ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

IT4409: Web Technologies and e-Services

Lec 14: Web Security

Outline

1. **What is web security?**
2. HTTPS
3. Session Management
4. Authentication
5. Common Web Attacks

What is web security?

- Website security is the act/practice of protecting websites from unauthorized access, use, modification, destruction, or disruption. (Mozilla)

- Effective website security requires design effort across the whole of the website:
  - Web application
  - Configuration of the web server
  - Policies for creating and renewing passwords
  - Client-side code.

Facts and Stats

- 95% of breached records came from only three industries in 2016
- There is a hacker attack every 39 seconds
- 43% of cyber attacks target small business
- The average cost of a data breach in 2020 will exceed $150 million
- In 2018 hackers stole half a billion personal records
- Over 75% of healthcare industry has been infected with malware over 2018
- Large-scale DDoS attacks increase in size by 500%

Facts and Stats

- Approximately $6 trillion is expected to be spent globally on cybersecurity by 2021
- By 2020 there will be roughly 200 billion connected devices
- Unfilled cybersecurity jobs worldwide will reach 3.5 million by 2021
- 95% of cybersecurity breaches are due to human error
- More than 77% of organizations do not have a Cyber Security Incident Response plan
- Most companies take nearly 6 months to detect a data breach, even major ones
- Share prices fall 7.27% on average after a breach
- Total cost for cybercrime committed globally has added up to over $1 trillion dollars in 2018

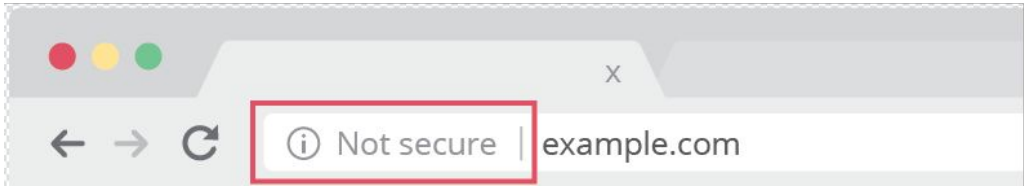
Outline

1. What is web security?
2. **HTTPS**
3. Session Management
4. Authentication
5. Common Web Attacks

HTTPS

- Hypertext transfer protocol secure (HTTPS) is the secure version of HTTP, which is the primary protocol used to send data between a web browser and a website.

- HTTPS is encrypted in order to increase security of data transfer.



- This is particularly important when users transmit sensitive data, such as by logging into a bank account, email service, or health insurance provider.

HTTPS

- HTTPS uses an encryption protocol to encrypt communications.

- The protocol is called Transport Layer Security (TLS), although formerly it was known as Secure Sockets Layer (SSL).

  - The private key - this key is controlled by the owner of a website and it's kept, as the reader may have speculated, private. This key lives on a web server and is used to decrypt information encrypted by the public key.

  - The public key - this key is available to everyone who wants to interact with the server in a way that's secure. Information that's encrypted by the public key can only be decrypted by the private key.

Outline

1. What is web security?
2. HTTPS
3. **Session Management**
4. Authentication
5. Common Web Attacks

Session Management

- A web session is a sequence of network HTTP request and response transactions associated to the same user.
- Modern and complex web applications require the retaining of information or status about each user for the duration of multiple requests.
- Therefore, sessions provide the ability to establish variables – such as access rights and localization settings – which will apply to each and every interaction a user has with the web application for the duration of the session.

Session Management

- Web applications can create sessions to keep track of anonymous users after the very first user request.

Session Management

- The disclosure, capture, prediction, brute force, or fixation of the session ID will lead to session hijacking (or sidejacking) attacks.
- An attacker is able to fully impersonate a victim user in the web application.
- Attackers can perform two types of session hijacking attacks, targeted or generic.

Outline

1. What is web security?
2. HTTPS
3. Session Management
4. **Authentication**
5. Common Web Attacks

Major security issues

- Prevent unauthorized users from accessing sensitive data
  - Authentication: identifying users to determine if they are one of the authorized ones
  - Access control: identifying which resources need protection and who should have access to them
- Prevent attackers from stealing data from network during transmission
  - Encryption (usually by Secure Sockets Layer)

Authentication

- Collect user ID information from end users ("logging in")
  - usually by means of browser dialog / interface
  - user ID information normally refers to username and password
- Transport collected user ID information to the web server
  - unsecurely (HTTP) or securely (HTTPS = HTTP over SSL)
- Verify ID and passwd with backend Realms ("security database")
  - Realms maintain username, password, roles, etc., and can be organized by means of LDAP, RDBMS, Flat-file, etc.
  - Validation: the web server checks if the collected user ID & passwd match with these in the realms.
- Keep track of previously authenticated users for further HTTP operations

WWW-Authenticate

- The authentication request received by the browser will look something like:
  - WWW-Authenticate = Basic realm="defaultRealm"
    - Basic indicates the HTTP Basic authentication is requested
    - realm indicates the context of the login
      - realms hold all of the parts of security puzzle
        - Users
        - Groups
        - ACLs (Access Control Lists)
- Basic Authentication
  - userid and password are sent base 64 encoded (might as well be plain text)
  - hacker doesn't even need to unencode all he has to do is "replay" the blob of information he stole over and over (this is called a "replay attack")

WWW-Authenticate

- Digest Authentication
  - attempts to overcome the shortcomings of Basic Authentication
  - WWW-Authenticate = Digest realm="defaultRealm" nonce="Server SpecificString"
  - see RFC 2069 for description of nonce, each nonce is different
  - the nonce is used in the browser in a 1-way function (MD5, SHA-1….) to encode the userid and password for the server, this function essentially makes the password good for only one time
- Common browsers don't use Digest Authentication but an applet could as an applet has access to all of the Java Encryption classes needed to create the creation of a Digest.

