Full credit is given to the above companies including the OS that this PDF file was generated!

PowerShell Get-Help on command 'New-NetIPsecPhase1AuthSet'

PS C:\Users\wahid> Get-Help New-NetIPsecPhase1AuthSet

NAME

New-NetIPsecPhase1AuthSet

SYNOPSIS

Creates a phase 1 authentication set that specifies the methods offered for main mode first authentication during IPsec negotiations.

SYNTAX

New-NetIPsecPhase1AuthSet [-AsJob] [-CimSession <CimSession[]>] [-Confirm] [-Default] [-Description <String>] -DisplayName <String> [-GPOSession <String>] [-Group <String>] [-Name <String>] [-PolicyStore <String>] -Proposal <CimInstance[]> [-ThrottleLimit <Int32>] [-WhatIf] [<CommonParameters>]

DESCRIPTION

The New-NetIPsecPhase1AuthSet cmdlet creates a set of authentication methods to use during IPsec negotiations. The first phase of authentication is typically a computer authentication method such as Kerberos v5, certificate, or pre-shared key authentication.

A phase 1 authentication set contains an ordered list of computer authentication proposals. A proposal is created by running the New-NetIPsecAuthProposal cmdlet. During the main mode negotiation, the first proposal that both peers have in common will be used for mutual authentication. A NetIPsecPhase1AuthSet object and a NetIPsecMainModeCryptoSet object provide all of the necessary security association (SA) parameters for a NetIPsecMainModeRule object. Use the Get-NetIPsecMainModeSA cmdlet to monitor the SAs that are created.

The newly created authentication set can be associated with one or more IPsec rules using the Set-NetIPsecRule cmdlet or the Set-NetIPsecMainModeRule cmdlet.

PARAMETERS

-AsJob [<SwitchParameter>]

Runs the cmdlet as a background job. Use this parameter to run commands that take a long time to complete.

-CimSession <CimSession[]>

Runs the cmdlet in a remote session or on a remote computer. Enter a computer name or a session object, such as the output of a New-CimSession (https://go.microsoft.com/fwlink/p/?LinkId=227967) or [Get-CimSession](https://go.microsoft.com/fwlink/p/?LinkId=227966)cmdlet. The default is the current session on the local computer.

-Confirm [<SwitchParameter>]

Prompts you for confirmation before running the cmdlet.

-Default [<SwitchParameter>]

Specifies the customized parameters for overriding the defaults for main mode encryption, making it the new default setting for encryption. For the default Phase1Authset object, the default Name parameter value is {E5A5D32A-4BCE-4e4d-B07F-4AB1BA7E5FE3}. To retrieve default settings,

query by using the default Name parameter value. To specify a different default cryptographic set, delete the current default set and use the Rename-NetIPsecPhase1AuthSet cmdlet to specify the default set with {E5A5D32A-4BCE-4e4d-B07F-4AB1BA7E5FE3}.

-Description <String>

Specifies that matching firewall rules of the indicated description are created. Wildcard characters are accepted. This parameter provides information about the firewall rule. This parameter specifies the localized, user-facing description of the IPsec rule.

-DisplayName <String>

Specifies that only matching firewall rules of the indicated display name are created. Wildcard characters are accepted. Specifies the localized, user-facing name of the firewall rule being created. When creating a rule this parameter is required. This parameter value is locale-dependent. If the object is not modified, this parameter value may change in certain circumstances. When writing scripts in multi-lingual environments, the Name parameter should be used instead, where the default value is a randomly assigned value. This parameter cannot be set to All.

-GPOSession <String>

Specifies the network GPO from which to retrieve the rules to be created. This parameter is used in the same way as the PolicyStore parameter. When modifying GPOs in Windows PowerShellr, each change to a GPO requires the entire GPO to be loaded, modified, and saved back. On a busy Domain Controller (DC), this can be a slow and resource-heavy operation. A GPO Session loads a domain GPO onto the local computer and makes all changes in a batch, before saving it back. This reduces the load on the DC and speeds up the Windows PowerShell cmdlets. To load a GPO Session, use the Open-NetGPO cmdlet. To save a GPO Session, use the Save-NetGPO cmdlet.

-Group <String>

Specifies that only matching firewall rules of the indicated group association are created. Wildcard characters are accepted. This parameter specifies the source string for the DisplayGroup parameter. If the DisplayGroup parameter value is a localizable string, then this parameter contains an indirect string. Rule groups can be used to organize rules by influence and allows batch rule modifications. Using the Set-NetFirewallRule cmdlets, if the group name is specified for a set of rules or sets, then all of the rules or sets in that group receive the same set of modifications. It is good practice to specify this parameter value with a universal and world-ready indirect @FirewallAPI name. The DisplayGroup parameter cannot be specified upon object creation using the New-NetFirewallRule cmdlet, but can be modified using dot-notation and the Set-NetFirewallRule cmdlet.

