Core Windows Processes

Task Manager

- built in GUI based Windows utility that allows users to see what is running on the Windows system
- provides information on resource usage, such as how much each process utilizes CPU and memory

Task Manager columns:

Type – each process falls into 1 of 3 categories (Apps, Background process, or Windows process)
Publisher – name of the program/file
PID – process identifier number. Windows assigns a unique process identifier each time a program starts
Process Name – file name of the process
CPU – amount of CPU (processing power) the process uses
Memory – amount of physical working memory utilized by the process

*Good columns to add are 'Image path name' and 'Command line'
*These 2 columns can quickly alert an analyst of any outliners with a given process

Name	PID	Status	User name	Image path name	Command line
System interrupts	-	Running	SYSTEM		
System Idle Process	0	Running	SYSTEM		
System	4	Running	SYSTEM	C:\Windows\system32\ntoskrnl.exe	
Registry	88	Running	SYSTEM	C:\Windows\system32\ntoskrnl.exe	
smss.exe	280	Running	SYSTEM	C:\Windows\System32\smss.exe	
svchost.exe	384	Running	LOCAL SERVICE	C:\Windows\System32\svchost.exe	C:\Windows\System32\svchost.exe -k LocalServiceNetworkRestricted -p

PID 384 is paired with a process name svchost.exe, but if the Image path name + Command line is not what it's expected to be, then we can investigate.

Tools: Process Hacker + Process Explorer to obtain more information about each Windows process.

User mode and Kernel mode

- processor in a computer running Windows has 2 different modes: user mode and kernel mode
- the processor switches between the 2 modes depending on what type of code is running on the processor

- applications run in user mode
- core operating system components run in kernel mode

User mode

- when a user mode application is started, Windows creates a process for the application
- process provides the application with a private virtual address space and a private handle table

*Each application runs in isolation, and if an application crashes, the crash is limited to that one application. Other applications and the OS are not affected by the crash

Kernel mode

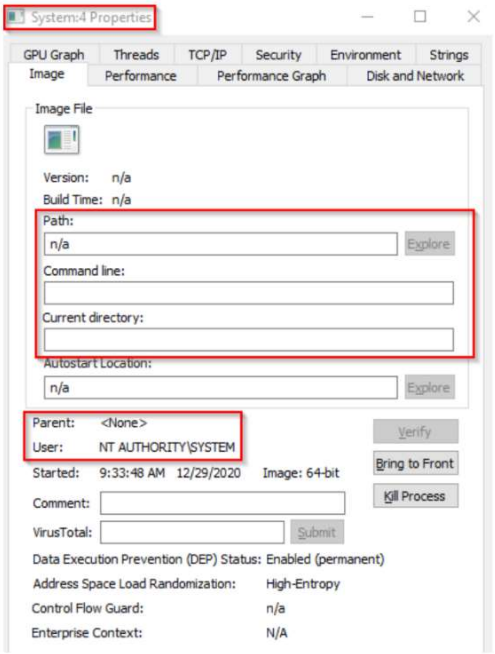
- all code that runs in kernel mode shares a single virtual address space
- a kernel mode driver is not isolated from other drivers and the OS

*If a kernel mode driver accidently writes to the wrong virtual address, data that belongs to the OS or another driver could be compromised

System
- first Windows process on the list is System
- PID for System is always 4

*System process (PID 4) is the home for a special kind of threat that runs only in kernel mode

Use Process Explorer + Process Hacker to view properties



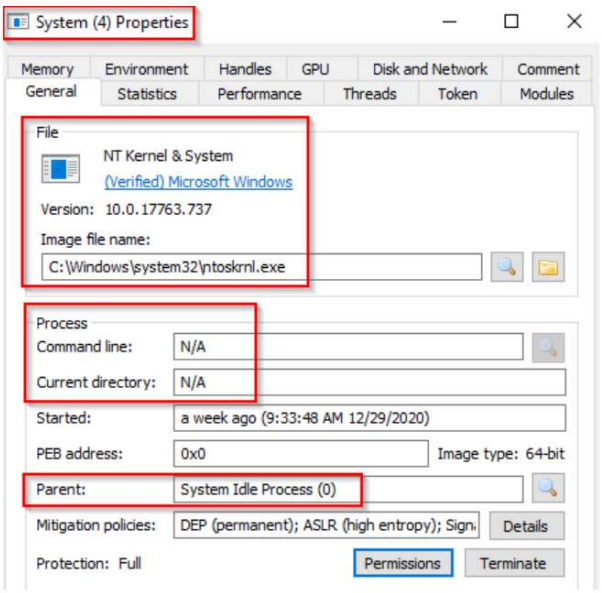


Image Path: C:\Windows\system32\ntoskrnl.exe (NT OS Kernel)
Parent Process: System Idle Process (0)

*You may notice that Process Hacker confirms this is legit (Verified) Microsoft Windows.

What is unusual behaviour for this process?
- A parent process (aside from System Idle Process (0))
- Multiple instances of System. (Should only be one instance)
- A different PID. (Remember that the PID will always be PID 4)
- Not running in Session 0

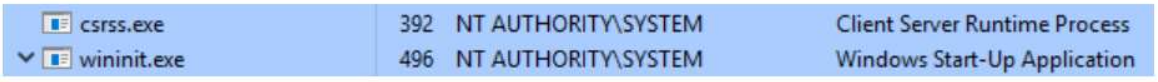
System > smss.exe
- Session Manager Subsystem (Windows Session Manager)
- responsible for creating new sessions
- first user mode started by the kernel

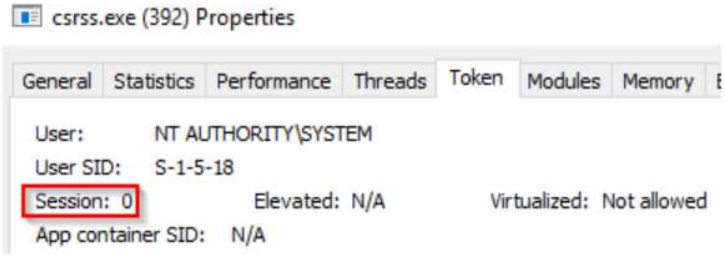
- process starts the kernel and user modes of the Windows subsystem
- subsystem includes win32k.sys (kernel mode), winsrv.dll (user mode), and csrss.exe (user mode)
- starts winlogon.exe (Windows Logon Manager)

Smss.exe starts csrss.exe and wininit.exe in Session 0 (isolated Windows session for the OS), and csrss.exe and winlogon.exe for Session 1 (user session).

