4. Windows Endpoint Security

I. Endpoint Security Controls

Types of Endpoints

- Workstations
  - Desktops and laptops
- Mobile Devices
  - Smartphone and tablets
- Servers
  - Email, web, database, file servers
- IoT Devices
  - Smart printers, cameras, appliances
  - OT, ICS, SCADA
- Networking Equipment
  - Routers, switches, firewalls

Security Controls

- Antivirus / Antimalware
  - Scans files and activities
  - Match patterns and signatures
- Endpoint Detection and Response (EDR)
  - Real-time monitoring and response
  - Agent-based deployment
  - Monitor process, file, registry, network activity
- Extended Detection and Response (XDR)
  - Integration of multiple security controls and telemetry
  - Runbooks and automated response to routine threats
- Data Loss Prevention
  - Protect sensitive data at rest, transit, and in processing
  - Access controls, data masking, prevention
- User and Entity Behavior Analytics (UBA)

    - Monitor user behavior patterns
    - Detection deviations from historic and contextual baseline
- HIDS / HIPS
    - Host-based Intrusion Detection System (HIDS)
    - Host-based Intrusion Prevention System (HIPS)
- Host-based Firewall
    - Controls incoming and outgoing traffic on a host

Monitoring

- Process Execution
    - Running processes
    - Executable files, PIDs, command-line arguments
    - Parent-child process hierarchy
- File System Changes
    - Creation, modification, deletion
    - File Integrity Monitoring (FIM)
- Network Connections
    - Traffic connections initiated from the endpoint
    - Associated processes and executables
- Registry Modifications
    - Monitor registry keys and values
    - Detect backdoors, persistence

II. Windows Network Analysis

`net view` - Displays list of resources shared on a computer

```
net view \\127.0.0.1
```

`net share` - Shares a network drive

```
net share Exfil=C:\Users\user\Downloads\exfil
```

`net session` - Displays all inbound connections coming into the system

`net use` - Map network drive

```
net use X: \\127.0.0.1\Exfil
```

`netstat -anob` - net connections on machine

III. Windows Process Analysis

- Processes
  - Running instances of programs and applications
    - Partitioned set of system resources (CPU, memory, I/O)
  - System (Windows) Processes
    - OS core functions
    - System, smss.exe, csrss.exe
  - User (Application) Processes
    - Initiated by users
    - chrome.exe, notepad.exe, minesweeper.exe
  - Service (Background) Processes
    - Background functions
    - Windows Update, Print Spooler, lsass.exe

`tasklist` - Running processes. Shows PID, Session Name etc

```
tasklist /FI "PID eq 2088" /M
tasklist /FI "IMAGENAME eq notmalware.exe"
```