Outline

1. What is web security?
2. HTTPS
3. Session Management
4. Authentication
5. **Common Web Attacks**

Common Web Attacks

Client side

- XSS
- CSRF

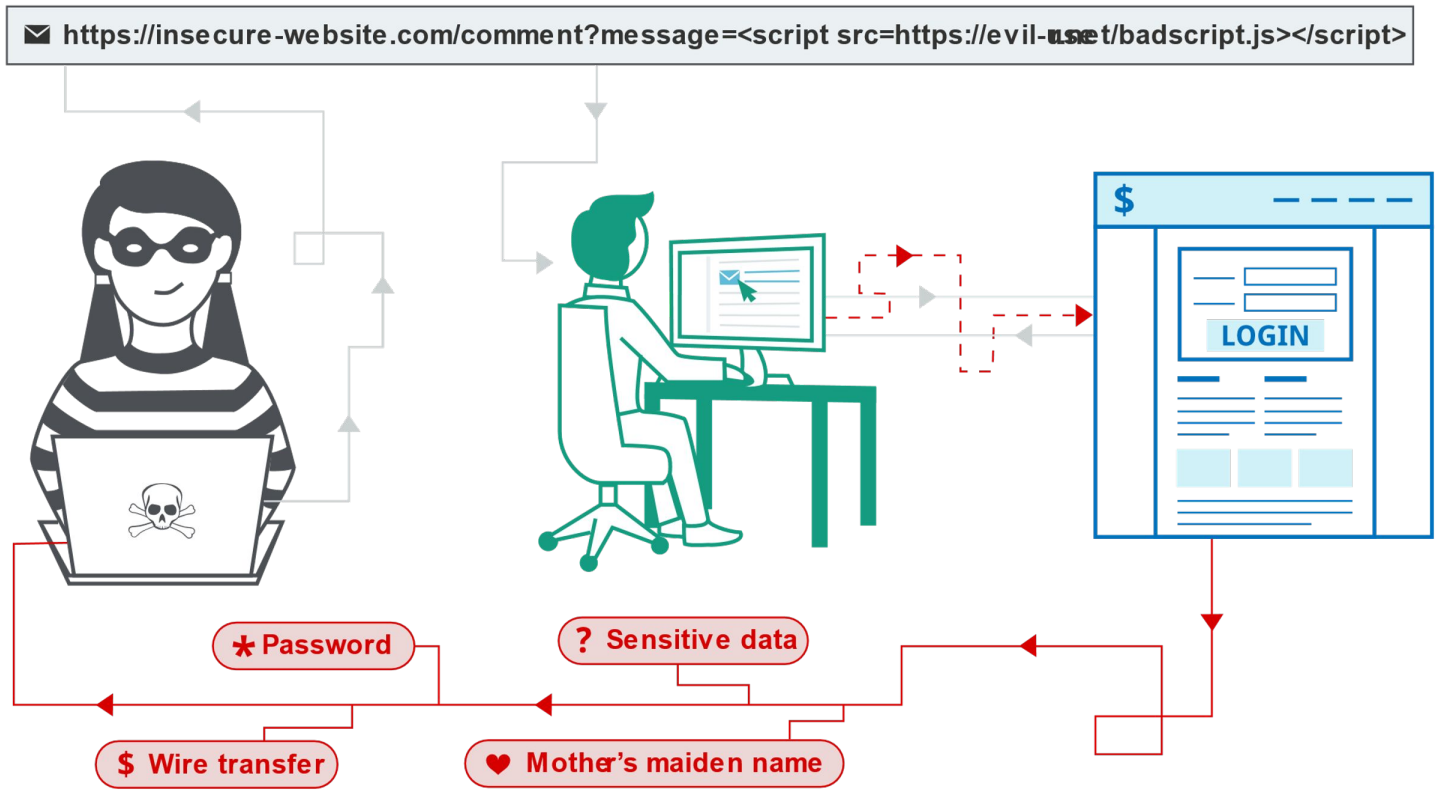
Server side

- SQLi
- Brute-force
- File upload
- Command injection

Cross-Site Scripting - XSS

- Cross-site scripting (XSS) is a security exploit which allows an attacker to inject into a website malicious client-side code.
- This code is executed by the victims and lets the attackers bypass access controls and impersonate users.
- XSS was the seventh most common Web app vulnerability in 2017 - OWASP

Cross-Site Scripting - XSS



Cross-Site Scripting - XSS

- There are three main types of XSS attacks. These are:

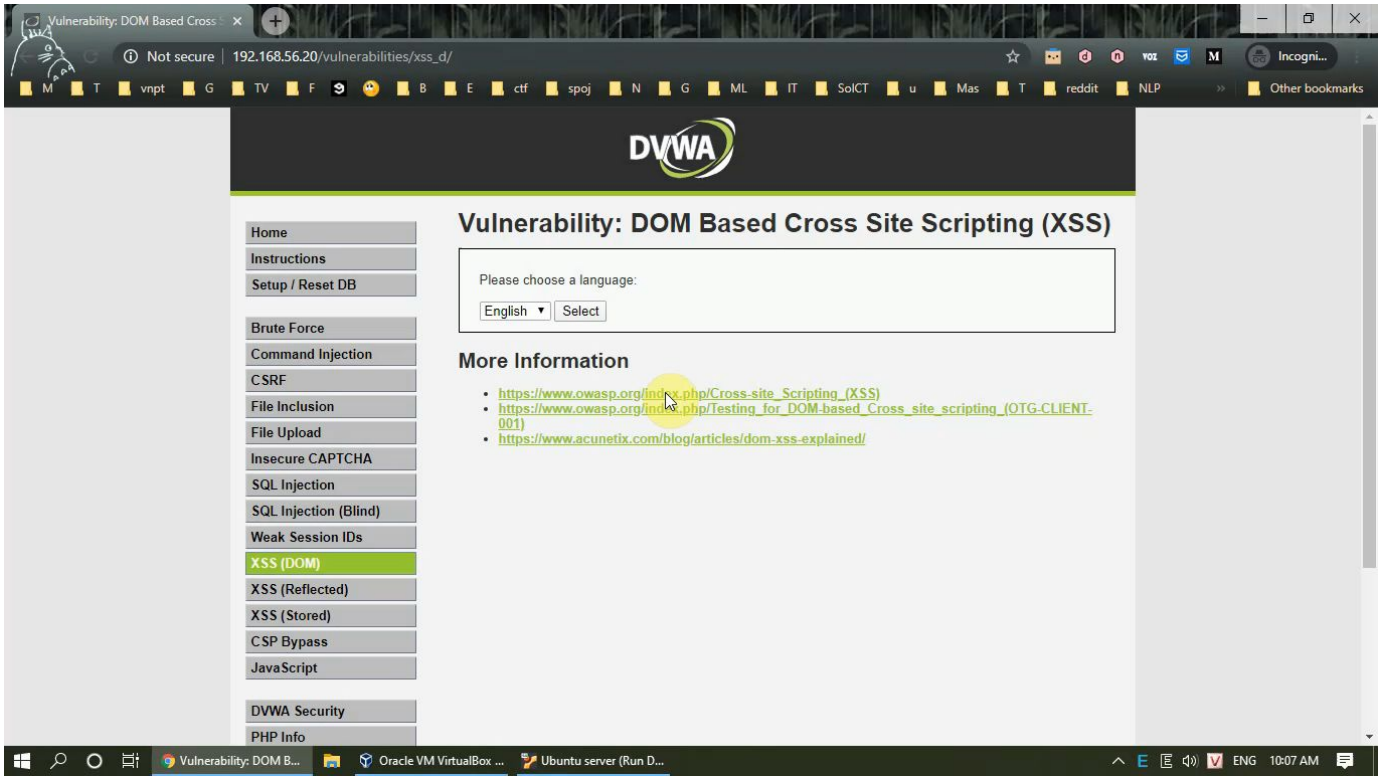
- Reflected XSS, where the malicious script comes from the current HTTP request.
- Stored XSS, where the malicious script comes from the website's database.
- DOM-based XSS, where the vulnerability exists in client-side code rather than server-side code.

