**Phần 16: Cách tạo dịch vụ systemd trong Linux**

1. **Dịch vụ systemd là gì?**

Nói một cách đơn giản, dịch vụ là " [quy trình nền](https://linuxhandbook.com/run-process-background/) " được bắt đầu hoặc dừng dựa trên một số trường hợp nhất định. Bạn không cần phải bắt đầu và/hoặc dừng nó theo cách thủ công. 'Tệp dịch vụ systemd' là tệp được viết theo định dạng sao cho systemd có thể phân tích cú pháp và hiểu được tệp đó, và sau đó thực hiện những gì bạn yêu cầu.

Về mặt kỹ thuật, thứ tôi gọi là 'tệp dịch vụ systemd' thực chất được gọi là tệp 'đơn vị systemd', nhưng vì hướng dẫn này nói về việc tạo dịch vụ nên tôi sẽ tiếp tục gọi đây là 'tệp dịch vụ systemd'.

1. **Hiểu cấu trúc cơ bản của tệp dịch vụ systemd**

Tệp dịch vụ systemd có ba phần quan trọng và cần thiết. Đó là các phần **[Unit]**, **[Service]**và **[Install]**. Phần mở rộng của tệp dịch vụ systemd là **.service**và chúng tôi sử dụng ký hiệu pound/hash ( **#**) cho các chú thích dòng đơn.

**Phần[Unit]​**

Phần Unit chứa thông tin chi tiết và mô tả về chính unit đó. Trong trường hợp của chúng tôi, nó chứa thông tin chi tiết về dịch vụ. Các thông tin chi tiết như 'mô tả của nó là gì', 'các phụ thuộc của nó là gì' và nhiều thông tin khác.

Dưới đây là các trường mà phần Đơn vị có:

* **Description**:- Tiêu đề dễ đọc của dịch vụ systemd.
* **After**:- Đặt sự phụ thuộc vào một dịch vụ. Ví dụ, nếu bạn đang cấu hình máy chủ web Apache, bạn muốn máy chủ khởi động sau khi mạng trực tuyến. Điều này thường bao gồm các mục tiêu hoặc các dịch vụ khác.
* **Before**:- Bắt đầu dịch vụ hiện tại trước dịch vụ được chỉ định. Trong ví dụ này, tôi nói rằng "Tôi muốn máy chủ web Apache chạy trước khi dịch vụ cho Nextcloud được bắt đầu". Điều này là do, trong trường hợp của tôi, máy chủ Nextcloud phụ thuộc vào máy chủ web Apache. Điều này cũng **After**bao gồm các mục tiêu hoặc dịch vụ khác.

**Phần[Service]​**

Phần Dịch vụ chứa thông tin chi tiết về việc thực hiện và chấm dứt dịch vụ.

Dưới đây là các trường mà phần Dịch vụ có:

* **ExecStart**:- Lệnh cần được thực hiện khi dịch vụ bắt đầu. Trong trường hợp của chúng tôi, chúng tôi muốn máy chủ Apache khởi động.
* **ExecReload**:- Đây là trường tùy chọn. Nó chỉ định cách khởi động lại dịch vụ. Đối với các dịch vụ thực hiện I/O đĩa (đặc biệt là ghi vào đĩa, như cơ sở dữ liệu), tốt nhất là tắt chúng một cách nhẹ nhàng và khởi động lại. Sử dụng trường này trong trường hợp bạn muốn có cơ chế khởi động lại cụ thể.
* **Type**:- Điều này chỉ ra loại khởi động của một quy trình cho một dịch vụ systemd nhất định. Các tùy chọn là **simple**, **exec**, **forking**, **oneshot**, **dbus**, **notify**và **idle**. (thông tin thêm [tại đây](https://www.freedesktop.org/software/systemd/man/systemd.service.html#Options) )
* **Restart**:- Đây là một trường tùy chọn khác nhưng rất có thể bạn sẽ sử dụng. Trường này chỉ định liệu một dịch vụ có nên được khởi động lại hay không - tùy thuộc vào hoàn cảnh. Các tùy chọn khả dụng là **no**, **on-success**, **on-failure**, **on-abnormal**, **on-watchdog**, **on-abort**và **always**.