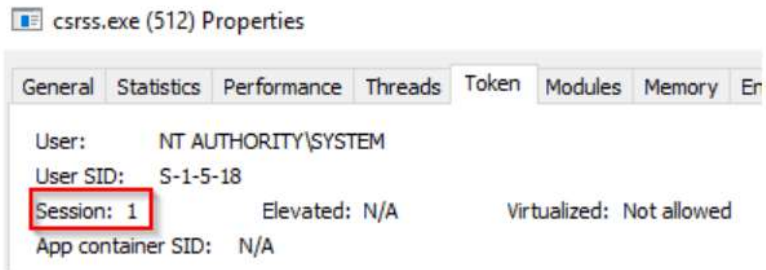
First child instance creates child instances in new sessions, done by smss.exe copying itself to the new session and self-terminating.

Session 0 (crss.exe + wininit.exe)





Session 1 (csrss.exe + winlogon.exe)



Any other subsystem listed in the 'Required' value of registry path is launched:
HKLM\System\CurrentControlSet\Control\Session Manager\Subsystems

*smss.exe is also responsible for creating environment variables, virtual memory paging files and starts winlogon.exe (Windows Logon Manager).

What is normal?

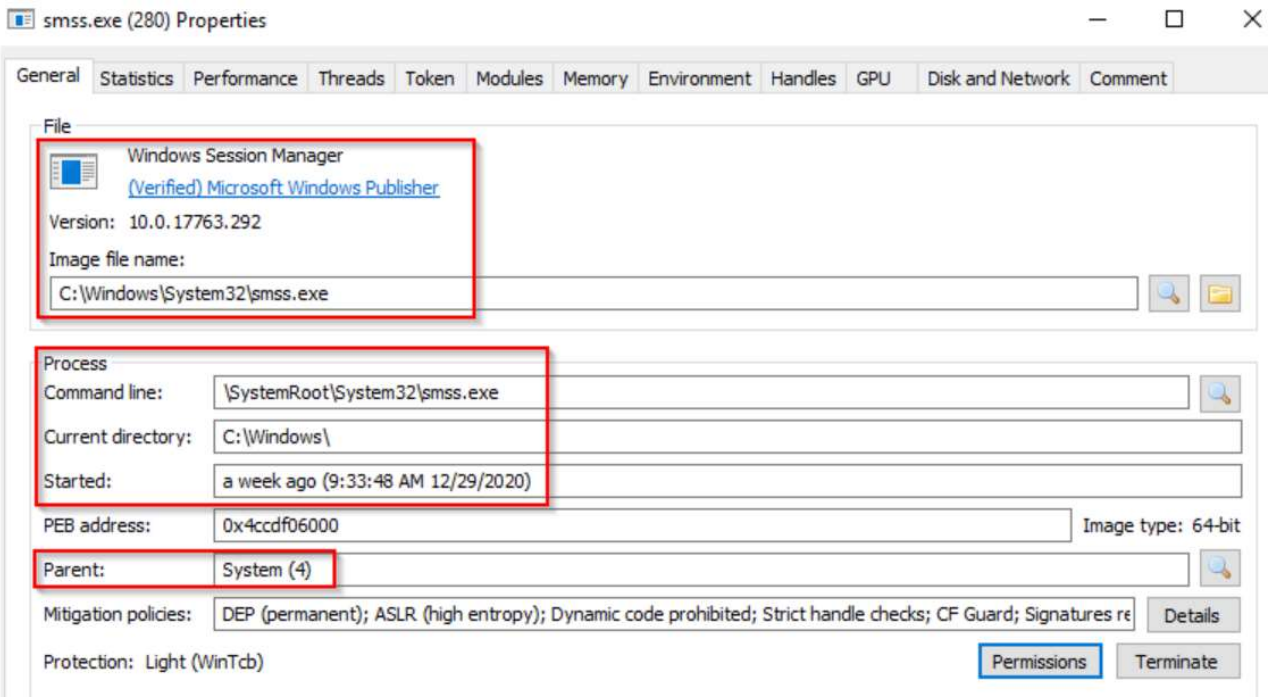


Image Path: %SystemRoot%\System32\smss.exe
Parent Process: System
Number of Instances: One master instance and child instance per session. The child instance exits after creating the session.
User Account: Local System
Start Time: Within seconds of boot time for the master instance

What is unusual?
- A different parent process other than System (4)
- The image path is different from C:\Windows\System32
- More than one running process. (children self-terminate and exit after each new session)
- The running User is not the SYSTEM user
- Unexpected registry entries for Subsystem

