10/11/2024

CCDC Tryout Challenge

Naqib Amini
CCDC COASTLINE

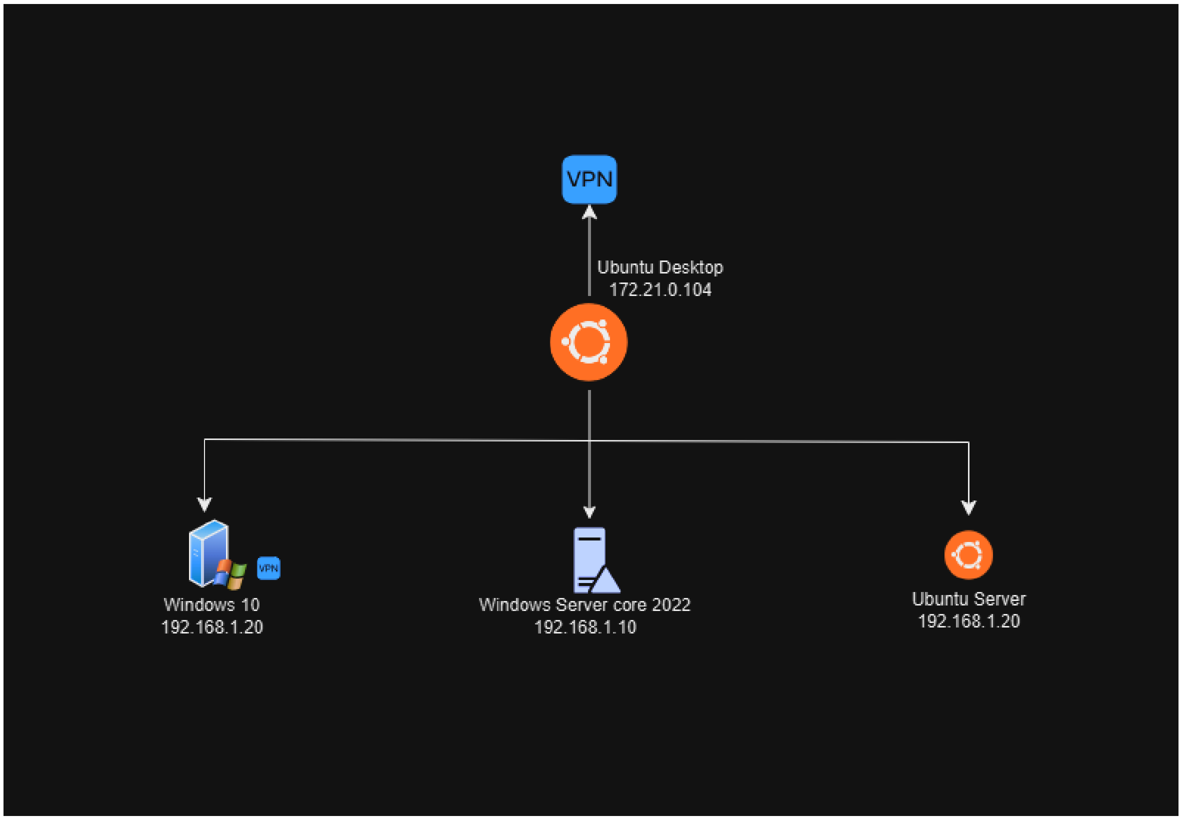
Contents

CCDC Tryout Challenge

Executive Summary

The CCDC.TRYOUTS is a small network, and students are challenged to set up and configure this environment following the best security practices. In this environment, there are a Windows Server Core 2022, Windows 10, and Ubuntu server in a private network connected to each other. Also, there is an Ubuntu Desktop connected to this network as a jump box, which is the only way to remotely connect to that private network. In addition, students are required to set up an Active Directory Domain Service on the windows server, connect other machines to it, set up Docker, Jenkins, LEMP and WordPress, and implement the required policies. As a student who participated in this challenge, I have documented an overview of taken steps and decisions I made and presented them below in this report.

Topology



Widows Sever 2022 (Domain Controller)

1. Active Directory and DNS

You are required to create, and AD domain named CCDC.TRYOUTS with the following components:

- Install and configure Active Directory Domain Services to create the domain CCDC.TRYOUTS.
- Configure the server to function as the DNS server for the domain.
- Create two Domain Administrator user accounts: John Hammond and Kevin Mitnick.
- Create three Regular Domain user accounts: David Bomball, Chuck Keith, Eugene Kaspersky.
- Use the naming conventions: first initial followed by last name (e.g., jdoe for John Doe).

I have installed and configured the Active Directory Domain Services and DNS on Windows Server 2022. To do that, followed the Microsoft Install Active Directory Domain Services (Level 100) Document. Then, I created all 5 users listed above with the first initial followed by last name each user. Finelly, I added John Hammond and Kevin Mitnick in the “Domain Admins” group to make sure they have the admin privileges.

