Configuration of WinRM

In this tutorial we will go through configuration of WinRM which is necessary for using WinRM connector It will cover configuration which we tested on multiple servers together with our connector. It cover just the basic stuff and if you want to study more about this topic you can use official documentation or 3rd party tutorials which will go deeper.

WinRM or Windows remote management, is a remote management protocol that uses Simple Object Access Protocol to interface with remote computers and servers, as well as Operating Systems and applications. WinRM is a command-line tool.

Check if Winrm is running

```
Test-WSMan
```

The output should be following:

If you get some error then you need to do the quick default configuration

Now execute the first command again and it should without error now.

Show current configuration

Display WinRM listener. It will show useful information about port, address, ... where WinRM is listening for incoming connections. After quick config you will probably see only one listener for HTTP.

```
winrm e winrm/config/listener
```

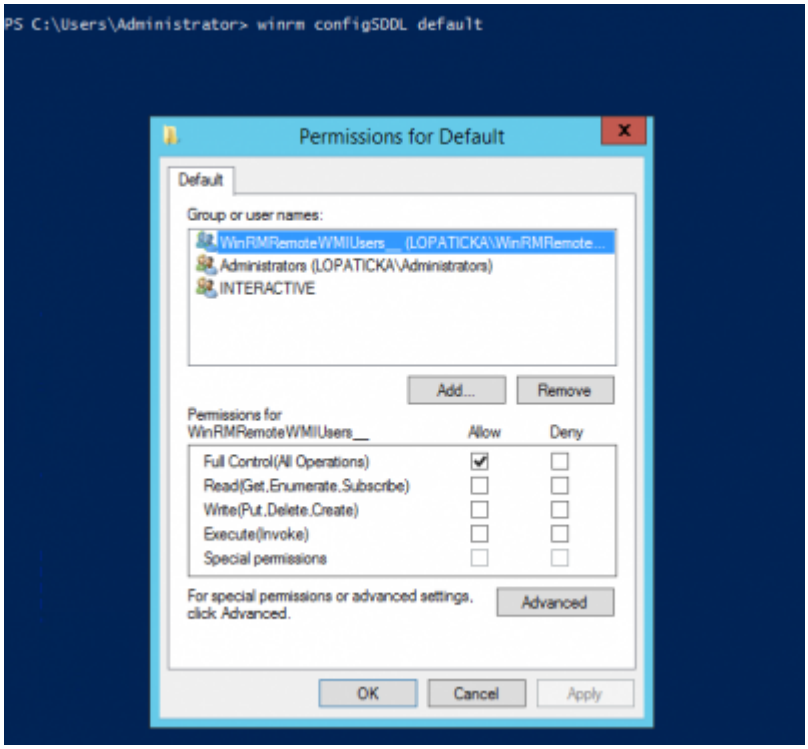
Display current winrm configuration

```
winrm get winrm/config
```

```
PS C:\Users\Administrator> winrm get winrm/config
Config
    MaxEnvelopeSizekb = 500
    MaxTimeoutms = 60000
    MaxBatchItems = 32000
    MaxProviderRequests = 4294967295
    Client
        NetworkDelayms = 5000
        URLPrefix = wsman
        AllowUnencrypted = false
        Auth
            Basic = true
            Digest = true
            Kerberos = true
            Negotiate = true
            Certificate = true
            CredSSP = false
        DefaultPorts
            HTTP = 5985
            HTTPS = 5986
        TrustedHosts
    Service
        RootSDDL = O:NSG:BAD:P(A;;GA;;;BA)(A;;GR;;;IU)(A;;GA;;;S-1-5-21-643154377-3231067919-152725009-1000)S:P(AU;FA;GA
;;;WD)(AU;SA;GXGW;;;WD)
        MaxConcurrentOperations = 4294967295
        MaxConcurrentOperationsPerUser = 1500
        EnumerationTimeoutms = 240000
        MaxConnections = 300
        MaxPacketRetrievalTimeSeconds = 120
        AllowUnencrypted = false
        Auth
            Basic = false
            Kerberos = true
            Negotiate = true
            Certificate = false
            CredSSP = true
            CbtHardeningLevel = Relaxed
        DefaultPorts
            HTTP = 5985
            HTTPS = 5986
        IPv4Filter = *
        IPv6Filter = *
        EnableCompatibilityHttpListener = false
        EnableCompatibilityHttpsListener = false
        CertificateThumbprint
        AllowRemoteAccess = true
    Winrs
        AllowRemoteShellAccess = true
        IdleTimeout = 7200000
        MaxConcurrentUsers = 10
        MaxShellRunTime = 2147483647
        MaxProcessesPerShell = 25
        MaxMemoryPerShellMB = 1024
        MaxShellsPerUser = 30
```

Show SDDL setting, this command will show dialog window

```
winrm configSDDL default
```



Authentications methods

	Type of user	Credential delegation	Message encryption
Basic	local	no	no
NTLM	local, domain	no	yes
Kerberos	domain	yes	yes
CredSSP	local, domain	yes	yes

You can configure trusted host which will be able to connect. If you don't want to specify this use

```
winrm set winrm/config/client '@{TrustedHosts="*"}'
```

We can use several methods for authentication.

