CS 410: Web Security

A1 (Part 1): Labs

WFP1: Command injection

- **Example #1**
    - **Probe the page to see how the URL parameters are used**
    - **Inject a command to dump the password file on server**
- **Example #2**
    - **Filter validates that an IP address is given via a regular expression (regexp)**
    - **Test if the filter handles newlines properly.**
    - **Recall the use of URL encoding to inject special characters**
    - **Bypass filter to obtain a directory listing**
- **Example #3**
    - **Filter redirects user to error page if injection detected, but does not terminate command**
    - **Use telnet, nc, or python to perform injection. Ignore redirects to obtain the results of `uname -a`**
    - **Note that if you use the echo command in the manual and want to give it a newline (`\n`), then echo needs to be called with the `-e` flag to allow for the ‘\’ to specify expressions (i.e. \n as a newline or `\x20` as a space)**

WFP1: Code injection

- **Example #1**
    - **The PHP that processes user input is**
      `$str = “echo \”Hello “.$_GET[‘name’].”!!!\”;”;`
    - **Examine error messages to identify the PHP function that consumes user input**
    - **Inject a call to `system` and attach the results of running `system(‘uname -a’)`**
    - **To ensure PHP script does not fail, do one of the following**
        - **Inject a comment at the end to eliminate PHP code after injection (e.g. ‘//’)**
        - **Terminate command and start another to consume subsequent code**

```
      “ . system(“echo hi”); $dummy=“
```

- **Example #2**
    - **PHP script does sorting via a lambda function based on user input**
    - **`usort` calls `create_function` with input supplied by the user**

```
sprintf(eval_code, "function " LAMBDA_TEMP_FUNCNAME "(%s){%s}",Z_STRVAL_PP(z_function_args),Z_STRVAL_PP(z_function_code));
…
retval = zend_eval_string(eval_code, NULL, eval_name TSRMLS_CC);
```

  - **The call is injectable by terminating function declaration and appending code**
  - **Test which one of these patterns generate syntax errors**
    - **Try** `?order=id;}//`
    - **Try** `?order=id);}//`
    - **Try** `?order=id));}//`
  - **Inject** `system()` **command to obtain output of “**`uname -a`**”**
- **Example #3**
  - **PHP script uses PCRE (Perl Compatible Regular Expression) to modify string**
  - **PHP API allows “**`/e`**” option PCRE_REPLACE_EVAL that allows PHP to** `eval` **new value as PHP code**
  - **Add the option to the pattern (e.g.** `pattern=/lamer/e`**) and interpret the error**
  - **Add code to the** `new` **parameter to return the result of a PHP call to** `phpinfo()`
- **Example #4**
  - **Inject characters to see which ones break the PHP script**
  - **Which call is the user input being evaluated in?**
  - **Does this call treat the string as PHP code?**
  - **Break out of the string syntax, inject a call to** `phpinfo()`**, and finish the syntax to obtain a similar output as #3**