

Unix System Administration

Chris Schenk

Lecture 24 – Tuesday Apr 15

Tax Day. Woo.

CSCI 4113, Spring 2008

Administrivia

- Guest lecture by Matthew Woitaszek
 - This Thursday April 17
 - High performance super-computing and clustering
- Quiz02 is available
 - Due in a week
- One more required lab – Perl
- One extra credit lab – LDAP
- Final Exam released last week of classes
 - Due at 10:00am Tuesday of finals week via email

Perl Basics

- Variables
 - @var – array, \$var – scalar, %var – hashmap
- Special variables @_ and \$_
 - Default variables for many functions
 - Reading a file puts each line into \$_
 - Calling 'print' without a variable prints \$_
 - Subroutines put arguments into array @_
- Open and read file syntax
 - open(HANDLE, “</path/to/file_to_read.txt”);
 - while(<HANDLE>) { }

Perl Regular Expressions

- Are the bomb
- Can match on variables or on `$_`
 - `if(/some regex/) { ... } -- uses $_`
 - `if($var =~ /regex/) { ... } -- uses $var`
- Can capture parts of the regex into variables
 - `if($var =~ /re(ge)x/ { print $1; } -- prints 'ge'`
- Can also ignore case with options at the end
 - `/another ReGeX/i -- 'i' ignores case`
- Do straight substitution
 - `'s/shit to replace/new shit to replace it with/'`

Perl + MACs

- Message Authentication Codes
 - Used to authenticate plain-text data
- 'emailreport' has no authentication
 - Actually a security hole, information leak
- MACs allow us to send plain text and a tag
 - Tag is verified at the other end
- MACs use a shared key on both ends
 - 'rndc' for BIND uses this exact concept

Perl + MACs (cont)

- Use the Digest::HMAC_SHA1 module (after installing libdigest-hmac-perl)
 - Use `Digest::HMAC_SHA1`;
- Create a new HMAC object to create a tag
 - `my $hmac = Digest::HMAC_SHA1->new("shared key");`
- Add data to be digested
 - `$hmac->add("just another perl hacker,");`
`my $digest = $hmac->b64digest();`
- Now, on the other end, verify tag before performing any tasks (such as sending email)!

Perl + MySQL

- Use the DBI module (after installing `libdbd-mysql-perl` package in Ubuntu)
 - `use DBI;`
`use DBD::mysql;`
- Define and set your variables
 - `my $user = "monitor";`
`my $pass = "wearewatchingyou";`
`my $db = "login_monitor";`
`my $host = "localhost";`
- Create a connect string and connect
 - `$cs = "dbi:mysql:$db:$host:3306";`
`$conn = DBI->connect($cs, $user, $pass);`

Perl + MySQL (cont)

- **Prepare a statement and execute**

```
- $stmt = $conn->prepare("SELECT * from  
logins");  
$stmt->execute();
```

- **Bind columns (results) and print them out**

```
- $stmt->bind_columns(undef, \$id, \$username,  
\$ip, \$time);  
while($stmt->fetch()) {  
    print "$username, $ip, $time\n";  
}
```

- **Close the connection**

```
- $conn->disconnect();
```