Universidad Nacional Autónoma de México

Facultad de Ingeniería

Sistemas Operativos

Semestre 2020 - 2

Reporte de Exposición

Windows Subsystem Linux

WSL

ALUMNOS:

Murrieta Villegas Alfonso
Valdespino Mendieta Joaquín

PROFESOR:

Ing. Gunnar Eyal Wolf Iszaevich

GRUPO TEORÍA: 5

1. Objetivos

1) Exponer un panorama general al alumno sobre una alternativa para conocer y empezar a manejar alguna distribución de Linux

2) Explorar un poco acerca de cómo se trabaja a fondo un subsistema en Windows

2. Introducción

Windows Subsystem for Linux o conocido como WSL es una característica incluida en la última versión de Windows que nos permite a los usuarios de Windows 10 usar una terminal de la distribución de linux que queramos (Y que esté disponible),esta arquitectura nos permite tener los directorios típicos de linux como son Home o Bin en Windows y los directorios de Windows en linux. Como bien menciona Microsoft:

"los objetivos principales son aumentar el rendimiento del sistema de archivos, así como agregar compatibilidad total con las llamadas del sistema".

Esta arquitectura quiere implementar y agregar una gran versatilidad a Windows sobre todo al momento de desarrollar programas donde es necesario utilizar un ambiente Linux.

Por otro lado, el conocer WSL es una forma sobre todo para nosotros de hacer conocer a nuestros compañeros una alternativa interesante y llamativa para conocer y aventurarse al mundo Linux, a pesar de que existen alternativas como máquinas virtuales o particiones duales, WSL da la ventaja de sin hacer realmente mucho tener una opción bastante cercana a lo que sería un ambiente Linux, obvio con sus considerables limitaciones.

3. Qué conforma a WSL? y cómo trabaja

WSL es una colección de componentes que permite que los binarios nativos de Linux ELF64 se ejecuten en Windows. Contiene componentes de modo de usuario y de modo de núcleo. Se compone principalmente de:

1 Servicio de administrador de sesión en modo de usuario que maneja el ciclo de vida de la instancia de Linux

2 Controladores de proveedores Pico (lxss.sys, lxcore.sys) que emulan un kernel de Linux traduciendo llamadas de sistema de Linux

3 Pico Procesos que alojan el modo de usuario no modificado Linux (por ejemplo, / bin / bash)

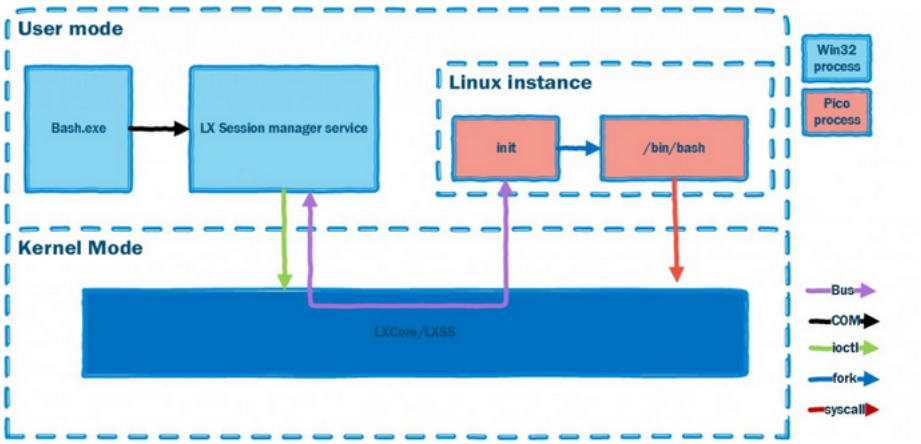

Figura 1: Imagen que describe la arquitectura general empleada en WSL 2

Es el espacio entre los binarios de Linux en modo de usuario y los componentes del kernel de Windows donde .ºcurre la magia". Al colocar los archivos binarios de Linux no modificados en los procesos de Pico, se permite

que las llamadas del sistema Linux se dirijan al núcleo de Windows. Los controladores lxss.sys y lxcore.sys traducen las llamadas del sistema Linux a las API de NT y emulan el kernel de Linux.

3.1. Pico Procesos

El objetivo de de los pico procesos es implementar una forma liviana de ejecutar un programa en un entorno aislado, con las dependencias del sistema operativo de la aplicación desacopladas del sistema operativo, en Windows* el proyecto encargado de esto se le denómino como DrawBridge.

Para comprender, la forma en que se realizan este tipo de procesos dentro de WSL, es necesario mencionar que, existen de forma general 2 tipos de procesos en esta capa de abstracción:

Proceso mínimo: Este es el tipo de proceso más rudimentario o general, específicamente un proceso marcado como un proceso mínimo le dice al resto del host que se salga del camino y no lo administre. Desde el punto de vista del kernel de Windows, es simplemente un espacio de direcciones vacío en modo usuario.

