

COS10011/60004  
Creating Web Applications

## Web Security

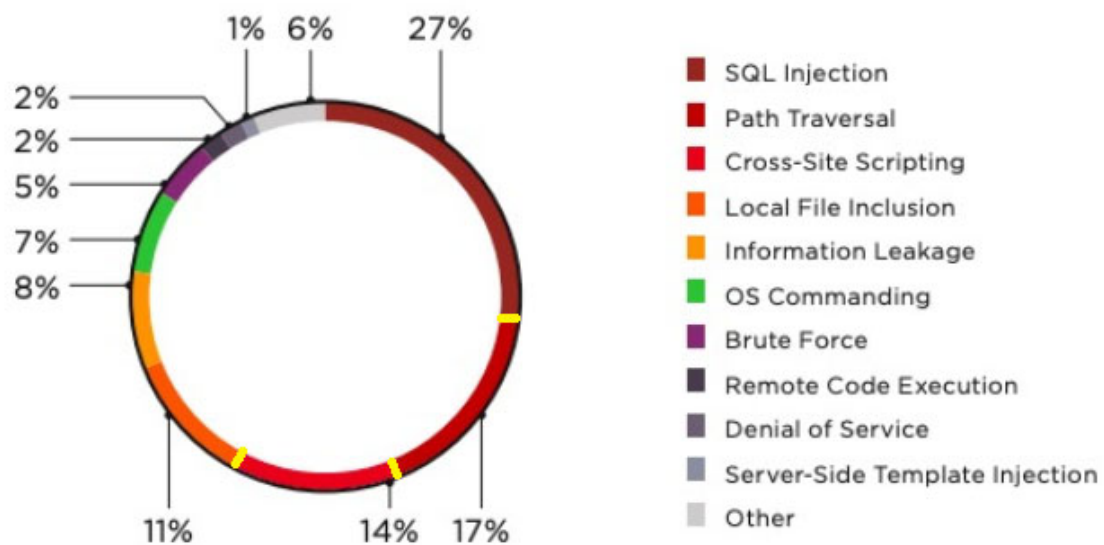


© Sw

## Fundamental problem

- The Internet was not designed with security as a primary goal
  1. **Everything** is on the Internet
  2. Everyone can **access** the Internet
  3. We can **upload stuff** onto the Internet
  4. Web/software developers are **not good at testing**
- Many ways to attack and/or take control of a Web site

# Top 10 web application attacks



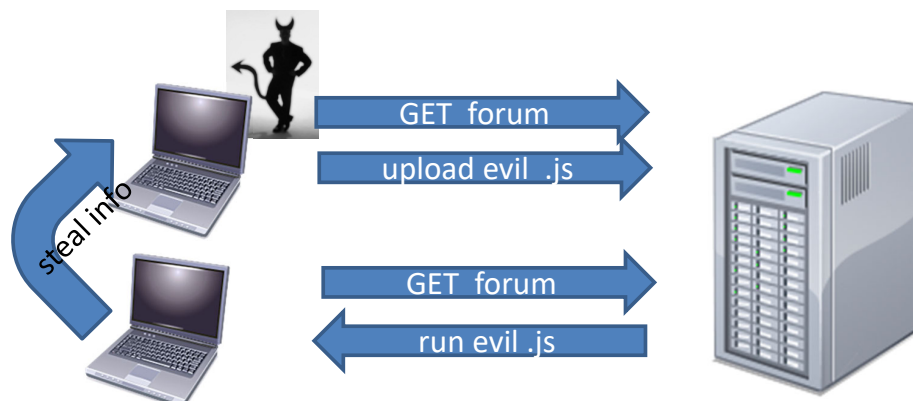
Reference: Attacks on web applications: 2018 in review

<https://www.ptsecurity.com/ww-en/analytics/web-application-attacks-2019/>

© Swinburne University of Technology

## Cross-site scripting (XSS)

- Attacker uploads **evil .js** to innocent server (forum/blog/twitter/whatever). It is stored (in db).
- Victim goes to web page on server. receives **.js**
- Script **steals** info from victim and sends it to Attacker



# SQL injection

- SQL injection is a code injection technique, in which malicious SQL statements are inserted into an entry field for execution.
- SQL injection attacks allow attackers to
  - dump the database contents to the attacker,
  - modify the data,
  - become administrators of the database server,
  - ...

## SQL injection

1) Attacker injects malicious SQL query via input data.



GET/POST + evil SQL



Web Server

2) Malicious SQL query is executed by the database.

run evil SQL



Database

private data

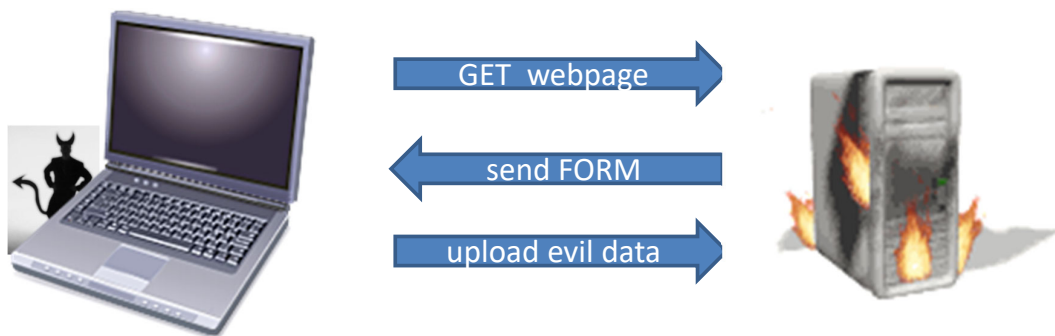
4) Attacker is granted access potentially act as database administrator.

