Stephen... So… here is some food for though.

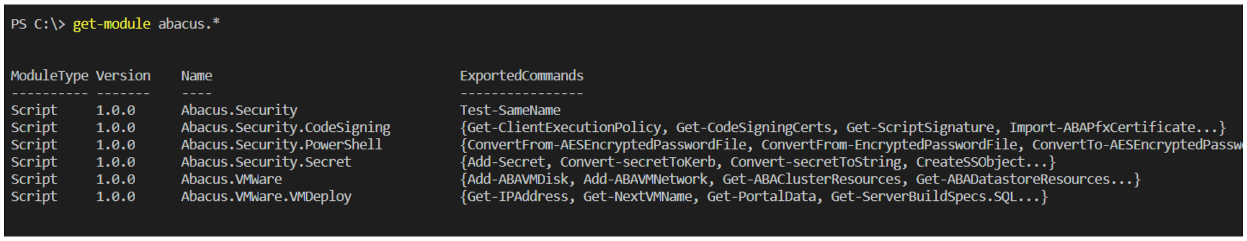
This weekend I have been continuing my research into using NameSpaces.

If someone is only writing a couple modules, or a single standalone module, a naming convention like “Abacus-ModuleName” works fine, and you see it done like this a lot because it is kind of natural and comfortable.

But when your code base grows in size and complexity, this quickly becomes unmanageable and un-scalable.

The reason they developed the NameSpace in .Net is because a lot of projects get large and complex really fast, especially in C#, and those guys needed a way to organize their code without stepping on their own code in other places, or each other’s code.

Here below, for an example, I created a couple example modules using namespaces.



```
PS C:\> get-module abacus.*


ModuleType Version    Name                              ExportedCommands
---------- -------    ----                              ----------------
Script     1.0.0      Abacus.Security                   Test-SameName
Script     1.0.0      Abacus.Security.CodeSigning       {Get-ClientExecutionPolicy, Get-CodeSigningCerts, Get-ScriptSignature, Import-ABAPfxCertificate...}
Script     1.0.0      Abacus.Security.PowerShell        {ConvertFrom-AESEncryptedPasswordFile, ConvertFrom-EncryptedPasswordFile, ConvertTo-AESEncryptedPassw
Script     1.0.0      Abacus.Security.Secret            {Add-Secret, Convert-secretToKerb, Convert-secretToString, CreateSSObject...}
Script     1.0.0      Abacus.VMWare                     {Add-ABAVMDisk, Add-ABAVMNetwork, Get-ABAClusterResources, Get-ABADatastoreResources...}
Script     1.0.0      Abacus.VMWare.VMDeploy            {Get-IPAddress, Get-NextVMName, Get-PortalData, Get-ServerBuildSpecs.SQL...}
```

What’s cool is you don’t need the **ABA** prefix in front of functions names just to make them unique.

This is because the namespace where the function lives becomes part of the name, without needing to prefix them.

But prefixes do have uses in the right circumstances. For example, my function above called “Import-ABAPfxCertificate” has the ABA prefix because that is exactly what it does, it is importing the Abacus PFX Certificate.

You don’t see VMWare or Microsoft putting “New-vmwVM” or “New-msVM” in front of their commands. That would be an example of zero scalability if PowerShell worked that way.

Their developers are using name spaces and they don’t need to do that, they both just say "New-VM", and you can call it from whichever namespace you want to use it from.

- VMWare.VimAutomation.Core\New-VM
- Hyper-V\New-VM

It wasn’t out of stupidity that Microsoft and VMWare both have the same command named **New-VM.**

That happened because that is the right way to do it, and they use namespaces in order to manage it with a scalable architecture.

This is scalable and manageable:

- Abacus.Security
- Abacus.Security.CodeSigning
- Abacus.Security.PowerShell
- Abacus.Security.PowerShell.Encryption
- Abacus.Security.Secret
- Abacus.VMWare
- Abacus.VMWare.VMDeploy

This is not scalable or manageable:

- Abacus-Security
- Abacus-PowerShell
- Abacus-CodeSigning
- Abacus-Encryption
- Abacus-Secret
- Abacus-VMWare
- Abacus-VMDeploy

What is nice about namespaces is that you can organize your code into sets and subsets, especially when your code base gets large and complex.

