

**Capstone project**

Identify a website susceptible to Parameter Tampering, manipulate the request/query, and produce a comprehensive report with accompanying screenshots and step-by-step instructions on the execution of

Created by

Vivek.k

**Index**

* Introduction
* Research
* Data collection
* Exploitation
* Impact analysis
* Mitigation
* Proof of concept
* Conclusion
* Reference

**Introduction**

**Cyber security** is the application of technologies, processes, and controls to protect systems, networks, programs, devices and data from cyber-attacks. In today’s interconnected world, where information is the new currency, safeguarding digital assets has become paramount from individuals to multinational corporations; everyone is a potential target for cyberattacks.

Why cybersecurity is important?

**Protection of Data**: It ensures that sensitive data such as personal information, financial details, and corrupted by unauthorized entities.

**Prevention of Cyber Attacks**: Effective cybersecurity measures help defend against various types of cyber-attacks such as malware, phishing, ransomware, and denial-of-service attacks, which can disrupt operations and cause financial losses.

**Financial loss:** cybercriminals often demand ransom or cause significant financial damage through data breaches and system disruptions.

**Reputation damage:** A data breach can severely damage on organization’s reputation and trust.

**Disruption of operations:** cyberattacks can cripple critical infrastructure and business operations.

At technology continues to evolve, so do the threats. Cybersecurity encompasses a wide range of measures, including:

Data security

Network security

Application security

Identify and access management

Disaster recovery and business continuity

**Research/Reconnaissance**

Google Dorks: Also known as, google hacking or google searching, are advanced search queries used to find specific information on the internet that might not be easily accessible through standard search methods. They leverage Google’s search operators to uncover hidden or sensitive data.

For example, I used google dorks to find this website

Intex: shopping

Inurl: https

**About website**

**Name:** NUSKHE by paras

**URL:** <https://nuskhebyparas.com/>

**Category:** online shopping Beauty products

**Usage:** all of them are using who are knowledge about the online shopping.

**Popularity:** Good

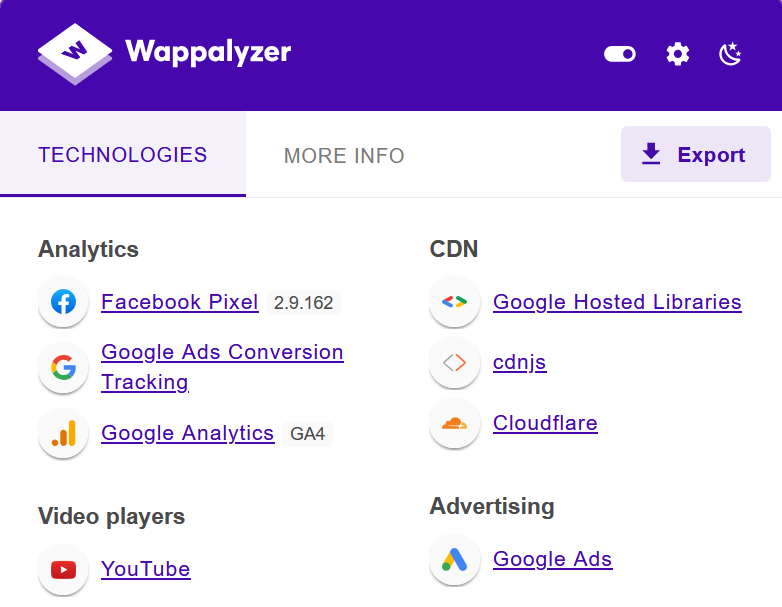
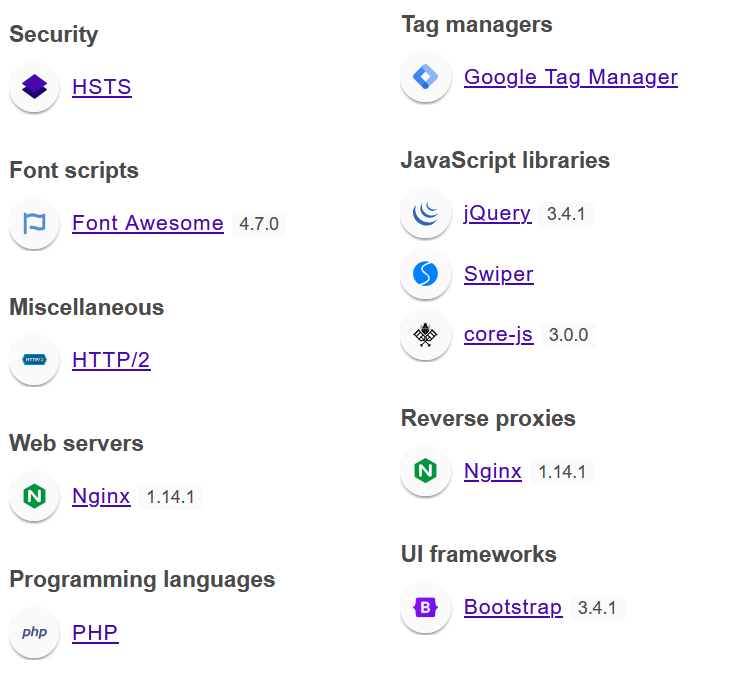
**Severity:** High

**Overall ranking:** 4/10

**Wappalyzer**

Used wappalyzer to identify the technologies used in this website like frontend, backend, web server, database CMS, etc.

It is a powerful toll that helps you uncover the technology used to build a website. It is like having a digital detective that unmasks the components behind any website you visit and **leads data provider that helps sales and marketing teams identify new prospects by the technologies they use**.

Analyzed every single research and found that it uses **Burp Suite** is an industry-standard tool for modern [security assessment](https://en.wikipedia.org/wiki/Information_technology_security_assessment) and [penetration testing](https://en.wikipedia.org/wiki/Penetration_test) of web applications.

**Burp Proxy and Interceptor**: Like other [web application security scanners](https://en.wikipedia.org/wiki/Dynamic_application_security_testing), one of the primary functionalities behind Burp Suite is its capability to act as a [proxy server](https://en.wikipedia.org/wiki/Proxy_server#Web_proxy_servers) for client-side HTTP requests. Penetration testers can intercept web servers' default HTTP requests variables (attributes, body parameters, cookies, headers) in real-time and edit these values.

In this website, identify the vulnerability and we can use the Burpsute capture and tampering the values or (manipulating). Let me explain how I done this in the next pages of the Document did. As mentioned previously the client’s e-commerce subdomain was vulnerable. In burp Suite, under the proxy tab which fills in as Man-in-the Middle attack has an option Intercept. To intercept a request. Explore to “Proxy Intercept” and Click on “Intercept if off When a request is intercepted, it very well may be seen in various ways. The choices accessible for a straightforward HTTP request are Raw, Headers, and Hex in view of the sort of request on burp suite. This request will contain host, cookie details, encoding, content-details and parameter such as currency, language, amount, quantity, product name, product id, purchase step, etc.

**Exploitation**

Parameter tampering is a type of exploitation technique where an attacker manipulates parameters exchanged between a client and a server in order to gain unauthorized access to resources or perform actions that are not intended by the application's designers. This can lead to security vulnerabilities and compromises if not properly mitigated. Here are some common types of exploitation techniques associated with parameter tampering:

1. **Changing Parameter Values**: Attackers modify parameters passed in HTTP requests to manipulate application behavior. For example, altering URL parameters, form fields, or hidden fields in web forms.
2. **Session Hijacking**: By manipulating session-related parameters (such as session IDs or tokens), attackers may attempt to take over an active session of an authenticated user.
3. **Hidden Field Manipulation**: Altering hidden fields in forms to inject unauthorized data or manipulate the flow of application logic.
4. **Cookie Manipulation**: Modifying cookie values (such as session cookies) to gain unauthorized access to user accounts or bypass authentication mechanisms.
5. **SQL Injection**: Although primarily targeting SQL queries, SQL injection can also involve parameter tampering when manipulating input parameters to execute unauthorized database queries.

**Impact**

1. **Data theft:** Attacker can access the sensitive information.
2. **Account takeover:** They can control of user accounts.
3. **System compromise:** They can unauthorized access to systems.
4. **Information Disclosure**: Attackers may exploit parameter-tampering vulnerabilities to access confidential information that should not be exposed, such as personal data, financial records, or proprietary business information.

**Mitigation**

**Implement strong authentication: user multi-factor authentication enforce complex passwords policies, and consider biometric authentication.**

**Input Validation**: Validate and sanitize all user-supplied input to prevent injection attacks and ensure that parameters are within expected ranges and formats.

