udp Escuela de Informática y Telecomunicaciones
FACULTAD DE INGENIERÍA

Introducción a Minix

Proyecto 3

Sistemas Operativos

Resumen

Este proyecto es la introducción básica a Minix. Deben de descargar e instalar Minix en un emulador (`Bochs`, `Qemu`, u otro de su elección), recompilar el kernel, y hacer un cambio simple para agregar capacidades de emulación a Minix.

1. Instalación de Minix

Deben instalar la última versión de Minix (3.3.0). Descarguen la imagen ISO de la página del proyecto. Deben de decidir en donde desean instalar Minix, ya sea en hardware real o en una máquina virtual, como `Qemu`, `Virtual Box`, `Bochs`, o `VMWare`. Noten que mientras una instalación nativa puede ejecutarse más rápido, una máquina virtual tiene la ventaja de que pueden reiniciar, pausar, o apagar la máquina fácilmente. Además, no se tienen que preocupar que la inestabilidad en su código comprometa la máquina donde están trabajando.

Dada la fragilidad del código, se les recomienda el utilizar un manejador de versiones para el código que estén alterando. No sería bueno que el trabajo de toda una noche se pierda al momento de probar la máquina virtual que quedo inaccesible, y su código está detrás de un disco duro que no se puede recuperar, o que les puede tomar más tiempo que el del proyecto para terminar.

Existe documentación en linea sobre la instalación de Minix sobre hardware, por ejemplo vean este enlace. Para la instalación en máquinas virtuales pueden guiarse a través de éstas instrucciones en linea. En general, la instalación involucra los siguientes pasos (independientemente del medio que elijan):

- particionar el disco duro,
- *bootear* Minix desde la imagen que descargaron,
- ejecutar el script de inicio, y
- reiniciar la máquina desde la instalación de Minix.

1.1. Configurando Minix

Una vez instalado Minix, hay algunas cosas que deben hacer para mejorar la usabilidad del sistema. La primera es hacer que la red funcione. Noten que la manera de establecer la red dependerá de la plataforma que estén utilizando. Para `Qemu`, deben de utilizar el *user mode stack*. Éste no requiere de privilegios de *root* y debería de funcionar al momento de instalar. Lo que necesita corregirse es la resolución de nombres (DNS). De manera similar a Linux, Minix utiliza el archivo `/etc/resolv.conf` para encontrar los servidores DNS. Revisen la página del manual `man` para determinar el formato de `/etc/resolv.conf` y provean una dirección IP válida para el servidor. Cuando todo esté configurado correctamente, deberán de poder escribir el host `google.com` y obtener la IP correspondiente.

Una vez la red esté activa, pueden instalar otras aplicaciones que les faciliten el uso del sistema. Por ejemplo, un editor de texto en consola (como `vim`), `openssh` (para intercambiar archivos entre Minix y su sistema), o una nueva shell (por ejemplo, `bash`). Noten que algunas de estas opciones pueden estar ya instaladas con la imagen que descargaron. Adicionalmente, deben de revisar el código de Minix que está en `/usr/src` y deben familiarizarse con los elementos que se encuentran en los distintos directorios.

1.2. Ambiente de desarrollo

Ahora que ya revisaron Minix, deben de preparar el ambiente de desarrollo en él. Noten que el trabajar directamente en Minix puede ser muy difícil, de tal manera que necesitan una manera conveniente de mover archivos entre su máquina y la instalación de Minix. Para esto pueden utilizar `rsync` para sincronizar la carpeta de Minix y la carpeta de desarrollo. Para poder ejecutar `rsync`, deben de instalar `openssh` en Minix —ver Sección 1.1.

Después de instalar los paquetes deben de reiniciar Minix para que el demonio `ssh` se ejecute, o bien pueden iniciarlo manualmente. Adicionalmente, deben de configurar la conexión a través del puerto 22 (puerto por defecto) para poder conectarse a su máquina virtual. Por ejemplo, en `Qemu` utilizando el modo usuario de redes, pueden agregar el parámetro `-redir tcp:2222::22` a la linea de comando. Este argumento permite que puedan accesar al puerto 22 de Minix a través del puerto 2222 de su máquina anfitrión. Se recomienda que copien su llave pública de `ssh` hacia el archivo `.ssh/authorized_keys`, y que agreguen una entrada en el archivo `.ssh/config` de la máquina de desarrollo (no Minix) que les permita hacer una conexión directa a través del comando `ssh minix`.

