Write Up:

During this lab, I used the SimSpace Kali-Hunt virtual machine to practice cracking different types of password files and to better understand the weaknesses involved in password storage and password policy enforcement. I will use the provided virtual machine to crack a shadow file using johnny, a SAM file using Ophcrack, and I will use a web based hash cracker called crackstation to show how easy and accessible password and hash cracking really is. I also found how unsecure not only my passwords could be, but also how password security is seen as more of annoyance rather than a necessity.

My experience began with launching the Kali-Hunt VM, which immediately showed performance issues. The virtual machine was extremely laggy and unresponsive at times, which made navigating the environment more difficult than expected. In addition, I discovered that several passwords in the environment were already cracked from previous users, so I had to clear old data and essentially start over to make sure I was performing the lab correctly. Internet connectivity inside the VM also did not work properly, which meant I could not rely on the built-in browser. Despite these obstacles, I proceeded with the assignment using the available local tools on my personal machine.

The first major task involved cracking the /etc/shadow file using Johnny. According to the lab instructions, first I opened the file manager and located the shadow file that was stored in a shared folder accessible from the VM, and I was able to locate it without any difficulty. Next I opened Johnny, as seen in figure 1, once I opened Johnny, I imported the shadow file directly into the program. I did not encounter any errors or problems, and the file loaded correctly on the first attempt.

One of the things that surprised and scared me the most was how fast the cracking process was. I expected the cracking attempt to take a long time, especially given how laggy the VM was, but Johnny cracked several passwords within the first five minutes. Examples of passwords it recovered included simple, predictable words such as “airforce,” “derf,” and “redbox.” These results clearly demonstrated how weak many user-selected passwords can be in real-world environments. This proves that weak passwords are easy to crack and should be avoided if possible. After taking the required screenshot, I cleared the session history as instructed in order to remove any evidence from the VM.

Next, I moved on to cracking the Windows SAM file using Ophcrack. Although Ophcrack is typically associated with Windows, the Kali VM came with a Linux version already installed, so I decided to use it instead of switching VMs. As seen in figure 2, Ophcrack ran smoothly despite the performance issues that affected other applications. Within a little over one minute, Ophcrack successfully cracked four different passwords. The passwords recovered included examples such as “HEYWOOD” and the more complex “N3verm0re.”

These results showed how both simple and moderately complex passwords can still be vulnerable depending on the hashing algorithm and the strength of the tables being used. The process was error-free, and I did not have to troubleshoot anything during this part of the lab.

Because the Firefox browser inside the VM crashed any time I tried to launch it, I could not test the hashes on CrackStation directly from SimSpace. Instead, as seen in figure 3, I copied the hash values from Ophcrack and transferred them to my personal computer. This made it much easier to perform the tasks I needed to in the lab, without the fear of crashes or missed inputs.

On my own system, I opened CrackStation.net, pasted the hash values, and used the online cracking engine to compare results. CrackStation successfully returned matching passwords for the hashes, though some had slight formatting differences. Even with these variations, the cracked results were consistent with what Ophcrack had found. This confirmed that LM and NT hashes, especially weak or common ones, are extremely easy to crack not only with local tools but also with publicly available online databases. I did not experience any problems with CrackStation, including CAPTCHA or input formatting issues, making this portion of the lab straightforward. The ease of use and how quickly it worked genuinely made me reconsider the passwords I use on a daily basis to secure my accounts.

The next task required locating the password file on my own computer. My personal machine runs Linux Mint, so I knew the primary password file would be located in the /etc/shadow directory. As shown in figure 4, accessing it required opening a terminal and navigating to the /etc directory using cd /etc. From there, I used sudo cat shadow to view the file.

Unlike the VM, my system required proper authentication to access the file, which demonstrates how Linux protects sensitive password information by restricting it to administrative users only. This step was much easier than using the virtual machines, and I had no difficulty locating or viewing the file once I used the correct command.

