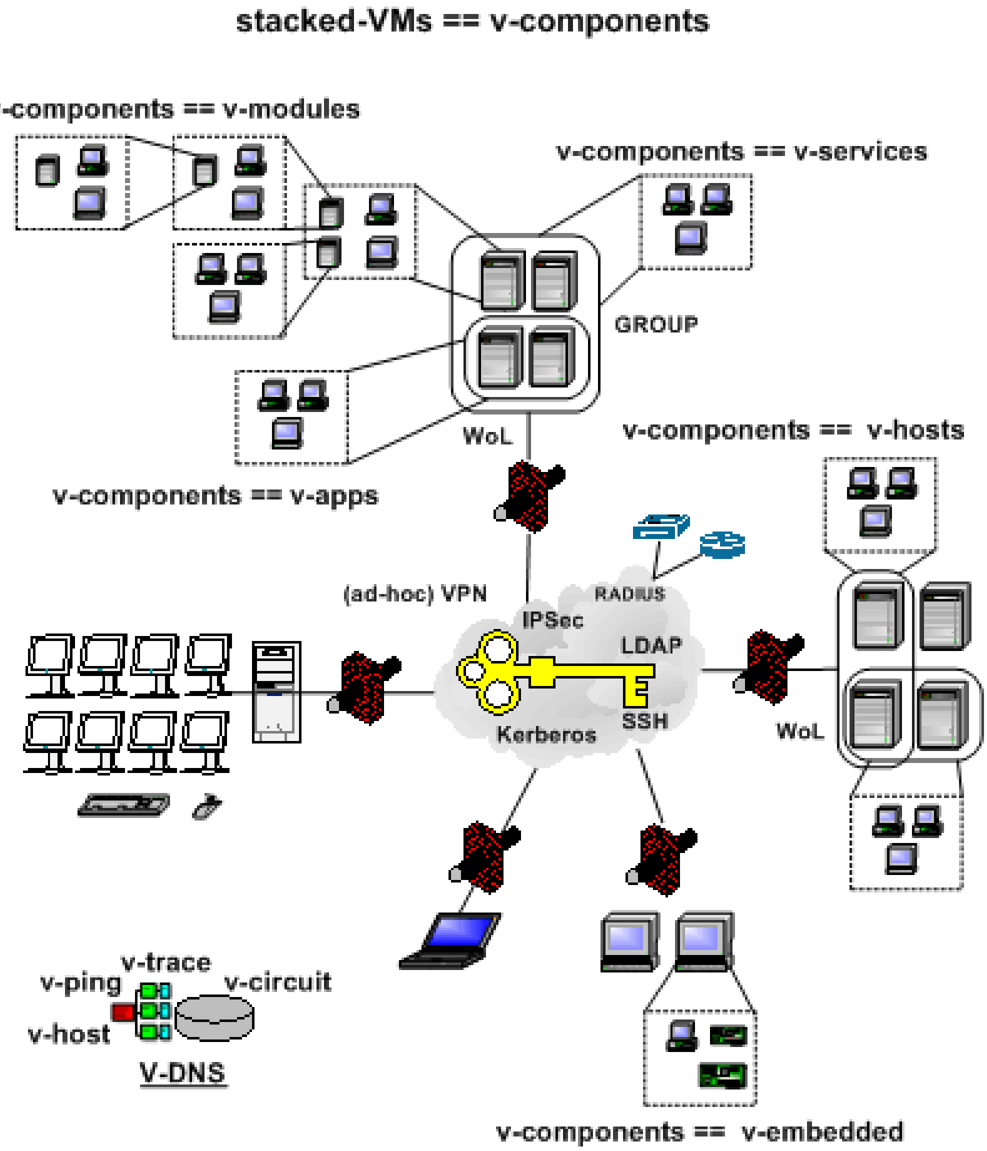


The UnifiedSessionsManager

CTYS-HOWTO

PDF-ONLINE-HELP

Version:01.12.002 - 2011.12.06

Release: International

Arno-Can Uestuensoez

Published by
Ingenieurbuero fuer Telekommunikations und Software-Systemloesungen
Arno-Can Uestuensoez

Zentnerstr. 34
D-80798 Muenchen
Germany

Phone:+49.89.27817287

https://arnocan.wordpress.com/
http://ctys.sourceforge.net/

The following text is required dur to formal reasons, if you are going to use these artifacts for your personal private purposes only you may probably not require to analyse it in detail. The application by scholars, students, and apprentices is specifically desired.

The reason of the introduction of this distinction is the experience of repetitive and ungoing unfair competition with 'criminal elements' contained, having a vast effect on the re-financiation of the further development for this project. Thus this step could be named 'self-defence'.

Software: GPL3

Basic-Documents: GFDL-1.3

Concepts+Interfaces+Documents: CCL - Creative Commons License - Non-Commercial, No-Derivs

Contents

List of Tables

List of Figures

Part I

Common Basics

Chapter 1

Preface

1.1 History

Version	Date	Author	Description
01.03.003.a01[146]	2008.02.11	Arno-Can Uestuensoez	Initial pre-release as embedded printable help
01.07.001.a01[147]	2008.08.03	Arno-Can Uestuensoez	First major update with numerous additions and partial review.
01.07.001.b02[148]	2008.08.11	Arno-Can Uestuensoez	Minor editorial updates. A lot of tests, some fixes.
01.07.001.b03[149]	2008.08.12	Arno-Can Uestuensoez	Minor editorial updates.
01.07.001.b04[150]	2008.08.16	Arno-Can Uestuensoez	Enhancement of documentation and Web-Site.
01.11.001[151]	2010.04.25	Arno-Can Uestuensoez	Major enhancements and updates.
01.11.002	2010.05.24	Arno-Can Uestuensoez	Documentation and web site enhancements.
01.11.003[151]	2010.05.31	Arno-Can Üstünsöz	Patch Default-Port VMware(TM)-Server-2.x, new tool ctys-beamer, add some documentation.
01.11.005[151]	2010.06.27	Arno-Can Üstünsöz	Alpha version of RDP plugin, bugfixes, added some documentation.
01.11.006[151]	2010.07.14	Arno-Can Üstünsöz	Alpha version of VBOX - VirtualBox(TM) plugin, bugfixes, added documentation, preparation of Typo3-Website.
01.11.008[151]	2010.07.30	Arno-Can Üstünsöz	Alpha-Version EnterpriseLinux, bugfixes, added documentation, First Gnome-Menues, ctys-scripts.
01.11.009[151]	2010.08.16	Arno-Can Üstünsöz	Alpha-Version gnome-starter, ctys-config, Fehlerbereinigungen, Ergänzung Dokumentation.
01.11.010[151]	2010.08.20	Arno-Can Üstünsöz	Verify GuetsOSs: ucLinux-QEMU(ARM+Coldfire), QNX-QEMU(x86), QNX-VBOX(x86),bugfixes, added documentation.
01.11.011[151]	2010.11.07	Arno-Can Üstünsöz	Verify New GuetsOSs: Android, MeeGo, RHEL, QNX. Version Updates: CentOS, Debian, OpenSUSE, OpenBSD, Ubuntu. Bugfixes, extension of documentation. menu generation.
01.11.014[151]	2010.11.22	Arno-Can Üstünsöz	Minor editorial.

1.2 Contact

Public maintenance:	acue@sf1_sourceforge.net
Administrative contact:	unifiedsessionsmanager@protonmail.com acue@UnifiedSessionsManager.eu
Commercial Services: The professional services are offered for end-customers only, so called 'body-leasers' are definetly not welcome.	Engineering Office Arno-Can Uestuensoez - www.i4p.com Ingenieurbuero Arno-Can Uestuensoez - www.i4p.com

1.3 Legal

All mentioned AMD products and their registered names are Trademarks of the Company Advanced Micro Devices, Inc.

All mentioned Google products and their registered names are Trademarks of the Google, Inc.

All mentioned Intel products and their registered names are Trademarks of the Company Intel, Inc.

All mentioned Microsoft products and their registered names are Trademarks of the Company Microsoft, Inc.

All mentioned Oracle products and their registered names are Trademarks of the Company Oracle, Inc.

QEMU is a trademark of Fabrice Bellard.

All mentioned RealVNC products and their registered names are Trademarks of the Company RealVNC Ltd.

All mentioned Red Hat products and their registered names are Trademarks of the Company Red Hat, Inc.

All mentioned Sun products and their registered names are Trademarks of the Sun Microsystems, Inc.

All mentioned SuSE products and their registered names are Trademarks of the Company Novell, Inc.

All mentioned VMware products and their registered names are Trademarks of the Company VMware, Inc.

Xen is a trademark of XenSource Inc.

If some is forgotten, it will be added immediately.

1.4 Acknowledgements

And, of course, I want to thank VMware for supporting their excellent VMware-Server and VMware-Player for free. The VMware-Workstation product initiated to my mind a major step of change and inspired a lot how software is commonly used and developed.

Many Thanks to Mr. Fabrice Bellard for his QEMU, which is the only and one test base for me to demonstrate a nested stack of VMs and it's integrated addressing including state propagation algorithms for now.

Great thank to the inventors of Xen at the university of Cambridge. UK, for their efficient VM.

And, of course, I would have probably no chance without "googling", so, even though it has to do something with business, many thanks for bringing the information of the whole world - and as soon as contacted of the remaining universe for sure - to my desktop. Hopefully I cope the current amount before the remaining universe comes into the scene.

I am meanwhile an enthusiastic user of CentOS/RHEL and OpenBSD, so I am glad having the opportunity to express my thank this way to all supporting persons an companies. Particularly RedHat Inc. for their actual open minded distribution policy and the CentOS team for their great work, and the OpenBSD team for their ongoing support for a base of real security.

And, last but not least, I want to thank very, very much to all the countless contributors for the numerous excellent Open- and Free-Software I use. Hopefully I can express my commitment and thanks with this piece of software, and my next following projects.

And finally I would like to express my thank to my friend Dirk and his wife Gisela, for their patience and enduring support. Their support at all enabled me reaching this milestone, despite of all the various and countless challenges and throwbacks to be managed.

Arno-Can Uestuensoez
Munich, Germany
March 2008

Chapter 2

Abstract

The "UnifiedSessionsManager" with it's main component "ctys" - "Commutate To Your Sessions" - is a unified and simplified shell-interface for intermixed operations and management of local and remote sessions on physical and virtual machines.

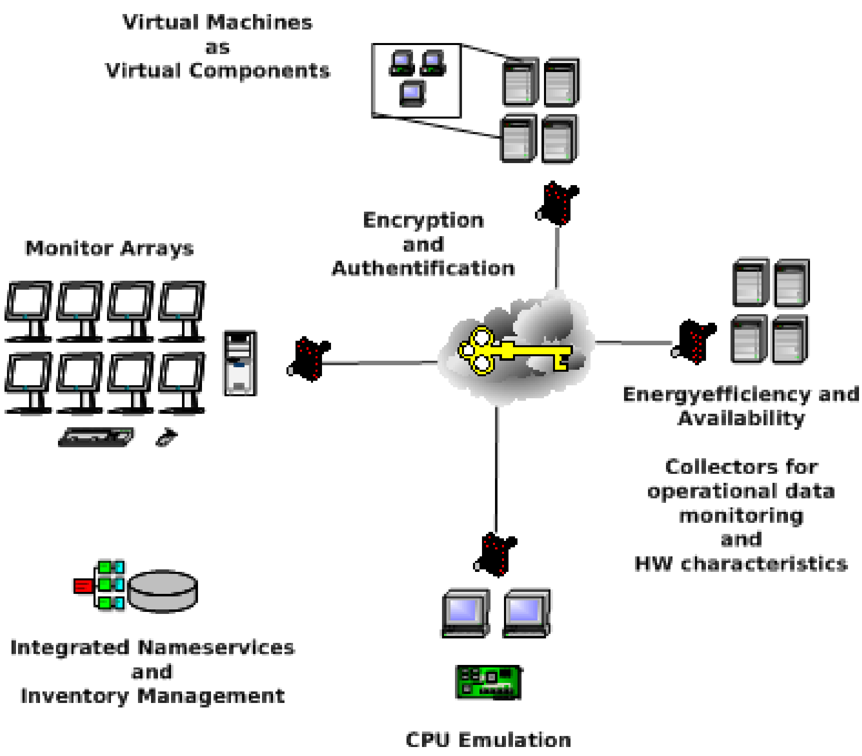


Figure 2.1: The UnifiedSessionsManager

The primary target was to combine facilities for the management of physical and virtual machines including the modelled sessions objects - alltogether combined with networking and security features - into a seamless interface.

- Management of Nested Multi-Level Stacks of Virtual Machines as Virtual Components
- Management of User-Interfaces on Monitor Arrays
- Support of Energy-Efficiency and enhanced Availability by transparent and dynamic Load-Management and integrated Wake-On-LAN
- Poll systems information for offered HW-Capabilities and Health-Monitoring
- Support of Integrated CPU Emulation for Cross-Development and Embedded Systems
- Support of Integrated Nameservices with Views and Hierarchical Groups

- Seamless access to all types of sessions by the definition of an Extended Address Schema
- Support of encryption by SSH and authentication/authorisation based on one or more of the common approaches by SSH, Kerberos and SUDO.

Usability

The emphasis is clearly on the integrated and simplified usability of actually much more complex interfaces. The main building block for this is the handling of the desktop presentation of the managed entities. This particularly comprises the handling of session windows on an X11 based desktop with logically combined screens by the so called "Xinerama" mode. But also some addressing facilities for disconnection and re-establishment of sessions to headless-running server entities.

- Support of seamless logical addressing for multiple screens.
- Sub-positioning by screen aliases as customized in standard "/etc/X11/xorg.conf".
- Handles multiple "Screen Layouts" independent from the actually loaded layout.
- Supports for multiple desktops with a desktop aware job-scheduler for "flicker-avoidance" of intermixed calls for display on multiple desktops.

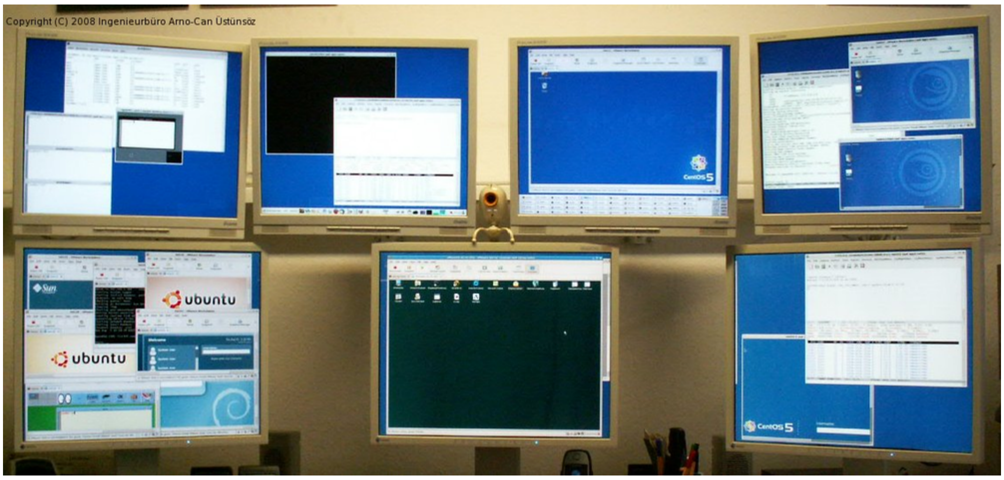


Figure 2.2: Physical Multi-Monitor Design

This screen layout contains(almost all entities are (TM)):

- 1x VMSTACK
- 4x CentOS, , 1x Fedora-8, 1x SuSE-9.3, 1x SuSE-10.2, 1x OpenSUSE-10.3, 1x 1x debian-4r3, Ubuntu-6.06.1, 1x Ubuntu-8.04, 1x OpenBSD-4.0, 1x OpenBSD-4.3, 1x Solaris-10, 1x MS-Windows2000
- 2x EMACSAM-Consoles
- 4x VNC-Consoles
- 5x X11-Consoles(here gnome-terminal)
- utilized by QEMU, XEN, and VMware
- Anyhow, this setup is far from the maximum frequently and easily utilized with the UnifiedSessionsManager.

Where all of them require different specific context-options due to presentation, security, and VM-Creation-Call requirements.

The benefit of the GROUPs and MACROs feature becomes quickly obvious, when the previous even simple example is shown by it's screenshot as resulting from the logical Xinerama-Screen.

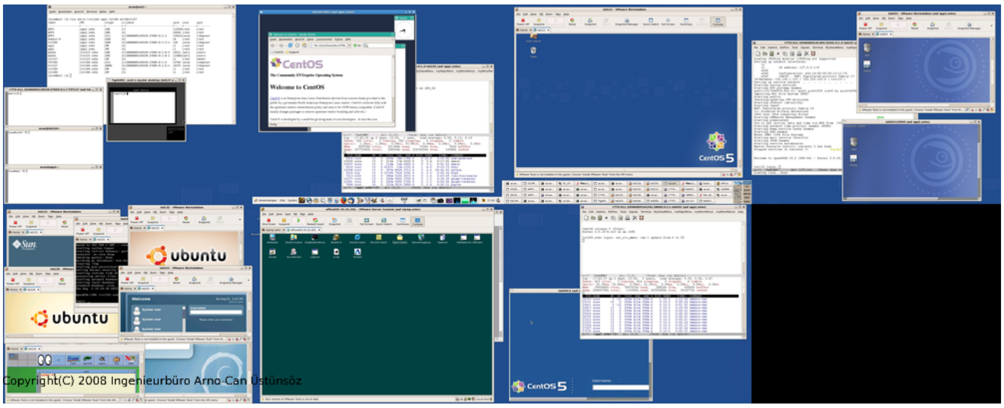


Figure 2.3: Logical Xinerama-Mode

The whole bunch of required calls could be pre-configured by MACROs and/or GROUPS with additional ordinary shell-facilities and for example could be reduced to the group "my-desktop". This includes the required boot of physical and virtual machines as well as the requested client for presentation and access on the local destop.

- ctys mydesktop

That's it.

The termination of the group could be prepared even more simple, when the STACK-Propagation feature of CANCEL is utilized, thus resulting in a call like:

- ctys -t PM -a cancel myhostlist

Which by default performs a native and recursive shutdown of the whole set of stacked machines executed top-down.

Current provided standard components are CLI , X11 , VNC , KVM (accelerator for QEMU), QEMU , VMW (VMware-Workstation/Server/Player), XEN , and PM (Linux, Solaris, OpenSolaris, OpenBSD, and FreeBSD). Any OS is supported, when control by hypervisor only is sufficient.

VirtualBox and OpenVZ are going to be intergrated next, as well as the specific upgrades for Server editions ov XEN and VMW.

Chapter 3

Feature Specification

3.1 Feature Introduction

The "UnifiedSessionsManager" comprises a number of first-time implemented features assembling to a solution for configuration and operation of environments with bulks of virtual and physical processing nodes.

Some to be mentioned are:

1. Management of User-Interfaces on Distributed Monitor-Arrays
2. Management of nested multi-level virtualizations
3. Support of Integrated CPU-Emulation for multiple architectures
4. Definition on an extended Address Schema
5. Support of Integrated Nameservices
6. Built-In support fo Encryption and Authentication
7. Introduction of GROUPS concept

3.2 Feature-Sum-Up

The following tables present an overview of the supproted components for current release. The listed PC, Workstation and Server based platforms with listed Hypervisors are supported and tested when marked with "OK". Additional platforms are going to be added for next versions("*").

The utility "ctys-genmconf" supports the detection and generation of relevant control data, the utility "ctys-plugins" verifies actual available operational states and resultingfeatures.

The main development and production platform for the UnifiedSessionsManager is CentOS.

The following pages show the current operational and test states of the various combinations of hypervisors, HostOS, and GuestOS. The actual operational states are visualized by specific colors as shown in next table.

Color	State
	The versions actually targeted to be supported with maximum available feature set.
	OK: Tested and operational in current release.
	NEXT: Sceduled for the next release. Probably already partly tested.
*	PLANNED: Intended for a later release.
-	OPEN: Technically possible, but for some reasons not yet planned to be implemented.

Table 3.1: Color coding of implementation and test states.

3.2.1 Supported Hypervisors

Supported Hypervisors on platforms as shown in the following tables.

Plugin	Supported Hypervisor	Versions		
		Previous	Current	InProcess
KVM	KVM		2.6.18/kvm-83 2.6.18-6/kvm-62 2.6.26-1/kvm-72	
OVZ	OpenVZ			
QEMU	Qemu	0.9.0	0.9.1,11.0.0,0.12.2	0.12.3
VBOX	VirtualBox(TM)	3.x	4.x	
VMW	VMware-Player(TM)	1.0.4	1.0.5,2.5.3,3.0.1	
VMW	VMware-Server (TM)	1.0.4,1.0.6,1.0.9	1.0.10,2.0.2	
VMW	VMware-Workstation(TM)	6.0.2,6.0.4,6.5.1	6.5.3,7.0.1	8.x
VMW	VMware-ESXi-Server(TM)			4.x.x
VMW	VMware-ESX-Server(TM)			4.1.0
XEN	Xen(TM)		3.0.3,3.1.0	3.3.0,3.3.1,3.4.2
XEN	Citrix-XenServer(TM)			5.5.0, 5.6.0

Table 3.2: Supported Hypervisors

3.2.2 Tested GuestOS support

The following table lists the already tested OS-Distribution vs. Containing Plugins. The containing plugins comprise the plugin itself as well as the required software and hypervisors.

Distribution	PMs		VMs						
	0	1+n	KVM	OVZ	VBOX	VMW	XEN	QEMU	
	x86	x86	x86	x86	x86	x86	x86	x86	ARM
					BSD				
DragonFlyBSD-2.10.1	X	*	X	-	X	X	-	X	-
FreeBSD-7	OK	*	OK	-	*	OK	-	OK	-
FreeBSD-8	OK	*	OK	-	X	OK	-	OK	-
NetBSD-5.0.1	*	*	*	-	OK	*	*	*	*
NetBSD-5.2	X	*	X	-	X	X	*	X	*
OpenBSD-4[1]	OK	OK	OK	-	OK	OK	-	OK	-
					Linux				
CentOS-5	OK	OK	OK	*	OK	OK	OK	OK	-
CentOS-6	OK	*	*	*	OK	*	*	*	
Debian-4-etch	OK	OK	-	-	-	OK	-	OK	-
Debian-5-lenny	OK	OK	OK	*	OK	OK	OK	OK	
Debian-6-squeeze	*	*	*	*	*	*	*	*	*
Oracle Linux 6	*	*	*	*	*	*	-	*	-
Oracle Linux 5	*	*	OK	*	OK	*	-	OK	-
Fedora-8	OK	OK	OK	-	-	OK	OK	OK	-
Fedora-10	OK	OK	OK	-	-	-	-	OK	-
Fedora-12	-	-	-	-	OK	-	-	-	-
Fedora-13	*	*	OK	-	OK	*	-	OK	-
Fedora-14	*	*	*	-	*	*	-	*	-
Fermilinux-5.5	*	*	*	-	*	*	*	*	-
Knoppix6.2	*	*	OK	-	*	*	*	OK	-
Knoppix-6.2.1 ADRIANE	OK	OK	OK	-	OK	*	*	OK	-
Mandriva-2010	OK	OK	OK	-	OK	OK	*	OK	-
Scientific Linux 5	OK	OK	OK	-	OK	OK	*	OK	-
Scientific Linux 6	X	X	X	-	X	X	*	X	-
openSUSE-10.3	OK	OK	-	-	-	OK	-	OK	-
openSUSE-11.1							OK		
openSUSE-11.2	OK	OK	OK	-	OK	OK	*	OK	-
openSUSE-11.3	OK	OK	OK	-	OK	*	*	OK	-
openSUSE-11.4	*	*	*	-	*	*	*	*	-
Puppet-Linux	*	*	*	-	*	*	*	*	*
RedHat-Enterprise Linux 5	OK	OK	OK	-	OK	X	X	OK	-
RedHat-Enterprise Linux 6beta	OK	OK	OK	-	OK	X	X	OK	-
Slackware-13.1	*	*	*	-	*	*	*	*	*

Table 3.3: Getestete GuestOS

Distribution	PMs		VMs						
	0	1+n	KVM	OVZ	VBOX	VMW	XEN	QEMU	
	x86	x86	x86	x86	x86	x86	x86	x86	ARM
SuSE-9.3	-	OK	-	-	-	OK	-	-	-
SuSE-10.2	-	OK	-	-	-	-	OK	-	-
Ubuntu-6.06.1-dapper	-	OK	-	-	-	OK	-	-	-
Ubuntu-7.10-gutsy	-	OK	-	-	-	OK	-	-	-
Ubuntu-8.04-hardy	OK	OK	OK	-	-	OK	OK	OK	-
Ubuntu-9.10	OK	OK	OK	*	OK	OK	*	OK	-
Ubuntu-10.10	OK	OK	OK	*	OK	OK	X	OK	-
Ubuntu-11.04	*	*	*	*	*	*	X	*	-
Solaris(TM)									
Solaris-10[2]	OK	OK	OK	-	OK	OK	-	OK	-
OpenSolaris-2009.6[4]	-	OK	OK	-	OK	OK	*	OK	-
ILLUMOS[5]	-	*	*	-	*	*	*	*	-
Nexenta[6]	-	*	*	-	*	*	*	*	-
OpenIndiana[7]	*	*	*	-	*	*	*	*	-
DOS									
FreeDOS[7]	-	-	*	-	*	*	*	OK	-
Balder[7]	-	-	*	-	*	*	*	OK	-
MS-Dos-5.x[7]	-	-	*	-	*	*	*	*	-
MS-Dos-6.x[7]	-	-	*	-	*	*	*	*	-
Windows									
MS-Windows-NT[7]	*	*	*	-	*	OK	*	*	-
MS-Windows-2000[7]	*	*	*	-	*	OK	*	*	-
MS-Windows-XP[7]	*	*	*	-	OK	OK	*	*	-
MS-Windows-2003[7]	*	*	*	-	*	OK	*	*	-
MS-Windows-7[7]	*	*	*	-	OK	*	*	*	-
MS-Windows-2008[7]	*	*	*	-	OK	*	*	*	-

Table 3.4: Tested GuestOS

Distribution	PMs		VMs						
	0	1+n	KVM	OVZ	VBOX	VMW	XEN	QEMU	
	x86	x86	x86	x86	x86	x86	x86	x86	ARM
Smartphone									
Android-2.2	*	*	OK	-	OK	-	-	OK	*
Windows-7-phone	*	*	*	-	*	*	-	*	*
Netbook									
Android-2.2	*	*	OK	-	OK	-	-	OK	*
MeeGo-1.0	*	*	(X)	-	OK	*	-	(X)	*
Windows-7-phone	*	*	*	-	*	*	-	*	*
Ubuntu-11.04									*
Tablet									
Android-2.2	*	*	OK	-	OK	-	-	OK	*
MeeGo-1.0	*	*	(X)	-	OK	*	-	(X)	*
Windows-7-phone	*	*	*	-	*	*	-	*	*
Ubuntu-11.04									*
Apple(TM) - OS									
Mac-OS-6.1.3	*	*	*	-	*	*	*	*	-

Table 3.5: Tested GuestOS

[5]No WoL for now.

[6]Some severe limitations may occur for Solaris, due the limitation of the "args" output of "ps" command to 80 characters. Thus the LIST action is faulty for some plugins, which means the instances are simply hidden due to argument-parts truncated by "ps". Some specific adaptations will follow. This depends on the argument ordering of the current command/wrapper and the actual contents beeing truncated. Supported Plugins: HOSTs and PM.

[7]Control by hypervisor only, no native support. Cygwin is foreseen for eventual future adaption. Tested with several versions, e.g. Windows-NT-Server, Windows-2000, and Windows-XP.

Distribution	PMs		VMs						
	0	1+n	KVM	OVZ	VBOX	VMW	XEN	QEMU	
	x86	x86	x86	x86	x86	x86	x86	x86	ARM
Embedded									
FreeRTOS	*	*	-	-	-	-	-	-	*
QNX	*	*	*	-	OK	-	-	(OK)	*
uCLinux	*	*	-	-	-	-	-	*	(OK)

Table 3.6: Tested GuestOS

3.2.3 Supported Native Plugins

The next table shows the passed tests of supported native plugins vs. OS-Distribution. The plugins including required hypervisors are to be executed on the listed OSs. Other OSs and versions might work as well.

Distribution	PMs	VMs						HOSTs			
	PM	KVM	OVZ	QEMU	VBOX	VMW	XEN	CLI	RDP	VNC	X11
BSD											
DragonFlyBSD-2.10.1	X		-	X	-	-	*	X	X	X	X
FreeBSD-7	OK		-	*	-	-	*	OK	*	OK	OK
FreeBSD-8	OK		-	X	-	-	X	OK	X	OK	OK
NetBSD-5.0.2	*		-	*	-	-	*	*	*	*	*
NetBSD-5.1	X		-	X	-	-	*	X	X	X	X
OpenBSD-4	OK	-	-	X	-	-	-	OK	X	OK	OK
Linux											
CentOS-5	OK	OK	*	OK	OK	OK	OK	OK	OK	OK	OK
CentOS-6	OK	*	*	*	OK	*	*	OK	OK	OK	OK
Debian-4-etch	OK	-	-	OK	-	OK	-	OK		OK	OK
Debian-5-lenny	OK	OK	*	OK	OK	OK	OK	OK	OK	OK	OK
Debian-6-squeeze	X	X	*	X	X	X	X	X	X	X	X
Enterprise Linux Server 5 / Unbreakable Linux	OK	OK	*	OK	X	-	OK	OK	(OK)	OK	OK
Fedora-8	OK	-	-	OK	-	-	OK	OK		OK	OK
Fedora-12	OK	*	-	*	*	*	*	OK		OK	OK
Fedora-13	*	*	-	*	*	*	*	OK	(OK)	OK	OK
Fedora-14	*	*	-	*	*	*	*	*	*	*	*
Fermilinux-5.5	*	*	-	*	*	*	*	*	*	*	*
Gentoo	*	-	-	-	-	-	-	*	*	*	*
Knoppix	OK	*	-	*	*	*	*	OK	(OK)	OK	(OK)
Mandriva-2010	OK	-	-	-	-	-	-	OK		OK	OK
openSUSE-10.3	OK	-	-	OK	-	OK	-	OK		OK	OK
openSUSE-11.1	OK	-	-	-	-	-	OK	OK		OK	OK
openSUSE-11.2	OK	OK	-	OK	*	*	*	OK		OK	OK
openSUSE-11.3	OK	OK	*	OK	OK	*	OK	OK	OK	OK	OK
OpenSUSE-11.4	*	*	-	*	*	*	*	*	*	*	*
RedHat-Enterprise Linux 5.5	OK	OK	*	OK	*	OK	OK	OK	OK	OK	OK
RedHat-Enterprise Linux 6.0 beta	OK	X	*	X	*	*	*	OK	OK	OK	OK
Scientific Linux 5	OK	OK	-	OK	*	OK	OK	OK	*	OK	OK
Scientific Linux 6	X	X	*	X	X	X	X	X	X	X	X
Sackware-13.1	*	*	-	*	*	*	*	*	*	*	*
SuSE-9.3	OK	-	-	-	-	OK	-	OK		OK	OK
SuSE-10.2	OK	-	-	-	-	-	-	OK		OK	OK

Table 3.7: Native Plugins vs. OS-Distribution

Distribution	PMs	VMs						HOSTs			
	PM	KVM	OVZ	QEMU	VBOX	VMW	XEN	CLI	RDP	VNC	X11
Ubuntu-6.06.1	-	-	-	-	-	-	-	OK		OK	OK
Ubuntu-7.10	-	-	-	-	-	-	-	OK		OK	OK
Ubuntu-8.04	OK	OK	-	(OK)[8]	-	-	-	OK		OK	OK
Ubuntu-9.10	OK	*	-	*	*	*	*	OK	*	OK	OK
Ubuntu-10.10	OK	OK	*	OK	X	X	X	OK	OK	OK	OK
Ubuntu-11.04	X	X	*	X	X	X	X	X	X	X	X
Hypervisor-Distributions											
ESXi	*	-	-	-	-	-	-	*		*	-
ESX-4.1.0	X	-	-	-	-	X	-	OK	X	OK	OK
XenServer-5.5.0[9]	X	-	-	-	-	-	X	OK	X	OK	OK
Solaris(TM)											
Solaris-10	(OK)	-	-	-	-	-	-	(OK)		(OK)	(OK)
OpenSolaris 2009.6	OK	-	-	X	X	-	*	OK	X	OK	OK
ILLUMOS	*	-	-	*	*	-	*	*	*	*	*
Nexenta	*	-	-	*	*	-	*	*	*	*	*
OpenIndiana	*	-	-	*	*	-	*	*	*	*	*
MS-Windows(TM)/Cygwin											
MS-WNT4[10]	X	-	-	X	X	X	-	X	X	X	X
MS-W2K[11]	X	-	-	X	X	X	-	OK	OK	OK	OK
MS-WXP[12]	X	-	-	X	X	X	-	OK	OK	OK	OK
MS-W2K3[13]	X	-	-	X	X	X	-	OK	X	X	OK
MS-W2K8R2[14]	X	-	-	X	X	X	-	OK	OK	OK	OK
MS-W7[15]	X	-	-	X	X	X	-	X	X	X	X
Apple(TM) - OS											
Mac-OSX-10.6.3	*	-	-	*	*	-	-	*	*	*	*
HP(TM) - OS											
HP-UX	*	-	-	*	-	-	-	*	*	*	*
IBM(TM) - OS											
AIX	*	-	-	*	-	-	-	*	*	*	*
Smartphone											
Android-2.2	*	-	-	-	-	-	-	*	*	*	*
MeeGo-1.0	OK	*	-	*	-	-	-	OK	*	*	*
Windows-Phone	*	-	-	-	-	-	-	*	*	*	*

Table 3.8: Native Plugins vs. OS-Distribution

[15]Compilation of 'qemu-system-x86_64' with support for '-name' option required.
[16]Requires Cygwin support and a .

Distribution	PMs	VMs						HOSTs			
	PM	KVM	OVZ	QEMU	VBOX	VMW	XEN	CLI	RDP	VNC	X11
Embedded											
FreeRTOS	*	-	-	-	-	-	-	*	*	*	*
QNX	*	-	-	-	-	-	-	*	*	*	*
RTEMS-Dev	-	-	-	*	*	-	-	*	*	*	*
uCLinux	*	-	-	-	-	-	-	*	*	*	*

Table 3.9: Native Plugins vs. OS-Distribution

[17]Compilation of 'qemu-system-x86_64' with support for '-name' option required.
[18]Requires Cygwin support and a .

Unterstützte Produkte und Versionen für die jeweiligen Plugins. Diese varirieren z.T. für die verschiedenen Plattformen.

Plugin / Toolset	Unterstütztes Produkt	Versionen		
		Vorversion	Aktuell	InBearbeitung
CLI	bash		3.2.39.1, >3.x	
	Cygwin(alpha)		1.7.9-1(2.738)	
RDP	rdesktop		1.6	
	krdc			3.5.10
	tsclient			0.150
	mstsc.exe			x
	vinagre			0.51
	VirtualBox		3.x	4.x
	MS-Windows(TM)		W4NT, W2K, WXP, W2K3, W2K8, W7	
	Cygwin(alpha)		1.7.9-1(2.738)	
VNC	RealVNC		3.3.7, 4.1.1, 4.1.3	
			4.1.1, 4.1.2	
	TigerVNC		1.0.90	
	TightVNC		1.2.9, 1.3.10, 2.0.2	
	MetaVNC			0.6.5
	QEMU/KVM		>0.9.x	
	UltraVNC			1.0.8.2
	VMware	W:6.x	W:7.x	
	XEN		>3.x	
	Cygwin(alpha)		1.7.9-1(2.738)	

Table 3.10: Durch HOSTs Plugins Unterstützte Clients und Desktops

Plugin / Toolset	Unterstütztes Produkt	Versionen		
		Vorversion	Aktuell	InBearbeitung
X11	gnome-terminal		2.22.3	
	xterm		235	
	emacs	>21.x	>22.x, 22.2.1	
	Cygwin(alpha)		1.7.9-1(2.738)	

Table 3.11: Durch HOSTs Plugins Unterstützte Clients und Desktops

Plugin / Toolset	Unterstütztes Produkt	Versionen		
		Vorversion	Aktuell	InBearbeitung
Desktop	Gnome		2.20.7	
	KDE		5.48	
	fvwm		2.5.26	
	xfce		x.x	
Shells	bash		3.2.39.1, >3.x	
	Cygwin(alpha)		1.7.9-1(2.738)	

Table 3.12: Unterstützte HOSTs-Plugin Sub-Komponenten

Plugin / Toolset	Unterstütztes Produkt	Versionen		
		Vorversion	Aktuell	InBearbeitung
QEMU	Qemu	0.9.0	0.9.1,0.11.0,0.12.2	0.12.3
	KQEMU			
	KVM			
VBOX	VirtualBox(TM)		3.1.2	3.2.8, 3.2.10, 4.x
VMW	VMware-Player(TM)	1.0.4	1.0.5,2.5.3,3.0.1	
	VMware-Server (TM)	1.0.4,1.0.6,1.0.9	1.0.10,2.0.2	
	VMware-Workstation(TM)	6.0.2,6.0.4,6.5.1	6.5.3,7.0.1	
XEN	Xen(TM)		3.0.3,3.1.0	3.3.0,3.3.1,3.4.2,4.0.0

Table 3.13: Unterstützte Server basierte VMs plugins

Plugin / Toolset	Unterstütztes Produkt	Versionen		
		Vorversion	Aktuell	InBearbeitung
VMW	VMware-ESX-Server(TM)		4.1.0	
	VMware-ESXi-Server(TM)			4.0.0
XEN	Citrix-XenServer(TM)		5.5.0	5.6.0

Table 3.14: Unterstützte Host basierte VMs plugins

3.2.4 Tested Client OSs

The following table lists the already tested client OSs.

Distribution	ctys			GUI						
	GROUP	DF	CF	X11		WM				
				X11	Xinerama	Gnome	KDE	fvwm	xfce	ffs.
BSD										
DragonFlyBSD-2.10.1	X	X	X	X	*	X	X	X	-	
FreeBSD-7										
FreeBSD-8	X	X	X	X	*	X	X	X	-	
NetBSD-5.2	X	X	X	X	*	X	X	X	-	
OpenBSD-4	X	X	X	X	*	X	X	X	-	
Linux										
CentOS-5	OK	OK	OK	OK	OK	OK	X	X	X	
CentOS-6	X	X	X	X	X	X	X	X	X	
Debian-5-lenny	OK	OK	OK	OK	OK	OK	X	X	X	
Debian-6-squeeze	X	X	X	X	X	X	X	X	X	
Enterprise-Linux Server	*	*	*	*	*	*	*	*	*	
Fedora-8										
Fedora-10	X	X	X	X	*	X	X	X	X	
Fedora-12	*	OK	*	OK	*	OK	*	-	*	
Fedora-13	*	*	OK	OK	*	OK	*	-	*	
Knoppix	X	X	OK	OK	*	OK	X	X	X	
Mandriva-2010	*	OK	*	OK	*	OK	*	*	*	
Scientific Linux	OK	OK	OK	OK	*	OK	OK	-	-	
openSUSE-11.2	OK	OK	OK	OK	*	OK	OK	OK	OK	
openSUSE-11.3	*	*	*	*	*	*	*	*	*	
RedHat-Linux Server 5.5	*	OK	*	*	*	OK	*	*	*	
RedHat-Linux Server 6.0 beta	*	*	*	*	*	*	*	*	*	
Ubuntu-6.06.1-dapper										
Ubuntu-7.10-gutsy										
Ubuntu-8.04-hardy	OK	OK	(OK)	OK		OK	OK	OK	OK	
Ubuntu-9.10	X	X	X	X	X	X	X	X	X	
Ubuntu-10.10	X	X	X	X	X	X	X	X	X	
Hypervisor-Distributions										
ESXi										
ESX	X	X	X	X	*	X	X	X	X	
XenServer-5.5.0	X	OK	X	OK	*	OK	X	X	OK	

Table 3.15: Getestete ClientOS

Distribution	ctys			GUI						
	GROUP	DF	CF	X11		WM				
				X11	Xinerama	Gnome	KDE	fvwm	xfce	ffs.
MS-Windows(TM)										
Windows7(TM)										

Table 3.16: Getestete ClientOS

Distribution	ctys			GUI						
	GROUP	DF	CF	X11		WM				
				X11	Xinerama	Gnome	KDE	fvwm	xfce	ffs.
Apple(TM)-OS										
Mac-OS	*	*	*	*	*					
Solaris(TM)										
Solaris-10	*	*	*	*	*	*	*	*	*	
OpenSolaris-2009.6	X	X	X	X	*	X	X	X	X	
ILLUMOS	*	*	*	*	*	*	*	*	*	
Nexenta	*	*	*	*	*	*	*	*	*	
OpenIndiana	*	*	*	*	*	*	*	*	*	
Windows										
MS-Windows-NT										
MS-Windows-2000	*	*	*	*	*	*	*	*	*	
MS-Windows-XP	*	*	*	*	*	*	*	*	*	
MS-Windows-200x	*	*	*	*	*	*	*	*	*	
Smartphones										
Android	*	*	*							*
MeeGo	*	*	*							*
Embedded										
QNX	*	*	*	-	-	-	-	-	-	*
uCLinux	*	*	*	-	-	-	-	-	-	*
FreeRTOS	*	*	*	-	-	-	-	-	-	*

Table 3.17: Getestete ClientOS

[17] Kein WoL.
[18] Einige Einschränkungen bei LIST.
[19] Unter ausschließlicher Kontrolle des hypervisors. Getested mit diversen Versionen, z.B. Windows-NT-Server, Windows-2000, und Windows-XP.

Part II

Help and Setup

3.3 ctys-help-on

SYNTAX

<ctys-command>

```
-H <help-option>

   <help-option>:=
         (man|html|pdf)][=((1-9)|<help-on-item>[,<help-on-item-list>])
       | (path|list|listall)
       | funcList=<any-function>][@<module-name>[@...]][,<any-function>...]
       | funcListMod=<any-function>][@<module-name>[@...]][,<any-function>...]
       | funcHead=<any-function>][@<module-name>[@...]][,<any-function>...]
       | _ONLINEHELP_
       | _HELP_
     )
```

DESCRIPTION

The **-H** option is the common generic option of all tools for the display of online help.

The default is the display of man pages within a commandline terminal. This could be any valid document within the search list defined by the variable MANPATH. The output format could be optionally specified as PDF and HTML documents.

This tool is also used within menu entries of the XDG desktop of Freedesktop.org for graphical display of online help. Therefore the current version provides the simple HTML lists **doc.html** for the **DOC-Package**, and the **base.html** file for the **BASE-Package**.

REMARK:
For the **-H** option of the call 'ctys -H man' and 'ctys-vhost -H man' the man parameter is mandatory. In all other cases the call '<any-other-ctys> -H' searches for 'man' output by default within MANPATH.
. **OPTIONS**

The following suboptions and parameters could be applied:

-H path
Displays current document and man path.

-H list
Lists available online documents and manpages.

-H listall
Lists available online documents and manpages including the documents available by MANPATH.

-H (man|html|pdf)[=([1-9]||[<help-on-item-list>])
Displays the requested information with one of the formats **man**, **pdf**, or **html**. The following viewers are preconfigured as shell variables within the configuration files and can be adapted as required:

```
CTYS_MANVIEWER=man
CTYS_PDFVIEWER=(acroread else kpdf else gpdf)
CTYS_HTMLVIEWER=(konqueror else firefox)
```

Default is **manpage** for the current process with **man 1 ...**. Additional constraints could be applied such as another man-section or a filename, which could be either literally matching or a string to be expanded. In case of expansion the first match is taken.

-H funcList[=[<any-function>][@<module-name>[@...]]]
List of function names, sorted by function names. In addition the file names and line numbers are displayed too.

-H funcListMod[=[<any-function>][@<module-name>[@...]]]
List of function names, sorted by file names. In addition the file names and line numbers are displayed too.

-H funcHead[=[<any-function>][@<module-name>[@...]]]
Displays the contents of function headers, sorted by file names. The following constraints could be applied:

- <any-function>: If given <any-function> than only this is displayed.
- <module-name>: If given <module-name>, than the functions contained within this module only are displayed.

-H (_ONLINEHELP_|_HELP_)
Displays the predefined online help for the installed package.

. EXAMPLES

<ctys-command> -H (_ONLINEHELP_|_HELP_)
Displays the predefined online help for the installed package.

<ctys-command> -H html=base
Displays a summary of links for all documents contained in the **BASE package**.

<ctys-command> -H html=doc
Displays the extended online help as contained in the **DOC package**.

<ctys-command> -H list
Lists available online documents and manpages.

<ctys-command> -H ctys
Displays the **manpage** for ctys with **man**.

<ctys-command> -H man=ctys
Displays the **manpage** for ctys with **man**.

<ctys-command> -H html=ctys
Displays the **manpage** for ctys with CTYS_HTMLVIEWER, by default **firefox** or **konqueror**.

<ctys-command> -H pdf=ctys
Displays the **manpage** for ctys with CTYS_PDFVIEWER, by default **kpdf**, **gpdf**, or **acroread**.

<ctys-command> -H pdf=howto
Displays the **ctys-howto-online.pdf**, which is displayed in alphabetical order before **ctys-howto-print.pdf**.

<**ctys-command**> **-H pdf=howto-print**
Displays the **ctys-howto-print.pdf**, which is the first appropriate match.

<**ctys-command**> **-H pdf=command-ref**
Displays the **ctys-command-reference.pdf**.

<**ctys-command**> **-H html=CLI,X11,VNC,VMW**
Displays the **manpage** for ctys-CLI, ctys-X11, ctys-VNC and ctys-VNM with CTYS_HTMLVIEWER, by default **firefox** or **konqueror**. For incomplete names a search with **find** is utilized for name expansion.

<**ctys-command**> **-H html=ctys-extractARPlst,extractMAClst**
Displays the **manpage** for ctys-extractARPlst and ctys-extractARPlst with CTYS_HTMLVIEWER, by default **firefox** or **konqueror**. For incomplete names a search with **find** is utilized for name expansion.

Chapter 4

ctys Setup

4.1 Installation

4.1.1 Basic Install

The current version of the UnifiedSessionsManager supports two options for installation, **rpm** based installation by standard mechanisms for example on CentOS and file-copy based installation by the installer **ctys-distribute** for network centric installation. This utility supports local installation as well as **scp** based installation by OpenSSH for secure distribution. In addition to physical installation the setting of symbolic links to a common pool is supported.

The following steps describe one possible variant of the file-copy based installation.

1. Download
   The package is provided as a simple gzipped-tar file[153, UnifiedSessionsManager]. It has to be downloaded and unpacked. and unpacked to local filesystem.

2. ctys-distribute
   The install script in the binary directory of the unpacked package

   ctys.<version>/bin/ctys-distribute

   has to be called. Several arguments could be applied, a list is displayed by "-h" option.

   The commonly appropriate variant is:

   ```
   ctys.<version>/bin/ctys-distribute -F 1 -P UserHomeCopy localhost
   ```

   When the rpm package is installed the call could be either the same or:

   ```
   ctys.<version>/bin/ctys-distribute -F 1 -P UserHomeLinkonly localhost
   ```

   The alternative is to set the PATH variable appropriate.

   The advantage of installation is the creation of local copies for configuration files in any case, what enables modification as required.

   An already present set of user specific configuration files is not removed by default, but could be forced to be replaced by '-F 2' option.

3. Check of required system-call permissions
   A number of required system-calls needs specific root call permissions, which has to be provided by one of the following means:

   - Execution as user "root"
     When using ctys as user root any required permission should be available.
   - ksu
     "ksu" is the sudo like call of kerberos [125, MIT-Kerberos][126, Heimdal] , therefore ".k5users" and/or ".k5login" in the home directory of the permission-offering user(root) has to be stored.
   - sudo
     Sudo[127, sudo] could be used aternatively or in combination with "ksu". Sudo provides a more finegrained access administration, but is not(yet?) integrated into kerberos authentication. The file "/etc/sudoers" has to be edited appropriately by visudo.

     The potential pitfall when using SUDO is the requirement of a PTY by default, which could be configured within sudoers, but should be kept due to additional constraints by usage of OpenSSH. The options "-Z" and "-z" handle this issue.

   Additionally the required tools such as bridge-utils has to be installed, as checked by the following calls.

   The ctys-plugins tool supports a means to check (almost) any internal system call for it's presence and available call permissions. Some limit occurs on "inherent destructive" calls, which have no option to be used for check purposes only. Typicall for this are the shutdown-commands, which on some platforms could not be analysed for access permission without actually shutting down the machine.

   The following calls could be used to validate the functionality of current installation:

   - Check of Client Functions
     This checks the client set of required function calls.

     ctys-plugins -d 64,p -T all
   - Check of Server Functions
     This checks the server set of required function calls

     ctys-plugins -d 64,p -T all -E

   The previous calls utilize the internall common wrapper function

   "checkedSetSUaccess"

   by setting a specific debugging flag with the value "64=D_SYS". This activates the trace of system calls only. The options are described within the User-Manual, for addtional technical help on this library function call the online help for call interfaces could be used:

   "ctys -H funchead=checkedSUaccess"

   Any missing but required component as listed by the previous call should be installed and the required access granted by an appropriate entry for sudo/ksu.

4. Hypervisors - Some specifics
   The hypervisors QEMU/KVM, VMW, and XEN has to be prepared as described in teh Release-Notes and teh configuration use-cases.

5. ctys configuration
   The next step should be the adaptation of the provided default configuration to local machine. Therefore the config file in the home directory of the user and/or the installed default files has to be edited. The provided configuration files contain the required description and additional examples.

6. Configure PMs
   The involved physical machines - PMs - should be configured by calling the tool ctys-genmconf in order to generate the PM configuration data. The result is stored by default within the directory "/etc/ctys.d".

   This step requires to be repeated when all hypervisors are installed and marked as operable by the check with ctys-plugins utility. This call updates the stored registration of capabilities available on the local machine, including the later required STACKCAP variable for verification of possible stacking capabilities, when a VM-Stack is going to be started. The update has to be performed before the creation of a cacheDB with the tool ctys-vdbgen .

7. Generate DHCP and/or MAC cache
   The access performance for plugins with stored configuration data - PMs and VMs - will be dramatically enhanced when generating a prefetched cache database.

   The first step required for the creation of the database is the creation of the mapping table generated by ctys-extractMAClst and/or ctys-extractARPlst , either from a valid "dhcpd.conf" of from simply "ping-ing" a list of hosts.

   The result consists of the data required for mapping of MAC addresses to IP/DNS addresses as provided by ctys-macmap . The format of stored data is documented and could be edited manually. It is an extended "/etc/ethers" format, the "/etc/ethers" database could be generated by the previous tools.

8. Generate cache of configurations for VMs and PMs
   The access to stored static configuration is performed by a two step approach controlled by the option "-c" . The first attempt is to read from the generated cache, the second is to scan the filesystem on the target entity in accordance to the provided call options.

   The prefetch of the configuration data in a local file database enhances the call performance dramatically, a factor of at least 10, but almost in any case of 100 and much more is common.

   The build of the cache database is handeled by the tool ctys-vdbgen for collecting the data and by the tool ctys-vhost for preprocessing required correlations and intermediate preprocessings.

9. ctys
   Once this point is reached, ctys should be operational as required. The first tests should be executed by simple dynamic plugins such as CLI, X11, and VNC. Call examples are given in Section II 'Help and Setup' on page 41 .

   In case of errors the option "-d" provides a scalable degree of debugging information.

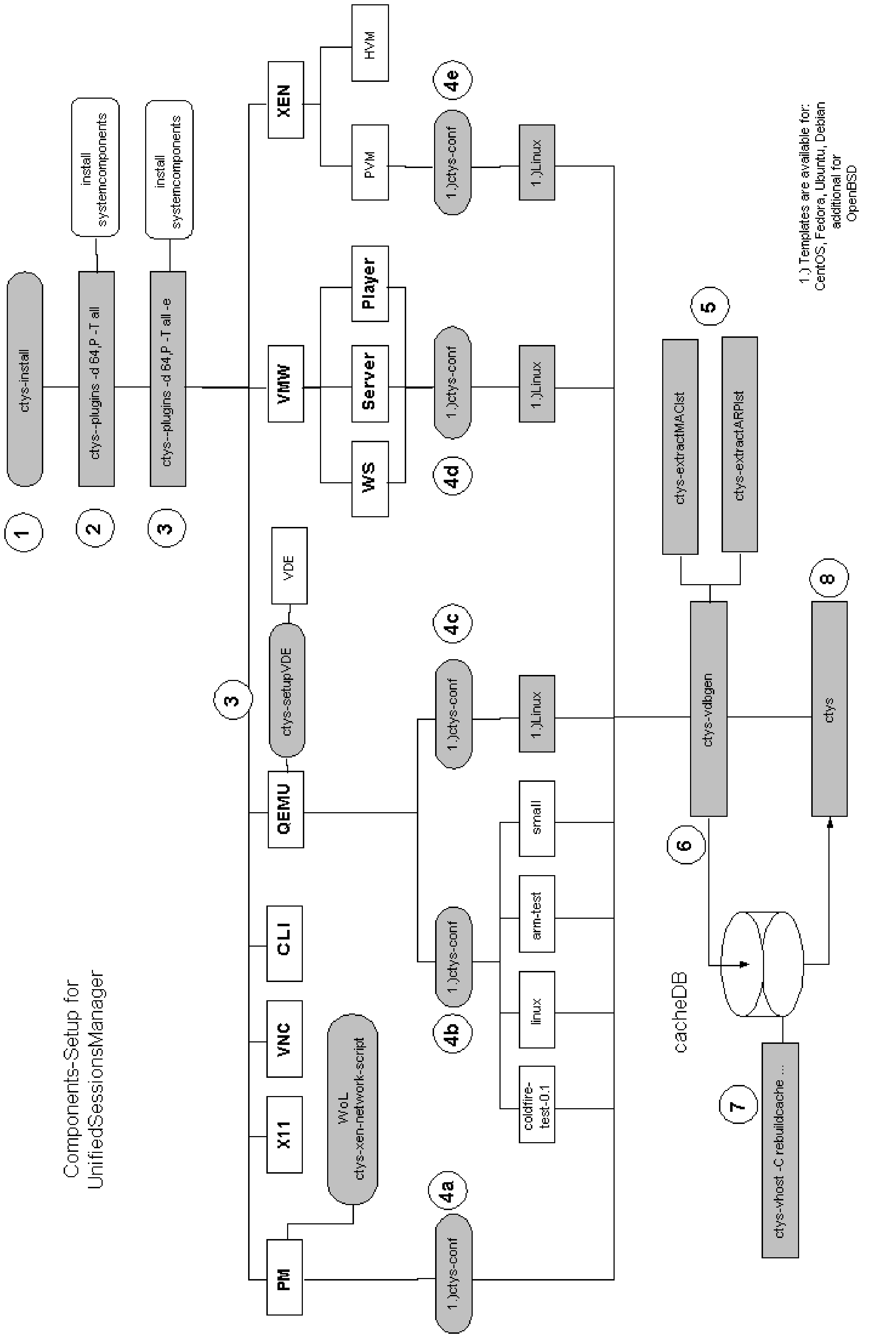


Figure 4.1: Install Steps

4.1.2 Security Environment

Some basic hints are given here only for the wide field of security and general access issues, additional support is available as commercial services only([?], [154], [?]).

General Remarks

Network Accounts
: When using UnifiedSessionsManager a network account with SSO is absolutly recommended. There are two exceptions, where the account should be a local account:

- The root account sould be a local-only account, of course.
- In case of required WoL for a machine running a virtual bridge, such as a Xen based machine, a local account is required. A "simple" shutdown may work without any difficulty. This is due to a currently required workaround particularly in case of Xen for setting of WoL on the physical NIC for offline operations of the NIC itself.

NFS
: Using NFS has some security risks, even though it is particularly a real benefit for development issues. Thus NFS versions <4 should not be used when crossing insecure network segments without additional provisions.

root Access
: Should be configured local only.

Router Configuration
: Router have to be configured for WoL, when the target NIC is outside the local segment. This is beyond the scope of this document.

Quick Setup for ssh

The only and one facility supported for communications between any type of entities is the usage of OpenSSH. Particularly the Handling of DISPLAY is deeply embedded into the UnifiedSessionsManager by usage of OpenSSH. Therefore the most benefit results from setting up SSH for SSO. The recommended setup is the usage of SSH in combination with Kerberos.

Quick Setup for ksu

The preferred authentication is the usage of Kerberos, which offers the access configuration facilities by ".k5login" and ".k5users". The usage is quite straight-forward, even though for some aspects not as flexible as "sudoers" is. Which is compensated by it's advance for networking purposes.

The only hard-wired restriction for usage of ksu applies, when network users are configured for CANCEL action on bridged NICs. When the WoL feature has to be applied, the NIC needs to be disconnected during the CANCEL procedure with continued access to restricted system resources. Thus this requires "sudoers" and a local user .

Quick Setup for sudo

Using sudo for authentication offers a perfect and straight forward setup for local users. For environments with NIS/NIS+ or any "distribution" utility a networked configuration update is available too. But anyhow, the initial access to a machine, and the "relay-user" is out of the scope of "sudoers".

The network access, particularly the SSO, is crucial for the applicability of ctys in a lager environment, thus sudo has to be used in companion with any network accounting facility only.

Quick Setup for LDAP

LADP is recommended for any distributed directory system. The user information should be handeled by LDAP, which has it's particular advantage when using a SMB based OS.

The setup in a heterogeneous environment is in details somewhat tricky, and requires some more description, though outside the scope of this document.

Quick Setup for autofs

Autofs is particularly the choice for a network login, when data has to be available within a secure segment only this could be simply based on NFS(version<4). The setup could particularly be based on LDAP in combination with Kerberos and SSH, with a lean centralized configuration. The description as provided by the project should suffice for the first steps.

Create a Local User

Even though almost anything could be configured to be accessed by networked users, the CANCEL action presets some additional requirements.

A local user is required for any CANCEL action on machines, where the network has to be disconnected as an intermediary step during the shutdown of the machine. Obviously, the process hangs, when some non-cached-modules has to be loaded via the disconnected connection, e.g. by NFS. This is required for now only for the Xen-3.0.x vesion, when setting WoL on a NIC, which is part of the virtual bridge, and may change for later versions. For details refer to Section **??** '**??**' on page **??** .

4.1.3 Setup Access Permissions

When the decision for the facilities to be used for authentication and authorization is made, and the configuration is finished, the specific required access permisssions for system calls by ctys could be established.

To check the actually present permissions for system calls the embedded debugging interface could be used. There are two basic approaches available:

- ctys-plugins
  This is the specific validation utility, which is a framework including some additional statistical display for the current states of plugins.

  ctys-plugins -d 64,P -T all -E

  A second variant validates the local client functionality.

ctys-plugins -d 64,P -T all

- ctys

  ctys -d 64,P -T all -a list

  This lists all used system-calls with their actual checked execution states.

The check calls are actually the verified system calls encapsulated by a wrapper and performed with a more or less harmless option. The result is verified and weighted dependent of the actual call environment.

4.2 Configuration

4.2.1 Plugins

The following listed configuration file defines the actually used plugins. For example the VMW plugin is active for Linux OSs only, but neither when running on Solaris nor on OpenBSD.

For conceptual information refer to Section ?? '??' on page ?? , particularly to the Section ?? '??' on page ?? .

```
#When set, the bootstrap-loader ignores the given
# <plugin-type>.
#This should be in case of dependencies such as of XEN from
#VNC utilized carefully. But on machines with OpenBSD
#e.g. the plugins VMW and XEN could be set to ignore safely.
#
#export PM_IGNORE=1
#export CLI_IGNORE=1
#export X11_IGNORE=1
#export VNC_IGNORE=1
#export XEN_IGNORE=1
#export VMW_IGNORE=1
#export QEMU_IGNORE=1
#export OVZ_IGNORE=1
#
#
# This could be configured conditionally for each of the
# following variables, and any other valid shell variable.
# This is particularly helpful in case of NFS mounted home
# directories sharing the identical user-configuration for
# multiple machines within a cluster:
#
#   MYHOST   : actual host
#   MYOS     : actual OS
#   MYOSREL  : release of actual OS
#   MYDIST   : actual distribution
#   MYREL    : release of actual distribution
#
```

```
#
# Some examples for ignoring of XEN-plugin on specific
# sets of nodes:
#
#   ->host01 only
#     [ "$MYHOST" == "host01" ]&&export XEN_IGNORE=1;
#
#   ->Any node NOT on "clust0*"
#     [ "${MYHOST#clust0*}" == "${MYHOST}" ]\
#           &&export XEN_IGNORE=1;
#
#   ->Any node IS on "clust0*"
#     [ "${MYHOST#clust0*}" != "${MYHOST}" ]\
#           &&export XEN_IGNORE=1;
#
#   ->Any node in domain "exe1"
#     [ "${MYHOST##*.exe1}" == "${MYHOST}" ]\
#           &&export XEN_IGNORE=1;
#
#   ->Any node NOT running OpenBSD
#     [ "${MYOS}" != "OpenBSD" ]\
#           &&export XEN_IGNORE=1;
#
#   ->Any node IS running Linux
#     [ "${MYOS}" == "Linux" ]\
#           &&export XEN_IGNORE=1;
#
#   ->Any node IS running CentOS
#     [ "${MYDIST}" == "CentOS" ]\
#           &&export XEN_IGNORE=1;
#
#   ->Any node which IS NOT final execution target
#     REMARK: for now patched: [ -z "$CTRL_EXECLOCAL" ]\
#     [ -z "‘echo $*|sed ....‘" ]\
#           &&export XEN_IGNORE=1;
#
#   ->....
#
#
# Almost any combination and any additional constraint
# could be added by means of bash.
#
# When using internal state variables the user is
# responsible for any resulting side effect.
#


##################
#Configuration of supported contexts for standard
#plugins.
#
```

```
#PM supports currently Linux and OpenBSD
[ "${MYOS}" != "OpenBSD" -a "${MYOS}" != "Linux" ]\
&&export PM_IGNORE=1;

#XEN is supported on Linux only as server, else as
#client only.
#VNC check will be done by plugin
#Check for "-e", because CTRL_EXECLOCAL is not yet
#initialized.
[ "${MYOS}" != "Linux" -a -n "‘echo " $* "|sed -n ’/ -e /p’‘" ]\
&&export XEN_IGNORE=1;

#VMW is supported on Linux only, else as client only.
#Native local client and VNC access for WS6 will be
#checked by plugin.

[ "${MYOS}" != "Linux" ]\
&&export VMW_IGNORE=1;

###########################################################
#                                                         #
#ATTENTION:                                               #
#  The plugins CLI+X11+VNC could be called MANDATORY for  #
#  others, so their "IGNORE-ance" might force unforseen   #
#  side-effects, think twice!!!                           #
#                                                         #
#  Same is true for PM when you require WoL and           #
#  controlled PM shutdown, what might be obvious!         #
#                                                         #
###########################################################
```

.

4.3 ctys-custom-setup

4.3.1 General

The installation of the UnifiedSessionsManager is either proceeded by utilization of or by installation of standard installation packages - such as rpm - which utilize the ctys-distribute call for local adjustment of the installation context.

The basic design of the ctys-distribute tool is the remote transformation and installation of the current local filesystem subtree for the actual executed version, thus it efectively clones it's own contained installation. Therefore the root of the installation subree for the actually called executable is evaluated and used as the source base for the current install procedure. The default behaviour is based on the PATH evaluation mechanism, whereas a specific version could be installed by simply calling ctys-distribute with a path-prefix. Due to enhanced application of the UNIX symbolic links the actual link-free path is evaluated before calling any further actions. This is also applied, when a shared installation is used by linked references only. The local user configuration is ignored and has to be adapted in addition to the installation.

In case of the application of **ctys-distribute** the customization could be simply done by removing or adding files to a custom tree. The directory trees foreseen for customization of third-party packages and the amount of forwarded documents are:

- doc

  `<install-base-dir>/ctys-<version>/doc`

  Copied and used by directory-subtree.

- downloads

  `<install-base-dir>/ctys-<version>/downloads`

  Copied by directory-subtree, used for current ctys version itself only.

- packages

  `<install-base-dir>/ctys-<version>/packages`

  Copied by directory-subtree, could be manually used for any content.

In addition the software itself could be extended by the plugins mechanism.

4.3.2 A short Receipt

Man-Pages only

The installed packages could be prepared for lean installation with man-pages only as follows:

1. Install BASE-package.
2. Install DOC-package.
3. Remove all files within the 'doc-tree' except the man-subtrees.

   `<install-base-dir>/ctys-<version>/doc`

4. Proceed with installation.

No Intro

The installed packages could be prepared for lean installation without the introduction material as follows:

1. Install BASE-package.

2. Install DOC-package.
3. Remove the 'intro-tree'.

   ```
   <install-base-dir>/ctys-<version>/doc/intro
   ```

4. Proceed with installation.

Add documents

The installed packages could be prepared with additional documents as follows:

1. Install BASE-package.
2. Install DOC-package.
3. Copy additional documents into the appropriate sub directories of the 'doc-tree'. Even though the documents will be distributed with the standard documents, the user interface components like HTML files are compiled static, therefore require manual customization.

   ```
   <install-base-dir>/ctys-<version>/doc
   ```

4. Proceed with installation.

.

4.4 ctys-multisite-install

4.4.1 General

The installation of the UnifiedSessionsManager is either proceeded by utilization of **ctys-distribute** or by installation of standard installation packages - such as rpm - which utilize the ctys-distribute call for local adjustment of the installation context.

The standard tools utilize their specific distribution mechanisms, e.g. by the RHEL environment for automation of software-distribution and update management.

The second choice for the UnifiedSessionsManager offered by **ctys-distribute** is the simple distribution by usage of **scp** from the **OpenSSH** package. This provides for several target installation contexts, so for private user-land setups too. In the latter case just a remote copy of the local effective installation is setup, which avoids particularly the requirement for administrator permissions. For required system tools and the appropriate setup still some administrator permissions are required.

4.4.2 A short Receipt

Sequential Install - One Target-User

The **ctys-distribute** utility provides for multi-site installations by default, but by sequential procedure only, the following call uses on each target host the same account. The default account is the callers login.

```
[<install-dir>/bin/]ctys-distribute -P uhc -l userX targetA targetB ... targetN
```

Sequential Install - Multiple Target-Users

The **ctys-distribute** utility provides for multi-site installations by default, but by sequential procedure only, the following call uses various individual accounts for each target host.

```
[<install-dir>/bin/]ctys-distribute -P uhc userA@targetA userB@targetB ... userN@tar
```

Parallel Install

This could be extended by a simple shell script for in-parallel execution:

```
for i in userA@targetA userB@targetB ... userN@targetN;do
   [<install-dir>/bin/]ctys-distribute -P uhc $i 2>&1 >/tmp/$i.out &
done
```

Parallel Install with Max-Tasks

The following script variant controls the actual maximum number of tasks performed in parallel:

```
#!/bin/bash

MAX=10;
n=0;
TARGETS="userA@targetA userB@targetB ... userN@targetN";

for i in ${TARGETS};do
    if ((n<MAX));then
       [<install-dir>/bin/]ctys-distribute -P uhc $i 2>&1 >/tmp/$i.out &
       let n++;
    else
       wait
    fi

done
```

.

4.5 ctys-verify-install

4.5.1 General

The installation of the UnifiedSessionsManager is based on standard interfaces and tools only, but in some cases prerequisites may be missing. For the simple verification of the current operational state the tool **ctys-plugins** is provided.

This offers an interface for the evaluation of almost any called system interface presenting the actual availability and access state.

4.5.2 A short Receipt

Client Prerequisites

The following call checks for the client prerequisites of all present plugin types.

```
ctys-plugins -T all
```

Server Prerequisites

The following variant checks for the server prerequisites of all present plugin types.

```
ctys-plugins -T all -E
```

Multisite-Verification by '–quick-tab'

The following call

```
ctys-plugins -T all -E --quick-tab $(ctys-groups -X -m 8 netscan/all)
```

or the variant with limitation of error messages and some ssh info:

```
ctys-plugins -T all -E --quick-tab $(ctys-groups -X -m 8 netscan/all)\
  2>/dev/null
```

Displays a short table of individual overall access states for selected plugins.

```
Hostname          | PM  | CLI X11 VNC RDP | KVM  QEMU  VBOX VMW  XEN
------------------+-----+-----------------+-------------------------
root@lab05.soho   | PAE | x   x   x   x   | -    -     -    S2   -
root@lab02.soho   | VMX | x   x   x   x   | -    QEMU  V    -    -
root@hydra.soho   | VMX | x   x   x   x   | -    -     -    -    -
```

```
root@delphi.soho   | VMX | x   x   x   x   | -    -    -   S1   -
root@olymp.soho    | VMX | x   x   x   x   | -    -    -   -    -
root@app1.soho     | VMX | x   x   x   x   | -    -    -   -    -
root@app2.soho     | VMX | x   x   x   x   | -    -    -   -    -
root@lab04         | VMX | x   x   x   x   | -    -    -   W7   -
acue@lab04         | VMX | x   x   x   x   | -    QEMU -   W7   -
vadmin@delphi.soho| VMX | x   x   x   x   | -    -    -   S1   -
acue@lab05.soho    | PAE | x   x   x   x   | -    -    -   S2   -
acue@lab02.soho    | VMX | x   x   x   x   | -    QEMU V   -    -
acue@delphi.soho   | VMX | x   x   x   x   | -    -    -   S1   -
root@lab03         | PAE | x   x   x   x   | -    -    -   -    HVM
root@lab01.soho    | PAE | x   x   x   -   | -    -    -   -    PAR
5000@lab01.soho    | PAE | x   x   x   -   | -    -    -   -    -
acue@lab03         | PAE | x   x   x   x   | -    -    -   -    -
acue@app1.soho     | VMX | x   x   x   x   | KVM  KVM  -   -    -
acue@app2.soho     | VMX | x   x   x   x   | KVM  KVM  -   -    -
```

Additional information is available by **ctys-plugins** .

.

4.6 ctys-web-setup

4.6.1 General

The DOC package contains the online help as a complete web site, which is available from the internet too.

4.6.2 A short Receipt

The setup is quite simple, the steps required are:

1. Install BASE and DOC-package.
   The shared installation of rpm-packages could be performed by:

   ```
   yum install 'ctys-*'
   ```

   or

   ```
   rpm -i ctys-base-<version>.noarch.rpm
   rpm -i ctys-doc-<version>.noarch.rpm
   ```

   The shared installation of tgz-packages could be performed by:

   ```
   tar -C /opt -zxf ctys-base-<version>.tgz
   tar -C /opt -zxf ctys-doc-<version>.tgz
   ```

   Private link and activation for a user including setup of personal configuration files.

   ```
   /opt/ctys-<version>/bin/ctys-distribute -F 1 -P uhl
   ```

   The following call replacess previous configuration files.

```
/opt/ctys-<version>/bin/ctys-distribute -F 2 -P uhl
```

2. When a local download mirror for software distribution is required, just copy the archives into the directory

   ```
   <install-base-dir>/ctys-<version>/downloads
   ```

   Which is refered to by a symbolic link from the directory

   ```
   .../doc
   ```

   The default path of 'downloads' for rpm-packages is:

   ```
   /opt/ctys-<version>/downloads
   ```

   The default path of 'downloads' for the local installation within a users HOME is:

   ```
   $HOME/lib/ctys-<version>/downloads
   ```

3. In case of Apache-Webserver just set a sysmbolic link:

   ```
   ln -s /opt/ctys-<version>/doc /var/www/html
   ```

   Do not forget the 'FollowSymLink' option and the permissions for each entry of the whole source path.

.

4.7 ctys-network-filesystem-design

4.7.1 General

In distributed environments with a worker and storage architecture the common location independent view of the processing nodes to the task data is the main design issue. This could be easily provided by defining a directory structure stored on a network filesystem like NFS or AFS with identical mount points on each machine. This has to be provided for the hypervisor access points to the stored VMs as well as for the contained filesystem structure of the GuestOSs.

The following figure depicts the views for Xen and QEMU based VMs.

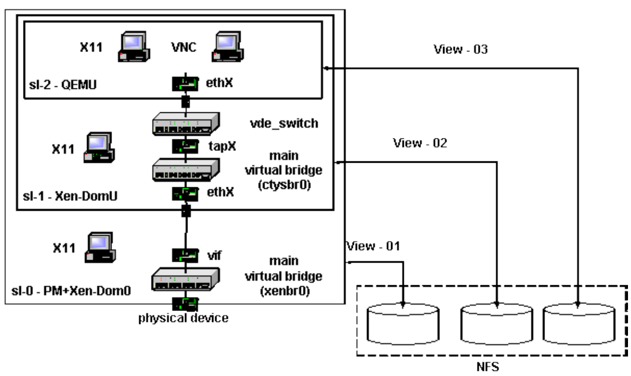


Figure 4.2: Network filesystem design

The main aspect is here the provided flexibility for the almost location independent roaming of generic VMs. Therefore two intermediate layers are introduced, first the interface of the hypervisors to stored VMs, second the execution and low-level access of the VMs to and from the PMs.

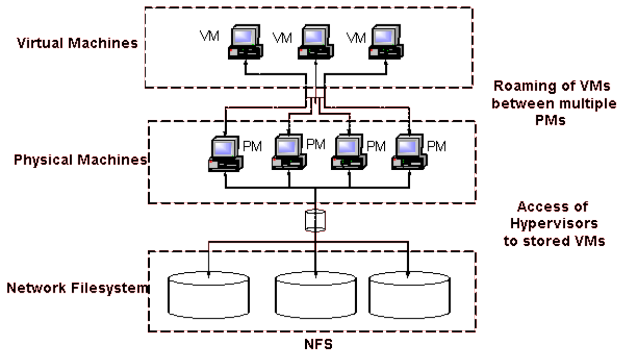


Figure 4.3: Network filesystem design - access locations

The result is a common view to the filesystem directory tree for each VM and it's contained GuestOS.

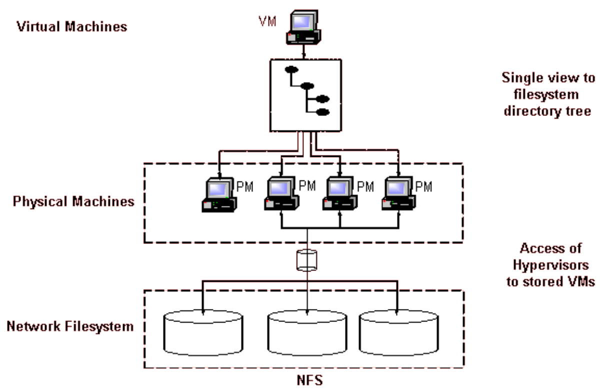


Figure 4.4: Network filesystem design - directory tree

The required access functionality for the addressing of specific stored VMs is provided by the UnifiedSessionsManager. This is e.g. required for some hypervisors and which reuqire a storage addressing slightly different from common filesystem names. The eventualy required conversion functions due to proprietary addressing of the Vendors is provided by the UnifiedSessionsManager.

Chapter 5

Gnome Setup

.

5.1 Abstract

The UnifiedSessionsManager is currently first of all a command line application, but designed for the application of complex multi-vendor and multi-screen environments of physical and virtual facilities. Thus it is a sessions management utility of modern workspaces, thus basically a graphics-control application.

The lifecycle of the UnifiedSessionsManager is going to evolve to the migration towards the extension for a graphical user interface in current versions. The first step for providing an additional graphical interface is the integration of the generated cache database into the Gnome based starter application. The first application here is a simple list based starter application for intermixed hypervisors, physical machines, and additionally the combination with the functionality for native logins into OSs by provided HOSTs plugins.

Initially two basic interfaces are provided from now own. The **CREATE** list element for the start of virtual machines, and physical machines either by Wake-On-LAN or (next) by IPMI. Second the **LOGIN** list element for native logins - either into Host-OSs running on the physical machines, or GuestOSs running within virtual machines.

Both applications are implemented quite simple in order to demostrate

- the efficienct integration of the database into an automated GUI application
- the simplicity of the implementation and possible customisation
- the simplicity of the seamless integration of distributed workspaces based on heterogeneous environments set up by physical and virtual machines, both intermixed with arbitrary runtime applications.

The following steps show the blueprint for the realworld application - these are actually almost the complete resulting application steps. These are described within the following chapters with additional application examples.

The following menu is provided as a basic menu and starting point as customisation pattern for automated installation by a script.

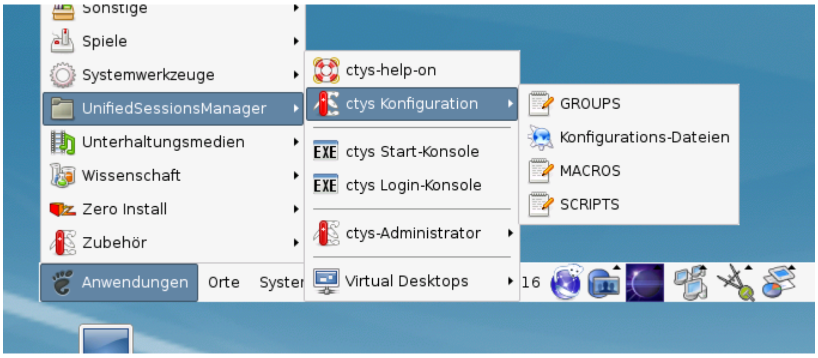


Figure 5.1: Gnome menu template

The next figure depicts the display of the database entries by the list 'ctys-CREATE-ALL', which contains all the complete unfiltered set of database record. The entry number '477' is selected for start, the resulting execution call is offered by the dialog box 'ctys-Selection' either for modification, or for execution by confirmation.

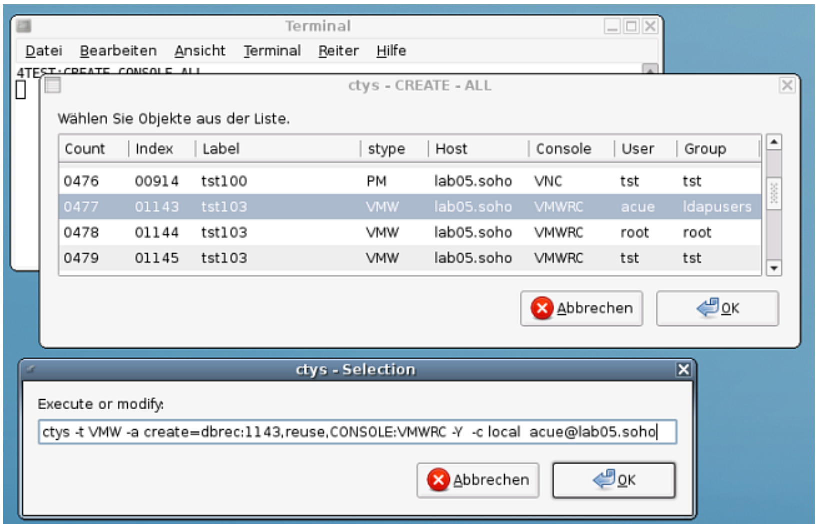


Figure 5.2: Gnome starter - CREATE

The internal data of the operational data storage, is integrated here by just a few lines of scripting with **zenity** into the Gnome based desktop, providing a graphical starter application. The scope of managed entities within the displayed list comprises here all supported VMs and PMs, including native logins by CLI, X11-Terminals, VNC, and RDP. Due to stored defaults, in addition to hypervisor consoles also the preconfigured native login applications could be started automatically.

This demonstrates the combination and setup of **ctys-vdbgen(1)** for automatic creation of the **cacheDB** and the **gnome-starter(1)** application for graphical presentation of the data within some minutes. The presented test environment on a medium-range machine for example contains about 350VMs used by NFS on several machines resulting in about 1400entries for VMs and PMs by multipath-access. The whole automated initial creation of the cacheDB takes some Minutes for scanning filesystems and presents than the shown graphical interface with a startup in the range of seconds for each selected VM.

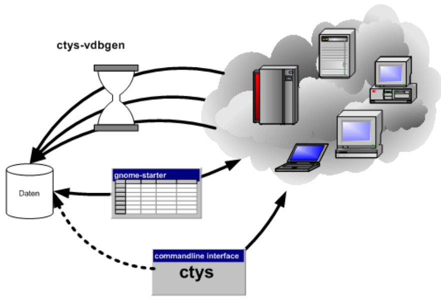


Figure 5.3: Database creation and application

The database contains the hypervisors KVM, QEMU, VBOX, VMW, and XEN, additionally PMs. For each entry also default login desktops or commandline applications are stored for automation of console interconnection as well as native logins - based on CLI, X11, VNC, and RDP. Additionally a first minor version of customisable menus for Gnome based desktops is integrated. These offer either for private menus, or shared common menus with a pattern for simple customisation.

The next important aspect when working with huge amounts of machines and consoles - either physical, or virtual, or just ordinary remote desktops - is the usability of the workspace on the desktop. This frequently requires the application of multiple displays.

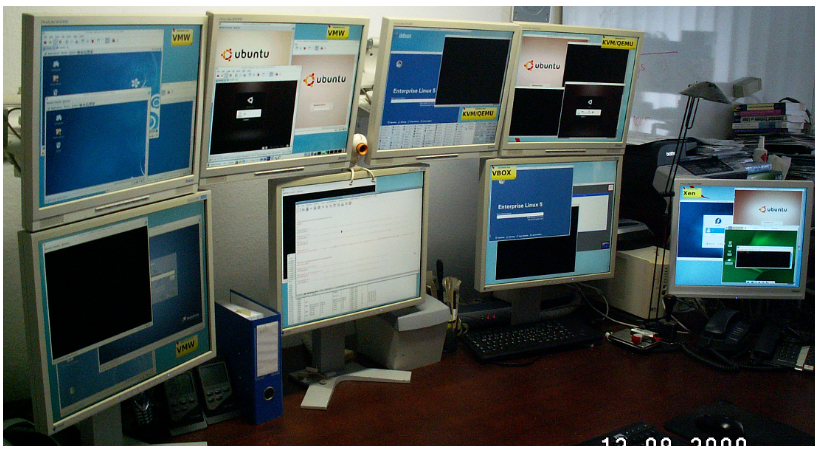


Figure 5.4: Gnome Xinerama

The Xinerama mode is provided by a specific extension for logical addressing of the standard X11 functionality.

The previous examples are just performed by a mouse-click - within seconds - no longer by painful manual execution or enduring custom-scripts development.

5.2 Overview

REMARK: This part and related specific utilities are temporary pre-releases for demonstration of first draft graphical integration due to requests. Thus these are in alpha-state, and are planned to be replaced by a graphical application in future versions.

The current version supports the manual integration of ctys into menu entries of any X11 based desktop. This document describes the manual setup of ctys based menu entries for Gnome desktops.

The additional standard Gnome utility required for GUI based start of VMs is **zenity**, which offers an easy-to-use minimalistic graphical interface particularly for list-elements. This simply could be called from a shell script and returns the selected data. Another **zenity** widget e.g. provides a text-box for modification and confirmation of the current call.

5.3 Gnome - Automated menu Creation with Templates

The following menu is provided as a pattern to be installed either as a private, or as a shared menu. The configuration files and install scripts are installed within the directory **$HOME/.ctys/xdg.d**.

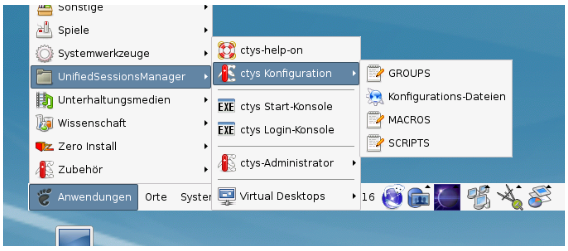


Figure 5.5: Gnome menu template

The private installation is performed by **ctys-xdg –menu-create** , whereas the shared installation is performed by the script **ctys-xdg –menu-create –menu-shared** The shared installation is stored within the common system directories and therefore requires root permissions. For the private installation the user permissions are sufficient.

The private base directories used are the standard directories as defined by the **Desktop Menu Specification-1.0** from **freedesktop.org**. These are **$HOME/.config**, **$HOME/.icons**, **$HOME/.local/share/applications**, and **$HOME/.local/share/desktop-directories**.

The following description offers a simple and fast setup of the predefined menus by just one script-call. But the whole process could be performed manually by the property menus too as described within the specific chapters.

5.3.1 Gnome - Private and Shared menus

Gnome offers facilities for the creation of custom menus, which could be either setup individually within a private environment, or as a shared menus for all users on a specific machine. The resulting functional possibilities are basically similiar, thus the intention of usage and flexibility is the major aspect for the design choice.

Whereas the resulting functionality is quite similiar, the physical storage of the configuration infomation is obviously different. The common shared menu entries for the machine are stored within the system directories, with write operations requiering root permissions. The private environments are stored commonly within individual subdirectories of the specific user, thus frequently are visible just for the owner.

The configuration decision here is to use the custom files for static setups as defined by Gnome, even though some configuration with toolsets for runtime dynamic configuration might be applicable. Nevertheless, due to the contibous automatic synchronisation of the Gnome desktop with it's configuration files, some - almost unlimited - runtime dynamic is available inherently by default.

The applied directory structures are designed and named in order to minimize the deviation between the shared and private variant.

5.3.2 Private menus

Customization-Hook - gnome-applications.menu

The menu structure of Gnome is defined by a basically tree based data structure, which is defined by XML syntax within a nested file structure. The configuration syntax is a powerful set of language elements particularly offering for file and directory based inclusion configuration data extension. This is extended by means of decisive syntax elements for suppression, additional, exclusive, and non-exclusive extension of present menu configuration data.

The UnifiedSessionsManager menus are hooked-in into the configuration file:

`$HOME/.config/menus/gnome-applications.menu`

This is done by the entry

```
<MergeFile>applications-merged/ctys-UnifiedSessionsManager.menu</MergeFile>
```

The entry is located within the file for example as depicted in the following figure, where the file itself is contained within a subdirectoy 'applications-merged' and addressed by a relative path:

```
<!DOCTYPE Menu
  PUBLIC '-//freedesktop//DTD Menu 1.0//EN'
  'http://standards.freedesktop.org/menu-spec/menu-1.0.dtd'>
<Menu>
  <Name>Applications</Name>
  <MergeFile type="parent">/etc/xdg/menus/gnome-applications.menu</MergeFile>
```

```
<!-- *** HOOK-START *** -->

  <MergeFile>applications-merged/ctys-UnifiedSessionsManager.menu</MergeFile>

<!-- *** HOOK-END *** -->

        <Menu>
                <Name>alacarte-made</Name>
                <Directory>alacarte-made.directory</Directory>
                <Include>
                        <Filename>alacarte-made.desktop</Filename>
                </Include>

                ...
                ...
                ...
```

The file is actually located within the pathname:

```
$HOME/.config/menus/applications-merged/ctys-UnifiedSessionsManager.menu
```

The menu definition is hereby assembled by the following file types in accordance to the standard Gnome and Common Desktop definitions:

- **menu**
  The structure definition for teh directory tree including hooks for node descrition by directory-files and desktop-files.
- **directory**
  Contains the attributes for specific tree-nodes of the directory tree, which includes particularly the Native Language Support - NLS - for the representation.
- **desktop**
  Contains the attributes for specific leaf-nodes of the directory tree, which includes the Native Language Support - NLS - for the representation, and particularly additinal information for the resulting execution call of the application.
- **graphical element**
  This represents the represented icons in various formats.

