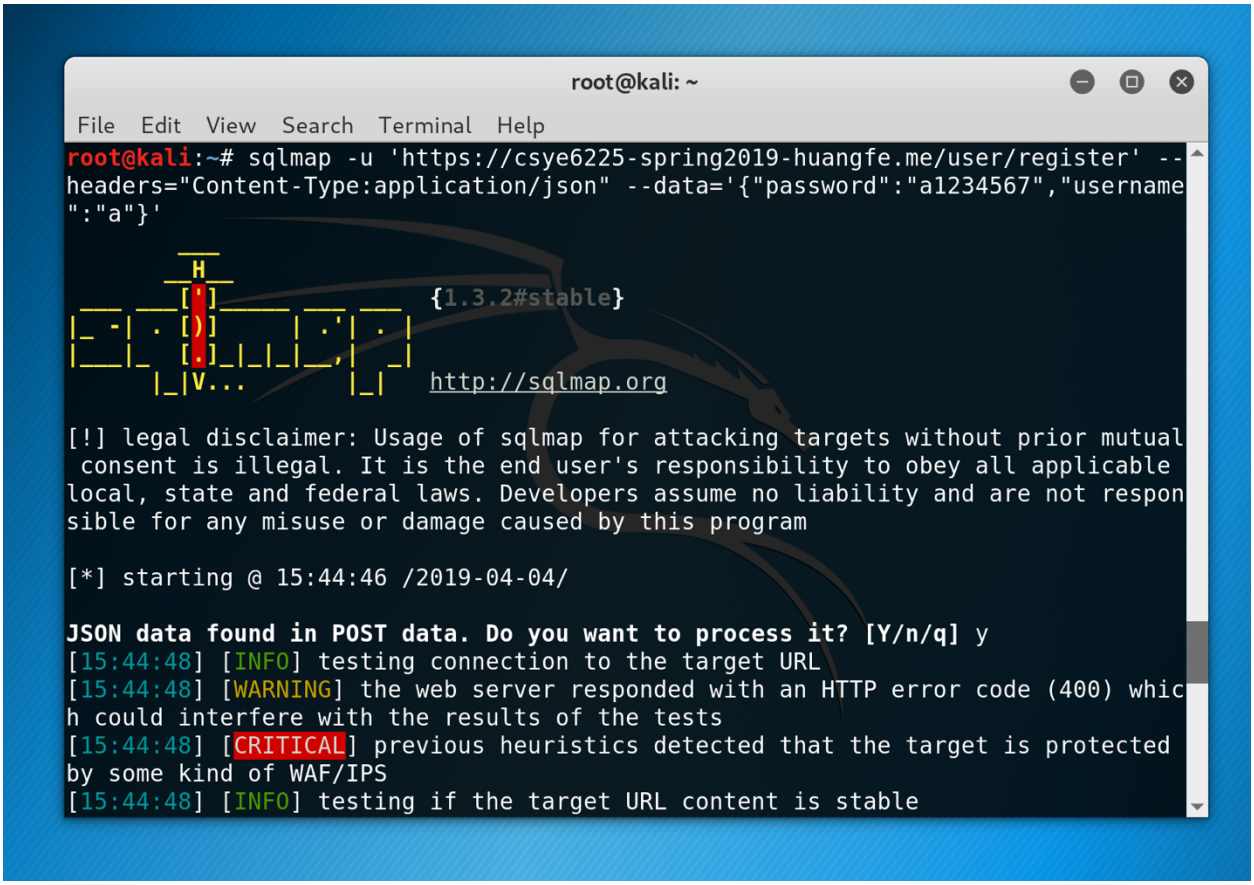
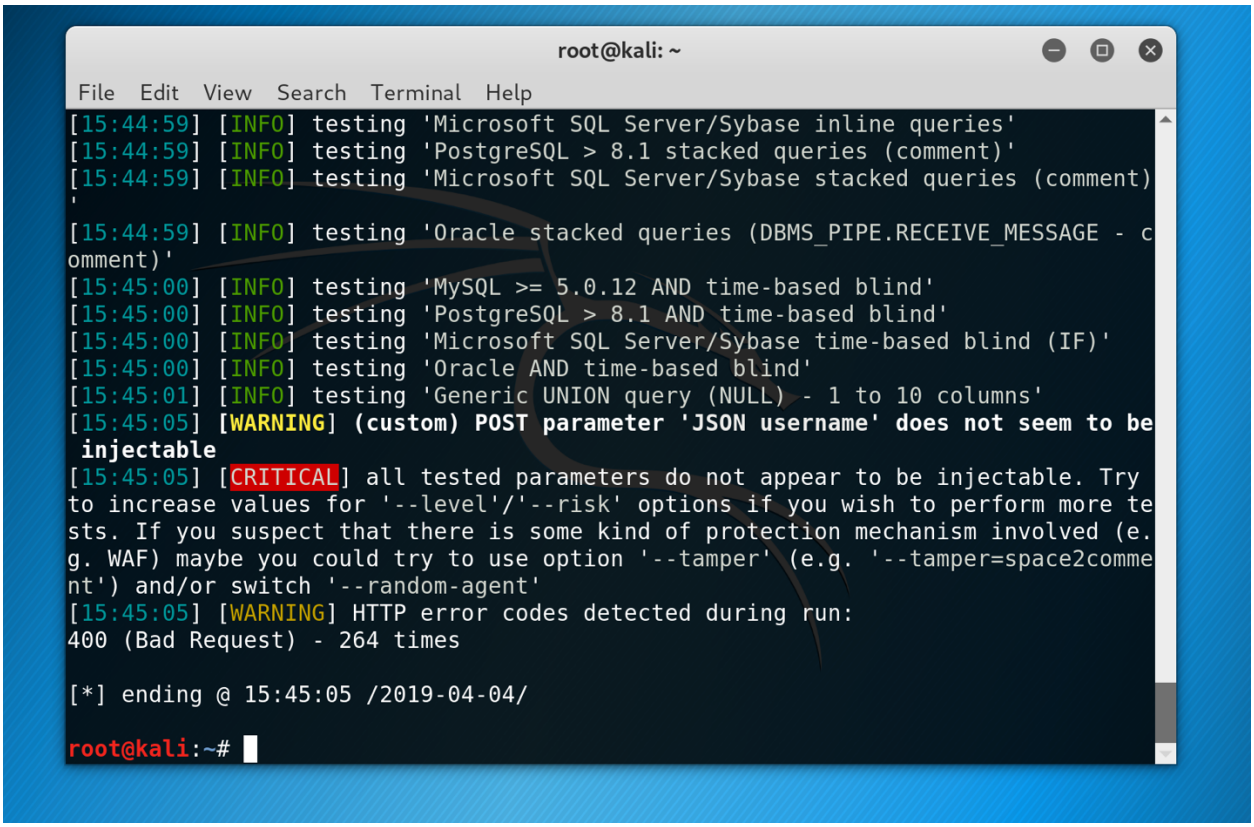
# Penetration Testing

