systemd

puntog@gmail.com

systemd

- Suite para la administración del inicio del sistema.
- Compatible con *SystemV* y *scripts* de inicio *LSB*
- Desarrollado por **Lennart Poettering** y **Kay Sievers**.
- Capacidad de paralización usando *sockets* y *D-Bus.*

systemd

- Permite inicio de demonios bajo demanda.
- Mantiene el control de los procesos usando grupos de control de Linux.
- Soporta instantáneas y restauración del estado del sistema.
- Monta y desmonta puntos de montaje.
- Implementa una lógica de control de servicio basado en la dependencia transaccional muy elaborada.

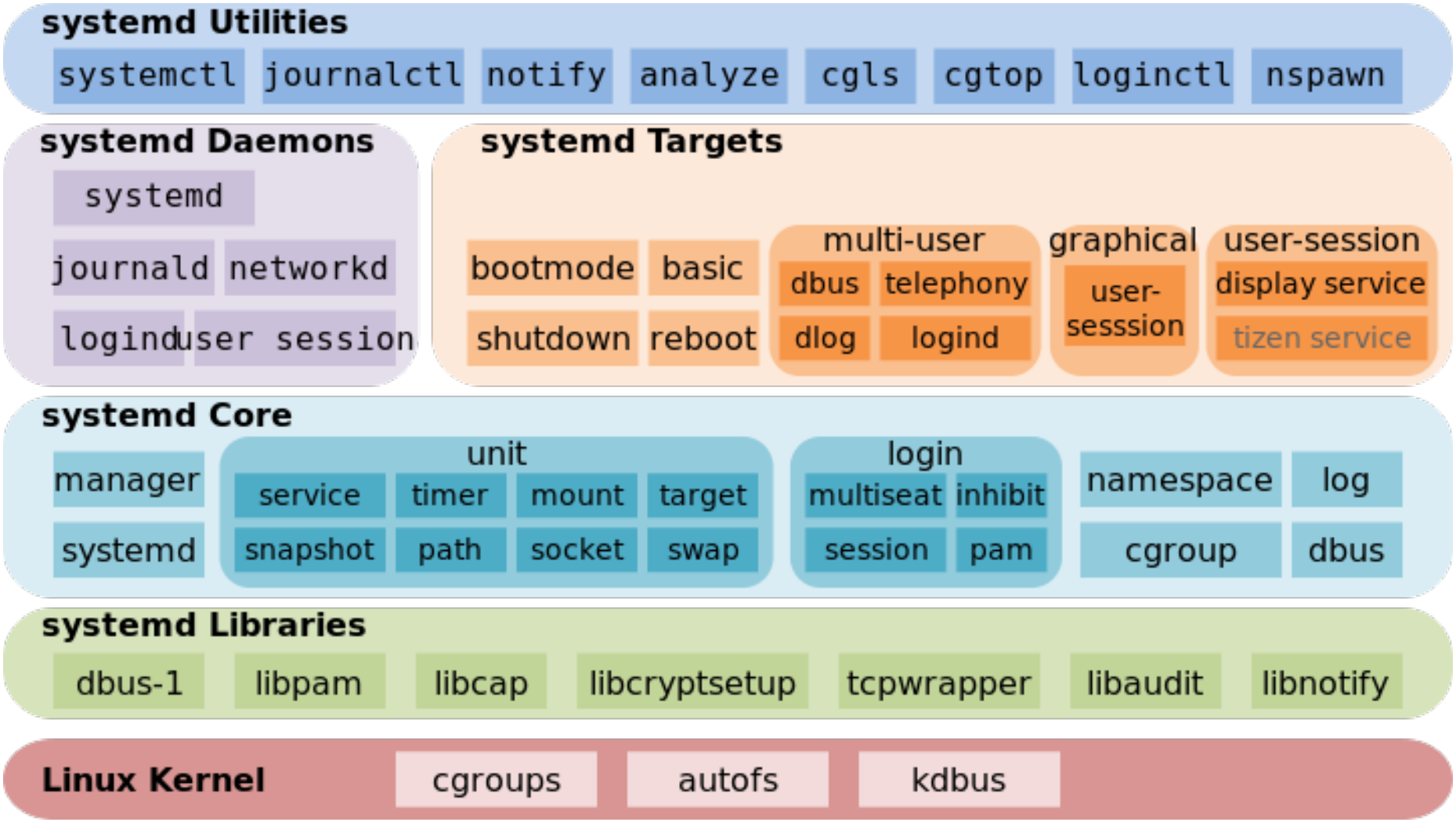
systemd

- Reemplaza la secuencia de inicio y los *runlevels* que eran controlados por el demonio *init*, con *scripts* en *shell* ejecutados bajo su control.
- Integra otros servicios como *logins*, consolas, dispositivos, bitácoras, entre otros.
- Ejecuta elementos en paralelo.
- Para la comunicación interprocesos utiliza *sockets* y *D-Bus.*

