Praktikum 24

Manajemen Service

POKOK BAHASAN:

- ✓ Services
- ✓ init daemon
- ✓ System V init
- ✓ systemd
- ✓ Menambahkan dan kustomisasi service baru

TUJUAN BELAJAR:

- ✓ Mahasiswa dapat mengidentifikasi service yang sedang berjalan
- ✓ Mahasiswa dapat membuat init daemon yang digunakan untuk menjalankan service
- ✓ Mahasiswa dapat menggunakan System V init
- ✓ Mahasiswa dapat menggunakan systemd
- ✓ Mahasiswa dapat menambahkan dan melakukan kustomisasi service baru

DASAR TEORI:

Systemd adalah pengelola sistem dan pengelola service di Linux . Systemd didesain untuk kompatibel dengan init script SysV, pengelola sistem dan service sebelum muncul systemd. Ketika system dibooting, systemd akam menjalankan program systemd-sysv-generator, yang berfungsi sebagai konversi antara systemd dan sysv. Program ini diletakkan di direktori /usr/lib/systemd/system-generators/ dan dijalankan secara otomatis oleh systemd pada saat proses bootstrapping setiap kali system di boot dan setiap kali systemd melakukan reload konfigurasinya.

Systemd dikembangkan pertama kali kali oleh RedHat untuk menggantikan Upstart yang tadinya digadang-gadang untuk menggantikan SysV sebagai sisyem init. Sekarang, Sistemd telah digunakan oleh berbagai distro Linux, termasuk Debian dan Ubuntu.

Systemd menggunakan konsep unit dalam mengatur resource dari OS. Unit-unit ini direpresentasikan oleh file konfigurasi yang terletak pada direktori yang ada pada table 24.2. Pada file ini, dideskripsikan informasi mengenai service system, listening socket, simpanan snapshot dari system state dan obyek-obyek lain yang relevan dengan sistem init. Pada tabel 24.1 disebutkan unit-unit yang digunakan oleh systemd.

Table 24.1. Available systemd Unit Types

Unit Type	File Extension	Description
Service unit	.service	A system service.
Target unit	.target	A group of systemd units.
Automount unit	.automount	A file system automount point.
Device unit	.device	A device file recognized by the kernel.
Mount unit	.mount	A file system mount point.
Path unit	.path	A file or directory in a file system.
Scope unit	.scope	An externally created process.
Slice unit	.slice	A group of hierarchically organized units that manage system processes.
Snapshot unit	.snapshot	A saved state of the systemd manager.
Socket unit	.socket	An inter-process communication socket.
Swap unit	.swap	A swap device or a swap file.
Timer unit	.timer	A systemd timer.

Tabel 24.2 Systemd Unit Files Locations

Directory	Description
/lib/systemd/system/	Systemd unit files distributed with installed RPM packages.
/run/systemd/system/	Systemd unit files created at run time. This directory takes precedence over the directory with installed service unit files.
/etc/systemd/system/	Systemd unit files created by systemctl enable as well as unit files added for extending a service. This directory takes precedence over the directory with runtime unit files.

Isi dari file konfigurasi unit pada direktori /etc/systemd/system

```
# cat atd.service
[Unit]
Description=ATD daemon

[Service]
Type=forking
ExecStart=/usr/bin/atd

[Install]
WantedBy=multi-user.target
```

Runlevel Systemd

SysV command	Systemd command	Deskripsi
runlevel	systemctl list-units --type target	Lists currently loaded target units.
telinit runlevel	systemctl isolate name.target	Changes the current target.

Sysvinit Runlevel	Systemd Target	Notes
0	runlevel0.target, poweroff.target	Halt the system
1, s, single	runlevel1.target, rescue.target	Single user mode.
2, 4	runlevel2.target, runlevel4.target, multi-user.target	User-defined/Site-specific runlevels. By default, identical to 3.
3	runlevel3.target, multi-user.target	Multi-user, non-graphical. Users can usually login via multiple consoles or via the network.
5	runlevel5.target, graphical.target	Multi-user, graphical. Usually has all the services of runlevel 3 plus a graphical login.
6	runlevel6.target, reboot.target	Reboot
emergency	emergency.target	Emergency shell

Unit

Systemd mengkatagori unit menjadi beberapa tipe berdasarkan tipe resourcenya. Cara yang paling mudah untuk menentukan tipe unit adalah dengan melihat ekstensi dari unit tersebut.Yaitu, service (.service), mount points (.mount), devais (.device), atau sockets (.socket).