Además, si están utilizando un versionador que utilice autenticación `ssh`, como `git`, pueden hacer cambios en o actualizar directamente el repositorio. Pueden utilizar el siguiente código como referencia inicial para el archivo `.ssh/config` (suponiendo que utilicen `quemu`)

```
Host minix
    Hostname localhost
    Port 2222
    User root
```

(Vean la siguiente guía para la configuración de llaves `ssh`.)

Ahora que ya pueden entrar a su máquina a través de `ssh` a través de `ssh minix`, deben crear una copia (limpia) del código fuente. Esto es necesario por dos razones. Primero, cuando hagan cambios que no funcionan y necesiten una referencia del código original pueden verlo inmediatamente. Segundo, necesitan una referencia para poder crear parches (es lo que usaremos para la entrega de código). Para hacer esto deben crear una copia en Minix

```
cp -r /usr/src /usr/src_clean
```

Ahora, pueden subir una copia del código que están desarrollando. A este punto no deberían modificar ningún código directamente en Minix (si así lo desean). Y pueden fácilmente subir el código a través de

```
rsync -rptOv minix:/usr/src/ minix_src_clean
```

En caso de que `rsync` falle, pueden continuar la sincronización hasta que no reciban más errores. Noten que está no es la única manera de mantener el código y copiarlo a su máquina virtual. Se les aconseja (nuevamente) utilizar un versionador y mantener el código entre las dos máquinas a través de esté, por ejemplo vean la guía de `git` en Minix.

1.3. Compilando y modificando el kernel

Ahora es momento de recompilar el kernel. Antes de hacer cualquier cambio, deben de asegurarse que pueden construir e instalar exitosamente el nuevo kernel. Cámbiense al directorio `/usr/src/tools` y ejecuten `make`. Revisen las opciones para ver como se puede instalar y reconstruir el kernel. Una vez instalado el kernel, y después de reiniciar la máquina, deben seleccionar la opción del nuevo kernel personalizado. Sino, ejecutará el kernel anterior.

Se recomienda el estar en el kernel estable cuando estén transfiriendo archivos de y hacia Minix, y aún más **importante cuando estén compilando** (recuerden, no son Chuck Norris). Para regresar al kernel estable pueden ejecutar el siguiente código (y recuerden escoger la opción correcta)

```
shutdown
boot d0p0
```

Ahora que pueden construir y cargar un nuevo kernel, es momento de hacer un cambio al sistema operativo. Primero, deben hacer una copia de la carpeta del código (o una nueva rama en su versionador)

```
rsync -rptOv minix_src_clean proj
```

Para las modificaciones del kernel, deben de imprimir el nombre de cada archivo que es ejecutado por el sistema operativo. Deben de localizar los archivos fuente del kernel que implementan la llamada al sistema `exec`. Después, deben de insertar una instrucción `printf` que emita `ejecutando: archivo`, donde `archivo` es el nombre del archivo que se ejecuta. Es decir, cuando se escribe `ls` en la terminal, debe de leerse el string `ejecutando: archivo-ls`, donde `archivo-ls` es el archivo que contiene el código fuente del comando `ls`. Noten que esta modificación requiere una simple linea de código, pero les proveerá con las facilidades necesarias para entender lo que sucede en el sistema operativo y familiarizarse con él. Posteriormente, deben de sincronizar sus cambios con Minix

```
rsync -rptOv proj/ minix:/usr/src/
```

El argumento de la fuente y el destino son importantes. Por lo general, no deberán de sincronizar de Minix a su ambiente de desarrollo, así que no deberían de tener problemas confundiendo los comandos. Sin embargo, el escribir scripts que automaticen este proceso es recomendable. (Noten que si usan un versionador solo es necesario el actualizar los repositorios.)

