Security Aspects in Wireline and Wireless IP

Mathias Johansson and Jonas Rutström
Signals and Systems, Uppsala University,
PO Box 528, SE-75120, Uppsala, Sweden.
E-mail: Mathias.Johansson@signal.uu.se
Jonas.Rutstrom@signal.uu.se
URL: www.signal.uu.se

Abstract

This paper considers security mechanisms for authorizing and monitoring IP connections over wireline as well as wireless networks. The objective is to minimize the risk of an unauthorized user gaining access to a network and to prevent an intruder from stealing a connection from an authorized user.

A system, which we call Network Access Login/Authentication Polling (NAL/AP), has been developed for wireline networks. It is based on polling of the authorized connections.

Wireless mobile IP demands a somewhat different approach and a method, given the name Network Access Login/Authentication Polling by Challenge-Response (NAL/APCR), based on periodic challenge-response requests, is presented.

I. Introduction

The evolution of mobile computers has created a new era in the computer-security industry. Security enters a new dimension when computers are not tied to a specific location. Any local network that is not set up properly can be an access point for anyone with a mobile computer. This problem is amplified when mobile computers operate in wireless environments. In the future, users can be located virtually anywhere and still have access to networks. This requires a strong security awareness.

A network administrator thus needs a security policy. The security policy shall define the limits of acceptable behaviour, and what the response to violations should be. Naturally, security policies will differ from organisation to organisation. The underlying platform and development tools should be chosen accordingly.

We will investigate the possibility for an unauthorized host to gain access to a network, and propose means for solving this problem in a specific scenario. In particular, we will propose a solution to:

1. Prevent an unauthorized user from receiving an unused IP address from a network.
2. Prevent an intruder from stealing an IP address from an authorized user.

We do not consider denial-of-service attacks, nor do we attempt to prevent eavesdropping. The objective is to provide means for continuous authentication. This problem has been studied earlier in the literature and several solutions exist for wireline networking. For example, see [4].

We will first describe our solution for wireline networks based on user authentication and continuous polling. Then we proceed by presenting a different approach for wireless mobile networks, designed to minimize additional traffic load while increasing network security.

II. The Wireline Solution

A. Implementation

We have implemented a solution to the problem stated above, based on a class C Ethernet network. The security system is based on Redhat Linux[1]. An overview is given in Figures 1 and 2. A firewall is implemented by the use of *ipchains*[2], a powerful tool to build packet-filtered firewalls. The main components of the system are implemented in Java which has given us the opportunity to create a homogeneous solution with seamless connections between the modules.

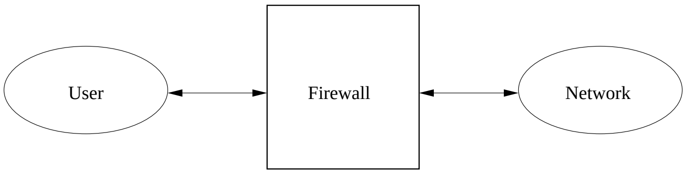

Fig. 1. All user traffic is filtered through a firewall.

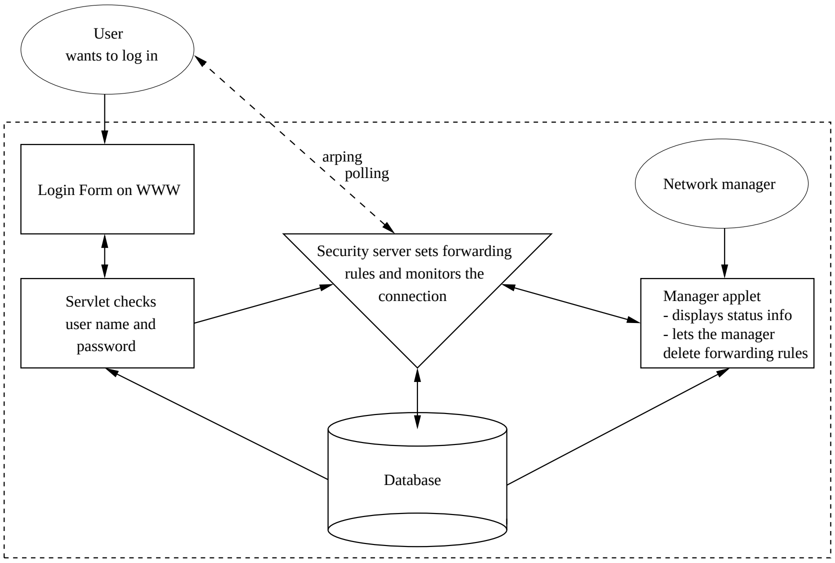

Fig. 2. Overview of the security system which implements the firewall.

