KMS activation with py-kms

Introduction

KMS activates Microsoft products on a local network, eliminating the need for individual computers to connect to Microsoft. To do this, *KMS* uses a client – server topology. *KMS* client locate *KMS* server by using *DNS* or a static configuration, then contact it by using Remote Procedure Call (*RPC*) and tries to activate against it.
KMS can activate both physical computers and virtual machines, but a network must meet or exceed the activation threshold (minimum number of computers that *KMS* requires). For activation, *KMS* clients on the network need to install a *KMS* client key (General Volume License Key, *GVLK*), so the product no longer asks Microsoft server but a user-defined server (the *KMS* server) which usually resides in a company’s intranet.

py-kms is a free open source *KMS* server emulator written in python, while Microsoft gives their *KMS* server only to corporations that signed a Select contract. Furthermore *py-kms* never refuses activation since is without restrictions, while the Microsoft *KMS* server only activates the products the customer has paid for.
py-kms supports *KMS* protocol versions 4, 5 and 6.
Although *py-kms* does neither require an activation key nor any payment, it is not meant to run illegal copies of Windows. Its purpose is to ensure that owners of legal copies can use their software without restrictions, e.g. if you buy a new computer or motherboard and your key will be refused activation from Microsoft servers due to hardware changes.

Activation with *py-kms* is achieved with the following steps:

1) Run *py-kms* on a computer in the network (this is *KMS* server or local host).
2) Install the product on client (or said remote host, which is the computer sending data to local host) and enter the *GVLK*.
3) Configure the client to use the *KMS* server.

Note that *KMS* activations are valid for 180 days, the activation validity interval, or 30 / 45 days with consumer-only products. To remain activated, *KMS* client computers must renew their activation by connecting to the *KMS* server at least once every 180 days.
For this to work, should be to guarantee that a *KMS* server is always reachable for the clients on the network.
To remember you can't activate Windows 8.1 (and above) on a *KMS* server hosted on the same machine (the *KMS* server must be a different computer than the client).

```
======================
| About GVLK keys |
======================
```

The *GVLK* keys for products sold via volume license contracts (renewal every 180 days) are published on Microsoft’s Technet web site.

Windows:
http://technet.microsoft.com/en-us/library/jj612867.aspx
Office 2010:
http://technet.microsoft.com/en-us/library/ee624355(v=office.14).aspx#section2_3
Office 2013:
http://technet.microsoft.com/en-us/library/dn385360.aspx

There are also not official keys for consumer-only versions of Windows that require activation renewal every 45 days (Windows 8.1) or 30 days (Windows 8).
A more complete and well defined list is available in the "*py-kms-ClientKeys.pdf*" file.

```
===============================
| SLMGR and OSPP commands |
===============================
```

The software License Manager (*slmgr.vbs*) is a Visual Basic script used to configure and retrieve Volume Activation information. The script can be run locally or remotely on the target computer, using the Windows-based script host (*wscript.exe*) or the command-based script host (*cscript.exe*), and administrators can specify which script engine to use. If no script engine is specified, *SLMGR* runs using the default script engine (note: it's recommended the *cscript.exe* script engine that resides in the *system32* directory).
The Software Licensing Service must be restarted for any changes to take effect. To restart it, can be used the Microsoft Management Console (*MMC*) Services or running the following command:

```
net stop sppsvc && net start sppsvc
```

The *SLMGR* requires at least one parameter. If the script is run without any parameters, it displays Help information. The general syntax of *slmgr.vbs* is as follows (using the *cscript.exe* as the script engine):

```
cscript slmgr.vbs /parameter
cscript slmgr.vbs [ComputerName] [User] [Password] [Option]
```

Command line options:
====================

[ComputerName] Name of a remote computer (default is local computer).
[User] Account with the required privilege on the remote computer.
[Password] Password for the account with required privileges on the remote compute.
[Option] Options are shown in the following table.

This table lists *SLMGR* more relevant command-line options, and a description of each. Most of the parameters configure the *KMS* host.

Global options:
==============

/ipk <ProductKey> Attempts to install a 5×5 product key for Windows or other application identified by the *ProductKey*. If the key is valid, this is installed. If a key is already installed, it's silently replaced.

