

§10 Контроль доступа к ~~файлам~~ ресурсам

UNIX, Win и др. ОС. Владельцы объектов

В ~~UNIX~~ реализована концепция унифицированного доступа к различным объектам системы: файлам, устройствам, памяти.

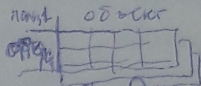
Субъектами доступа являются процессы запущенные от имени пользователя, ~~данные~~ операционной системы или программы, желающей получить некий ресурс.

Объектами доступа являются разделяемые ресурсы.

Арбитром доступа выступает ОС. и подсист. упр. файл.

Для каждого объекта доступа существует набор операций, которые с ним можно выполнять. Например для обычного файла это операции чтения, записи, удаления, выполнения, для директории операции просмотра, обращение к содержимому, создания элемента, \neq удаление.

ОС должна предоставить средства для задания прав пользователей по отношению к объектам дифференциально по операциям ~~напрямую~~ и по пользователям. Например данному пользователю разрешить читать и выполнять файл и запретить его удалить.



В качестве субъектов доступа могут выступать как отдельные пользователи, так и группы. В первом случае задание прав ~~каждому из них~~ ^{во многом является избыточным}, но это приводит к чрезмерной загрузке администратора, по выполнению рутинной работы повторение одних и тех же правил для пользователей с одинаковыми правами, поэтому используют оба подхода.

У каждого объекта доступа существуют владельцы. Владелец может быть отдельный пользо-
ватель \neq группа.

Существует особый пользователь (superuser (NetWare) administrator NT, root - Unix), который имеет все права по отношению к любым объектам доступа. Далее будет ~~вместо~~ для определенности рассматривать доступ к файлу от имени пользователя.

§2 Совместное использование файлов.

(Столдмантс, ОС)

В многопользовательской системе практически всегда необходима возможность совместного использования файлов пользователями. При этом возникают две ^{как организовать} проблемы: права доступа и ^{как} управление одновременным доступом.

Атомарные

2.1. Права доступа

Файловая система должна обеспечить возможность управляемого доступа к файлу со стороны множества пользователей. Обычно права доступа к файлу предоставляются пользователем или группам пользователей. Используются широкий диапазон прав доступа. В приведенном списке указаны права доступа, которые могут быть предоставлены определенному пользователю по отношению к некоторому файлу.

- Отсутствие. Пользователь не может обнаружить даже существование файла, не ~~узнавая~~ говоря ~~о доступе~~ уже о доступе к нему. Для обеспечения такого ограничения нужно запретить пользователю чтение пользовательского каталога, содержащего этот файл.
- Знание. Пользователь может обнаружить существование файла и установить его владельца. После этого пользователь может обратиться к владельцу для получения дополнительных прав доступа к файлу.
- Выполнение. Пользователь может загрузить и выполнить программу, однако выполнить копирование файла не может. Пользовательские программы часто бывают доступны с таким ограничением.
- Чтение. Пользователь может осуществить чтение файла для любой цели, включая копирование и выполнение. Некоторые системы могут различать чтение и копирование. В таком случае содержимое файла может быть введено на дисплей, однако выполнить копирование файла пользователю не удастся.
- Добавление. Пользователь может добавить данные в файл, часто только в его конец, но изменить его содержание или удалить содержимое файла ему не удастся, читать уже имеющиеся в файле данные ему также не разрешается. Это право полезно при накоплении данных из различных источников.
- Обновление. Пользователь может выполнять изменение, удаление или добавление данных в файле. Обычно сюда относятся операции начальной

записи в файл, полной или частичной пере-
записи, полного или частичного удаления дан-
ных, но не файла. Некоторые системы разни-
чают разные степени обновления.

- Изменение защиты. Пользователь может изменить права доступа, предоставленные другим пользователям. Обычно это право принадлежит только владельцу файла. В некоторых системах пользователь может распространить это право на других пользователей. Для защиты от злоупотребления этим механизмом владелец файла может определить, какие права могут быть изменены.
- Удаление. Пользователь может удалить файл из файловой системы.

Эти права могут рассматриваться как определенная иерархия, где обладание одним правом доступа влечет за собой обладание всеми предшествующими ему в иерархии правами. Так, если конкретный пользователь обладает правом обновления файла, то этому пользова-
телю принадлежат также права знания, выполнения, чтения и добавления.

Владелец данного файла ^{имеет субъект} ~~создавший~~ этот файл. Владелец обладает всеми перечисленными правами и может предоставить права остальным пользователям. Доступ может быть предоставлен различным классам пользователей:

Конкретный пользователь. Индивидуальные пользователи, определенные посредством своих идентификаторов.

Группы пользователей. Множество пользователей, ~~созда-~~
~~ющие посредством своих идентификаторов~~
~~определенных индивидуально~~. Система должна обладать некоторым способом отслеживания член-
ства ~~в~~ пользователей в группах.

Все. Все пользователи, имеющие доступ к системе. Файлы общего пользования доступны для всех пользователей.

2.2. Одновременный доступ.

Если права доступа позволяют добавлять или обновлять файл более чем одному пользователю, то в этом случае операци-
онной системой или системой управления файлами должен быть организован определенный порядок работы пользователей с файлом. Грубый подход позволяет пользователю заблокиро-
вать весь файл на время его обновления. Более тонкий
подход управления заключается в блокировании индивидуальных
записей при обновлении. На этапе проектирования совместно пользо-
вательского доступа необходимо решить ~~вопрос~~ ~~вопрос~~ взаимного
исключения и взаимоблокировки.

6. Модели ^{разграничения} ~~разграничения~~ доступа (разграничения доступа) ^{доступа}

(ст. Француз. стр. 58-65) (ст. Голланд. стр. 22-23) (ст. Стокгольм. стр. 22-23)

Все системы защиты строится на фундаментальном соотношении из политики информации безопасности, которая определяется способом управления доступом.

Защита чтения вверху
Защита записи внизу.

6. Основные определения.

Объект доступа - любой элемент системы, доступ к которому для субъектов доступа может быть ограничен произвольно, т.е. допускает изменение с течением времени (не имеет жесткого ограничения), иначе это не является объектом доступа. Например, файл, принтер.

Метод доступа к объекту назыв. операции, определенные для некоторого объекта. Например, для файла могут быть определены методы доступа "чтение", запись, добавление, удаление, владения, смены прав.

Субъект доступа - любая личность, способная инициировать выполнение операций над объектами. Например, пользователь.

В литературе по комп. безопасности нет единого представления о субъекте: 1. подход субъекты могут быть объектами или 2. подход субъекты и объекты не пересекаются.

Владелец доступа - субъект, которому принадлежит данный объект и который несет ответственность за целостность конфиденциальность info в этом объекте, за целостность и доступность объекта. Обычно владельцем автоматически является создатель данного объекта. Владелец может быть изменен с использованием соответствующего метода доступа к объекту. Владелец не может быть лишен некоторых прав доступа к объекту и отвечает за корректное ограничение прав доступа к объекту других субъектов.

Право доступа - право на выполнение доступа к объекту по некоторому методу или группе методов. Например, право на чтение файла.

Понятие метод доступа и право доступа не идентичны. Так, в ОС UNIX право на запись в файл, разрешает субъекту обращаться к файлу по методу записи и добавление.

Привилегия - право на доступ по некоторому методу ко всем объектам системы, поддерживающим данный метод доступа. Например в Win2k - привилегия отлаживать процессы - ко всем объектам типа процесс и поток по группе методов, исполняемых отладчиком.

6.2. Управление доступом. - ~~БД~~ Модель диспетчера доступа.

Упр. доступом является наиболее важной частью системных механизмов защиты от НСД. Характеристиками данных механизмов служат за основу классификации систем по уровню защищенности информации.

Правила упр. доступом устанавливаются администратором системы в соответствии с политикой безопасности ^{и с учетом типа объектов и систем}, а выполняются диспетчером доступа (DD), см. рис. ↓

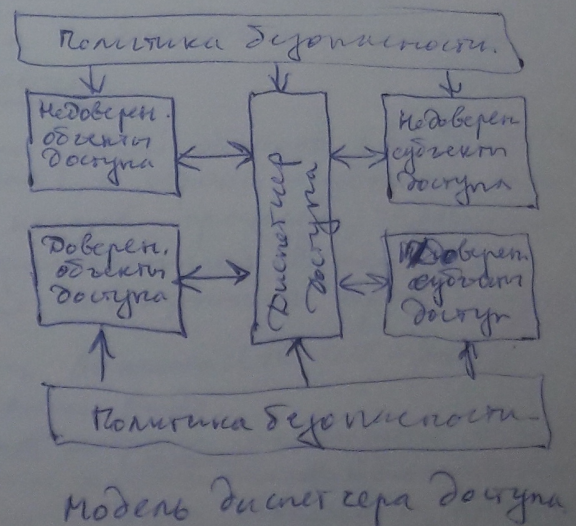
DD - подсистема, которая идентифицирует объекты и субъекты на сетевом, системном или функциональном уровне и принимает решение о предоставлении доступа на основе сравнения ^{фактор информации} ~~списков субъектов и объектов~~ (имени, пароли, адреса, метки безопасности и т.д.).

