Volatility

```
pipx install volatility3

pipx ensurepath
pipx completions
```

* Install symbols tables from github https://github.com/volatilityfoundation/volatility3

Windows Basics

https://volatility3.readthedocs.io/en/latest/volatility3.plugins.windows.html

```
vol -f <image> windows.info
vol -f <image> windows.pstree
windows.psscan
windows.pslist

vol.py -f “/path/to/file” -o “/path/to/dir” windows.dumpfiles --pid <PID>
vol.py -f “/path/to/file” -o “/path/to/dir” windows.memmap --dump --pid
<PID>
vol.py -f “/path/to/file” windows.handles --pid <PID>
vol.py -f “/path/to/file” windows.dlllist --pid <PID>
vol.py -f “/path/to/file” windows.cmdline

vol.py -f “/path/to/file” windows.netscan
vol.py -f “/path/to/file” windows.netstat


vol.py -f “/path/to/file” windows.registry.hivescan
vol.py -f “/path/to/file” windows.registry.hivelist

vol.py -f “/path/to/file” windows.registry.printkey
vol.py -f “/path/to/file” windows.registry.printkey --key
“Software\Microsoft\Windows\CurrentVersion”

vol.py -f “/path/to/file” windows.filescan

filedump

vol.py -f “/path/to/file” -o “/path/to/dir” windows.dumpfiles
vol.py -f “/path/to/file” -o “/path/to/dir” windows.dumpfiles --virtaddr
<offset>
vol.py -f “/path/to/file” -o “/path/to/dir” windows.dumpfiles --physaddr
<offset>
```

```
vol.py -f “/path/to/file” windows.malfind


vol.py -f “/path/to/file” windows.vadyarascan --yara-rules <string>
vol.py -f “/path/to/file” windows.vadyarascan --yara-file
“/path/to/file.yar”
vol.py -f “/path/to/file” yarascan.yarascan --yara-file “/path/to/file.yar”
```

Windows automation

volrun.sh

```
#!/bin/bash

# volrun version 1
# Credits to google gemini

# Function to display usage instructions
usage() {
    echo "Usage: $0 [-e] <path_to_memory_dump> <output_directory>"
    echo "  -e    Enable extended plugin set (shimcache, amcache,
userassist, etc.)"
    exit 1
}

# Parse optional flags
EXTENDED_MODE=false
while getopts "e" opt; do
    case ${opt} in
        e )
            EXTENDED_MODE=true
            ;;
        \? )
            usage
            ;;
    esac
done
shift $((OPTIND -1))

# Check if the mandatory positional arguments are provided
if [ "$#" -ne 2 ]; then
    usage
fi

# Assign positional arguments to variables
MEM_DUMP="$1"
OUTPUT_DIR="$2"

# Check if the memory dump file exists
```

```
if [ ! -f "$MEM_DUMP" ]; then
    echo "Error: Memory dump file '$MEM_DUMP' not found."
    exit 1
fi

# Create the output directory if it doesn't exist
mkdir -p "$OUTPUT_DIR"

# Base list of essential Windows plugins
plugins=(
    "windows.info"
    "windows.pslist"
    "windows.psscan"
    "windows.pstree"
    "windows.cmdline"
    "windows.dlllist"
    "windows.handles"
    "windows.netstat"
    "windows.netscan"
    "windows.malfind"
    "windows.registry.hivelist"
    "windows.registry.printkey"
    "windows.ssdt"
    "windows.driverscan"
)

# Extended list of plugins (deep dive / persistence / execution
artifacts)
extended_plugins=(
    "windows.registry.userassist"
    "windows.shimcache"
    "windows.amcache"
    "windows.poolscanner"
    "windows.vadinfo"
    "windows.modscan"
)

echo "================================================="
echo "Starting Volatility 3 analysis on: $MEM_DUMP"
echo "Outputs will be saved to: $OUTPUT_DIR"
echo "Total plugins to run: ${#plugins[@]}"
echo "================================================="




# Loop through each plugin and execute it
for plugin in "${plugins[@]}"; do
    # Generate a clean filename for the output (e.g.,
windows_registry_userassist.txt)
    safe_plugin_name=$(echo "$plugin" | tr '.' '_')
```

```
    output_file="$OUTPUT_DIR/${safe_plugin_name}.txt"

    echo "[*] Running $plugin..."

    # Execute volatility and redirect output to the text file
    vol -f "$MEM_DUMP" $plugin > "$output_file" 2>&1

    if [ $? -eq 0 ]; then
        echo "[+] Completed: Saved to $output_file"
    else
        echo "[-] Error or warning running $plugin. Check $output_file
for details."
    fi
    echo "-------------------------------------------------"
done

# If the -e flag was passed, merge the extended plugins into the main
array
if [ "$EXTENDED_MODE" = true ]; then
    echo "[+] Extended mode enabled. "

    for plugin in "${extended_plugins[@]}"; do
    safe_plugin_name=$(echo "extended_$plugin" | tr '.' '_')
    output_file="$OUTPUT_DIR/${safe_plugin_name}.txt"

    echo "[*] Running $plugin..."

    vol -f "$MEM_DUMP" $plugin > "$output_file" 2>&1

    if [ $? -eq 0 ]; then
        echo "[+] Completed: Saved to $output_file"
    else
        echo "[-] Error or warning running $plugin. Check $output_file
for details."
    fi
    echo "-------------------------------------------------"
done
fi


echo "Analysis complete. All results saved to '$OUTPUT_DIR'."
```