Fedora Environment on Windows Subsystem for Linux

Seth Jennings
Senior Software Engineer - Red Hat
sjenning@redhat.com

Agenda

- History of POSIX support in Windows
- Windows Subsystem for Linux
- Why?
- Syscall Emulation
- Filesystems
- Lifecycle
- Demo
- Q&A

History of POSIX support in Windows

- NT Kernel was written to support different user-mode subsystems
  - OS/2
  - Win32
  - POSIX
- Iterations of the POSIX subsystem
  - Microsoft POSIX subsystem
  - Windows Services for UNIX (SFU)
  - Subsystem for Unix-based Applications (SUA)/Intermix
  - Cygwin (non-Microsoft)
- All implemented in user-mode and required recompilation
  - Aimed at easing porting, not running ELF (native *nix) binaries

Windows Subsystem for Linux (WSL)

- Implemented primarily in kernel-mode
- Allows running native ELF binaries without recompilation
- When Linux (i.e. "Pico" processes) make syscalls, the NT kernel redirects the calls to a kernel-mode lxcore.sys driver that converts the Linux syscalls into NT kernel calls
  - These are very rarely 1:1

Why Fedora on Windows?

- Mindshare
- Very low barrier to entry exposure
- Developers using (voluntary or not) Windows
  - Targeting Linux in production
  - Using Linux native tools for development like git, ssh, rsync, etc

How syscalls work

- Linux uses System V x86_64 ABI calling convention
  - Store arguments in registers
  - Put syscall number in rax
  - Emit syscall instruction for ring transition to kernel-space
  - Kernel syscall handler saves off registers and calls function that handles syscall
  - Restore registers
  - Save return code in rax
  - Emit sysret for ring transition back to user-space

Linux Kernel ABI

- If you break it, Linus will curse at you in public and revert your change, no questions asked
- syscall arguments and numbers are compiled into ELF binaries
- Provides a nice consistent point of abstraction for people looking to run Linux ELF binaries but not the Linux kernel
- This is layer WSL targets for emulation
  - Illumos did the same thing with LX Branded Zones

Linux syscalls

- There are over 300
- Many have LOTS of combinations
- Edge cases where action might be undefined by the spec
  - Do as Linux kernel does
- No small task to emulate them all
- WSL emulates a subset of syscalls from 64-bit binaries
  - Unimplemented syscall or combination return error codes, typically fatal

