Управление вводом-выводом, файловой системой и безопасностью в ОС Windows. Часть 1

Лекция

Ревизия: 0.2

Последняя модификация: Vlad Kovtun

Дата последней модификации: 03.04.2015 15:16:00

История изменений

15.09.2010 – Версия 0.1. Первичный документ. Ковтун В.Ю.

09.09.2010 – Версия 0.2. Общая редакция. Ковтун В.Ю.

Содержание

Лекция 12. Управление вводом-выводом, файловой системой и безопасность в ОС Windows. Часть 1

Вопросы

1. Классы безопасности.
2. Компоненты системы защиты.
3. Защита объектов.
4. Права и привилегии учетных записей.
5. Аудит безопасности.
6. Вход в систему.
7. Политика ограниченного использования программ.

Классы безопасности

Четкие стандарты безопасности программного обеспечения, в том числе ОС, помогают правительству, корпорациям и индивидуальным пользователям защищать хранящиеся в компьютерных системах данные, составляющие личную и коммерческую тайну. Текущий стандарт на рейтинги безопасности, применяемый в США и многих других странах, — Common Criteria (CC). Однако, чтобы понять средства защиты, существующие в Windows, нужно знать историю системы рейтингов безопасности, повлиявшей на архитектуру системы защиты Windows, — Trusted Computer System Evaluation Criteria (TCSEC).

Trusted Computer System Evaluation Criteria

Национальный центр компьютерной безопасности (National Computer Security Center, NCSC [5]) был создан в 1981 году в Агентстве национальной безопасности (NSA) Министерства обороны США. Одна из задач NCSC заключалась в определении рейтингов безопасности (см. Таблица 1), отражающих степень защищенности коммерческих ОС, сетевых компонентов и приложений. Эти рейтинги, детальное описание доступно [6], были определены в 1983 году и часто называются «Оранжевой книгой» («Orange Book»).

Таблица 1. Рейтинг безопасности TCSEC

Рейтинг	Название
A1	Verified Design
B3	Security Domains
B2	Structured Protection
B1	Labeled Security Protection
C2	Controlled Access protection
C1	Discretionary Access Protection (obsolete)
D	Minimal Protection

Стандарт TCSEC состоит из рейтингов «уровней доверия» («levels of trust» ratings), где более высокие уровни строятся на более низких, за счет последовательного ужесточения требований к безопасности и проверке. Ни одна ОС не соответствует уровню A1 (Verified Design). Хотя некоторым ОС присвоен один из уровней B, уровень C2 **считается достаточным и является высшим для ОС общего назначения**.

В июле 1995 года Windows NT 3.5 (Workstation и Server) с Service Pack 3 первой из всех версий Windows NT получила подтверждение об уровне безопасности C2. В марте 1999 года организация ITSEC, (Information Technology Security) правительства Великобритании присвоила Windows NT 4 с Service Pack 3 уровень E3, эквивалентный

американскому уровню C2. Windows NT 4 с Service Pack 6a получила уровень C2 для сетевой и автономной конфигураций.

Основные требования к уровню безопасности C2 (они остались прежними) перечислены ниже.

Механизм безопасной регистрации, требующий уникальной идентификации пользователей. Доступ к компьютеру предоставляется лишь после аутентификации.

Управление избирательным доступом, позволяющее владельцу ресурса определять круг лиц, имеющих доступ к ресурсу, а также их права на операции с этим ресурсом. Владелец предоставляет пользователям и их группам различные права доступа.

Аудит безопасности, обеспечивающий возможность регистрации событий, связанных с защитой, а также любых попыток создания, удаления и обращения к системным ресурсам. При входе регистрируются идентификационные данные всех пользователей, что позволяет легко выявить любого, кто попытался выполнить несанкционированную операцию.

Защита при повторном использовании объектов, которая предотвращает просмотр одним из пользователей данных, удаленных из памяти другим, а также доступ к памяти, освобожденной предыдущим пользователем. Например, в некоторых ОС можно создать новый файл определенной длины, а затем просмотреть те данные, которые остались на диске и попали в область, отведенную под новый файл. Среди этих данных может оказаться конфиденциальная информация, хранившаяся в недавно удаленном файле другого пользователя. Так что защита при повторном использовании объектов устраняет потенциальную уязвимость в системе безопасности, заново инициализируя перед выделением новому пользователю все объекты, включая файлы и память. Windows также отвечает двум требованиям защиты уровня B.

Функциональность пути доверительных отношений (trusted path functionality), которая предотвращает перехват имен и паролей пользователей программами — троянскими конями. Эта функциональность реализована в Windows в виде входной сигнальной комбинации клавиш Ctrl+Alt+Del и не может быть перехвачена непривилегированными приложениями. Такая комбинация клавиш, известная как **SAS** (secure attention sequence), вызывает диалоговое окно входа в систему, обходя вызов его фальшивого эквивалента из троянского коня.

Управление доверительными отношениями (trusted facility management), которое требует поддержки набора ролей (различных типов учетных записей) для разных уровней работы в системе. Например, функции администрирования доступны только по одной группе учетных записей — Administrators (Администраторы).

Windows соответствует всем перечисленным требованиям.

Common Criteria

В январе 1996 года США, Великобритания, Германия, Франция, Канада и Нидерланды опубликовали совместно разработанную спецификацию оценки безопасности Common Criteria for Information Technology Security Evaluation (CCITSE). Эта спецификация, чаще называемая Common Criteria (CC) [7], является международным стандартом оценки степени защищенности продуктов.

CC гибче уровней доверия TCSEC и по структуре ближе ITSEC, чем TCSEC. CC включает две концепции:

профиля защиты (Protection Profile, PP) — требования к безопасности разбиваются на группы, которые легко определять и сравнивать;

объекта защиты (Security Target, ST) — предоставляет набор требований к защите, которые могут быть подготовлены с помощью PP.

Windows 2000 оценивалась на соответствие требованиям Controlled Access PP, эквивалентным TCSEC C2, и на соответствие дополнительным требованиям Common Criteria в октябре 2002 года. Подробности можно найти [8, 9]. К значимым требованиям, не включенным в Controlled Access PP, но предъявляемым по условиям Windows 2000 Security Target, относятся:

функции управления избирательным доступом (Discretionary Access Control Functions), основанные на применении криптографии. Они реализуются файловой системой Encrypting File System и Data Protection API в Windows 2000;

политика управления избирательным доступом (Discretionary Access Control Policy) для дополнительных пользовательских объектов данных (User Data Objects), например объекты Desktop и WindowStation (реализуются подсистемой поддержки окон Windows 2000), а также объекты Active Directory (реализуются службой каталогов в Windows 2000);

внутренняя репликация (Internal Replication) для гарантированной синхронизации элементов данных, связанных с защитой, между физически раздельными частями системы Windows 2000 как распределенной ОС. Это требование реализуется службой каталогов Windows 2000 с применением модели репликации каталогов с несколькими хозяевами;

утилизация ресурсов (Resource Utilization) для физических пространств дисков. Это требование реализуется файловой системой NTFS в Windows 2000;

блокировка интерактивного сеанса (Interactive Session Locking) и путь **доверительных отношений** (Trusted Path) для первоначального входа пользователя (initial user logging on). Это требование реализуется компонентом Winlogon в Windows 2000;

защита внутренней передачи данных (Internal Data Transfer Protection), чтобы обезопасить данные от раскрытия и несанкционированной модификации при передаче между физически раздельными частями системы Windows 2000 как распределенной ОС. Это требование реализуется службой IPSEC в Windows 2000;

систематическое устранение обнаруживаемых недостатков в системе защиты (Systematic Flaw Remediation). Это требование реализуется Microsoft Security Response Center и Windows Sustained Engineering Team.

Так, Windows XP Embedded, Windows XP Professional и Windows Server 2003 уже прошли проверку на соответствие Common Criteria. Набор критериев расширен по сравнению с тем, который применялся к Windows 2000. Комитет Common Criteria также рассмотрел Windows XP и Windows Server 2003 (Standard, Enterprise и Datacenter Edition) для оценки технологий следующих типов [10]:

- распределенной ОС;
- защиты конфиденциальных данных;
- управления сетью;
- службы каталогов;
- брандмауэра;
- VPN (Virtual Private Network);
- управления рабочим столом;
- инфраструктуры открытого ключа (Public Key Infrastructure, PKI);
- выдачи и управления сертификатами открытого ключа;
- встраиваемой ОС.

Компоненты системы защиты

Ниже перечислены главные компоненты и базы данных, на основе которых реализуется защита в Windows.

Монитор состояния защиты (Security Reference Monitor, SRM).

Компонент исполнительной системы (\Windows\System32\Ntoskrnl.exe), отвечающий за определение структуры данных маркера доступа для представления контекста защиты, за проверку прав доступа к объектам, манипулирование привилегиями (правами пользователей) и генерацию сообщений аудита безопасности.

Подсистема локальной аутентификации (local security anthentication subsystem, LSASS). Процесс пользовательского режима, выполняющий образ \Windows\System32\Lsass.exe, который отвечает за политику безопасности в локальной системе (например, круг пользователей, имеющих право на вход в систему, правила, связанные с паролями, привилегии, выдаваемые пользователям и их группам, параметры аудита безопасности системы), а также за аутентификацию пользователей и передачу сообщений аудита безопасности в Event Log. Основную часть этой функциональности реализует сервис локальной аутентификации Lsasrv (\Windows\System32\Lsasrv.dll) — DLL-модуль, загружаемый Lsass.

База данных политики LSASS. База данных, содержащая параметры политики безопасности локальной системы. Она хранится в разделе реестра `HKLM\SECURITY` и включает следующую информацию: каким доменам доверена аутентификация попыток

входа в систему, кто имеет права на доступ к системе и каким образом, кому предоставлены те или иные привилегии и какие виды аудита следует выполнять. База данных политики LSASS также хранит «секреты», которые включают в себя регистрационные данные, применяемые для входа в домены и при вызове Windows-сервисов.

Диспетчер учетных записей безопасности (Security Accounts Manager, SAM). Набор подпрограмм, отвечающих за поддержку базы данных, которая содержит имена пользователей и группы, определенные на локальной машине. Служба SAM, реализованная как `\Windows\System32\Samsrv.dll`, выполняется в процессе Lsass.

База данных SAM. База данных, которая в системах, отличных от контроллеров домена, содержит информацию о локальных пользователях и группах вместе с их паролями и другими атрибутами. На контроллерах домена SAM содержит определение и пароль учетной записи администратора, имеющего права на восстановление системы. Эта база данных хранится в разделе реестра `HKLM\SAM`.