- Четкая info, контролирующая параметры СЗИ и код программ относятся к ряду доверенных объектов, такие объекты Д.Д. надежно защищены от несанкционированного чтения и модификации.
- Для чтения и изменения доверенных объектов Д.Д. предусмотрены спец. доверенные интерпретаторы, которые используют лишь доверенные субъекты.
Например, ~~доверенный~~ ^{Linux} в режиме ¹ ~~возможна~~ ^{только} с функцией консоли
- Сами правила разграничения доступа должны удовлетворять след. требованиям:
 1. Должны соответствовать правилам безопасности организации
 2. Не должны допускать разрушающие воздействия на ИС. (объекты)
 3. Любой объект доступа должен иметь владельца
 4. Запрещать присутствие недоступных объектов ни по одному методу доступа.

Существует ~~различия~~ большое количество различных моделей разграничения доступа. О них можно прочесть, например, в статье "Модели RSBAC" на <http://www.linuxcenter.ru/lib>

- Discretionary access control (DAC)
- Mandatory access control (MAC)
- Functional Control (FC)
- Модель Белла - Ла Падулы
- Модель целостности Кларка-Уилсона (CWJ)
- Secure Information Modification (SIM)
- Simone Fisher-Нивнер's Privacy Model (PM)
- Role Compatibility (RC)

Рассмотрим наиболее типичные модели разграничения доступа. DAC и MAC



Примеры Типичные модели разграничения доступа

3

6.3.4. Отсутствие разграничения доступа.

Система Правил формулирует так:

1. Система может различать пользователей, но ~~Система не различает права для разных пользователей, т.е.~~
2. Для объектов доступа не существует владельцев
3. Могут быть определены возможности доступа для каждой пары объект-метод. (например, здесь атрибуты файла: RO, Hidden - исключаются из копирования).

Примерами могут служить DOS, Windows 9x.

6.4. DAC. Избирательное (дискреционное) разграничение доступа.

Система правил для discretionary access control - DAC формулирует след. образом:

1. Для любого объекта системы существует владелец.
 2. Владелец объекта может произвольно ограничивать доступ других субъектов к данному объекту.
 3. Для каждой тройки субъект-объект-метод однозначно определены возможности доступа.
 4. Существует хотя бы один привилегированный пользователь (админ), имеющий возможность обратиться к любому объекту по любому методу доступа (для реализации механизма удаления потенциально недоступных объектов).
- Для реализации 4 правила админу может потребоваться вначале одобрить себя владельцем этого объекта (см. Win2k3) затем дать себе права и обратиться к объекту.
 - У админа есть привилегия одобрить себе владельцем любого объекта. Препятств. владелец обнаруживает смену владельца. ~~Нельзя~~
 - Во избежание злоупотреблений привилегия владения объектом является односторонней процедурой (назад "хоченка" не вернется - только себе), таким образом сохраняется ответственность за некорректные действия с объектом.
 - Для определения прав доступа субъектов к объектам при DAC используют матрицу доступа.
- Строки этой матрицы представляют собой объекты (вирт. файл, запись, сегмент, поле), а строки - субъекты (пользователи, группы польз., процессы, терминалы, каналы, узлы, приложения). В каждой ячейке хранится совокупность прав доступа.
- | | | |
|----------|---------|----------|
| | объекты | |
| | o1 | o2 |
| субъекты | s1 | n1
n2 |
| | s2 | |
- Поскольку матрица доступа очень велика (10-100 Мбайт), она не хранится в системе в явном виде.
 - Для ~~содержания~~ сокращения объема матрицы используют объединение субъектов доступа в группы (т.н. матрица с ^{атрибутом} классификацией эквивалентности).
 - Или проводят декомпозицию матрицы по столбцам (объектам) и хранят атрибуты записей (описание владельца и права субъектов) вместе с объектом.

- На практике используются два способа кодирования столбцов матрицы (прав на объект):

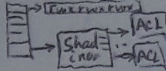
1. Вектор доступа (^{пример} UNIX/Linux) - вектор ориентированной длины, разбитый на несколько подвекторов (по субъектам). Такой способ имеет суз. ограничения на DAC.

2. ^{ACL}Список доступа (VAX/VMS, Window NT, Solaris, Linux со спец. модулем ACL) - список переменной длины, элементами которого являются структуры, содержащие:

- ид. субъекта
- права субъекта на объект
- различные флаги и атрибуты.

в Windows это модель метки безопасности, DACL, ACE, SACL

Например Linux в опции inode есть указатель на Shadow Inode.

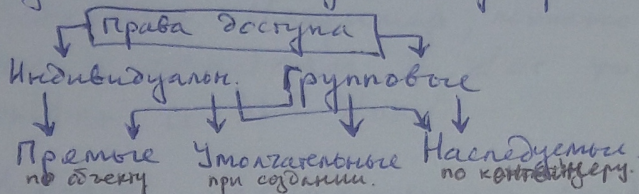


содержащий указатели на блоки с ACL

Такой способ позволяет реализовать мощный и гибкий DAC но требует больше оператив. и дисков. памяти, усложняет реализацию, ~~но~~ создает ^{возможности} проблемы ^{противоречивости} элементов списка: например, пользователю разрешено, а группе пользователей - запрещено.

Необходимо правила разрешения таких противоречий.

- При создании нового объекта владелец должен определить права доступа различных субъектов. Для упрощения работы пользователи новому объекту права могут наследоваться автоматически.



- Избирательное (дирекционное) разграничение доступа является наиболее распространенным механизмом разграничения доступа, что объясняется ~~не~~ ^{необходимостью} ~~необходимостью~~ для пользователей и простотой реализации.

- DAC ^{ACL} ~~считается~~ считается достаточным для систем коммерческого применения (Linux, Unix, Windows + большинство СУБД).

- В США ^{и России} запрещено хранить информацию, содержащую государственную тайну, в комп. системах, поддерживающих лишь избирательное огранич. доступа.

●

В.5. ^(замкнутой) Изолированная программная среда ^(Extend DAC)

5

Представляет собой расширение модели субъект-объект, доступа, DAC

Системные Правила формулируются так:

1. Для кажд. объекта сист. суц. владение.
2. Владелец произвольно ограничив. доступ друг. субъектам к своему объекту.
- New 3. Для каждой четверки субъект-объект-метод-процесс однозначно определена возможность доступа.
4. Существует хотя бы один привилег. пользов. имеющий возможность обратиться к любому объекту по любому методу.
- New 5. Для каждого субъекта определен список прог., к-рые он может запускать.

- В этой среде права субъекта на доступ к объекту определяются не только правами и привилегиями субъекта, но и процессом, с помощью которого идет обращение к объекту. Например, можно разрешить ~~доступ~~ обращаться к файлам *.doc только с помощью Open Office Writer и MS Word.
- Изолир. программная среда существенно повышает защищенность ~~от~~ ^{и данных} системы от разрушающих программных воздействий, включая программные закладки и комп. вирусы. Например, некоторый вирус попытается изменить файл *.doc, но получит отказ со стороны механизма доступа.
- Имеются сложности в администрировании, например при установке нового ПО, админ. должен модифицировать списки разрешенных прог. для пользователей.
- Изолир. прог. среда не защищает от утечки конфиденциальной info.
- Данная модель ~~реализуется~~ реализуется как в Linux так и в Windows NT, но её конфигурирование и поддержка очень трудоемки.

Пометие 6.6. Пример использования безопасности системы

6

Понятие многоуровневой системы безопасности

Когда для данных определено несколько уровней безопасности (U - unclassified - открытые info, C - confidential - для спец. подраз., S - secret - секретная, TS - top secret - сов. секретная.) обычно у военных применяют такие уровни секретности.

говорит о многоуровневой системе безопасности.

