ODPOWIEDŹ

W listingach co jest **pogrubione** to trzeba pokazać

Tym kolorem doprawiam Pablowe komentarze po środowym rozliczeniu tego rozdziału. Polecam korzystać z odpowiedzi a nie tylko teorii żeby sie w listingach nie zawalić. I na Boga nie referowałem tego jak Adi więc przeczytajcie wcześniej chociaż raz i bądźcie gotowi parafrazować.

4. **Do czego służy klasyfikacja procesów na ograniczone procesorem i ograniczone wejściem wyjściem oraz na jakiej podstawie zweryfikowano, że proces jest ograniczony wejściem-wyjściem? - weryfikacja przygotowanych w listingach obliczeń i wskazanie w nich wirtualnego czasu życia procesu realizującego odczyty z dysku /dev/sda.**

   ODPOWIEDZ:

   pokazanie że jednowątkowy (dd z natury jest jednowątkowy), czas wirtualny - ile faktycznie korzystal z jednostki przetwarzajacej czas wykonywany

   - Podstawowe stany wątku procesu to: **Zablokowany, Gotowy do wykonania, Wykonywany.**
   - **Proces ograniczony wejściem/wyjściem to taki, który przebywa większość czasu w stanie zablokowanym (ponad 50%).**
   - **Klasyfikacja procesów służy temu**, aby wiedzieć, jak usprawnić przetwarzanie zadań przez proces.
   - **Jak można usprawnić** przetwarzanie przez proces ograniczony wejściem/wyjściem? **Poprawić warunki realizacji operacji dyskowej. (np. mieć lepszy dysk, zmienić algorytm windy; użyć ionice).**
   - **Przełączenie kontekstu - odebranie jednostki przetwarzającej** procesowi i **przydzielenie jej innemu procesowi.**

   OPCJONALNIE:

   - Pod jakim warunkiem można usprawnić przetwarzanie? Że proces zgłasza operacje dyskowe.
   - Czy każdą operację można usprawnić? Nie każdą, ale operacje dyskowe można.
   - Na co czeka proces w stanie zablokowanym? Na pozostałe zasoby systemowe oprócz stron pamięci.
   - Co jeżeli nie będzie równo 50%? To wtedy jeszcze nie jest ograniczony.

   LISTING:

```
[root@254813 ttyid:0 czw maj 08 14:03:03 ~]# cat /sys/block/sda/queue/scheduler
```

none mq-deadline kyber **[bfq]**
[root@254813 ttyid:0 czw maj 08 14:03:39 ~]# time perf sched record ionice -c 1 -n 0 dd if=/dev/sda of=/dev/null bs=1G count=25
Trzeba jeszcze wykazać proces (dd) że jest jednowątkowy macie na komendach
25+0 records in
25+0 records out
26843545600 bytes (27 GB, 25 GiB) copied, 24,4098 s, 1,1 GB/s
[perf record: Woken up 1 times to write data]
[perf record: Captured and wrote 8,460 MB perf.data (69855 samples)]
real **0m26,879s**
user 0m0,187s
sys 0m12,326s

[root@254813 ttyid:0 czw maj 08 14:04:54 ~]# perf sched latency -p | grep -E "Task | dd"

Task	Runtime ms	Count	Avg delay ms	max delay ms	max delay start	max delay end
dd:60284	**11776.880 ms**	**3550**	avg: **0.024 ms**	max: 1.787 ms	max start: 119257.717469 s	max end: 119257.719256 s
Proces	**Czas w stanie wykonywanym (wykonanie i gotowość do wykonania)**	**Ilość przełączeń kontekstu**	średni czas oczekiwania na jednostkę przetwarzającą	najdłuższy czas oczekiwania procesu w stanie "gotowości" (ready) zanim dostał CPU.	Czas (w sekundach od uruchomienia systemu), kiedy zaczął się najdłuższy okres oczekiwania na CPU.	Czas (w sekundach od uruchomienia systemu), kiedy kończył się najdłuższy okres oczekiwania.

Task | Runtime ms | Count | Avg delay ms | Max delay ms | Max delay start | Max delay end |
dd:60284 | **11776.880 ms** | **3550** | avg: **0.024 ms** | max: 1.787 ms | max start: 119257.717469 s | max end: 119257.719256 s
[root@254813 ttyid:0 czw maj 08 14:05:19 ~]# echo "Real = "$(echo "**26.879***1000" | bc) "ms"
Real = **26879.000** ms
[root@254813 ttyid:0 czw maj 08 14:10:30 ~]# echo "Runtime = "$(echo "**11776.880**" | bc) "ms"
Runtime = 11776.880 ms
[root@254813 ttyid:0 czw maj 08 14:10:36 ~]# echo "Czas w stanie zablokowanym = "$(echo "**26879.000**-**11776.880**-(**3550***0.024)" |bc) "ms"
Czas w stanie zablokowanym = **15016.920** ms
[root@254813 ttyid:0 czw maj 08 14:12:25 ~]# echo "Procent swojego życia w jakim proces był w stanie zablokowanym = "$(echo "**15016.920***100/**26879.000**" | bc) "%"
Procent swojego życia w jakim proces był w stanie zablokowanym = 55 %

Aby obliczyć czas zablokowany musimy od czasu rzeczywistego życia procesu odjąć czas w stanie wykonywanym oraz iloczyn przełączeń kontekstu i średni czas oczekiwania na jednostkę przetwarzającą (czas wirtualny).

A żeby obliczyć procentowy udział blocked w życiu procesu dzielimy czas zablokowany przez czas rzeczywisty, pomnożony przez sto.

Ważna rzecz nie wykażecie tego ponad 50% - mówicie, że oVirt jest aktualnie tak wydajny (mówię to z bólem serca) że ciężko jest to osiągnąć, ale spowiadacie się w zamian z teorii.

5. **Co reprezentuje wartość %cpu dla procesu jednowątkowego? - dla jakiego procesu wartość %cpu może przekroczyć 100%? - wymagane zaprezentowanie %cpu dla procesu jednowątkowego w listingu.**

   ODPOWIEDŹ:

   - **%CPU jest to stosunek** wirtualnego czasu życia procesu do rzeczywistego czasu życia procesu.
   - Wirtualny czas życia procesu - To **suma czasu CPU**, jaki proces aktywnie zużył – czyli tylko wtedy, gdy naprawdę wykonywał operacje na CPU. **Obejmuje sys time, user time. NIE uwzględnia** czasu zablokowanego.
   - Może przekroczyć 100% jeśli sumaryczny czas wirtualny wątków procesu jest większy od rzeczywistego czasu życia procesu, **w przypadku procesów wielowątkowych**.

