Assignment: Conduct a System Check

Team members: Maryam Bunama, Eric Russon, Roman Kapitoulski

CPRG-217 | Date Due: Jun 18, 2023 | Instructor: Sanoop Sadique

Table of Contents

Table of Contents..1
Assignment Problem..2
Solution..2
Pseudocode..3
Each Member’s Role..4
Brief Summary..5

Assignment Problem

In this assignment, our boss has acquired a new client. This client would like us to do security audits. The audits must be done in person on all 50 of their Linux machines. The boss has allowed us to create and use 2 scripts that will allow us to take notes of each computer and display the needed information. The security audit should include the computer's name, users, groups, information about the computer, and the services being run on the computer, as well as its status.

Solution

>> Please see the attached code!

Our solution began with first planning out what we wanted the first script to lookup and record for us. Since we know the information we wanted to look for, we first began looking in the Linux system to see where we needed to find the information. The information we required are the name of the machine, information detailing the machine, the users of the machine, what groups they are part of, and the systems of the machine and their statuses. Once we located each piece of information, we then began writing the script to have python copy the information into lists and objects. The objects were created from the classes we defined, and each class corresponded to a piece of information we required. We then created methods for each class to convert the information contained in the object to dictionary format. With the dictionary containing all the information, we then copied it into the .json file.

For the second script, we needed it to be able to read the JSON file we previously made and display the information. To do this, we needed to first ensure that the JSON file we made was properly named and located. After confirming the file existence, we then had our script read the JSON file and output the information to the terminal. The output would then be formatted to ensure the data displayed is clear and simple to locate any needed information. We decided to recreate the tables seen in the JSON file to be the format displayed to the user.

When saving the information to the JSON file, we had to decide how to organize it. Our group decided to make tables. For each machine, we made 3 separate tables. The first table will show the machine name and the information about it. The second table will contain the users and their associated groups. The final table will contain the systems and their activity status. This makes it easy for anyone who obtains the JSON file to be able to read it, as well as use our second script.

Pseudocode

```
1. Check the computer info (Project_2_WriteData.py)
    a. Locate needed information
        i. Check for computer name/machine name
            1. if no machine name located:
                a. locate: host name
            2. write machine name to .json
        ii. Locate all users and associated groups (alphabetically)
            1. Check /etc/passwd for users and groups
                a. Record Users
                b. Lookup groups User is part of
                c. Record Groups
            2. write users and groups to .json
        iii. Retrieve processor info
            1. Read /proc/cpuinfo
                a. Retrieve vender_id
                b. Retrieve Model
                c. Retrieve Model name
                d. Retrieve Cache
            2. write cpu info to .json
        iv. Check status of currently running services on machine
            1. locate the current running services
            2. place services into lists
            3. copy status of server to list
            4. write services to .json
                a. optional: copy all services into a new temp text file.
                    i. read lines of temp text file
                    ii. copy information into lists
                    iii. write lists into JSON file
    b. Write information as a .json file
        i. Name file as Project_2.json
```

```
            ii.   Append Machine Name
            iii.  Append CPU Info
            iv.   Append Users
                      1.  Append Groups
            v.    Append Services
                      1.  Append Service Status
            vi.   Format .json file
            vii.  Save file to Temp directory
2.  Demonstrate .json file (Project_2_PrintData.py)
      a.  Check for .json file
            i.    If .json file doesn't exist/improperly named:
                      1.  Give an error.
                      2.  Quit program.
            ii.   Otherwise:
                      1.  Read file contents
                      2.  Record information
                      3.  Close file
      b.  Display information
            i.    format information into their proper categories
                      1.  Print Machine Name
                      2.  Print CPU Info
                      3.  Print Users
                            a.  Print associated groups
                      4.  Print Services
                            a.  Print Service Name
                            b.  Print Service Status
```

Each Member's Role

We agreed to divide each task in the programming process so that everyone can focus on a specific aspect based on their abilities.

Eric Russon
Eric was in charge of creating the body of the document, inserting the completed pseudocode, and ensuring that all the necessary information was communicated properly. Additionally, he aided in the process of creating the pseudocode and participated in the group discussions.

Maryam Bunama
Maryam was in charge of creating the second script and formatting the JSON file. She ensured that the coding used in her script would correlate with the coding of Roman. Her participation in the group chats aided everyone in the vision of the project and ensured everyone was completing their assigned parts. She also completed the final touch ups on the coding of both scripts and the final draft of the document before submitting.

Roman Kapitoulski
Roman was in charge of creating the first script. His contributions aided in searching up functions used to retrieve the information in machines, being able to store the information, and performing tests of the script on various machines.

Brief Summary

With everyone's effort and contribution, we successfully created two working scripts that were able to retrieve information off of the machines quickly and accurately, and be able to display the information clearly for users to read.