定是否出现失效节点。

在均衡算法中，每个路由节点上都设定了 Temp-id。它的值初始化为应用层的 fwid。当系统检测出故障节点后，算法将通过 Ioctl 机制交付该节点的 ID 号给均衡算法。Temp-id 随之发生变化。

```
if (fwid > faultid)
  then temp_id--;
```

调整的结果保证当前活动节点的 temp_id 构成一个连续的活动节点链表。

同时，负载均衡算法通过二次 Hash，屏蔽了故障节点。算法伪代码如下：

```
Input:ipaddr (source IP address),id ( the node id),N(the number of
      nodes);
hash (ipaddr,N,id)
     {
       h=ipaddr % N;
       if (h= =faultid)
          then
               i=ipaddr % M; // M denotes the number of active
               nodes
               if(i= =temp_id) then accept( );
                               else reject( );
          elseif (h= =id)
               then accept( );
               else reject( );
}
```

剩余的活跃节点将重新分配故障节点上的流量。所以两次 Hash 函数可以屏蔽故障节点，并保证流量平衡。图 3 是该分布式路由器故障检测模块的逻辑结构图。

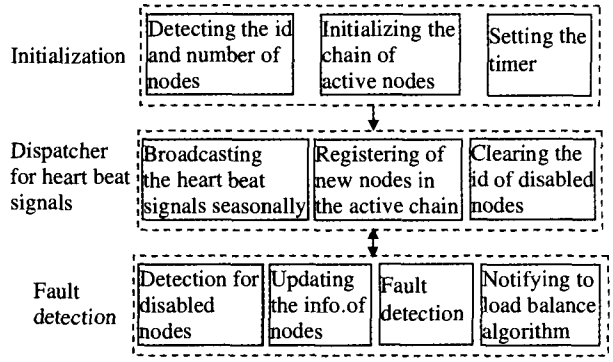

图 3 故障检测模块的逻辑结构图

4 仿真试验

4.1 仿真环境

我们构建一个 100M 以太网的环境来对上述模型进行仿真。由 5 台客户机（IP 地址随机从 172.168.1.X 抽取）和 3 台路由器节点（IP 地址分别是 192.168.1.1，192.168.1.2 和 192.168.1.3）。拓扑结构如图 4 所示。

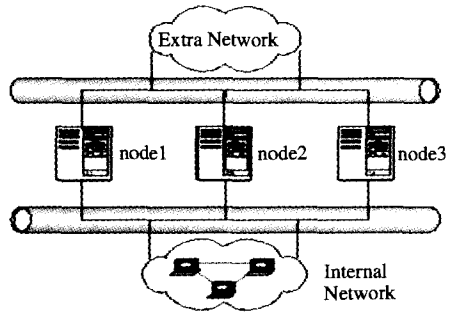

图 4 仿真试验的拓扑结构

4.2 结果分析

首先，我们测试了整个系统的高可用性 High Application（HA）。由客户机 172.168.1.18 发出对外部网络的访问请求。通过运行 netstat 命令，可以发现它的请求是由 IP 为 192.168.1.1 的路由节点进行转发。然后，关闭了该路由节点，来模拟节点发生故障的情形。20 秒后（具体时延可以设置），172.168.1.18 的对外请求已经被剩余的路由节点进行转发。

（下转第 44 页）

图5和6显示中码率情形下基本层使用两种采样方法的(码率-PSNR)曲线。图5和6中"DCT核"代表用DCT核分别进行下采样和上采样;"SVM"代表用SVM参考方法分别进行下采样和上采样。

从图5和6可以看到,在中码率情形下,基于DCT核的采样过滤器的性能仍然要优于SVM方法。PSNR的提高是1.5dB(City)和2.5dB(Harbour)。

结论 本文提出了一种用于改善SVC空间伸缩性的基于DCT核的上/下采样过滤器。试验表明,尽管该方法较SVM参考方法的计算复杂性略有增加,但是其低频响应特性和码率-PSNR性能都得到了较大的改善,有助于实现更好的SVC空间伸缩性。

