ANCORS: Adaptable Network COntrol and Reporting System*

SRI-CSL-98-01

Livio Ricciulli, Phillip Porras, Nachum Shacham

March 19, 1998

Abstract

ANCORS is a distributed tool suite that merges technology from network management, active networking, and distributed simulation in a unified paradigm to assist in the assessment, control, and design of computer networks. This paper explores some of the advantages that can be obtained from merging the three technologies, and describes how ANCORS integrates complementary elements of each. ANCORS's architecture offers substantial software reuse, scalability, and flexibility and supports an extensible mechanism to employ multiple network management protocols of varying degrees of complexity. This paper also describes network engineering and monitoring services that were implemented to prototype some of ANCORS's architectural ideas and provide practical experience for their refinement.

1 Introduction

The Internet will become increasingly dynamic. Changes in the Internet will affect both its control mechanisms and the nature of information exchanged.

- New trends in network design [12, 4, 3, 13, 9, 1] seek to render network protocols more flexible and extensible, and to thus improve their overall usefulness. Configuration changes can be as dynamic as interpreting and executing a few predefined instructions as a network packet is received, causing new protocols to be loaded on demand, or modifying, deleting, or adding more permanent objects that implement application-specific network services.

- The introduction of new technologies and services as they become available may change the nature of network traffic. The Internet phone, video broadcast, and the increasing interest of cellular phone companies in accessing services on the Internet are examples of future technologies that may greatly affect the Internet.

* This work was supported by DARPA contract number DABT63-97-C0040.

The current state of the art in network engineering, monitoring, and control must improve dramatically. It is becoming increasingly apparent that effective management of large, ever-changing networks depends on sophisticated monitoring to help understand the way a network changes and in detecting anomalous behavior (both malicious and nonmalicious). Current network management and control software suffers from serious scalability and flexibility constraints because it is oriented to the single administrative domain level. As new interdependencies arise in sharing resources beyond single administrative domains, monitoring capabilities, like application-specific protocols, should be able to change over time, should adapt to new conditions as they develop, and should be scalable.

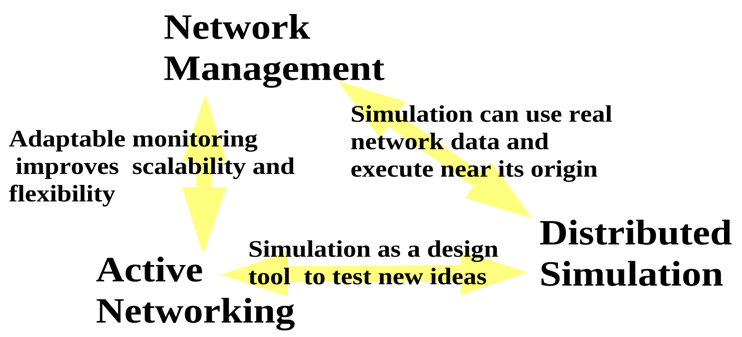
In addition to sophisticated and adaptable monitoring, future networks would greatly benefit from simulation services so that network engineers can experiment with new network technologies without compromising network operations. Current network engineering tools can scale only to small and relatively simple networks and are not interoperable. Tools will be required to scale far beyond current capabilities and will need to promote interoperability and model reuse. In addition to evaluating performance metrics to compare one design with another, network engineering tools should implement a development environment for validating new designs.

Besides offering powerful assessment and design tools, adaptable networks should also provide a flexible infrastructure to manage and distribute software. New standards being proposed to assist in the distribution and maintenance of software through the network [11, 10] would allow users to install or update software components by simply accessing HTML-like pages, thus providing more cost-effective mechanisms for distributing and maintaining application software. While the adoption of these mechanisms to dynamically add and maintain the code base of end-user applications may pose little technical challenge, extending such mechanisms to also deploy and maintain system-level software is more difficult. An addition of system-level networking software must be done very carefully to avoid potentially costly mistakes and the addition must also be propagated to the management infrastructure so that monitoring and control capabilities can change with the network itself.

The objective of our research is to streamline and, at the same time, enrich the management and monitoring of dynamic networks while adding new support into the network management paradigm to assist network designers. We revisited the concept of network management and extended its paradigm to make it the enabling technology for designing, testing, configuring, and monitoring network assets in the best possible way. ANCORS merges ideas from distributed network management, distributed simulation and active networking in a coherent, efficient, and user-friendly manner. Section 2 focuses on exploring some of the benefits that ANCORS gains by merging active networking with network management and distributed simulation. Section 3 discusses ANCORS's flexible system management framework that can deploy and manage our new multidomain infrastructure. Section 4 describes our initial implementation experiences, and Section 5 gives some concluding remarks.

