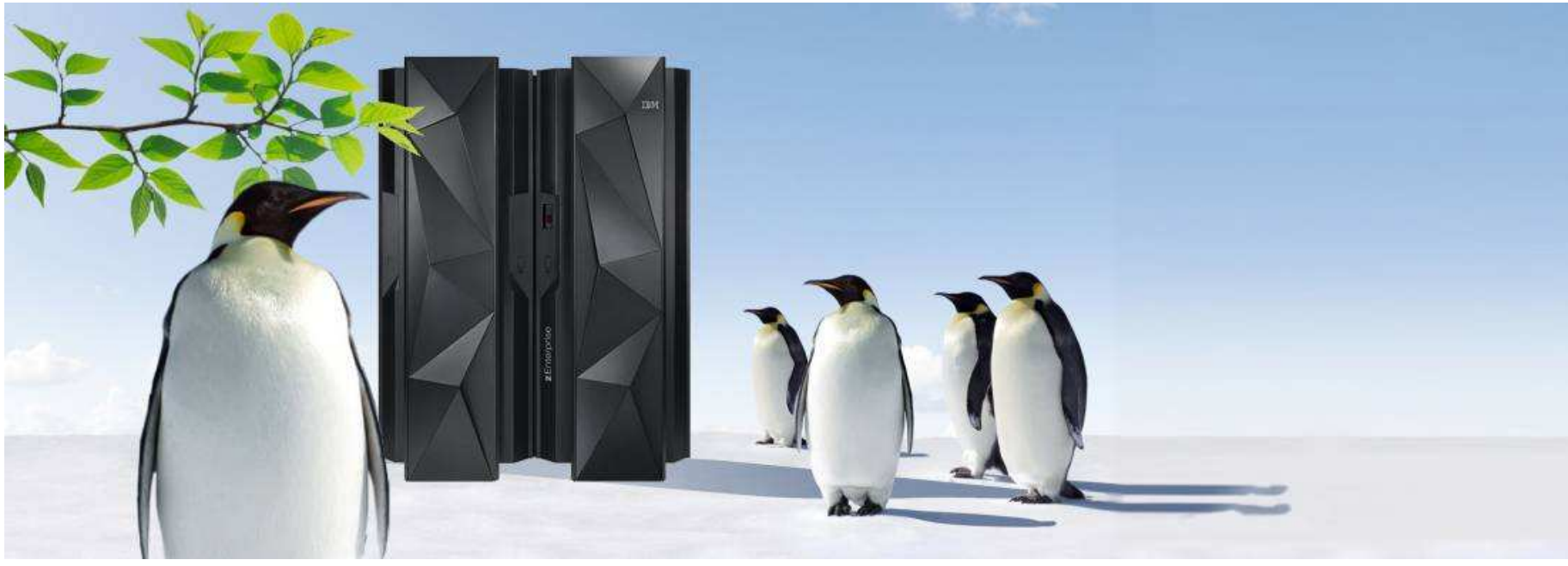
Dr. Stefan Reimbold
IBM Research & Development

Introducing the Linux Health Checker



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?