**Phần[Install]​**

Phần Cài đặt, như tên gọi, xử lý việc cài đặt tệp dịch vụ/đơn vị systemd. Phần này được sử dụng khi bạn chạy **systemctl enable**lệnh **systemctl disable**để bật hoặc tắt dịch vụ.

Dưới đây là các trường mà phần Cài đặt có:

* **WantedBy**:- Tương tự như các trường **After**và **Before**, nhưng điểm khác biệt chính là trường này được sử dụng để chỉ định "runlevel" tương đương systemd. Đây **default.target**là khi toàn bộ quá trình khởi tạo hệ thống hoàn tất--khi người dùng được yêu cầu đăng nhập. Hầu hết các dịch vụ hướng đến người dùng (như Apache, cron, GNOME-stuff, v.v.) đều sử dụng mục tiêu này.
* **RequiredBy**:- Trường này có thể rất giống với **WantedBy**, nhưng sự khác biệt chính là trường này chỉ định *các phụ thuộc cứng* . Nghĩa là, nếu phụ thuộc, dịch vụ này sẽ không thành công.

[**[ Unit] Chỉ thị Mục**](https://www.digitalocean.com/community/tutorials/understanding-systemd-units-and-unit-files#unit-section-directives)

Phần đầu tiên được tìm thấy trong hầu hết các tệp đơn vị là [Unit]phần. Phần này thường được sử dụng để xác định siêu dữ liệu cho đơn vị và cấu hình mối quan hệ của đơn vị với các đơn vị khác.

Mặc dù thứ tự phần không quan trọng systemdkhi phân tích tệp, phần này thường được đặt ở đầu vì nó cung cấp tổng quan về đơn vị. Một số chỉ thị phổ biến mà bạn sẽ tìm thấy trong [Unit]phần này là:

* **Description=**: Chỉ thị này có thể được sử dụng để mô tả tên và chức năng cơ bản của đơn vị. Chỉ thị này được trả về bởi nhiều systemdcông cụ khác nhau, vì vậy tốt nhất là đặt nó ngắn gọn, cụ thể và nhiều thông tin.
* **Documentation=**: Chỉ thị này cung cấp vị trí cho danh sách các URI để lập tài liệu. Chúng có thể là mancác trang có sẵn nội bộ hoặc URL có thể truy cập trên web. systemctl statusLệnh sẽ hiển thị thông tin này, cho phép dễ dàng khám phá.
* **Requires=**: Chỉ thị này liệt kê bất kỳ đơn vị nào mà đơn vị này về cơ bản phụ thuộc vào. Nếu đơn vị hiện tại được kích hoạt, các đơn vị được liệt kê ở đây cũng phải kích hoạt thành công, nếu không đơn vị này sẽ không hoạt động. Theo mặc định, các đơn vị này được khởi động song song với đơn vị hiện tại.
* **Wants=**: Chỉ thị này tương tự như Requires=, nhưng ít nghiêm ngặt hơn. Systemdsẽ cố gắng khởi động bất kỳ đơn vị nào được liệt kê ở đây khi đơn vị này được kích hoạt. Nếu các đơn vị này không được tìm thấy hoặc không khởi động được, đơn vị hiện tại sẽ tiếp tục hoạt động. Đây là cách được khuyến nghị để cấu hình hầu hết các mối quan hệ phụ thuộc. Một lần nữa, điều này ngụ ý kích hoạt song song trừ khi được sửa đổi bởi các chỉ thị khác.
* **BindsTo=**: Chỉ thị này tương tự như Requires=, nhưng cũng khiến đơn vị hiện tại dừng lại khi đơn vị liên quan kết thúc.
* **Before=**: Các đơn vị được liệt kê trong chỉ thị này sẽ không được khởi động cho đến khi đơn vị hiện tại được đánh dấu là đã khởi động nếu chúng được kích hoạt cùng lúc. Điều này không ngụ ý mối quan hệ phụ thuộc và phải được sử dụng kết hợp với một trong các chỉ thị trên nếu muốn.
* **After=**: Các đơn vị được liệt kê trong chỉ thị này sẽ được bắt đầu trước khi bắt đầu đơn vị hiện tại. Điều này không ngụ ý mối quan hệ phụ thuộc và phải thiết lập mối quan hệ này thông qua các chỉ thị trên nếu cần thiết.
* **Conflicts=**: Có thể sử dụng để liệt kê các đơn vị không thể chạy cùng lúc với đơn vị hiện tại. Việc khởi động một đơn vị có mối quan hệ này sẽ khiến các đơn vị khác dừng lại.
* **Condition...=**: Có một số chỉ thị bắt đầu bằng Conditionlệnh cho phép quản trị viên kiểm tra một số điều kiện nhất định trước khi khởi động đơn vị. Điều này có thể được sử dụng để cung cấp tệp đơn vị chung sẽ chỉ chạy khi trên các hệ thống phù hợp. Nếu điều kiện không được đáp ứng, đơn vị sẽ được bỏ qua một cách bình thường.
* **Assert...=**: Tương tự như các chỉ thị bắt đầu bằng Condition, các chỉ thị này kiểm tra các khía cạnh khác nhau của môi trường đang chạy để quyết định xem đơn vị có nên kích hoạt hay không. Tuy nhiên, không giống như Conditioncác chỉ thị, kết quả âm tính sẽ gây ra lỗi với chỉ thị này.

