Software Security Final Exam

1. **In your own words, choose a vulnerability from the OWASP Top 10 and explain how it can be exploited**

Hook: say a web application lets you make talking ASCII cows, but instead of just making a fun cow with speech bubbles, it also lets you run a command on the server. Command injections are a vulnerability that could turn something as innocent as a talking cow on your terminal to a gateway for full control over your server.

Thesis: command injection occurs when user inputs are passed directly into system commands and exploited to execute commands on the server. By exploiting this, an attacker could execute commands on the server.

Defense:

1. **User Input Manipulation**: The main vulnerability is caused by how the application takes user input for the mooing and cow parameters and passes it directly into the passthru() function. If we inject system commands into these parameters, we could manipulate the server to run those commands for us.
2. **Injecting Commands**: An easy way of exploiting this would be to just inject commands using the $(command) syntax. For example typing in http://MACHINE_IP:82/?mooing=test&cow=$(whoami), would make the server execute whoami and return it as output to a threat actor.
3. **System Enumeration**: If an attacker executes commands like id, uname -a, or ps -ef, they can gather critical information about the system. This would open the door for further, more serious attacks, such as privilege escalation or gaining access to sensitive data.
4. **Server Control**: If the threat actor's commands are successfully executed, they can essentially take control of the server, which would lead to a fully compromised system

Conclusion: So, by injecting commands through the cow or mooing parameters, a threat actor could break its intended function and execute commands on the server, potentially gaining full access to it

Call to action: prioritize input validation and proper sanitization techniques as well as best practices such as using prepared statements.

2. **In your own words, choose an encryption algorithm and what advantages it has in creating secure cipher text**

Hook: Today almost all our information is stored digitally - convenient for us, but even more convenient for hackers. And so, the number of data security threats have increased over the past few decades, with businesses and large companies being a particularly favorable target for hackers. This sets forth a need to balance availability, ease-of-use, and security, enabling easy operations for users while simultaneously ensuring data privacy and security.

Thesis: The Advanced Encryption Standard (AES) is one of the most widely used encryption algorithms available. Its efficient mix of speed, flexibility, and strong security makes it a great choice

for businesses and people seeking to safeguard their private information

Advantages:

1. It uses block ciphers and applies multiple rounds of substitution, shifting, and mixing to transform plain text into cipher text.
2. AES offers scalable security with key sizes of 128, 192, and 256 bits, allowing customization based on our needs.
3. The algorithm is flexible and adaptable for various applications like VPNs, Wi-Fi networks, mobile apps, and enterprise systems.
4. AES outperforms legacy algorithms like DES with faster encryption speeds and stronger resistance to brute-force attacks.
5. Proper implementation of AES ensures data remains secure, even against powerful computing systems, by preventing decryption without the correct key.
6. AES is highly resistant to known cryptographic attacks when correctly implemented.

Conclusion: AES is an encryption algorithm that strikes a good balance between speed and security, making it an essential tool for protecting sensitive data across different industries and applications. Its widespread use mixed with its flexibility and security features, ensures that businesses can rely on it to protect their information from unauthorized access.

3. **In your own words, choose an input validation method and explain how it ensures data integrity**

Hook: As systems become more complex and integrated, ensuring that the data inputs are valid and accurate becomes more crucial. A small mistake - or intentional manipulation - in data can lead to security vulnerabilities or malfunctions. To fight against this, input validation plays a critical role in maintaining data integrity.

Thesis: Semantic validation is an input validation technique that ensures the correctness and meaning of the data, making sure that it matches the format we expect but also adheres to the logical rules of the application and the real world (e.g. ending date can’t be before starting date)

Defense:

1. Semantic validation ensures that the inputted data makes sense in the context of the system or application or the real world by verifying its relevance and logic.
2. Unlike syntactic validation, which checks if data follows a specific structure (like an email address or phone number format), semantic validation checks whether the data fits within the logical constraints defined by the business or system rules. For example, ensuring that a user’s age is a positive integer within a specific range (e.g., 18–120 years) or that a date of birth is not in the future.
3. This helps prevent invalid data from entering the system, such as incorrect date entries or non-existent product IDs, which could cause errors or malfunctions.
4. By implementing semantic validation, systems can provide users with more meaningful feedback or error messages, ensuring that the input aligns with the expected outcome, thus improving both

data integrity and user experience.

Conclusion: Semantic validation is a crucial method for maintaining data integrity by ensuring that the input is not only syntactically correct but also makes sense. This technique protects systems from invalid or logically incorrect data, allowing businesses and apps to run more smoothly and securely while delivering accurate results to users.