Linux security

- Управление группами и пользователями
- Firewall
- Введение в Linux security modules
- Анализ исполняемого файла

?????????? ???????? ? ??????????????

Команды групп

Команда	Описание
sudo adduser username	интерактивно создает пользователя с паролем и группой. Поведение настраивается в файлах /etc/adduser.conf Нельзя использовать в скриптах. При помощи encryptfs-utils можно зашифровать домашнюю директорию --encrypt-home
sudo useradd username	создает пользователя без группы и пароля. Поведение настраивается флагами и /etc/login.defs. -m создает домашнюю директорию -d определяет домашнюю директорию -s определяет консоль по умолчанию
groups	Список групп
sudo visudo	Настройка прав групп -f fname определяет дополнительный файл из include директории
sudo usermod -a -G sudo sergey	Добавить пользователя в группу sudo. -a оставляет существующие группы. Без -a пользователь будет исключен из остальных групп
sudo userdel username	Удалить пользователя
last	Последние авторизовавшиеся пользователи

Информация о пользователях, паролях, группах

/etc/passwd /etc/shadow /etc/group /etc/gshadow

Служба nss отвечает за хранение данных в любых хранилищах (вплоть до SQL). /etc/nsswitch.conf

Определение библиотек, установленных у службы nss

```
~$ find /lib/ -name 'libnss_*'
/lib/x86_64-linux-gnu/libnss_compat.so.2
/lib/x86_64-linux-gnu/libnss_files.so.2
```

```
/lib/x86_64-linux-gnu/libnss_hesiod.so.2
/lib/x86_64-linux-gnu/libnss_systemd.so.2
/lib/x86_64-linux-gnu/libnss_dns.so.2
```

Настройка visudo

Элемент	Описание
#	Комментарий
% в начале строки	Работа с группой. %sudo обозначает настройку группы sudo
Структура назначения прав	%sudo ALL=(ALL) ALL обозначает, что группа sudo может выполнять на всех хостах от имени всех пользователей любые действия
NOPASSWD:	без ввода пароля sudo %sudo ALL=(ALL) NOPASSWD: ALL
User_Alias alias_name = username1, username2	Псевдоним для имен пользователей. User_Alias WEBADMINS = sergey, dimitry
Cmnd_Alias alias_name = cmnd1, cmnd2	Псевдоним для команд. Cmnd_Alias SOFTWARE /bin/apt /bin/apt-get Если команда перечисляется без субкоманд, то пользователю даются права на все субкоманды. Иначе, при перечислении субкоманд - только на них. Но при добавлении субкоманд требуется указывать полную строку. Например, если указать /ust/bin/systemctl status, то пользователь будет иметь право вызвать эту команду, но она нефункциональна - требует имя сервиса, а его задать уже нельзя. Можно использовать * /ust/bin/systemctl status * /ust/bin/systemctl * sshd В списке могут быть и файлы, доступные для редактирования. Например /usr/bin/nano /etc/ssh/sshd_config
WEBADMINS ALL=(ALL) SOFTWARE	Пользователи WEBADMINS могут запускать команды SOFTWARE через sudo
Host_Alias MAILSERVERS = smtp	Используется при распространении единого файла sudoers на все машины в сети. Ограничивает хосты, где работает данное правило WEBADMINS WEBSERVERS=(ALL) WEBCOMMANDS
Defaults timestamp_timeout = 0	Таймаут при необходимости ввода пароля sudo. Можно назначить только определенным пользователям Defaults:dmitry timestamp_timeout = 0

Элемент	Описание
@includedir dirname	для логического разбиения одного файла на группы. Подключается любой файл, если он не содержит в названии точку (.) или не заканчивается тильдой (~). Файлы анализируются в лексическом порядке. Например, /etc/sudoers.d/01_first анализируется перед /etc/sudoers.d/10_second. После анализа файлов в каталоге контроль возвращается к файлу, который содержал директиву @includedir.

Получается нужны шаблоны для определения действий в зависимости от роли пользователей. Их может быть немало)

Требования к паролю

```
sudo apt install libpam-pwquality
```

Настройки требований к паролю в файле /etc/security/pwquality.conf

Параметр	Описание
minlen=8	Минимальная длина пароля
minclass=3	Минимальное количество классов символов в пароле

Период обновления пароля

chage -l username

Блокировка пароля при неправильном вводе

Практически, нельзя устанавливать блокировку пароля после 3 неправильных вводов.

