# цель работы

Изучение архитектуры подсистемы безопасности и механизмов контроля доступа ОС Windows

# ход работы

На я зыке Си была реализована программа, получающая следующую информацию о процессах:

* имя процесса;
* описание процесса (Description);
* идентификатор процесса (PID);
* путь до исполняемого файла;
* имя и идентификатор родительского процесса;
* имя пользователя владельца процесса, SID;
* тип процесса (32-bit / 64-bit);
* среда исполнения (нативный код / CLR .NET);
* использование DEP, ASLR;
* список используемых динамических библиотек (DLL);
* список контроля доступа (ACL);
* владелец файла;
* уровень целостности;

Для получения всех ID процессов был использован макрос “EnumProcesses”, который в массив “aProcesses”, помещает ID процессов (рисунок 1)

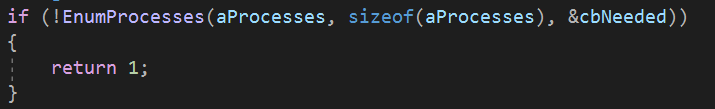


Рисунок 1 – Получение ID всех процессов

После этого для получения информации о процессе была использована функция “OpenProcess” с определенными аргументами (рисунок 2)

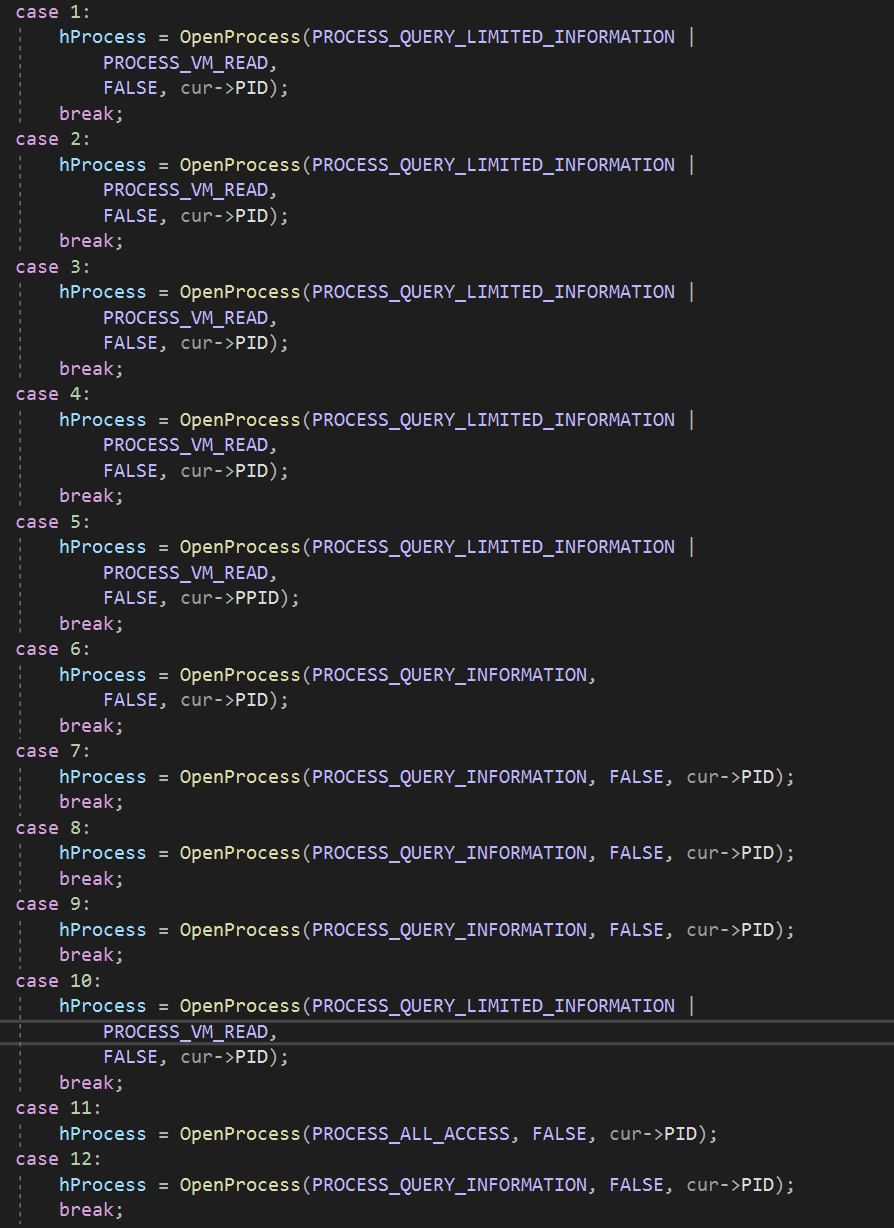


Рисунок 2- Пример аргументов

Для получения имени процесса использовался макрос “GetModuleBaseName”, который по дескриптору процесса передает в массив “ProcessName” имя процесса (Рисунок 3).



Рисунок 3 – Получение имени процесса

Для получения исполняемого файла использовался макрос “GetModuleFileNameEx”, который работает аналогично предыдущему, только передает в массив путь до исполняемого файла.

