

Sonarr Multiple Instances

Requirements and how to install multiple instances of Sonarr

Multiple Instances

It is possible to run multiple instances of Sonarr. This is typically done when one wants a 4K and 1080p copy of a series.

[Note that you can configure Sonarr to use a second Sonarr as a list. This is helpful if you wish to keep both in sync.](#)



Windows Multiple Instances

Linux Multiple Instances

Docker Multiple Instances

The following requirements should be noted:


- ▶ If non-docker, the same binaries (program files) should be used
- ▶ If non-docker, all instances *must* have a `-data=` or `/data=` argument passed
- ▶ If non-docker, different ports must be used
 - ▶ If docker, different external ports must be used
- ▶ Different download client categories must be used
- ▶ Different root folders must be used.
- ▶ If non-docker, disable automatic updates on all but 1 instance.

Windows Multiple Instances

This guide will show you how to run multiple instances of Sonarr on Windows using only one base installation. This guide was put together using Windows 10; if you are using a previous version of Windows (7, 8, etc.) you may need to adjust some things. This guide also assumes that you have installed Sonarr to the default directory, and your second instance of Sonarr will be called Sonarr-4K. Feel free to change things to fit your own installations, though.

Service (Windows)

Prerequisites (Service)

- ▶ [You must have Sonarr already installed](#)
- ▶ You must have [NSSM \(Non-Sucking Service Manager\)](#)  installed. To install, download the latest release (2.24 at the time of writing) and copy either the 32-bit or 64-bit nssm.exe file to C:/windows/system32.
 - ▶ If you aren't sure if you have a 32-bit or 64-bit system, check Settings => System => About => System type.

Configuring Sonarr Service

1. Open a Command Prompt administrator window. (To run as an admin, right click on the Command Prompt icon and choose "Run as administrator.")

2. If Sonarr is running, stop the service by running `nssm stop Sonarr` in Command Prompt.

3. Now we have to edit the existing Sonarr instance to explicitly point to its data directory. The default command is as follows:

```
sc config Sonarr binpath= "C:\ProgramData\Sonarr\bin\Sonarr.exe -data=C:\ProgramData\Sonarr"
```

This command tells the original instance of Sonarr to explicitly use

`C:\ProgramData\Sonarr` for its data directory. If you didn't use the default Sonarr install, or if your data folder is somewhere else, you may have to change your paths here.

Creating Sonarr-4K Service

1. Create a new folder where you'd like Sonarr-4K to live. Most use a similar place such as

```
C:\ProgramData\Sonarr-4K
```

2. Back in Command Prompt, create the new Sonarr-4K service using `nssm install Sonarr-4K`. A popup window will open where you can type your parameters for the new instance. For this example, we will use the following:

- ▶ Path: `C:\ProgramData\Sonarr\bin\Sonarr.exe`
- ▶ Startup directory: `C:\ProgramData\Sonarr\bin`
- ▶ Arguments: `-data=C:\ProgramData\Sonarr-4K`



Note that **Arguments** points to the *new* folder created in step 1. This is crucial, as it keeps all the data files from both instances in separate locations.

1. Click *Install service*. The window should close and the service will now be available to run.

2. Continue to [Configuring Sonarr-4k](#)

Tray App (Windows)

Prerequisites (Tray App)

- ▶ [You must have Sonarr already installed](#)
- ▶ Sonarr must be configured with a `/data=` argument to allow multiple instances
- ▶ Navigate to the Startup Folder for the current user `%appdata%\Microsoft\Windows\Start Menu\Programs\Startup` and edit the existing shortcut if needed.

Creating Sonarr-4K Tray App

- ▶ Right click and Create New Shortcut
- ▶ Path: `C:\ProgramData\Sonarr\bin\Sonarr.exe /data=C:\ProgramData\Sonarr-4K`
- ▶ Give the shortcut a unique name such as `Sonarr-4K` and finish the wizard.
- ▶ Double click the new shortcut to run and test.
- ▶ Continue to [Configuring Sonarr-4k](#)

Configuring Sonarr-4k

- ▶ Regardless of if you used the Service Method or the Tray App: Stop both services and both Apps
- ▶ Start Sonarr-4k (Service or Tray App)
- ▶ Open up Sonarr-4k and Navigate within the app to [Settings => General => Host](#)
- ▶ Change `Port Number` from `8989` to a different port e.g. `7879` so Sonarr and Sonarr4k do not conflict
- ▶ You should now be able to start both apps
- ▶ Continue to [Dealing with Updates](#)


Dealing with Updates

- ▶ Disable automatic updates in one of your instances
- ▶ If one Sonarr instance is updated, both instances will shutdown and only the updated one will start again. To fix this, you will have to manually start the other instance, or you may want to look into using the below powershell script to address the problem.

Windows Port Checker and Restarter PowerShell Script

- ▶ When you use two Sonarr instances and one of it is updating, it will kill all instances. Only the one which is updating will come back online.
- ▶ The below powershell script should be configured as a scheduled task.
- ▶ It checks the ports and if one is not online, it will (re-)start the scheduled task to launch Sonarr.

1. Create a new File and name it SonarrInstancesChecker.ps1 with the below code.

2. Edit the script with your actual service names, IP, and ports.
3. [Create a scheduled task](#)  to run the script on a repeating schedule.

