<COMPANY NAME>

TECHNICAL WRITER ENGINEERING

BRUNO AUGUSTO CAETANO COURA

TECHNICAL WRITING CASE

Answers of Technical Writing Case

SANTA RITA DO SAPUCAÍ

2021

Summary

1. Exercise 1

1.1. Compromised AWS Account Credentials

The document aims to describe helpful steps for the responsible person from Information Security (Infosec) and all employees to deal with possible leaked Amazon Web Services (AWS) credentials.
If the information security is compromised, the expert from Infosec needs to remove the user permissions in AWS, revoke the credentials, and generate new ones.

1.1.1. How to Remove the User Permissions in AWS?

Prerequisites:

A person who administers the AWS Identity and Access Management (IAM) permissions must be found. So, use the following command:

```
nu sec iam show group infosec-permissions-admin
```

Result:

{Suggestion: Insert a screenshot with the result.}

The specialist from Infosec can execute the subsequent actions to remove the user permissions.

Steps:

1. Remove all inline policies from the affected IAM user typing the instruction:

```
nu iam disallow <user> Source
```

Result: All inline policies must be removed.

2. Encounter all IAM groups for the affected user through the request:

```
nu sec iam show <user>
```

Result:

{Suggestion: Insert a screenshot with the result.}

3. Remove the user from all groups with:

```
nu sec iam remove <user> <groups>
```

Result: The user must be excluded from all groups.

> **Tip!**
> Use the command `nu sec iam show <user>` to verify if the user is not linked with any group.

4. Communicate the relevant stakeholders using the Slack Messaging Platform:

Example: Send the following message to the attacked user:

> *"Dear <User Name>,*
> *Your AWS permissions have been temporarily suspended pending an investigation into a possible compromise.*
> *Do not hesitate to contact Infosec if you have any concerns. Please, keep the incident private until the problem is solved."*

Example: Inform the Technical Manager or Engineering Manager with the text:

"Dear <Technical Manager Name> or <Engineering Manager Name>, There is a suspicious activity related to your engineer account in AWS. As a result, your AWS permissions have been temporarily blocked while an investigation is ongoing. Infosec has taken all appropriate actions.
Do not hesitate to contact Infosec if you have any concerns. Please, keep the incident private until the problem is solved."

The specialist determined by the Incident Lead in this section keeps articulating the steps to prevent the occurrence of possible attacks. The events involve the revocation of affected AWS credentials. For more details, refer to How to Revoke the User Credentials in AWS?.

Note!
The context of the leaked key can or cannot be known.

1.1.1.1. Visual Representation of AWS Permissions Removal Process

The following Unified Modeling Language (UML) Sequence Diagram describes the steps that remove the AWS permissions.