Для получения среды исполнения была использована функция “CLRCreateInstance” которая позволяет использовать один из трех интерфейсов, в данной работы был использован интерфейс “CLRMetaHost”. После этого был использован метод “GetVersionFromFile”, для получения информации об используемой среде исполняемого кода (рисунок 4)

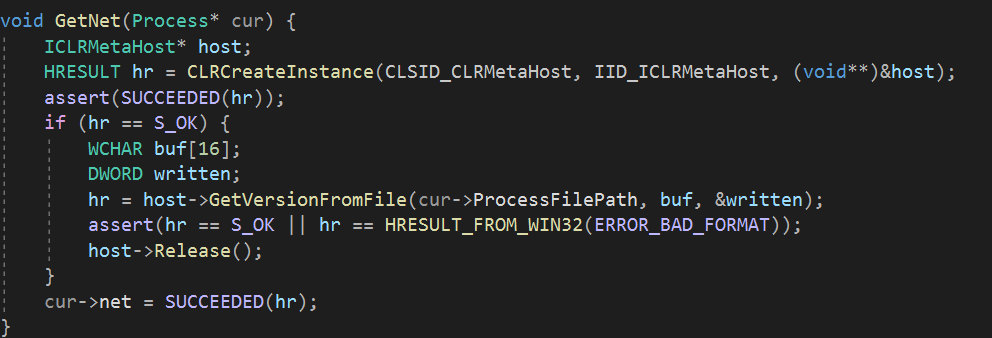


Рисунок 4 – Получение информации о среде исполняемого файла

Для получения описания процесса был использован макрос “VerQueryValue”, данный макрос возвращает указатель на начало интересующего поля в свойствах файла. Так как информация хранится блоками в таком виде: “\StringFileInfo\кодовая\_страница\имя\_параметра”, то сначала была получена кодовая страница, а уже затем нужный нам параметр (рисунок 5)

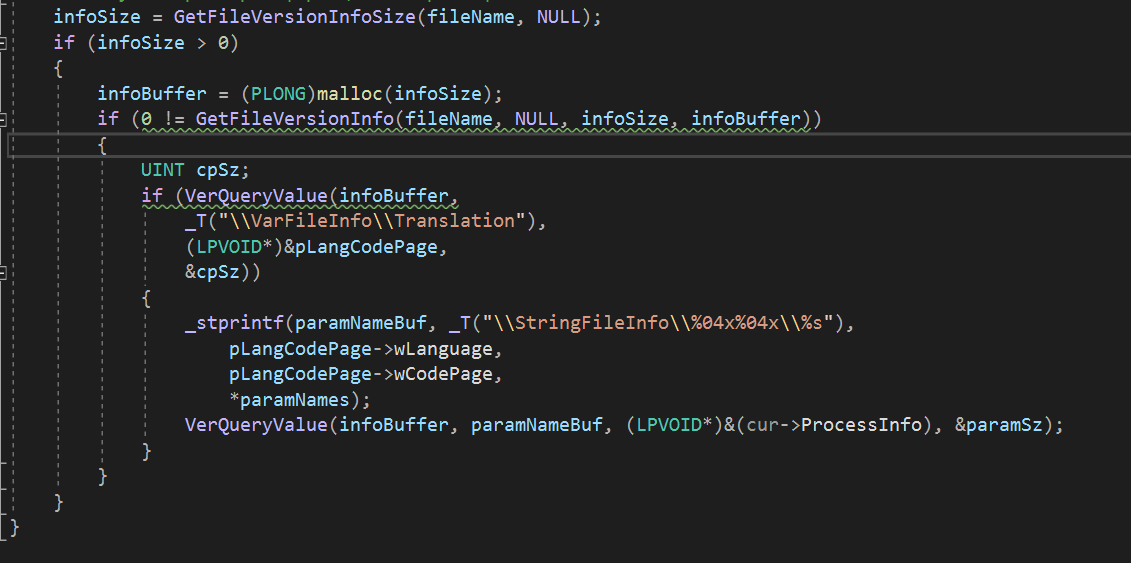


Рисунок 5 – Получение описание процесса

Для получения PPID, была использована функция “CreateToolhelp32Snapshot”. Данная функция, получает снапшот всех процессов в системе, с их модулями и нитями. Далее при помощи макроса “Process32First”, в случае успеха записывает в структуру “PROCESSENTRY32W”, о первом процесса. Для получения следующего процесса использовался макрос “Process32Next”. Таким перебирая все поля “th32ParentProcessID” находится PPID нужного процесса (рисунок 6)