1.SQL injection is a code injection technique, used to attack data-driven applications, in which diabolical SQL statements are inserted into an entry field for execution .

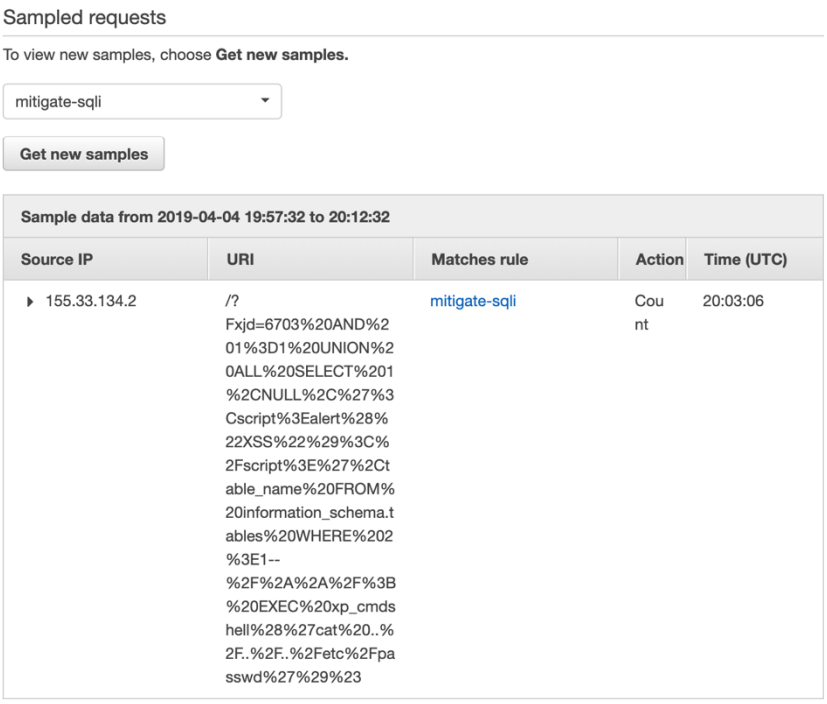
We use sqlmap for SQL injection test. For our web application, it checks username in the database when people register user. So, it should be a potential injection point.