Cross-Site Scripting - XSS

How to prevent XSS attacks

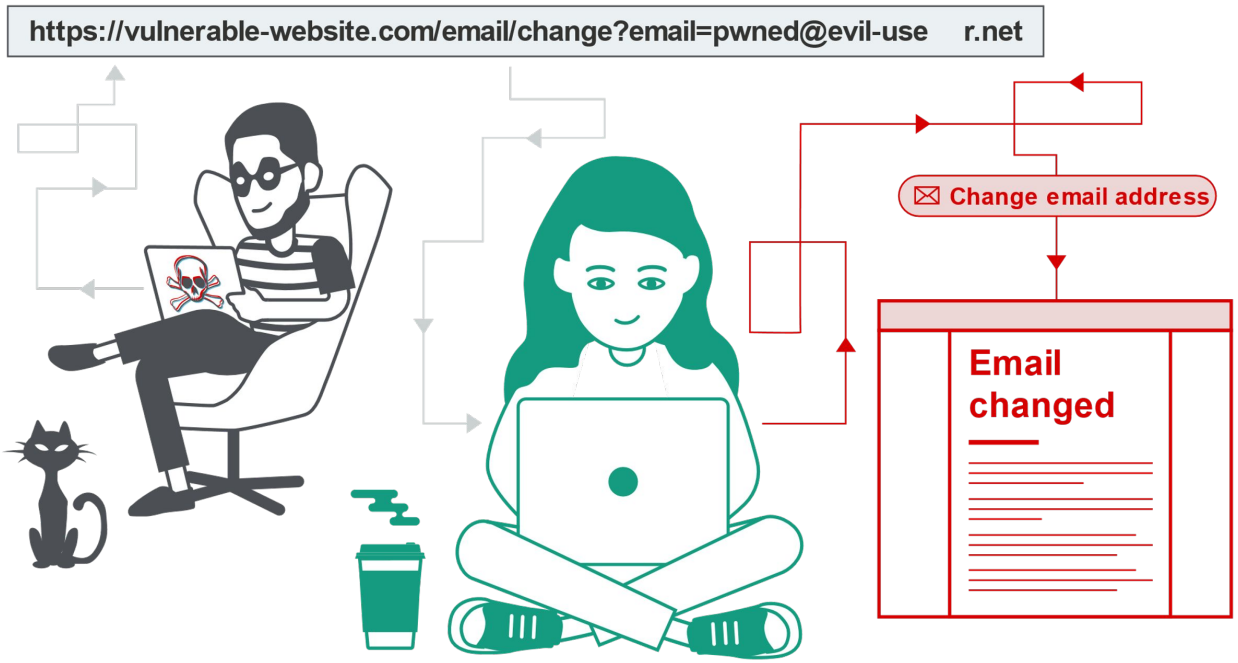
- Filter input on arrival
- Encode data on output
- Use appropriate response headers
- Content Security Policy

Cross-Site Scripting - XSS



Cross-Site Request Forgery - CSRF

- Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated.



Cross-Site Request Forgery - CSRF

- Preventing CSRF attacks:
  - Include a CSRF token within relevant requests

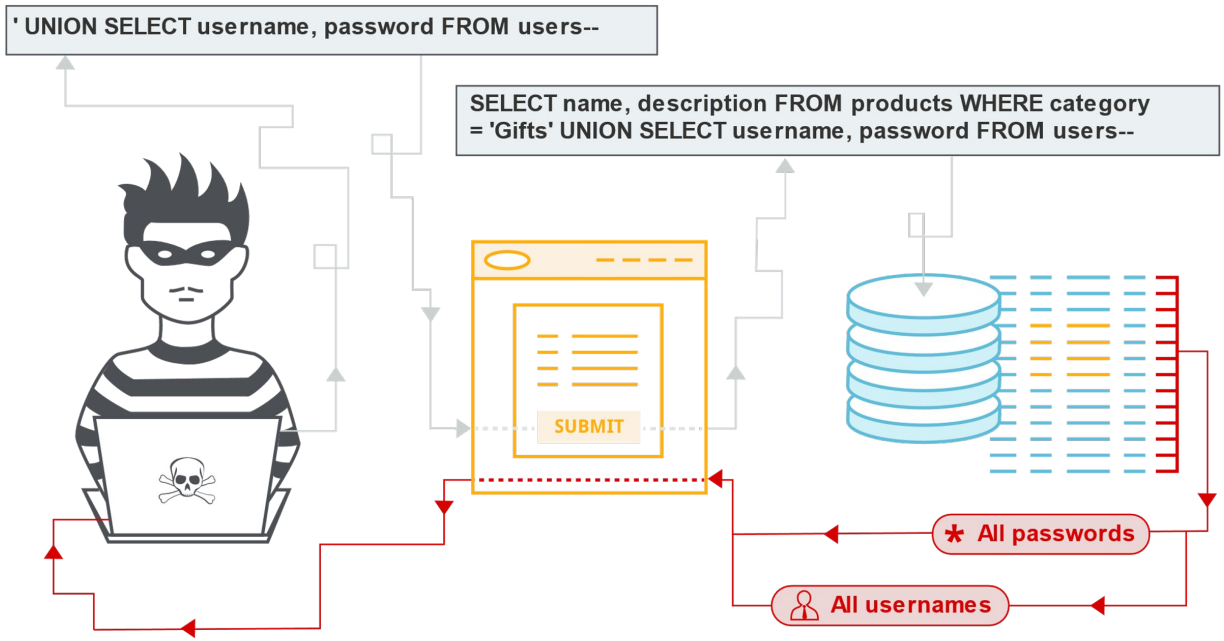
- The token should be:
  - Unpredictable with high entropy, as for session tokens in general.
  - Tied to the user's session.
  - Strictly validated in every case before the relevant action is executed.

