Название:	Чужая память
Категория:	Реверс-инжиниринг
Уровень:	Средняя
Очки:	400
Описание:	Чужая память, как мёд, всегда "вкуснее"
Теги:	Антиотладка, Внедрение данный в процесс
Автор:	ROP

Прохождение:

Смотрим, что делает исполняемый файл через декомпилятор.

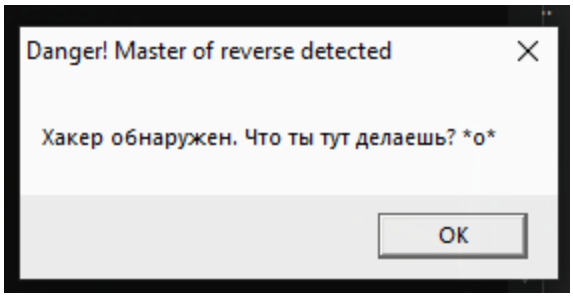
```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  PROCESSENTRY32 pe; // [rsp+40h] [rbp-40h] BYREF
  LPVOID lpBaseAddress; // [rsp+170h] [rbp+F0h]
  HANDLE hProcess; // [rsp+178h] [rbp+F8h]
  HANDLE hSnapshot; // [rsp+180h] [rbp+100h]
  unsigned int i; // [rsp+188h] [rbp+108h]
  int v9; // [rsp+18Ch] [rbp+10Ch]

  sub_401970(argc, argv, envp);
  hSnapshot = CreateToolhelp32Snapshot(2u, 0);
```

```
  if ( hSnapshot == (HANDLE)-1i64 )
    return 1;
  pe.dwSize = 304;
  v9 = 0;
  if ( Process32First(hSnapshot, &pe) )
  {
    while ( 1 )
    {
      if ( !strcmp(pe.szExeFile, "ida64.exe") )
      {
        MessageBoxA(0i64, &Text, "Danger! Master of reverse detected", 0);
        goto LABEL_11;
      }
      if ( !strcmp(pe.szExeFile, "ida.exe") )
      {
        MessageBoxA(0i64, &byte_405070, "Danger! Master of reverse detecte
d", 0);
        goto LABEL_11;
      }
      if ( !strcmp(pe.szExeFile, "notepad.exe") )
        break;
      if ( !Process32Next(hSnapshot, &pe) )
        goto LABEL_11;
    }
    v9 = 1;
  }
LABEL_11:
  CloseHandle(hSnapshot);
  if ( !v9 )
    return 1;
  hProcess = OpenProcess(0x1F0FFFu, 0, pe.th32ProcessID);
  if ( !hProcess )
    return 1;
  lpBaseAddress = VirtualAllocEx(hProcess, 0i64, 4ui64, 0x3000u, 0x40u);
  if ( lpBaseAddress )
  {
```

```
  for ( i = 0; i <= 0x5C; ++i )
    byte_404020[i] -= 48;
  if ( WriteProcessMemory(hProcess, lpBaseAddress, byte_404020, 0x5Dui6
4, 0i64)
    && CreateRemoteThread(hProcess, 0i64, 0i64, (LPTHREAD_START_ROUTI
NE)lpBaseAddress, 0i64, 0, 0i64) )
  {
    VirtualFreeEx(hProcess, lpBaseAddress, 0i64, 0x8000u);
    CloseHandle(hProcess);
    return 0;
  }
  else
  {
    VirtualFreeEx(hProcess, lpBaseAddress, 0i64, 0x8000u);
    CloseHandle(hProcess);
    return 1;
  }
 }
 else
 {
  CloseHandle(hProcess);
  return 1;
 }
}
```

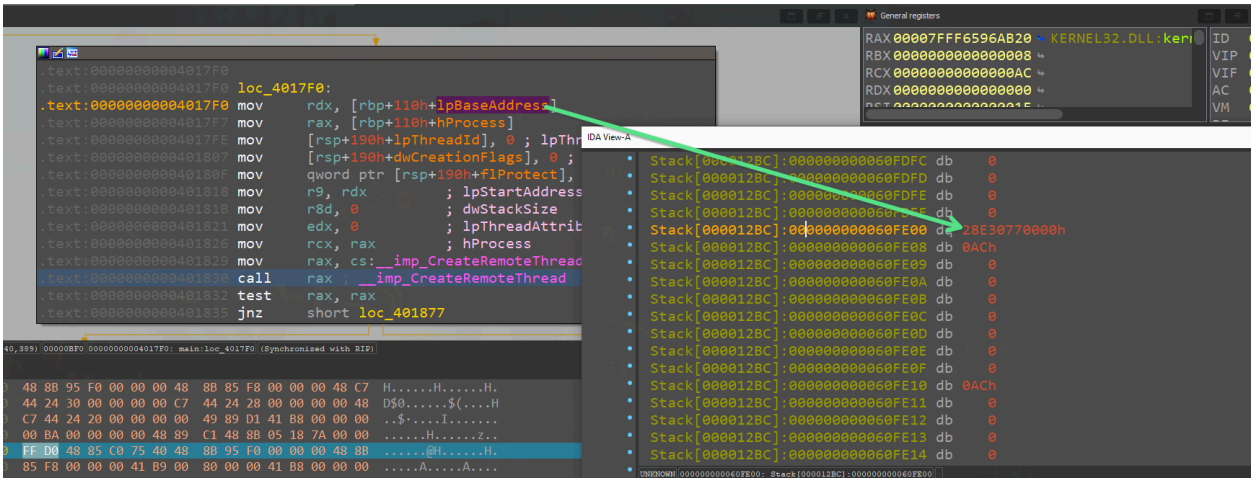
Данный кодп "перебирает" запущенные процессы и ищет `notepad.exe`. Затем он выделяет `0x8000` байт памяти в этом процессе и записывает дешифрованнй массив `byte_404020` в неё. В конце создаётся поток, который начинает выполняться с этой памяти.

