CSGE602055 Operating Systems
CSF2600505 Sistem Operasi
Week 05: Virtual Memory

Rahmat M. Samik-Ibrahim (ed.)

University of Indonesia

https://os.vlsm.org/Slides/os05.pdf
Always check for the latest revision!

REV345 11-Sep-2021

OS212[4]): Operating Systems 2021 - 2

OS A	OS B	OS C	OS INT
Every first day of the Week, **Quiz#1** and **Quiz#2**: 07:15-08:00			
Monday/Thursday 13:00 — 14:40 14:00 — finish	Monday/Thursday 15:00 — 16:40[1] 16:00 — finish	Monday/Thursday 13:00 — 14:40 13:00 — 14:40	Monday/Wednesday 08:00 — 09:40 09:00 — finish

Week	**Schedule & Deadline[2])**	**Topic**	**OSC10[3])**
Week 00	30 Aug - 05 Sep 2021	Overview 1, Virtualization & Scripting	Ch. 1, 2, 18.
Week 01	06 Sep - 12 Sep 2021	Overview 2, Virtualization & Scripting	Ch. 1, 2, 18.
Week 02	13 Sep - 19 Sep 2021	Security, Protection, Privacy, & C-language.	Ch. 16, 17.
Week 03	20 Sep - 26 Sep 2021	File System & FUSE	Ch. 13, 14, 15.
Week 04	27 Sep - 03 Oct 2021	Addressing, Shared Lib, & Pointer	Ch. 9.
Week 05	04 Oct - 10 Oct 2021	Virtual Memory	Ch. 10.
Week 06	11 Oct - 17 Oct 2021	Concurrency: Processes & Threads	Ch. 3, 4.
Week 07	01 Nov - 07 Nov 2021	Synchronization & Deadlock	Ch. 6, 7, 8.
Week 08	08 Nov - 14 Nov 2021	Scheduling + W06/W07	Ch. 5.
Week 09	15 Nov - 21 Nov 2021	Storage, Firmware, Bootloader, & Systemd	Ch. 11.
Week 10	22 Nov - 28 Nov 2021	I/O & Programming	Ch. 12.

[1]) **OS B:** Week00-Week05 (RMS); Week06-Week10 (MAM).

[2]) The **DEADLINE** of Week 00 is 05 Sep 2021, whereas the **DEADLINE** of Week 01 is 12 Sep 2021, and so on...

[3]) Silberschatz et. al.: **Operating System Concepts**, 10^{th} Edition, 2018.

[4]) This information will be on **EVERY** page two (2) of this course material.

STARTING POINT — https://os.vlsm.org/

- **Text Book** — Any recent/decent OS book. Eg. (**OSC10**) Silberschatz et. al.: **Operating System Concepts**, 10^{th} Edition, 2018. See also https://www.os-book.com/OS10/.
- **Resources**
  - **SCELE OS212** — https://scele.cs.ui.ac.id/course/view.php?id=3268. The enrollment key is **XXX**.
  - **Download Slides and Demos from GitHub.com** https://github.com/UI-FASILKOM-OS/SistemOperasi/: os00.pdf (W00), os01.pdf (W01), os02.pdf (W02), os03.pdf (W03), os04.pdf (W04), os05.pdf (W05), os06.pdf (W06), os07.pdf (W07), os08.pdf (W08), os09.pdf (W09), os10.pdf (W10).
  - **Problems** 195.pdf (W00), 196.pdf (W01), 197.pdf (W02), 198.pdf (W03), 199.pdf (W04), 200.pdf (W05), 201.pdf (W06), 202.pdf (W07), 203.pdf (W08), 204.pdf (W09), 205.pdf (W10).
  - **LFS** — http://www.linuxfromscratch.org/lfs/view/stable/
  - **OSP4DISS** — https://osp4diss.vlsm.org/
  - **DOIT** — https://doit.vlsm.org/001.html

Week 05: Memory

Week 05 Virtual Memory: Topics[1]

- Review of physical memory and memory management hardware
- Virtual Memory
- Caching
- Memory Allocation
- Memory Performance
- Working sets and thrashing