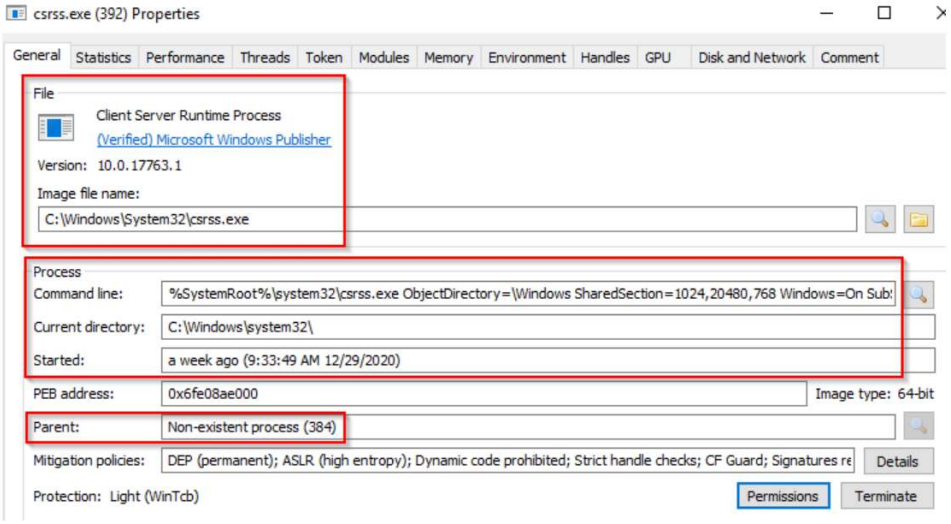
csrss.exe

- Client Server Runtime Process
- user mode side of the Windows subsystem
- process is always running and is critical to system operation
- if process is terminated > system failure

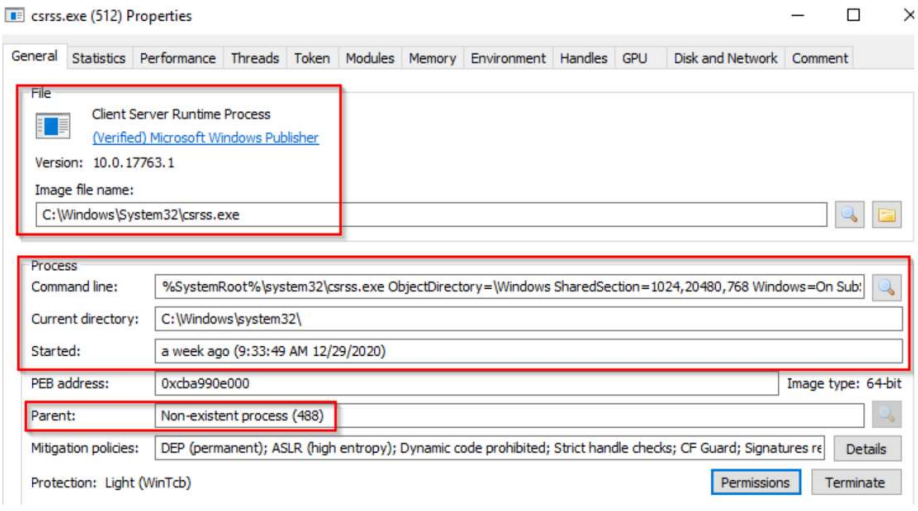
csrss.exe is called along with winlogon.exe from smss.exe at Windows start up.
*If either files are corrupted > smss.exe will tell the kernel to shut down the start up process with a Blue Screen of Death
Error code fault: 0xc000021a (STATUS_SYSTEM_PROCESS_TERMINATED)

What is normal?

Session 0 (PID 392)



Session 1 (PID 512)



- notice what is shown for the parent process for these 2 processes
- these processes are spawned by smss.exe, which self-terminates (Parent: Non-existent process)

Image Path: %SystemRoot%\System32\csrss.exe
Parent Process: Created by an instance of smss.exe
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 only Sessions 0 and 1 are often created.

What is unusual?
- An actual parent process. (smss.exe calls this process and self-terminates)
- Image file path other than C:\Windows\System32
- Subtle misspellings to hide rogue processes masquerading as csrss.exe in plain sight
- The user is not the SYSTEM user

wininit.exe

- Windows Initialization Process
- responsible for launching services.exe (Service Control Manager), lsass.exe (Local Security Authority), and lsaiso.exe within Session 0
- critical Windows process that runs in the background, along with its child processes

> services.exe
  lsass.exe

What is normal?

