- init
- systemd
- ps
  - -p
  - -u
  - aux
- unit
  - Unit types:
    - انواع واحدها (Unit Types) در systemd:
  - list-unit
  - list-units-files
    - state
    - وضعیت‌های یونیت در systemctl
  - مثال
  - --type
- بررسی یک سرویس مشص
  - cat
  - [Unit]
  - [Service]
  - [Install]
- Status
  - خروجی دستور
- دستوراتی که میتوانیم به سرویس ها بدهیم
  - start
  - stop
  - restart
  - enable
  - disable
  - status
  - is-active
  - reload

init

را راه می اندازد که مسول اجرای سایر برنامه با `init` هنگامی که کرنل بالا می اید کرنل . میباشد `systemd` از نوع `init` کانفیگ های مختلف می باشد. در توزیع های فعلی لیوکس

systemd

انواع پروسه های که `saystemd` راه می اندازد:

1. deamon
2. program
3. service
4. subservice

مشاهده init:

```
which init
```

output:

> /usr/sbin/`init`

اشاره به یک فایل دیگر دارد init در واقع:

```
readlink -f /sbin/init
```

output :

> /usr/lib/systemd/`systemd`

ps

دیدن همه پروسه ها

```
ps
# or
ls /proc
```

-p

یک می باشد . دید پروسه ID دارای init دارند. پروسه ID میدانیم که همه پروسه ها در لیوکس یک شماره ۱

```
ps -p 1
```

output:

```
PID  TTY  TIME     CMD
1    ?    0:00:01  systemd
```

```
ls /proc/1
```

دیدن همه پروسه ها: بصورت ابشاری

```
pstree |less
```

-u

دید پروسه های یک کاربر

```
ps -u <sername>
```

output:

```
PID TTY       TIME      CMD
2390 ?        00:00:00 systemd
2391 ?        00:00:00 (sd-pam)
2397 ?        00:00:00 pipewire
2398 ?        00:00:00 pipewire-media-
2399 ?        00:00:10 pulseaudio
```

aux

دیدن پروسه همعه کاربران

```
ps aux
```

output:

```
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.0  0.0 167076 11600 ?        Ss   08:48   0:01 /sbin/init
splash
root           2  0.0  0.0      0     0 ?        S    08:48   0:00 [kthreadd]
avahi       1091  0.0  0.0   7712  3840 ?        Ss   08:48   0:00 avahi-
daemon: running [TAJ.local]
root        1093  0.0  0.0  10628  5248 ?        Ss   08:48   0:00
/usr/lib/bluetooth/bluetoothd
root        1096  0.0  0.0  12380  3072 ?        Ss   08:48   0:00
/usr/sbin/cron -f -P
message+    1099  0.0  0.0  11244  6016 ?        Ss   08:48   0:01 @dbus-
daemon --system --address=systemd: --nofork --nopidfile --systemd-activation
--syslog-only

root        1129  0.0  0.0 243092  7728 ?        Ssl  08:48   0:00
/usr/libexec/power-profiles-daemon
nvidia-+    1130  0.0  0.0   5324  1924 ?        Ss   08:48   0:00
/usr/bin/nvidia-persistenced --user nvidia-persistenced --no-persistence-mode
--verbose
syslog      1133  0.0  0.0 222404  5504 ?        Ssl  08:48   0:00
/usr/sbin/rsyslogd -n -iNONE
```

1. **USER**: نام کاربری که فرآیند را اجرا کرده است.
2. **PID**: شناسه فرآیند (Process ID) که به هر فرآیند در سیستم یک شماره منحصربه‌فرد اختصاص می‌دهد.
3. **%CPU**: درصد استفاده از CPU توسط فرآیند در حال اجرا.
4. **%MEM**: درصد استفاده از حافظه (RAM) توسط فرآیند.
5. **VSZ (Virtual Size)**: اندازه مجازی فرآیند که نشان‌دهنده کل فضای حافظه‌ای است که فرآیند به خود اختصاص داده است، شامل حافظه واقعی و حافظه مجازی.
6. **RSS (Resident Set Size)**: اندازه مجموعه مقیم که نشان‌دهنده میزان حافظه فیزیکی است که فرآیند در حال استفاده است.
7. **TTY**: نام ترمینالی که فرآیند از آن راه‌اندازی شده است. برای مثال، `tty1`، `pts/0`.
8. **STAT (Process State)**: وضعیت فعلی فرآیند که می‌تواند شامل حالت‌های مختلفی باشد مانند:
    - `R` (Running): در حال اجرا
    - `S` (Sleeping): در حالت خواب

