549

AT891-0247 RMC

J. S. Langford, D. H. Steves and K. C. Witte

IBM Technical Disclosure Bulletin Vol. 34 No. 8 January 1992

TRUSTED PATH MECHANISM IN AIX

Disclosed is a mechanism for providing a Trusted Path between the system Trusted Computing Base (TCB) and the user in a UNIX*-based operating system. This mechanism allows the user to communicate with the TCB in a manner which is not susceptible to integrity, disclosure or availability attacks. In addition, the AIX** Trusted Path mechanism allows the administrator to tailor the mechanism for each account or terminal line, provides a high degree of security while still offering the user a flexible command environment and requires only simple processing by the terminal drivers.

The Trusted Computing Base is the part of the operating system which is privileged to perform the security functions of the system, including access control, accountability and authentication. For many of these functions, it is considered highly desirable to provide a method to insure both the TCB and the user that their communications are secure. As an example of this, consider the log-in program which authenticates the user and then assigns credentials to the user for the session. Since users must trust this program with their passwords (or other means of authentication), a very common attack on operating systems is to imitate the log-in program. In systems without a Trusted Path, there is no defense against this sort of attack.

There are three components of the Trusted Path subsystem in AIX. These are Secure Attention Key (SAK) processing, the Terminal State Manager (TSM) and the Trusted Shell. Secure Attention Key processing consists of detecting that the Secure Attention Key has been entered by a user and then notifying the Terminal State Manager of this occurrence. The Terminal State Manager will establish a clean environment for the terminal line and then execute the Trusted Shell. The Trusted Shell, in turn, provides the user with a safe environment for executing commands.

Terminal state management is the most important component and is implemented by the TSM program. The TSM program incorporates aspects of the init program and all of the functions of the getty and log-in programs. TSM is spawned by init for each defined terminal line in the system. TSM first does the line conditioning formerly done by getty. It then establishes a secure communications environment by:

1) opening the terminal device and making it accessible only by privileged processes.

2) revoking access to the terminal by all other processes. All processes sleeping in the terminal driver are killed, and any process with the terminal device open will be killed if it accesses the device.
3) It marks the terminal as trusted and registers itself with the terminal driver as the Terminal State Manager for this line.

These steps insure that any program which accesses the terminal after this point must be part of the TCB. At this point, it performs the log-in function by authenticating the user. After the user is logged in, the TSM program will normally drop the trusted state of the terminal before running the user's defined initial program.

When the user enters the Secure Attention Key, the terminal driver will notify the registered Terminal State Manager using the UNIX signaling facilities. When TSM receives this signal (SIGSAK), it then reestablishes the secure communications environment for that line using the three steps outlined above. At this point, however, it does not log the user into the system again, but instead runs the Trusted Shell for the user. When the user no longer requires the secure communication path, the user terminates the Trusted Shell and TSM will then rerun the user's initial program.

Secure Attention Key processing is the responsibility of the terminal line driver for each device which supports the Trusted Path mechanism. The Secure Attention Key in AIX is defined a two-key sequence: Control X Control R. The driver must recognize this sequence and must also delay passing the Control X key to an application since the application could use this key as a warning and then close the terminal prior to the receipt of the Control R character. The Secure Attention Key then would not be recognized and the malicious program can then reopen the terminal and pretend to be the Trusted Shell. After the driver recognizes the SAK it then sends a signal to the terminal state manager.

The Trusted Shell is implemented by the TSH program. This is a Korn Shell which has been modified to enhance its trustworthiness. The function and alias definition features are deleted and command history is not supported either. Since TSH is part of the TCB, it cannot be modified except by privileged components, and these features have that implicit or explicit effect. In addition, environment variables used directly by TSH are fixed and cannot be changed. For example, the IFS variable, which defines the field separator, is set to the space character. This closes major security holes in the system. Lastly, built-in commands have been added to provide the users with a way to exit from the system (logout) or return to their normal environment (shell).

Access revocation is a key basis for the Trusted Path mechanism. Access revocation is done in two steps. First, each entry in the system open file table which refers to the terminal device is marked as invalid. Any process which references one of these entries will be killed. An exception is made for the entry held by the process which is performing the access revocation. Second, the terminal driver itself will kill any process which is sleeping inside the driver waiting for an input or output operation to complete.

A special problem is posed by the fact that each process with a controlling terminal will, in effect, have implicit access rights for that terminal. Each such process will have a pointer to the state block structure for its controlling terminal and can access it by opening the /dev/tty device special file. This mode of access will, therefore, bypass all access checks. To close this back door, an access check was inserted into the /dev/tty driver's open routine, which will deny access to the process if the terminal is marked as being trusted and the process is not privileged.

While the described mechanism provides a Trusted Communications Path mechanism which is adequate for most normal users, there are some exceptional cases which must be dealt with. These include:

- programs which pass binary data over terminal lines, most notably the UUCP utilities. Since the data flowing over the lines is binary, there is a high probability that it will contain the Secure Attention Key sequence, which will trigger the Trusted Path processing and thus interfere with the data transfer on that line.
- users whose privilege is so great that they must be required to always use the Trusted Path.
- accounts whose initial program provides a more restricted environment than the Trusted Shell. An example of this is the system shutdown account maintained on many systems which allows operators to log into a special account to shut down the system. This account will have a high degree of privilege associated with it, yet the users with access to the account are only to use that privilege for its intended purpose. This is accomplished by having the initial program for that account perform the system shutdown. But if users can invoke the Trusted Shell with the Secure Attention Key, they can gain unrestricted access to the privileges of the account.

Because of these cases, the AIX Trusted Path mechanism is configurable, on a per-user account and per-terminal line basis.

To deal with the first case, the administrator may either disable the Trusted Path mechanism on a terminal line, if one line is used to transmit binary data, or may disable SAK processing on a per-account basis, so that accounts like the UUCP account may be used to transfer this data over the same terminal lines accessed by normal users. If the mechanism is disabled on a specific line, the Terminal State Manager never initiates Trusted Path processing for that line. If SAK processing is disabled for an account, TSM will issue a special control operation to the terminal to disable SAK recognition whenever that account is used.

In the second case, it is possible for an administrator to define the user's account such that the user may only run Trusted Programs (programs which are defined to be part of the TCB). For these accounts, the Terminal State Manager will verify that the initial program is part of the TCB and will not drop the trusted state of the terminal. If the program is not part of the TCB, TSM will fail the log-in sequence.

The third case is handled by allowing the administrator to disable the use of the Trusted Shell for an account. For these accounts, if the Secure Attention Key is entered by the user, the Terminal State Manager will log the user off the system rather than running the Trusted Shell.

The AIX Trusted Path mechanism is distinguished by its configurability as well as the simplicity of its implementation in the driver and its use of a "normal" shell for command invocation. Terminal driver support was designed to be simple due to the lack of a clean architecture for terminal drivers in general. Forcing more complicated function into the system terminal drivers would be unwise from both a system perspective and from a security perspective, since the complicated nature of terminal drivers would make it difficult to guarantee the correct implementation of the Trusted Path.

The use of a normal UNIX shell as the Trusted environment is also unique. This allows system administrators to perform their tasks securely and in a mostly normal fashion, since most of the traditional shell functions, such as pipes and redirection, are provided, along with the shell programming language. In other secure systems which provide a Trusted Path, the trusted command environment is restricted to simple command invocation via menus.

* Trademark of UNIX System Laboratories, Inc.
** Trademark of IBM Corp.