

awk 系列：如何使用 awk 和正则表达式过滤文本或文件中的字符串

2016-7-18 08:15

编译自：<http://www.tecmint.com/use-linux-awk-command-to-filter-text-string-in-files/>

作者：Aaron Kili

原创：LCTT <https://linux.cn/article-7586-1.html>

译者：wwy-hust



```
tecmint@tecmint ~ $ awk '/{print}' deals.txt
```

当我们在 Unix/Linux 下使用特定的命令从字符串或文件中读取或编辑文本时，我们经常需要过滤输出以得到感兴趣的部分。这时正则表达式就派上用场了。

什么是正则表达式？

正则表达式可以定义为代表若干个字符序列的字符串。它最重要的功能之一就是它允许你过滤一条命令或一个文件的输出、编辑文本或配置文件的一部分等等。

正则表达式的特点

正则表达式由以下内容组合而成：

- **普通字符**，例如空格、下划线、A-Z、a-z、0-9。
- 可以扩展为普通字符的**元字符**，它们包括：
 - **(.)** 它匹配除了换行符外的任何单个字符。
 - **(*)** 它匹配零个或多个在其之前紧挨着的字符。
 - **[character(s)]** 它匹配任何由其中的字符/字符集指定的字符，你可以使用连字符(-)代表字符区间，例如 [a-f]、[1-5]等。
 - **^** 它匹配文件中一行的开头。
 - **\$** 它匹配文件中一行的结尾。

你必须使用类似 **awk** 这样的文本过滤工具来过滤文本。你还可以把 **awk** 自身当作一个编程语言。但由于这个指南的适用范围是关于使用 **awk** 的，我会按照一个简单的命令行过滤工具来介绍它。

awk 的一般语法如下：

```
# awk 'script' filename
```

此处 '**script**' 是一个由 **awk** 可以理解并应用于 **filename** 的命令集合。

它通过读取文件中的给定行，复制该行的内容并在该行上执行脚本的方式工作。这个过程会在该文件中的所有行上重复。

该脚本 '**script**' 中内容的格式是 '**/pattern/ action**'，其中 **pattern** 是一个正则表达式，而 **action** 是当 **awk** 在该行中找到此模式时应当执行的动作。

如何在 Linux 中使用 awk 过滤工具

在下面的例子中，我们将聚焦于之前讨论过的元字符。

一个使用 **awk** 的简单示例：

下面的例子打印文件 **/etc/hosts** 中的所有行，因为没有指定任何的模式。

```
# awk '{print}' /etc/hosts
```



```
tecmint@tecmint ~ $ awk '{print}' /etc/hosts
```

awk 打印文件中的所有行

结合模式使用 **awk**

在下面的示例中，指定了模式 **localhost**，因此 **awk** 将匹配文件 **/etc/hosts** 中有 **localhost** 的行。

```
# awk '/localhost/{print}' /etc/hosts
```

```
tecmint@tecmint ~ $ awk '/localhost/{print}' /etc/hosts
```

awk 打印文件中匹配模式的行

在 awk 模式中使用通配符 (.)

在下面的例子中，符号 (.) 将匹配包含 loc、localhost、localnet 的字符串。

这里的正则表达式的意思是匹配 **l** 一个字符 **c**。

```
# awk '/l.c/{print}' /etc/hosts
```

```
tecmint@tecmint ~ $ awk '/l.c/{print}' /etc/hosts
```

使用 awk 打印文件中匹配模式的字符串

在 awk 模式中使用字符 (*)

在下面的例子中，将匹配包含 localhost、localnet、lines, capable 的字符串。

```
# awk '/l*c/{print}' /etc/localhost
```

```
tecmint@tecmint ~ $ awk '/l*c/{print}' /etc/hosts
```

使用 awk 匹配文件中的字符串

你可能也意识到 `(*)` 将会尝试匹配它可能检测到的最长的匹配。

让我们看一看可以证明这点的例子，正则表达式 `t*t` 的意思是在下面的行中匹配以 `t` 开始和 `t` 结束的字符串：

```
this is tecmint, where you get the best good tutorials, how to's,
guides, tecmint.
```

