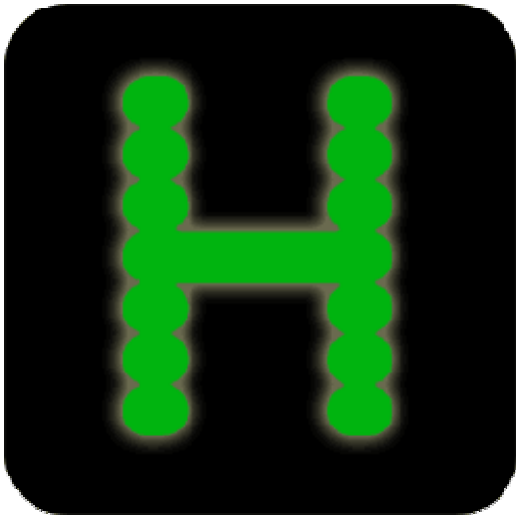
Hercules System/370, ESA/390, z/Architecture Emulator

Hercules – General Information

Version 3 Release 07



Contents

Contents 2
Figures 7
Tables 9
1. Preface 10
1.1 Edition information 10
1.2 What this book is about 10
1.3 Who should read this book 10
1.4 What you need to know to understand this book 10
1.5 How to use this book 10
1.6 Revision Notice 10
1.7 Readers Comments 11
1.8 Legal Advice 11
1.9 Trademarks 11
1.10 Acknowledgements 12
2. Related Publications 13
2.1 Hercules Emulator – General Information 13
2.2 Hercules Emulator – Installation Guide 13
2.3 Hercules Emulator – User Reference Guide 13
2.4 Hercules Emulator – Messages and Codes 13
2.5 Hercules Emulator – Reference Summary 13
3. Hercules Emulator Overview 14
3.1 Introduction 14
3.2 History of Hercules Development 17
3.3 Production use of the Hercules Emulator 19
3.4 Hercules Executables 19
3.5 Hercules Source Code 19
4. Implemented Features 23
4.1 Implemented architectural standard features 23
4.2 Implemented architectural optional features 23
4.3 Implemented optional features of z/Architecture 25
4.4 Not yet implemented optional z/Architecture features 26
4.5 Not yet implemented standard features 26
4.6 Partially implemented optional features 26
4.7 Not yet implemented features 26
4.8 Compliance 27
4.9 Related Products and Tools 27
5. Emulated Device Types 28
5.1 Local non-SNA 3270 Display or Printer 28
5.2 Console Printer-Keyboards 28
5.3 Card Readers 29
5.4 Card Punch 29
5.5 Line Printers 29
5.6 Tape Drives 30
5.7 Channel-to-Channel Adapters 33
5.8 FBA Direct Access Storage Devices 34
5.9 CKD Direct Access Storage Devices 34
5.10 Communication Lines 35
6. CCKD Compressed CKD DASD Devices 37
6.1 CCKD Introduction 37
6.2 CCKD Shadow Files 37
6.3 Compressed DASD File Structure 38
6.4 How it works 41
7. Shared Device Support 43
7.1 Usage of Shared Devices 43
7.2 Caching 44

7.3 Compression 44
7.4 Technical Approaches 45
7.5 Protocol 45
8. SCSI Tape Drives 50
8.1 Basics 50
8.2 Special Options 50
9. Hercules Dynamic Loader 52
9.1 Dependency Section 52
9.2 Registration Section 53
9.3 Resolver Section 53
9.4 Device Section 53
9.5 Final Section 53
10. ECPSVM – Extended Control Program Support VM/370 54
10.1 Technical Information 54
10.2 Troubleshooting 54
10.3 Implemented Assists 54
10.4 Non-Implemented Assists 55
10.5 Restrictions 56
11. Hercules Automated Operator (HAO) 57
11.1 Introduction 57
11.2 Defining Rules 57
11.3 Deleting Rules 57
11.4 Limitations 58
12. Summary of Hercules Changes 59
12.1 Hercules Evolution 59
12.2 Changes for Hercules Emulator Release 3.07.0 59
12.3 Changes for Hercules Emulator Release 3.06.0 60
12.4 Changes for Hercules Emulator Release 3.05.0 60
12.5 Changes for Hercules Emulator Release 3.04.1 61
12.6 Changes for Hercules Emulator Release 3.04.0 61
12.7 Changes for Hercules Emulator Release 3.03.1 62
12.8 Changes for Hercules Emulator Release 3.03.0 62
12.9 Changes for Hercules Emulator Release 3.02.0 62
12.10 Changes for Hercules Emulator Release 3.01.0 63
12.11 Changes for Hercules Emulator Release 3.00.0 63
12.12 Changes for Hercules Emulator Release 2.17.1 63
12.13 Changes for Hercules Emulator Release 2.17.0 64
12.14 Changes for Hercules Emulator Release 2.16.5 64
12.15 Changes for Hercules Emulator Release 2.16.4 64
12.16 Changes for Hercules Emulator Release 2.16.3 64
12.17 Changes for Hercules Emulator Release 2.16.2 65
12.18 Changes for Hercules Emulator Release 2.16.1 65
12.19 Changes for Hercules Emulator Release 2.16.0 65
12.20 Changes for Hercules Emulator Release 2.15.0 66
12.21 Changes for Hercules Emulator Release 2.14.0 66
12.22 Changes for Hercules Emulator Release 2.13.0 66
12.23 Changes for Hercules Emulator Release 2.12.0 67
12.24 Changes for Hercules Emulator Release 2.11.0 68
12.25 Changes for Hercules Emulator Release 2.10.0 68
12.26 Changes for Hercules Emulator Release 1.71.0 69
12.27 Changes for Hercules Emulator Release 1.70.0 69
12.28 Changes for Hercules Emulator Release 1.69.0 69
12.29 Changes for Hercules Emulator Release 1.68.0 70
12.30 Changes for Hercules Emulator Release 1.67.0 70
12.31 Changes for Hercules Emulator Release 1.66.0 70
12.32 Changes for Hercules Emulator Release 1.65.0 71
12.33 Changes for Hercules Emulator Release 1.64.0 71
12.34 Changes for Hercules Emulator Release 1.63.0 72

12.35 Changes for Hercules Emulator Release 1.62.0 72
12.36 Changes for Hercules Emulator Release 1.61.0 72
12.37 Changes for Hercules Emulator Release 1.60.0 73
12.38 Changes for Hercules Emulator Release 1.59.0 73
12.39 Changes for Hercules Emulator Release 1.58.0 73
12.40 Changes for Hercules Emulator Release 1.57.0 74
12.41 Changes for Hercules Emulator Release 1.56.0 75
12.42 Changes for Hercules Emulator Release 1.55.0 75
12.43 Changes for Hercules Emulator Release 1.54.0 75
12.44 Changes for Hercules Emulator Release 1.53.0 75
12.45 Changes for Hercules Emulator Release 1.52.0 76
12.46 Changes for Hercules Emulator Release 1.51.0 76
12.47 Changes for Hercules Emulator Release 1.50.0 76
12.48 Changes for Hercules Emulator Release 1.49.0 77
12.49 Changes for Hercules Emulator Release 1.48.0 77
12.50 Changes for Hercules Emulator Release 1.47.0 77
12.51 Changes for Hercules Emulator Release 1.46.0 78
12.52 Changes for Hercules Emulator Release 1.45.0 78
12.53 Changes for Hercules Emulator Release 1.44.0 78
12.54 Changes for Hercules Emulator Release 1.43.0 78
12.55 Changes for Hercules Emulator Release 1.42.0 79
12.56 Changes for Hercules Emulator Release 1.41.0 79
12.57 Changes for Hercules Emulator Release 1.40.0 80
12.58 Changes for Hercules Emulator Release 1.39.0 80
12.59 Changes for Hercules Emulator Release 1.38.0 80
12.60 Changes for Hercules Emulator Release 1.37.0 80
12.61 Changes for Hercules Emulator Release 1.36.0 81
12.62 Changes for Hercules Emulator Release 1.35.0 81
12.63 Changes for Hercules Emulator Release 1.34.0 81
12.64 Changes for Hercules Emulator Release 1.33.0 81
12.65 Changes for Hercules Emulator Release 1.32.0 82
13. Hercules Windows GUI 83
13.1 Introduction 83
13.2 Windows GUI Source Code 84
13.3 Hercules Windows GUI Evolution 84
13.4 Changes for Hercules Windows GUI Release 1.11.1 (Build 5265) 84
13.5 Changes for Hercules Windows GUI Release 1.11.0 (Build 5239) 84
13.6 Changes for Hercules Windows GUI Release 1.10.1 (Build 4909) 84
13.7 Changes for Hercules Windows GUI Release 1.10.0 (Build 4890) 85
13.8 Changes for Hercules Windows GUI Release 1.9.5 (Build 4734) 85
13.9 Changes for Hercules Windows GUI Release 1.8.15 (Build 4316) 86
13.10 Changes for Hercules Windows GUI Release 1.8.8 (Build 4207) 86
13.11 Changes for Hercules Windows GUI Release 1.8.6 (Build 4202) 86
13.12 Changes for Hercules Windows GUI Release 1.8.5 (Build 4200) 86
13.13 Changes for Hercules Windows GUI Release 1.6.8 (Build 3981) 86
13.14 Changes for Hercules Windows GUI Release 1.6.6 (Build 3910) 87
13.15 Changes for Hercules Windows GUI Release 1.6.4 (Build 3772) 87
13.16 Changes for Hercules Windows GUI Release 1.6.0 (Build 3438) 87
13.17 Changes for Hercules Windows GUI Release 1.5.0 (Build 3290) 88
13.18 Changes for Hercules Windows GUI Release 1.4.0 (Build 3164) 88
13.19 Changes for Hercules Windows GUI Release 1.3.0 (Build 2769) 88
13.20 Changes for Hercules Windows GUI Release 1.2.2 (Build 2573) 89
13.21 Changes for Hercules Windows GUI Release 1.1.3 (Build 2559) 89
14. Hercules Studio 90
14.1 Introduction 90
14.2 Hercules Studio Source Code 91
15. WinPcap Packet Capture Driver 92
15.1 WinPcap Packet Capture Driver Introduction 92

15.2 What does WinPcap 92
15.3 What kind of programs use WinPcap 93
15.4 What WinPcap cannot do 93
15.5 WinPcap Internals (Overview) 93
15.6 WinPcap Internals (Details) 94
15.7 NPF and NDIS 95
15.8 NPF Structure Basics 96
16. CTCI-W32 100
16.1 CTCI-W32 Introduction 100
16.2 CTCI-W32 Evolution 101
16.3 Changes for CTCI-W32 V3.2.1 (Build 160) 101
16.4 Changes for CTCI-W32 V3.2.0 (Build 156) 101
16.5 Changes for CTCI-W32 V3.1.7 101
16.6 Changes for CTCI-W32 V3.1.6 102
16.7 Changes for CTCI-W32 V3.1.2 102
16.8 Changes for CTCI-W32 V3.1.0 102
16.9 Changes for TunTap32 V2.1.0.404 102
16.10 Changes for FishPack V1.3.0.323 / TunTap32 V2.0.3.379 102
16.11 Changes for TunTap32 V2.0.0.367 103
16.12 Changes for FishPack V1.1.0.296 / TunTap32 V1.0.4.375 / tt32info V1.0.2.133 103
16.13 Changes for FishPack V1.0.2.288 / TunTap32 V1.0.2.360 / tt32info V1.0.2.131 103
16.14 Changes for FishPack V1.0.1.286 / TunTap32 V1.0.0.358 / tt32info V1.0.0.129 103
17. TN3270 Client 104
17.1 TN3270 Introduction 104
17.2 Vista tn3270 (Windows Platforms) 104
17.3 x3270 (Unix and Windows Platforms) 106
18. XMIT Manager 108
18.1 Introduction 108
18.2 Xmit File History 108
18.3 Common Use of Xmit Files 108
18.4 XMIT Manager Functionality 108
19. AWS Browse 111
19.1 Introduction 111
19.2 AWS Browse - Original Implementation 111
19.3 AWS Browse – Enhanced Implementation 111
19.4 AWS Browse Functionality 111
19.5 AWS Browse Evolution 112
19.6 Changes for AWS Browse V1.5.1 (Build 1805) 113
19.7 Changes for AWS Browse V1.5.0 (Build 1803) 113
19.8 Changes for AWS Browse V1.4.0 (Build 1483) 113
19.9 Changes for AWS Browse V1.3.0 (Build 1445) 114
19.10 Changes for AWS Browse V1.2.0 (Build 1278) 114
20. The MVS Turnkey System 115
20.1 What is the MVS Turnkey System 115
20.2 Installed FMIDs 115
20.3 PTFs for MVS 3.8J 117
20.4 Installed USERMODs for MVS 3.8J 117
21. Technical Support 119
21.1 Paid Support 119
21.2 Free Support 119
21.3 Forums 119
21.4 Hercules-390 120
21.5 H390-DOSVS 120
21.6 H390-MVS 120
21.7 Turnkey-MVS 121
21.8 H390-VM 121
21.9 Herc-4Pack 121
21.10 Hercules-390-Announce 122

21.11 Hercules-Advocacy....122
21.12 Hercules-S370ASM....122
22. Hercules Developers....123
22.1 Who are the Herculeans?....123
23. Hercules Users....128
23.1 What people are saying about Hercules....128
23.2 IBM Redbooks....129
24. OSI (Open Source Initiative)....130
24.1 Free Redistribution....130
24.2 Source Code....130
24.3 Derived Works....130
24.4 Integrity of the Author's Source Code....130
24.5 No Discrimination Against Persons or Groups....131
24.6 No Discrimination Against Fields of Endeavor....131
24.7 Distribution of License....131
24.8 License Must Not Be Specific to a Product....131
24.9 License Must Not Restrict Other Software....131
24.10 License Must Be Technology-Neutral....132
25. Q Public License, Version 1.0....133
25.1 Copyright Information....133
25.2 Introduction....133
25.3 Granted Rights....133
25.4 Limitations of Liability....134
25.5 No Warranty....134
25.6 Choice of Law....134
Appendix A. Links....135
Appendix B. Hercules Developer Photo Gallery....138
B1. Developer Workshop Enschede, December 2001....138
Developer Workshop Paris, September/October 2008....142

Figures

Figure 1: Hercules Hardware Console - Console window 15
Figure 2: Hercules Hardware Console - Device and status window 16
Figure 3: Hercules web browser interface 17
Figure 4: Hercules Lines of Code (LoC) 20
Figure 5: Hercules Number of Source Files 21
Figure 6: Hercules Average Source File Size 22
Figure 7: CCKD Device Header 38
Figure 8: CCKD Compressed Device Header 39
Figure 9: Primary Lookup Table 39
Figure 10: Secondary Lookup Table 40
Figure 11: Track or Block Group Image 40
Figure 12: CKD Header 40
Figure 13: FBA Header 40
Figure 14: Flags Byte 40
Figure 15: Compression Algorithm Bits 41
Figure 16: Free Space 41
Figure 17: Shared Device Server – Client Header Structure 46
Figure 18: Shared Device Server – Server Header Structure 48
Figure 19: Hercules Windows GUI Main Panel 83
Figure 20: Hercules Studio Main Panel 90
Figure 21: WinPcap Logo 92
Figure 22: WinPcap Architecture 94
Figure 23: NPF inside NDIS 96
Figure 24: NPF Device Driver 97
Figure 25: Packet Capture vs. Kernel-Level Dump 99
Figure 26: CTCI-W32 Implementation 100
Figure 27: Vista tn3270 Logo 104
Figure 28: Vista tn3270 Window 105
Figure 29: x3270 Logo 106
Figure 30: x3270 Window 107
Figure 31: XMIT Manager Logo 108
Figure 32: XMIT Manager Main Panel 109
Figure 33: XMIT Manager Member View 110
Figure 34: AWS Browse Main Window 112
Figure 35: MVS Turnkey System Logo 115
Figure 36: Hercules Author Roger Bowler 123
Figure 37: Hercules Developers, December 2001 124
Figure 38: Hercules Developers (Legend) 124
Figure 39: Greg Smith, Jay Maynard, Jan Jaeger and Fish 138
Figure 40: Willem Konynenberg and Fish 139
Figure 41: Fish and Greg Smith 140
Figure 42: Fish 141
Figure 43: Dave Wade, Jay Maynard, Volker Bandke, Bernard van der Helm, Roger Bowler 142
Figure 44: Dave, Jay, Iwan, Roger, Bernard and Volker (from left) 143

Figure 45: Roger Bowler and Iwan Warren 144
Figure 46: Volker Bandke and Jay Maynard 145
Figure 47: Dave Wade and Ivan Warren 146

Tables

Table 1: Local non-SNA 3270 Displays or Printers 28
Table 2: Console Printer Keyboards 28
Table 3: Card Readers 29
Table 4: Card Punch 29
Table 5: Line Printers 29
Table 6: Tape Drives 30
Table 7: Channel-to-Channel Adapters 33
Table 8: FBA Direct Access Storage Devices 34
Table 9: CKD Direct Access Storage Devices 34
Table 10: Communication Lines 35
Table 11: Shared Device Server – Client Request Codes 47
Table 12: Shared Device Server – Client Flags 48
Table 13: Shared Device Server – Server Response Codes 49
Table 14: MVS 3.8J FMIDs 117
Table 15: List of installed USERMODS for MVS 3.8J 118

1. Preface

1.1 Edition information

This edition applies to the Hercules S/370, ESA/390 and z/Architecture Emulator, Release 3.07.0, and to all subsequent versions, releases and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for your level of the software.

1.2 What this book is about

This book gives a general overview for the Hercules Emulator and related products. For guidance in operating or debugging Hercules or for the installation of the product there are additional manuals available. Please see Chapter “Related Publications” for more information on these manuals.

Please note that some information can be found in more than one manual. This redundancy is not intended to unnecessarily expand the manuals but should help to find all necessary information in one place.

1.3 Who should read this book

This book is mainly intended for people who are interested in a general introduction about the Hercules Emulator. It serves as a starting point for everything you need to know about Hercules.

1.4 What you need to know to understand this book

To understand this book you should be familiar with the Windows (XP, W2K) operating system. Some knowledge of TCP/IP configuration in a small network is required to understand the network connectivity.

Last but not least you should be familiar with IBM mainframe environments (hardware and software) and the underlying ideas and concepts as Hercules emulates IBM mainframe hardware.

1.5 How to use this book

This book is designed as an introduction to the Hercules Emulator and related products. If this is your first contact with the Hercules emulator you should go through the book chapter by chapter. If you are already familiar with Hercules you can pick up the chapters you are most interested in.

1.6 Revision Notice

Hercules Release:	Version 3 Release 07 Modification 0
Publication Number:	HEGI030700
SoftCopy Name:	HerculesGeneralInfo
Revision Number:	HEGI030700-01
Date:	June 28, 2010

1.7 Readers Comments

If you like or dislike anything about this book, please send an email to the email address below. Feel free to comment any errors or lack of clarity. Please limit your comments on the information in this specific book and also include the “Revision Notice” just above. Thank you for your help.

Send your comments by email to the Hercules-390 discussion group:

hercules-390@yahoogroups.com

1.8 Legal Advice

Hercules implements only the raw S/370, ESA/390, and z/Architecture instruction set, it does not provide any operating system facilities. This means that you need to provide an operating system or standalone program which Hercules can load from an emulated disk or tape device. You will have to write the operating system or standalone program yourself, unless you possess a license from IBM to run one of their operating systems on your PC, or use IBM programs and operating systems which have been placed in the public domain.

NOTE: It is YOUR responsibility to comply with the terms of the license for the operating system you intend to run on the Hercules Emulator.

1.9 Trademarks

The following is a list of trademark acknowledgments and copyright notices of product and company names mentioned in this book. Other product and company names in this book which are not listed below may be the trademarks or registered trademarks of their respective owners.

- IBM, System/370, ESA/390, z/Architecture, MVS, OS/390, z/OS, VM, VM/ESA, z/VM, VSE, VSE/ESA, z/VSE are trademarks or registered trademarks of International Business Machines Corporation (IBM).
- Windows 95, Windows 98, Windows ME, Windows NT, Windows 2000, Windows XP, Windows Vista, Windows 7, Windows Server 2000, Windows Server 2003, Windows Server 2008, Visual C++ Toolkit 2003, Visual C++ 2005 Express are trademarks of Microsoft Corporation.
- Linux is a trademark owned by Linus Torvalds. The Linux Mark Institute is the exclusive licensor of the Linux trademark on behalf of its owner Linus Torvalds.
- WinPcap is copyrighted by NetGroup, Politecnico di Torino (Italy).
- Cygwin is copyrighted by Red Hat, Inc.
- Vista tn3270 is copyrighted by Tom Brennan Software.
- Pentium, XEON are trademarks or registered trademarks of Intel Corporation.
- Athlon, Opteron are trademarks or registered trademarks of Advanced Micro Devices (AMD), Inc.
- Xmit Manager is copyrighted by Neal Johnston-Ward.
- FLEX-ES is a registered trademark of Fundamental Software, Inc.
- UMX Virtual Mainframe is a registered trademark of UMX Technologies.

1.10 Acknowledgements

The Hercules manuals would not have been possible without the assistance of many people and I would like to thank all those who helped me. In particular I would like to thank:

- The Hercules developers for their documentation on various websites from which I derived a great deal of information.
- Roger Bowler and Fish for proof-reading the manuals.
- Loris Degoianni for allowing me to use parts of the original WinPcap documentation.
- Tom Brennan for allowing me to use parts of his Vista tn3270 documentation.
- My colleagues for working with early previews of the documentation, beginning with just a few pages.
- Mike Cairns for reviewing and editing the manuals.

If anyone feels they have been forgotten on this list please let me know.

Peter Glanzmann

2. Related Publications

2.1 Hercules Emulator – General Information

The Hercules “General Information” manual provides an overview of the ideas and concepts of the Hercules Emulator as well as documentation of the emulators functionality. It explains what Hercules does and does not and helps you decide if the software fits to your needs and if it can fulfill all your requirements.

2.2 Hercules Emulator – Installation Guide

The Hercules "Installation Guide" shows you how to install Hercules and all related optional and required software components under the Microsoft Windows, Linux and Apple MacIntosh OS X operating systems.

After going through the installation guide you will have a working emulator environment ready to IPL a S370, S/390 or z/Architecture mainframe operating system.

2.3 Hercules Emulator – User Reference Guide

The Hercules “User Reference” leads you through all aspects of the emulator’s operation. It provides instruction in the operation of the Hercules Emulator with and without the Windows GUI. The usage details for all Hercules utilities are also covered in this guide.

After reading this manual you should be able to work with Hercules and the Hercules console, create virtual devices, understand backup / restore procedures and general housekeeping within the Hercules environment.

2.4 Hercules Emulator – Messages and Codes

The “Messages and Codes” manual gives you a detailed explanation of all Hercules related messages. It is the primary source for troubleshooting and debugging if you experience problems running Hercules.

2.5 Hercules Emulator – Reference Summary

The Hercules "Reference Summary" booklet lists all the system parameters, device definitions, console commands, Hercules utilities etc. along with their arguments.

This booklet is intended as a quick reference guide for experienced users. Consult the Hercules "User Reference Guide" for more detailed and additional information.

3. Hercules Emulator Overview

3.1 Introduction

Hercules is an open source software implementation of the mainframe System/370, ESA/390 and z/Architecture hardware. The Hercules emulator runs under Linux on several hardware platforms including the Intel Pentium PC, under Windows 98, Windows NT, Windows 2000, Windows XP, Windows Vista, Windows 7, Windows Server 2003, Windows Server 2008, Solaris, FreeBSD and under MAC OS X 10.3 and later.

Hercules was created by Roger Bowler and is currently maintained by Jay Maynard. Jan Jaeger designed and implemented many of the advanced features of Hercules, including dynamic reconfiguration, integrated console, interpretive execution and z/Architecture support. For a complete list of the Hercules developers see chapter 5 (“Who are the Herculeans?”).

With the Hercules Emulator, your PC can emulate an IBM mainframe processor. The mainframe can range from a 360 to a z10 – running in System/370 mode, ESA/390 mode or z/Architecture mode. Hercules executes S/370, ESA/390 and z/Architecture instructions and channel programs. It emulates mainframe I/O devices by using PC devices. As an example, 3390 DASD devices are emulated by large files on the hard disk of the PC and local 3270 screens are emulated by tn3270 sessions. Please note that not all 370 and 390 features have been implemented. Details about implemented, partially implemented and not implemented features can be found later in this chapter.

Hercules only implements the raw S/370, ESA/390 and z/Architecture instruction sets, it does not provide any operating system facilities. This means that you need to provide an operating system or standalone program which Hercules can load from an emulated disk or tape device. You will have to write the operating system or standalone program by yourself unless you have a license from IBM to run one of their operating systems on your PC. You may also use IBM programs and operating systems which have been placed in the public domain.

The following figures show the main panels of the Hercules Emulator Hardware Management Console (HMC) and the Hercules web browser interface.

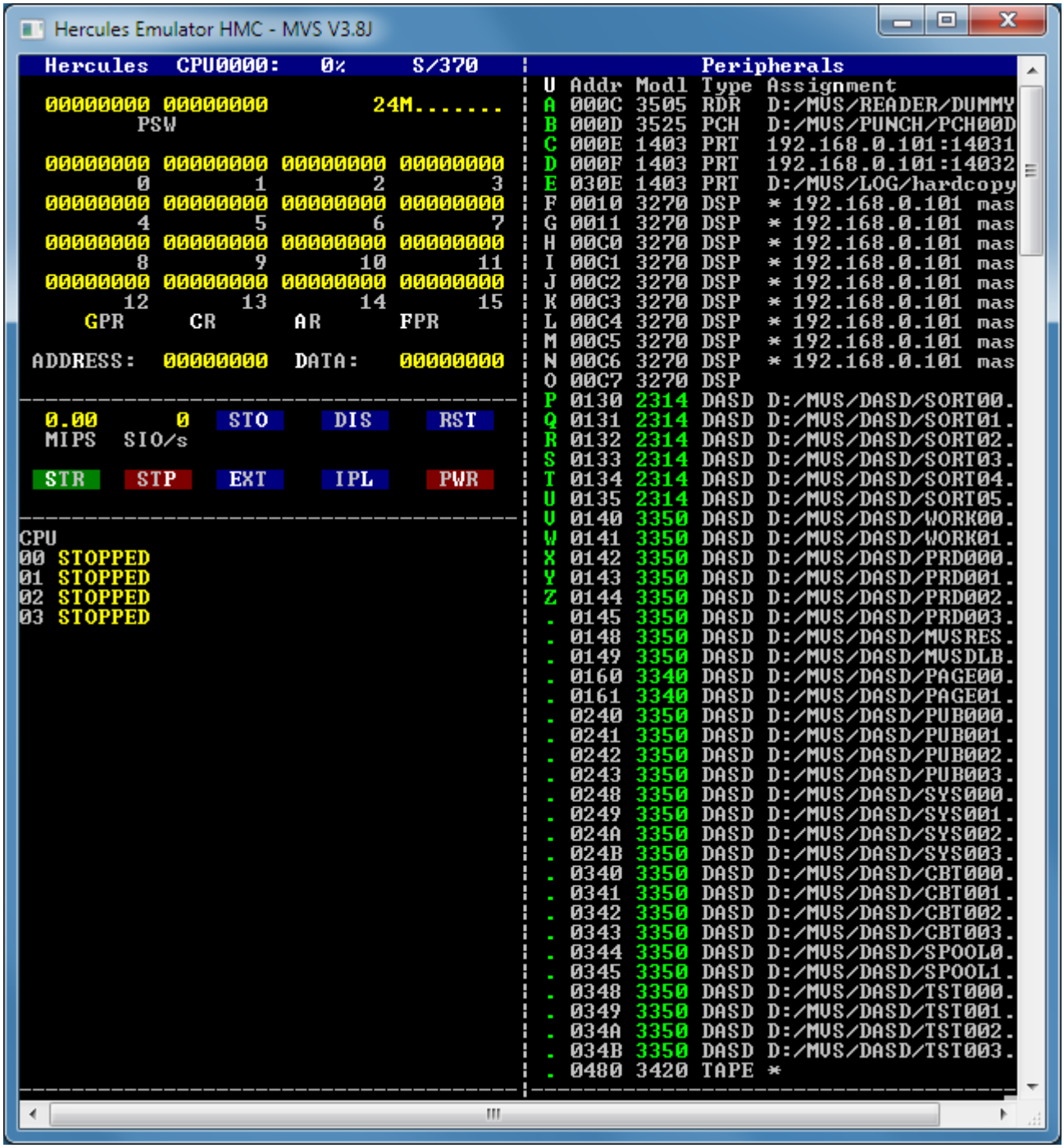


Figure 1: Hercules Hardware Console - Console window

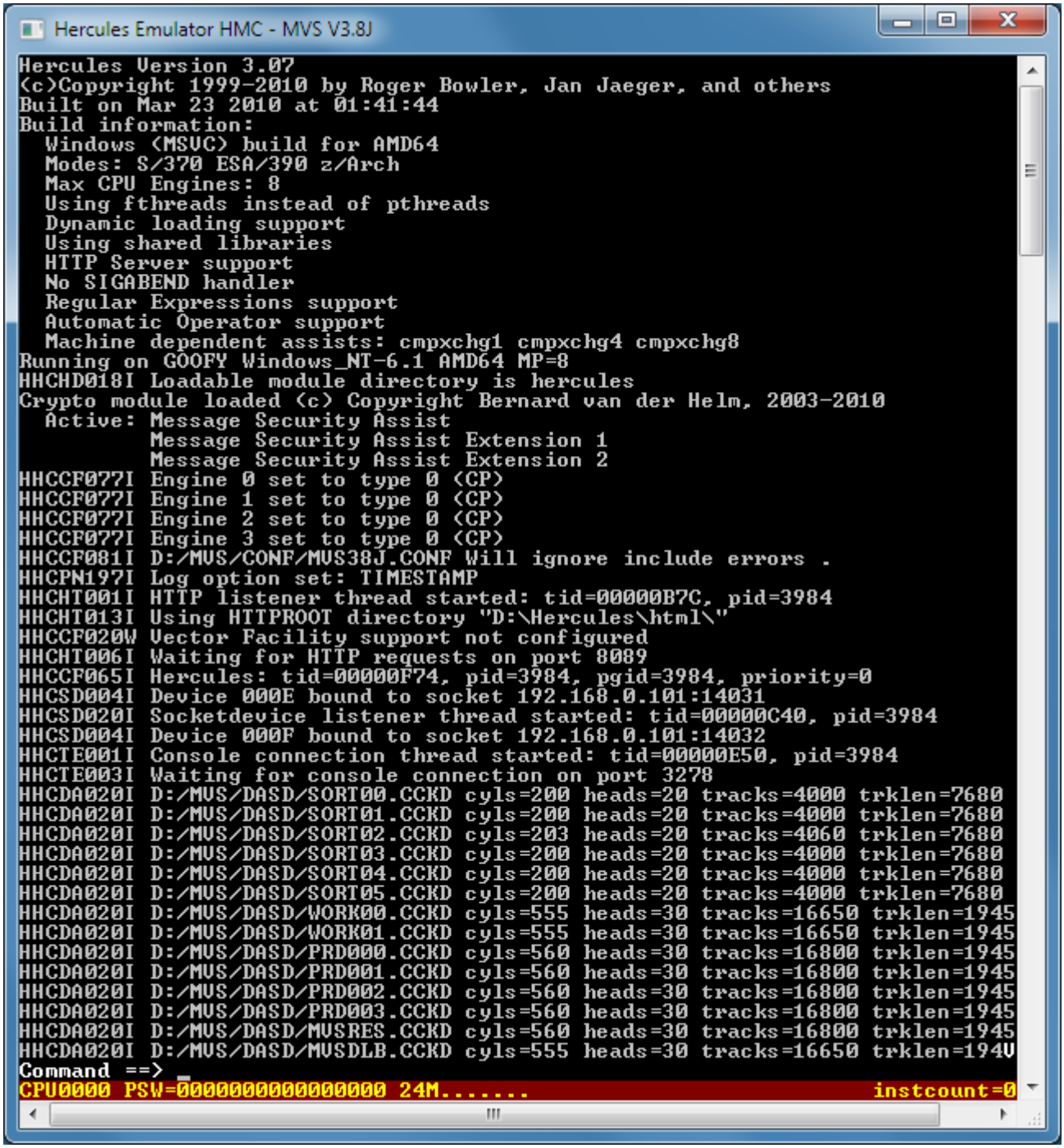


Figure 2: Hercules Hardware Console - Device and status window

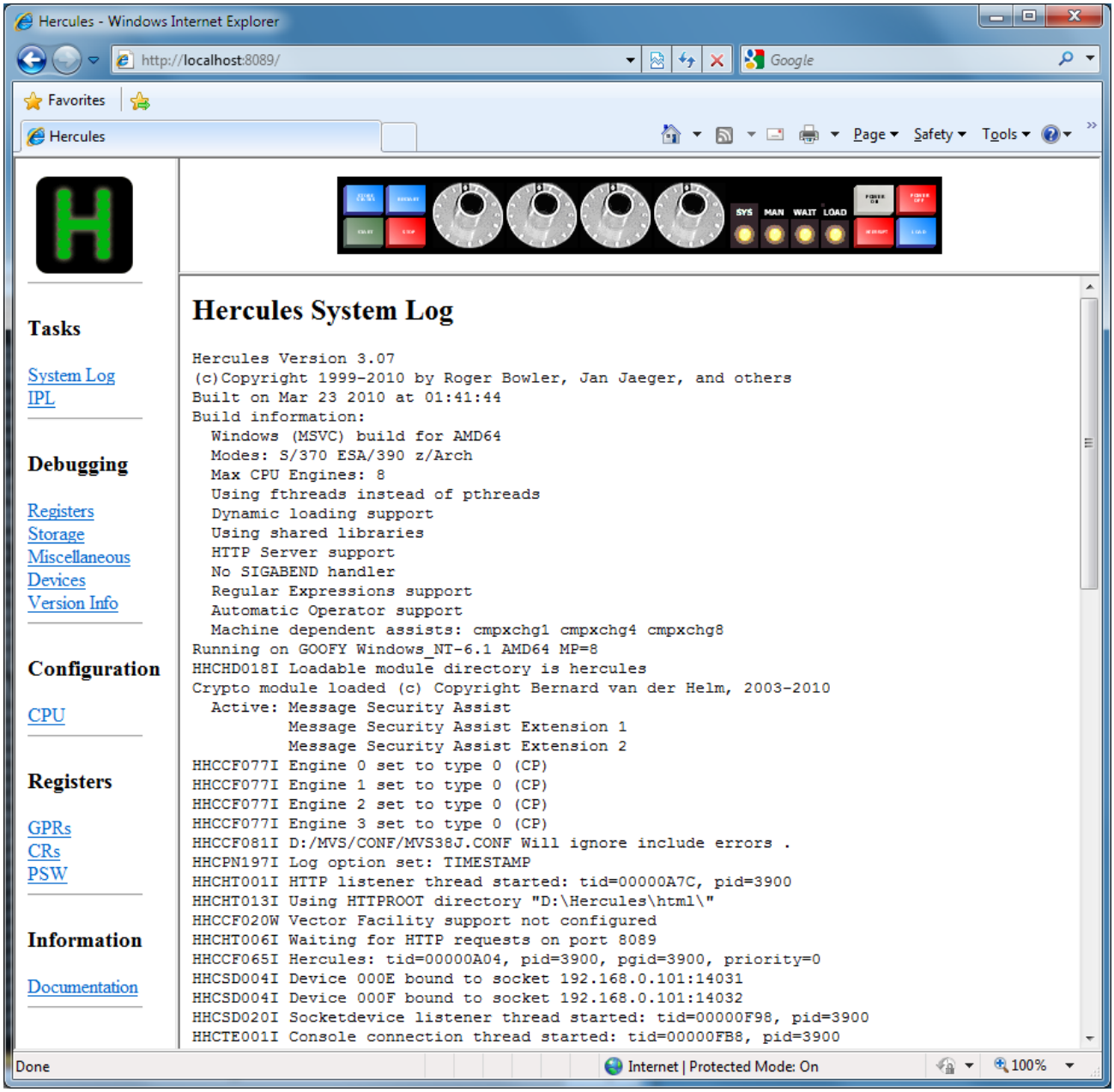


Figure 3: Hercules web browser interface

Hercules is OSI (Open Source Initiative) certified open source software and is released under the Q Public License. For details on OSI, see Chapter 24, for more information on the Q Public License please refer to Chapter 25.

