The following paper was originally published in the
Proceedings of the USENIX Symposium on Internet Technologies and Systems
Monterey, California, December 1997

Secure Public Internet Access Handler (SPINACH)

Elliot Poger, Mary G. Baker
Computer Science Department, Stanford University

For more information about USENIX Association contact:

1. Phone:	510 528-8649
2. FAX:	510 548-5738
3. Email:	office@usenix.org
4. WWW URL:	http://www.usenix.org/

Secure Public Internet Access Handler (SPINACH)

Elliot Poger, Mary G. Baker
Computer Science Department, Stanford University
{elliot,mgbaker}@mosquitonet.stanford.edu

Abstract: *This paper describes a system that controls access to computer networks through publicly accessible LANs, enabling network administrators to authorize users either on a permanent or occasional basis. The system has been designed with minimal assumptions about the software and hardware required of users, and requires very little specialized equipment within the network infrastructure. We enumerate the requirements for such a system, describe the design and implementation of the system, and note tradeoffs between security and efficiency.*

1. Motivation

In early 1996, Stanford University completed a new building to house its Computer Science Department. The new building includes Ethernet ports in every office, as well as in various public spaces: meeting rooms, lobbies, and lounges. Unfortunately, 18 months after the building opened, concerns about unauthorized users tapping into the department network have prevented the activation of network connections in publicly accessible areas (“public ports”). Similar problems plague many other buildings, especially on college campuses, where the desire for mobile connectivity is high but physical security is lax. Even though building designers had the foresight to include network connections in many parts of these buildings, political and security considerations have led to a frustrating waste of potential network connectivity. Those who desire network connectivity in public parts of the building are forced to use wireless network connections, which are often slow and expensive.

There are several reasons Stanford University, and the Computer Science Department in particular, do not want to allow unknown users access to the building network. Most importantly, we do not want to allow rogue users to attack other computers connected to the building network in offices and labs. Although hackers can already attack department computers over the Internet, we do not want to make these attacks, as well as eavesdropping on network traffic, any easier by allowing them access within our network. Also, some network services outside our department use the source IP address of transmissions to grant access. For example, some Internet services have been licensed for use at Stanford University and are made available to any host with a Stanford IP address, and we are obligated to prevent abuse of these licenses. In general, we want to minimize the chances that someone will misuse the Internet from a Stanford IP address, and if this misuse does occur, we want to identify the perpetrator so that we can hold him accountable. Perhaps less of a concern is that of bandwidth—we don’t want to allow unauthorized users to degrade everyone else’s service in the building by using network bandwidth to which they are not entitled. Since physical security in the building is minimal, as it is in many universities, libraries, and public institutions, we need a mechanism for restricting access through public network ports if these ports are to be activated.

Once we have an access control mechanism in place, we can allow specifically authorized users to connect to the high-bandwidth wired network in the building from public ports without compromising network security. To provide this access control, we have constructed the *Secure Public Internet Access Handler* (SPINACH). In SPINACH, a self-configuring router controls per-user access from a public subnet to a private one, using Kerberos or a similar mechanism to authenticate users and provide an audit path before users are granted access. With the exception of one custom software component on the router, SPINACH uses only standard protocols and software and requires only minimal software (telnet or web clients) on users’ machines.

The SPINACH system establishes a “prisonwall,” controlling the flow of packets between those hosts connected to public ports and the rest of the building network. As opposed to a firewall, which protects machines *inside* a particular network from malicious users *outside* the network [2][4], the prisonwall protects machines *outside* one portion of a network by refusing to forward packets that come from unauthorized hosts within. As users within the prisonwall authenticate themselves and thus activate network access for their hosts, SPINACH maintains an audit trail so that the users can be held accountable for traffic they generate on the network.

SPINACH has been designed with minimal assumptions regarding the network hardware available as well as the software installed on users' machines, so that it can be installed in a wide variety of institutions and require little ongoing oversight from network administrators. As such, it does not provide as high a level of security as some access control systems; however, it provides a useful level of security without requiring expensive network equipment or custom client software, and thus may be the most appropriate method of access control for some networks.

In this paper, we describe the design and implementation of the SPINACH system. Section 2 outlines the system requirements and policies. In Section 3, we describe the interfaces through which network users and administrators interact with SPINACH. Section 4 discloses the details of how we implement these policies and interfaces. The remainder of the paper describes the security tradeoffs in SPINACH, other systems with aims similar to ours, some possible future improvements to SPINACH, and conclusions we have drawn through this research.

2. System Requirements, Policies, and Definitions

The SPINACH system has two major functions: it controls the passage of network communications between public ports and the rest of the building network, and it provides a mechanism for unknown users to prove themselves as authorized so that they can have full network access. Both functions are implemented on the same network host, the *SPINACH router.* This section describes the requirements that the SPINACH router must fulfill, and the facilities that must be present within the network infrastructure and on hosts connected to public ports in order to implement both functions. SPINACH has been designed to require no special software on computers that users connect to public ports, and to require as little as possible of the network infrastructure, so that it can be deployed in any network installation with minimal expenditure of time and money.