Active Directory. Служба каталогов, содержащая базу данных со сведениями об объектах в домене. **Домен** — это совокупность компьютеров и сопоставленных с ними групп безопасности, которые управляются как единое целое. Active Directory хранит информацию об объектах домена, в том числе о пользователях, группах и компьютерах. Сведения о паролях и привилегиях пользователей домена и их групп содержатся в Active Directory и реплицируются на компьютеры, выполняющие роль контроллеров домена. Сервер Active Directory, реализованный как `\Windows\System32\Ntdsa.dll`, выполняется в процессе Lsass.

Пакеты аутентификации. DLL-модули, выполняемые в контексте процесса Lsass и клиентских процессов и реализующие политику аутентификации в Windows. DLL аутентификации отвечает за проверку пароля и имени пользователя, а также за возврат LSASS (в случае успешной проверки) детальной информации о правах пользователя, на основе которой LSASS генерирует **маркер** (token).

Процесс входа (Winlogon). Процесс пользовательского режима (`\Windows\System32\Winlogon.exe`), отвечающий за поддержку SAS и управление сеансами интерактивного входа в систему. Например, при регистрации пользователя Winlogon создаст оболочку — пользовательский интерфейс.

GINA (Graphical Identification and Authentication). DLL пользовательского режима, выполняемая в процессе Winlogon и применяемая для получения пароля и имени пользователя или PIN-кода смарт-карты. Стандартная GINA хранится в `\Windows\System32\Msgina.dll`.

Служба сетевого входа (Netlogon). Windows-сервис (`\Windows\System32\Netlogon.dll`), устанавливающий защищенный канал с контроллером домена, по которому посылаются запросы, связанные с защитой, например для интерактивного входа (если контроллер домена работает под управлением Windows NT), или запросы на аутентификацию от LAN Manager либо NT LAN Manager (v1 и v2).

Kernel Security Device Driver (KSecDD). Библиотека функций режима ядра, реализующая интерфейсы LPC (local procedure call), которые используются другими компонентами защиты режима ядра — в том числе **шифрующей файловой системой** (Encrypting File System, EFS) — для взаимодействия с LSASS в пользовательском режиме. KsecDD содержится в `\Windows\System32\Drivers\Ksecdd.sys`. На Рис. 1 показаны взаимосвязи между некоторыми из этих компонентов и базами данных, которыми они управляют.

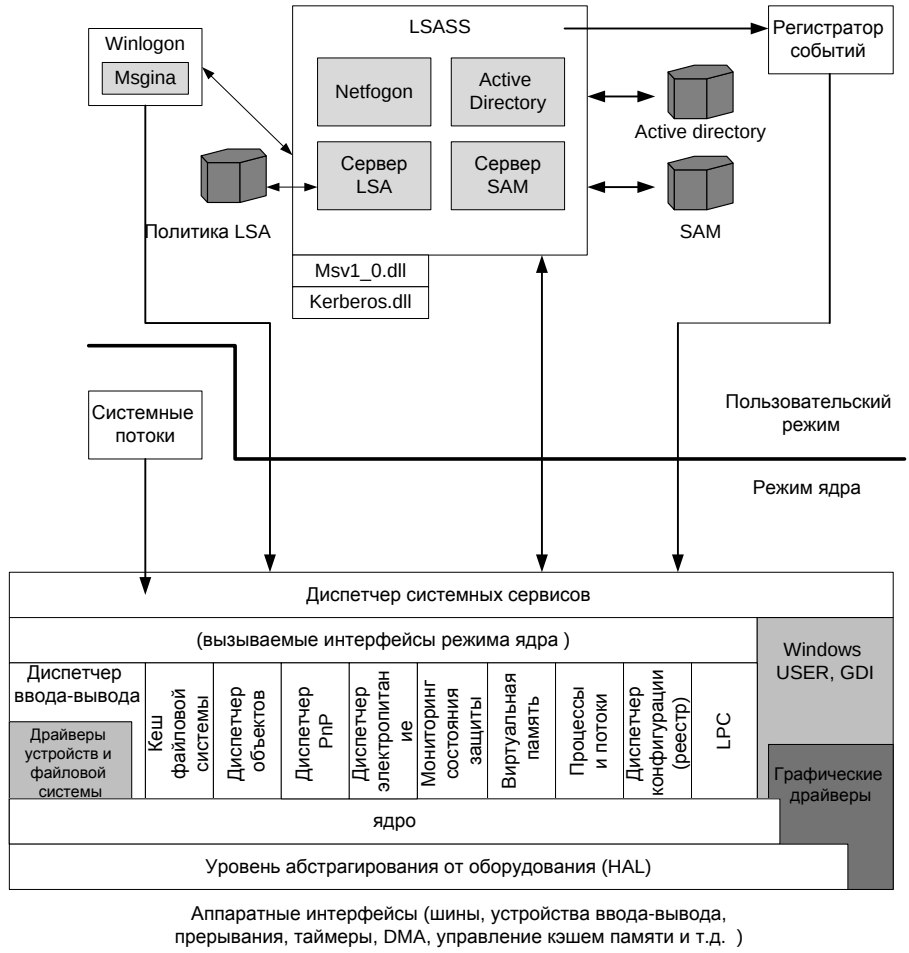

Рис. 1. Компоненты защиты Windows

Защита объектов

Защита объектов и протоколирование обращений к ним — вот сущность управления избирательным доступом и аудита. Защищаемые объекты Windows включают:

- файлы,
- устройства,
- почтовые ящики,
- каналы (именованные и анонимные),
- задания,
- процессы,
- потоки,
- события,
- пары событий,
- мьютексы,
- семафоры,
- порты завершения ввода-вывода,
- разделы общей памяти,
- LPC-порты,
- ожидаемые таймеры,
- маркеры доступа,
- тома,
- объекты WindowStation,
- рабочие столы,
- сетевые ресурсы,
- сервисы,
- разделы реестра,
- принтеры,
- объекты Active Directory.

Поскольку системные ресурсы, экспортируемые в пользовательский режим (и поэтому требующие проверки защиты), реализуются как объекты режима ядра, диспетчер объектов играет ключевую роль в их защите. Для контроля над операциями над объектом, система защиты должна быть уверена в правильности идентификации каждого пользователя. Именно по этой причине Windows требует от пользователя входа с аутентификацией, прежде чем ему будет разрешено обращаться к системным ресурсам. Когда какой-либо процесс запрашивает описатель объекта, диспетчер объектов и система защиты на основе идентификационных данных вызывающего

процесса определяют, можно ли предоставить ему описатель, разрешающий доступ к нужному объекту.

Контекст защиты потока может отличаться от контекста защиты его процесса. Этот механизм называется **олицетворением** (impersonation), или подменой. При олицетворении механизмы проверки защиты используют вместо контекста защиты процесса контекст защиты потока, а без олицетворения — контекст защиты процесса, которому принадлежит поток. Важно не забывать, что все потоки процесса используют одну и ту же таблицу описателей, поэтому, когда поток открывает какой-нибудь объект (даже при олицетворении), все потоки процесса получают доступ к этому объекту.

Проверка прав доступа

Модель защиты Windows требует, чтобы поток заранее — еще до открытия объекта — указывал, какие операции он собирается выполнять над этим объектом. Система проверяет тип доступа, запрошенный потоком, и, если такой доступ ему разрешен, он получает описатель, позволяющий ему (и другим потокам того же процесса) выполнять операции над объектом. Диспетчер объектов регистрирует права доступа, предоставленные для данного описателя, в таблице описателей, принадлежащей процессу.

Одно из событий, заставляющее диспетчер объектов проверять права доступа, — **открытие процессом существующего объекта по имени**. При открытии объекта по имени диспетчер объектов ищет его в своем пространстве имен. Если этого объекта нет во вторичном пространстве имен (например, в пространстве имен реестра, принадлежащем диспетчеру конфигурации, или в пространстве имен файловой системы, принадлежащем драйверу файловой системы), диспетчер объектов вызывает внутреннюю функцию `ObpCreateHandle`. Как и подсказывает ее имя, она создает элемент в таблице описателей, который сопоставляется с объектом. Однако `ObpCreateHandle` вызывает функцию исполнительной системы `ExCreateHandle` и создаст описатель, только если другая функция диспетчера объектов, `ObpIncrementHandleCount`, сообщает, что поток имеет право на доступ к данному объекту. Правда, реальную проверку прав доступа осуществляет другая функция диспетчера объектов, `ObpCheckObjectAccess`, которая возвращает результаты проверки функции `ObpIncrementHandleCount`.

`ObpIncrementHandleCount` передает `ObCheckObjectAccess` удостоверения защиты потока, открывающего объект, типы запрошенного им доступа (чтение, запись, удаление и т. д.), а также указатель на объект. `ObCheckObjectAccess` сначала блокирует защиту объекта и контекст защиты потока. Блокировка защиты объекта предотвращает ее изменение другим потоком в процессе проверки прав доступа, а блокировка контекста защиты потока не дает другому потоку того же или другого процесса изменить идентификационные данные защиты первого потока при проверке его прав доступа. Далее `ObCheckObjectAccess` вызывает метод защиты объекта, чтобы получить параметры защиты объекта. Вызов метода защиты может привести к вызову функции из другого компонента исполнительной системы, но многие объекты исполнительной системы полагаются на стандартную поддержку управления защитой, предлагаемую системой.

Если компонент исполнительной системы, определяя объект, не собирается заменять стандартную политику безопасности, он помечает тип этих объектов как использующий стандартную защиту. Всякий раз, когда SRM вызывает метод защиты объекта, он сначала проверяет, использует ли объект стандартную защиту. Объект со стандартной защитой хранит информацию о защите в своем заголовке и предоставляет метод защиты с именем `SeDefaultObjectMethod`. Объект, не использующий стандартную защиту, должен сам поддерживать информацию о защите и предоставлять собственный метод защиты. Стандартную защиту используют такие объекты, как мьютексы, события и семафоры. Пример объекта с нестандартной защитой — файл. У диспетчера ввода-вывода, определяющего объекты типа «файл», имеется драйвер файловой системы, который управляет защитой своих файлов (или решает не реализовать ее). Таким образом, когда система запрашивает информацию о защите объекта «файл», представляющего файл на томе NTFS, она получает эту информацию от драйвера файловой системы NTFS, который в свою очередь получает ее от метода защиты объекта «файл», принадлежащего диспетчеру ввода-вывода. Заметьте, что при открытии файла `ObCheckObjectAccess` не выполняется, так как объекты «файл» находятся во вторичных пространствах имен; система вызывает метод защиты объекта

«файл», только если поток явно запрашивает или устанавливает параметры защиты файла (например, через Windows-функции `SetFileSecurity` или `GetFileSecurity`).

