Por: Alice Dantas

Crontab – Tarefas Agendadas

O **Crontab** (acrônimo de *cron table*) é um utilitário usado para agendar tarefas (comandos, scripts ou programas) que serão executadas automaticamente em intervalos de tempo específicos.

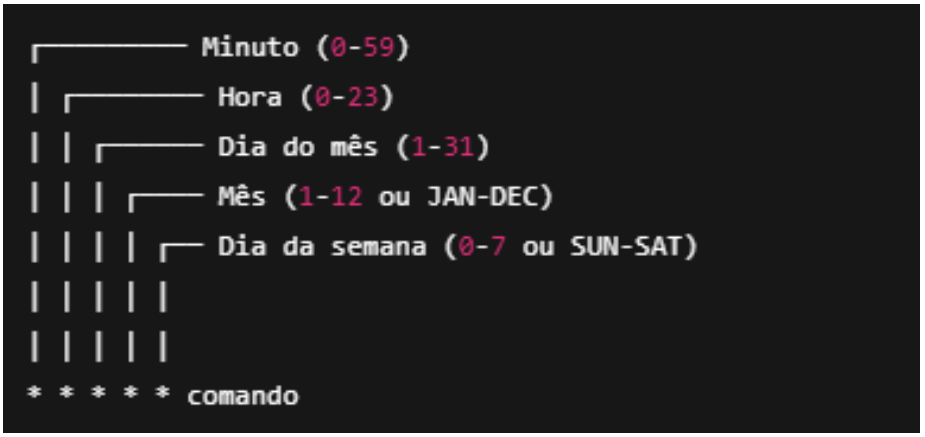
O serviço que executa essas tarefas é chamado de **cron daemon** (o nome vem do conceito de espíritos ou seres invisíveis da mitologia grega, que trabalhavam silenciosamente realizando tarefas), que roda constantemente em segundo plano, verificando os agendamentos definidos nos arquivos crontab e executando as tarefas no momento correto.

- Como funciona

O cron lê as tabelas de configuração (crontab) de cada usuário. Cada linha do crontab representa uma tarefa agendada, que contém:

- **Expressão de tempo:** Define quando a tarefa será executada.
- **Comando:** A ação que o sistema executará.

- Sintaxe da Crontab



- Caracteres Especiais na Crontab

Caractere	Nome	Função Principal
*	Coringa	Qualquer valor (todos os possíveis)
,	Separador de valores	Lista de valores
-	Intervalo	Faixa de valores
/	Passo	Executar de tanto em tanto tempo

Por: Alice Dantas

Entendendo melhor os caracteres especiais

* → Coringa (Qualquer valor)

Significa que vai rodar para todos os valores daquela coluna.

* * * * * comando → Executa **todo minuto, de todas as horas, todos os dias, todos os meses, todos os dias da semana.** Ou seja, o tempo inteiro.

0 5 * * * comando → Executa todos os dias às 5h da manhã. O * * * nos últimos três campos significa "todos os dias, todos os meses, qualquer dia da semana".

, → Lista de Valores (Escolho alguns)

Funciona como uma vírgula em uma lista.

0 5,17 * * * comando → Executa às **5h da manhã E às 17h, todos os dias.**

30 8,14,20 * * * comando → Executa às **8:30, 14:30 e 20:30 todos os dias.**

- → Intervalo (De... até...)

Significa uma faixa de valores. "De tal coisa até tal coisa."

0 9-17 * * * comando → Executa **de hora em hora, das 9h até 17h, todos os dias.**

*/5 9-17 * * * comando → Executa **a cada 5 minutos, durante o horário de 9h às 17h.** PS: O asterisco (*) na frente do / funciona como uma base, como se dissesse: "Em todos os valores possíveis dessa coluna, faça de tanto em tanto."

/ → Passo (De quanto em quanto)

É como dizer: "A cada tanto".

*/10 * * * * comando → Executa **a cada 10 minutos, todos os dias.**

0 */2 * * * comando → Executa **de 2 em 2 horas**, ou seja, às 0h, 2h, 4h, 6h, ..., 22h.

0 9-17/2 * * * comando → Executa **entre às 9h e 17h, de 2 em 2 horas.** Ou seja, 9h, 11h, 13h, 15h, 17h.

5-45/10 * * * * → Isso executa **entre os minutos 5 a 45, de 10 em 10 minutos, de cada hora, todos os dias.** Aqui não tem o *, pois você especificou o intervalo inicial (5-45). Se você não especifica, ele usa o padrão *, que significa "do menor ao maior possível".

Por: Alice Dantas

- Redirecionando saídas para arquivos de log

Ao executar qualquer comando *no terminal* ou *via Crontab*, ele gera saídas, que podem ser:

- **Saída padrão** → Resultados normais, como mensagens de sucesso, retornos, prints, etc.
- **Saída de erro** → Mensagens de erro, quando algo dá errado.

Se você não diz pra onde essas saídas vão, *no terminal,* elas aparecem na tela. *No Crontab*, como o comando roda em segundo plano, se você não redirecionar, essas saídas somem no limbo. Por isso você cria um arquivo de log, para que tudo seja registrado.

A sintaxe básica é:

`[comando] >> arquivo.log 2>&1` → **>>** adiciona (append) a saída padrão no arquivo arquivo.log. Se não existir, ele cria. Já **2>&1** envia a **saída de erro (2)** para o mesmo lugar que a **saída padrão (1)**, ou seja, também pro arquivo.log.

PS: O **&** indica que o que vem depois é um "descritor de arquivo" (número), e não um nome de arquivo. Um descritor de arquivo é simplesmente um número inteiro que o sistema operacional usa para representar um recurso de entrada ou saída (I/O).
Os descritores padrões do terminal são:

Número	Nome	Função
0	stdin	Entrada padrão (teclado, entrada)
1	stdout	Saída padrão (tela, resultado normal)
2	stderr	Saída de erro (mensagens de erro)

Então, `2>&1` diz: “jogue os erros (stderr) no mesmo lugar onde a saída padrão (stdout) foi.”

Exemplo:

`0 2 * * * /home/alice/backup.sh >> /home/alice/logs/backup.log 2>&1` → Executa o script de backup **todos os dias às 2h da manhã**, e tudo que o script fizer (ou errar) será salvo no arquivo backup.log.

Isso é fundamental no Crontab, porque, como os comandos rodam sem você ver, sem log você nunca saberia:

- Se o backup rodou.
- Se o script deu erro.
- Se faltou permissão.

Por: Alice Dantas

- Comandos para Gerenciar o Crontab do usuário

Comando	Descrição
crontab -e	Editar o crontab do usuário atual
crontab -l	Listar os agendamentos do usuário atual
crontab -r	Remover todos os agendamentos do usuário
crontab -u usuário -e	Editar o crontab de outro usuário (root)

Importante saber:

Existem outros lugares no sistema onde podem estar tarefas agendadas que não aparecem no **crontab -e do usuário**. São os **crontabs do sistema**, que são configurados em arquivos como:

Local	Função
/etc/crontab	Crontab do sistema.
/etc/cron.d/	Diretório com arquivos de agendamento específico de serviços.
/etc/cron.daily/	Scripts que rodam **diariamente**.
/etc/cron.hourly/	Scripts que rodam **de hora em hora**.
/etc/cron.weekly/	Scripts que rodam **semanalmente**.
/etc/cron.monthly/	Scripts que rodam **mensalmente**.

→ Essas tarefas são configuradas geralmente pelo próprio sistema, administradores ou programas que você instala.