# PortSwigger Information Disclosure Lab Notes

1. Information disclosure in error messages

This lab's verbose error messages reveal that it is using a vulnerable version of a third-party framework. To solve the lab, obtain and submit the version number of this framework.

 Solution

1. With Burp running, open one of the product pages.
2. In Burp, go to "Proxy" > "HTTP history" and notice that the GET request for product pages contains a productID parameter. Send the GET /product?productId=1 request to Burp Repeater. Note that your productId might be different depending on which product page you loaded.
3. In Burp Repeater, change the value of the productId parameter to a non-integer data type, such as a string. Send the request:

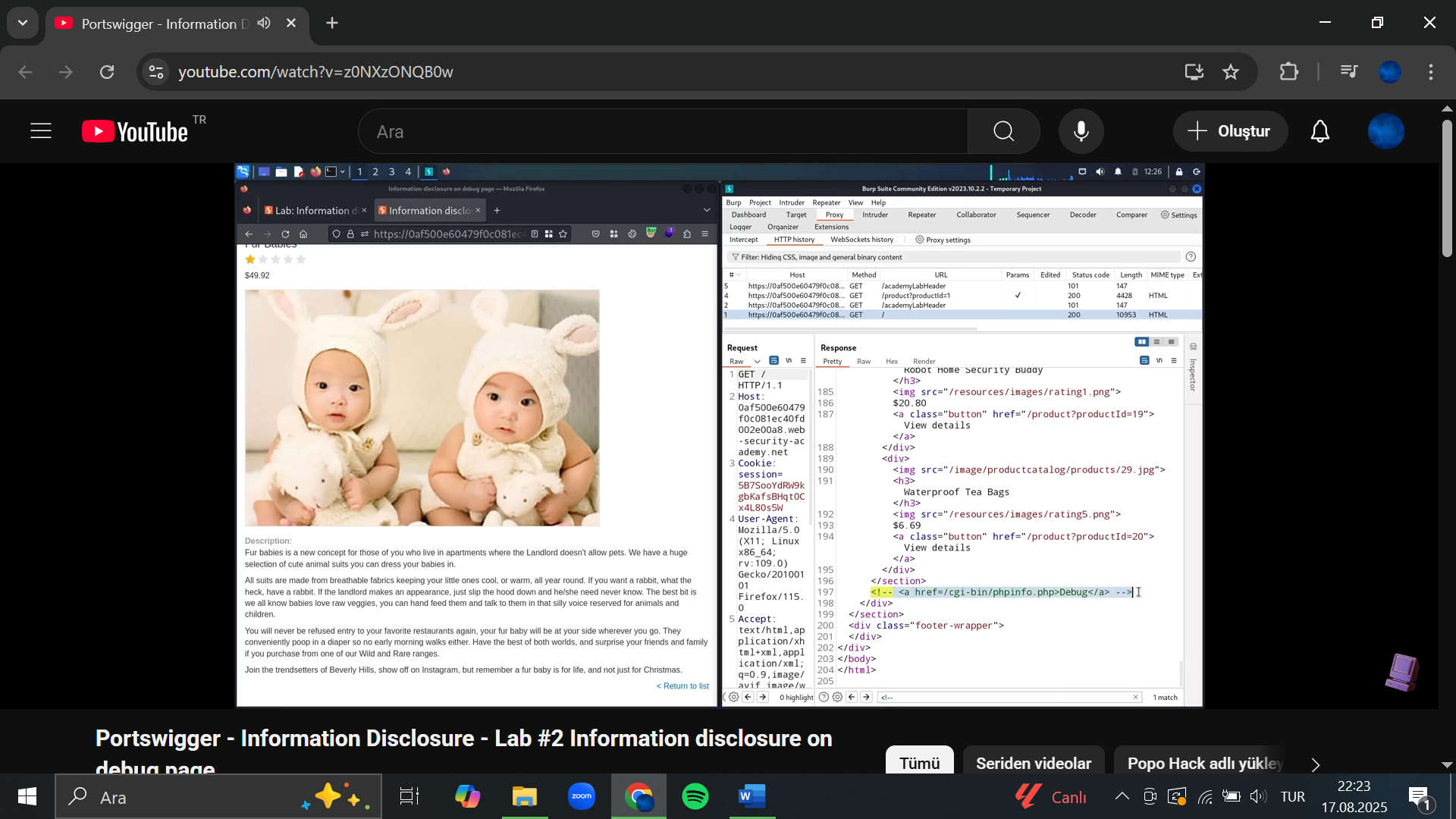
***GET /product?productId="example"***

1. The unexpected data type causes an exception, and a full stack trace is displayed in the response. This reveals that the lab is using Apache Struts 2 2.3.31.
2. Go back to the lab, click "Submit solution", and enter **2 2.3.31** to solve the lab.
3. Information disclosure on debug page

This lab contains a debug page that discloses sensitive information about the application. To solve the lab, obtain and submit the SECRET\_KEY environment variable.

