

Linux on IBM Z and LinuxONE

Troubleshooting



Note

Before using this document, be sure to read the information in [“Notices” on page 37](#).

This edition applies to all Linux distributions that are supported on IBM Z[®] and IBM[®] LinuxONE until otherwise indicated in new editions.

© **Copyright International Business Machines Corporation 2012, 2023.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

- Summary of changes.....v**
 - What's new in edition SC34-2612-07..... v
 - What's new in edition SC34-2612-06..... v
 - SC34-2612-05.....vi

- About this document..... vii**

- Chapter 1. Troubleshooting for Linux on IBM Z.....1**
 - Techniques for troubleshooting Linux on IBM Z problems.....1
 - Troubleshooting checklist.....2
 - Collecting data for general Linux on IBM Z problems.....3
 - Collecting data for performance and resource utilization problems.....4
 - Collecting data for network problems.....4
 - Collecting data for hung system problems.....5
 - Collecting data for middleware problems.....5

- Chapter 2. Tools for troubleshooting.....7**
 - Assumptions.....7
 - General tools.....7
 - Tools for performance and resource utilization.....11
 - Special tools.....20
 - Dump tools.....26

- Chapter 3. Contacting IBM Support..... 29**

- Chapter 4. Exchanging information with IBM..... 31**
 - Sending information to IBM Technical Support.....31
 - Receiving information from IBM Support.....31

- Appendix A. How to detect guest relocation..... 33**

- Accessibility..... 35**

- Notices.....37**
 - Trademarks.....37

- Index..... 39**

Summary of changes

Changes to the troubleshooting information for the latest releases are listed.

What's new in edition SC34-2612-07

This revision reflects changes for SC34-2612-07.

New information

- A new option for **dbginfo** prints an overview of the current system environment, see [“dbginfo.sh - Collect information for debugging”](#) on page 7.

Changed information

- Novell Technical Support (nts) became SUSE Customer Center (scc). This changes the default location of the output of **supportconfig** for newer versions of the command, see [“supportconfig - SUSE Linux Enterprise Server troubleshooting”](#) on page 9.
- New IBM hardware and branding is incorporated.
- The IBM Knowledge Center is now IBM Documentation.

Deleted information

- The **netstat** command is deprecated and its description is removed. Use sockets statistics instead, see [“ss - Display sockets statistics”](#) on page 25.
- The Service Request Tool is no longer used, and the link to it removed.

What's new in edition SC34-2612-06

This revision reflects changes for SC34-2612-06.

New information

- Distribution information updated for Red Hat® Enterprise Linux® 8.0 and SUSE Linux Enterprise Server 15.
- New tools **ss -r** and **hyptop** are described, see [“hyptop - Display hypervisor performance data”](#) on page 24 and [“ss - Display sockets statistics”](#) on page 25.
- Examples have been updated to reflect new hardware and new tool versions.
- RETAIN is replaced by new IBM Support tools.

Changed information

- New IBM hardware and branding is incorporated.

Deleted information

- OProfile has been removed.

SC34-2612-05

This edition contains the following changes compared to SC34-2612-04.

New information

- Information on how to detect if a Linux instance has been moved to another guest virtual machine or LPAR is included, see [Appendix A, “How to detect guest relocation,” on page 33](#).

Changed information

- None

This revision also includes maintenance and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Deleted information

- None

About this document

This document describes troubleshooting of Linux instances running on IBM Z and IBM LinuxONE servers. It contains troubleshooting checklists, describes what tools to use for what problem, as well as how to contact IBM support and transfer log files.

For details about IBM tested Linux environments, see www.ibm.com/systems/z/os/linux/resources/testedplatforms.html.

Unless stated otherwise, all IBM z/VM® related information in this document assumes a current z/VM version, see www.vm.ibm.com/techinfo/lpmigr/vmleos.html.

You can find the latest version of this document on the IBM Documentation at

<https://www.ibm.com/docs/en/linux-on-systems?topic=troubleshooting-guide>

Your Linux distribution might provide utilities for working with mainframe devices that are not described in this publication. See the documentation that is provided with your distribution to find out what utilities you can use.

For Linux on IBM Z documents that are adapted to a particular distribution, see one of the following web pages:

- SUSE Linux Enterprise Server documents at

ibm.com/docs/en/linux-on-systems?topic=distributions-suse-linux-enterprise-server

- Red Hat Enterprise Linux documents at

ibm.com/docs/en/linux-on-systems?topic=distributions-red-hat-enterprise-linux

- Ubuntu documents at

ibm.com/docs/en/linux-on-systems?topic=distributions-ubuntu-server

These publications are available on IBM Documentation at ibm.com/docs/en/linux-on-systems?topic=linuxone-library-overview

- *Device Drivers, Features, and Commands*
- *Using the Dump Tools*
- *How to use FC-attached SCSI devices with Linux on z Systems®*, SC33-8413
- *Networking with RoCE Express*, SC34-7745
- *KVM Virtual Server Management*, SC34-2752
- *Configuring Crypto Express Adapters for KVM Guests*, SC34-7717
- *Introducing IBM Secure Execution for Linux*, SC34-7721
- *openCryptoki - An Open Source Implementation of PKCS #11*, SC34-7730
- *libica Programmer's Reference*, SC34-2602
- *libzpc - A Protected-Key Cryptographic Library*, SC34-7731
- *Exploiting Enterprise PKCS #11 using openCryptoki*, SC34-2713
- *Secure Key Solution with the Common Cryptographic Architecture Application Programmer's Guide*, SC33-8294
- *Pervasive Encryption for Data Volumes*, SC34-2782
- *Enterprise Key Management for Pervasive Encryption of Data Volumes*, SC34-7740
- *How to set an AES master key*, SC34-7712

- *Troubleshooting*, SC34-2612
- *Kernel Messages*, SC34-2599
- *How to Improve Performance with PAV*, SC33-8414
- *How to Set up a Terminal Server Environment on z/VM*, SC34-2596

Chapter 1. Troubleshooting for Linux on IBM Z

To isolate and resolve problems with Linux on IBM Z, you can use the troubleshooting information. This information contains instructions for using the problem-determination resources that are provided with Linux on IBM Z.

Techniques for troubleshooting Linux on IBM Z problems

Troubleshooting is a systematic approach to solving a problem. The goal of troubleshooting is to determine why something does not work as expected and how to resolve the problem. Certain common techniques can help with the task of troubleshooting.

The first step in the troubleshooting process is to describe the problem completely. Problem descriptions help you and the IBM technical-support representative know where to start to find the cause of the problem. This step includes asking yourself basic questions:

- What are the symptoms of the problem?
- Where does the problem occur?
- When does the problem occur?
- Under which conditions does the problem occur?
- Can the problem be reproduced?

The answers to these questions typically lead to a good description of the problem, which can then lead you to a problem resolution.

What are the symptoms of the problem?

When starting to describe a problem, the most obvious question is "What is the problem?" This question might seem straightforward; however, you can break it down into several more-focused questions that create a more descriptive picture of the problem. These questions can include:

- Who, or what, is reporting the problem?
- What are the error codes and messages?
- How does the system fail? For example, is it a loop, hang, crash, performance degradation, or incorrect result?

Where does the problem occur?

Determining where the problem originates is not always easy, but it is one of the most important steps in resolving a problem. Many layers of technology can exist between the reporting and failing components. Networks, disks, and drivers are only a few of the components to consider when you are investigating problems.

The following questions help you to focus on where the problem occurs to isolate the problem layer:

- Is the problem specific to one platform or operating system, or is it common across multiple platforms or operating systems?
- Is the current environment and configuration supported?
- Do all users have the problem?
- (For multi-site installations.) Do all sites have the problem?

If one layer reports the problem, the problem does not necessarily originate in that layer. Part of identifying where a problem originates is understanding the environment in which it exists. Take some time to completely describe the problem environment, including the operating system and version, all corresponding software and versions, and hardware information. Confirm that you are running within an

environment that is a supported configuration; many problems can be traced back to incompatible levels of software that are not intended to run together or have not been fully tested together.

When does the problem occur?

Develop a detailed timeline of events leading up to a failure, especially for those cases that are one-time occurrences. You can most easily develop a timeline by working backward: Start at the time an error was reported (as precisely as possible, even down to the millisecond), and work backward through the available logs and information. Typically, you need to look only as far as the first suspicious event that you find in a diagnostic log.

To develop a detailed timeline of events, answer these questions:

- Does the problem happen only at a certain time of day or night?
- How often does the problem happen?
- What sequence of events leads up to the time that the problem is reported?
- Does the problem happen after an environment change, such as upgrading or installing software or hardware?

