December 1998

User Customization of the Internet Using Active Networking

Samir Chatterjee
Georgia State University

Lei Jin
Georgia State University

User Customization of the Internet Using Active Networking

Samir Chatterjee
Lei Jin
Computer Information Systems Department
Georgia State University

Abstract

This research proposes a new way of thinking about our current networks using a notion called active networking. Rather than just passive transport vehicle, active networks perform active computation on the messages that are in transit and hence facilitate far more functional capabilities than what is possible with today's Internet. The authors present an active network architecture to customize the Internet according to a user's need in which quality-of-service guarantees can be met for each individual application or end-user. We also discuss important issues related to implementation, security and management of such active networks.

Introduction

The authors propose a method to customize the Internet (or Intranets within organizations) to suit user's own needs using a controversial yet cutting-edge concept known as "Active Networking" (AN). Until now, our data communication networks were mainly passive vehicles for transporting packets from one point to another. Hardware inside such a communication network "cloud" typically includes routers, switches and gateways. These hardware run protocol software (namely IP or ATM) that pass user data opaquely without examination or modification. When a packet enters a network, routers only perform header computation and general actions taken are specified independent of the user process or application that generated the packet. But now there is a controversial idea called "active networking" where packets will be replaced by capsules containing active code (written in java) that will perform active computation at each cross-point in a network and be able to execute far more powerful functions than what we have witnessed so far. The Department of Defense has endorsed this as the most breakthrough research for the next decade and has set aside billions of dollars for funding [4].

One of the biggest problems facing network service provider is how to guarantee Quality-of-Service (QOS). End-users run several different types of applications (video-conferencing, medical-imaging, distance-learning, electronic-commerce) which traditionally require different levels of support in terms of bandwidth, delay, delay jitter and loss. Yet, service providers try to provide a "one-size-fits-all" solution which neither does justice to the end-user nor to the network that is being provisioned. We propose that using active networking concepts, it would be possible to cater to each individual end-user or application with the best possible performance guarantees. Hence we postulate that by using the network as an active computation engine (rather than a transport vehicle), a user can customize the Internet to suit his/her purpose.

In the next section, we discuss the driving forces that is leading towards this new paradigm. In Section 3, we propose an architecture for such AN. Section 4, discusses several critical issues that needs to be resolved in order to have such an AN. Finally we close this article with a set of adaptive applications that can really take advantage of this new paradigm and conclude with the current state of affair and pointers to future work.

Driving Forces

There are many reasons due to which active networking [6] is the next logical step in network evolution:

- *Technology Push*- when we look at end-user desktop machines, they have become tremendously powerful by running "applets". These applets are tiny computer programs that run locally on the system and can display far more information than was ever possible. Using the active programming language called Java, such applets are platform independent – meaning that these codes will run on all processors and over all operating systems. Could active networks leverage and extend these technologies for use "within" the network and make it even more powerful?
- *User Pull* - comes from the assortment of firewalls, web proxies, multicast routers, mobile proxies, video gateways, etc. that already perform user-driven computation at nodes "within" the network [1]. 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. The need to update the firewall to enable the use of new protocols is an impediment to their adoption.
- *Core Problems* - there are well-known core networking research problems that are still open. For example, congestion and flow control, scalable routing and multicasting applications especially at gigabit speeds are posing challenges that prove to be more difficult than first believed. Could active networks be a possible new paradigm to solve such well-known hard problems?

There are three principle advantages to basing the network architecture on the transmission of active programs rather than passive packets:

- Transmitting code provides a basis for adaptive protocols, enabling richer interactions than the exchange of fixed data formats.
- Capsules provide a means of implementing fine grained application-specific functions at strategic points within the network.
- The programming abstraction provides a powerful platform for user-driven customization of the infrastructure, allowing new services to be deployed at a faster pace than can be sustained by vendor driven standardization processes [1].

Architecture for Active Networks

To customize the Internet so that an end-user application can run with its stated QOS, a method to enhance computational activity within the network is required. There can be two ways of providing such enhancement. A discrete approach (also called programmable switches) in which a user would send their packets through such a "programmable" node [3] much the same way they do today. When a packet arrives, its header is examined and a program is dispatched to operate on its contents. The program actively processes the packet, possibly changing its content. A degree of customized computation is possible because the header of the message identifies which program should be run - so it is possible to arrange for different programs to be executed for different users or applications. In this approach, the addition of a new function to a network node would be the responsibility of the network service provider [5]. As with current networks, once a function is specified, each vendor would be free to implement the functionality in a manner consistent with the specification.

There is another far more powerful (though higher risk) approach to implementing functionality within the network. Known as the capsule approach, every message injected into a network is a program. We are replacing the passive packets of present day architectures with active "capsules" - miniature programs that are executed at each router they traverse. User data can be embedded within these mini-programs much the same way a PostScript file embeds a page's content. When a capsule arrives over a link to an active node, its contents are dispatched to a transient execution environment where they can safely be evaluated. We hypothesize that programs are composed of "primitive" instructions, that perform basic computations on the capsule contents, and can invoke external "methods", which may provide access to resources external to the transient environment. The execution of a capsule results in the scheduling of zero or more capsules for transmission on the outgoing links and may change the non-transient state of the node. Capsules containing routing algorithms will require access to such external methods so that it can interface with the routing tables while simple application capsules may just inform the node that less outgoing bandwidth is acceptable or not.

