

목차

Systemd 3

개요 3

Systemd 기본 정보 3

Unit File 4

 [Unit] Section 5

 [Service] Section 5

 [Install] Section 6

사용자 Unit 6

참조링크 7

참고사례 7

 kavita 7

Systemd

<https://www.freedesktop.org/software/systemd/>

Systemd 는 Linux 운영 체제 용 시스템 및 서비스 관리자입니다. SysV init 스크립트와 역 호환되도록 설계되었으며 부팅시 시스템 서비스의 병렬 시작, 데몬의 주문형 활성화 또는 종속성 기반 서비스 제어 논리와 같은 다양한 기능을 제공합니다. **Red Hat Enterprise Linux 7**에서 **systemd**는 기본 init 시스템으로 **Upstart**를 대체합니다.

개요

- [Systemd Unit 옵션](#)
- [Systemd 를 이용한 디렉토리/파일 모니터링 이벤트](#)
- [Systemd journal log 저장](#)
- [Systemd journal log 활용](#)

Systemd 기본 정보

Unit type

Unit Type	파일 확장자	설명
서비스 유닛	.service	시스템 서비스.
목표 단위	.target	시스템 단위 그룹입니다.
자동 장착 장치	.automount	파일 시스템 자동 마운트 지점.
장치 단위	.device	커널에서 인식하는 장치 파일입니다.
마운트 유닛	.mount	파일 시스템 마운트 지점.
경로 단위	.path	파일 시스템의 파일 또는 디렉토리.
범위 단위	.scope	외부에서 생성 된 프로세스입니다.
슬라이스 유닛	.slice	시스템 프로세스를 관리하는 계층 적으로 구성된 단위 그룹입니다.
스냅 샷 단위	.snapshot	systemd 관리자의 저장된 상태입니다.
소켓 유닛	.socket	프로세스 간 통신 소켓입니다.
단위 교체	.swap	스왑 장치 또는 스왑 파일.
타이머 유닛	.timer	시스템 타이머.

Unit file 위치

디렉토리	설명
/usr/lib/systemd/system/	설치된 RPM 패키지와 함께 배포되는 시스템 단위 파일.
/run/systemd/system/	런타임에 생성 된 시스템 단위 파일. 이 디렉토리는 설치된 서비스 유닛 파일이있는 디렉토리보다 우선합니다.
/etc/systemd/system/	systemctl enable 명령어로 생성 된 Systemd unit file. 서비스 확장을 위해 추가 된 unit file. 이 디렉토리는 런타임 단위 파일이있는 디렉토리보다 우선합니다.

systemd.conf 를 사용하여 기본 systemd 구성 재정의

systemd의 기본 구성은 컴파일 중에 정의되며의 systemd 구성 파일(/etc/systemd/system.conf)에서 찾을 수 있습니다. 이러한 기본값에서 벗어나 전체적으로 시스템 단위에 대해 선택한 기본값을 재정의하려면이 파일을 사용합니다.

예를 들어 90 초로 설정된 제한 시간 제한의 기본값을 대체하려면 DefaultTimeoutStartSec매개 변수를 사용하여 필요한 값을 초 단위로 입력하십시오.

```
DefaultTimeoutStartSec = 필수 값
```

service 명령과 systemctl 명령 비교

service	systemctl	설명
service name start	systemctl start name.service	서비스를 시작합니다.
service name stop	systemctl stop name.service	서비스를 중지합니다.
service name restart	systemctl restart name.service	서비스를 다시 시작합니다.
service name condrestart	systemctl try-restart name.service	실행중인 경우에만 서비스를 다시 시작합니다.
service name reload	systemctl reload name.service	구성을 다시로드합니다.
service name status	systemctl status name.service systemctl is-active name.service	서비스가 실행 중인지 확인합니다.
service -status-all	systemctl list-units -type service -all	모든 서비스의 상태를 표시합니다.

chkconfig 명령과 systemctl 명령 비교

chkconfig	systemctl	설명
chkconfig name on	systemctl enable name.service	서비스를 활성화합니다.
chkconfig name off	systemctl disable name.service	서비스를 비활성화합니다.
chkconfig -list name	systemctl status name.service systemctl is-enabled name.service	서비스가 활성화되어 있는지 확인합니다.
chkconfig -list	systemctl list-unit-files -type service	모든 서비스를 나열하고 활성화되었는지 확인합니다.
chkconfig -list	systemctl list-dependencies -after	지정된 단위 이전에 시작하도록 주문된 서비스를 나열합니다.
chkconfig -list	systemctl list-dependencies -before	지정된 단위 이후에 시작하도록 주문된 서비스를 나열합니다.