Получив информацию о защите объекта, `ObCheckObjectAccess` вызывает SRM-функцию `SeAccessCheck`, на которую опирается вся модель защиты Windows. Она принимает параметры защиты объекта, идентификационные данные защиты потока (в том виде, в каком они получены `ObCbeckObjectAccess`) и тип доступа, запрашиваемый потоком. `SeAccessCheck` возвращает `True` или `False` в зависимости от того, предоставляет ли она потоку запрошенный тип доступа к объекту.

Другое событие, заставляющее диспетчер объектов выполнять проверку прав доступа, — **ссылка процесса на объект по существующему описателю**. Подобные ссылки часто делаются косвенно, например при манипуляциях с объектом через Windows API с передачей его описателя. Допустим, поток, открывающий файл, запрашивает доступ для чтения из файла. Если у потока есть соответствующие права, определяемые его контекстом защиты и параметрами защиты файла, диспетчер объектов создает описатель данного файла в таблице описателей, которая принадлежит процессу — владельцу этого потока. Информация о предоставленном процессу типе доступа сопоставляется с описателем и сохраняется диспетчером объектов.

Впоследствии поток может попытаться что-то записать в этот файл через Windows-функцию `WriteFile`, передав в качестве параметра описатель файла. Системный сервис `NtWriteFile`, который `WriteFile` вызовет через `Ntdll.dll`, обратится к функции диспетчера объектов `ObReferenceObjectByHandle`, чтобы получить указатель на объект «файл» по его описателю. `ObReferenceObjectByHandle` принимает запрошенный тип доступа как параметр. Найдя в таблице описателей элемент, соответствующий нужному описателю, `ObReferenceObjectByHandle` сравнит запрошенный тип доступа с тем, который был предоставлен при открытии файла. В данном случае `ObReferenceObjectByHandle` укажет, что операция записи должна завершиться неудачно, так как вызывающий поток, открывая файл, не получил право на его запись.

Функции защиты Windows также позволяют Windows-приложениям определять собственные закрытые объекты и вызывать SRM-сервисы для применения к этим объектам средств защиты Windows. Многие функции режима ядра, используемые диспетчером объектов и другими компонентами исполнительной системы для защиты своих объектов, экспортируются в виде Windows-функций пользовательского режима. Например, эквивалентом `SeAccessCheck` для пользовательского режима является `AccessCheck`. Таким образом, Windows-приложения могут применять модель защиты Windows и интегрироваться с интерфейсами аутентификации и администрирования этой ОС.

Сущность модели защиты SRM отражает математическое выражение с тремя входными параметрами: идентификационными данными защиты потока, запрошенным типом доступа и информацией о защите объекта. Его результат — значения «да» или «нет», которые определяют, предоставит ли модель защиты запрошенный тип доступа.

Идентификаторы защиты

Для идентификации объектов, выполняющих в системе различные действия, Windows использует не имена (которые могут быть не уникальными), а **идентификаторы защиты** (security identifiers, SID). SID имеются у пользователей, локальных и доменных групп, локальных компьютеров, доменов и членов доменов. **SID** представляет собой числовое значение переменной длины, формируемое из номера версии структуры SID, 48-битного кода агента идентификатора и переменного количества 32-битных кодов **субагентов** и/или **относительных идентификаторов** (relative identifiers, RID). **Код агента идентификатора** (identifier authority value) определяет агент, выдавший SID. Таким агентом обычно является локальная система или домен под управлением Windows. Коды субагентов идентифицируют попечителей, уполномоченных агентом, который выдал SID, а RID — не более чем средство создания уникальных SID на основе **общего базового SID** (common-based SID). Поскольку длина SID довольно велика и Windows старается генерировать истинно случайные значения для каждого SID, вероятность появления двух одинаковых SID практически равна нулю.

В текстовой форме каждый SID начинается с префикса `S`, за которым следуют группы чисел, разделяемые дефисами, например:

```
S-1-5-21-1463437245-1224812800-863842198-1128
```

В этом SID номер версии равен 1, код агента идентификатора — 5 (центр безопасности Windows), далее идут коды четырех субагентов и один RID в конце (1128). Этот SID относится к домену, так что локальный компьютер этого домена получит SID с тем же номером версии и кодом агента идентификатора; кроме того, в нем будет столько же кодов субагентов.

SID назначается компьютеру при установке Windows (программой Windows Setup). Далее Windows назначает SID локальным учетным записям на этом компьютере. SID каждой локальной учетной записи формируется на основе SID компьютера с добавлением RID. RID пользовательской учетной записи начинается с 1000 и увеличивается на 1 для каждого нового пользователя или группы. Аналогичным образом `Dcpromo.exe` — утилита, применяемая при создании нового домена Windows, — выдает SID только что созданному домену. Новые учетные записи домена получают SID, формируемые на основе SID домена с добавлением RID (который также начинается с 1.000 и увеличивается на 1 для каждою нового пользователя или группы). RID с номером 1028 указывает на то, что его SID является 28-ым, выданным доменом.

Многим предопределенным учетным записям и группам Windows выдает SID, состоящие из SID компьютера или домена и предопределенного RID. Так, RID учетной записи администратора равен 500, а RID гостевой учетной записи — 501. Например, в основе SID учетной записи локального администратора лежит SID компьютера, к которому добавлен RID, равный 500:

```
S-1-5-21-13124455-12541255-61235125-500
```

Для групп Windows также определяет ряд встроенных локальных и доменных SID. Например, SID, представляющий любую учетную запись, называется **Everyone** или **World** и имеет вид `S-1-1-0`. Еще один пример — сетевая группа, т.е. группа, пользователи которой зарегистрировались на данном компьютере из сети. SID сетевой группы имеет вид `S-1-5-2`.

Наконец, Winlogon создает уникальный SID для каждого интерактивного сеанса входа. SID входа, как правило, используется в элементе **списка управления доступом** (access-control entry, ACE), который разрешает доступ на время сеанса входа клиента. Например, Windows-сервис может вызвать функцию `LogonUser` для запуска нового сеанса входа. Эта функция возвращает маркер доступа, из которого сервис может извлечь SID входа. Потом этот SID сервис может использовать в ACE, разрешающем обращение к интерактивным объектам WindowStation и Desktop из сеанса входа клиента. SID для сеанса входа выглядит как `S-1-5-5-0`, а RID генерируется случайным образом.

Таблица 2. Некоторые общеизвестные SID

SID	Группа	Описание
S-1-1-0	Everyone	Группа, включающая всех пользователей
S-1-2-0	Local	Пользователи, которые регистрируются на терминалах, локально (физически) подключенных к системе
S-1-3-0	Creator Owner ID	Идентификатор защиты, подлежащий замене идентификатором защиты пользователя, создавшего новый объект; этот SID применяется в наследуемых ACE
S-1-3-1	Creator Group ID	Идентификатор защиты, подлежащий замене SID основной группы, в которую входит пользователь, создавший новый объект; этот SID применяется в наследуемых ACE

Маркеры

Для идентификации контекста защиты процесса или потока SRM использует объект, называемый **маркером** (token), или **маркером доступа** (access token). В контекст защиты входит информация, описывающая привилегии, учетные записи и группы, сопоставленные с процессом или потоком. В процессе входа в систему (этот процесс рассматривается в конце главы) Winlogon создает начальный маркер, представляющий пользователя, который входит в систему, и сопоставляет его с начальным процессом

(или процессами) — по умолчанию запускается `Userinit.exe`. Так как дочерние процессы по умолчанию наследуют копию маркера своего создателя, все процессы в сеансе данного пользователя выполняются с одним и тем же маркером. Можно сгенерировать маркер вызовом Windows-функции `LogonUser` и применить его для создания процесса, который будет выполняться в контексте защиты пользователя, зарегистрированного с помощью функции `LogonUser`, с этой целью вы должны передать полученный маркер Windows-функции `CreateProcessAsUser`. Функция `CreateProcessWithLogon` тоже создает маркер, создавая новый сеанс входа с начальным процессом. Именно так команда `runas` запускает процессы с **альтернативными маркерами**.

Механизмы защиты в Windows используют два элемента маркера, определяя, какие объекты доступны и какие операции можно выполнять. Первый элемент состоит из SID учетной записи пользователя и полей SID групп. Используя SID-идентификаторы, SRM определяет, можно ли предоставить запрошенный тип доступа к защищаемому объекту, например к файлу в NTFS.

SID групп в маркере указывают, в какие группы входит учетная запись пользователя. При обработке клиентских запросов серверные приложения могут блокировать определенные группы для ограничения удостоверений защиты, сопоставленных с маркером. Блокирование группы дает почти тот же эффект, что и ее исключение из маркера. (Блокированные SID все же используются при проверке прав доступа, но об этом расскажем потом.)

Вторым элементом маркера, определяющим, что может делать поток или процесс, которому назначен данный маркер, является **список привилегий — прав, сопоставленных с маркером**. Примером привилегии может служить право процесса или потока, сопоставленного с маркером, на выключение компьютера. (Подробнее привилегии будут рассмотрены позже.) Поля основной группы маркера по умолчанию и **списка управления избирательным доступом** (discretionary access-control list, DACL) представляют собой атрибуты защиты, применяемые Windows к объектам, которые создаются процессом или потоком с использованием маркера. Включая в маркеры информацию о защите, Windows упрощает процессам и потокам создание объектов со стандартными атрибутами защиты, так как в этом случае им не требуется запрашивать информацию о защите при создании каждого объекта.

Маркер может быть **основным** (primary token) (идентифицирует контекст защиты процесса) и **олицетворяющим** (impersonation token) (применяется для временного заимствования потоком другого контекста защиты — обычно другого пользователя). Маркеры олицетворения сообщают уровень олицетворения, определяющий, какой тип олицетворения активен в маркере.

Остальные поля маркера служат для информационных нужд. Поле источника маркера содержит сведения (в текстовой форме) о создателе маркера. Оно позволяет различать такие источники, как диспетчер сеансов Windows, сетевой файл-сервер или RPC-сервер. Идентификатор маркера представляет собой **локально уникальный идентификатор** (locally unique identifier, LUID), который SRM присваивает маркеру при его создании. Исполнительная система поддерживает свой LUID — счетчик, с помощью которого она назначает каждому маркеру уникальный числовой идентификатор.