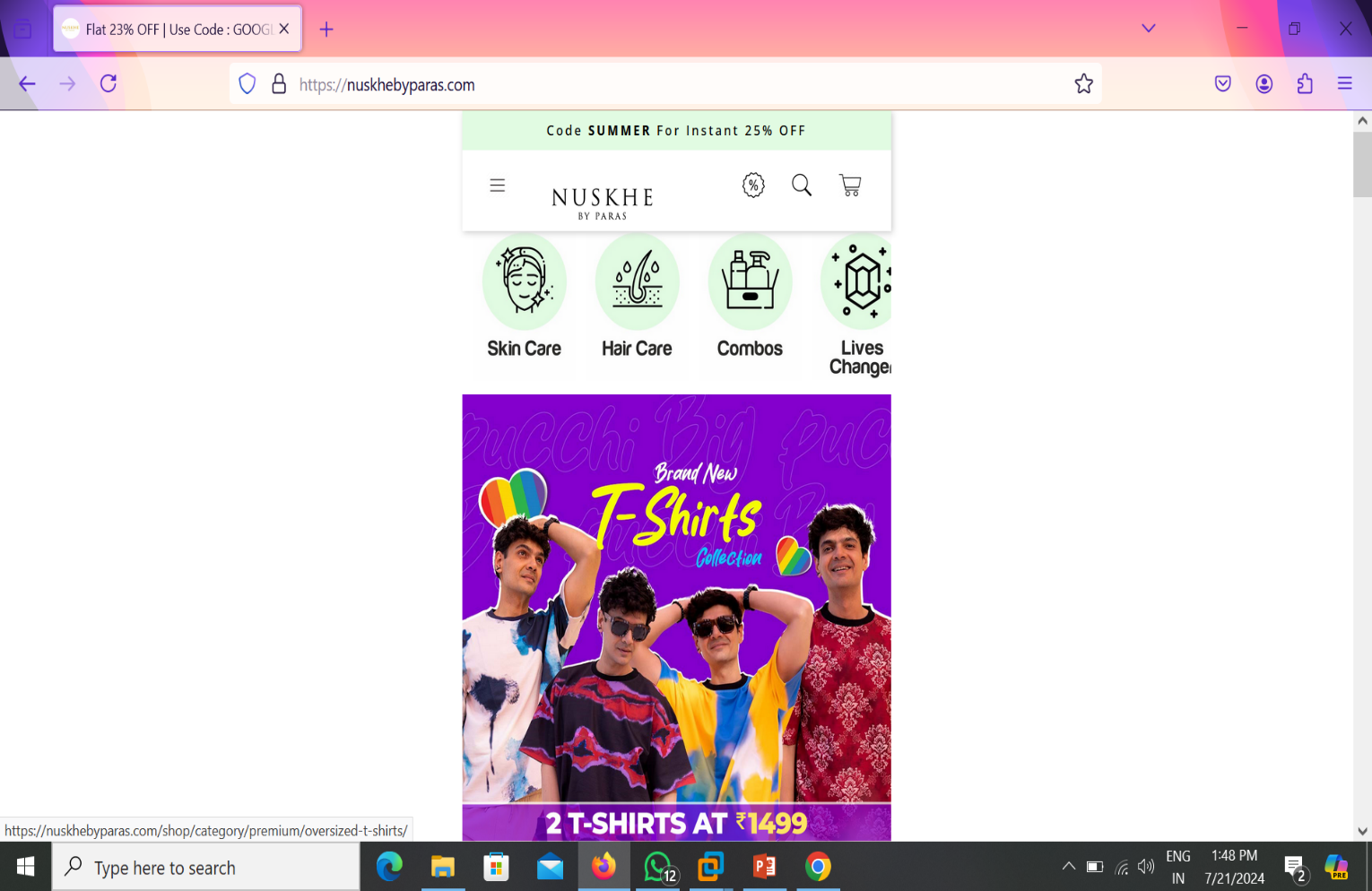
**Access Control**: Implement strong access control mechanisms to enforce least privilege principles and restrict access to sensitive functions and data based on user roles and permissions.

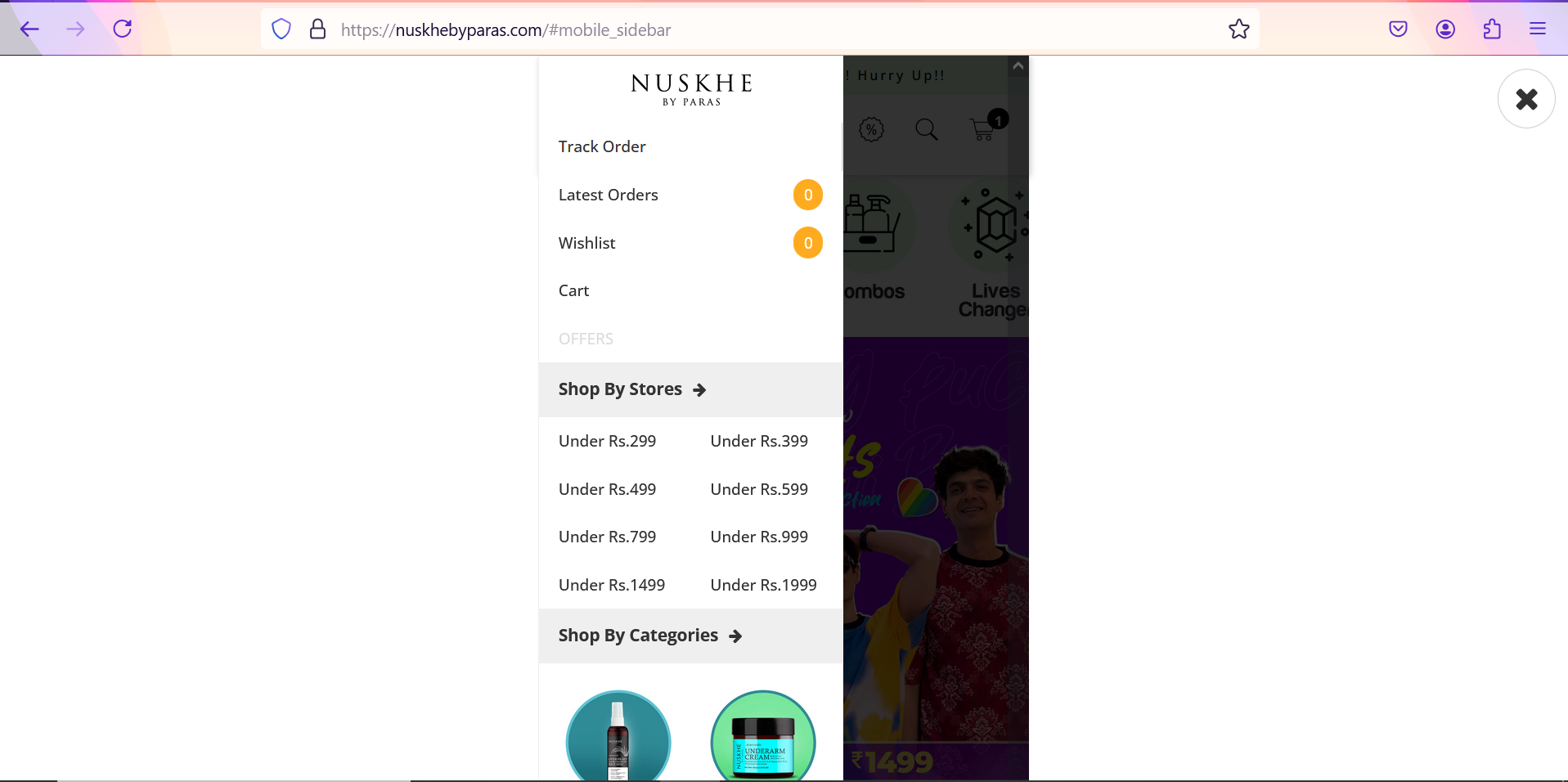
**Encryption**: Use strong encryption to protect sensitive data, both in transit and at rest, to mitigate the risk of data exposure even if parameters are tampered with.

**Regular Security Assessments**: Conduct regular security assessments, including vulnerability scanning and penetration testing, to identify and remediate parameter tampering vulnerabilities before they can be exploited by attackers.

**Proof of concept**

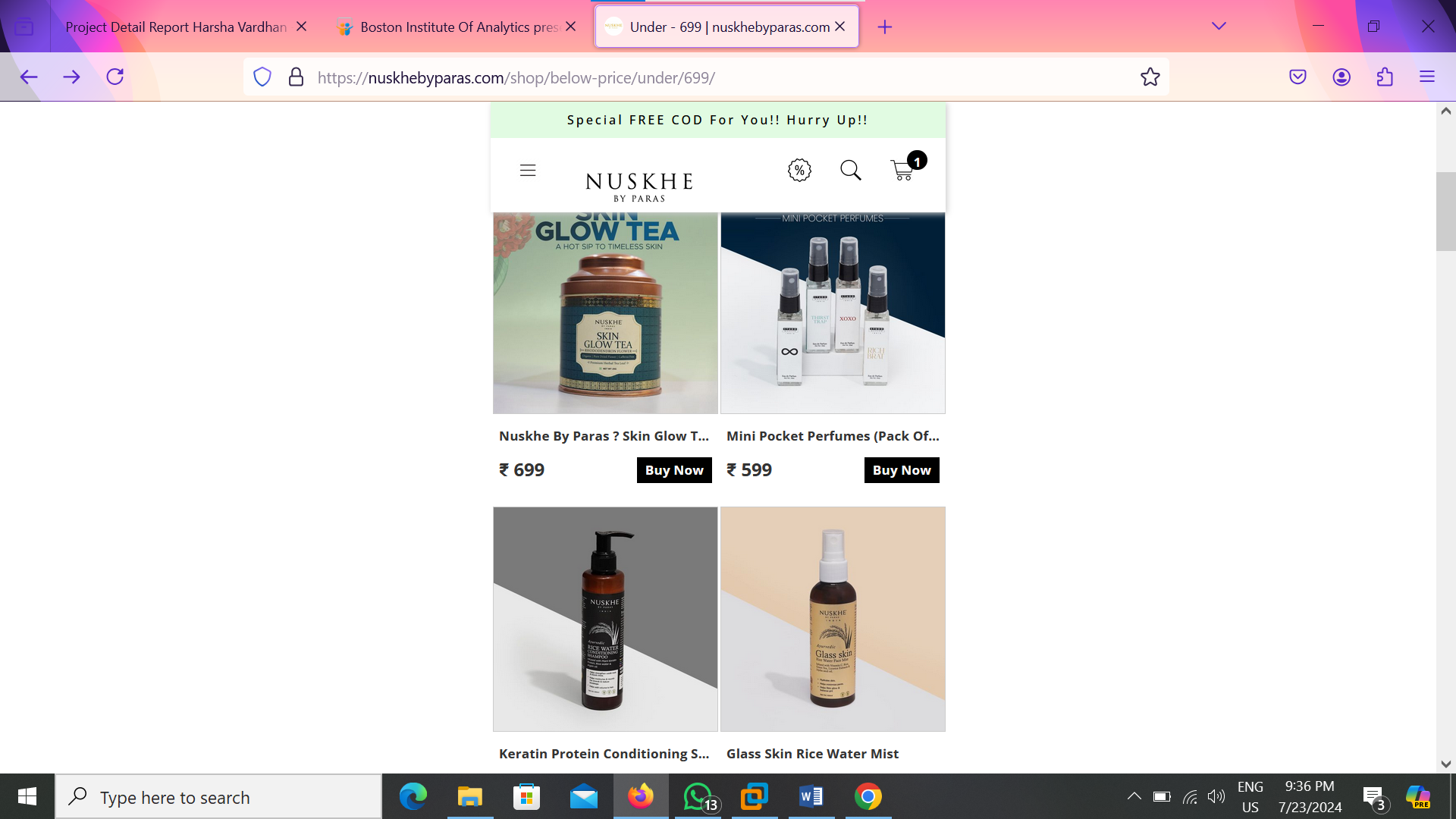
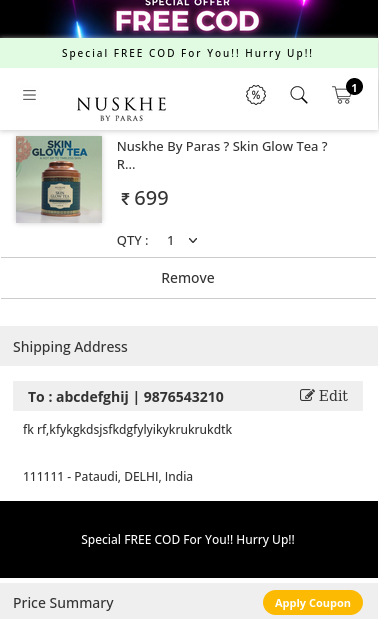
Open a website and navigate to the products page.

