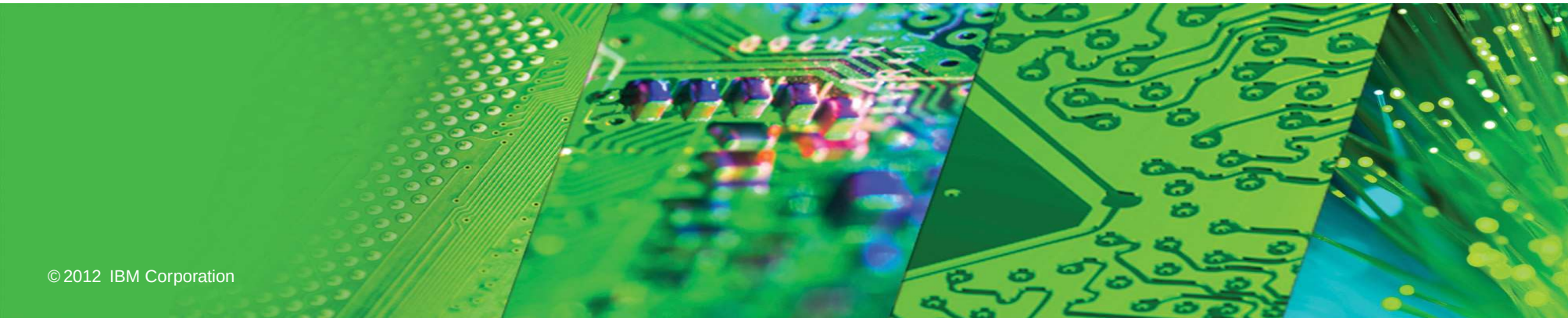
2012
IBM System z Technical University

Enabling the infrastructure for smarter computing

Introducing the Linux Health Checker

zLG29

Dr. Stefan Reimbold



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.

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply. All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions. This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area. All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

Is your Linux Healthy ?

- Definition of health:
  Health is the level of functional or metabolic efficiency of a living being.
  In humans, it is the general condition of a person's mind, body and spirit, usually meaning to be free from illness, injury or pain.

  source: http://en.wikipedia.org/wiki/Health

How Healthy is Your System ?

- health checks help you to maintain and increase health of your Linux instances
- health checks provide you with expert knowledge

Health Check

- What is a health check?
- How does it work?
- Can you show me an example?
- What is this new health care package for Linux instances?
- Why do your Linux instances need health care?
- How can you manage health?

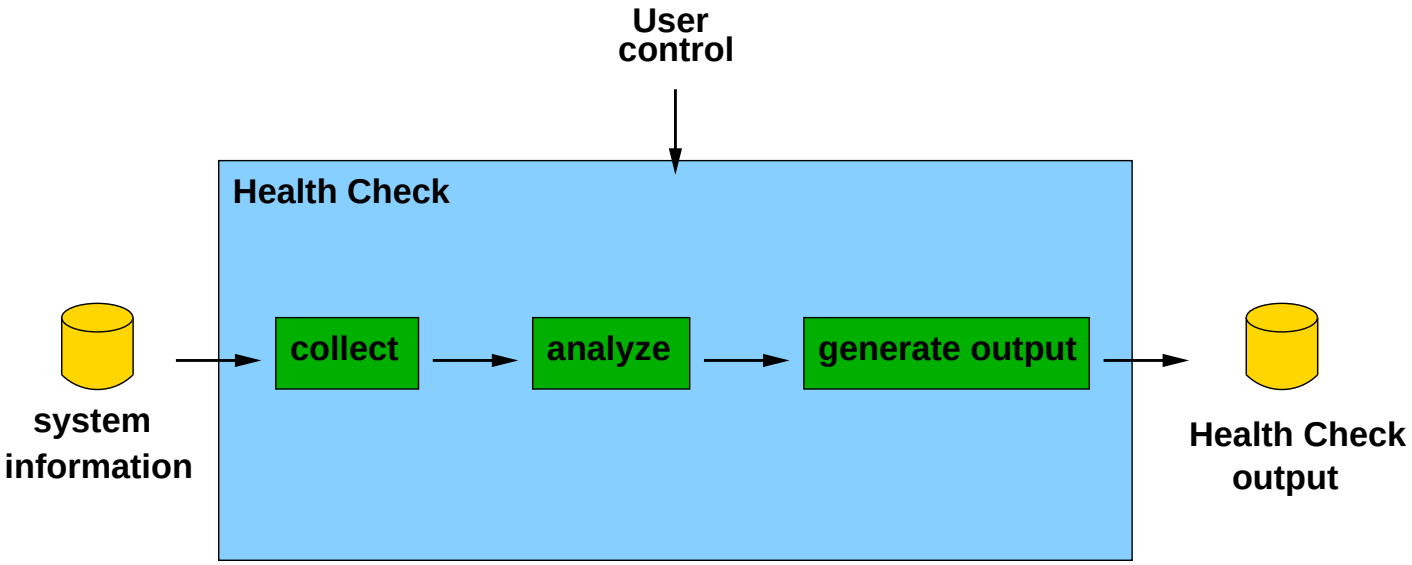
Linux HealthChecker

- check system configuration and status against best practices
- find potential problems before they cause an outage or affect performance
- identify settings that can be optimized
- report findings through exception messages
- Examples
  - configuration errors
  - deviations from best-practices
  - available hardware that is not exploited
  - single point-of-failures

How Can I Get it ?

- http://lnxhc.sourceforge.net
  - prebuild RPM packages
  - source files

How Does the Program Work ?



