



Systemy operacyjne Unix, Linux

Autor Wojciech Gumiński



Plan wykładu

Wprowadzenie

1. Pojęcie systemu operacyjnego
4. Wprowadzenie do SO Unix, Linux
5. Wprowadzenie do wyrażeń regularnych
6. Potoki i przekierowania
7. Archiwizacja i kompresja danych
8. Edycja tekstów
9. Praca w sieci
10. Zlecenie wykonywania poleceń w określonym czasie
11. Skrypty powłoki
12. Administracja systemem



Prowadzący

Wykłady

dr inż. Wojciech Gumiński

e-mail: Wojciech.Guminski @ pg.edu.pl

<http://guminski.net>

<http://kti.eti.pg.gda.pl/ktilab>

Konsultacje WETI p. 145, terminy w Moja PG

Laboratoria

dr inż. Krzysztof Cwalina

e-mail: Krzysztof.Cwalina @ pg.edu.pl

Konsultacje WETI p. 404, terminy w Moja PG



Literatura

Literatura

A. Silberschatz, P. B. Galvin; „*Podstawy Systemów Operacyjnych*”;

WNT 1998

<http://codex.cs.yale.edu/avi/os-book/OS9/index.html>

A. Tanenbaum, H. Boss; "*Systemy operacyjne*", Helion 2015



Zasady zaliczania

Laboratorium

- 7 ćwiczeń laboratoryjnych
- 3 sprawdziany w postaci testu zamkniętego 5 pyt. (3x5=15pkt.) na 3-cim, 5-tym i ostatnim laboratorium

Wykład

- Kolokwium zaliczające na ostatnim wykładzie (10 pkt.)

Ocena

- Zaliczenie sumą punktów z liniową skalą ocen tzn.
- >50% 3.0, >60% 3.5, >70% 4.0, >80% 4.5, >90 5.0
- Oceny pozytywne nie podlegają poprawie
- Punkty zdobyte na poprawce oceny negatywnej uśredniają się z dotychczas zdobytymi



Program laboratorium

- Wprowadzenie, operacje na plikach
- Strumienie, atrybuty plików, uprawnienia
- Przetwarzanie potokowe
- Archiwizacja
- Edytory i komunikacja
- Skrypty
- Środowisko graficzne i administracja systemu

Karta przedmiotu na stronie <http://ects.pg.edu.pl>



Program wykładu

- Pojęcie systemu operacyjnego
- Zadania i funkcje systemu operacyjnego
- Wprowadzenie do systemu Unix
- Atrybuty i uprawnienia
- Systemy plików
- Wyrażenia regularne
- Przetwarzanie potokowe
- Archiwizacja i kompresja danych
- Edycja i przetwarzanie tekstów
- Skrypty
- Komunikacja i praca w sieci
- Administracja systemem
- Systemy wbudowane i dedykowane



Laboratorium EA204

Konieczna rejestracja – dostępna wyłącznie z laboratorium:

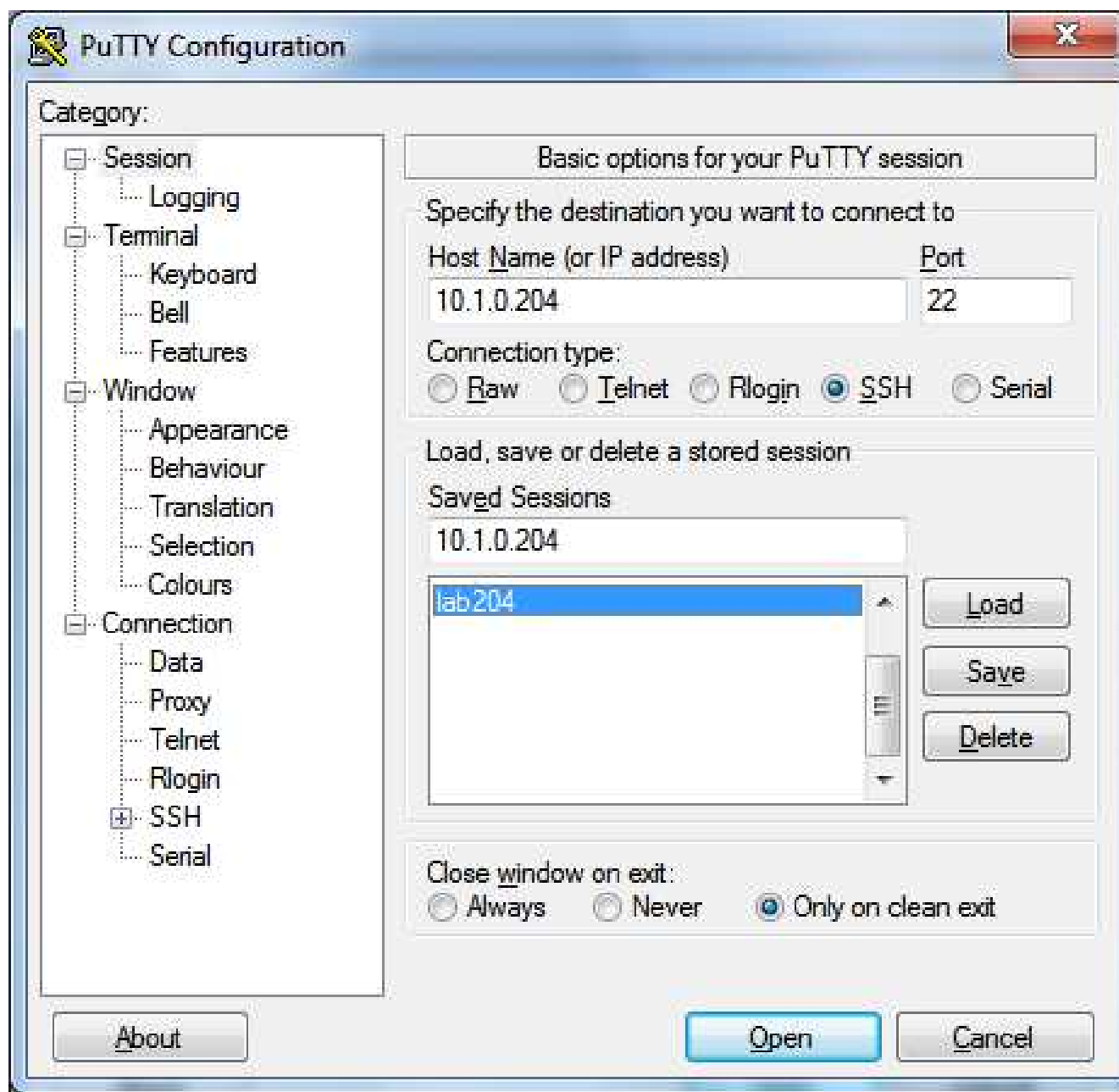
- <http://lab204/rejestracja.php>

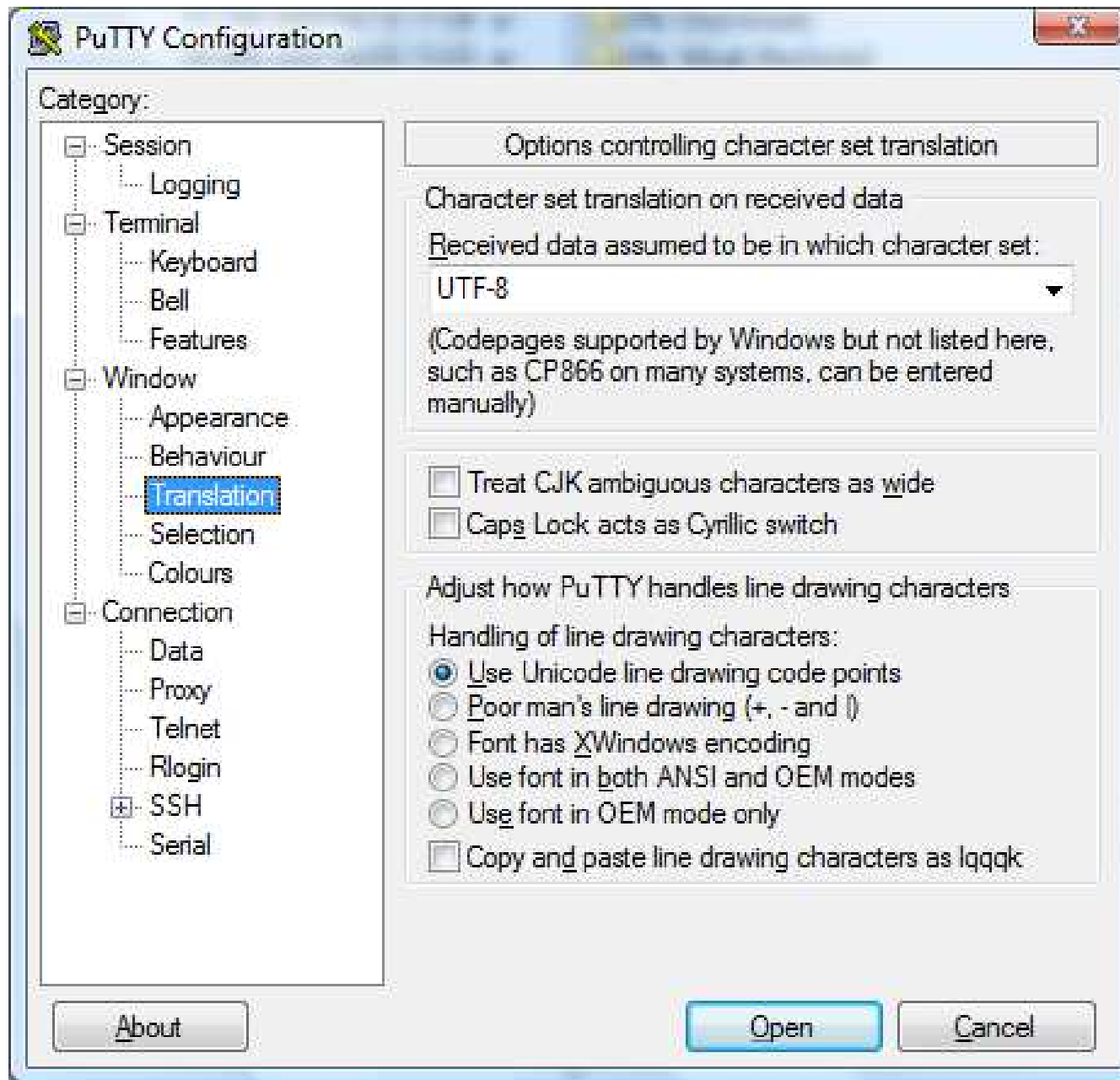
Połączenie SSH – wyłącznie z laboratorium