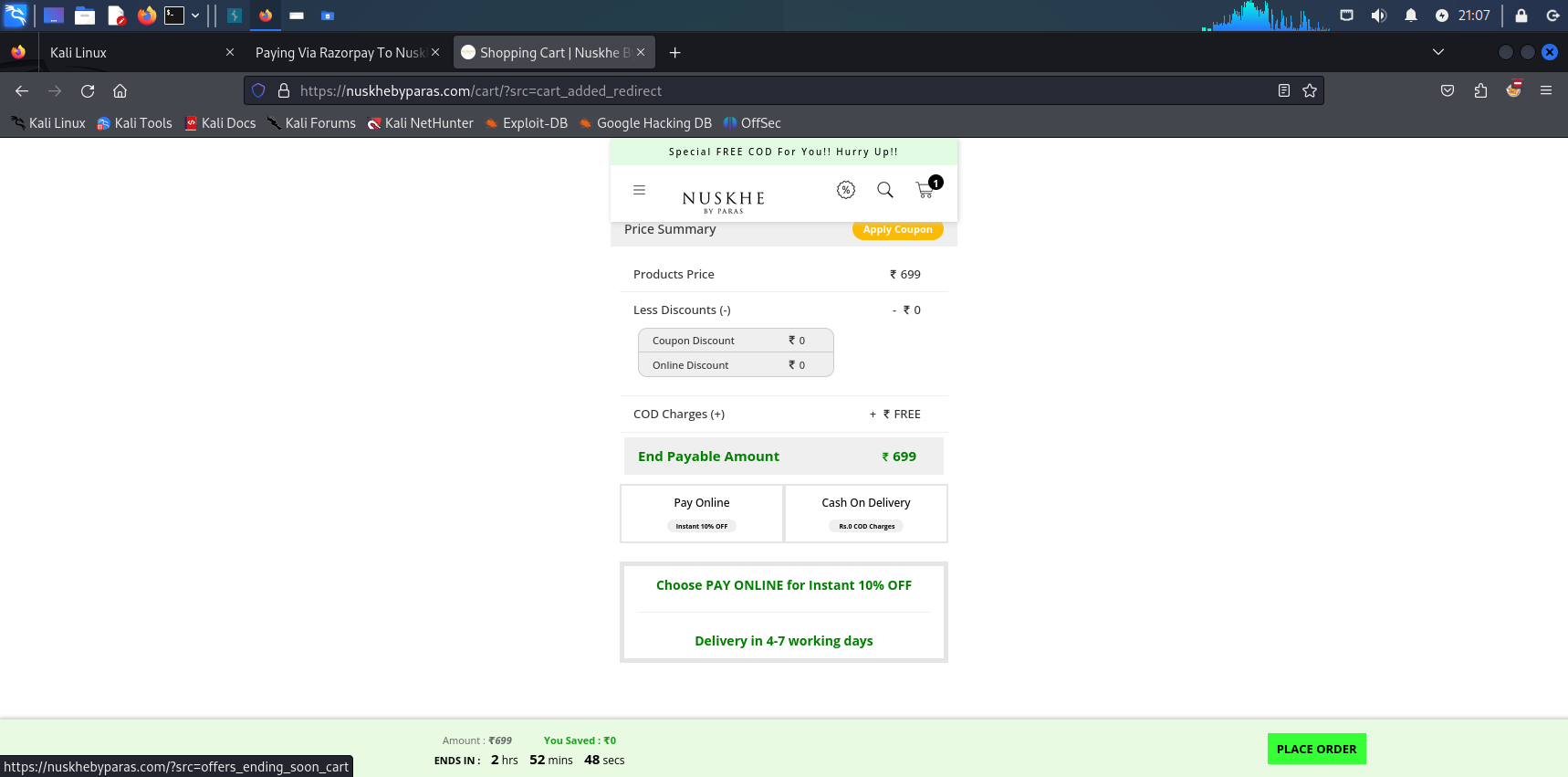
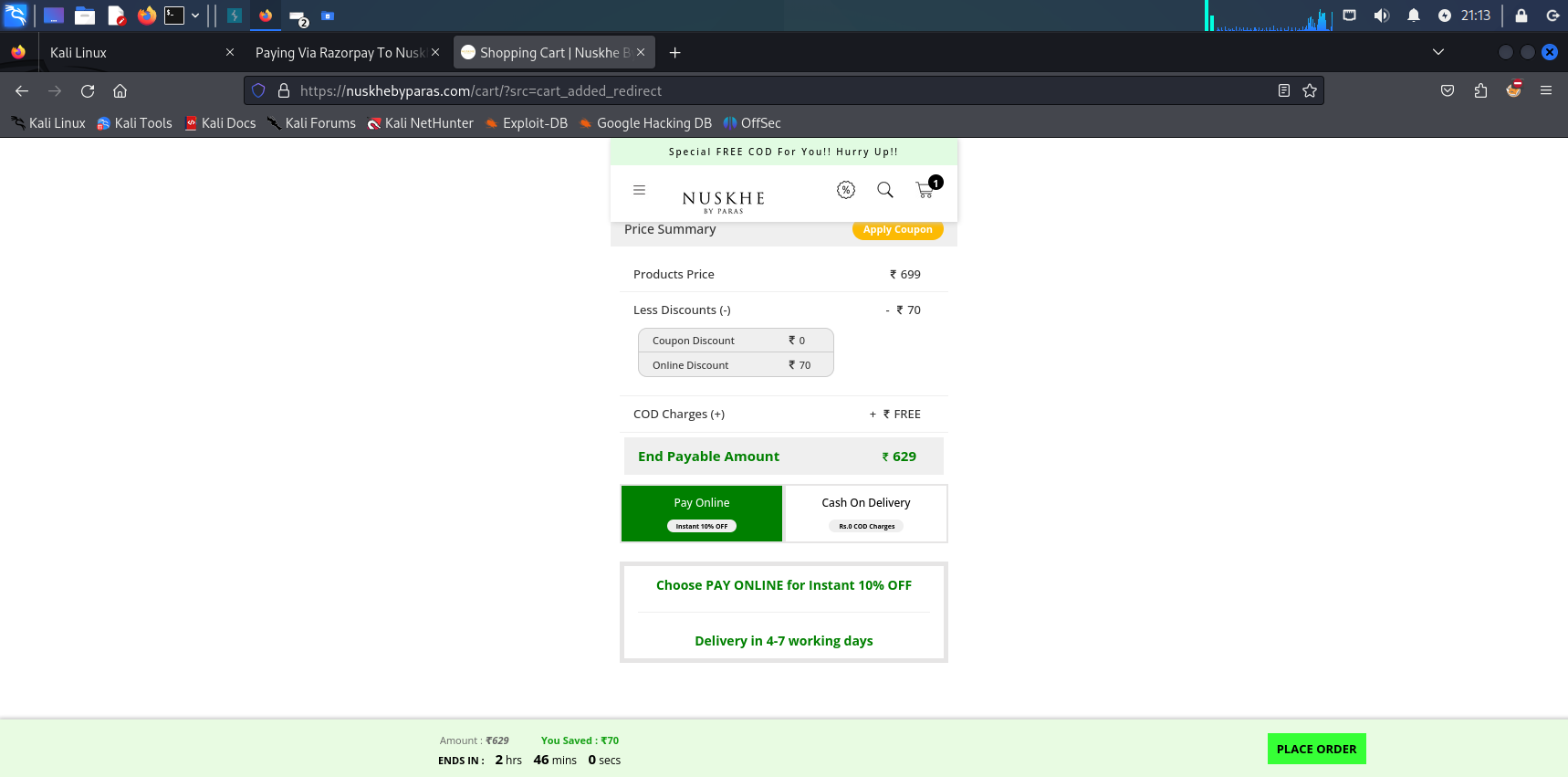
****



Now we are in products page.