/etc/pam.d/login

Firewall

Исходя из общей теории брандмауэров, можно сделать вывод: для enterprise решений каждый уровень модели TCP/IP должен быть защищен. Однако статичные правила работают неэффективно. Должны быть межуровневые взаимодействия, позволяющие передавать например информацию о подозрительной активности на уровне приложений, на уровень IP фильтрации и блокировать данную активность на уровне IP. Т е enterprise решение должно объединять в себе данные всех уровней в одну картину. Кроме этого, желательно объединение аппаратных и программных решений, т е данные о необходимости блокировки передаются между физическими устройствами.

Политики блокировки

Адреса - источники, которые нужно блокировать на внешнем интерфейсе

- Ваш IP адрес, lo
- IP адреса вашей локальной сети
- Частные адреса класса A, B и C
- Мультикастовые класса D и зарезервированные класса E
- Также можно блокировать Carrier-grade NAT сети (100.64.0.0-100.127.255.255) и TEST-NET (192.0.2.0-192.0.2.255)
- Есть стратегия блокировки ненадежных пулов (наверное, где-то есть список)
- Можно добавить блокировку бродкаста, но это редко используется

Возможно ограничение количества пакетов от источника, порты удаленного источника (>1023), блокировка по флагу статуса TCP соединения (флаг должен быть SYN, не ACK).

Нужно контролировать попытки сканирования портов (могут быть разные источники, но скоординированное сканирование).

netfilter - встроенный firewall. Консольные утилиты для управления netfilter:

- iptables - для v4 и v6 различные
- ebtables - mac briging правила
- arptables - правил трансляции arp

ufw - фронтенд для iptables

nftables - иная реализация

firewalld - реализация в RedHat

Какой firewall сейчас активен

Могут быть установлен и iptables, и nftables. Если вывод следующей команды не пустой, значит убедимся, что модуль nft активен:

```
sudo nft list ruleset
```

Проверяем модуль ядра:

```
lsmod | grep nf_tables
```

Если вывод типа

```
nf_tables             376832  114 nft_compat,nft_chain_nat
```

значит nftables загружен.

Модуль iptables еще присутствует в ядре, но как говорят - для вида.

```
sudo lsmod | grep ip_tables
ip_tables              32768  0
x_tables               65536  8
xt_conntrack,nft_compat,xt_tcpudp,xt_addrtype,xt_nat,xt_set,ip_tables,xt_MASQUERADE
```

Есть промежуточный пакет iptables-nft, который транслирует правила из iptables, Но работает именно nft. Однако редактировать nft таблицы с комментариями

```
# Warning: table ip filter is managed by iptables-nft, do not touch!
```

не стоит. Проблема вот в чем: если не удален/отключен iptables-nft, то после каждого использования команды iptables данные таблицы будут перезаписаны. Поэтому либо не использовать эту команду, включить сервис nft и все, либо создать другие таблицы.

В Alt Linux, nft по умолчанию не установлен.

Iptables

Существующие сессии ssh не блокируются.

Состоит из 4 таблиц:

- Filter table Базовая защита, обычно используется
- NAT table
- Mangle table Изменение сетевых пакетов при их прохождении через брандмауэр
- Security table Используется в системах с установленным SELinux

Каждая таблица состоит из цепочек правил. Filter table состоит из цепочек INPUT, FORWARD, OUTPUT

Просмотр существующих таблиц

```
sudo iptables -L <имя цепочки или ничего> --line-numbers -v
```

Сохранение правил после перезагрузки сервера

```
sudo apt install iptables-persistent
```

Затем сохраняем

```
sudo netfilter-persistent save
```

После этого правила сохраняются в /etc/iptables*

Структура команды

```
iptables <option> <chain> <matching criteria> <target>
```

Пример:

```
sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
```

Аргумент	Описание
Option параметры	
-A, -I	Добавить в конец или в определенное место цепочки. -A INPUT -I INPUT 1
-R	заменить цепочку
-P	Задает дефолтную политику для цепочки. `iptables -P FORWARD DROP`
-D цепочка номер	Удалить запись `sudo iptables -D DOCKER-USER 1`
-N	Создать пользовательскую цепочку правил
-F, -X	Удалить цепочки / пользовательские цепочки
-E	Переименовать пользовательскую цепочку
Matching criteria параметры	

