COMPUTER SECURITY REPORT

Junsong Yang

School of Computer Science
University of Nottingham

1. PASSWORDS

In this section, the designed password and authentication policy will be proposed and justified. First, the password policy will be explained in detail with additional authentication measures. Then, mechanisms of storing passwords will be entailed.

1.1. Password Policy

As Gollmann [1]suggested, the overall security may be diminished if one security mechanism is overstated. Users tend to bypass the mechanism if it is too inappropriate for them to properly work with, hence the overall security of the system may be weakened. By considering that, the following password policies are proposed.

1.1.1. Password Length

This policy enforces the minimum number of character required to use as a valid password. Generally, the short the password, the more likely and easily to be cracked by bruteforcing. Hence, by setting the minimum password length to ten, the difficulty for brute-forcing password cracking would be noticeably increased.

1.1.2. Password Format

This policy intended to accumulating the strength of a valid password by requiring what kind of character must be included in a password. By requiring the password to contain at least one lower and upper letter, one number, and one special character, combining with password length policy, the possibility of successful brute-force cracking would be significantly decreasing.

1.1.3. Password Ageing

This policy requires users to change their password periodically. The likelihood of password breaching would increase as time goes by, hence this is an appropriate approach to eliminate the risk of potential breaching.

1.1.4. Password Use

To further diminish the risk of potential password breaching over time, additional mechanism need to be employed to block users from using the same password twice. This policy is essential to assist the password ageing policy to fulfil its purpose.

1.1.5. Password Choice

The Dictionary attack is another approach frequently used for password cracking. The purpose of this policy is to prevent this attack. This problem can be addressed by preventing the user from using the password in public known dictionary.

1.1.6. Login Attempts

This policy is designed to reduce the risk of brute-force attack. By limiting the maximum number of failed login attempts the success rate of brute-force attack can be reduced significantly.

1.2. Additional Authentication Measures

Mechanisms must be implemented to address the repeated authentication problem. Between the time of check to time of use, user identity exploitation may occur as the authentication system does not keep track of what happened in between. Therefore, before some important actions like change password can be successfully performed, the user's identity needs to be checked again to ensure the action is legitimate.

1.3. Storing Passwords

As suggested by Gollmann, a password may be cached by browser [1]. Which suggests that storing raw password directly in the database is a bad practice. To maximise security, passwords should be hashed using one-way hash functions with salting and stretching approaches [2]. Since hashing cannot be reversed, the original password will remain hidden.

2. FIREWALLS

Firewalls are software or hardware located between networks and filter potentially malicious packets from in and out traffic

[3]. Figure 1 illustrates a network firewall which often stands between the local system and the internet compared to the host-based firewalls which located on individual machines. According to Gollmann, firewalls can also prevent unauthorised accesses of the internal-only services which block unnecessary or potentially dangerous access of external services from inside of the network [1].

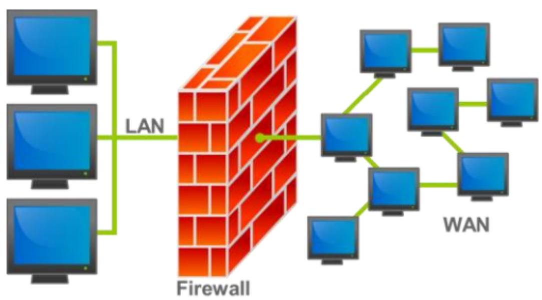


Fig. 1. Network Firewall

There several reasons that administrators may block port from normal traffic. As mentioned above, some services are only intended for internal access which suggests the necessity of blocking external access that may lead to security breaches. Another reason is that access external network from inside is also a potential breach in terms of the internal network as those ports are entrances to the network.

As for the internal network, some internal traffic is unnecessary even potentially harmful. A peer-to-peer communication protocol BitTorrent, in this case, is most likely been blocked in some internal network.

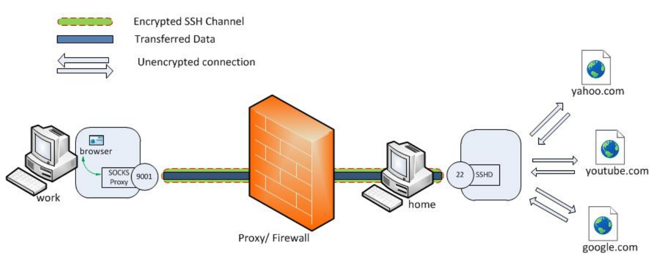


Fig. 2. (a)SSH Tunnelling [4]

Although some normal traffic is blocked or certain access rules are enforced by firewalls, there are ways to circumventing those rules and SSH tunnelling is one of them. SSH tunnelling is a way to establish an encrypted communication channel between two computers. In terms of bypassing the firewall rules, SSH tunnelling serves as a proxy. Figure 2 shows how SSH tunnelling works in general. Since the target server cannot be accessed directly, the request is first sent to the proxy server in the middle. Then the proxy server forwards the request to the target and returns the response back to the client. SSH tunnelling communicates with the proxy server by establishing an encrypted channel with the proxy server, and forward certain port on the local server to another port on the remote server.