2 Integrated Active Network Management and Design

Figure 1 illustrates the synergy derived from the merging of active networking concepts with network management and distributed simulation. Our preliminary experiences in merging



Network Management
Adaptable monitoring improves scalability and flexibility
Simulation can use real network data and execute near its origin
Distributed Simulation
Simulation as a design tool to test new ideas
Active Networking

Figure 1: Advantages of Merging Distributed Network Management, Distributed Simulation and Active Networking

these technologies under the ANCORS architecture indicate that we can provide an integrated environment for network technology development while affording substantial software reuse in supervising network operations. ANCORS gains from active networking the ability to dynamically deploy engineering, management, and data transport services at runtime. ANCORS leverages this capability to (1) coherently execute multiple simulations on the network in a scalable and integrated fashion to support design, (2) integrate network and system management with legacy standards (SNMP, CMIP) to result in a more flexible and scalable management framework, and (3) provide network management functions to supervise network operations, collect network statistics to be used as input to network engineering tools and higher-level assessment tools, and assist network operators in reacting to significant changes in the network.

Figure 2 depicts an architecture based on the integration of active networking concepts, network management, and distributed simulation. The architecture is divided into data, assessment, and control layers. The data layer operates at the data packet level and offers a set of services for the manipulation of network data. The assessment layer performs analytical reviews of network behavior to extract relevant semantic information from it. The control layer performs higher-order functions based on expert knowledge. All the services constituting these layers are deployable and use common system management support. ANCORS should distribute data-layer services at the inter-domain level (following the model of active networking) and limit the deployment of assessment and the control layer services to the domain level. However, depending on the amount of resource sharing resulting from the deployment of active networking, the services of the assessment layer may also be distributed across multiple domains. Because the control layer needs to possess a significant amount of authority to perform changes in the network, it should only be deployed within one domain. Several control services should then cooperate at the inter-domain level to exchange useful information for making better informed control decisions about their respective domains.

In this context a domain consists of collection of software and hardware objects that are managed by a single administrative authority.

A discussion about control services inter-domain information exchange is beyond the scope of this paper.

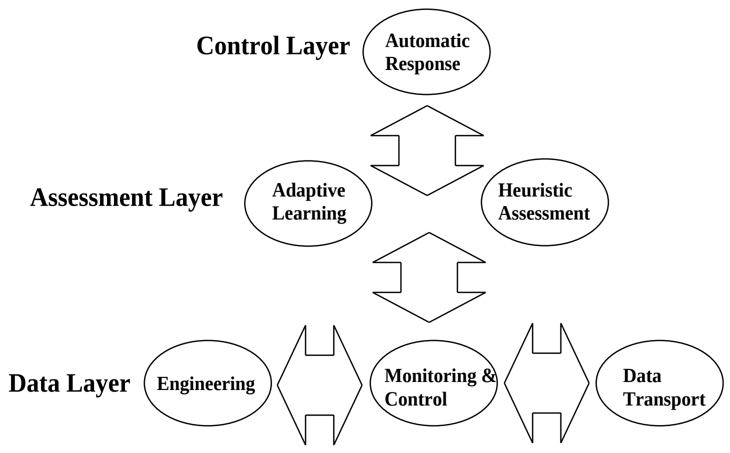


Figure 2: ANCORS's Architecture

2.1 Data Layer

Although merging three technical areas into one may offer definite advantages, it is still useful to recognize that the fundamental tasks to be performed can be separated. For this reason, we find it useful to decompose the data layer into three distinct kinds of data services that may benefit from dynamic deployment in the network and that naturally map to the three services we want to support: data transport, monitoring and control, and engineering.

2.1.1 Data Transport Services

Data transport services offer communication protocols, either quite general and extensible as proposed in [12, 3, 13, 9, 1], or more traditional services derived from the ones available today. In all cases, we assume that their deployment may be dynamic. The scope of this paper does not permit us to give a detailed description of these services and their possible applications. We instead focus on the engineering and monitoring and control services, which are our primary research emphasis. These services introduce the fundamental technology upon which ANCORS extends the network management paradigm to support planning, performance, and stability assessment.

2.1.2 Monitoring and Control

Monitoring and control services monitor the operation of network services, perform some initial analytical review of performance data to detect exceptional conditions, report relevant information to higher-level layers, and offer a mechanism for runtime configuration. These services perform tasks analogous to those performed by today's network management agents. In addition, specialized monitoring and control network services may be dynamically deployed to perform user-defined targeted analyses such as those proposed in [6].

The use of active networking to allow user-definable monitoring capabilities to be deployable gives ANCORS two major advantages: (1) it permits selective monitoring of a particular phenomenon as new network requirements and new usage patterns emerge over time, and (2) it improves monitoring scalability through an arbitrary degree of sophistication in the monitoring agents, thus allowing a fluid tradeoff of the amount of computation to be performed in the services distributed throughout the network with the amount of computation to be performed in the control stations.