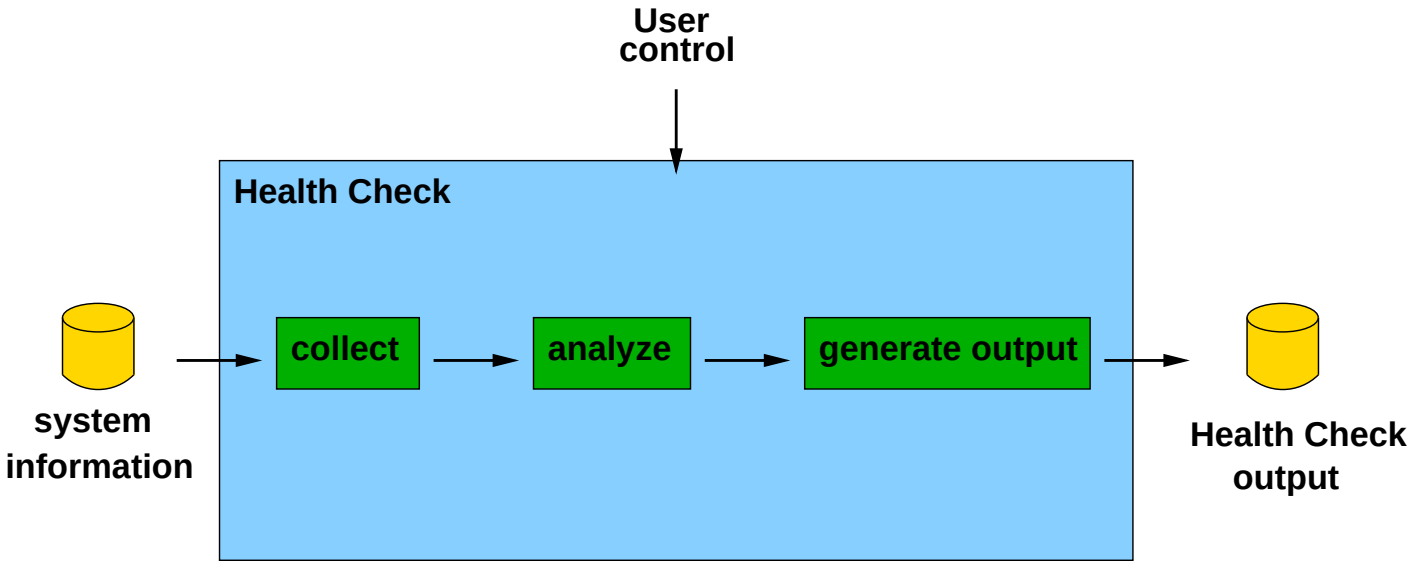
Linux HealthChecker

- 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.
```

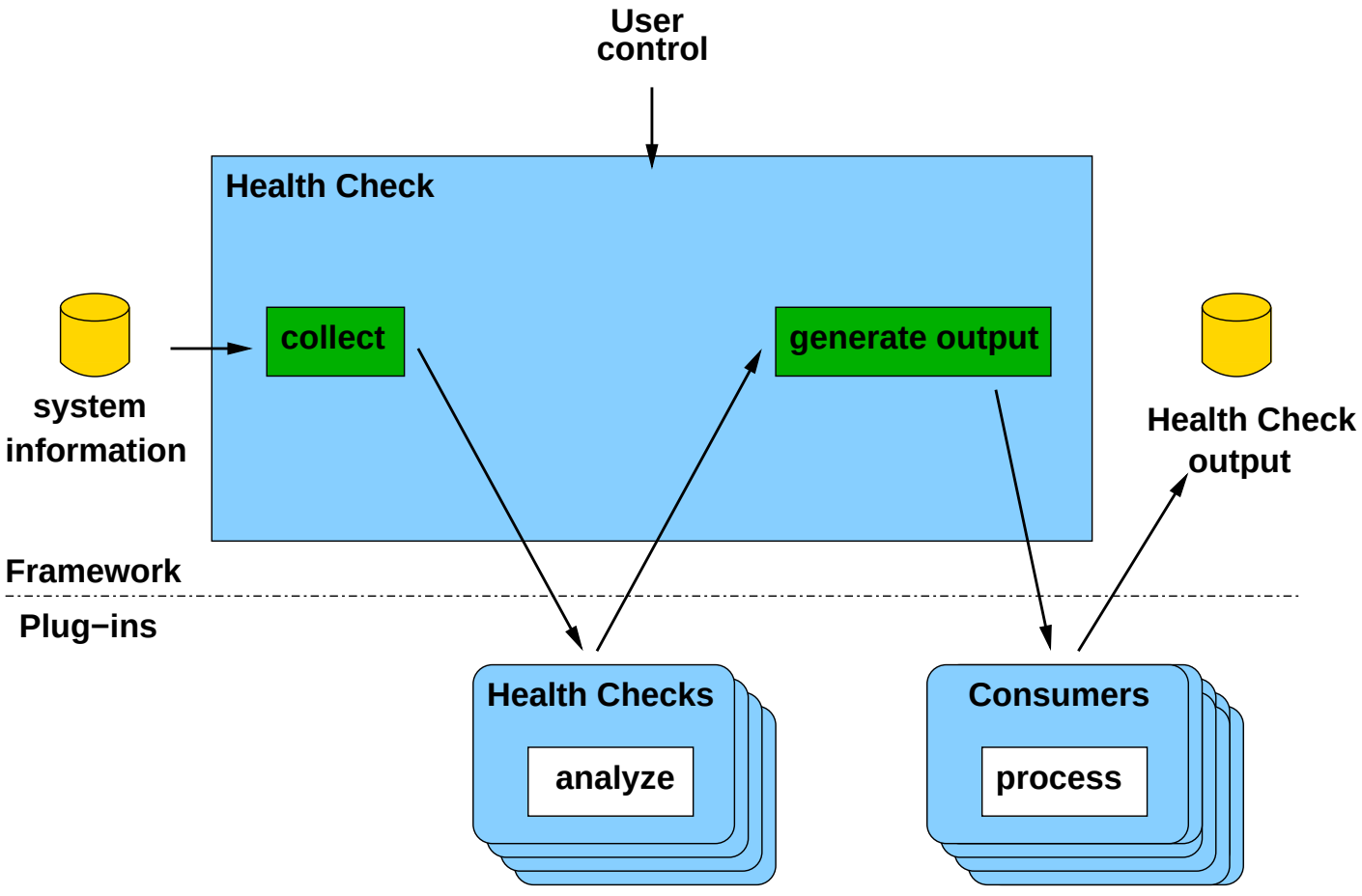
What is Health Care for Linux Instances ?

- 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

- Check whether the recommended runlevel is used and set as default
- Check whether the CPUs run with reduced capacity
- Verify the availability of System z cryptographic hardware support through a Common Cryptographic Architecture (CCA) stack
- Confirm that CPACF is enabled
- Verify the availability of System z cryptographic hardware support for PKCS#11 clear key cryptographic operations
- Verify the availability of System z cryptographic hardware support for PKCS#11 secure key cryptographic operations
- Check whether the path to the OpenSSL library is configured correctly
- Verify the availability of System z cryptographic hardware support through an OpenSSL stack
- Confirm that the System z cryptography kernel module is loaded

Checkers

- Identify I/O devices that are in use although they are on the exclusion list
- Check for CHPIDs that are not available
- Identify unusable I/O devices
- Check for an excessive number of unused I/O devices
- Identify I/O devices that are not associated with a device driver
- Verify that the bootmap file is up-to-date
- Identify standard DASD device nodes in the fstab file
- Check if filesystems are skipped by filesystem check (fsck)
- Check file systems for an adequate number of free inodes
- Check for read-only filesystems
- Verify that temporary files are deleted at regular intervals.
- Check file systems for adequate free space
- Confirm that automatic problem reporting is activated

Checkers

- Check if control program identification can display meaningful Linux instance names
- Verify that syslog files are rotated
- Check if swap space is available
- Ensure memory usage is within the threshold
- Identify bonding interfaces that are configured with single network interfaces