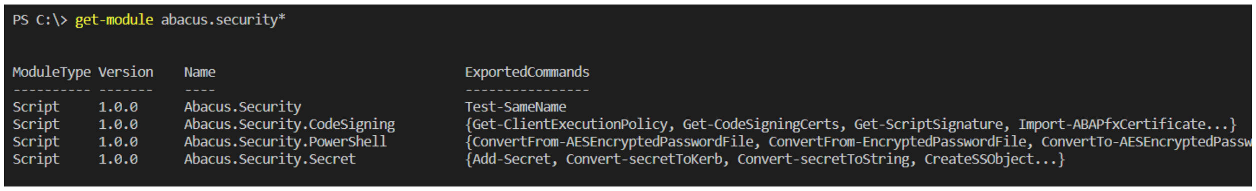
And you can call only the pieces you want without loading a bunch of other crap and functions just to use one feature or subset of a module.

Right now, the way you guys do stuff, you would need to load **all 100 functions** in the Abacus-Office365 module just to use any part of it.

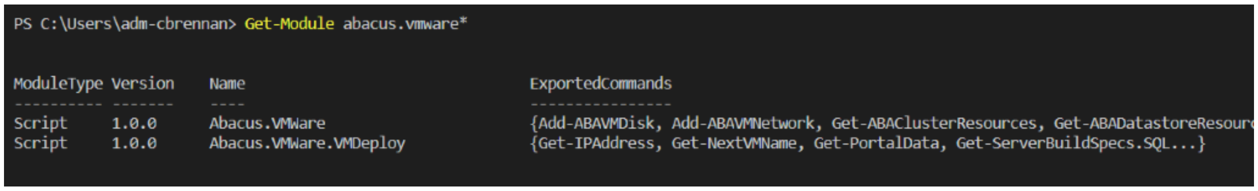
And with a namespace you can move functions around to other modules just by cut/paste without worrying about what module it currently is in or was in before, as long as it is in the root namespace, i.e. Abacus.Security.

So it becomes easier to organize, and a lot more manageable as well

Abacus Module Sets:

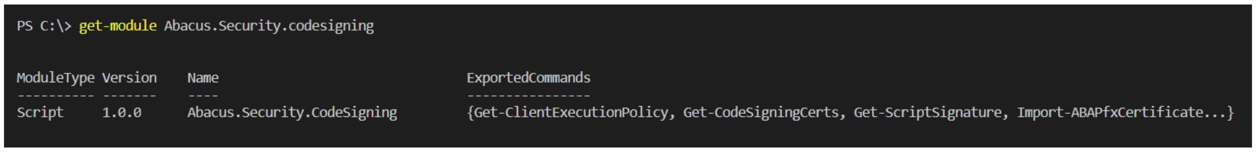


```
PS C:\> get-module abacus.security*


ModuleType Version    Name                              ExportedCommands
---------- -------    ----                              ----------------
Script     1.0.0      Abacus.Security                   Test-SameName
Script     1.0.0      Abacus.Security.CodeSigning       {Get-ClientExecutionPolicy, Get-CodeSigningCerts, Get-ScriptSignature, Import-ABAPfxCertificate...}
Script     1.0.0      Abacus.Security.PowerShell        {ConvertFrom-AESEncryptedPasswordFile, ConvertFrom-EncryptedPasswordFile, ConvertTo-AESEncryptedPassw
Script     1.0.0      Abacus.Security.Secret            {Add-Secret, Convert-secretToKerb, Convert-secretToString, CreateSSObject...}
```



