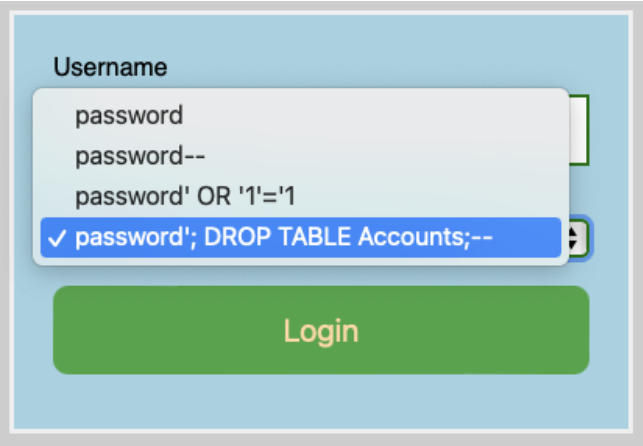
Introduction to Securing Express Applications

- Results of Cyberattacks against Websites:
  - Website defacement
  - Loss of website availability or in the worst case, total denial-of-service (DoS)
  - Leaking of sensitive customer data
  - An attacker gaining control over the website
  - An attacker using the website as a vector for other attacks
  - Loss of user trust in the website
  - Reputational damage

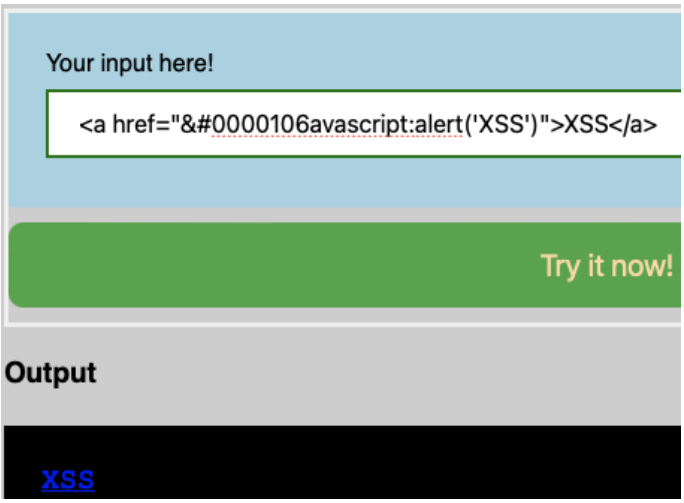
- Penetration Testing (Pen Testing): cyberattack is simulated in order to identify security vulnerabilities so that they can be discovered and remediated (also known as ethical hacking).

- CIA (Confidentiality, Integrity, and Availability):
  - Confidentiality: enforce access - who can see this, and who shouldn't
    - ex: implementing robust user authentication and encryption of important user data
  - Integrity: protect data from being changed or deleted, and damage can be reversed if done accidentally.
    - ex: database security, keeping backups, and using cryptography to check for changes
  - Availability: data being consistently, reliably available to those authorized.
    - ex: constant maintenance of hardware and software, monitoring servers and networks, and having a plan for any disasters

- OWASP (Open Web Application Security Project)
  - Top 10 Web Application Security Risks
    1. Broken Access Control: authorization isn't properly enforced, allowing attackers to access resources beyond their authorization..
       (ex: URL modification)
    2. Cryptographic Failures
       a. Sensitive Data Exposure: sensitive data is improperly or insufficiently protected. Insecure storage, the transmission of sensitive data, or revealing sensitive data to unauthorized parties.
    3. Injection: An attacker "injects" malicious code into an interpreter, usually through an input field, in order to gain access to information or damage a system



   a. Cross-Site Scripting (XSS): when a browser is tricked into running malicious javascript