- Identify bonding interfaces that aggregate qeth interfaces with the same CHPID
- Ensure nameserver is listed with correct address
- 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
- Identify network services that are known to be insecure

Checkers

- Ensure processes do not hog cpu time
- Ensure the system is running with optimal load
- Check the kernel message log for out-of-memory (OOM) occurrences
- Ensure processes do not hog memory
- Ensure that privilege dump is switched off
- Ensure kdump is configured and running
- Confirm that the dump-on-panic function is enabled
- Ensure that panic-on-oops is switched on
- Confirm that root logins are enabled for but restricted to secure terminals
- Screen users with superuser privileges
- Identify CDL-formatted DASD where the metadata area is used for storing data
- Confirm 4K block size on ECKD DASD devices
- Check Linux on z/VM for the nopav"DASD parameter

Checkers

- Identify active DASD alias devices without active base device
- Identify multipath setups that consist of a single path only
- Identify multipath devices with too few available paths or too many failed paths
- Spot getty programs on the /dev/console device
- Check for current console_loglevel
- Detect terminals with multiple device nodes
- Confirm that all available z/VM IUCV HVC terminals are enabled for logins
- Identify idle terminals
- Identify idle users
- Identify unused terminals (TTY)
- Check the privilege classes of the z/VM guest virtual machine on which the Linux instance runs

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_runlevel_recommended`
  - 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

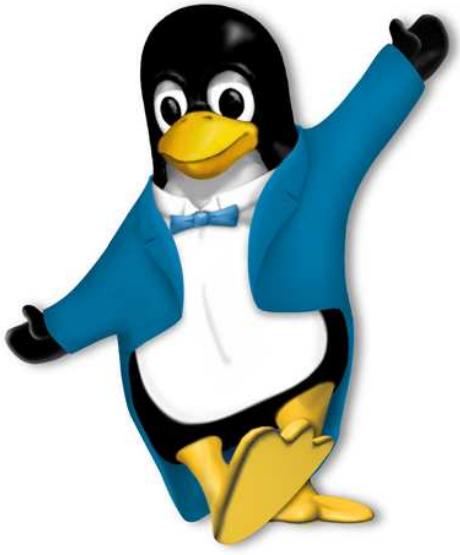
- Project page of the Linux Health Checker on SourceForge
  `http://lnxhc.sourceforge.net`
- Linux Health Checker User's Guide
  `http://www.ibm.com/developerworks/linux/linux390/documentation_dev.html`
- Linux on System z - Tuning Hints & Tips
  `http://www.ibm.com/developerworks/linux/linux390/perf/index.html`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