SQL Injection

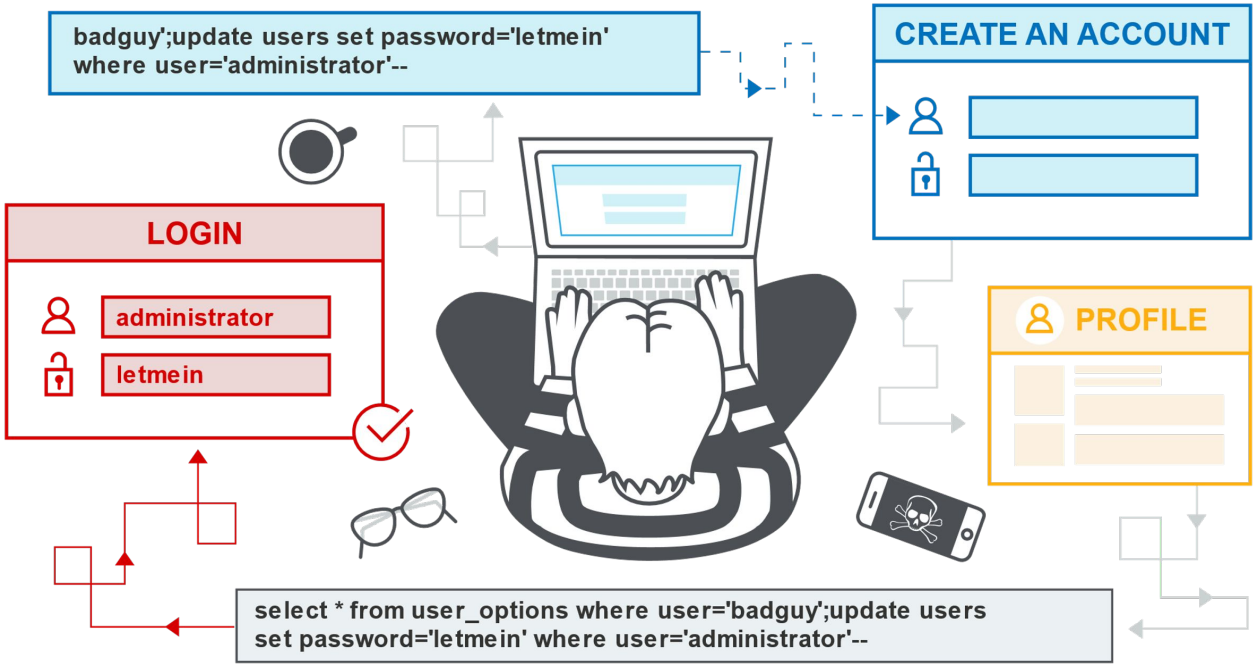
- A SQL injection attack consists of insertion or “injection” of a SQL query via the input data from the client to the application.

- SQL injection vulnerabilities enable malicious users to execute arbitrary SQL code on a database, allowing data to be accessed, modified, or deleted irrespective of the user's permissions.