Custom menu - ctys-UnifiedSessionsManager.menu

The hooked-in file contains the current version the whole structure definition of the ctys-menu-hierarchy as single standalone structure-definition file. The menu **UnifiedSessionsManager** is here included **as a sub-menu** of the **Applications** menu.

```
<!DOCTYPE Menu PUBLIC "-//freedesktop//DTD Menu 1.0//EN"
  "http://www.freedesktop.org/standards/menu-spec/menu-1.0.dtd">
<!-- Do not edit manually - generated and managed by xdg-desktop-menu -->
<Menu>
```

```
<Name>Applications</Name>

<AppDir>/homen/acue/.gnome/ctys/applications/</AppDir>
<DirectoryDir>/homen/acue/.gnome/ctys/desktop-directories/</DirectoryDir>


<Menu>
  <Name>UnifiedSessionsManager</Name>
  <Directory>ctys-UnifiedSessionsManager.directory</Directory>
  <Include>
    <Filename>ctys-help.desktop</Filename>
    <Filename>ctys-CREATE-CONSOLE.desktop</Filename>
    <Filename>ctys-LOGIN-CONSOLE.desktop</Filename>
  </Include>


  <Layout>
    <Merge type="all"/>
    <Filename>ctys-help.desktop</Filename>
    <Menuname>ctys-admin</Menuname>
    <Merge type="menus"/>
    <Merge type="files"/>
    <Separator/>
    <Filename>ctys-CREATE-CONSOLE.desktop</Filename>
    <Filename>ctys-LOGIN-CONSOLE.desktop</Filename>
    <Separator/>
    <Menuname>Sysadmin</Menuname>
    <Separator/>
    <Menuname>Desktops</Menuname>
  </Layout>


   <Menu>
     <Name>Sysadmin</Name>
     <Directory>ctys-Administrator.directory</Directory>

     <Menu>
<Name>hosts</Name>
<Directory>ctys-hosts.directory</Directory>
<Include>
          <Filename>ctys-localhost.desktop</Filename>
</Include>
     </Menu>

     <Directory>ctys-Administrator.directory</Directory>
     <Include>
       <Filename>ctys-root-term-localhost.desktop</Filename>
     </Include>
   </Menu>

   <Menu>
```

```
      <Name>Desktops</Name>
      <Directory>ctys-Desktops.directory</Directory>

      <Menu>
<Name>hosts</Name>
<Directory>ctys-hosts.directory</Directory>
<Include>
          <Filename>ctys-localhost.desktop</Filename>
</Include>
      </Menu>

      <Directory>ctys-Desktops.directory</Directory>
      <Include>
        <Filename>ctys-localhost.desktop</Filename>
      </Include>
    </Menu>


   <Menu>
     <Name>ctys-admin</Name>
     <Directory>ctys-admin.directory</Directory>
     <Include>
       <Filename>ctys-CONFIG.desktop</Filename>
       <Filename>ctys-GROUPS.desktop</Filename>
       <Filename>ctys-MACROS.desktop</Filename>
       <Filename>ctys-SCRIPTS.desktop</Filename>
     </Include>
   </Menu>

  </Menu>
</Menu>
```

.

Custom desktop-directories

The following entry depicts as an example representation the data for the entry node of the **configuration submenu** for the **UnifiedSessionsManager** itself.

```
[Desktop Entry]
Name=ctys configuration
Name[de]=ctys Konfiguration
Name[en_GB]=ctys configuration
Comment=Configuration Files
Comment[de_DE]=Konfigurations Dateien
Comment[en_GB]=Configuration Files
Type=Directory
Icon=applications-utilities
Encoding=UTF-8
```

Custom applications

The following entry depicts as an example representation the data for the entry node of the configuration for the **gnome-starter(1)** with **CREATE**. submenu for the UnifiedSessionsManager itself.

```
[Desktop Entry]
Type=Application
Version=1.0
Encoding=UTF-8
Name=ctys Create Console
Comment=ctys Create Console
Name[en_CA]=ctys Create Console
Name[en_GB]=ctys Create Console
Name[de_DE]=ctys Start-Konsole
Comment[de]=ctys Start-Konsole
#Possible absolute PATH:
#  Exec=/homen/acue/utils/gnome-starter CREATE CONSOLE ALL
Exec=gnome-starter CREATE CONSOLE ALL
#Possible absolute PATH:
#  Icon=/opt/i4p/patches/icons/misc-collection/ctys-exe.svg
#Possible absolute PATH:
#  ~/.icons
Icon=ctys-exe
Terminal=true
Categories=System;
```

Custom pixmaps

The pixmaps could be stored within several directories in multiple formats and resolution. The current choice for the UnifiedSessionsManager is **svg** and stored in a **pixmaps** directory coallocated with the directories **applications** and **desktop-directories** in accordance to the standard sub-structure of **/usr/shared** directory tree. This simplifies the common installation procedure for private and shared installs.

5.3.3 Shared menus

The share instalation is basically the same, the deviation is basically just within the root nodes for installation. The **applications**, **desktop-directories**, and **pixmaps** directories are allocated within the **/usr/shared** directory tree.

The **applications-merged** files are just copied into the **/etc/xdg/menus/applications-merged** subdirectory. Due to present inclusion of all files within this directory no file-patch is required in this case.

5.4 Gnome - Basic Manual menu Creation

The first step is to open the dialog box for the menu entries by mouse context menu by right-mouse-click.

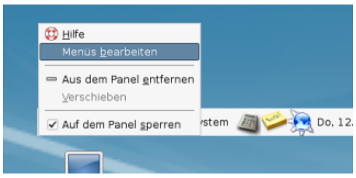


Figure 5.6: Manual Gnome menu creation - Open context

The dialogue box enables the creation and setting of submenus and entries.

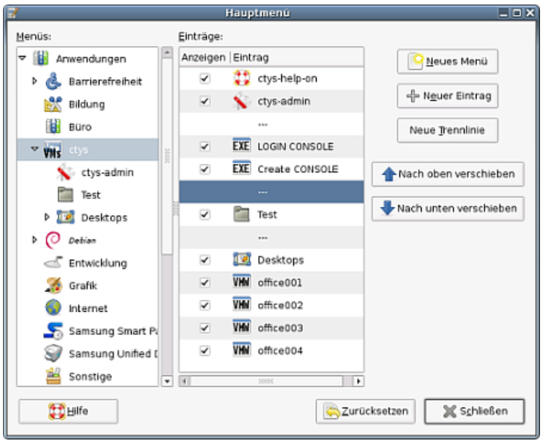


Figure 5.7: Manual Gnome menu creation - Dialogue

5.4.1 Entries for Scripts

A script is a shell executable, which could contain ctys calls intermixed with native shell calls. Thus this is particularly suitable for desktops containing various local applications. The setup of a script is quite easy due to simple syntax as well as by tool support with configurable standard editors. The following example shows the 'manuals01.sh' entry for the creation of the complete desktop for editing the manuals of ctys.

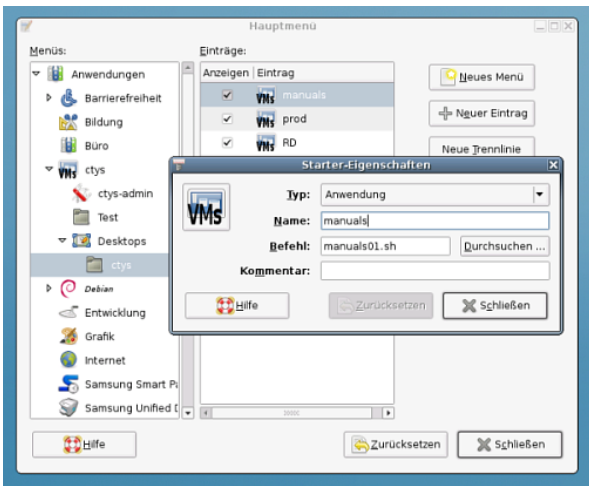


Figure 5.8: Script Entries

The setup is given as:

`manuals01.sh`

Where the content is:

```
#!/bin/bash


#
#Prepare environment
#
 . $(dirname ${0})/common.sh


#
#Start environment
#
if [ "$1" != "SETENV" ];then
    gnome-terminal --geometry=$(getGeometry  -g  180x20+0+0:A10) \
        --working-directory="$DOC_BLD_ROOT" \
        --title="DOC_BLD_ROOT" -x $0 SETENV&

    gnome-terminal --geometry=$(getGeometry  -g  180x10+0+350:A10) \
        --working-directory="$DOC_BLD_ROOT" \
        --title="DOC_BLD-01" -x $0 SETENV&

    gnome-terminal --geometry=$(getGeometry  -g  180x10+0+550:A10) \
        --working-directory="$DOC_BLD_ROOT" \
        --title="DOC_BLD-02" -x $0 SETENV&

    gnome-terminal --geometry=$(getGeometry  -g  180x10+0+750:A10) \
        --working-directory="$DOC_BLD_ROOT" \
        --title="DOC_BLD-03" -x $0 SETENV&

    nautilus --geometry=$(getGeometry  -g  1280x700+0+0:A20) $DOC_ALL&

    konq\u\eror --geometry=$(getGeometry  -g  1280x1048+0+0:A31) \
       $DOC_ALL_EN $DOC_ALL_DE&

    konq\u\eror --geometry=$(getGeometry  -g  1280x1048+0+0:A30) \
       $DOC_LIST_EN&

    konq\u\eror --geometry=$(getGeometry  -g  1280x1048+0+0:A00) \
       $DOC_BLD_ROOT $DOC_BLD_EN $DOC_BLD_DE &

    cd $BLD_ROOT && ctys desktops/dev/ctys/manuals01
else
    #
    #Set environment
    #
    if [ -z "$CTYS_ENVSET" ];then
```

```
        export CTYS_ENVSET=1
exec bash -i
    fi
fi
exit 0
```

The result is depicted in the following figure.

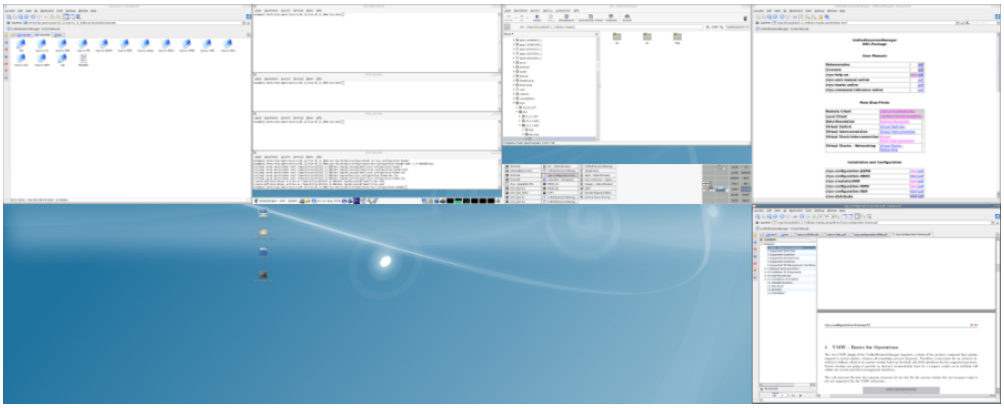


Figure 5.9: Script Entries - Resulting Desktop

5.4.2 Entries for GROUPs

GROUPS are sets of hosts and virtual machines for ctys only. This are particularly suitable for desktops build up by remote applications only, including remote desktops and consoles. The setup of groups allows for various specific parameters for each connection and executed desktop. In addition an overall task control is setup for a GROUP, which also could be setup by reusable modular includes for various purposes. The setup of a GROUP called by 'ctys' could be created in the same manner as script entries. The following example shows the 'ctys admin/admin0' group entry for the creation of the complete desktop for administration of some servers.

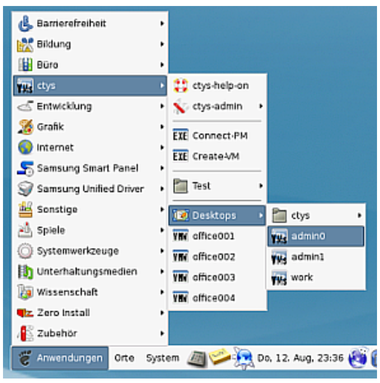


Figure 5.10: GROUP Entries - menu Entries

The menu entry could be setup as following, in this example:

```
/homen/acue/bin/ctys admin/admin0
```

With the content of the GROUP:

```
#  -*- mode: conf; -*-
#
#This groups contains all machines in the management group of the
#server group.
#

#
#fileserver - CentOS-5.4 - VMware-Server
root@delphi'( -t vnc -a create=l:DELPHI,reuse -g 1268x994:A00:ALL  -b 1,2)'

#
#backup-server - CentOS-5.4
root@olymp'(  -t vnc -a create=reuse,l:OLYMP  -g 1268x964:A10:ALL  -b 1,2)'

#
#database-server - CentOS-5.4 - KVM
root@app1'(   -t vnc -a create=reuse,l:APP1   -g 1268x994:A01:ALL  -b 1,2)'

#
#CUDA-server - CentOS-5.4 - KVM
root@app2'(   -t vnc -a create=reuse,l:APP2   -g 1268x994:A21:ALL  -b 1,2)'
```

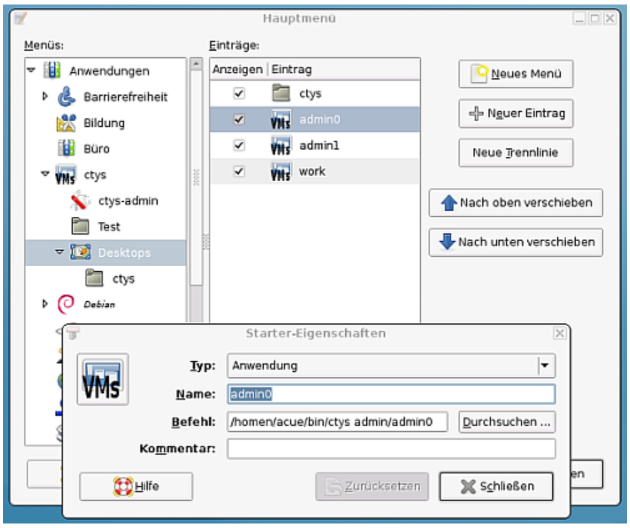


Figure 5.11: GROUP Entries

The result is depicted in the following figure.

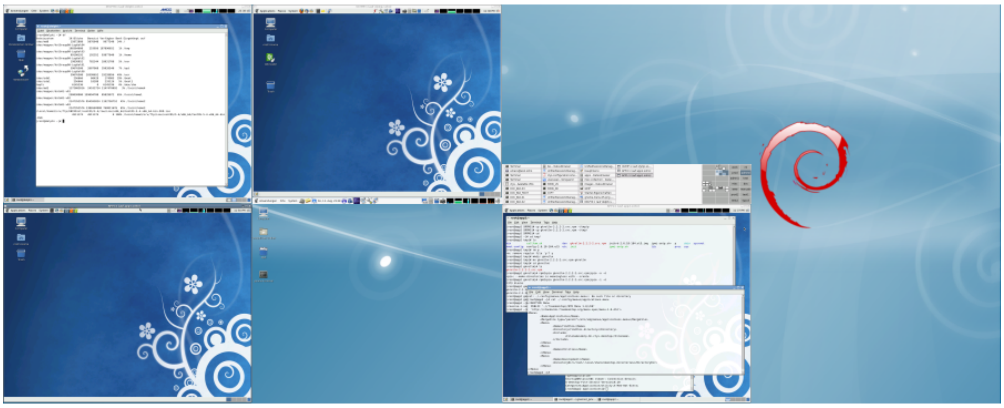


Figure 5.12: GROUP Entries - Resulting Desktop

5.4.3 ctys-help-on

The online help menu could be basically created with any tool from the package. The usage is:

```
ctys -H html=base
```

or

```
ctys -H html=doc
```

This opens a browser with the provided help file 'doc.html'. The preconfigured browser is konqueror by default for now, when not available firefox is used. Any browser could be customized by the user.

The script could be integrated into Gnome by just configuring a menu item and using the call for openning a (now still) draft online help by html and pdf files. Additionally the commandline interface man pages are available.

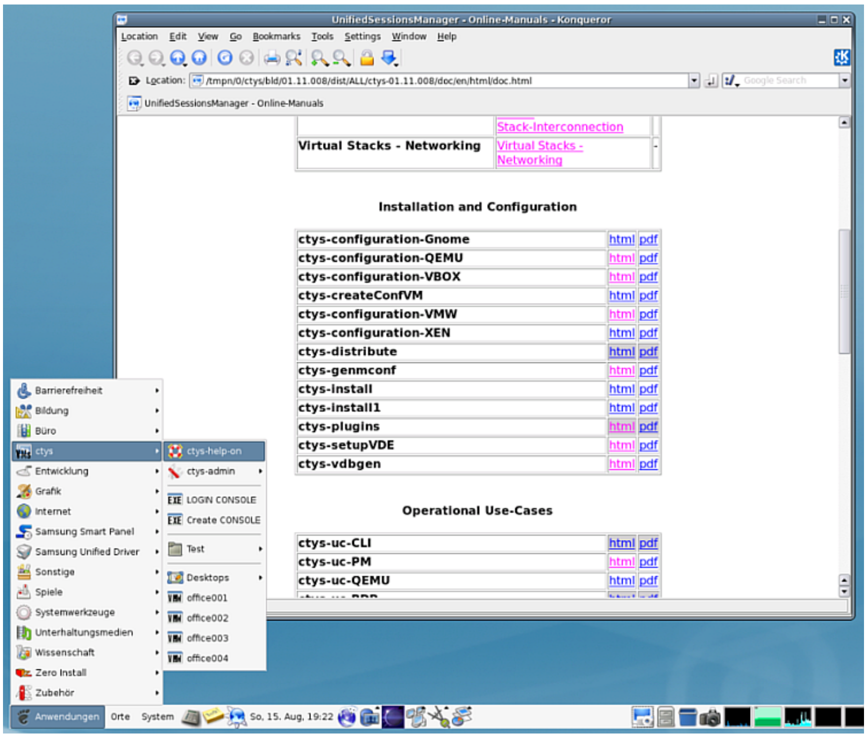


Figure 5.13: ctys-help-on - Online Help

5.4.4 GROUPS

The GROUPS objects are represented by files containing multiple host entries. These could be edited by a preconfigured editor with the following call, which could be used within menu entries.

```
ctys-groups -e
```

The started editor or filebrowser opens by default all configured directories within the CTYS_GROUPS_PATH. The preconfigured default is the Emacs editor, if not present vi, vim, konqueror, or nautilus are called. The user can customize any browser or file-manager as required.

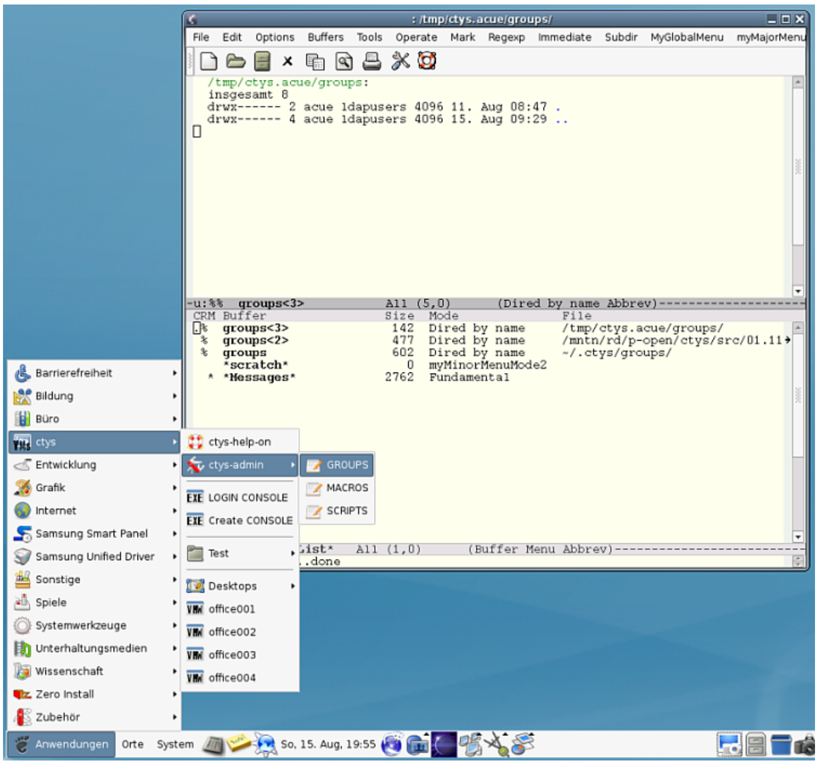


Figure 5.14: GROUPs - Emacs editor

This could be varied call-by-call e.g. by

```
CTYS_GROUPSEDIT=konqueror ctys-groups -e
```

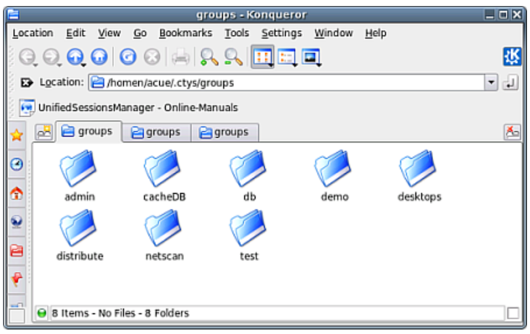


Figure 5.15: GROUPs - konqueror browser with tabs

5.4.5 MACROS

The MACROS are represented by files containing the set of definitions to be applied. These could be edited by a preconfigured editor with the following call, which could be used within menu entries.

```
ctys-macros -e
```

The started editor or filebrowser opens by default all configured directories within the CTYS_MACROS_PATH. The preconfigured default is the Emacs editor, if not present vi, vim, konqueror, or nautilus are called. The user can customize any browser or file-manager as required.

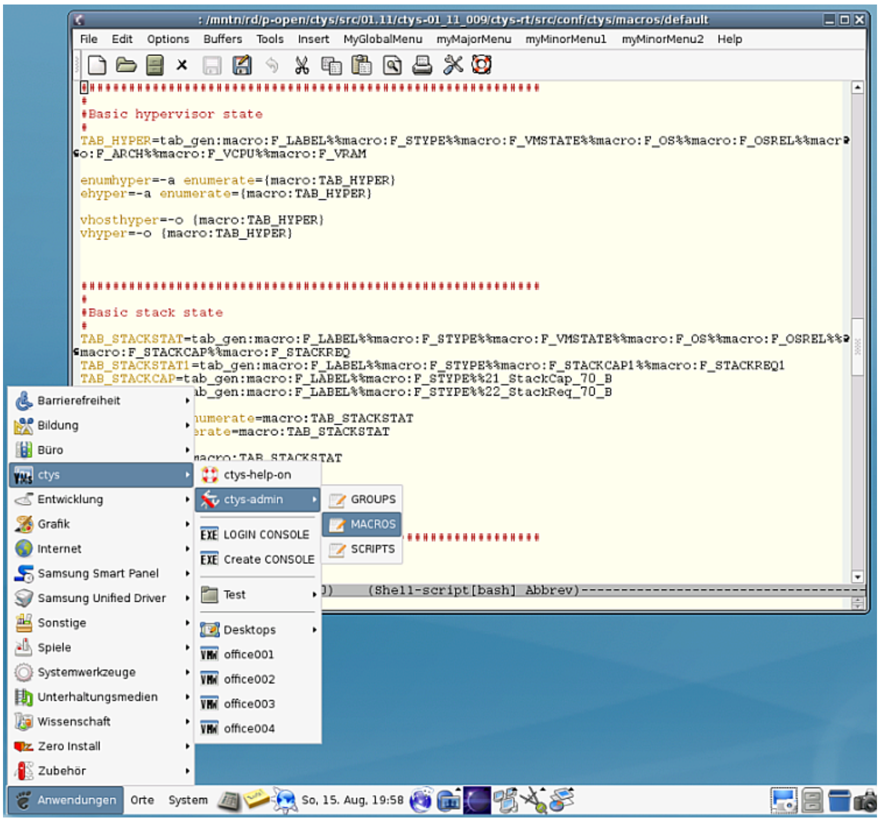


Figure 5.16: MACROS - Emacs editor

5.4.6 SCRIPTS

The SCRIPTS are contained in within the defined search path CTYS_SCRIPTS_PATH similar to the systems PATH variable. These could be edited by a preconfigured editor with the following call, which could be used within menu entries.

```
ctys-scripts -e
```

The started editor or filebrowser opens by default all configured directories within the CTYS_SCRIPTS_PATH. The preconfigured default is the Emacs editor, if not present vi, vim, konqueror, or nautilus are called. The user can customize any browser or filemanager as required.

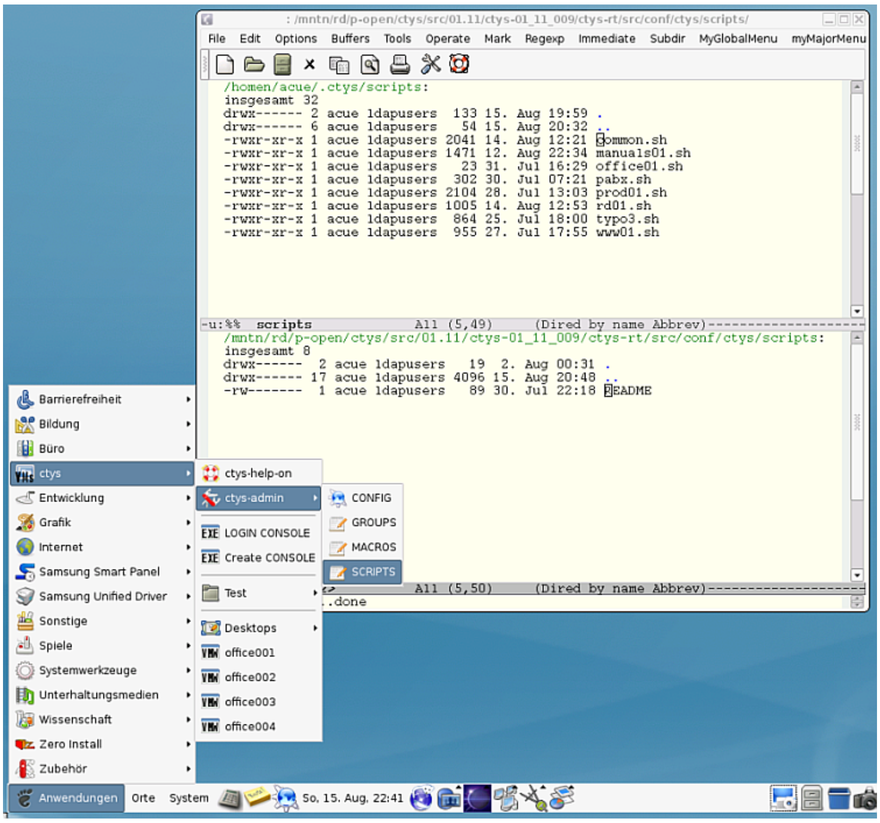


Figure 5.17: SCRIPTS - Emacs editor

This could be varied call-by-call e.g. by

```
CTYS_SCRIPTEDIT=konqueror ctys-scripts -e
```

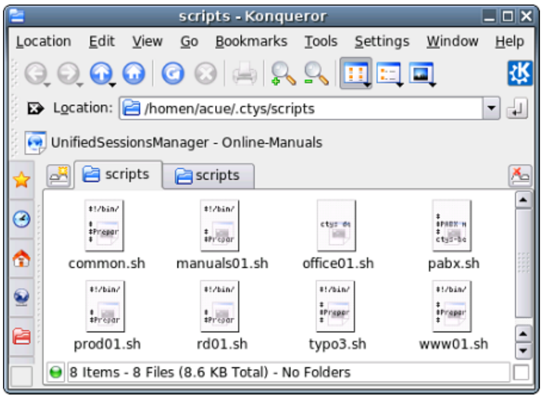


Figure 5.18: SCRIPTS - konqueror browser with tabs

5.4.7 CONFIGURATION

The configuration files could be browsed by **konqueror** with the following call:

```
ctys-config -e
```

The started editor or filebrowser opens by default all configured directories within the standard paths. The preconfigured default is the Emacs editor, if not present vi, vim, konqueror, or nautilus are called. The user can customize any browser or filemanager as required.

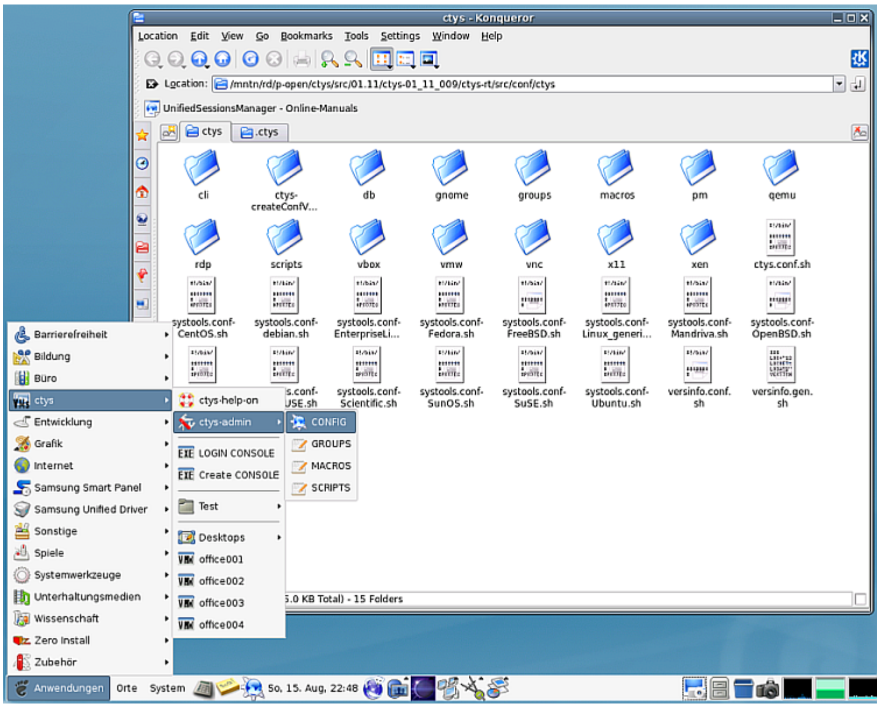


Figure 5.19: CONFIGURATION - konqueror browser

5.5 Graphical VM and PM Starter

The following example shows the configuration of a graphical starter based on **zenity**.

This is based on some pre-requirements in order to work. The main requirement is the presence of a cacheDB, which could be generated by the utility **ctys-vdbgen(1)** , for further information refer to the man pages of the tool. The second requirement is the static definition of the call parameters of the start call, which relies on the defaults of each plugin. These could be modified by the related configuration files, but may suffice the basic initial creation and reconnection(by REUSE) for each VM.

This works for all supported hypervisors due to the unique interface for all - providing basic features for each. Additional parameters such as the screen position and windows size are not provided for current version.

It is recomended to set the type of the menu entry to 'Start application in terminal' because some user interaction may be required. For example in case of SSH first time connection the new key has to be confirmed.

5.5.1 gnome-starter

The temporary utility **gnome-starter** accepts the parameters **PM**, **VM**, and **LIST**. The parameter PM sets the start of a PM login by usage of VNC, this also restricts the displayed choice-list to PMs only and additionally assures that the displayed records are unique each. The parameter VM works similiar for VMs, where only VMs, but of all supported types are displayed. Due to the generic standard-interface and apropriate defaults delivered, each could be started in the same manner.

The standard call syntax fo selection of subsets from the database is

```
gnome-starter <ACTION> <TARGET-TYPE> <SCOPE>
```

ACTION could be CREATE, LOGIN, or LIST. TARGET-TYPE is CONSOLE. SCOPE could be ALL, VM, or PM.

The current version does support for native access UNIX systems only. Thus even though guest systems could contain any OS supported by the hypervisor are successfully collected into the database, the selection in the starter for LOGIN actions may fail for non-supported systems. Specific parameters e.g. for individual screen positions are not provided by this version.

5.5.2 CREATE - PM and VM Starter

The PM starter could be configured in the same manner as any menu item. The required call is

```
gnome-starter CREATE CONSOLE PM
```

The generated standard call is:

```
ctys -t <type> \
     -a create=dbrec:<database-index>,reuse,CONSOLE:<current-default> \
     -Y  -c local <username>@<hostname>
```

This opens now 2 windows, the **zenity** list window and in addition a terminal window.

ATTENTION: The terminal window is required in cases, where user interaction is requiered. In some cases the system may even hang in an non-reachable console dialog, when a terminal is missing.

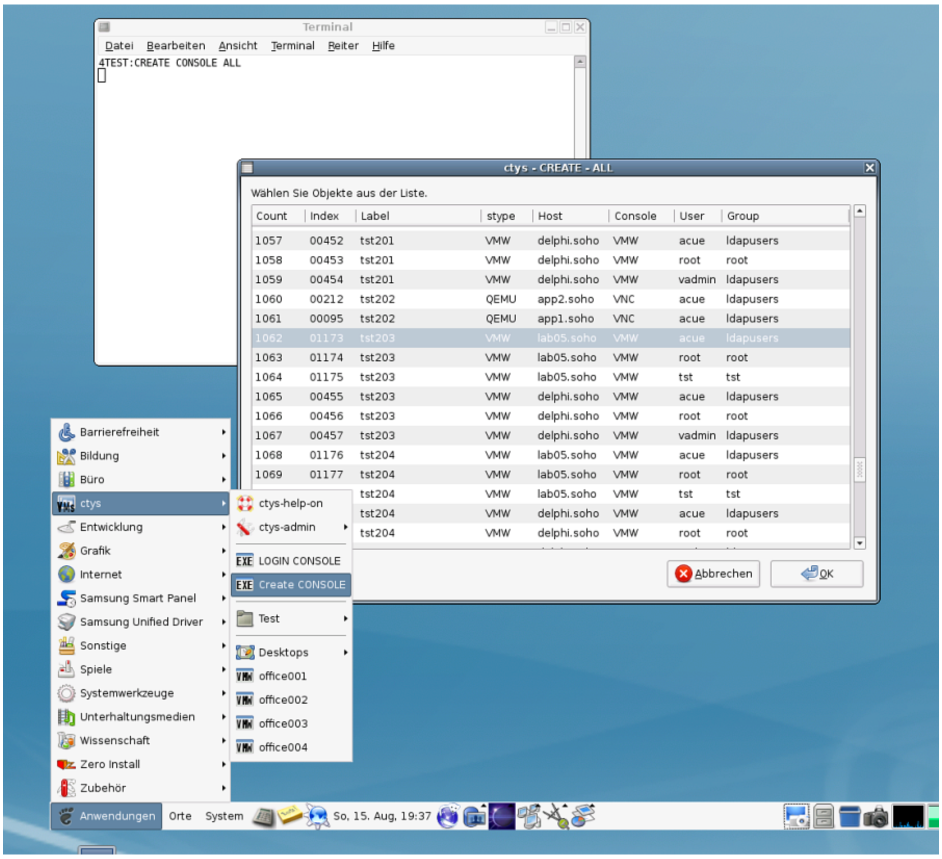


Figure 5.20: Gnome starter menu

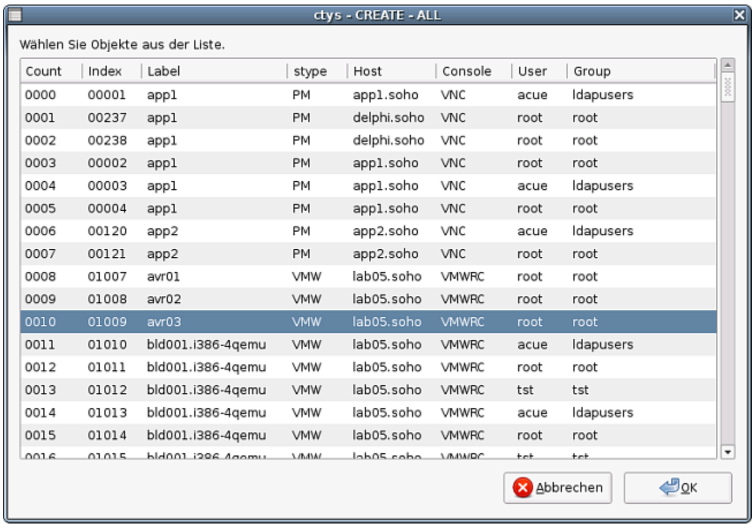


Figure 5.21: Gnome starter

After a specific entry is selected a second window is opened for call confirmation. This text-box allows for modification of the generated call if required. The confirmation of this second window starts the entry.

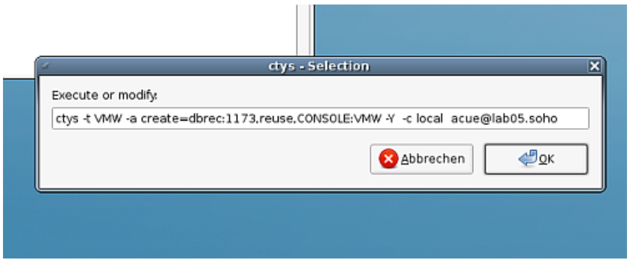


Figure 5.22: Call confirmation

The requested user interaction is here to confirm new host entry for OpenSSH.

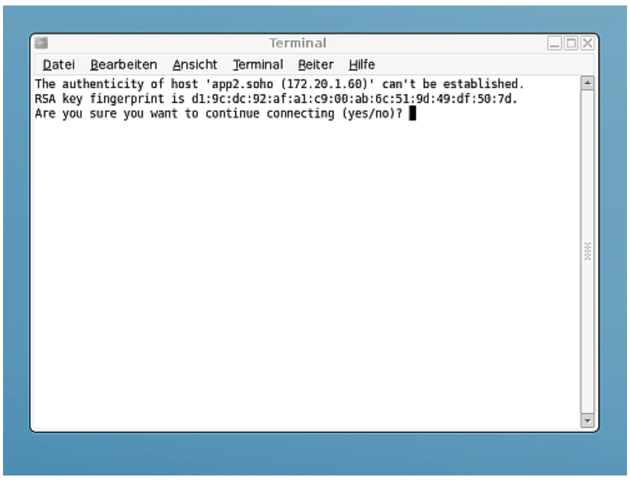


Figure 5.23: Gnome starter - SSH-Confirmation

The result is depicted in the following figure.

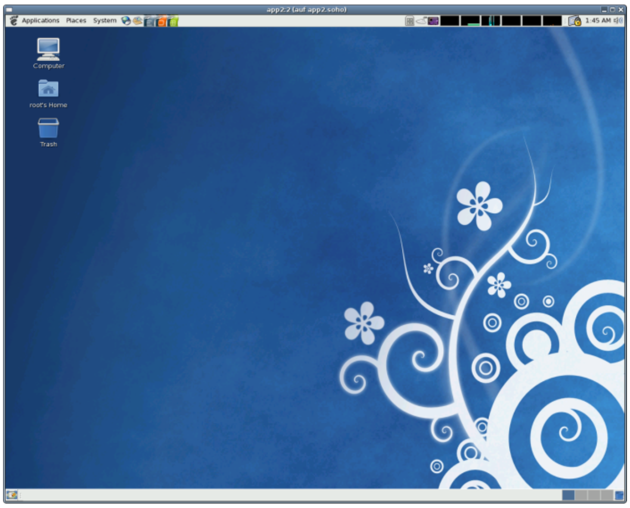


Figure 5.24: Gnome starter - Resulting Desktop

The VM starter could be configured in the same manner as any menu item. The required call is

```
gnome-starter CREATE CONSOLE VM
```

5.5.3 LOGIN - HOSTs Starter

The starter for LOGIN is similar but slightly different. This is due to required additional information related to the guest OS, such as the ip address of the contained guest system. Also the preconfigured default HOSTs CONSOLE is displayed and used instead of the CONSOLE to be attached to the hypervisor. In addition some helpful information related to the contained guest OS distribution are presented.

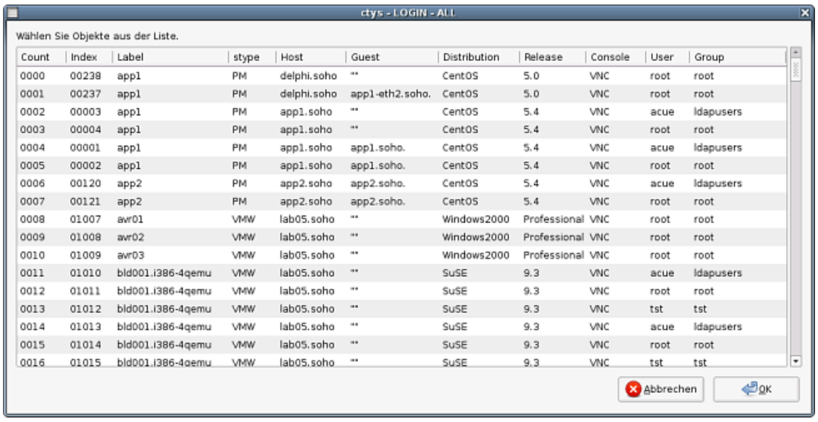


Figure 5.25: Gnome starter - LOGIN list

The resulting call is a complete call due to the much simpler structure for a simple login, than in case of CREATE, where additional data may be fetched form the database.

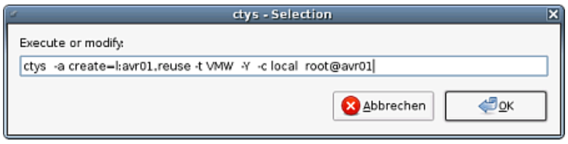


Figure 5.26: Gnome starter - LOGIN confirmation

5.5.4 Troubleshooting

In case of difficulties call the interface manually from the command line. The main pre-requirement is an existing cacheDB (refer to **ctys-vdbgen(1)**), which may not contain redundant entries. This could be verified by **ctys-vhost(1)** with the **-M** option.

When a call does not start from the menu entry the required PATH may not be complete. Thus one solution is to enter absolute pathnames for the executable, for example:

```
/home/yourHome/bin/ctys ...
```

5.6 Multiple Monitors - Xinerama Setup

The most important practical usability aspect when working with huge amounts of machines and consoles - either physical, or virtual, or just ordinary remote desktops - is the usability of the workspace on the desktop. The key for the usability is here the application of multiple display environments, where these could be either combined locally, or virtually from local and remote sites.

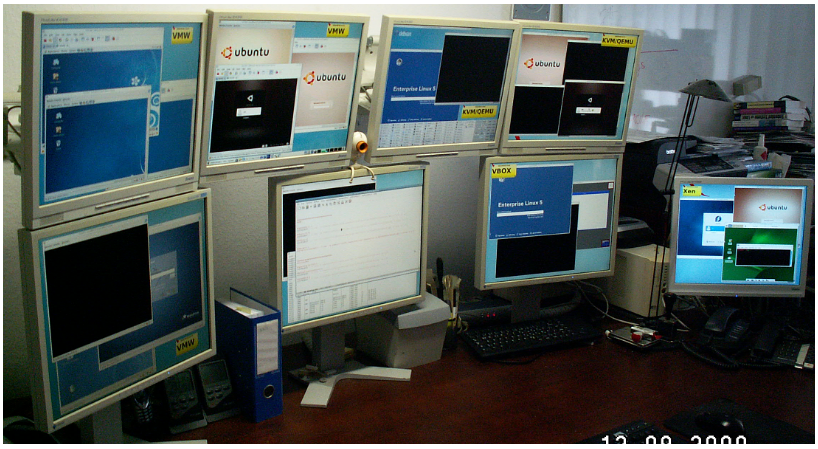


Figure 5.27: Gnome Xinerama

These local and remote pyhsical screens could be combined to a virtual screen by e.g. Xinerama (or distributed e.g. by Xrdb) what is supported by the UnifiedSessionsManager particularly through the extension of the **geometry** option of X11. This extension provided - beneath others - for an overlay of a virtual grid of screens on top of the combined displays, thus the physical screens could be addressed by user-defined labels, the required pixel-calculations are encapsulated by the mapping-functions of the UnifiedSessionsManager.
The following screenshot represents the logical view of the combined displays.

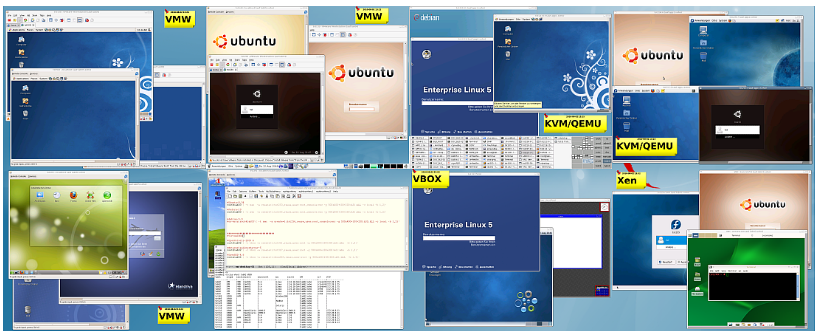


Figure 5.28: Gnome Xinerama

Another benefit from handling of logical addresses only is the independent addressing from physical changes, which occurs frequently when the hardware is exchanged. This is due to the automatic enumeration of components by the system, which varies widely by seemingly minor changes.

The mapping and address calculations are described in depth within the User-Manual of the DOC-Package(Remember the CCL-3.0 Licensing).

5.6.1 Physical and Logical Screens - A Sumup

Physical screens are by default supported for the X11 based desktops as independent terminal sessions. This particularly excludes the mouse movement between the screens by default.

One appropriate facility for the combination of pyhsical screens into on superpositined-screen as a combination is the so called Xinerama mode. This simply adds the pixel-arrays together and produces a resulting array of size by the overall sum of pixels. Therefore the new address range changes to the new screen size. In practical cases the order of the screens - meaning the actual physical position of a specific pixel-area - may change e.g. due to initialization or hardware-exchange. Particularly the hardware exchange, even the exchange of the slot position of a specific graphics card, may change the whole setup dramitically. Therefore the persistent storage of desktop scripts with geometry-positions should address logically without physical dependency.

5.6.2 Logical Display Addressing by the UnifiedSessionsManager

The applied scheme for logical addressing is - as the main philosophy - kept as simple as possible, therefore just relies on the standard means of the X11 configuration files. The specific approach is to utilize the X11 feature of user specific section/screen address labels for multiple display. These could be simply registered by editing the file

```
/etc/X11/xorg.conf
```

The custom labels are from then on valid screen aliasses for the specific array of pixels of the defined screen. Particularly in case of hardware changes the only thing to change is the label within the xorg.conf file, the bunch of user specific desktop configurations could remain unchanged.

An extract from an example file is given in the following figure. The Section **Server-Layout** is the logical overall screen, whereas the **Screen** sub-sections represent specific **pixel-arrays** mapped to a physical position. Thus by exchange of physical pixel arrays a re-positioning of physical devices could be adapated to previous definitions, and the legacy desktop configurations could be kept unchanged. In this particular case the screen names are defined in accordance to the 2-dimension array-like matrice-layout of the virtual screen. This eases the sddressing e.g. by '100x100+766+531:A21', which meand the positions are relative to the screen A21. instead of calculating '100x100+2146+1811', which may change completely for HW changes, without physical changes of the screen-position. This result from a logically double-level-remapping of pixels. For further details refer to the User-Manual(Remember the **CCL-3.0 License** with the full scope of **commercial restrictions**).

```
# nvidia-xconfig:
#    X configuration file generated by nvidia-xconfig
# nvidia-xconfig:
#    version 1.0  (buildmeister@builder58)  Tue Oct 20 21:25:04 PDT 2009

Section "ServerLayout"
    Identifier     "ALL"
```

```
    Screen      0  "A11" 1280 1024
    Screen      1  "A21" 2560 1024
    Screen      2  "A30" 3840 0
    Screen      3  "A01" 0 1024
    Screen      4  "A00" 0 0
    Screen      5  "A31" 3840 1024
    Screen      6  "A10" 1280 0
    Screen      7  "A20" 2560 0
    InputDevice    "Keyboard0" "CoreKeyboard"
    InputDevice    "Mouse0" "CorePointer"
    Option         "Xinerama" "1"
EndSection


Section "Files"
    FontPath        "unix/:7100"
EndSection

Section "Module"
    Load           "dbe"
    Load           "extmod"
    Load           "type1"
    Load           "freetype"
    Load           "glx"
EndSection

Section "ServerFlags"

# Removed Option "Xinerama" "1"
# Removed Option "Xinerama" "0"
    Option         "Xinerama" "1"
EndSection

Section "InputDevice"

    # generated from default
```

The following (independent) example depicts the resulting mapping required for actually unchanged virtual display positions in case of pyhsical re-ordering of pixels.

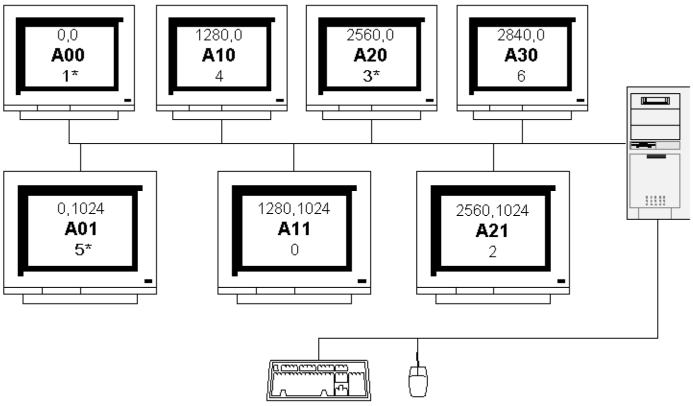


Figure 5.29: Physical Xinerama-Mapping

5.7 Examples

The following examples show some additional examplary cases of convenience integration into gnome by simple menus.

5.7.1 Demo-Desktop

The following desktop demonstrates the automated setup of a complex runtime environment. There are just some positioning restrictions due to limits of some proprietary client applications.

The required menu entry for starting is:

```
/home/userName/bin/ctys demo/vm-desktop-01
```

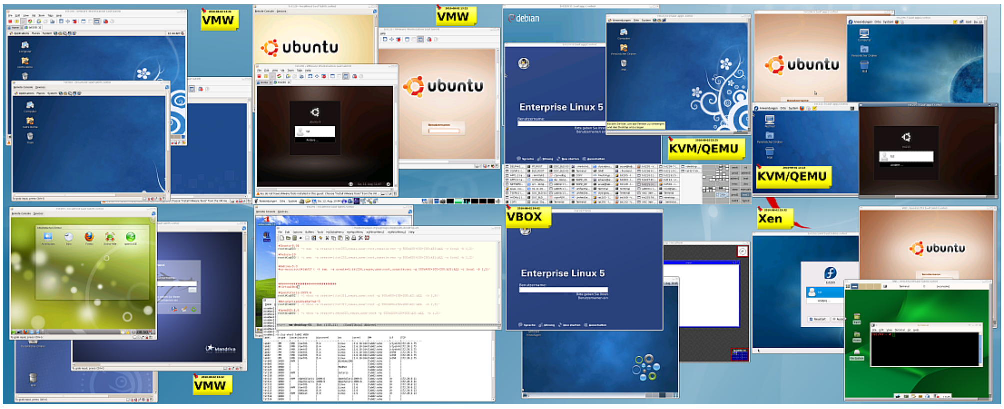
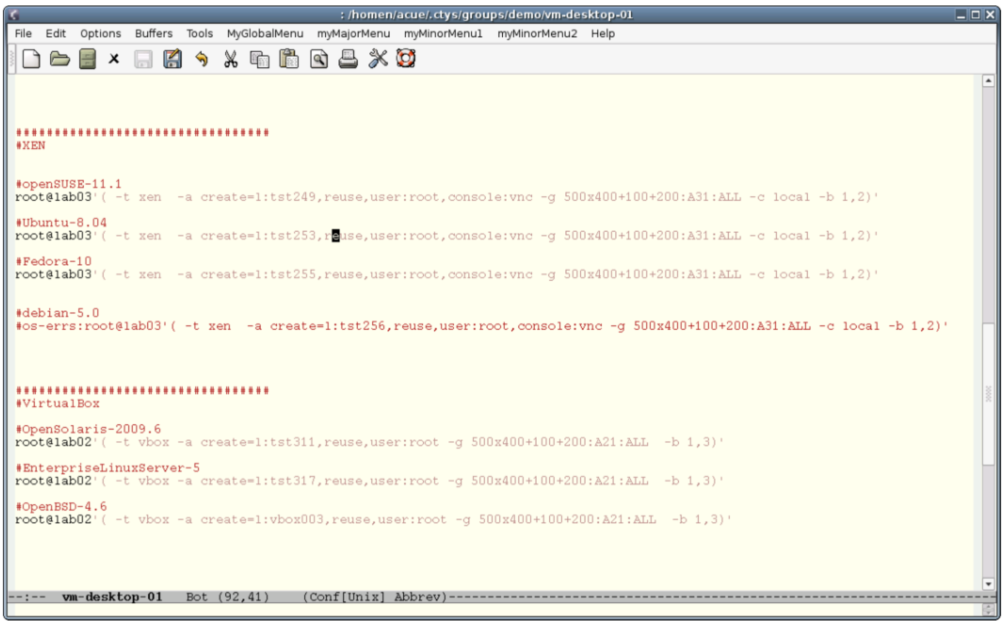


Figure 5.30: Gnome starter - Demo Desktop

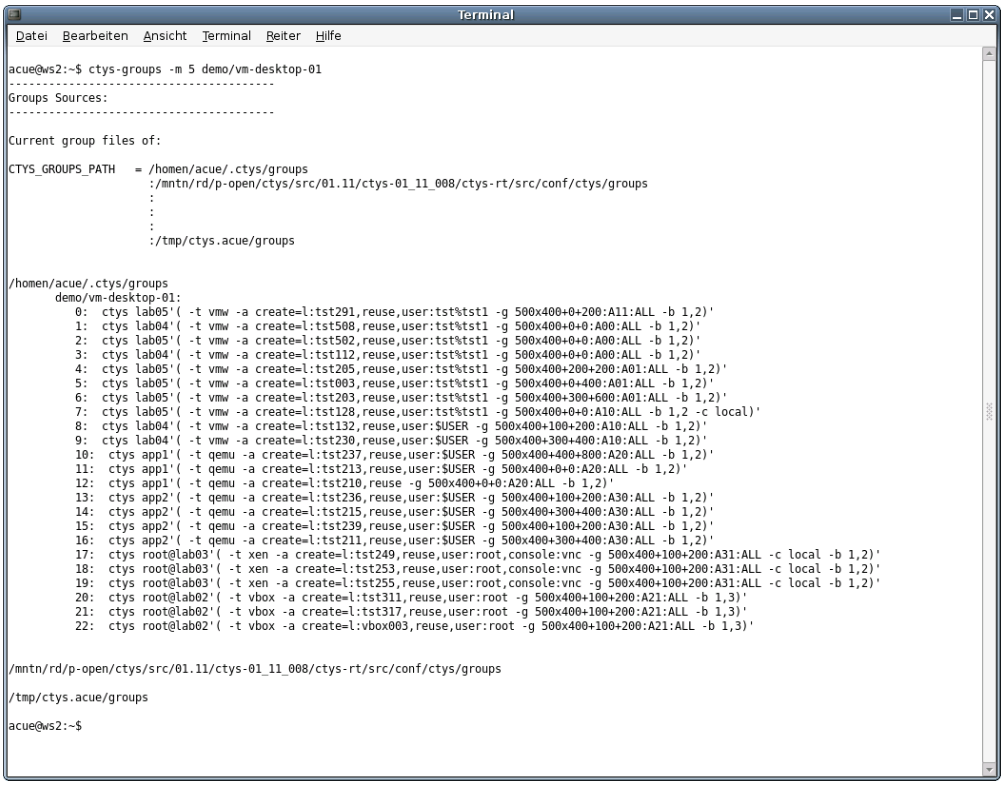
The following extract from the GROUPs file shows the configuration parts for XEN and VBOX.




```
##################################
#XEN

#openSUSE-11.1
root@lab03'( -t xen  -a create=l:tst249,reuse,user:root,console:vnc -g 500x400+100+200:A31:ALL -c local -b 1,2)'

#Ubuntu-8.04
root@lab03'( -t xen  -a create=l:tst253,reuse,user:root,console:vnc -g 500x400+100+200:A31:ALL -c local -b 1,2)'

#Fedora-10
root@lab03'( -t xen  -a create=l:tst255,reuse,user:root,console:vnc -g 500x400+100+200:A31:ALL -c local -b 1,2)'


#debian-5.0
#os-errs:root@lab03'( -t xen  -a create=l:tst256,reuse,user:root,console:vnc -g 500x400+100+200:A31:ALL -c local -b 1,2)'



##################################
#VirtualBox

#OpenSolaris-2009.6
root@lab02'( -t vbox -a create=l:tst311,reuse,user:root -g 500x400+100+200:A21:ALL  -b 1,3)'

#EnterpriseLinuxServer-5
root@lab02'( -t vbox -a create=l:tst317,reuse,user:root -g 500x400+100+200:A21:ALL  -b 1,3)'

#OpenBSD-4.6
root@lab02'( -t vbox -a create=l:vbox003,reuse,user:root -g 500x400+100+200:A21:ALL  -b 1,3)'
```


Figure 5.31: Gnome starter - Extract from GROUP

The ctys-groups(1) command could be used to resolv the entries within the GROUP for cut-and-paste operations.




```
acue@ws2:~$ ctys-groups -m 5 demo/vm-desktop-01
---------------------------------------
Groups Sources:
---------------------------------------

Current group files of:

CTYS_GROUPS_PATH   = /homen/acue/.ctys/groups
                     :/mntn/rd/p-open/ctys/src/01.11/ctys-01_11_008/ctys-rt/src/conf/ctys/groups
                     :
                     :
                     :
                     :/tmp/ctys.acue/groups


/homen/acue/.ctys/groups
       demo/vm-desktop-01:
          0:  ctys lab05'( -t vmw -a create=l:tst291,reuse,user:tst%tst1 -g 500x400+0+200:A11:ALL -b 1,2)'
          1:  ctys lab04'( -t vmw -a create=l:tst508,reuse,user:tst%tst1 -g 500x400+0+0:A00:ALL -b 1,2)'
          2:  ctys lab05'( -t vmw -a create=l:tst502,reuse,user:tst%tst1 -g 500x400+0+0:A00:ALL -b 1,2)'
          3:  ctys lab04'( -t vmw -a create=l:tst112,reuse,user:tst%tst1 -g 500x400+0+0:A00:ALL -b 1,2)'
          4:  ctys lab05'( -t vmw -a create=l:tst205,reuse,user:tst%tst1 -g 500x400+200+200:A01:ALL -b 1,2)'
          5:  ctys lab05'( -t vmw -a create=l:tst003,reuse,user:tst%tst1 -g 500x400+0+400:A01:ALL -b 1,2)'
          6:  ctys lab05'( -t vmw -a create=l:tst203,reuse,user:tst%tst1 -g 500x400+300+600:A01:ALL -b 1,2)'
          7:  ctys lab05'( -t vmw -a create=l:tst128,reuse,user:tst%tst1 -g 500x400+0+0:A10:ALL -b 1,2 -c local)'
          8:  ctys lab04'( -t vmw -a create=l:tst132,reuse,user:$USER -g 500x400+100+200:A10:ALL -b 1,2)'
          9:  ctys lab04'( -t vmw -a create=l:tst230,reuse,user:$USER -g 500x400+300+400:A10:ALL -b 1,2)'
          10:  ctys app1'( -t qemu -a create=l:tst237,reuse,user:$USER -g 500x400+400+800:A20:ALL -b 1,2)'
          11:  ctys app1'( -t qemu -a create=l:tst213,reuse,user:$USER -g 500x400+0+0:A20:ALL -b 1,2)'
          12:  ctys app1'( -t qemu -a create=l:tst210,reuse -g 500x400+0+0:A20:ALL -b 1,2)'
          13:  ctys app2'( -t qemu -a create=l:tst236,reuse,user:$USER -g 500x400+100+200:A30:ALL -b 1,2)'
          14:  ctys app2'( -t qemu -a create=l:tst215,reuse,user:$USER -g 500x400+300+400:A30:ALL -b 1,2)'
          15:  ctys app2'( -t qemu -a create=l:tst239,reuse,user:$USER -g 500x400+100+200:A30:ALL -b 1,2)'
          16:  ctys app2'( -t qemu -a create=l:tst211,reuse,user:$USER -g 500x400+300+400:A30:ALL -b 1,2)'
          17:  ctys root@lab03'( -t xen -a create=l:tst249,reuse,user:root,console:vnc -g 500x400+100+200:A31:ALL -c local -b 1,2)'
          18:  ctys root@lab03'( -t xen -a create=l:tst253,reuse,user:root,console:vnc -g 500x400+100+200:A31:ALL -c local -b 1,2)'
          19:  ctys root@lab03'( -t xen -a create=l:tst255,reuse,user:root,console:vnc -g 500x400+100+200:A31:ALL -c local -b 1,2)'
          20:  ctys root@lab02'( -t vbox -a create=l:tst311,reuse,user:root -g 500x400+100+200:A21:ALL -b 1,3)'
          21:  ctys root@lab02'( -t vbox -a create=l:tst317,reuse,user:root -g 500x400+100+200:A21:ALL -b 1,3)'
          22:  ctys root@lab02'( -t vbox -a create=l:vbox003,reuse,user:root -g 500x400+100+200:A21:ALL -b 1,3)'


/mntn/rd/p-open/ctys/src/01.11/ctys-01_11_008/ctys-rt/src/conf/ctys/groups

/tmp/ctys.acue/groups

acue@ws2:~$
```

Figure 5.32: Gnome starter - Display of resolved GROUP

5.7.2 Single Machine Entry

This example shows an entry for a single VM. The actual menu entry is written within one line:

```
/homen/acue/bin/ctys
   delphi'(
    -t vmw
    -a create=reuse,l:office001,b:/mntn/vmpool/vmpool03/vmw/office,user:acue
    -g 1268x994:A11:ALL
    -b 1
    -c local
   )'
```

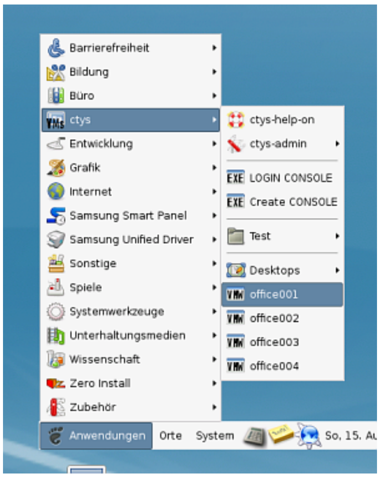


Figure 5.33: Gnome starter single entry - Menu

This menu entry starts a VM for example here on a VMware Server-2(TM). It has to be recognised that the whole command line is visible in clear text by 'ps' command on the local machine, and at least partially on the target machine. The intermediate connection is encrypted by OpenSSH. Additonally menu entries are stored within files, which must not contain any serious passwords at all. Thus for security reasons in this case the username is provided only, the password is entered interactively.

Figure 5.34: Gnome starter LOGIN - VMWRC Login

The started Windows2000(TM) desktop in this case could be seen as 'second level menu entry' for virtual applications.

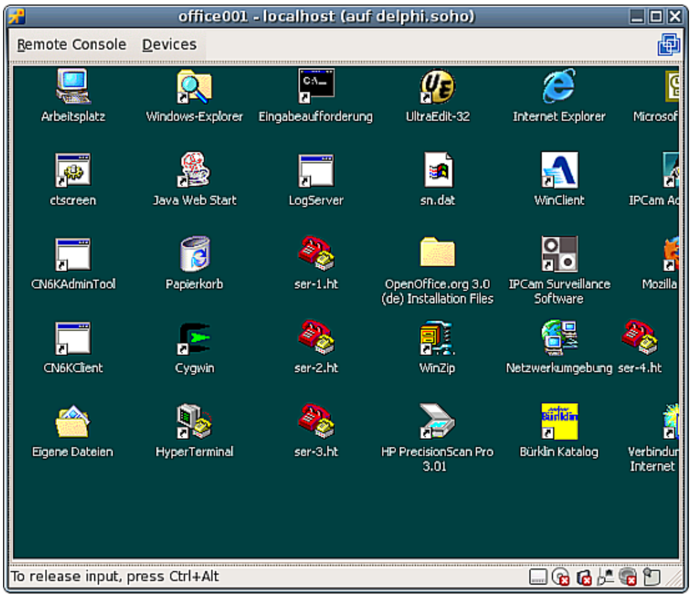


Figure 5.35: Gnome starter LOGIN - Started W2K desktop

5.7.3 PABX with VLAN-Gateway

This example shows an entry for a the maintenance of two PABXs, where one is an Asterisk PABX on a distinct VoIP VLAN.

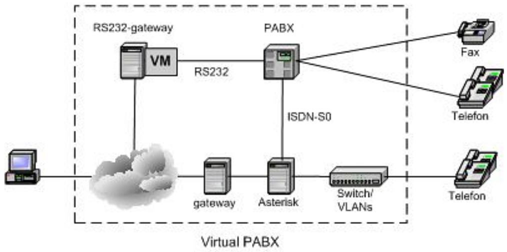


Figure 5.36: Gnome starter PABX - Interconnection Structure

The interconnection is setup by an intermediate gateway, which is passed by the utility 'ctys-beamer'.

```
ctys-beamer -Y --x11 -R root@tserv00 -b async --beam-this \
  ctys -Y -a create=l:PABX2,reuse root@192.168.50.1
```

The resulting starter script is

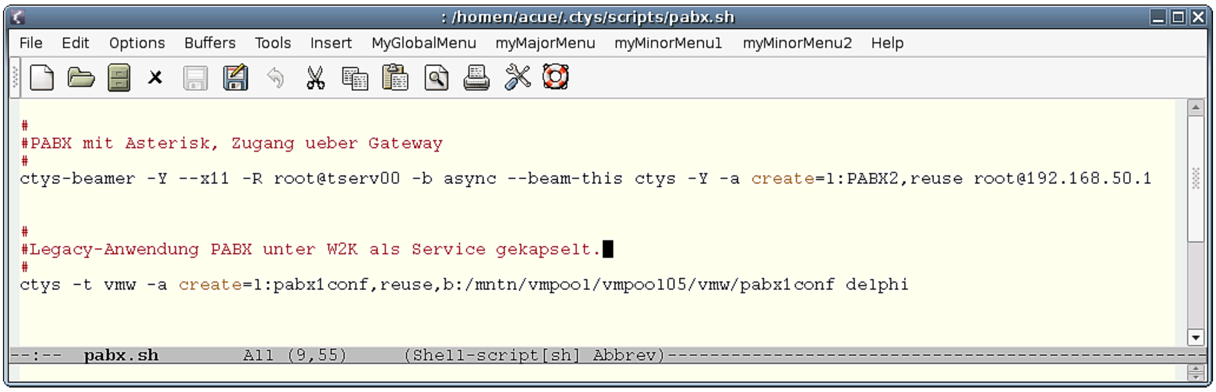


Figure 5.37: Gnome starter PABX - Script

The actual menu entry is:

```
pabx.sh
```

which could be started as:

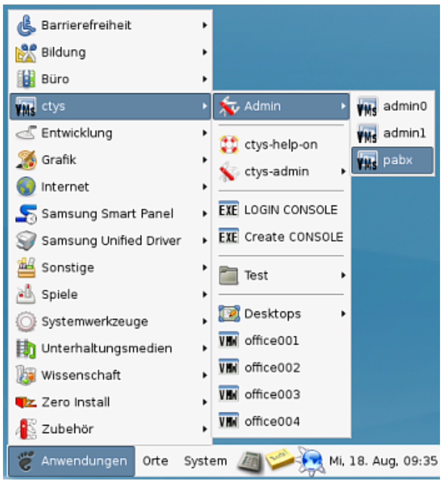


Figure 5.38: Gnome starter PABX - menu

This opens the two views, on the left the VM containing the legacy configuration and monitoring utily from the '90s' running on the local segment interconnected by an RS232 on one gateway. The right view shows the Asterisk master PABX which is running on the VoIP gateway within another VLAN, thus has to be interconnected by a TCP/IP gateway.

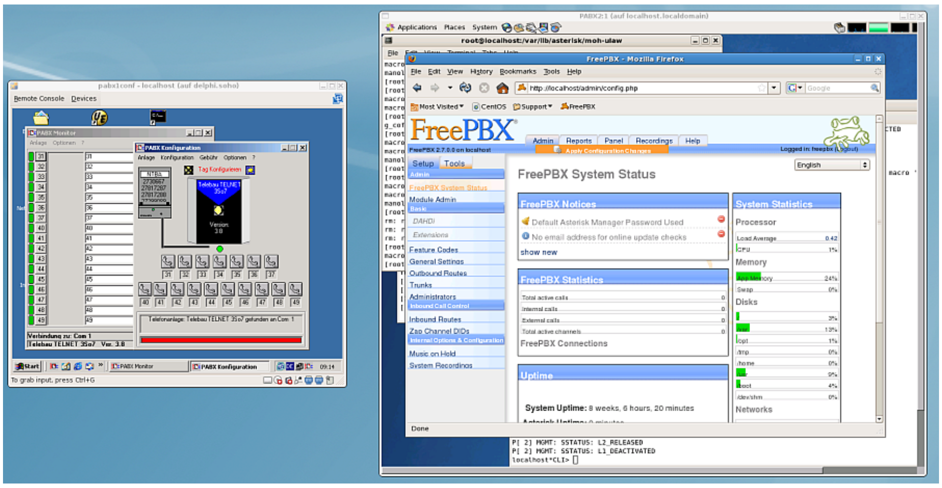


Figure 5.39: Gnome starter PABX - Asterisk

REMARK:
Due to some limits by OpenSSH for handling of specific interfaces on the targeted TCP/IP gateway, evantually some static routing for specific hosts on different VLANs is required.

Chapter 6

VMWE - ESX(TM) Setup

.

6.1 ESX - Basics for Operations

The current version supports as a first step the HOSTs plugins for administration and maintenance operations. The CLI plugin could be used in full and stub mode, whereas the X11 plugin could be used in stub mode with the standard installation. The remaining plugins require some additional software installations.

6.2 Supported HOST-OSs

The basic installation of ESX supports only the CLI plugin, thus some additional tools has to be installled.

ffs.

6.3 Supported GuestOSs

Any OS as supported by the hypervisor.

6.4 Supported Architectures

The whole set of available CPUs by Products is supported :

`x86, AMD64, x86_64`

6.5 Supported Interfaces

ffs.

6.6 Supported VM Management Interfaces

ffs.

6.7 Network Interconnection

ffs.

6.8 Installation of Components

6.8.1 ESX-4.1.0

ffs.

6.8.2 SSH-Access

ffs.

6.9 Install Procedures

ffs.

6.9.1 Supported/Tested Install-Mechanisms

The current version relies on the provided intstall mechanisms of the product supplier, and pre-requires an installed system.

6.10 Installation of GuestOS

ffs.

6.10.1 Installed Systems

OS	**name**	**Media**
CentOS-5.0	...	PXE,ISO

Table 6.1: Overview of Intsalled-VMW-VMs

Chapter 7

VMWE - ESXi(TM) Setup

.

7.1 ESXi - Basics for Operations

ffs.

7.2 Supported HOST-OSs

ffs.

7.3 Supported GuestOSs

ffs.

7.4 Supported Architectures

The whole set of available CPUs by Products is supported :

`x86, AMD64, x86_64`

7.5 Supported Interfaces

ffs.

7.6 Supported VM Management Interfaces

ffs.

7.7 Network Interconnection

ffs.

7.8 Installation of Components

7.8.1 ESXi-4.0.0

ffs.

7.8.2 SSH-Access

ffs.

7.9 Install Procedures

ffs.

7.9.1 Supported/Tested Install-Mechanisms

The current version relies on the provided intstall mechanisms of the product supplier, and pre-requires an installed system.

7.10 Installation of GuestOS

ffs.

7.10.1 Installed Systems

OS	**name**	**Media**
CentOS-5.0	...	PXE,ISO

Table 7.1: Overview of Intsalled-VMW-VMs

Chapter 8

QEMU/KVM Setup

.

8.1 QEMU/KVM - Basics for Operations

The ctys-QEMU plugin supports the emulation of various CPUs by QEMU as well as it's accelerator modules e.g. KVM and KQEMU(under development). The KVM accelerator of the Linux kernel is handled as a specific accelerator thus supported by the QEMU plugin.

The ctys-QEMU plugin of the UnifiedSessionsManager supports a subset of the QEMU command line options mapped to native options, whereas remaining options are just bypassed. Therefore a meta-layer for an abstract interface is defined, which is implemented by a wrapper script. The wrapper script is written in bash syntax and sourced into the runtime process, but could be used for native command line calls as well.

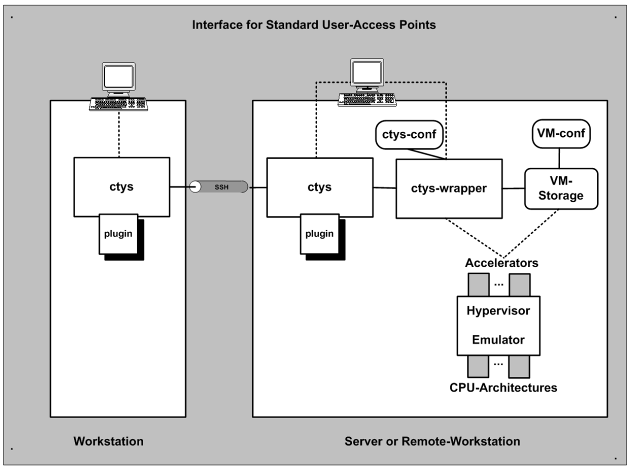


Figure 8.1: ctys distributed access

The main advance of using a wrapper script is the ability to perform dynamic scripting within the configuration file, which is standard bash-syntax with a few conventions. Templates for configuration files are supported within the **.ctys/ctys-createCofVM.d** directory. The whole set of the UnifiedSessionsManager framework is available within the wrapper scripts.

An installer for complete setup of a QEMU and/or KVM based VM is contained. The tool **ctys-createConfVM(1)** creates either interactively, or in batch-mode a local or remote configuration by detection of the actual platform and creation of a ready-to-use startup configuration. This configuration comprises a generic wrapper script and a specific configuration file. The installation of a GuestOS could be performed either by calling the wrapper-script or by calling ctys with the **BOOTMODE** set to **INSTALL** for ISO image boot, or to **PXE** for network based boot of the install medium. Once QEMU/KVM is setup, the boot of the VM could be performed from the virtual HDD.

Basic Use-Cases for application are contained within the document **ctys-uc-QEMU(7)**
.

.

8.2 Supported HOST-OSs

The QEMU plugin is supported an all released runtime environments of the UnifiedSessionsManager.

8.3 Supported GuestOSs

The native GuestOS support is the same as for the PMs and HOSTs plugins.

8.4 Supported Architectures

The whole set of QEMU's CPUs is supported, which includes for version 0.9.1:

`x86, AMD64, ARM, MIPS, PPC, PPC64, SH4, M68K, ALPHA, SPARK`

The call has to be configured within the configuration file. Ready-to-use templates for the provided QEMU tests are included for x86, Arm, Coldfire, and SPARC - Running Linux, uCLinux, and NetBSD.

8.5 Supported Interfaces

8.5.1 Overview

Qemu supports various interfaces for interconnection of it's hosted GuestOS to an external devices. Particularly the applicable interfaces for **CONSOLE** and **QEMUMONITOR** interconnection are of interest for the QEMU plugin as a hypervisor controller, whereas the support for native interfaces is handled by the HOSTs plugings.

The encapsulation of the interfaces for access from the **outside-HostOS** to the **inside-GuestOSs** is encaspsulated by the QEMU-VM via usage of specific virtualisation drivers. These drivers actually manipulate the payload-dataflow and are commonly interconnected to native operational peers of the GuestOS such as the LAN interfaces. The outer encapsulation by the UnifiedSessionsManager is a control only encapsulation and interconnects just the few interfaces required for the control of the hypervisor as well as the user interfaces.

Some addional tools are provided as helpers for configuration and management of HostOS operatinal interfaces. One example is here the **ctys-vnetctl(1)** script for the interconnection of the virtual QEMU network interface stubs to their operational HostOS peers.

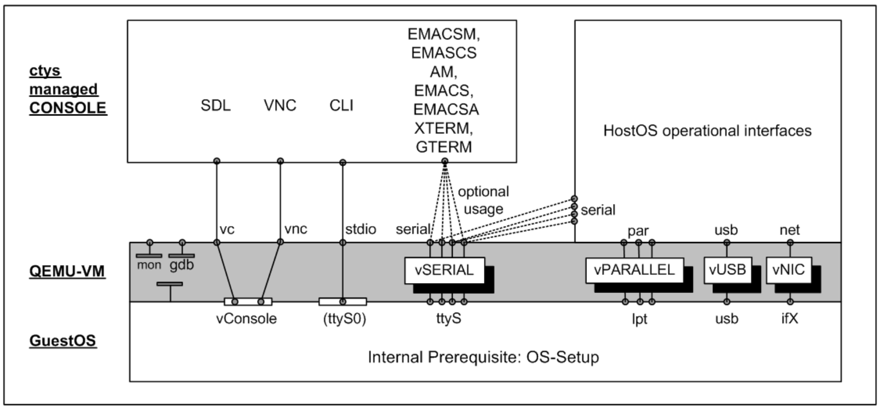


Figure 8.2: Supported Management CONSOLEs

In the previous layered interface depiction the serial interfaces could be optionally interconnected by CONSOLE entities as well as be used for HostOS devices.

The structure of the encapsulation and the supported components are depicted within the following figure.

The outer encapsulation by the UnifiedSessionsManager is divided into two parts. The first part is the custom wrapper-script for final execution of the QEMU VM by calling the VDE-wrapper. The ctys-wrapper script itself can represent a complex control flow but is mananaged as one entity only, thus not more than one VM instance should be implemented within the wrapper script. The second part of the interface is the call interface for specific CONSOLE types, which prepares additional execution environments for the various call contexts. It should be obvious that the two outer encapsulation

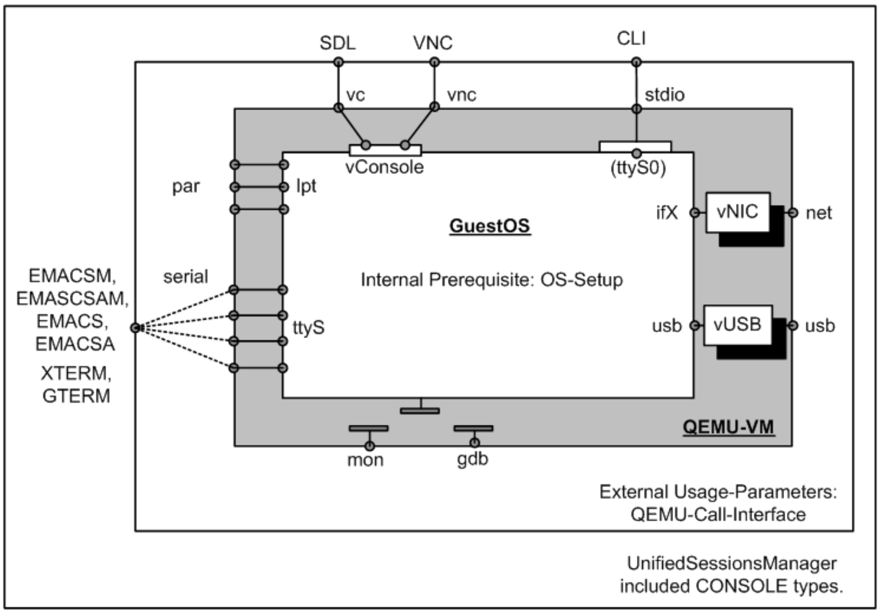


Figure 8.3: Supported QEMU Management Interconnection-Interfaces

components are required to cooperate seamless.

For the actual and final interconnection to the GuestOS there a two basic styles of CONSOLE types:

1. **Transparent standard IO-Devices**
   These are implemented by the virtual device drivers for keyboard, mouse, and display. The standard drivers of the GuestOS handles these transparently as standard user interfaces.

   Transparent IO-Devices are SDL and VNC based on a transient virtual HW, thus almost need no specific configuration for standard devices, but the activation for the QEMU VM.
2. **GuestOS custom IO-Devices**
   These are optional configurations for GuestOSs such as a serial console within Linux as a GuestOS. In case of Linux for example the user has to prepare the usage by a kernel parameter for boot time access and preset a tty-console-device in "/etc/inittab".

QEMU supports by default up to 4 serial devices. Within the UnifiedSessionsManager, one port is forseen for CLI , VNC , and SDL mode, two serial ports are foreseen for the remaining modes to be used by the framework. For the CLI console no extra monitoring port is allocated, the default values for **-nongraphic**, which are stdbuild/stdin with a multiplexed monitoring port, are used.

One serial device is reserved for an additional monitor port exclusively for the types SDL and VNC . For the remaining CONSOLE types, which are variants of CLI type, the monitoring port is multiplexed to the console port again, but now for an allocated common UNIX-Domain socket. This port is required in order to open a management interface. When suppressed some actions like CANCEL may not work properly. This is for example the case, for the final close of the stopped QEMU VM, which requires frequently a monitor action.

```
-serial mon:unix:\${MYQEMUMONSOCK},server,nowait
```

8.5.2 Bluetooth

ffs.

8.5.3 CDROM/DVD

The default wrapper-script contains one HDD as hda device for the **BOOTMODE:VHDD** and additionally one DVD/CDROM for the **BOOTMODE:INSTALL**. These could be extended as required.

For dynamic non-stop-configuration of a DVD/CDROM the following procedure has to be applied within the QEMUmonitor.

1. info block
2. eject <device>
3. change <device> <path-to-iso-file/path-to-dev-cdrom>

8.5.4 FDD

ffs.

8.5.5 HDD

ffs.

8.5.6 Network

The prefered network devices are based on the virtual switch provided by the VirtualSquare-VDE project. These are attached to TAP devices with root permissions and require from than on only user permissins for attaching VMs to the virtual switch. Even though any provided network connection could be utilized within the wrapper script, the current toolset supports the VDE utilities only.

The VDE project provides a wrapper for the qemu call, which replaces the qemu call by **vdeqemu**. The parameters "-net nic,macaddr=$MAC0" and "-net vde,sock=$QEMUSOCK" are used within the standard wrapper scripts.

For additional information refer to the chapter "Network Interconnection".

8.5.7 Parallel Ports

ffs.

8.5.8 SDL

SDL is the probably intended "standard" device, but has in some versions the drawback of cancelling the VM when the window is closed. Within ctys the safely detachable VNC connection is the preferred console. When for analysis of the boot process the BIOS output is required the CLI console could be applied.

8.5.9 Serial Ports

The setup of a serial console for QEMU is required for various CONSOLE types. Any CONSOLE providing an ASCII-Interface, except the syncronous un-detachable CLI console, requires serial access to the GuestOS. This is a little complicated to setup for the first time, but once performed successful, it becomes an easy task for frequent use.

The first thing to consider is the two step setup, which comprises the initial installation with a standard interface either by usage of SDL or by usage of the VNC console. The second step - after finishing the first successesfully - is to setup the required serial device within the GuestOS. This requires a native login as root. Detailed information is for example available at "Linux Serial Console HOWTO" [139, VANEMERY], "Serial-HOWTO" [140, GHAWKINS], and "Text-Terminal-HOWTO" [141, DSLAWYER].

The following steps are to be applied.

1. Install GuestOS by usage of SDL/ VNC as console.
2. Login into the GuestOS.
3. Adapt **/boot/grub/menu.lst**
   The header section:

   ```
   serial --unit=0 --speed=9600 --word=8 --parity=no --stop=1
   terminal --dumb serial console
   splashimage=(hd0,0)/grub/splash.xpm.gz

   default=<\#yourKernelWithConsole>
   ```

   Your target kernel for boot <yourKernelWithConsole>:

   ```
   kernel ... console=tty0 console=ttyS0,9600n8
   ```

4. Adapt **/etc/inittab**, here for CentOS-5.0

   ```
   S0:12345:respawn:/sbin/agetty ttyS0 9600 Linux
   ```

5. Adapt ctys-qemu-wrapper This requires adapted items, which depend on the choosen CONSOLE type. - A CLI , which is a synchronous console, requires:

   ```
   -nographic
   ```

   This switches off the graphic and defaults it's IO to the caller's CLI .
6. A "by-Window-Encapsulated-CLI", which is a non-"modal" console, called as a serial console by a UNIX-Domain socket within a X11-Window/Client, requires:
   ...
   This switches off the graphic and defaults it's IO to the caller's CLI .
7. Reboot.

8.5.10 STDIO

This attaches the console to the callers shell. Requires preconfiguration of a serial device within the GuestOS, for a template refer to setup of serial console.

8.5.11 USB

ffs.

This console replaces the SDL type when choosen. It works as a virtual Keyboard-Video-Mouse console by default and thus does not require pre-configuration of the GuestOS. But needs to be explicitly activated by the **-vnc** option.

8.6 Supported VM Management Interfaces - QEMUmonitor

The QEMU monitor port is supported as a local UNIX-Domain socket only. The socket name is assembled by a predefined environment variable and the PID of the master process for the final wrapper script , which is executing the QEMU VM and has to be configured by the user. For the various CONSOLE types different handling of the monitor port is applied:

- CLI0: No specific port, stdio is used for console as well as for monitor.
- SDL, VNC : Extra monitor port QEMUMONSOCK, used as multiplexed port, could be connected additionally by an ASC-II console.

CLI , XTERM, GTERM, EMACS, EMACSM, EMACSA, EMACSAM: Mapped monitor port in multiplex mode on UNIX-Domain socket QEMUMONSOCK for re-attacheable console port.

The base variable is "QEMUMONSOCK" , which contains by convention the substrings **ACTUALLABEL** and **ACTUALPID**. These two substrings will be replaced by their actual values evaluated when valid during runtime. The **ACTUALLABEL** is the label of the current VM, as will be provided to the commandline option **-name** of QEMU. The **ACTUALPID** is the master pid of the wrapper script, which will be evaluated by the internal utility "ctys-getMasterPid" . The master pid is displayed as the SPORT value, even though it is used as a part of the actual UNIX domain socket only.

