ctys-uc-**VMW**(7)
Use-Cases for **VMW**

September 29, 2020

Contents

List of Figures

.

1 General

Some of the provided following examples date to the first release which was 2007/2008. They still are applicable, because the interface is still the same, the archived examples perform on newer versions of Server-2.x, Player-2.x+3.x and WS-7.x exactyl as on the former versions.

2 Install and Configure a VM

The installation and configuration of a VM and required basic operational functionality in current version is foreseen to be performed by the provided tools from VMware Inc.(TM). The only partial exception is the automated creation of an inventory entry - still faulty in 1.X versions - for smarter operations.

The provided configuration by the product is fully sufficient for basic operations. In addition some optional entries related to the GuestOS - such as IP-Address, OS, Distribution, etc. - could be provided either as Keyed-Comments within the original vmx-file or in a standalone conf-file. The related details are described within the document **ctys-configuration-VMW(7)** .

3 CREATE a session

The following call starts a session:

```
ctys -t vmw -a create=f:vmware/tst-ctys/tst117/tst117.vmx,reuse app2
```

The previous call contains two specifics to be mentioned. First the filename option "f:" is used, which does a string comparison against the scanned absolute filepaths of configurations files available. The evaluation could be processed from cacheDB and/or from the native filesystem on the execution target. Due to specific handling of filenames just by pattern matching the following call leads to the same result, if unambiguous of course:

```
ctys -t vmw -a create=f:vmware/tst-ctys/tst117,reuse app2
```

If this is ambiguous, e.g. due to an backup directory, the following could be used too and might solve the problem:

```
ctys -t vmw -a create=f:vmware/tst-ctys/tst117/t,reuse app2
```

The second part to be mentioned is the **reuse** flag, which initiates simply as first trial a **connect**, when this fails, the VM session is created. Thus using the **reuse** flag can lead to some smart handling of sessions, where it is no longer required to remmember whether a session is already present or not. Therefore of course the appropriate configuration of the VM for headless background mode is required.

Another specific case is the usage of a VNCviewer session for a Workstation of Version-6 or later(?). The access requires to be configured by a static port as described within the VMware product manual. The UnifiedSessionsManager provides access by usage of the <machine-address> only, because it has the knowledge how to match for example the LABEL to a stored vncport. The following example shows a simple redundant access to the proprietary VMware console **CONSOLE:VMW** and the access to **CONSOLE:VNC**. The current version of ctys supports only the enumeration of one console for each call.

```
ctys -t vmw -a create=l:tst117,console:vnc,connect app2
```

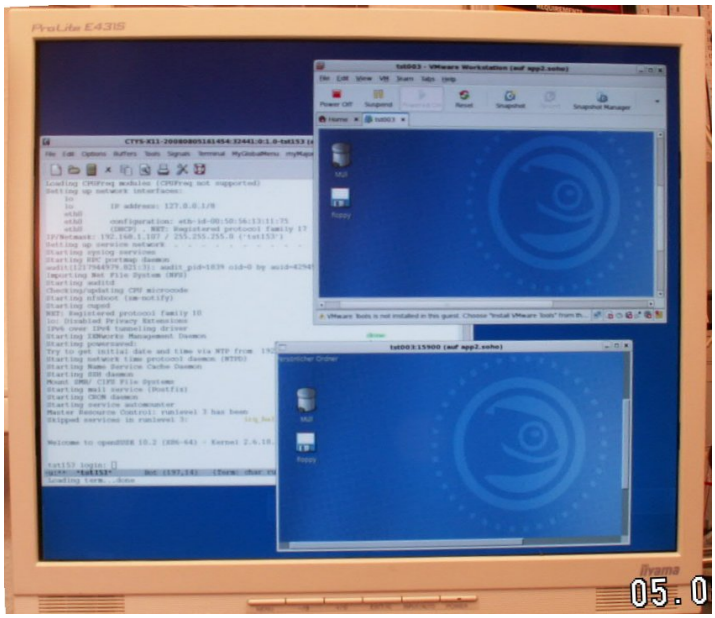

Figure 1: VMware WS6 with an additional VNCviewer Client Session.

4 CANCEL a session

The CANCEL behaviour could be widely configured for VMW. It is e.g. possible to configure an automatic close of the VM, once the GuetsOS is shutdown, when the last VM is stopped, the frontend closes too. This could be provided by command line options of VMware and is configured as default behaviour for the Unified-SessionsManager. The following call CANCELs the VMW without additional user interaction, thus any number of disconnected headless servers could be CANCELed too.

The UnifiedSessionsManager implements the standard behaviour, to try a native call to the GuestOS first, if that fails or a timeout is hit, than the VMware hypervisor interface **vmrun** is called.

```
ctys -t vmw -a cancel=f:vmware/tst-ctys/tst117/t,poweroff:0 app2
```

Additional variants are similar to the provided examples for XEN.

