Лекция 10. Безопасность

Операционные системы

1 января 2017 г.

Идентификаторы пользователя и группы

Команда id

`id` [⟨*настройки*⟩] [⟨*имя пользователя*⟩]

Пример

```
$ id
uid=1000(xubuntu) gid=1000(xubuntu)
groups=1000(xubuntu),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(
    lpadmin),123(sambashare)
```

Характеристики идентификаторов

ОС	Разрядность	Диапазон
UNIX	15	0 . . . 32 767
Linux $<$ 2.4	16	0 . . . 65 535
Linux $\geqslant$ 2.4	32	0 . . . 4 294 967 296

Таблица 1: диапазоны идентификаторов

Идентификаторы пользователя

Вид ID	Команда	Пояснение
real ID	`id -ru`	uid настоящего владельца (от родителя). Влияет на способность подавать сигнал (uid отправителя = 0 или ruid/euid отправителя = ruid/suid получателя).
effective ID	`id -u`	определяет права доступа к общим ресурсам (общая память, семафоры, . . .)
file system ID	—	влияет на uid владельца создаваемых файлов и для проверки прав доступа для доступа к файлу (Linux).
saved ID	—	Сохраняет привилегированный euid при временном понижении уровня привилегий.

Таблица 2: виды идентификаторов пользователя

Получение идентификаторов (POSIX)

Функции getuid() и т. д.

```
#include <unistd.h>
#include <sys/types.h>

uid_t getuid(void);
uid_t geteuid(void);
int getresuid(uid_t *ruid, uid_t *euid, uid_t *suid);

int setuid(uid_t uid);
int setresuid(uid_t ruid, uid_t euid, uid_t suid);
int setreuid(uid_t ruid, uid_t euid);
int setfsuid(uid_t fsuid);
```

Установка идентификаторов

		ruid	euid	fsuid	suid
setuid(e)	euid $= 0$	e	e	e	e
setuid(e)	euid $\neq 0$	—	e	e	—
setreuid(u, e)		u	e	e	e
setresuid(u, e, s)		u	e	e	s
setfsuid(f)		—	—	f	—

Таблица 3: влияние вызовов функций на идентификаторы пользователя

Изменение системных файлов

Пример

```
$ which passwd
/usr/bin/passwd
$ ls -lF /usr/bin/passwd
-rwsr-xr-x 1 root shadow 81824 Feb 23  2009 /usr/bin/passwd*
$ ls -lF /etc/passwd
-rw-r--r-- 1 root root 1390 Nov 17  2009 /etc/passwd
$
```

Получение прав пользователя

начало
  если *атрибут* `setuid` $=$ *1*, **то**
    `euid` и `fsuid` $\leftarrow$ `uid` владельца файла;
  иначе
    `euid` и `fsuid` $\leftarrow$ `uid` во время запуска программы (`execve()`);

Рис. 1: алгоритм определения идентификаторов процесса

Случаи вызова функции `setuid()`

- `passwd` вызывает `setuid(0)` для доступа к `/etc/passwd`.
- Пользователь входит в систему $\Rightarrow$ ответвляемый процесс суперпользователя делает `setuid()` для вошедшего пользователя.

Другие случаи использования идентификаторов

Влияние идентификаторов

- Определении возможности установки параметров планирования (`nice()`, приоритет реального времени `sched_setscheduler()`, ...);
- Установки пределов ресурсов для процессов (`setrlimit()`);
- Определении максимального количества уведомлений файловой системы (`inotify_*()`).

Способности процессов

Константа	Значение
CAP_KILL	Обходить проверки ограничений при вызове `kill()`
CAP_SYS_BOOT	Разрешить вызов `reboot()`
CAP_SYS_NICE	Пропустить проверку при вызовах `nice()`
CAP_SYS_TIME	Разрешить манипуляцию с системными часами и часами реального времени
	...

Таблица 4: способности процессов (расширенные атрибуты файлов $\geqslant$ 2.6.24)

Модули безопасности

Определения

Модули безопасности Linux: (*Linux Security Modules, LSM*) — программный каркас, позволяющий определять собственные алгоритмы безопасности.

Перехватчик: функция, вызываемая ядром перед выполнением операции, затрагивающей безопасность системы, возвращает 0, если операция разрешена.

Linux с усиленной безопасностью: (*Security Enhanced Linux, SELinux*) — модуль, разработанный агентством безопасности США (National Security Agency).

AppArmor: модуль, управляющий доступом на уровне приложений, идентифицирует исполняемые файлы по путям.