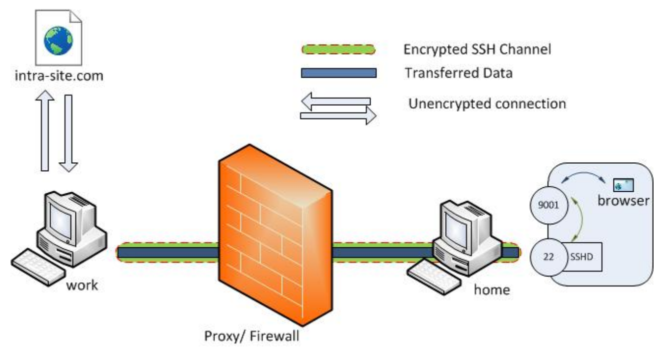


Fig. 3. (b) SSH Tunnelling [4]

Some usages of SSH tunnelling can be legitimate. As figure 3 demonstrates, the intra-site.com is a service only available from the internal network. By establishing the SSH tunnel with a machine inside the internal network, the internal services can be accessed from external. In this case, working remotely can be achieved. In fact, this approach is employed by most enterprises.

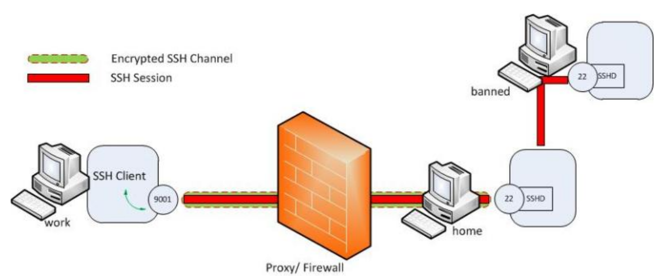


Fig. 4. (c) SSH Tunnelling [4]

Some usages of SSH tunnelling are disgraceful. In contrast to the example mentioned earlier, accessing blocked websites or services from the internal network, as figure 4demonstrates, exposes the internal network to the whole Internet. In this case, the internal is exposed under risks that the firewall built to eliminate which directly invalidated the firewall. Hence, this way of using ssh tunnelling is considered disreputable.

3. SERVER SECURITY

3.1. Exploitation Process

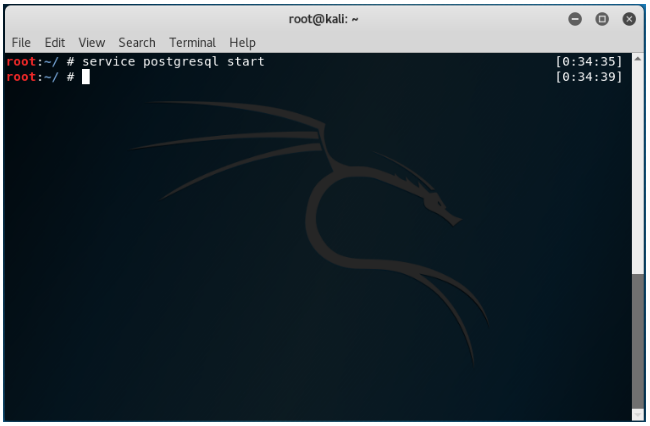


Fig. 5. Sarting postgresql service

Before the exploitation can begin, the necessary tools need to be prepared and initialised. This exploitation requires Metasploit framework as backend and Armitage as front end. Since the Metasploit framework requires PostgreSQL database, as figure 5 shows, the postgresql service needs to start first.