Аргумент	Описание
-m	Модуль.
--ctstate	Состояние пакета. Перечисляется через запятую без пробелов.
-p	Тип соединения, tcp/udp/icmp
--dport	порт назначения --dport ssh с опцией -m можно перечислять через запятую без пробелов `-m multiport --dport 2049,1080,3128` -m должна идти строго после -p ... через : диапазон портов, без multiport
--port	Определяет и входящий, и исходящий порты
-i -o	Интерфейс (входящий/исходящий) -i lo
-s -d	IP источника / приемника
	Кроме этого, еще есть фильтры по: • Текущее состояние соединения • Список портов (поддерживаемых многопортовым модулем) • MAC-адрес источника аппаратного Ethernet или физического устройства • Тип адреса, тип пакета канального уровня или диапазон IP-адресов • Различные части пакетов IPSec или политика IPSec • Тип протокола ICMP • Длина пакета • Время получения пакета • Каждый n-й пакет или случайные пакеты • Идентификатор пользователя, группы, процесса или группы процессов отправителя пакета • Поле типа обслуживания (TOS) заголовка IP (возможно, заданное в таблице управления) • Раздел TTL заголовка IP • Поле iptables mark (устанавливается в таблице управления) • Соответствие пакетов с ограниченной скоростью
Target параметры	

Аргумент	Описание
-j	Действие при соблюдении условия ACCEPT разрешить пакет DROP тихо откинуть пакет REJECT вернуть ответ о блокировании запрашивающей стороне RETURN вернуть в родительскую цепочку. Используется в случае родительских / дочерних цепочек. Еще есть очередь QUEUE Может быть переход к пользовательской цепочке, с возвратом обратно
-g	переход к другой цепочке без возврата обратно
-c	Инициализация счетчиков

В OUTPUT цепочке есть возможность фильтровать по пользователю, группе, процессу, заголовкам безопасности - полный цикл управления.

Также возможна настройка блокировки по времени. Также можно фильтровать каждый N пакет например для логирования. И это все на уровне ядра!

Разрешение доступа для ssh

```
sudo iptables -A INPUT -p tcp --dport ssh -j ACCEPT
```

Здесь для определения порта по названию использовался файл /etc/services То есть возможно прописать свое название службы в данный файл и использовать его в правилах.

```
cat /etc/services | grep http
# Updated from https://www.iana.org/assignments/service-names-port-numbers/service-names-port-
numbers.xhtml .
http            80/tcp          www             # WorldWideWeb HTTP
https           443/tcp                         # http protocol over TLS/SSL
http-alt        8080/tcp        webcache        # WWW caching service
http-serg       8081/tcp                        # test server
```

Затем правилом разрешаем доступ по имени службы. Однако не понятно, почему нужно определять тип соединения. Похоже, что из файла берется только номер порта.

```
sudo iptables -I INPUT 1 -p tcp --dport http-serg -j ACCEPT
```

Разрешение доступа к lo

```
sudo iptables -I INPUT 1 -i lo -j ACCEPT
```

Логирование

Лог всех пакетов (и входящие, и исходящие), на порту 8090.

```
sudo iptables -A INPUT -p tcp --dport 8090 -j LOG --log-prefix "IPTABLES-8090-IN: "
sudo iptables -A INPUT -p udp --dport 8090 -j LOG --log-prefix "IPTABLES-8090-IN: "
sudo iptables -A OUTPUT -p tcp --sport 8090 -j LOG --log-prefix "IPTABLES-8090-OUT: "
sudo iptables -A OUTPUT -p udp --sport 8090 -j LOG --log-prefix "IPTABLES-8090-OUT: "
```

Лог только новых соединений:

```
sudo iptables -A INPUT -p tcp --dport 8090 -m conntrack --ctstate NEW -j LOG --log-prefix
"NEW-8090-CONN: "
```

Просмотр лога в реальном времени:

```
sudo tail -f /var/log/syslog | grep "IPTABLES-8090"
# или
sudo tail -f /var/log/kern.log | grep "IPTABLES-8090"
```

Блокировка по географическому признаку

Работает на уровне ядра. Проверка установлен ли модуль.

```
lsmod | grep xt_geoip

sudo apt update
sudo apt install xtables-addons-common xtables-addons-source build-essential dkms
sudo modprobe xt_geoip
```

Загрузка геобазы.