2.1.3 Engineering Services

Engineering services aid in the design and testing of network services before their deployment. ANCORS incorporates distributed simulation to help operators explore and select the optimal deployment and configuration of network assets and develop new network technology. Hardware design tools have reached a very high level of sophistication and can assist hardware designers in all phases of design and development. Such tools can support a wide spectrum of levels of abstraction, from high-level purely behavioral specifications, to increasingly finer detailed structural layouts, all the way down to the actual design of the transistors on the silicon. Simulation is used throughout all phases of this design process, and it is the main mechanism that guides design choices. As for hardware, network design should also be carried out in an environment that can offer a variable degree of abstraction and that can offer simulation as a pivoting technology to guide development. We argue that, because of the organic nature of current networks and their fast evolution pace, design should be carried out in the real network itself. Future networks will need tools that can adapt their functionality and scope, and that can grow and change with the network itself.

- To generate results that accurately predict network behavior and performance, simulation and analysis must be closely tied to the actual, rather than on artificially generated, network traffic conditions. To that end, the tools should run on the network itself, taking the actual observable traffic conditions into consideration.
- Before committing network-wide changes such as the alteration of the network routing algorithm, an operator may want to conduct simulation experiments that can predict the behavior of the network under the new algorithm without affecting network reliability. That is, analysis and design tools should be available to a wide range of network operators, who could act independently or in collaboration with one another.
- New vulnerabilities may be discovered that threaten the survivability of a network. Operators should be able to install countermeasures dynamically to match evolving threats. Here, too, simulation experiments are needed to test efficacy of planned security measures and ascertain trade-offs between network performance and security.

Our distributed planning and simulation system leverages network management and introduces simulation as an additional network service. Integrating distributed simulation with network management has four main advantages: (1) it naturally supports reuse of both simulation software and network models, (2) the simulated models can use real network data produced by the monitoring agents, thus improving fidelity, (3) the consumers of the data (the simulation models) are placed close to the origin of the data to reduce overhead and (4) the monitoring and control capabilities of network management can be reused to monitor and control the simulations.

In practice, engineering services may mimic the behavior and performance of all other network services but differ from them in the following ways: (1) they live in a separate address space and are for the exclusive use of the network designers, (2) they operate protocols in a virtual timescale that may differ from physical time, and (3) they may generate synthetic network traffic that does not contain user data.

2.2 Assessment Layer

As shown in Figure 2, ANCORS's assessment layer interfaces with the monitoring and control services of the data layer and the automatic response service of the control layer. At the data layer, the monitoring and control services interface to both engineering and data transport services, thus providing a unique interface to the assessment layer. By designing our architecture in this way, we can achieve software reuse of the assessment and control layers and seamlessly integrate network engineering and data transport services within the same paradigm. At the assessment layer, two sophisticated analytical reviews of the network's data are performed. The first is a heuristic review of elements within the data, specifically looking for metrics within the reported results that represent exceptional or unexpected behavior. Boundary results for the metrics are specified at the initialization of the data collection agents, and may be dynamically updated by an administrator. The second form of assessment involves an adaptive learning algorithm that performs continuous statistical profiling of the network data. As data flows into the statistical profiling engine, the profile specific to the particular data acquisition agent is updated, and statistically stable results begin to emerge. The statistical profiling engine monitors the degree to which various user-defined metrics change with respect to the current operational behavior of the network. The algorithms used to provide the heuristic assessment and the statistical profiling of network data will be adapted from related research effort on information survivability [6].

2.3 Control layer

The output from the assessment layer is propagated to the control layer, where these results are displayed for the administrator or processed by an expert system decision engine capable of providing predetermined responses, given the receipt of various assessment results.

Response methods are predefined code segments deployed to the analysis target as part of the control-layer configuration space. Included with each valid response method are evaluation metrics for determining the circumstances under which the method should be dispatched. Formulating effective responses to detected exceptional activity is, itself, an extensive subtopic being pursued within the scope of our research. In many situations, the most effective response may be no response at all, in that every response imposes some cost in system performance or (worse) human time. The extent to which the control layer contains logic to filter out uninteresting analysis results may mean the difference between effective monitoring units and unmanageable (soon to be disabled) monitoring units. For certain analysis results, such as the detection of significant exceptional activity through heuristic analysis, the necessity for response invocation may be obvious. For other analysis results

The passage of time is explicitly controlled by predefined time-synchronization algorithms.

(such as the detection of a statistical anomaly in the packet stream to or from a network service) the control layer may require greater sophistication in the invocation logic.

It is important to tailor a response that is appropriate given the severity of the problem, and that provides a singular effect to address the problem without harming the flow of legitimate network traffic. The following general forms of response are available through the control layer as analysis results from the assessment layer are received:

- **Passive results dissemination:** The control layer can simply make the analysis results available for administrative review. We are currently exploring techniques to facilitate passive dissemination of analysis results by using already-existing network protocols such as SNMP, including the translation of analysis results into an ad-hoc MIB structure.

- **Assertive results dissemination:** The control layer can actively disseminate administrative alerts. While the automatic dissemination of alerts may help to provide timely review of problems by administrators, this approach may be the most expensive form of response, in that it requires human oversight.