5 LIST sessions

The simple LIST call

```
ctys -a list app2
```

produces the output:

```
TCP-container|TCP-guest        |label  |sesstype|c|user|group
\sout{---------}+\sout{-------------}+\sout{---}+\sout{----}+-+\sout{--+-------}
ws2.soho     |-                |tst100 |VNC     |C|acue|ldapusers
ws2.soho     |ws2.soho.        |ws2    |PM      |S|-   |-
ws2.soho     |-                |tst100 |SSH(XEN)|T|acue|ldapusers
app2.soho    |-                |APP2   |VNC     |C|root|root
app2.soho    |-                |APP2   |VNC     |S|root|root
app2.soho    |tst118           |tst117 |VMW     |S|acue|ldapusers
app2.soho    |tst113           |tst112 |VMW     |S|acue|ldapusers
app2.soho    |tst118           |tst117 |VMW     |C|acue|ldapusers
app2.soho    |tst113           |tst112 |VMW     |C|acue|ldapusers
app2.soho    |app2.soho.       |app2   |PM      |S|-   |-
app2.soho    |00:E0:81:2B:A1:F2|app2   |PM      |S|-   |-
```

This is the default case for two VMs running on app2 with DISPLAYFORWARDING to ws2, and **still** runnng a local client of CLIENTFORWARDING tests for the XEN plugin. The clients and servers for VMW are now coallocated on the server app2. The CONNECTIONFORWARDING mode is currently supported for:

```
Client and Server on different machines:

  CONNECTIONFORWARDING
  -> Workstation 6+ with VNC client
  -> Server with CONSOLE

Client and Server on same machine:

  DISPLAYFORWARDING
  -> Workstation 6+ with CONSOLE
  -> Workstation 6+ with VNC client
  -> Server with CONSOLE
```

Thus the following call starts a native frontend with CONNECTIONFORWARDING on server 1.0.4 version:

```
ctys -t vmw -a create=f:vmware/tst-ctys/tst112/t,reuse -L CF olymp
```

The specifics for VMW is, that for the headless-mode initially a complete set with display forwarding is started on the remote host. ctys starts additionally a local client attached to the configured remote port(default=904) by an encrypted tunnel. The startup of the local client requires in this version an interactive user and password. As far as currently known this has to be a valid local user, a kerberos user seem snot to work. Anyhow, for test purposes here the user **root** was used, which should not be done for productive purposes.

The following list call displays now the complete set of interconnected sessions, for completeness the XEN examples are included in the output.

```
ctys -a list localhost app2 olymp lab00
```

The following listing shows the two clients connected by CONNECTIONFORWARDING, which are a vncviewer connecting as a XEN console to tst100, and a proprietary frontend of VMW connecting to tst112. Both are interconnected by usage of a SSH tunnel implicitly created by the CORE plugin DIGGER
and listed as the session type SSH(XEN) and SSH(VMW).

```
TCP-container|TCP-guest        |label   |sesstype|c|user|group
\sout{---------}+\sout{-------------}+\sout{----}+\sout{----}+-+\sout{--+--------}
ws2.soho     |-                |tst100  |VNC     |C|acue|ldapusers
ws2.soho     |tst112           |tst112  |VMW     |C|acue|ldapusers
ws2.soho     |ws2.soho.        |ws2     |PM      |S|-   |-
ws2.soho     |-                |tst100  |SSH(XEN)|T|acue|ldapusers
ws2.soho     |-                |tst112  |SSH(VMW)|T|acue|ldapusers
app2.soho    |-                |APP2    |VNC     |C|root|root
app2.soho    |-                |APP2    |VNC     |S|root|root
app2.soho    |tst118           |tst117  |VMW     |S|acue|ldapusers
app2.soho    |tst118           |tst117  |VMW     |C|acue|ldapusers
app2.soho    |app2.soho.       |app2    |PM      |S|-   |-
app2.soho    |00:E0:81:2B:A1:F2|app2    |PM      |S|-   |-
olymp.soho   |tst112           |tst112  |VMW     |S|acue|ldapusers
olymp.soho   |tst112           |tst112  |VMW     |C|acue|ldapusers
olymp.soho   |olymp.soho.      |olymp   |PM      |S|-   |-
lab00.soho   |-                |tst101  |VNC     |C|acue|ldapusers
lab00.soho   |-                |LAB00   |VNC     |C|root|root
lab00.soho   |-                |LAB00   |VNC     |S|root|root
lab00.soho   |-                |Domain-0|XEN     |S|-   |-
lab00.soho   |tst100           |tst100  |XEN     |S|-   |-
lab00.soho   |tst101           |tst101  |XEN     |S|-   |-
lab00.soho   |lab00.soho.      |lab00   |PM      |S|-   |-
```