The security policy for our system states that *no user should be able to gain access to the network without passing a user authentication process.* The following two scenarios must be protected:

1. The user wants to connect his/her laptop to a private network. This is easily done by connecting the computer to an available Ethernet socket and retrieving an IP address by, for instance, DHCP [9]. Thus, the user gains unauthorized access to the network.
2. An authorized user is already logged in. In an unguarded moment, an intruder physically disconnects the authorized host and takes possession of the already authenticated connection with mailicious intentions.

The so described scenarios will be prevented by a login and control process. This process, which we call NAL/AP (Network Access Login/Authentication Polling), works as follows: When a new user wants to connect a laptop to the network he/she must pass an

[1] See *www.linux.org*, *www.linux.com* and *www.redhat.com*.
[2] Linux *ipchains* is a rewrite of the Linux IPv4 firewalling code. It is required to administer the IP packet filters in Linux kernel versions 2.1.102 and above.

authentication test where login name and password are requested. By default the user can only access the WWW port of the security server, everything else is forbidden. If the user passes the security test he/she will be allowed to use the network resources according to the security policy. If not, the user is denied further network access.

To prevent an unauthorized user from stealing an authorized user's IP address, we need to continuously (several times per second) verify that the authenticated host is connected. If the status of the active IP address changes, then the user is forced to a reauthentication.

Our solution comprises of the following components:

- A WWW interface that handles the Internet login
- A security server that manages the firewall and monitors the network connections
- A database to store the user information
- A user interface that gives the administrator feedback on the state of the network.

The WWW interface allows the user to identify himself. It is implemented as a Java servlet[3] and asks the user for login name and password. When the user is authenticated, the security server is requested to give the host network access.

The security server fulfills two important tasks. First, it manages the packet-filtered firewall through *ipchains*. A new firewall rule is inserted for each user passing the authentication test. This rule allows the user to access the network. When the user ends his/her network session the rule is deleted.

Second, the polling of authenticated users is performed by the use of *arping*[4]. If a request times out, then the forwarding rule is deleted and the user has to reauthenticate himself. Notice that this does not involve any encryption. We only check if the host responds to the echoing request. This is the reason why we poll the host continuously after authentication. After this, until a connection is broken, no further authentication is required.

The security server is also responsible for updating the database with session statistics for the monitored connections.

In our implementation, a Java applet informs the security administrator of the current status of every IP address in the network. This information is requested from the database on notifications by the security server.

B. Discussion

The obvious drawback of our solution is that it will generate redundant non-user traffic over the monitored IP connections. This is a necessary consequence of having to monitor the connections continuously[5].

Another potential drawback with our solution is the use of a login script that is accessed from the Internet. The Internet as such is an open and insecure network. Therefore care has to be taken to ensure that the password and user login may not be eavesdropped. This problem can be overcome by the use of SSL (Secure Sockets Layer) [6] and TLS (Transport Layer Security) [5][6].

In a realistic network situation with a large network generating bursty traffic, we may expect the polling to timeout even without any malicious interference. A user may be forced to reauthenticate himself due to net congestion and packet collisions. This flaw is serious but hard to avoid. If we could let the authentication take place on a lower

[3] Java servlets are server side scripts that generate HTML code. See *http://java.sun.com/products/servlet/index.html*

[4] Arping works by sending echoing requests by ARP packets to a host through a specified output interface.

[5] Note that the word *continuously* is being used in a rather loose context here. The polling is done at discrete time intervals, but they are made very small to preclude any connection interrupts resulting from hostile takeovers.

[6] Modules for SSL and TLS exist for most web servers.

level without user interaction, then the problem would be somewhat alleviated. Such a solution would however require some sort of agent installed on the host which would be highly undesirable in several aspects. The issue of trust becomes apparent if we let agents perform the authentication, and further, we do not wish to place any additional demands on the hosts. The system should work with any computer configuration to minimize the possibility of back doors and to maximize the platform independence and installation simplicity.

III. The Wireless Solution

With respect to information security, mobile hosts are by nature more vulnerable than their stationary counterparts. The Internet Mobile Host Protocol (IMHP) [1] is designed to support three different security models: no security, weak security, and strong security. The weak security model corresponds to a security level comparable to that of today's Internet. We shall assume that this is the model used in our scenario.

Under weak security, two main scenarios are being protected through authentication. First, a home agent and a mobile node shares a secret in order for the home agent to authenticate binding requests from the mobile node. Second, other IMHP entities authenticate bindings which are received in binding notifications[7]. Such entities may not use shared secrets, instead a binding request is sent to the home agent along with a random value, as an authenticator for the Binding Notify message. Under weak security, if the reply contains the same authenticator, the included binding can be considered authentic.

