Resolving IP to Hostname with PowerShell

By dwirch | 2016-01-15 0 Comment

Getting the hostname from an IP address (or vice versa) is no great magic, and can be done easily with PowerShell, by simply leveraging .Net to do the work.

We've all done something like this to get resolve an IP address:

```
[System.Net.Dns]::GetHostEntry("69.69.95.133").HostName
```

Or this to get an IP address from a hostname:

```
[System.Net.DNS]::GetHostAddresses("www.fortypoundhead.com").IPAddre
ssToString
```

But did you ever notice that sea of red when you specify an invalid IP address or hostname? When you do that, you'll get a big,red error message similar to the following:

```
[System.Net.Dns]::gethostentry("128.254.95.254").HostName
Exception calling "GetHostEntry" with "1" argument(s): "No such host
is known"
At line:1 char:1
+ [System.Net.Dns]::gethostentry("128.254.95.254").HostName
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Search

Search

How to Remove Notes from PowerPoint Slides

9 Ways to Fix the "No Connection" Error in Steam

How To Password Protect a PDF

How to Pair Bluetooth Headphones and Earphones

How to Make Windows 11 Look Like Windows 10

```
+ CategoryInfo : NotSpecified: (:) [], MethodInvocationException
+ FullyQualifiedErrorId : SocketException
```

I recently got asked how that nonsense could be avoided. Well, the answer is to simply encapsulate the process in a function that wraps up the .Net method to get the data back. It's pretty simple, and here is the code:

```
[CmdletBinding()]
Param(
[Parameter(Mandatory=$True,Position=1)]
[string]$IPAddress
)
# noise if we fail

$ErrorActionPreference = "silentlycontinue"
$Result = $null

# Pass the IP to .Net for name resolution.

$result = [System.Net.Dns]::gethostentry($IPAddress)

# process the results

If ($Result)
{
$MyResult = [string]$Result.HostName
}
Else
{
$MyResult = "unresolved"
}

# Send it to the output

$MyResult
```

Alternatively, you could also plug this directly into a standalone script. For example, lets say that you have a list of IP addresses in a file that you want

to resolve to host names. No problem:

```
# Get list from file, initialize
empty array

$ListOfIPs=Get-content ".\hosts.txt"
$ResultList = @()

# roll through the list, resolving as we
# go with the .Net DNS resolver

foreach ($IP in $ListOfIPs)
{
# We don't want to see any errors as we go, right?

$ErrorActionPreference = "silentlycontinue"
$Result = $null

# status to user, just so they know that something
# is still happening, then pass the current IP
# to .Net for name resolution.

write-host "resolving $IP"
$result = [System.Net.Dns]::gethostentry($IP)



# Enter into the array, with the returned results.

If ($Result)
{
$ResultList += "$IP," + [string]$Result.HostName
}
Else
{
$ResultList += "$IP,unresolved"
}
}
```

```
# send it out to a file, and inform the user we are done


$ResultList | Out-File .\resolved.txt
write-host "name resolution complete"
```

Again, pretty straightforward.

First, it grabs the list of IP addresses from a file called *hosts.txt*. Next, the script runs through the list, resolving the IP addresses (if it can), storing the results in $ResultList. Finally, the output is sent to another file called *resolved.txt*

I hope this is useful to someone, and drop a note in the comments if you have questions.

Powershell Scripting

Author: dwirch

Derek Wirch is a seasoned IT professional with an impressive career dating back to 1986. He brings a wealth of knowledge and hands-on experience that is invaluable to those embarking on their journey in the tech industry.

Related Posts

How To Create A Strong Password Generator in Python

Use PowerShell to Download the HTML of a Website

How to List All Windows Services with PowerShell

PowerShell vs Command Prompt: Pros and Cons

Post navigation

← SCOM 2012 Agent "Not Monitored"

List RDP Sessions on Remote Servers in PowerShell →

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

Name *

Email *

Website

Post Comment

This site uses Akismet to reduce spam. Learn how your comment data is processed.