Александр Эккерт (stannic.man@gmail.com)

КОВЫРЯЕМ БРОНЮ WINDOWS

ВЫЯСНЯЕМ, ЧТО ТАКОЕ ACL/DACL И КАК ЭТО МОЖНО ЗАЭКСПЛОИТИТЬ

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Сегодня мы поговорим о том, что несет в себе система контроля доступа в ОС Windows. Оговорюсь сразу, что данный вопрос немножко необычен для изучения, — вроде бы известно, о чем речь, однако на поверку выходит, что написано про нее довольно мало. Впрочем, для нас это не преграда — попробуем рассмотреть доспехи Windows изнутри и порассуждать о возможных способах их обхода.

ВВЕДЕНИЕ

Если присмотреться внимательнее, то ничего сверхсложного в системе контроля доступа ОС Windows нет. Как и в других операционных системах, главное, что нужно усвоить, — это кто и что может делать в операционной системе.

Система разграничения прав пользователей хорошо известна любителям *nix-like систем, тогда как в ОС Windows она довольно прозрачна (см. рисунок 1) и поэтому не столь заметна для обычного пользователя, исключая грамотных сисадминов, которые напрямую с этим связаны.

Поэтому для обеспечения контроля за операциями над неким абстрактным объектом в системе Windows должна быть уверена в правильности идентификации каждого пользователя. Именно по этой причине Windows требует от пользователя входа с аутентификацией прежде, чем ему будет позволено обращаться к системным ресурсам.

В целом система проверки прав доступа выглядит так: процесс запрашивает описатель объекта (об этом подробнее ниже), а уж потом диспетчер объектов и система защиты решают, можно ли этому процессу предоставить описатель, разрешающий доступ к объекту.

Модель контроля доступа в ОС Windows требует, чтобы процесс заранее — еще до открытия абстрактного объекта — указывал, какие операции он собирается выполнять над этим объектом. Ну типа, представь, Великая Отечественная, в хату вежливо стучатся фашисты с автоматами и говорят: «Это мы, бабка, фашисты, мы хотим отобрать у тебя хлеб, яйки, сметана и спросить, где партизаны».

В свою очередь, система («бабка») проверяет тип доступа, запрошенный процессом, и, если такой доступ разрешен, процесс получит описатель, который позволит ему (фашистам) выполнить операции над объектом.

Таким alarm-событием для системы, к примеру, является открытие объекта по его имени вызовом kernel-функции

