1.0 Install a Windows Server 2022

1.1 With 2-4 GB of RAM

1.2 With 40-60 GB of Storage

1.3 Network adapter -host only

1.4 Install vmware tools

1.5 Name it DC01

1.6 Set-up IP address of Windows Server to 192.168.1.200 /24  DNS 192.168.1.200 by Powershell

2.0 Install ADDS

2.1 Create a new Domain name it **yourfirstname.ca**

1.0 PS of Vmware Settings

2. PS of Vmware name [DC01]

3. PS of IP address is set into VM by powershell 192.168.1.200 /24 DNS 192.168.1.200

4.0 PS of ADDS is installed

5.0 PS of ADDS contain yourfirstname.ca

Theoretical Part

1. Describe in 350-500 words the purpose of ADDS
2. Describe in 350-500 words  the purpose of automation in Powershell
3. Describe in 350-500 words  the purpose of Powershell
4. A screenshot of a computer

   Description automatically generatedPS of VMWare Settings
5. A computer screen with text on it

   Description automatically generatedPS of VMWare Name (DC01)

A screenshot of a computer

Description automatically generated3.0 PS of IP address is set into VM by PowerShell 192.168.1.200 /24 DNS 192.168.1.200

A computer screen shot of a program

Description automatically generated4.0 PS of ADDS is installed

A computer screen with white text

Description automatically generated5.0 PS of ADDS contain yourfirstname.ca

1. Describe in 350-500 words the purpose of ADDS

Active Directory Domain Services (ADDS) is a server role in Active Directory that allows admins to manage and store information about resources from a network, as well as application data, in a database. It’s used in managing user attributes like passwords, group memberships and many other features.

ADDS serves as a centralized repository for user accounts, computer accounts, and other network resources. It allows organizations to create and manage user identities, enabling users to log in with a single set of credentials across multiple devices and services.

The major function of ADDS is to authenticate users and grant them appropriate access rights to network resources. Through the use of security groups and access control lists (ACLs), administrators can define who can access specific files, folders, printers, and applications. This ensures that only authorized users can access sensitive data.

ADDS provides a hierarchical structure called a domain tree, consisting of domains and organizational units (OUs). This structure makes it easier for administrators to organize and manage network resources. They can delegate administrative tasks to specific OUs, allowing localized control over user accounts, groups, and other objects.

ADDS allows administrators to define and enforce settings, configurations, and security policies for users and computers within the network. This ensures consistent and compliant system configurations, reducing security vulnerabilities and maintenance overhead.

ADDS is designed to scale with the needs of an organization. It supports multi-domain and multi-forest configurations, making it suitable for large enterprises. Additionally, it provides redundancy through features like domain controllers, which can replicate data to ensure high availability.

ADDS seamlessly integrates with various Microsoft services, including Exchange Server for email management, SharePoint for collaboration, and Azure AD for cloud-based identity and access management. This integration simplifies the deployment and management of these services.

ADDS incorporates security features such as fine-grained password policies, account lockout policies, and auditing capabilities. These features help organizations strengthen security, protect against unauthorized access, and meet compliance requirements.

ADDS infrastructure to Azure AD for a hybrid identity solution. This enables seamless authentication and access to cloud-based resources while maintaining centralized control and security.

ADDS enables Single Sign-On, allowing users to access multiple resources with a single set of credentials. This enhances user convenience and reduces the need to remember multiple passwords.

1. Describe in 350-500 words the purpose of automation in PowerShell

The purpose of automation in PowerShell is to increase efficiency and save time. Many administrative tasks, such as software deployment, user provisioning, and system configuration, involve executing numerous repetitive steps. Writing PowerShell scripts allows administrators to automate these tasks, reducing the time and effort required to complete them.

Automation ensures consistency and accuracy in system administration. Human errors are a common occurrence when performing repetitive tasks manually. By automating these tasks, PowerShell eliminates the potential for errors, ensuring that configurations and actions are applied uniformly. This reduces the risk of security vulnerabilities, misconfigurations, and other issues that can result from human mistakes.

PowerShell automation enables information technology professionals to manage large and complex environments with ease. This scalability is particularly valuable in modern cloud and virtualized infrastructures, where resources are provisioned and deprovisioned dynamically.

Automation promotes standardization by enforcing consistent configurations and procedures. PowerShell scripts can be used to define and enforce organizational standards for security, compliance, and performance. This ensures that all systems adhere to a predefined set of rules and guidelines, reducing the risk of non-compliance or vulnerabilities.

PowerShell's automation capabilities extend to remote systems, allowing administrators to manage resources across the network. This is especially valuable in scenarios where servers or devices are geographically dispersed. With PowerShell, you can remotely execute commands, retrieve information, and make changes on remote systems, simplifying the management of distributed environments.

Automation in PowerShell goes beyond immediate task execution. It allows administrators to schedule tasks to run at specific times or intervals. This scheduling capability is crucial for routine maintenance, backups, and monitoring. Additionally, PowerShell scripts can generate detailed reports and logs, providing insights into system health and performance over time.

1. Describe in 350-500 words the purpose of PowerShell

PowerShell is a go-to tool for system administrators. It allows them to perform a wide range of administrative tasks, including user management, server configuration, file system operations, network configuration, and more.

The purpose of PowerShell is to automate routine and complex tasks. Administrators can write scripts and automation workflows to handle tasks like software installations, updates, system backups, and monitoring.

PowerShell is a key tool for managing and configuring Windows servers and workstations. Administrators can use it to create and apply Group Policy settings, configure network settings, manage services, and set up security policies. This ensures that systems are properly configured and adhere to organizational standards.

PowerShell provides the ability to gather data from various sources and generate reports for monitoring system performance, security, and compliance. Scripts can be written to collect event logs, system metrics, and other relevant information, which can be analyzed or presented in a friendly format.

A crucial component in Windows environments, can be efficiently managed and maintained using PowerShell. IT professionals can create, modify, and delete user accounts, groups, organizational units (OUs), and implement complex permissions and security settings.

PowerShell is not limited to on-premises systems; it's essential in managing cloud resources and hybrid environments. Microsoft Azure, AWS, and other cloud platforms provide PowerShell modules and cmdlets for automating cloud-related tasks.

Developers and DevOps teams utilize PowerShell to automate software deployment, configuration management, continuous integration, and continuous delivery processes.

PowerShell was initially designed for Windows, it has expanded its reach to other operating systems, including Linux and macOS. With PowerShell Core (now known as PowerShell 7, users can write scripts that work across different platforms, making it a valuable tool for hybrid scenarios.