/ato [ActivationID] Prompts Windows to attempt online activation, for retail and volume systems with *KMS* host key. Specifying the *ActivationID* parameter isolates the effects of the option to the edition associated with that value.

/dli [ActivationID | All] Display license information. Specifying the *ActivationID* parameter displays the license information for the specified edition associated with that *ActivationID*. Specifying *All* will display all applicable installed products’ license information. Useful for retrieve the current *KMS* activation count from the *KM*S host.

/dlv [ActivationID | All] Display detailed license information.

/xpr [ActivationID] Display the activation expiration date for the current license state.

Advanced options:
================

/cpky

Some servicing operations require the product key to be available in the registry during Out-of-Box Experience (*OOBE*) operations. So this option removes the product key from the registry to prevent from being stolen by malicious code.

/ilc <licensefile>

Installs the *licensefile* specified by the required parameter.

/rilc

Reinstalls all licenses stored in *%SystemRoot%\system32\oem* and *%SystemRoot%\System32\spp\tokens*.

/rearm

Resets the activation timers.

/rearm-app <ApplicationID>

Resets the licensing status of the specified application.

/rearm-sku <ApplicationID>

Resets the licensing status of the specified *SKU*.

/upk [ActivationID]

Uninstalls the product key of the current Windows edition. After a restart, the system will be in an unlicensed state unless a new product key is installed.

/dti [ActivationID]

Displays installation *ID* for offline activation of the *KMS* host for Windows (default) or the application that is identified when its *ActivationID* is provided.

/atp [ConfirmationID][ActivationID]

Activate product with user-provided *ConfirmationID*.

KMS client options:
==================

/skms <Name[:Port] | : port> [ActivationID]

Specifies the name and the port of the *KMS* host computer to contact. Setting this value disables auto-detection of the *KMS* host. If the *KMS* host uses *IPv6* only, the address must be specified in the format *[hostname]:port*.

/skms-domain <FQDN> [ActivationID]

Sets the specific *DNS* domain in which all *KMS SRV* records can be found. This setting has no effect if the specific single *KMS* host is set with the */skms* option. Use this option, especially in disjoint namespace environments, to force *KMS* to ignore the *DNS* suffix search list and look for *KMS* host records in the specified *DNS* domain instead.

/ckms [ActivationID]

Removes the specified *KMS* hostname, address, and port information from the registry and restores *KMS* auto-discovery behavior.

/skhc

Enables *KMS* host caching (default), which blocks the use of *DNS* priority and weight after the initial discovery of a working *KMS* host. If the system can no longer contact the working *KMS* host, discovery will be attempted again.

/ckhc

Disables *KMS* host caching. This setting instructs the client to use *DNS* auto-discovery each time it attempts *KMS* activation (recommended when using priority and weight).

/sai <ActivationInterval>

Changes how often a *KMS* client attempts to activate itself when it cannot find a *KMS* host. Replace *ActivationInterval* with a number of minutes between 15 minutes an 30 days. The default setting is 120.

/sri <RenewalInterval>	Changes how often a *KMS* client attempts to renew its activation by contacting a KMS host. Replace *RenewalInterval* with a number of minutes between 15 minutes an 30 days. The default setting is 10080 (7 days).
/sprt <PortNumber>	Sets the *TCP* communications port on a *KMS* host. It replaces *PortNumber* with the *TCP* port number to use. The default setting is 1688.
/sdns	Enables automatic *DNS* publishing by the *KMS* host.
/cdns	Disables automatic *DNS* publishing by a *KMS* host.
/spri	Sets the priority of *KMS* host processes to *Normal*.
/cpri	Set the *KMS* priority to *Low*.
/act-type [ActivationType] [ActivationID]	Sets a value in the registry that limits volume activation to a single type. *ActivationType* 1 limits activation to Active Directory only; 2 limits it to *KMS* activation; 3 to token-based activation. The 0 option allows any activation type and is the default value.

