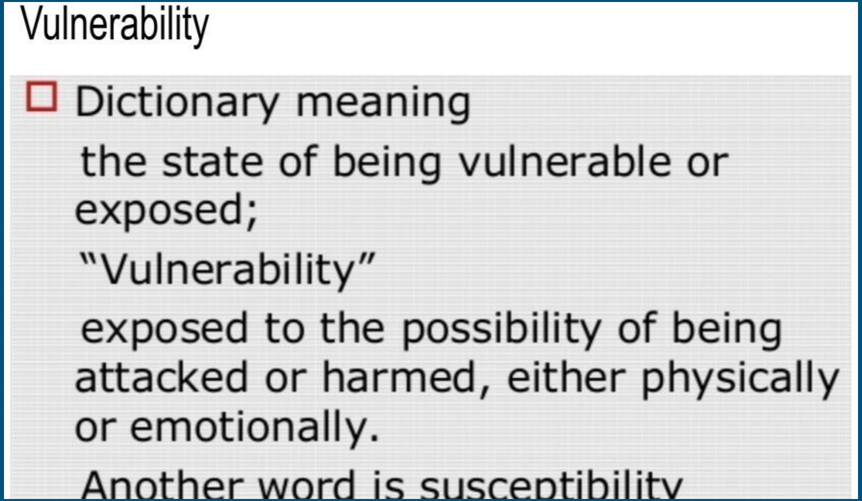
Security Testing

Security - Vulnerabilities & ZAP

Anshul Gautam & Aashish Khetarpal



Vulnerability

- Dictionary meaning

  the state of being vulnerable or exposed;

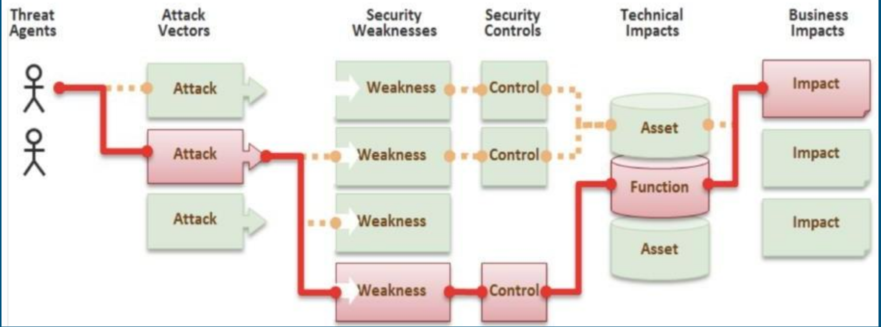
  "Vulnerability"

  exposed to the possibility of being attacked or harmed, either physically or emotionally.

  Another word is susceptibility

Zero Day

Attack Vector



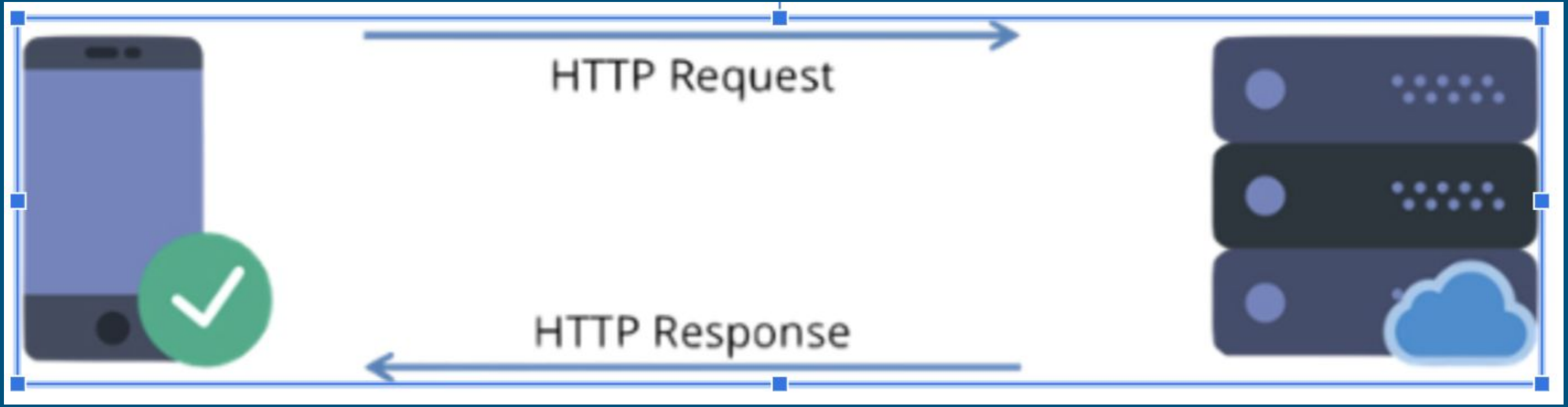
Attack Surface

Web Security Concepts

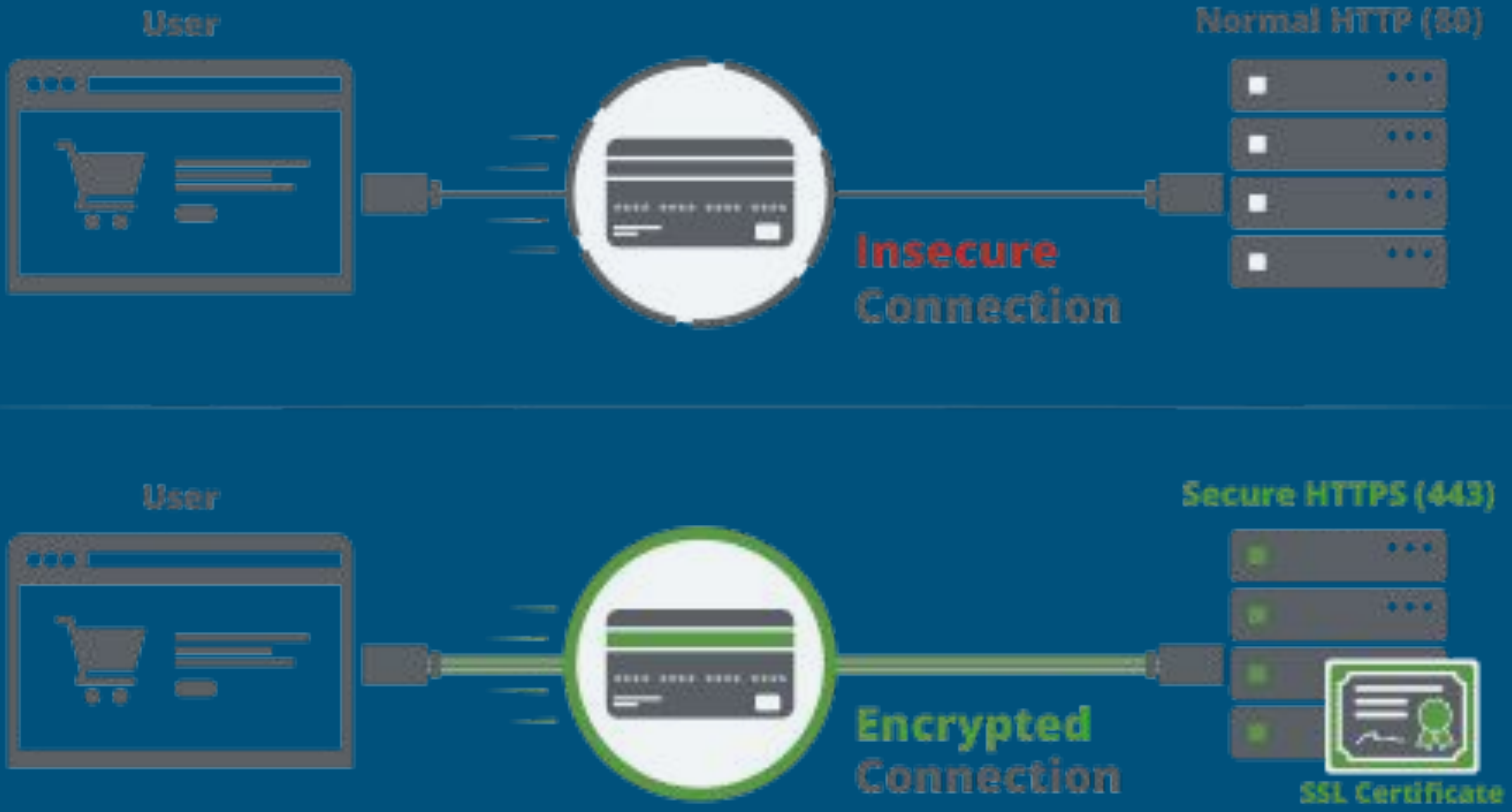
1. HTTP VS HTTPS
2. Security Headers
3. Session Management

