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

August 18, 2012

Contents

List of Figures

.
The XEN plugin is going to be reviewed now. The current description is still the first, even though still functional, some general enhancements are foreseen in order to harmonise the XEN plugin with the QEMU/KVM plugin and introduce a seamless integrated and conformant VirtualBox plugin. This particularly include a common installation and configuration interface.

1 CREATE a session

1.1 CREATE a session with CONSOLE:CLI

This call creates a new session by starting a DomU on the host lab00 and opening a CLI console access with the **-c** option within the caller's shell.

Due to pyGRUB an ANSI capable terminal seems to be required, thus starting it within EMACS **shell** will not work.

```
ctys  -t xen \
  -a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:cli \
  -z 2 -b 0,2 lab00'(-Z KSU)'
```

REMARK: Once a CLI console is attached from a caling shell, the focus might be released by shutdown of the attached VM only, or closing the window containing the session. In X11 environments a graphical console might be preferred for tests. Using CONSOLE:NONE for delayed attachement of a console is another option.

The local "-z 2" option forces a PTY to be created in any case by calling **ssh -t -t ...**. This avoids the remote **TERM=dumb** causing an error of pyGRUB. This is forced by default.

The local "-b 0,2" option forces a serial and non-background mode for interactive shells. Otherwise the console might not work. This is forced by default.

The Remote option "-Z KSU" raises permission by Kerberos on target machine, which is particularly required for the **'xm create ...'** call. This has to be set as required and may vary for sudo or native root-permissions. So, using defaults the required call is:

```
ctys -t xen \
  -a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:cli \
  lab00'(-Z KSU)'
```

The **CONSOLE:NONE** suboption just creates a server in so called headless-mode.

```
ctys -t xen -a create=l:tst100,CONSOLE:none lab00'(-Z KSU)'
```

The following calls just connect to a running instance. In this case the pathname is used.

```
ctys -t xen \
  -a create=p:\$HOME/xen/tst-ctys/tst100/tst100.conf,CONSOLE:cli,connect \
   lab00'(-Z KSU)'
```

As a variation a relative filename for comparison of "find" results could be used in variable length, as long as the match is unambiguous.

```
ctys -t xen \
   -a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:cli,connect \
   lab00'(-Z KSU)'
```

Same result with:

```
ctys -t xen \
   -a create=f:tst-ctys/tst100/tst100.conf,CONSOLE:cli,connect \
   lab00'(-Z KSU)'
```

Another call variation:

```
ctys -t xen \
   -a create=f:tst100/tst100.conf,CONSOLE:cli,connect \
   lab00'(-Z KSU)'
```

In this case the id, which is for Xen **id==pathname**, is used.

```
ctys -t xen \
   -a create=i:\$HOME/xen/tst-ctys/tst100/tst100.conf,CONSOLE:cli,connect \
   lab00'(-Z KSU)'
```

In the next case the label is used.

```
ctys -t xen -a create=l:tst100,CONSOLE:cli,connect lab00'(-Z KSU)'
```

The following call just connects to a running instance too, but uses the UUID.

```
ctys -t xen \
   -a create=u:6842caf91e3e43249ed596b8b9f2c5c2,CONSOLE:cli,connect \
   lab00'(-Z KSU)'
```

The same when using MAC address.

```
ctys -t xen \
   -a create=m:00:50:56:13:11:40,CONSOLE:cli,connect \
   lab00'(-Z KSU)'
```

The same when using IP address.

```
ctys -t xen \
    -a create=t:192.168.1.220,CONSOLE:cli,connect \
    lab00'(-Z KSU)'
```

1.2 CREATE a session with CONSOLE:XTERM

CONSOLE:XTERM This call creates a new session by starting a DomU on the host lab00 and opening a CLI console access with the "-c" option within a newly created xterm window.

```
ctys -t xen \
   -a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:xterm \
   lab00'(-Z KSU)'
```

The creation of PTY is not required. The Remote option "-Z KSU" raises permission by Kerberos, which is particularly required for the "xm create ..." call.

1.3 CREATE a session with CONSOLE:GTERM

Almost the same as XTERM, but a "gnome-terminal" is created instead.

```
ctys -t xen \
   -a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:gterm \
   lab00'(-Z KSU)'
```

1.4 CREATE a session with CONSOLE:EMACS

The EMACS console starts an EMACS and executes the call within a **shell** buffer named with LABEL of current XEN instance. The **ansi-term** is for now supported only when **ctys** is executed within as native call. The only drawback for the **shell** buffer is the lack of ansi-color support by ctys s and some restrictions due to lack of some ANSI terminal functions.

```
ctys -t xen \
   -a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:emacs \
   lab00'(-Z KSU)'
```

The following image shows an EMACS variant, where the ANSI-TERM mode is choosen and the window is slitted by the call automatically into two parts. The lower displays and prompt for the <exec-target>, whereas the top-window shows the prompt of a shell access into the GuestOS, which is given by the <machine-address> of the call.