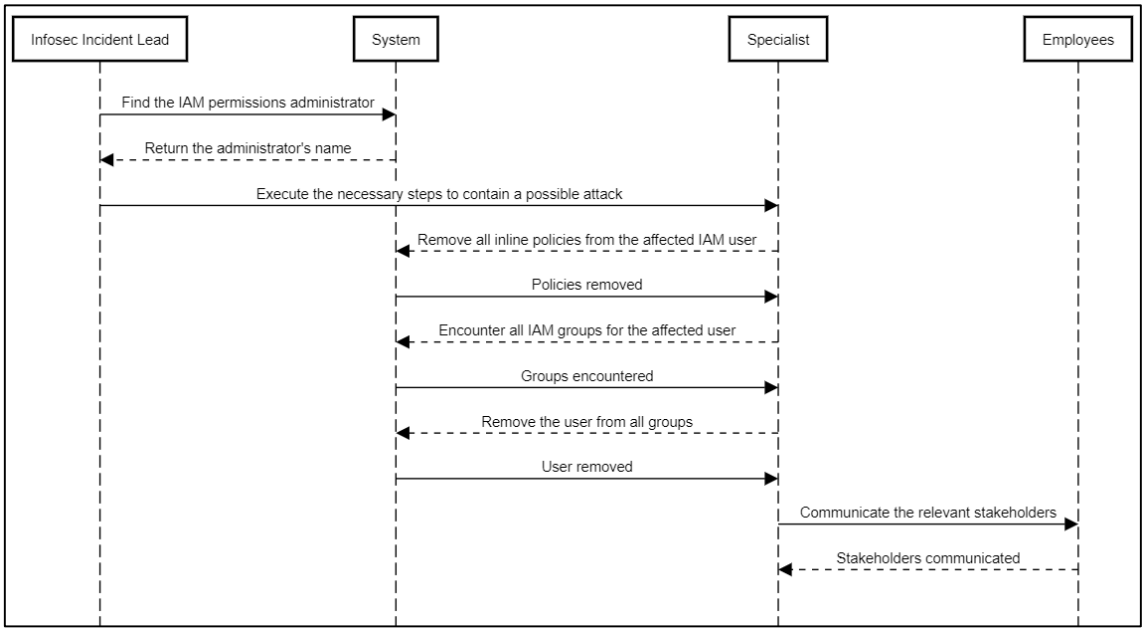

Figure 1 – AWS Permissions Removal Process.

1.1.2. How to Revoke the User Credentials in AWS?

At first, it is necessary to execute the same instruction described in the **Prerequisites**, located in the How to Remove the User Permissions in AWS?, to find a specialist who manages the IAM permissions.
The specialist can disable or delete the affected credentials using the AWS console. With this same console, the expert generates new credentials and a new key pair. The coming action is to send a message from Slack to interested employee. The text must contain the key pair and the instructions to update the AWS keys wherever they are referenced.
The employee can update the AWS keys utilizing a type of file used for configuring the user environment, the **bash_profile**. As options, the employee can work with one of the following files:

- **setupnu.sh** in https://github.com/<company name>/master/setupnu.sh.
- **nu-setup** in https://github.com/<company name>/nu-setup.

The IAM permissions administrator and employees can obtain more data about credentials, consulting a complete guide elaborated for the developers. For more information, refer to Working with AWS Credentials.

1.1.1.2. Visual Representation of AWS Credentials Revoke Process

The following Unified Modeling Language (UML) Sequence Diagram illustrates the steps that revoke the affected AWS credentials.