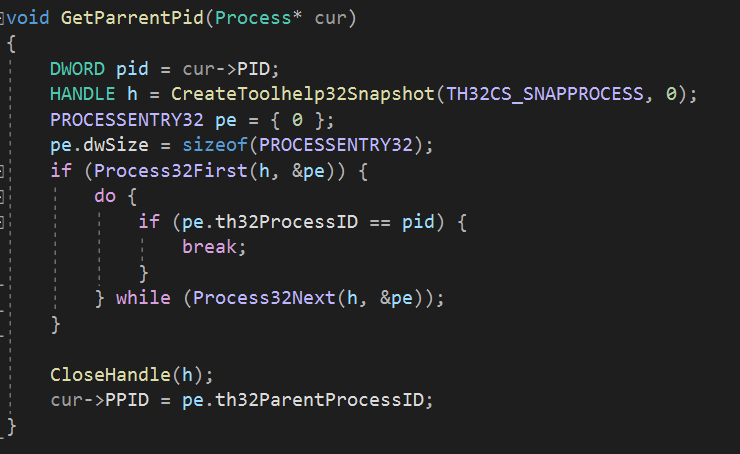


Рисунок 6 – Получение PPID

Для получения типа процесса использовалась функция “fnIsWow64Process”. (рисунок 7)

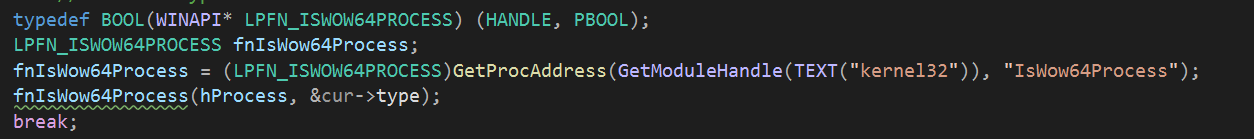


Рисунок 7 – Получение типа процесса

Для получения информации об использовании DEP и ASLR использовалась функция “GetProcessMitigationPolicy” (Рисунок 8)

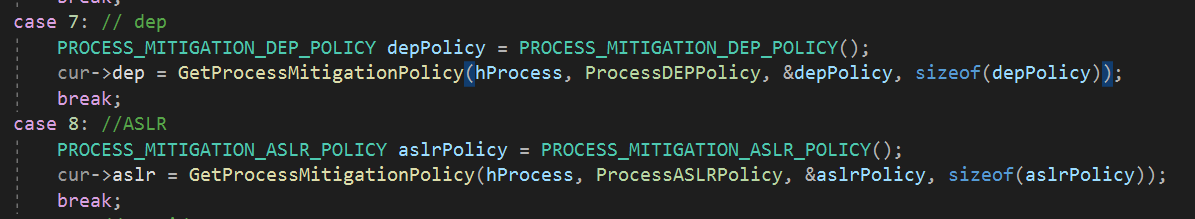


Рисунок 8 – Получение DEP и ASLR

Для получения имени владельца процесса и его SID был использован макрос “LookupAccountSid”, который по токену процесса получает нужную информацию. Перед вызовом данного макроса были использованы еще две функции: “OpenProcessToken” и “GetTokenInformation”. Первая по Handle’у процесса возвращает токен, а вторая извлекает определенного типа информации о маркере доступа. (Рисунок 9)

