Wrocław
University
of Science
and Technology

Administrowanie sieciowymi systemami operacyjnymi

Wykład 4

NAT i system plików

dr inż. Jarosław Rudy

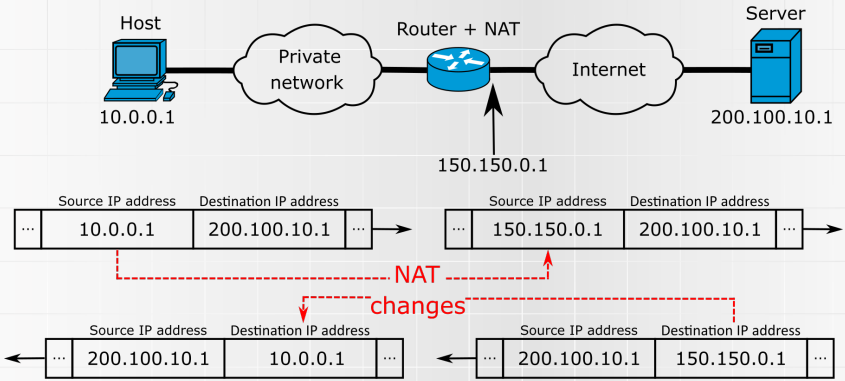
7 kwietnia 2021

HR EXCELLENCE IN RESEARCH

Network Address Translation (NAT)

- Zamiana źródłowych lub docelowych adresów IP przez router (bramę).
- One-to-one NAT.
- Stosowane głównie w IPv4.
- Pozwala na mapowanie adresu „całej" sieci (publicznego) na adresy lokalne hostów (prywatne).
- Umożliwia komunikację pomiędzy różnymi sieciami przy wyczerpującej się puli adresów IP.
- Możliwość ukrycia puli adresów prywatnych za pojedynczym adresem publicznym (maskarada IP, obecnie pojęcie używane zamiennie z NAT).
- Używane też do równoważenia obciążeń (dostęp do farmy serwerów).
- Konieczność korekcji sum kontrolnych (problemy z TCP):

Network Address Translation (NAT)



Źródło: [1]

Network Address Translation (NAT)

- Połączenia (gniazdka) w TCP/IP identyfikowane są przez czwórkę: 1) źródłowe IP, 2) docelowe IP, 3) źródłowy port, 4) docelowy port.
- Wiele komputerów lokalnych może wysyłać żądanie strony WWW.
- Najczęściej adres IP routera, port lokalny usługi oraz adres IP i port zdalny (np. `www.google.com:80`) są ustalone, to translacja wprost może być niejednoznaczna.
- Zachodzi potrzeba translacji numerów portów.
  - Network Address and Port Translation (NAPT).
  - Many-to-one NAT.
  - Port Address Translation (PAT).
  - NAT overload.
  - IP masquerading.

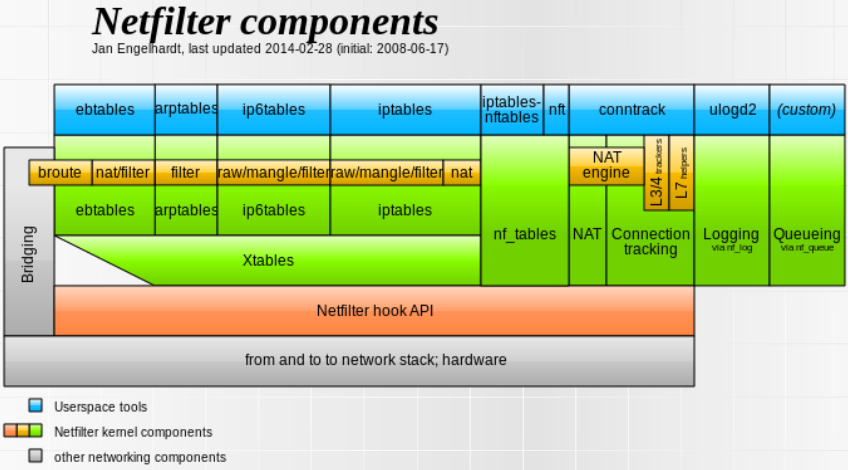
Firewall

- Filtracja pakietów.
- Najczęściej na podstawie portów TCP/UDP, ale nie tylko.
- Blokowanie i otwieranie portów, kontrola ruchu sieciowego.
- Generacje firewalli:
  - Filtry pakietów.
  - Śledzenie połączeń.
  - Warstwa aplikacji.

