Proof of Concept (PoC)

(made under Digisuraksha Parhari Foundation)

This PoC is all about two tools i.e LoadOrder & Hex2dec, how they work and how are they helpful in different cases of Digital Forensics and Cyber Security.

A. Tool 1: LoadOrder

1. History:

Developed by **Mark Russinovich** as part of the **Sysinternals Suite**, LoadOrder was created to assist administrators and forensic analysts in examining the **Windows driver load order** — an important part of the system boot process.

2. Description:

LoadOrder is a lightweight Windows **Sysinternals** utility that displays the order in which device drivers are loaded during system startup. It categorizes drivers based on their start type: Boot, System, Auto, and Manual. Analysts can use LoadOrder to identify **early-loading kernel modules**, **locate unsigned drivers**, and **assess the legitimacy of third-party driver paths**. It's particularly useful for detecting hidden or malicious drivers embedded deep into the OS load chain.

3. What Is This Tool About?

LoadOrder reveals the sequence in which device drivers are initialized in Windows, which is crucial in identifying if **malware or rootkits are loaded stealthily** before the OS and AV tools can act.

4. Key Characteristics / Features:

- Lists drivers based on their **load order groups**
- Flags **boot/start/system/manual** types
- Can spot **unsigned or suspicious** drivers
- Requires **Admin privileges** to run
- **Read-only**, no changes made to system
- Lightweight executable, no installation needed

5. Types / Modules Available:

- **GUI version**
- **Command-line listing (via driverquery + sc.exe)**
- Works with **Sigcheck** & **VirusTotal** for deeper inspection

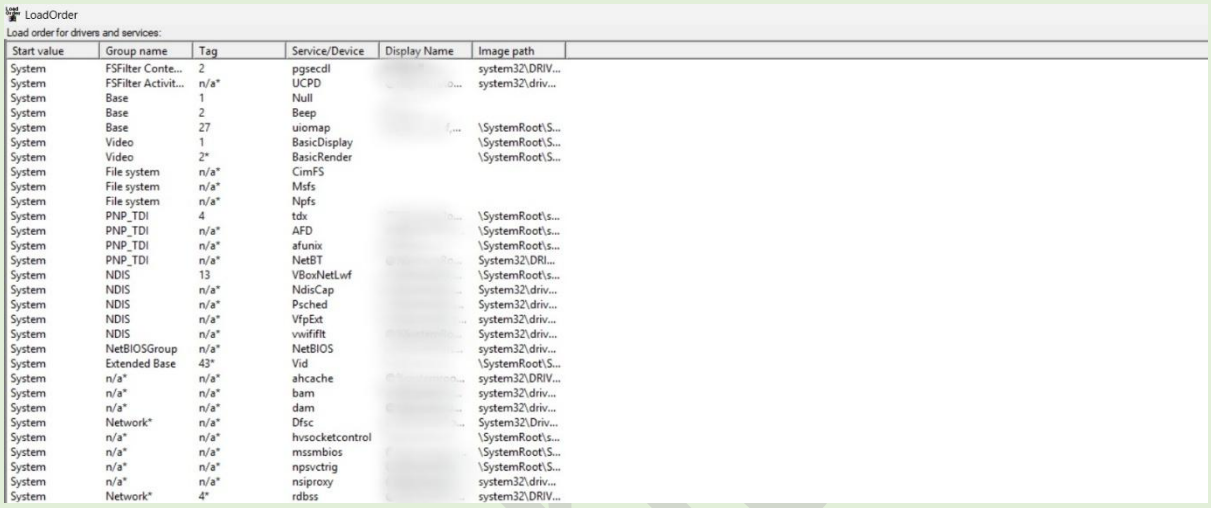
6. How Will This Tool Help?

- Identify **early-boot malware/rootkits**
- Detect **unauthorized drivers**
- Validate **driver integrity & signatures**
- Build a timeline of **driver loading**
- Assist in **incident response or forensic analysis**

7. Proof of Concept (PoC) Images:

(Personal Identifiers like usernames or full install paths have been blurred in these screenshots for privacy and clarity.)