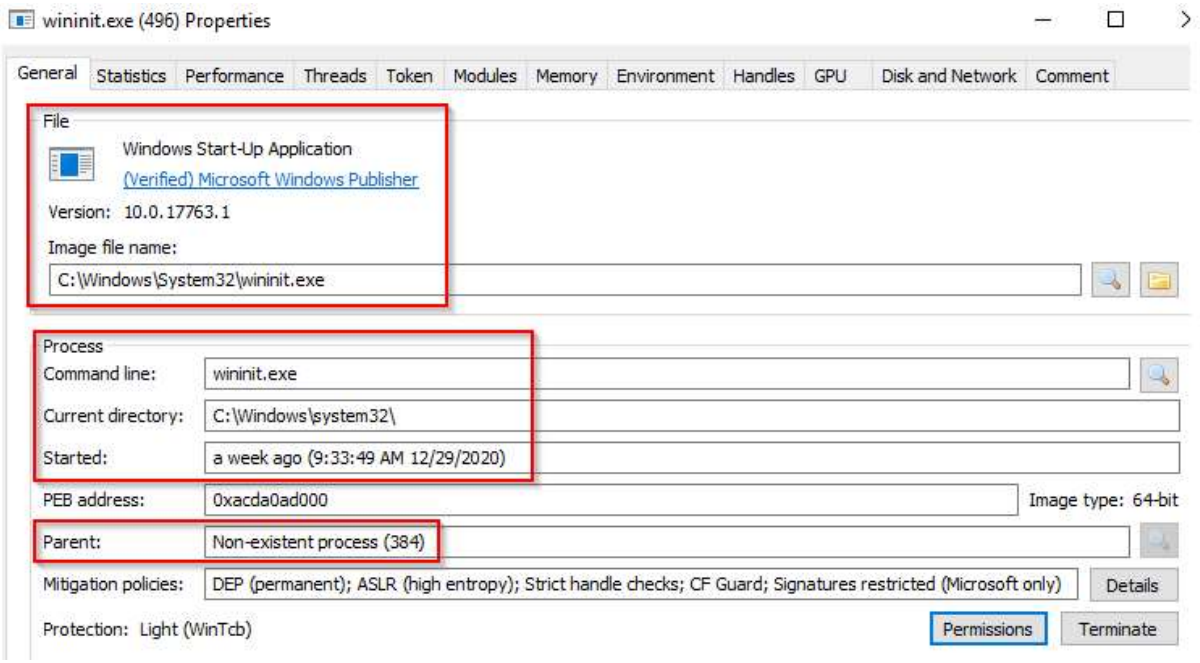


Image Path: %SystemRoot%\System32\wininit.exe
Parent Process: Created by an instance of smss.exe
Number of Instances: One
User Account: Local System
Start Time: Within seconds of boot time

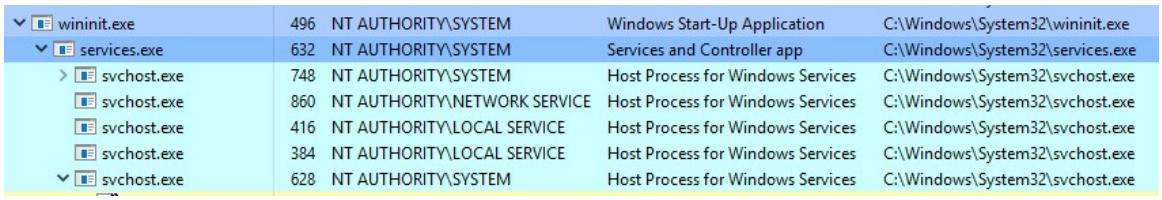
What is unusual?

- An actual parent process. (smss.exe calls this process and self-terminates)
- Image file path other than C:\Windows\System32
- Subtle misspellings to hide rogue processes in plain sight
- Multiple running instances
- Not running as SYSTEM

wininit.exe > services.exe
- next process is Service Control Manager (SCM) or services.exe
- responsible for handling system services; loading services, interacting with services and starting/ending services
- terminate process > Blue Screen of Death
- maintains a database that can be queried using a Windows built-in utility > sc.exe

Information regarding services is stored in the registry:
HKLM\System\CurrentControlSet\Services

- process also loads device drivers marked as auto-start into memory
- services.exe is the parent to several other key processes: svchost.exe, msmpeng.exe, spoolsv.exe



What is normal?

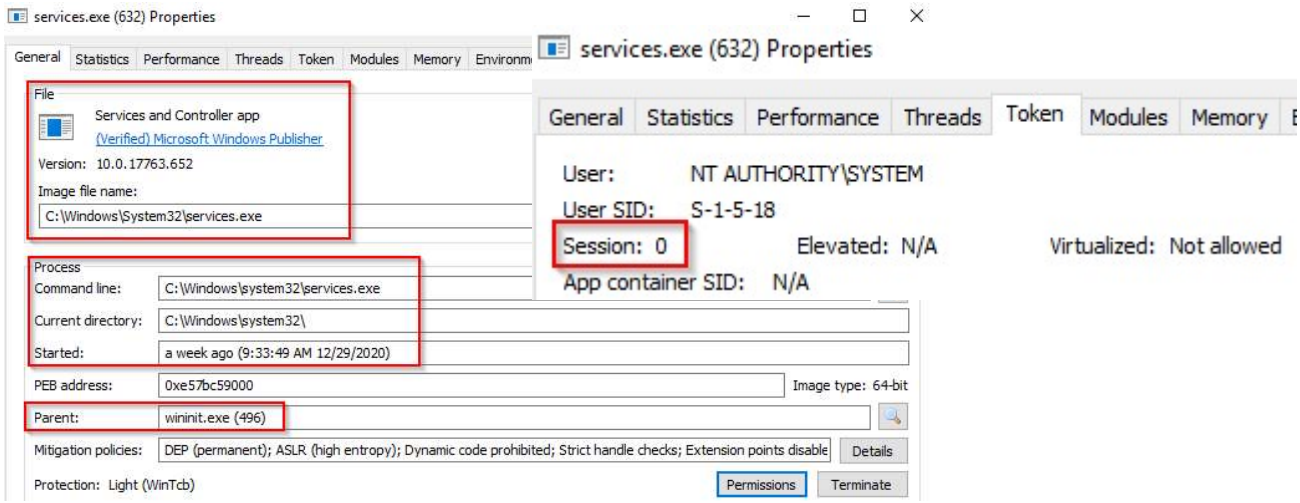
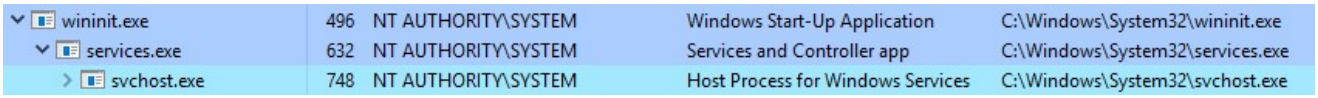


Image Path: %SystemRoot%\System32\services.exe
Parent Process: wininit.exe
Number of Instances: One
User Account: Local System
Start Time: Within seconds of boot time

What is unusual?
- A parent process other than wininit.exe
- Image file path other than C:\Windows\System32
- Subtle misspellings to hide rogue processes in plain sight
- Multiple running instances
- Not running as SYSTEM