- collect system information
- analyze collected sysinfo data
- generate output

Example

- detect channel pathes which are not available

```
# lscss
Device Subchan. DevType CU Type Use PIM PAM POM CHPIDs
------------------------------------------------------------------------------
0.0.4d64 0.0.0003 3390/0c 3990/e9 yes c0 c0 ff 34400000 00000000
0.0.4f2a 0.0.0016 3390/0c 3990/e9 yes ff c0 ff 3440494b 50515253
```

```
Device 0.0.4f2a has CHPIDs which are installed but not available.
```

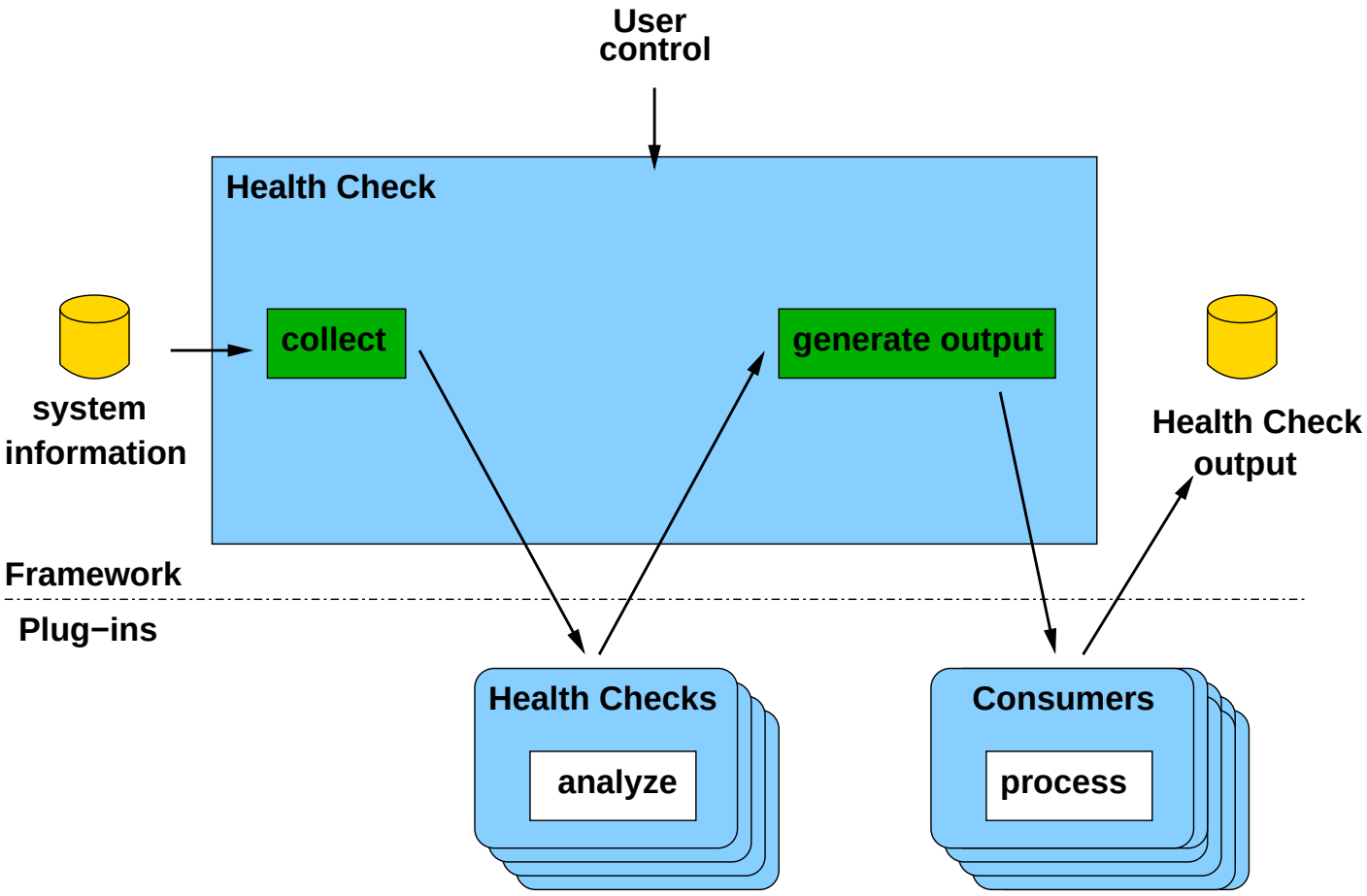
What is Health Care for Linux Instances ?

- health care for Linux instances means
  - collect data for health checks
  - run health checks to analyze the health data of Linux instances
  - inform the user about the result
- use the Linux Health Checker to manage these tasks

What is Health Care for Linux Instances ?

- Make Linux expert knowledge available to a wider audience
  - provide detailed messages
  - allow users to make informed decisions
- prevent problems
  - outages
  - performance degradation
- extend health care across IBM mainframe operating systems
  - z/OS Health Checker, z/VSE Health Checker and Linux Health Checker

How Does Linux Health Checker Work ?



What Can Linux Health Checker ?

- manage installed health checks
  - display health checks
  - modify check parameters
  - run health checks
- manage installed result consumers
- manage stored system information
- manage configuration profiles
  - create profiles with different health checks and check parameter settings
- develop own health checks and help other users

Requirements

- Linux Health Checker requires
  - Perl - version 5.8 or later
  - additional perl modules which are part of standard Linux distributions
- some health check modules might have additional software requirements

What makes it different ?