3.2 History of Hercules Development

Ever since Roger Bowler saw his first mainframe (a 360/30 at Charter Consolidated in Ashford, Kent, in about 1970) it has been his dream to own and operate a “real” computer.

Thirty years later he still had no personal mainframe so he created Hercules instead. It was a little toy that turned his Pentium PC into an IBM mainframe, almost.

Although the genesis of Hercules dates back to 1994 most of the work was done during a nine month period in 1999 while Roger Bowler was between contracts. By the autumn of that year he had implemented enough of the S/360 and ESA/390 architecture to be able to IPL and run OS/360 (MFT) and Jan Jaeger's ZZSA standalone program in ESA mode. Following this he gradually added the missing parts of the architecture, so that by the start of the year 2000 Hercules was (according to reports from IBMers) quite capable of running VSE/ESA and could even IPL OS/390 (albeit somewhat slowly!)

During spring 2000, others began to make significant contributions to the work:

- Dutch Owen created a neat system activity display, giving the console a more authentic mainframe look.
- Jay Maynard added S/370 mode virtual storage capability, making it possible to run MVS/370.
- Jan Jaeger added HMC, dynamic CHPID and CPU reconfiguration and the Interpretive Execution Facility (SIE) which allows Hercules to run VM/ESA.
- Peter Kuschnerus added the hexadecimal floating point instructions.
- Several researchers, including Valery Pogonchenko, Juergen Dobrinsky and Clem Clarke, doggedly experimented with optimization of the instruction interpretation (CPU) critical path, resulting in a quantum leap in performance.
- John Kozak was the first to successfully port Hercules to run under Windows, a milestone which brought Hercules to a significantly broader audience.
- Fish (David B. Trout) created the massively popular graphical front-end for Hercules under Windows.
- Volker Bandke created the phenomenally popular MVS Turnkey CD. MVS up and running on your PC in 15 Minutes.
- Willem Konynenberg added 3088-TCP/IP and binary floating point instructions, essential for people wanting to run Linux/390.
- Matt Zimmerman structured Hercules into a Debian package.
- Greg Smith added support for compressed DASD, and more recently for shared DASD which allows multiple instances of Hercules to be formed into a loosely coupled multiprocessor complex.
- Paul Leisy identified and corrected numerous subtle bugs in the architectural implementation.

In May 2000 the need to get a job meant that Roger Bowler could no longer continue full-time work on Hercules and he handed over control of the project to Jay Maynard, who continues to be the official maintainer and champion of Hercules. Jay's regular Hercules presentations at SHARE conferences attract large audiences and his session at SHARE 99 in San Francisco received a "Best Session Award" for a session in the MVS program.

In autumn 2000, IBM announced a new 64-bit z/Architecture (also known as ESAME or ESA Modal Extension). Using publicly available information together with his deep knowledge of the evolution of the S/360, S/370 and S/390 architecture, Jan Jaeger was able to predict the likely form that the 64-bit architecture extensions would take. This enabled him to design preliminary support for the new architecture and to implement many of the new instructions in advance of the publication of the full technical details in January 2001. During some busy weekends which followed, Roger Bowler added support in Hercules for 64-bit mode IDAW, Cross Memory and DAT, with the result that at the end of February 2001 only five weeks after publication of the z/Architecture principles of Operation manual, Hercules became the first (and for 18 months the only) non-IBM implementation of the new 64-bit mainframe architecture!

A series of performance enhancements by Greg Smith, Gabor Hoffer, Juergen Dobrinski and Paul Leisy during the period 2002 to 2004, coupled with a significant increase in the power of entry-level processors, brought Hercules up to a respectable MIPS level of around 25 mainframe MIPS an a 3 GHz Pentium CPU

in 2005. During the same period Hercules also kept pace with the evolution of the mainframe architecture, adding support for new features introduced by the latest IBM z990 and z9 processors.

In October 2005 the Hercules project was honored by NaSPA (www.naspa.com), the "Network and Systems Professionals Association", with three Herculeans (Roger Bowler, Jay Maynard and Volker Bandke) sharing the NaSPA 2005 Award for Technical Excellence.

Hercules is now by far the world's most popular S/390 emulator, with the number of installed system estimated at over 6000. That is more than ten times the combined total of commercial S/390 emulators installed world-wide! Looked at from another perpective, it means that around 20% of the mainframes in the world are now Hercules Emulators! And what is more, Hercules is the most widely-used S/390 emulator within IBM itself, where many IBM systems engineers and marketing representatives use Hercules as an inexpensive and readily available platform for testing and demonstration purposes.

3.3 Production use of the Hercules Emulator

Hercules is not intended to run production work. At the moment it is a "system programmer's toy" although it is rapidly becoming good enough to run a wide range of software with no reported problems.

If you need a production environment you should consider something like a FLEX-ES or an UMX system.

3.4 Hercules Executables

The executables as well as the source code for the current release of Hercules can be downloaded from the following site:

www.hercules-390.org

If you are interested in obtaining the latest snapshot builds (actual SVN version) of Hercules, then visit:

http://www.ivansoftware.com/snapshots/snapshots

3.5 Hercules Source Code

The complete source code for the current development version of Hercules is also available from the "Subversion" (SVN) repository. Subversion is an open source version control system. The previous repository for Hercules (Concurrent Versions System - CVS) has been replaced with Subversion.

To get the current Hercules source from the repository, you would do:

svn checkout svn://svn.hercules-390.org/hercules/trunk hercules

This will get the main trunk of the source tree to a directory named "hercules" beneath your current directory. No password is required. For more information on Subversion, go to http://subversion.tigris.org/.

Please note that this will get you the current development version of Hercules, which is not release quality and thus might not even work (since it's still under development). If you want the current, stable, release version of Hercules (i.e. one that is known to work properly) use the previously mentioned links instead.

If you are interested to follow the Hercules development closely, Ivan Warren has set up a SVN browsing service at:

http://www.ivansoftware.com/cgi-bin/viewvc.cgi/

3.5.1 Lines of Code (LoC)

The are regular statistics about the source code. The statistics show, that the total number of lines of code has increased, beginning with the measurements around March 2001, from about 100,000 lines to nearly 325,000 in January 2009.

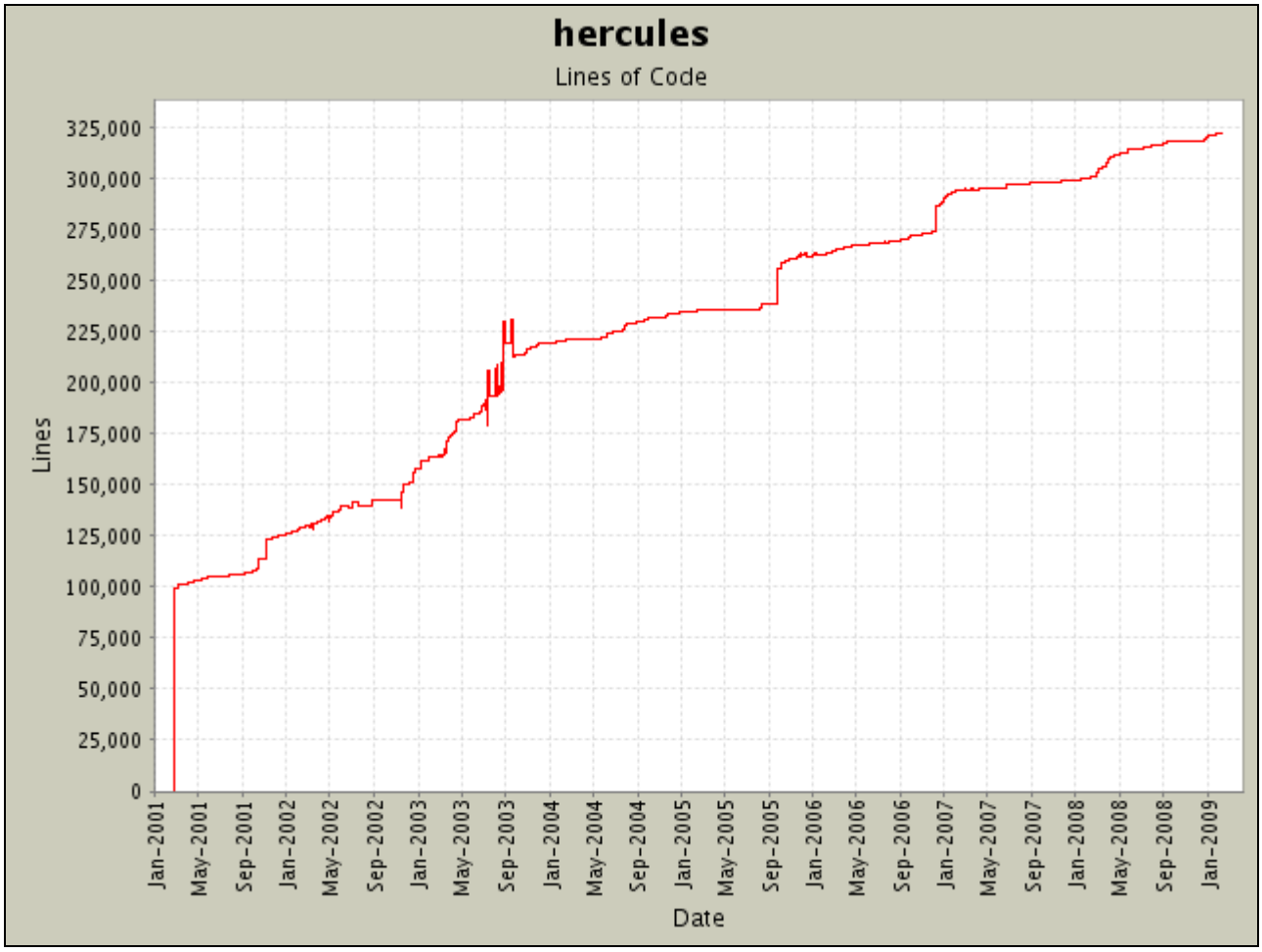


Figure 4: Hercules Lines of Code (LoC)

3.5.2 Number of Source Files

A similar evolution happened with the number of source files necessary to build the Hercules Emulator. The total number of files has increased, beginning with the measurements around March 2001, from a little bit more than 100 files to 475 files currently.

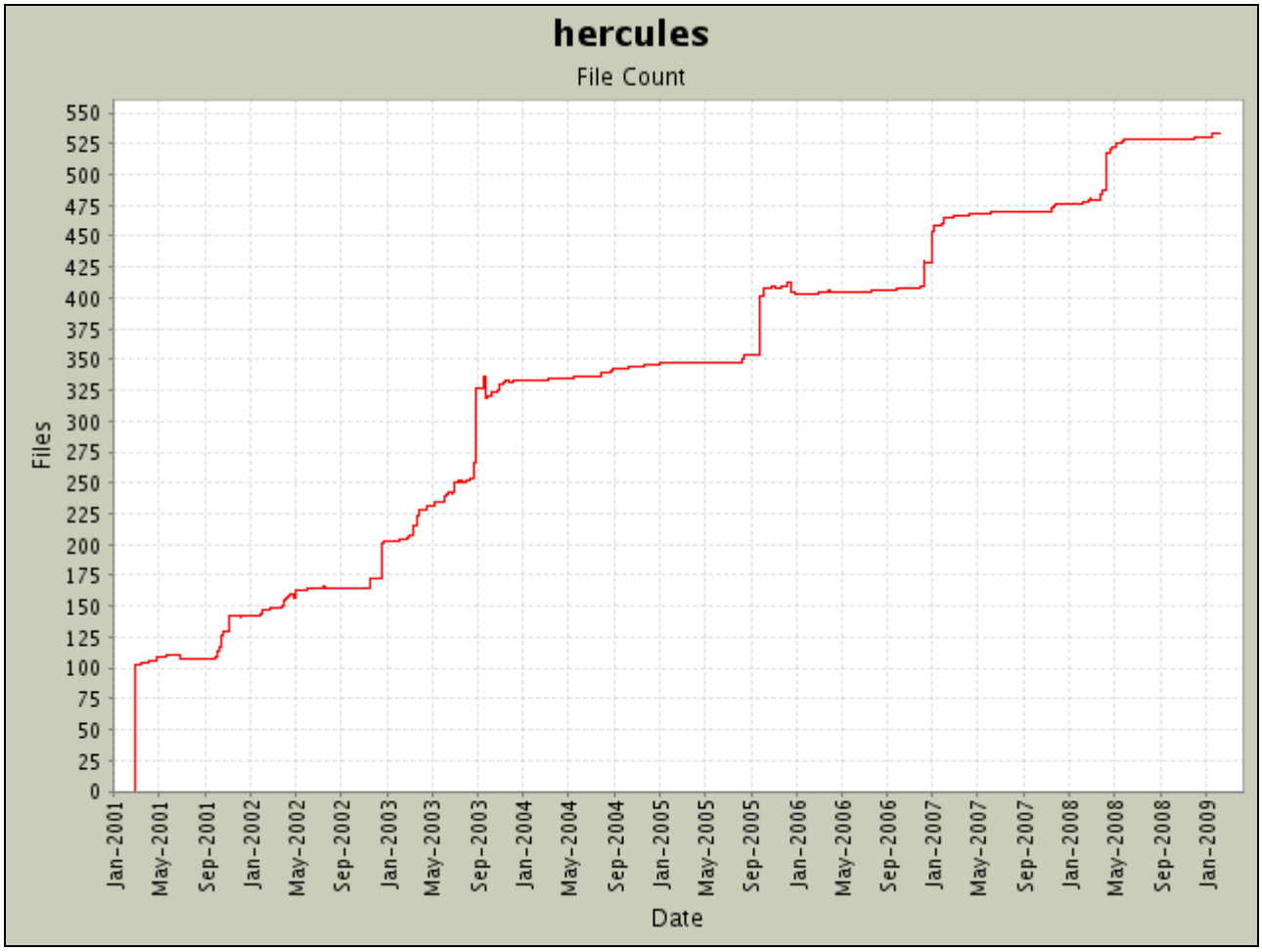


Figure 5: Hercules Number of Source Files

3.5.3 Average Source File Size

The opposite to the number of files happened with the number of lines of code per source file. The average file size in lines of code has decreased, also beginning with the measurements around March 2001, from nearly 1000 lines to less than 650 lines of code per file currently.

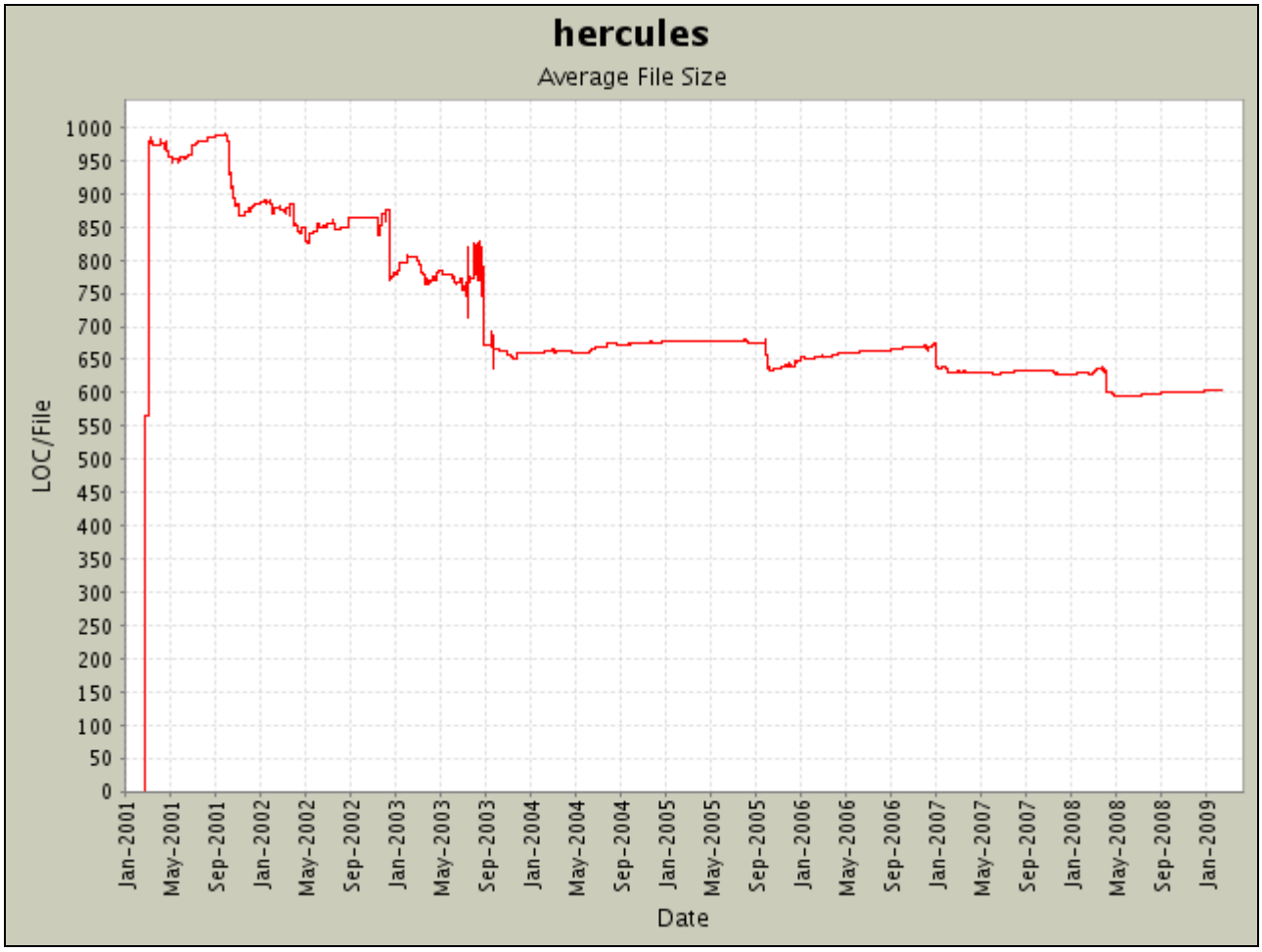


Figure 6: Hercules Average Source File Size

4. Implemented Features

4.1 Implemented architectural standard features

The following IBM mainframe standard architectural features have been implemented:

- Address-Limit Checking
- Commercial Instruction Set
- Decimal Instructions
- Hexadecimal Floating-Point Instructions
- 24-bit and 31-bit Addressing
- Key-Controlled Protection
- Page Protection
- Low-Address Protection
- Dynamic Address Translation
- 370-XA Channel Subsystem
- Channel Indirect Data Addressing
- Program Controlled Interruption (PCI)
- Channel Program Suspend / Resume
- Dual Address Space
- Access Register Mode
- Home Space Mode
- Branch and Save
- Conditional Swapping
- TOD Clock, Clock Comparator and CPU Timer
- MVCS / MVCP / MVCK / MVCSK / MVCDK Instructions
- TB / TPROT Instructions
- LURA / STURA Instructions
- BAKR / PC / PR / PT Instructions
- Linkage Stack
- Compare and Form Codeword and Update Tree Instructions

4.2 Implemented architectural optional features

The following IBM mainframe optional architectural features have been implemented:

- Access-List-Controlled Protection
- Binary Floating Point Instructions
- Branch and Set Authority

- Broadcasted Purging
- Checksum Instruction
- Compare and Move Extended Instructions
- Dynamic Reconfiguration
- Expanded Storage
- Fast Synchronous Data Mover Facility
- Floating-Point-Support Extensions
- Halfword-Immediate Instructions
- Branch-Relative Instructions
- Incorrect-Length-Indication Suppression
- Interpretive Execution (SIE)
- Move Inverse
- Move Page (Facility 2)
- MVS Assists
- Operational Extensions: Console Integration
- Private Space
- Set Address Space Control Fast
- Service-Call-Logical-Processor (SCLP) Facility
- Square Root
- Storage-Protection Override
- Storage Key Assist
- String Instructions
- Subspace Group
- Compare Until Substring Equal
- Concurrent Sense
- Suppression on Protection with Virtual-Address Enhancement
- Extended TOD Clock
- Compression
- Perform Locked Operation
- Vector Facility
- Multiple Controlled Data Space (VM Dataspaces)
- Extended Translation
- Extended Translation Facility 2
- Store System Information
- Cancel I/O Facility
- Program Event Recording
- Guest PER Enhancement

4.3 Implemented optional features of z/Architecture

- HFP Multiply-and-Add/Subtract Facility
- Message Security Assist
- Long-Displacement Facility
- DAT-Enhancement Facility
- Extended Translation Facility 3
- ASN-and-LX-Reuse Facility
- List-Directed Initial Program Load
- Modified CCW Indirect Data Addressing (MIDAW) Facility
- Extended-Immediate Facility
- Message-Security-Assist Extension 1
- Message-Security-Assist Extension 2
- DAT-Enhancement Facility 2
- Store-Clock-Fast Facility
- Store-Facility-List-Extended Facility
- ETF2-Enhancement Facility
- ETF3-Enhancement Facility
- PER-3 Facility
- TOD-Clock-Steering Facility
- Conditional-Emergency-Signal and Sense-Running-Status Facility
- Multiple Logical Channel Subsystems Facility
- Floating-Point-Support Enhancement Facility (FPR-GR-Loading, FPS-Sign-Handling and DFP-Rounding)
- Decimal Floating Point Facility
- IEEE-Exception-Simulation Facility
- Extract-CPU-Time Facility
- Conditional-SSKE Facility
- Compare-and-Swap-and-Store Facility
- Execute-Extensions Facility
- General-Instructions-Extension Facility
- Move-with-Optional-Specifications Facility
- Parsing-Enhancement Facility
- Compare-and-Swap-and-Store Facility 2
- Integrated 3270 (SYSG) Console
- Configuration-Topology Facility
- HFP-Unnormalized-Extensions Facility

4.4 Not yet implemented optional z/Architecture features

The following optional z/Architecture features have not yet been implemented:

- Enhanced-DAT Facility
- PFPO Facility
- Restore Subchannel Facility
- Integrated ASCII (SYSA) Console
- Set Program-Parameter Facility
- CPU-Measurement Counter Facility
- CPU-Measurement Sampling Facility

4.5 Not yet implemented standard features

The following standard architectural feature has not yet been implemented:

- Clear I/O (Full functionality for S/370)

4.6 Partially implemented optional features

The following optional architectural features have been partially implemented:

- Channel-Subsystem Call
- VM/370 Assists

4.7 Not yet implemented features

The following architectural features have not yet been implemented, either due to lack of IBM documentation, limited host system capability or lack of supporting hardware:

- Asynchronous Data Mover Facility
- Asynchronous Pageout Facility
- Coupling Links
- ESCON
- FICON
- MIF (Multiple Image Facility)
- Extended Sorting
- External Time Reference (Sysplex Timer)
- ICRF (Cryptography)
- Operational Extensions: Automatic Reconfiguration, Storage Reconfiguration, SCP-initiated Reset, Processor Availability
- PR/SM
- Program-Controlled re-IPL

4.8 Compliance

Hercules is compliant with IBM's ALS-1, ALS-2 and ALS-3 architectural level sets to the degree necessary to run all OS/390 versions through OS/390 V 2.10, all known versions of z/OS in both ARCHLVL 1 and ARCHLVL 2 modes and Linux and z/VM in both ESA/390 and z/Architecture mode.

4.9 Related Products and Tools

The Hercules Emulator itself relies on several optional and mandatory products and tools that are described later in this book. These products and tools are:

- Hercules Windows GUI (optional – Windows only)
- WinPcap Packet Capture Driver (optional, mandatory for TCP/IP functionality – Windows only)
- CTCI-W32 (optional, mandatory for TCP/IP functionality – Windows only)
- TN3270 Client (mandatory – all platforms)
- XMIT Manager (optional – Windows only)
- AWS Browse (optional – Windows only)
- The MVS Turnkey System

Although there are some software packages that are optional, their use is highly recommended. Some are necessary for full implementation of certain functionality (especially TCP/IP), others help in the daily use of Hercules itself (Windows GUI, AWS Browse) or for exchanging data with the Hercules world (XMIT Manager).

The MVS Turnkey System is a package of freely available parts of MVS 3.8J and lets you build up a fully functional MVS system in less than 15 minutes!

5. Emulated Device Types

Hercules emulates various hardware devices. The following sections and tables list and describe all supported device types in detail.

5.1 Local non-SNA 3270 Display or Printer

Device Type	Emulated by
IBM 3270	tn3270 Client Connection
IBM 3278	tn3270 Client Connection

Table 1: Local non-SNA 3270 Displays or Printers

To use this device a tn3270 client must connect to the host machine via the port number specified on the CNSLPORT statement (see “Hercules User Reference Guide”). A valid tn3270 device type such as IBM-3278 must be used. If the tn3270 client software supports it a device-type suffix can be used to connect to a specific device number.

5.2 Console Printer-Keyboards

Device Type	Emulated by
IBM 1052	Telnet Client Connection
IBM 3215	Telnet Client Connection
IBM 1052-C	Integrated on Hercules Console
IBM 3215-C	Integrated on Hercules Console

Table 2: Console Printer Keyboards

To use this device a tn3270 client must connect to the host machine via the port number specified on the CNSLPORT statement (see “Hercules User Reference Guide”).

5.3 Card Readers

Device Type	Emulated by
IBM 1442	Disk File(s) – ASCII or EBCDIC
IBM 2501	Disk File(s) – ASCII or EBCDIC
IBM 3505	Disk File(s) – ASCII or EBCDIC

Table 3: Card Readers

When defining this type of device the argument can be used to specify a list of file names containing card images (see “Hercules User Reference Guide”).

5.4 Card Punch

Device Type	Emulated by
IBM 3525	Disk File – ASCII or EBCDIC

Table 4: Card Punch

When defining this type of device the argument is used to specify the name of a file to which the punched output will be written (see “Hercules User Reference Guide”).

5.5 Line Printers

Device Type	Emulated by
IBM 1403	Disk File or Socket Device – ASCII
IBM 3211	Disk File or Socket Device – ASCII

Table 5: Line Printers

When defining this type of device the argument is used to specify the name of a file to which the printer output will be written (see “Hercules User Reference Guide”) or a socket device in the form "host:port". The output is written in the form of variable length lines of ASCII characters delimited by line feed or by carriage return/line feed sequences. Trailing blanks are removed from each line. Carriage control characters are translated to blank lines or ASCII form feed characters.

5.6 Tape Drives

Device Type	Emulated by
IBM 3410	Disk File, CD-ROM or SCSI Tape
IBM 3420	Disk File, CD-ROM or SCSI Tape
IBM 3480	Disk File, CD-ROM or SCSI Tape
IBM 3490	Disk File, CD-ROM or SCSI Tape
IBM 9347	Disk File, CD-ROM or SCSI Tape
IBM 8809	Disk File, CD-ROM or SCSI Tape
IBM 3422	Disk File, CD-ROM or SCSI Tape
IBM 3430	Disk File, CD-ROM or SCSI Tape

Table 6: Tape Drives

The following five types of emulation are supported:

- SCSI Tapes
- Optical Media Attach (OMA) Virtual Files
- AWSTAPE Virtual Files
- Fake Tape Virtual Files
- HET Virtual Tapes

5.6.1 SCSI Tapes

The argument specifies the tape device name (usually “/dev/nst0” for Linux or Windows or "\\.\Tape0" for Windows). The "0" in the name means tape drive number 0, the first or only host-attached SCSI tape drive, "1" means the second host-attached SCSI tape drive and so on. SCSI tapes are read and written using variable length EBCDIC blocks and filemarks exactly like a mainframe tape volume.