Proceso Pico: Es un proceso mínimo con un controlador de modo kernel del proveedor pico asociado para administrar ese espacio de direcciones vacío en modo usuario.

NOTA 1: El soporte inicial para procesos pico apareció por primera vez en Windows 8.1 y Windows Server 2012R2 pero se limitó sólo Drawbridge.

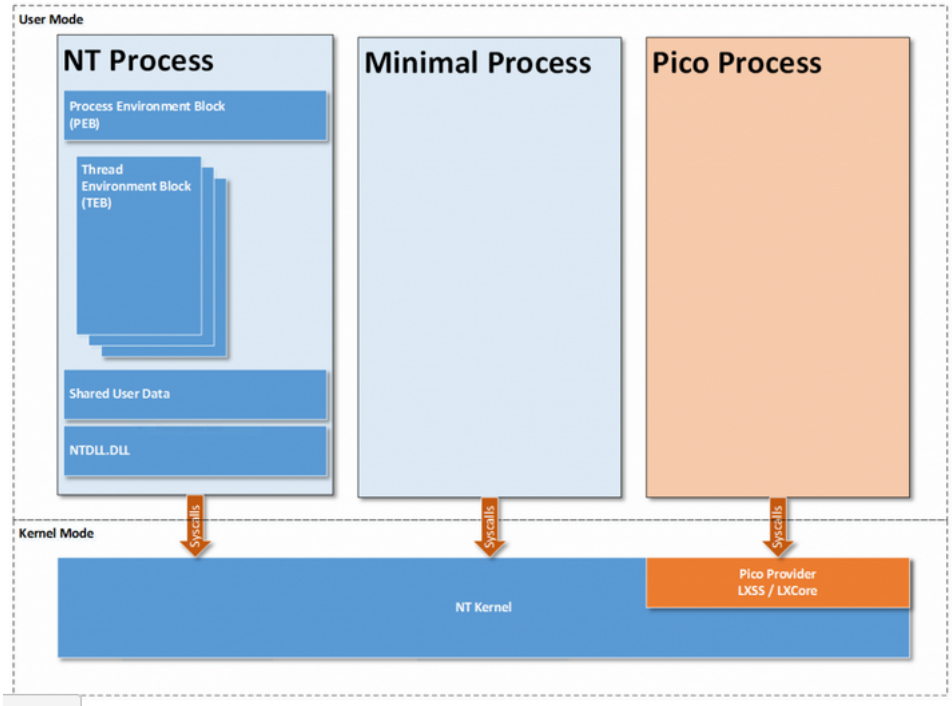

Figura 2: Tipos de procesos empleados en WSL

Por lo tanto, un proceso pico es simplemente un proceso mínimo asociado con un controlador de modo kernel del proveedor pico. Este proveedor pico muestra toda la interfaz del núcleo respecto a la parte del proceso en modo de usuario. El kernel de Windows pasa todas las llamadas del sistema y las excepciones que se originan en la parte del modo de usuario de un proceso pico al proveedor pico para que lo maneje como "mejor le parezca". Esto permite que el proveedor pico modele un contrato de **usuario / kernel** diferente y por separado de lo que normalmente proporcionaría Windows*.

NOTA 2: Independientemente de los comportamientos y las abstracciones que el proveedor pico expone al modo de usuario, la última decisión dependerá del núcleo de Windows. Cabe destacar que Microsoft menciona que partes del kernel de Windows tuvieron que actualizarse para admitir nuevos escenarios del proveedor pico.

A diferencia de los procesos NT tradicionales, cuando se crea un proceso mínimo, el espacio de direcciones en modo de usuario no se toca y no se crean subprocesos para ejecutarse en ese proceso. Varias ubicaciones en el núcleo se actualizaron inmediatamente para omitir la configuración del espacio de direcciones en modo de usuario, que incluyen:

1 El binario de modo de usuario ntdll.dll no está asignado al proceso de forma predeterminada.

2 El bloque de entorno de proceso (PEB) no se crea.

3 La sección de datos de usuario compartidos no está asignada al proceso. Este es un bloque de memoria asignada de solo lectura en todo el espacio de direcciones en modo de usuario para la recuperación eficiente de información común de todo el sistema.

3.2. Llamadas al sistema - syscall

Una syscall es un servicio proporcionado por el núcleo al que se puede llamar desde el modo de usuario que normalmente maneja las solicitudes de acceso a dispositivos u otras operaciones privilegiadas.

La forma en que se realiza una llamada al sistema depende del sistema operativo y la arquitectura del procesador, sin embargo, esto puede resumirse a tener ün contrato explícito.[en]tre el modo de usuario y el modo de núcleo llamado Interfaz Binaria de Aplicación (ABI). Para la mayoría de los casos, realizar una llamada al sistema se divide en 3 pasos:

1 Parámetros Marshall: el modo de usuario coloca los parámetros y el número de syscall en las ubicaciones definidas por el ABI.

2 Instrucción especial: el modo de usuario utiliza una instrucción de procesador especial para pasar al modo kernel para la llamada al sistema.

3 Maneje el retorno: después de que se realiza el servicio de syscall, el núcleo utiliza una instrucción especial del procesador para regresar al modo de usuario y el modo de usuario verifica el valor de retorno.

La convención de llamada para syscalls en WSL sigue la descripción de Linux ya que se están ejecutando los archivos binarios Linux ELF64 no modificados. WSL incluye controladores pico de modo kernel (lxss.sys y lxcore.sys) que son responsables de manejar las solicitudes de syscall de Linux en coordinación con el kernel NT. Los controladores no contienen código del kernel de Linux, sino que son una implementación de interfaces de kernel compatibles con Linux.

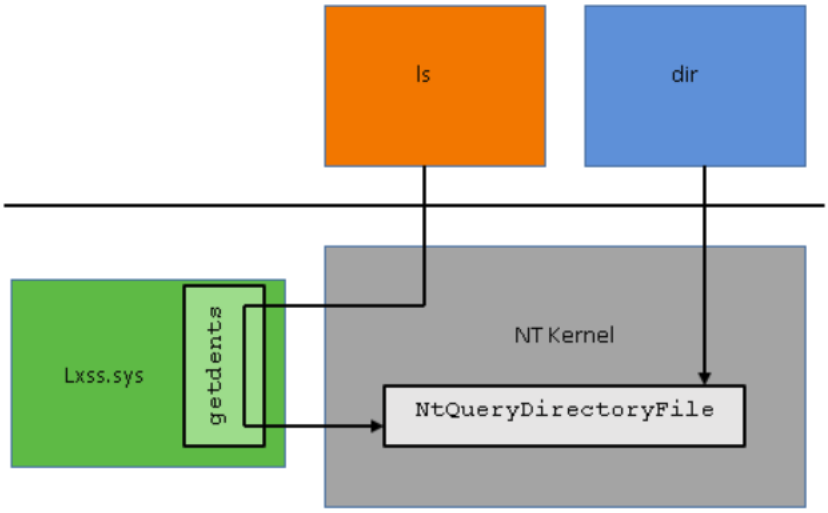

Figura 3: Diagrama de comunicación entre los Kernel para realizar una llamada al sistema

Dado que el kernel NT no sabe cómo manejar las llamadas al sistema de los procesos pico, guarda el estado de registro y reenvía la solicitud al controlador pico. El pico driver determina qué syscall de Linux se invoca al inspeccionar el registro rax y pasa los parámetros utilizando los registros definidos por la convención de llamada de syscall de Linux. Una vez que el controlador pico ha manejado la llamada syscall, regresa a NT, que restaura el estado del registro, coloca el valor de retorno en rax e invoca la instrucción sysret iretq para regresar al modo de usuario.

4. Conclusiones

Lo subsistemas de Windows para Linux es una alternativa bastante llamativa para usuarios de Windows que quieren indagar o curiosear en el mundo de Linux si tener la necesidad de usar una máquina virtual o hacer un dualbot, a pesar de que Microsoft planteé que WSL está destinado a desarrolladores de Windows que quieran ejecutar binarios de Linux, realmente dudamos que logré tener una mayor trascendencia debido a la poca versatilidad de desarrollo que se tiene todavía en la versión actual.

Por otro lado, sin duda, el haber realizado esta exposición no solamente nos ha ampliado nuestra visión al ver ventajas y desventajas de tener un ambiente nativo y no emulado, sino que hemos conocido un poco de cómo trabajan las llamadas al sistema y el kernel de los sistemas operativos.

5. Referencias

1) Microsoft (2016),"Windows Subsystem for Linux Documentation", Recuperado el 22 de febrero de 2020, de: https://docs.microsoft.com/en-us/windows/wsl/about

2) Santhosh Gautham (2018),"Windows Subsystem for Linux Review", Recuperado el 22 de febrero de 2020, de https://hackernoon.com/windows-subsystem-for-linux-review-981aa7bfa43d

3) Smith Ethan (2015),"WSL-Programs",Recuperado el 22 de febrero de 2020, de https://github.com/ethanhs/WSL-Programs

4) Ubunlog (s.f),"WSL: Cómo instalar y usar el susbistema Ubuntu en Windows 10",Recuperado el 22 de febrero de 2020, de https://ubunlog.com/wsl-como-instalar-y-usar-el-susbistema-ubuntu-en-windows-10/

5) Microsoft (s.f),"Learn About Windows Console & Windows Subsystem For Linux (WSL)",Recuperado el 22 de febrero de 2020, de https://devblogs.microsoft.com/commandline/learn-about-windows-console-and-windows-subsystem-for-linux-wsl/

6) Microsoft Developer (2019), "The new Windows subsystem for Linux architecture: a deep dive - BRK3068", Recuperado el 22 de febrero de 2020, de https://youtu.be/lwhMThePdIo