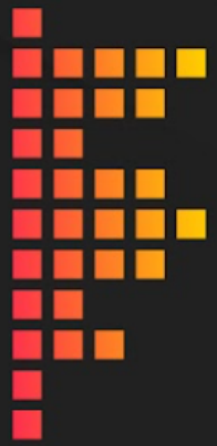


Пользователи и группы. Авторизация и аутентификация

☑ Проверка связи

- ☐ видно
- ☐ слышно
- ☐ rec

Анатолий Бурнашев

МТС.Digital	**Эксперт центра практик Надежность**
30+ лет в ИТ	10+ лет в ИТ-архитектуре
tg.me/ayburnashe	Буду рад знакомству

Пользователи и группы

Авторизация и аутентификация

AAA

- Authentication
- Authorization
- Accounting

Authentication

- Аутентификация, идентификация,процесс подтверждения пользователем своей “подлинности”.
- Ввод логина и пароля.

Authorization

- Авторизация, процесс наделения пользователя правами
- Предоставление доступа к каким-либо объектам

Accounting

- Запись информации о произошедших событиях

Файловая система

/etc/passwd

Файл /etc/passwd содержит следующие записи, разделенные двоеточиями:

- Имя пользователя
- Зашифрованный пароль
- Цифровой идентификатор пользователя (UID)

- Цифровой идентификатор группы пользователя (GID)
- Полное имя пользователя
- Домашний каталог пользователя
- Оболочка входа в систему

/etc/shadow

- Регистрационное имя.
- Зашифрованный пароль.
- Дата последнего изменения пароля.
- Минимальное число дней между изменениями пароля.

- Максимальное число дней между изменениями пароля.
- Количество дней до истечения срока действия пароля
- Количество дней по истечении срока действия пароля
- Срок действия учетной записи.
- Зарезервированное поле, которое в настоящее время всегда пустое.

Классическая аутентификация

- шелл должен существовать и быть перечисленным в /etc/shells
- root может залогиниться только с терминала, перечисленного в /etc/securetty

Блокировка пользователя делается несколькими методами:

- Установка shell в /bin/nologin
- Установка expiry date в прошлое
- Блокировка пароля (делаем его непроверяемым, добавляя "!")

Утилиты

- useradd
- passwd
- usermod
- userdel
- groupadd

- groupdel
- groupmod
- groups
- id
- newgrp

- gpasswd
- chgrp
- chown
- chmod

Типы пользователей

- root - суперпользователь
- Системные пользователи - системные процессы у которых есть учетные записи для управления привилегиями и правами доступа к файлам и каталогам.
  Создаются системой автоматически.
- Обычные пользователи - учетные записи пользователей, допущенных к управлению системой. Создаются системным администратором.
- Файл настроек для управления политиками при создании пользователей /etc/login.defs

flag	user	grop	other
#	rwx	rwx	rwx

flag	описание
-	
l	символическая ссылка
d	директория
b	блочное устройство
c	символьное устройство
p	канал (fifo)
s	unix socket

Маски прав

oct	bin	mask
0	000	---
1	001	--x
2	010	-w-
3	011	--wx
4	100	r--
5	101	r-x
6	110	rw-
7	111	rwx

SUID, SGID

- Восьмеричные значения для SUID и SGID - 4000 и 2000.
- Символьные: u+s и g+s.

- Для исполняемого файла
  - Файл будет исполнятся с UID/GID владельца файла
- Для директории
  - то все файлы созданные в ней будут наследовать UID/GID директории

Sticky bit

- Восьмеричные значения для sticky bit - 1000
- Символьные: +t.

- Для директории
  - Каталог с установленным sticky-битом означает, что удалить файл из этого каталога может только владелец файла или владелец каталога или суперпользователь.
  - если несколько пользователей пишут в один каталог, то при установленном sticky-bit они могут удалять только свои файлы

umask

- вычитаемая маска для определения прав файлов и каталогов при создании

- полная маска для каталога 777
  - umask - 022
  - созданный каталог с правами 755
- полная маска для файла 666
  - umask 0222
  - созданный файл с правами 644

capabilities

capabilities - это средства для управления привилегиями, которые в традиционных Unix-подобных системах были доступны только процессам, запущенным с правами root (uid==0).