Responding to these types of questions can give you a frame of reference in which to investigate the problem.

Under which conditions does the problem occur?

Knowing which systems and applications are running at the time that a problem occurs is an important part of troubleshooting. These questions about your environment can help you to identify the root cause of the problem:

- Does the problem always occur when the same task is being performed?
- Does a certain sequence of events need to happen for the problem to occur?
- Do any other applications fail at the same time?

Answering these types of questions can help you explain the environment in which the problem occurs and correlate any dependencies. Remember that just because multiple problems might have occurred around the same time, the problems are not necessarily related.

Can the problem be reproduced?

From a troubleshooting standpoint, the ideal problem is one that can be reproduced. Typically, when a problem can be reproduced you have a larger set of tools or procedures at your disposal to help you investigate. Consequently, problems that you can reproduce are often easier to debug and solve.

However, problems that you can reproduce can have a disadvantage: If the problem is of significant business impact, you do not want it to recur. If possible, re-create the problem in a test or development environment, which typically offers you more flexibility and control during your investigation.

- Can the problem be re-created on a test system?
- Are multiple users or applications encountering the same type of problem?
- Can the problem be re-created by running a single command, a set of commands, or a particular application?

Troubleshooting checklist

When you report a problem, provide as much information as possible about the circumstances.

Answering the following questions can help you or IBM support to determine the cause for problems that occur with Linux on IBM Z:

1. How does the problem manifest itself? What are the symptoms?

- When this problem occurs, is a specific error message or error code issued?
 - Is trace output of the operation available?
2. How long has the problem been occurring?
 - Is it a first time occurrence? When did it happen? (Date and time help to analyze the logs.)
 - How frequently does it occur?
 - Is there any pattern?
 3. If the problem occurred subsequent to some period of normal operation, did anything change in the environment?
 - Was an operating system patch applied?
 - Did the network environment change? For example, was a server moved or a domain migrated?
 - Did the system recently fail or abnormally terminate?
 4. If you know (for example, based on message prefixes or error codes), where does the problem occur? On one or more systems, production or test environment?
 5. Can you reproduce the problem on a test system (so that you do not negatively affect the production environment)? What steps are required to reproduce the problem?
 6. How many users are impacted?
 - Does this problem affect one, some, or all users?
 - Does the problem occur only for a user who was recently added to the environment, such as a new employee?
 - Do differences exist between the users who are affected and the users who are not affected?
 7. How many applications or business processes are impacted?
 - Does this problem affect one, some, or all applications or business processes?
 - Does the problem occur only for a new application or business process?
 - Do differences exist between the applications or business processes that are affected and the applications or business processes that are not affected by the problem?

In your report, describe the server and storage infrastructure in as much detail as possible:

- Machine setup, for example, IBM z16 or IBM z14 ZR1.
- Which driver and bundle is installed.
- Storage server, for example, DS8000®.
- Storage attachment, for example FICON®, or FCP.
- Disk configuration.
- Network, for example OSA (type, mode), HiperSockets.
- Network topologies.
- Middleware setup (databases, web servers, SAP, or Tivoli Storage Manager. Include version information, if relevant).

You can now collect additional diagnostic data that is required for an IBM technical-support representative to effectively troubleshoot the problem.

Collecting data for general Linux on IBM Z problems

Collect diagnostic data when a problem occurs. Then submit the diagnostic data to IBM Support. Whatever the problem, start with this general collection of data.

About this task

Collecting data before opening a support case can help you to answer the following questions:

- Do the symptoms match any known problems? If so, has a fix or workaround been published?
- Can the problem be identified and resolved without a code fix?
- When does the problem occur?

The diagnostic data that you collect, and the sources from which you collect that data, are dependent on the type of problem that you are investigating. A base set of information is typically always required. For specific symptoms, you might need to collect additional problem-specific data.

When you submit a problem to IBM Support, you must provide a base set of information.

Procedure

To collect general diagnostic data:

1. Collect the base set of diagnostic information by using **dbginfo.sh**.
2. Depending on your distribution, also collect distribution-specific information:
 - On SUSE Linux Enterprise Server, run **supportconfig**.
 - On Red Hat Enterprise Linux, run **sosreport**.
 - On Ubuntu Server, run **sosreport**.

Collecting data for performance and resource utilization problems

If performance or resource utilization is a problem, collect diagnostic data that you can use to diagnose and resolve the problem.

Procedure

To collect diagnostic data for performance diagnostics:

1. Start **sadc** (System Activity Data Collection) and provide **sar** files.
2. If running as guest under z/VM, collect z/VM **MONWRITE** data.
3. Attach the data files to the opened problem report.

Collecting data for network problems

If the network has a problem, collect diagnostic data that you can use to diagnose and resolve the problem.

Procedure

To collect diagnostic data for network diagnostics:

1. Provide a diagram of your network setup.
2. Use **netstat** to collect diagnostic data.

Note: **netstat** is deprecated as of SUSE Linux Enterprise Server 15. Use the socket statistics command **ss -r** instead, see “[ss - Display sockets statistics](#)” on page 25 .

3. Collect IP network configuration data, for example, by issuing **ip a 1** for the IP network configuration and **ip r 1** for the routing tables:

```
# ip a 1
# ip r 1
```

4. Some environments might have special routing rules that must be considered:

```
ip rule list
```

5. A typical network connection might be restricted by firewall rules and, thus, existing rules should also be reported.

For example, list rules and the default policy by issuing:

```
# iptables -vnl
```

Additionally, there can be NAT-rules. To list them, issue:

```
# iptables -t nat -vnl
```

6. A slow network is sometimes due to an incorrect DNS configuration. Check if the DNS servers are configured and reachable. Issue a **host** or **dig** command. The **ping** command might not always work due to firewall rules.
7. Attach the data files to the opened support case.

Collecting data for hung system problems

If the system hangs, collect diagnostic data that you can use to diagnose and resolve the problem.

Procedure

To collect diagnostic data for hung system diagnostics:

1. Create a kernel dump.

For more information about dump tools, see *Using the Dump Tools* available at

ibm.com/docs/en/linux-on-systems?topic=overview-using-dump-tools

2. Include the following files for SUSE Linux Enterprise Server:

- system.map
- kerntypes
- vmlinux (text)
- vmlinux (debug)

Include vmlinux (full) for Red Hat Enterprise Linux and Ubuntu.

Always include DBGINFO, see [“dbginfo.sh - Collect information for debugging”](#) on page 7.

Collecting data for middleware problems

If middleware is the problem, collect diagnostic data that you can use to diagnose and resolve the problem.

Procedure

To collect data for problems with any middleware product (for example, databases):

1. Contact the product support organization.
2. Collect the appropriate debug data as instructed.
3. Attach the data files to the opened support case.

Chapter 2. Tools for troubleshooting

A variety of troubleshooting tools are available to help you diagnose and resolve problems for Linux on IBM Z. Assumptions that are used for all tools are summarized here.

Assumptions

You need the correct authorization to use the commands. The examples assume that the file system is set up in a certain way.

Authority

Most of the tasks described require a user with root authority.

In particular, writing to procfs, and writing to most of the described sysfs attributes requires root authority.

Throughout, it is assumed that you have root authority.

sysfs and procfs

Most of the tasks described assume certain mount points for the file systems.

The mount point for the virtual Linux file system sysfs is assumed to be `/sys`. Correspondingly, the mount point for procfs is assumed to be `/proc`.

debugfs

It is assumed that debugfs is mounted at `/sys/kernel/debug`.

To mount debugfs, you can use this command:

```
# mount none -t debugfs /sys/kernel/debug
```

To mount debugfs persistently, add the following to `/etc/fstab`:

```
debugfs /sys/kernel/debug debugfs auto 0 0
```

General tools

Tools that can be used in most cases when debugging Linux on IBM Z problems.

dbginfo.sh - Collect information for debugging

The `dbginfo.sh` script collects various system-related files for debugging purposes. It captures the current system environment and generates a `.tar` file.

Note: The `dbginfo.sh` script also collects current debugging and trace information, which is saved in the virtual Linux file system `sysfs`. This information is not persistent across reboots. You must run the `dbginfo.sh` script before a Linux system is rebooted to avoid losing important debugging and trace information.

If the Linux system runs as z/VM guest operating system, `dbginfo.sh` also collects information about the z/VM guest setup.

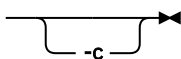
The `dbginfo.sh` script is part of the `s390-tools` package in SUSE Linux Enterprise Server and Ubuntu, and the `s390-utils` package in Red Hat Enterprise Linux.

The service and development team continuously improve `dbginfo.sh`. You can download the latest version from Github at github.com/ibm-s390-linux/s390-tools/releases. The `dbginfo.sh` script is included in the `s390-tools` .tar file.