Еще одна разновидность LUID - **идентификатор аутентификации** (authentication TD). Он назначается маркеру создателем при вызове функции `LsaLogonUser`. Если создатель не указывает LUID, то LSASS формирует LUID из LUID исполнительной системы. LSASS копирует идентификатор аутентификации для всех маркеров — потомков начального маркера. Используя этот идентификатор, программа может определить, принадлежит ли какой-то маркер тому же сеансу, что и остальные маркеры, анализируемые данной программой.

LUID исполнительной системы обновляет идентификатор модификации при каждом изменении характеристик маркера. Проверяя этот идентификатор, программа может обнаруживать изменения в контексте защиты с момента его последнего использования.

Маркеры содержат поле времени окончания действия, которое присутствует в них, начиная с Windows NT 3.1, но до сих пор не используется.

Олицетворение

Олицетворение (impersonation) — мощное средство, часто используемое в модели защиты Windows. Олицетворение также применяется в модели программирования

«клиент-сервер». Например, серверное приложение может экспортировать ресурсы (файлы, принтеры или базы данных). Клиенты, которые хотят обратиться к этим ресурсам, посылают серверу запрос. Получив запрос, сервер должен убедиться, что у клиента есть разрешение на выполнение над ресурсом запрошенных операций. Так, если пользователь на удаленной машине пытается удалить файл с сетевого диска NTFS, сервер, экспортирующий этот сетевой ресурс, должен проверить, имеет ли пользователь право удалить данный файл. Казалось бы, в таком случае сервер должен запросить учетную запись пользователя и SID-идентификаторы группы, а также просканировать атрибуты защиты файла. Но этот процесс труден для программирования, подвержен ошибкам и не позволяет обеспечить поддержку новых функций защиты. Поэтому Windows в таких ситуациях предоставляет серверу сервисы олицетворения.

Олицетворение позволяет серверу уведомить SRM (Security Reference Monitor) о временном заимствовании профиля защиты клиента, запрашивающего ресурс. После этого сервер может обращаться к ресурсам от имени клиента, а SRM — проводить проверку его прав доступа. Обычно серверу доступен более широкий круг ресурсов, чем клиенту, и при олицетворении сервер может терять часть исходных прав доступа. Также вероятно и обратное: при олицетворении сервер может получить дополнительные права.

Сервер олицетворяет клиент лишь в пределах потока, выдавшего запрос на олицетворение. Управляющие структуры данных потока содержат необязательный элемент для маркера доступа. Однако основной маркер потока, отражающий его реальные права, всегда доступен через управляющие структуры процесса.

За поддержку олицетворения в Windows отвечает несколько механизмов. Если сервер взаимодействует с клиентом через именованный канал, он может вызвать Windows-функцию `ImpersonateNamedPipeClient` и тем самым сообщить SRM о том, что ему нужно подменить собой пользователя на другом конце канала. Если сервер взаимодействует с клиентом через DDE (Dynamic Data Exchange) или RPC, то выдает аналогичный запрос на олицетворение через `DdelmpersonoteClient` или `RpcImpersonateClient`. Поток может создать маркер олицетворения просто как копию маркера своего процесса, вызвав функцию `ImpersonateSelf`. Для блокировки каких-то SID или привилегий поток может потом изменить полученный маркер олицетворения. Наконец, пакет SSPI (Security Support Provider Interface) может олицетворять своих клиентов через `ImpersonateSecurityContext`. SSPl реализует модель сетевой защиты вроде LAN Manager версии 2 или Kerberos.

После того как серверный поток завершает выполнение своей задачи, он возвращает себе прежний профиль защиты. Эти формы олицетворения удобны для выполнения определенных операций по запросу клиента и для корректного аудита обращений к объектам. Их недостаток в том, что нельзя выполнять всю программу в контексте клиента. Кроме того, маркер олицетворения не дает доступа к сетевым файлам или принтерам, если только они не поддерживают null-сеансы или не используется олицетворение **уровня делегирования** (delegation-level impersonation), причем удостоверения защиты достаточны для аутентификации на удаленном компьютере. (Null-сеанс создастся при анонимном входе.)

Если все приложение должно выполняться в контексте защиты клиента или получать доступ к сетевым ресурсам, клиент должен быть зарегистрирован в системе. Для этого предназначена Windows-функция `LogonUser`, которая принимает в качестве параметров имя учетной записи, пароль, имя домена или компьютера, тип входа (интерактивный, пакетный или сервисный) и **провайдер входа** (logon provider), а возвращает основной маркер. Серверный поток принимает маркер в виде маркера олицетворения, либо сервер запускает программу, основной маркер которой включает удостоверения клиента. С точки зрения защиты, процесс, создаваемый с применением маркера, который возвращается при интерактивном входе через `LogonUser`, например API-функцией `CreateProcessAsUser`, выглядит как программа, запущенная пользователем при интерактивном входе в систему. Недостаток этого подхода в том, что серверу приходится получать имя и пароль по учетной записи пользователя. Если сервер передает эту информацию по сети, он должен надежно шифровать ее, чтобы избежать получения имени и пароля злоумышленником, перехватывающим сетевой трафик.

Windows не позволяет серверам подменять клиенты без их ведома. Клиентский процесс может ограничить уровень олицетворения серверным процессом, сообщив при соединении с ним требуемый SQoS (Security Quality of Service). Процесс может указывать флаги `SECURITY_ANONYMOUS`, `SECURITY_IDENTIFICATION`,

SECURITY_IMPERSONATION и SECURITY_DELEGATION при вызове Windows-функции CreateFile. Каждый уровень позволяет серверу выполнять различный набор операций относительно контекста защиты клиента:

SecurityAnonymous — самый ограниченный уровень; сервер не может олицетворять или идентифицировать клиент;

SecurityIdentification — сервер может получать SID и привилегии клиента, но не получает право на олицетворение клиента;

SecurityImpersonation—сервер может идентифицировать и олицетворять клиент в локальной системе;

SecurityDelegation — наименее ограниченный уровень. Позволяет серверу олицетворять клиент в локальных и удаленных системах. Windows NT 4 и более ранние версии лишь частично поддерживают этот уровень олицетворения.

Если клиент не устанавливает уровень олицетворения, Windows по умолчанию выбирает SecurityImpersonation. Функция CreateFile также принимает модификаторы SECURITY_EFFECTIVE_ONLY и SECURITY_CONTEXT_TRACKING.

Первый из них не дает серверу включать/выключать какие-то привилегии или группы клиента на время олицетворения. А второй указывает, что все изменения, вносимые клиентом в свой контекст защиты, отражаются и на сервере, который олицетворяет этот клиент. Данный модификатор действует, только если клиентский и серверный процессы находятся в одной системе.

Ограниченные маркеры

Ограниченный маркер (restricted token) создается на базе основного или олицетворяющего с помощью функции CreateRestrictedToken и является его копией, в которую можно внести следующие изменения:

- удалить некоторые элементы из таблицы привилегий маркера;
- пометить SID-идентификаторы маркера атрибутом проверки только на запрет (deny-only);
- пометить STD-идентификаторы маркера как ограниченные.

Поведение SID с атрибутом проверки только на запрет (deny-only SID) и ограниченных SID (restricted SID) кратко поясняется в следующих разделах. Ограниченные маркеры удобны, когда приложение подменяет клиент при выполнении небезопасного кода. В ограниченном маркере может, например, отсутствовать привилегия на перезагрузку системы, что не позволит коду, выполняемому в контексте защиты ограниченного маркера, перезагрузить систему.

Дескрипторы защиты и управление доступом

Маркеры, которые идентифицируют удостоверения пользователя, являются лишь частью выражения, описывающего защиту объектов. Другая его часть — информация о защите, сопоставленная с объектом и указывающая, кому и какие действия разрешено выполнять над объектом. Структура данных, хранящая эту информацию, называется **дескриптором защиты** (security diescriptor). Дескриптор защиты включает следующие атрибуты.

Номер версии. Версия модели защиты SRM, использованной для создания дескриптора.

Флаги. Необязательные модификаторы, определяющие поведение или характеристики дескриптора. Пример — флаг SE_DACL_PROTECTED, который запрещает наследование дескриптором параметров защиты от другого объекта.

SID владельца. Идентификатор защиты владельца.

SID группы. Идентификатор защиты основной группы для данного объекта (используется только POSIX).

Список управления избирательным доступом (discretionary access-control list, DACL) Указывает, кто может получать доступ к объек-ту и какие виды доступа.

Системный список управления доступом (system access-control list, SACL), указывает какие операции и каких пользователей должны регистрироваться в журнале аудита безопасности.

Список управления доступом (access-control list, ACL) состоит из заголовка и может содержать элементы (access-control entries, ACE). Существует два типа ACL: DACL и SACL. В DACL каждый ACE содержит SID и маску доступа (а также набор флагов), причем ACE могут быть четырех типов - «доступ разрешен» (access allowed), «доступ отклонен» (access denied), «разрешенный объект» (allowed-objecl) и «запрещенный объект» (denied-object). Первый тип ACE разрешает пользователю доступ к объекту, а второй — отказывает в предоставлении прав, указанных в маске доступа.

Разница между ACE типа «разрешенный объект» и «доступ разрешен», а также между ACE типа «запрещенный объект» и «доступ отклонен» заключается в том, что эти типы используются только в Active Directory. ACE этих типов имеют поле **глобально уникального идентификатора** (globally unique identifier, GUID), которое сообщает, что данный ACE применим только к определенным объектам или подобъектам (с GUID-идентификаторами). Кроме того, необязательный GUID указывает, что тип дочернего объекта наследует ACE при его (объекта) создании в контейнере Active Directory, к которому применен ACE. (GUID — это гарантированно уникальный 128-битный идентификатор.)

За счет аккумуляции прав доступа, сопоставленных с индивидуальными ACE, формируется набор прав, предоставляемых ACL-списком. Если в дескрипторе защиты нет DACL (DACL = null), любой пользователь получает полный доступ к объекту. Если DACL пуст (т. е. в нем нет ACE), доступа к объекту не получает никто.