- **Dynamic oversight of the monitoring and control layer:** The control layer may provide limited control over the configuration of logging facilities within network components, requiring greater collection from the data layer as the assessment layer identifies significant exceptional activity.

- **Dynamic configuration changes:** The control layer may induce changes to the configuration of the network infrastructure in response to exceptional activity. Such actions may include reconfiguration of services, routing databases, or filtering rules.

3 ANCORS's System Management

ANCORS's support for the deployment and management of network services focuses on services that are fairly permanent and long-lived and that can benefit from having a separate system management infrastructure. The management of short-lived services should be directly embedded in the mechanisms that deploy and control them and therefore are integral parts of their dynamic loading mechanisms. This kind of micro-management, currently being addressed by several research projects [12, 3, 13], is beyond the scope of this paper.

From a system management point of view, all services constituting ANCORS's architecture (Figure 2) are equal. Their deployment, operation, and monitoring can be performed using the same mechanisms, and therefore they can be managed in the same paradigm. In the following discussion we offer a solution for their effective, scalable, and user-friendly management.

All classes of deployable network services from simple SNMP daemons to more sophisticated data transport services and assessment services require identical support functions from system management. The support functions can be broadly characterized as those achieving (1) process control, (2) configuration, or (3) monitoring.

Process control functions allow the loading and unloading of network services to and from network nodes. The physical location of the code that implements the network services may be different from the physical location of their deployment. For this reason, a reliable transport protocol such as TCP may be used to transfer the code. Some searching and browsing capabilities should be offered so that network operators may also easily locate the appropriate executable codes. Existing protocols such as HTTP and LDAP/X500 or newer and richer protocols such as the ones described in [10, 11] could be used for user-friendly code storage and retrieval.

Configuration functions write control data into the network services after they have been loaded onto the intended nodes for the purpose of integrating them into the network node and possibly tailoring the service to particular needs.

Monitoring functions result in reading data from the network service for the purpose of supervising its operation. This support function may be invoked directly from the network administrator interactively (or it may be invoked iteratively by the network monitoring and control services) for the purpose of collecting performance data.

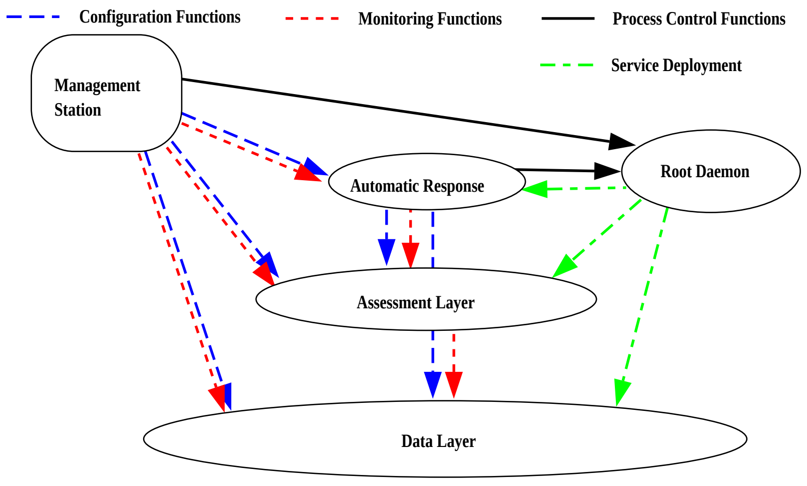
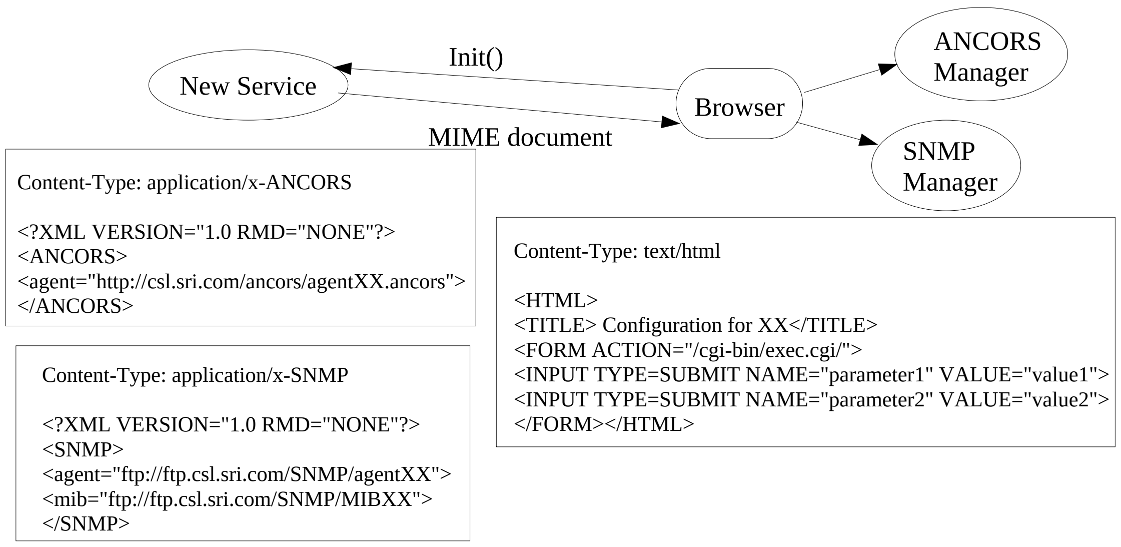


