Lab 1 – Crapple Security Training - Linux, CIA, and the Attacker Mindset

Employee: You (New Hire)
Supervisor: Seymour Bugs (Security Guru)

Introduction (Seymour Bugs):

Welcome to Crapple, newbie! I'm Seymour Bugs, your guide through the wild world of security. Forget those fluffy "welcome aboard" memos; here, we dive headfirst into the digital trenches. Week one is all about the fundamentals – Linux, the CIA triad (no, not the spy agency, although they are interested in us), and thinking like the bad guys (because, let's face it, they're thinking about us). A crucial aspect of security engineering is the ability to not only implement security measures but also to analyze their effectiveness and potential weaknesses. This lab will challenge you to think critically about security and develop your analytical skills.

Your Mission (Lab 1):

Your first task, should you choose to accept it (and you really don't have a choice), is to conquer the Linux command line and bend it to your will. Think of it as your security lightsaber. Then, we'll use this newfound power to explore some core security concepts. Don't worry, it's not rocket science... unless you're building rockets, in which case, you're really in the wrong department.

Step 1: Acquire Your Weapon (Docker Image):

I've prepared a special package for you: a Docker image pre-loaded with a lean, mean Linux distribution. Think of it as your own personal sandbox (where you can play with fire, within reason).

1. Open your terminal (or command prompt, if you're still living in the dark ages).
2. Type the following incantation: `docker pull dlambros/labs:lab1`. This downloads the image to your machine. Don't worry, it won't bite... unless you try to run it on a potato.
3. You can verify your image is downloaded by running `docker images`

Step 2: Enter the Sandbox (Run the Container):

Now, let's bring that sandbox to life:

1. Type: `docker run -it dlambros/labs:lab1`
2. Boom! You're inside a Linux container. It's like stepping into a tiny, virtual world. Don't get lost!

Step 3: Linux Kung Fu (Basic Commands):

Time to learn some Linux Kung Fu. These commands are your bread and butter (or, if you prefer, your tofu and sprouts).

1. **Navigation**: cd (change directory), pwd (print working directory), ls (list files). Practice moving around the file system. Imagine you're Indiana Jones searching for hidden artifacts, but instead of a whip, you have these commands.

2. **File Manipulation:** touch (create a file), echo (display text), cat (display file contents), cp (copy a file), mv (move a file), rm (remove a file). Relate this to data integrity. Create a file, write something in it, then intentionally mess it up. See how easy it is to break things? Now, try fixing it.

3. **Permissions:** chmod (change permissions), chown (change ownership). Crucial for least privilege. Experiment with different permission settings (read, write, execute) for different users. It's like deciding who gets the key to the executive washroom.

Step 4: The CIA Triad and Other Spooky Stuff:

Now, let's talk about the CIA – Confidentiality, Integrity, and Availability. No, not *those* guys.

1. **Confidentiality:**

   a. **Action:** Create a file called top_secret_crapple_plans.txt. Put some super-secret stuff in it (like the recipe for the perfect avocado toast). Use file permissions (chmod) to make sure only you can read it. Imagine you're protecting the launch codes… for the coffee machine.

   b. **Analysis:** Experiment with different chmod settings (e.g., 600, 644, 755). What was the setting you went with to meet requirements? Explain what each setting means and how it affects confidentiality. Research and explain how Access Control Lists (ACLs) can provide more granular control. Discuss the potential consequences of a data breach involving this file for Crapple.

2. **Integrity:**

   a. **Action:** Create a file. Calculate its checksum using sha256sum. This is like creating a digital fingerprint. Now, change the file's contents. Recalculate the checksum. See? It changed! This is how we know if someone has messed with our precious data.

   b. **Analysis:** Research different hashing algorithms (MD5, SHA-256, SHA-512). Which one is considered most secure and why? Explain how digital signatures use hashing to ensure both integrity and authenticity. How are checksums used to verify the integrity of downloaded software?

3. **Availability:**

    a. **Action:** Discuss how system downtime impacts availability. Imagine the Crapple servers going down right before a big product launch. Chaos! Simulate a mini-DoS attack (be gentle!) by creating a ton of files. Discuss how we can prevent this kind of digital mayhem.

    b. **Analysis:** Use `htop` to observe system resource usage (CPU, memory, disk I/O) during your simulated attack. Research different types of DoS attacks (e.g., flooding, amplification). What are some common mitigation techniques? What's the business impact of downtime for a company like Crapple?

4. **Least Privilege:**

    a. **Action:** The student user will attempt to access files in `/home/sensitive_data` and `/home/shared_data`.

    b. **Analysis:** Explain why the student user can or cannot access each directory. Relate this to file ownership and permissions (`chmod`). How does this demonstrate the principle of least privilege? If you are allowed in a directory, do you have the ability to `touch` files in it and why/why not?

5. **Attacker Mindset:**

    a. **Action:** Now, put on your black hat. How would you try to steal the avocado toast recipe, even with all these protections in place? Think like a villain!

    b. **Analysis:** Categorize the attacks you brainstormed (e.g., social engineering, technical exploits). Which ones are most likely? Which ones would have the biggest impact? Perform a basic threat model: What are the assets (e.g., the recipe)? What are the threats? What are the vulnerabilities?

Step 5: Secrecy, Trust, Threats, and Vulnerabilities:

1. **Secrecy and Trust:**

    a. **Analysis:** Can you have secrecy without trust? Can you have trust without secrecy? Explain your reasoning. How does cryptography help establish secrecy and trust, especially when trust is low?

2. **Threats and Vulnerabilities:**

    a. **Analysis:** What's the difference between a threat and a vulnerability? Perform a basic risk assessment for the scenarios in this lab: What are the threats? What are the vulnerabilities? What's the likelihood of each threat? What's the potential impact?

Step 6: Reflection Time (aka "Thinky Thoughts"):

Answer these questions thoughtfully, drawing on your analysis from the previous steps:

1. How does the concept of least privilege relate to confidentiality, integrity, and availability?

2. What is a real-world examples of a security breach that could have been prevented by better access control?

3. How can an attacker exploit vulnerabilities in file permissions to gain unauthorized access?

Step 7: Evidence Collection (aka "Show Your Work"):

To prove you actually did the lab and weren't just daydreaming about avocado toast, you need to provide evidence! Think of it as your digital alibi.

1. **Screenshots:** Take screenshots of key commands and their output. For example, show the output of `ls -l` after changing file permissions, or the checksums before and after modifying a file. Include screenshots of your resource usage analysis (`htop`).

2. **Command Log:** Copy and paste the commands you used into a text file. This shows your step-by-step journey through the lab. It's like a digital breadcrumb trail.

3. **Written Explanations:** For each exercise, write a brief explanation of what you did, why you did it, and what the results mean. Don't just say "I changed the permissions." Explain why that's important for confidentiality. Pretend you're explaining it to your grandma (who, hopefully, isn't a hacker). This is where you'll incorporate your analysis.

Step 8: Crafting Your Report (aka "Impress Seymour"):

Now, put it all together in a professional-looking report. Remember, presentation matters! Even if you're a security genius, a messy report makes you look like a… well, you get the idea.