- man capabilities

CAP_SYS_*

- CAP_SYS_ADMIN - Разрешить диапазон системных административных операций
- CAP_SYS_BOOT - Разрешить вызовы к reboot
- CAP_SYS_CHROOT - Разрешить вызовы к chroot

- Флаги (битовая маска) установки дополнительных прав на файлы и запущенные процессы, предоставляющие расширенные системные права без использования suid (sudo)
- В Linux каждый процесс (задача) имеет пять 64-битных чисел (наборов), содержащих биты разрешений (до Linux 2.6.25 они были 32-битными), которые можно посмотреть в /proc/<pid>/status

capabilities - для процесса

- Permitted - требуется запрос на выполнение
- Inheritable - можно унаследовать потомкам
- Effective - какие действия может выполнить процесс

- Bounding (ограничивающий набор) — до Linux 2.6.25 был общесистемным атрибутом, общим для всех потоков, предназначенным для описания набора, за пределы которого
  разрешения расширяться не могут. В настоящее время это набор для каждой задачи и является лишь частью логики execve
- Ambient (наружные, начиная с Linux 4.3) — добавлены, чтобы легче предоставлять разрешения не-root пользователю, без использования setuid или файловых разрешений

capabilities - для файла

- Permitted - разрешенный набор для файл
- Inheritable - список свойств которые можно унаследовать от парента
- Effective - бит разрешения для permitted set

