Lab 4 Advanced Web Exploitation  
CIS 450  
Krutarth Panchal

## Objective 1 - Section 2: Pastejacking

Creation of Vulnerable html file

A screen shot of a computer program

Description automatically generated  
Fig1   
  
As shown in the fig 1 we created a html file with code shown in the image.

Copying pastejacking.html into the browser and running it

A blue screen with white text

Description automatically generated  
Fig 2  
After that we copied the file to following location:  
 *var/www/html*

A screenshot of a computer

Description automatically generated  
Fig3

Navigated to the browser and opened our html file and it displayed “good” as hown in Fig3.  
  
Lab guide asked to copy “good” from the webpage and paste into the terminal and upon pasting it would display as “bad instead of “good” but that did not work for me. I rechecked the code and tried few times.

Changing the code to spawn Reverse shell and listening on port.

A screen shot of a computer

Description automatically generated  
Fig3  
As shown in the figure I edited the code in order to spawn the reverse shell as per as lab.

  
Fig 4  
Copied file to the var/www/html . I created a new file instead of editing old one as editing wasn’t working for me when I ran it.  
  
A screenshot of a computer

Description automatically generated

Fig 5  
As shown in figure I navigated to the browser and ran the edited file and copied the word ”good” displayed into the terminal. Still like last file it did not change the word for me this time as well.

A screenshot of a computer

Description automatically generated  
  
Fig 6  
  
As per as fig 6 I spawned a listener on another terminal to see if it would any connections were made but I wasn’t successful.  
  
Tried redoing the lab changed a few things but still it didn’t work.

## Objective 2 – Questions:

1. What is Server-Side Template Injection?

When untrusted user input is embedded unsanitized within a server-side template, an attacker can inject malicious code that gets executed on the server. This can lead to compromising sensitive data or even taking over the server.

2. What technologies or frameworks are likely vulnerable to SSTI? (Direct quote is allowed here.)  
Any web application using server-side templating engines without proper input validation is susceptible to SSTI. Here are few examples:

* Apache Thymeleaf (Java)
* Jinja2 (Python)
* Twig (PHP)
* Velocity (Java)
* FreeMarker (multiple languages)
* Blade (Laravel - PHP)

3. What are at least six server side templating technologies?

Here are the 6 server-side templating technlogies:

* Jinja2 (Python)
* Ruby ERB(Embedded Ruby)
* Twig (PHP)
* Velocity (Java)
* FreeMarker (multiple languages)
* Apache Thymeleaf (Java)
* Mako (Python)

4. A non-native template is needed to execute a successful SSTI. (T or F)

False. SSTI exploits the native syntax of the server-side template engine being used. The attacker injects malicious code understandable by that specific engine.

5. An attacker could use SSTI to establish a Reverse Shell on a target server. (T or F)

True. A successful SSTI attack can grant code execution on the server. Attackers can leverage this to establish a reverse shell, giving them remote access and control.

6. What is Pastejacking?

An attacker replaces the content of a legitimate web paste with malicious code. When someone visits the compromised paste link, the malicious code executes on their system.

7. Pastesjacking can't be used to establish a Reverse Shell. (T or F)

False. Pastejacking typically exploits client-side vulnerabilities. While it can steal information or redirect users, it usually cannot establish a reverse shell on the target server.

8. What is Server-Side Request Forgery (SSRF)?

An attacker tricks a vulnerable server into making unauthorized HTTP requests to an external resource. This can be used for internal network reconnaissance, data exfiltration, or even pivoting to other systems.

9. What is a Host Header? What are it's use cases?

The Host header in an HTTP request specifies the intended server within a virtual hosting environment. It helps identify the appropriate web application to serve the request. Use cases include:

* Hosting multiple websites on a single server
* Load balancing traffic across multiple servers

10. What is Deserialization?

The process of converting a serialized data stream back into its original object structure. Insecure deserialization can lead to Remote Code Execution (RCE) if an attacker can inject malicious code into the serialized data.