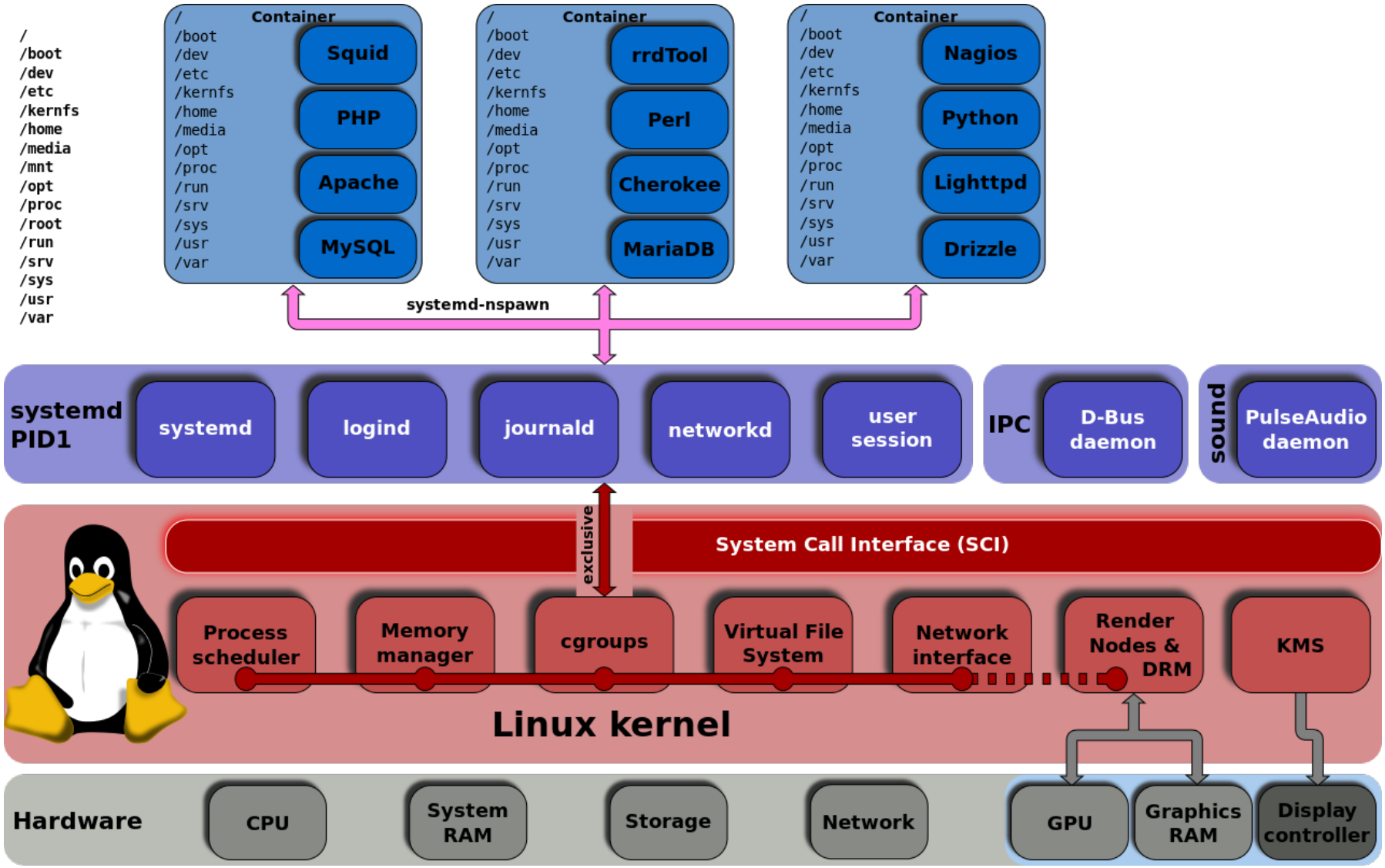
systemd

- Tres funciones generales:
  - Sistema y administrador de servicios
  - Plataforma de *software*
  - Enlace entre las aplicaciones y el *kernel*

Arquitectura



Componentes



Unidades

- systemd inicializa registros por cada demonio en un archivo de configuración (llamado archivo de unidad) que utiliza un lenguaje declarativo, reemplazando los *scripts* en *shell*.
- Unidades:
  - Servicio .service
  - Punto de montaje .mount
  - Dispositivo .device
  - Socket .socket

Tipo de Unidad	Extensión	Descripción
Servicio (Service)	.service	Servicio del sistema
Destino (Target)	.target	Grupo de unidades systemd
Automontaje (Automount)	.automount	Punto de montaje automático de un sistema de archivos
Dispositivo (Device)	.device	Dispositivo reconocido por el kernel
Punto de montaje (Mount)	.mount	Punto de montaje de un sistema de archivos
Ruta (Path)	.path	Un archivo o directorio en un sistema de archivos
Alcance (Scope)	.scope	Un proceso creado de manera externa
Slice	.slice	Grupo de unidades organizadas jerárquicamente que administra procesos del sistema
Instantánea (Snapshot)	.snapshot	Estado guardado del systemd
Socket	.socket	Socket para comunicación interprocesos
Intercambio (Swap)	.swap	Dispositivo de intercambio
Tiempo (Timer)	.timer	Un contador de systemd

Componentes principales

- systemd: Administrador del sistema y servicios.
