Week 8 Deliverables

Overview: This week, you have studied Web application vulnerabilities, password complexity, logs and analysis of logs, cryptographic algorithms, and installed a geolocation module allowing IP addresses to be aligned with a specific latitude and longitude. The Lab for this week demonstrates your knowledge of this additional knowledge applied using Python functionality.

Be sure to develop and test your Python code in the AWS Cloud9 IDE provided for the class.

You should continue to use the PEP Python Style guide mentioned in the book and found here:

https://www.python.org/dev/peps/pep-0008/

Some examples of Python Coding Style best practices include:

- Limit all lines to a maximum of 79 characters.
- Imports are always put at the top of the file, just after any module comments and before module globals and constants.
- Use 4 spaces for indentation.

Submission requirements for this project include 2 files. (Zipping them into one file is acceptable and encouraged):

- Python Application Tools Code
- PDF or Word file showing your Cryptographic puzzle solving skills along with the tests and log analysis documentation resulting from using your Python application tools

Python Applications for Lab8: (total 100 points):

This exercise **(50 points)** uses the AWS Cloud9 environment develop and fully test a set of tools and Web Forms to perform the following functionality:

a. Password Login form – This Python form allows a user to login to a simple web application with a username and password. A file can be used to store the username and password for validated users for this activity. No additional Web application functionality is needed after successful login other than a Greeting of your choice and the ability to update the password in a form.
b. Password update Form – This Python form allows a user to update a user's password after they have successfully logged in.
c. Authentication functions – These Python functions that will check the following NIST SP 800-63B criteria are met upon login or upon password update:
    - SHALL be at least 8 characters in length
    - SHOULD be no more than 64 characters in length
    - SHALL compare the prospective secrets against a list that contains values known to be commonly-used, expected, or compromised (Provided as CommonPasswords.txt)
    - If the chosen secret is found in the list, the application SHALL advise the subscriber that they need to select a different secret, SHALL provide the reason for rejection, and SHALL require the subscriber to choose a different value

- SHALL implement a time-based rate-limiting mechanism that effectively limits the number of failed authentication attempts that can be made on the subscriber's account. For this exercise throttling should start after 15 attempts.
- When the subscriber successfully authenticates, the verifier SHOULD disregard any previous failed attempts for that user from the same IP address

d. Logger – Create a log to log all failed login attempts. The Log should include date, time and IP address.

e. Log Analyzer – Create a Python log analyzer application that reads the log file created in part d to identify and geo-locate all IP addresses where more than 10 failed attempts in a period of less than 5 minutes. The geolocation should include the Lat/Long value provide from the IP Address location.

   A sample report might look like this:

   100.16.4.23 had 12 failed login attempts in a 5 minute period on Jul 7, 2019.

   100.16.4.23 has a Lat/Long of 41.2908816/-73.610759.

Hints:

1. Start early. This will take you longer than you think.
2. Leverage the File I/O, Flask and Data structures work previously performed in the class.
3. Use functions to enhance code reuse and modularity.
4. Use the AWS Cloud9 IDE.
5. Use Python Lists or other data structures to store the Common Passwords and then appropriate search functions to expedite comparisons.
6. You can use "request.environ['REMOTE_ADDR']" to obtain the client IP address. You will need to import the request package: "from flask import request".
7. You will need to load the ip2geotools Python module to perform the GeoLocation (sudo python3 -m pip install ip2geotools). You will need to import the IpCity Package (from ip2geotools.databases.noncommercial import DbIpCity). See the ip2geotools for additional method and objects available.
8. Be sure to send me questions, if you need assistance.

2. Using the Decrypting Secret Messages sites found in this week's readings, decrypt the following messages**. (30 points)**

a.
```
- .... .. ... / ... -.. . ...- / ...-- ----- ----- / -.-. .-.. .- ...
... / .... .- ... / ... --- -- . / ... - .-. .- -. --. . / .-. . --.-
..- . ... - ... .-.-.-
```
b.
```
U28gdGhpcyBpcyBiYXNlNjQuIE5vdyBJIGtub3cu
```
c.
```
--- Psuwb Ysm ----
W oa gc qzsjsf. Bc cbs qcizr dcggwpzm twuifs hvwg cih.
--- Sbr Ysm ---
```

Provide the decoded message along with the Cipher and any other parameters you used to solve each puzzle.

Hints:

1. Use the rumkin site
2. You will need to experiment some to narrow down the possible algorithms used. Some are more obvious than others.
3. You will know when you have selected the correct Cipher

3. Document your results of the application running from the AWS Cloud9 classroom environment. Provide your test results for each requirement in the Web application, associated functions and the log analyzer program. Describe the results of your NIST password complexity functions and how you tested each requirement. Include the Cipher tool results and write up in this document as well. (**20 points)**

Any submissions that do not represent work originating from the student will be submitted to the Dean's office and evaluated for possible academic integrity violations and sanctions.