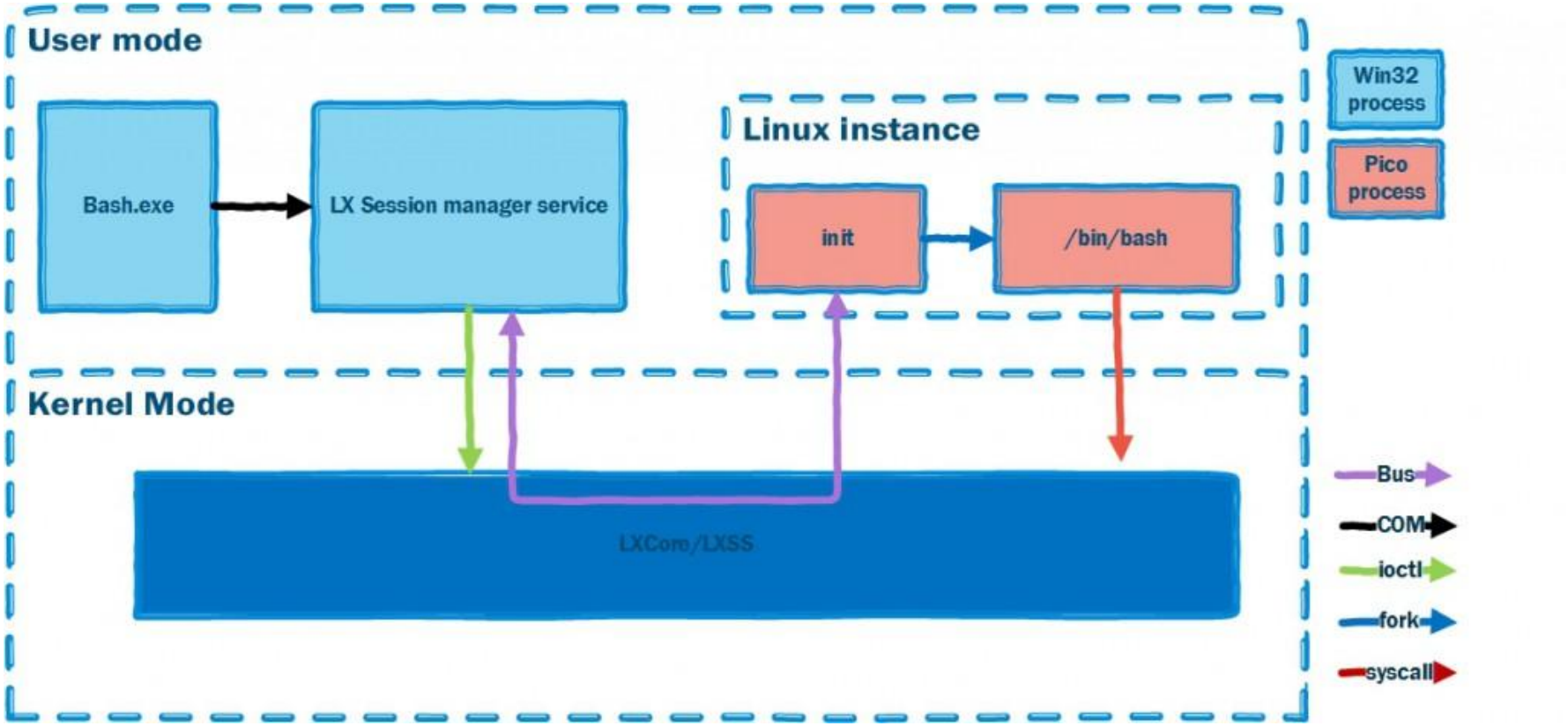
Filesystems

- lxcore.sys implements a VFS like that of the Linux kernel
- VolFs
  - Provides near full POSIX compatibility
    - permissions, symlinks, FIFO, sockets, etc
  - POSIX attributes are stored in NTFS extended attributes
  - Used for /, /root, and user home directory
    - /root and user home is preserved across distro app installations
- DrvFs
  - Provides interop with Windows
  - Mounts Windows drives by letter under /mnt (i.e. /mnt/c)
  - Limited POSIX compatibility
- TmpFs for in-memory filesystems (procfs, sysfs, etc)

Lifecycle

- fedora.exe makes COM call (think dbus) to user-space LX session manager service (lxss)
- lxss makes ioctl to NT kernel to create an LX instance
  - Basically a namespace for Linux/Pico processes
- /init and /etc/resolv.conf are injected
- /init does ??? (see figuring this out)
- /init forks /bin/bash
- No systemd
- When bash exits, init returns, LX instance is terminated

Lifecycle


User mode
Bash.exe
LX Session manager service
Linux instance
init
/bin/bash
Kernel Mode
LXCore/LXSS
Win32 process
Pico process
Bus
COM
ioctl
fork
syscall

Demo

Resources

- WSL Blog
  - https://blogs.msdn.microsoft.com/wsl/

Questions?

Backup slides

Pico Processes

- Win32 processes have a very particular structure that the NT kernel depends on as it routinely reaches up into user space to interact with applications
- Pico processes are black boxes to the NT kernel
- Linux processes are implemented as Pico processes
- The lxcore.sys kernel mode driver register to handle these processes

About Me

- Live in Austin, TX
- Started at IBM in 2007
  - AIX networking stack
  - Linux memory management
  - Authored zswap (Transparent Memory Compression)
- Joined Red Hat Kernel Team in 2013
  - Co-authored kpatch (Kernel Live Patching)
- Moved to work on Kubernetes and Openshift in 2016