[1] Source: ACM IEEE CS Curricula 2013

Week 05 Virtual Memory: Learning Outcomes[1]

- Explain memory hierarchy and cost-performance trade-offs. [Familiarity]
- Summarize the principles of virtual memory as applied to caching and paging. [Familiarity]
- Describe the reason for and use of cache memory (performance and proximity, different dimension of how caches complicate isolation and VM abstraction). [Familiarity]
- Defend the different ways of allocating memory to tasks, citing the relative merits of each. [Assessment]
- Evaluate the trade-offs in terms of memory size (main memory, cache memory, auxiliary memory) and processor speed. [Assessment]
- Discuss the concept of thrashing, both in terms of the reasons it occurs and the techniques used to recognize and manage the problem. [Familiarity]

[1] Source: ACM IEEE CS Curricula 2013

Virtual Memory

- Reference: (OSC10-ch10 demo-w05)
- Virtual Memory: Separation Logical from Physical.
- Virtual Address Space: logical view.
- Demand Paging
- Page Flags: Valid / Invalid
- Page Fault
- Demand Paging Performance
- Copy On Write (COW)
- Page Replacement Algorithm
  - Reference String
  - First-In-First-Out (FIFO)
  - Belady Anomaly
  - Optimal Algorithm
  - Least Recently Used (LRU)
  - LRU Implementation
  - Lease Frequently Used (LFU)
  - Most Frequently Used (MFU)

Memory Allocation Algorithm

- Page-Buffering Algorithms
- Allocation of Frames
- Fixed Allocation
- Priority Allocation
- Global vs. Local Allocation
- Non-Uniform Memory Access (NUMA)
- Thrashing
- Working-Set Model
- Shared Memory via Memory-Mapped I/O
- Kernel
  - Buddy System Allocator
  - Slab Allocator

TOP: Table of Processes (12-memory.c) (01)

See also https://osp4diss.vlsm.org/osp-101.html

```
/*
 * Copyright (C) 2016-2021 Rahmat M. Samik-Ibrahim
 * http://rahmatm.samik-ibrahim.vlsm.org/
 * This program is free script/software. This program is distributed in the
 * hope that it will be useful, but WITHOUT ANY WARRANTY; without even the
 * implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
# INFO: TOP (Table of Processes)
 * REV11 Tue 30 Mar 18:25:50 WIB 2021
 * REV07 Fri 26 Mar 22:52:06 WIB 2021
 * REV06 Thu 25 Mar 13:52:59 WIB 2021
 * REV05 Wed 27 Feb 19:16:52 WIB 2019
 * REV04 Mon 12 Mar 17:33:30 WIB 2018
 * START Mon 03 Oct 09:26:51 WIB 2016
 */

#define TOKEN  "OS212W05"
#define MSTARTS 0x125E4
// #define MSTARTS 0x2BE5
// #define MSTARTS 0xFE4
// #define MSTARTS 0x3E4
// #define MSTARTS 0x1E4

#define MSIZE14 0x40609
#define MSIZE13 0x40609
#define MSIZE12 0x40608
#define MSIZE11 0x40608
#define MSIZE10 0x20FE8
#define MSIZE09 0x20FE8
#define MSIZE08 0x1F609
```

TOP: Table of Processes (12-memory.c) (02)

```
#define MSIZE07 0x1F609
#define MSIZE06 0x1F608
#define MSIZE05 0x1F608
#define MSIZE04 0x1E609
#define MSIZE03 0x1E609
#define MSIZE02 0x1E609
#define MSIZE01 0x1E608
#define MSIZE00 0x1E608
#define LINE   75
#define MAXSTR 80
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>

typedef  unsigned char* uChrPtr;
void     chktoken (uChrPtr token);

void printLine(int line) {
   while(line-- > 0) putchar('x');
   putchar('\n');
   fflush(NULL);
}

uChrPtr GlobalChar[MSTARTS];
```