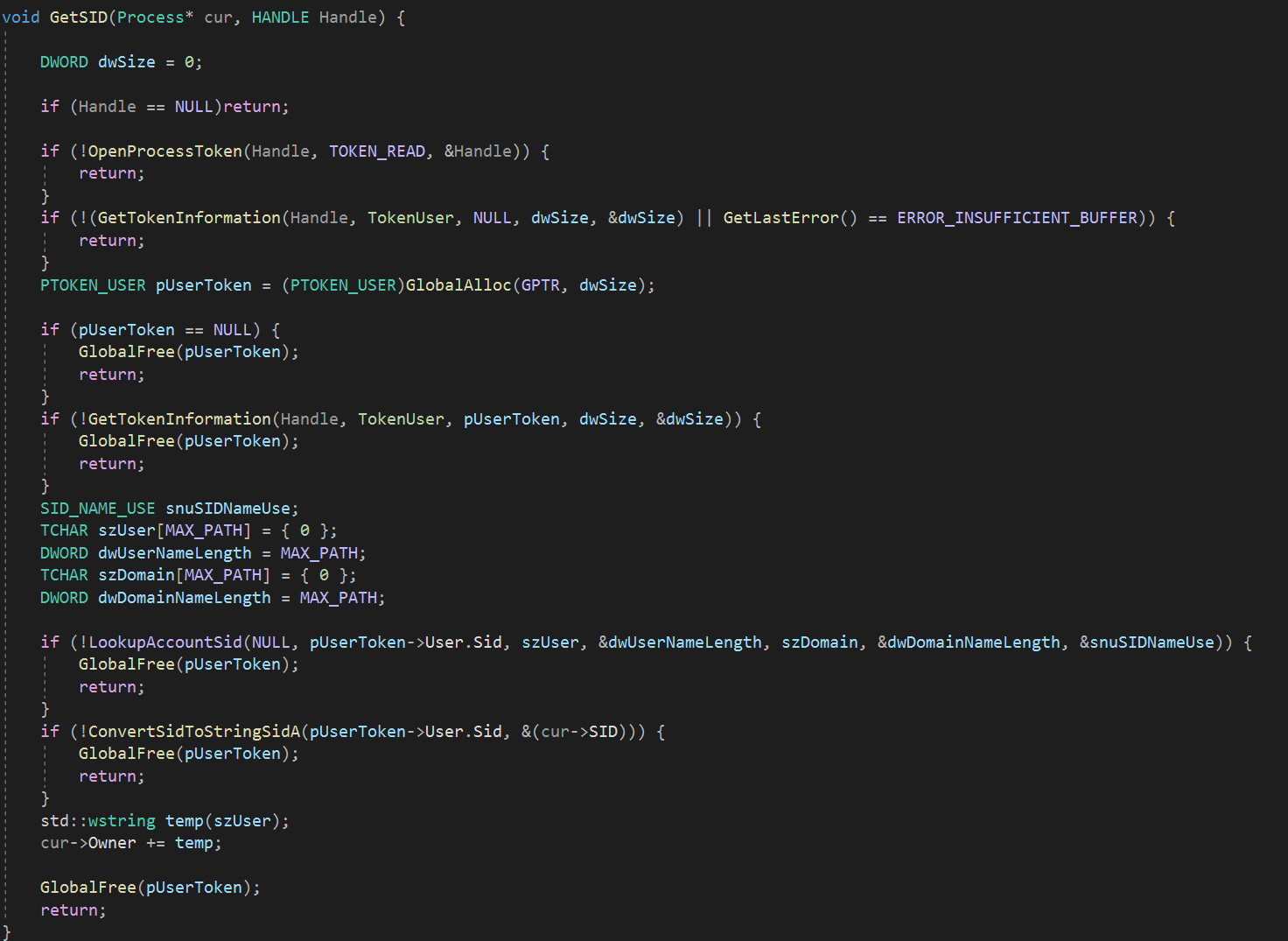


Рисунок 9 – Получение имени владельца процесса и его SID

Для получения уровня целостности была использована функция “GetSidSubAuthority”, которая по токену процесса получает нужную информацию. Так же использовались функции OpenProcessToken”, “GetTokenInformation” из прошлого пункта. (рисунок 10)

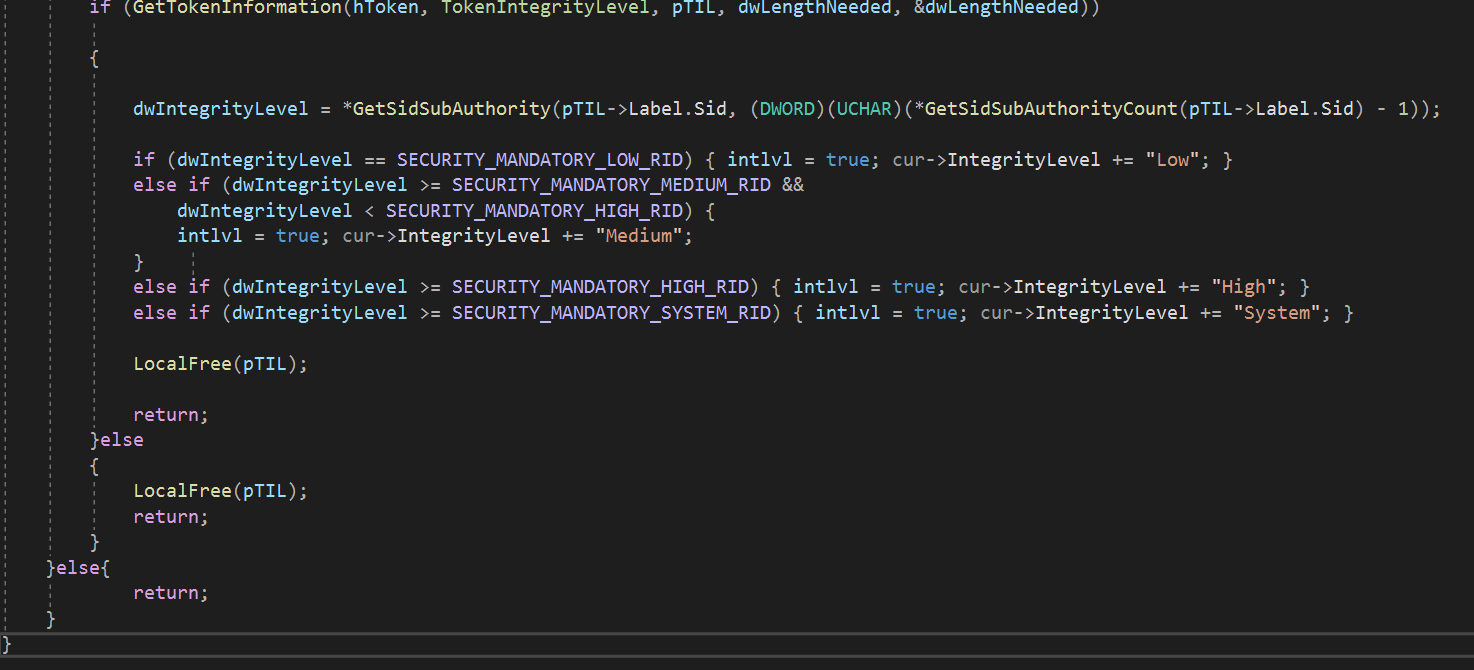


Рисунок 10 – Получение уровня целостности

Для получения привилегий процесса была использована функция “ LookupPrivilegeNameA”, которая по токену процесса получает все его привилегии. (рисунок 11)