- Basic - the second command will allow unencrypted data transfer, so it's not recommended to use it with HTTP. For some testing purpose it's ok.

```
winrm set winrm/config/service/auth '@{Basic="true"}'
winrm set winrm/config/service '@{AllowUnencrypted="true"}'
```

- NTLM

```
winrm set winrm/config/service/auth '@{Negotiate="true"}'
```

- Kerberos

```
winrm set winrm/config/service/auth '@{Kerberos="true"}'
```

- CredSSP

```
winrm set winrm/config/service/auth '@{CredSSP="true"}'
winrm set winrm/config/client/auth '@{CredSSP="true"}'
Enable-WSManCredSSP -Role Server
```

Permission configuration

If you want to use user which is not admin then we need a more configuration. If you want to use admin user you should ready to go even without it.

Now we need to set the right permissions. It's tested against NTLM, Kerberos and CredSSP auth It's tested with local user + group and with domain user + group. For the following steps you can use one of these groups WinRMRemoteWMIUsers__ or Remote Management Users It should work with both.

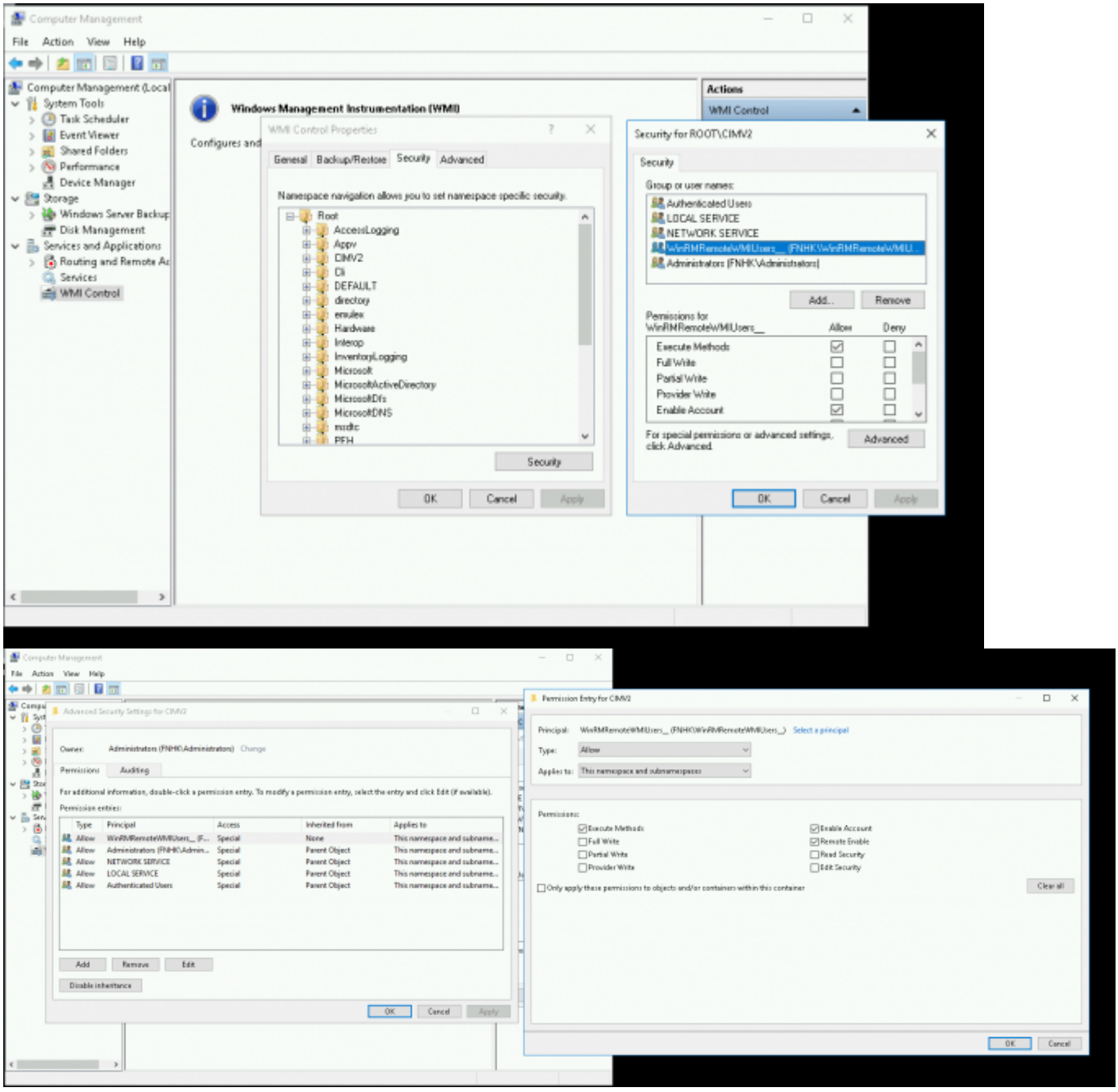
Assign user into group

Set WMI access for group.

- Computer Management → Services and Application → right click WMI Control → Properties
- In new dialog window → tab Security → Root → CIMV2 and click button Security
- Next dialog window will appear - you need to add group here