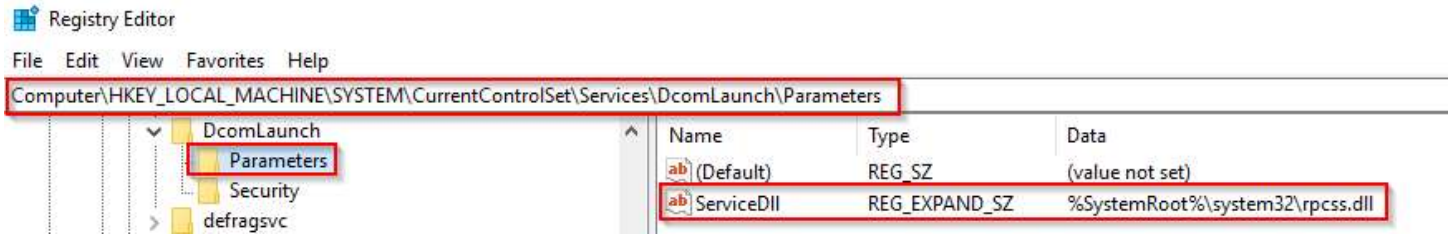
wininit.exe > services.exe > svchost.exe

- Service Host (Host Process for Windows Services)
- responsible for hosting and managing Windows services

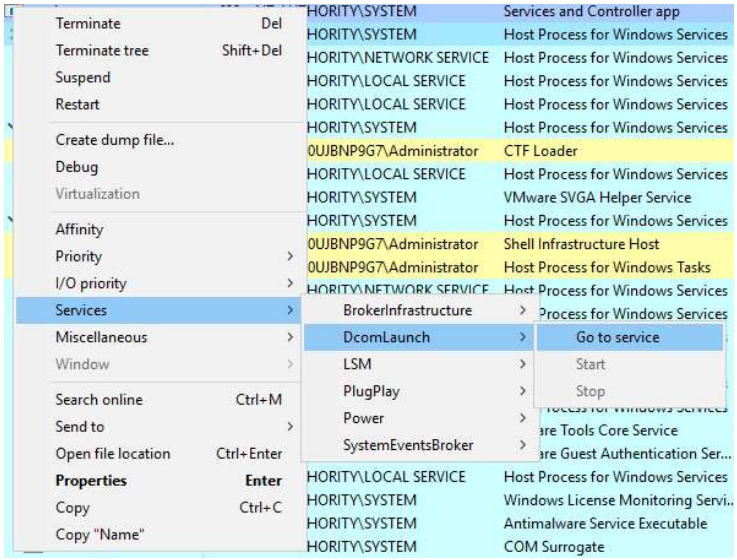


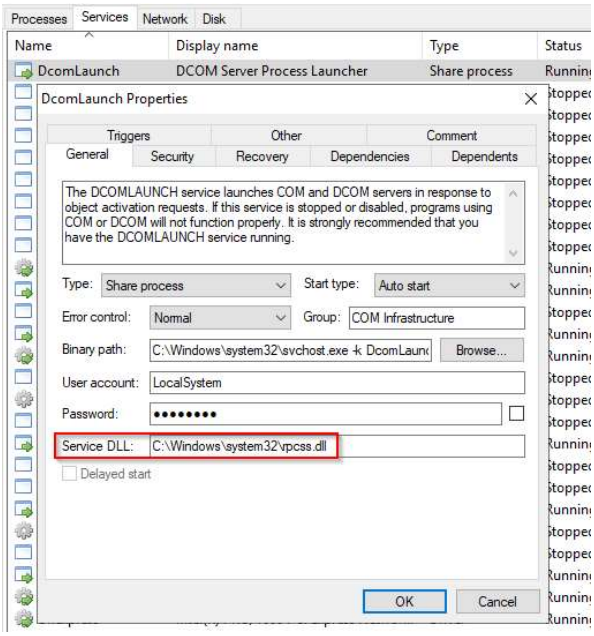
- services running in this process are implemented as DLLs
- the DLL to implement is stored in the registry for the service under 'Parameters' subkey in 'ServiceDLL'
- full path: HKLM\SYSTEM\CurrentControlSet\Services\SERVICE NAME\Parameters

Example below is the ServiceDLL value for the Dcomlaunch service



View process in Process Hacker (PID748)





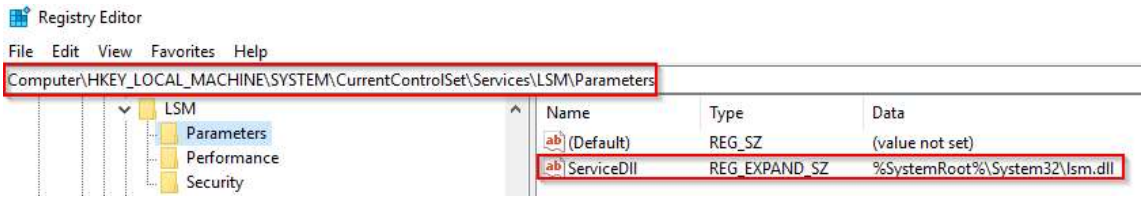
- binary path is listed: C:\Windows\system32\svchost.exe -k DcomLaunch
- key identifier in the binary path; -k (how a legitimate svchost.exe process is called)

