Installing a 2-tier Certificate Authority (PKI) hierarchy using AutomatedLab

Contents

[AL Tutorial Series Part 5 - Installing a 2-tier Certificate Authority (PKI) hierarchy using AutomatedLab 2](#_Toc401732122)

[Summary 2](#_Toc401732123)

[Installation 2](#_Toc401732124)

[Prerequisites for AutomatedLab 2](#_Toc401732125)

[Prerequisites for installing the two CA servers in the lab 2](#_Toc401732126)

[Defining the lab machines 2](#_Toc401732127)

[Customizing the configuration of the CA servers 4](#_Toc401732128)

[Supported PKI configurations in AutomatedLab 6](#_Toc401732129)

[What’s next? 6](#_Toc401732130)

[The Full Script 6](#_Toc401732131)

# AL Tutorial Series Part 5 - Installing a 2-tier Certificate Authority (PKI) hierarchy using AutomatedLab

## Summary

Following up on the first article “Installing a simple Certificate Authority (PKI) using AutomatedLab”, this article will show how to deploy a PKI environment as it would be typically deployed in a production environment. This time, the PKI environment will be deployed using two separate servers. One server will be a Root CA (Certificate Authority) server and the other will be a Subordinate CA server. The Root CA will be in a workgroup and the Subordinate CA will be in a domain. Subsequently, you will then be able to request and issue certificates from the Subordinate CA server to all computers and users in your lab. Also, if you want to be able to sign PowerShell scripts in the lab, a certificate can be created for this purpose.

The home, and where to get **AutomatedLab** is: <https://automatedlab.codeplex.com/>

## Installation

If you have a pre-2.5 version of AutomatedLab installed, please uninstall it and install the latest version from <http://automatedlab.codeplex.com/>.

The installation process of AutomatedLab is covered in the article [AutomatedLab Introduction - Part 2](http://blogs.technet.com/b/fieldcoding/archive/2014/07/27/automatedlab-introduction-part-2.aspx).

## Prerequisites for AutomatedLab

AutomatedLab requires **Hyper-V** and **PowerShell 3.0** (or higher). Hence, you need one of the following operating systems on the host where you want to install the lab:

* Windows Server 2012/2012R2
* Windows 8
* Windows 8.1

Although Windows Server 2008 R2 could work and Windows 10 hasn’t been tested, it is recommended at this point to use Windows Server 2012/2012R2 or Windows 8/8.1 on the host machine.

AutomatedLab scripts needs to be executing directly on the host where the lab environment (the VMs) will be installed/created.

For more information about the overall installation process, refer to the previous articles.

## Prerequisites for installing the two CA servers in the lab

There are really no prerequisites other than the servers where the certificate authority roles will be installed on, need to be at least Windows Server 2012.

## Defining the lab machines

First, for the Subordinate server, a domain is needed. Hence, a domain controller needs to be installed. The installation part of this, will is the same as in the first article.

The domain and the domain controller is defined like this:

$role = Get-LabMachineRoleDefinition -Role RootDC `

-Properties @{DomainFunctionalLevel = "Win2012R2"

ForestFunctionalLevel = "Win2012R2"}

Add-LabMachineDefinition -Name S1DC1 `

-MemoryInMb 512 `

-Network $labNetworkName `

-IpAddress 192.168.81.10 `

-DnsServer1 192.168.81.10 `

-DomainName test1.net `

-IsDomainJoined `

-Roles $role `

-InstallationUserCredential $installationCredential `

-ToolsPath $labSources\Tools `

-OperatingSystem 'Windows Server 2012 R2 SERVERDATACENTER'

Then, the Root CA need to be installed. This server will be a Stand-Alone CA server and not joined to any domain. Since the Certificate Authority (like domain controller) is a **role** in AutomatedLab, this **role** needs to be specified when defining the lab machine. The role is selected using the cmdlet **Get-LabMachineRoleDefinition** like this:

$role = Get-LabMachineRoleDefinition -Role CaRoot

Next, the lab machine can defined using the selected role.

Add-LabMachineDefinition -Name S1ROOTCA1 `

-MemoryInMb 512 `

-Network $labNetworkName `

-IpAddress 192.168.81.11 `

-DnsServer1 192.168.81.10 `

-DomainName test1.net `

-IsDomainJoined `

-Roles $role `

-InstallationUserCredential $installationCredential `

-ToolsPath $labSources\Tools `

-OperatingSystem 'Windows Server 2012 R2 SERVERDATACENTER'

Next, the Subordinate CA server needs to be defined. This is the code for doing this:

$role = Get-LabMachineRoleDefinition -Role CaSubordinate

Next, the lab machine can defined using the selected role. This command is the same as for defining the Root CA. Only the contents of the $role variable is different.

Add-LabMachineDefinition -Name S2SUBCA1 `

-MemoryInMb 512 `

-Network $labNetworkName `

-IpAddress 192.168.81.12 `

-DnsServer1 192.168.81.10 `

-DomainName test1.net `

-IsDomainJoined `

-Roles $role `

-InstallationUserCredential $installationCredential `

-ToolsPath $labSources\Tools `

-OperatingSystem 'Windows Server 2012 R2 SERVERDATACENTER'

Now when all the machines have been defined, start the installation of the lab like usual. This would mean, export the lab definition, import it (which also validates the configuration and reports any errors) and finally start the actual installation which will create virtual network, create base images, create Hyper-V VMs and install the roles found in the lab.