Reconstruyan el kernel, instálenlo, y reinicien el sistema. Si tuvieron éxito deberían de ver los cambios antes mencionados en los comandos.

1.4. Creando un parche

La última tarea es entregar los cambios realizados. Claramente no queremos todo el código fuente de Minix en cada entrega. Entonces, deben entregar un conjunto de parches de código para los archivos que modificaron. Un parche captura los cambios entre dos versiones de los archivos. Dado un archivo antiguo y el parche, uno puede utilizar un programa para crear el nuevo archivo.

Entonces, cuando tengan sus cambios listos para la entrega (incluyendo la recompilación, instalación, y evaluación), ejecuten

```
diff -ruNp minix_src_clean/ proj/ > patch
```

Esto creará un archivo parche, que deberá contener solamente las lineas que cambiaron intencionalmente. Para este proyecto el parche no deberá ser muy grande (más de unas 15 lineas). Deben evaluar que el parche que crearon funcione, creando otra copia del código fuente y ejecutándolo

```
cp -r minix_src_clean test
cd test
patch -p1 < ../patch
```

Esta será la forma en que se evaluarán las entregas, **y no deberán de producir ningún error o aviso de alerta**.

2. Instrucciones

2.1. Trabajo a realizar

1. Este proyecto se desarrollará en parejas.
2. Lean detalladamente las instrucciones a realizar (pueden que necesiten más de una pasada para entender y hacer funcionar todo).
3. Necesitarán más información que los enlaces provistos en este enunciado. Busquen información mientras resuelven los problemas. Y más importante, documéntenla para explicarla en su informe.
4. Deben de seguir la guía presentada en la Sección 1.
5. Deben de generar un parche de los cambios realizados, y evaluar su buen funcionamiento.

2.2. Entregables

Deben de entregar una carpeta con la siguiente estructura

- `codigo`: carpeta que contiene todos los parches que definen su proyecto.
- `informe`: el código fuente para generar el informe que presenta.

Estas carpetas deberán de ser comprimidas en un solo archivo, y subido al sitio del curso junto con un `PDF` del informe. El archivo comprimido deberá ser nombrado utilizando los RUTs de los integrantes siguiendo el patrón `rut-rut`, donde `rut` es el RUT de los integrantes sin puntos ni guión. Por ejemplo, si los integrantes tienen RUT `1234-5` y `6789-0` el archivo comprimido deberá llamarse `12345-67890`.

2.3. Fecha de entrega

La fecha de entrega es el día 26 de mayo. En horario de clase deberán de entregar el informe impreso. Y deberán subir los archivos al sitio del curso antes de la entrega impresa. Noten que la entrega digital de los documentos cierra al inicio de la hora de la clase, por lo que deberán subir sus archivos antes de la clase.

Se les recomienda el subir los archivos antes de la hora límite para evitar problemas con el servidor. No se considerarán correos que se envíen horas antes de la fecha límite de la entrega con problemas subiendo los documentos.

Se les recuerda nuevamente que el curso tiene una política de *tolerancia cero* para las entregas tardías.

2.4. Sobre el informe

- Deben preparar un informe de **máximo 4 páginas** de contenido (éstas no incluyen las figuras o referencias) utilizando `IEEEtran.cls` y la guía entregada en clase.
- El informe debe de contener al menos:
  - Resumen
  - Introducción (explicando Minix, su tecnología, e ideología detrás de la distribución)
  - Deben desarrollar los siguiente tópicos
    - Explicación sobre la instalación, incluyendo la plataforma de instalación: si fue en hardware o en una máquina virtual (¿cuál? ¿qué versión?). Explicar el tamaño de la máquina (disco) y la memoria asignada.
    - Explicar por qué la distribución contiene un usuario llamado `ast`.
    - Expliquen la herramienta que utilizaron para instalar el software en Minix.
    - Expliquen los comandos (pasos) para reconstruir el kernel e instalarlo.
    - Expliquen como cambiar el banner del kernel de Minix (es decir "`Minix 3.3.0 Copyright...`") que se muestra cuando Minix arranca.
    - Explique cual es el nombre de la llamada de sistema número 33 (decimal) de Minix, y explique como determinaron esa información.
  - Conclusión
- **Si se detecta plagio ustedes obtendrán un uno en el proyecto**, y se dará aviso a las autoridades correspondientes para que se tomen las sanciones del caso.