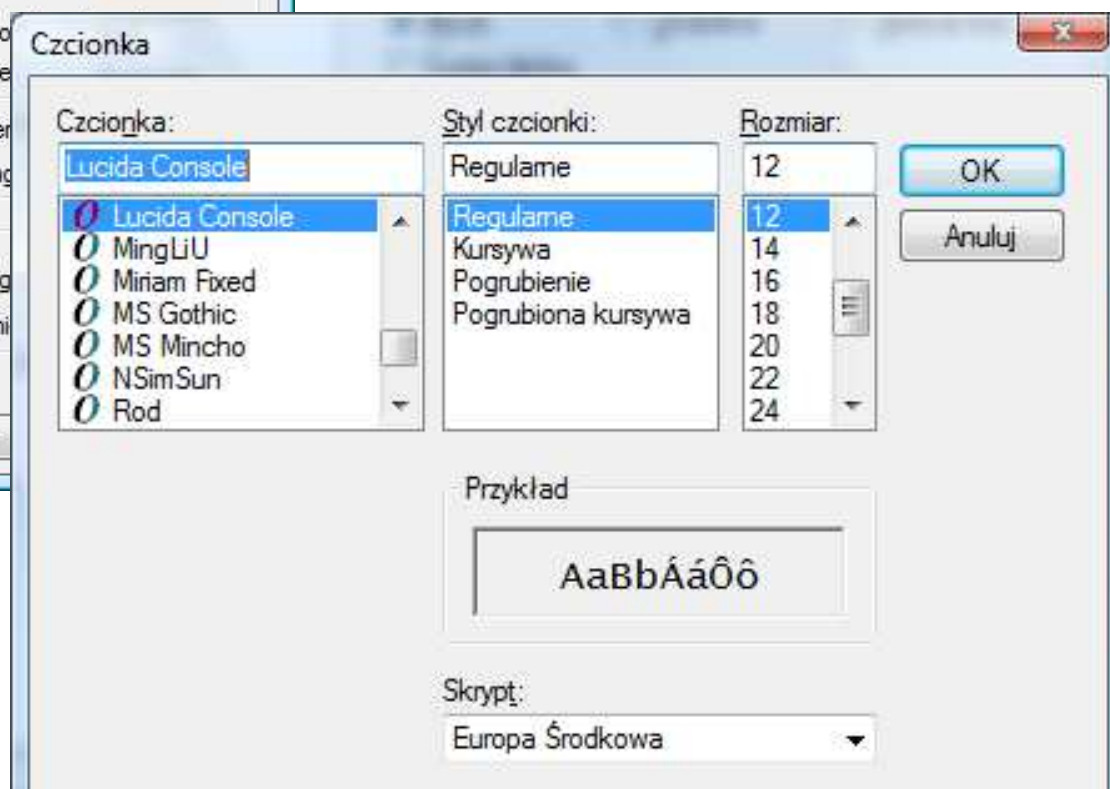
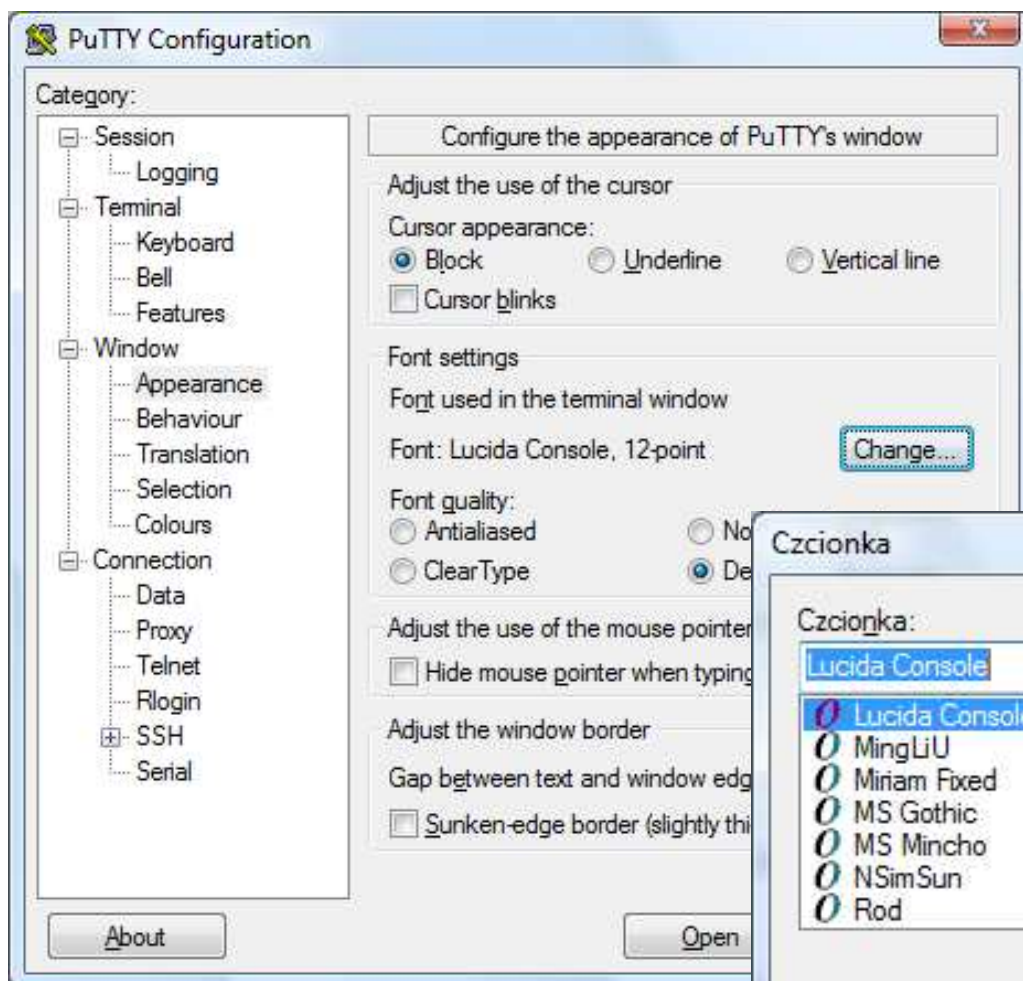
- `ssh s115115@lab204`

W systemie MS Windows program putty.exe

- Protokół: SSH
- Adres: lab204
- Port: 22
- Kodowanie znaków: UTF-8

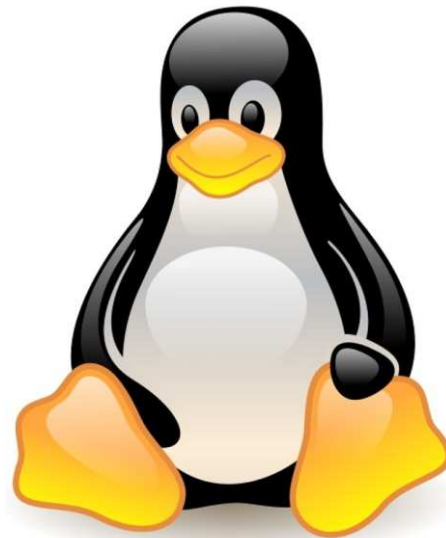








System operacyjny



symbian



Novell

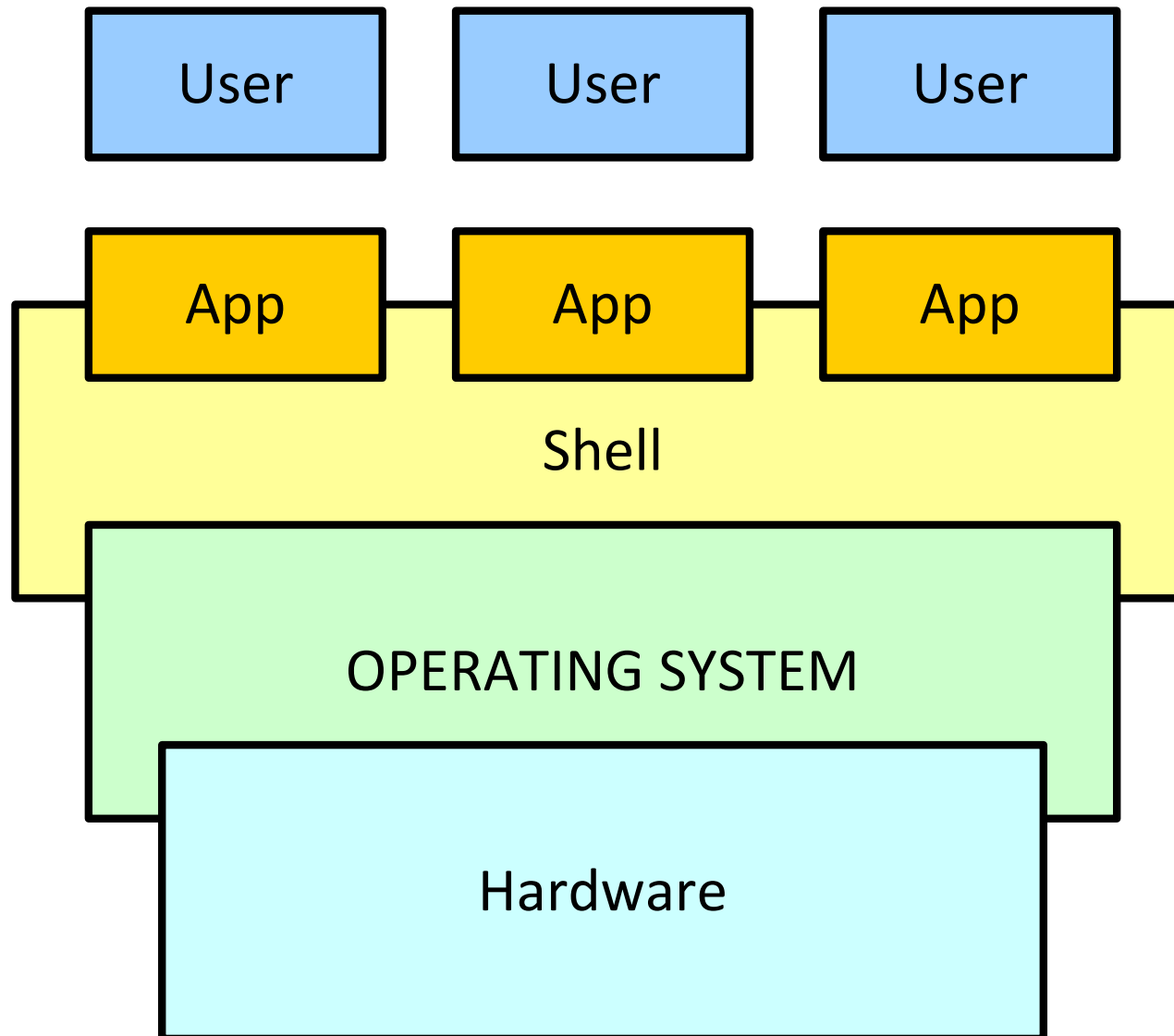


System operacyjny

Pojęcie systemu operacyjnego

System operacyjny jest programem, który działa jako pośrednik między użytkownikiem komputera a sprzętem komputerowym.

Podstawowym zadaniem systemu operacyjnego jest tworzenie środowiska, w którym użytkownik może wykonywać programy oraz sprawić, aby system komputerowy był wygodny w użyciu.





System operacyjny

Czym jest system operacyjny?

- środowisko uruchamiania programów
- dystrybutor zasobów (czas procesora, pamięć)
- program sterujący – programami użytkownika i systemowymi, oraz zarządzający dostępem do urządzeń wejścia-wyjścia.

Co jest zatem systemem operacyjnym?

- Tylko jądro (ang. kernel) działające w systemie komputerowym nieustannie;
- Czy wszystko to co dostarcza producent systemu operacyjnego?