Screenshot 1: System-Level Drivers



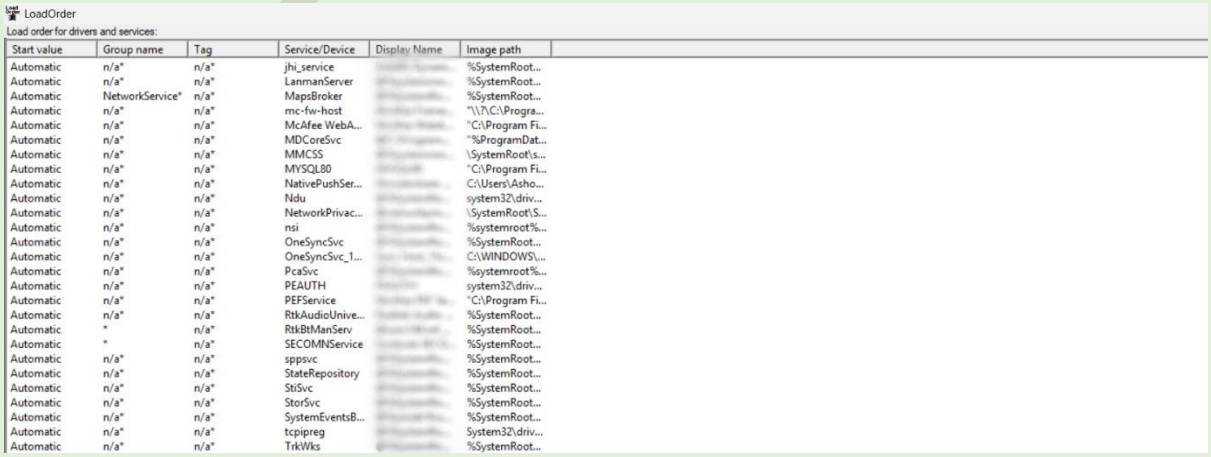
LoadOrder

Load order for drivers and services:

Start value	Group name	Tag	Service/Device	Display Name	Image path
System	FSFilter Conte...	2	pgsecdl		system32\DRIV...
System	FSFilter Activit...	n/a*	UCPD		system32\driv...
System	Base	1	Null		
System	Base	2	Beep		
System	Base	27	uiomap		\SystemRoot\S...
System	Video	1	BasicDisplay		\SystemRoot\S...
System	Video	2*	BasicRender		\SystemRoot\S...
System	File system	n/a*	CimFS		
System	File system	n/a*	Msfs		
System	File system	n/a*	Npfs		
System	PNP_TDI	4	tdx		\SystemRoot\s...
System	PNP_TDI	n/a*	AFD		\SystemRoot\s...
System	PNP_TDI	n/a*	afunix		\SystemRoot\s...
System	PNP_TDI	n/a*	NetBT		System32\DRI...
System	NDIS	13	VBoxNetLwf		\SystemRoot\s...
System	NDIS	n/a*	NdisCap		System32\driv...
System	NDIS	n/a*	Psched		System32\driv...
System	NDIS	n/a*	VfpExt		system32\driv...
System	NDIS	n/a*	vwififlt		System32\driv...
System	NetBIOSGroup	n/a*	NetBIOS		system32\driv...
System	Extended Base	43*	Vid		\SystemRoot\S...
System	n/a*	n/a*	ahcache		system32\DRIV...
System	n/a*	n/a*	bam		system32\driv...
System	n/a*	n/a*	dam		system32\driv...
System	Network*	n/a*	Dfsc		System32\Driv...
System	n/a*	n/a*	hvsocketcontrol		\SystemRoot\s...
System	n/a*	n/a*	mssmbios		\SystemRoot\S...
System	n/a*	n/a*	npsvctrig		\SystemRoot\S...
System	n/a*	n/a*	nsiproxy		system32\driv...
System	Network*	4*	rdbss		system32\DRIV...

