

CSGE602055 Operating Systems

CSF2600505 Sistem Operasi

Week 09: Storage, Firmware, Bootloader, & Systemd

Rahmat M. Samik-Ibrahim (ed.)

University of Indonesia

<https://os.vlsm.org/Slides/os09.pdf>

Always check for the latest revision!

REV362 21-Nov-2021

OS212⁴): Operating Systems 2021 - 2

OS A	OS B	OS C	OS INT
Every first day of the Week, Quiz#1: (07:40-07:50) and Quiz#2: 07:20-07:40			
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 - 31 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**, 10th 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**, 10th 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>

Agenda

- 1 Start
- 2 Schedule
- 3 Agenda
- 4 Week 09
- 5 Storage, Firmware, Bootloader, & Systemd
- 6 Storage Management
- 7 RAID
- 8 Legacy BIOS
- 9 UEFI
- 10 Operating System (Boot) Loader

Agenda (2)

- 11 GRUB Map
- 12 init (SYSV legacy)
- 13 UpStart - Ubuntu
- 14 The All New "systemd"
- 15 systemctl
- 16 Week 09: Check List
- 17 The End

Week 09 Storage, Firmware, Bootloader, & Systemd: Topics¹

- Storage
- Storage Arrays
- BIOS
- Loader
- Systemd

¹Source: ACM IEEE CS Curricula 2013

Week 09 Storage, Firmware, Bootloader, & Systemd: Learning Outcomes¹

- Storage [Usage]
- Storage Arrays [Usage]
- BIOS [Usage]
- Loader [Usage]
- Systemd [Usage]

¹Source: ACM IEEE CS Curricula 2013

Storage, Firmware, Bootloader, & Systemd

- Reference: (OSC10-ch11)
- Storage Capacity (2019)¹
 - Legacy 3.5" Floppy Disk (1.4MB) – obsolete?
 - SuperDisk (up to 240 MB) — never took off.
 - 4.7" Compact Disc (700MB) – obsolete?
 - 4.7" Digital Versatile Disc (up to 9GB) – ?
 - 4.7" Blu Ray (up to 128 GB) ⇒ DVD++.
 - Tape Cartridge (up to 15TB)
 - Robotic System (up to 250 PB per unit)
 - NASA, Google, Microsoft are still using this!
 - Cheap but slow.
 - Hard Disk Drives (up to 16 TB).
 - From Perpendicular Magnetic Recording to Shingled Magnetic Recording technology (+25% – writing problems).
 - Mechanical Disk Arm Scheduling (Until When?).
 - Solid-State Disks (up to 16 TB).
 - SSD Price > HDD Price.
 - Write Speed >> Read Speed.
 - (What is a) Flash Disk?

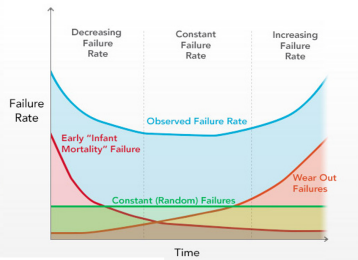
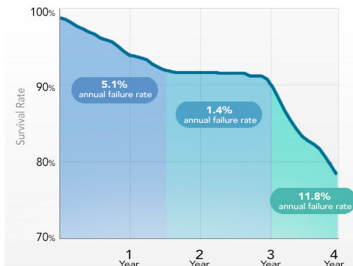
¹Subject to change

Storage Failure Rates

- MTDDL: Mean Time To Data Loss
- MTTF: Mean Time To Failure
- BackBlaze (Cloud Backup Services)

Drives Have 3 Distinct Failure Rates General Predicted Failure Rates

Hard Drive Survival Rates - Chart 1



<https://www.extremetech.com/computing/170748-how-long-do-hard-drives-actually-live-for>



Figure: BackBlaze — Failure Rates of 25000 DISKS

Storage Management

- Attached-Storage.
 - Host-Attached Storage: via I/O.
 - Network-Attached Storage (NAS): via distributed FileSystem.
 - Storage Area Network (SAN): dedicated Network.
- Formatting
 - Low Level (Physical)
 - High Level (FileSystem)
- Boot Block
- Disk Partition
 - "MBR"-scheme
 - upto 4 primary partition
 - upto 2 TB disk
 - "GPT"-scheme
 - "unlimited" partition
 - "unlimited" disk
 - redundancy
- Swap Space Management: On Partition or FileSystem?