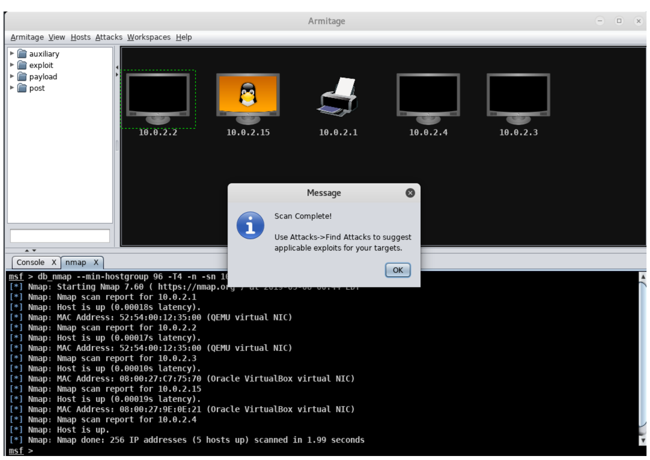


Fig. 6. Ping Scan

After the Armitage is successfully started, the first task is to detect all alive hosts within the same subnet using a ping scan provided by nmap. As figure 6 shows, all alive host with IP address can be matched by 10.0.2.* are shown here.

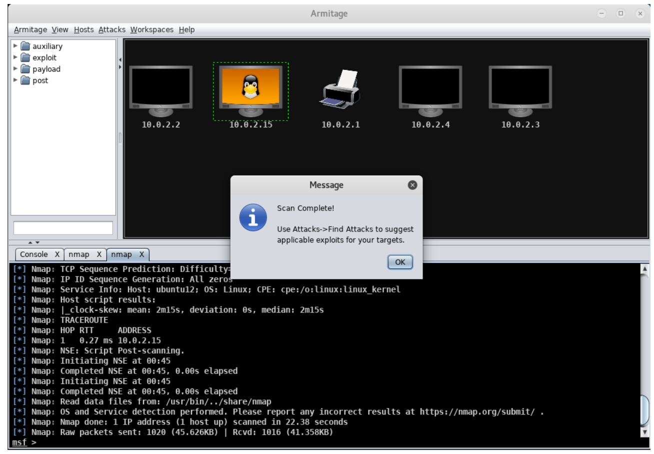
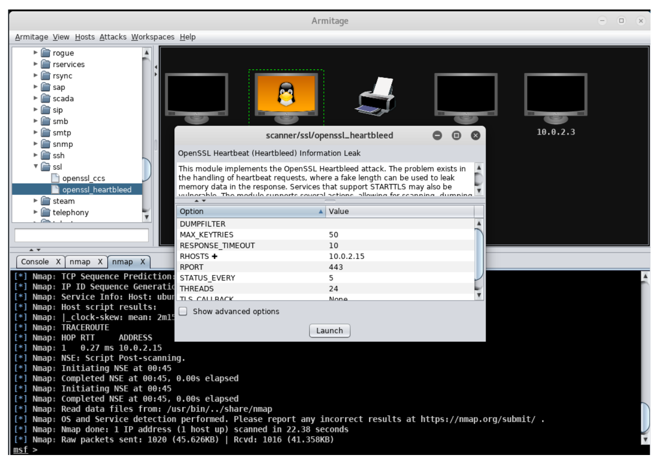
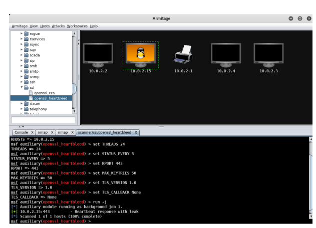


Fig. 7. Intense Scan

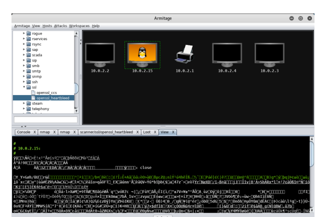
After all alive hosts are detected, the next step is to use intense scan provided by nmap to scan the target machine. As figure 7 shows, system version, open ports, services running and their version can be detected during this step. This step is essential because it can provide crucial information for the attacker to determine where the security vulnerabilities exist and how to attack this machine.



(a) Heartbleed Exploitation



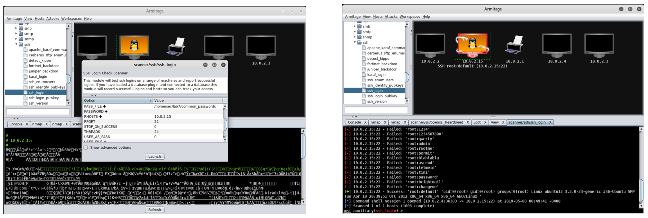
(b) Heartbleed Exploitation



(b) Heartbleed Exploitation

Fig. 8. Heartbleed Exploitation Results.

The next step as figure 8 suggests, the heartbleed exploitation is carried out. This attack can be successfully carried out due to the host running a very old version of system and services where the heartbleed vulnerability still exists.