6 ENUMERATE sessions

The following call displays the communications interfaces of the test-pool VMs. For additional information refer to User-Manual:"Display of Available Sessions".

```
ctys -a enumerate=macro:TAB\_CPORT,b:vmware/tst-ctys
```

Resulting to the display:

```
Label |stype|cport|PM      |MAC               |TCP
\sout{--}+\sout{-}+\sout{-}+\sout{----}+\sout{--------------}+\sout{-----------}
tst117|VMW  |     |ws2.soho|00:50:56:13:11:52 |192.168.1.240
tst115|VMW  |0    |ws2.soho|00:50:56:13:11:50 |192.168.1.235
tst117|VMW  |     |ws2.soho|00:50:56:13:11:52 |192.168.1.240
tst112|VMW  |     |ws2.soho|00:50:56:13:11:4D |192.168.1.235
tst003|VMW  |0    |ws2.soho|00:50:56:13:11:33 |192.168.1.133
tst005|VMW  |0    |ws2.soho|00:50:56:13:11:35 |192.168.1.135
tst103|VMW  |0    |ws2.soho|00:50:56:13:11:44 |192.168.1.223
tst106|VMW  |0    |ws2.soho|00:50:56:13:11:47 |192.168.1.226
tst111|VMW  |0    |ws2.soho|00:50:56:13:11:4C |192.168.1.234
tst120|VMW  |0    |ws2.soho|00:50:56:13:11:55 |192.168.1.208
tst128|VMW  |0    |ws2.soho|00:50:56:13:11:5C |192.168.1.212
tst002|VMW  |0    |ws2.soho|00:50:56:13:11:32 |192.168.1.132
tst111|VMW  |0    |ws2.soho|00:50:56:13:11:4C |192.168.1.234
```

7 Display of Available Sessions

Once the basic installation and setup is accomplished, first a "PATHNAME/PNAME" based start of a VM should be performed. The option **-c off** deactivates the use of the nameservice cache for an initially empty

cacheDB, thus suppresses several warnings and error messages of internally called tools.

The next step - after successful installation and configuration of the UnifiedSesssionsManager is the creation of a populated cacheDB by usage of **ctys-vdbgen** for storage of a list of actually available instances. This is by default applicable on distributed machines and is performed by default as parallel-tasks with minor dependency on the count on targets.
The following call of **ctys-vhost** lists all available VMs with given constraints, in this case all instances of VMW which could be started by the user "acue" on the host "app2". The set displayed has to be additionally of the set "tst-ctys", which is the testpool for the UnifiedSessionsManager.

```
ctys-vhost -o pm,label,ids app2 vmw acue tst-ctys
```

The **pm**, the **ids** and the **label** are displayed as a result.
The additional string 'app2 vmw acue tst-ctys' is used as a awk-regexpr and is evaluated as an AND based filter for each word. The whole query requires in this case about 1.4seconds and the following result is displayed. The average acces times are in the range of 0.6-0.8seconds in databases with about 2000 entries.

```
app2.soho;tst117;/homen/acue/vmware/tst-ctys/tst117/tst117.vmx
app2.soho;tst115;/homen/acue/vmware/tst-ctys/tst115/tst115.vmx
app2.soho;tst117;/homen/acue/vmware/tst-ctys/tst117.centos/tst117.vmx
app2.soho;tst111;/homen/acue/vmware/tst-ctys/tst111.OpenBSD-4.2/tst111.vmx
```

8 Change LIST Output by Custom Tables

The previous output, which is by default displayed in TERSE format could be formatted by a generic custom table. The following call displays the required canonical field indexes.

```
ctys-vhost -o pm,label,ids,titleidx app2 vmw acue tst-ctys
```

The indexes in title line are prefixes as an extended table title by **TITLEIDX**. The values are the so calle 'Canonical Indexes' of the database records to be used for definition of custom tables.

```
ContainingMachine(1);Label(3);ID(4)
app2.soho;tst117;/homen/acue/vmware/tst-ctys/tst117/tst117.vmx
app2.soho;tst115;/homen/acue/vmware/tst-ctys/tst115/tst115.vmx
app2.soho;tst117;/homen/acue/vmware/tst-ctys/tst117.centos/tst117.vmx
app2.soho;tst111;/homen/acue/vmware/tst-ctys/tst111.OpenBSD-4.2/tst111.vmx
```

This values could be now used to define the output table as:

```
ctys-vhost \
   -o pm,label,ids,tab\_gen:1\_PM\_7\%\%3\_label\_4\%\%4\_ID\_30 \
   app2 vmw acue tst-ctys
```