   LISTING:

```
[sysop@fedora ttyid:0 śro maj 14 20:59:04 ~]$ dd if=/dev/zero of=/dev/null bs=1M
count=500000 &
[1] 3652
[sysop@fedora ttyid:0 śro maj 14 20:59:55 ~]$ pid_dd=$!
[sysop@fedora ttyid:0 śro maj 14 20:59:57 ~]$ ps -o pcpu,pmem,nlwp,pid,cmd,time,etime -p
$pid_dd
%CPU %MEM NLWP     PID CMD                         TIME     ELAPSED
52.1  0.0    1    3652 dd if=/dev/zero of=/dev/nul 00:00:04       00:08
[sysop@fedora ttyid:0 śro maj 14 21:00:04 ~]$ kill $pid_dd
[1]+  Zakończony              dd if=/dev/zero of=/dev/null bs=1M count=500000
[sysop@fedora ttyid:0 śro maj 14 21:00:13 ~]$ echo "%CPU = time / etime * 100 = 50%"
```

Tutaj też nie wykażecie mniejszego %cpu niż 100 - ponownie wina wydajności oVirta.

Żeby obliczyć procentowy udział CPU w procesie, dzielimy czas wirtualny życia procesu (runtime) przez rzeczywisty czas procesu (elapsed) pomnożony przez 100. <-To trzeba mieć obliczone.

```
%CPU = time / etime * 100 = 50%
[sysop@fedora ttyid:0 śro maj 14 21:02:59 ~]$
[sysop@fedora ttyid:0 śro maj 14 21:02:59 ~]$ (sysbench cpu --threads=5 --time=90 run) &
sleep 5 && ps -o pid,%cpu $(pidof sysbench)
[1] 4141
sysbench 1.0.20 (using system LuaJIT 2.1.1713773202)
Running the test with following options:
Number of threads: 5
Initializing random number generator from current time
Prime numbers limit: 10000
Initializing worker threads...
Threads started!
    PID %CPU
   4141  251
[sysop@fedora ttyid:0 śro maj 14 21:03:34 ~]$ kill 4141
[1]+  Zakończony              ( sysbench cpu --threads=5 --time=90 run )
```

6. **Co reprezentuje wartość %mem dla procesu? - wymagane wykazanie w listingach wartości, na bazie których %mem jest wyliczany, czy wartość %mem może przekroczyć 100%? - wymagane zaprezentowanie %mem dla procesu jednowątkowego w listingu.**

ODPOWIEDŹ:

- %mem to **stosunek fizycznej pamięci ulotnej procesu** do **fizycznej pamięci ulotnej systemu operacyjnego.**
- licznik **pmap -x** -> RSS
- mianownik **free -w** -> mem total.
- nie może przekroczyć 100%.

LISTING:

```
[sysop@fedora ttyid:0 śro maj 14 21:08:09 ~]$ dd if=/dev/zero of=/dev/null bs=500M count=500000 &
[1] 4641
[sysop@fedora ttyid:0 śro maj 14 21:08:29 ~]$ pid_dd=$!
[sysop@fedora ttyid:0 śro maj 14 21:08:31 ~]$ ps -o pmem,rss,nlwp,pid,cmd -p $pid_dd
%MEM   RSS NLWP     PID CMD
 6.3 514052    1    4641 dd if=/dev/zero of=/dev/null bs=500M count=500000
[sysop@fedora ttyid:0 śro maj 14 21:08:37 ~]$ free -w
               total        used        free      shared     buffers       cache   available
Mem:         8088064     2494236     4524736       11788        7572     1422628     5593828
Swap:        8087548           0     8087548
[sysop@fedora ttyid:0 śro maj 14 21:08:43 ~]$ kill $pid_dd
```

Tutaj również wymagane jest obliczenie tego.

7. **[9] Który z segmentów procesu zawiera stertę, a który stos i czy są one segmentami anonimowymi? - wymagane wskazanie segmentu sterty i stosu w mapie pamięci wybranego procesu.**

ODPOWIEDŹ:

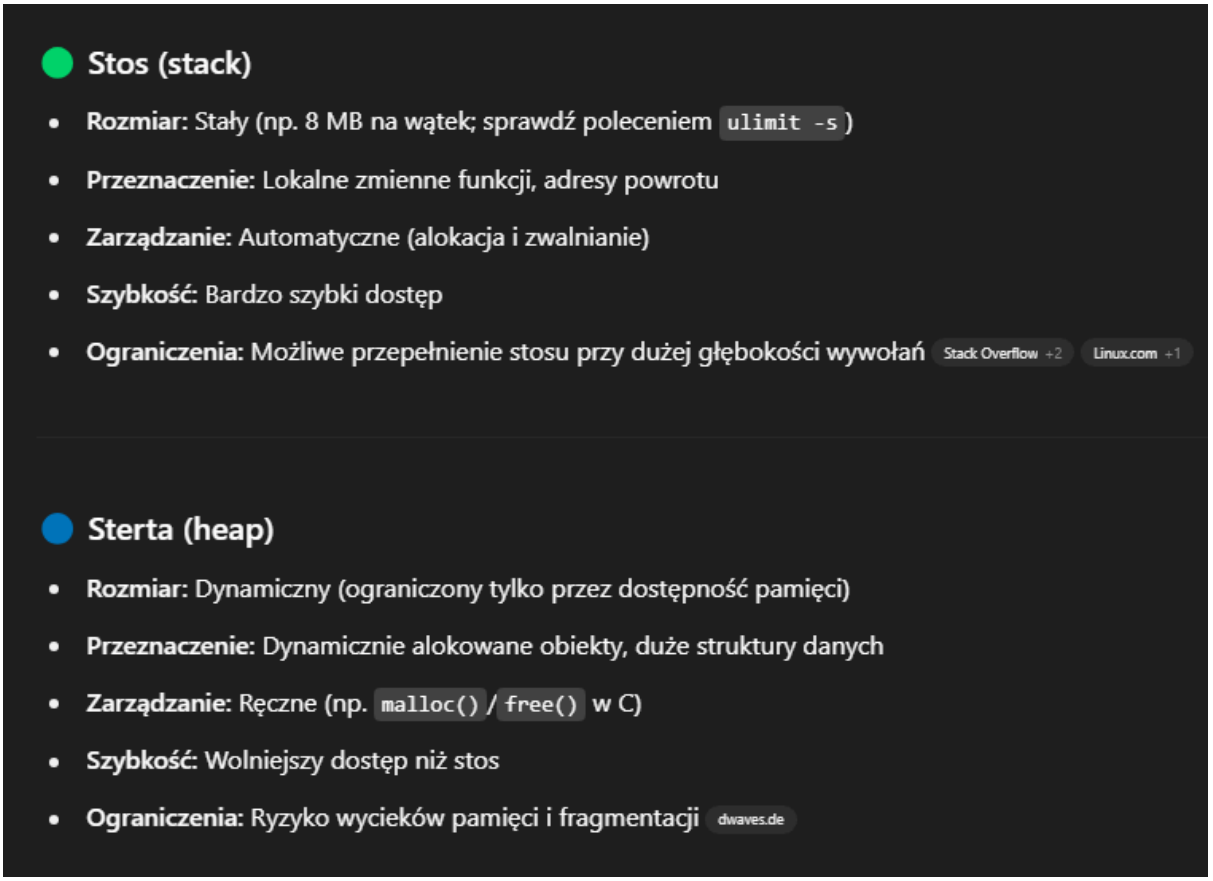
- **sterta - heap**; kolejność działania FIFO
- **stos - stack**; kolejność działania LIFO