- ▶ Security Options: Enable **Run with highest privileges**
 - ▶ Otherwise the script will be unable to manipulate services
- ▶ Trigger: **On Launch**
- ▶ Repeat task every: **5** or **10** minutes
- ▶ Action: **Start a Program**
- ▶ Program/script: **powershell**
- ▶ Argument: **-File D:\SonarrInstancesChecker.ps1**
 - ▶ Be sure to use the full path to your script's location

```
#####
### SonarrInstancesChecker.ps1
#####
### Keeps multiple Sonarr Instances up by checking the port
### Please use Sonarr's Discord or Reddit for support!
### https://wiki.servarr.com/sonarr/installation#windows-multi
#####
### Version: 1.1
### Updated: 2020-10-22
### Author: reloxx13
#####

### SET YOUR CONFIGURATION HERE ###
# Set your host ip and port correctly and use your service or scheduledtask names

# (string) The type how Sonarr is starting
# "Service" (default) Service process is used
# "ScheduledTask" Task Scheduler is used
$startType = 'Service'

# (bool) Writes the log to C:\Users\YOURUSERNAME\log.txt when enabled
# $false (default)
# $true
$logToFile = $false

$instances = @(
    [pscustomobject]@{ # Instance 1
        Name = 'Sonarr-V3'; # (string) Service or Task name (default: Sonarr-V3)
```

```

    IP    = '192.168.178.12'; # (string) Server IP where Sonarr runs (default:
    Port  = '7873'; # (string) Server Port where Sonarr runs (default: 7873)
  }
  [pscustomobject]@{ # Instance 2
    Name = 'Sonarr-4K'; # (string) Service or Task name (default: Sonarr-4K)
    IP    = '192.168.178.12'; # (string) Server IP where Sonarr runs (default:
    Port  = '7874'; # (string) Server Port where Sonarr runs (default: 7874)
  }
  # If needed you can add more instances here... by uncommenting out the below
  # [pscustomobject]@{ # Instance 3
  # Name='Sonarr-3D'; # (string) Service or Task name (default: Sonarr-3D)
  # IP='192.168.178.12'; # (string) Server IP where Sonarr runs (default: 192.1
  # Port='7875'; # (string) Server Port where Sonarr runs (default: 787
  # }
)

### DONT CHANGE ANYTHING BELOW THIS LINE ###

###
# This function will write to a log file or in console output
###
function Write-Log
{
    #Will write to C:\Users\YOURUSERNAME\log.txt

    Param(
        $Message,
        $Path = "$env:USERPROFILE\log.txt"
    )

    function TS { Get-Date -Format 'hh:mm:ss' }

    #Console output
    Write-Output "[$(TS)]$Message"

    #File Output
    if ($logToFile)
    {
        "[$(TS)]$Message" | Tee-Object -FilePath $Path -Append | Write-Verbose
    }
}

Write-Log 'START ====='


```

```

81 $instances | ForEach-Object {
82     Write-Log "Check $($_.Name) $($_.IP):$($_.Port)"
83
84     $PortOpen = ( Test-NetConnection $_.IP -Port $_.Port -WarningAction SilentlyC
85
86     if (!$PortOpen)
87     {
88         Write-Log "Port $($_.Port) is closed, restart $($startType) $($_.Name)!"
89
90         if ($startType -eq 'Service')
91         {
92             Get-Service -Name $_.Name | Stop-Service
93             Get-Service -Name $_.Name | Start-Service
94         }
95         elseif ($startType -eq 'ScheduledTask')
96         {
97             Get-ScheduledTask -TaskName $_.Name | Stop-ScheduledTask
98             Get-ScheduledTask -TaskName $_.Name | Start-ScheduledTask
99         }
100        else
101        {
102            Write-Log '[ERROR] STARTTYPE UNKNOWN! USE Service or ScheduledTask !'
103        }
104    }
105    else
106    {
107        Write-Log "Port $($_.Port) is open!"
108    }
109 }
110 Write-Log 'END ====='

```

Linux Multiple Instances

- ▶ [Swizzin Users](#) 
- ▶ Non-Swizzin Users
 - ▶ Ensure your first instance has the `-data=` argument passed.
 - ▶ Temporarily stop your first instance, so you can change the second instance's port `systemctl stop sonarr`
 - ▶ Disable automatic updates on one of your Sonarr Instances`



Below is an example script to create a Sonarr4K instance. The below systemd creation script will use a data directory of `/var/lib/sonarr4k/`. Ensure the directory exists or modify it as

needed.

```
1 | cat << EOF | sudo tee /etc/systemd/system/sonarr4k.service > /dev/null
2 | [Unit]
3 | Description=Sonarr4k Daemon
4 | After=syslog.target network.target
5 | [Service]
6 | User=sonarr
7 | Group=media
8 | Type=simple
9 |
10 | ExecStart=/opt/Sonarr/Sonarr -nobrowser -data=/var/lib/sonarr4k/
11 | TimeoutStopSec=20
12 | KillMode=process
13 | Restart=on-failure
14 | [Install]
15 | WantedBy=multi-user.target
16 | EOF
```

- Reload systemd:

```
1 | sudo systemctl -q daemon-reload
```

- Enable the Sonarr4k service:

```
1 | sudo systemctl enable --now -q sonarr4k
```

Docker Multiple Instances

- Simply spin up a second Docker container with a different name, ensuring the above requirements are met.