- DBGINFO
- monitoring
- health checking is like a medical check-up
  - analyzes current configuration and status
  - identifies weaknesses
  - presents you with actions to take before problems might occur
- monitoring is like a long-term ECG
  - observes selected data points in your system over time
  - discovers trends and otherwise interpret the results
- use health checking and monitoring in combination

Checkers

- Verify that the bootmap file is up-to-date
- Screen users with superuser privileges
- Check whether the path to the OpenSSL library is configured correctly
- Identify network services that are known to be insecure
- Identify unusable I/O devices
- Identify multipath setups that consist of a single path only
- Check for CHPIDs that are not available
- Confirm that automatic problem reporting is activated
- Identify I/O devices that are in use although they are on the exclusion list
- Ensure that panic-on-oops is switched on
- Identify I/O devices that are not associated with a device driver
- Check whether the CPUs run with reduced capacity
- Check for an excessive number of unused I/O devices

Checkers

- Spot getty programs on the /dev/console device
- Check Linux on z/VM for the ”nopav” DASD parameter
- Identify unused terminals (TTY)
- Check file systems for adequate free space
- Check file systems for an adequate number of free inodes
- Check whether the recommended runlevel is used and set as default
- Check the kernel message log for out-of-memory (OOM) occurrences
- Identify bonding interfaces that aggregate qeth interfaces with the same CHPID
- Check for an excessive error ratio for outbound HiperSockets traffic
- Check the inbound network traffic for an excessive error or drop ratio
- Identify qeth interfaces that do not have an optimal number of buffers
- Confirm that the dump-on-panic function is enabled

Example

```
# lnxhc run
Creating user directory '/root/.lnxhc'
Collecting system information
Changing user to 'root' for command '/sbin/multipath -ll'
Running checks (24 checks)
CHECK NAME                                HOST                  RESULT
=============================================================================
boot_zipl_update_required ............... r3515039              SUCCESS
css_ccw_availability .................... r3515039              SUCCESS
css_ccw_chpid ........................... r3515039              SUCCESS
css_ccw_ignored_online .................. r3515039              SUCCESS
css_ccw_no_driver ....................... r3515039              SUCCESS
css_ccw_unused_devices .................. r3515039              EXCEPTION-LOW

 >EXCEPTION css_ccw_unused_devices.many_unused_devices(low)
    Of 24 I/O devices, 19 (79.17%) are unused

dasd_zvm_nopav .......................... r3515039              SUCCESS
fs_disk_usage ........................... r3515039              SUCCESS
fs_inode_usage .......................... r3515039              SUCCESS
init_runlevel ........................... r3515039              SUCCESS
mm_oom_killer_triggered ................. r3515039              SUCCESS
net_bond_dev_chpid ...................... r3515039              NOT APPLICABLE
net_hsi_tx_errors ....................... r3515039              NOT APPLICABLE
net_inbound_packets ..................... r3515039              SUCCESS
```

Example

```
net_qeth_buffercount ..................... r3515039                EXCEPTION-MED

 >EXCEPTION net_qeth_buffercount.inefficient_buffercount(medium)
    These  network  interfaces do not have the expected number of
    buffers: eth0

ras_dump_on_panic ........................ r3515039                EXCEPTION-HIGH

 >EXCEPTION ras_dump_on_panic.no_standalone(high)
    The dump-on-panic function is not enabled

sec_non_root_uid_zero .................... r3515039                SUCCESS
sec_services_insecure .................... r3515039                SUCCESS
storage_invalid_multipath ................ r3515039                NOT APPLICABLE
sys_sysctl_call_home ..................... r3515039                NOT APPLICABLE
sys_sysctl_panic ......................... r3515039                EXCEPTION-MED

 >EXCEPTION sys_sysctl_panic.no_panic_on_oops(medium)
    The panic-on-oops setting is disabled

sys_sysinfo_cpu_cap ...................... r3515039                SUCCESS
sys_tty_console_getty .................... r3515039                SUCCESS
sys_tty_usage ............................ r3515039                EXCEPTION-MED

 >EXCEPTION sys_tty_usage.unused_ttys(medium)
    These terminals are unused: /dev/hvc, /dev/ttyS
```

Running Health Checker

- subcommands
  - check
    display, configure and manage health checks
  - consumer
    display, configure and manage consumers
  - devel
    access support functions for developing new health check plug-ins
  - profile
    display, modify and manage configuration profiles
  - run
    run health checks
  - sysinfo
    display and manage health check input data
    called system information

Running Health Checker

- seperate man pages for subcommands
  - `lnxhc-check(1)`
  - `lnxhc-consumer(1)`
  - `lnxhc-devel(1)`
  - `lnxhc-profile(1)`
  - `lnxhc-run(1)`
  - `lnxhc-sysinfo(1)`
- additional documentation
  - `lnxhc_writing_checks(7)`
  - `lnxhc_check_definitions(5)`
  - `lnxhc_check_descriptions(5)`
  - `lnxhc_check_exceptions(5)`
  - `lnxhc_check_program(7)`

Analyze Multiple Instances

- a single host can perform analysis for multiple remote hosts
- ensure the Linux Health Checker is installed and configured on all hosts
- on each remote host, collect system information

```
root@remote1:~# lnxhc sysinfo --collect --file remote1.sysinfo
root@remote2:~# lnxhc sysinfo --collect --file remote2.sysinfo
```

- transfer the system information data to a central host, for example, with scp
- on the central host, run the Linux Health Checker

```
root@remote2:~# lnxhc run --file remote1.sysinfo --file remote2.sysinfo
```

Problems Reported

- configuration errors
- deviation from best practices
- hardware running at reduced capacity
- unused accelerator hardware
- single point of failures

How Do I Interpret Results ?

- success
  check ran and found no problem
- exception
  check ran and found problems
- not applicable
  Check did not run because a requirement was not met
- failed sysinfo and failed chkprg
  Check did not run because system information could not be collected or there was a runtime error in the health check program

Detailed Problem Report

```
CHECK NAME                                HOST                    RESULT
=========================================================================
ras_dump_on_panic ..................... r3515039                EXCEPTION-HIGH

 >EXCEPTION ras_dump_on_panic.no_standalone(high)

  SUMMARY
    The dump-on-panic function is not enabled

  EXPLANATION
    Your Linux instance is not configured for dump-on-panic.

    Configure  dump-on-panic  to automatically create a dump if a
    kernel panic

    occurs.

  SOLUTION
    To configure dump-on-panic, complete these steps:

     1. Plan and prepare your dump device.
     2. Edit /etc/sysconfig/dumpconf and configure  the  dump-on-
    panic action.
        Possible actions are dump, dump_reipl, or vmcmd with a CP
    VMDUMP command.
     3. Activate the dumpconf service  with  chkconfig  and  then
    start the service.
```