3) DBMS returns unauthorised (**private**) data to attacker.

private data

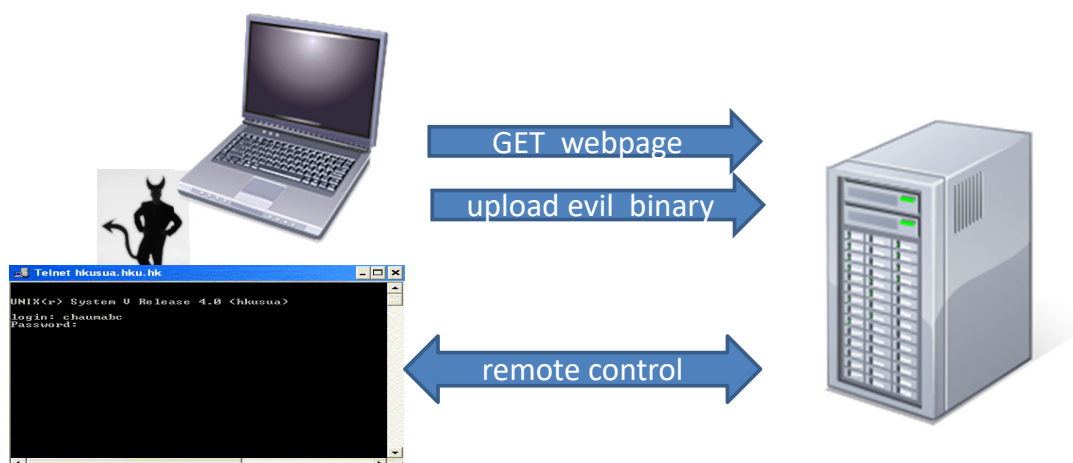
# Denial of Service

- Attackers flood the server with excessive requests and data.
- Some services or resources become unavailable to its intended users.



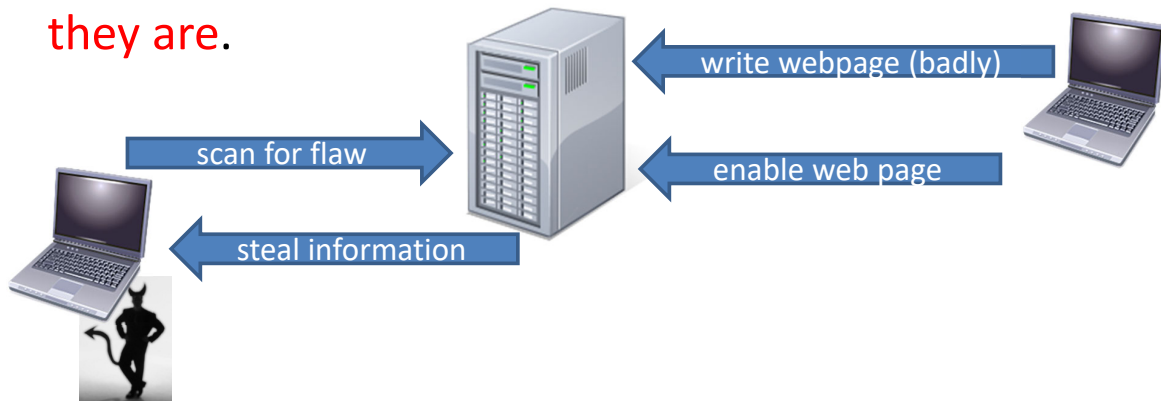
## passthrough functions

- Write (upload) **executable** to the web server.
- Run them.
- Gain **remote access**



# Directory traversal, .htaccess

- Web developer can't get script to work.  
Turns off **security**.
- Gets it working, puts on Internet
- Attacker **scans** servers, finds flaw
- Loads pages which are "not" on the internet – but **they are**.



## Fundamental solutions

1. **Don't** let people upload *any old thing* into your web sites
  - **Filter, Sanitise, Validate**
2. Don't **trust** the software you use
  - It will have **bugs**/vulnerabilities
3. Web development does not stop when you deliver the product
  - **Test, Monitor, Patch**

# Solutions

- Lock down forms - limit **length**
  - js to validate? But you can turn it off.
- **Filter/validate/sanitise** on the server:
  - php scripts
  - db functions
- Limit **permissions** on server
  - read-only, Apache user owns files
  - have a back-up
  - configure web server carefully.
- Web development does not stop when you deliver the product
  - **Test, Monitor, Patch**