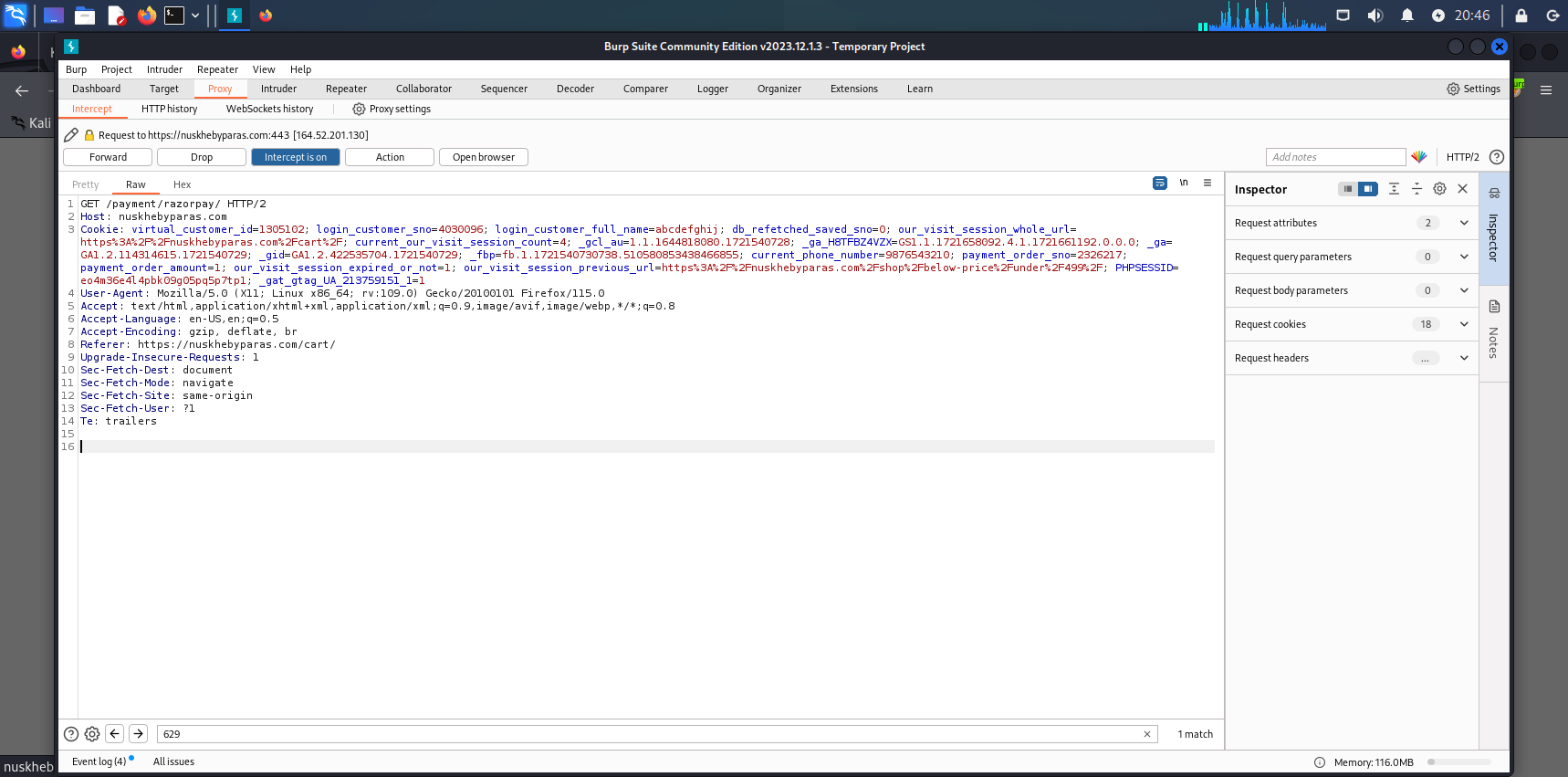
In this page, we are selecting the product, as you like.

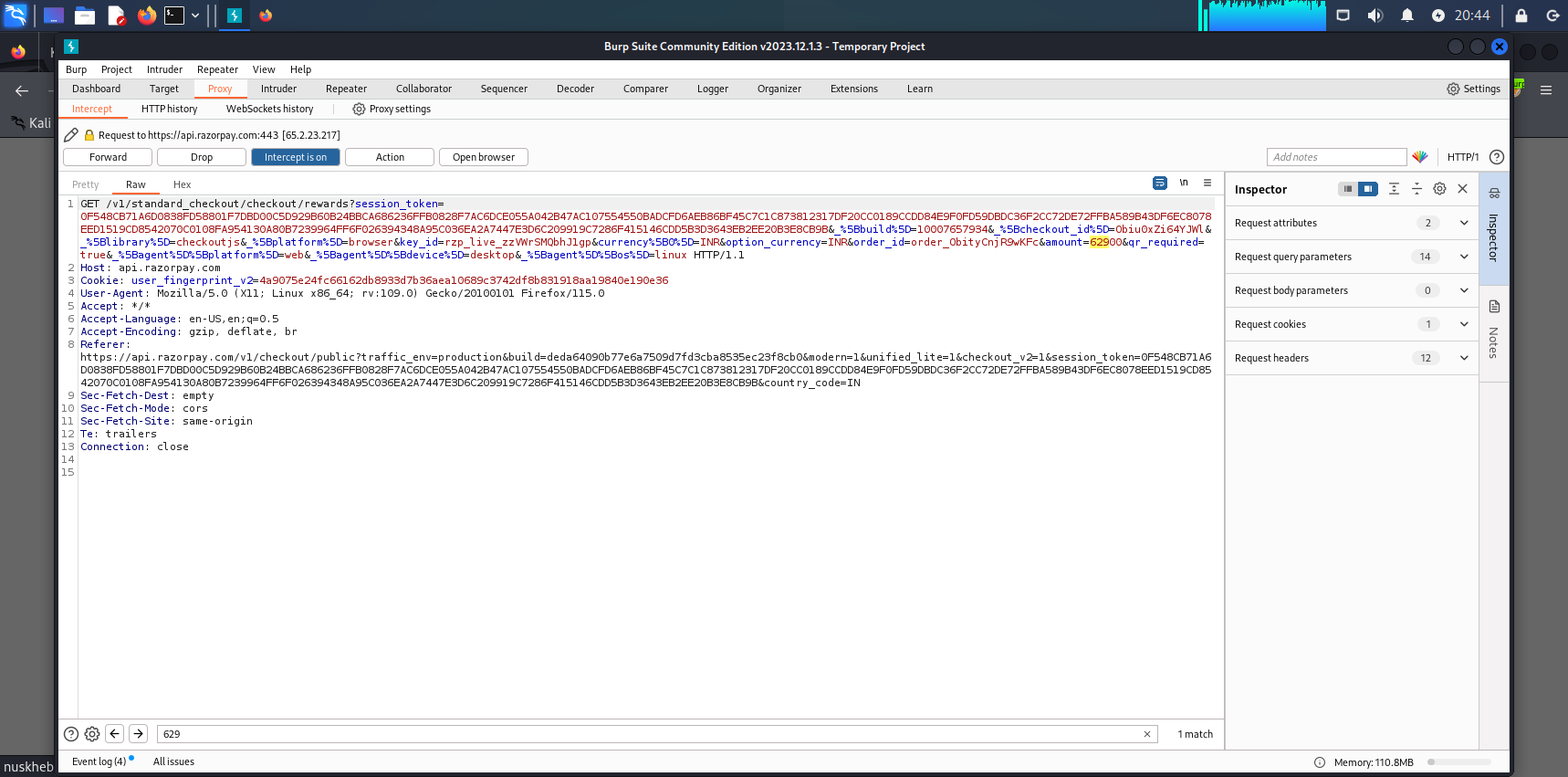
We can select a product and add to cart then enter the address, e-mail, mobile no etc.

Here we should use the Burp suite (proxy network) the intercept should be On to capture the values.



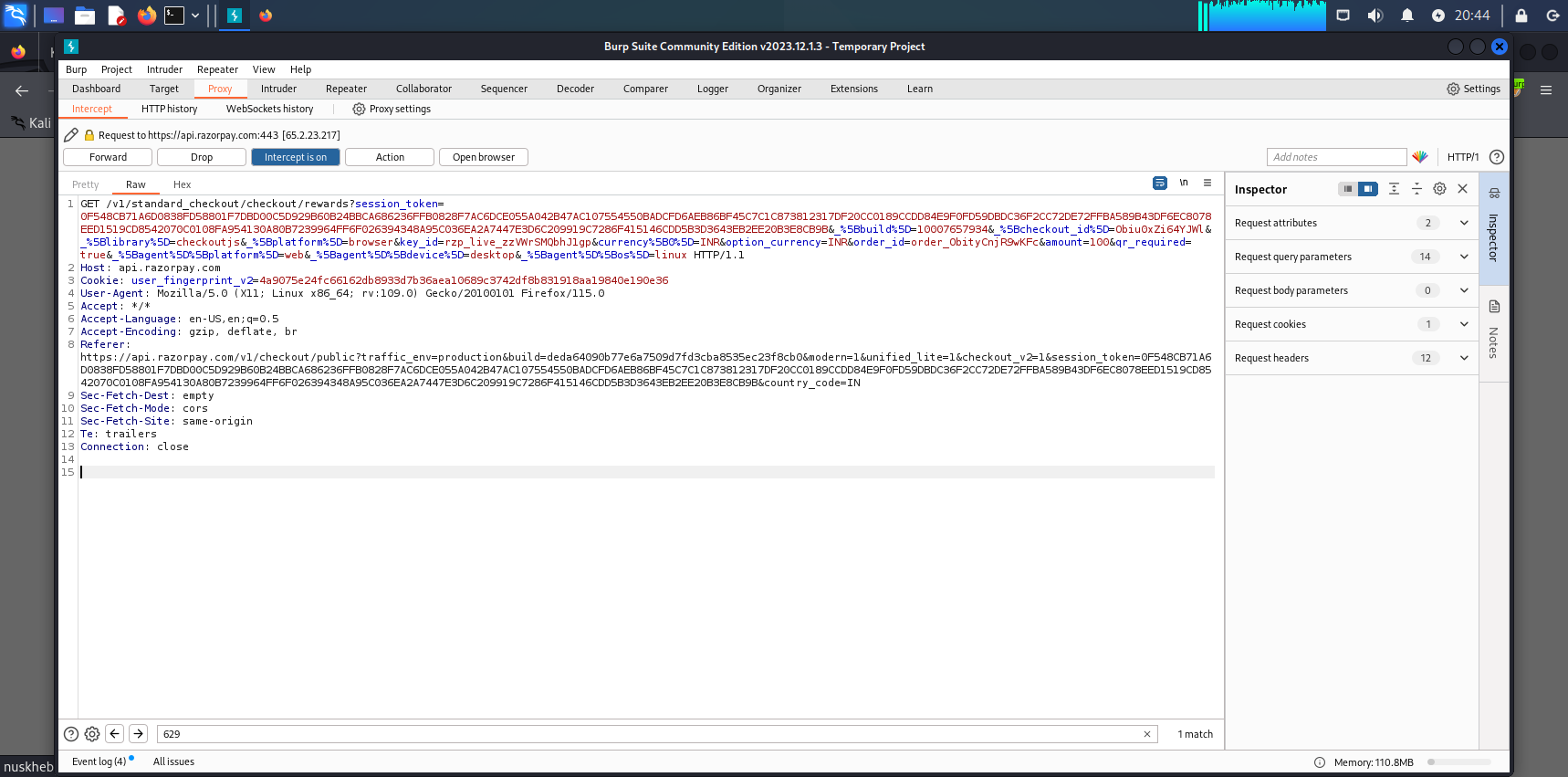
If we capture the values and that values should be forward.

Well use the Burp suite we can change or edit the prize values.

