Ethical Hacking

Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 41
Password Cracking

In this lecturer I will discuss about password authentication and password cracking. In this lecture we are going to explore different authentication mechanism. An authentication mechanism are, method is a way for you to prove that you are allowed to access something. Password have been the default method of authentication for as long as most of us have needed to prove the computer that we are allowed to access it. However, password are not the only authentication mechanism.

There are some authentication methods are there; number one something you know. Example of these are, your good old password, bank card pin or a safe word, when the alarm company calls your home. These are all example of using something to know to authenticate yourself, something you have. Examples are swipe card to access a secure area, a code send to your cell phone as part of a login process, to prove you have your cell phone or a secure id token that provides a constantly changing code, you need to enter to login access.

All are something you have that can be used to authenticate yourself. Something you are. This is where biometric security comes in. To access our data centre, we have to put our index finger on a fingerprint scanner after swiping a card. Unless you steal someone's index finger, you will not be able to access our data centre even if you have stolen a valid swipe card.

Other biometric system include retinal scan, the blood vessels at the back of the eye and iris scan, the coloured part of the eye. Other attribute used for authentication, a few other attributes that you occasionally see used for authentication, somewhere you are, something you can do, something you exalt, something you know.

Our focus in this session is password. Most of us see them as an inconvenience something you have to tolerate to be able to use a service you need access to. In this session we are going to explain how computer system have evolved in the way they

process your password. How modern online applications do authentication and why it is important to choose a strong password. Once you finish this session, you should have a knowledge of hashing algorithm, how password cracking works and what strong password really means.

(Refer Slide Time: 03:42)

There are different types of password are there. In the early days of computer and mainframes password were stored in a database as plain text. When you wanted to sign in, a gatekeeper application would ask you for your password. It would take whatever you type in and check if it was equal to whatever it had store in the database. And if true you were granted access.

As the internet evolved and grave malicious hackers started gaining unauthorized access to the system. Once they were in, they would immediately download the plaintext password database and have instant access to all user password. Developers and system administrator needed to come up with the solution to this problem and the solution they came up with was password hashing.

So, now we will discuss about password hashing. Think of a hashing algorithm as a machine. In one end you input any text or binary data, out, the other end you get a number that is a certain length. Let us say 32 digits long. The data you feed, it can be any size from a few bytes to many terabytes or larger. No matter what data you feed in, you get a 32 digit number that uniquely represent the data.

What is amazing about hashing algorithm mechanism is that if you feed something identical in, you get the same 32 digit number. If you feed in war and peace, you get a number. If you copy the book and feed it exactly the same text you get, the same number. If you change a single character in the novel, you will get a completely different number.

Hashing algorithms differ in the way they work and the most notable difference is the length of the number each one splits out; MD5 is there which is extremely popular, split about 128 binary digits, SHA 2 split are there, 256 bit. When system administrator and developer first encounter the security problem with password data base that stored as plain text, they turn to hashing algorithms for help. What they came up with this instead of storing your password in a database, they would just store a hash of your password. That is the number that a hashing algorithm generates when it operates on your password.

When a user changes their password or when a user account is created, the new password is typed in for the first time the computer security application test that password and run it through a hashing algorithm and store the resulting number in a database. The next time you try to sign in and enter your password the security system runs the password you enter to the same hashing algorithm and check if the resulting hash matches then you are allowed to access the account.

No longer a password stored in clear text in a database. If a hacker stills the user account database, they do not automatically have all passwords. All they have is a list of hashes. The storing hashes of passwords instead of password themselves was a major breakthrough in information security. The story unfortunately does not end here. Now that hashes are commonly used to authenticate user instead of plain text password, a hacker does not immediately have a list of all passwords when they steal the user accounts data base.

However, there is a way for a hacker to steal hashes and turn them that into passwords. The method is relatively simple. When a hackers steal a database of hashes passwords to reverse engineer the hashes convert them back to password. The hacker generates hashes from a dictionary of words. He think might be the password that were used. If any of those hashes matched with he has in the database, he has managed to reverse engineer a

hash and now knows what the original password is. For example, let us say you have stolen a password database and you have the hash of the password that mug uses.

You want to know the actual password for the mug account. So, you take the word banana and run it through the same hashing algorithm that the password database used. You end up with the number and if the number matches the hashes in the password database for user mug to know his password. If it does not match, then I try pear and apple and apple pear for 3 5 and progressively more words and more complex word combinations.

