Министерство науки и высшего образования

Пензенский Государственный Университет

ОТЧЕТ

по лабораторной работе №5

по курсу «Защита информации»

на тему «**Атрибуты безопасности в ОС Windows**»

Вариант 1

Выполнили студенты группы 18ВВ1:

Колобов К.А.

Амплеев Д.О.

Приняли:

Дубравин А.В.

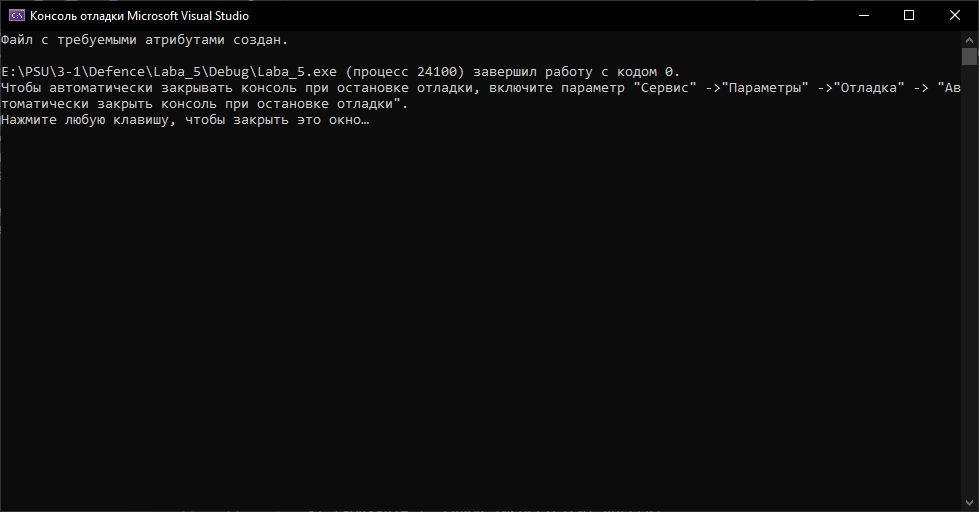
2020

**Лабораторное задание – вариант№1:**

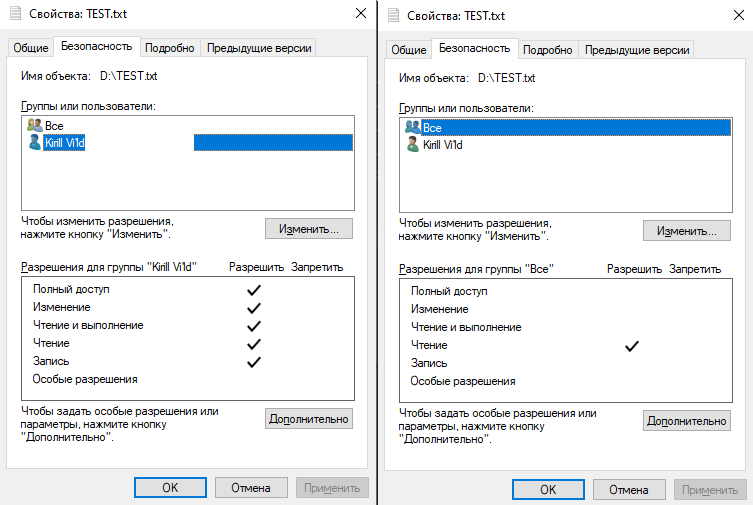
Разработать программу, выполняющую создание файла с правами на полный доступ к файлу для текущего пользователя и запретом записи для группы пользователей «Все».

**Результат работы программы:**

1. При успешном выполнении программа уведомляет о создании файла.