当你使用模式 `/t*t/` 时，会得到如下可能的结果：

```
this is t
this is tecmint
this is tecmint, where you get t
this is tecmint, where you get the best good t
this is tecmint, where you get the best good tutorials, how t
this is tecmint, where you get the best good tutorials, how tos, guides,
t
this is tecmint, where you get the best good tutorials, how tos, guides,
tecmint
```

在 `/t*t/` 中的通配符 `(*)` 将使得 awk 选择匹配的最后一项：

```
this is tecmint, where you get the best good tutorials, how to's,
guides, tecmint
```

结合集合 [character(s)] 使用 awk

以集合 `[a1]` 为例，awk 将匹配文件 `/etc/hosts` 中所有包含字符 `a` 或 `l` 或 `1` 的字符串。

```
# awk '/[a1]/{print}' /etc/hosts
```

```
tecmint@tecmint ~ $ awk '/[a1]/{print}' /etc/hosts
```

使用 *awk* 打印文件中匹配的字符

下一个例子匹配以 **K** 或 **k** 开始头，后面跟着一个 **T** 的字符串：

```
# awk '/[Kk]T/{print}' /etc/hosts
```

```
tecmint@tecmint ~ $ awk '/[Kk]T/{print}' /etc/hosts
```

使用 *awk* 打印文件中匹配的字符

以范围的方式指定字符

awk 所能理解的字符：

- **[0-9]** 代表一个单独的数字
- **[a-z]** 代表一个单独的小写字母
- **[A-Z]** 代表一个单独的大写字母
- **[a-zA-Z]** 代表一个单独的字母
- **[a-zA-Z 0-9]** 代表一个单独的字母或数字

让我们看看下面的例子：

```
# awk '/[0-9]/{print}' /etc/hosts
```

```
tecmint@tecmint ~ $ awk '/[0-9]/{print}' /etc/hosts
```

使用 *awk* 打印文件中匹配的数字

在上面的例子中，文件 `/etc/hosts` 中的所有行都至少包含一个单独的数字 `[0-9]`。

结合元字符 (^) 使用 *awk*

在下面的例子中，它匹配所有以给定模式开头的行：

```
# awk '/^fe/{print}' /etc/hosts  
# awk '/^ff/{print}' /etc/hosts
```

```
tecmint@tecmint ~ $ awk '/^fe/{print}' /etc/hosts
```

使用 *awk* 打印与模式匹配的行

结合元字符 (\$) 使用 *awk*

它将匹配所有以给定模式结尾的行：

```
# awk '/ab${print}' /etc/hosts  
# awk '/ost${print}' /etc/hosts  
# awk '/rs${print}' /etc/hosts
```

```
tecmint@tecmint ~ $ awk '/ab${print}' /etc/hosts
```

使用 `awk` 打印与模式匹配的字符串

结合转义字符 (\) 使用 `awk`

它允许你将该转义字符后面的字符作为文字，即理解为其字面的意思。

在下面的例子中，第一个命令打印出文件中的所有行，第二个命令中我想匹配具有 \$25.00 的一行，但我并未使用转义字符，因而没有打印出任何内容。

第三个命令是正确的，因为一个这里使用了一个转义字符以转义 \$，以将其识别为 '\$'（而非元字符）。

```
# awk '//{print}' deals.txt
# awk '/$25.00/{print}' deals.txt
# awk '/\ $25.00/{print}' deals.txt
```

```
tecmint@tecmint ~ $ awk '//{print}' deals.txt
```

结合转义字符使用 `awk`

总结

以上内容并不是 `awk` 命令用做过滤工具的全部，上述的示例均是 `awk` 的基础操作。在下面的章节中，我将进一步介绍如何使用 `awk` 的高级功能。感谢您的阅读，请在评论区贴出您的评论。

via: <http://www.tecmint.com/use-linux-awk-command-to-filter-text-string-in-files/>

本文由 **LCTT** ^[4] 原创编译, **Linux中国** ^[5] 荣誉推出

[1]: <http://www.tecmint.com/author/aaronkili/>

[2]: <https://github.com/wwy-hust>

[3]: <https://github.com/wxy>

[4]: <https://github.com/LCTT/TranslateProject>

[5]: <https://linux.cn/article-7586-1.html?pr>