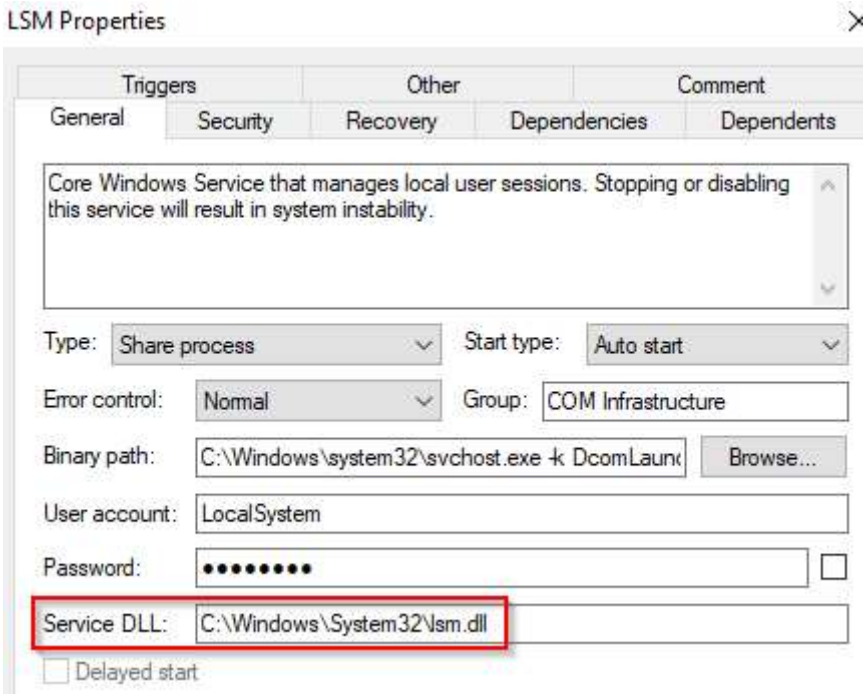
-k value is for DcomLaunch

- other services are running with the same binary path

BrokerInfrastructure	Background Tasks Infrastructure Servi...	Share process	Running	Auto start	748	C:\Windows\system32\svchost.exe -k DcomLaunch -p
LSM	Local Session Manager	Share process	Running	Auto start	748	C:\Windows\system32\svchost.exe -k DcomLaunch -p
Power	Power	Share process	Running	Auto start	748	C:\Windows\system32\svchost.exe -k DcomLaunch -p
PlugPlay	Plug and Play	Share process	Running	Demand start	748	C:\Windows\system32\svchost.exe -k DcomLaunch -p
SystemEventsBroker	System Events Broker	Share process	Running	Auto start (trigger)	748	C:\Windows\system32\svchost.exe -k DcomLaunch -p
DcomLaunch	DCOM Server Process Launcher	Share process	Running	Auto start	748	C:\Windows\system32\svchost.exe -k DcomLaunch -p
DeviceInstall	Device Install Service	Share process	Stopped	Demand start (trigger)		C:\Windows\system32\svchost.exe -k DcomLaunch -p

- each service will have a different value for ServiceDLL
- use LSM as an example





*Since svchost.exe will always have multiple running processes on any Windows system, this has been a target for malicious use. Adversaries create malware to try and hide amongst the legitimate svchost.exe processes (misspell the process to hide)

https://www.hexacorn.com/blog/2013/07/04/the-typographical-and-homomorphic-abuse-of-svchost-exe/

svchost
svch0st
svchosts
scvhost
svhost
svohost

What is normal?

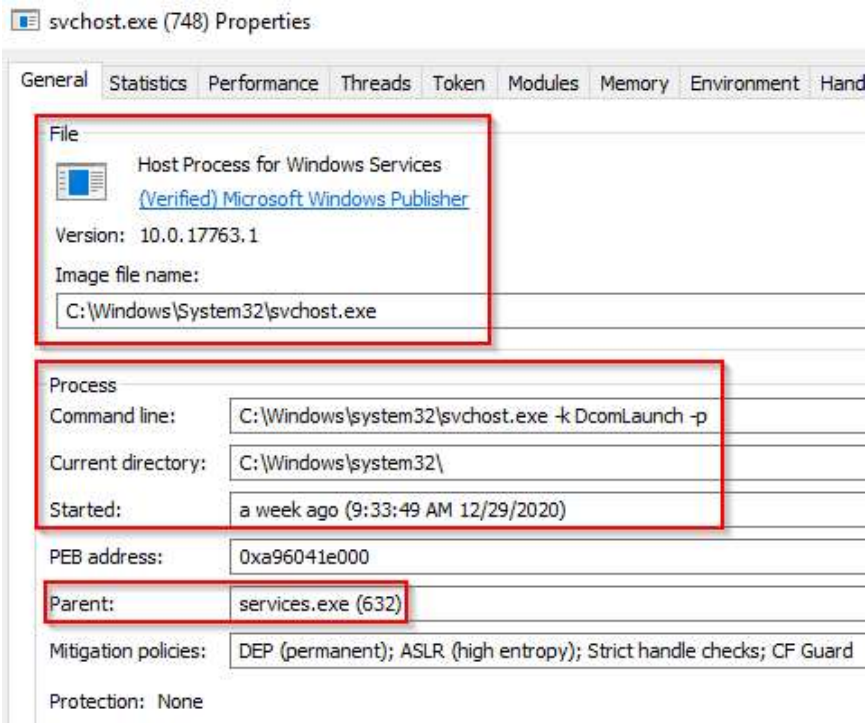


Image Path: %SystemRoot%\System32\svchost.exe
Parent Process: services.exe
Number of Instances: Many
User Account: Varies (SYSTEM, Network Service, Local Service) depending on the svchost.exe instance. In Windows 10, some instances run as the logged-in user.
Start Time: Typically within seconds of boot time. Other instances of svchost.exe can be started after boot.

What is unusual?
- A parent process other than services.exe
- Image file path other than C:\Windows\System32
- Subtle misspellings to hide rogue processes in plain sight
- The absence of the -k parameter

lsass.exe
- Local Security Authority Subsystem
- responsible for enforcing the security policy on the system
- it verifies users logging on to a Windows computer or server + handles password changes
- writes to the Windows Security Log

- uses authentication packages specified in: HKLM\System\CurrentControlSet\Lsa

*Lsass.exe is another process targeted by attackers. Common tools such as mimikatz are used to dump credentials, and attackers use this process to hide (change the name of malware or misspell the malware slightly)

What is normal?