- **oba są anonimowe**
    - anonimowość oznacza, że segment nie jest powiązany z konkretnym plikiem
- sterta zawiera dane
- **Różnica** między stosem a stertą. Stos ma **stały rozmiar**, a sterta **się dynamicznie zmienia.**
- Pamięć na stertę jest dynamicznie przyznawana procesowi

Jak odpowiadałem wystarczyło powiedzieć to co podkreśliłem.

Co to jest segment?

Fragment pamieci adresowej pamieci wirtualnej



LISTING:

```
[sysop@fedora ttyid:0 śro maj 14 21:19:55 ~]$ dd if=/dev/zero of=/dev/null bs=1M count=500000 &
[1] 5418
[sysop@fedora ttyid:0 śro maj 14 21:20:02 ~]$ pid=$!
[sysop@fedora ttyid:0 śro maj 14 21:20:04 ~]$ pmap -X $pid | egrep '\[heap\]|\[stack\]'
```

```
   555b99f9d000 rw-p 00000000 00:00      0   132   16   16       16       16
16   0       0          0          0          0     0    0     0        0
[heap]
   7ffe6a65f000 rw-p 00000000 00:00      0   132   16   16       16       16
16   0       0          0          0          0     0    0     0        0
[stack]
[sysop@fedora ttyid:0 śro maj 14 21:20:09 ~]$ kill $pid
[1]+  Zakończony              dd if=/dev/zero of=/dev/null bs=1M count=500000
```

Pole `pmap -X`	Wartość	Co znaczy?
`Dev`	`00:00`	brak urządzenia/plikowego urządzenia blokowego
`Inode`	`0`	nie ma powiązania z żadnym i-node (żadnym plikiem)
`Mapping`	`[heap]` / `[stack]`	podkreśla, że to wewnętrzny, prywatny obszar pamięci

Device, Inode

Tu starczy powiedzieć że nie wskazuje ani na żaden Device ani Inode czyli jest anonimowy zaznaczając odpowiednie zera kursorem.

8. **[10] Czy całość jego kodu i danych jest wczytywana z pliku programu ELF oraz plików bibliotek ELF do fizycznej pamięci ulotnej? - wymagane zaprezentowania odpowiednich wartości dla przykładowego nieanonimowego segmentu danych i segmentów tekstu w listingu z mapą pamięci procesu jednowątkowego, gdzie nie wszystkie strony segmentu są załadowane do ramek.**

   ODPOWIEDŹ:

   - **Czy całość jego kodu i danych jest wczytywana z pliku programu ELF oraz plików bibliotek ELF do fizycznej pamięci ulotnej?** Nie, część kodu/danych pochodzi z segmentów anonimowych.
   - **Wskaż dwa segmenty nieanonimowe, z których jeden zawiera dane, drugi tekst,** w których nie jest to załadowane do pamięci. (Dane r—- lub rw—- | tekst r-x—)
   - **Strona niezabrudzona** jest niezmodyfikowana od czasu załadowania z dysku
   - **Strona zabrudzona** zawiera zmodyfikowane dane, które nie zostały jeszcze zapisane na dysk
   - **Strona anonimowa** niepowiązana z żadnym plikiem na dysku
   - **Strona nieanonimowa** powiązana z konkretnym plikiem na dysku

   OPCJONALNIE:

- Z jakich elementów składa się przestrzeń adresowa procesu? Z segmentów pamięci. Przechowują one dane, kod/tekst, stertę, stos
- Skąd wiadomo, że całość np. tekstu nie jest załadowana do pamięci ulotnej? Pamięć wirtualna (Kbytes, łączny rozmiar stron) jest większa od fizycznej (RSS, łączny rozmiar ramek

LISTING:

```
[root@254813 ttyid:0 czw maj 08 18:54:34 ~]# dd if=/dev/zero of=/dev/null bs=1M count=300000 &
[1] 71526
[root@254813 ttyid:0 czw maj 08 18:54:40 ~]# pid=$!
[root@254813 ttyid:0 czw maj 08 18:54:42 ~]# pmap -x $pid | grep 'libc.so.6' | grep -E ' r-(x|--)'
00007f5be5ca6000       4       4             0 r---- libc.so.6
00007f5be5ca7000       1468     972       0 r-x-- libc.so.6
00007f5be5e16000       468     340         0 r---- libc.so.6
00007f5be5e8b000       16      16           16 r---- libc.so.6
[root@254813 ttyid:0 czw maj 08 18:54:44 ~]# kill 71526
[1]+  Zakończony          dd if=/dev/zero of=/dev/null bs=1M count=300000
```

9. **[12] Jaki rozmiar zajmują wszystkie strony procesu a jaki wszystkie przydzielone dla procesu ramki oraz jak nazywa się sytuacja, gdy potrzebna strona nie znajduje się w ramce procesu (należy wymienić możliwe warianty tej sytuacji oraz gdzie system będzie szukał potrzebnej strony)? - wymagane wskazanie odpowiednich wartości w listingach.**

   TEORIA:

   - Kolumny przedstawiają łączny rozmiar stron i ramek.
   - Zależność stron i ramek Jeśli jest więcej stron niż ramek, nie każda strona będzie mogła być przechowywana w ramce więc całość tekstu/danych nie będzie przechowywana w fizycznej pamięci ulotnej. Jeśli jest tyle samo stron i ramek to dane będą wczytywane w całości.
   - **Błąd strony** - próba wykonania operacji odczytu/zapisu na zawartości strony nieprzechowywanej w zbiorze roboczym procesu.
   - **Błąd poboczny** - strona jest w ramce ale nie występuje w zbiorze roboczym procesu (znajduje sie w swap cache), wtedy strony takie sa nieaktywne
   - **Błąd główny** - strona nie jest w ramce, strona jest w pliku regularnym albo swapie.
   - W ramie jest **swap cache.**
   - Na co dzieli się segment pamięci wirtualnej? Na ramki (podstawowa jednostka alokacji w pamięci operacyjnej-fizycznej)
   - Jakie strony procesu są załadowane do ramki? Tylko te potrzebne

- Czy może być więcej ramek niż stron? Nie, gdyby było ich więcej to w niektórych ramkach byłyby kopie stron
- Poszukiwania strony: swap cache -> swap -> plik programu/biblioteki

