

Program code: get the process id of lsass.exe

```
#include <windows.h>
#include <tlhelp32.h>
#include <stdio.h>

int main() {

    const char *targetProcName= "lsass.exe";
    int processID = 0;

    // snapshot of all processes in the system
    HANDLE processSnapshot = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);

    // initializing process entry structure
    PROCESSENTRY32 processEntry;
    processEntry.dwSize = sizeof(PROCESSENTRY32);

    // info about first process encountered in a system snapshot
    BOOL operationResult = Process32First(processSnapshot, &processEntry);

    printf("[*] Scanning for process: %s\n", targetProcName);

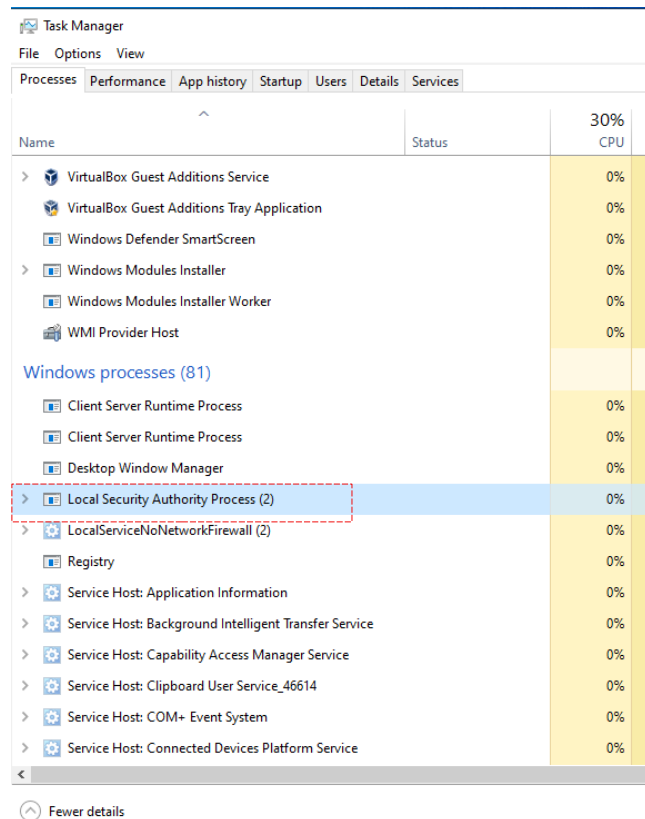
    // retrieve information about the processes
    while (operationResult) {

        // if we find the process: return process ID
        if (strcmp(targetProcName, processEntry.szExeFile) == 0) {
            printf("[+] Match found: %s (PID: %lu)\n", processEntry.szExeFile, processEntry.th32ProcessID);
            processID = processEntry.th32ProcessID;
            break;
        }

        operationResult = Process32Next(processSnapshot, &processEntry);
    }

    // closes an open handle (CreateToolhelp32Snapshot)
    CloseHandle(processSnapshot);

    return processID;
}
```



Program code: get the process id of lsass.exe

```
#include <windows.h>
#include <tlhelp32.h>
#include <stdio.h>
```

```
int main() {
```

```
    const char *targetProcName= "lsass.exe";
    int processID = 0;
```

```
    // snapshot of all processes in the system
```

```
    HANDLE processSnapshot = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);
```

```
    // initializing process entry structure
```

```
    PROCESSENTRY32 processEntry;
```

```
    processEntry.dwSize = sizeof(PROCESSENTRY32);
```

```
    // info about first process encountered in a system snapshot
```

```
    BOOL operationResult = Process32First(processSnapshot, &processEntry);
```

```
    printf("[*] Scanning for process: %s\n", targetProcName);
```

```
    // retrieve information about the processes
```

```
    while (operationResult) {
```

```
        // if we find the process: return process ID
```

```
        if (strcmp(targetProcName, processEntry.szExeFile) == 0) {
```

```
            printf("[+] Match found: %s (PID: %lu)\n", processEntry.szExeFile, processEntry.th32ProcessID);
```

```
            processID = processEntry.th32ProcessID;
```

```
            break;
```

```
        }
```

```
        operationResult = Process32Next(processSnapshot, &processEntry);
```