- You need to select these options in the checkboxes - Execute Methods, Enable Account and

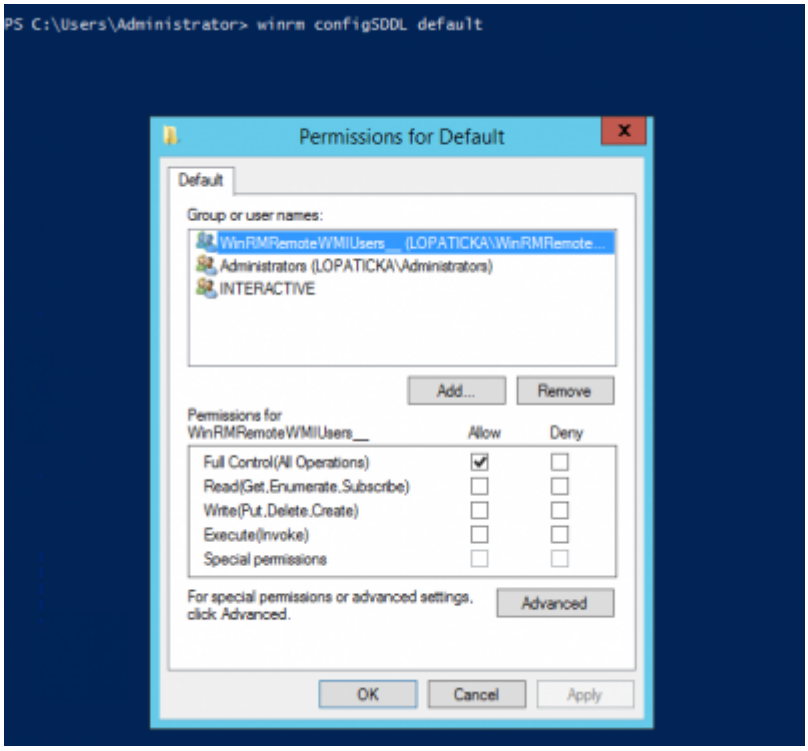
Remote Enable

- Click on Advanced - select and edit group → Set "Applies to" This namespace and subnamespaces
- Confirm all changes in dialog windows and close them



Set SDDL

- `winrm configSDDL default`



- Add group and give it Full Control
- Confirm changes

Restart WinRM

```
Restart-Service winrm
```

Debug

When you need to check if WinRM is ready for connection but you don't have access to the Windows server to check the configuration yourself use this tips.

Check if port is open and ready to connection, default ports are 5985 (HTTP) and 5986 (HTTPS): Linux

```
nc -vz HOST PORT
```

Windows

```
Test-WSMan -ComputerName HOST
or
Test-netConnection HOST -Port PORT
```

Now we know if we are able to connect to the WinRM port. In case the port is not accessible it can be probably blocked in firewall.

Next we want to try to connect to WinRM. Install pywinrm follow only the first part of installation, we don't need to install connector server. Open terminal (Linux) or powershell (Windows)

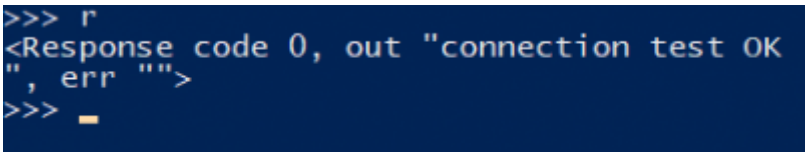
```
> python
>>> import winrm
```

```
>>> s = winrm.Session('http://HOST:5985/wsman', auth=('USER', 'PASS'),
transport='ntlm')
>>> r = s.run_ps('Write-Host connection test OK')
>>> r
```