Unit File

유닛 파일에는 유닛을 설명하고 동작을 정의하는 구성 지시문이 포함되어 있습니다. 여러 systemctl 명령이 백그라운드에서 유닛 파일로 작동합니다. 미세 조정을하려면 시스템 관리자가 수동으로 유닛 파일을 편집하거나 생성해야 합니다.

단위 파일은 일반적으로 다음 세 섹션으로 구성됩니다.

- [Unit] — 장치 유형에 종속되지 않는 일반 옵션을 포함합니다. 이러한 옵션은 장치 설명을 제공하고 장치의 동작을 지정하며 다른 장치에 대한 종속성을 설정합니다.

- [unit type] — 단위에 유형별 지시문이있는 경우 단위 유형을 따라 명명 된 섹션 아래에 그룹화됩니다. 예를 들어, 서비스 단위 파일에는 [Service] 섹션이 포함되어 있습니다.
- [Install] — `systemctl enable` 및 `disable` 명령에서 사용하는 장치 설치에 대한 정보가 포함되어 있습니다.

[Unit] Section

[Unit] Section keyword

지시어	설명
Description	단위에 대한 의미있는 설명입니다. 이 텍스트는 예를 들어 <code>systemctl status</code> 명령 출력에 표시됩니다.
Documentation	장치에 대한 설명서를 참조하는 URI 목록을 제공합니다.
After	장치가 시작되는 순서를 정의합니다. 이 장치는 After 에 지정된 장치가 활성화된 후에 만 시작됩니다. Requires 와는 달리 After 는 명시적으로 지정된 단위를 활성화하지 않습니다. Before 옵션은 After 와 반대로 동작합니다.
Requires	다른 장치에 대한 종속성을 구성합니다. Requires 에 나열된 장치는 이 장치와 함께 활성화됩니다. 필요한 장치 중 하나라도 시작되지 않으면 장치가 활성화되지 않습니다.
Wants	Requires 보다 약한 종속성을 구성합니다. 나열된 장치가 시작되지 않더라도 장치 활성화에 영향을 주지 않습니다. 이것은 사용자 지정 단위 종속성을 설정하는 데 권장되는 방법입니다.
Conflicts	Requires 와 반대되는 부정적인 종속성을 구성합니다. 즉 나열된 장치가 활성화되어 있으면 시작되지 않습니다.

[Service] Section

[Service] Section keyword

지시어	기술
Type	<p>ExecStart 기능 및 관련 옵션에 영향을 미치는 단위 프로세스 시작 유형을 구성합니다.</p> <ul style="list-style-type: none"> * simple - 기본값입니다. ExecStart로 시작된 프로세스는 서비스의 주요 프로세스입니다. * forking - ExecStart로 시작된 프로세스는 서비스의 기본 프로세스가 되는 하위 프로세스를 생성합니다. 시작이 완료되면 상위 프로세스가 종료됩니다. * oneshot - 이 유형은 simple과 유사 하지만 후속 장치를 시작하기 전에 프로세스가 종료됩니다. * dbus - 이 유형은 simple과 유사 하지만 주 프로세스가 D-Bus 이름을 얻은 후에 만 시작됩니다. * notify - 이 유형은 simple과 유사 하지만 <code>sd_notify()</code> 함수를 통해 알림 메시지를 보낸 후에 만 시작됩니다. * idle - simple과 유사하게 모든 작업이 완료 될 때까지 서비스 바이너리의 실제 실행이 지연되어 상태 출력과 서비스의 셸 출력이 혼합되는 것을 방지합니다.
ExecStart	장치가 시작될 때 실행할 명령 또는 스크립트를 지정합니다. ExecStartPre와 ExecStartPost는 ExecStart 전, 후에 실행되는 사용자 지정 명령을 지정할 수 있습니다. Type=oneshot는 순차적으로 실행되는 여러 사용자 지정 명령을 지정할 수 있습니다.
ExecStop	장치가 중지 될 때 실행할 명령 또는 스크립트를 지정합니다.
ExecReload	장치를 다시로드 할 때 실행할 명령 또는 스크립트를 지정합니다.