TOP: Table of Processes (12-memory.c) (03)

```
void main(void) {
   int   msize[] = {MSIZE00, MSIZE01, MSIZE02, MSIZE03, MSIZE04,
                    MSIZE05, MSIZE06, MSIZE07, MSIZE08, MSIZE09,
                    MSIZE10, MSIZE11, MSIZE12, MSIZE13, MSIZE14};
   int   ii, jj;
   int   myPID    = (int) getpid();
   char  strSYS1[MAXSTR], strOUT[MAXSTR];
   char* chrPTR;
   char* chrStr;

   printLine(LINE);
   printf("ZCZC chktoken\n");
   chktoken(TOKEN);
   printLine(LINE);

   sprintf(strSYS1, "top -b -n 1 -p%d | tail -5", myPID);
   system (strSYS1);
   sprintf(strSYS1, "top -b -n 1 -p%d | tail -1", myPID);
   printf("PART 1\n");
   printLine(LINE);
   for (ii=0; ii < (sizeof(msize)/sizeof(int)); ii++){
      chrStr = malloc(msize[ii]);
      FILE* filePtr=popen(strSYS1, "r");
      fgets(strOUT, sizeof(strOUT)-1, filePtr);
      pclose(filePtr);
      strOUT[(int) strlen(strOUT)-1]='\0';
      printf("%s [%X]\n", strOUT, msize[ii]);
      free(chrStr);
   }
```

TOP: Table of Processes (12-memory.c) (04)

```
    printf("\nPART 2\n");
    printLine(LINE);
    for (ii=0; ii < (sizeof(msize)/sizeof(int)); ii++){
        chrPTR = chrStr = malloc(msize[ii]);
        for (jj=0;jj<msize[ii];jj++)
            *chrPTR++='x';
        FILE* filePtr=popen(strSYS1, "r");
        fgets(strOUT, sizeof(strOUT)-1, filePtr);
        pclose(filePtr);
        strOUT[(int) strlen(strOUT)-1]='\0';
        printf("%s [%X]\n", strOUT, msize[ii]);
        free(chrStr);
    }
}
```

TOP: Table of Processes (13-chktoken.c) (05)

```
/*
 * Copyright (C) 2021 Rahmat M. Samik-Ibrahim
 * http://rahmatm.samik-ibrahim.vlsm.org/
 * This program is free script/software. This program is distributed in the
 * hope that it will be useful, but WITHOUT ANY WARRANTY; without even the
 * implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
 * REV05: Tue 30 Mar 14:55:36 WIB 2021
 * REV04: Tue 30 Mar 10:35:13 WIB 2021
 * REV03: Tue 30 Mar 08:36:56 WIB 2021
 * START: Mon 22 Mar 2021 16:14:36 WIB
 *
# INFO: chktoken(TOKEN) function
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

#define  MAXINPUT   256
#define  MAXCMD     MAXINPUT
#define  MAXOUTPUT  MAXINPUT
#define  RESULT     4

typedef           char  Chr;
typedef           char* ChrPtr;
typedef  unsigned char  uChr;
typedef  unsigned char* uChrPtr;
```

TOP: Table of Processes (13-chktoken.c) (05)

```
#define CMDSTRING "echo %s | sha1sum | cut -c1-4 | tr '[:lower:]' '[:upper:]' "
void mySHA1(uChrPtr input, uChrPtr output) {
    Chr  cmd[MAXCMD];
    sprintf(cmd, CMDSTRING, input);
    FILE* filePtr = popen(cmd, "r");
    fgets(output, RESULT+1, filePtr);
    output[RESULT]=0;
    pclose(filePtr);
}

void getTimeStamp(uChrPtr timeStamp) {
    time_t tt    =  time(NULL);
    struct tm tm = *localtime(&tt);
    sprintf(timeStamp, "%2.2d%2.2d", tm.tm_min, tm.tm_sec);
}

void     chktoken (uChrPtr token) {
    uChr    input [MAXINPUT];
    uChr    output[MAXOUTPUT];
    uChr    timeStamp[] = "MMSS";
    uChrPtr user        = getenv("USER");
    getTimeStamp(timeStamp);
    int     len   = strlen(timeStamp);
    strcpy(input,timeStamp);
    strcpy(input+len,user);
    len          += strlen(user);
    strcpy(input+len,token);
    len          += strlen(token);
    mySHA1(input,   output);
    printf("%s %s-%s\n", user, timeStamp, output);
}
```