Sử dụng các chỉ thị này và một số chỉ thị khác, thông tin chung về đơn vị và mối quan hệ của nó với các đơn vị khác và hệ điều hành có thể được thiết lập.

[**[ Install] Mục Chỉ thị**](https://www.digitalocean.com/community/tutorials/understanding-systemd-units-and-unit-files#install-section-directives)

Ở phía đối diện của tệp đơn vị, phần cuối cùng thường là [Install]phần. Phần này là tùy chọn và được sử dụng để xác định hành vi hoặc đơn vị nếu nó được bật hoặc tắt. Việc bật một đơn vị sẽ đánh dấu đơn vị đó được tự động khởi động khi khởi động. Về bản chất, điều này được thực hiện bằng cách chốt đơn vị đang đề cập vào một đơn vị khác nằm ở đâu đó trong dòng đơn vị sẽ được khởi động khi khởi động.

Vì lý do này, chỉ những đơn vị có thể được kích hoạt mới có phần này. Các chỉ thị bên trong sẽ chỉ ra những gì sẽ xảy ra khi đơn vị được kích hoạt:

* **WantedBy=**: WantedBy=Chỉ thị là cách phổ biến nhất để chỉ định cách một đơn vị nên được bật. Chỉ thị này cho phép bạn chỉ định mối quan hệ phụ thuộc theo cách tương tự như chỉ Wants=thị trong [Unit]phần này. Sự khác biệt là chỉ thị này được bao gồm trong đơn vị phụ trợ cho phép đơn vị chính được liệt kê vẫn tương đối sạch. Khi một đơn vị có chỉ thị này được bật, một thư mục sẽ được tạo bên trong /etc/systemd/systemđược đặt tên theo đơn vị được chỉ định với phần .wantsđược thêm vào cuối. Trong đó, một liên kết tượng trưng đến đơn vị hiện tại sẽ được tạo, tạo ra sự phụ thuộc. Ví dụ, nếu đơn vị hiện tại có WantedBy=multi-user.target, một thư mục được gọi multi-user.target.wantssẽ được tạo bên trong /etc/systemd/system(nếu chưa có sẵn) và một liên kết tượng trưng đến đơn vị hiện tại sẽ được đặt bên trong. Việc tắt đơn vị này sẽ xóa liên kết và xóa mối quan hệ phụ thuộc.
* **RequiredBy=**: Chỉ thị này rất giống với WantedBy=chỉ thị, nhưng thay vào đó chỉ định một sự phụ thuộc bắt buộc sẽ khiến việc kích hoạt không thành công nếu không được đáp ứng. Khi được bật, một đơn vị có chỉ thị này sẽ tạo một thư mục kết thúc bằng .requires.
* **Alias=**: Chỉ thị này cũng cho phép đơn vị được kích hoạt dưới một tên khác. Trong số những mục đích sử dụng khác, điều này cho phép nhiều nhà cung cấp của một chức năng có sẵn, để các đơn vị liên quan có thể tìm kiếm bất kỳ nhà cung cấp nào có tên bí danh chung.
* **Also=**: Chỉ thị này cho phép bật hoặc tắt các đơn vị như một bộ. Các đơn vị hỗ trợ luôn có sẵn khi đơn vị này hoạt động có thể được liệt kê ở đây. Chúng sẽ được quản lý như một nhóm cho các tác vụ cài đặt.
* **DefaultInstance=**: Đối với các đơn vị mẫu (sẽ đề cập sau) có thể tạo ra các thể hiện đơn vị có tên không thể đoán trước, điều này có thể được sử dụng làm giá trị dự phòng cho tên nếu không cung cấp tên thích hợp.