В многоуровневой системе безопасности должны выполняться следующие требования:

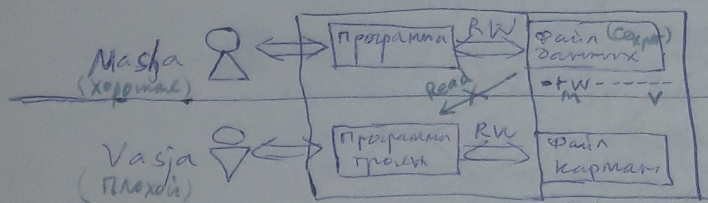
Запрет чтение вверх - простое требование безопасности. (simple security property) Любому субъекту разрешается чтение только в отношении объектов с тем же или более низким уровнем безопасности.

Запрет записи вниз - свойство звездоznа (*-property) субъект может выполнять запись только в отношении объектов с тем же или более высоким уровнем безопасности.

Пример использования: Защита от 'пропущенных копий'

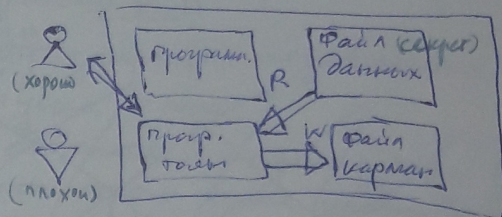
A) Описание системы

Здесь процесс используется для обхода стандартного механизма защиты: списка управл. доступом DAC



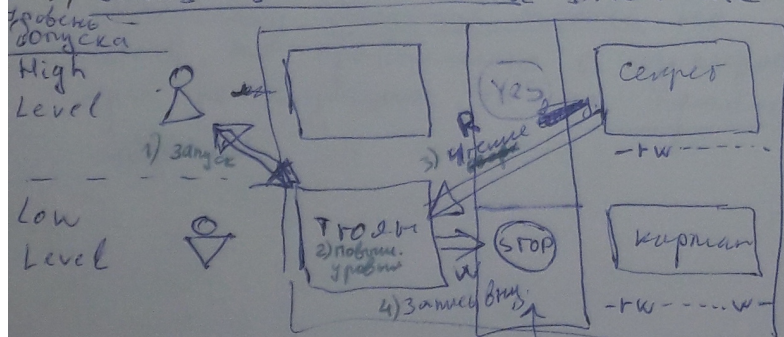
а) Создание файлов и программ со специфическими правами. Стандартная работа Mashi, установка процесса Vasya.

etc.



б) Запуск процесса Mashi, процесс-программ копирует файл данных (секрет). В файл карман. Выполняется манипулирующая работа. Теперь Vasya может читать секрет.

В) Защищенная система DAC + MAC



монитор обращений (DD)

Пусть:

Имеем 2 уровня доступа: секретный (Masha) и открытый (Vasya).

Процесс Mashi прививается к уровню доступа и запускает процесс, который повышает уровень, прочитать файл, удалив т.к. один уровень, а записать не надо из-за запрета записи вниз, далее, если есть разрешение DAC, т.к. политика безопасности имеет более высокий приоритет.

MAC-basics
6. Полномочное (мандатное) разгранич. доступу без контроля info потоков.

Mandatory access control обычно применяется в совокупности с DAC. Правила формулируются так:

1. Для любого объекта системы существует владелец.
 2. Владелец может произвольно ограничить доступ других субъектов к объекту.
 - повтор DAC. 3. Для каждой тройки субъект-объект-метод возможность доступа определена однозначно.
 - отличие от DAC 4. Существует хотя бы один привилег. пользователь имеющий возможность удалить любой объект (другие методы запрещены).
- New 5. В множестве объектов выделяется подмножество MAC, (т.е. не все объекты являются объектами полномочного (мандатного) разграничения доступа.)
- Повтор DAC 5. Каждый объект имеет атрибут секретности. Нулевое значение означает, что объект не секретен. Для 0-объекта админ может обратиться к нему по любому методу.

6. Каждый субъект имеет уровень допуска. 0 означает, что субъект не имеет допуска. Обычно администратор устанавливает уровень 0!!

- ~~Уточня: вверху запрещено~~
~~Доступ к объекту запрещен независимо от состояния атрибута допуска если:~~
- тот объект является объектом, контролируемым MAC и
 - атрибут секретности объекта строго выше уровня допуска субъекта, обращающегося к нему, $M_o > M_s$ и
 - производится попытка чтения объекта.

К объектам MAC обычно относятся только файлы. В идеале к объектам MAC следует относить только файлы, в которых может храниться секретная info (т.е. программы не относятся).

Поскольку данная модель не дает ощутимых преимуществ по сравнению с промислов. прог. средой, но сложнее при реализации, то на практике она почти не используется.

MAC extended
6.8. Полномогное (мандатное) разграничение доступа с контролем into потоков. ⑧

Обобщено применяется в совокупности с DAC, и имеет следующие правила разграничения доступа:

1. Для любого объекта системы существует владелец.
2. Владелец может произвольно ограничить доступ других субъектов к данному объекту.
3. Для каждой ~~платформы~~ субъект-объект-метод-процесс ^{время} возможность доступа определена однозначно в каждый момент времени. Т.е. если в некоторый момент времени доступ имеется, то в послед. моменты он может отсутствовать, права должны проверяться не только при открытии объекта, но и при последующей записи или чтении. \Rightarrow Существенно ~~страдает~~ ^{происходит} производительность системы.
4. Существует хотя бы один привилегир. пользователь, имеющий возможность удалить объект (хотя другие методы м.б. запрещены).
5. В множестве объектов выделяется подмножество MAC.
6. Каждый объект MAC имеет уровень секретности. К объектам админ. может обратиться по любому методу.
7. Каждый субъект имеет уров. доступа. Администр. и системным процессам назначается Ø доступ.
8. Работает правило "Чтение вверх запрещено" (NRU - Not Read Up).
9. Работает правило "Запись вниз запрещена" (NWD - Not Write Down).
10. Понижить уровень секретности объекта MAC может лишь процесс со спец. привилегией.

Использование данной модели ^{особенно правило 8} создает определенные неудобства пользователю, связанные с тем, что если уровень конфиденциальности процесса $>$ нуля, то все into в памяти процесса является секретной и не может быть записано в несекретный объект. Например,

- 1) Открываем текстов. редакторе doc1.doc несекретный
- 2) открываем doc2.doc секретной \Rightarrow уровень конфиденц. т. редактора \uparrow
- 3) Теряется возможность сохранить изменения внесенные в данном сеансе в несекретный документ.

Поэтому необходимо использовать ПО разработанное с учетом этой модели.

Второй пример: При создании нового объекта, процессом с ненулевым уровнем конфиденциальности ~~вынужден~~ ^{привычно} присвоить новому объекту уровень секретности не ниже.

6.9. Сравнительный анализ DAC и MAC моделей. (9)

Каждая из приведен. моделей разгранич. доступа имеет свои достоинства и недостатки.

Свойства модели	DAC		MAC	
	издирательн. разграничение доступа	изолирован. програм. среда	Полномочия без контро. л. потоков	разр. доступ с контро. л. потоков
✓ Защита от утечки info	Отсутствует	Отсутствует	Отсутствует	Имеется
Защищенность от разрушающих воздействий	Низкая	Высокая	Низкая	Низкая
✓ Сложность реализации	Низкая	Средн.	Средн.	Высокая
✓ Сложность администрир.	Низкая	Средняя	Вредная	Высокая
✓ Затраты ресурсов комп.	Низкие	Низкие	Низкие	Высокие
✓ Использование ПО, разраб. ботанного для других систем.	Возможно	Возможно	Возможно	Проблематично

- Из табл. видно, что MAC без контроля потоков уступает модели издир. доступа \Rightarrow что применять MAC нецелесообразно.
- Для обеспечения защищенности от утечки info придется применять MAC с контролем потоков.
- Изолирован. прог. среду применяют для обеспечения целостности программ и данных
- В остальных случаях наиболее эффективно использовать

6.10. Примеры реализации защищенных систем на Linux.

1. RSBAC - Rule Set Based Access Control
2. SELinux Security Enhanced Linux (поддерживается National Security Agency - NSA USA) разрабатывается с 2000 г. Входит в состав Fedora Core 3.
3. Medusa Security System (HP)
4. LIDS (Linux IDS)
5. MCBS3.0 ^{авт. Феникс, Утес-К} создан ~~авт.~~ в Мин. обороны России в 1997 г.
6. Trusted Linux

7. Идентификация и аутентификация

Безопасность многопользовательских систем во многом зависит от наличия в них надежных механизмов идентификации и аутентификации, — проверки подлинности. Такие механизмы предназначены для защиты от фальсификации субъектов и объектов доступа.

