Overview of Web Application Security

Web Application

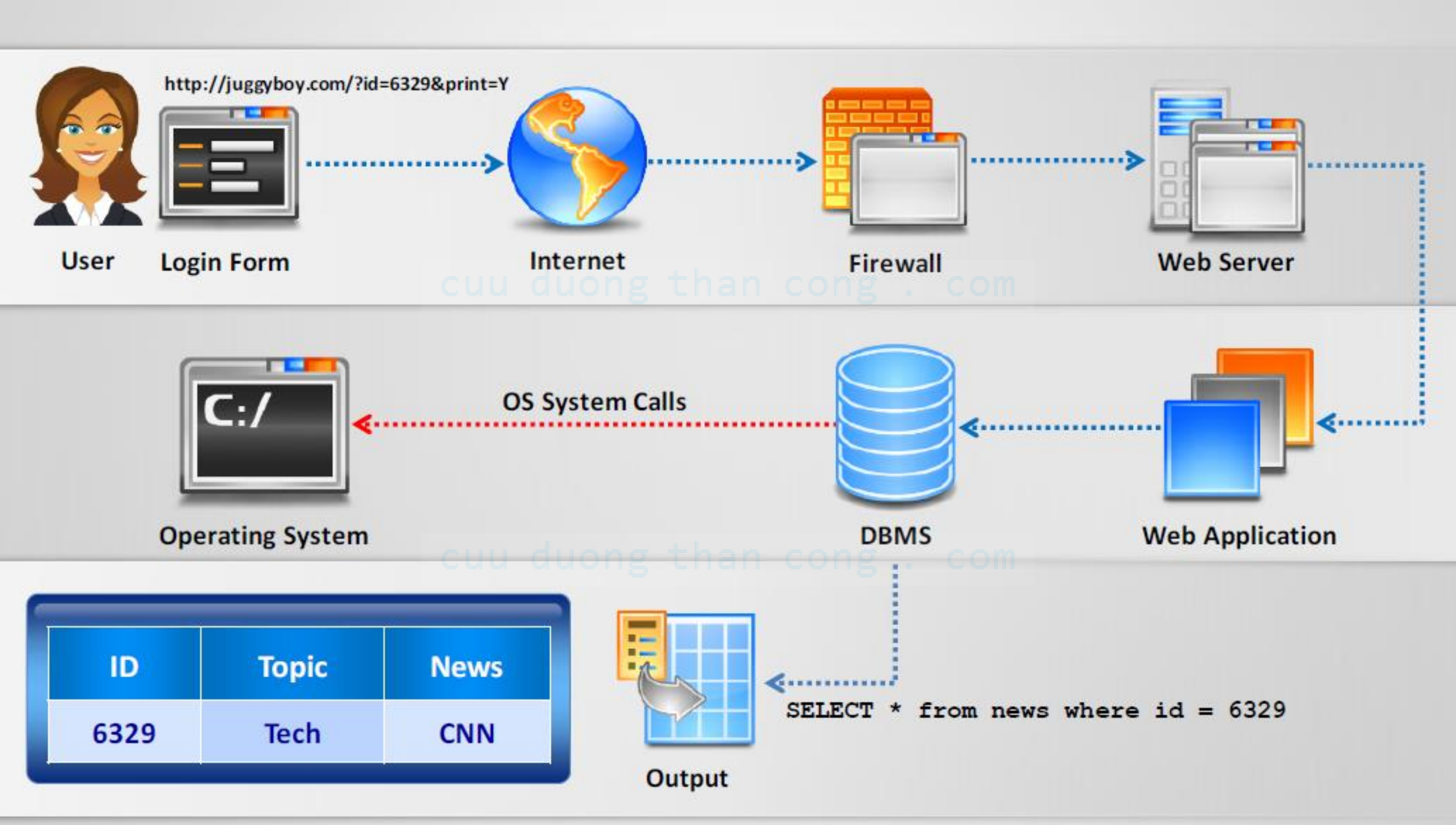
Web applications **provide an interface between end users and web servers** through a set of web pages that are generated at the server end or contain script code to be executed dynamically within the client web browser

Though web applications enforce certain security policies, they are **vulnerable to various attacks** such as SQL injection, cross-site scripting, session hijacking, etc.

Web technologies such as **Web 2.0** provide more attack surface for web application exploitation

Web applications and Web 2.0 technologies are invariably used to support **critical business functions** such as CRM, SCM, etc. and improve business efficiency