Authorization

- Running the `dbginfo.sh` script requires root authority.
- For z/VM guest operating systems, any privilege class gives you relevant troubleshooting data. Privilege class B gives you the most data.

Syntax

► `dbginfo.sh`  `-c`

where

-c

prints an overview of the environment and checks that any dependent commands are available. No data is saved.

Examples

To generate a diagnostic report with `dbginfo.sh`, issue `dbginfo . sh`. The output is similar to the following:

```
[root@system]# dbginfo.sh
dbginfo.sh: Debug information script version <version>
Copyright IBM Corp. 2002, 2022

Hardware platform      = s390x
Runtime environment    = z/VM
Kernel version        = <version> (<version>)
OS version / distro   = <version>
Date and time of info = 2022-11-02-10-42-09

1 of 18: Collecting sysfs
2 of 18: Collecting procfs
3 of 18: Collecting config files
4 of 18: Collecting command output
5 of 18: Collecting htop for LPAR - 5s output
6 of 18: Skip z/VM: no z/VM environment
7 of 18: Collecting network output
8 of 18: Collecting osa oat output
9 of 18: Collecting ethtool output
10 of 18: Collecting Traffic Control output
11 of 18: Collecting bridge output
12 of 18: Skip OpenVSwitch: ovs-vsctl not available
13 of 18: Collecting KVM output
14 of 18: Collecting container host output
  Kubernetes ...
15 of 18: Skip nvme: not available
16 of 18: Collecting log files
  0 logfiles over 50 MB
17 of 18: Postprocessing
18 of 18: Finalizing: Creating archive with collected data

Collected data was saved to:
>> /tmp/DBGINFO-2022-11-02-10-42-09-<hostname>-<serial>.tgz <<

Please review all collected data before sending to your service organization.
Please review all collected data before sending to your service organization.
```

To see an environment overview, issue `dbginfo.sh -c`, for example:


```

# dbginfo.sh -c
dbginfo.sh: Debug information script version <version>
Copyright IBM Corp. 2002, 2022

Hardware platform      = s390x
Runtime environment   = LPAR
LPAR Name:            <name>
Kernel version        = <version>
OS version / distro   = <version>
KVM host              = YES
container host        = Kubernetes: YES - docker: NO

Current user          = root (must be root for data collection)
Date and time         = Wed Nov  2 10:44:28 UTC 2022
Uptime                = 10:44:28 up 6 days,  5:24,  1 user,  load average: 0.04, 0.05, 0.01
Number of core dumps  = 0
zdevice onl/conf/offl = 4 / 4 / 1741
Log file check        = 0 logfiles over 50 MB

Working directory     = /tmp
Filesystem            1K-blocks    Used Available Use% Mounted on
/dev/disk/by-path/ccw-0.0.7f76-part1 21118816 8842804 11177876 45% /
-rw----- 1 root root 12529573 Nov  2 10:42 /tmp/DBGINFO-2022-11-02-10-42-09-<hostname>-<serial>.tgz

Tool/command dependency check successful:
    timeout          = YES
    dump2tar         = YES

This is a console output only - no data was saved using option -c !

```

supportconfig - SUSE Linux Enterprise Server troubleshooting

The **supportconfig** script gathers system troubleshooting information on SUSE Linux Enterprise Server systems. It captures the current system environment and generates a tar-archive.

The script file collects complementary information to the `dbginfo.sh` script. The **supportconfig** script is part of the `supportutils` package.

Authorization

Running the script requires root authority.

Syntax

See the **supportconfig** man page for more details.

➤ supportconfig ➤

Example

To run **supportconfig**, issue:

```
root@system:~ # supportconfig
```

Output

The script produces a tar ball. The location of the tar ball is given in the script output:

```

=====
Support Utilities - Supportconfig
Script Version: <version>
Script Date: <date>

[...]
=====
Gathering system information
Data Directory: /var/log/scc_<linux_sys>_221208_1206

Basic Server Health Check... Done
[...]
System Modules... Done
Memory Details... Done
Disk I/O... Done
B-tree File System... Skipped
[...]
Creating Tar Ball

==[ DONE ]=====
Log file tar ball: /var/log/scc_<linux_sys>_221208_1206.tbz
Log file size: 2.6M
Log file md5sum: d57010927e94c54463084c3b94dfccb5-f
=====

```

sos report - Generate debugging information

The **sos report** script gathers system troubleshooting information. It captures the current system environment and generates a tar file. Use it for Red Hat Enterprise Linux or Ubuntu Server systems.

Note: The **sosreport** script is deprecated and replaced by the **sos report** script.

The script file collects complementary information to the `dbginfo.sh` script. The **sos report** script is part of the `sos` package.

Authorization

Running the script requires root authority.

Syntax

See the **sos report** man page for details.

➤ `sos report` ➤

Example

To run **sos report**, issue the command:

```
[root@system]# sos report
```

Output

The script produces a `.tar` file. The location of the `.tar` file is given in the script output:

```
[root@system]# sos report
This command will collect diagnostic and configuration information
[...]
Press ENTER to continue, or CTRL-C to quit.
[...]
Setting up archive ...
Setting up plugins ...
Running plugins. Please wait ...

Finishing plugins
Finished running plugins
Creating compressed archive...

Your sosreport has been generated and saved in:
/var/tmp/sosreport-<linux_sys>-test123-2022-12-09-tnvspio.tar.xz
[...]
```

Tools for performance and resource utilization

Use these tools when debugging performance problems or measuring resource utilization.

sadc - System activity data collector

The **sadc** command samples system data a specified number of times at a specified interval measured in seconds. It writes to the specified output file or the standard output in binary format. The **sadc** command is a backend to the **sar** command.

Data about, for example, the following areas is captured:

- CPU utilization
- Disk I/O overview and on device level
- Network I/O and errors on device level
- Memory usage and swapping

The tools report statistics data over time and create average values for each item.

Starting sadc/sar as a service

Start sadc/sar by using the sysstat service. When started as a service, the data files are written to the `/var/log/sa` directory. The files are named `sa<dd>` and `sar<dd>` respectively, where `<dd>` is the current day's two-digit date. Both files are constantly updated during the day.

Procedure

Start the sysstat service.

- On Ubuntu, SUSE Linux Enterprise Server 12 or later, or Red Hat Enterprise Linux 7 or later: To start the sysstat service as a permanent service that persists across reboots, issue:

```
systemctl enable sysstat.service
```

To check that the service is started at boot time and to check that the service is active, issue:

```
systemctl status sysstat.service
```

If the service is not active, issue the following command to start it:

```
systemctl start sysstat.service
```

- On Red Hat Enterprise Linux 5 or 6: To start the sysstat service as a permanent service that persists across reboots, issue:

```
chkconfig sysstat on
```

To check that the service is started at boot time, issue:

```
chkconfig --list |grep sysstat
```

To check that the service is up and running, use the following command:

```
service sysstat status
```

If the service is not running, issue the following command to start it:

```
service sysstat start
```

- On SUSE Linux Enterprise Server 11: Configure the service using YaST to make data collection persistent across reboots.

To start the sysstat service directly, issue:

```
/etc/init.d/boot.sysstat start
```

This is not persistent across reboots.

To check the status of the sysstat service, issue:

```
/etc/init.d/boot.sysstat status
```

Results

To report performance data, include both the `sadc` and the `sar` data files with the problem report.

What to do next

After you collect the appropriate diagnostic data, you can complete the following tasks, as appropriate:

- [Chapter 3, “Contacting IBM Support,” on page 29](#)
- [Chapter 4, “Exchanging information with IBM,” on page 31](#)

Starting `sadc/sar` directly

If your problem requires data collection that is not covered by the `sar/sadc` defaults, you can start the tools manually. Start the tools manually, for example, when you need a smaller sampling interval than the default.

About this task

The sampling interval depends on the time period during which performance problems are seen. You can use a default sampling interval of 10 minutes. If performance problems occur for a couple of minutes occasionally, shorten the sampling interval to less than a minute.

Procedure

1. To start the `sadc` command directly, issue a command of the following form:

```
/usr/lib64/sa/sadc [options] [interval [count] > <sadc_outfile>
```

See the `sadc` man page for details.

For example:

```
[root@system:~]# /usr/lib64/sa/sadc 1 5 > sadc_outfile
[root@system:~]# /usr/lib64/sa/sadc -S DISK 10 > sadc_outfile
```

Omit the *count* parameter to let *sadc* sample data until it is stopped.

Use the *-S DISK* option to collect disk statistics. By default *sadc* does not report disks activity to prevent data files from growing too large.