7.1 Взаимодействие механизмов аутоиммунитет

- ① По способу использования аутентификатора, можно выделить следующие:
- Ключевая аут. - для доступа к системе используют персональные учетные info, например, логин и пароль. Не зависит от функциональной безопасности терминалов доступа.
 - Неключевая аут. - предназначены для привилегированных пользователей, функционально локализованных в системе для административных целей (например, с функцией администрирования сервера в аппаратной ВЧ).
 - Для объектов ключевая аут. основывается на сопоставлении характеристик объекта (например, чек-лист) эталону, расположенному в доверенной области хранения.
 - Для объектов неключевая аут. - основываются на размещении этих объектов в доверенных областях хранения. Например, если ~~создается~~ разместить info в графич. системе, функционирующей как RO, то подлинность объектов в этой обл. будет обеспечена автономно.
- ② По функциональности:
- односторонняя
 - двусторонняя
- ③ По процессу аутентификации может выполняться однократно или непрерывно.
- Однократное использование механизмов аут. используется в системах с централизованным сист. доступом.
- Непрерывный механ. аут. используется в распределенной среде взаимодействия, когда не исключаются возможности перехвата, подмены, перемешивания потоков данных уже после установления сетевых соединений.
- ④ По доказательству факторов: способы аутентификации можно разделить на три группы:
- пользователь знает некоторую интро (парольная аут., и др.)
 - пользователь имеет некоторый материальный объект (пластик. карта, смарт-карта, инкредитор + тредов. паролль)
 - пользователь является некоторым физическим фактором (биометрические данные, роспись, голос, жесты, поведенческие черты (ритми, координаты), клавиш. погрешн., распознавание лица)
- ⑤ По способу доступа, различают:
- Полностью автоматизированный - пользователь предоставляет все необходимые данные для аутентификации.
 - Смешанный - пользователь предоставляет часть данных, остальную часть система запрашивает у пользователя.
 - Ручной - пользователь предоставляет данные вручную.
- ⑥ По кратности описываются:
- односторонний
 - двусторонний

7.2. Методы аутентификации сравнительная хар-ка. ②

Различные методы аутентификации целесообразно группировать между собой по следующим характеристикам: стоимость и раскрутку, простота использования, стоимость реализации, надежность.

Метод аутент.	Стоимость к. реализации	Простота использования	Стоимость реализации	Надежность
Постоян. пароли	высок	низк.	низк.	высок
Сеансовые пароли	высок	сред.	высок	сред.
Ключевые дискиеты	высок	сред.	низк.	сред.
Электронные ключи	высок.	сред.	сред.	высок.
Биометрия	сверхвысок. не оценить	высок	высоко.	низкая

2.1 Аутентификация по условию постоянного паролю ^(см. далее) — пароль это последовательность буквенно-цифровых символов, предъявляемая при входе на защищенный терминал системы.

2.2 Аутентификация по сеансовому паролю. основывается на использовании одноразовых паролей, генерируемых заново при каждом входе в систему. Алгоритм генерации пароля основывается на синхронизации пользователя и сервера, им известен закон формирования очередного пароля. Знание ^{текущего} пароля, ~~если не~~ ~~используемого~~ ~~передаваемого~~ нарушителем, не позволяет ему предсказать очередной пароль.

Обычно архитектура системы включает:

- абонентскую часть — маломощные ус-во, вырабат. сеансовые пароли
- клиентскую часть ^{несколько} функциональных серверов, где пред. аутентификация
- серверную часть — сервер аутентификации, вырабатывает очередной пароль пользователя и сообщает его функционал. серверу.

Работа системы:

1. При подключении к функ. серверу ~~абонент~~ использует очередной сеансовый пароль
2. Функ. сервер отслеживает некоторый список сеансовых паролей, полученных от сервера аутентиф.
3. В случае совпадения пароля абонента с одним из паролей списка аутентификации успешно завершается и синхронизируется состояние ~~клиента~~ абонента и сервера.

Метод сеансовых паролей обладает высокой ~~надежностью~~ ^{надежностью} стойкостью, простотой, но требует надежной защиты синхронизации пользователя и сервера, иначе возможны DOS атаки

Пример PigIPas

2.3 Аутентификация на основе ключевых дисков (3)

Вне необходимо, ключевая info размещена на отдельной дискете, содержание которой считывается и проверяется программой аутентиф. в момент входа пользователя в систему.

Метод обеспечивает высокую стойкость и прост
в реализации, но обладает низкой надежностью,
и требует спец. организационных мер исклеща-
ющих неумышленное копирование ~~на~~ дискет,

24 Аутоидентификация на основе предобученных эмбеддингов

новых ключей (элементарных методов), микропроцессор + защищенная память, смарт-карты и другие носители информации. Пластиковые карты с магнитной лентой могут содержать доп. инф. пароли.

Практически не отличается от предыдущего метода, но более дорогой, т.к. каждому входу нужно оставить нестандартном устройством стилизации info. Киров со штрих кодом, покрашен перманентными

Кирюхи со своих кодов, покрывающих немощными
способом, считывание info в индифференциальных пу-
тах, дешевые, но удобные для поддержки

25 Аугентиорикация на основе биотриггерных технологий

см. добавку на след. стр.!

Наиболее простой и ненадежный метод, однако прост в использовании и ~~не требует~~ исключает возможность копирования кнопок. носители.

В качестве биометрических характеристик могут применяться: отпечатки ^{пальцев} ~~пальцев~~ ^{копировки, фотографии, форма и размер лица,} структура радужной оболочки глаза, ^{рисунков сетчатки глаза.} фотоизображение, компоненты инфракрасного спектра (температура), геометрическая форма и размер уха, запах изо рта, ^{звук} голосовое сечение и др.

Для систем распознающих ~~аппаратов~~ биометрическую аутентификацию характерно наличие ошибок 1-го и 2-го рода.

Ошибки 2-го рода — разрешение доступа неавторизованным.
в совет. системах Риск 2 = $10^{-5} - 10^{-2}$
Ошибки 1-го рода — запрет доступа авторизованного пользов.
в совет. системах Риск 1 = $10^{-6} - 10^{-3}$

Проблема биометрии в невозможности упрочения од-
ной попытки без ухудшения второго.

Потому биометрические технологии используются, в основном, для идентификации, а не для проверки

2.6 Аудиторская деятельность на основе личных наблюдений

Перспективному направлению ~~на~~ ^{по} аутоэкологизации ~~по~~ ^{по} ~~основе~~ ^{основе} личности, особенно ~~его~~ ^{его} может стать подтверждение подлинности пользования на основе его знаний и навыков, характеризующих уровень образования и культуры.

Идентификация и аутентификация, с пом. биометрии.

- 1) Высокая достоверность из-за уникальности биометрич. признаков. Каждый человек обладает своим неповторимым набором биометрических характеристик: отпечаток пальца, рисунок сетчатки, рукописный и клавиатурный почерк, мышечный тонус и т.д.
- 2) Неотделимость от личности пользователя.
- 3) Трудность фальсификации признаков, не требуется идентифицировать угрозы кражи и подбора ключевой info перестают быть актуальными — подделка биометрич. человека как правило дороже и затратнее, чем выгоды.

Недостатки

Но практическая реализация данного механизма аутентификации создает следующие проблемы:

1. Т.к. псевдопользователи не являются людьми и не имеют биометрических характеристик, то для их Аут. должен поддерживаться альтернативный механизм.
 2. Биометрич. хар-ки двух входов всегда различаются \Rightarrow необходимо использовать мат. аппарат теории распознавания образов \Rightarrow ошибки 1-го и 2-го рода.
 3. Большинство биометрич. характеристик человека постепенно меняется во времени \Rightarrow необходимо регулярно корректировать эталонный образ.
 4. Биометрич. хар. могут испытывать разные кратковремен. измен. (после сна палец \rightarrow не влезет, простыл \rightarrow не влезет).
 5. Требуется дорогостоящая аппаратура для получения образа и сложение вычисления для сравнения с эталоном.
 6. Возможен контроль над частотой выдачи при краже биометрической базы.
- В подавляющем большинстве случаев недостатки делают биометрию неприемлемым средством ^{подконтрольной} аутентиф.

Пример использования рукописного почерка для аутентификации.

- Пользователь ставит на специальном сенсорном планшете подпись.
 - Положение рисунка фиксирует скорость движения и сила нажима.
 - Четырехмерный образ подписи (ширина, высота, скорость, сила) сравнивается с эталоном.
 - Эталон создается заранее из 20 экземпляров подписи.
 - Сравнивается степень сходства образа и эталона:
 1. если различия незначительны — успешная аутент.
 2. если различия велики — отказ аутент.
 3. если различия заметны — успешная аут. и корректировка эталона.
 т.е. отвечает плавного изменения почерка.
 - Различ. кратковременных изменений система учесть неспособна, это можно рассматривать как её достоинство: большого, пыльного или эмоционально-расстроенного — не влезет в систему.
- Например, оператор банка "под дулом пистолета" просто не сможет снять деньги со счёта, а пользователь системы во-первых не взорвет планшета под

Методы аутентификации пользователей

5

7.3. Аут. по условно поставленной паролю. ^{Используется в большинстве}
^{для аут. пользователей.} ^{достоинство: надежность}
легко ^и ^{увеличить} стоимость за счет длины.

