PyCracker Version 1.0 Alpha

By: Kernel Coding Phantoms

a.k.a

Kevin Almeida

Chris Martin

Pamela Bleve
Contents
Installation – .................................................................................................................. 3
Introduction – .................................................................................................................. 3
Requirements – ................................................................................................................. 4
Main Page: ...................................................................................................................... 6
Attack tab: ....................................................................................................................... 7
Attack: ............................................................................................................................. 8
Wordlist: ......................................................................................................................... 11
Output Window: .............................................................................................................. 12
Testimonials: ................................................................................................................... 13
Installation –
Requires Aircrack-ng Suite
QT 5.4 for Linux
Python 2.7 for Linux
Kali Linux or similar Linux platform in a non-virtual environment
Compatible network card or adapter

To run the pyCracker software you first must open Terminal in Kali Linux. Once this is done you will change the directory using the “cd” command to look at the folder containing the program. This folder contains the GUI file, the main file which you execute, and the kcp file. All these files must be in the same directory for the program to run properly. Once you are in the directory you must run the command “python Cyber_Main_21.py”. This will start the program and you’re free to start scanning for routers around you.

***Disclaimer: For private use only. Not for commercial use or to make a profit***
***Any adaptations of this program must credit the original creators and to Mercy College***

Introduction –
What is PyCracker? PyCracker is a GUI based wireless network penetrator and auditing solution. PyCracker utilizes a simple user interface so someone with little knowledge can operate the software with almost no training. PyCracker supports all current wireless encryption types including WEP, WPA, and WPA2.

Wireless Types -

WEP encryption: Wired Equivalent Privacy is a security algorithm for IEEE 802.11 wireless networks. Introduced as part of the original 802.11 standard ratified in September 1999, its intention was to provide data confidentiality comparable to that of a traditional wired network. WEP, recognizable by the key of 10 or 26 hexadecimal digits, was at one time widely in use and was often the first security choice presented to users by router configuration tools. Although WEP is still used, other wireless encryption methods have been developed that are more secure.

WPA and WPA2 encryption: Wi-Fi protected access 1 and 2 are two security protocols and security certification programs developed by the Wi-Fi Alliance to secure wireless computer networks. The Alliance defined these in response to serious weaknesses researchers had found in the previous system, WEP (Wired Equivalent Privacy). WPA (sometimes referred to as the draft IEEE 802.11i standard) became available in 2003. The Wi-Fi Alliance intended it as an intermediate measure in anticipation of the availability of the more secure and complex WPA2. WPA2 became available in 2004 and is a common shorthand for the full IEEE 802.11i (or IEEE
802.11i-2004) standard. WPA2 supports between 8 and 63 ascii length passwords or a string of 64 hexadecimal digits.

WPS encryption: Wi-Fi protected setup is a network security standard that attempts to allow users to easily secure a wireless home network but could fall to brute-force attacks if one or more of the network’s access points do not guard against the attack. Created by the Wi-Fi Alliance and introduced in 2006, the goal of the protocol is to allow home users who know little of wireless security and may be intimidated by the available security options to set up Wi-Fi Protected Access, as well as making it easy to add new devices to an existing network without entering long passphrases.

Requirements –
PyCracker requires several different tools to utilize the program properly.

Kali Linux: https://www.kali.org/ - A version of Linux made for penetration testers and ethical hackers. Kali Linux comes with most of the tools required to make PyCracker work. For PyCracker to run properly, Kali must be installed or run as a dedicated operating system. Virtual Machines will not work. This is because the internal wireless card will not work properly when run from a virtual machine

Aircrack-ng suite: Aircrack is the backbone of PyCracker. We used the following programs from the suite to power our program:

- aircrack-ng
- aireplay-ng
- airmon-ng
- airodump-ng

Wifite: https://code.google.com/p/wifite/ - Wifite is a command line automated wireless auditor that is designed to combine the aircrack-ng suite into one easy to use tool. We have heavily modified this program to include a GUI to help the user crack wireless networks without any additional knowledge on the process.

QT Designer and PyQT: https://www.qt.io/,
http://www.riverbankcomputing.co.uk/software/pyqt/intro – QT Designer is the software tool used to create PyCracker’s Graphical User Interface. PyQt converts this user interface into the python language.

Reaver: https://code.google.com/p/reaver-wps/ - Reaver is a wps brute force password cracking tool. If the router being audited has wps enabled, Reaver can be utilized to crack the wps pin.

Crunch: http://sourceforge.net/projects/crunch-wordlist/ - Crunch is an easy to use tool used to generate wordlists for use in wpa and wpa2 password attacks.
Problems we faced: During the creation of PyCracker, we ran into many problems along the way.

- It took us a long time to find which tools we should implement in our program. Many different tools were tested before the final tools were selected and implemented into the program.
- When testing which tools and methods we should use, the router that was provided did not work properly.
- When we obtained a new router, the school network was not working properly (when PyCracker is used at home, it has no problems cracking wireless passwords. When used at school, sometimes the process does not work right).
- The wifite python code is over 3000 lines long and it took a very long time to find which parts we needed to modify and which needed to be kept unchanged.
- When the GUI was combined with the program that runs in the background, The UI would lock up while PyCracker was searching for wireless networks. This was because of two programs running off of the same thread. Starting and stopping threads in python had to be researched and implemented.
- Inserting a graphic into QT designer proved to be harder than it would seem. We also ran into scaling issues when the UI is expanded and contracted.
- Getting the background code to update the main UI was a problem and extensive research was done to combat this problem.
The main page of PyCracker hosts only one clickable interface button: Start. The start button starts the scanning of all wireless networks in the area. Once the progress bar fills up, all the networks found are then listed on the next page.
Once all the networks are found, they are displayed in the window. This window clearly shows all the information about the networks found including the ESSID or name of the network, the channel number, what type of encryption the network has, the power of the signal (the closer you are the stronger the signal will be), and if WPS is enabled.