LoadOrder lists drivers with **Start Value = System**. These include essential kernel-mode drivers such as **Beep**, **VBoxNetLwf**, and **NdisCap**, typically loaded early in the boot sequence by the OS or virtual machine environments.
All driver paths resolve to **%SystemRoot%\System32\drivers**, indicating they are standard and trusted components.

Screenshot 2: Automatically Started Drivers



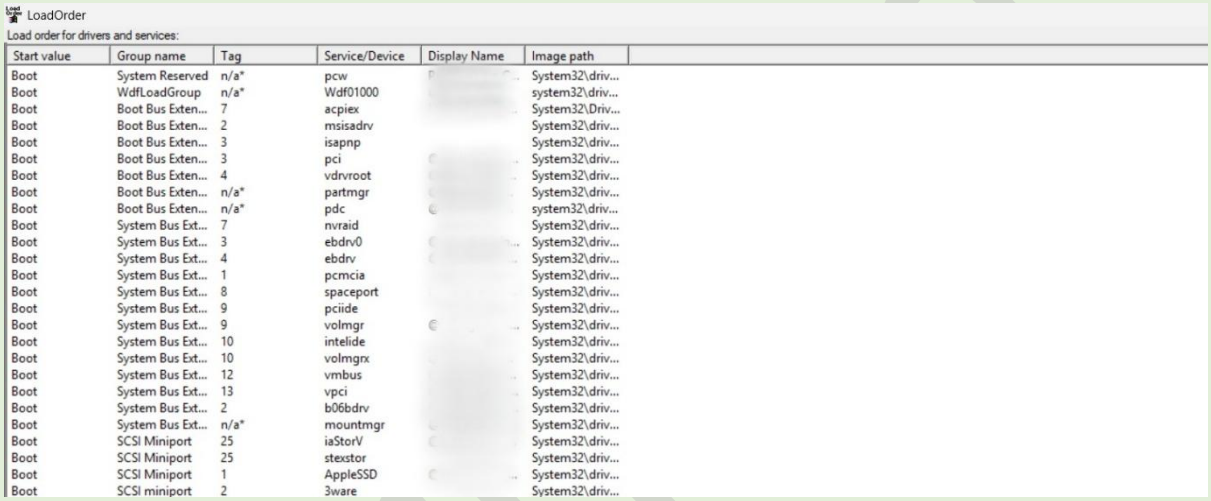
LoadOrder

Load order for drivers and services:

Start value	Group name	Tag	Service/Device	Display Name	Image path
Automatic	n/a*	n/a*	jhi_service		%SystemRoot...
Automatic	n/a*	n/a*	LanmanServer		%SystemRoot...
Automatic	NetworkService*	n/a*	MapsBroker		%SystemRoot...
Automatic	n/a*	n/a*	mc-fw-host		"\\?\C:\Progra...
Automatic	n/a*	n/a*	McAfee WebA...		"C:\Program Fi...
Automatic	n/a*	n/a*	MDCoreSvc		"%ProgramDat...
Automatic	n/a*	n/a*	MMCSS		\SystemRoot\s...
Automatic	n/a*	n/a*	MYSQL80		"C:\Program Fi...
Automatic	n/a*	n/a*	NativePushSer...		C:\Users\Asho...
Automatic	n/a*	n/a*	Ndu		system32\driv...
Automatic	n/a*	n/a*	NetworkPrivac...		\SystemRoot\S...
Automatic	n/a*	n/a*	nsi		%systemroot%...
Automatic	n/a*	n/a*	OneSyncSvc		%SystemRoot...
Automatic	n/a*	n/a*	OneSyncSvc_1...		C:\WINDOWS\...
Automatic	n/a*	n/a*	PcaSvc		%systemroot%...
Automatic	n/a*	n/a*	PEAUTH		system32\driv...
Automatic	n/a*	n/a*	PEFService		"C:\Program Fi...
Automatic	n/a*	n/a*	RtkAudioUnive...		%SystemRoot...
Automatic	*	n/a*	RtkBtManServ		%SystemRoot...
Automatic	*	n/a*	SECOMNService		%SystemRoot...
Automatic	n/a*	n/a*	sppsvc		%SystemRoot...
Automatic	n/a*	n/a*	StateRepository		%SystemRoot...
Automatic	n/a*	n/a*	StiSvc		%SystemRoot...
Automatic	n/a*	n/a*	StorSvc		%SystemRoot...
Automatic	n/a*	n/a*	SystemEventsB...		%SystemRoot...
Automatic	n/a*	n/a*	tcpipreg		System32\driv...
Automatic	n/a*	n/a*	TrkWks		%SystemRoot...

