Perl Basics for Pentesters Part 1

ALIEN CODERS

Sanjeev Jaiswal (Jassi)

Perl Programmer and Security Enthusiast

#nullhyd (2016)

What we will cover

Part 1

Perl Introduction

Control structures and loops

Functions to memorize

Perl data Types

Special Variable

File handling

Part 2

Regular Expression

Perl Helpers

Perl codes basic examples

Modules to know

Scripts for Pentesting

Future Scope

Demo of tools like dnsenum, fierce, nikto, sqlninja

Perl Fundamentals

http://www.aliencoders.org/

Perl or perl or PERL?

- When you refer a programming language say it **Perl**
- When you refer a script , let's say **perl**
- But never ever say **PERL,** use perl or Perl

***Perl mongers** and **Larry Wall** don't like it ;-)*

Perl has some **backronyms** though

Practical **E**xtraction and **R**eport **L**anguage, or

Pathologically **E**clectic **R**ubbish **L**ister.

And its **Perl** not **Pearl**

http://www.aliencoders.org/

Installing perl

- Try **perl -v** to check if it's installed or not

Unix/Linux

- Run curl -L http://xrl.us/installperlnix | bash in terminal

OSX

- Install command line toll Xcode
- Run curl -L http://xrl.us/installperlnix | bash in terminal

Windows

- install strawberry perl or activestate perl

Then install **cpan App::cpanminus** to install perl modules easily in future

Executing perl program

- perl <perl_program>
- chmod 755 and execute ./<perl_program>

Let’s try something more on CLI

- perl –d <perl_program> #Diagonise more
- perl –c <perl_program> #check if syntax is ok
- perl -e 'print "perl one-liner\n";'
- perl one-liner examples (palindrome, inplace-editing)

Who's who in Perl ;)

- shebang i.e #!
- print, say
- #comment
- $calar, @rray, %ash
- Comparison operators (> or gt <= or le)
- Reference in Perl
- %INC and @INC

Basic Example in Perl ;)

```
#!/usr/bin/perl #Shebang starts with #!
use strict;
use warnings;
# It's a comment and its just the basic
my $name = "Sanjeev Jaiswal"; #scalar
my $id = 10; # scalar
my $sal = 100.98; #scalar
my @name = ("Sanjeev", "Jaiswal"); #array
my %hash = ('fname'=>'Sanjeev', 'lname', 'Jaiswal'); #hash

print "$id, $name[0], $hash{'lname}\n";
print "$name\n" if ( $id < 100 );
```

http://www.aliencoders.org/

Loop Control

http://www.aliencoders.org/

Loop and control structures

- if, if else, if elsif else
- for, foreach
- while, do while
- next, unless, last
- return, exit

Loop and control structures

```
while(<>){
  next if /^\d+/;
  last if /^\W/;
  print $_;
}

print $_ foreach(1 .. 100);
print if(10 <= 10.0);
if($name eq 'sanjeev'){
  print "$name\n";
} elsif ($id >70){
  print "$id\n";
} else {
  print "not matched\n";
}
```

Functions to memorize

http://www.aliencoders.org/

Minimal functions you should know

- shift , push and chomp
- sort and reverse
- exec, system and eval
- warn, die
- join and split
- keys, values, each
- exists, defined, delete, unlink

Minimal examples ;)

- `chomp (my $u_input = <STDIN>); #chomps the user input`
- `my $f_elem = shift @array; # assign first element of an array`
- `push @arr, $elem; # Adding $elem at the last of @arr`
- `@sorted_num   = sort {$a <=> $b} @unsorted_num; #sort integer array`
- `@reverse_sort = sort {$b <=> $a} @unsorted_num; #reverse sort`
- `@reverse_sort = reverse sort @unsorted_arr # reverse sort of string array or`
- `@reverse_sort = sort {$b cmp $a} @unsorted_arr`
- `warn "Very high\n" if($num > 10);`
- `die "Very low\n" if($num < 2);`
- `system("ls -la", "dir" )`
- `exec("/bin/cat", "/home.txt");`
- ``` `ls -la`; #avoid backtick if possible ```
- `join(/\s/ , @array);`
- `split(/\s/, $string);`

http://www.aliencoders.org/

Perl File Handlers

http://www.aliencoders.org/

Manipulate file handling

- open(), close()
- >, >>, <
- +>, +>>, +<
- File testing -e, -f, -d, -s, -m etc.
- opendir, closedir, readdir

File Handling examples

```
open(FH, "<", "filename") or die "can't open: $!\n";
# > for write and >> for append
while ( defined(my $line = <FH>) ) { do something .. }
close(FH);

open(LS, "<", "ls -la|"); # use instead of ``
open(FIND, "find . -type f -name dns_info.pl |-"); #better than previous command

do something if -e $file; # -e means exists, -f is for file and -d for directory
do something if -s >0; #-s is for size and -m means modified

$dir = "/home/sanjeev/";
opendir ( DIR, $dir ) || die "Error in opening directory $dir\n";
while( ($file = readdir(DIR))){
     next if $file =~ m/\.{1,2}/;
     print("$file\n") if -f $file;
}
closedir(DIR);
```

ALIEN CODERS

Perl Special Variables

Most used special variables

- $0 – name of perl script being executed
- $^O – O.S.
- $! – current value of errno in scalar and string in list context
- $@ - error message from the last eval, do-FILE, or require command
- $_ - default input and search pattern space
- @_ - arguments passed to the given subroutine
- $$ - process number of the running program
- $? – status returned by the last pipe close, back tick or system command

Questions

http://www.aliencoders.org/