(a) SSH Login Exploitation (b) SSH Login Exploitation

Fig. 9. SSH Login Exploitation Results.

The next step is the most crucial one as the full access can be obtained after this step is successfully carried out. This step is to use provided password dictionary to carry out brute-force password cracking. As figure 9 shows, the root password was successfully acquired. Consequently, the full access to the host is now obtained. Since no prior knowledge about the host is provided, with the success of root password cracking, all information required further can be obtained.

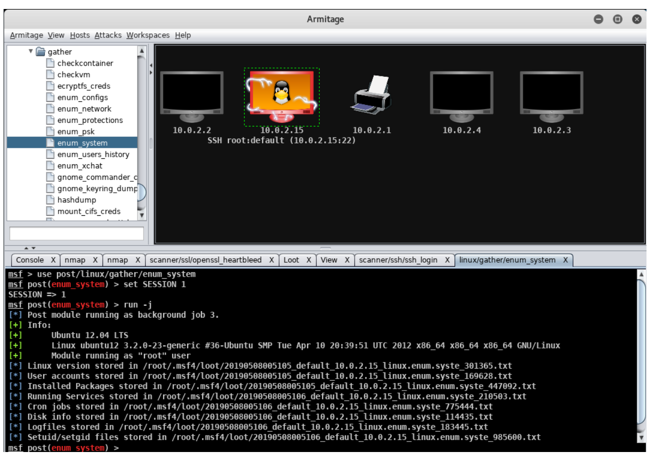


Fig. 10. Gather System Information

After the full access has been acquired, the last step of exploitation is to gather crucial system information like packages installed and their version, all currently running services, system information and kernel version, user list, and service list. As shown in figure 10, all this information is stored in text files.

3.2. Security Fixes

As mentioned in the previous section, the target host needs to be detected first before any exploitation can begin. In addition, the detection process was using the ping scan provided by nmap. Therefore, the first security vulnerability need to be fixed is to prevent the machine from being detected. In this case, a firewall rule was added to the Ubuntu machine.

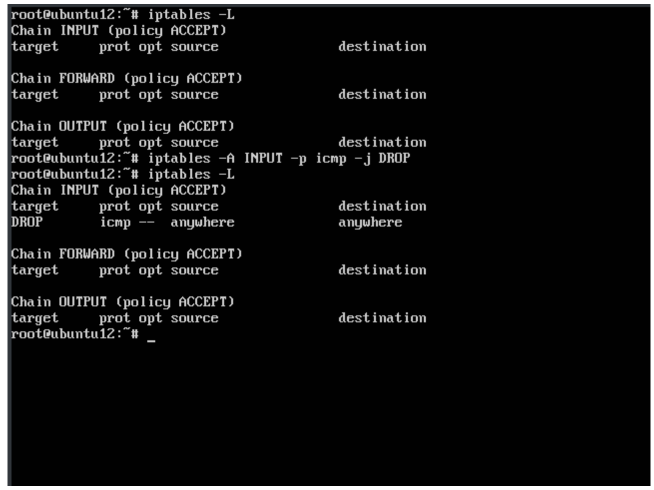


Fig. 11. IPTables

As shown in figure 11, the default firewall is quite permissive and allow any traffic by default. By adding a rule, all packet sent by ping action will be dropped.

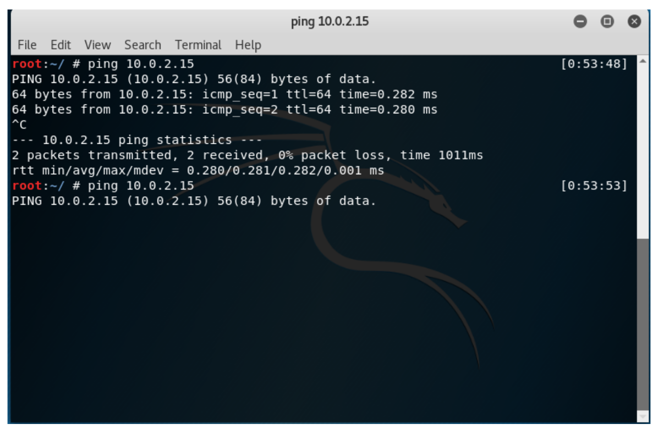


Fig. 12. IPTables Effect

Figure 12 shows the immediate effect after this rule being enforced. The first attempt suggests the firewall with default settings. It is obvious that the target host was responding as normal which is a positive confirmation of the target state. After the firewall being updated, the target stopped responding to the ping request. The target machine is either confirm nor deny, the ping requests. Hence, the attacker cannot determine the state of the target host. Therefore, in this case, the attacker may turn to the targets that return a positive response. Technically, other approaches can be employed to detect the state of the target, but psychologically this method is quite like effective.