The default socket-path is:

```
/var/tmp/qemumon.<ACTUALLABEL>.<ACTUALPID>.$USER
```

This will be replaced e.g. to:

```
/var/tmp/qemumon.arm-test.4711.tstuser1
```

Any terminal application like **unixterm** of VDE package, or **netcat/nc** could be used for interaction. The switch between QEMU monitor and a text console is the same as for the **-nographic** mode by **Ctrl-a-c**, for additional information refer to the QEMU user-manual. The monitor socket is utilized by internal management calls like CANCEL action by usage of **netcat/nc**.

REMARK: When terminating a CLI session, the prompt will be released by a monitor short-cut: **Ctrl-a x**. In some cases a **Ctrl-c** is sufficient.

The following controls are used for monitor:

Ctrl-a	**001**
x	170
c	143
S3	stop/cont
S4	savevm/loadvm[tagid]
S5	Ctrl-ax

Utilized QEMU-Monitor-Commands

The switch over between the guest console and the monitor console from within a **VNCviewer** client is performed by **Ctrl-Alt-(1|2)**. Where **Ctrl-Alt-2** switches to the Monitor, and **Ctrl-Alt-1** back to the GuestOS-Console. When nested VNCviewers are called the VNCviewer-Menu by default opended with **F8** could be used to mask either the **Ctrl** or the **Alt** key.

8.7 Network Interconnection

8.7.1 Overview

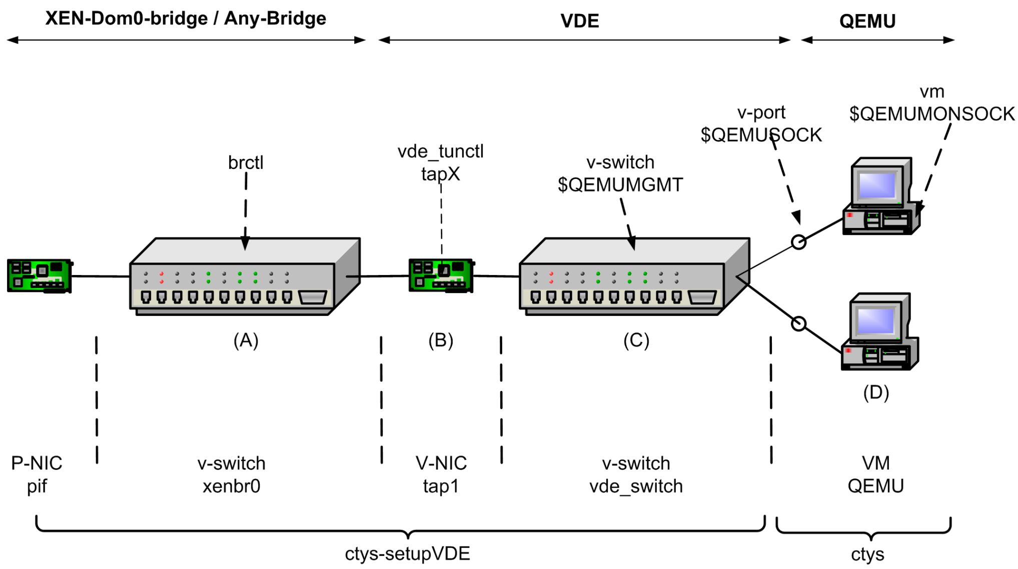


Figure 8.4: QEMU NIC interconnection

The QEMU plugin utilizes the VDE package exclusively for setting up network connections. The verified version is vde2 which is for the current version of ctys-QEMU a mandatory prerequisite. This is due to the following two features mainly:

- Availability of tunctl, which is named **vde_tunctl** within vde. A TAP device is mandatory for QEMU to be interconnected in a transparent bridging mode as applied here.
- The User-Space virtual switch **vde_switch** requires one tap-device to attach itself to the **external** network, whereas the users can attach to it's internal ports with their own permissions. This is permitted due to pre-assignment of it's owner by usage of **vde_tunctl** .

A short description of the install process for QEMU with network support, pxe-boot/install, and cdrom-boot/support based on the examples from QEMU is given in the HowTo of ctys. Downloads are available from [132, VDE2], and a very good decscription about networking with TAP could be read at [133, VirtualSquare].

Once the basic install and setup is completed, the whole process for the creation of the required networking environment is handled for local and remote setups by one call of the **ctys-vnetctl(1)** only .

The listed environment variables are to be used within the configuration scripts. These are particularly mandatory for to be present and accessible by usage of ctys. So the CANCEL action for example will open a connection to the QEMUMONSOCK and sends some monitor commands . The QEMUMGMT variable will be used to evaluate the related tap-device, and for final deletion of the swithc, when no more clients are present. All sockets are foreseen to be within UNIX domain only, as designed into the overall security principle. Anyhow some minor break might occur for the vnc port for now, and should be blocked by aditional firewall rules for remote access.

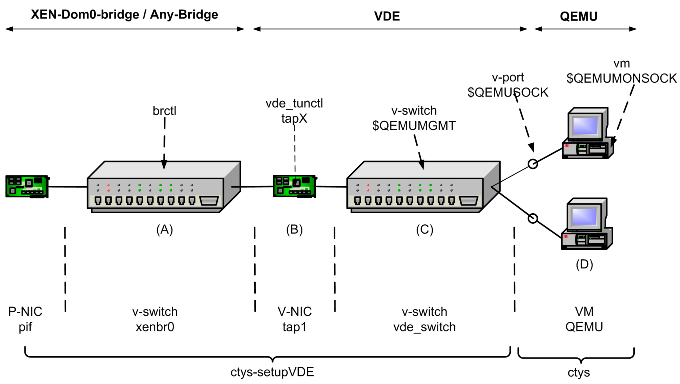


Figure 8.5: QEMU interconnection

The following variables are required partly to be modified with dynamic runtime data such as the actual USER id and the PID as shown in the examples. The definition and initialization is set in the central plugin-configuration file **qemu.conf**.

- QEMUMONSOCK The monitoring socket within local UNIX domain, will be used as

  ```
  -serial mon:unix\${QEMUMONSOCK},server,nowait
  ```

  Could be attached by the terminal emulations. It should be the first entry containing the serial console "ttyS0" too, which is for the provided ctys-examples assumed, and is the case.

  - nc -U $QEMUMONSOCK
  - unixterm $QEMUMONSOCK

- QEMUSOCK The socket to be attached to the vde_switch. One specific port, therefore specific QEMUSOCK is derived for each running QEMU instance.
  The network socket within local UNIX domain, will be used as

  ```
  -net vde,sock=\${QEMUSOCK}.
  ```

- QEMUMGMT The socket for management of the virtual switch. This could be also accessed by nc/netcat and unixterm. The utility **ctys-vnetctl(1)** widely utilizes this interface.

  - nc -U $QEMUMGMT or netcat -U $QEMUMGMT
  - unixterm $QEMUMGMT

8.7.2 Setup Networking for QEMU by VDE

Once QEMU and VDE2 are istalled successfully, either by delivered packages or by compilation and the "make install" call, the base package of QEMU is installed. The next step now is to create a runtime environment.

The current version therefore supports particularly the tools **ctys-vnetctl(1)** and **ctys-createConfVM(1)**

- **ctys-createConfVM(1)**
  handles the whole process of required preparation for the final GuestOS installation. Therefore several configuration and wrapper scripts are created. These comprise particularly ready-to-use installer executables in case of debian-5.x or CentOS-5.x as GuestOS.
- **ctys-vnetctl(1)**
  handles the complete process of creation and interconnection of a virtual switch by just one call.

The QEMU project offers some ready-to-use images, which could be used instead of the creation of a new VM.

- arm-test-0.2.tgz
- coldfire-test-0.1.tar.bz2
- linux-0.2.img.bz2
- mipsel-test-0.2.tgz
- mips-test-0.2.tgz
- small.ffs.bz2
- sparc-test-0.2.tgz

The only configuration required later is the setting of appropriate IP address, except for the coldfire image, which is based on DHCP. This is only true if you are using DHCP and have set the appropriate MAC addresses.

REMARK: The usage of bridged network with communications via the NIC of the host requires some additional effort. Particularly the creation of the required TAN-device with the frequently mentioned **tunctl** utility from the **UserModeMLInux** was somewhat difficult on CentOS-5.0. The package **vde** including a (not-found-documentation-for) utility **vde_tunctl** was the rescue-belt. STarting with the first version this is completely encapsulated by the utility **ctys-vnetctl(1)**

The resulting call to setup a compelte networking environment is

```
ctys-vnetctl -u <userName> create
```

Some deviation may occur in case of multiple interfaces, where the first is not active. In such cases it is sufficient to provide the option '-i' for selection of a specific interface.

At this point anything might be prepared for successful operations and the installation of a GuestOS could be performed as described within the following chapters.

For Qemu several excellent sites with install descriptions exist, thus here are just some shortcuts, which seem to be the most important items.

- **PXE-Boot**
  The following applies to a configured PXE environment based on PXELinux, check this with the call option '-d pf'.

```
vdeqemu -vnc :17 -k de -m 512
        -hda linux-0.2.img
        -net nic,macaddr=<mac>
        -net vde,sock={QEMUSOCK}
        -boot n
        -option-rom \${QEMUBIOS}/pxe-ne2k\_pci.bin \&
```

- **ISO-Image-Boot**
  The following applies to a boot CDROM image for installation, check this with the call option '-d pf'.

```
vdeqemu -vnc :17 -k de -m 512
        -hda linux-0.2.img
        -net nic,macaddr=<mac>
        -net vde,sock={QEMUSOCK}
        -boot d
        -cdrom \${QEMUBASE}/iso/install.iso \&
```

- **shared memory**
  Should be set in **/etc/fstab** as required. This value might be raised, when nested stacks are used, thus the bottom engine requires the sum of the resources of the stack. The entry might look like:

```
none /dev/shm tmpfs defaults,size=512M 0 0
```

  For a call like:

```
vdeqemu -m 512 ...
```

  The changes could be activated with

```
mount -o remount /dev/shm
```

- **Create image**
  To create an ISO image a call like the following could be applied

```
qemu-img create -f qcow myImage.qcow 4G
```

  which could be used as

```
qemu -cdrom installMedia.iso \
  -boot d myImage.qcow
```

8.7.3 Setup of the Network Interface for QEMU

The network interfaces are defined as default values within the templates for the configuration files. These could be customized as required. The default type is 'rtl8139' which may work in almost any case. Additional values are included as commented entries, thus could be easily exchanged.

For newer kernels the **virtio** driver should be used, but the kernel module requires to be pre-loaded. Therefore some pre-configuration of Linux is required.

8.7.4 PXE-Boot

The PXE based installation is possibly not the fastest, but it offers a common seamless solution for unified installation processes. Even though an image could be just copied and modified as required, some custom install procedures might be appreciated, when the install could be performed in batch-mode. One example is the usage of kickstart files for CentOS/RHEL. In case of PXE these files are almost the same for any install base, this spans from physical to virtual machines.

8.7.5 Install on USB-Sticks

FreeDOS - Balder for BIOS-Updates

The installation of FreeDOS on a bootable USB-Stick e.g. for BIOS updates requires the following steps.

1. Create configurationfiles and wrapper script by calling **ctys-createConfVM(1)** .
2. Executes first stage of installation including formatting of required boot device by calling

```
ctys -t qemu -a create=l:myLabel,instmode:FDD\%none\%USB\%/dev/sdg\%init localhost
```

   This call requires the configuration of the path to the FDD image within the configuration file, thus 'none' is provided within the call.

The installation requires the following actions by the user within the installed system.

1. Call of **fdisk** in order to partition the target device for installation.
2. Reboot.
3. Execute second stage of installation including the format of device by calling.

```
ctys -t qemu -a create=l:myLabel,instmode:FDD\%none\%USB\%/dev/sdg\%none localhost
```

   This call requires the configuration of the path to the FDD too and omits the initial formatting by setting the stage to none.

   The following actions by the user are quired.

4. Call of **format c: /S** in order to format the target device.
5. Call of **xcopy /E /N a: c:** for copy of executables.
6. Adaption of autoexec.bat.

That's it.

Linux - CentOS

The installation of Linux is even easier. Just call the installmode and assign the path to the USB device as inst-target. The installer reqcognizes the USB device and handles the partitioning.

.

8.8 Installation of Components

8.8.1 Install UnifiedSessionsManager

This is assumed to be done already when reading this document, else refer to the release notes.

8.8.2 Install VDE2-2.2.3 from Sources

1. Download and install source.
2. Configure

```
./configure --prefix=/opt/vde-2.2.3
```

3. make
4. set symbolic links

```
ln -s /opt/qemu-0.12.2 /opt/qemu
```

5. Verify installation by creation of a switch, requires root permissions.

```
ctys-vnetctl -u <user-switch> create
```

8.8.3 Install KVM as rpm

CentOS-5.4: The installation of the RPM package is quite forward. For KVM on CentOS-5.4 install the whole virtualization group. The QEMU package is installed here from the source.

Fedora: For Fedora the same procedure.

8.8.4 Install QEMU-0.12.2 from Sources

CentOS-5.4

1. Download and install sources.
2. Install gcc.
3. Configure

```
./configure --prefix=/opt/qemu-0.12.2
```

4. make
5. make install
6. set symbolic links

```
ln -s /opt/qemu-0.12.2 /opt/qemu
```

7. Verify installation

```
ctys-plugins -T QEMU,VNC,X11,CLI -E
```

When the last error message complains the absence of QEMUSOCK and QEMUMGMT, than anything seems to be perfect, just missing the final call for network setup by

8. Call "ctys-vnetctl", requires root permissions.

```
ctys-vnetctl -u <user-switch> create
```

9. Verify installation

```
ctys-plugins -T QEMU,VNC,X11,CLI -E
```

This should work now.

8.9 Install Procedures

8.9.1 Install with Manual Setup

This is just depicted for basic demonstration purposes only, thus presents a minimalistic approach by avoidance of enhanced settings such as network devices.

1. Create directory:

```
mkdir myLabel && cd myLabel
```

2. Create a Disk:

```
qemu-img create -f qcow2 disk.img 5G
```

3. Initialize disk image;

```
dd if=/dev/zero of=disk.img bs=1G count=5
```

4. Install from CDROM iso image:

```
qemu-system-x86_64 -hda disk.img \
  -cdrom ${PATHTOIMG}/CentOS-5.4-x86_64-bin-DVD.iso \
  -boot d
```

5. Boot form Disk-Image into GuestOS:

```
qemu-system-x86_64 -hda disk.img
```

8.9.2 Install with ctys-createConfVM

Install the same GuestOS as the HostOS

The following workflow is foreseen to setup both, pure QEMU based machines and KVM based VMs. The variants KVM as the KQEMU are parameterized accelerators only, which fit perfectly into the overall concept of QEMU. The given example is processed on a HOST machine running CentOS-5.4 and installs the same a GuestOS. The recipe is straight forward and avoids extended details and options for simplicity, e.g.

the GuestOS is by default set to the same as the HOST OS.

In case of errors during the start of the VM after the creation the first file to be checked is the file named '<label>.ctys'. This file contains the configuration settings related to the emulated hardware of the VM. Particularly the type of network interface card and the graphics card may cause problems and require to be changed for some GuestOS when appropriate drivers are not available.

1. Change to the HOST machine where the VM is executed. The whole procedure could be executed remote to, but for first trial the local execution avoids some details with required extended knowledge.
2. For the first time execution check the state of the installed components. The call for display of the state with required subcomponents is:

   ```
   ctys-plugins -T qemu -E
   ```

   The result should be in 'green' state. When errors related to QEMUSOCK and/or QEMUMGMT occur the utility 'ctys-vnetctl' is required to be executed. Due to the allocation of a TAP/TUN device this requires root permissions.

   When errors occur it could be helpful to check the actual required system components. This could be performed by the call:

   ```
   ctys-plugins -d 64,p -T qemu -E
   ```

3. Create a directory for storage of the set of files comprising the virtual machine and change into.
4. Install on a machine locally

   The present example is foreseen for interactive execution. Several of the interactively polled values could be pre-set by environment variables, the available variables and current default values are shown by the '–list-env-var-options' option(shortcut '–levo'). Additionally init files could be used to set common defaults. The whole procedure could be additionally performed either semi or fully automatic, which requires the whole set of values to be pre-set appropriately.

   ```
   ctys-createConfVM -t QEMU --label=tst253
   ```

   You will be asked several questions related to the new VM. The resulting configuration is displayed and stored to a configuration file within the machines directory.

   The parameters LABEL and MAC address are particularly important, because these define per convention the network access facilities of the VM. The MAC address is sufficient when DHCP is configured, else the TCP/IP address has to be set manually. For automatic consistency checks of the MAC and TCP/IP address match a '/etc/ethers' alike database could be setup either by ctys-extractMAC or by ctys-extractARP.
5. Check the contents of the configuration file and wrapper-script for the VM.
6. If not yet done, create a virtual switch, refer to 'ctys-vnetctl' for automation.

   ```
   ctys-vnetctl -u <switchuser> create root@app2
   ```

When executed remotely from a mounted filesystem this could disconnect in some cases the machine. This is due to the required reconfiguration of some network devices, where some of the re-establishment tools may be stored within the disconnedted filesystem. If this occurs use a local account with locally stored ctys files.

7. Start iterative checks by calling the created wrapper script from the commandline. This is shown here for demonstration purposes of the interface only. The actually required set of checks are performed silently by the QEMU/KVM plugin for verification before each call.

```
sh yourWrappername.sh --print --check
```

The following initial errors may occur:

(a) ERROR:Missing QEMUSOCK
SOLUTION: Call ctys-vnetctl

(b) ERROR: Missing boot image
SOLUTION:Set option "–bootmode=INSTALL"

(c) ERROR: Missing INSTCDROM
SOLUTION: Edit config file or use PXE.

(d) ERROR: Unknown display CONSOLE
SOLUTION: Set option "–console=VNC"

(e) ERROR: Missing VNCDISPLAY
SOLUTION: Set option "–vncaccesdisplay=77"
Here choosen "77" for VNC display.

(f) When now a final assembled call is displayed anything should be fine.

(g) Start the call with removed "–check" option.

(h) ERROR: TUNGETIFF and TUNSETSNDBUF
SOLUTION: Ignore these messages.

8. Start installation with the created wrapper script from the local commandline of the HOST machine.

```
sh yourWrappername.sh --console=vnc --vncaccessdisplay=77 --instmode --print
```

The important parameter is here '–instmode', either with or without suboptions. When missing this option the HDD is used as boot device and fails due to missing OS - of course, it is not yet installed.

Alternatively the installation could be performed by calling ctys with the INSTMODE suboption, which is forseen as the standard operation. The call could be the following from within the VMs subdirectory, when the defaults are used for INSTMODE:

```
ctys -t qemu -a create=ID:\$PWD/yourWrappername.ctys,INSTMODE'
```

When alteration of the INSTMODE suboptions is required the following could be applied:

```
ctys \
  -t qemu \
  -a create=ID:\$PWD/yourWrappername.ctys,INSTMODE:CD\%default\%VHDD\%default\%INIT
```

The warnings related to deprecated support of 'vdeq' could be ignored for curent version. The errors related to TUNGETIFF and TUNSETSNDBUF too.

The release CentOS-5.5 requires the change of the DVD medium during installation. Therefore with 'Ctrl-Alt-2' the user interface could be changed to the monitor terminal, where the commands as described in the chapter about CDROM/DVD could be utilized.

The current machine could be canceled either within the monitor or by the call:

```
ctys -t qemu -a cancel=id:\$PWD/yourWrappername.ctys,poweroff:0,force
```

9. Attach a console by usage of "vncviewer :77&", which could require a short startup timeout of the VM before the execution of the vncviewer. Now proceed with the native GuestOS installation.
10. After installation confirm the reboot of the GuestOS, but when the machine hangs just stop it and restart with "–bootmode=VHDD".
11. Once rebooted finish the post-install steps of the GuestOS.
12. That's it.

Install a GuestOS different from the HostOS

The following workflow is quite similar to the previous case, where the GuestOS is identical to the HostOS.

There are just a few deviations for the call of ctys-createConfVM, which requires the GuestOS now to be set either manually, or for frequent usage by a configuration file. The settings could be checked by the option '–levo'.

```
OS=Linux \
OSREL=2.6.26-1-amd64 \
DIST=debian \
DISTREL=5.0.0 \
MAC=00:50:56:16:11:0b \
ACCELERATOR=KVM  \
ctys-createConfVM \
   -t QEMU \
   --label=inst010
   --auto-all
```

When all defaults are pre-set in configuration files the option '–auto-all' could be used as given. The creation of the whole set of initial files than requires about 2-4seconds!

8.9.3 PXE-Boot

The PXE boot in the current versions of QEMU work on CentOS-5.4 from the box. Just start the VM by calling the wrapper script with the option "–bootmode=PXE" or use the "BOOTMODE:PXE" suboption for the create action of ctys.

8.9.4 ISO-Image and DHCP

The installation from an ISO image reuqires the fully qualified absolute pathname to be set within the conf-file. Additonally the option "–bootmode=INSTALL" or the suboption "BOOTMODE:INSTALL" for ctys is required.

8.9.5 Supported/Tested Install-Mechanisms

The actual applicable install mechanism partly depends on the host system, e.g. debootstrap is currently available on debian hosts only.

OS	**deboostrap**	**KSX**	**PXE**	**CD**	**FD**	**HD**	**USB**
Android-2.2	-			*)			
CentOS-5.0	-	X	X	X			
CentOS-5.4	-	X	X	X			
CentOS-5.5	-	*)	*)	X			
Debian-4.0r3-ARM	-	-		X			
Debian-4.0r3		-	X	X			
Debian-5.0.0		-	X	X			
Debian-5.0.6		-	*)	X			
Fedora 8	-	X	X	X			
Fedora 10	-	X	X	X			
Fedora 12	-	X	X	X			
Fedora 13	-	*)	*)	X			
FreeBSD-7.1	-	-		X			
FreeBSD-8.0	-	-		X			
FreeDOS/balder	-	-	-		X		
Mandriva-2010	-	-		X			
MeeGo-1.0	-	-		*)			
ScientificLinux-5.4.1	-	X	X	X			
OpenBSD-4.0	-	-	X				
OpenBSD-4.3	-	-	X				
OpenBSD-4.6(NOK)	-	-	X				
OpenBSD-4.7	-	-	*)				
OpenSuSE-10.2	-	-	X	X			
OpenSuSE-11.1	-	-		X			
OpenSuSE-11.2	-	-		X			
OpenSuSE-11.3	-	-		X			
Solaris 10	-	-		X			
OpenSolaris 2009.6	-	-		X			
Ubuntu-8.04-D		-		X			
Ubuntu-9.10-D		-		X			
Ubuntu-10.10-D		-		X			
uClinux-arm9	-	-		-			
uClinux-coldfire	-	-		-			

Table 8.1: Supported/Tested Install-Mechanisms

*) Under Test.

.

8.10 Installation of Guests

8.10.1 Android

Refer to the specific Use-Case **ctys-uc-Android** .

8.10.2 CentOS-5

The given example is based on the following configuration:

- Host executing *ctys-createConfVM(1)* on debian-5.0.0, no qemu installation.
- Target host for execution of QEMU-KVM with OpenSUSE-11.2, present qemu and kvm installation.
- GuestOS within the KVM based VM is CentOS (*ctys-uc-CentOS(7)*) .

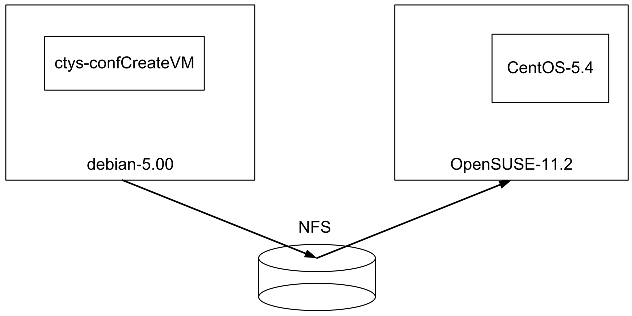


Figure 8.6: Distributed installation by 'ctys-confCreateVM'

The 'cross-installation' for a different machine requires various options to be set for the target execution environment of the hypervisor which cannot be detected automatically on the local machine. One example is the actual support for the architecture.

Several of the parameters has to be set by environment VARIABLE based options. The available options could be visualized by the call option '–list-environment-variable-options' or for short '-levo'. The initial default values wre displayed too, which is not available for dependent values. E.g. by convention wihtin ctys the LABEL of a VM is the hostname of the conatined GuestOS, therefore the value of TCP/IP parameters could be determined only after the LABEL value is defined. The options by environment variable are applicable for local execution only, else the interactive dialogue is available only. The most value of the pre-defined values is gained in combination with oneof the options '–auto' or '–auto-all'. These pre-define confirmation answers for the interactive dialogues.

The following call creates in place of appropriate default values fully automated the complete set of configuration files and wrapper-scripts, including a basic kickstart file.

```
ACCELERATOR=KVM \
DIST=CentOS  \
RELEASE=5.4 \
OS=Linux  \
OSVERSION=2.6.18 \
ctys-createConfVM \
  -t QEMU \
  --label=tst488 \
  --no-create-image \
  --auto-all
```

When dropping the '–auto-all' option an interactive dialogue is performed, where no default values are required neccessarily. The '–no-create-image' avoids the attempt to create a new image, this could be helpful when a present image should be just re-configured, e.g. for usage of an alternative ACCELERATOR, which could be altered by configuration file only. This could be done manually by editing the configuration file too, of course.

The start of the VM requires the execution of 'ctys-vnetctl' for creation of a virtual bridge and a virtual switch. The prerequisite is the complete installation of 'qemu' and 'vde', current tested version for vde is 'vde2-2.2.3' which is available from source-forge. The 'prefix' option should be set by the required 'configure' call to '/opt'. After installation of the sources of vde2 the following should be performed within the installation directory.

```
make clean
./configure --prefix=/opt/vde2-2.2.3
make
make install
ln -s /opt/vde2-2.2.3 /opt/vde
```

The virtual switch could be created by the following call.

```
ctys-vnetctl -u tstUser create
```

A start of the VM is typicall called by the following command:

```
ctys -t QEMU -a create=l:tst488,b:<base-path-VM>,reuse tstUser@testHost
```

8.10.3 CentOS-6

The version 6.0 of CentOS could be installed perfectly as a guest machine, and works with a quite good performance when libvirtio is used for HDD and NIC. Once the basic configuration call is performed the complete installation may take just a few minutes. The description relies on pre-configured installation sources, here an ISO image of the distribution DVD.

The given example is based on the following configuration:

- Callers host executing *ctys-createConfVM(1)* on debian-5.0.0, no qemu installation.
- Target host for execution of QEMU-KVM with CentOS-5.6, present qemu and kvm installation.
- GuestOS within the KVM based VM is CentOS (*ctys-uc-CentOS(7)*) .

The 'cross-installation' for a different machine requires various options to be set for the target execution environment of the hypervisor which cannot be detected automatically on the local machine. One example is the actual support for the architecture.

Several of the parameters has to be set by environment (*VARIABLE*) based options. The available options could be visualized by the call option (*'–list-environment-variable-options' or for short '-levo'.*) The initial default values wre displayed too, which is not available for dependent values. E.g. by convention wihtin ctys the LABEL of a VM is the hostname of the conatined GuestOS, therefore the value of TCP/IP parameters could be determined only after the LABEL value is defined. The options by environment variable are applicable for local execution only, else the interactive dialogue is available only. The most value of the pre-defined values is gained in combination with oneof the options (*'–auto'*) or (*'–auto-all'.*) These pre-define confirmation answers for the interactive dialogues.

The following call creates in place of appropriate default values fully automated the complete set of configuration files and wrapper-scripts, including a basic kickstart file. The following call options with default values apply:

1. **Remote Call - Cross Installation**

```
STARTERCALL=/usr/libexec/qemu-kvm \
WRAPPERCALL=/opt/vde/bin/vdeq \
ACCELERATOR=KVM \
DIST=CentOS \
RELEASE=6.0 \
OS=Linux \
OSVERSION=2.6.32 \
BOOTIMAGE_INST_SIZE=40G \
BOOTIMAGE_INST_BLOCKCOUNT=160 \
SMP=3 \
ctys-createConfVM \
  -t QEMU \
  --label=dev004 \
  --create-image \
  --auto-all \
  -C -D $PWD/dev004 \
  --virtiohdd \
  --virtionet
```

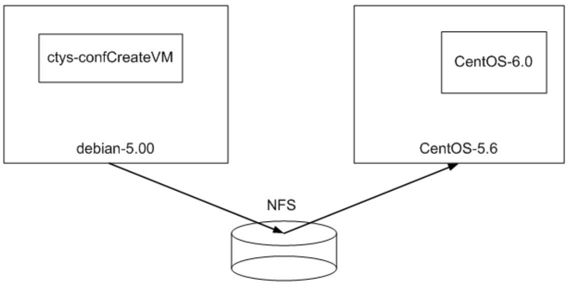


Figure 8.7: Distributed installation by 'ctys-confCreateVM'

2. **Local Call with different HostOS and GuestOS.**
   The values (*STARTERCALL, WRAPPERCALL, and ACCELERATOR*) are evaluated dynamically, but could be pre-set too.

```
DIST=CentOS \
RELEASE=6.0 \
OS=Linux \
OSVERSION=2.6.32 \
BOOTIMAGE_INST_SIZE=40G \
BOOTIMAGE_INST_BLOCKCOUNT=160 \
SMP=3 \
ctys-createConfVM \
  -t QEMU \
  --label=dev004 \
  --create-image \
  --auto-all \
  -C -D $PWD/dev004 \
  --virtiohdd \
  --virtionet
```

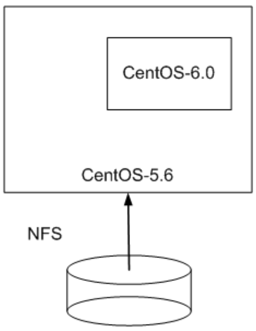


Figure 8.8: Local installation - different OS - by 'ctys-confCreateVM'

3. **Local Call with identical HostOS and GuestOS**
   The values (*STARTERCALL, WRAPPERCALL, ACCELERATOR, DIST, RELEASE, OS, and OSVERSION*) are evaluated dynamically, but could be pre-set too.

```
BOOTIMAGE_INST_SIZE=40G \
BOOTIMAGE_INST_BLOCKCOUNT=160 \
SMP=3 \
ctys-createConfVM \
  -t QEMU \
  --label=dev004 \
  --create-image \
  --auto-all \
  -C -D $PWD/dev004 \
  --virtiohdd \
  --virtionet
```

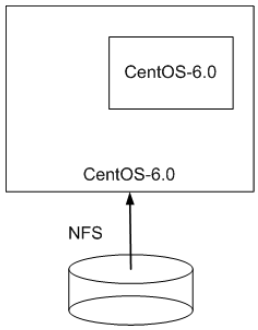


Figure 8.9: Local installation - same OS - by 'ctys-confCreateVM'

When dropping the (*'–auto-all'.*) option an interactive dialogue is performed, where no default values are required neccessarily. The (*'–no-create-image'.*) avoids the attempt to create a new image, this could be helpful when a present image should be just re-configured, e.g. for usage of an alternative (*ACCELERATOR*) , which could be altered by configuration file only. This could be done manually by editing the configuration file too, of course.

The start of the VM requires the execution of (*'ctys-vnetctl(1)'*) for creation of a virtual bridge and a virtual switch. The prerequisite is the complete installation of 'qemu' and 'vde', current tested version for vde is 'vde2-2.2.3' which is available from source-forge. The 'prefix' option should be set by the required 'configure' call to '/opt'. After installation of the sources of vde2 the following should be performed within the installation directory.

```
make clean
./configure --prefix=/opt/vde2-2.2.3
make
make install
ln -s /opt/vde2-2.2.3 /opt/vde
```

The virtual switch could be created by the following call.

```
ctys-vnetctl -u tstUser create
```

A start of the VM is typicall called by the following command:

```
ctys -t QEMU -a create=l:dev004,b:<base-path-VM>,reuse,instmode tstUser@testHost
```

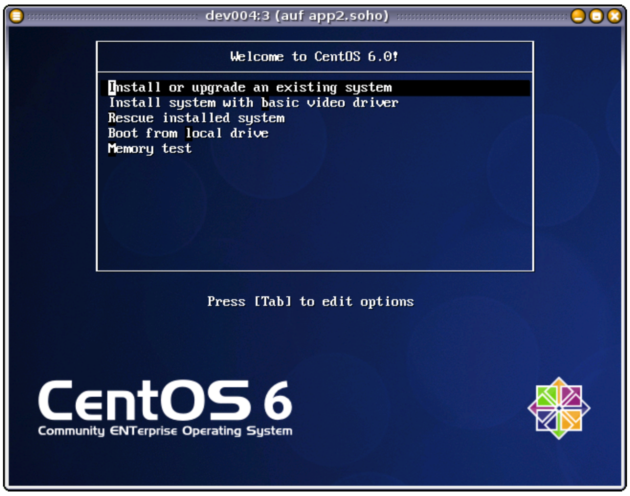


Figure 8.10: Installation screen

8.10.4 Debian-5

The installation of debian is straight forward in accordance to the generic description for CentOS. The following pitfalls have to be avoided when more than one version of QEMU/KVM is installed.

1. The variable QEMUBIOS has to point to the suitable set of BIOS modules for the actually executed version. By default the omission of the variable at all should provide that. The setting could be handeled by SHELL, config-QEMU-files, and within the specific VM.
2. Probably it is a good idea to deactivate the screensaver first, the screen saver for the HOST should suffice.

The basic setup of debian-lenny could be performed with following steps:

1. Install debian-lenny-5.0.0/5.x with basic settings, here Gnome.
2. Add: tightVNC server and client
3. Add: sshd
4. Add: bridge-utils
5. Add: socat (required for '-U'-UnixDomain-Option).

6. Configure password-less login for SSH e.g. ssh-copy-id or by Kerberos.
7. Install ctys e.g.:

```
ctys-distribute -F 1 -P UserHomeCopy root@tst210
```

   For activation of environment variables either a fresh login or the manual 'source' of the '$HOME/.bashrc' or the '$HOME/.profile' is required on the target machine.
8. Remote execution of eg.

```
ctys -a info root@tst210
```

9. Remote or local execution for preparation of the inventory data and the normalized performance characteristic for possible comparison by:

```
ctys-genmconf -P -x VM root@tst210
```

10. Check now results of remote execution for PM-section

```
ctys -a info root@tst210
```

11. Start a VNC session

```
ctys -a create=l:ROOT,reuse root@tst210
```

8.10.5 Debian-6

The version 6.0 of CentOS could be installed perfectly as a guest machine, and works with a quite good performance when libvirtio is used for HDD and NIC. Once the basic configuration call is performed the complete installation may take just a few minutes. The description relies on pre-configured installation sources, here an ISO image of the distribution DVD.

The given example is based on the following configuration:

- Callers host executing *ctys-createConfVM(1)* on debian-5.0.0, no qemu installation.
- Target host for execution of QEMU-KVM with CentOS-5.6, present qemu and kvm installation.
- GuestOS within the KVM based VM is CentOS (*ctys-uc-CentOS(7)*) .

The 'cross-installation' for a different machine requires various options to be set for the target execution environment of the hypervisor which cannot be detected automatically on the local machine. One example is the actual support for the architecture.

Several of the parameters has to be set by environment (*VARIABLE*) based options. The available options could be visualized by the call option (*'–list-environment-variable-options' or for short '-levo'.*) The initial default values wre displayed too, which is not available for dependent values. E.g. by convention wihtin ctys the LABEL of a VM is the hostname of the conatined GuestOS, therefore the value of TCP/IP parameters could be determined only after the LABEL value is defined. The options by environment variable are applicable for local execution only, else the interactive dialogue is available only. The most value of the pre-defined values is gained in combination with oneof the options (*'–auto'*) or (*'–auto-all'.*) These pre-define confirmation answers for the interactive dialogues.

The following call creates in place of appropriate default values fully automated the complete set of configuration files and wrapper-scripts, including a basic kickstart file. The following call options with default values apply:

1. **Remote Call - Cross Installation**

```
STARTERCALL=/usr/libexec/qemu-kvm \
WRAPPERCALL=/opt/vde/bin/vdeq \
ACCELERATOR=KVM \
DIST=CentOS \
RELEASE=6.0 \
OS=Linux \
OSVERSION=2.6.32 \
BOOTIMAGE_INST_SIZE=40G \
BOOTIMAGE_INST_BLOCKCOUNT=160 \
SMP=3 \
ctys-createConfVM \
  -t QEMU \
  --label=dev004 \
  --create-image \
  --auto-all \
  -C -D $PWD/dev004 \
  --virtiohdd \
  --virtionet
```

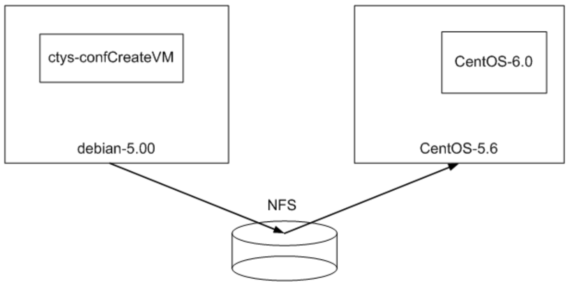


Figure 8.11: Distributed installation by 'ctys-confCreateVM'

2. **Local Call with different HostOS and GuestOS.**
   The values (*STARTERCALL, WRAPPERCALL, and ACCELERATOR*) are evaluated dynamically, but could be pre-set too.

```
DIST=CentOS \
RELEASE=6.0 \
OS=Linux \
OSVERSION=2.6.32 \
BOOTIMAGE_INST_SIZE=40G \
BOOTIMAGE_INST_BLOCKCOUNT=160 \
SMP=3 \
ctys-createConfVM \
  -t QEMU \
  --label=dev004 \
  --create-image \
  --auto-all \
  -C -D $PWD/dev004 \
  --virtiohdd \
  --virtionet
```

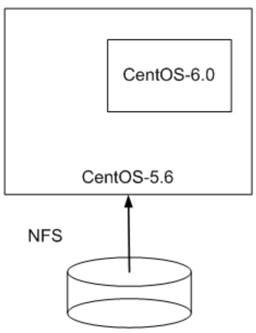


Figure 8.12: Local installation - different OS - by 'ctys-confCreateVM'

3. **Local Call with identical HostOS and GuestOS**
   The values (*STARTERCALL, WRAPPERCALL, ACCELERATOR, DIST, RELEASE, OS, and OSVERSION*) are evaluated dynamically, but could be pre-set too.

```
BOOTIMAGE_INST_SIZE=40G \
BOOTIMAGE_INST_BLOCKCOUNT=160 \
SMP=3 \
ctys-createConfVM \
  -t QEMU \
  --label=dev004 \
  --create-image \
  --auto-all \
  -C -D $PWD/dev004 \
  --virtiohdd \
  --virtionet
```

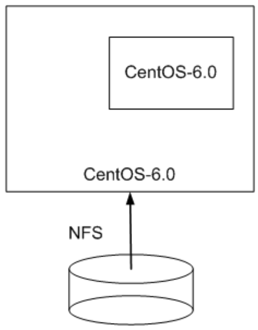


Figure 8.13: Local installation - same OS - by 'ctys-confCreateVM'

When dropping the (*'–auto-all'.*) option an interactive dialogue is performed, where no default values are required neccessarily. The (*'–no-create-image'.*) avoids the attempt to create a new image, this could be helpful when a present image should be just re-configured, e.g. for usage of an alternative (*ACCELERATOR*) , which could be altered by configuration file only. This could be done manually by editing the configuration file too, of course.

The start of the VM requires the execution of (*'ctys-vnetctl(1)'*) for creation of a virtual bridge and a virtual switch. The prerequisite is the complete installation of 'qemu' and 'vde', current tested version for vde is 'vde2-2.2.3' which is available from source-forge. The 'prefix' option should be set by the required 'configure' call to '/opt'. After installation of the sources of vde2 the following should be performed within the installation directory.

```
make clean
./configure --prefix=/opt/vde2-2.2.3
make
make install
ln -s /opt/vde2-2.2.3 /opt/vde
```

The virtual switch could be created by the following call.

```
ctys-vnetctl -u tstUser create
```

A start of the VM is typicall called by the following command:

```
ctys -t QEMU -a create=l:dev004,b:<base-path-VM>,reuse,instmode tstUser@testHost
```

Figure 8.14: Installation screen

8.10.6 Fedora-10

The installation is quite forward.

Install the base system as described in the generic 'Installation Procedure'. Follow these steps for a base system with hosts and PMs features.

1. Adapt '/etc/yum.repo.d/fedora.repo' appropriately.
2. Install - if no else is available - from the CentOS-5.4 package, use '–nodeps':
   - libsysfs-2.0.0-6.x86_64.rpm
   - vnc-4.1.2.14.el5_3.1.x86_64.rpm
   - vnc-server-4.1.2.14.el5_3.1.x86_64.rpm
   - bridge-utils-1.1-2.x86_64.rpm
3. Configure password-less login for SSH e.g. ssh-copy-id or by Kerberos.
4. Install ctys e.g.:

   ```
   ctys-distribute -F 1 -P UserHomeCopy root@tst240
   ```

   For activation of environment variables either a fresh login or the manual 'source' of the '$HOME/.bashrc' or the '$HOME/.profile' is required on the target machine.

5. Remote execution of eg.

```
ctys -a info root@tst240
```

6. Remote or local execution for preparation of the inventory data and the normalized performance characteristic for possible comparison by:

```
ctys-genmconf -P -x VM root@tst240
```

7. Check now results of remote execution for PM-section

```
ctys -a info root@tst240
```

8. Start a VNC session

```
ctys -a create=l:ROOT,reuse root@tst240
```

8.10.7 Fedora-15

The version 15 of Fedora could be installed perfectly as a guest machine, and works with a quite good performance when libvirtio is used for HDD and NIC. Once the basic configuration call is performed the complete installation may take just a few minutes. Just the following two specifics are to be considered:

1. Memory size:
   Is required to be about mor than 1G, e.g. 1.5G, else the installation hangs when unpacking initfs.
2. VGA type:
   Seems to be required of type 'std', the 'cirrus' driver may not work.

The description relies on pre-configured installation sources, here an ISO image of the distribution DVD.

The given example is based on the following configuration:

- Callers host executing *ctys-createConfVM(1)* on debian-5.0.0, no qemu installation.
- Target host for execution of QEMU-KVM with CentOS-5.6, present qemu and kvm installation.
- GuestOS within the KVM based VM is CentOS (*ctys-uc-CentOS(7)*) .

The 'cross-installation' for a different machine requires various options to be set for the target execution environment of the hypervisor which cannot be detected automatically on the local machine. One example is the actual support for the architecture.

Several of the parameters has to be set by environment (*VARIABLE*) based options. The available options could be visualized by the call option (*'–list-environment-variable-options' or for short '-levo'.*) The initial default values wre displayed too, which is not available for dependent values. E.g. by convention wihtin ctys the LABEL of a VM is the hostname of the conatined GuestOS, therefore the value of TCP/IP parameters could be determined only after the LABEL value is defined. The options by environment variable are applicable for local execution only, else the interactive dialogue is available only. The most value of the pre-defined values is gained in combination with oneof the options (*'–auto'*) or (*'–auto-all'.*) These pre-define confirmation answers for the interactive dialogues.

The following call creates in place of appropriate default values fully automated the complete set of configuration files and wrapper-scripts, including a basic kickstart file. The following call options with default values apply:

1. **Remote Call - Cross Installation**

```
STARTERCALL=/usr/libexec/qemu-kvm \
WRAPPERCALL=/opt/vde/bin/vdeq \
ACCELERATOR=KVM \
DIST=CentOS \
RELEASE=6.0 \
OS=Linux \
OSVERSION=2.6.32 \
BOOTIMAGE_INST_SIZE=40G \
BOOTIMAGE_INST_BLOCKCOUNT=160 \
SMP=3 \
ctys-createConfVM \
  -t QEMU \
  --label=dev004 \
  --create-image \
  --auto-all \
  -C -D $PWD/dev004 \
  --virtiohdd \
  --virtionet
```

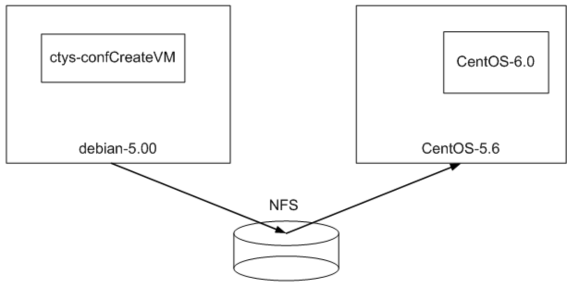


Figure 8.15: Distributed installation by 'ctys-confCreateVM'

2. **Local Call with different HostOS and GuestOS.**
   The values (*STARTERCALL, WRAPPERCALL, and ACCELERATOR*) are evaluated dynamically, but could be pre-set too.

```
DIST=CentOS \
RELEASE=6.0 \
OS=Linux \
OSVERSION=2.6.32 \
BOOTIMAGE_INST_SIZE=40G \
BOOTIMAGE_INST_BLOCKCOUNT=160 \
SMP=3 \
ctys-createConfVM \
  -t QEMU \
  --label=dev004 \
  --create-image \
  --auto-all \
  -C -D $PWD/dev004 \
  --virtiohdd \
  --virtionet
```

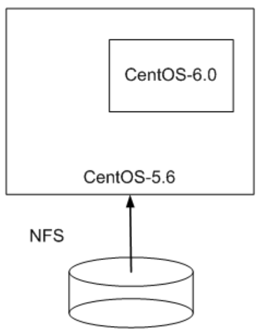


Figure 8.16: Local installation - different OS - by 'ctys-confCreateVM'

3. **Local Call with identical HostOS and GuestOS**
   The values (*STARTERCALL, WRAPPERCALL, ACCELERATOR, DIST, RELEASE, OS, and OSVERSION*) are evaluated dynamically, but could be pre-set too.

```
BOOTIMAGE_INST_SIZE=40G \
BOOTIMAGE_INST_BLOCKCOUNT=160 \
SMP=3 \
ctys-createConfVM \
  -t QEMU \
  --label=dev004 \
  --create-image \
  --auto-all \
  -C -D $PWD/dev004 \
  --virtiohdd \
  --virtionet
```

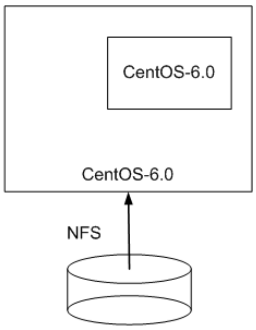


Figure 8.17: Local installation - same OS - by 'ctys-confCreateVM'

When dropping the (*'–auto-all'.*) option an interactive dialogue is performed, where no default values are required neccessarily. The (*'–no-create-image'.*) avoids the attempt to create a new image, this could be helpful when a present image should be just re-configured, e.g. for usage of an alternative (*ACCELERATOR*) , which could be altered by configuration file only. This could be done manually by editing the configuration file too, of course.

The start of the VM requires the execution of (*'ctys-vnetctl(1)'*) for creation of a virtual bridge and a virtual switch. The prerequisite is the complete installation of 'qemu' and 'vde', current tested version for vde is 'vde2-2.2.3' which is available from source-forge. The 'prefix' option should be set by the required 'configure' call to '/opt'. After installation of the sources of vde2 the following should be performed within the installation directory.

```
make clean
./configure --prefix=/opt/vde2-2.2.3
make
make install
ln -s /opt/vde2-2.2.3 /opt/vde
```

The virtual switch could be created by the following call.

```
ctys-vnetctl -u tstUser create
```

A start of the VM is typicall called by the following command:

```
ctys -t QEMU -a create=l:dev004,b:<base-path-VM>,reuse,instmode tstUser@testHost
```

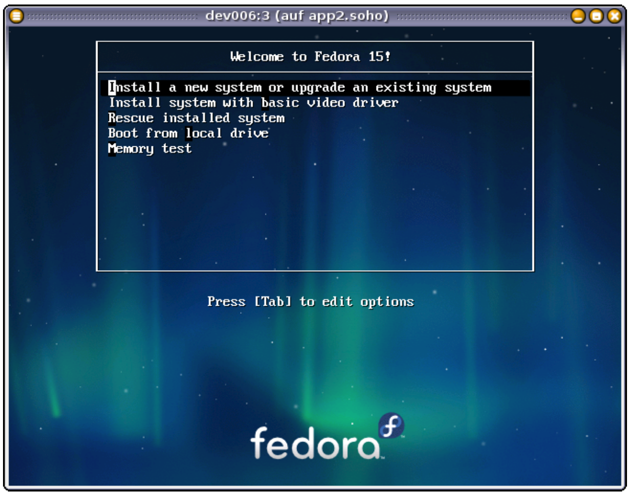


Figure 8.18: Installation screen

8.10.8 FreeBSD-7.1

1. Install by DVD-image
2. Add: tightVNC, bash, xorg, pciutils, gnome-sessions, fvwm

8.10.9 FreeBSD-8.0

1. ctys-createConfVM
2. Install by DVD-image
3. Configure network with DHCP, and ssh-login
4. chpass for setting bash
5. Add tightVNC and pci-utils
   ffs.

8.10.10 MeeGo

Refer to the specific Use-Case **ctys-uc-MeeGo** .

8.10.11 OpenBSD-4.3+SerialConsole - by PXE

The installation is straight forward with BOOTMODE=PXE. When for later access a serial console is required following additional steps has to be proceeded. A serial console is a prerequisite for EMACS, XTERM, and GTERM.

1. Boot and login into GuestOS if the root-filesystem cannot be mounted offline.
2. Create a file "/etc/boot.conf" with content such as:

```
set tty com0
stty com0 115200
set timeout 15
boot
```

+Edit the file "/etc/tty" and change the lines:

```
tty00   "/usr/libexec/getty std.115200" vt220 on secure
```

8.10.12 OpenBSD-4.6+SerialConsole - by PXE

The Version of kvm-83 on CentOS-5.4 requires following steps for boot of installation:

1. Use e1000 driver for NIC.
2. Boot by calling: 'bsd.rd -c'
3. Disable mpbios:'disable mpbios'

After install make the changes persistent with:

1. config -f -e /bsd
   - disable mpbios
   - quit

     The following packages are required in addition to the base installation.
2. tightVNC + tightVNCviewer
3. bash
4. pciutils
5. eventually gnome-session

   Once the installation is complete the following steps should be proceeded:
6. Aktivate X11Forwarding for SSH
7. Execute:

```
ctys -a info root@host
```

8. Execute:

   `ctys-genmconf -P -x VM root@host`

8.10.13 OpenSolaris-2009.6

ffs.

8.10.14 openSuSE-11.2

Install the base system as described in the generic 'Installation Procedure'. Follow these steps for a base system with hosts and PMs features.

1. Add your repository within yast and install.
   - bridge-utils
2. Download vde - verified with vde2-2.2.3
   [sourceforge.net/projects/vde sourceforge.net/projects/vde]
   - install sources
   - call 'configure –prefix=/opt/vde2-2.2.3'
   - call 'make'
   - call 'make install'
   - call 'ln -s /opt/vde2-2.2.3 /opt/vde'
3. Configure password-less login for SSH e.g. ssh-copy-id or by Kerberos.
4. Install ctys e.g.:

   ```
   ctys-distribute -F 2 -P UserHomeCopy root@tst214
   ```

   For activation of environment variables either a fresh login or the manual 'source' of the '$HOME/.bashrc' or the '$HOME/.profile' is required on the target machine.
5. Remote execution of eg.

   ```
   ctys -a info root@tst214
   ```

6. Remote or local execution for preparation of the inventory data and the normalized performance characteristic for possible comparison by:

   ```
   ctys-genmconf -P -x VM root@tst214
   ```

7. Check now results of remote execution for PM-section

   ```
   ctys -a info root@tst214
   ```

8. Start a VNC session

   ```
   ctys -a create=l:ROOT,reuse root@tst214
   ```

8.10.15 ScientificLinux-5.4.1

Install the base system as described in the generic 'Installation Procedure'. Follow these steps for a base system with hosts and PMs features.

1. Add your repository within yast and install.
   - bridge-utils
   - vnc
2. Configure password-less login for SSH e.g. ssh-copy-id or by Kerberos.
3. Install ctys e.g.:

   ```
   ctys-distribute -F 2 -P UserHomeCopy root@tst213
   ```

   For activation of environment variables either a fresh login or the manual 'source' of the '$HOME/.bashrc' or the '$HOME/.profile' is required on the target machine.
4. Remote execution of eg.

   ```
   ctys -a info root@tst213
   ```

5. Remote or local execution for preparation of the inventory data and the normalized performance characteristic for possible comparison by:

   ```
   ctys-genmconf -P -x VM root@tst213
   ```

6. Check now results of remote execution for PM-section

   ```
   ctys -a info root@tst213
   ```

7. Start a VNC session

   ```
   ctys -a create=l:ROOT,reuse root@tst213
   ```

8.10.16 Solaris-10

ffs.

8.10.17 Ubuntu-8.04

Install the base system as described in the generic 'Installation Procedure'. Follow these steps for a base system with hosts and PMs features.

1. Add your repository within yast and install. Here the debian distribution debian-5.0.0-lenny is used partly.
   - bridge-utils
   - openssh-server

- tightvnc client+server

2. Download vde - verified with vde2-2.2.3
   [sourceforge.net/projects/vde sourceforge.net/projects/vde]
   - install sources
   - call 'configure –prefix=/opt/vde2-2.2.3'
   - call 'make'
   - call 'make install'
   - call 'ln -s /opt/vde2-2.2.3 /opt/vde'
3. Configure password-less login for SSH e.g. ssh-copy-id or by Kerberos.
4. Install ctys e.g.:

   ```
   ctys-distribute -F 2 -P UserHomeCopy root@tst236
   ```

   For activation of environment variables either a fresh login or the manual 'source' of the '$HOME/.bashrc' or the '$HOME/.profile' is required on the target machine.
5. Remote execution of eg.

   ```
   ctys -a info root@tst236
   ```

6. Login as e.g. tst@tst236

   ```
   ssh -X tst@tst236
   ```

7. Local execution for preparation of the inventory data and the normalized performance characteristic for possible comparison by:

   ```
   sudo PATH=\$PATH:\$HOME/bin ctys-genmconf -P -x VM root@tst236
   ```

8. Check now results of remote execution for PM-section

   ```
   ctys -a info root@tst236
   ```

9. Start a VNC session

   ```
   ctys -a create=l:ROOT,reuse root@tst236
   ```

8.10.18 Ubuntu-9.10

Install the base system as described in the generic 'Installation Procedure'. Follow these steps for a base system with hosts and PMs features.

1. Add your repository within yast and install. Here the debian distribution debian-5.0.0-lenny is used partly.
   - bridge-utils
   - tightvnc client+server

2. For openssh-server the appropriate Ubuntu package is required due to version dependency.
   - Download package openssh-server
   - install: dpkt -i openssh-server...
3. Download vde - verified with vde2-2.2.3
   [sourceforge.net/projects/vde sourceforge.net/projects/vde]
   - install sources
   - call 'configure –prefix=/opt/vde2-2.2.3'
   - call 'make'
   - call 'make install'
   - call 'ln -s /opt/vde2-2.2.3 /opt/vde'
4. Configure password-less login for SSH e.g. ssh-copy-id or by Kerberos.
5. Install ctys e.g.:

   ```
   ctys-distribute -F 2 -P UserHomeCopy root@tst215
   ```

   For activation of environment variables either a fresh login or the manual 'source' of the '$HOME/.bashrc' or the '$HOME/.profile' is required on the target machine.
6. Remote execution of eg.

   ```
   ctys -a info root@tst215
   ```

7. Login as e.g. tst@tst215

   ```
   ssh -X tst@tst215
   ```

8. Local execution for preparation of the inventory data and the normalized performance characteristic for possible comparison by:

   ```
   sudo PATH=\$PATH:\$HOME/bin ctys-genmconf -P -x VM root@tst215
   ```

9. Check now results of remote execution for PM-section

   ```
   ctys -a info root@tst215
   ```

10. Start a VNC session

    ```
    ctys -a create=l:ROOT,reuse root@tst215
    ```

8.10.19 QNX-6.4.0

The installation is quite straight forward. The setup of the configuration file and the preparation of the install media could be performed by the following call on the target machine:

1. mkdir <directory>

2. Call this on target machine, due to pre-set environment variables for the first step of installation. This prepares the machine for the actuall installation:

```
INSTCDROM=/mntn/swpool/miscOS/QNX/6.4.0/raw/qnxsdp-6.4.0-200810211530-dvd.iso \
MAC=00:50:56:13:13:18 \
IP=172.20.6.23 \
DIST=QNX-SDP \
DISTREL=6.4.0 \
OS=QNX \
OSREL=6.2.3 \
ctys-createConfVM \
  -t qemu \
  --label=tst323 \
  --auto-all
```

3. Start installation from anywhere:

```
ctys -t qemu \
  -a create=l:tst323,reuse,instmode:CD\%default\%HDD\%default\%init,\
     b:/mntn/vmpool/vmpool05/qemu/test/tst-ctys/tst323 \
  -c local app1
```

4. Cancel installed machine for reboot.

```
ctys -t qemu -a cancel=l:tst323,poweroff root@lab02
```

5. Start QNX.

```
ctys -t qemu \
  -a create=l:tst323,reuse,b:/mntn/vmpool/vmpool05/qemu/test/tst-ctys/tst323 \
  -c local app1

\ \\
```

8.10.20 QEMU-arm-test

The configuration and integration of the provided test image for ARM based uCLinux is is quite straight forward. The setup of the configuration file and the preparation of the install media could be performed by the following call on the target machine:

1. mkdir <directory>
2. Copy and unpack the image-archive into the fresh directory.
3. Call this on target machine, due to pre-set environment variables for the first step of installation. This prepares the machine for the actuall installation:

```
ACCELERATOR=QEMU \
STARTERCALL=/opt/qemu/bin/qemu-system-arm \
ARCH=arm926 \
NETMASK=255.255.0.0 \
MAC=00:50:56:13:13:19 \
IP=172.20.6.24 \
DIST=QEMU-arm-test \
DISTREL=0.9.1-0.2 \
OS=ucLinux \
OSREL=2.x \
MEMSIZE=128 \
HDDBOOTIMAGE\_INST\_SIZE=2G \
HDDBOOTIMAGE\_INST\_BLOCKCOUNT=8 \
INST\_KERNEL=zImage.integrator \
INST\_INITRD=arm\_root.img \
ctys-createConfVM \
  -t QEMU \
  --label=tst324 \
  --auto-all
```

4. Either edit the sections or just append the following to the 'tst324.ctys' configuration file, refer to provided README:

```
KERNELIMAGE=zImage.integrator
INITRDIMAGE=arm\_root.img

\#For: -nographic
\#APPEND=\${APPEND:-console=ttyAMA0}

CPU=arm926
```

5. Start virtual machine:

```
ctys -t qemu \
  -a create=l:tst324,reuse\
     b:/mntn/vmpool/vmpool05/qemu/test/tst-ctys/tst324 \
  -c local app1
```

6. Cancel installed machine for reboot.

```
ctys -t qemu -a cancel=l:tst324,poweroff root@app1
```

8.10.21 QEMU-coldfire-test

The configuration and integration of the provided test image for Coldfire based uCLinux is is quite straight forward. The setup of the configuration file and the preparation of the install media could be performed by the following call on the target machine:

1. mkdir <directory>
2. Copy and unpack the image-archive into the fresh directory.
3. Call this on target machine, due to pre-set environment variables for the first step of installation. This prepares the machine for the actuall installation:

```
ACCELERATOR=QEMU \
STARTERCALL=/opt/qemu/bin/qemu-system-m68k \
ARCH=ColdFire \
NETMASK=255.255.0.0 \
MAC=00:50:56:13:13:1a \
IP=172.20.6.25 \
DIST=QEMU-coldfire-test \
DISTREL=0.9.1-0.1 \
OS=ucLinux \
OSREL=2.x \
MEMSIZE=128 \
HDDBOOTIMAGE\_INST\_SIZE=2G \
HDDBOOTIMAGE\_INST\_BLOCKCOUNT=8 \
INST\_KERNEL=vmlinux-2.6.21-uc0 \
ctys-createConfVM \
  -t QEMU \
  --label=tst325 \
  --auto-all
```

4. Either edit the sections or just append the following to the 'tst324.ctys' configuration file, refer to provided README:

```
KERNELIMAGE=vmlinux-2.6.21-uc0
INITRDIMAGE=;

\# -nographic
VGA\_DRIVER=" -nographic "
CPU=any
NIC=;
```

5. Start virtual machine:

```
ctys -t qemu \
  -a create=l:tst325,b:\$PWD,reuse,console:gterm \
  -d pf,1 \
  app1'(-d pf,1)'
```

8.10.22 UnbreakableLinux-5.4

ffs.

.

8.10.23 Installed Guest Systems

OS	**name**	**Inst-VM**	**Media**
Android-2.2	*)	QEMU	ISO
CentOS-5.0	tstxxx	QEMU, KVM	PXE,ISO
CentOS-5.4	tst131	QEMU, KVM	PXE,ISO
CentOS-5.5	inst012	KVM	ISO
CentOS-5.6		KVM	PXE,ISO
CentOS-6.0		KVM	PXE,ISO
Debian-4.0r3-ARM	tst102	QEMU	ISO
Debian-4.0r3	tst130	QEMU	PXE,ISO
Debian-5.0.0	tst210	KVM	PXE,ISO,debootstrap
Debian-5.0.6	inst013	KVM	ISO
Debian-6.0.0		KVM	PXE,ISO,debootstrap
Fedora8	tst240	KVM	PXE
Fedora10	tst239	KVM	ISO
Fedora12	tst211	KVM	ISO
Fedora13	inst011	KVM	ISO
Fedora15		KVM	PXE,ISO
FreeBSD-7.1	tst238	KVM	ISO
FreeBSD-8.0	tst218	KVM	ISO
Mandriva-2010	tst212	KVM	ISO
MeeGo-1.0	*)	QEMU	ISO
NetBSD-4.7	*)	KVM	ISO
ScientifiLinux-5.4.1	tst213	KVM	PXE,ISO
OpenBSD-4.0	tst124	QEMU	ISO
OpenBSD-4.3	tst127	QEMU	ISO
OpenBSD-4.7	*)	KVM	ISO
OpenSuSE-10.2	tst153	QEMU	PXE,ISO
OpenSuSE-11.2	tst214	KVM	PXE,ISO
OpenSuSE-11.3	inst014	KVM	ISO
OpenSuSE-11.4		KVM	PXE,ISO
OpenSolaris 2009.6	tst241	KVM	ISO
QNX-6.4.0	tst323	KVM	ISO
Solaris 10	tst217	KVM	ISO
Ubuntu-8.04-D	tst236	QEMU,KVM	ISO
Ubuntu-9.10-D	tst215	KVM	ISO
Ubuntu-10.10-D	inst015	KVM	ISO
uClinux-arm9	tst324	QEMU	(QEMU)
uClinux-coldfire	tst325	QEMU	(QEMU)

Table 8.2: Overview of Installed-QEMU/KVM-VMs

In addition various test packages with miscellaneous CPU emulations of QEMU are available. Example templates for integration scripts are provided for ARM, Coldfire, MIPS, and PPC.

.

8.11 Configuration Files

8.11.1 Directory Structure

The expected default directory structure for the assembly of the runtime call is as depicted in the following figure. VMs could be placed anywhere within the filesystem an are detected by the **ENUMERATE** action with provided **BASE** parameter.

```
$HOME
+---.ctys
|   |
|   +qemu       QEMU specific configuration files for user specific
|               settings, else the installed are used as default.
|
|
```

Figure 8.19: UnifiedSessionsManager QEMU file structure

The VMs could be installed anywhere, as long the configuration file and the wrapper file have the same filename prefix and allocated together with the boot image within the same directory. The naming convention provides the following variants:

1. The directoryname is the conffilename prefix.
2. The image filename and the conf filename have the same prefix.
3. The label is the conffile prefix.

8.11.2 Configuration File

The Initialization of the framework comprize mainly the bootstrap of initial hooks for a specific framework version.

The configuration file is sourced into the wrapper file, thus allowing some actual runtime variables set coallocated - with the for now - additionally set ENUMERATE parameters. In case of required runtime parameters these parameter has to be literally identical to their ENUMERATE peers.

8.11.3 Ctys-Wrapper File

The actions to perform whitin a wrapper script comprize

1. Initialization of the framework The Initialization of the framework comprize mainly the bootstrap of initial hooks for a specific framework version.
2. Assembly of the static generic call parts The next steps evaluate the dynamic call parameters as choosen by the user for this specific execution thread. First of all the parameters are extracted from the CLI and the generic part of the actual execution string is assembled.
3. Assembly of the specific call and final execution

Altough the wrapper script could be varied as required, the basic structure should be kept for simplicity.

8.11.4 Syntax Elements

For additional information on generated files refer to the description of **ctys-createConfVM(1)**

Chapter 9

VMW Setup

.

9.1 VMW - Basics for Operations

The ctys-VMW plugin defines a meta-layer for an abstract interface, which is in current version based on the final commandline call of the supported products. The ctys-VMW plugin supports a subset of the products native command line options mapped to the ctys call options, the remaining are bypassed as native options. Future versions are going to provide an abstract encapsulation layer by a common ctys-wrapper script and in addition utilization of the vendor provided management interfaces for batch-mode installation and operations.

The call structure fits into the common structure of ctys but for the current version the ctys-wrapper script is not yet supported for the VMW subsystem.

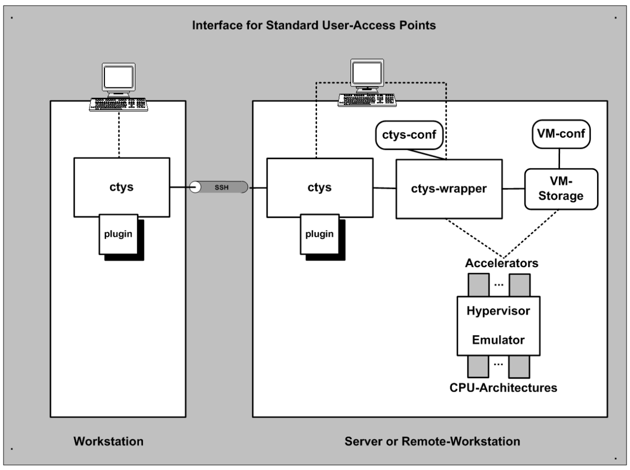


Figure 9.1: ctys distributed access

The VMW subsystem supports the standard configuration files created by the vendor utilities. Thus any existing installation with any already present VMs and vmx-files could be used without required adaptation. These could be either used by dynamic addressing, dynamic search, or by unmodified scan into the inventory database. The creation of new VMs is foreseen by the standandard procedures as defined by the vendor and supported by the products.

REMARK:

The current version requires the access to the VMX-configuration-files, thus these have to be either registered in the UnifiedSessionsManager inventory database, or provided by the call sub-options for scan-start **base:**, or by explicit addressing **id:**. For newer versions e.g. of VMware-Server-2x(TM) in case of usage of subdirectories for the actual storage of the virtual devices, the configuration and management information is probably still stored in the root of the configured storage. This should be post-merged manual with the virtual device into a unique shared directory comprisingly representing the VM.

The lack of the standard vmx-files is the missing of offline information about the GuestOS. The UnifiedSessionsManager provides for basically two options for the coallocated storage of additional extended GuestOS information.

- Coallocated **ctys-configuration-file** created by **ctys-createConfVM**
- Extension of the vendor configuration file by **ctys-keywords** within **comment-protected strings**

The **ctys-configuration-file** is defined by convention to be stored within the directory of the virtual machine, coallocated with the vmx-file and named with the same prefix as the vmx-file: **<vmx-file-prefix>.ctys** and the containing directory **..../<vmx-file-prefix>**. Even though this probably could varied, this has to be avaoided due to probable unanounced remove of the required features. The file could be created and added manually by the user with the same following optional "ctys-conf" vmx-file modifications and by the previous naming convention.

Alternatively the keywords could be stored within the vmx-file, due to the key-prefixes defined to begin with a valid vmx-comment character these are protected for unintended evaluation by the hypervisor. The provided example may suffice the most required offline information for the user-management of the VM, including the automated generation of a cache database for the network inventory.

```
#@#MAGICID-VMW

#@#VMSTATE="ACTIVE"
#@#SERNO="20080415051600"
#@#VERSION="01.01.001"
####MAC0 is provided by the eth0 of vmx-file!
#@#IP0="192.168.1.235"
#@#DIST="CentOS"
#@#DISTREL="5"
#@#OS="Linux"
#@#OSREL="2.6"
#@#CATEGORY="VM"
```

The current version is supported by the interactive installer which creates an appropriate addon-configuration file. For examples of CentOS installation as GuestOS refer to **ctys-uc-CentOS(7)** Once VMW is setup, the boot of the VM could be performed by the CREATE action of ctys. Basic Use-Cases for application are contained within the document **ctys-uc-VMW(7)**

9.2 Supported HOST-OSs

The VMW plugin is supported on all released runtime environments of the UnifiedSessionsManager where the products are available.

9.3 Supported GuestOSs

The native GuestOS support is the same as for the PMs and HOSTs plugins.

9.4 Supported Architectures

The whole set of available CPUs by Products is supported :

`x86, AMD64, x86_64`

9.5 Supported Interfaces

Current version supports the commandline interfaces of the products.

9.6 Supported VM Management Interfaces

The current version supports for basic management facilities by a vendor provided tools. These comprise mainly the creation of runtime entities and the cancellation of running instances a.k.a sessions. Library functions for vendor provided coding interfaces are due to the lack of actual requiremtn not utilized yet. The interface for the commandline based tools varies between the producty and versions. The additional requirements such as a valid account with appropriate permissions including the actual call interface may vary, e.g. for vmrun between Server-1.0.10 and Server-2.0.2. The following tools are utilized for now as required:

- vmware
- vmplayer
- vmware-cmd
- vmrun

9.7 Network Interconnection

NIC-bonding
The Installation of VMware is quite forward. Only some minor pitfalls occur for specific configurations with **NIC-bonding**. When a bonding device is utilized on Linux the **mode=6** is not supported, which is the ARP-negotiation of client and server machines. The success of the support could be easily checked when using a guest system and calling ping. The effect is the lost of about the half of the ping-answers. This is somewhat a pity, because the **mode=6** seems to be the fastest mode which even does not require support of intermediate network equipment. Any other mode seems to work properly.

9.8 Installation of Components

The provided examples are based - if not stated else - on CentOS-5.4, but may be applicable for any other distribution similar.

9.8.1 Server-1.0.10

The installation of VMware-Server(TM)-1.0.10 on CentOS-5.4 depends on the used kernel. The description is verified to be applicable to linux-2.6.29.

1. Download and install rpm of product from VMware(TM) Inc. e.g. VMware-server-1.0.10-203137.i386.rpm
2. Install new kernel, here linux-2.6.29.
3. Install init_mm patch for kernel and configure build-parameters. Set kernel-parameter for "Kernel-Hacking" activate "EXPORT UNUSED SYMBOLS" The patch could be downloaded from www.i4p.com.
4. Build kernel.
5. Install patches for "/usr/lib/vmware/modules/sources". The patch could be downloaded from www.i4p.com.
6. Reboot.
7. Configure vmware: vmware-config.pl

9.8.2 Server-2.0.2

The installation of VMware-Server(TM)-2.0.2 on CentOS-5.4 depends on the used kernel.

1. **Common**
   - Download and install rpm of product from VMware(TM) Inc. e.g. VMware-server-2.0.2-203138.x86_64.rpm
   - For CentOS-5.4 - install glibc-patch and edit "/usr/sbin/vmware-hostd" script. The patch could be downloaded from www.i4p.com.
   - Athentication with Kerberos
     For the following distributions the procedures are verified. But as far as known only the credentials/password are fetched form Kerberos. Currently - eventually due to knowledge-lack - missing the full SSO.
     The only file required be modified is '/etc/pam.d/vmware-authd'.
     – debian-5.0.0 - Kerberos Insert setting from '/etc/pam.d/sudo' **before** the standard contents.
       ```
       @include common-auth
       @include common-account
       ```
     – CentOS-5.4 - Kerberos Insert the following **before** the standard contents.
       ```
       auth      include system-auth
       account   include system-auth
       ```
   - Reboot.

- After creation of the first additional user within the VMware GUI, the counter/index for the next user has to be **once** incremented manually within the file '/etc/vmware/hostd/authorization.xml'.
  Else the creation of additional users fail with the error message expressing unavailability of the database.

2. **Standard kernel-2.6.18**
   - Configure vmware: vmware-config.pl
3. **New kernel-2.6.32.6** -Install, configure, and build the new kernel, here linux-2.6.32.6.
   - Install patches for "/usr/lib/vmware/modules/sources". The patch could be downloaded from www.i4p.com.
   - Configure vmware: vmware-config.pl
4. **Standalone Console**
   - Extract plugin for standalone console from Server installation and install this at "/opt/vmware/vmware-wmrc-x64". The patch could be downloaded from www.i4p.com.
   - Install "xdg-utils".

9.8.3 Player-1.0.5

Install the rpm package VMware-player-1.0.5-56455.i386.rpm for the standard kernel **linux-2.6.18-164.el5**.

9.8.4 Player-2.5.3

Install the rpm package VMware-player-1.0.5-56455.i386.rpm for the standard kernel **linux-2.6.18-164.el5**.

9.8.5 Player-3.0.1

Install the bundle-package VMware-Player-3.0.1-227600.x86_64.bundle for the standard kernel **linux-2.6.18-164.el5**.

9.8.6 Workstation-6.5.3

Install the bundle-package VMware-Workstation-6.5.3-185404.x86_64.bundle for the standard kernel **linux-2.6.18-164.el5** and execute vmware-config.pl.

9.8.7 Workstation-7.0.1

Install the bundle-package VMware-Workstation-Full-7.0.1-227600.x86_64.bundle* for the standard kernel **linux-2.6.18-164.el5**.

9.8.8 Standalone Console VMWRC

- Extract plugin for standalone console from Server installation and install this at "/opt/vmware/vmware-wmrc-x64". The patch could be downloaded from www.i4p.com.
- Install "xdg-utils".

9.9 Install Procedures

9.9.1 General Remarks

The most of the installation is performed with PXE if possible. Therefore the PXELINUX is switched to version 3.6.2 and a menu system for the management of the whole test-environment is setup. When difficulties occur due to specific network requirement, the CD-mount on ISO files option is used, which is almost in any case experienced to be quite safe for VMware products.

9.9.2 Installation and Maintenance by Product Console

For initial creation or basic maintenance the console of the specific product has to be started. The following options are available.

Product	Version	Available Console
Server	1.x	Proprietary
Server	2.x	Browser, VMWRC
Workstation	x	Proprietary, VNC
Player	-	-

The console could be started e.g. by usage of the X11 plugin. This is the case for the 2.x versions of the Server-Products.

```
ctys -t x11 \
     -a create=l:vmware,cmd:firefox%http:://127.0.0.1::8222 \
     delphi
```

The double column masks this character as a native to be bypassed, else it would be interpreted as a seperator of subarguments. The non-SSL port is required in some cases, where the other is not operable.

Any X11 program could be executed, e.g. the proprietary console by:

```
ctys -t x11 \
     -a create=l:vmware,cmd:vmware \
     delphi
```

Server-2.x

The example installation here is based on Server-2.0.2 and debian-5.0.0/amd64 installed by ISO image from a mounted local storage.

1. Start remote console by browser. Here as non-encrypted.

   ```
   ctys -t x11 \
        -a create=l:vmware,cmd:firefox%http:://127.0.0.1::8222 \
        delphi
   ```

   Here the non-encrypted local port is utilized, whereas the encrypted port is the default.

   It is recommended to use a the "Product Compatibility" for "Virtual Hardware" as "4" or "6". This assures the presence of a vmx-file.
2. Create a VM by 'Create Virtual Machine' command entry and perform the required steps. The following steps assume an ISO image as the install media.
3. Start the VM by calling:

   ```
   ctys -t vmw \
        -a create=l:debian-5.0.0,console:vmwrc,user:acue,\
                  p:/mntn/vmpool/vmpool01/vmw/templates/debian-5.0.0 \
        delphi
   ```

   This enters the native install procedure of the debian distribution.
4. Start installation by choosen install media.

9.9.3 PXE-Boot

The folowing steps are applied to an installation by PXE. This anyhow requires the proper setup of DHCP, TFTP, and one of HTTP/FTP/NFS. For some OSs a so calld **kickstart** file could be used to automate the whole procedure.

For Linux and BSD refer to [135, SYSLINUX] and [136, ETHERBOOT].

1. Create a VM by native means of a VMware product, but do not start it yet. When the base machine is created, close the VM.

   The common convention within ctys is, that the following items are all literally the same.
   - LABEL
   - DisplayName
   - <vmx-filenameprefix>
   - <directory-containing-vmx>
2. Edit the VMX file manually and apply following changes and addons:
   - Check "displayName" as mentioned before.

- Change the ethernet interface entries for the MAC-address and behaviour as required for PXE/DHCP.

3. The default behaviour is described as "generatedAddress", which could change and is somewhat challenging to be continuously maintained for PXE/DHCP. Therefore it should be changed to "static". The resulting entries depend on the actual product, but the essential entries seem to be common as for following example:

   - ethernet0.present = "TRUE"
   - ethernet0.addressType = "static"
   - ethernet0.address = "00:50:56:13:11:4D"

4. When the cache database is already populated by values from **/etc/dhcpd.conf** or a manually created database similar to **/etc/ethers**, the utility ctys-macmap could be used for management of address-mappings.

   - Change the UUID entries from dynamic behaviour to static values, otherwise they will change when the machine is reallocated. The values could be kept as already present, else should be generated by "uuidgen".
   - uuid.action = "keep"
   - uuid.location = "56 4d 66 ff 5a 76 d1 19-35 11 73 3d 0f 8d 26 9a"
   - uuid.bios = "56 4d 5e 88 71 0e a5 79-59 6c 34 15 44 a7 7e 96"

5. Add ctys-meta information as required for ENUMERATE. Additional values might be applied to the following example. The values are not recognized by VMware, thus has to be kept synchronous by the user. The main intention is to get cacheable information for off-line guests to be utilized by ENUMERATE and therefore by "ctys-vhost"

```
\end{enumerate}

#@#MAGICID-VMW

#@#VMSTATE="ACTIVE"
#@#SERNO="20080415051600"
#@#VERSION="01.01.001"
####MAC0 is provided by the eth0 of vmx-file!
#@#IP0="192.168.1.235"
#@#DIST="CentOS"
#@#DISTREL="5"
#@#OS="Linux"
#@#OSREL="2.6"
#@#CATEGORY="VM"
```

   (a) Close the file within the editor and open it again within the VMware frontend.
   (b) From now on the following steps could be already proceeded either by native call of "vmware", or by usage of the UnifiedSessionsManager, which has some advances for monitoring and LIST of current active sessions.

       The following example illustrates the call, when for a new machine the <machine-adress> is not yet registered within the cacheDB("ctys-vdbgen")

```
ctys -t vmw \
     -a create=p:\$HOME/vmware/tst-ctys/tst116/tst116.vmx,reuse \
     -c off \
     -C off \
     host1
```

- Start the VM and set the emulated BIOS appropriately, by entering with *F12*.
- Boot into PXE, which might be a simple or more advanced Menu, or just a command line for entering a boot string [135, SYSLINUX] .
- Install by means of current GuestOS and/or any generic installer.

9.9.4 ISO-Image and DHCP

The install procedure for usage of an ISO-Image is almos the same as for PXE, just a few formal differences apply.

- Add a CD/DVD-ROM-drive with a link to an ISO-Image, similar to an actual physical drive.
- Select the appropriate boot order, where instead or in addition to PXE the CD/DVD-drive has to be registered.

9.9.5 Remote Console for 2.x-versions - vmware-vmrc

The remote console plugin of the Server-2.0 versions could be started standalone too, thus integrates seamless as a graphic terminal into ctys.

The compressed plugin for various architectures is stored in the directory:

```
/usr/lib/vmware/webAccess/tomcat/apache-tomcat-6.0.16/webapps/ui/plugin
```

This directory contains the files:

- build_doNotErase.txt
- vmware-vmrc-linux-x64.xpi
- vmware-vmrc-linux-x86.xpi
- vmware-vmrc-win32-x86.exe
- vmware-vmrc-win32-x86.xpi

These can be uncompressed with **unzip**, the **plugins** directory is the only subdirectory required, which contains the executable **vmware-vmrc**. This requires the containing directory as base for relative search for shared libaries, thus should be the call directory for simplicity.

In ctys it is in current pre-configuration expected this directory to be renamed and located as one of the following directories. These are searched in the given order until first match.

(a) $MYADDONSPATH/vmware-rc-x64
The advance is here, that the whole package is distributed together with ctys when using **ctys-distribute** with a file copy mode.

(b) $HOME/vmware/vmware-rc-x64

(c) /opt/vmware/vmware-rc-x64

(d) /usr/bin
(e) /usr/local/bin
(f) /usr/share/vmware-rc
(g) /usr/share/vmware-rc-x64
(h) /usr/share/vmware-rc-x86
(i) /opt/bin
(j) /opt/vmware/vmware-rc

When calling ctys with the console type **VMWRC** now it should be considered whether the password will be provided by commandline or inserted into the dialogue mask when omitting. An alternative is the Single-Sign-On configuration.

The resulting call could be :

```
ctys -t vmw \
  -a create=l:tst502,reuse,user:root\%tst,console:vmwrc \
  -c local root@lab02
```

9.9.6 Supported/Tested Install-Mechanisms

The current version relies on the provided intstall mechanisms of the product supplier, and pre-requires an installed system.

9.10 Installation of GuestOS

9.10.1 CentOS

Installation is in general quite straight-forward. Here performed by PXE on a i386 machine with one CPU.

9.10.2 Debian-4.0_r3

Installation is quite straight-forward, once the required additional netboot-image is downloaded and in place. The differences and additions to the predefined environment are:

- Only the kernel image "linux" and the ramdisk "initrd.gz" are imported into a common boot environment with several UNIX variants as provided by [135, SYSLINUX].
- The following key has probably to be applied

```
debian-installer/allow\_unauthenticated=true
```

9.10.3 Debian-5.0.0

Refer to the example in chapter "Installation and Maintenance by Product Console".

9.10.4 Fedora 8

Installation is quite straight-forward and very similar to CentOS.

9.10.5 MS-Windows-NT-4.0-S

Installed from ISO image.

9.10.6 MS-Windows-2000-WS

Installed from ISO image.

9.10.7 OpenBSD

Installation is quite straight-forward, just the two-level boot has to be considered.

9.10.8 SuSE

Installation is quite straight-forward.

9.10.9 Installed Guest Systems

OS	**name**	**Inst-VM**	**Media**
CentOS-5.0	tst117	Server-1.x, 2.x	PXE,ISO
CentOS-5.1	tst112	Server-1.0.4	PXE,ISO
CentOS-5.2	tstxxx	Server-1.0.4	PXE,ISO
CentOS-5.3	tstxxx	Server-1.0.4	PXE,ISO
CentOS-5.4	tst131	Server-1.0.10, 2.0.2	PXE,ISO
CentOS-5.5	x	Server-2.0.2	PXE,ISO
Debian-4.0r3	tst106	Server-1.0.4	PXE,ISO
Debian-5.0.0	tst200	Server-1.0.10	PXE,ISO
Debian-5.0.0-amd64	debian-5.0.0	Server-2.0.2	ISO
Fedora 8	tst103	Server-1.0.4	PXE
Fedora 10	tstxxx	Server-1.0.10	ISO
Fedora 12	tst201	Server-1.0.10	ISO
FreeBSD-7.1	tst208	Server-2.0.2	ISO
FreeBSD-8.0	tst209	Server-2.0.2	ISO
Gentoo-2009	(tst231)	Server-2.0.2	ISO
Mandriva-2010-free	tst227	Server-2.0.2	ISO
Mandriva-2010	tst203	Server-2.0.2	ISO
OpenBSD-[2-4]	tstXYZ	WS5/6,Server-1.0.[1-5]	PXE,ISO
OpenBSD-4.0	tst109	Server-1.0.4	PXE
OpenBSD-4.2	tstxxx	Server-1.0.4	PXE
OpenBSD-4.3	tst155	Server-1.0.4	PXE
OpenBSD-4.6	tst207	Server-1.0.4	PXE
ScientifiLinux-5.4.1	tst204	Server-1.0.10	PXE,ISO
SuSE-9.3	tst003	WS6,Server-1.0.[345]	PXE,ISO
OpenSuSE-10.3	tst116	WS6,Server-1.0.[345]	PXE,ISO
OpenSuSE-11.2	tst205	WS6,Server-1.0.[345]	PXE,ISO
Solaris 10	tst115	WS6	ISO
OpenSolaris 2009.6	tst242	Server-2.0.2	ISO
Ubuntu-6.06.1-D	tst128	Server-1.0.4	ISO
Ubuntu-6.06.1-S	tst120	Server-1.0.4	PXE
Ubuntu-7.10-S	tst005	Server-1.0.4	PXE
Ubuntu-8.04-D	tst132	Server-1.0.4	ISO
Ubuntu-8.04-S	tst133	Server-1.0.4	PXE
Ubuntu-9.10-D	tst133	Server-2.0.2	ISO
Ubuntu-10.10-D	tstXYZ	Server-2.0.2	ISO
MS-Windows-2000WS	tstXYZ	WS4/5/6,Server-1.0.[1-5]	PXE,ISO
MS-Windows-2000S	tstXYZ	WS4/5/6,Server-1.0.[1-5]	PXE,ISO
MS-Windows-2003S	x	Server-2.0.2	ISO
MS-Windows-XP	tstXYZ	WS4/5/6,Server-1.0.10	PXE,ISO
MS-Windows-NT-4.0S	tstXYZ	WS4/5/6,Server-1.0.[1-5]	PXE,ISO

Table 9.1: Overview of Installed-VMW-VMs

Chapter 10

XEN Setup

.

10.1 XEN - Basics for Operations

The ctys-XEN plugin supports paravirtualized and hardware based VMs. Particularly a generic Python module is generated by the installer **ctys-createConfVM(1)** , which provides runtime dynamic re-allocation within filesystem and dynamic assembly of devices. The installer also copies the external runtime kernel into the VM directory, thus providing the usage in multiple distributions. Therefore a meta-layer for an abstract interface is defined, which is implemented by a wrapper script. The wrapper script is written in bash syntax and sourced into the runtime process, but could be used for native command line calls as well. The wrapper-script by default does not overwrite the preconfigured values of the Python module, which is read by the xm-call.

The main advance of using a wrapper script is the ability to perform dynamic scripting within the configuration file, which is standard bash-syntax with a few conventions. When activating the hardcoded CLI variable within the wrapper script the most of the parameters from the configuration file are superposed by command line variables.

