

ADMINISTRACJA SYSTEMEM LINUX II

Marcin Kujawski

ZASADY

- Wstań od komputera w trakcie przerwy
- Komunikacja na Zoom-ie
- Napisz swoje imię
- Jeśli skończyłeś zadanie – użyj ikony reakcji
- Materiały ze szkolenia na GITHUB

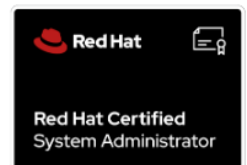
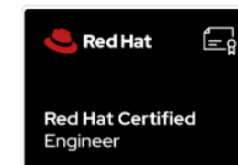
SAY HI!



O MNIE:



MARCIN KUJAŃSKI
TRENER



PRZERWY

DAY 1-5

10:30 – 10:45

12:30 – 13:15

14:45 – 15:00



AGENDA

Day 1:

- Strumienie i przekierowania I/O
- Wyrażenia regularne
- Narzędzia archiwizujące i kompresujące
- Monitoring i Performance
- Systemd + systemctl
- Planowanie zadań i zadań cyklicznych

Day 2:

- Dowiązania twarde i miękkie
- Filesystemy na Linuxie
- Tworzenie partycji i filesystemów
- Montowanie filesystemów
- LVM
- Szyfrowanie partycji
- RAID (opcjonalnie)

AGENDA

Day 3:

- Użytkownicy i grupy
- Rozszerzone prawa do plików
- ACL
- PAM
- LDAP i Kerberos

Day 4:

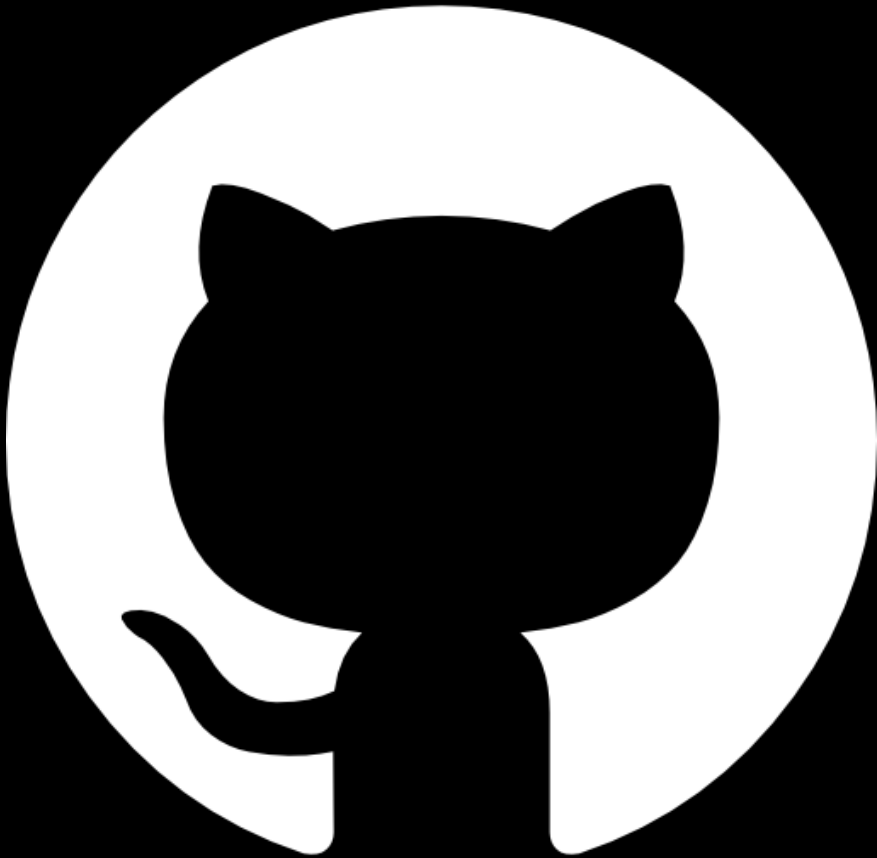
- Procesy i sygnały
- Konfiguracja jądra i moduły
- Journal
- Logi systemowe oraz Logrotate
- Konfiguracja Sieci
- Firewall i IPtables

AGENDA

Day 5:

- NFS
- NFS + Kerberos
- Samba
- SELinux
- LXC (optional)

GITHUB



Repo:

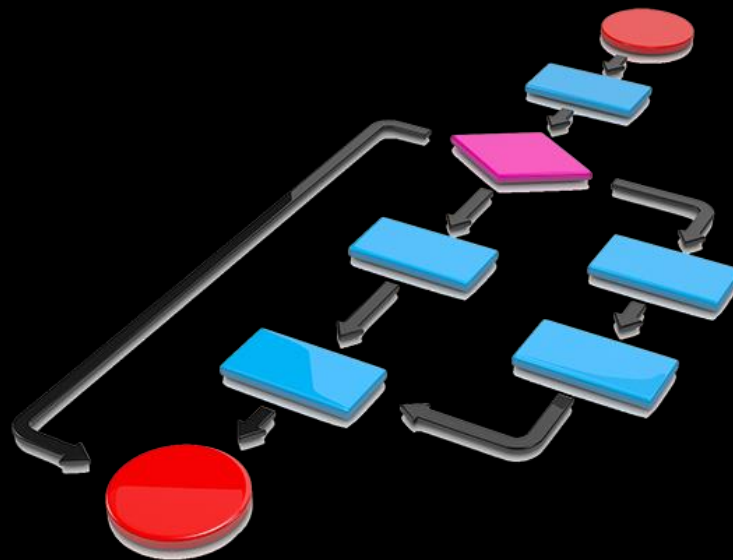
https://github.com/mariano-italiano/lin2_Apr.git

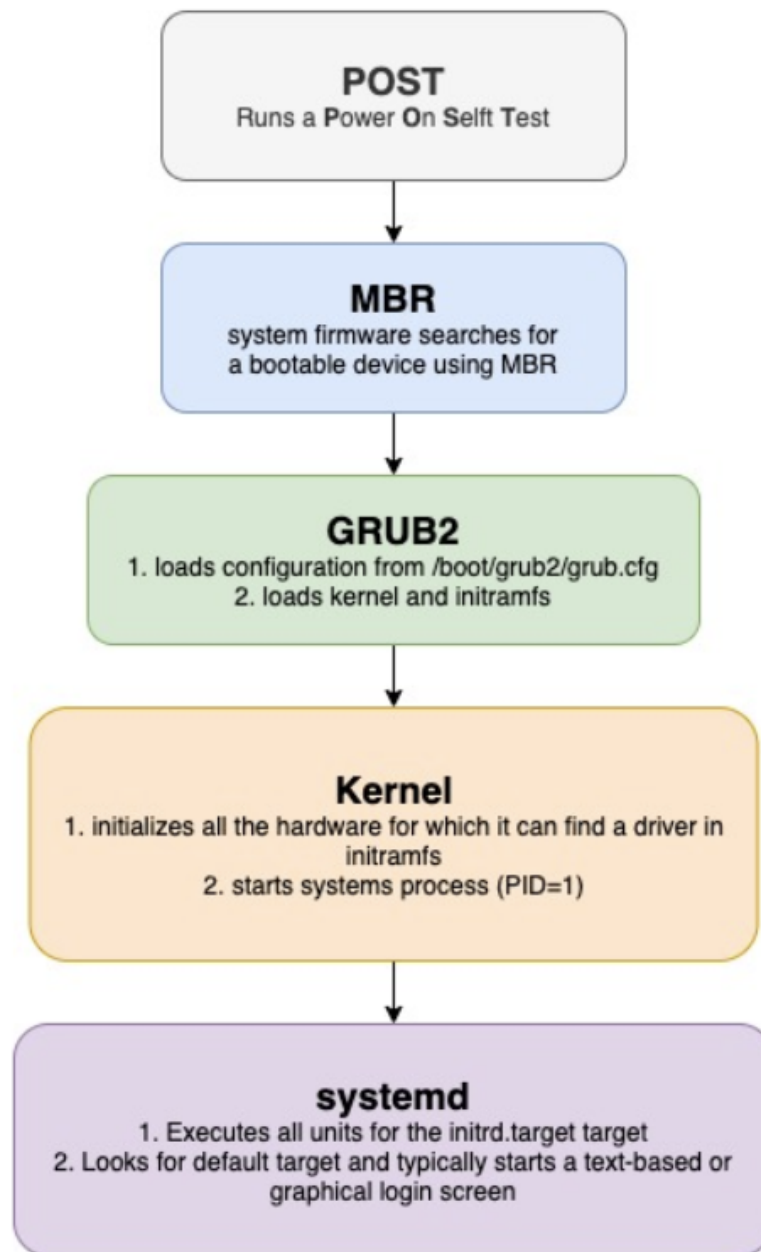
Klonowanie:

```
git clone <repo-url>
```

Cała historia + skrypty

PROCES BOOTOWANIA - LINUX





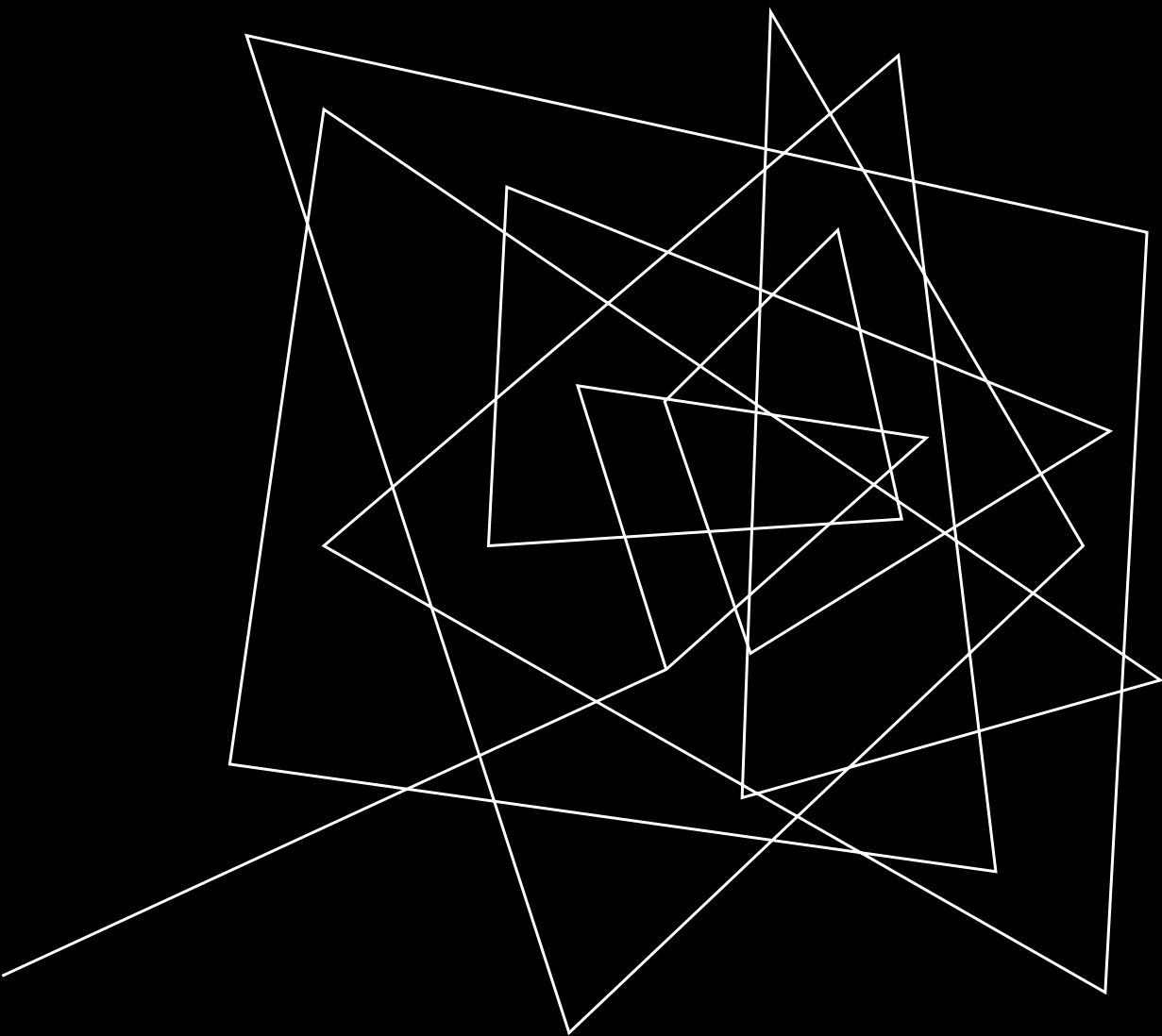
The machine is powered on. The system firmware, either modern UEFI or older BIOS, runs a Power On Self Test (POST) and starts to initialize some of the hardware.

The system firmware searches for a bootable device, either configured in the UEFI boot firmware or by searching for a Master Boot Record (MBR) on all disks, in the order configured in the BIOS.

The system firmware reads a boot loader from disk and then passes control of the system to the boot loader. On a CentOS/RHEL 8 system, the boot loader is the GRand Unified Bootloader version 2 (GRUB2). GRUB2 loads its configuration from /boot/grub2/grub.cfg file and displays a menu where you can select which kernel to boot.

After you select a kernel, or the timeout expires, the boot loader loads the kernel and initramfs from disk and places them in memory. An initramfs is an archive containing the kernel modules for all the hardware required at boot, initialization scripts, and more. On Red Hat Enterprise Linux 8, the initramfs contains an entire usable system by itself.

The systemd instance from the initramfs executes all units for the initrd.target target. This includes mounting the root file system on disk on to the /sysroot directory. Configured using /etc/fstab. The kernel switches (pivots) the root file system from initramfs to the root file system in /sysroot. systemd then re-executes itself using the copy of systemd installed on the disk. Systemd looks for a default target, either passed in from the kernel command line or configured on the system, then starts (and stops) units to comply with the configuration for that target, solving dependencies between units automatically



STRUMIENIE I PRZEKIEROWANIA I/O

(>> > < <<)

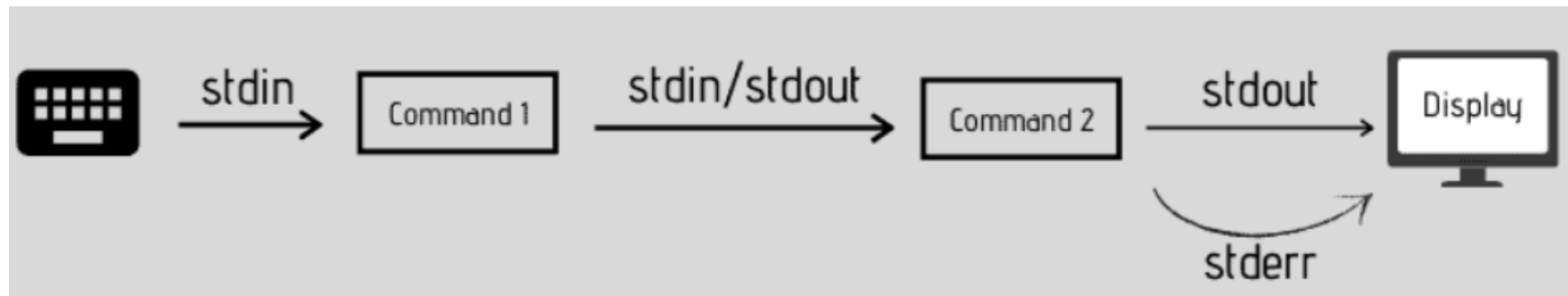
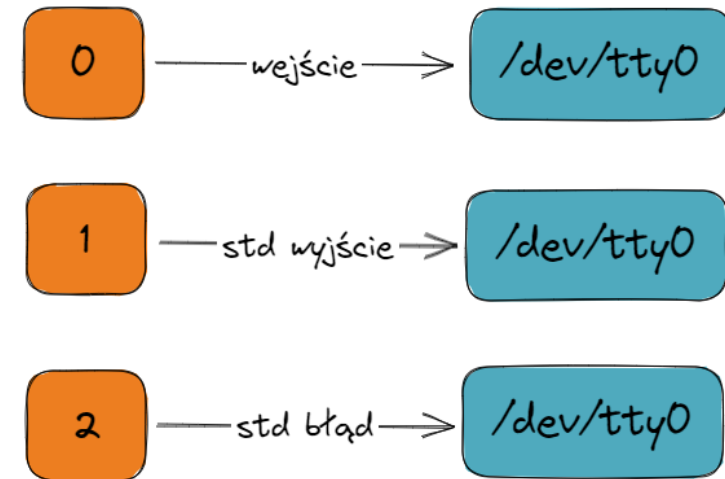
STRUMIENIE I PRZEKIEROWANIA I/O

Znak	Opis
>	Przekierowanie wyjścia do pliku (nadpisanie) \$ ls -lR > dirtree.txt
>>	Przekierowanie wyjścia do pliku, dopisanie do końca pliku (dopisanie) \$ echo "ostatnia linijka" > dirtree.txt
<	Przekierowanie wejścia na plik \$ sort < dirtree.txt

DESKRYPTORY PLIKÓW

Kiedy uruchamiamy polecenie, są trzy deskryptory plików, o których trzeba wspomnieć.