ACE, используемые в DACL, также имеют набор флагов, контролирующих и определяющих характеристики ACE, связанные с наследованием. Некоторые пространства имен объектов содержат объекты-контейнеры и объекты-листы (leaf objects). Контейнер может включать другие контейнеры и листы, которые являются его дочерними объектами. Примеры контейнеров — каталоги в пространстве имен файловой системы и разделы в пространстве имен реестра. Отдельные флаги контролируют, как ACE применяется к дочерним объектам контейнера, сопоставленного с этим ACE.

SACL состоит из ACE двух типов: системного аудита (system audit ACE) и объекта системного аудита (system audit-object ACE). Эти ACE определяют, какие операции, выполняемые над объектами конкретными пользователями или группами, подлежат аудиту. Информация аудита хранится в системном журнале аудита. Аудиту могут подлежать как успешные, так и неудачные операции. Как и специфические для объектов ACE из DACL, ACE объектов системного аудита содержат GUID, указывающий типы объектов или подобъектов, к которым применим данный ACE, и необязательный GUID, контролирующий передачу ACE дочерним объектам конкретных типов. При SACL, равном `null`, аудит объекта не ведется. (Об аудите безопасности мы расскажем позже.) Флаги наследования, применимые к DACL ACE, применимы к ACE системного аудита и объектов системного аудита.

Упрощенная схема объекта «файл» и его DACL представлена на Рис. 2.

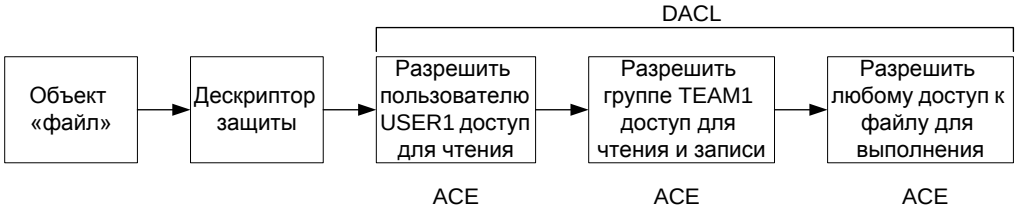

Рис. 2. Список управления избирательным доступом

Как показано на Рис. 2, первый ACE позволяет USER1 читать файл. Второй ACE разрешает членам группы TEAM 1 читать и записывать файл. Третий ACE, предоставляет доступ к файлу для выполнения всем пользователям.

Определение прав доступа

Для определения прав доступа к объекту используются два алгоритма:

- сравнивающий запрошенные права с максимально возможными для данного объекта и экспортируемый в пользовательский режим в виде Windows-функции `GetEffectiveRightsFromAcl`;

- проверяющий наличие конкретных прав доступа и активизируемый через Windows-функцию `AccessCheck` или `AccessCheckByType`.

Первый алгоритм проверяет элементы DACL следующим образом.

1. В отсутствие DACL (DACL = null) объект является незащищенным, и система защиты предоставляет к нему полный доступ.

2. Если у вызывающего потока имеется привилегия на захват объекта во владение (takc-owncrship privilege), система защиты предоставляет владельцу право на доступ для записи (write-owner access) до анализа DACL (что такое привилегия захвата объекта во владение и право владельца на доступ для записи, мы поясним чуть позже).

3. Если вызывающий поток является владельцем объекта, ему предоставляются права управления чтением (read-control access) и доступа к DACL для записи (write-DACL access).

4. Из маски предоставленных прав доступа удаляется маска доступа каждого ACE типа «доступ отклонен», SID которого совпадает с SID маркера доступа вызывающего потока.

5. К маске предоставленных прав доступа добавляется маска доступа каждого ACE типа «доступ разрешен», SID которого совпадает с SID маркера доступа вызывающего потока (исключение составляют права доступа, в предоставлении которых уже отказано).

После анализа всех элементов DACL рассчитанная маска предоставленных прав доступа возвращается вызывающему потоку как максимальные права доступа. Эта маска отражает полный набор типов доступа, которые этот поток сможет успешно запрашивать при открытии данного объекта.

Все сказанное применимо лишь к той разновидности алгоритма, которая работает в режиме ядра. Его Windows-версия, реализованная функцией `GetEffectiveRightsFromAcl`, отличается отсутствием шага 2, а также тем, что вместо маркера доступа она рассматривает SID единственного пользователя или группы.

Второй алгоритм проверяет, можно ли удовлетворить конкретный запрос на доступ, исходя из маркера доступа вызывающего потока. У каждой Windows-функции открытия защищенных объектов есть параметр, указывающий желательную маску доступа — последний элемент выражения, описывающего защиту объектов. Чтобы определить, имеет ли вызывающий поток право на доступ к защищенному объекту, выполняются следующие операции.

1. В отсутствие DACL (DACL = null) объект является незащищенным, и система защиты предоставляет к нему запрошенный тип доступа.

2. Если у вызывающего потока имеется привилегия на захват объекта во владение, система защиты предоставляет владельцу право на доступ для записи, а затем анализирует DACL. Однако, если такой поток запросил только доступ владельца для записи, система защиты предоставляет этот тип доступа и не просматривает DACL.

3. Если вызывающий поток является владельцем объекта, ему предоставляются права управления чтением и доступа к DACL для записи. Если вызывающий поток запросил только эти права, система защиты предоставляет их без просмотра DACL.

4. Просматриваются все ACE в DACL — от первого к последнему. Обработка ACE выполняется при одном из следующих условий:

a. STD в ACE типа «доступ отклонен» совпадает с незаблокированным STD (SID могут быть незаблокированными и заблокированными) или SID с атрибутом проверки только на запрет в маркере доступа вызывающего потока;

b. SID в ACE типа «доступ разрешен» совпадает с незаблокированным SID в маркере доступа вызывающего потока, и этот SID не имеет атрибута проверки только на запрет;

c. Идет уже второй проход поиска в дескрипторе ограниченных SID, и SID в ACE совпадает с ограниченным SID в маркере доступа вызывающего потока.

5. В случае ACE типа «доступ разрешен» предоставляются запрошенные права из маски доступа ACE; проверка считается успешной, если предоставляются все запрошенные права. Доступ к объекту не предоставляется в случае ACE типа «доступ отклонен» и отказа в предоставлении какого-либо из запрошенных прав.

6. Если достигнут конец DACL и некоторые из запрошенных прав доступа еще не предоставлены, доступ к объекту запрещается.

7. Если все права доступа предоставлены, но в маркере доступа вызывающего потока имеется хотя бы один ограниченный SID, то система повторно сканирует DACL в поисках ACE, маски доступа которых соответствуют набору запрошенных прав доступа. При этом также идет поиск ACE, SID которых совпадает с любым из ограниченных SID вызывающего потока. Поток получает доступ к объекту, если запрошенные права доступа предоставлялись после каждого прохода по DACL.

Поведение обоих алгоритмов проверки прав доступа зависит от относительного расположения разрешающих и запрещающих ACE. Возьмем для примера объект с двумя ACE, первый из которых указывает, что определенному пользователю разрешен полный доступ к объекту, а второй отказывает в доступе. Если разрешающий ACE предшествует запрещающему, пользователь получит полный доступ к объекту. При другом порядке этих ACE пользователь вообще не получит доступа к объекту.

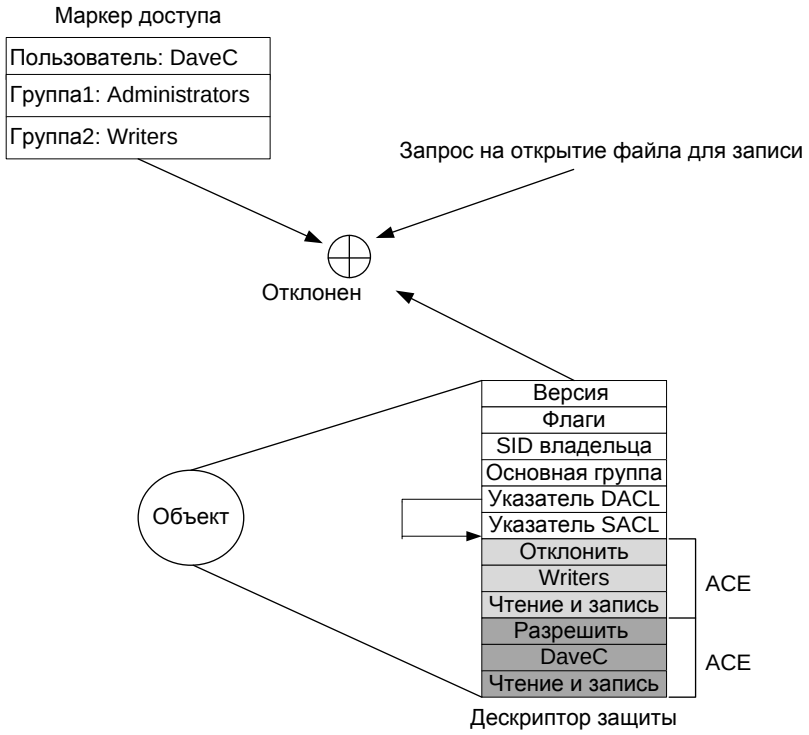

Рис. 3. Пример проверки прав доступа

Как уже говорилось, обработка DACL системой защиты при каждом использовании описателя процессом была бы неэффективной, поэтому SRM проверяет права доступа только при открытии описателя, а не при каждом его использовании. Так что, если процесс один раз успешно открыл описатель, система защиты не может аннулировать предоставленные при этом права доступа — даже когда DACL объекта изменяется. Учтите и вот еще что: поскольку код режима ядра обращается к объектам по указателям, а не по описателям, при использовании объектов операционной системой права доступа не проверяются. Иначе говоря, исполнительная система полностью доверяет себе в смысле защиты.

Права и привилегии учетных записей

Многие операции, выполняемые процессами, нельзя авторизовать через подсистему защиты доступа к объектам, так как при этом не происходит взаимодействия с конкретным объектом. Например, возможность обходить проверки прав доступа при открытии файлов для резервного копирования является атрибутом учетной записи, а не конкретного объекта. Windows использует как привилегии, так и права учетных записей, чтобы системный администратор мог управлять тем, каким учетным записям разрешено выполнять операции, затрагивающие безопасность.

Привилегия (privilege) — это право (right) учетной записи на выполнение определенной операции, затрагивающей безопасность, например на выключение

компьютера или изменение системного времени. Право учетной записи разрешает или запрещает конкретный тип входа в систему, скажем, локальный или интерактивный.

Системный администратор назначает привилегии группам и учетным записям с помощью таких инструментов, как MMC-оснастка Active Directory Users and Groups (Active Directory — пользователи и группы) или редактора локальной политики безопасности (Local Security Policy Editor).