HTTP

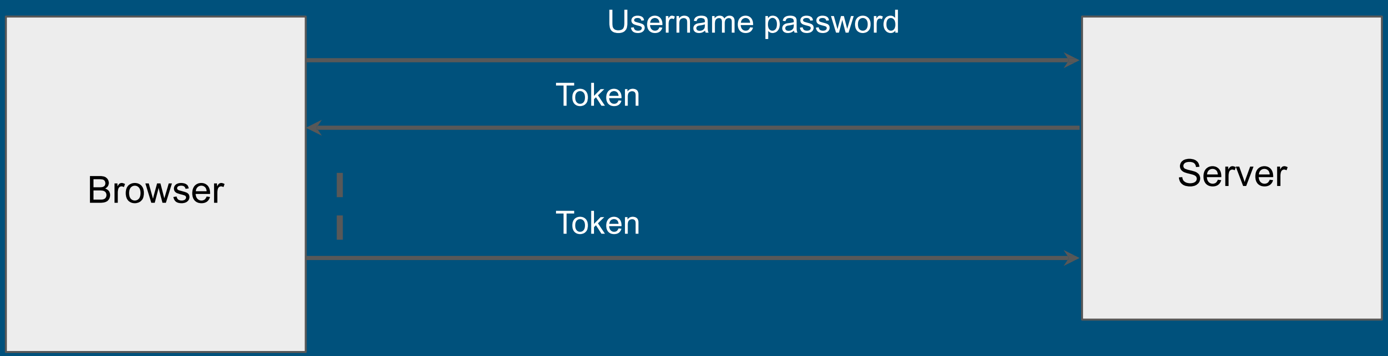
HTTP is a protocol which allows the fetching of resources, such as HTML documents.



HTTP vs HTTPS



Session Management

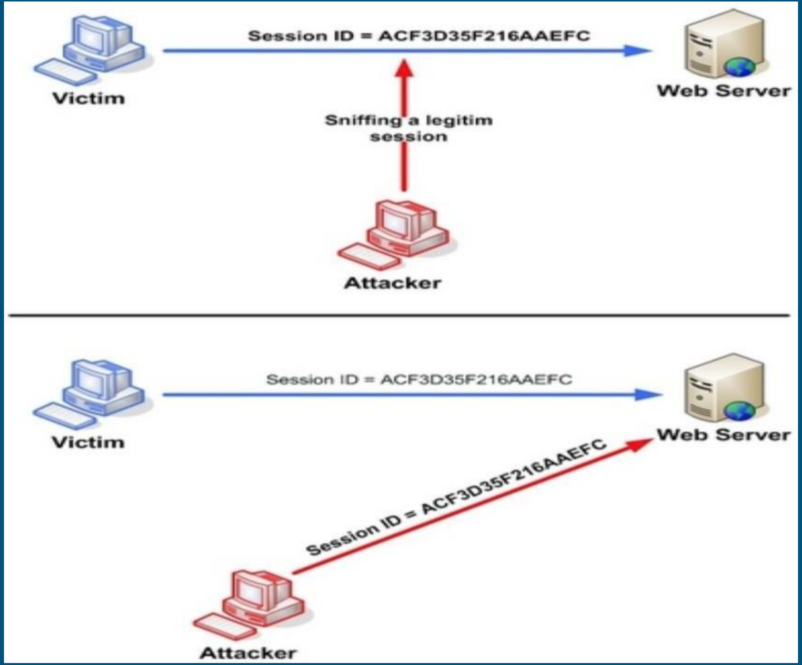


Where is this info stored?

Cookies

Local storage

Session Hijacking


Session ID = ACF3D35F216AAEFC
Victim
Web Server
Sniffing a legitim session
Attacker
Session ID = ACF3D35F216AAEFC
Victim
Web Server
Session ID = ACF3D35F216AAEFC
Attacker

OWASP

Open Web Application Security Project (OWASP) is a nonprofit foundation that works to improve the security of software.

● Website: owasp.org
● A bunch of cool tools: Zed Attack Proxy, Juice Shop, Proactive Controls, Software Assurance Maturity Model (SAMM), Application Security Verification Standard (ASVS)
● A global community of like-minded people, meetups and conferences

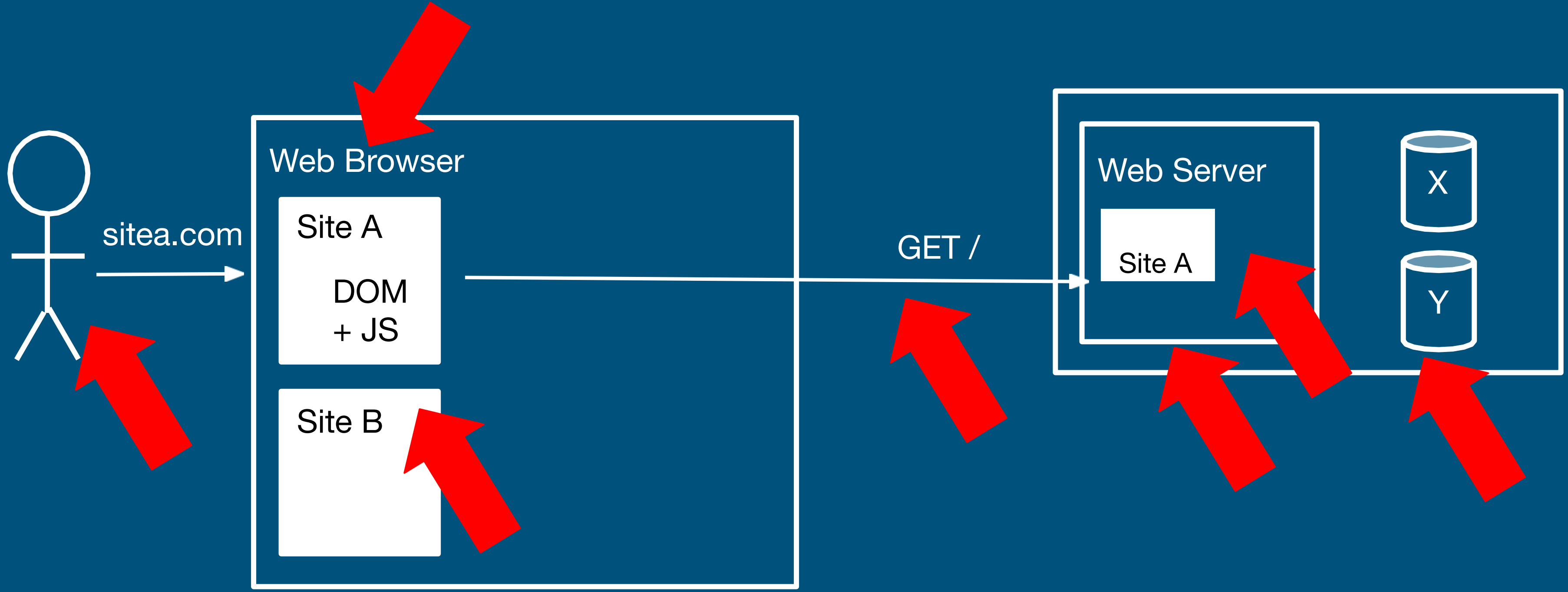
OWASP Top Ten

Globally recognized by developers as the first step towards more secure coding.