Ketika anda menjalankan :

```
#systemctl list-units
```

Maka, akan muncul semua unit yang ada pada systemd yang dalam status aktif .

```
UNIT                                    LOAD   ACTIVE SUB     DESCRIPTION
atd.service                             loaded active running ATD daemon
avahi-daemon.service                    loaded active running Avahi mDNS/DNS-SD Stack
dbus.service                            loaded active running D-Bus System Message Bus
dcron.service                           loaded active running Periodic Command Scheduler
dkms.service                            loaded active exited  Dynamic Kernel Modules System
getty@tty1.service                      loaded active running Getty on tty1

. . .
```

Arti dari masing-masing unit ditunjukkan sebagai berikut :

Informasi Unit	Keterangan
UNIT	berisi nama unit systemd
LOAD	ketika konfigurasi unit telah diparsing oleh systemd, maka konfigurasi unit akan diloading dan disimpan di memory
ACTIVE	berisi ringkasan informasi status unit. Menunjukkan apakah unit tersebut telah berjalan dengan baik atau belum.
SUB	berisi informasi status lebih detail tentang setiap unit. Informasi ini bervariasi tergantung tipe unit, status dan method yang dijalankan unit

Berikut ini diberikan beberapa tipe unit yang digunakan oleh systemd beserta deskripsinya.

Tipe Unit	Deskripsi
.service	A service unit describes how to manage a service or application on the server. This will include how to start or stop the service, under which circumstances it should be automatically started, and the dependency and ordering information for related software.
.socket	A socket unit file describes a network or IPC socket, or a FIFO buffer that systemd uses for socket-based activation. These always have an associated .service file that will be started when activity is seen on the socket that this unit defines.
.device	A unit that describes a device that has been designated as needing systemd management by udev or the sysfs filesystem. Not all devices will have .device files. Some scenarios where .device units may be necessary are for ordering, mounting, and accessing the devices.
.mount	This unit defines a mountpoint on the system to be managed by systemd. These are named after the mount path, with slashes changed to dashes. Entries within /etc/fstab can have units created automatically.
.automount	An .automount unit configures a mountpoint that will be automatically mounted. These must be named after the mount point they refer to and must have a matching .mount unit to define the specifics of the mount.
.swap	This unit describes swap space on the system. The name of these units must reflect the device or file path of the space.
.target	A .target unit is used to provide synchronization points for other units when booting up or changing states. They also can be used to bring the system to a new state. Other units specify their relation to targets to become tied to the target's operations.
.path	This unit defines a path that can be used for path-based activation. By default, a .service unit of the same base name will be started when the path reaches the specified state. This uses inotify to monitor the path for changes.
.timer	A .timer unit defines a timer that will be managed by systemd, similar to a cron job for delayed or scheduled activation. A matching unit will be started when the timer is reached.
.snapshot	A .snapshot unit is created automatically by the systemctl snapshot command. It allows you to reconstruct the current state of the system after making changes. Snapshots do not survive across sessions and are used to roll back temporary states.
.slice	A .slice unit is associated with Linux Control Group nodes, allowing resources to be restricted or assigned to any processes associated with the slice. The name reflects its hierarchical position within the cgroup tree. Units are placed in certain slices by default depending on their type.
.scope	Scope units are created automatically by systemd from information received from its bus interfaces. These are used to manage sets of system processes that are created externally.

Manajemen Service

Pada Debian versi sebelumnya, digunakan SysV init atau Upstart untuk manajemen servicenya. Baik SysV dan Upstart menggunakan init script yang diletakkan di direktori /etc/rc.d/init.d/. Tiap init script ini ditulish dalam bahasa pemrograman Bash dan digunakan oleh administrator untuk mengontrol status service dan daemon di OS. Tentunya kita masih ingat perintah ini

```
#service networking stop
```

atau perintah

```
#/etc/init.d/networking stop
```

Ketika systemd menggantikan sysV, maka perintah-perintah tersebut berubah sebagai berikut :

SysV command	Systemd command	Deskripsi
service <name> start	systemctl start <name>.service	Starts a service.
service <name> stop	systemctl stop <name>.service	Stops a service
service <name> restart	systemctl restart <name>.service	Restarts a service.
service <name> condrestart	systemctl try-restart <name>.service	Restarts a service only if it is running.
service <name> reload	systemctl reload <name>.service	Reloads configuration.
service <name> status	systemctl status <name>.service systemctl is-active <name>.service	Checks if a service is running.
service --status-all	systemctl list-units --type service --all .	Displays the status of all services

Beberapa perintah yang dulunya tidak ada di sysv, namun ada di systemd

Systemd command	Deskripsi
systemctl enable <name>.service	To configure the Apache HTTP Server to start automatically at boot time
systemctl is-enabled <name>.service	Untuk mengecek apakah service dapat dijalankan otomatis saat booting
systemctl disable <name>.service	To prevent a service unit that corresponds to a system service from being automatically started at boot time
systemctl mask <name>.service	In addition, you can mask any service unit to prevent it from being started manually or by another service.
systemctl unmask <name>.service	you can unmask any service unit to let it from being started manually or by another service.