2.1 Network Arrangement

The SPINACH system consists of a collection of public network ports on one or more LANs. These LANs are connected to the surrounding network infrastructure through a SPINACH router. The SPINACH router, an IP-routing Unix host (fully described in Section 4.1), forwards data packets between hosts on these public LANs and the outside networks. For routing purposes, hosts connected to the public ports are grouped into one or more IP subnets.

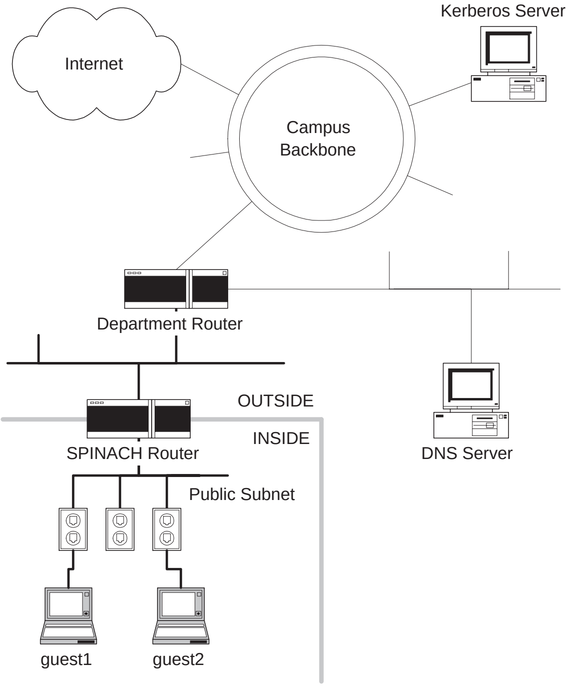

FIGURE 1. Network and security arrangement of the SPINACH system. The gray line running through the SPINACH router illustrates the prisonwall boundary, which separates the public subnet (inside) from the network as a whole (outside).

In our deployed SPINACH prototype (see Figure 1), the public ports are Ethernet ports located in publicly accessible areas of our building. These Ethernet ports are connected by a VLAN switch, so that data flows between them as if they were on the same LAN segment. Hosts connected to the public ports (labeled as "guest1" and "guest2" in Figure 1) are assigned addresses from one subnet, which we refer to as the "public subnet." The SPINACH router is connected to the same VLAN so it can route packets between the public subnet and the rest of the building network. In other SPINACH installations, some type of LAN other than Ethernet could be used, more than one LAN could be used to connect the public ports, and hosts could be arranged into more than one IP subnet, but for the purposes of this paper we assume the arrangement of our prototype system. Changing these parameters would require slight modifications to the routing and filtering software on the SPINACH router, but the system would function in basically the same way. For example, even a wireless LAN such as WaveLAN could be used for the public subnet, so long as the SPINACH system software were modified to accept WaveLAN, rather than Ethernet, link-layer addresses.

Figure 1 also shows the department Domain Name Service (DNS) server and campus Kerberos server. The Kerberos server provides authentication services for users affiliated with the University. Some other authentication service could work as well, with modifications to the user-authorization software on the SPINACH router; in this paper, we assume the use of Kerberos. The DNS server is needed for hosts on the public subnet to find the IP address of the campus Kerberos server.

Because all packets that travel between hosts on the public network ports ("inside the prisonwall") and hosts elsewhere ("outside the prisonwall") must be forwarded through the SPINACH router, the SPINACH router can filter out all packets that are deemed dangerous. The SPINACH router creates a security boundary between the public Ethernet ports and all other networks.

2.2 Security Policy

Being a research institution, we do not want to squelch the development or use of new network applications by instituting overly specific rules regarding exactly what traffic is allowed on the public subnet [5]. Thus, rather than taking the typical firewall approach by allowing only the use of certain prescribed protocols through proxies running at the security boundary, we filter traffic on a per-*user* basis. We restrict use of the network through public ports to those people whom we can hold accountable for their actions. The SPINACH router allows these trusted users unrestricted access to the network and prevents untrusted users from accessing the network at all.

Traffic to and from hosts within the public subnet can be divided into three types. *Outgoing* traffic travels from within the public subnet to hosts outside. *Incoming* traffic comes from hosts outside the public subnet and is destined for hosts within. *Internal* traffic moves between two hosts on the public subnet. The SPINACH router uses different packet-filtering policies for incoming and outgoing traffic, following a particular set of rules to determine whether a given packet will be forwarded towards its destination or dropped. Internal traffic is not affected by the SPINACH router at all.