The weak security model explains under which assumptions the proposed solution will work. As we can see, no mechanism prevents a malicious user from taking possession of a network connection from a mobile node when the host is temporarily unreachable by the authentication controller. In this case, an intruder will not be forced to authenticate its illegally acquired link. This problem could be solved with an extension to the IMHP protocol or by a high level mechanism.

In forming a solution for the given scenario, we must take into account the special characteristics of the wireless medium. In particular, bandwidth should be used more conservatively. Continuously sending small packets of data is not a viable solution for a bandlimited medium. Also, a polling message as described in Section II. is quite likely to timeout due to the mobile environment and in NAL/AP all connection breakups require reauthentication.

We propose a solution using a mechanism very similar to CHAP [8] using MD5 [7]. We call our approach NAL/APCR (Network Access Login/Authentication Polling by Challenge-Response).

NAL is performed just as described in Section II. by a home agent (as defined in [2]) or by some other entity on the home network. The entity which authorizes connections and performs the continuous identity control is referred to as the Authenticator.

For continuous identity control, APCR uses a 2-way handshake that is repeated at slightly randomized time intervals:

1. A challenge (as defined by [3]) is sent from the Authenticator to the mobile node.
2. The mobile node responds with a value calculated using a one-way hash function.
3. The Authenticator checks the response with its own calculation of the hash value. If the values match, then the authentication is valid, otherwise the connection should be terminated.
4. At random time intervals, the Authenticator sends new challenges to the mobile node, and repeats steps 1 to 3.

[7] Binding notifications are sent by the IMHP registration protocol in order to tell previous foreign agents about the mobile node's new location [1].

The challenge-response packet format closely follows the CHAP packet format (see [8]). If the authentication mechanism is implemented at the application layer, then the format could be somewhat simplified. The two most important fields are the *Identifier*, which must be changed each time a Challenge is sent, and the *Value*, which in a Challenge packet consists of a unique value such that the following values are unpredictable. In a Response packet, the Value consists of a one-way hash value calculated over the Challenge Identifier concatenated with a shared secret followed by the Challenge Value. The shared secret is suggested to be the password used in the Network Access Login phase.

In contrast to the wireline solution, a timeout will not lead to immediate rejection of the host. Instead, new challenges are sent until a retry counter expires, requiring the host to reauthenticate itself. This adds some robustness to the proposed solution.

It is important to notice that it is not the foreign agent (as defined in [2]) that carries out the authentication, as this would require a complex protocol for handing over the authentication to a new foreign agent. The Authenticator is always the same entity, whether it is the home agent or a separate entity on the home network.

IV. Conclusions

In NAL/AP, a high level of security is achieved with respect to continuous authentication, though care has to be taken when implementing NAL over an open network. When designing a continuous authentication protocol for wireless networks, NAL/AP is considered too expensive in terms of additional authentication traffic and is expected to cause too many connection disruptions due to natural fluctuations in wireless transmitting conditions.

NAL/APCR is introduced as a slightly more complex method compared to NAL/AP, but with properties that better suit a wireless network environment. In APCR, the period between new challenges together with the timeout interval of each Challenge-Response and the maximum number of retries regulate the period of time that an intruder may gain unauthorized access to the network. This parameter also governs the additional amount of traffic burden that continuous identity control places on the wireless network. Hence, we have to make a trade-off between the level of security and the use of network bandwidth. This is a natural consequence of the problem and should come as no surprise. Our objective however, has been to minimize additional network use while maximizing security. NAL/APCR allows the security/network designer to make a reasonable compromise between those aspects.

This paper has presented an initial survey of a particular security scenario. However, a secure system must also solve more general problems. In our future research we aim to extend our efforts to include the area of denial-of-service attacks and secure connections.

References

[1] C. Perkins, A. Myles, D. Johnson, “The Internet Mobile Host Protocol (IMHP)”, Proc. INET ’94 / JENC5 (642), 1994.
[2] C. Perkins, “IP Mobility Support”, IETF RFC 2002, 1996.
[3] C. Perkins, P. Calhoun, “Mobile IP Challenge/Response Extensions”, IETF Mobile IP Working Group, Internet Draft, October 1999.
[4] “The Bifrost Network Project”, http://www.data.slu.se/bifrost.
[5] T. Dierks, C. Allen, “The TLS Protocol Version 1.0”, IETF RFC 2246, 1999.
[6] “Introduction to SSL”, http://devedge.netscape.com/docs/manuals/security/sslin/contents.htm, 1998.
[7] R. Rivest, “The MD5 Message-Digest Algorithm”, IETF RFC 1321, April 1992.
[8] W. Simpson, “PPP challenge handshake authentication protocol (CHAP)”, IETF RFC 1994, August 1996.
[9] R. Droms, “Dynamic Host Configuration Protocol”, IETF RFC 2131, March 1997.