How Web Applications Work

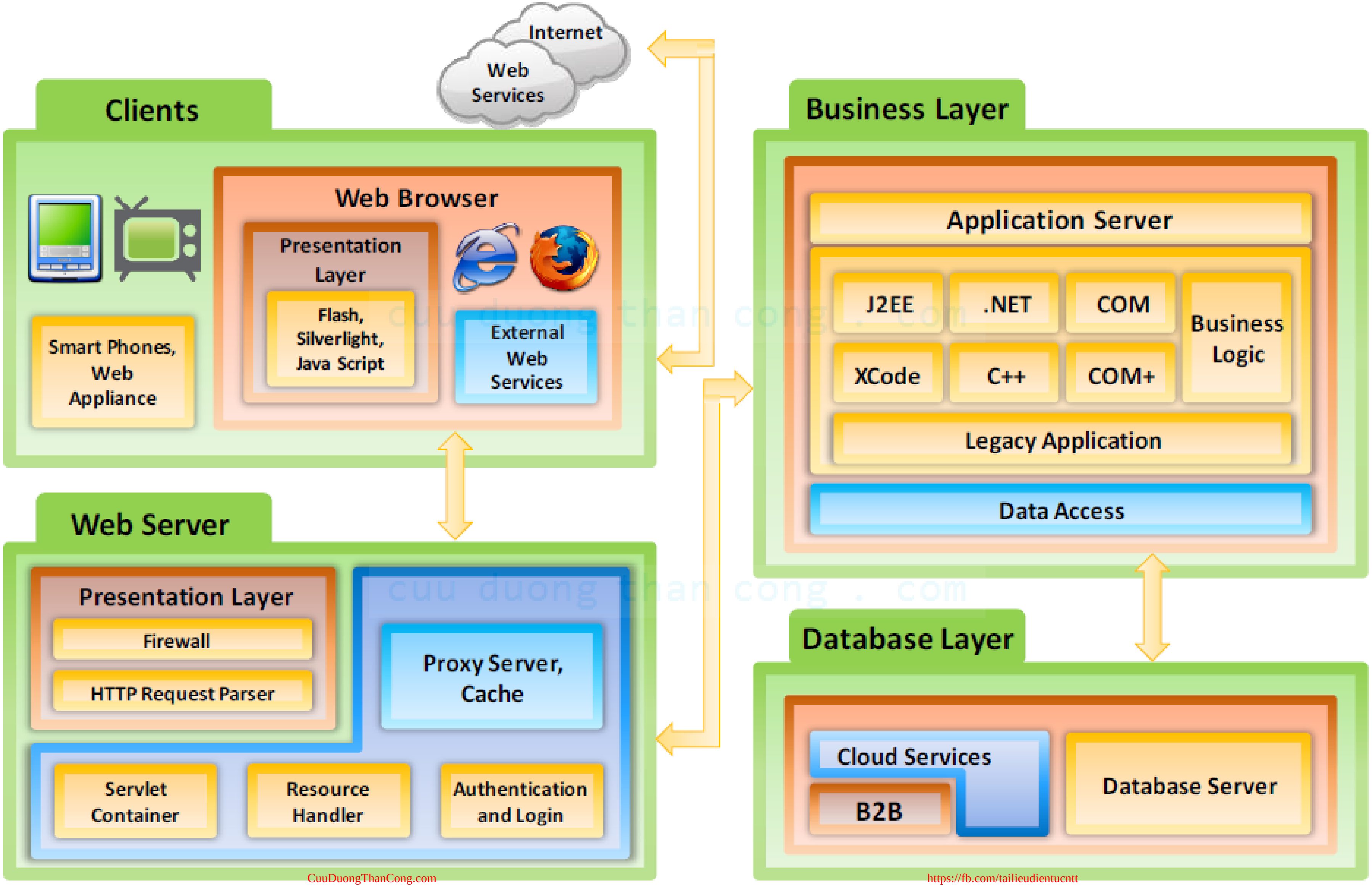


http://juggyboy.com/?id=6329&print=Y

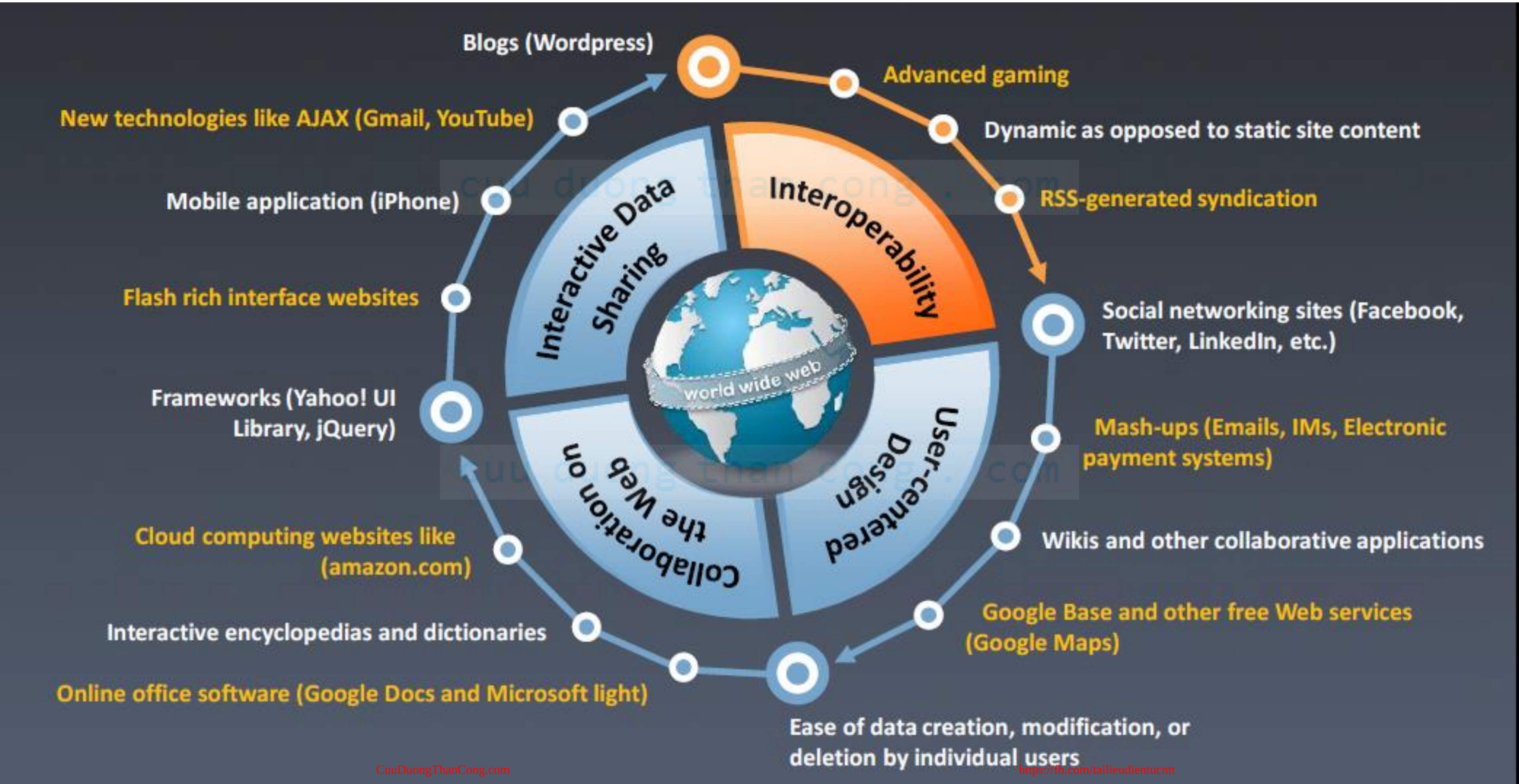
```
SELECT * from news where id = 6329
```

ID	Topic	News
6329	Tech	CNN

Web Application Architecture



Web 2.0



Blogs (Wordpress)

Advanced gaming

Dynamic as opposed to static site content

RSS-generated syndication

Social networking sites (Facebook, Twitter, LinkedIn, etc.)