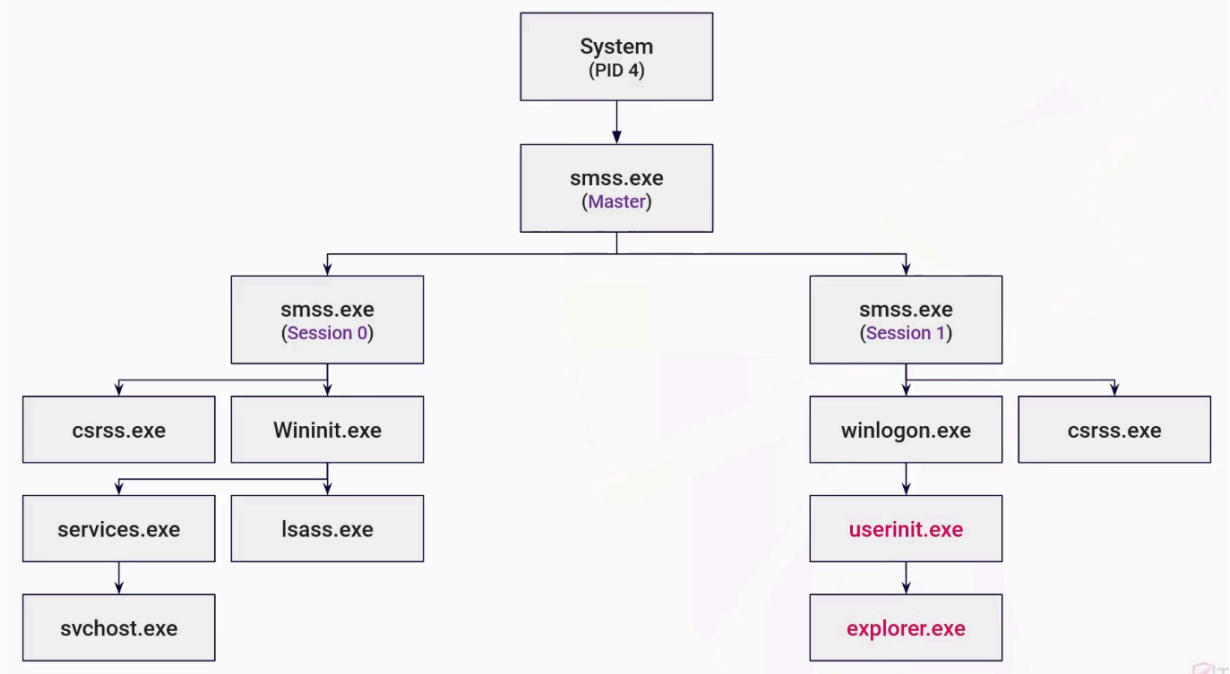
- `/M` : Show associated DLLs

```
wmic -
```

```
wmic process where processid=<ID number> get name, parentprocessid, processid
```

IV. Windows Core Processes

Process Tree



Task Manager

View and manage running processes.
Run as Administrator
Right click to add more columns

- Type
- Publisher
- PID
- Command line

Process Explorer

https://learn.microsoft.com/en-us/sysinternals/downloads/process-explorer

Run correct version based on system architecture
Run as Administrator
Output is color coded by type

Options

- Save
- Refresh Data
- Show/Hide lower pane

Right Click a Process

- Kill Process
- Kill Process Tree
- Restart
- Suspend
- Create Memory Dump
- Check VirusTotal.com
- Properties

Properties:

- Full executable path
- Command Line
- Directory
- Auto-start
- Parent Process
- User
- TCP/IP Network Info
-

System Process

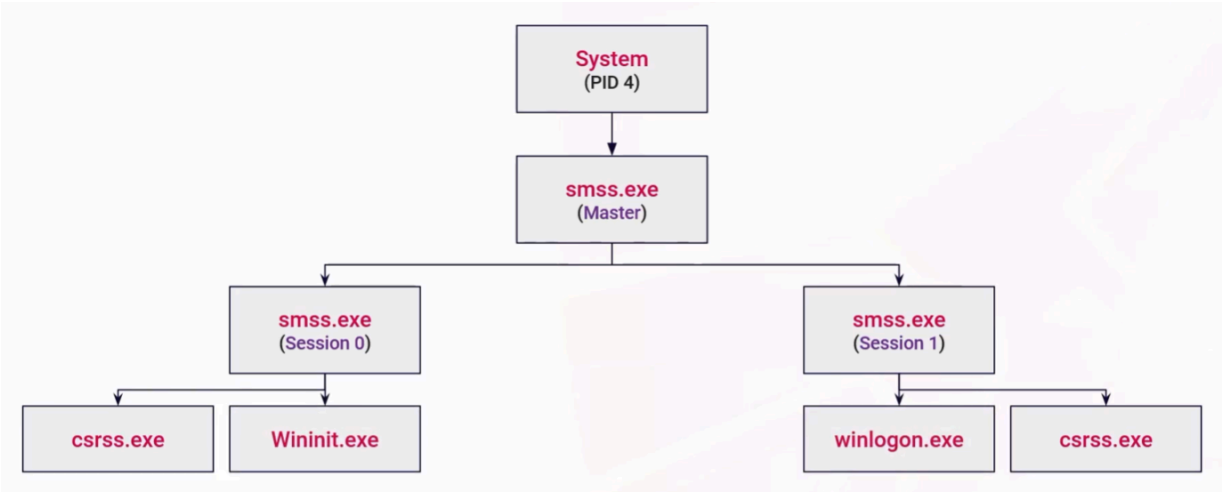
- Manages CPU, memory, disk
- Device drivers, hardware, process scheduling