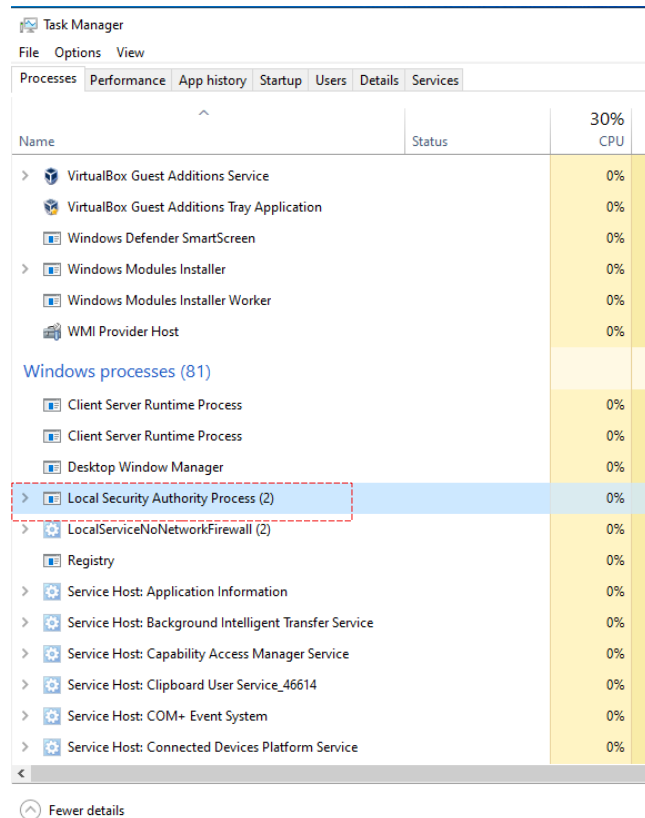
```
    }
```

```
    // closes an open handle (CreateToolhelp32Snapshot)
```

```
    CloseHandle(processSnapshot);
```

```
    return processID;
```

```
}
```



Task Manager

File Options View

Processes Performance App history Startup Users Details Services

Name	Status	30% CPU
> VirtualBox Guest Additions Service		0%
VirtualBox Guest Additions Tray Application		0%
Windows Defender SmartScreen		0%
> Windows Modules Installer		0%
Windows Modules Installer Worker		0%
WMI Provider Host		0%
Windows processes (81)		
Client Server Runtime Process		0%
Client Server Runtime Process		0%
Desktop Window Manager		0%
> Local Security Authority Process (2)		0%
> LocalServiceNoNetworkFirewall (2)		0%
Registry		0%
> Service Host: Application Information		0%
> Service Host: Background Intelligent Transfer Service		0%
> Service Host: Capability Access Manager Service		0%
> Service Host: Clipboard User Service_46614		0%
> Service Host: COM+ Event System		0%
> Service Host: Connected Devices Platform Service		0%

< Fewer details

Program code: get the process id of lsass.exe

```
#include <windows.h>
#include <tlhelp32.h>
#include <stdio.h>
```

```
int main() {
```

```
    const char *targetProcName= "lsass.exe";
    int processID = 0;
```

```
    // snapshot of all processes in the system
```

```
    HANDLE processSnapshot = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);
```

```
    // initializing process entry structure
    PROCESSENTRY32 processEntry;
    processEntry.dwSize = sizeof(PROCESSENTRY32);
```

```
    // info about first process encountered in a system snapshot
    BOOL operationResult = Process32First(processSnapshot, &processEntry);
```

```
    printf("[*] Scanning for process: %s\n", targetProcName);
```

```
    // retrieve information about the processes
    while (operationResult) {
```