Вариант 1 (не доделал)

Страница создания аккаунта Из России пока что нельзя создать новую учетку. Надеюсь скоро все изменится. Создали учетку, Управление аккаунтом - Управление ключами лицензирования. Создаем новый ключ, скачиваем конфиг на всякий случай.

Создаем директорию и скачиваем текущую базу

```
mkdir -p /usr/share/xt_geoip
cd /usr/share/xt_geoip

wget "https://download.maxmind.com/app/geoip_download?edition_id=GeoLite2-Country-CSV&license_key=ВАШ_КЛЮЧ&suffix=zip" -O GeoLite2-Country-CSV.zip

unzip GeoLite2-Country-CSV.zip -d geolite
```

Теперь нужно преобразовать базу в требуемый формат. Но перед этим в ubuntu возникло несколько проблем. Одна с размещением скрипта.

```
dpkg -L xtables-addons-common | grep xt_geoip
/usr/bin/xt_geoip_query
/usr/lib/x86_64-linux-gnu/xtables/libxt_geoip.so
/usr/libexec/xtables-addons/xt_geoip_build
/usr/libexec/xtables-addons/xt_geoip_build_maxmind
/usr/libexec/xtables-addons/xt_geoip_dl
/usr/libexec/xtables-addons/xt_geoip_dl_maxmind
/usr/share/man/man1/xt_geoip_build.1.gz
/usr/share/man/man1/xt_geoip_build_maxmind.1.gz
/usr/share/man/man1/xt_geoip_dl.1.gz
/usr/share/man/man1/xt_geoip_dl_maxmind.1.gz
/usr/share/man/man1/xt_geoip_query.1.gz
```

Далее нужно было преобразовать в формат при помощи python скрипта, поскольку Ubuntu изменил формат для базы. Или -

Вариант 2

На [странице сервиса] находим ссылку на текущую версию, скачиваем

```
wget https://download.db-ip.com/free/dbip-country-lite-2025-08.csv.gz -O dbip-country-lite.csv.gz
gunzip dbip-country-lite.csv.gz
```

Преобразовываем базу данных (поиск размещения скрипта в Варианте 1)

```
sudo /usr/libexec/xtables-addons/xt_geoip_build -D /usr/share/xt_geoip dbip-country-lite.csv
```

Пример разрешения / блокировки по геопризнаку

```
sudo iptables -A INPUT -m geoip --src-cc RU -j ACCEPT
sudo iptables -A INPUT -m geoip --src-cc CN,IR,KP -j DROP
sudo iptables -A INPUT -p tcp --dport https -m geoip --src-cc RU -j ACCEPT
```

Примеры настройки

Пример настройки iptables

Политика:

- Разрешить доступ к службам http, https на докере для всех
- Разрешить все исходящие соединения
- Разрешить доступ для ip1
- Docker должен функционировать корректно
- Все остальное запретить