So, to crack a password you need to take a very large dictionary of passwords and hashes. Each of them, then compare those hashes to what is in the password database you store and when you get a match you know the original password. The problem is that generating hashes of words takes time. Each word might take a few millisecond to hash. So, we need a very fast computer to do this. Or alternatively you can take a very large dictionary of well known passwords, generate hashes from all the word and store the word and their hashes.

Then every time you steal a password database, you can just re use that list of word and their hashes. You do not need to recreate the hashes every time. All you need to do is match your list of hashes with hashes in the password database and where you get a match, you have crack the password. Alternatively, you can also create your own password by using a programming or a predefined tool. What you have just described is called a rainbow table.

Rainbow table are a method commonly used by hackers to crack password databases that use ordinary hashing without any additional security. Rainbow table attack on hashes password database are very effective because they are fast. To help protect against this, a kinds of attacks developer and system administrator came up with the technique called salting password.

So, now we will discuss about how salt works. A rainbow table attack relies on a hacker being able to take a dictionary and precomputed hashes of the words in that dictionary and compare those hashes to the hashes in a password database. To defeat rainbow tables the information security community invented salted hashes. The concept is relatively

simple. When you create a new password instead of just running the password on its own through a hashing algorithm you do the following.

Generate a random little piece of text, put the text at the beginning of the password, then run the combination of the little piece of text and the password through a hashing algorithm. Then you store the little piece of text as plain text and the resulting hash. That little piece of text is called a salt.

When someone wants to sign in, they type their password. The security application takes the stored piece of text or salt, puts it at the front of the password that was entered and runs it through the same hashing algorithm to get a hash. It compares the resulting hash with the hash stored in the database and if they match you are granted access.

It important to note that the salt or little piece of text, is stored as plain text with the hash. It is also important to note that the salt is random for each password. In other words every password has its own special little piece of text. This is a relatively the simple concept, but it makes cracking hashed password significantly more difficult.

So, now the question is that why salt makes cracking more difficult? Recall the rainbow table or a dictionary of word and the hashes of those words. In the above example we use salt the little piece of text combined with our password to generate hashes. If a hacker wants to crack passwords he cannot use his rainbow table, because the rainbow table is just a hash of individual words.

He needs to combine those words with the stored salt to get the actual hash that is stored in the database. That makes cracking password much harder because it means a hackers rainbow table is useless and it forces into recompute hashes for every word in his dictionary. Here is an example of a password being created for someone called a good guy. The system administrator creates a new account on the system for user called good guy with the password apple.

The system automatically generate a short piece of text $y\ r\ t\ z\ d$. The system takes the short text and combines it with apple to create the text $y\ r\ t\ z\ d\ a\ p\ p\ l\ e$. It then runs $y\ r\ t\ z\ d\ a\ p\ p\ l\ e$ through a hashing algorithm and end up with a 128 bit number. The system store that number as the hash password for that particular account. So, when that person arrived at work and type sign in, it types apple as his password.

The system retrieves the record for that particular account that record is a hash and the text $y\ r\ t\ z\ d$ which is the salt. A system combines the, what apple that the person just typed in with the salt to make the text $y\ r\ t\ z\ d\ a\ p\ p\ l\ e$ and runs a hashing algorithm on that. The system check to see if the hash it retreat matches that has it just generated. It does match and it allows the person to access the system. Here are the some states a hacker text to crack the salted password.

A hacker arrives and manage to hack into the system and he steal the database of password hashes and salts. The hacker tries to use precomputed hashes of word in his English dictionary. One of the hashes is of the word apple, but that does not work because the hacker needs to combine the salt which is $y\ r\ t\ z\ d$ with the word apple before he hashes it. The hacker realize his precomputed rainbow table is useless. He needs to combine the salt for that particular password with the every word in its dictionary and then see which has his matches.