Tabel 24.5 Manajemen Daya

Systemd comman	Description
systemctl halt	Halts the system.
systemctl poweroff	Powers off the system.
systemctl reboot	Restarts the system.
systemctl suspend	Suspends the system.
systemctl hibernate	Hibernates the system.
systemctl hybrid-sleep	Hibernates and suspends the system

Percobaan 1 . Mengenal Systemd

1. Untuk melihat proses pertama dari OS. Opsi -p menunjukkan pid proses, opsi -h dipilih untuk menunjukkan hirarki

```
#pstree -ph
```

2. Untuk melihat versi dari systemd

```
#systemd –version
```

3. Untuk melihat waktu yang dibutuhkan oleh kernel, waktu yang dibutuhkan untuk initial RAM disk (initdisk) dan user program untuk diinisialisasi, dapat digunakan systemd-analyze time. Perhatikan bahwa waktu ini adalah waktu yang dibutuhkan untuk spawning sistem service. Bukan total keseluruhan sampai semua service selesai diinisialisasi

```
#systemd-analyze time
```

4. Jika anda ingin melihat waktu yang dibutuhkan untuk tiap unit melakukan inisialisasi maka

```
#systemd-analyze blame
```

5. Jika anda ingin melihat plot grafis dari waktu inisialisasi setiap unit, maka

```
#systemd-analyze plot > graph.svg
```

Setelah itu klik lewat explorer / file manager Debian untuk melihat graph.svg

6. Untuk melihat unit-unit yang ada dan statusnya aktif

```
#systemctl list-units
```

Untuk melihat seluruh unit-unit yang ada, termasuk yang statusnya failed dan inactive, tanpa kecuali

```
#systemctl list-units --all
```

Untuk melihat dari seluruh unit yang ada, yang mengalami status inactive

```
#systemctl list-units --all --state=inactive
```

Untuk melihat dari seluruh unit yang ada, yang bertipe service atau target saja :

```
#systemctl list-units --type=service
#systemctl list-units --type=target
```

7. Untuk melihat unit yang failed

```
#systemctl --failed
```

8. Perintah list-units hanya mendisplay unit yang telah diparsing dan diload di memory oleh systemd, karena systemd hanya membaca unit yang dibutuhkan saja. Untuk melihat semua unit file dalam path systemd (termasuk yang belum diload oleh systemd), digunakan perintah list-unit-files

```
#systemctl list-unit-files
```

Untuk melihat dari seluruh unit file yang ada, yang bertipe service :

```
#systemctl list-unit-file --type=service
```

9. Untuk menunjukkan dependensi unit

```
#systemctl list-dependencies  acpid.path
#systemctl list-dependencies  acpid.service
```

Percobaan 2. Menjalankan dan Menghentikan Service

1. Coba lihat listing semua service yang menyala di sistem anda. Perhatikan ada 4 kolom, yaitu unit, load, active, sub dan description.

```
#systemctl list-units -t service
```

Cari service cron

```
#systemctl list-units -t service | grep cron
```

Sekarang lakukan listing tanpa menampilkan keterangan (legend)

```
# systemctl list-units -t service --no-legend
```

Sekarang tampilkan semua service yang ada, tanpa menampilkan keterangan

```
# systemctl list-units -t service --no-legend --all
```

2. Jika belum ada, install service cron terlebih dahulu. Jika sudah ada langsung masuk ke langkah 3.

```
#apt-get install cron
```

Cek apakah status service cron active : active

```
#systemctl status cron
#systemctl status cron.service
```

3. Jika nilai active:active, Coba matikan service lewat systemd. Anda cukup memilih salah satu saja

```
# systemctl stop cron.service
# systemctl stop cron
```

Jika nilai active:inactive(dead), coba nyalakan service lewat systemd. Anda cukup memilih salah satu saja

```
# systemctl start cron.service
# systemctl start cron
```