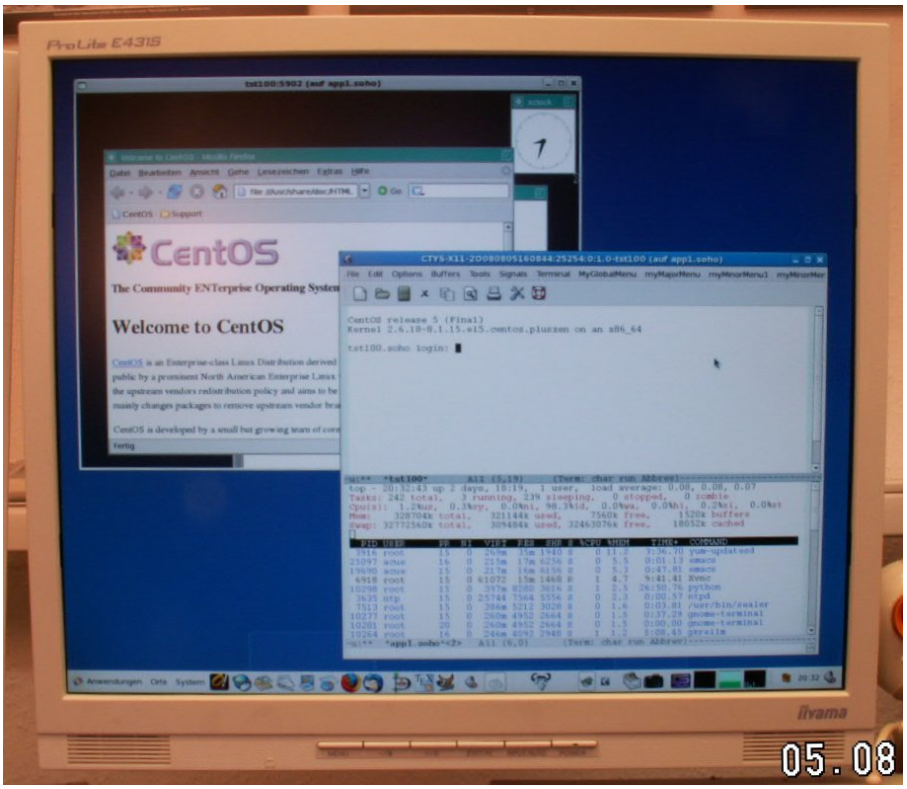

Figure 1: The "CONSOLE:EMACSAM" for a XEN Session

1.5 CREATE a session with CONSOLE:VNC

This call creates a session with an attached VNC viewer as a console. Therefore it is highly recommended to set the "vncunused=1" value in order to use a free port. When this is set to "vncunused=0" interference with native VNC servers might occur. The complete set of recommended VNC settings are:

```
vnc = 1
vncconsole = 1
vncunused = 1
```

The attachement of the console by a vncviewer is in ctys processed as a seperate step. Due to the asynchronous start of the DomU a timeout is implemented, which delays the start of the VNC console. This value coudl be configured by the user.
The resulting call for starting the session is:

```
ctys -t xen \
   -a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:vnc \
   lab00'(-Z KSU)'
```

The previous call implies the "-L DF" option for DISPLAYFORWARDING, the same call could be performed with for CONNECTIONFORWARDING.

```
ctys -t xen \
   -a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:vnc \
   -L CF \
   lab00'(-Z KSU)'
```

Now the advantage of a formal split for Client and Server, where the client is attached by a seperate step, should be clear. The result could be verified by calling

```
ctys -a list
```

on the client machine, which results e.g. to:

```
TCP-container |TCP-guest |label   |sesstype|c|user |group
\sout{----------}+\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
```

The previous output is the standard table displayed, but could be completely customized by the user.

The **sesstype** representing the session type **SSH(XEN)** displays the tunnel created by the internal DIGGER plugin and charachterizes it by **T** as a tunnel. The label is here the same as for the the VNC session, which is characterized by **C** as a client, attached to the sessions server, the Xen DomU tst100 on the remote machine lab00. The client(tst100) and server(tst100) are interconnected via the tunnel tst100. For additional customization, e.g. the SORT attribute refer to LIST action.