RAID: Redundant Array of In* Disks

- RAID 0, 1, 5, 6, 10, 100
- Note (<http://www.commodore.ca/windows/raid5/raid5.htm>):
 - RAID was created to enhance data performance, reliability and availability.
 - Striping, parity checking and mirroring are three primary functions of RAID systems.
 - RAID performs its functions transparent to the operating system.
 - Systems are typically defined by ranks consisting of five disks each connected to one or two Disk Array Controllers.
 - Different RAID levels provide varying degrees of speed and data protection.
- Problems with RAID
- Stable-Storage Implementation

BIOS, Boot, & Systemd

- Firmware
 - BIOS: Basic Input Output System.
 - UEFI: Unified Extensible Firmware Interface.
 - ACPI: Advanced Configuration and Power Interface.
- Operating System (Boot) Loader
 - BOOTMGT: Windows Bootmanager / Bootloader.
 - LILO: Linux Loader.
 - GRUB: GRand Unified Bootloader.
- Operating System Initialization
 - Init (legacy)
 - UpStart
 - Systemd

- Check Settings.
- Initialize CPU & RAM.
- POST: Power-On Self-Test.
- Initialize ports, LANS, etc.
- Load a Boot Loader.
- Handover to the Boot Loader.
- Provides "Native" (obsolete) Drivers only (not loadable).
- Provides "INT" services .
- Limitation.
 - Technology of 1970s.
 - 16 bits software.
 - 20 bits address space (1 MB).
 - 31 bits disk space (2 TB).

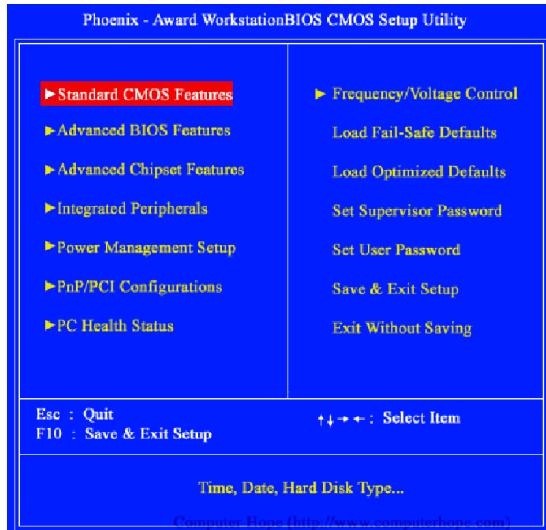


Figure: BIOS

- A Firmware Specification, not an Implementation!
- No (INT) service after boot.
- HII: Human Interface Infrastructure.
- Protected Mode.
- Flexible.
 - Technology of 2000s.
 - written in C.
 - (third party) loadable drivers and tools.
 - Emulate Legacy BIOS transition (MBR block, INT service).
 - UEFI Shell: environment shell for diagnostic (no need for DOS).
- Problems
 - Who controls the Hardware?
 - Is "Secure Boot" a good thing?
 - How about a **NASTY/LOCKING/TROJAN** UEFI implementation?
 - Different **DRIVERS**.



Figure: UEFI

Platform Initialization (PI) Boot Phases

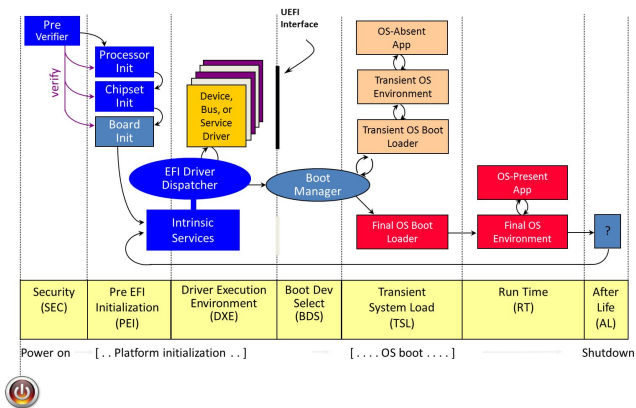


Figure: UEFI Boot Process¹.