    - `D` (Uninterruptible Sleep): در حالت خواب غیرقابل وقفه
    - `T` (Stopped): متوقف شده
    - `Z` (Zombie): فرآیند زامبی که خاتمه یافته اما هنوز توسط والدینش جمع‌آوری نشده است.
9. **START**: زمان شروع فرآیند.
10. **TIME**: مدت زمانی که فرآیند در حال اجرا بوده است.
11. **COMMAND**: دستوری که برای اجرای فرآیند استفاده شده است.

مثال کاربردی:

```
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.1  22528  1100 ?        Ss   Dec29   0:01 /sbin/init
root         2  0.0  0.0      0     0 ?        S    Dec29   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    Dec29   0:00
[ksoftirqd/0]
```

در این مثال:

- **USER** `root` است.
- **PID** برای فرآیند اول `1` است.
- **%CPU** صفر درصد است.
- **%MEM** مقدار 0.1 درصد است.
- **VSZ** برابر 22528 بایت است.
- **RSS** برابر 1100 بایت است.
- **TTY** `?` به معنای فرآیند بدون ترمینال است.
- **STAT** `Ss` به معنای فرآیند در حالت خواب با اولویت سیستمی است.
- **START** زمان شروع `Dec29` است.
- **TIME** مدت زمان اجرای 0:01 دقیقه است.
- **COMMAND** `'/sbin/init'` است که دستور اجرای فرآیند است.

unit

`unit` سیستم از هر `unit` ساخته شده است و یا یک مجموعه از سرویس ها یا میتواند سرویس ها. همچی `unit` اقدام باشد. `تنظیمات` دارند و `نوع` و `نام` ها.

`Unit types`:

1. **Service Units**: `.service`

2. **Socket Units**: `.socket`
3. **Target Units**: `.target`
4. **Mount Units**: `.mount`
5. **Automount Units**: `.automount`
6. **Swap Units**: `.swap`
7. **Path Units**: `.path`
8. **Timer Units**: `.timer`
9. **Slice Units**: `.slice`
10. **Scope Units**: `.scope`

در systemd، واحدها (units) اشیایی هستند که عملیات و فرآیندهای مختلف سیستم را نمایندگی می‌کنند. هر واحد نوع مشخصی دارد که عملکرد خاصی را تعیین می‌کند. در زیر به برخی از انواع واحدهای (unit types) معمول در systemd اشاره می‌شود و توضیحات آن‌ها:

انواع واحدها (Unit Types) در systemd:

1. **Service Units (واحدهای سرویس)**
   - **پسوند فایل:** `.service`
   - **توضیح:** سرویس‌ها (services) واحدهایی هستند که برای مدیریت و کنترل فرآیندها و دیمون‌ها (daemons) استفاده می‌شوند. مثال: `apache2.service` یا `nginx.service`.
   - **مثال:** راه‌اندازی یک وب‌سرور
2. **Socket Units (واحدهای سوکت)**
   - **پسوند فایل:** `.socket`
   - **توضیح:** این واحدها برای مدیریت سوکت‌های IPC (ارتباط بین فرآیندی) و شبکه استفاده می‌شوند. معمولاً با سرویس‌هایی که به اتصالات شبکه پاسخ می‌دهند، مرتبط هستند.
   - **مثال:** `sshd.socket`
3. **Target Units (واحدهای هدف)**
   - **پسوند فایل:** `.target`
   - **توضیح:** این واحدها برای سازمان‌دهی و گروه‌بندی سایر واحدها استفاده می‌شوند. آن‌ها نقاط سنکرون‌سازی را نمایندگی می‌کنند.
   - **مثال:** `multi-user.target` یا `graphical.target`
4. **Mount Units (واحدهای کوه‌گذاری)**

- **پسوند فایل:** `.mount`
- **توضیح:** این واحدها برای مدیریت نقاط کوه‌گذاری فایل‌سیستم‌ها استفاده می‌شوند. مشابه دستورات `mount` و `umount` در لینوکس.
- **مثال:** `home.mount`

