

PSCLOUDSTACK INTERNALS

Managing Cloudstack using Windows
PowerShell and the API Discovery Plugin



First of all: Thank you to Rohit Yadav for his work on the listApis discovery API
Without this API psCloudstack would never been created

AGENDA

14:30	Introduction
14:xx	Why psCloudstack
14:xx	psCloudstack internals
14:xx	Demo
14:55	Q C (Questions or Coffee)

A small agenda with (hopefully) most time spend on the Demo
The Q part of the Q || C can be moved to the poster session at 15:30

INTRODUCTION

\Hans van Veen, Mission Critical Engineer

\Working for Schuberg Philis since 2007

\Main areas of expertise are MS SQL, IIS, EMC & NetApp storage and Windows servers.

\30+ years of IT experience, of which 20+ years in OpenVMS.

\Love scripting!



Schuberg Philis provides outsourcing with a 100% guarantee on application availability. This can only be achieved by automating the work at hand. Well written (and documented!) scripts and tools like Chef, Jenkins, etc. are invaluable for this.

Why do I love scripting?

- I do not like to do things twice
- I am curious – curious to figure out how to do things simpler/smarter

WHY PSCLOUDSTACK

\Why psCloudstack

- Actually just because I can!
- No maintenance required!
- 'Native' PowerShell Cloudstack control

```
Windows PowerShell
PS C:\> createZone -name Bootcamp -dns1 8.8.8.8 -dns2 8.8.4.4 `
>> -internaldns1 192.168.56.11
>> -guestcidraddress "10.1.1.0/24" `
>> -networktype Advanced
>>

completionStatus      : True
affinitygroupid        : 
allocationstate       : Disabled
capacity              : 
description            : 
dhcpprovider          : VirtualRouter
displaytext           : 
dns1                  : 8.8.8.8
dns2                  : 8.8.4.4
domain                : 
domainid              : 
domainname            : 
guestcidraddress      : 10.1.1.0/24
id                    : 954e3e2e-1fcf-4302-9722-589cad0abcf4
internaldns1          : 192.168.56.11
internaldns2          : 
ip6dns1               : 
ip6dns2               : 
localstorageenabled   : false
name                  : Bootcamp
networktype           : Advanced
securitygroupsenabled : false
vlan                  : 
zonetoken             : 63025d4c-a46d-3e50-8fee-962851d8c122

PS C:\>
```

As said before, I love scripting, but I do not like doing things twice
No Perl, Python or Ruby required, keep my systems 'clean'

PSCLOUDSTACK FUNCTIONS

\Initialize-CSConfig	to create a new (or update an existing) configuration file
\Set-CSConfig	to activate a configuration
\Get-CSConfig	to read the active configuration
\Connect-CSManager	to connect to a Cloudstack server and to build the API functions of all to the user available API commands
\Invoke-CSApiCall	– the engine which issues the actual API call and returns the result

psCloudstack consists of 5 static functions; 3 to control the configuration, 1 to use the configuration to connect to the Cloudstack server and 1 to invoke commands and return results.

By default the configuration files are stored in the C:\Users\'Username'\AppData\Local folder

Set-CSConfig sets the CSCONFIGFILE environment variable to set the active configuration

Get-CSConfig uses this variable to find the active configuration

If the variable is not yet set the default

C:\Users\'Username'\AppData\Local\psCloudstack.config is used

GLOSSARY OF TERMS

API (Command)	Cloudstack API command as described in the API Documentation
API Function	PowerShell function implementing a Cloudstack API command
Cmdlet	Lightweight command used in the PowerShell environment

Apache CloudStack: API Documentation can be found at <http://cloudstack.apache.org/docs/api/>
 PowerShell Cmdlets: [http://msdn.microsoft.com/en-us/library/ms714395\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/ms714395(v=vs.85).aspx)

PowerShell uses a strict naming convention for Cmdlets but not for functions
 API functions names are identical to the Cloudstack API command name

BUILD THE POWERSHELL API FUNCTIONS

- \Connect-CSManager uses the results from listApis command to generate the API functions using a 'template'
- \Functions are 'marked' with the users API key. An API key mismatch will trigger the Connect-CSManager function.
- \2 types of API functions; synchronous and asynchronous. Asynch functions will wait for completion unless -NoWait or -Wait xxx is specified

The API key marking is meant to prevent usage of unauthorized API commands when switching configurations.

