Implementing Software within the network

7th April 2003

Abstract

Recently there has been a drastic change in the model of the Internet due to applications such as web proxies, content delivery networks and Active Networks that execute software at nodes in the network. This approach improves the performance of applications, but at the same time has some performance penalties if the software is executed at nodes away from the edges of the network. We make the argument that such software can only be executed at nodes (routers) at or close to the edge of the network.

1 Introduction

The traditional model of the Internet is one in which nodes within the network do the task of routing packets, whereas the end-nodes are responsible for maintaining state and ensuring correctness.

However, this model is changing now. Apart from routing packets, nodes in the network run other software as well. Some examples of this are web caches, content delivery networks and firewalls. Application-specific code is executed at some routers in each of these examples.

Active networks are networks in which application or user-specified code can be executed on internal nodes of the network. This makes the network more flexible while potentially increasing end-to-end performance of the application by processing at internal nodes in the network.

In this paper, we argue that despite the advantages of executing application-specific software at routers in the network, such software can be executed only at nodes at or near the edges of the network.

2 Current Software (in the network)

In this section, we present a few applications which implement software at routers in the network.

2.1 Firewalls

Firewalls implement filters that determine which packets should be passed transparently and which should be blocked. Although they have a peer relationship to other routers, they implement application and user-specific functions in addition to packet routing[7]. These firewalls are special routers at the edge of an administrative domain.

2.2 Web Proxies

Web proxies provide a user-transparent service that is tailored to the servicing of web pages. Web pages are cached at nodes within the network, from where requests can be served directly. Web caches within the client network are very successful today. E.g, AOL's web cache services a few million hits a day .

If cached content is static, the cache should be near the client nodes in order to reduce client response time. However, a lot of Internet content is dynamic now. In order to minimize the cost of updating the cache, the cache should be nearer to the service provider Such caches help to offload the web server.

2.3 Transport Gateways

These are nodes located at strategic points that bridge networks with vastly different bandwidth and reliability characteristics, e.g. at the junction between wireless and wired networks. To support mobile access to wired networks, TCP snooping[8] retains per-connection state information at wireless base stations.

2.4 Application Level Video Gateways

The current model for multicast transmission of video over the Internet assumes that a fixed average bandwidth is uniformly present throughout the network. Consequently, sources limit their transmission rates to accomodate the lowest bandwidth links, even though high bandwidth connectivity may be available to many of the participants. [4] proposes an architecture where application-specific gateways use techniques such as the transcoding of images among video conference users with differing bandwidth constraints. These video gateways are placed near the source and receivers of the video stream and at strategic locations such as the edges of an administrative domain and the junction between wireless and wired networks.

3 Counter-Arguments and Rebuttals

3.1 IP Multicast needs to execute software at all nodes in the network

There are several standards available for routing IP Multicast traffic[9]:

- Distance Vector Multicast Routing Protocol (DVMRP).
- Multicast Open Shortest Path First (MOSPF)
- Protocol Independent Multicast (PIM)

DVMRP has significant scaling problems because of the necessity to flood frequently. This limitation is exacerbated by the fact that early implementations of DVMRP did not implement pruning. MOSPF is best suited for environments that have relatively few source/group pairs active at any given time. It works less well in environments that have many active sources or environments that have unstable links.PIM supports two different types of multipoint traffic distribution patterns: dense and sparse. Dense-mode PIM uses Reverse Path Forwarding is similar to DVMRP, and has the same scaling problems as DVMRP. Sparse mode PIM works by defining Rendezvous Points via which multicast packets travel from sender to receivers.

Thus each of these methods has its own problems, which is a major reason why majority of the routers on the Internet today don't know how to handle multicasting. Most routers are set up to move traditional IP unicast packets - information that has a single, specific destination[10].

Arguably, the only successful multicast network is the MBone network [10]. The MBone allows multicast packets to travel through routers that are set up to handle only unicast traffic. Software that utilizes the MBone tunnels multicast packets in traditional unicast packets so that unicast routers can handle the information. The MBone is a loose confederation of sites that currently implement IP multicasting. In addition, most of the multicasting software for MBone runs on dedicated workstations at client sites. The MBone is a virtual overlay network, and multicasting facility is implemented at the edges of the network.

3.2 In the Active Networks architecture, user or application-specified software can be executed at nodes within the network

Active networks permit applications to inject programs into the nodes of local and wide area networks[1]. Active networks seek to address the problem of slow network service evolution by building programmability into the network infrastructure itself, thus allowing many new network services to be introduced more rapidly. Compared with other architectures that incorporate flexibility, such as Cisco's Tag Switching, this approach offers a larger degree of flexibility, but at the potential expense of performance and security. Applications that can be implemented using Active Networks include Reliable Multicast[11], Online Auctions and Web Caches.

The Active Networks architecture suggests that software can be run at potentially any node in the network. However, as we will argue, there are several problems with this.

One cannot have application software running on backbone routers. This is simply because these routers are routing packets at very high speeds (Gigabits/second). It is very difficult for these routers to perform tasks such as caching and retransmitting multicast data in detecting gaps in multicast sequence numbers and sending NACKS for Reliable Multicast, or authenticating bid updates and rejecting low bids for Online Auctions. All such tasks, however simple would waste too much time on a backbone router impeding the primary function of the router.