Метод эффективен при условии ~~выбора~~ правильного ~~выбора~~ паролей и соблюдения конфиденциальности при ~~его~~ ^{вводе и} ~~использовании~~. При самостоятельном ~~вы-~~
~~боре~~ пароли пользователями, возможны организации
контроля качества ~~выбора~~ паролей. Централизован-
ное распределение паролей требует значительных
усилий по безопасному их доведению и ограни-
чивает возможность ^{по} контролю действий администратора.
Чтобы исключить легко предоставляемые пароли и сохранить воз-
можность выбора легко запоминать ~~предоставляемых~~ ^{паролей} (программы).
Применяют следующие средства:

- Обучение пользователей (материалы рекомендательны)
- Генерирование паролей (один из лучших генераторов описан в FIPS PUB 181 с текстом на С, алгоритм генерирует ^{лучшие} ^{прогнозируемые} слова (loophcrack, John the Ripper))
- Реактивные проверки паролей (loophcrack, John the Ripper)
- Запредельные проверки паролей:
 - а) простая система контроля (длины, символы, ^{сочетания} ~~слова~~ и т.д.)
в сочетании с рекомендациями пользователям
но это дает info нарушителям.
 - б) словарь потенциально простых паролей
проблема размера, скорости, охвата, атаки на структуру
 - в) марковские модели генерирования микроу-
даваемых паролей (см. Столлингс 588 стр.)
идея основана на использовании ^{сравнении} ^{частот}
триграмм в 'плохих' словах и проверяемом
пароле.
 - г) алгоритм Спассфорда основанный на
фильтрах Блума ^{с проверкой} ^{специаль-}
^{ными} ^{фильтрами} ^и ^{генерирова-}
^{нием} ^{плохих} ^{слов} ^и ^{проверяемого} ^{пароля}

✓ Существует 3 основных угрозы для подсистем аутентификации,
если для аутентификации используются пароли:

- перехват (кража пароля)
- подбор
- обход системы аутентификации.

3.1. Для обеспечения надежной защиты от кражи паролей, подси-
стема защиты должна удовлетворять след. требованиям:

- пароль, вводимый пользователем, не отображается на экране
- пароль должен храниться ~~в надежном месте~~ ^{вне} от других
- пользователей, даже от администратора.
- немедленная смена пароля при компрометации
- регулярная смена паролей
- запрет записи паролей на бумагу и т.д. ^и ^{т.д.} ⇒ ^{легко} ^{получаемые} ^{пароли}

⑥

(Грубая сила)

- Существует множество программ реализующих методы подбора паролей. Все программные данные на 2 вида:

- Первые атаки имеют сложность зависящую от скорости аутентификатора, с шифрованием и обычно медленнее, Вторые атаки напротив быстрее, т.к. проводятся в неза-
- и защищенной среде и зависят от мощности локальной системы
- и горизонтальной реализации алгоритма шифрования
- и сравнении хэш-функций.

- Т.к. пользователю легче запомнить осмысленное слово, то большинство паролей ~~не~~ представляет собой слова естеств. языка. В итоге количество возможных вариантов пароли резко сокращается.

Обычно метод подбора используется след. образом:

- 4) Подбор пароля с использованием знаний о пользователе.

Иногда пользователи, чтобы не забыть пароли используют в качестве пароля: имени, имя, дату рождения, номер телефона, номер авто или содака, номер, детей и т.д. Если иметь такую info, то для успеха атаки достаточно провести 10-20 проб. Иногда пользуют. испод. повторением пароля

остаточного при

7.5 Защита от компрометации паролей.

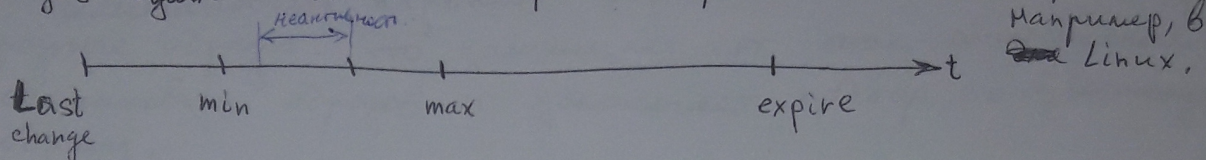
Б. Сисадмин: Ну, и пусть говорят, это использовать в качестве пароля или своего кета-дур-ноу! $RKLS - 5 < 316$, кис-вокс-ки!

Компрометация пароля - событие происходящее если, пароль пользователя стал известен другому пользователю в результате перехвата или подбора пароля.

Существует ряд методов позволяющих уменьшить ^{вероятность} компрометацию.

1. Ограничение срока действия пароля.

Может существовать несколько интервалов времени для контроля:



2. Проверка уникальности смененного пароля, необходимо помнить 5-10 ~~последних~~ образцов предыдущих паролей.

3. Ограничение на содержание пароля.

- Длина $> N$
- кол-во разных символов $> M$
- Кол-во видов символов $> K$ в первых L символах.
- отсутствие в списке "плохих" паролей.
- несовпадение с логином, именем и друг. учетной info.
- не слово и не комбинация слов

4. Активный контроль - с некоторой задержкой и для правителю и для террористов.

5. Блокировка терминала.

при многократной ошибке пользователя при вводе пароля/имени система выдает сообщение об ошибке и блокирует терминал. Параметры данного метода следующие:

- max кол-во неудачных попыток подряд
- Δt отключения счетчика неудач. попыток (время учета ошибок)
- ΔT блокировки терминала.

6. Блокировка пользователя. практически не реализуется от предыдущего.

7. Генерация паролей ОС. Когда пользователю нужно сменить пароль, он вводит соотв. команду и получает новый пароль от ОС, если пароль ему не нравится вводит еще запрос и т.д. Достоинство - сильный пароль, недостаток - неудобный ^{и не запоминается} пароль (можно использовать применение паролевого генератора или модели Спасофурда) Если пользователь входит в систему $\pi/3$ Internet, то ~~есть~~ угроза утери пароля отсутствует, и ~~также~~ такая модель аутентификации ~~идеальна~~ близка к идеальной, иначе применять ее нецелесообразно без применения генераторов удобных паролей.

8. Хранение паролей в зашифрованном виде. - обычно используют одну из известных криптографически стойких хеш-функций f для паролей ~~или~~ f^{-1} (обратной) не может быть вычислена за приемлемое время.

8

Хэш-функции должны быть обязательно криптостойкими, т.к. администратор может ~~и~~ увидеть ^{образ} или образ может передаваться по сети и быть перехвачен.

Односторонние хэш-функции для хранения паролей не отменяет зашифность хранения образов, просто это ещё один уровень безопасности. Элемент защиты.

9. Использование марканта (соли) в процедуре формирования образа пароля.

Одинаковый пароль должен соответствовать разным образам. Это защищает от атак обнаружения одинаковых образов и усложняет подбор по известному образу.

10. Пароль и отзыв. Система при входе пользователя выдаст ему случайную строку (вопрос) и ожидает правильного ответа (ответа).

~~10. Визуальный пароль.~~

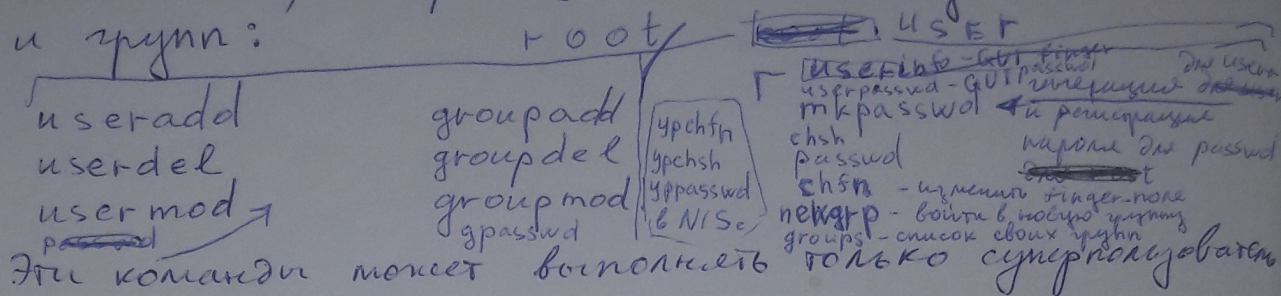
Часть из приведенных методов используются совместно.