Zbiór ramek procesu to zbiór roboczy procesu.
LISTING:

```
[sysop@fedora ttyid:0 śro maj 14 21:41:40 ~]$ dd if=/dev/zero of=/dev/null
bs=1M count=300000 &
[1] 6418
[sysop@fedora ttyid:0 śro maj 14 21:42:38 ~]$ pid=$!
[sysop@fedora ttyid:0 śro maj 14 21:42:39 ~]$ pmap -x "$pid" | tail -1
total kB         230932    3428    1140
[sysop@fedora ttyid:0 śro maj 14 21:42:45 ~]$ ps -o pid,min_flt,maj_flt -p
"$pid"
    PID  MINFL  MAJFL
   6418   946      0
[sysop@fedora ttyid:0 śro maj 14 21:42:50 ~]$ 300000+0 records in
300000+0 records out
314572800000 bytes (315 GB, 293 GiB) copied, 15,323 s, 20,5 GB/s

[1]+  Zakończono              dd if=/dev/zero of=/dev/null bs=1M count=300000
[sysop@fedora ttyid:0 śro maj 14 21:42:56 ~]$ free -w | awk '/Mem:/ {print
"swap-cache:", $7, "kB"}'
swap-cache: 1471448 kB
```

10. **[20] Jeżeli plik zawierający program/bibliotekę ELF zostanie usunięty, to skąd system operacyjny będzie doczytywał potrzebne strony, jaki jest związek pomiędzy niezwolnionymi blokami systemu plików a stronami? - omówienie statystyk węzłów i bloków systemu plików, aby stwierdzić czy po usunięciu pliku zostały one zwolnione wraz z usuniętym wpisem katalogowym.**

    TEORIA:

    - **Czy jeżeli plik programu zawierający stronę zostanie usunięty, to czy system operacyjny może wczytać tę stronę?**
        - Tak, z przestrzeni wymiany swap
        - **Brakujące strony kernel doczyta z tych samych bloków dysku, na które wciąż wskazuje i-węzeł pliku. Po usunięciu wpisu katalogowego inode i bloki nie są zwalniane, więc kernel ma skąd wczytać stronę. Swap zostanie użyty dopiero wtedy, gdy ta strona była anonimowa lub zabrudzona i wcześniej wyleciała z RAM-u.**
        - **Strona niezabrudzona** to taka która nie zmieniła zawartości od momentu wczytania do ramki

    - **Strona zabrudzona** zawiera zmodyfikowane dane, które nie zostały jeszcze zapisane na dysk
    - **Strona anonimowa** niepowiązana z żadnym plikiem na dysku
    - **Strona nieanonimowa** powiązana z konkretnym plikiem na dysku - dodaję tutaj to na razie ~Emka

    - **Swap cache** - nieciągły obszar pamięci operacyjnej złożonych z ramek zawierających strony nieaktywne.
- **Co to jest przestrzeń wymiany swap? Jest to obszar pamięci drugiego rzędu, w którym jądro przechowuje strony pamięci wirtualnej wysunięte z RAM-u. Pełni funkcje magazynu stron dla pamięci wirtualnej.** Może składać się z wielu magazynów stron. **Rodzaje magazynów stron:** partycja wymiany, plik wymiany, obszar wymiany. **Narzędzie *swapon*** dodaje magazyn stron, **natomiast *swapoff*** usuwa magazyn stron z przestrzeni wymiany.
- Jaki jest sens wymiatania z pamięci do pamięci? Taki jest sens, że strony trafiają do skompresowanej pamięci ulotnej, dzięki czemu odzyskujemy wolne ramki bez kosztownego zapisu na dysk.
- **Jaki jest związek pomiędzy niezwolnionymi blokami systemu plików a stronami? Bloki i węzły NIE** zostaną zwolnione dopóki nie zakończymy procesu.
- Usuwając plik usuwany jest **wpis katalogowy**
- Mechanizm pamięci wirtualnej blokuje zwolnienie tych bloków i węzłów
- Bloki i węzły zostaną zwolnione kiedy wszystkie procesy korzystające z tego pliku zostaną zakończone
- Zwolnienie jest blokowane bo system operacyjny musi mieć możliwość wczytywania tych stron do segmentu nieanonimowego
- Plik ma ***JEDEN*** węzeł i chuj
- standardowy blok systemu plików (4096 B) = 1 strona pamięci (4 KiB)

LISTING:

```
[sysop@fedora ttyid:0 śro maj 14 21:59:16 Dokumenty]$ cp /bin/dd ./mydd
[sysop@fedora ttyid:0 śro maj 14 21:59:20 Dokumenty]$ stat -c 'File: %n
Size=%s B  Blocks=%b (po %B B)' ./mydd
File: ./mydd  Size=66008 B  Blocks=136 (po 512 B)
[sysop@fedora ttyid:0 śro maj 14 21:59:36 Dokumenty]$ ./mydd if=/dev/zero
of=/dev/null bs=25M count=100000 &
[1] 7518
[sysop@fedora ttyid:0 śro maj 14 22:00:01 Dokumenty]$ pid=$!
[sysop@fedora ttyid:0 śro maj 14 22:00:02 Dokumenty]$ stat -f ./
  Plik: "./"
    ID: dd47b0d54e281a54 długość nazwy: 255     typ: ext2/ext3
```

```
rozmiar bloku: 4096       podstawowy rozmiar bloku: 4096
bloków: Razem: 755402     wolnych: 711492     dostępnych: 668075
Inody: razem: 196608      wolnych: 194206
[sysop@fedora ttyid:0 śro maj 14 22:00:07 Dokumenty]$ rm ./mydd
[sysop@fedora ttyid:0 śro maj 14 22:00:10 Dokumenty]$ stat -f ./
  Plik: "./"
    ID: dd47b0d54e281a54 długość nazwy: 255     typ: ext2/ext3
rozmiar bloku: 4096       podstawowy rozmiar bloku: 4096
bloków: Razem: 755402     wolnych: 711492     dostępnych: 668075
Inody: razem: 196608      wolnych: 194206
[sysop@fedora ttyid:0 śro maj 14 22:00:12 Dokumenty]$ kill $pid
[sysop@fedora ttyid:0 śro maj 14 22:00:16 Dokumenty]$
[1]+  Zakończony              ./mydd if=/dev/zero of=/dev/null bs=25M
count=100000
[sysop@fedora ttyid:0 śro maj 14 22:00:19 Dokumenty]$ stat -f ./
  Plik: "./"
    ID: dd47b0d54e281a54 długość nazwy: 255     typ: ext2/ext3
rozmiar bloku: 4096       podstawowy rozmiar bloku: 4096
bloków: Razem: 755402     wolnych: 711509     dostępnych: 668092
Inody: razem: 196608      wolnych: 194207
[sysop@fedora ttyid:0 śro maj 14 22:07:33 Dokumenty]$ echo "Blocks =
"$(echo "136/(4096/512)" | bc)" "
Blocks = 17
```

