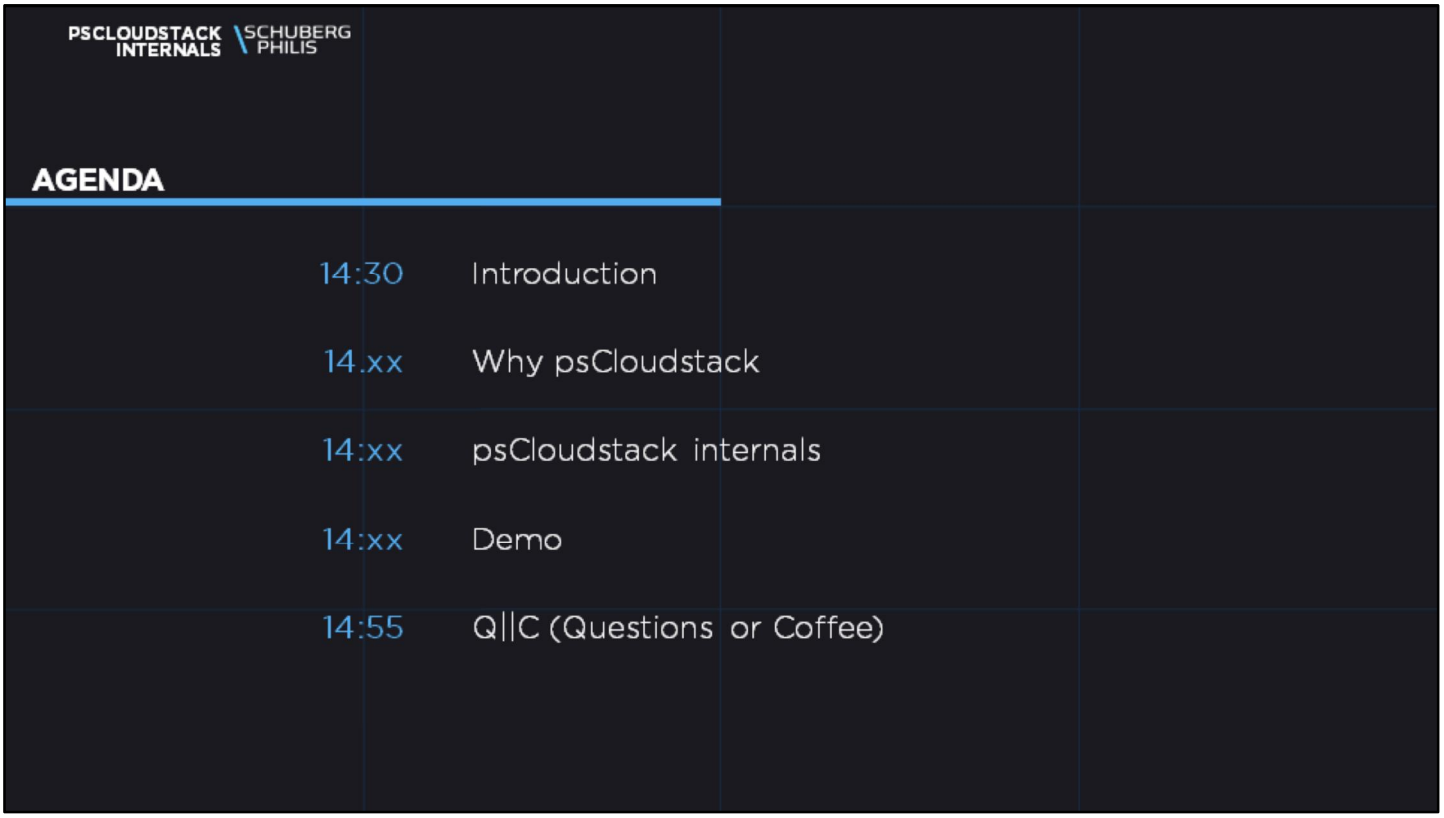
First of all: Thank you to Rohit Yadav for his work on the listApis discovery API
Whithout this API psCloudstack would never been created



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

PSCLOUDSTACK INTERNALS \SCHUBERG PHILIS

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.

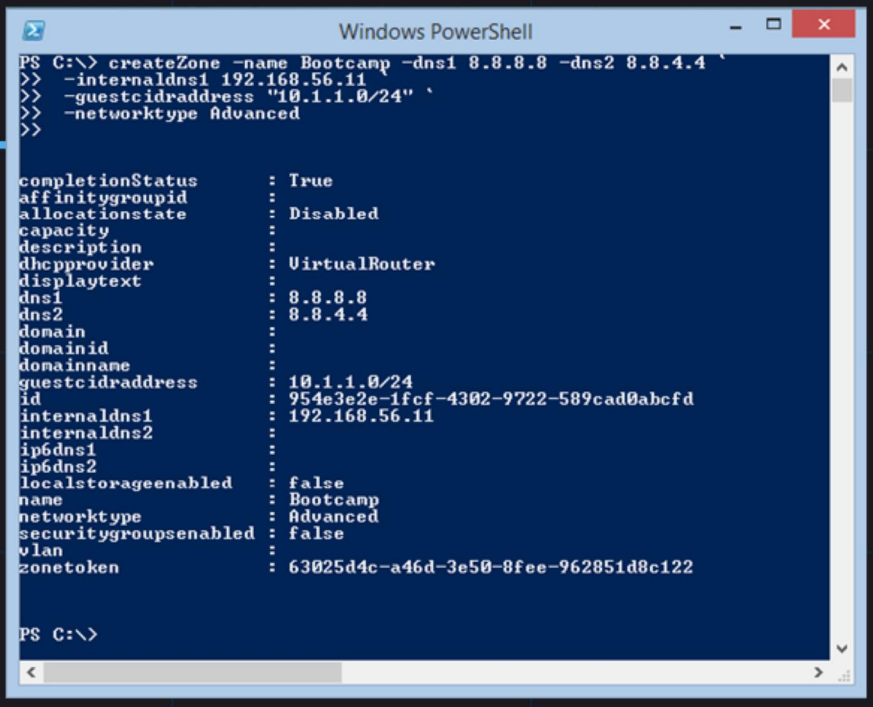
Why do I love scripting?
- I do not like to do things twice
- I am curious – curious to figure out how to do things simpler/smarter

PSCLOUDSTACK INTERNALS \SCHUBERG PHILIS

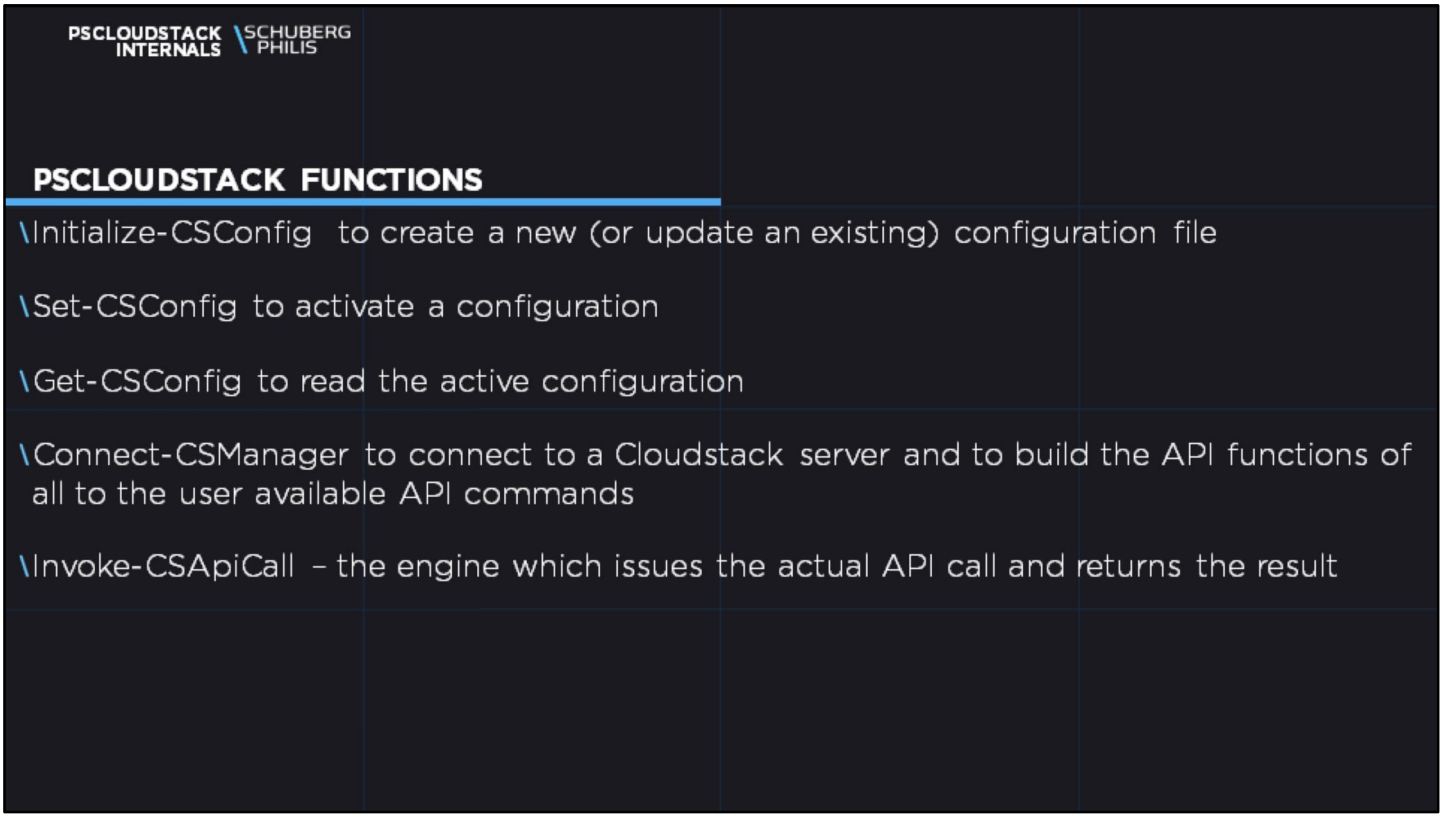
WHY PSCLOUDSTACK

\Why psCloudstack

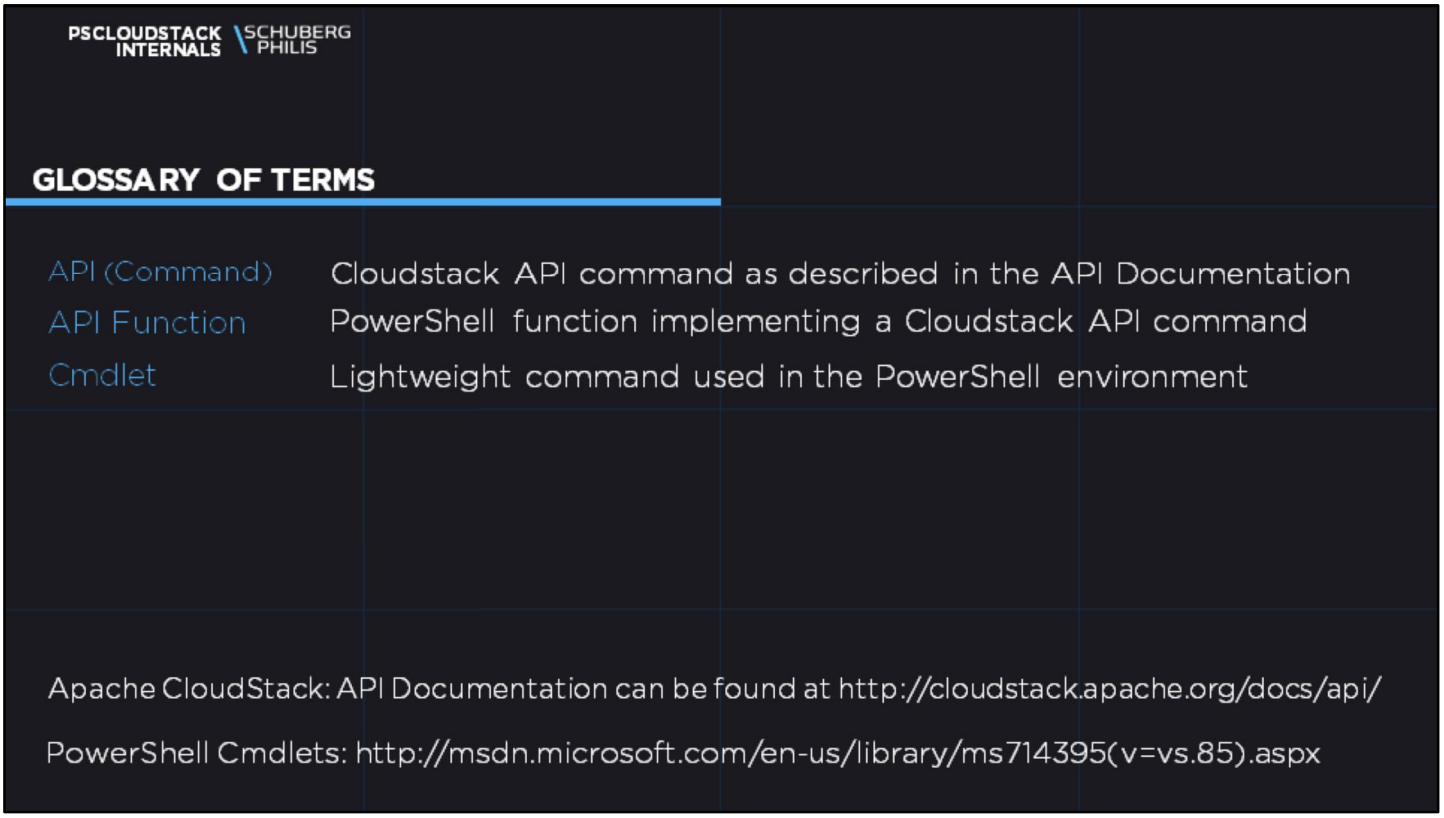
- Actually ..... just because I can!
- No maintenance required!
- 'Native' PowerShell Cloudstack control



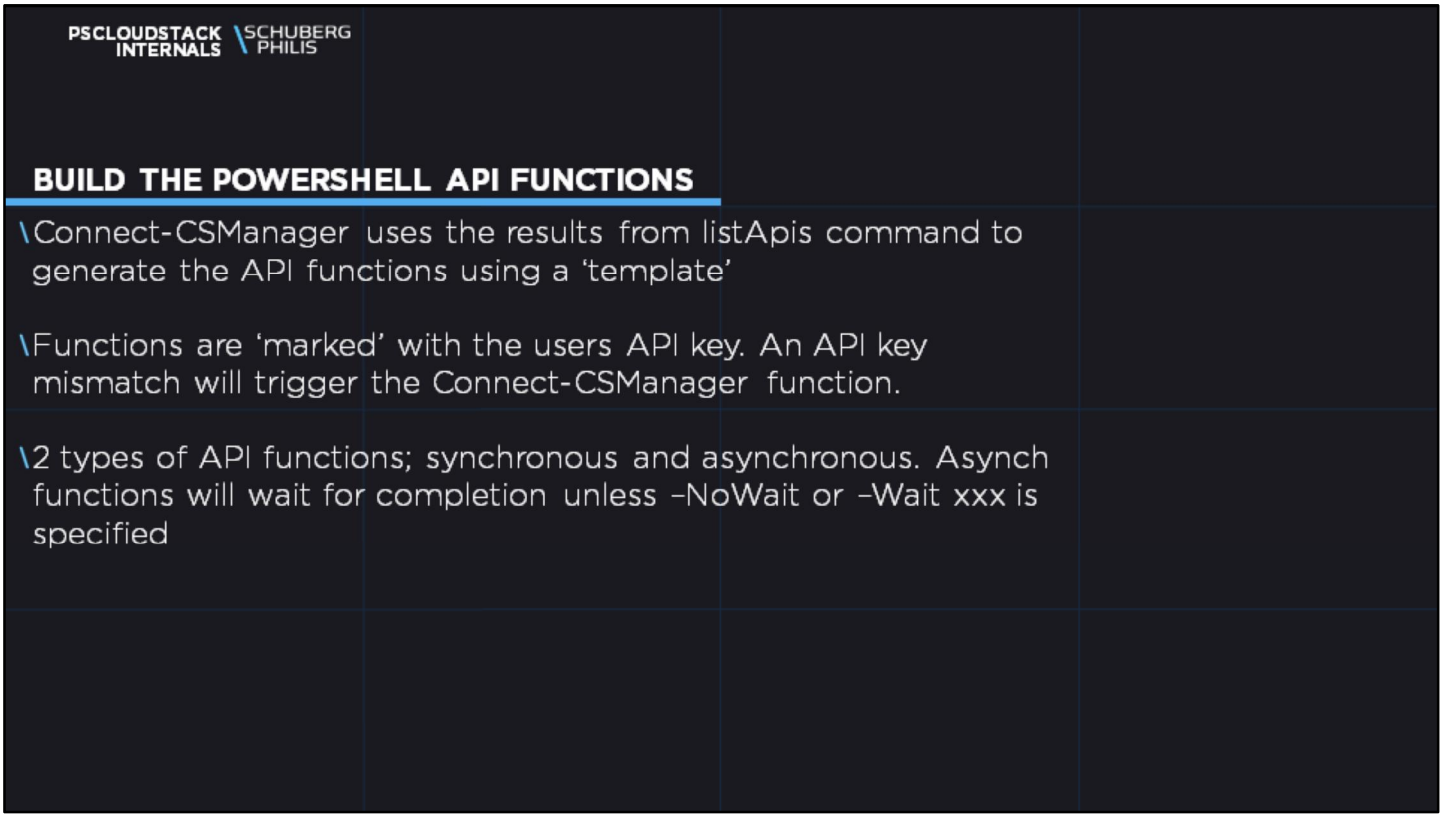
As said before, I love scripting, but I do not like doing things twice
No Perl, Python or Ruby required, keep my systems 'clean'



psCloudstack 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

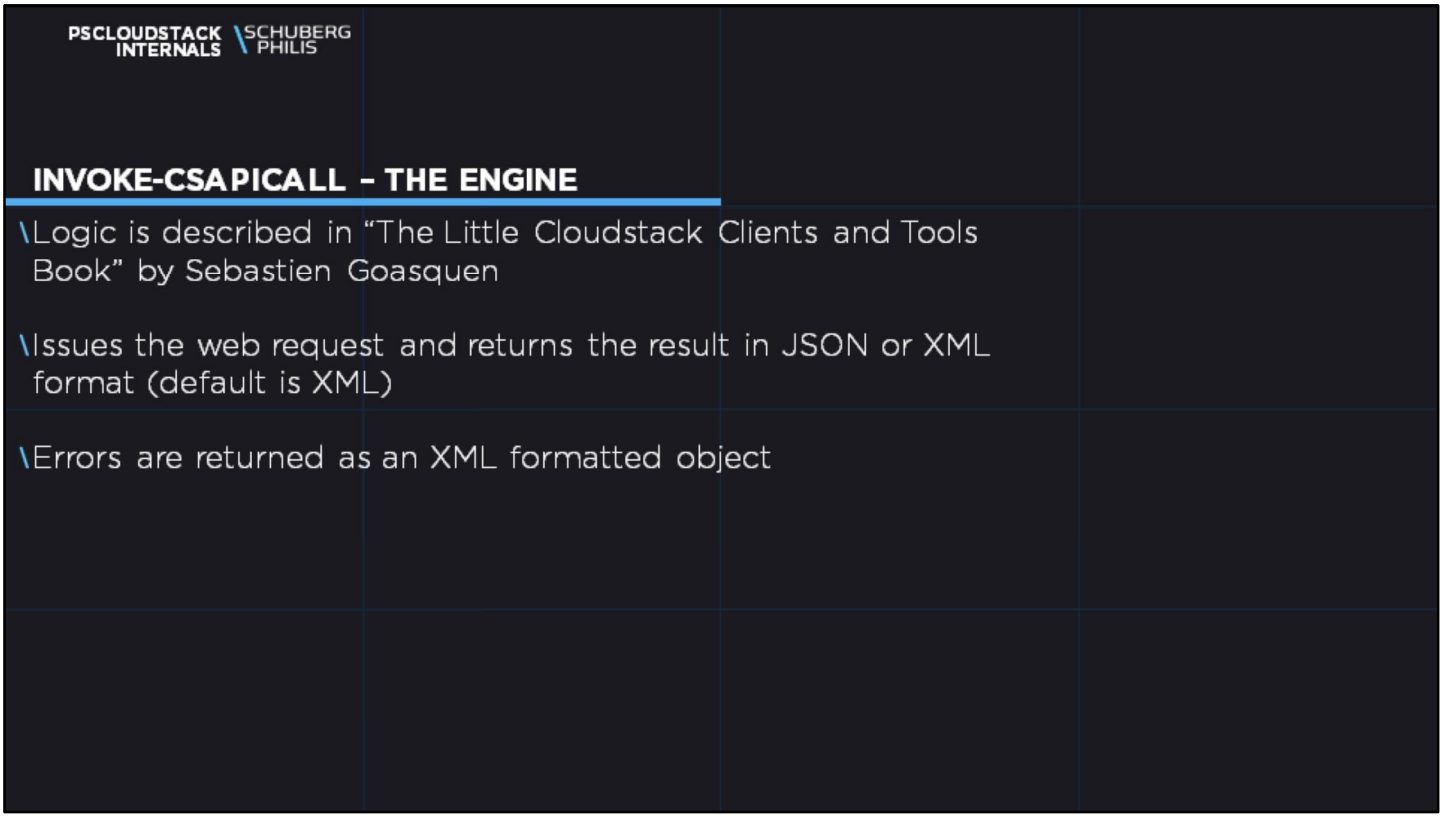


PowerShell uses a strict naming convention for Cmdlets but not for functions
API functions names are identical to the Cloudstack API command name



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

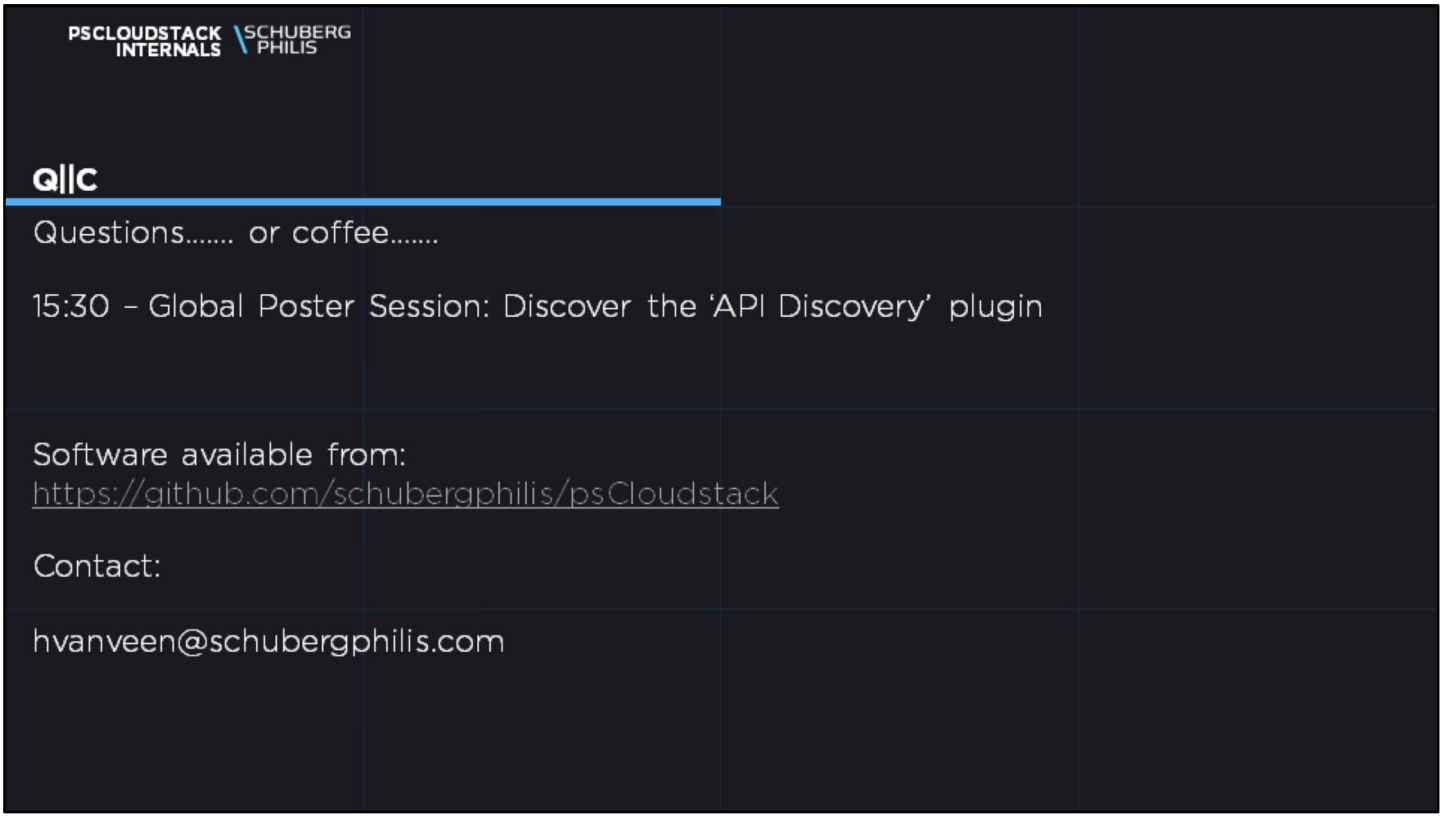
I have some slides which explain how listApis output is transformed into a PowerShell API function.
I'd rather spent my time on the demo, but for those interested look me up at the Global poster session later this afternoon.



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
```


PSCLOUDSTACK INTERNALS
SCHUBERG PHILIS
DEMO TIME



If there is time -> Questions.
No time left -> Questions at the Global Poster session

PSCLOUDSTACK INTERNALS | SCHUBERG PHILIS

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...}
```

PSCLOUDSTACK INTERNALS | SCHUBERG PHILIS

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
```

PSCLOUDSTACK INTERNALS | SCHUBERG PHILIS

FROM LISTAPIS TO PSCLOUDSTACK (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

PSCLOUDSTACK INTERNALS | SCHUBERG PHILIS

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
```