As could be seen in the following output, this table configuration is not really helpful. The field sizes are too short, and the common leading part of the pathnames for the ID fields is quite long.

```
PM     |labe|ID
\sout{---}+\sout{--+---------------------------}
app2.so|tst1|/homen/acue/vmware/tst-ctys/ts
app2.so|tst1|/homen/acue/vmware/tst-ctys/ts
app2.so|tst1|/homen/acue/vmware/tst-ctys/ts
app2.so|tst1|/homen/acue/vmware/tst-ctys/ts
```

The following changes might help in advance of usability:

```
ctys-vhost \
   -o pm,label,ids,tab\_gen:1\_PM\_11\%\%3\_label\_9\%\%4\_ID\_30\_L \
   app2 vmw acue tst-ctys
```

Although this is much more helpful, the raise of the ID value should Ahelp some more.

```
PM         |label    |ID
\sout{-------}+\sout{-----}+\sout{-------------------------}
app2.soho  |tst117   |are/tst-ctys/tst117/tst117.vmx
app2.soho  |tst115   |are/tst-ctys/tst115/tst115.vmx
app2.soho  |tst117   |-ctys/tst117.centos/tst117.vmx
app2.soho  |tst111   |/tst111.OpenBSD-4.2/tst111.vmx
```

Thus the final trial for usage and probably storage as a predefined MACRO is:

```
ctys-vhost \
   -o pm,label,ids,tab\_gen:1\_PM\_11\%\%3\_label\_9\%\%4\_ID\_50\_L app2 \
   vmw acue tst-ctys
```

The final result is:

```
PM         |label    |ID
\sout{-------}+\sout{-----}+\sout{-------------------------------------------}
app2.soho  |tst117   |/homen/acue/vmware/tst-ctys/tst117/tst117.vmx
app2.soho  |tst115   |/homen/acue/vmware/tst-ctys/tst115/tst115.vmx
app2.soho  |tst117   |omen/acue/vmware/tst-ctys/tst117.centos/tst117.vmx
app2.soho  |tst111   |acue/vmware/tst-ctys/tst111.OpenBSD-4.2/tst111.vmx
```

For getting some additional information on the actual installed distributions within the VMs the following call is used:

```
ctys-vhost \
   -o tab\_gen:3\_label\_9\%\%11\_Distro\_15\%\%12\_OS\_17\%\%7\_TCP\_18 \
   app2 vmw acue tst-ctys
```

The final result is:

```
label  |Distro        |OS          |TCP
\sout{---}+\sout{----------}+\sout{--------}+\sout{-----------}
tst117 |CentOS-5.0    |Linux-2.6   |192.168.1.240
tst115 |Solaris-10    |Solaris-10  |192.168.1.235
tst117 |CentOS-5.0    |Linux-2.6   |192.168.1.240
tst112 |CentOS-5.0    |Linux-2.6   |192.168.1.235
tst003 |SuSE-9.3      |Linux-2.6   |192.168.1.133
tst005 |Ubuntu-7.10-S |Linux-2.6   |192.168.1.135
tst103 |Fedora-8      |Linux-2.6   |192.168.1.223
tst106 |Debian-4.0r3  |Linux-2.6   |192.168.1.226
tst111 |OpenBSD-4.2   |OpenBSD-4.2 |192.168.1.234
tst120 |FreeBSD-6.1   |FreeBSD-6.1 |192.168.1.208
tst128 |NetBSD-4.0    |NetBSD-4.0  |192.168.1.212
tst002 |SuSE-9.3      |Linux-2.6   |192.168.1.132
tst111 |OpenBSD-4.2   |OpenBSD-4.2 |192.168.1.234
```

The decision is now to use tst117 as test machine.

9 Use MACROs for Custom Tables

The previous examples could be stored as MACROs and called just by their macro name. Several preconfigured macros arre available and could be listed with the utility **ctys-macros(1)** . Addtional Information on MACROs is available within the User-Manual.

10 SEE ALSO

ctys(1) , *ctys-createConfVM(1)* , *ctys-groups(1)* , *ctys-macros(1)* , *ctys-plugins(1)* , *ctys-vhost(1)* , *ctys-VMW(1)* , *vmware(1)*

11 AUTHOR

Arno-Can Uestuensoez <https://arnocan.wordpress.com/>
<https://unifiedsessionsmanager.sourceforge.io/>
<https://github.com/unifiedsessionsmanager>

12 COPYRIGHT

Copyright (C) 2008, 2009, 2010, 2011, 2020 Ingenieurbuero Arno-Can Uestuensoez
For BASE package following licenses apply,

- for software see GPL3 for license conditions,
- for documents see GFDL-1.3 with invariant sections for license conditions,

This document is part of the **DOC package**,

- for documents and contents from DOC package see
  'Creative-Common-Licence-3.0 - Attrib: Non-Commercial, Non-Deriv'
  with optional extensions for license conditions.

For additional information refer to enclosed Releasenotes and License files.