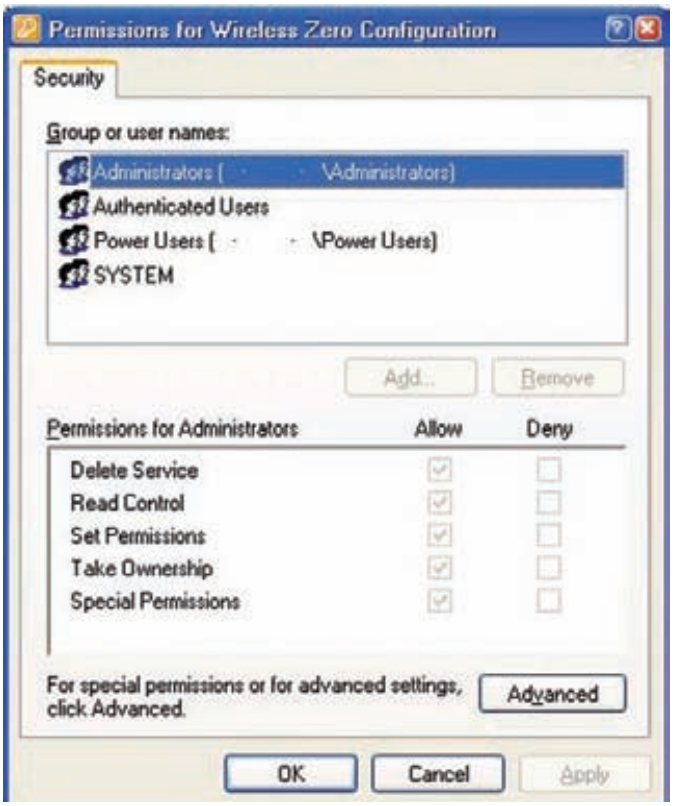

Рис.1. Хорошо знакомая сисадминам картинка

nt!ObOpenObjectByName. При вызове этой функции диспетчер объектов ищет его в своем пространстве имен. Не будем описывать сейчас то, что происходит при этом процессе, — долго, муторно и непонятно.

Все в итоге сводится к тому, что система вызывает «дьявола». Хотя нет, вру, на самом деле следует вызов функций nt!ObCheckObjectAccess → nt!SeAccessCheck (функция AccessCheck для пользовательского режима). Эта функция, наверное, является одной из ключевых для всей модели защиты ОС Windows (впрочем, как и другие Se*-функции). Она принимает параметры защиты объекта, идентификационные данные защиты процесса и запрашиваемый тип доступа и в зависимости от результата «рассмотрения» вернет TRUE или FALSE.

Но это так, самое общее и приблизительное описание лишь основного момента защиты. На самом деле процесс проверки доступа, прав и привилегий очень сложен.

ЭЛЕМЕНТЫ БРОНИ

К основным элементам модели доступа Windows можно отнести идентификаторы защиты — SID и маркеры доступа (так называемые токены).

SID — это идентификатор защиты, который Windows присваивает пользователям системы, локальным и доменным группам, локальным компьютерам, доменам и членам доменов. SID — это числовое значение переменной длины, которое ты наверняка не раз встречал, выглядит оно примерно так: S-1-5-21-12345678910-12345678910-12345678910-1228.

Например, S-1-1-0 означает группу, объединяющую всех пользователей. Группа S-1-2-0 объединяет пользователей, которые регистрируются на терминалах, физически подключенных к системе.

Маркеры доступа, наверное, основной элемент защиты Windows. Он описывает контекст защиты процесса (потока) и содержит в себе информацию, описывающую привилегии, учетные

```
kd> dt _TOKEN
   +0x000 TokenSource : _TOKEN_SOURCE
   +0x010 TokenId : _LUID
   +0x018 AuthenticationId : _LUID
   +0x020 ParentTokenId : _LUID
   +0x028 ExpirationTime : _LARGE_INTEGER
   +0x030 TokenLock : Ptr32 _ERESOURCE
   +0x034 ModifiedId : _LUID
   +0x03c SessionId : Uint4B
   +0x040 UserAndGroupCount : Uint4B
   +0x044 RestrictedSidCount : Uint4B
   +0x048 PrivilegeCount : Uint4B
   +0x04c VariableLength : Uint4B
   +0x050 DynamicCharged : Uint4B
   +0x054 DynamicAvailable : Uint4B
   +0x058 DefaultOwnerIndex : Uint4B
   +0x05c UserAndGroups : Ptr32 _SID_AND_ATTRIBUTES
   +0x060 RestrictedSids : Ptr32 _SID_AND_ATTRIBUTES
   +0x064 PrimaryGroup : Ptr32 Void
   +0x068 Privileges : Ptr32 _LUID_AND_ATTRIBUTES
   +0x06c DynamicPart : Ptr32 Uint4B
   +0x070 DefaultDacl : Ptr32 _ACL
   +0x074 TokenType : _TOKEN_TYPE
```

Рис. 2. Наиболее интересные поля маркера доступа

записи и группы, сопоставленные с процессом или потоком. Механизм защиты Windows использует два элемента маркера, определяя, какие элементы доступны и какие операции можно выполнить.

Первый элемент — это SID учетной записи пользователя и SID групп, к которым этот пользователь принадлежит. Данный элемент используется для определения, можно ли предоставить запрошенный тип доступа к защищаемому объекту, например чтению файла.