Note: In Ubuntu, the *sadc* tool is located in the */usr/lib/sysstat/sadc* directory. Hence, issue a command of the following form:

```
/usr/lib/sysstat/sadc [options] [interval [count] > <sadc_outfile>
```

2. Extract data and write records by using the **sar** command.

Use a command of the following form:

```
sar -A -f <sadc outfile> > <sar outfile>
```

For example:

```
[root@system:~]# sar -A -f sadc_outfile > sar_outfile
```

where:

- A**
reports all the collected statistics.
- f**
specifies the binary input file.

The **sar** command creates a collection of performance reports from the collected **sadc** data and writes these reports to an output file.

Results

To report performance data, include both the *sadc* and the *sar* data files with the problem report.

What to do next

After you collect the diagnostic data, you can complete the following tasks, as appropriate:

- [Chapter 3, “Contacting IBM Support,” on page 29](#)
- [Chapter 4, “Exchanging information with IBM,” on page 31](#)

iostat - Monitor input/output device load

The **iostat** command monitors system input/output device load by observing the time that the devices are active in relation to their average transfer rates.

The **iostat** report shows:

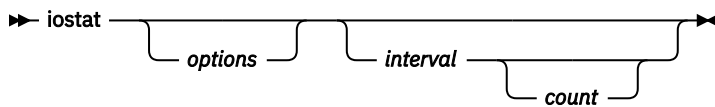
- CPU utilization
- Device queue information
- Service time

Authorization

Root access is required on Linux operating systems.

Syntax

See the **iostat** man page for the complete syntax and all options.



Parameters

- c**
Collects CPU statistics.
- d**
Collects disk statistics.
- t**
Prints a time stamp for each report.
- k**
Displays statistics in kilobytes per seconds instead of blocks per second.
- x**
Displays extended statistics, if available.

Examples

To generate a report with a sampling interval of 10 seconds, collecting disk statistics in KB per second, including a time stamp, and extended statistics, issue the command:

```
[root@system]# iostat -dtkx 10
```

z/VM MONWRITE - Collect CP *MONITOR data

If your Linux system runs as a guest operating system under z/VM and encounters performance problems, use the MONWRITE utility and include CP *MONITOR data in the problem report.

The z/VM monitor records are in binary format. Make sure that:

- The records are packed and tersed correctly.
- The record size settings are correct.
- The binary to ASCII conversion is made correctly.

For more information about how to collect and upload z/VM MONWRITE data, see www.ibm.com/vm/perf/tips/collect.html

Usage notes

- The `sadc` and `sar` files must cover the same time interval as the z/VM MONWRITE data.
- Use the default sampling time interval of 1 minute.

Collecting data using DASD statistics

The DASD statistics kernel function monitors the activities of the DASD device driver and the storage subsystem. It mainly records processing time of I/O operations within a given time interval.

Procedure

To collect diagnostic data by using DASD statistics:

1. Start DASD statistics with the following command:

```
# echo set on > /proc/dasd/statistics
```

2. Summarized histogram information is available in `/proc/dasd/statistics`, and can be extracted with the following command:

```
# cat /proc/dasd/statistics
```

3. Stop DASD statistics with the following command:

```
# echo set off > /proc/dasd/statistics
```

Results

DASD statistics creates a summary for all devices.

An IOCTL interface is available to collect the statistics for individual devices. To get DASD statistics for an individual DASD, first turn statistics collection on, then use the **tunedasd** command:

```
# echo set on > /proc/dasd/statistics  
# tunedasd -P /dev/dasd<xx>
```

What to do next

After you collect the appropriate diagnostic data, you can complete the following tasks:

- [Chapter 3, “Contacting IBM Support,” on page 29](#)
- [Chapter 4, “Exchanging information with IBM,” on page 31](#)

Displaying DASD performance data

The **dasdstat** command reports the statistics over time and gather data for individual devices or across all devices. The statistics include performance data about Parallel Access Volume (PAV) and High Performance FICON.

Before you begin

Before you can collect DASD performance statistics, the debug file system must be mounted, see [“debugfs” on page 7](#).

About this task

The **dasdstat** command is available as of Red Hat Enterprise Linux version 6.3, SUSE Linux Enterprise Server 11 SP3, and Ubuntu 16.04. Use **dasdstat** to gather DASD performance statistics and to display them.

Examples

The command can be used to get DASD performance statistics across all devices or for individual ones.

- To start gathering data for a summary of all available DASDs, issue:

```
# dasdstat -e global
```

- To start gathering data for a selected device `0.0.b223`, issue:

```
# dasdstat -e dasda 0.0.b223
```

- To stop gathering data for a single device, issue:

```
# dasdstat -d dasda
```

- To reset statistic counters for device `0.0.b223`, issue:

```
# dasdstat -r 0.0.b223
```

- To read data statistics for all devices and for a single device respectively, issue:

```
# dasdstat global  
# dasdstat dasda
```

Instead of using the **dasdstat** command, you can access the raw DASD performance data in debugfs. This data can be used, for example, to write scripts that evaluate dedicated DASD performance data.

For more information about accessing raw DASD performance data, see *Device Drivers, Features, and Commands*, SC33-8411 available at ibm.com/docs/en/linux-on-systems?topic=linuxone-distributions.

What to do next

After you collect the appropriate diagnostic data, you can complete the following tasks:

- Chapter 3, “Contacting IBM Support,” on page 29
- Chapter 4, “Exchanging information with IBM,” on page 31

ziomon - Collect FCP performance data

As of SUSE Linux Enterprise Server 11, Red Hat Enterprise Linux 6, Red Hat Enterprise Linux 5.8, and Ubuntu 16.04, use the **ziomon** tool to gather FCP performance data.

The monitor tool **ziomon** collects information and details about:

- The FCP configuration
- The system I/O traffic through FCP adapters
- The overall I/O latencies, adapter latencies, and fabric latencies
- The usage of the FCP resources

Use the **ziorep** tools to analyze the reports created by **ziomon**. The process is illustrated in [Figure 1 on page 16](#).

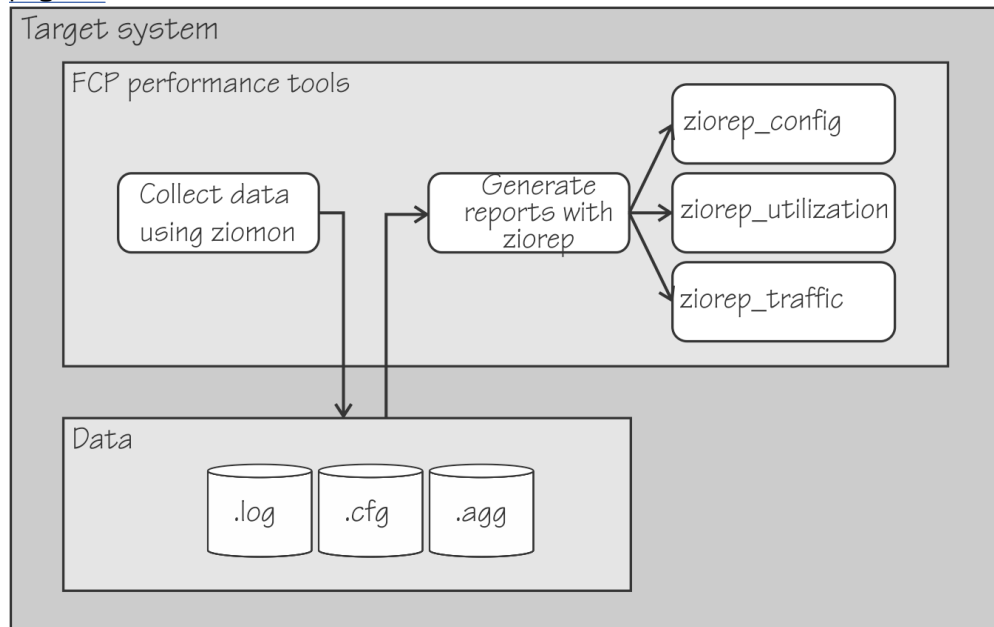


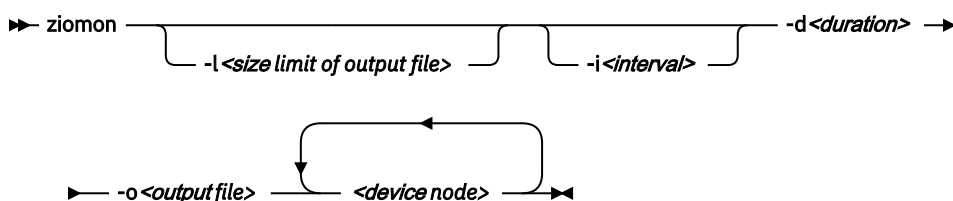
Figure 1. The FCP performance tools

Authorization

Root access is required on Linux operating systems.

ziomon syntax

See the **ziomon** man page for the complete syntax and all options.