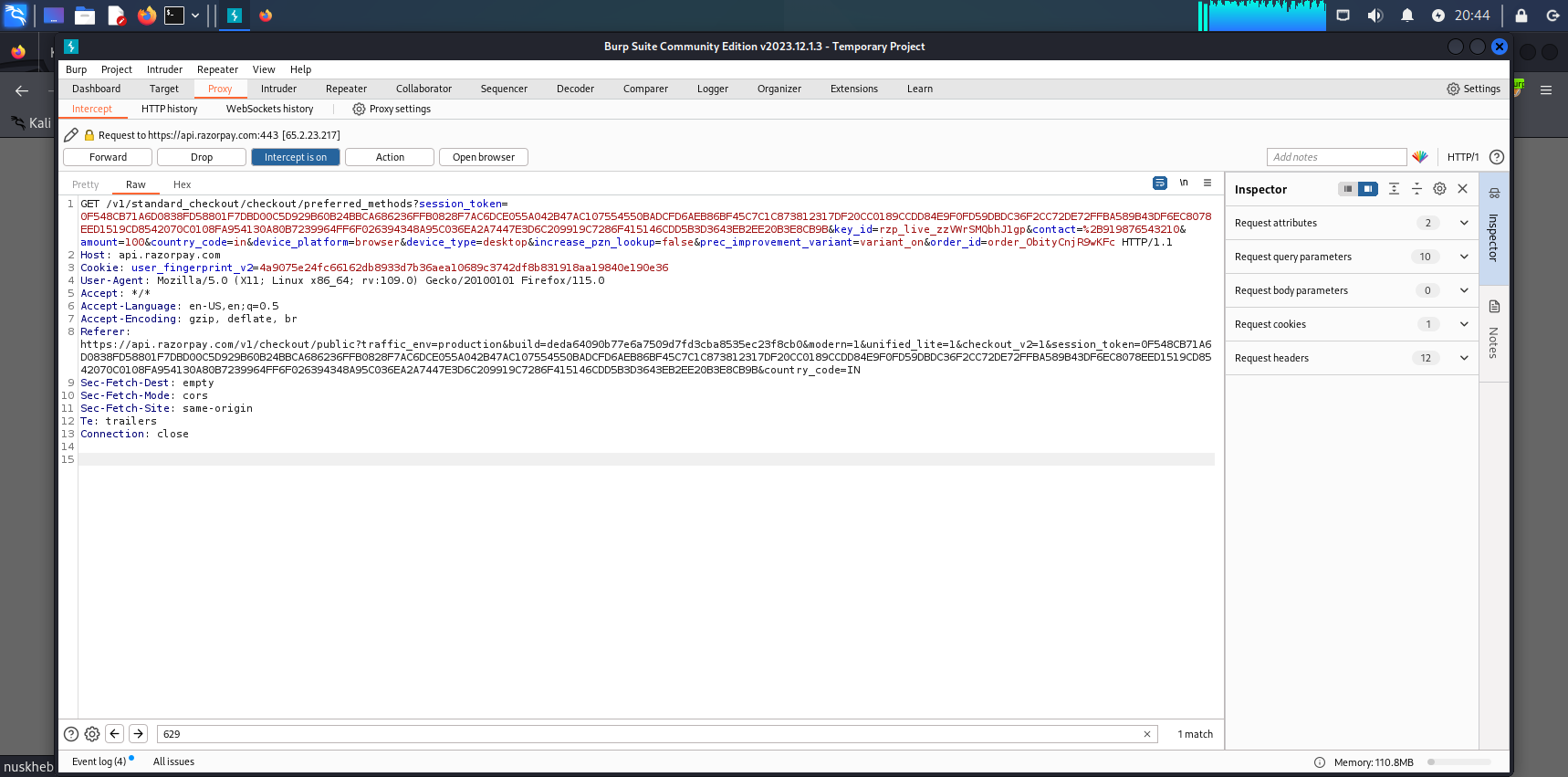


Now we can also see the prize value is 629rs.

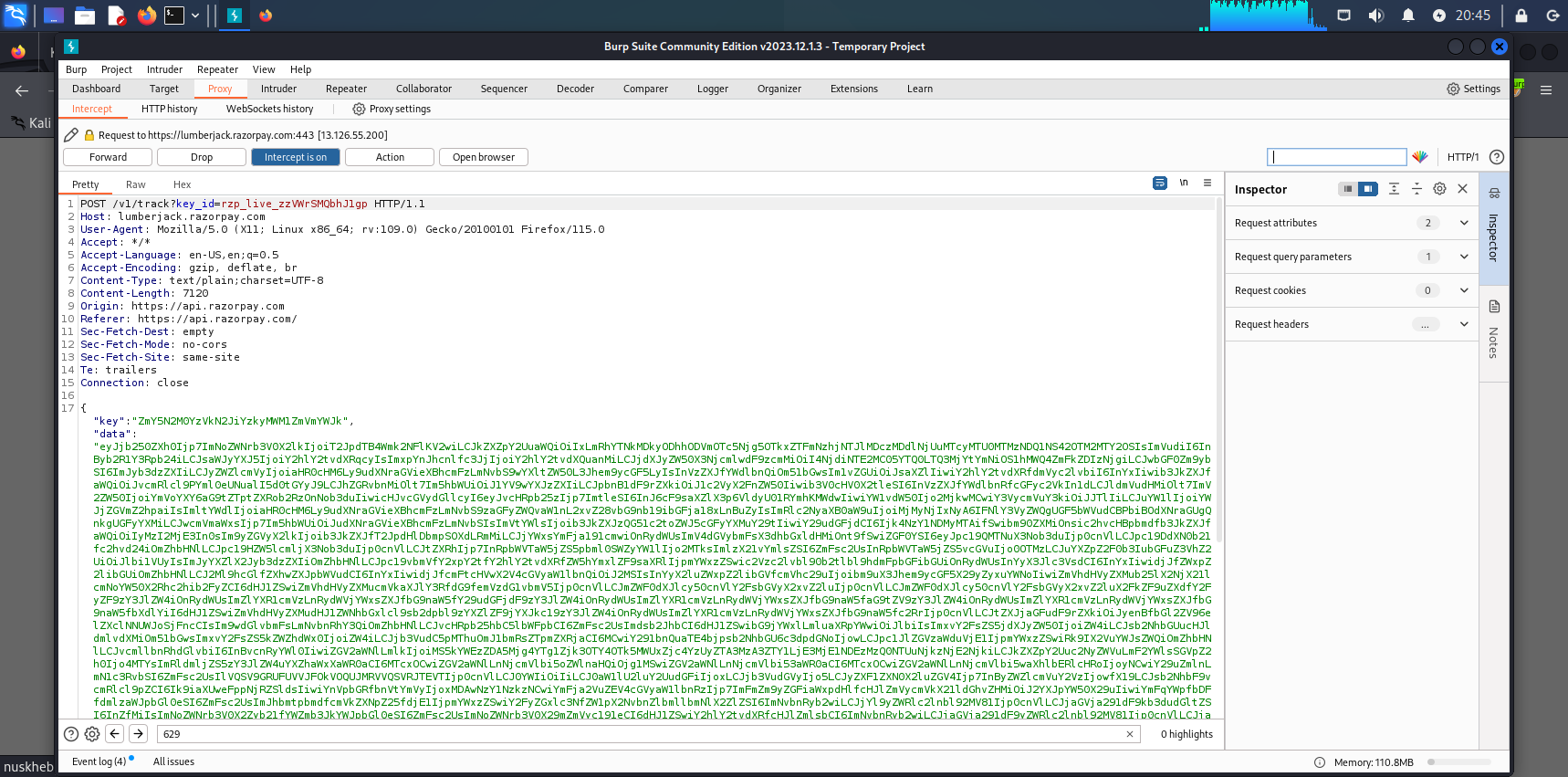
When we can see the amount prize 629rs it changes to 1rs. In addition, where we can see the 629rs it should be changes to **one** rupees.