Drivers with **Start Value = Automatic**, including third-party services such as **McAfee WebAdvisor**, **MYSQL80**, and **Wondershare NativePushService**.
These entries are useful in spotting non-standard drivers, especially those with custom install paths **(e.g., C:\Program Files\...)** which may indicate third-party software or potentially suspicious installations.

🔹 Screenshot 3: Boot-Time Drivers


LoadOrder

Load order for drivers and services:

Start value	Group name	Tag	Service/Device	Display Name	Image path
Boot	System Reserved	n/a*	pcw		System32\driv...
Boot	WdfLoadGroup	n/a*	Wdf01000		system32\driv...
Boot	Boot Bus Exten...	7	acpiex		System32\Driv...
Boot	Boot Bus Exten...	2	msisadrv		System32\driv...
Boot	Boot Bus Exten...	3	isapnp		System32\driv...
Boot	Boot Bus Exten...	3	pci		System32\driv...
Boot	Boot Bus Exten...	4	vdrvroot		System32\driv...
Boot	Boot Bus Exten...	n/a*	partmgr		System32\driv...
Boot	Boot Bus Exten...	n/a*	pdc		system32\driv...
Boot	System Bus Ext...	7	nvraid		System32\driv...
Boot	System Bus Ext...	3	ebdrv0		System32\driv...
Boot	System Bus Ext...	4	ebdrv		System32\driv...
Boot	System Bus Ext...	1	pcmcia		System32\driv...
Boot	System Bus Ext...	8	spaceport		System32\driv...
Boot	System Bus Ext...	9	pciide		System32\driv...
Boot	System Bus Ext...	9	volmgr		System32\driv...
Boot	System Bus Ext...	10	intelide		System32\driv...
Boot	System Bus Ext...	10	volmgrx		System32\driv...
Boot	System Bus Ext...	12	vmbus		System32\driv...
Boot	System Bus Ext...	13	vpci		System32\driv...
Boot	System Bus Ext...	2	b06bdrv		System32\driv...
Boot	System Bus Ext...	n/a*	mountmgr		System32\driv...
Boot	SCSI Miniport	25	iaStorV		System32\driv...
Boot	SCSI Miniport	25	stexstor		System32\driv...
Boot	SCSI Miniport	1	AppleSSD		System32\driv...
Boot	SCSI miniport	2	3ware		System32\driv...

Boot-critical drivers such as **pci.sys**, **intelide.sys**, and **AppleSSD.sys**, loaded with **Start Value = Boot**, are essential for disk, chipset, or bus initialization.
Such entries are particularly relevant in detecting early-stage rootkits that may attempt to replace or hook into low-level disk or memory drivers.

🧪 Sample Suspicious Entry (Hypothetical PoC)

During analysis, a suspicious driver named **win32svcdrv.sys** was found loaded in the Boot load group from the unusual path:
C:\ProgramData\win32svcdrv.sys
Signature verification using **Sigcheck** confirmed the file was unsigned, and **VirusTotal** flagged it as a keylogger rootkit variant.

Final Statement:

This demonstrates LoadOrder's capability to expose hidden or malicious drivers masquerading as system components.