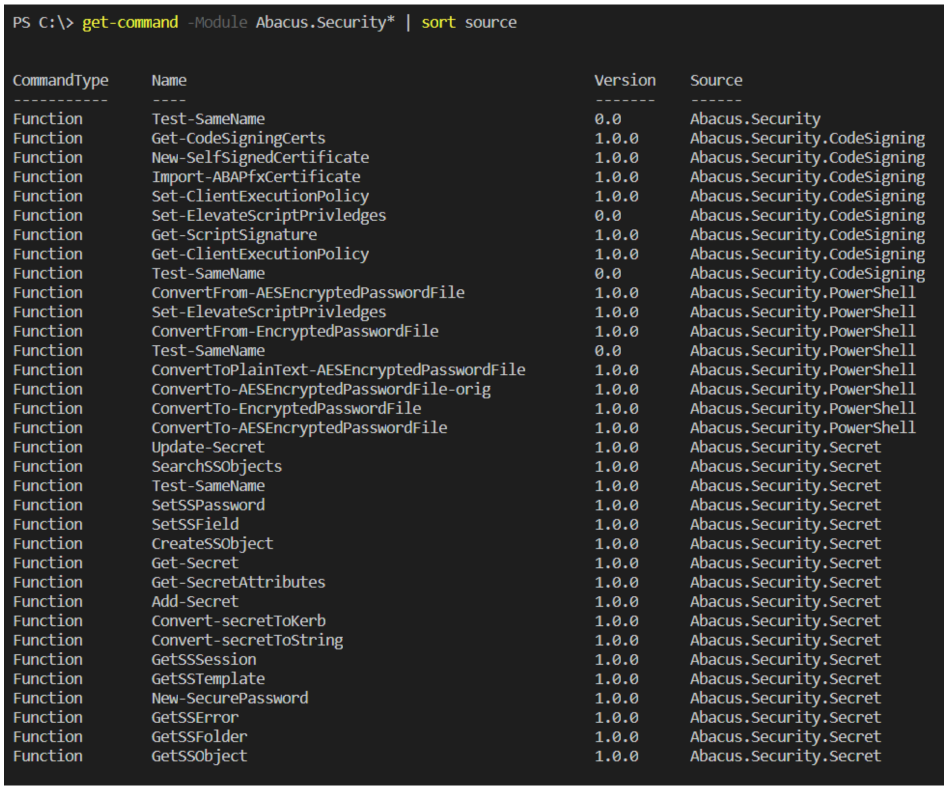
```
PS C:\Users\adm-cbrennan> Get-Module abacus.vmware*


ModuleType Version    Name                                ExportedCommands
---------- -------    ----                                ----------------
Script     1.0.0      Abacus.VMWare                       {Add-ABAVMDisk, Add-ABAVMNetwork, Get-ABAClusterResources, Get-ABADatastoreResour
Script     1.0.0      Abacus.VMWare.VMDeploy              {Get-IPAddress, Get-NextVMName, Get-PortalData, Get-ServerBuildSpecs.SQL...}
```

Abacus Module Sub-Set:



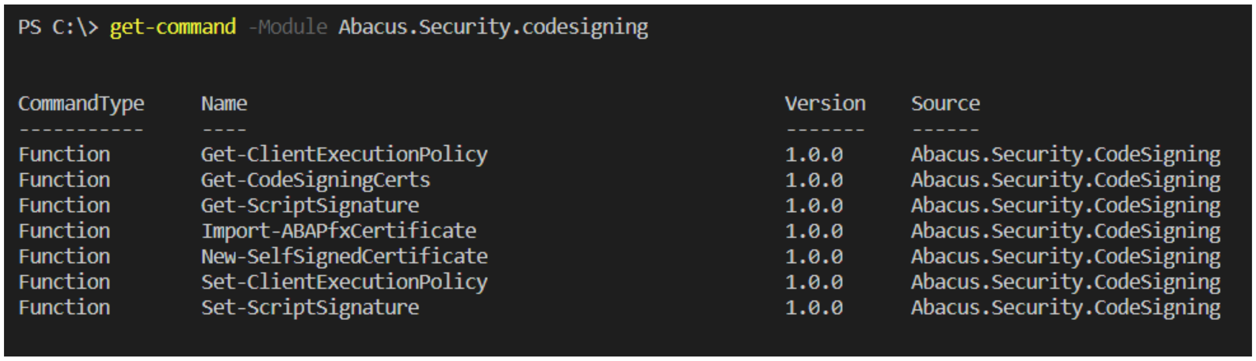
```
PS C:\> get-module Abacus.Security.codesigning


ModuleType Version    Name                              ExportedCommands
---------- -------    ----                              ----------------
Script     1.0.0      Abacus.Security.CodeSigning       {Get-ClientExecutionPolicy, Get-CodeSigningCerts, Get-ScriptSignature, Import-ABAPfxCertificate...}
```