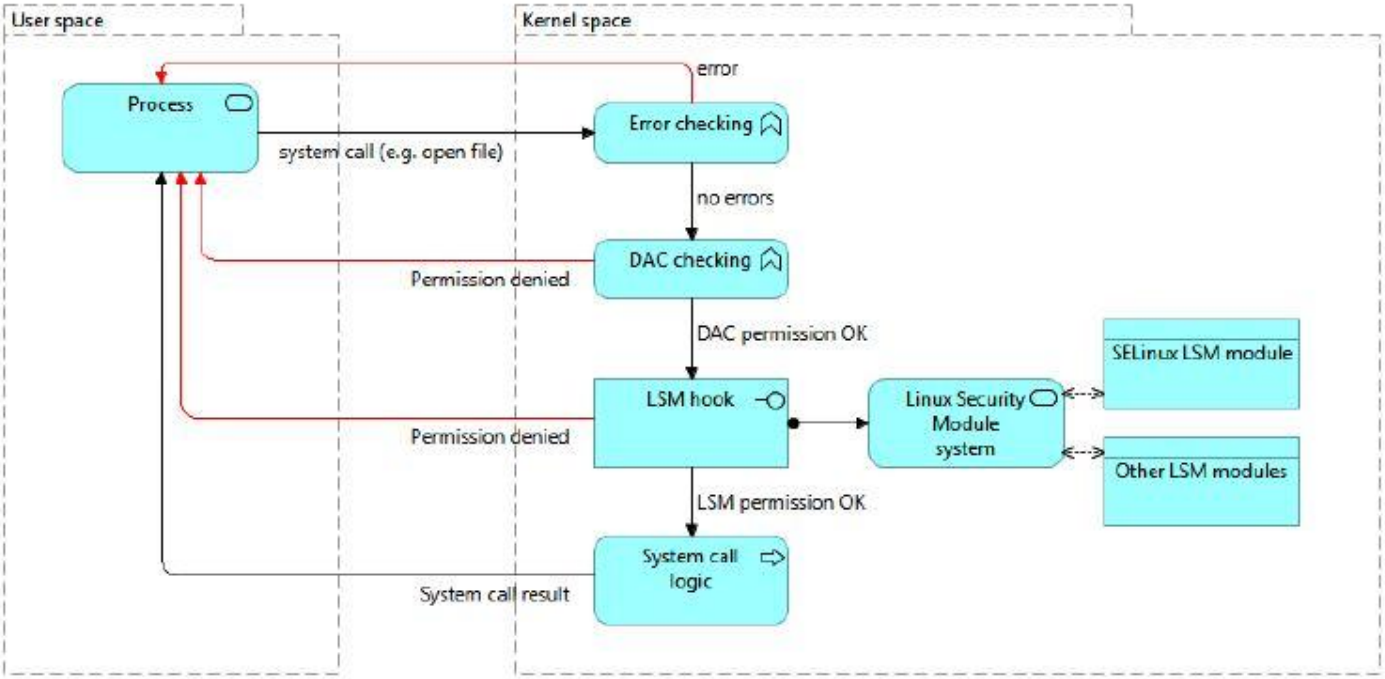
```
sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A INPUT -i lo -j ACCEPT
sudo iptables -A INPUT -s 1.1.1.1 -j ACCEPT
sudo iptables -I INPUT 1 -p icmp --icmp-type echo-request -j ACCEPT

sudo iptables -A INPUT -i docker0 -j ACCEPT
sudo iptables -P INPUT DROP

sudo iptables -A DOCKER-USER -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A DOCKER-USER -s 1.1.1.1 -p tcp --dport 5432 -j ACCEPT
sudo iptables -A DOCKER-USER -p tcp --dport 5432 -j DROP
```

???????? ? Linux security modules

Система безопасности основана на дополнительном хуке (LSM hook) при вызове системных (ядерных) функций. Система модульная, и SELinux - один из модулей, который предоставляет функции безопасности. DAC система находится ниже. Стек вызовов функции:



LSM hook не предоставляет функций безопасности - это просто хук. Далее идет вызов зарегистрированного модуля(ей) LSM Framework. В LSM Framework можно зарегистрировать один эксклюзивный модуль и неэксклюзивные.

Модуль	Особенности	Где используется	Тип
SELinux	Метки безопасности (label-based), очень гибкая и строгая политика	RHEL, Fedora, CentOS, Android	Экск.
AppArmor	Основан на путях (path-based), проще в настройке	Ubuntu, Debian, openSUSE	Экск.
Smack (Simplified Mandatory Access Control Kernel)	Проще, чем SELinux, тоже использует метки, но менее гибкий	Встроенные системы (Tizen, некоторые IoT-устройства)	Экск.
TOMOYO Linux	Основной акцент на управление доступом на основе процессов и истории их действий (обучение)	Япония, встраиваемые системы	Неэкск.