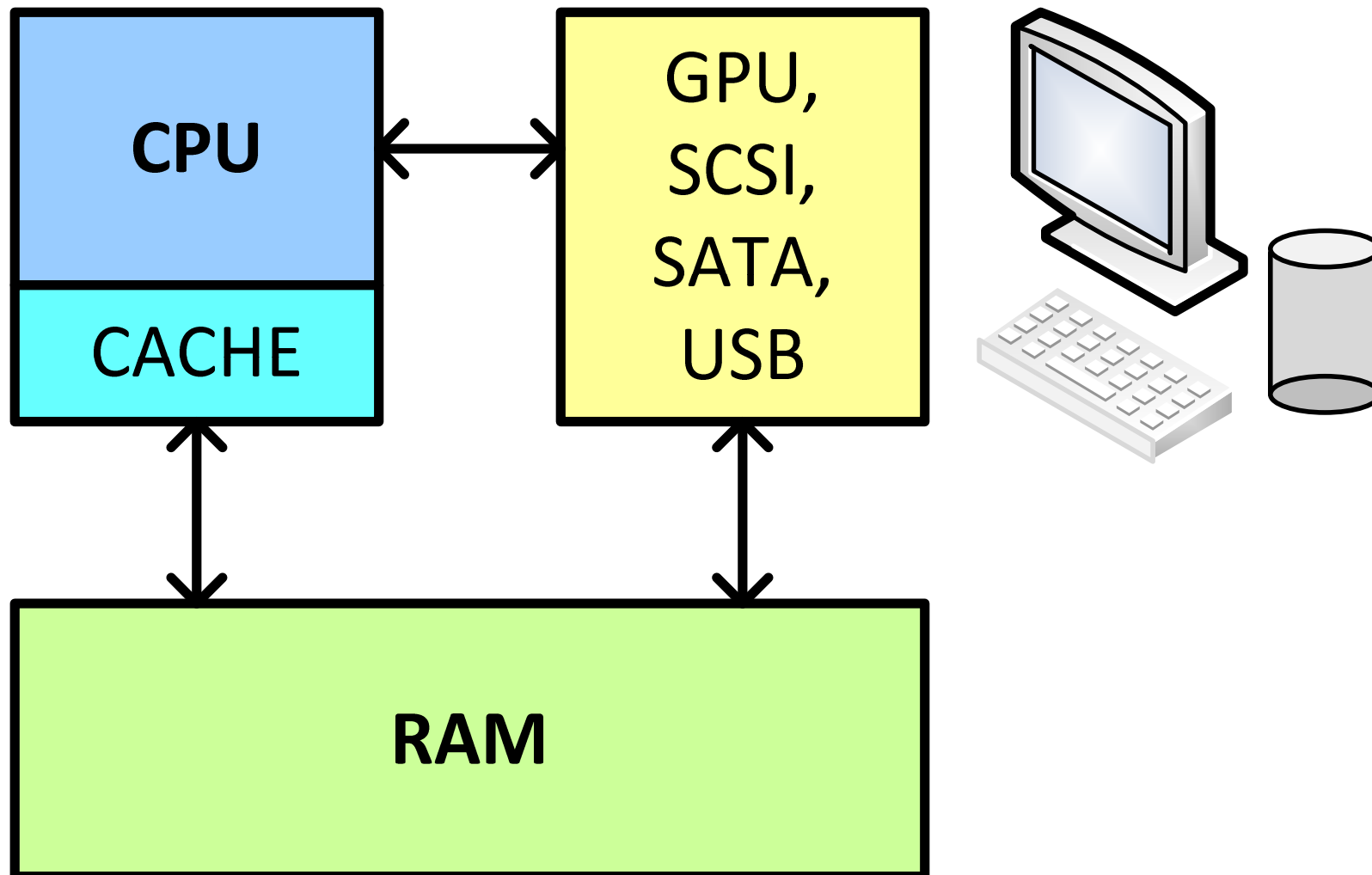


System operacyjny

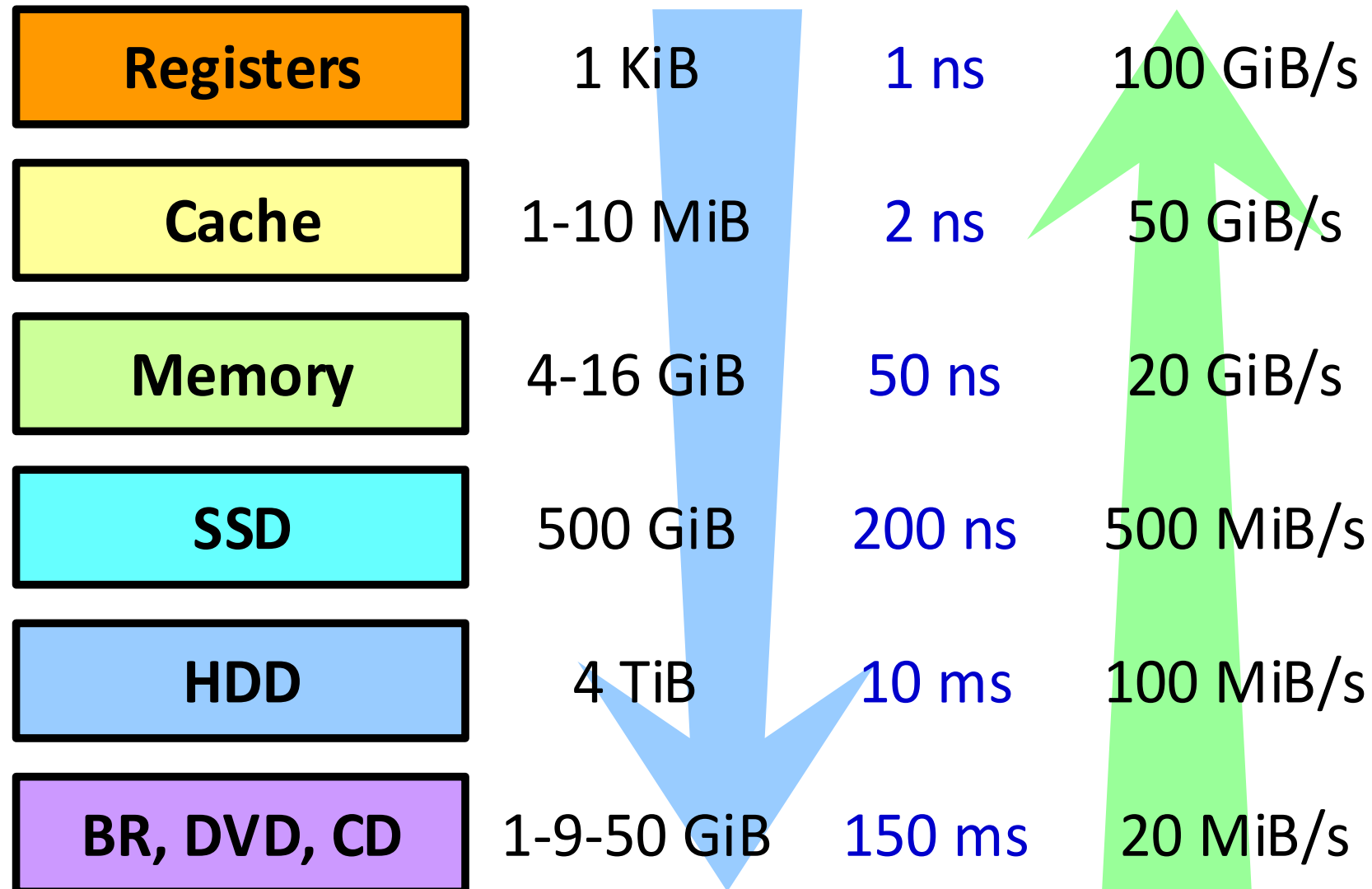
Historyczne spojrzenie na rodzaje systemów operacyjnych

- wsadowe
- wsadowe wieloprogramowe
- wielozadaniowe z podziałem czasu
- wielozadaniowe równoległe (wieloprocessorowe)
- rozproszone
 - podział zasobów
 - przyspieszenie obliczeń
 - niezawodność
 - komunikacja
- czasu rzeczywistego

System komputerowy wg architektury von Newmana



Porównanie pojemności i szybkości różnych pamięci



Registers	1 KiB	1 ns	100 GiB/s
Cache	1-10 MiB	2 ns	50 GiB/s
Memory	4-16 GiB	50 ns	20 GiB/s
SSD	500 GiB	200 ns	500 MiB/s
HDD	4 TiB	10 ms	100 MiB/s
BR, DVD, CD	1-9-50 GiB	150 ms	20 MiB/s

The diagram illustrates the memory hierarchy with capacity, access time, and transfer rate for different memory types. A large blue arrow points downwards from Registers to BR, DVD, CD, indicating increasing capacity and access time. A large green arrow points upwards from BR, DVD, CD to Registers, indicating increasing transfer rate and decreasing access time.



2. Podstawowe zadania systemu operacyjnego

2.1. Zarządzanie procesami

- tworzenie i usuwanie procesów użytkowych i systemowych
- wstrzymywanie i wznowianie procesów
- dostarczanie mechanizmów synchronizacji procesów
- dostarczanie mechanizmów komunikacji procesów
- dostarczanie mechanizmów obsługi zakleszczeń

2.2. Zarządzanie pamięcią operacyjną

- ewidencjonowanie zajętych obszarów pamięci z informacją o właścicielu
- przydzielanie i zwalnianie obszarów pamięci
- decydowanie o tym, które procesy mają być załadowane do zwalnianych obszarów pamięci



2.3. Zarządzanie plikami

- tworzenie i usuwanie plików
- tworzenie i usuwanie katalogów
- dostarczanie elementarnych operacji do manipulowania plikami i katalogami
- odwzorowanie plików na obszary pamięci pomocniczej
- składowanie plików na trwałych nośnikach

2.4. Zarządzanie pamięcią pomocniczą

- zarządzanie wolnymi obszarami
- przydzielanie pamięci
- planowanie przydziału obszarów pamięci dyskowej



2.5. Zarządzanie systemem wejścia-wyjścia

W SO UNIX podsystem wejścia-wyjścia ukrywający przed systemem operacyjnym osobliwości urządzeń we-wy.

Podsystem ten zawiera:

- moduły sterujące urządzeniami sprzętowymi
- interfejs modułów sterujących
- część zarządzającą pamięcią w tym: buforowanie, spooling i pamięć podręczna



2.6. Praca sieciowa

- współpraca procesów rozproszonych, które nie dzielą wspólnej pamięci, procesora ani zegara.

2.7. System ochrony

- ochrona plików, pamięci, procesora i innych zasobów

2.8. System interpretacji poleceń

- interpretacja poleceń wydawanych przez użytkownika - shell



3. Usługi systemu operacyjnego

- wykonanie programu
- operacje wejścia-wyjścia
- manipulowanie systemem plików
- komunikacja
- wykrywanie błędów
- przydzielanie zasobów
- rozliczanie
- ochrona



3.1. Funkcje systemowe

Funkcje systemowe albo wywołania systemowe (ang. *system calls*) tworzą interfejs pomiędzy wykonywanym programem a systemem operacyjnym.

