



# **Automatizar tarefas, administração com agendamento de trabalhos**

# Sumário

## Capítulo 1

Automatizar tarefas, administração com agendamento de trabalhos .....3

1.1. Mãos a obra.....4

## Capítulo 2

Gerenciando.....5

2.1. Objetivos.....5

2.1. Troubleshooting.....5

## Índice de tabelas

## Índice de Figuras

# **Capítulo 1**

## **Automatizar tarefas, administração com agendamento de trabalhos**

- Gerenciar cron e jobs;
- Configurar o acesso do usuário ao cron e em serviços.

## 1.1. Mãos a obra

O administrador de sistema em ambientes GNU/Linux gerencia muitas tarefas no dia a dia, como por exemplo a administração de usuários, segurança aplicada no servidor, tarefas de backup e entre outras mais. A automatização dessas tarefas é feita através de scripts, que são agendados periodicamente conforme a necessidade da empresa em questão.



*Como posso agendar a execução de meus scripts?*

O agendamento no sistema pode ser feito através dos comandos `at` e `crontab`, a diferença entre eles esta no sistema de agendamento.

**at** → Usado para agendamento único;

**cron** → Agendamentos periódicos.

### Opções de uso do crontab:

`-e` → Permite editar tarefas agendadas do usuário logado;



```
# crontab -e
```

```
# m h dom mon dow  command
```

Ao usar o comando uma linha de cabeçalho é exibido informando a função de cada coluna.

**m** → Define o minuto do agendamento onde é possível usar de 0 a 59;

**h** → Define a hora do agendamento onde é possível usar de 0 a 59;

**dom** (day of month) → Define o dia do mês do agendamento onde é possível usar de 1 a 31;

**mon** (month) → Define o mês do agendamento onde é possível usar de 1 a 12;

**dow** (day of week) → Define o dia da semana do agendamento onde é possível usar de 0 a 7;

**comand** → Caminho completo do binario ou script.

### **Exemplo:**

Executar um script personalizado no primeiro dia de cada mês às 23:30h

```
30 23 1 * * /root/scripts/backup.sh
```

### **Intervalos**

Na configuração de minutos, horas, dias, meses e dias da semana você pode usar os seguintes intervalos:

, (Vírgula) → Lista de valores : 1,4,5;

- (traço) → Intervalo de valores: 1-5;

\* (asterisco) → Tudo;

/ (barra) → Pulos entre valores: \*/5

**-l** → Permite listar as tarefas agendadas do usuário logado;



```
# crontab -l
```

```
# m h dom mon dow  command
30 23 1 * * /root/scripts/backup.sh
```

**-lu <usuário>** → Permite listar as tarefas agendadas um determinado usuário;



```
# crontab -lu aluno
```

```
no crontab for aluno
```

**-r** → Apaga o arquivo do usuário de um determinado usuário;



```
# crontab -r
```

## Agendamento geral

Através do arquivo `/etc/crontab` é possível configurar o agendamento geral do crontab usando diretórios para escutar scripts de forma periódica.



```
# vim /etc/crontab
```

```
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
```

### **Detalhes do arquivo /etc/crontab:**

**Shell** → Indica qual o tipo de Shell será utilizado para interpretar os scripts;

**Path** → Variável que armazena os diretórios onde ficam os arquivos; executáveis do sistema.

**Run-parts** → Comando usado para executar binários ou scripts de um diretório

### **Diretórios:**

/etc/cron.hourly : Executa de hora em hora;

/etc/cron.daily : Executa todo dia;

/etc/cron.weekly : Executa uma vez por semana;

/etc/cron.monthly : Executa uma vez por mês.

### **Restringir acesso**

Qualquer usuário pode criar, listar e excluir agendamentos no sistema, mas é possível liberar ou bloquear esse acesso através dos arquivos cron.allow e cron.deny no diretório /etc.

Bloquear usuário



```
# vim /etc/cron.deny
```

Digite o nome dos usuários que não poderão realizar agendamentos

Ex.: aluno

Abra um outro terminal e se logue com o usuário bloqueado, e use o comando `crontab -e` para criar um novo agendamento.



```
$ crontab -e
```

```
You (aluno) are not allowed to use this program (crontab)
See crontab(1) for more information
```



# Capítulo 2

## Gerenciando

### 2.1. Objetivos

•Troubleshooting: Trabalhar com os arquivos e comandos: /var/spool/cron/\*, crontab, at, atq, atrm

### 2.1. Troubleshooting



*Como gerenciar o agendamento de meus usuários?*

O administrador além de liberar ou bloquear os agendamentos dos usuários, ele pode também excluir quando necessário agendamentos únicos e periódicos.

Os agendamentos do cron feitos por usuário comuns, ficam armazenados no diretório /var/spool/cron/crontabs/<nome\_do\_usuario>.

Exemplo:



```
# ls -l /var/spool/cron/crontabs
```

```
total 4  
-rw----- 1 tux crontab 263 Out  7 00:26 tux
```

Em nosso exemplo o usuário tux tem agendamentos no cron.

Para visualizar o agendamento do usuário use o comando cat ou crontab -lu



```
# cat /var/spool/cron/crontabs/tux
```

ou



```
# crontab -lu tux
```

Excluir agendamento do usuário



```
# crontab -ru tux
```

### **Criar agendamento único com at**

O comando at pode ser usado por qualquer usuário para criar agendamentos únicos. Para usar o comando digite at hora:minuto (Enter), em seguida digite o comando a ser executado (Enter) e CTRL + D para gravar o agendamento. Vamos a prática:



*\$ at 21:05*

```
warning: commands will be executed using /bin/sh
at> /home/tux/script.sh
at> <EOT>
job 3 at Mon Oct 11 21:05:00 2010
```

Listar agendamentos com o comando atq



*\$ atq*

```
3 Mon Oct 11 21:05:00 2010 a tux
```

Exibe conteúdo do agendamento com o comando tail



*\$ tail -5 /var/spool/cron/atjobs/\**

```
    echo 'Execution directory inaccessible' >&2
    exit 1
}
/home/tux/script.sh
```

Remover agendamentos

Comando atrm n°\_do\_job



*\$ atrm 3*

O comando `at` também aceita outras opções para agendamentos. Veja a descrição:

`at now` → Agora;

`at tomorrow` → Amanhã;

`at today` → Hoje;

`at now + 10min` → Dentro de 10 minutos;

`at noon tomorrow` → Às 12:00 do próximo dia.

### Agendamento com mensagem de email

Através da opção `-m` do comando `at` é possível receber um email quando a tarefa é executada. Vamos a prática:



```
$ at -m 21:55
```

```
warning: commands will be executed using /bin/sh
at> /home/tux/script.sh
at> <EOT>
job 4 at Mon Oct 11 21:55:00 2010
```

Verifique um novo email no diretório `/var/mail/<nome_do_usuario>`



```
$ cat /var/mail/tux
```

## Restringir acesso

Qualquer usuário pode criar, listar e excluir agendamentos no sistema, mas é possível liberar ou bloquear esse acesso através dos arquivos `at.allow` e `at.deny` no diretório `/etc`.

Bloquear usuário



```
# vim /etc/at.deny
```

Digite o nome dos usuários que não poderão realizar agendamentos

Ex.: tux

Abra um outro terminal e se logue com o usuário bloqueado, e use o comando `at` para criar um novo agendamento.



```
$ at 22:00
```

```
You do not have permission to use at.
```