МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра "Захист інформації"

МУЛЬТИПОТОКОВІСТЬ ТА МУЛЬТИПРОЦЕСОРНІСТЬ В PYTHON

МЕТОДИЧНІ ВКАЗІВКИ
до лабораторної роботи №1
з курсу «Програмування скриптовими мовами. Частина 2»
для студентів спеціальності
«Кібербезпека»

Затверджено
на засіданні кафедри
"Захист інформації"
протокол № 01 від 17.08.2021 р.

Львів – 2021

Мультипотоковість та мультипроцесорність в Python: Методичні вказівки до лабораторної роботи №1 з курсу «Програмування скриптовими мовами. Частина 2» для студентів спеціальності «Кібербезпека» / Укл. *Я. Р. Совин* – Львів: Національний університет "Львівська політехніка", 2021. – 5 с.

Укладач: Я. Р. Совин, канд. техн. наук, доцент

Відповідальний за випуск: В. Б. Дудикевич, д.т.н., професор

Рецензенти: А. Я. Горпенюк, канд. техн. наук, доцент
Ю. Я. Наконечний, канд. техн. наук, доцент

Мета роботи – ознайомитись з модулями стандартної бібліотеки Python для мультипотокового і мультипроцесорного програмування на прикладі brute-force атаки на пароль**.**

1. ЗАВДАННЯ

1.1. Домашня підготовка до роботи

1. Вивчити теоретичний матеріал з лекцій №2-3 “Паралельні обчислення. Мультипотоковість та мультипроцесорність в Python”.

1.2. Виконати в лабораторії

1. Вивести інформацію про кількість процесорних ядер *n_cpu* для вашого ПК.
2. Здійснити підбір пароля заданої довжини, що використовує заданий набір символів та хеш-функцію (див. табл. 1), за його хешем (табл. 2).

   2.1. Не використовуючи мультипроцесорність. Виміряти і вивести час підбору паролю та знайдений пароль (підбір деяких паролів може зайняти декілька годин у випадку використання одного ядра).

   2.2. З використанням *Process* або *Pool* (використовуючи число процесів рівне числу ядер вашого комп’ютера). Виміряти і вивести час підбору паролю та знайдений пароль. Якщо пароль знайдено одним з процесів, то він повинен передаватися в основну програму, а виконання процесів повинно завершуватися.

   Для обчислення хешу використовуйте стандартну бібліотеку *hashlib* та utf-8 кодування символів паролю:

```
import hashlib
test_password  = 'my_passwd'
test_hash_obj = hashlib.sha224(test_password.encode('utf-8'))
digest = test_hash_obj.hexdigest()
print(digest) # 4b18bf88fab81f30a3bcf3de9397ab4aff16545694dc733ab6630efd
```

   У якості наборів символів можуть виступати великі і малі літери англійського алфавіту, цифри і спеціальні символи (відповідно до варіанту у табл. 1):

```
uppChar = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
lowChar = "abcdefghijklmnopqrstuvwxyz"
numChar = "0123456789"
spcChar = "!@#$%^&*_-"
```

Табл. 1

Варіанти завдань

Варіант	Довжина, символів	Набір символів	Хеш-функція
1.	6	uppChar + numChar	sha224
2.	6	uppChar + spcChar	sha384
3.	6	uppChar + numChar + spcChar	sha3_224
4.	6	lowChar + numChar	sha512
5.	6	lowChar + spcChar	blake2s
6.	6	lowChar + numChar + spcChar	blake2b
7.	5	lowChar + uppChar	sha224
8.	5	lowChar + uppChar + numChar	sha384
9.	5	lowChar + uppChar + spcChar	sha3_512
10.	5	lowChar + uppChar + numChar + spcChar	sha3_384
11.	6	uppChar + numChar	sha3_256
12.	6	uppChar + spcChar	md5

13.	6	uppChar + numChar + spcChar	sha256
14.	6	lowChar + numChar	sha1
15.	6	lowChar + spcChar	sha224
16.	6	lowChar + numChar + spcChar	sha384
17.	5	lowChar + uppChar	sha3_224
18.	5	lowChar + uppChar + numChar	sha512
19.	5	lowChar + uppChar + spcChar	blake2s
20.	5	lowChar + uppChar + numChar + spcChar	blake2b
21.	6	uppChar + numChar	sha224
22.	6	uppChar + spcChar	sha384
23.	6	uppChar + numChar + spcChar	sha3_512
24.	6	lowChar + numChar	sha3_384
25.	6	lowChar + spcChar	sha3_256
26.	6	lowChar + numChar + spcChar	md5
27.	5	lowChar + uppChar	sha256
28.	5	lowChar + uppChar + numChar	sha1
29.	5	lowChar + uppChar + spcChar	sha224
30.	5	lowChar + uppChar + numChar + spcChar	sha384