参考文献

1 Reichel J, Wien M, Schwarz H. Scalable Video Model 3.0, ISO/IEC JTC1/SC29/WG11 N6716, Oct. 2004

2 SVM 3.0 Software. ISO/IEC JTC1/SC29/WG11 N6717, Oct. 2004.

3 Benzler U. Spatial scalable video coding using a combined subband-DCT approach. IEEE Trans. Circuits Syst. Video Technology, 2000, 10(10): 1080~1087

4 Mokry R, Anastassiou D. Minimal error drift in frequency scalability for motion-compensated DCT coding. IEEE Trans. Circuits Syst. Video Tech., 1994, 4(10): 392~406

(上接第41页)

其次,通过使用Web Application Stress工具,使用5台客户机来对外发出请求,同时记录了每个路由节点的负载状态。

表1 采用均衡策略下的路由节点负载状况

IP \ Parameters	Sampling time	System flow	CPU Utilization	Memory used Whole memory
192.168.1.1	10:58am	7kbps	10%	16952k/255272k
	11:00am	8kbps	12%	17203k/255272k
192.168.1.2	10:58am	5kbps	9%	15802k/255272k
	11:00am	7kbps	8%	16001k/255272k
192.168.1.3	10:58am	9kbps	10%	17129k/255272k
	11:00am	8kbps	13%	17618k/255272k

表1是在10:58am和11:00am两次从3个路由节点上进行采样的数据。3台路由节点采用同样的硬件配置,所以根据负载均衡算法,3个节点的负载情况应该相似。而上述表格中的试验结果证明了上述的观点。

进一步,为了对采用和未采用负载均衡算法的情况做比较,我们记录了未采用负载均衡算法的路由节点192.168.1.1的状态。从表2和表1可以看出,通过使用负载均衡算法,CPU的平均占用率从18%降至12%。所以,我们的负载均衡算法可以有效的降低单个节点上负载。

表2 未采用均衡策略下的路由节点负载状况

IP \ Parameters	Sampling time	System flow	CPU Utilization	Memory used Whole memory
192.168.1.1	11:20am	12kbps	19%	20142k/255272k
	11:22am	10kbps	17%	19248k/255272k

对客户机而言,服务质量也得到了改善。表3中,TTFB表示客户机收到第一个字节的时间。TTLB表示客户机收到最后一个字节的时间。

表3 WAS客户机数据

State \ Parameters	TTEB		Avg	
	Client A	Client A	Client A	Client A
With balance	428.31ms	389.42ms	477.31ms	355.7ms
Without balance	793.55ms	578.33ms	956.12ms	763.13ms

从表3可以看出,通过使用我们的算法,客户机的TTFB和TTLB都减少了,这表明客户机将得到更短的响应时间。

结论 通过使用分布式结构,路由器可以提供并行处理的能力,避免了单点故障。在DR中,负载均衡分配。结合心跳和检测点的故障检测机制,我们提供了一个新型的分布式路由器。仿真试验结果证明该模型的可行性。

参考文献

1 Shfaq A, Arif G. Semi-distributed load balancing for massively parallel multicomputer systems. IEEE Transactions on Software Engineering, 1991, 17(10): 987~1004

2 Baumgartner K M, Wah B W. Gammon: a load balancing strategy local computer systems with multi-access networks. IEEE Transactions on Computers, 1989, 38(8): 1098~1109

3 Hac A. Load balancing in distributed systems: A summary. Performance Evaluation Review, 1989, 16(2-4): 17~19

4 Ferguson D, Yemini Y, Nikolaou C. Microecomnomic algorithms for load balancing in distributed computer systems. In: Proc. of the 8th Int'l Conf. on Distributed Computing Systems, 1988. 491~499

5 Plank J S, Li K, Puening M A. Diskless Checkpointings. IEEE Trans. Parallel and Distributed System, ~9(10)

6 Banerjee P, Abraham J A. Fault-secure algorithms for multiple processor systems. In: Proc. of the 11th Int'l Symp on Computer Architecture, 1984. 279~287

7 Edmonds J, Karp R M. Theoretical improvements in algorithmic efficiency for network flow algorithms. Journal of the ACM, 1972, 19(2): 248~264