[**Chỉ thị của từng đơn vị cụ thể**](https://www.digitalocean.com/community/tutorials/understanding-systemd-units-and-unit-files#unit-specific-section-directives)

Nằm giữa hai phần trước, bạn có thể sẽ tìm thấy các phần dành riêng cho từng loại đơn vị. Hầu hết các loại đơn vị đều cung cấp các chỉ thị chỉ áp dụng cho loại đơn vị cụ thể của chúng. Các chỉ thị này có sẵn trong các phần được đặt tên theo loại của chúng. Chúng tôi sẽ đề cập ngắn gọn về các chỉ thị này tại đây.

Các kiểu đơn vị device, target, snapshot, và scopekhông có chỉ thị cụ thể cho đơn vị nào và do đó không có phần liên quan nào cho kiểu của chúng.

**Phần [Service]**

Phần này [Service]được sử dụng để cung cấp cấu hình chỉ áp dụng cho các dịch vụ.

Một trong những điều cơ bản cần được chỉ định trong [Service]phần này là của Type=dịch vụ. Điều này phân loại các dịch vụ theo quy trình và hành vi daemonizing của chúng. Điều này quan trọng vì nó cho biết systemdcách quản lý đúng dịch vụ và tìm ra trạng thái của nó.

Chỉ Type=thị có thể là một trong những chỉ thị sau:

* **simple** : Quy trình chính của dịch vụ được chỉ định trong dòng bắt đầu. Đây là mặc định nếu các chỉ thị Type=và Busname=không được đặt, nhưng ExecStart=được đặt. Bất kỳ giao tiếp nào cũng phải được xử lý bên ngoài đơn vị thông qua một đơn vị thứ hai có loại thích hợp (như thông qua một .socketđơn vị nếu đơn vị này phải giao tiếp bằng ổ cắm).
* **forking** : Loại dịch vụ này được sử dụng khi dịch vụ fork một tiến trình con, thoát khỏi tiến trình cha gần như ngay lập tức. Điều này cho biết systemdtiến trình vẫn đang chạy mặc dù tiến trình cha đã thoát.
* **oneshot** : Kiểu này chỉ ra rằng quy trình sẽ diễn ra trong thời gian ngắn và cần systemdphải đợi quy trình thoát trước khi tiếp tục với các đơn vị khác. Đây là mặc định Type=và ExecStart=không được thiết lập. Nó được sử dụng cho các tác vụ một lần.
* **dbus** : Điều này chỉ ra rằng đơn vị sẽ lấy tên trên bus D-Bus. Khi điều này xảy ra, systemdsẽ tiếp tục xử lý đơn vị tiếp theo.
* **thông báo** : Điều này cho biết dịch vụ sẽ phát hành thông báo khi hoàn tất khởi động. Quá systemdtrình sẽ chờ điều này xảy ra trước khi tiến hành đến các đơn vị khác.
* **nhàn rỗi** : Điều này cho biết dịch vụ sẽ không được chạy cho đến khi tất cả các công việc được phân phối.