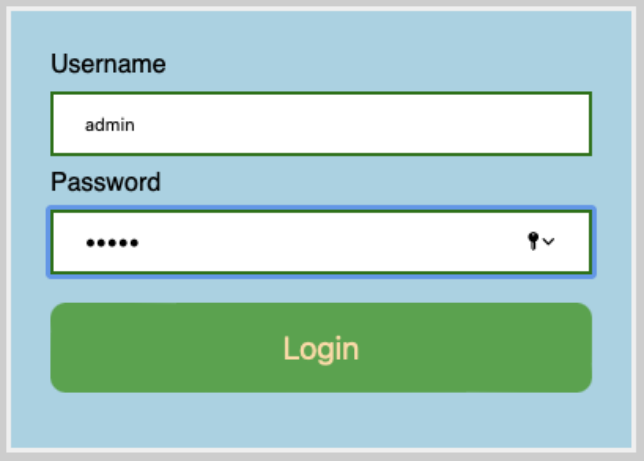


4. Insecure Design
5. Security Misconfiguration: Insecure, improper, or a lack of security configurations degrade the security of an environment.
   (ex: leaving unnecessary features enabled on server software)
   a. XML External Entities (XXE): a type of vulnerability that allows maliciously crafted data to produce unintended behavior on the backend of a website.
      (ex: a website using an XML processor can be hit by a malicious XML file)

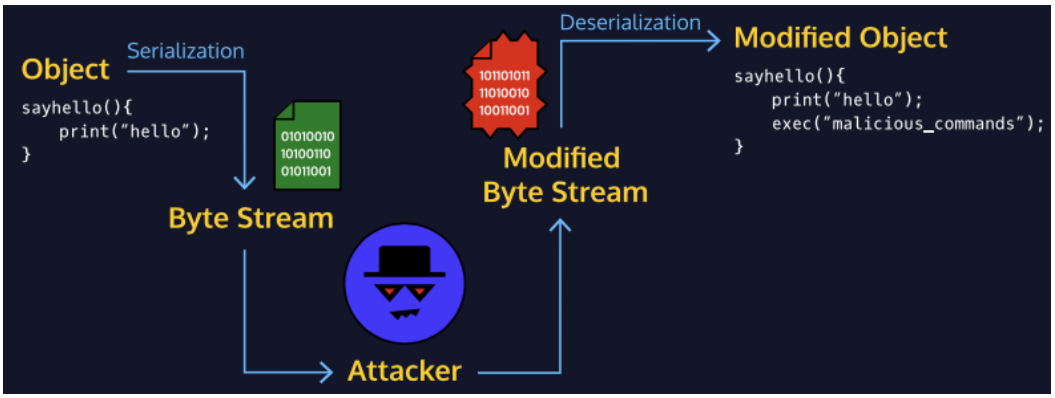
```
POST  /index.php/api/xmlrpc  HTTP/1.1
Host:  $host

<?xml  version="1.0"?>
  <!DOCTYPE  foo  [
    <!ELEMENT  methodName  ANY  >
    <!ENTITY  xxe  SYSTEM  "file:///etc/passwd"  >]>
<methodCall>
    <methodName>&xxe;</methodName>
</methodCall>
```

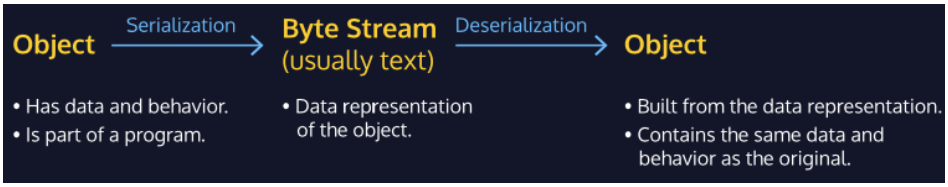
      i. `ENTITY` pointing to the `etc` password file, and the method name as `&xxe`
6. Vulnerable and Outdated Components
   a. Using Components with Known Vulnerabilities: Vulnerable components, such as out-of-date packages or software, are included within an environment, allowing attackers to use existing exploits to attack.
7. Identification and Authentication Failures
   a. Broken Authentication: an insecure authentication system allows attackers to impersonate other users.



8. Software and Data Integrity Failures
    a. Insecure Deserialization: data from an untrusted source is deserialized into an object, potentially containing malicious code or data, within a program. You can solve this is to not deserialize external data



    i. Serialization: the process of turning an object within a program into formatted data.
    ii. Deserialization: the process of turning formatted data into an object within code.



9. Security Logging and Monitoring Failures
    a. Insufficient Logging & Monitoring: overall lack of tools that monitor, record, and report events within a system.
10. Server-Side Request Forgery