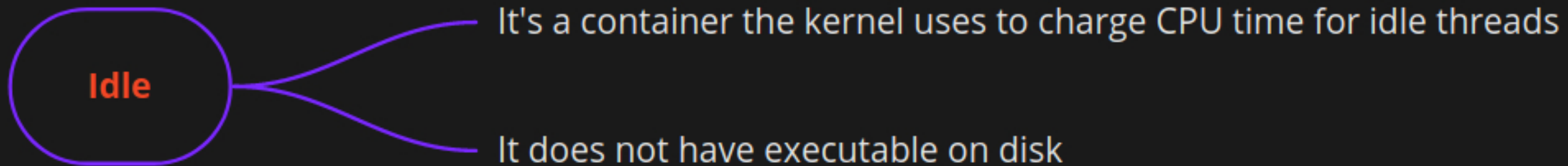


The Art of Memory Forensics

Ch.6 - Processes, Handles and Tokens



Critical System Processes





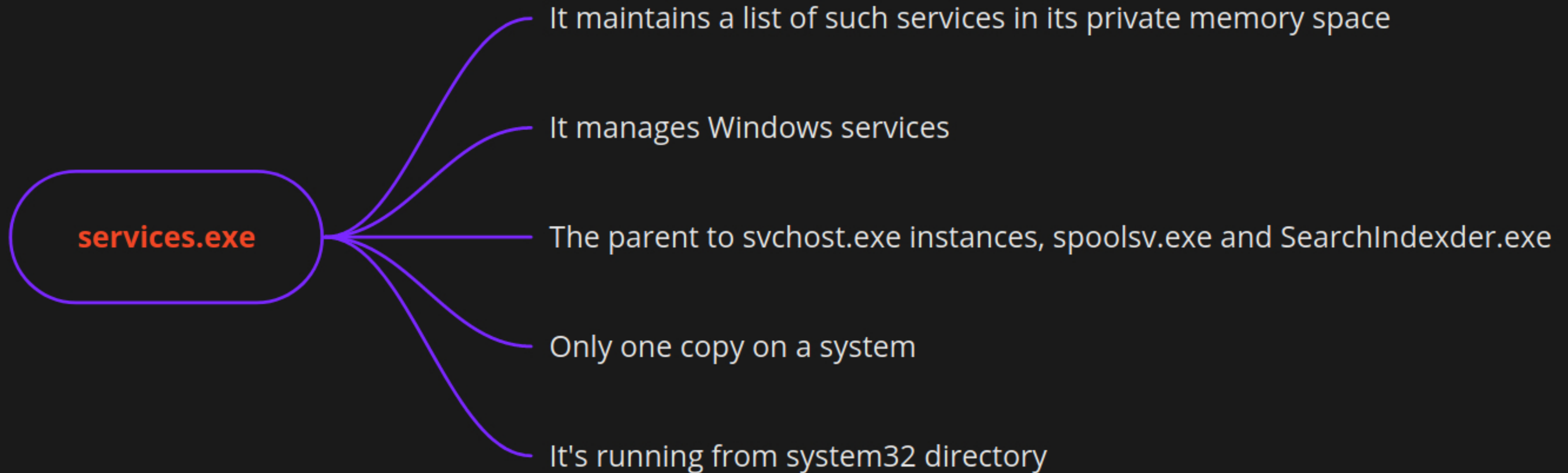
System

It appears to own any sockets or handles to files that kernel modules open

It does not have executable on disk

It's the default home for threads that run in kernel mode

PID 4



csrss.exe



```
graph LR; A((csrss.exe)) --- B[Attackers attempt to exploit the naming convention (csrss.exe, cssrs.exe, etc)]; A --- C[It plays role in creating and deleting processes and threads]; A --- D[It's located in the system32 directory]; A --- E[Each session gets a dedicated copy so expect to see multiple csrss processes]; A --- F[The client/server runtime subsystem]; A --- G[It maintains a private list of the objects that can be used to cross-reference with other data sources];
```

Attackers attempt to exploit the naming convention (csrss.exe, cssrs.exe, etc)

It plays role in creating and deleting processes and threads

It's located in the system32 directory

Each session gets a dedicated copy so expect to see multiple csrss processes

The client/server runtime subsystem

It maintains a private list of the objects that can be used to cross-reference with other data sources

svchost.exe

Attackers attempt to exploit the naming convention (scvhost.exe, svch0st.exe, svshost.exe, etc)

Its executable path should point to system32 directory

Child of services.exe

Multiple shared host processes can be running concurrently

It's a container for DLLs that implements services.