2. Security Configurations

- Disable the default local Administrator account on the Domain Controller
- Restrict all local accounts from logging into domain-joined machines where possible, enforcing domain authentication
- Ensure that only the Domain Admin users can log on to the Domain Controller
- Configure and enable SSH access on the Domain Controller, restricting to only the Domain Admin users

I listed all the local users and disable them to enforcing only domain authentication. Then, I created and implemented Deny Log on Locally and Deny log on through Remote Desktop Services group policies. I configure them in a way to make ensure restricting all local accounts from logging into domain-joined machines, and only the Domain Admins users can log on to the Domain Controller. In addition, I installed and configure SSH server on the Domain Controller. I changed the configuration file to only allow members of the Domain Admins to remotely connect via SSH. This is possible by adding “AllowGroups ccdc\"Domain Admins”” line in sshd_config.

Note: In this stage, I used GUI tools like RSAT and WAC to make the process easier.

3. Security Policies

3.1. Password Policy:

- Set a minimum password length that meets industry best practices
- Enforce password complexity requirements, including uppercase and lowercase, letters, numbers, and special characters
- Configure password expiry that meets industry best practices
- Implement account lockout policies that meet industry best practices

To impellent a strong password policy, I used the Microsoft Password policy recommendations and CIS Password Policy Guide. In the Group Policy Mangement, there is a Default Domain Policy that includes password policy, and this policy applies to the entire domain. To prevent conflicts and duplication, I decided to edit the password policy included in Default Domain Policy. Before making any changes, I backed up all the default group policies to make sure I can revert changes in future. Finely, I configured the password Policy with the:

- Minimum password length of 8 character
- Complexity requirement enabled
- 365 days password expiration
- Temporary account lockout for 15 minutes after 5 consecutive failed attempts

3.2. Audit Policy:

- Enable audit policies for all events and choose to log success, failure, or both.

To make sure creating a good Audit Policy, I used the stronger recommendations provided in Microsoft Audit Policy Recommendations. In this page Microsoft, listed Audit Policy Category or Subcategory and their default settings. Also, they have Baseline Recommendation and Stronger Recommendation for choosing log success, failure, or both. I created a group policy by going to Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Audit Policy.

3.3. Update Policy:

- Configure automatic download and install of updates on a weekly basis.

I configure the windows update policy to happen every Sunday at 3:00 AM. I decided this day and time to prevent any unavailability for users during workdays. In addition, on Monday, when users come back to use their computers, their device is already up to date and ready to use. I configured this in group policy by going to Computer Configuration > Policies > Administrative Templates > Windows Components > Windows Update > Configure Automatic Updates.

Windows 10 Workstation

1. Domain Integration

- Join the Windows 10 workstation to the CCDC.TRYOUTS domain.

To join the Windows 10 to the AD, we first add the IP Adress of the DC as the first DNS server on this machine. Then we go to Settings > System > About > Advanced System Settings > Computer Name > Change, select the Domain, add the Domain information and admin's credentials. Thus, we will have the Windows 10 machine in our AD.

2. User Access

- Ensure that the three Regular Domain users can log into this workstation using their domain credentials.

I have tested all users for logging in to this machine and all users can log in.

3. Docker and Jenkins

- Install Docker on the Windows 10 workstation.
- Deploy a Docker container running Jenkins:
  - Configure Jenkins to be accessible from other domain-joined machines via https://jenkins.ccdc.tryouts:1337
  - Ensure https is used with TLS/SSL

The docker official documentation for installing docker in windows is great resource that we can use in this regard. Using this document, helped me to securely install the docker on Windows 10 machine. In addition, I used the Jenkins Official Documentation to install Jenkins on docker. The challenge I faced in this stage was the Jenkins' TLS/SSL connection. After a lot of researches, I found a solution in Stack Overflow and after editing the docker commands, I was able to set up an TLS/SSL connection. Finelly, I tested everything to ensure the website is available over https://jenkins.ccdc.tryouts:1337.

4. SSH Access

- Enable SSH access on the workstation.
- Restrict SSH login to the Domain Administrators.

Same as DC, I installed and configure SSH server on the Windows 10 machine. I changed the configuration file to only allow members of the Domain Admins to remotely connect via SSH. This is possible by adding **AllowGroups ccdc\"Domain Admins”** line in sshd_config.

Ubuntu 22.04 Server

1. Domain Integration

- Join the Ubuntu server to the CCDC.TRYOUTS domain

I was able to find a walkthrough for the same version of Ubuntu that helped to join the server to AD. Join Ubuntu 22.04 to Microsoft Active Directory domain documents, walked me through, configuring Network settings, installing necessary packages, and using realm to join the domain, setting up home directories for domain users, configuring SSSD and PAM. As a result, I was able to join AD, and login to Ubuntu server with Domain users by entering their username and password.