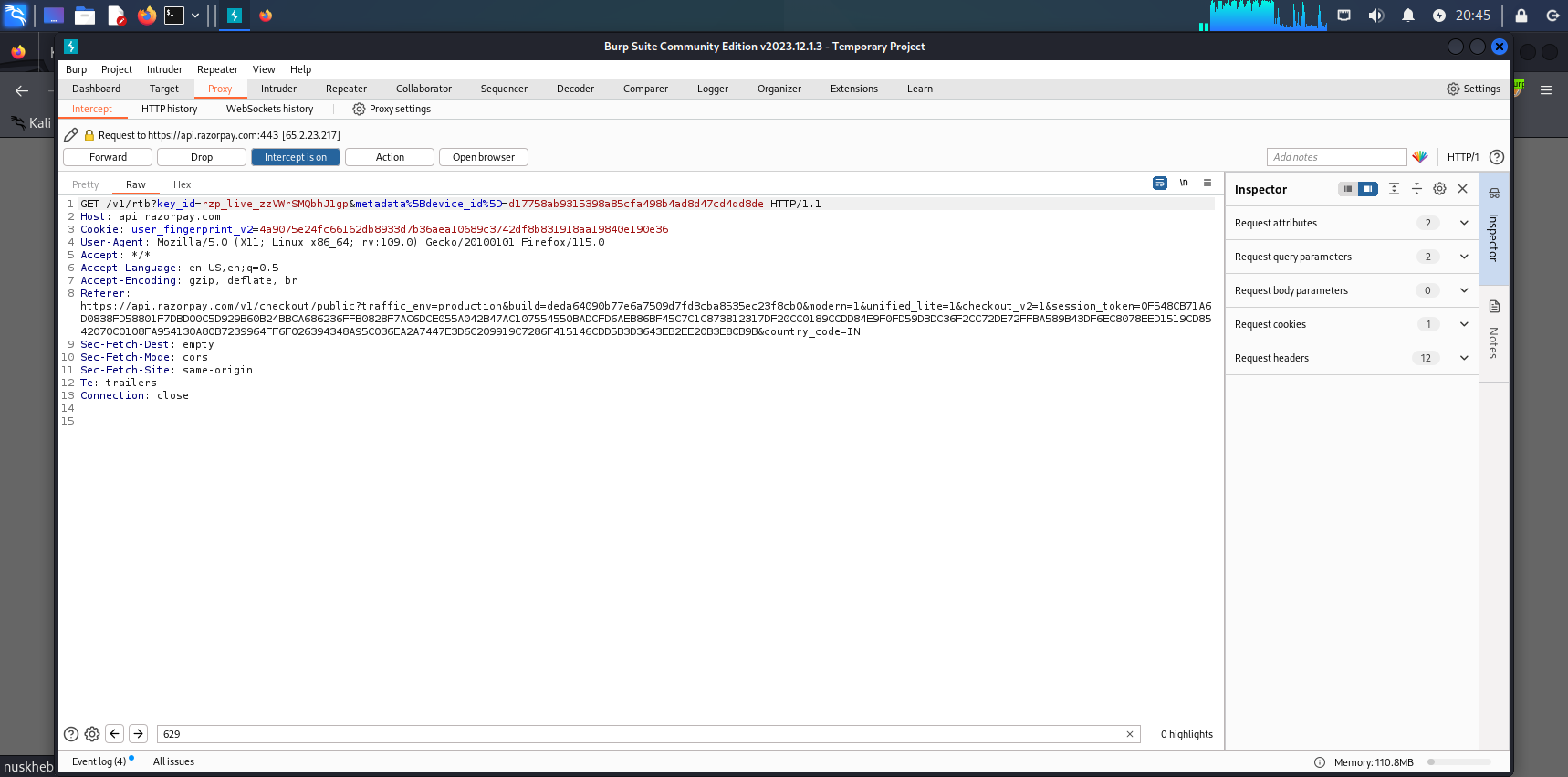


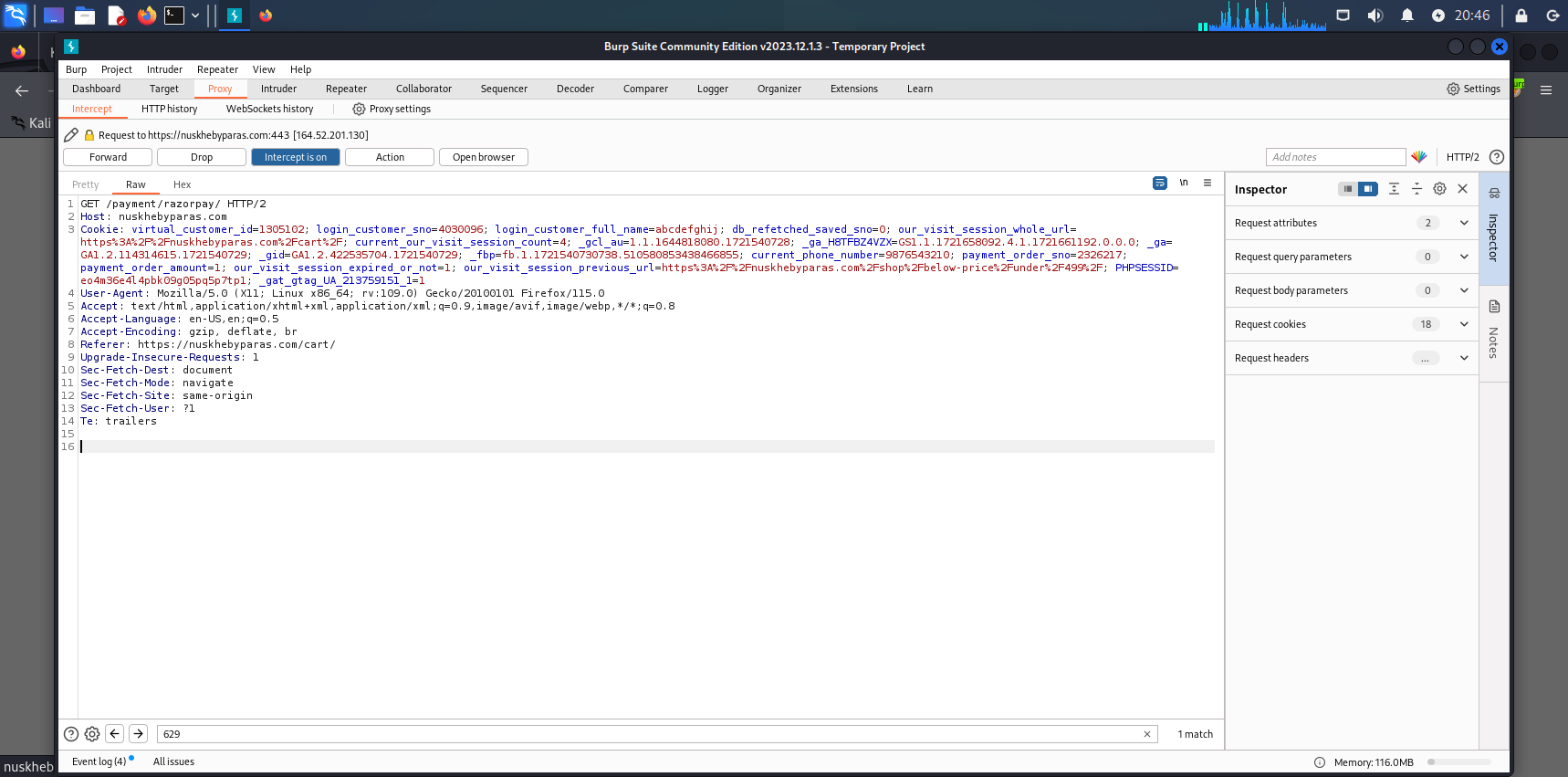
Here also see we are change the values.