```
Check boot_runlevel_recommended (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_differs=medium (active)
  default_runlevel_differs=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".
```

cpu_capacity

```
Check cpu_capacity (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:
  capacity_reduced=high (active)

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".
```

cpu_capacity

```
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".
```

crypto_cca_stack

```
Check crypto_cca_stack (active)
===============================
Title:
  Verify the availability of System z cryptographic hardware support through a
  Common Cryptographic Architecture (CCA) stack

Description:
  Applications using cryptographic operations by linking to CCA libraries can
  only exploit System z cryptography hardware if the CCA stack is configured
  correctly.

  Cryptographic coprocessor adapters must be available. Prerequisites for a well
  configured CCA stack that uses System z cryptographic hardware is a device
  driver to exploit cryptographic adapters, and the csulcca library.

  The CCA stack is required for secure key cryptography operations.

  This health check verifies that:

  - The Cryptographic Coprocessor is available

  - Required RPMs, such as 'csulcca', are available

Exceptions:
  crypto_coprocessors_not_available=high (active)
  rpms_not_installed=high (active)
```

crypto_cpacf

```
Check crypto_cpacf (active)
===========================
Title:
  Confirm that CPACF is enabled

Description:
  The CP Assist for Cryptographic Functions (CPACF) accelerates symmetric
  cryptographic algorithms. This check verifies that CPACF is enabled on the
  system.

  CPACF is a mandatory prerequisite for hardware-based acceleration of
  cryptographic operations in the following contexts:

  - the OpenSSL software stack (see also checks crypto_openssl_stack and
    crypto_openssl_stack_32bit)

  - the clear key openCryptoki (PKCS#11) software stack (see also checks
    crypto_opencryptoki_ckc and crypto_opencryptoki_ckc_32bit)

  - Linux kernel-internal cryptographic operations, such as dm-crypt and IPSec

  CPACF is also required for the availability of the /dev/prng pseudo random
  number generator device.
```

crypto_cpacf

```
  Optionally CPACF enables protected key operation for the Common Cryptographic
  Architecture (CCA) software stack (see also checks crypto_cca_stack,
  crypto_opencryptoki_skc, and opencryptoki_skc_32bit).

Exceptions:
  cpacf_not_enabled=medium (active)
```

crypto_opencryptoki_ckc_32bit

```
Check crypto_opencryptoki_ckc_32bit (inactive)
==============================================
Title:
  Verify the availability of System z cryptographic hardware support for PKCS#11
  clear key cryptographic operations

Description:
  Software that uses clear key cryptographic functions via openCryptoki (PKCS#11
  API) can exploit System z cryptographic hardware if the openCryptoki clear key
  cryptographic stack is set up correctly. If the setup is incorrect or
  incomplete, Linux may in some cases emulate cryptographic operations by
  software. However this emulation will decrease system performance.