Parameters

-i <interval>

Specifies the elapsed time between writing data to disk in seconds. Defaults to 60 seconds.

-d <duration>

Specifies the monitoring duration in minutes. Must be a multiple of the interval length.

-l <size limit of output file>

Defines the upper limit of the output files. Must include one of the suffixes M (megabytes), G (gigabytes), or T (terabytes). This limit is only a tentative value that can be slightly exceeded.

-o <output file>

Specifies the prefix for the log file, configuration file, and aggregation file.

<device>

Denotes one or more device names that are separated by blanks.

Examples

To generate a diagnostic report for devices /dev/sda and /dev/sdb, issue the command:

```
[root@system]# ziomon -i 20 -d 5 -l 50M -o trace_data /dev/sda /dev/sdb
```

Output

The **ziomon** tool creates two output files in the directory where it was started:

- *<output file>.cfg* holds various configuration data from the system
- *<output file>.log* holds the raw data samples that are taken during the data collection phase in a binary format
- *<output file>.agg* aggregates old sample data when the *.log* file grows larger than the allowed limit, thus freeing the log file for more recent data.

Usage notes

- Needs **vmalloc** space for each device node and CPU.
- The **ziomon** tool can be stopped with CTRL+C before the time period expires.

ziorep - Create FCP performance report

After you collect FCP performance data using the **ziomon** tool, use the **ziorep** tool to create appropriate FCP performance reports.

Three reporting tools are available as of SUSE Linux Enterprise Server 11, Red Hat Enterprise Linux 6, Red Hat Enterprise Linux 5.8, and Ubuntu 16.04.

- [“ziorep_config - Configuration report” on page 18](#)
- [“ziorep_utilization - Report on utilization details” on page 18](#)
- [“ziorep_traffic - Traffic report” on page 19](#)

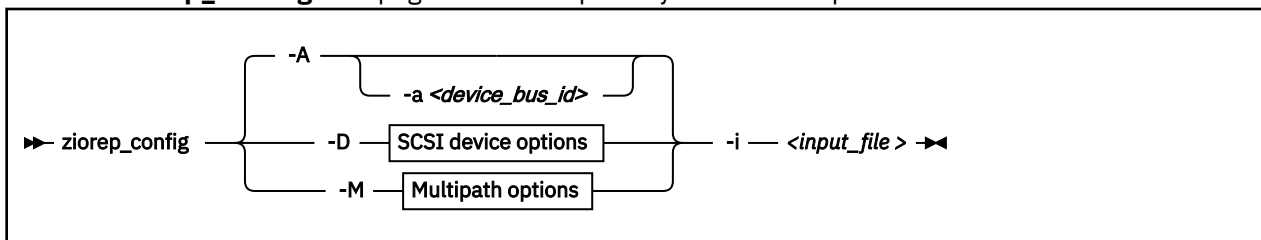
For more information about the FCP performance reports, see *How to use FC-attached SCSI devices with Linux on z Systems*, SC33-8413.

ziorep_config - Configuration report

Use the **ziorep_config** to report the configuration of the attached SCSI storage and to visualize the interconnection between the different layers that are involved in the SCSI attachment. Use the `.cfg` file from **ziomon** as input. The configuration data is reported according to configuration report type (adapter report, device report, or multipath report).

ziorep_config syntax

See the **ziorep_config** man page for the complete syntax and all options.



-i or --input <input_file>

Specifies the configuration file that is created by **ziomon** as source.

-a or --adapter <device_bus_id>

Limits the output to the list of FCP devices specified.

-A or --Adapter

Prints the adapter (FCP device) report, this report is the default.

-D or --Device

Prints the SCSI device report.

-M or --Map

Prints the multipath mapper report.

Example

To generate a device report for a specific SCSI device from a previous **ziomon** run that created the configuration file `trace_data.cfg`, and with an enabled first line (table header), issue the following command:

```
[root@system]# ziorep_config -D -t -l 0x4021400f00000000 -i trace_data.cfg
```

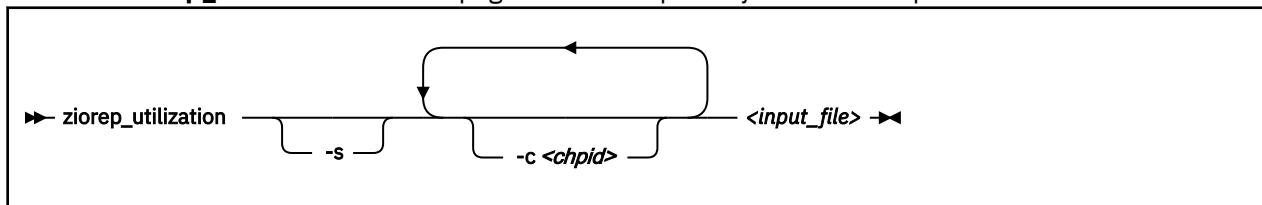
ziorep_utilization - Report on utilization details

Use the **ziorep_utilization** command to produce a report on the usage of FCP resources. Use the `.log` file from **ziomon** as input. There are two different reports available. The first report provides information on the physical adapters. This report includes usage statistics of the card's bus, CPU, and overall utilization.

The second report provides various metrics on the virtual adapter. This report includes statistics on the Queued Direct I/O (QDIO) queue that is used to transfer data between the Linux system and the FCP adapter. Statistics that are given are average and maximum utilization in each interval and the number of instances when the queue was full.

ziorep_utilization syntax

See the **ziorep_utilization** man page for the complete syntax and all options.



-s or --summary

shows a summary of the data.

-c or --chpid <chpid>

limits the data the specified FCP adapters. The format is a 2-byte hexadecimal number. You can specify multiple FCP channels by specifying `-c` multiple times.

Example

To generate a utilization report for the specific physical adapter with CHPID 50, issue the following command:

```
[root@system]# ziorep_utilization -c 50 trace_data.log
```

ziorep_traffic - Traffic report

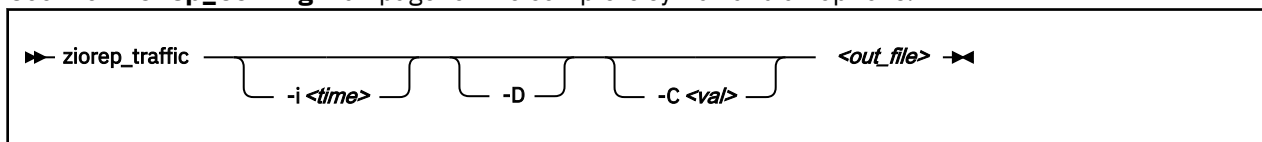
Use the **ziorep_traffic** command produces a report about the systems I/O traffic through FCP channels and traffic latency. Use the `.log` file from **ziomon** as input. There are two reports available, varying by the level of detail:

- The default report shows traffic information on a summary level.
- A report that is limited to certain devices that gives detailed traffic information.

Each device is identified by WWPN or LUN and gives information about I/O rate and throughput and latencies in the I/O subsystem, channel, and fabric. See the **ziorep_traffic** man page for the complete syntax and all options.

ziorep_traffic syntax

See the **ziorep_config** man page for the complete syntax and all options.



-i <time> or --interval <time>

Sets the aggregation interval to `<time>` in seconds. Must be a multiple of the interval size of the source data. Set to 0 to aggregate over all data.

-C or --collapse <val>

Specifies on what level you want to aggregate data. See the **ziorep_traffic** man page for more details about possible aggregation levels.

-D or --Device

Gives detailed information about the traffic.

For more information about **ziorep_traffic** and its options, see https://www.ibm.com/support/knowledgecenter/en/linuxonibm/com.ibm.linux.z.ldsg/ldsg_r_ziorep_traffic.html

Example

To generate a traffic report from the `trace_data.log` created by **ziomon**, for all devices with the data aggregated to a 60-second interval, issue the following command:

```
[root@system]# ziorep_traffic -i 60 trace_data.log
```

To generate the same traffic report and export it to a CSV file, issue the following command:

```
[root@system]# ziorep_traffic -i 60 trace_data.log -x
```

In this example, the data is exported to a file called `trace_data_traffic.csv`.

Obtaining QDIO performance statistics

As of SUSE Linux Enterprise Server 11 SP2, Red Hat Enterprise Linux 6.2, and Ubuntu 16.04, use the QDIO performance statistics to obtain information of QDIO devices. These statistics apply to FCP devices and to qeth devices.