- Nadzorowanie procesów
 - Zakończenie, zaniechanie
 - Załadowanie, wykonanie
 - Utworzenie procesu, zakończenie procesu
 - Pobranie atrybutów procesu, określenie atrybutów procesu
 - Czekanie czasowe
 - Oczekiwanie na zdarzenie, sygnalizacja zdarzenia
 - Przydział i zwolnienie pamięci

Operacje na plikach

- Utworzenie pliku, usunięcie pliku
- Otwarcie, zamknięcie
- Czytanie, pisanie, zmiana położenia
- Pobranie atrybutów pliku, określenie atrybutów pliku

Operacje na urządzeniach

- Zamówienie urządzenia, zwolnienie urządzenia
- Czytanie, pisanie, zmiana położenia
- Pobranie atrybutów urządzenia, określenie atrybutów urządzenia
- Logiczne przyłączania lub odłączanie urządzeń

Utrzymywanie informacji

- Pobranie czasu lub daty, określenie czasu lub daty
- Pobranie danych systemowych, określenie danych systemowych
- Pobranie atrybutów procesu, pliku lub urządzenia

Komunikacja

- Utworzenie, usunięcie połączenia komunikacyjnego
- Nadawanie, odbieranie komunikatów
- Przekazanie informacji o stanie
- Przyłączanie lub odłączanie urządzeń zdalnych



3.2. Programy systemowe

- Manipulowanie plikami
- Informowanie o stanie systemu
- Tworzenie i zmienianie zawartości plików
- Translatory języków programowania
- Ładowanie i wykonywanie programów
- Komunikacja

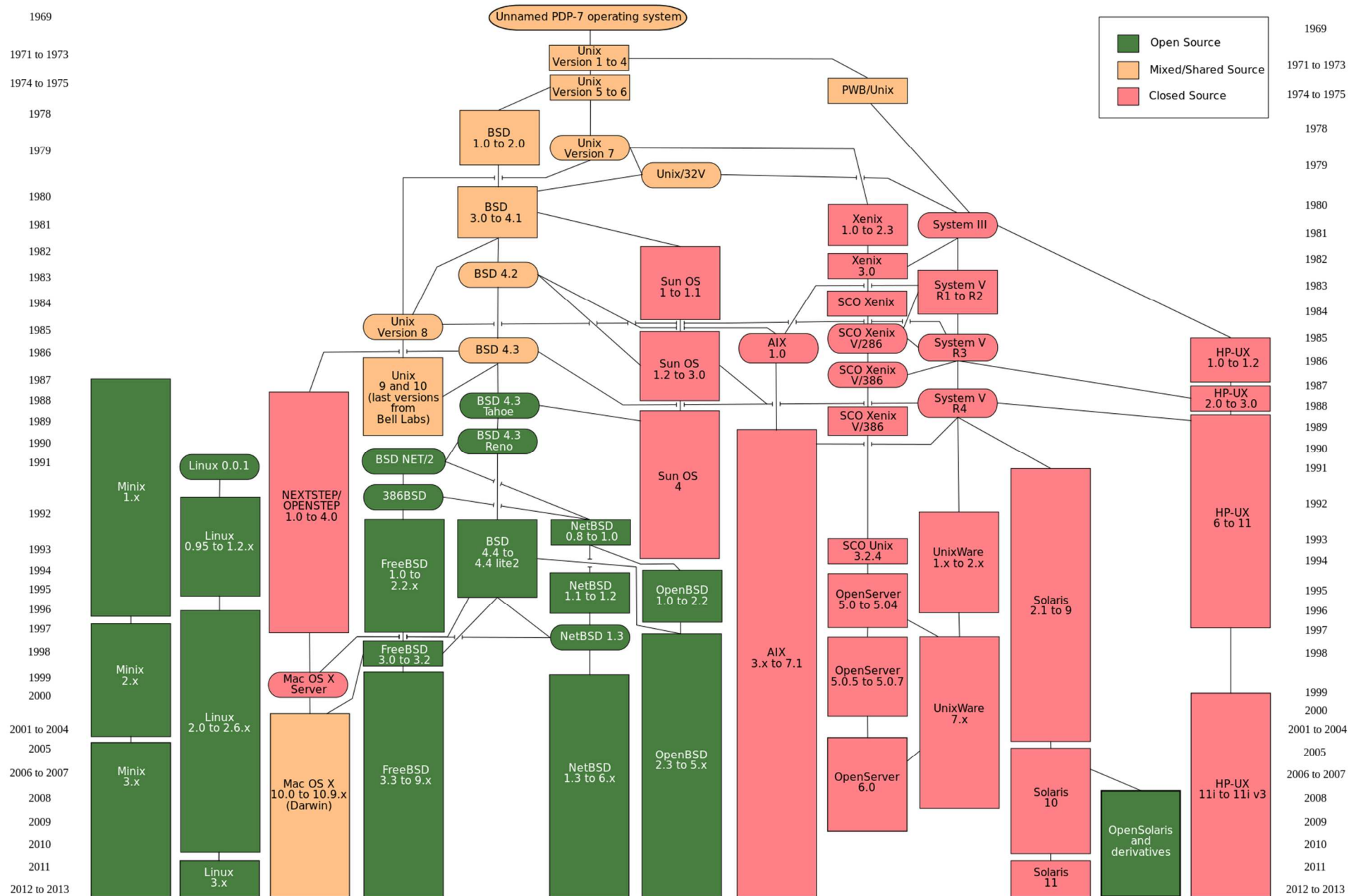


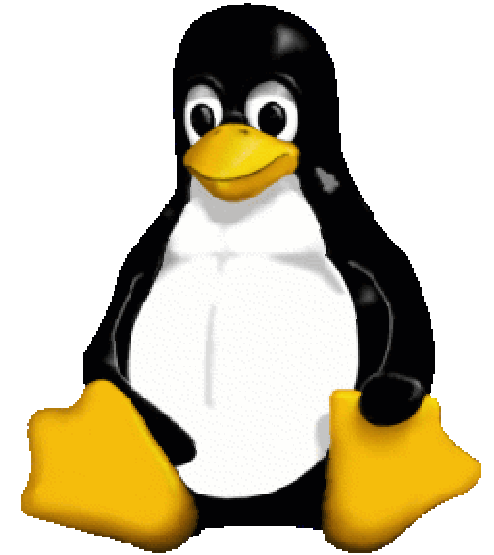
Unix

Linux



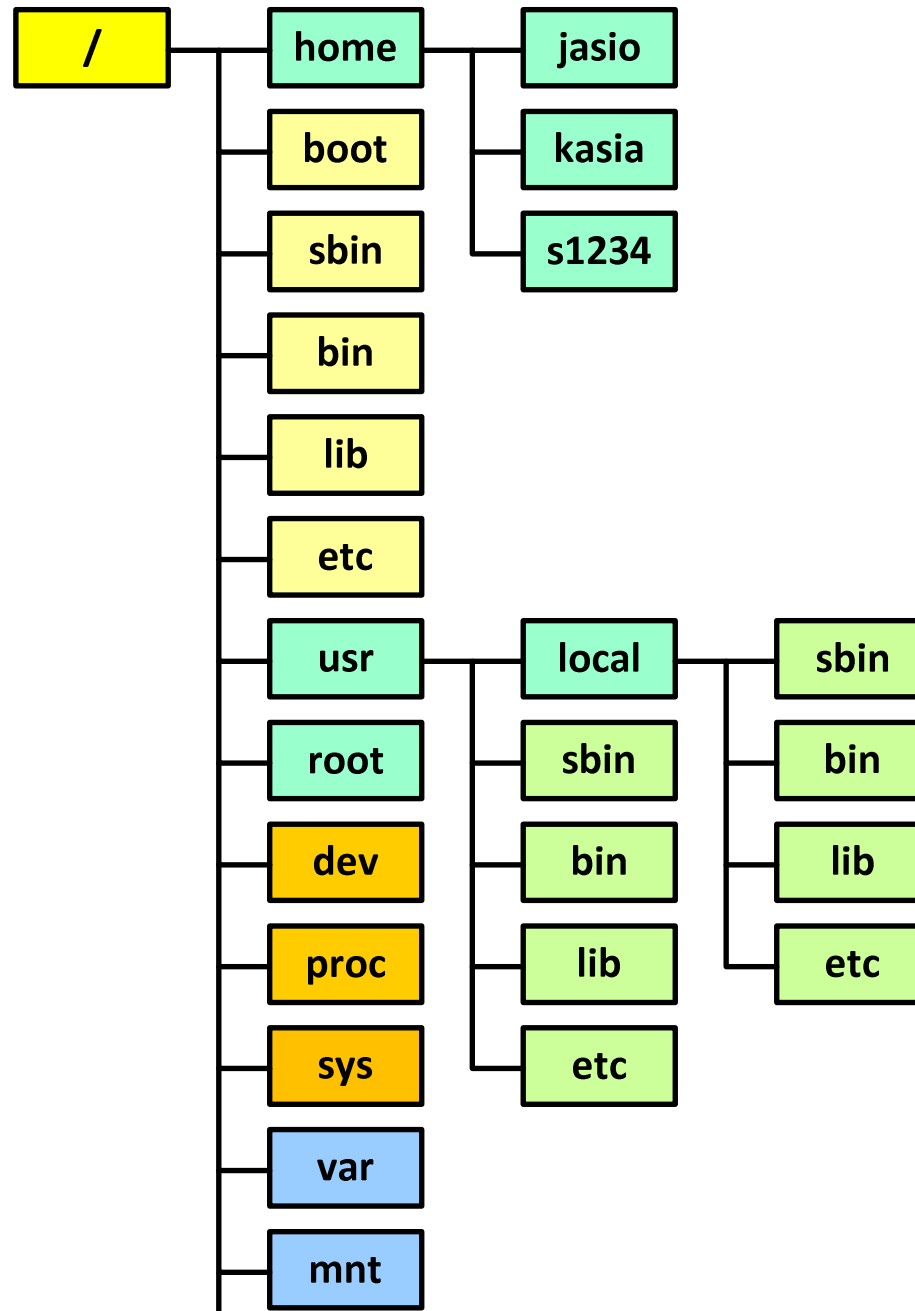
Systemy operacyjne - Pojęcie systemu operacyjnego





4. System operacyjny Unix, Linux

- System wieloprocesowy z podziałem czasu
- System wielodostępowy
- System napisany przez programistów dla programistów
- System napisany według zasady „proste jest lepsze”
- System z hierarchiczną strukturą katalogów
- System posługujący się plikami tekstowymi
- System, w którym złożone operacje realizuje się za pomocą wielu prostych poleceń – przetwarzanie potokowe





Atrybuty plików i uprawnienia do plików

4.1. Atrybuty plików i katalogów

Polecenie `ls` wypisuje nazwy plików i katalogów.

Polecenie `ls -l` wypisuje atrybuty plików w bieżącym katalogu

Polecenie **`ls -a` wypisuje nazwy plików łącznie z ukrytymi**

Jako nazwy ukryte w systemie Linux traktowane są wszystkie nazwy rozpoczynające się od kropki.

Przykład:

```
drwxr-xr-x 5 root root 4096 kwi 26 2005 mnt
-rw-r--r-- 1 root root 10448 gru 30 2004 nsh5.tar.bz2
drwxr-xr-x 2 root root 4096 sie 12 2004 opt
```



Typ plików:

-(f) - zwykły plik

d - katalog

l - dowiązanie symboliczne (link)

c - specjalny plik znakowy

b - specjalny plik blokowy

s - gniazdo (semafor)

p - łącze nazwane FIFO

Atrybuty uprawnień:

r – odczyt

w – zapis

x – wykonanie albo dostęp do katalogu



Uprawnienia (uprawnieni):

u – user – właściciel

g – group – grupa

o – others – inni

a – all – wszyscy

Inne atrybuty to:

- Liczba dowiązań,
- Właściciel,
- Grupa,
- Rozmiar,
- Czas modyfikacji,
- Nazwa



- **4.2. Zmiana uprawnień do plików i katalogów**

Polecenie `chmod [ugoa][+ -=][rwx] nazwa`

gdzie:

- + oznacza nadanie uprawnienia
- oznacza odebranie uprawnienia
- = oznacza nadanie uprawnienia i odebranie wszystkich innych

Uprawnienia można też podawać numerycznie w zapisie ósemkowym (r=4,w=2,x=1) np.:

`chmod 750 nazwa`

co jest równoważne poleceniu:

`chmod u+rwx,g=rx,o-rwx nazwa`



4.3. Znaczenie atrybutów w odniesieniu do plików i katalogów

Dla katalogów:

- r – możliwość odczytu zawartości katalogu (ls)
- w – możliwość modyfikacji zawartości katalogu (tworzenie i usuwanie plików (rm, mv, itp.)
- x – możliwość „wejścia” do katalogu (cd)

Dla plików:

- r – możliwość odczytu zawartości pliku (np. cat)
- w – możliwość modyfikowania zawartości pliku (np. vi)
- x – możliwość uruchamiania pliku (plik jest programem)



```
-  r  w  x  r  w  x  r  w  x  
   4  2  1  4  2  1  4  2  1  
   u      g      o
```

Uwaga!