-Name <String>

Specifies that only matching main mode cryptographic sets of the indicated name are created. Wildcard characters are accepted. This parameter acts just like a file name, in that only one rule with a given name may exist in a policy store at a time. During group policy processing and policy merge, rules that have the same name but come from multiple stores being merged, will overwrite one another so that only one exists. This overwriting behavior is desirable if the rules serve the same purpose. For instance, all of the firewall rules have specific names, so if an administrator can copy these rules to a GPO, and the rules will override the local versions on a local computer. GPOs can have precedence. So, if an administrator has a different or more specific rule the same name in a higher-precedence GPO, then it overrides other rules that exist. The default value is a randomly assigned value. When you want to override the defaults for main mode encryption, specify the customized parameters and set this parameter value, making it the new default setting for encryption.

-PolicyStore <String>

Specifies the policy store from which to retrieve the sets to be created.

A policy store is a container for firewall and IPsec policy. The acceptable values for this parameter are:

- PersistentStore: Sometimes called static rules, this store contains the persistent policy for the local computer. This policy is not from GPOs, and has been created manually or programmatically, during application installation, on the computer. Rules created in this store are attached to the ActiveStore and activated on the computer immediately. - ActiveStore: This store contains the currently active policy, which is the sum of all policy stores that apply to the computer. This is the resultant set of policy (RSOP) for the local computer (the sum of all GPOs that apply to the computer), and the local stores (the PersistentStore, the Static Windows Service Hardening (WSH), and the Configurable WSH). ---- GPOs are also policy stores. Computer GPOs can be specified as follows. ------ `-PolicyStore hostname`.

---- Active Directory GPOs can be specified as follows.

------ `-PolicyStore domain.fqdn.com\GPO_Friendly_Namedomain.fqdn.comGPO_Friendly_Name`.

------ Such as the following.

-------- `-PolicyStore localhost`

-------- `-PolicyStore corp.contoso.com\FirewallPolicy`

---- Active Directory GPOs can be created using the New-GPO cmdlet or the Group Policy Management Console. - RSOP: This read-only store contains the sum of all GPOs applied to the local computer.

- SystemDefaults: This read-only store contains the default state of firewall rules that ship with Windows Serverr 2012.

- StaticServiceStore: This read-only store contains all the service restrictions that ship with Windows Server 2012.

Optional and product-dependent features are considered part of Windows Server 2012 for the purposes of WFAS. - ConfigurableServiceStore: This read-write store contains all the service restrictions that are added for third-party services. In addition, network isolation rules that are created for Windows Store application containers will appear in this policy store. The default value is PersistentStore. The Set-NetIPsecMainModeCryptoSet cmdlet cannot be used to add an object to a policy store. An object can only be added to a policy store at creation time with the Copy-NetIPsecMainModeCryptoSet cmdlet or with this cmdlet.

-Proposal <CimInstance[]>

Associates the specified cryptographic proposal to the corresponding cryptographic set to be used in main mode negotiations. Separate multiple entries with a comma.

-ThrottleLimit <Int32>

Specifies the maximum number of concurrent operations that can be established to run the cmdlet. If this parameter is omitted or a value of `0` is entered, then Windows PowerShellr calculates an optimum throttle limit for the cmdlet based on the number of CIM cmdlets that are running on the computer. The throttle limit applies only to the current cmdlet, not to the session or to the computer.

-WhatIf [<SwitchParameter>]

Shows what would happen if the cmdlet runs. The cmdlet is not run.

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable,

OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https:/go.microsoft.com/fwlink/?LinkID=113216).

---------- EXAMPLE 1 ----------

PS C:\>$cert1Proposal = New-NetIPsecAuthProposal -Machine -Cert -Authority "C=US,O=MSFT,CN=Microsoft Root Authority" -AuthorityType Root

PS C:\>$cert2Proposal = New-NetIPsecAuthProposal -Machine -Cert -Authority "C=US,O=MYORG,CN=\'My Organizations Root Certificate\'" -AuthorityType Root

PS C:\>$certAuthSet = New-NetIPsecPhase1AuthSet -DisplayName "Computer Certificate Auth Set" -Proposal $cert1Proposal,$cert2Proposal

PS C:\>New-NetIPSecRule -DisplayName "Authenticate with Certificates Rule" -InboundSecurity Require -OutboundSecurity Request -Phase1AuthSet $certAuthSet.Name

This example creates a rule that requires that incoming connections are authenticated by using either of two computer certificates. The computer also requests authentication for outbound connections, but allows an outbound connection if authentication is not successful.