Права учетной записи

Права учетной записи не вводятся в действие монитором состояния защиты (Security Reference Monitor, SRM) и не хранятся в маркерах. За вход отвечает функция `LsaLogonUser`. В частности, WinLogon вызывает API-функцию `LogonUser`, когда пользователь интерактивно входит в систему, а `LogonUser` обращается к `LsaLogonUser`. Эта функция принимает параметр, указывающий тип выполняемого входа, который может быть интерактивным, сетевым, пакетным, сервисным, через службу терминала или для разблокировки (unlock).

В ответ на запросы входа служба локальной безопасности (Local Security Authority, LSA) извлекает назначенные пользователю права учетной записи из своей базы данных; эта операция выполняется при попытке пользователя войти в систему LSA сверяет тип входа с правами учетной записи и по результатам этой проверки отклоняет попытку входа, если у учетной записи нет права, разрешающего данный тип входа, или, напротив, есть право, которое запрещает данный тип входа. Права пользователей, определенные в Windows, перечислены в Таблица 3.

Windows-приложения могут добавлять или удалять права из учетной записи пользователя через функции `LsaAddAccountRights` и `LsaRemoveAccountRights` соответственно, а также определять, какие права назначены учетной записи, вызывая функцию `LsaEnumerateAccountRights`.

Таблица 3. Права учетной записи

Право	Описание
Deny logon interactively, (Отклонить интерактивный вход), Allow logon interactively (Интерактивный вход)	Используется для интерактивного схода, запрос на который исходит с локального компьютера
Deny logon over the network (Отказ в доступе к компьютеру из сети), Allow logon over the network (Доступ к компьютеру из сети)	Используется для входа, запрос на который исходит с удаленного компьютера
Deny logon through Terminal Services (Запретить вход в систему через службу терминалов), Allow logon through Terminal Services(Разрешать вход в систему через службу терминалов)	Используется для входа через клиент службы терминалов
Deny logon as a service (Отказать во входе в качестве службы), Allow logon as a service (Вход в качестве службы)	Применяется SCM при запуске сервиса под учетной записью определенного пользователя
Deny logon as a batch job (Отказ во входе в качестве пакетного задания), Allow logon as a batch job (Вход в качестве пакетного задания)	Используется при пакетном входе

Суперпривилегии

Несколько привилегий дают настолько широкие права, что пользователя, которому они назначаются, называют **«суперпользователем»** — он получает полный контроль над

компьютером. Эти привилегии позволяют получать неавторизованный доступ к закрытым ресурсам и выполнять любые операции. Но мы уделим основное внимание применению привилегии на запуск кода, который выдает привилегии, изначально не назначавшиеся пользователю, и при этом не будем забывать, что это может быть использовано для выполнения любой операции на локальном компьютере.

Debug programs (Отладка программ). Пользователь с этой привилегией может открыть любой процесс в системе независимо от его дескриптора защиты.

Take ownership (Смена владельца). Эта привилегия позволяет ее обладателю сменить владельца любого защищаемого объекта, просто вписав свой SID в поле владельца в дескрипторе защиты объекта.

Restore files and directories (Восстановление файлов и каталогов). Пользователь с такой привилегией может заменить любой файл в системе на свой так же, как было описано в предыдущем абзаце.

Load and unload device drivers (Загрузка и выгрузка драйверов устройств). Злоумышленник мог бы воспользоваться этой привилегией для загрузки драйвера устройства в систему. Такие драйверы считаются доверяемыми частями операционной системы, которые выполняются под системной учетной записью, поэтому драйвер мог бы запускать привилегированные программы, назначающие пользователю-злоумышленнику другие права.

Create a token object (Создание маркерного объекта). Эта привилегия позволяет создавать объекты «маркеры», представляющие произвольные учетные записи с членством в любых группах и любыми разрешениями.

Act as part of operating system (Работа в режиме операционной системы). Эта привилегия проверяется функцией `LsaRegisterLogonProcess`, вызываемой процессом для установления доверяемого соединения с LSASS. Злоумышленник с такой привилегией может установить доверяемое соединение с LSASS, а затем вызвать `LsaLogonUser` — функцию, используемую для создания новых сеансов входа. `LsaLogonUser` требует указания действительных имени и пароля пользователя и принимает необязательный список SID, добавляемый к начальному маркеру, который создается для нового сеанса входа. В итоге можно было бы использовать свои имя и пароль для создания нового сеанса входа, в маркер которого включены SID более привилегированных групп или пользователей. Заметьте, что расширенные привилегии не распространяются за границы локальной системы в сети, потому что любое взаимодействие с другим компьютером требует аутентификации контроллером домена и применения доменных паролей. А доменные пароли не хранятся на компьютерах (даже в зашифрованном виде) и поэтому недоступны злонамеренному коду.

События аудита может генерировать диспетчер объектов в результате проверки прав доступа. Их могут генерировать и непосредственно Windows-функции, доступные пользовательским приложениям. Это же право, разумеется, есть и у кода режима ядра. С аудитом связаны две привилегии: `SeSecurityPrivilege` и `SeAuditPrivilege`. Для управления журналом событий безопасности, а также для просмотра и изменения SACL объектов процесс должен обладать привилегией `SeSecurityPrivilege`. Однако процесс, вызывающий системные сервисы аудита, должен обладать привилегией `SeAuditPrivilege`, чтобы успешно сгенерировать запись аудита в этом журнале.

Решения об аудите конкретного типа событий безопасности принимаются в соответствии с политикой аудита локальной системы. Политика аудита, также называемая локальной политикой безопасности (local security policy), является частью политики безопасности, поддерживаемой LSASS в локальной системе, и настраивается с помощью редактора локальной политики безопасности.

При инициализации системы и изменении политики LSASS посылает SRM сообщения, информирующие его о текущей политике аудита. LSASS отвечает за прием записей аудита, генерируемых на основе событий аудита от SRM, их редактирование и передачу `Event Logger` (регистратору событий). Эти записи посылает именно LSASS (а не SRM), так как он добавляет в них сопутствующие подробности, например информацию, нужную для более полной идентификации процесса, по отношению к которому проводится аудит.

Рис. 4. Конфигурация Audit Policy редактора локальной политики безопасности

SRM посылает записи аудита LSASS через свое LPC-соединение. После этого Event Logger заносит записи в журнал безопасности. В дополнение к записям аудита, передаваемым SRM, LSASS и SAM тоже генерируют записи аудита, которые LSASS пересылает непосредственно Event Logger; кроме того, AuthZ API позволяет приложениям генерировать записи аудита, определенные этими приложениями. Вся эта схема представлена на Рис. 5.

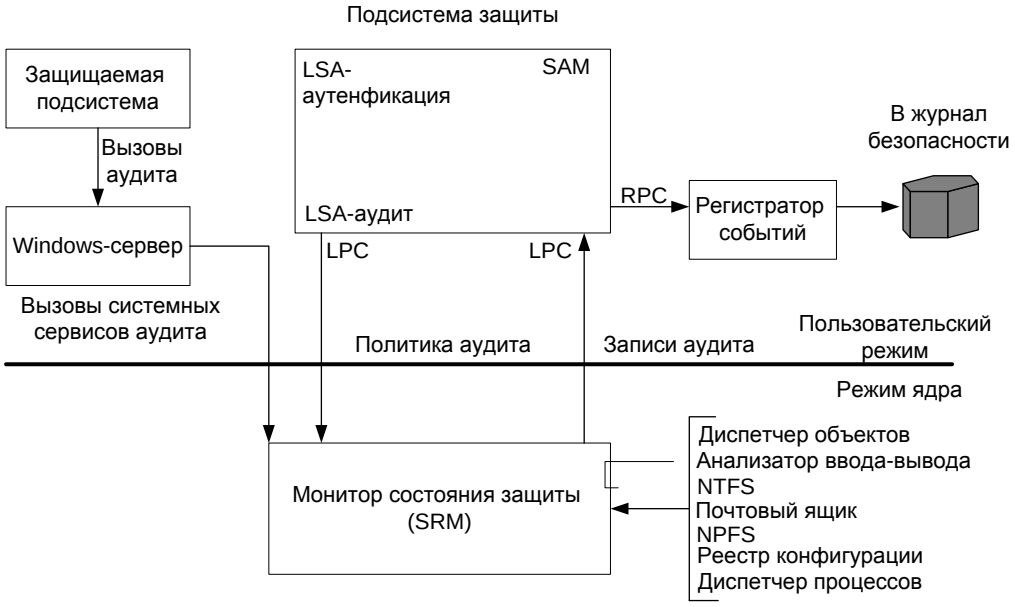

Рис. 5. Конфигурация Audit Policy редактора локальной политики безопасности

Рис. 6 обобщает изложенные в этой лекции концепции, иллюстрируя базовые структуры защиты процессов и потоков. Обратите внимание на то, что у объектов «процесс» и «поток» имеются ACL, равно как и у самих объектов «маркер доступа». Кроме того, на этой иллюстрации показано, что у потоков 2 и 3 есть маркер олицетворения, тогда как поток 1 по умолчанию использует маркер доступа своего процесса.