Figure 3: ANCORS's Management Architecture

This decomposition gives rise to the architecture depicted in Figure 3. In ANCORS, a root manager handles process control requests coming from the management stations or automatic response services to either load new services or terminate existing ones. The assessment layer interprets monitoring results from the data layer, and the automatic response services react to significant conditions as they are reported by the assessment layer. The automatic response services may reconfigure both the assessment services and the data-layer services in response to changes in the network behavior.

Notice from Figure 3 that ANCORS's architecture allows the traditional but nonscalable approach of having the management station directly monitor and control the data layer. This

Monitoring is typically associated with network management and control is associated with system management. Because of the flexibility introduced by active networking and ANCORS's ability to support multiple protocols, monitoring and control functionalities can be supported within the same management framework.



```
Content-Type: application/x-ANCORS

<?XML VERSION="1.0 RMD="NONE"?>
<ANCORS>
<agent="http://csl.sri.com/ancors/agentXX.ancors">
</ANCORS>
```

```
Content-Type: application/x-SNMP

<?XML VERSION="1.0 RMD="NONE"?>
<SNMP>
<agent="ftp://ftp.csl.sri.com/SNMP/agentXX">
<mib="ftp://ftp.csl.sri.com/SNMP/MIBXX">
</SNMP>
```

```
Content-Type: text/html

<HTML>
<TITLE> Configuration for XX</TITLE>
<FORM ACTION="/cgi-bin/exec.cgi/">
<INPUT TYPE=SUBMIT NAME="parameter1" VALUE="value1">
<INPUT TYPE=SUBMIT NAME="parameter2" VALUE="value2">
</FORM></HTML>
```

Figure 4: Services' management can be specified through the use of MIME encapsulation and the XML language.

aspect of our architecture can be very useful when simple network management technologies are employed that do not require ad-hoc distributed monitoring functions.

We recognize that Internet services must be open, simple and flexible, and therefore we do not intend to have a unique format for the information exchanges between network services and management stations. While we intend to formulate a protocol that best suits the nature of our adaptable management paradigm based on delegation [14], we recognize that legacy protocols and simple Web-based solutions should also be possible. Standards like SNMP should be supported in those cases where backward compatibility is desirable; HTML could be employed for services to be configured and monitored interactively. To support multiple protocols, ANCORS uses a discovery mechanism to probe newly deployed services. The idea is quite simple and is somewhat similar to the approach followed today on the Web. Each network service listens on a port assigned to it by the root manager. After deployment, the services respond to a predefined and universally agreed-upon command **init** (the equivalent of GET / in HTTP). The **init** command causes the network services to respond with a MIME-encapsulated reply. In general, the reply contains information to be used for the configuration and monitoring of the service itself and the configuration of other related services.

As shown in Figure 4, a service wanting to use SNMP, for example, replies with information encapsulated in an SNMP-specific MIME. The reply would specify the URL of the SNMP agent and the associated MIB to be loaded with the service. The management station then loads the requested SNMP agent and MIB and automatically adds the service to its SNMP manager application.

Figure 4 also shows an ANCORS MIME corresponding to the specific management application that we intend to develop in the near future, based on delegation. In a more simple scenario, the service can also reply with an encapsulated message in HTML format. The replay then allows the administrator to use standard HTML forms to configure and later

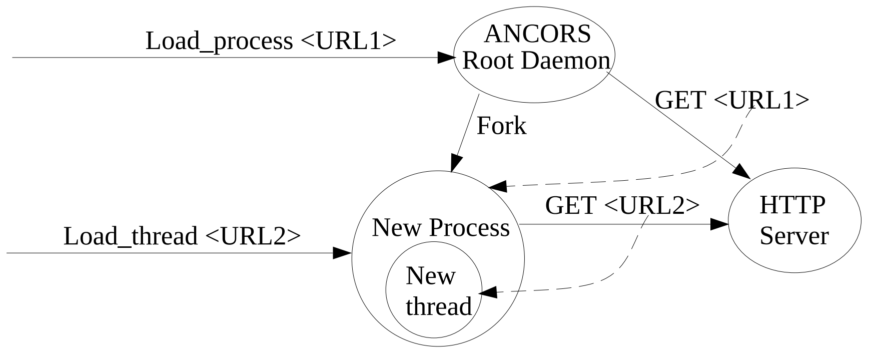


Figure 5: ANCORS daemons can spawn a new process or a thread within a process

interactively monitor the service through a browser. Yet another approach is for the service to respond by providing Java byte-code that can then be embedded either in the central management stations or in a distributed monitoring agent, using a scheme similar to the one proposed in [2]. Other interesting technologies that are being developed, and that may be included, are DRP [10] and NESTOR [15].