지시어	기술
Restart	이 옵션을 사용하면 <code>systemctl</code> 명령에 의한 깨끗한 종지를 제외하고 프로세스가 종료된 후 서비스가 다시 시작됩니다.
RemainAfterExit	True로 설정하면 모든 프로세스가 종료된 경우에도 서비스가 활성 상태로 간주됩니다. 기본값은 False입니다. 이 옵션은 Type=oneshot가 구성된 경우 특히 유용합니다.
limit	LimitCPU=, LimitFSIZE=, LimitDATA=, LimitSTACK=, LimitCORE=, LimitRSS=, LimitNOFILE=, LimitAS=, LimitNPROC=, LimitMEMLOCK=, LimitLOCKS=, LimitSIGPENDING=, LimitMSGQUEUE=, LimitNICE=, LimitRTPRIO=, LimitRTTIME=

[Install] Section

[Install] Section keyword

키워드	설명
Alias	Unit에 대한 추가 이름의 공백으로 구분된 목록을 제공합니다. <code>systemctl enable</code> 를 제외한 대부분의 <code>systemctl</code> 명령은 실제 Unit 이름 대신 별칭을 사용할 수 있습니다.
RequiredBy	이 Unit에 의존성을 가진 다른 Unit 목록입니다. 이 장치가 활성화되면 이 Unit을 Require 종속성으로 가진 다른 Unit들이 RequiredBy에 나열됩니다.
WantedBy	이 Unit에 약한 의존성을 가진 다른 Unit 목록입니다. 이 장치가 활성화되면 이 Unit을 Want 종속성으로 가진 다른 Unit들이 WantedBy에 나열됩니다.
Also	이 Unit과 함께 설치 또는 제거할 다른 Unit 목록을 지정합니다.
DefaultInstance	인스턴스화된 Unit으로 제한되는 이 옵션은 Unit가 활성화되는 기본 인스턴스를 지정합니다.

사용자 Unit

현재 RHEL7 이상 버전에서는 이 방법이 작동하지 않습니다. RedHat 정책에 따라 사용자 Unit은 비활성화 되어있습니다. RHEL8은 지원된다고 되어있으나 사실상 사용하기 어려운 실정입니다. 관련 내용은 아래 링크를 참고하시기 바랍니다.

- <https://access.redhat.com/solutions/3461241>
- <https://access.redhat.com/solutions/3523861>
- <https://access.redhat.com/solutions/4661741>
- <https://access.redhat.com/solutions/5101061>

일반 사용자가 `sudo` 명령어로 특정 서비스만 정지/기동 가능하게 설정하기

- <https://access.redhat.com/solutions/1293513>

Debian/Ubuntu 계열은 잘 동작합니다. :)

일반적인 Unit file들은 system unit file이다. 이것들은 대부분 실행하는데 root권한을 요구한다. 이 부분에서는 일반 사용자 권한으로 unit file을 생성하고 제어하는 방법을 설명한다.

먼저 일반 사용자가 unit file을 생성하려면 `.config/systemd/user/` 경로에 생성해야 한다.

예를 들어 jboss를 제어하는 unit file을 아래와 같이 생성한다.

.config/systemd/user/jboss.service

```
[Unit]
Description=JBoss EAP Systemctl script
After=NetworkManager.service

[Service]
Type=forking
ExecStart=/opt/server/jboss-eap-6/bin/init.d/jboss-as-standalone.sh start
ExecStop=/opt/server/jboss-eap-6/bin/init.d/jboss-as-standalone.sh stop
ExecReload=/opt/server/jboss-eap-6/bin/init.d/jboss-as-standalone.sh restart
PIDFile=/var/run/jboss-as/jboss-as-standalone.pid

TimeoutStartSec=600
TimeoutStopSec=600

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

그리고 systemd 에서 해당 unit file을 로딩해준다.

```
[user@host ~]$ systemctl --user daemon-reload
[user@host ~]$ systemctl --user enable myunit.service
[user@host ~]$ systemctl --user start myunit.service
[user@host ~]$ systemctl --user stop myunit.service
[user@host ~]$ systemctl --user status myunit.service
```

참조링크

- https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/system_administrators_guide/chap-managing_services_with_systemd
- <https://www.freedesktop.org/software/systemd/man/systemd.directives.html>
- <https://www.digitalocean.com/community/tutorials/understanding-systemd-units-and-unit-files>

참고사례

kavita

백그라운드 프로세스 동작 시키는 스크립트

```
[Unit]
Description=Kavita Daemon
```

```
[Service]
User=root
WorkingDirectory=/Kavita
Type=simple
ExecStart=/Kavita/Kavita
PIDFile=/Kavita/Kavita.pid
Restart=on-failure

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

From:
<https://atl.kr/dokuwiki/> - **AllThatLinux!**

Permanent link:
<https://atl.kr/dokuwiki/doku.php/systemd>

Last update: **2023/03/21 01:24**

