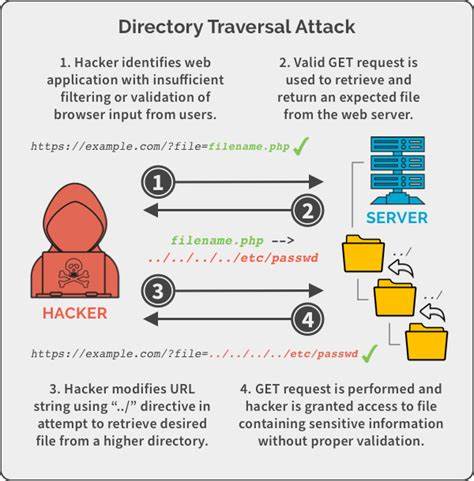
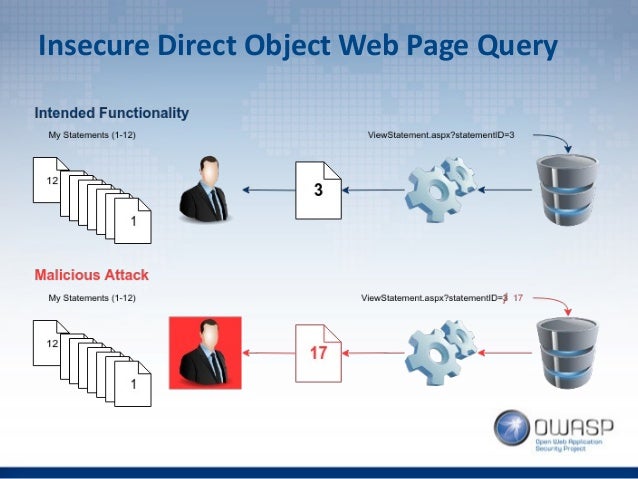
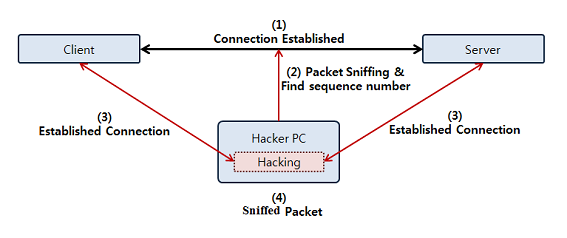