Image Path:	None or `C:\Windows\system32\ntoskrnl.exe`
Process ID:	4
Parent Process:	None
Number of Instances:	1
User Account:	Local System
Start Time:	At Boot

Session Manager Subsystem - smss.exe

- Windows Session Manager
- Initiates and Manages user sessions
- Launches child processes: `wininit.exe` and `csrss.exe`

Image Path:	`%SystemRoot%\System32\sms.exe`
Parent Process:	System (4)
Number of Instances:	1 master, 1 child instance per session (children self-terminate)
User Account:	Local System
Start Time:	Within seconds of boot



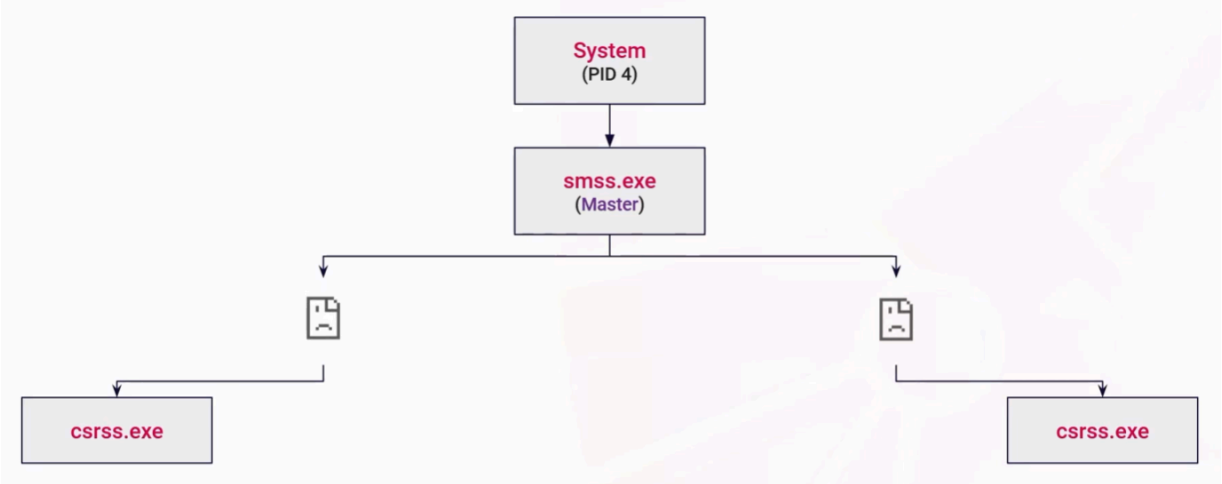
After creating their child processes, Session 0 and 1 will self-terminate.

Client/Server Runtime Subsystem - csrss.exe

- Manages console windows
- Imports DLLs for Windows API
- GUI tasks around shutdown

Image Path:	`%SystemRoot%\System32\csrss.exe`
Parent Process:	smss.exe (orphan process)
Number of Instances:	Two or more
User Account:	Local System
Start Time:	Within seconds of boot (first two instances)

Example of Orphan Process:



Windows Initialization - wininit.exe

- Initialize all essential services
- Session 0
- Spawns child processes (services.exe, lsass.exe)

Image Path:	`%SystemRoot%\System32\wininit.exe`
Parent Process:	smss.exe (orphan process)
Number of Instances:	1
User Account:	Local System
Start Time:	Within seconds of boot

Service Control Manager - services.exe

- Service Control Manager (SCM)
- Starts, stops, interacts with services
- Sets the `LastKnownGood CurrentControlSet` registry value