Detailed Problem Report

```
Check results:                Exceptions:                 Run-time:
  SUCCESS.......: 0             High.........: 1            Min per check.: 0.014s
  EXCEPTION.....: 1             Medium.......: 0            Max per check.: 0.014s
  NOT APPLICABLE.: 0            Low..........: 0            Avg per check.: 0.014s
  FAILED SYSINFO.: 0            Total........: 1            Total........: 0.030s
  FAILED CHKPROG.: 0
  Total.........: 1
```

Health Check Information

```
# lnxhc check --info fs_disk_usage
Check fs_disk_usage (active)
============================
Title:
  Check file systems for adequate free space

Description:
  Some applications and administrative tasks require an adequate amount of free
  space on each mounted file system. If there is not enough free space, these
  applications might no longer be available or the complete system might be
  compromised. Regular monitoring of disk space usage averts this risk.

Exceptions:
  critical_limit=high (active)
  warn_limit=low (inactive)

Parameters:
  critical_limit=95
        File system usage (in percent) at which to raise a high-severity
        exception. Valid values are integers in the range 1 to 100.

        Default value is "95".
...
```

Configuration Error

```
# lnxhc check --param critical_limit=60 fs_disk_usage
Setting value of parameter fs_disk_usage.critical_limit to '60'
Done.
# lnxhc run fs_disk_usage
Collecting system information
Running checks (1 checks)
CHECK NAME                                HOST                    RESULT
=============================================================================
fs_disk_usage ........................... r3515039                SUCCESS

1 checks run, 0 exceptions found (use 'lnxhc run --replay -V' for details)
```

Configuration Error

```
# lnxhc check --param critical_limit=30 fs_disk_usage
Setting value of parameter fs_disk_usage.critical_limit to '30'
Done.
# lnxhc run fs_disk_usage
Collecting system information
Running checks (1 checks)
CHECK NAME                                HOST                    RESULT
=========================================================================
fs_disk_usage ........................... r3515039                EXCEPTION-HIGH

 >EXCEPTION fs_disk_usage.critical_limit(high)
    The critical threshold of 30% disk space usage is exceeded on
    some file systems (/ 39%)


1 checks run, 1 exceptions found (use 'lnxhc run --replay -V' for details)
```

Configuration Error

```
# df
Filesystem      1K-blocks     Used Available Use% Mounted on
/dev/dasda1       6309976  2285092   3704348  39% /
devtmpfs           510204      148    510056   1% /dev
tmpfs              510204        0    510204   0% /dev/shm
```

How to write a check

```
# lnxhc devel --create-check ./my_check
Health check creation dialog
============================
This dialog supports the creation of a new health check. It queries the user
for answers to several questions. Once the dialog is finished, a directory
containing a skeleton of files will be created.

Some questions provide default answers which are shown in square brackets
("[]"). These answers are used if an empty value is entered. All answers can
be modified at the end of the dialog.

The following input options are available to control the dialog:
  ?.......: show help text for the current dialog question
  CTRL-C..: save data and end dialog, restart the dialog to continue

Generic health check characteristics
====================================

What programming language will be used to implement the check program?
(1..5)

 1..Perl
 2..Bash
 3..C
 4..Other scripting language
 5..Other compiled language
1
```

How to write a check

```
Enter the name and e-mail address of the check author:
Stefan.Reimbold@de.ibm.com

Enter the name of the component that is being checked:
password

Should the check run regularly? (y/n) [n]

Does the check require data from multiple hosts at once? (y/n) [n]

Does the check require data from multiple points in time at once (y/n) [n]

List all paths to additional files provided by the check relative to the check
directory (empty input to continue):

Does the check produce meaningful results with default parameters on a standard
Linux installation? (y/n) [y]

Is the component being checked part of a standard Linux installation? (y/n) [y]
```

How to write a check

```
System information
==================
A health check requires data about a system to perform its check function.
This data must not be collected by the check program itself. Instead you need
to specify this data as so-called "sysinfo items" so that the lnxhc framework
can obtain the data and provide it to the check program.

Enter the ID of a sysinfo item that is required by the check program:
password

What is the type of sysinfo item 'password'?

 1.. File
 2.. Program
 3.. Record
 4.. Reference
 5.. External
1
```

How to write a check

```
Specify the absolute path to the file to be read for file sysinfo item
'password':
/etc/shadow

Specify the user-ID that has access permissions to obtain the data of sysinfo
item 'password' (empty ID if no special permissions are required): []
root

Enter the ID of an additional sysinfo item that is needed by check (empty ID
to continue):
```

How to write a check

```
Exceptions
==========
A problem that can be reported by a health check is called an "exception".
Each health check must be able to report at least one exception.

Enter the ID of an exception that the check can report:
empty

What is the severity of exception 'empty'? (1..3)

 1.. Low
 2.. Medium
 3.. High
3

Enter the ID of an additional exception that the check can report (empty ID to
continue):

Health check parameters
=======================
Parameters are untyped string values which can be modified by users and
which are passed to the health check program. Parameters can be used to
allow users to customize some aspects of health check execution.

Enter the ID of a health check parameter (empty ID to continue):
```

How to write a check

```
Finalization dialog
===================
Below is the summary of information entered for the new check. You can
adjust each data item or finalize the check.

 1. Programming language.....: Perl
 2. Check author.............: Stefan.Reimbold@de.ibm.com
 3. Checked component........: password
 4. Run regularly............: No
 5. Multiple host data.......: No
 6. Multiple time data.......: No
 7. Extra files..............: <empty list>
 8. Works without configuration...: Yes
 9. Works with default software...: Yes
10. Sysinfo item ID..........: password
11.     Type.................: External
12. Exception ID.............: empty
13.     Severity.............: High
14. Parameter ID.............: <empty list>
(...)
Creating check in directory './my_check'.
Check was successfully created.
Use 'lnxhc run ./my_check' to run this check.
Please see each file for specific TODOs.
Done.
```