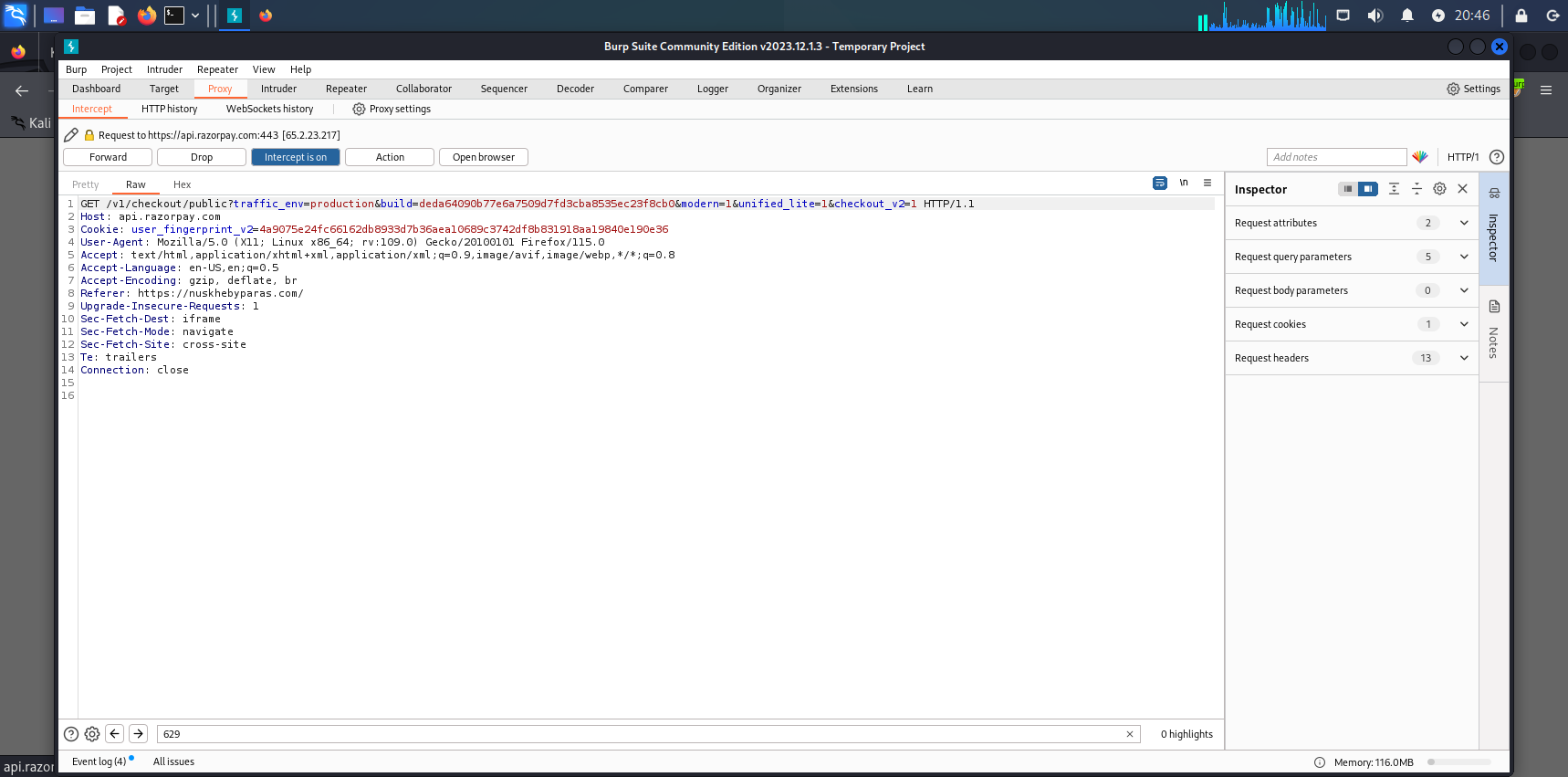


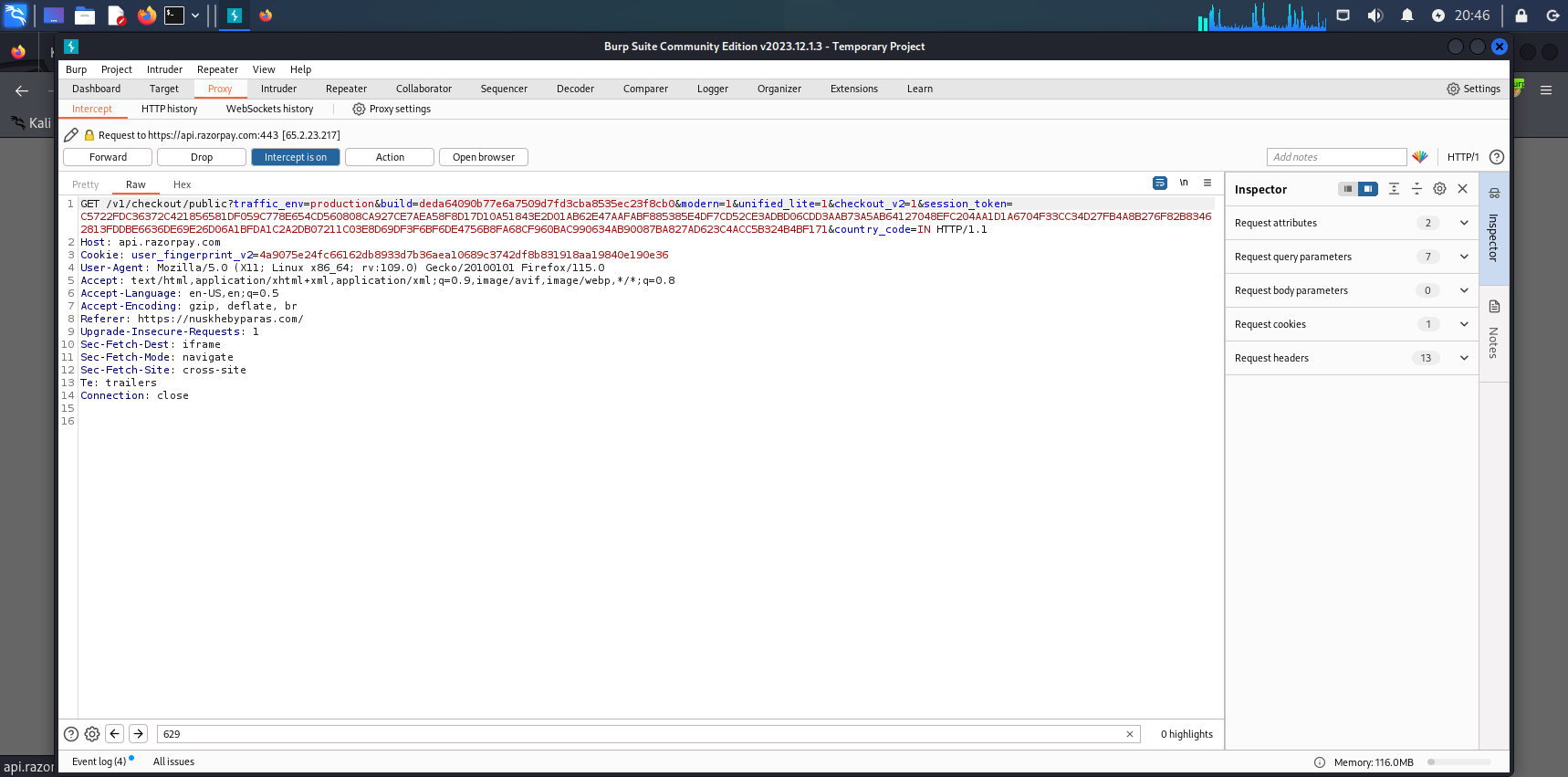
Then we continuously forwarded the values and the proxy changes the values we can see examples





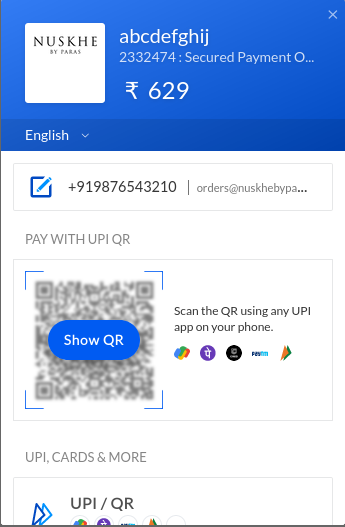
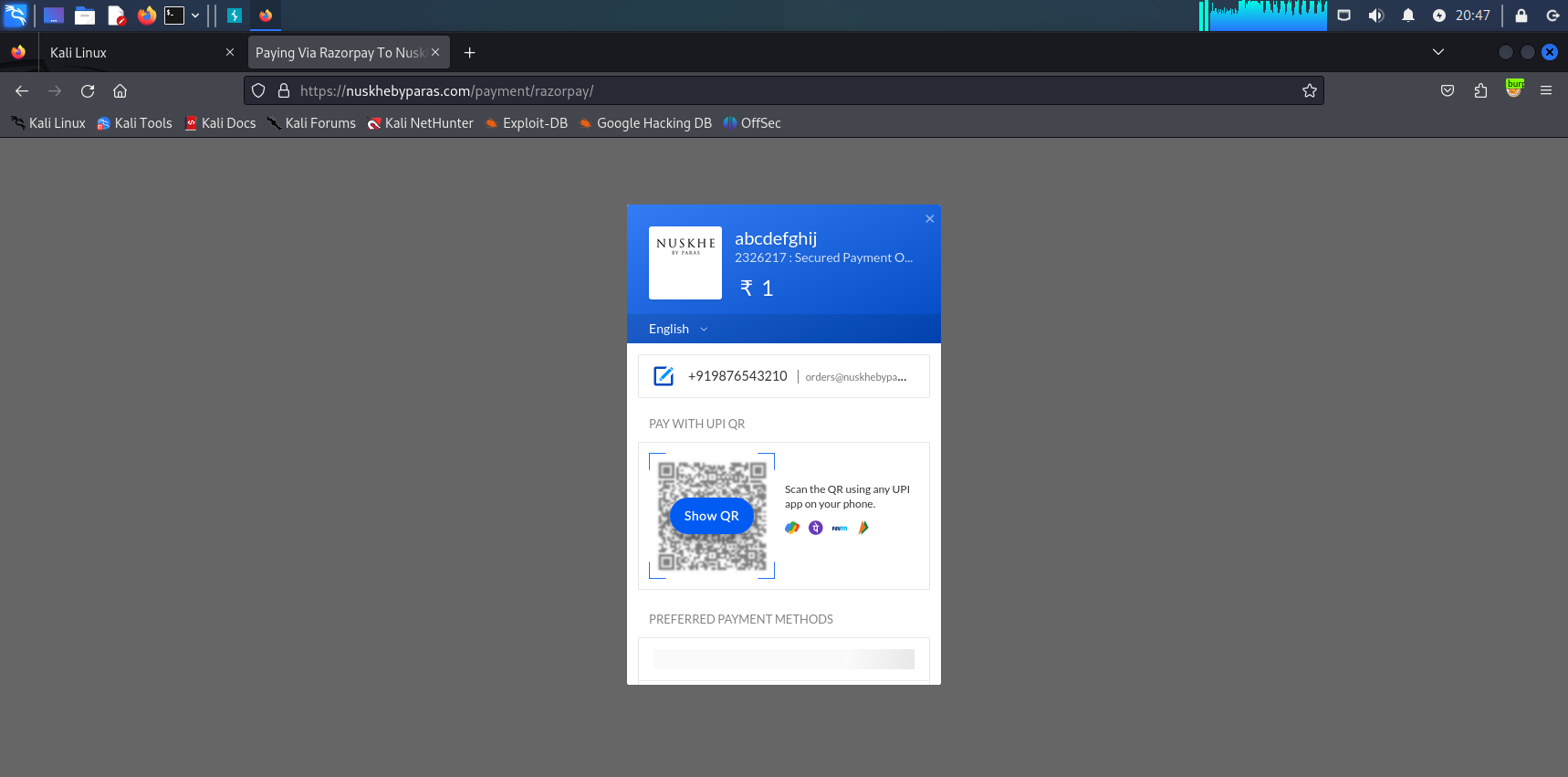