We try to inject with or whit out WAF. Whatever the firewall is open, we can resistance attack.





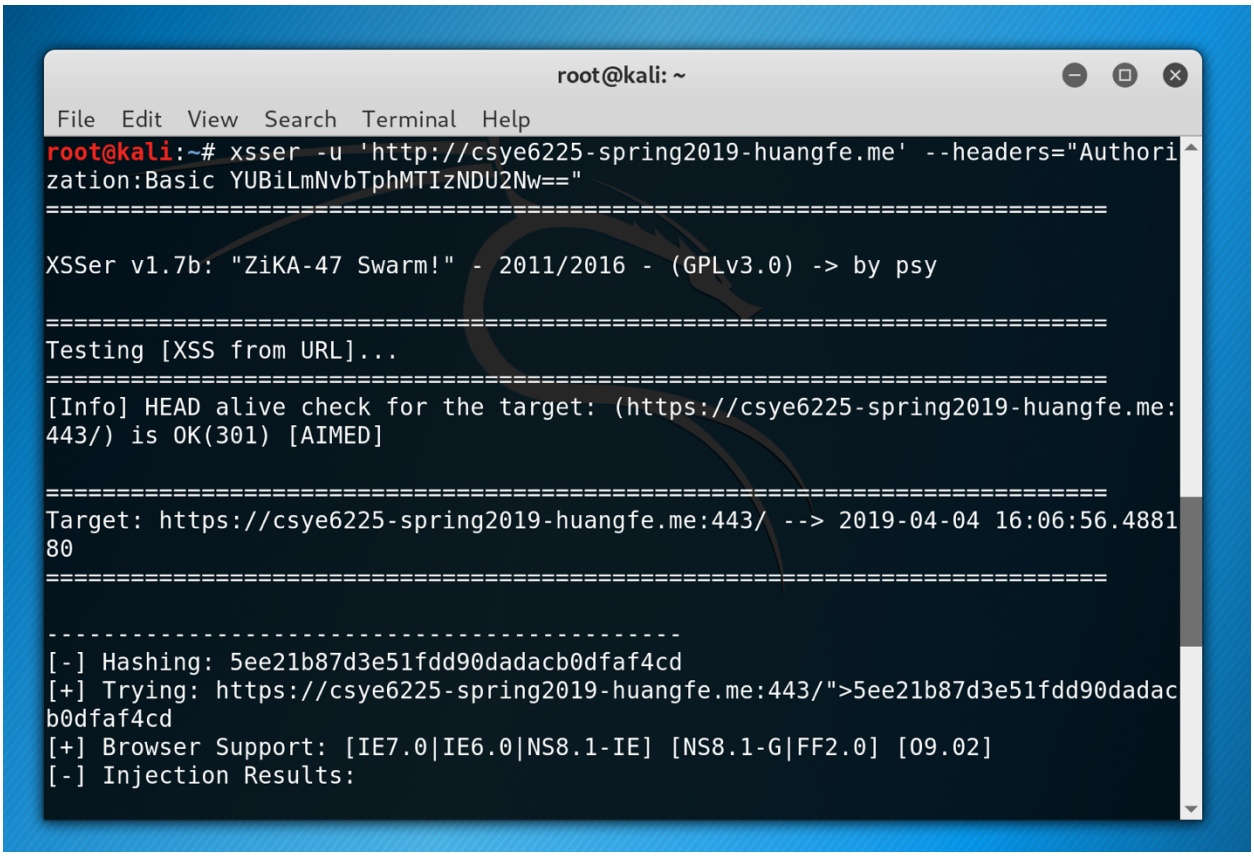

Without WAF, it still can recognize the attack and count it.