We think ANCORS's simple query mechanism can be the foundation for solving the problem of how to extend the monitoring and control capability as new network services are deployed. In addition to allowing backward compatibility, this scheme can support very simple and lightweight management solutions based on standard Web software, or it can also support more sophisticated solutions to bolster the power of network management as needed.

4 Implementation

We are currently prototyping most of the ideas we have outlined. We have primarily focused in implementing (1) a prototype of a root ANCORS daemon that dynamically accepts and instantiates network services, (2) a representative example of an engineering service that could be used to conduct very accurate, end-to-end quantitative experimentation, and (3) the integration of a monitoring and control service for intrusion detection [6].

4.1 ANCORS Root daemon

The current prototype of an ANCORS root daemon accepts commands to download network service from a remote location. The network service is specified as a URL; the ANCORS daemon, after downloading the service with an HTTP GET command, strips the HTML header from the received code and installs the service. As shown in Figure 5, the download command can either (1) trigger the ANCORS daemon to duplicate itself by using a fork system call to run the downloaded network service (as well to accept further commands), or (2) simply add a thread to an existing process. In either case, the downloaded code is initially accessed through a universally predefined entry point (**init()**). This initial configuration function can simply transfer control to the downloaded code for execution, or it can first gather runtime configuration data in a manner specific to each network service and then explicitly started. Our current system management prototype supports the deployment of native binary compatible code or Java applications. The deployment and configuration mechanism is fully backward compatible. Legacy services can be configured by downloading configuration files and specifying required command-line arguments so that existing software can be easily deployed without modification. In addition, we have implemented simple web-based configuration mechanisms for newly developed services. In these services, the configuration and monitoring functions are embedded inside the deployed service itself. These functions return HTML code that is fed to the network manager to gather some user-defined runtime parameters or for displaying usage data. The network engineer configures and monitors the services with HTML forms that are then pushed back through a CGI script to the created service. Each operation returns HTML forms that in turn may call other functions, thus allowing a hierarchical organization of HTML pages. Future extensions to our management system will allow the incorporation of existing NM software based on SNMP and Java and the creation of new decentralized management solutions based on the concept of delegation (perhaps using Java as the delegation language).

The ANCORS daemon offers a set of built-in primitives to the downloaded services that, in addition to standard native system functionality (I/O, memory management, networking), provide (1) multithreading (nonpreemptive), (2) LAN multicast emulation, and (3) global time synchronization. These primitives can provide support for distributed simulation network engineering applications, as well as some forms of sophisticated network monitoring.

4.2 Virtual Networking Using ANCORS

To date we have produced a representative example of an engineering network service that emulates a Unix kernel. The service was obtained by modifying a Linux operating system to allow its execution in user mode. The modifications of the operating system replaced all lower-level, hardware-dependent procedures and interfaces with user-level counterparts. We deleted the file system support and incorporated all necessary configuration procedures (like ifconfig and route) into the service itself as configuration functions. Memory management was completely deleted and replaced by user-level memory allocation functions (malloc and free). The scheduling was also completely replaced by nonpreemptive threading offered by the simulation package (CSIM [8]).

The resulting service executes in a virtual timescale, offers the identical networking behavior of a real Linux kernel and can therefore be used as a vehicle to instantiate high fidelity distributed simulations of virtual networks [7]. One of the model's configuration functions accepts several different timing configurations to approximate the protocol stack timing behavior of four different kernels (SunOS 4.13, SunOS 5.5, Linux 2.02, and BSD 2.2).

The virtual kernel offers the network application programming interface (API) of the real Linux counterpart and therefore can be used to reproduce a wide range of loading conditions. ANCORS's ability to add and delete threads can be used in this application to dynamically

For example, it could use a MIME-encapsulated document to specify configuration and monitoring operations to be performed by the management system.

All products and company names mentioned in this paper are the trademarks of their respective holders.

