

PROJECT PROFILE

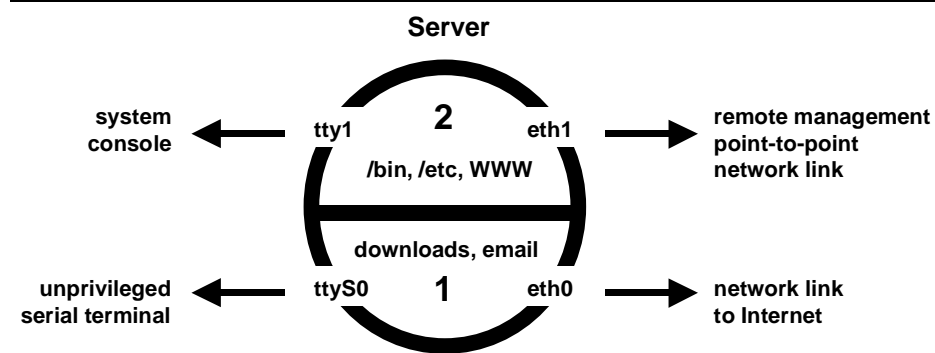
LOMAC: Low Water-Mark Integrity Protection for Linux

Problem

The increasing presence of integrated Internet functionality in Commercial Off-The-Shelf (COTS) applications is exposing greater and greater numbers of formerly isolated systems to malicious code and remote users. To offset this exposure, these systems need the kind of reliable integrity protection made possible by kernel-resident mandatory (“non-discretionary”) access control technology. Unfortunately, traditional implementations of this technology have focused on goals other than compatibility, hindering its adoption by requiring users to give up their existing investment in COTS operating systems or by interfering with the functionality of mission-critical COTS applications. LOMAC is an attempt to implement a kernel-resident access control enhancement for existing Linux-based systems that is sufficiently simple and COTS-compatible to encourage widespread adoption.

Solution

Instead of replacing existing COTS Linux kernels, LOMAC augments them by implementing a form of Low Water-Mark kernel-resident mandatory access control in a Loadable Kernel Module (LKM). Linux users can dynamically load this LKM just as they would a driver for a new device. In order to avoid costs that might hinder its adoption, LOMAC is designed to be compatible with existing COTS applications and configurations, and provides some widely applicable integrity protection even in its simplest default configuration.



Details

LOMAC provides protection by dividing a system into levels of increasing integrity and preventing the movement of potentially malicious users, code, and data from low-integrity levels to higher levels. The diagram above shows how the default configuration of LOMAC divides all of the files and devices on a Linux server (shown as a circle) into two levels numbered 1 and 2. Level 1 (shown as the bottom half of the circle) is for files and devices devoted to low-integrity data, such as potentially viral email attachments freshly arrived from the Internet. Level 2 (shown as the top half) is for files and devices dedicated to high-integrity data, such as system binaries and configuration information. In the event of a level-1 compromise, level 2 remains safe. LOMAC's primary responsibility is to ensure that (possibly viral) data does not flow from low-level

Research Focus

The Challenge of Compatibility

LOMAC's emphasis on compatibility with COTS applications and unmodified Linux kernels distinguishes it from previous attempts to implement mandatory access control in UNIX kernels. This emphasis led to a re-examination of the traditional access control models, based on compatibility rather than quality-of-protection criteria. Surprisingly, this examination identified the relatively obscure Low Water-Mark model as the access-matrix-based scheme most likely to support a highly-compatible Linux implementation.

Its key feature was its ability to automatically assign the appropriate privilege set (level) to any given subject (process) based solely on observing its observation behavior during run-time. (See the discussion of “demotion” in the Details section, left.) This ability frees LOMAC from the need for assumptions about the “roles” of users, the purposes of programs, and the use of role-choosing menus found in implementations of other more complex models, such as Type Enforcement and Clark-Wilson. Such assumptions and additional menus are not part of standard COTS Linux systems, and represent potential incompatibilities.