Một số chỉ thị bổ sung có thể cần thiết khi sử dụng một số loại dịch vụ nhất định. Ví dụ:

* **RemainAfterExit=**: Chỉ thị này thường được sử dụng với oneshotloại. Chỉ thị này cho biết dịch vụ phải được coi là hoạt động ngay cả sau khi quy trình thoát.
* **PIDFile=**:Nếu loại dịch vụ được đánh dấu là “forking”, chỉ thị này được sử dụng để thiết lập đường dẫn đến tệp chứa số ID tiến trình của tiến trình con chính cần được giám sát.
* **BusName=**: Chỉ thị này phải được đặt thành tên bus D-Bus mà dịch vụ sẽ cố gắng lấy khi sử dụng loại dịch vụ “dbus”.
* **NotifyAccess=**: Điều này chỉ định quyền truy cập vào ổ cắm sẽ được sử dụng để lắng nghe thông báo khi loại dịch vụ "thông báo" được chọn. Có thể là "none", "main" hoặc "all". Mặc định, "none" sẽ bỏ qua tất cả các thông báo trạng thái. Tùy chọn "main" sẽ lắng nghe các thông báo từ quy trình chính và tùy chọn "all" sẽ khiến tất cả các thành viên của nhóm kiểm soát dịch vụ được xử lý.

Cho đến nay, chúng ta đã thảo luận một số thông tin tiên quyết, nhưng chúng ta vẫn chưa thực sự xác định cách quản lý các dịch vụ của mình. Các chỉ thị để thực hiện việc này là:

* **ExecStart=**: Điều này chỉ định đường dẫn đầy đủ và các đối số của lệnh sẽ được thực thi để bắt đầu quy trình. Điều này chỉ có thể được chỉ định một lần (trừ các dịch vụ “oneshot”). Nếu đường dẫn đến lệnh được đặt trước bởi ký tự gạch ngang “-”, các trạng thái thoát khác không sẽ được chấp nhận mà không đánh dấu kích hoạt đơn vị là không thành công.
* **ExecStartPre=**: Có thể sử dụng lệnh này để cung cấp các lệnh bổ sung cần được thực thi trước khi tiến trình chính bắt đầu. Có thể sử dụng lệnh này nhiều lần. Một lần nữa, các lệnh phải chỉ định đường dẫn đầy đủ và có thể được đặt trước bằng “-” để chỉ ra rằng lỗi của lệnh sẽ được chấp nhận.
* **ExecStartPost=**: Điều này có cùng đặc điểm như ExecStartPre=ngoại trừ việc nó chỉ định các lệnh sẽ được chạy *sau khi* tiến trình chính được bắt đầu.
* **ExecReload=**: Chỉ thị tùy chọn này chỉ ra lệnh cần thiết để tải lại cấu hình của dịch vụ nếu có.
* **ExecStop=**: Điều này chỉ ra lệnh cần thiết để dừng dịch vụ. Nếu không cung cấp lệnh này, quy trình sẽ bị hủy ngay lập tức khi dịch vụ dừng.
* **ExecStopPost=**: Có thể sử dụng lệnh này để chỉ định các lệnh thực hiện sau lệnh dừng.
* **RestartSec=**:Nếu bật tính năng tự động khởi động lại dịch vụ, điều này sẽ chỉ định khoảng thời gian chờ trước khi thử khởi động lại dịch vụ.
* **Restart=**: Điều này cho biết các trường hợp mà systemddịch vụ sẽ cố gắng tự động khởi động lại. Điều này có thể được đặt thành các giá trị như “always”, “on-success”, “on-failure”, “on-abnormal”, “on-abort” hoặc “on-watchdog”. Những điều này sẽ kích hoạt khởi động lại theo cách dịch vụ đã dừng.
* **TimeoutSec=**: Cấu hình khoảng thời gian systemdchờ khi dừng hoặc dừng dịch vụ trước khi đánh dấu là không thành công hoặc buộc phải tắt dịch vụ. Bạn cũng có thể thiết lập thời gian chờ riêng biệt bằng TimeoutStartSec=và .TimeoutStopSec=

**Phần [Ổ cắm]**

Các đơn vị ổ cắm rất phổ biến trong systemdcác cấu hình vì nhiều dịch vụ triển khai kích hoạt dựa trên ổ cắm để cung cấp khả năng song song hóa và tính linh hoạt tốt hơn. Mỗi đơn vị ổ cắm phải có một đơn vị dịch vụ phù hợp sẽ được kích hoạt khi ổ cắm nhận được hoạt động.