SQL Injection



SQL Injection



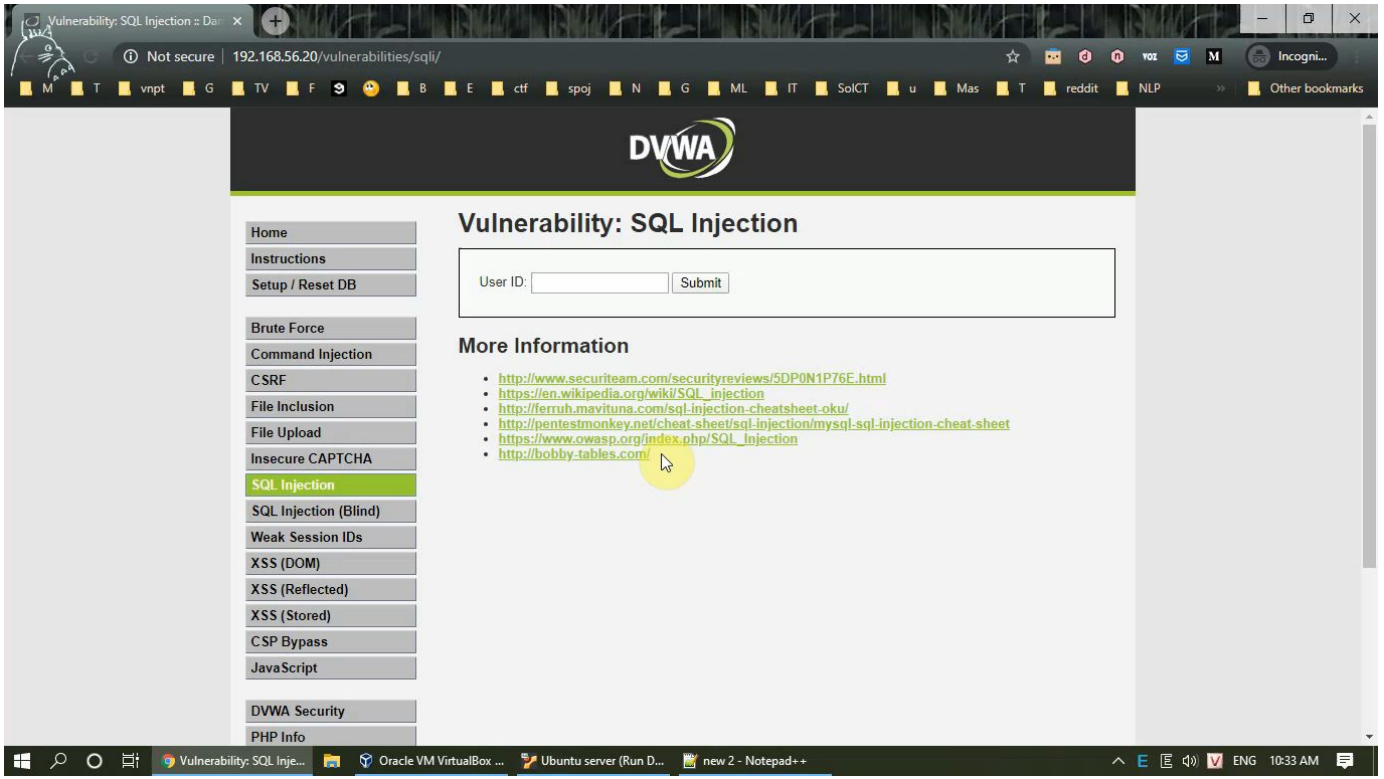
SQL Injection

- How to prevent: Using parameterized queries (also known as prepared statements) instead of string concatenation within the query.

- Before:
  - String query = "SELECT * FROM products WHERE category = '"+ input + "'";
  - Statement statement = connection.createStatement();
  - ResultSet resultSet = statement.executeQuery(query);

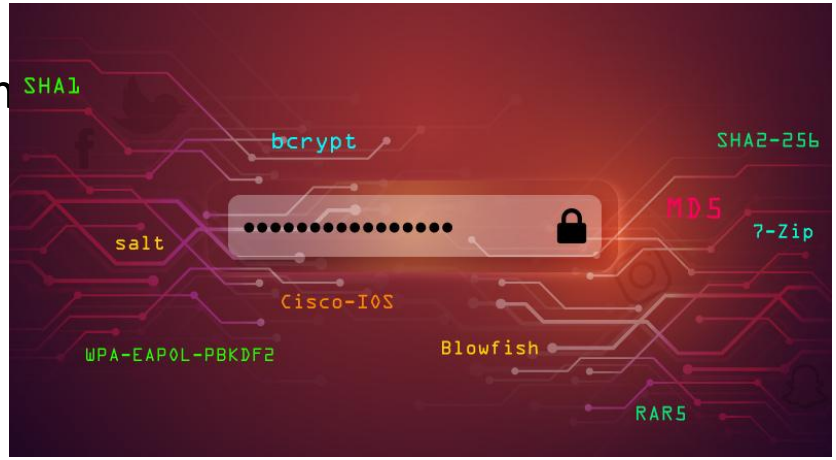
- After:
  - PreparedStatement statement = connection.prepareStatement("SELECT * FROM products WHERE category = ?");
  - statement.setString(1, input);
  - ResultSet resultSet = statement.executeQuery();

SQL Injection



Brute force

- A brute force attack, also known as an exhaustive search, is a cryptographic hack that relies on guessing possible combinations of a targeted password until the correct password is discovered.
  - Combination of letters and num
  - Use a dictionary