5. **Automount Units (واحدهای کوه‌گذاری خودکار)**
   - **پسوند فایل:** `.automount`
   - **توضیح:** این واحدها نقاط کوه‌گذاری خودکار را مدیریت می‌کنند و به `mount`ها اجازه می‌دهند به طور خودکار هنگام دسترسی، انجام شوند.
   - **مثال:** `home.automount`

6. **Swap Units (واحدهای مبادله)**
   - **پسوند فایل:** `.swap`
   - **توضیح:** این واحدها برای مدیریت فضای swap استفاده می‌شوند.
   - **مثال:** `swapfile.swap`

7. **Path Units (واحدهای مسیر)**
   - **پسوند فایل:** `.path`
   - **توضیح:** این واحدها برای نظارت بر مسیرهای فایل‌سیستم و اجرای سرویس‌ها در پاسخ به تغییرات در این مسیرها استفاده می‌شوند.
   - **مثال:** `cups.path`

8. **Timer Units (واحدهای تایمر)**
   - **پسوند فایل:** `.timer`
   - **توضیح:** این واحدها زمان‌بندی عملیات و اجرای سرویس‌ها در زمان‌های مشخص شده را مدیریت می‌کنند. مشابه cron jobs در لینوکس.
   - **مثال:** `backup.timer`

9. **Slice Units (واحدهای برش)**
   - **پسوند فایل:** `.slice`
   - **توضیح:** این واحدها برای مدیریت گروه‌های فرآیندی و تخصیص منابع استفاده می‌شوند.
   - **مثال:** `system.slice`

10. **Scope Units (واحدهای محدوده)**
    - **پسوند فایل:** `.scope`

- **توضیح:** ایجاد systemd این واحدها برای مدیریت مجموعه‌های فرآیندی که خارج از شده‌اند، استفاده می‌شوند.
- **مثال:** واحدهای `user.slice` برای تخصیص منابع به کاربر.

list-unit

دیدن مجموعه `unit` ها:

```
systemctl -list-units
```

list-units-files

دیدن فایلهای سرویس ها

```
systemctl list-units-file
```

state

فیلتر کرد براساس وضعیت سرویس

وضعیت‌های یونیت در `systemctl`

1. **enabled**: یونیت به طور خودکار در هنگام بوت سیستم فعال و اجرا می‌شود.
2. **disabled**: یونیت به طور خودکار در هنگام بوت سیستم فعال نمی‌شود، اما می‌توان آن را به صورت دستی فعال کرد.
3. **static**: یونیت خودش به تنهایی نمی‌تواند فعال شود، بلکه به عنوان وابستگی برای یونیت‌های دیگر استفاده می‌شود.
4. **masked**: یونیت عمداً غیرفعال شده و نمی‌تواند اجرا شود.
5. **generated**: یونیت‌هایی که به صورت خودکار توسط سیستم تولید شده‌اند.
6. **indirect**: یونیتی که به صورت غیرمستقیم از طریق alias اجرا می‌شود.
7. **alias**: یونیتی که به نام یونیت دیگری اشاره می‌کند.

مثال

```
systemctl list-units-file --state=disable
```

`--type`

مشص کردن نوع `unit` که میواهیم ببینیم.

`service` `socket` `target` `mount` `automount` `swap` `path` `timer` `slice` `scope`

```
systemd list-units --type=target
```

بررسی یک سرویس مشص

`cat`

مثال دیدن توضیحات آن سرویس : `Docker`

```
systemctl cat docker
```

> Output

[Unit]

- **Description:** توضیحات کوتاه درباره سرویس.
- **Documentation:** لینک مستندات سرویس.
- **After:** تعیین ترتیب شروع سرویس‌های دیگر قبل از این سرویس.
- **Wants:** سرویس‌هایی که این سرویس نیاز دارد ولی عدم موفقیت آنها باعث شکست این سرویس نمی‌شود.
- **Requires:** سرویس‌هایی که این سرویس حتماً به آنها نیاز دارد و عدم موفقیت آنها باعث شکست این سرویس می‌شود.