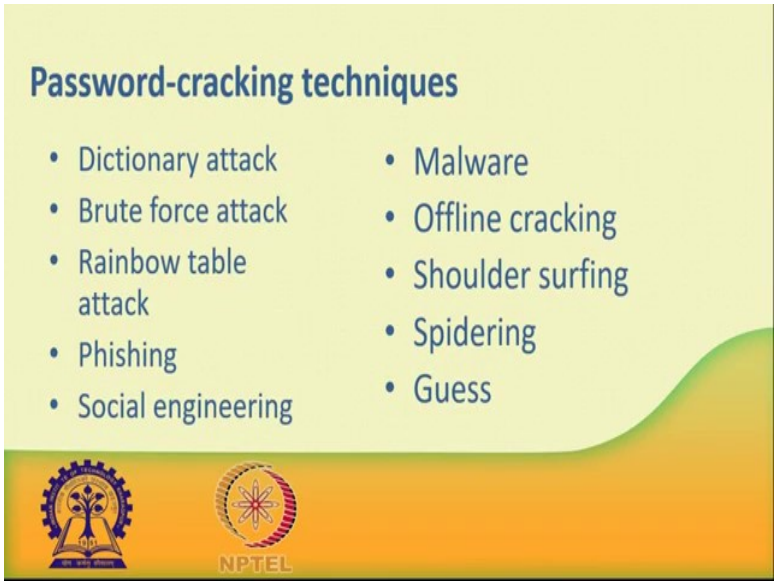
That means, he needs to recalculate hashes for his entire dictionary which is going to take significantly longer. As you can see from the above example, it is possible to crack passwords that use salts, it just takes much longer and requires more processing time. Hash password that use salts are what most modern authentication system use it, because it forces a hacker to hash every password that they want to guess.

You now have a working knowledge of how modern password authentication work on system like wordpress, Linux, windows and many other systems. You also understand why salt are useful because they prevent a hacker from very quickly cracking password hash by hashes, by using rainbow tables. Now that you understand the benefit of salted hashes it may seems obvious to you that everyone should use them when building authentication system. Unfortunately, they do not.

There are many example of custom build web application out there that did not use salts. They just use plain old hashes and when they are hacked, it is relative easy to reverse engineer the password hash using rainbow table. Now why strong passwords are important? If one of the services we use is compromised and hashes, hash password are stolen, even a teenager in his bedroom with the gaming PC under only 2000 dollar can try to turn your hash password into a plaintext password at a rate of 3.2 million cases per second and possibly much faster.

When you consider that your money linked in Google and many other well known brands happened successfully hack over the past few years it is quite likely that a service we used will have its hash password stolen sometime in the near future. This means that it is important to use passwords that are very difficult to crack. Any password with list at 12-character is weak and also use uppercase, lowercase, special character and numbers.

(Refer Slide Time: 20:58)



Now, we will discuss about some important password cracking technique used by hackers. Number 1: dictionary attack. The dictionary attack uses a simple file containing words that can be found in a dictionary. Hence it is rather straightforward name. In other words this attack uses exactly the kind of words that many people use as their password. Cleverly grouping words together such as super administrator guide, administrator, may be admin at NPTEL 2019 will not prevent your password from being cracked this way. Well not for more than a few extra second.

Number 2: brute force attack. Similar to the dictionary attack, the brute force attack comes with an added bonus for the hacker. Instead of simply using words a brute force attack lets them detect non dictionary word by working through all possible alphanumeric combinations from a a a 1 to z z z 10. It is not weak, provided your password is over a handful of characters long. But it will uncover your password eventually.

Brute force attack can be shorten by showing additional computing horsepower in terms of both processing power including harnessing the power of your video card that is GPU and machine number such as using the distributed computing model like online bit coins, minors.

Number 3: rainbow table attack, rain bow table are not as colourful as their name may imply, but for a hacker your password would will be at the end of it. In the most straight forward way possible you can boil a rainbow table down into a list of pre-computed hashes. The numerical value used when an encrypting a password this table contains hashes of all possible password combination for any given hashing algorithm.

Rainbow table are attractive as it reduce the time needed to crack a password hashes to simply just looking something up in a list. However, rainbow tables are used widely used. They required serious computing power to run and the table becomes useless if the hashes it is trying to find has been salted, but the addition of random characters to its password ahead of hashing the algorithm.

There is stock of salted rainbow table exciting, but this would be so large, has to be difficult to use, practice. They would likely only work with the predefined random character set and password stream below 12 characters. Phishing, there is an easy way to hack; ask the user for his or her password.

A phishing email leads the unsuspecting reader to a fake login page associated with whatever service it is the hacker wants to access, requesting the user to put right some terrible problem with a security, that page then skins their password and the hacker can go to use it for their own purpose. Why bother going to the a trouble of cracking the password when the user will happily give it to you anyway.

Then number 5 social engineering, social engineering takes the whole ask the user concept outside of the inbox that phishing tends to stick with and into the real world. A favourite of the social engineering is to call an office posing as an IT security tech guy and simply ask for the network access password. You had been amazed at how often this works. Some even have the necessary things to do a suit and name batch before walking into a business to ask the receptionist the same question face to face.