Mash-ups (Emails, IMs, Electronic payment systems)

Wikis and other collaborative applications

Google Base and other free Web services (Google Maps)

Ease of data creation, modification, or deletion by individual users

Online office software (Google Docs and Microsoft light)

Interactive encyclopedias and dictionaries

Cloud computing websites like (amazon.com)

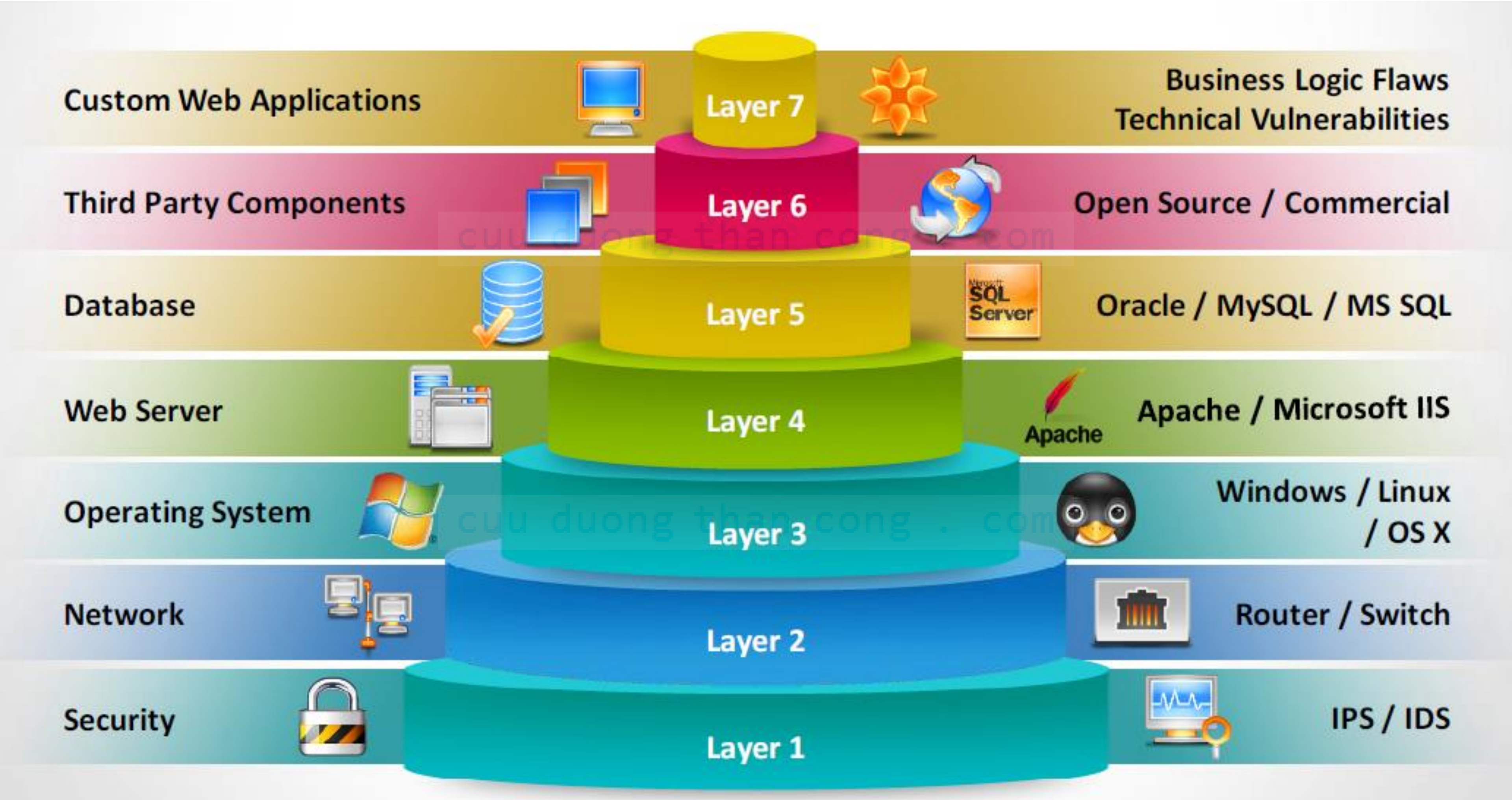
Frameworks (Yahoo! UI Library, jQuery)