Маркер доступа

Определения

Контекст безопасности: (*security context*) — текущие действующие атрибуты или правила безопасности.

Маркер доступа: (*access token*) — описание информации безопасности для сеанса.

Право: (right) — разрешает/запрещает вход в систему некоторого типа (интерактивный, . . .) — не хранятся в маркерах.

Привилегия: (privilege) — разрешает/запрещает выполнение некоторые операции, затрагивающей безопасность системы.

Домен Windows: (*Windows domain*) — набор компьютеров и связанных групп безопасности, управляемых как единое целое.

Идентификаторы защиты (SID)

Объекты идентификации

- пользователи;
- группы (локальные или доменные);
- локальные компьютеры;
- домены;
- члены доменов.

Идентификатор защиты (SID)

Определения

Идентификатор безопасности: (*Security identifier, SID*) — структура переменной длины для идентификации сущности.

Агент: сторона, выдавшая SID.

Субагент: попечитель, уполномоченный агентом.

Виды агентов

- локальная система;
- домен под управлением Windows.

Структура SID

Структура

- версия структуры SID;
- код агента идентификатора (48 бит);
- код субагента или код относительного идентификатора (RID — выбирается случайно) (32 бит, повтор n раз).

Пример SID

Пример

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

Состав

- версия структуры SID = 1;
- код агента идентификатора = 5 (центр безопасности Windows);
- коды субагентов (4 раза);
- RID = 1128.

Правила назначения SID

Назначение SID компьютеру домена

Локальный компьютер получает SID с тем же номером версии, кодом агента идентификатора, такими же кодами субагентов, что и у SID домена.

Правила назначения

- компьютеру (Windows Setup);
- локальным учётным записям (Windows) — на основе SID компьютера с добавлением RID — с 1000 и каждый раз +1;
- доменам — аналогично (`dcpromo.exe`);
- учётным записям доменов — аналогично на основе SID доменов.

Определение принадлежности SID

RID	Пользователь
500	Администратор
501	Гость

Таблица 5: RID основных пользователей

SID	Имя	Описание
`S-1-0-0`	Null	Пустая группа
`S-1-1-0`	World	Все пользователи
`S-1-2-0`	Local	Пользователи, регистрируемые на локально (физически) подключаемых терминалах

Таблица 6: общеизвестные SID

Монитор состояния защиты

Определения

Монитор состояния защиты: (*security reference monitor, SRM*) — компонент ОС (`ntoskrnl.exe`), отвечающий за:

- определение структуры маркера доступа для определения контекста зашиты;
- проверку прав доступа к объектам;
- манипулирование привилегиями (правами пользователями);
- генерацию сообщений аудита безопасности.

Вход в систему

LogonUser()

```
BOOL LogonUser(
  _In_     LPTSTR  lpszUsername,
  _In_opt_ LPTSTR  lpszDomain,
  _In_opt_ LPTSTR  lpszPassword,
  _In_     DWORD   dwLogonType,
  _In_     DWORD   dwLogonProvider,
  _Out_    PHANDLE phToken
);
```

LOGON32_LOGON_BATCH
LOGON32_LOGON_INTERACTIVE
LOGON32_LOGON_NETWORK
LOGON32_LOGON_SERVICE
...

Таблица 7: значения dwLogonType

Правило

При аутентификации система создаёт копию маркера для пользователя, каждый процесс получает его копию (изначально Userinit.exe).

Олицетворение

Определения

Олицетворение: (*impersonation*) — механизм, позволяющий процессу или потоку получать маркер другого пользователя.

Основной маркер: (*primary token*) — создан ядром, присвоен процессу при запуске.

Маркер олицетворения: (*impersonation token*) — дополнительный маркер для потока, позволяющий ему временно заимствовать профиль защиты другого пользователя.

Использование маркеров доступа

Windows API CreateProcessAsUser()

```
BOOL WINAPI CreateProcessAsUser(
  _In_opt_    HANDLE                hToken,
  /* остальные параметры CreateProcess() */
);
```

Windows API ImpersonateLoggedOnUser(), RevertToSelf()

```
BOOL WINAPI ImpersonateLoggedOnUser(
  _In_        HANDLE                hToken
);

BOOL WINAPI RevertToSelf(void);
```

Списки управления доступом

Определения

Элемент управления доступом: (*Access control entry, ACE*) — структура, содержащая набор прав доступа к охраняемому объекту и идентификатор безопасности (SID) субъекта, к которому эти права относятся.

Список управления доступом: (*Access control list, ACL*) — список прав доступа для охраняемого объекта (структур ACE).