11. **[21] Z jakiego obszaru pamięci ulotnej wybierane są niepuste ramki i jakie kryteria musi spełniać ramka wybrana przez algorytm wymiany? - analiza wyników free –w dotycząca różnicy w wartościach pomiędzy kolumną free i available w wierszu mem.**

    TEORIA:

    - Niepuste ramki są z swap cache. Ramka wybrana przez algorytm wymiany musi być niezabrudzona i nieanonimowa.
    - Swap cache - nieciągły obszar pamięci operacyjnej złożonych z ramek zawierających strony nieaktywne.
    - Ramki niepuste są ze swap cache, to ramki ze stronami nieaktywnymi, niezabrudzonymi i nieanonimowymi. W swap cache każda strona jest nieaktywna

    - Jeżeli takich ramek nie ma to:
      - Będzie zrzucał do przestrzeni swap strony zabrudzone i anonimowe
      - Będzie odbierał ramki procesom (niebezpieczne)
      - OOM
    - **Kryteria**:

    - **niezabrudzona** - to taka która nie zmieniła zawartości od momentu wczytania do ramki
    - **nieanonimowa** - to taka której zawartość została wczytana z pliku regularnego z programem lub biblioteką
    - **nieaktywna** - to taka która nie była używana przez proces przez jakiś czas
- **Free w listingu** free -w to rozmiar wolnych i pustych ramek
- **Available** w listingu free -w to rozmiar pustych ramek i niepustych ze swap cache i ramki ze stronami niezabrudzonymi z buforów. (Łączny rozmiar dostępnej pamięci operacyjnej)
- zram - skompresowanej pamięci ulotnej
- Bufor to fragmenty otwartych plików gdzie są przechowywane dane

Opcjonalne:

- Strona - najmniejsza jednostka pamięci wirtualnej przydzielana procesowi
- Ramka - najmniejsza jednostka pamięci fizycznej w RAM (odpowiada rozmiarowi strony)
- RAM jest dzielony na ramki, które mogą być przydzielane procesom jako miejsce na ich strony.

LISTING:

```
[root@254813 ttyid:0 czw maj 08 20:41:14 ~]# free -w
              total        used        free      shared     buffers       cache   available
Mem:        8088056     2263928     1438756       12412     3598380     1236488     5824128
Swap:       8087548        1016     8086532
[root@254813 ttyid:0 czw maj 08 20:41:14 ~]# swapon Ja mu nie pokazywałem
NAME       TYPE       SIZE USED PRIO
/dev/zram0 partition 7,7G   0B  100
```

12. **[24] Jakie zasoby systemowe trzeba monitorować i jakie działania można podjąć aby procedura OOM nie była zastosowana? - wymagane zaprezentowanie odpowiednich wartości w listingach.**

    TEORIA:

    - Należy regularnie kontrolować stan dostępnej pamięci operacyjnej oraz ile mamy wykorzystanej pamięci swap. (free -w - wiersz swap)

- Możemy powiększyć przestrzeń wymiany poprzez dodanie pliku wymiany lub zakończyć procesy, które nie są nam już potrzebne (np. te, które zrealizowały swoje zadania).
- OOM (out of memory) można rozpoznać przez monitorowanie wartości available (im mniej tym gorzej; nigdy nie będzie 0) i kiedy swap used będzie zbliżało się do total
- Działania żeby nie było OOM+
    - zwiększenie przestrzeni wymiany poprzez dodanie pliku wymiany
    - szybciej zakończyć niepotrzebne procesy zanim system sam je zakończy
- Jak działa procedura OOM?
    - Procedura zakończy procesy, które są niepotrzebne bez wykonania przydzielonych im zadań
- Wybranie samemu procesów jest korzystniejsze bo można uniknąć utraty danych
- Procedurze OOM nigdy nie podlegaja procesy na prawach uzytkownika uprzywilejowanego
- Przed usunieciem procesu zapisz

OPCJONALNE:

- Czy powinno się instalować system bez przestrzeni wymiany? - Nie, to przyspieszy OOM
- Działania systemu podczas presji pamięci (gdy brakuje ramek)
    - Wymiatanie stron - zapisanie stanu w przestrzeni wymiany i zastępowanie stron pamięci nowymi (stare są wymiatane do przestrzeni wymiany)
    - Redukcja zbioru roboczego procesu - odebranie części przydzielonych procesom ramek, będzie powodować więcej błędów stron i przestojów
    - Procedura OOM - wybierane są procesy, które natychmiast zostaną zakończone, bez wykonania przydzielonych im zadań.
- polecenie wymuszające utrwalenie z buforów - sync

LISTING:

```
[root@254813 ttyid:0 czw maj 08 20:59:33 ~]# watch -n2 free -w
```

free -w
vmstat -s U mnie nie wymagał

jeśli krasiuk chciałby żebyśmy pokazali coś co zapobiega OOM to:

Ode mnie tego nie wymagał.
fallocate -l 2G /swapfile
chmod 600 /swapfile
mkswap /swapfile
swapon /swapfile

tutaj tworzony jest plik wymiany i od razu po tych komendach widać w free -w w swap total że jest więcej miejsca o 2GiB.

KOMENDY

4.

#T1
cat /sys/block/sda/queue/scheduler

time perf sched record ionice -c 1 -n 0 dd if=/dev/sda of=/dev/null bs=1G count=70 &

(UWAGA MIRKI, odpalacie tego ps aux i szukacie DOKŁADNIE tego dd co odpaliliście [dd if=/dev/sda of=/dev/null bs=1G count=70 &] tam lista będzie i bierzecie jego pid bo jak odpowiadałem z Łysiakiem to sie przyjebał że nie wykazujemy tego co trzeba)

ps aux | grep dd

ps -o pid,nlwp -p <PID>

perf sched latency -p | grep -E "Task | dd"

echo “real - runtime - (count * avg_delay)” | bc -l

echo "Czas w stanie zablokowanym = "$(echo "real*1000 - runtime - (count * avg_delay)" | bc) "ms"

echo “(^wynik wyzej^ / real) * 100” | bc -l

echo "Procent swojego życia w jakim proces był w stanie zablokowanym = "$(echo "^wynik wyzej^*100/(real*1000)" | bc) "%"

echo “czas wirtualny = (sys + user)” | bc -l

echo "czas wirtualny = $(echo "sys + user" | bc -l)"

5.