Flash rich interface websites

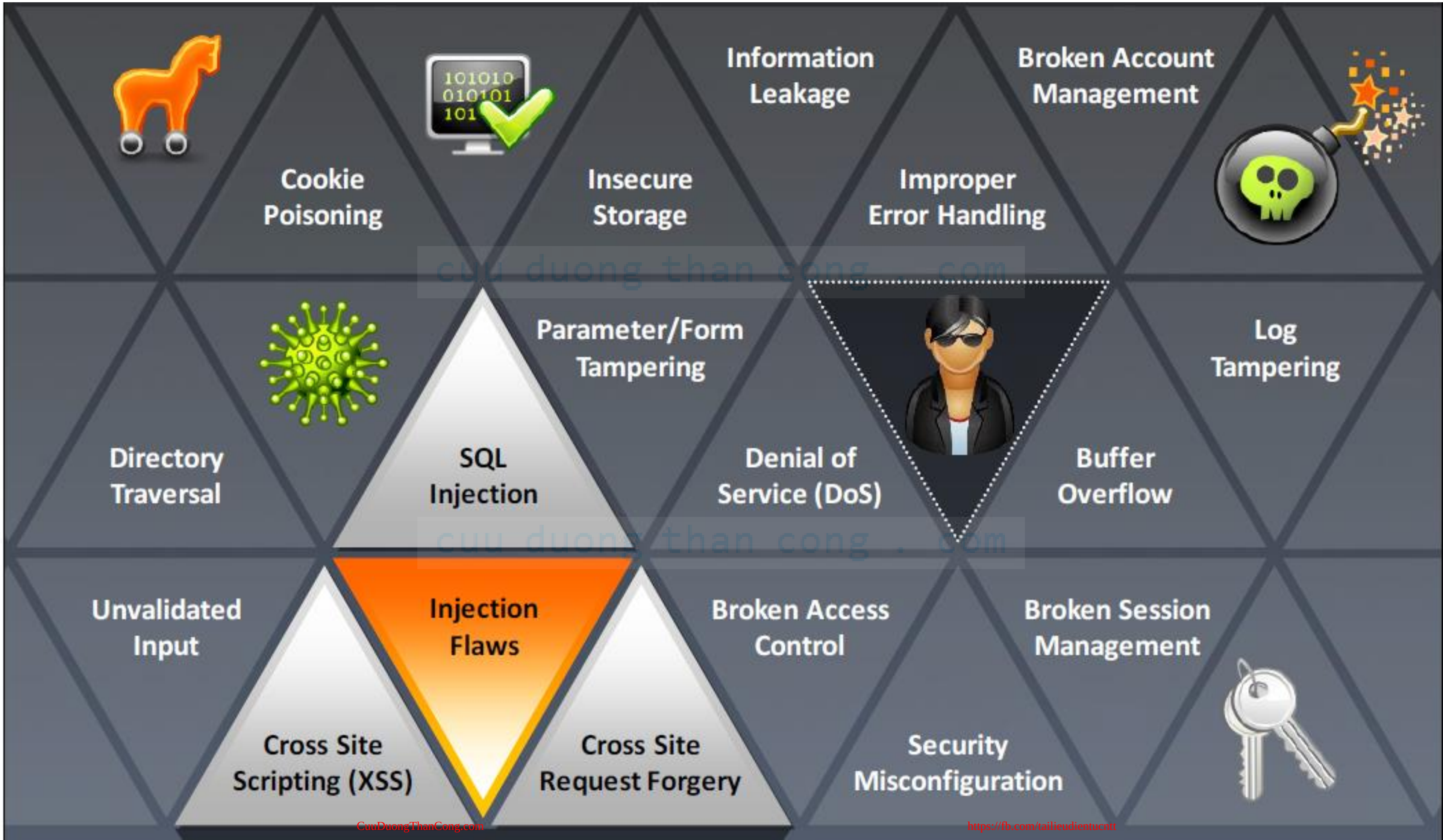
Mobile application (iPhone)

New technologies like AJAX (Gmail, YouTube)