TOP: Table of Processes (13-chktoken) (06)

```
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
ZCZC chktoken
cbkadal 5257-80A5
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
MiB Mem :    986.5 total,    157.1 free,    174.2 used,    655.2 buff/cache
MiB Swap:    488.0 total,    488.0 free,      0.0 used.    632.0 avail Mem

  PID    VIRT    RES    SHR   SWAP   CODE    DATA   USED nDRT
  864    6000   1528   1240      0      8     948   1528    0
PART 1
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
  864    6000   1528   1240      0      8     948   1528    0 [1E608]
  864    6000   2620   2292      0      8     948   2620    0 [1E608]
  864    6132   2620   2292      0      8    1080   2620    0 [1E609]
  864    6004   2620   2292      0      8     952   2620    0 [1E609]
  864    6004   2620   2292      0      8     952   2620    0 [1E609]
  864    6004   2620   2292      0      8     952   2620    0 [1F608]
  864    6004   2620   2292      0      8     952   2620    0 [1F608]
  864    6136   2620   2292      0      8    1084   2620    0 [1F609]
  864    6136   2624   2292      0      8    1084   2624    0 [1F609]
  864    6136   2624   2292      0      8    1084   2624    0 [20FE8]
  864    6136   2624   2292      0      8    1084   2624    0 [20FE8]
  864    6136   2624   2292      0      8    1084   2624    0 [40608]
  864    6136   2624   2292      0      8    1084   2624    0 [40608]
  864    6268   2624   2292      0      8    1216   2624    0 [40609]
  864    6264   2624   2292      0      8    1212   2624    0 [40609]
```

TOP: Table of Processes (13-chktoken) (07)

```
PART 2
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
  864     6004    2624    2292       0       8      952    2624     0 [1E608]
  864     6004    2736    2292       0       8      952    2736     0 [1E608]
  864     6004    2736    2292       0       8      952    2736     0 [1E609]
  864     6004    2736    2292       0       8      952    2736     0 [1E609]
  864     6004    2736    2292       0       8      952    2736     0 [1E609]
  864     6004    2736    2292       0       8      952    2736     0 [1F608]
  864     6004    2736    2292       0       8      952    2736     0 [1F608]
  864     6136    2736    2292       0       8     1084    2736     0 [1F609]
  864     6136    2736    2292       0       8     1084    2736     0 [1F609]
  864     6136    2736    2292       0       8     1084    2736     0 [20FE8]
  864     6136    2744    2292       0       8     1084    2744     0 [20FE8]
  864     6136    2748    2292       0       8     1084    2748     0 [40608]
  864     6136    2868    2292       0       8     1084    2868     0 [40608]
  864     6268    2868    2292       0       8     1216    2868     0 [40609]
  864     6268    2868    2292       0       8     1216    2868     0 [40609
```

Week 05: Check List (Deadline: 10 Oct 2021).

- Week 05 Token: **OS212W05**
- This page is https://os.vlsm.org/Slides/check05.pdf.
- More details: https://osp4diss.vlsm.org/W05.html.
- Assignment Check List:
  1. Read: (OSC10 chapter 10)
  2. Visit https://os.vlsm.org/GitHubPages/. Review **Last Week TOP 10 List** and pick at least 3 out of your 10 next neighbors. See https://cbkadal.github.io/os212/TXT/myrank.txt.
  3. Create your **TOP 10 List** of Week 05. **Do not use lecture material. Please be more creative!** (E.g. https://cbkadal.github.io/os212/W05/).
  4. Download https://os.vlsm.org/WEEK/WEEK05.tar.bz2.asc. The passphrase will follow. The result ("WEEK05-DEM005.txt") should be placed into a "W05/" folder and tarballed as "myW05.tar.bz2.asc"
  5. Update your log (e.g. https://cbkadal.github.io/os212/TXT/mylog.txt).
  6. Make SHA256SUM and sign it (detached, armor) as SHA256SUM.asc.

The End

- ☐ This is the end of the presentation.
- ☒ This is the end of the presentation.
- This is the end of the presentation.