¹Source Jarslstrom - 2014 - www.tianocore.org

Operating System (Boot) Loader

- General
 - How/Where to start the operating system?
 - What to do?
 - How many ways to boot?
 - How many types of OS?
- Disk Partition
 - MBR: Master Boot Record (1983).
 - GPT: GUID (Globally Unique Identifiers) Partition Table (2010s).
- GRUB: GRand Unified Boot system
 - Stage 1: a small boot.img inside the MBR.
 - Stage 1.5 (core.img): FileSystem drivers after MBR.
 - Stage 2: Kernel Selection: Windows, Linux, BSD, etc.
- GRUB2
 - More flexible than GRUB legacy.
 - More automated than GRUB legacy.
 - Accept MBR and GPT.
 - Stage 1.5 (core.img): generated from diskboot.img.
 - No 1024 cylinder restriction.

GNU GRUB 2

Locations of *boot.img*, *core.img* and the */boot/grub/* directory

Example 1: an MBR-partitioned harddisc with sector size of 512 or 4096Bytes



Example 2: a GPT-partitioned harddisc with sector size of 512 or 4096Bytes



Figure: GRUB¹.

¹Source Shmuel Csaba Otto Traian 2013

init (SYSV legacy)

- File: `/etc/inittab`.
- Folders: `/etc/rcX.d` — `X` = runlevel.
 - Seven (7) different runlevels:
 - 0 (shutdown).
 - 1 (single-user/admin).
 - 2 (multi-user non net).
 - 3 (standard).
 - 4 (N/A).
 - 5 (3+GUI).
 - 6 (reboot).
 - `SXX-YYY`: Start
 - `KXX-YYY`: Kill.
- One script at a time in order.
- dependency is set manually.

- Developer: Ubuntu.
- Folder: `/etc/init/`.
- Control: `initctl`.
 - `initctl list` – listing all processes managed by upstart.
- better support for hotplug devices.
- cleaner service management.
- faster service management.
- asynchronous.

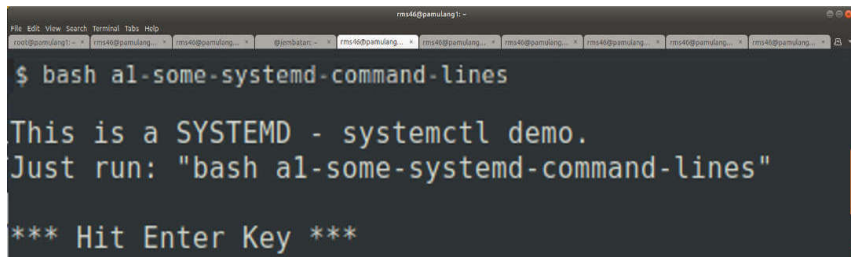
The All New "systemd"

- Replaces (SYSV) init and UpStart.
 - better concurrency handling: Faster!
 - better dependencies handling: No more "S(tarts)" and "K(ills)".
 - better crash handling: automatic restart option.
 - better security: group protection from anyone including superusers.
 - simpler config files: reliable and clean scripts.
 - hotplug: dynamic start/stop.
 - supports legacy systems (init).
 - overhead reducing.
 - unified management way for all distros.
 - bloated: doing more with more resources.
 - linux specific: NOT portable.

systemctl 01

```
for II in \
'systemctl list-unit-files | head -8; echo "(...)";
  systemctl list-unit-files| tail -8' \
'systemd-analyze blame | wc -l; echo "===";
  systemd-analyze blame | head -15' \
'systemctl --full | wc -l; echo "===";
  systemctl --full | head -10' \
'systemctl list-units | wc -l; echo "===";
  systemctl list-units | head -10' \
'systemctl list-units |grep .service|wc -l;echo "===";
  systemctl list-units|grep .service|head -10' \
'systemctl list-units | grep ssh.service' \
'systemctl status ssh.service' \
'systemctl is-enabled ssh' \
'journalctl' \
'journalctl -b' \
do
...
```