2. LEMP Stack and WordPress

- Install and configure a LEMP stack (Linux, Nginx, MySQL/MariaDB, PHP).
- Set up a basic WordPress blog that is accessible to other domain-joined machines via
- https://wordpress.ccdc.tryouts:443
- Ensure https is used with TLS/SSL

I used three articles from DigitalOcean to complete all these tasks. The first article I used was How To Install Linux, Nginx, MySQL, PHP (LEMP stack) on Ubuntu. It helped me get a basic understanding of LEMP Stack and how things work. Following this article, I was able to install and configure Nginx, MySQL, and PHP on ubuntu server. In addation, I could create users and databases in MySQL and connect them to my simple website using a PHP program. Afterward, I used the article How to Install WordPress with LEMP on Ubuntu. Using this article, I was able to download and configure WordPress website on the Nginx server. In addation, I could create a database and user for WordPress on MySQL and connect them to WordPress. Moving forward, I used the article How To Create a Self-Signed SSL Certificate for Nginx in Ubuntu. Following the instruction in this article, I was able to create a self-signed certificate for Nginx SSL/TLS connection and redirect any HTTP request to HTTPS. Finely, I tested https://wordpress.ccdc.tryouts:443 to ensure the encrypted connection.

3. WordPress Security

- Secure the WordPress installation by restricting file permissions and disabling directory browsing
- Create WordPress user accounts for each domain user and make sure they can make blog posts

After installing and configuring WordPress, I tested the file permissions and directory browsing to make sure there is no security vulnerability in the configurations. Then, I installed Active Directory Integration / LDAP Integration plug-in to integrate AD users to WordPress website. Subsequently, I followed the instruction in Step by step guide to setup LDAPS on Windows Server to add more security for LDAP connection. The most challenging task in this part was issuing a certificate and configuring Ubuntu server to trust the Certificate Authority. The documentation that I was using, didn't work from some point because the core version of the Domain Controller. After a lot of researches, I was able to find the command line solution and create the LDAPs connection. Finely, I added all users as **editors** on the site to make sure they can blog posts, and tested everything to ensure everything is good.

4. SSH Access and Privileges

- Enable SSH access on the Ubuntu server
- Ensure all Domain Users can log in via SSH with their domain credentials.
- Ensure that only the Domain Administrators have root/sudo access

The SSH was enabled on Ubuntu by default. Then, after joining to AD, all users could remotely connect to the server using their domain credentials via SSH. So, I didn't need to change any configuration for SSH at that point. However, I changed the /etc/sudoers file to ensure only Domain Admins have the root/sudo acces to the machine. I did this by adding **%domain\ admins ALL=(ALL) ALL** line in /etc/sudoers file and adding a comment in front of all other users/group with **ALL** privileges. A challenge in this part was the local Administrator user on local machine that still had the sudo privileges even the privileges wasn't specified for him in /etc/sudoer file nor in any other place. I found that the cause of the problem and solved it. The cause of problem was the exist of a user under the Administrator name on Domain Admins group on Active Directory. Even local Administrator user was disabled on the Domain Controller, but being part of the Domain Admins group, allowed the local Administrator user on Ubuntu to have the sudo privileges. After all, the solution was easy, either removing the administrator user Domain Admins group or changing the name of local administrator user on Ubuntu to something else. Finely, I tested the privileges to ensure the configurations are correct.

5. Update Policy

- Configure the server to automatically download and install ALL updates on a weekly basis

After a little bit of research, I learned about a great tool for Ubuntu that can handle automatic updates. This tool is Unattended Upgrades and includes a lot of good features. For example, we can configure automates security updates, selective updates, automatically reboot, and a lot of other features. The downside for this tool is that we can not specify a specific day week for update. However, by combining it with crontab, we can achieve this goal too. For doing that, I followed the instruction in Automatically Update Ubuntu 24.04 or 22.04 LTS using Unattended upgrades. Following this article, I could configure the Unattended Upgrades and combine it with crontab.

Conclusion

To sum up, the CCDC Tryout Challenge gave participants extensive practical experience configuring and safeguarding a small network environment. Working with several platforms, including Windows Server 2022, Windows 10, and Ubuntu, and integrating Active Directory, Docker, Jenkins, LEMP, and WordPress, allowed me to show that I could set up, protect, and oversee plenty of different services. From imposing strict security regulations to setting up smooth domain integration and safeguarding access via SSH and LDAPS, each activity highlighted important best practices. The difficulties I ran across, especially with SSL/TLS setups and handling privilege management problems, gave me the chance to learn more about troubleshooting methods and hone my problem-solving abilities. In the end, this task strengthened my comprehension of how to design a safe, reliable, and functional network environment.