I have some slides which explain how listApis output is transformed into a PowerShell API function.

I'd rather spent my time on the demo, but for those interested look me up at the Global poster session later this afternoon.

INVOKE-CSAPICALL - THE ENGINE

- \Logic is described in "The Little Cloudstack Clients and Tools Book" by Sebastien Goasquen
- \Issues the web request and returns the result in JSON or XML format (default is XML)
- \Errors are returned as an XML formatted object

The required signature for the web request is created using the Secret and API key values from the active configuration file.
 JSON error format is 'under investigation'


```

# =====
# The 'real' psCloudstack internals
# =====

param([parameter(Mandatory=$true,ValueFromPipeline=$true)][string]$Command,
[Parameter(Mandatory = $false)][string[]]$Parameters=$null,
[Parameter(Mandatory = $false)][ValidateSet("XML","JSON")][string]$Format="XML",

# =====
# Build the query string using the provided command and parameters
# =====

$Command = ([System.Web.HttpUtility]::UrlEncode($Command)).Replace("+","%20")
$queryString = "response={0}" -f $Format.ToLower()
$Parameters%(
    if ($_.Length -gt 0)
    {
        $prmName,$prmVal = $_ -Split "=",2
        $prmVal = ([System.Web.HttpUtility]::UrlEncode($prmVal)).Replace("+","%20")
        $queryString += ("&{0}={1}" -f $prmName, $prmVal)
    }
}
Write-Verbose "Query String: $queryString"

# =====
# Create the signature, add it to the query string and invoke the webrequest
# =====

Write-Verbose "Secured web request for api call: $Command"
$cryptString = ("apikey={0}&command={1}&{2}" -f $Connect.api, $Command,$queryString).split("&")|sort -join "&"
$crypt.key = [Text.Encoding]::ASCII.GetBytes($Connect.key)
$cryptBytes = $crypt.ComputeHash([Text.Encoding]::ASCII.GetBytes($cryptString.ToLower()))
$apiSignature = [System.Web.HttpUtility]::UrlEncode([System.Convert]::ToBase64String($cryptBytes))
Write-Verbose "Signature: $apiSignature"
# =====
# The signature is ready, create the final url and invoke the web request
# =====
$protocol = "http"; if ($Connect.UseSSL) { $protocol = "https" }
$baseUrl = "{0}/{1}/{2}/client/api?" -f $protocol,$Connect.Server,$Connect.SecurePort
$csUrl = "{0}command={1}&{2}&apikey={3}&signature={4}" -f $baseUrl, $Command, $queryString,$Connect.api,$apiSignature
$prefProgress = $progressPreference; $progressPreference = 'silentlyContinue'
$Response = Invoke-WebRequest "$csUrl" -ErrorVariable iwr

```

DEMO TIME

Q|A

Questions..... or coffee.....

15:30 – Global Poster Session: Discover the 'API Discovery' plugin

Software available from:

<https://github.com/schubergphilis/psCloudstack>

Contact:

hvanveen@schubergphilis.com

If there is time -> Questions.

No time left -> Questions at the Global Poster session

FROM LISTAPIS TO PSCLOUDSTACK (1)

What listApis in general returns:

```

name      : listApis
description : lists all available apis on the server,
              provided by the Api Discovery plugin
since     : 4.1.0
isasync   : false
related   :
params    : params
response  : {name, type, response, description...}

```

FROM LISTAPIS TO PSCLOUDSTACK (2)

What listApis returns - a closer look on params;

```
params => name      : name
        description : API name
        type        : string
        length      : 255
        required    : false
```

FROM LISTAPIS TO PSCLLOUDSTACK (3)

What listApis returns - a closer look on response;

name	description	type
----	-----	----
name	the name of the api command	string
type	response field type	string
response	api response fields	set
description	description of the api	string
since	version of CloudStack the api was in...	string
params	the list params the api accepts	set
related	comma separated related apis	string
isasync	true if api is asynchronous	boolean

FROM LISTAPIS TO PSCLOUDSTACK (4)

Which translates to Powershell function equivalents;

name	=>	Function name
description	=>	Function help synopsis & description
since	=>	- not used -
isasync	=>	Powershell code segments
related	=>	Function help links
params	=>	Function parameters
response	=>	Function output