If an Active node executes code belonging to one particular application only, the performance of that application is likely to improve provided it uses the Active Network correctly. However, if Active code is allowed to run on all routers in the network, the routers will probably have to run code belonging to different applications originating in different parts of the network. The penalty of running code belonging to one application might hurt the performance of other applications. Thus, it is best to run Active code belonging to an application closer to the application, where it is less likely to interfere with other applications. E.g, for Online Auctions bid processing for low bids can be offloaded to routers near the auction server. This would reduce the load on the Auction server and also reduces the cost of updating the router with the highest bid request.

Lastly, there is the issue of security in Active Networks. It is difficult to make the Active networks safe and flexible at the same time, and there is a tradeoff involved. One of the approaches suggested is to use type-safe higher level languages like Java for the Active node API. This approach trades off efficiency for security, thus hurting efficiency of code execution on Active Nodes. This only strengthens our arguments in the previous two paragraphs.

3.3 Content Delivery Networks (CDNs) execute software within the network

A CDN is a network optimized to deliver specific content, such as static web pages, transaction-based web sites, streaming media, or even real-time video or audio[5]. Its purpose is to quickly give users the most current content in a highly available fashion. There are two general approaches to building CDNs: the overlay approach and the network approach.

In the overlay approach, application-specific servers or caches at various points in the network handle the distribution of specific content types. The core network infrastructure, including routers and switches plays no part in content delivery. Good examples of this model are the CDNs deployed by companies such as Akamai, Digital Island and Speedera.

The network approach, on the other hand, deploys code to routers and switches so that they can recognize specific application types and make forward decisions on the basis of predefined policies. However, in these cases, most of the times such decisions are either taken in routers in the clients ISP or near the web server itself.

4 Arguments

4.1 High routing speeds at backbone routers

As discussed earlier, most backbone routers need to route packets at a very high rate. Assuming a 1 Gb/s link and 40 byte packets, the router needs to process around 3000000 packets every second, or one packet approximately every 300 nanoseconds.Executing application-specific software at these routers will kill the routing performance of these routers. [12] makes a case for implementing flow-monitoring algorithms necessary for network monitoring and usage-based billing in Application Specific Integrated Circuits (ASICs) near the router. These algorithms only look at the packet headers, and thus operate at the network level. Even though these algorithms can be written in just a few hundred lines of C code, they need to be implemented in hardware for efficiency purposes. It is difficult to perceive application software running at these routers that will need to look at packet payloads and perform much more complex operations.

4.2 Edge routers are special

In many cases, maximum performance enhancement can be achieved by running application-specific code at the edge routers. For example, snooping on TCP packets at wireless base stations is useful for wireless congestion control. Routers at the edge of a domain are the best candidates to perform admission control for packets. Proxy caches near a web server allow the load to be distributed and minimize the cost of updates for dynamic content. Web caches in ISPs minimize client-perceived latencies for static web pages at a very low cost. Special gateways at multicast senders and receivers can perform transcoding to adapt reliable multcast to heterogenous receivers. In all these cases, performance enhancements by placing software at the edge of the network clearly outweigh the benefits of doing something similar at other routers.

5 Conclusion

Implementing software at nodes within the network has advantages such as enhancing end to end application performance and increasing the flexibility of the network. However, in most cases such software can only be implemented at routers near the edge of the network. This includes routers near the client, near the Internet service and at strategic locations such as junctions of low-bandwidth and high-bandwidth networks. This argument will hold unless there is a significant improvement in router hardware or there are new applications that can justify the cost of implementing them at routers farther away from the edge of the network.

References

[1] D. Wetherall, U. Legedza, and J. Guttag, Introducing new internet services: Why and how, IEEE Network Magazine, special issue on Active and Programmable Networks, 1998.

[2] Ulana Legedza, David Wetherall, and John Guttag. Improving the performance of distributed applications using active networks. In Proceedings of IEEE INFOCOM '98, San Francisco, CA, April 1998.

[3] David L. Tennenhouse, Jonathan M. Smith, W. David Sincoskie, David J. Wetherall and Gary J. Minden, A Survey of Active Network Research, IEEE Communications Magazine, pp 80-86, vol. 35, num. 1, 1997.

[4] Elan Amir, Steve McCanne and Hui Zhang, An Application Level Video Gateway, In Proc. ACM Multimedia '95, San Francisco, CA.

[5] Irwin Lazar and William Terrill, Exploring Content Delivery Networking. IEEE Computer Society, "IT Professional" - August 2001.

[6] Eric Brewer, Software in the Network. Keynote address, IEEE InfoCom 2003.

[7] Greenwalk, M. et al. Designing an Academic Firewall: Policy, Practice and Experience with SURF. in Proc. of the 1996 Symp. on Network and Distributed Systems security. 1996. San Diego, CA.

[8] Balakrishnan, H. et al. Improving TCP/IP Performance over Wireless Networks. In Intl. Conf. on Mobile Computing and Networking. 1995. Berkeley, CA.

[9] Cisco Multicast Routing. http://www.cisco.com/warp/public/614/17.html

[10] Kevin Savetz, Neil Randall, and Yves Lepage, MBONE: Multicasting Tomorrow's Internet, 1996.

[11] Li-Wei H. Lehman, Stephen J. Garland and David L. Tennenhouse, Active Reliable Multicast, INFOCOM 1998.

[12] Estan and G. Varghese, New directions in traffic measurement and accounting, in Proceedings of the 2001 ACM SIGCOMM Internet Measurement Workshop, pp. 75-80, Nov. 2001.