# Vulnerability Scanning with Nikto

**Objective:**
Using Nikto to perform a vulnerability scan on a web server and explaining the results.

**1.Overview and Vulnerabilities found**
Nikto is a tool that can be installed on Linux and Windows operating system. It is used to scan any vulnerabilities on a web server. You can also use Nikto with other apps like Burp Suite to get a deeper scan of the web server. It can scan for various types of vulnerabilities, such as SQL injection, cross-site scripting, broken authentication, and other common web application flaws.

**Vulnerabilities**
1. Missing X-Frame-Options Header: Medium to High severity
Description:
The website is vulnerable to clickjacking attacks due to the absence of the X-Frame-Options header in the HTTP response headers, which could be used by attackers to trick users into performing unwanted actions by loading your website within iframes on other malicious websites.
Mitigation:
Add the following header to your web server responses:
X-Frame-Options: DENY or X-Frame-Options: SAMEORIGIN

2. X-Content-Type-Options Header Not Set: Medium severity
Description:
If the X-Content-Type-Options header is absent from the HTTP response headers, MIME type sniffing may be possible, resulting in browsers interpreting received content as a different MIME type than declared, which could result in malicious scripts executing or content injection or cross-site scripting (XSS) attacks.
Mitigation:
Configure server to add the following header:

X-Content-Type-Options: nosniff
This instructs browsers to strictly follow the MIME type declared by the server and not perform content type sniffing, reducing the risk of XSS and content injection vulnerabilities.

3. Cookie Without the HttpOnly Flag: Medium severity
Description:
These cookies lack the HttpOnly attribute, which prevents access via client-side scripts like JavaScript, and therefore can be stolen via cross-site scripting (XSS) attacks, compromising user sessions and potentially user accounts. These cookies can be stolen by bad people who use scripts to get into your computer and your account. They do not have a special feature that makes them harder to steal.
Mitigation:
Set the HttpOnly flag on all sensitive cookies, especially session cookies. By doing this, you can stop cookies from being sent through document.cookie and make it harder for attackers to steal cookies. Note that HttpOnly does not prevent XSS itself but mitigates cookie capture.

4. Usage of clientaccesspolicy.xml Beyond Silverlight: Informational / Potential Risk severity
Description:
The files are intended to allow the Silverlight application to access the user's browser's cookies, history, and other data. If the files are not properly configured, the Silverlight application could potentially access sensitive user data, such as login credentials or financial information. These files tell the computer how to use different parts of the website and may let the computer do things it is not supposed to do, such as seeing or changing information that it is not allowed to see or change. This can cause problems like letting people see or change information Check the settings of the clientaccesspolicy.xml file and make sure it lets in only the domains that are needed and gives them only the minimum access rights.
Mitigation:
Review the clientaccesspolicy.xml file configuration and ensure it only

allows necessary domains and minimal permissions. Remove or restrict this file if your application does not use Silverlight or similar cross-domain features.

5. Unrestricted HTTP Methods: High severity
Description:
Unrestricted HTTP methods like PUT, DELETE, TRACE, and CONNECT appear to be supported by the web server. These methods are not allowed in the HTTP specification, but they are commonly used in web applications. They can be used to perform various operations on the web server, such as updating, deleting, tracing, and connecting to other resources. However, they can also pose security risks Possible Some bad things you can do are: - Put bad files on the website (PUT) - Bring in or look for bad things on the website (TRACE).
Mitigation:
Limit the permitted HTTP methods to just those that are necessary, typically GET, POST, and maybe HEAD. Disable risky operations at the web server or application firewall level, including PUT, DELETE, TRACE, and CONNECT. Disable the use of the HTTP Strict Transport Security (HSTS) protocol. Disable the use of the HTTP Strict Transport Security (HSTS) protocol. Disable the use of the HTTP Strict Transport Security (HSTS).

6. Uncommon or Internal Headers Present: Informational severity
Description:
The web server's HTTP responses include uncommon or internal headers that might disclose server information or internal workings, increasing the attack surface. These headers can provide attackers with reconnaissance data.
Mitigation:
Audit and remove unnecessary or internal HTTP headers from responses unless explicitly required for application functionality or debugging. Minimize information disclosure to reduce reconnaissance risks.

**Conclusion**:

The scan revealed several key security header misconfigurations and server weaknesses that increase risk exposure to clickjacking, MIME sniffing attacks, XSS cookie theft, and unauthorized HTTP method exploitation. It is critical to implement security headers (X-Frame-Options, X-Content-Type-Options) correctly, harden cookie attributes, limit HTTP methods, and review policy files to ensure a robust security posture.