The code for this, looks like this:

Export-LabDefinition -Force -ExportDefaultUnattendedXml

Import-Lab -Path (Get-LabDefinition).LabFilePath

Install-Lab -NetworkSwitches -BaseImages -VMs

Install-Lab -Domains

At this point, the Domain Controller is installed and ready. Now the installation of the Certificate Authority needs to be started. This is done like this:

Install-Lab -CA

Notice that you do not need to instruct the cmdlet **Install-Lab** about how to install the CA servers and how they should be configured. This will done automatically.

Now, the lab is ready with a domain controller (hosting a domain of course) and with two Enterprise CA servers. Now, you can request and issue certificates (from the Subordinate CA) for use in your lab!

### Customizing the configuration of the CA servers

The first install was very easy as all configuration of the Root CA and the Subordinate CA is automatically done when plainly calling **Install-Lab -CA.** Now,let’s try installing a PKI environment where we define some of the configuration of the CA ourselves. Even though the default installation will work in the far majority of situations for a test lab, it could be necessary to specify certain parts of the configuration for the PKI environment.

First of all, the current lab needs to be removed. Do this by calling

Remove-Lab -Path <path to the **lab.xml** file>

The cmdlet **Remove-Lab** (turns off and) removes the VMs, the disks and finally the network adapter.

First off, the domain and the domain controller needs to be defined as before.

$role = Get-LabMachineRoleDefinition -Role RootDC `

-Properties @{DomainFunctionalLevel = "Win2012R2"

ForestFunctionalLevel = "Win2012R2"}

Add-LabMachineDefinition -Name S1DC1 `

-MemoryInMb 512 `

-Network $labNetworkName `

-IpAddress 192.168.81.10 `

-DnsServer1 192.168.81.10 `

-DomainName test1.net `

-IsDomainJoined `

-Roles $role `

-InstallationUserCredential $installationCredential `

-ToolsPath $labSources\Tools `

-OperatingSystem 'Windows Server 2012 R2 SERVERDATACENTER'

Now, when defining the role of the CA, we have the option of specifying configuration parameters. Take a look at the following:

$role = Get-LabMachineRoleDefinition `

-Role CaRoot @{CACommonName = "MySpecialRootCA1"

KeyLength = “4096”

ValidityPeriod = "Year"

ValidityPeriodUnits "20"}

Next, the lab machine can defined using the selected role with the customized configuration.

Add-LabMachineDefinition -Name S1ROOTCA1 -MemoryInMb 512 `

-Network $labNetworkName `

-IpAddress 192.168.81.11 `

-DnsServer1 192.168.81.10 `

-DomainName test1.net `

-IsDomainJoined `

-Roles $role `

-InstallationUserCredential $installationCredential `

-ToolsPath $labSources\Tools `

-OperatingSystem 'Windows Server 2012 R2 SERVERDATACENTER'

Last, the machine for Subordinate CA needs to be defined. This command is the same as before. Only the contents of the $role variable is different.

Take a look at the following:

$role = Get-LabMachineRoleDefinition `

-Role CaRoot @{CACommonName = "MySpecialRootCA1"

KeyLength = “4096”

ValidityPeriod = "Year"

ValidityPeriodUnits "20"}

For actually adding (defining) the machine, type the following command. This command is the same as for defining the Root CA. Only the contents of the $role variable is different.

Add-LabMachineDefinition -Name S1SUBCA1 -MemoryInMb 512 `

-Network $labNetworkName `

-IpAddress 192.168.81.11 `

-DnsServer1 192.168.81.10 `

-DomainName test1.net `

-IsDomainJoined `

-Roles $role `

-InstallationUserCredential $installationCredential `

-ToolsPath $labSources\Tools `

-OperatingSystem 'Windows Server 2012 R2 SERVERDATACENTER'

Just as before, perform the actual installation of the lab using:

Export-LabDefinition -Force -ExportDefaultUnattendedXml

Import-Lab -Path (Get-LabDefinition).LabFilePath

Install-Lab -NetworkSwitches -BaseImages -VMs

Install-Lab –Domains

Install-Lab -CA

Also notice that the command for installing the CAs is the same as previously. Difference is that now the parameters specified in the **role** parameters will be passed to the CA installation code.

Now, **this time**, the lab is ready with a Domain Controller (hosting a domain of course) and with a complete 2-tier PKI hierarchy configured **using our customized parameters**.

To see what parameters is possible to specify when installing a CA server in AutomatedLab, you can type the following:

Get-Help Install-LWLabCAServers -Parameter \*

Then you will see all possible parameters. To see just the names of the parameters (without the detailed information), type:

(Get-Command Install-LWLabCAServers).Parameters.Keys

Changing the parameters for the CA servers to be installed, requires you to know about how the corresponding configuration parameters for CA servers are working. Hence, it is only recommended to specify/customize parameters if you know about PKI and thereby have the need to specify/customize parameters.