The *most critical* security risks to web applications.

Updated every 2-3 years from 2003 to 2017 (2021 is recently out)

Securing the user



OWASP Top 10 (2017)

Injection

Broken Authentication

Sensitive Data Exposure

XML External Entities (XXE)

Broken Access Control

Security Misconfigurations

Cross Site Scripting (XSS)

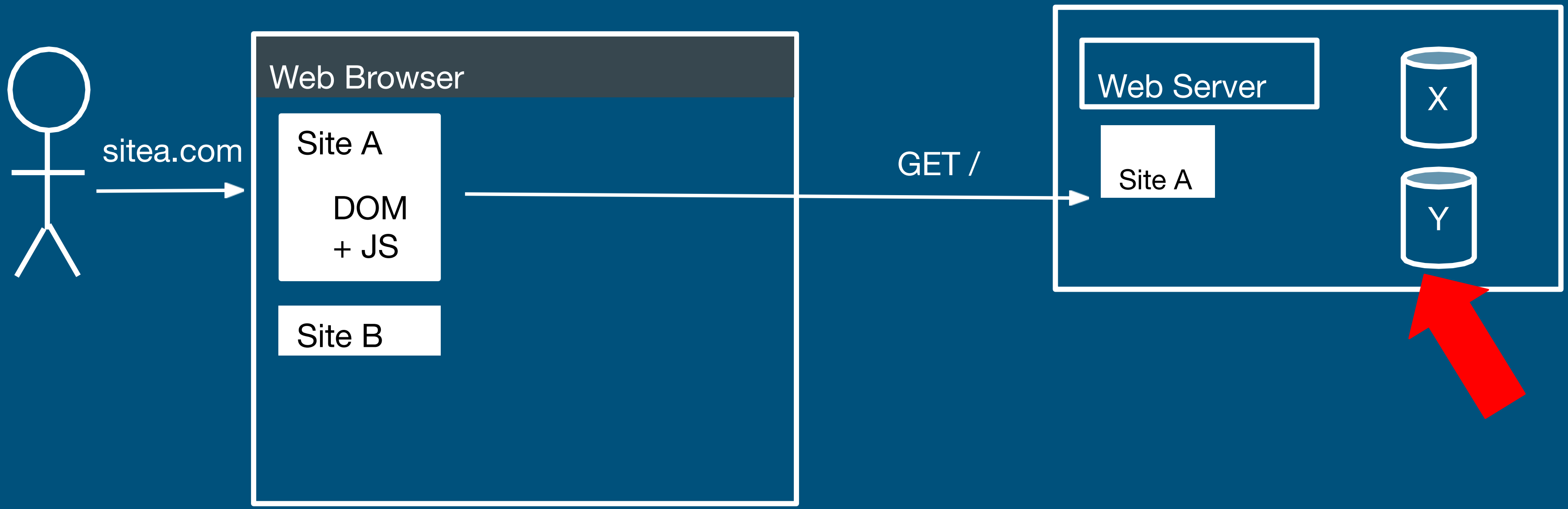
Insecure Deserialization

Using components with known vulnerabilities

Insufficient Logging and Monitoring

A1 - Injection

Sending hostile data to an interpreter (e.g. SQL, command line)



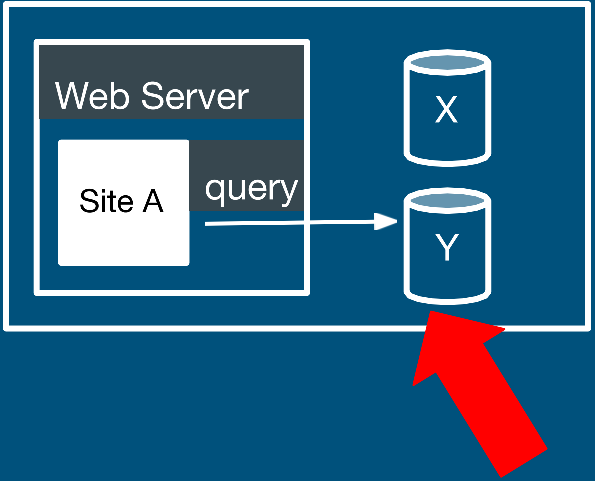
A1 - Injection

Sending hostile data to an interpreter (e.g. SQL, command line)

```
String query = "SELECT * FROM accounts WHERE
custID='" + request.getParameter("id") + "'";

id = " '; drop table accounts -- "
```

SQL statements combine *code* and *data*



SQLi Demo

https://application.security/

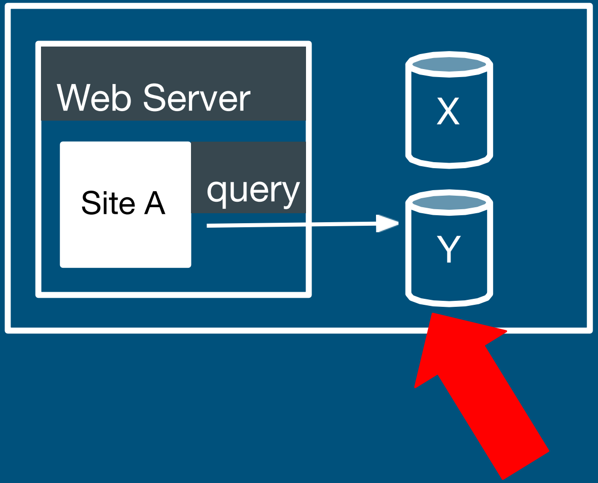
A1 - Injection

Prevention:

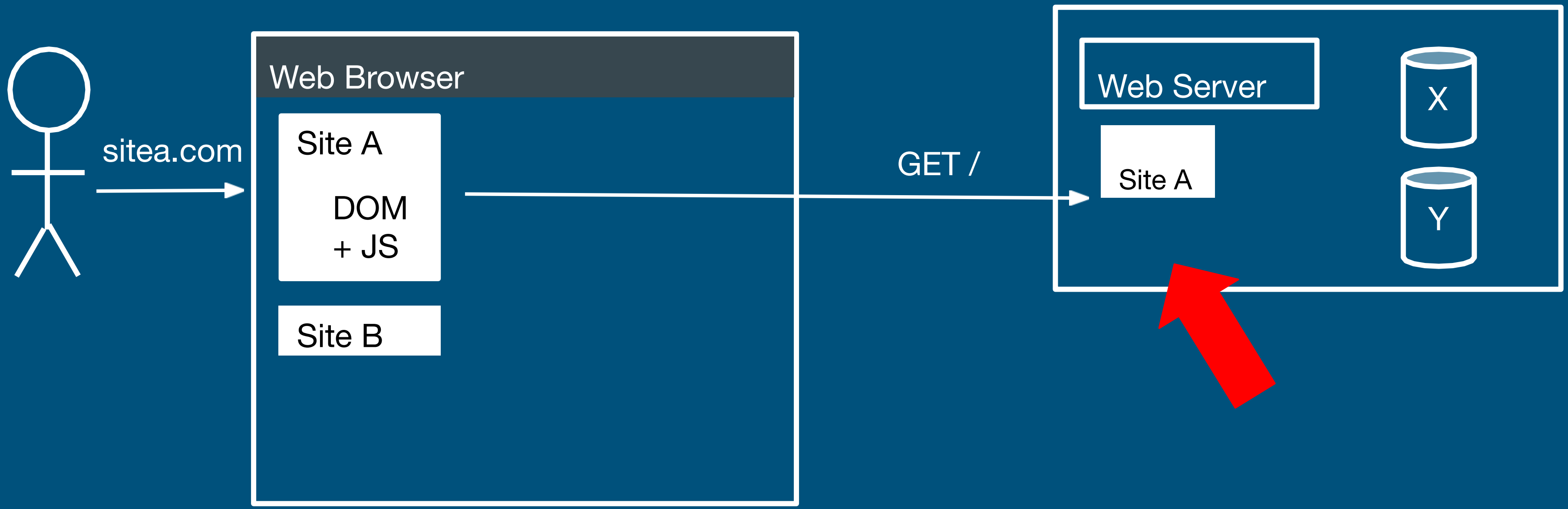
SQL statements combine *code* and *data*

=> Separate code and data

- Parameterise your queries
- Validate which data can be entered
- Escape special characters

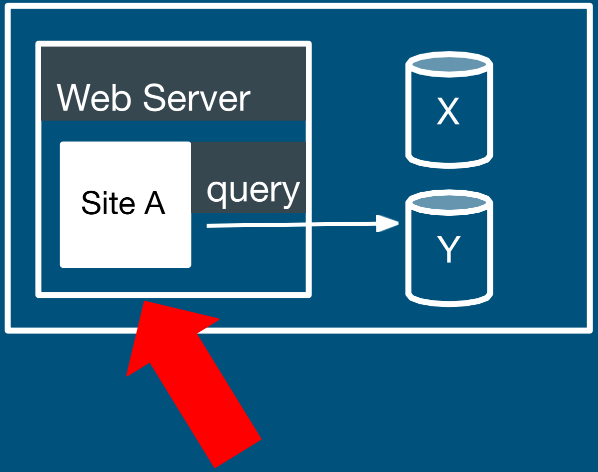


A2 - Broken Authentication



A2 - Broken Authentication

- Weak session management
- Credential stuffing
- Brute force
- Forgotten password
- No multi-factor authentication
- Sessions don't expire



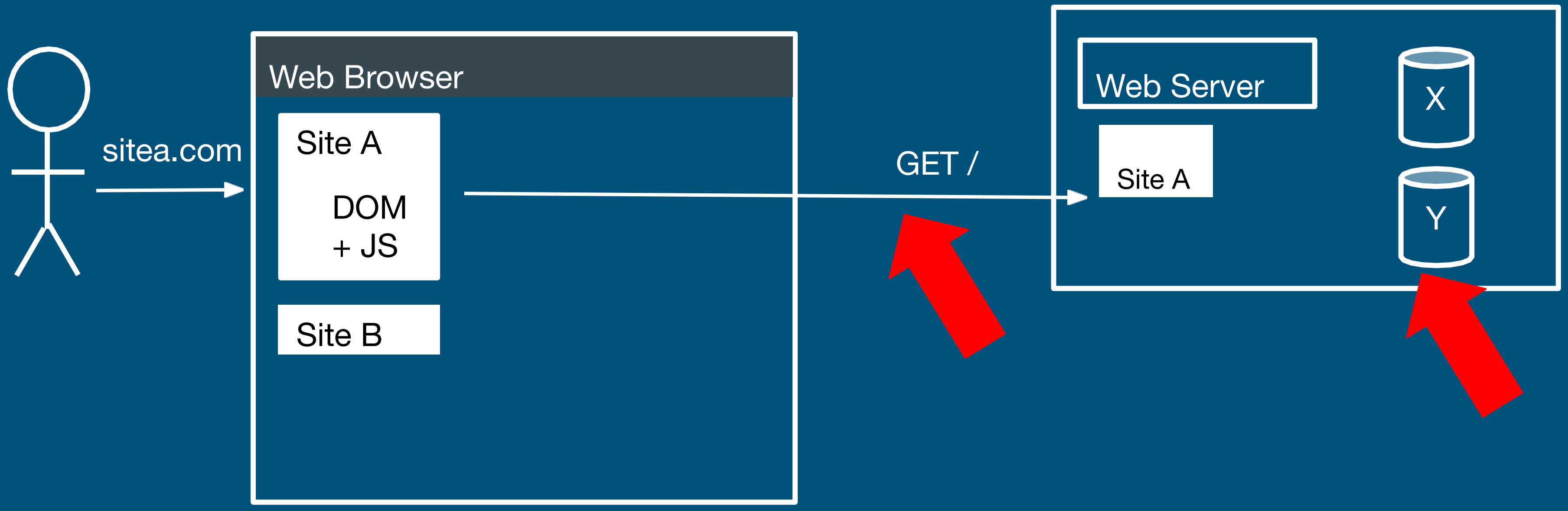
A2 - Broken Authentication

Prevention:

- Use good authentication libraries
- Use MFA
- Enforce strong passwords
- Detect and prevent brute force or stuffing attacks

