ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ

B. Abdukadirov

SECURITY SUPPORT IN THE LANGUAGE OF PERL SERVER SCRIPTS

(Fergana branch Tashkent University of Information Technologies
named after Muhammad al-Khwarizmi)

In 1995, the ubiquity of the Web was nothing more than a dream, and the programming languages and Web technologies resembled children's games. As a result, hackers were simply not interested in penetrating Web sites. However, at present the picture has changed and individual successful implementations have been replaced by a large variety of languages and technologies ready for use by hackers. And they every day "pluck" the unsuspecting employees of the company, organizations or government agencies. Drop any preconceived ideas about security. Without understanding the rules of the game, you just have to go with the flow. The main purpose of the description of programming languages is to present to Web programmers and security specialists the languages currently used, the functionality, syntax and potential risks associated with the use of these languages.

In order for two computers to "communicate" with each other, they must be properly programmed and "speak" in one language. There are many popular Web languages, and each one has its own advantages and disadvantages. On the one hand, easy and easy to learn HTML is probably the best choice. It has a simple style and is based only on a few complex concepts. On the other hand, if you need to provide interactivity, dynamic data updating, and a complex graphical display of information, it's better to use the Java language. However, in any case, in order to understand the way Web-server interacts with the client browser, and therefore, to find the vulnerabilities in the implementation of this process, one must have an idea of the main technologies.

Perl is a high-level programming language often called the scripting language in 1987 was created by Larry Wall. Currently, Perl is the most portable scripting language that can be used on AS / 400, supercomputers Cray, Digital VMS, MPE / ix from Hewlett Packard, Linux, Tandem, MacOS and all versions of Unix. Portability of Perl, its low cost, it is distributed free of charge and robustness contributed to its wide application on the Internet. The ego served as one of the most significant reasons for the explosive growth of the worldwide network.

The Perl language is extremely stable and flexible. It can be used to perform server operations, implement client scripts, or create standalone applications, for example, the universal dispatcher of distribution lists. However, the main purpose of Perl is to manage Web server scripts. At the same time, security was never a distin-

guishing feature of this language. As a result, for Web sites that use Perl, there are various flaws in the security subsystem. True, there are several ways to reduce the risk of a security breach.

The Perl code can be either very simple or extremely complex. Typically, the Perl language is used to process data entered by the user in the form fields.

Perl can be used as the language of server-side scripts, or just plan to do it, take into account a number of security breaches and corresponding countermeasures.

- Make sure that Web servers do not start with administrative privileges, i.e. with the rights of the account root (Unix) or Administrator (Windows). Otherwise, there is a danger of a cracker executing commands with high privileges.
- Always perform pre-processing of field values. Define a list of valid alphanumeric characters for the application, and then filter out any characters that do not belong to this set. For example, if there is a field for entering an e-mail address, you can use the regular expression pattern to identify incorrect information. This will identify the error and inform the user about the need to correct the input data. In any electronic address the following characters are allowed: a ... z, A..Z, 0-9, only one symbol e. hyphen (-), underscore (_), and period (.). Below is a simple expression that allows you to identify a dangerous idea, within which a field for entering an e-mail address

```
If ($email !~ /* [ \w.-]+\! e[\w.-|*$/ ) {
        print "<br><br> # Warning: An error has been detected in your
        email address. Re-enter the data. <br>"
} else {
        # Executing the rest of the Perl script
}
```

This code fragment can be deciphered as follows:

```
/       # Start the regular expression
*       # Sets the beginning of the line
[       #Defines the beginning of the list of characters
\w      # Defines an alphanumeric character including "_"
*       # Specifies the point "."
_       # Defines a dash or hyphen "-"
]       # End of the list of characters
+       #Sets 1 or more characters from the previous list
¥e      # Matches the "e"
[       # Defines the beginning of the list of characters
\w      # Specifies an alphanumeric character including "_"
.       # Specifies the point
-       # Specifies a dash or hyphen "–"
]       # End of the list of characters
```

If the condition !" Specified in the Perl expression is not fulfilled, the script returns an error message and the user will have to correct the input information.

- Limit the use of local operating system commands. When passing the parameters of the open () function. Systes (), fork (), or exec () allow the cracker to execute the system hacking commands. If you still need to use these functions, be sure to check the input variables.
- On Unix systems, you should not use the values of environment variables. Instead, set the $ PATH and $ IFS variables only in script files.
  $ENV{"PATH") = "/bin:/usr/bin:/usr/local/bin:/opt";
  $ENV{"IFS"} = " / ";
  If you do not explicitly manage these variables, the attacker can change their values and cause the program to execute another command, rather than the one that was planned.
- Check the size of the input variables or data entered in the form fields. To do this, you either need to check the length of the variables passed to the program, or use the $ ENV {CONTENT_LENGTH} field to limit the size of the data sent in the POST requests and sometimes the GET. If this is not done, the attacker will be able to send a large amount of data to the variable and lead to the crash of the Web-server, the system, or, worse, to overflow the buffer and execute arbitrary commands remotely.
- Try not to accept the path from the form fields. At least make sure that the path is relative, not absolute. Also, keep track of the dot (..) characters or the forward / backslash (/ or \). Otherwise, the attacker will be able to generate a request for a UNIX password file
  ../../../../../../../etc/passwd
  or a request to receive a backup copy of the Windows SAM file
  ../../../../../../../winnt/repaire/sam._
  If possible, use the validation check of the variable values used.
- By default, Perl scripts are stored as plain text. Therefore, after hacking into the Web-server, you can read the Perl files and extract various information from them, for example, user names and passwords for accessing the database. Several programs, such as Perl2exe (http://www.perl2exe.com), allow you to hide the Perl code. When they are used, an independent executable (.exe) is created, which makes the Perl source code and the interpreter unnecessary.

In general, the protection of Perl scripts is critical, so find or develop yourself a good function of analyzing input data and apply its day checking the value of each field of the user form.

References

1. Stuart M., Saumil S., Shriray S. Hacking in Web Attacks and Protection - Addison-Wesley Boston 2003 - 384 p.