Второй элемент — это список привилегий, сопоставленных с маркером. Он используется для определения того, что может делать поток. Например, программно выключать операционную систему. Маркер доступа описан структурой TOKEN (см. рисунки 2–4). Остальные поля маркера используются лишь для информационных нужд.

КОПНЕМ ПОГЛУБЖЕ

Ну и последний важный элемент защиты — это дескриптор защиты объекта. Если ты знаешь, на уровне ядра ОС Windows оперирует таким понятием, как объект (в отличие от никсов, где все — «файл»). То есть объектом будет являться файл, процесс, поток, примитивы синхронизации, APC, DPC, прерывание и так далее.

У каждого такого объекта есть свой описатель, который по сути своей является самостоятельной структурой. Но это не главное — укаждого из объектов есть заголовок, всегда описываемый одной и той же структурой OBJECT_HEADER. Вот она-то нас и интересует.

МАРКЕРЫ ДОСТУПА — ЭТО ОСНОВНОЙ ЭЛЕМЕНТ ЗАЩИТЫ WINDOWS. ОН ОПИСЫВАЕТ КОНТЕКСТ ЗАЩИТЫ ПРОЦЕССА (ПОТОКА) И ПРИВИЛЕГИИ ДОСТУПА ДЛЯ НЕГО

```
kd> !process 380 1
!process 380 1
Searching for Process with Cid == 380
PROCESS ff8027a0 SessionId: 0  Cid: 0380  Peb: 7ffdf000
  ParentCid: 0124  DirBase: 06433000  ObjectTable: ff7e0b68
  TableSize:  23.
  Image: cmd.exe
  VadRoot 84c30568 Clone 0 Private 77. Modified 0. Locked 0.
  DeviceMap 818a3368
  Token                             e22bc730
  ElapsedTime                       14:22:56.0536
```

Рис. 3. С использованием WinDBG можно найти адрес маркера доступа для процесса CMD.EXE...

```
kd> !token e22bc730
_TOKEN e22bc730
TS Session ID: 0
User: S-1-5-21-1787744166-3910675280-2727264193-500
Groups:
 00 S-1-5-21-1787744166-3910675280-2727264193-513
    Attributes - Mandatory Default Enabled
 01 S-1-1-0
    Attributes - Mandatory Default Enabled
 ....
```

Рис. 4. ...и просмотреть детали маркера — какими правами он наделен

Потому что именно в ней содержится указатель на дескриптор защиты объекта, в котором заключена информация о том, кто и что может делать с данным объектом.

Главное, что нужно уяснить, — дескриптор защиты хранит списки управления избирательным доступом (DACL). Они конкретно расписывают, кто может получить доступ к объекту и какой именно доступ может быть предоставлен. ACL'ы состоят из заголовка и перечисляемых элементов ACE. Каждый ACE содержит SID и маску доступа, причем ACE могут быть четырех типов: «доступ разрешен», «доступ отклонен», «разрешенный объект» и «запрещенный объект». Разница между типами «доступ разрешен» и «разрешенный объект» только в том, что последний тип используется лишь в Active Directory.

И ЧТО ТЕПЕРЬ СО ВСЕМ ЭТИМ ДЕЛАТЬ?

Главное, что, во-первых, доступ к объектам системы можно модифицировать. Каким образом — ищи код на диске. Он небольшой и в целом должен быть тебе понятен.

Во-вторых, можно получать доступ к защищенным объектам, используя огрехи самой системы. Ибо, как я уже говорил, контроль доступа в Windows — вещь сложная, а чем сложнее система, тем больше вероятность появления в ней уязвимостей.

