Technical Note: Cb Response Linux Sensor Kernel Module Unload Changes

Overview

Cb Response for Linux cbsensor kernel module hooks into the Linux Security Module (LSM) and kernel system calls. It does so by first saving the existing function pointers and then replacing them with its own functions which then call the original functions in a chain. This is a common practice for Linux security products.

When the cbsensor module is unloaded, it restores the original function pointers. If another kernel module replaced the original function pointers with its own, cbsensor will restore the function pointers to the values that it saved off. This will effectively disable the functions provided by modules loaded after cbsensor. If another module has saved off the original system call and/or LSM pointers to be called when they complete their own operations, then they will be calling cbsensor functions. If cbsensor unloads the following may occur (together or separately):

1. The modules loaded after cbsensor can have their functionality disabled
2. A system crash can occur if the modified system call or LSM operation is in progress and the now removed cbsensor function is called

Resolution

To combat this problem a change has been introduced in cbsensor version 6.1.7. Upon a request to unload (via rmmod for example), cbsensor checks to see if the LSM and system calls have changed since cbsensor modified them.

If the function pointers have not been changed since cbsensor loaded, then cbsensor will restore the original function pointers. However, the unload operation will still fail on this first call. This is because cbsensor cannot unload in the middle of execution and unloading itself would result in a crash. Therefore, on the first call to *rmmod* or similar, cbsensor will remove its character device node, restore the function pointers, and return error EBUSY. On a second call to *rmmod,* cbsensor will unload. This requirement to invoke *rmmod* twice has been incorporated into the RPM uninstall routine and should be transparent to most users assuming that cbsensor can unload (the function pointers have not been modified).

If the function pointer values have been changed, then the attempt to unload cbsensor will result in an error, EBUSY and refuse to unload. However, the cbsensor's character device node will be removed to prevent future use. This is done for two reasons

1. To avoid an update scenario in which the userspace cbdaemon is upgraded and cbsensor is not, as they may have incompatible APIs.
2. To provide a clear observable indication that cbsensor failed to unload and user intervention is required.

This leaves Cb Response non-functional until cbsensor is unloaded and reloaded using one of the mechanisms described below.

Special Case Handling

If the cbsensor unload has failed, there are options to return to normal operations.
Option 1: Reboot the system. This is the easiest solution.
Option 2: Remove other modules that have modified sycall or LSM hooks in the reverse order in which they were loaded. Assuming these modules correctly restore hooks as they found them, the hooks will contain the same values as cbsensor originally saved. At that point, cbsensor can be unloaded by executing *rmmod* **twice** for the reasons described above.

Conclusion

There are two important takeaways from this notice.

1. cbsensor detects if system call or LSM hooks have been modified since cbsensor loaded. If these hooks have been modified, then cbsensor refuses to unload to prevent a kernel crash and Cb Response is not operational until the situation is resolved.
2. cbsensor now requires that *rmmod* or similar calls that unload the cbsensor module be invoked twice to fully unload the module. The first call checks and restores system call and LSM hooks if it is safe to do so, and returns error EBUSY while it restores these hooks. The second call succeeds if system call and LSM hooks have not been modified since cbsensor was first loaded.