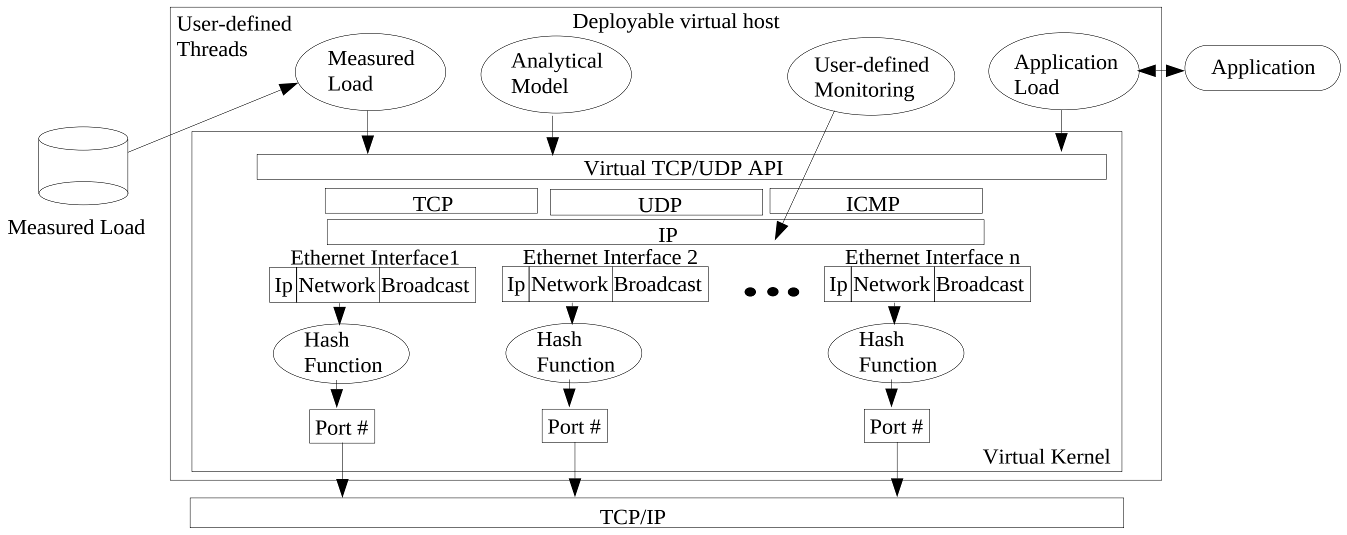


Figure 6: Deployable Virtual Host

change loading conditions (by adding or deleting user-defined loading threads) or by injecting user-defined monitoring probes into the kernel so that specific parameters can be observed. For the time being we have implemented some simple load models borrowed from classic queuing theory. As shown in Figure 6, the user-definable loads may be produced by either closely mimicking real load conditions recorded by network monitoring services or by linking some real applications to the virtual kernel to generate application-specific loads (perhaps originating from a real-time video stream).

The virtual kernels communicate with each other through TCP, and automatically configure themselves to participate in emulated multicast sessions that parallel the behavior of virtual Ethernet segments. Initially, all real hosts are aware of all other real hosts that may share the same virtual Ethernet segment. Each virtual Ethernet segment network address assigned to a virtual host is transformed through a hash function into a port number. When the virtual kernel initializes its virtual interfaces, the multicast emulation initialization procedure tries to connect to all known peers that may share a virtual Ethernet segment using the port associated with each virtual interface. Thus, if two or more virtual hosts share a virtual network address, and therefore use the same port, they establish a TCP connection used to tunnel virtual Ethernet packets. When a virtual host sends a simulation packet pertaining to a particular virtual Ethernet segment, it sends it to all virtual hosts that have connected to the associated port.

The deployment of a virtual network is achieved by downloading and configuring several virtual kernels through ANCORS daemons. All these operations can be performed either through a standard HTML browser or by using a script. We have so far instantiated several virtual networks running on a network of workstations including Sun SPARCstation 20s, UltraSPARCs, and Intel-based machines running BSD and Linux operating systems. Our experiments have, so far, only verified the behavioral semantics of our virtual network and we plan to conduct representative performance experiments to explore some interesting quantitative network design issues.

4.3 A Monitoring and Control Service for Intrusion Detection

We are also developing a prototype downloadable intrusion-detection module for ANCORS, which represents a monitoring and control service at the data layer. This intrusion-detection module can be dynamically configured and deployed to any network element that hosts an ANCORS daemon, and can then return analysis results to the assessment layer for correlation and perhaps control-layer responses. These dynamically distributable intrusion-detection service modules represent a significant departure from the previous centralized host-based, user-oriented, intrusion-detection efforts that suffer poor scalability and integration into large networks. The intrusion-detection services are adaptations of separate intrusion-detection research tools developed by SRI under the EMERALD (Event Monitoring Enabling Responses to Anomalous Live Disturbances) project, and represent one of the first distributed network surveillance security mechanisms that will fully integrate into an active network paradigm.

The ANCORS intrusion-detection service module consists of event filters, capable of interacting with the analysis target to analyze a variety of operational data, including audit data, system- or application-layer activity logs, and network traffic. The operational data is forwarded to the intrusion-detection module's analysis engines, which consist of both a signature analysis (an expert-system) engine and a statistical anomaly detection engine. Details of the architectural structure and analysis capabilities of the ANCORS intrusion-detection service can be found in [6].