The difference between these two approaches ("discrete" versus "capsule") are analogous to the differences between adding individual features to HTML and defining the Java language. In the former case the semantics of each feature must be considered, including its interactions with all other features; in the latter case the set of all possible programs must be considered in defining the language and its interface to the rest of the system. It may be easier to address concerns about security and efficiency for a single function as opposed to an entire language. With the discrete approach, it is possible to deploy active networking incrementally, beginning with simple AN functions that provide immediate benefits for users and providers, and expanding the set over time.

In summary, these networks are active in two ways:

- Switches or routers within the network perform computations on the user data flowing through them;
- Individuals can inject programs into the network, thereby tailoring the node processing.

Issues — Mobility, Safety and Efficiency

When network programs is to be transmitted across the communication substrate and loaded into a range of platforms, the following three critical implementation issues must be supported:

- Mobility - the ability to transfer programs and execute them on a range of platforms
- Safety - the ability to restrict the resources that programs can access
- Efficiency - enabling the above without compromising network performance, at least in the most common cases.

Fortunately, "active technologies" have been developed within the programming language and operating system communities to make this possible. Mobility may be achieved at several different levels of program representation: express the program in a high-level scripting language such as Tcl; adopt a platform-independent intermediate representation, typically a byte-coded virtual instruction set (such as Java); or transfer programs in binary format such as Omniware. For implementing safety, Scout and Exokernel programs that allocates and schedules resources on a "path" basis to allow to share low-level access to system resources could be looked at [2, 3]. All these techniques can prevent unauthorized access to resources that lie beyond the boundaries of the transient execution environment into which a capsule is dropped. Such an active node is shown in Figure 1.

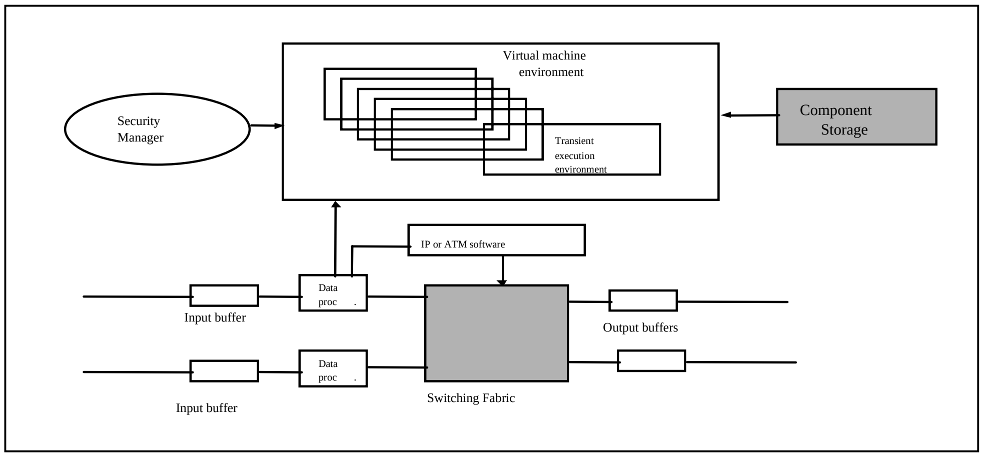

Figure 1. An Active Node (Interne Router) Architecture

Adaptive Applications and Conclusions

There is an untapped reservoir of applications that require sophisticated network-based services to support the distribution and fusion of information. IP-multicast services is one such promising application that can take advantage of active computation. We expect that there will always be applications that prefer to adapt their behavior dynamically to match the available bandwidth, as opposed to reserving bandwidth in advance. Application-specific multicast along with dynamic adaptation can jump-start video-based applications on the Internet.

Network congestion control can benefit from active networks. Intelligent discard may be signaled by the user, or media transformation (on-the-fly transformation of MPEG and JPEG data, including selective discard of discrete cosine transform) may be possible. Sensors monitoring some critical system (e.g., medical patient condition or home security) independently report to a central monitoring station. Again, if data is lost from one sensor, the information from the others increases in importance. Sensor fusion along with simulation can allow users to "see" composite images constructed by fusing information obtained from a number of sensors. Fusing data within the network reduces bandwidth required at the users, who are located at the periphery of the network. Placing application-specific computation near where it is needed also addresses the latency limitations.

In this paper, we have discussed a new research concept called active networking. We are presently developing active network algorithms to provide QOS in broadband multiservice networks. We strongly believe that active networking is the next logical step to solve the above problems and hence encourage more research in the area.

References

[1] David L Tenenhouse, David J. Wetherall, "Towards an Active Network Architecture", ACM Computer Communication Review, Vol. 26, No. 2, April 1996.
[2] G. Necula, P. Lee, "Safe Kernel Extensions Without Run-Time Checking", 2nd Symposium Op. Sys. Design and Implementation, OSDI'96, Seattle, WA 1996.
[3] T. Yemini, S. da Silva, "Towards Programmable Networks", Proc. IFIP/IEEE International Wkshp on Distributed Systems, Oct 1996.
[4] DARPA supported Active Networking Research. [*http://ito.darpa.mil/research/index.html*]
[5] S. Bhattacharjee, K. L. Calvert, and E. W. Zegura, "An Architecture for Active Networking", in Proc. Of High Performance Networking 1997.
[6] David L. Tenenhouse, et al, "A Survey of Active Network Research", IEEE Communications, January 1997.