Biểu Thức Chính Quy (Regex) trong Linux

1. Giới Thiệu về Regex

Biểu thức chính quy (Regular Expressions - Regex) là công cụ mạnh mẽ để tìm kiếm và xử lý văn bản theo mẫu.

2. Ký Tự Cơ Bản

Ký Tự Đơn

```
.  - Khớp với bất kỳ ký tự đơn nào
^  - Đầu dòng
$  - Cuối dòng
\  - Escape ký tự đặc biệt
```

Tập Hợp Ký Tự

```
[abc]   - Khớp với a, b, hoặc c
[^abc]  - Khớp với bất kỳ ký tự nào trừ a, b, c
[a-z]   - Khớp với chữ cái thường
[A-Z]   - Khớp với chữ cái hoa
[0-9]   - Khớp với chữ số
```

Số Lần Xuất Hiện

```
*      - 0 hoặc nhiều lần
+      - 1 hoặc nhiều lần
?      - 0 hoặc 1 lần
{n}    - Chính xác n lần
{n,}   - Ít nhất n lần
{n,m}  - Từ n đến m lần
```

3. Ví Dụ Với grep

Tìm Kiếm Cơ Bản

```
# Tìm từ 'error' trong file
grep "error" file.txt

# Tìm dòng bắt đầu bằng 'Start'
grep "^Start" file.txt
```

```
# Tìm dòng kết thúc bằng 'end'
grep "end$" file.txt
```

Tìm Kiếm Nâng Cao

```
# Tìm số điện thoại (format: XXX-XXX-XXXX)
grep -E "[0-9]{3}-[0-9]{3}-[0-9]{4}" file.txt

# Tìm địa chỉ email
grep -E "[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}" file.txt

# Tìm IP address
grep -E "([0-9]{1,3}\.){3}[0-9]{1,3}" file.txt
```

4. Ví Dụ Với sed

Thay Thế Văn Bản

```
# Thay thế 'old' bằng 'new'
sed 's/old/new/g' file.txt

# Thay thế chỉ lần xuất hiện đầu tiên trên mỗi dòng
sed 's/old/new/1' file.txt

# Thay thế từ lần xuất hiện thứ hai trở đi
sed 's/old/new/2g' file.txt
```

Xóa Dòng

```
# Xóa dòng trống
sed '/^$/d' file.txt

# Xóa dòng bắt đầu bằng #
sed '/^#/d' file.txt

# Xóa khoảng trắng đầu dòng
sed 's/^[ \t]*//' file.txt
```

5. Ví Dụ Với awk

Xử Lý Dữ Liệu

```
# In cột đầu tiên của các dòng chứa 'error'
awk '/error/ {print $1}' file.txt

# Tính tổng cột số
awk '{sum += $3} END {print sum}' file.txt

# Đếm số dòng khớp với mẫu
awk '/pattern/ {count++} END {print count}' file.txt
```

6. Các Mẫu Regex Hữu Ích

Kiểm Tra Định Dạng

```
# Kiểm tra số nguyên
^[0-9]+$

# Kiểm tra số thập phân
^[0-9]+\.[0-9]+$

# Kiểm tra URL
^(http|https):\/\/[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,}
```

Xử Lý Tệp Log

```
# Tìm lỗi trong log
grep -E "error|warning|critical" log.txt

# Tìm IP address và timestamp
grep -E "([0-9]{1,3}\.){3}[0-9]{1,3}.*\[.*\]" access.log
```

7. Lời Khuyên

1. **Kiểm Tra Regex**
    - Luôn test regex với dữ liệu mẫu nhỏ trước
    - Sử dụng công cụ online để kiểm tra regex
2. **Tối Ưu Hóa**
    - Tránh sử dụng regex quá phức tạp
    - Sử dụng nhóm bắt (\1, \2) khi cần tham chiếu
3. **Bảo Mật**
    - Cẩn thận khi sử dụng regex với sed -i
    - Backup dữ liệu trước khi thực hiện thay thế hàng loạt

4. **Hiệu Suất**
    - Sử dụng anchor (^ $) khi có thể
    - Tránh backreference và lookahead/lookbehind khi không cần thiết

8. Ví Dụ Thực Tế

Phân Tích Log Apache

```
# Tìm các request POST
grep -E '" POST .*" [0-9]{3} ' access.log

# Đếm số lượng request theo IP
awk '{print $1}' access.log | sort | uniq -c | sort -nr

# Tìm các request có thời gian phản hồi > 1s
awk '$NF > 1 {print $0}' access.log
```

Xử Lý Dữ Liệu CSV

```
# Lọc các dòng có email hợp lệ
awk -F',' '$3 ~ /^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}$/ {print}'
data.csv

# Tính tổng cột số
awk -F',' '{sum += $4} END {print sum}' data.csv
```