Właściciela pliku dotyczą tylko uprawnienia dla właściciela, uprawnienia grupy i innych nie mają znaczenia.

Członków grupy dotyczą tylko uprawnienia grupy.

Inni to użytkownicy, którzy nie są ani właścicielem pliku, ani członkiem grupy.



4.4. Modyfikowanie właściciela pliku

Zmiana właściciela

chown nowy_właściciel plik

zmiana grupy

chgrp nowa_grupa plik

albo równoczesna zmiana właściciela i grupy

chown nowy_właściciel.nowagrupa plik

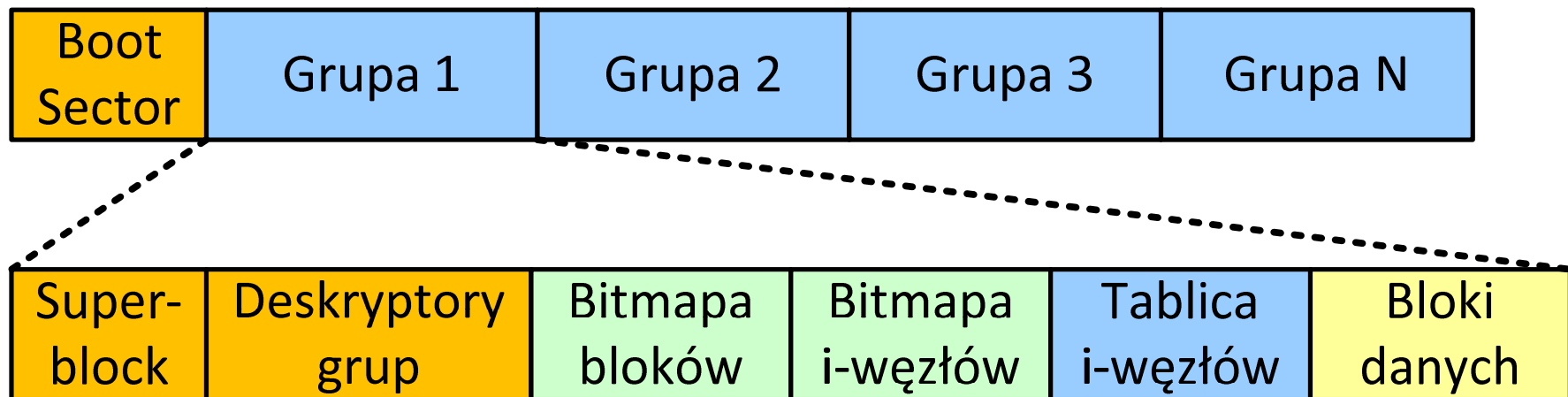


System plików

4.6. Budowa systemu plików

Macierzystym systemem plików Linuksa jest system **ext2**. Ostatnio stosowany system **ext3** ma identyczną budowę, ale dodatkowo wprowadza „księgowanie” zmian wprowadzanych na dysku co umożliwia szybką naprawę systemu plików w przypadku np. awarii zasilania, i gwarantuje wysoki stopień bezpieczeństwa spójności zapisywanych danych. Najnowsza wersja systemu plików **ext4** wprowadza 64-bitowe struktury danych rozszerzając obsługiwaną przestrzeń dyskową do 1 Exabajta (1mln TB) i rozmiar pojedynczego pliku do 16 TB. Wprowadzono oczywiście również szereg usprawnień obsługi systemu plików.

Logiczna budowa systemu plików ext2:



Bitmapy zajmują dokładnie jeden blok – przy bloku 4KB daje to 32K bitów co odpowiada 128MB danych na grupę.

Chcąc odczytać dane z pliku system najpierw musi odszukać nazwę pliku w katalogu, następnie odczytać i-węzeł pliku, a dopiero na podstawie jego zawartości odszukiwane są bloki z danymi pliku.

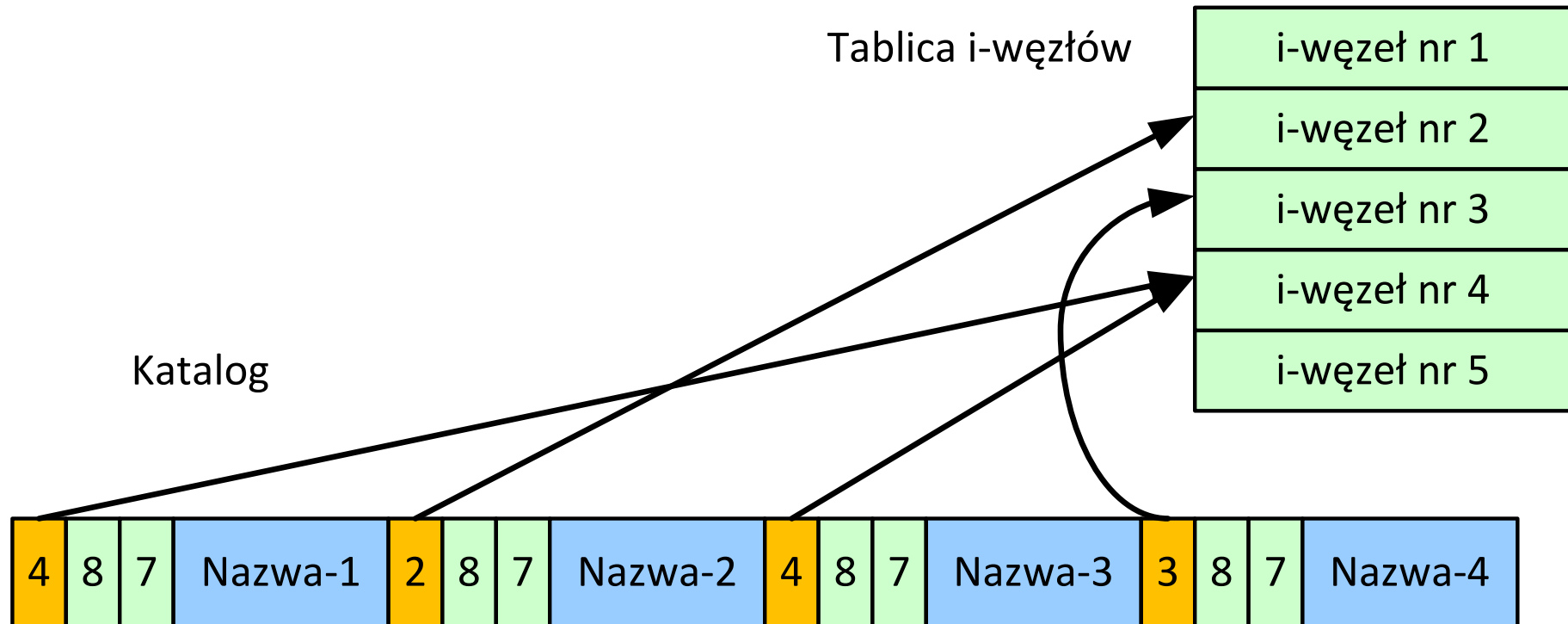
Logiczna budowa systemu plików ext2:

Zawartość katalogu:

Numer i-węzła	Długość wpisu	Długość nazwy	Nazwa pliku
------------------	------------------	------------------	-------------

- Wpis w katalogu o zmiennej długości pozwala na łatwą realizację długich nazw plików bez marnowania miejsca w przypadku gdy nazwy plików są krótkie.
- Maksymalna długość nazwy to 255 znaków.
- Pliki skasowane mają zero w miejscu numeru i-węzła.

Relacja katalogu do tablicy i-węzłów.



Do przechowywania adresów bloków z danymi w i-węźle wykorzystywana jest 15 elementowa tablica.

Dwanaście pierwszych elementów tej tablicy wskazuje na dwanaście pierwszych bloków danych pliku.

Trzynasty element jest wykorzystywany do adresowania pośredniego,

czternasty do adresowania podwójnie pośredniego, a piętnasty do adresowania potrójnie pośredniego.

Przedstawiony sposób adresowania bloków danych pozwala z jednej strony zwiększyć wydajność obsługi małych plików (do 12 bloków), z drugiej umożliwia przechowywanie bardzo dużych plików.



Dla przykładu: dla rozmiaru bloku wynoszącego 4KB
bajtów największy plik adresowany bezpośrednio
to $12 * 4KB = \mathbf{48KB}$,

adresowany pośrednio to
 $12 * 4KB + 4096 / 4 * 4KB = \mathbf{4MB} + 48KB$,

adresowany podwójnie pośrednio to
 $12 * 4KB + 4096 / 4 * 4KB + 4096 / 4 * 4096 / 4 * 4KB = \mathbf{4GB} +$
 $4MB + 48KB$,

a adresowany potrójnie pośrednio to ponad **4TB**,

Uwaga:

Wiele aplikacji 32-bitowych nie potrafi obsługiwać
plików o rozmiarze powyżej 2GB.



System plików **vFAT (File Allocation Table)**

- FAT12 1980 rozmiar dysku do 32MB (2.88MB)
- FAT16 1988 rozmiar dysku do 2GB
- FAT32(28) 1996 rozmiar dysku do 2TB (40GB Win7)
- exFAT (64) 2008 dysk 64ZB plik 16EB (512TB)

Cechy:

- Nazwy 8+3 znaków, nierozróżniana wielkość liter, od Windows 95 obsługa LFN do 255 znaków UTF-16, atrybuty obejmują tylko datę, czas oraz ReadOnly, Archive, System, Hidden, VolumeLabel(?).

Wady:

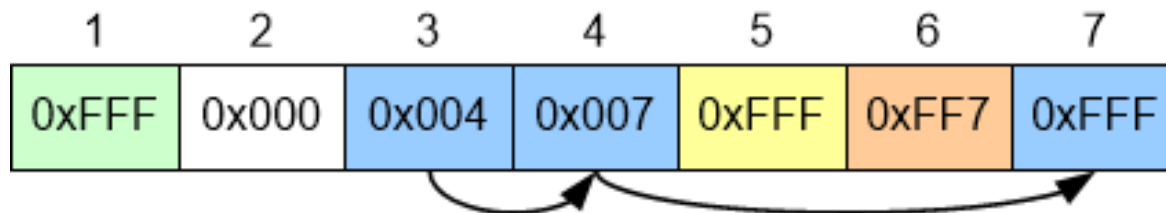
- FAT16 duży klaster 32KB, FAT32 olbrzymi FAT do 2GB.