How to write a check

```
# lnxhc run ./my_check
Collecting system information
Changing user to 'root' for command '/bin/cat /etc/passwd'
Running checks (1 checks)
CHECK NAME                                HOST                    RESULT
=============================================================================
my_check ............................... r3515039                EXCEPTION-HIGH

 >EXCEPTION my_check.empty(high)
    TODO: Write a short summary of the problem which includes all
    relevant information needed by advanced users to implement  a
    solution.


1 checks run, 1 exceptions found (use 'lnxhc run --replay -V' for details)
```

How to write a check

```
# TODO:
# 1. Check parameters for correct values (param\_*).
# 2. Access sysinfo data (filenames available in sysinfo\_*).
# 3. Perform analysis.
# 4. If an exception is found, write its ID and values for exception
#    template variables to file ex\_file.
#
# See 'man lnxhc_check_program' for more information.
#

#
# Sample exception reporting. TODO: call this only if an exception
# was identified.
#

lnxhc_exception($LNXHC_EXCEPTION_EMPTY);
```

How to write a check

```
# TODO:
# 1. Check parameters for correct values (param\_*).
# 2. Access sysinfo data (filenames available in sysinfo\_*).
# 3. Perform analysis.
# 4. If an exception is found, write its ID and values for exception
#    template variables to file ex\_file.
#
# See 'man lnxhc_check_program' for more information.
#

#
# Sample exception reporting. TODO: call this only if an exception
# was identified.
#
my @password = `cat /etc/shadow`;
foreach (@password) {
        chomp;
        if (/^(\w+):(\s*):/) {
                lnxhc_exception($LNXHC_EXCEPTION_EMPTY);
        }
}
```

How to write a check

```
# lnxhc run ./my_check
Collecting system information
Changing user to 'root' for command '/bin/cat /etc/shadow'
Running checks (1 checks)
CHECK NAME                                HOST                    RESULT
=============================================================================
my_check ............................... r3515039                EXCEPTION-HIGH

 >EXCEPTION my_check.empty(high)
    TODO: Write a short summary of the problem which includes all
    relevant information needed by advanced users to implement  a
    solution.


1 checks run, 1 exceptions found (use 'lnxhc run --replay -V' for details)
```

How to write a check

```
# cat ./my_check/exceptions
[summary empty]
TODO: Write a short summary of the problem which includes all relevant
information needed by advanced users to implement a solution.

[explanation empty]
TODO: Write a detailed text containing answers to the following questions:
 - What is the problem?
 - What is the impact on the checked component?
 - What are the steps to manually verify that the problem exists?

[solution empty]
TODO: Write a detailed text describing how the problem can be solved.

[reference empty]
TODO: List references to documentation which can help in understanding and
solving the problem.
```

How to write a check

- Example to look at
  - `boot_zipl_update_required`
  - in directory `/usr/lib/lnxhc/checks`

Summary

- Linux Health Checker provides health care for your Linux instances
  - for Linux on System z and other Linux platforms
- You can use the Linux Health Checker to
  - maintain and increase the health of your Linux instances
  - manage health checks and run them regularly

Summary

- share your expert knowledge and contribute your health checks
  - develop and share your health checks with other users
  - help us to improve health care for Linux instances
- see the lnxhc project for guidelines and how to register as a contributor
  - http://lnxhc.sourceforge.net/

Links

- Project page of the Linux Health Checker on SourceForge
  `http://lnxhc.sourceforge.net`
- Linux Health Checker User's Guide
  `http://www.ibm.com/developerworks/linux/linux390/documentation_dev.html`
- Linux on System z - Tuning Hints & Tips
  `http://www.ibm.com/developerworks/linux/linux390/perf/index.html`
- developerWorks
  `http://www.ibm.com/developerworks/linux/linux390`

Thank You !

For starting out with their very good presentations

- Peter Oberparleiter
- Hendrik Brückner

Questions ?

Dr. Stefan Reimbold
Diplom-Physiker

Linux on System z Service

Schoenaicher Strasse 220
D-71032 Boeblingen
Mail: Postfach 1380
D-71003 Boeblingen

Phone +49-7031-16-2368
Stefan.Reimbold@de.ibm.com

boot_zipl_update_required

```
Check boot_zipl_update_required (active)
========================================
Title:
  Verify that the bootmap file is up-to-date

Description:
  With a backlevel bootmap file, you might no longer be able to boot your Linux
  instance.

  This check compares the file metadata to verify that none of the boot data
  that is referenced by the bootmap file has been modified after the bootmap
  file was created. The boot data typically includes, a kernel image, initial
  RAM disk (initrd), and a kernel parameter file.

  A backlevel bootmap file can be the result of upgrading the kernel with a new
  kernel image without running "zipl" to update the bootmap file accordingly.

  This check applies only if the following assumptions are all true: - The boot
  device is a disk device. - The bootmap file has been created from
  specifications in the "zipl" configuration file, /etc/zipl.conf. -
  /etc/zipl.conf describes a single boot configuration that can but need not
  provide a boot menu.

  Distributions tools typically use "zipl" according to these assumptions when
  creating a boot disk.

Exceptions:
  outdated_bootmap=medium (active)
```

crypto_openssl_ibmca_config

```
Check crypto_openssl_ibmca_config (inactive)
============================================
Title:
  Check whether the path to the OpenSSL library is configured correctly

Description:
  If the libibmca.so path is not specified correctly in the openssl.cnf
  configuration file, "ssh" commands fail. An incorrect specification can also
  prevent logins to Linux.

Exceptions:
  so_file_path_not_correct=high (active)
```

css_ccw_availability

```
Check css_ccw_availability (active)
===================================
Title:
  Identify unusable I/O devices

Description:
  This check examines sysfs information to identify I/O devices for which the
  availability status indicates that they cannot be used.

Exceptions:
  unusable_device=high (active)
```