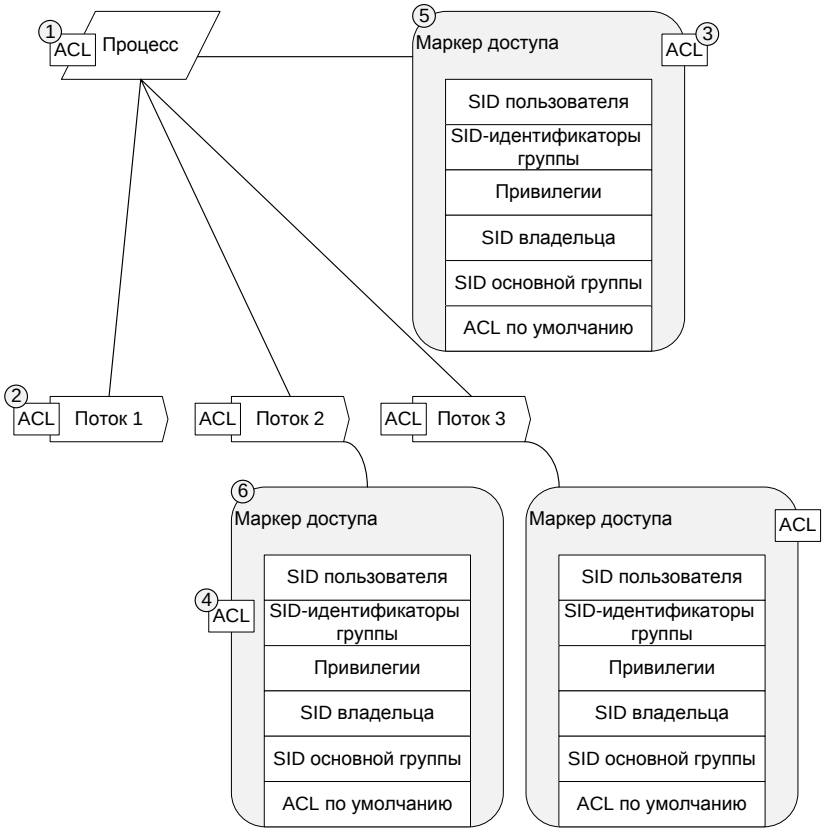

Рис. 6. Структуры защиты процессов и потоков

Вход в систему

При **интерактивном входе в систему** (в отличие от входа через сеть) происходит взаимодействие с процессами Winlogon, Lsass, одним или несколькими пакетами аутентификации, а также SAM или Active Directory. **Пакеты аутентификации** (authentication packages) — это DLL-модули, выполняющие проверки, связанные с аутентификацией. Пакетом аутентификации Windows для интерактивного входа в домен является Kerberos, а MSV1_0 — аналогичным пакетом для интерактивного входа на локальные компьютеры, доменного входа в доверяемые домены под управлением версий Windows, предшествовавших Windows 2000, а также для входа в отсутствие контроллера домена.

Winlogon — доверяемый процесс, отвечающий за управление взаимодействием с пользователем в связи с защитой. Он координирует вход, запускает первый процесс при входе в систему данного пользователя, обрабатывает выход из системы и управляет множеством других операций, имеющих отношение к защите, — вводом паролей при регистрации, сменой паролей, блокированием и разблокированием рабочих станций и т.д. Процесс Winlogon должен обеспечить невидимость операций, связанных с защитой, другим активным процессам. Winlogon гарантирует, что в ходе этих операций недоверяемый процесс не сможет перехватить управление рабочим столом и таким образом получить доступ к паролю.

Winlogon получает имя и пароль пользователя через **Graphical Identification and Authentication** (GINA) DLL Стандартная GINA — `\Windows\System32\Msgina.dll`. Msgina выводит диалоговое окно для входа в систему. Позволяя заменять Msgina другими GINA-библиотеками, Windows дает возможность менять механизмы идентификации пользователей. Например, сторонний разработчик может создать GINA для поддержки устройства распознавания отпечатков пальцев и для выборки паролей пользователей из зашифрованной базы данных.

Winlogon — единственный процесс, который перехватывает запросы на регистрацию с клавиатуры. Получив имя и пароль пользователя от GINA, Winlogon вызывает LSASS для аутентификации этого пользователя. Если аутентификация прошла успешно, процесс Winlogon активизирует оболочку. Схема взаимодействия между компонентами, участвующими в процессе регистрации, показана на Рис. 7.

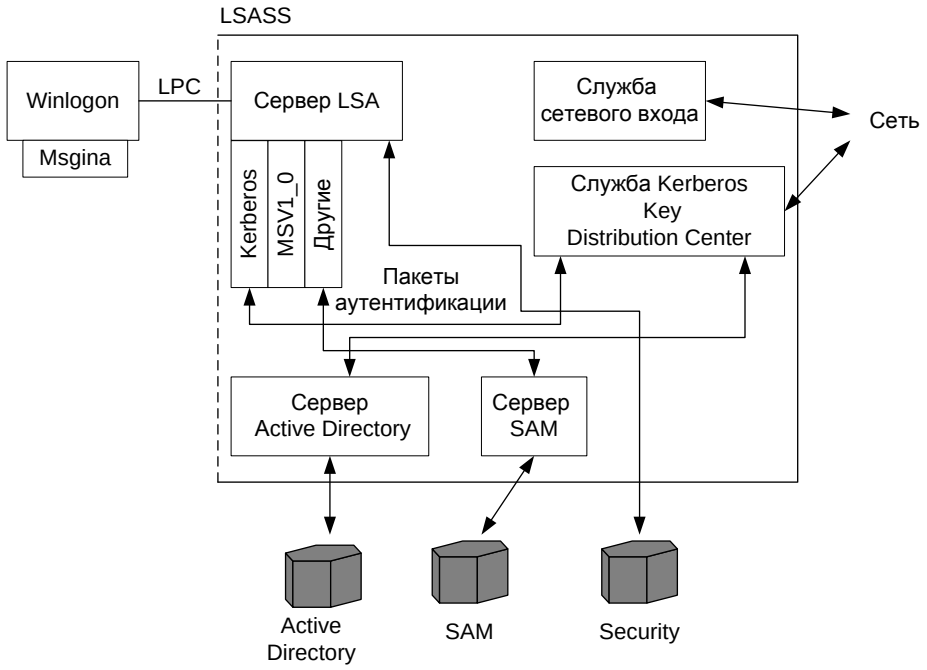

Рис. 7. Схема взаимодействия между компонентами, участвующими в процессе регистрации

Winlogon не только поддерживает альтернативные GINA, но и может загружать дополнительные DLL провайдеров доступа к сетям, необходимые для вторичной аутентификации. Это позволяет сразу нескольким сетевым провайдерам получать идентификационные и регистрационные данные в процессе обычного входа пользователя в систему. Входя в систему под управлением Windows, пользователь может одновременно аутентифицироваться и на UNIX-сервере. После этого он получит доступ к ресурсам UNIX-сервера с компьютера под управлением Windows без дополнительной аутентификации. Эта функциональность является одной из форм **унифицированной регистрации** (single sign-on).

Инициализация Winlogon

При инициализации системы, когда ни одно пользовательское приложение еще не активно, Winlogon выполняет ряд операций, обеспечивающих ему контроль над рабочей станцией с момента готовности системы к взаимодействию с пользователем.

1. Создает и открывает интерактивный объект `WindowStation` (например, `\Windows\WindowStations\WinSta0` в пространстве имен диспетчера объектов), представляющий клавиатуру, мышь и монитор. Далее создает дескриптор защиты станции с одним ACE, содержащим только системный SID. Этот уникальный дескриптор безопасности гарантирует, что другой процесс получит доступ к рабочей станции, только если Winlogon явно разрешит это.

2. Создает и открывает два объекта «рабочий стол»: для приложений (`\Windows\WinSta0\Default`, также известный как интерактивный рабочий стол) и Winlogon (`\Windows\WinSta0\Winlogon`), также известный как защищенный рабочий стол. Защита объекта «рабочий стол» Winlogon организуется так, чтобы к нему мог обращаться только Winlogon. Другой объект «рабочий стол» доступен как Winlogon, так и пользователям. Следовательно, пока активен объект «рабочий стол» Winlogon, никакой другой процесс не получает доступа к коду и данным, сопоставленным с этим рабочим столом. Эта функциональность используется Windows для защиты операций, требующих передачи паролей, я. также для блокировки и разблокировки рабочего стола.

3. До входа какого либо пользователя в систему видимым рабочим столом является объект «рабочий стол» Winlogon. После входа нажатие клавиш Ctrl+Alt+Del вызывает переключение объектов «рабочий стол» — с Default на Winlogon. (Это объясняет, почему после нажатия Ctrl+Alt+Del с рабочего стола исчезают все окна и почему они возвращаются, как только закрывается диалоговое окно Windows Security.) Таким образом, SAS всегда активизирует защищенный рабочий стол, контролируемый Winlogon.

4. Устанавливает LPC-соединение с LSASS через `LsaAuthenticationPort` (вызовом `LsaRegisterLogonProcess`). Это соединение понадобится для обмена информацией при входе и выходе пользователя из системы и при операциях с паролем.

Далее Winlogon настраивает оконную среду.

5. Инициализирует и регистрирует структуру данных оконного класса, которая сопоставляет процедуру Winlogon с создаваемым ею окном.

6. Регистрирует SAS, сопоставляя ее с только что созданным окном. Это гарантирует, что ввод пользователем SAS будет вызывать именно оконную процедуру Winlogon и что программы типа троянских коней не смогут перехватывать управление при вводе SAS.

7. Регистрирует окно, чтобы при выходе пользователя вызывалась процедура, сопоставленная с этим окном. Подсистема Windows проверяет, что запросивший уведомление процесс является именно Winlogon.

Этапы входа пользователя

Регистрация начинается, когда пользователь нажимает комбинацию клавиш SAS (по умолчанию — Ctrl+Alt+Del). После этого Winlogon вызывает GINA, чтобы получить имя и пароль пользователя. Winlogon также создает уникальный локальный SID для этого пользователя и назначает его данному экземпляру объекта «рабочий стол» (который представляет клавиатуру, экран и мышь). Winlogon передает этот SID в LSASS при вызове `LsaLogonUser`. Если вход пользователя прошел успешно, этот SID будет включен в **маркер процесса входа** (logon process token) — такой шаг предпринимается для защиты доступа к объекту «рабочий стол». Например, второй вход по той же учетной записи, но в другой системе, не предоставит доступа для записи к объекту «рабочий стол» первого компьютера, так как в его маркере не будет SID, полученного при втором входе.

После ввода имени и пароля пользователя Winlogon получает описатель пакета аутентификации вызовом Lsass-функции `LsaLookupAuthenticationPackage`. Эти пакеты перечисляются в разделе реестра `HKLM\SYSTEM\Cur-rentControlSet\Control\Lsa`. Winlogon передает пакету данные входа через `LsaLogonUser`. После того как пакет аутентифицирует пользователя, Winlogon продолжает процесс входа этого пользователя. Если ни один из пакетов не сообщает об успешной аутентификации, процесс входа прекращается.

Windows использует два стандартных пакета аутентификации при интерактивном входе: Kerberos и MSV1_0. Пакет аутентификации по умолчанию в автономной системе Windows — MSV1_0 (`\Windows\System32\Msvl_0.dll`); он реализует протокол LAN Manager 2. LSASS также использует MSV1_0 на компьютерах, входящих в домен, чтобы аутентифицировать домены и компьютеры под управлением версий Windows до Windows 2000, не способные найти контроллер домена для аутентификации. (Отключенные от сети портативные компьютеры относятся к той же категории.) Пакет аутентификации Kerberos (`\Windows\System32\Kerberos.dll`) используется на компьютерах, входящих в домены Windows. Этот пакет во взаимодействии со службами Kerberos, выполняемыми на контроллере домена, поддерживает протокол Kerberos версии 5 (ревизии 6). Данный протокол определен в RFC 1510 [11].