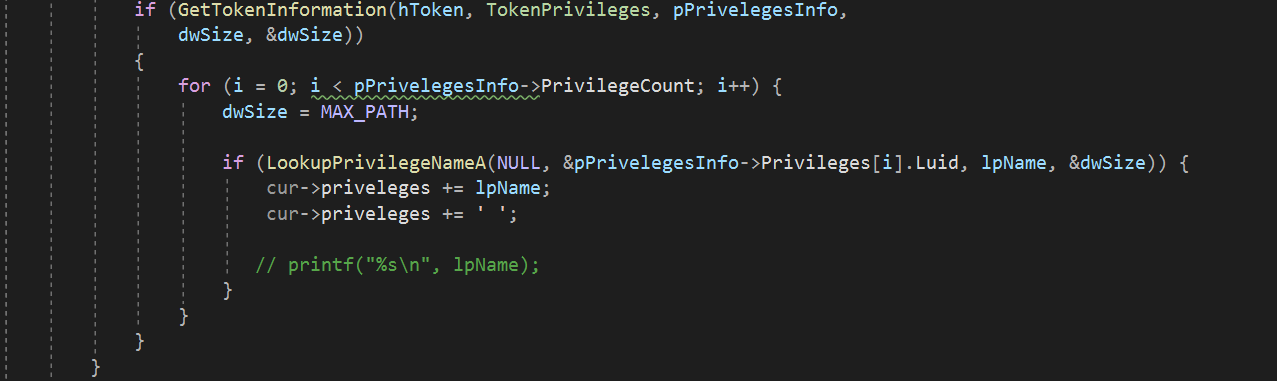


Рисунок 11 – Получение привилегией процесса

Для получения имени владельца файла и его SID были использованы функции “GetNamedSecurityInfoA” и “LookupAccountSidA”. Первая функция при помощи пути до выполняемого файла получает SID владельца файла, а вторая при помощи SID’a возвращает его имя (рисунок 12).