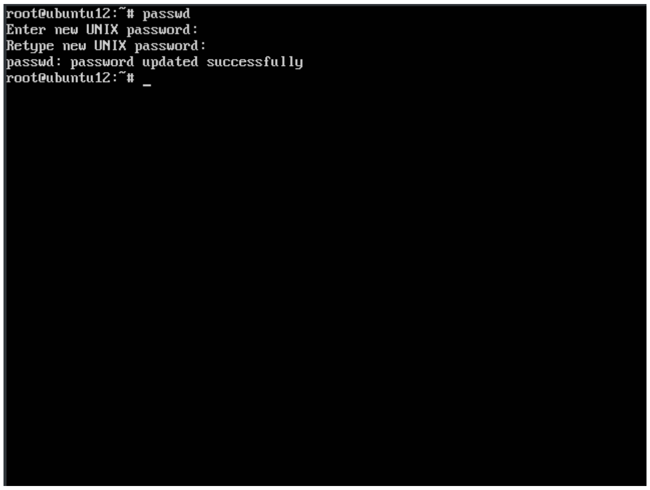


Fig. 13. Changing Password

The second crucial step of exploiting this machine is to crack the root password by brute-forcing. Therefore, the first step of fixing this vulnerability, as figure 13 shows, is to change the current password into a strong one which will significantly increase the cost of brute-forcing. Due to the time constraint, password checking against publicly known dictionaries is yet to be implemented.

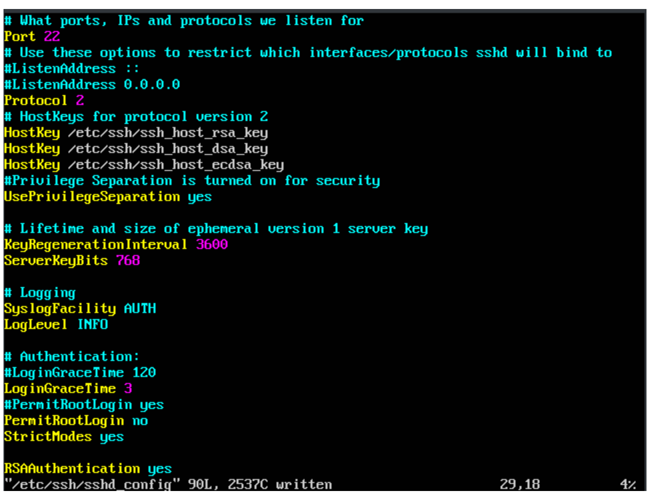


Fig. 14. Changing Password

Another factor that allows the brute-force cracking to continue is the maximum number of login attempts allowed was too big. As figure 14 depicts, the default setting was 120. This issue can be easily fixed by changing this number to a small one(in this case, it was set to three).

By default, almost no password policy was enforced. Other than policies implemented earlier, all of the password policies proposed for question one should be employed given enough time.

In the Unix system, the root is the most powerful user as it has permissions to do anything. Therefore, it would be a disaster for systems to have root password breaching. The ultimate solution to address this issue is to forbid user login as root as shown in figure 14. Furthermore, the only solution to prevent password leaking for all user is to prevent users from login using the password.

Fig. 15. Running Service List

Figure 15 shows the list of running services. Telnet and ftp service are provided by openbsd-inetd service which should never be used as they transferring all information in plain text without any encryption.

Due to the time constraint, all packages in the Ubuntu system are still in default version. Almost all packages alongside the kernel should be updated as they are hopelessly outdated. Many security breaches that should be fixed for a long time may still exist in this system.

4. REFERENCES

[1] Dieter Gollmann, *Computer Security*, Hoboken, N.J., 3rd ed. edition, 2011.

[2] Bruce Schneier Niels Ferguson and Tadayoshi Kohno, *Cryptography Engineering: Design Principles and Practical Applications*, Indianapolis, Ind., 2010.

[3] Ross Anderson, *Security Engineering: A Guide To Building Dependable Distributed Systems*, Indianapolis, Ind. ; Chichester, 2nd ed. edition, 2008.

[4] Buddhika Chamith, "Ssh tunneling explained," `https://chamibuddhika.wordpress.com/2012/03/21/ssh-tunnelling-explained/`, 2012, Accessed: 2019-05-01.