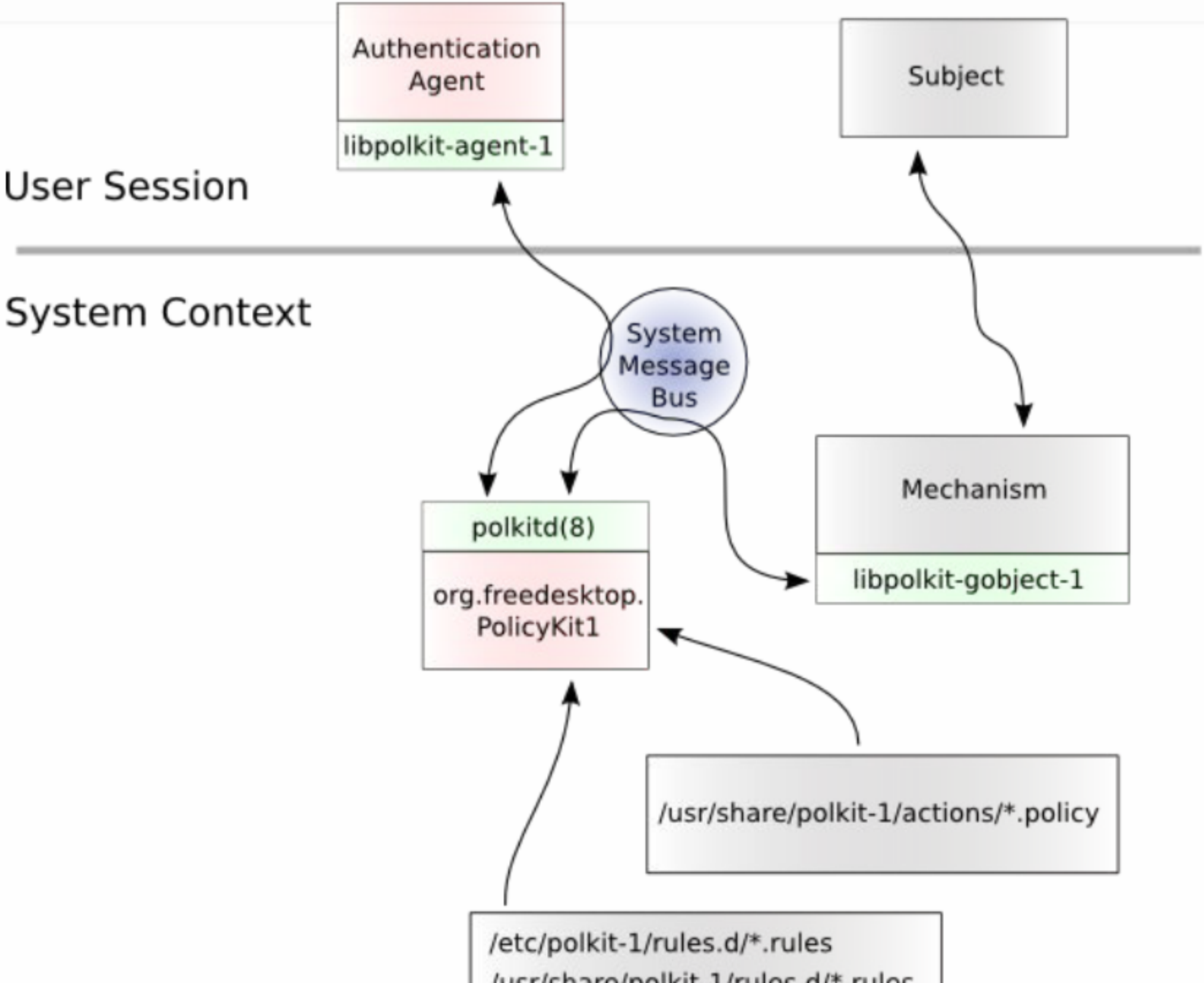
утилиты

- setcap
  - устанавливает и удаляет capabilities на исполняемый файл
  - setcap cap_net_admin,cap_net_raw+eip /usr/sbin/tcpdump
  - setcap -r /usr/bin/ping cap_net_raw

- getcap/getpcaps
  - узнать установленные capabilities
  - getcap /usr/sbin/tcpdump
  - getpcaps <pid>

capsh - тестирование установок capabilities

```
su - vagrant
export -p > ./savedenv
sudo capsh --caps="cap_net_raw+eip
cap_setpcap,cap_setuid,cap_setgid+ep" --keep=1 --user=vagrant -- -c
"source ./savedenv; rm ./savedenv; /usr/bin/env bash"
capsh --print
whoami
echo $$
```


Authentication Agent
libpolkit-agent-1
Subject
User Session
System Context
System Message Bus
polkitd(8)
org.freedesktop.PolicyKit1
Mechanism
libpolkit-gobject-1
/usr/share/polkit-1/actions/*.policy
/etc/polkit-1/rules.d/*.rules

- pkaction - служит для просмотра возможных действий, которые отслеживает PolicyKit.
- pkcheck - позволяет проверить, авторизовался ли процесс для выполнения действия.
- pkexec - позволяет пользователю выполнить действие или программу от имени другого пользователя.
- pkttyagent - позволяет выполнить текстовую авторизацию таким приложениям, которые запускаются без пользовательского графического окружения, например, ssh