Если запущены процессы `ida64.exe` или `ida.exe` выводятся предупреждения и программа завершается.



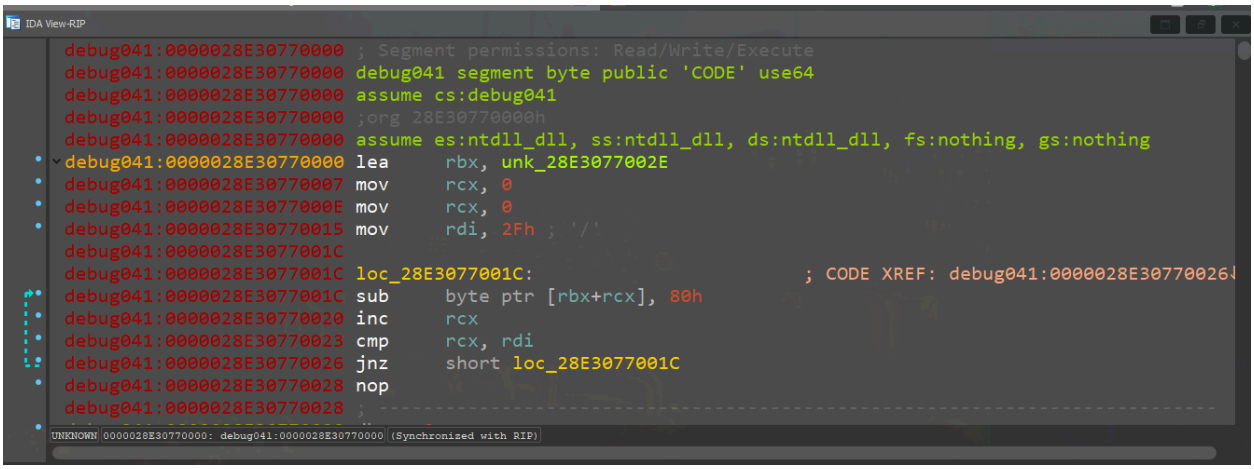
Проверки можно обойти, пропатчив имя процессов `ida64.exe` и `ida.exe` на другое.

Затем попадаем на код, что запустит поток в `notepad.exe`.

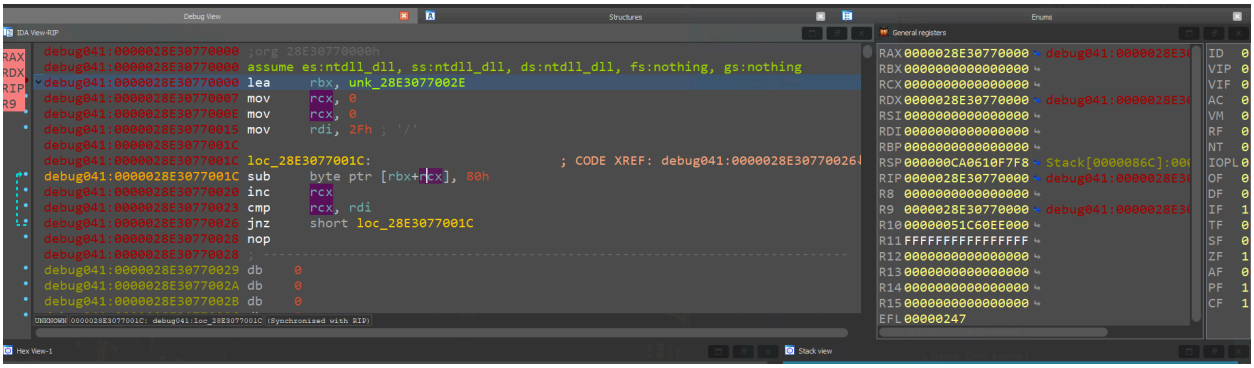


В `lpBaseAddress` будет адрес нужной памяти в `notepad.exe`.

Присоединяемся к процессу и переходим по нужному адресу.



Представляем байты как код, создаём точку останова на первой инструкции, выполняем процесс далее и выполняем код `task.exe` далее.



Выполняем весь этот код.

Далее будет дешифрованный флаг.