Overall, the most challenging part of the lab was working around the lagginess and instability of the SimSpace environment. Basic actions like moving windows, typing, and loading programs were slowed down significantly, which made the process feel more tedious than it should have been. However, the easiest part was locating my own system’s password file, which took only a few terminal commands.

The lab taught me how surprisingly easy it can be to crack weak or predictable passwords using free tools. I was also surprised by how fun and engaging the cracking process was once I got past the performance issues. This lab gave me a much clearer understanding of why strong password policies are necessary and how quickly poor password practices can lead to compromised accounts. I found out how unsecure some of my personal passwords were, which made me change them to be up to par with the NIST recommendations, as well as my family members as well. This lab made me realize how weak many peoples' security is in my personal life, and how little most of us do to protect these important accounts.

Reflection:

A strong password policy is essential for protecting user accounts, limiting unauthorized access, and reducing the likelihood of successful brute-force or dictionary-based attacks. Passwords still remain one of the most widely used authentication methods for users online, yet they also represent one of the most common points of failure in cybersecurity.

Password cracking remains as one of the most common attacks in todays technological landscape, according to the paper titled, Password Security: Cracking Techniques and Countermeasures, the authors state “A password-cracking attack is just a guessing attack. An attacker makes guesses about a user’s password until they guess correctly or give up. While the defender may limit the number of guesses an attacker is allowed, a password’s strength often depends on how hard it is for an attacker to model and reproduce how a user created their password.”(Al-Haija, Abu-Ghazaleh, Hafez, Mansour, Al-Jammal, 2024). Companies and organisations need to have password policies in place, to help mitigate or even prevent password attacks.

A well-designed password policy helps ensure users create credentials that resist guessing, cracking, or simple pattern-based attacks. Research shows that users tend to choose short, predictable, and easy-to-remember passwords, which makes them highly vulnerable to cracking tools such as John the Ripper, Ophcrack, and online hash-cracking databases. The ease with which common words like “redbox” or “airforce” were cracked during this lab highlights the importance of policies that encourage stronger password creation.

One major component of a strong password policy is minimum password length, which modern standards treat as far more important than complexity alone. The National Institute of Standards and Technology (NIST) recommends a minimum length of 15 characters to support passphrases instead of short, complex strings, they state “NIST guidance recommends that a password should be at least 15 characters long. At 100 billion guesses per second, it would take a computer more than five hundred years to guess all the possible combinations of 15 lowercase letters.” (NIST, 2017).

Passphrases, which contain unrelated words, numbers, and symbols significantly increase complexity while keeping it easy to remember. Complexity requirements such as mixing uppercase letters, lowercase letters, numbers, and symbols can help, but they are less effective when users rely on substitutions such as leetspeak, like replacing “o” with “0” or “e” with “3,” which cracking tools detect easily. The password “N3verm0re” recovered during this lab demonstrates this problem, even though it meets typical complexity requirements, it was still cracked quickly.

According to a peer reviewed source titled, Reinforcing cybersecurity with Bloom filters: a novel approach to password cracking efficiency, they state “The future of industrial cybersecurity hinges on the ongoing evolution and adaptation of these strategies, keeping pace with the changing digital landscape and the sophisticated nature of these cyber threats.”(Villafranca, Cano, 2024). This shows that just how password cracking evolves over time, so do our passwords. As

computers get more advanced and so do password cracking tools, passwords need to become more advanced to protect against attacks like dictionary based, brute force, and credential stuffing.

Another important element of password policy is preventing password reuse across different systems or accounts. Attackers frequently rely on credential-stuffing attacks, using previously leaked passwords to break into new accounts, because many users reuse the same password across multiple services. Organizations can reduce this risk by enforcing password history rules that prevent reusing old passwords, along with checking new passwords against known breached password lists. Even though many corporations and companies do enforce password history rules, not many do enough. Frequently adding a number or symbol, or even just capitalizing a previously lowercase letter will be enough to tell the system that the password is different, when in fact not nearly enough has been done to secure the accounts. NIST now recommends that organizations compare user passwords against public breach databases to prevent overly common or compromised passwords from being selected (NIST, 2017). With the prevalence and ease of access these password cracking tools offer, it makes sense why passwords are so vulnerable to security attacks, and how important strong passwords are today.