8. 🛠 How to Set Up & Run LoadOrder on a Windows System

Requirements:

- A Windows system (preferably a VM or test machine)
- Administrator access
- Internet connection (for download)

a. Step 1: Download the Tool

- Visit the official Microsoft Sysinternals page:
  🔗 https://learn.microsoft.com/en-us/sysinternals/downloads/loadorder
- Or use direct download:
  🔗 https://download.sysinternals.com/files/LoadOrder.zip
- Download the LoadOrder.zip file to your local system.

b. Step 2: Extract the Tool

- Right-click on LoadOrder.zip and choose Extract All...
- Navigate to the extracted folder — it contains a single file: LoadOrder.exe

c. Step 3: Run as Administrator

- Right-click on LoadOrder.exe
- Choose "Run as Administrator"
- If User Account Control (UAC) prompts you, click Yes

d. Step 4: View the Driver Load Order

- The GUI will open, displaying:
    - Driver name
    - Load order group
    - Load type (Boot/Auto/Manual/System)
    - Path to the .sys file
- You can scroll and inspect the order in which drivers are loaded.

e. Optional Step 5: Use with Sigcheck

- Download sigcheck.exe from:
  🔗 https://learn.microsoft.com/en-us/sysinternals/downloads/sigcheck
- Use command like:
- sigcheck C:\Windows\System32\drivers\example.sys

to verify digital signatures of specific drivers shown in LoadOrder.

9. Usage Scenarios & Examples

a. Spotting Non-Microsoft Drivers at Boot:

- Look for drivers in the Boot load group with suspicious names and non-standard paths.
- Example:
- LoadGroup: Boot
- Start: Boot
- ImagePath: C:\ProgramData\win32svcdrv.sys ❌
- DisplayName: win32svcdrv.sys

📌 Use tools like Sigcheck to confirm digital signatures. Unsigned + early load = 🚨 suspicious.

b. Identify Drivers Loading from User Space:

- Example:
- ImagePath: C:\Users\Admin\AppData\Roaming\driverX.sys
- This is uncommon. Kernel drivers should never load from user folders.

c. Investigate Driver Load Priority:

- Use Tag Numbers to analyze load order priority within a group.
    - Lower Tag = Higher priority (e.g., Tag 1 loads before Tag 3).
    - Drivers with identical LoadGroup can be sequenced using this tag.

d. Offline Registry-Based Forensics:

- From compromised systems, extract the following registry:
- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services
- Use LoadOrder in forensic VMs to analyze the exported hive (requires advanced setup).

e. Detect Rootkits and Kernel Hooks:

- Use LoadOrder output + memory forensics (like Volatility) to detect if a listed driver is actually not present in memory (a stealthy red flag).

10. 15-Liner Summary:

1. Lists Windows driver load order
2. Flags driver start types (boot/system/manual)
3. Detects unsigned drivers
4. Works without installation
5. Admin access required
6. No write access to system files
7. Lightweight and portable
8. Useful in forensic boot analysis
9. Works with other tools like Sigcheck
10. Can reveal hidden rootkits
11. Supports analysis of multiple OS versions
12. Helps during post-compromise analysis
13. Maps drivers to known-good baselines

14. Ideal for IR & SOC analysts
15. GUI makes analysis faster and easier

11. Time to Use / Best Case Scenarios:

- During **boot-time compromise detection**
- When suspecting **persistent malware**
- In **rootkit investigations**
- During **forensic boot disk analysis**
- When driver conflicts or BSODs occur

12. When to Use During Investigation:

- **In the early triage of infected systems**
- **After evidence of driver tampering**
- **During a post-breach autopsy**
- **When detecting kernel-level persistence**
- **In threat actor driver-based malware cases**

13. Best Person to Use & Required Skills:

- **Best User**: Digital Forensics Investigator / IR Analyst
- **Skills Needed**:
    - Understanding of Windows boot process

- Familiarity with .sys drivers
- Basic OSINT to verify driver legitimacy
- Comfort with Sysinternals tools

14. Flaws / Suggestions to Improve:

- Lacks CLI version for automation
- Doesn’t provide signature verification by default
- No export to CSV/JSON
- Could add a **"flag suspicious"** feature
- UI not updated in years
- No direct integration with AV threat intelligence