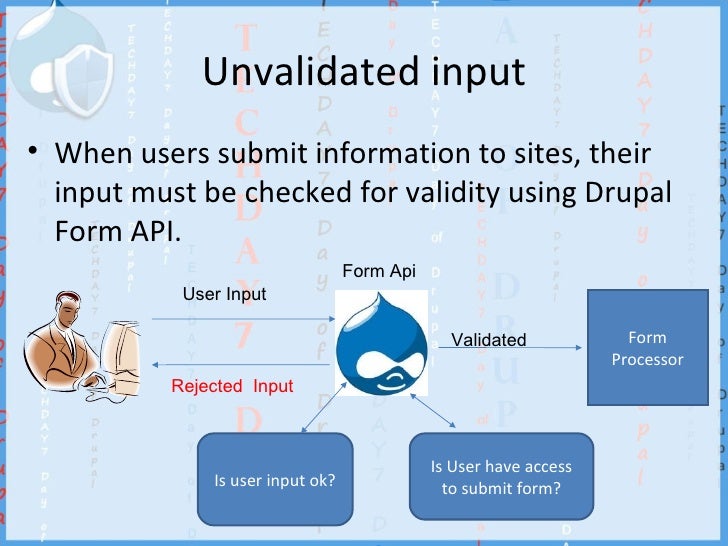
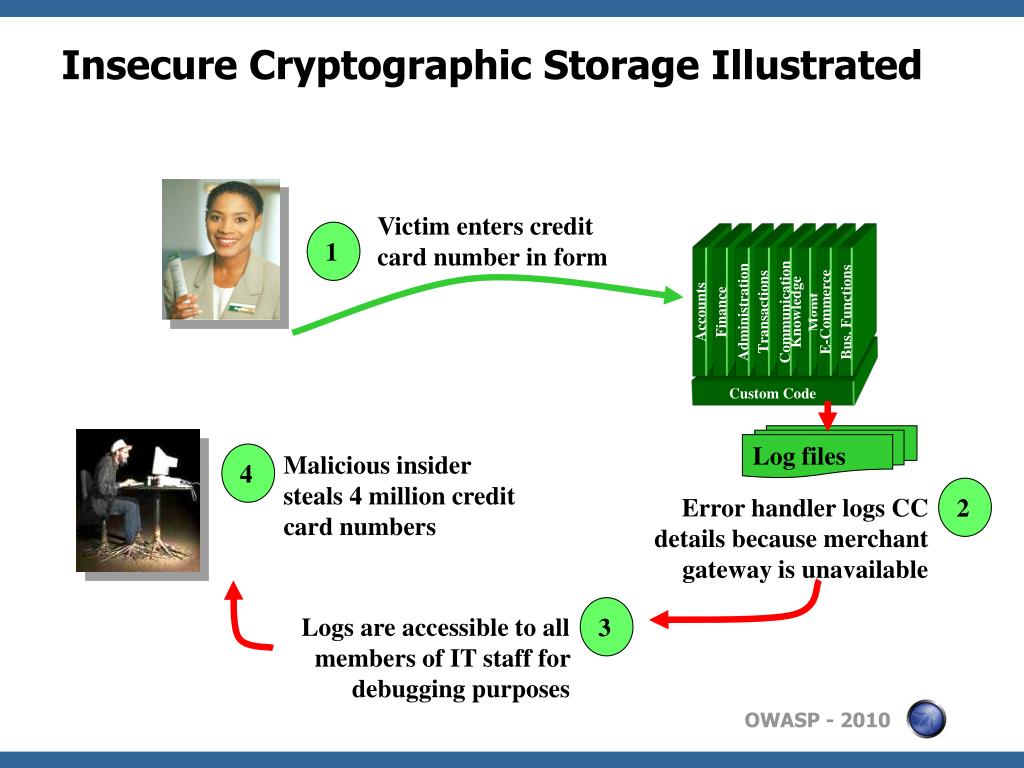
OWASP 10 attacks (top 10 excluded)