Image Path:	`%SystemRoot%\System32\services.exe`
Parent Process:	wininit.exe
Number of Instances:	1
User Account:	Local System
Start Time:	Within seconds of boot

Service Host - svchost.exe

- Hosts and manages Windows services
- Used to run service DLLs
- Runs with the -k parameter to differentiate instances/services (no -k could be suspicious)
- Common place to hide malware

Image Path:	`%SystemRoot%\System32\svchost.exe`
Parent Process:	services.exe
Number of Instances:	Many (typically 10+)
User Account:	Local System, Network Service, Local Service, or a user account
Start Time:	Within seconds of boot or whenever services start

Local Security Authority Subsystem Service - lsass.exe

- Authenticates users
- Implements local security policies
- Writes events to security event log

Image Path:	`%SystemRoot%\System32\lsass.exe`
Parent Process:	wininit.exe
Number of Instances:	1
User Account:	Local System
Start Time:	Within seconds of boot

Windows Logon - winlogon.exe

- Manages login and logout procedures
- Loads user profiles (NTUSER.DAT)
- Responds to the Secure Attention Sequence

Image Path:	`%SystemRoot%\System32\winlogon.exe`
Parent Process:	sms.exe (orphan process)
Number of Instances:	1 or more
User Account:	Local System
Start Time:	Within seconds of boot (Session 1)

Windows Explorer - explorer.exe

- Provides GUI for files, folders and system settings
- Manages the taskbar, Start Menu, and desktop
- Responsible for overall desktop environment

Image Path:	`%SystemRoot%\explorer.exe`
Parent Process:	userinit.exe (orphan process)
Number of Instances:	1 or more
User Account:	Logged-in User Account
Start Time:	When interactive user sessions begin

Process Investigation

- Parent Process
  - Is this the expected process hierarchy?
- Child Process
  - Is this the expected hierarchy?
- Command Line Arguments
  - `svchost` with no `-k` could be suspicious
- Process Names
  - Typos, lookalikes, copies
- User Account
  - Is this process running from the expected user account?
- Image Path
  - Is this process running the expected executable?

SANS Hunt Evil Cheat Sheet

https://sansorg.egnyte.com/dl/WFdH1hHnQI

SANS DFIR
DIGITAL FORENSICS & INCIDENT RESPONSE

Hunt Evil
POSTER

dfir.sans.org

$25.00
DFPS_FOR508_v4.11_0424
Poster was created by Rob Lee and Mike Pilkington
with support of the SANS DFIR Faculty

SANS DFIR CURRICULUM

SANSForensics | @SANSForensics | dfir.to/DFIRCast | dfir.to/LinkedIn

DIGITAL FORENSICS

- FOR498 Digital Acquisition and Rapid Triage — GBFA
- FOR500 Windows Forensic Analysis — GCFE
- FOR518 Mac and iOS Forensic Analysis & Incident Response — GIME
- FOR585 Smartphone Forensic Analysis In-Depth — GASF

INCIDENT RESPONSE & THREAT HUNTING

- FOR508 Advanced Incident Response, Threat Hunting & Digital Forensics — GCFA
- FOR509 Enterprise Cloud Forensics & Incident Response — GCFR
- FOR528 Ransomware and Cyber Extortion
- FOR572 Advanced Network Forensics: Threat Hunting, Analysis & Incident Response — GNFA
- FOR577 LINUX Incident Response and Threat Hunting
- FOR578 Cyber Threat Intelligence — GCTI
- FOR589 Cybercrime Intelligence
- FOR608 Enterprise-Class Incident Response & Threat Hunting — GEIR
- FOR610 REM: Malware Analysis Tools & Techniques — GREM
- FOR710 Reverse-Engineering Malware: Advanced Code Analysis
- SEC504 Hacker Tools, Techniques & Incident Handling — GCIH

Find Evil – Know Normal

Knowing what's normal on a Windows host helps cut through the noise to quickly locate potential malware.
Use the information below as a reference to know what's normal in Windows and to focus your attention on the outliers.