В середине 2000-х на багтреках промелькнуло несколько малозаметных сообщений о найденных багах в Windows XP, связанных с возможностью «несанкционированного» поднятия привилегий от Local Service до Local System. Суть уязвимости заключалась в том, что службам Windows SSDP и uPnP, действующим с правами Local Service, можно было изменять параметры любого сервиса в системе, после чего, используя стандартные привилегии запуска/останова службы (вспомни про запуск сервиса из командной строки — sc start/ sc stop), остановить ее и перезапустить с параметрами config, указав в параметре binPath путь к exe для старта:

```
CMD>sc config stupidService binPath=c:\virus.exe obj= \
    ".\LocalSystem" password=""
CMD>sc stop stupidService
CMD>sc start stupidService
```

```
lkd> dt _OBJECT_HEADER 88d0c008
   +0x000 PointerCount     : 36
   +0x004 HandleCount      : 1
   +0x004 NextToFree       : 0x00000001
   +0x008 Type             : 0x8a0ed388
   +0x00c NameInfoOffset   : 0 ''
   +0x00d HandleInfoOffset : 0 ''
   +0x00e QuotaInfoOffset  : 0 ''
   +0x00f Flags            : 0x20 ''
   +0x010 ObjectCreateInfo : 0x89e596a0
   +0x010 QuotaBlockCharged : 0x89e596a0
   +0x014 SecurityDescriptor : 0xe242b864
   +0x018 Body             : _QUAD
```

Рис. 5. Вот где собака порылась!

Идем далее. Хочу отметить, что серьезную брешь в безопасности образует стороннее программное обеспечение, особенно те программы, которые регистрируют себя в качестве Windows-сервиса. И все это опять-таки из-за особого отношения ОС Windows к такого типа программам — многие разработчики ПО оставляют локальной группе Everyone возможность конфигурировать создаваемый сервис вышеуказанным способом.

Особо трепетного отношения к себе требуют те доверенные программы, которые пытаются изменить характеристики какого-то файла при помощи вызова advapi32!SetFileSecurity (хотя и устаревшей) с маской доступа WRITE_DAC.

Необходимо также упомянуть о такой полусекретной технике, как обращение к системным вызовам напрямую через системные шлюзы INT2e/SYSENTER. Я уже как-то описывал ее в одном из прошлых номеров][. Ее суть состоит в прямом вызове прерывания с передачей в стек определенных параметров — в результате мы, во-первых, получаем обход любых юзермодных хуков системных функций, а во-вторых, для вызова опасных функций, типа NtLoadDriver, нам совсем не требуется повышения прав. В примере с тем же NtLoadDriver, скажем, система посмотрит на наши права и потребует установки привилегии Se_Load_Driver_Privilege вызовом AdjustPrivilege(), что не есть гуд. Однако в этот же самый момент мы совершенно спокойно можем напрямую обратиться к системному шлюзу INT2e/SYSENTER.

Ну и в заключение стоит упомянуть, что никто не мешает скомпрометировать сам ход выполнения функций проверки доступа и привилегий, таких как AccessCheck, PrivilegeCheck, AreAnyAccessesGranted и некоторых других, верно? Чуть подправим возвращаемые результаты, и будет нам счастье :).

ЗАКЛЮЧЕНИЕ

И про старуху бывает порнуха, как сказал кто-то из великих. Несмотря на то что с выходом семерки положение дел с правами и привилегиями значительно улучшилось, в защитном механизме Windows все еще можно отыскать лазейки, которые могут поставить на колени эту ОС.

В статье не рассмотрены такие понятия, как учетные записи и локальные аккаунты, и поверь мне, там тоже не все так чисто, как хотелось бы Microsoft. Но это уж оставим тебе в качестве домашнего задания. Удачного компилирования и да пребудет с тобой Сила! ⊐⊏

WWW

Об основах Windows Access Control можно прочитать здесь — bit.ly/pjLau, а также в неплохих статьях на тему: bit.ly/NMQkoy и bit.ly/VxYwtA.

DVD

Код, демонстрирующий смену DACL для файла/папки, ждет тебя на диске.