Number 6 malware, a key logger or screen scraper can be installed by malware which record everything you type or take screenshot during a login process and then forward a copy of this file to hacker central. Some malware will look for the existence of a web browser, client password file and copy this which useless properly encrypted will contain easily accessible saved password from the users browsing history.

Number 7 offline cracking, it is easy to imagine that passwords are safe when the system they protect lockout users after three or four wrong guesses. Blocking automatic guessing passwords applications well what would be true if it were not for the fact that most password hacking takes place offline using a set of hashes in a password file that has been obtained from a compromised system.

Often the target in question has been compromised via a hack on a third party which then provide access to the system servers and those all important user password hashes file. The password cracker can then take as long as they need to try and crack the code without altering the target system or individual user.

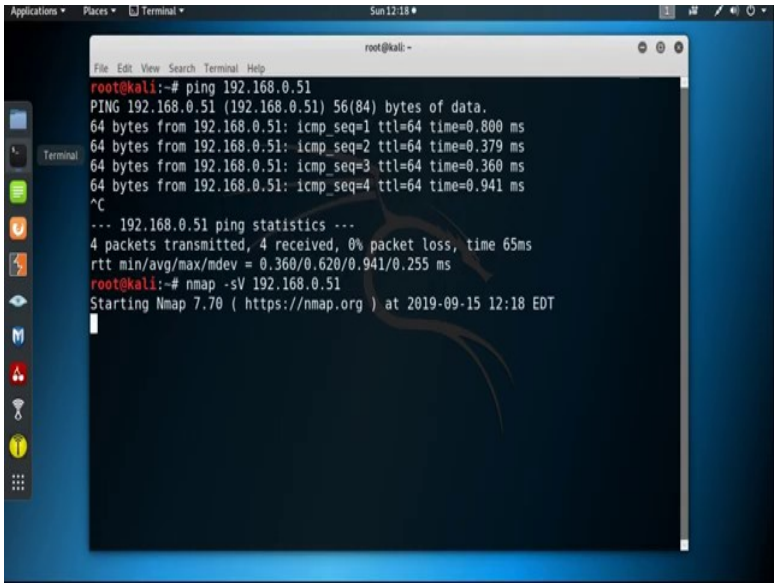
Shoulder surfing, the most confident of hackers will take the case of a parcel courier or a service technician or anything else that gets them access to an office building. Once they are in the service personnel uniform provides a kind of free pass to wander around in the hidden areas and make note of passwords being entered by genuine numbers of staff. It also provides an excellent opportunity to I ball all these post it not start to the front of LCD screens with logins trouble upon them.

Number 9 spidering, savvy hackers have realized that many corporate passwords are made up of words that are connected to the business itself. Studying corporate literature, website, sales material and even the website of competitors and listed customers can provide the ammunition to build a custom word list to use in a brute force attack. Really savvy hackers have automated the process and led a spidering application similar to those employed by leading search engines to identify keywords, collect and create the list for them.

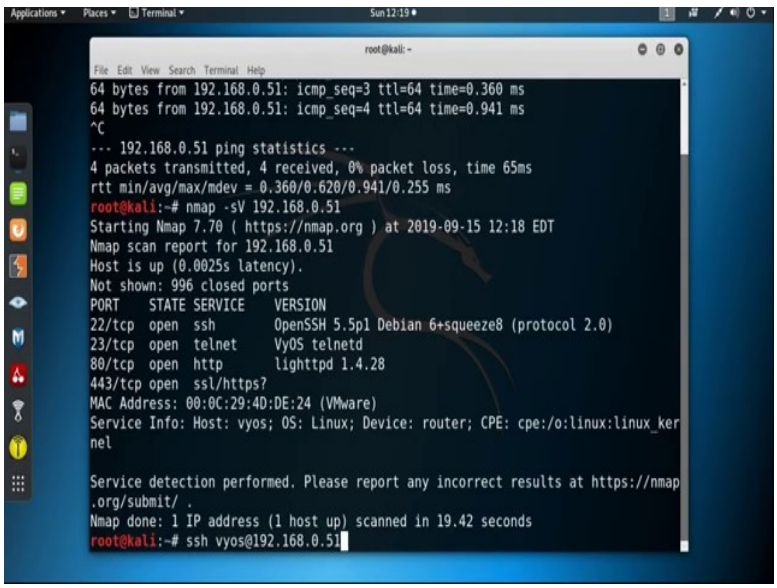
Number 10 guesses, the password crackers best friend of course, is the predictability of the user unless a truly random password has been created using software dedicated to the task a user generated random password is unlikely to be anything of the sort. Now I will show you how to perform dictionary attack. Now, suppose our target is 192.168.0.51.