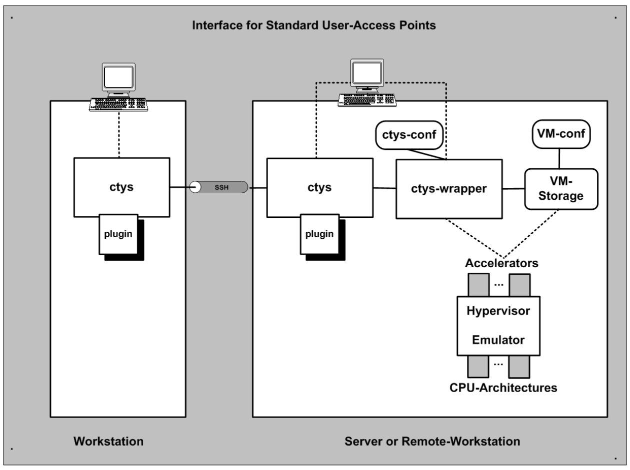


Figure 10.1: ctys distributed access

An installer for complete setup of a QEMU and/or KVM based VM is contained. The tool **ctys-createConfVM(1)** interactively creates a local or remote configu-

ration by detection of the actual platform and creation of a ready-to-use startup configuration.
The created configuration set comprises the generic wrapper-script, a specific configuration file, and some installation utilities such as a debootstrap-wrapper in case of debian installation. The actual installation of a GuestOS could be performed either by calling the wrapper-script or by calling ctys with the **INSTMODE**. When no additional suboptions are provided the generated defaults are used.

Some differences occur for the automated creation of required virtual HDDs. In case of offline-installers like debootstrap for simplicity reasons multiple virtual partitions in seperate image files will be created, whereas the OS based installation by default is performed by the partitioning of one virtual HDD.

Basic Use-Cases for application are contained within the document *ctys-uc-XEN(7)*.

10.2 Supported HOST-OSs

The current version is tested on: CentOS, debian, Fedora, OpenSUSE, ScientificLinux.

10.3 Supported GuestOSs

The following GuestOS are tested:
CentOS-5, debian-5.0, Fedora 8, OpenSolaris-2009.6, SL-5.4.1, SuSE-10.1, SuSE-10.2, SuSE-11.2, Ubuntu-8.04, and Ubuntu-9.10

10.4 Supported Architectures

The x86 platform fo 32bit and 64bit are supported with paravirtualized and hardware based virtualization environments.

10.5 Supported Interfaces

Any provided interface by Xen.

10.6 Supported VM Management Interfaces

The vm management is performed by xm and virsh.

10.7 Network Interconnection

The default is to provide one interface only, which could be any of the provided interfaces.

10.8 Installation of Components

The installation of the Dom0 is recommended to be based on a available distribution as provided if not required else. A quite straight forward installation could be set up by CentOS/RHEL-5, with additional updates The used from-the-box kernel is **2.6.18-8.1.15.el5.centos.plusxen**. For additional information refer to [74, CENTOS].

10.8.1 Xen-3.0.3

The installation of Xen depends on the used kernel. The installation of the standard kernel requires just the installation of the Xen kernel and modules.
Configure the kernel with

```
kernel: dom0_mem=1536M com1=115200,8n1
module: console=ttyS0,9600n8 console=tty1
```

10.9 Install Procedures

10.9.1 ctys-createConfVM

The current version provides a generic installer for all supported hypervisors. The functionality related to the plugin Xen includes the complete configuration of a VM, the creation of a generic wrapper-script, and some specific install utilities. The process could be performed either interactively by about 20 to 30 steps, or automatically, when appropriate default values within configuration files are in place. Refer to **ctys-createConfVM(1)** .

The provided install procedures are based on ISO images of CDROMs/DVDs and on network based installation, this comprises kickstart, debootstrap, and PXE. The network based installation requires one or more of the following services to be present.

```
DHCP + DNS
HTTP, FTP, NFS
TFTP
```

For additional services the following will be helpful:

- PXE/PXELinux
- Kerberos+LDAP+(Samba/SMB)+automount

10.9.2 Supported/Tested Install-Mechanisms

The actual applicable install mechanism partly depends on the host system, e.g. debootstrap is currently available on debian hosts only.

PARA Virtualization

OS	deboostrap	KS	PXE	CD	FD	HD	USB
CentOS-5.0	-	X	X	X			
CentOS-5.4	-	X	X	X			
Debian-4.0r3-ARM	-	-		X	-		
Debian-4.0r3		-	X	X			
Debian-5.0.0	X	-		X			
Fedora 8	-	X	X	X			
Fedora 10	-	X	X	X			
Fedora 12	-	X	X	X			
FreeBSD-7.1	-	-		X			
FreeBSD-8.0	-	-		X			
FreeDOS/balder	-	-		X			
Mandriva-2010	-			X			
ScientificLinux-5.4.1	-		X	X			
OpenBSD-4.0	-	-	X	X			
OpenBSD-4.3	-	-	X				
OpenSuSE-10.2	-	-	X	X			
OpenSuSE-11.1	-	-	X	X			
OpenSuSE-11.2	-	-		X			
Solaris 10	-	-		X			
OpenSolaris 2009.6	-	-		X			
Ubuntu-8.04-D		-	X	X			
Ubuntu-9.10-D		-		X			
uClinux-arm9	-	-	-		-	-	
uClinux-coldfire	-	-	-		-	-	-

Table 10.1: PARA Virtualization

HVM Virtualization

OS	deboostrap	KS	PXE	CD	FD	HD	USB
CentOS-5.0	-	X	X	X			
CentOS-5.4	-	X	X	X			
Debian-4.0r3-ARM	-	-		X			
Debian-4.0r3		-	X	X			
Debian-5.0.0		-	X	X			
Fedora 8	-	X	X	X			
Fedora 10	-	X	X	X			
Fedora 12	-	X	X	X			
FreeBSD-7.1	-	-		X			
FreeBSD-8.0	-	-		X			
FreeDOS/balder	-						
Mandriva-2010	-			X			
ScientificLinux-5.4.1	-	X	X	X			
OpenBSD-4.0	-	-	X	X			
OpenBSD-4.3	-	-	X				
OpenSuSE-10.2	-	-	X	X			
OpenSuSE-11.1	-	-		X			
OpenSuSE-11.2	-	-		X			
Solaris 10	-			X			
OpenSolaris 2009.6	-			X			
Ubuntu-8.04-D		-		X			
Ubuntu-9.10-D		-		X			
uClinux-arm9	-	-	-	-	-	-	-
uClinux-coldfire	-	-	-	-	-	-	-

Table 10.2: HVM Virtualization

10.10 Installation of Guests

10.10.1 CentOS-5

The given example is based on the following configuration:

- Host executing ctys-createConfVM on debian-5.0.0, no qemu installation.
- Target host for execution of Xen with Scientific Linux 5.4.1, present Xen installation.
- GuestOS within the Xen PARA virtualized VM is CentOS-5.4.

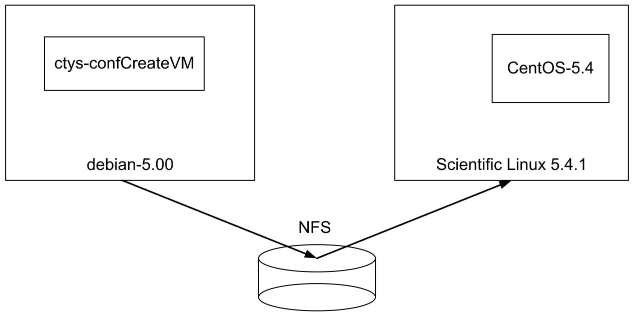


Figure 10.2: Distributed installation by 'ctys-confCreateVM

The 'cross-installation' for a different machine requires various options to be set for the target execution environment of the hypervisor which cannot be detected automatically on the local machine. One example is the actual support for the architecture.

Several of the parameters has to be set by environment VARIABLE based options. The available options could be visualized by the call option '–list-environment-variable-options' or for short '-levo'. The initial default values wre displayed too, which is not available for dependent values. E.g. by convention wihtin ctys the LABEL of a VM is the hostname of the conatined GuestOS, therefore the value of TCP/IP parameters could be determined only after the LABEL value is defined. The options by environment variable are applicable for local execution only, else the interactive dialogue is available only. The most value of the pre-defined values is gained in combination with oneof the options '–auto' or '–auto-all'. These pre-define confirmation answers for the interactive dialogues.

The following call creates in place of appropriate default values fully automated the complete set of configuration files and wrapper-scripts, including a basic kickstart file.

```
ACCELERATOR=PARA \
DIST=CentOS  RELEASE=5.4 \
OS=Linux     OSVERSION=xen-3.1.2-164.15.1.el5 \
ctys-createConfVM \
  -t XEN \
  --label=tst488 \
  --auto-all
```

When dropping the '–auto-all' option an interactive dialogue is performed, where no default values are required neccessarily. The environment variable based automation is available for local installation only, whereas the dialogue could be performed on a remote maschine.

The first step prepares the required infrastructure for starting the VM. The installation is foreseen to be performed by the facilities provided by the GuestOS. In case of CentOS the alternatieves PXE, CD/DVD, and KS(kickstart) are provided. The start of the intsallations could be executed by the following call.

```
ctys -t XEN -a create=l:tst488,b:<base-path-VM>,reuse,console:gterm,\
```

```
instmode:KS%default%HDD%default%init \
-c off \
tstUser@testHost'(-d pf)'
```

The **instmode** suboption initiates an installation, where the source and target paths are provided by default behaviour. The 'console:gterm' starts for the console a 'gnome-terminal', any supported type could be provided. The '-c off' deactivates the cache evaluation. The 'b:<base-path-VM>' suboption is a specific enhancement when working with NFS for reduction of filescans. The generic remote option '-d pf' is a debugger option, where 'pf' is a shortcut for 'printfinal', which traces all final assemblies of system calls just before execution. These calls are generally executable by cut-and-paste manually.

Alternatively either the direct call of the wrapper script or the Xen configuration file could be executed. E.g. by direct call of the generic Python script.

```
/usr/sbin/xm create /mntn/vmpool/vmpool05/Xen/test/tst-ctys/tst488/tst488.conf
  con=nongraphic
```

10.10.2 debian-5.0

(a) Add packages:
  i. bridge-utils
  ii. openssh-server
  iii. krb5
  iv. openldap
  v. vnc4server
  vi. vnc4viewer

(b) Configure udev and network

The following example shows the call for the update of configuration files for a previous installation of debian-5.0.0 PARA virtualized VM. Here the complete set of required variables is pre-set by environment variables, particularly a kernel is copied into the VM directory, where the origin is set to a non-standard PATH. The execution is performed in batch mode by usage of '–auto-all'.

```
DOMU_KERNEL=old/boot/vmlinuz-2.6.26-2-xen-amd64 \
DOMU_INITRD=old/boot/initrd.img-2.6.26-2-xen-amd64 \
INST_KERNEL=old/boot/vmlinuz-2.6.26-2-xen-amd64 \
INST_INITRD=old/boot/initrd.img-2.6.26-2-xen-amd64 \
ACCELERATOR=PARA \
DIST=debian \
DISTREL=5.0 \
OS=Linux \
OSREL=2.6.27.7-9-xen \
BRIDGE=eth1 \
MAC=00:0e:0c:36:a8:aa \
IP=172.20.1.78 \
ctys-createConfVM \
```

```
  -t XEN \
  --label=tst256 \
  --no-create-image \
  --auto-all
```

The following call starts the HVM based VM.

```
./tst256.sh --console=vnc
```

10.10.3 Fedora-8

Straight forward.

10.10.4 OpenSUSE-10.1

Somewhat tricky, dropped due to canceled support.

10.10.5 OpenSUSE-10.2

Somewhat tricky, dropped due to canceled support.

10.10.6 OpenSUSE-11.2

Standard-Installation on the PM, following the steps:

(a) Reconfigure the partition sizes for additional kernels
(b) Additional packages: Xen-Kernel, krb5, pam, cyrus-sasl.
(c) QEMU
    The configuration comprises:
(d) Eventually disable the firewall for a lab-machine.

10.10.7 Ubuntu-8.04

The following example shows the call for the update of configuration files for a previous installation of Ubuntu-8.04 as HVM on a debian-5.0.0 host. Here the complete set of required variables is pre-set by environment variables, and the execution is performed in batch mode by usage of '–auto-all'.

```
ACCELERATOR=HVM \
DIST=Ubuntu \
DISTREL=8.04 \
OS=Linux \
OSREL=2.6.27.7-9-xen \
BRIDGE=eth1 \
MAC=00:50:56:13:12:35 \
```

```
IP=172.20.2.53 \
ctys-createConfVM \
   -t XEN \
   --label=tst253 \
   --no-create-image \
   --auto-all
```

The local call of the wrapper script for the first test displays the resulting call.

```
./tst253.sh --console=vnc --print --check
```

The following call starts the HVM based VM.

```
./tst253.sh --console=vnc
```

10.10.8 Ubuntu-9.10

Standard-Installation on the PM, following the steps:

(a) Add vnc4, bridge-utils, openssh-server

10.10.9 Installed Guest Systems

OS	name	Inst-VM	Dom0	DomU
CentOS-5.0	tst100	Xen-3.0.2-PARA	X	X
CentOS-5.1	tst101	Xen-3.0.2-PARA	X	X
CentOS-5.4	tst228	Xen-3.0.2-PARA	X	X
CentOS-5.4	tst247	Xen-3.0.2-HVM	X	X
debian-5.0.0	tst256	Xen-3.0.2-PARA	X	X
debian-5.0.0	tst248	Xen-3.0.2-HVM	X	X
Fedora 8	tst114	Xen-3.0.2-PARA	X	X
OpenSolaris-2009.6	tst244	Xen-3.0.2-HVM		X
SL-5.4.1	tst258	Xen-3.0.2-HVM	X	X
SL-5.4.1	tst229	Xen-3.0.2-PARA	X	X
SuSE-10.1	tst153	Xen-3.0.2-PARA		(X)
SuSE-10.2	tst153	Xen-3.0.2-PARA		X
SuSE-11.2	tst250	Xen-3.0.2-PARA	X	X
Ubuntu-8.04	tst253	Xen-3.0.2-HVM	X	X
Ubuntu-9.10	tst252	Xen-3.0.2-PARA		X
Ubuntu-9.10	tst235	Xen-3.0.2-HVM		X

Table 10.3: Overview of Installed-Xen-VMs

10.11 Configuration Files

For additional information refer to configuration directory.

`$HOME/.ctys`

Chapter 11

XenServer Setup

.

11.1 XENS - Basics for Operations

The XenServer could be updated by standard mechanisms with yum based on the CentOS distribution. Thus the installation could be updated e.g. to the full scope of the plugins CLI, X11, VNC, and RDP, where for RDP the xrdp package has to be installed. This enables for the ease of administration by graphical desktops, whereas security is still assured by local UNIX domain socket access only with remote access through SSH tunnels. This is the standard interconnection facility provided by ctys.

11.2 Supported HOST-OSs

As provided by the product.

11.3 Supported GuestOSs

The whole set of provided OSs by the product.

11.4 Supported Architectures

The whole set of available CPUs by Products is supported :

`x86, AMD64, x86_64`

11.5 Supported Interfaces

ffs.

11.6 Supported VM Management Interfaces

ffs.

11.7 Network Interconnection

ffs.

11.8 Installation of Components

11.8.1 XenServer-5.5.0

ffs.

11.8.2 SSH-Access

ffs.

11.9 Install Procedures

ffs.

11.9.1 Supported/Tested Install-Mechanisms

The current version relies on the provided intstall mechanisms of the product supplier, and pre-requires an installed system.

11.10 Installation of GuestOS

ffs.

11.10.1 Installed Systems

OS	**name**	**Media**
CentOS-5.0	...	PXE,ISO

Table 11.1: Overview of Intsalled-VMW-VMs

Part III

Use-Cases for Plugins

Chapter 12

CLI Use-Cases

.

12.1 General

The CLI plugin itself is a pure command line interface, but due to the default activation of X11 forwarding any X11 command could be executed within a CLI session, thus any of the following examples could be used as a XTerm starter.

E.g. the interactive call of "xclock" will display correctly. This is particularly also true for the whole login-chain, when CLI is used for cascaded logins.

So, basically any firewall could be pierced in a secure manner by an SSH gateway in the DMZ. Which depends on the security facility of the gateway itself, of course. Particularly the "-Y" option which is mapped to the "-A" option of ssh for key-forwarding could be used to chain execution of X11 based applications.

HINT: Spaces within options, including suboptions, have to be masked by the reserved character '%'. This means just replace any SPACE with a '%' within any options/suboptions. The '%' itself is provided as '%%'.

12.2 Start a Local Interactive Session

This opens a second shell executed as login, almost the same as an ordinary shell call.

```
ctys -t CLI -a create=l:test
```

The "localhost" is hard-coded to behave as sub-shell call too.

```
ctys -t CLI -a create=l:test localhost
```

REMARK: Due to the implemented ambiguity-check for uniqueness of LABELs, only one localhost session is supported by the same label, when the label has to be non-altered, the usage of "-A 1" disables ambiguity-checks.

12.3 Start a Remote Interactive Session

This opens a second shell as a remote executed login.

```
ctys -t CLI -a create=l:test lab00
```

12.4 Execute a Remote Command

This opens a remote shell and executes the provided command only before termination. The connection will be kept open during the whole session, thus this is not executed in background mode by default.

```
ctys -t CLI -a create=l:test,cmd:uname\%-a lab00
```

The same forced to perform in background mode.

```
ctys -t CLI -a create=l:test,cmd:uname\%-a -b 1 lab00
```

12.5 Execute Multiple Remote Commands

The full scope of addressing of ctys is supported, thus the addressing of multiple targets, where each target could be a single host of a preconfigured hosts-group, is applicable. Intermixed addressing is supported too.

```
ctys -t CLI -a create=cmd:uname\%-a lab00 lab01
```

The same with parallel background execution:

```
ctys -t CLI -a create=cmd:uname\%-a -b 1 lab00 lab01
```

or

```
ctys -t CLI -a create=cmd:uname\%-a <host1> <group1> <host2> <group2>
```

The full scope of "include" files for group definitions and macros is applicable, thus e.g. tree-like reuse of groups could be applied.

Beginning with the current version path-based addressing of groups is supported, this allows for addressing like:

```
ctys -t CLI -a create dir0/subdir1/sbudir2/group-file
```

In combination with the enhanced features of "ctys-groups" for tree-views this allows for management of structured groups. Typical applications are the management of task-based destops for office applications, development environments, and the management of test-cases for Major-Projects.

Due to security reasons root-permission should be configured and handled properly, of course. It might be recognized that there is currently a chance (?) for users with appropriate skills and permissions to intercept the communications, when on the intermediate hops the message flow has to be re-encrypted after decryption.

The following call opens a session hop1 to lab01 via the intermediate relay lab00 by the session hop0.

```
ctys -t cli -a create=l:hop0cmd:ctys\%-t\%cli\%-a\%create=l:hop1\%lab00 lab01
```

The following call opens a session hop1 to lab01 via the intermediate relay lab00 by the session hop0 and starts a Xterm on lab01.

```
ctys -t cli -a create=l:hop0cmd:ctys\%-t\%cli\%-a\%create=l:hop1,cmd:xterm\%lab00 lab01
```

This approach is very similar to the equivalent usage of OpenSSH, and could be used in same manner to bypass routing as well as firewalls, when access and execution permissions on gateways are available.

REMARK: The utility "ctys-beamer" as a call-transformer eases the transformation of any call to a remote node via an arbitrary number of intermediate hops.

12.6 Start Xterm with tcsh

This call starts an interactive XTerm session running tcsh inside.

```
ctys -t cli -a create=l:tstcall,s:tcsh\%-c,cmd:xterm\%-e\%tcsh -b 1 lab00
```

12.7 Start gnome-terminal

This call starts an interactive gnome-terminal session running tcsh inside.

```
ctys -t cli -a create=l:tstcall,s:tcsh\%-c,cmd:gnome-xterminal\%-e\%tcsh -b 1 lab00
```

Chapter 13

RDP Use-Cases

.

13.1 General

The RDP plugin supports access to remote desktops by the RDP protocol. The access could be an application, terminal server, or hypervisor supporting the RDP protocol.

13.2 Start a Local Desktop Session

This opens a local session, where the server as well as the RDP client are executed locally.

```
ctys -t RDP -a create=l:tst1,RDPPORT:3389
```

The "localhost" is hard-coded to behave as a sub-shell call too, thus the following call is internally handled identical to the previous

```
ctys -t RDP -a create=l:tst1,RDPPORT:3389  \$USER@localhost
```

This case is called **DISPLAYFORWARDING** which is almost the same as the X11 display forwarding.

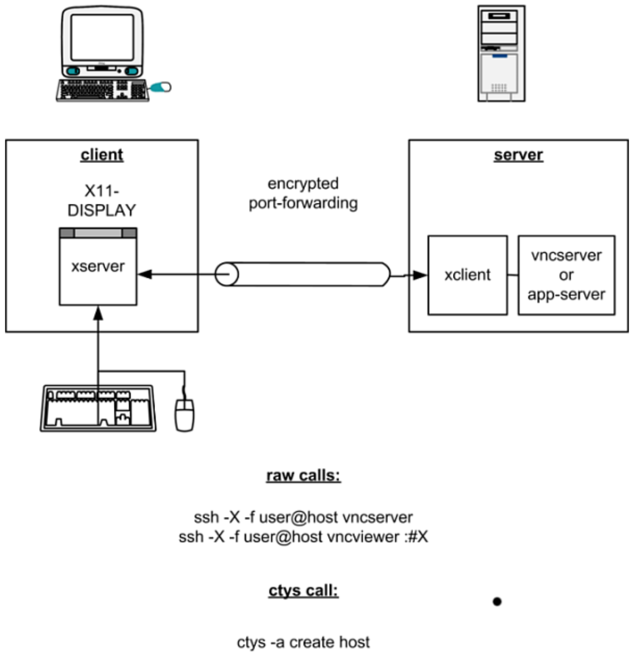


Figure 13.1: DISPLAYFORWARDING

13.3 Start a Remote Desktop with a Local Client

In case of a "Remote Desktop with Local Client" the server is running on the given <execution-target>, whereas the client is locally started on the caller's machine. This structure is called CONNECTIONFORWARDING and requires beneath the client and server processes a third, the connecting encrypted tunnel. The tunnel is established by means of OpenSSH and used as the local peer for the Client. This whole procedure of starting the processes and the establishment of the tunnel is controlled and preformed by ctys. The user has nothing else to do than setting the option **'-L CONNECTIONFORWARDING'** or for short **'-L CF'**.

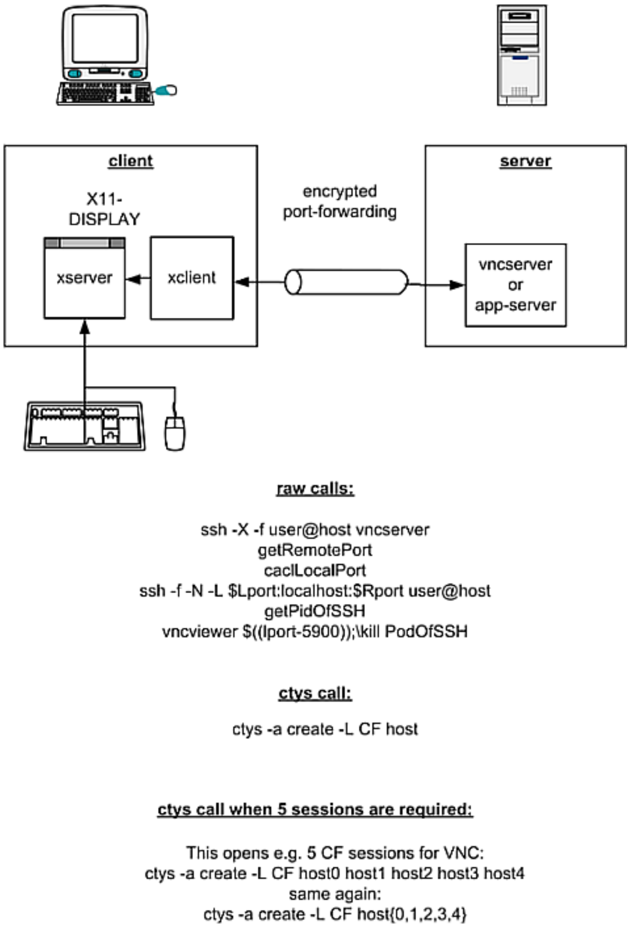


Figure 13.2: DISPLAYFORWARDING

The scenario performed behind the scene by ctys varies slightly from the previous. In case of CONNECTIONFORWARDING the whole process is set up in two steps.

(a) establishment of the encrypted tunnel

(b) start and connect the client process to the tunnel

The tunnel is established in the so called **one-shot mode**, where the connection is opened for an inital time period and closes automatically when the life-time threshhold is reached without an actual usage, or afterwards, when the client and server are disconnected. The period of the initial timeout for is defined by the variable "SSH_ONESHOT_TIMEOUT", which is by default set to 20seconds.

The following call starts a local client for a remote server.

```
ctys -t rdp -a create=l:tst -L CF lab00
```

The instances could be listed by the LIST action in several variants. The basic call with default selection executed on the caller workstation is:

```
ctys -t rdp -a list ws2
```

The standard assignment to LIST call is "tab_tcp,both", which displays:

```
TCP-container|TCP-guest|label |sesstype|c|user|group
\sout{---------}+\sout{-----}+\sout{--}+\sout{----}+-+\sout{--+-------}
ws2.soho     |-         |tst000|RDP     |C|acue|ldapusers
ws2.soho     |-         |tst001|RDP     |C|acue|ldapusers
ws2.soho     |ws2.soho.|ws2   |PM      |S|-   |-
ws2.soho     |-         |tst000|SSH(RDP)|T|acue|ldapusers
ws2.soho     |-         |tst001|SSH(RDP)|T|acue|ldapusers
```

Here the two tunnels could be identified as "sesstype=SSH(RDP)", and "c=T". This indicates, that the tunnels are created for the subsystem RDP with the session label "tst000" and "tst001".

The following call displays the same table, but with IDs instead of LABELs.

```
ctys -t rdp -a list=tab\_tcp,id ws2
```

Which results to the display:

```
TCP-cont|TCP-guest|id        |sesstype|c|user|group
\sout{----}+\sout{-----}+\sout{------}+\sout{----}+-+\sout{--+-------}
ws2.soho|-        |3389      |RDP     |C|acue|ldapusers
ws2.soho|-        |3390      |RDP     |C|acue|ldapusers
ws2.soho|-        |../pm.conf|PM      |S|-   |-
ws2.soho|-        |5950-3389 |SSH(VNC)|T|acue|ldapusers
ws2.soho|-        |5951-3390 |SSH(VNC)|T|acue|ldapusers
```

Indicating by the default ID of tunnels, that these are tunnels forwarding the ports "5950" to "3389" and "5951" to "3390".

The display could be changed as required by usage of specific free-customized tables, e.g. displaying LABEL and ID columns once.

The call with the whole set of involved machines as one call results to:

```
ctys -t rdp -a list=tab\_tcp,id ws2 lab00 lab01
```

```
ctys -t rdp -a list=tab\_tcp,id ws2 lab00 lab01

TCP-contai|TCP-guest|id         |sesstype|c|user|group
\sout{------}+\sout{-----}+\sout{------}+\sout{----}+-+\sout{--+-------}
ws2.soho  |-        |3389       |RDP     |C|acue|ldapusers
ws2.soho  |-        |3390       |RDP     |C|acue|ldapusers
ws2.soho  |-        |d/pm.conf  |PM      |S|-   |-
ws2.soho  |-        |5950-3389  |SSH(RDP)|T|acue|ldapusers
ws2.soho  |-        |5951-3390  |SSH(RDP)|T|acue|ldapusers
lab00.soho|-        |3784       |CLI     |C|acue|ldapusers
lab00.soho|-        |31206      |CLI     |C|acue|ldapusers
lab00.soho|-        |1          |VNC     |S|root|root
lab00.soho|-        |2          |VNC     |S|acue|ldapusers
lab00.soho|-        |           |XEN     |S|-   |-
lab00.soho|-        |e/xen/tst1|XEN     |S|-   |-
lab00.soho|-        |d/pm.conf  |PM      |S|-   |-
lab01.soho|-        |           |XEN     |S|-   |-
lab01.soho|-        |d/pm.conf  |PM      |S|-   |-
```

13.4 Start Remote Desktop Sessions by Native-RDP

This opens a remote session by using the RDP protocol via a remote connection to a boxed application or a terminal server. In this case actually the RDP client is attached 'from-outside' to an access port. This differs from the preferred 'localhost-access', where a pre-authorisation by SSH access is performed. Thus it is an exception to the common philosopy and therefore called 'INSECURE' alternatively to 'RDPHOST'.

The main application is the access to appliance-boxes when these provide an RDP access only, or to MS-Windows(TM) based OS.

```
ctys -t RDP -a create=l:tst1,RDPPORT:3389,RDPHOST:lab02
```

Same could be applied in a relay-configuration.

```
ctys -t RDP -a create=l:tst1,RDPPORT:3389,RDPHOST:lab02 lab05
```

Chapter 14

VNC Use-Cases

.

14.1 General

The VNC plugin supports access to remote desktops by the NFB protocol. The access could be either by combination of provided client and server programs to a native target, or by utilizing the client only either to an application or hypervisor supoorting the NFB protocol.

The automated signon when connecting a vncviewer to a vncserver requires a password as supported by vncpasswd. In order to avoid any user interaction for password requests the password is stored into the passwd file in **$HOME/.vnc** and is set to a default "install". This has to be changed once installed.

The default session type is **VNC**, thus the '-t vnc' option could be omitted within the following examples. The call

```
ctys -t VNC -a create=l:test
```

is identical to

```
ctys -a create=l:test
```

This behaviour could be changed within the configuration file 'ctys-conf.sh' by the variable 'DEFAULT_C_SESSIONTYPE'. For future safety of scripts despite the pre-set default the session type should be provided explicitly.

14.2 Start a Local Desktop Session

This opens a local session, where the VNCserver as well as the VNCviewer are executed locally.

```
ctys -t VNC -a create=l:tst1
```

The "localhost" is hard-coded to behave as a sub-shell call too, thus the following call is internally handeled identical to the previous

```
ctys -t VNC -a createl=l:tst1  \$USER@localhost
```

This case is called **DISPLAYFORWARDING** which is almost the same as the X11 display forwarding.

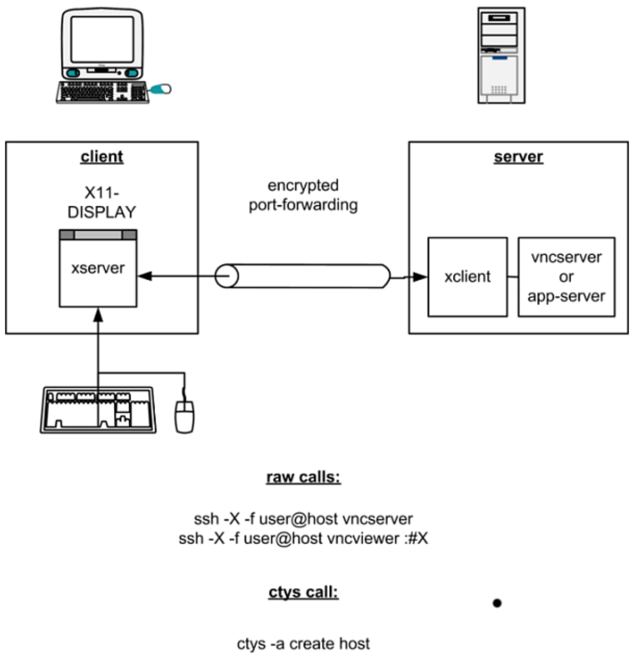


Figure 14.1: DISPLAYFORWARDING

14.3 Start a Remote Desktop Session

This call opens a remote desktop with DISPLAYFORWARDING, which is a coallocated VNCserver with a VNCviewer on the <execution-target>.

```
ctys -t vnc -a create=l:tst1 -L DISPLAYFORWARDINGF  <host>
```

The same could be written as:

```
ctys -t vnc -a create=l:tst1 -L DF lab00
```

The Client-Location "-L DISPLAYFORWARDING" is default for the original distribution, thus could be written as:

```
ctys -t vnc -a create=l:tst1 lab00
```

14.4 Start Bulk Desktop Sessions

This call opens 3 desktops on the remote host. The internal limit is set by default to 20.

```
ctys -t vnc -a create=bulk:3,l:tst lab00
```

The following call cancels all session by addressing their labels. The complete label is required here, which is an extended label by the incremental bulk-counter.

```
ctys -t vnc -a cancel=l:tst000,l:tst001,l:tst002 app2
```

The same function with usage of IDs.

```
ctys -t vnc -a cancel=i:2,i:3,i:4 app2
```

Current version supports as an implicit bulk addressing the keyword "ALL" only, which kills literally all VNC session where the appropriate permissions are available.

```
ctys -t vnc -a cancel=all app2
```

It should be recognized, that the CANCEL action is just a call to "vncserver -kill <display>" command, when this does not succeed, a "kill" will be placed. The clients are killed by UNIX-calls when required.
So the user is responsible for shutting down applications running within the CANCELed sessions.

14.5 Start a Remote Desktop with a Local Client

In case of a "Remote Desktop with Local Client" the server is started on the given <execution-target>, whereas the client is locally started on the caller's machine. This structure is called CONNECTIONFORWARDING and requires beneath the client and server processes a third, the connecting encrypted tunnel. The tunnel is established by means of OpenSSH and used as the local peer for the Client. This whole procedure of starting the processes and the establishment of the tunnel is controlled and preformed by ctys.
The scenario differs in all steps except the start of the server process from the previously described DISPLAYFORWARDING structure. In case of CONNECTIONFORWARDING the whole process is set up in three steps.

(a) start of server process
(b) establishment of the encrypted tunnel
(c) start and connect the client process to the tunnel

The tunnel is established in the so called **one-shot mode**, where the connection is opened for an inital time period and closes automatically when the life-time threshhold is reached, or afterwards, when the server disconnects. The period of the initial lifetime is defined by the variable "SSH_ONESHOT_TIMEOUT", which is by default set to 20seconds.

The following call starts a remote server with a local client.

```
ctys -t vnc -a create=l:tst -L CF lab00
```

REMARK: When the error message **Authentication Failure** is replied and no client window occurs, the reason is the differing passwd files of VNC. For the remote client by the option '-L CF' - ConnectionForwarding - the local passwd file of VNC has to contain the same password as the remote machine running the vncserver process.

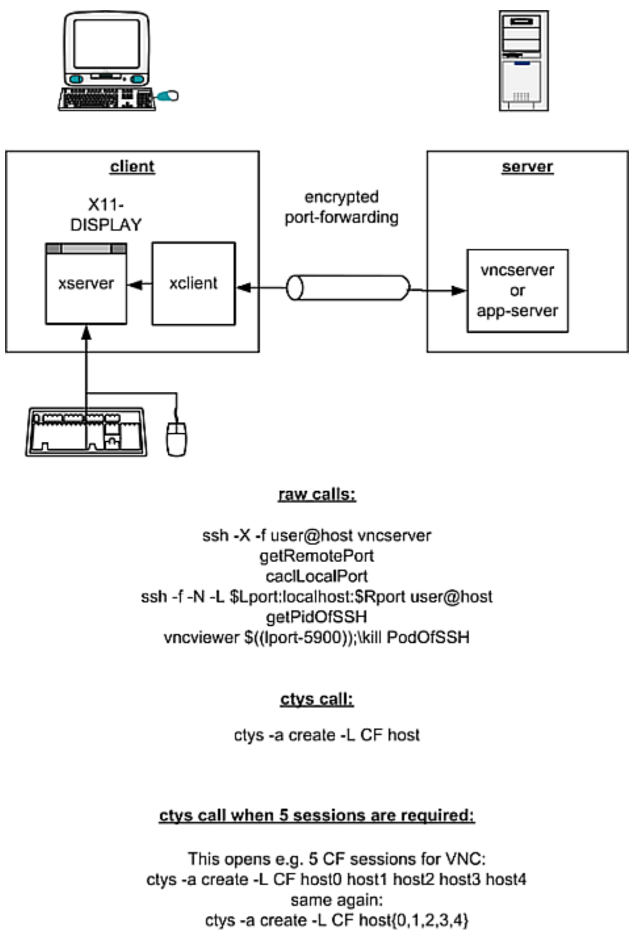


Figure 14.2: DISPLAYFORWARDING

The instances could be listed by the LIST action in several variants. The basic call with default selection executed on the caller workstation is:

```
ctys -a list ws2
```

The standard assignment to LIST call is "tab_tcp,both", which displays:

```
TCP-container|TCP-guest|label |sesstype|c|user|group
\sout{---------}+\sout{-----}+\sout{--}+\sout{----}+-+\sout{--+-------}
ws2.soho     |-        |tst000|VNC     |C|acue|ldapusers
ws2.soho     |-        |tst001|VNC     |C|acue|ldapusers
ws2.soho     |ws2.soho.|ws2   |PM      |S|-   |-
ws2.soho     |-        |tst000|SSH(VNC)|T|acue|ldapusers
ws2.soho     |-        |tst001|SSH(VNC)|T|acue|ldapusers
ws2.soho     |-        |tst000|VNC     |C|acue|ldapusers
ws2.soho     |-        |tst001|VNC     |C|acue|ldapusers
```

Here the two tunnels could be identified as "sesstype=SSH(VNC)", and "c=T". This indicates, that the tunnels are created for the subsystem VNC with the session label "tst000" and "tst001".

The following call displays the same table, but with IDs instead of LABELs.

```
ctys -a list=tab\_tcp,id ws2
```

Which results to the display:

```
TCP-cont|TCP-guest|id        |sesstype|c|user|group
\sout{----}+\sout{-----}+\sout{------}+\sout{----}+-+\sout{--+-------}
ws2.soho|-        |50        |VNC     |C|acue|ldapusers
ws2.soho|-        |51        |VNC     |C|acue|ldapusers
ws2.soho|-        |../pm.conf|PM      |S|-   |-
ws2.soho|-        |5950-5903 |SSH(VNC)|T|acue|ldapusers
ws2.soho|-        |5951-5904 |SSH(VNC)|T|acue|ldapusers
ws2.soho|-        |50        |VNC     |C|acue|ldapusers
ws2.soho|-        |51        |VNC     |C|acue|ldapusers
```

Indicating by the default ID of tunnels, that these are tunnels forwarding the ports "5950" to "5903" and "5951" to "5904".

The display could be changed as required by usage of specific free-customized tables, e.g. displaying LABEL and ID columns once.

The call with the whole set of involved machines as one call results to:

```
ctys -a list=tab\_tcp,id ws2 lab00 lab01
```

```
ctys -a list=tab\_tcp,id ws2 lab00 lab01

TCP-contai|TCP-guest|id        |sesstype|c|user|group
\sout{------}+\sout{-----}+\sout{------}+\sout{----}+-+\sout{--+-------}
ws2.soho  |-        |50        |VNC     |C|acue|ldapusers
ws2.soho  |-        |51        |VNC     |C|acue|ldapusers
ws2.soho  |-        |d/pm.conf |PM      |S|-   |-
ws2.soho  |-        |5950-5903 |SSH(VNC)|T|acue|ldapusers
ws2.soho  |-        |5951-5904 |SSH(VNC)|T|acue|ldapusers
lab00.soho|-        |3784      |CLI     |C|acue|ldapusers
lab00.soho|-        |31206     |CLI     |C|acue|ldapusers
lab00.soho|-        |1         |VNC     |S|root|root
lab00.soho|-        |2         |VNC     |S|acue|ldapusers
lab00.soho|-        |          |XEN     |S|-   |-
lab00.soho|-        |e/xen/tst1|XEN     |S|-   |-
lab00.soho|-        |d/pm.conf |PM      |S|-   |-
lab01.soho|-        |          |XEN     |S|-   |-
lab01.soho|-        |d/pm.conf |PM      |S|-   |-
```

Chapter 15

X11 Use-Cases

.

15.1 Xterm with Interactive bash

This opens a Xterm window with an interactive bash. Various consoles could be used, which are actually X-terminals such as Xterm ore gnome-terminal.

Due to the different usage of hyphens for the variuos graphical user interfaces the duboptions 'SH' and 'DH' - 'single hyphen' and 'double hyphen' were introduced. The "SH" suboption is here mandatory for the usage of Xterm, because the Xterm emulation requires a single-hyphen for it's options, default is "DH".

The default behaviour concerning the terminal emulation is to scan for a gnome-terminal first and prefer it if found, else an xterm emulation will be started by default.

```
ctys -t x11 -a create=l:tstx11 lab00
```

The same by call CONSOLE:XTERM alias.

```
ctys -t x11 -a create=l:tstx11,console:xterm lab00
```

The same by call args.

```
ctys -t x11 -a create=l:tstx11,cmd:xterm,sh lab00
```

The same by call args with explicit shell.

```
ctys -t x11 -a create=l:tstx11,cmd:xterm,c:bash\%-i,sh lab00
```

15.2 Gnome-Terminal with Interactive bash

This opens a Gnome-Terminal window with an interactive bash. The "DH" suboption is here mandatory but the default. This is required because the gnome-terminal

emulation requires a double-hyphen for it's options.
The the default behaviour is to open a gnome-terminal when detected.

```
ctys -t x11 -a create=l:tstx11 lab00
```

The same by call CONSOLE:GTERM alias.

```
ctys -t x11 -a create=l:tstx11,console:gterm lab00
```

The same as the default behaviour.

```
ctys -t x11 -a create=l:tstx11,dh lab00
```

The same as the default behaviour.

```
ctys -t x11 -a create=l:tstx11,cmd:gnome-terminal,c:bash\%-i,dh lab00
```

15.3 Emacs Session in shell-mode

Starts an remote Emacs in

```
ctys -t x11 -a create=l:tstx11,cmd:emacs\%-f\%shell,s:bash\%-i lab00
```

The same by call CONSOLE:EMACS alias.

```
ctys -t x11 -a create=l:tstx11,console:emacs lab00
```

15.4 Emacs Session with cd

Starts an remote Emacs:

```
ctys -t x11 -a create=l:tstx1,console:emacs,cd:/var/tmp app1
```

The result is:

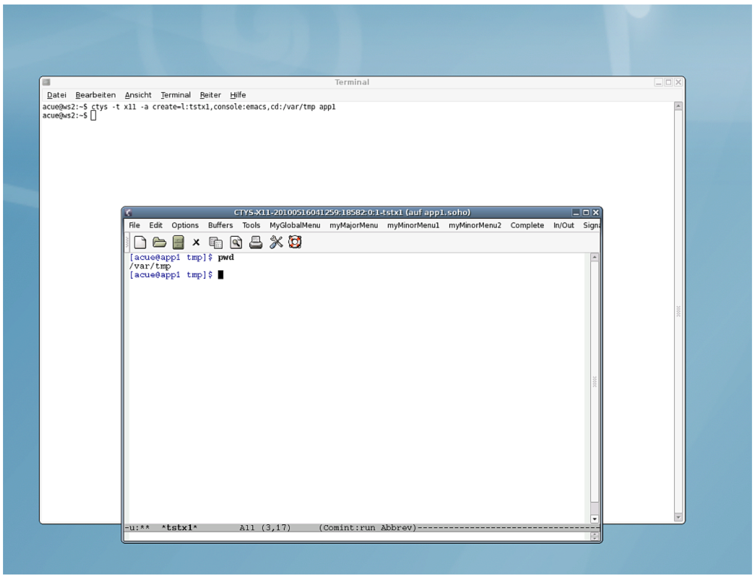


Figure 15.1: Emacs-Console with 'cd /var/tmp'

15.5 Single XClock

This starts an Xclock with an altered options keyword for windows title, additionaly the "SH" key is required for using single-hyphens.

```
ctys -t x11 -a create=l:tstx11,cmd:xclock,titlekey:name,sh lab00
```

Similar, but not the same. This starts an Xclock with NOTITLE, which suppresses the setting of the title for a window by the LABEL string. Thus this window is no longer recognized by LIST, or just with limits.

```
ctys -t x11 -a create=l:tstx11,cmd:xclock,notitle lab00
```

Chapter 16

VMWE - ESX(TM) Use-Cases

ffs.

Chapter 17

VMWE - ESXi(TM) Use-Cases

ffs.

Chapter 18

QEMU/KVM Use-Cases

.

18.1 Install and Configure a VM with ctys-createConfVM

The current version supports a new installer with minimal required remaining manual actions. The Installation process id described within the document **ctys-configuration-QEMU** for QEMU/KVM. This is basically a 2-stage-approach. The first stage is the call of the interactive tool **ctys-configuration-QEMU** for creation of a generic Wrapper-Script and an additional configuration file with appropriate values for most of practical cases. The second stage starts QEMU/KVM and begins the boot of the selected GuestOS from the configured bootmedia. This could be performed by several optional interfaces, either from the standard ctys call or by direct-execution of the wrapper-script.
The whole process is designed to be executed in a straight forward manner and should be prefered for the first steps instead of the following legacy-examples.

18.2 CREATE a session

The first tests and examples of the QEMU plugin are based on the "arm-tst" VM contained in the examples of QEMU. This is a ready to use VM, but the TCP/IP address is hardcoded to "10.0.0.2" thus might be required to be configured. The coldfire test VM contained in the QEMU examples supports DHCP, thus is ready to use within the network. Anyhow, for the first tests the actual usage of the network is not yet required. All following examples, if not stated else, rely on the provided configuration file "arm-test.ctys" and the QEMU VM "arm-test". These have to be installed as described within the examples chapter for Section **??** '**??**' on page **??** .

The first call now creates a session and starts the VM with VNC as a console which will be attached automatically.

```
ctys -t qemu -a create=p:\$HOME/qemu/tst/arm-test/arm-test.ctys,reuse,console:vnc lab00
```

When the **vde_switch** is not configured yet the following error message occurs:

```
Missing management socket for "vde\_switch"
   QEMUMGMT=/var/tmp/vde\_mgmt0.acue
Call: "ctys-vnetctl" on "lab00.soho"
```

The solution is simply to proceed as requested and just create the UNIX Domain sockets by the following call with root permissions:

```
ctys-steupVDE -u \$USER create
```

The call could be executed remote too:

```
ctys-steupVDE -u \$USER create root@lab00
```

The setup should be operational now. The support of the types of CONSOLEs depends on the actually implemented call within the "arm-tst.ctys" script, which is a shell script with a defined interface. The currently supported CONSOLE types by arm-test are: "CLI, SDL, VNC". The CLI and SDL types are supported as DISPLAYFORWARDING in synchronous mode only for this version.

The following call creates an SDL CONSOLE.

```
ctys -t qemu -a create=p:\$HOME/qemu/tst/arm-test/arm-test.ctys,reuse,console:sdl lab00
```

As might be expected, the following call creates a CLI CONSOLE.

```
ctys -t qemu -a create=p:\$HOME/qemu/tst/arm-test/arm-test.ctys,reuse,console:cli lab00
```

The monitor as configured within "arm-test.ctys" could be attache by usage of "netcat"

```
nc -U \${MYQEMUMONSOCK}
```

which could be generated by the function "netGetUNIXDomainSocket" and is derived from "QEMUMONSOCK" as raw-pattern, for additional information refer to the "arm-text.ctys" inline comments.

The QEMU monitor now could be entered by typing 'Ctrl-a'+'c'+'<RET>', the console is recovered by typing the same sequence again. For additional information refer to the QEMU User-Manual. A second terminal emulation to be used is the 'unixterm' command of VDE.

Alternatively EMACS could be used as a terminal emulation for CONSOLE access, either in "shell-mode" or in "ansi-term" mode. This work s the same way as an ordinary xterm session, where within the the "display-window" a cli is started connecting to a local UNIX domain socket. The socket has to be configured as a serial device within the GuestOS. For EMACS two additional variants exist for both modes, where the frame is divided into two windows, which connects the <execution-target> and the <machine-address> representing the GuestOS.

In the following example in the upper window a login-prompt of the GuestOS is displayed, whereas in the bottom window the "top" comman is shown for the hosting machine.

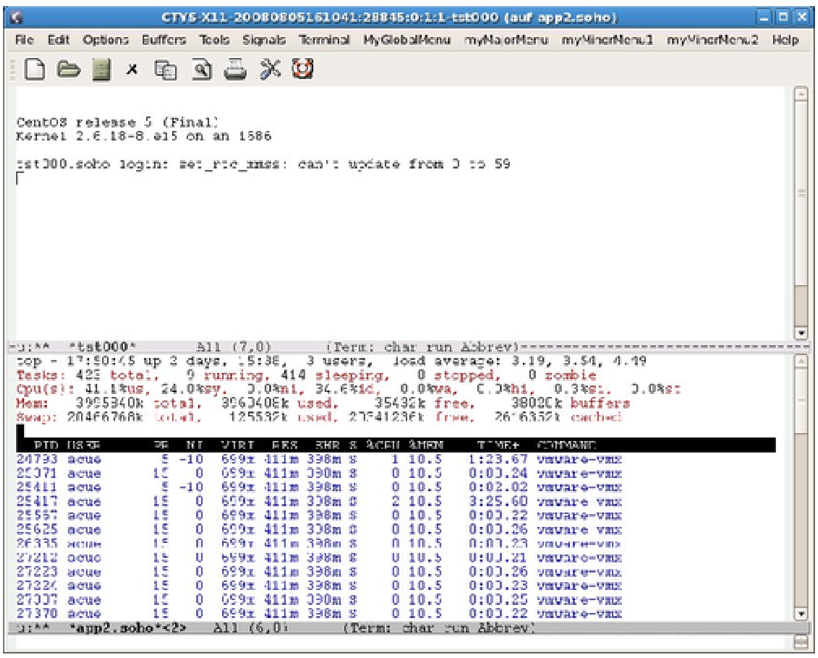


Figure 18.1: CONSOLE:EMACSAM for a QEMU Session

The console with pure CLI access could be combined with an VNC console allowing additional graphical access. This is particularly forseen, and will be offered soon, same as a debugging facility for GDB access to QEMU and to applications within the GuestOS.

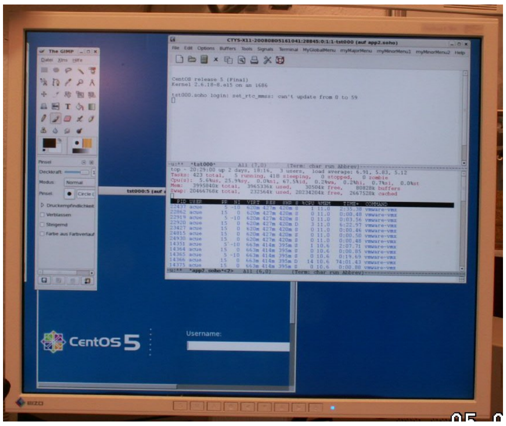


Figure 18.2: CONSOLE:EMACSAM and CONSOLE:VNC

18.3 CANCEL a session

The following call CANCELs the arm-test session.

```
ctys -t qemu -a cancel=p:\$HOME/qemu/tst/arm-test/arm-test.ctys,force,poweroff:0 lab00
```

18.4 LIST sessions

The following call LISTs all sessions:

```
ctys -a list lab00
```

resulting to:

```
TCP-container|TCP-guest  |label   |sesstype|c|user|group
\sout{---------}+\sout{-------}+\sout{----}+\sout{----}+-+\sout{--+-------}
lab00.soho   |-          |arm-test|VNC     |C|acue|ldapusers
lab00.soho   |-          |LAB00   |VNC     |C|root|root
lab00.soho   |-          |LAB00   |VNC     |S|root|root
lab00.soho   |tst109     |arm-test|QEMU-arm|S|acue|ldapusers
lab00.soho   |-          |Domain-0|XEN     |S|-   |-
lab00.soho   |lab00.soho.|lab00   |PM      |S|-   |-
```

The following call LISTs all sessions by usage of a specific LIST-MACRO for QEMU:

```
ctys macro:listconnpid lab00
```

Resulting in:

```
label   |stype   |c|DIS|cport|sport|pid  |PM        |TCP
\sout{----}+\sout{----}+-+\sout{-+---}+\sout{-}+\sout{-}+\sout{------}+\sout{--------}
LAB00   |VNC     |C|1  |     |     |18933|lab00.soho|
LAB00   |VNC     |S|1  |5901 |     |5642 |lab00.soho|
arm-test|QEMU-arm|S|17 |     |25704|25832|lab00.soho|
Domain-0|XEN     |S|   |     |     |     |lab00.soho|
lab00   |PM      |S|   |     |     |1    |lab00.soho|192.168.1.71
```

18.5 ENUMERATE sessions

The following call ENUMERATEs all stored session configurations within the sub-directory of the HOME.

```
ctys -t qemu -a enumerate=b:qemu/tst lab00
```

The following call displays a listing formatted as a table:

```
ctys -t qemu macro:listconn lab00
```

18.6 SHOW

The following call SHOWs dynamic data.

```
ctys -t qemu -a show lab00
```

18.7 INFO

Particularly the available capabilities for QEMU are displayed, which contains a list of all available CPUs and the related system boards.

```
ctys -t qemu -a info lab00
```

This leads to:

```
Node:lab00.soho  -  ctys(01\_04\_001A03)
   System      :Linux
   OS          :Linux
   RELEASE     :2.6.18-8.1.15.el5.centos.plusxen
   MACHINE     :x86\_64
   KERNEL\#CPU  :SMP-KERNEL
   CPU-INFO
     processor:0
         vendor\_id           :GenuineIntel
         cpu family          :6
         model               :22
         model name          :Intel(R) Celeron(R) CPU 420  @ 1.60GHz
         stepping            :1
         cpu MHz             :1599.853
         cache size          :512 KB

      Flags assumed equal for all processors on same machine:
         flags
            vmx(VT-x - Pacifica)    = 0
            svm(AMD-V - Vanderpool) = 0
            PAE                     = 1

   MEM-INFO
      MemTotal               :   523 M
      SwapTotal              :  2031 M

   SOFTWARE
      Mandatory:
         bash        :GNU bash, version 3.1.17(1)-release
                        (x86\_64-redhat-linux-gnu)
         gawk        :GNU Awk 3.1.5
         sed         :GNU sed version 4.1.5
         SSH         :OpenSSH\_4.3p2, OpenSSL 0.9.8b 04 May 2006
         top         :top: procps version 3.2.7

      Optional:
         wmctrl      :wmctrl is on this machine not available
         lm\_sensors  :sensors version 2.10.0 with libsensors version
                      2.10.0
         hddtemp     :hddtemp version 0.3-beta13

   PLUGINS     :QEMU CLI X11 VNC
      QEMU:        Plugin Version:01.01.001a00pre
                   Operational State:ENABLED
                   QEMU version:
                   ->QEMU-0.9.1
                      Magic-Number:QEMU\_091
                      Verified Prerequisites:
                      ->CLI-ValidatedBy(hookInfoCheckPKG)
                      ->X11-ValidatedBy(hookInfoCheckPKG)
                      ->VNC-ValidatedBy(hookInfoCheckPKG)
                      -><LocalClientCLI>
                      -><LocalClientX11>
                      -><LocalClientVNC>
                      -><LocalXserverDISPLAY>
                      -><delayedValidationOnFinalTarget>
```

```
                    -><QEMU-0.9.1>
                    ->\_/usr/local/bin/vde\_switch\_info-USER=
                         acue-ACCESS-PERMISSION-GRANTED
                    ->\_/usr/local/bin/unixterm\_info-USER=
                         acue-ACCESS-PERMISSION-GRANTED
                    ->\_/usr/local/bin/vdeq\_info-USER=
                         acue-ACCESS-PERMISSION-GRANTED
                    ->\_/usr/local/bin/vdeqemu\_info-USER=
                         acue-ACCESS-PERMISSION-GRANTED
                    -><QEMUSOCK=/var/tmp/vde\_switch0.acue\_info-USER=
                         acue-ACCESS-GRANTED>
                    -><QEMUMGMT=/var/tmp/vde\_mgmt0.acue\_info-USER=
                         acue-ACCESS-GRANTED>
                    -><CPU-Emulation:qemu-alpha>
                    -><CPU-Emulation:qemu-arm>
                    -><CPU-Emulation:qemu-armeb>
                    -><CPU-Emulation:qemu-cris>
                    -><CPU-Emulation:qemu-i386>
                    -><CPU-Emulation:qemu-img>
                    -><CPU-Emulation:qemu-m68k>
                    -><CPU-Emulation:qemu-mips>
                    -><CPU-Emulation:qemu-mipsel>
                    -><CPU-Emulation:qemu-ppc>
                    -><CPU-Emulation:qemu-ppc64>
                    -><CPU-Emulation:qemu-ppc64abi32>
                    -><CPU-Emulation:qemu-sh4>
                    -><CPU-Emulation:qemu-sh4eb>
                    -><CPU-Emulation:qemu-sparc>
                    -><CPU-Emulation:qemu-sparc32plus>
                    -><CPU-Emulation:qemu-sparc64>
                    -><CPU-Emulation:qemu-system-arm>
                    -><CPU-Emulation:qemu-system-cris>
                    -><CPU-Emulation:qemu-system-m68k>
                    -><CPU-Emulation:qemu-system-mips>
                    -><CPU-Emulation:qemu-system-mips64>
                    -><CPU-Emulation:qemu-system-mips64el>
                    -><CPU-Emulation:qemu-system-mipsel>
                    -><CPU-Emulation:qemu-system-ppc>
                    -><CPU-Emulation:qemu-system-ppc64>
                    -><CPU-Emulation:qemu-system-ppcemb>
                    -><CPU-Emulation:qemu-system-sh4>
                    -><CPU-Emulation:qemu-system-sh4eb>
                    -><CPU-Emulation:qemu-system-sparc>
                    -><CPU-Emulation:qemu-system-x86\_64>
                    -><CPU-Emulation:qemu-x86\_64>

CLI:             Plugin Version:01.01.001a02
                 Operational State:DISABLED

X11:             Plugin Version01.01.001a02
                 Operational State:DISABLED

VNC:             Plugin Version:01.02.001b01
                 Operational State:DISABLED
```

18.8 QEMU Examples

18.8.1 ARM

After installation of QEMU and VDE as described the utility **ctys-plugins** should be called for validation of the operational state of the QEMU installation. The following call verifies the different plugins operational states for server functionality.

```
ctys-plugins -T all -e
```

The client functionality could be verified with the call:

```
ctys-plugins -T all
```

Now, with a properly installed test environment from QEMU and the additional ctys call-scripts setup as described before, the following call should start the **arm-test** QEMU VM with and CONSOLE of type SDL.

```
ctys -t qemu -a create=f:qemu/tst/arm-test/arm-test.ctys,console:sdl lab00
```

In case of ambiguous filenames in the cacheDB e.g. due to multiple access paths on multiple nodes by NFS the following approaches could be applied

use "p:<pathname>"
When the full absolute path by **p:<pathname>** is provided, no local ambiguity may occur within the execution context. This is recommended for the first steps, because it does not require any additional action.
<machine-address> Additional entries may lead to non-ambiguity. This depends on the contents of the distributed caches and requires some knowledge of the system.

deactivate cacheDB:
Another quick solution is the disabling of any caching, therefore the options "-c off" and "-C off" could be set. This leads to a filesystem scan, which of course results in some performance degradation, which could be serious in case of deep filestructures with a "late match". The scan is performed by usage of the system utility **find**.

The supported CONSOLE types for the from-the-box "arm-test" VM are CLI, SDL, and VNC. Additional information is available as inline comment within the "arm-test.ctys" configuration from the

```
\$HOME/ctys/templates
```

directory. After this call an SDL terminal window should be opened. In case of networking problems the most common error is the forgotten call of ctys-vnetctl -u <USER> create.

18.8.2 Coldfire

```
ctys -t qemu -a create=f:/qemu/tst/coldfire-test-0.1/coldfire.ctys,console:cli lab00
```

Chapter 19

VBOX Use-Cases

.

19.1 General

This is an alpha version, where the post-install functionality is provided first. Thus the installation and configuration has to be performed by the provided means of the supplier for now.

19.2 Install and Configure a VBOX

For current version the standard facilities of VirtualBox(TM) has to be utilized.

19.3 CREATE a session

The following call starts a session:

```
ctys -t vbox -a create=l:vbox001,b:\$VMDIRPATH,console:vbox lab02
```

The filename option "b:" is used, which sets the root directory for the subtree to be scanned for the VM defined by the label parameter 'l:vbox001'. The evaluation could be processed from cacheDB and/or by scanning the filesystem on the execution target. The cacheDB is influenced by the option **'-c'**.

The console is of type VBOX, which is a tightly coupled synchronous console, where the execution states of client and server are correlated.

The directory VMDIRPATH could be any directory pointing to path containing the vdi-file and an optional ctys-file, either self or within the subtree..

19.4 CANCEL a session

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 vbox -a cancel=l:vbox001,poweroff:0 lab02
```

19.5 LIST sessions

The simple LIST call

```
ctys -a list lab02 ws2
```

produces the output:

```
TCP-container|TCP-guest        |label       |stype     |accel|c|user   |group
\sout{---------}+\sout{-------------}+\sout{--------}+\sout{-----}+\sout{-}+-+\sout{--}+\sout{------}
ws2          |-                |vbox001     |RDP       |-    |C|acue   |ldapusers
ws2          |ws2.soho.        |-           |-         |HVM  |S|-      |-
ws2          |ws2              |-           |-         |HVM  |S|-      |-
ws2          |-                |vbox001     |SSH(VBOX)|-    |T|acue   |ldapusers
lab02.soho   |-                |LAB02       |VNC       |-    |C|root   |root
lab02.soho   |-                |tstCF       |VNC       |-    |S|acue   |ldapusers
lab02.soho   |-                |LAB02chkusr |VNC       |-    |S|chkusr|ldapusers
lab02.soho   |-                |LAB02       |VNC       |-    |S|root   |root
lab02.soho   |08:00:27:D2:9E:D9|vbox001     |VBOX      |HVM  |S|acue   |ldapusers
lab02.soho   |-                |vbox004     |VBOX      |-    |S|root   |root
lab02.soho   |lab02.soho.      |-           |-         |HVM  |S|-      |-
```

This is the default case for two VMs - vbox001 and vbox004 - running on lab02 with CONNECTIONFORWARDING to ws2, and local only on lab02. The CONNECTIONFORWARDING mode is currently supported for any asynchronous console type. The intermediate SSH tunnel - here SSH(VBOX) is established automatically by the **-L CF** option.

The same call with the suboption 'machine' and 'titleidx'. The 'machine' suboption forces all available fields to be displayed. The 'tittleidx' displays the human readable names of the fields, including the canonical field index for selection, e.g. within custom tables.

```
ctys -a list=machine,titleidx lab02 ws2
```

The output is simply formatted as a text-string to the screen width, thus without consideration of the content.

```
ContainingMachine(1);SessionType(2);Label(3);ID(4);UUID(5);MAC(6);TCP(7);DISPLA
Y(8);ClientAccessPort(9);ServerAccessPort(10);PID(11);UID(12);GUID(13);C/S-Type
(14);JobID(15);IFNAME(16);RESERVED1(17);CONTEXTSTRG(18);EXECPATH(19);HYPERRELRU
N(20);ACCELERATOR(21);ARCH(22)
ws2;RDP;vbox001;;;;;;5950;;901;acue;ldapusers;CLIENT;;;;;/usr/bin/rdesktop;rdes
ktop-1.6.0.;;x86\_64
ws2;;;/etc/ctys.d/vm.conf;;00:23:54:2e:eb:96;172.20.1.70;;;;1;;;SERVER;bond0;;;
;Linux-2.6.26-ws2-deb-005-ws2-deb-005;debian-5.0;HVM;x86\_64
ws2;;;/etc/ctys.d/vm.conf;;00:23:54:2e:eb:96;;;;;1;;;SERVER;eth0;;;;Linux-2.6.2
6-ws2-deb-005-ws2-deb-005;debian-5.0;HVM;x86\_64
ws2;SSH(VBOX);vbox001;5950-5904;;;;;5950;5904;891;acue;ldapusers;TUNNEL;2010071
3081630:22795:0:1:1;;;;;;;
lab02.soho;VNC;LAB02;1;;;;1;;;27430;root;root;CLIENT;;;;;/usr/bin/vncviewer;Rea
lVNC-4.1.2;;x86\_64
lab02.soho;VNC;tstCF;3;;;;3;5903;;2213;acue;ldapusers;SERVER;20100712104607:190
37:0:1:1;;;;Xvnc;RealVNC-4.1.2;;x86\_64
lab02.soho;VNC;LAB02chkusr;2;;;;2;5902;;3267;chkusr;ldapusers;SERVER;2010071020
0016:31650:0:1:1;;;;Xvnc;RealVNC-4.1.2;;x86\_64
lab02.soho;VNC;LAB02;1;;;;1;5901;;6475;root;root;SERVER;;;;;Xvnc;RealVNC-4.1.2;
;x86\_64
lab02.soho;VBOX;vbox001;/mntn/vmpool/vmpool05/vbox/test/initial/vbox001/vbox001
.vdi;531f42f1-9c64-425e-bf88-319cbe592453;08:00:27:D2:9E:D9;;;5904;;5525;acue;l
dapusers;SERVER;;;;;/usr/lib/virtualbox/VBoxHeadless;VirtualBox-3.1.2;HVM;x86\_6
4
lab02.soho;VBOX;vbox004;;;;;;;;9920;root;root;SERVER;;;;;/usr/lib/virtualbox/VB
oxHeadless;VirtualBox-3.1.2;;
lab02.soho;;;/etc/ctys.d/vm.conf;;00:0E:0C:CF:5C:12;172.20.1.75;;;;1;;;SERVER;e
th0;;;;Linux-2.6.18-164.15.1.el5;CentOS-5.4;HVM;x86\_64
```

The formatted display of results could be arbitrarily varied by custom tables as shown in the following example. These could be either provided as a call parameter, or stored as MACROs persistently.

```
ctys -a list=machine,\
   tab\_gen:3\_label\_10\%\%2\_TYPE\_8\%\%1\_PM\_10\%\%5\_UUID\_12\%\%6\_MAC\_17\%\%7\_TCP\_13\
   lab02 localhost
```

The output results to:

```
label     |TYPE    |PM        |UUID        |MAC              |TCP
\sout{------}+\sout{----}+\sout{------}+\sout{--------}+\sout{------------}+\sout{---------}
vbox001   |RDP     |ws2       |            |                 |

\begin{center}\begin{tabular}{cc}
|ws2 & |00:23:54:2e:eb:96|172.20.1.70 \\
|ws2 & |00:23:54:2e:eb:96| \\
\end{tabular}\end{center}

vbox001   |SSH(VBOX|ws2       |            |                 |
LAB02     |VNC     |lab02.soho|            |                 |
tstCF     |VNC     |lab02.soho|            |                 |
LAB02chkus|VNC     |lab02.soho|            |                 |
LAB02     |VNC     |lab02.soho|            |                 |
vbox001   |VBOX    |lab02.soho|531f42f1-9c6|08:00:27:D2:9E:D9|
vbox004   |VBOX    |lab02.soho|            |                 |

\begin{center}\begin{tabular}{c}
|lab02.soho|            |00:0E:0C:CF:5C:12|172.20.1.75 \\
\end{tabular}\end{center}
```

19.6 ENUMERATE sessions

The following call displays the main identifier of the test-pool VMs. For additional information refer to Section 20.7 'Display of Available Sessions' on page 234 .

```
ctys -a enumerate=macro:TAB\_ENUM\_LST,b:/mntn/vmpool/vmpool05/vbox/test/initial lab02
```

The resulting display is:

```
label  |stype|TCP    |MAC    |UUID                                |ID
\sout{---}+\sout{-}+\sout{---}+\sout{---}+\sout{----------------------------}+\sout{-------------------
vbox004|VBOX |       |       |                                    |itial/vbox004/vbox004.vdi
vbox001|VBOX |       |       |531f42f1-9c64-425e-bf88-319cbe59|itial/vbox001/vbox001.vdi
vbox002|VBOX |       |       |                                    |itial/vbox002/vbox002.vdi
```

The same executed on a machine **with valid VirtualBox(TM)** installation and set 'machine' parameters.

```
ctys -a enumerate=machine,b:/mntn/vmpool/vmpool05/vbox/test/initial lab02
```

The resulting display is:

```
lab02.soho;VBOX;vbox004;/mntn/vmpool/vmpool05/vbox/test/initial/vbox004/vbox004.vdi;;;;;
;;;;;;;;;DISABLED;;;;;;;;;;VirtualBox-3.1.2;;;;;;;;;;;;;VERSION:VirtualBox-3.1.2;
lab02.soho;VBOX;vbox001;/mntn/vmpool/vmpool05/vbox/test/initial/vbox001/vbox001.vdi;531f
42f1-9c64-425e-bf88-319cbe592453;;;;5904;;;;;OpenBSD;;;;;DISABLED;;;;;;;;;;VirtualBox-3.
1.2;HVM;;;;;;;;x86\_64;;320;1;VERSION:VirtualBox-3.1.2;
lab02.soho;VBOX;vbox002;/mntn/vmpool/vmpool05/vbox/test/initial/vbox002/vbox002.vdi;;;;;
;;;;;;;;;DISABLED;;;;;;;;;;VirtualBox-3.1.2;;;;;;;;;;;;;VERSION:VirtualBox-3.1.2;
```

The same executed on a machine **without installed VirtualBox(TM)** and set 'machine' parameter.

```
ctys -a enumerate=machine,b:/mntn/vmpool/vmpool05/vbox/test/initial
```

The resulting display is:
The difference is the missing of any attribute where the usage of utilities contained within VirtualBox(TM) is required. This is basically the same case as the missing of access permissions.

```
ws2;VBOX;vbox004;/mntn/vmpool/vmpool05/vbox/test/initial/vbox004/vbox004.vdi;;;;;;;;;;;;
;;;DISABLED;;;;;;;;;;;;;;;;;;;;;;;;
ws2;VBOX;vbox001;/mntn/vmpool/vmpool05/vbox/test/initial/vbox001/vbox001.vdi;;;;;;;;;;;;
;;;DISABLED;;;;;;;;;;;;;;;;;;;;;;;;
ws2;VBOX;vbox002;/mntn/vmpool/vmpool05/vbox/test/initial/vbox002/vbox002.vdi;;;;;;;;;;;;
;;;DISABLED;;;;;;;;;;;;;;;;;;;;;;;;
```

19.7 Display of Available Sessions

Once the basic installation and setup is accomplished, first a file-scan based start of a VM should be performed. Therefore the root directory for scanned subtree should be set in order to reduce the actual scan duration. 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.
For further informatio refer to the CREATE action.

The next step - after successful installation and configuration of the UnifiedSessionsManager 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 scans a test group with 4VMs, where one has no access permission at all, thus is hidden.

```
ctys-vdbgen \
   --replace ctys-vdbgen \
   --replace \
   --cacheDB=/homen/acue/.ctys/db/vbox01 \
   --base=/mntn/vmpool/vmpool05/vbox/test/initial \
   lab0

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

The following call of "ctys-vhost" lists all available VMs with given constraints, in this case all instances of VBOX 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 -p /homen/acue/.ctys/db/vbox01/ -o  pm,label,ids .
```

The "pm", the "ids" and the "label" are displayed as result.
The additional string '.' is used as a awk-regexpr for any.

```
lab02.soho;vbox001;/mntn/vmpool/vmpool05/vbox/test/initial/vbox001/vbox001.vdi
lab02.soho;vbox002;/mntn/vmpool/vmpool05/vbox/test/initial/vbox002/vbox002.vdi
lab02.soho;vbox004;/mntn/vmpool/vmpool05/vbox/test/initial/vbox004/vbox004.vdi
```

19.8 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 -p /homen/acue/.ctys/db/vbox01/ -o  pm,label,ids,titleidx .
```

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);SSHport(27)
lab02.soho;vbox001;/mntn/vmpool/vmpool05/vbox/test/initial/vbox001/vbox001.vdi
lab02.soho;vbox002;/mntn/vmpool/vmpool05/vbox/test/initial/vbox002/vbox002.vdi
lab02.soho;vbox004;/mntn/vmpool/vmpool05/vbox/test/initial/vbox004/vbox004.vdi
```

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 \
   lab02
```

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{--+----------------------------}
lab02.s|vbox|/mntn/vmpool/vmpool05/vbox/tes
lab02.s|vbox|/mntn/vmpool/vmpool05/vbox/tes
lab02.s|vbox|/mntn/vmpool/vmpool05/vbox/tes
```

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 \
   lab02
```

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

```
PM          |label    |ID
\sout{-------}+\sout{-----}+\sout{--------------------------}
lab02.soho |vbox001  |st/initial/vbox001/vbox001.vdi
lab02.soho |vbox002  |st/initial/vbox002/vbox002.vdi
lab02.soho |vbox004  |st/initial/vbox004/vbox004.vdi
```

19.9 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" .

Chapter 20

VMW Use-Cases

.

20.1 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.

20.2 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)** .

20.3 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
```

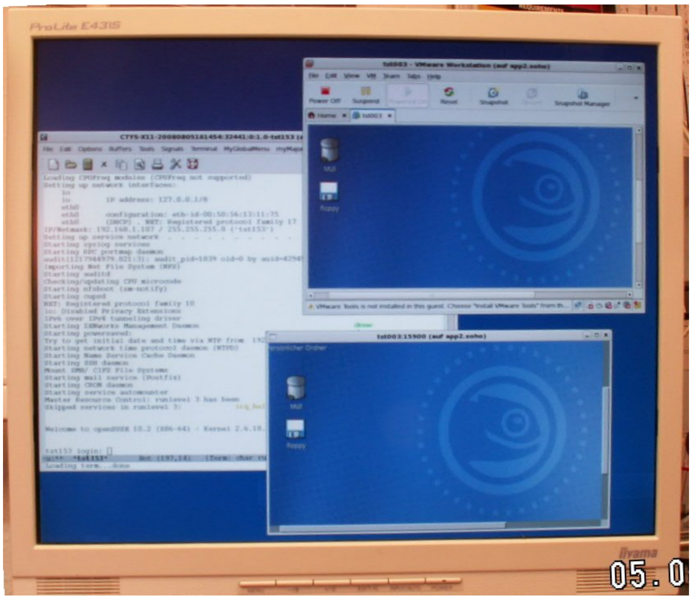


Figure 20.1: VMware WS6 with an additional VNCviewer Client Session.

20.4 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.

20.5 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|-   |-
```

20.6 ENUMERATE sessions

The following call displays the communications interfaces of the test-pool VMs. For additional information refer to Section 20.7 'Display of Available Sessions' on page 234 .

```
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
```

20.7 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 UnifiedSessionsManager 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 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
```

20.8 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.

20.9 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" .

Chapter 21

XEN Use-Cases

.

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.

21.1 CREATE a session

21.1.1 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)'
```

21.1.2 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.

21.1.3 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)'
```

21.1.4 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.

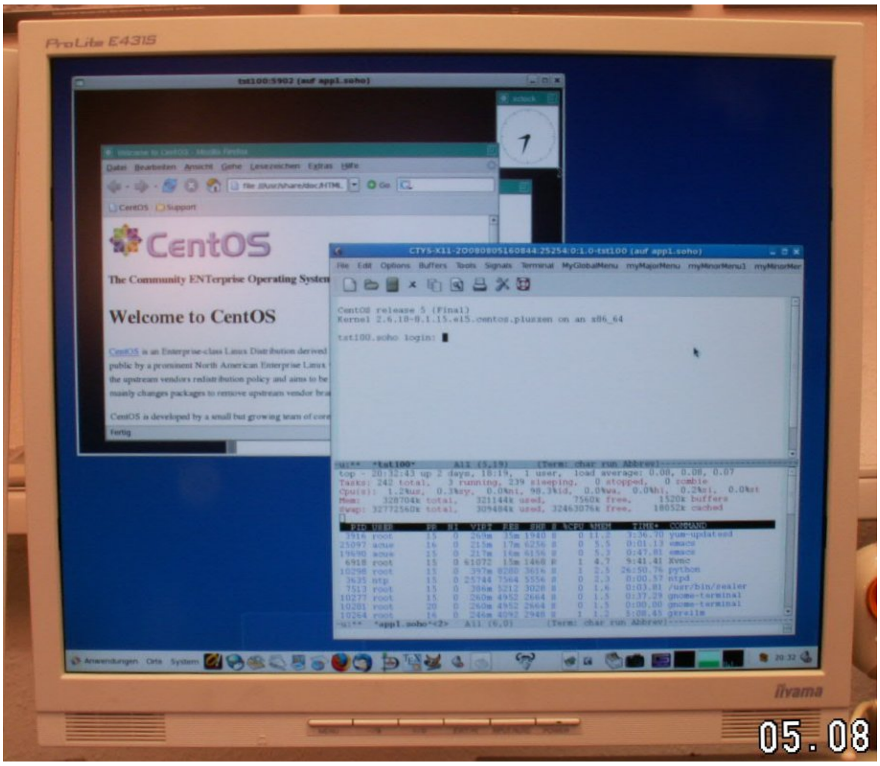


Figure 21.1: The "CONSOLE:EMACSAM" for a XEN Session

21.1.5 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|-    |-
```

21.1.6 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)'
```

21.1.7 RESUME from stat-file

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

21.1.8 RESUME with VNC

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

21.1.9 RESUME with EMACS

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

21.1.10 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 )'
```

21.2 CANCEL a session

21.2.1 CANCEL: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
```

21.2.2 CANCEL: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
```

21.2.3 CANCEL: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
```

21.2.4 CANCEL:PAUSE

Currently not yet available.

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

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

21.2.5 CANCEL:SUSPEND

Currently not yet available.

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

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

21.2.6 CANCEL: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
```

21.3 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 21.2: 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" .

21.4 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 21.3: TAB_CPORT by ENUMERATE

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

21.5 SHOW

Lists the dynamic global envvironement data on the target.

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

Same result with

```
ctys -a show lab00
```

21.6 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
```

Chapter 22

XenServer Use-Cases

ffs.

Chapter 23

PM Use-Cases

.

23.1 General

In addition to the provided examples additional man pages for specific use-cases are available. The most important are **ctys-WOL(7)** and **ctys-IPMI(7)**.

23.1.1 Resource-Checks

Due to the amount of required system tools and access permissions the utility **ctys-plugins** is introduced, which displays detailed reports for availability and required access permissions.

23.1.2 Configuration of Access Permissions

The plugin PM relays on a number of system resources, where numerous require by default root permissions.
These resources are mainly required in connection with CANCEL of a machine and in case of WoL/IPMI for the startup of a machine by CREATE.
The CREATE action is splitted into two security cases:

- Send WoL on local segment.
- Send WoL to a remote segment

For the packet distribution to the local segment the tool "ethtool" is used which requires root permissions on the interface to be used. For the remote distribution an own script is utilized, which does not require any specific permissions on the sending host, but some preparation on routers connecting the target segment.
The CANCEL action is only executed on the target machines, but requires on these several system facilities. The minimal requirement are the "shutdown-tools" on machines with non-bridged interfaces. On machines with bridged interfaces without an additional WoL interface some facilities for controlling the bridge from 'bridge-utils' are eventually required.
The following access permissions are needed for full functional scope of the current release of ctys. For standard machines without the neccessity of bridge shutdown for setting "wol g":

- /sbin/halt
- /sbin/reboot

- /sbin/poweroff
- /sbin/init
- /sbin/ethtool
- /sbin/ether-wake

For machines with bridges required to be shutdown for setting "wol g", typically Xen-3.0.2, the following additional permissions have to be configured:

- /usr/sbin/brctl
- /sbin/ip
- /sbin/ifup
- /sbin/ifdown
  For Xen the following tools with root-permissions are required:
- /usr/sbin/xm
- /usr/bin/virsh

The permissions should be configured as described in the related chapters.

Due to the timeout behaviour of ksu and sudo during probing when no user is configured, the default behaviour is not to probe for theese tools.

Additional information is provided in the example for handling WoL.

23.2 CREATE a Session

23.2.1 CREATE with WoL or RESUME

This will switch on a machine previously which is shutdown with WoL-Attribute. The execution host for ctys tasks is "lab01", so root permission as required has to be granted on "lab01" only.

RESUME is internally mapped to WoL, thus the same.

```
ctys -t pm -a create=wol,t:hostX,broadcast:hostXDirectedCast lab01
```

23.2.2 CREATE with CONSOLE

This will create a new session to a running machine with any of the CONSOLEs: CLI, XTERM, GTERM, EMACS, or VNC.

```
ctys -t pm -a create=CONSOLE:cli
```

23.3 CANCEL a Session

23.3.1 CANCEL:POWEROFF

This cancels anything running on top of the PM, but by default not the PM itself. Thus the whole VM stack running on top of the PM will be powered off by recursive stack handling.

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

The suboption FORCE just calls the lowest VM hypervisors for immediate - non-stack - poweroff. Contained upper VMs might not be able to store cached data, thus could be in erroneous state after switching them off abruptly.

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

The suboption WOL without a BROADCAST suboption sends a broadcast packet for WoL into local segment only. Therefore the first interface - which could be a bonding device - is used. When this has to be altered, the broadcast parameter has to be supplied.

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

The same with a "directed-broadcast".

```
ctys -t xen -a cancel=l:tst100,POWEROFF,wol,broadcast:192.168.3.255 lab01
```

Directed broadcast is preferred here, due to support of the native OpenBSD based routers, and the internal-only usage. This could be seen as "somewhat secure", because of fine-grained and rigorous filtering rules in addition.

23.3.2 CANCEL:REBOOT

This call REBOOTs all VMs running on top of PM, but not the PM itself. In addition a LABEL is supported, which just names the current session for display purposes.
Therefore an appropriate stack-propagation is performed.

```
ctys -t pm -a cancel=l:tst001,REBOOT lab01
```

The following call reboots in addition the PM itself.

```
ctys -t pm -a cancel=l:tst001,REBOOT,SELF lab01
```

This call just reboots the PM, NO STACK-PROPAGATION is performed, thus OSs within upper VMs might be corrupted, at least require some kind of recovery mechanisms.

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

23.3.3 CANCEL:PAUSE,SUSPEND

Current version just remaps this to a POWEROFF, therefore the user has to support the only supported Wake-Up mechanism WoL, refer to POWEROFF with WoL.

23.3.4 CANCEL:INIT

INIT is a transparently mapped action, which is almost the same as a native init-call. The difference is the call-relocation to the execution-target machine only.

Therefore the caller is responsible for the match of the requested init level with additional attributes, e.g. a WoL entry might not make too much sense, when called with "init 3".

```
ctys -t PM -a cancel=l:tst100,INIT:0 lab01
```

23.4 PM - Using Wake-On-Lan

The WoL feature is described in detail within the document **ctys-uc-WoL(7)**.

23.5 PM - Using Intelligent Platform Management Interface - IPMI

The WoL feature is described in detail within the document **ctys-uc-IPMI(7)**.

Part IV

Use-Cases for Operating Systems

Chapter 24

Android

.

24.1 General

The current document shows the basic installation of Android, which is a Linux variant.

The following environment is used here:

- Debian-5.0.6 with VirtualBox-3.2.10
- CentOS-5.4 with kvm-83 / Qemu-0.9.1
- eeeDroid-1.6
  The current description is based on the edition for i386 architecture. Download the image:

  `androidx86/eeeDroid_2008-12-20_1843Z.img`
- UnifiedSessionsManager - ctys-01.11.011

.

24.2 Setup of Host-OS and Hypervisor

The installation for the following variants has to be performed by the appropriate standard setup of the HostOS and , which quite straight forward:

- Debian with VirtualBox
  Install the download version instead of the OSE edition shipped with the distribution.
- CentOS with QEMU/KVM
  Here the standard distribution is installed. Additional packages are vde2-2.2.3 and Qemu-0.12.2, which are build and installed to '/opt'. The vde2-2.2.3 package for network encapsulation requires a symbolic link

  ```
  ln -s /opt/vde2-2.2.3 /opt/vde
  ```

  The wrapper vde may not be required, when the Qemu support option is compiled in, but this is not yet widely the case. Thus vde2 is still utilized as standard.

24.3 Setup of the UnifiedSessionsManager

24.3.1 Install tgz BASE-Package + DOC-Package on Debian

(a) Unpack the tar-gzip-archive and apply the standard installation procedure, where the call has to be executed by typing the fully qualified absolute path when ambiguity could occur. This is due to automatic usage of consistent libraries for the install procedure.

```
ctys-distribute -F 2 -P UserHomeCopy root@lab02
```

(b) Open a Remote Shell by call of CLI plugin:

```
ctys -t cli -a create=l:tst137 root@tlab02
```

(c) Check the plugins states by calling ctys-plugins:

```
ctys-plugins -T all -E
```

24.3.2 Install rpm BASE-Package + DOC-Package on CentOS

The following steps are required for a RPM based setup on CentOS. The installation is relocatable, but located at '/opt', and installed locally by 'ctys-distribute'.

(a) Install BASE package.

```
rpm -i ctys-base-01.11.011.noarch.rpm
```

(b) Now install a a local version, here by copy. The PATH prefix is important here, particularly in case of updates. The path is resolved to it's actual path by eliminating any symbolic link, and used for consistent link of libraries.

```
/opt/ctys-01.11.011/bin/ctys-distribute -F 2 -P UserHomeCopy
```

(c) Next the menu is setup.

```
ctys-xdg --menu-create
```

(d) Now the help is available as eihter a Gnome or KDE menu. Alternatively could be called from the commandline.

24.3.3 Setup of the Gnome Menu

The setup of the Gnome Menu is quite simple, the contained tool **ctys-xdg(1)** sets up a standard menu by the call:

```
ctys-xdg --menu-create
```

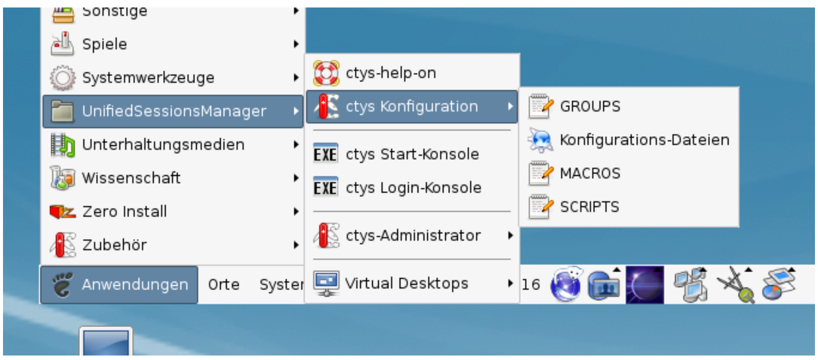


Figure 24.1: Create Menu

The setup could be targeted either for private menus or shared menus. Both setups are based on a menu template, which is stored in the configuration subdirectory 'xdg.d'. The call

```
ctys-xdg --menu-cancel
```

removes the installed files. For current version no checks for changed files is done. The menues could be edited and extended by the call

```
ctys-xdg --menu-edit
```

which opens the related directories for modification of '*.menu', '*.desktop', and '*.directory' files.

24.4 Creation and Installation

24.4.1 Creation and Installation on QEMU/KVM

The demo example VM is here named tst141, this is the hostname of GuestOS too.

(a) Login into the machine where QEMU/KVM is installed.
```
ssh -X ap2
```
(b) Change to the vmpool and create a directory and change into.
```
mkdir tst141
```
(c) Call the install and configuration utility for VMs. Here some values are set by environment variables, a complete list including the actually assigned values could be displayed by the option –**levo**.
```
ARCH=i386 \
DIST=Android \
DISTREL=1.6-r2 \
OS=Linux \
OSREL=2.6 \
ctys-createConfVM -t qemu --label=tst141
```

This call creates a virtual image(hda.img), the call-wrapper(tst140.sh), and the configuration file(tst140.ctys). The files are created from templates by assigning configuration values either from pre-configured default values, or interactive variation.

The resulting parameters are:
```
Not all values require to be set, some will be requested later
by dialogue.
Thus it is not neccessary to have values assigned to the complete
displayed set.


Actually used sources for default values:
  no-marker  = Pre-Set value, either from defaults configuration,
               or by commandline.
  no-value   = Either requested by dialog later, or the defaults
               of the finally
```

```
                 called application are used.
  (c)          = Read from actual configuration file, e.g. vmx-file.
  (d)          = Read from database.
  (g)          = Dynamically generated.
  (h)          = Used from current host as default.
  (m)          = Received from mapping definitions.

Applicable modifications:
  blue         = By call option, defines dependency for others.
  green        = By environment, 'could be set almost independent'
                 from other values.
  cyan         = By miscellaneous facilities, but is dependent from
                 others.
                 E.g. LABEL defines by convention the network 'hostname',
                 thus the TCP/IP params.
                 This could ..., but should not be altered!

Most of the missing values will be fetched during actual execution
of this tool by dynamic evaluation.


                     VAR name:Initial Value

               C_SESSIONTYPE:QEMU
                       LABEL:tst141
                         MAC:00:50:56:13:11:69 (m)
                          IP:172.20.2.245 (m)
                      BRIDGE:
                        DHCP:
                     NETMASK:
                         TCP:
                     GATEWAY:

                      EDITOR:acue

                        UUID:ff81f9d8-ba06-4c90-a801-484ad4841b50 (h)

                        DIST:Android
                     DISTREL:1.6-r2
                          OS:Linux
                       OSREL:2.6

                        ARCH:i386
                 ACCELERATOR:KVM  (h)
                         SMP:
                     MEMSIZE:512
                  KBD_LAYOUT:de

                 STARTERCALL:/usr/libexec/qemu-kvm
                 WRAPPERCALL:tst141.sh
```

```
                DEFAULTBOOTMODE:HDD

              DEFAULTINSTTARGET:/mntn/vmpool/vmpool05/kvm/test/tst-ctys/...
                                ...tst141/hda.img
          HDDBOOTIMAGE_INST_SIZE:8G
     HDDBOOTIMAGE_INST_BLOCKSIZE:256M
    HDDBOOTIMAGE_INST_BLOCKCOUNT:32
       HDDBOOTIMAGE_INST_BALLOON:y

                DEFAULTINSTMODE:CD
                   INSTSRCCDROM:/mntn/swpool/UNIXDist/../miscOS/Android/raw/..
                                ...android-x86/android-x86-1.6-r2.iso
              DEFAULTINSTSOURCE:/mntn/swpool/UNIXDist/../miscOS/Android/...
                                ...raw/android-x86/android-x86-1.6-r2.iso
                    INST_KERNEL:
                    INST_INITRD:

                        VMSTATE:ACTIVE


Remember that his is a draft pre-display of current defaults.
No consistency-checks for provided values are performed at this stage.
Some missing values are evaluated at a later stage dynamically.
```

An alternate call for the installation is the remote execution:

```
ctys -t qemu \
 -a create=l:tst140,id:${TST140}/tst140.ctys,instmode,console:sdl\
 app2
```

This starts the same by transforming to the target host 'app2' and calling the previous wrapper script.

The resulting files in both cases are:

- tst141.ctys
- tst141.sh
- hda.img

(d) Once the set of files is created the virtual machine is prepared for startup. For some other systems complete installation routines are available, e.g. debian and CentOS. The current state could be checked now by the following call.

```
./tst141.sh --console=vnc  --vncaccessdisplay=47 --print --check
```

This shows the current resulting call:

```
###########################
#Display call             #
###########################


QEMU_VERSION      = "qemu-0.9.1-kvm-83-maint-snapshot-20090205"
```

```
QEMU_MAGIC        = "QEMU_091"
QEMU_ACCELERATOR = "KVM"


ctys-uc-AndroidNAME   = "tst141.sh"
               +->STARTERCALL     = /usr/libexec/qemu-kvm
               +->REALSTARTERCALL = /usr/libexec/qemu-kvm


#The resulting call is:   #
--->
eval "/opt/vde/bin/vdeq  /usr/libexec/qemu-kvm    \
  -net nic,macaddr=00:50:56:13:11:69,model=rtl8139 \
  -net vde,sock=/var/tmp/vde_switch0.acue  \
  -name "tst141" -vga cirrus -localtime  -k de -m 512  -cpu qemu32  \
  -serial mon:unix:/var/tmp/qemumon.tst141.21844.acue,server,nowait \
  -daemonize  -vnc :47 \
  -boot c /mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141/hda.img"
<---

EXECALL:/opt/vde/bin/vdeq /usr/libexec/qemu-kvm
  -net nic,macaddr=00:50:56:13:11:69,model=rtl8139
  -net vde,sock=/var/tmp/vde_switch0.acue
  -name "tst141"
  -vga cirrus
  -localtime
  -k de
  -m 512
  -cpu qemu32
  -serial mon:unix:/var/tmp/qemumon.tst141.21844.acue,server,nowait
  -daemonize
  -vnc :47
  -boot c
  /mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141/hda.img
```

The installation is slightly different due to boot from install media.

```
./tst141.sh --console=vnc  --vncaccessdisplay=47 --print --instmode --check
###########################
#Display call             #
###########################


QEMU_VERSION      = "qemu-0.9.1-kvm-83-maint-snapshot-20090205"
QEMU_MAGIC        = "QEMU_091"
QEMU_ACCELERATOR = "KVM"


ctys-uc-AndroidNAME   = "tst141.sh"
```

```
               +->STARTERCALL     = /usr/libexec/qemu-kvm
               +->REALSTARTERCALL = /usr/libexec/qemu-kvm


#The resulting call is:   #
--->
eval "/opt/vde/bin/vdeq  /usr/libexec/qemu-kvm \
 -net nic,macaddr=00:50:56:13:11:69,model=rtl8139 \
 -net vde,sock=/var/tmp/vde_switch0.acue  \
 -name "tst141" -vga cirrus -localtime -k de -m 512 -cpu qemu32 \
 -serial mon:unix:/var/tmp/qemumon.tst141.23708.acue,server,nowait \
 -daemonize -vnc :47 -boot d \
 -cdrom /mntn/swpool/UNIXDist/../miscOS/Android/raw/...
       ...android-x86/android-x86-1.6-r2.iso \
 -hda /mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141/hda.img "
<---

EXECALL:/opt/vde/bin/vdeq /usr/libexec/qemu-kvm
  -net nic,macaddr=00:50:56:13:11:69,model=rtl8139
  -net vde,sock=/var/tmp/vde_switch0.acue
  -name "tst141"
  -vga cirrus
  -localtime
  -k de
  -m 512
  -cpu qemu32
  -serial mon:unix:/var/tmp/qemumon.tst141.23708.acue,server,nowait
  -daemonize
  -vnc :47
  -boot d
  -cdrom /mntn/swpool/UNIXDist/../miscOS/Android/raw/...
         ...android-x86/android-x86-1.6-r2.iso
  -hda /mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141/hda.img
```

The actual call starts the VM and displays the following screen.

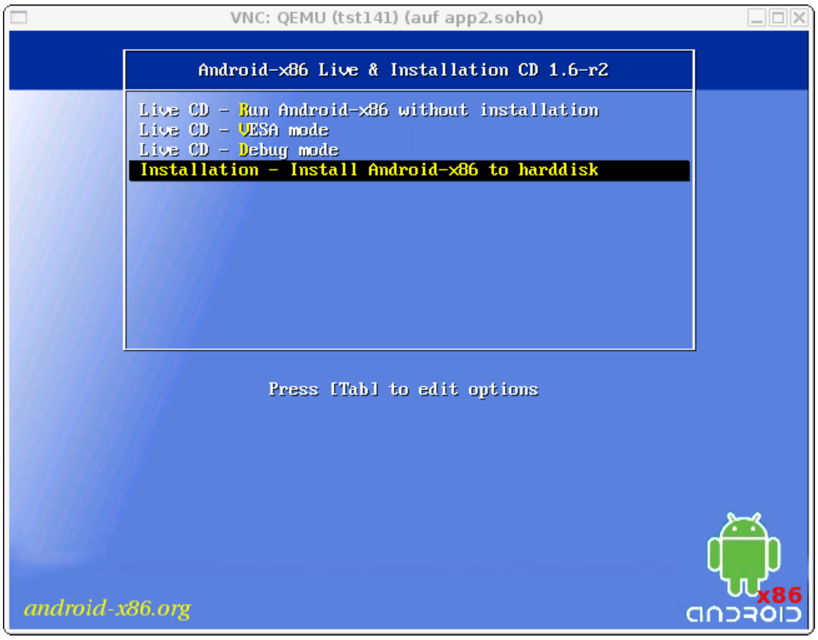


Figure 24.2: Install Menu on QEMU/KVM

The install procedure just installs here a life system on disk, thus proceeds quite fast. After the installation unmount the install media and boot into Android.

(e) In order to reboot just shutdown and boot again without the 'instmode' option. The shutdown could be proceeded by the 'quit' command within the monitor. The **monitor mode** is entered e.g. by **Ctrl-Alt-2**. One possible boot call for SDL console is:

```
ctys -t qemu \
   -a create=l:tst141,id:${PWD}/tst141.ctys,console:sdl \
   app2
```

The next starts with VNC console, which is default:

```
ctys -t qemu \
   -a create=l:tst141,id:${PWD}/tst141.ctys,console:vnc \
   app2
```

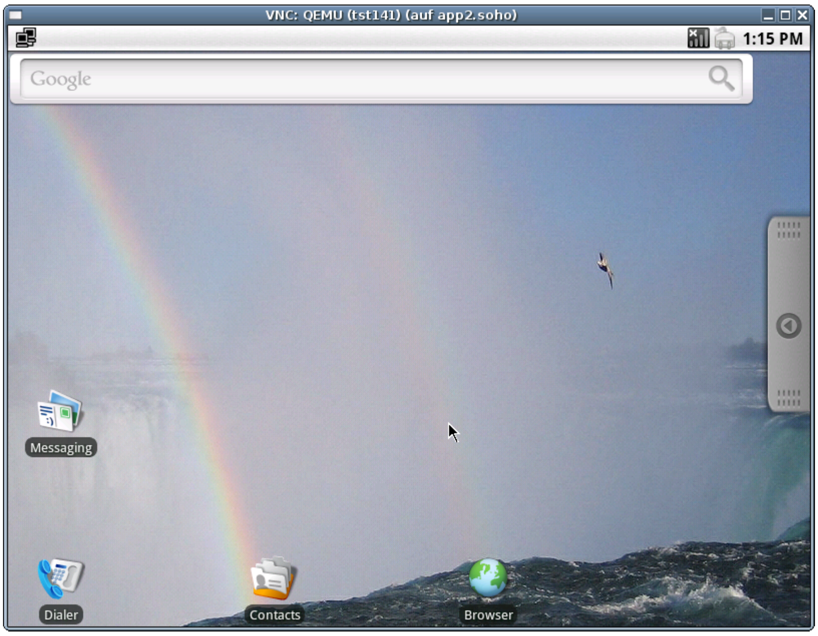


Figure 24.3: Install Menu on QEMU/KVM

When standard options are used the VM crashes when the screensaver is activated. Two workarounds are possible, first deactivating ACPI, second deactivating the screensaver. Here both are applied.

The following deactivates the screensaver - here called 'Screen timeout'. The menu order is:

`Settings -> Sound&display -> Screen timeout -> Never timeout`

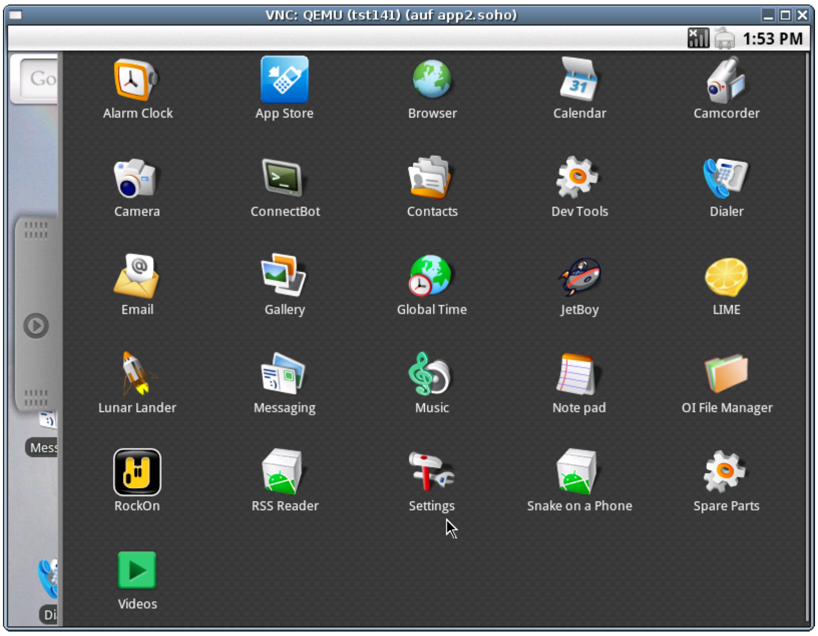


Figure 24.4: Deactivate screensaver - 01

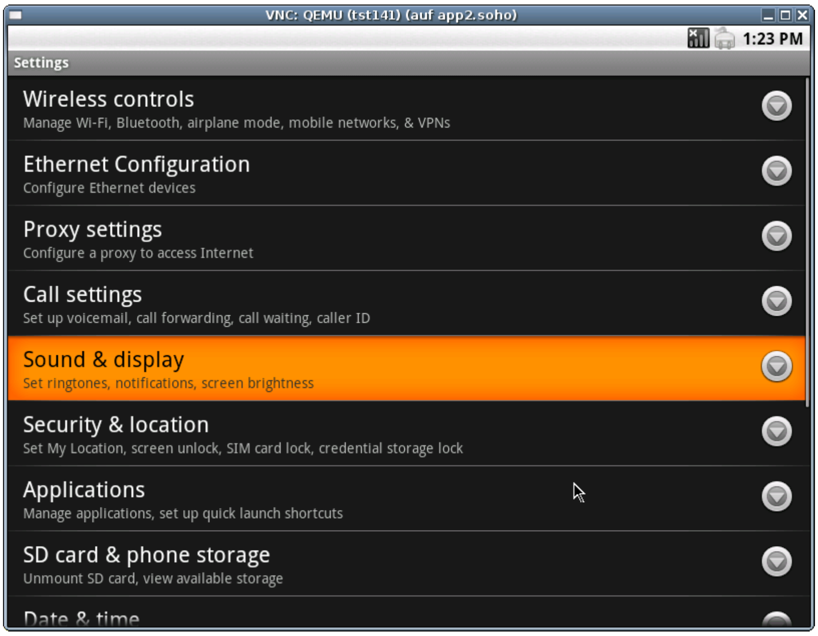


Figure 24.5: Deactivate screensaver - 02

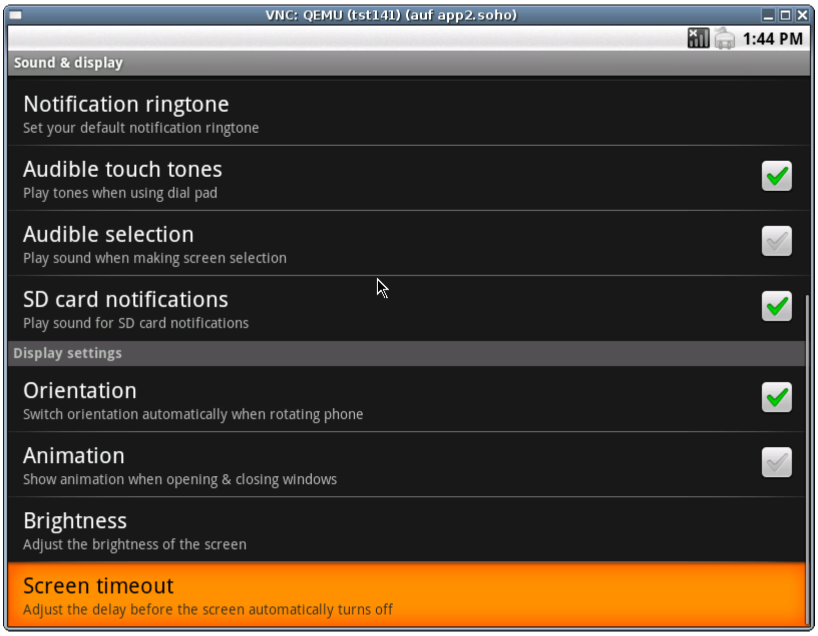


Figure 24.6: Deactivate screensaver - 03

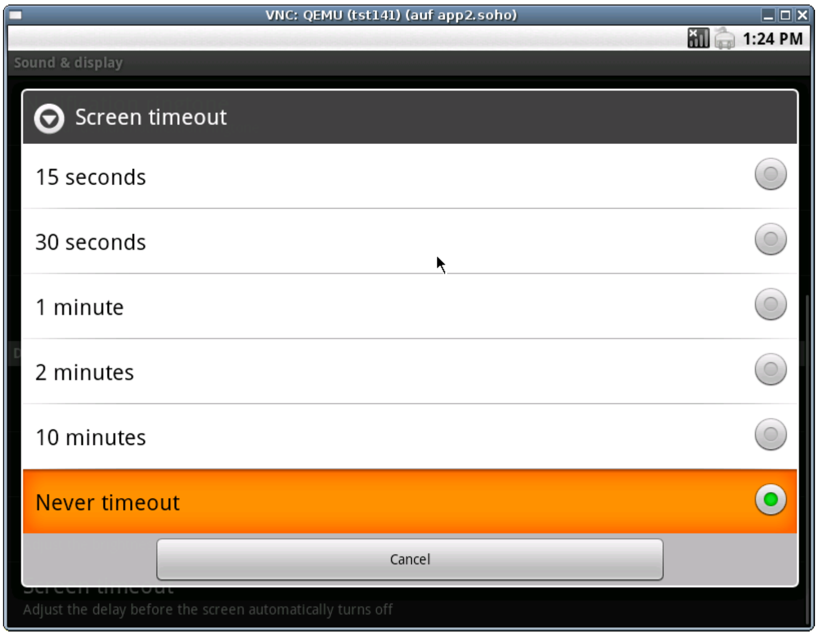


Figure 24.7: Deactivate screensaver - 04

For stable operations the following variation of predefined settings are applied maunally within the file 'tst141.ctys':

- Activate: NIC=$NIC:-pcnet
- Add: ARGSADD=" -no-acpi ";
- Eventually activate: VGADRIVER="-vga std"

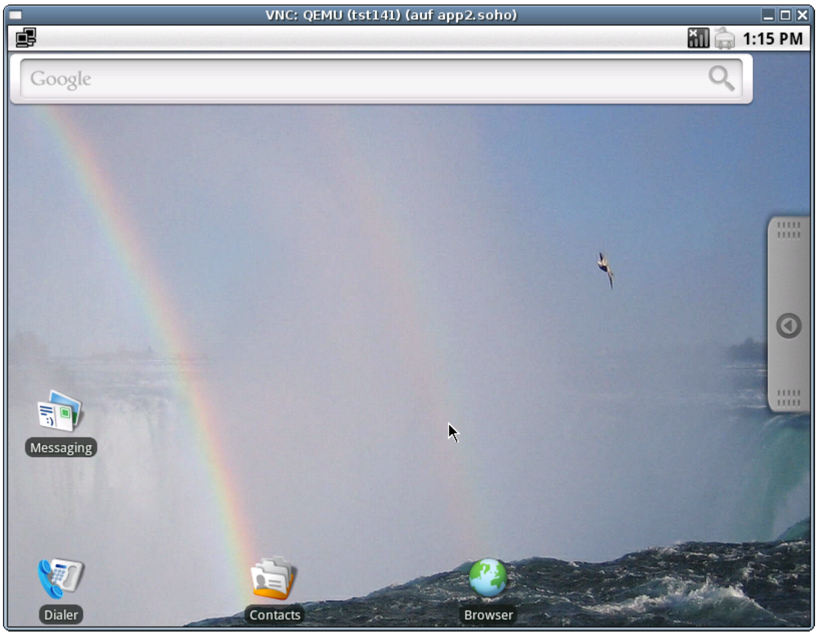


Figure 24.8: Android on QEMU/KVM

24.4.2 Creation and Installation on VirtualBox

The creation of the raw VM is the first step to be executed at the host operating system. This could be either performed locally or remote and requires the usage of the provided tools by VirtualBox(TM).

(a) Login into the machine where VirtualBox is installed.

```
ssh -X lab02
```

(b) Execute the VirtualBox(TM) console.

```
VirtualBox
```

(c) Create the VM, the machine is called here ’tst140’. The OS is ’Linux’, the version is ’Linux 2.6’.

Figure 24.9: Create Virtual Machine

(d) Set RAM to 512MByte.

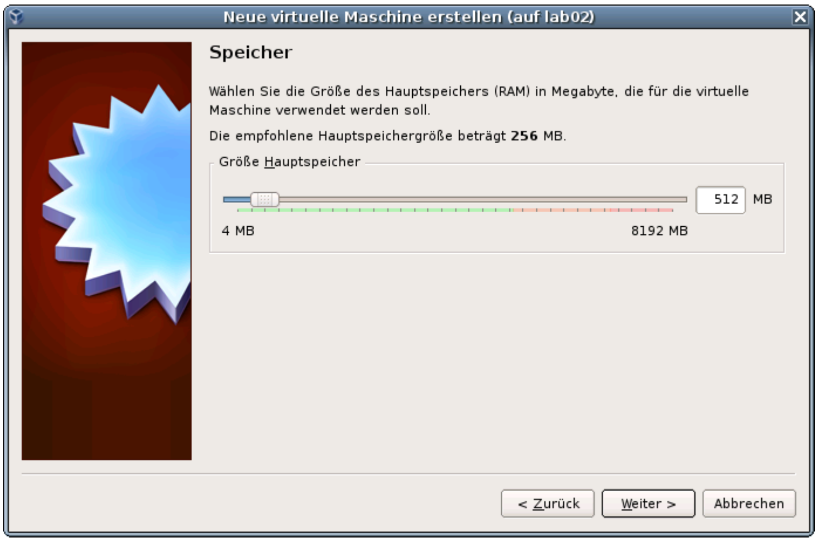


Figure 24.10: Set virtual RAM

(e) Create a virtual HDD, here 8GByte is choosen. When finished the raw VM is present and could be used as required, for basic functions of ctys no additional configuration is required.

Figure 24.11: Create Virtual HDD

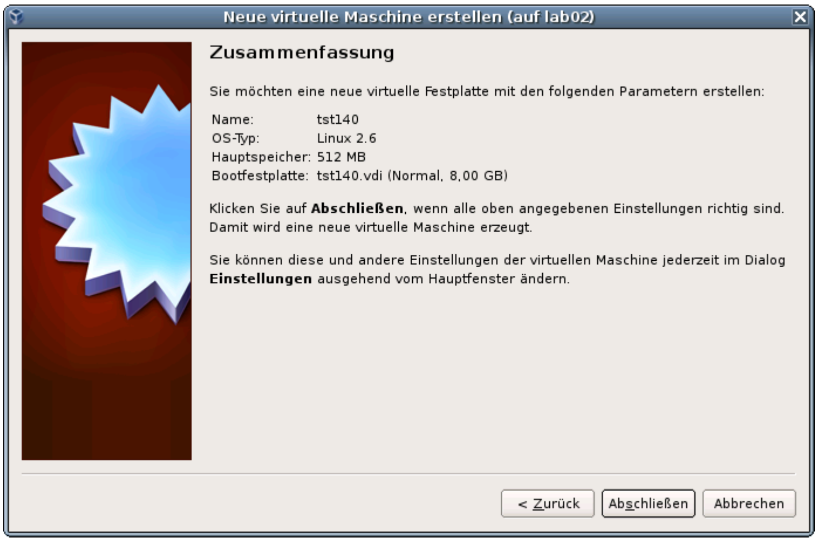


Figure 24.12: Check HDD image file

(f) The network device should be set to 'PCnet-Fast III' with DHCP, either NAT or bridged.

Figure 24.13: Network device

(g) The audio card has to be set to 'Sound Blaster 16'.

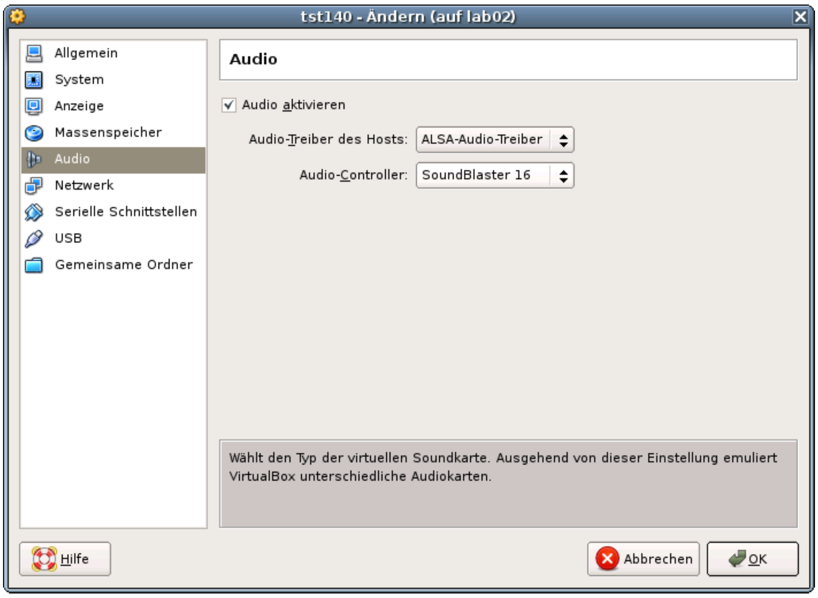


Figure 24.14: Audio device

(h) When additional information should be stored coallocated to the VM and scanned automatically into a database, than the tool **ctys-createConfVM(1)** should be applied. This generates additional detailed information related to the specific VM and the inherent guest OS. The call could be executed either interactive or automatic.

Call within the same directory for first inspection:

```
ARCH=i386 \
DIST=Android \
DISTREL=1.6-r2 \
OS=Linux OSREL=2.6 \
ctys-createConfVM -t vbox --label=tst140 --levo
```

This lists some defaults for the specific hypervisor. These could be preconfigured by specific template files within the configuration directory **ctys-createCOnfVM.d**. The result should look like the following:

```
Not all values require to be set, some will be requested later by
dialogue.
Thus it is not neccessary to have values assigned to the complete
displayed set.


Actually used sources for default values:
  no-marker  = Pre-Set value, either from defaults configuration,
               or by commandline.
  no-value   = Either requested by dialog later, or the defaults
               of the finally called application are used.
  (c)        = Read from actual configuration file, e.g. vmx-file.
  (d)        = Read from database.
  (g)        = Dynamically generated.
```

```
  (h)        = Used from current host as default.
  (m)        = Received from mapping definitions.

Applicable modifications:
  blue       = By call option, defines dependency for others.
  green      = By environment, 'could be set almost independent'
               from other values.
  cyan       = By miscellaneous facilities, but is dependent from
               others.
               E.g. LABEL defines by convention the network 'hostname',
               thus the TCP/IP params.
               This could ..., but should not be altered!

Most of the missing values will be fetched during actual execution of
this tool by dynamic evaluation.


                    VAR name:Initial Value

               C_SESSIONTYPE:VBOX
                       LABEL:tst140
                         MAC:08:00:27:A4:51:0B (c)
                          IP:
                      BRIDGE:
                        DHCP:
                     NETMASK:
                         TCP:
                     GATEWAY:

                      EDITOR:root

                        UUID:97d5a071-1914-477c-89c4-d47dd7adac74 (c)

                        DIST:Android
                     DISTREL:1.6-r2
                          OS:Linux
                       OSREL:2.6

                        ARCH:i386
                 ACCELERATOR:HVM  (c)
                         SMP:1 (c)
                     MEMSIZE:768 (c)
                  KBD_LAYOUT:de

                 STARTERCALL:/usr/bin/VirtualBox

             DEFAULTBOOTMODE:HDD

           DEFAULTINSTTARGET:/mntn/vmpool/vmpool05/vbox/test/...
                             ...tst-ctys/tst140/tst140.vdi
      HDDBOOTIMAGE_INST_SIZE:8192M
```

```
                    VMSTATE:ACTIVE


Remember that his is a draft pre-display of current defaults.
No consistency-checks for provided values are performed at this stage.
Some missing values are evaluated at a later stage dynamically.
```

When the call is finished without the '–levo' option the file 'tst140.ctys' with additional configuration information information is stored.

(i) The start of the VM could be proceeded either by calling VirtualBox, or by the VBOX plugin. Both require in current version the pre-configuration of the appropriate install procedure e.g. by attaching the install media. Here the boot image 'android-x86-1.6-r2.iso' is required.

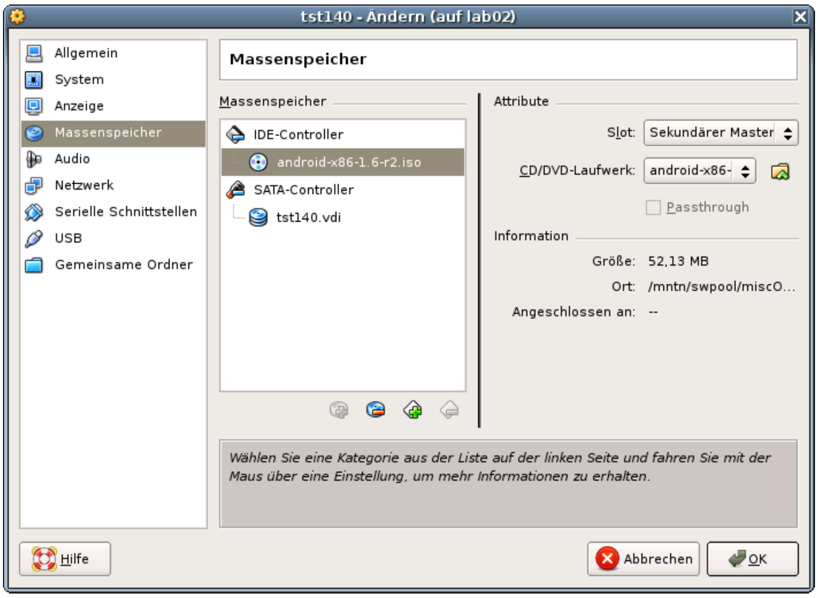


Figure 24.15: Install media

The following call starts the VirtualBox console.

```
VirtualBox
```

The following call variant starts the remote VM with a VirtualBox console:

```
ctys -t vbox \
 -a create=l:tst140,id:${TST140}/tst140.ctys,console:vbox\
 app2
```

(j) Now boot the VM and choose 'Installation Only' to start the installation.

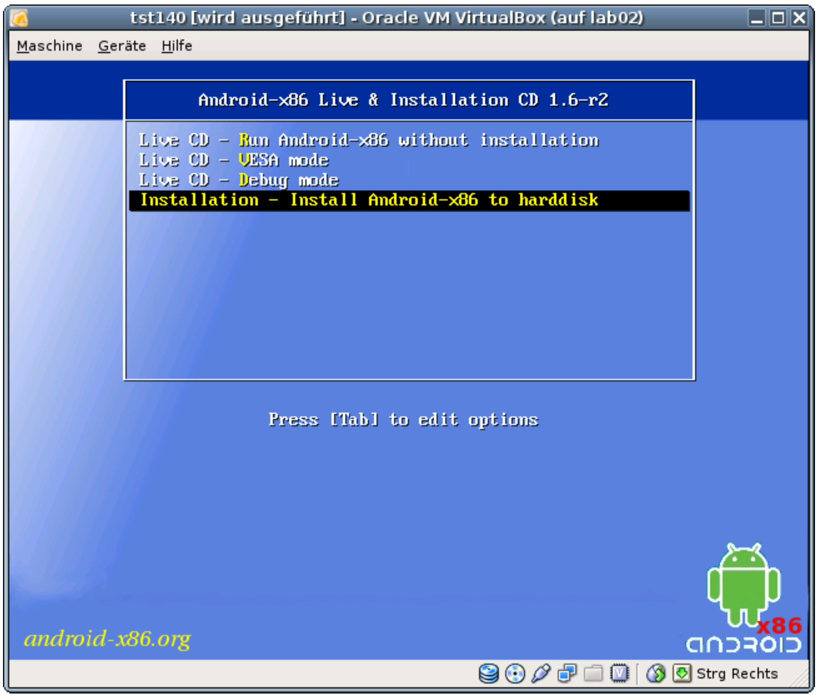


Figure 24.16: Install menue

(k) HDD partitioning.

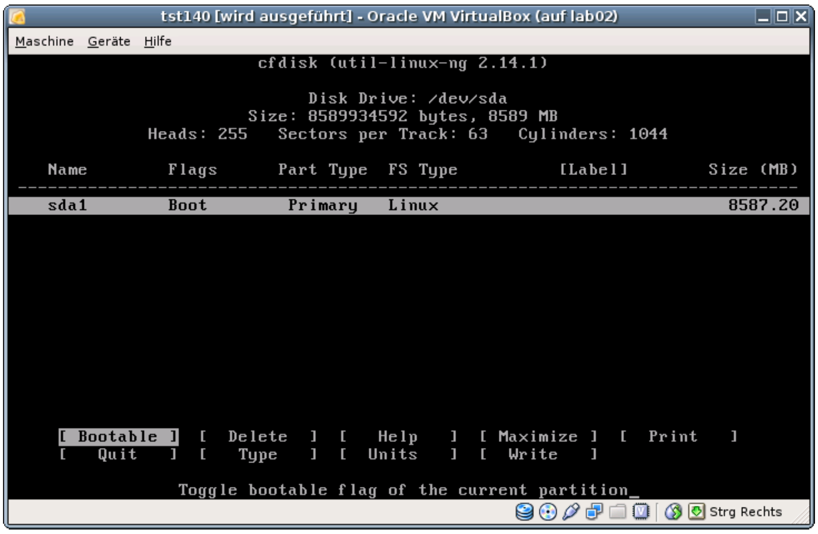


Figure 24.17: Format vHDD

(l) After the installation unmount the install media and boot into Android. In case of a first start the call could look like:

```
ctys -t vbox \
    -a create=l:tst140,id:${PWD}/tst140.ctys,console:vbox \
    app2
```

The default console is here RDP.

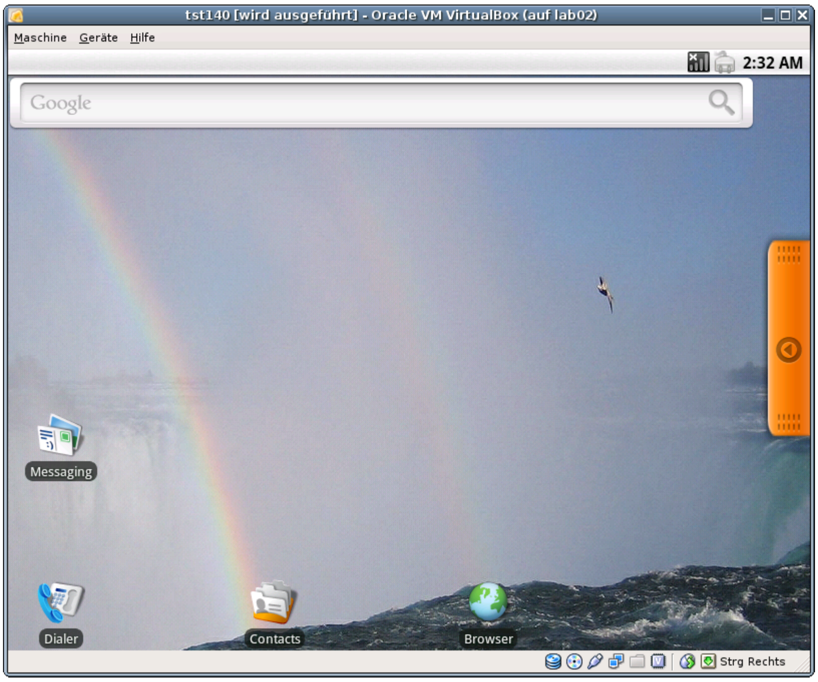


Figure 24.18: Android

Change into console with **Alt-F1**, **Alt-F7** returns to graphical display.

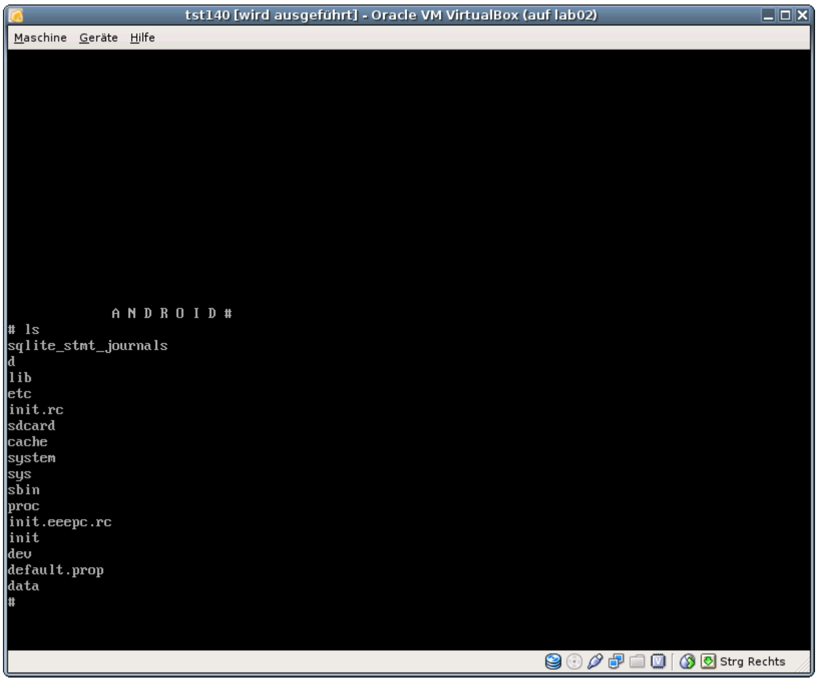


Figure 24.19: Android ASC-II Console

24.5 Creation of the Inventory - cacheDB

In case of a common mounted NFS filesystem for the pool VMs for simplicity just change into the directory of the VM on any machine. Call for the first check **ctys-vdbgenVM(1)** with the –**stdio** option for display only.

```
cd /mntn/vmpool/vmpool05/vbox/test/tst-ctys/tst140
ctys-vdbgen --append --base=$PWD --stdio -- root@lab02
cd /mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141
ctys-vdbgen --append --base=$PWD --stdio -- app2
```

When the result is displyed correctly just call without the '–stdio' option.

```
cd /mntn/vmpool/vmpool05/vbox/test/tst-ctys/tst140
ctys-vdbgen --append --base=$PWD  -- root@lab02
```

The following output should be displayed:

```
Prepare execution-call:

Require DB-PATH,        USE: DEFAULT_DBPATHLST="/homen/acue/.ctys/db/default"
Require DB-PATH,        USE: -o => "/homen/acue/.ctys/db/default"
APPEND mode                : ON(1)
STDIO mode off             : OFF(0)
Set TYPE scope          ADD: DEFAULT="-t ALL"
Preload TYPE set        ADD: DEFAULT="-T ALL"
For splitted operations ADD: DEFAULT="-b sync,seq "
Nameservice cache       OFF: DEFAULT="-c off "
Data cache              OFF: DEFAULT="-C off "

Resulting ENUMERATE     ADD: DEFAULT="-a enumerate=matchvstat:...
   ...active%disabled%empty,machine,b:/mntn/vmpool/vmpool05/vbox/...
   ...test/tst-ctys/tst140 -C off  -c off  -T ALL  "

-> generate DB(may take a while)...
-----------------------------------
START:14:55:11
------



------
END:14:55:38
DURATION:00:00:27
-----------------------------------
RET=0
-----------------------------------

Cached data:
```

```
  Mode:                     APPEND
  Pre-Appended:             835 records
  Appended:                 1 records
  Fetched Records Raw:       records
  Fetched Records Unique:    records
  Final:                    836 records

-----------------------------------
   ...finished.
```

The QEMU/KVM scan by:

```
cd /mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141
ctys-vdbgen --append --base=$PWD  -- app2
```

Should display:

```
Prepare execution-call:

Require DB-PATH,        USE: DEFAULT_DBPATHLST="/homen/acue/.ctys/db/default"
Require DB-PATH,        USE: -o => "/homen/acue/.ctys/db/default"
APPEND mode                : ON(1)
STDIO mode off             : OFF(0)
Set TYPE scope          ADD: DEFAULT="-t ALL"
Preload TYPE set        ADD: DEFAULT="-T ALL"
For splitted operations ADD: DEFAULT="-b sync,seq "
Nameservice cache       OFF: DEFAULT="-c off "
Data cache              OFF: DEFAULT="-C off "

Resulting ENUMERATE     ADD: DEFAULT="-a enumerate=matchvstat:active%...
   ...disabled%empty,machine,b:/mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141
   -C off  -c off  -T ALL  "

-> generate DB(may take a while)...
-----------------------------------
START:14:55:40
------


------
END:14:56:29
DURATION:00:00:49
-----------------------------------
RET=0
-----------------------------------

Cached data:
```

```
  Mode:                         APPEND
  Pre-Appended:                 836 records
  Appended:                     1 records
  Fetched Records Raw:           records
  Fetched Records Unique:        records
  Final:                        837 records

-----------------------------------
   ...finished.
```

This shows that only two(1+1) entries are appended to the existing database with 835 VM-Entries. Now check the database entry by calling:

```
ctys-vhost tst14
```

The following result should be displayed when the regular expression 'tst14.*' matches only twice:

```
label |stype|accel|distro |distrorel|os     |osrel|PM       |if|TCP
------+-----+-----+-------+---------+-------+-----+---------+--+------------
tst141|QEMU |KVM  |Android|1.6-r2   |Linux  |2.6  |app2.soho|0 |172.20.2.245
tst140|VBOX |HVM  |       |         |Linux26|     |lab02    |  |
```

24.6 Graphical Start of the Virtual Machine

This chapter demostrates the seamless integration of the hypevisors QEMU(emulation), QEMU/KVM, and VirtualBox(TM). The fully automatic generated database is synchronous with the graphical starter and offers the same and one user interface. This is the case for all supported plugins, due to missing native plugins for Android the LOGIN could not be demostrated for this special case.

24.6.1 Graphical Start of the Virtual Machine by QEMU/KVM

Now call the menue item for start of the VM 'tst141'.

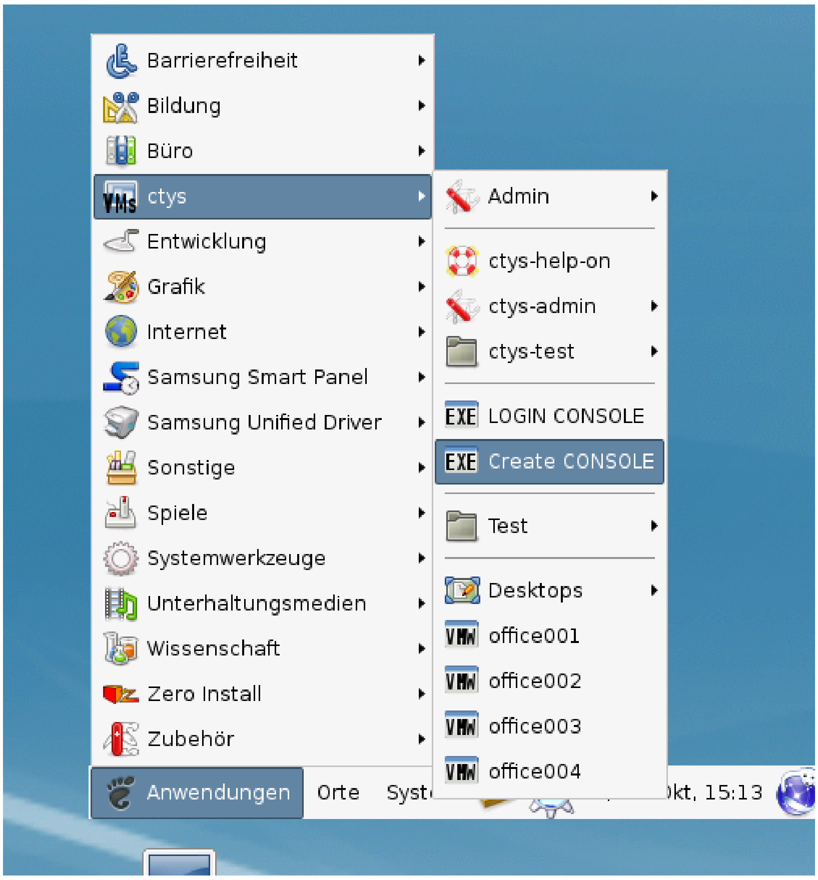


Figure 24.20: Android Start Menu

The created cacheDB record for thr VM 'tst140' is now automatically visible in the list of startable virtual machines.

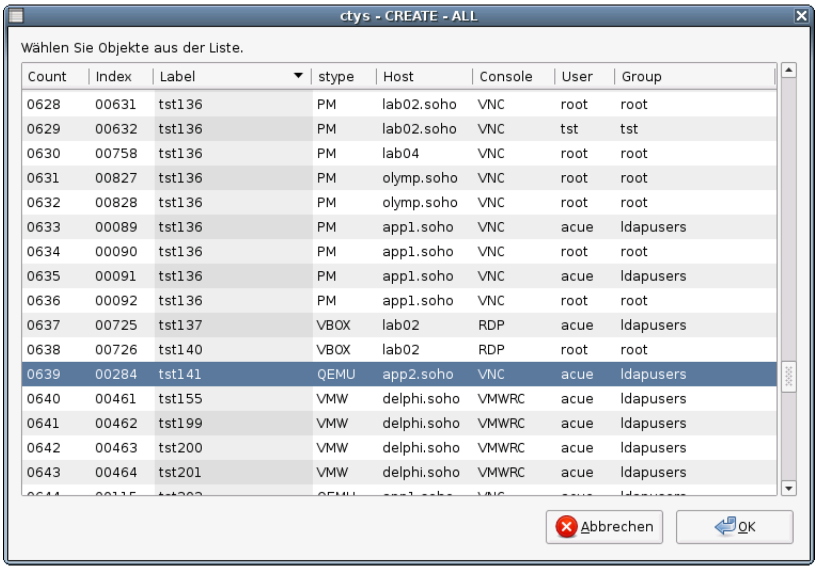


Figure 24.21: Android VM Selection

Confirm the selected entry.

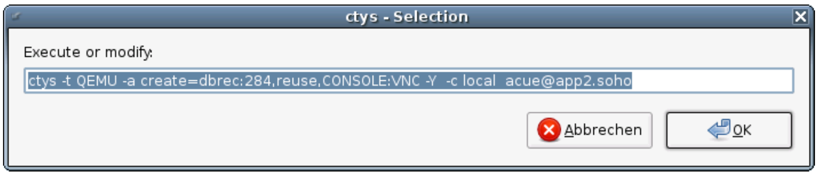


Figure 24.22: Android Call Confirmation

Boot...

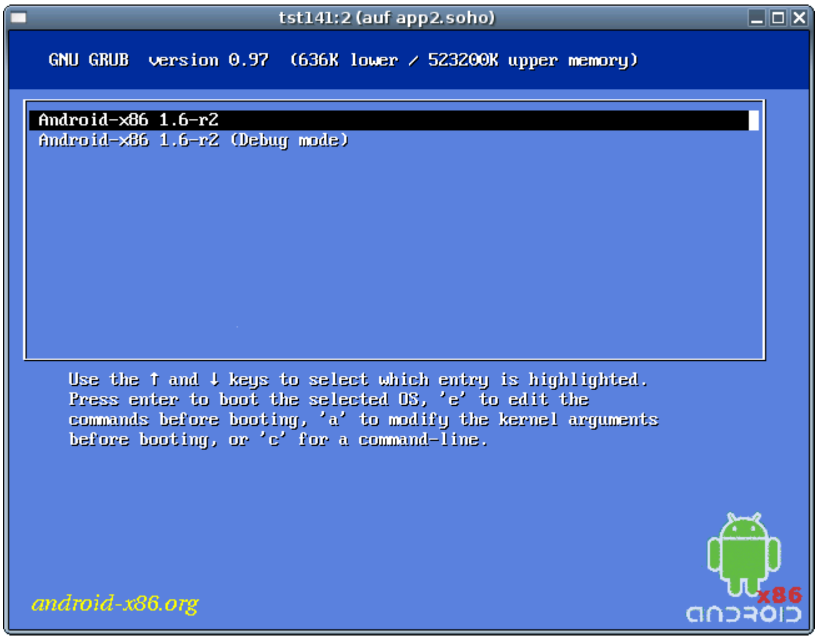


Figure 24.23: Boot Android

...and enjoy Android.

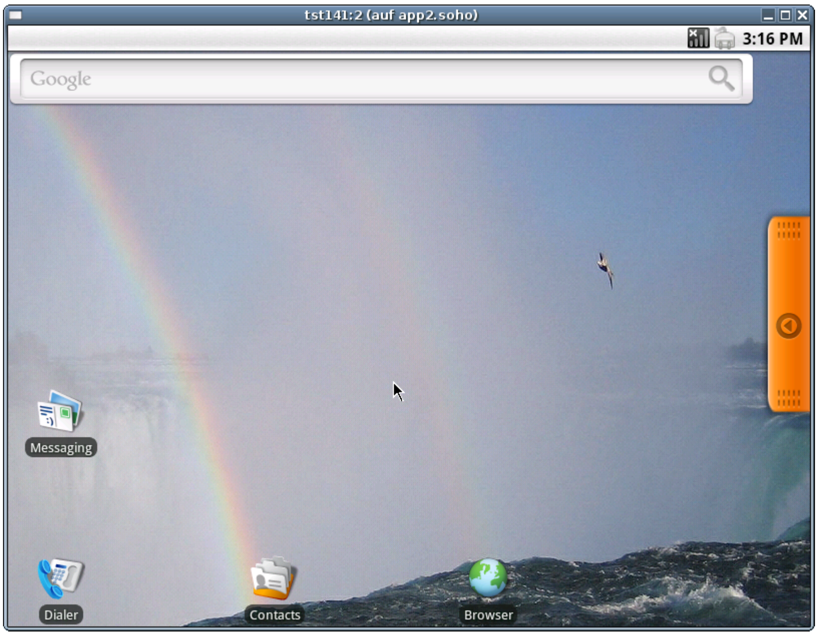


Figure 24.24: Enjoy Android

24.6.2 Graphical Start of the Virtual Machine by VBOX

Now call the menue item for start of the VM 'tst140'.

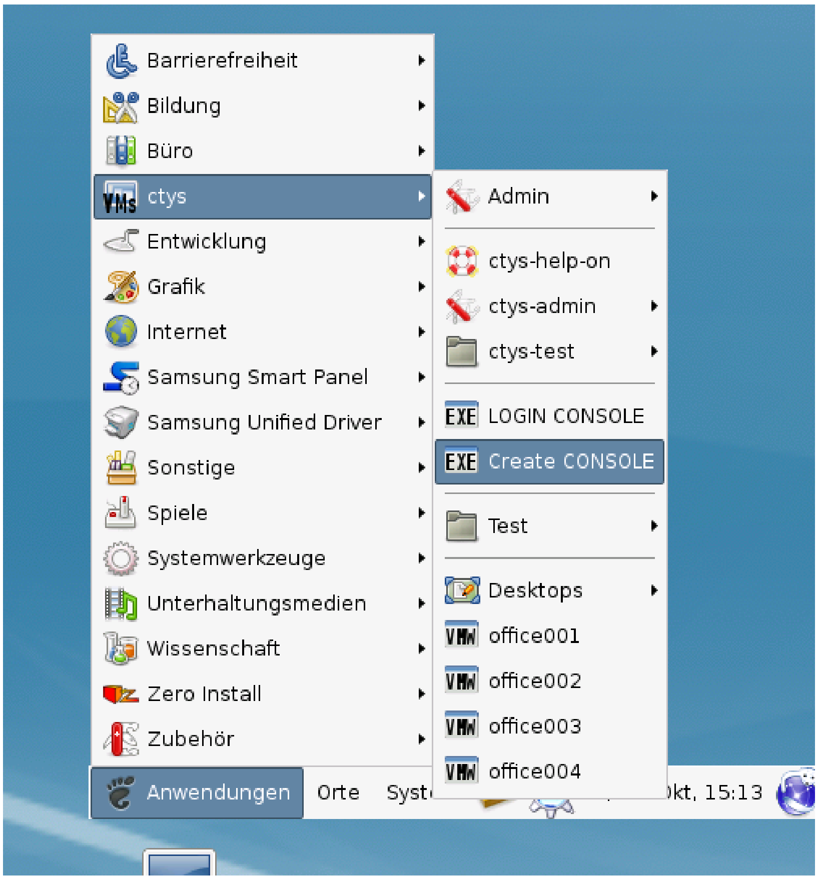


Figure 24.25: Android Start Menu

The created cacheDB record for thr VM 'tst140' is now automatically visible in the list of startable virtual machines.

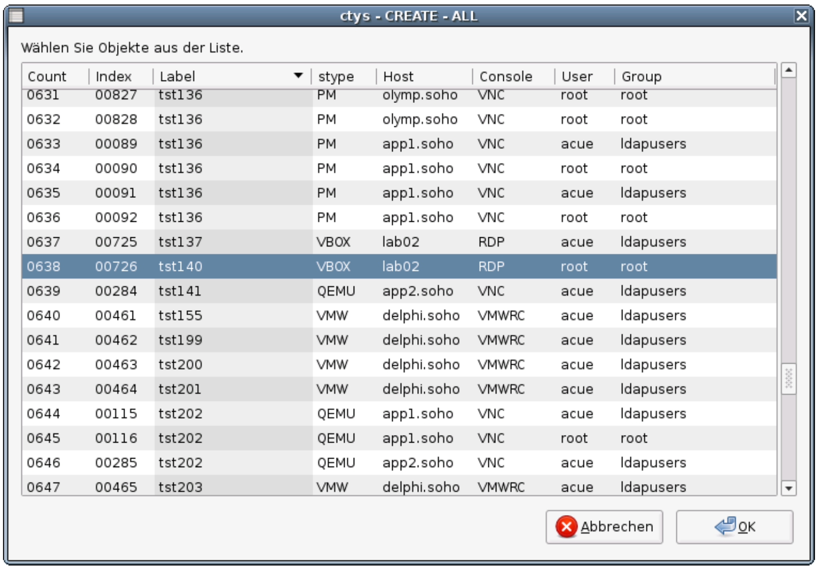


Figure 24.26: Android VM Selection

Confirm the selected entry.

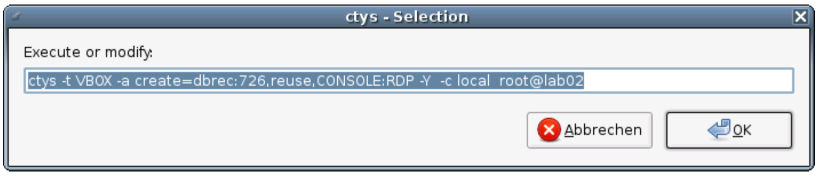


Figure 24.27: Android Call Confirmation

Boot ...

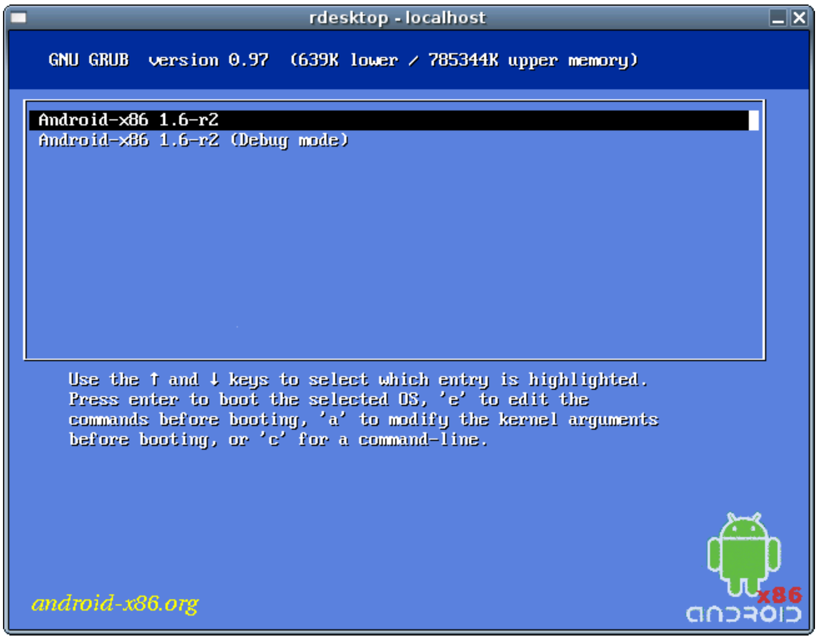


Figure 24.28: Boot Android

...and enjoy Android.

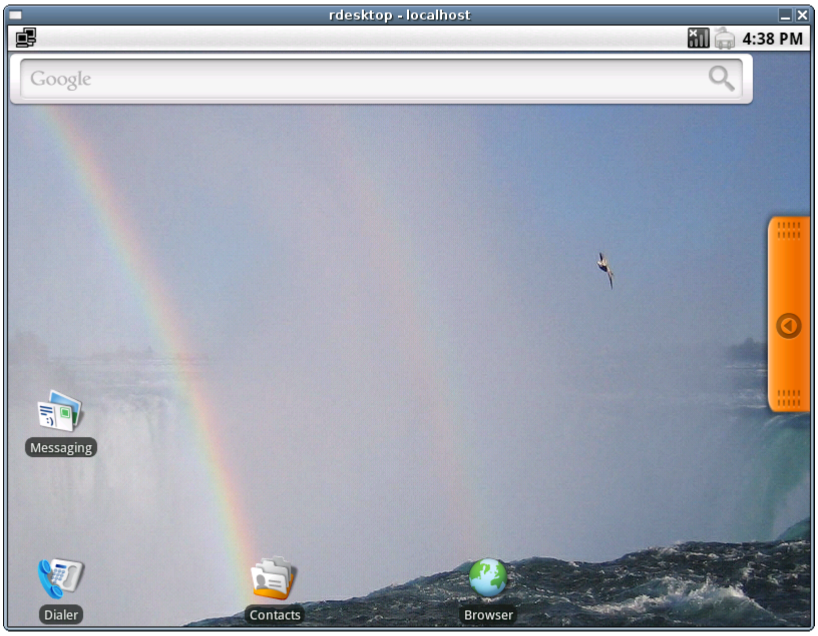


Figure 24.29: Enjoy Android

24.7 Manage the VM

For now no native plugin for Android is supported.

Chapter 25

CentOS

.

25.1 General

The current document shows the basic installation of CentOS, which is basically a derivative of RedHat(TM) Linux variant.

The following host environment is used here:

- CentOS-5.4 with kvm-83 / Qemu-0.9.1
- Debian-5.0.6 with VirtualBox-3.2.10
- CentOS-5.4 with VMware-Player-2
- CentOS-5.4 with VMware-Player-3
- CentOS-5.5 with VMware-Server-2.0.2
- CentOS-5.4 with VMware-Workstation-6
- CentOS-5.4 with VMware-Workstation-7
- OpenSUSE-11.3 with Xen-3.x

The following client environment is used here:

- CentOS-5.5
- UnifiedSessionsManager - ctys-01.11.011

The following common assumptions and simplifications are choosen, when multiple approaches are valid.

(a) The initial start of the machines are executed before scanning these into the inventory database. Thus the call is frequently executed by the suboption 'b:$PWD', which defines the filesystem scan to be started at the given directory, in this case the current dir. This is particularly helpful in NFS based distributed environments with processing nodes containing identical directory structures.

(b) The initial installation is proceeded by the vendor tools, when available. This avoids some deeper knowledge for the application of varios options.

(c) The example setups are generally the provided defaults by the distributions. This should be also the first trial to become familiar with the environment.

.

25.2 Setup of Host-OS and Hypervisor

The installation for the following variants has to be performed by the appropriate standard setup of the HostOS, which quite straight forward:

- Debian with VirtualBox
- CentOS with QEMU/KVM
- CentOS with VMware-Server
- CentOS with VMware-Player
- CentOS with Xen

25.3 Setup of the UnifiedSessionsManager

25.3.1 Install tgz-Packages

(a) Apply the standard installation procedure:

`ctys-distribute -F 2 -P UserHomeCopy root@myHost`

(b) Open a Remote Shell by call of CLI plugin:

`ctys -t cli -a create=l:myHost root@myHost`

(c) Check the plugins states by calling ctys-plugins:

`ctys-plugins -T all -E`

25.3.2 Install rpm-Packages

The following steps are required for a RPM based setup on CentOS. The installation is relocatable, but located at '/opt', and installed locally by 'ctys-distribute'.

(a) Install BASE package.

`rpm -i ctys-base-01.11.011.noarch.rpm`

(b) Now install a a local version, here by copy. The PATH prefix is important here, particularly in case of updates. The path is resolved to it's actual path by eliminating any symbolic link, and used for consistent link of libraries.

`/opt/ctys-01.11.011/bin/ctys-distribute -F 2 -P UserHomeCopy`

(c) Next the menu is setup.

`ctys-xdg --menu-create`

(d) Now the help is available as eihter a Gnome or KDE menu. Alternatively could be called from the commandline.

25.3.3 Setup of the Gnome Menu

The setup of the Gnome Menu is quite simple, the contained tool **ctys-xdg** sets up a standard menu by the call:

```
ctys-xdg --menu-create
```

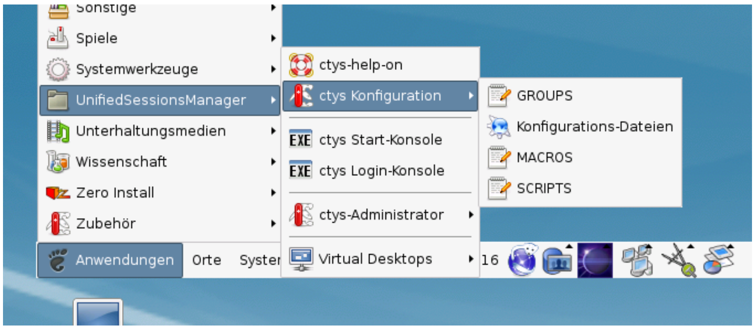


Figure 25.1: Default Menu

The call

`ctys-xdg --menu-cancel`

removes the installed files. For current version no checks for changed files is done.

The menues could be edited and extended by the call

`ctys-xdg --menu-edit`

which opens the related directories for modification of '*.menu', '*.desktop', and '*.directory' files.

25.4 Creation of the the Raw-VM

25.4.1 Creation of the Raw-VM with QEMU/KVM

The demo example VM is here named tst219, this is the hostname of GuestOS too.

(a) Login into the machine where VirtualBox is installed.

`ssh -X app2`

When just the processing node of mounted filesystem has to be changed, the following call could be applied. This works in case of identical mount paths:

`ctys -t cli -a create=l:tst220,cd:$PWD root@lab02`

(b) Change to the vmpool and create a directory and change into.

`mkdir tst219`

(c) Call the install and configuration utility for VMs. Here some values are set by environment variables, a complete list including the actually assigned values could be displayed by the option –**levo**.

```
ARCH=x86_64 \
DIST=CentOS \
DISTREL=5.5 \
OS=Linux \
OSREL=2.6 \
ctys-createConfVM -t qemu --label=tst219
```

This call creates a virtual image(hda.img), the call-wrapper(tst219.sh), and the configuration file(tst219.ctys). The files are created from templates by assigning configuration values either from pre-configured default values, or interactive

variation. The whole process of createion could be batch-proceeded by using the either teh –**auto**, or the –**auto-all** option when appropriate default values are preconfigured.
When no MAC database nor DHCP is available, the MAC and IP addresses might be provided too.

(d) Once the set of files is created the virtual machine is prepared for startup. For some other systems complete installation routines are available, e.g. debian and CentOS. The current state could be checked now by the following call.

```
./tst219.sh --console=vnc  --vncaccessdisplay=47 --print --instmode --check
```

(e) The installation could be started now e.g. on the install host by:

```
./tst219.sh --console=vnc  --vncaccessdisplay=47 --print --instmode
```

Alternatively a remote call could be proceeded:

```
ctys -t qemu -a create=l:tst219,b:${VMPATH},instmode app2
```

In case of appropriate defaults(refer to tst220.ctys) this starts e.g. the CD/DVD installation.

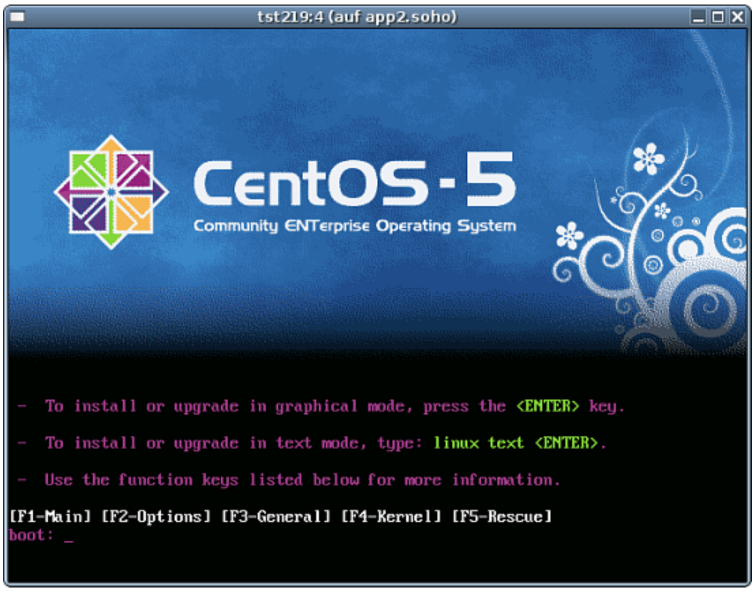


Figure 25.2: Start CentOS installation - CD/DVD

The following call with PXE is utilized here:

```
ctys \
   -t qemu \
   -a create=l:tst219,b:${VMPATH},instmode:PXE%default%HDD%default%init \
   app2
```

Resulting to:

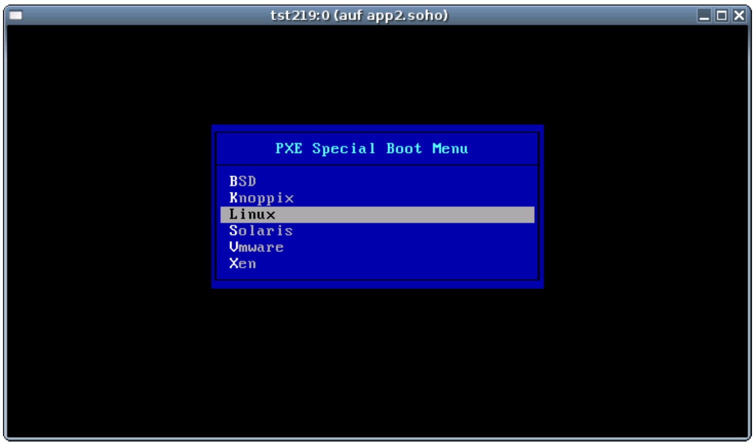


Figure 25.3: Start CentOS installation - PXE

(f) Once the appropriate install kernel is choosen, the following procedure is the common install procedure as described later.

25.4.2 Creation of the Raw-VM with VirtualBox

The creation of the raw VM is first step to be executed at the host oeprating system. This could be either performed locally or remote and requires the usage of the provided tools by VirtualBox(TM).

(a) Login into the machine where VirtualBox is installed.

`ssh -X lab02`

(b) Execute the VirtualBox(TM) console.

`VirtualBox`

(c) Create the VM, the machine is called here 'tst220'. When finished the raw VM is present and could be used as required, for basic functions of ctys no additional configuration is required.

   i. The OS is 'Linux', the version is 'Linux 2.6'.
   ii. Set RAM to 512MByte.
   iii. Create a virtual HDD, here 8GByte is choosen.

(d) When additional information is required to be stored coallocated to the VM and scanned automatically into a database, than the tool **ctys-createConfVM** should be applied. This generates additional detailed information related to the specific VM and the inherent guest OS. The call could be executed either interactive or automatic.

Call within the same directory for first inspection:

`ctys-createConfVM -t vbox --label=tst137 --levo`

This lists some defaults for the specific hypervisor. These could be preconfigured by specific template files within the configuration directory **ctys-createConfVM.d**. The following call actually generates the appropriate configuration

```
DIST=CentOS \
DISTREL=5.5 \
OS=Linux \
OSREL=2.6 \
```

```
MAC=00:50:56:13:12:14 \
IP=172.20.2.20 \
ARCH=x86_64 \
ctys-createConfVM --label=tst220 -t vbox
```

The result displayed with –**levo** is:

```
Not all values require to be set, some will be requested later
by dialogue.
Thus it is not neccessary to have values assigned to the complete
displayed set.


Actually used sources for default values:
  no-marker  = Pre-Set value, either from defaults configuration,
               or by commandline.
  no-value   = Either requested by dialog later, or the defaults
               of the finally called
               application are used.
  (c)        = Read from actual configuration file, e.g. vmx-file.
  (d)        = Read from database.
  (g)        = Dynamically generated.
  (h)        = Used from current host as default.
  (m)        = Received from mapping definitions.

Applicable modifications:
  blue       = By call option, defines dependency for others.
  green      = By environment, 'could be set almost independent'
               from other values.
  cyan       = By miscellaneous facilities, but is dependent from
               others.
               E.g. LABEL defines by convention the network 'hostname',
               thus the TCP/IP params.
               This could ..., but should not be altered!

Most of the missing values will be fetched during actual execution of
this tool by dynamic evaluation.


                      VAR name:Initial Value

                 C_SESSIONTYPE:VBOX
                         LABEL:tst220
                           MAC:00:50:56:13:12:14 (c)
                            IP:172.20.2.20 (m)
                        BRIDGE:
                          DHCP:
                       NETMASK:
                           TCP:
                       GATEWAY:

                        EDITOR:acue
```

```
                         UUID:e982e0c4-2de8-40f1-aa6e-8c615096b37c (c)

                         DIST:debian (h)
                      DISTREL:5.5
                           OS:Linux
                        OSREL:2.6

                         ARCH:x86_64
                  ACCELERATOR:HVM  (c)
                          SMP:1 (c)
                      MEMSIZE:768 (c)
                   KBD_LAYOUT:de

                  STARTERCALL:/usr/bin/VirtualBox

              DEFAULTBOOTMODE:HDD

            DEFAULTINSTTARGET:/mntn/vmpool/vmpool05/vbox/test/...
                              ...tst-ctys/tst220/tst220.vdi
       HDDBOOTIMAGE_INST_SIZE:8192M

                      VMSTATE:ACTIVE


Remember that his is a draft pre-display of current defaults.
No consistency-checks for provided values are performed at this stage.
Some missing values are evaluated at a later stage dynamically.
```

When the call is finished the file 'tst137.ctys' with additional configuration information information is stored.

(e) Add the install image as a bootable CD/DVD and set this as the boot device for the VM or use PXE. The procedures are identical after the boot of the kernel. This example uses PXE.

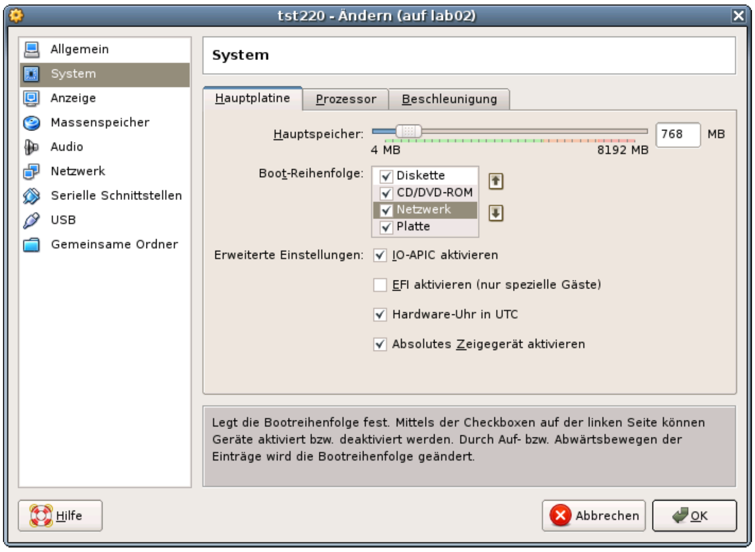


Figure 25.4: Set PXE

Use network install of CentOS-5.5.

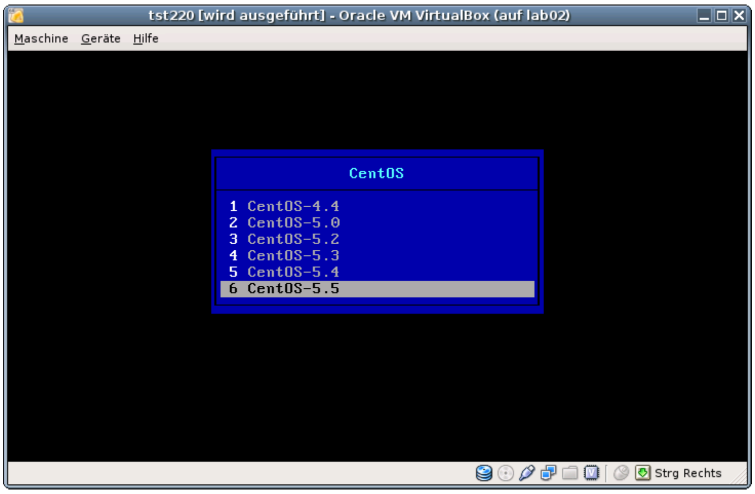


Figure 25.5: Set PXE

(f) The start of actual CentOS-5.5 install procedure, from now on all post-bootstrap procedured are equal. Here the start from the VirtualBox console is choosen.

25.4.3 Creation of the Raw-VM with VMware-Server-1

ffs.

25.4.4 Creation of the Raw-VM with VMware-Server-2

The installation of raw machine is performed here by the native vendor supported tools. These could be started e.g. by using the X11 plugin and execution of a remote command. The advance is the transparent encryption on the inter-node connections by SSH. Ths e.g. in case of problems with the https port the unencrypted http GUI could still be used in a secure manner for network connections. All connections are tunneld by OpenSSH, here the X-displayforwarding with the '-X' option. The start of the VMW console for RHEL-5.5 and VMware Server-2.0.2 is:

```
ctys -t x11 -a create=l:vmwcon,cmd:vmware root@lab05
```

This starts the default fornt end, here the Firefox browser.

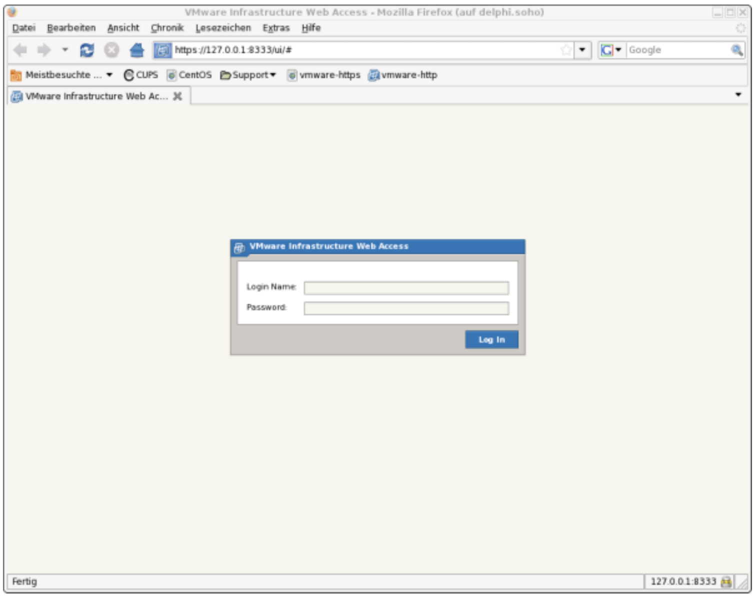


Figure 25.6: Start VMware Server Console

REMARK:
The current version of the **UnifiedSessiosnManager** requires by convention the coallocation of the VMX file and the boot HDD. Particularly the enumeration of VMs requires the presence of the VMX file. In some cases - for Server-2 when the allocation is altered from the defined storage - these are stored by default into different directories. This has to be considered for the allocation of new VMs.

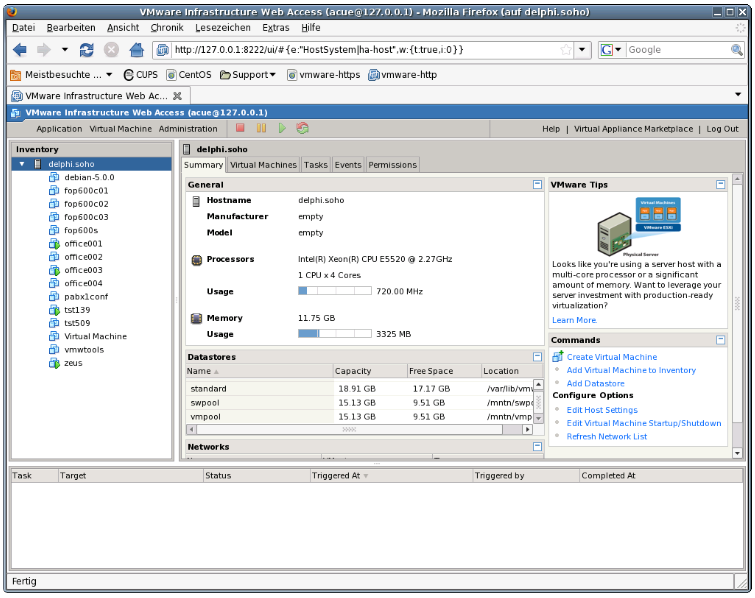


Figure 25.7: VMware Server Console

The virtual machine should be selected as hardware version **4** when maximum compatibility is required.

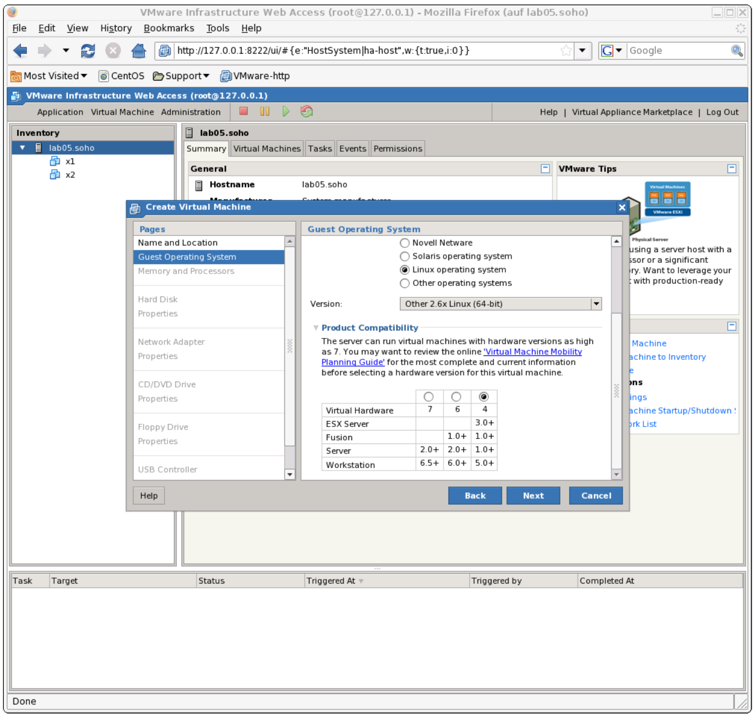


Figure 25.8: VMware Compatibility

For tight and vendor independent management of the VMs and PMs the MAC addresses should be assigned individually to each machine and centrally managed by DHCP.

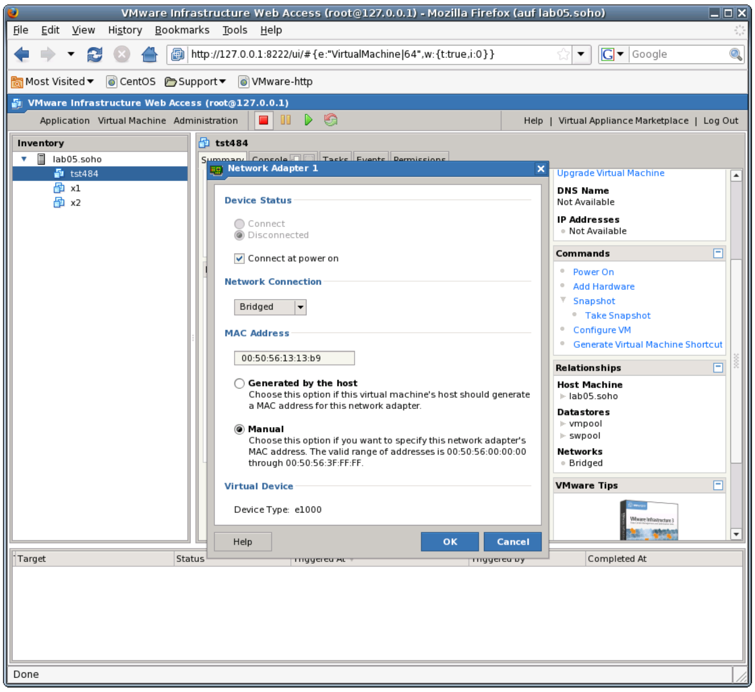


Figure 25.9: VMware set MAC address

In addition the UUID of the VM should be set to fixed by maunal edition of the VMX file. This has advantages for unambiguity in networked operating environments. The UUID is part of the <**machine-address**> and therefore stored within the database and could be used for persistent addressing. Thus should not be changed by a harmless move, due to an algorithm for assurance of generic unambiguity.

Once the setup is finished by means of the vendor tools, the following steps of installation could be proceeded either continued solely by the vendor provided environment, or by application of the UnifiedSessionsManager toolset. The **instmode** for adaption of the actual boot configuration is not yet supported, thus a normal startup by management of boot and installmedia by the vendor products has to be applied.

The following operational procedures within the GuestOS are similar for all hypervisors. Just a few exceptions exist for installing specific driver sets - so called Tools available e.g. for almost all VMware(TM) products. These have to be mounted as install media and installed by the provided installer.

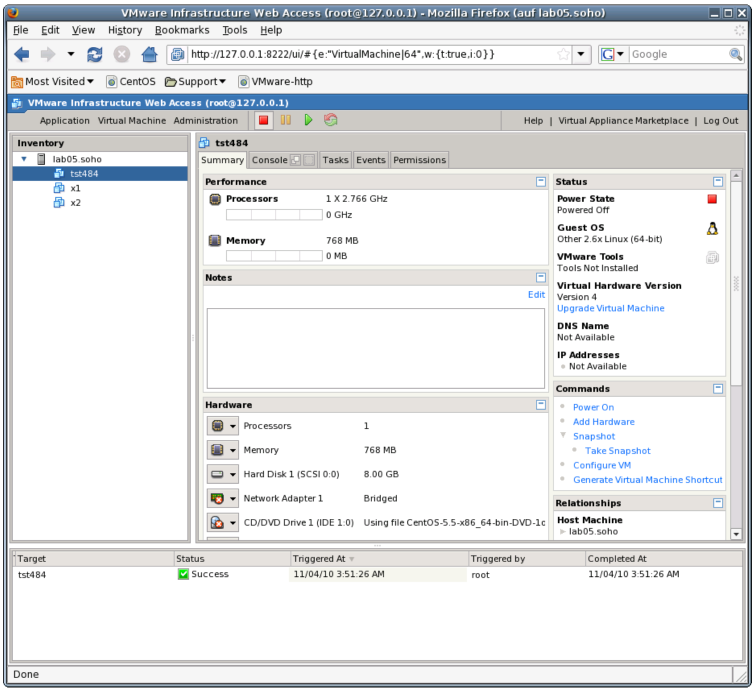


Figure 25.10: VMware Prepared VM

The following demonstrates the reallocation of the machine files to a common directory with the storage devices. The virtual HDD is stored within the directory

```
[Datastore] vmpool05/vmw/test/tst-ctys/tst484/tst484.vmdk
```

whereas the VM configuration files are stored by the system in the datastore to

```
[Datastore] tst484/...
```

(a) Remove the VM tst488 without deletion.

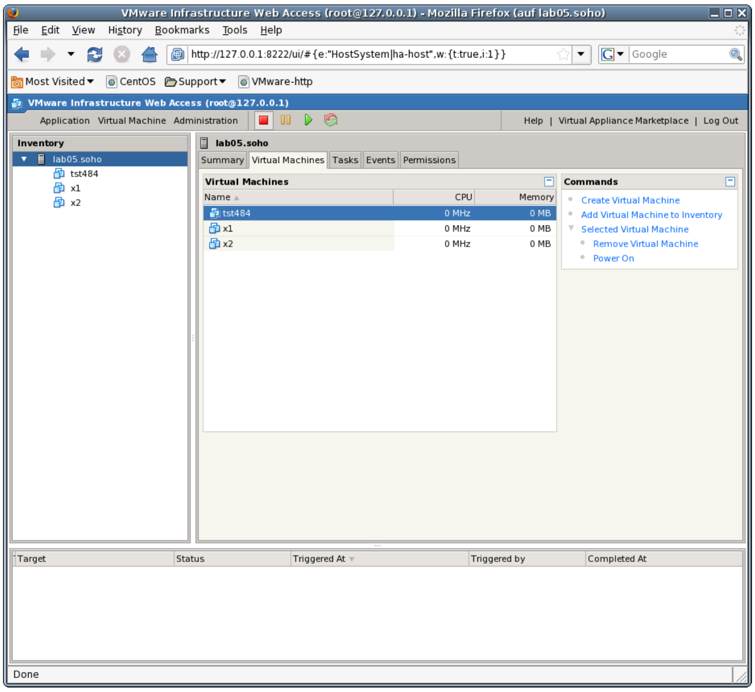


Figure 25.11: Remove VM

(b) Move the directory and concat all files within the same. Than check the filenames of storage devices within the VMX file, which are absolute filenamepaths anyway. Here:

```
scsi0:0.fileName = "/mntn/vmpo ..."
ide1:0.fileName = "/mntn/swpool/UNIXDist..."
```

(c) Make the UUID static by:

```
uuid.action = "keep"
```

(d) Make the MAC address static by:

- Delete:

```
ethernet0.addressType = "generated"
ethernet0.generatedAddress = "00:0c:29:9c:6a:6a"
```

- Add:

```
ethernet0.addressType = "static"
ethernet0.address = "00:50:56:13:11:33"
```

(e) Adapt - if required -

```
displayName = "tst484"
```

which is the so called **LABEL**.

For the management of the GuestOSs and integration into the database of the UnifiedSessionsManager the inventory functions by **ctys-createConfVM** and **ctys-vdbgen** should be at least post-applied once after finishing the guest installation.

The installed GuestOS is here the same as the HostOS, with the only difference, that the architecture has to be set to 'ARCH=i386'. This reduces the call for configuration creation to:

```
ARCH=i386 ctys-createConfVM -t vmw --label=tst484
```

The –**levo** display is:

```
Not all values require to be set, some will be requested later by dialogue.
Thus it is not neccessary to have values assigned to the complete displayed set.


Actually used sources for default values:
  no-marker  = Pre-Set value, either from defaults configuration,
               or by commandline.
  no-value   = Either requested by dialog later, or the defaults
               of the finally called
               application are used.
  (c)        = Read from actual configuration file, e.g. vmx-file.
  (d)        = Read from database.
  (g)        = Dynamically generated.
  (h)        = Used from current host as default.
  (m)        = Received from mapping definitions.

Applicable modifications:
  blue       = By call option, defines dependency for others.
  green      = By environment, 'could be set almost independent'
               from other values.
  cyan       = By miscellaneous facilities, but is dependent from others.
               E.g. LABEL defines by convention the network 'hostname',
               thus the TCP/IP params.
               This could ..., but should not be altered!

Most of the missing values will be fetched during actual execution of
this tool by dynamic evaluation.


                   VAR name:Initial Value

                      LABEL:tst484
                        MAC:00:50:56:13:13:B9 (c)
                         IP:172.20.6.184 (m)
                     BRIDGE:
                       DHCP:
                    NETMASK:
                        TCP:tst484 (m)
                    GATEWAY:

                     EDITOR:acue

                       UUID:564d99fb5a6c2897edce5b14279c6a6a (c)

                       DIST:CentOS (h)
                    DISTREL:5.5 (h)
                         OS:Linux (h)
                      OSREL:2.6.18-194.el5 (h)
```

```
        ARCH:x86_64 (h)
 ACCELERATOR:
         SMP:
     MEMSIZE:768 (c)
  KBD_LAYOUT:de

 STARTERCALL:/usr/bin/vmware


     VMSTATE:ACTIVE


Remember that his is a draft pre-display of current defaults.
No consistency-checks for provided values are performed at this stage.
Some missing values are evaluated at a later stage dynamically.
```

The result could be inspected e.g. by the following call with one of the standard macros. Called within the directory of the VM, therefore starting at the scan-base 'b:$PWD'.

```
ctys -t vmw "{MACRO:enumdefault},b:$PWD"
```

Resulting in the output:

```
label |stype|accel|distro|distrorel|os   |osrel     |PM        |if|TCP
------+-----+-----+------+---------+-----+----------+----------+--+-----------
tst484|VMW  |     |CentOS|5.5      |Linux|2.6.18-194|lab05.soho|0 |172.20.6.18
```

25.4.5 Creation of the Raw-VM with VMware-Player

ffs.

25.4.6 Creation of the Raw-VM with VMware-Workstation

ffs.

25.4.7 Creation of the Raw-VM with VMware-ESXi

ffs.

25.4.8 Creation of the Raw-VM with VMware-ESX

ffs.

25.4.9 Creation of the Raw-VM with Xen

The examples for installaltion of Xen GuestOSs are performed here on a RedHat-Enterprise-Linux - RHEL-5.5 server. The procedures are almost identicel to other derived distributions, e.g. CentOS, ScientificLinux, or EnterpriseLinux.

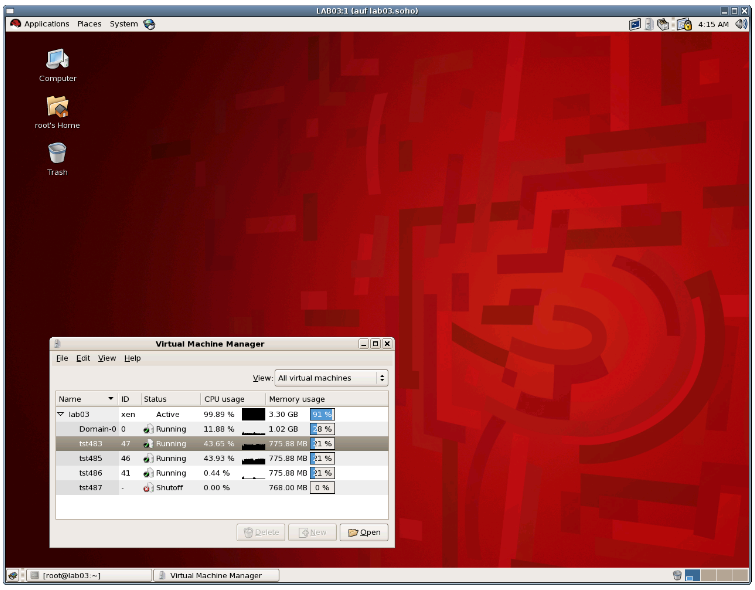


Figure 25.12: Remove VM

The Xen files, including the Python conf-file and the initial virtual devices are created by the utility **ctys-createConfVM**. Thus e.g. the MAC address has to be provided when networking is required. In some cases - e.g. for OpenSUSE or debian - it might be required to provide the virtual bridge too. This is due to internal detection a so called Xen-Bridge by searching for the first bridge containing a 'pethX' device, which sometimes varies. E.g. for debian the bridge is called in some releases simply 'eth0'. Thus when errors with networking due to missing bridge occurs, than just set an appropriate default. If this still does not suffice, than the variable 'FORCE_THIS_IS_XEN_BRIDGE=br0' may help. But be aware, when the machine is executed on different machines with various HostOSs, e.g. viy NFS. Than the bridge names may vary, and may require to be adapted. This is the reason of dynamic evaluation for the networking devices.

Now execute the call for the complete creation of the VM.

```
MAC=00:50:56:13:13:B7 \
DIST=CentOS \
DISTREL=5.5 \
OS=Linux \
OSREL=2.6.18 \
ctys-createConfVM -t XEN --label=tst482
```

This creates with the **–levo** check the output:

```
Not all values require to be set, some will be requested later by dialogue.
Thus it is not neccessary to have values assigned to the complete displayed se


Actually used sources for default values:
  no-marker  = Pre-Set value, either from defaults configuration,
               or by commandline.
  no-value   = Either requested by dialog later, or the defaults
               of the finally called
               application are used.
  (c)        = Read from actual configuration file, e.g. vmx-file.
  (d)        = Read from database.
  (g)        = Dynamically generated.
  (h)        = Used from current host as default.
  (m)        = Received from mapping definitions.

Applicable modifications:
  blue       = By call option, defines dependency for others.
  green      = By environment, 'could be set almost independent'
               from other values.
  cyan       = By miscellaneous facilities, but is dependent from others.
               E.g. LABEL defines by convention the network 'hostname',
               thus the TCP/IP params.
               This could ..., but should not be altered!

Most of the missing values will be fetched during actual execution of
this tool by dynamic evaluation.


                   VAR name:Initial Value
               C_SESSIONTYPE:XEN
                      LABEL:tst482
                        MAC:00:50:56:13:13:B7
                         IP:172.20.6.182 (m)
                     BRIDGE:
                       DHCP:
                    NETMASK:
                        TCP:tst482 (m)
                    GATEWAY:

                     EDITOR:acue

                       UUID:e589efe5-5fe5-4de8-890c-43484b5a64e4 (h)

                       DIST:CentOS
                    DISTREL:5.5
                         OS:Linux
                      OSREL:2.6.18
                       ARCH:x86_64 (h)
                ACCELERATOR:HVM  (h)
```

```
                         SMP:1
                     MEMSIZE:768
                  KBD_LAYOUT:de

                 STARTERCALL:/usr/sbin/xm
                 WRAPPERCALL:/usr/bin/sudo.sh

             DEFAULTBOOTMODE:HDD

           DEFAULTINSTTARGET:/mntn/vmpool/vmpool05/xen/test/tst-ctys/tst482/xvda.
      HDDBOOTIMAGE_INST_SIZE:8G
 HDDBOOTIMAGE_INST_BLOCKSIZE:256M
DDBOOTIMAGE_INST_BLOCKCOUNT:32
   HDDBOOTIMAGE_INST_BALLOON:y

             DEFAULTINSTMODE:CD
                INSTSRCCDROM:/mntn/swpool/UNIXDist/centOS/5.5/inst/isos/...
                             ...x86_64/CentOS-5.5-x86_64-bin-DVD-1of2.iso
          DEFAULTINSTSOURCE:/mntn/swpool/UNIXDist/centOS/5.5/inst/isos/...
                             ...x86_64/CentOS-5.5-x86_64-bin-DVD-1of2.iso

                  BOOTLOADER:/usr/lib/xen/boot/hvmloader
                 INST_KERNEL:/usr/lib/xen/boot/hvmloader

                     VMSTATE:ACTIVE


Remember that his is a draft pre-display of current defaults.
No consistency-checks for provided values are performed at this stage.
Some missing values are evaluated at a later stage dynamically.
```

The following call starts the initial installation of the VM:

```
ctys -t xen -a create=l:tst482,reuse,b:$PWD,instmode root@lab03
```

25.4.10 Creation of the Raw-VM with XenServer

ffs.

25.5 Installation of the GuestOS - CentOS

(a) Install CentOS-5.5. The following steps are almost identical to all hypervisors. The few exceptions are depicted when required, e.g. change of the QEMU/KVM reboot mode after installation.

(b) From now on for the tests the default settings are used, just the network interfaces are changed to DHCP.

Figure 25.13: Start CentOS installation - Anaconda

Figure 25.14: Proceed CentOS installation

Once the installation is completed for QEMU/KVM the boot mode has to be changed. This could be either processed completely within the so called monitor, or just by rebooting without the 'instmode' suboption. Therefore either change into the monitor mode by typing **Ctrl-Alt-2** and the **quit** command, or by the CANCEL call, The close of the defualt VNC console only will not stop the server:

`ctys -t qemu -a cancel=l:tst219,b:${VMDIRECTORYPATH},poweroff app2`

The syntax for this is similar for all supported hypervisors.

The following call starts the VM into standard operations.

`ctys -t qemu -a create=l:tst219,b:${VMDIRECTORYPATH}  app2`

Figure 25.15: Proceed CentOS installation

The machine now starts into the firstboot mode, where some basic system settings has to beset.

Figure 25.16: CentOS firstboot

Once the basic post-install configuration is finished the machine reboots into the normal operations mode.

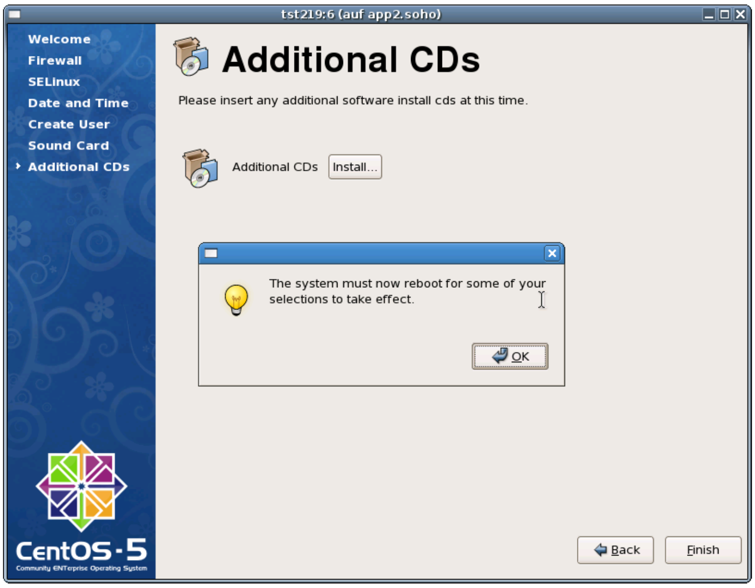


Figure 25.17: Finish post-install

Figure 25.18: Proceed CentOS operational boot

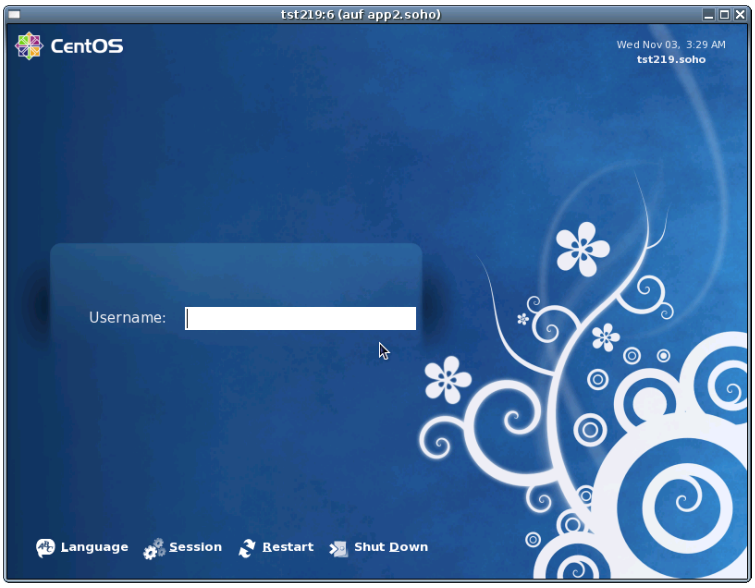


Figure 25.19: Final Login

25.6 Creation of the Inventory - cacheDB

In case of a common mounted NFS filesystem for the pool VMs for simplicity just change into the directory of the VM on any machine. Call for the first check **ctys-vdbgen** with the **–stdio** option for display only.

```
ctys-vdbgen --append --base=$PWD --stdio -- lab02
```

When the result is displyed correctly just call

```
ctys-vdbgen --append --base=$PWD -- lab02
```

The following output should be displayed:

```
Prepare execution-call:

Require DB-PATH,        USE: DEFAULT_DBPATHLST="/homen/acue/.ctys/db/default"
Require DB-PATH,        USE: -o => "/homen/acue/.ctys/db/default"
APPEND mode                : ON(1)
STDIO mode off             : OFF(0)
Set TYPE scope          ADD: DEFAULT="-t ALL"
Preload TYPE set        ADD: DEFAULT="-T ALL"
For splitted operations ADD: DEFAULT="-b sync,seq "
Nameservice cache       OFF: DEFAULT="-c off "
Data cache              OFF: DEFAULT="-C off "

Resulting ENUMERATE     ADD: DEFAULT="-a enumerate=...
    ...matchvstat:active%disabled%empty,machine,\
    b:/mntn/vmpool/vmpool05/vbox/test/tst-ctys/tst137 \
    -C off  -c off  -T ALL  "
```

```
-> generate DB(may take a while)...
-----------------------------------
START:08:38:35
------



------
END:08:39:03
DURATION:00:00:28
-----------------------------------
RET=0
-----------------------------------

Cached data:

  Mode:                        APPEND
  Pre-Appended:                834 records
  Appended:                    1 records
  Fetched Records Raw:          records
  Fetched Records Unique:       records
  Final:                       835 records

-----------------------------------
   ...finished.
```

This shows that only one entry is appended to the existing database with 834 VM-Entries. Now check the database entry by calling:

```
ctys-vhost tst137
```

The following result should be displayed:

```
label |stype|accel|distro|distrorel|os   |osrel|PM   |if |TCP
------+-----+-----+------+---------+-----+-----+-----+---+------------
tst137|VBOX |     |CentOS|1.0.0    |Linux|2.6  |lab02|0  |172.20.2.241
```

25.7 Graphical Start of the Virtual Machine

Now call the menue item for start of the VM 'tst137'.

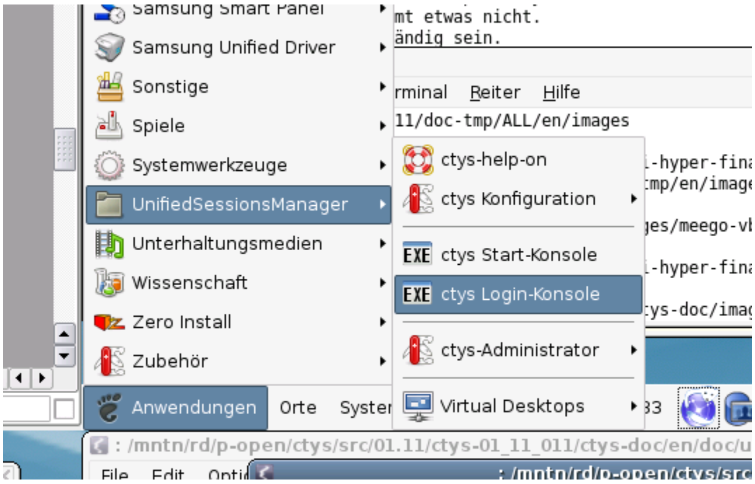


Figure 25.20: CentOS Start Menu

The created cacheDB record for thr VM 'tst137' is now automatically visible in the list of startable virtual machines.

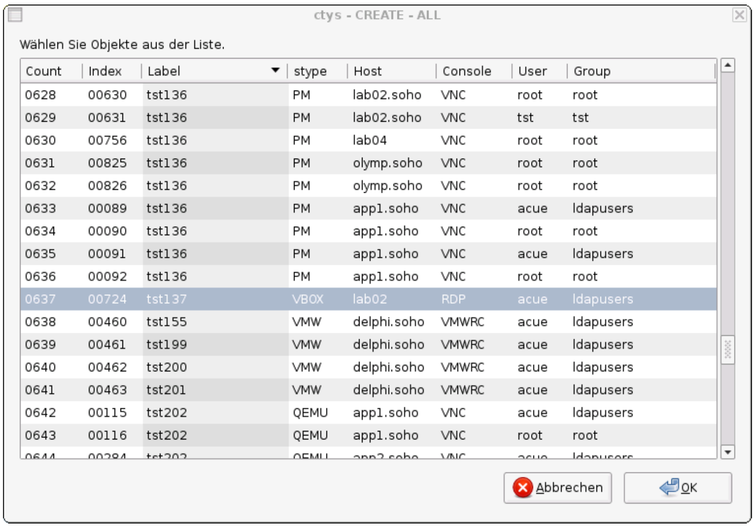


Figure 25.21: CentOS VM Selection

Confirm the selected entry.

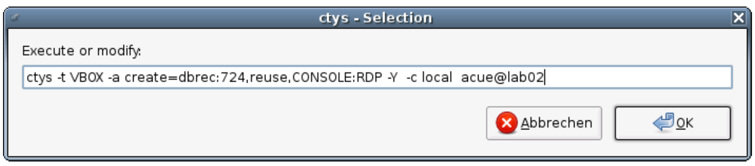


Figure 25.22: CentOS Call Confirmation

25.8 Manage the VM

25.8.1 Common Syntax

25.8.2 Prepare CentOS

(a) Set yum repository in '/etc/yum.repo.d/'
(b) Install the following additional Packages:
   i. openssh-server
   ii. make
   iii. gcc
   iv. kernel-devel
   v. kernel-netbook-devel

Almost absolutely required is a Single-Sign-On facility for OpenSSH. This is due to the required multiple remote remote calls for a number of operational modes. Recommended is either the usage of SSH-Keys, or Kerberos by GSSAPI.

25.8.3 Install UnifiedSessionsManager in GuestOS - CentOS

Apply standard procedure:

```
ctys-distribute -F 2 -P UserHomeCopy root@tst137
```

25.8.4 Open a Remote CLI-Terminal

Call CLI plugin:

```
ctys -t cli -a create=l:tst137 root@tst137
```

25.8.5 Check Plugins States

Call ctys-plugins:

```
ctys-plugins -T all -E
```

25.8.6 Open a Remote RDP-Desktop

ffs.

25.8.7 Open a Remote VNC-Desktop

Call VNC plugin:

```
ctys -t vnc -a create=l:tst137,reuse root@tst137
```

25.8.8 Open a Remote X11-Terminal

Call VNC plugin:

```
ctys -t x11 -a create=l:tst137,reuse root@tst137
```

Chapter 26

Debian

ffs.

Chapter 27

Fedora

ffs.

Chapter 28

MeeGo

.

28.1 General

The current document shows the basic installation of MeeGo, which is basically a derivative of RedHat(TM) Linux variant.

The following environment is used here:

- Debian-5.0.6 with VirtualBox-3.2.10
- CentOS-5.4 with kvm-83 / Qemu-0.9.1
- MeeGo-1.0.0
  The current description is based on the Netbook edition for ia32/i386 architecture. Download the image:
  `Netbooks/meego-netbook-ia32-1.0.0.20100524.1.img`
  Additionally download the packages as required. For the installation and execution of ctys at least the 'openssh-server' is required.
- UnifiedSessionsManager - ctys-01.11.011

.

28.2 Setup of Host-OS and Hypervisor

The installation for the following variants has to be performed by the appropriate standard setup of the HostOS, which quite straight forward:

- Debian with VirtualBox
- CentOS with QEMU/KVM

28.3 Setup of the UnifiedSessionsManager

28.3.1 Install tgz BASE-Package + DOC-Package on Debian

(a) Apply the standard installation procedure:
`ctys-distribute -F 2 -P UserHomeCopy root@tst137`

(b) Open a Remote Shell by call of CLI plugin:
`ctys -t cli -a create=l:tst137 root@tst137`

(c) Check the plugins states by calling ctys-plugins:
`ctys-plugins -T all -E`

28.3.2 Install rpm BASE-Package + DOC-Package on CentOS

The following steps are required for a RPM based setup on CentOS. The installation is relocatable, but located at '/opt', and installed locally by ctys-distribute(1) .

(a) Install BASE package.

`rpm -i ctys-base-01.11.011.noarch.rpm`

(b) Now install a a local version, here by copy. The PATH prefix is important here, particularly in case of updates. The path is resolved to it's actual path by eliminating any symbolic link, and used for consistent link of libraries.

`/opt/ctys-01.11.011/bin/ctys-distribute -F 2 -P UserHomeCopy`

(c) Next the menu is setup.

`ctys-xdg --menu-create`

(d) Now the help is available as eihter a Gnome or KDE menu. Alternatively could be called from the commandline.

28.3.3 Setup of the Gnome Menue

The setup of the Gnome Menu is quite simple, the contained tool **ctys-xdg(1)** sets up a standard menu by the call:

`ctys-xdg --menu-create`

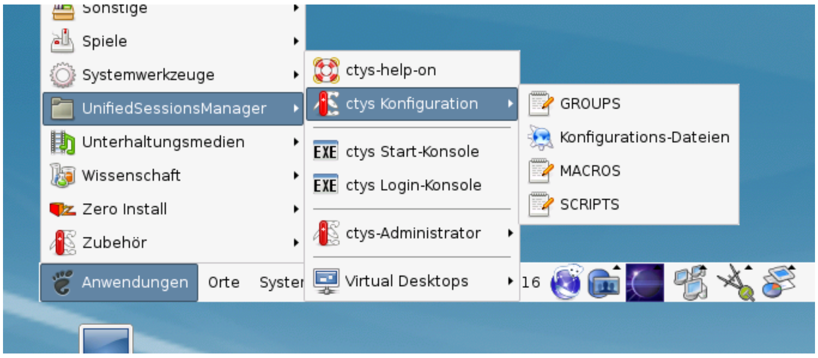


Figure 28.1: Create Menue

The call

`ctys-xdg --menu-cancel`

removes the installed files. For current version no checks for changed files is done.

The menues could be edited and extended by the call

`ctys-xdg --menu-edit`

which opens the related directories for modification of '*.menu', '*.desktop', and '*.directory' files.

28.4 Creation of the the Raw-VM

28.4.1 Creation of the Raw-VM with QEMU/KVM

The demo example VM is here named tst139, this is the hostname of GuestOS too.

(a) Login into the machine where VirtualBox is installed.

```
ssh -X lab02
```

(b) Change to the vmpool and create a directory and change into.

```
mkdir tst139
```

(c) Call the install and configuration utility for VMs. Here some values are set by environment variables, a complete list including the actually assigned values could be displayed by the option **–levo**.

```
ARCH=i386 \
DIST=MeeGo \
DISTREL=1.0.0 \
OS=Linux \
OSREL=2.6 \
ctys-createConfVM -t qemu --label=tst138
```

This call creates a virtual image(hda.img), the call-wrapper(tyt139.sh), and the configuration file(tst139.ctys). The files are created from templates by assigning configuration values either from pre-configured default values, or interactive variation.

(d) Once the set of files is created the virtual machine is prepared for startup. For some other systems complete installation routines are available, e.g. debian and CentOS. The current state could be checked now by the following call.

```
./tst138.sh --console=vnc  --vncaccessdisplay=47 --print --check
```

28.4.2 Creation of the Raw-VM with VirtualBox

The creation of the raw VM is first step to be executed at the host oeprating system. This could be either performed locally or remote and requires the usage of the provided tools by VirtualBox(TM).

(a) Login into the machine where VirtualBox is installed.

```
ssh -X lab02
```

(b) Execute the VirtualBox(TM) console.

```
VirtualBox
```

(c) Create the VM, the machine is called here 'tst137'. The OS is 'Linux', the version is 'Linux 2.6'.

Figure 28.2: Create Virtual Machine

(d) Set RAM to 512MByte.

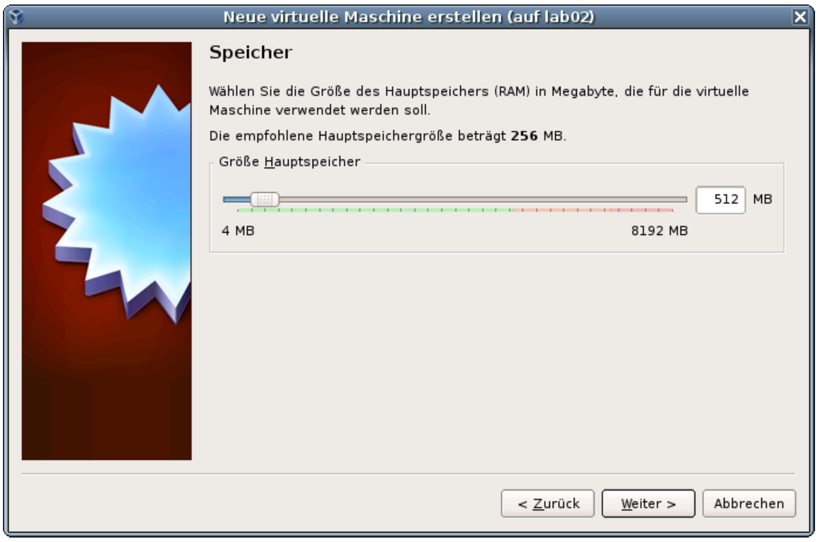


Figure 28.3: Set virtual RAM

(e) Create a virtual HDD, here 8GByte is choosen. When finished the raw VM is present and could be used as required, for basic functions of ctys no additional configuration is required.

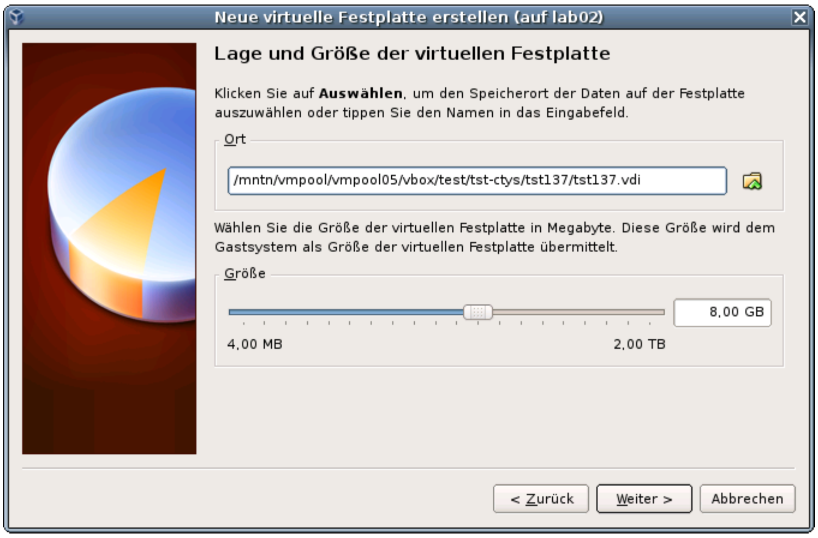


Figure 28.4: Create Virtual HDD

(f) When additional information should be stored coallocated to the VM and scanned automatically into a database, than the tool **ctys-createConfVM(1)** should be applied. This generates additional detailed information related to the specific VM and the inherent guest OS.

Figure 28.5: Check HDD image file

The call could be executed either interactive or automatic.

Call within the same directory for first inspection:

```
ctys-createConfVM -t vbox --label=tst137 --levo
```

This lists some defaults for the specific hypervisor. These could be preconfigured by specific template files within the configuration directory **ctys-createCOnfVM.d**. The result should look like the following:

```
Not all values require to be set, some will be requested later by
dialogue.
Thus it is not neccessary to have values assigned to the complete
displayed set.


Actually used sources for default values:
  no-marker  = Pre-Set value, either from defaults configuration, or
```

```
                by commandline.
  no-value   = Either requested by dialog later, or the defaults of
                the finally called
                application are used.
  (g)        = Dynamically generated.
  (c)        = Read from actual configuration file, e.g. vmx-file.
  (h)        = Used from current host as default.

Applicable modifications:
  blue       = By call option, defines dependency for others.
  green      = By environment, 'could be set almost independent'
                from other values.
  cyan       = By miscellaneous facilities, but is dependent from
                others.
                E.g. LABEL defines by convention the network
                'hostname', thus the TCP/IP params.
                This could ..., but should not be altered!

Most of the missing values will be fetched during actual execution
of this tool by dynamic evaluation.


                VAR name:Initial Value

           C_SESSIONTYPE:VBOX
                   LABEL:tst137
                     MAC:
                      IP:
                  BRIDGE:
                    DHCP:
                 NETMASK:
                     TCP:
                 GATEWAY:

                  EDITOR:acue

                    UUID:b1ff0d36-a552-41ce-be3c-4b3717c2e768 (c)

                    DIST:debian (h)
                 DISTREL:5.0.6 (h)
                      OS:Linux (h)
                   OSREL:2.6.26-2-amd64 (h)

                    ARCH:x86_64 (h)
             ACCELERATOR:HVM (c)
                     SMP:1 (c)
                 MEMSIZE:512 (c)
              KBD_LAYOUT:de

             STARTERCALL:/usr/bin/VirtualBox
```

```
          DEFAULTBOOTMODE:HDD

        DEFAULTINSTTARGET:/mntn/vmpool/vmpool05/vbox/test/tst-ctys...
                          .../tst137/tst137.vdi
  HDDBOOTIMAGE_INST_SIZE:8192M

                  VMSTATE:ACTIVE


Remember that his is a draft pre-display of current defaults.
No consistency-checks for provided values are performed at this stage.
Some missing values are evaluated at a later stage dynamically.
```

The following call generates the appropriate configuration

```
DIST=MeeGo \
DISTREL=1.0.0 \
OS=Linux \
OSREL=2.6 \
MAC=00:50:56:13:11:65 \
IP=172.20.2.241 \
ARCH=i386 \
ctys-createConfVM --label=tst137 -t vbox \
```

The result displayed with **–levo** is:

```
Not all values require to be set, some will be requested later by
dialogue.
Thus it is not neccessary to have values assigned to the complete
displayed set.


Actually used sources for default values:
  no-marker  = Pre-Set value, either from defaults configuration,
               or by commandline.
  no-value   = Either requested by dialog later, or the defaults
               of the finally called
               application are used.
  (g)        = Dynamically generated.
  (c)        = Read from actual configuration file, e.g. vmx-file.
  (h)        = Used from current host as default.

Applicable modifications:
  blue       = By call option, defines dependency for others.
  green      = By environment, 'could be set almost independent'
               from other values.
  cyan       = By miscellaneous facilities, but is dependent
               from others.
               E.g. LABEL defines by convention the network
               'hostname', thus the TCP/IP params.
               This could ..., but should not be altered!

Most of the missing values will be fetched during actual execution
of this tool by dynamic evaluation.
```

```
                 VAR name:Initial Value

            C_SESSIONTYPE:VBOX
                    LABEL:tst137
                      MAC:00:50:56:13:11:65
                       IP:172.20.2.241
                   BRIDGE:
                     DHCP:
                  NETMASK:
                      TCP:
                  GATEWAY:

                   EDITOR:acue

                     UUID:b1ff0d36-a552-41ce-be3c-4b3717c2e768 (c)

                     DIST:MeeGo
                  DISTREL:1.0.0
                       OS:Linux
                    OSREL:2.6

                     ARCH:i386
              ACCELERATOR:HVM (c)
                      SMP:1 (c)
                  MEMSIZE:512 (c)
               KBD_LAYOUT:de

              STARTERCALL:/usr/bin/VirtualBox

          DEFAULTBOOTMODE:HDD

        DEFAULTINSTTARGET:/mntn/vmpool/vmpool05/vbox/test/tst-ctys/...
                          ...tst137/tst137.vdi
   HDDBOOTIMAGE_INST_SIZE:8192M

                  VMSTATE:ACTIVE


Remember that his is a draft pre-display of current defaults.
No consistency-checks for provided values are performed at this stage.
Some missing values are evaluated at a later stage dynamically.
```

When the call is finished the file 'tst137.ctys' with additional configuration information information is stored.

(g) Add the install image as a bootable CD/DVD and set this as the boot device fir the VM:

    `Netbooks/meego-netbook-ia32-1.0.0.20100524.1.img`

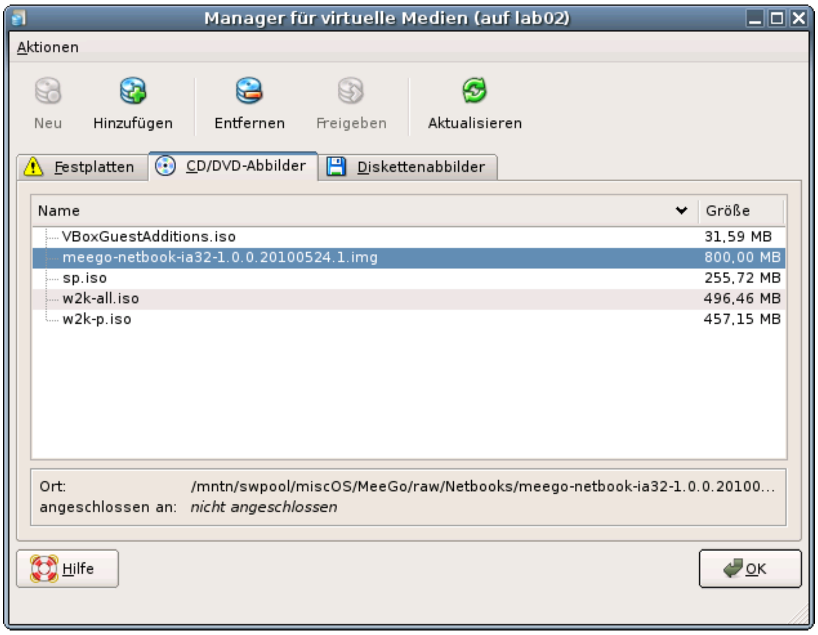


Figure 28.6: Register CD/DVD Install Sources

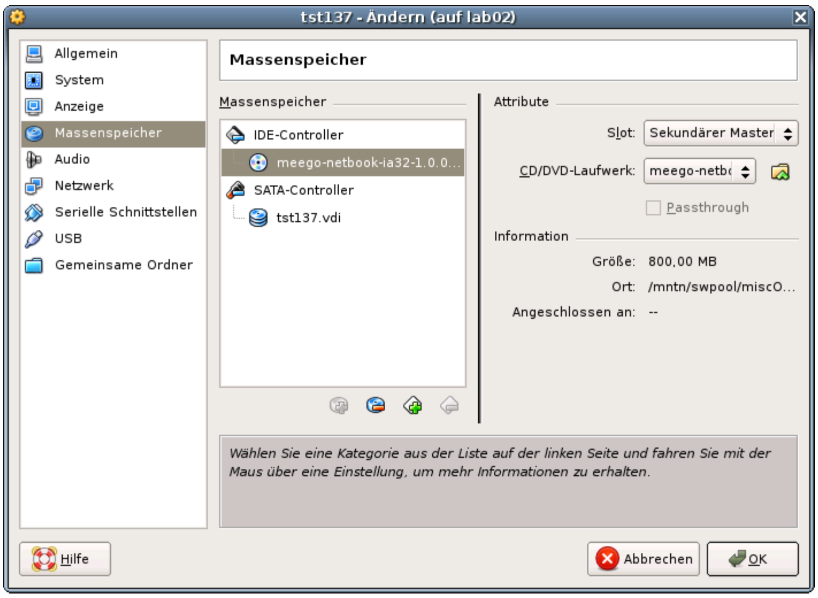


Figure 28.7: Connect CD/DVD Install Sources

(h) Set PAE for virtual CPU.

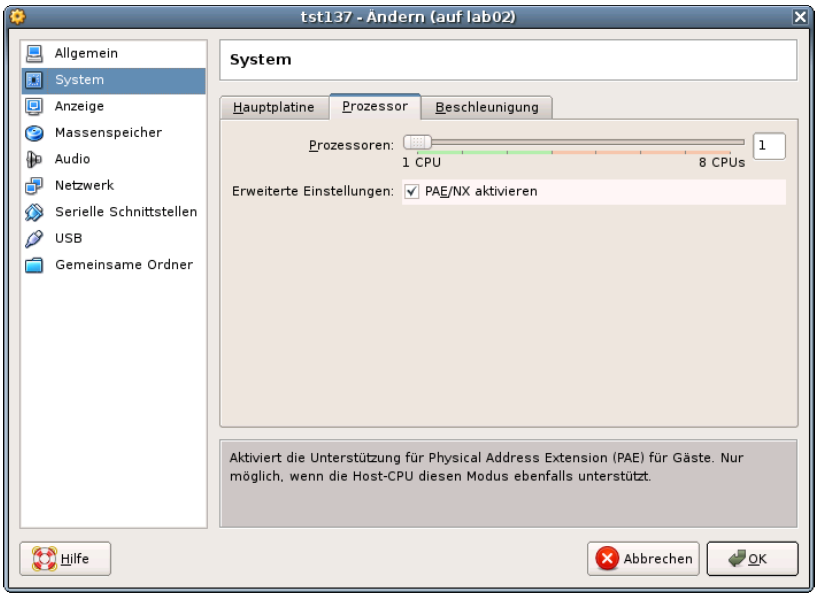


Figure 28.8: VirtualBox VCPU - PAE

28.5 Installation of the GuestOS - MeeGo

(a) The start of the VMs of QEMU/KVM and VirtualBox vary slightly, even tough the following native procedures within the GuestOS are identical.

i. Start QEMU/KVM
The start facilities of the plugin QEMU offer several options. Here the manual local start of the wrapper script is choosen. The first start of MeeGo is proceeded with the SDL console, this has some advantages for the later required 'quick-pressing' of the ESC key for the display of the boot menue. The option –**instmode** sets the bootdevice, here a preconfigured CD/DVD-image for boot.

```
./tst138.sh --console=sdl  --print --instmode
```

An alternate call for the start of the remote installation is:

```
ctys -t qemu \
 -a create=l:tst138,id:${TST138}/tst138.ctys,instmode,console:sdl\
 app2
```

This starts the same by transforming to the target host 'app2' and calling the previous wrapper script.

ii. Start VirtualBox
The start of the VM could be proceeded either by calling VirtualBox, or by the VBOX plugin. But both require in current version the pre-configuration of the appropriate install procedure. Either by mounted install media like a CD/DVD-image, or by usage of PXE for networl based installation.
The folloing call starts the VirtualBox console.

```
VirtualBox
```

The following call call for the starts the remote VM with a VirtualBox console:

```
ctys -t vbox \
 -a create=l:tst137,id:${TST137}/tst138.ctys,console:vbox\
 app2
```

(b) Now boot the VM and choose 'Installation Only'.

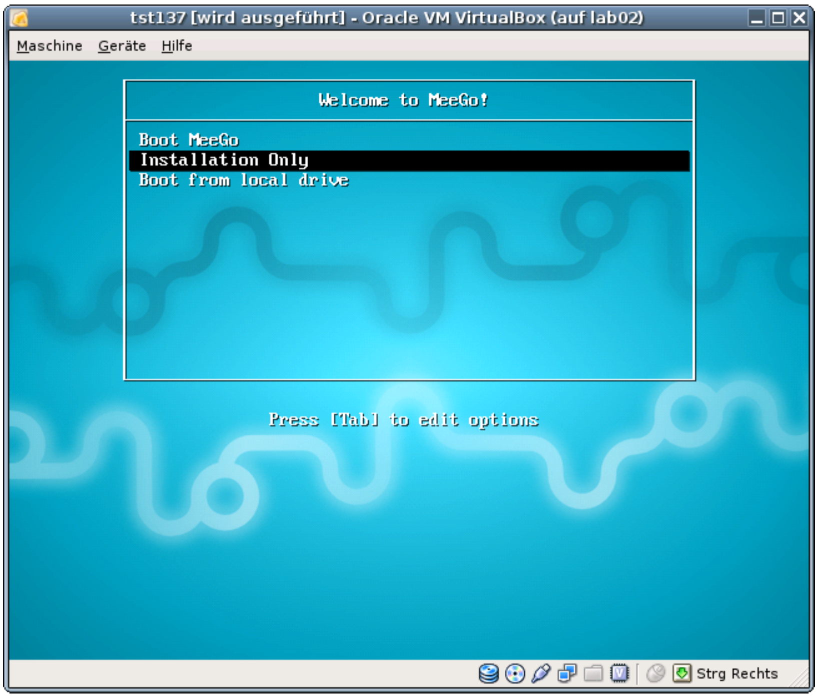


Figure 28.9: Install Menue

After some seconds the MeeGo screen occurs. The install procedure is quite similar to the RHEL based distributions.

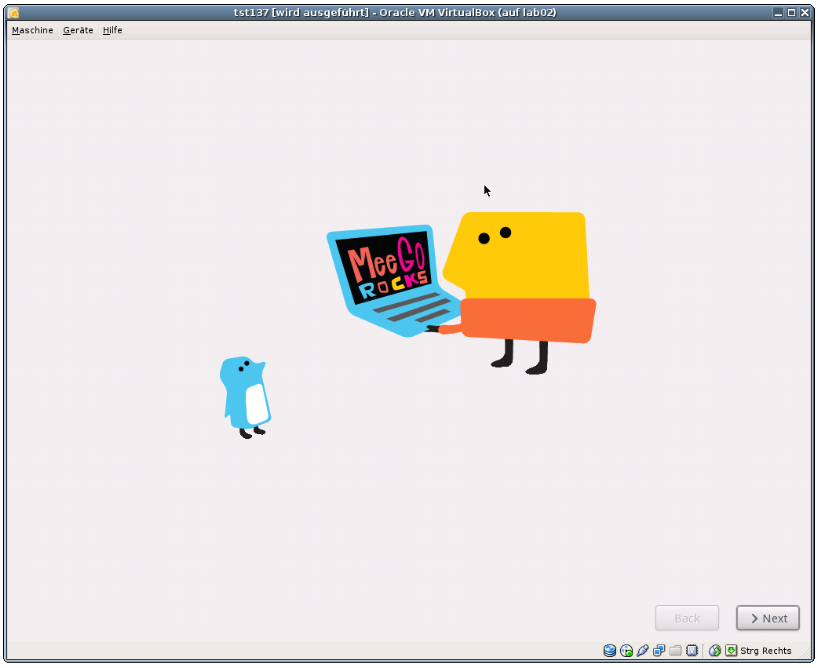


Figure 28.10: MeeGo Screen

(c) When the HARDDISK error is displayed just press init again. In this description the default is choosed.

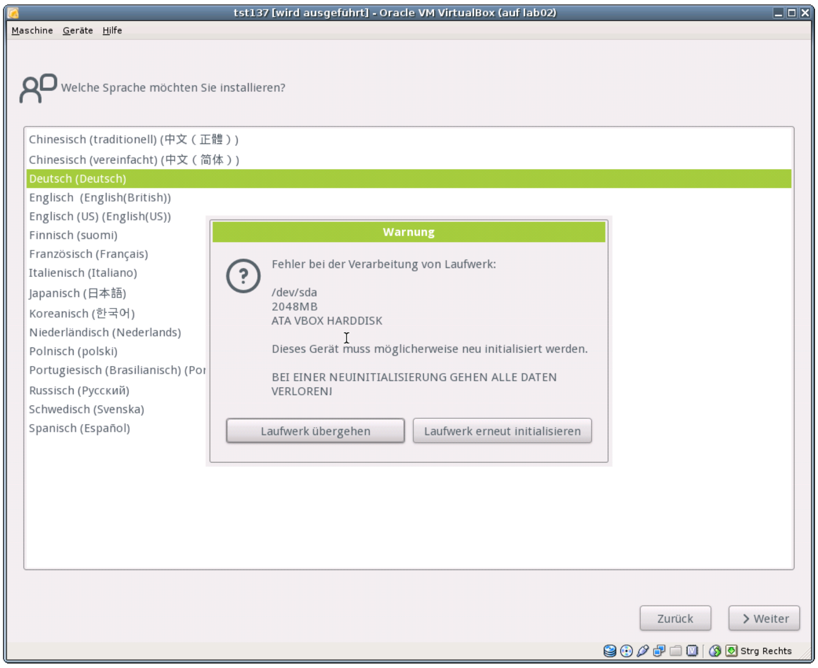


Figure 28.11: HDD-Init

(d) Once the installation is complete, unmount the CD/DVD image and reboot.

i. QEMU/KVM

In order to reboot just shutdown and boot again without the 'instmode' option. The shutdown could be proceeded by the 'quit' command within the monitor. The **monitor mode** is entered e.g. by **Ctrl-Alt-2**. One possible call is:

```
ctys -t qemu \
   -a create=l:tst138,id:${PWD}/tst138.ctys,console:sdl \
   app2
```

ii. VirtualBox
Simply reboot without mounted install media. In case of a fresh start the call could look like:

```
ctys -t vbox \
   -a create=l:tst137,id:${PWD}/tst137.ctys,console:vbox \
   app2
```

The default console is here RDP.

(e) Press ESC once immediately when the display mode first changes, the boot menue should now occur. If this fails just repeat it. Once the boot menue is visible press TAB and edit the boot parameters. Remove the keyword 'quiet' and append 'init 3'.

Now MeeGo should boot and the console login should occur. The default password for the root account is 'meego'.

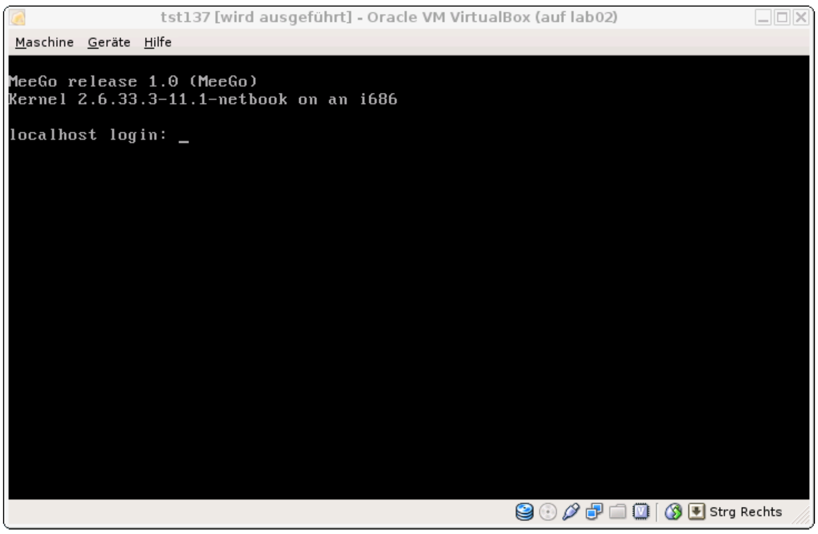


Figure 28.12: MeeGo ASC-II Console

- Set in the inittab the default boot level to 3. Edit '/etc/resolv.conf' and set your nameserver.

- Edit '/boot/extlinux/extlinux.conf' and change:
  - Remove 'quiet'
  - Comment 'menu hidden'

- Comment 'menu auto...'

  In level 3 install the patched library 'libglx.so' by replacing '/usr/lib/xorg/modules/extensions/libglx.so'. And change the mode to 'u+x,g+x,o+x'. Change mode for '/usr/bin/Xorg' to '+s'.

  The required patch and/or library is available from 'http://202.112.3.1/libglx.so'.

  REMARK: This is not the author's link, download is on your own responsibility. Anyhow, the personal test worked in a test-environment and seems to be OK.

(f) Reboot and start **twm** by calling **startx** from the ASC-II console.

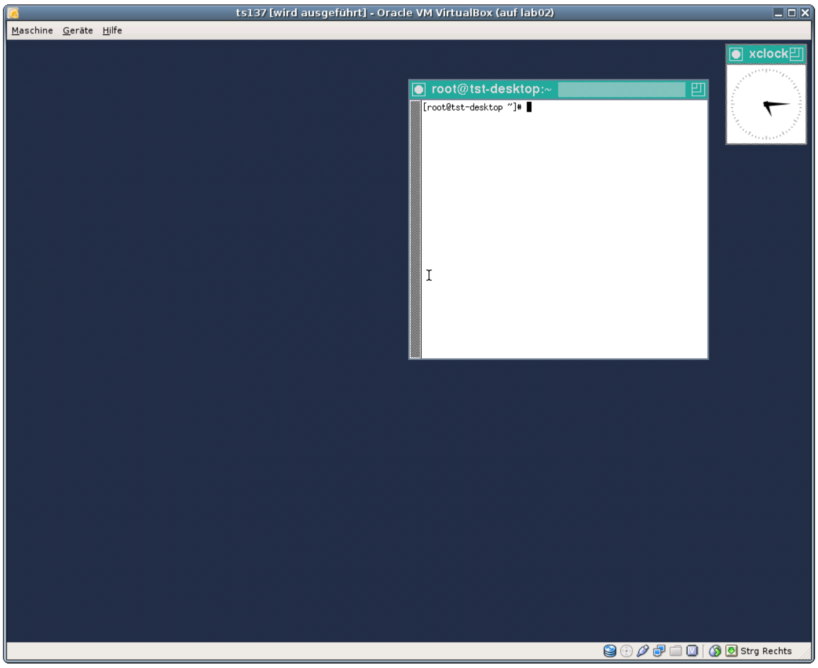


Figure 28.13: MeeGo X11 twm

(g) Call 'firstboot' from within an xterm, and set basic configurations, particularly your keyboard.

(h) **For VirtualBox only:**
Install the **VBoxGuestAdditions** and patch the **/etc/init.d/vboxadd-service** by extending

```
if [ -f /etc/redhat-release ]; then
```

to

```
if [ -f /etc/redhat-release -o -f /etc/meego-release ]; then
```

(i) Reboot. Either set init level to 5, or call from command line 'init 5'.

Figure 28.14: Welcome MeeGo on VirtualBox

Figure 28.15: Welcome MeeGo on QEMU/KVM

Anyhow, for me the instalation currently does not work stable with the original 'moblin-dm'. VirtualBox installation works 'sometimes'(???), Qemu doesn't work at all. But the twm based X11 desktop works perfectly, so basically some drivers must be in place. So I am going to solve this later, and additionally installing than the SDK packages too.

The target for now is to show the integration, therefore the current state is fine.

28.6 Creation of the Inventory - cacheDB

In case of a common mounted NFS filesystem for the pool VMs for simplicity just change into the directory of the VM on any machine. Call for the first check ctys-vdbgen(1) with the –**stdio** option for display only.

```
ctys-vdbgen --append --base=$PWD --stdio -- lab02
```

When the result is displyed correctly just call

```
ctys-vdbgen --append --base=$PWD -- lab02
```

The following output should be displayed:

```
Prepare execution-call:

Require DB-PATH,        USE: DEFAULT_DBPATHLST="/homen/acue/.ctys/db/default"
Require DB-PATH,        USE: -o => "/homen/acue/.ctys/db/default"
APPEND mode                : ON(1)
STDIO mode off             : OFF(0)
Set TYPE scope          ADD: DEFAULT="-t ALL"
Preload TYPE set        ADD: DEFAULT="-T ALL"
For splitted operations ADD: DEFAULT="-b sync,seq "
Nameservice cache       OFF: DEFAULT="-c off "
Data cache              OFF: DEFAULT="-C off "

Resulting ENUMERATE     ADD: DEFAULT="-a enumerate=...
    ...matchvstat:active%disabled%empty,machine,\
    b:/mntn/vmpool/vmpool05/vbox/test/tst-ctys/tst137 \
    -C off  -c off  -T ALL  "

-> generate DB(may take a while)...
-----------------------------------
START:08:38:35
------



------
END:08:39:03
DURATION:00:00:28
-----------------------------------
RET=0
-----------------------------------

Cached data:

  Mode:                      APPEND
  Pre-Appended:              834 records
  Appended:                  1 records
```

```
  Fetched Records Raw:      records
  Fetched Records Unique:   records
  Final:                   835 records

-----------------------------------
   ...finished.
```

This shows that only one entry is appended to the existing database with 834 VM-Entries. Now check the database entry by calling:

```
ctys-vhost tst137
```

The following result should be displayed:

```
label |stype|accel|distro|distrorel|os   |osrel|PM   |if |TCP
------+-----+-----+------+---------+-----+-----+-----+---+------------
tst137|VBOX |     |MeeGo |1.0.0    |Linux|2.6  |lab02|0  |172.20.2.241
```

28.7 Graphical Start of the Virtual Machine

Now call the menue item for start of the VM 'tst137'.

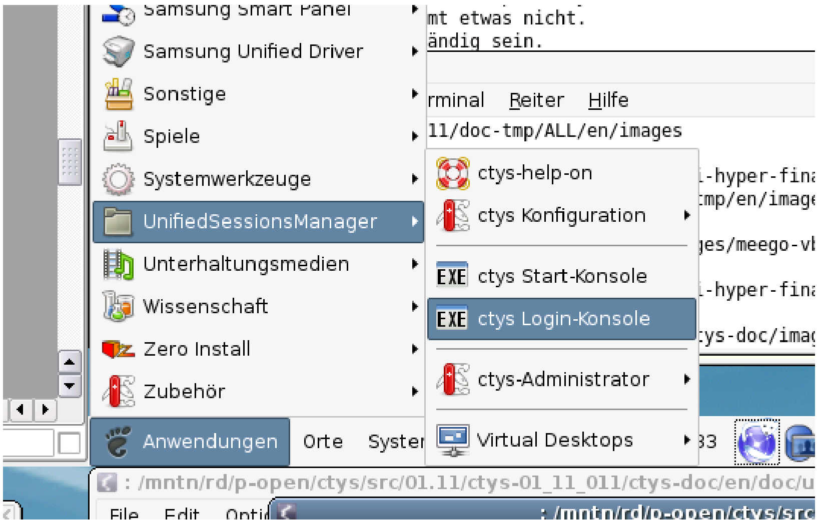


Figure 28.16: MeeGo Start Menue

The created cacheDB record for thr VM 'tst137' is now automatically visible in the list of startable virtual machines.

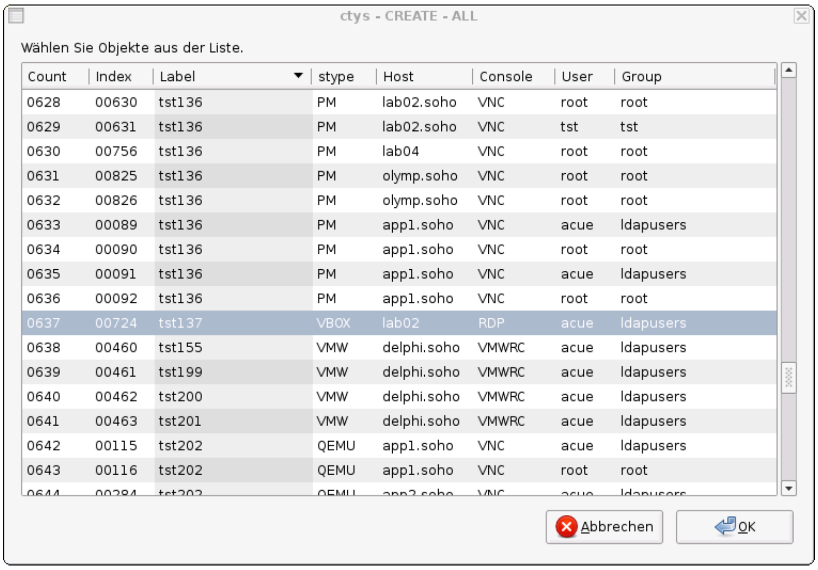


Figure 28.17: MeeGo VM Selection

Confirm the selected entry.

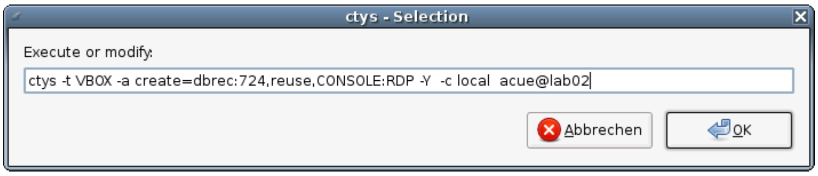


Figure 28.18: MeeGo Call Confirmation

28.8 Manage the VM

28.8.1 Prepare MeeGo

(a) Set yum repository in '/etc/yum.repo.d/'
(b) Install the following additional Packages:
   i. openssh-server
   ii. make
   iii. gcc
   iv. kernel-devel
   v. kernel-netbook-devel

28.8.2 Install UnifiedSessionsManager in GuestOS - MeeGo

Apply standard procedure:

```
ctys-distribute -F 2 -P UserHomeCopy root@tst137
```

28.8.3 Open a Remote Shell

Call CLI plugin:

```
ctys -t cli -a create=l:tst137 root@tst137
```

28.8.4 Check Plugins States

Call ctys-plugins:

```
ctys-plugins -T all -E
```

28.8.5 Open a Remote X11-Terminal

ffs.

28.8.6 Open a Remote VNC-Desktop

ffs.

Chapter 29

OpenBSD

ffs.

Chapter 30

OpenSUSE

ffs.

Chapter 31

QNX

ffs.

Chapter 32

Ubuntu

ffs.

Chapter 33

uCLinux

ffs.

Part V

Automation Procedures

Chapter 34

Common Session Options

34.1 Opening multiple ctyss

34.1.1 Multiple Calls

This creates 3 vncviewer desktops to 3 different hosts and one VMware-VM session, which will be done with one call when provided within one call or script:

```
ctys -a create=l:CONSOLE -g :1 host01
ctys -a create=l:CONSOLE -g :2 host02
ctys -a create=l:CONSOLE -g :3 host03
ctys -t vmw -a CREATE=f:'vmware/openbsd-001.vmx' \
     -g '600x400+2660+100' host01
```

34.1.2 One call

This call combines multiple session into one call, it opens 4 sessions on two hosts in seperate VNC desktops.

```
ctys  \
 --                                                    \
 host02'(-a create=l:F1,REUSE -g 400x400:3)'         \
 host02'(-a create=l:F2,REUSE -g 400x400+500+500:3)'\
 host01'(-a create=l:F3,REUSE -g 400x400:4)'         \
 host01'(-a create=l:F4,REUSE -g 400x400+500+500:4)'
```

This combines different sizes of client windows.

```
ctys \
 -b on                                                  \
 --                                                     \
 host02'(-a create=l:chk1,REUSE -g 400x400+10+10:4)'  \
 host02'(-a create=l:chk2,REUSE -g 400x340+10+440:4)' \
 host02'(-a create=l:chk3,REUSE -g 400x400+420+10:4)' \
 host02'(-a create=l:chk4,REUSE -g 440x940+830+10:4)' \
 host02'(-a create=l:chk5,REUSE -g 800x130+10+820:4)' \
 host02'(-a create=l:chk6,REUSE -g 300x150+440+440:4)'\
 host02'(-a create=l:chk7,REUSE -g 300x150+490+630:4)'
```

34.2 Different resolution on client and server

This displays a local window of size "500x500" on screen 4. The resolution of the server (vncserver or Xclient) is "1800x1800":

```
ctys \
 -a create=l:CONSOLE \
 -g '500x500:4' \
 -r '1800x1800' \
 host01
```

34.3 Dynamic move of sessions window

For moving a vncviewer window circular across all screens and additionally within the screen, just the following lines are required.

```
#
#cycle screens
#
local \_scr=; #screens
local \_pos=; #offsets within that screen
local \_POSLST="+600+100 +700+150 +800+200";
local \_POSLST="\$\_POSLST +700+300 +600+200";
#
for (( \_scr=0; \_scr<7; \_scr++));do
  #cycle within screen
  for pos in \$\_POSLST;do
    ctys \
      -a create=l:TST01,REUSE \
      -g "500x500\${\_pos}:\$scr" \
      host01
    sleep 3
  done
done
```

The first call simply CREATE a ctys-session by starting vncserver, whereas the following calls just attach a vncviewer-client to the already running ctys-session. If the ctys-session already exists, no CREATE of ctys-session will be done.

This usage of REUSE here is utilized for the specific default behaviour of RealVNC(and tightVNC), where by default no desktop sharing is allowed. The standard behaviour is therefore after authorization to vncserver, to kill all (so one) previously started viewer-instances. When sharing is activated, REUSE just creates the requested new client, but does not touch the previous clients.

Thus for performance reasons here the call of REUSE is preferred, instead of using RECONNECT key, which first explicitly "kills" all previously locally started vncviewer instances on the current host of vncviewer execution for the for targeted ctys-session, which is the vncserver to be attached to.

The following does the same, but opens an additional VMware-Server session, which will be placed on a the screen next to VNC. Here it is performed by two seperate calls.

```
#
#cycle screens

#
local \_scr=; #screens
local \_pos=; #offsets within that screen
local \_POSLST="+600+100 +700+150 +800+200";
local \_POSLST="\$\_POSLST +700+300 +600+200";
#
for (( \_scr=3; \_scr<7; \_scr++));do
  #cycle within screen
  for pos in \$\_POSLST ;do

    ctys -a create=l:TST01,REUSE            \
         -g "500x500\${\_pos}:\${scr}"         \
         host01

    ctys -t VMW                             \
         -a create=l:TST01-VMW,RECONNECT    \
         -g "500x500\${\_pos}:\$((scr+1))"     \
         host01

    sleep 3
  done
done
```

The following does the same, but performs just one call for both.

```
#
#cycle screens
#
local \_scr=; #screens
local \_pos=; #offsets within that screen
local \_POSLST="+600+100 +700+150 +800+200";
local \_POSLST="\$\_POSLST +700+300 +600+200";
#
for (( \_scr=3; \_scr<7; \_scr++));do
  #cycle within screen
  for pos in \$\_POSLST;do

    ctys                                        \
       host01"(                                 \
          -a create=l:TST01,REUSE               \
          -g 500x500\${\_pos}:\${scr}              \
       )"                                       \
       host01"(                                 \
          -t VMW                                \
          -a create=l:TST01-VMW,RECONNECT       \
          -g "500x500\${\_pos}:\$((scr+1))"        \
       )"

    sleep 3
  done
done
```

The following does the same and adds XEN. Which has to be executed on a different host than VMware is.

```
#
#cycle screens
#
local \_scr=; #screens
local \_pos=; #offsets within that screen
local \_POSLST="+600+100 +700+150 +800+200";
local \_POSLST="\$\_POSLST +700+300 +600+200";
#
for (( \_scr=3; \_scr<7; \_scr++));do
  #cycle within screen
  for pos in \$\_POSLST;do

    ctys                                    \
        host01"(                            \
            -t VNC                          \
            -a create=l:TST01,REUSE         \
            -g 500x500\${\_pos}:\${scr}        \
        )"                                  \
        host01"(                            \
            -t VMW                          \
            -a create=l:TST01-VMW,RECONNECT \
            -g "500x500\${\_pos}:\$((scr+1))"  \
        )"                                  \
        host02"(                            \
            -t XEN                          \
            -a create=l:TST01-XEN,RECONNECT \
            -g "500x500\${\_pos}:\$((scr+2))"  \
        )"

    sleep 10 #give some more time
  done
done
```

Now adding multiple desktop support.

```
#
#cycle screens
#
local \_scr=; #screens
local \_pos=; #offsets within that screen
local \_POSLST="+600+100 +700+150 +800+200";
local \_POSLST="\$\_POSLST +700+300 +600+200";
#
for (( \_scr=3; \_scr<7; \_scr++));do
  for desk in rd prod 3;do  #rd=1, prod=2, 3=admin
  #cycle within screen
    for pos in \$\_POSLST;do

      ctys                                   \
        host01"(                             \
          -t VNC                             \
          -a create=l:TST01,REUSE            \
          -g 500x500\${\_pos}:\${scr}            \
          -W \${desk}                         \
        )"                                   \
        host02"(                             \
          -t VMW                             \
          -a create=l:TST01-VMW,RECONNECT    \
          -g "500x500\${\_pos}:\$((scr+1))"      \
          -W \${desk}                         \
        )"                                   \
        host03"(                             \
          -t XEN                             \
          -a create=l:TST01-XEN,RECONNECT    \
          -g "500x500\${\_pos}:\$((scr+2))"      \
          -W \${desk}                         \
        )"

    done

    sleep 10 #give some more time
  done
done
```

34.4 Spanning Multiple Physical Screens

Currently it seems though, that in case of RealVNC 4.1.2 on CentOS-5.0 the following behaviour is given:

- A desktop with size bigger than a physical(better to say X11-configured) screen has to be configured when starting the vncserver. So e.g. the geometry parameter for the vncserver call could be "2560x1024" on a adjecent pair of "1280x1024" screens.
- When opening the vncviewer the desktop will be sized as given by the vncserver, but the displayed window is restricted by the current screen dimension.

As it seems to be, the oversized window is restricted to the size of the screen of the monitor in the middle of the calculated area of required size and offset. This is the device on which the resulting screen will be displayed. The maximum size given as an option to vncviewer will not change this behaviour.

This seems to be a bug of vncviewer, but anyhow, the horizontal scrollbar shows the correct proportions and the windows could be expanded manually to it's full size, spanning multiple screens.

This behaviour is a major drawback for automatic and final configuration of desktop layouts, requiring additional manual interaction. But it does only effect, when sizing a window bigger than a screen.

So with the following call a window could be created, which is spanning 4 adjecent screens, but has to expanded to it's final size manually:

```
ctys \
 -a create=l:BIG13 \
 -g '5120x1024:3' \
 host01
```

For debugging of the remote actions the following could be called:

```
ctys \
 -a create=l:BIG13 \
 -g '5120x1024:3' \
 -- \
 '(-d 6)' host01
```

Which sets the debugging level on the remote host "host01" to "6".

This spans now 2 screens:

```
ctys \
 -a create=l:CONSOLE \
 -g "2560+0:1" \
 host01
```

This resets to one screen:

```
ctys \
 -a create=l:CONSOLE \
 -g ":1" \
 host01
```

34.5 CREATE with tree-search for unique IDs

One of the real smart features of ctys is it's ability to use any of it's unique IDs for automatic search for related vmx-file within a defined subtree, which is by default HOME. Any directory prefix could be provided. The applicable IDs are

- the vmx-filename with any partial relative path-prefix
- the UUID
- the LABEL, a.k.a displayName of name/DomainName.

The following calls are possible:

- This starts a VMware session with the first matched UUID within subtree "vmware/dir2" relative to HOME on host01.

```
ctys \
 -t vmw \
 -a create=uuid:0101010...01,    \
    basepath:vmware/dir2 \
 host01
```

- This starts a VMware session with the first matched LABEL within subtree "vmware/dir2" relative to HOME on host01.

```
ctys \
 -t vmw \
 -a create=label:MatchMe,basepath:vmware/dir2    \
 host01
```

- This starts a VMware session with the first matched vmx-file within subtree "vmware/dir2" relative to HOME on host01.

```
ctys \
 -t vmw \
 -a create=filename:dir2/OpenBSD-01/OpenBSD-01.vmx,    \
    basepath:vmware \
 host01
```

- This starts a VMware session with the vmx-file on host01.

```
ctys \
 -t vmw \
 -a create=pname:\$HOME/dir2/OpenBSD-01/OpenBSD-01.vmx     \
 host01
```

34.6 Some session related calls

34.6.1 ENUMERATE

Enumerate the current available sessions of type VMW, where UUIDs are displayed additionaly to labels and vmx-files. The scan for vmx-files begins relative to the callers HOME directory within the subdirectories: "vmware/dir2" and "vmware/dir3" on host01.

```
ctys \
 -t vmw \
 -a enumerate=UUID,vmware/dir2\%vmware/dir3 \
 host1
```

34.6.2 LIST

List all current active sessions, where all attributes are visible, the fullpathname for vmx-files is displayed. Initially all type-plugins are loaded and listed due to load state. All CLIENT and SERVER processes located on the host01 are displayed. Warnings are suppressed.

```
ctys \
 -a list=all,fullpath,both \
 -W \
 -T \
 all host01
```

HINT:Loading of all present plugins could exhaust shell resources.

34.6.3 SHOW

This shows the current dynamic state of the remote hosts, therefore basic system information for OS, MACHINE, RAM, processes(by top), and current ALARMS of lm_sensors if installed.

```
ctys -a show host0{1,2}
```

34.6.4 INFO

This displays information related to static data of selected hosts, which contains installed OS, CPU-info, RAM-info, VNC-info, and wmctrl-info.

For practical purposes the CPU-Flags: VT-x, AMD-V, and PAE are displayed.

```
ctys -a info host0{1,2}
```

34.7 Some ctys related calls

34.7.1 Display Version and available plugin

The following call enumerates all actually loaded plugins as set by default:

```
ctys -v
```

The following call enumerates all actually available plugins and their versions.

```
ctys -v -T all
```

Chapter 35

Custom CLI

35.1 Groups

One of the most valuable features for the setup of custom desktops is the groups feature. Any setup of a X11 desktop including multiple desktops could be prepared and executed.

The following example illustrates the setup of a workspace with some basic remote desktops for their management. In this example the user root is used without encapsulation of the calls by sudo or ksu. The interactive call for the group file named "admin" is:

```
ctys admin
```

Where the group file has the following content. This example shows a number

```
#
#This groups contains all machines in the
#management group.
#
root@machine1'(-t vnc -a create=reuse,l:MACHINE1 \
               -g 1268x872:A20 -W admin -b 1,2)'
root@machine2'(-t vnc -a create=reuse,l:MACHINE2 \
               -g :A30 -W admin -b 1,2)'
root@machine3'(-t vnc -a create=reuse,l:MACHINE3 \
               -g :A00 -W admin -b 1,2)'
root@machine4'(-t vnc -a create=reuse,l:MACHINE4 \
               -g :A01 -W admin -b 1,2)'
root@machine5'(-t vnc -a create=reuse,l:MACHINE5
               -g :A21 -W admin -b 1,2)'
```

of specifics to be considered. First of all, the ssumption is made, that each root account has to be authorized by an interactive password request. Therefore the "-b 1,2" option is required. This forces a background but sequential execution, which causes a non-intermixed and sequential password request, but once authorized, the process is detached from the console. Thus multiple interactive password requests for daemons could be provided. The second point to be recognized is the complete support of the type and action information within the context options .

This allows the intermixed usage of several session types within one call. The next point is the usage of the "-D admin" option for the display of all admin tasks on the "admin" workspace, where the alias "admin" is a custom definition by the user.

The same resulting call could be provided by the following variant, which is more flexible, but requires therefore some additional call parameters. The interactive call for the group file named "admin" is now: The "-l root" option sets the target user

```
ctys -l root -b 1,2 admin
```

to be used for all resulting targets from the group "admin". The "-b 1,2" sets the value to be used for all subsequent internal subcalls. The group file has now the following content.

```
#
#This groups contains all machines in the
#management group.
#
machine1'(-t vnc -a create=reuse,l:MACHINE1 \
          -g 1268x872:A20 -W admin)'
machine2'(-t vnc -a create=reuse,l:MACHINE2 \
          -g :A30 -W admin)'
machine3'(-t vnc -a create=reuse,l:MACHINE3 \
          -g :A00 -W admin)'
machine4'(-t vnc -a create=reuse,l:MACHINE4 \
          -g :A01 -W admin)'
machine5'(-t vnc -a create=reuse,l:MACHINE5
          -g :A21 -W admin)'
```

35.2 Tables

35.2.1 Common Tables for ENUMERATE and LIST

This example shows a table definition for LIST and ENUMERATE the CPORT - ClientPort - of a plugin. This is usually the port for access by vncviewer.

```
"tab_gen:3_Label_10%%macro:F_STYPE%%9_cport_5%%\
  1_PM_15%%6_MAC_18%%7_TCP_15"
```

The defintion is used literally within LIST action as:

```
ctys \
 -a list=tab_gen:3_Label_10%%macro:F_STYPE%%\
    9_cport_5%%1_PM_15%%6_MAC_18%%7_TCP_15\
 lab00 lab01
```

The defintion is used literally within ENUMERATE action as:

```
ctys \
 -a enumerate=tab_gen:3_Label_10%%macro:F_STYPE%%\
    9_cport_5%%1_PM_15%%6_MAC_18%%7_TCP_15\
 lab00 lab01
```

The same definition could be used to define a macro for a table.

```
TAB_CPORT=tab_gen:3_Label_10%%macro:F_STYPE%%\
  9_cport_5%%1_PM_15%%6_MAC_18%%7_TCP_15
```

The macro and used within LIST action as:

```
ctys -a list=macro:TAB_CPORT lab00 lab01
```

The result of the LIST action example is:

```
    Label   |stype |cport|PM        |MAC               |TCP
    --------+------+-----+----------+------------------+------------
    testx11 |CLI   |     |lab00.soho|                  |
    testx11 |CLI   |     |lab00.soho|                  |
    LAB00   |VNC   |5901 |lab00.soho|                  |
    tst     |VNC   |5902 |lab00.soho|                  |
    Domain-0|XEN   |     |lab00.soho|                  |
    tst101  |XEN   |5928 |lab00.soho|00:50:56:13:11:41 |
    lab00   |PM    |     |lab00.soho|00:0E:0C:35:F8:48 |192.168.1.71
    Domain-0|XEN   |     |lab01.soho|                  |
    lab01   |PM    |     |lab01.soho|00:0E:0C:C3:CD:12 |192.168.1.72
    tst000  |VNC   |     |ws2.soho  |                  |
    tst001  |VNC   |     |ws2.soho  |                  |
```

The macro used within ENUMERATE action as:

```
ctys -a enumerate=macro:TAB_CPORT lab00 lab01
```

The result of the ENUMERATE action example is:

```
Label     |stype|cport|PM        |MAC              |TCP
----------+-----+-----+----------+-----------------+-------------
sparc-1   |QEMU |     |lab00.soho|00:50:56:13:11:49|QEMU
sparc-1   |QEMU |     |lab00.soho|00:50:56:13:11:49|QEMU
coldfire-t|QEMU |     |lab00.soho|00:50:56:13:11:49|QEMU
arm-test  |QEMU |     |lab00.soho|00:50:56:13:11:49|QEMU
linux     |QEMU |     |lab00.soho|00:50:56:13:11:49|QEMU
small     |QEMU |     |lab00.soho|00:50:56:13:11:49|QEMU
qemu-tst01|QEMU |     |lab00.soho|00:50:56:13:11:52|QEMU
arm-test  |QEMU |     |lab00.soho|00:50:56:13:11:49|QEMU
small     |QEMU |     |lab00.soho|00:50:56:13:11:49|QEMU
sparc-1   |QEMU |     |lab00.soho|00:50:56:13:11:49|QEMU
sparc-1   |QEMU |     |lab00.soho|00:50:56:13:11:49|QEMU
linux     |QEMU |     |lab00.soho|00:50:56:13:11:49|QEMU
coldfire-t|QEMU |     |lab00.soho|00:50:56:13:11:49|QEMU
linux001  |VMW  |     |lab00.soho|00:50:56:15:11:01|192.168.1.150
linux002  |VMW  |5977 |lab00.soho|00:50:56:15:11:02|192.168.1.151
tst100    |XEN  |     |lab00.soho|00:50:56:13:11:40|
tst100    |XEN  |     |lab00.soho|00:50:56:13:11:40|
tst100    |XEN  |     |lab00.soho|00:50:56:13:11:40|
tst100    |XEN  |     |lab00.soho|00:50:56:13:11:40|
tst101    |XEN  |     |lab00.soho|00:50:56:13:11:41|
tst101    |XEN  |     |lab00.soho|00:50:56:13:11:41|
lab00     |PM   |     |lab00.soho|00:0E:0C:35:F8:48|192.168.1.71
```

35.2.2 RAW Tables by MACHINE

35.2.3 Combined MACROS and Tables

The following examples shows stored table options for LIST and ENUMERATE action within macros.

Several of the predefined tables are generic, thus containing fields available from the display methods "ctys -a ENUMERATE=...", "ctys -a LIST=...", and "ctys-vhost -o tab_gen:...", thus these could be used literally in all of them, just the wrapping call convention varies a little. Thus the macro definitions for generic tables are used unaltered, only adaptive call convention superpositioning macros are defined.

The pure text replacement by macros allows for additional suboptions, when these are seamless concatenated by usage of an intermediate field seperator. Thus the content of a macro could be expanded without altering it's definition, even though in this draft version no specific macro-seperator is defined.

```
#!/bin/bash #4syncolors
########################################################
#
#PROJECT:      Unified Sessions Manager
#AUTHOR:       Arno-Can Uestuensoez -
#                unifiedsessionsmanager@protonmail.com
#MAINTAINER:   Arno-Can Uestuensoez -
#                unifiedsessionsmanager@protonmail.com
#SHORT:        ctys
#CALLFULLNAME:Commutate To Your Session
#LICENCE:      GPL3
#VERSION:      01_06_001a10
#
########################################################
#
#Copyright(C) 2008 Arno-Can Uestuensoez
#  (UnifiedSessionsManager)
#
#This program is free software: you can redistribute
#it and/or modify it under the terms of the GNU General
#Public License as published by the Free Software
#Foundation, either version 3 of the License, or (at
#your option) any later version.
#
#This program is distributed in the hope that it will
#be useful, but WITHOUT ANY WARRANTY; without even the
#implied warranty of MERCHANTABILITY or FITNESS FOR A
#PARTICULAR PURPOSE.  See the GNU General Public
#License for more details.
#
#You should have received a copy of the GNU General
#Public License along with this program.  If not,
#see <http://www.gnu.org/licenses/>.
#
########################################################
```

```
#####################################################
#
#Atoms with appropriate sizes.
#
#
#ContainingMachine(1);
#SessionType(2);
#Label(3);
#ID(4);
#UUID(5);
#MAC(6);
#TCP(7);
#DISPLAY(8);
#ClientAccessPort(9);
#VNCbasePort(10);
#
F_PM        = 1_PM_15
F_STYPE     = 2_stype_10
F_LABEL     = 3_label_10
F_ID        = 4_ID_25_L
F_UUID      = 5_UUID_32
F_MAC       = 6_MAC_18
F_TCP       = 7_TCP_15
F_DISP      = 8_DISP_4
F_CPORT     = 9_cport_5_L
F_SPORT     = 10_sport_5_L

#####################################################
#
#Specific additional MACROS for LIST
#PID(11);
#UID(12);
#GUID(13);
#C/S-Type(14)
#
F_PID       = 11_pid_5
F_UID       = 12_uid_8
F_GUID      = 13_gid_8
F_CST       = 14_cst_1


#####################################################
#
#Specific additional MACROS for ENUMERATE
#
#VNCbaseport(11);
#Distro(12);
#Distrorel(13);
#OS(14);
#OS(15);
#VersNo(16);
#SerialNo(17);
#Category(18)
#VMstate(19)
#hyperrel(20)
#StackCap(21)
#StackReq(22)
#HWcap(23)
#HWreq(24)
#execloc(25)
```

```
#reloccap(26)
#SSH(27)
#rsrv(28)
#rsrv(29)
#rsrv(30)
#rsrv(31)
#rsrv(32)
#rsrv(33)
#CTYSrel(34)
#netmask(35)
#Gateway(36)
#Relay(37)
#Arch(38)
#Platform(39)
#VRAM(40)
#VCPU(41)
#ContextStg(42)
#UserStrg(43)
#
F_VNCBASE   = 11_vncbase_7
F_DIST      = 12_distro_12
F_DISTREL   = 13_distrorel_15
F_OS        = 14_os_10
F_OSREL     = 15_osrel_10
F_VERNO     = 16_verno_9
F_SERNO     = 17_serno_14
F_CATEGORY  = 18_category_8
F_VMSTATE   = 19_VMstate_9
F_HYPERREL  = 20_hyperrel_15
F_STACKCAP  = 21_StackCap_15_B
F_STACKREQ  = 22_StackReq_15_B
F_HWCAP     = 23_HWcap_60_B
F_HWREQ     = 24_HWreq_25_B
F_EXECLOC   = 25_execloc_15
F_RELOCCAP  = 26_reloccap_8
F_SSHPORT   = 27_SSH_5
F_RSRV6     = 28_r_1
F_RSRV7     = 29_r_1
F_RSRV8     = 30_r_1
F_RSRV9     = 31_r_1
F_RSRV10    = 32_r_1
F_IFNAME    = 33_if_7
F_CTYSREL   = 34_CTYSrel_10
F_NETMASK   = 35_netmask_15
F_GATEWAY   = 36_Gateway_15
F_RELAY     = 37_Relay_15
F_ARCH      = 38_Arch_6
F_PLATFORM  = 39_Platform_10
F_VRAM      = 40_VRAM_5
F_VCPU      = 41_VCPU_4
F_CSTRG     = 42_ContextStg_20_B
F_USTRG     = 43_UserStrg_20_B
```

```
####################################################
#
#DEFAULT for ctys-vhost, when no "-o" option is
#selected. Change this carefully, otherwise ctys-vhost
#might come into trouble.
#
TAB_CTYS_VHOST_DEFAULT=tab_gen:macro:F_LABEL%%\
macro:F_STYPE%%macro:F_DIST%%macro:F_DISTREL%%\
macro:F_OS%%macro:F_OSREL%%macro:F_PM%%macro:F_TCP

listdefault=-a list=macro:TAB_CTYS_VHOST_DEFAULT
ldefault=-a list=macro:TAB_CTYS_VHOST_DEFAULT

enumdefault=-a enumerate=macro:TAB_CTYS_VHOST_DEFAULT
edefault=-a enumerate=macro:TAB_CTYS_VHOST_DEFAULT

vhostdefault=-o macro:TAB_CTYS_VHOST_DEFAULT
vdefault=-o macro:TAB_CTYS_VHOST_DEFAULT


####################################################
#
#Basic hypervisor state
#
TAB_HYPER=tab_gen:macro:F_LABEL%%macro:F_STYPE%%\
macro:F_VMSTATE%%macro:F_OS%%macro:F_OSREL%%\
macro:F_ARCH%%macro:F_VCPU%%macro:F_VRAM

enumhyper=-a enumerate=macro:TAB_HYPER
ehyper=-a enumerate=macro:TAB_HYPER
vhosthyper=-o macro:TAB_HYPER
vhyper=-o macro:TAB_HYPER

####################################################
#
#Basic stack state
#
TAB_STACKSTAT=tab_gen:macro:F_LABEL%%macro:F_STYPE%%\
macro:F_VMSTATE%%macro:F_OS%%macro:F_OSREL%%\
macro:F_STACKCAP%%macro:F_STACKREQ

enumstack=-a enumerate=macro:TAB_STACKSTAT
estack=-a enumerate=macro:TAB_STACKSTAT

vhoststack=-o macro:TAB_STACKSTAT
vstack=-o macro:TAB_STACKSTAT

####################################################
#
#connections with PID
#
# LABEL STYPE  DISP  CPORT  SPORT  PID PM  TCP
#
TAB_LST_CONNPID=tab_gen:macro:F_LABEL%%\
macro:F_STYPE%%macro:F_CST%%macro:F_DISP%%\
macro:F_CPORT%%macro:F_SPORT%%\
macro:F_PID%%macro:F_PM%%macro:F_TCP

connpid=macro:TAB_LST_CONNPID
listconnpid=-a list=macro:TAB_LST_CONNPID
```

```
#####################################################
#
#connections
#
# LABEL STYPE  DISP  CPORT  SPORT  PM  TCP
#
TAB_ENUMLST_CONNECT=tab_gen:macro:F_LABEL%%\
macro:F_STYPE%%macro:F_DISP%%macro:F_CPORT%%\
macro:F_SPORT%%macro:F_PM%%macro:F_TCP

conn=macro:TAB_ENUMLST_CONNECT
lconn=-a list=macro:TAB_ENUMLST_CONNECT
econn=-a enumerate=macro:TAB_ENUMLST_CONNECT
vconn=-o macro:TAB_ENUMLST_CONNECT

#####################################################
#
#interfaces
#
# LABEL STYPE  PM  TCP MAC
#
TAB_ENUMLIST_INTERFACES=tab_gen:macro:F_LABEL%%\
macro:F_STYPE%%macro:F_PM%%macro:F_TCP%%macro:F_MAC

interfaces=macro:TAB_ENUMLIST_INTERFACES
lif=-a list=macro:TAB_ENUMLIST_INTERFACES
eif=-a enumerate=macro:TAB_ENUMLIST_INTERFACES
vif=-o macro:TAB_ENUMLIST_INTERFACES

#####################################################
#
#conffiles
#
# LABEL STYPE  TCP  MAC  ID
#
TAB_ENUMLIST_CONF=tab_gen:macro:F_LABEL%%\
macro:F_STYPE%%macro:F_TCP%%macro:F_MAC%%4_ID_50_L

conf=macro:TAB_ENUMLIST_CONF
lconf=-a list=macro:TAB_ENUMLIST_CONF
econf=-a enumerate=macro:TAB_ENUMLIST_CONF
vconf=-o macro:TAB_ENUMLIST_CONF

#####################################################
#
#ids
#
# LABEL STYPE  TCP  MAC  UUID  ID
#
TAB_ENUMLIST_ID=tab_gen:macro:F_LABEL%%macro:F_STYPE%%\
macro:F_TCP%%macro:F_MAC%%macro:F_UUID%%macro:F_ID

id=macro:TAB_ENUMLIST_ID
lid=-a list=macro:TAB_ENUMLIST_ID
eid=-a enumerate=macro:TAB_ENUMLIST_ID
vid=-o macro:TAB_ENUMLIST_ID
```

Usage with VMSTATE

The predefined macros could be expanded by additional attributes, when these are resulting in an overall semantically correct definition One example application is the usage of the provided standard macro "enumhyper", which defines a table containing the main hypervisor and GuestOS attributes. The standard of ctys for ENUMERATE call is:

```
ctys -t vmw macro:enumhyper
```

The result of the ENUMERATE action is:

```
label     |stype |VMstate|os     |osrel|Arch|VCPU|VRAM
----------+------+-------+-------+-----+----+----+-----
tst117    |VMW   |ACTIVE |Linux  |2.6  |    |    |
tst115    |VMW   |ACTIVE |Solaris|10   |    |    |
tst116    |VMW   |ACTIVE |Linux  |2.6  |    |    |
tst112    |VMW   |ACTIVE |Linux  |2.6  |    |    |
tst003    |VMW   |ACTIVE |Linux  |2.6  |    |    |
tst005    |VMW   |ACTIVE |Linux  |2.6  |    |    |
tst103    |VMW   |ACTIVE |Linux  |2.6  |    |    |
tst106    |VMW   |ACTIVE |Linux  |2.6  |    |    |
tst111    |VMW   |ACTIVE |OpenBSD|4.2  |    |    |
tst120    |VMW   |ACTIVE |FreeBSD|6.1  |    |    |
tst128    |VMW   |ACTIVE |NetBSD |4.0  |    |    |
tst002    |VMW   |ACTIVE |Linux  |2.6  |    |    |
tst132    |VMW   |ACTIVE |Linux  |2.6  |    |    |
tst133    |VMW   |       |Linux  |2.6  |    |    |
tst155    |VMW   |       |OpenBSD|4.3  |    |    |
tst109    |VMW   |ACTIVE |OpenBSD|4.0  |    |    |
GRP02     |VMW   |       |other  |     |    |    |
GRP01-open|VMW   |       |other  |     |    |    |
GRP02     |VMW   |       |other  |     |    |    |
linux001  |VMW   |       |Linux  |2.6  |    |    |
linux002  |VMW   |       |Linux  |2.6  |    |    |
linux001  |VMW   |       |Linux  |2.6  |    |    |
```

This displays the default output of "MATCHVSTAT=ACTIVE%EMPTY". When the "MATCHVSTAT=ACTIVE" subset is required only the following call-extension to the macro could be applied.

```
ctys -t vmw macro:enumhyper,matchvstat:active
```

The "-t vmw" option sets SESSIONTYPE to VMW, thus all other are ignored. Additionally the macro "macro:enumhyper" is expanded by ",matchvstat:active" in order to display onyl active sessions, where the field VMSTATE has the kiteral value "ACTIVE". It is important to recognize the comma "," as field seperator here. The result of the ENUMERATE action is:

```
label |stype |VMstate  |os        |osrel |Arch  |VCPU|VRAM
------+------+---------+----------+------+------+----+-----
tst117|VMW   |ACTIVE   |Linux     |2.6   |      |    |
tst115|VMW   |ACTIVE   |Solaris   |10    |      |    |
tst116|VMW   |ACTIVE   |Linux     |2.6   |      |    |
tst112|VMW   |ACTIVE   |Linux     |2.6   |      |    |
tst003|VMW   |ACTIVE   |Linux     |2.6   |      |    |
tst005|VMW   |ACTIVE   |Linux     |2.6   |      |    |
tst103|VMW   |ACTIVE   |Linux     |2.6   |      |    |
tst106|VMW   |ACTIVE   |Linux     |2.6   |      |    |
tst111|VMW   |ACTIVE   |OpenBSD   |4.2   |      |    |
tst120|VMW   |ACTIVE   |FreeBSD   |6.1   |      |    |
tst128|VMW   |ACTIVE   |NetBSD    |4.0   |      |    |
tst002|VMW   |ACTIVE   |Linux     |2.6   |      |    |
tst132|VMW   |ACTIVE   |Linux     |2.6   |      |    |
tst109|VMW   |ACTIVE   |OpenBSD   |4.0   |      |    |
```

The previous example is now expanded to display additionally VMs with the state "TESTDUMMY", which are the test configuration files contained in the current version. These contain testpattern only, not foreseen to be used.

```
ctys -t vmw macro:enumhyper,matchvstat:active%testdummy
```

The result of the ENUMERATE action now contains additionally the available test-pattern:

```
label     |stype |VMstate  |os     |osrel |Arch|VCPU|VRAM
----------+------+---------+-------+------+----+----+-----
tst117    |VMW   |ACTIVE   |Linux  |2.6   |    |    |
tst115    |VMW   |ACTIVE   |Solaris|10    |    |    |
tst116    |VMW   |ACTIVE   |Linux  |2.6   |    |    |
tst112    |VMW   |ACTIVE   |Linux  |2.6   |    |    |
tst003    |VMW   |ACTIVE   |Linux  |2.6   |    |    |
tst005    |VMW   |ACTIVE   |Linux  |2.6   |    |    |
tst103    |VMW   |ACTIVE   |Linux  |2.6   |    |    |
tst106    |VMW   |ACTIVE   |Linux  |2.6   |    |    |
tst111    |VMW   |ACTIVE   |OpenBSD|4.2   |    |    |
tst120    |VMW   |ACTIVE   |FreeBSD|6.1   |    |    |
tst128    |VMW   |ACTIVE   |NetBSD |4.0   |    |    |
tst002    |VMW   |ACTIVE   |Linux  |2.6   |    |    |
tst132    |VMW   |ACTIVE   |Linux  |2.6   |    |    |
tst109    |VMW   |ACTIVE   |OpenBSD|4.0   |    |    |
tst01vmx  |VMW   |TESTDUMMY|1      |1     |1   |1   |2
tst05vmx  |VMW   |TESTDUMMY|1      |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1      |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1      |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1      |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1      |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1      |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1      |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1      |1     |1   |1   |2
```

The same where the rable is sorted by the 4-th column of the final result.

```
ctys -t vmw macro:enumhyper,matchvstat:active%testdummy,sort:4
```

The result of the ENUMERATE action now is additionally sorted by the "os" filed.

```
label     |stype |VMstate  |os      |osrel |Arch|VCPU|VRAM
----------+------+---------+--------+------+----+----+-----
tst01vmx  |VMW   |TESTDUMMY|1       |1     |1   |1   |2
tst05vmx  |VMW   |TESTDUMMY|1       |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1       |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1       |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1       |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1       |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1       |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1       |1     |1   |1   |2
tst11vmx  |VMW   |TESTDUMMY|1       |1     |1   |1   |2
tst120    |VMW   |ACTIVE   |FreeBSD |6.1   |    |    |
tst002    |VMW   |ACTIVE   |Linux   |2.6   |    |    |
tst003    |VMW   |ACTIVE   |Linux   |2.6   |    |    |
tst005    |VMW   |ACTIVE   |Linux   |2.6   |    |    |
tst103    |VMW   |ACTIVE   |Linux   |2.6   |    |    |
tst106    |VMW   |ACTIVE   |Linux   |2.6   |    |    |
tst112    |VMW   |ACTIVE   |Linux   |2.6   |    |    |
tst116    |VMW   |ACTIVE   |Linux   |2.6   |    |    |
tst117    |VMW   |ACTIVE   |Linux   |2.6   |    |    |
tst132    |VMW   |ACTIVE   |Linux   |2.6   |    |    |
tst128    |VMW   |ACTIVE   |NetBSD  |4.0   |    |    |
tst109    |VMW   |ACTIVE   |OpenBSD |4.0   |    |    |
tst111    |VMW   |ACTIVE   |OpenBSD |4.2   |    |    |
tst115    |VMW   |ACTIVE   |Solaris |10    |    |    |
```

Chapter 36

Applications of ctys-vhost

36.1 Database Generation ctys-vdbgen and companions

The cache database required by "ctys-vhost" is generated by the call of "ctys-vdbgen" , which internally simply calls "ctys -a ENUMERATE" . The output is literally stored into a first-level cache and correlated with additional information to a cacheDB for performance reasons. The resulting performance gain is simply more than 100-1000 for the majority of practical cases. The combination of the GROUP feature with "ctys-vdbgen" could be utilized to get smart calls and the build of different views as independent name-resolution subsets controlled by the "-p" option.

Anyhow, the presented set of functionality is mainly covering the collection of data from distributed VM stacks, including append of data to existing databases. The usage of multiple databases is covered. A temporary database sould be created, while the main is still processed, the later update simply requires a copy of the "enum.fdb" file, and a call to ctys-vhost(without the "-s" supress update option). The management of the data could be simply done by any ASC-II editor or any spread-sheet application like from OpenOffice or Microsoft(TM) MS-Office.

36.1.1 Initial Database

The following call generates the cacheDB for the user "tstusr" by usage of a predefined GROUP .

```
time ctys-vdbgen $USER-all
```

It is recommended to use the depicted trace call for a reduced and easy understandable progress tracking, which should be stored within an eventually prepared macro file.

```
time ctys-vdbgen $USER-all'(-d 2,w:0,s:16,p)'
```

For additional description of the utilized trace options refer to "-d" common option.

The following data is initially displayed.

```
Prepare execution-call:

Require DB-PATH,        USE: -o => "/homen/acue/.ctys/db/tststate1"
Stripped CLI            CLI:    => " tst-subgroup-00(-d 2,w:0,s:16,p)"
APPEND mode off => REPLACE : OFF(0)
STDIO mode off             : OFF(0)
Set TYPE scope          ADD: DEFAULT="-t ALL"
Preload TYPE set        ADD: DEFAULT="-T ALL"
Should speed-up         ADD: DEFAULT="-b sync,parallel "
Nameservice cache       OFF: DEFAULT="-c off "
Data cache              OFF: DEFAULT="-C off "

Resulting ENUMERATE     ADD: DEFAULT="-a enumerate=matchvstat:active,\
   machine,b:~%/etc/ctys.d -C off -c off -b sync,parallel \
   -T ALL -t ALL  tst-subgroup-00(-d 2,w:0,s:16,p)"

RESULTING              CALL:"ctys -a enumerate=matchvstat:active,\
   machine,b:~%/etc/ctys.d -C off -c off -b sync,parallel -T ALL \
   -t ALL  tst-subgroup-00(-d 2,w:0,s:16,p)>.../db/tststate1/enum.fdb"

-> generate DB(may take a while)...
-----------------------------------
START:19:50:30
------
ctys:acue@ws2.soho:1106:lib/groups:548:INFO:1:checkAndSetIsHostOrGroup:\
    delayed member expansion to next level for \
    SUB-GROUP=tst-subgroup-00-ws1
ctys:acue@ws2.soho:1106:CORE/EXEC:628:INFO:1:finalizeExecCall:delayed\
    member executed as SUBGROUP=acue@tst-subgroup-00-ws1
ctys:acue@app2.soho:28179:ENUMERATE/enumerate:234:16:2:scan4sessions:\
    START=19:50:46
ctys:acue@app2.soho:28179:ENUMERATE/enumerate:240:16:2:scan4sessions:\
    START-CLI=19:50:46
ctys:acue@app2.soho:28179:ENUMERATE/enumerate:249:16:2:scan4sessions:\
    FINISHED-CLI=19:50:46
ctys:acue@app2.soho:28179:ENUMERATE/enumerate:250:16:2:scan4sessions:\
    DURATION-CLI=00:00:00
...
...
```

Figure 36.1: TAB_CPORT by ctys-vhost

After this several minutes could be required for the now ongoing and probbable large ENUMERATE joblist. For this version no progress indicator is given. During the execution of the job(s) several output may be presented, which gives some hints in case of erroneous execution. The following data is initially displayed.

```
Warning: No xauth data; using fake authentication data for
   X11 forwarding.
/usr/bin/xauth:  error in locking authority file
   /homen/vadmin/.Xauthority
X11 connection rejected because of wrong authentication.
X11 connection rejected because of wrong authentication.
```

Figure 36.2: TAB_CPORT by ctys-vhost

When the job is completed, the final message is shown.

```
------
RET=0
------
...finished.

real    3m45.744s
user    0m3.880s
sys     0m4.260s
```

Figure 36.3: TAB_CPORT by ctys-vhost

As depicted, the job took about 4minutes. The results of the test example included here 19 remote accounts, on 8 machines and leads to 1297 VM configurations including fully functional VMs. As listed for first analysis of the cached content.

`ctys-vhost -S list`

This shows the complete cache, including the group definitions.

```
Current file-databases for "tstuser":
8k  160 /homen/tstuser/.ctys/db/default/macmap.fdb
248k  1297 /homen/tstuser/.ctys/db/default/enum.fdb

Current group files of:

CTYS_GROUPS_PATH   = /homen/tstuser/.ctys/groups
:/mntn/rd/p-int/tools/ctys/src/ctys.01_06_001a09/conf/ctys/groups

/homen/tstuser/.ctys/groups
4k   19/0      19  tstuser-all
4k   19/0      19  tstuser-test
4k   78/0      78  admin
4k   27/0      27  allVMs
4k   11/0      11  errgrp1
4k   26/0      26  test-lab
4k    2/0       2  tstgrp1
4k    9/0       9  tstgrp2
4k   55/0      55  tst-qemu

/mntn/rd/p-int/tools/ctys/src/ctys.01_06_001a09/conf/ctys/groups
4k    2/0       2  tst-001
4k    6/0       6  tst-002
4k    4/3      11  tst-nest-000-a00
...
...
...
4k    1/0       1  tst-nest-002-a20
```

Figure 36.4: TAB_CPORT by ctys-vhost

It has to be recognized here, that entries of multiple mounted NFS directories would occur once for each match. The second specific is the handling of multiple NIC interfaces within a VM, which will result in one entry for each NIC.

36.1.2 Append Data to present Database

The following workflow is used to append data for a new processing node to the cacheDB.

(a) Make a copy of the raw file database "enum.fdb" to a temporary cacheDB directory, e.g. "$HOME/.ctys/db/tmp". It is also possible to work completly on a copy, than additionally at least the "macmap.fdb" is required.

```
mkdir $HOME/.ctys/db/tmp

cp  \
  $HOME/.ctys/db/default/enum.fdb \
  $HOME/.ctys/db/tmp
```

(b) Call the update of the data by "append" for data of session type "QEMU".

```
ctys-vdbgen \
  -t qemu \
  -T all \
  --progress
  --cacheDB=$HOME/.ctys/db/tmp \
  --append \
  --base=qemu  \
  lab00'(-d 2,s:16,w:0,p)'
```

(c) Check the actual collected data by a simple diff.

```
diff \
  $HOME/.ctys/db/default/enum.fdb \
  $HOME/.ctys/db/tmp/enum.fdb
```

(d) Copy the new file database into your working directory.

```
cp  \
  $HOME/.ctys/db/tmp/enum.fdb \
  $HOME/.ctys/db/default
```

(e) Call ctys-vhost, any display call on the actual data may detect the time-stamp and performs a rebuild of the 2.stage cacheDB. The following leads to all "VMs" of type "QEMU" on the machine "lab00"

```
ctys-vhost -o l,pm,id QEMU tst lab00
```

When using a macro e.g. the following is displayed.

```
ctys-vhost macro:vconf,sort QEMU tst lab00
```

```
label |stype|TCP          |MAC              |ID
------+-----+-------------+-----------------+--------------------------------
tst000|QEMU |192.168.1.130|00:50:56:13:11:30|qemu/tst-ctys/tst000/tst000.conf
tst001|QEMU |192.168.1.131|00:50:56:13:11:31|qemu/tst-ctys/tst001/tst001.conf
...
...
tst147|QEMU |192.168.1.101|00:50:56:13:11:6f|qemu/tst-ctys/tst147/tst147.conf
tst148|QEMU |192.168.1.102|00:50:56:13:11:70|qemu/tst-ctys/tst148/tst148.conf
```

The available macro definitions of combined macros only could be displayed with

```
ctys-macros -c -D
```

The processing of the "enum.fdb" only requires frequently some seconds, whereas the(in current version deactivated) groups-cache may take several minutes, because any level of nested includes is valid as an entry point. Though even a small set of nested groups, particularly within stacked environments, could lead to a considerable number of "trees", but this is of course what CACHES are for.

(f) That's it.

For resonable scans it should not take more than 1-2 minutes. Could require longer on outdated machines, and/or for deep filesystem structures.

36.2 Group Addressing

36.2.1 Preconfigured Task-Groups

The usage of preconfigured groups has several advantages, where the first application might be the usage to easily update the cacheDB. A group could be used as a replacement for the <target-application-entity> . The following rebuilds the whole cacheDB for "$USER" by one call, once the group "$USER-all" is configured.

```
ctys-vdbgen $USER-all
```

36.3 DHCP Poll

The call of ctys-extractMAClst works on a dhcpd.conf as required for the widely common DHCP daemon. In current version it just extracts static entries for a fixed assignment of MAC-to-IP-Addresses. Dynamic assigned leases for address pools are not yet supported. So for using two pre-requisites have to be fulfilled:

(a) Any participating VM and PM has to have it's own assigned fixed mapping of an DNS-Name, TCP/IP-Address, and MAC-Address.

(b) Access to the dhcpd.conf file, e.g. as a copy is required. Another approach might be that for individual users with local caches a prepared macmap.fdb could be provided.

The following call generates the default target " /.ctys/db/default/macmap-fdb" mapping database:

```
ctys-extractMAClst -P /etc/dhcpd.conf
```

A second tool which is called "ctys-extractAPRlst" generates a macmap.fdb too, but it polls the local accessible DHCP server by using ping and evaluates the arp caches. Thus it will work on one segment only, which might be in the era of switched-LANs not a problem. And requires the hosts to be on-line, which is for the "ctys-extractMAClst" tool not required. The following call generates the default target " /.ctys/db/default/macmap-fdb" mapping database:

```
ctys-extractARPlst -P host01 host02 group01 host03 group02
```

When the whole domain has to be scanned, which could include any host, so particularly address-pools too, than the following could be applied:

```
for i in ‘host -l <mydomain>|awk ’{print \$1;}’‘;do
  ctys-extractMAClst \$i
done
```

Anyhow, due to the simple format of macmap.fdb, which MS-Excel compatible, this mapping file-db macmap.fdb could be edited manually too.

36.4 PXELinux - Address Translation

One example application for "ctys-vhost" is the resolution of PXE IP entries in the required specific form for PXELinux. Once the database is setup with "ctys-vdbgen" and "ctys-extractMAClst" or "ctys-extractARPlst" , the generation of

PXELinux addresses becomes an action of 0.x seconds, check it out!

The match for the regexpr should be as expected, this has to be considered when choosing the search string. Here it is assumed, that the only host/dns-names matching "inst" are of form "inst00[0-9]", and are foreseen as temporary install addresses for raw-templates, installed by kickstart-files. The following shell call generates and displays for any inst-machine an suitable PXELinux IP-Address. Let me mention, that this requires almost only the initial-overhead of 0.3seconds from an database of 120 DHCP entries when using macmaponly. When the MAC-Mapping cache is not used, instead the static cache DB based on pre-cached enum.fdb is used, the performance is the initial overhead of 0,5seconds from an database of about 450 VM entries.

```
\#!/bin/bash
for i in ‘ctys-vhost -C macmaponly -o m,d -M all inst‘;do
   echo -n "\${i\#*;}=\${i\%;*}->";
   gethostip \${i\#*;};
done"
```

This results to the output:

```
inst000=00:50:56:16:11:00->inst000.soho 192.168.1.80 C0A80150
inst001=00:50:56:16:11:01->inst001.soho 192.168.1.81 C0A80151
inst002=00:50:56:16:11:02->inst002.soho 192.168.1.82 C0A80152
...
...
```

So the MAC address could be set, the alphanumeric IP address could be configured, and the automatic configuration will be performed.

36.5 Formatted output by Generic Tables

The output of ctys-vhost could be formatted by the same means as for LIST and ENUMERATE of ctys. Therefore the "-o" output flag supports the common "tab_gen" option. The usage of macros is supported in exactly the same way as for ctys. The following call

```
ctys-vhost -o macro:TAB_CPORT olymp xen acue
```

displays the result:

```
Label |stype|cport|PM        |MAC              |TCP
------+-----+-----+----------+-----------------+-------------
tst100|XEN  |     |olymp.soho|00:50:56:13:11:40|192.168.1.220
tst101|XEN  |     |olymp.soho|00:50:56:13:11:41|192.168.1.221
tst104|XEN  |     |olymp.soho|00:50:56:13:11:44|192.168.1.224
```

Figure 36.5: TAB_CPORT by ctys-vhost

The output here is of course similar to the ENUMERATE ("TAB_CPORT by ENUMERATE") content, and is actually a pre-cached output of ENMUERATE. The complemantary output of dynamic runtime data is presented by LIST action in "TAB_CPORT by LIST" . The default output when the "-o" option is suppressed is to display a table named as "TAB_CTYS_VHOST_DEFAULT". which produces e.g. for the call

```
ctys-vhost QEMU acue app1
```

The output

```
label   |stype|distro  |distrorel   |os     |osrel   |PM       |TCP
--------+-----+--------+------------+-------+--------+---------+-------------
tst102  |QEMU |debian  |4.0r3       |Linux  |2.6.16  |app1.soho|192.168.1.222
tst000  |QEMU |CentOS  |5           |Linux  |2.6.18  |app1.soho|192.168.1.130
tst001  |QEMU |CentOS  |5           |Linux  |2.6.18  |app1.soho|192.168.1.131
...
...
```

Figure 36.6: Default table for ctys-vhost: TAB_CTYS_VHOST_DEFAULT

36.6 Filter by awk-Regular Expressions

36.6.1 Datastreams and Match Conditions

The cacheDB is organized set of lines within a flat file,where each record is a semicolon seperated list of attributes. This is compatible to almost any office spreadsheet program and could be easily imported to any relational DBMS. The actual column names including their canonical position index could be printed with the TITLEIDX option. Even though a column structure is present the majority of queries will be performed to the whole line as a flat string pattern, which delivers a fast match.

A repetitive application of multiple selection strings for application of the intermidiate results of the previous is supported. Alternatively some awk boolean operations could be applied.

Each given select string "<selector>" is evaluated by the awk-regexpr "$0 s", where the variable "s" has the value "<selector>". A valid value could be "." for any character, or it could be "(test104|tst153)" for matching any record which contains either "test104" of "tst153".

36.6.2 Wildcards and Ambiguity

The arguments of "ctys-vhost" are bypassed tranparently to a awk-regexp and are evaluated each as a chained filter on the results of the previous argument. The call with regular expressions in general matches on "$0" of awk, thus may lead to some specific results, when not considered thoroughly.

A typical pitfall might be, when the user has for each type of VM it's own subdirectory, with lowercase directory name of the hypervisor. For QEMU namely "qemu". When the following query is executed, this just scans the database and matches any record containing the string "qemu".

```
ctys-vhost -o ids qemu vadmin
```

The result is the subset of IDs(conf-file-paths) for "qemu", where the string "vadmin" matches. The following result is displayed

```
/homen/vadmin/qemu/tst/arm-test/arm-test.ctys
/homen/vadmin/qemu/tst/small/small.ctys
/homen/vadmin/qemu/tst/sparc-test/sparc-1.ctys
/homen/vadmin/qemu/tst/sparc-test/sparc-2.ctys
/homen/vadmin/qemu/tst/linux/linux.ctys
/homen/vadmin/qemu/tst/coldfire-test-0.1/coldfire-test-0.1.ctys
/homen/vadmin/vmware/develop/build/linux001.i386-4qemu/linux001.vmx
/homen/vadmin/vmware/develop/build/linux001.i386-4qemu.20070207_1/linux001.vmx
```

Figure 36.7: ctys-vhost awk-regexpr-1

which contains the lines

```
...
/homen/vadmin/vmware/develop/build/linux001.i386-4qemu/linux001.vmx
/homen/vadmin/vmware/develop/build/linux001.i386-4qemu.20070207_1/linux001.vmx
```

Figure 36.8: ctys-vhost awk-regexpr-1

This is due to a simple string match, where the production VM for the QEMU sources itself is matched as a "QEMU-VM". To avoid this the following regexpr could be used.

`ctys-vhost -o ids /qemu/ vadmin`

This results in:

```
/homen/vadmin/qemu/tst/arm-test/arm-test.ctys
/homen/vadmin/qemu/tst/small/small.ctys
/homen/vadmin/qemu/tst/sparc-test/sparc-1.ctys
/homen/vadmin/qemu/tst/sparc-test/sparc-2.ctys
/homen/vadmin/qemu/tst/linux/linux.ctys
/homen/vadmin/qemu/tst/coldfire-test-0.1/coldfire-test-0.1.ctys
```

Figure 36.9: ctys-vhost awk-regexpr-1

Finally the correct call would be:

`ctys-vhost -o ids QEMU vadmin`

The name of the hypervisor, which is the name of the plugin within ctys, is stored in the database literally, thus as uppercase "QEMU". Anyhow, the usage of this string within a pathname could still lead to unexpected results.

Chapter 37

System Resources

37.1 TAP/TUN by VDE

A TAP device in combination with virtual switches is applied for Stacked-Networking. The actual implementation is based on the package *"Virtual Distributed Ethernet"* - *VDE(vde2)*[132, sourceforgeVde].

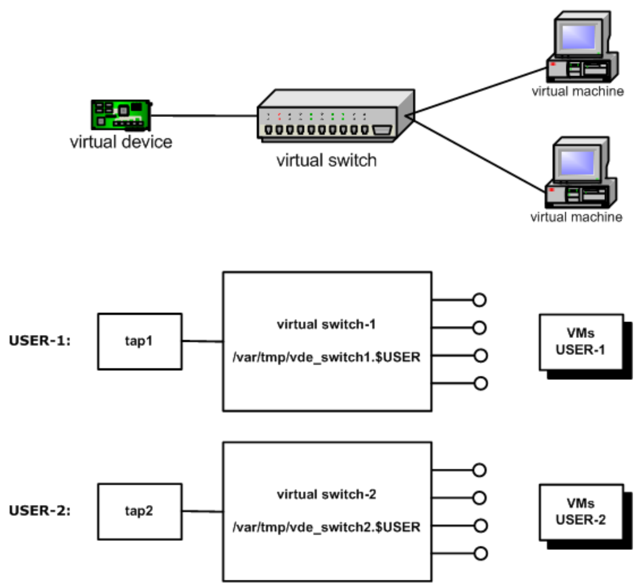


Figure 37.1: Virtual switch by vde_switch

The tool called "vde_tunctl" from *"Virtual Distributed Ethernet"* is used for in combination with the virtual switch "vde_switch". This supports user-space access and on demand creation of new interfaces with required non-privileged user permissions only. Just one initial TAP device is required to be created with root permissions. The depicted basic structure is assumed to be pre-configured for each user and interconnected to the main virtual-bridge.

The most of the required stecps are covered within ctys-vnetctl .

Chapter 38

Pre-Configured Desktops

38.1 Desktop-Assembly

As already depicted in the introduction of the features Desktops are the main forcus and most important building block of a User Interface and therefore an essential element for handling of huge and dynamic user environments.

The design of the ctys elements targets the automation of the complete assembly process of GUI-Elements, herein more or less complete application frontends only. These could be seen as an functional aspect of an overall task to be combined into a toolset supporting a complete workflow.

The process of usage of tools frewuently comprises two types of tools, the inital base set, which is usually more or less used during the whole process. A secondary set, which is required case-by-case or beginning from latter process step.

The desktop entity class supported by ctys is the VNC plugin. This is applied to the VM and PM plugins as consoles as well as within any additional task for the native GuestOSs.

Technically these two classes are splitted into two approaches and facilities.

(a) Semi-Dynamic Pre-Configuration
Based on the **xstartup** configuration file of VNC, which is actually a shell script executed during startup of the vncserver.

(b) Dynamic Post-Configuration
Based on the **-D** option, thus could be any frontend - except CLI - which could be redirected for integrated usage within a specific desktop.

38.2 Semi-Dynamic Pre-Configuration

This comprises a custom configuration of the installed file with a 'case' element in shell syntax, and additionally the **CREATE** suboption **VNCDESKIDLIST**. The latter defines a list of user-defined 'case' elements, which are executed during startup by the 'vncserver' with a shell-subcall. The installed xstartup contains a list with various operational examples.

The call

```
ctys -a create=l:tst2,VDIL:demo1%demo3%demo5 -g 800x600+200+200:A21  tst@app1
```

opens a window like the following.

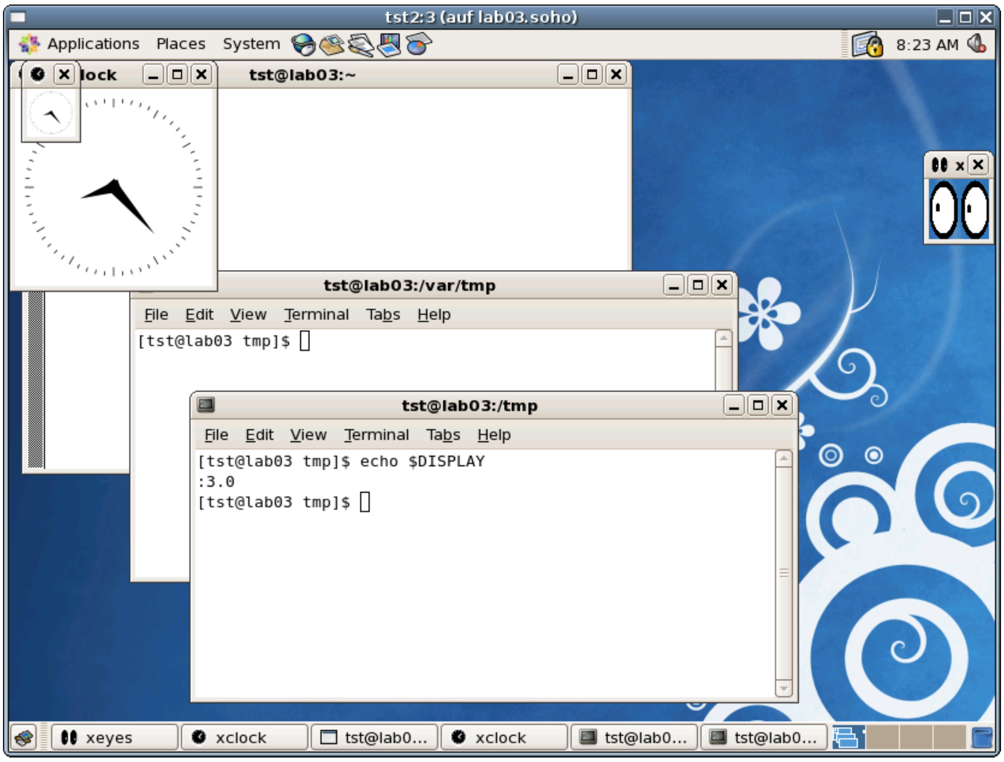


Figure 38.1: Pre-Configured Desktop by xstartup

the following extract of the example definition within the installed xstartup file controls the actual creation of the xclock, xterm and xeyes windows. Additionally two ctys-X11-plugin based xterminal and gnome-terminal windows are created, whereas in each a different initial working directory is opened.

```
for vdID in ${VNCDESKIDLIST};do
    case ${vdID} in
        demo1)
            xclock&
            xterm -geometry 80x24+10+10 -ls -title "$vdID" &
            ;;
        demo2)
            xclock -geometry 40x40+900+10&
            ;;
        demo3)
            xeyes -geometry 50x50+900+100&
            ;;
        demo4)
            xlogo -geometry 50x50+1000+200&
            ;;
        demo5)
            xclock -geometry 40x40+10+10&
            ${HOME}/bin/ctys -t x11 -a create=l:${vdID}-0,cd:/var/tmp \
                 -g 60x12+100+200
            ${HOME}/bin/ctys -t x11 -a create=l:${vdID}-1,cd:/tmp,console:gterm
```

```
                     -g 60x12+150+300
                ;;
            *)
                ;;
        esac
    done
```

This style of configuration provides facilities for almost any pre-definition of a set of static GUI-modules which could be combined by the applied **VNCDESKIDLIST** suboption.

6. Dynamic Post-Configuration
The dynamic configuration could either be used to fill an initially empty desktop, or to add on demand-addons to an preconfigured basic workspace.

The following call adds an emacs terminal with '/etc' as current working directory to the previous example.

```
ctys -t x11 -a create=l:tst1x,cd:/etc,console:emacs -g 40x6+20+250 \
    tst@lab03'(-D tst2 -T all -d pf)'
```

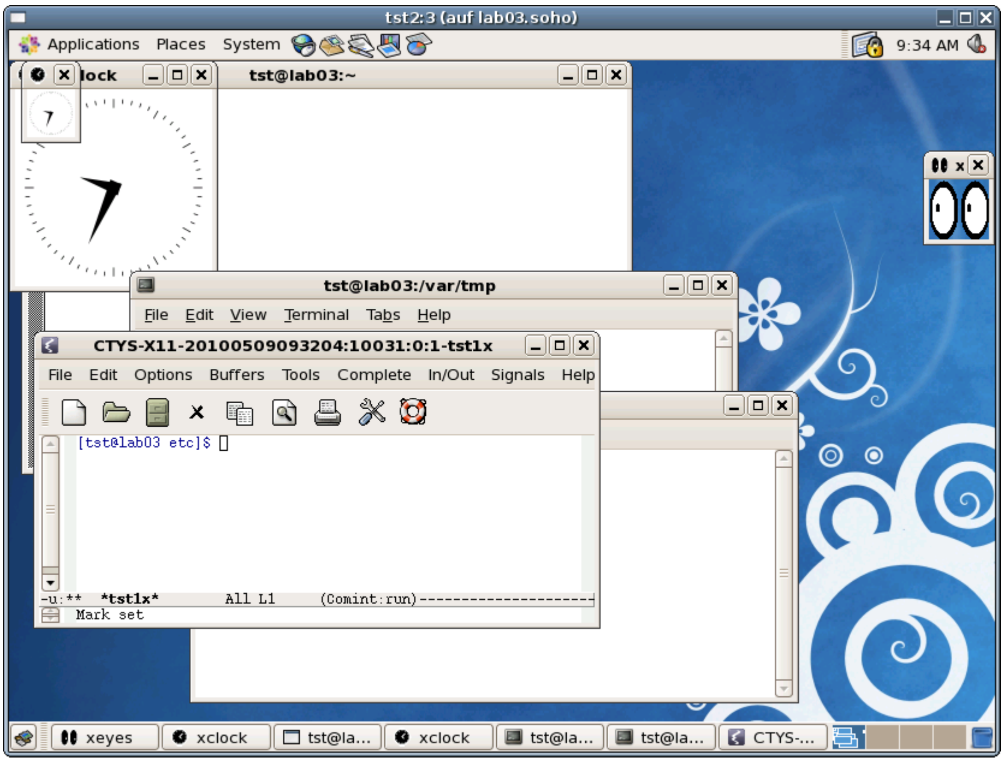


Figure 38.2: Pre-Configured Desktop with additional Emacs terminal session

The following call adds a VNC session to the previous example.

```
ctys -t vnc -a create=l:tst1x,reuse -g 300x200+20+250 \
    tst@lab03'(-D 3 -T all -d pf,1 )'
```

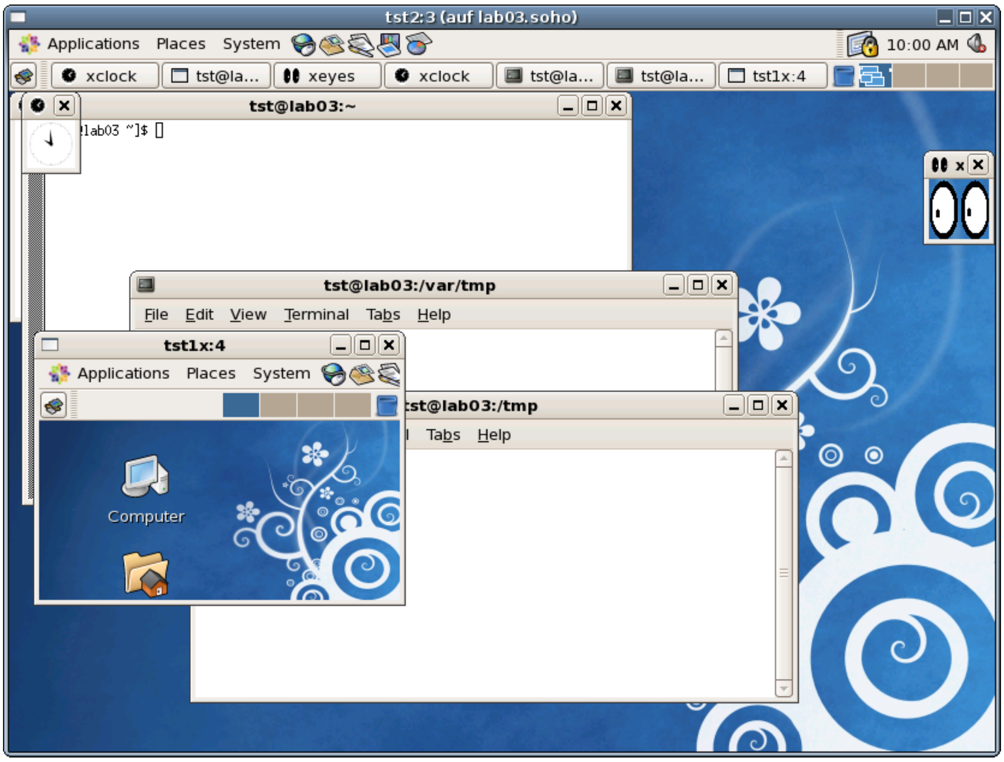


Figure 38.3: Pre-Configured Desktop with additional VNC session

38.3 Admin RAID-Info

The following example shows a preconfiguration of an example Desktop, where the health of the basic resources is inspected in detail. Therefore the main RAID systems are inspected and monitored by their proprietary tools, and the UPS supplying to the systems is displayed by the OpenSource "netups". Each remote host is displayed by VNC on a seperate screen, but could be moved and resized throughout the whole array as required.

The access to all of this tools is due to general philosophy permitted to local access only, thus requires an SSH session provided here by UnifiedSessionsManager. Direct remote access is either disabled by configuring UNIX-Domain sockets or by blocking the communications.

This example contains sessions of the plugin VNC only, thus representing HOSTS sessions to physical machines runnning their native OS(and/or Dom0 for XEN). No startup of a VM by an involved hypervisor is required.

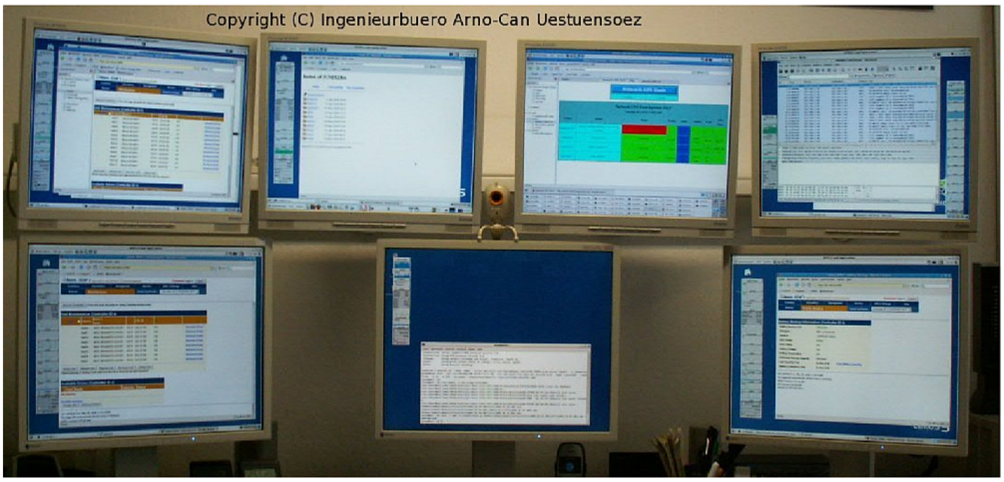


Figure 38.4: Administration of RAID and Power-Supply

The display array is configured as Xinerama mode for array-like addressing as described in Section ?? '??' on page ?? . Which has the following namebinding for the physical array-parts of the logical screen.

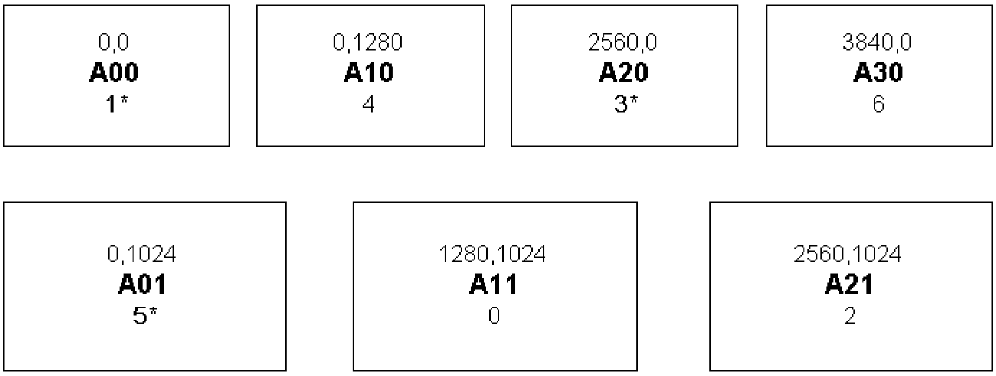


Figure 38.5: Logical Multi-Screen X11-Array-Style

Thus the screens with the logical address A00, A10, and A02 display the 3ware raid tool 3DM2 and "gkrellm". The screen A20 displays the management station with "netups" for the UPS array. The screen A30 displays the out-ot-band protocol analyser based on ethereal/wire-shark and gkrellm. The screen A10 displays the available UNIX-Distros in the html-view. The screen A11 is reserved as console for interaction.

The reason of describing such a scenario is the each-time effort required, when starting this desktop manually. Things become even worster, when the "moderate" count of sessions displayed on the "taskmanager" at the bottom of screen A20 is considered. Thus a manual startup of a multi-desktop/workspace environment could easily take 10minutes and more - each time. Therefore ths UnifiedSessionsManager was designed and implemented, in order to ease the definition of masses of scenarios with involved bulks of sessions. To shorten it up, the actual call to startup the above destop is:

ctys admin

That's it.

The call consists of the mandatory base call "ctys" of course, and an additional pre-configure group called "admin". The goup contains the following entries where each entry itself is a complete context specific configuration of a call.

```
#
#This groups contains all machines in the management group.
#
root@host1'( -t vnc -a create=reuse,l:HOST1 \
             -g 1268x872:A20 \
             -W admin -b 1,2)'

root@host2'( -t vnc -a create=reuse,l:HOST2 \
             -g 1268x994:A30 \
             -W admin -b 1,2)'

root@host3'( -t vnc -a create=reuse,l:HOST3 \
             -g 1268x994:A00 \
             -W admin -b 1,2)'

root@host4'( -t vnc -a create=reuse,l:HOST4 \
             -g 1268x958:A10 \
             -W admin -b 1,2)'

root@host5'( -t vnc -a create=reuse,l:HOST5 \
             -g 1268x994:A01 \
             -W admin -b 1,2)'

root@host6'( -t vnc -a create=reuse,l:HOST6 \
             -g 1268x994:A21 \
             -W admin -b 1,2)'
```

The most remarkable option is here the "-g" option, which defines the display size and position. The specifics for VNC to consider here is the missing "-r" option, which defines the actual server-resolution, whereas the g-option just defines the size of the client window. The "-r" option is actually supported for VNC only dynamically, some limited support for QEMU consoles is available too. Within ctys the "-r" option is adapted from the "-g" option when not provided explicitly. As a consequence the resulting "-r" option is the maximum size which could be displayed without restarting the vncserver, shrinking is supported by pure clipping with additional display of scrollbars.

The other point to mention is the logical addressing of the screens only, which makes it independent from physical reassignments of display ports, e.g. due to a motherboard replacement.

In addition to the only required call of "ctys admin", any entity could be called easily as a single session by just an "smart-cut-and-paste". This is due to the containment of the complete set of options within the context, where each of them could be used as a self-contained set to replace the group name.

The start of "HOST3" could be written as:

```
ctys root@host3'( \
  -t vnc -a create=reuse,l:HOST3 \
  -g 1268x994:A00 \
  -W admin -b 1,2)'
```

38.4 QEMU Test-Environment for VDE

This example depicts the test environment for the first basic testst of the tool "ctys-vnetctl" . In this scenario PMs only are used to perform the following tests manually.

Screen	Machine	Hypervisor	Testcase
A00	host01	VMPlayer+QEMU	Single NIC without present bridge.
A10	host02	VMServer+QEMU	Bonding device only, without present bridge.
A20	-		
A30	host04	XEN+QEMU	Single NIC with a default bridge present.
A01	host05	XEN+QEMU	Bonding+Additional WoL-NIC, with present Xen bridge on bond0.
A11	-		
A21	host07	VM-WS6+QEMU	Bonding+Additional WoL-NIC, without present bridge.

Table 38.1: QEMU Base Tests for "ctys-vnetctl"

Due to the various possible and very common cases of present and recognisable virtual bridges, the history of LAN configuration met by "ctys-vnetctl" could span a number of different configurations. For each such case at least the "create" and "cancel" argument has to be tested and validated by checks via "brctl", ifconfig, and "route". Also multiple bridges for different users and multiple calls for the same user has to be performed.

The tests listed as example in the previous table just span the usage of PMs, that is native access to a machine. Additional tests are performed for stacks within GuestOSs on the various stack levels and OSs. The tests above are performed on CentOS/RHEL-5.0 and additional systems.

However, the intention of this chapter is not to disclose test utilities and startegies, but to show one typical application of the UnifiedSessionsManager.
To open the following desktop with all of the previous listed scenarios the call

ctys tst-qemu

is sufficient.

Figure 38.6: First "Test-Field" of PMs for basic tests of "ctys-vnetctl"

That's it, with a little additional one-time pre-configuration.

```
root@host1'( \
 -t vnc \
 -a create=reuse,l:QEMU1 -g 1268x994:A01 \
 -b 1,2)'

root@host2'( \
 -t vnc \
 -a create=reuse,l:QEMU2 \
 -g 1268x994:A21 -b 1,2)'

root@host3'( \
 -t vnc \
 -a create=reuse,l:QEMU5 \
 -g 1268x958:A10 -b 1,2)'

root@host4'( \
 -t vnc \
 -a create=reuse,l:QEMU3 \
 -g 1268x994:A30 -b 1,2)'

root@host5'( \
 -t vnc \
 -a create=reuse,l:QEMU4 \
 -g 1268x994:A00 -b 1,2)'
```

Part VI

VM Management

Chapter 39

Cloning of VMs

39.1 Address Syntax

This document describes the automated change and configuration mainly provided by **ctys-createConfVM**. **ctys-cloneVM** is used for the creation of cloned VMs.

39.2 Preparations

The first step is the creation of a source VM. Which could be automated by **ctys-createConfVM**. This tool is used for **ctys-cloneVM** too in order to set new unique values for a cloned VM. Specific attributes such as the activation/deactivation for the inventory scanning by '#*#MAGICID-IGNORE' could be set by **ctys-attribute** .

Once this step is completed, any VM could be used as a template for the creation of additinal VMs. Therefore the virtual disk and the machine configuration data are seperated, thus the majority of the number of the machine parameters could be altered by either using configuration utilities or by an ordinary ASC-II editor.

39.3 Receipts

39.3.1 Supported features

The configuration could be proceeded in two basic variants:

(a) QEMU/KVM, Xen
The complete duplication and configuration could be proceeded by the utility **ctys-createConfVM**. The GuestOS has to be setup by common procedures in a second step.

(b) VBOX, VMW
The VM has to be setup by the native OEM tools and could be registered for ctys by post creation of configuration files and collecting these into the registry. The GuestOS has to be setup by common procedures in a second step.

39.3.2 Template Creation

The required template for the automated cloning could be any present VM. Thus an initial VM has to be present for the apllication of **ctys-cloneVM** . The creation of the initial VM is described in detail within the hypervisor specific subsections.

(a) **Create a Template-VM**
The following call could be used either for the complete creation of a template VM with the option –**create-image** or for creation of the inventory data only by usage of the option –**create-image**.
The following call creates a dummy VM with a small dummy-image in case of Xen.

```
MAC=00:11:22:33:44:55 \
IP=12.12.12.12 \
TCP=gecko \
HDDBOOTIMAGE_INST_SIZE=128M \
HDDBOOTIMAGE_INST_BLOCKSIZE=32M \
HDDBOOTIMAGE_INST_BLOCKCOUNT=4 \
ctys-createConfVM \
  -t XEN \
  --label=tst02 \
  --auto-all \
  --create-image
```

(b) **Adapt a Template-VM**
The provided features for the modification of existing VMs vary by actual hypervisors and contained OS. Some attributes may require simpl changes only, e.g. the number of reserved CPUs, whereas others like the change of the size of the storage may require the support of the hypervisor as well as the support of the embedded guest OS and the applied filesystems.

(c) **Create defaults file from the Source-VM**
The following call of *ctys-createConfVM(1)* creates the defaults file, which is used for pre-assignment of values from the source machine for the cloned machine. Some basic machine related IDs are required to be altered by command line assignment.

```
MAC=00:11:22:33:44:55 \
ctys-createConfVM \
  -t XEN \
  --label=tst02 \
  --defaults-file-create \
  --auto-all \
  --no-create-image \
  --no-save-para-kernel
```

39.3.3 Common Clone Call

Therefore the following call could be executed.

```
ctys-cloneVM \
  -t xen \
  --label=tst02clone \
  --label-old=tst02 \
  --target-directory=/mntn/vmpool/vmpool05/xen/test/tst-ctys-call \
  --ip=14.14.14.14 \
  --mac=44:44:44:44:44:44 \
  --tcp=abc \
```

```
--uuid=123412341234 \
--vm-state=DISABLED \
-f
```

This creates a complete clone, which just requires additional native configuration within the GuestOS.

39.4 QEMU/KVM

1. **Create a Test-VM**
   The following call creates a dummy VM with a small dummy-image.
   ```
   MAC=00:11:22:33:44:55 \
   IP=12.12.12.12 \
   TCP=gecko \
   HDDBOOTIMAGE_INST_SIZE=128M \
   HDDBOOTIMAGE_INST_BLOCKSIZE=32M \
   HDDBOOTIMAGE_INST_BLOCKCOUNT=4 \
   ctys-createConfVM \
     -t QEMU \
     --label=tst02 \
     --auto-all \
     --create-image
   ```

2. **Create defaults file from the Source-VM**
   The following call of
   ctys-createConfVM(1) creates the defaults file, which is used for pre-assignment of values from the source machine for the cloned machine. Some basic machine related IDs are required to be altered by command line assignment.
   ```
   MAC=00:11:22:33:44:55 \
   ctys-createConfVM \
     -t QEMU \
     --label=tst02 \
     --defaults-file-create \
     --auto-all \
     --no-create-image \
     --no-save-para-kernel
   ```

3. **Clone the Source-VM**
   The following call actually cloned the source VM.
   ```
   ctys-cloneVM \
     -t QEMU \
     --label=tst02clone \
     --label-old=tst02 \
     --target-directory=/mntn/vmpool/vmpool05/kvm/test/tst-ctys-call \
     --ip=14.14.14.14 \
     --mac=44:44:44:44:44:44 \
     --tcp=abc \
     --uuid=123412341234 \
     --vm-state=DISABLED \
   ```

```
  -f
```

39.5 VBOX

1. **Create a Test-VM**

   The initial creation is for now supported by the native OEM tools only. The following call starts a remote user interface of the graphical VirtualBox frontend.

   ```
   ctys \
     -t cli \
     -a label=l:VBOX,cmd:VirtualBox \
     myUser@myHost
   ```

   The following call of ctys-createConfVM creates extended configuration information for the ctys inventory.

   ```
   MAC=00:11:22:33:44:55 \
   IP=12.12.12.12 \
   TCP=gecko \
   HDDBOOTIMAGE_INST_SIZE=128M \
   HDDBOOTIMAGE_INST_BLOCKSIZE=32M \
   HDDBOOTIMAGE_INST_BLOCKCOUNT=4 \
   ctys-createConfVM \
     -t VBOX \
     --label=tst02 \
     --auto-all \
     --no-create-image
   ```

2. **Create defaults file from the Source-VM**
   The following call of
   ctys-createConfVM(1) creates the defaults file, which is used for pre-assignment of values from the source machine for the cloned machine. Some basic machine related IDs are required to be altered by command line assignment.

   ```
   MAC=00:11:22:33:44:55 \
   ctys-createConfVM \
     -t VBOX \
     --label=tst02 \
     --defaults-file-create \
     --auto-all \
     --no-create-image \
     --no-save-para-kernel
   ```

3. **Clone the Source-VM**
   The following call actually cloned the source VM.

   ```
   ctys-cloneVM \
     -t VBOX \
     --label=tst02clone \
     --label-old=tst02 \
     --target-directory=/mntn/vmpool/vmpool05/vbox/test/tst-ctys-call \
     --ip=14.14.14.14 \
   ```

```
--mac=44:44:44:44:44:44 \
--tcp=abc \
--uuid=123412341234 \
--vm-state=DISABLED \
-f
```

39.6 VMW

ffs.

39.7 XEN

ffs.

39.8 Display and modify attributes

The following examples demonstrate how to modify specific attributes within the various supported hypervisor configuration files.

1. List XEN configuration file:
   ```
   ctys-attribute --list tst02clone.conf
   ```
2. List CTYS configuration file:
   ```
   ctys-attribute --list tst02clone.ctys
   ```
3. Replace all values of an attribute in XEN configuration file:
   ```
   ctys-attribute \
     --attribute-name='kernelbase' \
     --attribute-replace=ALL  \
     --attribute-value-new="'/nboot1'"  \
     tst02clone.conf
   ```
4. Replace a string in all values of an attribute in XEN configuration file:
   ```
   ctys-attribute \
     --attribute-name='kernelbase' \
     --attribute-replace=ALL \
     --attribute-value-new="'/nboot1'" \
     --attribute-value-old="'/boot'" \
     tst02clone.conf
   ```
5. Replace string in **FIRST** value of an attribute in XEN configuration file:
   ```
   ctys-attribute \
     --attribute-name='kernelbase' \
     --attribute-replace=FIRST \
     --attribute-value-new="'/nboot1'" \
     --attribute-value-old="'/boot'" \
     tst02clone.conf
   ```
6. Replace string in **LAST** value of an attribute in XEN configuration file:
   ```
   ctys-attribute \
     --attribute-name='kernelbase' \
     --attribute-replace=FIRST \
     --attribute-value-new="'/nboot1'" \
     --attribute-value-old="'/boot'" \
     tst02clone.conf
   ```

7. Replace string in **2.** value of an attribute in CTYS configuration file:
```
ctys-attribute \
  --attribute-name='kernelbase' \
  --attribute-replace=FIRST \
  --attribute-value-new="'/nboot1'" \
  --attribute-value-old="'/boot'" \
  tst02clone.conf
```
8. Replace all values of an attribute in CTYS configuration file:
```
ctys-attribute \
  --attribute-name='WRAPPERCALL' \
  --attribute-replace=ALL  \
  --attribute-value-new="'123'"  \
  tst02clone.ctys
```
9. Create a new attribute key without a value assignment in CTYS configuration file:
```
ctys-attribute \
  --attribute-create=TOP \
  --attribute-name='#*#MAGICID-IGNORE' \
  --attribute-keyonly \
  tst02clone.ctys
```
10. Delete an attribute key without a value assignment in CTYS configuration file:
```
ctys-attribute \
  --attribute-delete=FIRST \
  --attribute-name='#*#MAGICID-IGNORE' \
  --attribute-keyonly \
  tst02clone.ctys
```
11. Create new attribute value assignment in CTYS configuration file:
```
ctys-attribute \
  --attribute-create=TOP \
  --attribute-name='NEWATTR' \
  --attribute-value-new="'123'"  \
  tst02clone.ctys
```
12. Create new attribute value assignment in CTYS configuration file:
```
ctys-attribute \
  --attribute-create=BOTTOM \
  --attribute-name='NEWATTR' \
  --attribute-value-new="'123'"  \
  tst02clone.ctys
```
13. Create new attribute value assignment in CTYS configuration file:
```
ctys-attribute \
  --attribute-create=15 \
  --attribute-name='NEWATTR' \
  --attribute-value-new="'123'"  \
  tst02clone.ctys
```
14. Delete first occurance of an attribute in CTYS configuration file:
```
ctys-attribute \
  --attribute-delete \
  --attribute-name='NEWATTR' \
  tst02clone.ctys
```
or

```
ctys-attribute \
  --attribute-delete=1 \
  --attribute-name='NEWATTR' \
  tst02clone.ctys
```

15. Delete ALL occurances of an attribute in CTYS configuration file:

```
ctys-attribute \
  --attribute-delete=ALL \
  --attribute-name='NEWATTR' \
  tst02clone.ctys
```

.

Chapter 40

Configure installation automation of GuestOSs

40.1 Installation Automation

This document describes the provisioning of automated installation procedures by **ctys-createConfVM** , either **semi-automatic** or **fully automatic** .
The almost only required configuration is the access path to the installation media for the GuestOS. The media could be accessed by various protocols, either local or remote, in general more or less any URI is supported.
The resulting degree of automation depends on the provided means of the GuestOS. When a method like **kickstart** ist provided this could be configured as a parameter and provides a full turn-key automation. In case of means of offline installation like **debootstrap** a full degree of installation automation is provided too.
The predefined parameters may be used in cases of configuration-only calls too, where specific parameters for defined versions coul be predefined. This also comprises the abort of the installation for unsupported configuration requests.
The only drawback of a missing default set is the mandatory requirement of interactive proceeding.

40.2 Preparations

The first step is the configuration of the installation soources for **ctys-createConfVM**. Theese are are looked up first in the user's home directory

`${HOME}/.ctys/ctys-createConfVM.d/defaults-${C_SESSIONTYPE}.ctys`

second within the installation currently executed path

`${MYCONFPATH}/ctys-createConfVM.d/defaults-${C_SESSIONTYPE}.ctys`

Additionally new files have to be registered in the file

`${MYCONFPATH}/ctys-createConfVM.d/hook.sh`

The priority is to use the first matched value.
The installation by usage of a boot media does not require additional configurations, whereas the application of PXE may require some additional setup. This comprises the network services including the appropriate setup of the local BIOS, and probably the use of some initial boot-up media.

Once this step is completed, the option ' **–auto** ' and **–auto-all** ' could be applied for semi or full-automation.

40.3 Receipts

First Call Example

The configuration could be proceeded by editing the appropriate configuration file as defined in the preparations section.

The sources could be either provides by local file access including NFS mounts, or by use of protocols such as FTP or HTTP as provided by the installer of the guest system. The setup of PXE requires various additional configurations.

The hypervisors **VirtualBox** and **VMware** are supported by interactive configuration and manual installation of guest systems only.

The following call creates a dummy VM with a small dummy-image by application of −**auto-all** , which supports the complete automation of configuration and guest systems.

```
MAC=00:11:22:33:44:55 \
IP=12.12.12.12 \
TCP=gecko \
HDDBOOTIMAGE_INST_SIZE=128M \
HDDBOOTIMAGE_INST_BLOCKSIZE=32M \
HDDBOOTIMAGE_INST_BLOCKCOUNT=4 \
ctys-createConfVM \
  -t XEN \
  --label=tst02 \
  --auto-all \
  --create-image
```

Attributes of the created VMs could be varied by application of the utility **ctys-attribute**.
Predefined Configuration Patterns
The following keys are available for selection. These partly depend on additional attributes, e.g. the ARCH or ACCEL keys.

Key	Description
ACCELERATOR	Available and actually supported accelerators for the HostOS.
ARCH	Architecture of the GuestOS distribution.
DIST	The GuestOS distribution to be installed.
DISTREL	The release of the distribution sources.
GuestOS	The OS contained in the source, here descriptive text only.
Instmode	The supported modes for interactive installation and automation by
	DEFAULTINSTMODE.
TYPE	The session type - actually the hypervisor - as runtime environment.

The current present configuration sources are listend within the following table. The configured paths has to be adapted within the configuration files. Missing configuration options are supported by interactive configuration only.

The distribution names and OS names are owned by their suppliers and may/are protected by trademarks.

Android

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
1.6	i386	QEMU	QEMU,KVM	CD	Android 1.6
2.2	i386	QEMU	QEMU,KVM	CD	Android 2.2

CentOS

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
5, 5.0	X86_64	QEMU	QEMU,KVM	CD,PXE	Linux-*
5, 5.0	X86_64	XEN	PARA,HVM	CD,PXE	Linux-*
5.5	X86_64	QEMU	QEMU,KVM	CD,PXE	Linux-*
5.5	X86_64	XEN	PARA,HVM	CD,PXE	Linux-*
5.*	X86_64	QEMU	QEMU,KVM	CD,PXE	Linux-*
5.*	X86_64	XEN	PARA,HVM	CD,PXE	Linux-*

debian

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
4.0_r3, 4.*	i386,amd64	QEMU	QEMU,KVM	CD,PXE	Linux-*
5.0.0	amd64	XEN	PARA,HVM	CD,PXE	Linux-*
5.0*	i386,amd64	XEN	PARA,HVM	CD,PXE	Linux-*

eeDroid

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
2.2	i386	QEMU	QEMU,KVM	CD	Linux-*

EnterpriseLinux

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
5, 5.0	X86_64	QEMU	QEMU,KVM	CD,PXE	Linux-*
5, 5.0	X86_64	XEN	PARA,HVM	CD,PXE	Linux-*
5.0	X86_64	QEMU	QEMU,KVM	CD,PXE	Linux-*
5.5	X86_64	QEMU	QEMU,KVM	CD,PXE	Linux-*
5.5	X86_64	XEN	PARA,HVM	CD,PXE	Linux-*
5.*	X86_64	QEMU	QEMU,KVM	CD,PXE	Linux-*
5.*	X86_64	XEN	PARA,HVM	CD,PXE	Linux-*

Fedora

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
8	X86_64	XEN	PARA,HVM	CD,PXE	Linux-*
10	X86_64	QEMU	QEMU,KVM	CD,PXE	Linux-*
10	X86_64	XEN	PARA,HVM	CD,PXE	Linux-*
12	X86_64	XEN	PARA,HVM	CD,PXE	Linux-*
13	X86_64	QEMU	QEMU,KVM	CD,PXE	Linux-*

FreeBSD

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
7.*	X86_64	XEN	PARA,HVM	CD,PXE	FreeBSD-7.*
7.*	X86_64	QEMU	QEMU,KVM	CD,PXE	FreeBSD-7.*
8.*	X86_64	XEN	PARA,HVM	CD,PXE	FreeBSD-8.*
8.*	X86_64	QEMU	QEMU,KVM	CD,PXE	FreeBSD-8.*

Gentoo

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
2008.0*	X86_64	XEN	PARA,HVM	CD,PXE	Linux-*
2008.0*	X86_64	QEMU	QEMU,KVM	CD,PXE	Linux-*

Knoppix

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
6.2.1, 6.2*	X86_64	XEN	PARA,HVM	CD	Linux-*
6.2.1, 6.2*	X86_64	QEMU	QEMU,KVM	CD	Linux-*

Mandriva

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
2009*	i386, X86_64	XEN	PARA,HVM	CD	Linux-*
2009*	i386, X86_64	QEMU	QEMU,KVM	CD	Linux-*
2010*	i386, X86_64	XEN	PARA,HVM	CD	Linux-*

MeeGo

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
1.0*	i386	QEMU	QEMU,KVM	CD	Linux-*

MS-Windows - MSProducts

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
WNT4WS	i386	XEN	HVM	CD	MS-Windows-NT-4.0-WS
WNT4S	i386	XEN	HVM	CD	MS-Windows-NT-4.0-S
W2KWS	i386	XEN	HVM	CD	MS-Windows-2000-WS
W2KS	i386	XEN	HVM	CD	MS-Windows-2000-Server
WXP	i386	XEN	HVM	CD	MS-Windows-XP
W7	x86_64	XEN	HVM	CD	MS-Windows-7
W2003	x86_64	XEN	HVM	CD	MS-Windows-2003
W2008	x86_64	XEN	HVM	CD	MS-Windows-2008
WNT4WS	i386	QEMU	KVM	CD	MS-Windows-NT-4.0-WS
WNT4S	i386	QEMU	KVM	CD	MS-Windows-NT-4.0-S
W2KWS	i386	QEMU	KVM	CD	MS-Windows-2000-WS
W2KS	i386	QEMU	KVM	CD	MS-Windows-2000-Server
WXP	i386	QEMU	KVM	CD	MS-Windows-XP
W7	x86_64	QEMU	KVM	CD	MS-Windows-7
W2003	x86_64	QEMU	KVM	CD	MS-Windows-2003
W2008	x86_64	QEMU	KVM	CD	MS-Windows-2008

NetBSD

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
5.0.2	X86_64	QEMU	QEMU,KVM	CD,PXE	NetBSD-5.0.2
5.*	X86_64	QEMU	QEMU,KVM	CD,PXE	NetBSD-5.*

OpenBSD

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
4.8	X86_64	QEMU	QEMU,KVM	CD,PXE	OpenBSD-4.8
4.8	X86_64	XEN	PARA,HVM	CD,PXE	OpenBSD-4.8

OpenSolaris

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
2009.6	i386(x86_64)	QEMU	QEMU,KVM	CD,PXE	OpenSolaris-2009.6
2009.6	i386(x86_64)	XEN	PARA,HVM	CD,PXE	OpenSolaris-2009.6

OpenSUSE

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
11.*	X86_64	QEMU	QEMU,KVM	CD	Linux-*
11.*	X86_64	XEN	PARA,HVM	CD	Linux-*

RHEL

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
6, 6.0	X86_64	QEMU	QEMU,KVM	CD	Linux-*
6, 6.0	X86_64	XEN	PARA,HVM	CD	Linux-*
5, 5.0	X86_64	QEMU	QEMU,KVM	CD	Linux-*
5, 5.0	X86_64	XEN	PARA,HVM	CD	Linux-*
5.5	X86_64	QEMU	QEMU,KVM	CD	Linux-*
5.5	X86_64	XEN	PARA,HVM	CD	Linux-*
5.*	X86_64	QEMU	QEMU,KVM	CD	Linux-*
5.*	X86_64	XEN	PARA,HVM	CD	Linux-*

Scientific

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
5.*	X86_64	QEMU	QEMU,KVM	CD	Linux-*
5.*	X86_64	XEN	PARA,HVM	CD	Linux-*

Solaris

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
10*	i386(x86_64)	QEMU	QEMU,KVM	CD,PXE	Solaris-10*
10*	i386(x86_64)	XEN	PARA,HVM	CD,PXE	Solaris-10*

Ubuntu

DISTREL	ARCH	TYPE	ACCELERATOR	Inst	GuestOS
8.04	amd64	QEMU	QEMU,KVM	CD,PXE	Linux-*
8.04	amd64	XEN	PARA,HVM	CD,PXE	Linux-*
9.10	amd64	QEMU	QEMU,KVM	CD,PXE	Linux-*
9.10	amd64	XEN	HVM	CD,PXE	Linux-*
10.10	amd64	QEMU	QEMU,KVM	CD,PXE	Linux-*
10.10	amd64	XEN	HVM	CD,PXE	Linux-*

Part VII

Special Use-Cases

Chapter 41

Development Environments

.

41.1 Setup of a PHP development environment

41.1.1 General

This document describes the setup of a virtualized development environment as a personal workspace. The intention for this described setup is the provisioning and versioning of a developemnt environment particularly for the development of PHP based web applications including a web servers and database servers. The setup described in this document consists of the following system components

- Database-server
- Web-Server
- Clients
- Remote Displays

All components could be executed on one physical machine but are described in the given example as a distributed setup located on different PMs. The following environment is used here as a virtual development environment, which is used for modultest too.

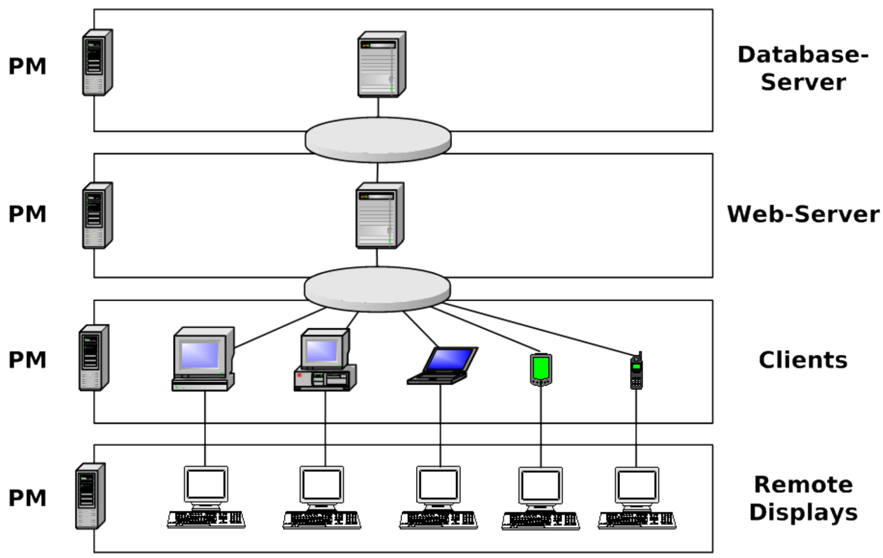


Figure 41.1: A Distributed PHP development environment

The ordinary development environment is extended by the administrative applications, which are hosted witin VMs too. This comprises the management applications for the MySQL server. The basic utility requires at least MS-Windows-XP as runtime environment, whereas the Eclipse based MySQL-Explorer could be executed on Linux as well. The second application is the Network Management and Monitoring entity based on Nagios. This is hosted on a virtual Linux machine running CentOS.

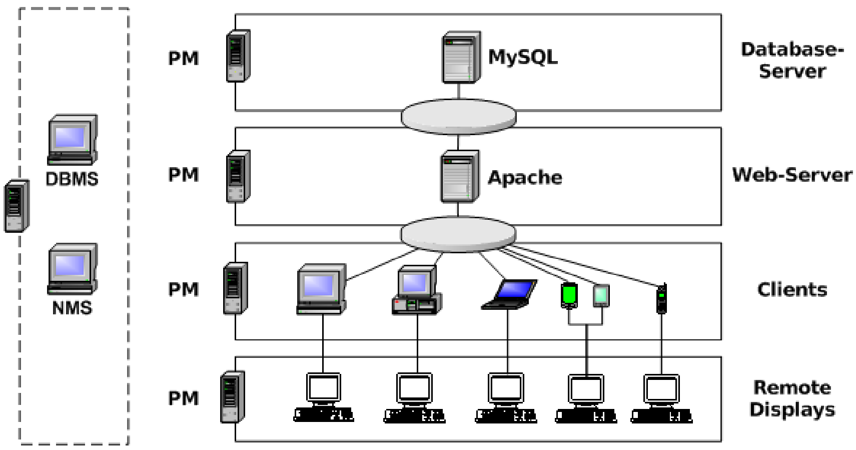


Figure 41.2: A Distributed PHP development environment

The client virtualization particularly provides a cheap and simplified facility for the development and test of various clients with miscellaneous operating systems and applications. This particularly simplifies the versioning and storage of pre-configured complete system sets and provides for development, test and systems integration environments.

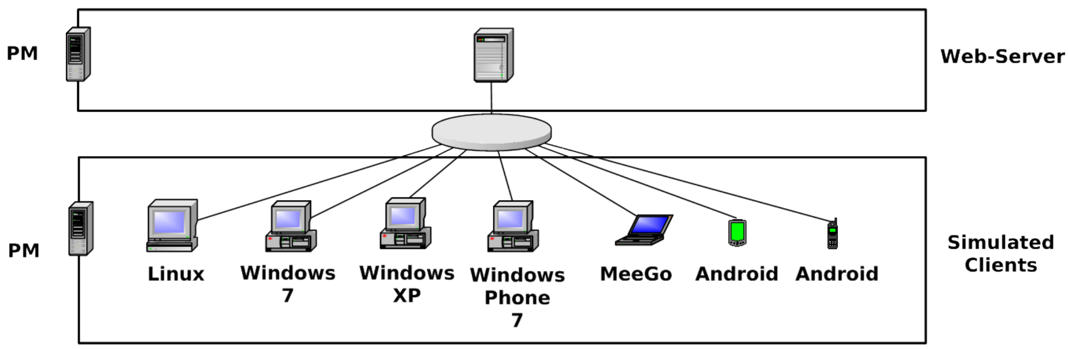


Figure 41.3: Simulated clients

.

41.1.2 Setup of Host-OS, Hypervisor, and GuestOS

The installation for the following variants has to be performed by the appropriate standard setup of the HostOS and hypervisor.

CentOS with QEMU/KVM

Here the standard distribution is installed. Additional packages are vde2-2.2.3 and Qemu-0.12.2, which are build and installed to '/opt'. The vde2-2.2.3 package for network encapsulation requires a symbolic link

```
ln -s /opt/vde2-2.2.3 /opt/vde
```

The wrapper vde may not be required, when the Qemu support option is compiled in, but this is not yet widely the case. Thus vde2 is still utilized as standard. For additional information refer to ctys-configuration-QEMU(7) and ctys-uc-CentOS(7) .

Debian with VirtualBox

The OSE edition is missing RDP access, thus supports synchronous consoles for display only. When RDP is required, which is recommended, install the download version instead of the OSE edition shipped with the distribution. The alternative is to install the OSE edition and additionally the extension components which are licnesed slightly different. For additional information refer to ctys-configuration-VBOX(7) and ctys-uc-debian(7) .

CentOS with VMware-Server and Workstation

This straight forward. For additional information refer to ctys-configuration-VMW(7) and ctys-uc-CentOS(7) .

41.1.3 Setup of the UnifiedSessionsManager

Install tgz BASE-Package + DOC-Package on Debian

1. Unpack the tar-gzip-archive and apply the standard installation procedure, where the call has to be executed by typing the fully qualified absolute path when ambiguity could occur. This is due to automatic usage of consistent libraries for the install procedure.

   ```
   ctys-distribute -F 2 -P UserHomeCopy root@lab02
   ```

2. Open a Remote Shell by call of CLI plugin:

   ```
   ctys -t cli -a create=l:tst137 root@tlab02
   ```

3. Check the plugins states by calling ctys-plugins:

   ```
   ctys-plugins -T all -E
   ```

For additional information refer to ctys-multisite-install(7) and ctys-distribute(1) .

Install rpm BASE-Package + DOC-Package on CentOS

The following steps are required for a RPM based setup on CentOS. The installation is relocatable, but located at '/opt', and installed locally by 'ctys-distribute'.

1. Install BASE package.

   ```
   rpm -i ctys-base-01.11.011.noarch.rpm
   ```

2. Now install a a local version, here by copy. The PATH prefix is important here, particularly in case of updates. The path is resolved to it's actual path by eliminating any symbolic link, and used for consistent link of libraries.

```
/opt/ctys-01.11.011/bin/ctys-distribute -F 2 -P UserHomeCopy
```

3. Next the menu is setup.

   ```
   ctys-xdg --menu-create
   ```

4. Now the help is available as eihter a Gnome or KDE menu. Alternatively could be called from the commandline.

For additional information refer to ctys-multisite-install(7) and ctys-distribute(1) .

Setup of the Gnome Menu

The first supported user interface is Gnome, others are going to follow. The current version provides a desktop based GUI only, thus the provided command line interfaces are integrated into the desktop and could be started by means of the windows manager(screenshots). The next step is the setup of a custom menue, which could be extended by the templates provided by the tool ctys-xdg(1) . The first supported user interface is Gnome, others are going to follow. The current version provides a desktop based GUI only, thus the provided command line interfaces are intergated into the desktop and could be started by means of the window manager. Additional information is available by ctys-configuration-Gnome(7)
The setup of the Gnome Menu is quite simple, the call

```
ctys-xdg --menu-create
```

sets up a basic standard menu:

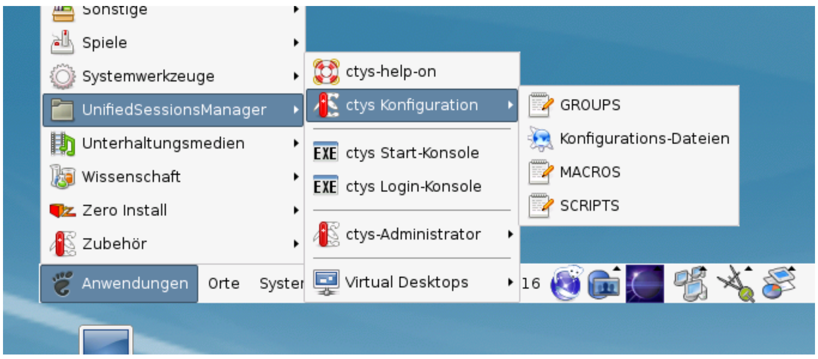


Figure 41.4: Create Menu

The setup could be targeted either for private menus or shared menus. Both setups are based on a menu template, which is stored in the configuration subdirectory 'xdg.d'. The call

```
ctys-xdg --menu-cancel
```

removes the installed files. For current version no checks for changed files is done. The menues could be edited and extended by the call

```
ctys-xdg --menu-edit
```

which opens the related directories for modification of '*.menu', '*.desktop', and '*.directory' files.

This could be easily extende to a project specific menu set containing single machines as well as complete desktops and documentation sets.

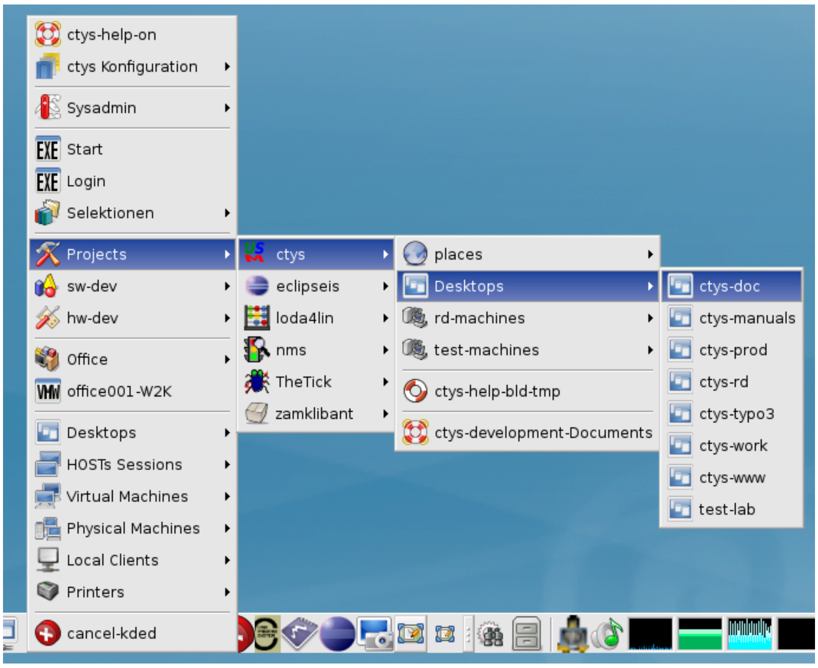


Figure 41.5: Custom project menu

The alternative could be a tool related environment.

Figure 41.6: Custom tool related menu

41.1.4 Creation of the Inventory - cacheDB

The optional creation of a inventory database enables the use of user defined shortcuts, so called labels. This database could be created fully automatic with a little preconfiguration such as providing the list of the remote servers and the remote directory tree to be scanned. Additional information is available from **ctys-vdbgenVM(1)** and e.g. **ctys-uc-CentOS(7)**

Graphical Start of Virtual Machines

Once the menu is created and the VMs are either scanned into the database or added manually, these could be started by the systems menu entries. This is independent from the actual used OS. The basics patterns and configuration options are depicted in the document **ctys-configuration-Gnome(7)**

Creation of a virtual Appliance for Eclipse

Eclipse is one of the most accepted development and applications environment, but lacks in some cases from version dependencies of it's components. Thus the creation of a virtual appliance with a consistent set of components porvides a simple means for the distribution of comprising development environments.

Creation of a Development Desktop

ffs.

Creation of a Project Menu

ffs.

41.1.5 Manage the VM

ffs.

Chapter 42

Test Environments

Chapter 43

High-Volume and High-Performance Environments

Chapter 44

Standard production systems

Chapter 45

Special Purpose Use-Cases

Part VIII

Miscellanous Applications

Chapter 46

Performance Measures

46.1 Background, Parallel and Cached Operations

The following call deactivates the caching of data and operates in synchronous and sequential mode. This causes each target to be listed sequentially, which take for the actual test configuration about 72 seconds. This is caused particularly by the outdated machine lab01.

```
time ctys -a list=tab_tcp,id -C off -b seq,sync \
          lab00 lab01 app1 app2
```

The following call works similar, but activates the caching of, thus a result is displayed after processing all targets.

```
time ctys -a list=tab_tcp,id -C on -b seq,sync \
          lab00 lab01 app1 app2
```

When changing the "-b" option for altering the synchronism, the task duration is almost just the period required by the slowest machine, with minor overhead.

The following call now activates the parallel operation but still occupies the console caused by "sync" suboption.

```
time ctys -a list=tab_tcp,id -C off -b par,async \
          lab00 delphi olymp app1 app2
```

The overal required processing time is now aboout 30seconds, which is the individual time required by lab01. Without the lab01 it takes 17seconds as expected, which is 11seconds each with about 1-2seconds when bundling overhead.

The performance gain is in a linear manner, thus handling of 10 or more machines definetly benefits from these concept, which makes quickly a difference of 11-18 seconds vs. several minutes.

Even though the overall performance could be still enhanced, in comparison to manual handling of even 4 machines only with about 5-10 VMs of multiple types on each, this required processing time of 11-18seconds can not be blamed.

46.2 Caching of Data

The second main field of performance measures is the caching of static data for network nodes and VMs with their GuestOSs and hypervisors and caching some local runtime data of for repetitive access.

The caching of local runtime data is particularly helpful, when ID transformations for mapping and remapping of access keys has to be performed. This data has one-call life-period and is foreseen just for real short time caching in the range of 1second within a call.

The second area of caching is the local caching of network and VM data, which could be classified as static. This contains static configured DHCP data including the DNS entry, and the ENUMERATED VM configurations, including the information of their installed GuestOSs.

The whole set of local data is stored and managed by the tool ctys-vhost. Which will be used internally for cached querries for several kinds of name resolution. When the local cache database is not present ctys silently switches to polling mode, which is actually a "find-grep-scan" of the remote filesystem for appropriate VM-configuration files. The performance impact of uncached operations depends on the call parameters and the dimension of the scanned directory tree, but could be immense.

When the data is appropriately cached the average access time of ctys-vhost to an VM record is within one second, for some hundred stored entities about 0.5-0.8 seconds. Which depends on several influencing parameters, of course. Anyhow, the performance gain is about a factor of 100 or more.

The local database has to be prepared by the tool ctys-vdbgen and ctys-extractMAClst and/or ctys-extractARPlst. Additionally a configuration file for the PM should be genrated by "ctys-genmconf".

46.3 Combined Operations

The generation of the caching database itself and the access to the cached data are the best examples for the comparison of cached and non-cached access to distributed work environments.

The draft tests are specific to the environment and some differences in the individual nodes, but they are representative.

In a test-group of 7 machines with a common NFS mounted pool, and a sum of 365 VM entries, which is about an average of 50 one-layer-VMs on each node, the following performance was measured.

Nr.	Access Method	"-b" option	duration
1	Collection of data with ENUMERATE	sync,sequential	4:16
2	Collection of data with ENUMERATE	sync,parallel	1:14
3	Query ctys-vhost, first access with rebuild cache	n.a.	1:53seconds
4	Query ctys-vhost for all accespoints of "linux001"	n.a.	0.6seconds

Table 46.1: Performance effects of "-b" option

In a production-group of 9 machines with a common NFS mounted pool, multiple users with deep home-directory-trees, and a sum of 1768 one-layer-VM entries, which is about an average 200 VMs on each node, the following performance was measured.

Nr.	Access Method	"-b" option	duration
1	Collection of data with ENUMERATE	sync,sequential	15:26
2	Collection of data with ENUMERATE	sync,parallel	3:56
3	Query ctys-vhost, first access with rebuild cache	n.a.	2:06
4	Query ctys-vhost for all accespoints of "linux001"	n.a.	0.8seconds

Table 46.2: Performance effects of "-b" option

Chapter 47

VM Stacks

47.1 Setup of the Stack Environment

In current version the stacked operation of more than 2 nested layers is primarily for test reasons. This is going to be changed for daily business step-by-step, whereas stack-aware appliances are developed.

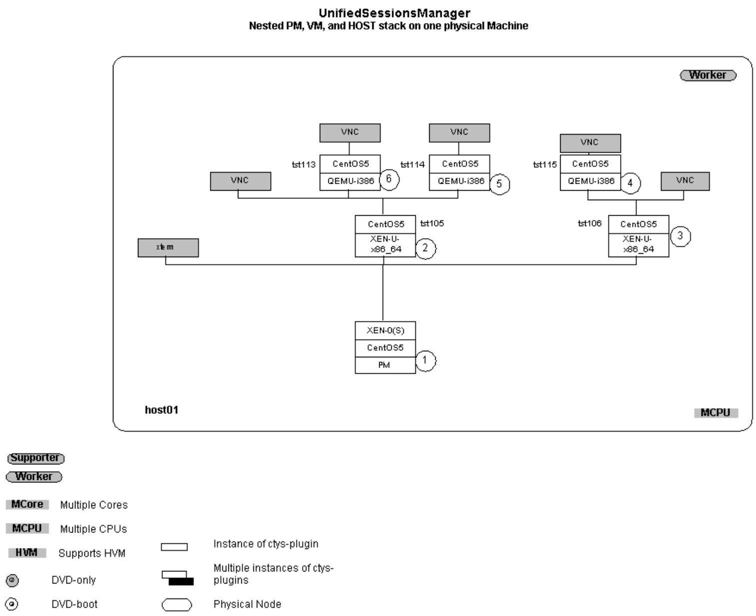


Figure 47.1: Example for a simple Stack

The standard case for now is up to 2 layers for the combination of QEMU with Xen and/or VMware hypervisors. In some cases a combined network simulation group is setup into one VM, thus could be frozen and debugged as one unit when required.

The following architecture is a common operational environment for a small network, which already is designed Service-Centric by specific Stack-Components.

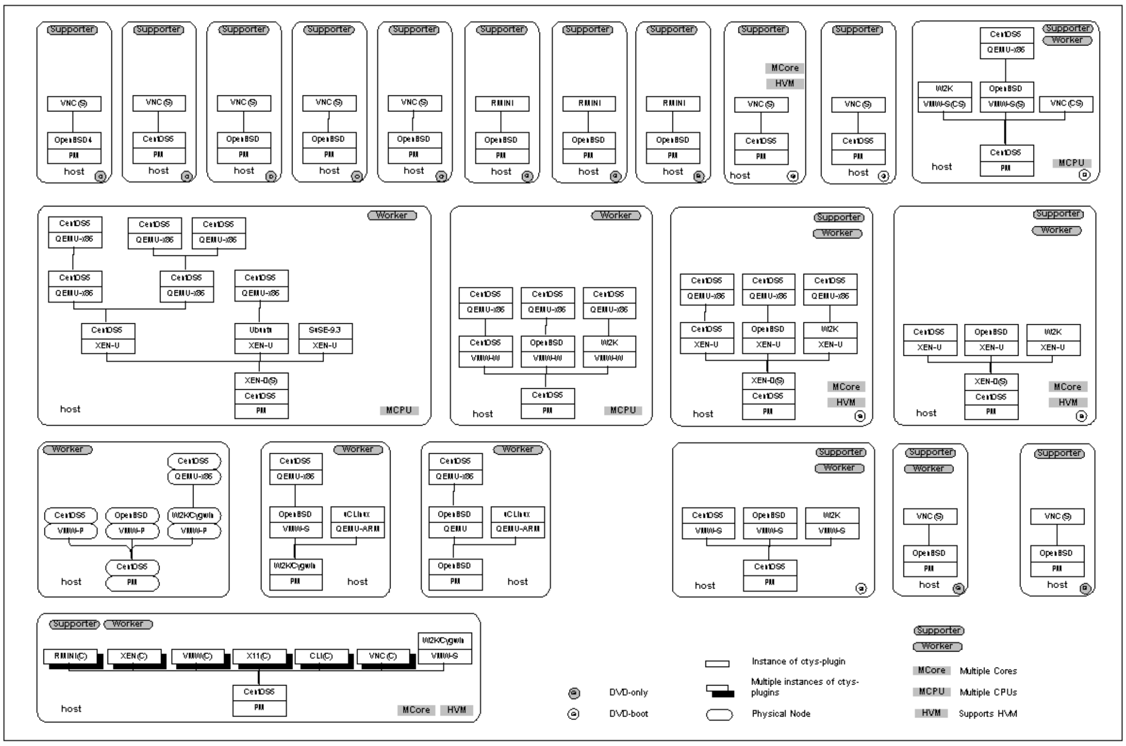


Figure 47.2: Example for a Small Stacked-Network

For now the author is looking forward for the arrival of the first announced 6-core and 12-core CPUs by AMD and for sure by Intel too, meanwhile preparing it's software appropriately by usage of 2-core and 4-core CPUs.

The main building block for nesting of stacks available for now is QEMU as a pure generic emulator. Therefore as already mentioned the KQEMU module is avoided. For usage of VM bld/build/output/doc-tmp/en/images in general, but nested stack elements - here QEMU - in particular, the setup of a common network structure for seamless usage within each participating entitiy is essential. This for example enables the usage of the same VM within multiple stack levels and within various VMs. Thus a common filesystem hierarchy is defined based on NFS for VM distribution and nesting.

For performance critical cases an additional local structure is set up for enhancing acces to stored data.

The second issue for distribution and nesting of VMs is a SSO - Single-Sign-On - authentication and authorization. This is required for the actual payload users within the GuestOSs as well as for the sessions management of the hypervisors by UnifiedSessionsManager itself. The choice for the examples and the actually installed environment is a combination of Kerberos, LDAP, Samba, NFS, and Automount/Autofs. This is particularly combined with "ksu" and "sudo" for the internall call of system utilities by ctys.

For the given examples the following structure is defined as basis, which is used within all VMs and PMs. Therefore each custom script and filepath could be used within each VM, which could be instanciated independent of it's execution environment.

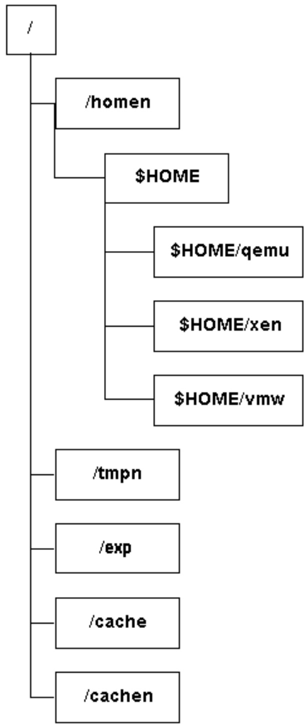


Figure 47.3: Filesystem Structure

The usage of NFS as the global storage requires an appropriate network performance of course. Thus bonding of GiB lines is widely utilized, what offers a quite well throughput with limited costs. This is recommended for central servers and heavily loaded application servers homing more than 10-20 VMs, what depends on the actual load of each. The physical structure of distribution within a network is as depicted in the following figure.

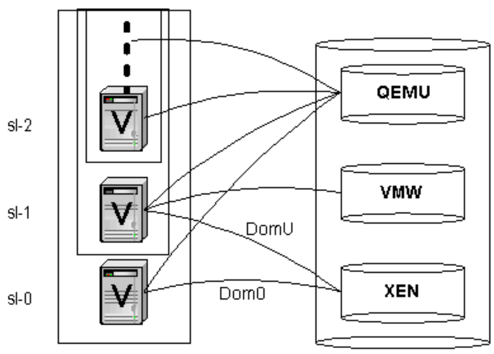


Figure 47.4: Stacked VMs and Filesystem Access

The network contains the generic VMs which could be relocated to various execution sites and levels. The only available vertically arbitrarily stackable VM is QEMU, whereas the VMW and XEN hypervisors require a "dominant" ownership of all present CPUs and could be used for current versions EXOR only on a single machine.

47.2 Multi-Layer VMStacks

47.2.1 Prepare a PM for Restart by WoL

This call prepares the physical machine for wakeup by WoL packets.

#

```
#
#First version of stack-aware tests, with canonical syntax.
#
#

########################################################
#
#generic preparation of a machine for WoL restart
#
tst-prep    = \

  #
  #local options
  #
  -t pm \
  -a cancel=poweroff:0,force,self,wol \
  -b 1,3 \

  -- \

  #
  #remote-only options
  #
  '(-Z ksu)'
```

47.2.2 Xen with upper QEMU-VMs

This example shows the creation of one "column" of a VM-Stack, where 3 Stack-Layers are created.

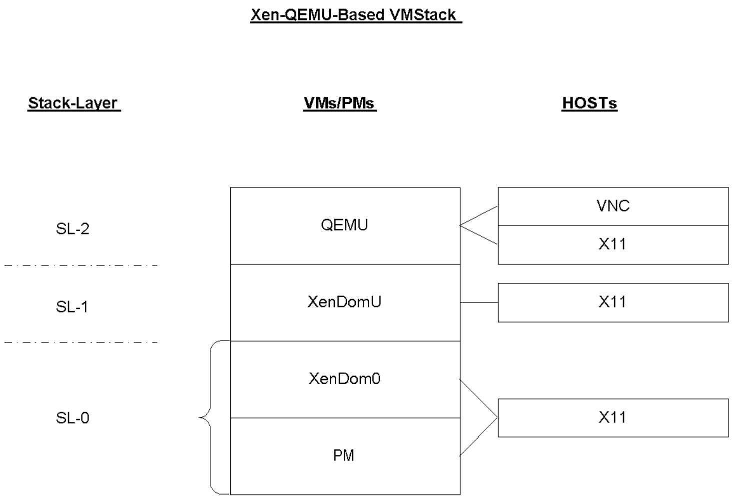


Figure 47.5: Example for a Single Stack-Column

The following virtual network components are involved in the interconnection of the various VM-Stack-Layers to the physical NIC and though to the actual physical network.

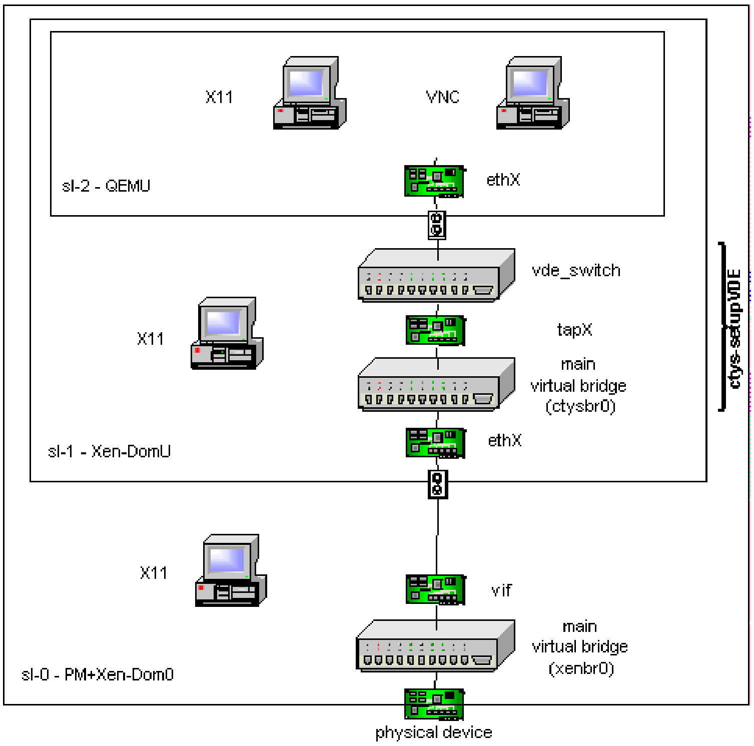


Figure 47.6: Example for a Single Stack-Column of Network Components

The following example is the literal content of a specific MACRO for test purposes. This particularly includes the remote call of "ctys-vnetctl" for remote and automated creation of the virtual bridge and swithc within the "SL-1".

```
###########
#
#  XEN+QEMU: app1
#
#  Description:  First successful stack-start
#
tst-01a = VMSTACK'{\


 #######################################################
 #1.) SL-0: Relay for WoL on base PM
 #######################################################
 ws1( \
   -t pm  \
   -Z ksu \
   -a create=l:app1,t:app1-eth2,m:00:E0:81:2B:10:33,wol,reuse \
   -c off \
 ) \
```

```
######################################################
#2.) SL-0: Creation of native acces by xterm.
######################################################
app1( \
  -t x11 \
  -Z ksu \
  -a create=l:STACK-APP1,console:xterm \
  -g 80x13+0+800:A00 \
) \



######################################################
#3.) SL-1: First level VM, a Xen-DomU here
######################################################
app1( \
  -t xen \
  -Z ksu \
  -a create=l:tst104,reuse,console:none \
  -c local \
) \



######################################################
#4.) SL-1: Creation of native acces by xterm.
######################################################
tst104( \
  -t x11 \
  -Z ksu \
  -a create=l:TST104,console:xterm \
  -g 80x13+0+600:A00 \
) \



######################################################
#5.) SL-1: Creation of required TAP device and a
#          bridge when missing, a switch for
#          non-privileged USER-SPACE usage by user.
######################################################
root@tst104( \
  -t cli \
  -Z ksu \
  -a create=l:tstVDE,cmd:ctys-vnetctl%-f%-u%acue%create \
) \



######################################################
#6.) SL-2: Creation of second-level VM, QEMU here.
######################################################
tst104( \
  -t qemu \
  -Z ksu \
  -a create=l:tst127,reuse,console:none,user:wol1 \
  -c local \
) \
```

```
######################################################
#7.) SL-2: Creation of native acces by xterm.
######################################################
wol1@tst127( \
  -t x11 \
  -Z ksu \
  -a create=l:TST127,console:xterm \
  -g 80x13+0+400:A00 \
) \
```

```
######################################################
#8.) SL-2: Creation of native acces by VNC.
######################################################
wol1@tst127( \
  -t vnc \
  -Z ksu \
  -a create=l:mydsk,reuse \
  -g 400x300+0+0:A00 \
) \
}'
```

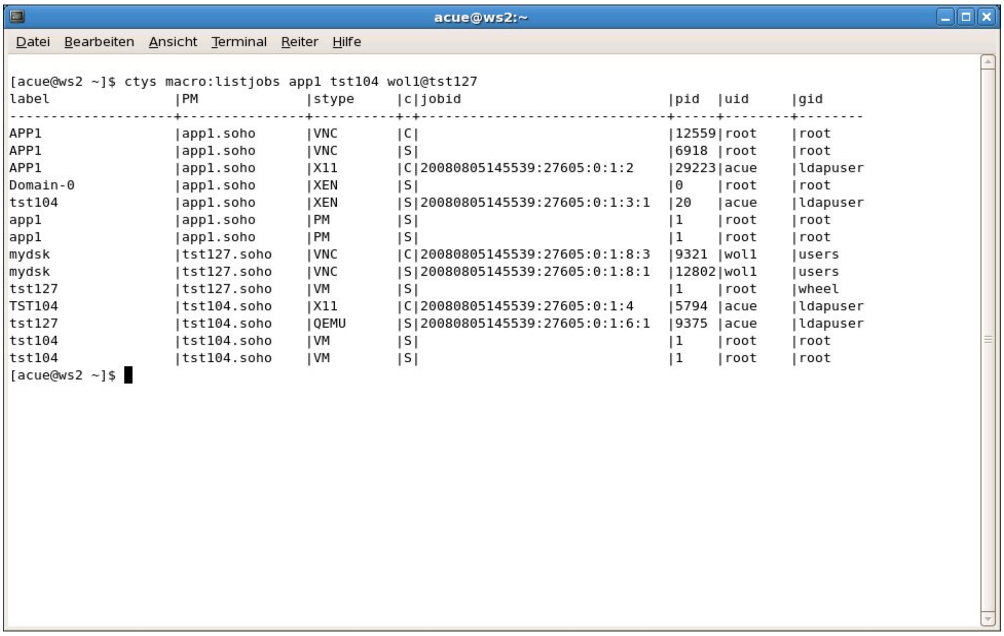
The resulting display should be as follows, some hostnames has to be adapted of course.

Figure 47.7: VMSTACK tst-01a in action.

The list call

```
list '{macro:listjobs}' app1 tst104 wol1@tst127
```

should result in an output similar to the following.



```
[acue@ws2 ~]$ ctys macro:listjobs app1 tst104 woll@tst127
label               |PM              |stype     |c|jobid                        |pid  |uid     |gid
--------------------+----------------+----------+-+-----------------------------+-----+--------+--------
APP1                |app1.soho       |VNC       |C|                             |12559|root    |root
APP1                |app1.soho       |VNC       |S|                             |6918 |root    |root
APP1                |app1.soho       |X11       |C|20080805145539:27605:0:1:2   |29223|acue    |ldapuser
Domain-0            |app1.soho       |XEN       |S|                             |0    |root    |root
tst104              |app1.soho       |XEN       |S|20080805145539:27605:0:1:3:1 |20   |acue    |ldapuser
app1                |app1.soho       |PM        |S|                             |1    |root    |root
app1                |app1.soho       |PM        |S|                             |1    |root    |root
mydsk               |tst127.soho     |VNC       |C|20080805145539:27605:0:1:8:3 |9321 |woll    |users
mydsk               |tst127.soho     |VNC       |S|20080805145539:27605:0:1:8:1 |12802|woll    |users
tst127              |tst127.soho     |VM        |S|                             |1    |root    |wheel
TST104              |tst104.soho     |X11       |C|20080805145539:27605:0:1:4   |5794 |acue    |ldapuser
tst127              |tst104.soho     |QEMU      |S|20080805145539:27605:0:1:6:1 |9375 |acue    |ldapuser
tst104              |tst104.soho     |VM        |S|                             |1    |root    |root
tst104              |tst104.soho     |VM        |S|                             |1    |root    |root
[acue@ws2 ~]$
```

Figure 47.8: LIST of running VM instances

As displayed, the job data of the user root is visible only to root. Currently this feature is under development, thus not yet consequently applied to all plugins.

Chapter 48

User Access - Secure permissions

48.1 SSH, SSL, IPsec, and VPNs

The only supported remote access method of ctys-tools is the usage of SSH, any other method might be used in companion.

Even though SSH keys could be utilized and some other methods like r-files could be used, these are not discussed here.

The only and one SSO approach used and supported by the author is a kerberos and LDAP based SSO in companion with kerberised SSH.

With the usage of automount NFS(will be changed soon to AFS), a seamless authorisation and environment support throughout stacked VMs is supplied.

For root access a local account should remain.

48.2 Authentication - ksu and sudo

48.2.1 Basics

Due to the timeout behaviour of ksu and sudo during probing when no user is configured, the default behaviour is not to probe for theese tools.

By default the user permissions are used only, thus any systems maintenance operations could be performed by usage of the root user, or any other with root permissions.

This could be utilized for example by usage of k5login with kerberos.

When ksu or sudo has to be used, which means internally probed for access permissions, this has to be preconfigured by the environment variables "USE_SUDO" and/or "USE_K5USERS". Alternatively this could be set call-by-call and different for local and remote components with the option "-Z". When using "sudo" it has to be decided whether pty is required or not, if yes, than the "-z" options has to be set.

48.2.2 sudoers

Sudoers is some more flexible adaptable than k5users, it particularly allows access-profiles by distinguishing several call-options, e.g. CANCEL, CREATE, etc.. The approach of lean call options in difference to supplying more action keys on first level-options just makes it little more challenging, but not impossible.

This is particularly helpful when restricting access selectively to specific call options. E.g. in case of Xen to the required Dom0 for LIST action call.

When using sudoers the following entries are required for this version.

The users as required:

User_Alias CTYS_USER = chkusr, wol1

At least one user with complete local acces is required, when a bridge has to be shutdown for WoL setting, and this is the only LAN connection to the caller. Thus during the shutdown the NFS mounts are on those machines out of service for a short time where additional calls to scripts are reuqired. So the shutdown could be performed from a local copy only.

If not tty should be pre-required for sudo. If the default remains, the "-z" option is required.

Defaults requiretty

For machines without required bridges the following permissions are required:

```
Cmnd_Alias CTYS_CMD = \
  /usr/bin/which \
  ,/sbin/halt ,/sbin/ethtool ,/sbin/reboot ,/sbin/init
  ,/sbin/poweroff
```

For machines to be used to send WoL packets to local segment addtionally the following permissions are required:

```
Cmnd_Alias CTYS_CMD = \
   /sbin/ether-wake
```

When a bridge is configured on the target, which has to be shutdown for setting "wol g", which is required for Xen-3.0.2, following root permissions are reuqired

```
Cmnd_Alias CTYS_CMD = \
   ,/etc/xen/scripts/network-bridge ,/sbin/ip \
   ,/usr/sbin/brctl ,/sbin/ifup ,/sbin/ifdown \
   ,/usr/bin/head
```

When running Xen on local machine the following has to be added:

```
Cmnd_Alias CTYS_CMD = \
   ,/usr/sbin/xm ,/usr/bin/virsh \
```

For the users assignement an entry like this has to be set:

```
CTYS_USER ALL = (root) NOPASSWD: CTYS_CMD
```

48.2.3 k5users

For a present kerberos environment the utilization of krb5users seems to be the smartest approach, even though the call itself cannot be matched by specific CLI-options. This lack for selective access profiles could be worked around by defining different users and appropriate chroot-environments easily.

For machines without required bridges the following permissions are required:

```
<user>@<realm> \
  /usr/bin/which \
  /sbin/halt  /sbin/ethtool /sbin/reboot /sbin/init \
  /sbin/poweroff
```

For machines to be used to send WoL packets to local segment addtionally the following permissions are required:

```
   /sbin/ether-wake
```

When a bridge is configured on the target, which has to be shutdown for setting "wol g", which is required for Xen-3.0.2, following root permissions are reuqired

```
   /etc/xen/scripts/network-bridge /sbin/ip \
   /usr/sbin/brctl /sbin/ifup /sbin/ifdown \
   /usr/bin/head
```

When running Xen on local machine the following has to be added:

```
   /usr/sbin/xm /usr/bin/virsh
```

When setting up a router for remote WoL some configuration for sending Ethernet-Broadcasts or a directed IP-broadcast is required.

Part IX

Appendices

Chapter 49

Current Loaded Plugins

This section enumerates the current loaded static libraries and the dynamic loaded plugins. Which will be partly detected automaticaly and loaded as predefined or On-Demand.
The following list is generated with the call:

"ctys -T all -v"

REMARK: For limited environents this could produce errors due to memory exhaustion. The error messages ar not obvious!!!

```
--------------------------------------------------------------------------

UnifiedSessionsManager Copyright (C) 2007, 2008, 2010 Arno-Can Uestuensoez

This program comes with ABSOLUTELY NO WARRANTY; for details
refer to provided documentation.
This is free software, and you are welcome to redistribute it
under certain conditions. For details refer to "GNU General Public
License - version 3" <http://www.gnu.org/licenses/>.

--------------------------------------------------------------------------
PROJECT          = Unified Sessions Manager
--------------------------------------------------------------------------
CALLFULLNAME     = Commutate To Your Session
CALLSHORTCUT     = ctys

AUTHOR           = Arno-Can Uestuensoez - unifiedsessionsmanager@protonmail.com
MAINTAINER       = Arno-Can Uestuensoez - unifiedsessionsmanager@protonmail.com
VERSION          = 01_10_011
DATE             = 2010.03.18

LOC              = 117540 CodeLines
LOC-BARE         = 59063 BareCodeLines (no comments and empty lines)
LOD              = 0 DocLines, include LaTeX-sources

TARGET_OS        = Linux:   CentOS/RHEL, Fedora, ScientificLinux,
                            debian, Ubuntu,
                            (gentoo,) mandriva,
                            (knoppix,) (dsl,)
                            SuSE/openSUSE
```

```
                   BSD:     OpenBSD, FreeBSD
                   Solaris: Solaris-10, OpenSolaris
                   Windows: (WNT/Cygwin), (W2K/Cygwin), (WXP/Cygwin),
                            (W2Kx/Cygwin)

TARGET_VM        = KVM, (OpenVZ), QEMU, (VirtualBox,) VMware, Xen
TARGET_WM        = fvwm, Gnome, (KDE,) X11

GUEST_OS         = ANY(some with limited native-acces support)
--------------------------------------------------------------------------
COPYRIGHT        = Arno-Can Uestuensoez - unifiedsessionsmanager@protonmail.com
LICENCE          = GPL3
--------------------------------------------------------------------------
EXECUTING HOST   = ws2.soho
--------------------------------------------------------------------------
LIBRARIES(static-loaded - generic):

  Nr   Library                                      Version
  -------------------------------------------------------------
  00   bootstrap.01.01.004.sh                       01.10.010
  01   base.sh                                      01.07.001b01
  02   libManager.sh                                01.02.002c01
  03   cli.sh                                       01.07.001b06
  04   misc.sh                                      01.06.001a12
  05   security.sh                                  01.06.001a05
  06   help.sh                                      01.10.002
  07   geometry.sh                                  01.07.001b06
  08   wmctrlEncapsulation.sh                       01.07.001b06
  09   groups.sh                                    01.11.001
  10   network.sh                                   01.11.001


PLUGINS(dynamic-loaded - ctys specific):

  Nr   Plugin                                       Version
  -------------------------------------------------------------

  00   CORE/CACHE.sh                                01.07.001b01
  01   CORE/CLI.sh                                  01.07.001b06
  02   CORE/COMMON.sh                               01.02.002c01
  03   CORE/CONFIG/hook.sh                          01.06.001a14
  04   CORE/DIGGER/hook.sh                          01.07.001b06
  05   CORE/DIGGER/list.sh                          01.02.001b01
  06   CORE/ENV.sh                                  01.02.002c01
  07   CORE/EXEC.sh                                 01.06.001a15
  08   CORE/GENERIC.sh                              01.10.011
  09   CORE/HELP.sh                                 01.02.002c01
  10   CORE/LABELS.sh                               01.07.001b05
  11   CORE/STACKER/hook.sh                         01.07.001b06
  12   CORE/VMs.sh                                  01.02.002c01
```

```
  13   GENERIC/hook.sh                            01.02.001b01
  14   GENERIC/LIST/list.sh                       01.10.008
  15   GENERIC/ENUMERATE/enumerate.sh             01.02.001b01

  16   HOSTs/CLI/hook.sh                          01.06.001a09
  17   HOSTs/CLI/session.sh                       01.01.001a01
  18   HOSTs/CLI/list.sh                          01.10.008
  19   HOSTs/CLI/info.sh                          01.02.001b01
  20   HOSTs/VNC/hook.sh                          01.02.001b01
  21   HOSTs/VNC/session.sh                       01.06.001a15
  22   HOSTs/VNC/list.sh                          01.07.001b05
  23   HOSTs/VNC/info.sh                          01.02.001b01
  24   HOSTs/X11/hook.sh                          01.06.001a09
  25   HOSTs/X11/session.sh                       01.01.001a01
  26   HOSTs/X11/list.sh                          01.01.001a00
  27   HOSTs/X11/info.sh                          01.02.001b01

  28   VMs/QEMU/hook.sh                           01.10.008
  29   VMs/QEMU/config.sh                         01.01.001a01pre
  30   VMs/QEMU/session.sh                        01.10.008
  31   VMs/QEMU/enumerate.sh                      01.10.008
  32   VMs/QEMU/list.sh                           01.10.008
  33   VMs/QEMU/info.sh                           01.01.001a00pre
  34   VMs/VMW/hook.sh                            01.10.009
  35   VMs/VMW/session.sh                         01.10.009
  36   VMs/VMW/enumerate.sh                       01.06.001a09
  37   VMs/VMW/list.sh                            01.10.009
  38   VMs/VMW/info.sh                            01.02.001b01
  39   VMs/XEN/hook.sh                            01.10.008
  40   VMs/XEN/config.sh                          01.01.001a01
  41   VMs/XEN/session.sh                         01.07.001b06
  42   VMs/XEN/enumerate.sh                       01.01.001a01
  43   VMs/XEN/list.sh                            01.10.008
  44   VMs/XEN/info.sh                            01.01.001a00

  45   PMs/PM/hook.sh                             01.10.008
  46   PMs/PM/session.sh                          01.01.001a00
  47   PMs/PM/enumerate.sh                        01.01.001a01
  48   PMs/PM/list.sh                             01.10.008
  49   PMs/PM/info.sh                             01.01.001a00


CTYS-INTERNAL-SUBCALLS:

  Nr   Component                                  Version
  ------------------------------------------------------------
  00   ctys                                       01_10_011
  01   ctys-callVncserver.sh                      01_10_011
  02   ctys-callVncviewer.sh                      01_10_011
  03   ctys-createConfQEMU.sh                     01_10_011
  04   ctys-distribute.sh                         01_10_011
```

```
  05   ctys-dnsutil.sh                          01_10_011
  06   ctys-extractARPlst.sh                    01_10_011
  07   ctys-extractMAClst.sh                    01_10_011
  08   ctys-genmconf.sh                         01_10_011
  09   ctys-groups.sh                           01_10_011
  10   ctys-getMasterPid.sh                     01_10_011
  11   ctys-install.sh                          01_10_009
  12   ctys-install1.sh                         01_10_009
  13   ctys-macros.sh                           01_10_011
  14   ctys-macmap.sh                           01_10_011
  15   ctys-plugins.sh                          01_10_011
  16   ctys-vnetctl.sh                         01_10_011
  17   ctys-smbutil.sh                          01_10_011
  18   ctys-vdbgen.sh                           01_10_011
  19   ctys-vhost.sh                            01_10_011
  20   ctys.sh                             01_10_011
  21   ctys-wakeup.sh                           01_10_011
  22   ctys-xen-network-bridge.sh               01_10_011

  Helpers:

  00   getCPUinfo.sh                            01_10_011
  01   getFSinfo.sh                             01_10_011
  02   getHDDinfo.sh                            01_10_011
  03   getMEMinfo.sh                            01_10_011
  04   getPerfIDX.sh                            01_10_011
  05   getVMinfo.sh                             01_10_011

  Tiny-Helpers:

  00   getCurArch.sh                            OK
  01   getCurCTYSRel.sh                         OK
  02   getCurDistribution.sh                    OK
  03   getCurGID.sh                             OK
  04   getCurOS.sh                              OK
  05   getCurOSRel.sh                           OK
  06   getCurRelease.sh                         OK
  07   getSolarisUUID.sh                        OK
  08   pathlist.sh                              OK


----------------------------------------------------------------------
OPTIONAL/MANDATORY PREREQUISITES:


  bash:GNU bash, version 3.2.25(1)-release (x86_64-redhat-linux-gnu)

  SSH:OpenSSH_4.3p2, OpenSSL 0.9.8e-fips-rhel5 01 Jul 2008

  VNC:VNC Viewer Free Edition 4.1.2 for X - built Mar 24 2009 19:52:30
```

```
  wmctrl:wmctrl 1.07



----------------------------------------------------------------------
CURRENT ARG-MEM-USAGE:


  ArgList(bytes):"env|wc -c" => 3284
  ArgList(bytes):"set|wc -c" => 810677
```

Chapter 50

File Formats

50.1 Common Tools and Formats

The most common internal file format is the output by MACHINE suboption. This is a semicolon seperated file format, which optionally supports a TITLE line for each field. The format is compatible to MS-Excel.

The most important data collector and generator ctys-vdbgen is just a wrapper for the call of ctys with the action ENUMERATE. The replied data is transparently stored in the main file database "enum.fdb", which could be inspected and edited by any ASC-II editor or a common spreadsheet application. The caching boosts the performance beginning with the factor of 10, which could be much more, depending of various specific circumstances. Even though it could theoretically slow down, no practical example occurred.

The generated enum.fdb is just a raw static cache of distributed configuration data for VMs, which may lack some additional information. Therefore another file database is generated in order to map TCP/IP data of the Guest-OSs within VMs to their MAC-Addresses and DNS names. This file is called the macmap.fdb. It could be generated by the usage of the tools ctys-extractMAClst and/or ctys-extractARPlst.

The prefetched database "'.fdb" are pre-processed on a second level by aggregating and correlating the data into a common final static cache-database "ctysd-vhost.statcache.cdb". Additional dynamic intermediate data my be cached within tmp directory. The databases could be listed and inspected by the tool ctys-vhost.

ctys-vhost -S list
: Lists source databases of first level cache "fdb", which includes the non-cached group files.

ctys-vhost -C list
: Lists statcache caches of second level prefetch, which includes the cached group files.

Some additional options exist within ctys-vhost for listing of members and various output.

50.2 Groups

The groups concept as described within the introduction supports the assembly of arbitrate entities into a common entity, which will be handled as one logical instance. This includes the nesting of includes of groups. Therefore a file has to be supported in a defined subdirectory, with the name as the literal group name to be used within ctys arguments as a replacement for a <machine-address>.

The group file supports the keyword "#include" which has to be left-most on a line followed by a groupname. The number of includes and the level of nesting is not limited. Recursion loops are checked and aborted by a pre-defined level. The subdirectory containing the group will be search from a given list of directories by the path variable "CTYS_GROUPS_PATH", which uses a colon seperated list of directory paths.

The ordinary host entries could be one on each line, or a list of more than one comma-seperated entries on a line. For each entry could be context specific options set the same as on the command line arguments. An example with nested includes is supported in the install sub-directory conf/ctys/groups. Following is an example with context options.

```
#
#This groups contains all machines supporting VMs to be used by
#user acue.
#
#Almost any machine is currently accessible by NFS, thus allows
#a simple means of load-balancing throughout the memgers of
#this group.
#
#The only distinction has to be done by type of VM, which
#depends on the actual running kernel.
#
#
#REMINDER: Any option is dominant from it’s first occurance on,
#          until overwritten. This is one of the limits of
#          missing namespaces and smart usable assembled data
#          structures within bash. But anyway the real huge
#          benfit from using bash is the opportunity for almost
#          anyone to add his own modules by simple shell scripts.
```

```
#
#          It has to be recognized, that due to some basic
#          requirements - e.g. grouping of potential
#          Desktop/Workspace display, the jobs are re-ordered.

#          So it is recommended, that when any context-option
#          is required, that ALL might be assigned it's own
#          options.
#
#
#          So, it's accepted!
#
#
#OpenBSD PM, for tests only. Supports no VMs, Just OpenBSD-PM and
#common HOSTs-(CLI,X11,VNC)
root@host1'(-a enumerate=machine)'

#Multi-Display-WS with local emulation for usage of
#Win-Equipment, such as Dymo-LabelWriter 320 with it's own
#drivers within VMWare and W2K.
host2'(-a enumerate=machine)'

#WMware WS workstation, DualOpteron-244
host3'(-a enumerate=machine)' user5@host0'(-a enumerate=machine)'

#Experimental Dual-PIII-Coppermine, yum-install of
#VMware-server/player-variants

host4'(-a enumerate=machine)' user3@host5'(-a enumerate=machine)'

#Main fileserver with DualP-III-Thualatin
host5'(-a enumerate=machine)' user3@host10'(-a enumerate=machine)'

#Backup and Experimental HVM by Core2-Duo-6300
host6'(-a enumerate=machine)' user2@host9'(-a enumerate=machine)'

#Experimantal Celeron - S420
root@host6'(-a enumerate=machine)'

#Experimental Celeron 2,4GHz
root@host7'(-a enumerate=machine)'

#app1 as primary paravirtualized DualOpteron-Xen-Server
host8'(-a enumerate=machine,b:/homen/userA%/home1/xen)'
```

On the ctys call the group could be given context options, even when it's members have context options. But not the include-statements. Due to the philosophy, that the last wins, the outer . this is the CLI options - will replace options left of it, within the groups. When using ctys-vhost the groups will be cached as a pre-resolved subtree in common database (ENUMERATE) format. This will be recursively done for each group file as described in the following chapters.

50.3 Macros

Thus a macro can contain any part of a call except the command itself. The whole set of required options including the execution target or only a subset of options could be stored within a macro. The macro and it's content are stored within a file which could be edited by each user or provided as a common defaults file. A macro is defined within the default file "default" which is searched in the order:

1. "$HOME/.ctys/macros/default"
2. "<actual-call-conf-path>/macros/default"

The <actual-call-conf-path> is evaluated from the resolved symbolic link of the call. The following call syntax is provided:

```
MACRO:(
<macro-name>
[%<macro-file-db>]
  [%(
    ECHO
    |EVAL]
  )
]
)
```

MACROs could be nested and chained as required. Even though the recursion depth could be arbitrary a counter is implemented, which sets a threshold limiting recursive processing. This is set by the configuration variable CTYS_MAXRECURSE. The variable protects all recursion depths, thus should be handled carefully. Default is 15 levels. The keyword "MACRO" prefixes the actual macro alias with the following parts.

<macro-name>
: The actual name of the alias to be replaced.

<macro-file-db>
: The default macro file could be altered by this new filename. The "macros" directories will be scanned for a file with given name.

ECHO
: The given macro is inserted by "echo" command into the replacement position, which is the default behaviour.

EVAL
: The macro is evaluated on the callers site by "eval" call and the result is inserted into the insertion position.

50.4 Static Import File Databases - fdb

50.4.1 macmap.fdb

This file contains the output of the standard call to one of the tools "ctys-extractMAClst" or "ctys-extractARPlst". Which is a three colon semicolon seperated table, callee file-database. The record format is

<DNS-name>;<MAC-address>;<TCP/IP-address>

The default format is expected by the post-processing tools. The tools ctys-extractMAClst and ctys-extractARPlst could be used by themself for search and output to stdout too.

ctys-vhost allows by the flag "MACMAP" the optional usage of this database when only output suitable is selected for the "-o" option. Or allows the forced usage by "MACMAPONLY". ctys-macmap works natively on macmap.fdb.

50.4.2 enum.fdb

The enum.fdb file is the first level cached raw output from the ENUMERATE=MACHINE call. Therefore the call of ctys-vdbgen is used, which is a wrapper for the ctys call. Any option of ctys could be provided for ctys-vdbgen and will be passed through transparently. Therefore the flag "-T" and the given targets, e.g. as a group, could be used to constrain the collected and stored data. Using this features several databases could be used independently with different scopes of network view.

The record format is as shown with the call "ENUMERATE=MACHINE,TITLE":

```
"ContainingMachine;SessionType;Label;ID;UUID;MAC;TCP/IP;\
VNCAccessPort;VNCBasePort;VNCDisplay;Distribution;OS;\
VersionNr;SerialNr;Category"

host.soho;QEMU;tst100;\
/homen//tst100-01.01.001.x86\_64/tst100-inst.conf;\
;00:50:56:13:11:40;;;;;;;;;
```

Space are not allowed, multiple entries will be ignored, just the first will be used.

50.5 Static Pre-Fetch Cache Databases - cdb

The Pre-Fetch Caches are the second level, where some time consuming preprocessing and correlation is performed. This also includes the finale resolution of the group files by replacing each <machine-address> entry with the appropriate ENUMERATE result, and resolving the whole resulting include-tree.

The common enum.fdb entries are stored within one database file, whereas the groups are resolved with their whole tree for each defined group file.

The record format is the common ENUMERATE format.

50.5.1 statcache.cdb

This is the cached enum.fdb. In addition to enum.fdb the macmap.fdb is correlated to any ITCP relevant field. Therefore - if available - any item should now contain a MAC-address and the related TCP/IP-address. This allows by simple search operations the evaluation of the GuestOSs TCP/IP-address from the VMs configuration file. Thus the native access to the GuestOS.

50.6 Dynamic Runtime Cache Databases

Some internal data is cached into files, which could be controlled by flags. This is due to reuse of data spanning more than one call.

50.7 Configuration Entries for ctys

50.7.1 Actual Processing vs. Administrative Display

The first point to mention is the actual functionality supported by the various configuration options. The native configuration files of the various hypervisors support values in order to actually effect their managed VM. The parameters displayed with ENUMERATE option are read out from the native configuration with highest priority, where possible. When this fails or is simply not provided, the manual stored data is used.

50.7.2 Configuration File Variants

The various supported VMs have partly quite different configuration options for their VM files. Therefore some addons tho the existing files and additional specific files for ctys are defined. The basic convention for the naming and location of VMs configuration files is defined as:

1. Any VM has it's own directory, wherein all related files are stored. These are the bld/build/output/doc-tmp/en/images and the configuration files. This is commonly pre-required for all VMs.
2. The name of the directory is the same as the VM name, which could be the display name as well as a DomU name. For the directory name some variations could be set, e.g. due to versioning or backups. The tools like ctys-vhost search by default for the first match by alphabetical order only, which eliminates the usage of backup files.
3. The configuration file will be named the same as the containing directory, but with the appropriate post-fix, which is e.g. 'vmx' or 'conf'.
4. Additional files, to be optionally used for ctys information when required are:
   (a) Same filenamepath as the VMs config-file, but with the post-fix 'ctys'.
   (b) The same file stored in the parent directory.
   (c) When common definitions to be used, these finally are searched within the CTYSCONF file.

The scan for resolution will be performed until a match occurs, thus the first match wins. The order of search is:

1. <vmx-file>
   (a) The native configuration file of the VM. The scanning subsystem first tries to match standard information provided by the native syntax of the VM.
   (b) Second the ctys specific entries will be scanned.
2. ${<vmx-file>%.*}.ctys
   The ctys-file coallocated within the same directoy as the plugin specific configuration file.
3. 'dirname <vmx-file>'.ctys
   The parent directory of the plugins configuration directory.
4. ${CTYSCONF}
   The predefined ctys configuration file.

50.7.3 Keywords

The following keywords are defined:

#@#ARCH
: The supported architecture of the VM. For additional description refer to Section 50.7.7 'Virtual Hardware-Platform' on page 465 .

#@#CTYSRELEASE
The identified of the release, which was used to create the current record, helpful for distributed access.

#@#CATEGORY
The category of this Machine. Currently VM or PM.

#@#CSTRG
A private context string supported for the plugin.

#@#DIST
The distribution name of the GuestOS.

#@#DISTREL
The release of the distribution of the GuestOS.

#@#EXECLOCATION

#@#GATEWAY
The gateway to be used by the GuestOS. This could be registered specific to each interface.

#@#HWCAP
The provided harware capacity.

#@#HWREQ
The required harware capacity.

#@#HYPERREF
The release of the hypervisor used to create the VM.

#@#IP[0-9]*
The IP parameter is tightly correlated with the MAC parameter. For additional description refer to Section 50.7.4 'Interface Keywords' on page 461 .

#@#INST_EDITOR
Responsible person installed the VM.

#@#INST_DATE
Date of installation.

#@#INST_CTYSREL
CTYS-Release of installation.

#@#INST_VERNO
Version number of VM.

#@#INST_SERNO
Serial number of VM.

#@#INST_UID
User ID.

#@#INST_GID
Group ID.

#@#INST_HOST
Host of installer execution.

#@#INST_HOST_DIST
Distribution running on host.

#@#INST_HOST_DISTREL
Distribution release running on host.

#@#INST_HOST_OS
OS on host.

#@#INST_HOST_OSREL
OS version on host.

#@#INST_QEMUBASE
QEMU executable.

#@#INST_QEMUBASE_HYPERREL
QEMU executable release.

#@#INST_QEMUBASE_MAGICID
QEMU magic ID.

#@#INST_QEMUBASE_ACCELERATOR
Type of supported QEMU accelerator.

#@#INST_QEMUKVM
QEMU executable.

#@#INST_QEMUKVM_HYPERREL
QEMU with KVM executable release.

#@#INST_QEMUKVM_MAGICID
QEMU with KVM magic ID.

#@#INST_QEMUKVM_ACCELERATOR
Type of supported KVM accelerator.

#@#LABEL
Label of VM, which is used as unique name. Could be e.g. the display name or a DomU name.

#@#MAC[0-9]*
The MAC parameter is tightly correlated with the IP parameter. For additional description refer to Section 50.7.4 'Interface Keywords' on page 461 .

#@#MAGICID-<plugin>
A magicID to for classification of the managing plugin for this type. For additional description refer to Section 50.7.5 'MAGICID' on page 464 .

#@#OS
The name of the OS.

#@#OSREL
The release of the GuestOS.

#@#PLATFORM
The supported virtual HW as execution base for the GuestOS. Advanced support for this option is provided by QEMU. For additional description refer to Section 50.7.7 'Virtual Hardware-Platform' on page 465 .

#@#RELOCCAP
The provided reloaction capacity. For additional description refer to Section 50.7.9 'Execution Location and Relocation' on page 467 .

#@#SERNO
A free serial number for administration by the user. The tools generate the recommended format by utilising the "date" tool with the call:

date +%Y%m%d%H%M%S,

which might be sufficently unique, when seen within it's context.

#@#SESSIONTYPE
The type of session. This field will be evaluated by the plugin itself, passed through to the main dispatcher.

#@#SPORT
The serverport for management access to the hypervisor. This is supported by Xen and QEMU.

#@#STACKCAP
The offered stacking capacity of the VM. For additional description refer to Section 50.7.8 'Stacking Entries' on page 467 .

#@#STACKREQ
The required stacking capacity by the VM. For additional description refer to Section 50.7.8 'Stacking Entries' on page 467 .

#@#USTRG
An arbitrary user string. For now just trusted, means no specific validation is done. Particularly the size should be lept small, which means some bytes only, as a reminder for the VMs task for example.

#@#UUID
The UUID of the machine.

#@#VCPU
The number of configured virtual CPUs. For Xen called within Dom0 with the "PM" argument used the actual present CPUs, which is the sum of all CPU cores, is additionally displayed.

<Dom0-cpus>/<total-number-of-cores>

#@#VERSION
The release of the OS, which is the kernel release for Linux.

#@#VMSTATE
The state of the VM. For additional description refer to Section 50.7.6 'VMSTATE' on page 464 .

#@#VNCACCESSPORT
The port number of VNC access for the GuestOS, if required to be defined explicitly.

#@#VNCACCESSDISPLAY
The DISPLAY when required to be fixed.

#@#VNCBASEPORT
The baseport when to be calculated with usage of display.

#@#VRAM
The configured RAM for the GuestOS, when called with "VM".

For Xen called within Dom0 with the "PM" argument used the total amount of physical RAM is additionally displayed.

<Dom0-RAM>/<total-physical-RAM>

Additional data could be provided for and by the specific plugin. This is required for example in the specific configuration file defined for the QEMU plugin by ctys. Which has to be wrapped due to it's call-option only interface, to be unified in accordance to the common usage with the remaining VMs.

50.7.4 Interface Keywords

Specific consideration is required for the interface configuration and it's representation within the GuestOS, particularly when multiple interfaces are configured.

As mentioned before, the first attempt of the ENUMERATE action is to readout the native values of the VM. This is commonly for the MAC address only, where in case of XEN "arbitrary" default values are set when values are not present. In case of VMware the hypervisor eventually resets these values dynamically if not configured to be static.

Thus the first requirement of the UnifiedSessionsManager is the static configuration of MAC addresses.

The next point is the missing interface index within Xen and the varying prefix of the interfacenames within OpenBSD guests for example. Therefore ctys numbers the interface by the order they were found. This is done for the native configured interfaces within the configuration files, where no index is provided. In case of names containing a numbering part, this number is kept. When redundancy occurs, the first match wins, and a warning is displayed for the second, which is dropped. When applicable the binding of interface names to the actualy HW devices should be fixed, this is particularly true, when hotplug Ethernet devices are used. This could be done by configuring MAC addresses into ifcfg-scripts and/or by usage of such tools as "ifrename".

The next step is to evaluate eventually configured TCP/IP parameters, like the IP address from the configuration files. This is currently "somehow" supported by Xen only, but is not indexed too. Thus ctys adds an "IP[0-9*]" options, which provides the whole set of required parameters for the VMs as depicted within the following description. When redundancy occurs, the behaviour is the same as for MAC addresses.

The correlation between the MAC address representing the harware item, and the TCP/IP parameters representing the upper communications protocol layers is given by the index values evaluated before. Thus the first TCP/IP entity with the address "0" is assigned to the first AMC address with the same index. The rest is worked out as might be expected. For TCP/IP addresses without an assigne MAC interface a warning is displayed.
The following syntax is available to be applied.

```
<INDEXEDENTRY>[0-9]* = <entity>

  <entity> =:
    <elements>[,<entity>]

  <elements> =:
    <entry0>[%<entry1>[%<entry2>[%<entry3>[...]]]]
```

The specific adaption for MAC addresses:

```
MAC[0-9]* = <mac-entity>
```

Where a missing index is equal to 0. The specific adaption for IP:

```
IP[0-9]* = <ip-entity>

<ipentity> =:
  <ip-elements>[,<ip-entity>]
```

```
<ip-elements> =:
  [<dotted-IP-addr>]%[<netmask>]%[<relay>]\
  %[<ifname>]%[<ssh-port>]%[<gateway>]
```

Where a missing index is equal to 0. The values mostly are edited independent from the actual configuration. Thus particularly do not neccesarily represent a static configuration. In case of DHCP these should be configured too, but DHCP might use fixed address assignments.

This is also the style the ctys-extractMAClst utility relies on.

<dotted-IP-addr>
TCP/IP address in numerical form should be preferred. Netmask - if present - has to be provided within it's own field.

<netmask>
The netmask.

<relay> This is the local interconnection device, which could be a virtual bridge, switch/hub or a router. This version supports bridges and switches/hubs only. Support for shorewall will follow soon, check it out.

<ifname>
The name of the interface. Multiple IP addresses on the same interface are supported, thus could have the form "<ifname>:#ifnum". The consistency of the interface names is within the responsibility of the user.

<ssh-port>
An alternate port for SSH. OpenSSH supports different ports for each interface.

<gateway>
An individual gateway for networks to be accessed by current interface. Additional settings are required, else the default is used.

The following is an example for configuration of the IP addresses only.

```
#@#IP2 = "11.0.0.11,11.0.11.0,11.11.0.0"
```

The next example provides the full scope of possible information to be stored.

```
#@#IP3 = \
  "11.0.0.1%255.255.255.0%xenbr0%eth3%222,\
   11.0.1.0%255.255.255.0%xenbr1%eth3:1%223"
```

Yes, the <CR> is for LaTeX only.

50.7.5 MAGICID

The MAGICID defines specific behaviour for the filesystem scanners how to recognize the found configuration. When the plugin value only is found, it will immediately accept the file as a valid configuration. The values could be used for <plugin> in order to control the scan behaviour of ENUMERATE.

- <plugin>
  The SESSIONTYPE of the owning plugin.
- IGNORE
  Ignores the file without any further processing.
- NOENUM
  ffs.

50.7.6 VMSTATE

The state of the VM, currently ACTIVE is supported only.

50.7.7 Virtual Hardware-Platform

The virtual hardware platform is closely correlated to the architecture supported by the VM. These are dependent on the type and capabilities of the utilized hypervisor.

Currently the following non-comprehensive list of supported architectures and platforms is available, which depends partly on the actually used physical hardware base. For additional information refer to the specific documentation of the VMs.

- Architecture - ARCH
  - i386
    VMWARE, XEN, QEMU
  - x86_64, amd64
    VMWARE, XEN, QEMU
  - arm9
    QEMU, for an example with debian installation refer to Section **??** '**??**' on page **??** , for the provided example by QEMU refer to Section **??** '**??**' on page **??** .
  - coldfire, PowerPC, MIPS, SPARC, SH3, ...
    For display of a list of the current installed release of QEMU the INFO action could be used. An example display is provided within Section 18.7 'INFO' on page 215 .
    Preconfigured VM configuration files requireing less adaption to local filesystem are available for the provided examples by QEMU, for additional information refer to .
- Platform - PLATFORM
  - pc-standard-platform
    A standard PC platform with widely available emulated hardware components is supported by VMware, Xen, and QEMU. This includes particularly the option to configure the available RAM and the number of present CPUs.
  - ...
    QEMU supports a variety of architectures of standard and embedded devices. For additional information refer to user-manual.

The hardware relevant entries describe the present hardware. Therefore the following generic format to each entry is applied.

```
<entity> =:
[<#cnt>x]<elements>[%<add-elements>][,<entity>]

<elements> =:
<component>-<version>-<architecture>[-<opt-attr>]
<opt-attr> =: <attr>[-<opt-attr>]
<#cnt>     =: number present

<add-elements> =: <elements>[%<add-elements>]
```

The overall order of the entries contain the following elements, which could be either physical or virtual.

1. CPUs:

```
<vendor>-<family>-<model>-<stepping>-<frequency>-<cache>\
[-<VM-cap>]

<VM-cap> := (SVM|VMX)|PAE
```

This format could vary for non-Linux OSs. Non applicable or available but mandatory fields are padded with dots('.').

2. RAM:

```
<RAM>,<SWAP>
```

3. HDD:

   `<device-lst> := <device>[%<device-lst>]`

4. FS:

   `<home-lst> := <home>[0-9]*-<size>[%<home-lst>]`

   Currently the home-partitions onyl are displayed by "ctys-genmconf".

50.7.8 Stacking Entries

The stack relevant entries describe the offered VM and PM capacity as well as the required base. Therefore the following generic format to each entry is applied.

```
<entity> =: <elements>[%<add-elements>][,<entity>]

<elements> =:
<component>-<version>-<architecture>[-<opt-attr>]

<opt-attr> =: <attr>[-<opt-attr>]

<add-elements> =: <elements>[%<add-elements>]
```

The overall order of the entries contain the following elements.

1. VM plugin.
2. A list of present hypervisor capabilites.
3. Release of present stack support.
4. Additional information.

50.7.9 Execution Location and Relocation

The execution locations and the supported and/or allowed relocation behaviour. Currently just for display. For current supported values refer to RELOCCAP and EXECLOCATION .

Chapter 51

Miscellaneous

51.1 Basic EXEC principle

The basic execution principle of ctys is first to analyse the given options and build up an call array. Therefore several distinctions have to be made as resulting from permutation and superposing of the expanded CLI arguments. These array is finally executed in sets dependent from multiple criteria. Some examples are:

- Grouping of common sessions for each desktop, due to reliability and addressing gaps when shifting windows between desktops.
- Grouping of sessions for each remote server, but only if not ConnectionForward is choosen, because the current OpenSSH release does not support MuxDemux of multiple XSessions with different Displays.
- Splitting of Remote Server and Local Client execution of ctys, for VMs even though the VM-configuration is available on the server site only, which requires a remote component of ctys to be executed.
- ...and so on.

The implementation of ctys is pure bash with usage of the common shell components such as awk and sed. The whole design is based on unique set of sources which will be executed as the local initial call and client starter part as well as the remote server execution script. In case of DISPLAYFORWARDING the local component just initiates the remote co-allocated execution of Client and Server component. For the case of LOCALONLY both will be locally executed, thus the ctys acts locally as initial caller, server starter and client starter script.

To assure consistency and compatibility the remote and local versions will be checked and the execution is proceeded only in case of an match of both versions.

51.2 PATH

First of all, this is normally just required when handling different versions during development. This is particularly true, when during development a version is executed which is not contained within the standard PATH. This is particularly to be recognized, when executing the remote component which relies on the PATH mechanism too. Therefore the two environment variables are defined:

```
R_PATH:  Replaces path for remote execution.
L_PATH:  Replaces path for local execution.
```

The local component L_PATH is required for local execution too, because the following subcalls of ctys will be executed based on PATH mechanism, which is most often different to initial path-prefixed test-call. For example this is for calling a test version for starting a local client and remote server from a test path without changing PATH:

```
V=01\_01\_007a01;\
export R\_PATH=/mntbase/ctys/src/ctys.\$V/bin:\$PATH;\
export L\_PATH=\$R\_PATH;\
ctys.\$V/bin/ctys -t vmw \
   -a create=b:\$HOME/vmware/dir2\%\$HOME/vmware/dir3,\
      l:"GRP01-openbsd-4.0-001",REUSE
   -g 800x400+100+300:3 \
   -L CF \
   -- '(-d 99)' app2
```

The same for common user with standard install will be:

```
ctys -t vmw \
     -a create=\
        base:\$HOME/vmware/dir2\%\$HOME/vmware/dir3\
         ,label:"GRP01-openbsd-4.0-001"\
         ,REUSE
     -g 800x400+100+300:3 \
     -L CF \
     app2
```

51.3 Configuration files

The configuration of ctys is performed in 4 steps, first has highest priority.

1. Environment Variable If an environment variable is set, it dominates other settings and it's value is kept.
2. $HOME/.ctys/ctys.conf Config-File sourced: $HOME/.ctys/ctys.conf
3. <install dir>/conf/ctys.conf Config-File sourced: <install dir>/conf/ctys.conf
4. Embedded defaults in ctys.

Chapter 52

LICENSES

Additionally a seperate document including all licenses is contained within the package.

ctys-Licenses-01.11-print.pdf

52.1 CCL-3.0 With Attributes

```
License

THE WORK (AS DEFINED BELOW) IS PROVIDED UNDER THE TERMS OF THIS
CREATIVE COMMONS PUBLIC LICENSE ("CCPL" OR "LICENSE"). THE WORK IS
PROTECTED BY COPYRIGHT AND/OR OTHER APPLICABLE LAW. ANY USE OF THE
WORK OTHER THAN AS AUTHORIZED UNDER THIS LICENSE OR COPYRIGHT LAW IS
PROHIBITED.

BY EXERCISING ANY RIGHTS TO THE WORK PROVIDED HERE, YOU ACCEPT AND
AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE. TO THE EXTENT THIS
LICENSE MAY BE CONSIDERED TO BE A CONTRACT, THE LICENSOR GRANTS YOU
THE RIGHTS CONTAINED HERE IN CONSIDERATION OF YOUR ACCEPTANCE OF SUCH
TERMS AND CONDITIONS.

1. Definitions

   1. "Adaptation" means a work based upon the Work, or upon the Work
   and other pre-existing works, such as a translation, adaptation,
   derivative work, arrangement of music or other alterations of a
   literary or artistic work, or phonogram or performance and includes
   cinematographic adaptations or any other form in which the Work may
   be recast, transformed, or adapted including in any form
   recognizably derived from the original, except that a work that
   constitutes a Collection will not be considered an Adaptation for
   the purpose of this License. For the avoidance of doubt, where the
   Work is a musical work, performance or phonogram, the
   synchronization of the Work in timed-relation with a moving image
   ("synching") will be considered an Adaptation for the purpose of
   this License.

   2. "Collection" means a collection of literary or artistic works,
   such as encyclopedias and anthologies, or performances, phonograms
```

```
or broadcasts, or other works or subject matter other than works
listed in Section 1(f) below, which, by reason of the selection and
arrangement of their contents, constitute intellectual creations,
in which the Work is included in its entirety in unmodified form
along with one or more other contributions, each constituting
separate and independent works in themselves, which together are
assembled into a collective whole. A work that constitutes a
Collection will not be considered an Adaptation (as defined above)
for the purposes of this License.

3. "Distribute" means to make available to the public the original
and copies of the Work through sale or other transfer of ownership.

4. "Licensor" means the individual, individuals, entity or entities
that offer(s) the Work under the terms of this License.

5. "Original Author" means, in the case of a literary or artistic
work, the individual, individuals, entity or entities who created
the Work or if no individual or entity can be identified, the
publisher; and in addition (i) in the case of a performance the
actors, singers, musicians, dancers, and other persons who act,
sing, deliver, declaim, play in, interpret or otherwise perform
literary or artistic works or expressions of folklore; (ii) in the
case of a phonogram the producer being the person or legal entity
who first fixes the sounds of a performance or other sounds; and,
(iii) in the case of broadcasts, the organization that transmits
the broadcast.

6. "Work" means the literary and/or artistic work offered under the
terms of this License including without limitation any production
in the literary, scientific and artistic domain, whatever may be
the mode or form of its expression including digital form, such as
a book, pamphlet and other writing; a lecture, address, sermon or
other work of the same nature; a dramatic or dramatico-musical
work; a choreographic work or entertainment in dumb show; a musical
composition with or without words; a cinematographic work to which
are assimilated works expressed by a process analogous to
cinematography; a work of drawing, painting, architecture,
sculpture, engraving or lithography; a photographic work to which
are assimilated works expressed by a process analogous to
photography; a work of applied art; an illustration, map, plan,
sketch or three-dimensional work relative to geography, topography,
architecture or science; a performance; a broadcast; a phonogram; a
compilation of data to the extent it is protected as a
copyrightable work; or a work performed by a variety or circus
performer to the extent it is not otherwise considered a literary
or artistic work.

7. "You" means an individual or entity exercising rights under this
License who has not previously violated the terms of this License
with respect to the Work, or who has received express permission
```

from the Licensor to exercise rights under this License despite a previous violation.

8. "Publicly Perform" means to perform public recitations of the Work and to communicate to the public those public recitations, by any means or process, including by wire or wireless means or public digital performances; to make available to the public Works in such a way that members of the public may access these Works from a place and at a place individually chosen by them; to perform the Work to the public by any means or process and the communication to the public of the performances of the Work, including by public digital performance; to broadcast and rebroadcast the Work by any means including signs, sounds or images.

9. "Reproduce" means to make copies of the Work by any means including without limitation by sound or visual recordings and the right of fixation and reproducing fixations of the Work, including storage of a protected performance or phonogram in digital form or other electronic medium.

2. Fair Dealing Rights. Nothing in this License is intended to reduce, limit, or restrict any uses free from copyright or rights arising from limitations or exceptions that are provided for in connection with the copyright protection under copyright law or other applicable laws.

3. License Grant. Subject to the terms and conditions of this License, Licensor hereby grants You a worldwide, royalty-free, non-exclusive, perpetual (for the duration of the applicable copyright) license to exercise the rights in the Work as stated below:

1. to Reproduce the Work, to incorporate the Work into one or more Collections, and to Reproduce the Work as incorporated in the Collections; and,

2. to Distribute and Publicly Perform the Work including as incorporated in Collections.

The above rights may be exercised in all media and formats whether now known or hereafter devised. The above rights include the right to make such modifications as are technically necessary to exercise the rights in other media and formats, but otherwise you have no rights to make Adaptations. Subject to 8(f), all rights not expressly granted by Licensor are hereby reserved, including but not limited to the rights set forth in Section 4(d).

4. Restrictions. The license granted in Section 3 above is expressly made subject to and limited by the following restrictions:

1. You may Distribute or Publicly Perform the Work only under the terms of this License. You must include a copy of, or the Uniform Resource Identifier (URI) for, this License with every copy of the

Work You Distribute or Publicly Perform. You may not offer or impose any terms on the Work that restrict the terms of this License or the ability of the recipient of the Work to exercise the rights granted to that recipient under the terms of the License. You may not sublicense the Work. You must keep intact all notices that refer to this License and to the disclaimer of warranties with every copy of the Work You Distribute or Publicly Perform. When You Distribute or Publicly Perform the Work, You may not impose any effective technological measures on the Work that restrict the ability of a recipient of the Work from You to exercise the rights granted to that recipient under the terms of the License. This Section 4(a) applies to the Work as incorporated in a Collection, but this does not require the Collection apart from the Work itself to be made subject to the terms of this License. If You create a Collection, upon notice from any Licensor You must, to the extent practicable, remove from the Collection any credit as required by Section 4(c), as requested.

2. You may not exercise any of the rights granted to You in Section 3 above in any manner that is primarily intended for or directed toward commercial advantage or private monetary compensation. The exchange of the Work for other copyrighted works by means of digital file-sharing or otherwise shall not be considered to be intended for or directed toward commercial advantage or private monetary compensation, provided there is no payment of any monetary compensation in connection with the exchange of copyrighted works.

3. If You Distribute, or Publicly Perform the Work or Collections, You must, unless a request has been made pursuant to Section 4(a), keep intact all copyright notices for the Work and provide, reasonable to the medium or means You are utilizing: (i) the name of the Original Author (or pseudonym, if applicable) if supplied, and/or if the Original Author and/or Licensor designate another party or parties (e.g., a sponsor institute, publishing entity, journal) for attribution ("Attribution Parties") in Licensor's copyright notice, terms of service or by other reasonable means, the name of such party or parties; (ii) the title of the Work if supplied; (iii) to the extent reasonably practicable, the URI, if any, that Licensor specifies to be associated with the Work, unless such URI does not refer to the copyright notice or licensing information for the Work. The credit required by this Section 4(c) may be implemented in any reasonable manner; provided, however, that in the case of a Collection, at a minimum such credit will appear, if a credit for all contributing authors of Collection appears, then as part of these credits and in a manner at least as prominent as the credits for the other contributing authors. For the avoidance of doubt, You may only use the credit required by this Section for the purpose of attribution in the manner set out above and, by exercising Your rights under this License, You may not implicitly or explicitly assert or imply any connection with, sponsorship or endorsement by the Original Author, Licensor and/or

Attribution Parties, as appropriate, of You or Your use of the Work, without the separate, express prior written permission of the Original Author, Licensor and/or Attribution Parties.

4.

For the avoidance of doubt:

1. Non-waivable Compulsory License Schemes. In those jurisdictions in which the right to collect royalties through any statutory or compulsory licensing scheme cannot be waived, the Licensor reserves the exclusive right to collect such royalties for any exercise by You of the rights granted under this License;

2. Waivable Compulsory License Schemes. In those jurisdictions in which the right to collect royalties through any statutory or compulsory licensing scheme can be waived, the Licensor reserves the exclusive right to collect such royalties for any exercise by You of the rights granted under this License if Your exercise of such rights is for a purpose or use which is otherwise than noncommercial as permitted under Section 4(b) and otherwise waives the right to collect royalties through any statutory or compulsory licensing scheme; and,

3. Voluntary License Schemes. The Licensor reserves the right to collect royalties, whether individually or, in the event that the Licensor is a member of a collecting society that administers voluntary licensing schemes, via that society, from any exercise by You of the rights granted under this License that is for a purpose or use which is otherwise than noncommercial as permitted under Section 4(b).

5. Except as otherwise agreed in writing by the Licensor or as may be otherwise permitted by applicable law, if You Reproduce, Distribute or Publicly Perform the Work either by itself or as part of any Collections, You must not distort, mutilate, modify or take other derogatory action in relation to the Work which would be prejudicial to the Original Author's honor or reputation.

5. Representations, Warranties and Disclaimer

UNLESS OTHERWISE MUTUALLY AGREED BY THE PARTIES IN WRITING, LICENSOR OFFERS THE WORK AS-IS AND MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND CONCERNING THE WORK, EXPRESS, IMPLIED, STATUTORY OR OTHERWISE, INCLUDING, WITHOUT LIMITATION, WARRANTIES OF TITLE, MERCHANTIBILITY, FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT, OR THE ABSENCE OF LATENT OR OTHER DEFECTS, ACCURACY, OR THE PRESENCE OF ABSENCE OF ERRORS, WHETHER OR NOT DISCOVERABLE. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO SUCH EXCLUSION MAY

NOT APPLY TO YOU.

6. Limitation on Liability. EXCEPT TO THE EXTENT REQUIRED BY APPLICABLE LAW, IN NO EVENT WILL LICENSOR BE LIABLE TO YOU ON ANY LEGAL THEORY FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, PUNITIVE OR EXEMPLARY DAMAGES ARISING OUT OF THIS LICENSE OR THE USE OF THE WORK, EVEN IF LICENSOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

7. Termination

1. This License and the rights granted hereunder will terminate automatically upon any breach by You of the terms of this License. Individuals or entities who have received Collections from You under this License, however, will not have their licenses terminated provided such individuals or entities remain in full compliance with those licenses. Sections 1, 2, 5, 6, 7, and 8 will survive any termination of this License. 2. Subject to the above terms and conditions, the license granted here is perpetual (for the duration of the applicable copyright in the Work). Notwithstanding the above, Licensor reserves the right to release the Work under different license terms or to stop distributing the Work at any time; provided, however that any such election will not serve to withdraw this License (or any other license that has been, or is required to be, granted under the terms of this License), and this License will continue in full force and effect unless terminated as stated above.

8. Miscellaneous

1. Each time You Distribute or Publicly Perform the Work or a Collection, the Licensor offers to the recipient a license to the Work on the same terms and conditions as the license granted to You under this License.

2. If any provision of this License is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this License, and without further action by the parties to this agreement, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable.

3. No term or provision of this License shall be deemed waived and no breach consented to unless such waiver or consent shall be in writing and signed by the party to be charged with such waiver or consent.

4. This License constitutes the entire agreement between the parties with respect to the Work licensed here. There are no understandings, agreements or representations with respect to the Work not specified here. Licensor shall not be bound by any additional provisions that may appear in any communication from

Bibliography

Books

UNIX

[1] Juergen Gulbins: *UNIX Version 7, bis System V.3,* Springer-Verlag Berlin Heidelberg; 1988; ISBN: 3-540-19248-4

[2] Maurice J. Bach: *The Design Og The UNIX Operating System,* Prentice Hall, Inc.; 1986; ISBN: 0-13-201757-1

[3] Sebastian Hetze, Dirk Hohndel, Martin Mueller, Olaf Kirch: *LINUX Anwender Handbuch,* LunetIX Softair; 1994; ISBN: 3-929764-03-2

[4] Rob Flickenger: *LINUX SERVER HACKS,* O'Reilly&Associates, Inc.; 2003; ISBN: 0-596-00461-3

[5] Olaf Kirch: *LINUX Network Administrator's Guide,* O'Reilly&Associates, Inc.; 1995; ISBN: 0-56592-087-2

[6] Daniel P. Bovet, Marco Cesati: *Understanding the LINUX KERNEL,* O'Reilly&Associates, Inc.; 2003; ISBN: 0-596-00002-2

[7] Daniel P. Bovet, Marco Cesati: *Understanding the LINUX KERNEL(2nd. Ed.),* O'Reilly&Associates, Inc.; 2003; ISBN: 0-596-00213-0

[8] Wolfgang Mauerer: *Linux Kernelarchitektur - Konzepte, Strukturen und Algorithmen von Kernel 2.6,* Carl Hanser Verlag Muenchen-Wien; 2004; ISBN: 3-446-22566-8

[9] Alessandro Rubini: *LINUX Device Drivers,* O'Reilly&Associates, Inc.; 1998; ISBN: 1-565592-292-1

[10] Alessandro Rubini, Jonathan Corbet: *LINUX Device Drivers(2nd. Ed.),* O'Reilly&Associates, Inc.; 2001; ISBN: 0-596-00008-1

[11] Jonathan Corbet, Alessandro Rubini, Greg Kroah-Hartman: *LINUX Device Drivers(3nd. Ed.),* O'Reilly&Associates, Inc.; 2005; ISBN: 0-596-00590-3

[12] Juergen Quade, Eve-Katharina Kunst: *Linux-Treiber entwickeln,* dpunkt.verlag GmbH; 2004; ISBN: 3-89864-238-0

[13] Wehrle, Paehlke, Ritter, Mueller, Bechler: *Linux Netzwerkarchitektur - Design und Implementerung von Netzwerkprotokollen im Linux-Kern,* Addison Wesley, Inc.; 2002; ISBN: 3-8273-1509-3

[14] Brandon Palmer, Jose Nazario: *Secure Architectures with OpenBSD,* Addison Wesley, Inc.; 2004; ISBN: 0-321-19366-0

[15] Michael W. Lucas: *Absolute OpenBSD - UNIX for the Practical Paranoid,* No Starch Press, Inc.; 2003; ISBN: 1-886411-99-9

[16] Chris Tyler: *FEDORA LINUX,* O'Reilly&Associates, Inc.; 2006; ISBN: 1-596-52682-2

[17] Benjamin Mako Hill, Jono Bacon, Corey Burger, Jonathan Jesse, Ivan Krstic: *The Official ubuntu Book,* Pearson Education, Inc.; 2007; ISBN: 0-13-243594-2

[18] Jonathan Oxer, Kyle Rankin, Bill Childers: *UBUNTU Hacks - Tips&Tools for Exploring, Using, and Tuning Linux,* O'Reilly&Associates, Inc.; 2006; ISBN: 1-596-52720-9

[19] Tim SchÃ¼rmann: *(K)Ubuntu - Installieren - Einrichten - Anwenden,* Open Source Press; 2006; ISBN: 3-937514-30-9

[20] Michael Urban, Brian Thiemann: *FreeBSD 6 Unleashed,* Sams Publishing, Inc.; 2006; ISBN: 0-672-32875-5

[21] Marshall Kirk McKusick, George V. Neville-Neil: *The Design and Implementation of the FreeBSD 6 Operating System,* Addison Wesley, Inc.; 2005; ISBN: 0-201-70245-2

Security

[22] Guenter Schaefer: *Netzsicherheit - Algorithmische Grundlagen und Protokolle,* dpunkt.verlag GmbH; 2003; ISBN: 3-89864-212-7

[23] Alexande Geschonneck: *Computer Forensik - Systemeinbrueche erkennen, ermitteln, aufklaeren,* dpunkt.verlag GmbH; 2004; ISBN: 3-89864-253-4

[24] Jonathan Hassel: *RADIUS - Securing Public Access to Privaze Resources,* O'Reilly&Associates, Inc.; 2002; ISBN: 0-596-00322-6

[25] Jason Garman: *Kerberos - The Definitive Guide,* O'Reilly&Associates, Inc.; 2003; ISBN: 1-596-00403-6

[26] Daniel J. Barret, Richard E. Silverman & Robert G.Byrnes: *SSH The Secure Shell - The Definitive Guide,* O'Reilly&Associates, Inc.; 2005; ISBN: 1-596-00895-3

[27] John Viega, Matt Messier & Pravir Chandra: *Network Security with OpenSSL,* O'Reilly&Associates, Inc.; 2002; ISBN: 0-596-00270-X

[28] Jonathan Hassel: *RADIUS,* O'Reilly&Associates, Inc.; 2002; ISBN: 0-596-00322-6

[29] Gerald Carter: *LDAP - System Administration,* O'Reilly&Associates, Inc.; 2003; ISBN: 1-596592-491-6

[30] Matthias Reinwarth, Klaus Schmidt: *Verzeichnisdienste - Telekommunikation Aktuell,* VDE Verlag GmbH; 1999; ISBN: 3-8007-2373-5

[31] Dieter Kluenter, Jochen Laser: *LDAP verstehen, OpenLDAP einsetzen,* dpunkt.verlag GmbH; 2003; ISBN: 3-89864-217-8

[32] Daniel J. Barret, Richard E. Silverman & Robert G.Byrnes: *LINUX Security Cookbook,* O'Reilly&Associates, Inc.; 2003; ISBN: 1-565-00391-9

[33] Charlie Scott, Paul Wolfe & Mike Erwin: *Virtual Private Networks,* O'Reilly&Associates, Inc.; 1999; ISBN: 1-565592-529-7

[34] Bill McCarty: *SELINUX - NSA's Open Source Security Enhanced Linux,* O'Reilly&Associates, Inc.; 2004; ISBN: 1-565-007161-7

[35] Rober L. Ziegler: *Linux Firewalls,* New Riders Publishing, Inc.; 2000; ISBN: 0-7357-0900-9

[36] D. Brent Chapman and Elisabeth D. Zwicky: *Einrichten von Internet Firewalls,* O'Reilly&Associates, Inc.; 1996; ISBN: 3-930673-31-2

[37] Wolfgang Barth: *Das Firewall-Buch,* Nicolaus Millin Verlag GmbH; 2004; ISBN: 3-89990-128-2

[38] Joseph Kong: *Designing BSD Rootkits - An Introduction to Kernel Hacking,* No Starch Press, Inc.; 2007; ISBN: 978-1-593271-142-8

[39] Cyrus Peikari, Anton Chuvakin (Peter Klicman, Andreas Bildstein, Gerald Richter): *Kenne deinen Feind - Fortgeschrittene Sicherheitstechniken,* O'Reilly&Associates, Inc.; 2004; ISBN: 3-89721-376-1

[40] Ryan Russel et al.: *Die mitp-Hacker-Bibel,* mitp-Verlag/Bonn; 2002; ISBN: 3-8266-0826-3

[41] Wallace Wang: *Steal This Computer Book 4.0 - What They Won't Tell You About The Internet,* No Starch Press, Inc.; 2007; ISBN: 1-59327-105-0

[42] Johnny Long: *Google Hacking - For Penetration Testers,* Syngress Publishing, Inc.; 2005; ISBN: 1-931836-36-1

[43] Thomas Bechtold, Peter Heinlein: *Snort, Acid & Co. - Einbruchserkennung mit Linux,* Open Source Press; 2004; ISBN: 3-937514-01-3

[44] Syngress Autorenteam: *Snort 2.0 Intrusion Detection,* mitp-Verlag/Bonn; 2003; ISBN: 3-6266-1304-X

Networks

[45] Charles E. Spurgeon: *Ethernet - The Definitive Guide,* O'Reilly&Associates, Inc.; 2000; ISBN: 1-56592-660-9

[46] Olaf Kirch: *LINUX - Network Administrators Guide,* O'Reilly&Associates, Inc.; 1995; ISBN: 1-56592-087-2

[47] Hal Stern, Mike Eisler & Ricardo Labiaga: *Managing NFS and NIS,* O'Reilly&Associates, Inc.; 2001; ISBN: 1-56592-510-6

[48] Paul Albitz & Cricket Liu: *DNS and Bind,* O'Reilly&Associates, Inc.; 2001; ISBN: 1-596-00158-4

[49] James M. Kretchmar: *Open Source Network Administration,* Pearson Education, Inc.; 2004; ISBN: 0-13-046210-1

[50] Douglas R. Mauro, Kevin J. Schmidt: *Essential SNMP(1.nd Ed.,* O'Reilly&Associates, Inc.; 2001; ISBN: 0-596-00020-0

[51] Douglas R. Mauro, Kevin J. Schmidt: *Essential SNMP(2.nd Ed.,* O'Reilly&Associates, Inc.; 2005; ISBN: 0-596-00840-6

[52] James D. Murray: *Wimdows NT SNMP,* O'Reilly&Associates, Inc.; 1998; ISBN: 1-56592-338-3

[53] Marshall T. Rose: *The Simple Book(2nd. Ed.),* Prentice Hall, Inc.; 1996; ISBN: 0-13-451659-1

[54] David Perkins, Evan McGinnis: *Understanding SNMP MIBs,* Prentice Hall, Inc.; 1997; ISBN: 0-13-437708-7

[55] Mathias Hein, David Griffiths: *SNMP Simple Network Management Protocol Version 2,* International Thomson Publishing GmbH; 1994; ISBN: 3-929821-51-6

[56] Peter Erik Mellquist: *SNMP++ An Object-Oriented Approach to Developing Network Management Applications,* Hewlett-Packard(TM) Professional Books; 1997; ISBN: 0-13-264607-2

[57] Marshall T. Rose: *The Open Book - A Practical Perspective on OSI,* Prentice Hall, Inc.; 1990; ISBN: 0-13-643016-3

[58] Uyless Black: *Network Management Standards(2nd.Ed.),* McGraw-Hill, Inc.; 1994; ISBN: 0-07-005570-X

[59] Wolfgang Barth: *NAGIOS - System and Network Monitoring,* No Starch Press, Inc.; 2006; ISBN: 1-59327-070-4

[60] Heinz-Gerd Hegering, Sebastian Abeck, Bernhard Neumair: *Integriertes Management vernetzter Systeme,* dpunkt.verlag; 1999; ISBN: 3-932588-16-9

[61] Walter Gora, Reinhard Speyerer: *ASN.1 Abstract Syntax Notation One(2nd.Ed.),* DATACOM-Verlag; 1990; ISBN: 3-89238-023-6

[62] Klaus H. Stoettinger: *Das OSI-Referenzmodell,* DATACOM-Verlag; 1989; ISBN: 3-89238-021-X

Embedded Systems

[63] Derek J. Hatley,Imtiaz A. Pirbhai: *Strategien fuer die Echtzeit-Programmierung,* Carl Hanser Verlag Muenchen-Wien; 1988; ISBN: 3-446-16288-7

[64] Edmund Jordan: *Embedded Systeme mit Linux programmieren - GNU-Softwaretools zur Programmierung ARM-basierender Systeme,* Franzis Verlag GmbH; 2004; ISBN: 3-7723-5599-4

[65] Michael Barr, Anthony Massa: *Programming Embedded Systems(2nd.Ed.),* O'Reilly&Associates, Inc.; 2006; ISBN: 1-596-00983-6

[66] John Lombardo: *Embedded Linux,* New Riders Publishing; 2001; ISBN: 0-7357-0998-X

[67] Craig Hollabaugh, Ph.D.: *Embedded Linux - Hardware, Software, and Interfacing,* Addison Wesley, Inc.; 2002; ISBN: 0-672-32226-9

[68] Bob Smith, John Hardin, Graham Phillips, and Bill Pierce: *LINUX APPLIANCE DESIGN - A Hands-On Guide to Building Linux Appliances,* No Starch Press, Inc.; 2007; ISBN: 978-1-59327-140-4

[69] David E. Simon: *An Embedded Software Primer,* Addison Wesley, Inc.; 1999; ISBN: 0-201-61569-X

[70] John Waldron: *Introduction to RISC Assembly Language Programming,* Addison Wesley, Inc.; 1999; ISBN: 0-201-39828-1

[71] Steve Furber: *ARM System Architecture,* Addison Wesley, Inc.; 1996; ISBN: 0-201-40352-8

Online References

Obvious, but to be written due to German-Law:
"It should be recognized, that the given links within this and following sections are solely owned by and are within the exclusive responsibility of their owners. The intention to reference to that sites is neither to take ownership of their work, nor to state commitment to any given statement on their sites. It is much more to honour the work, and thank to the help, the author advanced from by himself. Last but not least, the intention is to support a short cut for the users of UnifiedSessionsManager to sources of required help."

OSs

[72] RedHat(TM) Enterprise Linux: *http://www.redhat.com*

[73] RedHat(TM) Enterprise Linux-Doc: *http://www.redhat.com/docs*

[74] CentOS: *http://www.centos.org*

[75] CentOS-Doc: *http://www.centos.org/docs*

[76] Scientific Linux: *http://www.scientificlinux.org*

[77] Debian: *http://www.debian.org*

[78] OpenSuSE: *http://www.opensuse.org*

[79] OpenBSD: *http://www.openbsd.org*

[80] OpenBSD-FAQ: *http://www.openbsd.org/faq/index.html*

[81] OpenBSD-PF: *http://www.openbsd.org/faq/pf/index.html*

[82] FreeBSD: *http://www.freebsd.org*

[83] NetBSD: *http://www.netbsd.org*

[84] uCLinux: *http://www.uclinux.org*

[85] Linux on ARM: *http://www.linux-arm.com*

[86] eCos: *http://ecos.sourceware.org*

Hypervisors/Emulators

kvm

[87] KVM: *http://de.wikipadia.org/wiki/Kernel_ based_ Virtual_ Machine*

QEMU

[88] QEMU(TM): *http://www.qemu.org*

[89] QEMU-CPU support: *http://fabrice.bellard.free.fr/qemu/status.html*

[90] QEMU-User Manual: *http://fabrice.bellard.free.fr/qemu/qemu-doc.html*

[91] QEMU-OS support: *http://www.claunia.com/qemu*

[92] QEMU-Debian: *http://909ers.apl.washington.edu/ dushaw/ARM*

[93] QEMU-Debian on an emulated ARM machine; Aurelien Jarno: *http://www.aurel32.net*

[94] QEMU-Debian kernel and initrd for qemu-arm-versatile; Aurelien Jarno: *http://people.debian.org/ aurel32*

[95] QEMU-Debian ARM Linux on QEMU; Brian Dushaw: *http://909ers.apl.washington.edu/ dushaw/ARM*

[96] QEMU-Ubuntu-Installation/QemuEmulator: *http://help.ubuntu.com/community/Installation/QemuEmulator*

[97] QEMU-Ubuntu-VMwarePlayerAndQemu: *http://wiki.ubuntu.com/VMwarePlayerAndQemu*

[98] QEMU-OpenBSD: *http://141.48.37.144/openbsd/qemu.html - Dag Leine Institut fuer Physikalische Chemie, Martin Luther Universitaet Halle*

[99] QEMU-NetBSD - Running NetBSD on emulated hardware: *http://www.netbsd.org/ports/emulators.html*

[100] QEMU-OpenSolaris: *http://www.opensolaris.org/os/project/qemu/host*

[101] QEMU-OpenSolaris Networking: *http://www.opensolaris.org/os/project/qemu/Qemu_ Networking*

[102] QEMUlator: *http://qemulator.createweb.de*

SkyEye

[103] SkyEye: *http://skyeye.sourceforge.net*

VMware

[104] VMware(TM) Inc.: *http://www.vmware.com*

[105] VMware-OpenBSD - HowTo Install VMWare-Tools: *http://openbsd-wiki.org*

[106] VMware-Communities: *http://communities.vmware.com*

[107] VMware-Forum: *http://vmware-forum.de*

[108] VMware-Any-Any-Patch from Petr Vandrovec: *http://kinikovny.cvnt.cz/ftp/pub/vmware*

[109] Open Virtual Machine Tools: *http://open-vm-tools.sourceforge.net*

Xen

[110] RED HAT(TM) ENTERPRISE LINUX 5.1 - Virtualization: *http://www.redhat.com/rhel/virtualization*

[111] Virtual Machine Manager - VMM: *http://virtual-manager.et.redhat.com*

[112] libvirt: *http://www.libvirt.org*

[113] Xen(TM): *http://www.xen.org*

[114] XenWiki - Xen-Networking: *http://wiki.xensource.com/xenwiki/XenNetworking*

[115] Xen-CentOS: *Creating and Installing a CentOS5 domU instance*

[116] Xen-Fedora: *http://fedoraproject.org/wiki/FedoraXenQuickstartFC6*

[117] Xen-OpenSolaris: *http://www.opensolaris.org/os/community/xen*

[118] Gerd Hoffmann: *"Install SuSE as Xen guest."*

Security

[119] OpenSSH: *http://www.openssh.org*

[120] SSH Communications Security(TM): *http://www.ssh.com*

[121] SSH Tectia(TM): *SSH Tectia*

[122] OpenSSH-FAQ: *http://www.openbsd.org/openssh/faq.html*

[123] OpenSSL: *http://www.openssl.org*

[124] OpenLDAP: *http://www.openldap.org*

[125] MIT-Kerberos: *http://web.mit.edu/kerberos/www*

[126] Heimdal-Kerberos: *http://www.h5l.org*

[127] sudo - "superuser do"(check google for actual link): *http://www.sudo.ws*

Specials

FreeDOS

[128] *FreeDOS*

[129] *Balder*

Dynagen/Dynamips

[130] Dynagen, by Greg Anuzelli:
Dynagen

[131] Dynamips, Cisco(TM) 7200 Simulator:
Dynamips, Cisco(TM) 7200 Simulator

QEMU-Networking with VDE

[132] VDE-Virtual Distributed Ethernet:
http://sourceforge.net/projects/vde

[133] VirtualSquare: *http://virtualsquare.org*

[134] VirtualSquare: *Basic Networking*

PXE

[135] By H. Peter Anvin: *SYSLINUX - PXELINUX -ISOLINUX*

[136] PXE-ROM-Images: *Etherboot*

Routing

[137] "Linux Advanced Routing&Traffic Control", by Bert Hubert:
LARTC-HOWTO

Scratchbox

[138] *Scratchbox*

Serial-Console

[139] By van Emery: *"Linux Serial Console HOWTO"*

[140] By David S. Lawyer / Greg Hawkins: *"Serial-HOWTO"*

[141] By David S. Lawyer: *"Text-Terminal-HOWTO"*

Miscellaneous

[142] IEEE: *http://www.ieee.org*

[143] The GNU Netcat: *Netcat*

[144] Netcat Wikipedia: *Netcat Wikipedia*

[145] By van Emery: *"Linux Gouge"*

UnfiedSessionsManager Versions

[146] The first public version of 2008.02.11, by Arno-Can Uestuensoez. Available as online help only by "ctys -H print"(more than 230pg. as ACII-Only): *"UnifiedSesionsManager" http://sourceforge.net/projects/ctys*

[147] The second public version of 2008.07.10, by Arno-Can Uestuensoez: *"UnifiedSesionsManager" http://sourceforge.net/projects/ctys*

[148] The second public version with minor updates of 2008.08.6, by Arno-Can Uestuensoez: *"UnifiedSesionsManager" http://sourceforge.net/projects/ctys*

[149] Minor editorial updates of 2008.08.12, by Arno-Can Uestuensoez: *"UnifiedSesionsManager" http://sourceforge.net/projects/ctys*

[150] Enhanced documentation, 2008.08.16, by Arno-Can Uestuensoez: *"UnifiedSesionsManager" http://sourceforge.net/projects/ctys*

[151] Major enhancements and feature updates, beginning 2010/02, by Arno-Can Uestuensoez: *"UnifiedSesionsManager" http://sourceforge.net/projects/ctys*

Sponsored OpenSource Projects

Support is available exclusively by direct contact only.

[152] Ingenieurbuero Arno-Can Uestuensoez - OpenSource:
http://www.i4p.org

[153] UnifiedSessionsManager:
http://ctys.sourceforge.net/
http://sourceforge.net/projects/ctys
https://github.com/ArnoCan/ctys

Commercial Support

Commercial support and additional services are available exclusively by direct contact only.

[154] Ingenieurbuero Arno-Can Uestuensoez:
https://arnocan.wordpress.com/
http://www.i4p.de
http://www.i4p.eu