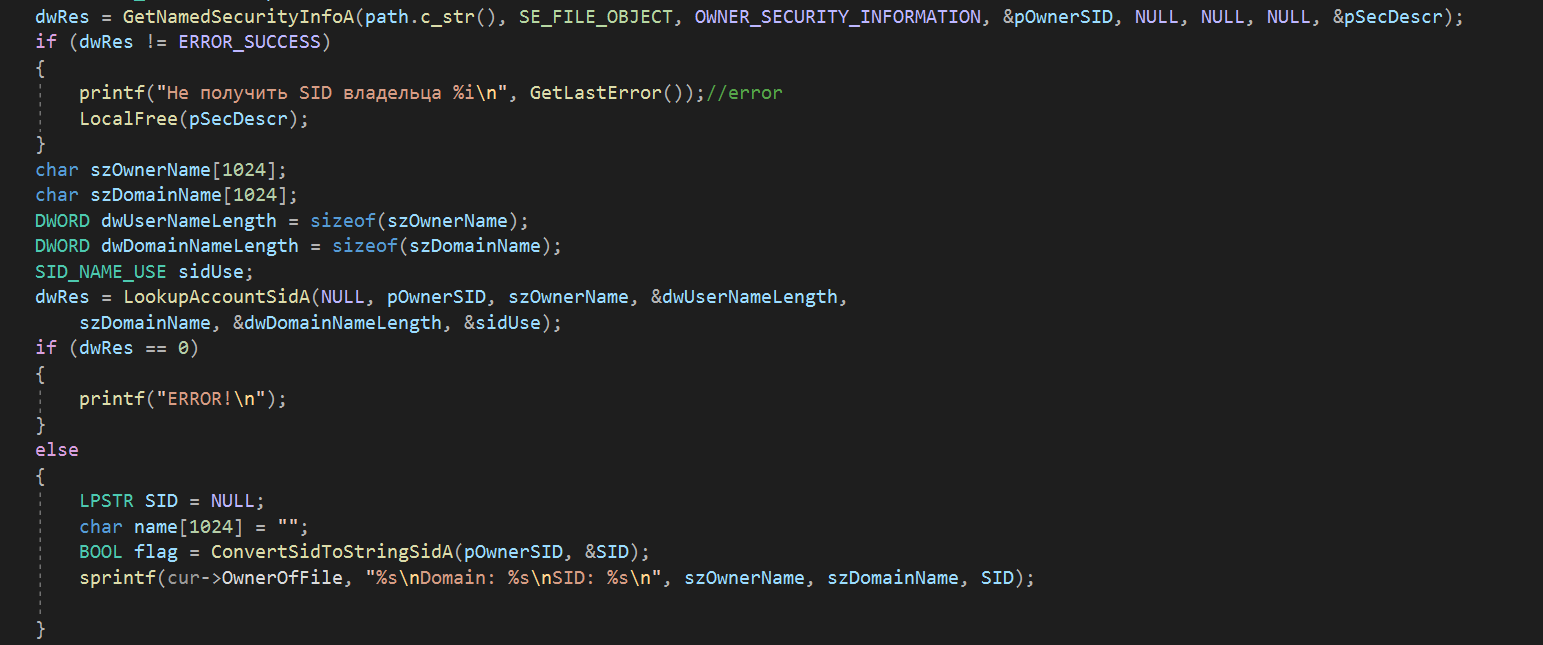


Рисунок 12 – Получения имени владельца файла и его SID

Для получения списка контроля доступа были использованы следующие функции: “ GetNamedSecurityInfoA”, “GetAclInformation”, “GetAce” и “ LookupAccountSidA”. Первая функция возвращает ссылку на DACL указанного файла. При помощи этой ссылки функция “GetAclInformation”, возвращает информацию о списке контроля доступа указанного файла. Функция “GetAce” возвращает указатель на адрес ACE нужного файла. И последняя функция возвращает имя используя SID текущего ACE (рисунок 13)

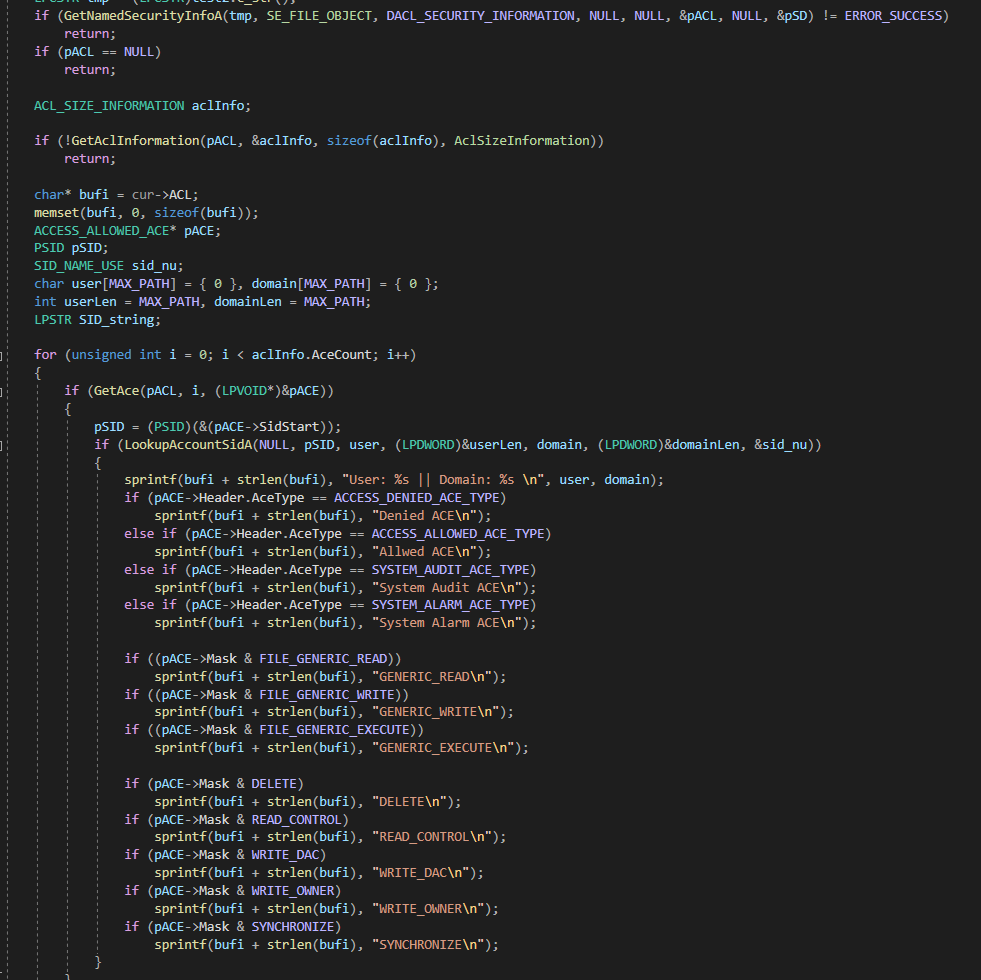


Рисунок 13 – Получение ACL

Для получения уровня целостности исполняемого файла была использована функций “GetNamedSecurityInfoW” и “ GetAce”. Первая функция возвращает ссылку на SACL указанного файла, а вторая при помощи SACL’a возвращает sid, в котором хранится уровень целостности (рисунок 14)



Рисунок 14 – Получение уровня целостности

Для получения используемых dll файлов была использована функция “CreateToolhelp32Snapshot ” с аргументом “TH32CS\_SNAPMODULE”. Данная функция создает снапшот заданного процесса со всеми его модулями. Далее при помощи макроса “Module32Next”, функция проходит по всем модулям и из каждого модуля достает используемую dll (рисунок 15)

