



# 1

Be a Ninja!



# Demystifying Security Ninjas

Manuel Zamith  
Nuno Barros

Security in Software Engineering



**U.PORTO**  
**FEUP** FACULDADE DE ENGENHARIA  
UNIVERSIDADE DO PORTO



# 2

Be a Ninja!



- Injection is used by an attacker to introduce (or "inject") code into a vulnerable computer program and change the course of execution.
- Injection can result in data loss or corruption, lack of accountability, or denial of access. Injection can sometimes lead to complete host takeover.
- Examples:
  - SQL Injection
  - Shell injection
  - HTML script injection



# Injection



# 3

Be a Ninja!



# Broken Authentication and Session Management

- This type of flaws can be extremely serious not only because putting business at very high risk, lose confidential data is an option, but also because it can lead to opening of backdoors to malicious attackers, on the entire company.
- A single set of strong authentication and session management controls that should strive to:
  - Meet all the context requirements defined in OWASP Application Security Verification Standard (ASVS) areas V2 (Authentication) and V3 (Session Management)



# 4

Be a Ninja!



- Cross-site Scripting (XSS) refers to client-side code injection attack wherein an attacker can execute malicious scripts (also commonly referred to as a malicious payload) into a legitimate website or web application.
- XSS flaws occur when an application includes user supplied data in a page sent to the browser without properly validating or escaping that content.




# Cross-site Scripting (XSS)

<

>

# 5

Be a Ninja!



# Insecure Direct Object References

- Also called IDOR, this vulnerability refers to a reference to an internal implementation object, such as a file, directory or a database key is exposed without any access control. In such cases those references can be manipulated by an attacker intending to access unauthorised data.
- The only real solution, as of now, is to implement an access control system.



# 6

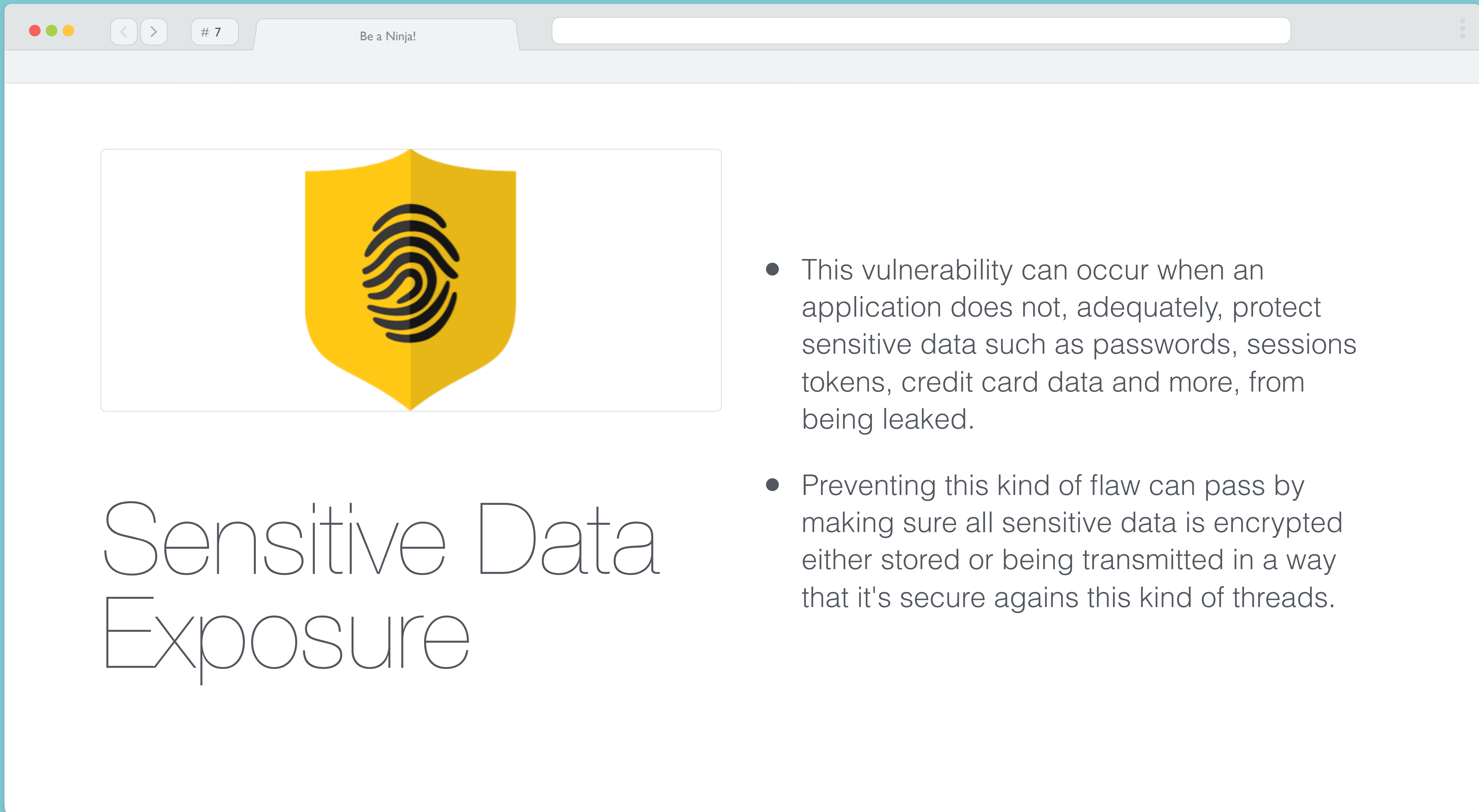
Be a Ninja!

- Security misconfiguration vulnerabilities could occur if a component is susceptible to attack due to an insecure configuration option. These vulnerabilities often occur due to insecure default configuration, poorly documented default configuration, or poorly documented side-effects of optional configuration.
- Good security requires having a secure configuration defined and deployed for the application, frameworks, application server, web server, database server, and platform.



# Security Misconfiguration





# Sensitive Data Exposure

- This vulnerability can occur when an application does not, adequately, protect sensitive data such as passwords, sessions tokens, credit card data and more, from being leaked.
- Preventing this kind of flaw can pass by making sure all sensitive data is encrypted either stored or being transmitted in a way that it's secure against this kind of threats.



# 8

Be a Ninja!



- Function level access control vulnerabilities could result from insufficient protection of sensitive request handlers within an application.
- Making sure the right people have access to appropriate functions of your app is a growing concern. Hiding these functions is not enough. You must enforce your access restrictions from the server as well.



# Missing Function Level Access Control




<

>

# 9

Be a Ninja!



# Cross-site Request Forgery (CSRF)

- Also known as CSRF, this is an attack that forces a currently authenticated user to perform unwanted actions in the system, without his knowledge.
- While in XSS the attacker exploits the trust a user has for a website, with CSRF on the other hand, the attacker exploits the trust a website has against a user's browser.



# 10

Be a Ninja!



- Both commercial and free open source software components are regularly included in web application development projects.
- However, with these developer benefits come a few downsides. As a result of this approach to software development the popular third party components are very widely used. This means that they are known to anyone looking to attack a web application, and that any vulnerability discovered provides a rich target to exploit.




# Using Components with Known Vulnerabilities

<

>

# 11

Be a Ninja!



Unvalidated  
Redirects and  
Forwards

- This flaw occurs when a malicious attacker is able to redirect a user to an untrusted website when a trusted website link is visited.
- This might look nonsense but the best way to get rid of this flaw is to avoid the usage of redirects and forwards. Though, if it's imperative to use them, try not to involve user parameters in the destination calculation.