Now let us use the tool *nmap* to find out which service is running on that particular system. Let us wait for the result ok.

(Refer Slide Time: 30:21)

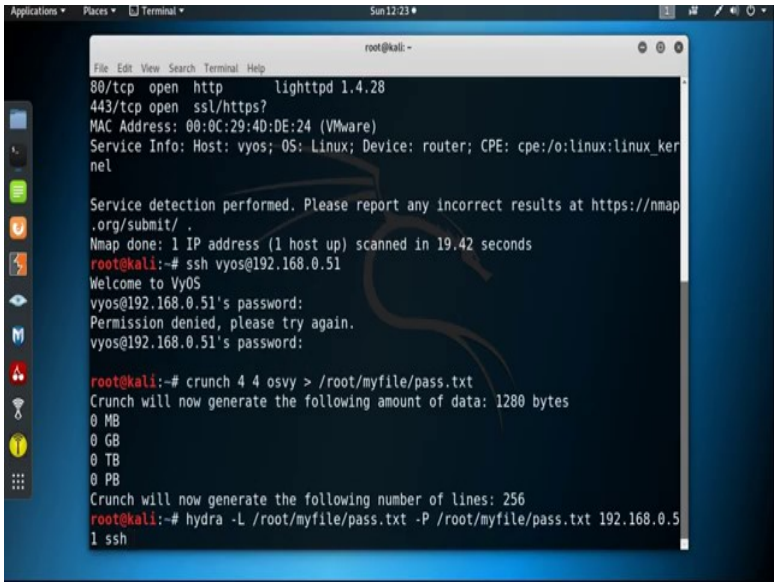


(Refer Slide Time: 31:03)



There is the result, port 22 *tcp* is open and *ssh* service is running; port 23 *tcp* port is open; *telnet* service is running. Port 80 is also open; *http* service is running; port 443 *tcp* port is also open and *https* services is running and hostname possibly *vyos* and operating system is Linux and device is router, ok. Let us try to connect with SSL service using the hostname vyos. Its asking for password.

(Refer Slide Time: 32:17)



So, I am putting password as admin, permission denied please try again, ok. So, actually I do not know the password. So, to break this password we can perform a dictionary attack. So, to perform a dictionary attack first we need a dictionary. So, we can use any pre stored dictionary; otherwise you can also use the tool $crunch$ to create your own dictionary.

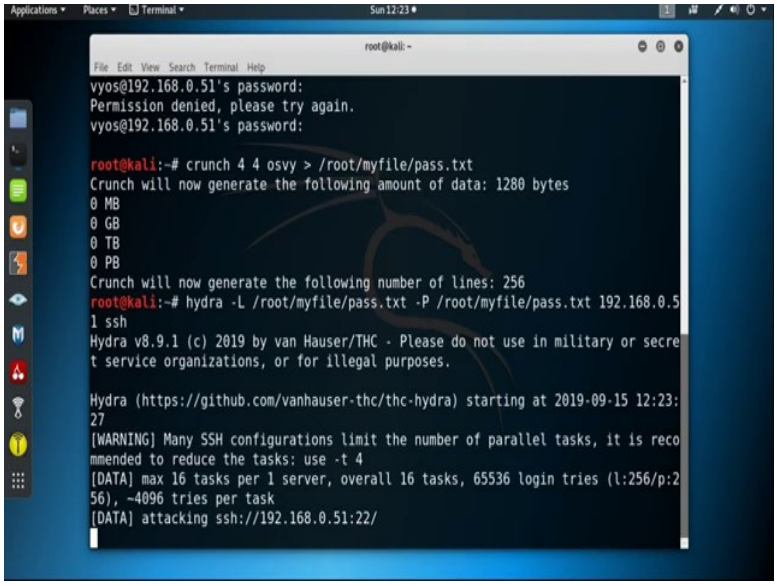
Here I am using the tool $crunch$ to create my own dictionary. $Crunch$ a tool name then minimum length of the password. So, suppose I am considered minimum length of password is 4 and maximum length of password I am considering maximum length as 4 and the characters from where it basically create the password. So, I am considering osvy and then store this in the folder root my file and file name is $pass.txt$. So, until now generate the password which total 256 password are there.