A strong password policy must be both adequately secure, yet also not convoluted enough for people to resort to unsecure practices like adding a number or symbol to the end of an already insecure password. When password rules become too convoluted, many people will resort to having the same password for many websites or systems, leading to an easy cyber attack when everything is secured with the same password. It is common knowledge that when asked to change passwords too many times, people will often default to one password, making small variations on it, as remembering a whole new password every few months is difficult. Instead, organizations should encourage the use of longer, more memorable passwords and support password managers, which help users maintain unique, secure passwords without needing to remember them all manually.

Looking at the password policies in my work, school and home environment, they generally follow the traditional complexity rules: requiring uppercase letters, numbers, and symbols. While these rules provide some level of security, they do not fully commit to the recent NIST recommendations about emphasizing length, which I am definitely guilty of. Based on what I observed in this lab, the current policies could be improved by encouraging longer passphrases, removing unnecessary forced password resets, and incorporating breached-password checks. Using these updated practices along with following the NIST guidelines and recommendations, along with common sense, would make the password policy more effective and reduce the likelihood that an attacker could successfully crack or guess user passwords.

This lab demonstrated just how quickly weak or moderately complex passwords can fall to automated cracking tools. The outcomes of this lab prove the importance of using strong passwords and policies not just in work environments but also in personal and home environments too. By focusing on strong passwords using length and uniqueness as foundations, we can significantly reduce the exposure to password cracking and credential stealing in the workplace and home respectably.

Appendices:

Figure 1: A screenshot of Johnny successfully cracking passwords.