Netfilter

- Netfilter to zestaw narzędzi (framework) umożliwiający:
  - filtrację pakietów,
  - modyfikację pakietów (w tym NAT),
  - routing pakietów.
- Moduły jądra:
  - „Haczyki” (hook) umożliwiające dostęp do pakietów,
  - `iptables`, `ip6tables`, `arptables`, `ebtables`,
  - `nf_tables`
  - inne.
- Programy przestrzenii użytkownika (komendy):
  - `iptables`, `ip6tables`, `arptables`, `ebtables`,
  - `nft`
  - inne.

Netfilter



Źródło: [2]

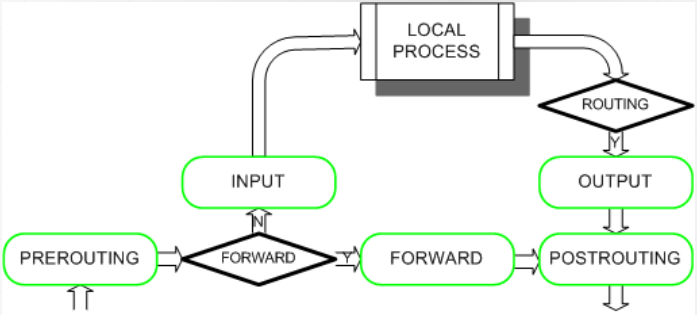
iptables

- Następca `ipchain`, poprzednik `nftables`/`nft`.
- Osobne komendy/moduły dla osobnych protokołów:
  - `ebtables` – Ethernet,
  - `arptables` – ARP,
  - `iptables` – IPv4,
  - `ip6tables` – IPv6.
- Dużo możliwości.
- Zależność od protokołu.
- Problem duplikacja kodu.

iptables

- Najważniejsze pojęcia: łańcuchy (chains), tabele, reguły (rules), cele (targets), polityki (policies).
- Pięć domyślnych łańcuchów:
  - `PREROUTING` – tuż po otrzymaniu pakietu od interfejsu,
  - `POSTROUTING` – tuż przed wysłaniem pakietu do interfejsu,
  - `FORWARD` – podczas routingu (przekierowania),
  - `INPUT` – tuż przed dostarczeniem pakietu do lokalnego procesu,
  - `OUTPUT` – tuż po stworzeniu pakietu przez lokalny proces.
- Łańcuch `BROUTING` dla `ebtables` (przed `prerouting`).
- Możliwość tworzenia własnych łańcuchów (podobnie do wywołania funkcji).

iptables



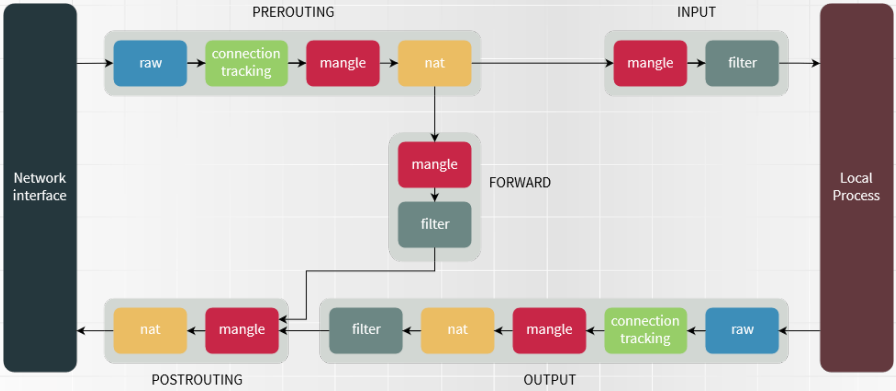
Źródło: [3]

iptables

Reguły grupowane są w tabele. Najczęściej wspierane są następujące tabele:

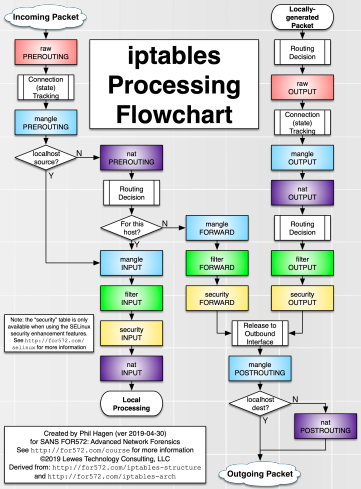
- `filter` – domyślna tabela, używana do decyzji czy pakiet może dotrzeć do celu czy powinien być odrzucony,
- `mangle` – ogólna modyfikacja pakietu (pole TTL itp.),
- `nat` – realizacja NAT (zamiana adresów źródłowych i docelowych),
- `raw` – dostęp do surowego pakietu przed kontrolą stanu połączenia lub wyłączenie kontroli.
- `security` – specyficzna dla `SELinux`, służy do implementacji kontekstu bezpieczeństwa.
- `broute` – dla `ebtables`.

