> *Security and performance—It is extremely important that the program filter out any possible harmful information received from outside the application. It is much easier to remove the harmful data initially, or reject the data initially, before it has been used or saved. With major breaches in security today, this is an absolute MUST DO.*

You could have also restricted which programs called the PHP program. The $__SERVER global variable can assist you.

```
if($__SERVER['HTTP_REFERER'] == 'http://www.mysite.com/lab.html')
{ //  code executed if valid location }
else {exit;}
```

> *For more information on the $__SERVER array, visit*
> *Examples*: *http://php.net/manual/en/reserved.variables.server.php*
> *Video*: *https://www.thenewboston.com/videos.php?cat=11&video=17047*

This example code would reject any program that calls the lab.php program (except for the lab.html program that is located at the site). The exit; command closes the program if it is not called from the correct HTML page.

The filter methods demonstrated will not keep someone from entering "asabsbabsa" as a dog name. However, these methods will keep any entry from being harmful.Example 4-4. Partial list of the bottom of lab.php with the clean_input method

```
else
{
print "<p>Missing or invalid parameters. Please go back to the lab.html page to enter valid
information.<br />";
print "<a href='lab.html'>Dog Creation Page</a>";
}
```

The true part of the if statement mentioned includes all the active code in the lab.php file. If one or more parameters is missing, the else section (at the bottom of the lab.php file) will request that the user go back to the lab.html page to properly enter the information.

Do It

Adjust the code in the new lab.php file (download it from the book's web site). Add code to filter for bad gender code and make sure that gender information has been received from the HTML file. Make sure to pass your property through the clean_input method to remove any harmful data.

Additional HTML Input Security

As you can see from all the code you have seen in this chapter so far, whenever a text box is used on an HTML form, additional code in several areas must be included to validate what the user has typed. Text boxes are necessary when you need to allow the user flexibility in what they can enter (such as a form that includes name and address). However, you can use other form objects when you want to limit the user's response to a particular list of possible values (such as two-letter abbreviations for states). This would provide more valid data because the user won't be able to enter a typo or enter invalid data.