1. Атрибуты созданного файла



**Листинг:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include<iostream>

#include<fstream>

#include<windows.h>

#include<conio.h>

#include <aclapi.h>

int GetCurrentUserSid(PSID\* pSID)

{

const int MAX\_NAME = 256;

DWORD i, dwSize = 0;

HANDLE hToken;

PTOKEN\_USER user;

TOKEN\_INFORMATION\_CLASS TokenClass = TokenUser;

if (!OpenProcessToken(GetCurrentProcess(), TOKEN\_READ | TOKEN\_QUERY, &hToken))

return GetLastError();

if (!GetTokenInformation(hToken, TokenClass, NULL, 0, &dwSize))

{

DWORD dwResult = GetLastError();

if (dwResult != ERROR\_INSUFFICIENT\_BUFFER)

{

return FALSE;

}

}

user = (PTOKEN\_USER)LocalAlloc(GPTR, dwSize);

if (!GetTokenInformation(hToken, TokenClass, user, dwSize, &dwSize))

{

return FALSE;

}

DWORD dw\_sid\_len = GetLengthSid(user->User.Sid);

\*pSID = (SID\*)LocalAlloc(GPTR, dw\_sid\_len);

CopySid(dw\_sid\_len, \*pSID, user->User.Sid);

return 0;

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

HANDLE hFile;

EXPLICIT\_ACCESS EA[2];

PSID EveryoneSID = NULL, CurrentUserSID = NULL;

SID\_IDENTIFIER\_AUTHORITY SIDAuthWorld = SECURITY\_WORLD\_SID\_AUTHORITY;

SID\_IDENTIFIER\_AUTHORITY SIDAuthCU = SECURITY\_NT\_AUTHORITY;

PACL pACL = NULL;

DWORD dwRes, dwDisposition;

PSECURITY\_DESCRIPTOR pSD = NULL;

SECURITY\_ATTRIBUTES SA;

LONG lRes;

HKEY hkSub = NULL;

ZeroMemory(&EA, 2 \* sizeof(EXPLICIT\_ACCESS));

AllocateAndInitializeSid(&SIDAuthWorld, 1, SECURITY\_WORLD\_RID, 0, 0, 0, 0, 0, 0, 0, &EveryoneSID);

EA[0].grfAccessPermissions = GENERIC\_READ;

EA[0].grfAccessMode = SET\_ACCESS;

EA[0].grfInheritance = NO\_INHERITANCE;

EA[0].Trustee.TrusteeForm = TRUSTEE\_IS\_SID;

EA[0].Trustee.ptstrName = (LPTSTR)EveryoneSID;

GetCurrentUserSid(&CurrentUserSID);

EA[1].grfAccessPermissions = GENERIC\_ALL;

EA[1].grfAccessMode = SET\_ACCESS;

EA[1].grfInheritance = NO\_INHERITANCE;

EA[1].Trustee.TrusteeForm = TRUSTEE\_IS\_SID;

EA[1].Trustee.ptstrName = (LPTSTR)CurrentUserSID;

dwRes = SetEntriesInAcl(2, EA, NULL, &pACL);

pSD = (PSECURITY\_DESCRIPTOR)LocalAlloc(LPTR, SECURITY\_DESCRIPTOR\_MIN\_LENGTH);

InitializeSecurityDescriptor(pSD, SECURITY\_DESCRIPTOR\_REVISION);

if (!SetSecurityDescriptorDacl(pSD, TRUE, pACL, FALSE))

std::cout << "SetSecurityDescriptorDacl Error" << GetLastError() << std::endl;

SA.nLength = sizeof(SECURITY\_ATTRIBUTES);

SA.lpSecurityDescriptor = pSD;

SA.bInheritHandle = FALSE;

lRes = RegCreateKeyEx(HKEY\_CURRENT\_USER, L"Key", 0, 0, 0, KEY\_ALL\_ACCESS, &SA, &hkSub, &dwDisposition);

hFile = CreateFile(L"D:\\TEST.txt", GENERIC\_READ, 0, &SA, CREATE\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, NULL);

if (!hFile)

std::cout << "Не удалось создать файл." << std::endl;

else

std::cout << "Файл с требуемыми атрибутами создан." << std::endl;

Cleanup:

if (EveryoneSID)

FreeSid(EveryoneSID);

if (CurrentUserSID)

FreeSid(CurrentUserSID);

if (pACL)

LocalFree(pACL);

if (pSD)

LocalFree(pSD);

if (hkSub)

RegCloseKey(hkSub);

return 0;

}