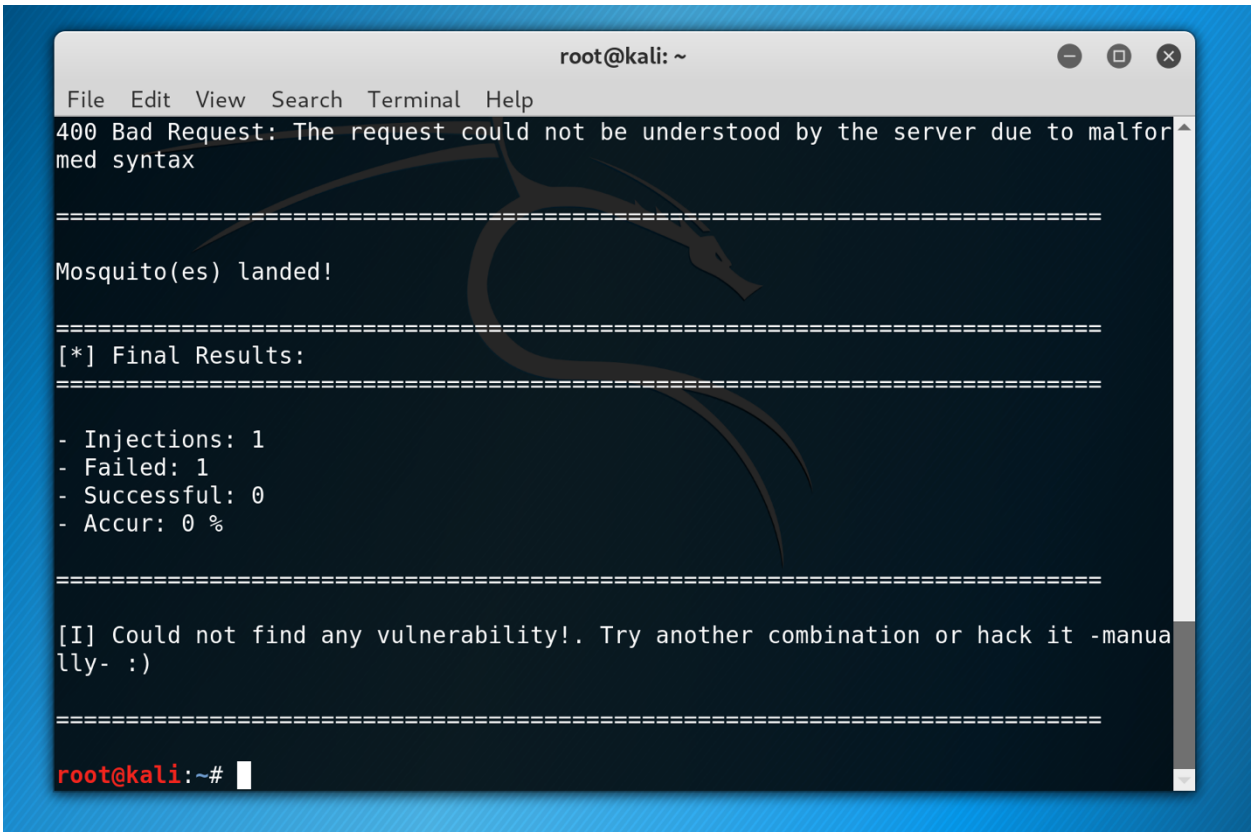


We choose this attack because we use MySQL database, and SQL injection is a big threat towards our web application. We must set a string protection for our database.

2. Cross-site scripting (XSS) is a type of injection security attack in which an attacker injects data, such as a malicious script, into content from otherwise trusted websites. Cross-site scripting attacks happen when an untrusted source is allowed to inject its own code into a web application, and that malicious code is included with dynamic content delivered to a victim's browser.

We use xsser for xss injection test. We try to inject with or whit out WAF. Whatever the firewall is open, we can resistance attack.






By the way, the WAF even block the request when we try to upload a xml file as attachment.