The Office Software Protection Platform script (*ospp.vbs*) can help you to configure and test volume license editions of Office client products.
You must open a command prompt by using administrator permissions and navigate to the folder that contains the script. The script is located in the folder of Office installation (\Office14 for Office 2010, \Office15 for Office 2013, \Office16 for Office 2016):

%installdir%\Program Files\Microsoft Office\Office15

If you are running 32-bit Office on a 64-bit operating system, the script is located in the folder:

%installdir%\Program Files (x86)\Microsoft Office\Office15

Running *OSPP* requires the *cscript.exe* script engine. To see the Help file, type the following command, and then press ENTER:

```
cscript ospp.vbs /?
```

The general syntax is as follows:

```
cscript ospp.vbs [Option:Value] [ComputerName] [User] [Password]
```

Command line options:
=====================

[Option] Specifies the option and value to use to activate a product, install or uninstall a product key, install and display license information, set *KMS* host name and port, and remove *KMS* host. The options and values are listed in the table below.

[ComputerName] Name of the remote computer. If a computer name is not provided, the local computer is used.

[User] Account that has the required permission on the remote computer.

[Password] Password for the account. If a user account and password are not provided, the current credentials are used.

Global options:
==============

/act Activates installed Office product keys.

/inpkey:<value> Installs a product key (replaces existing key) with a user-provided product key.

/unpkey:<value> Uninstalls an installed product key with the last five digits of the product key to uninstall (as displayed by the */dstatus* option).

/inslic:<value> Installs a license with user-provided path of the *.xrm-ms* license.

/dstatus Displays license information for installed product keys.

/dstatusall Displays license information for all installed licenses.

```
/dhistoryacterr        Displays the failure history for MAK / retail activation.

/dinstid               Displays Installation ID for offline activation.

/actcid:<value>        Activates product with user-provided ConfirmationID.

/rearm                 Resets the licensing status for all installed Office product keys.

/rearm:<value>         Resets the licensing status for an Office license with a user-provided SKUID
                       value. Use this option with the SKUID value specified by using the /dstatus
                       option if you have run out of rearms and have activated Office through KMS or
                       Active Directory-based activation to gain an additional rearm.

/ddescr:<value>        Displays the description for a user-provided error code.


KMS client options:
==================

/dhistorykms                 Displays KMS client activation history.

/dcmid                       Displays KMS client computer ID ( CMID ).

/sethst:<value>              Sets a KMS host name with a user-provided hostname.

/setprt:<value>              Sets a KMS port with a user-provided port number.

/remhst                      Removes KMS hostname ( sets port to default ).

/cachst:<value>              Allows or denies KMS host caching. Parameter value can be TRUE or FALSE.

/actype:<value>              ( Windows 8 and later only ) Sets volume activation type. Parameter value
                             can be: 1 ( for Active Directory-based ), 2 ( for KMS ), 0 ( for both ).

/skms-domain:<value>         ( Windows 8 and later only ) Sets the specific DNS domain in which all KMS
                             SRV records can be found. This setting has no effect if the specific
                             single KMS host is set by the /sethst option. Parameter value is the Fully
                             Qualified Domain Name ( FQDN ).
```

/ckms-domain (Windows 8 and later only) Clears the specific DNS domain in which all *KMS SRV* records can be found. The specific *KMS* host is used if it is set by the */sethst* option. Otherwise, auto-discovery of the KMS host is used.

============================

| Activation Procedure |

============================

The product asks for a key during installation. So it needs to enter the *GVLK*.
Then user can set the product to use, while *KMS* server must already be running on server machine
and activation occurs automatically. Finally can be enabled specific commands to speed up the process, more
useful later to extend activation for another 180 (or 45) days.

Resuming ('*//nologo*' option of *cscript* needs to hide startup logo):

Windows

=======

```
cd Windows\System32\
cscript //nologo slmgr.vbs /upk
                                    ( facoltative, to uninstall existing product key )
cscript //nologo slmgr.vbs /ipk XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
                                    ( put your product's GVLK )
cscript //nologo slmgr.vbs /skms kms-server[:tcp-port]
                                    ( example, "cscript //nologo slmgr.vbs /skms 192.168.0.100:1688" )
cscript //nologo slmgr.vbs /ato
cscript //nologo slmgr.vbs /dlv
                                    ( facoltative, to view license informations )
```