Abacus Security Module Commands:



```
PS C:\> get-command -Module Abacus.Security* | sort source


CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Function        Test-SameName                                      0.0        Abacus.Security
Function        Get-CodeSigningCerts                               1.0.0      Abacus.Security.CodeSigning
Function        New-SelfSignedCertificate                          1.0.0      Abacus.Security.CodeSigning
Function        Import-ABAPfxCertificate                           1.0.0      Abacus.Security.CodeSigning
Function        Set-ClientExecutionPolicy                          1.0.0      Abacus.Security.CodeSigning
Function        Set-ElevateScriptPrivledges                        0.0        Abacus.Security.CodeSigning
Function        Get-ScriptSignature                                1.0.0      Abacus.Security.CodeSigning
Function        Get-ClientExecutionPolicy                          1.0.0      Abacus.Security.CodeSigning
Function        Test-SameName                                      0.0        Abacus.Security.CodeSigning
Function        ConvertFrom-AESEncryptedPasswordFile               1.0.0      Abacus.Security.PowerShell
Function        Set-ElevateScriptPrivledges                        1.0.0      Abacus.Security.PowerShell
Function        ConvertFrom-EncryptedPasswordFile                  1.0.0      Abacus.Security.PowerShell
Function        Test-SameName                                      0.0        Abacus.Security.PowerShell
Function        ConvertToPlainText-AESEncryptedPasswordFile        1.0.0      Abacus.Security.PowerShell
Function        ConvertTo-AESEncryptedPasswordFile-orig            1.0.0      Abacus.Security.PowerShell
Function        ConvertTo-EncryptedPasswordFile                    1.0.0      Abacus.Security.PowerShell
Function        ConvertTo-AESEncryptedPasswordFile                 1.0.0      Abacus.Security.PowerShell
Function        Update-Secret                                      1.0.0      Abacus.Security.Secret
Function        SearchSSObjects                                    1.0.0      Abacus.Security.Secret
Function        Test-SameName                                      1.0.0      Abacus.Security.Secret
Function        SetSSPassword                                      1.0.0      Abacus.Security.Secret
Function        SetSSField                                         1.0.0      Abacus.Security.Secret
Function        CreateSSObject                                     1.0.0      Abacus.Security.Secret
Function        Get-Secret                                         1.0.0      Abacus.Security.Secret
Function        Get-SecretAttributes                               1.0.0      Abacus.Security.Secret
Function        Add-Secret                                         1.0.0      Abacus.Security.Secret
Function        Convert-secretToKerb                               1.0.0      Abacus.Security.Secret
Function        Convert-secretToString                             1.0.0      Abacus.Security.Secret
Function        GetSSSession                                       1.0.0      Abacus.Security.Secret
Function        GetSSTemplate                                      1.0.0      Abacus.Security.Secret
Function        New-SecurePassword                                 1.0.0      Abacus.Security.Secret
Function        GetSSError                                         1.0.0      Abacus.Security.Secret
Function        GetSSFolder                                        1.0.0      Abacus.Security.Secret
Function        GetSSObject                                        1.0.0      Abacus.Security.Secret
```

Abacus CodeSigning Module Commands:



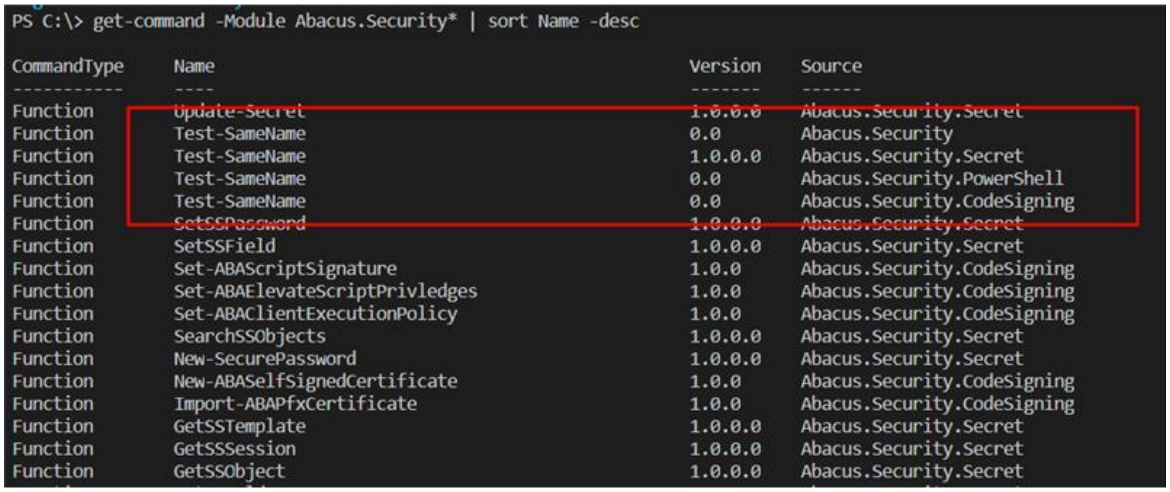
```
PS C:\> get-command -Module Abacus.Security.codesigning


CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Function        Get-ClientExecutionPolicy                          1.0.0      Abacus.Security.CodeSigning
Function        Get-CodeSigningCerts                               1.0.0      Abacus.Security.CodeSigning
Function        Get-ScriptSignature                                1.0.0      Abacus.Security.CodeSigning
Function        Import-ABAPfxCertificate                           1.0.0      Abacus.Security.CodeSigning
Function        New-SelfSignedCertificate                          1.0.0      Abacus.Security.CodeSigning
Function        Set-ClientExecutionPolicy                          1.0.0      Abacus.Security.CodeSigning
Function        Set-ScriptSignature                                1.0.0      Abacus.Security.CodeSigning
```

Function Naming:

Function names live in their own namespace, so you can write code without worrying about stepping on other function names.

Also, since the namespace where the function lives becomes part of the name, you don't need to prefix any of them with ABA to make them unique.

Here I have an example of 4 functions all with the same name, and it works fine! By design.



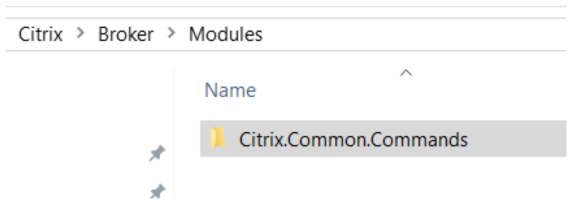
```
PS C:\> get-command -Module Abacus.Security* | sort Name -desc

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Function        Update-Secret                                      1.0.0.0    Abacus.Security.Secret
Function        Test-SameName                                      0.0        Abacus.Security
Function        Test-SameName                                      1.0.0.0    Abacus.Security.Secret
Function        Test-SameName                                      0.0        Abacus.Security.PowerShell
Function        Test-SameName                                      0.0        Abacus.Security.CodeSigning
Function        SetSSPassword                                      1.0.0.0    Abacus.Security.Secret
Function        SetSSField                                         1.0.0.0    Abacus.Security.Secret
Function        Set-ABAScriptSignature                             1.0.0      Abacus.Security.CodeSigning
Function        Set-ABAElevateScriptPrivledges                     1.0.0      Abacus.Security.CodeSigning
Function        Set-ABAClientExecutionPolicy                       1.0.0      Abacus.Security.CodeSigning
Function        SearchSSObjects                                    1.0.0.0    Abacus.Security.Secret
Function        New-SecurePassword                                 1.0.0.0    Abacus.Security.Secret
Function        New-ABASelfSignedCertificate                       1.0.0      Abacus.Security.CodeSigning
Function        Import-ABAPfxCertificate                           1.0.0      Abacus.Security.CodeSigning
Function        GetSSTemplate                                      1.0.0.0    Abacus.Security.Secret
Function        GetSSSession                                       1.0.0.0    Abacus.Security.Secret
Function        GetSSObject                                        1.0.0.0    Abacus.Security.Secret
```

This alleviates a huge naming problem!.