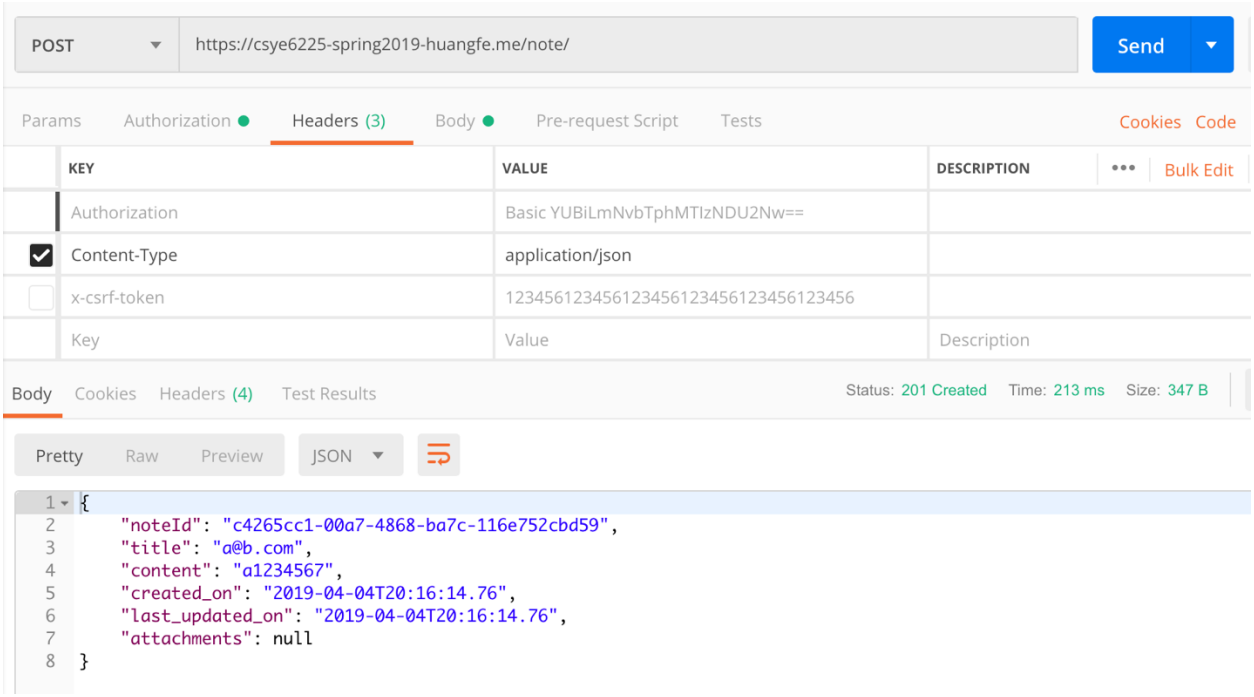
3.Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated.

CSRF attacks specifically target state-changing requests, not theft of data, since the attacker has no way to see the response to the forged request.

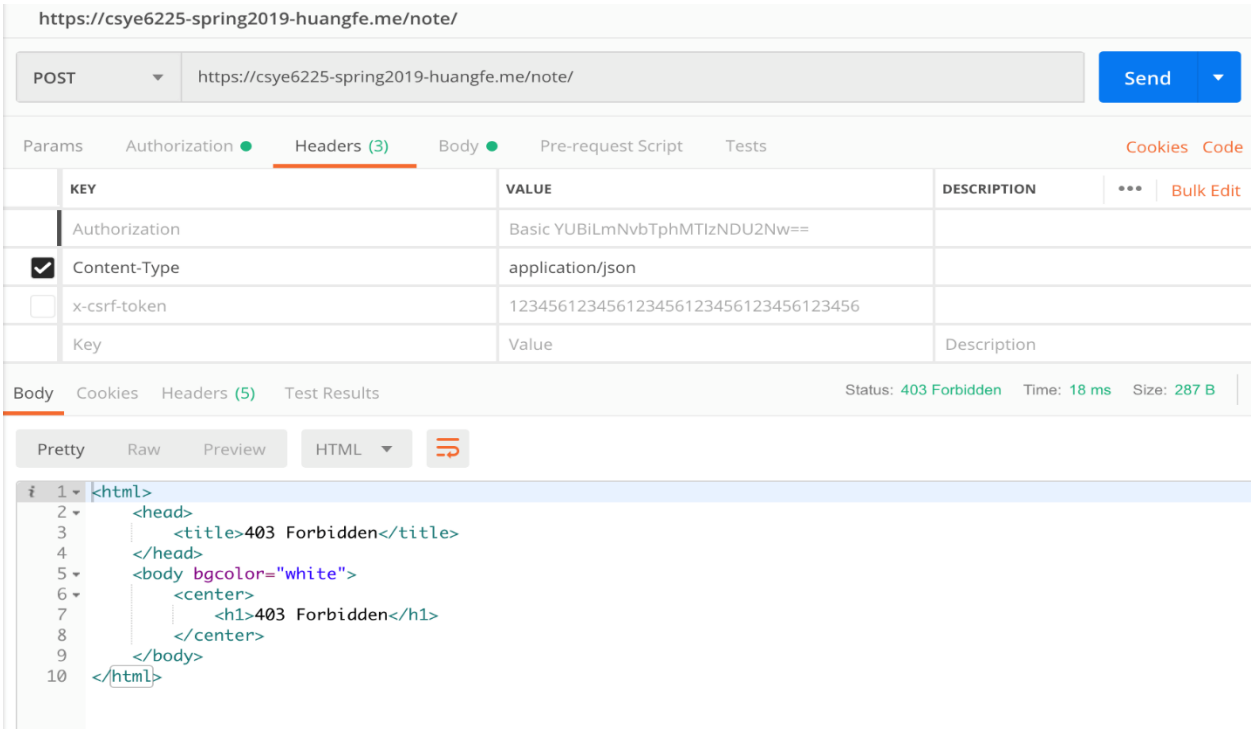
Since we use REST API for our web application, which is stateless, and we don't render actual web pages, in fact there is no need to detect CSRF.

