GIDISS Trusted Linux
Implementación de la ventana confiable

Duilio Javier Protti
dprotti@fceia.unr.edu.ar

Grupo de Investigación y Desarrollo en Ingeniería de Software
Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Universidad Nacional de Rosario, Argentina

3 de agosto de 2004

Introducción

El presente informe acompaña la modificación del kernel de Linux versión 2.4.14, realizada como trabajo práctico para la materia "Métodos Formales de Seguridad Informática". La modificación pretende implementar un camino confiable entre el kernel y el usuario del sistema, en forma de una ventana confiable mostrada en la última línea de la pantalla de una terminal de texto. Se analizan varias alternativas para la implementación, comentando ventajas y desventajas de cada una.

Cómo el kernel muestra la imagen en el monitor

La imagen que se muestra en el monitor se refresca aproximadamente 60 veces por segundo desde una imagen guardada en la memoria de una terminal. En la mayoría de las PC, donde la terminal de consola se emula utilizando varios dispositivos físicos como la placa de video, el puerto PCI o AGP, el teclado y otros, esa imagen se guarda en la memoria de la placa de video. Para una terminal de texto el almacenamiento de ésta imagen utiliza mucha menos memoria que para una terminal gráfica, ya que en lugar de guardarse cada píxel de la pantalla se utiliza un mecanismo más eficiente en espacio.

Cuando se escribe algo en la consola, esto primero pasa por la etapa de procesamiento tty, y luego la salida de ésta etapa se envía al driver[1] de consola. En linux, el driver de consola emula una terminal Linux, muy parecida a una terminal VT100. Éste driver procesa su entrada buscando secuencias de escape VT100. Los caracteres que no son parte de una secuencia de escape, son convertidos en Unicode, usando alguna

[1] Controlador de dispositivo.

de las cuatro tablas disponibles en el kernel si la consola no se encontraba ya en modo UTF-8. Luego se busca en una tabla que describe la correspondencia entre caracteres Unicode y posiciones de fuente. Éstas posiciones son índices de 8 o 9 bits que son escritos en la memoria de video causando que se dibujen los caracteres correspondientes guardados en la memoria ROM de la placa de video. Se pueden cargar fuentes propias en la ROM de la placa de video utilizando *setfont*, mapas de caracteres Unicode usando *loadunimap* y también la tabla de correspondencia Unicode-posición de fuente con *mapscrn*.

Scrolling

Existen dos maneras de hacer scroll en una consola: la primera, llamada *hard scrolling*, consiste en dejar el texto en la memoria de la terminal tal cual está, pero cambiando el origen visible de la pantalla. El origen visible de la pantalla es la posición de memoria a partir de la cual se comienza a leer la imagen que deberá refrescarse en el monitor (esto es un registro en hardware, lo que significa que éste método solo estará disponible en máquinas que posean dicho registro). Éste método es muy rápido. La segunda forma de hacer scroll es llamada *soft scrolling*, y consiste en mover todo el texto de la pantalla hacia arriba o hacia abajo en la memoria de video. Ésto es mucho mas lento que el primer método.

Consolas Virtuales

Al cambiar de consola virtual, el contenido de la pantalla de la anterior consola virtual es copiado en la memoria del kernel, y el contenido de la nueva consola virtual (guardado en el kernel), es copiado en la memoria de video. Solo la parte visible de la pantalla es copiada, no toda la memoria de video, por lo que cambiar de consola virtual implica perder la información de scroll.

Requerimientos de la ventana confiable

La ventana confiable debe actuar como un camino confiable desde el kernel y los programas confiables hasta el usuario. Para esto se eligió, en el diseño de GTL[2], tomar un área del monitor y destinarla para escritura exclusiva del kernel y los programas confiables, y hacer imposible para los programas de usuario no confiables escribir en ésa área, incluso creando nuevos dispositivos, abriendo nuevas consolas, cambiando entre consolas virtuales, cambiando disciplinas de línea, remapeando la tabla de fuentes o cambiando la resolución del driver de video. Para lograr estos objetivos, en la actual implementación solo se admiten consolas VGA en modo texto, ningñ otro tipo de consola es permitido.

