Hacking Zipped Files with Python

# learning outcomes

LEsson learning oUTCOMES

• Students will be able to modify a Python program that interacts with zipped files.

• Students will be able to develop a loop in Python capable of brute-force guessing a password.

• Students will be able to recover hidden files from password-protected, compressed zipped files.

# Lesson Details

**Interconnection:** This lesson comes after an introduction to Loops in Python, and connects to ethical hacking concepts of password security and password cracking using tools like Hashcat and John the Ripper.

**Assessment:** Observe students accessing files and opening content; If grading is to occur, students can be asked to submit the text or a special code in the hidden file; Have one or more students per group stand and share their approach to the program.

**Extension Activities:** Password cracking in Hashcat or John the Ripper is a next ethical hacking activity, reading and writing files could be a next programming exercise in Python, or performing encryption/decryption in Python.

**Differentiated Learning Opportunities:** Teacher could produce a different zipped file with alphanumeric characters in the password to brute force using itertools for more advanced learners; for beginners, use instructors/assistants/counselors to circulate the room and assist where students get stuck, and allow peers to support one another in achieving the goal.

# lesson

**Lesson 1 Details:**

**Warm Up:** Inform students that an employee in a government agency (or a major corporation) has been fired for posting confidential information to several web sites and online forums. Before the employee was escorted from their desk by security, they hid all their documents in a password-protected .ZIP file on their desktop named ***topsecret.zip***, and they refuse to provide the password. You have been hired to crack the password and extract the contents of the zipped file to see what the employee was hiding.

In the wastebasket next to the employee’s desk, the security team found the following text scrawled on a torn sticky note: University0… but the rest of the note is missing. Security thinks it could be part of the password to the zipped file.

To help you practice cracking the file, the security team produced a similar zipped file, ***test.zip***, encrypted with a password starting with University0, followed by an additional digit. See if you can manually brute-force that file by trying the passwords University00, University01, and so on. (First to discover the password gets an extra GenCyber Buck! [optional])

You can find both zipped files, *test.zip* and *topsecret.zip*, as well as the sticky note as *file-13.jpeg*, online at <http://tinyurl.com/GCWA19>

**Lesson:** The security team believes the password for ***topsecret.zip*** is perhaps an additional **5 digits** after University0, so they know they’re going to have to use a computerized app or tool to crack the password. They have enough programming skill internally to crack a password with a single extra digit (like the one for *test.zip*) in Python, and they’ve created the Python 3.X file ***zipit.py*** below (and online at http://tinyurl.com/GCWA19) to help you get the basic idea:

import zipfile

myzip = zipfile.ZipFile("test.zip")

password = "University0" + str(7)

try:

myzip.extractall(pwd=bytes(password, encoding="utf-8"))

print("File extracted! The password was " + password)

except Exception as e:

print(e)

To use the Python program to crack the zip file, **both files must be located in the same directory** (e.g. put both *test.zip* and *zipit.py* on the Desktop before running *zipit.py* to crack the file). You’ll know the program was able to crack the zip file’s password when a new file, *test.txt*—the content from the zip file, appears next to the *test.zip* and *zipit.py* files.

Your challenge is to modify this code to brute-force the password by adding as many as five digits (or more) after the partial password University0 that we found on the partial sticky note in the fired employee’s trashcan.

After you’ve cracked the file yourself, you can help your peers or teammates with their programs, but remember: no hands-on-keyboard help, all typing and coding must be their own.

**Note to instructors/facilitators:** If students need help getting started, guide them to write a loop to crack ***test.zip*** using brute force from 0 to 9 using a variable inside the str() function, but let them help one another and find their own way through the remainder of the exercise. Circulate the room (with several facilitators), helping any students who get stuck or run into errors. Most common error (besides typos) will likely be having zipit.py and test.zip in different locations – have students move them into the same folder and try again.

Lesson developed by Dr. Bryson Payne, University of North Georgia – bryson.payne@ung.edu