The following output shows both machines, the localhost as client and the lab00 as the server. The call is varied to

```
ctys -a list localhost lab00
```

and displays:

```
TCP-container |TCP-guest  |label   |sesstype|c|user |group
\sout{----------}+\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
lab00.soho    |-          |Domain-0|XEN     |S|-    |-
lab00.soho    |tst100     |tst100  |XEN     |S|-    |-
lab00.soho    |lab00.soho.|lab00   |PM      |S|-    |-
```

1.6 CREATE a session with CONSOLE:NONE

This call enters so called "headless-mode".

```
ctys -t xen \
   -a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:none \
   lab00'(-Z KSU)'
```

1.7 CREATE a session with RESUME from stat-file

```
ctys -t xen -a create=l:tst100,RESUME:mystate.stat lab01
```

1.8 CREATE a session with RESUME with VNC

```
ctys -t xen \
   -a create=l:tst100,RESUME,CONSOLE:VNC \
   lab01
```

1.9 CREATE a session with RESUME with EMACS

```
ctys -t xen -a create=l:tst100,RESUME,CONSOLE:EMACS lab01
```

1.10 CREATE a multiple sessions

The following call creates two sessions with one call. Both sessions are here located on the physical machine lab00 and use **ksu** for raise of access permissions.

```
ctys -t xen -- '(-Z KSU)' \
 lab00'( -a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:none )' \
 lab00'( -a create=f:xen/tst-ctys/tst101/tst101.conf,CONSOLE:none )'
```

The same could be varied for example to use different "-Z" options with **KSU** as default:

```
ctys -t xen -- '(-Z KSU)' \
 lab00'(-a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:none )' \
  lab00'(-a create=f:xen/tst-ctys/tst101/tst101.conf,CONSOLE:none -Z SUDO)'
```

The same could be varied for example to use different "-Z" options with none as default:

```
ctys -t xen -- \
 lab00'(-a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:none -Z KSU )'
 lab00'( -a create=f:xen/tst-ctys/tst101/tst101.conf,CONSOLE:none -Z SUDO )'
```

It might be obvious howto use different physical hosts:

```
ctys -t xen -- \
  lab00'( -a create=f:xen/tst-ctys/tst100/tst100.conf,CONSOLE:none -Z KSU )'
  lab01'( -a create=f:xen/tst-ctys/tst101/tst101.conf,CONSOLE:none -Z SUDO )'
```

2 CANCEL a session

2.1 CANCEL a session with POWEROFF

This call stops the DomU addressed by it's PATHNAME.

```
ctys -t xen \
   -a cancel=poweroff:0,p:/homen/acue/xen/tst-ctys/tst100/tst100.conf \
   lab00'(-Z KSU)'
```

For the following calls caching is used by default, which could lead to errors when ambiguity of addressed targets occur. When ambiguity occurs, additional <machine-address> parts might resolve this.
As a work around for handling multiple copies, such as backups, with identical address contents, one of the following approaches might help:

- The cache could be deactivated by using the options "-c off" and/or "-C off" .
- The usage of a PATHNAME might resolve ambiguity too, when resolved on the target only. Be aware, that this could be naturally ambiguous too in NFS environments with automount, of course. The latter will be frequently the case when configuring a load balancing environment by mounting VM collections.
- The cache could be rebuild by an appropriate selection in combination of a review of the contents of the filesystem.

The actual internal call of **ctys-vhost** is displayed within the trace output including the actual used parameters and could be called and varied after cut-and-paste to command line for validation purposes. A listing of all actual contained instances with ambiguous addresses is listed by the ctys-vhost option "-M all" . .
Same by using the LABEL as address.

```
ctys -t xen -a cancel=l:tst100,POWEROFF:0 lab01
```

When ambiguity occurs, e.g. like depicted by followin example:

```
ctys-vhost -o machine -s -M all lab00 XEN tst100

lab00.xyz;XEN;tst100;\
    /root/xen/tst100/tst100.conf;\
    6842caf91e3e43249ed596b8b9f2c5c2;\
    00:50:56:13:11:40;192.168.1.220;;;;CentOS;Linux;5;;PM

lab00.xyz;XEN;tst100;\
    /root/xen/tst100/tst100-inst.conf;;\
    00:50:56:13:11:40;192.168.1.220;;;;;;;;;

lab00.xyz;XEN;tst100;\
    /homen/chkusr/xen/tst-ctys/tst100/tst100.conf;\
    6842caf91e3e43249ed596b8b9f2c5c2;\
    00:50:56:13:11:40;192.168.1.220;;;;CentOS;Linux;5;\
    20080427002200;VM
```