Further discussion of these issues can be found in “LOMAC: Low Water-Mark Integrity Protection for COTS Environments” in the proceedings of the 2000 IEEE Symposium on Security and Privacy.

- Tim Fraser
Principal Investigator,
Secure Execution
Environment Group

Details (continued)

objects (files) to high-level objects. Data flows when a subject (process, possibly a Trojan horse or the agent of a malicious user) reads from one object and subsequently writes to another. In order to prevent flows from low-level objects to high-level objects, LOMAC assigns a level to each object and subject upon creation. Once assigned, an object's level never changes. Subject levels, however, can change over time. Whenever a subject reads an object with a level lower than its own, LOMAC "demotes" the subject, reducing its level to match the object's. The Low Water-Mark model derives its name from this demotion behavior. Regardless of the level of the object it is currently reading, a subject's level is determined by the lowest-levelled object it read in the past - the "low water-mark" of its past behavior, so to speak. LOMAC derives its name from "Low Water-Mark" and "MAC," for Mandatory Access Control.

A subject's level determines its level of privilege - while LOMAC allows any subject to read any object, it prevents subjects from writing to objects whose levels are higher than their own. This simple scheme provides useful protection against viruses, Trojan horses, and users intent upon misuse. For example, consider a Trojan horse embedded in an object downloaded from the Internet. The default LOMAC configuration shown in the diagram assigns all downloaded objects a comfortably low level of 1. When the Trojan object is read by a particular subject, it might perform a stack smashing (buffer overflow) attack to force the subject to execute viral code at root privilege. To a Linux machine without LOMAC, this would be a grave situation indeed, since a viral subject with root privilege could infect any object it chooses. However, with LOMAC, the act of reading the level-1 Trojan object would have safely demoted the viral subject to level 1. Despite its root privilege, the viral subject would be unable to spread its infection by writing to any object with a level above 1. In the default LOMAC configuration shown in the diagram, these high-level objects include all system binaries, configuration files, and most other non-downloaded content as well.

LOMAC uses demotion to automatically and transparently ensure that those subjects which serve remote clients operate at an appropriately low level. In general, LOMAC assigns a new subject the same level as the subject who created it. In its default configuration, LOMAC assigns level 2 (the highest) to the first subject (the idle/init process), which initializes the Linux system by creating a new high-levelled subject for each system server application. These servers continue to execute at level 2 until they read from a (potentially dangerous) level-1 object and are demoted. It is important that those subjects which serve (potentially malicious) clients on the Internet be demoted to a comfortably low level as soon as they read their first client request. Once at a low level, LOMAC will be able to contain their misbehavior should they become compromised as described above. In order to ensure that this demotion occurs without user intervention (transparently) and without any help from the server applications (automatically), LOMAC assigns levels to all devices that provide access to the system, including console devices, serial lines, and Network Interface Cards (NICs). As shown in the diagram, LOMAC's default configuration assigns level 1 (the lowest) to the eth0 NIC leading to the Internet, causing subjects which serve Internet clients to be demoted upon reading their first client requests from the NIC. Hosts with more than one NIC can assign different levels to each, providing subjects which serve clients on internal networks with more privilege than those who serve more dangerous external clients. LOMAC does not treat NICs as objects, however - LOMAC does not prevent writes to NICs regardless of their level, as data written to a NIC cannot cause a flow between objects on the local host.

Additional Information

For additional information regarding LOMAC, please contact Tim Fraser at 443-259-2350 (tfraser@nai.com) or download the LOMAC distribution from <ftp://ftp.tislabs.com/pub/lomac>. LOMAC is Free Open-Source software; the distribution includes the LOMAC manual and the full source of the experimental LOMAC prototype under the GNU General Public License.

1/5/01



Who's watching your network

For more information on NAI Labs, contact your authorized NAI Sales Representative, or visit <http://www.nailabs.com>.

CORPORATE Headquarters

3965 Freedom Circle
Santa Clara, CA 95054-1203
Tel (800) 764-3337*
Fax (888) 203-9258

*Call for additional Worldwide Sales Offices