5.6.2 Optical Media Attach (OMA) Virtual Files

These are read-only files which usually reside on CD-ROM. OMA virtual tapes consist of one CD-ROM file corresponding to each physical file of the emulated tape. An ASCII test file, called the tape descriptor file (TDF), specifies the names of the files which make up the virtual tape. The argument specifies the name of the tape descriptor file (for example “/cdrom/tapes/uaa196.tdf”).

5.6.3 AWSTAPE Virtual Files

An AWSTAPE file contains a complete tape in one physical file. AWSTAPE files consist of variable length EBCDIC blocks. Each block is preceded by a 6-byte header. Filemarks are represented by a 6-byte header with no data. This is the same format as is used by IBM’s P/390 machines.

The argument specifies the location of the file (for example “ICKDSF.IPL”). The suffix of the file can be anything, though it is recommended to use “.AWS” for clarity.

5.6.4 Fake Tape Virtual Files

Fake Tape virtual files contain a complete tape in one file. FakeTape files consist of variable length EBCDIC blocks. Each block is preceded by a 12-byte ASCII-hex-character header. Filemarks are represented by a 12-byte character header with no data. The FakeTape format is used by the Flex-ES system from Fundamental Software Inc (FSI). The argument specifies the location of the FakeTape file (for example “`ickdsf.fkt`”). "FLEX-ES" and "FakeTape" are trademarks of Fundamental Software, Inc.

5.6.5 HET Virtual Files

A HET file also contains a complete tape in one physical file and has the same structure as the AWS-TAPE format with the added ability to have compressed data. The first argument specifies the location of the file. The filename must end with the suffix “.HET” to be recognized by Hercules as a HET file (for example “023178.HET”).

HET files can be compressed with two compression methods, ZLIB and BZIP2. The level of compression can also be controlled from 1 (for fastest compression) to 9 (for best compression). Default for compression is 4, which is a compromise between speed and compression rate.

5.6.6 Basic ACF Support

The ACF (Automatic Cartridge Feeder) is a feature on cartridge type tape drives (3480, 3490, etc.) that automatically loads a new tape when a tape is removed (ejected) from the drive. There is no real control over this functionality by the host as the device just keeps on feeding tapes one after the other.

Although the ACF features is unique to catridge type tape systems the emulation accepts to use the same technique for emulated ½ inch tapes reel drives as well. ACF is supported through a file that contains a list of files (emulated tapes or catridges) that will be loaded one after the other.

To manually reset the ACF to the top of the stack the DEVINIT command (see "Hercules User Reference Guide" for details) can be used to "reload" the ACF feature.

5.6.7 VTAPE Automount Support

Starting with Hercules version 3.06 a new AUTOMOUNT option is available that allows guest operating systems to directly mount, unmount and query tape device filenames for themselves, without any intervention on the part of the Hercules operator. Automount support is enabled via the AUTOMOUNT configuration file system parameter.

An example guest automount program for VSE called "TMOUNT" is provided in the util subdirectory of the Hercules source code distribution.

Briefly, the 0x4B (Set Diagnose) CCW is used to mount or unmount a file onto a tape drive, and the 0xE4 (Sense ID) CCW opcode is used to query the name of the currently mounted file.

For mounts, the 0x4B CCW specifies the filename of the file to be mounted onto the drive. The file must reside in the specified AUTOMOUNT directory or the automount request will be rejected. To unmount the currently mounted file, simply do a mount of the special filename "OFFLINE".

To query the name of the currently mounted file, the 0xE4 CCW is used. Note however that the 0xE4 (Sense ID) CCW opcode cannot be used by itself since the drive may also already natively support the Sense ID CCW opcode. Instead, it must be preceded by (command-chained from) a 0x4B CCW with a

data transfer length of one byte. The following 0xE4 command is the one that then specifies the I/O buffer and buffer length of where the query function is to place the device's currently mounted host filename.

In summary:

MOUNT: X'4B', <filename>, X'20', <length>

UNMOUNT: same thing but use filename "OFFLINE" instead

QUERY: X'4B', <buffer>, X'60', 1

X'E4', <buffer>, X'20', <buffersize>

Please refer to the previously mentioned “TMOUNT” sample for more information.

5.6.8 Multivolume Support, EOT, Tape File Size Limitations

Because multivolume support is not necessarily VOL1-HDR1/EOV/EOF based a certain number of new features have to be implemented in order to let the guest program manage the multivolume on its own (examples: VM / DDR, DOS Tape Spooled output, etc.).

Multivolume support resides in the capacity of a drive to indicate to the controling program that it is about to reach the end of the physical tape and that measures have to be taken to close the current volume and request a new media.

Three options are available:

- MAXSIZE [K | M]=nnnn. The resulting file size is limited to the amount specified. MAXSIZE specifies bytes, MAXSIZEK specifies a multiple of 1024 bytes and MAXSIZEM specifies a multiple of 1024*1024 bytes. Specifying a size of 0 indicates that there is no limit on the size of the file. The default is 0 (unlimited file size).
- STRICTSIZE={ 0 | 1}. Upon reaching the tape file size limit, depending on STRICTSIZE, the tape file will or will not be truncated to enforce the maxsize limit. The limit is only enforced during a write type operation (that is, if the file already exists and the program only reads the file, then the file will not be truncated regardless of the strictsize setting). This affects any write that starts below the limit but that would extend beyond the limit.

  This parameter only affects compressed HET files. On AWS tapes the limit is always enforced but the file is not truncated (i.e. the write does not occur, because AWS tapes are never truncated and the effects of the write are known in advance (no compression)). Regardless of strictsize, any write operation (Write, Write TM) will return a Unit Check with Equip Check to the program if the file size exceeds the predefined limit. If strictsize is 0 the write will actually have been performed on the tape file. If strictsize is 1 the file will be truncated on the preceeding tape block boundary.

  Care must be taken that regardless of the 'strictsize' setting, the tape may become unusable for the guest program should an event such as absence of a Tape Mark occur for example. This option has no effect if MAXSIZE is 0.
- EOTMARGIN=nnnn. This option specifies, in bytes, the threshold before reaching maxsize during which an indication will be returned to the program to indicate that an EOT marker has been reached for a write type operation. The indication of reaching near-capacity is indicated to the program by presenting Unit Exception in the CSW on a Write type operation along with Channel End and Device End.

  For certain device types, sense information may also indicate this information independently of a write operation. The purpose of this option is to allow the program to determine that it is time to change and request a new tape.

5.7 Channel-to-Channel Adapters

Device Type	Emulated by
IBM 3088	TCP Socket, TUN/TAP Driver

Table 7: Channel-to-Channel Adapters

The first argument specifies the emulation type (see below) while the remaining arguments depend on the chosen emulation type. The following types of emulation are supported.

5.7.1 CTCI Channel-to-Channel Link to TCP/IP Stack

The CTCI (Channel-to-Channel Link to TCP/IP Stack) is a point-to-point IP connection with the TCP/IP stack of the driving system on which Hercules is running. This implementation is only for the Linux version of Hercules. For Windows the below CTCI protocol must be used.

5.7.2 CTCI Channel-to-Channel Link to Win32 TCP/IP Stack

The CTCI (Channel-to-Channel Link to Win32 TCP/IP Stack) is a modified Win32 version of the CTCI protocol for the windows systems. Please note that the protocol name in the Hercules configuration file is the same (CTCI) even though the actual implementation is different. Earlier versions of Hercules have used the protocol name CTCI-W32 in the configuration file. See chapter 16 (“CTCI-W32”) for details.

5.7.3 CTCT Channel-to-Channel Emulation via TCP Connection

The CTCT (Channel-to-Channel Emulation via TCP Connection) protocol is an emulated CTCA (Channel-to-Channel Adapter) to another Hercules system. CTCT currently only supports IP traffic so it cannot be used to transport NJE, SNA, PVM, etc. workload at present.

5.7.4 LCS LAN Channel Station Emulation

The LCS is an emulated LAN Channel Station Emulation. This emulation mode appears to the operating system running in the Hercules machine as an IBM 8232 LCS device, an IBM 2216 router, a 3172 running ICP (Interconnect Communications Program), the LCS3172 driver of a P/390 or an IBM OSA (Open Systems Adapter).

Rather than a point-to-point link this emulation creates a virtual Ethernet adapter through which the guest operating system running in the Hercules Machine can communicate. As such this mode is not limited to TCP/IP traffic but will in fact handle any Ethernet frame. Please note that the SNA mode is currently not implemented.

5.8 FBA Direct Access Storage Devices

Device Type	Emulated by
IBM 3310	Disk File(s)
IBM 3370	Disk File(s)
IBM 9332	Disk File(s)
IBM 9335	Disk File(s)
IBM 9336	Disk File(s)
IBM 0671	Disk File(s)

Table 8: FBA Direct Access Storage Devices

The argument for this device type specifies the name of a file that contains the FBA DASD image or the INET address of a Hercules shared device server. The file consists of fixed length 512-byte records each of which represents one physical block of the emulated disk.

5.9 CKD Direct Access Storage Devices

Device Type	Emulated by
IBM 2305	Disk File(s)
IBM 2311 (Model 1)	Disk File(s)
IBM 2314 (Model 1)	Disk File(s)
IBM 3330 (Model 1, 2 and 11)	Disk File(s)
IBM 3340 (Model 1, 2, 35 and 70)	Disk File(s)
IBM 3350 (Model 1)	Disk File(s)
IBM 3375 (Model 1)	Disk File(s)
IBM 3380 (Model 1, 2, 3, A, B, D, E, J, and K)	Disk File(s)
IBM 3390 (Model 1, 2, 3, 9, 27 and 54)	Disk File(s)
IBM 9345 (Model 1 and 2)	Disk File(s)

Table 9: CKD Direct Access Storage Devices

The argument for this device type specifies the name of a file that contains the FBA DASD image or the INET address of a Hercules shared device server. The file consists of a 512-byte device header record followed by fixed length track images. The length of each track image depends on the emulated device type and is always rounded up to the next multiple of 512 bytes.

Volumes larger than 2 GB (eg: 3390 model 3 or 9) can be supported by spreading the data across more than one file similarly to the original IBM P/390 implementation. Each of the files contains a whole number of cylinders. The first file, containing cylinders 0 to 2518 in the case of a 3390, usually has "_1" as the last two characters of its name. The ckddasd driver allocates the remaining files by replacing the last character of the file name by the character "2", "3" etc. This implementation was originally necessary as the IBM P/390 ran under the OS/2 operating system which supported a maximum file size of 2GB.

Hercules also has implemented so called "Large File Support". If the operating system supports large file sizes (or 64-bit offsets) then volumes larger than 2 GB can be kept in a single file.

5.10 Communication Lines

Device Type	Emulated by
IBM 2703	TCP Socket

Table 10: Communication Lines

Communication Lines are the preliminary 2703 BSC support. This protocol describes a BSC emulation line entry to either two Hercules engines or a custom made program emulating a 2780, 3780 or 3x74, or a custom made program interfacing to a real BSC line.

The communication is emulated over a TCP connection. All bytes are transferred as-is (except for doubling DLE in transparent mode) just like it would be over a real BSC link. Emulated EIA (DCD, DTR, CTS, etc.) or X.21/V.11 leads are treated differently depending on the DIAL option selected.

The line emulates a point-to-point BSC link, there is no point-to-multipoint handling. The communication protocol is basic. Every character written by the guest program with a WRITE CCW is transferred untranslated and untouched to the remote end, except for Transparent BSC rules, which deem that DLE characters are doubled when the program has previously written a DLE / STX sequence.

Dial data is originally as follows:

```
x x x x 0 0 0 0   :   Dial # 0
x x x x 0 0 0 1   :   Dial # 1
.
.
x x x x 1 0 0 0   :   Dial # 8
x x x x 1 0 0 1   :   Dial # 9
x x x x 1 1 0 0   :   EON
x x x x 1 1 0 1   :   SEP
```

In order to perform an outgoing call, the data must follow these specifications:

```
n [n [n]] SEP n [n [n]] SEP n [n [n]] SEP . . . n [n [n]] [EON]
```

Where n is any dialing number from 0 to 9 and SEPp is the separator. The first four groups of digits represent the IP address. The last group represents a TCP port number. For example if "*" is the SEP character representation, the following will issue a TCP connection to 192.168.0.1 port 8888:

```
192*168*0*1*8888
```

The EON is optional. If it is present it must be the last character of the dial data.

6. CCKD Compressed CKD DASD Devices

Alternatively to the standard CKD DASD devices it is possible to specify the name of a file containing a compressed CKD DASD image (CCKD). The CKD driver will automatically detect whether the file contains a regular CKD image or a compressed CCKD image.

6.1 CCKD Introduction

Using compressed DASD files you can significantly reduce the file space required for emulated DASD files and possibly gain a performance boost as less physical I/O occurs. Both CKD (Count-Key-Data) and FBA (Fixed-Block-Architecture) emulation files can be compressed.

In regular (uncompressed) files each CKD track or FBA block occupies a specific spot in the emulation file. The offset of the track or block in the file can be directly calculated knowing the track or block number and the maximum size of the track or block.

In compressed files each track image or group of blocks may be compressed by ZLIB or BZIP2 and only occupies the space neccessary for the compressed image. The offset of a compressed track or block is obtained by performing a two-table lookup. The lookup tables themselves reside in the emulation file.

In the event of a catastrophic failure (for example a Hercules crash, an operating system crash or a power failure), the compressed emulation file on the host's physical disk may be out of sync if the host operating system defers physical writes to the file system containing the emulation file. A number of techniques have been provided to minimize emulation file corruption in such an event.

A compressed file may occupy only 20% of the disk space required by an uncompressed file. In other words you may be able to have 5 times more emulated volumes using compressed DASD files. However compressed files are more sensitive to failures and corruption may occur.

6.2 CCKD Shadow Files

A compressed CKD or FBA DASD can have more than one physical file. These additional files are called shadow files. The function is implemented as a kind of snapshot, where a new shadow file can be created on demand. An emulated DASD is represented by a base file and zero or more shadow files. All files are opened read-only except for the current file which is opened read-write.

A shadow file contains all the changes made to the emulated DASD since it was created and until the next shadow file is created. The moment of the shadow files creation can be thought of as a snapshot of the current emulated DASD at that time. If the shadow file is later removed then the emulated DASD reverts back to the state it was at when the snapshot was taken.

Using shadow files you can keep the base file on a read-only device such as CD-ROM, or change the base file attributes to read-only ensuring that this file can never be corrupted.

There can be up to 8 shadow files in use at any time for an emulated DASD device. The base file is designated *file[0]* and the shadow files are *file[1]* to *file[8]*. The highest numbered file in use at a given time is the current file where all writes will occur. Track reads start with the current file and proceed down until a file is found that actually contains the track image.

Hercules console commands are provided to manage shadow files, commands are available for:

- Creating a new shadow file
- Removing a shadow file with backwards merge (commit all changes / updates)
- Removing a shadow file without backwards merge (discard all changes / updates)
- Compressing the current file
- Displaying shadow file status and statistics

- Performing a chkdsk on an active shadow file

6.3 Compressed DASD File Structure

A compressed DASD file has 6 types of spaces:

- A device header
- A compressed device header
- A primary lookup table
- Secondary lookup tables
- Track or block group images
- Free spaces

The first 3 types only occur once at the beginning of the file in order. The rest of the file is occupied by the other 3 space types.

6.3.1 Device Header

The first 512 bytes of a compressed DASD file contains a device header. The device header contains an eye-catcher that identifies the file type (CKD or FBA and base or shadow). The device type and file size is also specified in this header. The header is identical to the header used for uncompressed CKD files except for the eye-catcher:

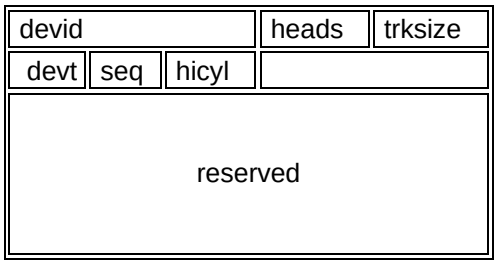


Figure 7: CCKD Device Header

6.3.2 Compressed Device Header

The next 512 bytes contains the compressed device header. This contains file usage information such as the amount of free space in the file:

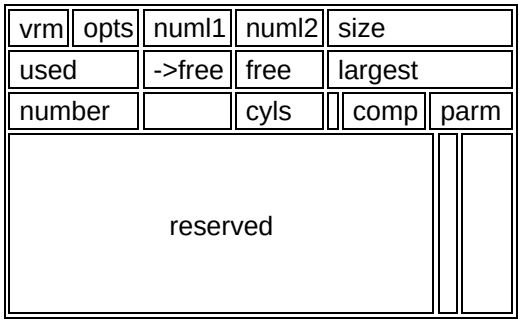


Figure 8: CCKD Compressed Device Header

6.3.3 Primary Lookup Table

After the compressed device header is the primary lookup table, also called the level 1 table or l1tab. Each 4 byte unsigned entry in the l1tab contains the file offset of a secondary lookup table or level 2 table or l2tab. The track or block group number being accessed divided by 256 gives the index into the l1tab. That is, each l1tab entry represents 256 tracks or block groups. The number of entries in the l1tab is dependent on the size of the emulated device:

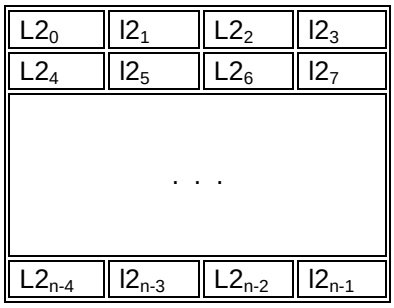


Figure 9: Primary Lookup Table

6.3.4 Secondary Lookup Table

Following the l1tab, in no particular order, are l2tabs, track or block group images and free spaces. Each secondary lookup table (or l2tab) contains 256 8-byte entries. The entry is indexed by the remainder of the track or block group number divided by 256. Each entry contains an unsigned 4 byte offset and an unsigned 2 byte length of the track or block group image:

0 ->image	length	unused
1 ->image	length	unused
. . .		
255 ->image	length	unused

Figure 10: Secondary Lookup Table

6.3.5 Track or Block Group Image

A track or block group image contains two fields, a 5-byte header and a variable amount of data that may or may not be compressed. The length in the l2tab entry includes the length of the header and the data.

hdr	track or block group data

Figure 11: Track or Block Group Image

The 5 byte header contains a 1 byte flag field and 4 bytes that identify the track or block group. The format of the identifier depends on whether the emulated device is CKD or FBA:

CKD Header

flags	CC	HH

Figure 12: CKD Header

The 2 byte CC is the cylinder number for the track image and the HH is the head number. These numbers are stored in big-endian byte order. When the flag byte is zeroed, the 5 byte header is identical to the Home Address (or HA) for the track image. The data, which may or may not be compressed, begins with the R0 count and ends with the end-of-track (or eot) marker, which is a count field containing 8 0xff's. The HA plus the uncompressed track data comprise the track image.

FBA Header

flags	nnnn

Figure 13: FBA Header

The 4 byte nnnn field is the FBA block group number in big-endian byte order. The data contains 120 FBA blocks which may or may not be compressed. Uncompressed, the FBA block group is 60K. The header for FBA, unlike CKD, is not used as part of the uncompressed image.

The flags byte contains 8 bits in the format

0	0	0	0	0	0	**c**	**c**

Figure 14: Flags Byte

The first 6 bits are always zero but may be used in future releases. The last two bits, cc in the figure above, indicate the compression algorithm for the data portion:

0 0	Data is uncompressed
0 1	Data is compressed using zlib
1 0	Data is compressed using bzip2
1 1	Not valid

Figure 15: Compression Algorithm Bits

6.3.6 Free Space

Free space contains a 4-byte offset to the next free space, a 4-byte length of the free space and zero or more bytes of residual data

->next	length	residual

Figure 16: Free Space

The minimum length of a free space is 8 bytes. The free space chain is ordered by file offset and no two free spaces are adjacent. The compressed device header contains the offset to the first free space. The chain is terminated when a free space has zero offset to the next free space. The free space chain is read when the file is opened for read-write and written when the file is closed; while the file is opened, the free space chain is maintained in storage.

6.4 How it works

This chapter explains in detail how the compressed CKD DASD implementation works.

6.4.1 Reading

A track or block group image is read while executing a channel program or by the readahead thread. An image has to be read before it is updated or written to. An image may be cached. If an image is cached then the channel program may complete synchronously. This means that if all the data a channel program accesses is cached and Hercules does not have to perform physical I/O, then the channel program runs synchronously within the SSCH or SIO instruction in the CPU thread. All DASD channel programs are started synchronously. If a CCW in the channel program requires physical I/O then the channel program is interrupted and restarted at that CCW asynchronously in a device I/O thread.

All compressed devices share a common cache; the devices can be a mixture of FBA and / or CKD device types. Each cache entry contains a pointer to a 64K buffer containing an uncompressed track or block group image. If the track or block group image being read is not found in the cache then the oldest (or least recently used or LRU) entry that is not busy is stolen. A cache entry is busy if it is being read, last accessed by an active channel program, updated but not yet written or being written. If no cache entries are available then the read must enter a cache wait. When images are detected to be accessed sequentially then the readahead thread(s) may be signalled to read following sequential images.

6.4.2 Writing

When a cache entry is updated or written to a bit is turned on indicating the cache entry has been updated. When a cache wait occurs or (more likely) during garbage collection a cache flush is performed. When the cache is flushed, if any entries have the updated bit on then the writer thread(s) are signalled. The writer thread selects the oldest cache entry with the updated bit on, compresses the image and writes it to the file. The new image is written to a new space in the file and then the space previously occupied by the image is freed. In certain circumstances the image may be written under stress. A stress write occurs when a reading thread is in a cache wait or when a high percentage of cache entries are pending write. In this circumstance the compression parameters are relaxed to reduce the CPU requirements. An image written under stress is likely to take up more space than the same image written not under stress. The writer thread(s) run 1 nicer than the CPU thread(s); compression is a CPU intensive activity.

6.4.3 Garbage Collection

The primary function of the garbage collector is to keep the emulated compressed DASD files as small as possible. This is after all the main reason for using compressed DASD files. Another function is to perform emulation file synchronization.

A single garbage collector thread runs for all compressed devices. By default it wakes up at 5 second intervals. The garbage collector performs space recovery for each compressed device in the order that the device was defined or attached. After space recovery the garbage collector flushes the cache to force all outstanding writes. Once all the writes have been completed a file synchronization (“fsync()”) may optionally be performed. This commits any outstanding host I/O to the physical disk. Finally free space is flushed (to be explained later).

We see that with the fsync option enabled that the physical disk file has a coherent emulation file at the end of each garbage collection cycle. Space freed since the last garbage collection cycle completed is not available for allocation until the current garbage collection cycle completes. This free space is called pending free space. That is, previous track or block group images are not overwritten until the current garbage collection completes. If a catastrophic error occurs, then the emulation file should be recoverable at least up to the point of the last garbage collection cycle.

However, performing an fsync() may decrease performance. You can increase the garbage collection interval to reduce the number of fsync()’s, but this may also increase the probability of a cache wait occurring. You can increase the size of the cache to decrease this probability, but you may increase paging or have to decrease the size of emulated memory.

Another possibility is to not enable the fsync option. This is the default. In this circumstance, by default, freed space is not available until 2 garbage collection cycles complete. That is, pending free space is not an attribute but a count. You have the option to explicitly set the pending free space count. However by increasing the free space count or by increasing the garbage collection interval then you may be increasing the size of the emulation file.

At the very end of the garbage collection cycle the free space is flushed. This means that the pending free space count is decremented for all free spaces with a non-zero count. If the count goes to zero and the preceding space is a free space with a zero count then the spaces are combined.

The space recovery process of the garbage collector simply attempts to move some amount of used space towards the beginning of the file causing free space to move towards the end of the file. When a free space reaches the end of the file, the file is truncated reducing its size. The amount of used space moved depends on the ratio of free space to used space and on the number of free spaces. The larger the numbers, the more space the garbage collector attempts to move. That is, the garbage collector attempts to decrease the ratio of free space vs. used space and to decrease the number of free spaces. Within a cycle, the garbage collector might not move the selected amount of used space if the moves are detected to be counter-productive (i.e. the offset of the new space is greater than the current offset).

7. Shared Device Support

Shared Device Support allows multiple Hercules instances to share devices. The device will be local to one instance and remote to all other instances. The local instance is the server for that device and the remote instances are the clients.

It is not necessary to IPL an operating system on the device server. Any number of Hercules instances can act as a server in a "Hercplex".

7.1 Usage of Shared Devices

To use a device on a remote system, instead of specifying a file name on the device statement, an IP address or a dns name is specified in the format:

```
devnum  devtype  ip_address_or_name:port:devnum
```

For example:

```
0100  3390  localhost:3990:0100
```

which says, there is a shared device server on the local host, listening on port 3990 and we want to use its 0100 device as our 0100 device. Because the default port number is actually 3990 and the default remote device number is the local device number we could shorten the above statement, providing we do not have actually a file named 'localhost', to

```
0100  3390  localhost
```

Interestingly the instance on the local host listening on port 3990 could itself have a statement like this:

```
0100  3390  192.168.200.1::0200
```

which means that this Hercules instance will use device 0200 on the server at 192.168.200.1 listening on port 3990. The original instance will have to hop through this second instance to get to the real device.

Device sharing can also be split between multiple instances. For example, suppose the following definitions for instance A:

```
SHRDPORT 3990
0100  3390  localhost:3991
0101  3390  mvscat.dasd
```

And for instance B we have the following definitions:

```
SHRDPORT 3991
0100  3390  mvsres.dasd
0101  3390  localhost
```

In this case each instance acts as both a client and a server. Both instances of Hercules are running on the same machine.

The above examples may be more clear if we specify also all the default values. To show this we define the same configuration but in this case the Hercules instances are running on separate physical machines.

Hercules instance A (machine IP 192.168.200.1):

```
SHRDPORT 3990
0100  3390  192.168.200.2:3991:0100   # Remote on 192.168.200.2 (mvsres.dasd)
0101  3390  mvscat.dasd               # Local on  192.168.200.1 (mvscat.dasd)
```

Hercules instance B (machine IP 192.168.200.2):

```
SHRDPORT 3991
0100  3390  mvsres.dasd               # Local  on 192.168.200.2 (mvsres.dasd)
0101  3390  192.168.200.1:3990:0101   # Remote on 192.168.200.1 (mvscat.dasd)
```

When "SHRDPORT" is specified in the Hercules configuration the thread "shared_server" is started at the end of Hercules initialization. In the example above neither Hercules instance can initialize their devices until the server is started on each system. In this case the device trying to access a server gets the 'connecting' bit set on in the DEVBLK and the device still needs to initialize. After the shared server is started a thread is attached for each device that is connecting, to complete the connection (the device init handler).

7.2 Caching

Cached records (i.e. CKD tracks or FBA blocks) are kept independently on both the client and server sides. Whenever the client issues a START request to initiate a channel program, the server will return a list of records to purge from the clients cache. These will have been updated by other clients since the last START request. If the list is too large the server will indicate that the client should purge all records for the device.

7.3 Compression

Data that would normally be transferred uncompressed between the client and the host can optionally be compressed by specifiying the "COMP=n" keyword on the device configuration statement or on the attach command.

For example:

```
0100  3390  192.168.2.12  COMP=3
```

The value n of the "COMP=n" keyword is the zlib compression parameter which must be a number between 1 and 9. A value closer to 1 means less compression but less processor time to perform the compression. A value closer to 9 means the data is compressed more but also more processor time is required to compress the data.

If the server is on localhost then you should not specify compression. Otherwise you are just stealing processor time from Hercules to facilitate compression/decompression. If the server is on a local network then a low value such as 1, 2 or 3 is recommended. There is a tradeoff curve, attempting to trade CPU cycles for network traffic to derive an optimal throughput.

If the devices on the server are compressed devices (i.e. CCKD or CFBA) then the records (track images or block groups) may be transferred compressed regardless of the "COMP=" settings. This depends on whether the client supports the compression type (zlib or bzip2) of the record on the server and whether the record is actually compressed in the server cache.

An example may help to explain this: Suppose on the client that you execute one or more channel programs to read a record on a CKD track, update a record on the same track, and then read another (or the same) record on the track.

For the first read the server will read the track image and pass it to the client as it was originally compressed in the file. To update a portion of the track image the server must uncompress the track image so data in it can be updated. When the client next reads from the track image the track image is uncompressed.

Specifiying "COMP=n" means that uncompressed data sent to the client will be compressed. If the data to be sent to the client is already compressed then the data is sent as is, unless the client has indicated that it does not support that compression algorithm.

7.4 Technical Approaches

There are (at least) two approaches to sharing devices. One is to execute the channel program on the server system. The server will need to request informations from the client system, such as the CCW and the data to be written and will need to send to the client data that has been read and status information. The second approach is to execute the channel program on the client system. Here the client system makes requests to the server system to read and write data.

The second approach is currently implemented. The first approach arguably emulates more correctly. However an advantage of the implemented approach is that this is easier as only the client sends requests and only the server sends responses.

Both client and server have a DEVBLK structure for the device. Absurdly perhaps, in originally designing an implementation for shared devices it was not clear what type of process should be the server. It was a quantum leap forward to realize that the server could be just another Hercules instance.

7.5 Protocol

The following sections describe the protocol used for the communication between the client and the server in the Shared Device Server implementation.

Please note that knowledge of the protocol used as described below is not necessary for exploitation of the Shared Device Server functionality. This information is presented purely for the more technically interested readers of this book.

7.5.1 Client Header and Data Part

The client sends an eight byte request header and maybe some data.

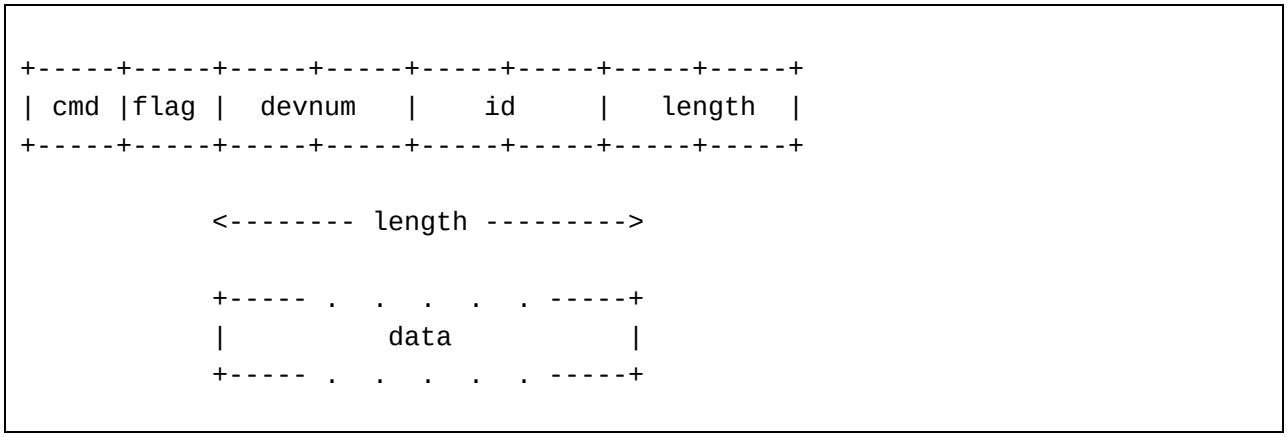


Figure 17: Shared Device Server – Client Header Structure

'cmd' identifies the client request. The requests are:

Cmd	Request	Explanation
0xe0	CONNECT	Connect to the server. This requires the server to allocate resources to support the connection. Typically issued during device initialization or after being disconnected after a network error or timeout.
0xe1	DISCONNECT	Disconnect from the server. The server can now release the allocated resources for the connection. Typically issued during device close or detach.
0xe2	START	Start a channel program on the device. If the device is busy or reserved by another system then wait until the device is available unless the NOWAIT flag bit is set, then return a BUSY code. Once START succeeds then the device is unavailable until the END request.
0xe3	END	Channel program has ended. Any waiters for the device can now retry.
0xe4	RESUME	Similar to START except a suspended channel program has resumed.
0xe5	SUSPEND	Similar to END except a channel program has suspended itself. If the channel program is not resumed then the END request is not issued.
0xe6	RESERVE	Makes the device unavailable to any other system until a RELEASE request is issued. Must be issued within the scope of START / END.
0xe7	RELEASE	Makes the device available to other systems after the next END request. Must be issued within the scope of START / END.
0xe8	READ	Read from a device. A 4-byte `record' identifier is specified in the request data to identify what data to read in the device context. Must be issued within

Cmd	Request	Explanation
		the scope of START / END.
0xe9	WRITE	Write to a device. A 2-byte `offset' and a 4-byte `record' is specified in the request data, followed by the data to be written. `record' identifies what data is to be written in the device context and `offset' and `length' identify what to update in `record'. Must be issued within the scope of START / END.
0xea	SENSE	Retrieves the sense information after an i/o error has occurred on the server side. This is typically issued within the scope of the channel program having the error. Client side sense or concurrent sense will then pick up the sense data relevant to the i/o error. Must be issued within the scope of START / END.
0xeb	QUERY	Obtain device information, typically during device initialization.
0xec	COMPRESS	Negotiate compression parameters. Notifies the server what compression algorithms are supported by the client and whether or not data sent back and forth from the client or server should be compressed or not. Typically issued after CONNECT. This action should actually be SETOPT or some such; it was just easier to code a COMPRESS specific SETOPT (less code).