Модуль	Особенности	Где используется	Тип
Yama	Маленький LSM для ограничения `ptrace` (трассировки процессов)	Включён в ядро по умолчанию (Ubuntu, Debian, Arch)	Неэкск.
Landlock	Новый LSM (с 2021 г., Linux 5.13+), даёт приложениям возможность ограничивать **себя** (sandboxing)	Современные Debian/Ubuntu, используется для sandbox-браузеров и контейнеров	Неэкск.
LoadPin	Гарантирует загрузку модулей ядра только из доверенной FS	Chromebook, сервера	Неэкск.
Integrity (IMA/EVM)	Контроль целостности файлов (подписи, хэши)	Корпоративные Linux-системы	Неэкск.
Lockdown	Ограничивает root-доступ к ядру (часть upstream ядра с 5.4)	Включён в Debian, Ubuntu, Fedora	Неэкск.
Capability	Базовый модуль, модель минимально необходимых привилегий. Всегда включён.		Неэкск.
bpf	Контроль безопасности eBPF-программ		Неэкск.
ipe	Политики целостности (разрешение/запрет запуска бинарников)		Неэкск.

Список используемых LSM модулей

```
cat /sys/kernel/security/lsm
lockdown,capability,landlock,yama,apparmor,tomoyo,bpf,ipe,ima,evm
```

Естественно модули для непересекающихся задач не конфликтуют.

?????? ????????????? ?????

ldd ./program

```
$ ldd /bin/ls
        linux-vdso.so.1 (0x00007fff2b9fe000)
        libselinux.so.1 => /lib/x86_64-linux-gnu/libselinux.so.1 (0x00007f3cdd5b3000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f3cdd1b0000)
        /lib64/ld-linux-x86-64.so.2 (0x00007f3cdd7e0000)
```

Показывает список динамических библиотек, которые будут загружены при запуске программы.
ldd фактически выполняет часть программы через загрузчик, нельзя запускать ldd на подозрительных бинарниках.

readelf -d
Читает динамическую таблицу ELF-файла и показывает список библиотек, от которых он зависит.

```
$ readelf -d /usr/bin/mate-calc | grep NEEDED
 0x0000000000000001 (NEEDED)             Совм. исп. библиотека: [libgtk-3.so.0]
 0x0000000000000001 (NEEDED)             Совм. исп. библиотека: [libgdk-3.so.0]
 0x0000000000000001 (NEEDED)             Совм. исп. библиотека: [libpango-1.0.so.0]
 0x0000000000000001 (NEEDED)             Совм. исп. библиотека: [libatk-1.0.so.0]
```

objdump -p

Аналогично readelf, но может показать больше контекста.

```
objdump -p /usr/bin/mate-calc | grep
NEEDED
  NEEDED               libgtk-3.so.0
  NEEDED               libgdk-3.so.0
  NEEDED               libpango-1.0.so.0
  NEEDED               libatk-1.0.so.0
```

В рантайме — strace

Если вы хотите узнать, какие библиотеки реально подгружаются при запуске (в том числе через dlopen() во время работы):

```
strace -e open ./program 2>&1 | grep '\.so'
```

Если программа статически слинкована (т.е. все зависимости включены внутрь ELF), то ни ldd, ни readelf не покажут зависимостей.

Проверить это можно так:

```
file ./program
```

Пример:

```
# динамическая линковка
file /usr/bin/mate-
calc

/usr/bin/mate-calc: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically
linked, interpreter /lib64/ld-linux-x86-64.so.2,
BuildID[sha1]=58d991a9b15b10d59d235c79b2132fde101cf1fc, for GNU/Linux 3.2.0, stripped

# статическая линковка
... statically linked ...
```

Пример анализа зависимостей для /usr/bin/grep.

1. Проверим, какой это тип ELF-файла

```
file /usr/bin/grep
/usr/bin/grep: ELF 64-bit LSB executable, x86-64, dynamically linked, interpreter \
/lib64/ld-linux-x86-64.so.2, for GNU/Linux 3.2.0, stripped
```

Значит, это динамически слинкованный ELF, и у него есть внешние зависимости.

2. Смотрим список зависимостей через ldd

```
ldd /usr/bin/grep
        linux-vdso.so.1 (0x00007fff43dfe000)
        libpcre2-8.so.0 => /lib/x86_64-linux-gnu/libpcre2-8.so.0 (0x00007f3f5a8b0000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f3f5a4a0000)
        /lib64/ld-linux-x86-64.so.2 (0x00007f3f5aaf0000)
```

Интерпретация:

- libpcre2-8.so.0 — библиотека регулярных выражений PCRE2.