About this task

To look at these debug logs use the Linux file system `debugfs`, which must be mounted. See [“debugfs” on page 7](#) for details.

These statistics are located in `<debugfs_mount>/qdio/<device_bus_id>/statistics` where `<debugfs_mount>` is the mount point for `debugfs` and `<device_bus_id>` is the bus ID of an FCP or qeth device.

Procedure

- To collect QDIO performance statistics for the device `fc00`, issue the command:

```
# echo 1 > /sys/kernel/debug/qdio/0.0.fc00/statistics
```

- To stop collecting QDIO performance statistics, issue:

```
# echo 0 > /sys/kernel/debug/qdio/0.0.fc00/statistics
```

- To read the collected data, issue:

```
# cat /sys/kernel/debug/qdio/0.0.fc00/statistics
```

What to do next

After you collect the appropriate diagnostic data, you can complete the following tasks:

- [Chapter 3, “Contacting IBM Support,” on page 29](#)
- [Chapter 4, “Exchanging information with IBM,” on page 31](#)

Special tools

Tools for special circumstances that can be used when debugging Linux on IBM Z problems.

s390dbf traces - Use the kernel debug feature

All device drivers and other kernel components write debug log records. These records are available after a system crash. You can also read and save them on a running system.

To look at these debug logs use the Linux file system `debugfs`, which must be mounted. See [“debugfs” on page 7](#) for details.

Below the `s390dbf` directory each registered component is represented by a subdirectory with the name of that component. The subdirectories contain files that represent different views of the debug log. Available views are: `hex_ascii`, `sprintf`, `flush`, `pages`, and `level`.

The debug information that is written to the logs depends on the debug level that is set for that log. The debug level ranges from 0 for the least detail to 6 for the most detail. The default level is 4. Only debug entries with a level that is lower or equal to the actual level are written to the log.

To set or change a debug level, from the s390dbf subdirectory for the component you want to work with, issue:

```
echo <value> > level
```

Examples

- To collect the maximum amount of debug information, issue:

```
echo 6 > level
```

- To flush the debug log buffer for the component, issue:

```
echo - > flush
```

- The kernel debug feature uses wraparound memory buffers. To increase the buffer size, read it first and then enter a higher value with the following command:

```
echo 10 > pages
```

scsi_logging_level - Use the SCSI logging feature

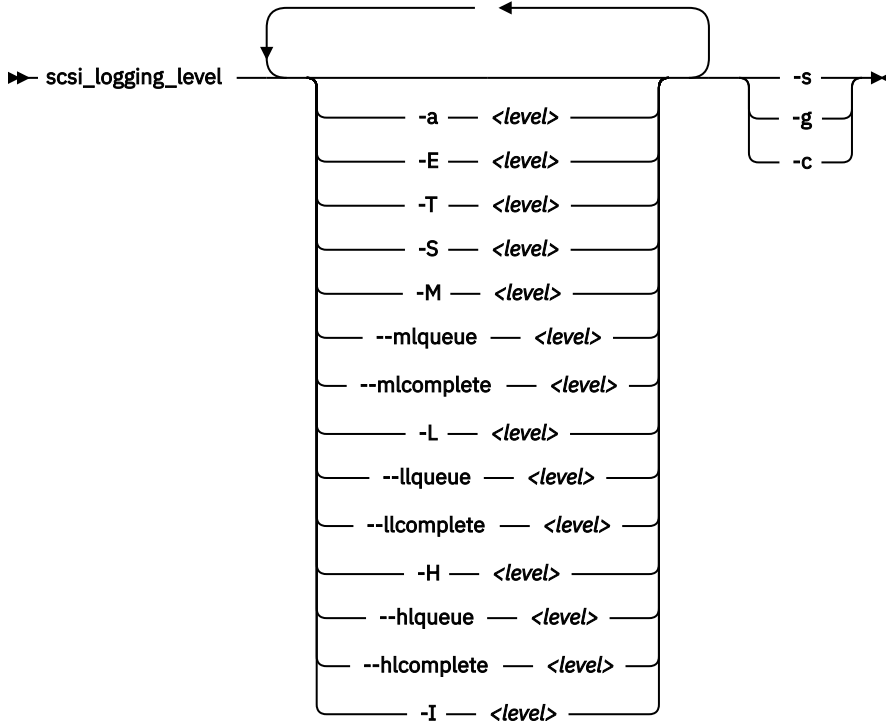
Use the **scsi_logging_level** command to create, set, or get the SCSI logging level.

The SCSI logging feature prints SCSI messages of the various log areas to the kernel message log file `/var/log/messages`. This feature is used to get information about:

- Issues with the LUN discovery.
- Issues with SCSI error handling and recovery, for example, problems with dirty fibers.

scsi_logging_level syntax

See the **scsi_logging_level** man page for the complete syntax and all options.



Parameters:

-a or --all <level>

specifies value for all SCSI_LOG fields.

-E or --error <level>

specifies SCSI_LOG_ERROR.

-T or --timeout <level>

specifies SCSI_LOG_TIMEOUT.

-S or --scan <level>

specifies SCSI_LOG_SCAN.

-M or --midlevel <level>

specifies SCSI_LOG_MLQUEUE and SCSI_LOG_MLCOMPLETE.

--mlqueue <level>

specifies SCSI_LOG_MLQUEUE.

--mlcomplete <level>

specifies SCSI_LOG_MLCOMPLETE.

-L or --lowlevel <level>

specifies SCSI_LOG_LLQUEUE and SCSI_LOG_LLCOMPLETE.

- llqueue <level>**
specifies SCSI_LOG_LLQUEUE.
- llcomplete <level>**
specifies SCSI_LOG_LLCOMPLETE.
- H or --highlevel <level>**
specifies SCSI_LOG_HLQUEUE and SCSI_LOG_HLCOMPLETE.
- hlqueue <level>**
specifies SCSI_LOG_HLQUEUE.
- hlcomplete <level>**
specifies SCSI_LOG_HLCOMPLETE.
- I or --ioctl <level>**
specifies SCSI_LOG_IOCTL.
- s or --set**
creates and sets the logging level as specified on the command line.
- g or --get**
gets the current logging level.
- c or --create**
creates the logging level as specified on the command line.
- v or --version**
displays version information.
- h or --help**
displays help text.

You can specify several SCSI_LOG fields by using several options. When multiple options specify the same SCSI_LOG field, the most specific option has precedence.

Examples

- To display the logging word of the SCSI logging feature and each logging level, use the following command:

```
# scsi_logging_level -g
```

- To set an appropriate log level for the most important log areas with only negligible impact on the performance, use the following command:

```
# scsi_logging_level --mlcomplete 1 -T 7 -E 5 -S 7 -I 0 -a 0 -s
```

- You can add the SCSI logging level to the kernel boot parameters. To add it, specify for example:

```
"scsi_mod.scsi_logging_level=4605"
```

- To switch off the logging feature, use the following command:

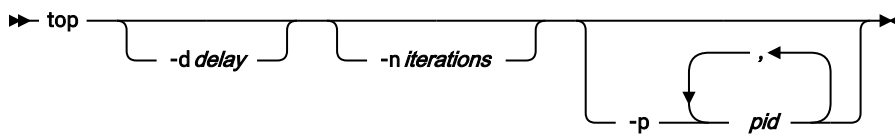
```
# scsi_logging_level -s -a 0
```

top - See resource usage

The **top** command provides a dynamic, real-time view of a running system and shows resource usage on a thread level. It can show, for example, CPU usage and detailed memory usage.

Syntax

See the **top** man page for the complete syntax and all options.



Parameters

- b**
Writes the output for each interval into a file.
- d**
Specifies the delay time interval in seconds.
- n**
Indicates that the maximum number of iterations **top** should produce before it ends.
- p**
Limits the output to the specified processes.

In the running **top** program, use the F key to configure the displayed columns. Use the W key to write the current configuration to `~/ .toprc`. This is the default directory.

Example

To write 180 iterations 1 second apart into a file, issue:

```
[root@system]# top -b -d 1 -n 180 >top.log 2>&1
```

hyptop - Display hypervisor performance data

Use the **hyptop** command to obtain a dynamic real-time view of a hypervisor environment on IBM Z.

Depending on the data available from the hypervisor, it shows, for example, CPU and memory information about LPARs or z/VM guest virtual machines.

System names provided by **hyptop** are either LPAR names as shown on the SE or HMC, or z/VM guest IDs that identify z/VM guest virtual machines.

The **hyptop** command provides two main windows:

- A list of systems that the hypervisor is currently running (`sys_list`). This is the default.
- One system in more detail (`sys`).

You can run **hyptop** in interactive mode (default) or in batch mode with the **-b** option.