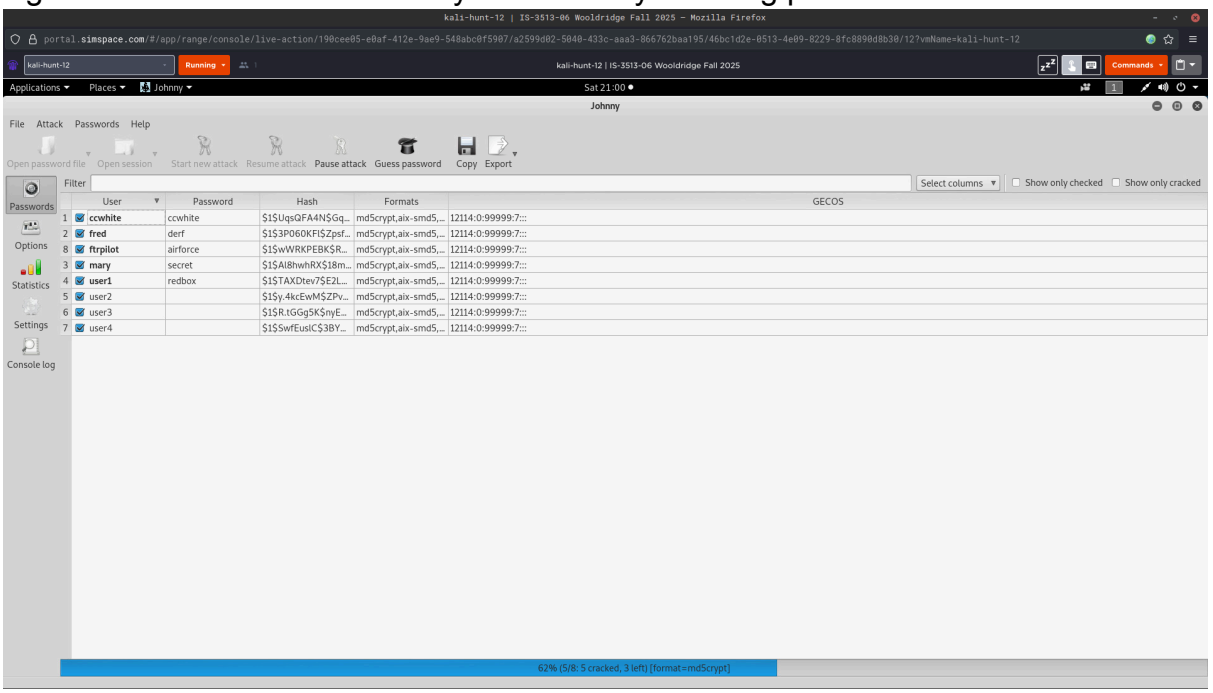

Figure 2: A screenshot of ophcrack craving 4 passwords in one minute.

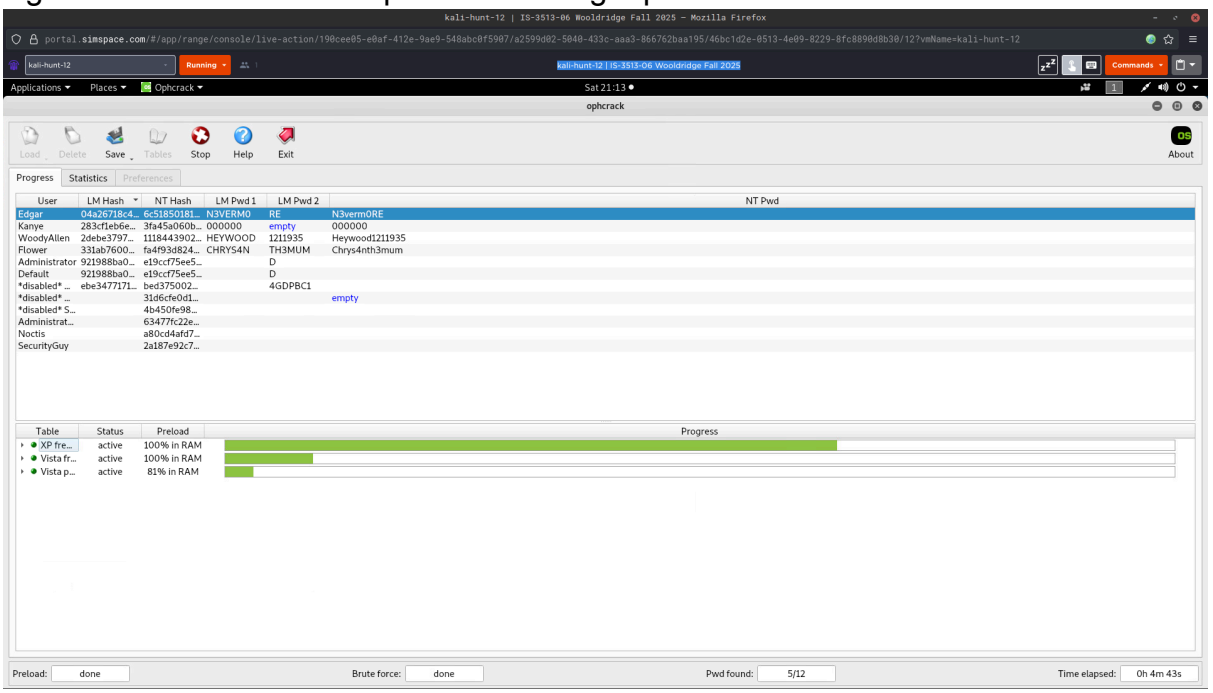

Figure 3: A screenshot of crackstation cracking a hash.