#T1
dd if=/dev/zero of=/dev/null bs=1M count=500000 &

pid_dd=$!

ps -o pcpu,pmem,nlwp,pid,cmd,time,etime -p $pid_dd

kill $pid_dd

echo "%CPU = time / etime * 100 = 50%"

echo "%CPU = $(echo "time / etime * 100" | bc -l)"

enter

enter

enter

(sysbench cpu --threads=5 --time=90 run) & sleep 5 && ps -o pid,%cpu $(pidof sysbench)

```
kill $pid_dd
```

6.

```
#T1
dd if=/dev/zero of=/dev/null bs=500M count=500000 &
pid_dd=$!
ps -o pmem,rss,nlwp,pid,cmd -p $pid_dd
free -w
kill $pid_dd
```

9.

```
#T1
dd if=/dev/zero of=/dev/null bs=1M count=500000 &
pid=$!
pmap -X $pid | egrep '\[heap\]|\[stack\]'
kill $pid
```

10.

```
#T1
dd if=/dev/zero of=/dev/null bs=1M count=3000000 &
pid=$!
pmap -x $pid | grep 'libc.so.6' | grep -E ' r-(x|--)'
kill $pid
```

12.

```
#T1

dd if=/dev/zero of=/dev/null bs=1M count=3000000 &

pid=$!

pmap -x "$pid" | tail -1

ps -o pid,min_flt,maj_flt -p "$pid"

free -w | awk '/Mem:/ {print "swap-cache:", $7, "kB"}'
```

20.

```
#T1

cd Dokumenty

cp /bin/dd ./mydd

stat -c 'File: %n  Size=%s B  Blocks=%b (po %B B)' ./mydd

./mydd if=/dev/zero of=/dev/null bs=25M count=100000 &

pid=$!


stat -f ./

rm ./mydd

stat -f ./

kill $pid

stat -f ./

echo "Zwolnionych bloków  = "$(echo "136/(4096/512)" | bc)" "
```

21.

```
#T1
```

free -w

24.

#T1

free -w

ewentualnie (vmstat -s)
ostatnio nie wymagał/\

TYLKO TEORIA (EASIER SCROLLING)

4. **Do czego służy klasyfikacja procesów na ograniczone procesorem i ograniczone wejściem wyjściem oraz na jakiej podstawie zweryfikowano, że proces jest ograniczony wejściem-wyjściem? - weryfikacja przygotowanych w listingach obliczeń i wskazanie w nich wirtualnego czasu życia procesu realizującego odczyty z dysku /dev/sda.**

   ODPOWIEDZ:

   - Podstawowe stany wątku procesu to: **Zablokowany, Gotowy do wykonania, Wykonywany.**
   - **Proces ograniczony wejściem/wyjściem to taki, który przebywa większość czasu w stanie zablokowanym (ponad 50%).**
   - **Klasyfikacja procesów służy temu**, aby wiedzieć, jak usprawnić przetwarzanie zadań przez proces.
   - **Jak można usprawnić** przetwarzanie przez proces ograniczony wejściem/wyjściem? **Poprawić warunki realizacji operacji dyskowej. (np. mieć lepszy dysk, zmienić algorytm windy; użyć ionice).**
   - **Przełączenie kontekstu - odebranie jednostki przetwarzającej** procesowi i **przydzielenie jej innemu procesowi.**

   OPCJONALNIE:

   - Pod jakim warunkiem można usprawnić przetwarzanie? Że proces zgłasza operacje dyskowe.
   - Czy każdą operację można usprawnić? Nie każdą, ale operacje dyskowe można.
   - Na co czeka proces w stanie zablokowanym? Na pozostałe zasoby systemowe oprócz stron pamięci.
   - Co jeżeli nie będzie równo 50%? To wtedy jeszcze nie jest ograniczony.

Aby obliczyć czas zablokowany musimy od czasu rzeczywistego życia procesu odjąć czas w stanie wykonywanym oraz iloczyn przełączeń kontekstu i średni czas oczekiwania na jednostkę przetwarzającą (czas wirtualny).

A żeby obliczyć procentowy udział blocked w życiu procesu dzielimy czas zablokowany przez czas rzeczywisty, pomnożony przez sto.

5. **Co reprezentuje wartość %cpu dla procesu jednowątkowego? - dla jakiego procesu wartość %cpu może przekroczyć 100%? - wymagane zaprezentowanie %cpu dla procesu jednowątkowego w listingu.**

ODPOWIEDŹ:

- **%CPU jest to stosunek** wirtualnego życia procesu do rzeczywistego życia procesu.
- Wirtualny czas życia procesu - To **suma czasu CPU**, jaki proces aktywnie zużył – czyli tylko wtedy, gdy naprawdę wykonywał operacje na CPU. **Obejmuje sys time, user time. NIE uwzględnia** czasu zablokowanego.
- Może przekroczyć 100% jeśli sumaryczny czas wirtualny wątków procesu jest większy od rzeczywistego czasu życia procesu, **w przypadku procesów wielowątkowych**.

6. Co reprezentuje wartość %mem dla procesu? - wymagane wykazanie w listingach wartości, na bazie których %mem jest wyliczany, czy wartość %mem może przekroczyć 100%? - wymagane zaprezentowanie %mem dla procesu jednowątkowego w listingu.

ODPOWIEDŹ:

- %mem to **stosunek fizycznej pamięci ulotnej procesu do fizycznej pamięci ulotnej systemu operacyjnego.**
- licznik **pmap -x** -> RSS
- mianownik **free -w** -> mem total.
- nie może przekroczyć 100%.