With **hyptop** you can easily monitor z/VM guests, for example, to determine and understand high steal times.

For more information, see the man page, or the **hyptop** description in *Device Drivers, Features, and Commands*, available from: ibm.com/docs/en/linux-on-systems?topic=overview-device-drivers-features-commands

Example

To start **hyptop** with the `sys_list` window in interactive mode, enter:

```
[root@system]# hyptop
```

- If your Linux instance is running in an LPAR that has permission to see the other LPARs, the output looks like the following example:


```

12:30:48 | cpu-t: IFL(18) CP(3) UN(3)                                     ?=help
system #core core mgm Core+ Mgm+ online
(str) (#) (%) (%) (hm) (hm) (dhm)
S05LP30 10 461.14 10.18 1547:41 8:15 11:05:59
S05LP33 4 133.73 7.57 220:53 6:12 11:05:54
S05LP50 4 99.26 0.01 146:24 0:12 10:04:24
S05LP02 1 99.09 0.00 269:57 0:00 11:05:58
TRX2CFA 1 2.14 0.03 3:24 0:04 11:06:01
S05LP13 6 1.36 0.34 4:23 0:54 11:05:56
TRX1 19 1.22 0.14 13:57 0:22 11:06:01
TRX2 20 1.16 0.11 26:05 0:25 11:06:00
S05LP55 2 0.00 0.00 0:22 0:00 11:05:52
S05LP56 3 0.00 0.00 0:00 0:00 11:05:52
413 823.39 23.86 3159:57 38:08 11:06:01

```

- If your Linux instance runs in a z/VM guest virtual machine that has permission to see the other z/VM guest virtual machines, the output looks like the following example:

```

12:32:21 | CPU-T: UN(16)                                               ?=help
system #cpu cpu Cpu+ online memuse memmax wcu
(str) (#) (%) (%) (hm) (dhm) (GiB) (GiB) (#)
T6360004 6 100.31 959:47 53:05:20 1.56 2.00 100
DTCVSW1 1 0.00 0:00 53:16:42 0.01 0.03 100
T6360002 6 0.00 166:26 40:19:18 1.87 2.00 100
OPERATOR 1 0.00 0:00 53:16:42 0.00 0.03 100
T6360008 2 0.00 0:37 30:22:55 0.32 0.75 100
T6360003 6 0.00 3700:57 53:03:09 4.00 4.00 100
NSLCF1 1 0.00 0:02 53:16:41 0.03 0.25 500
PERFSVM 1 0.00 0:53 2:21:12 0.04 0.06 0
TCPIP 1 0.00 0:01 53:16:42 0.01 0.12 3000
DIRMAINT 1 0.00 0:04 53:16:42 0.01 0.03 100
DTCVSW2 1 0.00 0:00 53:16:42 0.01 0.03 100
RACFVM 1 0.00 0:00 53:16:42 0.01 0.02 100
75 101.57 5239:47 53:16:42 15.46 22.50 3000

```

ps - Report a snapshot of the current processes

The **ps** command gives comprehensive statistics data on process level and reports a snapshot of the current processes.

See the **ps** man page for the complete syntax and all options.

Example

The following sample command shows every process in an easily readable format:

```

[root@system]# ( DELAY=10; while [ true ]; do echo "*** "`date`;
ps -eLo pid,user,%cpu,
%mem,wchan:15,nwchan,stat,time,flags,etime,command:50;
sleep $DELAY; done ) | tee psinfo.out

```

ss - Display sockets statistics

Use the **ss** command to analyze the sockets that run on a Linux system.

The **ss** command replaces **netstat**, which is deprecated. The options for **ss** are similar to the **netstat** options.

Issue **ss -r** for a table of statistics with resolved addresses. For more information, see the man page.

Example

To start **ss -r**, issue:

```
[root@system]# ss -t
Netid State Recv-Q Send-Q Local Address:Port Peer Address:Port
u_str ESTAB 0 0 /run/dbus/system_bus_socket 19523 * 1382
u_str ESTAB 0 0 /run/systemd/journal/stdout 18592 * 16592
u_str ESTAB 0 0 * 18219 * 16595
u_str ESTAB 0 0 * 18086 * 636
u_str ESTAB 0 0 /run/systemd/journal/stdout 16595 * 18219
u_str ESTAB 0 0 * 16601 * 1375
u_str ESTAB 0 0 /run/systemd/journal/stdout 16593 * 18175
u_str ESTAB 0 0 /run/systemd/journal/stdout 16615 * 16614
u_str ESTAB 0 0 * 18221 * 16597
u_str ESTAB 0 0 /run/systemd/journal/stdout 1531 * 18389
u_str ESTAB 0 0 * 18175 * 16593
[...]
```

tcpdump - Collect traffic information for a network interface

The **tcpdump** network analysis tool dumps traffic collected for a given network interface.

tcpdump syntax

See the **tcpdump** man page for the complete syntax and all options.

```
➔ tcpdump -s <length> -X -i <interface>
```

Parameters

-s <length>

Writes <length> of data from each packet rather than the default 65535 bytes.

-X

Writes each packet in hexadecimal and in ASCII format.

-i <interface>

Identifies the network interface.

Example

To dump network traffic for interface eth0, issue the command:

```
[root@system]# tcpdump -s 65000 -X -i eth0
```

Dump tools

When the system hangs, create a memory dump.

The following dump tools are available:

- The DASD dump tool writes the memory dump directly to a DASD partition. It supports both ECKD and FBA DASDs.
- The tape dump tool writes the memory dump directly to an ESCON/FICON tape device.
- The SCSI dump tool writes the memory dump into file system. It is supported for LPAR and as of z/VM 5.4.
- The kdump feature is a two stage dump tool. This feature is made available through a Linux kernel and initial RAM disk that are preloaded in memory, along with a production system. It is available as of Red Hat Enterprise Linux 6.4, SUSE Linux Enterprise Server 11 SP3, and Ubuntu 16.04..
- VMDUMP (for z/VM guest operating systems) writes the memory dump to z/VM spool space (VM reader). VMDUMP uses a dump format specific to z/VM, the dump must be converted. Do not use VMDUMP to dump large VM guests; the dump process is very slow.

As of LinuxONE III, you can use NVMe devices as stand-alone dump devices for Linux in LPAR mode or in a DPM partition.

For more information about dump tools, see *Using the Dump Tools* available at

ibm.com/docs/en/linux-on-systems?topic=overview-using-dump-tools

In particular, the *Using the Dump Tools* book contains a chapter about handling large dumps that describes how to split large dumps, for example, with **makedumpfile**.

Chapter 3. Contacting IBM Support

IBM Support provides assistance with product defects, answers FAQs, and helps users resolve problems with the product.

Before you begin

After trying to find your answer or solution by using other self-help options such as technotes, you can contact IBM Support. Before contacting IBM Support, your company or organization must have an active IBM software maintenance agreement (SWMA), and you must be authorized to submit problems to IBM.

Procedure

To contact IBM Support about a problem:

1. Define the problem, gather background information, and determine the severity of the problem.
2. Gather diagnostic information.
3. Submit the problem to IBM Support in one of the following ways:
 - Using IBM Support Assistant (ISA):
 - Online through the [IBM Support Portal](#): You can open, update, and view all your service request notifications here.
 - By phone: For the phone number to call in your region, see the [Directory of worldwide contacts web page](#).

Results

If the problem that you submit is for a software defect, IBM Support creates a software patch. Missing or inaccurate documentation is normally corrected in the next documentation update. The patch is sent to the Linux distributor for inclusion. Whenever possible, IBM Support provides a workaround that you can implement until the patch is available.

Chapter 4. Exchanging information with IBM

To diagnose or identify a problem, you might need to provide IBM Support with data and information from your system. In other cases, IBM Support might provide you with tools or utilities to use for problem determination.

Sending information to IBM Technical Support

To reduce the time that is required to resolve your problem, you can send trace and diagnostic information to IBM Technical Support.

Procedure

To submit diagnostic information to IBM Technical Support:

1. Open a support case.
2. Collect the diagnostic data that you need. Diagnostic data helps reduce the time that it takes to resolve your case.

See the following topics:

- [“Collecting data for general Linux on IBM Z problems” on page 3](#)
- [“Collecting data for performance and resource utilization problems” on page 4](#)
- [“Collecting data for network problems” on page 4](#)
- [“Collecting data for hung system problems” on page 5](#)
- [“Collecting data for middleware problems” on page 5.](#)

3. Compress the files by using the `.zip` or `.tar` file format.

4. Transfer the files to IBM.

You can use one of the following methods to transfer the files to IBM:

- Standard data-upload methods: FTPS, SFTP, and HTTPS

There are two servers available for uploading data:

- `testcase.boulder.ibm.com` (US only)
- `ecurep.ibm.com` (international)

For upload instructions, see *Enhanced Customer Data Repository (ECuRep) - Send data*, available at <https://www.ibm.com/support/docview.wss?uid=ibm10739631>.