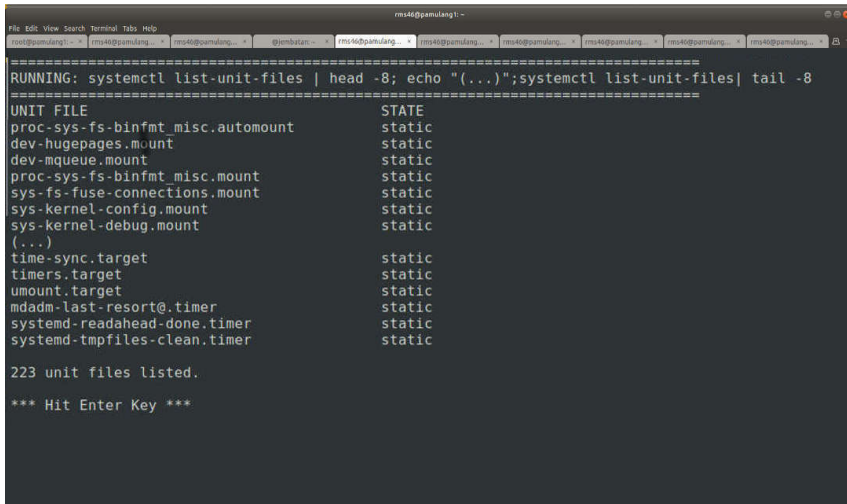
systemctl 02

A terminal window titled 'rms46@pamulang1: -' with multiple tabs. The active tab shows the command '\$ bash a1-some-systemd-command-lines' and its output: 'This is a SYSTEMD - systemctl demo. Just run: "bash a1-some-systemd-command-lines" *** Hit Enter Key ***'.

```
rms46@pamulang1: -  
root@pamulang1:~$ bash a1-some-systemd-command-lines  
This is a SYSTEMD - systemctl demo.  
Just run: "bash a1-some-systemd-command-lines"  
*** Hit Enter Key ***
```

Figure: bash a1-some-systemd-command-lines

systemctl 03



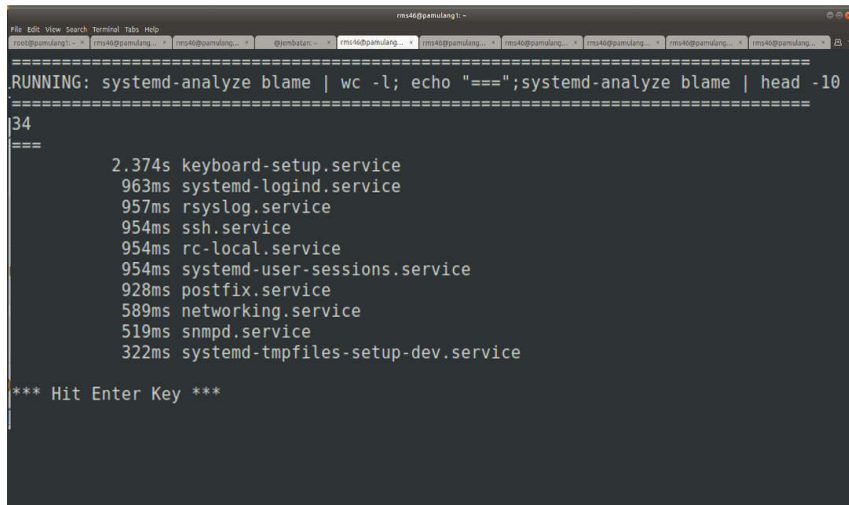
```
rms46@pamulang: ~  
File Edit View Search Terminal Tabs Help  
root@pamulang: ~ * rms46@pamulang: ~ * rms46@pamulang: ~ * @jembatan: ~ * rms46@pamulang: ~ * rms46@pamulang: ~ * rms46@pamulang: ~ * rms46@pamulang: ~ * rms46@pamulang: ~ * rms46@pamulang: ~ *  
=====  
RUNNING: systemctl list-unit-files | head -8; echo "...";systemctl list-unit-files| tail -8  
=====
```