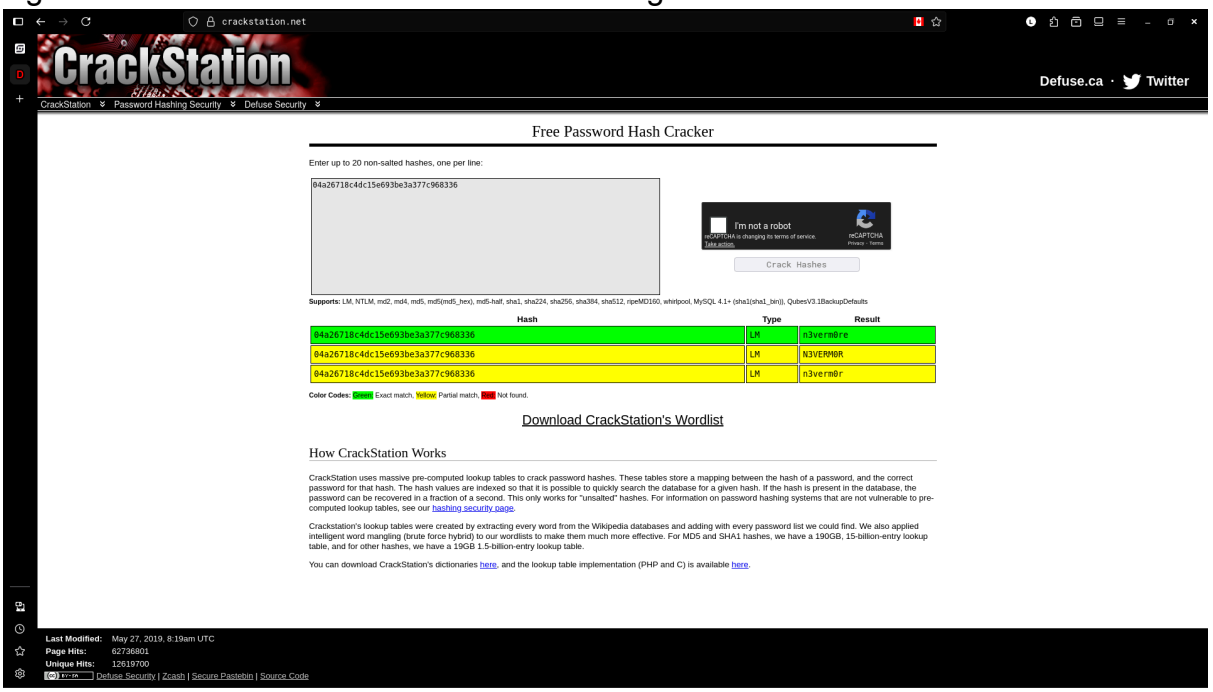

Figure 4: A screenshot of a Linux terminal, showing the shadow file highlighted.

Redacted for Privacy

Bibliography:

NIST. (2017). *Digital identity guidelines* (NIST Special Publication 800-63B). National Institute of Standards and Technology.

Stobert, E., & Biddle, R. (2014). The password life cycle: User behavior in managing passwords. *Eleventh Symposium On Usable Privacy and Security (SOUPS)*.

Cybersecurity and Infrastructure Security Agency. (n.d.). *Multi-factor authentication (MFA)*. U.S. Department of Homeland Security. Retrieved November 16, 2025 https://www.cisa.gov/MFA

How do I create a good password?. NIST. (2025, August 20). https://www.nist.gov/cybersecurity/how-do-i-create-good-password

Tasic, I., Villafranca, A., & Cano, M. D. (2024). Reinforcing cybersecurity with Bloom filters: a novel approach to password cracking efficiency. *EURASIP Journal on Information Security*, *2024*(1), 35.

Al-Haija, Q. A., Abu-Ghazaleh, R., Hafez, A., Mansour, S., & Al-Jammal, Y. (2024, June). Password Security: Cracking Techniques and Countermeasures. In *International Conference on Data Analytics & Management* (pp. 91-99). Singapore: Springer Nature Singapore.