The SPINACH router forwards all *outgoing* traffic from those hosts on the public subnet which a user has authorized using the procedure described in Section 3.2. All outgoing packets from unauthorized hosts are dropped, except packets addressed to the trusted DNS or Kerberos server; this traffic is necessary for hosts within the public subnet to authorize themselves. Once a user has authorized a host on the public subnet, the SPINACH router forwards all outgoing traffic from that particular host. An audit trail which records the identity of the user who authorized this host enables network administrators to hold the user accountable for any malicious traffic that originates from this host.

The SPINACH router forwards all *incoming* traffic, because we are solely concerned with hosts inside the prisonwall wreaking havoc upon the rest of the network, rather than the reverse. Information coming into the prisonwall from outside is not considered a security threat, because it is assumed that any hosts inside the prisonwall that are trying to extract secret information from outside machines would have to initiate such transactions from within the prisonwall, and unauthorized hosts are not allowed to send outgoing traffic in the first place.

The SPINACH router exerts no control whatsoever over *internal* traffic; these packets are carried directly from one public port to another through the LAN which connects them. Thus, any hosts that are connected inside the prisonwall must tolerate a hostile network environment.

In addition to policies regarding the awarding of network access to users, there must be policies regarding the removal of network access. At present, the SPINACH router authorizes network access for four hours at a time; the length of this timeout is a parameter we plan to experiment with, as described in Section 7. If a user wants to remain connected to the network for longer than this period, he must re-authorize his connection using the procedure described in Section 3.2.

2.3 Types of Users

In many SPINACH installations, it will be appropriate to group users according to the access permissions that should be granted to them, as well as the resources that are available to authenticate them. In our prototype installation here in Stanford's Computer Science Department, we have identified three such types of users: "Department Users," "University Users," and "Guests."

Department Users already have access to the building network in private offices and labs, but desire to connect temporarily in another part of the building, for example, to check e-mail while sitting in a conference room or lounge. Since they already have access to the building network, but simply want to connect in a different physical location for convenience, we should have no security concerns about allowing them to connect through public ports. Also, Department Users already have authentication records in the campuswide SUID (Stanford University Identification) database.

University Users already have access to Stanford's computer network in the public computer labs, and perhaps in the residence halls, but do not presently have the ability to connect to the network within the Computer Science building. System administrators within the CS Department are rightfully concerned about allowing them unrestricted access to networks within our building that they have not been able to use in the past. Like Department Users, University Users already have entries in the SUID database.

Guests are not in the SUID database and thus do not currently have the ability to access Stanford's network at all. Typically this group contains visitors from industry and other universities who are in the CS Department to meet with professors and students or attend symposia. Quite often these visitors bring their own laptop computers and would like to connect to their home networks through the Internet to access their e-mail or retrieve files. Before the implementation of the SPINACH system, there was no established mechanism for allowing these short-term visitors network resources, so guests have been forced to use low-bandwidth, high-cost wireless connections or informally borrow the use of a desktop machine in some willing person's office. Because relationships with these outsiders are important to Stanford, we should provide a mechanism for them to utilize our network resources in some reasonable way while they are visiting.

In general, different types of users may be extended different access rights on the network, at the discretion of the network administrator. In our case, due to the concerns of department network administrators, University Users are currently denied network access; Department and authorized Guest users are allowed unrestricted network access.

2.4 Hardware and Software Requirements of the Client

Especially because we have the various classes of users described above, it is important that we support many different configurations of hosts with minimal assumptions about the software present on these machines. Even University and Department users have a variety of platforms: DOS, Windows 3.1, Windows 95, Macintosh, and various flavors of Unix. We cannot foresee all platforms visitors from off-campus will use. Thus, writing and maintaining special network access software for such a large and growing number of platforms would be a burden on our network administrators. Also, visiting users would need to install this custom software on their computers to use our system, and that could be a hassle for them. We would thus like to rely solely on client software that most users will already have installed on their networked computers, or can easily obtain from other sources.

We *can* assume that the user's computer has some basic network software on it, since the user presumably has been using it to connect to some other network. Almost all networked computers will have either a telnet client or a web browser; if a visitor's computer has neither of these, they can most likely obtain one easily from a number of sources. (Our prototype system requires users to run a telnet client; an alternative web interface is currently under construction.) In addition, an increasing number of networked computers have Dynamic Host Configuration Protocol [3] (DHCP) and/or Kerberos [8] clients—for example, the widely-used Windows 95 operating system includes DHCP client software. In the design of our access restriction system, we require only a telnet client on the visitor's computer; if a DHCP or Kerberos client is present, we use it to simplify the configuration and authorization processes.

2.5 Requirements of the Network Infrastructure

Although it is less of a concern than the minimal software requirements on the client end, we also want to minimize the amount of maintenance overhead on the SPINACH router and elsewhere in the network. The less of a burden we place on network administrators, the less resistance we will encounter in deploying our system both within our department and in other institutions.

We take advantage of the existing campuswide Kerberos authentication service, as well as the departmental DNS server, to simplify some users' connection process as

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**