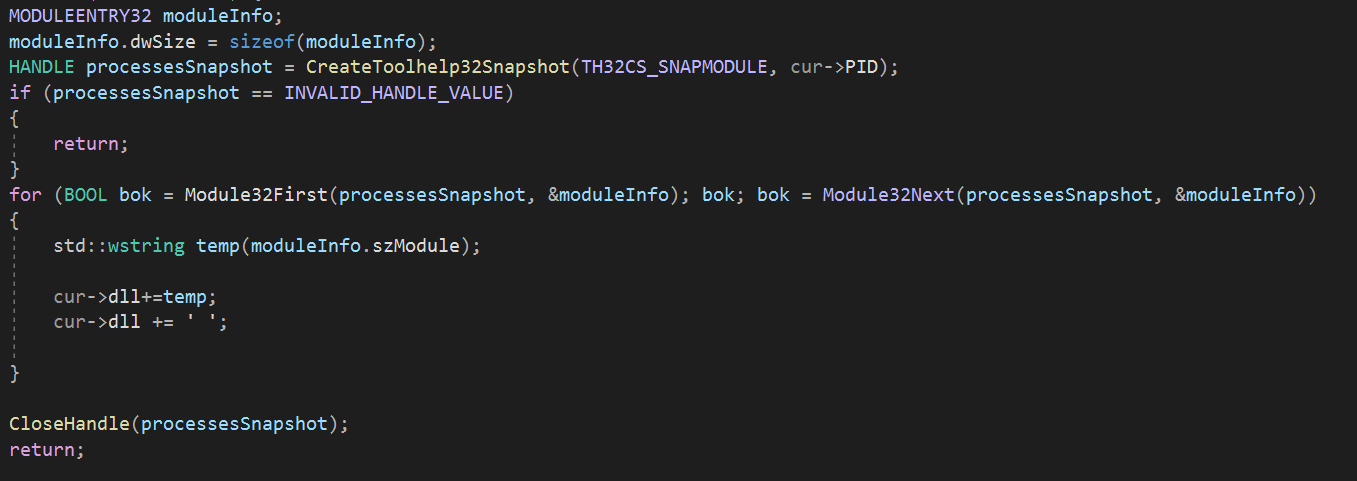


Рисунок 15 – Получение используемых dll

Вся полученная информация выводится в консоль и .JSON файл (рисунки 16-17)