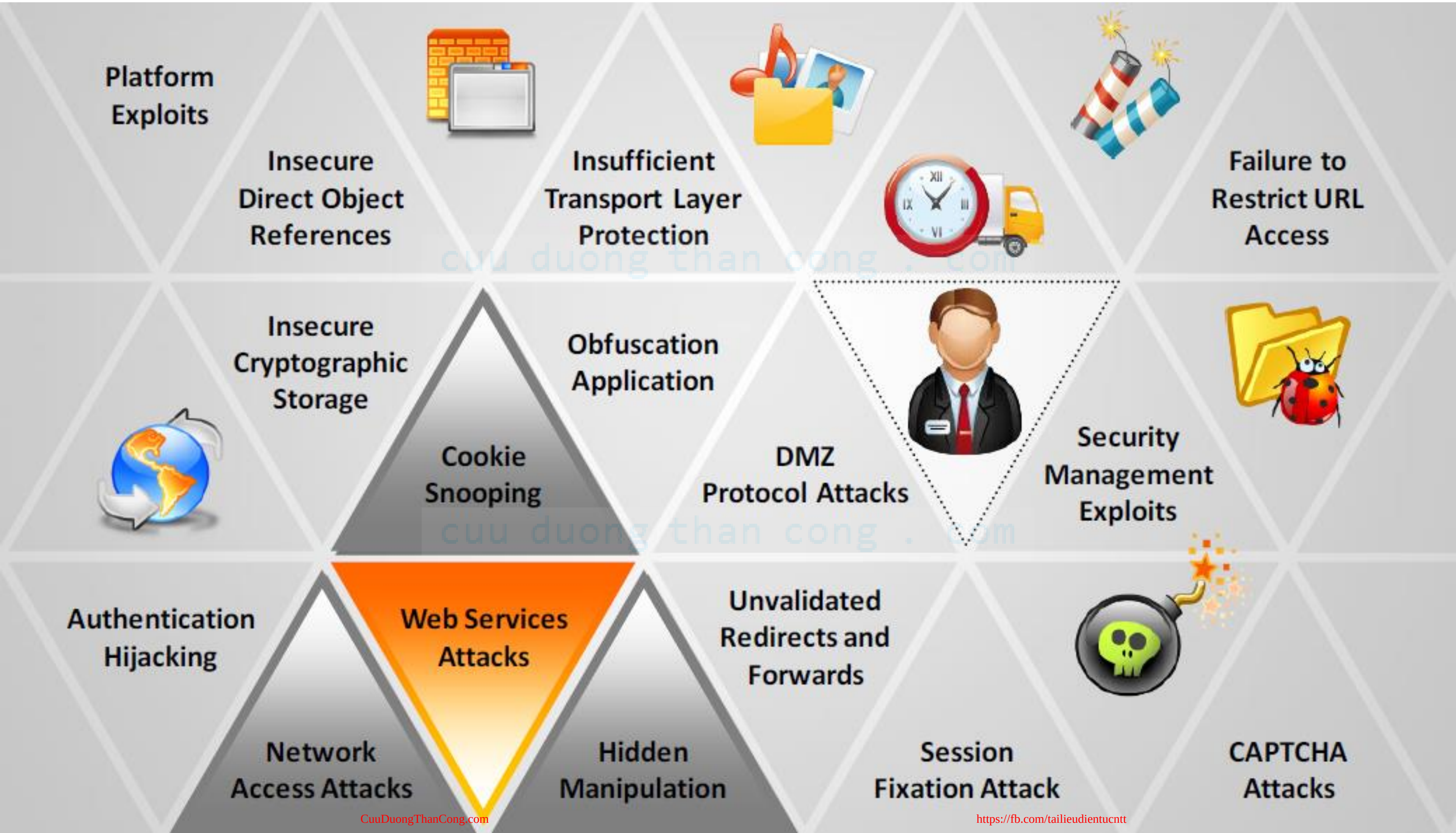
Vulnerability Stack



Web Application Threats



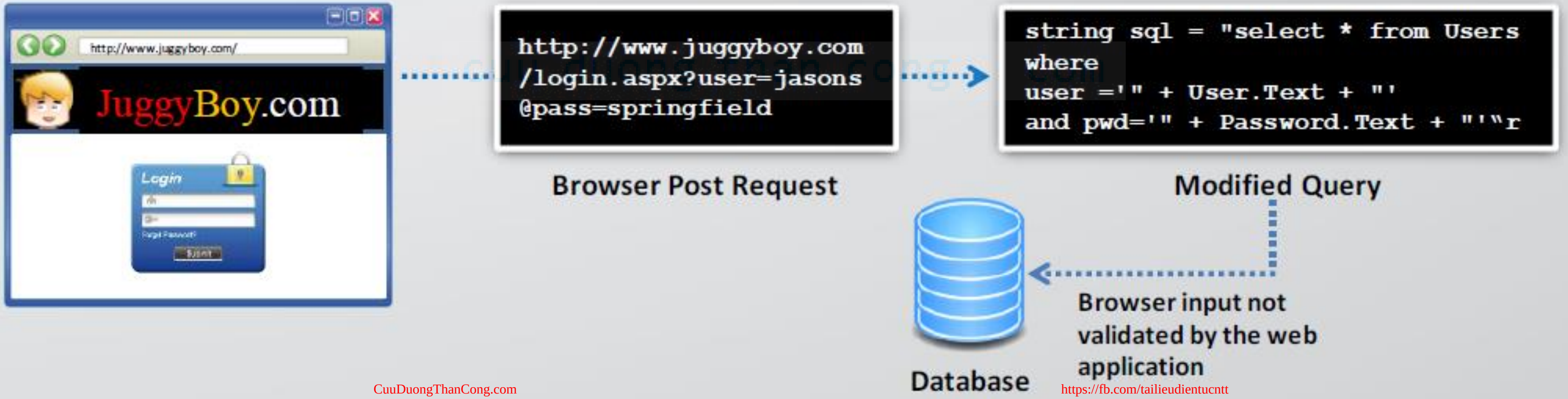
Web Application Threat



Unvalidated Input

Input validation flaws refers to a web application vulnerability where **input from a client is not validated** before being processed by web applications and backend servers

An attacker exploits input validation flaws to perform cross-site scripting, buffer overflow, injection attacks, etc. that result in **data theft and system malfunctioning**



```
http://www.juggyboy.com
/login.aspx?user=jasons
@pass=springfield
```

Browser Post Request

```
string sql = "select * from Users
where
user ='" + User.Text + "'
and pwd='" + Password.Text + "'"r
```