4. Cek kembali apakah status service cron

```
#systemctl status cron.service
```

5. Lakukan restart. Anda cukup memilih salah satu saja

```
#systemctl restart cron.service
#systemctl restart cron
```

6. Cek kembali status service cron

```
#systemctl status cron
```

7. Sekarang, mari kita buat service cron berjalan saat booting

```
#systemctl enable cron
```

Reboot komputer anda dan cek status dari cron

```
# reboot
#systemctl status cron
```

Kemudian, mari kita buat service cron tidak dijalankan saat booting. Reboot komputer anda dan cek status dari cron

```
#systemctl disable cron
# reboot
#systemctl status cron
```

8. Anda dapat menggunakan perintah : is-active untuk mengecek apakah status cron aktif

```
#systemctl is-active cron
```

Anda dapat menggabungkan perintah diatas dengan script bash berikut. Apakah anda mengerti fungsi --quiet disini ?

```
#systemctl is-active cron --quiet; \
> if [ "$?" -eq "0" ]; then echo "service is active"; fi
```

Percobaan 3. Menyalakan dan mematikan system

1. Anda dapat mematikan OS dengan perintah poweroff

```
#systemctl poweroff
```

2. Anda dapat melakukan reboot dengan perintah reboot

```
#systemctl reboot
```

3. Anda dapat melakukan halt. Bedanya dengan powetoff, halt hanya mematikan cpu, sementara poweroff menjalankan skrip untuk melakukan shutdown

```
#systemctl  halt
```

4. Anda dapat melakukan suspend. Suspend disini bersifat suspend to RAM. Ini berarti mematikan power untuk hampir seluruh komponen, kecuali RAM. Dengan cara ini user dapat menghemat sebagian besar baterai. Suspend ini sangat disarankan jika user menutup laptop dan tidak aktif untuk beberapa waktu

```
#systemctl suspend
```

5. Anda dapat melakukan hibernate. Pada dasarnya hibernate adalah adalah suspend to disk. Ini berarti sistem akan menyimpan state terakhirnya pada swap space dan melakukan poweroff terhadap mesin.

```
#systemctl hibernate
```

Percobaan 4 .SSH dan systemd

1. Install ssh-server di PC anda

```
#apt-get install ssh openssh-server
```

2. Jalankan service ssh anda di server

```
#systemctl start ssh.service
```

3. Cek status service ssh di server anda. Apabila status Loaded: loaded dan Active:active(running) berarti server ssh anda sudah berjalan dan melakukan listening di port 22

```
#systemctl status ssh.service
```

4. Anda dapat mengecek bahwa Debian secara otomatis menambahkan file konfigurasi unit ke direktori /lib/systemd/system. Ada 3 file ssh, yaitu ssh.service, ssh@.service dan ssh.socket. Buka file ssh.service. Inilah file konfigurasi unit untuk service ssh

```
#cd  /lib/systemd/system
#ls -al | grep ssh
#cat ssh.service
```

5. Pada komputer server, catat nomor IP server. Kemudian buatlah username dan password di server

```
#ifconfig
#useradd  <nama_user_baru>
#passwd  <nama_user_baru>
```

6. Sekarang, lewat computer client, cek nomor IP client anda. Kemudian, mari kita login ke computer server.Masukkan password user_baru

```
#ifconfig
#ssh <nama_user_baru>@<no_IP_server>
```

7. Anda telah login di komputer server. Cek nomor IP server anda. Berhasilkah anda login ? Jika hasil ifconfig berbeda, maka anda telah login ke server. Kemudan exit dari server

```
$ifconfig
$exit
```

8. Sekarang kita akan mencoba membuat service ssh baru, yaitu service ssh-second
9. Salin file konfigurasi ssh untuk ssh-second

```
#cp /etc/ssh/sshd_config /etc/ssh/sshd-second_config
```

10. Buka file /etc/ssh/sshd-second_config

```
#nano /etc/ssh/sshd-second_config
```

Edit bagian Port dari 22 menjadi 22220 dan tambahkan baris PidFile dibawahnya. Simpan perubahan yang anda buat

```
Port 22220
PidFile /var/run/sshd-second.pid
```

11. Salin file unit konfigurasi ssh di /lib/systemd/system/ssh.service

```
#cp /lib/systemd/system/ssh.service /etc/systemd/system/ssh-second.service
#nano /lib/systemd/system/ssh.service
```

Ubah bagian Description dan After pada [Unit]

```
Description=OpenSSH server second instance daemon
After=  network.target auditd.service ssh.service
```

Kemudian ubah juga baris ExecStart pada [Service]. Simpan perubahan yang anda buat

```
ExecStart=/usr/sbin/sshd -D -f /etc/ssh/sshd-second_config $SSHD_OPTS
```