PolKit: пример

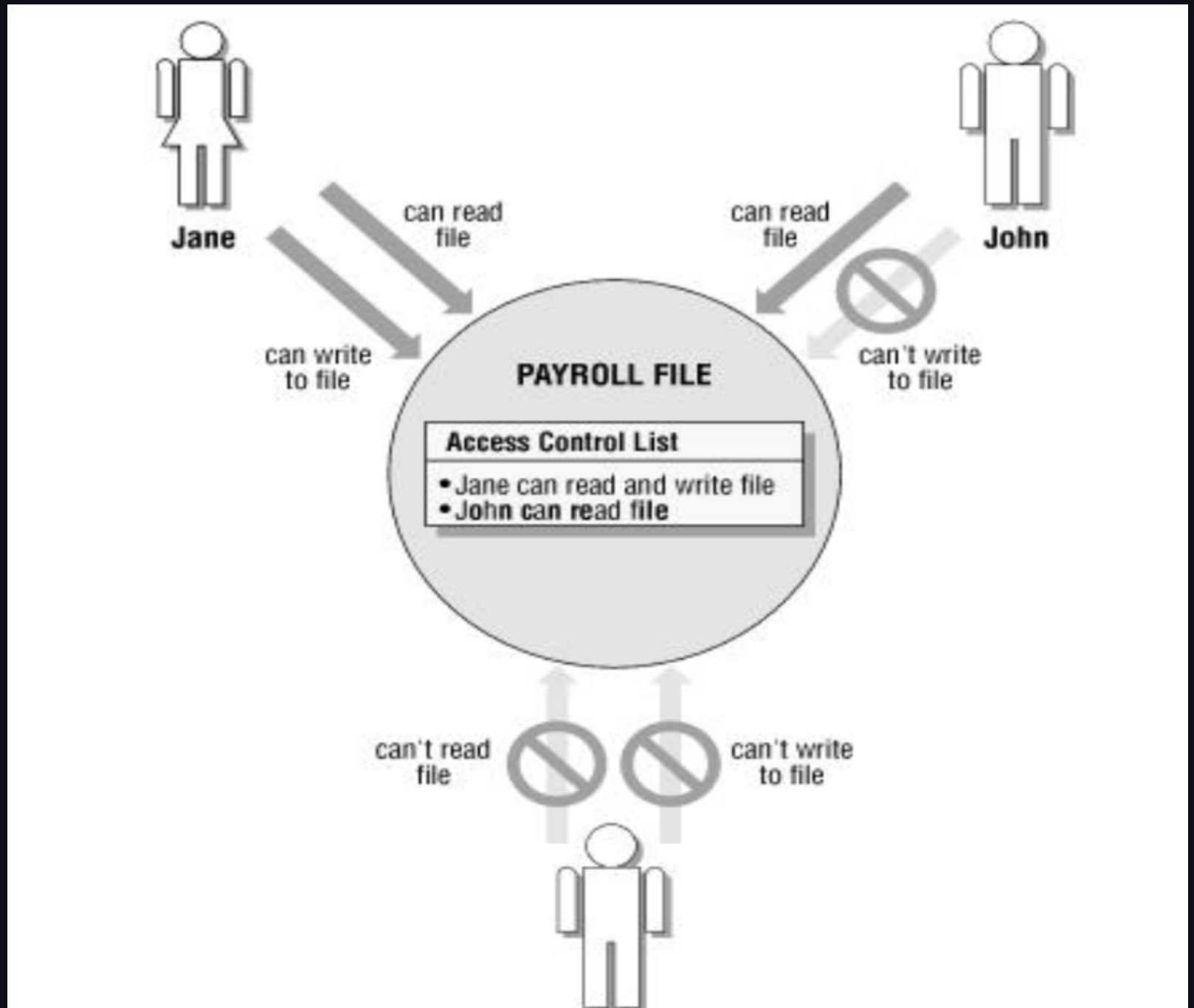
Включим логгирование

```
/etc/polkit-1/rules.d/00-access.rules

polkit.addRule(function(action, subject) {
  polkit.log("action=" + action);
  polkit.log("subject=" + subject);
});
```

PolKit: пример2

Право на запуск сервисов systemd

```
/etc/polkit-1/rules.d/01-systemd.rules

polkit.addRule(function(action, subject) {
  if (action.id.match("org.freedesktop.systemd1.manage-units") &&
subject.user === "otus") {
   return polkit.Result.YES;
  }
});
```


Jane
John
can read file
can read file
can write to file
can't write to file
PAYROLL FILE
Access Control List
• Jane can read and write file
• John can read file
can't read file
can't write to file

ACL: пример

есть папка upload с правами 777. Все пишут туда должен быть юзер который может удалять все в этой папке

```
# mkdir /usr/local/upload
# chmod 777 /usr/local/upload
# setfacl -m d:u:vagrant:rwx -R /usr/local/upload/
# getfacl /usr/local/upload

# su - otus
$ mkdir /usr/local/upload/test1
$ touch /usr/local/upload/test1/myfile
# su - vagrant
$ rm -f /usr/local/upload/test1/myfile
```