Bằng cách phá vỡ kiểm soát socket bên ngoài dịch vụ, socket có thể được khởi tạo sớm và các dịch vụ liên quan thường có thể được khởi động song song. Theo mặc định, tên socket sẽ cố gắng khởi động dịch vụ cùng tên khi nhận được kết nối. Khi dịch vụ được khởi tạo, socket sẽ được chuyển đến dịch vụ đó, cho phép dịch vụ bắt đầu xử lý bất kỳ yêu cầu đệm nào.

Để chỉ định ổ cắm thực tế, các chỉ thị sau đây là phổ biến:

* **ListenStream=**: Điều này định nghĩa một địa chỉ cho một socket luồng hỗ trợ giao tiếp tuần tự, đáng tin cậy. Các dịch vụ sử dụng TCP nên sử dụng loại socket này.
* **ListenDatagram=**: Điều này định nghĩa một địa chỉ cho một socket datagram hỗ trợ các gói tin truyền thông nhanh, không đáng tin cậy. Các dịch vụ sử dụng UDP nên thiết lập loại socket này.
* **ListenSequentialPacket=**: Điều này định nghĩa một địa chỉ cho giao tiếp tuần tự, đáng tin cậy với các datagram có độ dài tối đa, bảo toàn ranh giới tin nhắn. Điều này thường được tìm thấy nhất đối với các socket Unix.
* **ListenFIFO**:Cùng với các kiểu nghe khác, bạn cũng có thể chỉ định bộ đệm FIFO thay vì ổ cắm.

Có nhiều loại chỉ thị lắng nghe khác nhau, nhưng những chỉ thị nêu trên là phổ biến nhất.

Các đặc điểm khác của ổ cắm có thể được kiểm soát thông qua các chỉ thị bổ sung:

* **Accept=**: Điều này xác định xem một phiên bản bổ sung của dịch vụ có được khởi động cho mỗi kết nối hay không. Nếu đặt thành false (mặc định), một phiên bản sẽ xử lý tất cả các kết nối.
* **SocketUser=**: Với socket Unix, chỉ định chủ sở hữu của socket. Đây sẽ là người dùng root nếu không được thiết lập.
* **SocketGroup=**: Với socket Unix, chỉ định chủ sở hữu nhóm của socket. Đây sẽ là nhóm gốc nếu cả this và above đều không được thiết lập. Nếu chỉ the SocketUser=được thiết lập, systemdsẽ cố gắng tìm một nhóm phù hợp.
* **SocketMode=**: Đối với socket Unix hoặc bộ đệm FIFO, điều này sẽ thiết lập quyền cho thực thể được tạo.
* **Service=**:Nếu tên dịch vụ không khớp với .sockettên, dịch vụ có thể được chỉ định bằng chỉ thị này.

**Phần [Núi]**

Đơn vị gắn kết cho phép quản lý điểm gắn kết từ bên trong systemd. Điểm gắn kết được đặt tên theo thư mục mà chúng kiểm soát, với thuật toán dịch được áp dụng.

Ví dụ, dấu gạch chéo đầu tiên bị xóa, tất cả các dấu gạch chéo khác được dịch thành dấu gạch ngang “-”, và tất cả các dấu gạch ngang và ký tự không in được được thay thế bằng mã thoát kiểu C. Kết quả của bản dịch này được sử dụng làm tên đơn vị gắn kết. Các đơn vị gắn kết sẽ có sự phụ thuộc ngầm định vào các gắn kết khác ở trên nó trong hệ thống phân cấp.

Các đơn vị gắn kết thường được dịch trực tiếp từ /etc/fstabcác tệp trong quá trình khởi động. Đối với các định nghĩa đơn vị được tạo tự động và các định nghĩa mà bạn muốn định nghĩa trong tệp đơn vị, các chỉ thị sau đây rất hữu ích:

* **What=**: Đường dẫn tuyệt đối đến tài nguyên cần được gắn kết.
* **Where=**: Đường dẫn tuyệt đối của điểm gắn kết nơi tài nguyên sẽ được gắn kết. Đường dẫn này phải giống với tên tệp đơn vị, ngoại trừ việc sử dụng ký hiệu hệ thống tệp thông thường.
* **Type=**: Kiểu hệ thống tập tin của mount.
* **Options=**: Bất kỳ tùy chọn gắn kết nào cần được áp dụng. Đây là danh sách được phân tách bằng dấu phẩy.
* **SloppyOptions=**: Giá trị boolean xác định liệu việc gắn kết có thất bại hay không nếu có tùy chọn gắn kết không được nhận dạng.
* **DirectoryMode=**: Nếu cần tạo thư mục cha cho điểm gắn kết, điều này sẽ xác định chế độ cấp phép của các thư mục này.
* **TimeoutSec=**: Cấu hình khoảng thời gian hệ thống sẽ chờ cho đến khi thao tác gắn kết được đánh dấu là không thành công.

**Phần [Automount]**

Đơn vị này cho phép một .mountđơn vị liên quan được tự động gắn kết khi khởi động. Cũng giống như .mountđơn vị, các đơn vị này phải được đặt tên theo đường dẫn của điểm gắn kết đã dịch.

Phần này [Automount]khá đơn giản, chỉ cho phép hai tùy chọn sau:

* **Where=**: Đường dẫn tuyệt đối của điểm automount trên hệ thống tệp. Đường dẫn này sẽ khớp với tên tệp ngoại trừ việc sử dụng ký hiệu đường dẫn thông thường thay vì bản dịch.
* **DirectoryMode=**:Nếu điểm gắn kết tự động hoặc bất kỳ thư mục cha nào cần được tạo, điều này sẽ xác định cài đặt quyền của các thành phần đường dẫn đó.

**Phần [Hoán đổi]**

Đơn vị hoán đổi được sử dụng để cấu hình không gian hoán đổi trên hệ thống. Các đơn vị phải được đặt tên theo tệp hoán đổi hoặc thiết bị hoán đổi, sử dụng cùng một bản dịch hệ thống tệp đã thảo luận ở trên.

Giống như các tùy chọn gắn kết, các đơn vị hoán đổi có thể được tạo tự động từ /etc/fstabcác mục nhập hoặc có thể được cấu hình thông qua một tệp đơn vị chuyên dụng.

Phần [Swap]của tệp đơn vị có thể chứa các chỉ thị sau để cấu hình:

* **What=**: Đường dẫn tuyệt đối đến vị trí của không gian hoán đổi, có thể là tệp hoặc thiết bị.
* **Priority=**: Điều này sử dụng một số nguyên biểu thị mức độ ưu tiên của hoán đổi đang được cấu hình.
* **Options=**: Bất kỳ tùy chọn nào thường được đặt trong /etc/fstabtệp đều có thể được đặt bằng chỉ thị này. Danh sách được phân tách bằng dấu phẩy được sử dụng.
* **TimeoutSec=**: Khoảng thời gian systemdchờ để hoán đổi được kích hoạt trước khi đánh dấu hoạt động là lỗi.

**Phần [Đường dẫn]**

Một đơn vị đường dẫn xác định đường dẫn hệ thống tệp systmedcó thể theo dõi các thay đổi. Một đơn vị khác phải tồn tại và sẽ được kích hoạt khi phát hiện hoạt động nhất định tại vị trí đường dẫn. Hoạt động đường dẫn được xác định thông qua inotifycác sự kiện.

Phần [Path]của một tệp đơn vị có thể chứa các chỉ thị sau:

* **PathExists=**: Chỉ thị này được sử dụng để kiểm tra xem đường dẫn có tồn tại hay không. Nếu có, đơn vị liên quan sẽ được kích hoạt.
* **PathExistsGlob=**: Điều này giống như trên, nhưng hỗ trợ biểu thức glob của tệp để xác định sự tồn tại của đường dẫn.
* **PathChanged=**: Điều này theo dõi vị trí đường dẫn để tìm kiếm thay đổi. Đơn vị liên quan được kích hoạt nếu phát hiện thay đổi khi tệp được theo dõi bị đóng.
* **PathModified=**: Lệnh này theo dõi những thay đổi giống như lệnh trên, nhưng nó kích hoạt khi ghi tệp cũng như khi tệp bị đóng.
* **DirectoryNotEmpty=**: Chỉ thị này cho phép systemdkích hoạt đơn vị liên quan khi thư mục không còn trống.
* **Unit=**: Điều này chỉ định đơn vị kích hoạt khi các điều kiện đường dẫn được chỉ định ở trên được đáp ứng. Nếu bỏ qua điều này, systemdsẽ tìm kiếm .servicetệp có cùng tên đơn vị cơ sở với đơn vị này.
* **MakeDirectory=**: Điều này xác định xem systemdcó tạo cấu trúc thư mục của đường dẫn đang đề cập trước khi theo dõi hay không.
* **DirectoryMode=**: Nếu tùy chọn trên được bật, nó sẽ thiết lập chế độ cấp phép cho bất kỳ thành phần đường dẫn nào phải được tạo.