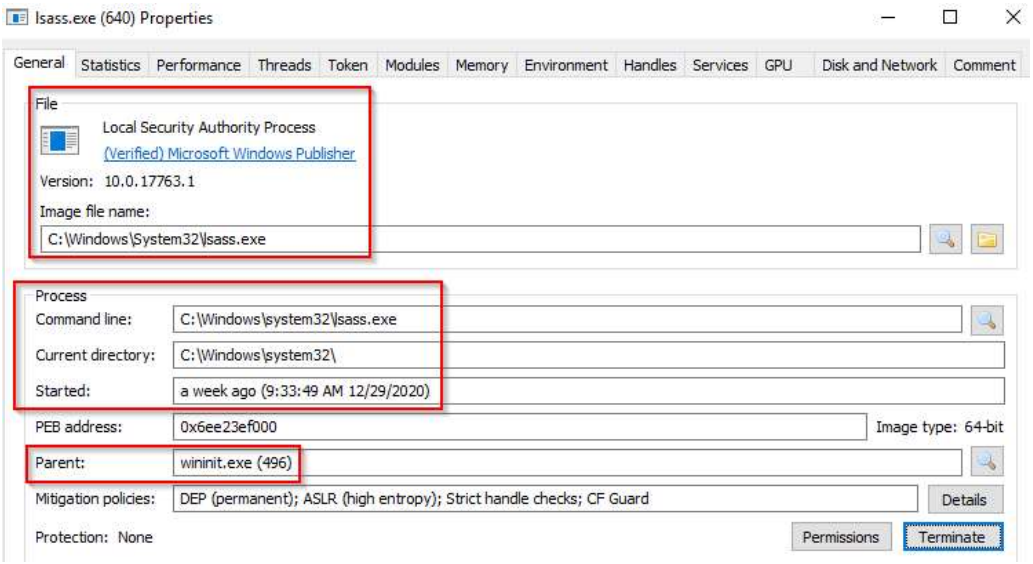


Image Path: %SystemRoot%\System32\lsass.exe
Parent Process: wininit.exe
Number of Instances: One
User Account: Local System
Start Time: Within seconds of boot time

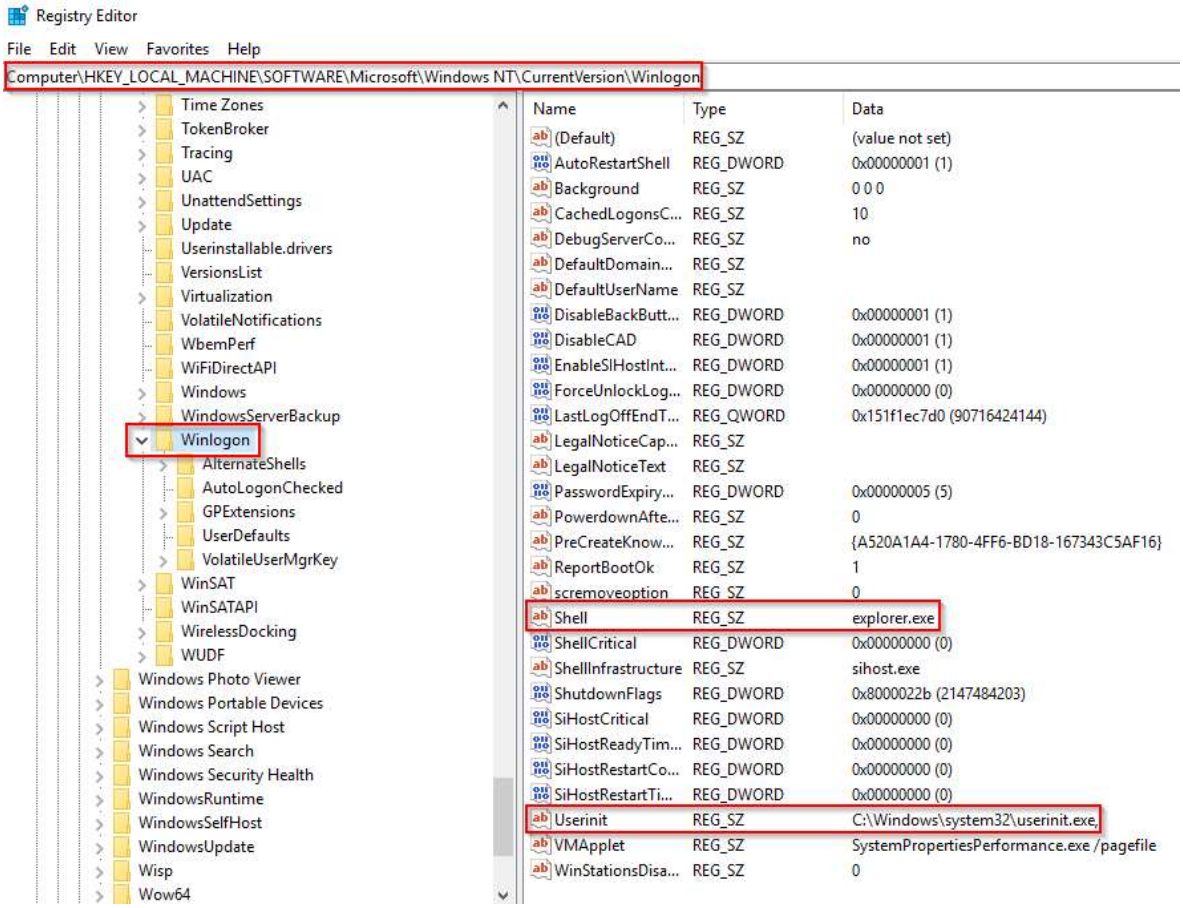
What is unusual?
- A parent process other than wininit.exe
- Image file path other than C:\Windows\System32
- Subtle misspellings to hide rogue processes in plain sight
- Multiple running instances
- Not running as SYSTEM

winlogon.exe
- Window Logon
- responsible for handling the Secure Attention Sequence (SAS), CTRL+ALT+DELETE combination users press to enter username and password

- responsible for loading the user profile
- loads the NTUSER.DAT
- userinit.exe loads the user's shell

> Userinit
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon

*Specifies the programs that Winlogon runs when a user logs on
*Winlogon runs userinit.exe, which runs scripts/establishes network connections/starts explorer.exe (Windows User Interface)



*smss.exe launches this process along with a copy of csrss.exe within Session 1

What is normal?