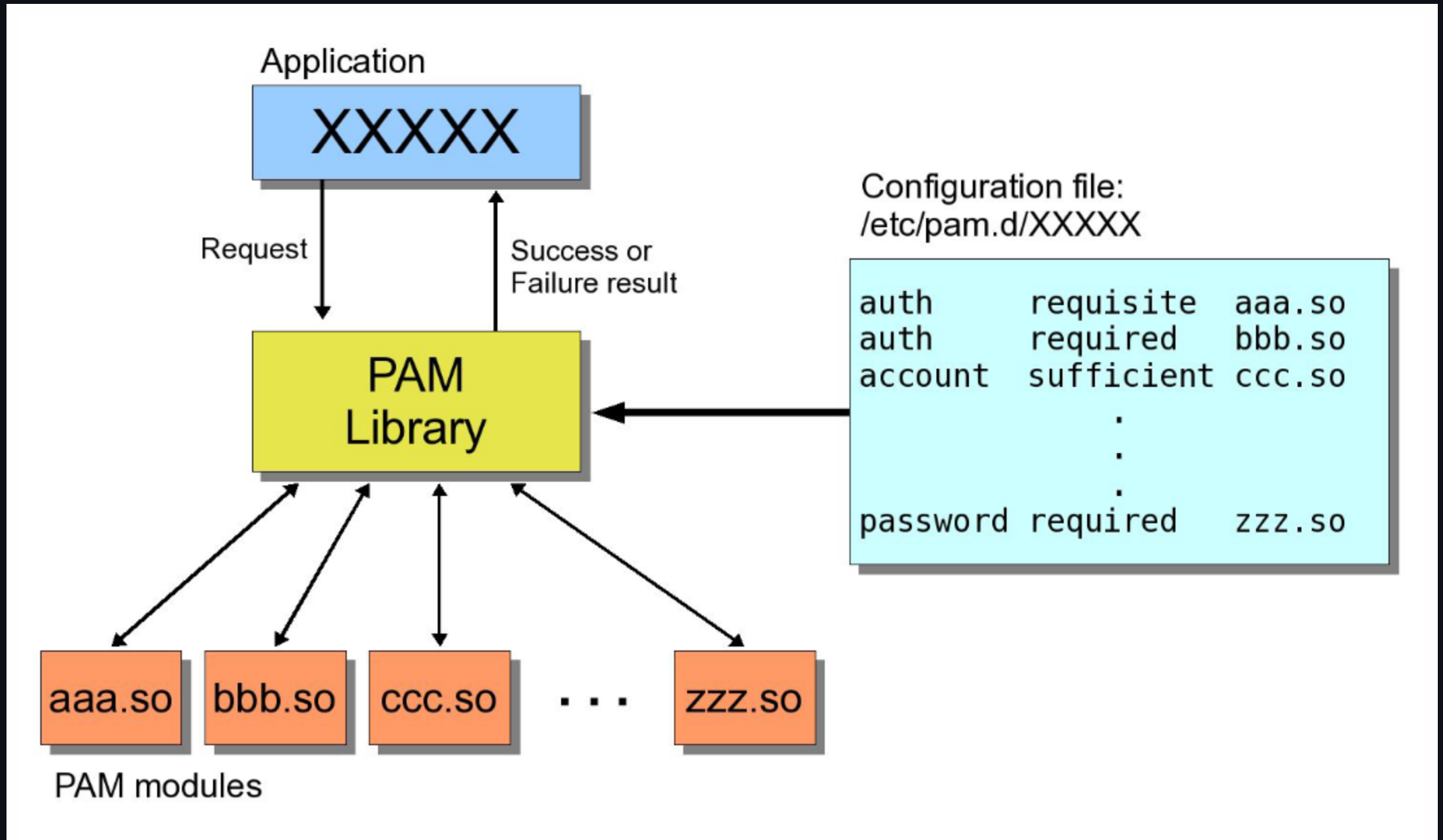
Расширенные аттрибуты ФС

- chattr
- lsattr

аттрибуты:

- a: - файл может быть открыт только в режиме добавления
- A: - не обновлять время перезаписи
- c: - автоматически сжимать при записи на диск
- C: - отключить копирование при записи
- i: - сделать неизменяемым
- s: - безопасное удаление с последующей перезаписью нулями

PAM


Application
XXXXX
Request
Success or
Failure result
PAM
Library
Configuration file:
/etc/pam.d/XXXXX
auth requisite aaa.so
auth required bbb.so
account sufficient ccc.so
.
.
.
password required zzz.so
aaa.so
bbb.so
ccc.so
. . .
zzz.so
PAM modules

- Конфигурация для каждого сервиса определяется в формате:
  type control module-path module-arguments
- В конфигурации, внутри каждого стека (type) есть один или несколько модулей, которые просматриваются по порядку (сверху-вниз) и,
  в зависимости от control-a(sufficient, required) проверка либо продолжается,
  либо прекращается.

PAM: типы модулей

1. Модуль аутентификации используется для аутентификации пользователей или создания и удаления учетных данных.
2. Модуль управления учетными записями выполняет действия, связанные с доступом, истечением учетных данных или записей, правилами и ограничениями для паролей и т. д.
3. Модуль управления сеансами используется для создания и завершения сеансов.
4. Модуль управления паролями выполняет действия, связанные с изменением и обновлением пароля.

PAM control

- required - Для успешного завершения проверки этот модуль должен сработать, проверка продолжается вне зависимости от результата.
- requisite - как required, только при ошибке проверка прекращается. Возвращается ошибка первого required/requisite модуля вернувшего ошибку.
- sufficient - При успехе возвращается OK и проверка завершается
- optional - опциональный модуль. Успех или Ошибка важны только в случае если это единственный модуль в стеке.
- include - включить в текущий стек модулей стек модулей из файла
- substack - включить в текущий стек результат модулей из файла

PAM: пример

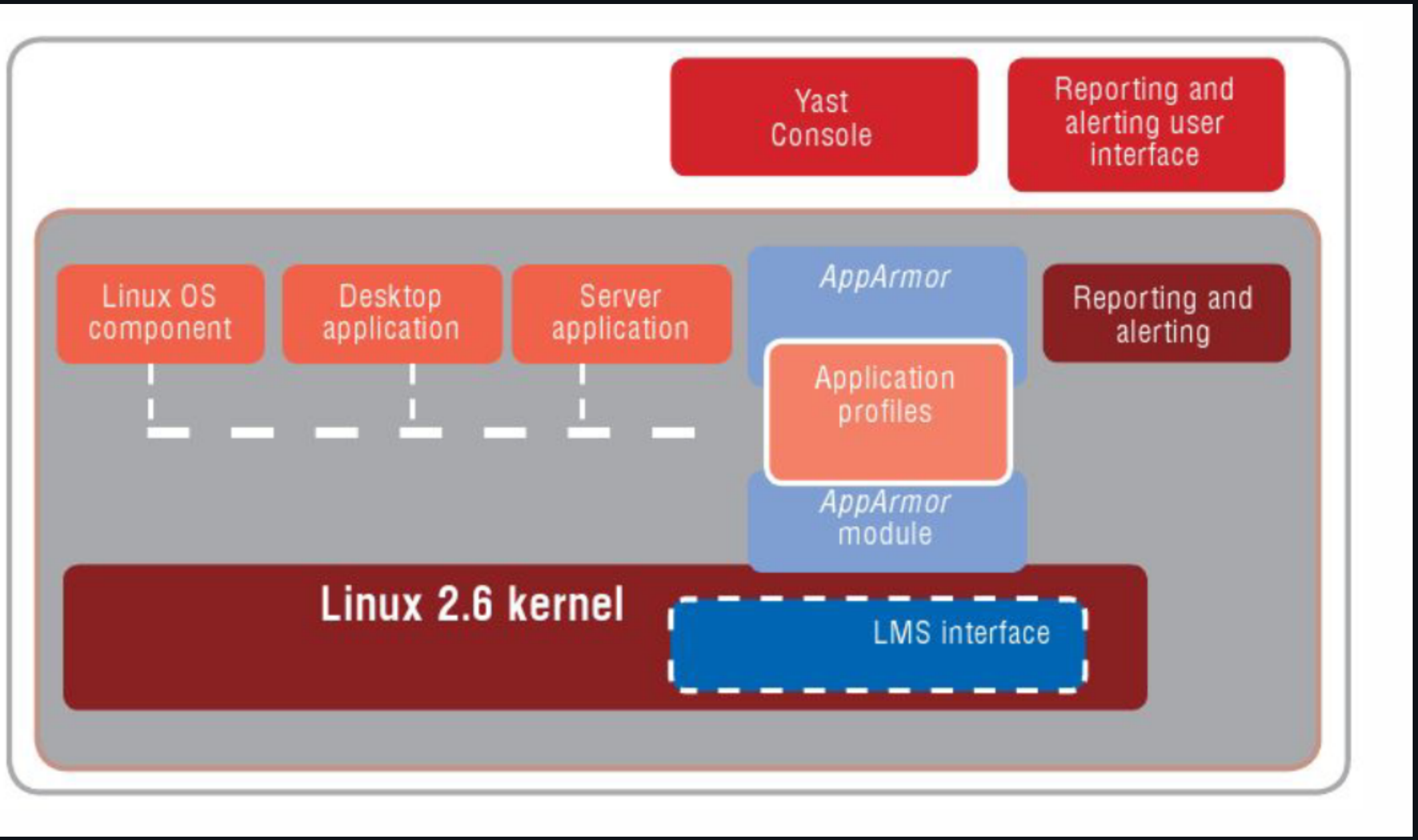
С помощью пакета pam_script можно быстро и гибко расширять AAA-процесс и использовать возможности PAM.