System

Image Path: N/A for `System.exe` – Not generated from an executable image
Parent Process: None
Number of Instances: One
User Account: Local System
Start Time: At boot time
Description: The `System` process is responsible for most kernel-mode threads. Modules run under `System` are primarily drivers (.sys files), but also include several important DLLs as well as the kernel executable, `ntoskrnl.exe`.

smss.exe

Image Path: `%SystemRoot%\System32\smss.exe`
Parent Process: System
Number of Instances: One master instance and another child instance per session. Children exit after creating their session.
User Account: Local System
Start Time: Within seconds of boot time for the master instance
Description: The Session Manager process is responsible for creating new sessions. The first instance creates a child instance for each new session. Once the child instance initializes the new session by starting the Windows subsystem (`csrss.exe`) and `wininit.exe` for Session 0 or `winlogon.exe` for Session 1 and higher, the child instance exits.

wininit.exe

Image Path: `%SystemRoot%\System32\wininit.exe`
Parent Process: Created by an instance of `smss.exe` that exits, typically appearing as an orphan process.
Number of Instances: One
User Account: Local System
Start Time: Within seconds of boot time
Description: `wininit.exe` starts key background processes within Session 0. It starts the Service Control Manager (`services.exe`), the Local Security Authority process (`lsass.exe`), and `lsaiso.exe` for systems with Credential Guard enabled. Note that prior to Windows 10, the Local Session Manager process (`lsm.exe`) was also started by wininit.exe. As of Windows 10, that functionality has moved to a service DLL (lsm.dll) hosted by `svchost.exe`.

RuntimeBroker.exe

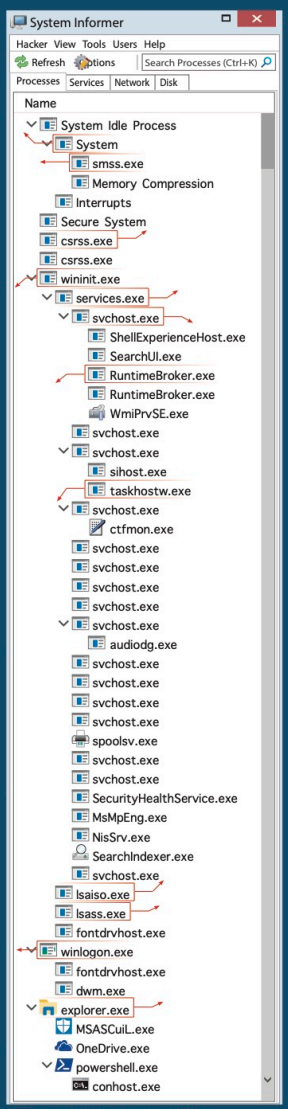
Image Path: `%SystemRoot%\System32\RuntimeBroker.exe`
Parent Process: `svchost.exe`
Number of Instances: One or more
User Account: Typically the logged-on user(s)
Start Time: Start times vary greatly
Description: `RuntimeBroker.exe` acts as a proxy between the constrained Universal Windows Platform (UWP) apps (formerly called Modern or Metro apps) and the full Windows API. UWP apps have limited capability to interface with hardware and the file system. Broker processes such as `RuntimeBroker.exe` are therefore used to provide the necessary level of access for UWP apps. Generally, there will be one `RuntimeBroker.exe` for each UWP app. For example, starting `Calculator.exe` will cause a corresponding `RuntimeBroker.exe` process to initiate.

taskhostw.exe

Image Path: `%SystemRoot%\System32\taskhostw.exe`
Parent Process: `svchost.exe`
Number of Instances: One or more `taskhostw.exe` processes are normal.
User Account: Task processes can be owned by logged-on users and/or by local service accounts.
Start Time: Start times vary greatly
Description: The generic host process for Windows Scheduled Tasks. Upon initialization, `taskhostw.exe` runs a continuous loop listening for trigger events. Example trigger events that can initiate a task include a defined time schedule, user logon, system startup, Idle CPU time, a Windows log event, or workstation lock/unlock. There are more than 200 tasks pre-configured on a default installation of Windows 11 Enterprise (though not all are enabled). All executable files (DLLs & EXEs) used by the default Windows 10+ scheduled tasks are signed by Microsoft. This process replaced the older `taskhost.exe` and `taskhostex.exe` processes.