iptables



Źródło: [4]

iptables



Źródło: [5]

iptables

- Reguła składa się z kryteriów (warunków) oraz tzw. celu.
- Kryteria mogą dotyczyć:
  - adresów IP,
  - numerów portów,
  - interfejsu,
  - TTL,
  - inne.
- Reguły sprawdzane są jedna po drugiej (kolejność ma znaczenie!) aż do znalezienia dopasowania (match), po czym stosowany jest cel („akcja”).
- przetwarzanie pakietu kończy się w przypadku napotkaniu celu terminalnego (akceptacja lub odrzucenie pakietu).

iptables

Przykładowe cele:

- `ACCEPT` – akceptacja pakietu, terminalny,
- `DROP` – ciche odrzucenie pakietu, terminalny,
- `REJECT` – odrzucenie pakietu z poinformowaniem nadawcy (reset dla TCP, nieosiągalny host dla UDP i ICMP), terminalny,
- `RETURN` – powrót do wywołującego łańcucha,
- `LOG` – zapisanie w logach systemowych,
- `MASQUERADE`, `REDIRECT`, `SNAT`, `DNAT` – wykonanie NAT,
- `TTL` – zmiana pola TTL pakietu,
- skok do zadanego łańcucha.

Łańcuchy mają domyślne polityki (typowo `RETURN` dla łańcuchów własnych oraz `ACCEPT` dla łańcuchów predefiniowanych).

iptables

- Komenda `iptables` – wyświetlanie i modyfikacja reguł.
- Reguły składają się z warunków, potem celu (`-j`) i parametrów celu, przypomina to proces warunkowego wywołania funkcji:

  ```
  if WARUNEK then jump/call CEL PARAMETRY
  ```

- Wyświetlenie reguł (w tym domyślnej polityki) dla zadanej tabeli `TABELA` (domyślnie `filter`) i zadanego `ŁAŃCUCHA` (domyślnie wszystkie):

  ```
  iptables -t TABELA -L ŁAŃCUCH
  ```

- Usunięcie wszystkich reguł dla `ŁAŃCUCH`:

  ```
  iptables -t TABELA -F ŁAŃCUCH
  ```

- Zmiana polityki łańcucha wbudowanego (tylko na `ACCEPT` lub `DROP`):

  ```
  iptables -t TABELA -P ŁAŃCUCH POLITYKA
  ```

iptables

- Dodanie reguły `REGUŁA` na koniec tabeli `TABELA` dla łańcucha `ŁAŃCUCH`:

  ```
  iptables -t TABELA -A ŁAŃCUCH REGUŁA
  ```

- Usunięcie reguły:

  ```
  iptables -t TABELA -D ŁAŃCUCH REGUŁA
  ```

- Usunięcie reguły nr `NUM` (licząc od 1):

  ```
  iptables -t TABELA -D ŁAŃCUCH NUM
  ```

- Wstawienie reguły jako numer `NUM` (domyślnie 1):

  ```
  iptables -t TABELA -I ŁAŃCUCH NUM REGUŁA
  ```

- Zastąpienie reguły numer `NUM`:

  ```
  iptables -t TABELA -R ŁAŃCUCH NUM REGUŁA
  ```

iptables

Przykłady użycia iptables

- Blokowanie ruchu z danego IP

```
iptables -A INPUT -s 10.10.10.10 -j DROP
```

- Blokowanie ruchu z całej podsieci

```
iptables -A INPUT -s 10.10.10.0/24 -j DROP
```

```
iptables -A INPUT -s 10.10.10.0/255.255.255.0 -j DROP
```

- Blokowanie połączeń SSH

```
iptables -A INPUT -p tcp -m tcp --dport ssh -s 10.10.10.10 -j DROP
```

```
iptables -A INPUT -p tcp -m tcp --dport ssh -j DROP
```

iptables

- Połączenia SSH nawiązywane tylko z zewnątrz

```
iptables -A INPUT -p tcp -m tcp --dport ssh -s 10.10.10.10
-m state --state NEW,ESTABLISHED -j ACCEPT
```

```
iptables -A OUTPUT -p tcp -m tcp --sport 22 -d 10.10.10.10
-m state --state ESTABLISHED -j ACCEPT
```

- Blokowanie pakietów ICMP typu echo request (typ 8).

```
iptables -A INPUT -p icmp -m icmp --icmp-type 8 -j DROP
```