  The openCryptoki clear key cryptographic stack comprises openCryptoki together
  with the ICA token, the libica, possibly the System z cryptography kernel
  device driver and access to system cryptographic hardware features like CPACF
  and cryptographic adapters.
```

crypto_opencryptoki_ckc_32bit

```
  This health check verifies that:

  - The Cryptographic hardware (coprocessor and/or accelerator adapters) is
    available

  - Required RPMs, such as 'openCryptoki', and 'libica' are available

  - openCryptoki is initialized

  - The ICA token is configured

Exceptions:
  32bit_rpms_not_installed=high (active)
  crypto_adapters_not_available=medium (active)
  ica_token_not_configured=high (active)
  opencryptoki_not_initialized=high (active)
```

crypto_opencryptoki_ckc

```
Check crypto_opencryptoki_ckc (active)
======================================
Title:
  Verify the availability of System z cryptographic hardware support for PKCS#11
  clear key cryptographic operations

Description:
  Software that uses clear key cryptographic functions via openCryptoki (PKCS#11
  API) can exploit System z cryptographic hardware if the openCryptoki clear key
  cryptographic stack is set up correctly. If the setup is incorrect or
  incomplete, Linux may in some cases emulate cryptographic operations by
  software. However this emulation will decrease system performance.

  The openCryptoki clear key cryptographic stack comprises openCryptoki together
  with the ICA token, the libica, possibly the System z cryptography kernel
  device driver and access to system cryptographic hardware features like CPACF
  and cryptographic adapters.
```

crypto_opencryptoki_ckc

```
  This health check verifies that:

  - The Cryptographic hardware (coprocessor and/or accelerator adapters) is
    available

  - Required RPMs, such as 'openCryptoki', and 'libica' are available

  - openCryptoki is initialized

  - The ICA token is configured

Exceptions:
  crypto_adapters_not_available=medium (active)
  ica_token_not_configured=high (active)
  opencryptoki_not_initialized=high (active)
  rpms_not_installed=high (active)
```

crypto_opencryptoki_skc_32bit

```
Check crypto_opencryptoki_skc_32bit (inactive)
==============================================
Title:
  Verify the availability of System z cryptographic hardware support for PKCS#11
  secure key cryptographic operations

Description:
  Secure key cryptographic operations require a Cryptographic Coprocessor. In
  order to use secure key cryptography via the Public Key Cryptographic Standard
  11 (PKCS#11) API, openCryptoki together with the CCA token must be installed
  and configured correctly.

  This health check verifies that:

  - The Cryptographic Coprocessor is available

  - Required RPMs, such as 'openCryptoki', and 'csulcca' are available

  - openCryptoki is initialized

  - The CCA token is configured

Exceptions:
  32bit_rpms_not_installed=high (active)
  cca_token_not_configured=high (active)
  crypto_coprocessors_not_available=high (active)
  opencryptoki_not_initialized=high (active)
```

crypto_opencryptoki_skc

```
Check crypto_opencryptoki_skc (active)
======================================
Title:
  Verify the availability of System z cryptographic hardware support for PKCS#11
  secure key cryptographic operations

Description:
  Secure key cryptographic operations require a Cryptographic Coprocessor. In
  order to use secure key cryptography via the Public Key Cryptographic Standard
  11 (PKCS#11) API, openCryptoki together with the CCA token must be installed
  and configured correctly.

  This health check verifies that:

  - The Cryptographic Coprocessor is available

  - Required RPMs, such as 'openCryptoki', and 'csulcca' are available

  - openCryptoki is initialized

  - The CCA token is configured

Exceptions:
  cca_token_not_configured=high (active)
  crypto_coprocessors_not_available=high (active)
  opencryptoki_not_initialized=high (active)
  rpms_not_installed=high (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)
```

crypto_openssl_stack_32bit

```
Check crypto_openssl_stack_32bit (inactive)
===========================================
Title:
  Verify the availability of System z cryptographic hardware support through an
  OpenSSL stack

Description:
  The applications using cryptographic operations by linking to OpenSSL
  libraries can exploit System z cryptographic hardware only if the OpenSSL
  stack is configured correctly.