winlogon.exe

Image Path: `%SystemRoot%\System32\winlogon.exe`
Parent Process: Created by an instance of `smss.exe` that exits, typically appearing as an orphan process.
Number of Instances: One or more
User Account: Local System
Start Time: Within seconds of boot time for the first instance (for Session 1). Start times for additional instances occur as new sessions are created, typically through Remote Desktop or Fast User Switching logons.
Description: Winlogon handles interactive user logons and logoffs. It launches `LogonUI.exe`, which uses a credential provider to gather credentials from the user, ultimately passing the credentials to `lsass.exe` for validation. Once the user is authenticated, `winlogon.exe` loads the user's `NTUSER.DAT` into `HKCU` and starts the user's shell (usually `explorer.exe`) via `userinit.exe`. `dwm.exe` and `fontdrvhost.exe` are common children of this process and are responsible for display management.



Process listing from Windows 10 Enterprise

csrss.exe

Image Path: `%SystemRoot%\System32\csrss.exe`
Parent Process: Created by an instance of `smss.exe` that exits, typically appearing as an orphan process.
Number of Instances: Two or more
User Account: Local System
Start Time: Within seconds of boot time for the first two instances (for Session 0 and 1). Start times for additional instances occur as new sessions are created, although often only Sessions 0 and 1 are created.
Description: The Client/Server Run-Time Subsystem is the user-mode process for the Windows subsystem. Its duties include managing processes and threads, importing many of the DLLs that provide the Windows API, and facilitating shutdown of the GUI during system shutdown. An instance of `csrss.exe` will run for each session. Session 0 is for services and Session 1 for the local console session. Additional sessions are created through the use of Remote Desktop and/or Fast User Switching. Each new session results in a new instance of `csrss.exe`.

services.exe

Image Path: `%SystemRoot%\System32\services.exe`
Parent Process: `wininit.exe`
Number of Instances: One
User Account: Local System
Start Time: Within seconds of boot time
Description: Implements the Unified Background Process Manager (UBPM), which is responsible for background activities such as services and scheduled tasks. `Services.exe` also implements the Service Control Manager (SCM), which specifically handles the loading of services and device drivers marked for auto-start. In addition, once a user has successfully logged on interactively, the SCM (`services.exe`) considers the boot successful and sets the Last Known Good control set (`HKLM\SYSTEM\Select\LastKnownGood`) to the value of the CurrentControlSet.

svchost.exe

Image Path: `%SystemRoot%\system32\svchost.exe`
Parent Process: `services.exe` (most often)
Number of Instances: Many (generally at least 10 and often more than 50)
User Account: Varies between Local System, Network Service, or Local Service accounts. Windows 10+ also has "per-user services" running under a user account context with Medium integrity level.
Start Time: Typically close to boot time. However, services can be started after boot (e.g., at logon), resulting in new instances of `svchost.exe` long after boot time.
Description: Generic host process for Windows services. It is used for running service DLLs. Windows differentiates multiple instances of `svchost.exe`, using the "`-k`" parameter pointing to Service Host Groups within the registry. Typical "`-k`" parameters include DcomLaunch, RPCSS, LocalService, netsvcs, NetworkService, UnistackSvcGroup, and more. The "`-s`" parameter identifies the service, such as LanmanServer, WinRM, or Winmgmt. "`-p`" signifies policy enforcement. Malware authors often take advantage of the ubiquitous nature of `svchost.exe` and use it either to host a malicious DLL as a service, or to blend in using a malicious process named `svchost.exe` or similar spelling. In Windows 10 version 1703, Microsoft changed the default grouping of similar services for systems with more than 3.5 GB of RAM. In such cases, most services will now run under their own instance of `svchost.exe` resulting in more than 50 instances of `svchost.exe`.