- Negacja warunku.

```
iptables -A INPUT -p tcp -m multiport ! --dports 22,80,443 -j DROP
```

iptables

- Określenie interfejsów wejściowych i wyjściowych

```
iptables -A INPUT -i lo -j ACCEPT
```

```
iptables -A OUTPUT -o wlan0 -d 121.18.238.0/29 -j DROP
```

- Limitowanie liczby pakietów

```
iptables -A INPUT -p icmp -m limit --limit 1/sec
--limit-burst 1 -j ACCEPT
```

- Zapis pakietu do logów

```
iptables -A INPUT -p tcp -m tcp --tcp-flags FIN,SYN FIN,SYN
-j LOG --log-prefix=iptables:
```

iptables

- Zmiana odbiorcy/nadawcy

```
iptables -t nat -A POSTROUTING -j SNAT --to-source 123.123.123.123
```

```
iptables -t nat -A PREROUTING -j DNAT --to-destination 123.123.123.123:22
```

- Maskarada IP

```
iptables -t nat -A PREROUTING -j POSTROUTING
```

- Przekierowanie do lokalnego portu

```
iptables -t nat -A PREROUTING -j REDIRECT --to-ports 8080
```

- Zmniejszenie pola TTL

```
iptables -t mangle -A PREROUTING -j TTL --ttl-dec 1
```

nft i nftables

- Następca dla `iptables`.
- Bardziej elastyczny.
- Ujednolicony (brak konieczności osobnych komend dla poszczególnych protokołów).
- Unika duplikacji kodu.
- Nieco bardziej „rozwlekły” zapis:

```
nft add rule ip filter output ip daddr 1.2.3.4 drop
```

w porównaniu do `iptables`

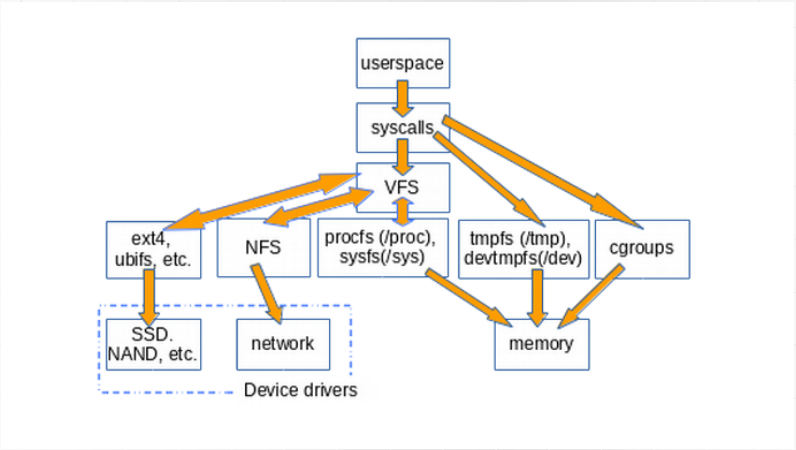
```
iptables -A OUTPUT -d 1.2.3.4 -j DROP
```

System plików

Niektóre elementy oprogramowania systemu plików (File System, FS)

- Wirtualny system plików (VFS)
  - ujednolicony interfejs dostępu do różnych konkretnych systemów plików, korzystają z niego funkcje systemowe,
  - mapuje wirtualne funkcje na operacje specyficzne dla danego FS (łatwe dodawanie nowych FS przez implementację `open`, `read`, `write` itd.),
  - część FS (np. `/dev`) omija VFS.
- Urządzenia blokowe – obsługują zapis i odczyt urządzeń fizycznych, pojedyncze odpowiada za jedną (wirtualną) partycję.
- Urządzenia znakowe – występują opcjonalnie, wykorzystywane do obsługi urządzeń (np. tworzenie partycji).

System plików



Źródło: [6]

System plików

Najważniejsze elementy uniksowego systemu plików

- Superblok – przechowuje ważne informacje ogólne (globalne metadane) o systemie plików.
- Tablica i-węzłów – zawiera i-węzły (i-nodes), które przechowują metadane poszczególnych plików. Zajmuje około 20-30% miejsca.
- Bloki danych – przechowują zawartość samych plików (zależnie od typu pliku, dla plików zwykłych są to dane użytkownika). Zajmują większość miejsca.

System plików

Superblok

- Przechowuje między innymi:
  - rozmiar FS, rozmiar bloku itd.,
  - status FS,
  - parametry techniczne,
  - umiejscowienie innych elementów (m.in. tablicy i-węzłów).
- Błędy w superbloku mogą uniemożliwić zamontowanie (a tym samym odczyt i zapis danych) systemu plików.
- FS przechowuje kopie superbloku.