The following call resolves ambiguity by deactivating cached operations:

```
ctys -t xen -a cancel=l:tst100,POWEROFF:0 -C off lab01
```

Similar with additional deactivation of nameservice caching, which anyhow is used sparsely for LIST action in current version.

```
ctys -t xen -a cancel=l:tst100,POWEROFF:0 -c off -C off lab01
```

2.2 CANCEL a session with RESET

Similar call to previous, but reboots after resetting hypervisor. When SELF is selected the hosting machine will be RESET too, else the GuestOS within the hypervisor only.

```
ctys -t xen -a cancel=l:tst100,RESET -c off -C off lab01
```

The following call ignores the eventual contained VMs within a stacked XEN instance. Actually the only VM supported to be executed nested within another is of type QEMU.

```
ctys -t xen -a cancel=l:tst100,FORCE,RESET -c off -C off lab01
```

2.3 CANCEL a session with REBOOT

Similar call to previous, but reboots after shutdown.

```
ctys -t xen -a cancel=l:tst100,REBOOT -c off -C off lab01
```

The following call ignores the eventual contained VMs within a stacked XEN instance. Actually the only VM supported to be executed nested within another is of type QEMU.

```
ctys -t xen -a cancel=l:tst100,FORCE,REBOOT -c off -C off lab01
```

Same with pathname, should be used for tests, due it's evaluation means for a missing label.

```
ctys -t xen \
   -a cancel=FORCE,REBOOT,p:/homen/chkusr/xen/tst-ctys/tst100/tst100.conf \
   -c off -C off lab01
```

2.4 CANCEL a session with PAUSE

Currently not yet available.

```
ctys -t xen -a cancel=l:tst100,PAUSE lab01
```

```
ctys -t xen -a cancel=l:tst100,FORCE,PAUSE lab01
```

2.5 CANCEL a session with SUSPEND

Currently not yet available.

```
ctys -t xen -a cancel=l:tst100,SUSPEND lab01
```

```
ctys -t xen -a cancel=l:tst100,FORCE,SUSPEND lab01
```

2.6 CANCEL a session with INIT

Calls UNIX **init** call with provided level. This call is somewhat limited for now, RESET and REBOOT should be preferred.

```
ctys -t xen -a cancel=l:tst100,INIT:0 lab00
```

```
ctys -t xen -a cancel=l:tst100,FORCE,INIT:6 lab00
```

3 LIST sessions

List sessions. The following call lists all sessions as MACHINE format raw records, a prefix title with given raw indexes is displayed. The provided indexes are the values to be used to define custom tables to be stored as macros.

```
ctys -t xen -a list=machine,titleidx lab00
```

A simple call with default values displays the standard output.

```
ctys -a list lab01
```

The following result is displayed.

```
TCP-container|TCP-guest  |label   |sesstype|c|user|group
\sout{---------}+\sout{-------}+\sout{----}+\sout{----}+-+\sout{--+---}
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|-   |-
```

When the subsystem XEN is selected the output is reduced to XEN only.

```
ctys -t xen -a list lab01
```

The following result is displayed.

```
TCP-container|TCP-guest  |label   |sesstype|c|user|group
\sout{---------}+\sout{-------}+\sout{----}+\sout{----}+-+\sout{--+---}
lab00.soho   |-          |Domain-0|XEN     |S|-   |-
lab00.soho   |tst100     |tst100  |XEN     |S|-   |-
lab00.soho   |tst101     |tst101  |XEN     |S|-   |-
```

A running configuration with two XEN sessions, where one session tst101 is connected by DISPLAYFORWARDING **DISPLAYFORWARDING** and a second tst100 is connected by CONNECTIONFORWARDING **CONNECTIONFORWARDING** is displayed with the call

```
ctys -t xen -a list localhost lab01
```

as follows.

```
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
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|-   |-
```

The next call lists all communication related informations by usage of the predefined custom table stored as macro 'TAB_CPORT'

```
ctys -a list=macro:TAB\_CPORT localhost lab00
```

results to:

```
Label     |stype   |cport|PM        |MAC              |TCP
\sout{------}+\sout{----}+\sout{-}+\sout{------}+\sout{-------------}+\sout{-----------}
tst100    |VNC     |     |ws2.soho  |                 |
ws2       |PM      |     |ws2.soho  |00:1D:60:A5:89:06|192.168.1.70
tst100    |SSH(XEN)|5950 |ws2.soho  |                 |
tst101    |VNC     |     |lab00.soho|                 |
LAB00     |VNC     |     |lab00.soho|                 |
LAB00     |VNC     |5901 |lab00.soho|                 |
Domain-0  |XEN     |     |lab00.soho|                 |
tst100    |XEN     |5900 |lab00.soho|00:50:56:13:11:40|
tst101    |XEN     |5902 |lab00.soho|00:50:56:13:11:41|
lab00     |PM      |     |lab00.soho|00:0E:0C:35:F8:48|192.168.1.71
```

