Linux, controladores y programación del núcleo

Curso práctico de 4 días - 28h
Ref.: LDI - Precio 2025: 1 710€ sin IVA

Este curso le permitirá dominar el desarrollo de controladores de periféricos robustos adaptados a las distintas distribuciones de Linux. Conocerás los distintos tipos de periféricos, la gestión de memoria, la implementación de protocolos de red y los periféricos USB.

OBJETIVOS PEDAGÓGICOS

Al término de la formación, el alumno podrá:

Dominar el desarrollo de controladores periféricos

Una comprensión detallada del funcionamiento interno del núcleo

Saber desarrollar e integrar nuevos elementos en el núcleo de Linux

Escritura de un controlador periférico en modo carácter o bloque

Los numerosos ejercicios y casos prácticos progresivos se llevarán a cabo en una red de servidores Linux. Todos los programas desarrollados durante las sesiones prácticas están disponibles en forma de esqueletos que los participantes pueden completar por sí mismos.

PROGRAMA

última actualización: 05/2024

1) Presentación del núcleo

- Visión general del sistema y función del núcleo.
- Sitios de referencia.
- Características específicas de los núcleos 3.x y 4.x.
- Ciclos de desarrollo del núcleo, parches.
- Modo de funcionamiento (supervisor y usuario). Llamadas al sistema.
- Organización de fuentes (Include/linux, Arch, Kernel, Documentación, etc.).
- Principio de compilación del núcleo y los módulos.
- Dependencias y símbolos.
- Exportación de símbolos.
- Carga del núcleo (soporte, argumentos, etc.).

Trabajo práctico : Compilación e instalación de un kernel 3.x.

2) Herramientas que puede utilizar

- Herramientas de desarrollo (Gcc, Kbuild, Kconfig y Makefile, etc.).
- Herramientas de depuración (GDB, KGDB, ftrace, etc.).
- Entorno de depuración (Linux Trace Toolkit, etc.).
- Herramienta de gestión de versiones (Git, etc.).
- Rastrea las llamadas al sistema (ptrace...).

Trabajo práctico : Instalar todas las herramientas y fuentes para generar un módulo para el kernel. Configurar el sistema para que cargue los módulos automáticamente al arrancar. Escribir y probar módulos sencillos.

3) Gestión y programación de hilos

- Los diferentes tipos de periféricos.
- Contextos de funcionamiento del núcleo. Protección de variables globales.

PARTICIPANTES

Desarrolladores Linux/Unix.

REQUISITOS PREVIOS

Buenos conocimientos de Linux/Unix y programación en C.

COMPETENCIAS DEL FORMADOR

Los expertos que imparten la formación son especialistas en las materias tratadas. Han sido validados por nuestros equipos pedagógicos, tanto en el plano de los conocimientos profesionales como en el de la pedagogía, para cada curso que imparten. Cuentan al menos con entre cinco y diez años de experiencia en su área y ocupan o han ocupado puestos de responsabilidad en empresas.

MODALIDADES DE EVALUACIÓN

El formador evalúa los progresos pedagógicos del participante a lo largo de toda la formación mediante preguntas de opción múltiple, escenificaciones de situaciones, trabajos prácticos, etc.
El participante también completará una prueba de posicionamiento previo y posterior para validar las competencias adquiridas.

MEDIOS PEDAGÓGICOS Y TÉCNICOS

• Los medios pedagógicos y los métodos de enseñanza utilizados son principalmente: ayudas audiovisuales, documentación y soporte de cursos, ejercicios prácticos de aplicación y ejercicios corregidos para los cursillos prácticos, estudios de casos o presentación de casos reales para los seminarios de formación.
• Al final de cada cursillo o seminario, ORSYS facilita a los participantes un cuestionario de evaluación del curso que analizarán luego nuestros equipos pedagógicos.
• Al final de la formación se entrega una hoja de presencia por cada media jornada de presencia, así como un certificado de fin de formación si el alumno ha asistido a la totalidad de la sesión.

MODALIDADES Y PLAZOS DE ACCESO

La inscripción debe estar finalizada 24 horas antes del inicio de la formación.