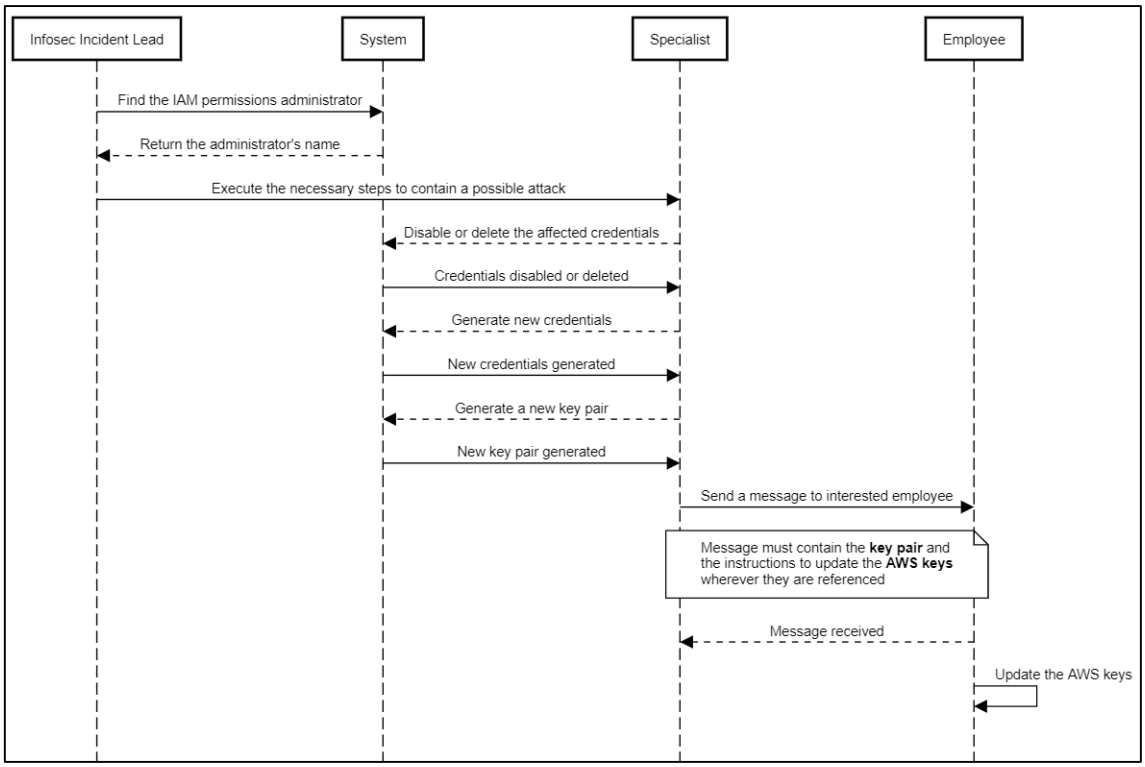

Figure 2 – AWS Credentials Revoke Process.

2. Exercise 2

2.1. Tips to Elaborate a Documentation for the Team

The tips present in this section aim to assist an engineer in writing a technical document. Based on Exercise 1, the composition process can contain the following items and sub-items:

- First Page
- Summary
- Titles and Subtitles
- Content
    - Prerequisites
    - Steps
    - Commands
    - Results
    - Examples
    - Tips and Notes

- Links
- UML Sequence Diagram

As extra information, ask for a team member to review the documentation. Also, create a versioning process because the document can have constant updates.

Note!

Microsoft Word is the tool used to develop this Technical Writing Case.

2.1.1. First Page

The **First Page** is the primary contact the reader has with your document. It works as a *business card*. It is a way to stimulate the person to read your report.
The recommendation is to insert the name of the interested area, the author name, the title, the subtitle, when necessary, the location, and the year of publication of the document.

2.1.2. Summary

The **Summary** enumerates the titles and subtitles of the document, informing their respective pages.
The reader has an overview of the content, facilitating and directing to the desired information. For more details, see the **Summary** contained in this document.

2.1.3. Titles and Subtitles

Titles and **Subtitles** are fundamental in building a good document. Think of a self-explanatory text.
It is recommended to establish a hierarchy, with different levels limiting up to 5 levels maximum. Use the **Heading 1** for the **Titles** and from **Heading 2** to **Heading 5** for the **Subtitles**. Also, utilize the **Ordered List** icon in the **Paragraph** pane to enumerate each title and subtitle.

2.1.4. Content

There are some questions that the writer needs to answer: *What is the purpose of this document? What is the target audience?*
Start with a short description of the intention of the content. If the information is in step by step, the writer can describe the **Prerequisites**, if applicable, and the **Steps** to reach the final result with success.
If the actions are ordered as a list, the instructions can initiate with an infinitive verb. For more details, refer to How to Remove the User Permissions in AWS?. The operations can also be in a continuous text, as in How to Revoke the User Credentials in AWS?.
O text can assume the following setting:

- Font: Arial
- Font size: 12
- Line spacing: 1.5

If a word, which has an acronym, is used for the first time, it is advisable to put the meaning accompanied by the abbreviated form in parentheses. *Amazon Web Services (AWS)* is an example. Do not use expressions that can sound aggressive: master-slave, black book, and others. Also, avoid abbreviations in a technical document. For example:

- Infosec's taken all appropriate actions.
- Infosec has taken all appropriate actions.

The **Commands**, which are essential to execute a specific event, can be set with the **Courier New** font or other different types. The **Results** can be text or a figure, as a screenshot. The **Examples**, **Tips**, and **Notes** are also relevant to help the reader with extra information.

2.1.5. Links

The **Links** point to internal contents, which are in the same document, and external, as is the link to Working with AWS Credentials. The **Links** become references that help the reader in search of more knowledge and understanding.

2.1.6. UML Sequence Diagram

UML Sequence Diagram is a type of visual representation regarding the sequences of a process. To build the diagrams, add the flows by using the https://sequencediagram.org/. Other sites also offer tools that can be used. The charts present in Figure 1 and Figure 2 have the sequences in the following files:

Figure 1 – AWS Permissions Removal

Figure 2 – AWS Credentials Revoke Pr

3. Exercise 3

3.1. API Document Template

This Application Programming Interface (API) Document Template supports the reader to implement a code that searches for the number of occurrences of a word in a given text.