11. Задержка (алгоритмическая) при вводе правил/карав пароли.

12. Challenge для сетевых паролей (^{ни пароль} ~~пароль~~ ^{ни} ~~хэш~~ ^{хэш} по сети не передаётся).

12.2. Создание и управление счетами пользователей.

В UNIX используют следующие команды для добавления, модификации счетов пользователей и групп:



Информация о зарегистрированных пользователях и группах для небольших систем хранится в специальных системных файлах; а для больших систем

в специальных базах данных.

Файлы: **/etc/passwd**, **/etc/shadow**, **/etc/group**, **/etc/gshadow**

Команды: **vipw** (edit **/etc/passwd**), **visg** (edit **/etc/group**)

12.2.1 Структура файла **/etc/passwd** (man 5 passwd)

Это текстовый файл, который содержит список учетных записей пользователей системы. Каждая запись представляет собой строку следующего формата:

account:password:UID:GID:GECOS:directory:shell

Файл **passwd** должен быть доступен всем на чтение, но запись в него должна быть разрешена только **root**. Записи в **passwd** зашифрованы пароли от прочтения их часто размещают в файле **shadow**, доступном для чтения только **root**.

Описание полей **passwd**:

- account** - уникально, имя пользователя в системе, не должно начинаться с буквы.
- password** - зашифрованный пароль или звездочка или X.
- UID** - уникальный идентификатор пользователя, так, что **root** в системе имеет UID=0.
- GID** - уникальный идентификатор основной группы пользователя, должен существовать в **/etc/group**.
- GECOS** - информационное поле (необязательно) содержит полное имя пользователя, office, home phone, work phone.
- directory** - домашний каталог пользователя (**\$HOME**), если не указан, то **/**.
- shell** - программа, выполняемая при входе в систему, если ничего не указано то **/bin/sh**, если указан несуществующий исполняемый файл, пользователь не сможет войти в систему с помощью **login**.

Для работы с **passwd** (Pluggable Authentication Modules for Linux Linux-PAM)

Если в поле пароль стоит *, то пользователь не сможет войти в систему с помощью **login**, но сможет с помощью **rootlogin**.

Например:

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
mail:x:500:500:mail:/var/mail:/bin/nobash
```


passwd
ehage

- A reserved field

он просто регулирует
число в ней и парит

Программа

r
o
o
t

- /usr/sbin/useradd
- (/usr/sbin/adduser → useradd)
- /usr/sbin/userdel
- /usr/sbin/usermod
- /usr/sbin/grpck
- /usr/sbin/groupadd
- /usr/sbin/groupdel
- /usr/sbin/groupmod

- это дистрибутив (добавить)
- это сборка soft.)
- это дистриб (удалить)
- это дистриб (модифицировать)
- проверка корректности /etc/group
- после удаления, рутами вытаскивает принудительно BFS
- если не удалился, то удалился
- смена GID и group name, но не состава
- смена пароля
- графич. смена пароля
- смена shell
- смена GECOS

r
o
o
t

- /usr/bin/passwd
- /usr/bin/userpasswd
- /usr/bin/chsh
- /usr/bin/chfn

- программа, запускаемая для аутентификации пользователя. Управляемое не исполняемым; не исполняемым

/bin/login

r
o
o
t

- /usr/bin/gpasswd
- gpasswd group - установка пароля
- gpasswd -a user group - добавление пользователя
- gpasswd -d user group - удаление пользователя
- gpasswd -r group - сброс пароля
- gpasswd -R group - запрет смены в эту группу

администрирование /etc/group и /etc/gshadow
графиков если сконфигурирована система с определением SHADOW

Каждая группа может иметь администраторов, пользователей и пароль.
Root может назначить администратора(ов) с которым - А и пользователей этой группы - М и максимум 16 параметров.
gpasswd [-A user1,...] [-M user1,...] groupname

r
o
o
t

- gpasswd -A user1, ... group
- gpasswd -M user1, ... group

Админ. группы может add, delete, user, удалить пароль для группы (-r), если пароль нет, то только члены группы могут использовать newgrp для работы с группой. Если пароль установлен, то члены группы должны использовать newgrp.

Не входящие в члены группы, (даже администраторы группы) не могут проводить действия от имени этой группы.

3340.7 Модель принадлежности объектов ОС UNIX (1)

С каждым процессом в UNIX связаны два идентификатора: пользователя, от имени которого был создан этот процесс Real User ID - UID и реальный идентификатор группы GID, ^{команда id это просмотр} однако при проверке прав доступа используют не эти, а эффективные идентификаторы Effective User ID - EUID и EGID.

Чаще всего EUID, EGID совпадают с UID и GID, но введение эффективных идентификаторов позволяет получить ~~процессу~~ пользователю некоторые виды доступа, которые ему явно не разрешены, но только с помощью ~~механизма~~ ограниченного числа приложений, хранящихся в файлах с установленными привилегиями смены идентификаторов. (пример такой программы является login, passwd).

Рассмотрим права доступа ~~к файлу~~ в UNIX на файл или директорию более конкретно:

- 1) права ^{на файл} определены для трех субъектов: owner, group, other. (ugo) это хорошо
команды `show` `chgrp` только если user член новой группы
команда `show` - в BSD только root в SYSV root и user но бо возвращает не 3-е
- 2) определены три операции с ~~файлами~~: read, write, execute. (rwx) для каждого из 3-х субъектов
команда `chmod`
- 3) могут быть установлены три дополнительных атрибута файла
Sticky bit - t - сохранить образ выполненного файла в кэше после завершения выполнения
setUID, SUID - s - установить UID процесса при выполнении (на время выполнения)
setGID, SGID - s - установить GID - //

- 4) Дополнительные механизмы определения прав:
Solaris 8 - ACL
Linux, FreeBSD - флаги:

символ <code>users</code> и <code>superusers</code>	<code>noappend</code>	для файла и директории ^{только root} символ <code>append</code> ^{символ} <code>append</code> ^{символ} <code>append</code>
команда <code>chflags</code> - в FreeBSD	<code>nolink</code>	файл, директория ^{нельзя удалить, но можно переместить} файл, директория <code>nolink</code> ^{нельзя удалить, но можно переместить}
команда <code>chattr</code> - в Linux (параметр <code>lsattr</code>)	<code>immutable</code>	ничего нельзя, для файла ^{удаление, изменение, переименование запрещены, для директории можно удалять файлы, но не их имя и создавать - нет.} <code>immutable</code> ^{удаление, изменение, переименование запрещены, для директории можно удалять файлы, но не их имя и создавать - нет.}

- 6) Некоторые операции может делать только root:

- mount, umount
- chroot
- mknod
- date
- chown
- лимит ресурсов
- приоритет процессов
- host-имя
- конфиг. сети
- останов системы

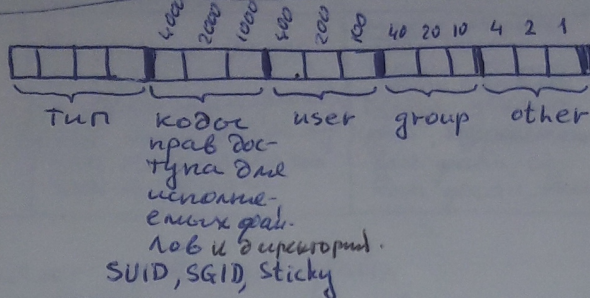
- 7) Лимиты на разд. ресурсы (диск. процес, кол. во ф., inode, block) кол. во процес. % процес. время

§4 Установка Прав доступа к файлам в UNIX

①

~~16 битов прав доступа вместе с 3 битами~~
~~содержащих вместе с битами типа файла~~
 Принадлежность файла, т.е. его UID и GID, а также

① Биты прав доступа и типа файла содержатся в ~~16~~ ~~битовом~~ слове в inode файла.



Биты доступа и тип представляются собой 16-битовое слово след. стр. 10

② Изменить права может только владелец и root.

③ Используются два способа описания прав: числовой и символический (777 и rwx).

Соответствие числовых и символических кодов. Для трёх младших битов.

oct	u	g	8-ое число	2-ое число	символ. код прав
0000	000	00	0	000	- - -
1000	100	10	1	001	- - x
2000	200	20	2	010	- w -
3000	300	30	3	011	- w x
4000	400	40	4	100	r - -
5000	500	50	5	101	r - x
6000	600	60	6	110	r w -
7000	700	70	7	111	r w x
8					
9					
10					
11					
12					
13					
14					
15					

Для старших битов 8-ое число будет как указано в табл.

③ Изменить ~~ф~~ владельца и ~~г~~ группу файла можно командами `chown` и `chgrp`.

<code>chgrp</code>	GID	new-group	файла
<code>chown</code>	UID	new-owner	файла
<code>chown</code>	UID	new-owner	GID new-group файла
<code>chown -R</code>	UID	new-owner	GID dir

④ При создании файла, ^{каталог} он получает хозяина от процесса (user) его создавшего, а group ~~е~~ либо от группы процесса, либо от группы каталога (зависит от ОС и установок файловой системы).

⑤ Проверка прав доступа производится в последовательности: 1) root, 2) хозяин, 3) ACL, 4) группа, 5) остальные.

• Узнать свой UID, GID можно командой `id`.

`id`
`id -a`
`id user`

• Узнать UID, GID и тип файла можно командой `ls`

`ls -i` (inode)
`ls -l` (long)
`ls -F` (format для типов)

2



(1)


1. W

1

7

① $-\infty$

7 r-x



①



③ $\begin{matrix} \text{---} \\ \text{---} \end{matrix}$ \rightarrow $\begin{matrix} \text{---} \\ \text{---} \end{matrix}$

North Fork
Fork
Fork


4) $t + u$

②

$-w-$

Бессмыслица

CO Graduate



①

~~type = 1000 (wage)~~

звонили
наши соседи

~~the~~ 2, a 2

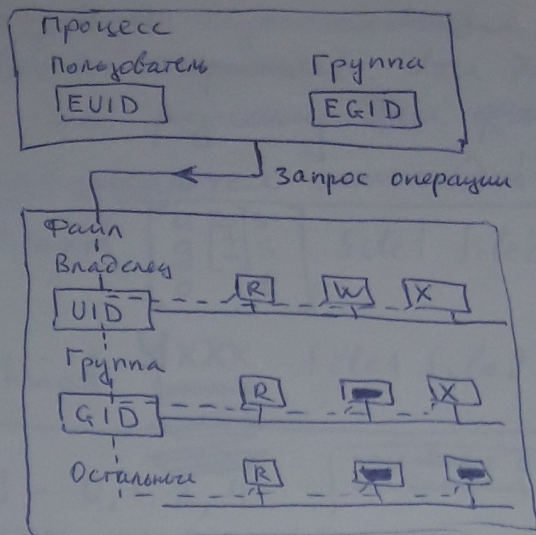
гити право
гити кнзе

- Ocean sand

26/35

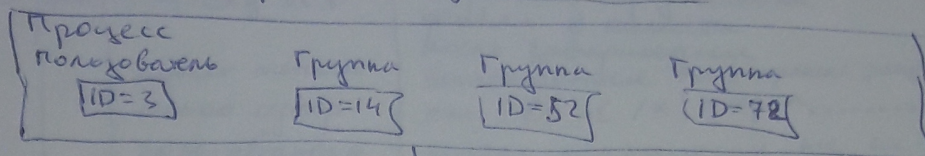
9a) Проверка прав доступа в UNIX по вектору прав.

3

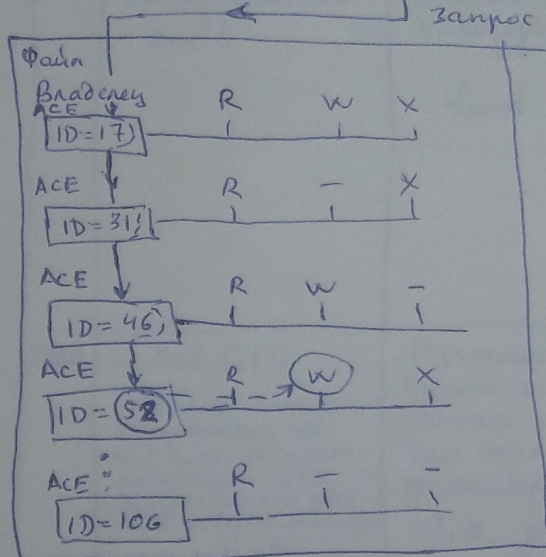


9b) Проверка прав доступа в ACL

UNIX no



ACE - Access Control Element



10

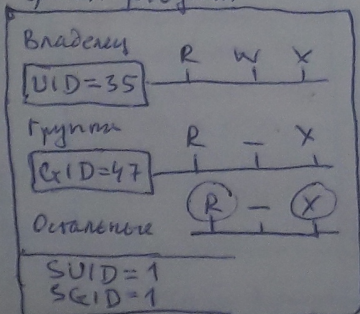
Смена эффективных идентификаторов процесса.

Процесс А (Старый)

RUID=12 EUID=12
RGID=23 EGID=23

Выполнить program

файл program



Процесс А (Продолжае)

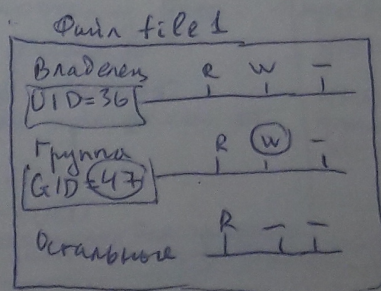
RUID=12 EUID=35
RGID=23 EGID=47

Смена идентиф.

Процесс А (Заканчивае)

RUID=12 EUID=12
RGID=23 EGID=23

Запись в file1



§6 Биты SGID, SUID, Sticky.

4

это атрибуты изменяющие стандартное выполнение различных операций. Эти биты также интерпретируются по разному для файлов и каталогов.

Устанавливаются командой chmod