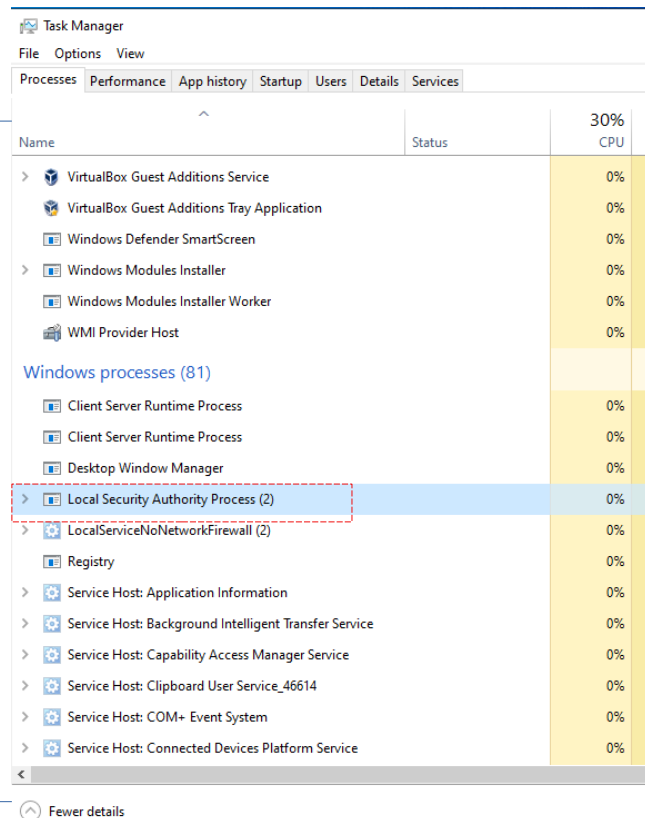
```
        // if we find the process: return process ID
        if (strcmp(targetProcName, processEntry.szExeFile) == 0) {
            printf("[+] Match found: %s (PID: %lu)\n", processEntry.szExeFile, processEntry.th32ProcessID);
            processID = processEntry.th32ProcessID;
            break;
        }
```

```
        operationResult = Process32Next(processSnapshot, &processEntry);
    }
```

```
    // closes an open handle (CreateToolhelp32Snapshot)
    CloseHandle(processSnapshot);
```

```
    return processID;
```

```
}
```



Program code: get the process id of lsass.exe

```
#include <windows.h>
#include <tlhelp32.h>
#include <stdio.h>

int main() {

    const char *targetProcName= "lsass.exe";
    int processID = 0;

    // snapshot of all processes in the system
    HANDLE processSnapshot = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);

    // initializing process entry structure
    PROCESSENTRY32 processEntry;
    processEntry.dwSize = sizeof(PROCESSENTRY32);

    // info about first process encountered in a system snapshot
    BOOL operationResult = Process32First(processSnapshot, &processEntry);

    printf("[*] Scanning for process: %s\n", targetProcName);

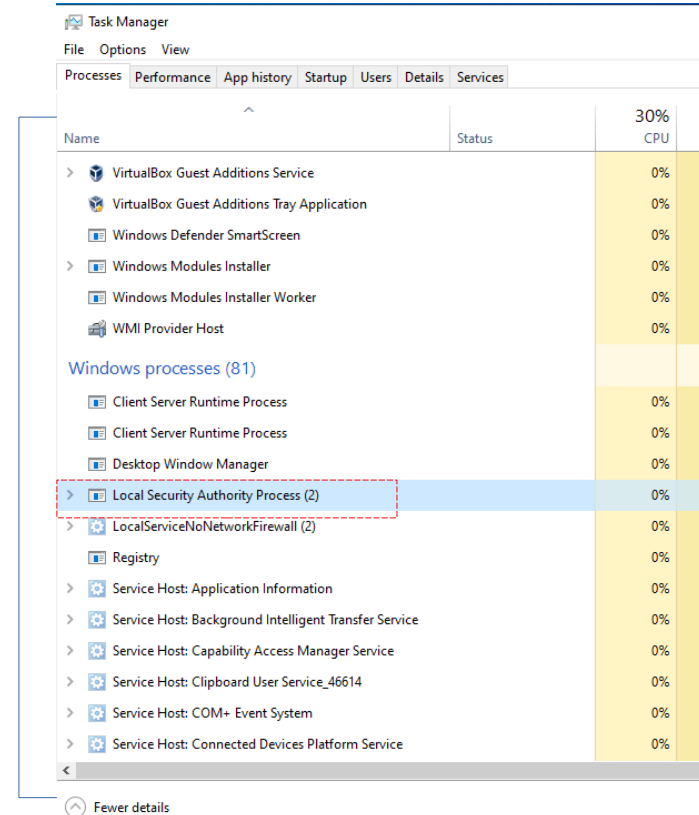
    // retrieve information about the processes
    while (operationResult) {

        // if we find the process: return process ID
        if (strcmp(targetProcName, processEntry.szExeFile) == 0) {
            printf("[+] Match found: %s (PID: %lu)\n", processEntry.szExeFile, processEntry.th32ProcessID);
            processID = processEntry.th32ProcessID;
            break;
        }

        operationResult = Process32Next(processSnapshot, &processEntry);
    }

    // closes an open handle (CreateToolhelp32Snapshot)
    CloseHandle(processSnapshot);

    return processID;
}
```