It is basically store all possible combination created by osvy. Now our dictionary is ready. Now I can use this dictionary to perform a dictionary attack over ssh service. So, to perform the dictionary attack now I am using the tool $hydra$. $Hydra$, then you can use as capital L to provide the username. So, I am using the same dictionary for username and password.

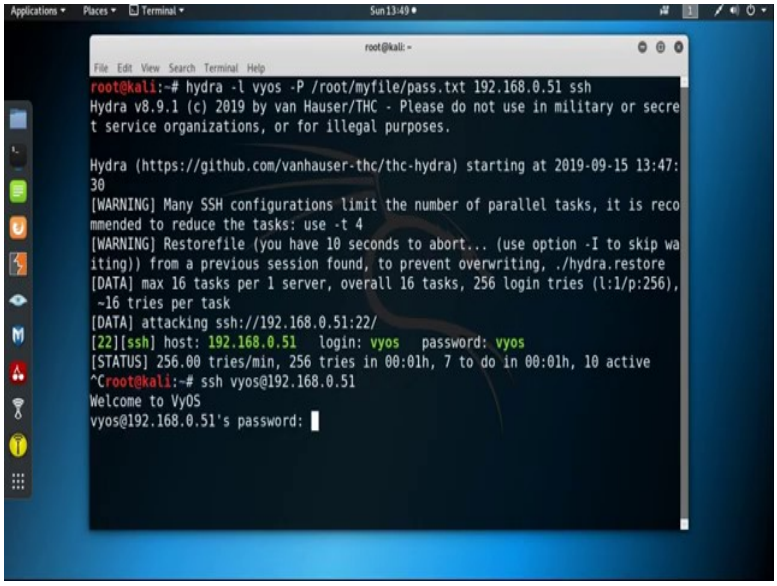
So, it is under $/root/myfile/pass.txt$ then $-P$ to provide the dictionary. It is also under $/root/myfile/pass.txt$ and then the IP address 192.168.0.5 and the service is

ssh, attacking ssh. So, it basically try with a all possible user ID and password which is stored inside the dictionary $pass.txt$.

(Refer Slide Time: 35:50)



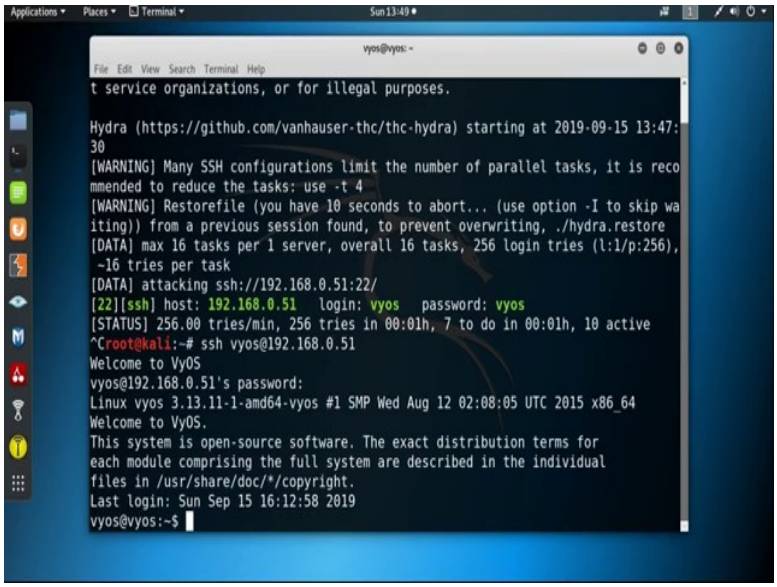
(Refer Slide Time: 36:14)



Now, suppose I know the user ID. So, in that case only use small l to specify the user ID. So, $hydra - l\ vyos$, this is the user-name and $-P$ and provide the dictionary, IP address with the service name, ok. We got the user id login name is vyos and password is also vyos. So, you got the password vyos. Now use this password to login into that particular

system using ssh service, ssh then username $vyos$@192.168.0.51 and welcome to vyos and it asking for the password; password is also vyos.

(Refer Slide Time: 38:20)



Now, now successfully able to penetrate inside the system 192.168.0.51. In next session I will show you how to crack the password using phishing attack. So, $hydra$ is one of the tool which we can use to break the password by using the dictionary attack. Some other tool is also there like $ncrack$ $medusa$. So, you can use any of the tool to attack in a service which is running in the victim machine.

Thank you.