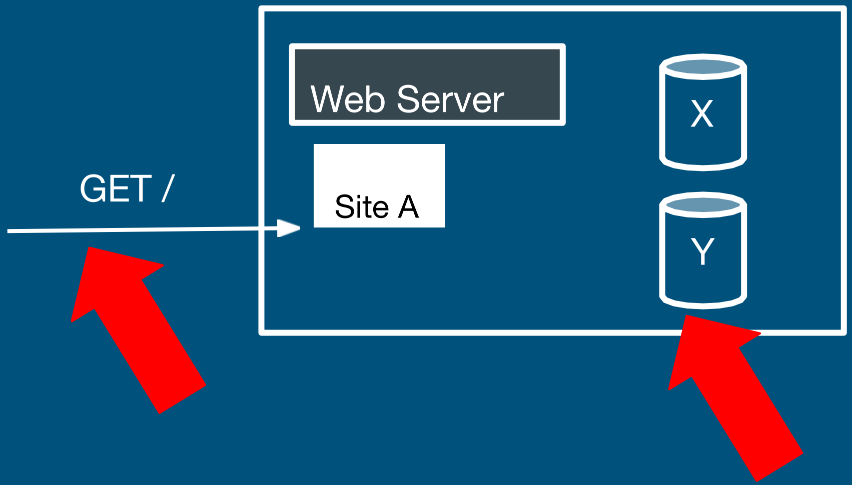
Session Fixation Demo

A3 - Sensitive Data Exposure

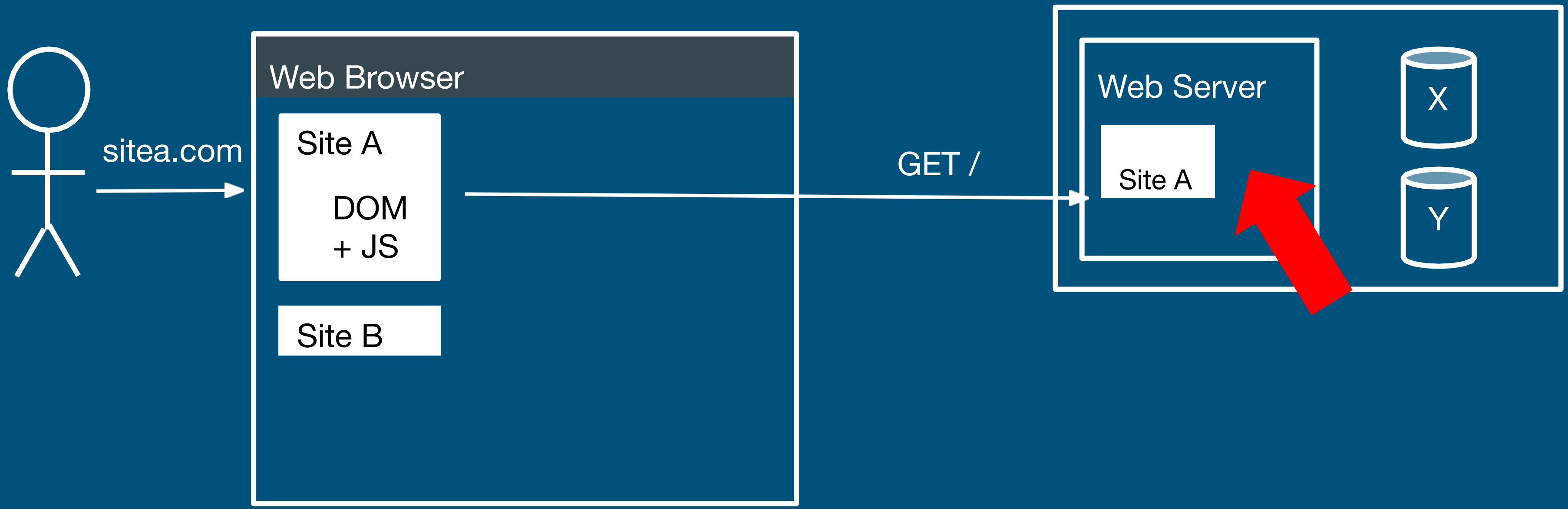


A3 - Sensitive Data Exposure

- Clear-text data transfer
- Unencrypted storage
- Weak crypto or keys
- Certificates not validated
- Exposing PII or Credit Cards



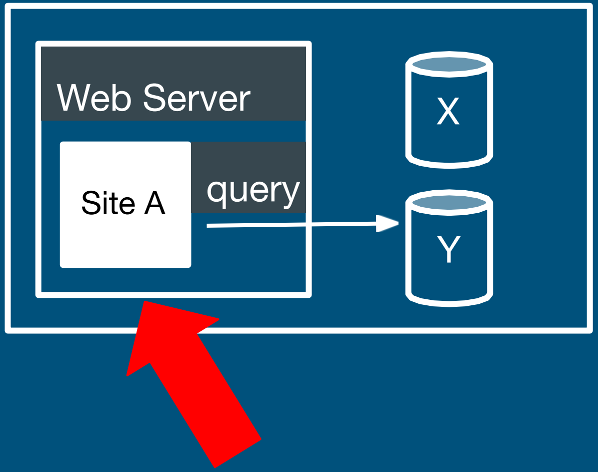
A4 - XML External Entities (XXE)



A4 - XML External Entities (XXE)

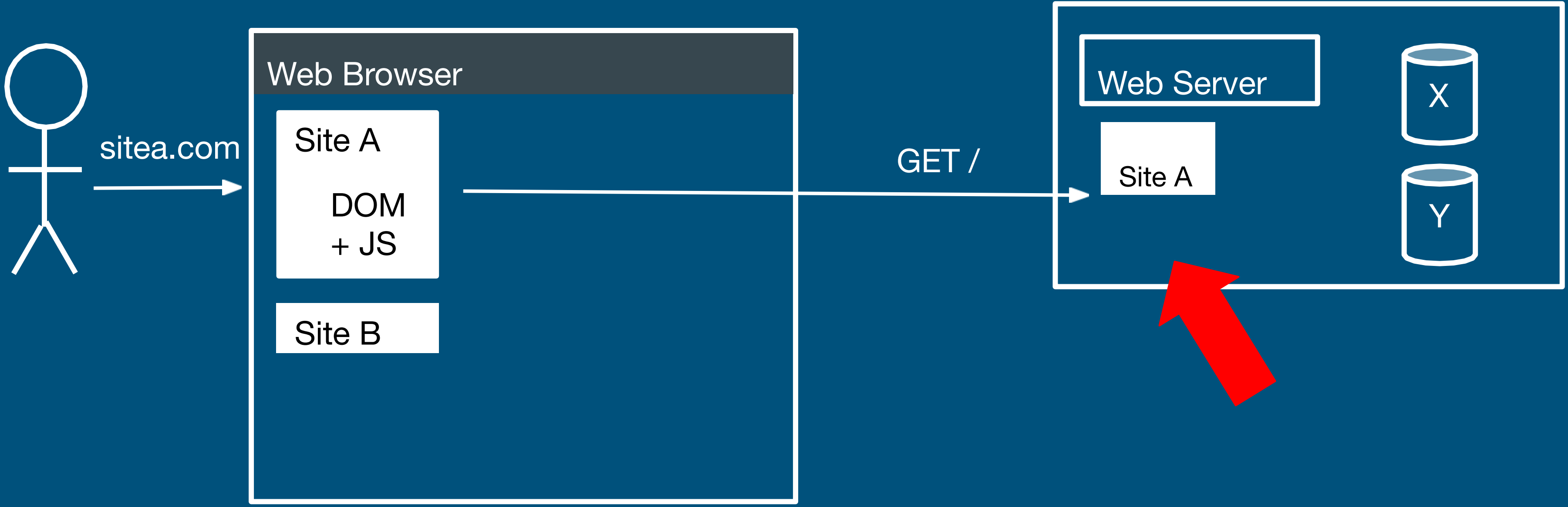
The application accepts XML,
and assumes it is safe

