Lab 12 – Ethan Roepke

Identify the vulnerability	What harm could it do?	How to fix it
1. When I check the file permissions for /etc/shadow, this is readable to the outside world which is not practical.	Having /etc/shadow readable to the world gives attackers access to obtain hashed passwords that are stored. The attacker will be able to use brute-force to crack the password and gain unauthorized access to user accounts. This can cause much more harm with being able to gain access to more servers or users or trying to climb to gain higher level access.	To simply fix this, we need to change the file permissions for the world from *r- - to* - - -. This will protect from attackers from being able to decode the passwords without having access to /etc/passwd. To go a step further to more protection we can only allow the owner to be able to read and write.
2. When using *cat /etc/passwd,* I have found an inaccurate and suspicious user. The user was Bob, who was fired from the company and should have no access at all.	This is a vulnerability because Bob was fired but still has access to the company information even not being with the team no more. As well as Bob being a user in the server, there is also a backdoor user as well. Backdoor is a way of bypassing normal authentication and gaining access. With the backdoor, Bob will be able to gain access to the server from anywhere and doing many malicious attacks to the server.	To fix this, Bob is no longer an employee for the company so we just need to delete his user all around. We will do sudo *userdel Bob.* Since backdoor could be Bob or others being able to bypass the authentication, we will do the same by using *sudo userdel Backdoor*
3.When running *ufw status,* it returned as inactive which lets us	This is a vulnerability to the server because it can	To be able to fix this we will do *sudo ufw enable*

know that we have no firewall enabled. This is a host based firewall that is implemented into the server itself to protect from unauthorized access. The host based firewall is a second layer of defense for you.	cause many different types of harm, including increased to unauthorized access, services or applications running on the server are at risk, and compromising of the system.	then we can check if it is active by doing *ufw status.* After checking if active, you will be able to add the rules for IP address, ports and what to allow to income and go out.
4. I was looking through files to see if any rules were inaccurate and when I did *nano /etc/ssh/sshd_config,* I scanned through and *PermitEmptypasswords* was set to yes. The line above in bold text says *change to yes(NOT recommended).* This obviously will not be an ideal real world practice since we do not want to be able to log in with no password.	This can be harmful because you can login into ssh with no password required. This is terrible practice because allows unrestricted access to user accounts with no authentication at all. Attackers can use brute force which attempt many username/password and find the users who have not set a password and gain access. A big consequence is compromising accounts, data loss, and manipulation.	To fix this, if we go into /etc/ssh/sshd_config we can change PermitEmptyPasswords from yes to no to require a password to be setup.
5. In the Postfix file, SMTP server is the crucial security measure to enforce encrypted connections but was disabled. The section under the *postfix submission* configuration, the ability to exclusively permit encrypted connections has been rendered ineffective. Another feature allowing precise control over who can email you is achieved by uncommenting the line *smtpd_reject_unlisted_recipient*	It is commented out so it is in plaintext protocol which means it is not encrypted and returns with possible personal data in plaintext. While a file is being transferred, the man in the middle can modify the file without no one knowing.	To fix this, we will uncomment the specific lines and will encrypt the files so when they are being sent and received, the man in middle is not able to modify the file.
6. I ran a nmap scan and got multiple ports open. I tested each	FTP is a plaintext protocol which transfers	FTP is not encrypted but SFTP is encrypted so

port and checked all them, but port 21 (ftp) returned with text saying "Better hack yourself before you wreck yourself."	files between clients and servers. This will include many secure data that doesn't want to be leak so a man in the middle attacker can be dangerous. An attacker would be able to intercept and modify data during file transferring leading to unauthorized access. An attacker gaining access to manipulate files.	instead of using port 21 we will use port 22. SFTP will encrypt the information being transferred and uses SSH which provides a various authentication methods.
7. Port 80 or HTTP is a plaintext protocol and when I ran *netstat -tln,* I viewed all the ports being used and port 80 was one that was used. The issue with port 80 is that it is not encrypted.	This can cause harm because since port 80 is not encrypted, any data being exchanged in server will be sent in plain text. The lack of encryption can be easily intercepted by attackers. If a HTTP has a login page, they attackers could easily intercept the credentials causing access to user accounts.	HTTP is not encrypted but HTTPS is encrypted so simply we need to change from port 80 to port 443. HTTPS is important to use, especially with login pages that require sensitive data that can help keep that information encrypted.