Activity overview

Previously, you discussed how the Bash shell helps you communicate with a computer’s operating system.

When you communicate with the shell, the commands in the shell can take input and return output or error messages.

In this lab activity, you’ll use the `echo` command to examine how input is received and how output is returned in the shell. Next, you’ll use the `expr` command to further explore input and output while performing some basic calculations in the shell.

This activity will build foundations in understanding how you communicate with the Linux operating system through the shell. As a security analyst, you'll need to input commands into the shell and recognize when the shell returns either output or an error message.

Next, you'll explore the scenario!

Scenario

As a security professional, it’s important to understand the concept of communicating with your computer via the shell.

In this scenario, you have to input a specified string of text that you want the shell to return as output. You'll also need to input a few mathematical calculations so the OS (operating system) can return the result.

Here’s how you’ll do this: **First**, you’ll use the `echo` command to generate some output in the shell. **Second**, you’ll use the `expr` command to perform basic mathematical calculations. **Next**, you’ll use the `clear` command to clear the Bash shell window. **Finally**, you’ll have an opportunity to explore the `echo` and `expr` commands further.

Get ready to examine input and output in the Bash shell!

Task 1. Generate output with the echo command

The `echo` command in the Bash shell outputs a specified string of text. In this task, you'll use the `echo` command to generate output in the Bash shell.

1. Type `echo hello` into the shell and press **ENTER**.

The command to complete this step:

```
1  echo hello
```

The `hello` string should be returned:

```
1  hello
```

The command `echo hello` is the **input** to the shell, and `hello` is the **output** from the shell.

1. Rerun the command, but include quotation marks around the string data. Type `echo "hello"` into the shell and press **ENTER**.

The command to complete this step:

```
echo "hello"
```

The `hello` string should be returned again:

```
hello
```

Task 2. Generate output with the expr command

In this task, you'll use the `expr` command to generate some additional output in the Bash shell. The `expr` command performs basic mathematical calculations and can be useful when you need to quickly perform a calculation.

Imagine that the system has shown you that you have 32 alerts, but only 8 required action. You want to calculate how many alerts are false positives so that you can provide feedback to the team that configures the alerts.

1. Calculate the number of false positives using the `expr` command.

Type `expr 32 - 8` into the shell and press **ENTER**.

The command to complete this step:

```
1  expr 32 - 8
```

The following result should be returned:

```
1  24
```

Note: *The* `expr` *command requires that all terms and operators in an expression are separated by spaces. For example:* `expr 32 - 8`*, and* ***not*** `expr 32-8`.

2. Type `expr 3500 * 12` into the shell and press **ENTER**.

The command to complete this step:

```
1   expr 3500 * 12
```

The correct result should now be returned:

```
1   42000
```

Task 3. Clear the Bash shell

In this task, you'll use the `clear` command to clear the Bash shell of all existing output. This allows you to start with the cursor at the top of the Bash shell window.

When you work in a shell environment, the screen can fill with previous input and output data. This can make it difficult to process what you're working on. Clearing the screen allows you to create a clutter-free text environment to allow you to focus on what is important at that point in time.

- Type `clear` into the shell and press **ENTER**.

The command to complete this step:

```
1   clear
```

Note: *All previous commands and output will be cleared, and the user prompt and cursor will return to the upper left of the shell window.*