

Lesson 5: Setting the Trap

Paragraph 1: A Plan Takes Shape (Cipher, Dot Notation) Noah and Olivia sat in the nearly empty computer lab, the glow of their screens illuminating their determined expressions. The school day had ended, but their work was just beginning. Noah leaned forward, scrolling through lines of code as Olivia watched over his shoulder. “We know Glitch is testing the system,” Noah murmured, his fingers tapping rhythmically on the keyboard. “So let’s give him something to find.” A **cipher** is a method of encoding information so that it appears scrambled or hidden. In this case, Noah plans to use a digital cipher to disguise the trap as something valuable. To make sure every action is recorded properly, they will use **dot notation**, which allows them to track specific data interactions in a structured way.

Paragraph 2: Baiting the Hacker (String Concatenation) Olivia smirked, already seeing where this was going. “You mean a trap? A file that looks valuable enough to bait him in?” Noah nodded. “Exactly. If he thinks he’s hacking something important, he’ll leave a trail.” Olivia crossed her arms, considering. “Alright, but how do we make sure we catch him? He’s smart—if it looks too easy, he’ll suspect something.” Noah grinned. “That’s why we use a cipher. We hide the real trap inside something that looks encrypted. If he tries to crack it, we log everything he does using dot notation to structure the output.” The cipher acted as an encoded message, something Glitch would think contained valuable data but was really just a mechanism to track him. Dot notation would allow them to track exactly which parts of the system he interacted with, ensuring every one of his actions was recorded.

Paragraph 3: A Cipher Inside a Cipher (String Concatenation) Olivia raised an eyebrow. “So... a cipher inside a cipher?” Noah chuckled. “Something like that. We’ll also use **string concatenation** to make sure any data Glitch tries to retrieve will still look legitimate, keeping him busy while we track him.” **String concatenation** is a programming technique that joins multiple strings of text together, allowing them to construct realistic, misleading data inside their trap to make it appear authentic.

Paragraph 4: Constructing the Decoy (Test Variable) They spent the next hour crafting the perfect decoy: a file named `admin_access.tmp`, embedded deep within the school’s shared network. To the untrained eye, it looked like a cache of encrypted administrator permissions. But behind the scenes, it contained a silent tracker, designed to log Glitch’s access, document his methods, and report back to them. “If he tries to read it, we’ll know,” Noah explained. “If he tries to edit it, we’ll see how. Either way, we’ll get a lead.” The system relies on a **test variable**, a special placeholder in the code that checks if an action has occurred. In this case, the test variable will determine whether Glitch has accessed or modified the file.

Paragraph 5: A Critical Error (Logical Operator, Test Variable) Olivia grinned. “This is genius. But let’s double-check everything before we set it live.” She leaned over Noah’s shoulder, scanning the code. As she read through the trap’s logic, her excitement faded. Something wasn’t right. Frowning, she pointed to a section of the script. “Wait. This won’t work.” Noah turned to her, confused. “What do you mean?” “The trigger only activates if the file is modified, right?” Olivia asked. Noah nodded. “But what if he just copies it? Or scans it without changing anything? We won’t get any data,” Olivia continued. “Also, I just noticed a **logical operator** issue. If Glitch accesses the file in an unexpected way, the **test variable** might not

trigger correctly.” **Logical operators** control the flow of decision-making in programs, ensuring that specific conditions are met before an action occurs. If not structured properly, logical operators can cause the trap to fail by allowing unintended access without triggering the tracker.

Paragraph 6: Rethinking the Plan (Cipher’s First Debug) Noah’s expression shifted as he realized the flaw in their plan. She was right. If Glitch was careful enough, he could slip in and out without triggering anything. Their whole plan hinged on the assumption that he would edit the file, but that wasn’t a guarantee. Olivia smirked. “Looks like Cipher needs to debug his own trap.” Debugging is the process of identifying and fixing errors in code. Noah now has to refine their plan to ensure the trap works under all conditions.

Paragraph 7: A Name Takes Hold (Cipher’s Identity) She leaned back, triumphant. “Cipher, huh? I like that. You should keep it.” Noah rolled his eyes, but the nickname stuck. The name fits—Noah specializes in solving and creating ciphers, turning his problem-solving skills into an identity.