Список управления избирательным доступом: (*Discretionary access control list, DACL*) — список управления доступом (ACL), определяющий права доступа субъектов к текущему объекту.

Системный список управления доступом: (*System access control list, SACL*) — список управления доступом (ACL), определяющий операции заданных субъектов над текущим объектом, которые должны регистрироваться в журнале аудита безопасности.

Дескриптор защиты

Определение

Дескриптор защиты: (*security descriptor*) — структура данных для описания защитной информации для охраняемого объекта. Включает:

- SID владельца;
- SID первичной группы;
- DACL (необязательно);
- SACL (необязательно);

Правила DACL

- DACL не задан (== NULL) $\Rightarrow$ все пользователи имеют полный доступ;
- DACL пуст (не содержит ACE) $\Rightarrow$ ни один пользователь не получает прав.

Уровни целостности

SID	Ур.	Имя	Описание
`S-1-16-0x0000`	0	Untrusted	Процессы, запускаемые группой `Anonymous`
`S-1-16-0x1000`	1	Low	Internet Explorer в защищённом режиме
`S-1-16-0x2000`	2	Medium	Обычные приложения при включённом UAC
`S-1-16-0x3000`	3	High	Обычные приложения при отключённом UAC, административные приложения при включённом UAC, запущенные с запросом повышения прав
`S-1-16-0x4000`	4	System	Службы, приложения системного уровня (`Wininit`, `Winlogon`, `Smss`, ...)

Таблица 8: SID некоторых уровней целостности

Условия создания ограниченного маркера

Группы

- Встроенные администраторы;
- Администраторы домена;
- Операторы архивирования;
- Криптографические операторы;
- ...

Привилегии

- SeBackupPrivilege;
- SeCreateTokenPrivilege;
- SeImpersonatePrivilege;
- SeLoadDriverPrivilege;
- ...

Отфильтрованный административный маркер

Правила создания копии административного маркера

- Уровень целостности устанавливается в Medium;
- Все упомянутые SID помечаются как имеющие силу только в отказе (deny-only);
- Удаляются все привилегии кроме некоторых (отключение, изменение часового пояса, . . .)

Пример манифеста приложения

Пример

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0">
  <trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">
    <security>
      <requestedPrivileges>
        <requestedExecutionLevel level="asInvoker" uiAccess="false"/>
      </requestedPrivileges>
    </security>
  </trustInfo>
</assembly>
```

Значения настроек манифеста

Значения `level`

`asInvoker`: как у родителя;

`highestAvailable`: наиболее высокие возможные привилегии;

`requireAdministrator`: с маркером полных административных прав.

Результат `uiAccess = "true"`

- Приложение должно иметь цифровую подпись;
- Приложение должно быть установлено только в безопасном месте (`%ProgramFiles%`, `%SystemRoot%`, ...);
- При запуске от обычного пользователя получает средний уровень целостности (между `0x2000` и `0x3000`) и высокий (`0x3000`) от администратора;
- Запрос повышения прав доступа не выводится.

Запрос на повышение прав доступа

Алгоритм вывода запроса

1. Запуск образа, требующего административных прав, активизирует службу информации приложений (Application Information Service, AIS, `Appinfo.dll` в `Svchost.exe`);
2. AIS запускает `Consent.exe`;
3. `Consent.exe` делает снимок экрана, накладывает затемнение, переключается на безопасный рабочий стол, выводит изображение в качестве фона, выводит диалоговое окно (зависящее от наличия цифровых подписей Microsoft, . . . , обычных/административных прав);
4. При отказе возвращается ошибка отказа в доступе;
5. При согласии создаётся процесс при помощи `CreateProcessAsUser()`, родителем устанавливается процесс, инициировавший запуск.

Пример запуска программы

Пример

```
SHELLEXECUTEINFO cShellExInfo =
{
  sizeof (SHELLEXECUTEINFO),     // cbSize
  SEE_MASK_FLAG_NOASYNC |
  SEE_MASK_FLAG_NO_UI |
  SEE_MASK_NOCLOSEPROCESS,       // fMask
  m_hWndPrompt,                  // hWnd
  _T("runas"),                   // lpVerb
  lpctszFilePath,                // lpFile
  lpctszParameters,              // lpParameters
  lpctszCurrentDir,              // lpDirectory
  // ...
```

Пример запуска программы (окончание)

Пример (окончание)

```
    SW_SHOWNORMAL,                  // nShow
    NULL                            // hInstApp
};
//
BOOL bSuccess = ShellExecuteEx(&cShellExInfo);
```