# BASH Lab 5 Parsing and Searching

Regular Expressions are very powerful tools for searching and parsing text. They are available in in most shells and programming languages. Therefore, it is important to know the basics of regular expressions.

1) Do the first five lessons from <http://regexone.com/>. In the site, there is a link to the solution just below the box for your answer in case you get stuck. If you view the solution, make sure you understand it before you move on.

2) Use echo and grep -E to test the answers from the review exercise on slide 8 of Parsing and Searching. For example, this will test the first example for social security numbers.

These pass the regex:

[john@localhost ~]$ echo "123-45-6789" | grep -E "[0-9]{3}[- ]?[0-9]{2}[- ]?[0-9]{4}"

123-45-6789

[john@localhost ~]$ echo "123 45 6789" | grep -E "[0-9]{3}[- ]?[0-9]{2}[- ]?[0-9]{4}"

123 45 6789

[john@localhost ~]$ echo "123456789" | grep -E "[0-9]{3}[- ]?[0-9]{2}[- ]?[0-9]{4}"

123456789

This does not pass the regex:

[john@localhost ~]$ echo "a23-45-6789" | grep -E "[0-9]{3}[- ]?[0-9]{2}[- ]?[0-9]{4}"

3) Let’s assume you have come across malware where the file name fits this pattern:  
 Four characters that are upper- or lower-case letters,  
 two numbers,  
 and a file extension of either .exe, .bat, .cmd, or .dll

Write a regular expression to match them. Your regex should match these:  
 evil01.bat  
 abcd24.dll  
 crud99.exe  
 viru45.dll

These names should not be matched:  
 sys01.bat  
 fred1.exe  
 virusfile45.dll  
 evil10.jpg

To speed your testing, you can put all the test cases above into a text file, and then use commands like this:  
cat test.txt | grep –E “your regex here”  
or  
grep –E “your regex here” test.txt

4) On slide 9, what directories are being searched in the example? Write a command to cat all the .html files in /var/www.

Note: There are different flavors of regular expression. The regexone.com site uses Perl Compatible Regular Expressions (PCRE), while grep does not use PCRE by default. The BASH grep does not support the \d “any digit” abbreviation for [0-9], unless you replace the –E flag with –P to enable PCRE.  
 echo "123-45-6789" | grep -P "\d{3}[- ]?\d{2}[- ]?\d{4}" does work,  
 echo "123-45-6789" | grep -E "\d{3}[- ]?\d{2}[- ]?\d{4}" does not

# Hand in

Hand in your answers to questions 3 and 4.