System plików vFAT (File Allocation Table)

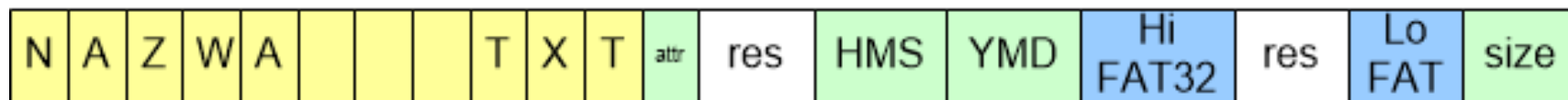
Logiczna budowa systemu plików



Budowa tablicy alokacji plików (na przykładzie FAT12)



Wpis w katalogu (32B)



System plików vFAT (File Allocation Table)

LFN – Long File Names

- Do przechowywania długiej nazwy (do 255 znaków) wykorzystywanych jest wiele wpisów w katalogu, każdy zawiera 11 znaków UTF-16 i dodatkowo istnieje wpis z krótką (8+3) wersją nazwy.

D	U	G	A	N	A	~	1	T	X	T
---	---	---	---	---	---	---	---	---	---	---

.	t	x	t							
---	---	---	---	--	--	--	--	--	--	--

D	ł	u	g	a		N	a	z	w	a
---	---	---	---	---	--	---	---	---	---	---



System plików **NTFS (New Technology File System)**

Wybrane cechy:

- „*Journaling*” i transakcyjność operacji na plikach
- Twarde linki do katalogów, montowanie woluminów w katalogach
- Kompresja plików i szyfrowanie systemu plików
- Rozszerzalne atrybuty plików
- Limity przestrzeni użytkowników (ang. *quota*)
- Pliki rzadkie – sektory wypełnione zerami nie zajmują miejsca na dysku
- Czasy w UTC (dokładność 100ns)
- Linki symboliczne (od Windows Vista)
- Możliwość zmiany rozmiaru woluminów w czasie pracy



System plików **NTFS (New Technology File System)**

Budowa oparta o metapliki:

- \$MFT
- \$MFTMirr
- \$LogFile
- \$Volume
- \$AttrDef
- .
- \$Bitmap
- \$Boot
- \$BadClus
- \$Secure
- \$UpCase
- \$Extend\ \$Quota
- \$Extend\

Porównanie pojemności system plików

	ext3	ext4	FAT32	NTFS
Długość nazwy	255 B	255 B	255 UTF-16	255 UTF-16
Rozmiar dysku	32TB	1EB	2TB (40GB)	16EB (128TB)
Rozmiar pliku	4TB (4GB/2TB)	16TB	2GB	16EB (16TB)
Rozmiar bitmapy (dla dysku 2TB)	64MB	64MB	2048MB	64MB
Rozmiar bloku danych	4KB	4KB	32KB	4KB
<i>Journaling</i>	TAK	TAK	NIE	TAK

4.7 Tworzenie i utrzymanie systemu plików.

4.7.1. Nazwy dysków

`/dev/cdrom` - domyślny napęd optyczny

`/dev/fd0` - pierwszy napęd dyskietek

`/dev/hda` - master w pierwszym kontrolerze IDE

`/dev/hdb` - slave w pierwszym kontrolerze IDE

`/dev/hdc` - master w drugim kontrolerze IDE

`/dev/sda` - pierwszy dysk typu SCSI (SATA, USB)

`/dev/sdb` - drugi dysk typu SCSI (SATA, USB)

Współczesne dystrybucje Linuksa traktują wszystkie dyski jako dyski typu SCSI niezależnie od technologii interfejsu komunikacyjnego.



4.7.2. Nazwy partycji

- `/dev/hda1` - pierwsza partycja na pierwszym dysku IDE
- `/dev/sdb4` - czwarta partycja na drugim dysku SATA
- `/dev/sdb5` - pierwsza partycja rozszerzona (5) na drugim dysku SATA

Uwaga: numeracja partycji rozpoczyna się od wartości 1.

4.7.3. Tworzenie partycji

`fdisk /dev/sda`

- polecenia

m – lista poleceń

p – lista partycji

n – utwórz nową partycję

w – zapisz zmiany i zakończ



4.7.4. Tworzenie systemu plików (formatowanie partycji)

- partycja wymiany

mkswap /dev/sda1

- partycja z systemem plików ext2

mkfs /dev/sda4

- partycja z systemem plików ext3 (ext2 z księgowaniem)

mkfs -j /dev/sda3

- partycja z systemem plików ext4

mkfs.ext4 /dev/sda2

4.7.5. Sprawdzanie spójności systemu plików na partycji (system plików powinien być odmontowany)

fsck /dev/sda4



4.7.6. Podłączanie partycji wymiany do pamięci wirtualnej

swapon /dev/sda1

4.7.7. Montowanie systemu plików w strukturze katalogów

mount /dev/sda2 /folder

4.7.8. Odmontowywanie partycji

swapoff /dev/sda1

umount /dev/sda2

albo

umount /folder



4.5. Przeszukiwanie systemu plików

which nazwa

podaje pełną ścieżkę dostępu do programu nazwa

find ~/ -name 'skrypt*'

poszukuje, począwszy od katalogu domowego, plików, których nazwa rozpoczyna się od frazy skrypt

find ~/ -mtime -1

poszukuje, począwszy od katalogu domowego, plików, których czas modyfikacji jest nie większy niż 1 dzień

find ~/ -mtime +15

poszukuje, począwszy od katalogu domowego, plików, których ostatnia modyfikacja miała miejsce dawniej niż 15 dni temu



find ~/ -size +1M

poszukuje, począwszy od katalogu domowego, plików, których rozmiar jest większy od 1MB

find ~/ -size -100c

poszukuje, począwszy od katalogu domowego, plików, których rozmiar nie jest większy od 100 znaków

find ~/ -type f -exec ls -l {} \;

poszukuje wszystkich plików zwykłych i wypisuje dla każdego znalezionej pliku atrybuty poleceniem ls -l



5. Wprowadzenie do wyrażeń regularnych

Wyrażenie regularne (ang. *regular expression*) jest zbiorem uogólnionych reguł opisujących napis (ciąg znaków). Jeżeli pewien napis spełnia reguły wyrażenia to mówimy, że *pasuje* do tego wyrażenia.

Elementy budowy wyrażeń regularnych:

Atomy (ang. *atoms*)

- . - kropka - dowolny pojedynczy znak alfanumeryczny
- [] - lista znaków ujętych w nawiasy kwadratowe - pojedynczy znak z listy w nawiasach kwadratowych
- () - wyrażenie regularne ujęte w nawiasy okrągłe - wszystko co pasuje do wyrażenia w nawiasie

znak - znak alfanumeryczny - dokładnie ten znak



Zakotwiczenia (ang. *assertions*)

^ - początek tekstu

\$ - koniec tekstu

Kwantyfikatory – powtórzenia

+ - jedno lub więcej wystąpień atomu

? - zero lub jedno wystąpienie atomu

* - zero lub więcej wystąpień atomu

{2} - dokładnie dwa wystąpienia atomu

{2,} - przynajmniej dwa wystąpienia atomu

{2,5} - od dwóch do pięciu wystąpień atomu

Znaki specjalne poprzedzamy symbolem \ (ang. *backslash*)



Przykłady:

al - każdy tekst zawierający parę liter al. np. kalka, albo, malwa

^ko - każdy tekst rozpoczynający się od znaków ko. np. korba, kok

^[0-9]{9}\$ - napis składający się z dokładnie 9-ciu cyfr. np. 500600900

^[0-9]{2}-[0-9]{3}\$ - napis odpowiadający polskiemu kodowi pocztowemu.



5.1. Symbole wieloznaczne w nazwach plików

? – zastępuje jeden znak na danej pozycji

***** - zastępuje dowolną liczbę znaków

[znaki] – na danej pozycji może wystąpić tylko znak z listy w nawiasach kwadratowych

[!znaki] – na danej pozycji nie może wystąpić znak z listy rozpoczynającej się od symbolu !

()|() – wyrażenia ujęte w nawiasy okrągłe i połączone symbolem | oznaczają logiczną alternatywę

Uwaga: Symbole wieloznaczne w nazwach plików nie są wyrażeniami regularnymi.



Potoki i przekierowania



6. Potoki i przekierowania

6.1. Przekierowania

Przekierowania umożliwiają zmianę domyślnego wyjścia lub wejścia programów. Pozwalają np. zapisywać wyniki działania poleceń lub przygotowywać zestawy danych wejściowych dla programów.

Przykłady:

Przekierowanie wyjścia do nowego pliku

ls >plik

Przekierowanie wyjścia z dopisaniem na końcu pliku

ls >>plik



Przekierowanie wejścia z pliku

cat <plik

Przekierowanie równocześnie wejścia i wyjścia

cat <plik1 >plik2

Przekierowanie komunikatów o błędach do urządzenia
/dev/null

ls -R / 2>/dev/null

Przekierowanie komunikatów o błędach do standardowego
wyjścia

ls -R / 2>&1



6.2. Potoki

W przetwarzaniu potokowym możemy uczynić standardowe wyjście jednego programu wejściem kolejnego, rozdzielając odpowiednie polecenia symbolem | (ang. *pipe*).

Przykłady:

```
cat plik | more
```

```
ls -l / | sort -n k5,5
```

```
ls -l / | sort -r k9
```

```
cat /etc/passwd | sort -t: -n k3,3 | more
```

```
cat /etc/passwd | grep Anna | sort | more
```



6.3. Polecenia przydatne do przetwarzania potokowego

more – polecenie zatrzymuje wyświetlanie po zapełnieniu każdego pełnego ekranu, kontynuacja klawiszem spacji, wcześniejsze zakończenie klawiszem q.

less – polecenie podobne do more dodatkowo umożliwia przewijanie linia po linii w górę i w dół oraz przeszukiwanie zawartości wyświetlanego tekstu (/ , ?), zakończenie konieczne klawiszem q.



sort – sortuje zawartość pliku

- r odwrotny kierunek sortowania

- n porównuj wartości numeryczne

- t<znak> traktuj znak jako separator pól

- k<n> rozpocznaj porównywanie od n pola

- k<n>,<m> porównywanie pól od n do m

np. **sort k=3,3 plik**

- wypisuje na ekranie zawartość pliku posortowaną według 3-go wyrazu w kolejnych liniach



grep – wyszukuje linii tekstu pasujących do wyrażenia (**egrep**)

- v wyszukuj niepasujące
- c tylko zliczaj liczbę linii
- n wypisuj numery linii
- i ignoruj różnice dużych małych liter

np. **egrep -c '^s[0-9]{5,6}' /etc/passwd**

- wypisuje liczbę linii w pliku zawierających napis

cut- wycina z linii pliku wybrane pola

- d<znak> traktuj znak jako separator pól
- f<n> wybierz tylko pole n
- f<n>,<m> wybierz pola n i m
- f<n>-<m> wybierz pola od n do m
- c<n>- wybierz od n tego znaku linii

np. **cut -d: -f5 /etc/passwd**

- wypisuje tylko 5-te pole z linii pliku traktując jako separator pól dwukropek



sed - program do przetwarzania linii tekstu

np. **sed s/wzor/nowy/g plik**

- wyszukuje w kolejnych liniach pliku napisów pasujących do wyrażenia regularnego *wzor* i wymienia je na napis *nowy*

tee - rozdzielenie standardowego wyjścia równocześnie na ekran i do pliku

ls - wyjście tylko na ekran

ls >plik - wyjście tylko do pliku

ls | tee plik - wyjście na ekran i do pliku



awk - język programowania program do przetwarzania tekstów

-F<znak> traktuj znak jako separator pól

BEGIN blok przetwarzania przed analizą linii

END blok przetwarzania po analizie linii

Dostępne polecenia: print, if, for, operacje arytmetyczne itp.