lsaiso.exe

Image Path: `%SystemRoot%\System32\lsaiso.exe`
Parent Process: `wininit.exe`
Number of Instances: Zero or one
User Account: Local System
Start Time: Within seconds of boot time
Description: When Virtualization-based Security (VBS) is enabled (used with Credential Guard), the functionality of `lsass.exe` is split between two processes—itself and `lsaiso.exe`. Most of the functionality stays within `lsass.exe`, but the important role of safely storing account credentials moves to `lsaiso.exe`. It provides safe storage by running in a context that is isolated from other processes through hardware virtualization technology. When remote authentication is required, `lsass.exe` proxies the requests using an RPC channel with `lsaiso.exe` in order to authenticate the user to the remote service. Note that if VBS is not enabled, `lsaiso.exe` should not be running on the system.

lsass.exe

Image Path: `%SystemRoot%\System32\lsass.exe`
Parent Process: `wininit.exe`
Number of Instances: One
User Account: Local System
Start Time: Within seconds of boot time
Description: The Local Security Authentication Subsystem Service process is responsible for authenticating users by calling an appropriate authentication package specified in `HKLM\SYSTEM\CurrentControlSet\Control\Lsa`. Typically, this will be Kerberos for domain accounts or MSV1_0 for local accounts. In addition to authenticating users, `lsass.exe` is also responsible for implementing the local security policy (such as password policies and audit policies) and for writing events to the security event log. Only one instance of this process should occur and it should rarely have child processes (Encrypting File System is a known exception).

explorer.exe

Image Path: `%SystemRoot%\explorer.exe`
Parent Process: Created by an instance of `userinit.exe` that exits, typically appearing as an orphan process.
Number of Instances: One or more per interactively logged-on user
User Account: Logged-on user(s)
Start Time: First instance starts when the owner's interactive logon begins
Description: At its core, Explorer provides users access to files. Functionally, though, it is both a file browser via Windows Explorer (though still `explorer.exe`) and a user interface providing features such as the user's Desktop, the Start Menu, the Taskbar, the Control Panel, and application launching via file extension associations and shortcut files. `Explorer.exe` is the default user interface specified in the Registry value `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell`, though Windows can alternatively function with another interface such as `cmd.exe` or `powershell.exe`. Notice that the legitimate `explorer.exe` resides in the `%SystemRoot%` directory rather than `%SystemRoot%\System32`. Multiple instances per user can occur, such as when the option "Launch folder windows in a separate process" is enabled.

V. Windows Registry

Saved in `C:\Windows\System32\config` and `C:\Users\<USER>\NTUSER.DAT`

Registry Editor
`regedit.msc`

- `HKEY_CLASSES_ROOT` (HKCR)
  - Registered apps
  - file associations
- `HKEY_LOCAL_MACHINE` (HKLM)
  - Store system settings that apply to all users
- `HKEY_USERS` (HKU)
  - Contains user settings
  - Users separated by sub-keys
- `HKEY_CURRENT_USER` (HKCU)
  - User specific settings for currently logged in user
- `HKEY_CURRENT_CONFIG` (HKCC)
  - Contains info about current hardware profile

Updating Registry Values from Command Prompt

```
reg add "HKCU\SOFTWARE\Microsoft\Notepad" /v lfFaceName /t REG_SZ /d "Comic Sans MS" /f
```

- `/v` - Specify value name
- `/t` - Type of value, (Can be found in Registry column)
- `/d` - Data to add
- `/f` - force change without confirmation

VI. Windows Autoruns

Run Keys

This section focuses on registry items that cause applications/services to automatically run at login. They are called "Run Keys"

Registry Location:

- `HKCU\Software\Microsoft\Windows\CurrentVersion\Run`
- `HKCU\Software\Microsoft\Windows\CurrentVersion\RunOnce`
  - value self-deletes after running
- `HKLM\Software\Microsoft\Windows\CurrentVersion\Run`
- `HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnce`