12. Lakukan reload

```
# systemctl daemon-reload
```

13. Sekarang cek status dari ssh dan ssh-second

```
#systemctl status ssh.service
#systemctl status ssh-second.service
```

14. Restart service ssh.service dan start service ssh-second.service

```
#systemctl restart ssh.service
#systemctl start ssh-second.service
```

15. Sekarang cek status dari ssh-second. Apabila status Loaded: loaded dan Active:active(running) berarti anda telah berhasil menjalankan service ssh-second.

```
#systemctl status ssh-second.service
```

16. Apabila status Loaded: loaded dan Active:active(exited) atau Loaded: loaded dan Active:failed maka restart lagi ssh-second.service

```
#systemctl start ssh-second.service
```

17. Cek dengan netsat untuk memastikan bahwa ssh-second telah melakukan listening di port 22220

```
#netstat -ntlupn | grep ssh
```

Anda akan melihat ada 2 service, ssh dan ssh-second berada pada port yang berbeda dan melakukan listening. Ini berarti server telah siap untuk diakses client.

18. Sekarang, pada computer client. Masukkan password <userbaru>

```
#ssh -p 22220  <nama_user_baru>@<no_IP_server>
```

19. Anda telah login di komputer server. Cek nomor IP server anda. Berhasilkah anda login Kemudian exit dari server jika sudah selesai

```
$ifconfig
$exit
```

Percobaan 5. Membangun service dengan Systemd

1. Pada direktori /etc/systemd/system, buatlah file test.service.

```
#cd /etc/systemd/system
#nano test.service
```

File tersebut berisi

```
[Unit]
Description = my first service unit
After = network.target
[Service]
Type = forking
ExecStart  =  /home/<user-name>/script/test.sh
[Install]
WantedBy = multi-user.target
```

2. Pada bagian [service] , ExecStart ini berisi perintah yang akan dijalankan oleh service ketika di-start. Buatlah script test.sh yang diletakkan di direktori /home/<user-name>/script/.

```
#cd /home/<user-name>/script/
#nano test.sh
```

File test.sh tersebut berisi :

```
#!/bin/bash
touch /home/<user>/testbootfile
```

Pada terminal, pada direktori dimana file test.sh berada lakukan

```
# chmod +x test.sh
```

3. Sekarang, kita lakukan reload daemon

```
# systemctl daemon-reload
```

4. Kemudian, cek status, jika status inactive,maka kita lakukan start service baru. Jika active, maka kita lakukan restart service baru

```
# systemctl status test.service
# systemctl start test.service
```

5. Cek status service anda. Apakah aktif ? Jika inactive mungkin disebabkan bahwa task yang dikerjakan sistem telah selesai.

```
# systemctl status test.service
```

6. Sekarang cek apakah service tersebut telah membentuk file testbootfile. Jika sudah ada, maka berarti service anda telah terpasang dengan baik

```
#cd /home/<user>/
#ls -al | grep testbootfile
```

7. Sekarang, hapus dulu file testbootfile. Kemudian, coba kita buat test.service berjalan setelah komputer booting dengan perintah enable. Setelah itu reboot.

```
#rm /home/<user>/testbootfile
# systemctl  enable test.service
#reboot
```

8. Sekarang cek apakah service tersebut telah membentuk file testbootfile. Jika sudah ada, maka berarti service anda dijalankan sistem segera setelah booting

```
#ls -al | grep testbootfile
```

Referensi

1. https://www.linux.com/learn/managing-services-linux-systemd

2. https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/ System_Administrators_Guide/sect-Managing_Services_with_systemd-Services.html#sect-Managing_Services_with_systemd-Services-List

3. https://www.howtogeek.com/216454/how-to-manage-systemd-services-on-a-linux-system/

4. https://blog.sleeplessbeastie.eu/2015/04/27/how-to-manage-system-services-on-debian-jessie/

5. https://www.digitalocean.com/community/tutorials/how-to-configure-a-linux-service-to-start-automatically-after-a-crash-or-reboot-part-1-practical-examples

6. https://fedoraproject.org/wiki/SysVinit_to_Systemd_Cheatsheet

7. https://www.freedesktop.org/wiki/Software/systemd/

8. https://www.digitalocean.com/community/tutorials/understanding-systemd-units-and-unit-files#where-are-systemd-unit-files-found

9. http://patrakov.blogspot.co.id/2011/01/writing-systemd-service-files.html

10. https://www.digitalocean.com/community/tutorials/how-to-use-systemctl-to-manage-systemd-services-and-units

11. https://www.tecmint.com/create-new-service-units-in-systemd/