css_ccw_chpid

```
Check css_ccw_chpid (active)
============================
Title:
  Check for CHPIDs that are not available

Description:
  Unavailable CHPIDs can cause I/O stalls and errors and might result in
  required I/O devices that are not visible within Linux. This check analyzes
  sysfs status information to identify CHPIDs that are unavailable because of a
  "configure standby" or a "vary offline" operation. These operations are
  commonly performed as part of hardware maintenance procedures and need to be
  reverted after maintenance has finished.

Exceptions:
  unused_cfg_off=low (active)
  unused_vary_off=low (active)
  used_cfg_off=high (active)
  used_vary_off=high (active)
```

css_ccw_ignored_online

```
Check css_ccw_ignored_online (active)
=====================================
Title:
  Identify I/O devices that are in use although they are on the exclusion list

Description:
  The I/O device exclusion list prevents Linux from sensing and analyzing I/O
  devices that are available to Linux but not required.

  An initial exclusion list can be included in the boot configuration using the
  "cio_ignore" kernel parameter. On a running Linux instance, the list can be
  changed temporarily through the /proc/cio_ignore procfs interface or with the
  "cio_ignore" command. Rebooting restores the exclusion list of the boot
  configuration.

  I/O devices that are in use (online) might be required and should then not be
  on the exclusion list. If these devices become unavailable and reappear after
  some time, they are ignored and remain unavailable to Linux. If they are added
  to the cio_ignore parameter in the boot configuration, they will also be
  unavailable after rebooting Linux.

Exceptions:
  online_devices_ignored=medium (active)
```

css_ccw_no_driver

```
Check css_ccw_no_driver (active)
================================
Title:
  Identify I/O devices that are not associated with a device driver

Description:
  When an I/O device is sensed, the associated device driver should
  automatically be loaded. I/O devices that are not associated with a device
  driver cannot be used properly.

  Possible reasons for this problem are that the required device driver module
  has been unloaded, that an existing association between the device and the
  device driver has been removed, or that the device is not supported.

  This check identifies devices that, in sysfs, do not have a symbolic link to a
  device driver.

Exceptions:
  no_driver=medium (active)
```

css_ccw_unused_devices (1)

```
Check css_ccw_unused_devices (active)
=====================================
Title:
  Check for an excessive number of unused I/O devices

Description:
  Even when they are unused (offline), I/O devices consume memory and CPU time
  both during the boot process and when I/O configuration changes occur on a
  running system. In particular, when new I/O devices or I/O paths become
  available or when existing I/O devices or I/O paths become unavailable,
  resources are wasted to unused I/O devices.

  This check uses the "lscss" command to identify unused I/O devices.

Exceptions:
  many_unused_devices=low (active)
```

css_ccw_unused_devices (2)

```
Parameters:
  device_print_limit=5
        Threshold for the absolute number of unused (offline) I/O devices. If
        the number of unused I/O devices exceeds this threshold, an exception is
        issued. Valid values are positive integers.

        Default value is "5".

  ratio_limit=50
        Threshold for the percentage of unused (offline) I/O devices. If this
        threshold is exceeded, an exception is issued. Valid values are integers
        in the range 1 to 100.

        Default value is "50".
```

dasd_zvm_nopav

```
Check dasd_zvm_nopav (active)
=============================
Title:
  Check Linux on z/VM for the "nopav" DASD parameter

Description:
  This check examines the Linux on z/VM configuration for occurrences of the
  "nopav" DASD kernel or module parameter. The "nopav" parameter disables
  parallel access volume (PAV and HyperPAV) for Linux in LPAR mode but has no
  effect for Linux on z/VM. For Linux on z/VM you cannot disable PAV through
  Linux settings; configuration steps on z/VM are required instead.

Exceptions:
  nopav_in_dasd_param=low (active)
```

fs_disk_usage (1)

```
Check fs_disk_usage (active)
============================
Title:
  Check file systems for adequate free space

Description:
  Some applications and administrative tasks require an adequate amount of free
  space on each mounted file system. If there is not enough free space, these
  applications might no longer be available or the complete system might be
  compromised. Regular monitoring of disk space usage averts this risk.

Exceptions:
  critical_limit=high (active)
  warn_limit=low (inactive)
```

fs_disk_usage (2)

```
Parameters:
  critical_limit=30
        File system usage (in percent) at which to raise a high-severity
        exception. Valid values are integers in the range 1 to 100.

        Default value is "95".

  mount_points=
        A list of mount points, separated by colons (:). The file systems
        mounted at the specified mount points are to be checked for free space.
        If the list is empty, all mounted file systems are checked.

        Example:

        /mnt:/home/mymnt/usr/data/myapp

        Default value is "".

  warn_limit=80
        File system usage (in percent) at which to raise a low-severity
        exception. Valid values are integers in the range 1 to 100.

        Default value is "80".
```

fs_inode_usage (1)

```
Check fs_inode_usage (active)
=============================
Title:
  Check file systems for an adequate number of free inodes

Description:
  Many Linux file systems maintain metadata about file system objects (for
  example, files or folders) in inodes. Each object has a separate inode. When a
  file system runs out of free inodes, no further files or folders can be
  created, even if plenty of free disk space is available.

  Some applications and administrative tasks require an adequate number of free
  inodes on each mounted file system. If there are not enough free inodes, these
  applications might no longer be available or the complete system might be
  compromised. Regular monitoring of inode usage can avert this risk.

Exceptions:
  critical_limit=high (active)
  warn_limit=low (inactive)
```

fs_inode_usage (2)

```
Parameters:
  critical_limit=95
        Usage of the available inodes of the file system (in percent) at which
        to raise a high-severity exception. Valid values are integers in the
        range 1 to 100.

        Default value is "95".

  mount_points=
        A list of mount points, separated by colons (:). The file systems
        mounted at the specified mount points are to be checked for free inodes.
        If the list is empty, all mounted file systems are checked. Example:
        /mnt:/home/mymnt:/usr/data/myapp

        Default value is "".

  warn_limit=80
        Usage of the available inodes of the file system (in percent) at which
        to raise a low-severity exception. Valid values are integers in the
        range 1 to 100.

        Default value is "80".
```