For connecting via HTTPS use this lane. The difference is in URL where we need to use https and port 5986. Then we are using one more argument where we specify path to trust store

```
>>> s = winrm.Session('https://HOST:5986/wsman', auth=(HOST, PASS),
transport='ntlm', ca_trust_path='/etc/ssl/certs/CRT.pem')
```

After executing "r" you should see this:



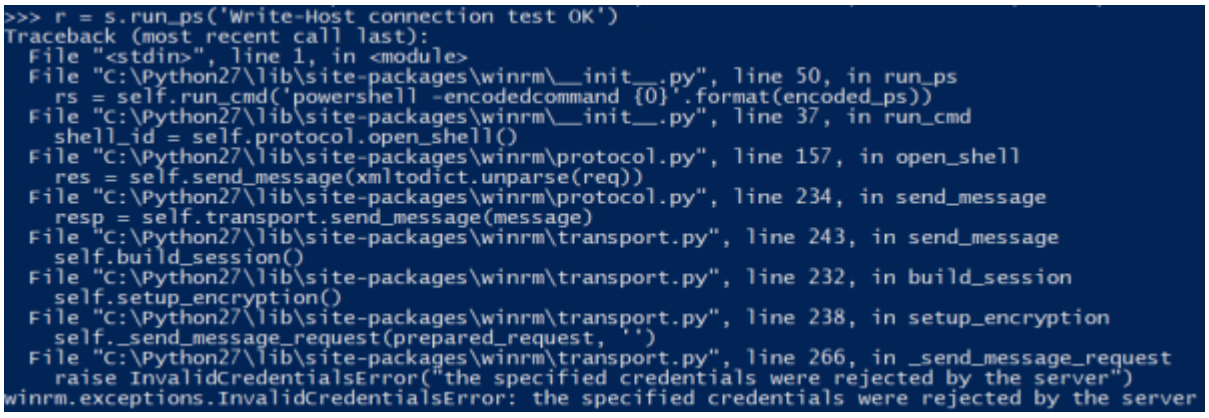
Now what we did here? We connect to WinRM via ntlm and executed command Write-Host which is just basic output to console. If there is some misconfiguration in Windows server you will probably get error after executing line

```
r = s.run_ps('Write-Host connection test OK')
```

Commons errors

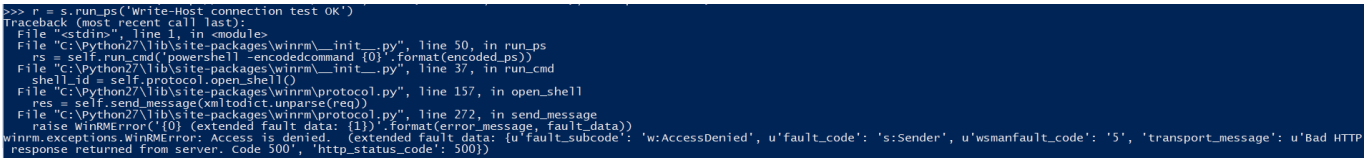
the specified credentials were rejected by the server - this error can be caused by:

- wrong username or password
- user is not in group



Access denied 500 - this error can be caused by:

- wrong username or password
- WinRM SDDL is not configured



CredSSP handshake error If you get this error when you trying to use CredSSP over HTTPS connection, the problem can be that there is configured certificate thumbprint directly in config/service class 'requests_credssp.exceptions.AuthenticationException'>("Server did not response with a CredSSP token after step Step 1. TLS Handshake - actual '",)

```
winrm set winrm/config/service '@{CertificateThumbprint=""}'
```

HTTPS support

The best case is to use HTTPS connection to connect to WinRM. To achieve this we need to do some more configuration on the server and on the client. We need to create HTTPS listener and for this we will need some certificate. In this tutorial we will cover setting up WinRM with self signed certificate. The configuration will be same if we want to use some other certificate, so if you already have certificate you can skip the part where we are generating one.

The tested way to generate self signed certificate on linux via tutorial which can be found here you should follow whole process except the part with finals steps because for our purpose we don't need to import it to browsers.

Now we have certificate which is imported in our windows server and now we can configure the HTTP listener

```
winrm create winrm/config/Listener?Address=*+Transport=HTTPS
'@{Hostname="HOSTNAME"; CertificateThumbprint="THUMBPRINT"}'
for deleting
winrm delete winrm/config/Listener?Address=*+Transport=HTTPS
```

Restart WinRM

```
Restart-Service winrm
```

Next step is to validate if we can connect to HTTPS listener so follow instruction in section debug and validate if HTTPS port is accessible. Before we try to execute some powershell command via WinRM we need to import this certificate into client trust store and pass the path to this store as parameter - see debug section