**Phần [Timer]**

Đơn vị hẹn giờ được sử dụng để lên lịch các tác vụ hoạt động tại một thời điểm cụ thể hoặc sau một khoảng thời gian trễ nhất định. Loại đơn vị này thay thế hoặc bổ sung một số chức năng của daemon cronvà at. Phải cung cấp một đơn vị liên quan sẽ được kích hoạt khi đạt đến thời gian hẹn giờ.

Phần [Timer]của tệp đơn vị có thể chứa một số chỉ thị sau:

* **OnActiveSec=**: Chỉ thị này cho phép kích hoạt đơn vị liên quan tùy theo .timerkích hoạt của đơn vị đó.
* **OnBootSec=**: Chỉ thị này được sử dụng để chỉ định khoảng thời gian sau khi hệ thống khởi động khi thiết bị liên quan cần được kích hoạt.
* **OnStartupSec=**: Chỉ thị này tương tự như bộ đếm thời gian ở trên, nhưng liên quan đến thời điểm systemdquá trình thực sự bắt đầu.
* **OnUnitActiveSec=**: Tính năng này sẽ cài đặt bộ đếm thời gian dựa theo thời điểm thiết bị liên quan được kích hoạt lần cuối.
* **OnUnitInactiveSec=**: Điều này đặt bộ đếm thời gian theo thời điểm đơn vị liên quan được đánh dấu lần cuối là không hoạt động.
* **OnCalendar=**: Điều này cho phép bạn kích hoạt đơn vị liên quan bằng cách chỉ định giá trị tuyệt đối thay vì giá trị tương đối cho một sự kiện.
* **AccuracySec=**: Đơn vị này được sử dụng để thiết lập mức độ chính xác mà bộ hẹn giờ phải tuân thủ. Theo mặc định, đơn vị liên quan sẽ được kích hoạt trong vòng một phút sau khi bộ hẹn giờ đạt đến. Giá trị của chỉ thị này sẽ xác định giới hạn trên của cửa sổ trong đó systemdlịch trình kích hoạt sẽ diễn ra.
* **Unit=**: Chỉ thị này được sử dụng để chỉ định đơn vị sẽ được kích hoạt khi hết thời gian. Nếu không được đặt, systemdsẽ tìm kiếm .serviceđơn vị có tên trùng với đơn vị này.
* **Persistent=**: Nếu cài đặt này được thiết lập, systemdthiết bị liên quan sẽ được kích hoạt khi bộ hẹn giờ hoạt động nếu bộ hẹn giờ này được kích hoạt trong khoảng thời gian bộ hẹn giờ không hoạt động.
* **WakeSystem=**: Thiết lập chỉ thị này cho phép bạn đánh thức hệ thống khỏi trạng thái tạm dừng nếu hết thời gian hẹn khi ở trạng thái đó.

**Phần [Slice]**

Phần [Slice]của tệp đơn vị thực sự không có bất kỳ .slicecấu hình đơn vị cụ thể nào. Thay vào đó, nó có thể chứa một số chỉ thị quản lý tài nguyên thực sự khả dụng cho một số đơn vị được liệt kê ở trên.

Một số chỉ thị chung trong [Slice]phần này, cũng có thể được sử dụng trong các đơn vị khác, có thể được tìm thấy trong systemd.resource-controltrang hướng dẫn. Những chỉ thị này có hiệu lực trong các phần cụ thể của đơn vị sau:

* [Slice]
* [Scope]
* [Service]
* [Socket]
* [Mount]
* [Swap]