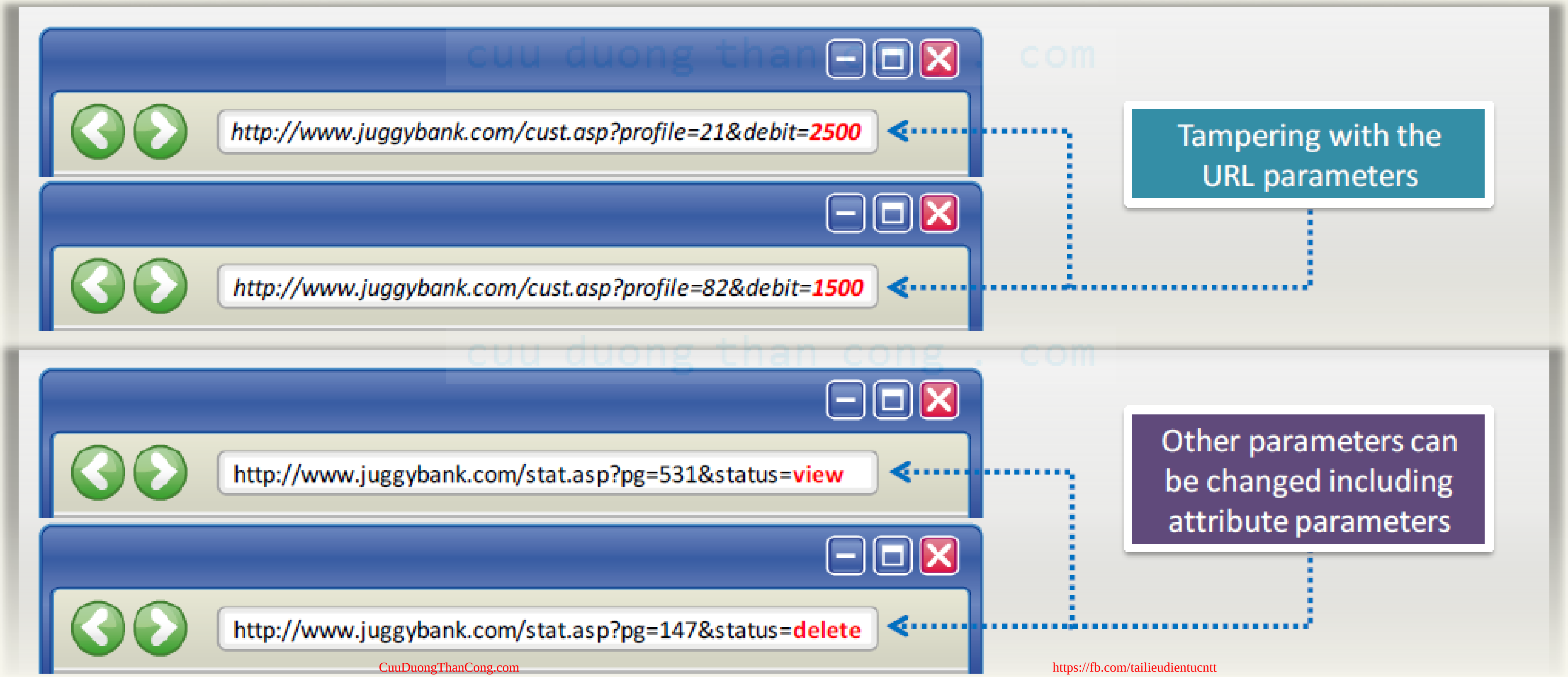
Modified Query

Browser input not validated by the web application

Database

Parameter/Form Tampering

- A web parameter tampering attack involves the **manipulation of parameters exchanged** between client and server in order to modify application data such as user credentials and permissions, price, and quantity of products
- A parameter tampering attack **exploits vulnerabilities** in integrity and logic validation mechanisms that may result in XSS, SQL injection, etc.



http://www.juggybank.com/cust.asp?profile=21&debit=2500

http://www.juggybank.com/cust.asp?profile=82&debit=1500

Tampering with the URL parameters

http://www.juggybank.com/stat.asp?pg=531&status=view

http://www.juggybank.com/stat.asp?pg=147&status=delete

Other parameters can be changed including attribute parameters

What is the OWASP Top Ten?

"The Ten Most Critical Web Application Security Risks"

The OWASP Top Ten is a prioritized list of the top ten most critical web application security *risks*.

- It's an awareness tool, ***not*** a standard
- Released in 2003, 2004, 2007, 2010, 2013, 2017
- Developed using the OWASP Risk Rating Methodology [1]
- 2017 rc1 based on results from a 2016 open data call
- It's about risks, not (just) about vulnerabilities
- Not intended to be "airtight, non-overlapping, or a strict taxonomy"
- Constantly changing….

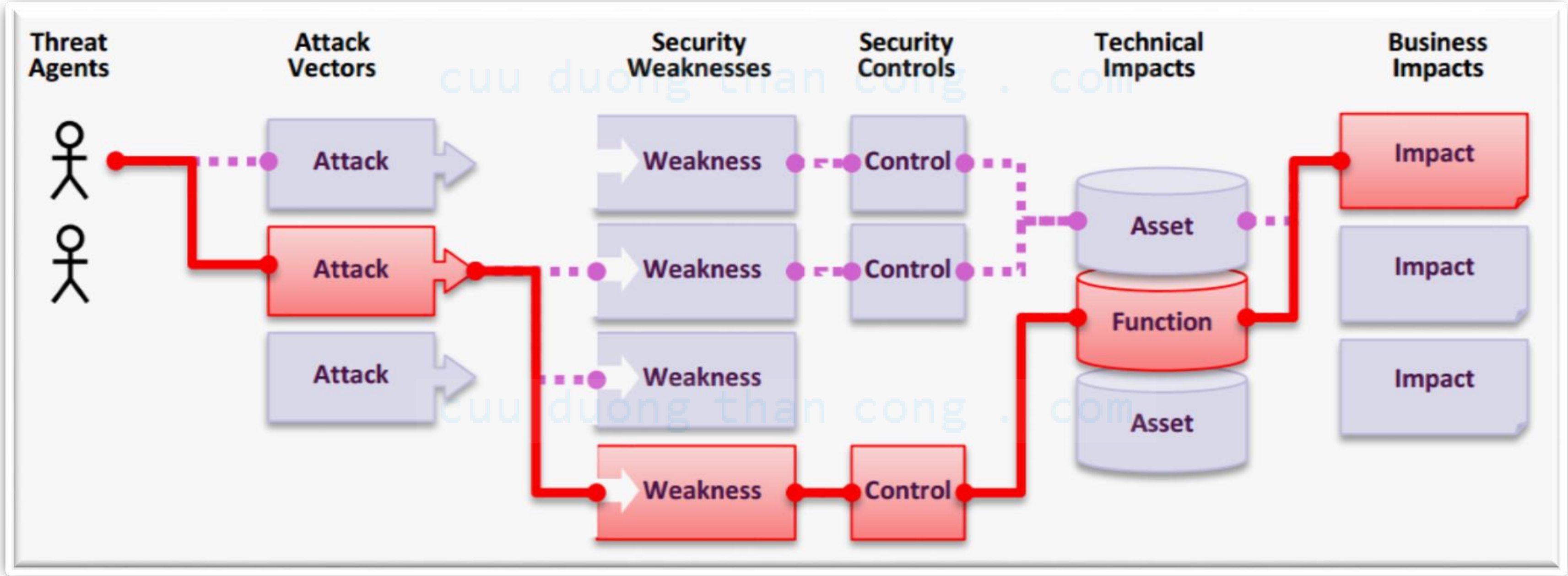
1. https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology

What the OWASP Top Ten Isn’t

- It is *not* a standard
- It doesn’t stop at 10
  - Clickjacking
  - Denial of Service
  - Deserialization of Untrusted Data
  - Expression Language Injection
  - Many, many more!
- Risk is environment-specific
  - Your Top Ten risks may vary
- Like industry compliance standards (PCI, HIPPA,…), not an end-goal
  - Should be considered a *minimum* baseline for application security

What Is Risk?

- Risk is the intersection of a threat, a weakness and an asset.