Zgodnie ze składnią języka C

np. **awk -F: '{ print \$1 }' /etc/passwd**

- wypisuje pierwszy wyraz z każdej linii pliku /etc/passwd, traktując jako separator wyrazów : (dwukropek)

np. **awk -F: '{if(\$4==501) print \$1" "\$5}' /etc/passwd**

- wypisuje z pliku /etc/passwd pierwszy i piąty wyraz pod warunkiem, że czwarty wyraz to 501



7. Archiwizacja i kompresja danych

tar - utworzenie jednoplikowego archiwum z wielu plików i katalogów oraz wyodrębnianie plików z archiwum

c - tworzy archiwum

u - aktualizuje archiwum

x - wyodrębnia pliki z archiwum

f <nazwa> zapisuje/odczytuje do/z pliku

v - wypisuje nazwy przetwarzanych plików na ekranie

z - stosuje kompresję gzip

j – stosuje kompresję bzip2



gzip

- bezstratna kompresja zawartości jednego pliku

-1 - najszybsza kompresja

-9 - najlepsza kompresja

gunzip

- dekompresja zawartości pliku

bzip2

- obsługa jak gzip, inny, nowszy algorytm kompresji efektywniejszy dla dużych plików

bunzip2

- obsługa jak gunzip

xz

- kompresja algorytmami zgodnymi z 7-zip

xz -d

- dekompresja xz

zip

- popularny, występujący w wielu systemach operacyjnych, program do kompresji danych umożliwiający kompresję wielu plików i katalogów w jednym archiwum - niestandardowy dla Linuksa

unzip

- dekompresor programu zip



Przykłady:

Utworzenie archiwum z wszystkich plików i podkatalogów katalogu
/home/s99111

```
tar -cf archiwum.tar /home/s99111/
```

jw. ale dodatkowo z kompresją gzip

```
tar -c /home/s99111/ | gzip > archiwum.tgz
```

```
tar -czf archiwum.tgz /home/s99111/
```

```
tar -c /home/s99111/ | bzip2 > archiwum.tar.bz2
```

rozpakowanie skompresowanego archiwum

```
tar -xzf archiwum.tgz
```

```
gunzip < archiwum.tar.gz | tar -x
```

```
bunzip2 < archiwum.tar.bz2 | tar -x
```



Edycja tekstów



8. Edytor tekstów vi (vim)

vi jest pełnoekranowym edytorem tekstów. **vim** jest wersją rozszerzoną m.in. o wielokrotne cofanie operacji edycyjnych, kolorowanie składni plików i inne.

vi charakteryzuje się kilkoma trybami pracy:

- tryb poleceń
- tryb wstawiania
- tryb nadpisywania
- tryb wizualny

Przedstawione tu polecenia stanowią wybrany zbiór najczęściej stosowanych, po kompletny opis wszystkich odsyłam do podręczników.



Uruchamianie

- **view nazwa** - tylko do odczytu pliku nazwa
- **vi nazwa** - otwarcie pliku do edycji
- **vi +23 nazwa** - otwarcie do edycji od linii numer 23
- **vi +/słowo** - otwarcie do edycji od pierwszego wystąpienia podanego słowa w tekście

Przejdźcie do trybu edycji

- **i** - wstawianie znaków przed kursorem
- **I** - wstawianie znaków przed bieżącą linią
- **a** - wstawianie znaków za kursorem
- **A** - wstawianie znaków za bieżącą linią
- **o** - wstawianie do nowej linii poniżej bieżącej
- **O** - wstawianie do nowej linii powyżej bieżącej
- **r** - nadpisywanie od kursora
- **R** - nadpisywanie całej linii



Przejdźcie do trybu poleceń

- ESC

Polecenia

Zastępowanie i usuwanie

- s - zastąpienie pojedynczego znaku
- S - zastąpienie całej linii
- x - usunięcie pojedynczego znaku
- d kursor - usunięcie zgodnie z kierunkiem kursora
- dd - usunięcie całej linii
- D - usunięcie do końca linii



Kończenie pracy

- **:q** - zakończ – tylko jeśli nie było zmian
- **:q!** - zakończ i porzuć wprowadzone zmiany
- **:w** - zapisz
- **:w nazwa** - zapisz kopię jako (waga zapisz kopię, a nie zapisz jako)
- **:x** - zakończ zapisując jeśli były zmiany
- **:wq** - zapisz i zakończ

Poszukiwanie

- **/wzór** - poszukiwanie wzoru w kierunku końca tekstu
- **/** - powtórne poszukiwanie
- **?wzór** - poszukiwanie wzoru w kierunku początku tekstu
- **?** - powtórne poszukiwanie

Wzór jest traktowany jak wyrażenie regularne.



Poszukiwanie i zamiana

- **:g/szu/s//zam/**
 - wyszukaj pierwszego wystąpienia szu w kolejnych liniach i zamień na zam
- **:g/szu/s//zam/g**
 - wyszukaj każdego wystąpienia szu w kolejnych liniach i zamień na zam
- **:g/szu/s//zam/gc**
 - wyszukaj każdego wystąpienia szu w kolejnych liniach i zamień na zam po potwierdzeniu przez użytkownika
- **:g/wzor/s/szu/zam/gc**
 - wyszukaj każdego wystąpienia szu w liniach pasujących do wzoru i zamień na zam po potwierdzeniu przez użytkownika



Inne polecenia

- J - połącz z kolejną linią
- u - cofnij ostatnią operację zmiany tekstu
- . - powtórz ostatnią operację edycji
- U - cofnij wszystkie zmiany w bieżącej linii
- ^G - wypisz aktualny status
- y - skopiuj wskazany tekst do bufora
- p - wstaw tekst za kursorem
- P - wstaw tekst przed kursorem

Poprzedzenie polecenia liczbą powoduje wielokrotne powtórzenie polecenia np. 23dd – usuwa kolejne 23 wiersze.



Inne edytory tekstu

Do innych popularnych edytorów systemu Linux należą:

- pico,
- nano,
- mcedit,
- joe,
- emacs

Są to edytory pełnoekranowe, najczęściej z opisem obsługi wypisanym w linii stanu.



Praca w sieci



9. Praca w sieci

Do pracy w sieci potrzebne jest właściwa konfiguracja:

- adresu IP i maski sieci dla interfejsu sieciowego
- adresu IP bramy zapewniającej wyjście „na świat”
- adresu IP serwera usługi DNS, aby zapewnić translację adresów domenowych na adresy IP

Konfiguracja do pracy w sieci, w której jest serwer DHCP:

- **dhclient**



Konfiguracja ręczna do pracy w sieci

- **ifconfig eth0 <IP> netmask <MASKA>**
- **route add default gw <BRAMA>**
- **vim /etc/resolv.conf**
 - **nameserver <DNS>**

Konfiguracja ręczna do pracy w sieci

- **ip addr add <IP>/<MASKA> dev eth0**
- **ip route add default via <BRAMA>**
- **vim /etc/resolv.conf**
 - **nameserver <DNS>**



Poczta elektroniczna

Wysyłanie poczty:

mail user@serwer.domena

echo "Pozdrawiam" | mail -s "Temat" user

mail -s "Temat" user < plik

Czytanie poczty

mail

- h – lista e-maili
- 1 – czytaj pierwszy list
- 2 – czytaj drugi list itd.
- n – czytaj następny list
- d – usuń ostatnio czytany list
- q – wyjdź



Standardowo (choć jest to zależne od konfiguracji) przeczytana poczta jest przenoszona ze skrzynki pocztowej serwera pocztowego do pliku mbox w katalogu domowym użytkownika.

Ponowne czytanie przeczytanej poczty (z pliku mbox)

mail -f mbox

Inne konsolowe programy pocztowe

- **mutt**
- **pine**



Transfer plików FTP (File Transfer Protocol)

Przykładowa sesja:

ftp

>open ftp.task.gda.pl

>user anonymous

>pass user@localhost

>passive

>ls

>cd pub

>ls

>binary

>get plik

>bye



Transfer plików z wykorzystaniem protokołu SSH

Transfer plików za pomocą protokołu FTP ma poważną lukę bezpieczeństwa. Zarówno dane uwierzytelnienia użytkownika (login i hasło) jak i transmitowane dane są przesyłane tekstem jawnym. Do transmisji plików szyfrowanym łączem wykorzystujemy polecenie scp.

Przykłady:

scp s123456@lab204:/home/s123456/plik ~/plik

Skopiuj plik z hosta lab204 z katalogu /home/s123456 uwierzytelniając się jako użytkownik s123456 do swojego katalogu domowego. Oczywiście trzeba będzie podać hasło użytkownika s123456.

scp ./plik2 s123456@10.1.0.204:/home/s123456/plik2



WWW (World Wide Web)

Przeglądarki pracujące w trybie tekstowym:

- **elinks**
- **links**
- **lynx**

Automatyczne pobieranie treści:

- **wget <http://guminski.net/linux/so2013.zip>**
- **wget <ftp://ftp.task.gda.pl/welcome.msg>**



CRON



10. Zlecenie poleceń do wykonania w określonym czasie

- **at HH:MM** - dodanie nowego zadania do kolejki
- **atq** - wyświetlenie kolejki zadań do wykonania
- **atrm N** - usunięcie zadania o numerze N z kolejki

Dodając nowe zadanie podajemy kolejne polecenia z wiersza poleceń i kończymy kombinacją klawiszy Ctrl-D.

Polecenia **at** mogą używać wyłącznie użytkownicy wymienieni w pliku **/etc/at.allow** albo wszyscy, którzy nie są wymienieni w pliku **/etc/at.deny**.



Okresowe wykonywanie poleceń

Polecenie **crontab** umożliwia zlecanie systemowi wykonywania określonych poleceń w zadanych powtarzających się chwilach czasu.

- **crontab -e** - edycja tablicy zadań okresowych
- **crontab -r** - usunięcie tablicy zadań okresowych
- **crontab -l** - wypisanie tablicy zadań okresowych

Polecenia **crontab** mogą używać wyłącznie użytkownicy wymienieni w pliku **/etc/cron.allow** albo wszyscy, którzy nie są wymienieni w pliku **/etc/cron.deny**.



Składnia pliku crontab:

MIN. GODZ. DZIEŃ MIES. DZIEŃ-TYGODNIA POLECENIE

Przykłady:

- *** * * * * polecenie_co_minute**
- **0 0 * * * polecenie_codziennie_o_polnocy**
- **0 12 * * 0 polecenie_w_poludnie_w_kazda_niedziele**
- **45 6 * 1-7,9-12 1-5 w_dni_robocze_bez_sierpnia**
- **59 8-20/3 * * * co_trzecia_godzinie_od_8-mej_do_20-tej**
- **0 0 13 * 5 w_kazdy_piątek_i_kazdego_13-tego**



W pliku crontab dopuszczalne jest używanie liczb rozdzielonych przecinkami, zakresów rozdzielonych minusem i wyborów co n-tej wartości po znaku / (ang. slash).

W miejsce miesięcy i dni tygodnia można używać angielskich skrótów trzyliterowych np.: *jan, feb, ..., mon, tue,*

W większości systemów zarówno liczba 0 jak i liczba 7 mogą pełnić funkcję oznaczenia niedzieli.

Jeżeli określony jest równocześnie dzień miesiąca i dzień tygodnia to polecenie działa niezależnie dla obu warunków.