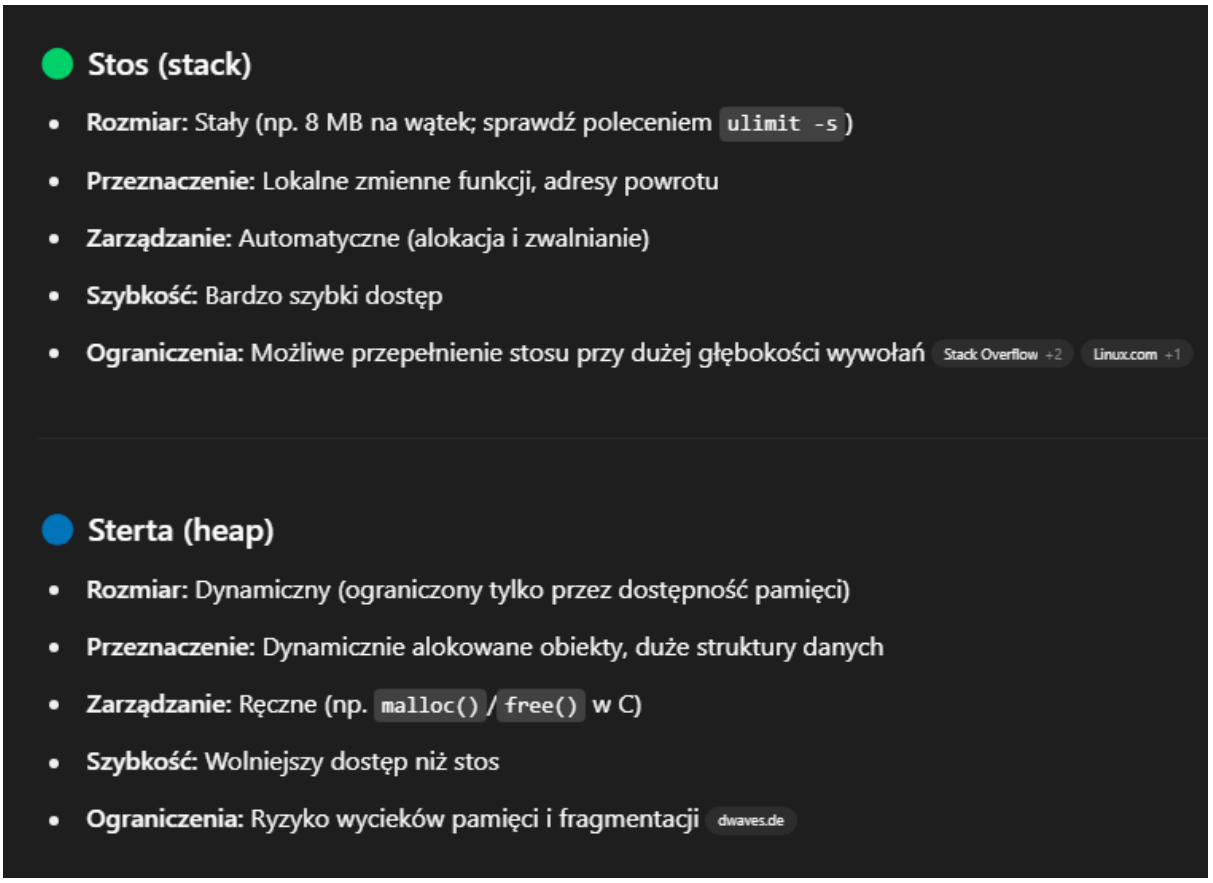
9. Który z segmentów procesu zawiera stertę, a który stos i czy są one segmentami anonimowymi? - wymagane wskazanie segmentu sterty i stosu w mapie pamięci wybranego procesu.

ODPOWIEDŹ:

segment to fragment pamięci wirtualnej (jednolity obszar pamięci wirtualnej)

- **sterta - heap**; kolejność działania FIFO
- **stos - stack**; kolejność działania LIFO
- **oba są anonimowe**
  - anonimowość oznacza, że segment nie jest powiązany z konkretnym procesem (może być używany przez wiele)
- sterta zawiera dane

- **Różnica** między stosem a stertą. Stos ma **stały rozmiar**, a sterta **się dynamicznie zmienia.**
- Pamięć na stertę jest dynamicznie przyznawana procesowi



10. Czy całość jego kodu i danych jest wczytywana z pliku programu ELF oraz plików bibliotek ELF do fizycznej pamięci ulotnej? - wymagane zaprezentowania odpowiednich wartości dla przykładowego nieanonimowego segmentu danych i segmentów tekstu w listingu z mapą pamięci procesu jednowątkowego, gdzie nie wszystkie strony segmentu są załadowane do ramek.

ODPOWIEDŹ:

- **Czy całość jego kodu i danych jest wczytywana z pliku programu ELF oraz plików bibliotek ELF do fizycznej pamięci ulotnej?** Nie, część kodu/danych pochodzi z segmentów anonimowych.
- **Wskaż dwa segmenty nieanonimowe, z których jeden zawiera dane, drugi tekst,** w których nie jest to załadowane do pamięci. (Dane r—- lub rw—- | tekst r-x—)
- **Strona niezabrudzona** jest niezmodyfikowana od czasu załadowania z dysku

- **Strona zabrudzona** zawiera zmodyfikowane dane, które nie zostały jeszcze zapisane na dysk
- **Strona anonimowa** niepowiązana z żadnym plikiem na dysku
- **Strona nieanonimowa** powiązana z konkretnym plikiem na dysku

OPCJONALNIE:

- Z jakich elementów składa się przestrzeń adresowa procesu? Z segmentów pamięci. Przechowują one dane, kod/tekst, stertę, stos
- Skąd wiadomo, że całość np. tekstu nie jest załadowana do pamięci ulotnej? Pamięć wirtualna (Kbytes, łączny rozmiar stron) jest większa od fizycznej (RSS, łączny rozmiar ramek

12. Jaki rozmiar zajmują wszystkie strony procesu a jaki wszystkie przydzielone dla procesu ramki oraz jak nazywa się sytuacja, gdy potrzebna strona nie znajduje się w ramce procesu (należy wymienić możliwe warianty tej sytuacji oraz gdzie system będzie szukał potrzebnej strony)? - wymagane wskazanie odpowiednich wartości w listingach.

łączny rozmiar stron i łączny rozmiar ramek

TEORIA:

- Parametry ilosc stron | ramek nazywaja sie: KBYTES to wszystkie strony, a RSS do ramek
- Kolumny przedstawiają łączny rozmiar stron i ramek.
- Zależność stron i ramek Jeśli jest więcej stron niż ramek, nie każda strona będzie mogła być przechowywana w ramce więc całość tekstu/danych nie będzie przechowywana w fizycznej pamięci ulotnej. Jeśli jest tyle samo stron i ramek to dane będą wczytywane w całości.
- **Błąd strony** - próba wykonania operacji odczytu zapisu na zawartości strony nieprzechowywanej w zbiorze roboczym procesu.
- **Błąd poboczny** - strona jest w ramce ale nie występuje w zbiorze roboczym procesu(w swap cache) (stony nieaktywne)
- **Błąd główny** - strona nie jest w ramce, ramka jest w pliku regularnym albo swapie.
- W ramie jest **swap cache.**
- **Pamiec swap w systemie znajduje sie w partycji ZRAM w pamieci RAM w nowoczesnym systemie, kiedys na dysku. Swap jest skopresowany**
- Na co dzieli się segment pamięci wirtualnej? Na ramki (podstawowa jednostka alokacji w pamięci operacyjnej-fizycznej)

- Jakie strony procesu są załadowane do ramki? Tylko te potrzebne
- Czy może być więcej ramek niż stron? Nie, gdyby było ich więcej to w niektórych ramkach byłyby kopie stron

Zbiór ramek procesu to zbiór roboczy procesu.

20. Jeżeli plik zawierający program/bibliotekę ELF zostanie usunięty, to skąd system operacyjny będzie doczytywał potrzebne strony, jaki jest związek pomiędzy niezwolnionymi blokami systemu plików a stronami? - omówienie statystyk węzłów i bloków systemu plików, aby stwierdzić czy po usunięciu pliku zostały one zwolnione wraz z usuniętym wpisem katalogowym.