Office
======

Note that you’ll have to install a volume license (*VL*) version of Office. Office versions downloaded from *MSDN* and/or *Technet* are non-*VL*.

```
cd ProgramFiles\Microsoft Office\OfficeXX
                                    ( XX = 14 for Office 2010, 15 for Office 2013, 16 for Office 2016 )
cscript //nologo ospp.vbs /inpkey:XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
                                    ( put your product's GVLK )
cscript //nologo ospp.vbs /sethst:kms-server
                                    ( example, "cscript //nologo ospp.vbs /sethst:192.168.0.100" )
cscript //nologo ospp.vbs /setprt:tcp-port
                                    ( example, "cscript //nologo ospp.vbs /setprt:1688" )
cscript //nologo ospp.vbs /act
cscript //nologo ospp.vbs /dstatus
                                    ( facoltative, to view license informations )
```

==================
| py-kms Usage |
==================

server.py
=========

<u>How to run server.py manually:</u>

* *Linux users*:

```
user@user ~ $ cd ~/path/to/folder/py-kms
user@user ~/path/to/folder/py-kms $ python server.py [options]
```

Using "*ifconfig*" command you can get your *KMS IP*:

```
      user@user ~/path/to/folder/py-kms $ ifconfig

            eth0 Link encap: Ethernet HWaddr xx:xx:xx:xx.....
            inet addr: 192.168.1.102 Bcast 192.168.1.255 Mask: 255.255.255.0
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
```

```
RX Packets: 6 errors: 0 dropped, etc.. 0
TX packets: 3 errors:0, etc.. 0
colisions: 0 txqueuelen: 1000
RX bytes: 1020 TX Bytes: 708

lo Link encap: Local Loopback
inet addr: 127.0.0.1 Mask 255.0.0.0
UP Loopback running MTU: 65536 Metric: 1
RX packets 4: errors: 0 etc 0
TX packets 4: errors: 0 etc 0
```

In the example above is 192.168.1.102, so is valid:

```
user@user ~/path/to/folder/py-kms $ python server.py 192.168.1.102 1688
```

To stop "*server.py*", in the same bash window where code running, more simply press CTRL+C. Alternatively use "*kill <pid>*" command (you can type "*ps aux*" first and have the process <*pid*>) or "*killall <name_of_server>*" in a new bash window.

- *Windows users*:

  !! Guide to complete !!

How to run server.py automatically at start:

- *Linux users*:

  copy all files from *py-kms* folder to */usr/bin*, then:

```
echo 'kms:x:501:65534::/nonexistent:/bin/false' >> /etc/passwd
echo 'kms:*:16342:0:99999:7:::' >> /etc/shadow

echo '[Unit]' > /etc/systemd/system/pykms.service
echo 'Description=PyKMS Server' >> /etc/systemd/system/pykms.service
echo 'After=multi-user.target' >> /etc/systemd/system/pykms.service
echo >> /etc/systemd/system/pykms.service

echo '[Service]' >> /etc/systemd/system/pykms.service
echo 'ExecStart=/usr/bin/python /usr/bin/server.py' >> /etc/systemd/system/pykms.service
echo 'Restart=always' >> /etc/systemd/system/pykms.service
echo 'RestartSec=1' >> /etc/systemd/system/pykms.service
echo 'Type=simple' >> /etc/systemd/system/pykms.service
echo >> /etc/systemd/system/py-kms.service
echo '[Install]' >> /etc/systemd/system/pykms.service
echo 'WantedBy=multi-user.target' >> /etc/systemd/system/pykms.service

systemctl enable pykms.service
systemctl start pykms.service
```

* *Windows users*:

!! Guide to complete !!

Options:

ip <IPADDRESS>

Instructs *py-kms* to listen on *IPADDRESS* (can be an hostname too). If this option is not specified, *IPADDRESS* 0.0.0.0 is used.

port <PORT>

Define *TCP PORT* the *KMS* service is listening on. Default is 1688.

-e or --epid <EPID>

Use *EPID* as Windows *EPID*.
Enhanced Privacy ID (*EPID*) is a cryptographic scheme for providing anonymous signatures.
If no *EPID* is specified, a random *EPID* will be generated.