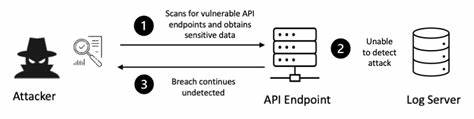
1. Directory traversal is an attack that allows an attacker to access files and directories that they should not be able to access. This can be done by exploiting vulnerabilities in the application's file system permissions. For example, an attacker might be able to access a file containing sensitive data by entering a specially crafted URL that includes the path to the file. 
2. Insecure direct object references occur when an application exposes a reference to a sensitive object, such as a user record, without proper access controls. This can allow an attacker to access the object without authorization. For example, an attacker might be able to access a user record by entering the user's ID in a URL.

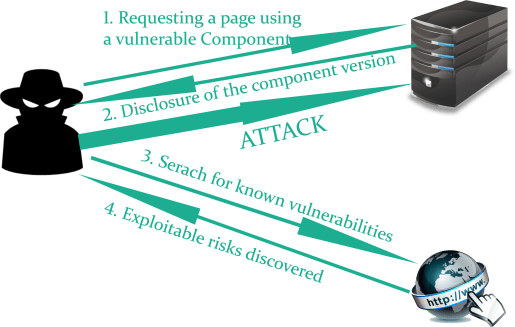


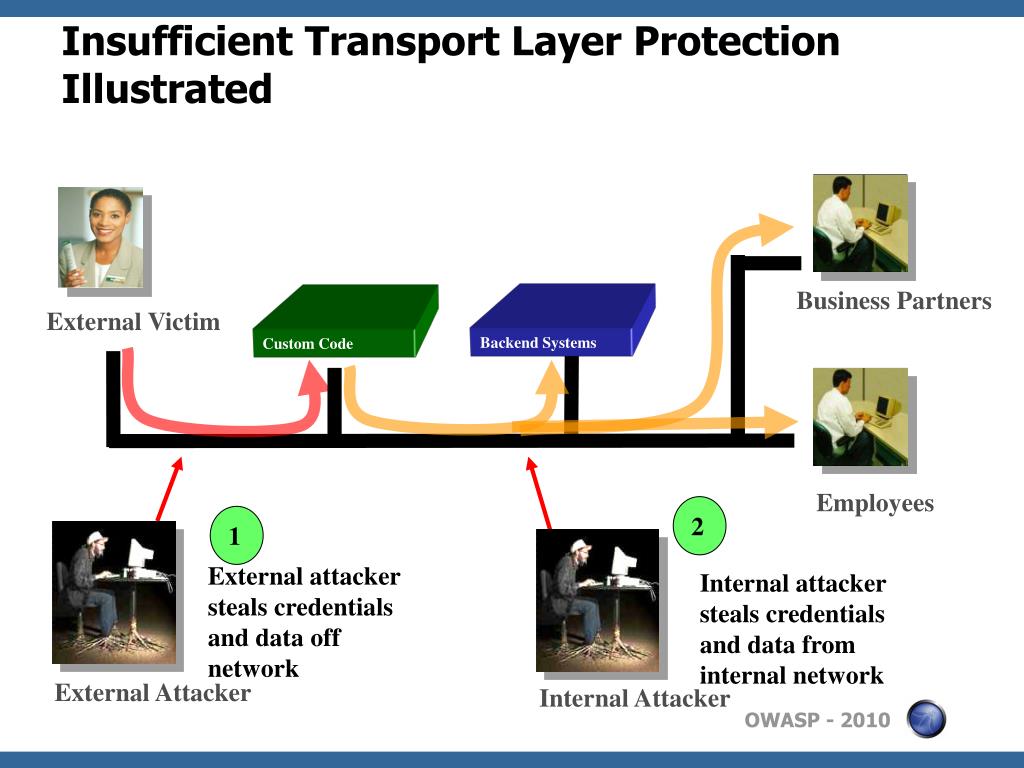
1. Improper error handling can expose sensitive information to attackers, such as stack traces or error messages that contain user data. This information can be used to exploit other vulnerabilities or to launch targeted attacks. For example, an attacker might be able to use an error message to determine the version of the application that is running, which could then be used to exploit a known vulnerability.
2. Insufficient session management can allow attackers to hijack sessions and take over the authenticated user's account. This can be done by exploiting vulnerabilities in the session management code or by stealing the session cookie. For example, an attacker might be able to hijack a session by using a malicious cookie that they have tricked the user into clicking on.



1. Unvalidated input can allow attackers to inject malicious code into an application. This can be done by entering specially crafted data into forms or by exploiting vulnerabilities in the application's input validation code. For example, an attacker might be able to inject malicious code into an application by entering a script tag into a form field. 
2. Insecure cryptographic storage can allow attackers to steal sensitive data, such as passwords or credit card numbers. This can be done by exploiting vulnerabilities in the application's cryptographic code or by using weak encryption algorithms. For example, an attacker might be able to steal an encrypted password if the application is using a weak encryption algorithm. 
3. Insufficient logging and monitoring can make it difficult to detect and respond to attacks. This is because it can be difficult to identify suspicious activity without adequate logging and monitoring. For example, an attacker might be able to launch a denial-of-service attack against an application without being detected if the application is not logging enough information.



1. Using components with known vulnerabilities can introduce vulnerabilities into an application. This is because components that are known to be vulnerable are often exploited by attackers. For example, an attacker might be able to exploit a vulnerability in a third-party library that is used by an application. 
2. Insufficient testing can lead to vulnerabilities being overlooked. This is because it is difficult to test for all possible vulnerabilities, especially in complex applications. For example, an attacker might be able to exploit a vulnerability that was not found during testing.



1. Poor code quality can make an application more vulnerable to attack. This is because poor code quality can make it easier for attackers to find and exploit vulnerabilities. For example, an attacker might be able to exploit a buffer overflow vulnerability that was caused by poor coding practices.