Табл. 2

Варіанти завдань

Варіант	Хеш паролю
1.	807e6cafe2be56fb6be420d3e17e06a8fb9139053b6da2cdbc287f44
2.	1dff50155be465154c592e6f926cf44cc42005cf66e9dda93394561914ce5eae8bfceeed77f21586d3975547136b172e
3.	196b492918daab1a3af13bed0d60f8e72d238f08fb584209e513dcb6
4.	b8fa8a9d4d7802bb27694454d6e27577ff144bcc92d8c78c74a2536deb53bd0f2c9a281d89f2dba9fdbdca123fa9691714755bedfdc67997dba7dd51a6f5e4bc
5.	713417394e2b74aee7fa54d09375b0c78d288a90eba27a560cfc698c5b54886c
6.	6f0874bd7ba418106ac15555ea927aef34bc05c8ba92cd2e4c3065e7751ccc125fd88f19eec18f007e9f033c8dd2749edf573ac8aeec47d073f5832553fcea9c
7.	090ec244dc136b692f252ea2409b9c499794f2db7b37da089dcc9b8a
8.	4cdcbdf09ac94c80c006d7f73a034c92b5ed5d96cf84553670c73ad626682f677030efe22d3b2b0fecf2612a5dedc8ea
9.	6a087d0adc6d4503e2b12a06f9d843ddbd5eafc0549a97943f0139a8038131b29e86e12094fbe0879a17cfcb793bf8976a4b2fb43ede226e373f1513fbc24213
10.	177153b38d910937a0b0b757d2f502a167f79a8a50370027b8f91544e92c48b0521028f0accc694cbbbd7cba33e1180c
11.	e6c49956fbf1773468b757df603e4652d1df175d8d0f9ad5d89386d7a1d3f6ab
12.	cea9c4b6672eb0925438bf6b3ad7276f
13.	abd90779cc179a42b911cae0d2d9bc52fe7130b9c928ecb1d4f4d7158cd4ec93
14.	48057c0620bac2a0f4895f200eb24cc6362bc310
15.	c718d45ac0cf5ad433a69c02b45893375a4ad8c07f1656f4bcfc3041
16.	20f5fe0395c7abed4d7fb81e66bb81855b712a772d241b648510b4bd42d452da0a78292b57bdbb5661b5e6734195f26d
17.	f2112f9998c877cc1c6768c0017f9bef2dfc174f6caa858e10649c2d
18.	e5ddd824e5d0530c94ab2eafa4e31b35c572c368a63bec11be57ec128ad3272be13e8c82019812f0eee035d03b9d4495fde9c2d7ced75fd676b225bf6b2b900d
19.	f6cdd18a322df702c076e508a62e9973a70da5363b438e0cf2b7a2d333ec38f7
20.	03a1402a13ef499737844a1239b09f8a774cbcbdbad505f77a8d7dc31878f81edbd1615ebabae60826dd094f02af54ce86699096b58ee91addee5671ee8661dc
21.	edcbf6a2e20e3fe8d6579743242daeab2c42aea23be53f556900c1b2
22.	2b697b9f191f2299c2b7fb5caeb1cae4bb2fe2bab7387d01bf8e91840d3064a827b99c5470b20bc4cb2199f0180cd4e2
23.	1760a764876c6156a43929d69b20d8bf08076d21542c34f68963bda119861d95092a5e56b5b17c3eeef5f286f518130ea686efa352af444c08711cdce7e88c88
24.	6278e8c870934707f6c7178aa9e1b13dc0987b719ef6d2537cbadb73c708d8e4e4f56bdf819449626f1cea6aa39f6090
25.	1635905380b43d701d4b914337d33c784e090557a7d49accf68f7bb9501d391f
26.	ac803cce0e7f573f63bc7a17dd75cd3b
27.	435679defd017c4201bf06d0af1479912ec485dd249b3992ba178d29ed11f300
28.	abacd2a88fcdc84d6f0050ea618f5387d230350a
29.	8cdb044f44c86cbb501b2a496804a40cabb4b762efe9c1fc87760b51
30.	98295d27f05cda6e7fb0ac9a68a3f2dfefd7d55c15dfe79dd4c3714c6e5c68606458a538d86c4a82504da74c9007a516

2. ЗМІСТ ЗВІТУ

1. Мета роботи.
2. Лістинг програми.
3. Результати роботи програм (текст і скріншоти).
4. Висновок.

3. КОНТРОЛЬНІ ЗАПИТАННЯ

1. Для яких задач краще використовувати процеси і потоки ?
2. Яким чином можна здійснювати обмін даними між процесами ?
3. Які є засоби синхронізації ?
4. У чому відмінність потоків та процесів ?

4. СПИСОК ЛІТЕРАТУРИ

1. Password Cracking. https://materials.rangeforce.com/tutorial/2019/06/17/Password-Cracking/.
2. Password Cracking Countermeasures. https://materials.rangeforce.com/tutorial/2019/06/18/Password-Cracking-Counter/.
3. Password cracking. https://en.wikipedia.org/wiki/Password_cracking.
4. How to Crack a Password. https://www.guru99.com/how-to-crack-password-of-an-application.html.
5. Most popular password cracking techniques: learn how to protect your privacy. https://cybernews.com/best-password-managers/password-cracking-techniques/.

НАВЧАЛЬНЕ ВИДАННЯ

МУЛЬТИПОТОКОВІСТЬ ТА МУЛЬТИПРОЦЕСОРНІСТЬ В PYTHON

МЕТОДИЧНІ ВКАЗІВКИ
до лабораторної роботи №1
з курсу «Програмування скриптовими мовами. Частина 2»
для студентів спеціальності
«Кібербезпека»

Укладач: Я. Р. Совин, канд. техн. наук, доцент