-l or --lcid <LCID>

Do not randomize the locale *ID* part of the *EPID* and use *LCID* instead.
The Language Code Identifier (LCID) describes localizable information in Windows. This structure is used to identify specific languages for the purpose of customizing software for particular languages and cultures. For example, it can specify the way dates, times, and numbers are formatted as strings. It can also specify paper sizes and preferred sort order based on language elements.
The *LCID* must be specified as a decimal number (example: 1049 for "Russian - Russia"). By default *py-kms* generates a valid locale *ID* but this may lead to a value which is unlikely to occur in your country. You may want to select the locale *ID* of your country instead. See *https://msdn.microsoft.com/en-us/library/cc233982.aspx* for
a list of valid *LCID*s. Note that some of them are not recognized by *.NET Framework 4.0*.
If an *EPID* is manually specified, this setting is ignored. Default is a fixed *LCID* of 1033 (English - US).

-w or --hwid <HWID>

Use specified *HWID* for all products.
Hardware Identification is a security measure used by Microsoft upon the activation of the Windows operating system. As part of the Product Activation system, a unique *HWID* number is generated when the operating system is first installed. The *HWID* identifies the hardware components that the system is utilizing, and this number is communicated to Microsoft. Every 10 days and at every reboot the operating system will generate another *HWID* number and compare it to the original to make sure that the operating system is still running on the same device. If the two *HWID* numbers differ too much then the operating system will shut down until Microsoft reactivates the product. The theory behind *HWID* is to ensure that the operating system is not being used on any device other than the one for which it was purchased and registered.

HWID must be an 16-character string of hex characters that are interpreted as a series of 8 bytes (big endian). Default is "*364F463A8863D35F*". To auto generate the *HWID*, type "*random*".

-c or --client-count <CLIENTCOUNT>

Use this flag to specify the current *CLIENTCOUNT*. Default is 26. Remember that a number >25 is required to enable activation.

-a or --activation-interval <ACTIVATIONINTERVAL>

Instructs clients to retry activation every *ACTIVATIONINTERVAL* minutes if it was unsuccessful, e.g. because it could not reach the server. The default is 120 minutes (2 hours).

-r or --renewal-interval <RENEWALINTERVAL>

Instructs clients to renew activation every *RENEWALINTERVAL* minutes. The default is 10080 minutes (7 days).

-s or --sqlite

Use this option to store request information from unique clients in an *SQLite* database.

-v or --loglevel <{CRITICAL, ERROR, WARNING, INFO, DEBUG}>

Activate verbose logging. Use this flag to set a loglevel. The default is *ERROR*.

(example: *user@user ~/path/to/folder/py-kms $ python server.py -v INFO*
produces in "*pykms_server.log*" this messages:

```
  Mon, 12 Jun 2017 22:09:00 INFO     TCP server listening at 0.0.0.0 on port 1688.
  Mon, 12 Jun 2017 22:09:00 INFO     HWID: 364F463A8863D35F )
```

-f of --logfile <LOGFILE>

Create a "*LOGFILE.log*" logging file. The default is named "*pykms_server.log*"

(example: *user@user ~/path/to/folder/py-kms $ python server.py 192.168.1.102 8080 -f ~/path/to/folder/py-kms/newfile.log -v INFO -w random*

produces in "*newfile.log*" this messages:

```
Mon, 12 Jun 2017 22:09:00 INFO     TCP server listening at 192.168.1.102 on port 8080.
Mon, 12 Jun 2017 22:09:00 INFO     HWID: 58C4F4E53AE14224 )
```

client.py

=========

"*client.py*" is only for testing the server.
If something does not work, it may have the cause that *py-kms* does not work correctly. You can test this with the *KMS* client "*client.py*", running on the same machine where you started "*server.py*".

Options:

ip <IPADDRESS>

Define *IPADDRESS* (or hostname) of *py-kms*' *KMS* Server. This parameter is always required.

port <PORT>

Define *TCP PORT* the *KMS* service is listening on. Default is 1688.

-m or --mode <PRODUCTNAME>

Use this flag to manually specify a Microsoft *PRODUCTNAME* for testing the *KMS* server. The default is Windows81.

-c or --cmid <CMID>

Use this flag to manually specify a *CMID* to use. If no *CMID* is specified, a random one will be generated.

The Microsoft *KMS* host machine identifies *KMS* clients with a unique Client Machine ID (*CMID*, example: ae3a27d1-b73a-4734-9878-70c949815218).
For a *KMS* client to successfully activate, the *KMS* server needs to meet a threshold, which is a minimum count for *KMS* clients. Once a *KMS* server records a count which meets or exceeds threshold, *KMS* clients will begin to activate successfully. Each unique *CMID* recorded by *KMS* server adds towards the count

threshold for *KMS* clients. This are retained by the *KMS* server for a maximum of 30 days after the last activation request with that *CMID*.
Note that duplicate *CMID* only impacts on *KMS* server machine count of client machines. Once *KMS* server meets minimum threshold, *KMS* clients will activate regardless of *CMID* being unique for a subset of specific machines or not.

-n or --name <MACHINENAME>
Use this flag to manually specify an ASCII *MACHINENAME* to use. If no *MACHINENAME* is specified a random one will be generated.

-v or --loglevel <{CRITICAL, ERROR, WARNING, INFO, DEBUG}>
Activate verbose logging. Use this flag to set a loglevel. The default is *ERROR*.

-f of --logfile <LOGFILE>
Create a "*LOGFILE.log*" logging file. The default is named "*pykms_client.log*".

example:

```
user@user ~/path/to/folder/py-kms $ python server.py -v DEBUG
user@user ~/path/to/folder/py-kms $ python client.py 0.0.0.0 1688 -v DEBUG
```

If things are ok, you should see something like this:
in "*pykms_server.py*":

```
Mon, 12 Jun 2017 22:09:00 INFO     TCP server listening at 0.0.0.0 on port 1688.
Mon, 12 Jun 2017 22:09:00 INFO     HWID: 364F463A8863D35F
Mon, 12 Jun 2017 22:09:40 INFO     Connection accepted: 127.0.0.1:42708
Mon, 12 Jun 2017 22:09:40 INFO     RPC bind request received.
Mon, 12 Jun 2017 22:09:40 DEBUG    RPC Bind Request Bytes: 'A_LONG_STRING_OF_BYTES'
Mon, 12 Jun 2017 22:09:40 DEBUG    RPC Bind Request: None, None
Mon, 12 Jun 2017 22:09:40 DEBUG    RPC Bind Response: None
Mon, 12 Jun 2017 22:09:40 DEBUG    RPC Bind Response Bytes: 'A_LONG_STRING_OF_BYTES'
Mon, 12 Jun 2017 22:09:40 INFO     RPC bind acknowledged.
Mon, 12 Jun 2017 22:09:41 INFO     Received activation request.
Mon, 12 Jun 2017 22:09:41 DEBUG    RPC Message Request Bytes: 'A_LONG_STRING_OF_BYTES'
Mon, 12 Jun 2017 22:09:41 DEBUG    RPC Message Request: None
Mon, 12 Jun 2017 22:09:41 INFO     Received V6 request on Mon Jun 12 22:09:41 2017.
Mon, 12 Jun 2017 22:09:41 DEBUG    KMS Request Bytes: 'A_LONG_STRING_OF_BYTES'
Mon, 12 Jun 2017 22:09:41 DEBUG    KMS Request: None
```

```
Mon, 12 Jun 2017 22:09:41 INFO     Machine Name: 9M
Mon, 12 Jun 2017 22:09:41 INFO     Client Machine ID: 6cacf167-e3fb-432c-8412-4b345efde259
Mon, 12 Jun 2017 22:09:41 INFO     Application ID: Windows
Mon, 12 Jun 2017 22:09:41 INFO     SKU ID: Windows 8.1 Enterprise
Mon, 12 Jun 2017 22:09:41 INFO     License Status: Grace Period
Mon, 12 Jun 2017 22:09:41 INFO     Request Time: 2017-06-12 20:09:40  (UTC)
Mon, 12 Jun 2017 22:09:41 INFO     Server ePID: 03612-00206-282-283942-03-1033-14393.0000-0022017
Mon, 12 Jun 2017 22:09:41 INFO     KMS V6 Response: None
Mon, 12 Jun 2017 22:09:41 INFO     KMS V6 Structure Bytes: 'A_LONG_STRING_OF_BYTES'
Mon, 12 Jun 2017 22:09:41 DEBUG    RPC Message Response: None
Mon, 12 Jun 2017 22:09:41 DEBUG    RPC Message Response Bytes: 'A_LONG_STRING_OF_BYTES'
Mon, 12 Jun 2017 22:09:41 INFO     Responded to activation request.
Mon, 12 Jun 2017 22:09:41 INFO     Connection closed: 127.0.0.1:42708
```

in "*py-kms_client.py*":

```
Mon, 12 Jun 2017 22:09:40 INFO     Connecting to 0.0.0.0 on port 1688...
Mon, 12 Jun 2017 22:09:40 INFO     Connection successful !
Mon, 12 Jun 2017 22:09:40 DEBUG    RPC Bind Request: None, None
Mon, 12 Jun 2017 22:09:40 DEBUG    RPC Bind Request Bytes: 'A_LONG_STRING_OF_BYTES'
Mon, 12 Jun 2017 22:09:40 INFO     Sending RPC bind request...
Mon, 12 Jun 2017 22:09:40 INFO     RPC bind acknowledged.
Mon, 12 Jun 2017 22:09:40 DEBUG    Request Base Dictionary: None
Mon, 12 Jun 2017 22:09:40 INFO     Request V6 Data: None
Mon, 12 Jun 2017 22:09:41 INFO     Request V6: 'A_LONG_STRING_OF_BYTES'
Mon, 12 Jun 2017 22:09:41 DEBUG    RPC Message Request: None
Mon, 12 Jun 2017 22:09:41 DEBUG    RPC Message Request Bytes: 'A_LONG_STRING_OF_BYTES'
Mon, 12 Jun 2017 22:09:41 DEBUG    Response: 'A_LONG_STRING_OF_BYTES'
Mon, 12 Jun 2017 22:09:41 INFO     Received V6 response
Mon, 12 Jun 2017 22:09:41 INFO     KMS Host ePID: 03612-00206-282-283942-03-1033-14393.0000-0022017
Mon, 12 Jun 2017 22:09:41 INFO     KMS Host HWID: 364F463A8863D35F
Mon, 12 Jun 2017 22:09:41 INFO     KMS Host Current Client Count: 26
Mon, 12 Jun 2017 22:09:41 INFO     KMS VL Activation Interval: 120
Mon, 12 Jun 2017 22:09:41 INFO     KMS VL Renewal Interval: 10080
```

else you'll see an error message. Next try to launch "*server.py*" from another machine where *IPADDRESS* is the hostname or address of your *KMS* server. If that fails while it works locally, you’ll most likely have to configure your firewall that it accepts incoming connections on *TCP* port 1688.

| Supported Products |

- Windows Vista
- Windows 7
- Windows 8
- Windows 8.1
- Windows 10 (up to 1607)
- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2
- Windows Server 2016
- Office 2010, Project 2010, Visio 2010
- Office 2013, Project 2013, Visio 2013
- Office 2016, Project 2016, Visio 2016

Note that it is possible to activate all versions in the *VL* (Volume License) channel, so long as you provide the proper key to let Windows know that it should be activating against a *KMS* server. *KMS* activation can't be used for Retail channel products, however you can install a *VL* product key specific to your edition of Windows even if it was installed as Retail. This effectively converts Retail installation to *VL* channel and will allow you to activate from a *KMS* server. This is not valid for Office's products, so Office, Project and Visio must be volume license versions.
Newer version may work as long as the *KMS* protocol does not change.

| References |

[1] http://forums.mydigitallife.info/threads/50234-Emulated-KMS-Servers-on-non-Windows-platforms
[2] https://github.com/myanaloglife/py-kms
[3] http://wiki.mcpstars.org/computer/python_kms_server
[4] https://github.com/CNMan/balala/blob/master/pkconfig.csv
[5] https://github.com/Wind4/vlmcsd