UNIT FILE	STATE
proc-sys-fs-binfmt_misc.automount	static
dev-hugepages.mount	static
dev-mqueue.mount	static
proc-sys-fs-binfmt_misc.mount	static
sys-fs-fuse-connections.mount	static
sys-kernel-config.mount	static
sys-kernel-debug.mount	static
(...)	
time-sync.target	static
timers.target	static
umount.target	static
mdadm-last-resort@.timer	static
systemd-readahead-done.timer	static
systemd-tmpfiles-clean.timer	static

```
223 unit files listed.  
  
*** Hit Enter Key ***
```

Figure: systemctl list-unit-files

systemctl 04

A terminal window titled 'rms46@pamulang1: ~' showing the output of the command 'systemd-analyze blame'. The output lists various system services and their boot times. The command is repeated twice, and the output is separated by a separator line '===='. The first output shows the boot time for each service, and the second output shows the same information again. The terminal window has a menu bar with 'File Edit View Search Terminal Tabs Help' and a tab bar with multiple tabs labeled 'rms46@pamulang1: ~'.

```
=====  
RUNNING: systemd-analyze blame | wc -l; echo "===";systemd-analyze blame | head -10  
=====  
34  
===  
2.374s keyboard-setup.service  
963ms systemd-logind.service  
957ms rsyslog.service  
954ms ssh.service  
954ms rc-local.service  
954ms systemd-user-sessions.service  
928ms postfix.service  
589ms networking.service  
519ms snmpd.service  
322ms systemd-tmpfiles-setup-dev.service  
  
*** Hit Enter Key ***
```

Figure: systemd-analyze blame

systemctl 05

```
rms40@pamulang1: ~
File Edit View Search Terminal Tabs Help
rms40@pamulang1: ~
RUNNING: systemctl --full | wc -l; echo "===";systemctl --full | head -6
=====
97
=====
UNIT
proc-sys-fs-binfmt-misc.automount                                loaded active waiting Arbitrary Executable File Formats File System Automount Point
sys-devices-pci0000:00:0000:00:05:0-host0-target0:0:0:0:0-black-sda-sda1.device loaded active plugged QEMU_HARDDISK 1
sys-devices-pci0000:00:0000:00:05:0-host0-target0:0:0:0:0-black-sda-sda2.device loaded active plugged QEMU_HARDDISK 2
sys-devices-pci0000:00:0000:00:05:0-host0-target0:0:0:0:0-black-sdb.device loaded active plugged QEMU_HARDDISK
sys-devices-pci0000:00:0000:00:05:0-host0-target0:0:1:0:0:1-black-sdb-sdb1.device loaded active plugged QEMU_HARDDISK 1
*** Hit Enter Key ***

RUNNING: systemctl list-units | wc -l; echo "===";systemctl list-units | head -6
=====
97
=====
UNIT
proc-sys-fs-binfmt-misc.automount                                loaded active waiting Arbitrary Executable File Formats File System Automount Point
sys-devices-pci0000:00:0000:00:05:0-host0-target0:0:0:0:0-black-sda-sda1.device loaded active plugged QEMU_HARDDISK 1
sys-devices-pci0000:00:0000:00:05:0-host0-target0:0:0:0:0-black-sda-sda2.device loaded active plugged QEMU_HARDDISK 2
sys-devices-pci0000:00:0000:00:05:0-host0-target0:0:0:0:0-black-sdb.device loaded active plugged QEMU_HARDDISK
sys-devices-pci0000:00:0000:00:05:0-host0-target0:0:1:0:0:1-black-sdb-sdb1.device loaded active plugged QEMU_HARDDISK 1
*** Hit Enter Key ***

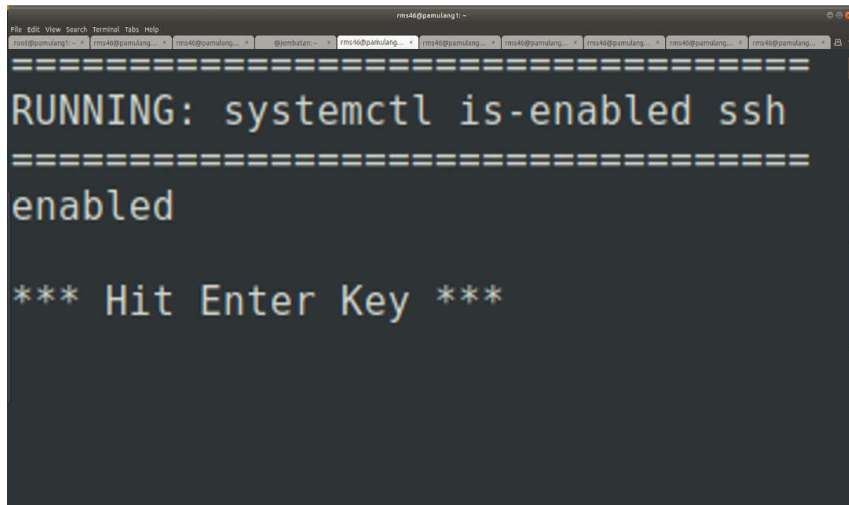
RUNNING: systemctl list-units | grep .service | wc -l; echo "===";systemctl list-units | grep .service | head -6
=====
12
=====
UNIT
ssct.service                                                    loaded active exited LSB: process and login accounting
icpidd.service                                                  loaded active running ACPI event daemon
console-setup.service                                           loaded active exited Set console font and keymap
cron.service                                                    loaded active running Regular background program processing daemon
dbus.service                                                    loaded active running D-Bus System Message Bus
jetty@tty1.service                                              loaded active running Getty on tty1
*** Hit Enter Key ***
```