Systemowe okresowe wykonywanie poleceń

W pliku **/etc/crontab** przechowywana jest systemowa tablica zadań do okresowego wykonania. Składnia podobna jak w poleceniu crontab, ale przed poleceniem podajemy nazwę użytkownika z uprawnieniami którego polecenie zostanie wykonane.

W pliku tym zwykle są umieszczone wywołania uruchamiające wszystkie skrypty umieszczone w katalogach */etc/cron.hourly/*, */etc/cron.daily/*, */etc/cron.weekly/*, */etc/cron.monthly/* odpowiednio co godzinę, codziennie, co tydzień i raz w miesiącu. Ułatwia to administrację czynnościami powtarzanymi okresowo.



POLITECHNIKA
GDAŃSKA

Systemy operacyjne - Unix, Linux

Skrypty



Skrypty powłoki

Dowolny zestaw poleceń zapisanych w pliku z nadanym atrybutem wykonywalności staje się wykonywalnym skryptem powłoki.

Przykład:

```
#!/bin/bash
```

```
echo Hello
```

Znak # (ang. *hash*) rozpoczyna komentarz.

Komentarz kończy się z końcem linii.

Pierwsza linia skryptu powinna rozpoczynać się od komentarza, w którym bezpośrednio po wykrzykniku podany jest interpreter poleceń zawartych w skrypcie.

np. **#!/bin/sh**



Instrukcje sterujące stosowane w przetwarzaniu wsadowym

if cmd; then cmd; fi

if cmd; then cmd; else cmd; fi

Jeśli polecenie po if zakończy się sukcesem, to wykonane zostaje polecenia po then, w innym przypadku polecenia po else.

case word in; nazwa) cmd;; nazwa[|nazwa]) cmd;; esac

w zależności od wartości słowa, poszukiwana jest nazwa i wykonywane polecenie po).



for nazwa in words; do cmd; done

Dla nazwy przyjmującej wartości z listy słów wykonuj polecenia po do.

for ((wyr ; wyr ; wyr)); do cmd; done

Pętla for jak w języku C.

np. `for ((i=0; $i<10; i=$(($i+1)))); do echo $i; done`

while cmd; do cmd; done

until cmd; do cmd; done

W czasie gdy (while) albo aż do (until) polecenie przed do kończy się sukcesem wykonuj polecenia po do.



nazwa=wartość – przypisanie wartości do symbolu nazwa

\$nazwa – wartość pamiętana w symbolu nazwa

\$(polecenie) – wykonanie polecenia i wstawienie jego wyniku w miejscu wywołania

\$((wyrażenie)) – obliczenie wartości wyrażenia arytmetycznego. Dopuszczalne są wszystkie operatory z języka C oraz ** jako potęgowanie.



Symbole predefiniowane

\$# - liczba parametrów przekazanych do skryptu przy uruchomieniu

\$@ - wszystkie parametry jako jeden łańcuch znaków

\$1, \$2 .. \$9 - wartość pierwszego, drugiego itd. parametru przekazanego do skryptu

\$0 – nazwa skryptu

\$\$ - PID powłoki

\$? – kod zakończenia ostatniego potoku

#! – PID procesu ostatnio uruchomionego do pracy w tle

Wyrażenia warunkowe

- | | |
|----------------------------|--|
| [-d plik] | – plik istnieje i jest katalogiem |
| [-f plik] | – plik istnieje i jest zwykłym plikiem |
| [-s plik] | – plik istnieje i ma rozmiar większy od zera |
| [-r plik] | – plik istnieje i możemy go czytać |
| [-w plik] | – plik istnieje i możemy go modyfikować |
| [-x plik] | – plik istnieje i możemy go uruchamiać |
| [plik1 -nt plik2] | – plik1 jest nowszy od pliku2 |
| [plik1 -ot plik2] | – plik1 jest starszy od pliku2 |
| [-z napis] | – napis jest pusty |
| [-n napis] | – napis nie jest pusty |



[napis1 == napis2] – napis1 jest identyczny z napisem2. Można stosować relacje ==, !=, <, >

[wart1 **-eq** wart2] – wartość1 jest numerycznie równa wartości2. Można stosować następujące relacje:

- **-eq** równe
- **-ne** różne
- **-lt** mniejsze niż
- **-le** mniejsze lub równe
- **-gt** większe niż
- **-ge** większe lub równe

Wyrażenia warunkowe możemy łączyć relacjami logicznymi:

- **-o** lub
- **-a** i



Przykłady skryptów

```
#!/bin/bash
```

```
i=1
```

```
while [ $i -le 10 ]; do
```

```
    echo $i
```

```
    i=$((i+1))
```

```
    sleep 1
```

```
done
```

-wypisuje liczby od 1 do 10, kolejno co 1 sekundę



```
#!/bin/bash
```

```
if [ -z $1 ]; then
```

```
    echo Używaj podając wyrażenie do obliczenia
```

```
fi
```

```
echo $(( $1 ))
```

- oblicza wartość wyrażenia arytmetycznego podanego jako pierwszy parametr uruchomienia programu

```
#!/bin/bash
```

```
for nazwa in $(ls); do
```

```
    if [ -f $nazwa -a -x $nazwa ]; then
```

```
        echo $nazwa
```

```
    fi
```

```
done
```

- wypisuje tylko te nazwy plików, które są programami



Administracja i utrzymanie systemu



Konta użytkowników

groupadd	- utworzenie nowej grupy
groupdel	- usunięcie grupy
useradd	- założenie nowego konta
usermod	- modyfikacja parametrów konta
userdel	- sunięcie konta
adduser	- skrypt automatyzujący zakładanie konta
chfn	- zmiana informacji dodatkowych o koncie
finger	- sprawdzenie informacji o koncie
passwd	- zmiana hasła użytkownika
chgrp	- zmiana grupy pliku
chown	- zmiana właściciela pliku
chmod	- zmiana atrybutów uprawnień do pliku



Informacje o kontach użytkowników są przechowywane w pliku **/etc/passwd**, a informacje o hasłach i terminach ważności konta w pliku **/etc/shadow**.

Plik **/etc/passwd** jest czytelny dla wszystkich, pliku **/etc/shadow**, ze względów bezpieczeństwa, nie może odczytywać żaden użytkownik.

Jeżeli przy zakładaniu konta jest tworzony katalog użytkownika to są do niego kopiowane wszystkie pliki z katalogu **/etc/skel/** (oczywiście użytkownik staje się właścicielem tych plików).

Przykład:

```
useradd -s /bin/bash -d /home/jasio -m -g users jasio
```



Instalacja oprogramowania

Metoda pierwsza - **Repozytoria**:

korzystanie z oprogramowania umieszczonego w repozytoriach przez twórców danej dystrybucji.

Zalety: automatyczne rozwiązywanie wszystkich zależności, łatwość zarządzania oprogramowaniem.

Wady: ograniczenie tylko do oficjalnego oprogramowania dla danej wersji określonej dystrybucji.

Realizacja Debian (Ubuntu, Knoppix):

apt update

apt install program

Realizacja RedHat (CentOS, Fedora):

yum install program



Instalacja oprogramowania

Metoda druga - **Pakiety**:

korzystanie z oprogramowania przygotowanego do instalacji na określonej platformie sprzętowej.

Zalety: większy zestaw dostępnego oprogramowania.

Wady: konieczność ręcznego spełnienia zależności, kłopotliwe zarządzanie oprogramowaniem.

Realizacja Debian (Ubuntu, Knoppix):

`dpkg -i program-wersja-platforma.deb`

Realizacja RedHat (CentOS, Fedora):

`rpm -i program-wersja-platforma.rpm`



Instalacja oprogramowania

Metoda trzecia – **Kompilacja ze źródeł:**

korzystanie z oprogramowania udostępnianego przez twórców w postaci kodu źródłowego.

Zalety: dedykowana dla określonego systemu funkcjonalność

Wady: kłopotliwe zarządzanie oprogramowaniem, trudne spełnienie zależności.

Realizacja:

```
tar -xzf program-wersja.tar.gz
```

```
cd program-wersja
```

```
./configure
```

```
make
```

```
sudo make install
```



Konfiguracja usług

Wszystkie zainstalowane usługi posiadają skrypty startowe w katalogu `/etc/init.d`. Wszystkie skrypty realizują przynajmniej opcje start i stop. Dodatkowo często również: restart, reload, status itp..

Uruchamianie usługi:

`/etc/init.d/usługa start`

Zatrzymywanie usługi:

`/etc/init.d/usługa stop`

W najnowszych dystrybucjach Linuksa uruchamianie i zatrzymywanie usług można zrealizować poleceniem

`service usług start`

`service usług stop`



Konfiguracja usług

W celu automatycznego uruchamiania i zatrzymywania usług przy zmianach poziomu uruchomienia (runlevel) w katalogach /etc/rc0.d, /etc/rc1.d do /etc/rc6.d znajdują się linki symboliczne do skryptów w katalogu /etc/init.d.

Skrypt którego nazwa linku rozpoczyna się od dużej litery S jest uruchamiany, a skrypt, którego nazwa linku rozpoczyna się od dużej litery K jest zatrzymywany.

Zgoda na automatyczne uruchamianie usługi:

chkconfig usługa on (RedHat)

update-rc.d usługa defaults (Debian)

Wyłączenie automatycznego uruchamiania usługi:

chkconfig usługa off

update-rc.d -f usługa remove



Inicjacja systemu

Tryby pracy procesu init:

- 0 - zamknięcie systemu
- 1 - tryb pracy jednego użytkownika (root)
- 2-4 - tryby pracy wielu użytkowników bez trybu graficznego
- 5 - tryb pracy wielu użytkowników w środowisku X-Window
- 6 - restart systemu

Plik **/etc/inittab** zawiera konfigurację domyślnego trybu pracy systemu. Tryb domyślny można zmienić dopisując w menu startowym za nazwą jądra systemu odpowiedni numer.



Inicjacja systemu z wykorzystaniem systemd

W najnowszych dystrybucjach Linuksa coraz większą popularność zdobywa nowy zarządca uruchamiania usług systemowych - **systemd**. Do najważniejszych jego zalet należy zaliczyć bardziej precyzyjne niż w **init.d** określenie kolejności i zależności między usługami oraz możliwość uruchamiania usług równolegle co skraca czas uruchamiania systemu.

Przykłady użycia:

systemctl start usługa

systemctl stop usługa

systemctl reload usługa

systemctl enable usługa



Inicjacja systemu z wykorzystaniem systemd

Konfiguracja usług znajduje się w folderach:
/etc/systemd/system/ oraz **/lib/systemd/system/**

Przykładowa zawartość pliku `ssd.service`:

[Unit]

Description=OpenBSD Secure Shell server

After=network.target

[Service]

EnvironmentFile=-/etc/default/ssh

ExecStart=/usr/sbin/sshd -D \$SSHD_OPTS

ExecReload=/bin/kill -HUP \$MAINPID

KillMode=process

Restart=on-failure

RestartPreventExitStatus=255

Type=notify

[Install]

WantedBy=multi-user.target

Alias=sshd.service



Informacja o systemie i jego stanie

uname	- nazwa systemu
uname -a	- pełne informacje o systemie i wersji
df	- wolna przestrzeń dyskowa
du	- zajęta przestrzeń w katalogu
uptime	- czas od restartu systemu i statystyki obciążenia procesora
top	- lista działających procesów plus statystyki
lspci	- lista urządzeń PCI
lsusb	- lista urządzeń USB
lsmod	- lista załadowanych sterowników urządzeń

Informacji dostarczają również pliki z katalogu /proc np.:
/proc/cpuinfo, /proc/meminfo, /proc/devices itp.