Figure 2: TAB_CPORT by LIST

As could be recognized, the VMs tst100 and tst101 has no TCP values displayed, even though these are present. The reason is simply the decision to only display data which could be fetched easily unambiguously. The TCP address is only in a simple 1-to-1 relation, when no additional interfaces are present, and when the mapping information of the actual TCP stack and the ctys configuration including it's cacheDB are consistent. Additionally all services has to be setup properly, e.g. when using **host** or **dig**. Another point is that the VM has to be connected to the managing nameservices. Thus the complete automatic implementation is somewaht advanced and is shifted for now. In current version the user has to poll the missing information by additional tools, such as ctys-vhost, ctys-macmap, or ctys-dnsutil, or simply by **host** or **dig**.
Anyhow, the ENUMERATE action displays the TCP addresses as they are configured within the configuration file, refer for the output of the same common generic table "TAB_CPORT by ENUMERATE" as an complementary example. Additionally the same table could be used for **ctys-vhost** with a similar result to ENUMERATE: "TAB_CPORT by VHOST" .

4 ENUMERATE sessions

The following call enumerates all VMs

```
ctys -t xen -a enumerate=machine,title,b:xen localhost lab00
```

The complementary example for the common generic table "TAB_CPORT by LIST" could be generated by the call

```
ctys -a enumerate=macro:TAB\_CPORT,b:xen\%/etc/ctys.d localhost lab00
```

and displays some of the basic differences in the output strategy. As the following output depicts, here the fields for the TCP address are filled, whereas no cport is displayed.
The TCP addresses are ere displayed as statically configured within the configuration file. The cport is the communications port for the client processes, in this case the VNC port, which is dynamically allocated due to preconfigured **vncunused=1**. Thus the value is defined during runtime only, so it is not displayed by ENUMERATE, which displays the statically configured data.

```
Label |stype|cport|PM        |MAC              |TCP
\sout{--}+\sout{-}+\sout{-}+\sout{------}+\sout{-------------}+\sout{---------}
tst100|XEN  |     |ws2.soho  |00:50:56:13:11:40|192.168.1.220
tst101|XEN  |     |ws2.soho  |00:50:56:13:11:41|192.168.1.221
tst104|XEN  |     |ws2.soho  |00:50:56:13:11:44|192.168.1.224
ws2   |PM   |     |ws2.soho  |00:1D:60:A5:89:06|192.168.1.70
tst100|XEN  |     |lab00.soho|00:50:56:13:11:40|192.168.1.220
tst101|XEN  |     |lab00.soho|00:50:56:13:11:41|192.168.1.221
tst104|XEN  |     |lab00.soho|00:50:56:13:11:44|192.168.1.224
lab00 |PM   |     |lab00.soho|00:0E:0C:35:F8:48|192.168.1.71
```

Figure 3: TAB_CPORT by ENUMERATE

Additionally the same table could be used for **ctys-vhost** with a similar result to ENUMERATE: "TAB_CPORT by VHOST" .

5 SHOW

Lists the dynamic global envvironement data on the target.

```
ctys -t xen -a show lab00
```

Same result with

```
ctys -a show lab00
```

6 INFO

Lists static data for configured UnifiedSessionsManager with configuration relevant resource data.
The following call lists the initialized XEN plugin with implicitly loaded additional uninitialized plugins.

```
ctys -t xen -a info lab01
```

The following call lists all available plugings with their resulting init states on the target.

```
ctys -a info lab01
```

7 SEE ALSO

ctys(1) , *ctys-CLI(1)* , *ctys-configuration-XEN(7)* , *ctys-createConfVM(1)* , *ctys-plugins(1)* , *ctys-uc-XEN(7)* , *ctys-vhost(1)* , *virsch(18)* , *ctys-VNC(1)* , *ctys-X11(1)* , *xm(1)*

8 AUTHOR

Maintenance:	<acue_sf1@sourceforge.net>
Homepage:	<http://www.UnifiedSessionsManager.org>
Sourceforge.net:	<http://sourceforge.net/projects/ctys>
Berlios.de:	<http://ctys.berlios.de>
Commercial:	<http://www.i4p.com>

9 COPYRIGHT