- libc.so.6 — стандартная библиотека C (glibc).
- ld-linux-x86-64.so.2 — динамический загрузчик ELF.

3. Альтернатива: readelf -d

```
readelf -d /usr/bin/grep | grep NEEDED
 0x0000000000000001 (NEEDED)             Shared library: [libpcre2-8.so.0]
 0x0000000000000001 (NEEDED)             Shared library: [libc.so.6]
```

Это подтверждает те же зависимости.

4. Проверим, какие библиотеки реально подгружаются при запуске (в том числе через dlopen())

```
strace -e openat /usr/bin/grep "test" /etc/passwd 2>&1 | grep '\.so'
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libpcre2-8.so.0", O_RDONLY|O_CLOEXEC) = 3
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
```

5. Проверим динамический загрузчик (интерпретатор ELF)

```
readelf -l /usr/bin/grep | grep 'interpreter'
[Requesting program interpreter: /lib64/ld-linux-x86-64.so.2]
```

Это та программа, которая первым делом загружает бинарь и его библиотеки.

Итог:

Тип проверки	Инструмент	Что показывает
Тип ELF	`file`	Статически или динамически слинкован
Динамические зависимости	`ldd`, `readelf -d`, `objdump -p`	Список библиотек
Реальные загрузки во время работы	`strace`	Библиотеки, подгружаемые во время исполнения
ELF-интерпретатор	`readelf -l`	Путь к динамическому загрузчику

Можно проверить, какие именно функции приложение импортирует из библиотек.

Теперь углубимся на уровень **символов (функций)**, которые исполняемый файл импортирует из библиотек — например, какие функции `grep` реально использует из `libc`, `libpcre2` и других.

1. Проверим импортированные символы (динамически загружаемые функции)

Для этого можно использовать:

```
readelf -Ws /usr/bin/grep | grep ' UND '
```

или короче:

```
readelf -Ws /usr/bin/grep | grep UND | less
```

- `-W` — показывает полные строки.
- `-s` — выводит таблицу символов.
- `UND` означает *undefined symbol* — то есть функция **определена в внешней библиотеке** (т.е. импортируется).

Пример вывода (сокращённо)

```
Num:    Value  Size Type    Bind   Vis      Ndx Name
  45: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND strcmp@GLIBC_2.2.5
  72: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND malloc@GLIBC_2.2.5
  89: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND regexec@LIBPCRE2_8
 114: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND free@GLIBC_2.2.5
 132: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND fprintf@GLIBC_2.2.5
```

Расшифровка:

- `strcmp`, `malloc`, `free`, `fprintf` → из `libc`
- `regexec` → из `libpcre2-8.so.0`

---

2. Посчитать, сколько внешних символов программа импортирует

Можно примерно оценить так:

```
readelf -Ws /usr/bin/grep | grep ' UND ' | wc -l
```

Например, может вывести:

```
125
```

значит, `grep` использует около 125 внешних функций из библиотек.

---

3. Уточнить, из какой именно библиотеки берётся символ

Команда `ldd` показывает пути библиотек, а `readelf` — имена функций, но если ты хочешь **увидеть, какой именно `.so` экспортирует нужную функцию**, можно воспользоваться `objdump` или `nm`.

Пример: узнать, где определена функция `regexec`

```
for lib in /lib/x86_64-linux-gnu/*.so*; do
    nm -D $lib 2>/dev/null | grep ' regexec$' && echo "→ $lib"
done
```

Результат:

```
000000000001f230 T regexec
→ /lib/x86_64-linux-gnu/libpcre2-8.so.0
```

---

4. Альтернатива: `objdump -T`

Если хочешь посмотреть **все экспортируемые символы библиотеки**:

```
objdump -T /lib/x86_64-linux-gnu/libc.so.6 | grep printf
```

Это покажет строку вроде:

```
0000000000054350 g    DF .text  00000000000001a5  GLIBC_2.2.5 printf
```

Резюме

Задача	Команда	Что показывает
Список импортированных функций	`readelf -Ws program	grep UND`
Кол-во импортов	`readelf -Ws program	grep UND
Где определён символ	`nm -D /lib/...	grep <имя>`
Все экспортируемые символы из .so	`objdump -T libname.so`	Список функций, предоставляемых библиотекой