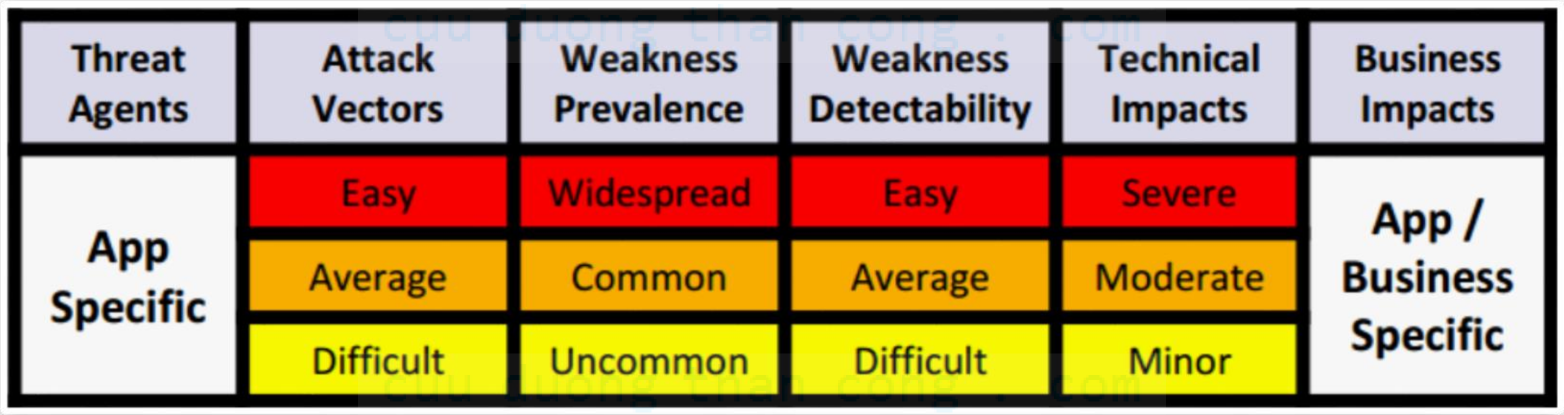


Risk = Likelihood * Impact

Image Source: OWASP Top Ten 2017 rc1

OWASP Risk Calculation

- Each risk is calculated using generic vulnerability facts, based on the OWASP Risk Rating Methodology [1]…



Threat Agents	Attack Vectors	Weakness Prevalence	Weakness Detectability	Technical Impacts	Business Impacts
App Specific	Easy	Widespread	Easy	Severe	App / Business Specific
	Average	Common	Average	Moderate	
	Difficult	Uncommon	Difficult	Minor	

…but impact is environment and business specific!