- Secure data upload methods: FTPS, SFTP, and HTTPS
- [IBM Support Assistant](#)

All of these data exchange methods are explained on the [IBM Technical Support website](#).

Receiving information from IBM Support

Occasionally an IBM technical-support representative might ask you to download diagnostic tools or other files. You can use FTP to download these files.

Before you begin

Ensure that your IBM technical-support representative provided you with the preferred server to use for downloading the files and the exact directory and file names to access.

Procedure

To download files from IBM Support:

1. Use FTP to connect to the site that your IBM technical-support representative provided and log in as anonymous. Use your email address as the password.
2. Change to the appropriate directory:
 - a) Change to the `/fromibm` directory.

```
cd fromibm
```

- b) Change to the directory that your IBM technical-support representative provided.

```
cd nameofdirectory
```

3. Enable binary mode for your session.

```
binary
```

4. Use the **get** command to download the file that your IBM technical-support representative specified.

```
get filename.extension
```

5. End your FTP session.

```
quit
```


Appendix A. How to detect guest relocation

Information about guest relocations are stored in the s390 debug feature (s390dbf). You can access this information in a kernel dump or from a running Linux instance.

About this task

You can detect if a Linux instance has been moved to another guest virtual machine or LPAR. One available mechanism for guest relocation is z/VM Single System Image (SSI).

You can access the s390 debug feature `lgr` from a live system or with the **crash** tool from a kernel dump. When the debug feature contains only one entry, no relocation has been detected and the entry identifies the boot virtual guest machine. For each detected relocation one additional entry is written.

Procedure

Choose the method that suits your purpose:

- Use the **crash** tool to read from a kernel dump. Issue a command of this form:

```
# crash <vmlinux files> <dump file>
crash> s390dbf lgr hex_ascii
```

- Use the **cat** command on a live system to read the debugfs entry for `lgr`. Issue a command of this form:

```
# cat /sys/kernel/debug/s390dbf/lgr/hex_ascii
```

Example

Assume that one relocation of the guest virtual machine ZVMGUEST has been detected from a z/VM in LPAR VM000A to a z/VM in LPAR VM000B. You can see this in the kernel dump:

```
# crash vmlinux dump
crash> s390dbf lgr hex_ascii
00 01317816806:277332 3 - 00 .. | ... IBM2817000000000000EAA1402 ... VM000A ... ZVMGUEST
00 01317866806:277332 3 - 00 .. | ... IBM2817000000000000EAA1402 ... VM000B ... ZVMGUEST
```

Alternatively, you can see such a relocation from a running system:

```
# cat /sys/kernel/debug/s390dbf/lgr/hex_ascii
00 01317816806:277332 3 - 00 .. | ... IBM2817000000000000EAA1402 ... VM000A ... ZVMGUEST
00 01317866806:277332 3 - 00 .. | ... IBM2817000000000000EAA1402 ... VM000B ... ZVMGUEST
```

For more information about the complete s390dbf record, see the `struct lgr_info` definition in the Linux kernel source code.

For network problems with a relocated guest, see <http://www.ibm.com/vm/virtualnetwork/lgrgrat.html>.

Accessibility

Accessibility features help users who have a disability, such as restricted mobility or limited vision, to use information technology products successfully.

Documentation accessibility

The Linux on IBM Z and LinuxONE publications are in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when you use the PDF file and want to request a Web-based format for this publication send an email to eservdoc@de.ibm.com or write to:

IBM Deutschland Research & Development GmbH
Information Development
Department 3282
Schoenaicher Strasse 220
71032 Boeblingen
Germany

In the request, be sure to include the publication number and title.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

IBM and accessibility

See the IBM Human Ability and Accessibility Center for more information about the commitment that IBM has to accessibility at

www.ibm.com/able

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml

Adobe is either a registered trademark or trademark of Adobe Systems Incorporated in the United States, and/or other countries.

The registered trademark Linux is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Index

Special Characters

[/debug, mount point 7](#)
[/proc, mount point 7](#)
[/sys, mount point 7](#)
[/sys/kernel/debug, mount point 7](#)

A

[accessibility 35](#)
[assumptions 7](#)
authority
 [root 7](#)

C

checklists
 troubleshooting
 [Linux on Z 2](#)
collecting data for general Linux on Z problems [3](#)
commands, Linux
 [scsi_logging_level 22](#)
conventions [7](#)

D

[DASD dump tool 26](#)
[DASD statistics 14](#)
dasdstat
 [troubleshooting tool 15](#)
data
 [for general Linux on Z problems 3](#)
dbginfo.sh [7](#)
diagnostic data
 [collecting for hung system problems 5](#)
 [collecting for middleware problems 5](#)
 [collecting for network problems 4](#)
 [collecting for performance problems](#)
 [starting sadc/sar directly 12](#)
 [collecting with sadc 11](#)
 [sending to IBM 31](#)
dump tools
 [DASD 26](#)
 [kdump 26](#)
 [SCSI 26](#)
 [tape 26](#)
 [VMDUMP 26](#)

G

[general Linux on Z problems](#)
 [collecting data for 3](#)
[guest relocation 33](#)

H

hung system
 troubleshooting
 [collecting diagnostic data 5](#)
hyptop [24](#)

I

iostats
 [troubleshooting tool 13](#)

K

[kdump 26](#)

L

[Linux on Z](#)
 [troubleshooting checklist 2](#)

M

middleware
 troubleshooting
 [collecting diagnostic data 5](#)
[MONWRITE 14](#)
mount point
 [debugfs 7](#)
 [procfs 7](#)
 [sysfs 7](#)

N

network
 troubleshooting
 [collecting diagnostic data 4](#)

P

performance
 [statistics, QDIO 20](#)
performance problems
 [collecting data with sadc 11](#)
 [collecting diagnostic data 4](#)
problem determination
 [exchanging information with IBM Support 31](#)
[ps 25](#)

Q

[QDIO performance](#)
 [statistics 20](#)

R

relocation
 guest [33](#)
root
 authority [7](#)

S

s390dbf [20](#)
sadc
 starting as service [11](#)
 troubleshooting tool [11](#)
sar
 starting as service [11](#)
SCSI dump tool [26](#)
scsi_logging_level, Linux command [22](#)
sending data to IBM [31](#)
socket statistics [25](#)
sos report [10](#)
ss, socket statistics [25](#)
starting sadc/sar directly
 troubleshooting
 collecting diagnostic data [12](#)
statistics
 QDIO performance [20](#)
supportconfig [9](#)

T

tape dump tool [26](#)
tcpdump [26](#)
top [23](#)
troubleshooting
 checklist
 Linux on Z [2](#)
 collecting data for general Linux on Z problems [3](#)
 collecting data for hung system problems [5](#)
 collecting data for middleware problems [5](#)
 collecting data for network problems [4](#)
 collecting data for performance problems
 starting sadc/sar directly [12](#)
 collecting data with sadc [11](#)
 collecting performance data [4](#)
 general tools [7](#)
 identifying problems, techniques for [1](#)
 Linux on z [1](#)
 performance tools [11](#)
 sending data to IBM [31](#)
 special tools [20](#)
 tools
 sadc [11](#)
troubleshooting and support
 contacting IBM Support [29](#)
 exchanging information with IBM Support [31](#)
 troubleshooting techniques [1](#)
troubleshooting tools
 DASD statistics [14](#)
 dasdstat [15](#)
 dbginfo.sh [7](#)
 dump tools [26](#)
 hyptop [24](#)
 iostats [13](#)

troubleshooting tools (*continued*)

 MONWRITE [14](#)
 ps [25](#)
 s390dbf [20](#)
 sos report [10](#)
 ss [25](#)
 supportconfig [9](#)
 tcpdump [26](#)
 top [23](#)
 ziomon [16](#)
 ziorep [17](#)
 ziorep_config [18](#)
 ziorep_traffic [19](#)
 ziorep_utilization [18](#)

V

VMDUMP [26](#)

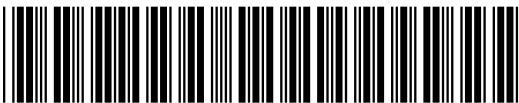
Z

ziomon [16](#)
ziorep [17](#)
ziorep_config [18](#)
ziorep_traffic [19](#)
ziorep_utilization [18](#)



Part Number:

SC34-2612-07



(1P) P/N: