

Documentación del subsistema de Windows para Linux

Artículo • 21/03/2023

El Subsistema de Windows para Linux (WSL) permite a los desarrolladores ejecutar un entorno de GNU/Linux, incluida la mayoría de las herramientas de línea de comandos, utilidades y aplicaciones, directamente en Windows, sin modificar y sin la sobrecarga de una máquina virtual tradicional o una configuración de arranque dual.

[Instalación de WSL](#)

Saber más

- [¿Qué es el Subsistema de Windows para Linux \(WSL\)?](#)
- [Novedades de WSL 2](#)
- [Comparación de WSL 1 con WSL 2](#)
- [Preguntas más frecuentes](#)

Introducción

- [Instalación de WSL](#)
- [Instalación de Linux en Windows Server](#)
- [Pasos de instalación manual](#)
- [Procedimientos recomendados para configurar un entorno de desarrollo de WSL](#)

Únase al Programa Windows Insider y pruebe las características en versión preliminar de WSL.

Para probar las características o actualizaciones más recientes de WSL, únase al [Programa Windows Insider](#). Una vez que se haya unido a Windows Insider, puede elegir el canal del que le gustaría recibir las compilaciones preliminares en el menú de configuración de Windows. Puede elegir entre:

- Canal para desarrolladores: actualizaciones más recientes, pero con poca estabilidad.
- Canal beta: ideal para los usuarios pioneros; las compilaciones son más confiables que las del canal para desarrolladores.

- Canal de Versión Preliminar: correcciones de la versión preliminar y características clave de la siguiente versión de Windows justo antes de que esté disponible para el público general.

Blogs del equipo

- Publicación de información general con una colección de vídeos y blogs ↗
- Blog de línea de comandos ↗ (activo)
- Blog del subsistema de Windows para Linux (historial)

Envío de comentarios

- Seguimiento de problemas de GitHub: WSL ↗
- Documentación de seguimiento de problemas de GitHub: WSL ↗

Vídeos relacionados

ASPECTOS BÁSICOS DE WSL

1. [¿Qué es el Subsistema de Windows para Linux \(WSL\)? ↗](#) | Una pregunta de desarrollador (0:40)
2. [Soy desarrollador de Windows. ¿Por qué debería usar WSL? ↗](#) | Una pregunta de desarrollador (0:58)
3. [Soy desarrollador de Linux. ¿Por qué debería usar WSL? ↗](#) | Una pregunta de desarrollador (1:04)
4. [¿Qué es Linux? ↗](#) | Una pregunta de desarrollador (1:31)
5. [¿Qué es una distribución de Linux? ↗](#) | Una pregunta de desarrollador (1:04)
6. [¿En qué se diferencia WSL de una máquina virtual o un arranque dual? ↗](#) | Una pregunta de desarrollador
7. [¿Por qué se creó el Subsistema de Windows para Linux? ↗](#) | Una pregunta de desarrollador (1:14)
8. [¿Cómo accedo a los archivos de mi equipo en WSL? ↗](#) | Una pregunta de desarrollador (1:41)
9. [¿Cómo se integra WSL con Windows? ↗](#) | Una pregunta de desarrollador (1:34)
10. [¿Cómo configuro una distribución de WSL para que se inicie en el directorio principal en el Terminal? ↗](#) | Una pregunta de desarrollador (0:47)
11. [¿Puedo usar WSL para scripts? ↗](#) | Una pregunta de desarrollador (1:04)
12. [¿Por qué quiero usar herramientas de Linux en Windows? ↗](#) | Una pregunta de desarrollador (1:20)

13. En WSL, ¿puedo usar distribuciones distintas de las de Microsoft Store? ↗ | Una pregunta de desarrollador (1:03)

DEMOSTRACIONES DE WSL

1. [WSL2: ¡Escriba código más rápido en el Subsistema de Windows para Linux! ↗](#) | Tabulaciones y espacios (13:42)
2. [WSL: Ejecución de aplicaciones de Linux con interfaz gráfica de usuario ↗](#) | Tabulaciones y espacios (17:16)
3. [WSL 2: Conexión de dispositivos USB ↗](#) | Tabulaciones y espacios (10:08)
4. [Aprendizaje automático acelerado por GPU con WSL 2 ↗](#) | Tabulaciones y espacios (16:28)
5. [Visual Studio Code: Desarrollo remoto con SSH, máquinas virtuales y WSL ↗](#) | Tabulaciones y espacios (29:33)
6. [Actualizaciones de las herramientas de desarrollo de Windows: WSL, Terminal, Administrador de paquetes y mucho más ↗](#) | Tabulaciones y espacios (20:46)
7. [Compilación de aplicaciones Node.js con WSL ↗](#) | Destacados (3:15)
8. [Nueva característica de reclamación de memoria en WSL 2 ↗](#) | Demostración (6:01)
9. [Desarrollo web en Windows \(en 2019\) ↗](#) | Demostración (10:39)

ANÁLISIS PROFUNDOS DE WSL

1. [WSL en Windows 11: Demostraciones con Craig Loewen y Scott Hanselman ↗](#) | Miércoles de Windows (35:48)
2. [Distribuciones de WSL y Linux: Hayden Barnes y Kayla Cinnamon ↗](#) | Miércoles de Windows (37:00)
3. [Personalización del terminal con Oh My Posh y distribuciones Linux en WSL ↗](#) | Miércoles de Windows (33:14)
4. [Los desarrolladores web Sarah Tamsin y Craig Loewen charlan sobre el desarrollo web, la creación de contenido y WSL ↗](#) | Perspectivas de desarrollo (12:22)
5. [Cómo WSL accede a los archivos de Linux desde Windows ↗](#) | Profundización (24:59)
6. [Arquitectura del Subsistema de Windows para Linux: descripción detallada ↗](#) Build 2019 (58:10)

¿Qué es el Subsistema de Windows para Linux?

Artículo • 05/12/2023

Subsistema de Windows para Linux (WSL) es una característica de Windows que permite ejecutar un entorno Linux en la máquina Windows, sin necesidad de una máquina virtual independiente ni de arranque dual. WSL está diseñado para proporcionar una experiencia perfecta y productiva para los desarrolladores que quieren usar Windows y Linux al mismo tiempo.

- Use WSL para instalar y ejecutar varias distribuciones de Linux, como Ubuntu, Debian, Kali, etc. [Instale distribuciones de Linux](#) y reciba actualizaciones automáticas de [Microsoft Store](#), importe distribuciones de Linux no disponibles en [Microsoft Store](#), o bien [cree una distribución propia de Linux para el cliente](#).
- Almacene archivos en un sistema de archivos Linux aislado, específico de la distribución instalada.
- Ejecute herramientas de línea de comandos, como BASH.
- Ejecute herramientas de línea de comandos comunes de BASH, como `grep`, `sed`, `awk` u otros archivos binarios ELF-64.
- Ejecute scripts de Bash y aplicaciones de línea de comandos de GNU/Linux, como:
 - Herramientas: vim, emacs, tmux.
 - Lenguajes: [NodeJS](#), JavaScript, [Python](#), Ruby, C/C++, C# & F#, Rust, Go, etc.
 - Servicios: SSHD, [MySQL](#), Apache, lighttpd, [MongoDB](#), [PostgreSQL](#).
- Instala software adicional mediante el administrador de paquetes de distribución de GNU/Linux.
- Invoca aplicaciones de Windows mediante un shell de línea de comandos de tipo UNIX.
- Invoca aplicaciones de GNU/Linux en Windows.
- [Ejecución de aplicaciones gráficas GNU/Linux](#) que están integradas directamente en el escritorio de Windows
- Use la GPU del dispositivo para acelerar las cargas de trabajo de aprendizaje automático que se ejecutan en Linux.

[Instalación de WSL](#)

[https://www.youtube-nocookie.com/embed/48k317kOxqg ↗](https://www.youtube-nocookie.com/embed/48k317kOxqg)

¿Qué es WSL 2?

WSL 2 es el tipo de distribución predeterminado al instalar una distribución de Linux. WSL 2 usa tecnología de virtualización para ejecutar un kernel de Linux en una máquina virtual (VM) de utilidad ligera. Las distribuciones de Linux se ejecutan como contenedores aislados dentro de la máquina virtual administrada de WSL 2. Las distribuciones de Linux que se ejecutan desde WSL 2 compartirán el mismo espacio de nombres de red, árbol de dispositivos (distinto de `/dev/pts`), CPU, kernel, memoria o intercambio, `/init` binario, pero tienen su propio espacio de nombres PID, espacio de nombres de montaje, espacio de nombres de usuario, espacio de nombres de grupo de C y proceso `init`.

WSL 2 aumenta el rendimiento del sistema de archivos y agrega compatibilidad completa de llamadas del sistema en comparación con la arquitectura de WSL 1. Obtenga más información sobre cómo [se comparan WSL 1 y WSL 2](#).

Las distribuciones de Linux individuales se pueden ejecutar con la arquitectura de WSL 1 o WSL 2. Cada distribución se puede actualizar o degradar en cualquier momento, y puedes ejecutar distribuciones de WSL 1 y WSL 2 en paralelo. Vea [Establecimiento del comando de versión de WSL](#).

<https://www.youtube-nocookie.com/embed/MrZolfGm8Zk>

Relación entre Microsoft y Linux

Obtenga más información sobre los [recursos de Linux en Microsoft](#), incluidas las herramientas de Microsoft que se ejecutan en Linux, cursos de aprendizaje de Linux, arquitectura de soluciones en la nube para Linux y noticias, eventos y asociaciones de Microsoft y Linux. [Relación entre Microsoft y Linux](#)

Comparación de las versiones de WSL

Artículo • 10/11/2023

Obtenga más información sobre las distintas versiones de WSL, como por ejemplo el motivo por el que WSL 2 es ahora la predeterminada, así como los escenarios o excepciones específicos que pueden garantizar el cambio de la distribución de Linux instalada a la arquitectura anterior de WSL 1.

Comparación de WSL 1 con WSL 2

En esta guía se compara WSL 1 con WSL 2, incluidas las [excepciones para usar WSL 1 en lugar de WSL 2](#). Las principales diferencias entre WSL 1 y WSL 2 son el uso de un kernel de Linux real dentro de una VM administrada, la compatibilidad completa con las llamadas del sistema y el rendimiento en los sistemas operativos Linux y Windows. WSL 2 es la versión predeterminada actual al instalar una distribución de Linux y usa lo más grande y reciente en tecnología de virtualización para ejecutar un kernel de Linux dentro de una máquina virtual (VM) de utilidad ligera. Si su distribución ejecuta actualmente WSL 1 y desea actualizar a WSL 2, consulte [actualización de WSL 1 a WSL 2](#).

Comparación de características

Característica	WSL 1	WSL 2
Integración entre Windows y Linux	✓	✓
Tiempos de arranque rápidos	✓	✓
Huella de recurso pequeña en comparación con las máquinas virtuales tradicionales	✓	✓
Se ejecuta con las versiones actuales de VMWare y VirtualBox	✓	✓
VM administradas	✗	✓
Kernel de Linux completo	✗	✓
Compatibilidad completa con las llamadas del sistema	✗	✓
Rendimiento entre sistemas de archivos del sistema operativo	✓	✗
compatibilidad con systemd	✗	✓
Compatibilidad de IPv6	✗	✓

Como puede ver en la tabla de comparación anterior, la arquitectura de WSL 2 supera a WSL 1 de varias maneras, a excepción del rendimiento en los sistemas de archivos del sistema operativo, lo que se puede solucionar almacenando los archivos del proyecto en el mismo sistema operativo que las herramientas que se ejecutan para trabajar en el proyecto.

WSL 2 solo está disponible en Windows 11 o Windows 10, versión 1903, compilación 18362 o posterior. Para comprobar la versión de Windows, selecciona la **tecla del logotipo de Windows + R**, escribe **winver** y selecciona **Aceptar**. (También puedes escribir el comando **ver** en el símbolo del sistema de Windows). Es posible que tengas que [actualizar a la versión más reciente de Windows](#). En el caso de las compilaciones inferiores a 14393, no se admite WSL en absoluto.

Para más información sobre las últimas actualizaciones de WSL 2, consulta el [blog de la línea de comandos de Windows](#), incluida la [compatibilidad con Systemd que ya está disponible en WSL](#) y la [actualización de WSL de septiembre de 2023](#) para obtener más información sobre la compatibilidad con IPv6.

ⓘ Nota

WSL 2 funcionará con [VMWare 15.5.5 y versiones posteriores](#) y [VirtualBox 6 y versiones posteriores](#). Obtenga más información en nuestras [preguntas más frecuentes](#).

Novedades de WSL 2

WSL 2 es una revisión importante de la arquitectura subyacente y usa tecnología de virtualización y un kernel de Linux para habilitar las nuevas características. Los principales objetivos de esta actualización son aumentar el rendimiento del sistema de archivos y agregar compatibilidad completa con las llamadas del sistema.

- [Requisitos del sistema de WSL 2](#)
- [Establecimiento de la versión de la distribución de Linux de WSL 1 a WSL 2.](#)
- [Preguntas más frecuentes sobre WSL 2](#)

Arquitectura de WSL 2

Una experiencia de VM tradicional puede presentar un arranque lento, aislamiento y un consumo excesivo de recursos, además de exigir tiempo para su administración. WSL 2 no tiene estos atributos.

WSL 2 ofrece las ventajas de WSL 1, incluida una integración perfecta entre Windows y Linux, tiempos de arranque más breves y una superficie de recursos pequeña. Además, no requiere ninguna configuración ni administración de las VM. Aunque WSL 2 usa una VM, se administra y se ejecuta en segundo plano, lo que te permite disfrutar de la misma experiencia de usuario que WSL 1.

Kernel de Linux completo

Microsoft ha creado el kernel de Linux en WSL 2 a partir de la rama estable más reciente. Para ello, se basó en el origen disponible en [kernel.org](#). Este kernel se ha ajustado especialmente para WSL 2, y se ha optimizado su tamaño y rendimiento para ofrecer una increíble experiencia de Linux en Windows. El kernel recibirá actualizaciones de Windows, lo que significa que obtendrás las correcciones de seguridad y las mejoras del kernel más recientes sin que debas administrarlo por tu cuenta.

El [kernel de Linux para WSL 2 es de código abierto](#). Si quieres obtener más información, consulta la entrada de blog [Envío de un kernel de Linux con Windows](#), escrita por el equipo que lo creó.

Obtenga más información en las [Notas de la versión del Subsistema de Windows para el kernel de Linux](#).

Mayor rendimiento de E/S de archivos

Las operaciones de uso intensivo de archivos, como git clone, npm install, apt update, apt upgrade y otras, son considerablemente más rápidas con WSL 2.

El aumento de la velocidad real dependerá de la aplicación que se esté ejecutando y de cómo interactúes con el sistema de archivos. Las versiones iniciales de WSL 2 se ejecutan hasta 20 veces más rápido en comparación con WSL 1 al desempaquetar un archivo tarball comprimido, y aproximadamente entre 2y 5 veces más rápido cuando se usan git clone, npm install y cmake en distintos proyectos.

Compatibilidad completa con las llamadas del sistema

Los archivos binarios de Linux usan llamadas del sistema para ejecutar funciones, como el acceso a archivos, la solicitud de memoria, la creación de procesos, etc. Mientras que WSL 1 usaba una capa de traducción creada por el equipo de WSL, WSL 2 incluye su propio kernel de Linux con total compatibilidad con las llamadas del sistema. Entre las ventajas se incluyen las siguientes:

- Un conjunto de aplicaciones nuevo y completo que se pueden ejecutar en WSL, como [Docker](#), entre otros.
- Todas las actualizaciones del kernel de Linux están listas para su uso inmediato. (No tienes que esperar a que el equipo de WSL implemente actualizaciones y agregue los cambios).

Excepciones para usar WSL 1 en lugar de WSL 2

Te recomendamos que uses WSL 2, ya que ofrece un rendimiento mayor y compatibilidad completa con las llamadas del sistema. Sin embargo, hay algunos escenarios específicos en los que podrías preferir el uso de WSL 1. Considera el uso de WSL 1 en los casos siguientes:

- Los archivos de proyecto se deben almacenar en el sistema de archivos de Windows. WSL 1 ofrece un acceso más rápido a los archivos montados desde Windows.
 - Si vas a usar tu distribución de WSL para Linux para acceder a los archivos de proyecto en el sistema de archivos de Windows, y estos archivos no se pueden almacenar en el sistema de archivos de Linux, obtendrás un mayor rendimiento en los sistemas de archivos del sistema operativo mediante WSL 1.
- Un proyecto requiere la compilación cruzada con las herramientas de Windows y Linux en los mismos archivos.
 - El rendimiento de los archivos en los sistemas operativos Windows y Linux es mayor en WSL 1 que en WSL 2, por lo que, si usas aplicaciones Windows para acceder a archivos de Linux, obtendrás un mayor rendimiento con WSL 1.
- El proyecto necesita acceso a un puerto serie o un dispositivo USB. *Sin embargo*, la compatibilidad con dispositivos USB ahora está disponible para WSL 2 a través del proyecto USBIPD-WIN. Consulte [Conexión de dispositivos USB](#) para ver los pasos de configuración.
- WSL 2 no incluye compatibilidad para acceder a los puertos serie. Obtenga más información en las [preguntas más frecuentes](#) o en el [problema del repositorio de WSL en GitHub sobre la compatibilidad de serie ↗](#).
- Tiene requisitos de memoria estrictos
 - El uso de memoria de WSL 2 aumenta y se reduce al usarla. Cuando un proceso libera memoria, se devuelve automáticamente a Windows. Sin embargo, a partir de ahora WSL 2 no libera todavía las páginas almacenadas en caché de nuevo en Windows hasta que se cierra la instancia de WSL. Si tiene sesiones de WSL de larga duración o tiene acceso a una gran cantidad de archivos, esta memoria caché puede ocupar memoria en Windows. Estamos realizando un seguimiento

del trabajo para mejorar esta experiencia en el [problema 4166 del repositorio de GitHub de WSL](#).

- Para quienes usan VirtualBox, es posible que se deba tener en cuenta la versión que se está ejecutando y si es compatible con WSL 2. (Consulte el [problema 798 del repositorio de GitHub de WSL](#) para obtener una explicación completa). Parece que la versión 6.1.16 de VirtualBox funciona con WSL 2, pero que otras versiones pueden experimentar problemas).
- Si se basa en una distribución de Linux para tener una dirección IP en la misma red que el equipo host, es posible que tenga que configurar una solución alternativa para ejecutar WSL 2. WSL 2 se ejecuta como máquina virtual de Hyper-V. Se trata de un cambio con respecto al adaptador de red con puente usado en WSL 1, lo que significa que WSL 2 usa un servicio de traducción de direcciones de red (NAT) para su red virtual, en lugar de hacer puente en la tarjeta de interfaz de red (NIC) del host, lo que da lugar a una dirección IP única que cambiará en el reinicio. Para obtener más información sobre el problema y la solución alternativa que reenvía los puertos TCP de los servicios de WSL 2 al sistema operativo host, consulte el [problema 4150 del repositorio de GitHub de WSL](#) sobre el modo de puente NIC (solución alternativa de TCP).

Nota

Considera la posibilidad de probar la [extensión Remote WSL](#) de VS Code para poder almacenar los archivos de proyecto en el sistema de archivos de Linux, mediante las herramientas de línea de comandos de Linux, pero también con VS Code en Windows para crear, editar, depurar o ejecutar el proyecto en un explorador de Internet, sin experimentar las ralentizaciones de rendimiento asociadas con el trabajo en los sistemas de archivos de Windows. [Más información.](#)

WSL en Microsoft Store

WSL ha quitado la funcionalidad de actualización de la imagen del sistema operativo Windows y la ha incluido en un paquete que está disponible a través de Microsoft Store, lo que agiliza las actualizaciones y el mantenimiento en cuanto estén disponibles, ya que no es preciso esperar a una actualización del sistema operativo Windows.

WSL se incluyó originalmente en el sistema operativo Windows como un componente opcional que debía habilitarse para instalar una distribución de Linux. La versión de WSL que se encuentra en Microsoft Store tiene la misma experiencia de usuario, ya que es el mismo producto, pero recibe actualizaciones y mantenimiento como un paquete de Microsoft Store, en lugar de como una actualización completa del sistema operativo. A

partir de la versión 19044 de Windows, al ejecutar el comando `wsl.exe --install` se instalará la actualización de mantenimiento de WSL desde Microsoft Store. ([Consulte la entrada de blog que anuncia esta actualización ↗](#)). Si ya usa WSL, puede actualizarlo para asegurarse de que recibe las características y el mantenimiento de WSL más recientes desde Store. Para ello, debe ejecutar `wsl.exe --update`.

Comandos básicos para WSL

Artículo • 05/12/2023

Los comandos de WSL siguientes se enumeran en un formato compatible con PowerShell o el símbolo del sistema de Windows. Para ejecutar estos comandos desde una línea de comandos de distribución de Bash o Linux, debe reemplazar `wsl` por `wsl.exe`. Para obtener una lista completa de comandos, ejecute `wsl --help`. Si aún no lo ha hecho, le recomendamos [actualizar a la versión de WSL instalada desde Microsoft Store](#) para recibir actualizaciones de WSL tan pronto como estén disponibles. [Obtenga más información sobre cómo instalar WSL a través de Microsoft Store.](#)

Instalación

PowerShell

```
wsl --install
```

Instale WSL y la distribución de Ubuntu de Linux predeterminada. [Más información](#). También puede usar este comando para instalar distribuciones adicionales de Linux mediante la ejecución de `wsl --install <Distribution Name>`. Para obtener una lista válida de nombres de distribución, ejecute `wsl --list --online`.

Entre las opciones se incluyen:

- `--distribution`: especifique la distribución de Linux que se va a instalar. Para encontrar distribuciones disponibles, ejecute `wsl --list --online`.
- `--no-launch`: instale la distribución de Linux, pero no la inicie automáticamente.
- `--web-download`: instale desde un origen en línea en lugar de usar la tienda Microsoft.

Cuando WSL no está instalada, las opciones incluyen:

- `--inbox`: instala WSL mediante el componente de Windows en lugar de usar el almacén de Microsoft. (*Las actualizaciones de WSL se recibirán a través de las actualizaciones de Windows, en lugar de insertarse como disponibles a través de la tienda*).
- `--enable-wsl1`: habilita WSL 1 durante la instalación de la versión de Microsoft Store de WSL habilitando también el componente opcional "Subsistema de Windows para Linux".
- `--no-distribution`: no instale una distribución al instalar WSL.

⚠️ Nota

Si ejecuta WSL en Windows 10 o una versión anterior, es posible que tenga que incluir la marca `-d` con el comando `--install` para especificar una distribución:
`wsl --install -d <distribution name>`.

Enumeración de las distribuciones de Linux disponibles

PowerShell

```
wsl --list --online
```

Consulte una lista de las distribuciones de Linux disponibles a través de la tienda en línea. Este comando también se puede especificar como `wsl -l -o`.

Enumeración de las distribuciones de Linux instaladas

PowerShell

```
wsl --list --verbose
```

Consulte una lista de las distribuciones de Linux instaladas en la máquina Windows, incluido el estado (si la distribución se está ejecutando o detenida) y la versión de WSL que ejecuta la distribución (WSL 1 o WSL 2). [Comparación de WSL 1 con WSL 2](#). Este comando también se puede especificar como `wsl -l -v`. Entre las opciones adicionales que se pueden usar con el comando list se incluyen: `--all` para enumerar todas las distribuciones, `--running` para enumerar solo las distribuciones que se están ejecutando actualmente o `--quiet` para mostrar solo los nombres de la distribución.

Establecimiento de la versión de WSL en 1 o 2

PowerShell

```
wsl --set-version <distribution name> <versionNumber>
```

Para designar la versión de WSL (1 o 2) en la que se ejecuta una distribución de Linux, reemplace `<distribution name>` por el nombre de la distribución y `<versionNumber>`, por 1 o 2. [Comparación de WSL 1 con WSL 2](#). WSL 2 solo está disponible en Windows 11 o Windows 10, versión 1903, compilación 18362 o posterior.

Advertencia

El cambio entre WSL 1 y WSL 2 puede llevar mucho tiempo y provocar errores debido a las diferencias entre las dos arquitecturas. En el caso de las distribuciones con proyectos grandes, se recomienda realizar copias de seguridad de archivos antes de intentar una conversión.

Establecimiento de la versión de WSL predeterminada

PowerShell

```
wsl --set-default-version <Version>
```

Para establecer una versión predeterminada de WSL 1 o WSL 2, reemplace `<Version>` por el número 1 o 2 para representar la versión de WSL en la que desea que la instalación se establezca de forma predeterminada para las nuevas instalaciones de distribución de Linux. Por ejemplo: `wsl --set-default-version 2`. [Comparación de WSL 1 con WSL 2](#). WSL 2 solo está disponible en Windows 11 o Windows 10, versión 1903, compilación 18362 o posterior.

Establecimiento de la distribución de Linux predeterminada

PowerShell

```
wsl --set-default <Distribution Name>
```

Para establecer la distribución de Linux predeterminada que usarán los comandos de WSL para ejecutarse, reemplace `<Distribution Name>` por el nombre de la distribución de Linux que prefiera.

Cambio al directorio principal

```
PowerShell
```

```
wsl ~
```

~ se puede usar con wsl para iniciar en el directorio principal del usuario. Para volver de cualquier directorio al principal desde un símbolo del sistema de WSL, puede usar el comando cd ~.

Ejecución de una distribución de Linux específica desde PowerShell o CMD

```
PowerShell
```

```
wsl --distribution <Distribution Name> --user <User Name>
```

Para ejecutar una distribución de Linux específica con un usuario específico, reemplace <Distribution Name> por el nombre de la distribución de Linux que prefiera (por ejemplo, Debian) y <User Name>, por el nombre de un usuario existente (por ejemplo, raíz). Si el usuario no existe en la distribución de WSL, recibirá un error. Para imprimir el nombre de usuario actual, use el comando whoami.

Actualización de WSL

```
PowerShell
```

```
wsl --update
```

Actualice la versión de WSL a la versión más reciente. Entre las opciones se incluyen:

- --web-download: descargue la actualización más reciente de GitHub en lugar de la tienda Microsoft.

Comprobación del estado de WSL

```
PowerShell
```

```
wsl --status
```

Consulte la información general sobre la configuración de WSL, como el tipo de distribución predeterminado, la distribución predeterminada y la versión del kernel.

Comprobación de la versión de WSL

PowerShell

```
wsl --version
```

Compruebe la información de versión sobre WSL y sus componentes.

Comando help

PowerShell

```
wsl --help
```

Consulte una lista de las opciones y los comandos disponibles con WSL.

Ejecutar como usuario específico

PowerShell

```
wsl -u <Username>` , `wsl --user <Username>
```

Para ejecutar WSL como usuario especificado, reemplace `<Username>` por el nombre de un usuario que exista en la distribución de WSL.

Cambio del usuario predeterminado para una distribución

PowerShell

```
<DistributionName> config --default-user <Username>
```

Cambie el usuario predeterminado para el inicio de sesión de su distribución. El usuario ya debe existir dentro de la distribución para convertirse en el usuario predeterminado.

Por ejemplo: `ubuntu config --default-user johndoe` cambiaría el usuario predeterminado de la distribución de Ubuntu al usuario "johndoe".

ⓘ Nota

Si tiene problemas para averiguar el nombre de la distribución, use el comando `wsl -l`.

⚠ Advertencia

Este comando no funcionará para las distribuciones importadas, ya que estas distribuciones no tienen un iniciador ejecutable. En su lugar, puede cambiar el usuario predeterminado para las distribuciones importadas mediante el archivo `/etc/wsl.conf`. Consulte las opciones de montaje automático en el documento [Opciones de configuración avanzada](#).

Apagar

PowerShell

```
wsl --shutdown
```

Finaliza inmediatamente todas las distribuciones en ejecución y la máquina virtual de utilidad ligera de WSL 2. Este comando puede ser necesario en instancias que requieren que reinicie el entorno de máquina virtual de WSL 2, como [cambiar los límites de uso de memoria](#) o realizar un cambio en el archivo `.wslconfig`.

Terminate

PowerShell

```
wsl --terminate <Distribution Name>
```

Para finalizar la distribución especificada o detener su ejecución, reemplace `<Distribution Name>` por el nombre de la distribución de destino.

Identificar la dirección IP

- `wsl hostname -i` para la dirección IP de la distribución de Linux instalada a través de WSL 2 (la dirección de máquina virtual de WSL 2)
- `cat /etc/resolv.conf` para la dirección IP de la máquina Windows como se ve en WSL 2 (la máquina virtual WSL 2)

Importación y exportación de una distribución

PowerShell

```
wsl --export <Distribution Name> <FileName>
```

PowerShell

```
wsl --import <Distribution Name> <InstallLocation> <FileName>
```

Importa el archivo tar especificado como una nueva distribución. El nombre de archivo puede ser - para la entrada estándar. Entre las opciones se incluyen:

- `--vhdx`: especifica que la distribución import/export debe ser un archivo .vhdx en lugar de un archivo tar (solo se admite con WSL 2)
- `--version`: solo para importación, especifica si se va a importar la distribución como una distribución de WSL 1 o WSL 2.

Importación de una distribución en contexto

PowerShell

```
wsl --import-in-place <Distribution Name> <FileName>
```

Importa el archivo .vhdx especificado como una nueva distribución. El disco duro virtual debe tener formato en el tipo de sistema de archivos ext4.

Anular el registro de una distribución de Linux o desinstalarla

Aunque las distribuciones de Linux se pueden instalar a través de Microsoft Store, no se pueden desinstalar a través de Store.

Para anular el registro y desinstalar una distribución de WSL:

PowerShell

```
wsl --unregister <DistributionName>
```

Al reemplazar `<DistributionName>` por el nombre de la distribución de Linux de destino,

se anulará el registro de esa distribución de WSL para que se pueda reinstalar o limpiar.

Precaución: Una vez que se ha anulado el registro, todos los datos, la configuración y el software asociados a esa distribución se perderán de manera permanente. Si se vuelve a instalar desde Store, se instalará una copia limpia de la distribución. Por ejemplo, `wsl --unregister Ubuntu` quitaría Ubuntu de las distribuciones disponibles en WSL. Al ejecutar `wsl --list`, se verá que ya no aparece en la lista.

También puede desinstalar la aplicación de distribución de Linux en la máquina Windows igual que cualquier otra aplicación de Store. Para reinstalar, busque la distribución en Microsoft Store y seleccione "Iniciar".

Montar un disco o dispositivo

PowerShell

```
wsl --mount <DiskPath>
```

Conecte y monte un disco físico en todas las distribuciones de WSL 2. Para ello, reemplace `<DiskPath>` por la ruta de acceso del directorio o el archivo donde se encuentra el disco. Consulte [Montaje de un disco Linux en WSL 2](#). Las opciones son:

- `--vhd`: especifica que `<Disk>` hace referencia a un disco duro virtual.
- `--name`: monte el disco con un nombre personalizado para el punto de montaje.
- `--bare`: conecte el disco a WSL 2, pero no lo monte.
- `--type <Filesystem>`: tipo de sistema de archivos que se va a usar al montar un disco. Si no se especifica, se establece de manera predeterminada en ext4. Este comando también se puede especificar como `wsl --mount -t <Filesystem>`. Puede detectar el tipo de sistema de archivos mediante el comando `blkid <BlockDevice>` como, por ejemplo: `blkid <dev/sdb1>`.
- `--partition <Partition Number>`: número de índice de la partición que se va a montar. Si no se especifica, se establece de forma predeterminada en todo el disco.

- `--options <MountOptions>`: hay algunas opciones específicas del sistema de archivos que se pueden incluir al montar un disco. Por ejemplo, **opciones de montaje ext4 ↴**, como `wsl --mount -o "data-ordered"` o `wsl --mount -o "data=writeback"`. Sin embargo, en este momento solo se admiten las opciones específicas del sistema de archivos. No se admiten opciones genéricas, como `ro`, `rw` o `noatime`.

ⓘ Nota

Si ejecuta un proceso de 32 bits para acceder a `wsl.exe` (una herramienta de 64 bits), puede que tenga que ejecutar el comando de la siguiente manera:

```
C:\Windows\Sysnative\wsl.exe --command .
```

Desmontar discos

PowerShell

```
wsl --unmount <DiskPath>
```

Desmontar un disco dado en la ruta de acceso del disco, si no se da ninguna ruta de acceso de disco, este comando desmontará y desasociará todos los discos montados.

Comandos de WSL en desuso

PowerShell

```
wslconfig.exe [Argument] [Options]
```

PowerShell

```
bash [Options]
```

PowerShell

```
lxrun /[Argument]
```

Estos comandos eran la sintaxis `wsl` original para configurar las distribuciones de Linux instaladas con WSL, pero se han reemplazado por la sintaxis del comando `wsl` o `wsl.exe`.

Instalación de Linux en Windows con WSL

Artículo • 28/08/2023

Los desarrolladores pueden acceder a la potencia de Windows y Linux al mismo tiempo en una máquina Windows. Subsistema de Windows para Linux (WSL) permite a los desarrolladores instalar una distribución de Linux (como Ubuntu, OpenSUSE, Kali, Debian, Arch Linux, etc.) y usar aplicaciones, utilidades y herramientas de línea de comandos de Bash directamente en Windows, sin modificar, sin la sobrecarga de una máquina virtual tradicional o una configuración de arranque dual.

Prerrequisitos

Para ejecutar los siguientes comandos, debe ejecutar Windows 10 versión 2004 y posteriores (compilación 19041 y posteriores) o Windows 11. Si está en versiones anteriores, consulte [la página de instalación manual](#).

Comando de instalación de WSL

Ahora puede instalar todo lo que necesita para ejecutar WSL con un solo comando. Abra PowerShell o el símbolo del sistema de Windows como **administrador**; para ello, haga clic con el botón derecho y seleccione "Ejecutar como administrador", escriba el comando `wsl --install` y reinicie la máquina.

PowerShell

```
wsl --install
```

Este comando habilitará las características necesarias para ejecutar WSL e instalará la distribución Ubuntu de Linux. ([Esta distribución predeterminada se puede cambiar](#)).

Si está ejecutando una compilación anterior o simplemente prefiere no usar el comando `install` y desea instrucciones paso a paso, consulte [Pasos de instalación manual de WSL para versiones anteriores](#).

La primera vez que inicie una distribución de Linux recién instalada, se abrirá una ventana de la consola y se le pedirá que espere a que los archivos se descompriman y se almacenen en el equipo. Todos los inicios posteriores deberían tardar menos de un segundo en completarse.

ⓘ Nota

El comando anterior solo funciona si WSL no está instalado. Si ejecuta `wsl --install` y ve el texto de ayuda de WSL, intente ejecutar `wsl --list --online` para ver una lista de distribuciones disponibles y ejecute `wsl --install -d <DistroName>` para instalar una distribución. Para desinstalar WSL, consulte [Desinstalación de la versión heredada de WSL o Anulación del registro o desinstalación de una distribución de Linux](#).

Cambio de la distribución predeterminada de Linux instalada

De forma predeterminada, la distribución de Linux instalada será Ubuntu. Se puede cambiar mediante la marca `-d`.

- Para cambiar la distribución instalada, escriba: `wsl --install -d <Distribution Name>`. Reemplace `<Distribution Name>` por el nombre de la distribución que desea instalar.
- Para ver una lista de las distribuciones de Linux disponibles para descargar a través de la tienda en línea, escriba `wsl --list --online` o `wsl -l -o`.
- Para instalar distribuciones de Linux adicionales después de la instalación inicial, también puede usar el comando `wsl --install -d <Distribution Name>`.

💡 Sugerencia

Si desea instalar distribuciones adicionales desde dentro de una línea de comandos de Linux/Bash (en lugar de hacerlo desde PowerShell o el símbolo del sistema), debe usar .exe en el comando `wsl.exe --install -d <Distribution Name>` o para enumerar las distribuciones disponibles: `wsl.exe -l -o`.

Si experimenta un problema durante el proceso de instalación, consulte la [sección de instalación de la guía de solución de problemas](#).

Para instalar una distribución de Linux que no aparece como disponible, puede [importar cualquier distribución de Linux](#) mediante un archivo TAR. O bien, en algunos casos, como con Arch Linux[↗], puede instalarla mediante un archivo `.appx`. También puede crear su propia [distribución personalizada de Linux](#) para usarla con WSL.

Configuración de la información de usuario de Linux

Una vez que haya instalado WSL, deberá crear una cuenta de usuario y una contraseña para la distribución de Linux recién instalada. Consulte la guía [Procedimientos recomendados para configurar un entorno de desarrollo de WSL](#) para obtener más información.

Configuración y procedimientos recomendados

Se recomienda seguir nuestra guía [Procedimientos recomendados para configurar un entorno de desarrollo de WSL](#) para obtener un tutorial paso a paso sobre cómo configurar un nombre de usuario y una contraseña para las distribuciones de Linux instaladas mediante comandos básicos de WSL, la instalación y personalización de Terminal Windows, la configuración para el control de versiones de Git, la edición y depuración de código mediante el servidor remoto de VS Code, los procedimientos recomendados para el almacenamiento de archivos, la configuración de una base de datos, el montaje de una unidad externa, la configuración de la aceleración de la GPU, etc.

Comprobación de la versión de WSL que se está ejecutando

Para enumerar las distribuciones de Linux instaladas y comprobar en qué versión de WSL está establecida cada una, puede escribir el comando `wsl -l -v` en PowerShell o en el Símbolo del sistema de Windows.

Para establecer la versión predeterminada en WSL 1 o WSL 2 cuando se instala una nueva distribución de Linux, use el comando `wsl --set-default-version <Version#>`, reemplazando `<Version#>` por 1 o 2.

Para establecer la distribución predeterminada de Linux que se usa con el comando `wsl`, escriba `wsl -s <DistributionName>` o `wsl --setdefault <DistributionName>`, reemplazando `<DistributionName>` por el nombre de la distribución de Linux que le gustaría usar. Por ejemplo, en PowerShell/CMD, escriba `wsl -s Debian` para establecer la distribución predeterminada en Debian. Ahora, al ejecutar `wsl npm init` desde PowerShell, se ejecutará el comando `npm init` en Debian.

Para ejecutar una distribución de WSL específica desde PowerShell o el Símbolo del sistema de Windows sin cambiar la distribución predeterminada, use el comando `wsl -d <DistributionName>`, reemplazando `<DistributionName>` por el nombre de la distribución que desea usar.

Obtenga más información en la guía [Comandos básicos para WSL](#).

Actualización de la versión de WSL 1 a WSL 2

Las nuevas instalaciones de Linux, instaladas con el comando `wsl --install`, se establecerán en WSL 2 de forma predeterminada.

El comando `wsl --set-version` se puede usar para cambiar de WSL 2 a WSL 1 o para actualizar distribuciones de Linux instaladas previamente de WSL 1 a WSL 2.

Para ver si la distribución de Linux está establecida en WSL 1 o WSL 2, use el comando `wsl -l -v`.

Para cambiar de versión, use el comando: `wsl --set-version <distro name> 2`. Sustituya `<distro name>` por el nombre de la distribución de Linux que quiera actualizar. Por ejemplo, `wsl --set-version Ubuntu-20.04 2` establecerá la distribución de Ubuntu 20.04 para usar WSL 2.

Si instaló WSL manualmente antes de que estuviera disponible el comando `wsl --install`, es posible que también tenga que [habilitar el componente opcional de máquina virtual](#) usado por WSL 2 e [instalar el paquete de kernel](#) si aún no lo ha hecho.

Para obtener más información, consulte la [Referencia de comandos para WSL](#) para obtener una lista de comandos WSL, [Comparación de WSL 1 con WSL 2](#) para obtener instrucciones sobre cuál usar en su escenario de trabajo, o [Procedimientos recomendados para configurar un entorno de desarrollo de WSL](#) para obtener instrucciones generales sobre cómo configurar un buen flujo de trabajo de desarrollo con WSL.

Maneras de ejecutar varias distribuciones de Linux con WSL

WSL admite la ejecución de tantas distribuciones de Linux diferentes como quiera instalar. Esto puede incluir la elección de distribuciones de [Microsoft Store](#), la [importación de una distribución personalizada](#) o la [creación de su propia distribución personalizada](#).

Hay varias maneras de ejecutar las distribuciones de Linux una vez instaladas:

- **Instalar Terminal Windows(recomendado)** Terminal Windows admite tantas líneas de comandos como quiera instalar y le permite abrirlos en varias pestañas o paneles de ventanas y cambiar rápidamente entre varias distribuciones de Linux u otras líneas de comandos (PowerShell, Símbolo del sistema, CLI de Azure, etc.). Puede personalizar completamente el terminal con combinaciones de colores únicas, estilos de fuente, tamaños, imágenes de fondo y métodos abreviados de teclado personalizados. [Más información.](#)
- Para abrir directamente la distribución de Linux, puede ir al menú Inicio de Windows y escribir el nombre de las distribuciones instaladas. Por ejemplo: "Ubuntu". Se abrirá Ubuntu en su propia ventana de consola.
- Desde el Símbolo del sistema de Windows o PowerShell, puede escribir el nombre de la distribución instalada. Por ejemplo: `ubuntu`
- Desde el Símbolo del sistema de Windows o PowerShell, puede abrir la distribución de Linux predeterminada dentro de la línea de comandos actual; para ello, escriba: `wsl.exe`.
- Desde el Símbolo del sistema de Windows o PowerShell, puede usar la distribución de Linux predeterminada dentro de la línea de comandos actual sin escribir una nueva; para ello, escriba: `wsl [command]`. Reemplazar `[command]` por un comando de WSL, como `wsl -l -v`, para enumerar las distribuciones instaladas, o `wsl pwd`, para ver dónde está montada la ruta de acceso del directorio actual en WSL. En PowerShell, el comando `get-date` indicará la fecha del sistema de archivos de Windows, y `wsl date` indicara la fecha del sistema de archivos de Linux.

El método que seleccione debe depender de lo que esté haciendo. Si ha abierto una línea de comandos de WSL dentro de un Símbolo del sistema de Windows o una ventana de PowerShell y desea salir, escriba el comando `exit`.

¿Quiere probar las características de la versión preliminar de WSL más recientes?

Para probar las características o actualizaciones más recientes para WSL, únase al [Programa Windows Insider](#). Una vez que se haya unido a Windows Insider, puede elegir el canal en que quiera recibir las compilaciones preliminares desde el menú de configuración de Windows para recibir automáticamente las actualizaciones o las características de versión preliminar de WSL asociadas a esa compilación. Puede elegir entre:

- Canal para desarrolladores: actualizaciones más recientes, pero con poca estabilidad.
- Canal beta: ideal para los usuarios pioneros; las compilaciones son más confiables que las del canal para desarrolladores.
- Canal de Versión Preliminar: correcciones de la versión preliminar y características clave de la siguiente versión de Windows justo antes de que esté disponible para el público general.

Recursos adicionales

- [Blog Windows Command Line: Install WSL with a single command now available in Windows 10 version 2004 and higher ↗](#) (Instalación de WSL con un solo comando ahora disponible en Windows 10, versión 2004 y posteriores)

Pasos de instalación manual para versiones anteriores de WSL

Artículo • 05/12/2023

Por motivos de simplicidad, por lo general se recomienda usar `wsl --install` para instalar el Subsistema de Windows para Linux, pero si ejecuta una compilación anterior de Windows, es posible que no se admita. Hemos incluido los pasos de instalación manual a continuación. Si experimenta un problema durante el proceso de instalación, consulte la [sección de instalación de la guía de solución de problemas](#).

Paso 1: Habilitación del Subsistema de Windows para Linux

Antes de instalar distribuciones de Linux en Windows, debe habilitar la característica opcional "Subsistema de Windows para Linux".

Abra PowerShell como **administrador** (menú Inicio > **PowerShell** > haga clic con el botón derecho en > Ejecutar como administrador) y escriba este comando:

```
PowerShell  
  
dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart
```

Ahora se recomienda continuar con el paso 2, Actualización a WSL 2, pero si solo quiere instalar WSL 1, ahora puede **reiniciar** el equipo y dirigirse al [Paso 6: Instalación de la distribución de Linux que quiera](#). Para actualizar a WSL 2, **espere para reiniciar** la máquina y continúe con el paso siguiente.

Paso 2: comprobación de los requisitos para ejecutar WSL 2

Para actualizar a WSL 2, debe ejecutar Windows 10...

- Para sistemas x64: **versión 1903** o posterior, con la **compilación 18362.1049** o posterior.
- Para sistemas ARM64: **versión 2004** o posterior, con la **compilación 19041** o posterior.

o Windows 11.

ⓘ Nota

Las compilaciones anteriores a 18362 no admiten WSL 2. Use el [Asistente para Windows Update](#) para actualizar su versión de Windows. La compatibilidad con Windows versión 1903 también es solo para sistemas x64. Si usa una versión arm64 de Windows, deberá actualizar a Windows 10 versión 2004 o posterior para obtener acceso completo a WSL 2. Para obtener más información, consulta [compatibilidad con WSL 2 que viene a Windows 10 versiones 1903 y 1909](#).

Para comprobar la versión y el número de compilación, seleccione la [tecla del logotipo de Windows + R](#), escriba `winver` y seleccione **Aceptar**. [Actualice a la versión más reciente de Windows](#) en el menú Configuración.

ⓘ Nota

Si está ejecutando Windows 10, versión 1903 o 1909, abra "Configuración" en el menú de Windows, vaya a "Actualización & seguridad" y seleccione "Buscar actualizaciones". El número de compilación debe ser 18362.1049 o posterior o 18363.1049 o posterior, con la compilación secundaria posterior a .1049. Leer más: [La compatibilidad con WSL 2 estará disponible en breve para las versiones 1903 y 1909 de Windows 10](#).

Paso 3: Habilitación de la característica Máquina virtual

Antes de instalar WSL 2, debe habilitar la característica opcional **Plataforma de máquina virtual**. La máquina necesitará [funcionalidades de virtualización](#) para usar esta característica.

Abre PowerShell como administrador y ejecuta:

```
PowerShell
```

```
dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all  
/norestart
```

Reinicia la máquina para completar la instalación de WSL y la actualización a WSL 2.

Paso 4: Descarga del paquete de actualización del kernel de Linux

El paquete de actualización del kernel de Linux instala la versión más reciente del [kernel de Linux de WSL 2](#) para ejecutar WSL dentro de la imagen del sistema operativo Windows. (Para ejecutar [WSL desde Microsoft Store](#), con actualizaciones insertadas con más frecuencia, use `wsl.exe --install` o `wsl.exe --update`).

1. Descargue la versión más reciente:

- [Paquete de actualización del kernel de Linux en WSL 2 para máquinas x64](#)

ⓘ Nota

Si estás usando una máquina ARM64, descarga el [paquete ARM64](#) en su lugar. Si no está seguro de qué tipo de máquina tiene, abra el símbolo del sistema o PowerShell y escriba: `systeminfo | find "System Type"`.

Advertencia: En versiones de Windows que no están en inglés, es posible que tenga que modificar el texto de búsqueda, traduciendo la cadena "System Type" (Tipo de sistema). Es posible que también tenga que escapar las comillas del comando find. Por ejemplo, en alemán, `systeminfo | find '"Systemtyp"'`.

2. Ejecuta el paquete de actualización que descargaste en el paso anterior. (Haga doble clic para ejecutarlo. Se le pedirán permisos elevados. Seleccione "Sí" para aprobar esta instalación).

Una vez completada la instalación, vaya al paso siguiente: configuración de WSL 2 como versión predeterminada al instalar nuevas distribuciones de Linux. (Omita este paso si quiere que las nuevas instalaciones de Linux se establezcan en WSL 1).

ⓘ Nota

Para obtener más información, consulta el artículo [cambios en la actualización del kernel de Linux en WSL2](#), disponible en el [blog de la línea de comandos de Windows](#).

Paso 5: Definición de WSL 2 como versión predeterminada

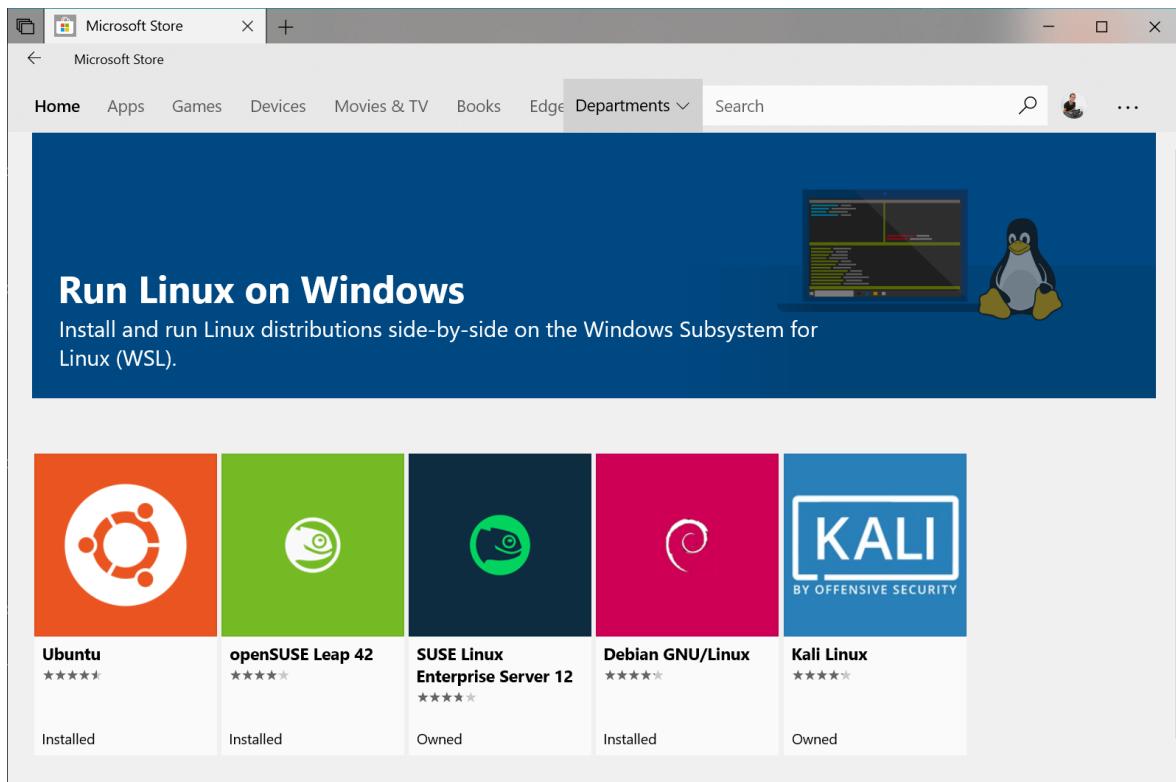
Abra PowerShell y ejecute este comando para establecer WSL 2 como versión predeterminada al instalar una nueva distribución de Linux:

```
PowerShell

wsl --set-default-version 2
```

Paso 6: Instalación de la distribución de Linux que quiera

1. Abre [Microsoft Store](#) y selecciona tu distribución de Linux favorita.



En los vínculos siguientes se abrirá la página de Microsoft Store para cada distribución:

- [Ubuntu 18.04 LTS ↗](#)
- [Ubuntu 20.04 LTS ↗](#)
- [Ubuntu 22.04 LTS ↗](#)
- [OpenSUSE Leap 15.1 ↗](#)
- [SUSE Linux Enterprise Server 12 SP5 ↗](#)
- [SUSE Linux Enterprise Server 15 SP1 ↗](#)
- [Kali Linux ↗](#)
- [Debian GNU/Linux ↗](#)
- [Fedora Remix for WSL ↗](#)

- [Pengwin ↗](#)
- [Pengwin Enterprise ↗](#)
- [Alpine WSL ↗](#)
- [Raft \(prueba gratuita\) ↗](#)
- [Alma Linux ↗](#)

2. En la página de la distribución, selecciona "Obtener".

Screenshots

Description

Ubuntu on Windows allows one to use Ubuntu Terminal and run Ubuntu command line utilities including bash, ssh, git, apt and many more.

To use this feature, one first needs to use "Turn Windows features on or off" and select "Windows Subsystem for Linux", click OK, reboot, and use this app.

The above step can also be performed using Administrator PowerShell prompt: Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux

More

Available on

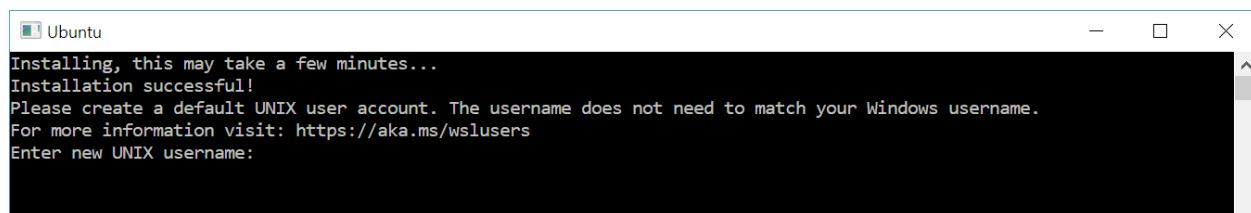
PC

Features

Ubuntu
bash
ssh

La primera vez que inicies una distribución de Linux recién instalada, se abrirá una ventana de la consola y se te pedirá que esperes un minuto o dos para que los archivos se descompriman y se almacenen en tu equipo. Todos los inicios posteriores deberían tardar menos de un segundo en completarse.

Tendrás que [crear una cuenta de usuario y una contraseña para la nueva distribución de Linux](#).



¡ENHORABUENA! Ha instalado y configurado correctamente una distribución de Linux completamente integrada con el sistema operativo Windows.

Solución de problemas de instalación

Si experimenta un problema durante el proceso de instalación, consulte la [sección de instalación de la guía de solución de problemas](#).

Descarga de distribuciones

Hay varios escenarios en los que quizás no pueda (o no quiera) instalar distribuciones de Linux de WSL a través de Microsoft Store. Es posible que esté ejecutando una SKU de sistema operativo de escritorio de Windows Server o de mantenimiento a largo plazo (LTSC) que no sea compatible con Microsoft Store, o que sus administradores o directivas de red corporativa no permitan el uso de Microsoft Store en su entorno. En estos casos, aunque WSL esté disponible, es posible que tenga que descargar las distribuciones de Linux directamente.

Si la aplicación Microsoft Store no está disponible, puede descargar e instalar manualmente distribuciones de Linux a través de estos vínculos:

- [Ubuntu ↗](#)
- [Ubuntu 22.04 LTS ↗](#)
- [Ubuntu 20.04 ↗](#)
- [Ubuntu 20.04 ARM ↗](#)
- [Ubuntu 18.04 ↗](#)
- [Ubuntu 18.04 ARM ↗](#)
- [Ubuntu 16.04 ↗](#)
- [Debian GNU/Linux ↗](#)
- [Kali Linux ↗](#)
- [SUSE Linux Enterprise Server 12|↗](#)
- [SUSE Linux Enterprise Server 15 SP2 ↗](#)
- [SUSE Linux Enterprise Server 15 SP3 ↗](#)
- [openSUSE Tumbleweed ↗](#)
- [openSUSE Leap 15.3 ↗](#)
- [OpenSUSE Leap 15.2 ↗](#)
- [Oracle Linux 8.5 ↗](#)
- [Oracle Linux 7.9 ↗](#)
- [Fedora Remix for WSL ↗](#)

Al hacerlo, los paquetes de `<distro>.appx` se descargarán en una carpeta de tu elección.

Si lo prefiere, también puede descargar sus distribuciones preferidas a través de la línea de comandos y puede usar PowerShell con el cmdlet [Invoke-WebRequest](#). Por ejemplo,

para descargar Ubuntu 20.04:

PowerShell

```
Invoke-WebRequest -Uri https://aka.ms/wslubuntu2004 -OutFile Ubuntu.appx -  
UseBasicParsing
```

💡 Sugerencia

Si la descarga tarda mucho tiempo, establece `$ProgressPreference = 'SilentlyContinue'` para desactivar la barra de progreso.

También tiene la opción de usar la utilidad [de línea de comandos curl](#) para la descarga. Para descargar Ubuntu 20.04 con curl:

Consola

```
curl.exe -L -o ubuntu-2004.appx https://aka.ms/wslubuntu2004
```

En este ejemplo, se ejecuta `curl.exe` (no solo `curl`) para garantizar que, en PowerShell, se invoque el ejecutable de curl real, no el alias de curl de PowerShell para [Invoke-WebRequest](#).

Una vez descargada la distribución, vaya a la carpeta que contiene la descarga y ejecute el siguiente comando en ese directorio, donde `app-name` es el nombre del archivo .appx de la distribución de Linux.

Powershell

```
Add-AppxPackage .\app_name.appx
```

Una vez que el paquete Appx haya terminado de descargarse, puede empezar a ejecutar la nueva distribución haciendo doble clic en el archivo appx. (El comando `wsl -l` no mostrará que la distribución está instalada hasta que se complete este paso).

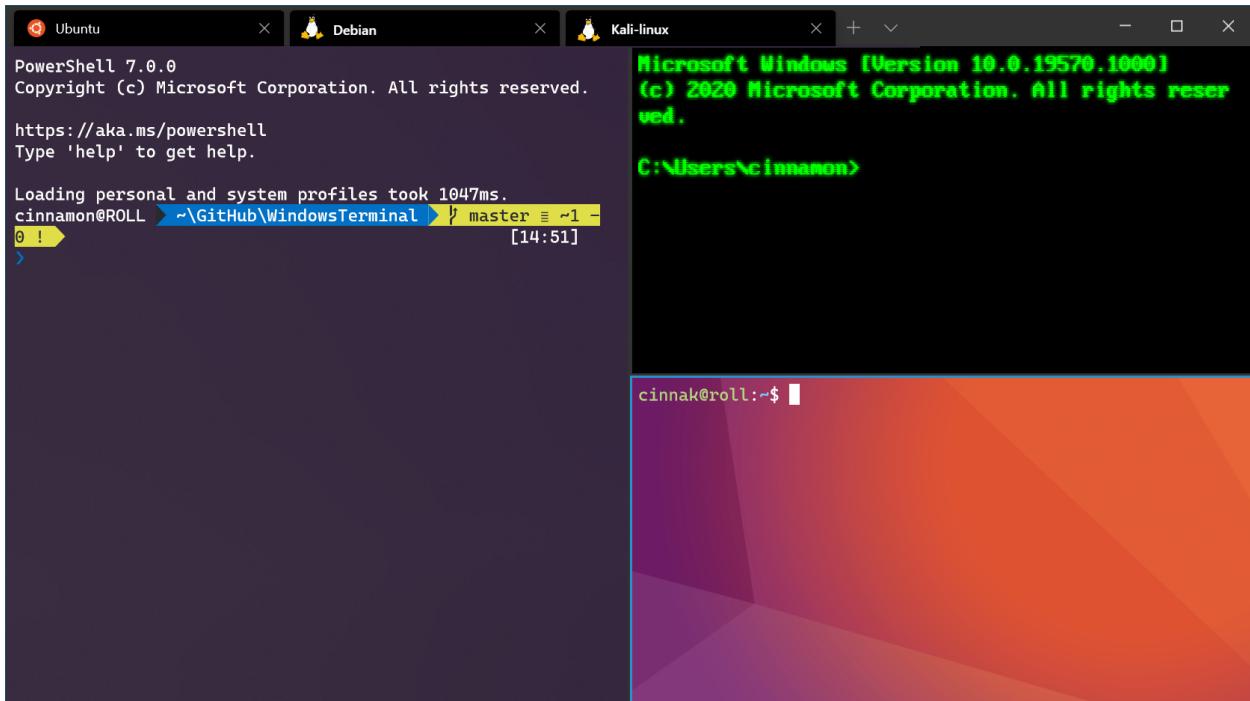
Si usa Windows Server, o tiene problemas para ejecutar el comando anterior, puede encontrar instrucciones de instalación alternativas en la página de la documentación de [Windows Server](#) para instalar el archivo `.appx` cambiándolo a un archivo zip.

Una vez instalada la distribución, siga las instrucciones para [crear una cuenta de usuario y una contraseña para la nueva distribución de Linux](#).

Instalación de Terminal Windows (opcional)

Terminal Windows permite abrir varias pestañas o varios paneles para mostrar y cambiar rápidamente entre varias distribuciones de Linux u otras líneas de comandos (PowerShell, símbolo del sistema, PowerShell, CLI de Azure, etc.). Puede personalizar completamente el terminal con combinaciones de colores únicas, estilos de fuente, tamaños, imágenes de fondo y métodos abreviados de teclado personalizados. [Más información](#).

Instalación de Terminal Windows.



Guía de instalación de Windows Server

Artículo • 21/03/2023

El Subsistema de Windows para Linux (WSL) está disponible para su instalación en Windows Server 2019 (versión 1709) y versiones posteriores. En esta guía se describen los pasos necesarios para habilitar WSL en tu máquina.

Instalación de WSL en Windows Server 2022

[Windows Server 2022](#) ahora admite una instalación simple de WSL con el comando:

Bash

```
wsl --install
```

Ahora puede instalar todo lo que necesita para ejecutar WSL en Windows Server 2022 si especifica este comando en PowerShell o el símbolo del sistema de Windows de un **administrador** y, a continuación, reinicia la máquina.

Este comando habilitará los componentes opcionales necesarios, descargará el kernel de Linux más reciente, establecerá WSL 2 como predeterminado e instalará una distribución de Linux automáticamente (*Ubuntu de forma predeterminada*).

Consulte los documentos estándar de WSL para más información sobre lo siguiente:

- [Cambio de la distribución predeterminada de Linux instalada.](#)
- [Configuración del nombre de usuario y la contraseña de Linux.](#)
- [Comprobación de la versión de WSL que se está ejecutando](#)
- [Actualización de paquetes.](#)
- [Adición de distribuciones adicionales.](#)
- [Uso de Git con WSL.](#)

Instalación de WSL en versiones anteriores de Windows Server

Para instalar WSL en Windows Server 2019 (versión 1709 y posteriores), puede seguir los pasos de instalación manual que se indican a continuación.

Habilitación del Subsistema de Windows para Linux

Para poder ejecutar distribuciones de Linux en Windows, debes habilitar la característica opcional "Subsistema de Windows para Linux" y reiniciar.

Abra PowerShell **como administrador** y ejecute:

```
PowerShell
```

```
Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-  
Subsystem-Linux
```

Descarga de una distribución de Linux

Consulte la sección [Descarga de distribuciones](#) de la página de instalación manual para obtener instrucciones y vínculos para descargar la distribución de Linux que prefiera.

Extracción e instalación de una distribución de Linux

Ahora que has descargado una distribución de Linux, para extraer su contenido e instalarlo manualmente, sigue estos pasos:

1. Extrae el contenido del paquete `<DistributionName>.appx` mediante PowerShell:

```
PowerShell
```

```
Rename-Item .\Ubuntu.appx .\Ubuntu.zip  
Expand-Archive .\Ubuntu.zip .\Ubuntu
```

2. Una vez descargada la distribución, vaya a la carpeta que contiene la descarga y ejecute el siguiente comando en ese directorio, donde `app-name` es el nombre del archivo .appx de la distribución de Linux.

```
Powershell
```

```
Add-AppxPackage .\app_name.appx
```

⊗ Precaución

Error 0x8007007e en la instalación: Si recibes este error, significa que tu sistema no es compatible con WSL. Asegúrate de que estás ejecutando la compilación 16215 de Windows o una posterior. **Comprueba el número de compilación.** Además, **confirma que WSL esté habilitado** y que el equipo se haya reiniciado después de haber habilitado la característica.

3. Agregue la ruta de acceso de la distribución de Linux a la variable de entorno PATH de Windows (`C:\Users\Administrator\Ubuntu` en este ejemplo) mediante PowerShell:

PowerShell

```
$userenv = [System.Environment]::GetEnvironmentVariable("Path", "User")
[System.Environment]::SetEnvironmentVariable("PATH", $userenv +
";C:\Users\Administrator\Ubuntu", "User")
```

Ahora puedes escribir `<DistributionName>.exe` para iniciar la distribución desde cualquier ruta de acceso. Por ejemplo: `ubuntu.exe`.

Una vez completada la instalación, puede [crear una cuenta de usuario y una contraseña para la nueva distribución de Linux](#).

Configurar un entorno de desarrollo de WSL

Artículo • 20/11/2023

Una guía paso a paso sobre los procedimientos recomendados para configurar un entorno de desarrollo de WSL. Obtenga información sobre cómo ejecutar el comando para instalar el shell de Bash predeterminado que utiliza Ubuntu o puede configurarse para instalar otras distribuciones de Linux, utilizar comandos de WSL básicos, configurar Visual Studio Code o Visual Studio, Git, el Administrador de credenciales de Windows, bases de datos como MongoDB, Postgres o MySQL, configurar la aceleración de GPU, ejecutar aplicaciones de GUI y mucho más.

Introducción

Subsistema de Windows para Linux viene con el sistema operativo Windows, pero debe habilitarlo e instalar una distribución de Linux para poder empezar a usarlo.

Para usar el comando simplificado --install, debe ejecutar una compilación reciente de Windows (compilación 20262+). Para comprobar la versión y el número de compilación, seleccione la **tecla del logotipo de Windows + R**, escriba **winver** y seleccione **Aceptar**. Puede actualizar mediante el [menú Configuración](#) o el [Asistente de actualización de Windows ↗](#).

Si prefiere instalar una distribución de Linux distinta de Ubuntu o prefiere completar estos pasos manualmente, consulte la [página de instalación de WSL](#) para obtener más detalles.

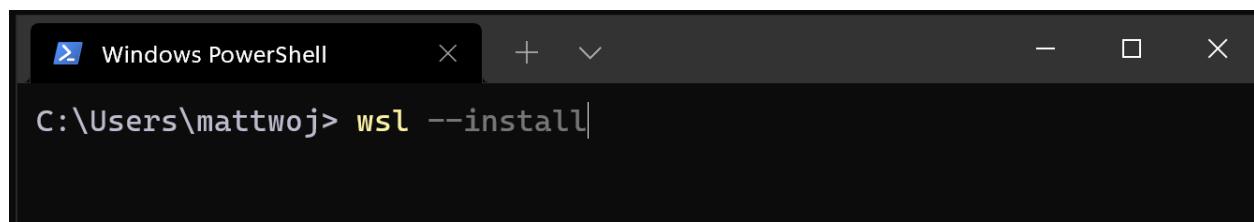
Abra PowerShell (o el símbolo del sistema de Windows) y escriba:

```
PowerShell  
wsl --install
```

El comando --install realiza las acciones siguientes:

- Habilita los componentes opcionales de WSL y Plataforma de máquina virtual.
- Descarga e instala el kernel de Linux más reciente.
- Establece WSL 2 como valor predeterminado.
- Descarga e instala una distribución de Ubuntu Linux (es posible que sea necesario reiniciar)

Tendrá que reiniciar la máquina durante este proceso de instalación.

A screenshot of a Windows PowerShell window titled "Windows PowerShell". The command "wsl --install" is being typed into the command line. The window has standard minimize, maximize, and close buttons at the top right.

Consulte el artículo de [solución de problemas de instalación](#) si tiene algún problema.

Configuración del nombre de usuario y la contraseña de Linux

Una vez completado el proceso de instalación de la distribución de Linux con WSL, abra la distribución (Ubuntu de forma predeterminada) mediante el menú Inicio. Se le pedirá que cree un **nombre de usuario** y una **contraseña** para la distribución de Linux.

- El **nombre de usuario** y la **contraseña** son específicos de cada distribución de Linux individual que instala y no tienen relación con su nombre de usuario de Windows.
- Tenga en cuenta que mientras escribe la **contraseña**, no aparecerá nada en la pantalla. Esto se denomina escritura ciega. No verá lo que está escribiendo, esto es completamente normal.
- Cuando haya creado el **nombre de usuario** y la **contraseña**, la cuenta será el usuario predeterminado de la distribución e iniciará sesión automáticamente al inicio.
- Recuerda que esta cuenta se considerará el administrador de Linux y tendrá la capacidad de ejecutar comandos administrativos `sudo` (es decir, de superusuario).
- Cada distribución de Linux que se ejecuta en el WSL tiene sus propias cuentas de usuario y contraseñas de Linux. Tendrás que configurar una cuenta de usuario de Linux cada vez que reinstales, restablezcas o agregues una distribución.

ⓘ Nota

Las distribuciones de Linux instaladas con WSL son una instalación por usuario y no se pueden compartir con otras cuentas de usuario de Windows. ¿Se produce un error de nombre de usuario? [StackExchange: ¿Qué caracteres debo usar o no en los nombres de usuario en Linux?](#)

```
Ubuntu
Installing, this may take a few minutes...
Installation successful!
Please create a default UNIX user account. The username does not need to match your Windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username:
```

Para cambiar o restablecer la contraseña, abra la distribución de Linux y escriba el comando: `passwd`. Tendrás que escribir la contraseña actual, la contraseña nueva y, a continuación, confirmarla.

Si olvidaste la contraseña de la distribución de Linux:

1. Abre PowerShell y escribe la raíz de la distribución de WSL predeterminada mediante el comando: `wsl -u root`.

Si necesitas actualizar la contraseña olvidada de una distribución que no es la predeterminada, usa el comando `wsl -d Debian -u root` (recuerda que debes reemplazar `Debian` por el nombre de la distribución de destino).
2. Una vez abierta la distribución de WSL en el nivel raíz en PowerShell, puede usar este comando para actualizar la contraseña: `passwd <username>` donde `<username>` es el nombre de usuario de la cuenta en la distribución cuya contraseña ha olvidado.
3. Tendrás que escribir una contraseña UNIX nueva y confirmarla. Cuando vea que la contraseña se ha actualizado correctamente, cierra WSL en PowerShell mediante el comando: `exit`.

Actualización de paquetes

Se recomienda actualizar regularmente los paquetes utilizando el administrador de paquetes preferido para la distribución. Para Ubuntu o Debian, use el comando:

Bash

```
sudo apt update && sudo apt upgrade
```

Windows no actualiza automáticamente las distribuciones de Linux. Se trata de una tarea que la mayoría de los usuarios de Linux prefieren controlar por sí mismos.

Adición de distribuciones adicionales

Para agregar distribuciones adicionales de Linux, puede instalarlas a través de Microsoft Store [🔗](#), mediante el comando `--import` o mediante la instalación de prueba de su propia distribución personalizada. También puede configurar imágenes de WSL personalizadas para su distribución en toda la empresa.

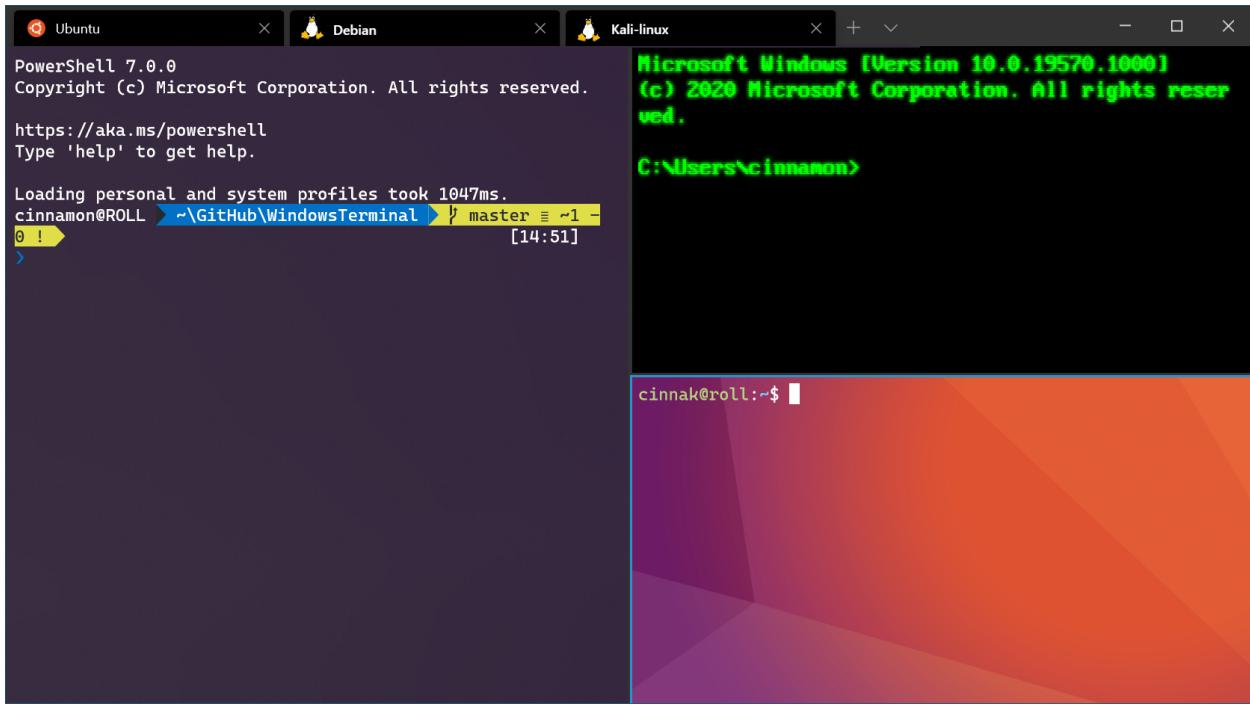
Configurar Terminal Windows

Terminal Windows puede ejecutar cualquier aplicación con una interfaz de línea de comandos. Entre sus características principales se incluyen varias pestañas, paneles, compatibilidad con caracteres Unicode y UTF-8, un motor de representación de texto acelerado para GPU y la capacidad de crear sus propios temas y personalizar texto, colores, fondos y métodos abreviados.

Cada vez que se instala una nueva distribución de Linux de WSL, se creará una nueva instancia para ella dentro de Terminal Windows que se puede personalizar con sus preferencias.

Se recomienda usar WSL con Terminal Windows, especialmente si tiene previsto trabajar con varias líneas de comandos. Consulte la documentación de Terminal Windows para obtener ayuda sobre cómo configurarlo y personalizar sus preferencias, como:

- [Instalar Terminal Windows o Terminal Windows \(versión preliminar\)](#) desde Microsoft Store
- [Usar la paleta de comandos](#)
- Configurar [acciones personalizadas](#) como métodos abreviados de teclado para que el terminal se adapte de forma natural a sus preferencias
- Configuración del [perfil de inicio predeterminado](#)
- Personalice la apariencia: [tema](#), [combinaciones de colores](#), [nombre e directorio inicial](#), [imagen de fondo](#), etc.
- Aprenda a usar [argumentos de línea de comandos](#), como abrir un terminal con varias líneas de comandos divididas en paneles o pestañas de ventana
- Más información sobre la [característica de búsqueda](#)
- Buscar [sugerencias y trucos](#), como cambiar el nombre o colorear una pestaña, usar interacciones del mouse o habilitar el "modo Terremoto"
- Buscar tutoriales sobre cómo configurar [un símbolo del sistema personalizado](#), [perfiles SSH](#) o [títulos de pestaña](#)
- Buscar una [galería de terminales personalizada](#) y una guía de solución de problemas

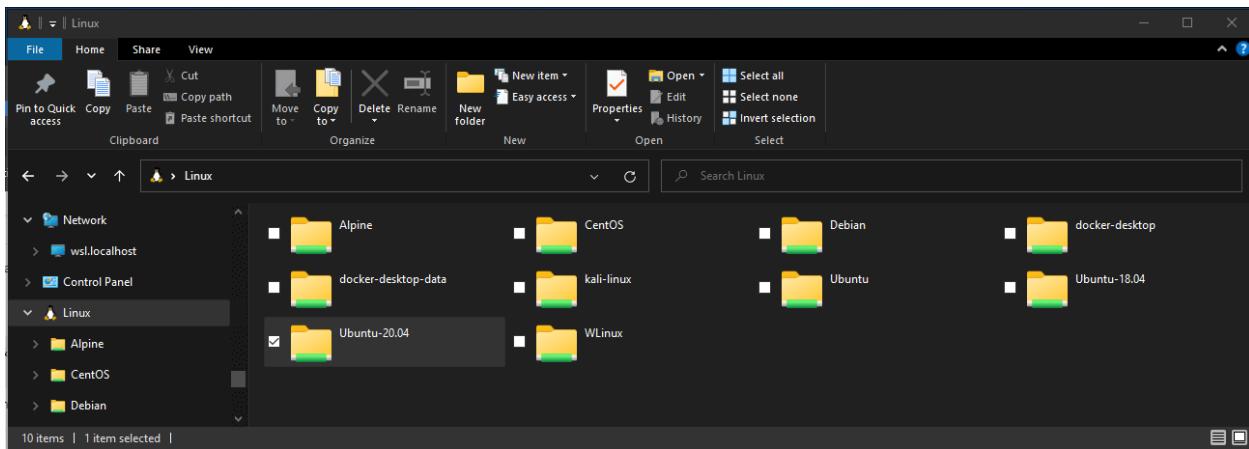


File Storage

- Para abrir su proyecto de WSL en el Explorador de archivos de Windows, introduzca: `explorer.exe .`
Asegúrese de agregar el punto al final del comando para abrir el directorio actual.
- Almacene los archivos del proyecto en el mismo sistema operativo que las herramientas que planea usar.
Para lograr la máxima velocidad de rendimiento, almacene los archivos en el sistema de archivos de WSL si está trabajando en ellos con herramientas Linux en una línea de comandos de Linux (Ubuntu, OpenSUSE, etc.). Si trabaja en una línea de comandos de Windows (PowerShell, símbolo del sistema) con herramientas Windows, almacene los archivos en el sistema de archivos de Windows. Se puede acceder a los archivos en los sistemas operativos, pero puede ralentizar significativamente el rendimiento.

Por ejemplo, al almacenar los archivos de proyecto de WSL, haz lo siguiente:

- Usa el directorio raíz del sistema de archivos de Linux: `\wsl$<DistroName>\home\<UserName>\Project`
- No use el directorio raíz del sistema de archivos de Windows: `C:\Users\<UserName>\Project` o `/mnt/c/Users/<UserName>/Project$`.



Configurar el editor de código favorito

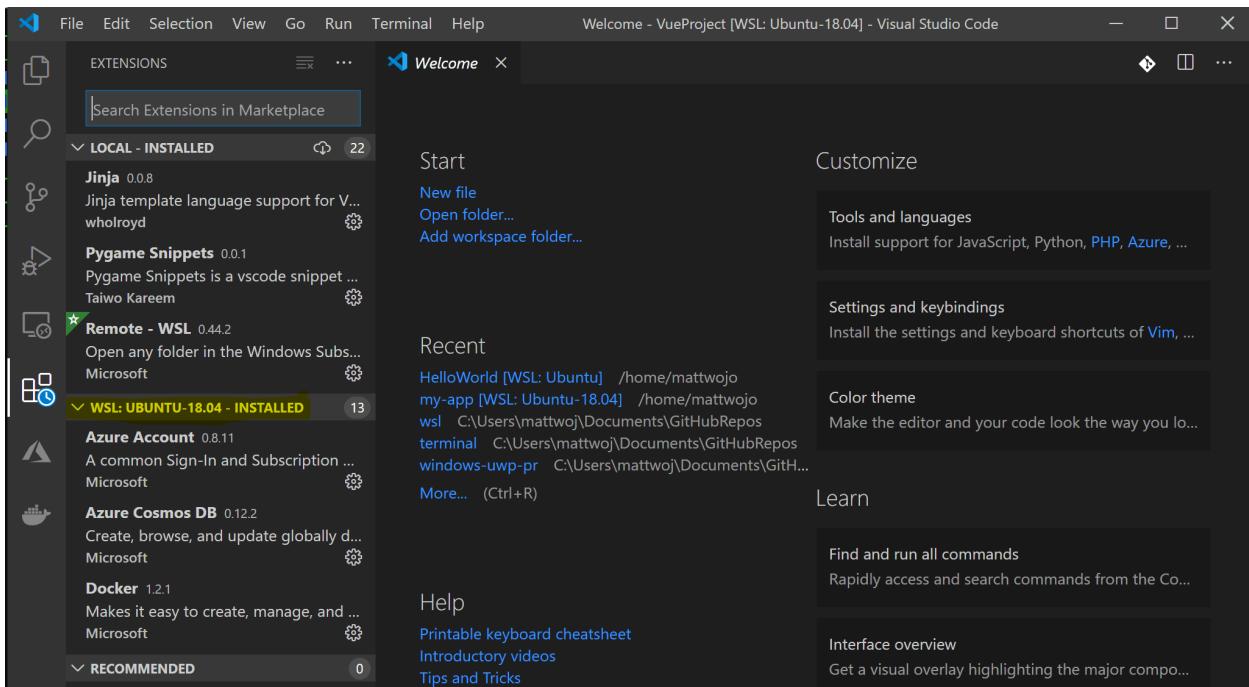
Se recomienda usar Visual Studio Code o Visual Studio, ya que admiten directamente el desarrollo remoto y la depuración con WSL. Visual Studio Code le permite utilizar el WSL como un entorno de desarrollo completo. Visual Studio ofrece compatibilidad nativa con WSL para el desarrollo multiplataforma de C++.

Usar Visual Studio Code

Siga esta guía paso a paso para [Empezar a usar Visual Studio Code con WSL](#), que incluye la instalación del [paquete de extensiones de desarrollo remoto](#). Esta extensión permite ejecutar WSL, SSH o un contenedor de desarrollo para editar y depurar con el conjunto completo de características de Visual Studio Code. Cambiar rápidamente entre diferentes entornos de desarrollo independientes y realizar actualizaciones sin preocuparse por el impacto en su máquina local.

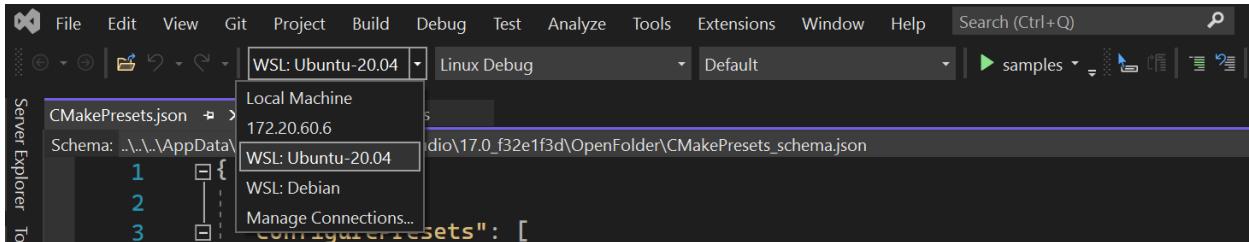
Una vez instalado y configurado VS Code, puede abrir el proyecto WSL con un servidor remoto de VS Code escribiendo: `code .`

Asegúrese de agregar el punto al final del comando para abrir el directorio actual.



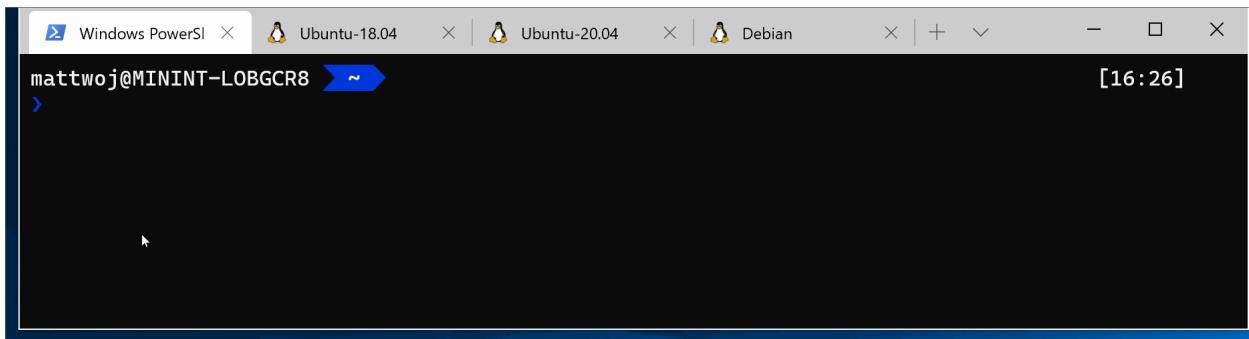
Usar Visual Studio

Siga esta guía paso a paso para [Empezar a usar Visual Studio con WSL para el desarrollo multiplataforma de C++](#). Visual Studio 2022 permite compilar y depurar proyectos de CMake en Windows, distribuciones de WSL y conexiones SSH desde la misma instancia de Visual Studio.



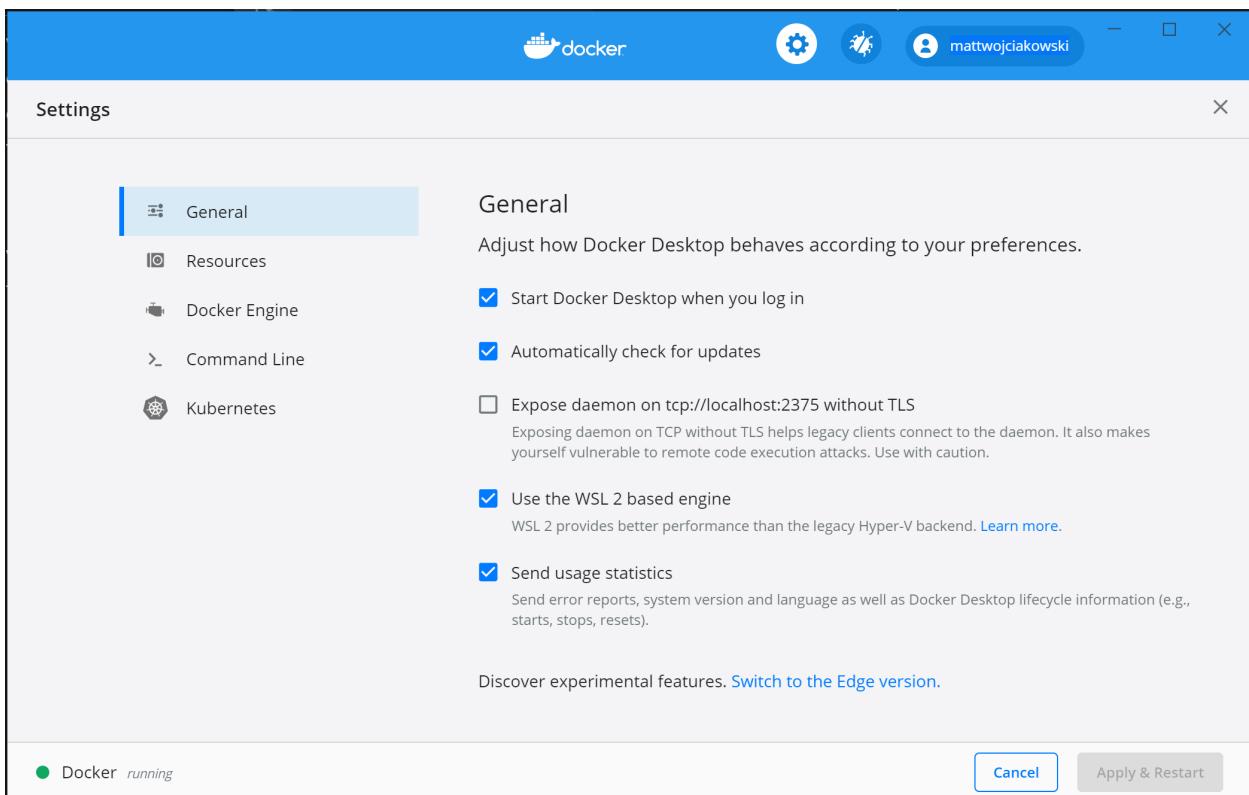
Configurar la administración de versiones con Git

Siga esta guía paso a paso para [Empezar a usar Git en WSL](#) y conectar el proyecto al sistema de control de versión Git, junto con el administrador de credenciales para la autenticación, mediante el uso de archivos de Git Ignore, la comprensión de los finales de línea Git y el uso de los comandos Git integrados en VS Code.



Configuración de contenedores de desarrollo remoto con Docker

Siga esta guía paso a paso para [Empezar a trabajar con contenedores remotos de Docker en WSL 2](#) y conectar el proyecto a un contenedor de desarrollo remoto con Docker Desktop para Windows.



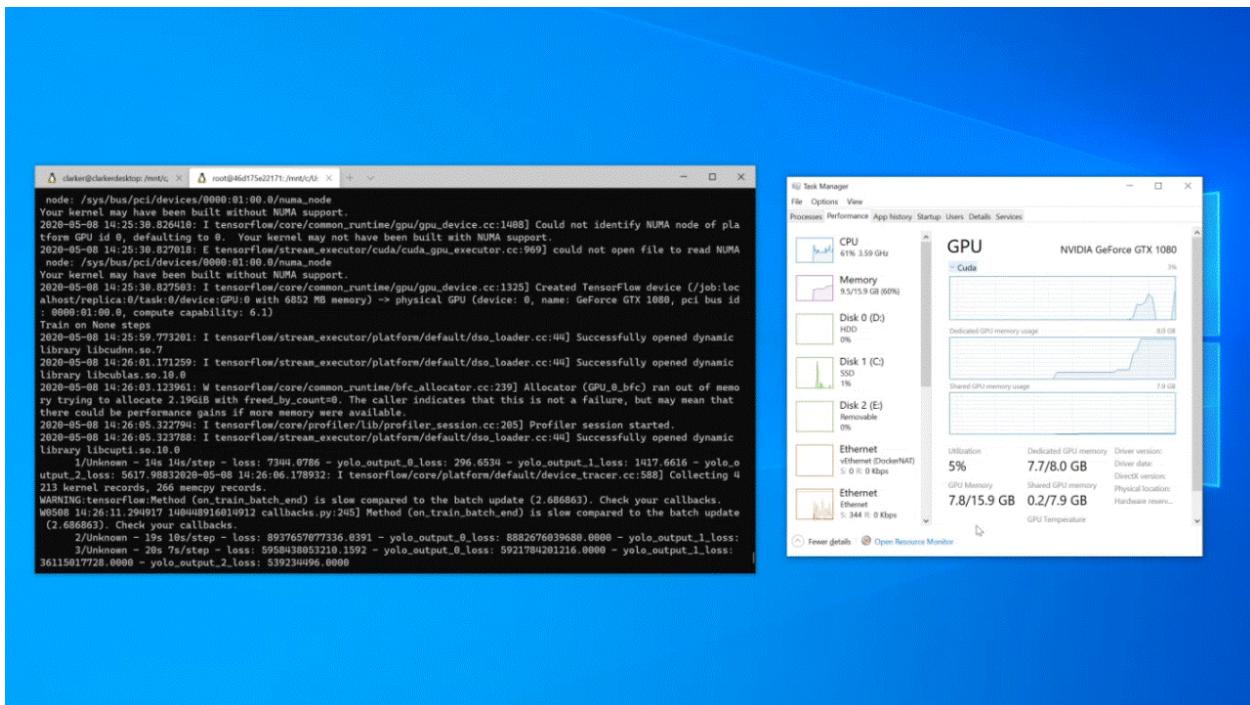
Configuración de una base de datos

Siga esta guía paso a paso para [Empezar a trabajar con bases de datos en WSL](#) y conectar el proyecto a una base de datos en el entorno de WSL. Introducción a MySQL, PostgreSQL, MongoDB, Redis, Microsoft SQL Server o SQLite.

```
Ubuntu
db version v3.6.3
git version: 9586e557d54ef70f9ca4b43c26892cd55257e1a5
OpenSSL version: OpenSSL 1.1.1  11 Sep 2018
allocator: tcmalloc
modules: none
build environment:
  distarch: x86_64
  target_arch: x86_64
mattwojo@MININT-LOBGCR8:~$ sudo service mongodb start
 * Starting database mongodb
[m  OK ]
mattwojo@MININT-LOBGCR8:~$
```

Configurar la aceleración de GPU para un rendimiento más rápido

Siga esta guía paso a paso para configurar el [entrenamiento de aprendizaje automático acelerado por GPU en WSL](#) y aprovechar la GPU (unidad de procesamiento gráfico) del equipo para acelerar las cargas de trabajo pesadas de rendimiento.



Comandos básicos de WSL

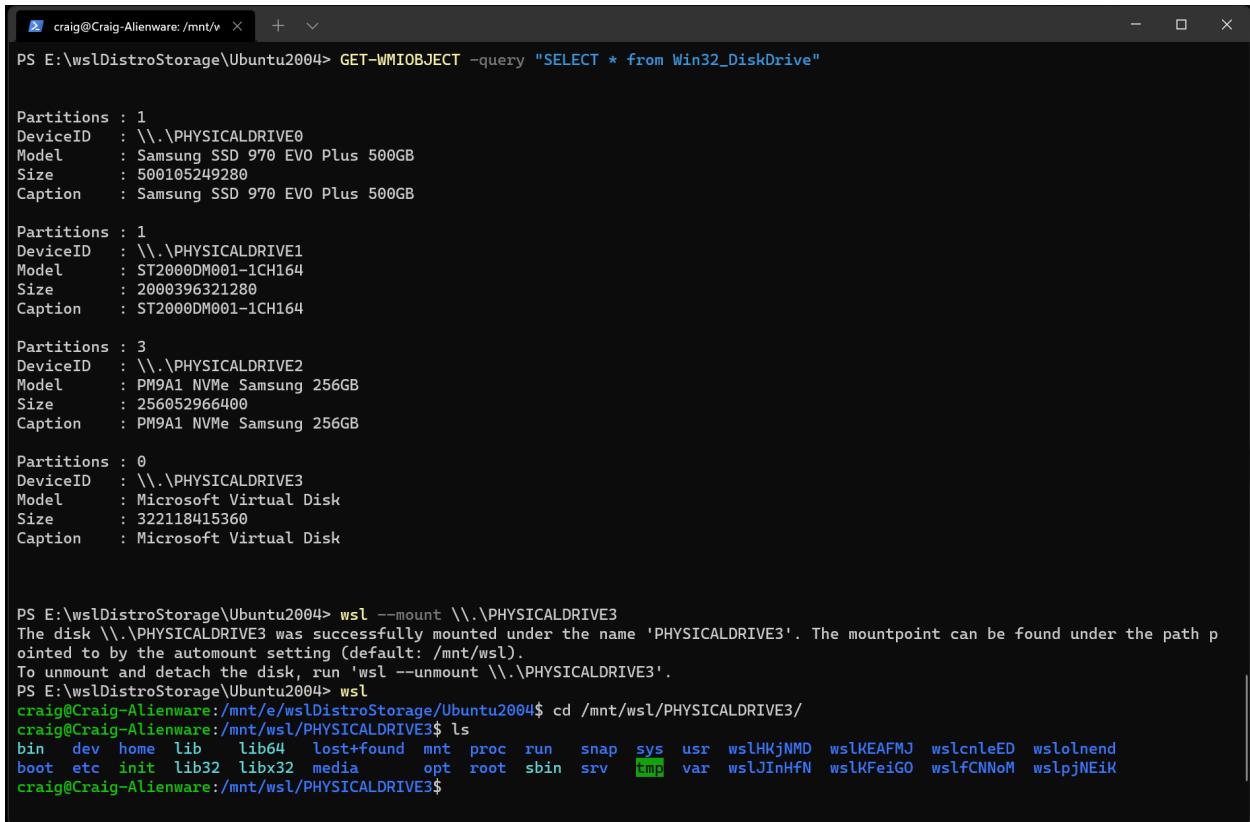
Las distribuciones de Linux que se instalan a través de WSL se administran mejor mediante PowerShell o el símbolo del sistema de Windows (CMD). Consulte la [guía de referencia de comandos de WSL](#) para obtener una lista de comandos básicos con los que familiarizarse al usar el WSL.

Además, muchos comandos son interoperables entre Windows y Linux. Estos son algunos ejemplos:

- **Ejecutar herramientas de Linux a partir de una línea de comandos de Windows:** abra PowerShell y muestre el contenido del directorio `C:\temp>` mediante el comando de Linux `ls -la` escribiendo: `wsl ls -la`
- **Combinar comandos de Linux y Windows:** en este ejemplo, el comando `ls -la` de Linux se usa para enumerar archivos en el directorio y, a continuación, el comando `findstr` de PowerShell se usa para filtrar los resultados de las palabras que contienen "git": `wsl ls -la | findstr "git"`. Esto también se podría hacer mezclando el comando de Windows `dir` con el comando de Linux `grep`: `dir | wsl grep git`.
- **Ejecutar una herramienta de Windows directamente desde la línea de comandos del WSL:** `<tool-name>.exe` por ejemplo, para abrir el archivo `.bashrc` (el script de shell que se ejecuta cada vez que se inicia la línea de comandos de Linux), escriba: `notepad.exe .bashrc`
- **Ejecutar la herramienta Windows ipconfig.exe con la herramienta Grep de Linux:** en Bash, escriba el comando `ipconfig.exe | grep IPv4 | cut -d: -f2` o en PowerShell escriba `ipconfig.exe | wsl grep IPv4 | wsl cut -d: -f2`. Este ejemplo muestra el uso de la herramienta ipconfig en el sistema de archivos de Windows para mostrar los valores actuales de configuración de red TCP/IP y su posterior filtrado a solo el resultado IPv4 con grep, una herramienta de Linux.

Montar una unidad externa o USB

Siga esta guía paso a paso para [Empezar a montar un disco Linux en el WSL 2](#).



```
craig@craig-Alienware: /mnt/v < + >
PS E:\wslDistroStorage\Ubuntu2004> GET-WMIOBJECT -query "SELECT * from Win32_DiskDrive"
Partitions : 1
DeviceID   : \\.\PHYSICALDRIVE0
Model      : Samsung SSD 970 EVO Plus 500GB
Size       : 500105249280
Caption    : Samsung SSD 970 EVO Plus 500GB

Partitions : 1
DeviceID   : \\.\PHYSICALDRIVE1
Model      : ST2000DM001-1CH164
Size       : 2000396321280
Caption    : ST2000DM001-1CH164

Partitions : 3
DeviceID   : \\.\PHYSICALDRIVE2
Model      : PM9A1 NVMe Samsung 256GB
Size       : 256052966400
Caption    : PM9A1 NVMe Samsung 256GB

Partitions : 0
DeviceID   : \\.\PHYSICALDRIVE3
Model      : Microsoft Virtual Disk
Size       : 322118415360
Caption    : Microsoft Virtual Disk

PS E:\wslDistroStorage\Ubuntu2004> wsl --mount \\.\PHYSICALDRIVE3
The disk \\.\PHYSICALDRIVE3 was successfully mounted under the name 'PHYSICALDRIVE3'. The mountpoint can be found under the path pointed to by the automount setting (default: /mnt/wsl).
To unmount and detach the disk, run 'wsl --unmount \\.\PHYSICALDRIVE3'.
PS E:\wslDistroStorage\Ubuntu2004> wsl
craig@craig-Alienware:/mnt/e/wslDistroStorage/Ubuntu2004$ cd /mnt/wsl/PHYSICALDRIVE3/
craig@craig-Alienware:/mnt/wsl/PHYSICALDRIVE3$ ls
bin  dev  home  lib  lib64  lost+found  mnt  proc  run  snap  sys  usr  wslHKjNMD  wslKEAFMJ  wslcnleED  wsloInend
boot  etc  init  lib32  libx32  media  opt  root  sbin  srv  tmp  var  wslJInHFN  wslKFeiGO  wslfCNNoM  wslpjNEiK
craig@craig-Alienware:/mnt/wsl/PHYSICALDRIVE3$
```

Ejecución de aplicaciones de GUI de Linux

Siga este tutorial para aprender a configurar y [ejecutar aplicaciones de GUI de Linux en WSL](#).

Recursos adicionales

- [Configurar el entorno de desarrollo en Windows](#): obtenga más información sobre cómo configurar el entorno de desarrollo para su lenguaje o marco preferido, como React, Python, NodeJS, Vue, etc.
- [Solución de problemas](#): busque problemas comunes, dónde notificar errores, dónde solicitar nuevas características y cómo contribuir a los documentos.
- [Preguntas más frecuentes](#): busque una lista de las preguntas más frecuentes.
- [Notas de la versión](#): revise las notas de la versión de WSL para obtener un historial de actualizaciones de compilación anteriores. También puede encontrar las [notas de la versión del kernel de Linux de WSL](#).

Introducción a Visual Studio Code con el Subsistema de Windows para Linux

Artículo • 07/10/2023

Visual Studio Code con la extensión WSL permite usar WSL como entorno de desarrollo a tiempo completo directamente desde VS Code. Puede:

- desarrollar en un entorno basado en Linux
- usar de cadenas de herramientas y utilidades específicas de Linux
- ejecutar y depurar las aplicaciones basadas en Linux desde la comodidad de Windows y, a la vez, mantener el acceso a herramientas de productividad como Outlook y Office
- usar el terminal integrado de VS Code para ejecutar la distribución de Linux que prefiera
- aprovechar las características de VS Code, como la [el autocompletado de código de IntelliSense](#), [la detección de errores](#), [la compatibilidad con la depuración](#), [los fragmentos de código](#) y [las pruebas unitarias](#)
- administrar fácilmente el control de versiones con la [compatibilidad con Git](#) integrada de VS Code
- ejecutar comandos y extensiones de VS Code directamente en los proyectos de WSL
- editar archivos en el sistema de archivos de Linux o montado en Windows (por ejemplo, /mnt/c) sin preocuparse por problemas de ruta de acceso, compatibilidad binaria u otros desafíos entre sistemas operativos

Instalación de VS Code y la extensión WSL

- Visite la [página de instalación de VS Code](#) y seleccione el instalador de 32 o 64 bits. Instale Visual Studio Code en Windows (no en el sistema de archivos WSL).
- Cuando se le pida que **seleccione tareas adicionales** durante la instalación, asegúrese de activar la opción **Agregar a PATH** para que pueda abrir fácilmente una carpeta en WSL mediante el comando de código.
- Instale el [paquete de extensiones de desarrollo remoto](#). Este paquete de extensiones incluye la extensión WSL, además de las extensiones Remote - SSH y Dev Containers, lo que le permite abrir cualquier carpeta en un contenedor, en un equipo remoto o en WSL.

Importante

Para instalar la extensión WSL, necesitará la [versión 1.35 de mayo](#) o posterior de VS Code. No se recomienda usar WSL en VS Code sin la extensión WSL, ya que perderá la compatibilidad con autocompletar, depuración, linting, etc. Hecho divertido: esta extensión WSL se instala en \$HOME/.vscode/extensions (escriba el comando `ls $HOME\.vscode\extensions\` en PowerShell).

Actualización de la distribución de Linux

Algunas distribuciones de WSL en Linux carecen de bibliotecas que el servidor de VS Code requiere para iniciarse. Puede agregar bibliotecas adicionales a la distribución de Linux mediante su administrador de paquetes.

Por ejemplo, para actualizar Debian o Ubuntu, use:

Bash

```
sudo apt-get update
```

Para agregar wget (para recuperar contenido de servidores web) y certificados de ca (para permitir que las aplicaciones basadas en SSL comprueben la autenticidad de las conexiones SSL), escriba:

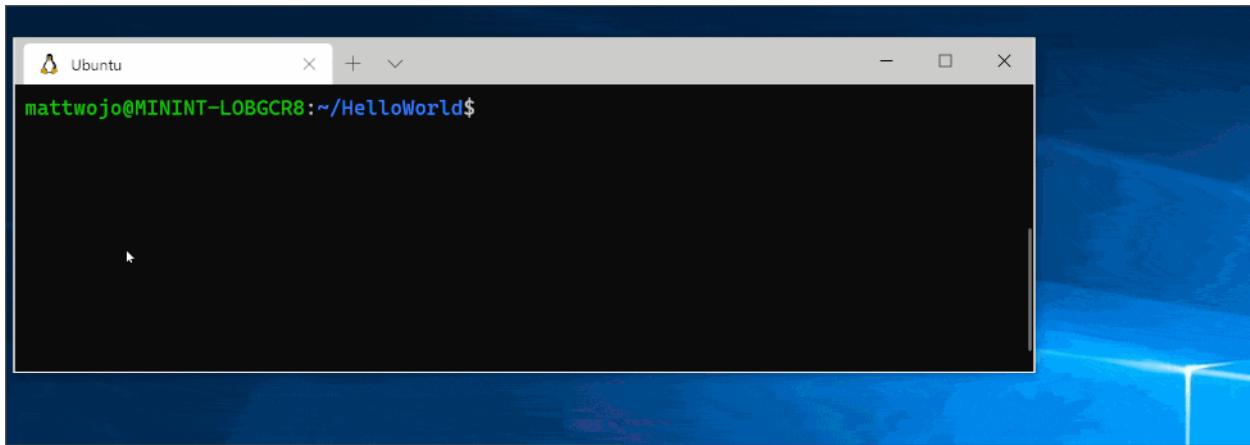
Bash

```
sudo apt-get install wget ca-certificates
```

Abrir un proyecto de WSL en Visual Studio Code

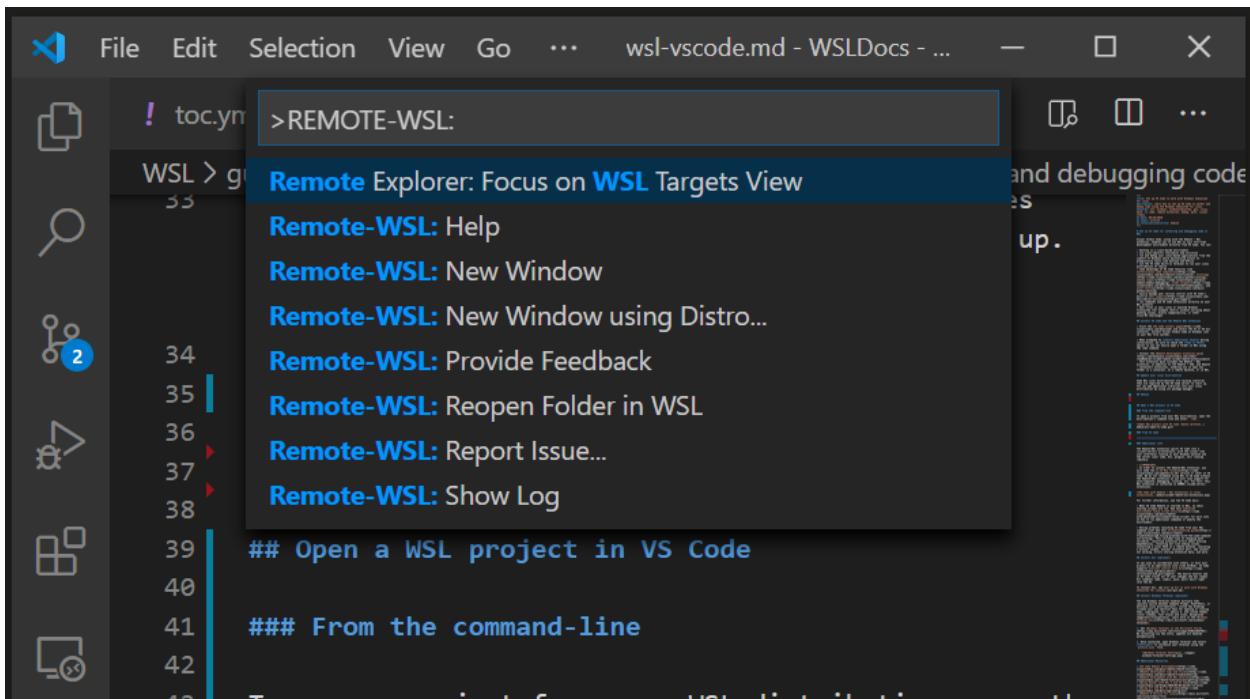
Desde la línea de comandos

Para abrir un proyecto desde la distribución de WSL, abra la línea de comandos de la distribución y escriba: `code .`



Desde VS Code

También puede acceder a más opciones de WSL en VS Code mediante el acceso directo: **CTRL+SHIFT+P** en VS Code para abrir la paleta de comandos. Si después escribe **WSL**, verá una lista de las opciones disponibles, lo que le permitirá volver a abrir la carpeta en una sesión de WSL, especificar en qué distribución desea abrirla y mucho más.



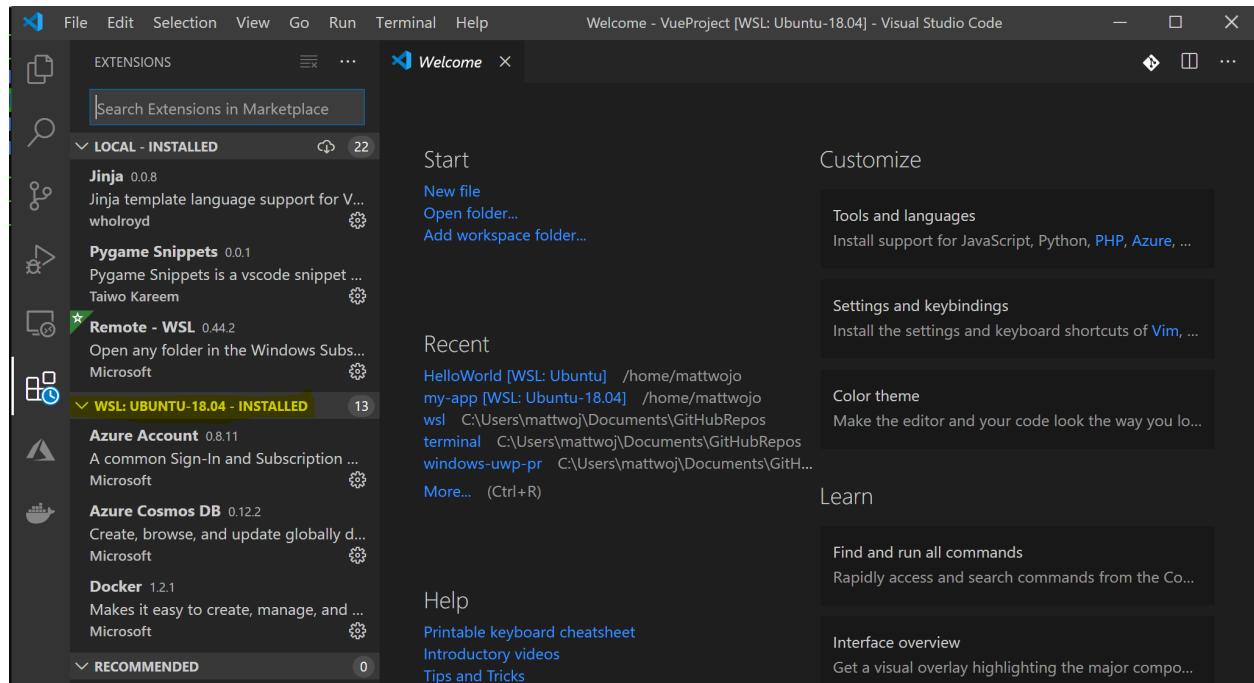
Extensiones en WSL en VS Code

La extensión WSL divide VS Code en una arquitectura "cliente-servidor", con el cliente (la interfaz de usuario) que se ejecuta en el equipo Windows y el servidor (código, Git, complementos, etc.) que se ejecuta de forma remota en la distribución de WSL.

Cuando se ejecuta la extensión WSL, al seleccionar la pestaña "Extensiones", se mostrará una lista de extensiones divididas entre el equipo local y la distribución de WSL.

La instalación de una extensión local, como un [tema](#), solo debe realizarse una vez.

Algunas extensiones, como la [extensión de Python](#) o cualquier otra que controle tareas como la detección de errores o la depuración, deben instalarse por separado en cada distribución de WSL. VS Code mostrará un ícono de advertencia junto con un botón verde "Instalar en WSL", si tiene una extensión instalada localmente que no está instalada en la distribución de WSL.



Para obtener más información, consulte la documentación de VS Code:

- Cuando VS Code se inicia en WSL, no se ejecutan scripts de inicio de shell. Consulte este [artículo sobre el script de configuración de entorno avanzado](#) para obtener más información sobre cómo ejecutar comandos adicionales o modificar el entorno.
- ¿Tiene problemas para iniciar VS Code desde la línea de comandos de WSL? Esta [guía de solución de problemas](#) incluye sugerencias sobre cómo cambiar variables de ruta de acceso, resolver errores de extensión relacionados con dependencias que faltan, resolver problemas de finalización de línea de Git, instalar un VSIX local en un equipo remoto, iniciar una ventana del explorador, puerto localhost bloqueado, sockets web que no funcionan, errores al almacenar datos de extensión, etc.

Instalar GIT (opcional)

Si planeas colaborar con otras personas u hospedar el proyecto en un sitio de código abierto (como GitHub), VS Code admite el [control de versiones con GIT](#). La pestaña

Control de código fuente de VS Code realiza un seguimiento de todos los cambios y tiene comandos GIT comunes (agregar, confirmar, enviar cambios e incorporar cambios) integrados directamente en la interfaz de usuario.

Para instalar Git, consulte [Configuración de Git para que funcione con el Subsistema de Windows para Linux](#).

Instalación de Terminal Windows (opcional)

El nuevo Terminal Windows habilita varias pestañas (cambia rápidamente entre el símbolo del sistema, PowerShell o varias distribuciones de Linux), enlaces de teclado personalizados (crea tus propias teclas de método abreviado para abrir o cerrar pestañas, copiar y pegar, etc.), emojis ☺ y temas personalizados (esquemas de colores, estilos y tamaños de fuente, imagen de fondo/desenfoque/transparencia). Obtenga más información en la [documentación de Terminal Windows](#).

1. Obtener [Terminal Windows en microsoft Store](#): Al instalar a través de la tienda, las actualizaciones se controlan automáticamente.
2. Una vez instalado, abre Terminal Windows y selecciona **Configuración** para personalizar el terminal con el archivo `profile.json`.

Recursos adicionales

- [Documentación de WSL en VS Code](#)
- [Tutorial de WSL en VS Code](#)
- [Sugerencias y trucos de desarrollo remoto](#)
- [Uso de Docker con WSL 2 y VS Code](#)
- [Uso de C++ y WSL en VS Code](#)
- [Servicio R remoto para Linux](#)

Algunas de las extensiones adicionales que puedes considerar son las siguientes:

- [Keymaps from other editors](#): estas extensiones pueden ayudarte a sentirte como en casa con tu entorno en caso de que realices la transición desde otro editor de texto (como Atom, Sublime, Vim, eMacs, Notepad++, etc.).
- [Settings Sync](#): te permite sincronizar la configuración de VS Code entre diferentes instalaciones mediante GitHub. Si trabajas en diferentes máquinas, te ayuda a mantener el entorno coherente entre ellas.
- [Debugger for Chrome](#): una vez que haya terminado de desarrollar en el lado servidor con Linux, deberá desarrollar y probar el lado cliente. Esta extensión

integra el editor de VS Code con el servicio de depuración del explorador Chrome, lo que permite que las operaciones sean un poco más eficaces.

Introducción al uso de Git en Subsistema de Windows para Linux

Artículo • 07/10/2023

Git es el sistema de control de versiones más utilizado. Con Git, puede realizar un seguimiento de los cambios realizados en los archivos, por lo que tiene un registro de lo que se ha hecho y tiene la capacidad de revertir a versiones anteriores de los archivos si es necesario. Git también facilita la colaboración, lo que permite que los cambios de varias personas se combinen en un origen.

Git se puede instalar en Windows Y en WSL

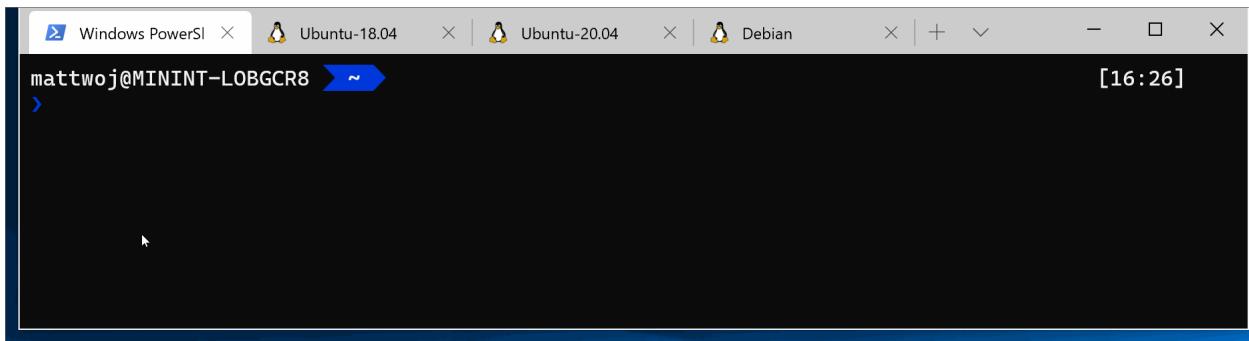
Una consideración importante: al habilitar WSL e instalar una distribución de Linux, va a instalar un nuevo sistema de archivos, separado de Windows unidad NTFS C:\ en la máquina. En Linux, las unidades no tienen letras. Tienen puntos de montaje. La raíz del sistema de archivos / es el punto de montaje de la partición raíz, o carpeta, en el caso de WSL. No todo bajo / es la misma unidad. Por ejemplo, en mi portátil, he instalado dos versiones de Ubuntu (20.04 y 18.04), así como Debian. Si abro esas distribuciones, selecciono el directorio raíz con el comando cd ~, y después escribo el comando explorer.exe ., se abrirá el Explorador de archivos de Windows y me mostrará la ruta de acceso al directorio de esa distribución.

Distribución Linux	Ruta de acceso de Windows para acceder a la carpeta principal
Ubuntu 20.04	\wsl\$\Ubuntu-20.04\home\username
Ubuntu 18.04	\wsl\$\Ubuntu-18.04\home\username
Debian	\wsl\$\Debian\home\username
Windows PowerShell	C:\Users\username

💡 Sugerencia

Si busca acceder al directorio de archivos de Windows desde la línea de comandos de distribución de WSL, en lugar de C:\Users\username, se accedería al directorio mediante /mnt/c/Users/username, porque la distribución de Linux ve el sistema de archivos de Windows como una unidad montada.

Deberá instalar Git en cada sistema de archivos con el que quiera usarlo.



Installing Git

Git ya está instalado con la mayoría de las distribuciones de Subsistema de Windows para Linux, pero es posible que quiera actualizar a la versión más reciente. También deberá configurar el archivo de configuración de Git.

Para instalar Git, consulte el sitio [Descarga de Git para Linux](#). Cada distribución de Linux tiene su propio administrador de paquetes y comando de instalación.

Para la versión estable más reciente de Git en Ubuntu/Debian, escriba el comando:

```
Bash  
sudo apt-get install git
```

ⓘ Nota

También puede instalar [Git para Windows](#) si aún no lo ha hecho.

Configuración del archivo de configuración de Git

Para configurar el archivo de configuración de Git, abra una línea de comandos para la distribución en la que está trabajando y establezca el nombre con este comando (reemplazando "Su nombre" por su nombre de usuario preferido):

```
Bash  
git config --global user.name "Your Name"
```

Establezca el correo electrónico con este comando (reemplazando "youremail@domain.com" por el correo electrónico que prefiera):

Bash

```
git config --global user.email "youremail@domain.com"
```

💡 Sugerencia

Si aún no tiene una cuenta de GitHub, puede [registrarse para obtener una en GitHub](#). Si nunca has trabajado con GIT, las [guías de GitHub](#) pueden resultarte de ayuda para empezar. Si tiene que editar la configuración de Git, puede hacerlo con un editor de texto integrado como nano: `nano ~/.gitconfig`.

Se recomienda [proteger su cuenta con autenticación en dos fases \(2FA\)](#).

Configuración del Administrador de credenciales de Git

El [Administrador de credenciales de Git \(GCM\)](#) es un asistente de credenciales de Git seguro, creado sobre [.NET](#) que puede usarse tanto con WSL1 como con WSL2. Habilita la compatibilidad con la autenticación multifactor para repositorios de GitHub, [Azure DevOps](#), Azure DevOps Server y Bitbucket.

GCM se integra en el flujo de autenticación para servicios como GitHub y, una vez que se ha autenticado en el proveedor de hospedaje, solicita un nuevo token de autenticación. A continuación, almacena el token de forma segura en el [Administrador de credenciales de Windows](#). Después de la primera vez, puede usar Git para comunicarse con el proveedor de hospedaje sin necesidad de volver a autenticarse. Solo accederá al token en el Administrador de credenciales de Windows.

Para usar GCM con WSL, debe estar en Windows 10 versión 1903 o posterior. Esta es la primera versión de Windows que incluye la herramienta `wsl.exe` necesaria que GCM usa para interoperar con Git en sus distribuciones WSL.

Se recomienda instalar el [Último Git para Windows](#) para compartir la configuración de & credenciales entre WSL y el host de Windows. El Administrador de credenciales de Git se incluye con Git para Windows y la versión más reciente se incluye en cada nueva versión de Git para Windows. Durante la instalación, se le pedirá que seleccione un asistente de credenciales, con GCM establecido como predeterminado.

Si tiene un motivo para no instalar Git para Windows, puede instalar GCM como una aplicación de Linux directamente en la distribución de WSL, pero tenga en cuenta que

hacerlo significa que GCM se ejecuta como una aplicación de Linux y no puede usar las características de almacenamiento de credenciales o autenticación del sistema operativo Windows host. Consulte el repositorio de GCM para obtener instrucciones sobre cómo [configurar WSL sin Git para Windows](#).

Para configurar GCM para usarlo con una distribución WSL, abra su distribución y escriba este comando:

Si el GIT instalado es >= v2.39.0

Bash

```
git config --global credential.helper "/mnt/c/Program Files/Git/mingw64/bin/git-credential-manager.exe"
```

si el GIT instalado es >= v2.36.1

Bash

```
git config --global credential.helper "/mnt/c/Program Files/Git/mingw64/libexec/git-core/git-credential-manager.exe"
```

si la versión es < v2.36.1, escriba este comando:

Bash

```
git config --global credential.helper "/mnt/c/Program Files/Git/mingw64/bin/git-credential-manager-core.exe"
```

ⓘ Nota

El uso de GCM como asistente de credenciales para una instalación de Git de WSL significa que GCM no respeta cualquier configuración establecida en Git de WSL (de forma predeterminada). Esto se debe a que GCM se ejecuta como una aplicación de Windows y, por lo tanto, usará la instalación de Git para Windows para consultar la configuración. Esto significa que cosas como la configuración del proxy para GCM deben establecerse tanto en Git para Windows como en Git de WSL, ya que se almacenan en archivos diferentes (`%USERPROFILE%/.gitconfig` y `\wsl$\distro\home\$USER\.gitconfig`). Puede configurar WSL para que GCM use la configuración de Git de WSL, pero esto significa que la configuración del proxy será única para la instalación de WSL específica y no se compartirá con otros usuarios o el host de Windows.

Git con SSH

El Administrador de credenciales de Git solo funciona con HTTP(S) remotos. Todavía puede usar Git con SSH:

- [Azure DevOps SSH](#)
- [GitHub SSH ↗](#)
- [Bitbucket SSH ↗](#)

Configuración adicional para Azure

Si tiene previsto trabajar con [Azure Repos ↗](#) o [Azure DevOps ↗](#), se requiere alguna configuración adicional:

Bash

```
git config --global credential.https://dev.azure.com.useHttpPath true
```

Ahora cualquier operación de Git que realice dentro de su distribución WSL usará GCM. Si ya tiene credenciales almacenadas en caché para un host, accederá a estas desde el Administrador de credenciales. Si no es así, recibirá una respuesta de diálogo que le solicitará sus credenciales, aunque esté en una consola Linux.

💡 Sugerencia

Si usa una clave de GPG para la seguridad de firma de código, es posible que tenga que [asociar la clave GPG con el correo electrónico de GitHub ↗](#).

Adición de un archivo Ignore de Git

Se recomienda [agregar un archivo .gitignore ↗](#) a sus proyectos. GitHub ofrece [una colección de plantillas .gitignore útiles ↗](#) con configuraciones de archivos .gitignore recomendadas organizadas según su caso de uso. Por ejemplo, esta es la [plantilla de .gitignore predeterminada de GitHub para un proyecto de Node.js ↗](#).

Si elige [crear un nuevo repositorio usando el sitio web de GitHub ↗](#), hay casillas de verificación disponibles para inicializar su repositorio con un archivo README, un archivo .gitignore configurado para su tipo de proyecto específico y opciones para agregar una licencia si la necesita.

Git y VS Code

Visual Studio Code incluye compatibilidad integrada con Git, incluida una pestaña de control de código fuente que mostrará los cambios y controlará una variedad de comandos Git automáticamente. Obtenga más información sobre la [compatibilidad con Git de VS Code](#).

Terminaciones de línea de Git

Si está trabajando con la misma carpeta de repositorio entre Windows, WSL o un contenedor, asegúrese de configurar terminaciones de línea coherentes.

Dado que Windows y Linux usan diferentes terminaciones de línea predeterminadas, Git puede notificar un gran número de archivos modificados que no tienen diferencias aparte de sus terminaciones de línea. Para evitar que esto suceda, puede deshabilitar la conversión final de línea mediante un archivo `.gitattributes` o globalmente en Windows. Consulte este [documento de VS Code sobre cómo resolver problemas de terminaciones de líneas de Git](#).

Recursos adicionales

- [WSL y VS Code](#)
- [Laboratorio de aprendizaje de GitHub: Cursos en línea](#)
- [Herramienta de visualización de Git](#)
- [Herramientas de Git: Almacenamiento de credenciales](#)

Introducción a las bases de datos en el Subsistema Windows para Linux

Artículo • 03/10/2023

Esta guía paso a paso le ayudará a empezar a conectar su proyecto en WSL a una base de datos. Introducción a MySQL, PostgreSQL, MongoDB, Redis, Microsoft SQL Server o SQLite.

Requisitos previos

- Ejecutar Windows 11 o Windows 10, [actualizado a la versión 2004](#), la compilación 19041 o posterior.
- Instale una distribución de Linux mediante WSL y cree un nombre de usuario y una contraseña de Linux.

Diferencias entre los sistemas de base de datos

Algunas [opciones populares](#) para un sistema de base de datos incluyen:

- [MySQL](#) (SQL)
- [PostgreSQL](#) (SQL)
- [Microsoft SQL Server](#) (SQL)
- [SQLite](#) (SQL)
- [MongoDB](#) (NoSQL)
- [Redis](#) (NoSQL)

MySQL es una base de datos relacional de SQL código abierto, que organiza los datos en una o varias tablas en las que los tipos de datos pueden estar relacionados entre sí. Es verticalmente escalable, lo que significa que una máquina definitiva hará el trabajo para usted. Actualmente es el más utilizado de los cuatro sistemas de base de datos.

PostgreSQL (a veces conocido como Postgres) es una base de datos relacional de SQL que pone énfasis en la extensibilidad y el cumplimiento de los estándares. Ahora también puede controlar JSON, pero por lo general funciona mejor para los datos estructurados, el escalado vertical y las necesidades compatibles con ACID, como el comercio electrónico y las transacciones financieras.

Microsoft SQL Server incluye SQL Server en Windows, SQL Server en Linux y SQL en Azure. Estos también son sistemas de administración de bases de datos relacionales

configurados en servidores con la función principal de almacenar y recuperar datos según lo solicitado por las aplicaciones de software.

SQLite es una base de datos autocontenido de código abierto, basada en archivos, "sin servidor", conocida por su portabilidad, confiabilidad y buen rendimiento incluso en entornos de poca memoria.

MongoDB es una base de datos de documentos NoSQL diseñada para trabajar con JSON y almacenar datos sin esquemas. Es escalable horizontalmente, lo que significa que varias máquinas más pequeñas harán el trabajo por usted. Es bueno para la flexibilidad y los datos no estructurados, y el almacenamiento en caché de análisis en tiempo real.

Redis es un almacén de estructura de datos en memoria NoSQL de código abierto Usa pares clave-valor para el almacenamiento en lugar de los documentos.

Instalar MySQL

Para instalar MySQL en una distribución de Linux que se ejecuta en WSL, basta con que siga las instrucciones sobre la [instalación de MySQL en Linux](#) en los documentos de MySQL. Es posible que primero deba [habilitar la compatibilidad con systemd](#) en el archivo de configuración `wsl.conf`.

Ejemplo de uso de la distribución de Ubuntu:

1. Abra la línea de comandos de Ubuntu y actualice los paquetes disponibles: `sudo apt update`
2. Una vez actualizados los paquetes, instala MySQL con `sudo apt install mysql-server`.
3. Confirma la instalación y obtén el número de versión `mysql --version`.
4. Inicie MySQL Server/compruebe el estado: `systemctl status mysql`
5. Para abrir el símbolo del sistema de MySQL, escriba: `sudo mysql`
6. Para ver qué bases de datos tiene disponibles, escriba lo siguiente en el símbolo del sistema MySQL: `SHOW DATABASES;`
7. Para crear una nueva base de datos, escriba: `CREATE DATABASE database_name;`
8. Para eliminar una base de datos, escriba: `DROP DATABASE database_name;`

Para más información sobre cómo trabajar con bases de datos MySQL, vea la [documentación de MySQL](#).

Para trabajar con bases de datos MySQL en VS Code, pruebe la [extensión MySQL](#).

También puede ejecutar el script de seguridad incluido. Esto cambia algunas de las opciones predeterminadas menos seguras para cosas como inicios de sesión raíz remotos y usuarios de ejemplo. Este script también incluye pasos para cambiar la contraseña del usuario raíz de MySQL. Para ejecutar el script de seguridad:

1. Inicie un servidor MySQL: `sudo service mysql start`
2. Inicie las indicaciones del script de seguridad: `sudo mysql_secure_installation`
3. El primer mensaje le preguntará si desea configurar el COMPONENTE VALIDAR CONTRASEÑA, que puede usarse para probar la solidez de su contraseña MySQL. Si desea establecer una contraseña sencilla, no debe establecer este componente.
4. Después, establecerá/cambiará una contraseña para el usuario raíz de MySQL, y decidirá si desea quitar usuarios anónimos, si desea permitir que el usuario raíz inicie sesión de forma local y remota, si desea quitar la base de datos de prueba y, por último, si desea volver a cargar las tablas de privilegios inmediatamente.

Instalación de PostgreSQL

Para instalar PostgreSQL en WSL (es decir, Ubuntu):

1. Abre el terminal de WSL (es decir, Ubuntu 18.04).
2. Actualiza los paquetes de Ubuntu: `sudo apt update`.
3. Una vez que los paquetes se hayan actualizado, instala PostgreSQL (y el paquete - contrib que tenga algunas utilidades útiles) con: `sudo apt install postgresql postgresql-contrib`.
4. Confirma la instalación y obtén el número de versión `psql --version`.

Hay 3 comandos que debes conocer una vez instales PostgreSQL:

- `sudo service postgresql status` para comprobar el estado de la base de datos.
- `sudo service postgresql start` para empezar a ejecutar la base de datos.
- `sudo service postgresql stop` para detener la ejecución de la base de datos.

El usuario administrador predeterminado, `postgres`, necesita tener una contraseña asignada para poder conectarse a una base de datos. Para establecer una contraseña:

1. Escribe el comando: `sudo passwd postgres`.
2. Recibirás un aviso para escribir la nueva contraseña.
3. Cierra y vuelve a abrir el terminal.

Para ejecutar PostgreSQL con `psql` shell:

1. Inicia el servicio Postgres: `sudo service postgresql start`.

2. Conéctate al servicio Postgres y abre el shell de psql: `sudo -u postgres psql`.

Una vez que hayas escrito correctamente el shell de psql, verás que el cambio de la línea de comandos tiene el siguiente aspecto: `postgres=#`.

ⓘ Nota

Como alternativa, puedes abrir el shell de psql si cambias al usuario Postgres por `su - postgres` y, a continuación, escribes el comando `psql`.

Para salir de `postgres=#` escriba: `\q` o use la tecla de método abreviado: Ctrl+D

Para ver qué cuentas de usuario se han creado en la instalación de PostgreSQL, usa `psql -c "\du"` desde el terminal WSL o simplemente `\du`, si tienes abierto el shell de psql.

Este comando mostrará las columnas siguientes: Nombre de usuario de la cuenta, Lista de atributos de roles y Miembro de grupo(s) de roles. Para salir y volver a la línea de comandos, escribe `q`.

Para más información sobre cómo trabajar con bases de datos PostgreSQL, vea la [documentación de PostgreSQL](#).

Para trabajar con bases de datos PostgreSQL en VS Code, pruebe la [extensión PostgreSQL](#).

Instalación de MongoDB

Para instalar MongoDB, consulte los documentos de Mongod: [Instalación de MongoDB Community Edition en Linux](#)

La instalación de MongoDB puede requerir pasos ligeramente diferentes en función de la distribución de Linux que se use para la instalación. Tenga en cuenta también que la instalación de MongoDB puede diferir en función del número de versión que desee instalar. Use la lista desplegable versión de la esquina superior izquierda de la documentación de MongoDB para seleccionar la versión que se alinea con el objetivo. Por último, es posible que tenga que [habilitar la compatibilidad con systemd](#) en el archivo de configuración `ws1.conf` de la distribución de Linux que usa con WSL. El comando `systemctl` forma parte del sistema systemd init y puede que no funcione si la distribución usa systemv.

VS Code permite trabajar con bases de datos de MongoDB mediante la [extensión de Azure CosmosDB](#). Puedes crear, administrar y consultar las bases de datos de

MongoDB desde VS Code, así como conectarte a ellas. Para obtener más información, visite la documentación de VS Code: [Trabajar con MongoDB](#).

Obtén más información en los documentos de MongoDB:

- [Introducción al uso de MongoDB](#)
- [Crear usuarios](#)
- [CRUD: Crear, leer, actualizar, eliminar](#)
- [Documentos de referencia](#)

Instalar Microsoft SQL Server

Para instalar SQL Server en una distribución de Linux ejecutada por WSL: [SQL Server en Linux](#). Para trabajar con bases de datos de Microsoft SQL Server en VS Code, pruebe la extensión [MSSQL](#).

Instalar SQLite

Para instalar SQLite en WSL (es decir, Ubuntu):

1. Abre el terminal de WSL (es decir, Ubuntu 18.04).
2. Actualiza los paquetes de Ubuntu: `sudo apt update`.
3. Una vez actualizados los paquetes, instale SQLite3 con `sudo apt install sqlite3`.
4. Confirma la instalación y obtén el número de versión `sqlite3 --version`.

Para crear una base de datos de prueba, denominada "example.db", escriba: `sqlite3 example.db`

Para ver una lista de las bases de datos de SQLite, escriba: `.databases`

Para ver el estado de la base de datos, escriba: `.dbinfo ?DB?`

La base de datos estará vacía después de la creación. Puede crear una nueva tabla para la base de datos con `CREATE TABLE empty (kol INTEGER);`.

Ahora, al escribir, `.dbinfo ?DB?` se mostrará la base de datos que ha creado.

Para salir del símbolo del sistema de SQLite, escriba: `.exit`

Para obtener más información sobre cómo trabajar con una base de datos de SQLite, vea la [documentación de SQLite](#).

Para trabajar con bases de datos de SQLite en VS Code, pruebe la [extensión SQLite](#).

Instalación de Redis

Para instalar Redis en WSL (es decir, Ubuntu):

1. Abre el terminal de WSL (es decir, Ubuntu 18.04).
2. Actualiza los paquetes de Ubuntu: `sudo apt update`.
3. Una vez actualizados los paquetes, instale Redis con: `sudo apt install redis-server`.
4. Confirma la instalación y obtén el número de versión `redis-server --version`.

Para empezar a ejecutar el servidor de Redis: `sudo service redis-server start`

Compruebe si redis funciona (redis-cli es la utilidad de interfaz de línea de comandos para comunicarse con Redis): `redis-cli ping` debe devolver una respuesta de "PONG".

Para dejar de ejecutar el servidor de Redis: `sudo service redis-server stop`

Para más información sobre cómo trabajar con una base de datos de Redis, vea la [documentación de Redis](#).

Para trabajar con bases de datos de Redis en VS Code, pruebe la [extensión Redis](#).

Vea los servicios que ejecutan y configuran alias de perfil.

Para ver los servicios que actualmente se están ejecutando en la distribución de WSL, escriba: `service --status-all`

Escribir `sudo service mongodb start` o `sudo service postgres start` y `sudo -u postgrest psql` puede resultar tedioso. Sin embargo, puedes considerar la posibilidad de configurar los alias en el archivo `.profile` de WSL para que estos comandos sean más rápidos de usar y más fáciles de recordar.

Para configurar tu propio alias personalizado, o acceso directo, a fin de ejecutar estos comandos:

1. Abre el terminal de WSL y escribe `cd ~` para asegurarte de que te encuentras en el directorio raíz.
2. Abre el archivo `.profile`, que controla la configuración del terminal, con el editor de texto de terminal Nano: `sudo nano .profile`.

3. En la parte inferior del archivo (no cambies la configuración `# set PATH`), agrega lo siguiente:

```
Bash
```

```
# My Aliases
alias start-pg='sudo service postgresql start'
alias run-pg='sudo -u postgres psql'
```

Esto te permitirá escribir `start-pg` para empezar a ejecutar el servicio PostgreSQL y `run-pg` para abrir el shell `psql`. Puedes cambiar `start-pg` y `run-pg` a cualquier nombre que quieras, pero ten cuidado de no sobrescribir un comando que Postgres ya uses.

4. Una vez que hayas agregado los nuevos alias, sal del editor de texto de Nano con **Control + X**, selecciona **Y** (Sí) cuando se te pida que guardes y presiona Entrar (el nombre de archivo debe ser `.profile`).
5. Cierra y vuelve a abrir el terminal de WSL y, a continuación, prueba los nuevos comandos de alias.

Solucionar problemas

Error: argumento fdatasync de sincronización de directorios no válido

Asegúrese de que ejecuta la distribución de Linux en modo WSL 2. Para obtener ayuda para cambiar de WSL 1 a WSL 2, vea [Establecer la versión de distribución en WSL 1 o WSL 2](#).

Recursos adicionales

- [Configuración del entorno de desarrollo en Windows](#)

Introducción a los contenedores remotos de Docker en WSL 2

Artículo • 04/01/2024

Esta guía paso a paso le ayudará a empezar a desarrollar con contenedores remotos mediante la **configuración de Docker Desktop para Windows con WSL 2** (Subsistema de Windows para Linux, versión 2).

Docker Desktop para Windows proporciona un entorno de desarrollo para compilar, enviar y ejecutar aplicaciones dockerizadas. Al habilitar el motor basado en WSL 2, puede ejecutar contenedores de Linux y Windows en Docker Desktop en la misma máquina. (Docker Desktop es gratuito para uso personal y pequeñas empresas, para obtener información sobre los precios de Pro, Team o Business, consulte las [Preguntas más frecuentes sobre el sitio de Docker](#)).

ⓘ Nota

Se recomienda usar Docker Desktop debido a su [integración con Windows y el Subsistema de Windows para Linux](#). Sin embargo, aunque Docker Desktop admite la ejecución de contenedores de Linux y Windows, **no** puede ejecutar ambos simultáneamente. Para ejecutar contenedores de Linux y Windows simultáneamente, tendría que instalar y ejecutar una instancia de Docker independiente en WSL. Si necesita ejecutar contenedores simultáneos o prefiere instalar un motor de contenedor directamente en la distribución de Linux, siga las instrucciones de instalación de Linux para ese servicio de contenedor, como [Instalar el motor de Docker en Ubuntu](#) o [Instalar Podman para ejecutar contenedores de Linux](#).

Introducción a los contenedores de Docker

Docker es una herramienta que se usa para crear, implementar y ejecutar aplicaciones mediante contenedores. Los contenedores permiten a los desarrolladores empaquetar una aplicación con todas las partes que necesita (bibliotecas, marcos de trabajo, dependencias, etc.) y enviar todo como un paquete. El uso de un contenedor garantiza que la aplicación se ejecutará de la misma forma, independientemente de la configuración personalizada o de las bibliotecas instaladas anteriormente en el equipo en el que se ejecute, que podría diferir del equipo que se usó para escribir y probar el

código de la aplicación. Esto permite a los desarrolladores centrarse en la escritura de código sin preocuparse por el sistema en el que se ejecutará el código.

Los contenedores de Docker son similares a las máquinas virtuales, pero no crean un sistema operativo virtual completo. En su lugar, Docker permite a la aplicación usar el mismo kernel de Linux que el sistema en el que se ejecuta. Esto permite que el paquete de la aplicación solo requiera partes que aún no existen en el equipo host, lo que reduce el tamaño del paquete y mejora el rendimiento.

La disponibilidad continua, mediante el uso de contenedores de Docker con herramientas como [Kubernetes](#), es otro motivo para la popularidad de los contenedores. Esto permite crear varias versiones del contenedor de la aplicación en momentos diferentes. En lugar de tener que deshacer todo el sistema para las actualizaciones o el mantenimiento, cada contenedor (y sus microservicios específicos) se pueden reemplazar sobre la marcha. Puedes preparar un nuevo contenedor con todas las actualizaciones, configurar el contenedor para producción y simplemente apuntar al nuevo contenedor una vez que esté listo. También puedes archivar versiones diferentes de la aplicación mediante contenedores y mantenerlas en ejecución como reserva de seguridad si es necesario.

Para más información, consulte [Introducción a los contenedores de Docker](#).

Requisitos previos

- WSL versión 1.1.3.0 o posteriores.
- Windows 11 de 64 bits: Home o Pro versión 21H2 o posteriores, o Enterprise o Education versión 21H2 o posteriores.
- Windows 10 de 64 bits (recomendado): Home o Pro 22H2 (compilación 19045) o posteriores, o Enterprise o Education 22H2 (compilación 19045) o posteriores.
(Mínimo): Inicio o Pro 21H2 (compilación 19044) o posteriores, o Enterprise o Education 21H2 (compilación 19044) o posteriores. [Actualización de-Windows](#)
- Procesador de 64 bits con [traducción de direcciones de segundo nivel \(SLAT\)](#).
- 4 GB de RAM del sistema.
- Habilite la virtualización de hardware en BIOS.
- [Instale WSL y configure un nombre de usuario y una contraseña para la distribución de Linux que se ejecuta en WSL 2](#).
- [Instalar Visual Studio Code](#) (*opcional*). Esto proporcionará la mejor experiencia, incluida la capacidad de codificar y depurar dentro de un contenedor remoto de Docker y conectado a la distribución de Linux.
- [Instalar Terminal Windows](#) (*opcional*). Esto proporcionará la mejor experiencia, incluida la capacidad de personalizar y abrir varios terminales en la misma interfaz

(incluidos Ubuntu, Debian, PowerShell, la CLI de Azure o lo que prefiera usar).

- [Regístrese para obtener un identificador de Docker en Docker Hub ↗](#) (opcional).
- Consulte el [Contrato de licencia de Docker Desktop ↗](#) para obtener actualizaciones sobre los términos de uso.

Para más información, consulte los [requisitos del sistema de Docker para instalar Docker Desktop en Windows ↗](#).

Para obtener información sobre cómo instalar Docker en Windows Server, consulte [Introducción: Preparación de Windows para contenedores](#).

Nota

WSL puede ejecutar distribuciones en modo WSL versión 1 o WSL 2. Para comprobarlo, abre PowerShell y escribe: `wsl -l -v`. Asegúrese de que la distribución está establecida para usar WSL 2 escribiendo: `wsl --set-version <distro> 2`. Reemplace `<distro>` por el nombre de la distribución (por ejemplo, Ubuntu 18.04).

En WSL versión 1, debido a diferencias fundamentales entre Windows y Linux, el motor de Docker no se pudo ejecutar directamente dentro de WSL, por lo que el equipo de Docker desarrolló una solución alternativa mediante máquinas virtuales de Hyper-V y LinuxKit. Sin embargo, dado que WSL 2 ahora se ejecuta en un kernel de Linux con capacidad de llamada completa del sistema, Docker puede ejecutarse completamente en WSL 2. Esto significa que los contenedores de Linux se pueden ejecutar de forma nativa sin emulación, lo que da lugar a un mejor rendimiento e interoperabilidad entre las herramientas de Windows y Linux.

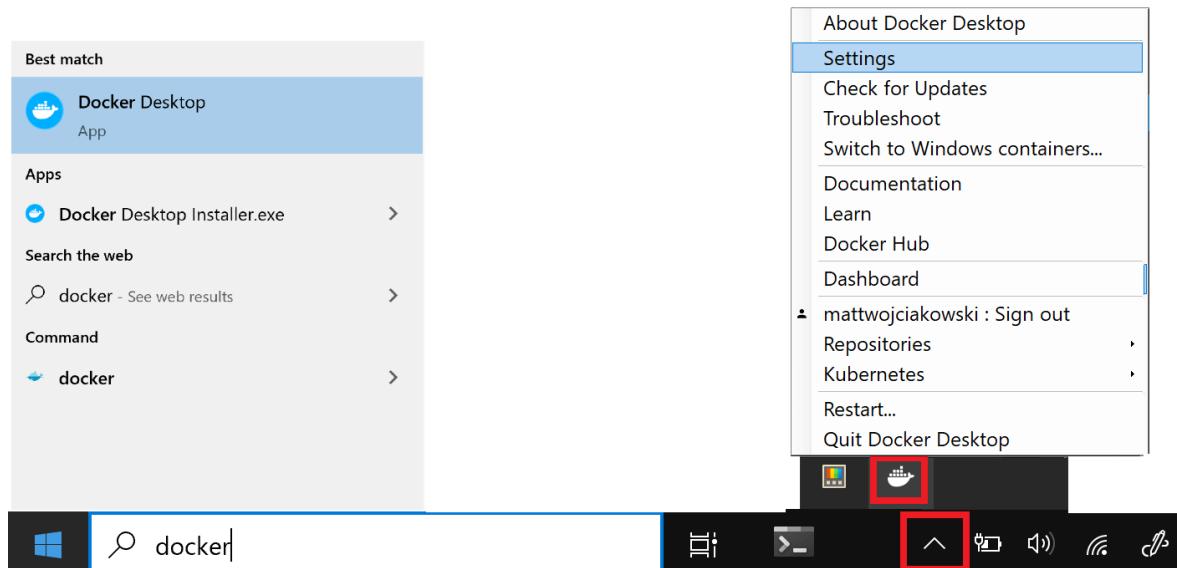
Instalar Docker Desktop

Con el back-end de WSL 2 compatible con Docker Desktop para Windows, puede trabajar en un entorno de desarrollo basado en Linux y compilar contenedores basados en Linux, mientras usa Visual Studio Code para editar y depurar código y ejecutar el contenedor en el explorador Microsoft Edge en Windows.

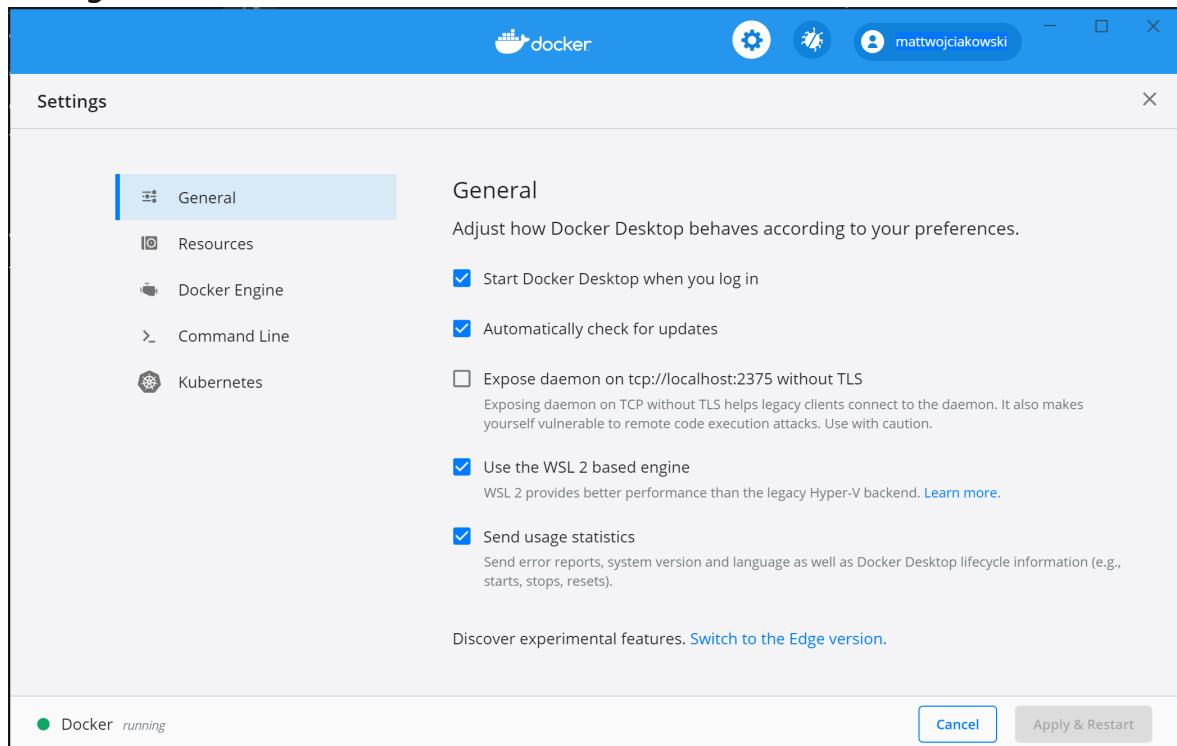
Para instalar Docker (después de [instalar WSL](#)):

1. Descargue [Docker Desktop ↗](#) y siga las instrucciones de instalación.
2. Una vez instalado, inicie Docker Desktop desde el menú Inicio de Windows y, a continuación, seleccione el ícono de Docker en el menú de iconos ocultos de la barra de tareas. Haga clic con el botón derecho en el ícono para mostrar el menú

de comandos de Docker y seleccione "Configuración".

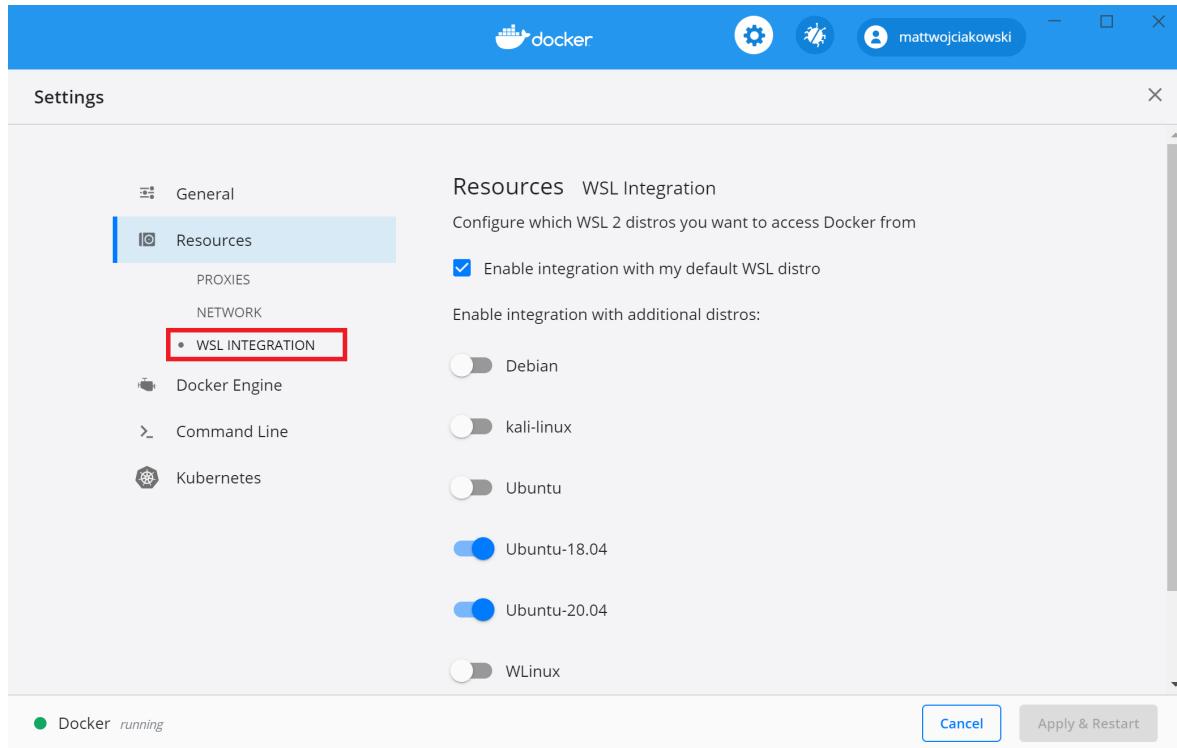


3. Asegúrese de que "Usar el motor basado en WSL 2" esté activado en Configuración>General.



4. Seleccione entre las distribuciones de WSL 2 instaladas en las que desea habilitar la integración de Docker; para ello, vaya a: Configuración>Recursos>Integración de

WSL.



5. Para confirmar que Docker se ha instalado, abra una distribución de WSL (por ejemplo, Ubuntu) y muestre la versión y el número de compilación escribiendo:
`docker --version`

6. Compruebe que la instalación funciona correctamente mediante la ejecución de una imagen de Docker integrada simple mediante: `docker run hello-world`

💡 Sugerencia

Aquí tiene algunos comandos de Docker útiles que debe conocer:

- Enumerar los comandos disponibles en la CLI de Docker, para lo que debes escribir: `docker`
- Enumerar información de un comando específico con: `docker <COMMAND> --help`
- Enumerar las imágenes de Docker en el equipo (que es simplemente la imagen de Hola mundo en este momento), con: `docker image ls --all`
- Enumere los contenedores de la máquina, con: `docker container ls --all` o `docker ps -a` (sin la marca -a show all, solo se mostrarán los contenedores en ejecución).
- Enumere la información de todo el sistema relativa a la instalación de Docker, incluidas las estadísticas y los recursos (memoria y CPU) a su disposición en el contexto de WSL 2, con: `docker info`

Desarrollo en contenedores remotos mediante VS Code

Para empezar a desarrollar aplicaciones mediante Docker con WSL 2, se recomienda usar VS Code, junto con las extensiones WSL, Dev Containers y Docker.

- [Instalar la extensión de WSL en VS Code ↗](#). Esta extensión le permite abrir el proyecto de Linux que se ejecuta en WSL en VS Code (no es necesario preocuparse por problemas de ruta de acceso, compatibilidad binaria u otros desafíos entre sistemas operativos).
- [Instalar la extensión Dev Containers de Visual Studio Code ↗](#). Esta extensión le permite abrir la carpeta o repositorio del proyecto dentro de un contenedor, aprovechando el conjunto de características completo de Visual Studio Code para realizar el trabajo de desarrollo dentro del contenedor.
- [Instalar la extensión de Docker en VS Code ↗](#). Esta extensión agrega la funcionalidad para compilar, administrar e implementar aplicaciones en contenedor desde VS Code. (Necesita la extensión Dev Containers para usar realmente el contenedor como entorno de desarrollo).

Vamos a usar Docker para crear un contenedor de desarrollo para un proyecto de aplicación existente.

1. En este ejemplo, usará el código fuente de mi tutorial de [Hola mundo para Django](#) en los documentos de configuración del entorno de desarrollo de Python. Puede omitir este paso si prefiere usar su propio código fuente del proyecto. Para descargar mi aplicación web HelloWorld-Django desde GitHub, abra un terminal WSL (Ubuntu por ejemplo) y escriba: `git clone`
`https://github.com/mattwojo/helloworld-django.git`

ⓘ Nota

Almacene siempre el código en el mismo sistema de archivos en el que está usando las herramientas. Esto dará como resultado un rendimiento de acceso a archivos más rápido. En este ejemplo, usamos una distribución de Linux (Ubuntu) y queremos almacenar nuestros archivos de proyecto en el sistema de archivos WSL `\ws\`. El almacenamiento de archivos de proyecto en el sistema de archivos de Windows ralentizaría considerablemente el uso de herramientas de Linux en WSL para acceder a esos archivos.

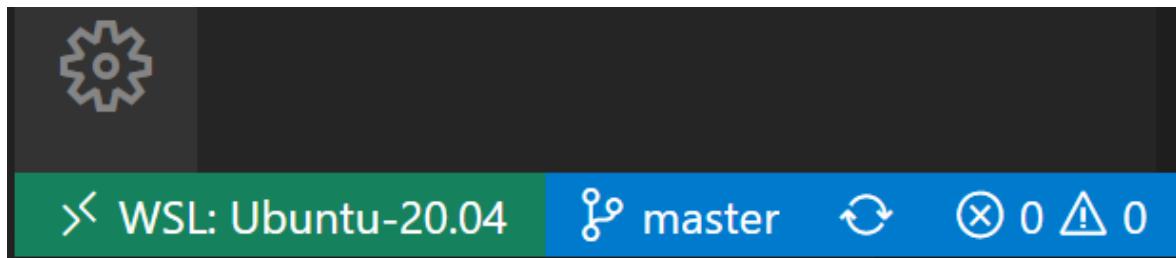
2. Desde el terminal WSL, cambie los directorios a la carpeta de código fuente de este proyecto:

```
Bash  
cd helloworld-django
```

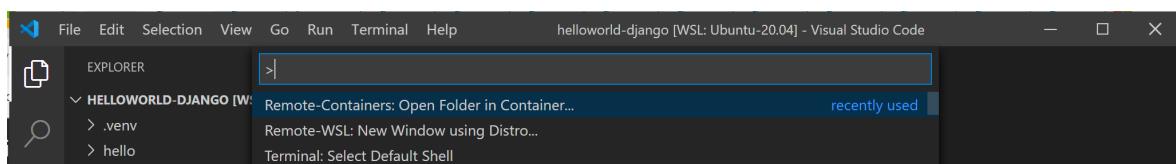
3. Abra el proyecto en VS Code que se ejecuta en el servidor de extensión WSL local escribiendo:

```
Bash  
code .
```

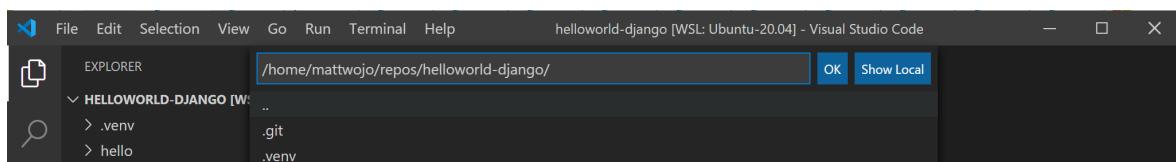
Confirme que está conectado a la distribución de Linux de WSL comprobando el indicador remoto verde en la esquina inferior izquierda de la instancia de VS Code.



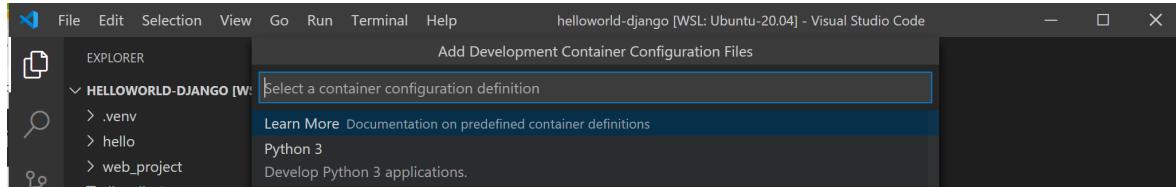
4. En la paleta de comandos de VS Code (Ctrl + Mayús + P), escriba: **Dev Containers: Volver a abrir en contenedor**, ya que usamos una carpeta ya abierta mediante la extensión WSL. Use de forma alternativa **Dev Containers: Abrir carpeta en contenedor...** para elegir una carpeta WSL mediante el recurso compartido `\wsl$` local (desde Windows). Consulte [Inicio rápido: Abrir una carpeta existente en un contenedor](#) de Visual Studio Code para obtener más detalles. Si estos comandos no se muestran al empezar a escribir, compruebe que ha instalado la extensión Dev Containers vinculada anteriormente.



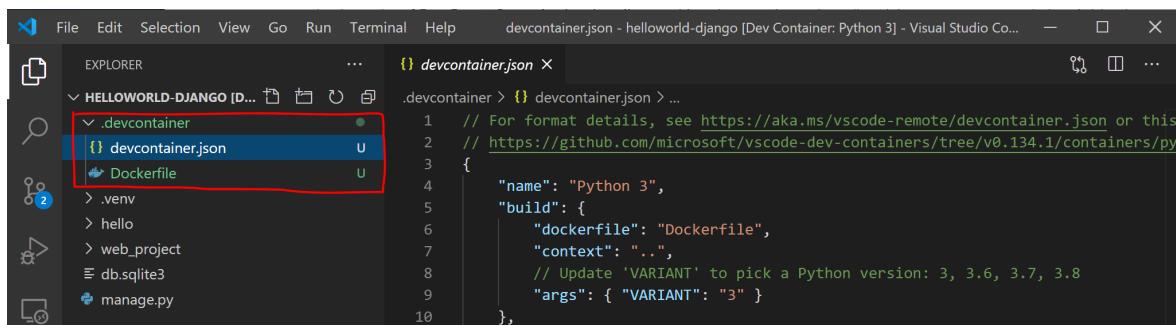
5. Seleccione la carpeta del proyecto que desea incluir en contenedores. En mi caso, esto es `\wsl\Ubuntu-20.04\home\mattwojo\repos\helloworld-django\`



6. Aparecerá una lista de definiciones de contenedor, ya que todavía no hay ninguna configuración de dev container en la carpeta del proyecto (repositorio). La lista de definiciones de configuración de contenedor que aparece se filtra en función del tipo de proyecto. Para mi proyecto de Django, seleccionaré Python 3.

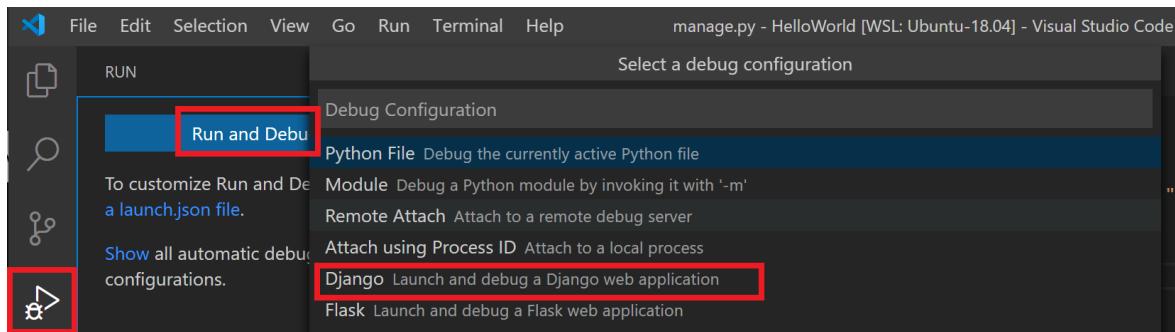


7. Se abrirá una nueva instancia de VS Code, se empezará a compilar nuestra nueva imagen y, una vez completada la compilación, se iniciará el contenedor. Verá que ha aparecido una nueva carpeta de `.devcontainer` con información de configuración de contenedor dentro de un archivo `Dockerfile` y `devcontainer.json`.

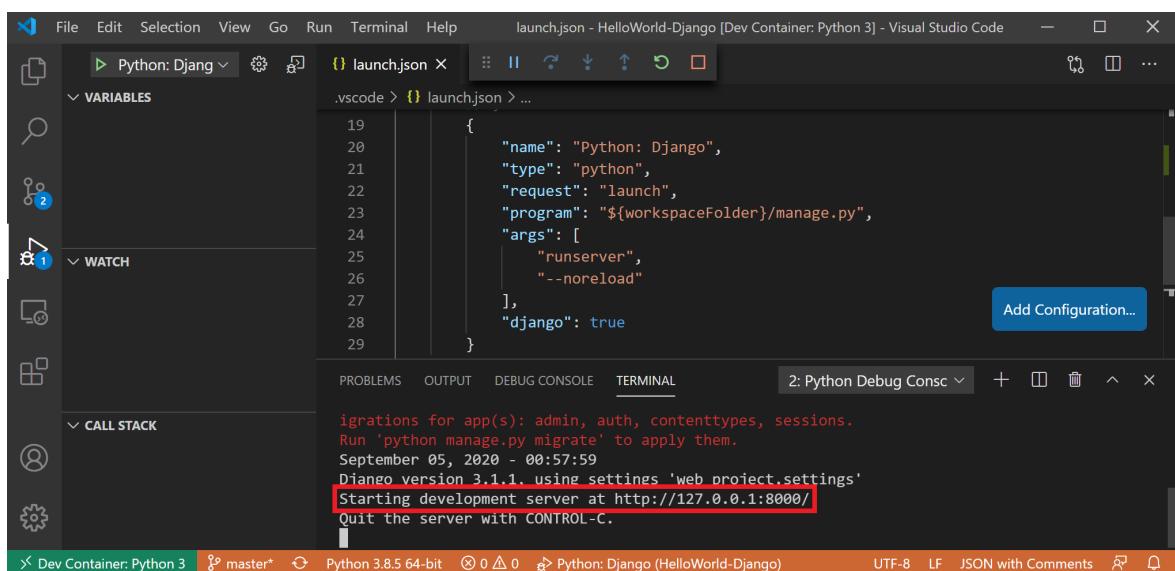


8. Para confirmar que el proyecto todavía está conectado a WSL y dentro de un contenedor, abra el terminal integrado de VS Code (Ctrl + Mayús + ~). Para comprobar el sistema operativo, escriba: `uname` y la versión de Python con: `python3 --version`. Puede ver que el `uname` volvió como "Linux", por lo que aún está conectado al motor de WSL 2 y el número de versión de Python se basará en la configuración del contenedor que puede diferir de la versión de Python instalada en la distribución de WSL.

9. Para ejecutar y depurar la aplicación dentro del contenedor mediante Visual Studio Code, abra primero el menú **Ejecutar** (Ctrl+Mayús+D o seleccione la pestaña de la barra de menús de la izquierda). A continuación, seleccione **Ejecutar y depurar** para seleccionar una configuración de depuración y elija la configuración que mejor se adapte al proyecto (en mi ejemplo, será "Django"). Esto creará un archivo `launch.json` en la carpeta `.vscode` del proyecto con instrucciones sobre cómo ejecutar la aplicación.



10. En VS Code, seleccione **Ejecutar>Iniciar depuración** (o simplemente presione la tecla **F5**). Esto abrirá un terminal dentro de VS Code y verá un resultado que dice algo parecido a: "Iniciando el servidor de desarrollo en <http://127.0.0.1:8000/>". Salga del servidor con CONTROL-C". Mantenga presionada la tecla Control y seleccione la dirección que se muestra para abrir la aplicación en el explorador web predeterminado y ver el proyecto que se ejecuta dentro de su contenedor.



Ahora ha configurado correctamente un contenedor de desarrollo remoto mediante Docker Desktop, con tecnología del back-end de WSL 2, que puede codificar, compilar, ejecutar, implementar o depurar mediante VS Code.

Solucionar problemas

Contexto de Docker de WSL en desuso

Si usaba una versión preliminar técnica anterior de Docker para WSL, es posible que tenga un contexto de Docker denominado "wsl" que ahora está en desuso. Puede comprobar esto con el comando: `docker context ls`. Puede quitar este contexto de "wsl" para evitar errores con el comando: `docker context rm wsl` ya que desea usar el contexto predeterminado tanto para Windows como para WSL2.

Entre los posibles errores que puede encontrar con este contexto de wsl en desuso se incluyen: `docker wsl open //./pipe/docker_wsl: The system cannot find the file specified.` O `error during connect: Get http://%2F%2Fpipe%2Fdocker_wsl/v1.40/images/json?all=1: open //./pipe/docker_wsl: The system cannot find the file specified.`

Para obtener más información sobre este problema, consulte [Configuración de Docker en el sistema Windows para Linux \(WSL2\) en Windows 10](#).

Problemas para encontrar la carpeta de almacenamiento de imágenes de Docker

Docker crea dos carpetas de distribución para almacenar datos:

- `\wsl$\docker-desktop`
- `\wsl$\docker-desktop-data`

Puede encontrar estas carpetas abriendo la distribución de Linux de WSL y escribiendo: `explorer.exe .` para ver la carpeta en el Explorador de archivos de Windows. Escriba: `\wsl\<distro name>\mnt\wsl` y reemplace `<distro name>` por el nombre de la distribución (es decir, Ubuntu-20.04) para ver estas carpetas.

Obtenga más información sobre la localización de ubicaciones de almacenamiento de Docker en WSL. Consulte este [problema desde el repositorio de WSL](#) o esta [publicación de StackOverflow](#).

Para obtener más ayuda con problemas generales de solución de problemas en WSL, consulte el documento [solución de problemas](#).

Recursos adicionales

- [Documentos de Docker: Procedimientos recomendados para Docker Desktop con WSL 2](#)
- [Comentarios sobre Docker Desktop para Windows: Archivo de un problema](#)
- [Blog de VS Code: Instrucciones para elegir un entorno de desarrollo](#)
- [Blog de VS Code: Uso de Docker en WSL 2](#)
- [Blog de VS Code: Uso de contenedores remotos en WSL 2](#)
- [Podcast de Hanselminutes: Haciendo que Docker sea encantador para desarrolladores con Simon Ferquel](#)

Tutorial: Compilación y depuración de C++ con WSL 2 y Visual Studio 2022

Artículo • 01/04/2024

Visual Studio 2022 presenta un conjunto de herramientas de C++ nativo para el desarrollo del Subsistema de Windows para Linux versión 2 (WSL 2). Este conjunto de herramientas ya está disponible en [Visual Studio 2022 17.0](#) o versiones posteriores.

WSL 2 es la nueva versión recomendada del [Subsistema de Windows para Linux](#) (WSL). Proporciona un mejor rendimiento del sistema de archivos de Linux, compatibilidad con la GUI y compatibilidad completa con llamadas del sistema. El conjunto de herramientas de WSL 2 de Visual Studio permite usar Visual Studio para compilar y depurar código de C++ en distribuciones de WSL 2 sin agregar una conexión SSH. Ya puede compilar y depurar código de C++ en distribuciones de WSL 1 mediante el [conjunto de herramientas de WSL 1](#) nativo introducido en Visual Studio 2019 versión 16.1.

El conjunto de herramientas de WSL 2 de Visual Studio admite proyectos de Linux basados en MSBuild y CMake. CMake es nuestra recomendación para todo el desarrollo multiplataforma de C++ con Visual Studio. Aconsejamos el uso de CMake porque compila y depura el mismo proyecto en Windows, WSL y sistemas remotos.

Para ver una presentación en vídeo de la información de este tema, visite el [vídeo Depuración de C++ con distribuciones de WSL 2 y Visual Studio 2022](#).

Información previa sobre el conjunto de herramientas de WSL 2

La compatibilidad multiplataforma de C++ en Visual Studio da por supuesto que todos los archivos de origen se originan en el sistema de archivos de Windows. Cuando se establece como destino una distribución de WSL 2, Visual Studio ejecuta un comando `rsync` local para copiar archivos del sistema de archivos de Windows al de WSL. La copia de `rsync` local no requiere la intervención del usuario. Se produce automáticamente cuando Visual Studio detecta que se está usando una distribución de WSL 2. Para obtener más información sobre las diferencias entre WSL 1 y WSL 2, consulte [Comparación de WSL 1 con WSL 2](#).

La integración de valores preestablecidos de CMake en Visual Studio admite el conjunto de herramientas de WSL 2. Para obtener más información, consulte [Integración de valores preestablecidos de CMake en Visual Studio y Visual Studio Code](#) y

Configuración y compilación con valores preestablecidos de CMake en Visual Studio. También hay información más avanzada en este artículo en la sección [Consideraciones sobre proyectos avanzados de WSL 2 y CMake](#).

Instalación de las herramientas de compilación

Instale las herramientas necesarias para compilar y depurar en WSL 2. Instalará una versión reciente de CMake mediante la implementación binaria de CMake de Visual Studio en un paso posterior.

1. Instale WSL y una distribución de WSL 2 con las instrucciones que se indican en [Instalación de WSL](#).
2. Suponiendo que su distribución use `apt` (en este tutorial se usa Ubuntu), use los siguientes comandos para instalar las herramientas de compilación necesarias en su distribución de WSL 2:

Bash

```
sudo apt update  
sudo apt install g++ gdb make ninja-build rsync zip
```

Los comandos `apt` anteriores instalan:

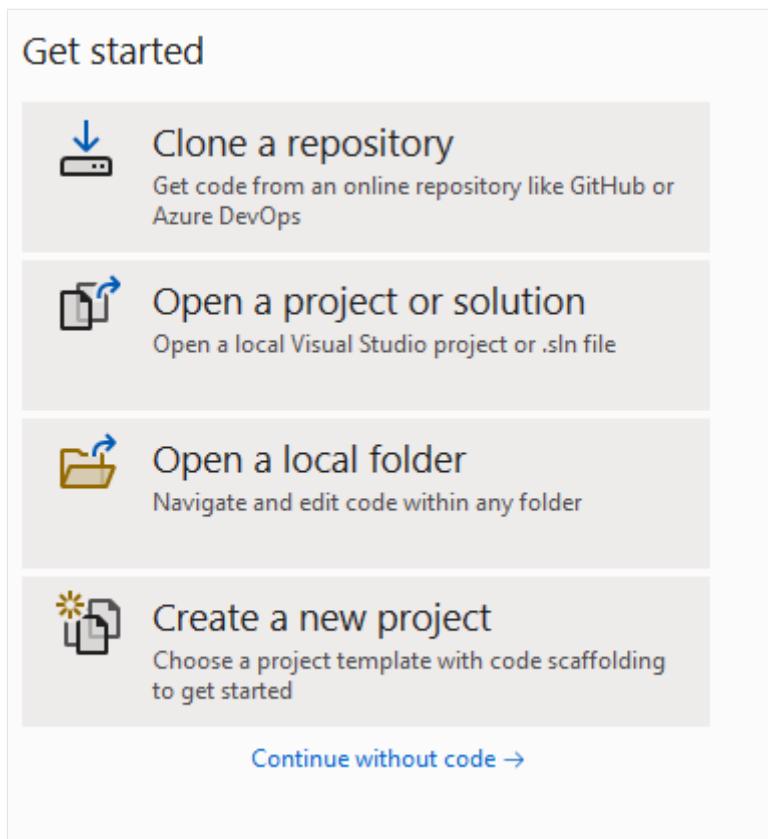
- Un compilador de C++
- `gdb`
- `CMake`
- `rsync`
- `zip`
- Un generador de sistema de compilación subyacente

Desarrollo de CMake multiplataforma con una distribución de WSL 2

En este tutorial se usan GCC y Ninja en Ubuntu, así como Visual Studio 2022 17.0, versión preliminar 2 o posteriores.

Visual Studio define un proyecto de CMake como una carpeta con un archivo `CMakeLists.txt` en la raíz del proyecto. En este tutorial, creará un proyecto de CMake mediante la plantilla **Proyecto de CMake** de Visual Studio:

3. En la pantalla Tareas iniciales de Visual Studio, seleccione **Crear un proyecto**.

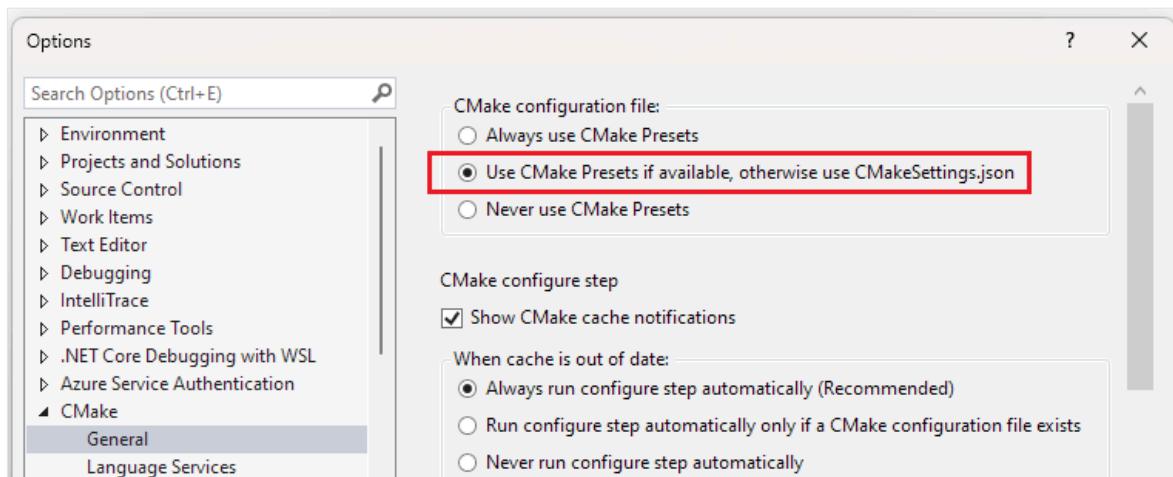


Las opciones disponibles

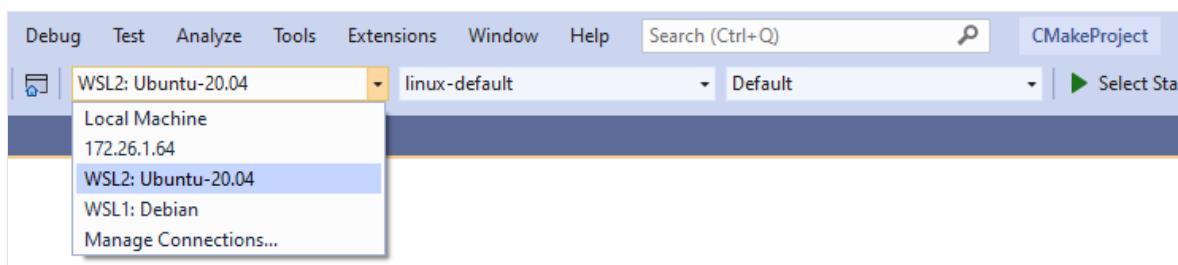
son: Clonar un repositorio, Abrir un proyecto o una solución, Abrir una carpeta local, Crear un nuevo proyecto o Continuar sin código.":::

4. En el cuadro de texto **Buscar plantillas**, escriba "cmake". Elija el tipo **Proyecto de CMake** y seleccione **Siguiente**. Asigne al proyecto un nombre y una ubicación y seleccione **Crear**.

5. Habilite la integración de valores preestablecidos de CMake de Visual Studio. Seleccione **Herramientas > Opciones > CMake > General**. Elija **Prefer using CMake Presets for configure, build, and test** (Preferir el uso de valores preestablecidos de CMake para configurar, compilar y probar) y seleccione **Aceptar**. En su lugar, podría haber agregado un archivo `CMakePresets.json` a la raíz del proyecto. Para obtener más información, vea [Habilitación de la integración de valores preestablecidos de CMake](#).



6. Para activar la integración, en el menú principal, seleccione **Archivo>Cerrar carpeta**. Aparece la página **Tareas iniciales**. En **Abrir recientes**, seleccione la carpeta que acaba de cerrar para volver a abrirla.
7. Hay tres listas desplegables en la barra de menús principal de Visual Studio. Use la lista desplegable de la izquierda para seleccionar el sistema de destino activo. Este es el sistema en el que se invoca CMake para configurar y compilar el proyecto. Visual Studio realiza las consultas en las instalaciones de WSL con `wsl -l -v`. En la imagen siguiente, **WSL2: Ubuntu-20.04** está seleccionado como **sistema de destino**.

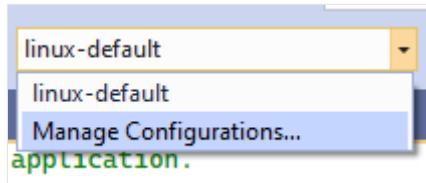


ⓘ Nota

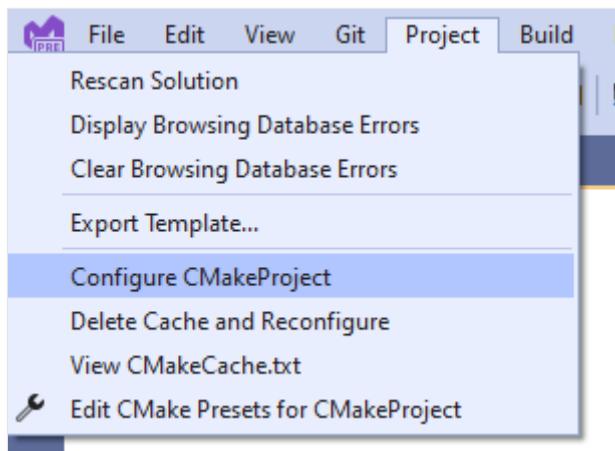
Si Visual Studio empieza a configurar el proyecto automáticamente, lea el paso 11 para administrar la implementación binaria de CMake y, luego, continúe con el paso siguiente. Para personalizar este comportamiento, consulte [Modificación de la configuración automática y las notificaciones de caché](#).

8. Use la lista desplegable del centro para seleccionar el valor preestablecido de configuración activo. Los valores preestablecidos de configuración indican a Visual Studio cómo invocar CMake y generar el sistema de compilación subyacente. En el paso 7, el valor preestablecido de configuración activo es el valor **linux-default** creado por Visual Studio. Para crear un valor preestablecido de

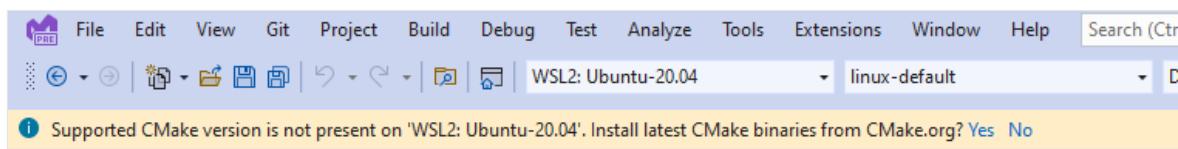
configuración personalizado, seleccione **Administrar configuraciones...**. Para obtener más información sobre cómo configurar valores preestablecidos, consulte [Selección de un valor preestablecido de configuración](#) y [Edición de valores preestablecidos](#).



9. Use la lista desplegable de la derecha para seleccionar el valor preestablecido de compilación activo. Los valores preestablecidos de compilación indican a Visual Studio cómo invocar la compilación. En la ilustración del paso 7, el valor preestablecido de compilación activo es el valor **Predeterminado** creado por Visual Studio. Para obtener más información sobre los valores preestablecidos de compilación, vea [Selección de un valor preestablecido de compilación](#).
10. Configure el proyecto en WSL 2. Si la generación del proyecto no se inicia automáticamente, invoque la configuración de forma manual con **Proyecto>Configurar nombre del proyecto**.

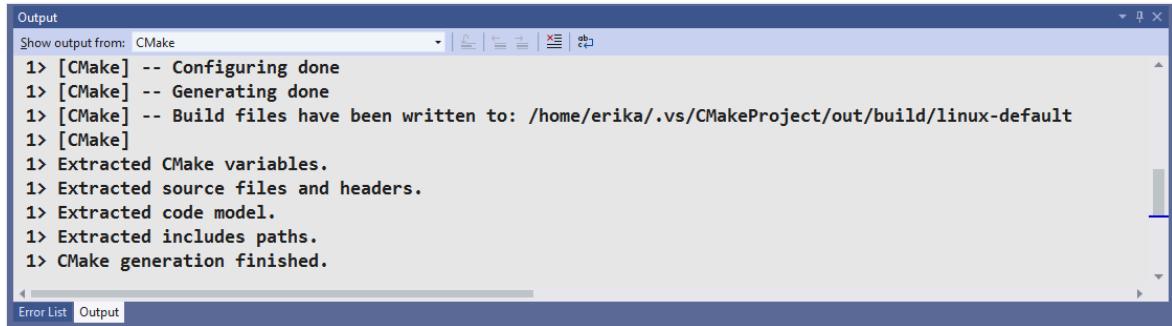


11. Si no tiene una versión compatible de CMake instalada en su distribución de WSL 2, Visual Studio le pedirá debajo de la cinta de opciones del menú principal que implemente una versión reciente de CMake. Seleccione **Sí** para implementar archivos binarios de CMake en la distribución de WSL 2.



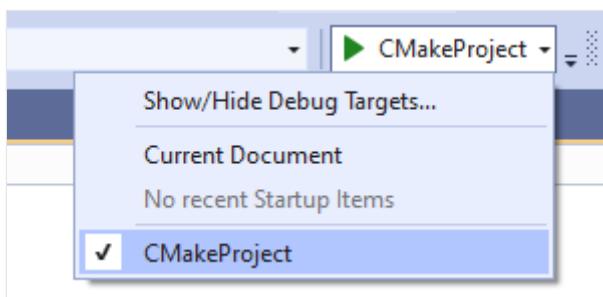
12. Confirme que el paso de configuración se completó y que ve el mensaje de **generación de CMake finalizada** en la ventana Salida debajo del panel de CMake.

Los archivos de compilación se escriben en un directorio del sistema de archivos de la distribución de WSL 2.

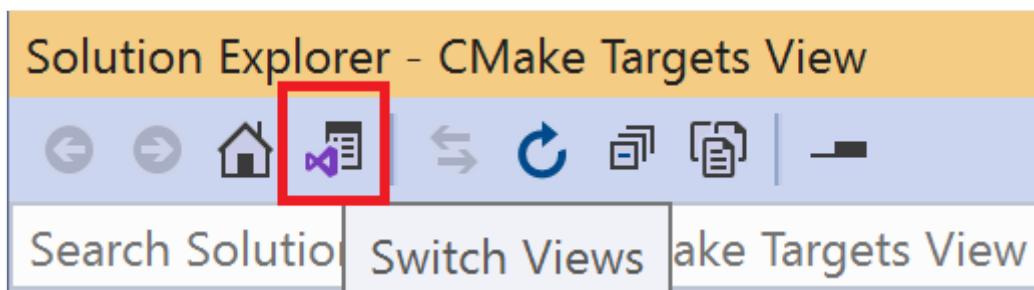


The screenshot shows the Visual Studio Output window with the title 'Output' at the top. It displays the following text:
1> [CMake] -- Configuring done
1> [CMake] -- Generating done
1> [CMake] -- Build files have been written to: /home/erika/.vs/CMakeProject/out/build/linux-default
1> [CMake]
1> Extracted CMake variables.
1> Extracted source files and headers.
1> Extracted code model.
1> Extracted includes paths.
1> CMake generation finished.

13. Seleccione el destino de depuración activo. En el menú desplegable de depuración se enumeran todos los destinos de CMake disponibles para el proyecto.



14. Expanda la subcarpeta del proyecto en el **Explorador de soluciones**. En el archivo `CMakeProject.cpp`, establezca un punto de interrupción en `main()`. También puede navegar a la vista de destinos de CMake si selecciona el botón selector de vista en el **Explorador de soluciones**, resaltado en la captura de pantalla siguiente:



15. Seleccione **Depurar>Iniciar**, o bien presione **F5**. El proyecto se compila, el ejecutable se inicia en la distribución de WSL 2 y Visual Studio detiene la ejecución en el punto de interrupción. La salida del programa (en este caso, "Hello CMake.") está visible en la ventana de la consola de Linux:

The screenshot shows the Visual Studio 2022 interface with a CMake project open. The code editor displays CMakeProject.cpp with the following content:

```
// CMakeProject.cpp : Defines the entry point for the application.
#include "CMakeProject.h"
using namespace std;
int main()
{
    cout << "Hello CMake." << endl;
    return 0;
}
```

The Autos window shows the following variable values:

Name	Type
cout	std::ostream
main	int (void) 0x55555555551a9 <...> int (void)

The Linux Console Window displays the output: Hello CMake.

Ya ha creado y depurado una aplicación de C++ con WSL 2 y Visual Studio 2022.

Consideraciones sobre proyectos avanzados de WSL 2 y CMake

Visual Studio solo proporciona compatibilidad nativa con WSL 2 para proyectos de CMake que usan `CMakePresets.json` como archivo de configuración activo. Para migrar de `CMakeSettings.json` a `CMakePresets.json`, consulte [Habilitación de la integración de valores preestablecidos de CMake en Visual Studio](#).

Si ha establecido como destino una distribución de WSL 2 y no quiere usar el conjunto de herramientas de WSL 2, en la asignación de proveedor de configuración remota de Visual Studio en `CMakePresets.json`, establezca `forceWSL1Toolset` en `true`. Para obtener más información, consulte la [asignación de proveedor de configuración remota de Visual Studio](#).

Si `forceWSL1Toolset` se establece en `true`, Visual Studio no conserva una copia de los archivos de origen en el sistema de archivos de WSL. En su lugar, accede a los archivos de origen en la unidad de Windows montada (`/mnt/...`).

En la mayoría de los casos, es mejor usar el conjunto de herramientas de WSL 2 con distribuciones de WSL 2, ya que WSL 2 es más lento cuando los archivos de proyecto se almacenan en el sistema de archivos de Windows. Para obtener más información sobre

el rendimiento del sistema de archivos en WSL 2, consulte [Comparación de WSL 1 con WSL 2](#).

Especifique la configuración avanzada (como la ruta de acceso al directorio de WSL 2 donde se copia el proyecto, las opciones de origen de copia y los argumentos del comando rsync) en la asignación de proveedor de configuración remota de Visual Studio en `CMakePresets.json`. Para obtener más información, consulte la [asignación de proveedor de configuración remota de Visual Studio](#).

Los encabezados del sistema se siguen copiando automáticamente en el sistema de archivos de Windows para proporcionar la experiencia nativa de IntelliSense. Puede personalizar qué encabezados se incluyen en esta copia o se excluyen de ella en la asignación de proveedor de configuración remota de Visual Studio en `CMakePresets.json`.

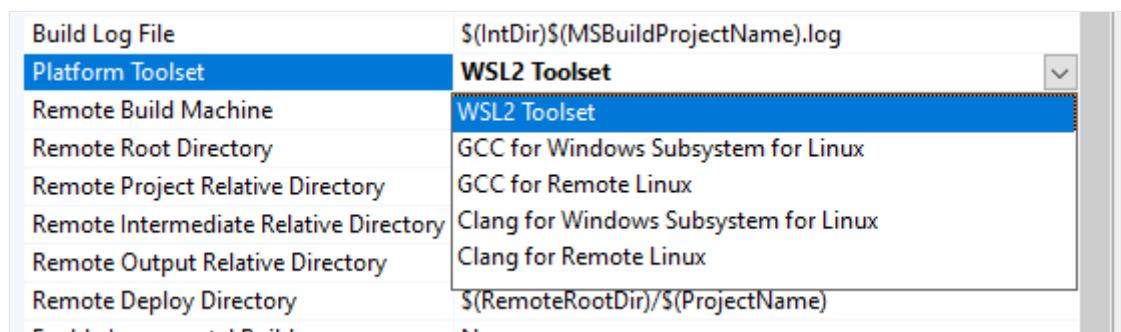
Puede cambiar el modo de IntelliSense o especificar otras opciones de IntelliSense en la asignación de proveedor de configuración de Visual Studio en `CMakePresets.json`. Para obtener más información sobre la asignación de proveedor, consulte [Asignación de proveedor de configuración remota de Visual Studio](#).

WSL 2 y proyectos de Linux basados en MSBuild

Se recomienda el uso de CMake para todo el desarrollo multiplataforma de C++ con Visual Studio, ya que permite compilar y depurar el mismo proyecto en Windows, WSL y sistemas remotos.

Pero es posible que tenga un proyecto de Linux basado en MSBuild.

Si tiene un proyecto de Linux basado en MSBuild, puede actualizarlo al conjunto de herramientas de WSL 2 en Visual Studio. Haga clic con el botón derecho en el proyecto en el Explorador de soluciones y elija **Propiedades > General > Conjunto de herramientas de plataforma**:



Si ha establecido como destino una distribución de WSL 2 y no quiere usar el conjunto de herramientas de WSL 2, en el menú desplegable **Conjunto de herramientas de la plataforma**, seleccione el conjunto de herramientas **GCC para el Subsistema de Windows para Linux o Clang para el Subsistema de Windows para Linux**. Si se selecciona alguno de estos conjuntos de herramientas, Visual Studio no conserva una copia de los archivos de origen en el sistema de archivos de WSL y, en su lugar, accede a los archivos de origen a través de la unidad de Windows montada (`/mnt/...`). Los encabezados del sistema se siguen copiando automáticamente en el sistema de archivos de Windows para proporcionar una experiencia nativa de IntelliSense. Personalice los encabezados que se incluyen en esta copia o se excluyen de ella en **Páginas de propiedades>General**.

En la mayoría de los casos, es mejor usar el conjunto de herramientas de WSL 2 con distribuciones de WSL 2, ya que WSL 2 es más lento cuando los archivos de proyecto se almacenan en el sistema de archivos de Windows. Para obtener más información, consulte [Comparación de WSL 1 con WSL 2](#).

Consulte también

[Vídeo: Depuración de C++ con distribuciones de WSL 2 y Visual Studio 2022 ↗](#)

[Descarga de Visual Studio 2022 ↗](#)

[Creación de un proyecto de CMake para Linux en Visual Studio](#)

[Tutorial: Depuración de un proyecto de CMake en una máquina remota Windows](#)

Comentarios

¿Le ha resultado útil esta página?

 Sí

 No

[Proporcionar comentarios sobre el producto ↗](#) | [Obtener ayuda en Microsoft Q&A](#)

Introducción a la aceleración de GPU para ML en WSL

Artículo • 07/10/2023

El aprendizaje automático (ML) se está convirtiendo en una parte clave de muchos flujos de trabajo de desarrollo. Tanto si es científico de datos, ingeniero de aprendizaje automático como si va a comenzar su recorrido de aprendizaje con ML, el Subsistema de Windows para Linux (WSL) ofrece un entorno excelente para ejecutar las herramientas de aprendizaje automático aceleradas por GPU más comunes y populares.

Hay muchas maneras diferentes de configurar estas herramientas. Por ejemplo, [NVIDIA CUDA en WSL](#), [TensorFlow-DirectML](#) y [PyTorch-DirectML](#) ofrecen diferentes formas de usar la GPU para ML con WSL. Para más información sobre los motivos para elegir uno frente a otro, consulte [Aprendizaje automático acelerado por GPU](#).

En esta guía se muestra cómo configurar:

- NVIDIA CUDA si tiene una tarjeta gráfica NVIDIA y ejecuta un contenedor de marcos de ML de ejemplo
- TensorFlow-DirectML y PyTorch-DirectML en la tarjeta gráfica AMD, Intel o NVIDIA

Requisitos previos

- Asegúrese de que está ejecutando [Windows 11](#) o [Windows 10, versión 21H2](#) o posterior.
- [Instale WSL y configure un nombre de usuario y una contraseña para la distribución de Linux](#).

Configuración de NVIDIA CUDA con Docker

1. [Descarga e instalación del controlador más reciente para la GPU de NVIDIA](#)
2. Instale [Docker Desktop](#) o instale el motor de Docker directamente en WSL mediante la ejecución del comando siguiente.

Bash

```
curl https://get.docker.com | sh
```

Bash

```
sudo service docker start
```

3. Si instaló el motor de Docker directamente, [instale NVIDIA Container Toolkit](#) siguiendo los pasos que se indican a continuación.

Configure el repositorio estable para NVIDIA Container Toolkit mediante la ejecución de los siguientes comandos:

Bash

```
distribution=$( . /etc/os-release; echo $ID$VERSION_ID )
```

Bash

```
curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo gpg --dearmor -o /usr/share/keyrings/nvidia-docker-keyring.gpg
```

Bash

```
curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list | sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-docker-keyring.gpg] https://#g' | sudo tee /etc/apt/sources.list.d/nvidia-docker.list
```

Instale los paquetes y dependencias en tiempo de ejecución de NVIDIA mediante la ejecución de los comandos:

Bash

```
sudo apt-get update
```

Bash

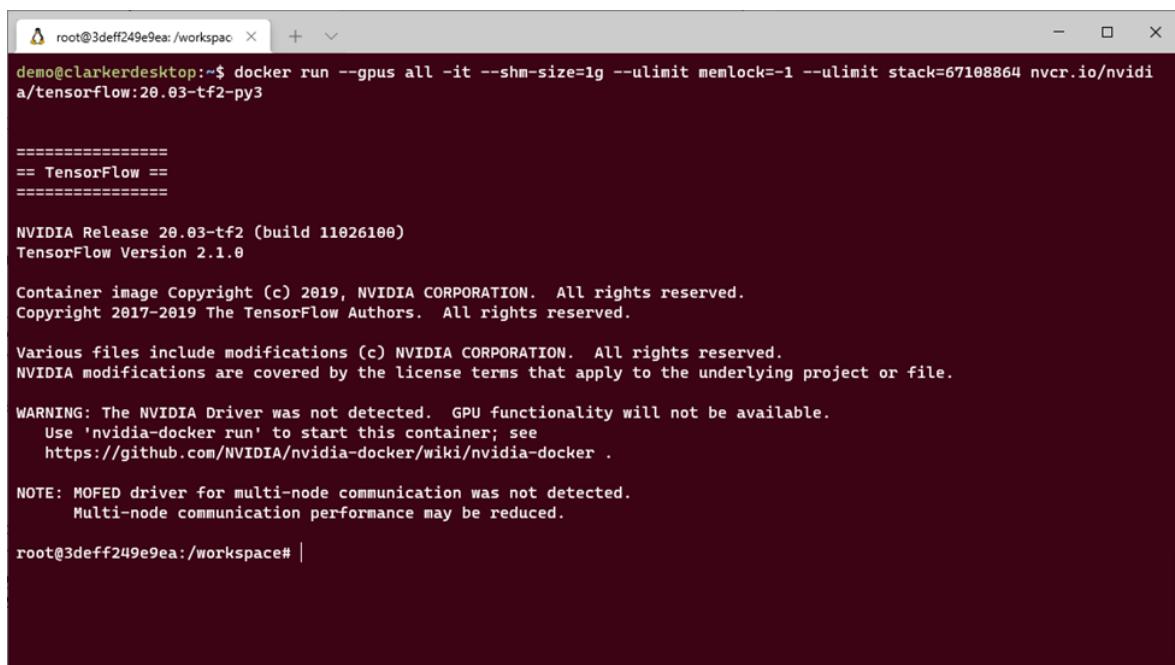
```
sudo apt-get install -y nvidia-docker2
```

4. Ejecute un contenedor y un ejemplo de marco de aprendizaje automático.

Para ejecutar un contenedor de marco de aprendizaje automático y empezar a usar la GPU con este contenedor nvidia NGC TensorFlow, escriba el comando :

Bash

```
docker run --gpus all -it --shm-size=1g --ulimit memlock=-1 --ulimit stack=67108864 nvcr.io/nvidia/tensorflow:20.03-tf2-py3
```



The screenshot shows a terminal window with the following text:

```
root@3deff249e9ea:/workspace ~ + - x
demo@clarkerdesktop:~$ docker run --gpus all -it --shm-size=1g --ulimit memlock=-1 --ulimit stack=67108864 nvcr.io/nvidia/tensorflow:20.03-tf2-py3

=====
== TensorFlow ==
=====

NVIDIA Release 20.03-tf2 (build 11026100)
TensorFlow Version 2.1.0

Container image Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
Copyright 2017-2019 The TensorFlow Authors. All rights reserved.

Various files include modifications (c) NVIDIA CORPORATION. All rights reserved.
NVIDIA modifications are covered by the license terms that apply to the underlying project or file.

WARNING: The NVIDIA Driver was not detected. GPU functionality will not be available.
Use 'nvidia-docker run' to start this container; see
https://github.com/NVIDIA/nvidia-docker/wiki/nvidia-docker .

NOTE: MOFED driver for multi-node communication was not detected.
Multi-node communication performance may be reduced.

root@3deff249e9ea:/workspace# |
```

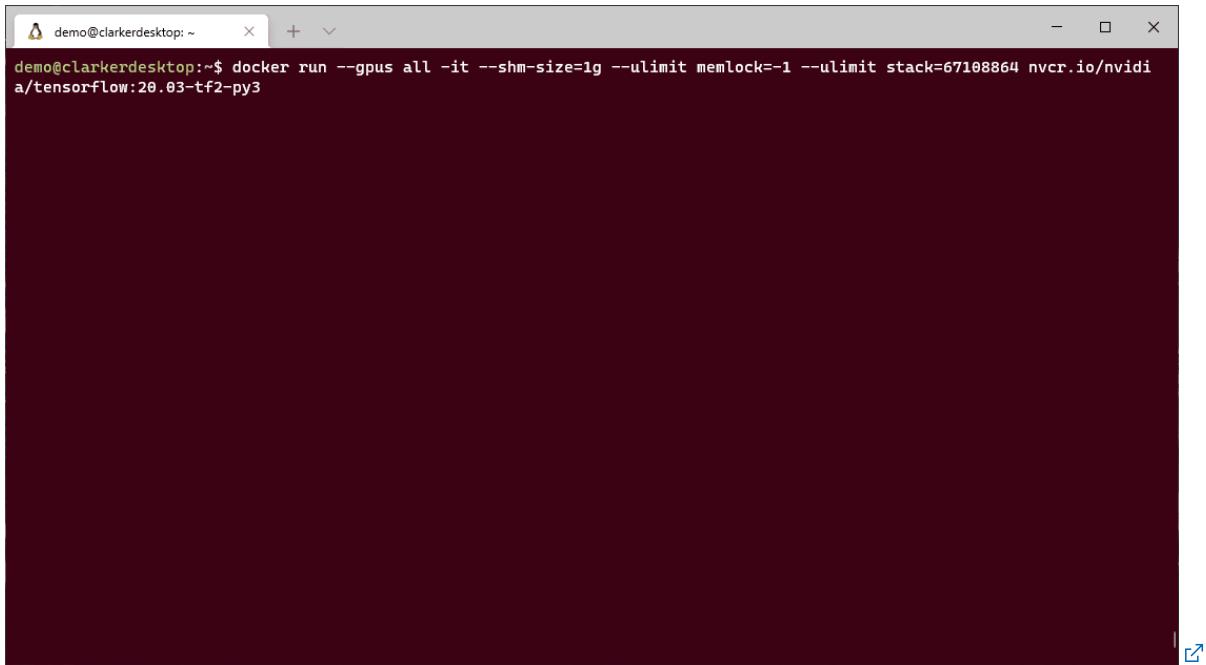
Puede ejecutar un ejemplo de modelo entrenado previamente integrado en este contenedor mediante la ejecución de los comandos:

Bash

```
cd nvidia-examples/cnn/
```

Bash

```
python resnet.py --batch_size=64
```

A screenshot of a terminal window titled "demo@clarkerdesktop: ~". The command entered is "docker run --gpus all -it --shm-size=1g --ulimit memlock=-1 --ulimit stack=67108864 nvcr.io/nvidia/tensorflow:20.03-tf2-py3". The terminal is dark-themed with white text.

Encontrará más formas de configurar y usar NVIDIA CUDA en la [Guía del usuario de NVIDIA CUDA en WSL](#).

Configuración de TensorFlow-DirectML o PyTorch-DirectML

1. Descargue e instale el controlador más reciente desde el sitio web de los proveedores de GPU: [AMD](#), [Intel](#) o [NVIDIA](#).
2. Configure un entorno de Python.

Se recomienda configurar un entorno de Python virtual. Existen muchas herramientas que puede usar para configurar un entorno de Python virtual; en este caso, usaremos la característica [miniconda de Anaconda](#).

Bash

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
```

Bash

```
bash Miniconda3-latest-Linux-x86_64.sh
```

Bash

```
conda create --name directml python=3.7 -y
```

```
Bash
```

```
conda activate directml
```

3. Instale el marco de aprendizaje automático respaldado por DirectML de su elección.

TensorFlow-DirectML:

```
Bash
```

```
pip install tensorflow-directml
```

PyTorch-DirectML:

```
Bash
```

```
sudo apt install libblas3 libomp5 liblapack3
```

```
Bash
```

```
pip install pytorch-directml
```

4. Ejecute una muestra de adición rápida en una sesión interactiva de Python para [TensorFlow-DirectML](#) o [PyTorch-DirectML](#) para asegurarse de que todo funciona.

Si tiene preguntas o tiene problemas, visite el [repositorio de DirectML en GitHub ↗](#).

Varias GPU

Si tiene varias GPU en la máquina, también puede acceder a ellas dentro de WSL. Sin embargo, solo podrá acceder de uno en uno. Para elegir una GPU específica, establezca la variable de entorno siguiente en el nombre de la GPU tal como aparece en el administrador de dispositivos:

```
Bash
```

```
export MESA_D3D12_DEFAULT_ADAPTER_NAME=<NameFromDeviceManager>
```

Esto hará una coincidencia de cadena, por lo que si se establece en "NVIDIA" que coincidirá con la primera GPU que comienza con "NVIDIA".

Recursos adicionales

- [Guía para configurar NVIDIA CUDA en WSL ↗](#)
- [Guía para configurar TensorFlow con DirectML en WSL](#)
- [TensorFlow con ejemplos de DirectML ↗](#)
- [Guía para configurar PyTorch con DirectML en WSL](#)
- [PyTorch con ejemplos de DirectML ↗](#)

Ejecución de aplicaciones de GUI de Linux en el Subsistema de Windows para Linux

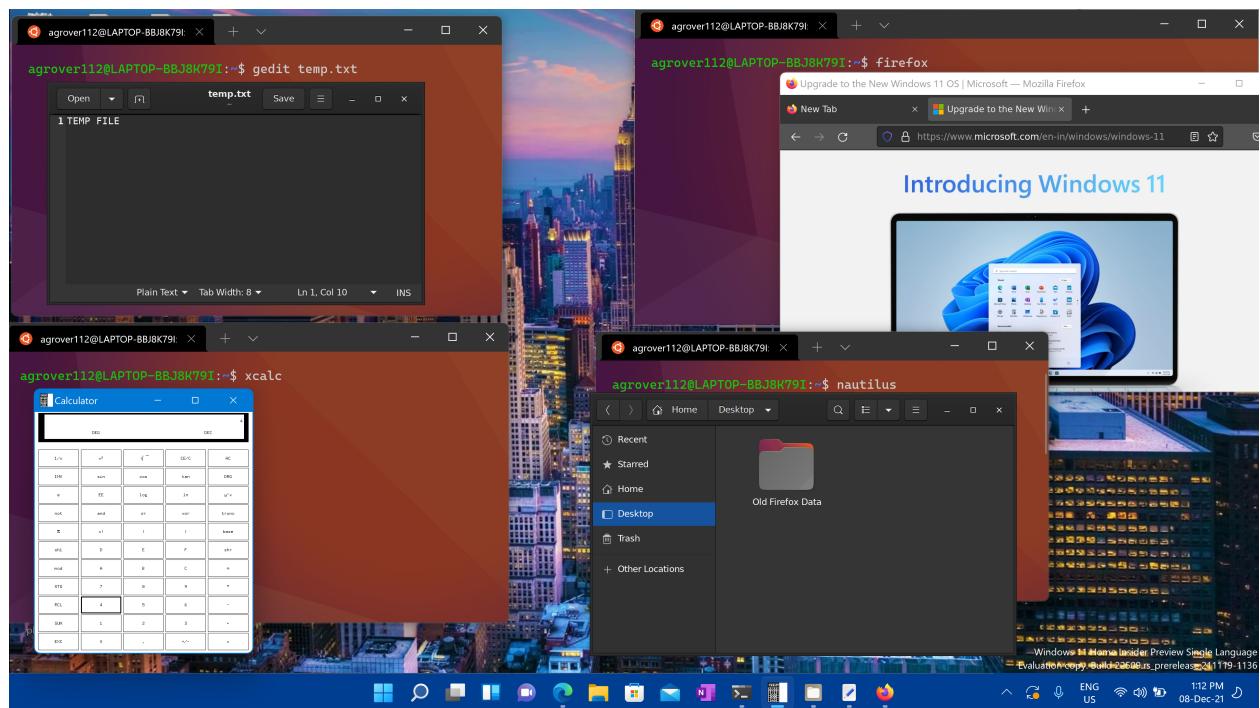
Artículo • 13/01/2024

Subsistema de Windows para Linux (WSL) ahora admite la ejecución de aplicaciones GUI de Linux (X11 y Wayland) en Windows en una experiencia de escritorio totalmente integrada.

WSL 2 permite que las aplicaciones de GUI de Linux se sientan nativas y naturales para usarlas en Windows.

- Inicio de aplicaciones de Linux desde el menú Inicio de Windows
- Anclar aplicaciones De Linux a la barra de tareas de Windows
- Uso de alt-tab para cambiar entre aplicaciones de Linux y Windows
- Cortar y pegar entre aplicaciones de Windows y Linux

Ahora puede integrar aplicaciones de Windows y Linux en el flujo de trabajo para una experiencia de escritorio sin problemas.



Instalación de compatibilidad con aplicaciones de GUI de Linux

Requisitos previos

- Tendrás que estar en **La compilación 19044+ de Windows 10 o Windows 11** para acceder a esta característica.
- **Controlador instalado para vGPU**

Para ejecutar aplicaciones de GUI de Linux, primero debe instalar el controlador que coincide con el sistema siguiente. Esto le permitirá usar una GPU virtual (vGPU) para que pueda beneficiarse de la representación de OpenGL acelerada por hardware.

- [Intel Controlador de GPU](#)
- [AMD Controlador de GPU](#)
- [Controlador de GPU NVIDIA](#)

Instalación nueva: no hay instalación previa de WSL

Ahora puede instalar todo lo que necesita para ejecutar el Subsistema de Windows para Linux (WSL) si escribe este comando en PowerShell o el símbolo del sistema de Windows del administrador y, a continuación, reinicia la máquina.

```
PowerShell
```

```
wsl --install
```

Una vez que la máquina haya terminado de reiniciarse, la instalación continuará y se le pedirá que escriba un nombre de usuario y una contraseña. Esta será la credencial de Linux para la distribución de Ubuntu.

Ya está listo para empezar a usar aplicaciones de GUI de Linux en WSL.

Para más información, consulte la [instalación de WSL](#).

Instalación de WSL existente

Si ya tiene WSL instalado en el equipo, puede actualizar a la versión más reciente que incluye compatibilidad con la GUI de Linux mediante la ejecución del comando update desde un símbolo del sistema con privilegios elevados.

1. Haga clic en **Inicio**, escriba **PowerShell**, haga clic con el botón derecho en **Windows PowerShell** y, después, haga clic en **Ejecutar como administrador**.
2. Escriba el comando WSL update:

```
PowerShell
```

```
wsl --update
```

3. Tendrá que reiniciar WSL para que la actualización surta efecto. Puede reiniciar WSL ejecutando el comando shutdown en PowerShell.

```
PowerShell
```

```
wsl --shutdown
```

ⓘ Nota

Las aplicaciones de GUI de Linux solo se admiten con WSL 2 y no funcionarán con una distribución de Linux configurada para WSL 1. Obtenga información sobre [cómo cambiar la distribución de WSL 1 a WSL 2](#).

Ejecución de aplicaciones de GUI de Linux

Puede ejecutar los siguientes comandos desde el terminal linux para descargar e instalar estas aplicaciones de Linux populares. Si usa una distribución diferente a Ubuntu, puede usar un administrador de paquetes diferente al de apt. Una vez instalada la aplicación Linux, puede encontrarla en su menú **Inicio** bajo el nombre de la distribución. Por ejemplo: `Ubuntu -> Microsoft Edge`.

ⓘ Nota

La compatibilidad con aplicaciones de GUI en WSL no proporciona una experiencia de escritorio completa. Se basa en el escritorio de Windows, por lo que es posible que no se admita la instalación de aplicaciones o herramientas centradas en el escritorio. Para solicitar soporte técnico adicional, puede presentar un problema en el [repositorio de WSLg en GitHub](#).

Actualización de los paquetes en la distribución

```
Bash
```

```
sudo apt update
```

Instalación del Editor de texto Gnome

Gnome Text Editor es el editor de texto predeterminado del entorno de escritorio GNOME.

Bash

```
sudo apt install gnome-text-editor -y
```

Para iniciar el archivo bashrc en el editor, escriba: `gnome-text-editor ~/ .bashrc`

ⓘ Nota

El editor de texto de GNOME  reemplaza a gedit como editor de texto predeterminado de GNOME/Ubuntu en Ubuntu 22.10. Si ejecuta una versión anterior de Ubuntu y quiere usar gedit , el anterior editor de texto predeterminado, use `sudo apt install gedit -y`.

Instalación de GIMP

GIMP es un editor de gráficos raster de código abierto y gratuito que se usa para la manipulación de imágenes y la edición de imágenes, dibujo de forma libre, transcodificación entre diferentes formatos de archivo de imagen y tareas más especializadas.

Bash

```
sudo apt install gimp -y
```

Para iniciararlo, escriba: `gimp`

Instalación de Nautilus

Nautilus, también conocido como Archivos GNOME, es el administrador de archivos para el escritorio GNOME. (Similar al Explorador de archivos de Windows).

Bash

```
sudo apt install nautilus -y
```

Para iniciararlo, escriba: `nautilus`

Instalar VLC

VLC es un reproductor y marco multimedia multiplataforma gratuito y código abierto que reproduce la mayoría de los archivos multimedia.

Bash

```
sudo apt install vlc -y
```

Para iniciararlo, escriba: `vlc`

Instalación de aplicaciones X11

X11 es el sistema de ventanas de Linux y se trata de una colección variada de aplicaciones y herramientas que se incluyen con él, como `xclock`, calculadora `xcalc`, `xclipboard` para cortar y pegar, `xev` para pruebas de eventos, etc. Consulte los [documentos de x.org](#) para obtener más información.

Bash

```
sudo apt install x11-apps -y
```

Para iniciararlo, escriba el nombre de la herramienta que desea usar. Por ejemplo:

- `xcalc`, `xclock`, `xeyes`

Instalación de Google Chrome para Linux

Para instalar Google Chrome para Linux:

1. Cambie los directorios a la carpeta temporal: `cd /tmp`
2. Use `wget` para descargarlo: `wget https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb`
3. Instale el paquete: `sudo apt install --fix-missing ./google-chrome-stable_current_amd64.deb`

*La opción `--fix-missing` se usa para corregir las dependencias que faltan que pueden surgir durante el proceso de instalación. `./` en el comando especifica el directorio actual donde se encuentra el archivo `.deb`. Si el archivo `.deb` se encuentra en un directorio diferente, deberá especificar la ruta de acceso al archivo en el comando.

Para iniciararlo, escriba: `google-chrome`

Instalación del explorador Microsoft Edge para Linux

Busque información sobre cómo [instalar el explorador Microsoft Edge para Linux mediante la línea de comandos en el sitio de Edge Insider](#). Seleccione **Obtener instrucciones** en la sección Instalación de la línea de comandos de la página.

Para iniciararlo, escriba: `microsoft-edge`

Solucionar problemas

Si tiene algún problema al iniciar aplicaciones de GUI, consulte esta guía en primer lugar: [Diagnóstico de problemas de tipo "no se puede abrir la presentación" con WSLg](#)

Instalación de Node.js en el Subsistema de Windows para Linux (WSL2)

Artículo • 13/07/2023

Si usa Node.js para fines profesionales, si es importante encontrar la velocidad de rendimiento y la compatibilidad de llamadas del sistema, si desea ejecutar [contenedores de Docker](#) que aprovechen las áreas de trabajo de Linux y eviten tener que mantener scripts de compilación de Linux y Windows, o simplemente prefiere usar una línea de comandos de Bash, seguramente querrá instalar Node.js en el Subsistema de Windows para Linux (más concretamente en WSL 2).

El Subsistema de Windows para Linux (WSL) le permite instalar la distribución de Linux que prefiera (Ubuntu es nuestra distribución predeterminada) para que pueda tener coherencia entre el entorno de desarrollo (donde escribe código) y el de producción (el servidor en el que se implementa el código).

ⓘ Nota

Si es la primera vez que usa Node.js para sus desarrollos y desea empezar a aprender rápidamente, [instale Node.js en Windows](#). Esta recomendación también le resultará útil si planea usar un entorno de producción de Windows Server.

Instalación de WSL 2

WSL 2 es la versión más reciente disponible en Windows y se recomienda para flujos de trabajo de desarrollo de Node.js profesionales. Para habilitar e instalar WSL 2, sigue los pasos que se describen en la [documentación de instalación de WSL](#). Estos pasos incluirán la elección de una distribución de Linux (por ejemplo, Ubuntu).

Una vez que hayas instalado WSL 2 y una distribución de Linux, abre la distribución de Linux (puedes encontrarla en el menú Inicio de Windows), y comprueba la versión y el nombre de código mediante el comando: `lsb_release -dc`.

Se recomienda actualizar la distribución de Linux con regularidad, incluso inmediatamente después de instalarla, para asegurarse de que tiene los paquetes más recientes. Windows no controla automáticamente esta actualización. Para actualizar la distribución, usa el comando: `sudo apt update && sudo apt upgrade`.

Instalación de Terminal Windows (opcional)

Terminal Windows es un shell de línea de comandos mejorado que permite ejecutar varias pestañas, con el fin de que pueda cambiar rápidamente de línea de comandos de Linux, símbolo del sistema de Windows, PowerShell, CLI de Azure, o lo que prefiera usar. También puede crear enlaces de teclado personalizados (teclas de método abreviado para abrir o cerrar pestañas, copiar y pegar, etc.), usar la característica de búsqueda, personalizar el terminal con temas (combinaciones de colores, estilos y tamaños de fuente, imagen de fondo/desenfoque/transparencia), etc. [Encontrará más información en la documentación de Terminal Windows](#).

[Instalación de Terminal Windows desde Microsoft Store](#): si la instalación se realiza a través de Microsoft Store, las actualizaciones se controlan automáticamente.

Instalación de nvm, node.js y npm

Además de elegir si se desea realizar la instalación en Windows o WSL, al instalar Node.js es preciso elegir más opciones. Se recomienda usar un administrador de versiones, ya que las versiones cambian con mucha rapidez. Es probable que tenga que cambiar de versión de Node.js en función de las necesidades de los distintos proyectos en los que se trabaje. El administrador de versiones de Node, más comúnmente denominado nvm, es la forma más habitual de instalar varias versiones de Node.js. Te guiaremos por los pasos necesarios para instalar nvm y luego usarlo para instalar Node.js y el administrador de paquetes de Node (npm). Hay [administradores de versiones alternativos](#) para tener en cuenta, que también se tratan en la sección siguiente.

ⓘ Importante

Siempre se recomienda quitar cualquier instalación existente de Node.js o npm del sistema operativo antes de instalar un administrador de versiones, ya que los distintos tipos de instalación pueden provocar conflictos extraños y confusos. Por ejemplo, la versión de Node que se puede instalar con el comando `apt-get` de Ubuntu está obsoleta actualmente. Para obtener ayuda con la eliminación de instalaciones anteriores, consulta [Cómo eliminar nodejs de Ubuntu](#).

1. Abra la línea de comandos de Ubuntu (o la distribución que prefiera).
2. Instala cURL (una herramienta que se usa para descargar contenido de Internet en la línea de comandos) con: `sudo apt-get install curl`

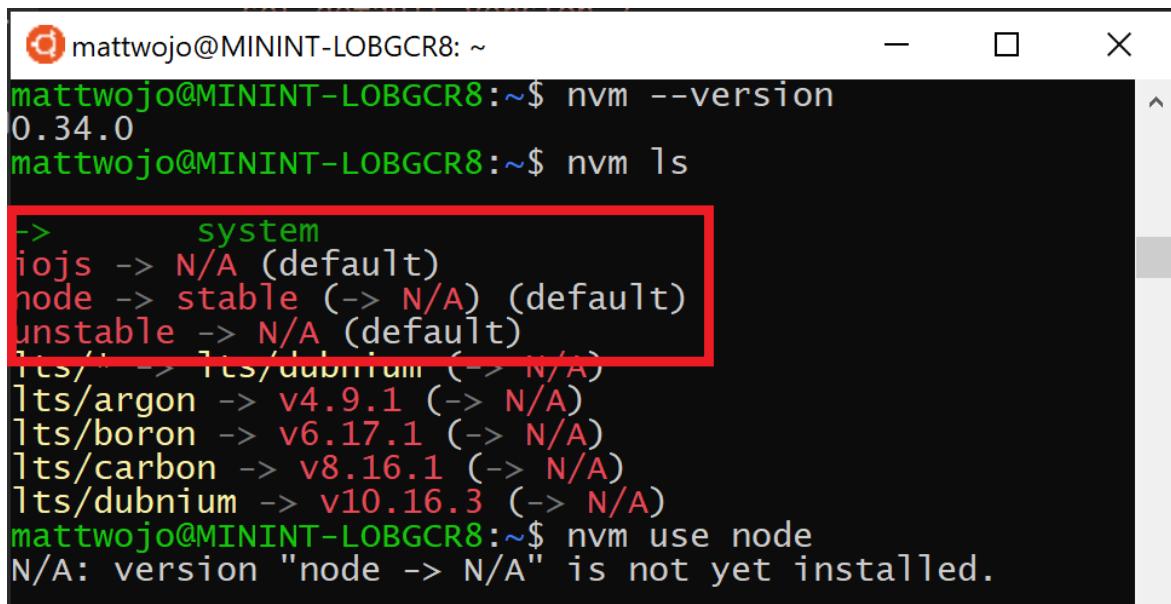
3. Instala nvm con: `curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/master/install.sh | bash`

ⓘ Nota

La instalación de una versión más reciente de NVM con cURL reemplazará la anterior, lo que dejará intacta la versión de Node utilizada para instalar NVM. Consulte la página del proyecto de GitHub para obtener información de la versión más reciente de NVM ↴.

4. Para verificar la instalación, escribe: `command -v nvm`... esto debería devolver "nvm"; si recibes "No se encuentra el comando" o ninguna respuesta, cierra el terminal actual, vuelve a abrirlo e inténtalo de nuevo. [Obtén más información en el repositorio de GitHub de nvm ↴](#).

5. Enumera qué versiones de Node están instaladas actualmente (en este momento no debe haber ninguna): `nvm ls`



The screenshot shows a terminal window with the following content:

```
mattwojo@MININT-LOBGCR8:~$ nvm --version
0.34.0
mattwojo@MININT-LOBGCR8:~$ nvm ls
->      system
iojs -> N/A (default)
node -> stable (-> N/A) (default)
unstable -> N/A (default)
lts/* -> lts/dubnium (-> N/A)
lts/argon -> v4.9.1 (-> N/A)
lts/boron -> v6.17.1 (-> N/A)
lts/carbon -> v8.16.1 (-> N/A)
lts/dubnium -> v10.16.3 (-> N/A)
mattwojo@MININT-LOBGCR8:~$ nvm use node
N/A: version "node -> N/A" is not yet installed.
```

A red box highlights the first four lines of the output, specifically the versions system, iojs, node, and unstable.

6. Instale las versiones de LTS actual y estable de Node.js. En un paso posterior, aprenderá a cambiar entre las versiones activas de Node.js con un comando `nvm`.

- Instale la versión LTS estable actual de Node.js (recomendada para aplicaciones de producción): `nvm install --lts`
- Instale la versión actual de Node.js (para probar las características y mejoras de Node.js más recientes, pero con más probabilidad de presentar problemas): `nvm install node`

7. Enumera qué versiones de Node están instaladas: `nvm ls`... ahora deberías ver las dos versiones que acabas de instalar.

```
mattwojo@MININT-LOBGCR8: ~
mattwojo@MININT-LOBGCR8:~$ nvm ls
->      v10.16.3
      v12.9.0
      system
default -> node  (-> v12.9.0)
node -> stable  (-> v12.9.0) (default)
stable -> 12.9  (-> v12.9.0) (default)
iojs -> N/A (default)
unstable -> N/A (default)
lts/* -> lts/dubnium (-> v10.16.3)
lts/argon -> v4.9.1 (-> N/A)
lts/boron -> v6.17.1 (-> N/A)
lts/carbon -> v8.16.1 (-> N/A)
lts/dubnium -> v10.16.3
mattwojo@MININT-LOBGCR8:~$
```

8. Comprueba que Node.js está instalado y la versión predeterminada actualmente con: `node --version`. Después, comprueba que también tienes npm, con: `npm --version` (También puedes usar `which node` o `which npm` para ver la ruta de acceso utilizada para las versiones predeterminadas).

9. Para cambiar la versión de Node.js que deseas usar para un proyecto, crea un directorio de proyecto `mkdir NodeTest`, escribe el directorio `cd NodeTest` y, a continuación, escribe `nvm use node`, para cambiar a la versión actual, o bien `nvm use --lts` para cambiar a la versión de LTS. También puedes usar el número específico de cualquier versión adicional que hayas instalado, como `nvm use v8.2.1`. (Para enumerar todas las versiones de Node.js disponibles, usa el comando: `nvm ls-remote`).

Si usas NVM para instalar Node.js y NPM, no es necesario usar el comando SUDO para instalar nuevos paquetes.

Administradores de versiones alternativos

Aunque nvm es actualmente el administrador de versiones más popular para Node, existen algunas alternativas que se deben tener en cuenta:

- `n` es una alternativa a `nvm` de larga duración que consigue lo mismo con comandos ligeramente diferentes y se instala a través de `npm` en lugar de con un script de Bash.
- `fpm` es un administrador de versiones más reciente, lo que exige que sea mucho más rápido que `nvm`. (También utiliza [Azure Pipelines](#)).

- [Volta](#) es un nuevo administrador de versiones del equipo de LinkedIn que notifica la velocidad mejorada y la compatibilidad multiplataforma.
- [asdf-vm](#) es una única CLI para varios lenguajes, como ike gvm, nvm, rbenv & pyenv y muchos más, todo en uno.
- [nvs](#) (comutador de versiones de Node) es una alternativa `nvm` multiplataforma con la capacidad de [integrarse con VS Code](#).

Instalar Visual Studio Code

Se recomienda usar Visual Studio Code con el [paquete de la extensión Remote-development](#) en los proyectos de Node.js. Esto divide VS Code en una arquitectura "cliente-servidor", con el cliente (la interfaz de usuario de VS Code) que se ejecuta en el sistema operativo Windows y el servidor (código, Git, complementos, etc.) que se ejecuta "remotamente" en la distribución Linux de WSL.

ⓘ Nota

Este escenario "remoto" es un poco diferente del que puede estar acostumbrado. WSL admite una distribución de Linux real en la que se ejecuta el código del proyecto, independientemente del sistema operativo Windows, pero todavía en la máquina local. La extensión Remote-WSL se conecta con el subsistema de Linux como si fuera un servidor remoto, aunque no se está ejecutando en la nube... todavía se ejecuta en la máquina local, en el entorno de WSL que ha habilitado para ejecutarse junto con Windows.

- Se admiten linting e Intellisense basados en Linux.
- El proyecto se compilará automáticamente en Linux.
- Puedes usar todas las extensiones que se ejecutan en Linux ([ES Lint](#), [NPM IntelliSense](#), [fragmentos de código de ES6](#), etc.).

Otros editores de código, como IntelliJ, Sublime Text, Brackets, etc., también funcionarán con un entorno de desarrollo Node.js de WSL 2, pero es posible que no tengan el mismo tipo de características remotas que ofrece VS Code. Estos editores de código pueden tener problemas para acceder a la ubicación de red compartida de WSL (\wsl\$\Ubuntu\home) e intentarán compilar los archivos de Linux mediante las herramientas de Windows, aunque puede que no sean las que quiere. La extensión Remote-WSL de VS Code controla automáticamente esta compatibilidad, con otros IDE que puede que necesite para configurar un servidor X. La [compatibilidad con la ejecución de aplicaciones de GUI en WSL](#) (como un IDE del editor de código) estará disponible próximamente.

Los editores de texto basados en terminal (vim, emacs y nano) también son útiles para realizar cambios rápidos directamente en la consola. En el artículo sobre cómo [elegir inteligentemente uno de estos tres editores de texto basados en terminal: Emacs, Nano o Vim](#) no solo se explican perfectamente algunas de las diferencias entre los distintos editores, sino también cómo usar cada uno de ellos.

Para instalar VS Code y la extensión Remote-WSL:

1. [Descarga e instala VS Code para Windows](#). VS Code también está disponible para Linux, pero el Subsistema de Windows para Linux no admite aplicaciones de GUI, por lo que deberemos instalarlo en Windows. Pero no te preocupes. Igualmente podrás realizar la integración con la línea de comandos y las herramientas de Linux mediante la extensión Remote-WSL.
2. Instala la [extensión Remote-WSL](#) de VS Code. De este modo, podrás usar WSL como entorno de desarrollo integrado y la compatibilidad y las rutas se controlarán automáticamente. [Más información](#).

Importante

Si ya tienes VS Code instalado, deberás asegurarte de que dispones de la [versión 1.35 de mayo](#) o una posterior a fin de poder instalar la [extensión Remote-WSL](#). No te recomendamos que uses WSL en VS Code sin la extensión Remote-WSL, ya que perderás la compatibilidad con Autocompletar, la depuración, la detección de errores, etc. Dato curioso: Esta extensión de WSL se instala en `$HOME/.vscode-server/extensions`.

Extensiones útiles de VS Code

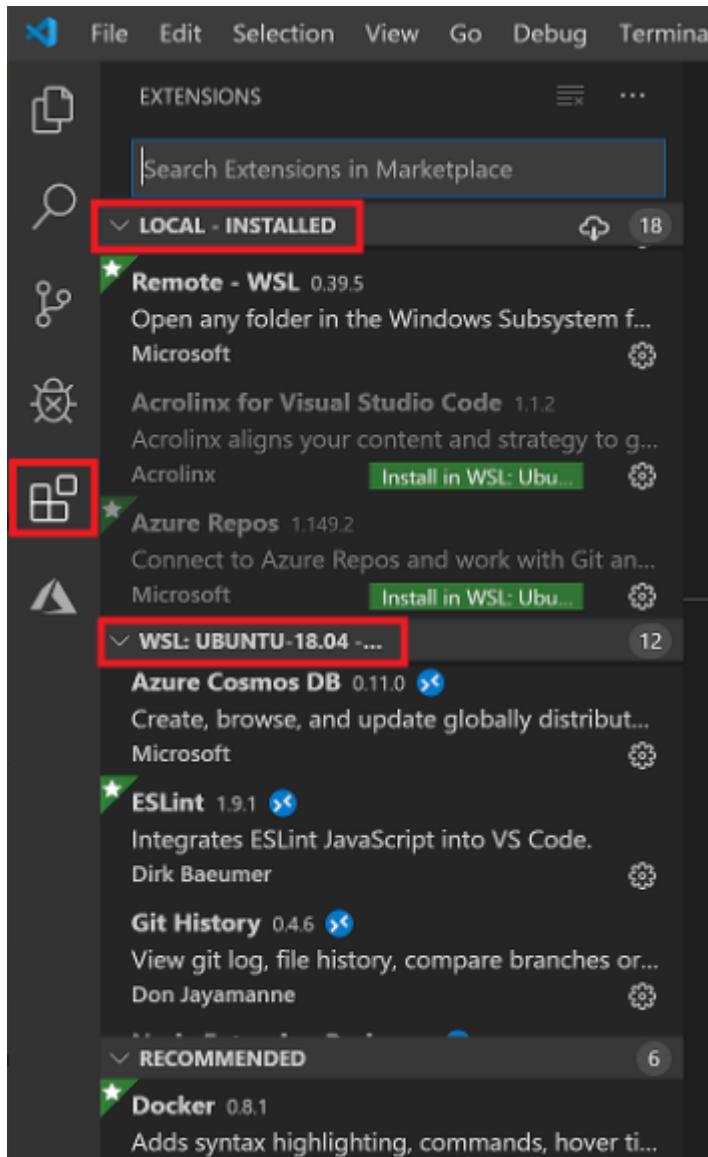
Aunque VS Code incluye muchas características para el desarrollo con Node.js listas para usar, existen algunas extensiones útiles que puedes instalar y que se encuentran disponibles en el [paquete de extensiones de Node.js](#). Instálalas todas o selecciona y elige la opción que te resulte más útil.

Para instalar el paquete de extensiones de Node.js:

1. Abre la ventana Extensiones (Ctrl+Mayús+X) en VS Code.

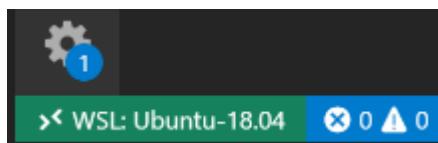
La ventana Extensiones ahora está dividida en tres secciones (porque has instalado la extensión Remote-WSL).

- "Local - Installed": las extensiones instaladas para utilizarlas con el sistema operativo Windows.
- "WSL:Ubuntu-18.04-Installed": las extensiones instaladas para utilizarlas con el sistema operativo Ubuntu (WSL).
- "Recommended": las extensiones recomendadas por VS Code según los tipos de archivo del proyecto actual.



2. En el cuadro de búsqueda de la parte superior de la ventana Extensiones, escribe:

Paquete de extensiones de Node o el nombre de cualquier extensión que estés buscando. La extensión se instalará para las instancias locales o WSL de VS Code, dependiendo de dónde tengas abierto el proyecto actual. Puedes indicarlo si seleccionas el vínculo remoto en la esquina inferior izquierda de la ventana de VS Code (en verde). Ofrecerá la opción de abrir o cerrar una conexión remota. Instala las extensiones de Node.js en el entorno "WSL:Ubuntu-18.04".



Algunas de las extensiones adicionales que puedes considerar son las siguientes:

- [JavaScript Debugger](#): una vez que hayas terminado de desarrollar en el lado servidor con Node.js, deberás desarrollar y probar el lado cliente. Esta extensión es un depurador de JavaScript basado en DAP. Depura Node.js, Chrome, Edge, WebView2, extensiones de VS Code y más.
- [Keymaps from other editors](#): estas extensiones pueden ayudarte a sentirte como en casa con tu entorno en caso de que realices la transición desde otro editor de texto (como Atom, Sublime, Vim, eMacs, Notepad++, etc.).
- [Settings Sync](#): te permite sincronizar la configuración de VS Code entre diferentes instalaciones mediante GitHub. Si trabajas en diferentes máquinas, te ayuda a mantener el entorno coherente entre ellas.

Configuración de GIT (opcional)

Si quiere configurar GIT para un proyecto de Node.js en WSL, consulte el artículo [Introducción al uso de GIT en el Subsistema de Windows para Linux](#) en la documentación de WSL.

Introducción a Linux y Bash

Artículo • 21/03/2023

Este tutorial ayudará a los nuevos a Linux a empezar a instalar y actualizar paquetes mediante la distribución ubuntu de Linux que se instala de forma predeterminada mediante WSL, así como el uso de algunos comandos básicos con la línea de comandos de Bash.

Instalación y actualización de software

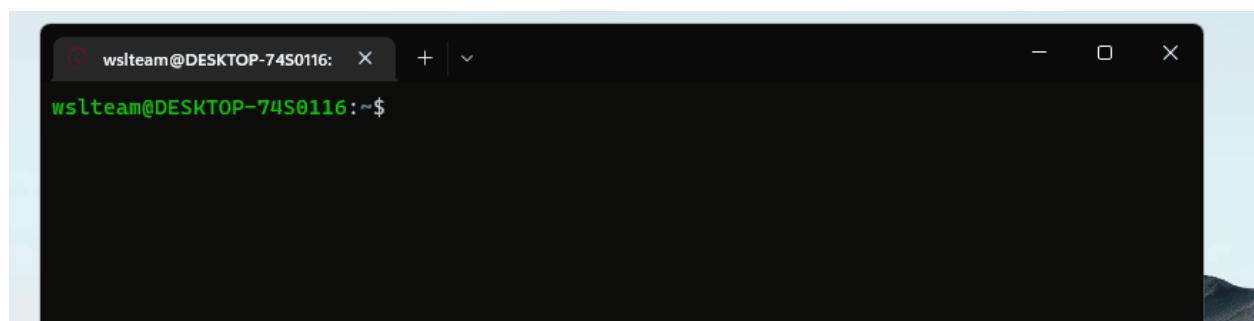
Puede instalar y actualizar programas de software directamente desde la línea de comandos mediante el administrador de paquetes preferido para la distribución que está ejecutando.

En Ubuntu, por ejemplo, actualice primero la lista de software disponible mediante la ejecución de "sudo apt update". A continuación, puede obtener directamente el software mediante el comando "sudo apt-get install" seguido del nombre del programa que desea instalar:

```
Bash  
sudo apt-get install <app_name>
```

Para actualizar los programas que ya se han instalado, puede ejecutar:

```
Bash  
sudo apt update && sudo apt upgrade
```



Sugerencia

Las distintas distribuciones de Linux suelen tener distintos administradores de paquetes y requerirán el uso de un comando de instalación específico del administrador de paquetes asociado. Por ejemplo, el administrador de paquetes principal para Arch Linux se denomina **pacman** ↗ y el comando de instalación sería `sudo pacman -S <app_name>`. El administrador de paquetes principal para OpenSuse se denomina **Zypper** ↗ y el comando de instalación sería `sudo zypper install <app_name>`. El administrador de paquetes principal para Alpine se denomina **apk** ↗ y el comando de instalación sería `sudo apk add <app_name>`. Los dos administradores de paquetes principales para las distribuciones de Red Hat, como CentOS, son **YUM** y **RPM** ↗ y un comando de instalación podría ser `sudo yum install <app_name>` o `sudo rpm -i <app_name>`. Consulte la documentación de la distribución con la que está trabajando para averiguar con qué herramientas están disponibles para instalar y actualizar software.

Trabajar con archivos y directorios

Para ver la ruta de acceso del directorio en el que está actualmente, use el comando "pwd":

```
Bash
```

```
pwd
```

Para crear un directorio, use el comando "mkdir" seguido del nombre del directorio que desea crear:

```
Bash
```

```
mkdir hello_world
```

Para cambiar los directorios, use el comando "cd" seguido del nombre del directorio al que desea ir:

```
Bash
```

```
cd hello_world
```

Para ver el contenido del directorio en el que se encuentra actualmente, escriba "ls" en la línea de comandos:

Bash

```
ls
```



A screenshot of a Windows Subsystem for Linux (WSL) terminal window titled "Bash". The window shows the command "ls" being typed at the prompt "wslteam@DESKTOP-74S0116:~\$". The terminal has a dark background with white text. The window frame is light blue, and there are standard window controls (minimize, maximize, close) at the top right.

De forma predeterminada, el comando "ls" imprimirá solo el nombre de todos los archivos y directorios. Para obtener información adicional, como la última vez que se modificó un archivo o permisos de archivo, use la marca "-l":

Bash

```
ls -l
```

Puede crear un archivo a través del comando "touch" seguido del nombre del archivo que desea crear:

Bash

```
touch hello_world.txt
```

Puede editar archivos con cualquier editor gráfico de texto descargado o la extensión VS Code Remote – WSL. Puede obtener más información sobre cómo empezar a trabajar con VS Code [aquí](#).

Si prefiere editar un archivo directamente desde la línea de comandos, deberá usar un editor de línea de comandos, como vim, emacs o nano. Muchas distribuciones vienen con uno o varios de estos programas instalados, pero siempre puede instalar estos programas siguiendo las instrucciones de instalación que se describen en la guía anterior [aqui](#).

Para editar su archivo con su método de edición preferido, simplemente ejecute el nombre del programa seguido del nombre del archivo que desea editar:

Bash

```
code hello_world.txt
```

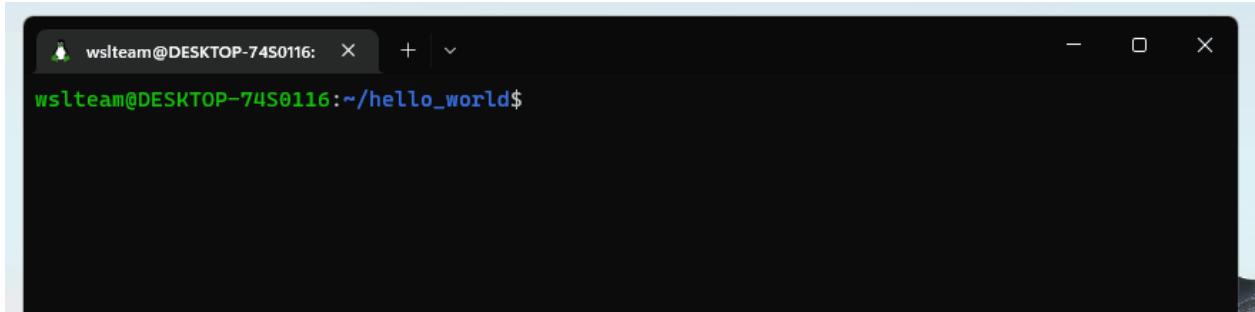
```
Bash
```

```
notepad.exe hello_world.txt
```

Para ver el contenido de un archivo en la línea de comandos, use el comando "cat" seguido del archivo que desea leer:

```
Bash
```

```
cat hello_world.txt
```



Uso de canalizaciones y operadores de redireccionamiento

Una canalización "|" redirige la salida de un comando como entrada a otro comando. Por ejemplo, lhscmd | rhscmd dirigiría la salida de lhscmd a rhscmd. Las canalizaciones se pueden usar de varias maneras de realizar rápidamente las tareas a través de la línea de comandos. A continuación se muestran solo algunos ejemplos sencillos de cómo se pueden usar las canalizaciones.

Imagine que desea ordenar rápidamente el contenido de un archivo. Tome el ejemplo de fruits.txt siguiente:

```
Bash
```

```
cat fruits.txt
```

```
Orange
```

```
Banana
```

```
Apple
```

```
Pear
```

```
Plum  
Kiwi  
Strawberry  
Peach
```

Puede ordenar rápidamente esta lista mediante una canalización:

```
Bash  
  
$ cat fruits.txt | sort  
  
Apple  
Banana  
Kiwi  
Orange  
Peach  
Pear  
Plum  
Strawberry
```

De forma predeterminada, la salida del comando 'cat' se envía a la salida estándar; sin embargo, el "|" nos permite redirigir la salida como entrada a otro comando, "ordenar".

Otro caso de uso es la búsqueda. Puede usar "grep", que es un comando útil que busca entradas para una cadena de búsqueda determinada.

```
Bash  
  
cat fruits.txt | grep P  
  
Pear  
Plum  
Peach
```

También puede usar operadores de redireccionamiento como ">" para pasar la salida a un archivo o secuencia. Por ejemplo, si desea crear un nuevo archivo de .txt con el

contenido ordenado de fruit.txt:

```
Bash
```

```
cat fruits.txt | sort > sorted_fruit.txt
```

```
Bash
```

```
$ cat sorted_fruit.txt
```

Apple

Banana

Kiwi

Orange

Peach

Pear

Plum

Strawberry

De forma predeterminada, la salida del comando sort se envía a la salida estándar; Sin embargo, el operador '>' nos permite redirigir la salida a un nuevo archivo denominado sorted_fruit.txt.

Puede usar canalizaciones y operadores de redireccionamiento de muchas maneras interesantes para completar tareas de forma más eficaz directamente desde la línea de comandos.

Contenido recomendado

- Microsoft Learn: [Introduction to Bash \(Introducción a Bash\)](#)
- Línea de comandos para principiantes ↗
- Microsoft Learn: [Introducción a WSL](#)

Trabajo en sistemas de archivos Windows y Linux

Artículo • 21/03/2023

Hay una serie de consideraciones que se deben tener en cuenta al trabajar entre sistemas de archivos Windows y Linux. En esta guía se describen algunas y se incluyen algunos ejemplos de interoperabilidad para combinar comandos basados en Windows y Linux.

Almacenamiento y rendimiento de archivos entre sistemas de archivos

Se recomienda no trabajar en los sistemas operativos con los archivos, a menos que tenga una razón específica para hacerlo. Para lograr la máxima velocidad de rendimiento, almacene los archivos en el sistema de archivos de WSL si está trabajando en una línea de comandos de Linux (Ubuntu, OpenSUSE, etc.). Si trabaja en una línea de comandos de Windows (PowerShell, símbolo del sistema), almacene los archivos en el sistema de archivos de Windows.

Por ejemplo, al almacenar los archivos de proyecto de WSL, haz lo siguiente:

- Usa el directorio raíz del sistema de archivos de Linux: `\wsl$\Ubuntu\home<user name>\Project`
- No use el directorio raíz del sistema de archivos de Windows: `/mnt/c/Users/<user name>/Project$` o `C:\Users\<user name>\Project`.

Cuando vea `/mnt/` en la ruta de acceso del archivo de una línea de comandos de WSL, significa que está trabajando desde una unidad montada. Por lo tanto, la unidad C:/ del sistema de archivos Windows (`C:\Users\<user name>\Project`) tendrá este aspecto cuando se monte en una línea de comandos de WSL: `/mnt/c/Users/<user name>/Project$`. Es posible almacenar los archivos de proyecto en una unidad montada, pero la velocidad de rendimiento mejorará si los almacena directamente en la unidad `\wsl$`.

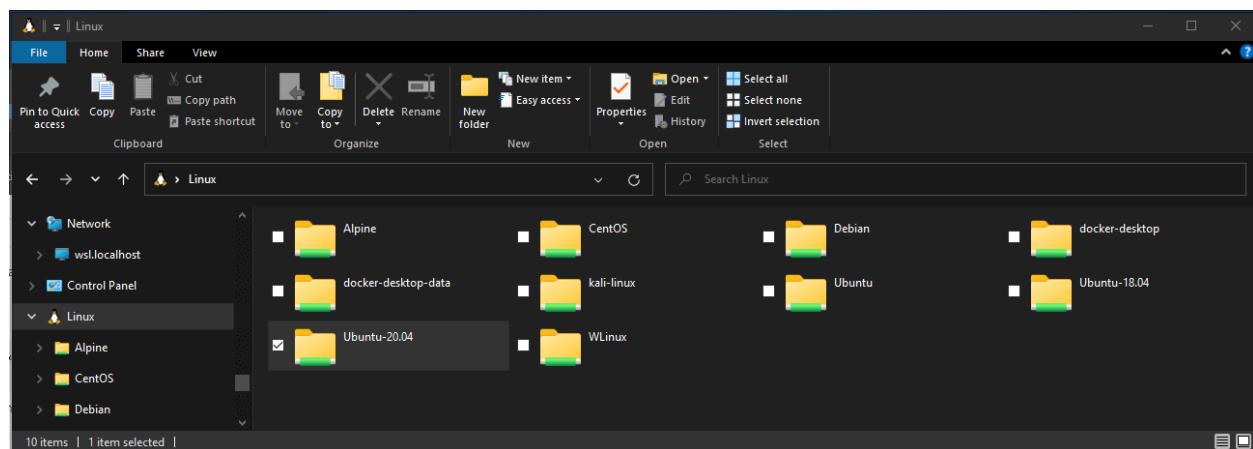
Visualización del directorio actual en el Explorador de archivos de Windows

Para ver el directorio donde se almacenan los archivos, abra el Explorador de archivos de Windows desde la línea de comandos mediante:

```
Bash  
explorer.exe .
```

También puede usar el comando `powershell.exe /c start ..`. Asegúrese de agregar el punto al final del comando para abrir el directorio actual.

Para ver todas las distribuciones de Linux disponibles y sus sistemas de archivos raíz en el Explorador de archivos de Windows escriba `\wsl$` en la barra de direcciones.



Distinción de mayúsculas y minúsculas en el nombre de archivo y el directorio

La distinción entre mayúsculas y minúsculas determina si las letras mayúsculas (FOO.txt) y minúsculas (foo.txt) se controlan como distintas (distinguen mayúsculas de minúsculas) o equivalentes (sin distinción entre mayúsculas y minúsculas) en un nombre de archivo o directorio. Los sistemas de archivos Windows y Linux controlan la distinción de mayúsculas y minúsculas de maneras diferentes: Windows no distingue mayúsculas de minúsculas y Linux sí las distingue. Obtenga más información sobre cómo ajustar la distinción de mayúsculas y minúsculas, especialmente al montar discos con WSL, en el artículo sobre el procedimiento de [ajuste de la distinción de mayúsculas y minúsculas](#).

Interoperabilidad entre comandos de Windows y Linux

Las herramientas y los comandos de Windows y Linux se pueden usar indistintamente con WSL.

- Ejecuta las herramientas de Windows (por ejemplo, notepad.exe) desde una línea de comandos de Linux (por ejemplo, Ubuntu).
- Ejecuta herramientas de Linux (por ejemplo, grep) desde una línea de comandos de Windows (por ejemplo, PowerShell).
- Comparte las variables de entorno entre Linux y Windows. (Compilación 17063 o posterior)

Ejecución de herramientas de Linux desde una línea de comandos de Windows

Ejecuta los archivos binarios de Linux desde el símbolo del sistema de Windows (CMD) o PowerShell mediante `wsl <command>` (o `wsl.exe <command>`).

Por ejemplo:

```
PowerShell  
  
C:\temp> wsl ls -la  
<- contents of C:\temp ->
```

Los archivos binarios se invocan de esta manera:

- Usan el mismo directorio de trabajo que el símbolo del sistema actual de CMD o PowerShell.
- Se ejecutan como usuario predeterminado de WSL.
- Tienen los mismos derechos administrativos de Windows que el terminal y el proceso de llamada.

El comando de Linux que sigue a `wsl` (o `wsl.exe`) se controla como cualquier comando que se ejecute en WSL. Funcionan cosas como sudo, canalizaciones y redireccionamientos de archivos.

Ejemplo de uso de sudo para actualizar la distribución de Linux predeterminada:

```
PowerShell  
  
C:\temp> wsl sudo apt-get update
```

Después de ejecutar este comando, se mostrará el nombre de usuario predeterminado de la distribución de Linux y se te pedirá la contraseña. Después de escribir la contraseña correctamente, la distribución descargará las actualizaciones.

Combinación de comandos de Windows y Linux

Estos son algunos ejemplos de la combinación de comandos de Linux y Windows mediante PowerShell.

Para usar el comando de Linux `ls -la` para enumerar archivos y el comando de PowerShell `findstr` para filtrar los resultados de las palabras que contienen "git", combina los comandos siguientes:

PowerShell

```
wsl ls -la | findstr "git"
```

Para usar el comando de PowerShell `dir` para enumerar archivos y el comando de Linux `grep` para filtrar los resultados de las palabras que contienen "git", combina los comandos siguientes:

PowerShell

```
C:\temp> dir | wsl grep git
```

Para usar el comando de Linux `ls -la` para enumerar archivos y el comando de PowerShell `> out.txt` para imprimir esa lista en un archivo de texto denominado "out.txt", combina los comandos siguientes:

PowerShell

```
C:\temp> wsl ls -la > out.txt
```

Los comandos pasados a `wsl.exe` se reenvían al proceso de WSL sin modificaciones. Las rutas de acceso de archivo deben especificarse en el formato de WSL.

Para usar el comando de Linux `ls -la` para enumerar los archivos de la ruta de acceso del sistema de archivos de Linux `/proc/cpuinfo` mediante PowerShell, utiliza el siguiente comando:

PowerShell

```
C:\temp> wsl ls -la /proc/cpuinfo
```

Para usar el comando de Linux `ls -la` para enumerar los archivos de la ruta de acceso del sistema de archivos de Windows `C:\Program Files` mediante PowerShell, utiliza el siguiente comando:

```
PowerShell
```

```
C:\temp> wsl ls -la "/mnt/c/Program Files"
```

Ejecución de herramientas de Windows desde Linux

WSL puede ejecutar herramientas de Windows directamente desde la línea de comandos de WSL mediante `[tool-name].exe`. Por ejemplo, `notepad.exe`.

Las aplicaciones que se ejecutan de esta manera tienen las siguientes propiedades:

- Conservan el directorio de trabajo como el símbolo del sistema de WSL (en la mayoría de los casos; las excepciones se explican a continuación).
- Tienen los mismos derechos de permiso que el proceso de WSL.
- Se ejecutan como el usuario activo de Windows.
- Aparecen en el administrador de tareas de Windows como si se ejecutaran directamente desde el símbolo del sistema de CMD.

Los archivos ejecutables de Windows que se ejecutan en WSL se administran de forma similar a los ejecutables nativos de Linux; las canalizaciones, los redireccionamientos e incluso los procesos en segundo plano funcionan según lo previsto.

Para ejecutar la herramientas de Windows `ipconfig.exe`, usa la herramienta de Linux `grep` para filtrar los resultados de "IPv4", y usa la herramienta de Linux `cut` para quitar los campos de columna. Desde una distribución de Linux (por ejemplo, Ubuntu), escribe lo siguiente:

```
Bash
```

```
ipconfig.exe | grep IPv4 | cut -d: -f2
```

Vamos a probar un ejemplo de combinación de comandos de Windows y Linux. Abre tu distribución de Linux (por ejemplo, Ubuntu) y crea el archivo de texto `touch foo.txt`. Ahora usa el comando de Linux `ls -la` para enumerar los archivos directos y sus detalles de creación, además de la herramienta de Windows PowerShell `findstr.exe`

para filtrar los resultados, de modo que solo se muestre el archivo `foo.txt` en los resultados:

```
Bash
```

```
ls -la | findstr.exe foo.txt
```

Las herramientas de Windows deben incluir la extensión de archivo, coincidir con el caso del archivo y ser ejecutables. Los no ejecutables, incluidos los scripts por lotes y comandos nativos de CMD como `dir`, se pueden ejecutar con el comando `cmd.exe /C`.

Por ejemplo, para enumerar el contenido del directorio C:\ del sistema de archivos de Windows, escribe:

```
Bash
```

```
cmd.exe /C dir
```

O bien usa el comando `ping` para enviar una solicitud de eco al sitio web microsoft.com:

```
Bash
```

```
ping.exe www.microsoft.com
```

Los parámetros se pasan al archivo binario de Windows sin modificar. Por ejemplo, el siguiente comando abrirá `C:\temp\foo.txt` en `notepad.exe`:

```
Bash
```

```
notepad.exe "C:\temp\foo.txt"
```

También funcionará lo siguiente:

```
Bash
```

```
notepad.exe C:\\temp\\\\foo.txt
```

Compartir variables de entorno entre Windows y WSL con WSLENV.

WSL y Windows comparten `WSLENV`, una variable de entorno especial creada para unir las distribuciones de Windows y Linux que se ejecutan en WSL.

Propiedades de la variable `WSLENV`:

- Se comparte; existe en entornos de Windows y WSL.
- Se trata de una lista de variables de entorno que se pueden compartir entre Windows y WSL.
- Puede formatear variables de entorno para que funcionen correctamente en Windows y WSL.
- Puede ayudar en el flujo entre WSL y Win32.

ⓘ Nota

Antes de la versión 17063, la única variable de entorno de Windows a la que podía tener acceso WSL era `PATH`, por lo que se podían iniciar ejecutables de Win32 desde WSL. A partir de 17063, `WSLENV` comienza a ser compatible. `WSLENV` distingue entre mayúsculas y minúsculas.

Marcas de `WSLENV`

Hay cuatro marcas disponibles en `WSLENV` que pueden influir en la forma en que se traduce la variable de entorno.

Marcas `WSLENV`:

- `/p`: traduce la ruta de acceso entre las rutas de acceso de estilo de WSL/Linux y las rutas de acceso de Win32.
- `/1`: indica que la variable de entorno es una lista de rutas de acceso.
- `/u`: indica que esta variable de entorno solo debe incluirse al ejecutar WSL desde Win32.
- `/w`: indica que esta variable de entorno solo debe incluirse al ejecutar Win32 desde WSL.

Las marcas se pueden combinar según sea necesario.

[Obtenga más información sobre `WSLENV`](#), incluidas las preguntas más frecuentes y ejemplos de configuración del valor de `WSLENV` en una concatenación de otras variables de entorno predefinidas, cada una con una barra diagonal como sufijo seguida de marcas para especificar cómo se debe traducir el valor y pasar las variables con un script. En este artículo también se incluye un ejemplo para configurar un entorno de

desarrollo con el [lenguaje de programación Go](#), configurado para compartir una variable GOPATH entre WSL y Win32.

Deshabilitación de la interoperabilidad

Los usuarios pueden deshabilitar la capacidad de ejecutar herramientas de Windows para una única sesión de WSL mediante la ejecución del siguiente comando como raíz:

Bash

```
echo 0 > /proc/sys/fs/binfmt_misc/WSLInterop
```

Para volver a habilitar los archivos binarios de Windows, sal de todas las sesiones de WSL y vuelve a ejecutar bash.exe o ejecuta el siguiente comando como raíz:

Bash

```
echo 1 > /proc/sys/fs/binfmt_misc/WSLInterop
```

La deshabilitación de la interoperabilidad no se conservará entre sesiones de WSL: la interoperabilidad se habilitará de nuevo cuando se inicie una nueva sesión.

Configuración avanzada en WSL

Artículo • 03/02/2024

Los archivos [wsl.conf](#) y [.wslconfig](#) se usan para configurar opciones de configuración avanzadas, por distribución (`wsl.conf`) y globalmente en todas las distribuciones de WSL 2 (`.wslconfig`). Esta guía abarcará cada una de las opciones de configuración, cuándo usar cada tipo de archivo, dónde almacenar el archivo, los archivos de configuración de ejemplo y las sugerencias.

¿Cuál es la diferencia entre wsl.conf y .wslconfig?

Puede configurar las opciones de las distribuciones de Linux instaladas que se aplicarán automáticamente cada vez que inicie WSL de dos maneras mediante:

- [.wslconfig](#) para configurar las **opciones globales** en todas las distribuciones instaladas que se ejecutan en WSL 2.
- [wsl.conf](#) para configurar las **opciones locales** por distribución para cada distribución de Linux que se ejecuta en WSL 1 o WSL 2.

Ambos tipos de archivo se usan para configurar las opciones de WSL, pero la ubicación donde se almacena el archivo, el ámbito de la configuración, el tipo de opciones que se pueden configurar y la versión de WSL que ejecuta la distribución afectan a qué tipo de archivo elegir.

WSL 1 y WSL 2 se ejecutan con una arquitectura diferente y afectarán a las opciones de configuración. WSL 2 se ejecuta como una máquina virtual ligera, por lo que usa la configuración de virtualización que le permite controlar la cantidad de memoria o procesadores usados (lo que puede resultar familiar si usa Hyper-V o VirtualBox).

[Comprobación de la versión de WSL que se está ejecutando.](#)

La regla de los 8 segundos para los cambios de configuración

Debe esperar hasta que el subsistema que ejecuta la distribución de Linux deje de ejecutarse por completo y se reinicie para que aparezcan las actualizaciones de configuración. Normalmente, esto tarda unos 8 segundos después de cerrar TODAS las instancias del shell de distribución.

Si inicia una distribución (por ejemplo, Ubuntu), modifica el archivo de configuración, cierra la distribución y vuelve a iniciarla, puede pensar que los cambios de configuración han entrado inmediatamente en vigor. Este no es el caso actualmente, ya que el subsistema todavía podría estar en ejecución. Debe esperar a que el subsistema se detenga antes de volver a iniciarse para dar tiempo suficiente para que se recojan los cambios. Puede comprobar si la distribución de Linux (shell) sigue ejecutándose después de cerrarla mediante PowerShell con el comando : `wsl --list --running`. Si no se está ejecutando ninguna distribución, recibirá la respuesta: "No hay distribuciones en ejecución". Ahora puede reiniciar la distribución para ver las actualizaciones de configuración aplicadas.

El comando `wsl --shutdown` es una vía rápida para reiniciar las distribuciones de WSL 2, pero cerrará todas las distribuciones en ejecución, así que úselo con prudencia. También puede usar `wsl --terminate <distroName>` para finalizar al instante una distribución específica que se está ejecutando.

wsl.conf

Configure las **opciones locales** con `wsl.conf` por distribución para cada distribución de Linux que se ejecute en WSL 1 o WSL 2.

- Almacenado en el directorio de `/etc` de la distribución como un archivo unix.
- Se usa para configurar las opciones por distribución. La configuración establecida en este archivo solo se aplicará a la distribución específica de Linux que contiene el directorio donde se almacena este archivo.
- Se puede usar para distribuciones ejecutadas por una versión, WSL 1 o WSL 2.
- Para llegar al directorio de `/etc` de una distribución instalada, use la línea de comandos de la distribución con `cd /` para acceder al directorio raíz, después `ls` para enumerar los archivos o `explorer.exe .` para verlos en el Explorador de archivos de Windows. La ruta de acceso debe tener un aspecto similar a:
`/etc/wsl.conf`.

ⓘ Nota

El ajuste de la configuración por distribución con el archivo `wsl.conf` solo está disponible en la compilación 17093 de Windows y versiones posteriores.

Opciones de configuración para `wsl.conf`

El archivo wsl.conf configura los valores por distribución. (*Para obtener la configuración global de las distribuciones de WSL 2, consulte [.wslconfig](#).*)

El archivo wsl.conf admite cuatro secciones: `automount`, `network`, `interop` y `user`. (*Modelado después de convenciones de archivo .ini, las claves se declaran en una sección, como los archivos .gitconfig*). Consulte [wsl.conf](#) para obtener información sobre dónde almacenar el archivo wsl.conf.

compatibilidad con systemd

Muchas distribuciones de Linux ejecutan "systemd" de forma predeterminada (incluido Ubuntu) y WSL ha agregado recientemente compatibilidad con este administrador de servicios o sistema para que WSL sea aún más similar al uso de sus distribuciones de Linux favoritas en una máquina sin sistema operativo. Necesitará la versión 0.67.6+ de WSL para habilitar el sistema. Compruebe la versión de WSL con el comando `wsl --version`. Si necesita actualizar, puede obtener la [versión más reciente de WSL en Microsoft Store](#). Obtenga más información en el [anuncio del blog](#).

Para habilitar systemd, abra el archivo `wsl.conf` en un editor de texto mediante `sudo` para permisos de administrador y agregue estas líneas a `/etc/wsl.conf`:

```
Bash  
[boot]  
systemd=true
```

Después tendrá que cerrar su distribución de WSL usando `wsl.exe --shutdown` de PowerShell para reiniciar sus instancias de WSL. Una vez reiniciada la distribución, systemd debería estar en ejecución. Puede confirmar con el comando: `systemctl list-unit-files --type=service`, que mostrará el estado de los servicios.

Configuración de montaje automático

Etiqueta de sección de wsl.conf: `[automount]`

[] Expandir tabla

key	value	default	notas
enabled	boolean	true	<code>true</code> hace que las unidades fijas (es decir, <code>C:/</code> o <code>D:/</code>) se monten automáticamente con DrvFs bajo <code>/mnt</code> . <code>false</code> significa que las

key	value	default	notas
			unidades no se montarán automáticamente, pero podría montarlas de forma manual o a través de <code>fstab</code> .
mountFsTab	boolean	true	<code>true</code> establece <code>/etc/fstab</code> para que se procese en el inicio de WSL. <code>/etc/fstab</code> es un archivo donde puedes declarar otros sistemas de archivos, como un recurso compartido de SMB. Por lo tanto, puedes montar estos sistemas de archivos automáticamente en WSL en el inicio.
root	cadena	<code>/mnt/</code>	Establece el directorio donde se montarán automáticamente las unidades fijas. De manera predeterminada se establece en <code>/mnt/</code> , por lo que la unidad C del sistema de archivos de Windows se monta en <code>/mnt/c/</code> . Si cambia <code>/mnt/</code> a <code>/windir/</code> , debería esperar ver la unidad C fija montada en <code>/windir/c/</code> .
options	Lista separada por comas de valores, como uid, gid, etc., consulte las opciones de montaje automático a continuación.	cadena vacía	Los valores de opción de montaje automático se enumeran a continuación y se anexan a la cadena de opciones de montaje de DrvFs predeterminada. Solo se pueden especificar opciones específicas de DrvFs.

Estas opciones de montaje automático se aplican como opciones de montaje para todas las unidades montadas automáticamente. Para cambiar las opciones solo para una unidad específica, use en su lugar el archivo `/etc/fstab`. No se admiten las opciones que el binario de montaje analizaría normalmente en una marca. Si quiere especificar explícitamente esas opciones, tiene que incluir en `/etc/fstab` cada unidad para la que quiera hacerlo.

Opciones de montaje automático

Establecer diferentes opciones de montaje para las unidades de Windows (DrvFs) puede controlar cómo se calculan los permisos de archivo para los archivos de Windows. Las siguientes opciones están disponibles:

 Expandir tabla

Clave	Descripción	Valor predeterminado
uid	Id. de usuario que se usa para el propietario de todos los archivos	Id. de usuario predeterminado de su distribución WSL (en la primera instalación, el valor predeterminado es 1000)
gid	Id. de grupo que se usa para el propietario de todos los archivos	Id. de grupo predeterminado de su distribución WSL (en la primera instalación, el valor predeterminado es 1000)
umask	Máscara octal de permisos que se van a excluir para todos los archivos y directorios	022
fmask	Máscara octal de permisos que se van a excluir para todos los archivos	000
dmask	Máscara octal de permisos que se van a excluir para todos los directorios	000
metadata	Indica si se agregan metadatos a archivos de Windows para admitir permisos del sistema Linux	deshabilitado
mayúsculas y minúsculas	Determina los directorios tratados como sensibles a mayúsculas y minúsculas y si los nuevos directorios creados con WSL tendrán la marca establecida. Consulte distinción entre mayúsculas y minúsculas para obtener una explicación detallada de las opciones. Las opciones incluyen <code>off</code> , <code>dir</code> o <code>force</code> .	<code>off</code>

De manera predeterminada, WSL establece el uid y el gid en el valor del usuario predeterminado. Por ejemplo, en Ubuntu, el usuario predeterminado es uid=1000, gid=1000. Si este valor se usa para especificar una opción gid o uid diferente, se sobrescribirá el valor de usuario predeterminado. De lo contrario, siempre se anexará el valor predeterminado.

Máscara del modo de creación de archivos de usuario (umask) establece el permiso para los archivos recién creados. El valor predeterminado es 022, solo puede escribir datos, pero cualquier usuario puede leer datos. Los valores se pueden cambiar para reflejar diferentes configuraciones de permisos. Por ejemplo, `umask=077` cambia el permiso para que sea completamente privado, ningún otro usuario puede leer ni escribir datos. Para especificar más permisos, también se pueden usar fmask (archivos) y dmask (directorios).

⚠ Nota

Las máscaras de permisos se colocan a través de una operación OR lógica antes de aplicarse a archivos o directorios.

¿Qué es DrvFs?

DrvFs es un complemento del sistema de archivos a WSL diseñado para admitir la interoperabilidad entre WSL y el sistema de archivos de Windows. DrvFs permite que WSL monte unidades con sistemas de archivos compatibles en /mnt, como /mnt/c, /mnt/d, etc. Para más información sobre cómo especificar el comportamiento predeterminado de distinción entre mayúsculas y minúsculas al montar unidades o directorios de Windows o Linux, consulte la página de [distinción entre mayúsculas y minúsculas](#).

Configuración de red

Etiqueta de sección de wsl.conf: [network]

[] Expandir tabla

key	value	default	notas
generateHosts	boolean	true	true establece WSL para generar /etc/hosts. El archivo hosts contiene una asignación estática de direcciones IP correspondientes a nombres de host.
generateResolvConf	boolean	true	true establece WSL para generar /etc/resolv.conf. resolv.conf contiene una lista de DNS que es capaz de resolver un nombre de host determinado en su dirección IP.
hostname	cadena	Nombre de host de Windows	Establece el nombre de host que se usará para la distribución de WSL.

Configuración de interoperabilidad

Etiqueta de sección de wsl.conf: [interop]

Estas opciones están disponibles en la compilación 17713 de Insider y versiones posteriores.

[\[+\] Expandir tabla](#)

key	value	default	notas
enabled	boolean	true	La configuración de esta clave determinará si WSL admitirá el inicio de procesos de Windows.
appendWindowsPath	boolean	true	Establecer esta clave determinará si WSL agregará elementos de ruta de acceso de Windows a la variable de entorno \$PATH.

Configuración de usuario

Etiqueta de sección de wsl.conf: [user]

Estas opciones están disponibles en la compilación 18980 y versiones posteriores.

[\[+\] Expandir tabla](#)

key	value	default	notas
default	cadena	Nombre de usuario inicial creado en la primera ejecución	Al establecer esta clave se especifica el usuario con el que se ejecutará cuando se inicie por primera vez una sesión de WSL.

Configuración de arranque

La configuración de arranque solo está disponible en Windows 11 y Server 2022.

Etiqueta de sección de wsl.conf: [boot]

[\[+\] Expandir tabla](#)

key	value	default	notas
comando	cadena	""	Cadena del comando que desea ejecutar cuando se inicia la instancia de WSL. Este comando se ejecuta como el usuario raíz. Por ejemplo, <code>service docker start</code> .

Archivo wsl.conf de ejemplo

El archivo de ejemplo de `wsl.conf` que aparece a continuación muestra algunas de las opciones de configuración disponibles. En este ejemplo, la distribución es Ubuntu-20.04 y la ruta de acceso del archivo es `\wsl.localhost\Ubuntu-20.04\etc\wsl.conf`.

Bash

```
# Automatically mount Windows drive when the distribution is launched
[automount]

# Set to true will automount fixed drives (C:/ or D:/) with DrvFs under the
# root directory set above. Set to false means drives won't be mounted
# automatically, but need to be mounted manually or with fstab.
enabled = true

# Sets the directory where fixed drives will be automatically mounted. This
# example changes the mount location, so your C-drive would be /c, rather than
# the default /mnt/c.
root = /

# DrvFs-specific options can be specified.
options = "metadata,uid=1003,gid=1003,umask=077,fmask=11,case=off"

# Sets the `/etc/fstab` file to be processed when a WSL distribution is
# launched.
mountFsTab = true

# Network host settings that enable the DNS server used by WSL 2. This
# example changes the hostname, sets generateHosts to false, preventing WSL
# from the default behavior of auto-generating /etc/hosts, and sets
# generateResolvConf to false, preventing WSL from auto-generating
# /etc/resolv.conf, so that you can create your own (ie. nameserver 1.1.1.1).
[network]
hostname = DemoHost
generateHosts = false
generateResolvConf = false

# Set whether WSL supports interop processes like launching Windows apps and
# adding path variables. Setting these to false will block the launch of
# Windows processes and block adding $PATH environment variables.
[interop]
enabled = false
appendWindowsPath = false

# Set the user when launching a distribution with WSL.
[user]
default = DemoUser

# Set a command to run when a new WSL instance launches. This example starts
# the Docker container service.
[boot]
command = service docker start
```

.wslconfig

Configure las **opciones globales** con `.wslconfig` en todas las distribuciones instaladas que se ejecutan en WSL.

- El archivo `.wslconfig` no existe de manera predeterminada. Debe crearlo y almacenarlo en su directorio `%UserProfile%` para aplicar estas opciones de configuración.
- Se usa para configurar las opciones globalmente en todas las distribuciones de Linux instaladas que se ejecutan como la versión WSL 2.
- Solo se puede usar para **distribuciones ejecutadas por WSL 2**. Las distribuciones que se ejecutan como WSL 1 no se verán afectadas por esta configuración, ya que no se ejecutan como una máquina virtual.
- Para acceder al directorio, en PowerShell, use `%UserProfile%` para acceder al directorio de `cd ~` principal (que suele ser el perfil de usuario, `C:\Users\<UserName>`) o puede abrir el Explorador de archivos de Windows y escribir `%UserProfile%` en la barra de direcciones. La ruta de acceso debe tener un aspecto similar a: `C:\Users\<UserName>\.wslconfig`.

WSL detectará la existencia de estos archivos, leerá el contenido y aplicará automáticamente las opciones de configuración cada vez que inicie WSL. Si falta el archivo o tiene un formato incorrecto (formato de marcado incorrecto), WSL seguirá iniciándose de forma normal sin aplicar las opciones de configuración.

Opciones de configuración para `.wslconfig`

El archivo `.wslconfig` configura las opciones globalmente para todas las distribuciones de Linux que se ejecutan con WSL 2. (*Para la configuración por distribución, consulte [wsl.conf](#)*).

Consulte [.wslconfig](#) para obtener información sobre dónde almacenar el archivo `.wslconfig`.

ⓘ Nota

La configuración de las opciones globales con `.wslconfig` solo está disponible para las distribuciones que se ejecutan como WSL 2 en la compilación 19041 de Windows y en versiones posteriores. Tenga en cuenta que es posible que tenga que ejecutar `wsl --shutdown` para apagar la máquina virtual WSL 2 y, después, reiniciar la instancia de WSL para que estos cambios surtan efecto.

Este archivo puede contener las siguientes opciones que afectan a la máquina virtual que alimenta cualquier distribución de WSL 2:

Configuración principal de WSL

Etiqueta de sección de .wslconfig: [ws12]

[+] Expandir tabla

key	value	default	notas
kernel	path	Bandeja de entrada proporcionada por el kernel compilado de Microsoft	Ruta de acceso absoluta de Windows a un kernel de Linux personalizado.
memory	tamaño	50 % de memoria total en Windows u 8 GB, lo que sea menor; en compilaciones anteriores a la 20175: 80 % de la memoria total en Windows	Cantidad de memoria que se va a asignar a la máquina virtual WSL 2.
procesadores	number	El mismo número de procesadores lógicos en Windows	Número de procesadores lógicos que se van a asignar a la máquina virtual de WSL 2.
localhostForwarding	boolean	true	Booleano que especifica si los puertos vinculados a comodín o localhost en la máquina virtual de WSL 2 deben poder conectarse desde el host a través de localhost:port.
kernelCommandLine	cadena	Blank	Argumentos adicionales de la línea de

key	value	default	notas
			comandos del kernel.
safeMode	boolean	false	Ejecute WSL en "Modo seguro", que deshabilita muchas características y está pensada para usarse para recuperar distribuciones que están en estados incorrectos. Solo está disponible para Windows 11 y WSL versión 0.66.2 y versiones posteriores.
swap	tamaño	25 % del tamaño de memoria en Windows redondeado a los GB más cercanos	Cantidad de espacio de intercambio que se va a agregar a la máquina virtual de WSL 2, 0 para ningún archivo de intercambio. El almacenamiento de intercambio es la RAM basada en disco que se usa cuando la demanda de memoria supera el límite en el dispositivo de hardware.
swapFile	path	%USERPROFILE%\AppData\Local\Temp\swap.vhdx	Ruta de acceso absoluta de Windows al disco duro

key	value	default	notas
			virtual de intercambio.
pageReporting	boolean	true	La configuración predeterminada true permite a Windows reclamar memoria no utilizada asignada a la máquina virtual de WSL 2.
guiApplications	booleano*	true	Booleano para activar o desactivar la compatibilidad con aplicaciones GUI (WSLg ↗) en WSL. Solo está disponible para Windows 11.
debugConsole	booleano*	false	Booleano para activar una ventana de consola de salida que muestra el contenido de dmesg al inicio de una instancia de distribución de WSL 2. Solo está disponible para Windows 11.
nestedVirtualization	booleano*	true	Booleano para activar o desactivar la virtualización anidada, lo que permite que otras máquinas virtuales anidadas se

key	value	default	notas
			ejecuten dentro de WSL 2. Solo está disponible para Windows 11.
vmIdleTimeout	número*	60000	Número de milisegundos que una máquina virtual está inactiva, antes de apagarla. Solo está disponible para Windows 11.
networkingMode**	string	NAT	Si el valor es <code>mirrored</code> , se activa el modo de red reflejado. Las cadenas predeterminadas o no reconocidas producen redes NAT.
firewall**	bool	true	Establecer esto en true permite que las reglas del Firewall de Windows, así como las reglas específicas del tráfico de Hyper-V, filtren el tráfico de red de WSL.
dnsTunneling**	bool	false	Cambios en la forma en que las solicitudes DNS se envían de WSL a Windows
autoProxy*	bool	false	Exige que WSL use la información del

key	value	default	notas
			proxy HTTP de Windows

Las entradas con el valor `path` deben ser rutas de acceso de Windows con barras invertidas escapadas, por ejemplo: `C:\\Temp\\\\myCustomKernel`

Las entradas con el valor `size` deben ser un tamaño seguido de una unidad, por ejemplo `8GB` o `512MB`.

Las entradas con un `*` después del tipo de valor solo están disponibles en Windows 11.

Las entradas con `**` después del tipo de valor requieren la [versión 22H2 de Windows](#) o una posterior.

Configuración experimental

Estas opciones son versiones preliminares opcionales de las características experimentales que pretendemos establecer como valor predeterminado en el futuro.

Etiqueta de sección de `.wslconfig`: `[experimental]`

 Expandir tabla

Nombre del valor	Valor	Valor predeterminado	Notas
<code>autoMemoryReclaim</code>	cadena	deshabilitado	Libera automáticamente la memoria almacenada en caché después de detectar el uso de CPU inactivo. Establézcalo en <code>gradual</code> para la versión lenta y en <code>dropcache</code> para la liberación instantánea de memoria almacenada en caché.
<code>sparseVhd</code>	bool	false	Cuando se establece en true, cualquier VHD recién creado se definirá como disperso automáticamente.
<code>useWindowsDnsCache**</code>	bool	false	Solo se aplica cuando <code>ws12.dnsTunneling</code> se establece en true. Cuando esta opción se establece en false, las solicitudes DNS de túnel desde Linux omitirán los nombres almacenados en caché en Windows

Nombre del valor	Valor	Valor predeterminado	Notas
			para colocar siempre las solicitudes en la conexión.
<code>bestEffortDnsParsing</code> **	bool	false	Solo se aplica cuando <code>ws12.dnsTunneling</code> se establece en true. Cuando se establece en true, Windows extraerá la pregunta de la solicitud DNS e intentará resolverla, ignorando los registros desconocidos.
<code>initialAutoProxyTimeout</code> *	cadena	1000	Solo se aplica cuando <code>ws12.autoProxy</code> se establece en true. Configura el tiempo (en milisegundos) que esperará WSL para recuperar la información del proxy HTTP al iniciar un contenedor WSL. Si la configuración del proxy se resuelve después de este tiempo, se debe reiniciar la instancia de WSL para usar la configuración de proxy recuperada.
<code>ignoredPorts</code> **	cadena	null	Solo se aplica cuando <code>ws12.networkingMode</code> se establece en <code>mirrored</code> . Especifica los puertos a los que se pueden enlazar las aplicaciones Linux, incluso si ese puerto se usa en Windows. Esto permite que las aplicaciones escuchen en un puerto de tráfico puramente dentro de Linux, por lo que esas aplicaciones no se bloquean aunque ese puerto se use para otros fines en Windows. Por ejemplo, WSL permitirá enlazar al puerto 53 de Linux para Docker Desktop, ya que solo escucha solicitudes del contenedor de Linux. El formato debe ser una lista separada por comas, por ejemplo: <code>3000,9000,9090</code> .
<code>hostAddressLoopback</code> **	bool	false	Solo se aplica cuando <code>ws12.networkingMode</code> se establece en <code>mirrored</code> . Cuando se establece en true, permitirá que el contenedor se conecte al host o que el host se conecte al contenedor mediante una dirección IP asignada al host. Tenga en cuenta que

Nombre del valor	Valor	Valor predeterminado	Notas
			siempre se puede usar la dirección de bucle invertido 127.0.0.1. Esta opción también permite usar todas las direcciones IP locales asignadas.

Las entradas con un * después del tipo de valor solo están disponibles en Windows 11.

Las entradas con ** después del tipo de valor requieren la [versión 22H2 de Windows](#) ↗ o una posterior.

Archivo .wslconfig de ejemplo

El archivo de ejemplo de `.wslconfig` que aparece a continuación muestra algunas de las opciones de configuración disponibles. En este ejemplo, la ruta de acceso del archivo es `C:\Users\<UserName>\.wslconfig`.

Bash

```
# Settings apply across all Linux distros running on WSL 2
[wsl2]

# Limits VM memory to use no more than 4 GB, this can be set as whole
numbers using GB or MB
memory=4GB

# Sets the VM to use two virtual processors
processors=2

# Specify a custom Linux kernel to use with your installed distros. The
default kernel used can be found at https://github.com/microsoft/WSL2-Linux-
Kernel
kernel=C:\\temp\\\\myCustomKernel

# Sets additional kernel parameters, in this case enabling older Linux base
images such as Centos 6
kernelCommandLine = vsyscall=emulate

# Sets amount of swap storage space to 8GB, default is 25% of available RAM
swap=8GB

# Sets swapfile path location, default is
%USERPROFILE%\AppData\Local\Temp\swap.vhdx
swapfile=C:\\temp\\\\wsl-swap.vhdx

# Disable page reporting so WSL retains all allocated memory claimed from
Windows and releases none back when free
pageReporting=false
```

```
# Turn on default connection to bind WSL 2 localhost to Windows localhost
localhostForwarding=true

# Disables nested virtualization
nestedVirtualization=false

# Turns on output console showing contents of dmesg when opening a WSL 2
# distro for debugging
debugConsole=true

# Enable experimental features
[experimental]
sparseVhd=true
```

Recursos adicionales

- Blog de la línea de comandos de Windows: Configuración automática de WSL ↗
- Blog de la línea de comandos de Windows: Chmod/Chown, DrvFs, metadatos de archivo ↗



Colaborar con nosotros en GitHub

El origen de este contenido se puede encontrar en GitHub, donde también puede crear y revisar problemas y solicitudes de incorporación de cambios. Para más información, consulte [nuestra guía para colaboradores](#).



Comentarios de Windows Subsystem for Linux

Windows Subsystem for Linux es un proyecto de código abierto. Seleccione un vínculo para proporcionar comentarios:

↗ Abrir incidencia con la documentación

↗ Proporcionar comentarios sobre el producto

Permisos de archivo para WSL

Artículo • 21/03/2023

En esta página se detalla cómo se interpretan los permisos de archivo de Linux en el Subsistema de Windows para Linux, en especial al acceder a recursos dentro de Windows en el sistema de archivos NT. En esta documentación se supone que tienes un conocimiento básico de la [estructura de permisos del sistema de archivos de Linux ↗](#) y del [comando umask ↗](#).

Al acceder a los archivos de Windows desde WSL, los permisos de archivo se calculan a partir de los permisos de Windows o se leen de los metadatos que WSL ha agregado al archivo. Estos metadatos no están habilitados de forma predeterminada.

Metadatos de WSL en archivos de Windows

Cuando los metadatos están habilitados como opción de montaje en WSL, se pueden agregar e interpretar atributos extendidos en archivos de Windows NT para proporcionar los permisos del sistema de archivos de Linux.

WSL puede agregar cuatro atributos extendidos de NTFS:

Nombre del atributo	Descripción
\$LXUID	Id. del propietario del usuario
\$LXGID	Id. del propietario del grupo
\$LXMOD	Modo de archivo (octales y tipo de permisos de sistemas de archivos; por ejemplo: 0777)
\$LXDEV	Dispositivo, si es un archivo de dispositivo

Además, cualquier archivo que no sea un archivo o directorio normal (por ejemplo, vínculos simbólicos, FIFO [PEPS], dispositivos de bloques, sockets de UNIX y dispositivos de caracteres) también tiene un [punto de repetición de análisis](#) de NTFS. De este modo, es mucho más rápido determinar el tipo de archivo en un directorio específico sin tener que consultar sus atributos extendidos.

Escenarios de acceso a archivos

A continuación, se muestra una descripción de cómo se determinan los permisos al acceder a los archivos de diferentes maneras con el Subsistema de Windows para Linux.

Acceso a archivos en el sistema de archivos de la unidad de Windows (DrvFS) desde Linux

Estos escenarios se producen cuando accedes a los archivos de Windows desde WSL, más probablemente a través de `/mnt/c`.

Lectura de los permisos de un archivo de Windows existente

El resultado depende de si el archivo ya tiene metadatos.

El archivo DrvFS no tiene metadatos (valor predeterminado)

Si el archivo no tiene metadatos asociados, convertimos los permisos vigentes del usuario de Windows en bits de lectura, escritura o ejecución, y los establecemos en el mismo valor para el usuario, el grupo y otros. Por ejemplo, si tu cuenta de usuario de Windows tiene acceso de lectura y ejecución, pero no tiene acceso de escritura al archivo, se mostrará como `r-x` para el usuario, el grupo y otros. Si el archivo tiene el atributo "Solo lectura" establecido en Windows, no concederemos el acceso de escritura en Linux.

El archivo tiene metadatos

Si el archivo tiene metadatos, simplemente usamos esos valores de metadatos en lugar de convertir los permisos vigentes del usuario de Windows.

Cambio de los permisos de archivo de un archivo de Windows existente mediante chmod

El resultado depende de si el archivo ya tiene metadatos existentes.

El archivo chmod no tiene metadatos (valor predeterminado)

Chmod solo tendrá un efecto. Si quitas todos los atributos de escritura de un archivo, se establecerá el atributo "Solo lectura" en el archivo de Windows, ya que se trata del mismo comportamiento de CIFS (Sistema de archivos de Internet común), que es el cliente SMB (Bloque de mensajes del servidor) en Linux.

El archivo chmod tiene metadatos

Chmod cambiará o agregará metadatos en función de los metadatos ya existentes del archivo.

Ten en cuenta que no puedes concederte un mayor acceso del que tienes en Windows, aunque los metadatos indiquen lo contrario. Por ejemplo, podrías establecer los metadatos para mostrar que tienes permisos de escritura en un archivo mediante `chmod 777`, pero si intentas acceder a ese archivo, tampoco podrás escribir en él. Esto se debe a la interoperabilidad, ya que los comandos de lectura o escritura en los archivos de Windows se enrutan a través de los permisos de usuario de Windows.

Creación de un archivo en DriveFS

El resultado depende de si los metadatos están habilitados.

Los metadatos no están habilitados (valor predeterminado)

Los permisos de Windows del archivo recién creado serán los mismos que si creas el archivo en Windows sin un descriptor de seguridad específico. Este heredará los permisos del elemento primario.

Los metadatos están habilitados

Los bits de permiso del archivo se establecen para seguir el comando umask de Linux, y el archivo se guardará con metadatos.

¿Qué usuario y grupo de Linux son propietarios del archivo?

El resultado depende de si el archivo ya tiene metadatos existentes.

El archivo de usuario no tiene metadatos (valor predeterminado)

En el escenario predeterminado, al montar las unidades de Windows de forma automática, especificamos que el identificador de usuario (UID) de cualquier archivo se establece en el identificador del usuario de WSL, y el identificador de grupo (GID) se establece en el identificador de grupo principal del usuario de WSL.

El archivo de usuario tiene metadatos

El UID y el GID especificados en los metadatos se aplican como propietario del usuario y del grupo del archivo.

Acceso a archivos de Linux desde Windows mediante `\wsl$`

El acceso a los archivos de Linux a través de `\wsl$` usará el usuario predeterminado de la distribución de WSL. Por lo tanto, cualquier aplicación de Windows que acceda a archivos de Linux tendrá los mismos permisos que el usuario predeterminado.

Creación de un archivo nuevo

El valor de umask predeterminado se aplica al crear un nuevo archivo dentro de una distribución de WSL desde Windows. Dicho valor es `022` o, en otras palabras, permite todos los permisos excepto los de escritura para grupos y otros.

Acceso a archivos en el sistema de archivos raíz de Linux desde Linux

Los archivos creados, modificados o a los que se accede en el sistema de archivos raíz de Linux siguen las convenciones estándar de Linux, como la aplicación de umask a un archivo recién creado.

Configuración de permisos de archivo

Puedes configurar los permisos de archivo dentro de las unidades de Windows mediante las opciones de montaje de `wsl.conf`. Las opciones de montaje te permiten establecer las máscaras de permisos `umask`, `dmask` y `fmask`. `umask` se aplica a todos los archivos, `dmask` se aplica solo a los directorios y `fmask` se aplica solo a los archivos. Luego, estas máscaras de permisos se someten a una operación OR lógica cuando se aplican a archivos, por ejemplo: Si tienes un valor de `umask` de `023` y uno de `fmask` de `022`, la máscara de permisos resultante para los archivos será `023`.

Más información: [Opciones de configuración por distribución con `wsl.conf`](#).

Acceso a aplicaciones de red con WSL

Artículo • 18/11/2023

Hay algunas consideraciones que debe tener en cuenta al trabajar con aplicaciones de red y WSL. De forma predeterminada, WSL usa una [arquitectura basada en NAT](#) y se recomienda probar el nuevo [modo de red reflejado](#) para obtener las características y mejoras más recientes.

Modo de red predeterminado: NAT

De forma predeterminada, WSL usa una arquitectura basada en NAT (traducción de direcciones de red) para las redes. Tenga en cuenta las siguientes consideraciones al trabajar con una arquitectura de red basada en NAT:

Acceso a las aplicaciones de red de Linux desde Windows (localhost)

Si vas a crear una aplicación de red (por ejemplo, una aplicación que se ejecute en NodeJS o SQL Server) en la distribución de Linux, puedes acceder a ella desde una aplicación de Windows (como el explorador de Internet Edge o Chrome) mediante `localhost` (como lo harías de manera habitual).

Acceso a aplicaciones de red de Windows desde Linux (dirección IP del host)

Si quieras acceder a una aplicación de red que se ejecuta en Windows (por ejemplo, una aplicación que se ejecuta en NodeJS o SQL Server) desde tu distribución de Linux (por ejemplo, Ubuntu), debes usar la dirección IP de la máquina host. Aunque no se trata de un escenario común, puedes seguir estos pasos para que funcione.

1. Obtenga la dirección IP de la máquina host mediante la ejecución de este comando desde la distribución de Linux: '`ip route show | grep -i default | awk '{ print $3}'`'
2. Conéctate a cualquier servidor de Windows con la dirección IP copiada.

La imagen siguiente muestra un ejemplo con la conexión a un servidor de Node.js que se ejecuta en Windows a través de curl.

The screenshot shows two terminal windows side-by-side. The left window is a WSL terminal on a Windows host, displaying the command `cat /etc/resolv.conf` which includes a line `nameserver 192.168.96.193` highlighted with a red box. Below it, the command `curl http://192.168.96.193:5000/` is run, returning the response "Hello World!". The right window is a PowerShell window on the host, showing the command `node app.js` running successfully with the message "Server running at http://127.0.0.1:5000/" and "Received HTTP request".

Conexión a través de direcciones IP remotas

Si se usan direcciones IP remotas para conectarse a las aplicaciones, estas se tratarán como conexiones desde la red de área local (LAN). Esto significa que tendrás que asegurarte de que la aplicación puede aceptar conexiones LAN.

Por ejemplo, es posible que tengas que enlazar la aplicación a `0.0.0.0` en lugar de a `127.0.0.1`. En el ejemplo de una aplicación de Python que usa Flask, esto se puede hacer con el comando `app.run(host='0.0.0.0')`. Tenga en cuenta la seguridad al realizar estos cambios, ya que esto permitirá las conexiones desde la LAN.

Acceso a la distribución de WSL 2 desde la red de área local (LAN)

Al usar una distribución de WSL 1, si el equipo está configurado para ser accesible desde la LAN, también se podrá acceder a las aplicaciones que se ejecuten en WSL en la LAN.

Esto no ocurre de forma predeterminada en WSL 2. WSL 2 tiene un adaptador Ethernet virtualizado con su propia dirección IP única. Actualmente, para habilitar este flujo de trabajo, tendrás que seguir los mismos pasos que para una máquina virtual normal. (Estamos buscando maneras de mejorar esta experiencia).

A continuación se muestra un ejemplo de uso del comando [Netsh interface portproxy](#) Windows para añadir un proxy de puerto que escucha en el puerto de su host y conecta ese proxy de puerto a la dirección IP para la VM WSL 2.

```
PowerShell

netsh interface portproxy add v4tov4 listenport=<yourPortToForward>
listenaddress=0.0.0.0 connectport=<yourPortToConnectToInWSL> connectaddress=
(wsl hostname -I)
```

En este ejemplo, deberá actualizar `<yourPortToForward>` a un número de puerto, por ejemplo `listenport=4000`. `listenaddress=0.0.0.0` significa que las solicitudes entrantes se aceptarán desde CUALQUIER dirección IP. La dirección de escucha especifica la dirección IPv4 para la que se va a escuchar y se puede cambiar a valores que incluyen: dirección IP, nombre NetBIOS del equipo o nombre DNS del equipo. Si no se especifica una dirección, el valor predeterminado es el equipo local. Debe actualizar el valor `<yourPortToConnectToInWSL>` a un número de puerto al que desee que WSL se conecte, por ejemplo `connectport=4000`. Por último, el valor `connectaddress` debe ser la dirección IP de la distribución de Linux instalada a través de WSL 2 (la dirección de máquina virtual de WSL 2), que se puede encontrar escribiendo el comando: `wsl.exe hostname -I`.

Por lo tanto, este comando puede tener un aspecto similar al siguiente:

PowerShell

```
netsh interface portproxy add v4tov4 listenport=4000 listenaddress=0.0.0.0 connectport=4000 connectaddress=192.168.101.100
```

Para obtener la dirección IP, use:

- `wsl hostname -I` para la dirección IP de la distribución de Linux instalada a través de WSL 2 (la dirección de máquina virtual de WSL 2)
- `cat /etc/resolv.conf` para la dirección IP de la máquina Windows como se ve en WSL 2 (la máquina virtual WSL 2)

Con `listenaddress=0.0.0.0` se escuchará en todos los [puertos IPv4 ↗](#).

ⓘ Nota

El uso de "i" en minúsculas con el comando `hostname` generará un resultado diferente al de usar una "I" mayúscula. `wsl hostname -i` es la máquina local (127.0.1.1 es una dirección de diagnóstico de marcador de posición), mientras que `wsl hostname -I` devolverá la dirección IP de la máquina local tal como se ve en otras máquinas y se debe usar para identificar la `connectaddress` de distribución de Linux que se ejecuta a través de WSL 2.

Acceso IPv6

- `wsl hostname -i` para la dirección IP de la distribución de Linux instalada a través de WSL 2 (la dirección de máquina virtual de WSL 2)
- `ip route show | grep -i default | awk '{ print $3}'` para la dirección IP de la máquina Windows como se ve en WSL 2 (la máquina virtual WSL 2)

Con `listenaddress=0.0.0.0` se escuchará en todos los [puertos IPv4](#).

Redes en modo reflejo

Puede [establecer networkingMode=mirrored en \[wsl2\] en el archivo.wslconfig](#) para habilitar las redes en modo reflejo. Al habilitar este cambio de WSL en una arquitectura de red completamente nueva que tiene el objetivo de "crear reflejo" de las interfaces de red que tiene en Windows en Linux, para agregar nuevas características de red y mejorar la compatibilidad.

Estas son las ventajas actuales para habilitar este modo:

- Compatibilidad de IPv6
- Conectar a servidores Windows desde Linux mediante la dirección `localhost 127.0.0.1`
- Compatibilidad mejorada de redes para VPN
- Compatibilidad con multidifusión
- Conectar a WSL directamente desde la red de área local (LAN)

ⓘ Nota

Ejecute el siguiente comando en la ventana de PowerShell con privilegios de administrador para [configurar la configuración del firewall de Hyper-V](#) para permitir conexiones entrantes: `Set-NetFirewallHyperVVMSetting -Name '{40E0AC32-46A5-438A-A0B2-2B479E8F2E90}' -DefaultInboundConnection Allow 0 New-NetFirewallHyperVRule -Name MyWebServer -DisplayName "My Web Server" -Direction Inbound -VMCreatorId "{40E0AC32-46A5-438A-A0B2-2B479E8F2E90}" -Protocol TCP -LocalPorts 80.`

Este nuevo modo aborda los problemas de red detectados con una arquitectura basada en NAT (traducción de direcciones de red). Busque problemas conocidos o envíe comentarios sobre los errores identificados en el [repositorio de productos WSL en GitHub](#).

Tunelización de DNS

Si se establece [dnsTunneling=true en \[wsl2\] en el archivo .wslconfig](#), WSL usa una característica de virtualización para responder a las solicitudes DNS desde WSL, en lugar de solicitarlas a través de un paquete de red. Esta característica está destinada a mejorar la compatibilidad con VPN y otras complejas redes configuradas.

Proxy automático

Al establecer [autoProxy=true en \[wsl2\] en el archivo .wslconfig](#) se aplica WSL para usar la información del proxy HTTP de Windows. Si ya tiene un proxy configurado en Windows, habilitar esta característica hará que ese proxy también se establezca automáticamente en WSL.

WSL y firewall

En las máquinas que ejecutan Windows 11 22H2 y versiones posteriores, con WSL 2.0.9 y versiones posteriores, la característica de firewall de Hyper-V se activará de forma predeterminada. Esto garantizará que:

- Consulte [Firewall de Windows Defender con seguridad avanzada](#) para obtener más información sobre las características de seguridad de Windows que se aplicarán automáticamente a WSL.
- Consulte [Configuración del firewall de Hyper-V](#) para obtener más información sobre cómo aplicar estas reglas y configuraciones tanto localmente como a través de herramientas en línea como Intune.

Uso de systemd para administrar servicios de Linux con WSL

Artículo • 19/07/2024

Subsistema de Windows para Linux (WSL) ahora admite systemd, un sistema de inicialización y un administrador de servicios que usan muchas distribuciones populares de Linux, como Ubuntu, Debian, etc. ([¿Qué es systemd?](#)).

El valor predeterminado del sistema de inicialización ha cambiado recientemente de SystemV, con [systemd ahora el valor predeterminado para la versión actual de Ubuntu](#) que se instalará con el [wsl --install](#) comando predeterminado. Las distribuciones de Linux distintas de la versión actual de Ubuntu pueden seguir usando el init de WSL, similar al de SystemV. Para cambiar a systemd, vea [Habilitación del sistema](#).

¿Qué es systemd en Linux?

Según [systemd.io](#): "systemd es un conjunto de bloques de creación básicos para un sistema Linux. Proporciona un sistema y administrador de servicios que se ejecuta como PID 1 e inicia el resto del sistema".

Principalmente un sistema de inicialización y un administrador de servicios, systemd incluye características como el inicio a petición de demonios, el mantenimiento de puntos de montaje y montaje automático, la compatibilidad con instantáneas y el seguimiento de procesos mediante grupos de control de Linux.

La mayoría de las distribuciones principales de Linux ahora ejecutan systemd, por lo que habilitarlo en WSL acerca aún más la experiencia al uso de Linux sin sistema operativo. Consulte el [anuncio de vídeo con demostraciones](#) systemd o [ejemplos de uso de systemd](#) a continuación para obtener más información sobre lo que systemd tiene que ofrecer.

¿Cómo habilitar systemd?

Systemd es [ahora el valor predeterminado para la versión actual de Ubuntu](#) que se instalará con el comando [wsl --install](#) predeterminado .

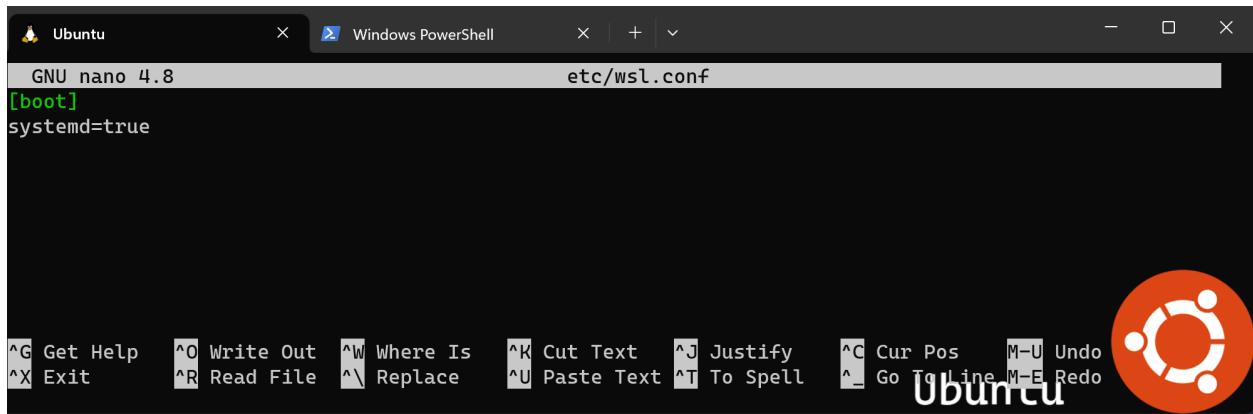
Para habilitar systemd para cualquier otra distribución de Linux que se ejecute en WSL 2 (cambiando el valor predeterminado de mediante el init de systemv):

1. Asegúrese de que la versión de WSL sea 0.67.6 o posterior. (Para comprobarlo, ejecute `wsl --version`. Para actualizar, ejecute `wsl --update` o [descargue la versión más reciente en Microsoft Store](#).)
2. Abra una línea de comandos para la distribución de Linux y escriba `cd /` para acceder al directorio raíz y, a continuación `ls` para enumerar los archivos. Verá un directorio denominado "etc" que contiene el archivo de configuración de WSL para la distribución. Abra este archivo para que pueda realizar una actualización con el editor de texto Nano escribiendo: `nano /etc/wsl.conf`.
3. Agregue estas líneas en el archivo `wsl.conf` que ahora tiene abierto para cambiar el init usado para systemd:

```
Bash

[boot]
systemd=true
```

4. Salga del editor de texto Nano (Ctrl + X, seleccione Y para guardar el cambio). A continuación, deberá cerrar la distribución de Linux. Puede usar el comando `wsl.exe --shutdown` en PowerShell para reiniciar todas las instancias de WSL.



Una vez reiniciada la distribución de Linux, systemd estará en ejecución. Puede confirmar mediante el comando: `systemctl list-unit-files --type=service`, que mostrará el estado de los servicios asociados a la distribución de Linux.

Obtenga más información sobre [configuración avanzada en WSL](#), incluida la diferencia entre los archivos de configuración de `wsl.conf` (específicos de la distribución) y `.wslconfig` (global), cómo actualizar la configuración de montaje automático, etc.

Vídeo de demostración de SystemD

Microsoft se ha asociado con Canonical para ofrecer soporte técnico a systemd en WSL. Craig Loewen (PM para WSL en Microsoft) y Oliver Smith (PM para Ubuntu en WSL en Canonical) anuncian compatibilidad con systemd y muestran algunas demostraciones de lo que habilita.

[https://learn-video.azurefd.net/vod/player?show=tabs-vs-spaces&ep=wsl-partnering-with-canonical-to-support-systemd&locale=es-es&embedUrl=%2Fwindows%2Fwsl%2Fsystemd ↗](https://learn-video.azurefd.net/vod/player?show=tabs-vs-spaces&ep=wsl-partnering-with-canonical-to-support-systemd&locale=es-es&embedUrl=%2Fwindows%2Fwsl%2Fsystemd)

- [Anuncio del blog de compatibilidad con systemd ↗](#)
- [Los tutoriales de Oliver basados en estas demostraciones en el blog de Ubuntu ↗](#) incluyen "Uso de snap para crear una instancia nextcloud en minutos en WSL", "Administración de los proyectos web con LXD" y "[Ejecución de un bot de eco de .Net como un servicio systemd en Ubuntu WSL](#)" ↗
- [Demostración de microk8s de Craig en GitHub ↗](#)

Ejemplos de systemd

Algunos ejemplos de aplicaciones Linux que dependen de systemd son:

- [snap ↗](#): un sistema de empaquetado de software e implementación desarrollado por Canonical para sistemas operativos que usan el kernel de Linux y el sistema de inicialización systemd. Los paquetes se denominan "snaps", la herramienta de línea de comandos para compilar snaps se denomina "Snapcraft", el repositorio central donde se pueden descargar o instalar los snaps se denomina "Snap Store" y el demonio necesario para ejecutar snaps (descargar desde la tienda, montar en su lugar, limitar y ejecutar aplicaciones fuera de ellas) se denomina "snapd". Todo el sistema se conoce a veces como "snappy". Intente ejecutar el comando: `snap install spotify`.
- [microk8s ↗](#): una instancia de Kubernetes de producción mínima, de código abierto y de bajo nivel, que automatiza la implementación, el escalado y la administración de aplicaciones en contenedores. Siga las instrucciones para [instalar MicroK8s en WSL2 ↗](#), consulte el [Tutorial de introducción ↗](#) o vea el vídeo en [Kubernetes en Windows con MicroK8s y WSL 2 ↗](#).
- [systemctl ↗](#): una utilidad de línea de comandos que se usa para controlar e inspeccionar el sistema y para ayudarle a interactuar con los servicios en la distribución de Linux. Pruebe el comando: `systemctl list-units --type=service` para ver qué servicios están disponibles y su estado.

Algunos tutoriales relacionados que muestran formas de usar systemd:

- [Información y uso de Systemd ↗](#)
- [Esenciales de Systemd: Trabajar con los servicios, las unidades y el diario ↗](#)
- [Procedimientos para los procesos de espacio aislado con Systemd en Ubuntu 20.04 ↗](#)

¿Cómo afecta la habilitación de systemd a la arquitectura de WSL?

La habilitación del soporte técnico de systemd requiere cambios en la arquitectura de WSL. Como systemd requiere PID 1, el proceso de inicialización de WSL iniciado dentro de la distribución de Linux se convierte en un proceso secundario del sistema. Dado que el proceso de inicialización de WSL es responsable de proporcionar la infraestructura para la comunicación entre los componentes de Linux y Windows, cambiar esta jerarquía requiere replantearse algunas de las suposiciones realizadas con el proceso de inicialización de WSL. Se tuvieron que realizar modificaciones adicionales para garantizar un apagado limpio (ya que el apagado está controlado ahora por systemd) y para tener compatibilidad con [WSLg](#), el componente de WSL que ejecuta interfaces gráficas de usuario (GUI) de Linux o las aplicaciones de Linux que se muestran en ventanas en lugar de la línea de comandos.

También es importante tener en cuenta que, con estos cambios, los servicios con sistema NO mantendrán activa la instancia de WSL. La instancia de WSL permanecerá activa de la misma manera que hizo antes de esta actualización, sobre la que puede consultar más información en esta [entrada de blog sobre soporte técnico de tareas en segundo plano de 2017 ↗](#).

 Colaborar con nosotros en
GitHub

El origen de este contenido se puede encontrar en GitHub, donde también puede crear y revisar problemas y solicitudes de incorporación de cambios. Para más información, consulte [nuestra guía para colaboradores](#).



Comentarios de Windows Subsystem for Linux

Windows Subsystem for Linux es un proyecto de código abierto.

Seleccione un vínculo para proporcionar comentarios:

 [Abrir incidencia con la documentación](#)

 [Proporcionar comentarios sobre el producto](#)

Entorno empresarial: Configuración del Subsistema de Windows para Linux para tu empresa

Artículo • 22/11/2023

Esta guía está pensada para administradores de TI o analistas de seguridad responsables de configurar entornos de trabajo empresariales con el objetivo de distribuir software entre varias máquinas y mantener un nivel coherente de configuración de seguridad en esas máquinas de trabajo.

Muchas empresas usan [Microsoft Intune](#) y [Microsoft Defender](#) para administrar esta configuración de seguridad. Sin embargo, la configuración de WSL y el acceso a distribuciones de Linux en este contexto requiere una configuración específica. Esta guía proporciona lo que necesita saber para habilitar el uso seguro de Linux con WSL en un entorno empresarial.

Configuración empresarial recomendada con Microsoft Defender para punto de conexión, Intune y controles de redes avanzadas

Hay varias maneras de configurar un entorno empresarial seguro, pero se recomienda lo siguiente para configurar un entorno seguro que use WSL.

Requisitos previos

Para empezar, asegúrese de que todos los dispositivos empresariales tengan instaladas las siguientes versiones mínimas:

- Windows 10 22H2 o posterior, o Windows 11 22H2 o posterior
 - Las características avanzadas de red solo están disponibles en Windows 11 22H2 o versiones posteriores.
- [WSL versión 2.0.9](#) o posterior
 - Puede comprobar la versión de WSL ejecutando `wsl --version`.

Habilitación de la integración de Microsoft Defender para punto de conexión (MDE)

[Microsoft Defender para punto de conexión](#) es una plataforma de seguridad empresarial para puntos de conexión diseñada para evitar, detectar, investigar y responder a amenazas avanzadas. MDE ahora se integra con WSL como [complemento WSL](#), lo que permite a los equipos de seguridad ver y supervisar continuamente los eventos de seguridad en todas las distribuciones de WSL en ejecución con Defender para punto de conexión, a la vez que afecta al rendimiento mínimo en las cargas de trabajo del desarrollador.

Vea [Microsoft Defender para punto de conexión complemento para WSL](#) para obtener más información sobre cómo empezar.

Configuración de las opciones recomendadas con Intune

[Microsoft Intune](#) es una solución de administración de puntos de conexión basada en la nube. Administra el acceso de los usuarios a los recursos de la organización y simplifica la administración de aplicaciones y dispositivos en todos los dispositivos, incluidos los dispositivos móviles, los equipos de escritorio y los puntos de conexión virtuales. Puede usar Microsoft Intune para administrar dispositivos dentro de su organización, que ahora también incluye la administración del acceso a WSL y su configuración de seguridad clave.

Vea [Configuración de Intune para WSL](#) para obtener instrucciones sobre el uso de Intune para administrar WSL como componente de Windows y la configuración recomendada.

Uso de características y controles avanzados de red

A partir de Windows 11 22H2 y WSL 2.0.9 o posterior, las reglas de firewall de Windows se aplicarán automáticamente a WSL. Esto garantiza que las reglas de firewall establecidas en el host de Windows se aplicarán automáticamente a todas las distribuciones de WSL de forma predeterminada. Para obtener instrucciones sobre cómo personalizar la configuración del firewall para WSL, visite [Configuración del firewall de Hyper-V](#).

Además, se recomienda configurar [las opciones en \[wsl2\]](#) el [.wslconfig](#) archivo para que se adapten a su escenario empresarial específico.

Redes en modo reflejado

`networkingMode=mirrored` habilita las redes en modo reflejado. Este nuevo modo de red mejora la compatibilidad con entornos de red complejos, especialmente VPN y mucho

más, así como la adición de compatibilidad con nuevas características de red que no están disponibles en el modo NAT predeterminado, como IPv6.

Tunelización de DNS

`dnsTunneling=true` cambia la forma en que WSL obtiene información de DNS. Esta configuración mejora la compatibilidad en diferentes entornos de red y usa características de virtualización para obtener información de DNS en lugar de un paquete de red. Se recomienda activarlo si experimenta algún problema de conectividad y puede resultar especialmente útil al usar VPN, configuración avanzada del firewall, etc.

Proxy automático

`autoProxy=true` exige que WSL use la información del proxy HTTP de Windows. Se recomienda activar esta configuración al usar un proxy en Windows, ya que hará que ese proxy se aplique automáticamente a las distribuciones de WSL.

Creación de una imagen de WSL personalizada

Lo que se conoce comúnmente como una "imagen" no es más que una instantánea del software y sus componentes guardada en un archivo. En el caso del Subsistema de Windows para Linux, la imagen constaría del subsistema, sus distribuciones y el software y los paquetes instalados en la distribución.

Para empezar a crear la imagen de WSL, primero [instale el Subsistema de Windows para Linux](#).

Una vez instalado, use Microsoft Store para Empresas para descargar e instalar la distribución de Linux adecuada para usted. Crear una cuenta con [Microsoft Store para Empresas](#).

Exportación de la imagen de WSL

Ejecute `wsl --export <Distro> <FileName>` para exportar la imagen de WSL personalizada. Esto encapsulará la imagen en un archivo .tar y la preparará para su distribución en otras máquinas. Puede [crear distribuciones personalizadas, como CentOS, RedHat y mucho más mediante la guía de distribución personalizada](#).

Distribución de la imagen de WSL

Ejecute `wsl --import <Distro> <InstallLocation> <FileName>` para distribuir la imagen de WSL desde un recurso compartido o un dispositivo de almacenamiento. Esto importará el archivo .tar especificado como una nueva distribución.

Actualización y revisión de distribuciones y paquetes de Linux

Se recomienda usar las herramientas de administrador de configuración de Linux para supervisar y administrar el espacio de usuario de Linux. Hay una gran variedad de administradores de configuración de Linux entre los que elegir. Vea esta entrada de blog sobre [Running Puppet rápidamente en WSL 2 ↗](#).

Acceso al sistema de archivos de Windows

Cuando un binario de Linux dentro de WSL accede a un archivo de Windows, lo hace con los permisos de usuario del usuario de Windows que ejecutó `wsl.exe`. Por lo tanto, aunque un usuario de Linux tenga acceso raíz dentro de WSL, no podrá realizar operaciones de nivel de administrador de Windows en Windows si el usuario de Windows no tiene esos permisos. Con respecto al acceso ejecutable de Windows y archivos de Windows desde WSL, ejecutar un shell como `bash` tiene los mismos permisos de nivel de seguridad que ejecutar `powershell` desde Windows que ese usuario.

Compatible

- Uso interno compartido de una imagen aprobada mediante `wsl --import` y `wsl --export`
- Creación de una distribución de WSL propia para su empresa con el [repositorio del iniciador de distribuciones de WSL ↗](#)
- Supervisión de eventos de seguridad dentro de distribuciones de WSL mediante Microsoft Defender para punto de conexión (MDE)
- Usar la configuración del firewall para controlar las redes en WSL (incluye la sincronización de la configuración del firewall de Windows con WSL)
- Controlar el acceso a WSL y su configuración de seguridad clave con Intune o directiva de grupo

Esta es una lista de las características que aún no admitimos, pero estamos investigando.

No se admiten actualmente.

A continuación se muestra una lista de las características solicitadas con más frecuencia que actualmente no se admiten en WSL. Estas solicitudes se encuentran en nuestro trabajo pendiente y estamos investigando cómo agregarlas.

- Administración de actualizaciones y revisiones de las distribuciones y los paquetes de Linux mediante herramientas de Windows
- Actualización del contenido de distribuciones de WSL mediante Windows Update
- Control de las distribuciones a las que pueden acceder los usuarios de la empresa
- Control del acceso raíz para los usuarios

Importación de cualquier distribución de Linux que se va a usar con WSL

Artículo • 07/10/2023

Puede utilizar cualquier distribución Linux dentro del Subsistema de Windows para Linux (WSL), aunque no esté disponible en la [Microsoft Store](#), importándola con un archivo tar.

En este artículo se muestra cómo importar la distribución de Linux, [CentOS](#), para su uso con WSL mediante la obtención de su archivo tar mediante un contenedor de Docker. Este proceso se puede aplicar para importar cualquier distribución de Linux.

Obtener un archivo tar para la distribución

En primer lugar, deberá obtener un archivo tar que contenga todos los archivos binarios de Linux para la distribución.

Puede obtener un archivo tar de varias maneras, dos de las cuales incluyen:

- Descargue un archivo tar proporcionado. Puede encontrar un ejemplo de Alpine en la sección "Mini root Filesystem" del sitio de [descargas de Alpine Linux](#).
- Busque un contenedor de distribución de Linux y exporte una instancia como un archivo tar. En el ejemplo siguiente se mostrará este proceso mediante el [contenedor CentOS](#).

Obtención de un archivo tar para el ejemplo de CentOS

En este ejemplo, usaremos Docker dentro de una distribución de WSL para obtener el archivo tar para CentOS.

Requisitos previos

- Debe tener [WSL habilitado con una distribución de Linux instalada que ejecute WSL 2](#).
- Debe tener [Docker Desktop para Windows instalado con el motor WSL 2 habilitado e integración activado](#). Consulte el [contrato de licencia de Docker Desktop](#) para obtener actualizaciones sobre los términos de uso.

Exportación del archivo tar desde un contenedor

1. Abra la línea de comandos (Bash) para una distribución de Linux que ya haya instalado desde Microsoft Store (Ubuntu en este ejemplo).

2. Inicie el servicio "docker":

```
Bash  
sudo service docker start
```

3. Ejecute el contenedor CentOS dentro de Docker:

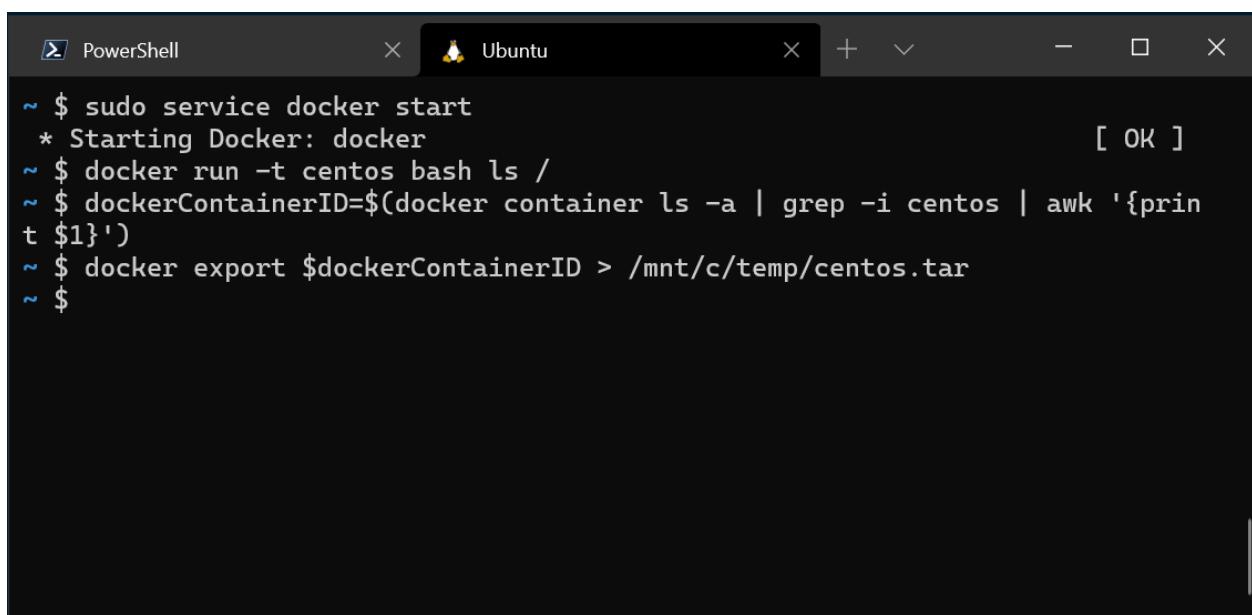
```
Bash  
docker run -t centos bash ls /
```

4. Tome el identificador de contenedor de CentOS mediante grep y awk:

```
Bash  
dockerContainerID=$(docker container ls -a | grep -i centos | awk '{print $1}')
```

5. Exporte el identificador del contenedor a un archivo tar en la unidad c montada:

```
Bash  
docker export $dockerContainerID > /mnt/c/temp/centos.tar
```



```
PowerShell      ×   Ubuntu      ×   +   ▾   -   □   ×  
~ $ sudo service docker start  
* Starting Docker: docker [ OK ]  
~ $ docker run -t centos bash ls /  
~ $ dockerContainerID=$(docker container ls -a | grep -i centos | awk '{print $1}')  
~ $ docker export $dockerContainerID > /mnt/c/temp/centos.tar  
~ $
```

Este proceso exporta el archivo tar CentOS desde el contenedor de Docker para poder importarlo para usarlo localmente con WSL.

Importación del archivo tar en WSL

Una vez que tenga un archivo tar listo, puede importarlo mediante el comando : `wsl --import <Distro> <InstallLocation> <FileName>`.

Ejemplo de importación de CentOS

Para importar el archivo tar de distribución de CentOS en WSL:

1. Abra PowerShell y asegúrese de que tiene una carpeta creada donde desea almacenar la distribución.

```
PowerShell

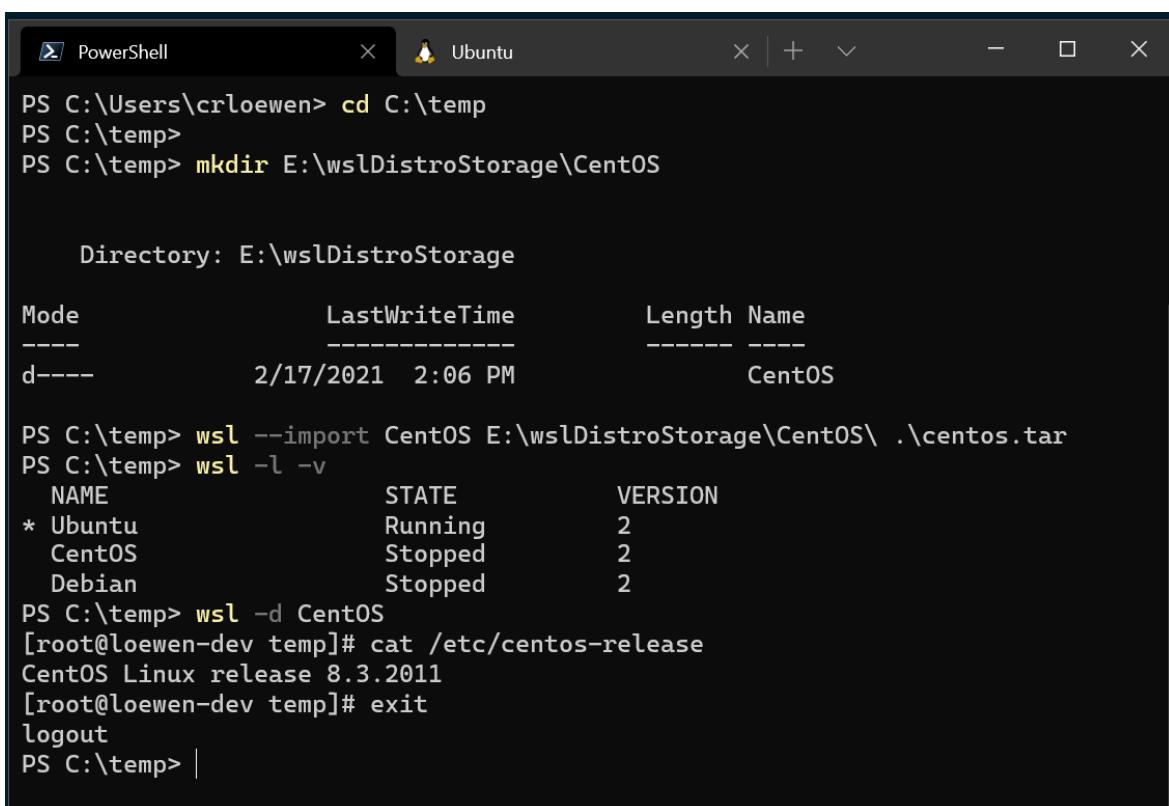
cd C:\temp
mkdir E:\wslDistroStorage\CentOS
```

2. Use el comando `wsl --import <DistroName> <InstallLocation> <InstallTarFile>` para importar el archivo tar.

```
PowerShell

wsl --import CentOS E:\wslDistroStorage\CentOS .\centos.tar
```

3. Use el comando `wsl -l -v` para comprobar qué distribuciones ha instalado.



```
PowerShell
Ubuntu

PS C:\Users\crloewen> cd C:\temp
PS C:\temp>
PS C:\temp> mkdir E:\wslDistroStorage\CentOS

Directory: E:\wslDistroStorage

Mode                LastWriteTime         Length Name
----                -----          ----
d----       2/17/2021  2:06 PM           2 CentOS

PS C:\temp> wsl --import CentOS E:\wslDistroStorage\CentOS\ .\centos.tar
PS C:\temp> wsl -l -v
  NAME                  STATE        VERSION
* Ubuntu                Running      2
  CentOS                Stopped      2
  Debian                Stopped      2
PS C:\temp> wsl -d CentOS
[root@loewen-dev temp]# cat /etc/centos-release
CentOS Linux release 8.3.2011
[root@loewen-dev temp]# exit
logout
PS C:\temp> |
```

4. Por último, use el comando `wsl -d CentOS` para ejecutar la distribución de CentOS Linux recién importada.

Agregar componentes específicos de WSL como un usuario predeterminado

De forma predeterminada, al usar `--import`, siempre se inicia como usuario raíz. Puede configurar su propia cuenta de usuario, pero tenga en cuenta que el proceso de configuración variará ligeramente en función de cada distribución de Linux diferente.

Para configurar la cuenta de usuario con la distribución de CentOS que acabamos de importar, primero abra PowerShell y arranque en CentOS mediante el comando :

PowerShell

```
wsl -d CentOS
```

A continuación, abra la línea de comandos de CentOS. Use este comando para instalar las herramientas de configuración de sudo y contraseña en CentOS, crear una cuenta de usuario y establecerla como el usuario predeterminado. En este ejemplo, el nombre de usuario será "caloewen".

ⓘ Nota

Querrá agregar el nombre de usuario al archivo sudoers para que el usuario pueda usar sudo. El comando `adduser -G wheel $myUsername` añade el usuario `myUsername` al grupo de ruedas. A los usuarios del grupo de ruedas se les conceden automáticamente privilegios de sudo y se pueden realizar tareas que requieren permisos elevados.

Bash

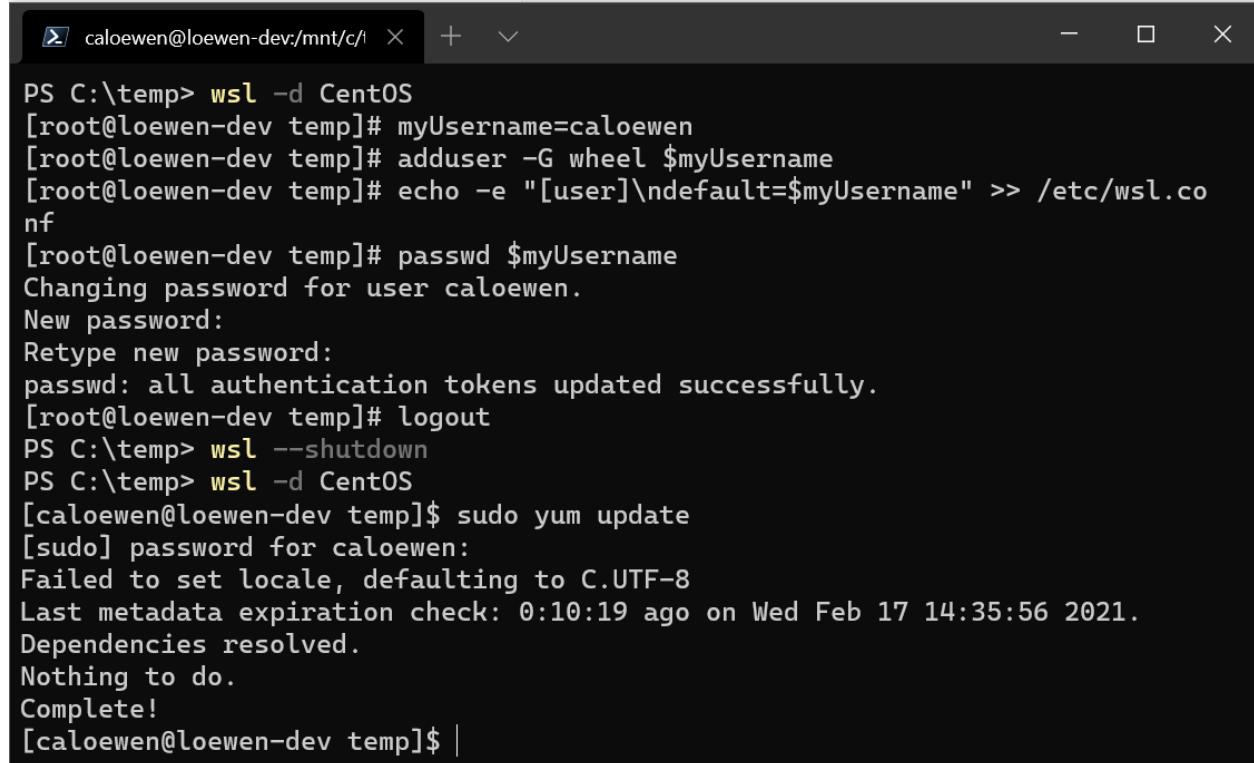
```
yum update -y && yum install passwd sudo -y  
myUsername=caloewen  
adduser -G wheel $myUsername  
echo -e "[user]\ndefault=$myUsername" >> /etc/wsl.conf  
passwd $myUsername
```

Ahora debe salir de esa instancia y asegurarse de que todas las instancias de WSL finalizan. Vuelva a iniciar la distribución para ver el nuevo usuario predeterminado mediante la ejecución de este comando en PowerShell:

PowerShell

```
wsl --terminate CentOS
wsl -d CentOS
```

Ahora [caloewen@loewen-dev]\$ verá como salida en función de este ejemplo.



The screenshot shows a Windows terminal window with a dark theme. The title bar says "caloewen@loewen-dev:/mnt/c/" and the window has standard minimize, maximize, and close buttons. The terminal content is as follows:

```
PS C:\temp> wsl -d CentOS
[root@loewen-dev temp]# myUsername=caloewen
[root@loewen-dev temp]# adduser -G wheel $myUsername
[root@loewen-dev temp]# echo -e "[user]\ndefault=$myUsername" >> /etc/wsl.conf
[root@loewen-dev temp]# passwd $myUsername
Changing password for user caloewen.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@loewen-dev temp]# logout
PS C:\temp> wsl --shutdown
PS C:\temp> wsl -d CentOS
[caloewen@loewen-dev temp]$ sudo yum update
[sudo] password for caloewen:
Failed to set locale, defaulting to C.UTF-8
Last metadata expiration check: 0:10:19 ago on Wed Feb 17 14:35:56 2021.
Dependencies resolved.
Nothing to do.
Complete!
[caloewen@loewen-dev temp]$ |
```

Para obtener más información sobre cómo configurar las opciones de WSL, consulte [Configuración de valores con .wslconfig y wsl.conf](#).

Compilación de su propia distribución de Linux

Puedes crear tu propia distribución personalizada de Linux, empaquetada como una aplicación para UWP, que se comportará exactamente igual que las distribuciones de WSL disponibles en Microsoft Store. Para saber cómo, consulte [Creación de una distribución Linux personalizada para WSL](#).

Creación de una distribución personalizada de Linux para WSL

Artículo • 07/10/2023

Use nuestro ejemplo de código abierto WSL para compilar paquetes de distribución de WSL para Microsoft Store o para crear paquetes de distribución de Linux personalizados para transferir localmente. Puede encontrar el [repositorio del iniciador de distribuciones](#) en GitHub.

Este proyecto habilita:

- Mantenedores de distribución de Linux para empaquetar y enviar una distribución de Linux como appx que se ejecuta en WSL
- Desarrolladores para crear distribuciones de Linux personalizadas que se pueden transferir localmente en su máquina de desarrollo

Fondo

Distribuyamos distribuciones de Linux para WSL como aplicaciones para UWP a través de Microsoft Store. Puede instalar esas aplicaciones que se ejecutarán en WSL: el subsistema que se encuentra en el kernel de Windows. Este mecanismo de entrega tiene muchas ventajas, como se describe en una [entrada de blog anterior](#).

Transferir localmente un paquete de distribución de Linux personalizado

Puede crear un paquete de distribución de Linux personalizado como una aplicación para transferir localmente en la máquina personal. Tenga en cuenta que el paquete personalizado no se distribuirá a través de Microsoft Store a menos que se envíe como mantenedor de distribución. Para configurar la máquina para transferir localmente las aplicaciones, deberá habilitarla en la configuración del sistema en "Para desarrolladores". Asegúrese de tener el modo de desarrollador o de transferir localmente las aplicaciones seleccionadas.

Para mantenedores de Linux Distro

Para enviar a la Tienda, deberá trabajar con nosotros para recibir la aprobación de publicación. Si es propietario de la distribución de Linux interesado en agregar la

distribución a Microsoft Store, póngase en contacto con wslpartners@microsoft.com.

Introducción

Siga las instrucciones en el repositorio GitHub de Distro Launcher [↗](#) para crear un paquete de distro Linux personalizado.

Blogs del equipo

- Open Sourcing de una muestra de WSL para mantenedores de distribuciones Linux y carga lateral de distribuciones Linux personalizadas [↗](#)
- Blog de línea de comandos [↗](#)

Proporcionar comentarios

- Repositorio de GitHub del iniciador de distribuciones [↗](#)
- Seguimiento de problemas de GitHub para WSL [↗](#)

Montaje de un disco Linux en WSL 2

Artículo • 13/10/2023

Si quiere acceder a un formato de disco Linux que no es compatible con Windows, puede usar WSL 2 para montar el disco y acceder a su contenido. En este tutorial se tratarán los pasos para identificar el disco y la partición que se van a adjuntar a WSL 2, cómo montarlos y cómo acceder a ellos.

Si está conectando una unidad externa y no tiene éxito con estas instrucciones de montaje, puede probar las instrucciones para [Conectar dispositivos USB](#). El `wsl --mount` comando no admite actualmente USB, unidades flash ni lectores de tarjetas SD ([obtenga más información sobre este problema ↗](#)).

ⓘ Nota

Se requiere acceso de administrador para adjuntar un disco a WSL 2. El comando WSL 2 `mount` no admite el montaje de un disco (o particiones que pertenecen al disco) que se está usando actualmente. `wsl --mount` siempre adjunta todo el disco aunque solo se solicite una partición. No se puede montar el disco de instalación de Windows.

Requisitos previos

Tendrá que estar en Windows 11 compilación 22000, una posterior o ejecutar la versión de Microsoft Store de WSL. Para comprobar la versión de WSL y Windows, use el comando: `wsl.exe --version`

Diferencias entre el montaje de una unidad externa con formato de Windows y el formato de Linux

Las unidades externas con formato para Windows suelen usar el formato del sistema de archivos NTFS. Las unidades externas con formato para Linux suelen usar el formato del sistema de archivos Ext4.

Si ha montado una unidad con formato NTFS en el sistema de archivos de Windows, puede acceder a esa unidad desde la distribución de Linux mediante WSL creando un directorio montado (`sudo mkdir /mnt/d`, reemplazando "d" por cualquier letra de unidad

que quiera usar) y, a continuación, mediante el complemento de interoperabilidad del sistema de archivos `drvfs` con el comando:

Bash

```
sudo mount -t drvfs D: /mnt/d
```

[Más información sobre los escenarios de montaje](#).

Si tienes una unidad con formato Ext4, no puedes montarla en el sistema de archivos de Windows. Para montar una unidad con formato Ext4 en la distribución de Linux con WSL, puede usar el comando `wsl --mount` siguiendo las instrucciones siguientes.

Montaje de un disco sin particiones

Si tiene un disco que no tiene particiones, puede montarlo directamente mediante el comando `wsl --mount`. En primer lugar, debe identificar el disco.

1. **Identificar el disco** : para enumerar los discos disponibles en Windows, ejecute:

PowerShell

```
GET-CimInstance -query "SELECT * from Win32_DiskDrive"
```

Las rutas de acceso de los discos están disponibles en las columnas 'DeviceID'.

Normalmente bajo el formato `\.\PHYSICALDRIVE*`.

2. **Montar el disco** : con PowerShell puede montar el disco mediante la ruta de acceso detectada anteriormente, ejecute:

PowerShell

```
wsl --mount <DiskPath>
```

```

PS E:\wslDistroStorage\Ubuntu2004> GET-WMIOBJECT -query "SELECT * from Win32_DiskDrive"

Partitions : 1
DeviceID   : \\.\PHYSICALDRIVE0
Model      : Samsung SSD 970 EVO Plus 500GB
Size       : 500105249280
Caption    : Samsung SSD 970 EVO Plus 500GB

Partitions : 1
DeviceID   : \\.\PHYSICALDRIVE1
Model      : ST2000DM001-1CH164
Size       : 2000396321280
Caption    : ST2000DM001-1CH164

Partitions : 3
DeviceID   : \\.\PHYSICALDRIVE2
Model      : PM9A1 NVMe Samsung 256GB
Size       : 256052966400
Caption    : PM9A1 NVMe Samsung 256GB

Partitions : 0
DeviceID   : \\.\PHYSICALDRIVE3
Model      : Microsoft Virtual Disk
Size       : 322118415360
Caption    : Microsoft Virtual Disk

PS E:\wslDistroStorage\Ubuntu2004> wsl --mount \\.\PHYSICALDRIVE3
The disk \\.\PHYSICALDRIVE3 was successfully mounted under the name 'PHYSICALDRIVE3'. The mountpoint can be found under the path pointed to by the automount setting (default: /mnt/wsl).
To unmount and detach the disk, run 'wsl --unmount \\.\PHYSICALDRIVE3'.
PS E:\wslDistroStorage\Ubuntu2004> wsl
craig@craig-Alienware:/mnt/e/wslDistroStorage/Ubuntu2004$ cd /mnt/wsl/PHYSICALDRIVE3/
craig@craig-Alienware:/mnt/wsl/PHYSICALDRIVE3$ ls
bin  dev  home  lib  lib64  lost+found  mnt  proc  run  snap  sys  usr  wslHkjNMD  wslKEAFMJ  wslcnleED  wslolnend
boot  etc  init  lib32  libx32  media  opt  root  sbin  srv  tmp  var  wslJInHfN  wslKFeiGO  wslfCNNoM  wslpjNEiK
craig@craig-Alienware:/mnt/wsl/PHYSICALDRIVE3$
```

Montaje de un disco con particiones

Si tiene un disco del que no está seguro sobre el formato de archivo en el que se encuentra o cuántas particiones tiene, puede seguir los pasos que se indican a continuación para montarlo.

- Identificar el disco** : para enumerar los discos disponibles en Windows, ejecute:

```

PowerShell

GET-CimInstance -query "SELECT * from Win32_DiskDrive"
```

Las rutas de acceso de los discos se muestran después de 'DeviceID', normalmente en el formato `\\.\PHYSICALDRIVE*`.

- Enumere y seleccione las particiones que se van a montar en WSL 2**: una vez identificado el disco, ejecute:

```

PowerShell

wsl --mount <DiskPath> --bare
```

Esto hará que el disco esté disponible en WSL 2. En el caso de nuestro ejemplo, `<DiskPath>` es `\\.\PHYSICALDRIVE*`.

3. Una vez adjuntada, la partición puede ser enumerada ejecutando el siguiente comando dentro de WSL 2:

```
Bash
```

```
lsblk
```

Se mostrarán los dispositivos de bloque disponibles y sus particiones.

Dentro de Linux un dispositivo de bloque se identifica como `/dev/<Device><Partition>`. Por ejemplo, `/dev/sdb3`, es el número de partición 3 del disco `sdb`.

Ejemplo:

```
Bash
```

```
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sdb      8:16   0    1G  0 disk 
└─sdb2   8:18   0   50M  0 part 
  ├─sdb3   8:19   0  873M  0 part 
  └─sdb1   8:17   0 100M  0 part 
sdc      8:32   0 256G  0 disk /
sda      8:0    0 256G  0 disk
```

Identificación del tipo de sistema de archivos

Si no conoce el tipo de sistema de archivos de un disco o partición puede usar este comando:

```
Bash
```

```
blkid <BlockDevice>
```

Esto generará el tipo de sistema de archivos detectado (con el formato `TYPE="<Filesystem>"`).

Montaje de las particiones seleccionadas

Una vez que haya identificado las particiones que desea montar, ejecute este comando en cada una:

```
PowerShell
```

```
wsl --mount <DiskPath> --partition <PartitionNumber> --type <Filesystem>
```

ⓘ Nota

Si desea montar todo el disco como un único volumen (es decir, si el disco no tiene particiones), `--partition` se puede omitir.

Si se omite, el tipo de sistema de archivos predeterminado es "ext4".

Acceso al contenido del disco

Una vez montado, se puede acceder al disco en la ruta de acceso a la que apunta el valor de configuración: `automount.root`. El valor predeterminado es `/mnt/wsl`.

Desde Windows, se puede acceder al disco desde Explorador de archivos; para ello, vaya a: `\\\wsl$\\<Distro>\\<Mountpoint>` (elija cualquier distribución de Linux).

Desmontaje de los discos

Si desea desmontar y desasociar el disco de WSL 2, ejecute:

PowerShell

```
wsl --unmount <DiskPath>
```

Montar un VHD en WSL

ⓘ Nota

WSL de Microsoft Store [↗](#) presenta un nuevo argumento para montar directamente un VHD: `wsl --mount --vhd <pathToVHD>`

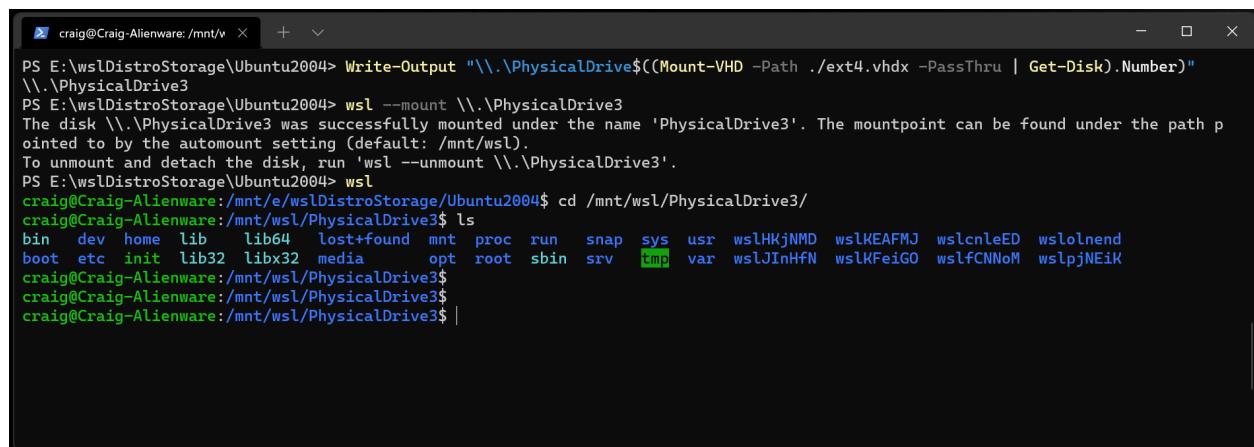
También puede montar archivos de disco duro virtual (VHD) en WSL mediante `wsl --mount`. Para ello, primero debe montar el VHD en Windows mediante el comando [Mount-VHD](#) en Windows. Asegúrese de ejecutar este comando con privilegios de administrador. A continuación se muestra un ejemplo en el que usamos este comando y también se genera la ruta de acceso del disco. Asegúrese de reemplazar `<pathToVHD>` con la ruta de acceso del VHD real.

PowerShell

```
Write-Output "\\.\PhysicalDrive$((Mount-VHD -Path <pathToVHD> -PassThru | Get-Disk).Number)"
```

Puede usar la salida anterior para obtener la ruta de acceso de este VHD y montarlo en WSL siguiendo las instrucciones de la sección anterior.

También puede usar esta técnica para montar e interactuar con los discos duros virtuales de otras distribuciones de WSL, ya que cada distribución de WSL 2 se almacena a través de un archivo de disco duro virtual denominado: `ext4.vhdx`. De forma predeterminada, los VHD de las distribuciones de WSL 2 se almacenan en esta ruta de acceso: `C:\Users\[user]\AppData\Local\Packages\[distro]\LocalState\[distroPackageName]`, tenga cuidado al acceder a estos archivos del sistema, este es un flujo de trabajo para usuario avanzado. Asegúrese de ejecutar `wsl --shutdown` antes de interactuar con este disco para estar seguro de que no está en uso.



The screenshot shows a Windows PowerShell window with the following command history:

```
PS E:\wslDistroStorage\Ubuntu2004> Write-Output "\\.\PhysicalDrive$((Mount-VHD -Path ./ext4.vhdx -PassThru | Get-Disk).Number)"\\.\PhysicalDrive3
PS E:\wslDistroStorage\Ubuntu2004> wsl --mount \\.\PhysicalDrive3
The disk \\.\PhysicalDrive3 was successfully mounted under the name 'PhysicalDrive3'. The mountpoint can be found under the path pointed to by the automount setting (default: /mnt/wsl).
To unmount and detach the disk, run 'wsl --unmount \\.\PhysicalDrive3'.
PS E:\wslDistroStorage\Ubuntu2004> wsl
craig@craig-Alienware:/mnt/e/wslDistroStorage/Ubuntu2004$ cd /mnt/wsl/PhysicalDrive3/
craig@craig-Alienware:/mnt/wsl/PhysicalDrive3$ ls
bin dev home lib lib64 lost+found mnt proc run snap sys usr wslHKjNMD wslKEAFMJ wslcnleED wslolnend
boot etc init lib32 libx32 media opt root sbin srv tmp var wslJInHfN wslKFeiGO wslfcNNoM wslpjNEiK
craig@craig-Alienware:/mnt/wsl/PhysicalDrive3$
craig@craig-Alienware:/mnt/wsl/PhysicalDrive3$ 
craig@craig-Alienware:/mnt/wsl/PhysicalDrive3$ |
```

Referencia de línea de comandos

Montaje de un sistema de archivos específico

De forma predeterminada, WSL 2 intentará montar el dispositivo como ext4. Para especificar otro sistema de archivos, ejecute:

PowerShell

```
wsl --mount <DiskPath> -t <FileSystem>
```

Por ejemplo, para montar un disco como fat, ejecute:

```
wsl --mount <Diskpath> -t vfat
```

ⓘ Nota

Para enumerar los sistemas de archivos disponibles en WSL 2, ejecute: `cat /proc/filesystems`

Cuando un disco se ha montado a través de WSL 2 (sistema de archivos Linux), ya no está disponible para montarse a través de un controlador ext4 en el sistema de archivos de Windows.

Montaje de una partición específica

De forma predeterminada, WSL 2 intenta montar todo el disco. Para montar una partición específica, ejecute:

```
wsl --mount <Diskpath> -p <PartitionIndex>
```

Esto solo funciona si el disco es MBR (registro de arranque maestro) o GPT (tabla de particiones GUID). [Obtenga información sobre los estilos de partición: MBR y GPT](#).

Especificación de opciones de montaje

Para especificar las opciones de montaje, ejecute:

PowerShell

```
wsl --mount <DiskPath> -o <MountOptions>
```

Ejemplo:

PowerShell

```
wsl --mount <DiskPath> -o "data=ordered"
```

ⓘ Nota

Solo se admiten las opciones específicas del sistema de archivos en este momento.

No se admiten opciones genéricas, como `ro`, `rw`, `noatime`,

Adjuntar el disco sin montarlo

Si el esquema de disco no es compatible con ninguna de las opciones anteriores, puede adjuntar el disco a WSL 2 sin montarlo ejecutando:

PowerShell

```
wsl --mount <DiskPath> --bare
```

Esto hará que el dispositivo de bloque esté disponible dentro de WSL 2 para que se pueda montar manualmente desde allí. Use `lsblk` para enumerar los dispositivos de bloque disponibles dentro de WSL 2.

Especificación del nombre del montaje

ⓘ Nota

Esta opción solo está disponible con [WSL desde Microsoft Store](#)

De forma predeterminada, el nombre del punto de montaje se genera en función del disco físico o el nombre del VHD. Esto se puede invalidar con `--name`. Ejemplo:

PowerShell

```
wsl --mount <DiskPath> --name myDisk
```

Desasociar un disco

Para desasociar un disco de WSL 2, ejecute:

PowerShell

```
wsl --unmount [DiskPath]
```

Si `Diskpath` se omite, todos los discos adjuntados se desmontan y desasocian.

Nota

Si un disco no se puede desmontar, WSL 2 puede ser forzado a salir mediante la ejecución de `wsl --shutdown`, que desasociará el disco.

Limitaciones

- En este momento, solo se pueden adjuntar discos completos a WSL 2, lo que significa que no es posible adjuntar solo una partición. Concretamente, esto significa que no es posible usar `wsl --mount` para leer una partición en el dispositivo de arranque, ya que ese dispositivo no se puede desasociar de Windows.
- Solo los sistemas de archivos que se admiten de forma nativa en el kernel se pueden montar mediante `wsl --mount`. Esto significa que no es posible usar controladores de sistema de archivos instalados (por ejemplo: ntfs-3g) mediante una llamada a `wsl --mount`.
- Los sistemas de archivos no compatibles directamente con el kernel se pueden montar a través de una conexión `--bare` y, a continuación, invocar al controlador FUSE correspondiente.

Conexión de dispositivos USB

Artículo • 05/02/2024

En esta guía se describen los pasos necesarios para conectar un dispositivo USB a una distribución de Linux que se ejecuta en WSL 2 mediante el proyecto de código abierto USB/IP, [usbipd-win](#).

Configurar el proyecto USB/IP en tu máquina Windows permitirá escenarios USB comunes para desarrolladores como flashear un Arduino o acceder a un lector de tarjetas inteligentes.

Requisitos previos

- Ejecutar Windows 11 (compilación 22000 o posterior). (*La compatibilidad con Windows 10 es posible, consulte la nota siguiente*).
- Se requiere una máquina con un procesador x64. (actualmente no se admiten x86 y Arm64 con usbipd-win).
- WSL está [instalado y configurado con la versión más reciente](#).
- Distribución Linux [instalada y configurada en WSL 2](#).

ⓘ Nota

Para comprobar la versión y el número de compilación de Windows, seleccione la **tecla del logotipo de Windows + R**, escriba `winver` y seleccione **Aceptar**. Para actualizar a la versión de Windows más reciente, seleccione **Inicio > Configuración > Windows Update > Buscar actualizaciones**. Para comprobar la versión del kernel de Linux, abra la distribución de Linux y escriba el comando : `uname -a`. Para actualizar manualmente al kernel más reciente, abra PowerShell y escriba el comando : "`wsl --update`".

ⓘ Importante

WSL ahora es compatible con Windows 10 y Windows 11 a través de Microsoft Store, lo que significa que los usuarios de Windows 10 ahora tienen acceso a las versiones más recientes del kernel sin necesidad de compilar desde el origen. Consulte [WSL en Microsoft Store ya está disponible con carácter general en Windows 10 y 11](#) para obtener información sobre cómo actualizar a la versión de WSL compatible con Store. Si no puede actualizar a la versión de WSL compatible con Store y recibir automáticamente actualizaciones del kernel, consulte el

[repositorio del proyecto USBIPD-WIN](#) para obtener instrucciones sobre cómo conectar dispositivos USB a una distribución de Linux que se ejecute en WSL 2 mediante la compilación de su propio kernel WSL 2 habilitado para USBIP.

Instalación del proyecto USBIPD-WIN

La compatibilidad con la conexión de dispositivos USB no está disponible de forma nativa en WSL, por lo que deberá instalar el proyecto usbipd-win de código abierto.

1. Vaya a la [página de la versión más reciente del proyecto usbipd-win](#).
2. Seleccione el archivo .msi, que descargará el instalador. (Es posible que reciba una advertencia que le pida que confirme que confía en el instalador descargado).
3. Ejecute el archivo de instalación usbipd-win_x.msi descargado.

ⓘ Nota

También puede instalar el proyecto usbipd-win mediante el **Administrador de paquetes de Windows** (winget). Si ya ha instalado winget, sólo tiene que utilizar el comando: `winget install --interactive --exact dorssel.usbipd-win` para instalar usbipd-win. Si deja fuera --interactive, winget puede reiniciar inmediatamente el equipo si es necesario para instalar los controladores.

Con esta acción se instalará:

- Un servicio denominado `usbipd` (nombre para mostrar: host de dispositivo USBIP). Puede comprobar el estado de este servicio mediante la aplicación Services en Windows.
- Una herramienta de línea de comandos `usbipd`. La ubicación de esta herramienta se agrega a la variable de entorno PATH.
- Una regla de firewall llama a `usbipd` para permitir que todas las subredes locales se conecten al servicio. Puede modificar esta regla de firewall para ajustar el control de acceso.

Instalar un dispositivo USB

Antes de conectar el dispositivo USB, asegúrese de que hay abierta una línea de comandos de WSL. Esto mantendrá activa la máquina virtual ligera WSL 2.

ⓘ Nota

En este documento se supone que tiene `usbipd-win` 4.0.0 o una versión posterior instalada ↗

1. Para enumerar todos los dispositivos USB conectados a Windows, abra PowerShell en modo *de administrador* y escriba el siguiente comando. Una vez que aparezcan los dispositivos, seleccione y copie el identificador de bus del dispositivo que desea asociar a WSL.

```
PowerShell
```

```
usbipd list
```

2. Antes de conectar el dispositivo USB, el comando `usbipd bind` debe usarse para compartir el dispositivo, lo que le permite estar conectado a WSL. Esto requiere privilegios de administrador. Seleccione el identificador de bus del dispositivo que desea usar en WSL y ejecute el siguiente comando. Después de ejecutar el comando, compruebe que el dispositivo se comparte con el comando `usbipd list` de nuevo.

```
PowerShell
```

```
usbipd bind --busid 4-4
```

3. Para conectar el dispositivo USB, ejecute el siguiente comando. (Ya no es necesario usar un símbolo del sistema de administrador con privilegios elevados). Asegúrese de que un símbolo del sistema de WSL esté abierto para mantener activa la máquina virtual ligera de WSL 2. Tenga en cuenta que, siempre que el dispositivo USB esté conectado a WSL, Windows no puede usarlo. Una vez conectado a WSL, cualquier distribución que se ejecute como WSL 2 puede usar el dispositivo USB. Compruebe que el dispositivo está conectado mediante `usbipd list`. Desde el símbolo del sistema WSL, ejecute `lsusb` para comprobar que aparece el dispositivo USB y que se puede interactuar con el uso de herramientas de Linux.

```
PowerShell
```

```
usbipd attach --wsl --busid <busid>
```

4. Abra Ubuntu (o su línea de comandos WSL preferida) y enumere los dispositivos USB conectados mediante el comando :

```
Bash
```

```
lsusb
```

Debería ver el dispositivo que acaba de conectar y poder interactuar con él mediante herramientas normales de Linux. En función de la aplicación, es posible que tenga que configurar reglas udev para permitir que los usuarios que no sean raíz accedan al dispositivo.

5. Una vez que haya terminado de usar el dispositivo en WSL, puede desconectar físicamente el dispositivo USB o ejecutar este comando desde PowerShell:

```
PowerShell
```

```
usbipd detach --busid <busid>
```

Para obtener más información sobre cómo funciona, consulte el [blog de la línea de comandos de Windows](#) y el [repositorio usbipd-win en GitHub](#).

Para ver una demostración en vídeo, consulte [WSL 2: Conectar dispositivos USB \(pestañas frente a mostrar espacios\)](#).



Colaborar con nosotros en GitHub

El origen de este contenido se puede encontrar en GitHub, donde también puede crear y revisar problemas y solicitudes de incorporación de cambios. Para más información, consulte [nuestra guía para colaboradores](#).



Comentarios de Windows Subsystem for Linux

Windows Subsystem for Linux es un proyecto de código abierto. Seleccione un vínculo para proporcionar comentarios:

[Abrir incidencia con la documentación](#)

[Proporcionar comentarios sobre el producto](#)

Ajuste de la distinción de mayúsculas y minúsculas

Artículo • 21/03/2023

La distinción entre mayúsculas y minúsculas determina si las letras mayúsculas (FOO.txt) y minúsculas (foo.txt) se controlan como distintas (distinguen mayúsculas de minúsculas) o equivalentes (sin distinción entre mayúsculas y minúsculas) en un nombre de archivo o directorio.

- Distingue entre mayúsculas y minúsculas: FOO.txt ≠ foo.txt ≠ Foo.txt
- Sin distinción entre mayúsculas y minúsculas: FOO.txt = foo.txt = Foo.txt

Diferencias entre la distinción entre mayúsculas y minúsculas de Windows y Linux

Al trabajar con archivos y directorios de Linux y Windows, es posible que tenga que ajustar cómo se controla la distinción de mayúsculas y minúsculas.

Comportamiento estándar:

- El sistema de archivos de Windows trata los nombres de archivo y directorio como no distingue mayúsculas de minúsculas. FOO.txt y foo.txt se tratarán como archivos equivalentes.
- El sistema de archivos de Linux trata los nombres de archivo y directorio como distingue mayúsculas de minúsculas. FOO.txt y foo.txt se tratarán como archivos distintos.

El sistema de archivos de Windows admite la configuración de la distinción entre mayúsculas y minúsculas con marcas de atributo por directorio. Aunque el comportamiento estándar no distingue mayúsculas de minúsculas, puede asignar una marca de atributo para distinguir mayúsculas y minúsculas de directorio, de modo que reconozca los archivos y carpetas de Linux que solo pueden diferir por mayúsculas y minúsculas.

Esto puede ser especialmente cierto al montar unidades en el sistema de archivos Subsistema de Windows para Linux (WSL). Cuando se trabaja en el sistema de archivos WSL, se ejecuta Linux, por lo que los archivos y directorios se tratan como distinguen mayúsculas de minúsculas de forma predeterminada.

 Nota

En el pasado, si tenía archivos cuyo nombre solo difiere por caso, Windows no pudo acceder a estos archivos, ya que las aplicaciones de Windows tratan el sistema de archivos como sin distinción entre mayúsculas y minúsculas y no pueden distinguir entre archivos cuyos nombres solo difieren en el caso. Aunque Windows Explorador de archivos mostrará ambos archivos, solo se abrirá uno independientemente de lo que seleccione.

Cambiar la distinción entre mayúsculas y minúsculas de archivos y directorios

En los pasos siguientes se explica cómo cambiar un directorio en el sistema de archivos de Windows para que distingue mayúsculas de minúsculas y reconocerá archivos y carpetas que solo difieren por mayúsculas y minúsculas.

Advertencia

Algunas aplicaciones de Windows, con la suposición de que el sistema de archivos no distingue mayúsculas de minúsculas, no use el caso correcto para hacer referencia a los archivos. Por ejemplo, no es raro que las aplicaciones transforme los nombres de archivo para usar mayúsculas o minúsculas. En los directorios marcados como distinguen mayúsculas de minúsculas, esto significa que estas aplicaciones ya no pueden acceder a los archivos. Además, si las aplicaciones de Windows crean nuevos directorios en un árbol de directorios en el que se usan archivos que distinguen mayúsculas de minúsculas, estos directorios no distinguen mayúsculas de minúsculas. Esto puede dificultar el trabajo con herramientas de Windows en directorios confidenciales entre mayúsculas y minúsculas, por lo que tenga cuidado al cambiar la configuración de distinción de mayúsculas y minúsculas del sistema de archivos de Windows.

Inspección de la distinción de mayúsculas y minúsculas actual

Para comprobar si un directorio distingue mayúsculas de minúsculas en el sistema de archivos de Windows, ejecute el comando :

PowerShell

```
fsutil.exe file queryCaseSensitiveInfo <path>
```

Reemplace por <path> la ruta de acceso del archivo. En el caso de un directorio en el sistema de archivos de Windows (NTFS), tendrá un <path> aspecto similar al siguiente: C:\Users\user1\case-test o si ya está en el user1 directorio, podría ejecutar lo siguiente: fsutil.exe file setCaseSensitiveInfo case-test

Modificación de la distinción de mayúsculas y minúsculas

La compatibilidad con la distinción de mayúsculas y minúsculas por directorio comenzó en Windows 10, compilación 17107. En Windows 10, compilación 17692, se actualizó la compatibilidad para incluir la inspección y modificación de la marca de distinción de mayúsculas y minúsculas de un directorio desde WSL. La distinción entre mayúsculas y minúsculas se expone mediante un atributo extendido denominado system.wsl_case_sensitive. El valor de este atributo será 0 para los directorios que no distinguen mayúsculas de minúsculas y 1 para los directorios que distinguen mayúsculas de minúsculas.

Para cambiar la distinción entre mayúsculas y minúsculas de un directorio, es necesario ejecutar **permisos elevados** (ejecutar como administrador). Cambiar la marca de distinción de mayúsculas y minúsculas también requiere permisos de "Escribir atributos", "Crear archivos", "Crear carpetas" y "Eliminar subcarpetas y archivos" en el directorio. [Consulte la sección de solución de problemas para obtener más información sobre esto.](#)

Para cambiar un directorio en el sistema de archivos de Windows de modo que distingue mayúsculas de minúsculas (FOO ≠ foo), ejecute PowerShell como administrador y use el comando :

PowerShell

```
fsutil.exe file setCaseSensitiveInfo <path> enable
```

Para volver a cambiar un directorio del sistema de archivos de Windows al valor predeterminado sin distinción entre mayúsculas y minúsculas (FOO = foo), ejecute PowerShell como administrador y use el comando :

PowerShell

```
fsutil.exe file setCaseSensitiveInfo <path> disable
```

Un directorio debe estar vacío para cambiar el atributo de marca de distinción de mayúsculas y minúsculas en ese directorio. No se puede deshabilitar la marca de

distinción entre mayúsculas y minúsculas en un directorio que contenga carpetas o archivos cuyos nombres solo difieren en mayúsculas y minúsculas.

Herencia de distinción entre mayúsculas y minúsculas

Al crear nuevos directorios, esos directorios heredarán la distinción entre mayúsculas y minúsculas de su directorio primario.

⚠️ Advertencia

Hay una excepción a esta directiva de herencia cuando se ejecuta en modo WSL 1. Cuando una distribución se ejecuta en modo WSL 1, la marca de distinción entre mayúsculas y minúsculas por directorio no se hereda; Los directorios creados en un directorio con distinción entre mayúsculas y minúsculas no distinguen automáticamente mayúsculas de minúsculas. Debe marcar explícitamente cada directorio como distingue mayúsculas de minúsculas.

Opciones de distinción de mayúsculas y minúsculas para montar una unidad en el archivo de configuración de WSL

La distinción entre mayúsculas y minúsculas se puede administrar al montar una unidad en el Subsistema de Windows para Linux mediante el archivo de configuración de WSL. Cada distribución de Linux que haya instalado puede tener su propio archivo de configuración de WSL, denominado `/etc/wsl.conf`. Para obtener más información sobre cómo montar una unidad, consulte [Introducción al montaje de un disco Linux en WSL 2](#).

Para configurar la opción de distinción entre mayúsculas y minúsculas en el `wsl.conf` archivo al montar una unidad:

1. Abra la distribución de Linux que va a usar (es decir, . Ubuntu).
2. Cambie los directorios hasta que vea la `etc` carpeta (esto puede requerir que se encuentre desde `cd ..` el `home` directorio).
3. Enumere los archivos del `etc` directorio para ver si ya existe un `wsl.conf` archivo (use el `ls` comando o `explorer.exe .` para ver el directorio con Windows Explorador de archivos).
4. Si el `wsl.conf` archivo aún no existe, puede crearlo mediante: `sudo touch wsl.conf` o mediante la ejecución `sudo nano /etc/wsl.conf`, que creará el archivo al guardarlo desde el editor nano.

5. Las siguientes opciones están disponibles para agregar al `wsl.conf` archivo:

Configuración predeterminada: `dir` para habilitar la distinción de mayúsculas y minúsculas por directorio.

Bash

```
[automount]
options = case = dir
```

Distinción entre mayúsculas y minúsculas no disponible (todos los directorios de las unidades NTFS montadas no distinguen mayúsculas de minúsculas): `off`

Bash

```
[automount]
options = case = off
```

Trate todos los directorios de la unidad (NTFS) como distingue mayúsculas de minúsculas: `force`

Bash

```
[automount]
options = case = force
```

Esta opción solo se admite para montar unidades en distribuciones de Linux que se ejecutan como WSL 1 y puede requerir una clave de registro. Para agregar una clave de registro, puede usar este comando desde un símbolo del sistema con privilegios elevados (administrador): `reg.exe add HKLM\SYSTEM\CurrentControlSet\Services\lxss /v DrvFsAllowForceCaseSensitivity /t REG_DWORD /d 1.`

Tendrá que reiniciar WSL después de realizar cambios en el `wsl.conf` archivo para que esos cambios surtan efecto. Puede reiniciar WSL mediante el comando : `wsl --shutdown`

💡 Sugerencia

Para montar una unidad (que usa el complemento del sistema de archivos DrvFs para que el disco esté disponible en /mnt, como /mnt/c, /mnt/d, etc.) con una configuración de distinción de mayúsculas y minúsculas específica para todas las unidades, use `/etc/wsl.conf` como se ha descrito anteriormente. Para establecer las opciones de montaje predeterminadas para una unidad específica, use el `/etc/fstab` archivo [para especificar estas opciones](#). Para obtener más opciones de

configuración de WSL, consulte [Configuración por configuración de inicio de distribución con wslconf](#).

Cambio de la distinción entre mayúsculas y minúsculas en una unidad montada en una distribución de WSL

Las unidades con formato NTFS montadas en una distribución WSL no distinguen mayúsculas de minúsculas de forma predeterminada. Para cambiar la distinción entre mayúsculas y minúsculas de un directorio de una unidad montada en una distribución WSL (es decir, Ubuntu), siga los mismos pasos que se enumeran anteriormente para el sistema de archivos de Windows. (Las unidades EXT4 distinguen mayúsculas de minúsculas de forma predeterminada).

Para habilitar la distinción de mayúsculas y minúsculas en un directorio (`FOO ≠ foo`), use el comando :

Bash

```
fsutil.exe file setCaseSensitiveInfo <path> enable
```

Para deshabilitar la distinción entre mayúsculas y minúsculas en un directorio y volver al valor predeterminado sin distinción entre mayúsculas y minúsculas (`FOO = foo`), use el comando :

Bash

```
fsutil.exe file setCaseSensitiveInfo <path> disable
```

ⓘ Nota

Si cambia la marca que distingue mayúsculas de minúsculas en un directorio existente para una unidad montada mientras se ejecuta WSL, asegúrese de que WSL no tenga referencias a ese directorio o, de lo contrario, el cambio no será efectivo. Esto significa que ningún proceso de WSL debe abrir el directorio (o sus descendientes) como directorio actual.

Configuración de la distinción de mayúsculas y minúsculas con Git

El sistema de control de versiones de Git también tiene una configuración que se puede usar para ajustar la distinción entre mayúsculas y minúsculas de los archivos con los que está trabajando. Si usa Git, puede que desee ajustar la [git config core.ignorecase](#) configuración.

Para establecer Git que distingue mayúsculas de minúsculas (FOO.txt ≠ foo.txt), escriba:

```
git config core.ignorecase false
```

Para establecer Git que no distingue mayúsculas de minúsculas (FOO.txt = foo.txt), escriba:

```
git config core.ignorecase true
```

Establecer esta opción en false en un sistema de archivos que no distingue mayúsculas de minúsculas puede provocar errores confusos, conflictos falsos o archivos duplicados.

Para más información, consulte la [documentación de Configuración de Git](#).

Solución de problemas

Mi directorio tiene archivos con mayúsculas y minúsculas y requieren distinción entre mayúsculas y minúsculas, pero las herramientas de Windows FS no reconocerán estos archivos.

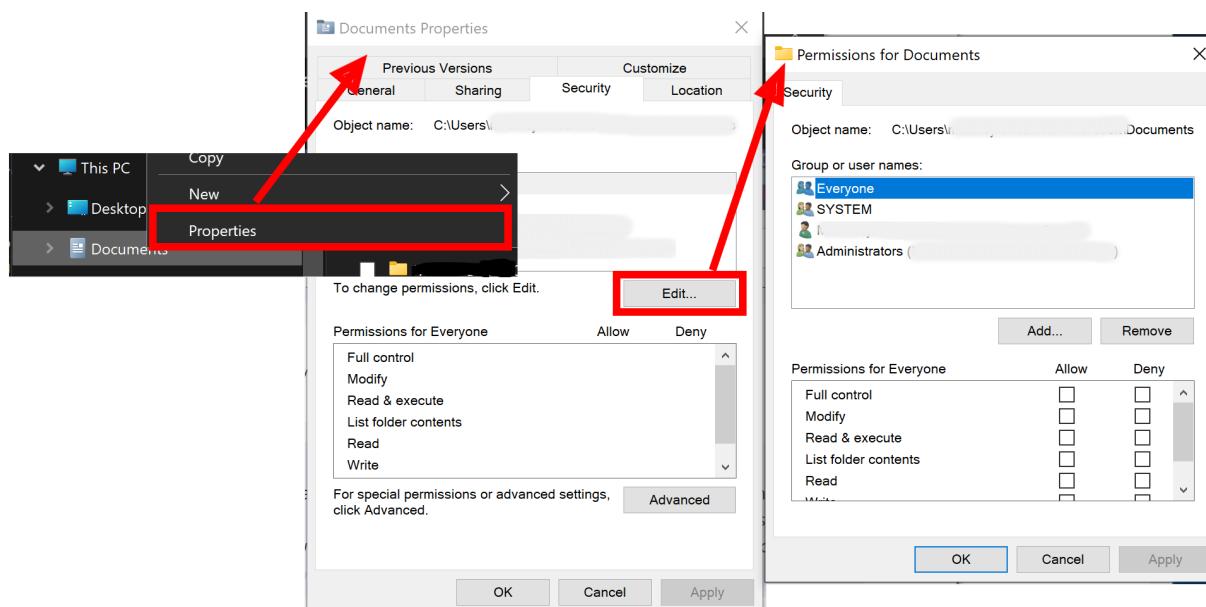
Para usar herramientas del sistema de archivos de Windows para trabajar en un directorio linux que contiene archivos de mayúsculas y minúsculas mixtas, deberá crear un directorio nuevo y establecerlo en distinguir mayúsculas de minúsculas y, a continuación, copiar los archivos en ese directorio (mediante git clone o untar). Los archivos permanecerán en mayúsculas y minúsculas mixtas. (Tenga en cuenta que, si ya ha intentado mover los archivos a un directorio que no distingue mayúsculas de minúsculas y hubo conflictos, es probable que haya algunos archivos que se sobrescribieron y que ya no estarán disponibles).

Error: El directorio no está vacío

No puede cambiar la configuración de distinción entre mayúsculas y minúsculas en un directorio que contenga otros archivos o directorios. Intente crear un nuevo directorio, cambiar la configuración y, a continuación, copiar los archivos de mayúsculas y minúsculas en él.

Error: Acceso denegado

Asegúrese de que tiene los permisos "Escribir atributos", "Crear archivos", "Crear carpetas" y "Eliminar subcarpetas y archivos" en el directorio necesario para cambiar la distinción de mayúsculas y minúsculas. Para comprobar esta configuración, abra el directorio en Windows Explorador de archivos (desde la línea de comandos, use el comando : `explorer.exe .`). Haga clic con el botón derecho en el directorio y seleccione **Propiedades** para abrir el ventana Propiedades documento y, a continuación, seleccione **Editar** para ver o cambiar los permisos del directorio.



Error: Se requiere un volumen NTFS local para esta operación.

El atributo de distinción entre mayúsculas y minúsculas solo se puede establecer en directorios dentro de un sistema de archivos con formato NTFS. Los directorios del sistema de archivos WSL (Linux) distinguen mayúsculas de minúsculas de forma predeterminada (y no se pueden establecer para que no distinguen mayúsculas de minúsculas mediante la herramienta `fsutil.exe`).

Recursos adicionales

- DevBlog: distinción entre mayúsculas y minúsculas por directorio y WSL ↗
- DevBlog: compatibilidad mejorada con la distinción entre mayúsculas y minúsculas por directorio en WSL ↗

Administración del espacio en disco de WSL

Artículo • 10/11/2023

En esta guía se explica cómo administrar el espacio en disco usado por las distribuciones de Linux instaladas mediante WSL 2, entre las que se incluyen:

- [Cómo comprobar la cantidad de espacio de disco disponible en el VHD](#)
- [Cómo expandir el tamaño del VHD](#)
- [Cómo reparar el VHD si se produce un error](#)
- [Cómo localizar el archivo .vhdx y la ruta de acceso del disco para las distribuciones de Linux instaladas](#)

Subsistema de Windows para Linux (WSL 2) usa una plataforma de virtualización para instalar distribuciones de Linux junto con el sistema operativo Windows host, creando un disco duro virtual (VHD) para almacenar archivos para cada una de las distribuciones de Linux que instale. Estos VHD usan el [tipo de sistema de archivos ext4](#) y se representan en el disco duro de Windows como un archivo `ext4.vhdx`.

WSL 2 cambia automáticamente el tamaño de estos archivos VHD para satisfacer las necesidades de almacenamiento. De forma predeterminada, cada archivo VHD usado por WSL 2 se asigna inicialmente una cantidad máxima de 1 TB de espacio en disco (antes de la [versión 0.58.0 de WSL](#), este valor predeterminado se estableció en un máximo de 512 GB y 256 GB máximo antes de eso).

Si el espacio de almacenamiento requerido por los archivos de Linux supera este tamaño máximo, verá errores que indican que se ha quedado sin espacio en disco. Para corregir este error, siga las instrucciones siguientes sobre [cómo expandir el tamaño del disco duro virtual de WSL 2](#).

Cómo comprobar el espacio en disco disponible

Compruebe la cantidad de espacio en disco disponible en el disco duro virtual para una distribución de Linux instalada con WSL 2 mediante el comando de Linux `df`.

Para comprobar el espacio en disco disponible, abra una línea de comandos de PowerShell y escriba este comando (reemplazando por `<distribution-name>` el nombre de distribución real):

PowerShell

```
wsl.exe --system -d <distribution-name> df -h /mnt/wslg/distro
```

Si este comando no funciona, actualice a la versión de Store de WSL mediante el `wsl --update` comando o pruebe `wsl df -h /`.

El resultado incluirá .

- **Sistema de archivos:** Identificador del sistema de archivos VHD
- **Tamaño:** Tamaño total del disco (la cantidad máxima de espacio asignado al VHD)
- **Usado:** Cantidad de espacio usado actualmente en el VHD
- **Disponibilidad:** Cantidad de espacio restante en el VHD (tamaño asignado menos cantidad usada)
- **Use%:** Porcentaje de espacio restante en disco (Tamaño usado / asignado)
- **Montado en:** Ruta del directorio donde está montado el disco

Si ve que está cerca de alcanzar la cantidad disponible de espacio en disco asignado al disco duro virtual o ya ha recibido un error debido a que no queda espacio en disco, consulte la sección siguiente para ver los pasos sobre cómo expandir la cantidad máxima de espacio en disco asignado al VHD asociado a la distribución de Linux. La cantidad de espacio en disco asignado al VHD por WSL siempre mostrará la cantidad máxima predeterminada (1 TB en la versión más reciente de WSL), incluso si la cantidad de espacio en disco en el dispositivo Windows real es menor que esa. WSL monta un VHD que se expandirá a medida que lo use, por lo que la distribución de Linux ve que puede crecer hasta el tamaño máximo asignado de 1 TB.

Cómo ampliar el tamaño de su disco duro virtual WSL 2

Para ampliar el tamaño del VHD para una distribución de Linux más allá de la cantidad máxima predeterminada de 1 TB de espacio de disco asignado, siga los pasos que se indican a continuación. (*Para versiones anteriores de WSL que aún no se han actualizado, este valor predeterminado máximo puede establecerse en 512 GB o 256 GB*).

1. Finaliza todas las instancias de WSL mediante el comando `wsl.exe --shutdown`.
2. Copie la ruta de acceso del directorio al archivo `ext4.vhdx` asociado a la distribución de Linux instalada en el equipo. Para obtener ayuda, vea [Cómo localizar el archivo vhdx y la ruta de acceso del disco para la distribución de Linux](#).

3. Abra el símbolo del sistema de Windows con privilegios de administrador y, a continuación, abra el intérprete de comandos `diskpart` escribiendo:

```
Símbolo del sistema de Windows
```

```
diskpart
```

4. Ahora tendrá un `DISKPART>` aviso. Escriba el siguiente comando, reemplazando `<pathToVHD>` por la ruta de acceso del directorio al `ext4.vhdx` archivo asociado a la distribución de Linux (copiada en el paso 2).

```
Símbolo del sistema de Windows
```

```
Select vdisk file="<pathToVHD>"
```

5. Muestra los detalles asociados a este disco virtual, incluido el **Tamaño virtual**, que representa el tamaño máximo actual al que se asigna el VHD:

```
Símbolo del sistema de Windows
```

```
detail vdisk
```

6. Tendrá que convertir el **Tamaño virtual** en megabytes. Por ejemplo, si **Tamaño virtual: 512 GB**, es igual a **512000 MB**. El nuevo valor que escribirá debe ser mayor que este valor inicial. Para duplicar el tamaño virtual de 512 GB a 1024 GB, escribiría el valor en MB como: **1024000**. Tenga cuidado de no especificar un valor superior al que realmente desee, ya que el proceso de reducción de un tamaño de disco virtual es mucho más complicado.

7. Escriba el valor del nuevo tamaño máximo que desea asignar a esta distribución de Linux mediante el símbolo del sistema `DISKPART>` de Windows:

```
Símbolo del sistema de Windows
```

```
expand vdisk maximum=<sizeInMegaBytes>
```

8. Salga del símbolo del `DISKPART>` sistema:

```
Símbolo del sistema de Windows
```

```
exit
```

9. Inicie esta distribución de Linux. (*Asegúrese de que se ejecuta en WSL 2. Puede confirmarlo mediante el comando wsl.exe -l -v. WSL 1 no se admite*).
10. Para indicar a WSL que puede ampliar el tamaño del sistema de archivos de esta distribución, ejecute estos comandos desde la línea de comandos de su distribución WSL. Podría mostrarse este mensaje en respuesta al primer comando **mount**: "/dev: none ya está montado en /dev". Este mensaje se puede omitir de forma segura.

Bash

```
sudo mount -t devtmpfs none /dev  
mount | grep ext4
```

11. Copie el nombre de esta entrada, que tendrá el siguiente aspecto: `/dev/sdX` (en que la X representa cualquier otro carácter). En el ejemplo siguiente, el valor de X es **b**:

Bash

```
sudo resize2fs /dev/sdb <sizeInMegabytes>M
```

Con el ejemplo anterior, cambiamos el tamaño del vhd a **2048000**, por lo que el comando sería: `sudo resize2fs /dev/sdb 2048000M`.

ⓘ Nota

También puede que necesite instalar **resize2fs**. En caso afirmativo, puede usar este comando para instalarlo: `sudo apt install resize2fs`.

La salida tendrá una apariencia parecida a la siguiente:

Bash

```
resize2fs 1.44.1 (24-Mar-2021)  
Filesystem at /dev/sdb is mounted on /; on-line resizing required  
old_desc_blocks = 32, new_desc_blocks = 38  
The filesystem on /dev/sdb is now 78643200 (4k) blocks long.
```

La unidad virtual (ext4.vhdx) para esta distribución de Linux ahora se ha ampliado correctamente al nuevo tamaño.

ⓘ Importante

Le recomendamos que no modifique, mueva ni acceda a los archivos relacionados con WSL que se encuentran dentro de su AppData carpeta mediante herramientas o editores de Windows. Al hacerlo, podrías dañar tu distribución de Linux. Si quiere acceder a los archivos de Linux desde Windows, puede usar la ruta de acceso `\wsl$<distribution-name>\.`. Abra la distribución de WSL y escriba `explorer.exe .` para ver esa carpeta. Para obtener más información, consulte la entrada de blog: [Acceso a archivos de Linux desde Windows ↗](#).

Cómo reparar un error de montaje de VHD

Si se produce un error relacionado con el "montaje del disco de distribución", esto podría deberse a un apagado repentino o a una interrupción del suministro eléctrico y puede provocar que el VHD de distribución de Linux se cambie a solo *lectura* para evitar la pérdida de datos. Puede reparar y restaurar la distribución mediante el `e2fsck` comando de Linux siguiendo los pasos que se indican a continuación.

Use el comando `lsblk` para identificar el nombre del dispositivo de bloque.

Cuando WSL 2 instala una distribución de Linux, monta la distribución como un disco duro virtual (VHD) con su propio sistema de archivos. Linux hace referencia a estas unidades de disco duro como "bloquear dispositivos" y puede ver información sobre ellas mediante el `lsblk` comando .

Para buscar los nombres de los dispositivos de bloqueo que está usando WSL 2, abra la distribución y escriba el comando: `lsblk`. (O abra PowerShell y escriba el comando: `wsl.exe lsblk`.) La salida tendrá un aspecto similar al siguiente:

Bash

```
NAME MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda    8:0      0 363.1M  1 disk
sdb    8:16     0   8G  0 disk [SWAP]
sdc    8:32     0  1.5T  0 disk
sdd    8:48     0   1T  0 disk /mnt/wslg/distro
```

La información sobre el dispositivo de bloqueo incluye:

- **NOMBRE:** El nombre asignado al dispositivo será sd[a-z], haciendo referencia al disco SCSI con una designación de letra para cada disco que se usa. `sda` siempre es la distribución del sistema.
- **MAJ:MIN:** Representa los números usados por el kernel de Linux para identificar internamente los dispositivos con el primer número que representa el tipo de dispositivo (8 se usa para discos SCSI o interfaz de sistema de equipos pequeños).
- **RM:** Háganos saber si el dispositivo es extraíble (1) o no (0).
- **TAMAÑO:** Tamaño total del volumen.
- **RO:** Háganos saber si el dispositivo es de solo lectura (1) o no (0).
- **TIPO:** Hace referencia al tipo de dispositivo (disco en este caso).
- **MOUNTPOINTS:** Hace referencia al directorio actual en el sistema de archivos donde se encuentra el dispositivo de bloque (SWAP es para memoria inactiva preconfigurada, por lo que no hay ningún punto de montaje).

Error de reserva de solo lectura

Si WSL encuentra un "error de montaje" al abrir una distribución de Linux, la distribución se puede establecer como de solo lectura como reserva. Si esto sucede, la distribución puede mostrar el siguiente error durante el inicio:

PowerShell

```
An error occurred mounting the distribution disk, it was mounted read-only as a fallback.
```

Cuando se inicia una distribución como de solo lectura, los intentos de escritura en el sistema de archivos producirán un error similar al siguiente:

Bash

```
$ touch file
touch: cannot touch 'file': Read-only file system
```

Para reparar un error de montaje de disco en WSL y restaurarlo de nuevo a un estado utilizable o grabable, puede usar el `wsl.exe --mount` comando para volver a montar el disco con los pasos siguientes:

1. Cierre todas las distribuciones de WSL abriendo PowerShell y escribiendo el comando :

PowerShell

```
wsl.exe --shutdown
```

2. Abra PowerShell como administrador (en un símbolo del sistema con privilegios elevados) y escriba el comando mount, reemplazando <path-to-ext4.vhdx> por la ruta de acceso al archivo .vhdx de la distribución. Para obtener ayuda para localizar este archivo, vea [Cómo localizar el archivo VHD y la ruta de acceso del disco para la distribución de Linux](#).

PowerShell

```
wsl.exe --mount <path-to-ext4.vhdx> --vhd --bare
```

3. Use el `wsl.exe lsblk` comando de PowerShell para identificar el nombre del dispositivo de bloque para la distribución (sd[a-z]) y, a continuación, escriba el siguiente comando para reparar el disco (reemplazando <device> por el nombre de dispositivo de bloque correcto, como "sdc"). El `e2fsck` comando comprueba los sistemas de archivos ext4 (el tipo utilizado por las distribuciones instaladas con WSL) en busca de errores y los repara en consecuencia.

PowerShell

```
wsl.exe sudo e2fsck -f /dev/<device>
```

ⓘ Nota

Si solo tiene instalada una única distribución de Linux, es posible que encuentre un error "ext file in use" y tendrá que **instalar** una distribución adicional para ejecutar `wsl.exe lsblk`. Puede **desinstalar** la distribución una vez completada la reparación.

4. Una vez completada la reparación, desmonte el disco en PowerShell escribiendo:

PowerShell

```
wsl.exe --unmount
```

⚠ Advertencia

Puede usar el comando: `sudo mount -o remount,rw /` para devolver una distribución de solo lectura a un estado utilizable o grabable, pero todos los cambios estarán en memoria, por lo que se perderán cuando se reinicie la

distribución. Se recomienda usar los pasos indicados anteriormente para montar y reparar el disco en su lugar.

Cómo localizar el archivo .vhdx y la ruta de acceso del disco para la distribución de Linux

Para buscar la ruta de acceso del directorio y el archivo .vhdx para una distribución de Linux, abra PowerShell y use el siguiente script, reemplazando <distribution-name> por el nombre de distribución real:

PowerShell

```
(Get-ChildItem -Path HKCU:\Software\Microsoft\Windows\CurrentVersion\Lxss |  
Where-Object { $_.GetValue("DistributionName") -eq '<distribution-name>' }  
).GetValue("BasePath") + "\ext4.vhdx"
```

El resultado mostrará una ruta de acceso similar a %LOCALAPPDATA%\Packages\<PackageFamilyName>\LocalState\<disk>.vhdx. Por ejemplo:

PowerShell

```
C:\Users\User\AppData\Local\Packages\CanonicalGroupLimited.UbuntuonWindows_7  
9rhkp1fndgsc\LocalState\ext4.vhdx
```

Esta es la ruta de acceso al ext4.vhdx archivo asociado a la distribución de Linux que ha enumerado.

Preguntas más frecuentes sobre el subsistema de Windows para Linux

Preguntas más frecuentes

General

¿Qué es el Subsistema de Windows para Linux (WSL)?

El Subsistema de Windows para Linux (WSL) es una característica del sistema operativo Windows que permite ejecutar un sistema de archivos Linux, junto con herramientas de línea de comandos y aplicaciones de GUI de Linux, directamente en Windows, junto con el escritorio y las aplicaciones tradicionales de Windows.

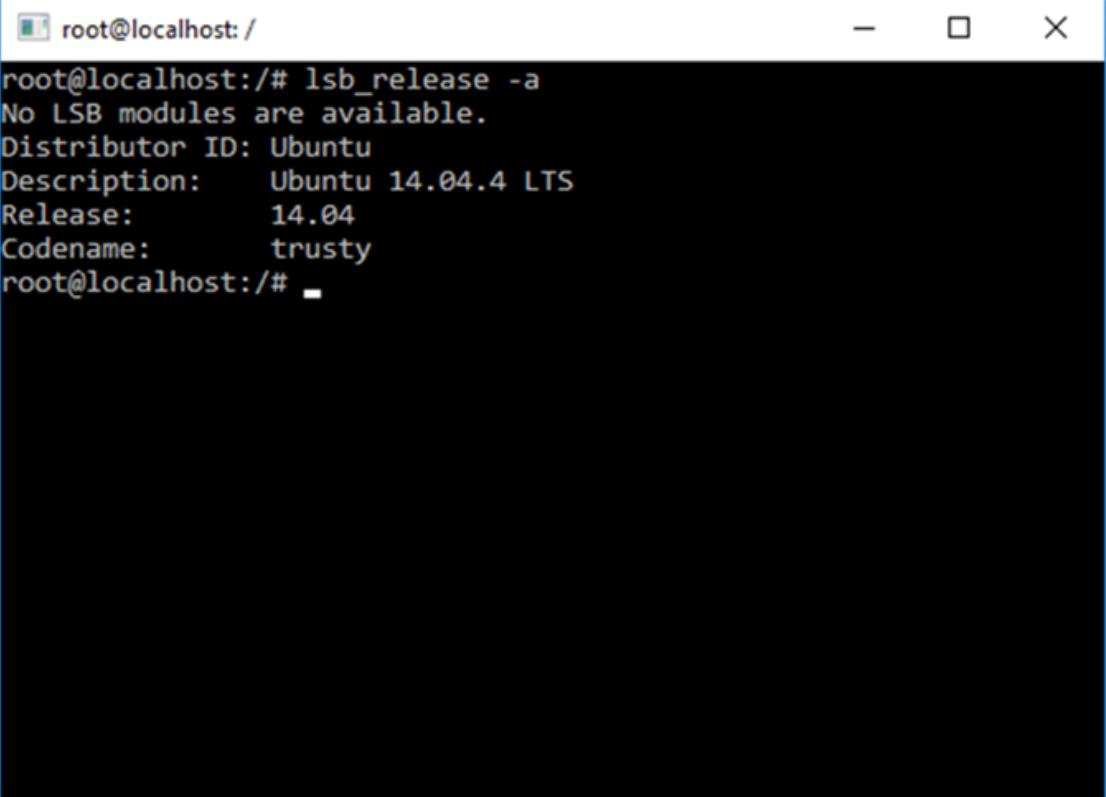
Consulta la página [Acerca de](#) para obtener más detalles.

¿Para quién es WSL?

Se trata principalmente de una herramienta para desarrolladores, especialmente desarrolladores web, aquellos que trabajan en proyectos de código abierto o implementaciones en entornos de servidor Linux. WSL es para cualquier persona que quiera usar Bash, herramientas comunes de Linux (`sed`, `awk`, etc.) y marcos de trabajo primero en Linux (Ruby, Python, etc.), pero también disfruta con herramientas de productividad de Windows.

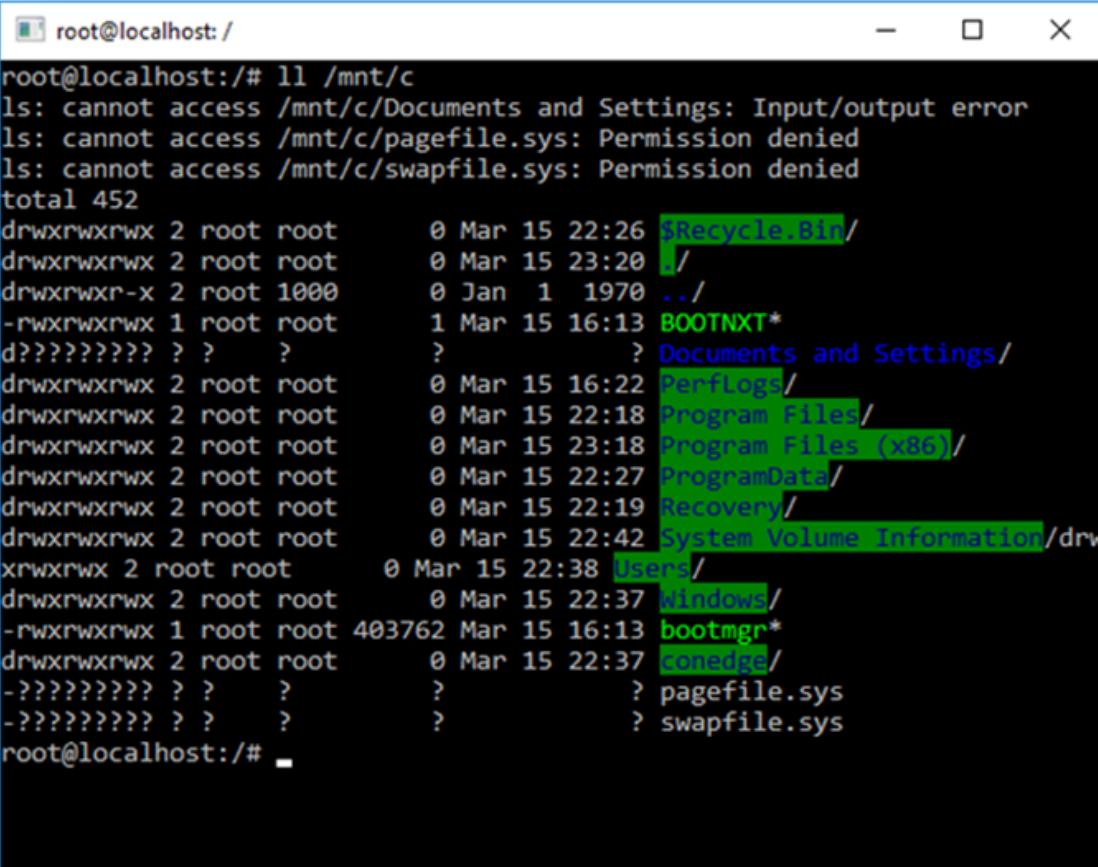
¿Qué puedo hacer con WSL?

WSL le permite ejecutar Linux en un shell de Bash con su elección de distribución (Ubuntu, Debian, OpenSUSE, Kali, Alpine, etc.). Con Bash, puedes ejecutar aplicaciones y herramientas de Linux de línea de comandos. Por ejemplo, escribe `lsb_release -a` y presiona Entrar para ver los detalles de la distribución de Linux que se está ejecutando actualmente:



```
root@localhost:/# lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 14.04.4 LTS
Release:        14.04
Codename:       trusty
root@localhost:/#
```

También puede acceder al sistema de archivos del equipo local desde el shell de Bash de Linux; encontrará las unidades locales montadas en la `/mnt` carpeta . Por ejemplo, la unidad `c:` se monta en `/mnt/c:`



```
root@localhost:/# ls /mnt/c
ls: cannot access /mnt/c/Documents and Settings: Input/output error
ls: cannot access /mnt/c/pagefile.sys: Permission denied
ls: cannot access /mnt/c/swapfile.sys: Permission denied
total 452
drwxrwxrwx 2 root root      0 Mar 15 22:26 $Recycle.Bin/
drwxrwxrwx 2 root root      0 Mar 15 23:20 /
drwxrwxr-x 2 root 1000      0 Jan  1 1970 ...
-rw-rwxrwx 1 root root     1 Mar 15 16:13 BOOTNXT*
d????????? ? ?      ?      ? Documents and Settings/
drwxrwxrwx 2 root root     0 Mar 15 16:22 PerfLogs/
drwxrwxrwx 2 root root     0 Mar 15 22:18 Program Files/
drwxrwxrwx 2 root root     0 Mar 15 23:18 Program Files (x86)/
drwxrwxrwx 2 root root     0 Mar 15 22:27 ProgramData/
drwxrwxrwx 2 root root     0 Mar 15 22:19 Recovery/
drwxrwxrwx 2 root root     0 Mar 15 22:42 System Volume Information/drw
xrwxrwx 2 root root      0 Mar 15 22:38 Users/
drwxrwxrwx 2 root root     0 Mar 15 22:37 Windows/
-rw-rwxrwx 1 root root 403762 Mar 15 16:13 bootmgr*
drwxrwxrwx 2 root root     0 Mar 15 22:37 Conedge/
-????????? ? ?      ?      ? pagefile.sys
-????????? ? ?      ?      ? swapfile.sys
root@localhost:/#
```

¿Puede describir un flujo de trabajo de desarrollo típico que incorpore WSL?

WSL va destinado a personas desarrolladoras y está pensado para usarse como parte de un bucle de desarrollo interno. Supongamos que Sam crea una canalización de CI/CD (integración & entrega continuas) y quiere probarla primero en una máquina local (portátil) antes de implementarla en la nube. Sam puede habilitar WSL (& WSL 2 para mejorar la velocidad y el rendimiento) y, después, usar una instancia original de Ubuntu Linux (en el portátil) con el comando y la herramienta de Bash que prefiera. Después de verificar localmente la canalización del desarrollo, Sam puede insertar esa canalización de CI/CD en la nube. Para ello, debe convertirla en un contenedor de Docker e insertar dicho contenedor en una instancia en la nube, que se ejecute en una máquina virtual Ubuntu preparada para producción.

¿Qué es Bash?

Bash [🔗](#) es un conocido shell basado en texto, que cuenta con un lenguaje de comandos. Es el shell predeterminado incluido en Ubuntu y otras distribuciones de Linux. Los

usuarios pueden escribir comandos en un shell para ejecutar scripts o ejecutar comandos y herramientas para realizar varias tareas.

¿Cómo funciona?

Consulte este artículo en el blog de la línea de comandos de Windows: [Una profundización en cómo WSL permite que Windows acceda a archivos de Linux ↗](#), que se detallan en detalle sobre la tecnología subyacente.

¿Por qué debo usar WSL en lugar de Linux en una VM?

WSL necesita menos recursos (CPU, memoria y almacenamiento) que una máquina virtual completa. Asimismo, WSL también te permite ejecutar aplicaciones y herramientas de línea de comandos de Linux junto con la línea de comandos de Windows, aplicaciones de escritorio y de Store, así como la posibilidad de acceder a los archivos de Windows desde Linux. Esto te permite usar las aplicaciones de Windows y las herramientas de línea de comandos de Linux en el mismo conjunto de archivos, si así lo deseas.

¿Por qué debo usar, por ejemplo, Ruby en Linux en lugar de en Windows?

Algunas herramientas multiplataforma se crearon suponiendo que el entorno en el que se ejecutan se comporta como Linux. Por ejemplo, algunas herramientas suponen que pueden obtener acceso a rutas de acceso de archivos muy largas o que existen archivos o carpetas específicos. Esto suele provocar problemas en Windows, que a menudo se comporta de forma diferente de Linux.

Muchos lenguajes como Ruby y Node.js a menudo se prescriben y ejecutan excelentes en Windows. Sin embargo, no todos los propietarios de bibliotecas Ruby Gem o node/NPM portan sus bibliotecas para que admitan Windows, y muchas tienen dependencias específicas de Linux. Esto a menudo puede dar como resultado que los sistemas que se crearon con herramientas y bibliotecas de este tipo sufran errores de compilación y, a veces, del entorno de ejecución o tengan comportamientos no deseados en Windows.

Estos son solo algunos de los problemas que han provocado que muchas personas pidan a Microsoft que mejore las herramientas de línea de comandos de Windows, lo

que nos llevó a asociarnos con Canonical para permitir que las herramientas de línea de comandos nativas de Bash y Linux se ejecuten en Windows.

¿Qué significa esto para PowerShell?

Al trabajar con proyectos de OSS, existen numerosos escenarios en los que es enormemente útil usar Bash desde un símbolo del sistema de PowerShell. La compatibilidad con Bash es complementaria y fortalece el valor de la línea de comandos en Windows, permitiendo que PowerShell y la comunidad de PowerShell aprovechen otras tecnologías populares.

Puedes leer más en el blog del equipo de PowerShell: [Bash para Windows: por qué es impresionante y qué significa para PowerShell](#) ↗.

¿Qué procesadores admite WSL?

WSL admite CPU x64 y Arm.

¿Cómo accedo a la unidad C:?

Los puntos de montaje para unidades de disco duro en el equipo local se crean automáticamente y proporcionan un acceso fácil al sistema de archivos de Windows.

/mnt/<letra> de unidad/

Un ejemplo de uso sería `cd /mnt/c` para acceder a c:\

¿Qué debo hacer para instalar el Administrador de credenciales de GIT? (¿Cómo puedo usar mis permisos de GIT de Windows en WSL?)

Consulte el tutorial [Introducción al uso de Git en Subsistema de Windows para Linux](#), que incluye una sección sobre la configuración del Administrador de credenciales de Git y el almacenamiento de tokens de autenticación en el Administrador de credenciales de Windows.

¿Cómo uso un archivo de Windows con una aplicación de Linux?

Uno de los beneficios de WSL es que puedes acceder a tus archivos mediante aplicaciones o herramientas de Windows y Linux.

WSL monta las unidades fijas de la máquina que están en la carpeta `/mnt/<drive>` en tus distribuciones de Linux. Por ejemplo, la unidad `C:` se monta en `/mnt/c/`

Si usas tus unidades montadas, puedes editar el código (por ejemplo, `C:\dev\myproj\` usando [Visual Studio](#) / o [VS Code](#)) y compilar o probar ese código en Linux accediendo a los mismos archivos a través de `/mnt/c/dev/myproj`.

Obtenga más información en [el artículo Trabajar en sistemas de archivos Windows y Linux](#).

¿Los archivos de la unidad de Linux son diferentes de la unidad de Windows montada?

1. Los archivos de la raíz de Linux (es decir, `/`) se controlan mediante WSL que se alinea con el comportamiento de Linux, incluidos, entre otros:

- Archivos que contienen caracteres de nombre de archivo de Windows que no son válidos
- Vínculos simbólicos creados para usuarios que no son administradores
- La opción de cambiar atributos de archivo a través de chmod y chown
- Distinción de mayúsculas y minúsculas en archivos o carpetas

2. Los archivos de las unidades montadas se controlan mediante Windows y tienen los siguientes comportamientos:

- Compatibilidad para la distinción de mayúsculas y minúsculas
- Todos los permisos se establecen para reflejar mejor los permisos de Windows

¿Cómo desinstalo una distribución de WSL?

Para quitar una distribución de WSL y *eliminar todos los datos asociados a esa distribución de Linux*, ejecute `wsl --unregister <distroName>` donde `<distroName>` es el nombre de la distribución de Linux, que se puede ver en la lista del `wsl -l` comando.

Además, puede desinstalar la aplicación de distribución de Linux en la máquina igual que cualquier otra aplicación de la tienda.

Para más información sobre los comandos wsl, consulte el artículo [Comandos básicos para WSL](#).

¿Cómo ejecuto un servidor OpenSSH?

OpenSSH se incluye con Windows como una característica opcional. Consulte el documento [Instalación de OpenSSH](#). Los privilegios de administrador en Windows son necesarios para ejecutar OpenSSH en WSL. Para ejecutar un servidor OpenSSH, ejecute la distribución de WSL (es decir, Ubuntu) o Terminal Windows como administrador. Hay varios recursos que abarcan escenarios ssh con WSL. Consulte los artículos de blog de Scott Hanselman: [How to SSH into a Windows 10 Machine from Linux OR Windows OR anywhere](#) (Cómo ↗ conectarse mediante SSH a WSL2 en Windows 10 desde una máquina externa ↗), [THE EASY WAY how to SSH into Bash and WSL2 on Windows 10 from an external machine](#) ↗ (Cómo usar OpenSSH integrado de Windows 10 para CONECTARSE automáticamente a una máquina Linux remota ↗).

¿Cómo cambio el idioma de visualización de WSL?

La instalación de WSL intentará cambiar automáticamente la configuración regional de Ubuntu para que coincida con la configuración regional de la instalación de Windows. Si no quieres que esto pase, puedes ejecutar este comando para cambiar la configuración regional de Ubuntu una vez finalizada la instalación. Tendrá que volver a iniciar la distribución de WSL para que este cambio surta efecto.

En el ejemplo siguiente se cambia la configuración regional a en-US:

```
Bash
```

```
sudo update-locale LANG=en_US.UTF8
```

¿Por qué no tengo acceso a Internet desde WSL?

Algunos usuarios han informado de problemas con aplicaciones de firewall específicas que bloquean el acceso a Internet en WSL. Los firewalls que dan error son:

1. Kaspersky
2. AVG
3. Avast
4. Symantec Endpoint Protection
5. F-Secure

En algunos casos, si desactivas el firewall podrás acceder. En otros casos, simplemente tener instalado el firewall parece bloquear el acceso a Internet.

¿Cómo obtengo acceso a un puerto desde WSL en Windows?

WSL comparte la dirección IP de Windows, ya que se ejecuta en Windows. Así pues, puedes obtener acceso a cualquier puerto en localhost; por ejemplo, si tienes contenido web en el puerto 1234, puedes obtener acceso <https://localhost:1234> al explorador de Windows. Para obtener más información, consulte [Acceso a aplicaciones de red](#).

¿Cómo puedo hacer una copia de seguridad de mis distribuciones de WSL o moverlas de una unidad a otra?

La mejor manera de realizar copias de seguridad o mover las distribuciones es a través de los [comandos de exportación e importación](#) disponibles en Windows versión 1809 y posteriores. Puedes exportar toda la distribución a un tarball mediante el comando `wsl --export`. Después, puede volver a importar esta distribución en WSL mediante el `wsl -import` comando, que puede asignar un nombre a una nueva ubicación de unidad para la importación, lo que le permite realizar copias de seguridad y guardar los estados de (o mover) las distribuciones de WSL.

Tenga en cuenta que los servicios de copia de seguridad tradicionales que hacen copia de seguridad de archivos en las carpetas de AppData (como Copias de seguridad de Windows) no dañarán los archivos de Linux.

¿Puedo usar WSL para escenarios de producción?

WSL se ha diseñado y diseñado para usarse con flujos de trabajo de desarrollo de bucles internos. Hay características de diseño en WSL que lo hacen excelente para este propósito, pero puede hacer que sea difícil para escenarios relacionados con la producción en comparación con otros productos. Nuestro objetivo es dejar claro cómo WSL difiere de un entorno de máquina virtual normal, por lo que puede tomar la decisión sobre si se ajusta a sus necesidades empresariales.

Las principales diferencias entre WSL y un entorno de producción tradicional son:

- WSL tiene una máquina virtual de utilidad ligera que se inicia, detiene y administra automáticamente los recursos.
- Si no tiene identificadores de archivo abiertos para los procesos de Windows, la máquina virtual WSL se apagará automáticamente. Esto significa que, si lo usa

como servidor web, conéctese mediante SSH para ejecutar el servidor y, a continuación, salir, la máquina virtual podría apagarse porque detecta que los usuarios han terminado de usarlo y limpiarán sus recursos.

- Los usuarios de WSL tienen acceso total a sus instancias de Linux. El usuario puede modificar la duración de la máquina virtual, las distribuciones de WSL registradas, etc..
- WSL proporciona automáticamente acceso a archivos de Windows.
- Las rutas de acceso de Windows se anexan a la ruta de acceso de forma predeterminada, lo que podría provocar un comportamiento inesperado para determinadas aplicaciones Linux en comparación con un entorno de Linux tradicional.
- WSL puede ejecutar ejecutables de Windows desde Linux, lo que también podría dar lugar a un entorno diferente al de una máquina virtual Linux tradicional.
- El kernel de Linux usado por WSL se actualiza automáticamente.
- El acceso a GPU en WSL se produce a través de un `/dev/dxg` dispositivo, que enruta las llamadas de GPU a la GPU de Windows. Esta configuración es diferente de una configuración tradicional de Linux.
- Hay otras diferencias más pequeñas en comparación con Linux sin sistema operativo y se espera que surjan más diferencias en el futuro, ya que se prioriza el flujo de trabajo de desarrollo de bucles internos.

¿Cómo puedo transferir mis archivos WSL de una máquina a otra?

Hay varias maneras de realizar esta tarea:

- La manera más sencilla es usar el comando para exportar la `wsl --export --vhd` distribución de WSL a un archivo VHD. Después, puede copiar este archivo en otra máquina e importarlo mediante `wsl --import --vhd`. Consulte el [documento de comandos](#) para obtener más información.
- La implementación anterior requiere mucho espacio en disco. Si no tiene mucho espacio en disco, puede usar técnicas de Linux para mover los archivos:
 - Use `tar -czf <tarballName> <directory>` para crear un tarball de los archivos. Después, puede copiar estos archivos específicos en la nueva máquina y ejecutarlos `tar -xzf <tarballName>` para extraerlos.
 - También puede exportar una lista de paquetes instalados mediante `apt`, un comando como el siguiente: `dpkg --get-selections | grep -v deinstall | awk '{print $1}' > package_list.txt` y volver a instalar esos mismos paquetes en otro equipo con un comando como `sudo apt install -y $(cat package_list.txt)` después de transferir el archivo.

WSL 2

¿WSL 2 utiliza Hyper-V? ¿Estará disponible en Windows 10 Home y Windows 11 Home?

WSL 2 está disponible en todas las SKU de escritorio donde WSL está disponible, incluidos Windows 10 Home y Windows 11 Home.

La versión más reciente de WSL usa la arquitectura de Hyper-V para habilitar su virtualización. Esta arquitectura estará disponible en el componente opcional "Plataforma de máquina virtual". Este componente opcional estará disponible en todas las SKU. Pronto podrás ver más detalles sobre esta experiencia cuando se acerque el lanzamiento de WSL 2.

¿Qué pasará con WSL 1? ¿Se abandonará?

Actualmente no tenemos planes para dejar en desuso WSL 1. Puedes ejecutar distribuciones de WSL 1 y WSL 2 en paralelo, y puedes actualizar y degradar cualquier distribución en cualquier momento. La adición de WSL 2 como una nueva arquitectura presenta una plataforma mejor para que el equipo de WSL entregue características que convierten WSL en un método increíble para ejecutar un entorno de Linux en Windows.

¿Podré ejecutar WSL 2 y otras herramientas de virtualización de terceros, como VMware o VirtualBox?

Algunas aplicaciones de terceros, como VMware y VirtualBox, no pueden funcionar cuando Hyper-V está en uso, lo que significa que no podrán ejecutarse mientras WSL 2 esté habilitado. Sin embargo, VirtualBox y VMware han publicado versiones compatibles con Hyper-V y WSL2 recientemente. Puede obtener más información sobre los [cambios de VirtualBox aquí](#) y sobre los [cambios de VMware aquí](#). Para solucionar problemas, eche un vistazo a las [discusiones de problemas de VirtualBox en el repositorio de WSL en GitHub](#).

Trabajamos de forma constante en las soluciones para admitir la integración de terceros de Hyper-V. Por ejemplo, exponemos un conjunto de API llamado [Hypervisor Platform](#) que los proveedores de virtualización externos pueden usar para hacer que su software sea compatible con Hyper-V. Esto permite que las aplicaciones usen la arquitectura de

Hyper-V para su emulación, como [Google Android Emulator](#) y VirtualBox 6 y versiones posteriores, que ahora son compatibles con Hyper-V.

Consulte el repositorio de problemas de WSL para obtener más información sobre problemas de [WSL 2 con VirtualBox 6.1](#).

*Si busca una máquina virtual Windows, VMWare, Hyper-V, VirtualBox y Parallels vm downloads están [disponibles en el Centro](#) de desarrollo de Windows.

¿Puedo acceder a la GPU en WSL 2? ¿Hay planes para aumentar la compatibilidad de hardware?

Hemos publicado compatibilidad para acceder a la GPU dentro de las distribuciones de WSL 2. Esto significa que ahora puede usar WSL para escenarios de aprendizaje automático, inteligencia artificial y ciencia de datos más fácilmente cuando hay involucrados conjuntos de macrodatos. Consulte el tutorial [Introducción a la compatibilidad con GPU](#). A partir de ahora, WSL 2 no incluye compatibilidad de serie ni compatibilidad con dispositivos USB. Estamos investigando la mejor manera de agregar estas características. Sin embargo, la compatibilidad con USB ahora está disponible a través del proyecto USBIPD-WIN. Consulte [Conexión de dispositivos USB](#) para conocer los pasos necesarios para configurar la compatibilidad con dispositivos USB.

¿Puede WSL 2 usar aplicaciones de red?

Sí, en general, las aplicaciones de red funcionarán mejor y serán más rápidas con WSL 2, ya que ofrece compatibilidad completa de llamadas del sistema. Sin embargo, la arquitectura de WSL 2 usa componentes de red virtualizados, lo que significa que WSL 2 se comportará de forma similar a una máquina virtual: las distribuciones de WSL 2 tendrán una dirección IP diferente a la máquina host (sistema operativo Windows). Para obtener más información, consulte [Acceso a aplicaciones de red con WSL](#).

¿Puedo ejecutar WSL 2 en una máquina virtual?

Sí. Debes asegurarte de que la máquina virtual tiene habilitada la virtualización anidada. Para habilitarlo en el host de Hyper-V primario, ejecuta el siguiente comando en una ventana de PowerShell con privilegios de administrador:

```
Set-VMProcessor -VMName <VMName> -ExposeVirtualizationExtensions $true
```

Asegúrate de reemplazar "<VMName>" por el nombre de tu máquina virtual.

¿Puedo usar wsl.conf en WSL 2?

WSL 2 admite el mismo archivo `wsl.conf` que se usa en WSL 1. Esto significa que todas las opciones de configuración que hayas establecido en una distribución de WSL 1, como el montaje automático de unidades de Windows, la habilitación o deshabilitación de la interoperabilidad, el cambio del directorio donde se montarán las unidades de Windows, etc., funcionarán en WSL 2. Puede obtener más información sobre las opciones de configuración de WSL en la página [Administración de distribución](#). Obtenga más información sobre la compatibilidad con el montaje de unidades, discos, dispositivos o discos duros virtuales (VHD) en el artículo [Montaje de un disco Linux en WSL 2](#).

¿Dónde puedo enviar mis comentarios?

Los problemas del repositorio de productos de WSL  le permiten:

- **Buscar problemas existentes** para ver si hay algún asociado que experimenta el mismo problema. Tenga en cuenta que, en la barra de búsqueda, puede quitar "is:open" para incluir los problemas que ya se han resuelto en la búsqueda. Considere la posibilidad de comentar o indicar que le gusta algún problema abierto para el que le gustaría expresar interés de seguir como una prioridad.
- **Presentación de un nuevo problema.** Si ha experimentado un problema con WSL y parece que no existe, puede seleccionar el botón verde *Nuevo problema* y, a continuación, elegir *WSL - Informe de errores*. Deberá incluir un título para el problema, el número de compilación de Windows (ejecute `cmd.exe /c ver` para ver el número de compilación actual), tanto si ejecuta WSL 1 o 2, el número de versión actual del kernel de Linux (ejecute `wsl.exe --status` o `cat /proc/version`), el número de versión de la distribución (ejecute `lsb_release -r`), cualquier otra versión de software implicada, los pasos de reproducción, el comportamiento esperado, el comportamiento real y los registros de diagnóstico si están disponibles y son adecuados. Para más información, consulte [Contribución a WSL](#) .
- **Para presentar una solicitud de característica**, seleccione el botón verde *Nuevo problema* y, a continuación, seleccione *Solicitud de característica*. Tendrá que atender algunas preguntas de descripción de la solicitud.

También puede:

- **Presentar un problema de documentación** mediante el [repositorio de documentos de WSL](#) . Para contribuir a los documentos de WSL, consulte la [guía para colaboradores de Microsoft Docs](#).

- Presentar un problema de Terminal Windows mediante el [repositorio de productos de Terminal Windows](#) si el problema está más relacionado con Terminal Windows, la consola de Windows o la interfaz de usuario de la línea de comandos.

Si quieres mantenerte al día con las últimas noticias de WSL, puedes hacerlo con:

- Nuestro [blog del equipo de línea de comandos](#).
- Twitter. Siga [@craigaloewen](#) en Twitter para obtener información sobre noticias, actualizaciones, etc.

Comentarios

¿Le ha resultado útil esta página?

 Sí

 No

[Proporcionar comentarios sobre el producto](#)

Solucionar problemas del subsistema de Windows para Linux

Artículo • 23/11/2023

A continuación se abordan algunos escenarios comunes de solución de problemas asociados con WSL, pero considere la posibilidad de buscar también los problemas notificados en el [repositorio de productos de WSL en GitHub](#).

Presentación de un problema, un informe de errores o una solicitud de característica

Los [problemas del repositorio de productos de WSL](#) le permiten:

- **Buscar problemas existentes** para ver si hay algún asociado que experimenta el mismo problema. Tenga en cuenta que, en la barra de búsqueda, puede quitar "is:open" para incluir los problemas que ya se han resuelto en la búsqueda. Considere la posibilidad de comentar o indicar que le gusta algún problema abierto para el que le gustaría expresar interés de seguir como una prioridad.
- **Presentación de un nuevo problema.** Si ha experimentado un problema con WSL y parece que no existe, puede seleccionar el botón verde *Nuevo problema* y, a continuación, elegir *WSL - Informe de errores*. Deberá incluir un título para el problema, el número de compilación de Windows (ejecute `cmd.exe /c ver` para ver el número de compilación actual), tanto si ejecuta WSL 1 o 2, el número de versión actual del kernel de Linux (ejecute `wsl.exe --status` o `cat /proc/version`), el número de versión de la distribución (ejecute `lsb_release -r`), cualquier otra versión de software implicada, los pasos de reproducción, el comportamiento esperado, el comportamiento real y los registros de diagnóstico si están disponibles y son adecuados. Para más información, consulte [Contribución a WSL](#).
- **Para presentar una solicitud de característica**, seleccione el botón verde *Nuevo problema* y, a continuación, seleccione *Solicitud de característica*. Tendrá que atender algunas preguntas de descripción de la solicitud.

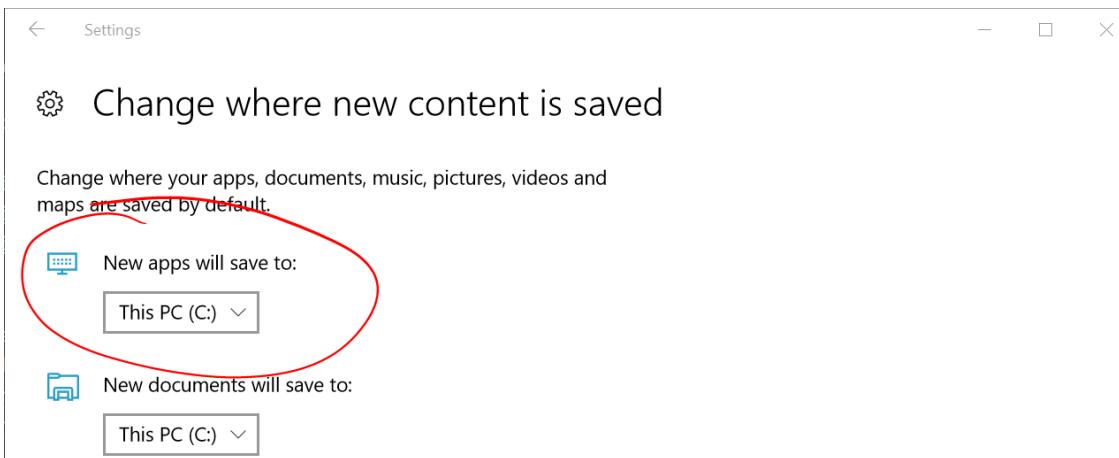
También puede:

- **Presentar un problema de documentación** mediante el [repositorio de documentos de WSL](#). Para contribuir a los documentos de WSL, consulte la [guía para colaboradores de Microsoft Docs](#).

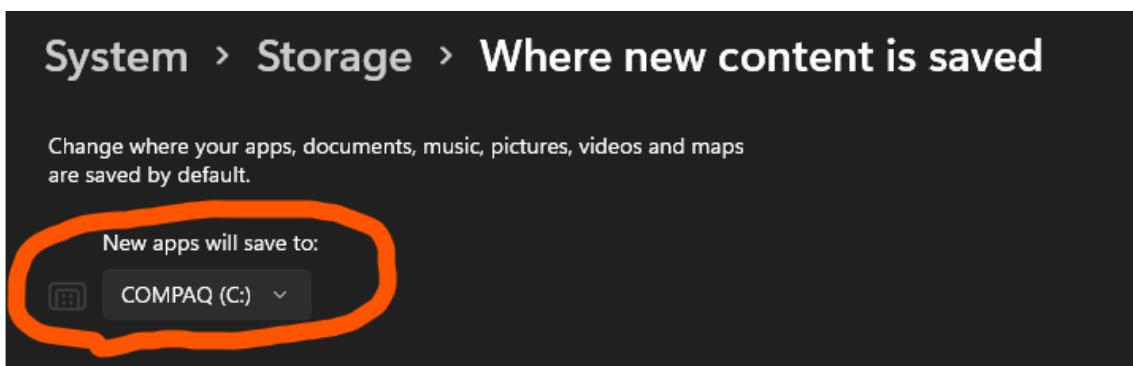
- Presentar un problema de Terminal Windows mediante el [repositorio de productos de Terminal Windows](#) si el problema está más relacionado con Terminal Windows, la consola de Windows o la interfaz de usuario de la línea de comandos.

Problemas de instalación

- Error 0x80070003 en la instalación
 - El Subsistema de Windows para Linux solo se ejecuta en la unidad del sistema (normalmente se trata de la unidad `c:`). Asegúrate de que las distribuciones estén almacenadas en la unidad del sistema:
 - En Windows 10, abra **Configuración ->Sistema ->Almacenamiento ->Más opciones de almacenamiento: Cambiar la ubicación de almacenamiento del contenido nuevo**



- En Windows 11, abra **Configuración ->Sistema ->Almacenamiento ->Configuración avanzada de almacenamiento ->Ubicación de almacenamiento del contenido nuevo**

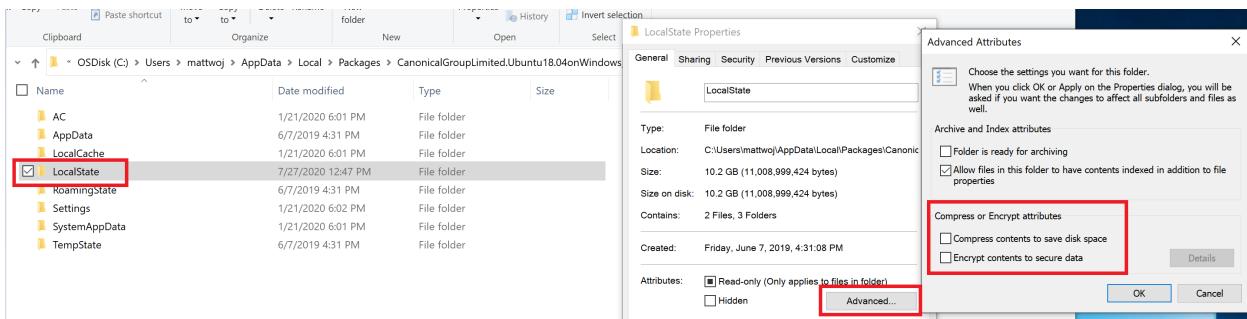


- Error 0x8007019e de WslRegisterDistribution
 - El componente opcional del Subsistema de Windows para Linux no está habilitado:
 - Abra el **Panel de control ->Programas y características ->Activar o desactivar la característica de Windows** -> Seleccione Subsistema de Windows para Linux o use el cmdlet de PowerShell mencionado al comienzo de este artículo.

- **Error en la instalación 0x80070003 o 0x80370102**
 - Asegúrate de que la virtualización está habilitada dentro del BIOS del equipo. Las instrucciones sobre cómo hacerlo variarán de un equipo a otro y lo más probable es que esta característica esté en opciones relacionadas con la CPU.
 - WSL2 requiere que la CPU admita la característica de traducción de direcciones de segundo nivel (SLAT), que se introdujo en procesadores Intel Nehalem (primera generación Intel Core) y AMD Opteron. Las CPU anteriores (como Intel Core 2 Duo) no podrán ejecutar WSL2, incluso si la plataforma de máquina virtual está instalada correctamente.
- **Error al intentar actualizar:** `Invalid command line option: wsl --set-version`

Ubuntu 2

 - Asegúrese de que el Subsistema de Windows para Linux esté habilitado y de que usa la compilación de Windows 18362 o posterior. Para habilitar WSL, ejecute este comando en un símbolo del sistema de PowerShell con privilegios de administrador: `Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux`.
- **La operación solicitada no se pudo completar debido a una limitación del sistema de disco virtual. Los archivos de disco duro virtual deben estar sin comprimir y sin cifrar y no deben ser dispersos.**
 - Anule la selección de la casilla "Compress contents" ("Comprimir contenido") (y también la de "Cifrar los contenidos" si está activada). Para ello, abra la carpeta de perfil de la distribución de Linux. Debe encontrarse en una carpeta del sistema de archivos de Windows, como
`%USERPROFILE%\AppData\Local\Packages\CanonicalGroupLimited....`
 - En este perfil de distribución de Linux, debe haber una carpeta denominada LocalState. Haga clic con el botón derecho en ella para mostrar un menú de opciones. Seleccione Propiedades > Opciones avanzadas y, a continuación, asegúrese de que las casillas "Comprimir contenido para ahorrar espacio en disco" y "Cifrar contenido para proteger datos" no estén seleccionadas (activadas). Si se le pregunta si quiere aplicar esto solo a la carpeta actual o a todas las subcarpetas y archivos, seleccione "solo esta carpeta", ya que solo quiere borrar la marca de compresión. A continuación, el comando `wsl --set-version` debería funcionar.



① Nota

En mi caso, la carpeta LocalState de la distribución de Ubuntu 18.04 se encontraba en C:\Users<my-user-name>\AppData\Local\Packages\CanonicalGroupLimited.Ubuntu18.04onWindows_79rhkp1fndgsc.

Consulte el [subproceso n.º 4103 de la documentación sobre WSL en GitHub](#), en el que se realiza el seguimiento de este problema para obtener información actualizada.

- El término 'wsl' no se reconoce como nombre de cmdlet, función, archivo de script o programa ejecutable.
 - Asegúrate de que el [componente opcional del Subsistema de Windows para Linux esté instalado](#). Además, si usa un dispositivo ARM64 y ejecuta este comando desde PowerShell, recibirá este error. En su lugar, ejecuta `wsl.exe` desde [PowerShell Core](#) o el símbolo del sistema.
- Error: el Subsistema de Windows para Linux no tiene ninguna distribución instalada.
 - Si recibe este error después de haber instalado las distribuciones de WSL:
 1. Ejecute la distribución al menos una vez antes de invocarla desde la línea de comandos.
 2. Compruebe si puede ejecutar cuentas de usuario independientes. La ejecución de la cuenta de usuario principal con permisos elevados (en modo de administrador) no debería producir este error, pero debe asegurarse de que no ejecuta accidentalmente la cuenta de administrador integrada que viene con Windows. Se trata de una cuenta de usuario independiente y no mostrará ninguna distribución de WSL instalada por diseño. Para obtener más información, consulte [Habilitar y deshabilitar la cuenta integrada de administrador](#).
 3. El archivo ejecutable de WSL solo se instala en el directorio del sistema nativo. Cuando se ejecuta un proceso de 32 bits en un Windows de 64 bits (o

en ARM64, cualquier combinación no nativa), el proceso no nativo hospedado realmente ve una carpeta System32 diferente. (El que ve un proceso de 32 bits en Windows x64 se almacena en disco en \\Windows\\SysWOW64). Para acceder al sistema "nativo" System32 desde un proceso hospedado, busque en la carpeta virtual: \\Windows\\sysnative. En realidad, no estará presente en el disco, pero el solucionador de rutas de acceso del sistema de archivos la encontrará.

- **Error: Esta actualización solo se aplica a las máquinas con el Subsistema de Windows para Linux.**
 - Para instalar el paquete MSI de actualización del kernel de Linux, WSL es necesario y debe habilitarse primero. Si se produce un error, verá el mensaje: This update only applies to machines with the Windows Subsystem for Linux.
 - Hay tres posibles motivos para ver este mensaje:
 1. Todavía tiene una versión antigua de Windows que no es compatible con WSL 2. Consulte el paso 2 para conocer los requisitos de la versión y los vínculos de la actualización.
 2. WSL no está habilitado. Tendrá que volver al paso 1 y asegurarse de que la característica WSL opcional está habilitada en la máquina.
 3. Después de habilitar WSL, es necesario un reinicio para que surta efecto. Reinicie la máquina e inténtelo de nuevo.
- **Error: WSL 2 requiere una actualización en su componente de kernel. Para obtener más información, visite <https://aka.ms/wsl2kernel>.**
 - Si falta el paquete del kernel de Linux en la carpeta %SystemRoot%\\system32\\lxss\\tools, se mostrará este error. Para resolverlo, instale el paquete MSI de actualización del kernel de Linux en el paso 4 de estas instrucciones de instalación. Es posible que deba desinstalar el paquete MSI desde "Agregar o quitar programas" e instalarlo de nuevo.

Problemas comunes

Tengo la versión 1903 de Windows 10 y sigo sin ver las opciones de WSL 2

Probablemente, se debe a que la máquina aún no ha adquirido la adaptación para WSL 2. La forma más sencilla de resolverlo es dirigirse a Configuración de Windows y

hacer clic en "Buscar actualizaciones" para instalar las actualizaciones más recientes en el sistema. Consulte las [instrucciones completas sobre cómo adquirir la adaptación](#).

Si pulsa "Buscar actualizaciones" y sigue sin recibir la actualización, puede [instalar KB KB4566116 manualmente](#).

Error: 0x1bc cuando `wsl --set-default-version 2`

Puede ocurrir cuando el valor de "Mostrar idioma" o "Configuración regional del sistema" no es Inglés.

PowerShell

```
wsl --set-default-version 2
Error: 0x1bc
For information on key differences with WSL 2 please visit
https://aka.ms/wsl2
```

El error real de `0x1bc` es:

PowerShell

```
WSL 2 requires an update to its kernel component. For information please
visit https://aka.ms/wsl2kernel
```

Para obtener más información, consulte el problema [5749](#).

No se puede acceder a los archivos WSL desde Windows

Un servidor de archivos de protocolo 9P proporciona el servicio en el lado de Linux para permitir que Windows tenga acceso al sistema de archivos de Linux. Si no puede acceder a WSL con `\wsl$` en Windows, podría deberse a que 9P no se inició correctamente.

Para comprobarlo, puede consultar los registros de inicio mediante: `dmesg |grep 9p`, lo que le mostrará los errores. Una salida correcta se parece a la siguiente:

Bash

```
[ 0.363323] 9p: Installing v9fs 9p2000 file system support
[ 0.363336] FS-Cache: Netfs '9p' registered for caching
[ 0.398989] 9pnet: Installing 9P2000 support
```

Consulte [este subproceso de GitHub](#) para más explicaciones sobre este problema.

No se puede iniciar la distribución WSL 2 y solo se ve "WSL 2" en la salida

Si el idioma para mostrar no es el inglés, es posible que vea una versión truncada de un texto de error.

PowerShell

```
C:\Users\me>wsl  
WSL 2
```

Para resolver este problema, visite <https://aka.ms/wsl2kernel> e instale el kernel manualmente siguiendo las instrucciones de esa página de documentación.

command not found al ejecutar el .exe de Windows en Linux

Los usuarios pueden ejecutar archivos ejecutables de Windows como notepad.exe directamente desde Linux. A veces, puede que se encuentre con el mensaje "comando no encontrado", como en el siguiente ejemplo:

Bash

```
$ notepad.exe  
-bash: notepad.exe: command not found
```

Si no hay ninguna ruta de acceso Win32 en el parámetro \$PATH, interop no encontrará el archivo .exe. Para comprobarlo, puede ejecutar `echo $PATH` en Linux. Se espera que aparezca una ruta de acceso Win32 (por ejemplo, /mnt/c/Windows) en la salida. Si no se muestra ninguna ruta de acceso de Windows, lo más probable es que el shell de Linux haya sobrescrito el parámetro PATH.

A continuación se muestra un ejemplo en el que la ruta /etc/profile en Debian contribuyó al problema:

Bash

```
if [ "`id -u`" -eq 0 ]; then  
    PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"  
else  
    PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games"  
fi
```

La manera correcta de hacerlo en Debian consiste en quitar las líneas anteriores. También puede anexar el parámetro \$PATH durante la asignación, como se indica a continuación. No obstante, esto genera [otros problemas](#) con WSL y VSCode.

Bash

```
if [ "`id -u`" -eq 0 ]; then
    PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:$PATH"
else
    PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games:$PATH"
fi
```

Si quiere obtener más información, consulte las incidencias [5296](#) y [5779](#).

"Error: 0x80370102 The virtual machine could not be started because a required feature is not installed" (Error: 0x80370102 No se pudo iniciar la máquina virtual porque una característica necesaria no está instalada).

Habilite la característica de Windows de plataforma de máquina virtual y asegúrese de que la virtualización esté habilitada en el BIOS.

1. Consulte [Requisitos de sistema de Hyper-V](#).
2. Si la máquina es una máquina virtual, habilite [virtualización anidada](#) manualmente. Inicie PowerShell con el administrador y ejecute el comando siguiente, reemplazando <VMName> por el nombre de la máquina virtual en el sistema host (puede encontrar el nombre en el Administrador de Hyper-V):

PowerShell

```
Set-VMProcessor -VMName <VMName> -ExposeVirtualizationExtensions $true
```

3. Siga las instrucciones del fabricante del equipo sobre cómo habilitar la virtualización. En general, esto puede implicar el uso del BIOS del sistema para asegurarse de que estas características se habilitan en la CPU. Las instrucciones para este proceso pueden variar de un equipo a otro. Consulte [este artículo](#) de Bleeping Computer para ver un ejemplo.
4. Reinicie el equipo después de habilitar el componente opcional [Virtual Machine Platform](#).

5. Asegúrese de que el inicio del hipervisor está habilitado en la configuración de arranque. Puede validar esto ejecutando lo siguiente (PowerShell con privilegios elevados):

```
PowerShell
```

```
bcdedit /enum | findstr -i hypervisorlauchtype
```

Si ve `hypervisorlauchtype Off`, el hipervisor está deshabilitado. Para habilitar este elemento, ejecute una instancia de PowerShell con privilegios elevados:

```
bcdedit /set hypervisorlauchtype Auto
```

6. Además, si tiene instalados hipervisores de terceros (por ejemplo, VMware o VirtualBox), asegúrese de que tengan las versiones más recientes que pueden admitir ([VMware 15.5.5+ ↗](#) y [VirtualBox 6+ ↗](#)) o de que estén desactivados.

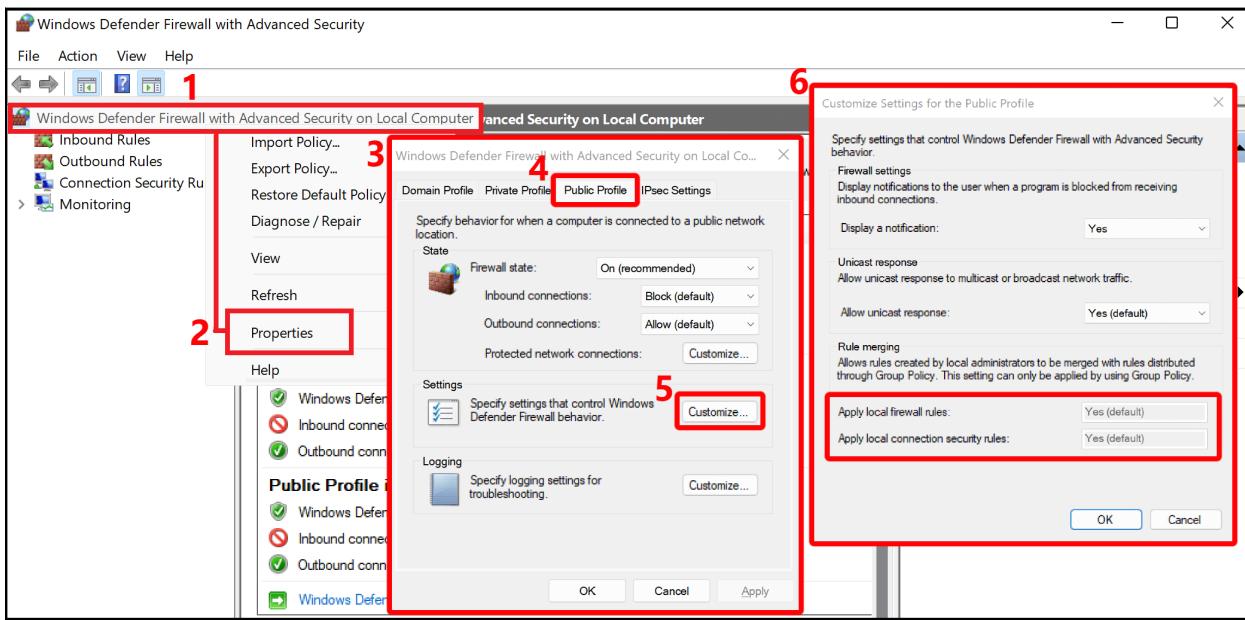
7. Si recibe este error en una máquina virtual de Azure, compruebe que el [inicio seguro](#) está deshabilitado. [La virtualización anidada no se admite en máquinas virtuales de Azure](#).

Obtenga más información sobre cómo [configurar la virtualización anidada](#) al ejecutar Hyper-V en una máquina virtual.

El subsistema WSL no tiene ninguna conexión de red en mi máquina de trabajo o en un entorno empresarial

Es posible que, en los entornos empresariales, la [configuración de Firewall de Windows Defender](#) esté establecida para bloquear el tráfico de red no autorizado. Si la [combinación de reglas locales](#) se establece en "No", las redes de WSL no funcionarán de forma predeterminada. Para permitir las conexiones, el administrador deberá agregar una regla de firewall.

Para confirmar la configuración de la combinación de reglas locales, siga estos pasos:



1. Abra "Firewall de Windows Defender con seguridad avanzada" (*que no es lo mismo que "Firewall de Windows Defender", del Panel de control*).
2. Haga clic con el botón derecho en la pestaña "Firewall de Windows Defender con seguridad avanzada en equipo local".
3. Seleccione el elemento "Propiedades".
4. Seleccione la pestaña "Perfil público" en la nueva ventana que se abrirá.
5. Seleccione el elemento "Personalizar", que se ubica en la sección "Configuración".
6. Compruebe la ventana "Personalizar configuración para el perfil público" que se abrirá para asegurarse de que el parámetro "Combinación de reglas" está establecido en "No". Esto bloqueará el acceso al subsistema WSL.

Puede encontrar instrucciones sobre cómo cambiar esta configuración de Firewall en [Configuración del firewall](#) de Hyper-V.

WSL no tiene conectividad de red si se conecta a una VPN.

Si después de conectar a una VPN en Windows, Bash pierde la conectividad de red, prueba esta solución desde dentro de Bash. Esta solución alternativa te permitirá invalidar manualmente la resolución de DNS a través de `/etc/resolv.conf`.

1. Toma nota del servidor DNS de la VPN con `ipconfig.exe /all`
2. Haz una copia del `resolv.conf` existente `sudo cp /etc/resolv.conf /etc/resolv.conf.new`
3. Desvincula el elemento `resolv.com` actual `sudo unlink /etc/resolv.conf`
4. `sudo mv /etc/resolv.conf.new /etc/resolv.conf`
5. Edite `/etc/wsl.conf` y agregue este contenido al archivo. (Puede encontrar más información sobre esta configuración en [Configuración avanzada](#)).

```
[network]
generateResolvConf=false
```

6. Abre `/etc/resolv.conf` y
 - a. Elimine la primera línea del archivo que tiene un comentario que describe la generación automática
 - b. Agrega la entrada de DNS de (1) anterior como primera entrada de la lista de servidores DNS.
 - c. Cierra el archivo.

Una vez que hayas desconectado la VPN, tendrás que revertir los cambios a `/etc/resolv.conf`. Para ello, haz lo siguiente:

1. `cd /etc`
2. `sudo mv resolv.conf resolv.conf.new`
3. `sudo ln -s ../../run/resolvconf/resolv.conf resolv.conf`

Problemas de Cisco Anyconnect VPN con WSL en modo NAT

Cisco AnyConnect VPN modifica las rutas de forma que impide que NAT funcione. Hay una solución alternativa específica de WSL 2: Consulte la [Guía del administrador de Cisco AnyConnect Secure Mobility Client, versión 4.10: Solución de problemas de AnyConnect](#).

Problemas de conectividad de WSL con VPN cuando el modo de red reflejado está activado

El modo de red reflejado es actualmente una [configuración experimental en la configuración de WSL](#). La arquitectura de red NAT tradicional de WSL se puede actualizar a un modo de red completamente nuevo denominado "Modo de red reflejado". Cuando el experimental `networkingMode` se establece en `mirrored`, las interfaces de red que tiene en Windows se reflejan en Linux para mejorar la compatibilidad. Obtenga más información en el blog de la línea de comandos: [Actualización de septiembre de 2023 de WSL](#).

Se ha comprobado y confirmado que algunas VPN son incompatibles con WSL, entre ellas:

- "Bitdefender" versión 26.0.2.1
- "OpenVPN" versión 2.6.501
- "Mcafee Safe Connect" versión 2.16.1.124

Consideraciones al usar autoProxy para la creación de reflejos de HttpProxy en WSL

La creación de reflejo del proxy HTTP/S se puede configurar mediante la configuración `autoProxy` de la [sección experimental del archivo de configuración de WSL](#). Al aplicar esta configuración, tenga en cuenta estas consideraciones:

- **Proxy PAC:** WSL configurará la configuración en Linux estableciendo la variable de entorno "`WSL_PAC_URL`". Linux no admite servidores proxy PAC de forma predeterminada.
- **Interacciones con WSLENV:** las variables de entorno definidas por el usuario tienen prioridad sobre las especificadas por esta característica.

Cuando se habilita, se aplica lo siguiente a la configuración de proxy en las distribuciones de Linux:

- La variable de entorno de Linux, `HTTP_PROXY`, se establece en uno o varios servidores proxy HTTP que se encuentran instalados en la configuración del proxy HTTP de Windows.
- La variable de entorno de Linux, `HTTPS_PROXY`, se establece en uno o varios servidores proxy HTTPS que se encuentran instalados en la configuración del proxy HTTP de Windows.
- La variable de entorno de Linux, `NO_PROXY`, se establece para omitir los servidores proxy HTTP/S que se encuentran en los destinos de configuración de Windows.
- Cada variable de entorno, excepto `WSL_PAC_URL`, se establece en minúsculas y mayúsculas. Por ejemplo, `HTTP_PROXY` y `http_proxy`.

Obtenga más información en el blog de la línea de comandos: [Actualización de septiembre de 2023 de WSL](#).

Consideraciones sobre redes con tunelización DNS

Cuando WSL no se puede conectar a Internet, puede deberse a que la llamada DNS al host de Windows está bloqueada. Esto se debe a que la configuración de red existente bloquea el paquete de red para DNS enviado por la máquina virtual WSL al host de Windows. La tunelización DNS soluciona este problema utilizando una característica de virtualización para comunicarse con Windows directamente, lo que permite resolver el

nombre DNS sin enviar un paquete de red. Esta característica debería mejorar la compatibilidad de la red y permitirle obtener una mejor conectividad a Internet aunque tenga una VPN, una configuración de firewall específica u otras configuraciones de red.

La tunelización DNS se puede configurar mediante la configuración `dnsTunneling` de la [sección experimental del archivo de configuración de WSL](#). Al aplicar esta configuración, tenga en cuenta estas consideraciones:

- Docker nativo puede tener problemas de conectividad en WSL cuando está habilitada la tunelización DNS, si la red tiene una directiva para bloquear el tráfico DNS a: 8.8.8.8
- Si usa una VPN con WSL, active la tunelización DNS. Muchas VPN usan directivas NRPT, que solo se aplican a las consultas de DNS de WSL cuando está habilitada la tunelización DNS.
- El archivo `/etc/resolv.conf` de la distribución de Linux tiene una limitación máxima de 3 servidores DNS, mientras que Windows puede usar más de 3 servidores DNS. El uso de la tunelización DNS elimina esta limitación: Linux puede usar ahora todos los servidores DNS de Windows.
- WSL usará sufijos DNS de Windows en el orden siguiente (similar al orden usado por el cliente DNS de Windows):
 1. Sufijos DNS globales
 2. Sufijos DNS complementarios
 3. Sufijos DNS por interfaz
 4. Si el cifrado DNS (DoH, DoT) está habilitado en Windows, el cifrado se aplicará a las consultas de DNS de WSL. Si los usuarios quieren habilitar DoH, DoT dentro de Linux, deben deshabilitar la tunelización DNS.
- Las consultas de DNS desde contenedores Docker (Docker Desktop o Docker nativo ejecutándose en WSL) omitirán la tunelización DNS. No se puede aprovechar la tunelización DNS para aplicar la configuración y las directivas DNS del host al tráfico DNS de Docker.
- Docker Desktop tiene su propia manera (diferente de la tunelización DNS) de aplicar la configuración y las directivas DNS del host a las consultas de DNS desde contenedores de Docker.

Obtenga más información en el blog de la línea de comandos: [Actualización de septiembre de 2023 de WSL](#).

Problemas con la dirección del tráfico entrante recibido por el host de Windows a la máquina virtual WSL

Cuando se usa el modo de red reflejado (el conjunto experimental `networkingMode` en `mirrored`), el tráfico entrante recibido por el host de Windows nunca se dirige a la máquina virtual Linux. Este tráfico es el siguiente:

- Puerto UDP 68 (DHCP)
- Puerto TCP 135 (resolución de punto de conexión DCE)
- Puerto UDP 5353 (mDNS)
- Puerto TCP 1900 (UPnP)
- Puerto TCP 2869 (SSDP)
- Puerto TCP 5004 (RTP)
- Puerto TCP 3702 (WSD)
- Puerto TCP 5357 (WSD)
- Puerto TCP 5358 (WSD)

WSL configurará automáticamente ciertas opciones de red de Linux al usar el modo de red reflejado. No se admiten las configuraciones de usuario de estas opciones mientras se usa el modo de red reflejado. Esta es la lista de opciones que WSL configurará:

Nombre de la opción de configuración	Valor
https://sysctl-explorer.net/net/ipv4/accept_local/	Habilitado (1)
https://sysctl-explorer.net/net/ipv4/route_localnet/	Habilitado (1)
https://sysctl-explorer.net/net/ipv4/rp_filter/	Deshabilitado (0)
https://sysctl-explorer.net/net/ipv6/accept_ra/	Deshabilitado (0)
https://sysctl-explorer.net/net/ipv6/autoconf/	Deshabilitado (0)
https://sysctl-explorer.net/net/ipv6/use_tempaddr/	Deshabilitado (0)
addr_gen_mode	Deshabilitado (0)
disable_ipv6	Deshabilitado (0)
https://sysctl-explorer.net/net/ipv4/arp_filter/	Habilitado (1)

Problemas del contenedor de Docker en WSL2 con el modo de red reflejado habilitado cuando se ejecuta en el espacio de nombres de red predeterminado

Hay un problema conocido en el que los contenedores de Docker Desktop con puertos publicados (`docker run -publish/-p`) no se crearán. El equipo de WSL está trabajando con el equipo de Docker Desktop para solucionar este problema. Para solucionar el

problema, use el espacio de nombres de red del host en el contenedor Docker. Establezca el tipo de red a través de la opción "--network host" que se usa en el comando "docker run". Una solución alternativa consiste en enumerar el número de puerto publicado en el valor de `ignoredPorts` de la sección experimental del archivo de configuración de WSL.

Sufijos DNS en WSL

Según las configuraciones del archivo `.wslconfig`, WSL tendrá el siguiente comportamiento wrt DNS sufijos:

Cuando networkingMode está establecido en NAT:

Caso 1) De forma predeterminada, no hay ningún sufijo DNS configurado en Linux

Caso 2) Si la tunelización DNS está habilitada (`dnsTunneling` se establece en `true` en `.wslconfig`) Todos los sufijos DNS de Windows están configurados en Linux, en el valor "search" de `/etc/resolv.conf`

Los sufijos se configuran en `/etc/resolv.conf` en el orden siguiente, similar al orden en que el cliente DNS de Windows intenta sufijos al resolver un nombre: sufijos DNS globales primero, luego sufijos DNS complementarios y, a continuación, sufijos DNS por interfaz.

Cuando se produce un cambio en los sufijos DNS de Windows, ese cambio se reflejará automáticamente en Linux

Caso 3) Si la tunelización DNS está deshabilitada y el proxy DNS de SharedAccess está deshabilitado (`dnsTunneling` se establece en `false` y `dnsProxy` se establece en `false` en `.wslconfig`) Se configura un único sufijo DNS en Linux, en el valor "dominio" de `/etc/resolv.conf`

Cuando se produce un cambio en los sufijos DNS de Windows, ese cambio no se refleja en Linux

El sufijo DNS único configurado en Linux se elige entre los sufijos DNS por interfaz (se omiten los sufijos DNS globales y complementarios)

Si Windows tiene varias interfaces, se usa una heurística para elegir el sufijo DNS único que se configurará en Linux. Por ejemplo, si hay una interfaz VPN en Windows, el sufijo se elige de esa interfaz. Si no hay ninguna interfaz VPN, se elige el sufijo de la interfaz que es más probable que proporcione conectividad a Internet.

Cuando networkingMode está establecido en Mirrorred:

Todos los sufijos DNS de Windows están configurados en Linux, en la configuración "search" de /etc/resolv.conf

Los sufijos se configuran en /etc/resolv.conf en el mismo orden que en el caso 2) desde el modo NAT.

Cuando se produce un cambio en los sufijos DNS de Windows, ese cambio se reflejará automáticamente en Linux

Nota: Los sufijos DNS complementarios se pueden configurar en Windows mediante [SetInterfaceDnsSettings - Aplicaciones Win32 | Microsoft Learn](#), con la marca **DNS_SETTING_SUPPLEMENTAL_SEARCH_LIST** establecida en el parámetro **Settings**

Solución de problemas de DNS en WSL

La configuración de DNS predeterminada cuando WSL inicia un contenedor en modo NAT es que el dispositivo NAT del host de Windows actúe como el "servidor DNS" para el contenedor WSL. Cuando las consultas DNS se envían desde el contenedor WSL a ese dispositivo NAT en el host de Windows, el paquete DNS se reenvía desde el dispositivo NAT al servicio de acceso compartido en el host; la respuesta se envía en la dirección inversa al contenedor WSL. Este proceso de reenvío de paquetes para el acceso compartido requiere una regla de firewall para permitir ese paquete DNS de entrada, que el servicio HNS crea cuando WSL solicita inicialmente a HNS que cree la red virtual NAT para su contenedor WSL.

Debido a esta NAT: diseño de acceso compartido, hay algunas configuraciones conocidas que pueden interrumpir la resolución de nombres de WSL.

1. Una empresa puede insertar una directiva que no permita reglas de firewall definidas localmente, solo lo que permite reglas definidas por directivas empresariales.

Cuando una empresa establece esto, se omite la regla de firewall creada por HNS, ya que es una regla definida localmente. Para que esta configuración funcione, Enterprise debe crear una regla de firewall para permitir el puerto UDP 53 al servicio de acceso compartido o WSL se puede establecer para usar túnel DNS. Puede ver si está configurado para no permitir reglas de firewall definidas localmente mediante la ejecución de lo siguiente. Tenga en cuenta que esto mostrará la configuración de los 3 perfiles: Dominio, Privado y Público. Si se establece en cualquier perfil, los paquetes se bloquearán si la vNIC de WSL se asigna a ese perfil (el valor predeterminado es Público). Este es solo un fragmento de código del primer perfil de firewall que se devuelve en PowerShell:

PowerShell

```
PS C:\> Get-NetFirewallProfile -PolicyStore ActiveStore
Name : Domain
Enabled : True
DefaultInboundAction : Block
DefaultOutboundAction : Allow
AllowInboundRules : True
AllowLocalFirewallRules : False
```

AllowLocalFirewallRules:False means the locally defined firewall rules, like that by HNS, will not be applied or used.

2. Y Enterprise puede insertar la directiva de grupo y la configuración de directivas MDM que bloquean todas las reglas de entrada.

Esta configuración invalida cualquier regla de firewall de entrada permitida. Por lo tanto, esta configuración bloqueará la regla de firewall UDP creada por HNS y, por tanto, impedirá que WSL resuelva los nombres. Para que esta configuración funcione, **WSL debe establecerse para usar la tunelización DNS**. Esta configuración siempre bloqueará el proxy DNS NAT.

Desde la directiva de grupo:

Configuración del equipo \ Plantillas administrativas \ Red \ Conexiones de red \ Firewall de Windows Defender \ Perfil de dominio | Perfil estándar

"Firewall de Windows Defender: No permitir excepciones" - Habilitado

Desde la directiva MDM:

./Vendor/MSFT/Firewall/MdmStore/PrivateProfile/Shielded

./Vendor/MSFT/Firewall/MdmStore/DomainProfile/Shielded

./Vendor/MSFT/Firewall/MdmStore/PublicProfile/Shielded

Puede ver si está configurado para no permitir ninguna regla de firewall de entrada ejecutando lo siguiente (vea las advertencias anteriores en perfiles de firewall). Este es solo un fragmento de código del primer perfil de firewall que se devuelve en PowerShell:

powerShell

```
PS C:\> Get-NetFirewallProfile -PolicyStore ActiveStore
Name : Domain
```

Enabled	: True
DefaultInboundAction	: Block
DefaultOutboundAction	: Allow
AllowInboundRules	: False

AllowInboundRules: False means that no inbound Firewall rules will be applied.

3. Un usuario pasa por las aplicaciones de configuración de seguridad de Windows y comprueba el control de "Bloquea todas las conexiones entrantes, incluidas las de la lista de aplicaciones permitidas."

Windows admite una participación de usuario para la misma configuración que una empresa a la que se hace referencia en el #2 anterior. Los usuarios pueden abrir la "página Configuración de seguridad" de Windows, selecciona la opción "Firewall & protección de red", selecciona el perfil de firewall que quieren configurar (dominio, privado o público) y en "Conexiones entrantes" compruebe el control con la etiqueta "Bloquea todas las conexiones entrantes, incluidas las de la lista de aplicaciones permitidas."

Si se establece para el perfil público (este es el perfil predeterminado para la vNIC de WSL), se bloqueará la regla de firewall creada por HNS para permitir que los paquetes UDP accedan compartidos.

Debe desactivarse para que la configuración del proxy DNS NAT funcione desde WSL o WSL se puede establecer para usar la tunelización DNS.

4. La regla de firewall de HNS para permitir que los paquetes DNS tengan acceso compartido pueden dejar de ser válidos, haciendo referencia a un identificador de interfaz WSL anterior. Este es un error en HNS que se ha corregido con la versión más reciente de Windows 11. En versiones anteriores, si esto ocurre, no se puede detectar fácilmente, pero tiene una solución sencilla:

- Detener WSL

1. `wsl.exe -shutdown`

- Elimine la antigua regla de firewall de HNS. Este comando de PowerShell debe funcionar en la mayoría de los casos:


```
Get-NetFirewallRule -Name "HNS*" | Get-NetFirewallPortFilter | where Protocol -eq UDP | where LocalPort -eq 53 | Remove-NetFirewallRule
```
- Eliminación de todos los puntos de conexión de HNS

1. Nota: si se usa HNS para administrar otros contenedores, como MDAG o Windows Sandbox, también se deben detener.

2. `hnsdiag.exe delete all`

- Reinicio o reinicio del servicio HNS
- Cuando se reinicia WSL, HNS creará nuevas reglas de firewall, que tienen como destino correctamente la interfaz WSL.

Iniciar WSL o instalar una distribución devuelve un código de error

Siga las instrucciones para [recopilar registros de WSL](#) en el repositorio de WSL en GitHub para recopilar registros detallados y archivar un problema en nuestro GitHub.

Actualización de WSL

Hay dos componentes del Subsistema de Windows para Linux que pueden requerir actualización.

1. Para actualizar el Subsistema de Windows para Linux, use el comando `wsl --update` en PowerShell o CMD.
2. Para actualizar los archivos binarios de usuario de distribución de Linux específicos, use el comando `apt-get update | apt-get upgrade` en la distribución de Linux que quiere actualizar.

Errores de actualización de apt-get

Algunos paquetes usan características que aún no se han implementado. `udev`, por ejemplo, todavía no se admite y produce varios errores de tipo `apt-get upgrade`.

Para corregir los problemas relacionados con `udev`, sigue estos pasos:

1. Escribe lo siguiente en `/usr/sbin/policy-rc.d` y guarda los cambios.

Bash

```
#!/bin/sh
exit 101
```

2. Agrega los permisos de ejecución en `/usr/sbin/policy-rc.d`:

Bash

```
chmod +x /usr/sbin/policy-rc.d
```

- Ejecute los siguientes comandos:

Bash

```
dpkg-divert --local --rename --add /sbin/initctl  
ln -s /bin/true /sbin/initctl
```

"Error: 0x80040306" en la instalación

Esto se relaciona con el hecho de que no se admite la consola heredada. Para desactivar la consola heredada:

- Abre cmd.exe.
- Haga clic con el botón derecho en barra de título -> Propiedades -> y desactive Usar consola heredada.
- Haga clic en Aceptar

"Error: 0x80040154" después de una actualización de Windows

La característica Subsistema de Windows para Linux puede deshabilitarse durante una actualización de Windows. Si esto ocurre, debes volver a habilitar la característica de Windows. Las instrucciones para habilitar el Subsistema de Windows para Linux se pueden encontrar en la [guía de instalación manual](#).

Cambiar el idioma de visualización

La instalación de WSL intentará cambiar automáticamente la configuración regional de Ubuntu para que coincida con la configuración regional de la instalación de Windows. Si no quieres que esto pase, puedes ejecutar este comando para cambiar la configuración regional de Ubuntu una vez finalizada la instalación. Ten en cuenta que tendrás que reiniciar bash.exe para que este cambio surta efecto.

En el ejemplo siguiente se cambia la configuración regional a en-US:

Bash

```
sudo update-locale LANG=en_US.UTF8
```

Problemas de instalación después de una restauración del sistema de Windows

1. Elimina la carpeta `%windir%\System32\Tasks\Microsoft\Windows\Windows Subsystem for Linux.`
Nota: No lo hagas si tu característica opcional está completamente instalada y en funcionamiento.
2. Habilitar la característica opcional WSL (si aún no lo está)
3. Reiniciar
4. `Ixrun /uninstall /full`
5. Instala bash.

Sin acceso a Internet en WSL

Algunos usuarios han informado de problemas con aplicaciones de firewall específicas que bloquean el acceso a Internet en WSL. Los firewalls que dan error son:

1. Kaspersky
2. AVG
3. Avast
4. Symantec Endpoint Protection

En algunos casos, si desactivas el firewall podrás acceder. En otros casos, simplemente tener instalado el firewall parece bloquear el acceso a Internet.

Si usa el Firewall de Microsoft Defender, la desactivación de "Bloquear todas las conexiones entrantes, incluidas las de la lista de aplicaciones permitidas" permite el acceso.

Error de permiso denegado al usar ping

En cuanto a la [Actualización de aniversario de Windows \(versión 1607\)](#), es necesario tener **privilegios de administrador** en Windows para ejecutar ping en WSL. Para ejecutar ping, ejecuta Bash en Ubuntu en Windows como administrador o ejecuta `bash.exe` desde un símbolo del sistema de CMD/PowerShell con privilegios de administrador.

Para versiones posteriores de Windows ([compilación 14926+](#)) ya no es necesario tener privilegios de administrador.

Bash se bloquea

Si mientras trabajas con Bash, ves que Bash se bloquea (o interbloquea) y no responde a las entradas, ayúdanos a diagnosticar el problema mediante la recopilación y la generación de informes de un volcado de memoria. Ten en cuenta que con estos pasos se bloqueará tu sistema. No lo hagas si no te sientes cómodo al hacerlo o guarda el trabajo antes de hacerlo.

Para recopilar un volcado de memoria

1. Cambia el tipo de volcado de memoria a "volcado de memoria completa". Al cambiar el tipo de volcado, toma nota del tipo actual.
2. Sigue los [pasos ↗](#) para configurar el bloqueo mediante el control de teclado.
3. Reproduce el bloqueo o interbloqueo.
4. Bloquea el sistema con la secuencia de claves de (2).
5. El sistema se bloqueará y recopilará el volcado de memoria.
6. Una vez que se reinicie el sistema, informa memory.dmp a secure@microsoft.com. La ubicación predeterminada del archivo de volcado es %SystemRoot%\memory.dmp o C:\Windows\memory.dmp si C: es la unidad del sistema. En el correo electrónico, indica que el volcado de memoria es para el equipo de WSL o de Bash en Windows.
7. Restaura el tipo de volcado de memoria a la configuración original.

Comprobar el número de compilación

Para encontrar la arquitectura de tu PC y el número de compilación de Windows, abre: **Configuración>Sistema>Acerca de**

Busca los campos **Compilación del SO** y **Tipo de sistema**.

The screenshot shows two sections of system information. The top section, 'Device specifications', includes fields like Device name (Desktop-PC), Processor, Installed RAM, Device ID, Product ID, System type (64-bit operating system, x64-based processor), and Pen and touch (No pen or touch input is available for this display). The bottom section, 'Windows specifications', includes Edition (Windows 11 Pro), Version (Dev), Installed on (26.02.2022), OS build (22563.1), Experience (Windows Feature Experience Pack 1000.22563.1.0), Microsoft Services Agreement, and Microsoft Software Licence Terms.

Para encontrar el número de compilación de Windows Server, ejecuta lo siguiente en PowerShell:

```
PowerShell  
systeminfo | Select-String "^\$OS Name", "^\$OS Version"
```

Confirmar que WSL está habilitado

Para confirmar que el Subsistema de Windows para Linux está habilitado, puede ejecutar lo siguiente en una ventana PowerShell con privilegios elevados:

```
PowerShell  
Get-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux
```

Problemas de conexión del servidor de OpenSSH

Al intentar conectarte al servidor SSH se produce el error siguiente: "Connection closed by 127.0.0.1 port 22" (Conexión cerrada por 127.0.0.1 puerto 22).

1. Asegúrate de que el servidor OpenSSH esté en ejecución:

```
Bash
```

```
sudo service ssh status
```

y que has seguido este tutorial: <https://ubuntu.com/server/docs/service-openssh> ↗

2. Detén el servicio sshd e inicia sshd en modo de depuración:

```
Bash
```

```
sudo service ssh stop  
sudo /usr/sbin/sshd -d
```

3. Comprueba los registros de inicio y asegúrate de que HostKeys estén disponibles y de no ver mensajes de registro como:

```
BASH
```

```
debug1: sshd version OpenSSH_7.2, OpenSSL 1.0.2g 1 Mar 2016  
debug1: key_load_private: incorrect passphrase supplied to decrypt  
private key  
debug1: key_load_public: No such file or directory  
Could not load host key: /etc/ssh/ssh_host_rsa_key  
debug1: key_load_private: No such file or directory  
debug1: key_load_public: No such file or directory  
Could not load host key: /etc/ssh/ssh_host_dsa_key  
debug1: key_load_private: No such file or directory  
debug1: key_load_public: No such file or directory  
Could not load host key: /etc/ssh/ssh_host_ecdsa_key  
debug1: key_load_private: No such file or directory  
debug1: key_load_public: No such file or directory  
Could not load host key: /etc/ssh/ssh_host_ed25519_key
```

Si ve estos mensajes y faltan las claves en `/etc/ssh/`, tendrá que volver a generar las claves o simplemente purgar e instalar el servidor OpenSSH:

```
BASH
```

```
sudo apt-get purge openssh-server  
sudo apt-get install openssh-server
```

"No se encontró el ensamblado al que se hace referencia." al habilitar la característica opcional WSL.

Este error está relacionado con un estado de instalación incorrecta. Complete los pasos siguientes para intentar corregir este problema:

- Si estás ejecutando el comando para habilitar la característica WSL desde PowerShell, intenta usar la GUI en su lugar desde el menú Inicio: busca "activar o desactivar las características de Windows" y, a continuación, en la lista, selecciona "Subsistema de Windows para Linux", lo que instalará el componente opcional.
 - Actualiza la versión de Windows. Para ello, ve a Configuración, Actualizaciones y haz clic en "Buscar actualizaciones".
 - Si se produce un error en ambos casos y necesita acceder a WSL, considere la posibilidad de realizar una actualización local. Para ello, vuelva a instalar Windows 10 mediante el uso de medios de instalación y seleccione "Conservar todo" para asegurarse de que las aplicaciones y los archivos se van a conservar. Puedes encontrar instrucciones sobre cómo hacerlo en la página [Reinstalar Windows 10](#).

Corrección de errores de permisos (relacionados con SSH)

Si ves este error:

Para corregirlo, anexa lo siguiente al archivo `/etc/wsl.conf`:

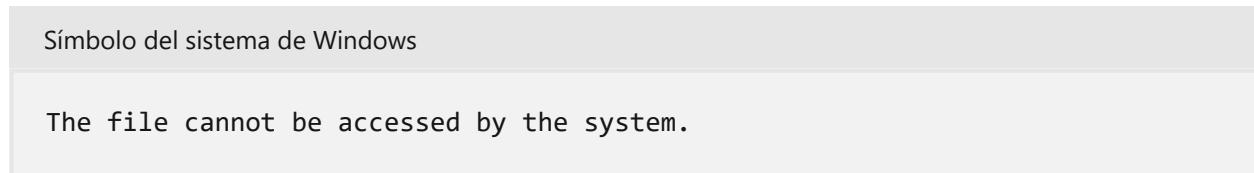
```
Bash

[automount]
enabled = true
options = metadata uid=1000 gid=1000 umask=0022
```

Ten en cuenta que, al agregar este comando, se incluirán metadatos y se modificarán los permisos de archivo en los archivos de Windows que se muestran en WSL. Para obtener más información, consulta los [Permisos del sistema de archivos](#).

No se puede usar WSL de forma remota mediante OpenSSH en Windows

Si está utilizando openssh-server en Windows e intenta acceder a WSL de forma remota, es posible que vea este error:



Se trata de un [problema conocido](#) cuando se usa la versión de Store de WSL. Puede solucionarlo hoy con el uso de WSL1 o la versión de WSL integrada de Windows. Consulte <https://aka.ms/wslstoreinfo> para obtener más información.

La ejecución de comandos de Windows devuelve un error dentro de una distribución

Algunas distribuciones [disponibles en Microsoft Store](#) aún no son totalmente compatibles con la ejecución de comandos de Windows de forma predefinida. Si recibe un error `-bash: powershell.exe: command not found` al ejecutar `powershell.exe /c start .` o cualquier otro comando de Windows, siga estos pasos para resolverlo:

1. En la distribución de WSL, ejecute `echo $PATH`.

Si no incluye `/mnt/c/Windows/system32`, hay algo que redefine la variable PATH estándar.

2. Consulte la configuración del perfil con `cat /etc/profile`.

Si contiene la asignación de la variable PATH, edite el archivo para comentar el bloque de asignación de PATH con un carácter # .

3. Compruebe si `wsl.conf` está presente `cat /etc/wsl.conf` y asegúrese de que no contenga `appendWindowsPath=false`. En caso contrario, coméntelo.

4. Para reiniciar la distribución, escriba `wsl -t` seguido del nombre de la distribución o ejecute `wsl --shutdown` en CMD o PowerShell.

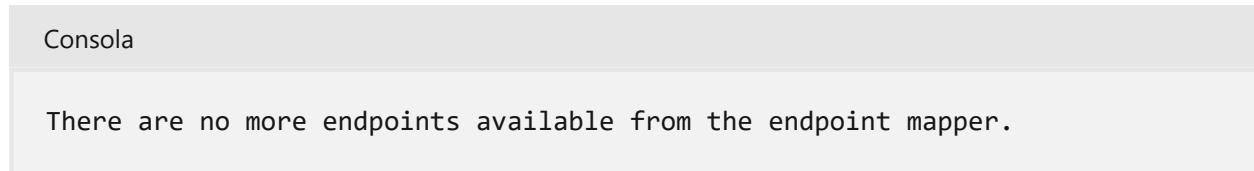
No se puede realizar el arranque después de instalar WSL 2

Somos conscientes de un problema que afecta a los usuarios en el que no pueden realizar el arranque después de instalar WSL 2. Mientras realizamos el diagnóstico completo del problema, los usuarios han comunicado que este puede solucionarse si [se cambia el tamaño del búfer](#) o [se instalan los controladores adecuados](#). Consulte este [problema de GitHub](#) para ver sus actualizaciones más recientes.

Errores de WSL 2 cuando el componente ICS está deshabilitado

La conexión compartida a Internet (ICS) es un componente necesario de WSL 2. El servicio de red de host (HNS) usa el servicio ICS para crear la red virtual subyacente en la que se basa WSL 2 para la conexión compartida de host, NAT, DNS y DHCP.

Deshabilitar el servicio ICS (SharedAccess) o bien ICS mediante la directiva de grupo impedirá que se cree la red HNS de WSL. Esto producirá errores al crear una nueva imagen de la versión 2 de WSL y el siguiente error al intentar convertir una imagen de la versión 1 a la versión 2.



Los sistemas que requieren WSL 2 deben dejar el servicio ICS (SharedAccess) en el estado de inicio predeterminado, Manual (desencadenar inicio), y cualquier directiva que deshabilite ICS debe sobrescribirse o quitarse. Aunque deshabilitar el servicio ICS interrumpirá WSL 2 y no es una acción recomendada, se pueden deshabilitar secciones de ICS siguiendo [estas instrucciones](#).

Uso de versiones anteriores de Windows y WSL

Hay varias diferencias a tener en cuenta si ejecuta una versión anterior de Windows y WSL, como Windows 10 Creators Update (octubre de 2017, compilación 16299) o Actualización de aniversario (agosto de 2016, compilación 14393). Se recomienda [actualizar a la versión Windows más reciente](#). Por si no es posible, se describen algunas de las diferencias a continuación.

Diferencias entre los comandos de interoperabilidad:

- `bash.exe` se ha reemplazado por `wsl.exe`. Los comandos de Linux se pueden ejecutar desde el símbolo del sistema de Windows o desde PowerShell, pero para las primeras versiones de Windows, puede que tengas que usar el comando `bash`. Por ejemplo: `C:\temp> bash -c "ls -la"`. Los comandos de WSL pasados a `bash -c` se reenvían al proceso de WSL sin modificaciones. Las rutas de acceso de archivo deben especificarse en el formato de WSL y se debe tener cuidado al escapar caracteres relevantes. Por ejemplo, `C:\temp> bash -c "ls -la /proc/cpuinfo"` o `C:\temp> bash -c "ls -la \"\mnt/c/Program Files\""`.

- Para ver qué comandos están disponibles para una distribución determinada, ejecuta `[distro.exe] /?`. Por ejemplo, con Ubuntu: `C:\> ubuntu.exe /?`.
- La ruta de acceso de Windows se incluye en `$PATH` de WSL.
- Al llamar a una herramienta de Windows desde una distribución de WSL en una versión anterior de Windows 10, deberás especificar la ruta de acceso del directorio. Por ejemplo, para llamar a la aplicación Bloc de notas de Windows desde la línea de comandos de WSL, escriba:
`/mnt/c/Windows/System32/notepad.exe`.
- Para cambiar el usuario predeterminado a `root`, use este comando en PowerShell
`C:\> 1xrun /setDefaultUser root` y, a continuación, ejecute Bash.exe para iniciar sesión en `C:\> bash.exe`. Restablezca la contraseña con el comando de contraseña de la distribución (`$ passwd username`) y, a continuación, cierre la línea de comandos de Linux: `$ exit`. Desde el símbolo del sistema de Windows o PowerShell, restablezca el usuario predeterminado a su cuenta de usuario de Linux normal: `C:\> 1xrun.exe /setDefaultUser username`.

Desinstalación de la versión heredada de WSL

Si originalmente instaló WSL en una versión de Windows 10 anterior a Creators Update (octubre de 2017, compilación 16299), se recomienda migrar los archivos, los datos, etc. necesarios de la distribución de Linux anterior que instaló a una distribución más reciente instalada a través de Microsoft Store. Para quitar la distribución heredada de la máquina, ejecute lo siguiente desde una línea de comandos o una instancia de PowerShell: `wsl --unregister Legacy`. También tiene la opción de quitar manualmente la distribución heredada anterior mediante la eliminación de la carpeta `%localappdata%\lxss\` (y todo su contenido) mediante el Explorador de archivos de Windows o PowerShell: `rm -Recurse $env:localappdata/lxss/`.

Notas de la versión del subsistema de Windows para Linux

Artículo • 21/03/2023

Compilación 21364

Para obtener información general de Windows sobre la compilación 21364, visite el [blog de Windows](#).

- Las aplicaciones de GUI ya están disponibles. Para obtener más información, consulte [esta entrada de blog](#).
- Resuelva el error al acceder a los archivos a través de \\wsl.localhost\.
- Corrija un posible interbloqueo en el servicio LsassManager.

Compilación 21354

Para obtener información general de Windows sobre la compilación 21354, visite el [blog de Windows](#).

- Cambie el prefijo \wsl a \wsl.localhost para evitar problemas cuando haya una máquina en la red denominada "wsl". \wsl\$ seguirá funcionando.
- Habilite el ícono de acceso rápido de Linux para procesos de wow.
- Actualice el problema en el que la versión 2 siempre se pasaba a través de wslapi RegisterDistribution.
- Cambie la máscara fmask del directorio /usr/lib/wsl/lib a 222 para que los archivos se marquen como ejecutables [GH 3847]
- Corrija el bloqueo del servicio wsl si la plataforma de máquina virtual no está habilitada.

Compilación 21286

Para obtener información general de Windows sobre la compilación 21286, visite el [blog de Windows](#).

- Se incorporó el comando wsl.exe --cd para definir el directorio de trabajo actual de un comando.
- Se mejoró la asignación de NTSTATUS a los códigos de error de Linux. [GH 6063]
- Se mejoró la creación de informes de error de wsl.exe --mount.
- Se agregó una opción a /etc/wsl.conf para habilitar los comandos de inicio:

Consola

```
[boot]
command=<string>
```

Compilación 20226

Para obtener información general de Windows sobre la compilación 20226, visite el [blog de Windows](#).

- Se corrigió el bloqueo en el servicio LxssManager. [GH 5902]

Compilación 20211

Para obtener información general de Windows sobre la compilación 20211, visite el [blog de Windows](#).

- Introduzca `wsl.exe --mount` para montar discos físicos o virtuales. Para obtener más información, consulte el tema sobre el [acceso a los sistemas de archivos de Linux en Windows y WSL 2](#).
- Se corrigió el bloqueo en el servicio LxssManager al comprobar si la VM estaba inactiva. [GH 5768]
- Se agregó compatibilidad con los archivos VHD comprimidos. [GH 4103]
- Se garantizó que las bibliotecas del modo de usuario de Linux instaladas en `c:\windows\system32\lxss\lib` se conservan en la actualización del sistema operativo. [GH 5848]
- Se agregó la capacidad de enumerar las distribuciones disponibles que se pueden instalar con `wsl --install --list-distributions`.
- Las instancias de WSL ahora se terminan cuando el usuario cierra la sesión.

Compilación 20190

Para obtener información general de Windows sobre la compilación 20190, visite el [blog de Windows](#).

- Se corrigieron los errores que impedían que se iniciaran las instancias de WSL1. [GH 5633]
- Se corrigió un bloqueo que se producía al redirigir la salida del proceso de Windows. [GH 5648]

- Se agregó la opción %userprofile%\wslconfig para controlar el tiempo de espera de inactividad de la VM (wsl2.vmIdleTimeout=<time_in_ms>).
- Se agregó compatibilidad para iniciar alias de ejecución de la aplicación desde WSL.
- Se agregó compatibilidad para la instalación del kernel y las distribuciones de WSL2 en wsl.exe --install.

Compilación 20175

Para obtener información general de Windows sobre la compilación 20175, visite el [blog de Windows](#).

- Se ajustó la asignación de memoria predeterminada de la máquina virtual WSL2 para que sea un 50 % de la memoria del host o 8 GB, el tamaño que sea inferior [GH 4166].
- Se cambió el prefijo \\wsl\$ por \\wsl para admitir el análisis de URI. Todavía se admite la ruta de acceso \\wsl\$ anterior.
- Se habilitó la virtualización anidada para WSL2 de forma predeterminada en amd64. Puede deshabilitarla en %userprofile%\wslconfig ([wsl2] nestedVirtualization=false).
- Se modificó el comando wsl.exe --update para que exija iniciar Microsoft Update.
- Se agregó compatibilidad para cambiar el nombre de un archivo de solo lectura en DrvFs.
- Se garantizó que los mensajes de error siempre se muestren en la página de códigos correcta.

Compilación 20150

Para obtener información general de Windows sobre la compilación 20150, visite el [blog de Windows](#).

- Para obtener más información sobre el proceso de GPU WSL2, consulte el [blog de Windows](#).
- Introduzca la opción de la línea de comandos wsl.exe --install para configurar fácilmente WSL.
- Introduzca la opción de la línea de comandos wsl.exe --update para administrar las actualizaciones en el kernel de WSL2.
- Establezca WSL2 como valor predeterminado.
- Aumente el tiempo de espera de cierre correcto de la VM WSL2.
- Corrija la condición de carrera virtio-9p al asignar la memoria del dispositivo.
- No ejecute un servidor 9p con privilegios elevados si UAC está deshabilitado.

Compilación 19640

Para obtener información general de Windows sobre la compilación 19640, visite el [blog de Windows](#).

- [WSL2] Mejoras de estabilidad para Virtio-9p (drvfs).

Compilación 19555

Para obtener información general de Windows sobre la compilación 19555, visita el [blog de Windows](#).

- [WSL2] Uso de un grupo de memoria para limitar la cantidad de memoria que usan las operaciones de instalación y conversión [GH 4669]
- Inclusión de wsl.exe cuando el componente opcional del Subsistema de Windows para Linux no esté habilitado para mejorar la detectabilidad de características.
- Cambio de wsl.exe para imprimir el texto de ayuda si el componente opcional WSL no está instalado
- Corrección de la condición de carrera al crear instancias
- Creación de un archivo wslclient.dll que contiene toda la funcionalidad de la línea de comandos
- Prevención de bloqueos durante la detención del servicio LxssManagerUser
- Corrección del error rápido de wslapi.dll cuando el parámetro distroName es NULL

Compilación 19041

Para obtener información general de Windows sobre la compilación 19041, visita el [blog de Windows](#).

- [WSL2] Eliminación de la máscara de señal antes de iniciar los procesos
- [WSL2] Actualización del kernel de Linux a la versión 4.19.84
- Control de la creación de symlink /etc/resolv.conf cuando symlink no es relativo

Compilación 19028

Para obtener información general de Windows sobre la compilación 19028, visita el [blog de Windows](#).

- [WSL2] Actualización del kernel de Linux a la versión 4.19.81.
- [WSL2] Cambiar el permiso predeterminado de/dev/net/tun a 0666 [GH 4629].

- [WSL2] Ajustar la cantidad predeterminada de memoria asignada a la VM de Linux para que el 80 % sea parte de la memoria del host.
 - [WSL2] Corrección del servidor de interoperabilidad para controlar las solicitudes con tiempo de espera, para que los autores de llamadas incorrectas no puedan bloquear el servidor.

Compilación 19018

Para obtener información general de Windows sobre la compilación 19018, visita el [blog de Windows](#).

- [WSL2] Uso de la caché = mmap como valor predeterminado para los montajes de 9p a fin de corregir las aplicaciones dotnet
 - [WSL2] Correcciones para la retransmisión de localhost [GH 4340]
 - [WSL2] Introducción de un montaje tmpfs compartido de distribución transversal para compartir el estado entre distribuciones
 - Corrección de la restauración de la unidad de red persistente para \\wsl\$

Compilación 19013

Para obtener información general de Windows sobre la compilación 19013, visita el [blog de Windows](#).

- [WSL2] Mejora del rendimiento de memoria de la VM de la utilidad WSL. La memoria que ya no esté en uso se liberará de nuevo al host.
 - [WSL2] Actualización de la versión del kernel a 4.19.79. (Adición de CONFIG_HIGH_RES_TIMERS, CONFIG_TASK_XACCT, CONFIG_TASK_IO_ACCOUNTING, CONFIG_SCHED_HRTICK y CONFIG_BRIDGE_VLAN_FILTERING).
 - [WSL2] Corrección de la retransmisión de entrada para controlar los casos en los que stdin es un identificador de canalización que no está cerrado [GH 4424].
 - Asegúrese de comprobar que \\wsl\$ no distingue mayúsculas de minúsculas.

Consola

```
[ws12]
pageReporting = <bool>      # Enable or disable the free memory page reporting
feature (default true).
idleThreshold = <integer> # Set the idle threshold for memory compaction, 0
disables the feature (default 1).
```

Compilación 19002

Para obtener información general de Windows sobre la compilación 19002, visita el [blog de Windows](#).

- [WSL] Corrección de un problema con el control de algunos caracteres Unicode: <https://github.com/microsoft/terminal/issues/2770>
- [WSL] Corrección de casos poco frecuentes en los que se podría anular el registro de distribuciones si se iniciaban inmediatamente después de una actualización de una compilación a otra.
- [WSL] Corrección de un problema leve por el que wsl.exe se apagaba cuando no se cancelaban los temporizadores de inactividad de la instancia.

Compilación 18995

Para obtener información general de Windows sobre la compilación 18995, visita el [blog de Windows](#).

- [WSL2] Corrección de un problema que provocaba que los montajes DrvFs dejaran de funcionar después de que se interrumpiera una operación (por ejemplo, Ctrl-C) [GH 4377]
- [WSL2] Corrección del control de mensajes hvsocket muy grandes [GH 4105]
- [WSL2] Corrección de un problema con la interoperabilidad cuando stdin es un archivo [GH 4475]
- [WSL2] Corrección de un bloqueo de servicio cuando se encontraba un estado de red inesperado [GH 4474]
- [WSL2] Consulta del nombre de distribución desde el servidor de interoperabilidad si el proceso actual no tiene la variable de entorno
- [WSL2] Corrección de un problema con la interoperabilidad cuando stdin es un archivo
- [WSL2] Actualización de la versión del kernel de Linux a 4.19.72
- [WSL2] Agrega la capacidad de especificar parámetros adicionales de línea de comandos de kernel mediante .wslconfig

```
[ws12]
kernelCommandLine = <string> # Additional kernel command line arguments
```

Compilación 18990

Para obtener información general de Windows sobre la compilación 18990, visita el [blog de Windows](#).

- Mejora del rendimiento de los listados de directorios en \\wsl\$
- [WSL2] Insertar entropía de arranque adicional [GH 4416]
- [WSL2] Corrección para la interoperabilidad de Windows al usar su/sudo [GH 4465]

Compilación 18980

Para obtener información general de Windows sobre la compilación 18980, visita el [blog de Windows](#).

- Corrige la lectura de los vínculos simbólicos que deniegan FILE_READ_DATA. Esto incluye todos los vínculos simbólicos que crea Windows para la compatibilidad con versiones anteriores, como "C:\Document and Settings" y un conjunto de vínculos simbólicos en el directorio de perfil del usuario.
- Hace que el estado inesperado del sistema de archivos no sea irrecuperable [GH 4334, 4305]
- [WSL2] Agrega compatibilidad para arm64 si tu CPU/firmware admite virtualización
- [WSL2] Permite a los usuarios sin privilegios ver el registro del kernel
- [WSL2] Corrige la retransmisión de salida cuando se han cerrado los sockets stdout/stderr [GH 4375]
- [WSL2] Compatibilidad con la batería y el adaptador de AC
- [WSL2] Actualización del kernel de Linux a 4.19.67
- Adición de la capacidad de establecer el nombre de usuario predeterminado en /etc/WSL.conf:

```
[user]
default=<string>
```

Compilación 18975

Para obtener información general de Windows sobre la compilación 18975, visita el [blog de Windows](#).

- [WSL2] Se han corregido varios problemas de confiabilidad de localhost [GH 4340]

Compilación 18970

Para obtener información general de Windows sobre la compilación 18970, visita el [blog de Windows](#).

- [WSL2] Sincronización de la hora con la hora del host cuando el sistema se reanuda desde el estado de suspensión [GH 4245]
- [WSL2] Crea vínculos simbólicos de NT en volúmenes de Windows cuando sea posible.
- [WSL2] Crea distribuciones en los espacios de nombres UTS, IPC, PID y Mount.
- [WSL2] Corrección de retransmisión de puerto localhost cuando el servidor se enlaza a localhost directamente [GH 4353]
- [WSL2] Corrección de interoperabilidad cuando se redirige la salida [GH 4337]
- [WSL2] Compatibilidad con la traducción de vínculos simbólicos de NT absolutos.
- [WSL2] Actualización del kernel a 4.19.59
- [WSL2] Definición correcta de la máscara de subred para eth0.
- [WSL2] Cambio de la lógica para interrumpir el bucle de trabajo de la consola cuando se señale el evento de salida.
- [WSL2] Expulsión del VHD de distribución cuando la distribución no está en ejecución.
- [WSL2] Corrección de la biblioteca de análisis de configuración para controlar correctamente los valores vacíos.
- [WSL2] Compatibilidad con el escritorio de Docker al crear montajes de distribución cruzados. Una distribución puede participar en este comportamiento si se agrega la siguiente línea al archivo /etc/wsl.conf:

```
[automount]
crossDistro = true
```

Compilación 18945

Para obtener información general de Windows sobre la compilación 18945, visita el [blog de Windows](#).

WSL

- [WSL2] Permite que se acceda a los sockets TCP de escucha en WSL2 desde el host mediante localhost:port
- [WSL2] Correcciones de errores de instalación/conversión y diagnósticos adicionales para un seguimiento de los problemas futuros [GH 4105]
- [WSL2] Mejora la capacidad de diagnóstico de los problemas de red de WSL2

- [WSL2] Actualización de la versión del kernel a 4.19.55
- [WSL2] Actualización del kernel con las opciones de configuración necesarias para Docker [GH 4165]
- [WSL2] Aumento del número de CPU asignadas a la VM de utilidad ligera para que sea el mismo que el host (se ha limitado previamente a 8 por CONFIG_NR_CPUS en la configuración del kernel) [GH 4137]
- [WSL2] Creación de un archivo de intercambio para la VM ligera WSL2
- [WSL2] Permite que los montajes de usuarios sean visibles a través de la distribución \\wsl\$\\distro (por ejemplo, sshfs) [GH 4172]
- [WSL2] Mejora el rendimiento del sistema de archivos 9p
- [WSL2] Asegura que la ACL de VHD no crezca sin límite [GH 4126]
- [WSL2] Actualización de la configuración de kernel para admitir squashfs y xt_conntrack [GH 4107, 4123]
- [WSL2] Corrección para la opción interop.enabled /etc/wsl.conf [GH 4140]
- [WSL2] Devuelve ENOTSUP si el sistema de archivos no es compatible con EA
- [WSL2] Corrección de CopyFile colgado con \\wsl\$
- Cambio del valor predeterminado de umask a 0022 y adición de la configuración de filesystem.umask a /etc/wsl.conf
- Corrección de wslpath para resolver correctamente los vínculos simbólicos, esto se retomó en 19h1 [GH 4078]
- Introducción del archivo %UserProfile%\.wslconfig para retocar la configuración de WSL2

```
[ws12]
kernel=<path>          # An absolute Windows path to a custom Linux
kernel.
memory=<size>           # How much memory to assign to the WSL2 VM.
processors=<number>       # How many processors to assign to the WSL2 VM.
swap=<size>              # How much swap space to add to the WSL2 VM. 0
for no swap file.
swapFile=<path>          # An absolute Windows path to the swap vhd.
localhostForwarding=<bool> # Boolean specifying if ports bound to wildcard
or localhost in the WSL2 VM should be connectable from the host via
localhost:port (default true).

# <path> entries must be absolute Windows paths with escaped backslashes,
for example C:\\Users\\\\Ben\\\\kernel
# <size> entries must be size followed by unit, for example 8GB or 512MB
```

Para obtener información general de Windows sobre la compilación 18917, visita el [blog de Windows](#).

WSL

- ¡WSL 2 ya está disponible! Consulta el [blog](#) para más detalles.
- Corrección de una regresión en la que iniciar procesos de Windows a través de vínculos simbólicos no funcionaba correctamente [GH 3999]
- Agrega las opciones wsl.exe --list --verbose, wsl.exe --list --quiet y wsl.exe --import --version a wsl.exe
- Agrega la opción wsl.exe --shutdown
- Plan 9: Permite abrir un directorio para escribir en él correctamente

Compilación 18890

Para obtener información general de Windows sobre la compilación 18890, visita el [blog de Windows](#).

WSL

- Pérdida de socket sin bloqueo [GH 2913]
- La entrada de EOF en el terminal puede bloquear lecturas posteriores [GH 3421]
- Actualización del encabezado de resolv.conf para hacer referencia a wsl.conf [descrito en GH 3928]
- Interbloqueo en código de eliminación de epoll [GH 3922]
- Control de espacios de los argumentos en -import y -export [GH 3932]
- La extensión de los archivos mmap no funciona correctamente [GH 3939]
- Corrección del problema de funcionamiento del acceso a ARM64 \\wsl\$
- Agrega el mejor ícono predeterminado para wsl.exe

Compilación 18342

Para obtener información general de Windows sobre la compilación 18342, visita el [blog de Windows](#).

WSL

- Hemos agregado la posibilidad de que los usuarios accedan a los archivos de Linux en una distribución de WSL desde Windows. Se puede acceder a estos archivos a través de la línea de comandos; también las aplicaciones de Windows, como el

explorador de archivos, VSCode, etc., pueden interactuar con estos archivos. Para acceder a los archivos, vaya a \\wsl\$\\<distro_name> o consulte la lista de las distribuciones en ejecución en \\wsl\$.

- Agrega etiquetas de información de CPU adicionales y corregir valores de Cpus_allowed[_list] [GH 2234]
- Compatibilidad con exec desde un subprocesso no líder [GH 3800]
- Trata errores de actualización de la configuración como no irrecuperables [GH 3785]
- Actualización de binfmt para administrar correctamente los desplazamientos [GH 3768]
- Habilita la asignación de unidades de red para Plan 9 [GH 3854]
- Compatibilidad con Windows -> Linux y Linux -> Traducción de rutas de acceso de Windows para montajes de enlace
- Crea secciones de solo lectura para asignaciones en archivos abiertos de solo lectura

Compilación 18334

Para obtener información general de Windows sobre la compilación 18334, visita el [blog de Windows](#).

WSL

- Rediseña el modo en que se asigna la zona horaria de Windows a una zona horaria de Linux [GH 3747]
- Corrección de pérdidas de memoria y adición de nuevas funciones de traducción de cadenas [GH 3746]
- SIGCONT en un grupo de subprocessos sin subprocessos es no-op [alvent 3741]
- Muestra correctamente los descriptores de archivo de socket y epoll en /proc/self/fd

Compilación 18305

Para obtener información general de Windows sobre la compilación 18305, visita el [blog de Windows](#).

WSL

- pthreads pierden acceso a los archivos cuando sale el subprocesso principal [GH 3589]

- TIOCSCTTY debe omitir el parámetro "force", a menos que sea necesario [GH 3652]
- Mejoras en la línea de comandos de wsl.exe y adición de la funcionalidad de importación y exportación.

```
Usage: wsl.exe [Argument] [Options...] [CommandLine]
```

Arguments to run Linux binaries:

If no command line is provided, wsl.exe launches the default shell.

```
--exec, -e <CommandLine>
    Execute the specified command without using the default Linux shell.

-- 
    Pass the remaining command line as is.
```

Options:

```
--distribution, -d <DistributionName>
    Run the specified distribution.
```

```
--user, -u <UserName>
    Run as the specified user.
```

Arguments to manage Windows Subsystem for Linux:

```
--export <DistributionName> <FileName>
    Exports the distribution to a tar file.
    The filename can be - for standard output.
```

```
--import <DistributionName> <InstallLocation> <FileName>
    Imports the specified tar file as a new distribution.
    The filename can be - for standard input.
```

```
--list, -l [Options]
    Lists distributions.
```

Options:

```
--all
    List all distributions, including distributions that are
currently
    being installed or uninstalled.
```

```
--running
    List only distributions that are currently running.
```

```
-setdefault, -s <DistributionName>
    Sets the distribution as the default.
```

```
--terminate, -t <DistributionName>
    Terminates the distribution.
```

```
--unregister <DistributionName>
    Unregisters the distribution.

--upgrade <DistributionName>
    Upgrades the distribution to the WslFs file system format.

--help
    Display usage information.
```

Compilación 18277

Para obtener información general de Windows sobre la compilación 18277, visita el [blog de Windows](#).

WSL

- Corrección del error "Interfaz no compatible" que se presentó en la compilación 18272 [GH 3645]
- Omite la marca MNT_FORCE para la llamada del sistema umount [GH 3605]
- Cambia la interoperabilidad de WSL para usar la API de CreatePseudoConsole oficial
- No mantiene ningún valor de tiempo de espera cuando se reinicia FUTEX_WAIT

Compilación 18272

Para obtener información general de Windows sobre la compilación 18272, visita el [blog de Windows](#).

WSL

- **ADVERTENCIA:** Hay un problema en esta compilación que hace que WSL sea inoperable. Al intentar iniciar la distribución, verás el error "Interfaz no compatible". El problema se ha solucionado y estará en la compilación del modo anticipado de Insider de la próxima semana. Si instaló esta compilación, puede revertir a la compilación anterior de Windows desde "Volver a la versión anterior de Windows 10" en Configuración->Actualización & seguridad->Recuperación.

Compilación 18267

Para obtener información general de Windows sobre la compilación 18267, visita el [blog de Windows](#).

WSL

- Corrección del problema por el que un proceso inerte no podía recogerse y permanecía de manera indefinida.
- WslRegisterDistribution se bloquea si el mensaje de error supera la longitud máxima [GH 3592]
- Permite que fsync se complete correctamente para los archivos de solo lectura en DrvFs [GH 3556]
- Asegúrate de que los directorios /bin y /sbin existan antes de crear los vínculos simbólicos en ellos [GH 3584]
- Se agregó un mecanismo de tiempo de espera de finalización de instancias para las instancias de WSL. El tiempo de espera está establecido actualmente en 15 segundos, por lo que la instancia finalizará 15 segundos después de que termine el último proceso de WSL. Para finalizar una distribución de inmediato, usa:

```
wslconfig.exe /terminate <DistributionName>
```

Compilación 17763 (1809)

Para obtener información general de Windows sobre la compilación 17763, visita el [blog de Windows](#).

WSL

- Comprobación de permisos de la llamada del sistema setpriority demasiado estricta para cambiar la misma prioridad de subprocessos [GH 1838]
- Asegúrate de que se use el tiempo de interrupción no sesgado para el tiempo de arranque para evitar que se devuelvan valores negativos para clock_gettime (CLOCK_BOOTTIME) [GH 3434]
- Controla los vínculos simbólicos en el intérprete binfmt de WSL [GH 3424]
- Mejor control de la limpieza de descriptores de archivo del líder del grupo de subprocessos.
- Cambia WSL para usar KeQueryInterruptTimePrecise en lugar de KeQueryPerformanceCounter para evitar el desbordamiento [GH 3252]

- Ptrace attach puede producir un valor de devolución incorrecto de las llamadas del sistema [GH 1731]
- Corrección de varios problemas relacionados con AF_UNIX [GH 3371]
- Corrección del problema que podía provocar un error de interoperabilidad de WSL si el directorio de trabajo actual tiene menos de 5 caracteres de longitud [GH 3379]
- Evita las conexiones de bucle invertido con error con retraso de un segundo a puertos no existentes [GH 3286]
- Agrega el archivo de código auxiliar /proc/sys/fs/file-max [GH 2893]
- Información más precisa sobre el ámbito IPV6.
- Compatibilidad con PR_SET_PTRACER [GH 3053]
- El sistema de archivos de canalización borra accidentalmente el evento epoll desencadenado por el borde [GH 3276]
- El ejecutable de Win32 iniciado mediante el vínculo simbólico de NTFS no respeta el nombre del vínculo simbólico [GH 2909]
- Compatibilidad mejorada con zombie [GH 1353]
- Agrega entradas de wsl.conf para controlar el comportamiento de interoperabilidad de Windows [GH 1493]

```
[interop]

enabled=false # enable launch of Windows binaries; default is true

appendWindowsPath=false # append Windows path to $PATH variable;
default is true
```

- La corrección para getsockname no siempre devuelve el tipo de familia de socket de UNIX [GH 1774]
- Agrega compatibilidad para TIOCSTI [GH 1863]
- Los sockets sin bloqueo en el proceso de conexión deben devolver EAGAIN para los intentos de escritura [GH 2846]
- Compatibilidad con la interoperabilidad en VHD montados [GH 3246, 3291]
- Corrección del problema de comprobación de permisos en la carpeta raíz [GH 3304]
- Compatibilidad limitada para los ioctl de teclado TTY KDGKBTYPE, KDGKBMODE y KDSKBMODE.
- Las aplicaciones de interfaz de usuario de Windows deben ejecutarse incluso cuando se inician en segundo plano.
- Agrega la opción wsl -u o --user [GH 1203]
- Corrección de problemas de inicio de WSL cuando el inicio rápido está habilitado [GH 2576]

- Los sockets de UNIX deben conservar las credenciales del mismo nivel desconectadas [GH 3183]
- Sockets de UNIX sin bloqueo con error de manera indefinida con EAGAIN [GH 3191]
- case=off es el nuevo tipo de montaje drvfs predeterminado [GH 2937, 3212, 3328]
 - Consulta el [blog ↗](#) para más información.
- Agrega wslconfig/terminate para detener las distribuciones en ejecución.
- Corrección del problema con las entradas del menú contextual del shell de WSL que no controlan correctamente las rutas de acceso con espacios.
- Expone la distinción de mayúsculas y minúsculas por directorio como atributo extendido
- ARM64: Emula las operaciones de mantenimiento de la memoria caché. Resuelve un [problema de dotnet ↗](#).
- DrvFs: solo caracteres con escape eliminado en el intervalo privado que corresponden a un carácter de escape.
- Corrección error de desviación por uno en la validación de longitud del intérprete del analizador ELF [GH 3154]
- Temporizadores absolutos de WSL con una hora en el pasado no se activan [GH 3091]
- Asegúrate de que los puntos de reanálisis recién creados se muestren como tales en el directorio principal.
- Crea de forma atómica directorios con distinción entre mayúsculas y minúsculas en DrvFs.
- Se ha corregido un problema adicional en el que las operaciones de varios subprocessos podían devolver ENOENT, aunque el archivo existiese. [GH 2712]
- Se corrigió un error de inicio de WSL cuando UMCI está habilitado. [GH 3020]
- Agrega el menú contextual del explorador para iniciar WSL [GH 437, 603, 1836]. Para usar, mantén presionada la tecla Mayús y haz clic con el botón derecho en una ventana del explorador.
- Corrección del comportamiento de no bloqueo de sockets de UNIX [GH 2822, 3100]
- Corrección del comando NETLINK que se bloquea, tal como se muestra en GH 2026.
- Agrega compatibilidad para las marcas de propagación de montaje [GH 2911].
- Corrección del problema de truncamiento que no provoca eventos inotify [GH 2978].
- Agrega la opción --exec a wsl.exe para invocar un solo binario sin un shell.
- Agrega la opción --distribution a wsl.exe para seleccionar una distribución específica.

- Compatibilidad limitada para dmesg. Ahora, las aplicaciones pueden registrarse en dmesg. El controlador WSL registra información limitada en dmesg. En el futuro, esto puede extenderse para llevar a cabo otros diagnósticos e información del controlador.
 - Nota: dmesg se admite actualmente a través de la interfaz de dispositivo `/dev/kmsg`. Aún no se admite la interfaz syscall `syslog`. Y, por lo tanto, algunas de las opciones de la línea de comandos de `dmesg`, como `-s` y `-c`, no funcionan.
- Cambia el gid y el modo de dispositivos serie predeterminados para que coincidan con los nativos [GH 3042]
- DrvFs ahora admite atributos extendidos.
 - Nota: DrvFs tiene algunas limitaciones en cuanto al nombre de los atributos extendidos. No se permiten algunos caracteres (como "/", ":" y "*"), y los nombres de atributos extendidos no distinguen mayúsculas de minúsculas en DrvFs.

Compilación 18252 (saltar)

Para obtener información general de Windows sobre la compilación 18252, visita el [blog de Windows](#).

WSL

- Traslado de los binarios init y bsdtar fuera del archivo dll lcssmanager y a una carpeta de herramientas independiente
- Corrección de la carrera en torno al cierre del descriptor de archivo cuando se usa CLONE_FILES
- Controla campos opcionales en /proc/pid/mountinfo al traducir rutas de DrvFs
- Permite que DrvFs mknod se complete correctamente sin compatibilidad con metadatos para S_IFREG
- Los archivos de solo lectura creados en DrvFs deben tener el atributo readonly establecido [GH 3411]
- Agrega el asistente /sbin/mount.drvfs para controlar el montaje de DrvFs.
- Usa el cambio de nombre de POSIX en DrvFs.
- Permite la traducción de rutas de acceso en volúmenes sin un GUID de volumen.

Compilación 17738 (rápida)

Para obtener información general de Windows sobre la compilación 17738, visita el [blog de Windows](#).

WSL

- Comprobación de permisos de la llamada del sistema setpriority demasiado estricta para cambiar la misma prioridad de subprocessos [GH 1838]
- Asegúrate de que se use el tiempo de interrupción no sesgado para el tiempo de arranque para evitar que se devuelvan valores negativos para clock_gettime (CLOCK_BOOTTIME) [GH 3434]
- Controla los vínculos simbólicos en el intérprete binfmt de WSL [GH 3424]
- Mejor control de la limpieza de descriptores de archivo del líder del grupo de subprocessos.

Compilación 17728 (rápida)

Para obtener información general de Windows sobre la compilación 17728, visita el [blog de Windows](#).

WSL

- Cambia WSL para usar KeQueryInterruptTimePrecise en lugar de KeQueryPerformanceCounter para evitar el desbordamiento [GH 3252]
- Ptrace attach puede producir un valor de devolución incorrecto de las llamadas del sistema [GH 1731]
- Corrección de varios problemas relacionados con AF_UNIX [GH 3371]
- Corrección del problema que podía provocar un error de interoperabilidad de WSL si el directorio de trabajo actual tiene menos de 5 caracteres de longitud [GH 3379]

Compilación 18204 (saltar)

Para obtener información general de Windows sobre la compilación 18204, visita el [blog de Windows](#).

WSL

- El sistema de archivos de canalización borra accidentalmente el evento epoll desencadenado por el borde [GH 3276]
- El ejecutable de Win32 iniciado mediante el vínculo simbólico de NTFS no respeta el nombre del vínculo simbólico [GH 2909]

Compilación 17723 (rápida)

Para obtener información general de Windows sobre la compilación 17723, visita el [blog de Windows](#).

WSL

- Evita las conexiones de bucle invertido con error con retraso de un segundo a puertos no existentes [GH 3286]
- Agrega el archivo de código auxiliar /proc/sys/fs/file-max [GH 2893]
- Información más precisa sobre el ámbito IPV6.
- Compatibilidad con PR_SET_PTRACER [GH 3053]
- El sistema de archivos de canalización borra accidentalmente el evento epoll desencadenado por el borde [GH 3276]
- El ejecutable de Win32 iniciado mediante el vínculo simbólico de NTFS no respeta el nombre del vínculo simbólico [GH 2909]

Compilación 17713

Para obtener información general de Windows sobre la compilación 17713, visita el [blog de Windows](#).

WSL

- Compatibilidad mejorada con zombie [GH 1353]
- Agrega entradas de wsl.conf para controlar el comportamiento de interoperabilidad de Windows [GH 1493]

```
[interop]

enabled=false # enable launch of Windows binaries; default is true

appendWindowsPath=false # append Windows path to $PATH variable;
default is true
```

- La corrección para getsockname no siempre devuelve el tipo de familia de socket de UNIX [GH 1774]
- Agrega compatibilidad para TIOCSTI [GH 1863]
- Los sockets sin bloqueo en el proceso de conexión deben devolver EAGAIN para los intentos de escritura [GH 2846]
- Compatibilidad con la interoperabilidad en VHD montados [GH 3246, 3291]

- Corrección del problema de comprobación de permisos en la carpeta raíz [GH 3304]
- Compatibilidad limitada para los ioctl de teclado TTY KDGKBTYPE, KDGKBMODE y KDSKBMODE.
- Las aplicaciones de interfaz de usuario de Windows deben ejecutarse incluso cuando se inician en segundo plano.

Compilación 17704

Para obtener información general de Windows sobre la compilación 17704, visita el [blog de Windows](#).

WSL

- Agrega la opción wsl -u o --user [GH 1203]
- Corrección de problemas de inicio de WSL cuando el inicio rápido está habilitado [GH 2576]
- Los sockets de UNIX deben conservar las credenciales del mismo nivel desconectadas [GH 3183]
- Sockets de UNIX sin bloqueo con error de manera indefinida con EAGAIN [GH 3191]
- case=off es el nuevo tipo de montaje drvfs predeterminado [GH 2937, 3212, 3328]
 - Consulta el [blog](#) para más información.
- Agrega wslconfig/terminate para detener las distribuciones en ejecución.

Compilación 17692

Para obtener información general de Windows sobre la compilación 17692, visita el [blog de Windows](#).

WSL

- Corrección del problema con las entradas del menú contextual del shell de WSL que no controlan correctamente las rutas de acceso con espacios.
- Expone la distinción de mayúsculas y minúsculas por directorio como atributo extendido
- ARM64: Emula las operaciones de mantenimiento de la memoria caché. Resuelve un [problema de dotnet](#).
- DrvFs: solo caracteres con escape eliminado en el intervalo privado que corresponden a un carácter de escape.

Compilación 17686

Para obtener información general de Windows sobre la compilación 17686, visita el [blog de Windows](#).

WSL

- Corrección error de desviación por uno en la validación de longitud del intérprete del analizador ELF [GH 3154]
- Temporizadores absolutos de WSL con una hora en el pasado no se activan [GH 3091]
- Asegúrate de que los puntos de reanálisis recién creados se muestren como tales en el directorio principal.
- Crea de forma atómica directorios con distinción entre mayúsculas y minúsculas en DrvFs.

Compilación 17677

Para obtener información general de Windows sobre la compilación 17677, visita el [blog de Windows](#).

WSL

- Se ha corregido un problema adicional en el que las operaciones de varios subprocessos podían devolver ENOENT, aunque el archivo existiese. [GH 2712]
- Se corrigió un error de inicio de WSL cuando UMCI está habilitado. [GH 3020]

Compilación 17666

Para obtener información general de Windows sobre la compilación 17666, visita el [blog de Windows](#).

WSL

ADVERTENCIA: Hay un problema que impide que WSL se ejecute en algunos conjuntos de chips AMD [GH 3134]. Hay una corrección lista y en camino en la rama de compilación de Insider.

- Agrega el menú contextual del explorador para iniciar WSL [GH 437, 603, 1836]. Para usar, mantén presionada la tecla Mayús y haz clic con el botón derecho en una ventana del explorador.
- Corrección del comportamiento de no bloqueo de sockets de Unix [GH 2822, 3100]
- Corrección del comando NETLINK que se bloquea, tal como se muestra en GH 2026.
- Agrega compatibilidad para las marcas de propagación de montaje [GH 2911].
- Corrección del problema de truncamiento que no provoca eventos inotify [GH 2978].
- Agrega la opción --exec a wsl.exe para invocar un solo binario sin un shell.
- Agrega la opción --distribution a wsl.exe para seleccionar una distribución específica.

Compilación 17655 (saltar)

Para obtener información general de Windows sobre la compilación 17655, visita el [blog de Windows](#).

WSL

- Compatibilidad limitada para dmesg. Ahora, las aplicaciones pueden registrarse en dmesg. El controlador WSL registra información limitada en dmesg. En el futuro, esto puede extenderse para llevar a cabo otros diagnósticos e información del controlador.
 - Nota: dmesg se admite actualmente a través de la interfaz de dispositivo `/dev/kmsg`. Aún no se admite la interfaz syscall `syslog`. Y, por lo tanto, algunas de las opciones de la línea de comandos de `dmesg`, como `-S` y `-C`, no funcionan.
- Se ha corregido un problema en el que las operaciones de varios subprocessos podían devolver ENOENT, aunque el archivo existiese. [GH 2712]

Compilación 17639 (saltar)

Para obtener información general de Windows sobre la compilación 17639, visita el [blog de Windows](#).

WSL

- Cambia el gid y el modo de dispositivos serie predeterminados para que coincidan con los nativos [GH 3042]
- DrvFs ahora admite atributos extendidos.
 - Nota: DrvFs tiene algunas limitaciones en cuanto al nombre de los atributos extendidos. En particular, no se permiten algunos caracteres (como "/", ":" y "*"), y los nombres de atributos extendidos no distinguen mayúsculas de minúsculas en DrvFs.

Compilación 17133 (rápida)

Para obtener información general de Windows sobre la compilación 17133, visita el [blog de Windows](#).

WSL

- Corrección de un bloqueo en WSL. [GH 3039, 3034]

Compilación 17128 (rápida)

Para obtener información general de Windows sobre la compilación 17128, visita el [blog de Windows](#).

WSL

- Ninguno

Compilación 17627 (saltar)

Para obtener información general de Windows sobre la compilación 17627, visita el [blog de Windows](#).

WSL

- Agrega compatibilidad para las operaciones compatibles con pi de Futex. [GH 1006]
 - Ten en cuenta que las prioridades no son actualmente una característica admitida de WSL, por lo que hay limitaciones, pero el uso estándar debería estar desbloqueado.
- Compatibilidad con el Firewall de Windows para los procesos de WSL. [GH 1852]

- Por ejemplo, para permitir que el proceso de Python de WSL escuche en cualquier puerto, use el cmd de Windows con privilegios elevados: `netsh.exe advfirewall firewall add rule name=wsl_python dir=in action=allow program="C:\users\<username>\appdata\local\packages\canonicalgrouplimited.ubuntuonwindows_79rhkp1fndgsc\localstate\rootfs\usr\bin\python2.7" enable=yes`
- Para más información sobre cómo agregar reglas de firewall, consulte el [vínculo ↗](#)
- Respeta el shell predeterminado del usuario al usar wsl.exe. [GH 2372]
- Informa de todas las interfaces de red como Ethernet. [GH 2996]
- Mejor control del archivo /etc/passwd dañado. [GH 3001]

Consola

- Sin correcciones.

Resultados de LTP:

Pruebas en curso.

Compilación 17618 (saltar)

Para obtener información general de Windows sobre la compilación 17618, visita el [blog de Windows ↗](#).

WSL

- Presenta la funcionalidad de pseudoconsola para la interoperabilidad de NT [GH 988, 1366, 1433, 1542, 2370, 2406].
- El mecanismo de instalación heredado (lxrun.exe) está en desuso. El mecanismo admitido para la instalación de distribuciones es a través de Microsoft Store.

Consola

- Sin correcciones.

Resultados de LTP:

Pruebas en curso.

Compilación 17110

Para obtener información general de Windows sobre la compilación 17110, visita el [blog de Windows](#).

WSL

- Permite que/init finalice desde Windows [GH 2928].
- DrvFs ahora usa la distinción de mayúsculas y minúsculas por directorio de manera predeterminada (equivalente a la opción de montaje "case=dir").
 - El uso de "case=force" (el comportamiento anterior) requiere del establecimiento de una clave de registro. Ejecuta el siguiente comando para habilitar "case=force" si necesitas usarlo: reg add HKLM\SYSTEM\CurrentControlSet\Services\lxss /v DrvFsAllowForceCaseSensitivity /t REG_DWORD /d 1
 - Si tienes directorios existentes creados con WSL en una versión anterior de Windows que deben distinguir entre mayúsculas y minúsculas, usa fsutil.exe para marcarlos con distinción de mayúsculas y minúsculas:fsutil.exe file setcasesensitiveinfo <path> enable
- La llamada del sistema uname devuelve cadenas de finalización NULL.

Consola

- Sin correcciones.

Resultados de LTP:

Pruebas en curso.

Compilación 17107

Para obtener información general de Windows sobre la compilación 17107, visita el [blog de Windows](#).

WSL

- Compatibilidad con TCSETSF y TCSETSW en los puntos de conexión de master pty [GH 2552].
- Iniciar procesos de interoperabilidad simultáneos puede dar lugar a EINVAL [GH 2813].

- Corrección de PTRACE_ATTACH para mostrar el estado de seguimiento adecuado en /proc/pid/status.
- Corrección de la carrera en la que los procesos de corta duración clonados con las marcas CLEARTID y SETTID podrían salir sin borrar la dirección de TID.
- Muestra un mensaje al actualizar los directorios del sistema de archivos de Linux al pasar de una compilación anterior a 17093. Para más detalles sobre los cambios del sistema de archivos de 17093, consulta las notas de la versión de [17093 ↗](#).

Consola

- Sin correcciones.

Resultados de LTP:

Pruebas en curso.

Compilación 17101

Para obtener información general de Windows sobre la compilación 17101, visita el [blog de Windows ↗](#).

WSL

- Compatibilidad con signalfd. [GH 129]
- Admite nombres de archivo que contienen caracteres NTFS no válidos mediante su codificación como caracteres Unicode privados. [GH 1514]
- El montaje automático se revertirá al modo de solo lectura cuando no se admita la escritura. [GH 2603]
- Permite pegar los pares suplentes Unicode (como los caracteres emoji). [GH 2765]
- Los pseudoarchivos en /proc y /sys deben devolver los valores read ready y write ready de select, poll, epoll, etc. [GH 2838]
- Corrección de un problema que podría hacer que el servicio pudiera entrar en un bucle infinito cuando el Registro se haya alterado o esté dañado.
- Corrección de los mensajes de netlink para trabajar con la versión más reciente (de nivel superior 4.14) de iproute2.

Consola

- Sin correcciones.

Resultados de LTP:

Pruebas en curso.

Compilación 17093

Para obtener información general de Windows sobre la compilación 17093, visita el [blog de Windows](#).

Importante:

Al iniciar WSL por primera vez después de actualizar a esta compilación, tiene que hacer algún trabajo para actualizar los directorios del sistema de archivos de Linux. Esto puede tardar varios minutos, por lo que es posible que WSL se inicie lentamente. Esto solo debería ocurrir una vez por cada distribución que haya instalado desde Store.

- Compatibilidad mejorada de distinción de mayúsculas y minúsculas en DrvFs.
 - DrvFs admite ahora la distinción de mayúsculas y minúsculas por directorio. Se trata de una nueva marca que se puede establecer en los directorios para indicar que todas las operaciones de esos directorios deben tratarse como con distinción de mayúsculas y minúsculas, lo que permite que las aplicaciones de Windows abran correctamente archivos que solo difieren en mayúsculas y minúsculas.
 - DrvFs tiene nuevas opciones de montaje que controlan la distinción de mayúsculas y minúsculas por directorio.
 - case=force: todos los directorios se tratan con distinción de mayúsculas y minúsculas (excepto la raíz de la unidad). Los nuevos directorios creados con WSL se marcan con distinción de mayúsculas y minúsculas. Este es el comportamiento heredado, excepto para marcar los nuevos directorios que distinguen mayúsculas de minúsculas.
 - case=dir: solo los directorios con la marca de distinción de mayúsculas y minúsculas por directorio se tratan como que distinguen mayúsculas de minúsculas; los otros directorios no distinguen mayúsculas de minúsculas. Los nuevos directorios creados con WSL se marcan con distinción de mayúsculas y minúsculas.
 - case=off: solo los directorios con la marca de distinción de mayúsculas y minúsculas por directorio se tratan como que distinguen mayúsculas de minúsculas; los otros directorios no distinguen mayúsculas de minúsculas. Los nuevos directorios creados con WSL se marcan como sin distinción de mayúsculas y minúsculas.

- Nota: Los directorios creados por WSL en versiones anteriores no tienen esta marca establecida, por lo que no se tratarán con la distinción entre mayúsculas y minúsculas si usas la opción "case=dir". Próximamente se presentará una manera de establecer esta marca en los directorios existentes.
- Ejemplo de montaje con estas opciones (en el caso de las unidades existentes, primero debes desmontar antes de poder montar con distintas opciones): sudo mount -t drvfs C: /mnt/c -o case=dir
- Por ahora, case=force sigue siendo la opción predeterminada. Se cambiará a case=dir en el futuro.
- Ahora puedes usar barras diagonales en las rutas de acceso de Windows al montar DrvFs, por ejemplo: sudo mount -t drvfs //server/share /mnt/share
- WSL ahora procesa el archivo /etc/fstab durante el inicio de la instancia [GH 2636].
 - Esto se hace antes de montar automáticamente las unidades de DrvFs; las unidades que ya se hayan montado mediante fstab no se volverán a montar automáticamente, lo que te permite cambiar el punto de montaje para unidades específicas.
 - Este comportamiento se puede desactivar mediante wsl.conf.
- Los archivos mount, mountinfo y mountstats de /proc convierten correctamente en caracteres especiales, como barras diagonales inversas y espacios [GH 2799]
- Los archivos especiales creados con DrvFs, como los vínculos simbólicos de WSL, o fifos y sockets cuando los metadatos están habilitados, ahora se pueden copiar y mover desde Windows.

WSL es más configurable con wsl.conf

Hemos agregado un método para que configures automáticamente cierta funcionalidad en WSL que se aplicará cada vez que inicies el subsistema. Esto incluye opciones de montaje automático y configuración de red. Obtén más información en la publicación de blog en <https://aka.ms/wslconf>

AF_UNIX permite conexiones de socket entre procesos de Linux en procesos nativos de WSL y Windows

Las aplicaciones de WSL y Windows ahora pueden comunicarse entre sí a través de sockets de Unix. Imagina que quieres ejecutar un servicio en Windows y hacer que esté disponible para las aplicaciones de WSL y Windows. Ahora, eso es posible con los sockets de Unix. Lee más en nuestra publicación de blog en <https://aka.ms/afunixinterop>

WSL

- Compatibilidad de mmap() con MAP_NORESERVE [GH 121, 2784]
- Compatibilidad con CLONE_PTRACE y CLONE_UNTRACED [GH 121, 2781]
- Controla la señal de terminación no SIGCHLD en el clon [GH 121, 2781]
- Código auxiliar /proc/sys/fs/inotify/max_user_instances y /proc/sys/fs/inotify/max_user_watches [GH 1705]
- Error al cargar binarios de ELF que contienen encabezados de carga con desplazamientos distintos de cero [GH 1884]
- Se eliminan los bytes de página finales al cargar imágenes.
- Reduce los casos en los que execve finaliza el proceso de forma silenciosa

Consola

- Sin correcciones.

Resultados de LTP:

Pruebas en curso.

Compilación 17083

Para obtener información general de Windows sobre la compilación 17083, visita el [blog de Windows](#).

WSL

- Se ha corregido la comprobación de errores relacionada con epoll [GH 2798, 2801, 2857]
- Se han corregido bloqueos al desactivar ASLR [GH 1185, 2870]
- Asegúrate de que las operaciones de mmap aparezcan como atómicas [GH 2732]

Consola

- Sin correcciones.

Resultados de LTP:

Pruebas en curso.

Compilación 17074

Para obtener información general de Windows sobre la compilación 17074, visita el [blog de Windows](#).

WSL

- Se ha corregido el formato de almacenamiento de los metadatos de DrvFs [GH 2777]
Importante: Los metadatos de DrvFs creados antes de esta compilación se mostrarán incorrectamente o no aparecerán en absoluto. Para corregir los archivos afectados, usa chmod y chown para volver a aplicar los metadatos.
- Se ha corregido un problema con varias señales y llamadas del sistema reinicias.

Consola

- Sin correcciones.

Resultados de LTP:

Pruebas en curso.

Compilación 17063

Para obtener información general de Windows sobre la compilación 17063, visita el [blog de Windows](#).

WSL

- DrvFs admite metadatos de Linux adicionales. Esto permite establecer el propietario y el modo de los archivos con chmod/chown, y también la creación de archivos especiales, como fifos, sockets de Unix y archivos de dispositivo. Esta opción está deshabilitada de forma predeterminada, ya que sigue siendo experimental. **Nota:** Se ha corregido un error en el formato de metadatos usado por DrvFs. Si bien los metadatos funcionan en esta compilación para experimentación, las compilaciones futuras no leerán correctamente los metadatos creados por esta compilación. Es posible que tengas que actualizar manualmente el propietario de los archivos modificados, y se tendrán que volver a crear los dispositivos con un identificador de dispositivo personalizado.

Para habilitar, monta DrvFs con la opción de metadatos (para habilitarlo en un montaje existente, primero debes desmontarlo):

Bash

```
mount -t drvfs C: /mnt/c -o metadata
```

Los permisos de Linux se agregan como metadatos adicionales al archivo; no afectan a los permisos de Windows. Recuerda que la edición de un archivo mediante un editor de Windows puede quitar los metadatos. En este caso, el archivo volverá a sus permisos predeterminados.

- Se han agregado opciones de montaje a DrvFs para controlar archivos sin metadatos.
 - uid: el identificador de usuario que se usa para el propietario de todos los archivos.
 - gid: el identificador de grupo que se usa para el propietario de todos los archivos.
 - umask: máscara octal de permisos que se van a excluir para todos los archivos y directorios.
 - fmask: máscara octal de permisos que se van a excluir para todos los archivos normales.
 - dmask: máscara octal de permisos que se van a excluir para todos los directorios.

Por ejemplo:

```
mount -t drvfs C: /mnt/c -o uid=1000,gid=1000,umask=22,fmask=111
```

Combina con la opción de metadatos para especificar los permisos predeterminados para los archivos sin metadatos.

- Se presentó una nueva variable de entorno, `WSLENV`, para configurar cómo fluyen las variables de entorno entre WSL y Win32.

Por ejemplo:

Bash

```
WSLENV=GOPATH/1:USERPROFILE/pu:DISPLAY
```

`WSLENV` es una lista de variables de entorno delimitadas por signos de dos puntos que se puede incluir al iniciar procesos de WSL desde Win32, o procesos de Win32

desde WSL. Cada variable puede tener como sufijo una barra diagonal seguida de marcas para especificar cómo se va a traducir.

- p: El valor es una ruta de acceso que se debe traducir entre las rutas de acceso de WSL y las rutas de acceso de Win32.
- l: El valor es una lista de rutas de acceso. En WSL, se trata de una lista delimitada por signos de dos puntos. En Win32, se trata de una lista delimitada por punto y coma.
- u: Solo se debe incluir el valor al invocar WSL desde Win32
- w: Solo se debe incluir el valor al invocar Win32 desde WSL

Puedes establecer `WSLENV` en `.bashrc` o en el entorno de Windows personalizado para el usuario.

- `drvfs` monta correctamente las marcas de tiempo de conservación desde `tar`, `cp -p` (GH 1939)
- Los vínculos simbólicos de `drvfs` informan el tamaño correcto (GH 2641)
- La lectura/escritura funciona para tamaños de E/S muy grandes (GH 2653)
- `waitpid` funciona con identificadores de grupo de procesos (GH 2534)
- Rendimiento de `mmap` significativamente mejorado para regiones de reserva grandes; mejora el rendimiento de `ghc` (GH 1671)
- Compatibilidad de `personality` con `READ_IMPLIES_EXEC`; corrige `maxima` y `clisp` (GH 1185)
- `mprotect` admite `PROT_GROWSDOWN`; corrige `clisp` (GH 1128)
- Correcciones de errores de página en modo de sobreconfirmación; corrige `sbsl` (GH 1128)
- `clone` admite más combinaciones de marcas
- Compatibilidad con `select/epoll` de archivos de `epoll` (antes de una no-op).
- Notifica a `ptrace` de llamadas del sistema sin implementar.
- Omite las interfaces que no están listas al generar los servidores de nombres `resolv.conf` [GH 2694]
- Enumera las interfaces de red sin dirección física. [GH 2685]
- Mejoras y correcciones de errores adicionales.

Herramientas de Linux disponibles para desarrolladores en Windows

- La cadena de herramientas de la línea de comandos de Windows incluye bsdtar (tar) y curl. Lee [este blog ↗](#) para obtener más información acerca de cómo agregar estas dos nuevas herramientas y ver cómo dan forma a la experiencia de los desarrolladores en Windows.
- AF_UNIX está disponible en el SDK de Windows Insider (17061+). Lee [este blog ↗](#) para más información sobre AF_UNIX y cómo pueden usarlo los desarrolladores de Windows.

Consola

- Sin correcciones.

Resultados de LTP:

Pruebas en curso.

Compilación 17046

Para obtener información general de Windows sobre la compilación 17046, visita el [blog de Windows ↗](#).

Fijo

WSL

- Permite que los procesos se ejecuten sin un terminal activo. [GH 709, 1007, 1511, 2252, 2391, etc.]
- Mejor compatibilidad con CLONE_VFORK y CLONE_VM. [GH 1878, 2615]
- Omite los controladores de filtro TDI para las operaciones de red de WSL. [GH 1554]
- DrvFs crea vínculos simbólicos de NT cuando se cumplen determinadas condiciones. [GH 353, 1475, 2602]
 - El destino del vínculo debe ser relativo, no debe cruzar ningún punto de montaje ni vínculo simbólico, y debe existir.
 - El usuario debe tener SE_CREATE_SYMBOLIC_LINK_PRIVILEGE (esto normalmente requiere que inicies wsl.exe con privilegios elevados), a menos

que esté activado el modo de desarrollador.

- En todas las demás situaciones, DrvFs todavía crea vínculos simbólicos de WSL.
- Permite ejecutar instancias de WSL con privilegios elevados y no elevados simultáneamente.
- Compatibilidad con /proc/sys/kernel/yama/ptrace_scope
- Agregue wslpath para realizar conversiones de rutas de acceso de WSL<->Windows. [GH 522, 1243, 1834, 2327, etc.]

```
wslpath usage:  
  -a      force result to absolute path format  
  -u      translate from a Windows path to a WSL path (default)  
  -w      translate from a WSL path to a Windows path  
  -m      translate from a WSL path to a Windows path, with '/' instead  
        of '\\'  
  
EX: wslpath 'c:\users'
```

Consola

- Sin correcciones.

Resultados de LTP:

Pruebas en curso.

Compilación 17040

Para obtener información general de Windows sobre la compilación 17040, visita el [blog de Windows](#).

Fijo

WSL

- No hay correcciones desde 17035.

Consola

- No hay correcciones desde 17035.

Resultados de LTP:

Pruebas en curso.

Compilación 17035

Para obtener información general de Windows sobre la compilación 17035, visita el [blog de Windows](#).

Fijo

WSL

- En ocasiones, el acceso a los archivos de DrvFs puede producir errores con EINVAL. [GH 2448]

Consola

- Algo de pérdida de color al insertar o eliminar líneas en modo VT.

Resultados de LTP:

Pruebas en curso.

Compilación 17025

Para obtener información general de Windows sobre la compilación 17025, visita el [blog de Windows](#).

Fijo

WSL

- Inicia procesos iniciales en un nuevo grupo de procesos en primer plano [GH 1653, 2510].
- Correcciones de entrega de SIGHUP [GH 2496].
- Genera un nombre de puente virtual predeterminado si no se proporciona ninguno [GH 2497].
- Implementa /proc/sys/kernel/random/boot_id [GH 2518].

- Más correcciones de la canalización stdout/stderr de interoperabilidad.
- Llamada del sistema syncfs de código auxiliar.

Consola

- Corrección de la traducción de VT de entrada para consolas de terceros [GH 111]

Resultados de LTP:

Pruebas en curso.

Compilación 17017

Para obtener información general de Windows sobre la compilación 17017, visita el [blog de Windows](#).

Fijo

WSL

- Omite los encabezados de programa ELF vacíos [GH 330].
- Permite que LxssManager cree instancias de WSL para usuarios no interactivos (compatibilidad con ssh y tareas programadas) [GH 777, 1602].
- Compatibilidad con escenarios de WSL->Win32->WSL ("inicio") [GH 1228].
- Compatibilidad limitada para la terminación de las aplicaciones de consola que se invocan a través de la interoperabilidad [GH 1614].
- Compatibilidad con las opciones de montaje para devpts [GH 1948].
- Inicio de bloqueo secundario de Ptrace [GH 2333].
- Faltan algunos eventos de EPOLLET [GH 2462].
- Devuelve más datos para PTRACE_GETSIGINFO.
- Getdents con Iseek proporciona resultados incorrectos.
- Corrección de algunos bloqueos de la aplicación de interoperabilidad de Win32, en espera de la entrada en una canalización que no tiene más datos.
- Compatibilidad de O_ASYNC con archivos tty/pty.
- Mejoras y correcciones de errores adicionales

Consola

- No hay cambios relacionados con la consola en esta versión.

Resultados de LTP:

Pruebas en curso.

Actualización Fall Creators Update

Compilación 16288

Para obtener información general de Windows sobre la compilación 16288, visita el [blog de Windows](#).

Fijo

WSL

- Inicializa y notifica correctamente uid, gid y mode para los descriptores de archivo de socket [GH 2490]
- Mejoras y correcciones de errores adicionales

Consola

- No hay cambios relacionados con la consola en esta versión.

Resultados de LTP:

Sin cambios desde 16273

Compilación 16278

Para obtener información general de Windows sobre la compilación 162738, visita el [blog de Windows](#).

Fijo

WSL

- Anula explícitamente la asignación de las vistas asignadas de secciones basadas en archivos al anular el estado de LX MM [GH 2415]

- Mejoras y correcciones de errores adicionales

Consola

- No hay cambios relacionados con la consola en esta versión.

Resultados de LTP:

Sin cambios desde 16273

Compilación 16275

Para obtener información general de Windows sobre la compilación 162735, visita el [blog de Windows ↗](#).

Fijo

WSL

- No hay cambios relacionados con WSL en esta versión.

Consola

- No hay cambios relacionados con la consola en esta versión.

Resultados de LTP:

Sin cambios desde 16273

Compilación 16273

Para obtener información general de Windows sobre la compilación 16273, visita el [blog de Windows ↗](#).

Fijo

WSL

- Se ha corregido un problema por el que DrvFs a veces indicaba un tipo de archivo incorrecto para los directorios [GH 2392]
- Permite la creación de sockets NETLINK_KOBJECT_UEVENT para desbloquear programas que usan UEVENT [GH 1121, 2293, 2242, 2295, 2235, 648, 637]
- Agrega compatibilidad para conexión sin bloqueo [GH 903, 1391, 1584, 1585, 1829, 2290, 2314]
- Implementa la marca de llamada del sistema de clon CLONE_FS [GH 2242]
- Corrección de problemas relativos a la falta de control de tabulaciones o comillas correctamente en la interoperabilidad de NT [GH 1625, 2164]
- Resuelve el error de acceso denegado al intentar volver a iniciar instancias de WSL [GH 651, 2095]
- Implementa las operaciones de Futex FUTEX_REQUEUE y FUTEX_CMP_REQUEUE [GH 2242]
- Corrección de permisos para varios archivos de SysFs [GH 2214]
- Corrección del bloqueo de pila de Haskell durante la instalación [GH 2290]
- Implementa las marcas binfmt_misc "C", "O" y "P" [GH 2103]
- Agregue /proc/sys/kernel /shmmmax /shmmn & /threads-max [GH 1753].
- Agrega compatibilidad parcial para la llamada del sistema ioprio_set [GH 498]
- Código auxiliar SO_REUSEPORT & que agrega compatibilidad para SO_PASSCRED para sockets de netlink [GH 69].
- Devuelve distintos códigos de error de RegisterDistribuiton si se está instalando o desinstalando actualmente una distribución.
- Permite la anulación del registro de distribuciones de WSL parcialmente instaladas a través de wslconfig.exe
- Corrección de bloqueo de prueba de socket de Python desde UDP::msg_peek
- Mejoras y correcciones de errores adicionales

Consola

- No hay cambios relacionados con la consola en esta versión.

Resultados de LTP:

Total de pruebas: 1904

Total de pruebas omitidas: 209

Total de errores: 229

Compilación 16257

Para obtener información general de Windows sobre la compilación 16257, visita el [blog de Windows](#).

Fijo

WSL

- Implementa la llamada del sistema prlimit64
- Agrega compatibilidad con ulimit -n (setrlimit RLIMIT_NOFILE) [GH 1688]
- Código auxiliar de MSG_MORE para sockets TCP [GH 2351]
- Corrección del comportamiento del vector auxiliar de AT_EXECAF no válido [GH 2133]
- Corrección del comportamiento de copiar y pegar para consola/tty, y agrega un mejor control de búfer completo [GH 2204, 2131]
- Establece AT_SECURE en vector auxiliar para los programas set-user-ID y set-group-ID [GH 2031]
- El punto de conexión maestro de psuedoterminal no controla TIOCPGRP [GH 1063]
- La corrección de lseek hace rebobinar directorios en LxFs [GH 2310]
- /dev/ptmx se bloquea después de un uso pesado [GH 1882]
- Mejoras y correcciones de errores adicionales

Consola

- Corrección para líneas horizontales o guiones bajos en todas partes [GH 2168]
- Corrección para el orden de procesamiento cambiado hace que NPM sea más difícil de cerrar [GH 2170]
- Se ha agregado la nueva combinación de colores:
<https://blogs.msdn.microsoft.com/commandline/2017/08/02/updating-the-windows-console-colors/>

Resultados de LTP:

Sin cambios desde 16251

Compatibilidad con llamadas del sistema

A continuación se muestra una lista de llamadas del sistema nuevas o mejoradas que tienen alguna implementación en WSL. Las llamadas del sistema de esta lista se admiten

en al menos un escenario, pero puede que no se admitan todos los parámetros en este momento.

prlimit64

Problemas conocidos

Problema de GitHub 2392: WSL no reconoce carpetas de Windows... ↗

En la compilación 16257, WSL tiene problemas al enumerar archivos o carpetas de Windows a través de `/mnt/c/....`. Este problema se ha corregido y debe publicarse en la compilación de Insider durante la semana que comienza el 14/8/2017.

Compilación 16251

Para obtener información general de Windows sobre la compilación 16251, visita el [blog de Windows](#) ↗ .

Fijo

WSL

- Quita la etiqueta beta del componente opcional de WSL, consulta la [publicación de blog](#) ↗ para más información.
- Inicializa correctamente el uid y gid del conjunto guardado para los archivos binarios set-user-ID y set-group-ID en ejecución [GH 962, 1415, 2072]
- Se ha agregado compatibilidad con ptrace PTRACE_O_TRACEEXIT [GH 555]
- Se ha agregado compatibilidad con ptrace PTRACE_GETFPREGS y PTRACE_GETREGSET con NT_FPREGSET [GH 555]
- Se ha corregido ptrace para que se detenga en las señales omitidas
- Mejoras y correcciones de errores adicionales

Consola

- No hay cambios relacionados con la consola en esta versión.

Resultados de LTP:

Número de pruebas superadas: 768
Número de pruebas no superadas: 244
Número de pruebas omitidas: 96

Compilación 16241

Para obtener información general de Windows sobre la compilación 16241, visita el [blog de Windows](#).

Fijo

WSL

- No hay cambios relacionados con WSL en esta versión.

Consola

- Corrección para la generación de un carácter equivocado para DEC de líneas cruzadas, que se notificó originalmente [aquí](#)
- Corrección para cuando no se muestre texto de salida en la página de códigos 65001 (utf8)
- No transfieras los cambios hechos en los valores RGB de un color a otras partes de la paleta en el cambio de selección. Esto hará que la hoja de propiedades de la consola sea mucho más fácil de usar.
- Ctrl+S no parece funcionar correctamente
- Un-Bold/-Dim totalmente ausentes de los códigos de escape de ANSI [GH 2174]
- La consola no admite correctamente los temas de color VIM [GH 1706]
- No se pueden pegar caracteres determinados [GH 2149]
- El cambio de tamaño de reflujo interactúa de forma extraña con el cambio de tamaño de una ventana de Bash cuando el contenido está en la línea de comandos/edición [GH ConEmu 1123]
- Ctrl-L deja la pantalla desfasada [GH 1978]
- Error de representación de consola al mostrar VT en HDPI [GH 1907]
- Los caracteres Japansese parecen extraños con el carácter Unicode U+30FB [GH 2146]
- Mejoras y correcciones de errores adicionales

Compilación 16237

Para obtener información general de Windows sobre la compilación 16237, visita el [blog de Windows](#).

Fijo

- Usa atributos predeterminados para archivos sin EA en lxfs (root, root, 0000)
- Se ha agregado compatibilidad para distribuciones que usan atributos extendidos
- Corrección del relleno de las entradas devueltas por getdents y getdents64
- Corrección de la comprobación de permisos para la llamada del sistema shmctl SHM_STAT [GH 2068]
- Se ha corregido el estado inicial incorrecto de epoll para ttys [GH 2231]
- Corrección de readdir de DrvFs que no devuelve todas las entradas [GH 2077]
- Corrección de readdir de LxFs cuando los archivos están desvinculados [GH 2077]
- Permite que los archivos drvfs no vinculados se vuelvan a abrir a través de procfs
- Se ha agregado la invalidación de la clave del Registro global para deshabilitar las características de WSL (interoperabilidad/montaje de unidades)
- Corrección del recuento incorrecto de bloques en "stat" para DrvFs (y LxFs) [GH 1894]
- Mejoras y correcciones de errores adicionales

Compilación 16232

Para obtener información general de Windows sobre la compilación 16232, visita el [blog de Windows](#).

Fijo

- No hay cambios relacionados con WSL en esta versión.

Compilación 16226

Para obtener información general de Windows sobre la compilación 16226, visita el [blog de Windows](#).

Fijo

- Compatibilidad con llamadas del sistema relacionadas con xattr (getxattr, setxattr, listxattr, removexattr).
- Compatibilidad con security.capability xattr.
- Compatibilidad mejorada con determinados sistemas de archivos y filtros, incluidos los servidores SMB que no son de MS. [GH 1952]
- Compatibilidad mejorada para los marcadores de posición de OneDrive, los marcadores de posición de GVFS y los archivos comprimidos del sistema operativo compacto.
- Mejoras y correcciones de errores adicionales

Compilación 16215

Para obtener información general de Windows sobre la compilación 16215, visita el [blog de Windows](#).

Fijo

- WSL ya no requiere el modo de desarrollador.
- Compatibilidad con uniones de directorios en drvfs.
- Controla la desinstalación de paquetes appx de distribución de WSL.
- Actualiza procfs para mostrar las asignaciones privadas y compartidas.
- Agrega la capacidad de wslconfig.exe de limpiar las distribuciones que están parcialmente instaladas o desinstaladas.
- Se ha agregado compatibilidad para IP_MTU_DISCOVER para sockets TCP. [GH 1639, 2115, 2205]
- Inferencia de la familia de protocolos para rutas a AF_INADDR.
- Mejoras en los dispositivos serie [GH 1929].

Compilación 16199

Para obtener información general de Windows sobre la compilación 16199, visita el [blog de Windows](#).

Fijo

- No hay cambios relacionados con WSL en estas versiones.

Compilación 16193

Para obtener información general de Windows sobre la compilación 16193, visita el [blog de Windows](#).

Fijo

- Condición de carrera entre el envío de SIGCONT y un grupo de subprocessos de terminación [GH 1973]
- Cambia dispositivos tty y pty para notificar FILE_DEVICE_NAMED_PIPE en lugar de FILE_DEVICE_CONSOLE [GH 1840]
- Corrección de SSH para IP_OPTIONS
- Se ha cambiado el montaje de DrvFs al demonio de inicialización [GH 1862, 1968, 1767, 1933]
- Se ha agregado compatibilidad en DrvFs para los siguientes vínculos simbólicos de NT.

Compilación 16184

Para obtener información general de Windows sobre la compilación 16184, visita el [blog de Windows](#).

Fijo

- Se ha quitado la tarea de mantenimiento de paquetes APT (lxrun.exe /update)
- Se ha corregido la salida que no se muestra en los procesos de Windows en node.js [GH 1840]
- Relaja los requisitos de alineación en lxcore [GH 1794]
- Se ha corregido el control de la marca AT_EMPTY_PATH en un número de llamadas del sistema.
- Se ha corregido el problema por el que la eliminación de archivos DrvFs con identificadores abiertos hace que el archivo muestre un comportamiento indefinido [GH 544, 966, 1357, 1535, 1615]

- /etc/hosts heredará ahora entradas del archivo de hosts de Windows (%windir%\system32\drivers\etc\hosts) [GH 1495]

Compilación 16179

Para obtener información general de Windows sobre la compilación 16179, visita el [blog de Windows](#).

Fijo

- Sin cambios en WSL esta semana.

Compilación 16176

Para obtener información general de Windows sobre la compilación 16176, visita el [blog de Windows](#).

Fijo

- [Se ha habilitado la compatibilidad serie](#)
- Se ha agregado la opción de socket de IP IP_OPTIONS [GH 1116]
- Se ha implementado la función pwritev (al cargar el archivo en nginx/PHP-FPM) [GH 1506]
- Se han agregado opciones de socket IP IP_MULTICAST_IF & IPV6_MULTICAST_IF [GH 990].
- Compatibilidad con la opción de socket IP_MULTICAST_LOOP & IPV6_MULTICAST_LOOP [GH 1678].
- Se ha agregado la opción de socket IP(V6)_MTU para el nodo de aplicaciones, traceroute, dig, nslookup, host
- Se ha agregado la opción de socket de IP IPV6_UNICAST_HOPS
- [Mejoras del sistema de archivos](#)
 - Permite el montaje de rutas UNC
 - Habilita la compatibilidad con CDFS en drvfs
 - Controla correctamente los permisos para los sistemas de archivos de red en drvfs
 - Agrega compatibilidad para unidades remotas a drvfs

- Habilita la compatibilidad con CDFS en drvfs
- Mejoras y correcciones adicionales

Resultados de LTP

Sin cambios desde 15042

Compilación 16170

Para obtener información general de Windows sobre la compilación 16170, visita el [blog de Windows](#).

Hemos lanzado una nueva [publicación de blog](#) en la que se describen nuestros esfuerzos para probar WSL.

Fijo

- Compatibilidad para la opción de socket IP_ADD_MEMBERSHIP & IPV6_ADD_MEMBERSHIP [GH 1678].
- Agrega compatibilidad para PTRACE_OLDSETOPTIONS. [GH 1692]
- Mejoras y correcciones adicionales

Resultados de LTP

Sin cambios desde 15042

Compilación 15046 a Windows 10 Creators Update

No hay más correcciones o características de WSL planeadas para su inclusión en Windows 10 Creators Update. Las notas de la versión de WSL se reanudarán en las próximas semanas para las adiciones destinadas a la siguiente actualización principal de Windows. Para obtener información general de Windows sobre la compilación 15046 y futuras publicaciones de Insider, visita el [blog de Windows](#).

Compilación 15042

Para obtener información general de Windows sobre la compilación 15042, visita el [blog de Windows](#).

Fijo

- Corrección para un interbloqueo al quitar una ruta de acceso que termina en ".."
- Se ha corregido un problema por el que FIONBIO no devuelve 0 en caso correcto [GH 1683]
- Se ha corregido un problema con lecturas de longitud cero de sockets de datagramas de inet
- Corrección del posible interbloqueo debido a la condición de carrera en la búsqueda de inode de drvfs [GH 1675]
- Se ha ampliado la compatibilidad con los datos suplementarios de los sockets de Unix; SCM_CREDENTIALS y SCM_RIGHTS [GH 514, 613, 1326]
- Mejoras y correcciones adicionales

Resultados de LTP:

Número de pruebas superadas: 737

Número de pruebas no superadas (con error, omitidas, etc.): 255

Compilación 15031

Para obtener información general de Windows sobre la compilación 15031, visita el [blog de Windows](#).

Fijo

- Se ha corregido un error en el que el time(2) se comportaba incorrectamente de manera esporádica.
- Se ha corregido un problema donde las llamadas del sistema de *SIGPROCMASK podían dañar la máscara de señal.
- Ahora, devuelva la longitud completa de la línea de comandos en la notificación de creación del proceso de WSL. [GH 1632]
- WSL ahora informa de la salida del subprocesso a través de ptrace para bloqueos de GDB. [GH 1196]

- Se ha corregido un error en el que ptys se bloqueaba después de una gran E/S de tmux [GH 1358]
- Se ha corregido la validación del tiempo de espera en muchas llamadas del sistema (futex, semtimedop, ppoll, sigtimedwait, itimer, timer_create)
- Se ha agregado compatibilidad con eventfd EFD_SEMAPHORE [GH 452]
- Mejoras y correcciones adicionales

Resultados de LTP:

Número de pruebas superadas: 737

Número de pruebas no superadas (con error, omitidas, etc.): 255

Compilación 15025

Para obtener información general de Windows sobre la compilación 15025, visita el [blog de Windows](#).

Fijo

- Corrección del error que dañó grep 2.27 [GH 1578]
- Se ha implementado la marca EFD_SEMAPHORE para la llamada del sistema eventfd2 [GH 452]
- Se ha implementado /proc/[pid]/net/ipv6_route [GH 1608]
- Compatibilidad de E/S controlada por señal para sockets de secuencia de Unix [GH 393, 68]
- Compatibilidad con F_GETPIPE_SZ y F_SETPIPE_SZ [GH 1012]
- Implementa la llamada del sistema recvmsg() [GH 1531]
- Se ha corregido un error en el que epoll_wait() no esperaba [GH 1609]
- Implementa /proc/version_signature
- La llamada del sistema Tee ahora devuelve un error si ambos descriptores de archivo hacen referencia a la misma canalización
- Se ha implementado el comportamiento correcto de SO_PEERCRED para sockets de Unix
- Se ha corregido el control de parámetros no válidos de llamada del sistema tkill
- Cambios para aumentar el rendimiento de drvfs
- Corrección menor para el bloqueo de E/S de Ruby
- Se ha corregido recvmsg() que devuelve EINVAL para la marca MSG_DONTWAIT para sockets inet [GH 1296]
- Mejoras y correcciones adicionales

Resultados de LTP:

Número de pruebas superadas: 732

Número de pruebas no superadas (con error, omitidas, etc.): 255

Compilación 15019

Para obtener información general de Windows sobre la compilación 15019, visita el [blog de Windows](#).

Fijo

- Se ha corregido un error que informaba incorrectamente del uso de CPU en procfs para herramientas como htop (GH 823, 945, 971)
- Al llamar a open() con O_TRUNC en un archivo existente inotify ahora genera IN MODIFY antes de IN_OPEN
- Correcciones para el SO_ERROR de sockets de Unix getsockopt para habilitar postgres [GH 61, 1354]
- Implementa /proc/sys/net/core/somaxconn para el lenguaje GO
- La tarea en segundo plano de actualización del paquete apt-get ahora se ejecuta oculta de la vista
- Borra el ámbito de IPv6 localhost (error de Spring-Framework (Java)).
- Mejoras y correcciones adicionales

Resultados de LTP:

Número de pruebas superadas: 714

Número de pruebas no superadas (con error, omitidas, etc.): 249

Compilación 15014

Para obtener información general de Windows sobre la compilación 15014, visita el [blog de Windows](#).

Fijo

- Ctrl + C ahora funciona según lo previsto
- htop y ps auxw ahora muestran el uso de recursos correcto (GH 516)

- Traducción básica de excepciones de NT a señales. (GH 513)
- fallocate ahora genera el error ENOSPC cuando se está quedando sin espacio en lugar de EINVAL (GH 1571)
- Se ha agregado/proc/sys/kernel/sem.
- Se han implementado las llamadas del sistema semop y semtimedop
- Se han corregido errores de nslookup con la opción de socket IP_RECVTOS & IPV6_RECVTCLASS (GH 69)
- Compatibilidad con las opciones de socket IP_RECVTTL e IPV6_RECVHOPLIMIT
- Mejoras y correcciones adicionales

Resultados de LTP:

Número de pruebas superadas: 709

Número de pruebas no superadas (con error, omitidas, etc.): 255

Resumen de llamadas del sistema

Total de llamadas del sistema: 384

Total implementado: 235

Total de procesos con stub: 22

Total no implementado: 127

Compilación 15007

Para obtener información general de Windows sobre la compilación 15007, visita el [blog de Windows](#).

Problema conocido

- Hay un error conocido en el que la consola no reconoce algunas entradas Ctrl + `<key>`. Esto incluye al comando Ctrl-C, que actuará como una pulsación normal de la tecla "c".
 - Solución: Asigna una tecla alternativa a Ctrl+C. Por ejemplo, para asignar Ctrl+K a Ctrl+C, haz lo siguiente: `stty intr \^k`. Esta asignación es por terminal y tendrá que hacerse *cada vez* que se inicie Bash. Los usuarios pueden explorar la opción para incluirlo en su `.bashrc`

Fijo

- Se ha corregido el problema que hacía que la ejecución de WSL consumiera el 100 % de un núcleo de CPU.
- Ahora se admite la opción de socket IP_PKTINFO, IPV6_RECV_PKTINFO. (GH 851, 987)
- Trunca la dirección física de la interfaz de red a 16 bytes en Ixcore (GH 1452, 1414, 1343, 468, 308)
- Mejoras y correcciones adicionales

Resultados de LTP:

Número de pruebas superadas: 709

Número de pruebas no superadas (con error, omitidas, etc.): 255

Compilación 15002

Para obtener información general de Windows sobre la compilación 15002, visita el [blog de Windows](#).

Problema conocido

Dos problemas conocidos:

- Hay un error conocido en el que la consola no reconoce algunas entradas Ctrl + <key>. Esto incluye al comando Ctrl-C, que actuará como una pulsación normal de la tecla "c".
 - Solución: Asigna una tecla alternativa a Ctrl+C. Por ejemplo, para asignar Ctrl+K a Ctrl+C, haz lo siguiente: `stty intr \^k`. Esta asignación es por terminal y tendrá que hacerse *cada vez* que se inicie Bash. Los usuarios pueden explorar la opción para incluirlo en su `.bashrc`
- Mientras se ejecuta WSL, un subprocesso del sistema consumirá el 100 % de un núcleo de CPU. La raíz de esto se ha solucionado y corregido internamente.

Fijo

- Todas las sesiones de Bash ahora deben crearse en el mismo nivel de permiso. Se bloquearán los intentos de iniciar una sesión en un nivel diferente. Esto significa que las consolas de administrador y que no son de administración no se pueden ejecutarse al mismo tiempo. (GH 626)

- Se han implementado los siguientes mensajes NETLINK_ROUTE (requiere el administrador de Windows)
 - RTM_NEWADDR (admite `ip addr add`)
 - RTM_NEWRROUTE (admite `ip route add`)
 - RTM_DELADDR (admite `ip addr del`)
 - RTM_DELROUTE (admite `ip route del`)
- La comprobación de tareas programadas para actualizar los paquetes ya no se ejecutará en una conexión de uso medido (GH 1371)
- Se ha corregido un error en el que la canalización se bloquea, es decir, `bash -c "ls -alR /" | bash -c "cat"` (GH 1214)
- Se ha implementado la opción de socket TCP_KEEPCNT (GH 843)
- Se ha implementado la opción de socket IP_MTU_DISCOVER INET (GH 720, 717, 170, 69)
- Se ha quitado la funcionalidad heredada para ejecutar archivos binarios de NT desde init con búsqueda de rutas de NT. (GH 1325)
- Corrección del modo de `/dev/kmsg` para permitir el acceso de lectura de grupo/otro (0644) (GH 1321)
- Se ha implementado `/proc/sys/kernel/random/uuid` (GH 1092)
- Se ha corregido el error en el que la hora de inicio del proceso se mostraba como el año 2432 (GH 974)
- Se ha cambiado la variable de entorno TERM predeterminada a `xterm-256color` (GH 1446)
- Se ha modificado la forma en que se calcula la confirmación del proceso durante la bifurcación del proceso. (GH 1286)
- Se ha implementado `/proc/sys/vm/overcommit_memory`. (GH 1286)
- Se ha implementado el archivo `/proc/net/route` (GH 69)
- Se ha corregido el error en el que el nombre de acceso directo se localizaba incorrectamente (GH 696)
- Se ha corregido la lógica de análisis de `elf` que valida incorrectamente los encabezados de programa debe ser menor que (o igual a) `PATH_MAX`. (GH 1048)
- Se ha implementado la devolución de llamada de `statfs` para `procfs`, `sysfs`, `cgroupfs` y `binfmtfs` (GH 1378)
- Se han corregido ventanas de `AptPackageIndexUpdate` que no se cerraban (GH 1184, también se describe en GH 1193)
- Se ha agregado compatibilidad con la personalidad de ASLR `ADDR_NO_RANDOMIZE`. (GH 1148, 1128)
- Se han mejorado `PTRACE_GETSIGINFO`, `SIGSEGV` para los seguimientos de la pila `gdb` adecuados durante AV (GH 875)
- Ya no se produce un error en el análisis de `elf` para los archivos binarios de `patchelf`. (GH 471)

- DNS de VPN propagado a/etc/resolv.conf (GH 416, 1350)
- Mejoras en el cierre de TCP para la transferencia de datos más confiable. (GH 610, 616, 1025, 1335)
- Ahora devuelve el código de error correcto cuando se abran demasiados archivos (EMFILE). (GH 1126, 2090)
- El registro de auditoría de Windows ahora informa el nombre de la imagen en la creación del proceso de auditoría.
- Ahora se producen errores leves al iniciar bash.exe desde una ventana de Bash
- Se ha agregado un mensaje de error cuando interop no puede acceder a un directorio de trabajo en LxFs (es decir, notepad.exe .bashrc)
- Se ha corregido un problema en el que la ruta de acceso de Windows se truncaba en WSL
- Mejoras y correcciones adicionales

Resultados de LTP:

Número de pruebas superadas: 690

Número de pruebas no superadas (con error, omitidas, etc.): 274

Compatibilidad con llamadas del sistema

A continuación se muestra una lista de llamadas del sistema nuevas o mejoradas que tienen alguna implementación en WSL. Las llamadas del sistema de esta lista se admiten en al menos un escenario, pero puede que no se admitan todos los parámetros en este momento.

```
shmctl  
shmget  
shmdt  
shmat
```

Compilación 14986

Para obtener información general de Windows sobre la compilación 14986, visita el [blog de Windows](#).

Fijo

- Se han corregido comprobaciones de error con IOCTL de NetLink y Pty
- La versión del kernel ahora informa 4.4.0-43 por coherencia con Xenial
- Bash.exe ahora se inicia cuando la entrada se dirige a "nul" (GH 1259)
- Los identificadores de subprocesso ahora se han declarado correctamente en procfs (GH 967)
- Las marcas IN_UNMOUNT | IN_Q_OVERFLOW | IN_IGNORED | IN_ISDIR ahora se admiten en inotify_add_watch() (GH 1280)
- Implementa timer_create y llamadas del sistema relacionadas. Esto habilita la compatibilidad con GHC (GH 307)
- Se ha corregido un problema en el que ping devolvía una hora de 0,000 ms (GH 1296)
- Devuelve el código de error correcto cuando se abran demasiados archivos.
- Se ha corregido un problema en WSL donde la solicitud de NetLink de datos de la interfaz de red generaba un error con EINVAL si la dirección de hardware de la interfaz es de 32 bytes (por ejemplo, la interfaz Teredo)
 - Ten en cuenta que la utilidad "ip" de Linux contiene un error en el que se bloqueará si WSL informa de una dirección de hardware de 32 bytes. Se trata de un error en "ip", no en WSL. La utilidad "ip" codifica de forma rígida la longitud del búfer de cadena que se usa para imprimir la dirección de hardware, y el búfer es demasiado pequeño como para imprimir una dirección de hardware de 32 bytes.
- Mejoras y correcciones adicionales

Resultados de LTP:

Número de pruebas superadas: 669

Número de pruebas no superadas (con error, omitidas, etc.): 258

Compatibilidad con llamadas del sistema

A continuación se muestra una lista de llamadas del sistema nuevas o mejoradas que tienen alguna implementación en WSL. Las llamadas del sistema de esta lista se admiten en al menos un escenario, pero puede que no se admitan todos los parámetros en este momento.

`timer_create`

`timer_delete`

`timer_gettime`

Compilación 14971

Para obtener información general de Windows sobre la compilación 14971, visita el [blog de Windows](#).

Fijo

- Debido a circunstancias más allá de nuestro control, no hay ninguna actualización en esta compilación para el subsistema de Windows para Linux. Las actualizaciones programadas regularmente se reanudarán en la próxima versión.

Resultados de LTP:

Sin cambios desde 14965

Número de pruebas superadas: 664

Número de pruebas no superadas (con error, omitidas, etc.): 263

Compilación 14965

Para obtener información general de Windows sobre la compilación 14965, visita el [blog de Windows](#).

Fijo

- Compatibilidad con RTM_GETLINK y RTM_GETADDR del protocolo NETLINK_ROUTE de sockets de Netlink (GH 468)
 - Habilita los comandos ifconfig e ip para la enumeración de red
- /sbin está ahora en la ruta de acceso del usuario de manera predeterminada
- La ruta de acceso de usuario de NT ahora se anexa a la ruta de acceso de WSL de manera predeterminada (es decir, ahora puedes escribir notepad.exe sin agregar System32 a la ruta de acceso de Linux)
- Se ha agregado compatibilidad para /proc/sys/kernel/cap_last_cap

- Los archivos binarios de NT ahora se pueden iniciar desde WSL cuando el directorio de trabajo actual contiene caracteres que no son ANSI (GH 1254)
- Permite el apagado en el socket de flujo Unix desconectado.
- Se ha agregado compatibilidad para PR_GET_PDEATHSIG.
- Se ha agregado compatibilidad para CLONE_PARENT
- Se ha corregido un error en el que la canalización se bloquea, es decir, bash -c "ls -alR /" | bash -c "cat" (GH 1214)
- Controla las solicitudes para conectarse al terminal actual.
- Marca `/proc/<pid>/oom_score_adj` como grabable.
- Agrega la carpeta/sys/fs/cgroup.
- sched_setaffinity debe devolver el número de máscara de bits de afinidad
- Corrige la lógica de validación de ELF que presupone incorrectamente que las rutas de intérprete deben tener menos de 64 caracteres de longitud. (GH 743)
- Los descriptores de archivos abiertos pueden mantener abierta la ventana de la consola (GH 1187)
- Se ha corregido el error en el que no se podía completar rename() con la barra diagonal final en el nombre de destino (GH 1008)
- Implementa el archivo /proc/net/dev
- Se han corregido los pings de 0,000 ms debido a la resolución del temporizador.
- Se ha implementado /proc/self/environ (GH 730)
- Correcciones de errores y mejoras adicionales

Resultados de LTP:

Número de pruebas superadas: 664

Número de pruebas no superadas (con error, omitidas, etc.): 263

Compilación 14959

Para obtener información general de Windows sobre la compilación 14959, visita el [blog de Windows](#).

Fijo

- Se ha mejorado la notificación del proceso PICO para Windows. Encuentra información adicional en el [blog de WSL](#).
- Se ha mejorado la estabilidad de interoperabilidad con Windows
- Se ha corregido el error 0x80070057 al iniciar bash.exe cuando Protección de datos de empresa (EDP) está habilitado
- Correcciones de errores y mejoras adicionales

Resultados de LTP:

Número de pruebas superadas: 665

Número de pruebas no superadas (con error, omitidas, etc.): 263

Compilación 14955

Para obtener información general de Windows sobre la compilación 14955, visita el [blog de Windows](#).

Fijo

- Debido a circunstancias más allá de nuestro control, no hay ninguna actualización en esta compilación para el subsistema de Windows para Linux. Las actualizaciones programadas regularmente se reanudarán en la próxima versión.

Resultados de LTP:

Número de pruebas superadas: 665

Número de pruebas no superadas (con error, omitidas, etc.): 263

Compilación 14951

Para obtener información general de Windows sobre la compilación 14951, visita el [blog de Windows](#).

Nueva característica: Interoperabilidad de Windows/Ubuntu

Ahora se pueden invocar archivos binarios de Windows directamente desde la línea de comandos de WSL. Esto da a los usuarios la capacidad de interactuar con el entorno y el sistema Windows de una manera que antes no era posible. Como ejemplo rápido, ahora es posible que los usuarios ejecuten los siguientes comandos:

Bash

```
$ export PATH=$PATH:/mnt/c/Windows/System32  
$ notepad.exe  
$ ipconfig.exe | grep IPv4 | cut -d: -f2  
$ ls -la | findstr.exe foo.txt  
$ cmd.exe /c dir
```

Puedes encontrar más información en:

- [Blog del equipo de WSL para Interop](#)
- [Documentación sobre sistemas de archivos WSL](#)

Fijo

- Ubuntu 16.04 (Xenial) ahora se instala para todas las instancias nuevas de WSL. Los usuarios con instancias existentes de 14.04 (Trusty) no se actualizarán automáticamente.
- Ahora se muestra la configuración regional establecida durante la instalación
- Mejoras del terminal, que incluido un error en el que la redirección de un proceso de WSL a un archivo no siempre funciona
- La duración de la consola debe estar asociada a la duración de bash.exe
- El tamaño de la ventana de la consola debe usar un tamaño visible, no el tamaño del búfer
- Correcciones de errores y mejoras adicionales

Resultados de LTP:

Número de pruebas superadas: 665

Número de pruebas no superadas (con error, omitidas, etc.): 263

Compilación 14946

Para obtener información general de Windows sobre la compilación 14946, visita el [blog de Windows](#).

Fijo

- Se ha corregido un problema que impedía la creación de cuentas de usuario de WSL para los usuarios con nombres de usuario de NT que contienen espacios o comillas.
- Cambia VolFs y DrvFs para que devuelvan 0 para el recuento de vínculos de directorio en stat
- Compatibilidad con la opción de socket IPV6_MULTICAST_HOPS.
- Limite a un único bucle de E/S de la consola por tty. Por ejemplo, el comando siguiente es posible:
 - bash -c "echo data" | bash -c "ssh user@example.com 'cat > foo.txt'"
- Reemplaza espacios con tabulaciones en/proc/cpuinfo (GH 1115)
- DrvFs aparece ahora en mountinfo con un nombre que coincide con el volumen de Windows montado
- /home y /root ahora aparecen en mountinfo con los nombres correctos
- Correcciones de errores y mejoras adicionales

Resultados de LTP:

Número de pruebas superadas: 665

Número de pruebas no superadas (con error, omitidas, etc.): 263

Compilación 14942

Para obtener información general de Windows sobre la compilación 14942, visita el [blog de Windows](#).

Fijo

- Se solucionaron varias comprobaciones de errores, incluido el bloqueo de red "ATTEMPTED EXECUTE OF NOEXECUTE MEMORY" que bloqueaba SSH

- Ahora se incluye la compatibilidad de inotify con las notificaciones generadas desde aplicaciones Windows en DrvFs
- Implementa TCP_KEEPIDLE y TCP_KEEPINTVL para mongod. (GH 695)
- Implementa la llamada del sistema pivot_root
- Implementa la opción de socket para SO_DONTROUTE
- Correcciones de errores y mejoras adicionales

Resultados de LTP:

Número de pruebas superadas: 665

Número de pruebas no superadas (con error, omitidas, etc.): 263

Compatibilidad con llamadas del sistema

A continuación se muestra una lista de llamadas del sistema nuevas o mejoradas que tienen alguna implementación en WSL. Las llamadas del sistema de esta lista se admiten en al menos un escenario, pero puede que no se admitan todos los parámetros en este momento.

`pivot_root`

Compilación 14936

Para obtener información general de Windows sobre la compilación 14936, visita el [blog de Windows](#).

Nota: WSL instalará Ubuntu versión 16.04 (Xenial) en lugar de Ubuntu 14.04 (Trusty) en una próxima versión. Este cambio se aplicará a los usuarios de Insider que instalen nuevas instancias (lxrun.exe /install o a la primera ejecución de bash.exe). Las instancias existentes con Trusty no se actualizarán automáticamente. Los usuarios pueden actualizar su imagen de Trusty a Xenial con el comando do-release-upgrade.

Problema conocido

WSL está experimentando un problema con algunas implementaciones de sockets. La comprobación de errores se manifiesta como un bloqueo con el error "ATTEMPTED EXECUTE OF NOEXECUTE MEMORY". La manifestación más común de este problema es un bloqueo cuando se usa ssh. La causa principal se ha corregido en las compilaciones internas y se enviará a los usuarios de Insider lo antes posible.

Fijo

- Se ha implementado la llamada del sistema chroot
- Mejoras en inotify, ~~incluida la compatibilidad con las notificaciones generadas desde aplicaciones Windows en DrvFs~~
 - Corrección: La compatibilidad de inotify con los cambios que se originan en aplicaciones Windows no está disponible en este momento.
- El enlace de sockets a `IPV6::<port n>` ahora admite `IPV6_V6ONLY` (GH 68, 157, 393, 460, 674, 740, 982, 996).
- Se ha implementado el comportamiento de WNOWAIT para la llamada del sistema waitid (GH 638)
- Compatibilidad con las opciones de socket de IP `IP_HDRINCL` e `IP_TTL`
- `read()` de longitud cero debe volver inmediatamente (GH 975)
- Controla correctamente los nombres de archivo y los prefijos de nombre de archivo que no incluyen un terminador NULL en un archivo .tar.
- Compatibilidad de epoll para `/dev/null`
- Corrección del origen de hora de `/dev/alarm`
- bash -c ahora puede redirigir a un archivo
- Correcciones de errores y mejoras adicionales

Resultados de LTP:

Número de pruebas superadas: 664

Número de pruebas no superadas (con error, omitidas, etc.): 264

Compatibilidad con llamadas del sistema

A continuación se muestra una lista de llamadas del sistema nuevas o mejoradas que tienen alguna implementación en WSL. Las llamadas del sistema de esta lista se admiten en al menos un escenario, pero puede que no se admitan todos los parámetros en este momento.

`chroot`

Compilación 14931

Para obtener información general de Windows sobre la compilación 14931, visita el [blog de Windows](#).

Fijo

- Debido a circunstancias más allá de nuestro control, no hay ninguna actualización en esta compilación para el subsistema de Windows para Linux. Las actualizaciones programadas regularmente se reanudarán en la próxima versión.

Compilación 14926

Para obtener información general de Windows sobre la compilación 14926, visita el [blog de Windows](#).

Fijo

- Ping ahora funciona en consolas que no tienen privilegios de administrador
- Ahora también se admite Ping6, sin privilegios de administrador.
- Compatibilidad de inotify con archivos modificados a través de WSL. (GH 216)
 - Marcas admitidas:
 - inotify_init1: LX_O_CLOEXEC, LX_O_NONBLOCK
 - Eventos de inotify_add_watch: LX_IN_ACCESS, LX_IN MODIFY, LX_IN_ATTRIB, LX_IN_CLOSE_WRITE, LX_IN_CLOSE_NOWRITE, LX_IN_OPEN, LX_IN_MOVED_FROM, LX_IN_MOVED_TO, LX_IN_CREATE, LX_IN_DELETE, LX_IN_DELETE_SELF, LX_IN_MOVE_SELF
 - Atributos de inotify_add_watch: LX_IN_DONT_FOLLOW, LX_IN_EXCL_UNLINK, LX_IN_MASK_ADD, LX_IN_ONESHOT, LX_IN_ONLYDIR
 - Salida de lectura: LX_IN_ISDIR, LX_IN_IGNORED
 - Problema conocido: La modificación de archivos desde aplicaciones de Windows no genera ningún evento
- El socket de Unix ahora es compatible con SCM_CREDENTIALS

Resultados de LTP:

Número de pruebas superadas: 651

Número de pruebas no superadas (con error, omitidas, etc.): 258

Compilación 14915

Para obtener información general de Windows sobre la compilación 14915, visita el [blog de Windows](#).

Fijo

- Socketpair para sockets de datagramas de Unix (GH 262)
- Compatibilidad con el socket de Unix para SO_REUSEADDR
- Compatibilidad con el socket de Unix para SO_BROADCAST (GH 568)
- Compatibilidad con el socket de Unix para SOCK_SEQPACKET (GH 758, 546)
- Adición de compatibilidad para socket de datagramas de Unix send, recv y shutdown
- Corrección de comprobación de errores debido a la validación de parámetros mmap no válida para direcciones no fijas. (GH 847)
- Compatibilidad para suspender o reanudar los estados del terminal
- Compatibilidad para TIOCPKT ioctl para desbloquear la utilidad de pantalla (GH 774)
 - Problema conocido: Las teclas de función no funcionan
- Se ha corregido una carrera en TimerFd que podía hacer que LxpTimerFdWorkerRoutine accediera a un miembro liberado "ReaderReady" (GH 814)
- Habilita la compatibilidad con llamadas del sistema reinicias para futex, poll y clock_nanosleep
- Se ha agregado compatibilidad con montaje de enlace
- Compatibilidad para dejar de compartir el espacio de nombres de montaje
 - Problema conocido: Al crear un nuevo espacio de nombres de montaje con `unshare(CLONE_NEWNS)`, el directorio de trabajo actual continuará señalando al espacio de nombres anterior
- Mejoras y correcciones de errores adicionales

Compilación 14905

Para obtener información general de Windows sobre la compilación 14905, visita el [blog de Windows](#).

Fijo

- Ahora se admiten llamadas del sistema reinicias (GH 349, GH 520)

- Los vínculos simbólicos a directorios que terminan en / ahora están operativos (GH 650)
- Se ha implementado RNDGETENTCNT ioctl para /dev/random
- Se han implementado los archivos /proc/[pid]/mounts, /proc/[pid]/mountinfo y /proc/[pid]/mountstats
- Correcciones de errores y mejoras adicionales

Compilación 14901

Primera compilación de Insider para la versión posterior a la Actualización de aniversario de Windows 10.

Para obtener información general de Windows sobre la compilación 14901, visita el [blog de Windows](#).

Fijo

- Se ha corregido un problema de barra diagonal final
 - Ahora funcionan los comandos como `$ mv a/c/ a/b/`
- Al instalar ahora se pregunta si la configuración regional de Ubuntu debe establecerse en la configuración regional de Windows
- Compatibilidad de procfs para la carpeta ns
- Se ha agregado mount y umount para los sistemas de archivos tmpfs, procfs y sysfs
- Corrección de la firma ABI mknod[at] de 32 bits
- Los sockets de Unix se movieron al modelo dispatch
- Se debe respetar el tamaño del búfer de recepción del socket de INET mediante setsockopt
- Implementa la marca de mensaje de recepción de socket de Unix `MSG_CMSG_CLOEXEC`
- Redirección de canalización stdin/stdout de proceso de Linux (GH 2)
 - Permite la canalización de comandos bash -c en CMD. Ejemplo: `>dir | bash -c "grep foo"`
- Bash ahora se puede instalar en sistemas con varios archivos de página (GH 538, 358)
- El tamaño predeterminado del búfer del socket de INET debe coincidir con el de la configuración predeterminada de Ubuntu
- Alinea las llamadas del sistema xattr con listxattr
- Solo se devuelven interfaces con una dirección IPv4 válida de SIOCGIFCONF

- Corrección de la acción predeterminada de la señal cuando se inserta por ptrace
- Implementa /proc/sys/vm/min_free_kbytes
- Usa valores de registro de contexto de equipo al restaurar el contexto en sigreturn
 - Esto resuelve el problema por el que java y javac se bloqueaban para algunos usuarios
- Implementa /proc/sys/kernel/hostname

Compatibilidad con llamadas del sistema

A continuación se muestra una lista de llamadas del sistema nuevas o mejoradas que tienen alguna implementación en WSL. Las llamadas del sistema de esta lista se admiten en al menos un escenario, pero puede que no se admitan todos los parámetros en este momento.

`waitid`

`epoll_pwait`

Compilación 14388 para la Actualización de aniversario de Windows 10

Para obtener información general de Windows sobre la compilación 14388, visita el [blog de Windows](#).

Fijo

- Correcciones para prepararse para la Actualización de aniversario de Windows 10 el 2/8
 - Puedes encontrar más información sobre WSL en la Actualización de aniversario en nuestro [blog](#)

Compilación 14376

Para obtener información general de Windows sobre la compilación 14376, visita el [blog de Windows](#).

Fijo

- Se quitaron algunas instancias en las que apt-get se bloquea (GH 493)
- Se ha corregido un problema por el que los montajes vacíos no se controlaban correctamente
- Se ha corregido un problema por el que ramdisks no se montaban correctamente
- Cambio en la aceptación de sockets de Unix para admitir marcas (GH parcial 451)
- Se ha corregido la pantalla azul relacionada con la red común
- Se ha corregido la pantalla azul al acceder a /proc/[pid]/task (GH 523)
- Se ha corregido un uso elevado de la CPU en algunos escenarios de pty (GH 488, 504)
- Correcciones de errores y mejoras adicionales

Compilación 14371

Para obtener información general de Windows sobre la compilación 14371, visita el [blog de Windows](#).

Fijo

- Se ha corregido la carrera de tiempo con SIGCHLD y wait() al usar ptrace
- Se han corregido comportamientos cuando las rutas de acceso tienen una / final (GH 432)
- Se ha corregido un problema con el error de cambio de nombre o desvinculación debido a identificadores abiertos a elementos secundarios
- Correcciones de errores y mejoras adicionales

Compilación 14366

Para obtener información general de Windows sobre la compilación 14366, visita el [blog de Windows](#).

Fijo

- Corrección en la creación de archivos mediante vínculos simbólicos
- Se ha agregado listxattr para Python (GH 385)
- Correcciones de errores y mejoras adicionales

Compatibilidad con llamadas del sistema

- A continuación se muestra una lista de llamadas del sistema nuevas o mejoradas que tienen alguna implementación en WSL. Las llamadas del sistema de esta lista se admiten en al menos un escenario, pero puede que no se admitan todos los parámetros en este momento.

listxattr

Compilación 14361

Para obtener información general de Windows sobre la compilación 14361, visita el [blog de Windows](#).

Fijo

- DrvFs ahora distingue entre mayúsculas y minúsculas cuando se ejecuta en Bash en Ubuntu en Windows.
 - Los usuarios pueden usar case.txt y CASE.TXT en sus unidades de /mnt/c
 - La distinción de mayúsculas y minúsculas solo se admite en Bash en Ubuntu en Windows. Fuera de Bash, NTFS notificará los archivos correctamente, pero puede producirse un comportamiento inesperado al interactuar con los archivos de Windows.
 - La raíz de cada volumen (es decir, /mnt/c) no distingue entre mayúsculas y minúsculas
 - Puedes encontrar más información sobre cómo controlar estos archivos en Windows [aquí](#).
- Compatibilidad con pty/tty considerablemente mejorada. Ahora se admiten aplicaciones como TMUX (GH 40)
- Se ha corregido el problema de instalación por el que las cuentas de usuario no siempre se creaban
- Se ha optimizado la estructura de argumentos de línea de comandos que permite una lista de argumentos extremadamente larga. (GH 153)
- Ahora puedes usar delete y chmod en archivos read_only de DrvFs
- Se han corregido algunas instancias en las que el terminal se bloquea en la desconexión (GH 43)
- chmod y chown ahora funcionan en dispositivos tty
- Permite la conexión a 0.0.0.0 y :: como localhost (GH 388)

- Sendmsg/recvmsg ahora controla una longitud de vector de E/S >1 (GH 376 parcial)
- Los usuarios ahora pueden dejar de participar en archivos de hosts generados automáticamente (GH 398)
- La configuración regional de Linux se ajusta automáticamente con la configuración regional de NT durante la instalación (GH 11)
- Se ha agregado el archivo /proc/sys/vm/swappiness (GH 306)
- strace ahora se cierra correctamente
- Permite que las canalizaciones se vuelvan a abrir a través de /proc/self/fd (GH 222)
- Oculta directorios en %LOCALAPPDATA%\lxss de DrvFs (GH 270)
- Mejor control de bash.exe ~. Ahora se admiten comandos como "bash ~ -c ls" (GH 467)
- Ahora, los sockets notifican las lecturas de epoll disponibles durante el cierre (GH 271)
- lxrun /uninstall funciona mejor para eliminar archivos y carpetas
- Se ha corregido ps -f (GH 246)
- Se ha mejorado la compatibilidad con aplicaciones X11, como xEmacs (GH 481)
- Se ha actualizado el tamaño inicial de la pila de subprocessos para que coincida con la configuración predeterminada de Ubuntu, y se notifica el tamaño correctamente a la llamada del sistema get_rlimit (GH 172, 258)
- Se ha mejorado la notificación de nombres de imagen de proceso PICO (por ejemplo, para auditoría)
- Se ha implementado el comando /proc/mountinfo para df
- Se ha corregido el código de error de vínculo simbólico para el nombre secundario . y .
- Correcciones de errores y mejoras adicionales

Compatibilidad con Llamadas del sistema

A continuación se muestra una lista de llamadas del sistema nuevas o mejoradas que tienen alguna implementación en WSL. Las llamadas del sistema de esta lista se admiten en al menos un escenario, pero puede que no se admitan todos los parámetros en este momento.

`GETTIMER`

`MKNODAT`

`RENAMEAT`

`SENDFILE`

`SENDFILE64`

Compilación 14352

Para obtener información general de Windows sobre la compilación 14352, visita el [blog de Windows](#).

Fijo

- Se ha corregido un problema en el que los archivos grandes no se descargaban o se creaban correctamente. Esto debe desbloquear npm y otros escenarios (GH 3, GH 313)
- Se han quitado algunas instancias en las que los sockets se bloquean
- Se han corregido algunos errores de ptrace
- Se ha corregido un problema con WSL que permite nombres de archivo de más de 255 caracteres
- Se ha mejorado la compatibilidad con caracteres distintos del inglés
- Agrega datos de zona horaria de Windows actuales y los establece como valores predeterminados
- Id. de dispositivo único para cada punto de montaje (corrección de JRE: GH 49)
- Se ha corregido el problema con rutas de acceso que contienen "." y ".."
- Se ha agregado compatibilidad con Fifo (GH 71)
- Se ha actualizado el formato de resolv.conf para que coincida con el formato nativo de Ubuntu
- Algunas limpiezas de procfs
- Se ha habilitado ping para consolas de administrador (GH 18)

Compatibilidad con llamadas del sistema

A continuación se muestra una lista de llamadas del sistema nuevas o mejoradas que tienen alguna implementación en WSL. Las llamadas del sistema de esta lista se admiten en al menos un escenario, pero puede que no se admitan todos los parámetros en este momento.

FALLOCATE

EXECVE

LGETXATTR

FGETXATTR

Compilación 14342

Para obtener información general de Windows sobre la compilación 14342, visita el [blog de Windows](#).

Puedes encontrar información sobre VolFs y DriveFs en el [blog de WSL](#).

Fijo

- Se ha corregido el problema de instalación cuando el usuario de Windows tenía caracteres Unicode en el nombre de usuario
- La solución alternativa de apt-get update udev en las preguntas más frecuentes se proporciona ahora de forma predeterminada en la primera ejecución
- Vínculos simbólicos habilitados en directorios DriveFs (`/mnt/<drive>`)
- Los vínculos simbólicos ahora funcionan entre DriveFs y VolFs
- Se ha solucionado el problema de análisis de ruta de acceso de nivel superior:ls // funcionará ahora como se espera
- La instalación de npm en DriveFs y las opciones -g ahora funcionan
- Se ha corregido un problema que impedía que el servidor PHP se iniciara
- Se han actualizado los valores de entorno predeterminados, como \$PATH para que coincidan con Ubuntu nativo
- Se ha agregado una tarea de mantenimiento semanal en Windows para actualizar la caché de paquetes apt
- Se ha corregido un problema con la validación de encabezado de ELF, WSL ahora es compatible con todas las opciones de Melkor
- El shell Zsh funciona
- Ahora se admiten los binarios de Go precompilados
- Indicar bash.exe en la primera ejecución ahora se localiza correctamente
- /proc/meminfo ahora devuelve la información correcta
- Ahora se admiten sockets en VFS
- /dev ahora se monta como tempfs
- Ahora se admite Fifo
- Los sistemas de varios núcleos ahora se muestran correctamente en /proc/cpuinfo
- Mejoras adicionales y mensajes de error que se descargan durante la primera ejecución
- Mejoras de llamadas del sistema y corrección de errores. Lista de llamadas del sistema admitidas a continuación.
- Correcciones de errores y mejoras adicionales

Problemas conocidos

- No se resuelve ".." correctamente en DriveFs en algunos casos

Compatibilidad con llamadas del sistema

A continuación se muestra una lista de llamadas del sistema nuevas o mejoradas que tienen alguna implementación en WSL. Las llamadas del sistema de esta lista se admiten en al menos un escenario, pero puede que no se admitan todos los parámetros en este momento.

FCHOWNAT

GETEUID

GETGID

GETRESUID

GETXATTR

PTRACE

SETGID

SETGROUPS

SETHOSTNAME

SETXATTR

Compilación 14332

Para obtener información general de Windows sobre la compilación 14332, visita el [blog de Windows](#).

Fijo

- Mejor generación de resolv.conf, incluida la priorización de entradas de DNS
- Problema con el movimiento de archivos y directorios entre unidades /mnt y no /mnt
- Los archivos tar ahora se pueden crear con vínculos simbólicos
- Se ha agregado el directorio predeterminado /run/lock en la creación de instancias
- Actualiza /dev/null para que devuelva la información de estadísticas correcta
- Errores adicionales al descargar durante la primera ejecución
- Mejoras de llamadas del sistema y corrección de errores. Lista de llamadas del sistema admitidas a continuación.
- Correcciones de errores y mejoras adicionales

Compatibilidad con llamadas del sistema

A continuación se muestra la nueva llamada del sistema que tiene alguna implementación en WSL. La llamada del sistema de esta lista se admite en al menos un escenario, pero puede que no se admita todos los parámetros en este momento.

READLINKAT

Compilación 14328

Para obtener información general de Windows sobre la compilación 14332, visita el [blog de Windows](#).

Nuevas características

- Ahora admite usuarios de Linux. Al instalar Bash en Ubuntu en Windows, se solicitará la creación de un usuario de Linux. Para más información, visita <https://aka.ms/wslusers>.
- El nombre de host ahora se establece en el nombre del equipo de Windows, ya no más en @localhost
- Para más información sobre la compilación 14328, visita: <https://aka.ms/wip14328>

Fijo

- Mejoras de vínculos simbólicos para archivos que no son `/mnt/<drive>`
 - La instalación de npm ahora funciona
 - jdk / jre ahora se puede instalar con las instrucciones que se encuentran [aquí](#).
 - Problema conocido: los vínculos simbólicos no funcionan para los montajes de Windows. La funcionalidad estará disponible en una compilación posterior
- top y htop ahora se muestran
- Mensajes de error adicionales para algunos errores de instalación
- Mejoras de llamadas del sistema y corrección de errores. Lista de llamadas del sistema admitidas a continuación.
- Correcciones de errores y mejoras adicionales

Compatibilidad con llamadas del sistema

A continuación hay una lista de llamadas del sistema que tienen alguna implementación en WSL. Las llamadas del sistema de esta lista se admiten en al menos un escenario, pero puede que no se admitan todos los parámetros en este momento.

ACCEPT
ACCEPT4
ACCESS
ALARM
ARCH_PRCTL
BIND
BRK
CAPGET
CAPSET
CHDIR
CHMOD
CHOWN
CLOCK_GETRES
CLOCK_GETTIME
CLOCK_NANOSLEEP
CLONE
CLOSE
CONNECT
CREAT
DUP
DUP2
DUP3
EPOLL_CREATE
EPOLL_CREATE1
EPOLL_CTL
EPOLL_WAIT
EVENTFD
EVENTFD2
EXECVE
EXIT
EXIT_GROUP
FACCESSAT
FADVISE64
FCHDIR

FCHMOD
FCHMODAT
FCHOWN
FCHOWNAT
FCNTL64
FDATASYNC
FLOCK
FORK
FSETXATTR
FSTAT64
FSTATAT64
FSTATFS64
FSYNC
FTRUNCATE
FTRUNCATE64
FUTEX
GETCPU
GETCWD
GETDENTS
GETDENTS64
GETEGID
GETEGID16
GETEUID
GETEUID16
GETGID
GETGID16
GETGROUPS
GETPEERNAME
GETPGID
GETPGRP
GETPID
GETPPID
GETPRIORITY
GETRESGID
GETRESGID16
GETRESUID
GETRESUID16
GETRLIMIT

GETRUSAGE
GETSID
GETSOCKNAME
GETSOCKOPT
GETTID
GETTIMEOFDAY
GETUID
GETUID16
GETXATTR
GET_ROBUST_LIST
GET_THREAD_AREA
INOTIFY_ADD_WATCH
INOTIFY_INIT
INOTIFY_RM_WATCH
IOCTL
IOPRIO_GET
IOPRIO_SET
KEYCTL
KILL
LCHOWN
LINK
LINKAT
LISTEN
LLSEEK
LSEEK
LSTAT64
MADVISE
MKDIR
MKDIRAT
MKNOD
MLOCK
MMAP
MMAP2
MOUNT
MPROTECT
MREMAP
MSYNC
MUNLOCK

MUNMAP
NANOSLEEP
NEWUNAME
OPEN
OPENAT
PAUSE
PERF_EVENT_OPEN
PERSONALITY
PIPE
PIPE2
POLL
PPOLL
PRCTL
PREAD64
PROCESS_VM_READV
PROCESS_VM_WRITEV
PSELECT6
PTRACE
PWRITE64
READ
READLINK
READV
REBOOT
RECV
RECVFROM
RECVMSG
RENAME
RMDIR
RT_SIGACTION
RT_SIGPENDING
RT_SIGPROCMASK
RT_SIGRETURN
RT_SIGSUSPEND
RT_SIGTIMEDWAIT
SCHED_GETAFFINITY
SCHED_GETPARAM
SCHED_GETSCHEDULER
SCHED_GET_PRIORITY_MAX

SCHED_GET_PRIORITY_MIN
SCHED_SETAFFINITY
SCHED_SETPARAM
SCHED_SETSCHEDULER
SCHED_YIELD
SELECT
SEND
SENDMMMSG
SENDMSG
SENDTO
SETDOMAINNAME
SETGID
SETGROUPS
SETHOSTNAME
SETITIMER
SETPGID
SETPRIORITY
SETREGID
SETRESGID
SETRESUID
SETREUID
SETRLIMIT
SETSID
SETSOCKOPT
SETTIMEOFDAY
SETUID
SETXATTR
SET_ROBUST_LIST
SET_THREAD_AREA
SET_TID_ADDRESS
SHUTDOWN
SIGACTION
SIGALTSTACK
SIGPENDING
SIGPROCMASK
SIGRETURN
SIGSUSPEND
SOCKET

SOCKETCALL
SOCKETPAIR
SPLICE
STAT64
STATFS64
SYMLINK
SYMLINKAT
SYNC
SYSINFO
TEE
TGKILL
TIME
TIMERFD_CREATE
TIMERFD_GETTIME
TIMERFD_SETTIME
TIMES
TKILL
TRUNCATE
TRUNCATE64
UMASK
UMOUNT
UMOUNT2
UNLINK
UNLINKAT
UNSHARE
UTIME
UTIMENSAT
UTIMES
VFORK
WAIT4
WAITPID
WRITE
WRITEV

Notas de la versión del Subsistema de Windows para el kernel de Linux

Artículo • 21/03/2023

Hemos agregado compatibilidad con distribuciones de WSL 2, [que usan un kernel de Linux completo](#). Este kernel de Linux es de código abierto y su código fuente está disponible en el repositorio [WSL2-Linux-Kernel](#). Este kernel de Linux se entrega al equipo a través de Microsoft Update, y sigue una programación de versiones independiente del Subsistema de Windows para Linux, que se entrega como parte de la imagen de Windows.

5.15.57.1

Fecha de lanzamiento: versión preliminar 02/08/2022

[Vínculo a la versión oficial de GitHub](#)

- Versión inicial de kernel de WSL2 basada en la serie de kernel v5.15
- Versión rolling-lts/wsl/5.15.57.1
- Actualización a la versión estable de kernel v5.15.57
- Habilitación de mitigaciones Retbleed en compilaciones de x86_64
- Habilitación de nftables y control de tráfico
- Habilitación del controlador VGEM
- Corrección de regresiones del sistema de archivos 9p desde el último kernel de WSL2 v5.10
- Habilitación de la compatibilidad con el dispositivo de reloj del protocolo de tiempo de precisión (PTP)
- Habilitación del módulo de seguridad Landlock Linux (LSM)
 - <https://landlock.io/>
- Habilitación del grupo de controles varios (CGroup)
 - <https://www.kernel.org/doc/html/v5.15/admin-guide/cgroup-v2.html#misc>
- Deshabilitación de la compatibilidad con el Sistema de archivos distribuido de Ceph

5.10.102.1

Fecha de lanzamiento: versión preliminar 09/05/2022

[Vínculo a la versión oficial de GitHub](#)

- Versión rolling-lts/wsl/5.10.102.1
- Actualización a la versión 5.10.102 superior de kernel estable
- Deshabilitación de BPF sin privilegios de manera predeterminada
- Se puede volver a habilitar al establecer el kernel.unprivileged_bpf_disabled sysctl en 0
- Se ha actualizado la versión Dxgkrnl a la 2216
- Corrección del acceso de matriz de límites para ioctl()
- Implementación de la espera de mensajes del bus de VM sincronizados como "eliminables" para permitir que se elimine un proceso que espera una llamada síncrona al host
- Vaciar el dispositivo para la finalización cuando el proceso se destruya a fin de evitar un interbloqueo cuando se elimine el proceso invitado

5.10.93.2

Fecha de lanzamiento: versión preliminar 08/02/2022

[Vínculo a la versión oficial de GitHub ↗](#)

- Versión rolling-lts/wsl/5.10.93.2
- Actualización a la versión 5.10.93 superior de kernel estable
- Habilitación de controladores de serie USB CH341 y CP210X
- Corrección de las instrucciones de compilación de README.md para incluir la dependencia de dwarves para pahole
- Se ha actualizado la versión Dxgkrnl a la 2111
- Se ha quitado el límite de asignaciones del sistema existentes y totales
- Vaciar correctamente el dispositivo para la finalización durante la limpieza del proceso
- Se ha corregido SPDX-License-Identifier para d3dkmthk.h

5.10.81.1

Fecha de lanzamiento: versión preliminar 01/02/2022

[Vínculo a la versión oficial de GitHub ↗](#)

- Versión rolling-lts/wsl/5.10.81.1
- Actualización a la versión 5.10.81 superior de kernel estable
- Unificación de las configuraciones de kernel mediante la habilitación de las opciones que faltan en arm64
- Habilitación de opciones ACPI no específicas del arco

- Habilitación de opciones relacionadas con RAID del asignador de dispositivos
- Habilitación de Btrfs
- Habilitación de la compresión LZO y ZSTD

5.10.74.3

Fecha de lanzamiento: versión preliminar 10/11/2021

[Vínculo a la versión oficial de GitHub ↗](#)

- Versión rolling-lts/wsl/5.10.74.3
- Actualización a la versión 5.10.74 superior de kernel estable
- Habilitación del formato de tipo BPF (CONFIG_DEBUG_INFO_BTF) para que lo usen las herramientas de eBPF (microsoft/WSL#7437)
- Se ha actualizado la versión Dxgkrnl a la 2110
- Habilitación de los marcos de archivos de sincronización y uso compartido de búferes (CONFIG_DMA_SHARED_BUFFER, CONFIG_SYNC_FILE) para el uso de Dxgkrnl
- Corrección del error de compilación de Dxgkrnl con versiones GCC anteriores a la 8.1 (microsoft/WSL#7558)

5.10.60.1

Fecha de lanzamiento: 02/11/2021 (versión preliminar 05/10/2021)

[Vínculo a la versión oficial de GitHub ↗](#)

- Versión rolling-lts/wsl/5.10.60.1
- Actualización a la versión 5.10.60 superior de kernel estable
- Habilitación de virtio-pmem con compatibilidad con direcciones relativas a PCI BAR
- Habilitación de la compatibilidad con vPCI en Hyper-V para arm64
- Habilitación de la compatibilidad con io_uring
- Habilitación de la compatibilidad con USB a través de IP
- Habilitación de la compatibilidad con bloqueos por subproceso paravirtualizados para x86_64
- Actualización del controlador dxgkrnl para recoger correcciones de errores y limpiezas de código
- Habilitación de la compatibilidad del cliente NFS para NFSv4.1
- Habilitación de las opciones de configuración del kernel USB para interactuar con un Arduino a través de USB

- Archivo README.md específico para WSL2

5.10.43.3

Fecha de lanzamiento: versión preliminar 12/07/2021

[Vínculo a la versión oficial de GitHub ↗](#)

- Versión rolling-lts/wsl/5.10.43.3
- Actualización a la versión 5.10.43 superior de kernel estable
- Controlador dxgkrnl mejorado
- Nueva revisión de arm64 de Linux en la serie Hyper-V (v9)
- Use siempre la interfaz de hiperllamadas de Hyper-V en invitados arm64 para admitir la ejecución en todas las versiones de Windows

5.10.16.3

Fecha de lanzamiento: 20/07/2021 (versión preliminar 16/04/2021)

[Vínculo a la versión oficial de GitHub ↗](#)

- Corrige [GH 5324 ↗](#).
- Agrega compatibilidad para discos cifrados con LUKS mediante wsl --mount

5.4.91

Fecha de lanzamiento: versión preliminar 22/02/2021

[Vínculo a la versión oficial de GitHub ↗](#)

5.4.72

Fecha de lanzamiento: 21/01/2021

[Vínculo a la versión oficial de GitHub ↗](#)

- Corrección de la configuración de la versión 5.4.72

5.4.51-microsoft-standard

Fecha de lanzamiento: Versión preliminar: 22/10/2020

[Vínculo a la versión oficial de GitHub ↗](#).

- Versión estable: 5.4.51

4.19.128-microsoft-standard

Fecha de lanzamiento: 15/09/2020

[Vínculo a la versión oficial de GitHub ↗](#).

- Versión estable de 4.19.128
- Corrección de los daños en la memoria IOCTL del controlador dxgkrnl

4.19.121-microsoft-standard

Fecha de lanzamiento: Versión preliminar

[Vínculo a la versión oficial de GitHub ↗](#).

- Controladores: hv: vmbus: hook up dxgkrnl
- Se ha agregado compatibilidad para el proceso de GPU

4.19.104-microsoft-standard

Fecha de lanzamiento: 09/06/2020

[Vínculo a la versión oficial de GitHub ↗](#).

- Actualización de la configuración de WSL para 4.19.104

4.19.84-microsoft-standard

Fecha de lanzamiento: 11/12/2019

[Vínculo a la versión oficial de GitHub ↗](#).

- Versión estable de 4.19.84

Notas de la versión del Subsistema de Windows para Linux en Microsoft Store

Artículo • 21/03/2023

Las notas de la versión de [WSL dentro de Microsoft Store](#) pueden encontrarse en la [página de versiones del repositorio de Github de Microsoft/WSL](#). Consulte esa lista para obtener las actualizaciones más recientes.

Problemas conocidos:

- El inicio del Subsistema de Windows para Linux desde la sesión cero no funciona actualmente (por ejemplo, desde una conexión SSH).