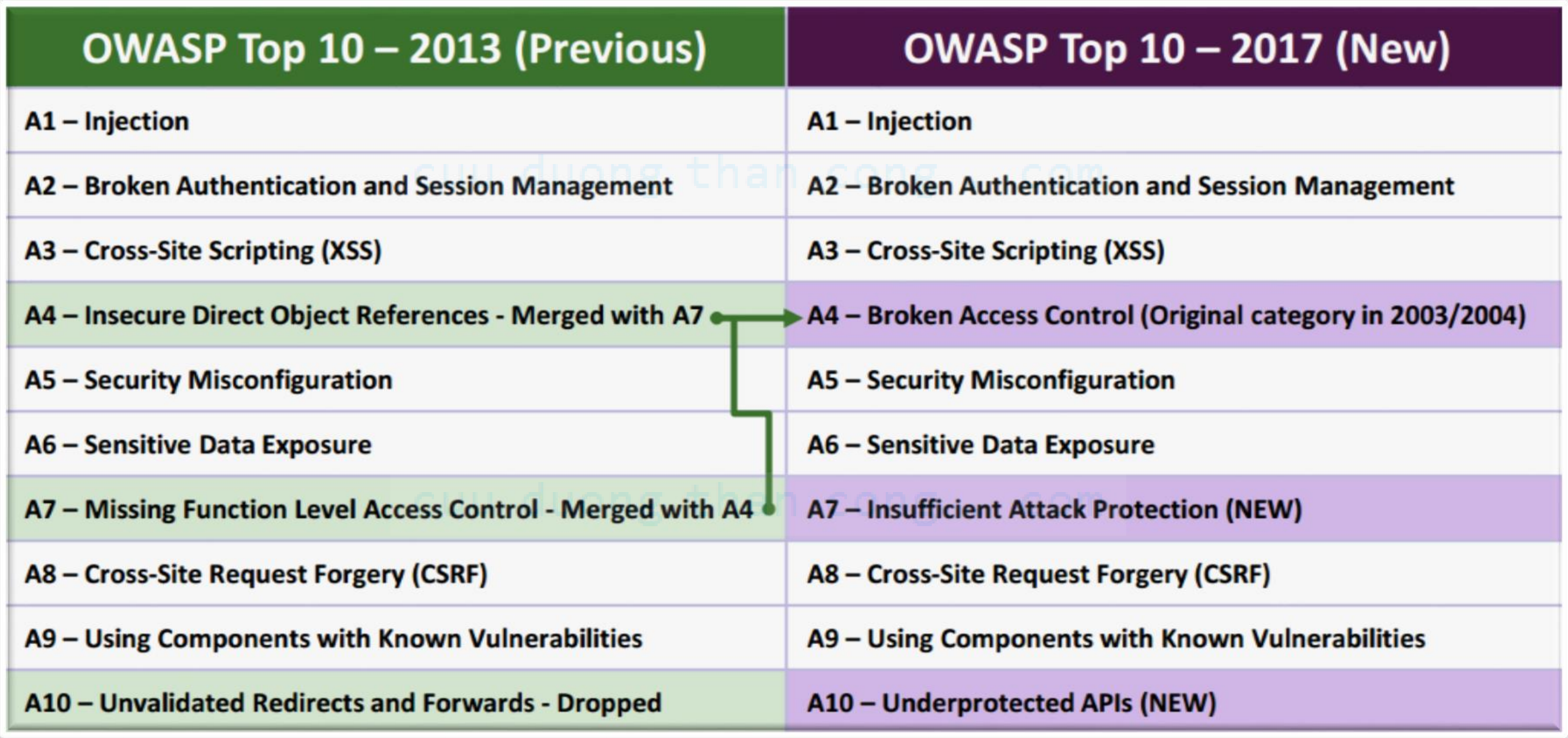
Image Source: OWASP Top Ten 2017 rc1

[1] https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology

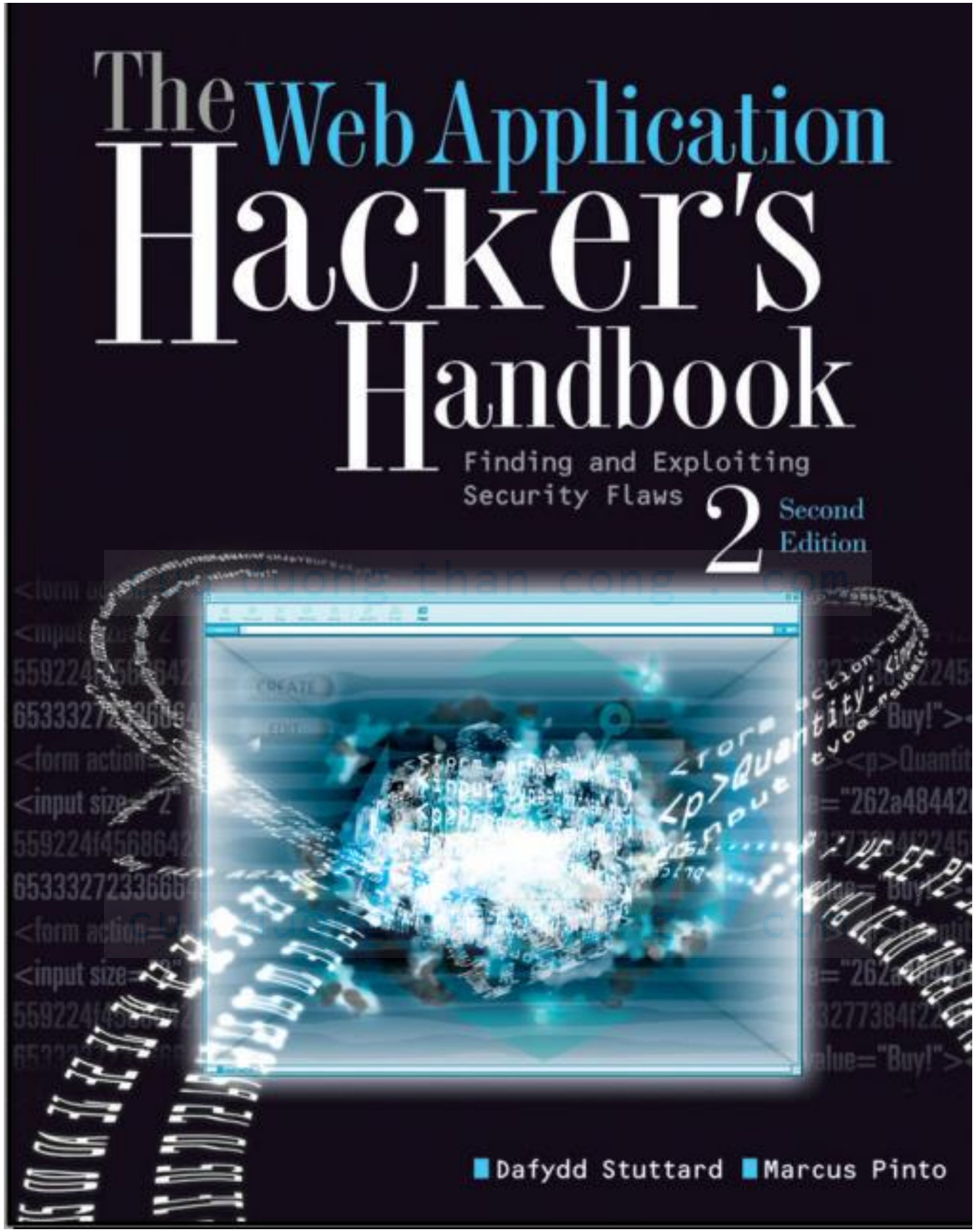
OWASP Top Ten - 2017 rc1

- A1 - Injection
- A2 – Broken Authentication and Session Management
- A3 – Cross-Site Scripting (XSS)
- A4 – Broken Access Control
- A5 – Security Misconfiguration
- A6 – Sensitive Data Exposure
- A7 – Insufficient Attack Protection
- A8 – Cross-Site Request Forgery (CSRF)
- A9 – Using Components with Known Vulnerabilities
- A10 – Underprotected APIs

What changed?



OWASP Top 10 – 2013 (Previous)	OWASP Top 10 – 2017 (New)
A1 – Injection	A1 – Injection
A2 – Broken Authentication and Session Management	A2 – Broken Authentication and Session Management
A3 – Cross-Site Scripting (XSS)	A3 – Cross-Site Scripting (XSS)
A4 – Insecure Direct Object References - Merged with A7	A4 – Broken Access Control (Original category in 2003/2004)
A5 – Security Misconfiguration	A5 – Security Misconfiguration
A6 – Sensitive Data Exposure	A6 – Sensitive Data Exposure
A7 – Missing Function Level Access Control - Merged with A4	A7 – Insufficient Attack Protection (NEW)
A8 – Cross-Site Request Forgery (CSRF)	A8 – Cross-Site Request Forgery (CSRF)
A9 – Using Components with Known Vulnerabilities	A9 – Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards - Dropped	A10 – Underprotected APIs (NEW)



The Web Application Hacker's Handbook

Finding and Exploiting Security Flaws

2 Second Edition

Dafydd Stuttard Marcus Pinto