Пакет аутентификации MSV1_0 принимает имя пользователя и хэшированную версию пароля и посылает локальному SAM запрос на получение информации из учетной записи, включая пароль, группы, в которые входит пользователь, и список ограничений по данной учетной записи. Сначала MSV1_0 проверяет ограничения, например разрешенное время или типы доступа. Если ограничения из базы данных SAM запрещают регистрацию пользователя в это время суток, MSV1_0 возвращает LSA статус отказа.

Далее MSV1_0 сравнивает хэшированный пароль и имя пользователя с теми, которые хранятся в SAM. В случае кэшированного доменного входа MSV1_0 обращается к кэшированной информации через функции LSASS, отвечающие за сохранение и получение «секретов» из базы данных LSA (ветвь реестра SECURITY). Если эти данные совпадают, MSV10 генерирует LUID сеанса входа и создает собственно сеанс входа вызовом LSASS. При этом MSV1_0 сопоставляет данный уникальный идентификатор с сеансом и передает данные, необходимые для того, чтобы в конечном счете создать маркер доступа для пользователя.

ПРИМЕЧАНИЕ. MSV1_0 не кэширует весь хэш пароля пользователя в реестре, так как это позволило бы любому лицу, имеющему физический доступ к системе, легко

скомпрометировать доменную учетную запись пользователя и получить доступ к зашифрованным файлам и к сетевым ресурсам, к которым данный пользователь имеет право обращаться. Поэтому MSV1_0 кэширует лишь половину хэша. Этой половины достаточно для проверки правильности пароля пользователя, но недостаточно для получения доступа к ключам EFS и для аутентификации в домене вместо этого пользователя, так как эти операции требуют полного хеша.

Если MSV1_0 нужно аутентифицировать пользователя с удаленной системы, например при его регистрации в доверяемом домене под управлением версий Windows до Windows 2000, то MSV1_0 взаимодействует с экземпляром Netlogon в удаленной системе через службу Netlogon (сетевого входа в систему). Netlogon в удаленной системе взаимодействует с пакетом аутентификации MSV1_0 этой системы, передавая результаты аутентификации системе, в которой выполняется вход.

Базовая последовательность действий при аутентификации Kerberos в основном та же, что и в случае MSV1 0. Однако в большинстве случаев доменный вход проходит на рабочих станциях или серверах, включенных в домен (а не на контроллере домена), поэтому пакет в процессе аутентификации должен взаимодействовать с ними через сеть. Взаимодействие этого пакета со службой Kerberos на контроллере домена осуществляется через TCP/IP-порт Kerberos (88). Служба Kerberos Key Distribution Center (`\Windows\System32\Kdcsvc.dll`), реализующая протокол аутентификации Kerberos, выполняется в процессе Lsass на контроллерах домена.

После проверки хэшированной информации об имени и пароле пользователя с помощью объектов учетных записей пользователей (user account objects) Active Directory (через сервер Active Directory, `\Windows\System32\Ntdsa.dll`) Kdcsvc возвращает доменные удостоверения LSASS, который при успешном входе передает через сеть результат аутентификации и удостоверения пользователя той системе, где выполняется вход.

ПРИМЕЧАНИЕ. Приведенное здесь описание аутентификации пакетом Kerberos сильно упрощено, и тем не менее оно иллюстрирует роль различных компонентов в этом процессе. Хотя протокол аутентификации Kerberos играет ключевую роль в обеспечении распределенной защиты доменов в Windows, его детальное рассмотрение выходит за рамки нашей лекции.

Как только учетные данные аутентифицированы, LSASS ищет в базе данных локальной политики разрешенный пользователю тип доступа - интерактивный, сетевой, пакетный или сервисный. Если тип запрошенного входа в систему не соответствует разрешенному, вход прекращается. LSASS удаляет только что созданный сеанс входа, освобождая его структуры данных, и сообщает Winlogon о неудаче. Winlogon в свою очередь сообщает об этом пользователю. Если же запрошенный тип входа в систему разрешается, LSASS добавляет любые дополнительные идентификаторы защиты (например, Everyone, Interactive и т.п.). Затем он проверяет в своей базе данных привилегии, назначенные всем идентификаторам данного пользователя, и включает эти привилегии в маркер доступа пользователя.

Собрав всю необходимую информацию, LSASS вызывает исполнительную систему для создания маркера доступа. Исполнительная система создает основной маркер доступа для интерактивного или сервисного сеанса и маркер олицетворения для сетевого сеанса. После успешного создания маркера доступа LSASS дублирует его, создавая описатель, который может быть передан Winlogon, а cbopi описатель закрывает. Если нужно, проводится аудит операции входа. На этом этапе LSASS сообщает Winlogon об успешном входе и возвращает описатель маркера доступа, LUID сеанса входа и информацию из профиля, полученную от пакета аутентификации (если она есть).

Далее Winlogon просматривает параметр реестра `HKLM\SOFTWARE\Microsoft\Windows NT\Current Version\Winlogon\Userinit` и создает процесс для запуска программ, указанных в строковом значении этого параметра (там могут присутствовать имена нескольких EXE-файлов, разделенные запятыми). Значение этого параметра по умолчанию приводит к запуску `Userinit.exe`, который загружает профиль пользователя, а затем создает процесс для запуска программ, перечисленных в `HKCU\SOFTWARE\Microsoft\Windows NT\Current Version\Winlogon\Shell`, если такой параметр есть. Если же этого параметра нет, Userinit.exe обращается к параметру `HKLM\SQFTWARE\ Microsoft\Windows NT\Current Version\Winlogon\Shell`, который по умолчанию задает `Explorer.exe`. После этого `Userinit` завершается — вот почему Process Explorer показывает `Explorer.exe` как процесс, не имеющий предка.

Политики ограниченного использования программ

Злонамеренный код вроде вирусов и червей создает все больше проблем. В Windows XP введен механизм **Software Restriction Policies** (Политики ограниченного использования программ), который позволяет администраторам контролировать образы и сценарии, выполняемые в их системах. Узел Software Restriction Policies в редакторе локальной политики безопасности служит интерфейсом управления для политик выполнения кода на компьютере, хотя возможны и политики, индивидуальные для пользователей; в последнем случае применяются доменные политики групп.

Узел Software Restriction Policies (Политики ограниченного использования программ) содержит несколько глобальных параметров.

Параметр **Enforcement** (Принудительный) определяет, как применяются политики ограничения — к библиотекам вроде DLL, только к пользователям или к пользователям и администраторам.

Параметр **Designated File Types** (Назначенные типы файлов) регистрирует расширения файлов, которые считаются исполняемыми.

Параметр **Trusted Publishers** (Доверенные издатели) контролирует, кто имеет право решать, каким издателям сертификатов можно доверять.

При настройке параметра для конкретного сценария или образа администратор может указать системе распознавать этот сценарий или образ по его пути, хэшу, зоне Интернета (как определено в Internet Explorer) или по криптографическому сертификату, а также сопоставить его с уровнем безопасности Disallowed (Не разрешено) либо Unrestricted (Неограниченный).

Политики ограниченного использования программ применяются внутри различных компонентов, где файлы рассматриваются как содержащие исполняемый код. Некоторые из таких компонентов перечислены ниже.

Windows-функция `CreateProcess` (`\Windows\System32\Kernel32.dll`) пользовательского режима применяет эти политики к исполняемым образам.

Код загрузки DLL в Ntdll (`\Windows\System32\Ntdll.dll`) применяет эти политики к DLL.

Командная оболочка Windows (`\Windows\System32\Cmd.exe`) применяет эти политики к командным файлам.

Компоненты Windows Scripting Host, запускающие сценарии, — `\Windows\System32\Cscript.exe` (для сценариев командной строки), `\Windows\System32\Wscript.exe` (для UI-сценариев) и `\Windows\System32\Scrobj.dll` (для объектов-сценариев) — применяют эти политики к сценариям.

Каждый из этих компонентов определяет, действуют ли политики ограничения, по значению параметра реестра `HKLM\SOFTWARE\Policies\Microsoft\Windows\Safer\CodeIdentifiers\TransparentEnabled`. Если он равен 1, то политики действуют. Далее каждый из компонентов проверяет, подпадает ли код, который он собирается выполнить, под действие одного из правил, указанных в подразделе раздела `Codeldentifiers`, и, если да, следует ли разрешить выполнение. Если ни одно из правил к данному коду не относится, его выполнение зависит от политики по умолчанию, определяемой параметром `DefaultLevel` в разделе `Codeldentifiers`.

Software Restriction Policies — мощное средство для предотвращения запуска неавторизованного кода и сценариев, но только при правильном применении. Если политика по умолчанию не запрещает выполнение, то в образ, который не разрешено запускать в данной системе, можно внести минимальные изменения, и это позволит обойти правило и запустить данный образ.

Литература

1. Э. Таненбаум. Современные операционные системы. 3-ое изд. –СПб.: Питер, 2010. – 1040 с.

2. Э. Таненбаум, А. Вудхалл. Операционные системы: разработка и реализация. Классика CS. –СПб.: Питер, 2006. –576 с.

3. М. Руссинович, Д. Соломон. Внутреннее устройство Microsoft Windows: Windows Server 2003, Windows XP, Windows 2000. Мастер-класс. / Пер. с англ. -4-е изд. –М.: Издательско-торговый дом «Русская редакция»; СПб.: Питер; 2005. -992 с.

4. Microsoft Development Network. URL: http://msdn.com

5. NCSC. URL: http://www.radium.ncsc.mil/tpep

6. Рейтинги степени защищенности коммерческих ОС, сетевых компонентов и приложений. URL: http://www.radium.ncsc.mil/tpep/library/rainbow/5200.28-STD.html

7. Common Criteria. URL: http//csrc.nistgov/cc

8. Результаты оценки Windows 2000 на соответствие требованиям Controlled Access PP. URL: http://niap.nist.gov/cc-scheme.html

9. Результаты оценки Windows 2000 на соответствие дополнительным требованиям CC: http://niap.nist.gov/cc-scheme/CCEVS-VID402-ST.pdf

10. Результаты оценки Windows XP & Windows Server 2003. URL: http://niap.nist gov/cc-scheme/in_evaluation.html

11. RFC1510. URL: http://www.ietf.org