- Prevent password cracking:
  - Long and complex password
  - Account lock out

Brute force

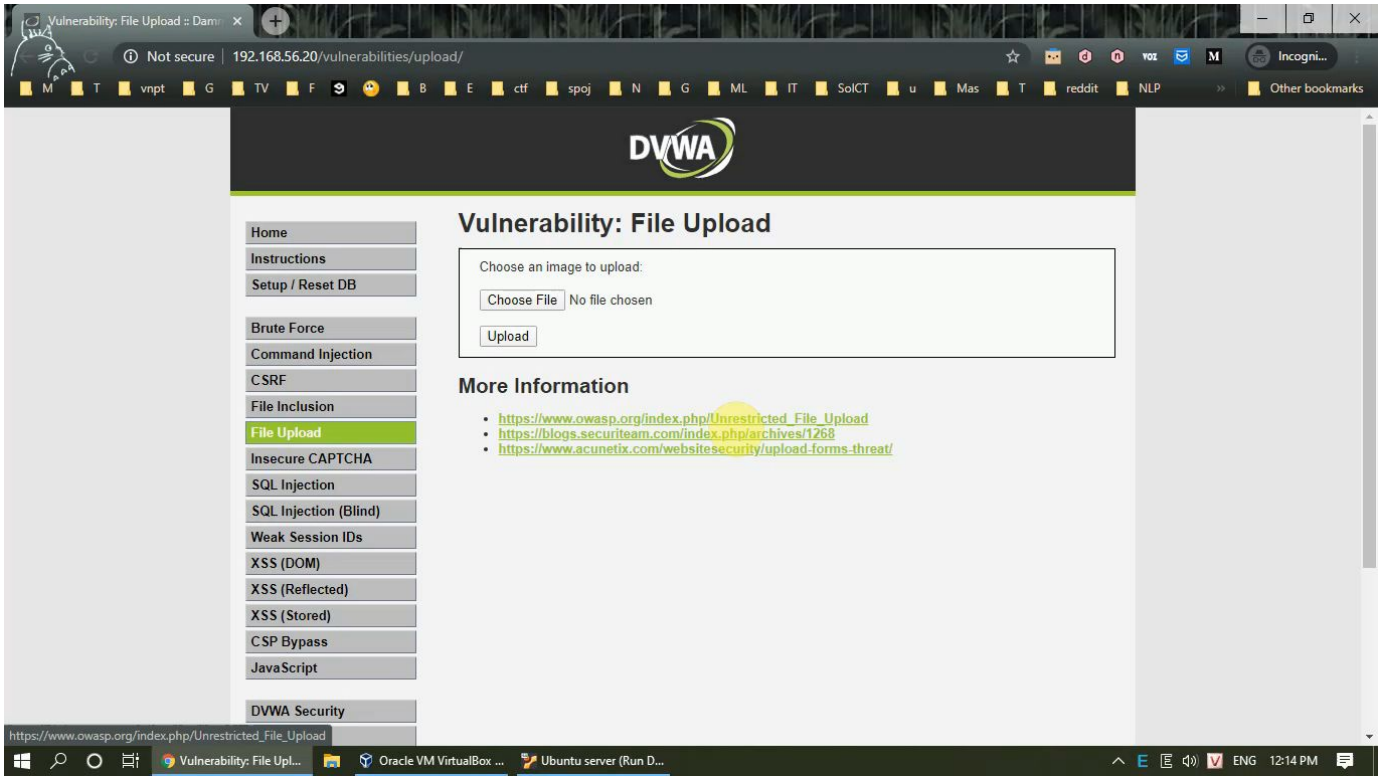
File upload

- Uploaded files represent a significant risk to applications.
- The first step in many attacks is to get some code to the system to be attacked.
- Then the attack only needs to find a way to get the code executed.

File upload

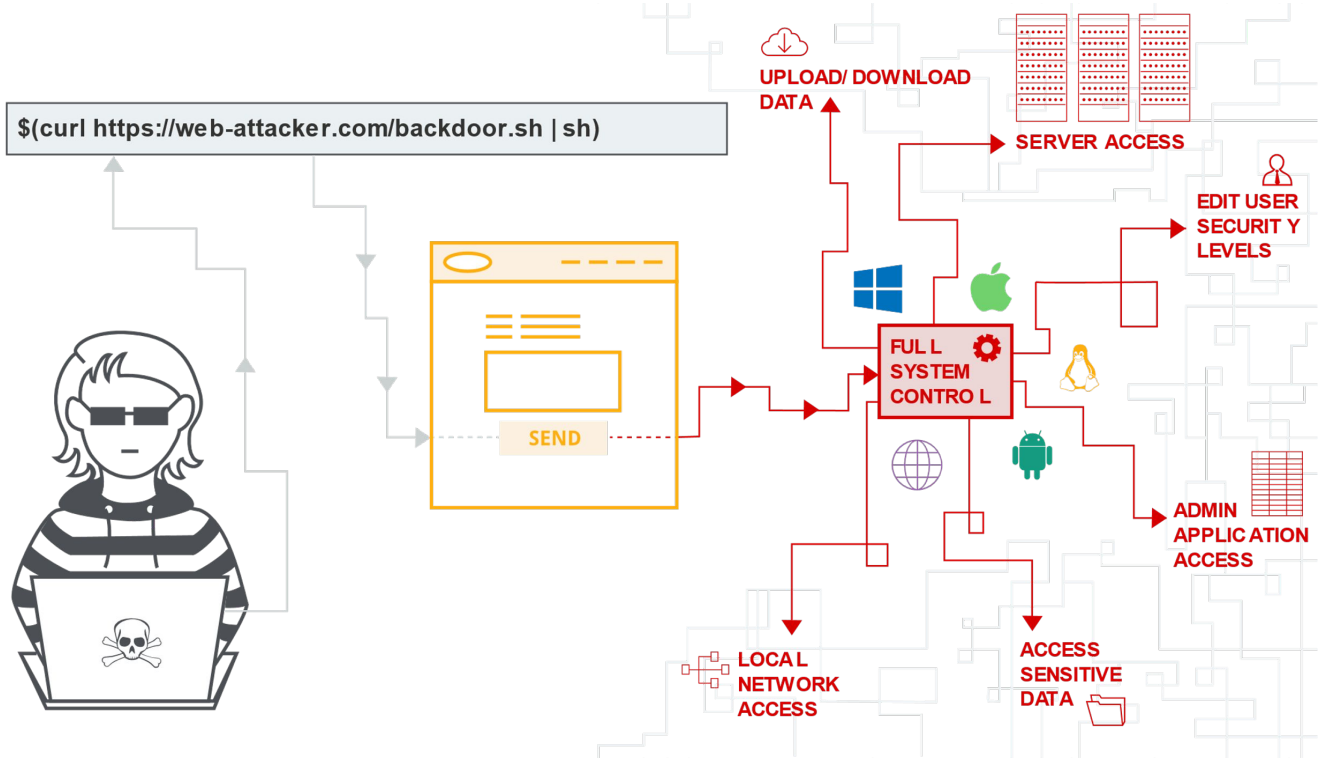
- Prevention Methods:
  - The file types allowed to be uploaded should be restricted to only those that are necessary for business functionality.
  - Never accept a filename and its extension directly without having a whitelist filter.
  - The application should perform filtering and content checking on any files which are uploaded to the server.
  - It is necessary to have a list of only permitted extensions on the web application.
  - All the control characters and Unicode ones should be removed from the filenames and their extensions without any exception.
  - Limit the filename length.
  - Uploaded directory should not have any “execute” permission and all the script handlers should be removed from these directories.
  - Limit the file size to a maximum value in order to prevent denial of service attacks.
  - The minimum size of files should be considered.
  - Use Cross Site Request Forgery protection methods.