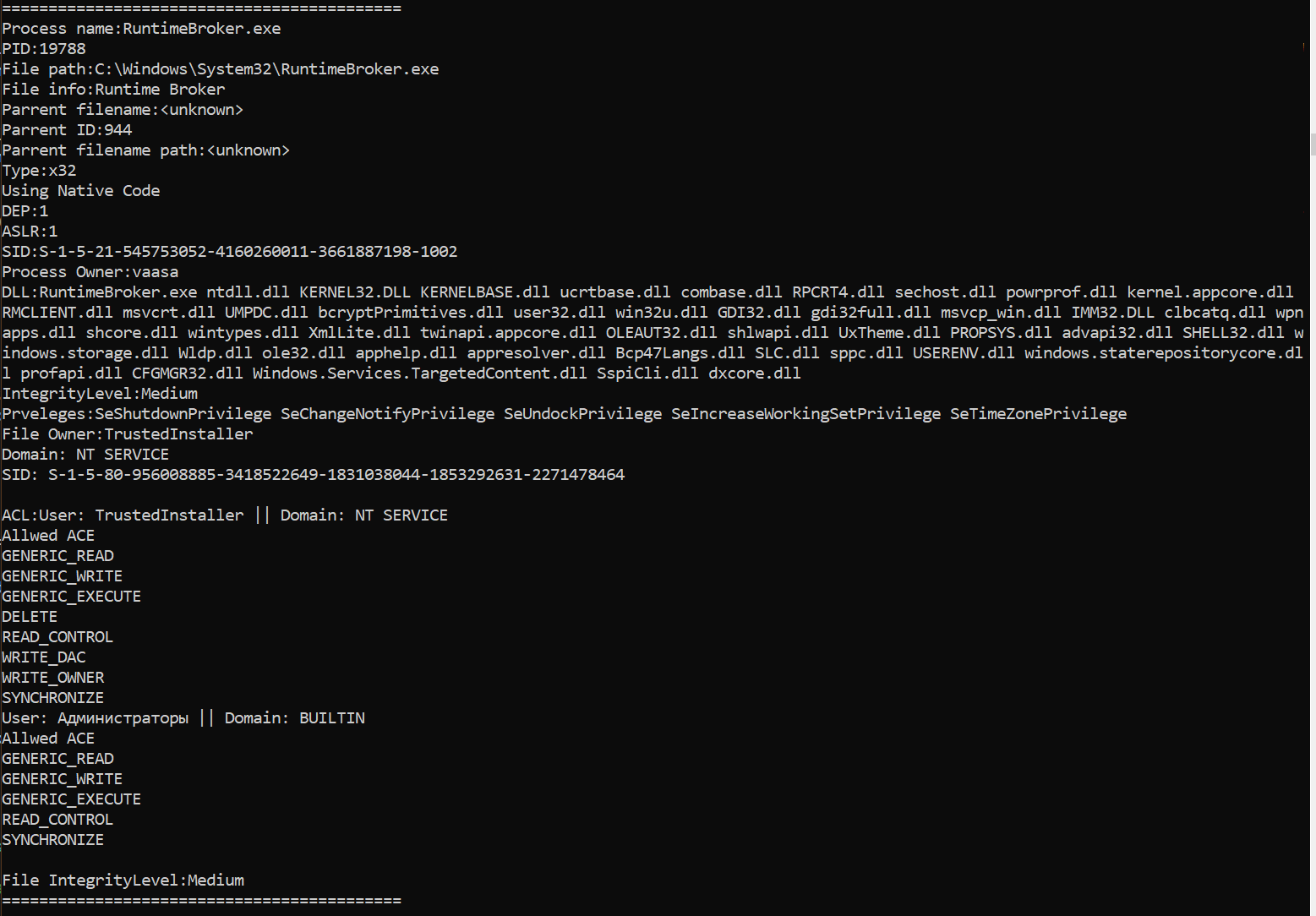


Рисунок 16 – Вывод в консоль

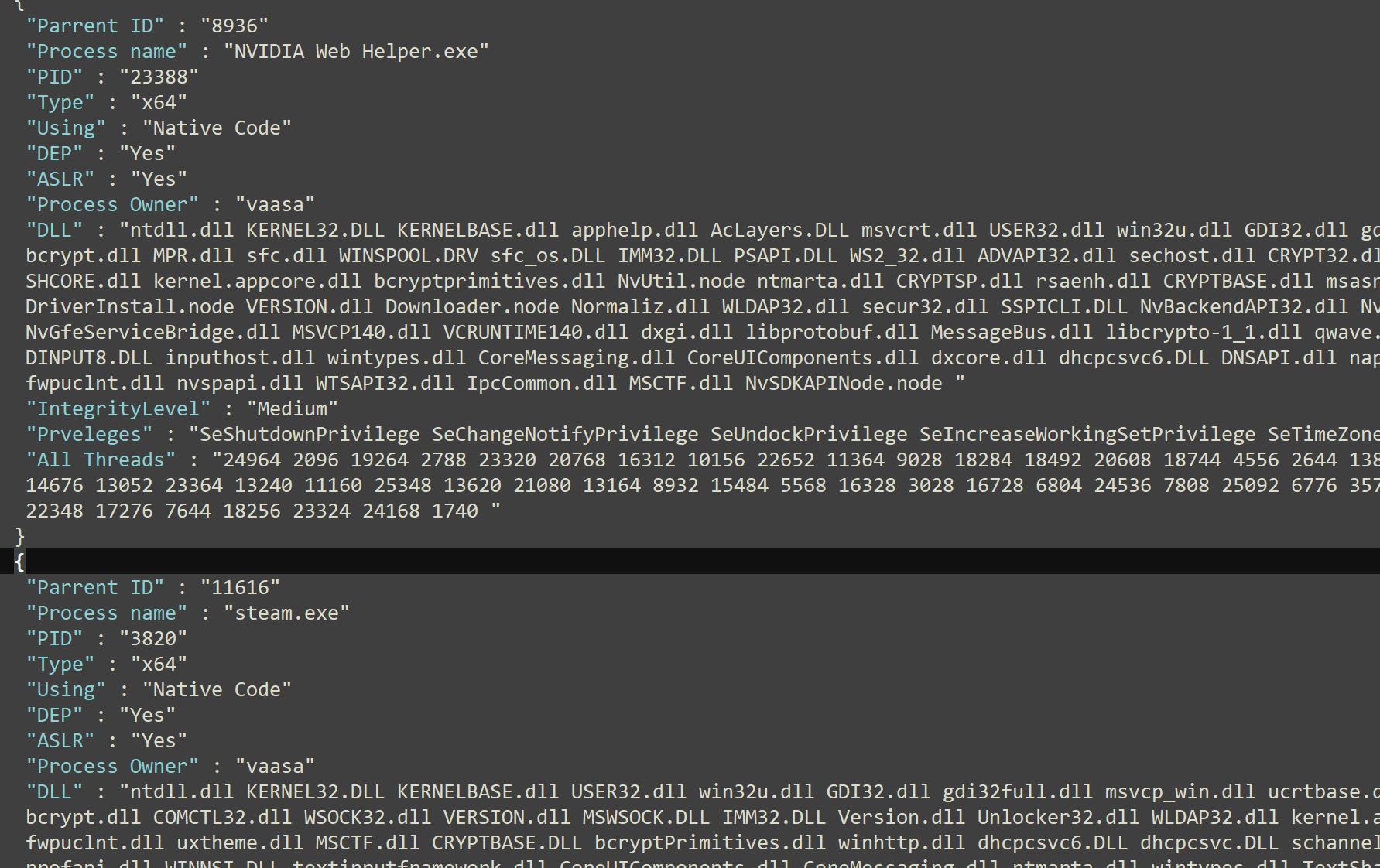


Рисунок 17 – Вывод в json файл

# Дополнительное задание

Для данной лабораторной работы дополнительное задание выглядело следующим образом: “ Реализовать вывод потоков процесса”

Для этого была реализована функция “GetThreads” (Рисунок 18)



Рисунок 18 – Реализованная функция

Данная функция первым делом создает снапшот всех запущенных потоков. После этого идет перебор всех потоков, и если в каком-либо потоке владельцем потока является текущий процесс, то он добавляется в список потоков процесса.