Steps:

1. Create a `<method_name>` method with a string called `<variable_name>` and `<string_array_name_1>` as an array of words.

   Example:

   ```
   public static void <method_name>(String
   <variable_name>, String[] <string_array_name_1>) {
   }
   ```

2. Use the `if` and `else` functions to test the conditions.

If the length of the <string_array_name_1> is greater than <number_of_words>, the software displays an error message informing that there are too many words.

Example:

```
if (<string_array_name_1>.length > <number_of_words>)
{
     System.err.println("Too many words!");
}
```

3. Use the `else` function to test what happens when the length of <string_array_name_1> is less than or equal to <number_of_words>.
   a. Declare an integer array that receives the value that the **`length`** function returns from <string_array_name_1>.

      Example:

      ```
      int[] <int_array_name> = new
      int[<string_array_name_1>.length];
      ```

   b. Implement the `for` structure to repeat a block code a certain amount of times.
      Start with zero (`int i = 0`), adding one unit (`i++`) until reaching the length of <string_array_name_1>.

      Example:

      ```
      for (int i = 0; i < <string_array_name_1>.length;
      i++) {
      }
      ```

c. Declare a string array that receives the value that the **`split`** function returns from `<variable_name>`.

The **`split`** function can use the following characters:

- `\"`: double quotes character
- `\'`: single quote character
- `\t`: tab space character
- `\n`: new line character
- `\b`: back space character
- `\f`: form feed character
- `\r`: carriage return character

Example:

```
String[] <string_array_name_2> =
<variable_name>.split("[ \"\'\t\n\b\f\r]", 0);
```

d. Add another repetition structure that relates to the length of the `<string_array_name_2>`.

Example:

```
for (int j = 0; j < <string_array_name_2>.length;
j++) {
}
```

e. Use the `if` function to compare the results of the information stored in `<string_array_name_1>` and `<string_array_name_2>`. Sum one unit to `<int_array_name>` if the contents are equal.

Example:

```
if
(<string_array_name_2>[j].compareTo(<string_array
_name_1>[i]. == 0) {
      <int_array_name>[i]++;
}
```

f. Use the `for` structure to print on the screen the number of occurrences of given content.

Example:

```
for (int j = 0; j < <string_array_name_1>.length;
j++) {
      System.out.println(<string_array_name_1>[j]
      + ":" + <int_array_name>);
}
```

The reader can find the complete code in the following template:

Result:

```
public static void <method_name>(String <variable_name>, String[] <string_array_name_1> {
   if (<string_array_name_1>.length > <number_of_words>) {
      System.err.println("Too many words!");
   } else {
      int[] <int_array_name> = new int[<string_array_name_1>.length];
      for (int i = 0; i < <string_array_name_1>.length; i++) {
        String[] <string_array_name_2> = <variable_name>.split("[ \"\'\t\n\b\f\r]", 0);
        for (int j = 0; j < <string_array_name_2>.length; j++) {
          if (<string_array_name_2>[j].compareTo(<string_array_name_1>[i]. == 0) {
            <int_array_name>[i]++;
          }
        }
      }
      for (int j = 0; j < <string_array_name_1>.length; j++) {
        System.out.println(<string_array_name_1>[j] + ":" + <int_array_name>);
      }
   }
}
```

Figure 3 – API Template.

3.1.1. Practical Examples

To illustrate the code, please, observe the following practical examples:

Practical Example 1:

```
<number_of_words> = 5

<method_name>("<Company Name> was founded on May 6, 2013. Today, <Company Name> has offices in some countries, such as Brazil, Argentina, Mexico, and Colombia.", {"<Company Name>", "May", "2013", "Brazil", "Argentina", "Mexico", "Colombia"});
```

Result:

```
Too many words!
```

Practical Example 2:

```
<number_of_words> = 5

<method_name>("<Company Name> was founded on May 6, 2013. Today, <Company Name> has offices in some countries, such as Brazil, Argentina, Mexico, and Colombia.", {"<Company Name>", "May", "2013"});
```

Result:

```
<Company Name>: 2
May: 1
2013: 1
```