Result: If we enforce CSRF in AWS WAF, any post requests without x-csrf-token header will be blocked. Since we don't check the actual value of x-csrf-token, any value of 36 length will be allowed.

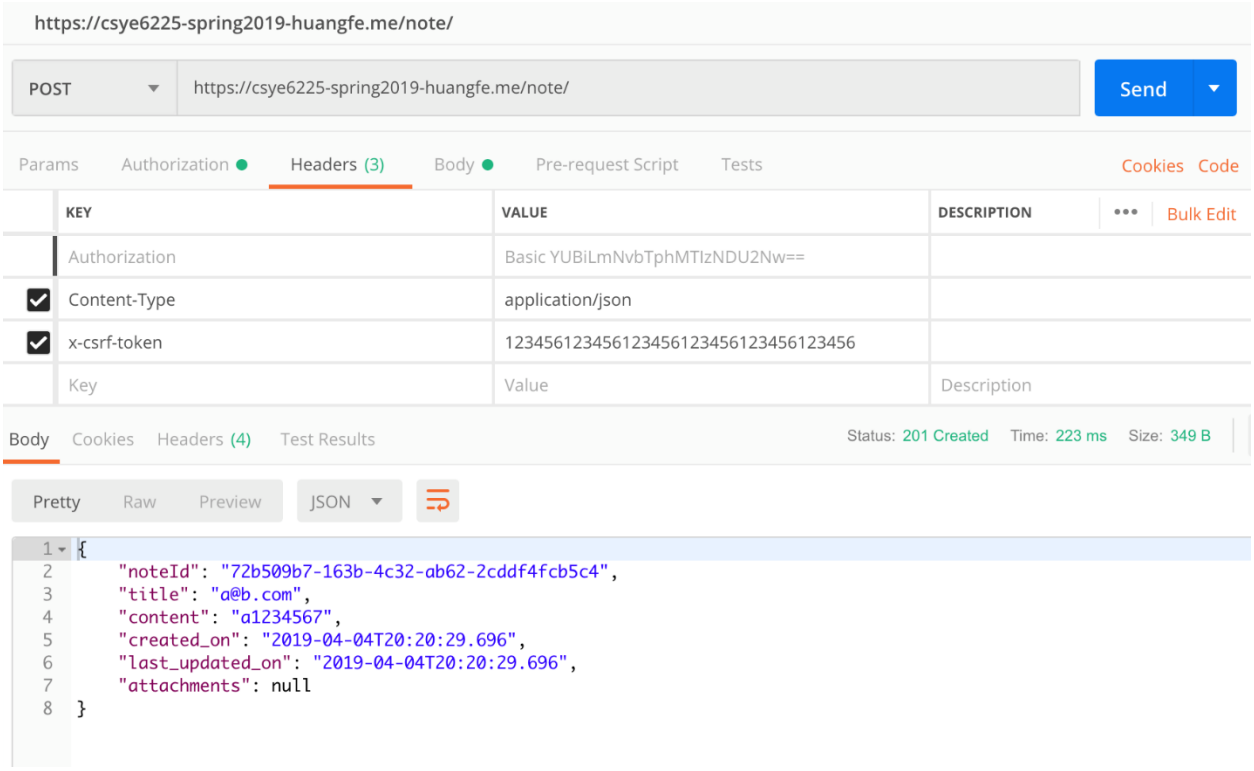
Without WAF:




With WAF & without x-csrf-token header:




With WAF & with x-csrf-token header:




Why do we choose this attack?

For most of web applications, it is developer's responsibility to consider all kinds of attack, and CSRF is one of the most common attacks to think about. Therefore, we select this attack.