2.5. Notas de interés y restricciones

- Su informe debe de estar realizado siguiendo la guía de documentos técnicos discutida en clase. Recuerde que se penalizará cada falta de ortografía y de redacción con 1 % de la nota final hasta un 20 %.

- No se aceptarán entregas fuera de la hora de entrega. Esto incluye copia digital fuera del plazo, o copia impresa fuera del horario de clase. Además, no se considerará una entrega completa sino existe la entrega digital y la física.
- Para evitar problemas al subir sus archivos (por ejemplo, problemas con el tamaño de la entrega, o disponibilidad del servidor) se les recomienda el subir los archivos antes de la hora de entrega. Noten que la hora de entrega no es la hora para empezar a subir los archivos, sino un límite para tener un punto de corte. Programen su entrega y háganla antes de la fecha límite.
- En caso se sobrepase el límite de páginas estipulado, se evaluará lo presentado hasta el límite solicitado. Para evitar problemas, sigan las instrucciones y sean concisos.
- El parche que entreguen debe de contener las lineas que sean necesarias para el desarrollo del proyecto. No generen cambios innecesarios que entorpezcan la revisión del mismo.
- Dada la complejidad del código que puede generar en este proyecto, se recomienda el entregar un archivo `README` comprensivo, que contenga información necesaria para la compilación de su código. Por ejemplo, dependencias que deben existir en la computadora que compila, versione de las distintas herramientas (gcc, ld, nasm, etc.), e instrucciones detalladas para la compilación y ejecución de su aplicación. Si construyeron scripts para la instalación deben de agregarlos en la carpeta `codigo`.
- Se recomienda que utilicen un manejador de versiones para mantener su código (por ejemplo, git).
- Usted debe de hacerse responsable de que su código compile, enlace (link), y se ejecute correctamente en cualquier ambiente (se calificará en un ambiente Unix con los emuladores `bochs` o `qemu`).

3. Tips

- Este proyecto puede parecer mucho más sencillo que los proyectos anteriores. Sin embargo, no se engañen por que lo que deben de hacer es la punta del iceberg. Deben de familiarizarse con Minix y con sus archivos y funcionalidades. El objetivo del proyecto es la familiarización del estudiante con Minix, y si no aprovechan este tiempo para conocerlo a profundidad, tendrán problemas en el siguiente proyecto.

Tabla 1: Ponderación del proyecto.

	Informe	**Práctica**	**Subtotal**
Resumen	2		2
Introducción	5		5
Instalación	10	10	20
Usuario `ast`	10		10
Herramienta instalación	10	5	15
Reconstrucción kernel	10		10
Banner	10		15
Llamada 33	10		5
Conclusiones	3		3
Parche		15	15
Total	70	30	100

- Revisen la documentación oficial de Minix para poder resolver problemas —que por lo general serán comunes, y les han pasado a otras personas antes que ustedes.
- Establezcan metas iniciales y etapas para avanzar en el proyecto. Por ejemplo, instalar Minix, luego poder comunicarse e instalar programas en él, y luego poder modificarlo. Esta tarea puede demorar más de lo que creen. Además, por su experiencia anterior en los otros proyectos, **no empiecen el proyecto días antes de la entrega**. Planifiquen su entrega y realicen avances con tiempo.

4. Ponderación

El proyecto está ponderado 30 % de práctica y 70 % del informe (esto incluye escritura, estructuración de ideas, redacción, etc.) que será evaluado a través del informe escrito, y del código entregado (parche). La evaluación será general y se evaluarán los aspectos teóricos, prácticos, y la presentación (a través del informe) en conjunto. El detalle de la ponderación está en la Tabla 1.