- pam_succeed_if - разрешает вход только заданным группам auth required pam_succeed_if.so gid=1000,2000

- pam_deny - все запретить
- pam_unix - проверка пароля через shadow
- pam_mysql - проверка пользователя через mysql
- pam_cracklib - установка политик пароля
- pam_rootok - разрешения руту обходить доп проверки
- pam_limits - установка лимитов для пользователей

- Ограничение доступа пользователей по дням недели
- добавим строку
  - account required pam_time.so
- в файлы
  - /etc/pam.d/sshd
- в /etc/security/time.conf добавим запрет
  - sshd;*;otus;Al

AppArmor


Yast Console
Reporting and alerting user interface
Linux OS component
Desktop application
Server application
AppArmor
Application profiles
AppArmor module
Reporting and alerting
Linux 2.6 kernel
LMS interface

AppArmor: пример

/etc/apparmor.d/usr.bin.nginx

```
#include <tunables/global>

/usr/sbin/nginx {
  #include <abstractions/base>
  #include <abstractions/lxc/container-base>
  capability dac_override,
  capability dac_read_search,
  capability net_bind_service,
  capability setgid,
  capability setuid,

  /data/www/safe/* r,
  deny /data/www/unsafe/* r,
  /etc/group r,
  /etc/nginx/conf.d/ r,
  /etc/nginx/mime.types r,
  /etc/nginx/nginx.conf r,
  /etc/nsswitch.conf r,
  /etc/passwd r,
  /etc/ssl/openssl.cnf r,
  /run/nginx.pid rw,
  /usr/sbin/nginx mr,
  /var/log/nginx/access.log w,
  /var/log/nginx/error.log w,
}
```

AppArmor: Режимы работы

1. Фиксации/Обучения: нарушения профиля разрешаются и сохраняются в журнале. Полезно для тестирования и разработки новых профилей
2. Предписаний/Ограничений: принуждает следовать политике профиля, при этом также записывает нарушения в журнал.

- aa-complain переводит профиль в режим обучения (complain).
- aa-enforce переводит профиль в режим ограничений (enforce).
- aa-status - посмотреть статус
- подгружаем новый профиль
    - cat /etc/apparmor.d/usr.bin.nginx | sudo apparmor_parser -a

Вопросы?