When the user decides which network they want to attack, they simply select the proper number using the drop down window and hit the Check button.
When the correct network is selected, the user interface shows what encryption type the network has. When a WEP network is selected, IV packets are automatically generated and the User Interface will automatically switch over to the Output tab which will show the process and the result.

When a WPA or WPA2 network is selected, the user is faced with a few choices of attack:

- If WPS is enabled on the network, the user can select a WPS pin attack using Reaver. This will attempt to brute force the WPS pin to obtain the wireless password.
- If WPS is not enabled or the user does not wish to attack using this method, then PyCracker will try to capture a handshake with the router. When this capture file is created, the user then has two additional options in which the password can be cracked:
  - Wordlist: The user can supply a WPA/WPA2 wordlist to assist in the cracking process. These wordlists can easily be obtained at various websites online and once extracted can be many gigabytes in size.
  - Brute Force method: If the user does not wish to supply a wordlist, then PyCracker will try to brute force the password using a combination of crunch and aircrack-ng.
Brute force window:

When brute force is selected, the user is presented with a few options.

- **Password length**: PyCracker supports brute force of passwords between 8 (the minimum length of a WPA password) and 14 characters long.
- **Filters**: The user can select filters based on what they think the password could be. One two or all of the filters can be selected.
  - Lower Case: selects all lower case letters to be used
  - Upper Case: selects all Upper case letters to be used
  - Numeric: selects all numbers 0-9 to be used.

When the brute force choices are selected, aircrack will open and the cracking will begin.
Aircrack-ng 1.2 rc1

[00:00:24] 24316 keys tested (1116.19 k/s)

Current passphrase: 111117k0

Master Key: 54 B6 EC 59 3A 3E 5E A4 67 B6 35 AC C4 35 52 03
B1 8B 64 E8 AC 7D DC BC 73 E1 CC 1E D5 B6 75 6F

Transient Key: B7 9B BC DE 3A A7 1A D4 09 4F 15 E3 83 5A E0 ED
85 69 BA 34 6F 1A 78 89 12 C9 46 61 A1 59 D6 C7
D3 97 66 24 D1 27 57 93 EC 86 30 CB ED 81 33 21
EAPOL HMAC: F5 69 25 29 B4 19 DF D9 B1 A9 C5 54 49 84 04 D7
Wordlist:

A wordlist which is essentially a dictionary of pre computed list of words which can be used to speed up the cracking process. Wordlists can be created and customized using crunch or premade wordlists can be found online on various websites.
This is where all the final outputs are displayed. The final result will also be displayed here.
When the password is eventually cracked, the user will then be able to export the password into a text file.

Testimonials:
Kevin Almeida

Throughout my experience during the Cyber Security competition I learned a lot about Linux, Wireless cracking tools, Python and more. Before this competition I had very minimal experience with Linux, especially developing a program for Linux. It took many hours of research and struggle to adapt my programming skills to Linux. Once I got over the initial headache, things started to go smoothing and our program’s development accelerated. During this competition I learned how to implement threading and other advanced principles to my programs. I highly recommend that anyone in the Cybersecurity major join the next contest. You get to work in a team environment which will emulate a real world situation. Regardless of skill you will definitely learn a lot and in result become more knowledgeable on the subject. I am very proud of the accomplishment my team and I have made and will continue working together to further develop our program.

Chris Martin

The Cybersecurity competition is a great opportunity for students of both undergraduate and graduate programs to utilize what they have learned during their studies at Mercy College. Working with other students on a project that would have been near impossible to finish by myself has been a great learning experience. At the beginning of the project, none of our group members had ever played with any wireless cracking tools before. This project has allowed a hands on experience which will be very beneficial once we graduate and move on in a real work environment. We not only learned how to utilize these Wi-Fi cracking tools, but I know I learned a ton about the Python programming language by working with my group and asking them questions. We also learned a lot about how networks are compromised, which may grant us knowledge about defending wireless networks in the future. Even though our project took many weeks to complete, the overall learning experience was very beneficial and worth the time and effort that our group put into this project. I highly recommend that every student should take part in future Cybersecurity competitions.

Pamela Bleve

The idea of attending a cybersecurity competition has been for me an opportunity for academic and professional growth. This experience gave me the chance to put theoretical studies into practice, it has been very intense and this has helped us to create a great group. All of us were united in a common goal, motivating each other and spending many days together. We grew up together and each one took the best from the other, each one with a different
academic background, and this was great. Each member of the group has been indispensable and has given 100% contribution.

I can summarize my experience in some points:
- reasoning together and resolving any difficulties of the project,
- learning from scratch,
- facing intense days with the passion and the desire to smile...always.

I would strongly recommend to attend this competition to everyone...
CyberSecurity Competition = Challenge yourself, challenge your knowledge and challenge your ability to learn!