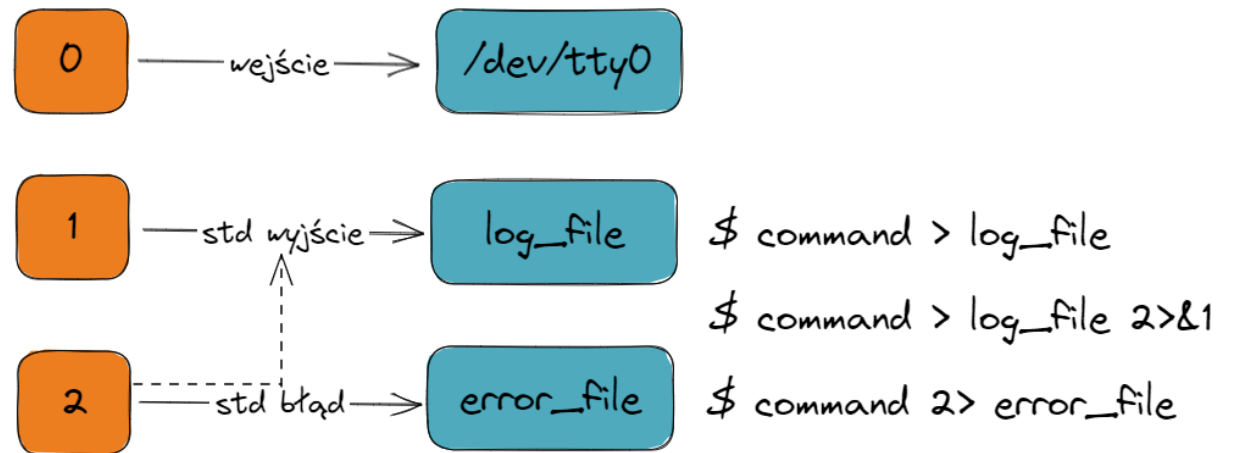
- zero to wejście danych,
- jeden to standardowe wyjście,
- dwa to standardowy błąd.



DESKRYPTORY PLIKÓW

A co, jeśli chcielibyśmy przekierować standardowy błąd i standardowe wyjście do tego samego pliku?

```
ls -la > wyjscie.std  
ls -la >> wyjscie.dopisanie  
ls -la >& wyjscie.w.tle  
ls -la 2> wyjscie.z.bledem
```





WYRAŻENIA REGULARNE CZYLI /(REG)EX/*Y

REGEXY

Wyrażenie regularne jest to ciąg znaków określający wzór wyszukiwania.

Zwykle wyrażenia te używane są z narzędziem takim jak grep do wyciągania istotnych informacji z pliku.

Regex'y są stosunkowo złożone z bardzo wieloma odmianami, są niemalże językiem programowania samym w sobie.

SILNIKI REGEX

Silnik regex lub silnik wyrażeń regularnych to kawałek oprogramowania, który rozumie ciągi wyrażeń regularnych i tłumaczy je, aby znaleźć dopasowany tekst. Istnieje wiele silników regex, na przykład silniki, które są dostarczane z językami programowania takimi jak Java, Perl czy Python. W Linuksie silnikami wyrażeń regularnych są narzędzia linuksowe: sed i AWK.

Istnieją dwa typy silników regex w Linuksie:

- Podstawowe Wyrażenie Regularne (BRE)
- Rozszerzone Wyrażenie Regularne (ERE)

Sed i AWK rozumieją oba silniki. Polecenie grep również rozumie ERE, ale musimy użyć opcji -E, co jest równoważne użyciu egrep.

Znak	Wyjaśnienie
.	będzie pasował do każdego pojedynczego znaku
[]	używany do określenia zakresu znaków
[^]	negacja - dopasuje wszystko, co nie jest wewnątrz nawiasów
*	dopasuje zero lub więcej poprzednich elementów
+	dopasuje jeden lub więcej z poprzednich elementów
?	dopasuje zero lub jeden z poprzednich elementów
{n}	dopasuje "n" numerów poprzednich elementów
{n,}	będzie pasować do "n" liczby lub więcej poprzednich elementów
{n,m}	będzie pasować do "n" i "m" liczby elementów
{,m}	będzie pasować do mniejszej lub równej "m" liczby elementów
\	jest znakiem ucieczki, używanym w przypadku konieczności włączenia jednego z metaznaków do wyszukiwania

TWORZENIE ARCHIWUM I KOMPRESJA

Kompresja plików pozwala zaoszczędzić miejsce na dysku i/lub czas transmisji sieciowej. Zawsze zwiększamy efektywność kompresji kosztem dłuższego czasu jej trwania.

Domyślnym programem do zarządzania archiwami dostępny na Linux'ach jest TAR, ale istnieją również inne narzędzia jak: gzip, bzip2, zip, xz, cpio etc.

TWORZENIE ARCHIWUM I KOMPRESJA

- **TAR:** zaprojektowany po to, aby tworzyć jedno archiwum z wielu plików lub katalogów oraz po to aby nimi manipulować. Archiwum może być zwykły plik lub urządzenie
- **CPIO:** odczytuje i zapisuje pliki w formacie binarnym lub ASCII
- **GZIP:** wykorzystuje kodowanie Lempel-Ziv (LZ77) i tworzy pliki .gz
- **BZIP2:** używa algorytmu kompresji tekstu z sortowaniem blokowym Burrowsa-Wheelera i kodowanie Huffmana, produkuje pliki .bz2
- **XZ:** produkuje pliki .xz. a także obsługuje starszy format .lzma, jedna z efektywniejszych kompresji

MONITORING ORAZ PERFORMANCE



TUNING TOOLS

Sieć:

- Nmap
- Ethtool
- Ip
- Netstat
- Iptraf
- Tcpdump
- Wireshark

OS:

- Top
- Sar
- /proc & /sys
- Log files

Performance:

- Iostat
- Vmstat
- XYstat
- Sar
- Stress-ng

TUNING PROFILES - TUNED

Red Hat (oraz linuxy z rodziny CentOS) posiadają wbudowany w OS performance tuning który nazywa się tuned.

Demon tuned jest potężnym procesem który dynamicznie i w pełni automatycznie przeprowadza tuning systemu. Performance serwera Linux bazuje na informacjach, jakie zbierane są przez systemowy monitoring oraz komponenty, które składają się na cały system operacyjny aby dostarczyć end-userowi wymaganą charakterystykę performance'ową jaką sobie zażyczy lub sam zdefiniuje.



SYSTEMD

CO TO JEST SYSTEMD?

Systemd jest podstawowym elementem budulcowym systemu Linux.

Systemd zapewnia:

- agresywną paralelizację usług
- uruchamianie demonów na żądanie
- śledzi procesy używając grup kontrolnych
- utrzymuje punkty montowania i automontowania
- implementuje rozbudowaną logikę kontroli usług opartą na zależnościach transakcyjnych

ZADANIA SYSTEM (D)AEMONA

- Jest menadżerem systemu i usług
- Zastępuje rozwiązania takie jak: SysVinit, upstart
 - Systemd wspiera skrypty inicjujące SysV i LSB i działa jako zamiennik sysvinit.
- Podstawową jednostką organizacyjną jest "unit,,
- W unitach można zdefiniować zależności między nimi

SYSV VS SYSTEMD

SysV Command	Systemd Command	Description
halt	systemctl halt	Halts the system.
poweroff	systemctl poweroff	Powers off the system.
reboot	systemctl reboot	Restarts the system.
pm-suspend	systemctl suspend	Suspends the system.
pm-hibernate	systemctl hibernate	Hibernates the system.
pm-suspend-hybrid	systemctl hybrid-sleep	Hibernates and suspends the system.

SYSV VS SYSTEMD

Runlevel	Target Unit	Target Unit Description
0	runlevel0.target / poweroff.target	Wyłączenie systemu
1	runlevel1.target / rescue.target	Znany też jako Single User, używany głównie do troubleshootowania systemu i jego naprawy oraz do innych zadań administratorskich .
2	runlevel2.target / multi-user.target	Target użytkownika. Defaultowo identyczny jak target 3.
3	runlevel3.target / multi-user.target	Target obsługujący wielu użytkowników, jednak bez interfejsu graficznego. Wiele użytkowników może logować się na lokalną konsolę lub terminal, obsługuje też zdalny dostęp przez sieć. Domyślny target dla 99% serwerów.
4	runlevel4.target / multi-user.target	Target użytkownika. Defaultowo identyczny jak target 3.
5	runlevel5.target / graphical.target	Target z obsługą grafiki oraz wielu użytkowników. Wiele użytkowników może logować się na lokalną konsolę lub terminal, obsługuje też zdalny dostęp przez sieć.
6	runlevel6.target / reboot.target	Powoduje restart systemu.

SYSTEMD SERVICE FILE

Dyrektywy sekcji [Unit]:

- **Zależności:**
 - **Requires** – zależność bezpośrednia. Podczas aktywowania jednostki systemd przystąpi również do aktywowania tej ostatniej. Jeżeli aktywacja ta się nie powiedzie, systemd zdezaktywuje też jednostkę wyjściową.
 - **Wants** – zależność jedynie na potrzeby aktywacji. Po aktywowaniu danej jednostki systemd przystępuje do aktywowania jej zależności Wants, ale nie sprawdza już, czy na pewno udało się je aktywować.
 - **Requisite** – jednostki, które muszą być już aktywne. Przed aktywowaniem danej jednostki z zależnościami Requisite systemd najpierw sprawdza status tych zależności. Jeżeli nie są one jeszcze aktywne, operacja aktywacji jednostki z takimi zależnościami zakończy się niepowodzeniem.
 - **Conflicts** (konflikt) – zależności negatywne. Podczas aktywowania jednostki z zależnością Conflicts systemd automatycznie dezaktywuje taką zależność. Jednoczesna aktywacja dwóch jednostek tworzących względem siebie konflikty zakończy się niepowodzeniem.
- **Kolejność:**
 - **Before** (przed) – aktualna jednostka zostanie aktywowana przed podanymi dalej jednostkami.
 - **After** (po) – aktualna jednostka aktywowana jest dopiero po podanych jednostkach.

SYSTEMD SERVICE FILE

Dyrektywy sekcji [Service]:

- **ExecStart** - określa pełną ścieżkę i argumenty polecenia, które ma być wykonane w celu uruchomienia procesu. Może to być określone tylko raz (z wyjątkiem usług "oneshot"). Jeśli ścieżka do polecenia jest poprzedzona znakiem myślnika "-", niezerowe statusy wyjścia będą akceptowane bez oznaczania aktywacji jednostki jako nieudanej.
- **ExecStop** - wskazuje polecenie potrzebne do zatrzymania usługi. Jeśli nie zostanie podane, proces zostanie zabity natychmiast po zatrzymaniu usługi.
- **Restart** - wskazuje to okoliczności, w których systemd będzie próbował automatycznie zrestartować usługę. Może być ustawione na wartości takie jak "always", "on-success", "on-failure", "on-abnormal", "on-abort" lub "on-watchdog". Wywołają one ponowne uruchomienie zgodnie ze sposobem, w jaki usługa została zatrzymana.

SYSTEMD SERVICE FILE

Dyrektywy sekcji [Install]:

- WantedBy - Wants [Unit]
- RequiredBy - Requires [Unit]

Sekcja [Install] jest bardzo ważna, ponieważ jest to mechanizm pozwalający na włączanie jednostek bez modyfikowania żadnych plików konfiguracyjnych. Podczas normalnego działania systemd całkowicie ignoruje sekcję [Install]. Założmy jednak, że usługa jest wyłączona i chcielibyśmy ją włączyć. W takiej sytuacji system odczyta zawartość sekcji [Install], a to oznacza, że włączenie serwisu zmusza go do sprawdzenia zależności multi-user.target wymienianej w opcji WantedBy lub RequiredBy. W efekcie system tworzy dowiązanie symboliczne do pliku serwisu w katalogu konfiguracji systemu.

PLANOWANIE ZADAŃ I ZADAŃ CYKLICZNYCH



CRON VS AT

- **cron jest to wbudowane narzędzie systemowe, które odpowiedzialne jest za planowanie powtarzających się zadań, takich jak cykliczne uruchamianie skryptu lub backup.**

jedna definicja = wiele (lub jedno) zadanie

- **at jest narzędziem używanym do zaplanowania zadania do późniejszego wykonania, at odczytuje polecenia ze standardowego wejścia i grupuje je w zadanie, które wykonuje się tylko raz.**

jedna definicja = jedno zadanie

CRON

mm hh dm mth dw komenda

* * * * * echo „test cron job” > /tmp/plik1

0 22 */10 6-8 * echo „test cron job” > /tmp/plik1

Interpretacja: O godzinie 22:00 każdego 10 dnia miesiąca w każdym miesiącu od czerwca do sierpnia.

<https://crontab.guru/>

AT

```
echo „test cron job” > /tmp/plik1 | at 14:25
```

```
echo „test cron job” > /tmp/plik1 | at 1pm
```

```
echo „test cron job” > /tmp/plik1 | at 8:00 tomorrow
```

```
echo „test cron job” > /tmp/plik1 | at now +5 minutes
```

```
echo „test cron job” > /tmp/plik1 | at now +16 hours
```

FILESYSTEMY LINUX



CECHY FILESYSTEMU

- Hierarchiczny
- Nazwa pliku jest tylko jedną z właściwości jego inode'u, który jest najbardziej podstawowym i fundamentalnym obiektem
- Inody są przechowywane na filesystemie i zawierają takie informacje jak prawa, rozmiar timestamp itd.
- Lista wspieranych typów filesystemu dostępna w `/proc/filesystems`
- system plikowy - urządzenie blokowe - możemy go kopiować do pliku i używać jakby to był system plikowy (`dd`)

TYPY

EXT4:

- **ext4 (ang. Fourth Extended File System) - czwarta wersja rozszerzonego systemu plików, następca ext3**
- **do niedawna najbardziej popularny system plików Linux**

XFS:

- **64-bitowy system plików zaprojektowany z przeznaczeniem do użycia go w systemie operacyjnym IRIX (wersja UNIX-a firmy SGI). Aktualnie jest dostępna również jego implementacja dla systemu Linux**
- **XFS pozwala na obsługę dużych dysków twardych, maksymalny rozmiar woluminu jest ograniczony do 16 milionów TB, natomiast rozmiar pojedynczego pliku może wynosić maksymalnie 2^{63} bajtów czyli ponad 8 milionów TB (dokładnie 8 388 608 TB)**
- **XFS posiada szereg cech zaawansowanego systemu plików do zastosowań serwerowych oraz dla wydajnych stacji roboczych**
- **realtime subvolume - udostępnienie procesowi możliwości zarezerwowania dla siebie pasma dostępu do podanego pliku, o podanej szerokości (w bajtach na sekundę)**

TYPY

JFS:

- JFS (ang. Journaled File System) - 64-bitowy system plików z księgowaniem, opracowany przez firmę IBM
- IBM wprowadził JFS do systemu AIX 3.1 i był on głównym systemem plików dla AIX-a przez ponad 10 lat
- zapis plików poprzez transakcje - zapis w dzienniku zdarzenia aktualizacji, zanim rzeczywiście uaktualni się plik, gdy operacja przebiegnie pomyślnie, wpis w dzienniku jest oznaczany jako wykonany
- mniejsza wydajność niż XFS, BTRFS, ZFS właśnie przez dodatkową czynność zapisu do dziennika

BTRFS:

- produkt Oracle, skrót od „B-tree File System”, powstał jako odpowiedź na braki w systemach plików Linuksowych, które nie zapewniały same z siebie takich funkcji jak np. snapshoty czy sumy kontrolne
- CoW (Copy on Write) - większa wydajność i mniejsze użycie zasobów
- Redhat się wycofał ze względu na umowy z Oraclem - został wprowadzony by ominąć ZFSa - ze względu na licencjonowanie



PO CO TYLE TYPÓW ??

- Różne zastosowania produkcyjne
- Licencjonowanie
- Wydajność vs bezpieczeństwo danych
- Prostota obsługi oraz utrzymania

HARD AND SOFT LINKS



SOFT LINKS / HARD LINKS

SOFT LINK:

- Dowiązanie miękkie - skrót, po usunięciu plik dowiązania cały czas istnieje ale nie wskazuje na nic (bo plik usunęliśmy).
Liczba i-nodów (i-węzłów) = 1

HARD LINK:

- Dowiązanie twarde - duplikat, wszystkie zmiany dokonane na dowiązaniu lub na pliku oryginalnym są widoczne wszędzie, po usunięciu pliku link zostaje i nie usuwa się, nadal jest dostępny, prawa nadawane na plik lub dowiązanie są powielane na duplikacie.
Liczba i-nodów (i-węzłów) = 2.

DYSKI I PARTYCJE



RODZAJE PARTYCJI

Istnieją dwa standardy układu tabeli partycjonowania:

- **MBR:**
 - Standard znany także jako ms-dos, można uznać za standard pierwotny. MBR to wciąż najbardziej powszechnie stosowana tabela partycjonowania, która wiąże się z ograniczeniami.
- **GPT:**
 - Nowszy typ rekordu jest częścią standardu UEFI (Unified Extensible Firmware Interface), następcy nieco podstarzałego BIOS-u. Standard ten istnieje już od wielu lat, ale w przeszłości nie był wspierany przez producentów sprzętu. Zmiana nastąpiła z nadejściem systemu Windows 8 czyli około 2012 roku, kiedy standard ten stał się wymagany przez Microsoft dla swojego nowego systemu.

RODZAJE PARTYCJI - LIMITY

MBR:

- Standard BIOS
- Nie pozwala utworzyć więcej niż czterech partycji głównych. Te partycje są określane jako główne.
- Rozmiar partycji dysku nie może przekroczyć 2 TB.

GPT:

- Standard UEFI
- Partycjonowanie innymi narzędziami niż MBR
- Kompatybilny wstecznie
- Pozwala utworzyć do 128 partycji podstawowych
- Maksymalny rozmiar pojedynczej partycji to 9.4 ZiB (zebibitów)

MBR VS GPT

- **Master Boot Record**

- Boot loader (446 bytes)
- Partition table (64 bytes)
- Magic number (2 bytes)

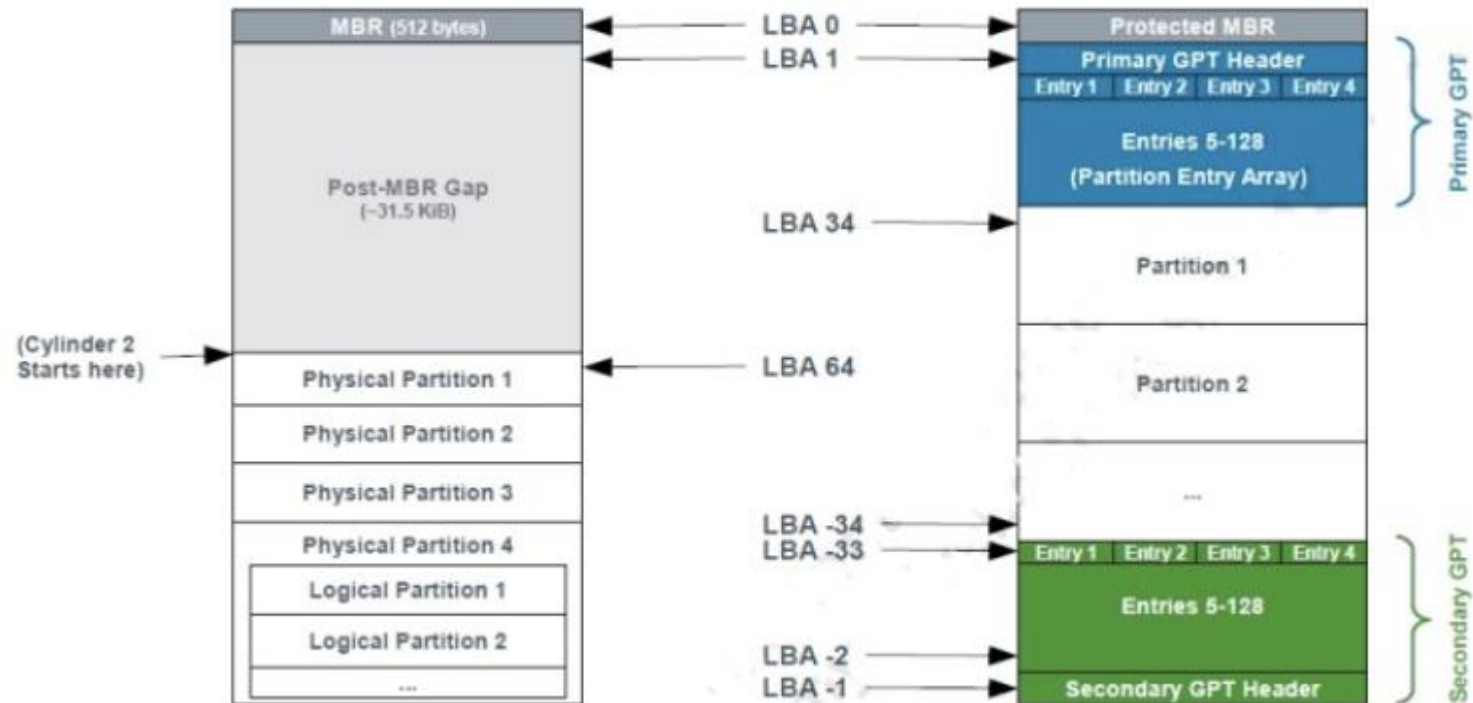
- Limited to 2 TiB hard disks

- **Globally Unique Partition Table**

- Protected MBR (512 bytes)
- GPT Header (512 bytes)
- Partition entries (128 bytes ea.)

- Limited to 9.2 ZiB hard disks

Konwersja jest
możliwa jednak
wiąże się z utratą
danych !!!



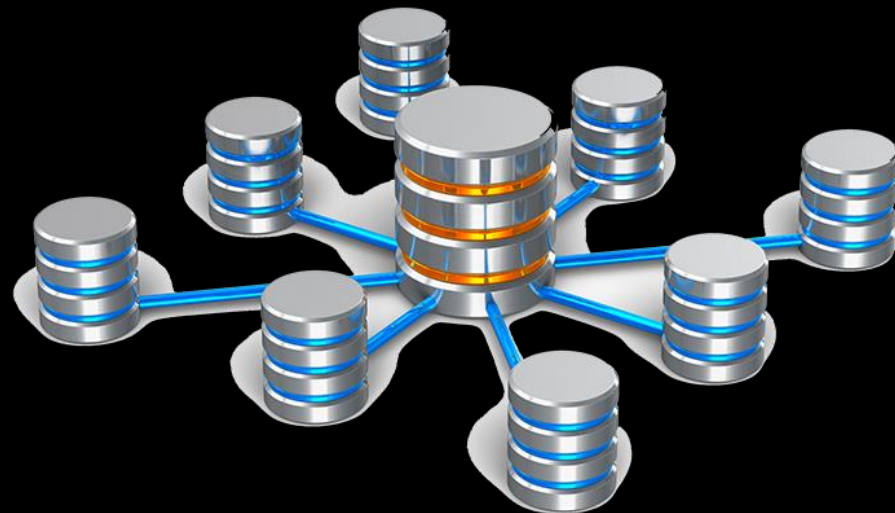


MAGIC COMMAND

```
$ for i in `seq 0 31`; do echo "- - -" > /sys/class/scsi_host/host$i/scan; done
```

LVM

LOGICAL VOLUME MANAGEMENT



LVM

- Logical Volume Management dzieli jedną wirtualną partycję na wiele kawałków, z których każdy może znajdować się na różnych partycjach i/lub dyskach.
- Łatwość w rozszerzalności zarówno dla partycji jak i systemu plikowego.
- Fizyczne wolumeny grupowane w logiczne byty: pvs, vgs, lvs.
- LVM ma lepszą skalowalność niż RAID:
 - wolumeny logiczne mogą być łatwo zmieniane
 - możliwość dodawania nowych urządzeń

LVM - STRUKTURA

Fizyczny Dysk

Wolumen fizyczny

Grupa wolumenowa

Logiczny wolumen

Filesystem

+

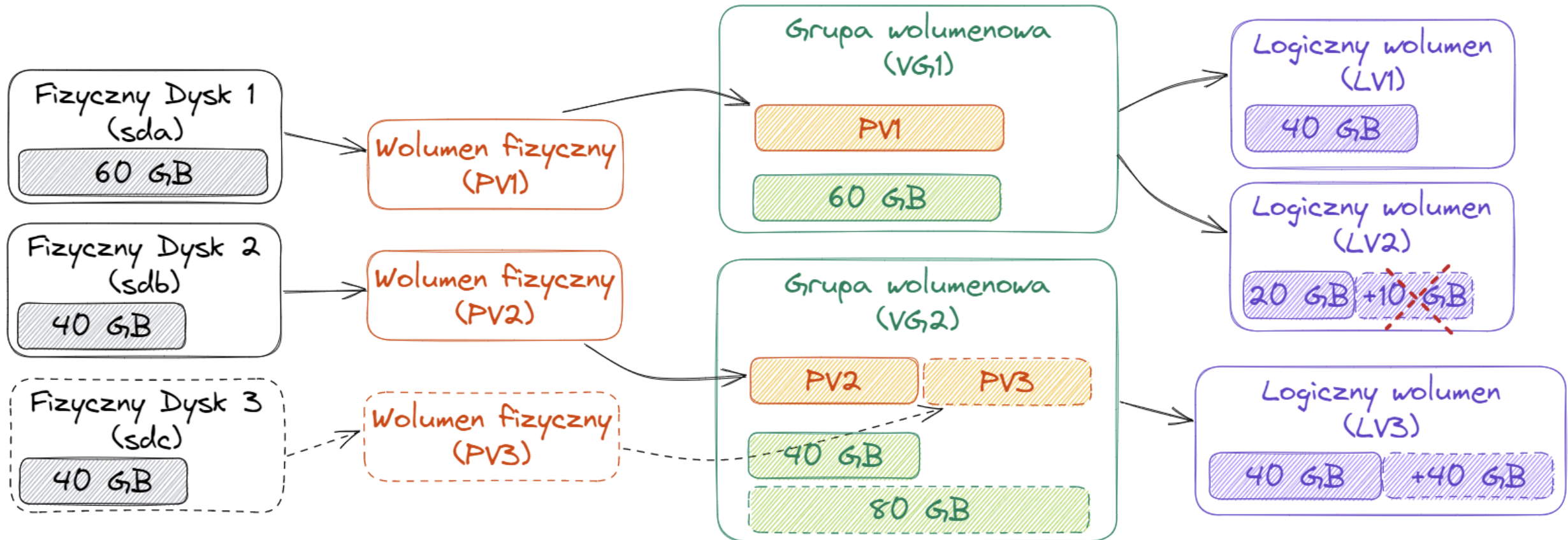
Fizyczny Dysk

Wolumen fizyczny

Logiczny wolumen

Filesystem

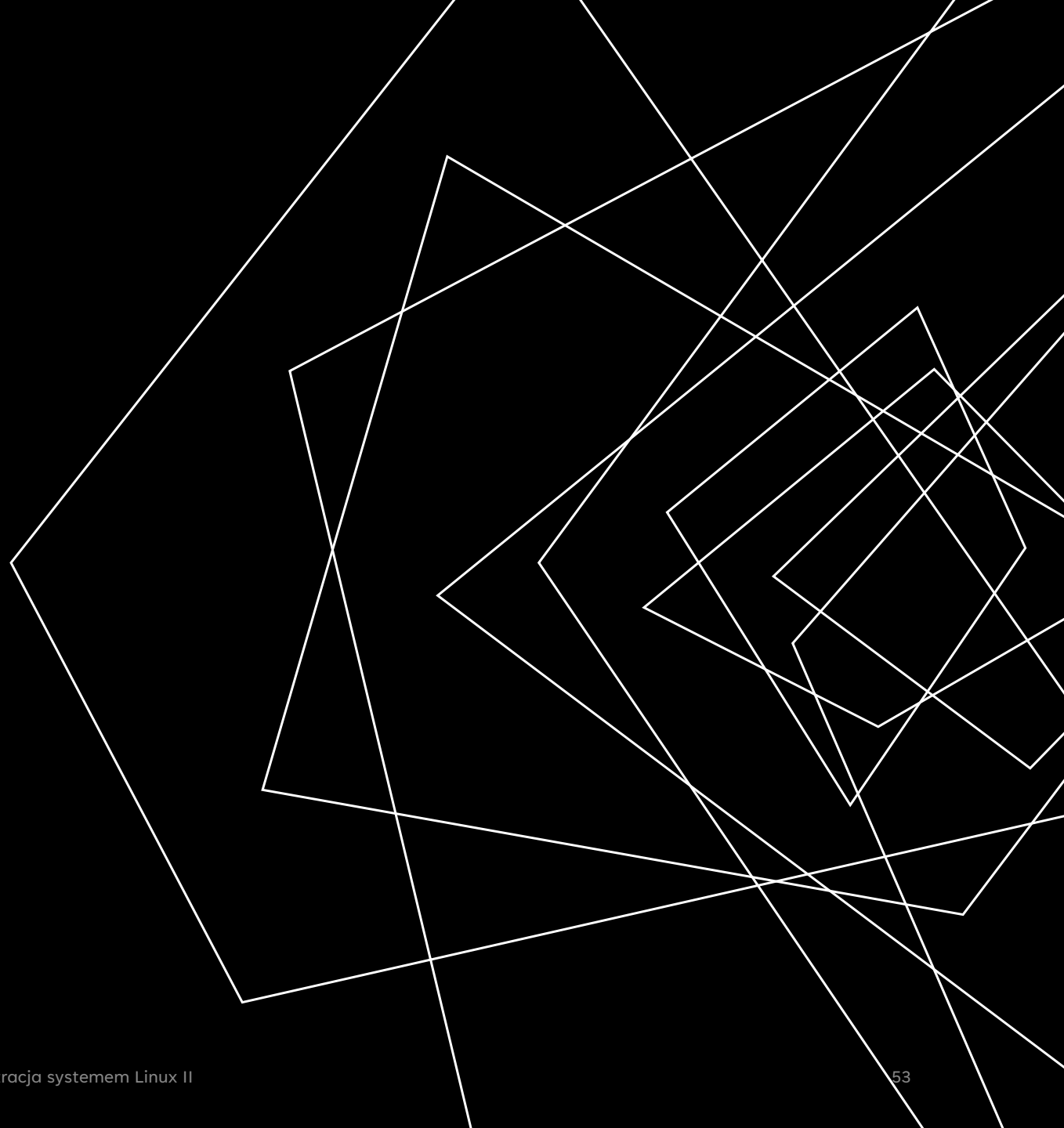
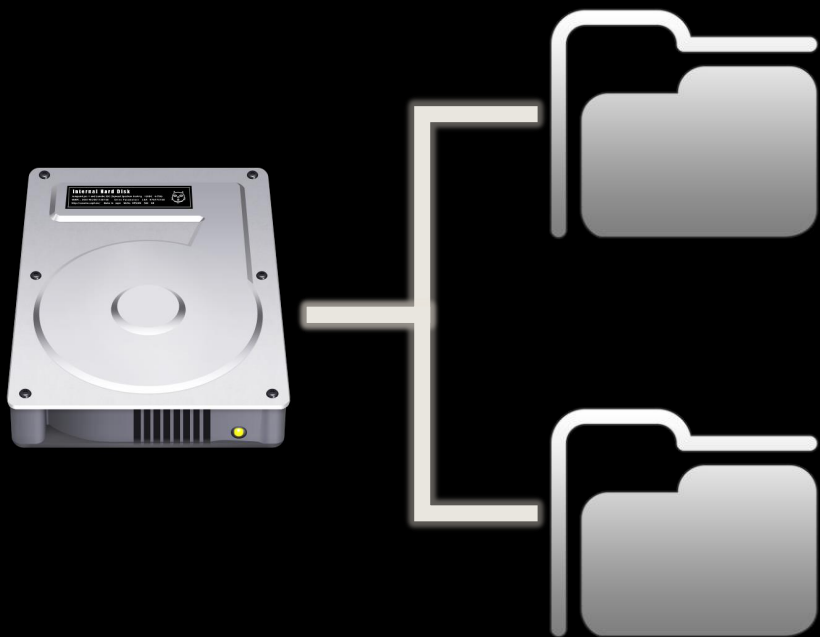
LVM - ARCHITEKTURA



SUBLV

- SubLV inaczej subwolumen
- Wolumen Logiczny (LV) jest często kombinacją innych ukrytych wolumenów logicznych, np. podczas tworzenia mirror/stripe LV
- SubLV korzystają z urządzeń fizycznych lub z innych SubLV
- SubLV przechowują bloki danych wolumenów logicznych, bloki parzystości RAID oraz metadane RAID

MONTOWANIE ZASBÓW DYSKOWYCH



/ETC/FSTAB

Struktura pliku

- **Device:** zwykle podana nazwa lub UUID urządzenia (sda1/sda2/etc)
- **Mount Point:** wskazuje katalog, gdzie urządzenie będzie montowane
- **File System Type:** pokazuje typ używanego systemu plików
- **Options:** aktywne opcje montowania, w przypadku korzystania z wielu opcji należy je oddzielić przecinkami
- **Backup Operation:** definicja backupu dla fs (1/0), w którym:
 - 1 = kopia zapasowa narzędzia zrzutu partycji
 - 0 = brak kopii zapasowej
- **File System Check Order:** definicja kolejności sprawdzania. 0 oznacza, że fsck nie sprawdzi systemu plików. Liczby wyższe niż to reprezentują kolejność czeków. Główny system plików powinien być ustawiony na 1, a inne partycje na 2.

OPCJE MONTOWANIA

DEFAULT:

- **rw** (read-write);
- **suid** (respect setuid and setgid bits);
- **dev** (interpret characters and block devices on the filesystem);
- **exec** (allow executing binaries and scripts);
- **auto** (mount the filesystem when the -a option of the mount command is used);
- **nouser** (make the filesystem not mountable by a standard user);
- **async** (perform I/O operations on the filesystem asynchronously).

SWAP



SWAP

- Plik wymiana możliwy na jednej lub więcej dedykowanych partycji lub plikach.
- Można przydzielać swap dynamicznie (on-demand)
- Każdy obszar ma swój priorytet i obszary o niższym priorytecie nie są używane dopóki obszary o wyższym priorytecie nie zostaną zapełnione.
- W większości sytuacji zalecany rozmiar swapu to całkowita ilość pamięci RAM w systemie.
- Pamięć SWAP w większości czasu jest używana do buforowania zawartości plików, aby nie trafiała na dysk więcej niż jest to konieczne, lub w nieoptymalnej kolejności lub czasie.
- Warto również zauważyć, że w pamięć Linuxa używana przez jądro, w przeciwieństwie do pamięci aplikacji, nigdy nie jest swapowana (w porównaniu do innych systemów operacyjnych)

SZYFROWANIE / LUKS



SZYFROWANIE DYSKÓW

- Enkrypcja na poziomie urządzeń blokowych
- Linux Unified Key Setup (LUKS) - domyślnie zaprojektowane narzędzie pod Linuxa
- LUKS używa modułu „cryptsetup”
- Łatwa migracja partycji na inne dyski lub systemy
- Może być użyta do szyfrowania SWAPa

MACIERZE DYSKOWE / RAID



MACIERZE / RAID

(REDUNDANT ARRAY OF INDEPENDENT DISKS)

- RAID (Redundant Array of Independent Disks) rozkłada operacje wejścia/wyjścia na wiele dysków. Zwiększa tym wydajność w interfejsach kontrolerów dyskowych, takich jak SCSI, które mogą wydajnie wykonywać pracę równolegle.
- RAID może być zaimplementowany albo w oprogramowaniu albo w sprzęcie.
- Przy implementacji sprzętowej system operacyjny nie jest w rzeczywistości bezpośrednio świadomy używania RAID.
 - Na przykład trzy dyski twarde 512 GB (dwa na dane, jeden na bit kontroli parzystości) skonfigurowane w RAID-5 będą wyglądać jak pojedynczy dysk o pojemności 1 TB.
- Jednakże jedną z wad stosowania sprzętowego RAID jest to, że w przypadku awarii kontrolera dysku, musi on zostać zastąpiony kompatybilnym, który może nie być łatwy do zdobycia.
- W przypadku korzystania z RAID programowego te same dyski mogą być podłączone i współpracować z dowolnym kontrolerem dysków (sprzęt niskiej i średniej klasy).

MACIERZE / RAID

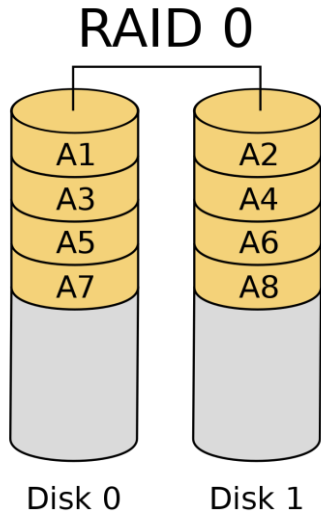
(REDUNDANT ARRAY OF INDEPENDENT DISKS)

- Zaletą urządzeń RAID jest możliwość zapewnienia lepszej wydajności, redundancji, lub obu.
- Striping (RAID 0) zapewnia lepszą wydajność poprzez rozłożenie informacji na wiele urządzeń, dzięki czemu możliwe są jednoczesne zapisy i odczyty.
- Mirroring (RAID 1) zapisuje te same informacje na wielu dyskach, zapewniając lepszą redundancję.
- mdadm jest używany do tworzenia i zarządzania urządzeniami RAID.
- Po utworzeniu, nazwa macierzy `/dev/mdX` może być używana jak każde inne urządzenie, takie jak np. `/dev/sda1`.

RAID - TYPY

- RAID 0: stripping
- RAID 1: mirroring
- RAID 5: striping z rozłożoną parzystością
- RAID 6: striping z podwójnie rozłożoną parzystością
- RAID 10: mirroring i stripping
- Poziomy RAID mogą być łączone/mixowane, np.:
 - RAID 10
 - RAID 01

RAID - ARCHITEKTURA

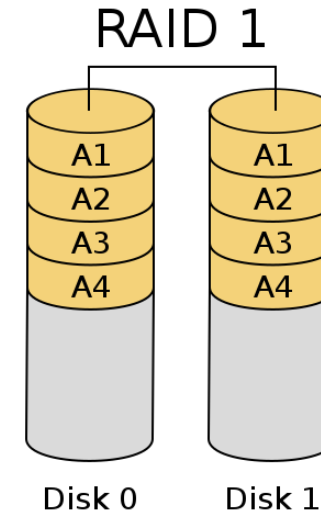


Korzyści:

- przestrzeń wszystkich dysków jest widziana jako całość
- przyspieszenie zapisu i odczytu w porównaniu do pojedynczego dysku

Wady:

- brak odporności na awarię dysków
- awaria pojedynczego dysku powoduje utratę wolumenu



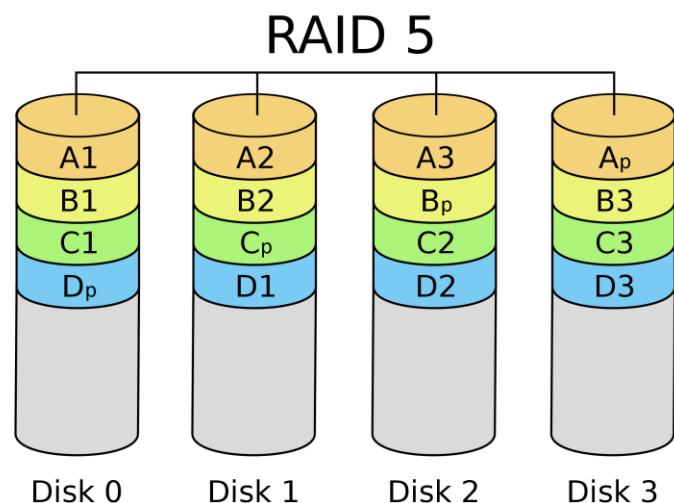
Korzyści:

- odporność na awarię N - 1 dysków przy N-dyskowej macierzy
- możliwe zwiększenie szybkości odczytu
- możliwe zmniejszenie czasu dostępu

Wady:

- możliwa zmniejszona szybkość zapisu
- utrata pojemności (całkowita pojemność jest taka jak pojemność najmniejszego dysku)

RAID - ARCHITEKTURA

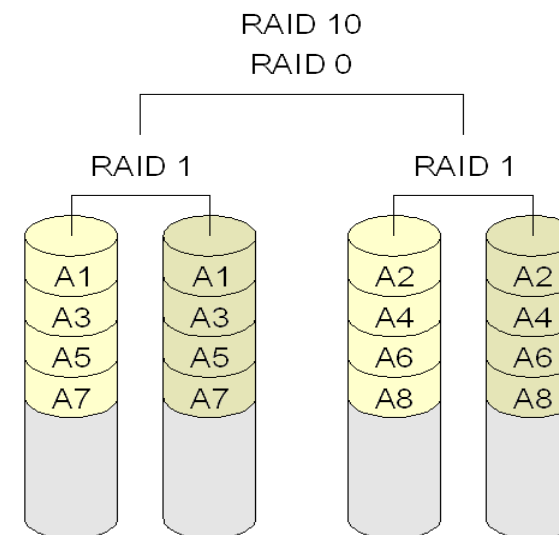


Korzyści:

- odporność na awarię jednego dysku
- zwiększona szybkość odczytu - podobnie jak RAID 0

Wady:

- zmniejszona szybkość zapisu z powodu konieczności obliczania sum kontrolnych
- w przypadku awarii dysku dostęp do danych jest spowolniony z powodu obliczeń danych na podstawie pozostałych danych i sum kontrolnych



Korzyści:

- szybkość macierzy RAID 0
- awaria jednego dysku powoduje wyłączenie jedynie tego dysku, a nie całego dysku logicznego
- prostsza w implementacji niż RAID 3, 5 i 6

Wady:

- większy koszt przechowywania danych (współczynnik nadmiarowości = 100%)

UŻYTKOWNICY I GRUPY



TOŻSAMOŚĆ LINUX

- Zapisana w /etc/passwd
- Standard tożsamości POSIXowej
 - nadrzędne prawo jest prawo właścicielskie

Dwie koncepcje i podejścia dla grup użytkowników:

- koncepcja grupy prywatnej
 - Mandriva / Redhat / Centos / Fedora
- koncepcja grupy publicznej (users)
 - SUSE / FreeBSD

/ETC/SHADOW

- Hasła użytkowników przechowywane są w pliku /etc/shadow.
- Hasła są przechowywane w postaci zahashowanej, aby chronić poufne dane przed dostępem nieautoryzowanym (dokładniej jest to skrót hasha).
- W zależności od konfiguracji systemu, hasła mogą być szyfrowane różnymi algorytmami skrótu: MD5, SHA-256, SHA-512 itp.
- W pliku /etc/shadow przechowywane są również informacje o dacie ostatniej zmiany hasła, liczbie dni, które muszą minąć przed zmianą hasła oraz minimalnej liczbie dni między zmianami hasła.

JOHN THE RIPPER

- Szybkim program do łamania haseł, obecnie dostępnym dla wielu odmian systemu Unix, Windows, DOS, BeOS i OpenVMS. Jego głównym celem jest wykrywanie słabych haseł uniksowych. Oprócz kilku typów haseł crypt(3), najczęściej spotykanych w różnych systemach uniksowych, wspierane są także hashe LM z Windowsa, oraz wiele innych haseł i szyfrów.
- Github repo: <https://github.com/openwall/john>
- Instalacja (przez snap):

```
$ dnf install epel-release -y
$ dnf install snapd -y
$ systemctl enable --now snapd.socket
$ ln -s /var/lib/snapd/snap /snap
$ snap install john-the-ripper
```

HASHOWANIE

- Hashowanie (tworzenie skrótu) najprościej wytłumaczyć jako proces generowania danych wyjściowych o stałym rozmiarze z danych wejściowych o zmiennym rozmiarze.
- Cały proces jest możliwy dzięki zastosowaniu specjalnych wzorów matematycznych znanych pod nazwą tzw. funkcji mieszających (inaczej algorytmów haszujących).
- Dla przykładu algorytm hashujący SHA-256 po przepuszczeniu przez niego jakichkolwiek danych (czyt. danych wejściowych) wyprodukuje skrót o długości 256 bitów, podczas gdy inny algorytm, SHA-1 zawsze wygeneruje skrót o długości 160 bitów.
- SHA = Secure Hash Algorithms (Bezpieczne Algorytmy Mieszające)

HASHOWANIE

Z technicznego punktu widzenia funkcja skrótu kryptograficznego musi spełniać trzy właściwości, aby uznać ją za skuteczną i bezpieczną:

- odporność na kolizję: właściwość dzięki której niemożliwe jest znalezienie dwóch odrębnych danych wejściowych, które w wyniku przepuszczenia ich przez funkcję hashującą dadzą taki sam hash.
- preimage resistance: właściwość dzięki której niemożliwe jest "przywrócenie" danych przepuszczonych przez funkcję skrótu (odkrycie danych wejściowych z posiadanych danych wyjściowych).
- second-preimage resistance: właściwość dzięki której niemożliwe jest odnalezienie drugiego zestawu danych wejściowych, który koliduje z innym określonym zestawem danych wejściowych.

HASHOWANIE

```
$ openssl passwd -6 -salt xyz mojesilnehaslo
```

Gdzie:

- -1 wygeneruje hasło hashowane algorytmem MD5
- -5 wygeneruje hasło hashowane algorytmem SHA256
- -6 wygeneruje hasło hashowane algorytmem SHA512 (zalecane)

```
$ echo "mojesilnehaslo" | md5sum
```

```
$ echo "mojesilnehaslo" | sha256sum
```

```
$ echo "mojesilnehaslo" | sha512sum
```


/ETC/SUDOERS

Definicja nadania uprawnień ma postać:

KTO GDZIE=(JAKO_KTO) POLECENIE

- **KTO** - pole wskazuje na listę użytkowników i grup (grupy rozpoczyna się od znaku %), których definicja dotyczy
- **GDZIE** - pole jest istotne tylko gdy konfiguracja sudo współdzielona jest pomiędzy wieloma komputerami, wskazuje na komputery, których definicja dotyczy, w środowiskach jedno komputerowych można użyć słów **ALL** lub **localhost**
- **JAKO_KTO** - lista użytkowników, których uprawnienia można wykorzystać. W konfiguracji pole to można zaniechać
- **POLECENIE** - polecenia lub lista poleceń, których definicja ta dotyczy Lista to wymienione nazwy po przecinkach

ROZSZERZONE PRAWA DO PLIKÓW



SUID

SetUID - pozwala na określenie pod jakim identyfikatorem użytkownika jest wykonywany program, dla plików oznacza to, że proces będzie miał prawa użytkownika który jest właścicielem pliku wykonywalnego, a nie użytkownika ten plik uruchamiającego.

Trzeba być bardzo uważnym ustawiając ten bit i nie nadawać praw administratora (root-a) przez przypadek. Przykładem takim jest "passwd", "mount", "su" czy "ping". Ustawiany przyjmuje wartość 4 co oznacza binarnie 100.

```
$ chmod 4755 plik
```

SGID

SetGID - pozwala na określenie pod jakim identyfikatorem grupy jest wykonywany program, w przypadku użycia z katalogiem wszystkie podkatalogi i pliki w nim tworzone posiadają grupę katalogu nadrzędnego.

Ustawiany jest jako drugi bit cyfry ósemkowej (binarnie 010).

```
$ chmod 2755 plik  
$ chmod 2644 plik
```

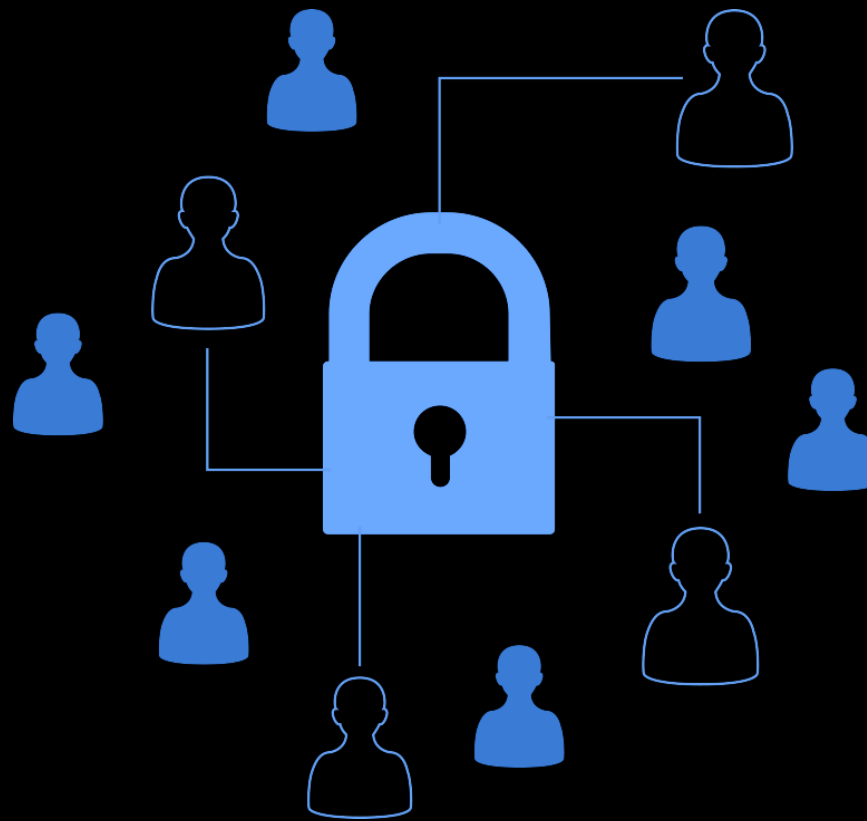
BIT STICKY

Bit Sticky - pozwala programowi na pozostanie w pamięci po zakończeniu działania. Jest pozostałością z przeszłości, ponieważ nie ma potrzeby aby programy pozostawały w pamięci po zakończeniu. Najczęściej stosuje się go do katalogu. Gdy jest użyty z katalogiem to użytkownicy mogą kasować tylko te pliki, dla których mają wprost uprawnienia do zapisu, nawet jeśli mają prawo zapisu do katalogu, np. katalog /tmp.

Ustawia się go jako dodatkowa 1 przed standardowymi 3 bitami.

```
$ chmod 1755 plik
```

ACLKI



ACCESS CONTROL LIST

- ACL – (Access Control List), które zapewniają możliwość ustawienia na plikach i katalogach rozszerzonej grupy uprawnień
- Aby możliwe było użycie ACL system plików musi być podmontowany z włączonym wsparciem dla ACL, opcja `acl` musi być dodana do montowanego systemu plików w pliku `/etc/fstab`. W systemie plików XFS takie wsparcie jest domyślnie włączone, w systemach plików EXT wsparcie ACL musi być aktywowane
- ACL podzielone są na dwie kategorie:
 - access ACL – dostępowe ACLki, ustawiane na plikach i/lub katalogach
 - default ACL – domyślne ACLki, ustawiane tylko na poziomie katalogu. Pliki i podkatalogi wchodzące w skład katalogu z ustawionymi domyślnymi ACL dziedziczą domyślne ACL katalogu nadrzędnego

ACCESS CONTROL LIST

Po czym poznać, że plik ma ACLki?

Znak plus (+) informuje o przypisanych do pliku lub katalogu listach ACL.

```
$ ls -l
total 0
drwxrwx---+ 2 student student 6 Jul 6 14:04 katalog
```

Ciekawostka:

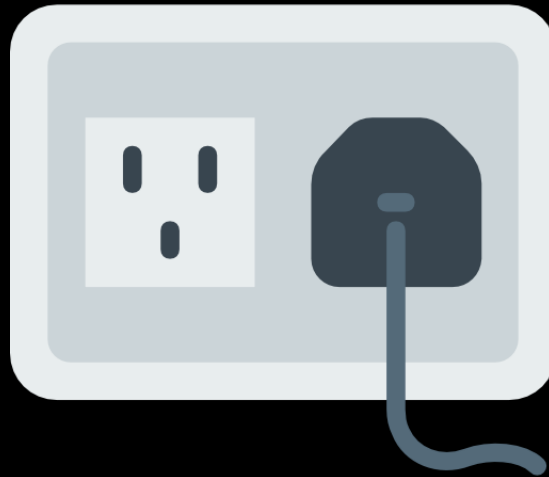
`tar` nie wspiera ACL, należy użyć `star`, który pracuje z tymi samymi opcjami co `tar` a dodatkowo wspiera ACL

ACLKI - ZAKRES

Zakres:

- **u[ser]:UID:perms:** uprawnienia przypisywane do konkretnego użytkownika (nazwa użytkownika lub UID). Użytkownik musi się znajdować w pliku `/etc/passwd`
- **g[roup]:GID:perms:** uprawnienia przypisywane do konkretnej grupy (nazwa użytkownika lub GID). Użytkownik musi się znajdować w pliku `/etc/group`
- **m[ask]:perms:** maksymalne uprawnienia jakie konkretny użytkownik lub grupa może mieć na pliku lub katalogu. Np. `rw-` oznacza, że żaden użytkownik lub grupa nie będzie miał większych uprawnień niż odczyt i zapis
- **o[ther]:perms:** uprawnienia przypisane użytkownikom nie należącym do grupy będącej właścicielem.

PLUGGABLE AUTHENTICATION MODULES



PAM - CHARAKTERYSTYKA

- Zastępuje tradycyjne metody autentykacji użytkowników (passwd / shadow)
- Elastyczny system autentykacji umożliwiający tworzenie aplikacji całkowicie niezależnych od metody autentykacji
- Modularny mechanizm działania - używa zestawu „modułów autentykacyjnych”, które umożliwiają implementację dowolnych metod autentykacji i używanie ich z dowolnymi aplikacjami
- PAM używa osobnej biblioteki, z której korzystają aplikacje wymagające metod autentykacji użytkowników
- Pozwala mieszać metody oraz tworzyć własne mechanizmy autoryzacji
- Biblioteka „libpam” znajduje się w katalogu /usr/lib64, a moduły, noszące nazwy zaczynające się od „pam_” znajdują się w katalogu /usr/lib64/security

PAM – DEEP DIVE

Aplikacja jest wyposażona w interfejs do biblioteki Linux-PAM i jest niezależna od metody autentykacji. Aby dokonać autentykacji użytkownika, aplikacja komunikuje się z Linux-PAM. System PAM określa na podstawie konfiguracji, odpowiedniej dla danej aplikacji sposób autentykacji i określa, które moduły i w jaki sposób będą realizowały autentykację.

Aplikacja może opcjonalnie udostępniać PAM własny moduł realizujący konwersację z modułami PAM. Po ustaleniu uprawnień PAM przesyła odpowiednie informacje o uprawnieniach użytkownika do aplikacji.

PAM – DEEP DIVE

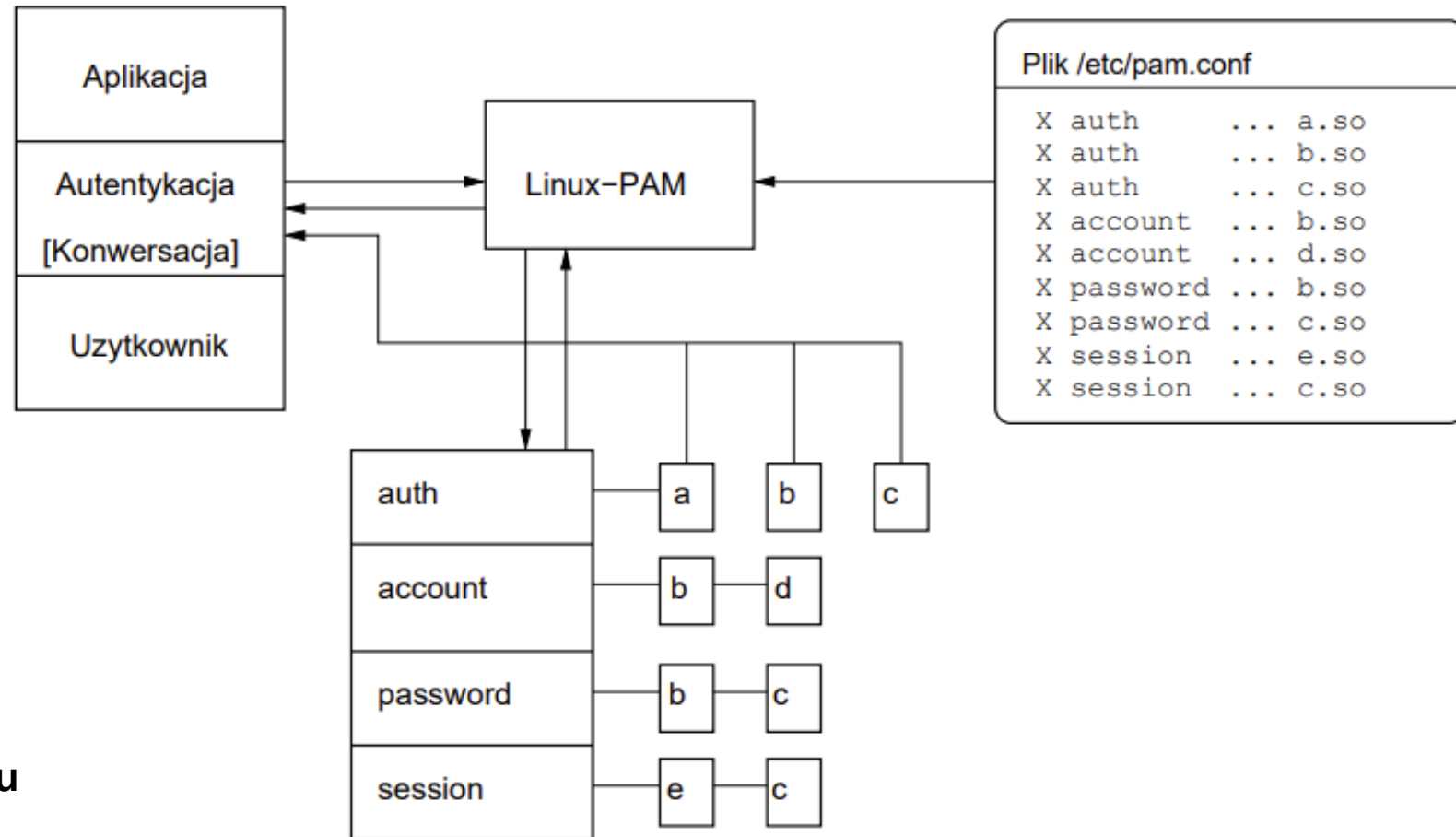
System PAM ma cztery podstawowe grupy modułów:

- moduł auth (ang. authentication management) - Umożliwia identyfikację użytkownika na podstawie hasła lub innych metod. Oprócz tego ma możliwość przypisania użytkownika do odpowiedniej grupy.
- moduł account (ang. account management) - Zarządzanie kontami niezależne od autentykacji użytkowników. Moduł ustala uprawnienia na podstawie dostępnych zasobów systemu, pory dnia, czy lokalizacji użytkownika.
- moduł session (ang. session management) - Zarządzanie sesją obejmuje działania, jakie system podejmuje przed otwarciem czy zamknięciem sesji użytkownika. Działania te mogą obejmować rejestrowanie informacji o użytkowniku, modyfikacje środowiska lub montowanie odpowiednich systemów plików.
- moduł password (ang. password management) - Odpowiedzialny za uaktualnianie baz danych o użytkownikach, zawierających ich hasła, dane osobowe, czy listy uprawnień.

PAM - ARCHITEKTURA

Proces uwierzytelnienia:

1. Aplikacja łączy się Linux-PAM
2. Linux-PAM wczytuje plik konfiguracyjny odpowiedni dla tej aplikacji
3. Linux-PAM wykonuje po kolei moduły ("akcje") zdefiniowane w kolejnych wierszach pliku konfiguracyjnego
Uwaga: może się zdarzyć, że nie wszystkie zostaną wykonane (np. wybrany moduł może przerwać proces uwierzytelnienia)
4. Aplikacja dostaje zwrotną informację o pomyślności przebiegu procesu uwierzytelnienia.



PAM - KONFIGURACJA

Konfiguracja PAM znajduje się w pliku `/etc/pam.conf` lub w plikach w katalogu `/etc/pam.d`, gdzie nazwa pliku wskazuje na nazwę usługi. Składnia plików konfiguracyjnych ma postać:

```
(service-name)  module-type  control-flag  module-path  arguments
```

Gdzie:

- **service-name:** tylko dla `pam.conf`, określa typ usługi (`login`, `su`, `ftpd`)
- **module-type:** określa moduł (`auth`, `account`, `session`, `password`)
- **control-flag:** definiuje reakcję systemu PAM na rezultat pracy modułu (`requisite`, `required`, `sufficient`, `optional`)
- **module-path:** pełna ścieżka dostępu do danego modułu
- **arguments:** opcjonalne argumenty dla modułu

PAM – KONTROLA ZACHOWANIA

- **requisite:**– gdy pojawi się błąd, mechanizm PAM natychmiast kończy działanie
- **required:** mechanizm PAM zgłosi błąd ale analiza stosu reguł będzie przebiegać dalej (przydatne gdy testy dostępu są wielostopniowe, czyli wiele modułów z właśnie tą zasadą nie zgłosi błędu)
- **sufficient:** sukces tego modułu jest wystarczający do zakończenia działania mechanizmu PAM, chyba że wcześniej któryś z modułów z regułą **required** zgłosił błąd (rezultat OK – sukces)
- **optional:** jeżeli jest to jedyny moduł w danej sekcji, rezultat jego testu jest ważny dla mechanizmu PAM, w innym przypadku jest ignorowany

PAM – KONTROLA ZACHOWANIA

 required	X	✓	✓	✓
 required	✓	✓	✓	✓
 requisite	✓	X	✓	✓
 optional	✓		X	X
 sufficient	✓		✓	X
 requisite				✓
Results	X	X	✓	✓

PAM MODULES

Nazwa modułu	Funkcja	Typ
pam_cracklib	eliminuje słabe hasła	Standardowy
pam_deny	realizuje odmowę dostępu	Standardowy
pam_time	ogranicza dostęp na podstawie ram czasowych	Standardowy
pam_group	przydziela do grup użytkowników	Standardowy
pam_limits	ogranicza dostęp na podstawie dostępnych zasobów	Standardowy
pam_listfile	reguluje dostęp na podstawie list użytkowników	Standardowy
pam_tally	realizuje odmowę dostępu w zależności od liczby nieudanych logowań	Standardowy
pam_unix	realizuje standardowe uniksowe zarządzanie autentykacją	Standardowy
pam_warn	moduł logujący informacje do Syslog	Standardowy
pam_smb	grupa modułów współpracujących z serwerem Samba	Dodatkowy
pam_mysql	autentykacja na podstawie bazy danych MySQL	Dodatkowy
pam_ldap	współpraca z serwisem LDAP	Dodatkowy
pam_proftpd	autentykacja dla serwera ProFtpd	Dodatkowy

PAM – USE CASES

Możliwość logowania się root-a bez podawania hasła

- **Pierwsza linia pliku /etc/pam.d/system-auth powinna wyglądać następująco:**

```
auth requisite pam_rootok.so
```

Możliwość logowania się użytkownika o UID 500 bez podawania poprawnego hasła, czyli praktycznie bez podawania hasła.

- **Cała sekcja w pliku /etc/pam.d/system-auth powinna wyglądać następująco:**

```
auth requisite pam_succeed_if.so uid = 500
```

PAM – USE CASES

Możliwość logowania się dowolnego użytkownika bez podawania hasła.

- **Pierwsza linia pliku /etc/pam.d/system-auth powinna wyglądać następująco:**

```
auth requisite pam_permit.so
```

Blokowanie dostępu do SSH dla wybranych użytkowników.

- **Do pliku /etc/pam.d/sshd dopisujemy wiersz:**

```
account required pam_listfile.so item=user sense=deny  
file=/etc/ssh/sshd.deny onerr=succeed
```

Tworzymy plik /etc/ssh/sshd.deny i w kolejnych wierszach wprowadzamy nazwy użytkowników, dla których ssh ma być niedostępne.

PAM – USE CASES

Możliwość blokowania użytkownika, który uwierzytelnił się błędnie 3 razy oraz odblokowanie jego konta po 5 minutach:

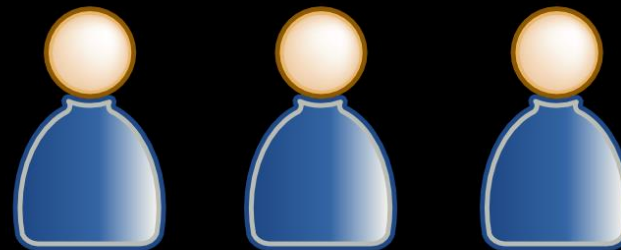
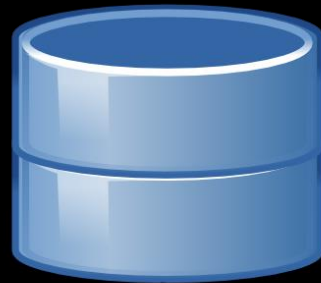
- **Edycja plików /etc/pam.d/system-auth oraz /etc/pam.d/password-auth i dodanie na początku plików linijki:**

```
auth required pam_tally2.so file=/var/log/tallylog deny=3  
even_deny_root unlock_time=300
```

A następnie w 3 linijce w sekcji „account”:

```
account          required          pam_tally2.so
```

LDAP



LDAP

LDAP (Lightweight Directory Access Protocol) to protokół sieciowy, który służy do zarządzania katalogami, takimi jak katalog użytkowników, grup i innych obiektów. LDAP składa się z następujących elementów:

- **Root DSE (Directory Service Entry):** Jest to najwyższy poziom drzewa katalogu LDAP. Zawiera informacje o samym katalogu, takie jak jego nazwa, wersja, obsługiwane metody uwierzytelniania itd.
- **Schemat:** jest to struktura, która opisuje, jakie atrybuty i klasy obiektów są przechowywane w katalogu. Schemat LDAP jest zdefiniowany w pliku schematu, który jest przechowywany na serwerze LDAP. Dzięki schematom LDAP można łatwo definiować i zarządzać obiektami w katalogu LDAP.
- **Entry:** reprezentuje pojedynczy obiekt w katalogu, taki jak użytkownik, grupa lub jednostka organizacyjna (OU). Każdy wpis jest jednoznacznie identyfikowany przez unikalną nazwę Distinguished Name (DN)
 - **Attribute=Value**

LDAP - ARCHITEKTURA

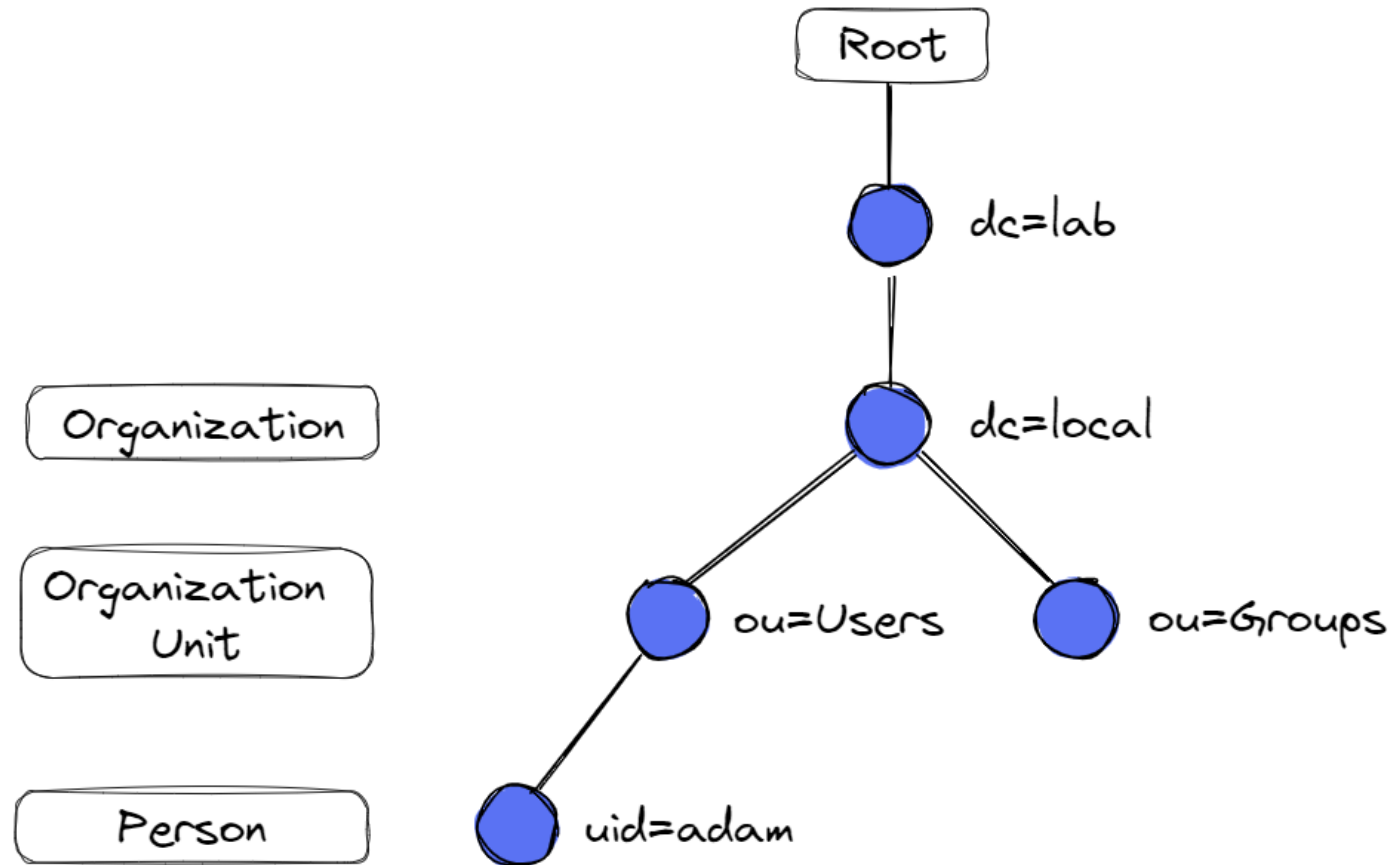
Architektura LDAP składa się z kilku elementów

- **klientów LDAP:** jest aplikacją lub urządzeniem, które łączy się z serwerem LDAP w celu uzyskania dostępu do katalogu
- **serwerów LDAP:** oprogramowanie, które zarządza katalogiem LDAP i udostępnia go klientom
- **protokołu LDAP:** zestaw zasad i procedur, które służą do przesyłania danych między klientami LDAP, a serwerami LDAP

Architektura LDAP jest hierarchiczna!

LDAP - ARCHITEKTURA

Drzewo LDAP



LDIF

LDAP posiada swój format pliku zwany LDIF (Ldap data interchange format), który reprezentuje dane i polecenia LDAP. LDIF działa w oparciu o pary klucz-wartość:

- **jedną deklaracją na linię**
- **klucz i wartość odseparowana dwukropkiem i spacją**

```
dn: cn=Dev,ou=Groups,dc=lab,dc=local
objectClass: top
objectClass: posixGroup
gidNumber: 600
```

KERBEROS



KERBEROS

Kerberos to protokół bezpieczeństwa sieci komputerowej, który uwierzytelnia żądania usług pomiędzy dwoma lub więcej zaufanymi hostami w niezaufanej sieci (np. jak Internet).

Wykorzystuje on kryptografię tajnych kluczy do uwierzytelniania aplikacji klient-serwer i weryfikacji tożsamości użytkowników.

Kerberos jest używany między innymi w uwierzytelnianiu Posix, Active Directory, NFS i Samba. Jest to również alternatywny system uwierzytelniania dla SSH, POP i SMTP.

KDC

Kerberos wykorzystuje kryptografię klucza symetrycznego i Key Distribution Center (KDC) do uwierzytelniania i weryfikacji tożsamości użytkowników.

KDC obejmuje trzy aspekty:

- ticket-granting server (TGS): serwer przyznający tickety, który łączy użytkownika z serwerem usług
- baza danych Kerberos: przechowuje hasła i identyfikatory wszystkich zweryfikowanych użytkowników
- authentication server (AS): serwer uwierzytelniający, który przeprowadza wstępne uwierzytelnienie

UWIERZYTELNIANIE

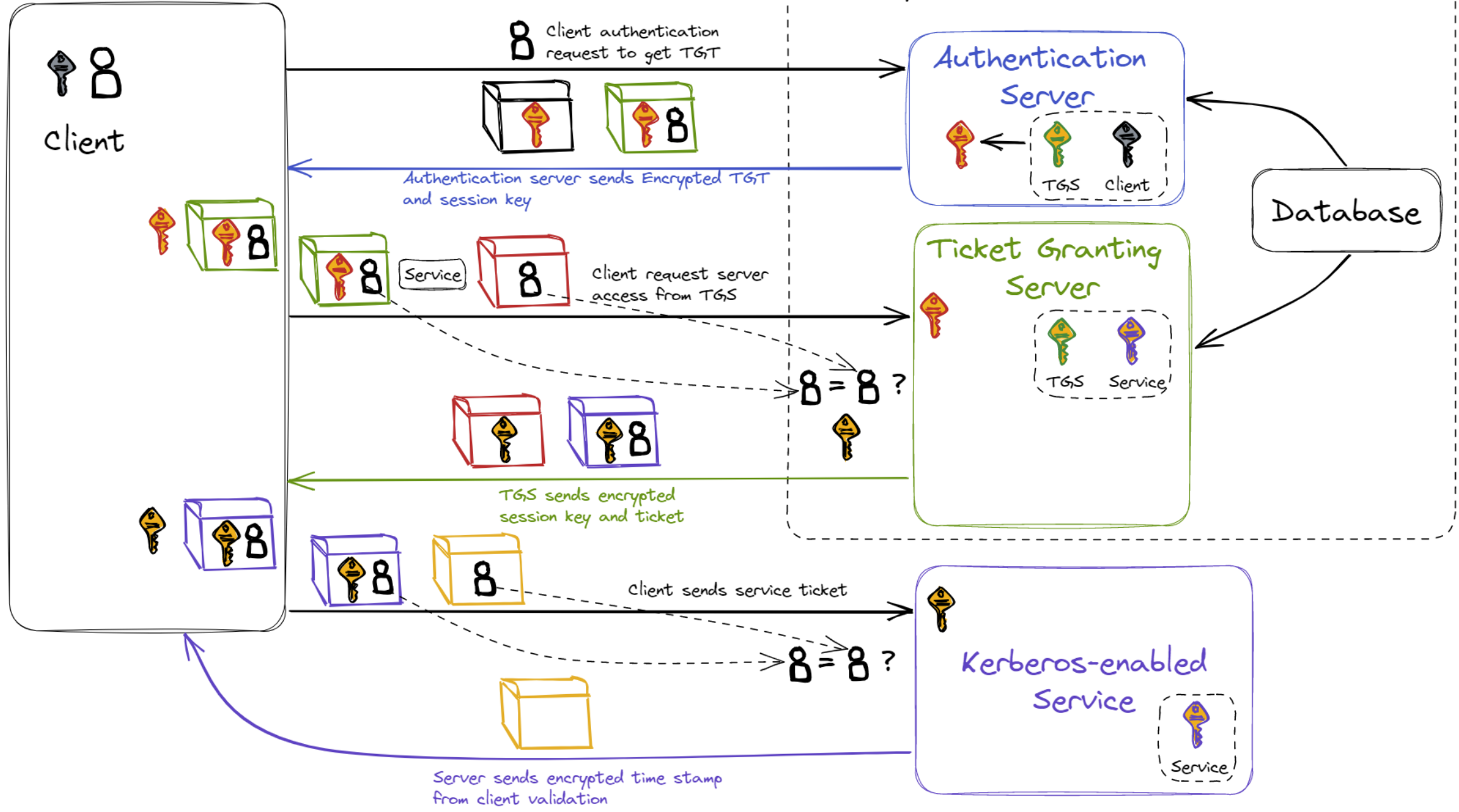
Podczas uwierzytelniania Kerberos przechowuje na urządzeniu użytkownika końcowego ticket dla każdej sesji. Zamiast hasła, usługa Kerberos szuka właśnie tego ticketu wymaganego do poświadczenia.

Uwierzytelnianie Kerberos odbywa się w środowisku, w którym KDC jest upoważniony do uwierzytelniania usługi, hosta lub użytkownika (Kerberos realm).

PROCES AUTENTYKACJI

1. Klient inicjuje potrzebę żądania usługi w imieniu użytkownika.
2. Serwer autentykujący (AS) przesyła nazwę użytkownika do centrum dystrybucji kluczy (KDC), gdzie przeprowadza uwierzytelnienie klienta.
3. KDC weryfikuje credentiale i jeśli uwierzytelnienie się powiedzie, KDC tworzy unikatowy identyfikator (TGT) uwierzytelniający użytkownika oraz szyfruje go za pomocą tajnego klucza TGS.
4. Ostatecznie zwraca zaszyfrowaną wartość do stacji użytkownika skąd możliwa jest autentykacja do serwera aplikacyjnego.

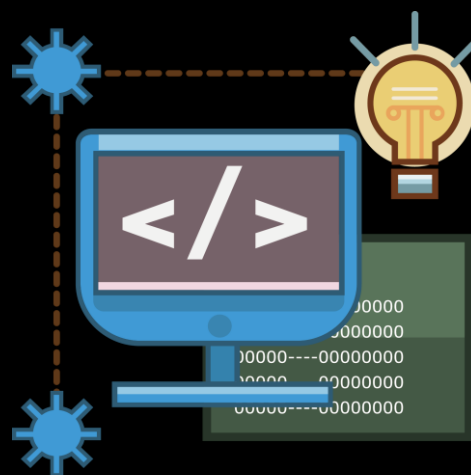
Kerberos Authentication Flow



IPA SERVER

- **FreeIPA jest zintegrowanym, scentralizowanym rozwiązaniem do zarządzania usługami:**
 - 389 Directory Server (LDAP)
 - MIT Kerberos (secure authentication)
 - NTP
 - DNS
 - Dogtag (certificate authority).
- **Serwer FreeIPA składa się z interfejsu webowego i narzędzi administracyjnych wiersza poleceń.**
- **FreeIPA jest zbudowana na bazie dobrze znanych komponentów Open Source i standardowych protokołów z bardzo dużym naciskiem na łatwość zarządzania i automatyzację zadań instalacyjnych i konfiguracyjnych.**

PROCESY I SYGNAŁY



PROCES

- Każdy program wykonujący zestaw instrukcji prowadzących do wykonania określonego zadania.
- Przykładem programu może być zarówno polecenie `mv`, jak i dużo bardziej skomplikowany skrypt powłoki Bash czy klient poczty Thunderbird.
- Procesem nazwiemy wykonywaną podczas działania systemu instancję programu wraz z dodatkowymi wykorzystywanymi przez niego zasobami.

ZOMBIE

Każdy proces posiada:

- PID (process ID) - liczba będąca identyfikatorem procesu
- PPID (parent process ID) - identyfikator swojego rodzica, czyli procesu, z którego został utworzony
- GPID (group process ID) - identyfikator grupy procesów

Pierwszym w przestrzeni użytkownika procesem w systemie jest init. Jest on uruchamiany przez jądro i nie posiada PPID. Jeśli któryś z rodziców zakończy się przed zakończeniem działania wszystkich procesów dzieci („osieroci” swoje dzieci) i zostaną one oznaczone jako procesy ZOMBIE.

Rolą init generalnie jest ich „adopcja”, czyli ustawienie ich PPID na 1.

STANY PROCESÓW

Procesy mogą mieć jeden z poniższych stanów:

- **Running:** aktualnie aktywny i wykorzystujący zasoby na systemie
- **Waiting/Sleeping:** czekający na informację z dysku lub innych źródeł zewnętrznych (zależnych)
- **Stopped:** zatrzymany lub wywołany kombinacją klawiszy Ctrl-Z
- **Zombie:** nieprawidłowo zamknięty przez proces nadrzędny

Daemon to proces działający w tle, którego jedynym celem jest dostarczenie pewnej specyficznej usługi dla użytkowników systemu. Zazwyczaj kończą swoją nazwę z (d) na końcu: sshd, httpd, chronyd itd.

LIMITY I PRIORYTETY

`ulimit`

- wbudowany w Basha
- plik konfiguracyjny: `/etc/security/limits.conf`
- pozwala na przeglądanie i ograniczanie procesów
 - soft limit: miękki limit jest zarządzany przez każdego użytkownika, a jego maksymalna wartość nie może przekroczyć twardego limitu
 - hard limit: twardy limit zasobów określa fizyczny limit zasobów dla użytkownika, hard limit jest maksymalną wartością dla soft limitu, tylko root może zmieniać hard limit

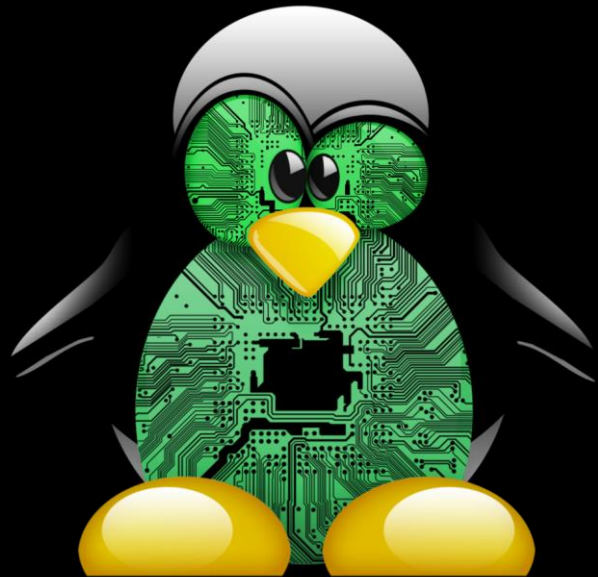
`nice/renice`

- pozwala zmieniać priorytety dla procesów
- odwrócona logika (mała wartość = większy priorytet i vice versa)
- `nice` - do uruchamiania procesów
- `renice` - do już działających procesów

SYGNAŁY

- Jeden z mechanizmów komunikacji międzyprocesowej, w skrócie nazywanych IPC (Inter-Process Communication).
- W rzeczywistości spora część sygnałów jest wygenerowana przez samo jądro w odpowiedzi na różne wyjątkowe sytuacje.
- Informacja niesiona przez sygnał może pochodzić spoza procesu.
- Działają asynchronicznie.
- W momencie wystąpienia sygnału, działanie programu jest przerywane i wykonywana jest procedura obsługi sygnału (programista może dostarczyć swoją własną funkcję obsługi sygnału). Po jej zakończeniu wykonanie programu jest kontynuowane od miejsca, w którym zostało przerwane.
- Podstawowych sygnałów z Linuxie jest 31.

KONFIGURACJA KERNELA I MODUŁY



SYSCTL

- Pozwala na odczyt i zapis zmiennych `sysctl`, czyli parametrów konfiguracyjnych jądra.
- Dzięki niemu można modyfikować parametry jądra systemu online.
- Ustawienia zapisane w: `/proc/sys/`
- Pliki konfiguracyjne:
 - `/etc/sysctl.d/`
 - `/etc/sysctl.conf`

MODUŁY

- Pełnią funkcję rozszerzenia jądra systemu.
- Mogą być ładowane lub usuwane dynamicznie, w zależności od potrzeb.
- Często zawierają sterowniki urządzeń, protokoły sieciowe itp.
- Znajdują się w katalogu `/lib/modules<wersja_kernels>`
- Skompilowane zazwyczaj dla konkretnych wersji jądra
- Mają rozszerzenie `*.ko` (Kernel Object)
- `modprobe.d` - folder konfiguracyjny dla `modprobe`

JOURNAL



JOURNALD

- Zintegrowany z Systemd mechanizm zarządzania logami na systemie.
- Posiada ustrukturyzowane logi o określonym formacie, które są z reguły dużo bardziej czytelne.
- Będąc częścią systemu inicjalizującego, startuje niemal równo z systemem i loguje dużo informacji z wczesnego etapu bootowania.
- Zarządzany komandą `journalctl`.
- Plik konfiguracyjny: `/etc/systemd/journald.conf`

PROS/CONS

Plusy:

- wczesny start i możliwość logowania
- znacząco szybsze i dużo łatwiejsze wyszukiwanie informacji
- sam potrafi dbać o to, ile miejsca logi będą zajmowały na dysku
- posiada mechanizmy mające przeciwdziałać uszkodzeniu logów (sprawdzanie stanu dziennika i w przypadku wykrycia błędu tworzenie nowego pliku;
- kompatybilny z syslogiem
- możliwość forwardingu logów

Minusy:

- ścisłe powiązanie z Systemd, starsze dystrybucje niekompatybilne
- binarny format, który sprawia, że wiele potężnych narzędzi administracyjnych, nie działa
- binarny format jest mniej odporny na uszkodzenie danych
- brak oficjalnego standardu - dwie różne wersje Journald potencjalnie mogą nie być w stanie odczytać pliku dziennika

LOGI SYSTEMOWE



PLIKI LOGÓW

- `/var/log/messages`: ogólne logi systemowe
- `/var/log/auth.log`: logi związane z autentykacją i logowaniem
- `/var/log/secure`: logi związane z autentykacją i logowaniem
- `/var/log/kern.log`: logi jądra
- `/var/log/cron.log`: logi crona
- `/var/log/maillog`: logi mailowego agenta
- `/var/log/apt/`: logi APTa
- `/var/log/dnf.log`: logi DNFa
- `/var/log/boot.log`: logi z bootowania systemu
- `/var/log/utmp` lub `/var/log/wtmp`: logi logowań na systemie

LOGROTATE

- Mechanizm automatycznej rotacji logów, który kompresuje i usuwa stare pliki logów.

```
<global directive 1>  
<global directive 2>
```

- Prosta konfiguracja per usługa

```
<file path matchers 1> {  
    <directive 1>  
    <directive 2>  
    ...  
    <directive n>  
}
```

- Plik konfiguracyjny: `/etc/logrotate.conf`
 - lub katalog: `/etc/logrotate.d/`

- Możliwość uruchomienia z opcją dry-run (-d): `$ logrotate -d <conf_file>`

- Wymuszenie rotacji z opcją force (-f): `$ logrotate -f <conf_file>`

FORWARD LOGÓW

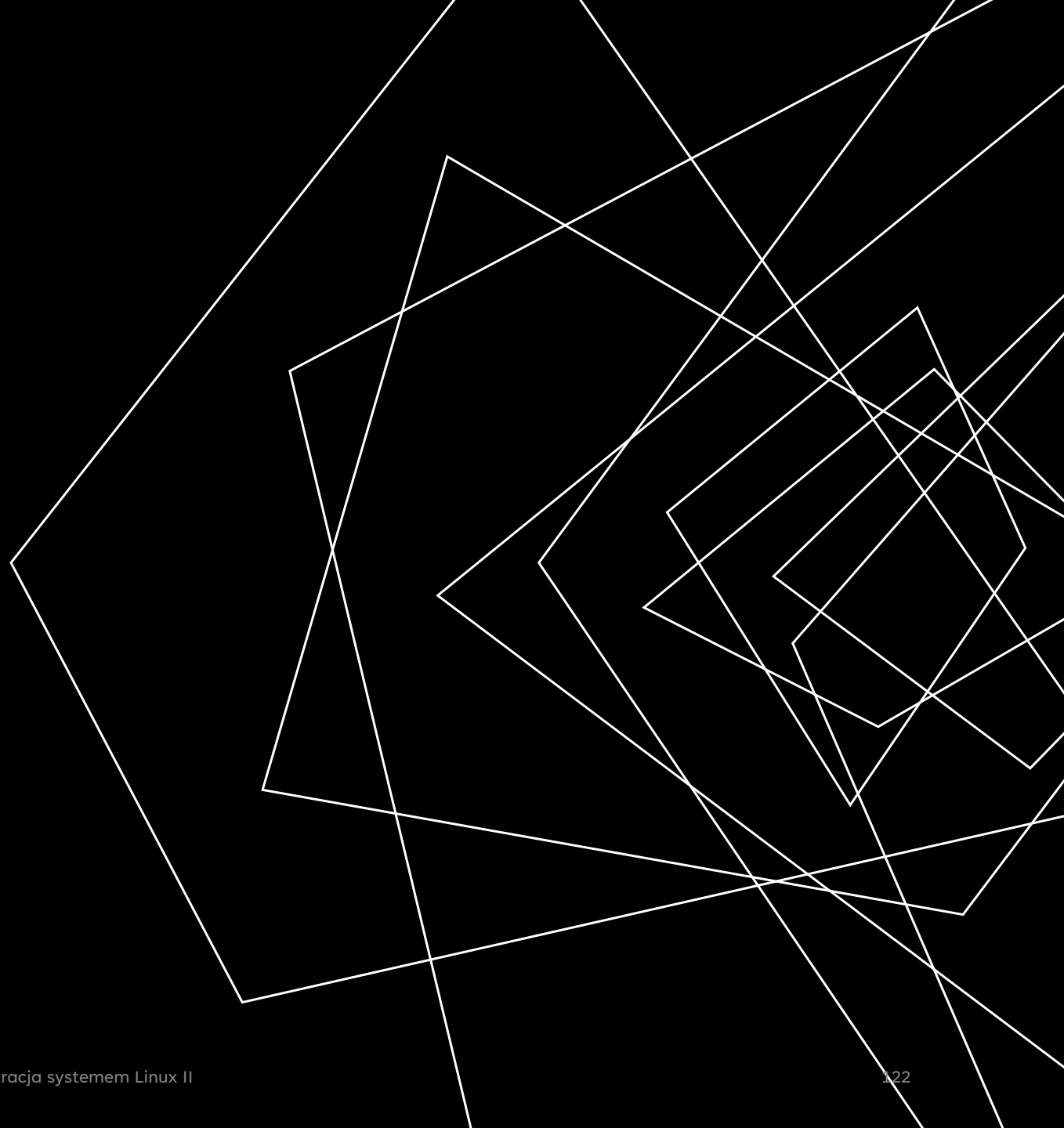
TCP:

- Oznaczenie po stronie klienta „@@”
- Protokół, który sprawdza i waliduje przesłane pakiety

UDP:

- Oznaczenie po stronie klienta „@”
- Brak sprawdzania poprawności wysłanych danych
- Używany kiedy logi są mało ważne lub kiedy potrzeba szybkiej transmisji

KONFIGURACJA SIECI



NETWORK MANAGER

- Myślenie połączeniami
- Jest częścią Systemd
- Domyślny dla nowszych systemów Linux (usługa domyślnie włączona)
- Posiada swój commandline `nmcli`
- Lepsze narzędzie niż `ip` oraz `ifconfig`

NETWORK TROUBLESHOOTING

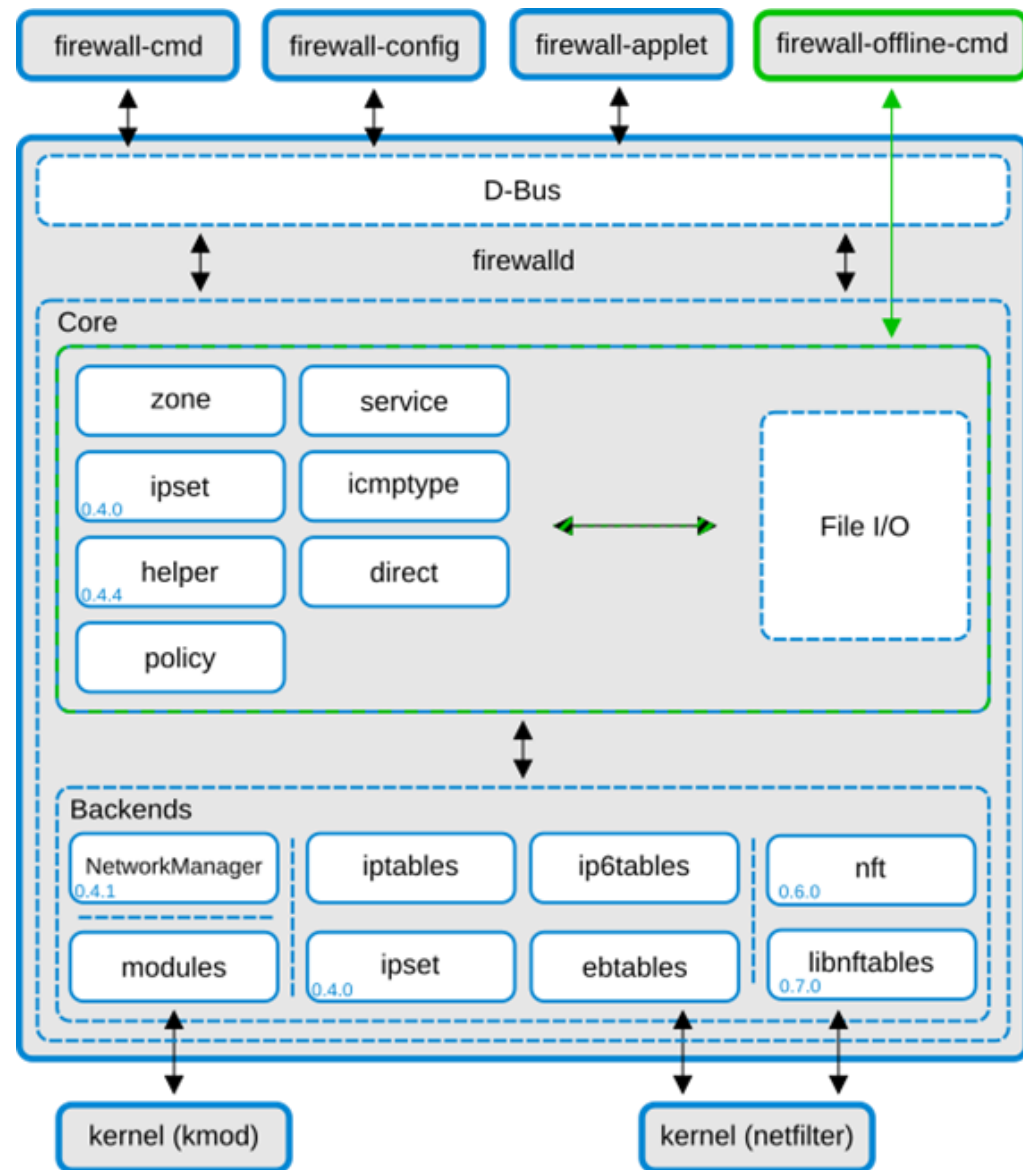
- nslookup
- ping
- traceroute
- dig
- host
- iptraf
- tcpdump
- wireshark
- netstat
- /etc/hosts
- /etc/nsswitch.conf
- /etc/resolv.conf

FIREWALLD I IPTABLES



FIREWALLD

- Alternatywa dla `iptables`.
- Prostszy w konfiguracji zapory.
- Posiada predefiniowane obiekty: `zones`, `services` itd.
- Domyślny firewall w systemach CentOS/RHEL.
- Backend wykorzystuje `nftables` (zamiennik lub nowsza wersja `iptables`).
- Wszystko trzymane w jednej tablicy.



IPTABLES

- Jest częścią pakietu `iptables-services`.
- Działa w oparciu bezpośrednio z jądrem systemu.
- Konfliktuje z `firewalld`, tak więc usługa musi być wyłączona, aby móc korzystać z `iptables`.
- Potężne narzędzie używane do zarządzania ruchem sieciowym poprzez filtrowanie pakietów.

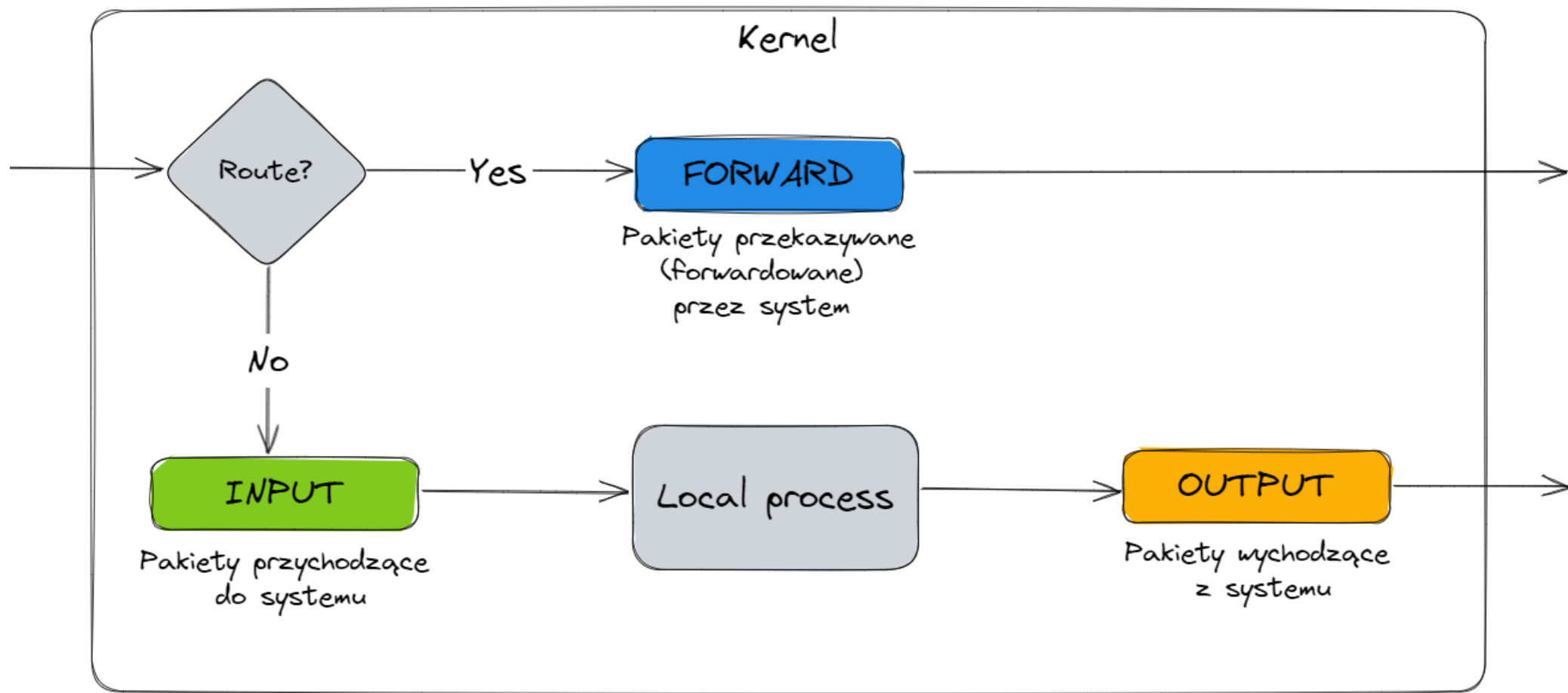
IPTABLES - ZASADA DZIAŁANIA

Technika filtrowania pakietów jest podzielona na trzy rodzaje struktur:

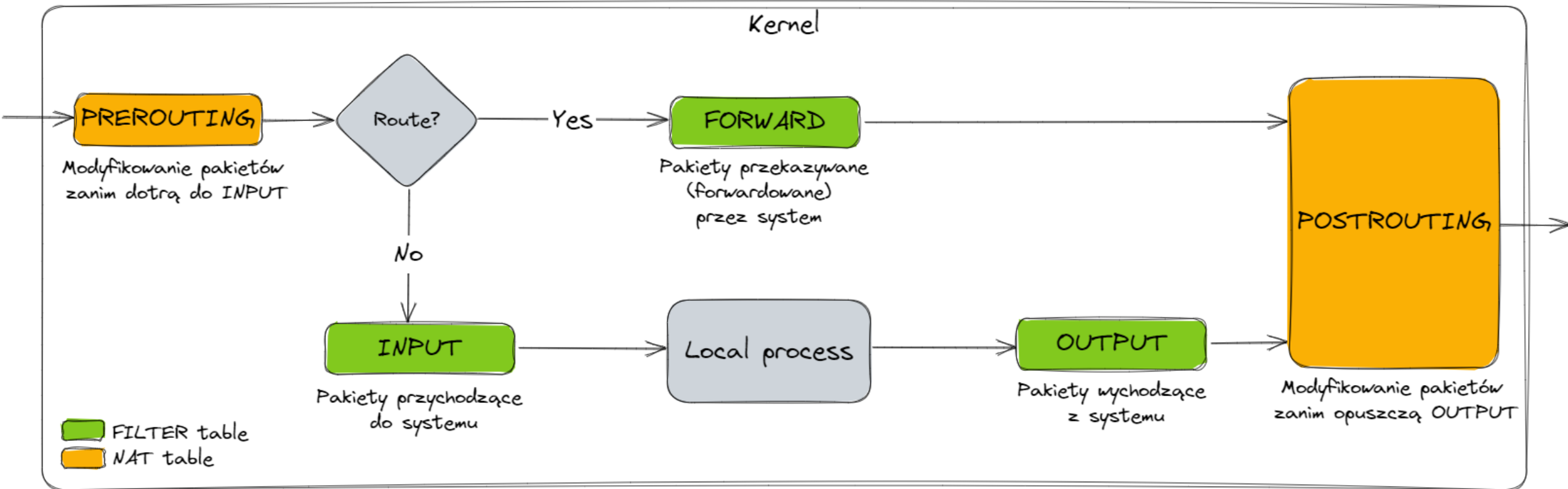
- tablice (tables)
- łańcuchy (chains)
- cele (targets)

Tablica to coś, co pozwala na obsługę pakietów w określony sposób. Tabele połączone są ze sobą łańcuchami. Łańcuchy te umożliwiają inspekcję ruchu na różnych etapach, np. kiedy dociera on do interfejsu sieciowego (INPUT) lub zanim zostanie wysłany na zewnątrz (OUTPUT). Konfiguracja polega na tym, aby prawidłowo dopasować określone pakiety z łańcuchem oraz powiązać je z celem. Cel określa, czy pakiet jest dozwolony czy odrzucony.

IPTABLES - FILTER TABLE



IPTABLES - NAT TABLE



IPTABLES - TARGETY

Iptables oferuje 3 możliwe targety (cele):

- **ACCEPT** - pozwala na nawiązanie połączenia,
- **DROP** - blokuje połączenie, nie wchodząc w żaden sposób w interakcję ze źródłem,
- **REJECT** - blokuje próbę połączenia, ale również wysyła komunikat o błędzie (odrzuconiu pakietu). Zwykle ma to na celu powiadomienie źródła, że próba połączenia została zablokowana przez zaporę.

IPTABLES – RULES

```
iptables [-t table] [mode] [chain] [rulenum] [rule-specification] [options]
```

Table:

- nat
- filter

Mode:

- APPEND / -A
(bez numeru ruli)
- INSERT / -I
(wymagany numer ruli)

Rule-specification:

- -p icmp -j ACCEPT
- -m state --state ESTABLISHED -j ACCEPT
- -d 192.168.1.100 -j REJECT

Options:

- -v (verbose)
- -n (numeric output)

NETWORK FILE SYSTEM



NFS

- **Network File System (NFS).**
- **System plików klient/serwer.**
- **Wieloplatformowy i rozproszony protokół systemu plików udostępniający zasoby przez sieć.**
- **Obsługiwane wersje NFS to NFSv3 i NFSv4, a domyślna wersja NFS to 4.2.**

KONFIGURACJA

- `/etc/nfs.conf` - główny plik konfiguracyjny dla deamona NFS
- `/etc/nfsmount.conf` - plik NFS odpowiedzialny za montowanie
- `/etc/exports` - plik definiujący udostępniane zasoby

Używane usługi/porty:

- `nfs` - 2049
- `rpc-bind` - 111
- `mountd` - 20048
- `idmapd` - dokonuje mapowania nazw dla NFS (`user@domain`) na identyfikatory użytkowników i grup, wymagany dla NFSv4

/ETC/EXPORTS

`<resource>` `<server> (options)` `<network> (options)`

- **Resource:** zasób na dysku, który chcemy udostępnić
- **Server:** adres IP serwera, któremu zasób będzie udostępniony
- **Network:** adres sieci, której zasób będzie udostępniony
- **Options:** opjce udostępniania

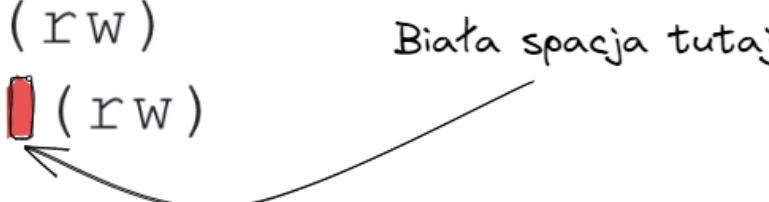
/ETC/EXPORTS

- **rw/ro**: umożliwia odczyt i zapis lub tylko odczyt (odpowiednio) dla plików
- **sync**: mówi serwerowi NFS, by wykonywał operacje zapisu (zapisywanie informacji na dysku), gdy jest to wymagane (domyślnie włączona)
- **secure**: używana do zdefiniowania zabezpieczeń (np. keberos):
 - **sec=krb5:krb5i:krb5p** (authentication only:integrity protection:privacy protection)
- **all_squash** - mapuje wszystkie UID i GID z żądań klienta na anonimowego użytkownika
- **no_all_squash** - mapuje wszystkie UID i GID z klienta na identyczne UID i GID na serwerze
- **root_squash** - mapuje żądania od użytkownika root lub UID/GID 0 od strony klienta na anonimowy UID/GID po stronie serwera
- **no_root_squash** - mapuje żądania od użytkownika root po stronie klienta na użytkownika root na serwerze NFS (security!!!) - nie powinno się jej używać

NFS TYPO

```
/nfs-share 192.168.1.81(rw)
/nfs-share 192.168.1.81 (rw)
```

Biała spacja tutaj



- Linia 1: zapis oznacza, że host 192.168.1.81 ma prawa do zapisu i odczytu dla /nfs-share
- Linia 2: zapis oznacza, że host 192.168.1.81 ma prawa **tylko do odczytu** (default) oraz **wszyscy inni**, którzy mają prawa do **zapisu i odczytu**

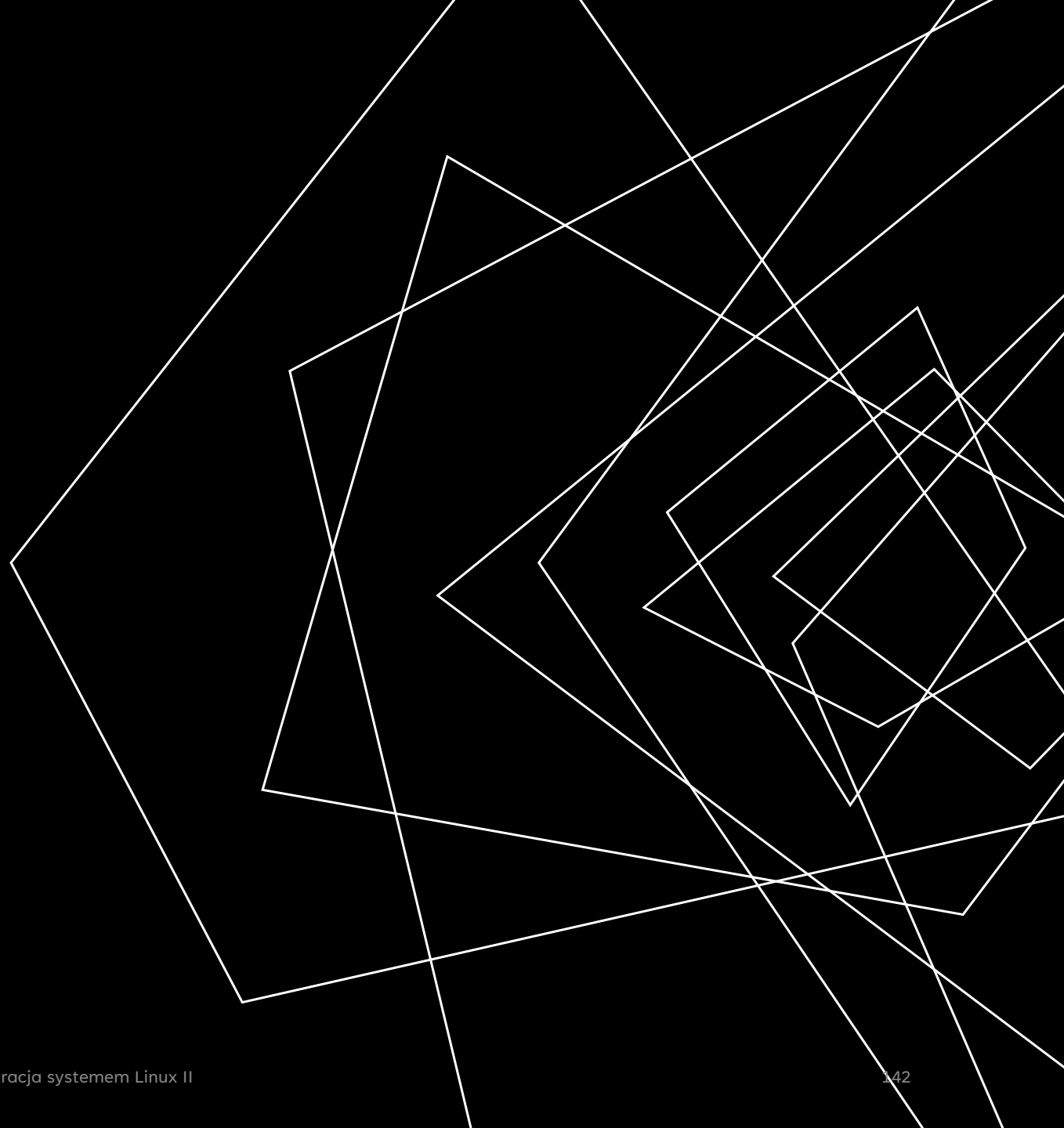
SAMBA



SAMBA

- Samba to open-source'owy i najpopularniejszy serwer plików i drukarek.
- Umożliwia użytkownikom końcowym dostęp do współdzielonych zasobów między Linuxem a Windowsem.
- Ma implementację protokołów z Windowsa, czyli SMB (Server Message Block) i CIFS (Common Internet File System).
- Samba jest również nazywana sieciowym systemem plików.
- Protokół klient/serwer.

SECURITY ENHANCED LINUX



SELINUX

- SELinux to zaawansowany mechanizm kontroli dostępu, wbudowany w większość dystrybucji Linuksa.
- Opracowany przez amerykańską Agencję Bezpieczeństwa Narodowego (NSA) w celu lepszej ochrony systemów operacyjnych.
- SELinux jest domyślnie skonfigurowany i aktywny dla dystrybucji Centos8 i nowszych.
- Definiuje on zakres działań użytkownika i/lub procesów, ograniczając je do ich własnych zakresów, dzięki czemu proces/użytkownik może współdziałać tylko z określonymi innymi procesami/typami plików.
- **SELinux znacząco zwiększa bezpieczeństwo serwera !!!**

POLITYKA

- Głównym mechanizmem bezpieczeństwa SELinux jest polityka, czyli zestaw zasad określających bezpieczeństwo i prawa dostępu dla poszczególnych elementów systemu - użytkowników, ról, procesów i plików.
- Polityka definiuje, w jaki sposób każdy z tych elementów jest ze sobą powiązany i mówi do czego można lub nie można uzyskać dostępu.
- Obiektem w SELinux jest wszystko, co może być przedmiotem działania. Może to być plik, katalog, socket, port itp.

TRYBY PRACY

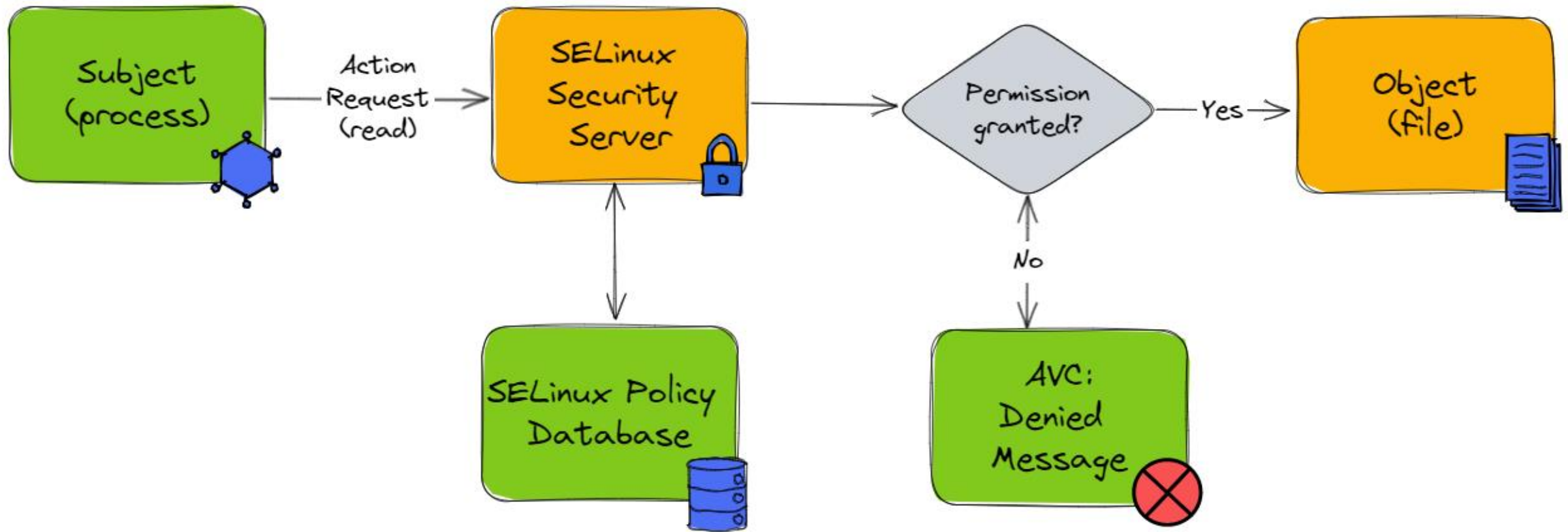
SELinux może znajdować się w jednym z trzech możliwych trybów:

- **Enforcing (wymuszony)** - egzekwuje swoją politykę i upewnia się, że wszelkie próby nieautoryzowanego dostępu przez użytkowników i procesy są odrzucane. Odmowy dostępu są zapisywane w odpowiednich plikach dziennika.
- **Permissive (pobłażliwy)** - jest stanem półaktywnym. SELinux nie stosuje swojej polityki w tym trybie, więc dostęp nie jest odrzucany, jednak wszelkie naruszenia są rejestrowane dzienniku audytu.
- **Disabled** - wyłączony.

DOSTĘP

- Kiedy aplikacja lub proces, znany jako 'subject', wysyła żądanie dostępu do obiektu, takiego jak plik, SELinux sprawdza za pomocą wektorowej pamięci podręcznej (AVC), gdzie uprawnienia podmiotów i obiektów są cache'owane.
- Jeśli SELinux nie jest w stanie podjąć decyzji o dostępie na podstawie uprawnień z pamięci podręcznej, wysyła żądanie do serwera bezpieczeństwa.
- Serwer bezpieczeństwa sprawdza kontekst aplikacji lub procesu lub pliku. Kontekst bezpieczeństwa jest przypisany z bazy danych polityki SELinuxa.
- Pozwolenie jest wówczas udzielane lub odmawiane. Jeśli dostęp jest odmówiony, wówczas mamy komunikat "avc: denied" w logach /var/log/messages.

DOSTĘP



KONTEKSTY

- SELinux opiera swoje zasady na kontekście typu. Nazwy kontekstów typu zwykle kończą się na t.
 - kontekstem typu dla serwera WWW jest 'httpd',
 - dla plików i katalogów znajdujących się w /tmp i /var/tmp jest 'tmp_t',
 - dla plików i katalogów znajdujących się w /var/www/html jest 'httpd_sys_content_t',
 - kontekst typu dla portów serwera WWW to 'http_port_t'.
- Z kontekstem SELinuxa możemy wchodzić w interakcję za pomocą opcji '-Z':
 - ls -Z, ps -Z itd.
- Zmiany kontekstu dokonujemy za pomocą polecenia `chcon` (zmiana tymczasowa) lub `semanage` (zmiana w bazie - permanentna)

system_u:object_r:httpd_sys_content_t:s0

Użytkownik SELinux Rola Typ Poziom

BOOLEANS

- SELinux boolean – to pojedynczy ciąg znaków, który zmienia reakcje SELinux, innymi słowy pozwalają na włączenie lub wyłączenie pewnych zasad w polityce.
- Włączenie/wyłączenie booleanów jest wygodne, gdyż użytkownik nie musi znać zasad pisania polityki SELinux.
- Przykładowe zastosowanie to np. umożliwienie usługom dostępu do woluminów NFS, bez przeładowywania lub rekompilacji zasad SELinux.
- Polecenie `getsebool -a` lub `semanage boolean` wyświetla aktualne statusy boolean'ów.

A series of white, thin, overlapping geometric lines and polygons on a black background, located on the left side of the slide.

THANK YOU

Marcin Kujawski

<https://marcinkujawski.pl/>