winlogon.exe

```
graph LR; A(winlogon.exe) --- B[It helps load user profiles]; A --- C[It initiates the screen saver when necessary]; A --- D[It presents the interactive logon prompt]; A --- E[Its executable is located in the system32 directory]; A --- F[It responds to Secure process monitors files and directories for changes on systems that implement Windows File Protection (WFP)];
```

It helps load user profiles

It initiates the screen saver when necessary

It presents the interactive logon prompt

Its executable is located in the system32 directory

It responds to Secure process monitors files and directories for changes on systems that implement Windows File Protection (WFP)

explorer.exe

```
graph LR; A(explorer.exe) --- B[It has access to sensitive material such as the documents you open]; A --- C[It handles variety of user interactions such as folder navigation, presenting the start menu, etc]; A --- D[It has access to credentials you use to log-in to FTP sites via Windows Explorer]; A --- E[One for each logged-on user];
```

It has access to sensitive material such as the documents you open

It handles variety of user interactions such as folder navigation, presenting the start menu, etc

It has access to credentials you use to log-in to FTP sites via Windows Explorer

One for each logged-on user

smss.exe

A mind map diagram with a central node 'smss.exe' in a rounded rectangle. Three lines branch out to the right, connecting to three text nodes: 'It's the first user-mode process that starts during the boot sequence', 'The session manager', and 'It creates the sessions that isolate OS services from users who may log on via console or Remote Desktop Protocol (RDP)'.

```
graph LR; A(smss.exe) --- B(It's the first user-mode process that starts during the boot sequence); A --- C(The session manager); A --- D(It creates the sessions that isolate OS services from users who may log on via console or Remote Desktop Protocol (RDP))
```

It's the first user-mode process that starts during the boot sequence

The session manager

It creates the sessions that isolate OS services from users who may log on via console or Remote Desktop Protocol (RDP)

Detecting DKOM Attacks

Direct Kernel Object Manipulation

One of the most common attacks is it to hide a certain process by unlinking its entry from the doubly linked list by overwriting the Flink and Blink pointers of surrounding objects so they point **around** the `_EPROCESS` structure of the process they want to hide

`psslist` and `pstree` are susceptible to this attack because they rely on the linked list

`psscan` is not susceptible to this attack because it uses the pool-scanning approach so it finds the `_EPROCESS` structure even if it was unlinked from the list

Prolaco malware author used this technique and it's explained on pages 160, 161

Analysis Objectives

```
graph LR; A[Analysis Objectives] --- B[Map SIDs to usernames]; A --- C[Detect lateral movement]; A --- D[Profile process behaviors]; A --- E[Detect privilege escalation]; B --- B1[A process' token contains numerical SID values that can be translated into a string then resolved into a user or group, which enables us to determine the primary user account.]; C --- C1[Some hacking techniques leave obvious artifacts in a process' token, as in the jump of process' security context of Domain Admin or Enterprise Admin]; D --- D1[If a process plans to engage in a task, it first ensures the privileges is present and enabled in its token so the analysis of the privileges a process acquired can provide some clues about what the process did or planned to do.]; E --- E1[Some tools such as Process Explorer can be deceived into reporting that a process has fewer privileges than it actually has so we use memory forensics to determine the truth];
```

Map SIDs to usernames

A process' token contains numerical SID values that can be translated into a string then resolved into a user or group, which enables us to determine the primary user account.

Detect lateral movement

Some hacking techniques leave obvious artifacts in a process' token, as in the jump of process' security context of Domain Admin or Enterprise Admin

Profile process behaviors

If a process plans to engage in a task, it first ensures the privileges is present and enabled in its token so the analysis of the privileges a process acquired can provide some clues about what the process did or planned to do.

Detect privilege escalation

Some tools such as Process Explorer can be deceived into reporting that a process has fewer privileges than it actually has so we use memory forensics to determine the truth

Extracting and Translating SIDs in Memory



```
graph LR; A(Extracting and Translating SIDs in Memory) --- B(ConvertSidToStringSid API translates the numerical data in _SID structure to human-readable); A --- C(LookupAccountSid API returns an account name for a given SID); A --- D(getsid volatility plugin can find each process' token, extract the numerical component of _SID structure, translate them into strings, and map the strings to use and group names (learn more on page 168));
```

ConvertSidToStringSid API translates the numerical data in `_SID` structure to human-readable

LookupAccountSid API returns an account name for a given SID

getsid volatility plugin can find each process' token, extract the numerical component of `_SID` structure, translate them into strings, and map the strings to use and group names (learn more on page 168)

Privileges



A privilege must be present in the process' token before the process can enable it

Administrators decide which privileges are present by configuring them in the Local Security Policy (LSP) as shown in figure 6-9 on page 171

Volatility plugin `privs` is used to analyze privileges, learn more from pages 172,173