- systemctl: Controla el estado del sistema systemd y administrador de servicios.
- systemd-analyze: se utiliza para tener estadísticas sobre el rendimiento del sistema de carga.
- Subsistema cgroups.

Componentes auxiliares

- systemd-consoled: provee un demonio para consola de usuario.
- systemd-journald: demonio responsable de la bitácora de eventos.
- systemd-logind: demonio que administra el ingreso de usuarios.
- networkd: demonio que maneja la configuración de la red.

Componentes auxiliares (2)

- systemd-timedate controla la configuración de fecha y hora.
- udev: administrador de dispositivos para el *kernel* de *Linux*, el cual administra el directorio */dev*.
- libudev: librería estándar para utilizar *udev*.

systemd

- Ya no existe /etc/inittab
- /etc/systemd/system enlaces a /usr/lib/systemd/system contiene scripts
- En el caso de Debian las unidades se encuentran en **`/lib/systemd/system`**

Targets

- Corresponden a los runlevels:
  - basic.target
  - multiuser.target
  - graphical.target
  - ctrl-alt-del.target
  - default.target
  - display-manager.service
  - getty.target
- **`man 7 systemd.special`**

Comandos

- Ver el estado de todos los servicios (systemd y SysV/LSB)
  - **`systemctl`**
- Obtener más información de un servicio:
  - **`systemctl status`** ***`servicio`***
- Habilitar un servicio
  - **`systemtcl enable`** ***`servicio`***