## Supported PKI configurations in AutomatedLab

AutomatedLab supports 1-tier and 2 tier deployments of PKI. This means that you can deploy a Root CA solely or you can deploy a Root CA and a Subordinate CA to this Root CA.

Also, only deployments of PKI in the same AD forest are supported. Deployment of a Root CA in one AD forest and a Subordinate CA in another AD forest where this Subordinate CA is subordinate to the first mentioned Root CA, is not supported.

## What’s next?

The next article discusses how to manage software inside your lab and how to run custom tasks leveraging the AutomatedLab infrastructure.

## The Full Script

$start = Get-Date

#Some definitions about folder paths

#This folder contains two sub folders

# -ISOs - Stores all the DVD images

# -PostInstallationActivities - any scripts to customize the environment after installation

$labSources = 'E:\LabSources'

$vmDrive = 'D:'

$labName = 'PKITypical1'

#this folder stores the XML files that contain all the information about the lab

$labPath = "$vmDrive\$labName"

#create the target directory if it does not exist

if (-not (Test-Path $labPath)) { New-Item $labPath -ItemType Directory | Out-Null }

#create an empty lab template and define where the lab XML files and the VMs will be stored

New-LabDefinition -Path $labPath -VmPath $labPath -Name $labName -ReferenceDiskSizeInGB 60

#make the network definition

Add-LabVirtualNetworkDefinition -Name $labName -IpAddress 192.168.81.1 -PrefixLength 24

#and the domain definition with the domain admin account

Add-LabDomainDefinition -Name test1.net -AdminUser administrator -AdminPassword Password1

#these images are used to install the machines

Add-LabIsoImageDefinition -Name Server2012 -Path $labSources\ISOs\en\_windows\_server\_2012\_r2\_with\_update\_x64\_dvd\_4065220.iso -IsOperatingSystem

#these credentials are used for connecting to the machines. As this is a lab we use clear-text passwords

$installationCredential = New-Object PSCredential('Administrator', ('Password1' | ConvertTo-SecureString -AsPlainText -Force))

#the first machine is the root domain controller. Everything in $labSources\Tools get copied to the machine's Windows folder

$role = Get-LabMachineRoleDefinition -Role RootDC @{ DomainFunctionalLevel = 'Win2012R2'; ForestFunctionalLevel = 'Win2012R2' }

Add-LabMachineDefinition -Name S1DC1 `

-MemoryInMb 512 `

-IsDomainJoined `

-DomainName test1.net `

-Network $labName `

-IpAddress 192.168.81.10 `

-DnsServer1 192.168.81.10 `

-InstallationUserCredential $installationCredential `

-ToolsPath $labSources\Tools `

-OperatingSystem 'Windows Server 2012 R2 SERVERDATACENTER' `

-Roles $role

#the second will be a member server configured as Root CA server. Everything in $labSources\Tools get copied to the machine's Windows folder

$role = Get-LabMachineRoleDefinition -Role CaRoot

Add-LabMachineDefinition -Name S1ROOTCA1 `

-MemoryInMb 512 `

-Network $labName `

-IpAddress 192.168.81.20 `

-DnsServer1 192.168.81.10 `

-InstallationUserCredential $installationCredential `

-ToolsPath $labSources\Tools `

-OperatingSystem 'Windows Server 2012 R2 SERVERDATACENTER' `

-Roles $role

#the thrid will be a member server configured as Subordinate CA server. Everything in $labSources\Tools get copied to the machine's Windows folder

$role = Get-LabMachineRoleDefinition -Role CaSubordinate

Add-LabMachineDefinition -Name S2SUBCA1 `

-MemoryInMb 512 `

-IsDomainJoined `

-DomainName test1.net `

-Network $labName `

-IpAddress 192.168.81.30 `

-DnsServer1 192.168.81.10 `

-InstallationUserCredential $installationCredential `

-ToolsPath $labSources\Tools `

-OperatingSystem 'Windows Server 2012 R2 SERVERDATACENTER' `

-Roles $role

#This all has created the lab configuration in memory. Next step is to export it to XML. You could have made the

#lab definitions in XML as well or can do modifications in the XML if this seems easier.

Export-LabDefinition -Force -ExportDefaultUnattendedXml

#Set trusted hosts to '\*' and enable CredSSP

Set-LabHostRemoting

#Now the XML files needed to be reimported. Some basic checks are done for duplicate IP addresses, machine names, domain

#membership, etc. If this reports errors please run "Test-LabDefinition D:\LabSettings\Lab.xml" for more information

Import-Lab -Path (Get-LabDefinition).LabFilePath

#Now the actual work begins. First the virtual network adapter is created and then the base images per OS

#All VMs are diffs from the base.

Install-Lab -NetworkSwitches -BaseImages -VMs

#This sets up all domains / domain controllers

Install-Lab -Domains

#Install CA server(s)

Install-Lab -CA

#Start all machines what have not yet started

Install-Lab -StartRemainingMachines

$end = Get-Date

Write-Host "Setting up the lab took $($end - $start)"