System plików

- I-węzeł przechowuje metadane pliku
  - typ pliku,
  - właściciel, właściciel grupowy,
  - prawa dostępu,
  - rozmiar pliku (dla `ext4` limit 16 TiB),
  - czas ostatniego dostępu, modyfikacji i zmiany stanu i-węzła,
  - liczba (twardych) dowiązań,
  - położenie na dysku (tablica indeksowa na bloki danych, wskaźniki bezpośrednie i pośrednie).
- I-węzeł nie przechowuje nazwy ani zawartości pliku!
- Numery i-węzła unikalny tylko w obrębie urządzenia.

System plików

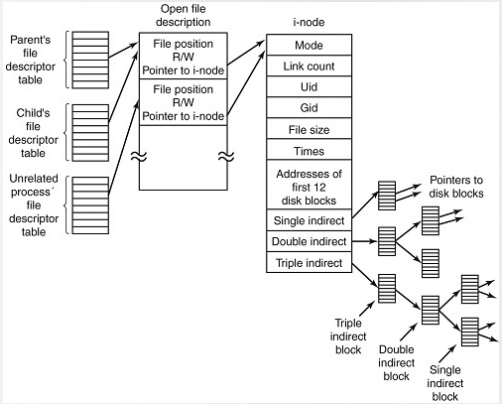
Typy plików

- Pliki zwykłe – przechowują dane użytkowników (pliki tekstowe, obrazki, muzyka, programy itp.)
- Katalogi:
  - zawierają wpisy katalogowe w postaci mapowania nazwa pliku → i-węzeł,
  - wpisy stanową dowiązania (tzw. dowiązania twarde),
  - wpisy `.` oraz `..` – „specjalne" dowiązania twarde tworzone automatycznie,
  - na ten sam i-węzeł (i tym samym plik fizyczny) może istnieć wiele dowiązań twardych (niekoniecznie w jednym katalogu. System może usunąć plik, dopiero gdy usuniemy wszystkie dowiązania (poza `.` i `..`),
  - katalog główny (korzeń) FS to katalog wskazany w superbloku. Jego katalogiem nadrzędnym jest on sam.

System plików

- Dowiązania symboliczne (tzw. miękkie, symlinki) – przechowują ścieżkę do innego pliku tzw. celu. Cel nie musi być na tym samym urządzeniu, ani w ogóle istnieć. Operacje na symlinku przenoszone są na cel.
- Urządzenia blokowe i znakowe.
- Potoki nazwane (FIFO) – komunikacja jednokierunkowa, plik tworzony komendą `mkfifo` lub `mknod`.
- Gniazdka UNIX-owe (Unix domain sockets) – lokalna komunikacja dwukierunkowa, zbliżona do komunikacji TCP/IP. „Adresem" jest plik.

System plików



Źródło: [7]

System plików

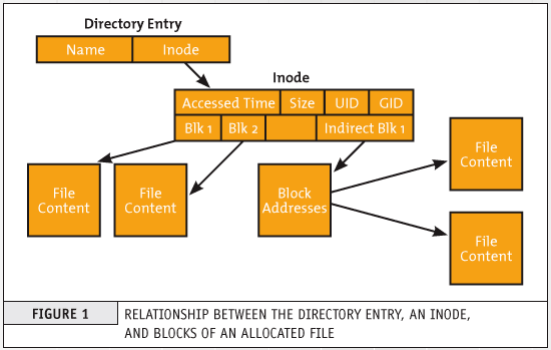


FIGURE 1 RELATIONSHIP BETWEEN THE DIRECTORY ENTRY, AN INODE, AND BLOCKS OF AN ALLOCATED FILE

Źródło: [7]

Dla rozmiaru bloku 4 kB i rozmiaru wskaźnika 4 bajty (16 TB miejsca):

- bezpośrednie wskazanie: około 50 kB,
- pośrednie 1-poziomu: około 4 MB,
- pośrednie 2-poziomu: około 4 GB,
- pośrednie 3-poziomu: około 4 TB.