Figure: systemctl --full; systemctl list-units

systemctl 06

```
rms46@pamulang1: ~
File Edit View Search Terminal Tabs Help
rms46@pamulang1 ~
=====
RUNNING: systemctl list-units | grep ssh.service
=====
ssh.service                                loaded active running    OpenBSD Secure Shell server
*** Hit Enter Key ***
=====
RUNNING: systemctl status ssh.service
=====
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled)
   Active: active (running) since Sun 2020-04-26 03:00:24 WIB; 3h 33min ago
   Process: 653 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 686 (sshd)
   CGroup: /system.slice/ssh.service
           └─ 686 /usr/sbin/sshd -D
              3247 sshd: demo [priv]
              3253 sshd: demo@pts/0
              3254 -bash
              3391 bash a1-some-systemd-command-lines
              3550 systemctl status ssh.service
*** Hit Enter Key ***
```

Figure: systemctl status ssh.service

A terminal window titled 'rms46@pamulang1: ~' with multiple tabs. The active tab shows the command 'systemctl status ssh' and its output. The output is displayed in a monospaced font with a dark background and light-colored text. The text is as follows:

```
=====  
RUNNING: systemctl is-enabled ssh  
=====  
enabled  
  
*** Hit Enter Key ***
```

Figure: systemctl is-enabled ssh

Week 09: Check List (Deadline: 21 Nov 2021).

□ Week 09: Assignment ([os09.pdf](#)). (Eg. **cbkadal**).

- Visit <https://osp4diss.vlsm.org/#idx0709>
- Week 08 - 10 will be about building "Linux From Scratch (LFS)"
 - 1 Read OSC10 chapter 11
 - 2 Try Demos in <https://github.com/UI-FASILKOM-OS/SistemOperasi/tree/master/Demos/>.
 - 3 Try Previous FinalTerm Problems (<https://rms46.vlsm.org/2/204.pdf>).
 - 4 Linux From Scratch 11.0
 - (a) Fetch and Extract File [WEEK09.tar.bz2.asc](#).
 - (b) Install (update) scripts to your \$HOME/bin/, check PATH to \$HOME/bin/, check \$LFS, and run Version Check
 - (c) Chapter 05 "Compiling a Cross-Toolchain" (if not finished)
 - (d) Chapter 06 "Cross Compiling Temporary Tools"
 - (e) Chapter 07 "Entering Chroot and Building Additional Temporary Tools" (start at least)
 - (f) Chapter 08 "Installing Basic System Software" (optional)
 - 5 Update your bookmark links. See C.B. Kadal's "LINKS/".
 - 6 (Optional) Any suggestions/tips for the next semester class? See C.B. Kadal's "TIPS/".
 - 7 Review your peer links.
 - 8 Update your log. See C.B. Kadal's "mylog.txt"
 - 9 Submit your Week 09 Assignment (See Week 03).

The End

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