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<foo>&xxe;</foo>
```



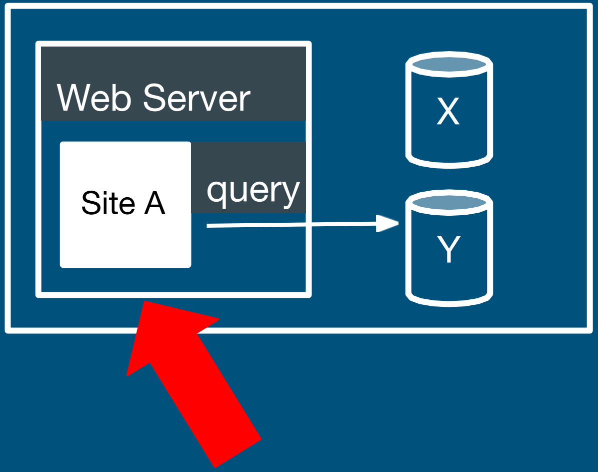
Can allow accessing sensitive resources, command execution, recon, or cause denial of service.

A5 - Broken Access Control

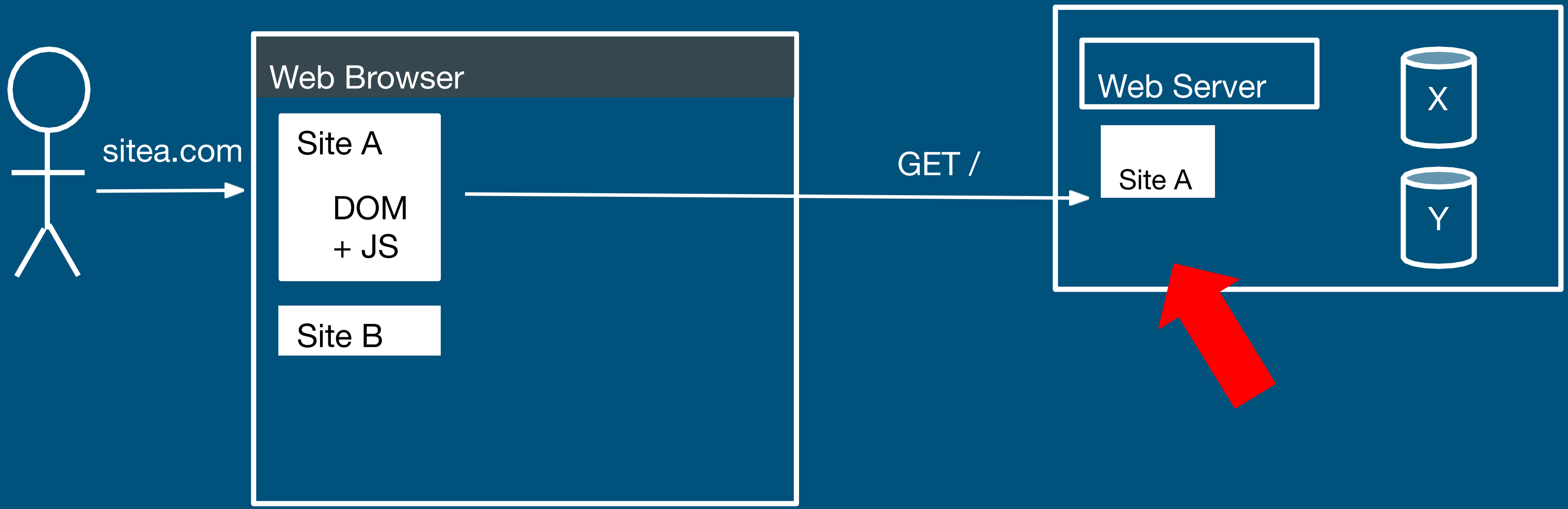


A5 - Broken Access Control

- Access hidden pages
- Elevate to an administrative account
- View other people's data
- Modifying cookies or JWT tokens

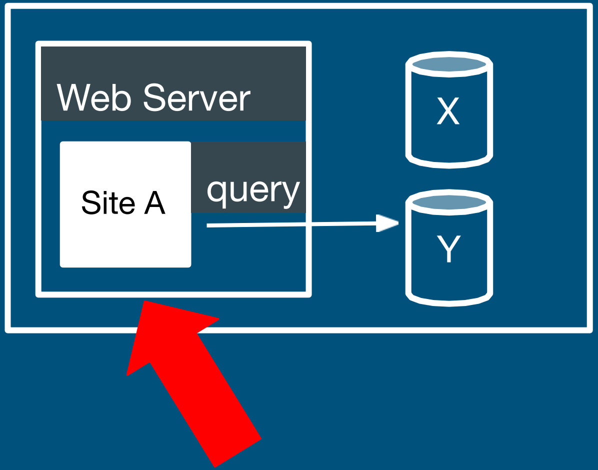


A6 - Security Misconfiguration

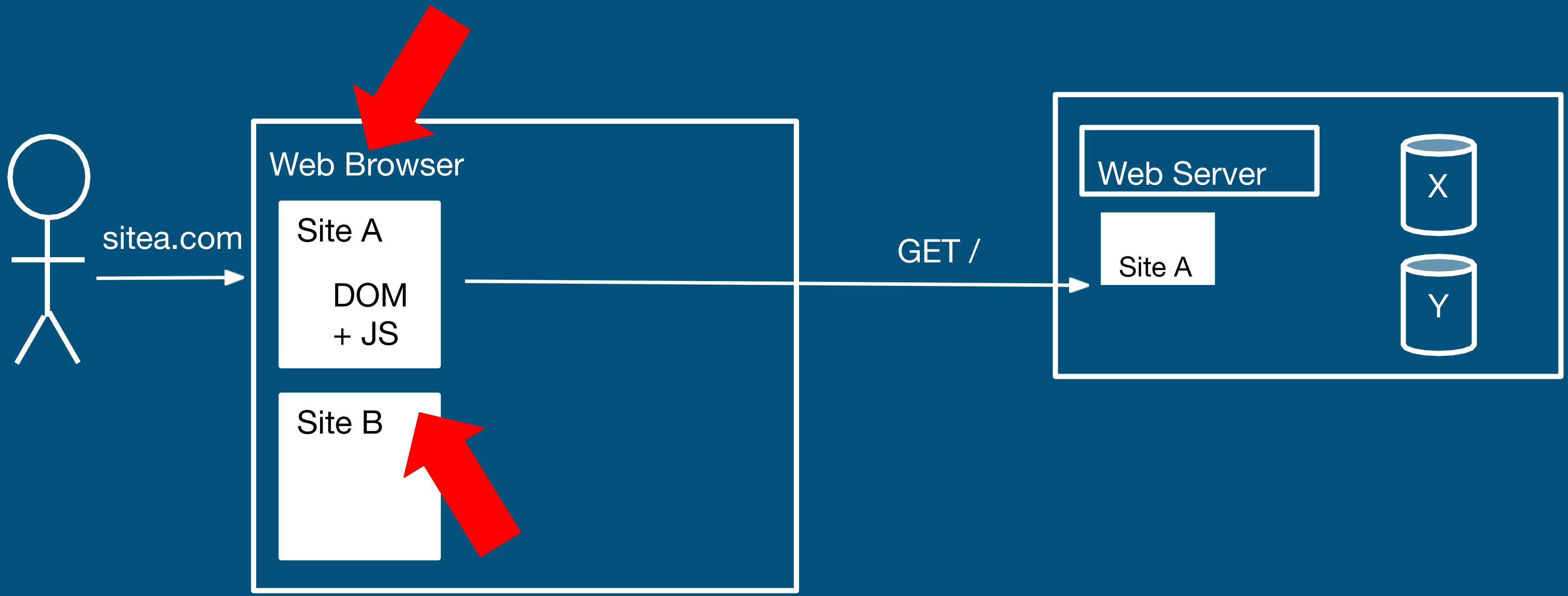


A6 - Security Misconfiguration

- Security features not configured properly
- Unnecessary features enabled
- Default accounts not removed
- Error messages expose sensitive information



A7 - Cross-Site Scripting (XSS)

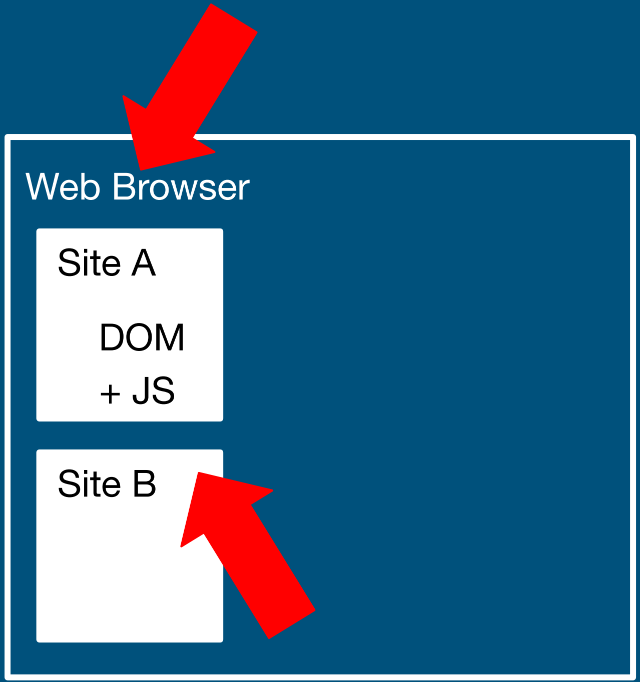


A7 - Cross-Site Scripting (XSS)

HTML mixes content, presentation and code into one string (HTML+CSS+JS)

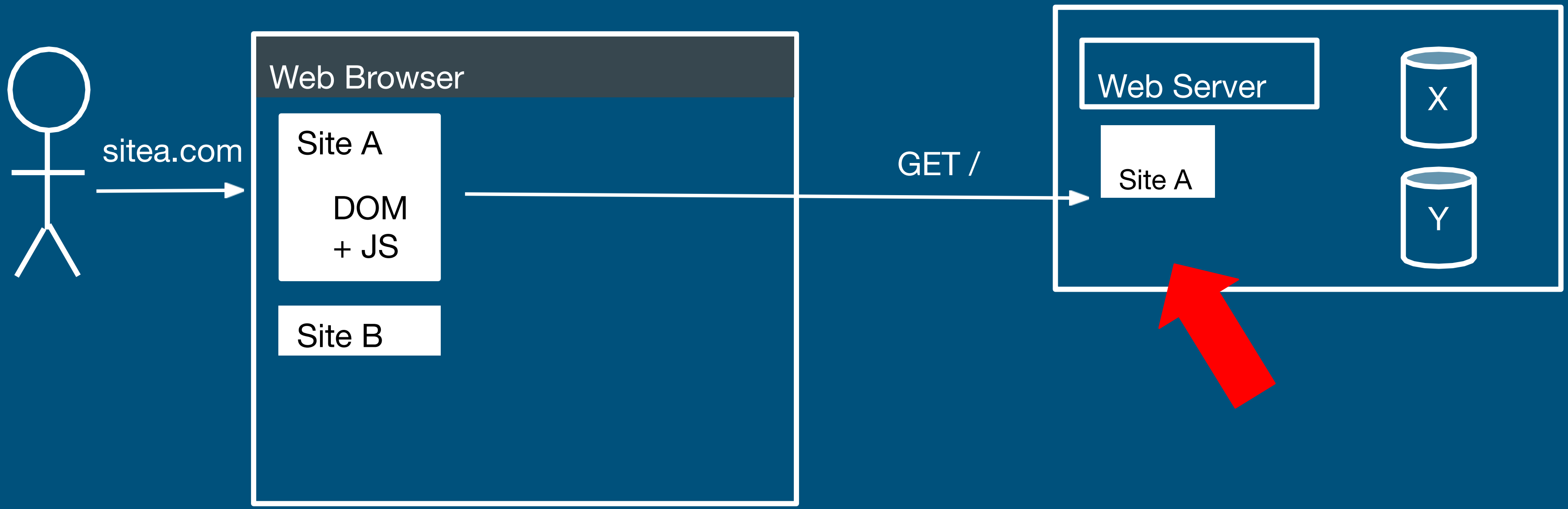
If an attacker can alter the DOM, they can do *anything* that the user can do.

XSS can be found using automated tools.



Stored Cross Site Scripting Demo

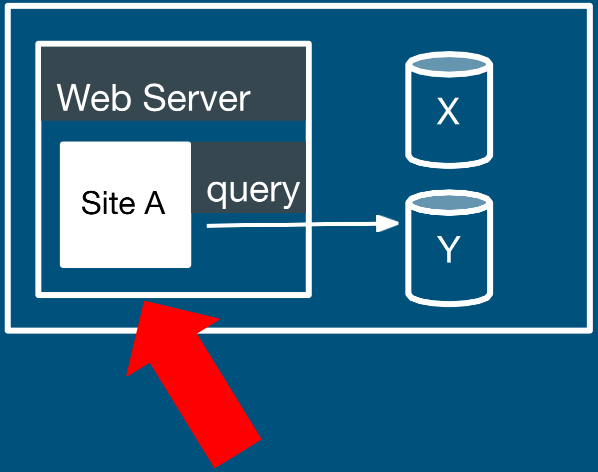