```
chmod [u[=+|-]s][g[=+|-]s][o[=+|-]t] file1 file2...

chmod uXXX file1 file2...
      gXXX
      oXXX
      uXXX
```

После выполнения chmod ls -l даст следующий результат:

```
4755 -> rwsr-xr-x
6700 -> rws--S---
0644 -> rw-r--r--
1555 -> r-xr-xr-t
7777 -> rwsrwsrwt
7666 -> rwSrwsrwt
```

Бит X означает изменение на s, S, t, T

заглавные буквы отображаются, когда право X на данный файл отсутствует.

u - 0, 1, 2, 4

XXX - код прав доступа.

Симв	код	название	файл	директор
t	1000	- Sticky bit может менять только супер-пользователь root	Сохранить образ выполн. файла в памяти после завершения выполнения для ускорения запуска см. ls -l /* grep ^.....t сейчас практически не используется. find / -type d -perm -1000	Позволяет пользователю удалить только файлы которыми он владеет или имеет права на запись, несмотря на то что у каталога права drwxr-xr-x, что по идее разрешает удалять и то что угодно, но используют для /tmp.
S	2000	- Set GID может менять пользователь, но его GID, должен совпадать с GID директором и файлом в которые меняет	Изменение привилегий EUID, только на время выполнения файла имеющего этот атрибут для процессов порожденных этим файлом (программой). Т.о. запущенный файл начнет работать от имени своего UID и/или GID, а не от имени запустившего его процесса (пользователя). Полезно для passwd, которое меняет файлы /etc/passwd и /etc/shadow, потому установка SUID на файл /usr/bin/passwd делает выполнение этого процесса от имени root. find / -perm -2000 find / -perm -4000	Вновь созданные файлы этого каталога будут наследовать владельца-группу по владельцу группы каталога. Т.о. System V имитирует поведение версии BSD, у которой это правило по умолчанию. Иначе владельцы группы наследуют от процесса.
S	4000	- Set UID может менять пользователь, но его UID должен совпадать с UID файла по соображениям безопасности SUID применим только к bin. файлам, но не к скриптам !!!	Полезно для passwd, которое меняет файлы /etc/passwd и /etc/shadow, потому установка SUID на файл /usr/bin/passwd делает выполнение этого процесса от имени root. find / -perm -2000 find / -perm -4000	
1		обязательное шифрование файла	Шифрование работает только для программ а не для скриптов (за исключением скриптов perl)	

§7 Команда umask

5

Устанавливает режим доступа к вновь создаваемым файлам и каталогам, позволяет отсрочить создание файлов и каталогов с указанными правами. Общий формат команды: `umask [-S] [mode]` mode - может быть в числовой и символьной формах
`umask` - пока текущее значение
`umask nnn` - установить нового значения $nnn \in \{000-777\}$
 Как правило $u=rwx, g=rwx, o=---$ - установить маску генерируемую из указанных прав.

устанавливает в общесистемных файлах настройки профиля (`/etc/profile`), изменить который может только суперпользователь. Пользователь может установить свое значение `umask`, выполнив команду `umask`, либо задав его в собственном файле конфигурации (`~/.profile`, `~/.bash-profile`). Действие команды длится до выхода из среды или до ввода нового значения `umask`.

Команда `umask` задает восьмеричное число `nnn`, которое при создании новых файлов и каталогов вычитается из стандартных значений режимов доступа. Стандартный режим доступа устанавливается системой без использования маски.

Стандартный режим доступа для каталога 777
 Стандартный режим доступа для файла 666
 (система не позволяет создавать файлы с установленными битами выполнения, эти биты устанавливаются отдельно командой `chmod`).

указанная маска записывается в системной оболочке и наследуется дочерними процессами.
 Интерпретация значения `umask`.

Цифра в команде <code>umask</code>	Результурующий режим для файла	Результурующий режим для каталога
0	6	7
1	6	6
2	4	5
3	4	4
4	2	3
5	2	2
6	0	1
7	0	0

Из таблицы следует, что

`umask 002` устанавливает режим 664 для файла 775 для каталога
`umask 026` устанавливает режим 640 для файла 751 для каталога

В символьной форме, если права не указываются, то будут даны только группа, то для нее права `u=rwx, g=rwx, o=---` т.е. установившись с `umask u=rwx, g=rwx, o=---` группа не укажет, то для нее права не изменятся. `umask u=rwx, g=rwx, o=---` (или 0 - остальные значения) можно использовать для изменения команд `chmod`

(6)

определение режима, командой `umask`,
 Другой подход заключается в использовании следующего алгоритма.

1. Запишите полную строку режима, эквивалентную ~~444~~. числу 777.
2. Под ней запишите строку режима, соответствующую значению `umask`
3. Вычеркните из первой строки те символы, которые дублируются в тех же позициях во второй строке. Вы получите строку режима для каталогов.
4. Вычеркните из полученной строки все символы `x`. Вы получите строку режима для файлов.

Пример: Пусть значение `umask` равно 002, тогда:

- | | |
|--------------------------------------|------------------------------|
| 1. Полная строка режима | <code>rw-rwxrwx</code> (777) |
| 2. Значение <code>umask</code> (002) | <code>-----w-</code> |
| 3. Строка режима для каталогов | <code>rw-rwxr-x</code> (775) |
| 4. Строка режима для файлов | <code>rw-rw-r--</code> (664) |

Обычно разрешения на запись каталогов устанавливаются такими, `drwx-----` или `drwx--x--x` или `drwxr-x---` или `drwxr-xr-`
Задание:

1. Определите текущее значение `umask`, затем измените его на новое значение и проверьте полученные изменения.

Решение:

```
$ umask
022
$ touch file1
$ ls -l file1
-rw-r--r--  ... file1
$ umask 002
$ umask
002
$ touch file2
$ ls -l file2
-rw-rw-r--  ... file2
```

Решение:

<code>umask</code>	каталог	файл
022	755	644
027	750	640
002	775	664
006	771	660
007	770	660

2. Определите режимы доступа для каталогов и файлов для следующих значений `umask`:
 022, 027, 002, 006, 007

echo 0 > /proc/sys/kernel/cap-bound - отнять
у всех процессов все доп. права, т.е.
система без возможности суперпользователя.

echo 100000 > /proc/sys/fs/file-max - в системе
разрешено открыто 100000 файлов.

прав доступа для
Знаете ли вы? разных типов файлов различно:

- 1) Для обычных файлов права в целом имеют
тот смысл, как описаны выше r, w, x
подробно рассмотрим далее.
- 2) Для спец. файлов - устройств, именованных
каналов и сокетов права r и w имеют
такой же смысл, право x для них не
применимо, и не пишется.
- 3) Для символической связи права вообще не
используются, т.к. контролируются целе-
вым файлом, поэтому ОС ставит права $rwxrwxrwx$
- 4) Для каталогов права доступа не столь
очевидны. r и w ^{подробно} рассмотрим далее.
 - r - просмотр имен (и только имен)
файлов находясь в каталоге.
 - x - для полн. доп. инфа $ls -l$ сис-
тема должна записать в метаданные
файлов, то требует права x .
Если вы хотите перейти в каталог
то тоже право x , а также x
для всех каталогов в пути к
целевому. (темные каталоги).
 x без r
 - w - удаление и добавл. файлов (не
учит. права самого файла).
но без x - это не достижимо.

Изменение групповой собственности на файлы и директории.

- Чтобы user мог быть членом одной и более групп для просмотра в какие группы вы входите используйте команду \$ groups.
\$ groups
vasja.

- Для просмотра в какие группы входит другой пользователь используйте команду groups username.
vasja \$ groups root
root: root, bin daemon sys adm wheel

Ваши файлы и директории принадлежат одной из групп в которую вы входите, ~~это~~ это известно как понятие групповой собственности.

- Для добавления user в новую группу используют usermod -G group username.
Если он уже есть, то удалится из группы.
- Чтобы просмотреть групповую собственность на ваши файлы используйте команду \$ ls -gl

- Вы можете изменить групповую собственность на файлы или директории командой:
\$ chgrp group-name file|directory-name

Вы должны быть членом ^{той} группы на которую ссылается в этой команде.

- Чтобы изменить текущую группу ^(GID) используют newgrp. Вы должны войти в список той группы на которую меняете, ~~без~~ без указания группы изменение ^{password} на стандартно заданную в файле /etc/passwd происходит при выполнении процесса login (без пароля) для новой группы.

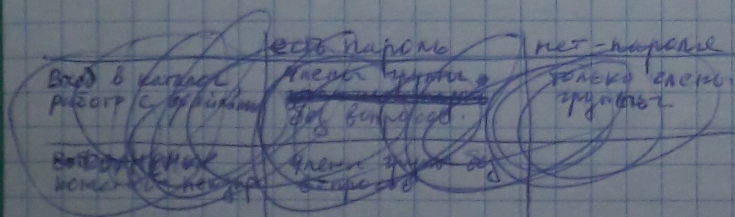
\$ newgrp

или
\$ newgrp students
используйте id

Создаваемые файлы будут принадлежать группе students. При этом не проверяется GID при доступе к объектам (автоматически).

- Чтобы создать новую группу # groupadd students

- Чтобы манипулировать группой passwd (может использовать root и администратор группы).



Безопасность Windows NT

У win2k используется графический интерфейс работы с группами и ресурсами.

Конкретным действием управления 2 элементами

1. Маркер доступа - связывается с конкретным процессом;
2. Детектор записи - связывается с конкретным объектом.

Маркер доступа

идентификатор доступа SID
идентификатор доступа группы привилегий
Получиваемые по умолчанию
Символ конкретной группы ACL

Факты
Виды
Системы. системы контроля
достоинства (SACL)
Системы организационного
контроля качества
(OACL)

ACL - access control list

заголовоч	АСГ
заголовоч	АСЕнгу
наименование	
идентификатор	идентификатор

Маска досипна

Алгебраические различия: год, ив, те, ес, кте

NTFS Permissions

V. 1.2 - NT3.51, NT4.0

V. 3.0 - W2K

V. 3.1 - WXP, W2K3

Синхронизатор базовые и особые разрешения для папок и файлов.

Базовое разрешение	Значение для папок	Значение для файлов
Чтение (Read)	Обзор папки	просмотр содержимого
Запись (Write)	Добавление ф. и подпапок	запись данных
Чтение и выполнение (Read & Execute)	Обзор папки, наследуется ф. и подпапками	просмотр содержимого и запуск исполн. ф.-ла
Список содержимого папки (List Folder Contents)	Обзор папки, наследуется только подпапками	Не применимо
Изменить (Modify)	Обзор и создание ф. и подпапок, удаление папки	Чтение, запись, удаление переименование.
Полный доступ (Full Control)	Все.	Все

Базовые разрешения образуются объединением особых разрешений.

Особые разреш.	Full Control	Modify	Read & Execute	List Folder Content	Read	Write
Traverse Folder/Execute File	X	X	X	X		
List Folder/Read Data	X	X	X	X	X	
Read Attributes	X	X	X	X	X	
Read Attributes	X	X	X	X	X	
Read Ext. Attr.	X	X	X	X	X	
Create File/Write Data	X	X				X
Create Folders/Append Data	X	X				X
Write Attribute	X	X				X
Write Attribute	X	X				X
Write Ext. Attr.	X	X				X
Write Ext. Attr.	X	X				X
Delete Subst. & Files	X					
Delete	X	X				
Delete	X	X				
Read Permis.	X	X	X	X	X	X
Change Permis.	X					
Take Ownership	X					

Можно наследовать разрешение от родит. папок и использовать флаги

☐ Inherit from parent ... Унаследовать от родителей

☐ Replace permission ... Заменить для дочерних

Иерархия разрешений

Явный Deny
Явный Allow
Унаследованный Deny
Унаследованный Allow