CMD Version

```
reg query "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run"
reg query "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce"
reg query "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run"
reg query
"HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce"
```

PowerShell Version

```
Get-ItemProperty -Path
"Registry::HKCU\Software\Microsoft\Windows\CurrentVersion\Run"
Get-ItemProperty -Path
"Registry::HKCU\Software\Microsoft\Windows\CurrentVersion\RunOnce"
Get-ItemProperty -Path
"Registry::HKLM\Software\Microsoft\Windows\CurrentVersion\Run"
Get-ItemProperty -Path
"Registry::HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnce"
```

Autoruns

Autoruns is a part of the Sysinternals Suite of Tools. It has a comprehensive list of autostart locations and can even tell when programs will autostart with an application.
https://learn.microsoft.com/en-us/sysinternals/downloads/autoruns
https://github.com/p0w3rsh3ll/AutoRuns

Potential threats are highlighted in red.

Right Click:
- View Properties
- Scan VirusTotal

Compare Baselines:

You can use p0w3rsh3ll/AutoRuns to compare a benign baseline registry against a potentially malicious one. This will help you discover malicious edits

```
Import-Module .\AutoRuns.psm1
Get-Module -Name AutoRuns
Get-Command -Module AutoRuns
Get-PSAutorun -VerifyDigitalSignature | Where { -not($_.isOSbinary)} | New-
AutoRunsBaseline -Verbose -FilePath .\Baseline.ps1 #Create Baseline

Compare-AutoRunsBaseLine -ReferenceBaseLineFile .\Baseline.ps1 -
DifferenceBaseLineFile .\CurrentState.ps1 -Verbose
```

VII. Windows Service Analysis

`services.msc` - Run as Administrator

Right click for properties and look at path to executable

```
sc query state= all
sc qc BackupService
```

VIII. Windows Scheduled Tasks

Sysinternals Autoruns
Scheduled Tasks and Services

IX. Windows Event Logs

`Event Viewer`

- Application
- Security
- Setup
- System

Filter Events

`Create Custom View...`

Security Event IDs

- 4720 - A user account was created
- 4722 - A user account was enabled
- 4723 - An attempt was made to change an account's password
- 4724 - An attempt was made to reset an account's password
- 4738 - A user account was changed
- 4725 - A user account was disabled
- 4726 - A user account was deleted
- 4732 - A member was added to a security-enabled local group
- 4688 - A new process has been created
- 1102 - The audit log was cleared
- 7045 - A service was installed in the system
- 7030 - The Service Control Manager tried to take a corrective action (Restart the service)
- 7035 - The Service Control Manager is transitioning service to a running state
- 7036 - The Service Control Manager has reported that a service has entered the running state
-

`wevutil`

```
wevutil qe security /c:5 /f:text /rd:true
```

- `/c` - Number of events returned
- `/f` - Format

- `/rd` - Reverse Direction

```
wevutil qe security /c:5 /f:text /rd:true /q:"*[System [(EventID=4720)]]"
```

```
Get-WinEvent
```

```
Get-WinEvent -LogName System
```

```
Get-WinEvent -FilterHashtable @{logname='Security'; ID=4624;} -MaxEvents 2 |
Format-List *
```

X. Sysmon

`Event Viewer` > Applications and Service Logs > Microsoft > Windows > Sysmon

- https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon
- https://github.com/SwiftOnSecurity/sysmon-config
- Event ID 1 - Process Creation
- Event ID 3 - Network Connection
- Event ID 5 - Process Terminated
- Event ID 7 - Image Loaded
- Event ID 8 - CreateRemoteThread
- Event ID 10 - ProcessAccess
- Event ID 11 - FileCreate
- Event ID 12, 13, 14 - Registry Events
- Event ID 15 - FileCreateStreamHash
- Event ID 22 - DNSEvent (DNS Query)
- Event ID 29 - FileExecutableDetected