Comandos (2)

- Listar el estado de todos los servicios que tienen *script* de inicio
  - **`systemctl list-unit-files --type=service`**
- Iniciar un servicio
  - **`systemctl start servicio`**
- Detener un servicio
  - **`systemctl stop servicio`**

Análisis del estado del sistema

- Analizar el estado del sistema
  - **`systemctl status`**
- Listar unidades en ejecución
  - **`systemctl`**
  - **`systemctl list-units`**
- Listar unidades que han fallado
  - **`systemctl --failed`**
- Listar unidades instaladas
  - **`systemctl list-unit-files`**

Manejo de Unidades

- Iniciar una unidad inmediatamente
  - **`systemctl start unidad`**
- Detener una unidad inmediatamente
  - **`systemctl stop unidad`**
- Recargar una unidad
  - **`systemctl restart unidad`**
- Ver el estado de una unidad
  - **`systemctl status unidad`**
- Saber si una unidad está activa o no
  - **`systemctl is-active unidad`**

Manejo de Unidades (2)

- Habilitar una unidad para que inicie al cargar el sistema
  - **`systemctl enable unidad`**
- Deshabilitar una unidad para que no inicie al cargar el sistema
  - **`systemctl disable unidad`**

Manejo de unidades (2)

- Enmascarar una unidad para que haga imposible se inicie
  - **systemctl mask unidad**
- Desenmascarar una unidad
  - **`systemctl unmask unidad`**
- Recargar systemd para unidad nuevas o cambios
  - **`systemctl daemon-reload`**

Administración de encendido

- Reiniciar el sistema
  - **`systemctl reboot`**
- Apagar el sistema
  - **`systemctl poweroff`**
- Suspender el sistema
  - **`systemctl suspend`**
- Hibernar el sistema
  - **`systemctl hibernate`**

Ejemplos

- Ver árbol de cgroup
  - **`systemd-cgls`**
- Ver el modo en que inicia por defecto
  - **`systemctl get-default`**
- Cambiar que inicie en modo texto
  - **`ln -sf`**
    `/lib/systemd/system/multiuser.target`
    `/lib/systemd/system/default.target`
  - **`systemctl set-default graphical.target`**
  - **`systemctl set-default default.target`**
- Desactivar servicio apache2
  - **`systemctl disable apache2`**

Crear un archivo de unidad

- La sintáxis está basada en las especificaciones XDG Desktop de los archivos .desktop, similar a los archivos .ini de Microsoft Windows.
- Los archivos de las unidades se encuentran en:
  - /lib/systemd/system: *Unidades de paquetes instalados*
  - /etc/systemd/system: *Unidades instaladas por el administrador*

Dependencias

- Requieren de un diseño correcto de la unidad
- Si **A** **requiere** que la unidad **B** esté corriendo antes que **A** sea iniciado, se requiere agregar:
  - `Requires=B`
  - `After=B`

  Antes de la sección `[Unit]` de **A**

Dependencias (2)

- Si la dependencia es **opcional** se requiere agregar:
  - `Wants=B`
  - `After=B`
- `Wants=` y `Requieres=` no implica un `After`, es decir si no se especifica `After=` las dos unidades se inician en paralelo.
- Las dependencias se colocan por lo general en servicios.

Tipos de servicios

- El tipo de servicio se define con el parámetro `Type=` en la sección `[Service]`.

Type=simple

- El servicio se inicia inmediatamente.
- El proceso no debe crear proceso hijo (fork).

Type=forking

- El servicio se inicia cuando se hace el fork y el proceso padre ha termiando.
- Se usa para iniciar demonios.
- Se debe especificar `PIDFile=`

Type=notify

- Similar a Type=simple, pero el demonio enviará una señal a systemd cuando esté listo.

Type=dbus

- El servicio se considera listo cuando el BusName especificado aparece en el bus del sistema DBus.