15. Good About the Tool:

- Fast and lightweight
- Works out of the box
- Provides critical insights into boot drivers
- Helps uncover stealthy rootkits
- Ideal companion tool in IR kit

B. Tool Name: Hex2dec

1. History

Hex2dec is a utility originally developed by **Mark Russinovich** and later maintained as part of the **Sysinternals Suite**, now under **Microsoft**. Designed as a command-line tool to aid developers and security analysts, it quickly converts **hexadecimal numbers to decimal (and vice versa)** for debugging, reverse engineering, and digital forensics tasks.

2. Description

Hex2dec is a tiny command-line conversion tool from the **Microsoft Sysinternals Suite**. It allows users to quickly and accurately convert numbers between hexadecimal and decimal formats — crucial when dealing with **raw memory addresses, registry dumps, PE file offsets, or encoded hex strings in forensic logs**.

3. What Is This Tool About?

It's a bi-directional number converter between hexadecimal and decimal systems, used extensively in reverse engineering, forensics, memory analysis, and malware investigations.

4. Key Characteristics / Features

- Hex to Decimal and Decimal to Hex conversions

- Standalone CLI binary (hex2dec.exe)
- Part of Microsoft Sysinternals tools
- Portable (no installation needed)
- Minimal UI – clean terminal output
- Accepts both interactive input and command-line arguments
- High-precision integer support
- Lightweight (under 200KB)
- Perfect for scripting or quick conversions
- Reliable output without fluff
- Compatible with all modern Windows versions
- No dependencies
- Secure (digitally signed by Microsoft)
- Integrates into incident response workflows
- Openly accessible for free

5. Types / Modules Available

- hex2dec.exe
- hex2dec64.exe
- hex2dec64a.exe

6. How Will This Tool Help?

- Translates hex-encoded memory addresses
- Interprets malware payloads or registry keys
- Helps map PE file structures (hex offsets)
- Decodes process dump values

- Speeds up manual binary analysis
- Avoids manual conversion errors
- Assists in creating readable logs or documentation

7. Proof of Concept (PoC) – Installation and Usage Guide

Step-by-Step Installation:

1. Download from Sysinternals (Microsoft):
   Link: https://learn.microsoft.com/en-us/sysinternals/downloads/hex2dec
2. Extract:
   The file will be in .zip format. Extract it using Windows Explorer or any zip extractor.
3. Usage:
   No installation needed. Just double-click or run from Command Prompt:
4. hex2dec
5. (Optional) Add to PATH:
   Copy hex2dec.exe to C:\Windows\System32 for global access via terminal.

8. Usage Scenarios & Examples

a. Convert Hexadecimal to Decimal:

> hex2dec 0x7A69 (type in cmd prompt)

Output:

Hex: 0x7A69

Dec: 31337

b. Convert Decimal to Hexadecimal:

> hex2dec 1337 (type in cmd prompt)

Output:

Dec: 1337

Hex: 0x539

c. Use in Malware Analysis:

Malware may use encoded registry paths like 0x80000002. Use Hex2dec to decode this:

> hex2dec 0x80000002 (type in cmd prompt)

Now you know it's referencing HKEY_LOCAL_MACHINE.

d. Reverse Engineering PE Headers:

While analyzing PE file headers, hex offsets like 0x3C point to e_lfanew. Convert to decimal for offset indexing in hex editors:

> hex2dec 0x3C (type in cmd prompt)

Output:

Hex: 0x3C

Dec: 60

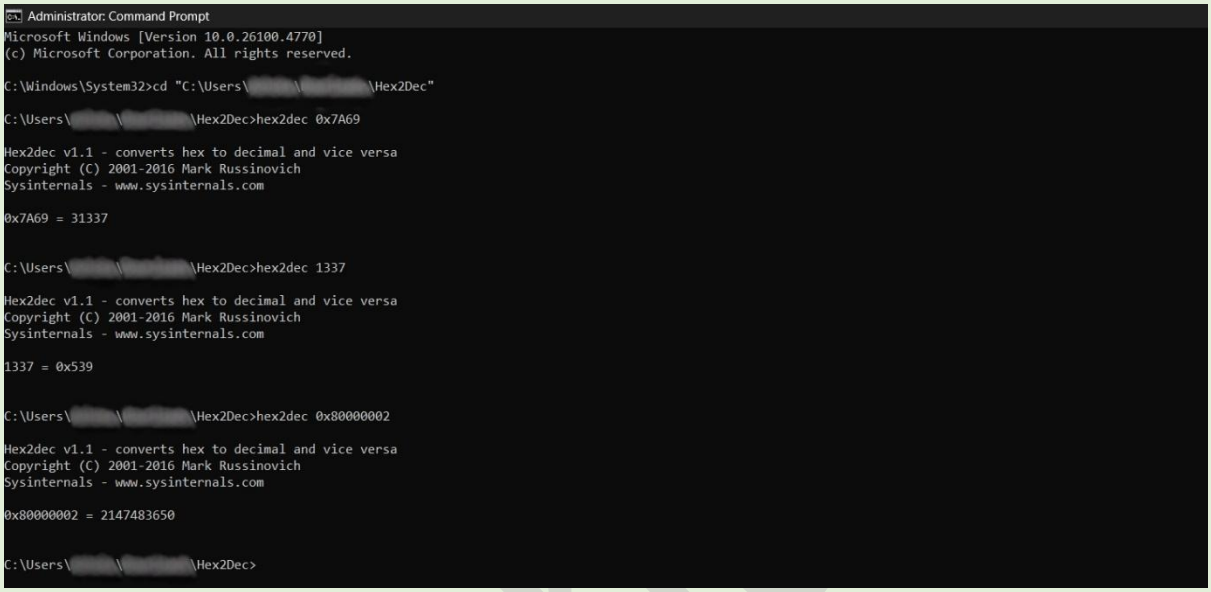
e. Decode File Metadata:

File systems or binary logs may log timestamps in hex (like 0x5FD5A3D2). Use:

hex2dec 0x5FD5A3D2

Then use the decimal value in a timestamp converter or Unix epoch parser.

Screenshot:



9. 15-Liner Summary:

1. Microsoft Sysinternals tool
2. Converts hex <–> decimal
3. CLI only, no GUI
4. Extremely lightweight
5. Instant output
6. No install required
7. Works on all Windows systems
8. Useful for memory & registry decoding
9. Free and secure
10. Accepts command-line arguments

11. Helps in PE structure decoding
12. Reduces manual conversion errors
13. Integrates in analyst workflow
14. Trusted by reverse engineers
15. Recommended for malware & OSINT analysis

10. When to Use This Tool

- **Binary file inspection**
- **Malware payload decoding**
- **Registry investigation**
- **Memory address tracing**
- **Reverse engineering tasks**
- **Parsing encoded threat actor logs**

11. Best Person to Use This Tool & Required Skills

Best Fit For:

- Security Researchers
- Malware Analysts
- Memory Forensics Experts
- Low-level Debuggers

Skills Needed:

- Basic knowledge of hex/dec number systems
- Understanding of file structures or memory dumps
- Familiarity with Command Line Interface

12. Flaws / Suggested Improvements

- No support for batch conversion
- Lacks GUI (not beginner-friendly)
- No conversion for binary or octal
- Can't handle floating-point formats
- Lacks direct clipboard or export support
- No timestamp interpretation mode
- No logging or scripting API

13. Why This Tool Is Good

- Instant results
- Completely portable
- No bloat or overhead
- Works in air-gapped systems
- Made and signed by Microsoft
- Fully trusted in enterprise environments
- Streamlines daily analyst workflows

-by **Sarthaka Subhankara Singh**

Serial no : 401
Intern ID : 438
Tool ID : 268269
made under **Digisuraksha Parhari Foundation**