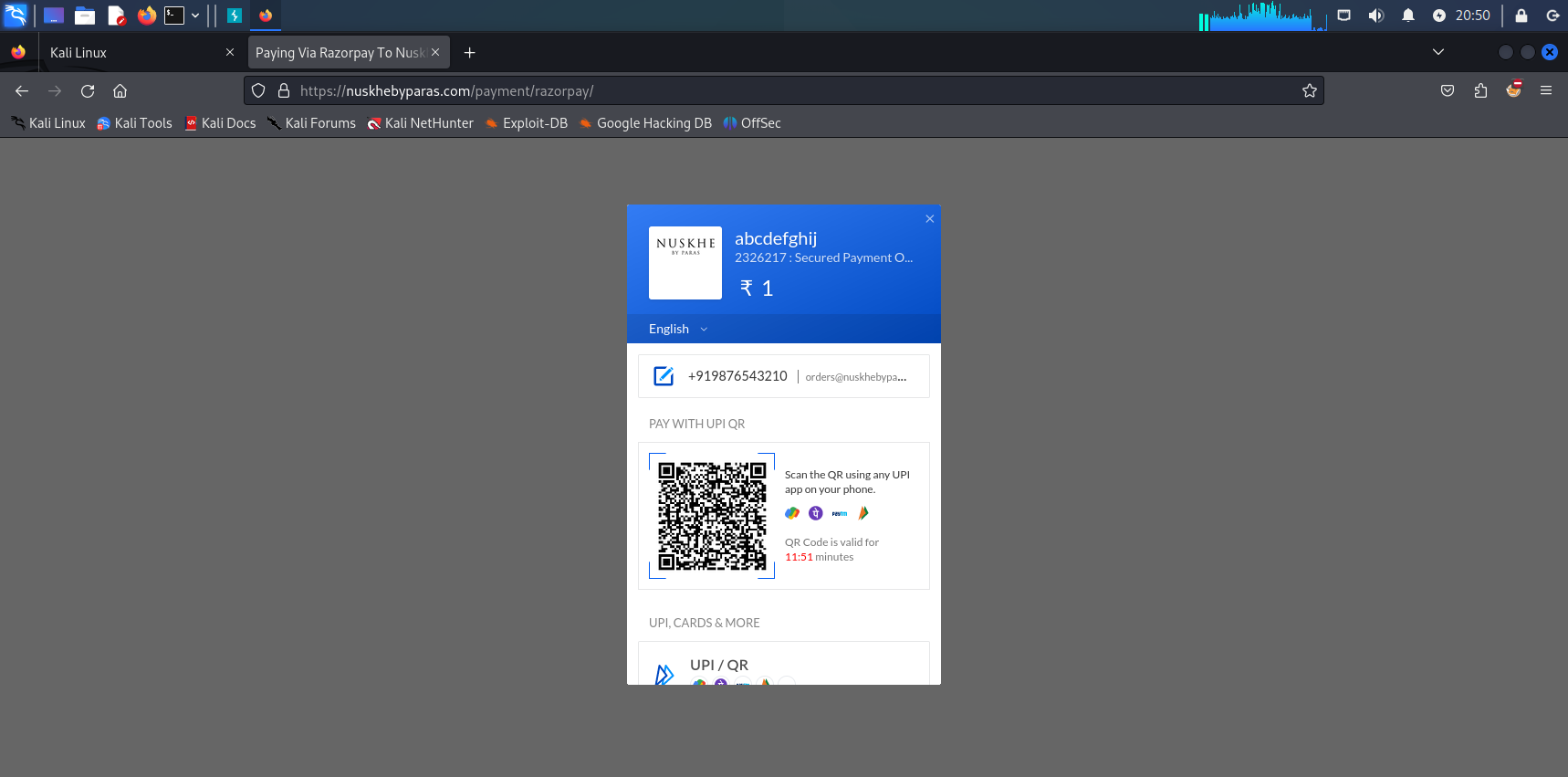
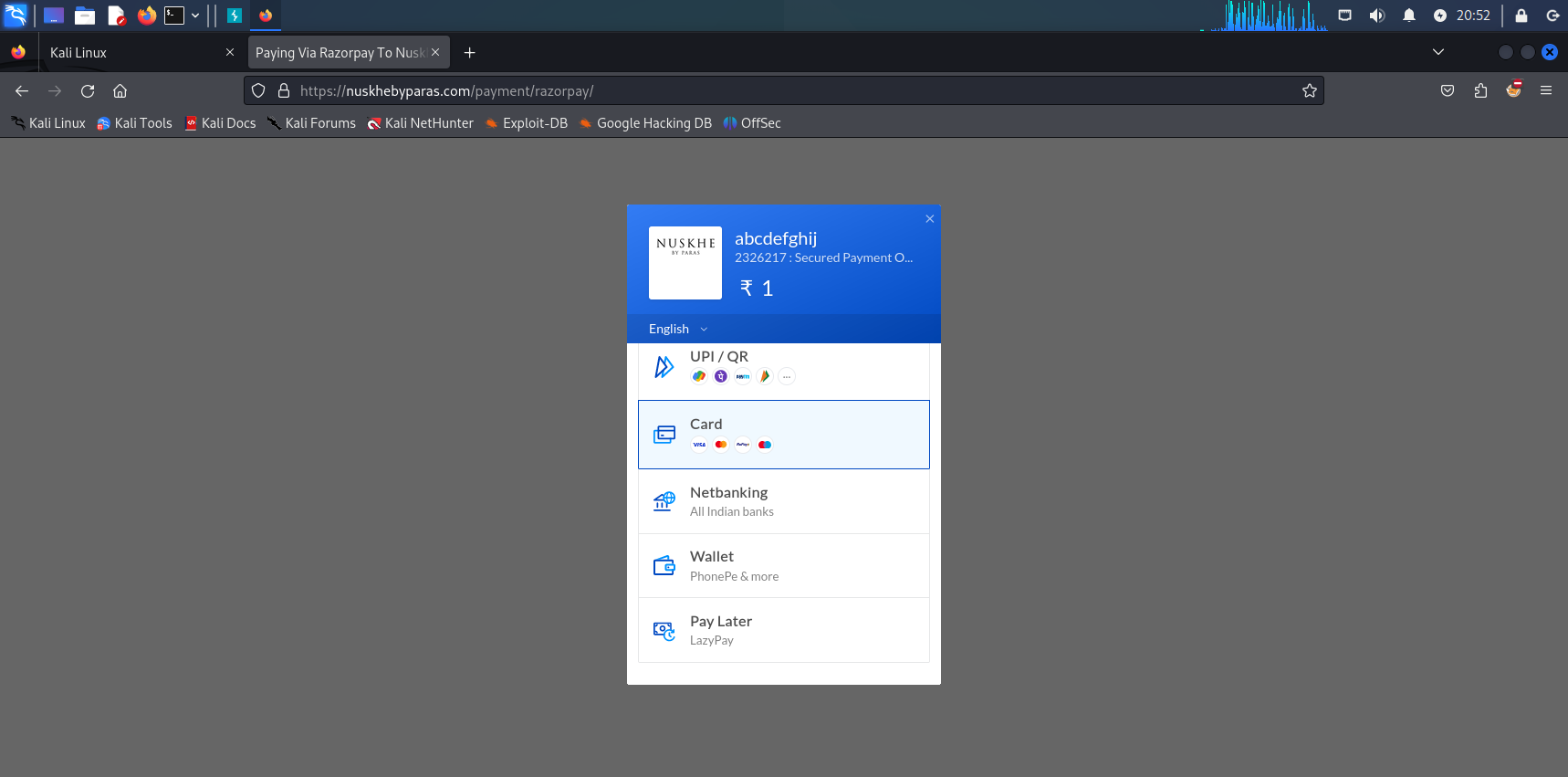




Here we can see the Burp suite. (Not suggested Because it’s illegal)

It was a successfully change as we can see the prize what we capture in Burp.

We can manipulate the prize. A serious vulnerability must be fixed.

**Impact Analysis**

The impact analysis of parameter tampering typically involves assessing several key aspects:

1. **Data Integrity**: Parameter tampering can compromise the integrity of data transmitted between client and server. For example, an attacker might alter parameters related to financial transactions or user privileges, leading to unauthorized actions or financial loss.
2. **Authentication and Authorization**: If parameters related to authentication tokens, session IDs, or user roles are tampered with, attackers can gain unauthorized access to parts of the application or perform actions reserved for privileged users.
3. **Application Logic**: Tampering with parameters can bypass client-side validation and trigger unexpected behavior in the application. This can potentially lead to application crashes, data corruption, or exposure of sensitive information.
4. **Security Vulnerabilities**: Parameter tampering often exposes security vulnerabilities such as SQL injection, where altered parameters could manipulate database queries to extract or modify sensitive information.
5. **User Experience**: Unauthorized parameter tampering can disrupt the intended user experience of an application, causing errors, unexpected behavior, or loss of functionality.
6. **Legal and Compliance Issues**: Depending on the nature of the tampering and the data affected, there may be legal implications such as violating data protection regulations.
7. **Reputation and Trust**: Successful parameter tampering attacks can damage an organization's reputation and erode user trust, especially if sensitive data or financial losses are involved.
8. **Operational Disruption**: In some cases, parameter tampering can disrupt normal application operation, leading to downtime or loss of service availability.

**Mitigation**

Authentication bypass vulnerabilities are a critical threat to web applications. Effective mitigation requires a multi-layered approach that addresses various aspects of authentication process.

Importance of a layered defense

**Depth of Protection**: Each layer in a defense strategy serves as an additional barrier that must be overcome by attackers. This makes it more challenging and time-consuming for them to achieve their objectives, potentially deterring opportunistic attacks altogether.

No single mitigation is foolproof. The cybersecurity landscape is constantly evolving. With attackers becoming increasingly sophisticated. What works today might not be effective tomorrow.

### Secure Cookies

Cookies can contain sensitive information, and securing them is a critical aspect of web security. Setting the ‘Secure’ attribute on cookies ensures that they are only sent over SSL/TLS connections. Additionally, the ‘Http Only’ attribute can be used to prevent access to cookie data via client-side scripts, further protecting against [cross-site scripting](https://www.imperva.com/learn/application-security/cross-site-scripting-xss-attacks/) (XSS) attacks.

These measures ensure that cookies, often a target for tampering, are afforded a significant level of protection, safeguarding user data and session information.

**Fundamental mitigation strategies**

**Strong Authentication and Access Control**: Implementing strong authentication mechanisms (e.g., multi-factor authentication) and enforcing strict access control policies ensures that only authorized users and devices can access sensitive information and systems.

**Encryption**: Encrypting sensitive data both at rest and in transit ensures that even if data is intercepted or accessed by unauthorized parties, it remains unreadable without the decryption key.

**Input Validation and Sanitization**:

* **Server-side Validation**: Validate and sanitize all input parameters received from clients before processing them. This includes checking data types, lengths, formats, and ranges to ensure they adhere to expected values and patterns.
* **Client-side Validation**: Implement client-side validation to provide immediate feedback to users and reduce the likelihood of submitting invalid or malicious inputs.
* **Continuous monitoring and threat intelligence:** stay updated on the least threats and vulnerabilities.
* **Logging, Monitoring, and Auditing**: Maintain comprehensive logging, monitoring, and auditing mechanisms to detect and respond to suspicious activities or tampering attempts promptly.

The threat of parameter tampering is a persistent concern. It represents a class of vulnerabilities that can be exploited to alter the behavior of web applications, leading to unauthorized access, data breaches, and a host of other security issues.

The Web Application Firewall (WAF) provides out-of-the-box security for web applications. It detects and prevents cyber threats, ensuring seamless operations and peace of mind. Deploy WAF on-premises, in AWS, Azure, and GCP, or as a cloud service.

Improve WAF is a key component of a comprehensive [Web Application and API Protection](https://www.imperva.com/learn/application-security/web-application-and-api-protection-waap/) (WAAP) stack that secures from edge to database. Provides the best website protection in the industry – PCI-compliant, automated security that integrates analytics to go beyond OWASP Top 10 coverage, and reduces the risks created by third-party code.

WAF can secure:

* Active and legacy applications
* Third-party applications
* APIs
* Cloud applications, containers, and virtual machines (VMs)

**Conclusion**

Parameter tampering attacks is crucial for ensuring the security and integrity of applications and systems. By implementing fundamental mitigation techniques, organizations can effectively reduce the risk of attackers manipulating parameters to gain unauthorized access, bypass security controls, or alter the intended functionality of an application.

By integrating these measures into their cybersecurity practices, organizations can bolster their defenses against parameter tampering attacks, safeguard sensitive data, maintain regulatory compliance, and uphold user trust. Continuous assessment, adaptation to emerging threats, and proactive security measures are essential for maintaining robust protection in an evolving threat landscape.

Implementing a combination of these fundamental techniques significantly enhances an organization’s ability to detect, prevent, and respond to parameter tampering attacks. By continually assessing risks, adapting defenses, and fostering a proactive security stance, organizations can effectively mitigate threats and safeguard their systems, data, and reputation in an increasingly complex cybersecurity.

**References**

Google Dorks:

<https://www.exploit-db.com/google-hacking-database>

Payload all the things:

<https://github.com/swisskyrepo/PayloadsAllTheThings>

OWSAP:

<https://cheatsheetseries.owasp.org/>

Wappalyzer:

<https://www.wappalyzer.com/>

Portswigger:

<https://portswigger.net/burp>

<https://www.imperva.com/learn/application-security/parameter-tampering/>

YouTube:

<https://youtu.be/RV7KBJnwzAM?si=-YwxMDhNjdWOSG8B>

**Thank you**