File upload


Vulnerability: File Upload :: Dam
Not secure | 192.168.56.20/vulnerabilities/upload/
Incogni...
Other bookmarks
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
DVWA Security
Vulnerability: File Upload
Choose an image to upload:
Choose File
No file chosen
Upload
More Information
https://www.owasp.org/index.php/Unrestricted_File_Upload
https://blogs.securiteam.com/index.php/archives/1268
https://www.acunetix.com/websitesecurity/upload-forms-threat/
https://www.owasp.org/index.php/Unrestricted_File_Upload
Vulnerability: File Upl...
Oracle VM VirtualBox ...
Ubuntu server (Run D...
ENG
12:14 PM

Command Injection

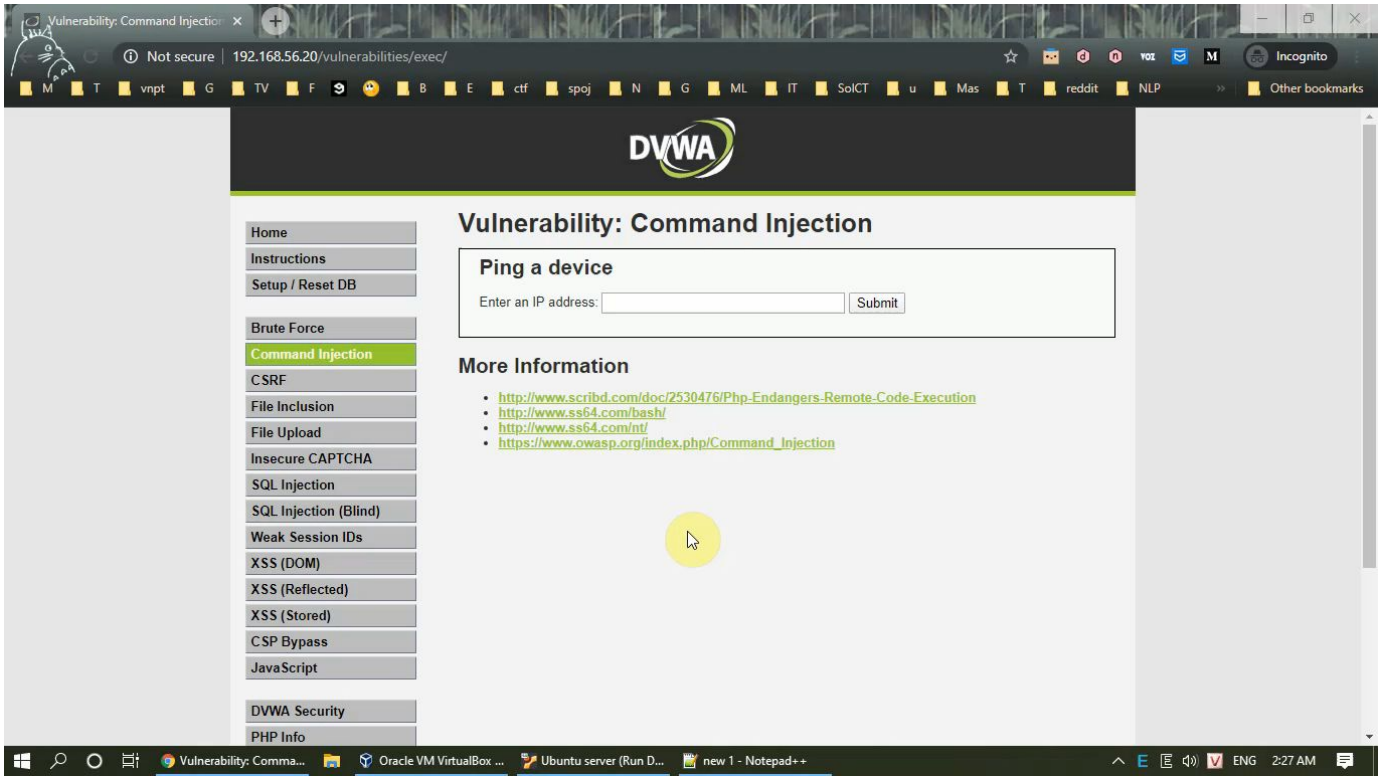
- Command injection is an attack in which the goal is execution of arbitrary commands on the host operating system via a vulnerable application.



Command Injection

- How to prevent:
  - Validating against a whitelist of permitted values.
  - Validating that the input is a number.
  - Validating that the input contains only alphanumeric characters, no other syntax or whitespace.

Command Injection



ĐẠI HỌC BÁCH KHOA

25 YEARS ANNIVERSARY SOICT

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you for your attentions!

soict.hust.edu.vn/ fb.com/groups/soict