init_runlevel

```
Check init_runlevel (active)
============================
Title:
  Check whether the recommended runlevel is used and set as default

Description:
  Running Linux with an unsuitable runlevel can mean that required services are
  not available, or it can mean that unnecessary processes degrade performance
  or security.

  Linux runlevels are usually expressed as integers in the range 0 to 6, where 0
  and 6 are reserved for halt and reboot. The meaning of runlevels 1 to 5 differ
  between distributions. See the "init" man page of your distribution for
  details.

Exceptions:
  current_runlevel=medium (active)
  default_runlevel=medium (active)

Parameters:
  recommended_runlevel=3
        The recommended runlevel for the Linux instance. Valid values are
        integers in the range 1 to 5.

        Default value is "3".
```

mm_oom_killer_triggered

```
Check mm_oom_killer_triggered (active)
======================================
Title:
  Check the kernel message log for out-of-memory (OOM) occurrences

Description:
  When a Linux instance runs out of memory, the OOM killer recovers memory by
  killing one or more processes. If important process get killed, they might
  need to be restarted and protected from the OOM killer.

  Frequent OOM occurrences indicate that too little memory is available for a
  given workload or that an application is consuming an undue amount of memory.
  Awareness of OOM occurrences can disclose resource shortages or help identify
  malfunctioning applications.

Exceptions:
  processes_killed=medium (active)
```

net_bond_dev_chpid

```
Check net_bond_dev_chpid (active)
=================================
Title:
  Identify bonding interfaces that aggregate qeth interfaces with the same CHPID

Description:
  Bonding setups are mainly used to increase availability or performance. A
  bonding interface is a logical interface that aggregates multiple slave
  interfaces. Slave interfaces that are configured with the same CHPID do not
  offer path redundancy or increased bandwidth, so neither goal can be achieved.

Exceptions:
  same_chpid_devices=medium (active)
```

net_hsi_tx_errors

```
Check net_hsi_tx_errors (active)
================================
Title:
  Check for an excessive error ratio for outbound HiperSockets traffic

Description:
  This check examines the transmit (TX) error ratio for HiperSockets network
  interfaces (hsi). A high TX error ratio can be caused by one or more slow
  receivers that require attention.

Exceptions:
  slow_hsi_receivers=medium (active)

Parameters:
  txerror_ratio=1
        Threshold for the percentage of TX errors by total TX packets for
        HiperSockets network interfaces. If the ratio of TX errors exceeds this
        threshold, an exception is raised. Valid values are integers in the
        range 1 to 100.

        Default value is "1".
```

net_inbound_packets (1)

```
Check net_inbound_packets (active)
==================================
Title:
  Check the inbound network traffic for an excessive error or drop ratio

Description:
  This check examines network interfaces for a high received (RX) error ratio or
  high ratio of dropped RX packets. Problems with received packets lead to
  performance degradation as packets have to be resent by the originator. High
  RX error and drop ratios can be caused by insufficient memory.

Exceptions:
  rxpkts_dropped=medium (active)
```

net_inbound_packets (2)

```
Parameters:
  rxdrop_ratio=1
        Threshold for the percentage of dropped RX packets by total RX packets.
        If the ratio of dropped RX packets exceeds this threshold for a network
        interface, an exception message is issued. Valid values are integers in
        the range 1 to 100.

        Default value is "1".

  rxerror_ratio=1
        Threshold for the percentage of RX errors by total RX packets. If the
        ratio of RX errors exceeds this threshold for a network interface, an
        exception message is issued. Valid values are integers in the range 1 to
        100.

        Default value is "1".
```

net_qeth_buffercount (1)

```
Check net_qeth_buffercount (active)
===================================
Title:
  Identify qeth interfaces that do not have an optimal number of buffers

Description:
  The most suitable number of buffers for a particular interface depends on the
  available memory. To allow for memory constraints, many Linux distributions
  use a small number of buffers by default. On Linux instances with ample memory
  and a high traffic volume, this can lead to performance degradation, as
  incoming packets are dropped and have to be resent by the originator. This
  check uses a set of rules that correlate memory size and number of buffers to
  evaluate the settings for each qeth interface.

Exceptions:
  inefficient_buffercount=medium (active)
```

net_qeth_buffercount (2)

```
Parameters:
  recommended_buffercount=<=500MB:16,<=1GB:32,<=2GB:64,>2GB:128
        The rule set used to evaluate the interface settings. The rule set
        comprises a set of comma-separated rules. Each rule specifies a
        particular memory size or implies a range of memory sizes and the number
        of buffers to be used. The rules are evaluated from left to right. The
        first rule that applies to the available memory defines the number of
        buffers demanded by the check.

        Each rule has the form:

        <operator><memsize>:<buffer_count>

        Where: <operator> is one of these comparison operators: == (equal), <=
        (equal or smaller), >= (equal or greater), < (smaller), > (greater)

        <memsize> specifies an amount of memory. Valid values are numbers
        followed by one of the units KB (for kilobyte), MB (for megabyte), or GB
        (for gigabyte).

        <buffer_count> is the number of buffers to be used for the specified
        memory size. Valid values are 16, 32, 64 and 128.

        Example:

        <=500MB:16,<=1GB:32,<=2GB:64,>2GB:128

        The rule set of the example demands 16 buffers if the memory is 500 MB
        or less, 32 buffers if the memory is more than 500 MB but not more than
        1 GB, 64 buffers if the memory is more than 1 GB but not more than 2 GB,
        and 128 buffers if the memory is more than 2 GB.

        Default value is "<=500MB:16,<=1GB:32,<=2GB:64,>2GB:128".
```