Tipos de servicios (2)

Type=oneshot

- Scripts que hace una tarea y finalizan.
- `RemainAfterExit=yes` para considerar el servicio como activo después de que el proceso terminó.

Type=idle

- `systemd` detiene la ejecución del servicio binario hasta que todo los trabajos sean despachados.

Bitácoras con systemd

- Ver la bitácora de un servicio
  - `journalctl -u unidad.servicio`
- Ver todas las entradas en la bitácora
  - `journalctl`
- Ver solo los mensajes de la carga actual
  - `journalctl -b`
- Ver los mensajes al estilo dmesg
  - `journalctl -k`

Ejemplo (1)

- Crear el archivo /etc/systemd/system/raton.py

```
#!/usr/bin/python
import SocketServer
from BaseHTTPServer import BaseHTTPRequestHandler

class MyHandler(BaseHTTPRequestHandler):
    def do_GET(self):
        self.send_response(200)
        self.send_header("Content-type", "text/plain")
        self.end_headers()
        self.wfile.write('<:3)~~~\n')

httpd = SocketServer.TCPServer(("", 8888), MyHandler)
httpd.serve_forever()
```

Ejemplo (2)

- Darle permiso de ejecución y probarlo (se puede probar con el navegador web)
- Crear el archivo de unidad /lib/systemd/system/raton.py.service

```
[Unit]
Description=Mouse Logging Service

[Service]
ExecStart=/etc/systemd/system/raton.py
StandardOutput=null

Install]
WantedBy=multi-user.target
Alias=mouselogger.service
```

Ejemplo (3)

- Habilitar el servicio

  `systemctl enable raton.py.service`
- Arrancar el servicio

  `systemctl start raton.py.service`

Ejemplo 2 (1)

- El siguiente programa crea un servicio UDP que escucha en el puerro 10000 y regresa el mensaje recibido con la transformación Rot13 (escrito en PHP):
- Crear archivo rot13.php

```
<?php
$sock = socket_create(AF_INET, SOCK_DGRAM, SOL_UDP);
socket_bind($sock, '0.0.0.0', 10000);
for (;;) {
    socket_recvfrom($sock, $message, 1024, 0, $ip, $port);
    $reply = str_rot13($message);
    socket_sendto($sock, $reply, strlen($reply), 0, $ip, $port);
}
?>
```

Ejemplo 2 (2)

- Probar servidor

```
php rot13.php
```

- Probar cliente

```
nc -u 127.0.0.1 10000
```

Ejemplo 2 (3)

- Crear el archivo de unidad:

```
[Unit]
Description=ROT13 demo service
After=network.target
StartLimitIntervalSec=0
[Service]
Type=simple
Restart=always
RestartSec=1
User=nobody
ExecStart=/usr/bin/env php
/etc/systemd/system/rot13.php
[Install]
WantedBy=multi-user.target
```

Ejemplo 2 (4)

- Habilitar el servicio

```
systemctl start rot13
```

- Iniciar el servicio

```
systemctl enable rot13
```

Actividad

- Hacer un servicio que active el servidor web instalado desde código fuente.

Referencias

- https://www.linux.com/learn/tutorials/788613-understanding-and-using-systemd
- http://0pointer.de/blog/projects/systemd-for-admins-1.html
- http://lwn.net/Articles/389149/
- https://en.wikipedia.org/wiki/Systemd
- https://wiki.archlinux.org/index.php/Systemd
- https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/System_Administrators_Guide/chap-Managing_Services_with_systemd.html
- http://unix.stackexchange.com/questions/47695/how-to-write-startup-script-for-systemd
- https://www.digitalocean.com/community/tutorials/how-to-use-systemctl-to-manage-systemd-services-and-units
- https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/System_Administrators_Guide/sect-Managing_Services_with_systemd-Unit_Files.html
- https://www.freedesktop.org/wiki/Software/systemd/TipsAndTricks/
- https://www.digitalocean.com/community/tutorials/understanding-systemd-units-and-unit-files