ACCESIBILIDAD DE LAS PERSONAS CON DISCAPACIDAD

¿Tiene alguna necesidad específica de accesibilidad? Póngase en contacto con la Sra. FOSSE, interlocutora sobre discapacidad, en la siguiente dirección psh-accueil@orsys.fr para estudiar de la mejor forma posible su solicitud y su viabilidad.

- Representación de hilos (estado, estructura task_stru, thread_info, etc.).
- Hilos, contexto de ejecución.
- El planificador de Linux y la preferencia.
- Crear un hilo del núcleo (kthread_create, wakeup_process...).

Trabajo práctico : Crear un módulo que cree un hilo del kernel durante la inserción y lo descargue durante rmmod. Escribir un módulo de registro de tiempo de eventos de alta precisión. Escribir un módulo que proporcione información sobre las estructuras internas de los procesos.

4) Gestión de memoria, tiempo y proc

- Organización de la memoria para arquitecturas UMA y NUMA.
- Espacio de direcciones del usuario y del núcleo.
- Busca personas a la carta.
- Asignaciones de memoria, buddy allocator, kmalloc, slabs y memory pools.
- Gestión del acceso a la memoria (cachés y MMU).
- Problemas relacionados con la sobrerreserva de memoria.
- Gestión de memoria en x86 y ARM, uso de Hugepages.
- Optimización de las llamadas al sistema (IAPX32, VDSO).
- Sincronización y espera en el núcleo, colas de espera, mutexes y finalizaciones.
- Ticks y jiffies en Linux.
- Reloj en tiempo real (RTC), implementación de temporizadores.
- Interfaz de temporizador de alta resolución, sellos.
- Herramientas específicas del núcleo, listas enlazadas, kfifo y container_of.
- La interfaz del núcleo con /proc a través de procfs.

Trabajo práctico : Uso de temporizadores y marcas de tiempo. Implementación del acceso procfs. Implementación de la asignación de memoria en el kernel y optimización mediante slabs.

5) Dispositivo en modo carácter

- Escritura de controladores de dispositivos de caracteres.
- VFS (Sistema de archivos virtual).
- Métodos asociados a los periféricos de caracteres.
- Gestión de las interrupciones DMA y acceso al hardware.
- Registro y optimización de controladores de dispositivos de caracteres.

Trabajo práctico : Escritura progresiva de un controlador periférico en modo carácter. Implementación de la sincronización de E/S entre hilos y con la rutina de interrupción. Implementación de la asignación de memoria.

6) Marco de controladores de Linux - sysfs

- Presentación del marco, kobject, kset y kref.
- Controlador, controlador de dispositivo, bus y objetos de clase.
- Uso y generación de atributos presentados en sysfs.
- Interfaz con métodos de conexión en caliente, emparejamiento, sondeo y liberación.
- Gestión del firmware.
- Gestión de la energía, métodos de gestión de la energía.

Trabajo práctico : Implementación de un bus, un controlador y un controlador de dispositivo. Adaptación del controlador de dispositivo de caracteres. Ejemplo de uso de la interfaz.

7) Periféricos y sistemas de archivos en modo bloque

- Principio de los periféricos en modo bloque. Registro de controladores.
- Llamada de retorno de lectura y escritura. Soporte para formateo y operaciones avanzadas.
- Programador de E/S en bloque para el núcleo.
- Diseño de sistemas de archivos.
- Registro de un nuevo sistema de archivos.

Trabajo práctico : Ejemplo de controlador de dispositivo virtual completo. Ejemplo de un sistema de archivos personalizado.

8) Interfaces y protocolos de red
- Gestión de interfaces de red en Linux.
- Uso de skbuff.
- Ganchos Netfilter.
- Integración de un protocolo.
Trabajo práctico : Ejemplo de controlador de red para un dispositivo virtual. Implementación del protocolo de red.

9) Controladores para periféricos USB
- Principio de los periféricos USB. Interfaz con el módulo USB-core.
- Interacción del dispositivo con el núcleo Linux.
- Construcción de un URB (bloque de solicitud USB).
- Aparatos USB.
Trabajo práctico : Registro de un controlador USB. Escritura de un controlador en modo isócrono.

FECHAS

Contacto