Alternativas analizadas para implementar la ventana confiable

En una etapa temprana del desarrollo se analizó la posibilidad de implementar la ventana confiable modificando la disciplina de línea de la consola de sistema, que es una

[2]GIDISS Trusted Linux.

consola Linux, una emulación casi exacta de una terminal VT100, y luego por supuesto admitir solamente este tipo de disciplina de línea para una consola, y ninguna otra más. Los dos problemas que plantea esta alternativa son:

- La modificación es a muy alto nivel, con lo cual la cantidad de código a modificar es muy grande, y se debe examinar mucho código para asegurar que no se deja un camino abierto a bajo nivel para llegar al driver de video y de esa manera posiblemente acceder al contenido que se muestra en el monitor.
- Se restringe al sistema a utilizar una ñica disciplina de línea (se pensaba emular una terminal VT100-s-bot, que es una terminal con una línea de estado en la parte inferior de la pantalla), lo que claramente atenta contra la usabilidad del sistema.

También se contempló la posibilidad de implementar la ventana confiable utilizando **framebuffer**, y nuevamente, admitir solo consolas que utilicen éste driver. La desventaja de esta alternativa es que la implementación en éste caso es muy sensible a cambios de hardware, pues si bien framebuffer presenta una interfaz comñ para acceder al hardware de video, ésto se logra implementando dicha interfaz para *cada* placa de video soportada, con lo cual para hacer utilizable el sistema se deberían implementar posiblemente varios drivers. Éste costo extra se justificaría si pretendiésemos utilizar la ventana confiable en modo gráfico, pero el requerimiento está planteado solamente para una consola de texto.

La alternativa elegida: modificación del driver VGA

Finalmente se decidió modificar el driver VGA, pues esto presenta la ventaja de hacerse a bajo nivel, con lo cual el total de código a modificar es mínimo y además las capas superiores de software no resultan afectadas por el cambio. Por ejemplo, se puede seguir utilizando consolas virtuales, así como cualquier disciplina de línea disponible, cualquier conjunto de caracteres disponible, cualquier resolución admitida por el hardware en modo texto, fuentes intercambiables, hard y soft scrolling, etc. La desventaja encontrada es que dado que la modificación fue a muy bajo nivel, no fue posible permitir la escritura en la ventana confiable desde espacio de usuario mediante los mecanismos habituales, sino que se debió agregar una llamada al sistema para tal fin (`trusted_syslog()`).

Conclusión

La modificación propuesta se incluye en la actual versión de desarrollo de GTL, que puede accederse por CVS anónimo en cvs.sourceforge.net:/cvsroot/gtl, nombre de módulo “gtl2004dev”. También puede obtenerse el parche para ser aplicado a un kernel 2.4.14 original en la sección de parches del projecto en SourceForge en http://sourceforge.net/projects/gtl.

La ventana confiable implementada como última línea de la pantalla, actualmente tiene el inconveniente de no admitir scrolling, por lo cual los mensajes mostrados no deberían tener una longitud superior al ancho de la pantalla (en caracteres)

para evitar así que el usuario no pudiera ver parte de algunos mensajes.

Añ así, la alternativa elegida para la implementación permitió que el total de código a modificar sea mínimo, así como también permitió que el impacto sobre la usabilidad del sistema sea casi imperceptible. Para que ésto último sea totalmente cierto, trabajos futuros deberán implementar la ventana confiable en modo gráfico, para permitir el uso del sistema con entornos gráficos, que son los de uso más comñ en la mayoría de ámbitos en la actualidad.

Referencias

[1] Maximiliano Cristiá: "System and Security model of GIDISS Trusted Linux 0.1.1 - Z version". GIDISS, http://www.fceia.unr.edu.ar/gidis, 2003.

[2] David S. Lawyer: *"Text Terminal HOWTO"*, http://linuxdoc.org/HOWTO/Text-Terminal-HOWTO.html, 2001.

[3] Andries Brouwer: *"Keyboard and console HOWTO"*, http://www.tldp.org/HOWTO/Keyboard-and-Console-HOWTO.html, 1998.