A8 - Insecure Deserialization



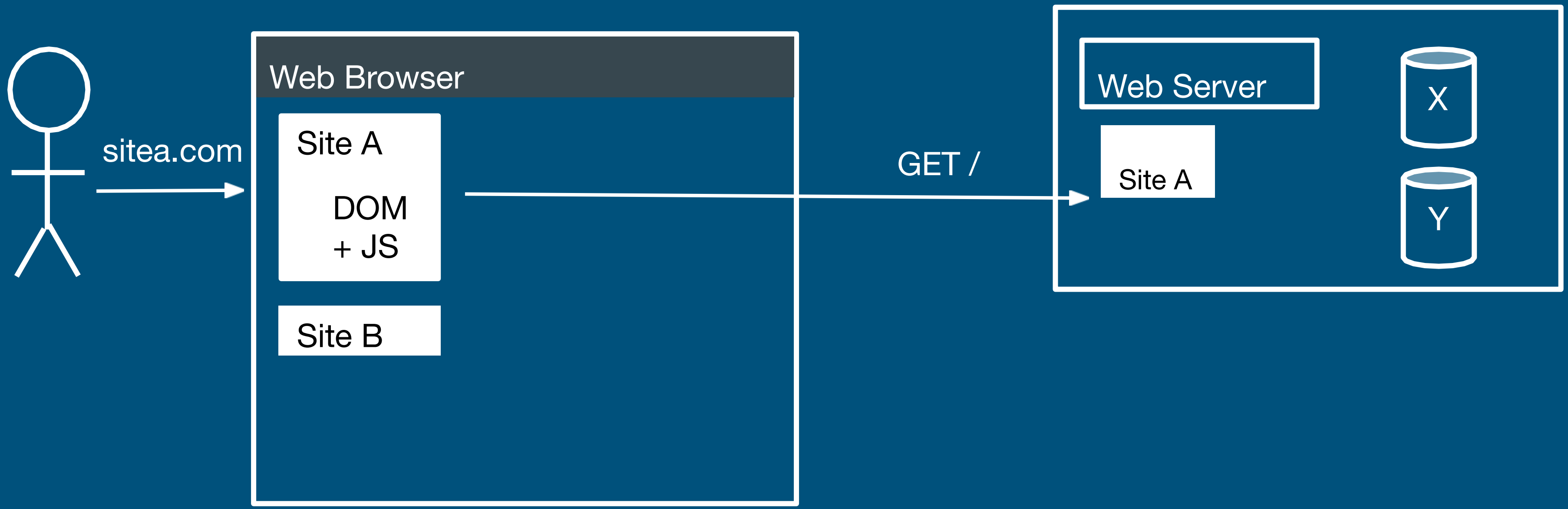
A8 - Insecure Deserialization

Programming languages allow you to turn a tree of objects into a string that can be sent to the browser.

If you *deserialize* untrusted data, you may allow objects to be created, or code to be executed.



A9 - Using Components with Known Vulnerabilities



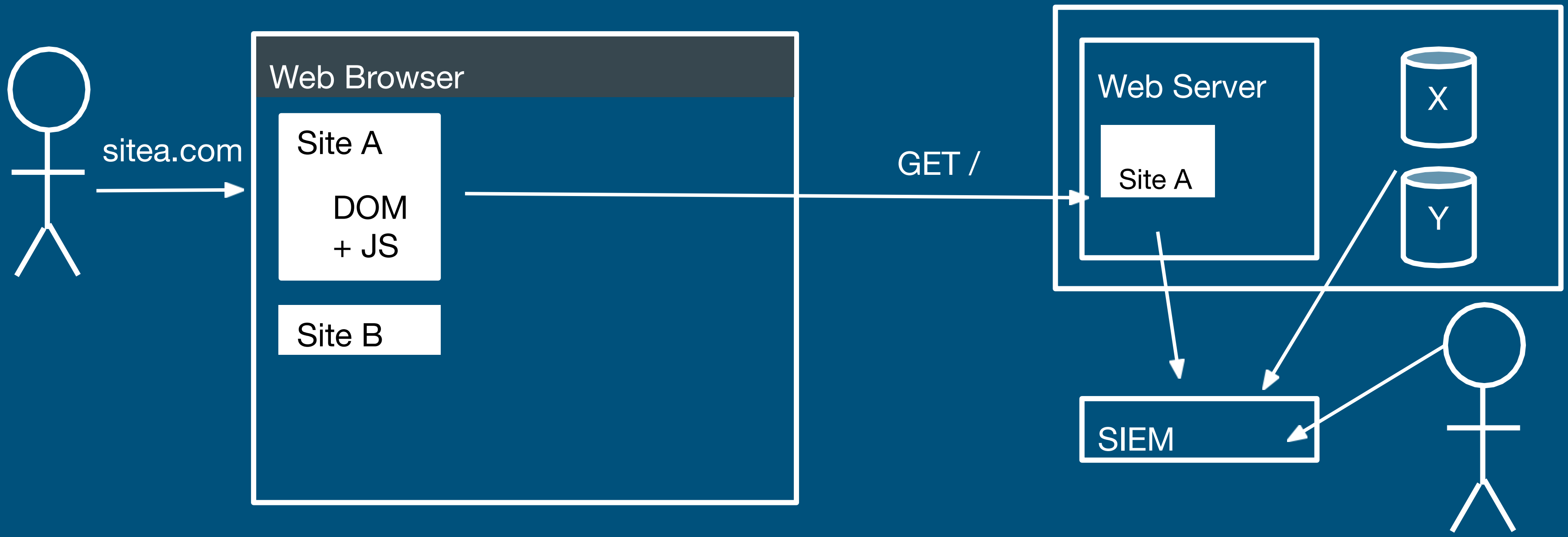
A9 - Using Components with Known Vulnerabilities

Modern applications contain a *lot*
of third-party code.

It’s hard to keep it all up to date.

Attackers can enumerate the
libraries you use, and
develop exploits.

A10 - Insufficient Logging & Monitoring

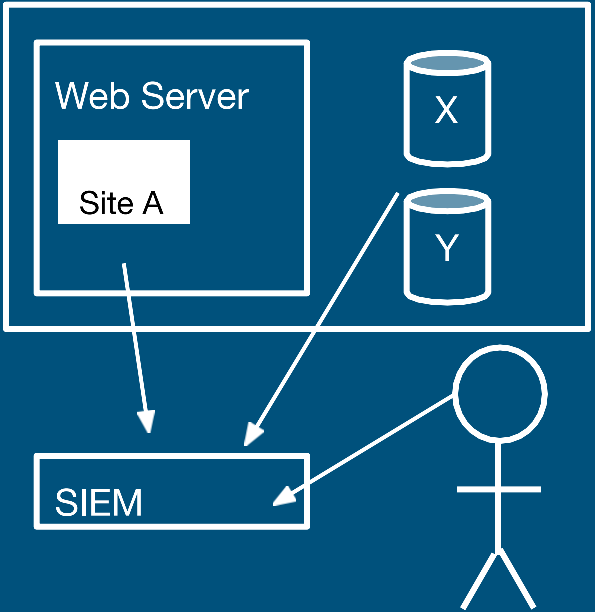


A10 - Insufficient Logging & Monitoring

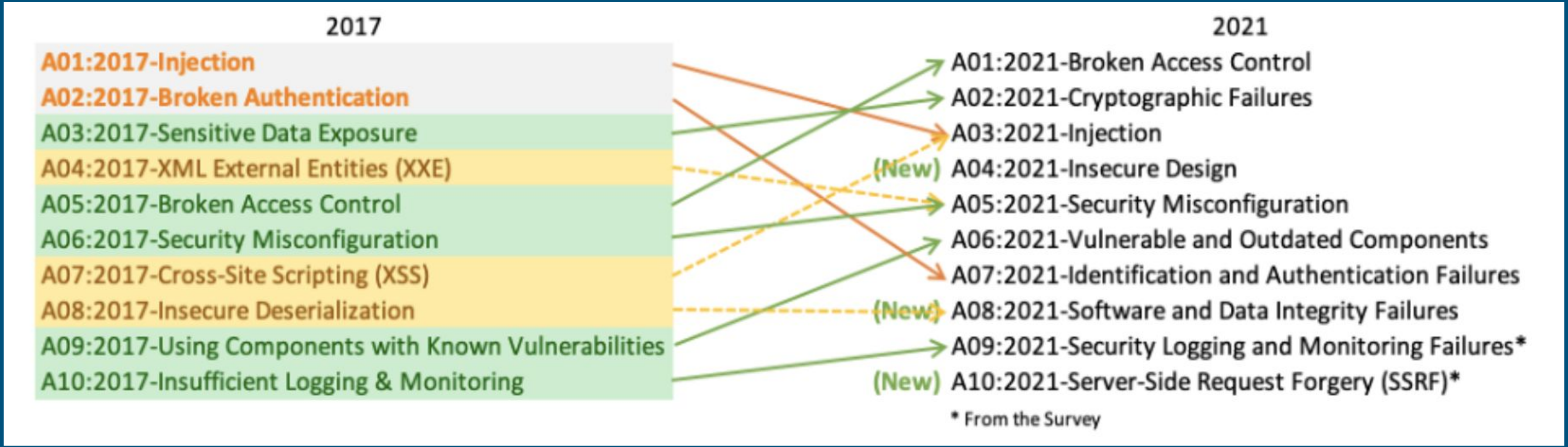
You can't react to attacks that you don't know about.

Logs are important for:

- Detecting incidents
- Understanding what happened
- Proving who did something



2017 vs 2021



2017		2021
A01:2017-Injection		A01:2021-Broken Access Control
A02:2017-Broken Authentication		A02:2021-Cryptographic Failures
A03:2017-Sensitive Data Exposure		A03:2021-Injection
A04:2017-XML External Entities (XXE)	(New)	A04:2021-Insecure Design
A05:2017-Broken Access Control		A05:2021-Security Misconfiguration
A06:2017-Security Misconfiguration		A06:2021-Vulnerable and Outdated Components
A07:2017-Cross-Site Scripting (XSS)		A07:2021-Identification and Authentication Failures
A08:2017-Insecure Deserialization	(New)	A08:2021-Software and Data Integrity Failures
A09:2017-Using Components with Known Vulnerabilities		A09:2021-Security Logging and Monitoring Failures*
A10:2017-Insufficient Logging & Monitoring	(New)	A10:2021-Server-Side Request Forgery (SSRF)*

* From the Survey

Let’s get our hands dirty using ZAP

ZAP?

- **OWASP ZAP** (short for Zed Attack Proxy) is an **open-source web application security scanner.**
- One of the most active **OWASP** projects.
- Ideal for beginners.
- Used by professionals.
- When used as a **proxy server** it allows the user to manipulate all of the traffic that passes through it.
- This **cross-platform** tool is written in **Java** and is available in all of the popular operating systems including **Microsoft Windows, Linux and Mac OS X.**