Fundamental to the intrusion-detection module's ability to support rapid deployment to a variety of ANCORS hosts, and to conduct a variety of analyses on heterogeneous event streams, is the implementation of a pluggable configuration library that allows strong separation between the analysis semantics and the code-base. The ANCORS intrusion-detection module consists of two major components, the analysis code-base, and a dynamically pluggable configuration library called the *resource object*. The resource object is highly target specific, containing all of the operating parameters for the intrusion-detection module, as well as the native methods needed to retrieve and format the target event stream. As intrusion-detection modules are deployed from one analysis target to another, no code modifications are necessary. Rather, one must construct, or select from a preconstructed library, the resource object appropriate for the target host and analysis objectives. Upon deployment, the ANCORS daemon receives both the intrusion-detection code module and a resource object, which it instantiates to provide dynamically deployable and customizable surveillance. A detailed discussion of the ANCORS intrusion-detection module's abilities to detect misuse and other exceptional activities on TCP/IP gateway machines can be found in [5].

5 Conclusion

As the dynamic deployment of networking services becomes standard technology to support user applications, network operators will require an efficient and flexible infrastructure to assist them in network design, configuration, and monitoring. The quality of future network management, monitoring, and engineering tools and standards will be crucial in determining the speed at which networking will evolve toward a more dynamic architecture. In ANCORS, network monitoring, control, and design can coexist in an integrated paradigm. The synergy of combining distributed simulation, network monitoring and control, and active networking will dramatically increase the power of network management and engineering. We have shown how a unified, yet very extensible, system management framework can be derived from current Web technology to provide compatibility with legacy standard (SNMP) and virtually unlimited extensibility to introduce more powerful management technologies as they become available. We have also described some services we have implemented that can be used in the context of our new network management framework.

In the near future, we plan to bridge our work with existing active networking technologies to provide an integrated platform for merging data transport protocols and their associated deployment mechanisms with our extensible engineering and management support. In addition, we will use our infrastrucuture to conduct network engineering experiments to advance the understanding of end-to-end network behavior and offer a user-friendly environment for the development of new network technologies.

References

[1] D. Scott Alexander, Marianne Shaw, Scott M. Nettles, and Jonathan M. Smith. Active bridging. *To appear in the Proceedings of the ACM SIGCOMM'97 Conference*, Cannes, France, September 1997.

[2] F. Barillaud, L. Deri, and M. Feredun. Network management using internet technologies. *Integrated Network Management V*, San Diego, 1997.

[3] L. Peterson J. Hartman, U. Manber and T. Proebsting. Liquid software: A new paradigm for networked systems. Technical Report 96-11, University of Arizona, 1996.

[4] U. Legedza, D. J. Wetherall, and J. V. Guttag. Improving the performance of distributed applications using active networks. *Submitted to IEEE INFOCOM'98*, 1998.

[5] P.A. Porras and A.Valdes. Live traffic analysis of tcp/ip gateways. *To appear in Proceedings of the Network and Distributed System Security Symposium*, San Diego, March, 1998.

[6] P.A. Porras and P.G. Neumann. Emerald: Event monitoring enabling responses to anomalous live disturbances. *Proceedings of the National Information Systems Security Conference*, Baltimore, MD, October, 1997.

[7] L. Ricciulli. High-fidelity distributed simulation of local area networks. *To appear in Proceedings of the 31st Annual Simulation Symposium*, Boston, April, 1998.

[8] H. Schwetman. Csim: A c-based, process-oriented simulation language. Technical report, MCC, 1989.

[9] Jonathan Smith, David Farber, Carl A. Gunter, Scott Nettle, Mark Segal, William D. Sincoskie, David Feldmeier, and Scott Alexander. Switchware: Towards a 21st century network infrastructure. *http://www.cis.upenn.edu/ switchware/papers/sware.ps*, 1997.

[10] Arthur van Hoff, John Giannandrea, Mark Hapner, Steve Carter, and Milo Medin. *The HTTP Distribution and Replication Protocol.* http://www.marimba.com/standards/drp.html, August 25, 1997.

[11] Arthur van Hoff, Hadi Partovi, and Tom Thai. *Specification for the Open Software Description (OSD) Format.* http://www.microsoft.com/standards/osd/, August 11, 1997.

[12] D. J. Wetherall, J. V. Guttag, and D. L. Tennenhouse. Ants: A toolkit for building and dynamically deploying network protocols. *Submitted to IEEES OPENARCH'98*, 1998.

[13] Y. Yemini and S. da Silva. Towards programmable networks. *IFIP/IEEE International Workshop on Distributed Systems: Operations and Management*, L'Aquila, Italy, October 1996.

[14] Y. Yemini, G. Goldszmidt, and S. Yemini. Network management by delegation. *Second International Symposium on Integrated Network Management*, Washington DC, April 1991.

[15] Y. Yemini, A. V. Konstantinou, and Danilo Florissi. Nestor : Network self management and organization. *http://www.cs.columbia.edu/dcc/nestor/*, 1998.