 Solution

1. With Burp running, browse to the home page.
2. Go to the "Target" > "Site Map" tab. Right-click on the top-level entry for the lab and select "Engagement tools" > "Find comments". Notice that the home page contains an HTML comment that contains a link called "Debug". This points to /cgi-bin/phpinfo.php.
3. In the site map, right-click on the entry for /cgi-bin/phpinfo.php and select "Send to Repeater".



1. In Burp Repeater, send the request to retrieve the file. Notice that it reveals various debugging information, including the SECRET\_KEY environment variable.
2. Go back to the lab, click "Submit solution", and enter the SECRET\_KEY to solve the lab.
3. Source code disclosure via backup files

This lab leaks its source code via backup files in a hidden directory. To solve the lab, identify and submit the database password, which is hard-coded in the leaked source code.

 Solution

1. Browse to /robots.txt and notice that it reveals the existence of a /backup directory. Browse to /backup to find the file ProductTemplate.java.bak. Alternatively, right-click on the lab in the site map and go to "Engagement tools" > "Discover content". Then, launch a content discovery session to discover the /backup directory and its contents.

A screenshot of a computer

AI-generated content may be incorrect.

1. Browse to /backup/ProductTemplate.java.bak to access the source code.

A screenshot of a computer

AI-generated content may be incorrect.

1. In the source code, notice that the connection builder contains the hard-coded password for a Postgres database.
2. Go back to the lab, click "Submit solution", and enter the database password to solve the lab.
3. **Authentication bypass via information disclosure**

This lab's administration interface has an authentication bypass vulnerability, but it is impractical to exploit without knowledge of a custom HTTP header used by the front-end.

To solve the lab, obtain the header name then use it to bypass the lab's authentication. Access the admin interface and delete the user carlos.

You can log in to your own account using the following credentials: wiener:peter

 Solution

1. In Burp Repeater, browse to GET /admin. The response discloses that the admin panel is only accessible if logged in as an administrator, or if requested from a local IP.
2. Send the request again, but this time use the TRACE method:

***TRACE /admin***

A screenshot of a computer

AI-generated content may be incorrect.

Key points about TRACE:

* When a client sends a TRACE request to a server, the server responds by echoing back the exact request message it received.
* This helps the client verify that:
  + The request is not being modified by proxies, gateways, or other intermediaries.
  + Headers and other request data are arriving as expected.
* It is mainly a diagnostic tool, not used in normal web applications.

1. Study the response. Notice that the ***X-Custom-IP-Authorization*** header, containing your IP address, was automatically appended to your request. This is used to determine whether the request came from the localhost IP address.
2. Go to **Proxy > Match and replace**.
3. Under **HTTP match and replace rules**, click **Add**. The **Add match/replace rule** dialog opens.
4. Leave the **Match** field empty.
5. Under **Type**, make sure that **Request header** is selected.
6. In the **Replace** field, enter the following:

***X-Custom-IP-Authorization: 127.0.0.1***

1. Click **Test**.
2. Under **Auto-modified request**, notice that Burp has added ***the X-Custom-IP-Authorization*** header to the modified request.
3. Click **OK**. Burp Proxy now adds the X-Custom-IP-Authorization header to every request you send.
4. Browse to the home page. Notice that you now have access to the admin panel, where you can delete carlos.

Popo Hack solution after step 3:

**4.Exploit the header**

* 1. Added this header manually and set it to localhost:
  2. X-Custom-IP-Authorization: 127.0.0.1
  3. Server now believed the request was coming from localhost → authentication bypassed.

1. **Delete Carlos**
   1. Once inside the admin interface, sent a request to delete user *carlos* → lab solved.
2. Information disclosure in version control history

This lab discloses sensitive information via its version control history. To solve the lab, obtain the password for the administrator user then log in and delete the user carlos.

 Solution

1. Open the lab and browse to /.git to reveal the lab's Git version control data.
2. Download a copy of this entire directory. For Linux users, the easiest way to do this is using the command:

***wget -r https://YOUR-LAB-ID.web-security-academy.net/.git/***

Windows users will need to find an alternative method, or install a UNIX-like environment, such as Cygwin, in order to use this command.

1. Explore the downloaded directory using your local Git installation. Notice that there is a commit with the message ***"Remove admin password from config".***
2. Look closer at the diff for the changed admin.conf file. Notice that the commit replaced the hard-coded admin password with an environment variable ADMIN\_PASSWORD instead. However, the hard-coded password is still clearly visible in the diff.
3. Go back to the lab and log in to the administrator account using the leaked password.
4. To solve the lab, open the admin interface and delete carlos.