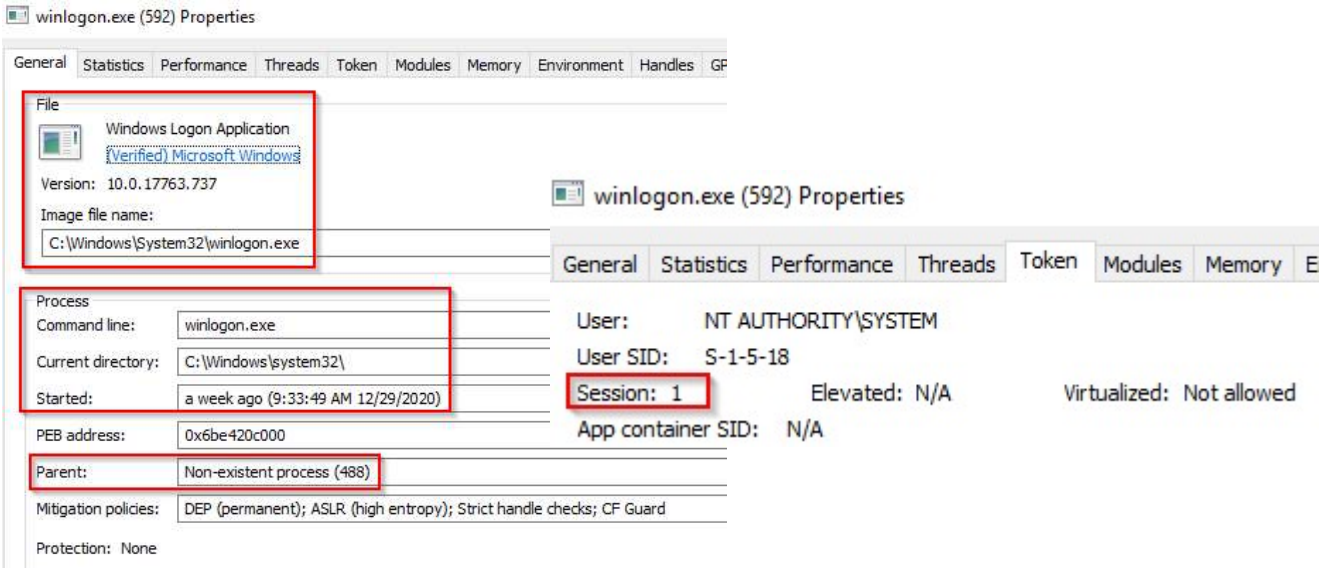


Image Path: %SystemRoot%\System32\winlogon.exe
Parent Process: Created by an instance of smss.exe that exits, so analysis tools usually do not provide the parent process name.
Number of Instances: One or more
User Account: Local System
Start Time: Within seconds of boot time for the first instance (for Session 1). Additional instances occur as new sessions are created, typically through Remote Desktop or Fast User Switching logons.

What is unusual?
-An actual parent process. (smss.exe calls this process and self-terminates)
- Image file path other than C:\Windows\System32
- Subtle misspellings to hide rogue processes in plain sight
- Not running as SYSTEM
- Shell value in the registry other than explorer.exe

explorer.exe
- Windows Explorer
- gives the user access to their folders + files

Winlogon runs userinit.exe, which launches the value in:
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell

*userinit.exe exists after spawning explorer.exe > parent process is non-existent

What is normal?

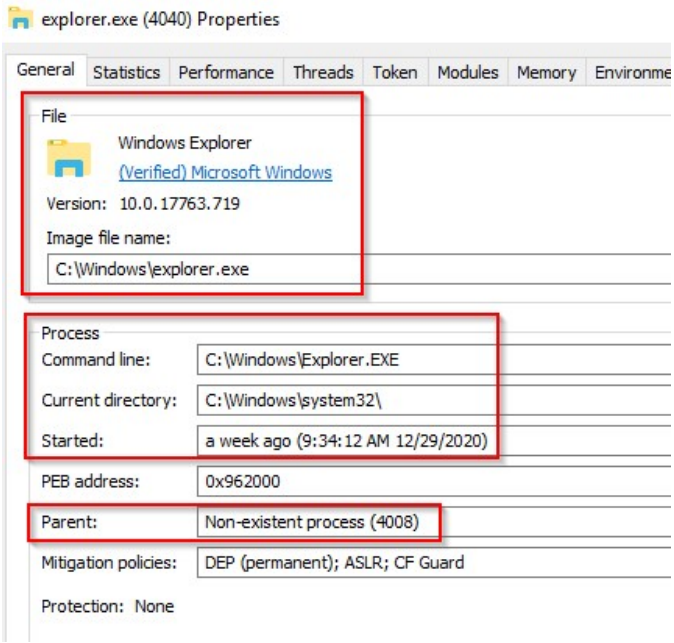


Image Path: %SystemRoot%\explorer.exe
Parent Process: Created by userinit.exe and exits
Number of Instances: One or more per interactively logged-in user
User Account: Logged-in user(s)
Start Time: First instance when the first interactive user logon session begins

What is unusual?
- An actual parent process. (userinit.exe calls this process and exits)
- Image file path other than C:\Windows
- Running as an unknown user
- Subtle misspellings to hide rogue processes in plain sight
- Outbound TCP/IP connections