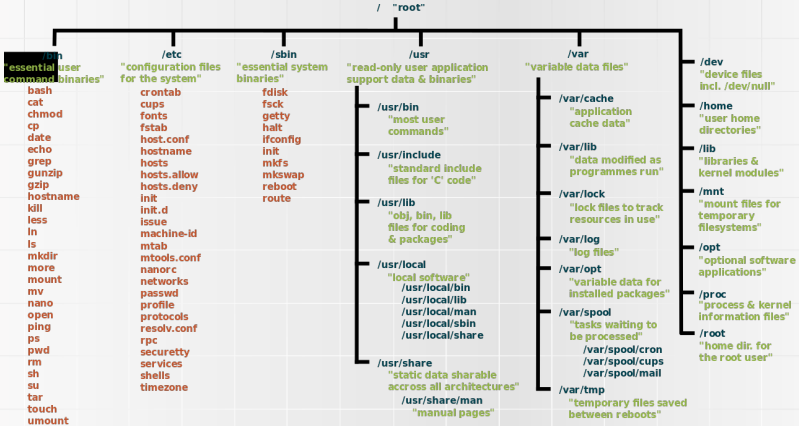
Interpretacja ścieżki

- wejście – ścieżka (elementy/komponenty oddzielone ukośnikiem),
- wyjście – odpowiedni i-węzeł lub informacja o nieistniejącym pliku.

1. Jeśli ścieżka `path` zaczyna się od korzenia (ścieżka bezwzględna[1]) to `wd = iget(root)`, w przeciwnym razie (ścieżka względna) `wd = iget(cwd)`.
2. Dopóki jest kolejny element `comp` w ścieżce `path`:
   - 2.1 Zweryfikuj że `wd` jest katalogiem i mamy odpowiednie prawa do `wd` (jeśli nie to błąd).
   - 2.2 Jeśli `comp == .. && wd == iget(root)` to `continue`.
   - 2.3 Przeczytaj wpisy katalogu `wd`.
     - 2.3.1 Jeśli `comp` jest wśród wpisów to `wd = iget(comp)`.
     - 2.3.2 W przeciwnym razie `return no_file`.
3. `return wd`.

[1] Uwaga na początkową tyldę!

System plików



Źródło: [8]

Dodatkowo: `/boot`, `/media`, `/mnt`, `/tmp`, `/srv`.

Instalowanie systemu plików

- System plików jest jednym drzewem z korzeniem w węźle `/`.
- Dostęp do systemów plików na innych urządzeniach realizowany jest w ramach tego samego drzwa – nowe FS dołączane są jako poddrzewa zaczepione w danym katalogu (węźle).
- Instalacja automatyczna systemów plików zdefiniowanych w:
  - BIOS-ie/UEFI,
  - programie bootującym (np. `grub`),
  - w skryptach startowych systemu operacyjnego.

Instalowanie systemu plików

- Instalacja ręczna z wykorzystaniem komendy `mount`. Podstawowa składnia

  `mount -t <type> -<options> <device> <dir>`

  - `type` – typ systemu plików (np. `ext4`),
  - `device` – zwykle ścieżka urządzenia blokowego w `/dev`, ale nie zawsze (np. NFS),
  - `dir` – ścieżka do katalogu służącego jako punkt montowania.
- Montowanie wymaga zainstalowanego odpowiedniego modułu dla danego FS.
- Montowanie przesłania zawartość katalogu przez zamontowany FS oraz właściciela/prawa katalogu przez roota/prawa roota.
- Wylistowanie zamontowanych FS poprzez `mount`, `mount -l`, podgląd pliku `/etc/mtab` lub komendę `findmnt`.
- Odmontowanie komendą `umount`.

Instalowanie systemu plików

- Type `auto` (`mount` zgadnie typ FS odpytując urządzenie).
- Wywołanie `mount -a` spróbuje zamontować wszystkie FS z pliku konfiguracyjnego `/etc/fstab`, z wyjątkiem tych z opcją `noauto`
- Opcje `user` i `users`.
- Jeśli do `mount` podano urządzenie bez katalogu lub katalog bez urządzenia, to `mount` spróbuje wywnioskować brakujący element z `/etc/fstab`.
- Tradycyjnie w Uniksie plik `/etc/mtab` zawierał zamontowane urządzenia, ale nie zawsze był to stan aktualny.
- W Linuksie `/etc/mtab` działa poprawnie ponieważ jest realizowany jako symlink do pliku `/proc/self/mounts`.