TEORIA:

- **Czy jeżeli plik programu zawierający stronę zostanie usunięty, to czy system operacyjny może wczytać tę stronę?**
  - Tak, z przestrzeni wymiany swap
  - **Brakujące strony kernel doczyta z tych samych bloków dysku, na które wciąż wskazuje i-węzeł pliku. Po usunięciu wpisu katalogowego inode i bloki nie są zwalniane, więc kernel ma skąd wczytać stronę. Swap zostanie użyty dopiero wtedy, gdy ta strona była anonimowa lub zabrudzona i wcześniej wyleciała z RAM-u. (nwm czy to dobrze, ale raczej to nie jest po prostu swap)**
  - **Strona niezabrudzona** to taka która nie zmieniła zawartości od momentu wczytania do ramki
  - **Strona zabrudzona** zawiera zmodyfikowane dane, które nie zostały jeszcze zapisane na dysk
  - **Strona anonimowa** niepowiązana z żadnym plikiem na dysku
  - **Strona nieanonimowa** powiązana z konkretnym plikiem na dysku - dodaję tutaj to na razie ~Emka
- **Co to jest przestrzeń wymiany swap? Jest to obszar pamięci drugiego rzędu, w którym jądro przechowuje strony pamięci wirtualnej wysunięte z RAM-u. Pełni funkcje magazynu stron dla pamięci wirtualnej.** Może składać się z wielu magazynów stron. **Rodzaje magazynów stron:** partycja wymiany, plik wymiany, obszar wymiany. **Narzędzie *swapon*** dodaje magazyn stron, **natomiast *swapoff*** usuwa magazyn stron z przestrzeni wymiany.
- Jaki jest sens wymiatania z pamięci do pamięci? Taki jest sens, że strony trafiają do skompresowanej pamięci ulotnej, dzięki czemu odzyskujemy wolne ramki bez kosztownego zapisu na dysk.

- **Jaki jest związek pomiędzy niezwolnionymi blokami systemu plików a stronami? Bloki i węzły NIE** zostaną zwolnione dopóki nie zakończymy procesu.
- Usuwając plik usuwany jest **wpis katalogowy**
- Mechanizm pamięci wirtualnej blokuje zwolnienie tych bloków i węzłów
- Bloki i węzły zostaną zwolnione kiedy wszystkie procesy korzystające z tego pliku zostaną zakończone
- Zwolnienie jest blokowane bo system operacyjny musi mieć możliwość wczytywania tych stron do segmentu nieanonimowego
- Plik ma ***JEDEN*** węzeł
- standardowy blok systemu plików (4096 B) = 1 strona pamięci (4 KiB)

21. Z jakiego obszaru pamięci ulotnej wybierane są niepuste ramki i jakie kryteria musi spełniać ramka wybrana przez algorytm wymiany? - analiza wyników free –w dotycząca różnicy w wartościach pomiędzy kolumną free i available w wierszu mem.

TEORIA:

- Niepuste ramki są z przestrzeni wymiany (swap cache). Ramka wybrana przez algorytm wymiany musi być niezabrudzona i nieanonimowa, gdy skoncza sie strony spelniajace te warunki, system zacznie wymiatac do swapa strony, ktore nie spelniaja ww. warunkow anonimowe, zabrudzone i niekatywne
- Swap cache - nieciągły obszar pamięci operacyjnej złożonych z ramek zawierających strony nieaktywne.
- ogranicza zbior roboczy procesu

- Ramki niepuste są ze swap cache, to ramki ze stronami nieaktywnymi, niezabrudzonymi i nieanonimowymi. W swap cache każda strona jest nieaktywna.
- Kryteria
    - ramka nie może być używana przez żaden aktywny proces (nieaktywna)
    - ramka musi zawierać dane, które nie są potrzebne w najbliższej przyszłości
    - ramka musi być czysta (niezmodyfikowana) lub brudna (zmodyfikowana) w zależności od używanego algorytmu wymiany
    - nieanonimowa
- **Strona nieaktywna** to strona która nie była używana przez proces przez jakiś czas

- **Strona niezabrudzona** to taka która nie zmieniła zawartości od momentu wczytania do ramki
- **Strona nieanonimowa** to strona której zawartość została wczytana z pliku regularnego z programem lub biblioteką
- **Free w listingu** free -w to rozmiar wolnych i pustych ramek
- **Available** w listingu free -w to rozmiar pustych ramek i niepustych ze swap cache i ramki ze stronami niezabrudzonymi z buforów. (Łączny rozmiar dostępnej pamięci operacyjnej)
- Bufor to fragmenty otwartych plików gdzie są przechowywane dane

Opcjonalne:

- Strona - najmniejsza jednostka pamięci wirtualnej przydzielana procesowi
- Ramka - najmniejsza jednostka pamięci fizycznej w RAM (odpowiada rozmiarowi strony)
- RAM jest dzielony na ramki, które mogą być przydzielane procesom jako miejsce na ich strony.

24. Jakie zasoby systemowe trzeba monitorować i jakie działania można podjąć aby procedura OOM nie była zastosowana? - wymagane zaprezentowanie odpowiednich wartości w listingach.

TEORIA:

przed OOM najpierw usuwa nieużywane dane z cache potem wymiata strony do swapu a potem OOM dopiero

- Należy regularnie kontrolować stan dostępnej pamięci operacyjnej oraz ile mamy wykorzystanej pamięci swap. (free -w - kolumna swap)
- Możemy powiększyć przestrzeń wymiany poprzez dodanie pliku wymiany lub zakończyć procesy, które nie są nam już potrzebne (np. te, które zrealizowały swoje zadania).
- OOM (out of memory) można rozpoznać przez monitorowanie wartości available (im mniej tym gorzej; nigdy nie będzie 0) i kiedy swap used będzie zbliżało się do total
- Działania żeby nie było OOM
    - zwiększenie przestrzeni wymiany poprzez dodanie pliku wymiany
    -
- Jak działa procedura OOM?
    - Procedura zakończy procesy, które są niepotrzebne bez wykonania przydzielonych im zadań (nie wybiera procesów na prawach użytkownika uprzywilejowanego)

- Wybranie samemu procesów jest korzystniejsze bo można uniknąć utraty danych
- Przed wyjebaniem procesu zapisz

OPCJONALNE:

- Działania systemu podczas presji pamięci (gdy brakuje ramek)
    - Wymiatanie stron - zapisanie stanu w przestrzeni wymiany i zastępowanie stron pamięci nowymi (stare są wymiatane do przestrzeni wymiany)
    - Redukcja zbioru roboczego procesu - odebranie części przydzielonych procesom ramek, będzie powodować więcej błędów stron i przestojów
    - Procedura OOm - wybierane są procesy, które natychmiast zostaną zakończone, bez wykonania przydzielonych im zadań.
- polecenie wymuszające utrwalenie z buforów - sync