[Service]

- **Type:** نوع اعلان شروع سرویس.
- **ExecStart:** فرمان شروع سرویس.
- **ExecReload:** فرمان بارگذاری مجدد سرویس.
- **TimeoutStartSec:** زمان انتظار برای شروع سرویس.
- **Restart:** پیکربندی برای راه‌اندازی مجدد سرویس در صورت خروج.
- **RestartSec:** زمان تاخیر قبل از راه‌اندازی مجدد.
- **StartLimitBurst:** تعداد دفعات شروع در یک بازه زمانی مشخص.
- **StartLimitInterval:** بازه زمانی برای StartLimitBurst.
- **LimitNPROC:** حداکثر تعداد پروسه‌ها.
- **LimitCORE:** حداکثر اندازه فایل‌های هسته.
- **TasksMax:** حداکثر تعداد کارها.
- **Delegate:** اجازه دادن به سرویس برای کنترل گروه‌های cgroup.
- **KillMode:** کنترل کدام پروسه‌ها در هنگام توقف سرویس کشته شوند.
- **OOMScoreAdjust:** تنظیم امتیاز Out-Of-Memory (OOM) killer.

[Install]

- **WantedBy:** تعیین هدفی که سرویس به آن لینک شده است.

Status

برای بررسی وضعیت فعلی یک سرویس در سیستم استفاده می‌شود.به عنوان مثال، برای بررسی وضعیت سرویس apache2 از دستور زیر استفاده می‌شود:

```
systemctl status apache2
```

خروجی دستور

خروجی این دستور شامل اطلاعات جامعی درباره وضعیت سرویس مورد نظر است. به طور معمول خروجی به شکل زیر است:

```
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor
preset: enabled)
   Active: active (running) since Mon 2024-12-30 09:00:00 UTC; 1 day 10h ago
     Docs: https://httpd.apache.org/docs/2.4/
```

```
 Main PID: 1234 (apache2)
    Tasks: 55 (limit: 4915)
   Memory: 24.5M
   CGroup: /system.slice/apache2.service
           ├─1234 /usr/sbin/apache2 -k start
           ├─1235 /usr/sbin/apache2 -k start
           └─1236 /usr/sbin/apache2 -k start
```

- **Loaded:** نشان می‌دهد که فایل سرویس بارگذاری شده است و آیا سرویس برای شروع خودکار پیکربندی شده است یا خیر.
- **Active:** وضعیت فعلی سرویس را نشان می‌دهد که آیا سرویس در حال اجرا است یا نه.
- **Docs:** لینک به مستندات مربوط به سرویس.
- **Main PID:** شناسه فرآیند اصلی سرویس.
- **Tasks:** تعداد وظایفی که سرویس در حال حاضر اجرا می‌کند.
- **Memory:** میزان حافظه‌ای که توسط سرویس استفاده شده است.
- **CGroup:** مسیری که نشان دهنده مکان سرویس در سیستم گروه‌های کنترلی است.

آیا توضیحات بیشتری نیاز دارید یا سوال دیگری دارید؟

دستوراتی که میتوانیم به سرویس ها بدهیم

start

- **توضیح:** این دستور برای شروع یک سرویس خاص استفاده می‌شود.
- **مثال:**

```
systemctl start apache2
```

stop

- **توضیح:** این دستور برای متوقف کردن یک سرویس خاص استفاده می‌شود.
- **مثال:**

```
systemctl stop docker
```

restart

- **توضیح:** این دستور برای راه‌اندازی مجدد یک سرویس استفاده می‌شود.
- **مثال:**

```
systemctl restart nginx
```

enable

- **توضیح:** این دستور برای فعال کردن یک سرویس به منظور شروع خودکار آن در زمان راه‌اندازی سیستم استفاده می‌شود.
- **مثال:**

```
systemctl enable ssh
```

disable

- **توضیح:** این دستور برای غیرفعال کردن یک سرویس به منظور جلوگیری از شروع خودکار آن در زمان راه‌اندازی سیستم استفاده می‌شود.
- **مثال:**

```
systemctl disable mysql
```

status

- **توضیح:** این دستور برای بررسی وضعیت فعلی یک سرویس استفاده می‌شود.
- **مثال:**

```
systemctl status apache2
```

is-active

- **توضیح:** این دستور برای بررسی فعال بودن یا نبودن یک سرویس استفاده می‌شود.
- **مثال:**

```
systemctl is-active docker
```

reload

- **توضیح:** این دستور برای بارگذاری مجدد پیکربندی یک سرویس بدون توقف کامل آن استفاده می‌شود.
- **مثال:**

```
systemctl reload apache2
```