Instalowanie systemu plików

Przykładowa zawartość pliku `/etc/fstab`

```
# <file system> <mount point>   <type>  <options>    <dump>  <pass>
proc            /proc           proc    defaults     0       0
# / was on /dev/sda1 during installation
UUID=b4fd978d-d7c7-45a5-8588-6d3a39e576fc /  ext3  errors=remount-ro 0       1
# swap was on /dev/sda5 during installation
UUID=8e448e21-04a1-4100-97e2-151f1d7f0b85 none   swap  sw              0       0

/dev/scd0     /media/cdrom    udf,iso9660 user,noauto     0       0
/dev/fd0      /media/floppy   auto        rw,user,noauto  0       0
```

Instalowanie systemu plików

- UUID – Universally Unique IDentifier (RFC 4122).
  - liczba 128-bitowa w postaci szesnastkowej,
  - różne wersje (losowy, timestamp, adres MAC itd.),
  - prawdopodobieństwo kolizji niezerowe, ale zaniedbywalnie niskie.
- Może być użyty do identyfikacji FS (także w `/etc/fstab`.
- Przechowywany jest w superbloku.
- Problemy z wyszukiwaniem (komenda `ls -l /dev/disk/by-uuid/`).
- Komenda `blkid` – wydobywanie informacji (w tym etykieta i UUID) z urządzenia.
- Komenda `lsblk` – listowanie urządzeń. Z opcją `-fs` podaje etykiety i UUID-y.

Wirtualne systemy plików

System plików `procfs`

- montowany jako `/proc`,
- zawiera katalogi z informacjami dla poszczególnych procesów. Przykładowo, w katalogu `/proc/1234` można znaleźć pliki z informacjami o procesie z `PID=1234`.
- zawiera obraz niektórych struktur jądra (w tym dla części możliwy jest zapis). Przykładowo:
  - `/proc/mounts` – lista zamontowanych FS,
  - `/proc/filesystems` – lista znanych FS.
- pliki są „puste" – zwykle mają rozmiar 0 bajtów, zaś zawartość jest generowana na podstawie stanu systemu (snapshot) w momencie odczytu.

Wirtualne systemy plików

- `/proc/PID/status` – informacje podstawowe (stan, zużycie pamięci itp.),
- `/proc/PID/cwd` – katalog aktualny procesu,
- `/proc/PID/fd` – katalog z symlinkami do otwartych deskryptorów plików,
- `/proc/PID/mem` – binarny obraz pamięci wirtualnej procesu,
- `/proc/PID/environ` – zmienne środowiskowe (klucz-wartość) procesu,
- `/proc/PID/maps` – zmapowane obszary pamięci (pliki, stos, sterta itd.),
- `/proc/PID/root` – symlink do korzenia FS (czyli / z wyjątkiem zastosowania tzw. `chroot jail`),
- `/proc/cpuinfo` – informacje o procesorach,
- `/proc/sys` – konfiguracja dla `sysctl` (możliwość zmiany parametrów jądra na bieżąco).

Wirtualne systemy plików

System plików `sysfs`

- występuje w nowszych dystrybucjach Linuksa,
- montowany jako `/sys`,
- zawiera wartości (parametry) struktur jądra,
- służy zarówno do odczytu jak i zapisu (alternatywa do `sysctl`),
- częściowo pokrywa się z `/proc`,
- `/sys/module` – zainstalowane moduły,
- `/sys/dev` oraz `/sys/devices` – hierarchia urządzeń.

Udev i urządzenia wymienne

Narzędzie (system) `udev`.

- Służy do dynamicznego zarządzania urządzeniami (wpisami w `/dev`).
- Szczególnie przydatny z urządzeniami wymiennymi (hotplug/hotswap).
- Demon `udevd`, wychwytuje zdarzenia od jądra i komunikuje się z dalszymi demonami (np. `udisksd-daemon`, `NetworkManage`).
- Korzysta z `sysfs`
- System `udev` pracuje w przestrzeni użytkownika (user space)
  - zwiększa bezpieczeństwo,
  - zwiększa stabilność,
  - ułatwia nazewnictwo i obsługę niektórych urządzeń.

Udev i urządzenia wymienne

- Możliwość nadawania urządzeniom nazw statycznych – stałe nazwy pomiędzy rebootami i ich niezależność od kolejności podłączania/wykrywania urządzeń.
- Baza reguł konfiguracji `/etc/udev/rules.d/` w postaci warunek-akcja. Każda zmiana konfiguracji (hotplug) wywołuje pasujące akcje.
- `autofs/automount`
  - programy/demony do monitorowania i montowania FS na „śledzonych" punktach montowania.
  - Główna konfiguracja „szablonów" w `/etc/autofs/auto.master`

```
/media/misc  /etc/autofs/auto.misc
/media/net   /etc/autofs/auto.net   --timeout=60
```

Udev i urządzenia wymienne

- Montowanie dysku USB o danym UUID w `/mnt/flash`. Zawartość pliku `/etc/autofs/auto.master`:

  ```
  /mnt    /etc/autofs/auto.mnt
  ```

  następnie wykonanie komend

  ```
  _ID=$( blkid --output value --match-tag PARTUUID /dev/sdXY )
  printf "%s %s\n" "flash -fstype=auto :PARTUUID=" "${_ID}" > /etc/autofs/auto.mnt
  ```

  co zapisze do `/etc/autofs/auto.mnt` wpis postaci

  ```
  flash -fstype=auto :PARTUUID=XYZ
  ```

- przykład reguły `etc/udev/rules.d/99-togglemouse.rules`

  ```
  ACTION=="add" , ATTRS{idProduct}=="c52f" \
  , ATTRS{idVendor}=="046d" , ENV{DISPLAY}=":0" \
  , ENV{XAUTHORITY}="/run/user/1000/gdm/Xauthority" \
  , RUN+="/usr/bin/xinput --disable 16"
  ```

Pliki a sieć komputerowa

- Clustered file system („klastrowy" system plików) – system plików zamontowany jednocześnie na wielu hostach.
  - Shared-disk file system – blokowy dostęp do wspólnych udziałów dyskowych poprzez Storage Area Network (SAN). Przykłady: BWFS, GFPS, OCFS.
  - Distributed file system – dostęp poprzez zwykłe API systemu plików (w przeciwieństwie do Distributed Data Store). Przykłady: HDFS, DFS (Microsoft).
  - Network-Attached Storage (NAS) – dysk podłączony z pośrednictwem sieci.
- Sieciowy system plików – dowolny system plików korzystający z protokołów sieciowych takich jak NFS czy SMB (wszystkie powyższe).
- Często potrzeba korekcji błędów (fault-tolerant file system) lub wydajności/nadmiarowości (parallel file system).

NFS

- Network File System (NFS) – protokół dla rozproszonych systemów plików. Pozwala na dostęp do zdalnych plików tak jakby były to zwykłe pliki lokalne.
- Klient NFS jest w pakiecie `nfs-common` lub `nfs-utils`.
- Montowanie NFS jest zbliżone do standardowego z wyjątkiem typu i ścieżki podanej z uwzględnieniem adresu zdalnego hosta:

  ```
  mount -t nfs 12.34.56.78:/home /mnt/nfs/home
  ```

- Wymagany jest pakiet `rpcbind` zarówno do mapowania portów (wykrywania portów serwera NFS), jak i do realizacji blokad plików (dla NFS v2 i v3).
- Udziały montowane wielokrotnie (opcje `sharecache` i `nosharecache`).
- Blokowanie plików na serwerze (opcje `lock` i `nolock`).
- Problem z mapowaniem użytkowników (centralna baza typu LDAP lub korzystanie z NFS v4 i `idmapd`.

NFS

Serwer NFS – podstawowa konfiguracja.

- Pakiet `nfs-kernel-server`.
- Demony `nfsd`, `mountd`, `lockd`, `rquota.d`, `statd`.
- Wymaga zainstalowania pakietu/demona `portmap` (z wyjątkiem NFS v4).
- Umożliwienie nasłuchiwania poza localhostem

```
perl -pi -e 's/^OPTIONS/#OPTIONS/' /etc/default/portmap
echo "portmap: 192.168.1." >> /etc/hosts.allow
/etc/init.d/portmap restart
```

- Konfiguracja współdzielonych udziałów w `/etc/exports`. Format: `udział adres(opcje)`. Np.

```
/example 192.168.1.0/255.255.255.0(rw,no_root_squash,subtree_check)
```

NFS

Przykładowe opcje udziałów NFS

- `rw` (read-write), `ro` (read-only, domyślna).
- `root_squash` (domyślna), `no_root_squash`, `all_squash` – zmiana żądającego UID/GID na anonimowy.
- `anonuid`, `anongid` – UID i GID dla użytkownika anonimowego.
- `sync`, `async` – odpowiedź po/przed zaksięgowaniem operacji. „Brak" domyślnej wartości.
- `nohide`, `hide` (domyślna), `crossmnt` – traktowanie „podudziałów".
- `no_subtree_check`, `subtree_check` – sprawdzanie poprawności udziałów. „Brak" domyślnej wartości.

Bibliografia

- https://en.wikipedia.org/wiki/Network_address_translation
- https://en.wikipedia.org/wiki/Netfilter
- https://www.globo.tech/learning-center/linux-native-firewall-introduction-to-iptables/
- https://www.booleanworld.com/depth-guide-iptables-linux-firewall/
- https://stuffphilwrites.com/2014/09/iptables-processing-flowchart/
- https://opensource.com/article/19/3/virtual-filesystems-linux
- https://pc-freak.net/blog/find-filesystem-directory-eating-filesystem-inodes-linux/
- https://en.wikipedia.org/wiki/Unix_filesystem