ras_dump_on_panic

```
Check ras_dump_on_panic (active)
================================
Title:
  Confirm that the dump-on-panic function is enabled

Description:
  With the dump-on-panic function enabled, a dump is automatically created if a
  kernel panic occurs. Without this function you have to create a dump yourself.
  Dumps can only be created if dump tools and possibly dump devices are in
  place.

Exceptions:
  no_dumpconf=high (active)
  no_kdump=high (active)
  no_kdump_dumpconf=medium (active)
  no_kdump_standalone=low (active)
  no_standalone=high (active)
```

sec_non_root_uid_zero

```
Check sec_non_root_uid_zero (active)
====================================
Title:
  Screen users with superuser privileges

Description:
  This check examines the output of command "getent passwd" to identify user
  names that run with numerical user ID (UID) 0. These users have superuser
  privileges that are conventionally associated with user "root".

  Users with UID 0 and the processes started by these users can inadvertently or
  maliciously disrupt, damage, manipulate, or destroy a system. Generally, UID 0
  must be assigned sparingly and only to trusted user names. Security policies
  often restrict UID 0 to user name "root".

Exceptions:
  non_root_uid0=medium (active)

Parameters:
  trusted_superusers=root
        A list of user names that are trusted to run as superusers with UID 0.
        In the list, the user names are separated by blanks.

        Default value is "root".
```

sec_services_insecure

```
Check sec_services_insecure (active)
====================================
Title:
  Identify network services that are known to be insecure

Description:
  This check finds network services that are active but known to be insecure.
  Such services can compromise your data and system security. An example of an
  insecure network service is a network file system service that does not
  provide user authentication. Any user who can reach this service can access
  the data. Other network services might be considered insecure because they do
  not encrypt credentials and data. If network traffic from such services is
  intercepted, data might be disclosed to unauthorized parties and the system
  might become vulnerable to intrusion.

  Examples of insecure network services are ftp, rsh, rlogin, and telnet.

Exceptions:
  insecure_services=medium (active)

Parameters:
  insecure_services=tftp telnet rsh rlogin
        A list of insecure network services to check for. In the list, services
        are separated by blanks. The default includes the most commonly used
        insecure network services Add any services that are installed on your
        system and that you consider insecure.

        Default value is "tftp telnet rsh rlogin".
```

storage_invalid_multipath

```
Check storage_invalid_multipath (active)
========================================
Title:
  Identify multipath setups that consist of a single path only

Description:
  Through a correctly configured multipath setup, a Linux instance has two or
  more independent connections to the same physical storage device. This path
  redundancy can be used for load balancing and to maintain availability if one
  of the paths fails. Multipath setups with only a single path cannot achieve
  either of these goals.

Exceptions:
  single_path=medium (active)
```

sys_sysctl_call_home

```
Check sys_sysctl_call_home (active)
===================================
Title:
  Confirm that automatic problem reporting is activated

Description:
  When Linux experiences a kernel panic, the automatic problem reporting feature
  sends collected problem data to the IBM service organization Hence a system
  crash automatically leads to a new Problem Management Record (PMR), which can
  be processed by IBM service.

  Omit this check unless a hardware support agreement with IBM is in place and
  the hardware is enabled for the Remote Support Facility.

Exceptions:
  call_home_inactive=low (active)
  call_home_not_avail=low (active)
```

sys_sysctl_panic

```
Check sys_sysctl_panic (active)
===============================
Title:
  Ensure that panic-on-oops is switched on

Description:
  If the Linux instance experiences a kernel oops, the instance can no longer be
  trusted to work correctly. The panic-on-oops setting ensures that the Linux
  instance is stopped if this occurs.

Exceptions:
  no_panic_on_oops=medium (active)
```

sys_sysinfo_cpu_cap (1)

```
Check sys_sysinfo_cpu_cap (active)
==================================
Title:
  Check whether the CPUs run with reduced capacity

Description:
  External events or reconfigurations might cause CPUs to run with reduced
  capacity. This check examines the CPU capacity-adjustment indication and
  capacity-change reason codes of the System z mainframe.

Exceptions:
  reduced_cpu_capacity=high (active)
```

sys_sysinfo_cpu_cap (2)

```
Parameters:
  acceptable_cap_adj=100
        The lowest acceptable CPU capacity-adjustment indication. The default
        value is 100, for regular capacity. Lower values indicate reduced
        capacity. An exception is raised if the System z mainframe reports a
        capacity-adjustment indication below this value.

        Change this value only if your System z mainframe intentionally runs
        with reduced capacity, for example, in power-saving mode. Valid values
        are integers in the range 1 to 100.

        Default value is "100".

  expected_cap_rs=0
        The expected capacity-change reason. The default value is 0, for regular
        operations without capacity changes. An exception is raised if the
        System z mainframe reports a capacity-change reason other than this
        value.

        Change this value to 1 if your System z mainframe runs in power-saving
        mode.

        Default value is "0".
```

sys_tty_console_getty

```
Check sys_tty_console_getty (active)
====================================
Title:
  Spot getty programs on the /dev/console device

Description:
  In Linux, /dev/console is a generic device node that, depending on the
  environment and setup, is mapped to one of the available terminal devices
  (TTY). This terminal device is then represented by its own, specific device
  node and by /dev/console. If getty programs are configured for both device
  nodes, they interfere with each other, so that users cannot log in.

Exceptions:
  prg_on_console=medium (active)
```

sys_tty_usage

```
Check sys_tty_usage (active)
============================
Title:
  Identify unused terminals (TTY)

Description:
  Verify that terminal (TTY) devices are used, for example, by login programs.

  Terminal devices are intended to provide a user interface to a Linux instance.
  Without an associated program, a terminal device does not serve this purpose.

Exceptions:
  unused_ttys=medium (active)

Parameters:
  exclude_tty=tty
        A list of blank-separated terminal devices to be exempt from this check,
        for example, because they are deliberately unused.

        Terminals are specified by their device node without the leading /dev/.
        Use an asterisk (*) to match any string of characters. For example,
        "ttyS3 hvc*" excludes /dev/ttyS3, /dev/hvc0, /dev/hvc1, ...

        Default value is "tty".
```