Table 11: Shared Device Server – Client Request Codes

'flag' qualifies the client request and varies by the request:

Flag	Client Flag	Explanation
0x80	NOWAIT	For START, if the device is unavailable then return BUSY instead of waiting for the device.
0x40	QUERY	Identifies the QUERY request:
0x41	DEVCHAR	Device characteristics data
0x42	DEVUID	Device identifier data
0x43	DEVUSED	High used track / block (for dasdcopy)
0x48	CKDCYLS	Number of cylinders for CKD device
0x4c	FBAORIGIN	Origin block for FBA
0x4d	FBANUMBLK	Number of FBA blocks
0x4e	FBABLKSIZ	Size of a FBA block
0x3x	COMP	For WRITE, data is compressed at offset 'x':
0x2x	BZIP2	using bzip2

Flag	Client Flag	Explanation
0x1x	LIBZ	using zlib
0xxy		For COMPRESS, identifies the compression algorithms supported by the client (0x2y for bzip2, 0x1y for zlib, 0x3y for both) and the zlib compression factor 'y' for sending otherwise uncompressed data back and forth. If 'y' is zero (default) then no uncompressed data is compressed between client and server.

Table 12: Shared Device Server – Client Flags

'devnum' identifies the device number on the server instance. The device number may be different than the device number on the client instance.

'id' identifies the client to the server. Each client has a unique, positive (non-zero) identifier. For the initial CONNECT request 'id' is zero. After a successful CONNECT the server returns in the response header the identifier to be used for all other requests (including subsequent CONNECT requests). This is saved in dev -> rmtid.

'length' specifies the length of the data following the request header. Currently length is non-zero for READ / WRITE requests.

7.5.2 Server Header and Data Part

The server sends an eight byte response header and maybe some data.

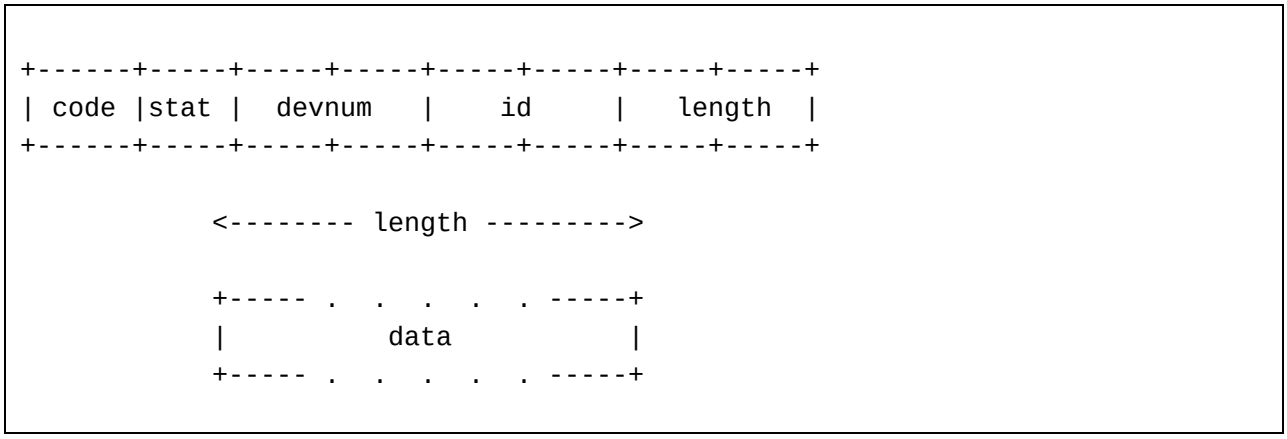


Figure 18: Shared Device Server – Server Header Structure

'code' indicates the response to the request:

Code	Response	Explanation
0x00	OK	OK indicates success, other codes may also indicate success but qualified in some manner.
0x80	ERROR	An error occurred. The server provides an error message in the data section.
0x40	IOERR	An i/o error occurred during a READ/WRITE request. The status byte has the `unitstat' data. This should signal the client to issue the SENSE request to obtain the current sense data.
0x20	BUSY	Device was not available for a START request and the NOWAIT flag bit was turned on.
0x10	COMP	Data returned is compressed. The status byte indicates how the data is compressed (zlib or bzip2) and at what offset the compressed data starts (0 .. 15). This bit is only turned on when both the `code' and `status' bytes would otherwise be zero.
0x08	PURGE	START request was issued by the client. A list of `records' to be purged from local cache is returned. These are `records' that have been updated since the last START/END request from the client by other systems. Each record identifier is a 4-byte field in the data segment. The number of records then is `length' / 4. If the number of records exceeds a threshold (16) then `length' will be zero indicating that the client should purge all locally cached records for the device.

Table 13: Shared Device Server – Server Response Codes

'stat' contains status information as a result of the request. For READ / WRITE requests this contains the 'unitstat' information if an IOERR occurred.

'devnum' identifies the server device number.

'id' specifies the system identifier for the request.

'length' is the length of the data returned.

8. SCSI Tape Drives

This section provides some additional information about Hercules's SCSI tape drive support.

8.1 Basics

Real SCSI tape drives used with Hercules must provide a certain minimum set of "IBM compatible" support in their SCSI command set/behavior in order to work properly with Hercules. Furthermore the Hercules device-type used on the Hercules configuration file device statement must match the level of support/behavior that the SCSI device(s) provide.

For example all SCSI tape drives used with Hercules must provide the ability to set variable-length blocks as well as long erase-gaps. Long erase-gaps allows new data to be appended to the end of existing data without having to write a tape-mark to separate the new data from the old existing data first.

Another example would be using a model of SCSI tape drive that happens to report physical block-id values in a format different from the way real IBM mainframe tape drives report them. 3480/3490 tape drives report their block-ids (used in Read Block Id and Locate CCW's) in a very specific format wherein bits 1-7 of the high-order byte of the reported 4-byte block-id indicates the tape's physical segment where the location of the lower 22-bit block number physically exists on the tape. The block-id segment is used to allow the tape drive to quickly position itself to the approximate location where the desired block actually resides on the tape and thus allows high-speed positioning for the Locate CCW.

If the model of SCSI tape drive used with Hercules does not use this same block-id format then it can not be used with Hercules as a 3480 or 3490 model tape drive with certain defined options.

If the SCSI tape drive used reports its block-ids using a 32-bit block-id value (the same way a 3590 model tape drive does), then similarly it should be defined to Hercules as a model 3590 device-type, since that is how it is behaving with respect the format of the returned blockid values. If you wish to define it in Hercules as a model 3480 or 3490, then you will need to use certain defined options before it will work properly as the model of drive you wish it to emulate.

It should be noted that partial support for 3590 device emulation is possible with judicious use of the mentioned special options, but full or complete 3590 support is unlikely due to lack of publicly available documentation. Details regarding 3590 CCW handling is restricted (confidential) IBM proprietary information and is not normally available outside of IBM. Not long ago IBM was required by US law to publish such information but unfortunately for Hercules this is no longer the case.

8.2 Special Options

Host-attached SCSI tapes are read and written using variable length EBCDIC blocks and filemarks exactly like a mainframe tape volume. As a result they can be freely used/exchanged, i.e. SCSI tapes created on a real mainframe can subsequently be read by Hercules and a SCSI tape created by Hercules can be subsequently read on a mainframe, thus providing a convenient means of exchanging data between the two.

The only required device statement parameter for SCSI attached tape drives is the name of the device as it is known by the host. However depending on what actual model of SCSI tape drive is used and how it behaves, it may be necessary to specify one or more optional parameters for Hercules to provide proper emulation of the desired device type.

For example: a Quantum 'DLT' (Digital Linear Tape) SCSI tape drive does not return or use a block-id format compatible with 3480/3490 drives. It uses a full 32-bit block-id just like the model 3590 does. It also does not support the Erase Gap CCW at all. Thus in order to use a Quantum DLT drive with Hercules, you must specify some special additional options to prevent the Erase Gap command from being issued to the drive as well as to inform Hercules that the drive uses 32-bit block-ids.

At present the only optional device statement parameters supported are those listed below. Please note that the below options define how the actual SCSI hardware device behaves, which is completely different from the way the emulated device will appear to behave to your guest. That is to say, if you define your tape drive to Hercules as a 3480 device then Hercules will perform 3480 device type emulation such that the device appears to your guest operating system as if it were a 3480 device. If the actual SCSI device behaves as a 3590 device however (perhaps using/returning 32-bit block-ids instead of the expected 22-bit format block-ids that 3480's use) then you must specify the "--blkid-32" special option on your Hercules device statement. This is so that Hercules's emulation logic knows that it needs to translate 22-bit block-ids to 32-bit ones before sending them to the actual SCSI hardware and vice versa: to translate 32-bit block-ids from the actual SCSI drive into 22-bit format block-ids that your guest expects from a 3480 device.

8.2.1 Special Option --no-erg

This option is intended to prevent issuing the Erase Gap command to those SCSI tape drives that do not support it (such as the Quantum DLT series). It causes Hercules's device emulation logic to ignore any Erase Gap commands issued to the drive and to return immediate 'success' instead.

This option should only be used for drives such as the Quantum which support switching from read mode to write mode in the middle of a data stream without the need of an intervening Erase Gap command. Specifying it for any other model SCSI drive may cause incorrect functioning as a result of the Erase Gap command not being issued to the actual SCSI hardware.

Check the manufacturer information for your particular model of SCSI-attached tape drive (and/or use Fish's "ftape" Windows utility) to determine whether or not this option is needed for your particular drive.

8.2.2 Special Option --blkid-32

This option indicates that the SCSI-attached tape drive only supports 32-bit block-ids (as used by 3590 drives) and not the 22-bit format used by 3480/3490 drives. You should only specify this option if you intend to define the drive as a model 3480 or 3490 device and then only if your actual SCSI drive uses 32-bit block-ids. If you define your Hercules tape drive as a model 3590 device however, then this option is not needed since model 3590 drives are already presumed to use 32-bit block-ids.

Specifying this option on a 3480/3490 device statement will cause Hercules device emulation logic to automatically translate the actual SCSI tape drive's 32-bit block-id into 22-bit format before returning it back to the guest operating system. This is the format the guest will expect for a model 3480/3490 drive. The guest's 22-bit format block-id will also be translated into 32-bit format before sending it to the actual SCSI hardware, since that is the format that the actual hardware requires it to be in.

8.2.3 Special Option --blkid-22

This is the opposite of the above –blkid-32 option.

9. Hercules Dynamic Loader

The Hercules Dynamic Loader is intended to supply a loading and linking mechanism whereby routines, commands, instructions and functions can be dynamically added to Hercules without the need to rebuild or even restart Hercules.

The loader can be controlled by the following Hercules commands:

- ldmod <module list> Load modules named in module list
- rmmod <module list> Unload modules named in module list
- lsmod List all modules and entry points
- lsdep List all dependencies

The loader can also be controlled by some configuration statements:

- LDMOD <module list> Load modules named in module list
- MODPATH <pathname> Specifies where the modules are loaded from

The loader has 2 basic functions; module load and module unload. When the module load function receives a path name along with the module name, the module is loaded from that specific path. If no path is given, the module is loaded from the default library search order. Note that this is different from the standard search order.

Additionally there are two resolve functions. The first one will return the entry point of a symbol name, the second one will return the previous entry point, i.e. the entry point that was active before the current entry point was registered. This function is intended to allow a module to call the original routine.

There are some special considerations for systems that do not support the concept of back-linking. Back-linking is the operating system support of dynamically resolving unresolved external references in a dynamic module, with the main module or other loaded modules. Cygwin does not support back-linking.

Unload will remove all references to a specific module but currently it will not actually remove the loaded module from memory. This is because there is no safe way (yet) to synchronize unloading of code and besides, it may still be in use. This should however pose no practical limitations.

When a module lists a new dependency that dependency will be registered. Unloading the module does not remove the dependency; this is to be consistent with the previous note about unloading.

The following subchapters describe some of the Hercules Dynamic Loader sections in more detail.

9.1 Dependency Section

The dependency section is - for all practical purposes - called before the module is loaded. Its function is to check that there are no incompatibilities between this module and the version of Hercules that we are running. Dependencies are identified by name.

Each dependency then has a version code and a size code, where the version code is a character string and the size code an integer value. If the version or size codes do not match with those in the Hercules main module, the module cannot be loaded. The version is usually a character string that identifies the version of the component, and the size specification is the size of the component in the case of structures or unions.

When a dependency is given that has not yet been registered, it will be registered such that it can be checked in subsequent module loads.

9.2 Registration Section

The registration exports labels and their associated entry points to Hercules, such that the symbols and associated entry points may be known to Hercules and any other module that may have been loaded. The registration section is called once during module load.

If we have registered a function that is also called from this DLL then it must also be listed in the resolver section. This to ensure that the symbol is properly resolved when other modules are loaded.

9.3 Resolver Section

The resolver section imports the entry points of symbols that have been previously registered. When a symbol is requested that has not been previously registered then the resolve function will search the loaded modules for that symbol and register it implicitly. This latter function is mainly provided to support systems that do not have back-link support (most notably Cygwin).

The resolver may be called multiple times, the first time it is called is during module load immediately after the registration section is called. It is subsequently called when other modules are loaded or unloaded.

9.4 Device Section

The device section is to register device drivers with Hercules. It associates device types with device handlers. If a device handler is not registered for a specific device type and a loadable module with the name of "hdtxxxx" (where xxxx is the device type) exists, then that module is loaded.

Search order:

1. The most recently registered (i.e. loaded) device of the requested device type.
2. Device driver in external loadable module, where the module name is hdtxxxx (where xxxx is the device type, i.e. module name hdtlcs for device type LCS or hdt2703 for device type 2703)
3. If the device is listed in the alias table, then external module hdtyyyy will be loaded, where yyyy is the base name as listed in the alias table. The device name is always mapped to lower case when searching for loadable modules.

9.5 Final Section

The final section is called once when the module is unloaded or when Hercules terminates. The final section is intended to be used to perform cleanup or indicate cleanup action to be taken. It may set a shutdown flag that is used within this DLL to indicate that all local functions must now terminate.

10. ECPSVM – Extended Control Program Support VM/370

This chapter describes some details about the implemented ECPSVM (Extended Control Program Support VM/370) functions in Hercules.

10.1 Technical Information

The CP Assists provide the VM SCP with various microcoded instructions to shorten the supervisor path-length. All microcoded instructions are priviledged instructions and have an opcode of E6xx. They are a native representation of what the SCP would do in a similar case. For all cases where the assist is not able to resolve a situation the E6XX instructions resolve to a no-op, thus leaving the responsability of the task to the original CP Code.

The VM Assists alters the behaviour of certain priviledged instructions when executed in problem state (controled by the Problem State bit in the PSW) either by completely simulating the instruction (when feasable), branching directly to the CP support module for that instruction (therefore bypassing program interruption processing and instruction decoding) or otherwise generating a program interruption. The VM Virtual Interval Timer assist allows updating of a Virtual Machine virtual interval timer directly by the microcode.

Both CP and VM Assists are controlled by real Control Register 6 which controls availability and behaviour of the assists.

10.2 Troubleshooting

In the event that a certain CP or VM Assist disrupts normal operations it is possible to selectively disable each discrete component. In this situation the best method of diagnoses is to disable all VM and CP Assists (except STEVL and SSM if done prior to IPL) and to enable each feature until the problem reoccurs. If it is unknown whether the problem lies in the VM or CP Assist it is also possible to enable/disable the entire group of assists.

ECPSVM STA allows you to see how often each assist is invoked. The hit and hit-ratio make it possible to determine how effective the assists are. A low hit ratio may be normal in some situations, for example the LPSW hit ratio will be very low when running VM under VM as most PSW switches cannot be resolved by the assist.

A low invocation count simply shows that in that particular situation the related assist is not used often, for example there are very few LCTLs when running CMS. Some assists are just invoked once at IPL, such as STEVL. This is normal behaviour.

10.3 Implemented Assists

The next sections list the implemented CP and VM Assists.

10.3.1 CP Assists

The following CP Assists are implemented:

- FREEX, FRETX (CP Free Storage Management)
- DISP0, DISP1, DISP2 (CP Dispatching)
- PGLOCK, PGULOCK (Real Frame Locking/Unlocking)

- TRANBRNG, TRANLOCK (Virtual Frame Addressing/Locking)
- SCNRU, SCVNU (Real/Virtual Device Control Block Scan)
- STEVL (Store ECPS:VM Support Level)

10.3.2 VM Assists

The following VM Assists are implemented:

- Virtual Interval Timer
- LPSW Simulation
- SSM Simulation
- SVC Simulation
- LCTL Simulation

10.4 Non-Implemented Assists

The next sections list the non-implemented CP and VM Assists.

10.4.1 CP Assists

The following CP Assists are not (yet) implemented:

- FREE, FRET (Original CP Storage Management, replaced by FREEX, FRETX)
- CCWNG, DFCCW, DNCCW, UXCCW (CCW/CSW Translation Assists)
- LCSPG (Locate Changed Shared Page/Segment Invalidation)
- VIPT, VIST (Virtual Translation Page/Segment Invalidation)
- LINK, RETURN (SVC 8/SVC 12)
- Prefered Machine Assists (Insufficient information available…)

10.4.2 VM Assists

The following VM Assists are not (yet) implemented:

- V=R Shadow Table Bypass Assists (including LRA instruction)
- SIO
- DIAG
- IUCV
- STxSM
- ISK, SSK, ISKE, SSKE, IVSK (Extended Key Operations Assists)
- VM Assists for MVS

10.5 Restrictions

ECPS:VM will not work in an AP or MP system. An AP or MP generated system locks the control blocks being manipulated by the assisted functions (VMBLOK, RDEVBLOK, VDEVBLOK, etc.). However the current ECPS:VM implementation does not lock any of those structures. Therefore the CP will fairly quickly abend as it will discover some of the control blocks that have not been locked when they should have been (various LOKXXX abends). Consequently ECPS:VM must be disabled when an AP or MP system is used.

11. Hercules Automated Operator (HAO)

This section describes the Hercules Automated Operator (HAO) feature. This facility allows you to automatically execute panel commands in response to certain messages being issued.

11.1 Introduction

The Hercules Automated Operator (HAO) allows to "fire" panel commands in response to certain messages issued on the Hercules console.

To use the HAO facility it is necessary to define a rule consisting of a target and an associated command. The target is just a regular expression pattern used to match against the text of the various messages that Hercules issues as it runs. Whenever a match is found the rule "fires" and its associated command is executed automatically.

The Hercules Automated Operator facility is only for those messages issued by Hercules to its HMC (Hardware Management Console). It cannot be used for messages the guest operating system (i.e. MVS, VM, VSE etc.) may issue to any of its terminals or consoles.

11.2 Defining Rules

To define a HAO rule the following command is used:

HAO TGT <tgt>

This defines the rule's "target" match pattern (a simple regular expression). This command has to be completed by the command:

HAO CMD <cmd>

this defines the rule's associated panel-command.

The target pattern is a simple regular expression value as defined by whatever regular expression support your host build platform happens to provide. For Windows it must be a Perl Compatible Regular Expression (PCRE). For other supported platforms it might be some other supported regular expression syntax. Check your host platforms programming documentation for further details.

The associated panel-command <cmd> is whatever command you wish to issue in response to a message being issued that matches the given pattern (target). The command may be any text string one desires, it does not have to be a valid Hercules command but this is the expected use.

11.3 Deleting Rules

To delete a fully or partially defined HAO rule first use the the following command to list all of the defined (or partially defined) rules:

HAO LIST [nnn]

This gives you the list of all rules with the specified identifier or lists the rule with identifier 'nnn'. Then use the next command to delete the specific rule identified by the identifier 'nnn':

HAO DEL <nnn>

All rules are assigned numbers as they are defined and are subsequently identified by their numeric value. It is also possible to delete all defined or partially defined rules by issuing the command:

HAO CLEAR

11.4 Limitations

The current implementation limits the total number of defined rules to 64. If you need to define more than 64 rules you will either have to build Hercules for yourself - with previously increasing the value of the HAO_MAXRULE constant in hao.c - or else request one of the Hercules developers to make this change for you.

Note that there is currently no way to define a command whose arguments vary based on actual message text. That is to say there is currently no way to say:

"Reply with the command 'devinit <cuu> filename' in response to message text 'HHCXXnnnI Device 'cuu' intervention required' where 'cuu' is whatever cuu was identified in the message."

The HAO is not very sophisticated yet, only simple plain-text commands may be defined and issued. No automatic substitution is done based on message text with the exception of normal 'DEFSYM' symbol substitution as this is a normal panel-command feature supported separately from the HAO. This may possibly change in the future depending on user need/demand.

All defined rules are checked for a match each time Hercules issues a message. There is currently no way to specify "stop processing subsequent rules". If a message is issued that matches two or more rules each associated command is issued in sequence. The recommendation is to choose your rules target patterns with care to avoid undesired results.

12. Summary of Hercules Changes

12.1 Hercules Evolution

This documentation is based on the Hercules S/370, ESA/390 and z/Architecture Emulator Release 3.07.0. This chapter briefly lists the changes for all Hercules releases back to version 1.32.0 released October 1999. A detailed description with all changes (any functional enhancement as well as bug fix) of all recent releases can be found in the Hercules “Changes” document, which is delivered with the source files.

12.2 Changes for Hercules Emulator Release 3.07.0

Release date: March 10, 2010

- Fast Synchronous Data Mover Facility
- Diagnose 210, 250, 260
- Extended Diagnose 204 feature
- Complete Diagnose 24
- Configuration-Topology feature
- HFP-Unnormalized-Extensions Facility
- CMPSC performance improvements
- Uptime panel command
- Raise XPNDSIZE limit to 1048576 MB
- MAXCPU and LPARNUM configuration statements
- Add capacity model identifiers to MODEL configuration statement
- SCLPROOT configuration statement
- Add "NOCLEAR" option to printer and card punch devices
- Socket printer support
- 3705 SNA device support
- TTY and 2741 support for 2703
- Tracing enhancements
- Allow "configure --enable-external-gui" for Unix builds
- Enable tun/tap emulation for 64-bit Windows builds
- 64-bit Windows support
- Raise MAX_CPU_ENGINES limit to 64
- Numerous bug fixes

12.3 Changes for Hercules Emulator Release 3.06.0

Release date: January 11, 2009

- Integrated 3270 (SYSG) console support
- HMC DVD-RAM read/write support
- 64-bit native version now supported on Mac OS X
- Ability to specify IFL, zIIP and zAAP engine types
- Console-like message handling
- Tape automount CCW support
- CKD Locate Record Extended CCW
- Support for FLEX-ES "Fake Tape" tape images
- More complete 3490 and 3590 tape support
- Solaris build support
- Free BSD build support
- Panel enhancements:
  - Display virtual storage in primary, secondary and home space
  - Display and modify PSW fields by panel command
  - Modify control registers by panel command
  - Specify IPL parameter by PARM operand
  - New panel commands: automount, cmdtgt, ctc, herc, msghld, pscp, scp, sfk
- LEGACYSENSEID system parameter
- New instruction feature support (introduced with System z10):
  - Parsing-Enhancement Facility
  - Message-Security-Assist Extension 2
  - General-Instructions-Extension Facility
  - Execute-Extensions Facility
  - Move-with-Optional-Specifications Facility
  - Compare-and-Swap-and-Store Facility 2
- Many emulation fixes

12.4 Changes for Hercules Emulator Release 3.05.0

Release date: June 23, 2007

- Prebuilt Cygwin binary no longer supplied; building Cygwin version from source still supported
- New system features: Compare-and-Swap-and-Store, Conditional SSKE, Decimal Floating Point, Floating Point Support Enhancement
- Extract-CPU-Time Facility
- Multiple Logical Channel Subsystems Facility
- 3590 Tape Support

- 3990-6 Control Unit and ECKD support
- Many performance improvements
- Many emulation fixes
- Major SCSI tape fixes
- Added floating point instructions CGER, CGDR and CGXR
- Address range options for instruction tracing and stepping
- Update GPR registers via gpr panel command
- Console connection keep-alive
- Customizable 3270 connection Screen ("Hercules Logo")
- DASDCONV quiet and stdin options
- Hercules Automatic Operator
- Enhanced symbol substitution
- Miscellaneous new panel commands: qd, fpc, traceopt, logopt, cd, pwd, timerint, defsym
- Miscellaneous new system parameters: IGNORE, INCLUDE, LOGOFILE, MANUFACTURER, MODEL, MOUNTED_TAPE_REINIT, PANTITLE, PLANT, TIMERINT

12.5 Changes for Hercules Emulator Release 3.04.1

Release date: March 25, 2006

- Fix to allow building for Intel-based Mac OS X.

  Note: This version only applies to the Mac OS X 10.4 (Tiger) platform. Version 3.04 is current for all other platforms.

12.6 Changes for Hercules Emulator Release 3.04.0

Release date: February 24, 2006

- CCKD garbage collection fix.
- Reworked timing functions.
- Codepage 1047 conversion tables.
- Fixed “off-by-one-day” bug with SYSEPOCH other than 1900.
- Added warning, if SYSEPOCH is not 1900 or 1960.
- New config parameter YROFFSET.
- New 2305 CKD disk emulation.
- Added floating point instructions CEGR, CDGR and CXGR.
- Added support for cgi-bin dynamic modules.
- Instruction fixes for PLO and CVB.
- Fix for Windows ..\relative path dasd files.

12.7 Changes for Hercules Emulator Release 3.03.1

Release date: December 30, 2005

- Fix translation exception bug that was causing some Linux kernels to panic.
- TOD Clock-Steering Facility.
- Fix bug in shadow file filename processing on native Windows.
- Performance improvements in TM instruction family.
- Support for Linux zipl LOADPARM of PROMPT.

12.8 Changes for Hercules Emulator Release 3.03.0

Release date: December 20, 2005

- Pure Win-32 application ("MSVC Hercules"), Native Windows version no longer requires Cygwin
- SMP host integrity fixes
- ALS5, z9 and other architectural enhancements
- Restructured cryptographic support no longer depends on libgcrypt
- Support emulation of up to 32 CPUs; maximum without special build options now 8
- Enhanced semigraphical control panel now uses all of larger console windows
- Many emulation fixes
- Integrated 1052-C / 3215-C console support
- Tapecopy support for writing as well as reading tapes

12.9 Changes for Hercules Emulator Release 3.02.0

Release date: December 11, 2004

- Significant Performance Improvements (overall improvement about 30%)
- SIE Performance almost the same as native
- SCSI tape support in Windows
- MAC OS X CTC networking support
- Suspend / Resume facility
- ASN-and-LX-Reuse Facility / Enable or disable ASN-and-LX-Reuse in configuration
- Extended Translation Facility 3
- DAT-enhancement facility
- Immediate CCWs now correctly handled when Suppress Incorrect Length Indication is specified
- 3270 option provided to control connection to group of devices
- 3270 connections can be limited by IP address
- Remaining 26 binary floating point instructions
- IPL Clear, System Reset, and System Reset Clear operator commands
- Pentium 4 optimizations enabled in GCC

12.10 Changes for Hercules Emulator Release 3.01.0

Release date: November 30, 2003

- Bypass gcc 2.96 optimizer bug that caused incorrect instruction execution
- Added command-line control panel command history
- Message security assist
- Fixed device interrupt pending on IPL that caused OS/360 to have IPLed twice
- Added pthreads trace function for debugging
- Fish threads code rewritten, closer to POSIX thread functionalitiy, while still performing better
- Fixed incompatibility with Windows NT telnet client
- Performance and integrity enhancements for RS instructions

12.11 Changes for Hercules Emulator Release 3.00.0

Release date: October 02, 2003

- Dynamically loaded module support for devices, instructions and operator console panels
- Shared and remote DASD support
- ALS4 (z/990) instruction support
- Simplified network adapter specifications
- New device emulations: 2703, 3410, 3490, 9347
- ECPS:VM support
- Reworked process priority handling
- Greatly improved interval timer resolution
- Internal consistency checking improvements
- Corrected 3270 session disconnect processing
- Instruction disassembler in control panel
- Tape read backward fixes
- Fix for double memory consumption bug on Windows
- OMA tape processing fixes
- Message logging restructuring
- S/370 I/O race condition fixes
- Manual pages for some commands

12.12 Changes for Hercules Emulator Release 2.17.1

Release date: February 12, 2003

- Corrected RPM installed files permissions
- Corrected dasdload verbosity level
- Corrected card reader eof / intrq option handling, added * to designate no file loaded

- Correct SLB instruction condition code
- Fix dasdutil.c track conversion function

12.13 Changes for Hercules Emulator Release 2.17.0

Release date: February 01, 2003

- Restructured DASD subsystem, better use of memory, compressed FBA support, framework for shared DASD
- New dasdcopy utilitiy replaces ckd2cckd, and cckd2ckd and adds compressed FBA support
- Native support for Mac OS X 10.2 and above
- Reworked CTC and LCS emulation
- SMP host integrity fixes
- Fixes for compile errors with gcc 3.x
- S/370 dual address space and MVS assist fixes
- Renumbered all messages to consistent format, removed duplicate numbers, started message documentation
- Added options for 1052/3215 consoles and card readers
- Numerous instruction and I/O emulation fixes

12.14 Changes for Hercules Emulator Release 2.16.5

Release date: July 08, 2002

- Corrected serious CCKD image file corruption error
- Allow tape files to be opened for input if on CD-ROM

12.15 Changes for Hercules Emulator Release 2.16.4

Release date: July 03, 2002

- Read backward support for emulated tape
- Added 9313, 9332 and 9335 to list of supported devices

12.16 Changes for Hercules Emulator Release 2.16.3

Release date: July 02, 2002

- CTC fix for TurboLinux bug
- 3287 printer support via TN3270
- S/370 extended memory fixes
- Ctcadpt.c compilation fix for FreeBSD
- Fixed 3270 ERASE ALL UNPROTECTED command to not count data read
- Fixes to ckdtab in dasdtab.c

- Retrofitted cckd chkdsk fixes/enhancements
- FBA fixes
- Compatibility fixes for CCKD and Hercules 2.17

12.17 Changes for Hercules Emulator Release 2.16.2

Release date: May 20, 2002

- Fixed 3350 dasdtab entry
- Fixed 370 interval timer error
- Control panel attach command bug fixed

12.18 Changes for Hercules Emulator Release 2.16.1

Release date: May 04, 2002

- fthreads locking fixes
- dasdload bug fix
- FBA DASD devices allow any size disk
- Control panel attach command bug fixed
- Windows versions finally accessible from main page

12.19 Changes for Hercules Emulator Release 2.16.0

Release date: April 20, 2002

- PER support
- S/370 multiprocessor support
- Licensed software restriction
- Performance modifications
- Interrupt subclass priorities
- dasdcat program
- Updated TCP/IP documentation
- CTCI support for windows
- Print to unix pipe
- Preliminary Lan Channel Station (LCS) support
- HTTP server
- Various fixes

12.20 Changes for Hercules Emulator Release 2.15.0

Release date: December 04, 2001

- Autoconf added to ease portability
- Numerous instruction fixes
- TUN/TAP support for Linux kernels beyond 2.4.6
- Timer fixes
- Synchronous I/O
- Support for IPL from CD-ROM as with HMC
- CTC hang at shutdown fixed
- CTC TCP/IP now works with VM/ESA
- Compressed CKD endianness and RAS fixes
- Hot reader support
- Machine checks now reported for host exceptions, loops, and wait states

12.21 Changes for Hercules Emulator Release 2.14.0

There was no release 2.14.0

12.22 Changes for Hercules Emulator Release 2.13.0

Release date: July 05, 2001

- Restrict TODEPOCH to 1900, 1928, 1960, 1970 or 1988 and correct offset calculation
- HET unmount option
- Quiet command
- Panel instruction disassembly
- CMPSC corrections
- CTCT CTC over TCP/IP
- Sundry instruction and channel fixes
- Numerous instruction fixes
- CKD trace command
- Performance enhancements
- CGEBR/CGDBR instructions
- CEGBR/CDGBR instructions
- CKD 9345 support
- Storage Key Assist
- Move Page Facility 2

12.23 Changes for Hercules Emulator Release 2.12.0

Release date: May 04, 2001

- Numerous instruction fixes
- FBA and CKD read-only support
- Enable ISKE/RRBE/SSKE in S/370 mode
- CCKD corrections
- CMPSC fixes for expansion
- Correct prefix alignment for ESA/390 guest in 64-bit mode SIE
- Card reader multiple files and EBCDIC autopad support
- Support for built-in TUN driver of Linux kernel 2.4.x
- Device I/O thread throttling
- Small optimization of vstore/vfetch and TPI
- Sense/Set Path Group ID for DASD
- Dynamic device threads
- Fast interrupt processing for MCK and PER
- Allow HET-files to reside on read-only media
- Utilities display versioning and copyright info
- Present device end on terminating console session
- sh panel command
- 9221 power-off diagnose
- Debug format enhancements
- Fix for device threads
- Sundry new ESAME instructions and corrections
- Improved interrupt processing
- Incorrect-Length-Indication-Suppression facility
- S/370 timer interval fixes
- 64-bit Interpretative Execution
- IEEE floating point
- 64-bit panel updates
- LPM fixes and display subchannel command
- Fix amode 64 in load_psw
- Multiply Logical instructions
- Environment variables to override filenames of hercules.rc, hercules.cnf and hercifc
- Floating point enhancements
- Country codepage tables

12.24 Changes for Hercules Emulator Release 2.11.0

Release date: February 09, 2001

- Sundry new ESAME instructions and corrections
- Panel display instruction operands
- TRAP and RP instructions
- TP instruction
- Tape data chaining patch
- Bypass Cygwin stack problem
- Fixes for Windows port
- SSK/ISK/RRB fix for 2K storage keys
- Extended Translation Facility 2
- Divide Logical instructions

12.25 Changes for Hercules Emulator Release 2.10.0

Release date: February 02, 2001

- z/Architectur support
- TUN/TAP support for CTC
- OSTAILOR VSE option
- 2K/4K storage key support
- Fully functional CMPSC instruction
- Fix read-only AWSTAPE
- Sundry new ESAME instructions
- Format-2 2K/4K IDAW
- ESAME 5-level DAT
- ESAME ASN authorization and ALET translation
- ESAME linkage-stack instructions
- ESAME subspace replacement
- ESAME DUCT format changes
- Unloaded tape drive support
- Extended floating point
- Divide Single instructions
- EPSW instruction
- Compressed CKD updates
- Timer update correction
- Fix MVCLE instruction
- Interval Timer fix

12.26 Changes for Hercules Emulator Release 1.71.0

Release date: January 18, 2001

- Compressed CKD DASD release 2 with improved performance, shadow file support and better reliability
- Hercules Emulated Tape (HET) format support
- Make HET bzip2 compression optional, analogous to CCKD bzip2
- Fix for track overflow record zeroing
- Clarified licensing discussions in FAQ
- Treat printer X'37' CCW as NOP
- Treat X'E503' MVS/XA assist instruction as no-op
- Read commands from hercules.rc at startup
- New tapelist program prints contents of 80-byte record tapes
- Increased MAXDBLK from 3000 to 40000 and MAXATTR from 10000 to 40000 in dasdload

12.27 Changes for Hercules Emulator Release 1.70.0

Release date: December 03, 2000

- New file hercwin32.zip contains build scripts for Win32 versions
- More performance enhancements
- ALS-1 and ALS-2 support completion
- Extended translation facility
- Pick up correct float.c module
- Distribute windows binaries as well as Linux binaries
- Fix orienting bug in CKD DASD search CCW processing
- Obtain TOD clock lock when accessing or updating 370 interval timer
- Change license to the QPL Open Source Definition compliant license

12.28 Changes for Hercules Emulator Release 1.69.0

Release date: October 29, 2000

- Correct AXR and SXR instruction results when significance exception raised
- Correct CD and CDR instruction condition code logic
- Do not generate support for square root instructions in 370 mode
- Floating point arithmetic tuning
- Performance optimization fixes
- Spelling corrections
- Fixed version number

12.29 Changes for Hercules Emulator Release 1.68.0

Release date: October 08, 2000

- Rewritten and updated FAQ
- Compressed CKD DASD support
- Many performance improvements
- DASD I/O optimizations
- Simplified building on non-Intel architectures
- Fix for random bug in MP instruction
- Treat all 3505 card reader read CCWs the same

12.30 Changes for Hercules Emulator Release 1.67.0

Release date: September 04, 2000

- Win32 portability changes
- Fix for 64K segment length checking in 370 DAT
- Fix for TPI storing interrupt code when no interrupt pending
- Skip to channel 9 and channel 12 support
- Panel refresh rate speedup and command
- Fix storage protection override on fetch
- SIE support with S/370 and ESA/390 modes and vector support
- Bugfix for MXR instruction
- CONCS, DISCS and RCHP instructions
- Fix flags on intermediate subchannel status
- Break SYSCONS output lines when too long
- Floating point instructions SQDR and SQER
- Lock Page instruction

12.31 Changes for Hercules Emulator Release 1.66.0

Release date: August 03, 2000

- Simplify logmsg and DEVTRACE macro definitions
- Prevent incorrect length indication on CONTROL NOP CCW
- Complete 370 HIO processing
- Correct nullification of TPI and TSCH
- Add device locking to MSCH
- Correct TPROT instruction
- Correct address wrapping on assist instructions
- Change interrupt logic to use longjmp on all interrupts

- Clear remainder of ASTE when loading ASTE with ASF=0 in translate_asn
- Add (incomplete) PLO instruction
- Fix CLCL interruption problem
- Fix address wrap in MVO
- Make ED and EDMK perform a trial run
- Fix address wraparound in MVO
- Fix CR15 corruption in form_stack_entry, fix nullification in form_stack_entry and unstack_registers
- Fix loss of interrupts in PR

12.32 Changes for Hercules Emulator Release 1.65.0

Release date: July 22, 2000

- Track overflow processing fixes
- Added TOD clock update to STCK, STCKE DIAG 204 and TRACE processing
- Fixed READ DEVICE CHARACTERISTICS alternate track values for 3380 and 3390
- Skeletal CMPSC instruction
- Added support for 3340 and 3375 DASD
- Corrected interval timer update increment
- float.c optimization for new instruction decode and execution
- Fix program check on TIC CCW
- Fix program check on NOP CCW
- Instruction decode and execution restructure
- Added fomit-frame-pointer to compiles for improved performance
- Fix STCKE instruction

12.33 Changes for Hercules Emulator Release 1.64.0

Release date: July 04, 2000

- Added track overflow processing for CKD DASD
- Makefile change to allow RPM building with RPM_BUILD_ROOT
- Added NetBSD build definitions to makefile
- Moved version definition to version.h and removed makefile dependency for source modules
- Package change, tarball now explodes into hercules<version> subdirectory
- Fix backward going TOD clock
- Suppress superfluous HHC701/HHC702 messages
- Rework cpu.c to decode instructions by macro
- Bypass bug in IBM telnet client

12.34 Changes for Hercules Emulator Release 1.63.0

Release date: June 18, 2000

- 3270 CCW processing improvements
- OSTAILOR generalization and new pgmtrace panel command
- VM IUCV instruction correction and DIAGNOSE improvements
- CPU timer and clock comparator improvements
- 3480 READ BLOCK ID and LOCATE CCW support
- Networking support via virtual CTCA
- Restructured CPU execution by function call instead of switch statement
- Support for IEBCOPY sequential datasets in dasdload
- New dasdls command lists the VTOC of a CKD DASD volume
- New AWSTAPE handling commands: tapesplt, tapemap
- MAKE INSTALL target to install in /usr/bin

12.35 Changes for Hercules Emulator Release 1.62.0

Release date: June 03, 2000

- Still more multiprocessor improvements
- Dynamic CPU reconfiguration
- Basic vector facility
- Floating point version 6
- READ AND RESET BUFFERED LOG CCW (X'A4')
- WRITE SPECIAL CKD CCW (X'01')
- FBA DASD model reporting fixes

12.36 Changes for Hercules Emulator Release 1.61.0

Release date: May 21, 2000

- More multiprocessor improvements
- New startall / stopall panel commands
- STIDP stores processor address in first digit of CPU id
- Correction to IPTE instruction for S/370
- Dummy HIO instruction for S/370
- Support for emulated 0671 FBA DASD
- FBA device reserve/release CCW support
- New OSTAILOR configuration option allows selective suppression of program check messages

12.37 Changes for Hercules Emulator Release 1.60.0

Release date: May 14, 2000

- Multiprocessor locking improvements
- Machine check and channel report word
- New attach/detach/define commands to allow dynamic addition and deletion of devices from the configuration
- Compare and Swap and Purge (CSP) instruction
- Broadcasted purging
- Fix LASP instruction SASN authorizing using wrong AX, if bits 29-31 are 010 and SASN ¬= PASN
- Fix SAC instruction special operation exception, setting secondary space mode when ASF=0
- Remove intdrag option and replace drag comand by toddrag command
- New extpending flag to improve performance
- Allow longer host name in console connected message
- Floating point version 5 including fixes

12.38 Changes for Hercules Emulator Release 1.59.0

Release date: April 30, 2000

- Missing interrupt after CSCH instruction
- S/370 DAT support
- Tape device sense bytes improvements
- Read Buffered Log (CCW X'24') for tape devices
- Reject Sense ID CCW for 3420 tape devices
- Suppress unprintable characters in HMC messages
- Suppress attention interrupt if subchannel not enabled
- New interrupt drag factor to improve performance
- New toddrag and intdrag configuration options and drag control panel command allow drag factors to be set
- Light optimizations on CPU critical path
- Eliminate fetch protection override in S/370 mode

12.39 Changes for Hercules Emulator Release 1.58.0

Release date: April 22, 2000

- Support for CKD DASD volumes exceeding 2GB, such as 3390 model 3
- 3274-1D SELECT RB/RMP/WRT commands
- Support for 3270 14-bit SBA addressing and inbound SFE order
- Command reject if Write Structured Field CCW issued to a 3270 without extended attributes

- Fix missing CSW_IL indication when CCW count exhausted
- Do not set unit exception if CCW count is zero
- Suppress space switch event program check messages
- Branch tracing and cross memory tracing for BALR, BASR, BASSM, BAKR, BSA, BSG, SSAR, PC, PT and PR instructions
- New diagnose instruction to stop CPU
- Drag factor option slows down TOD clock to decrease overhead on very slow machines
- Correction to PR instruction
- Correction to LASP instruction
- Make CLCLE/MVCLE/CKSM instructions conditional features
- Enable channel measurement mode
- Modifiy program_check() to handle shadow registers correctly
- Change DAT to favour PSTD in TEA to give reduction in page fault path length
- Avoid clearing registers at CPU reset
- Leave GPR, AR and FÜR intact during CPU reset for SADUMP
- Zeroize field for called space identification in PC stack entry
- New CCW X'8D' (Write Update Key and Data) required by STOW
- Fix for 0B7 abend in "D M=CHP" command
- Floating point version 4 including fixes
- Fix incorrect second operand address in MVCIN instruction
- Correct sign of result in SRP instruction
- Erase Gap (CCW X'17') for tape devices
- Activate MIPS counter on control panel
- Suppress tracing of ISK, SCK and DP instructions

12.40 Changes for Hercules Emulator Release 1.57.0

Release date: March 30, 2000

- Fix program check 0032 due to wrong stack entry being updated
- Fix wrong SSTD loaded by LASP instruction
- Bypass main storage lock in single CP configuration
- Fix incorrect condition code in PGIN instruction
- Corrections to expanded storage instructions
- New STCPS and SCHM instructions
- Set more appropriate sense bytes for tape errors

12.41 Changes for Hercules Emulator Release 1.56.0

Release date: March 28, 2000

- Fix incorrect unit exception on SCSI tape FSB/BSB CCW
- Fix unit check on AWSTAPE write
- Close SCSI tape after tape is ejected
- Detect tapemark during SCSI tape FSB/BSB CCW
- Suppress HMC response prompt
- Expanded storage support
- Move Page Facility 2
- Correct signed length error in MVCK/MVCS/MVCP
- Undetected cc=3 in SRP instruction
- Wrong remainder in DP instruction when dividend is less than divisor
- Specification exception in DP instruction should have higher priority than data exception

12.42 Changes for Hercules Emulator Release 1.55.0

Release date: March 22, 2000

- FBA minidisk support
- Additional diagnose functions
- Allow real storage frames to be marked unusable

12.43 Changes for Hercules Emulator Release 1.54.0

Release date: March 18, 2000

- Address wraparound improvements
- Floating point version 3
- Correction to SLDA/SRA instructions
- Recognize tabs and end-of-file character in ASCII cardrdr files
- Hercules-specific diagnose instructions
- Correct missing timer interrupt when interval timer goes from zero to negative
- Enable HMC system console
- Correct sign propagation in multiply instruction
- Reduce CPU thread priority

12.44 Changes for Hercules Emulator Release 1.53.0

Release date: March 01, 2000

- Add BSF/FSF/BSB/FSB CCW support for tape devices

- Allow final short block in OMA fixed block files
- Allow processing of read-only AWSTAPE files and SCSI tapes
- Skeleton ctcadpt module for future 3088 support
- Correctly nullify IC/NI/OI/XI/CLM/STCM/ICM/TRT instructions on page translation exception
- Improved floating point support
- Correct shift result when shift count exceeds 31
- Fix incorrect MVCL cc=3 when destination length is 1

12.45 Changes for Hercules Emulator Release 1.52.0

Release date: February 19, 2000

- Prevent incorrect length indication on 3270 Select CCW
- 2K storage protection for S/370
- prevent wait for console port
- Allow keyword parameters in configuration file
- New SYSEPOCH and TZOFFSET parameters
- Adjust TRACE and DIAG204 for extended TOD
- Set TOD clock in STCK instruction

12.46 Changes for Hercules Emulator Release 1.51.0

Release date: February 15, 2000

- 3270 read buffer fix for OS/360 NIP
- Floating Point instructions
- Remove 32-bit pointer dependency from dasdload for Alpha
- HMC system console support
- Correct condition code after decimal overflow
- Set reference and change bits for PSA access
- New CRLF option for printer and card punch

12.47 Changes for Hercules Emulator Release 1.50.0

Release date: February 10, 2000

- Remove interval timer debugging message
- Fix hung console resulting from attention interrupt fix in release 1.49
- Seek and Set Sector (CCW=27) for Itel 7330 DASD controller
- Correct SIGP handling of non-existent CPUs
- Extended TOD clock bit in processor features
- Alternate control panel help text

- Card reader end-of-file option
- Card reader ASCII/EBCDIC auto-detection
- Fix SIGP RESTART to target correct CPU
- Allow VTOC size and location to be specified for dasdload

12.48 Changes for Hercules Emulator Release 1.49.0

Release date: February 05, 2000

- Alternate control panel
- Present attention interrupt when console connects
- Fix dasdload CVOL logic
- Fix dasdload initialization of empty PDS
- Allow device size to be specified for dasdload
- Add dummy Set Clock instruction

12.49 Changes for Hercules Emulator Release 1.48.0

Release date: January 31, 2000

- Fix dasdload to handle note lists (prevents 32D abends)
- I/O interrupt performance enhancements
- Correctly detect overflow in signed Add/Subtract instructions
- Fix track overflow problem
- 3270 Read Modified CCW

12.50 Changes for Hercules Emulator Release 1.47.0

Release date: January 23, 2000

- Allow tn3270 or telnet client to connect to a specific device number
- Align control panel instruction counter
- Ensure panel display does not corrupt TEA
- STIDP incorrectly propagates high order bit of CPU model
- Fix byte-ordering problem with CKD DASD header on non-Intel machines
- STIDC instruction
- Extended TOD clock (STCKE and SCKPF instructions)
- 3211 Load FCB and Diagnostic Read CCW
- 3270 Read Buffer CCW
- Fix console.c to inhibit input while console has status pending

12.51 Changes for Hercules Emulator Release 1.46.0

Release date: January 11, 2000

- HSCH instruction
- SIGP instruction
- Suppress tracing of page faults
- Display control registers and access registers after program check
- Add regs parameter to program_check function calls
- New panel command to perform store status function
- Suppress tracing of CCW file protect and end of cylinder errors

12.52 Changes for Hercules Emulator Release 1.45.0

Release date: January 08, 2000

- Make MVCL/CLCL interruptible
- Diagnose 204
- Read Channel Subsystem Info
- Fix incorrect register count in TRACE instruction
- Correct nullification of STM/LM/LAM/STAM/STCTL/LCTLSTCM and SS instructions whose operands cross a page boundary
- Suppression on Protection with Virtual-Address enhancement
- Select correct address space for MVCS/MVCP
- Correct registers after CLCL/CLCLE with non-zero condition code
- Defer clock comparator interrupt while instruction stepping
- Remove 32K limit on data chained write CCWs for non-CKD devices
- Correct overrun error on data chained write for FBA DASD

12.53 Changes for Hercules Emulator Release 1.44.0

Release date: January 01, 2000

- Support for 9336 FBA DASD
- Read Replicated Data command for FBA DASD
- Prevent recursive program check after instruction fetch error
- Operand tracing for MVCL/CLCL and RRE instructions

12.54 Changes for Hercules Emulator Release 1.43.0

Release date: December 27, 1999

- New control panel command devlist
- Write Update Data (X'85') CCW for CKD devices

- Makefile changed to use $(cc) instead of cc
- Fix dat.c to prevent ASN translation specification exception (program check X'0017') if subspace group facility is installed and ASN is one
- Fix cpu.c to clear ILC before fetching instruction to prevent PSW being backed up if access error occurs during instruction fetch
- Correct program check ILC when instruction is nullified
- Obtain CPU model number for STIDP from configuration file
- Prevent wait after devinit
- Open printer with O_SYNC to ensure buffers flushed
- Fix xmem.c to prevent loop in program_call when loading 4-word ETE
- Improved TLB lookup

12.55 Changes for Hercules Emulator Release 1.42.0

Release date: December 16, 1999

- New makefile builds both S/370 and ESA/390 executables
- 3480 Set Path Group Id and Unassign CCWs
- CFC and UPT instructions
- Card punch support
- Erase (X'11') CCW for CKD devices
- Correct setting of translation exception address
- Correct file mode when opening printer file
- Correct condition code for shift arithmetic instructions

12.56 Changes for Hercules Emulator Release 1.41.0

Release date: December 07, 1999

- Set reference and change bits correctly for main storage accesses by channel, dat, xmem, stack, block and service modules
- New devinit command
- Reject control panel virtual storage display command if CR1=0
- Fix dasdload to correctly write EOF record for empty file and correctly fill block overhead fields in format4 DSCB
- Diagnose functions MSSFCALL and SCPEND
- Corrections to service.c and assist.c
- Alpha platform portability definitions
- 3480 Assign CCW

12.57 Changes for Hercules Emulator Release 1.40.0

Release date: November 30, 1999

- New DASDISUP program performs OS/360 IEHIOSUP function
- Correct SCSW handling for suspend/resume
- Forward space file CCW for tape devices
- 3480 load display CCW and sense path group id CCW
- Fix handling of OMA tape headers to correctly recognize tape mark and to align headers to 16-byte boundary
- EBCDIC character translation of CCW data displays
- Fix command reject for CKD read commands outside the domain of a locate record

12.58 Changes for Hercules Emulator Release 1.39.0

Release date: November 24, 1999

- Concurrent sense
- I/O initial status interruption
- Channel program suspend/resume function and RSCH instruction
- Read Device Characteristics CCW for 3480
- Fix incorrect command reject on Sense Subsystem Status CCW
- Increase 2370 write buffer size to prevent console I/O error when using Zap function of ZZSA
- Fix error in dat.c causing wrong bytes to be fetched or stored when operand crosses page boundary
- Remove temporary fix to ckddasd.c introduced in version 1.37

12.59 Changes for Hercules Emulator Release 1.38.0

Release date: November 22, 1999

- New panel commands to allow storage alteration
- Fix incorrect I/O parameter on attention interrupt
- Clear PMCW correctly during I/O reset
- Change 3270 control unit type to 3274-1D
- Fix restart command broken by version 1.37

12.60 Changes for Hercules Emulator Release 1.37.0

Release date: November 19, 1999

- Storage range display
- EBCDIC character translation of storage displays
- New breakpoint command

- Messages go to log file as well as screen if stdout is redirected
- Fix missing interrupt caused by channel.c failing to obtain device lock before setting interrupt pending
- Fix incorrect condition code 1 in attention SCSW built by console.c
- New Read Channel Path Information service call
- Temporary fix to ckddasd.c multitrack search
- Addition of Read Device Characteristics and Sense Subsystem Status commands for CKD devices
- New DASDPDSU program to unload PDS members from a CKD volume

12.61 Changes for Hercules Emulator Release 1.36.0

Release date: November 12, 1999

- Clear subchannel instruction
- Correct fault causing control panel display corruption

12.62 Changes for Hercules Emulator Release 1.35.0

Release date: November 09, 1999

- Improved control panel user interface
- New control panel commands: start, stop, restart, ipl, loadparm
- New loadcore command to load disk image files
- S/370 interval timer
- Allow 31-bit mode linkage in locking instructions
- Support for PCI in ESA/390 mode as well as S/370 mode
- Correct problem causing false channel protection checks

12.63 Changes for Hercules Emulator Release 1.34.0

Release date October 29, 1999

- New DASDLOAD program to create a CKD volume from unloaded PDS files
- Correct CKD module to prevent record not found error on multitrack Read Count CCW

12.64 Changes for Hercules Emulator Release 1.33.0

Release date: October 26, 1999

- Write support for SCSI tapes and AWSTAPE files
- Correct handling of REWIND command for AWSTAPE files
- Correct bug in Subtract logical instruction
- Ensure unique TOD clock values for Store Clock

- Correction to unstacking process for PR instruction
- Implementation of Read Multiple CKD command

12.65 Changes for Hercules Emulator Release 1.32.0

Release date: October 18, 1999

- Support for virtual tapes in OMA (Optical Media Attach) format
- SCSI tape support (read-only)
- Minor corrections to CKD DASD support

13. Hercules Windows GUI

13.1 Introduction

The Hercules Windows GUI provides a graphical interface to the Hercules Emulator itself. Hercules has only a semi-graphical user interface, the Windows GUI gives a full graphical user interface. Working with native Hercules one has to deal with many batch-oriented or line-command utilities, whereas with the Windows GUI these utilities are directly usable from the GUI itself without having to know the exact command-line syntax.

The Windows GUI also helps in creating and maintaining the configuration files and helps working with the Hercules log files. The Hercules Windows GUI is also known under the short names HercWinGUI or HercGUI. The Hercules Windows GUI is written and maintained by David B. Trout, known as “Fish”.

The following figure shows the main panel of the Hercules Windos GUI.

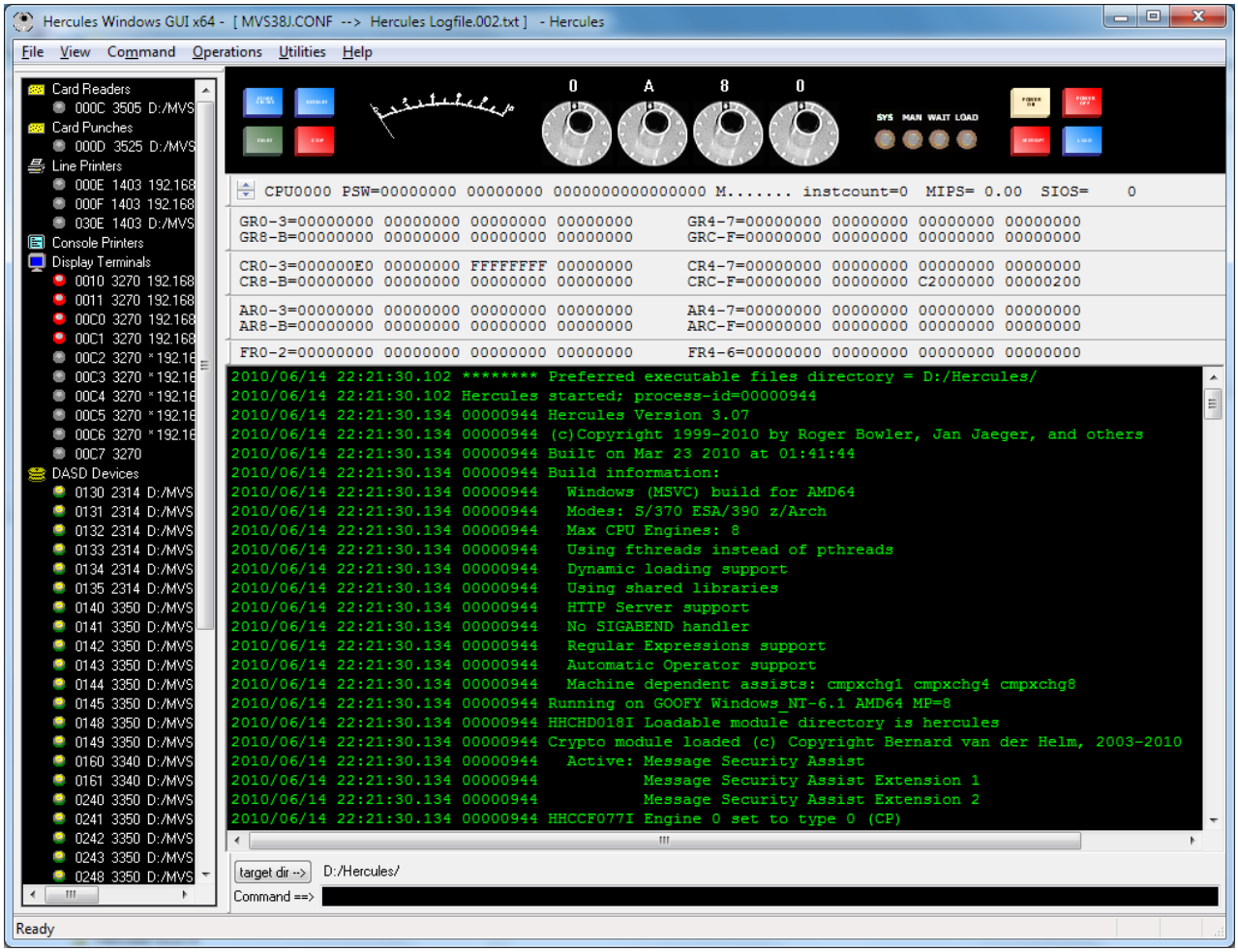


Figure 19: Hercules Windows GUI Main Panel

13.2 Windows GUI Source Code

The source code for the HercGUI is no longer available due to the ongoing effort to try and commercialise the software. Although the source was available earlier the HercGUI was never open source. It was and still is copyrighted intellectual property.

13.3 Hercules Windows GUI Evolution

This documentation is based on the Hercules Windows GUI Release 1.10.0 (Build 4890). The following sections briefly list the changes for all HercGUI releases back to version 1.1.3 (Build 2559).

13.4 Changes for Hercules Windows GUI Release 1.11.1 (Build 5265)

Release date: March 01, 2007

- Added support for print-to-pipe to "Add printer" dialog.
- Various bug fixes.

13.5 Changes for Hercules Windows GUI Release 1.11.0 (Build 5239)

Release date: February 24, 2007

- Converted to Visual Studio 2005 with 64-bit support.
- Redesigned the "System Configuration" dialog to support new Hercules configuration statements.
- Updated the "Display / Alter Storage" dialog to use FishLib HexEdit control.
- Updated all menu handlers to new version of BCMenu.
- Added new "Beep when utility is done" preference option.
- Added 3390-54 DASD support.
- Added support for displaying 64-bit registers.
- Various bug fixes.

13.6 Changes for Hercules Windows GUI Release 1.10.1 (Build 4909)

Release date: October 04, 2006

- Add 3-pixel left margin to console display.
- Add code to help increase chances of "Oops!" Crash Dump dialog automatically appearing for the rare case of whenever a Hercules crash should occur (needs Hercules V3.05.0 or later!)
- Changed behaviour of 'Stop' and 'Start' buttons to always issue 'STOPALL' or 'STARTALL' command regardless of whether NUMCPU is greater than one or not.
- Various bug fixes.

13.7 Changes for Hercules Windows GUI Release 1.10.0 (Build 4890)

Release date: August 16, 2006

- Added support for new YROFSET statement.
- Added support for optional YROFFSET argument on SYSEPOCH statement.
- Added some new code pages to listbox on Misc/Other tab of System Configuration dialog.
- Log file now opened in shared mode instead of exclusive mode, so it can be viewed via Notepad for example, while the Hercules GUI is still up and running.
- Removed support for CKD2CCKD and CCKD2CKD utilities, which both have long been replaced by the DASDCOPY utility.
- Added support for the new PANTITLE statement (silently ignored).
- Added support for the '-r' raw init option to DASDINIT (skips writing VOL1 label).
- Removed 'conspawn' from distribution in favour of Hercules's version.
- Added support (by user request) for changing the System Status and Registers bars fonts.
- Added support for Windows XP visual styles.
- The System Status bar now contains a spinner control allowing quick and easy selection of which CPU's information should be displayed.
- Various bug fixes.

13.8 Changes for Hercules Windows GUI Release 1.9.5 (Build 4734)

Release date: December 22, 2005

- HercGUI now logs which preferred executables directory is being used at Power On.
- The default Logging Preference option is now "Automatic".
- New Misc2 Preference option to suppress manual edit reminder warning message box.
- Button added to command-line bar to set 'sh'ell command target directory.
- Complete redesign of Systen Configuration dialog.
- Added support for some new Hercules configuration file statements.
- Moved the "Modify settings" and "Modify devices" into the "File" menu.
- Added support for new DASDCONV utility.
- Tweak Percent Utilization Meter handling.
- File "cpu-types.txt" updated and corrected.
- CTCI/CTCT device statements now use the 'new' format (devtype 3088 deprecated).
- Added support for new 1052-C and 3215-C Integrated Console Printer devices.
- Add support to TAPECOPY utility dialog for copying SCSI tape to or from AWS files.
- The Devices Configuration dialog now remembers its previous size and position.
- Improved restoration handling of main window's previous size and position.
- Updated some existing minimum, maximum and default "registry tweak" values.
- Various bug fixes.

13.9 Changes for Hercules Windows GUI Release 1.8.15 (Build 4316)

Release date: August 15, 2005

- Save and restore previously used device-type and volser in DASDINIT dialog.
- Various bug fixes.

13.10 Changes for Hercules Windows GUI Release 1.8.8 (Build 4207)

Release date: December 25, 2004

- Various bug fixes.

13.11 Changes for Hercules Windows GUI Release 1.8.6 (Build 4202)

Release date: December 22, 2004

- Various bug fixes.

13.12 Changes for Hercules Windows GUI Release 1.8.5 (Build 4200)

Release date: December 20, 2004

- Configuration file description length limitation removed.
- Most DASD device dialogs now auto-detect the actual Hercules compressed/uncompressed CKD/FBA DASD type.
- CTCI and LCS devices now require both even/odd addresses to be defined.
- Tape device dialogs now accept SCSI device file filenames.
- AUTO_SCSI_MOUNT configuration file statement accepted.
- HETINIT (initialize tape) dialog: VOLSER spinner now supports partially numeric VOLSERs (e.g. if VOLSER is RDF001, the spinner will auto-increment the ‘001’ portion).
- HercGUI windows title no longer shows full path.
- Added new groupname, ipaddr/mask and noprompt parameters support to terminal display device configuration dialogs.
- The system and/or device configuration can now be modified, while Hercules is up und running.
- New FORMAT advanced logging options: date, time, and process-id.
- Display/Alter memory dialog now has an address input field to allow to enter the specific address.
- New HercGUI and Hercules help menus to display distributed HTML documentation.
- Console messages with embedded CR’s (carriage-returns) now handled more gracefully.
- Various bug fixes.

13.13 Changes for Hercules Windows GUI Release 1.6.8 (Build 3981)

Release date: unknown

- Various bug fixes.

13.14 Changes for Hercules Windows GUI Release 1.6.6 (Build 3910)

Release date: unknown

- Rearranged controls in the HETINIT utility dialog.
- Added support for new Hercules version 3.0 CPU Version code statement.
- Added support for new Hercules version 3.0 device-range device statement format.
- Various bug fixes.

13.15 Changes for Hercules Windows GUI Release 1.6.4 (Build 3772)

Release date: unknown

- Added support for new the new 'maxsize', 'eotmargin' and 'readonly' tape options.
- Added support for the new 3.0 DASDCOPY utility.
- Added a new "Save Configuration File As..." command to the File menu.
- Completely redesigned the "Add New CTC Device" dialog.
- Modified the dialog centering logic.
- Added "support" for recognizing (but not otherwise processing or validating in any way) the new control file statements introduced in Hercules version 3.0.
- Added compressed FBA support to the DASDINIT utility dialog.
- The main System Configuration dialog now has a checkbox for enabling shared device support.
- 3.0 Shared Device support: The "Add New DASD Device" dialog now has a 'Remote?' checkbox for shared device support.
- Various bug fixes.

13.16 Changes for Hercules Windows GUI Release 1.6.0 (Build 3438)

Release date: unknown

- Added silent acceptance support for new control file statements as well as device statements for unknown/unsupported device-types.
- Added "Load Tape" and "Unload Tape" to the Device List's right-click context menu for tape devices.
- Added "unknown device types" category to Device List pane.
- Added a new "Edit Configuration File" menu selection to the File menu to invoke Notepad for the currently opened configuration file.
- New help menu entry ("Open HercGUI readme.html").
- Modified the 'Ctrl+Left/Right-Arrow' and 'Escape' key keyboard key handling.
- New "Special command-line arguments" preference setting.
- Put back support for the standard MFC status bar that was originally removed long ago.
- Various bug fixes.

13.17 Changes for Hercules Windows GUI Release 1.5.0 (Build 3290)

Release date: unknown

- Support for new CCKD control file statement.
- Support for new HTTPORT/HTTPROOT control file statements.
- Add "Ignore" button to Configuration File Parse Errors dialog.
- Added "Load Parm" field to IPL dialog.
- New "Re-open configuration file" menu selection.
- Slightly improved CPU utilization meter accuracy.
- Other various minor technical fixes/enhancements.
- Various bug fixes.

13.18 Changes for Hercules Windows GUI Release 1.4.0 (Build 3164)

Release date: unknown

- Make all device dialogs slightly wider.
- Groupbox all dialogs.
- Add spinner to DASDINIT size field and other dialogs where a numeric value is specified.
- DASDLS: auto-add filename to list.
- Update CTCA dialog for CTCI-W32.
- Shrink control panel images if display resolution is 800 x 600 so they fit on the screen.
- Utilities: make necessary DASDINIT utility changes and add support for new DASDCAT utility.
- Support for new Hercules parameters.
- Device statement edit: remove highlight and trim trailing blanks.
- Configurable (via registry) various upper/lower range values.
- System config dialog: selectable list of known CPU types controlled by "cpu-types.txt" file in preferred Configuration Files directory.
- Handling of PGMPRDOS control file statement.
- Advanced logging options: wrap-around logfile; memory and disk limits.
- Changed "about" box.
- Device List panel right-click menu: display subchannel status and start / stop tracing.
- Cygwin issue fixed by adding or modifying a registry entry value for message "HHC020I Cannot obtain xxx MB of main storage: Not enough memory".
- Various bug fixes.

13.19 Changes for Hercules Windows GUI Release 1.3.0 (Build 2769)

Release date: unknown

- Command-line arguments now supported.

- Add socket device support for card reader.
- Preference option to ignore configuration file parse errors.
- Device Configuration dialog: Right-click menu to manually edit device statement.
- CFCCIMAGE support removed.
- Hercules Version 1 support dropped.
- Change in locking logic for better performance on multi-processor systems.
- Partial CTCA support.
- Add icons and bitmaps to menus.
- Prevent shutting down GUI if Hercules or any utility processes are still running.
- Preference option for exit of GUI when Hercules is powered off.
- Assign reasonable identifying titles to the Font & Color dialogs.
- New Hercules console logging options.
- Add "Don't ask again" checkbox to IPL dialog. Reset whenever configuration has changed.
- Add support for new multifile reader option.
- Merge all preference dialogs into one multi-tabbed property sheet dialog.
- Tweaked color of PowerOn button to look more realistic.
- Various bug fixes.

13.20 Changes for Hercules Windows GUI Release 1.2.2 (Build 2573)

Release date: unknown

- Various bug fixes.

13.21 Changes for Hercules Windows GUI Release 1.1.3 (Build 2559)

Release date: unknown

- The last control file used is now remembered across executions.
- New "Close control file" menu command.
- Add *.jcl to reader file filter.
- New preferences dialog to specify your preferred file extensions for all Open / Save dialogs.
- Card reader devinit settings are now remembered across executions.
- It is now possible to add one line of descriptive text as a comment to the control file.
- The main window position and size is now remembered and restored across executions.
- Support for the new Hercules DEVTMAX control file statement.
- Support for the new Hercules Version 2.12 card reader.
- Information regarding the version of the GUI is also written to the console log file.
- Various bug fixes.

14. Hercules Studio

14.1 Introduction

The Hercules Studio provides a graphical user interface under Linux to the Hercules Emulator itself. Hercules has only a semi-graphical user interface, the Windows Studio gives a full graphical user interface. The Hercules Studio is the Linux counterpart to the Hercules Windows GUI described in the previous sections.

Hercules Studio is written and maintained by Jacob Dekel. See Appendix A for download links.

The following figure shows the main panel of the Hercules Studio.

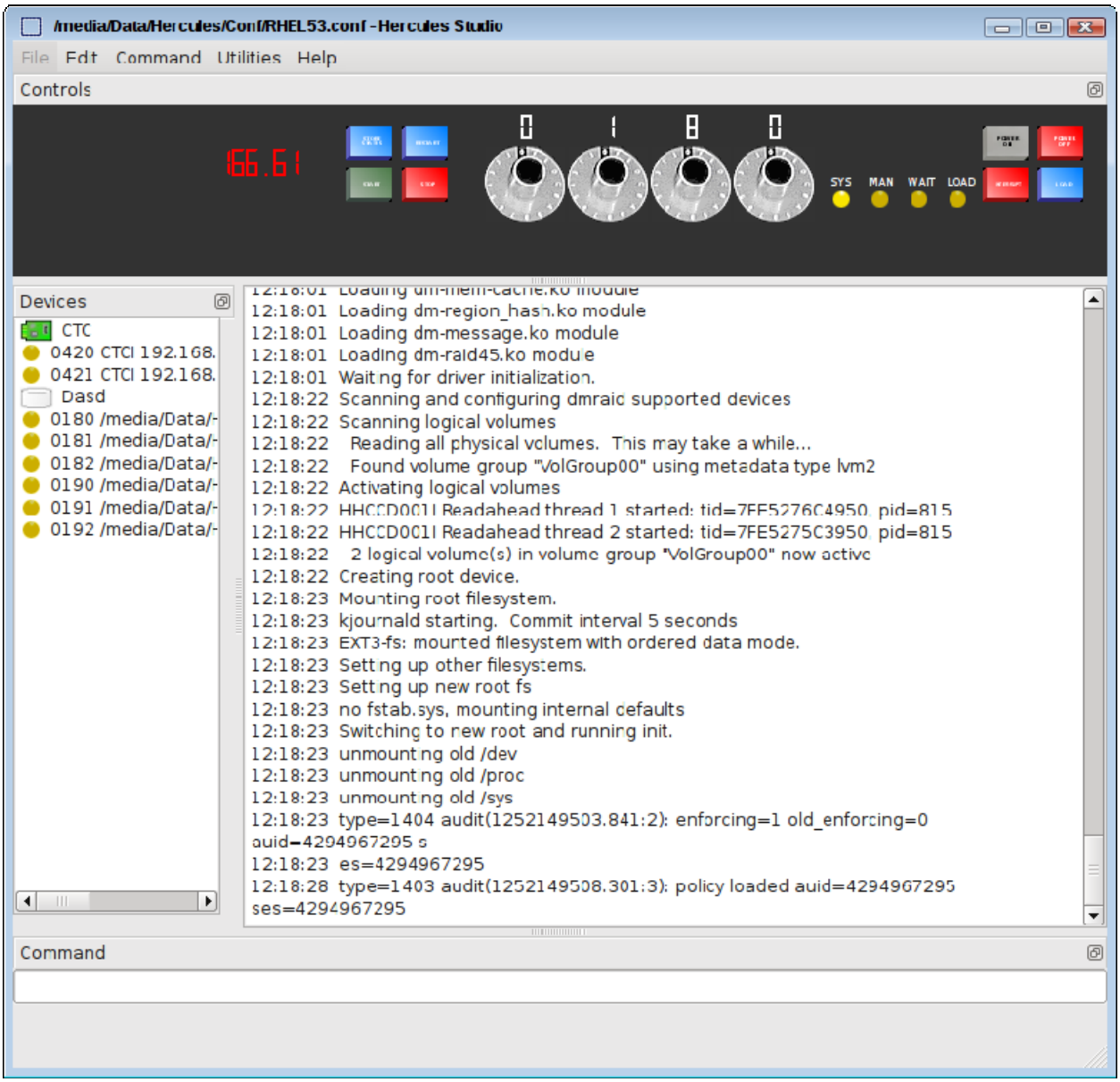


Figure 20: Hercules Studio Main Panel

14.2 Hercules Studio Source Code

The source code for the Hercules Studio is available, the source repository can be accessed with Subversion (SVN) like follows:

"svn co https://hercstudio.svn.sourceforge.net/svnroot/hercstudio/trunk/HerculesStudio"

The INSTALL file located in the source root directory contains complete instructions on building and installing Hercules Studio form source.

15. WinPcap Packet Capture Driver

Figure 21: WinPcap Logo

15.1 WinPcap Packet Capture Driver Introduction

WinPcap is an architecture for packet capture and network analysis for the Win32 platform, developed at Politecnico di Torino in Italy. The packet filter is a device driver that adds to Windows 95, 98, ME, NT, 2000 and XP the ability to capture and send raw data from a network card with the possibility to filter and store the captured packets in a buffer.

WinPcap includes an API that can be used to directly access the functions of the packet driver, offering a programming interface independent from the Windows operating system. It also exports a set of high level capture primitives that are compatible with libpcap, the well known Unix capture library. These functions allow a program to capture packets in a way independent from the underlying network hardware and operating system.

WinPcap is a free, public system and is released under a BSD-style license. It can be downloaded from www.winpcap.org, where the necessary documentation can also be found.

The sections about the WinPcap internals (chapters 15.5 to 15.8.4) have been taken from the original WinPcap documentation with the kind permission of the WinPcap team. Many thanks to Loris Degoianni.

15.2 What does WinPcap

WinPcap is a system for direct network access under Windows. Most networking applications access the network through widely used system primitives such as sockets. This approach allows easily transfer data on a network as the operating system copes with low level details (protocol handling, flow reassembly, etc.) and provides an interface similar to the one used to read and write to a file.

Sometimes however this “easy” way is not adequate. Raw access to the network without the intermediation of entities such as protocol stacks is required.

The purpose of WinPcap is to provide this level of access to Win32 applications. It provides facilities to:

- Capture raw packets, both the ones destined for the local machine and packets exchanged by other hosts (on shared media)
- Filter the packets according to user-specified rules, before dispatching them to the applications
- Transmit raw packets to the network
- Gather statistical values in the network traffic

This set of capabilities is obtained by means of a device driver that is installed inside the networking portion of the Win32 kernels plus some DLLs. All these features are exported through a programming interface, easily exploitable by the applications and portable to different operating systems.

15.3 What kind of programs use WinPcap

WinPcap can be used by different kind of tools for network analysis, troubleshooting, security and monitoring. In particular, classic tools that rely on WinPcap are:

- Network and protocol analyzers
- Network monitors
- Traffic loggers
- Traffic generators
- User-level bridges and routers
- Network intrusion detection system (NIDS)
- Network scanners
- Security tools
- Hercules Emulator

15.4 What WinPcap cannot do

WinPcap receives and sends the packets independently from the protocols of the host, like TCP/IP. This means that it is not able to block, filter or manipulate the traffic generated by other programs on the same machine. It simply sniffs the packets that transit on the wire. Therefore it cannot be used by applications like traffic shapers, QoS schedulers and personal firewalls.

15.5 WinPcap Internals (Overview)

WinPcap is an architecture for packet capture and network analysis for the Win32 platform. It includes a kernel-level packet filter, a low-level dynamic link library (packet.dll), and a high-level and system independent library (wpcap.dll).

Why is WinPcap an architecture rather than just a library? This is, because packet capture is a low level mechanism that requires a strict interaction with the network adapter and with the operating system, in particular with its networking implementation, so a simple library is not sufficient.

First, a capture system needs to bypass the protocol stack in order to access the raw data transiting on the network. This requires a portion running inside the kernel of OS, interacting directly with the network interface drivers. This portion is very system dependent, and in WinPcap it is realized as a device driver, called Netgroup Packet Filter (NPF). WinPcap currently is provided in different versions of the driver for Windows 95, Windows 98, Windows ME, Windows NT 4, Windows 2000, Windows XP and Windows Vista. These drivers offer both basic features like packet capture and injection, as well as more advanced ones like a programmable filtering system and a monitoring engine. The first one can be used to restrict a capture session to a subset of the network traffic (e.g. it is possible to capture only the ftp traffic generated by a particular host), the second one provides a powerful but simple to use mechanism to obtain statistics on the traffic (e.g. it is possible to obtain the network load or the amount of data exchanged between two hosts).

Second, the capture system must export an interface that user-level applications will use to take advantage of the features provided by the kernel driver. WinPcap provides two different libraries: *packet.dll* and *wpcap.dll*.

The first one offers a low-level API that can be used to directly access the functions of the driver, with a programming interface independent from the Microsoft OS.

The second one exports a more powerful set of high level capture primitives that are compatible with libpcap, the well known Unix capture library. These functions allow to capture packets in a way independent from the underlying network hardware and operating system.

The following figure shows the various components of WinPcap:

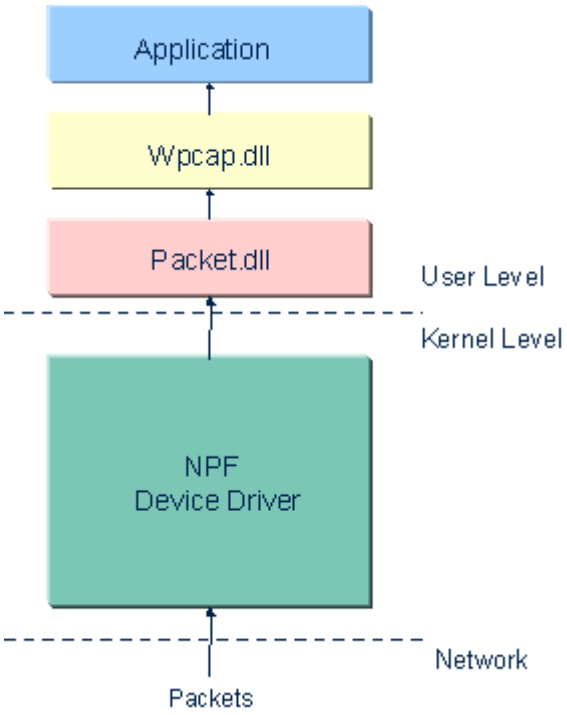


Figure 22: WinPcap Architecture

15.6 WinPcap Internals (Details)

The following sections document the internals of the Netgroup Packet Filter (NPF), the kernel portion of WinPcap. Normal users are probably interested in how to use WinPcap and not in its internal structure. Therefore the information present in this module is destined mainly to WinPcap developers and maintainers, or to the people interested in how the driver works. In particular, a good knowledge of OSes, networking and Win32 kernel programming and device driver development is required to profitably read this section.

NPF is the WinPcap component that does the hard work, processing the packets that transit on the network and exporting capture, injection and analysis capabilities to user-level.

The following paragraphs will describe the interaction of NPF with the OS and its basic structure.

15.7 NPF and NDIS

NDIS (Network Driver Interface Specification) is a standard that defines the communication between a network adapter (or, better, the driver that manages it) and the protocol drivers (that implement for example TCP/IP). Main NDIS purpose is to act as a wrapper that allows protocol drivers to send and receive packets onto a network (LAN or WAN) without caring either the particular adapter or the particular Win32 operating system.

NDIS supports three types of network drivers:

- **Network interface card or NIC drivers**. NIC drivers directly manage network interface cards, referred to as NICs. The NIC drivers interface directly to the hardware at their lower edge and at their upper edge present an interface to allow upper layers to send packets on the network, to handle interrupts, to reset the NIC, to halt the NIC and to query and set the operational characteristics of the driver. NIC drivers can be either miniports or legacy full NIC drivers.

  Miniport drivers implement only the hardware-specific operations necessary to manage a NIC, including sending and receiving data on the NIC. Operations common to all lowest level NIC drivers, such as synchronization, is provided by NDIS. Miniports do not call operating system routines directly; their interface to the operating system is NDIS. A miniport does not keep track of bindings. It merely passes packets up to NDIS and NDIS makes sure that these packets are passed to the correct protocols.

  Full NIC drivers have been written to perform both hardware-specific operations and all the synchronization and queuing operations usually done by NDIS. Full NIC drivers, for instance, maintain their own binding information for indicating received data.

- **Intermediate drivers**. Intermediate drivers interface between an upper-level driver such as a protocol driver and a miniport. To the upper-level driver, an intermediate driver looks like a miniport. To a miniport, the intermediate driver looks like a protocol driver. An intermediate protocol driver can layer on top of another intermediate driver although such layering could have a negative effect on system performance. A typical reason for developing an intermediate driver is to perform media translation between an existing legacy protocol driver and a miniport that manages a NIC for a new media type unknown to the protocol driver. For instance, an intermediate driver could translate from LAN protocol to ATM protocol. An intermediate driver cannot communicate with user-mode applications, but only with other NDIS drivers.

- **Transport drivers or protocol drivers**. A protocol driver implements a network protocol stack such as IPX/SPX or TCP/IP, offering its services over one or more network interface cards. A protocol driver services application-layer clients at its upper edge and connects to one or more NIC driver(s) or intermediate NDIS driver(s) at its lower edge.

The next figure shows the position of NPF inside the NDIS stack:

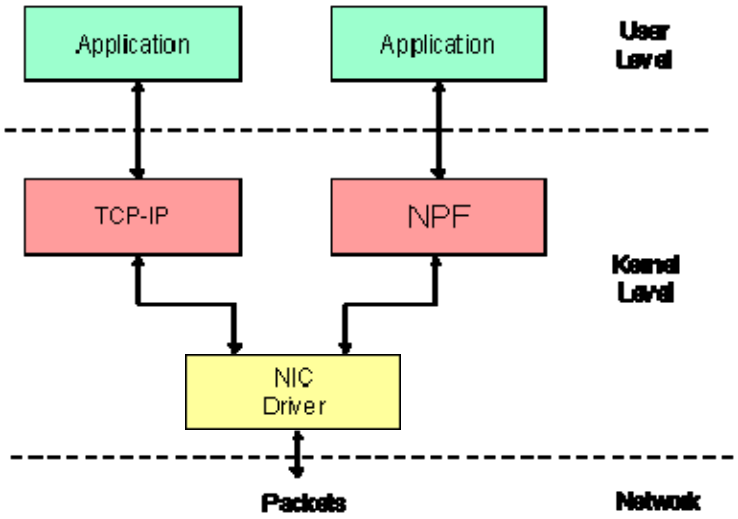


Figure 23: NPF inside NDIS

NPF is implemented as a protocol driver. This is not the best possible choice from the performance point of view, but allows reasonable independence from the MAC layer and as well as complete access to the raw traffic.

Notice that the various Win32 operating systems have different versions of NDIS: NPF is NDIS 5 compliant under Windows 2000 and its derivations (like Windows XP), NDIS 3 compliant on the other Win32 platforms.

The interaction with the OS is normally asynchronous. This means that the driver provides a set of callback functions that are invoked by the system when some operation is required to NPF. NPF exports callback functions for all the I/O operations of the applications: open, close, read, write, ioctl, etc.

The interaction with NDIS is asynchronous as well: events like the arrival of a new packet are notified to NPF through a callback function (Packet_tap() in this case). Furthermore, the interaction with NDIS and the NIC driver takes always place by means of non blocking functions: when NPF invokes a NDIS function, the call returns immediately; when the processing ends, NDIS invokes a specific NPF callback to inform that the function has finished. The driver exports a callback for any low-level operation, like sending packets, setting or requesting parameters on the NIC, etc.

15.8 NPF Structure Basics

The next figure shows the structure of WinPcap, with particular reference to the NPF driver:

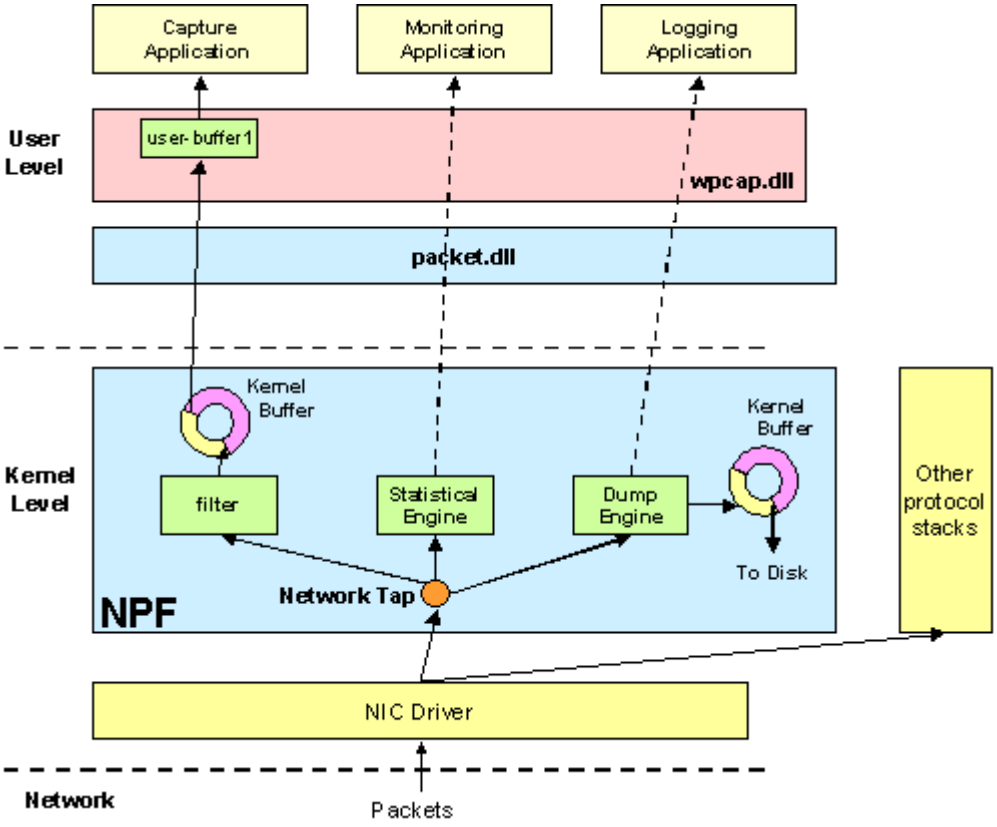


Figure 24: NPF Device Driver

NPF is able to perform a number of different operations: capture, monitoring, dump to disk, packet injection. The following paragraphs will describe shortly each of these operations.

15.8.1 Packet Capture

The most important operation of NPF is packet capture. During a capture, the driver sniffs the packets using a network interface and delivers them intact to the user-level applications. The capture process relies on two main components:

- A packet filter that decides if an incoming packet has to be accepted and copied to the listening application. Most applications using NPF reject far more packets than those accepted, therefore a versatile and efficient packet filter is critical for good over-all performance. A packet filter is a function with boolean output that is applied to a packet. If the value of the function is true the capture driver copies the packet to the application; if it is false the packet is discarded. NPF packet filter is a bit more complex, because it determines not only if the packet should be kept, but also the amount of bytes to keep. The filtering system adopted by NPF derives from the BSD Packet Filter (BPF), a virtual processor able to execute filtering programs expressed in a pseudo-assembler and created at user level. The the application takes a user-defined filter (e.g. “pick up all UDP packets”) and, using wpcap.dll, compiles them into a BPF program (e.g. “if the packet is IP and the *protocol type* field is equal to 17, then return true”). Then, the application uses the *BIOCSETF* IOCTL to inject the filter in the kernel. At this point, the program is executed for every incoming packet, and only the conformant packets are accepted. Unlike traditional solutions, NPF does not *interpret* the filters, but it *executes* them. For performance reasons, before using the filter NPF feeds it to a JIT compiler that translates it into a native 80x86 function. When a packet is captured, NPF calls this native function instead of invoking the filter interpreter, and this makes

the process very fast. The concept behind this optimization is very similar to the one of Java jitters.

- A circular buffer to store the packets and avoid loss. A packet is stored in the buffer with a header that maintains information like the timestamp and the size of the packet. Moreover an alignment padding is inserted between the packets in order to speed-up the access to their data by the applications. Groups of packets can be copied with a single operation from the NPF buffer to the applications. This improves performances because it minimizes the number of reads. If the buffer is full when a new packet arrives, the packet is discarded and hence it's lost. Both kernel and user buffer can be changed at runtime for maximum versatility: packet.dll and wpcap.dll provide functions for this purpose.

The size of the user buffer is very important because it determines the *maximum* amount of data that can be copied from kernel space to user space within a single system call. On the other hand, it can be noticed that also the *minimum* amount of data that can be copied in a single call is extremely important. In presence of a large value for this variable, the kernel waits for the arrival of several packets before copying the data to the user. This guarantees a low number of system calls, i.e. low processor usage, which is a good setting for applications like sniffers. On the other side, a small value means that the kernel will copy the packets as soon as the application is ready to receive them. This is excellent for real time applications (like, for example, ARP redirectors or bridges) that need the better responsiveness from the kernel. From this point of view, NPF has a configurable behavior that allows users to choose between best efficiency or best responsiveness (or any intermediate situation).

The wpcap library includes a couple of system calls that can be used both to set the timeout after which a read expires and the minimum amount of data that can be transferred to the application. By default, the read timeout is 1 second, and the minimum amount of data copied between the kernel and the application is 16K.

15.8.2 Packet Injection

NPF allows to write raw packets to the network. To send data, a user-level application performs a WriteFile() system call on the NPF device file. The data is sent to the network as is, without encapsulating it in any protocol, therefore the application will have to build the various headers for each packet. The application usually does not need to generate the FCS because it is calculated by the network adapter hardware and it is attached automatically at the end of a packet before sending it to the network.

In normal situations, the sending rate of the packets to the network is not very high because of the need of a system call for each packet. For this reason, the possibility to send a single packet more than once with a single write system call has been added. The user-level application can set, with an IOCTL call (code pBIOCSWRITEREP), the number of times a single packet will be repeated: for example, if this value is set to 1000, every raw packet written by the application on the driver's device file will be sent 1000 times. This feature can be used to generate high speed traffic for testing purposes: the overload of context switches is no longer present, so performance is remarkably better.

15.8.3 Network Monitoring

WinPcap offers a kernel-level programmable monitoring module, able to calculate simple statistics on the network traffic. The idea behind this module is shown in Figure 2: the statistics can be gathered without the need to copy the packets to the application that simply receives and displays the results obtained from the monitoring engine. This allows to avoid great part of the capture overhead in terms of memory and CPU clocks.

The monitoring engine is made of a *classifier* followed by a *counter*. The packets are classified using the filtering engine of NPF, that provides a configurable way to select a subset of the traffic. The data that pass the filter go to the counter, that keeps some variables like the number of packets and the amount of

bytes accepted by the filter and updates them with the data of the incoming packets. These variables are passed to the user-level application at regular intervals whose period can be configured by the user. No buffers are allocated at kernel and user level.

15.8.4 Dump to Disk

The dump to disk capability can be used to save the network data to disk directly from kernel mode.

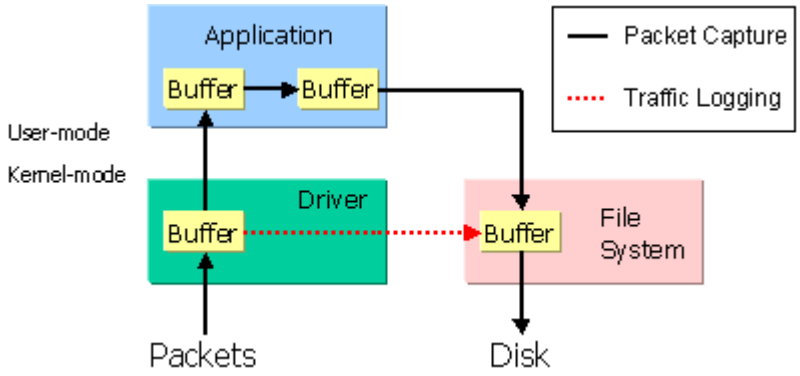


Figure 25: Packet Capture vs. Kernel-Level Dump

In traditional systems, the path covered by the packets that are saved to disk is the one followed by the black arrows in Figure 3: every packet is copied several times, and normally 4 buffers are allocated: the one of the capture driver, the one in the application that keeps the captured data, the one of the stdio functions (or similar) that are used by the application to write on file, and finally the one of the file system.

When the kernel-level traffic logging feature of NPF is enabled, the capture driver addresses the file system directly, hence the path covered by the packets is the one of the red dotted arrow: only two buffers and a single copy are necessary, the number of system call is drastically reduced, therefore the performance is considerably better.

Current implementation dumps the to disk in the widely used libpcap format. It gives also the possibility to filter the traffic before the dump process in order to select the packet that will go to the disk.

16. CTCI-W32

16.1 CTCI-W32 Introduction

Since Hercules runs as a user process under the control of the driving system (Windows in this case), it does not normally have direct access to the driving systems network adapter(s). Until recently this presented a problem in establishing connectivity between the network and the TCP/IP stack of an operating system running under Hercules.

But with CTCI-W32 and WinPcap (described earlier in this book) it is now possible to establish a virtual point-to point link between the TCP/IP stack running under Hercules and Windows' TCP/IP stack. Allowing the use of Windows as a router to pass Ethernet frames between the Hercules TCP/IP stack and the rest of the network as shown in the diagram below:

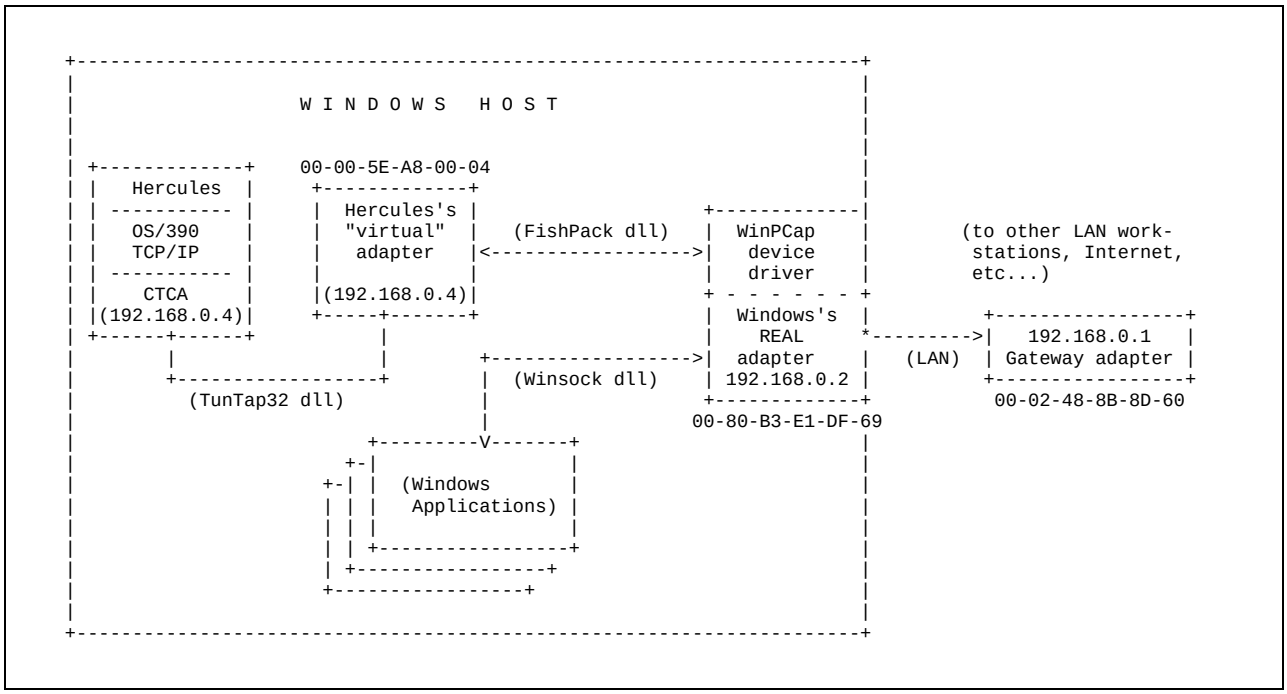


Figure 26: CTCI-W32 Implementation

The TunTap32 dll uses the FishPack dll to send/receive packets on the actual physical adapter via WinPcaps device driver. By responding appropriately to ARP (Address Resolution Protocol) packets the driver makes Windows think there is another physical adapter connected somewhere on the local network. However this adapter is only a virtual one, created simply by having TunTap32 respond to Windows ARP request packets, using a MAC address constructed from Hercules' assigned 'fake' IP address.

Hercules' virtual adapter MAC address is calculated to be 00-00-5E-xn-nn-nn, where xn-nn-nn' is always the last three octets of the IP address assigned to Hercules with the high-order x'80' bit OR'ed on (that's what the 'x' means in 'xn-nn-nn'). Thus in the above example the MAC address of Hercules' virtual adapter would be 00-00-5E-A8-00-04, because the last three octets of Hercules' IP address 192.168.0.4 are '168' ('A8' in hex), '0' and '4'.

TunTap32 dll simply reads packets from, and writes packets to, the real Windows adapter via the FishPack dll. From Windows' point of view these packet come from a real workstation somewhere on the

local network and Windows summariliy does whatever it does with them – which is usually route them to the default gateway and thus to the “outside” world.

When the packets come back to Windows, it sends them back to the adapter they came from (the Hercules’ virtual adapter) by preparing the Ethernet packet with the MAC address of Hercules’ virtual adapter and writing this packet onto the LAN.

The real adapter, which is physically part of the network, sees the packet and asks the WinPcap device driver whether or not it is interested in it. The WinPcap driver, as a packet sniffer, is always interested in all packets and thus saves the packet in its internal device driver buffer.

The next time TunTap32 dll does an I/O to the real adapter (via FishPack dll and via the WinPcap device driver) it receives the packet, examines it and notices that its destination MAC address matches Hercules’ virtual adapter and hence returns the packet to Hercules.

16.2 CTCI-W32 Evolution

This documentation is based on the CTCI-W32 Release 3.2.1. The following sections briefly list the changes for all CTCI-W32 implementation changes, back to FishPack Release 1.0.1 (Build 286), TunTap32 Release 1.0.0 (Build 358) and tt32info Release 1.0.0 (Build 129).

Because the CTCI-W32 package consists in earlier versions of three different components, only changed components are mentioned in the following chapters. If one component is missing in a description of changes, then it has not changed and the last description and version is still current.

Beginning with CTCI-W32 version 3.1.0 all three components are now contained in one package called CTCI-W32 and are always upgraded at the same time.

16.3 Changes for CTCI-W32 V3.2.1 (Build 160)

Release date: March 01, 2007

- Various bug fixes.

16.4 Changes for CTCI-W32 V3.2.0 (Build 156)

Release date: February 24, 2007

- CTCI-W32 is now built using the latest version of Microsoft's Visual C++ product (Visual Studio 2005).
- CTCI-W32 now works with version 4.0 of WinPcap. WinPcap 4.0 fully supports 64-bit Windows systems.
- Support for using CTCI-W32 with 64-bit applications running on 64-bit Windows operating systems. For this a new utility called FWPCUtil is included in the distribution.
- Added support for specifying driver / dll buffer sizes to TT32Test.
- Added "About" dialog box to TT32Test.

16.5 Changes for CTCI-W32 V3.1.7

Release date: October 04, 2006

- TT32Test now no longer automatically adds peer-to-peer route entries to Windows' routing table.
- TunTap32: Fixed a hang that would sometimes occur on Win9x systems.

- Various bug fixes.

16.6 Changes for CTCI-W32 V3.1.6

Release date: August 16, 2006

- All components: Minor cleanup.
- Various bug fixes.

16.7 Changes for CTCI-W32 V3.1.2

Release date: May 06, 2006

- Various bug fixes.

16.8 Changes for CTCI-W32 V3.1.0

Release date: April 18, 2006

- Support for WinPcap V3.1
- If a router is used in the LAN, then it should be no longer necessary to have 'IP Forwarding' registry entry enabled.
- It is no longer necessary to keep 'Checksum Offloading' disabled.
- The 'tt32Test' testing utility has been greatly enhanced.
- The preferred base addresses of all DLLs (TunTap32.dll and FishPack.dll) have been changed in order to maximize the available virtual storage.
- Various bug fixes.

16.9 Changes for TunTap32 V2.1.0.404

Release date: September 11, 2005

- New "tuntap32_open_ex" function to work around the 'errno' clobber issue.
- Various bug fixes.

16.10 Changes for FishPack V1.3.0.323 / TunTap32 V2.0.3.379

Release date: July 14, 2003

- CTCI-W32 now supports the new version 3.0 of WinPcap.
- Allow both ":" and "-" as the MAC address separator.
- Various bug fixes.

16.11 Changes for TunTap32 V2.0.0.367

Release date: November 19, 2002

- LCS (LAN Channel Station) support.

16.12 Changes for FishPack V1.1.0.296 / TunTap32 V1.0.4.375 / tt32info V1.0.2.133

Release date: October 29, 2002

- FishPack modified to auto-create and start the WinPcap NPF kernel driver service.
- Various bug fixes.

16.13 Changes for FishPack V1.0.2.288 / TunTap32 V1.0.2.360 / tt32info V1.0.2.131

Release date: May 04, 2002

- Various bug fixes.

16.14 Changes for FishPack V1.0.1.286 / TunTap32 V1.0.0.358 / tt32info V1.0.0.129

Release date: unknown

- Various bug fixes.

17. TN3270 Client

17.1 TN3270 Introduction

Working with Hercules requires a TN3270 terminal emulation. Although in general every IBM 3270 terminal emulator will work together with Hercules, two emulators are recommended and supported by the Hercules development:

- Vista tn3270 from Tom Brennan for Windows platforms
- x3270 from Paul Mattes for Unix and Windows platforms

The following sections give an introduction to the supported TN3270 terminal emulations.

17.2 Vista tn3270 (Windows Platforms)

Figure 27: Vista tn3270 Logo

Vista tn3270 is a Windows program designed to emulate IBM 3270 terminals connected to a mainframe via IP link. Vista has some unique features designed especially for programmers, such as built-in multiple cut and paste buffers, fully tailorable keyboard, extensive select / copy / paste functions – including "SelectJCL", which can pick out dataset names, parms and other items with a single mouse click.

Vista runs on any current Windows platform and due to its lightweight implementation can even run on older machines with little memory and disk space. Vista uses bitmapped raster fonts for the clearest text possible. There are two font sets in 73 sizes from 4x6 to 16x36 pixels to suite any monitor size and terminal model preferences, either fullscreen or windowed.

Vista tn3270 has among many others the following features:

- TN3270 SNA terminal support, which emulates most functions of 327x terminal models 2,3,4,5, along with user specified screen sizes up to 72 lines by 200 characters wide.
- Up to 26 concurrent host sessions
- Easy reconnect and terminal model alteration
- Fully customizable keyboard
- Customizable toolbar and popup keypads
- IND$FILE file transfer
- Macro/script processing language
- International code page support
- Up to 9 multiple cut/paste buffers

- Auto keyboard unlock
- Type ahead buffer
- Capture screen to a file
- Horizontal, vertical or crosshair ruler
- Screen level undo/redo function
- Find function for finding data on a complex screen
- Screen print with definable settings for each terminal type
- Fully customizable screen colors with brightness control
- Window auto-sized to font
- Smart selection of words on screen using “SelectJCL” function
- Smart replace of words on screen using “PasteJCL” function
- SSL for encrypted sessions

The following picture shows a sample Vista tn3270 window:

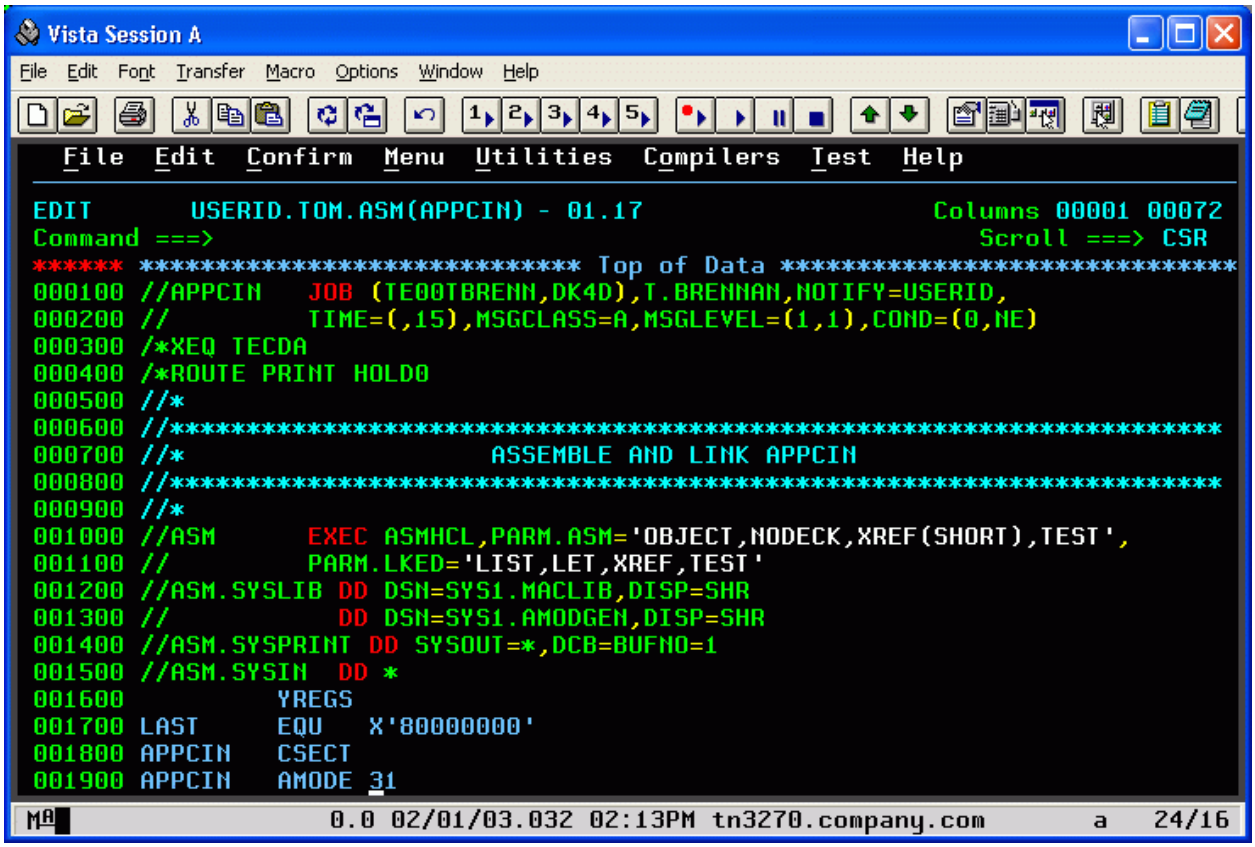


Figure 28: Vista tn3270 Window

Vista is written and maintained by Tom Brennan, an IBM mainframe systems programmer. Originally developed for his own use to overcome the limitations of other terminal emulations, Vista is now widely used amongst mainframe users. A 30 day trial version or the fully licensed version of Vista tn3270 is available from http://www.tombrennansoftware.com.

17.3 x3270 (Unix and Windows Platforms)

Figure 29: x3270 Logo

x3270 is an IBM 3270 terminal emulator for the X Window System and Windows. It runs on most UNIX-like operating systems, e.g. Linux, Mac OS X, Solaris, and Cygwin. It also runs natively on Windows. x3270 runs over a TELNET connection, emulating either an IBM 3279 (color) or 3278 (monochrome) terminal.

The window created by x3270 can use its own font for displaying characters and is an accurate representation of an IBM 3278 or 3279.

x3270 began life as 3270tool, a 3270 emulator for Suntools (Sun's original proprietary windowing environment). 3270tool was developed by Robert Viduya at Georgia Tech. 3270tool was then ported to X11R4 by Jeff Sparkes and given the name x3270. Paul Mattes has been modifying and extending x3270 since version 3.1 in 1993 and is the current maintainer. Over the years a number of people have made significant contributions to x3270.

x3270 supports the following features:

- The full TN3270E protocol
- SSL / TLS (via the OpenSSL library) for encrypted sessions
- APL2 characters
- Non-english character sets, including Russian, Turkish, Hebrew and DBCS Chinese and Japanese
- IND$FILE file transfer
- NVT mode (emulating a color xterm)
- A pop-up keypad for 3270-specific keys
- A scrollbar
- Printer session integration
- Extensive debugging and scripting facilities

The following picture shows a sample x3270 window:

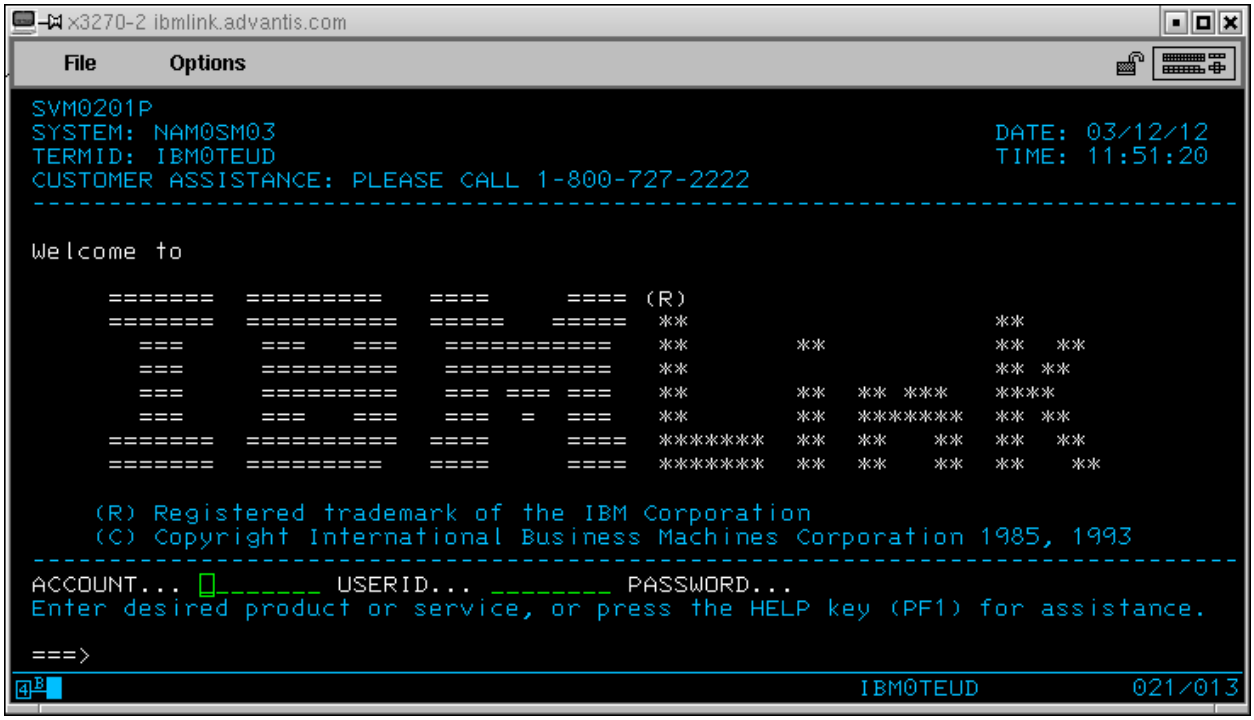


Figure 30: x3270 Window

Please note, that x3270 does not yet support graphics. x3270 is distributed as source code, can be used for free and is downloadable from http://x3270.bgp.nu.

x3270 is available in several different forms:

- x3270 is for use on a graphic display
- c3270 is a curses-based version for use on a dumb terminal (e.g. a serial terminal or a Linux console)
- wc3270 is the Windows console version of c3270
- s3270 is a displayless version for writing screen-scraping scripts
- tcl3270 is similar to s3270, but integrated with Tcl
- pr3287 is for printer emulation
- wpr3287 is the native Windows version of pr3287
- x026 is an IBM 026 keypunch emulator

18. XMIT Manager

Figure 31: XMIT Manager Logo

18.1 Introduction

The XMIT Manager is a Windows based tool that allows for the manipulation of IBM mainframe created Xmit format files. The XMIT manager was developed by Neal Johnston-Ward. With XMIT Manager you can open Xmit files and view or extract the data within them, whether binary or text based. Xmit files with partitioned datasets or sequential dataset contents are dealt with similarly through the graphical interface.

The XMIT Manager is no longer available directly through Neal's website but is now available via the CBT Tape website (www.cbttape.org).

18.2 Xmit File History

The Xmit file format was developed by IBM as a means of packaging various data types into a sequential dataset which then could be transmitted over the network to another system. Xmit is actually a short term for transmit. The transmit feature is part of TSO/E and is called “Interactive Data Transmission Facility”.

Through issuing TRANSMIT commands through TSO/E it is possible to send a dataset or just a message to other users on the system or to users on different systems linked through Network Job Entry (NJE) under JES.

The RECEIVE command is used to receive the Xmit data at the target end. This decodes the sent xmit file and restores the data on the target system in the original format.

18.3 Common Use of Xmit Files

What is of interest to many mainframe users, especially developers and system programmers, is the Xmit format itself. This format can package partioned datasets (PDS) into a sequential file that preserves all the PDS information.

In order to use mainframe Xmit files with XMIT Manager on a Windows platform, the Xmit files must be transferred (FTP or IND$FILE file transfer) as a binary file to the PC.

For more details about the Xmit file format please consult the IBM TSO/E documentation (“TSO/E Customisation”). For information on file formats that may be contained in the Xmit file (such as a PDS or a sequential file) please refer to the IBM DFSMS documentation (“DFSMS Using Datasets”).

18.4 XMIT Manager Functionality

After opening an Xmit file the XMIT Manager shows the contents of the file. The Xmit file is decoded and on the left hand side of the screen (on the tab pages) you can see information about the Xmit file, its content PDS file and the message enclosed (if any). Clicking on each tab brings up the relevant

information about the Xmit file itself and its data. Information is left blank if it was not present in the file or could not be decoded for any reason. The following information can be in the Xmit file:

- **Xmit File –** This panel shows the detals of the Xmit file itself (as a container file) such as size, file name, creation date.
- **Xmit Content –** This panel describes the file that is contained within the Xmit file, such as its size, format and dataset name.
- **Transfer Info** – Contains the Xmit transmit details when the file was created such as the userid that created it, the JES node from which it was sent and the target userid and JES node (if applicable).
- **Message** – If the Xmit file was created with the message option, then the message will be displayed here.

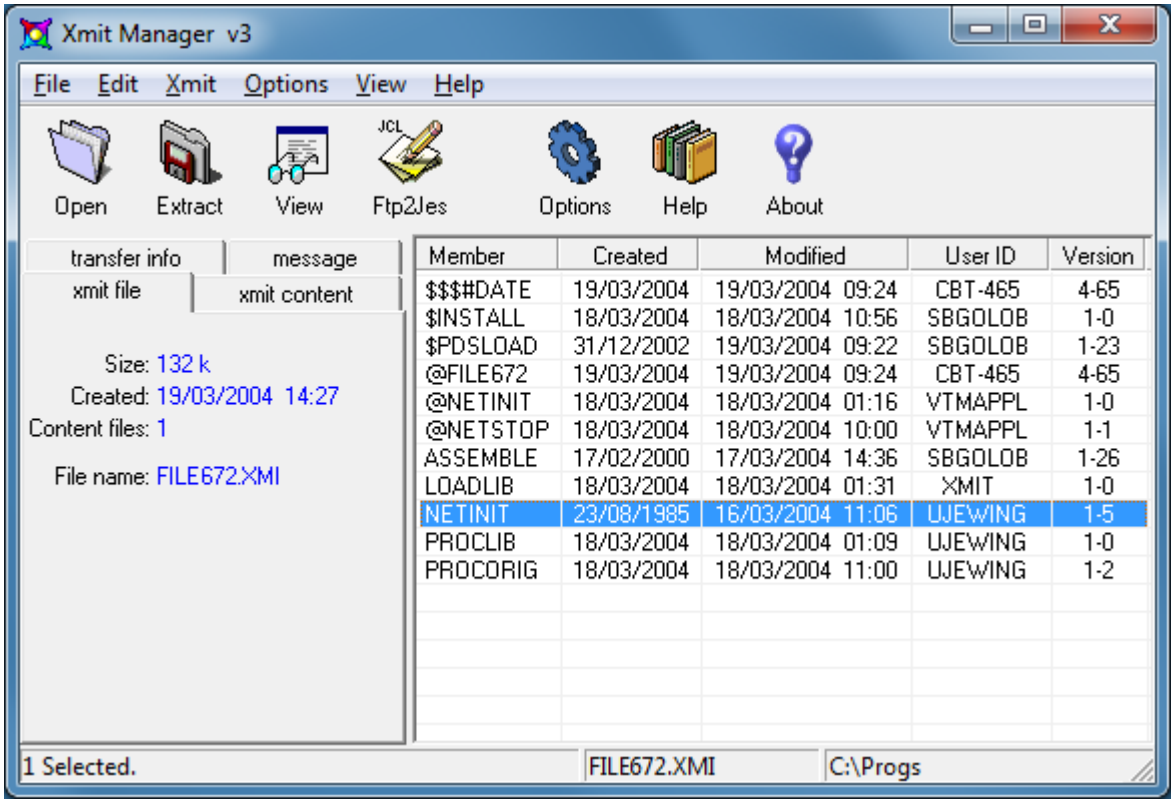


Figure 32: XMIT Manager Main Panel

On the right of the screen the members of any PDS file are displayed. The members can be selected by clicking on them. Double clicking a member will automatically open that member in a view screen. The data contained within the Xmit file can be viewed in a separate window. For sequential datasets the entire file is opened for view in the right side window.

The type of view is determined by the format type of the data to be viewed. Data with record format of fixed (F or FB) will be treated as text and is displayed as ASCII text. Data that has a record format of undefined (U) will be viewed as binary data.

Data selected for viewing will appear in a separate window. This window has some basic menu options such as printing and printer setup, as well as the ability to save the screen data (which is the same as extracting the data to a file).

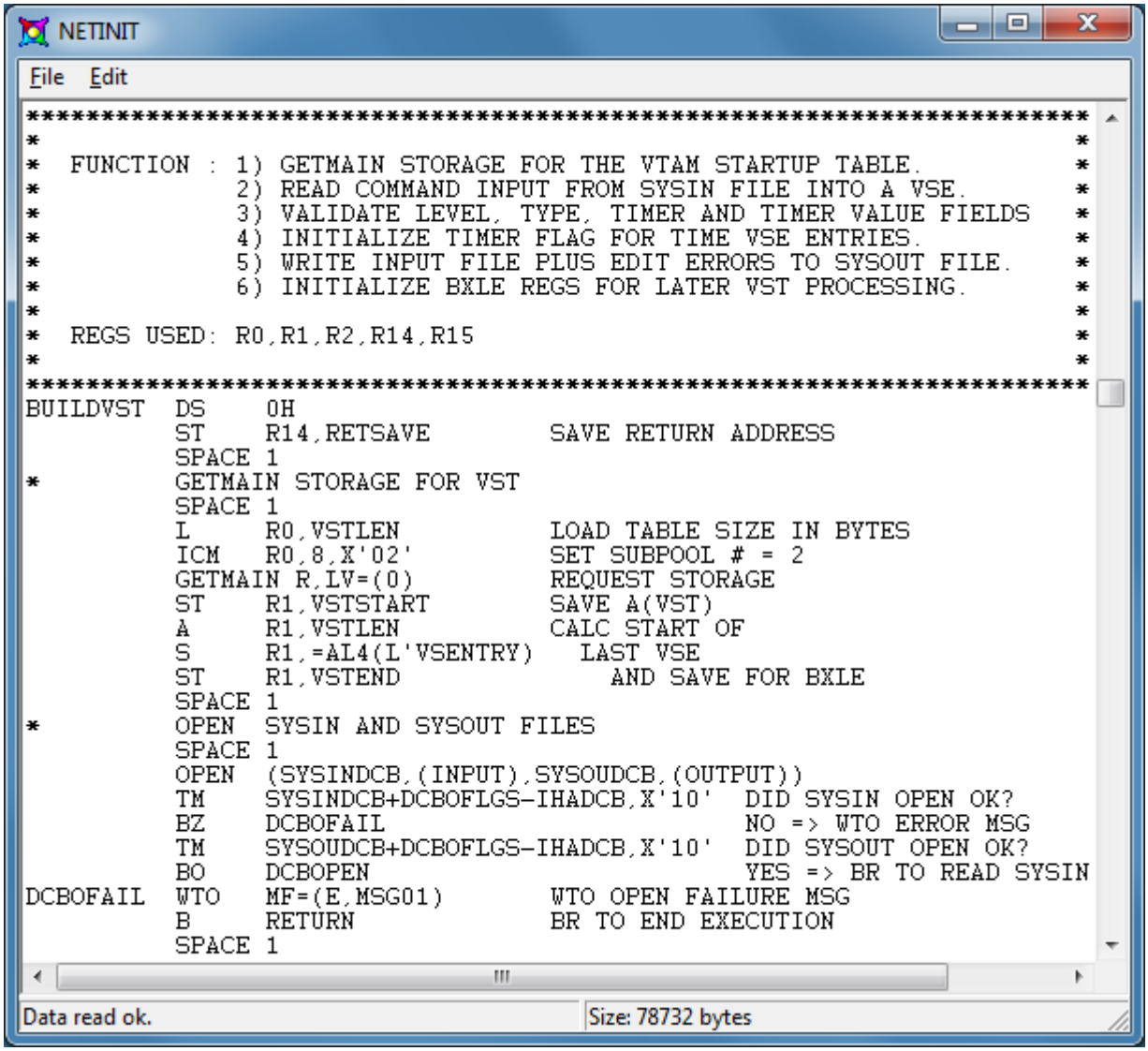


Figure 33: XMIT Manager Member View

The XMIT Manager is very useful for analyzing the contents of an Xmit file (i.e. from the CBT collection) before uploading it to a mainframe system. It is also useful if you only need one or a few members from a large Xmit file and do not wish to transfer the entire file to the mainframe. It is also possible to use Windows based file systems for long term retention of mainframe data in Xmit format.

19. AWS Browse

19.1 Introduction

The AWS Browse Utility can be used to view the contents of emulated mainframe tapes (AWS tapes and HET tapes) from the Windows desktop without having to start a mainframe operating system. There are currently two implementation of AWS browse.

The first (and older one) is from Rob Storey. Because this version does not have a real version identification number and there is currently a newer, improved utility available, we will use version 0.9.x to denote this utility.

The newer implementation of AWS Browse comes from David B Trout (“Fish”) and has more features than the original program. As the version from Rob Storey is no longer supported the following sections provide details only of this newer functionality.

The enhanced AWSBROWSE can be downloaded directly from Fish's website:

http://www.softdevlabs.com/Hercules/hercgui-index.html

19.2 AWS Browse - Original Implementation

This Version of the AWS Browse Utility (recognizable through the filename AWSBROWS.EXE – note the missing “E”!) is the original implementation and was written by Rob Storey. It supports only AWS tape files and has some limitations in its functionality. This implementation can be regarded as “functionally stabilized” and will not be updated in the future.

Because of this and the existence of ongoing bugs in the original implementation, we will go into no further detail about this software and recommend the use of the newer implementation (see section below). Nevertheless this utility has been widely used since the inception of Hercules and deserves credit for enormously easing the use of AWS tapes under Hercules.

19.3 AWS Browse – Enhanced Implementation

This version of the AWS Browse utility (recognizable through the filename AWSBROWSE.EXE – note the additional “E”!) is an enhanced implementation and is written and maintained by David B Trout (“Fish”). This software also supports HET (“Hercules Emulated Tape”) files. It is strongly recommended to use this software in place of older implementations.

19.4 AWS Browse Functionality

The AWS Browse utility provides a display of AWS and HET tape file contents. It supports both ZLIB and BZIP2 compressed file formats.

The tape data can be displayed in EBCDIC and ASCII. The hex display part is user configurable (font as well as layout). The utility also has a hex/text search capability and the clipboard is supported for all kinds of data. Printing and a print preview is also available.

The next figure shows the AWS Browse utility main window, currently browsing an AWS tape file.

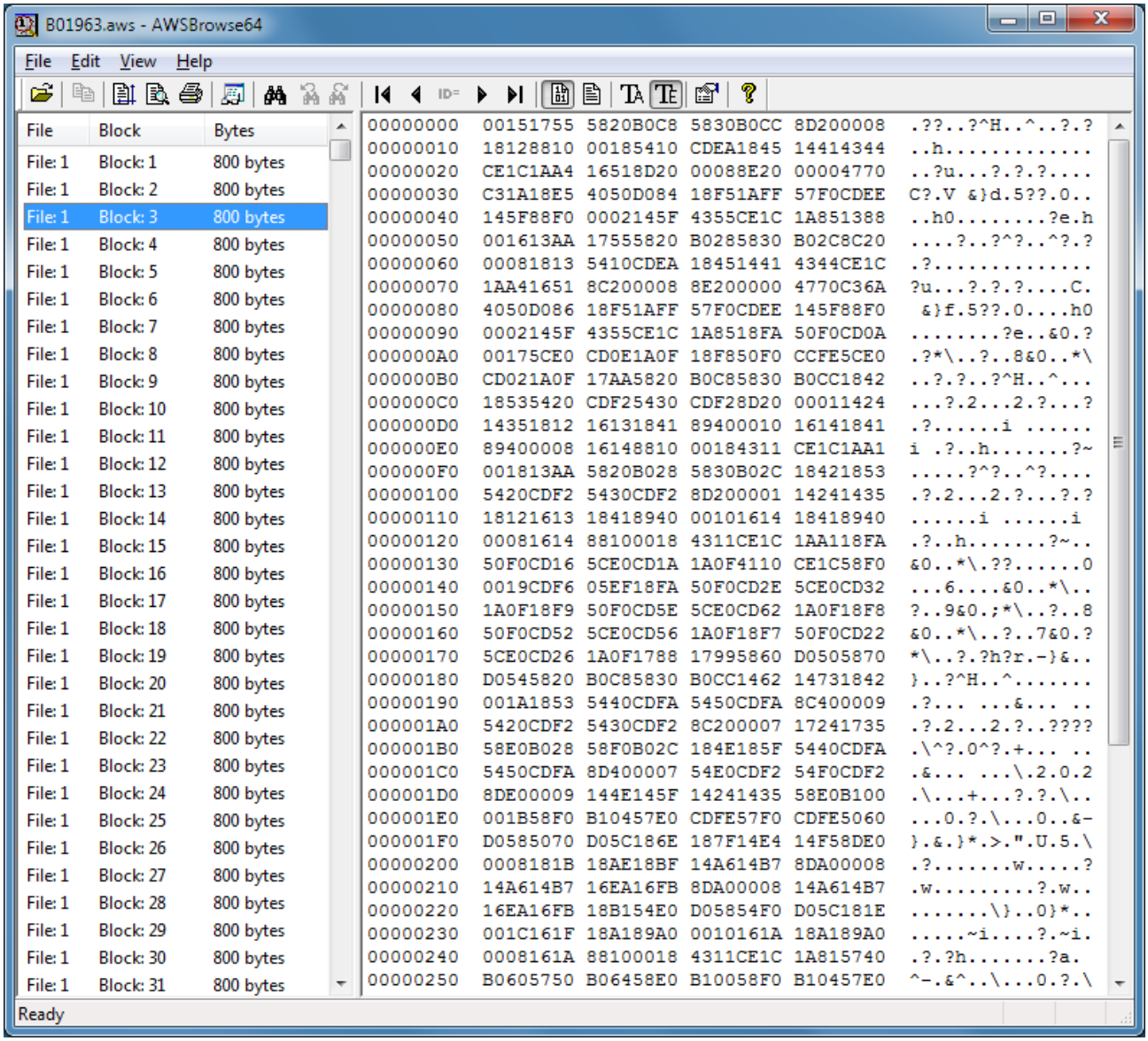


Figure 34: AWS Browse Main Window

19.5 AWS Browse Evolution

This documentation is based on the AWS Browse Utility Release 1.5.1 (Build 1805). The following sections briefly list the changes for all AWS Browse releases back to version 1.2.0 (Build 1278).

19.6 Changes for AWS Browse V1.5.1 (Build 1805)

Release date: March 01, 2007

19.6.1 Fixes

- Fixed Visual Studio 2005 SP1 manifest issue.

19.7 Changes for AWS Browse V1.5.0 (Build 1803)

Release date: February 24, 2007

19.7.1 Fixes

- Minor UI fixes.

19.7.2 Enhancements

- VS2005 and 64-bit support.
- Added "File – Close" command.
- New tape summary report.
- New "Go To" command to jump to start of any standard label file.
- Recognize all Bus-Tech flags.
- Trailing garbage detection 7 recovery.
- Registration of file-type associations with default icons.
- Update to latest versions of BCMenu and FishLib.

19.8 Changes for AWS Browse V1.4.0 (Build 1483)

Release date: August 16, 2006

19.8.1 Fixes

- Fix block data display to show 32-bit block displacements instead of 16.
- Fix divide-by-zero bug in Progress Dialog handling.

19.8.2 Enhancements

- Report FishLib DLL version information too.
- Support for searching within block, file or entire tape.

19.9 Changes for AWS Browse V1.3.0 (Build 1445)

Release date: unknown

19.9.1 Enhancements

- Support for block sizes grater than 64 kB (controllable via registry entry settings).
- Support for extremely large files (greater than 4 GB).
- Support for Bus-Tech ZLIB compressed AWS files.
- Support for toggling display of hex view on or off.
- Support for specifiying text area width when in text-only viewing mode.

19.10 Changes for AWS Browse V1.2.0 (Build 1278)

Release date: unknown

19.10.1 Enhancements

- Much more functional und friendly user interface, much faster than old AWSBROWS.EXE..
- Supports both AWS and HET files (including spanned block files).
- Supports both ZLIB and BZIP2 compressed file formats.
- User configurable (font and layout) hex display.
- Hex / text search capability.
- Displays both EBCDIC or ASCII.
- “Ditto”-like toolbar buttons for quick positioning to next / previous block or file.
- Clipboard support for both hex and text as well as binary data.
- Printing and print preview support.

20. The MVS Turnkey System

Figure 35: MVS Turnkey System Logo

20.1 What is the MVS Turnkey System

The MVS Tur(n)key system is a package of freely available parts of the IBM MVS 3.8J Operating System. The Turnkey System does not (to the best knowledge) contain any software copyrighted or licenced by IBM or any other company. The system was created by Volker Bandke.

The Turnkey System lets you have a MVS system up and running in less than 15 minutes. This distribution is a turnkey system. You insert the CDROM, run setup, answer a few questions, and you have a fully functional MVS 3.8J system on your hard disk. There is even an automatic mode, which only asks you one question altogether. Installation is usually a matter of a few minutes.

The Turnkey system runs equally well on Linux as on Windows platforms. It comes complete with Hercules and a text mode 3270 emulator. All the required Cygwin modules for the Windows environment are included on the CD as well.

The CD can be ordered at http://www.cbttape.org/cdrom.htm. It can also be downloaded as an .iso file from http://www.ibiblio.org/jmaynard/turnkey-mvs-3.zip and can be burned to a CD. A comprehensive cookbook can be found at http://www.bsp-gmbh.com/turnkey/cookbook/index.html.

The MVS Turnkey system is the perfect starting point for the Hercules Emulator.

20.2 Installed FMIDs

The following table shows all the functions (FMIDs) that are installed (ACCEPTED) in the MVS 3.8J Turnkey system:

FMID	Description
EAS1102	System Assembler
EBB1102	Base Control Program
EBT1102	BTAM
EDE1102	Display Exception Monitor Facility
EDM1102	Data Management
EDS1102	DM Support
EER1400	EREP
EGA1102	GAM
EGS1102	GAM Subroutines
EIP1102	IPCS
EJE1103	JES2 + 3800 enh.
EMF1102	MF/1
EMI1102	MICR/OCR
EML1102	Multi-Leaving WS
EMS1102	MSS
EPM1102	Program Management
EST1102	System Support
ESU1102	SU Bit String
ESY1400	SMP4
ETC0108	TCAM
ETI1106	TIOC
ETV0108	TSO / VTAM
EUT1102	Utilities
EVT0108	VTAM
EXW1102	External Writer
FBB1221	MVS Processor support

FMID	Description
FDM1133	3800 Enhancements - Data Management Utilities
FDS1122	MVS Processor Support
FDS1133	3800 enhancements - Data Management Utilities
FDZ1610	DSF
FUT1133	3800 enhancements - Data Management Utilities

Table 14: MVS 3.8J FMIDs

20.3 PTFs for MVS 3.8J

Some people have spent a lot of time locating, collecting and combining old PTF data to make them available for the Turnkey system. So far there are 1443 PTFs that have been received into the system.

A few PTFs have not been installed, as they introduce possible problems. PTF UY49086 is known to break TGET and TPUT services and no fix is available. Others refer to a SYSGEN symbol of &SGDTCAM,which is not provided by any of the PTFs available.

20.4 Installed USERMODs for MVS 3.8J

The following table shows the user modifications (USERMODS) that have been installed (APPLIED) into the MVS Turnkey system.

FMID	Description
M026200	Support for 3375/3380/3390 DASD
M026302	Support for 3375/3380/3390 DASD
M026305	Support for 3375/3380/3390 DASD
M026404	Support for 3375/3380/3390 DASD
M026405	Support for 3375/3380/3390 DASD
M036408	Support for 3375/3380/3390 DASD
ZP60003	Allow blank lines in IFOX00 assembler source
ZP60004	Add Highlighting to MVS Console
ZP60006	Print SIO counts on dataset disposition messages
ZUM0001	TSO Authorization Table
ZUM0002	SMF Exit IEFACTRT

FMID	Description
ZUM0003	WTO Exit IEECVXIT for Autopilot
ZUM0004	Add CMD1 to Subsystem name table for # command subsystem
ZUM0005	Remove JES2 from MSTRJCL
ZUM0006	Add BSP1 to Subsystem name table for autopilot

Table 15: List of installed USERMODS for MVS 3.8J

21. Technical Support

21.1 Paid Support

Hercules does not have any official (formal) paid support in place…

…HOWEVER…

…if you ask the group whether or not there is anyone out there willing to provide it, you will probably get more than one positive response.

It is known that at least one person already provides Hercules Technical Support on a paid basis and as indicated previously, it is likely that there are others who are already doing such or willing to. Please send email to any of the people involved with Hercules development should you feel you require paid technical support.

The best approach for obtaining fee-paid technical support for Hercules would be to post an inquiry to the main Hercules-390 forum (see sections below) and simply ask whether anyone out there would be willing to provide it.

21.2 Free Support

Since Hercules is an Open Source product owned by no one in particular and copyrighted by many (lots of *very sharp* people have contributed over the years to Hercules's success), there is no official "Technical Support Department"

As with most Open Source products support os available via a dedicated group of individuals and enthusiasts who are willing to put in their spare time helping others with whatever problems and/or questions others may have regarding Hercules. This dedicated group of Hercules enthusiasts is known as "the user community" and in this authors opinion, vie the Hercules-390 forum this group provides an excellent first point for all Hercules technical support.

If your question or concern regarding Hercules is not already addressed in the FAQ on the website or within other areas of Hercules source-code and documentation,,consider posting your question to either the main **Hercules-390** forum or one of the more "focused" forums discussed in the next section:

21.3 Forums

There are several discussion groups for users of the Hercules ESA/390 mainframe emulator. The Hercules-390 forum is in fact your primary source for Hercules support and you are strongly encouraged to subscribe. There is a vibrant, active community of over 5000 members, many of which are very knowledgeable in many different areas of mainframe technology, both hardware and software/OS points of view.

In addition to the main Hercules-390 forum, other more specialized Hercules forums exist to provide more focused support for a variety of popular IBM mainframe operating systems, such as DOS, VM, and MVS (see sections below).

Note: *the use of Yahoo! as host for the Hercules-390 and related forums should in no way be interpreted as an endorsement of the Yahoo! service.*

21.4 Hercules-390

This is the primary discussion group for the Hercules Emulator.

Community email addresses:

Post message: hercules-390@yahoogroups.com
Subscribe: hercules-390-subscribe@yahoogroups.com
Unsubscribe: hercules-390-unsubscribe@yahoogroups.com
List owner: hercules-390-owner@yahoogroups.com

Files and archives at:

http://groups.yahoo.com/group/hercules-390

21.5 H390-DOSVS

This forum discusses DOS/VS and DOS/VSE under Hercules.

Community email addresses:

Post message: h390-dosvs@yahoogroups.com
Subscribe: h390-dosvs-subscribe@yahoogroups.com
Unsubscribe: h390-dosvs-unsubscribe@yahoogroups.com
List owner: h390-dosvs-owner@yahoogroups.com

Files and archives at:

http://groups.yahoo.com/group/h390-dosvs

21.6 H390-MVS

This forum is for Hercules-390 users whom are trying to run MVS.

Community email addresses:

Post message: h390-mvs@yahoogroups.com
Subscribe: h390-mvs-subscribe@yahoogroups.com
Unsubscribe: h390-mvs-unsubscribe@yahoogroups.com
List owner: h390-mvs-owner@yahoogroups.com

Files and archives at:

http://groups.yahoo.com/group/h390-mvs

21.7 Turnkey-MVS

This forum provides support for Volker Bandke's "MVS Tur(n)key System.

Community email addresses:

Post message: turnkey-mvs@yahoogroups.com
Subscribe: turnkey-mvs-subscribe@yahoogroups.com
Unsubscribe: turnkey-mvs-unsubscribe@yahoogroups.com
List owner: turnkey-mvs-owner@yahoogroups.com

Files and archives at:

http://groups.yahoo.com/group/turnkey-mvs

21.8 H390-VM

This forum is for Hercules-390 users, that are trying to run VM/370, VM/SP, & VM/ESA.

Community email addresses:

Post message: h390-vm@yahoogroups.com
Subscribe: h390-vm-subscribe@yahoogroups.com
Unsubscribe: h390-vm-unsubscribe@yahoogroups.com
List owner: h390-vm-owner@yahoogroups.com

Files and archives at:

http://groups.yahoo.com/group/h390-vm

21.9 Herc-4Pack

Support for Andy Norrie's 4 pack Hercules VM system.

Community email addresses:

Post message: herc-4pack@yahoogroups.com
Subscribe: herc-4pack-subscribe@yahoogroups.com
Unsubscribe: herc-4pack-unsubscribe@yahoogroups.com
List owner: herc-4pack-owner@yahoogroups.com

Files and archives at:

http://groups.yahoo.com/group/herc-4pack

21.10 Hercules-390-Announce

General announcements regarding Hercules can be found in this forum.

Community email addresses:

Post message:	hercules-390-announce@yahoogroups.com
Subscribe:	hercules-390-announce-subscribe@yahoogroups.com
Unsubscribe:	hercules-390-announce-unsubscribe@yahoogroups.com
List owner:	hercules-390-announce-owner@yahoogroups.com

Files and archives at:

http://groups.yahoo.com/group/hercules-390-announce

21.11 Hercules-Advocacy

Advocacy of issues relating to the Hercules emulator for IBM mainframe systems.

Community email addresses:

Post message:	hercules-advocacy@yahoogroups.com
Subscribe:	hercules-advocacy-subscribe@yahoogroups.com
Unsubscribe:	hercules-advocacy-unsubscribe@yahoogroups.com
List owner:	hercules-advocacy-owner@yahoogroups.com

Files and archives at:

http://groups.yahoo.com/group/hercules-advocacy

21.12 Hercules-S370ASM

Forum discussing use of S/370 assembler with the Hercules emulator.

Community email addresses:

Post message:	hercules-s370asm@yahoogroups.com
Subscribe:	hercules-s370asm-subscribe@yahoogroups.com
Unsubscribe:	hercules-s370asm-unsubscribe@yahoogroups.com
List owner:	hercules-s370asm-owner@yahoogroups.com

Files and archives at:

http://groups.yahoo.com/group/hercules-s370asm

22. Hercules Developers

22.1 Who are the Herculeans?

Hercules is currently maintained by the following persons:

- Roger Bowler (original author)
- Jay Maynard (current maintainer)
- Volker Bandke
- Jan Jaeger
- Greg Smith
- David B. Trout (“Fish”)
- Bernard van der Helm
- Ivan Warren

The following picture shows the original author of Hercules – Roger Bowler.

Figure 36: Hercules Author Roger Bowler

The next photo shows the maintainance group on the occasion of a development meeting in Enschede (Netherlands), December 2001. Additional photos of some of the developers can be found in the photo gallery at the end of this manual.

Figure 37: Hercules Developers, December 2001

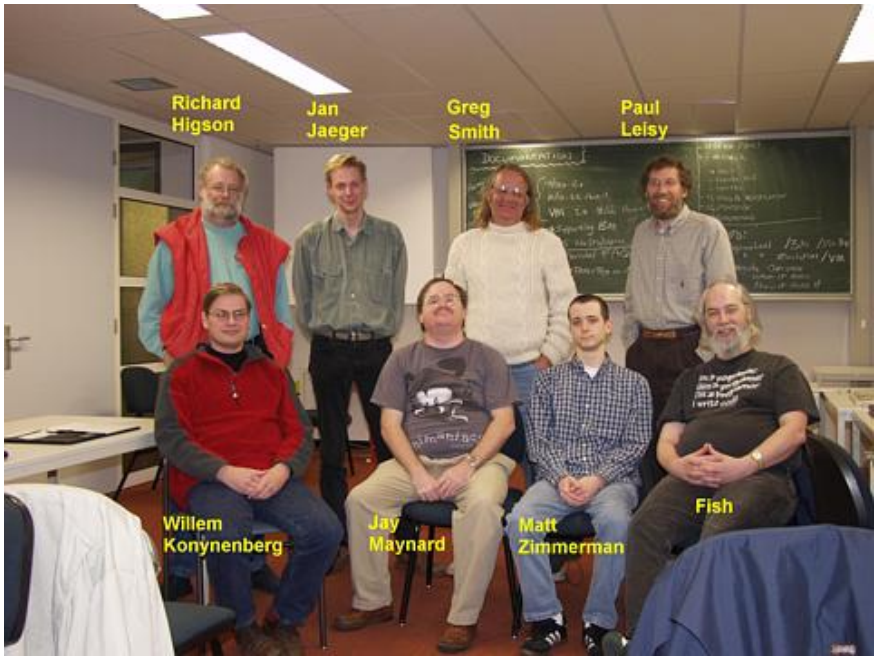


Figure 38: Hercules Developers (Legend)

The following people are developers who have made substantial contributions in the past:

- Richard Higson
- Willem Konynenberg
- Paul Leisy
- Matt Zimmerman

The following people are amongst those who have contributed to the project, either as coder, tester, documentor or combination:

- David Barth
- Malcolm Beattie
- Mario Bezzi
- Mike Cairns
- Marcin Cieslak
- Clem Clarke
- Vic Cross
- Jacob Dekel
- Guy Desbiens
- Juergen Dobrinski
- Fritz Elfert
- Tomas Fott
- Martin Gasparovic
- Mark Gaubatz
- Steve Gay
- Peter Glanzmann
- Roland Goetschi
- Graham Goodwin
- Paul Gorlinsky
- Harold Grovesteen
- Glen Herrmannsfeldt
- Brandon Hill
- Gabor Hoffer
- Dan Horak
- Soren Jorvang
- John Kozak
- Nobumichi Kozawa
- Peter Kuschnerus

- Kevin Leonard
- Albert Louw
- Peter Macdonald
- Thomas Masek
- Rick McKelvy
- John Mckown
- Dave Morton
- Christophe Nillon
- Mike Noel
- Andie Norrie
- Dutch Owen
- Max H. Parke
- Reed H. Petty
- Jim Pierson
- Valery Pogonchenko
- Pasi Pirhonen
- Wolfhard Reimer
- Emerson Santos
- Jeff Savit
- Axel Schwarzer
- Paul Scott
- Daniel Seagraves
- Victor Shkamerda
- Enrico Sorichetti
- John Summerfield
- Mark Szlaga
- Adam Thornton
- Adrian Trenkwalder
- Ronen Tzur
- Ard van der Leeuw
- Kris van Hees
- Kees Verruijt
- Rod Zazubek
- Bjoern A. Zeeb

The following people also supported the project in one or another way:

- Tim Alpaerts
- Bertus Bekker
- Giorgion de Nunzio
- Rick Fochtman
- Alex Friis
- Sam Golob
- Achim Haag
- Cory Hamasaki
- Tony Harminc
- Richard Higson
- Jim Keohane
- Sam Knutson
- Tim Pinkawa
- Mike Ross
- Daniel Rudin
- Rich Smrcina
- Henk Stegeman
- Mark S. Waterbury

If anyone feels they have been unfairly omitted from either of these lists, please inform us as described in 1.7 (“Readers Comments”).

23. Hercules Users

23.1 What people are saying about Hercules

"Never in my wildest dreams did I expect to see MVS running on a machine that I personally own. Hercules is a marvelous tool. My thanks to you all for a job very well done."

Reed H. Petty

"I do miss my mainframe a lot, and playing with Herc sure brings back memories. Just seeing the IBM message prefixes, and responding to console messages again was a wonderful nit of nostalgia."

Bob Brown

"I do have installed your absolutely fantastic /390 emulator. You won't believe what I felt I saw the prompt. Congratulations, this is a terrific software. I really have not had such a fascinating and interesting time on my PC lately."

IBM Large Systems Specialist

"Such simulators have been available for a long time. One of the most complete (up to modern 64-bit z/Architecture) is Hercules."

Michael Hack, IBM Thomas J. Watson Research Center

"An apparently excellent emulator, that allows those open source developers with an 'itch to scratch' to come to the S/390 table and contribute."

Mike MacIsaac, IBM

"BTW grab a copy of Hercules and you can test it at home. It's a very good S/390 and zSeries (S/390 64bit) emulator."

Alan Cox

"It works even better, than I imagined. Hercules is a fine piece of software."

Dave Sienkiewicz

"Hercules is a systems programmers dream come true."

René Vincent Jansen

"Aside from the electric trains my parents got me in 1953, this is the best toy I've ever been given, bar none."

Jeffrey Broido

“Congratulations to you and your team on a fine piece of work!”

Rich Smrcina

“Congratulations on a magnificent achievement!”

Mike Ross

“For anyone thinking running Hercules is toomuch trouble or too hard or whatever, I came home from work one day and my 13 year old 8th grade son had MVS running under VM under Hercules on Linux. He had gotten all the information about how to do this from the internet. When he complained about MVS console configuration and figuring out how to get it to work with VM, I knew he had felt all the pain he ever needed to feel about mainframes.”

Scott Ledbetter, StorageTek

“I am running a fully graphical Centos z/Linux environment on my desktop. The Hercules emulator is an amazing feat of engineering. I just wanted to send my compliments to the team for an excellent job! Thanks much for making this product part of the open-source community!”

Roby Gamboa

“I have DOS and DOS/VS running on Hercules with some demo applications, both batch and on-line. It does bring back some good memories. My compliments go to the Hercules team. Thank you ”

Bill Carlborg

“ This is stunning piece of work. To say that I am blown away is an understatement. I have a mainframe on my notebook!!!!!! P.S. Now if I can just remember my JCL”

Roger Tunnicliffe

23.2 IBM Redbooks

For about eighteen months the IBM Redbook SG24-4987 “Linux for S/390”, which is available at http://www.redbooks.ibm.com/abstracts/sg244987.html, contained a chapter written by Richard Higson, describing how to run Linux /390 under Hercules. Then suddenly, all mention of Hercules was mysteriously removed from the online edition of the book!

24. OSI (Open Source Initiative)

24.1 Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

Rationale: By constraining the license to require free redistribution, we eliminate the temptation to throw away many long-term gains in order to make a few short-term sales dollars. If we didn't do this, there would be lots of pressure for cooperators to defect.

24.2 Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost–preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

Rationale: We require access to un-obfuscated source code because you can't evolve programs without modifying them. Since our purpose is to make evolution easy, we require that modification be made easy.

24.3 Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

Rationale: The mere ability to read source isn't enough to support independent peer review and rapid evolutionary selection. For rapid evolution to happen, people need to be able to experiment with and redistribute modifications.

24.4 Integrity of the Author’s Source Code

The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

Rationale: Encouraging lots of improvement is a good thing, but users have a right to know who is responsible for the software they are using. Authors and maintainers have reciprocal right to know what they're being asked to support and protect their reputations.

Accordingly, an open-source license **must** guarantee that source be readily available, but **may** require that it be distributed as pristine base sources plus patches. In this way, "unofficial" changes can be made available but readily distinguished from the base source.

24.5 No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

Rationale: In order to get the maximum benefit from the process, the maximum diversity of persons and groups should be equally eligible to contribute to open sources. Therefore we forbid any open-source license from locking anybody out of the process.

Some countries, including the United States, have export restrictions for certain types of software. An OSD-conformant license may warn licensees of applicable restrictions and remind them that they are obliged to obey the law; however, it may not incorporate such restrictions itself.

24.6 No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

Rationale: The major intention of this clause is to prohibit license traps that prevent open source from being used commercially. We want commercial users to join our community, not feel excluded from it.

24.7 Distribution of License

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

Rationale: This clause is intended to forbid closing up software by indirect means such as requiring a non-disclosure agreement.

24.8 License Must Not Be Specific to a Product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

Rationale: This clause forecloses yet another class of license traps.

24.9 License Must Not Restrict Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

Rationale: Distributors of open-source software have the right to make their own choices about their own software.

Yes, the GPL is conformant with this requirement. Software linked with GPLed libraries only inherits the GPL if it forms a single work, not any software with which they are merely distributed.

24.10 License Must Be Technology-Neutral

No provision of the license may be predicated on any individual technology or style of interface.

Rationale: This provision is aimed specifically at licenses which require an explicit gesture of assent in order to establish a contract between licensor and licensee. Provisions mandating so-called "click-wrap" may conflict with important methods of software distribution such as FTP download, CD-ROM anthologies, and web mirroring; such provisions may also hinder code re-use. Conformant licenses must allow for the possibility that (a) redistribution of the software will take place over non-Web channels that do not support click-wrapping of the download, and that (b) the covered code (or re-used portions of covered code) may run in a non-GUI environment that cannot support popup dialogues.

25. Q Public License, Version 1.0

25.1 Copyright Information

THE Q PUBLIC LICENSE version 1.0, Copyright (C) 1999 Trolltech AS, Norway. Everyone is permitted to copy and distribute this license document.

25.2 Introduction

The intent of this license is to establish freedom to share and change the software regulated by this license under the open source model.

This license applies to any software containing a notice placed by the copyright holder saying that it may be distributed under the terms of the Q Public License version 1.0. Such software is herein referred to as the Software. This license covers modification and distribution of the Software, use of third-party application programs based on the Software, and development of free software which uses the Software.

25.3 Granted Rights

1. You are granted the non-exclusive rights set forth in this license provided you agree to and comply with any and all conditions in this license. Whole or partial distribution of the Software, or software items that link with the Software, in any form signifies acceptance of this license.

2. You may copy and distribute the Software in unmodified form provided that the entire package, including - but not restricted to - copyright, trademark notices and disclaimers, as released by the initial developer of the Software, is distributed.

3. You may make modifications to the Software and distribute your modifications, in a form that is separate from the Software, such as patches. The following restrictions apply to modifications:

   a. Modifications must not alter or remove any copyright notices in the Software.

   b. When modifications to the Software are released under this license, a non-exclusive royalty-free right is granted to the initial developer of the Software to distribute your modification in future versions of the Software provided such versions remain available under these terms in addition to any other license(s) of the initial developer.

4. You may distribute machine-executable forms of the Software or machine-executable forms of modified versions of the Software, provided that you meet these restrictions:

   a. You must include this license document in the distribution.

   b. You must ensure that all recipients of the machine-executable forms are also able to receive the complete machine-readable source code to the distributed Software, including all modify cations, without any charge beyond the costs of data transfer, and place prominent notices in the distribution explaining this.

   c. You must ensure that all modifications included in the machine-executable forms are available under the terms of this license.

5. You may use the original or modified versions of the Software to compile, link and run application programs legally developed by you or by others.

6. You may develop application programs, reusable components and other software items that link with the original or modified versions of the Software. These items, when distributed, are subject to the following requirements:

a. You must ensure that all recipients of machine-executable forms of these items are also able to receive and use the complete machine-readable source code to the items without any charge beyond the costs of data transfer.

b. You must explicitly license all recipients of your items to use and re-distribute original and modified versions of the items in both machine-executable and source code forms. The recipients must be able to do so without any charges whatsoever, and they must be able to re-distribute to anyone they choose.

c. If the items are not available to the general public, and the initial developer of the Software requests a copy of the items, then you must supply one.

25.4 Limitations of Liability

In no event shall the initial developers or copyright holders be liable for any damages whatsoever, including - but not restricted to - lost revenue or profits or other direct, indirect, special, incidental or consequential damages, even if they have been advised of the possibility of such damages, except to the extent invariable law, if any, provides otherwise.

25.5 No Warranty

The Software and this license document are provided AS IS with no warranty of any kind, including the warranty of design, merchantability and fitness for a particular purpose.

25.6 Choice of Law

This license is governed by the Laws of England.

Appendix A. Links

- **The Hercules System/370, ESA/390, and z/Architecture Emulator**

  http://www.hercules-390.org

- **Hercules Developer Snapshots (Ivan Warren)**

  http://www.ivansoftware.com/snapshots/snapshots

- **Hercules PDF Documentation (Peter Glanzmann)**

  http://hercdoc.glanzmann.org

- **The MVS Tur(n)key System, Version 3 (Volker Bandke)**

  http://www.bsp-gmbh.com/turnkey/index.html

  http://www.mvs-turnkey.de

- **Hercules WinGUI ("Fish", David B. Trout)**

  http://www.softdevlabs.com/Hercules/hercgui-index.html

- **CTCI-W32 ("Fish", David B. Trout)**

  http://www.softdevlabs.com/Hercules/ctci-w32-index.html

- **Hercules Studio (Jacob Dekel)**

  http://www.mvsdasd.org/hercstudio

- **WinPcap, Politecnico di Torino**

  http://www.winpcap.org

- **Vista tn3270, Tom Brennan Software**

  http://www.tombrennansoftware.com

- **X3270, Paul Mattes**

  http://x3270.bgp.nu

- **AWSBROWSE (“Fish”, David B. Trout)**

  http://www.softdevlabs.com/Hercules/hercgui-index.html

- **XMIT Manager**

  www.cbttape.org

- **CBT MVS Utilities Tape (CBTTAPE)**

  www.cbttape.org

- **Microsoft Visual C++ 2008 Express**

  http://www.microsoft.com/express/download/

- **ZLIB**

  http://www.zlib.net

  http://www.softdevlabs.com/Hercules/ZLIB1-1.2.3-bin-lib-inc-vc2008-x86-x64.zip

- **BZIP2**

  http://www.bzip.org

  http://www.softdevlabs.com/Hercules/BZIP2-1.0.5-bin-lib-inc-vc2008-x86-x64.zip

- **PCRE**

  http://www.pcre.org

  http://www.softdevlabs.com/Hercules/PCRE-6.4.1-bin-lib-inc-vc2008-x86-x64.zip

Appendix B. Hercules Developer Photo Gallery

B1. Developer Workshop Enschede, December 2001

The following pictures show some of the Hercules developers during their workshop in Enschede (Netherlands) in December 2001.

Figure 39: Greg Smith, Jay Maynard, Jan Jaeger and Fish

Figure 40: Willem Konynenberg and Fish

Figure 41: Fish and Greg Smith

Figure 42: Fish

Developer Workshop Paris, September/October 2008

The following pictures show some of the Hercules developers during their workshop in Paris (France) in September/October 2008.

Figure 43: Dave Wade, Jay Maynard, Volker Bandke, Bernard van der Helm, Roger Bowler

Figure 44: Dave, Jay, Iwan, Roger, Bernard and Volker (from left)

Figure 45: Roger Bowler and Iwan Warren

Figure 46: Volker Bandke and Jay Maynard

Figure 47: Dave Wade and Ivan Warren

Hercules Emulator

Hercules System/370, ESA/390,
z/Architecture Emulator

General Information

Version 3 Release 07

HEGI030700-01