---------- EXAMPLE 2 ----------

PS C:\>$mkerbauthprop = New-NetIPsecAuthProposal -Machine -Kerberos

```
PS C:\>$mntlmauthprop = New-NetIPsecAuthProposal -Machine -NTLM

PS C:\>$p1Auth = New-NetIPsecPhase1AuthSet -DisplayName "First Computer Auth"
-Proposal $mkerbauthprop,$mntlmauthprop

PS C:\>$ukerbauthprop = New-NetIPsecAuthProposal -User -Kerberos

PS C:\>$unentlmauthprop = New-NetIPsecAuthProposal -User -NTLM

PS C:\>$anonyauthprop = New-NetIPsecAuthProposal -Anonymous

PS C:\>$p2Auth = New-NetIPsecPhase2AuthSet -DisplayName "Second User Auth"
-Proposal $ukerbauthprop,$unentlmauthprop,$anonyauthprop

PS C:\>New-NetIPSecRule -DisplayName "Authenticate Both Computer and User"
-InboundSecurity Require -OutboundSecurity Require -Phase1AuthSet $p1Auth.Name
-Phase2AuthSet $p2Auth.Name
```

This example creates a rule that requires a first, or computer, authentication
and attempts an optional second, or user, authentication.

---------- EXAMPLE 3 ----------

PS C:\>$proposal1 = (New-NetIPsecMainModeCryptoProposal -Encryption DES3 -Hash MD5 -KeyExchange DH1)

PS C:\>$proposal2 = (New-NetIPsecMainModeCryptoProposal -Encryption AES192 -Hash MD5 -KeyExchange DH14)

PS C:\>$proposal3 = (New-NetIPsecMainModeCryptoProposal -Encryption DES3 -Hash MD5 -KeyExchange DH19)

PS C:\>$mmCryptoSet = New-NetIPsecMainModeCryptoSet -DisplayName "Main Mode Crypto Set" -Proposal $proposal1,$proposal2,$proposal3

PS C:\>New-NetIPsecMainModeRule -DisplayName "Custom Main Mode Rule" -MainModeCryptoSet $mmCryptoSet.Name

This example creates a main mode rule linked to a cryptographic set that
contains three cryptographic proposals.

---------- EXAMPLE 4 ----------

PS C:\>$cert1Proposal = New-NetIPsecAuthProposal -Machine -Cert -Authority "C=US,O=MSFT,CN=Microsoft Root Authority" -AuthorityType Root

PS C:\>$cert2Proposal = New-NetIPsecAuthProposal -Machine -Cert -Authority "C=US,O=MYORG,CN='My Organizations Root Certificate'" -AuthorityType Root

PS C:\>$certAuthSet = New-NetIPsecPhase1AuthSet -DisplayName "Computer Certificate Auth Set" -Proposal $cert1Proposal,$Cert2Proposal

PS C:\>New-NetIPsecMainModeRule -DisplayName "Main Mode Authenticate with Certificates Rule" -Phase1AuthSet $certAuthSet.Name

This example creates a main mode rule that requires that incoming connections are authenticated by using either of two computer certificates.

---------- EXAMPLE 5 ----------

PS C:\>$proposal1 = New-NetIPsecAuthProposal -Machine -Cert -Authority "C=US,O=MSFT,CN=Microsoft Root Authority" -AuthorityType Root

PS C:\>$poAuthSet = New-NetIPsecPhase1AuthSet -DisplayName "Computer Certificate Auth Set" -Proposal $proposal1

PS C:\>$proposal2 = New-NetIPsecMainModeCryptoProposal -Encryption DES3 -Hash MD5 -KeyExchange DH1

PS C:\>$mmCryptoSet = New-NetIPsecMainModeCryptoSet -DisplayName "dhgroup2:3des-sha256,3des-sha384" -Proposal $proposal2

PS C:\>New-NetIPsecMainModeRule -DisplayName "Alternate Main Mode Rule" -LocalAddress Any -RemoteAddress 192.168.0.5 -Phase1AuthSet $poAuthSet.Name -MainModeCryptoSet $mmCryptoSet.Name

This example creates a main mode rule that specifies using alternate authentication and security methods for clients that communicate with the server at address 192.168.0.5 only.

REMARKS

To see the examples, type: "get-help New-NetIPsecPhase1AuthSet -examples".
For more information, type: "get-help New-NetIPsecPhase1AuthSet -detailed".
For technical information, type: "get-help New-NetIPsecPhase1AuthSet -full".
For online help, type: "get-help New-NetIPsecPhase1AuthSet -online"