Program code: get the process id of lsass.exe

```
#include <windows.h>
#include <tlhelp32.h>
#include <stdio.h>

int main() {

    const char *targetProcName= "lsass.exe";
    int processID = 0;

    // snapshot of all processes in the system
    HANDLE processSnapshot = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);

    // initializing process entry structure
    PROCESSENTRY32 processEntry;
    processEntry.dwSize = sizeof(PROCESSENTRY32);

    // info about first process encountered in a system snapshot
    BOOL operationResult = Process32First(processSnapshot, &processEntry);

    printf("[*] Scanning for process: %s\n", targetProcName);

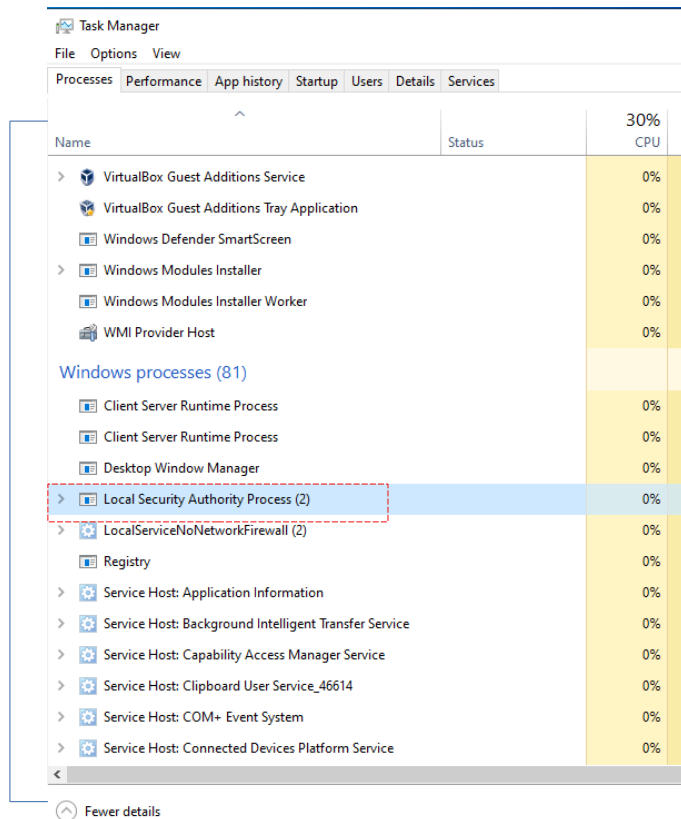
    // retrieve information about the processes
    while (operationResult) {

        // if we find the process: return process ID
        if (strcmp(targetProcName, processEntry.szExeFile) == 0) {
            printf("[+] Match found: %s (PID: %lu)\n", processEntry.szExeFile, processEntry.th32ProcessID);
            processID = processEntry.th32ProcessID;
            break;
        }

        operationResult = Process32Next(processSnapshot, &processEntry);
    }

    // closes an open handle (CreateToolhelp32Snapshot)
    CloseHandle(processSnapshot);

    return processID;
}
```



Program code: get the process id of lsass.exe

```
#include <windows.h>
#include <tlhelp32.h>
#include <stdio.h>
```

```
int main() {
```

```
    const char *targetProcName= "lsass.exe";
    int processID = 0;
```

```
    // snapshot of all processes in the system
```

```
    HANDLE processSnapshot = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);
```

```
    // initializing process entry structure
```

```
    PROCESSENTRY32 processEntry;
```

```
    processEntry.dwSize = sizeof(PROCESSENTRY32);
```

```
    // info about first process encountered in a system snapshot
```

```
    BOOL operationResult = Process32First(processSnapshot, &processEntry);
```

```
    printf("[*] Scanning for process: %s\n", targetProcName);
```

```
    // retrieve information about the processes
```

```
    while (operationResult) {
```

```
        // if we find the process: return process ID
```

```
        if (strcmp(targetProcName, processEntry.szExeFile) == 0) {
```

```
            printf("[+] Match found: %s (PID: %lu)\n", processEntry.szExeFile, processEntry.th32ProcessID);
```

```
            processID = processEntry.th32ProcessID;
```

```
            break;
```

```
        }
```

```
        operationResult = Process32Next(processSnapshot, &processEntry);
```

```
    }
```

```
    // closes an open handle (CreateToolhelp32Snapshot)
```

```
    CloseHandle(processSnapshot);
```

```
    return processID;
```

```
}
```