For example, I was handcuffed and couldn't use the logical names I wanted to name my functions in my VM-Deploy module because Bryan had already used the most logical names in his Abacus-VMWare module.

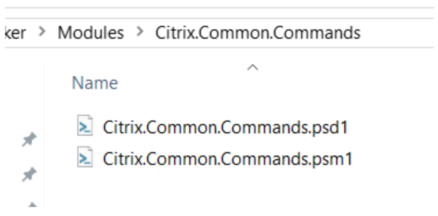
Without a namespace to organize and manage the code in a hierarchical naming convention, I couldn't name my functions in a natural way the way I wanted. Otherwise I would have been stepping on Bryan's code (function names).

Examples: from Microsoft/VMWare/Citrix


Citrix > Broker > Modules

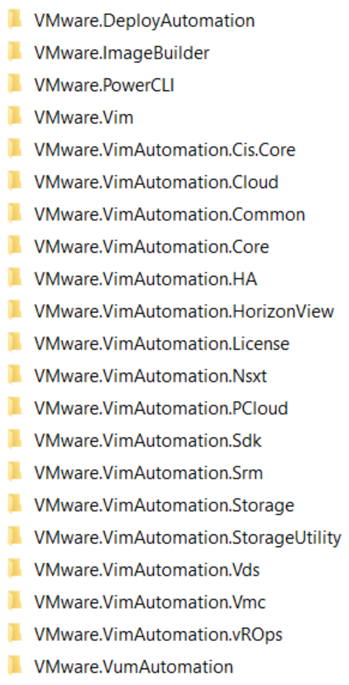
Name

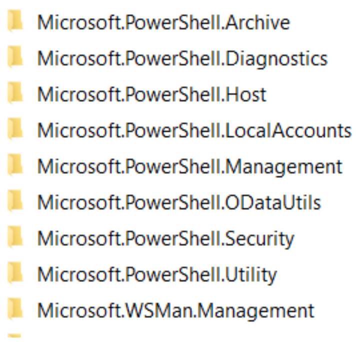
Citrix.Common.Commands


ker > Modules > Citrix.Common.Commands

Name

Citrix.Common.Commands.psd1
Citrix.Common.Commands.psm1


VMware.DeployAutomation
VMware.ImageBuilder
VMware.PowerCLI
VMware.Vim
VMware.VimAutomation.Cis.Core
VMware.VimAutomation.Cloud
VMware.VimAutomation.Common
VMware.VimAutomation.Core
VMware.VimAutomation.HA
VMware.VimAutomation.HorizonView
VMware.VimAutomation.License
VMware.VimAutomation.Nsxt
VMware.VimAutomation.PCloud
VMware.VimAutomation.Sdk
VMware.VimAutomation.Srm
VMware.VimAutomation.Storage
VMware.VimAutomation.StorageUtility
VMware.VimAutomation.Vds
VMware.VimAutomation.Vmc
VMware.VimAutomation.vROps
VMware.VumAutomation


Microsoft.PowerShell.Archive
Microsoft.PowerShell.Diagnostics
Microsoft.PowerShell.Host
Microsoft.PowerShell.LocalAccounts
Microsoft.PowerShell.Management
Microsoft.PowerShell.ODataUtils
Microsoft.PowerShell.Security
Microsoft.PowerShell.Utility
Microsoft.WSMan.Management

Summary

So now I know that a part of what I was starting to do before while I was at EZE was correct.

Using the namespaces as shown above to name and organize the module sets was the correct way to do it.

The one part I wasn't sure about even as I was doing it was how I was naming my functions.

And I was wrong about what I was doing.

I was trying to mash the namespace in with the function name.

Like this:

```
New-ECI.EMI.Automation.VM
```

I knew what I wanted, which was to tie the function name into where the function lives in the module set. But what I was doing just felt wrong for some reason.

Well… it was wrong.

This is how it supposed to be done.

```
ECI.EMI.Automation\New-VM
```

or Abacus style:

```
Abacus.VMWare.VMDeploy\New-VM
```