  The following cryptographic hardware can be exploited if available:

  - CPACF instructions in the CPU

  - Cryptographic Accelerator adapters

  - Cryptographic Coprocessor adapters

  Prerequisites for a well configured OpenSSL stack that uses System z
  cryptographic hardware are:

  - The enablement of CPACF, a device driver to exploit cryptographic adapters
    (if adapters are available)

  - The libica library

  - The openssl-ibmca engine for OpenSSL being installed and configured
```

crypto_openssl_stack_32bit

```
  Configuring the OpenSSL stack to exploit System z cryptographic hardware
  accelerates applications using cryptographic functions and offloads CPU cycles
  to cryptographic adapters. The availability of cryptographic adapters is
  optional because libica provides a software fallback for the functions
  provided by the adapters.

  This health check verifies that:

  - The Cryptographic Coprocessor or Accelerator is available

  - Required RPMs, such as 'openSSL', 'openssl-ibmca', and 'libica' are
    available

  - OpenSSL is configured with the 'ibmca' engine

Exceptions:
  32bit_rpms_not_installed=high (active)
  crypto_adapters_not_available=medium (active)
  ibmca_not_configured=high (active)
```

crypto_openssl_stack

```
Check crypto_openssl_stack (active)
===================================
Title:
  Verify the availability of System z cryptographic hardware support through an
  OpenSSL stack

Description:
  The applications using cryptographic operations by linking to OpenSSL
  libraries can exploit System z cryptographic hardware only if the OpenSSL
  stack is configured correctly.

  The following cryptographic hardware can be exploited if available:

  - CPACF instructions in the CPU

  - Cryptographic Accelerator adapters

  - Cryptographic Coprocessor adapters

  Prerequisites for a well configured OpenSSL stack that uses System z
  cryptographic hardware are:

  - The enablement of CPACF, a device driver to exploit cryptographic adapters
    (if adapters are available)

  - The libica library

  - The openssl-ibmca engine for OpenSSL being installed and configured
```

crypto_openssl_stack

```
  Configuring the OpenSSL stack to exploit System z cryptographic hardware
  accelerates applications using cryptographic functions and offloads CPU cycles
  to cryptographic adapters. The availability of cryptographic adapters is
  optional because libica provides a software fallback for the functions
  provided by the adapters.

  This health check verifies that:

  - The Cryptographic Coprocessor or Accelerator is available

  - Required RPMs, such as 'openSSL', 'openssl-ibmca', and 'libica' are
    available

  - OpenSSL is configured with the 'ibmca' engine

Exceptions:
  crypto_adapters_not_available=medium (active)
  ibmca_not_configured=high (active)
  rpms_not_installed=high (active)
```

crypto_z_module_loaded

```
Check crypto_z_module_loaded (active)
=====================================
Title:
  Confirm that the System z cryptography kernel module is loaded

Description:
  Loading the System z cryptography kernel module (named 'z90crypt or
  'zcrypt_pcixcc' or 'zcrypt_cex2a') is required to exploit cryptographic
  adapters. This check verifies that the kernel module is loaded.

  The System z cryptography kernel module is a mandatory prerequisite for
  cryptographic operations in the following contexts:

  - the secure key openCryptoki (PKCS#11) software stack (see also checks
    crypto_opencryptoki_skc and crypto_opencryptoki_skc_32bit)

  - the Common Cryptographic Architecture (CCA) software stack (see also check
    crypto_cca_stack)
```

crypto_z_module_loaded

```
  In addition the module is a prerequisite for accelerating and off-loading RSA
  operations in the following contexts:

  - the OpenSSL software stack (see also checks crypto_openssl_stack and
    crypto_openssl_stack_32bit)

  - the clear key openCryptoki (PKCS#11) software stack (see also checks
    crypto_opencryptoki_ckc and crypto_opencryptoki_ckc_32bit)

  In these contexts RSA operations will be computed in software if the
  cryptographic kernel module is not available.

  Finally, loading the kernel module is also required to implement true random
  number generation based on cryptographic adapter hardware.

Exceptions:
  module_not_loaded=medium (active)
```

css_ccw_blacklist

```
Check css_ccw_blacklist (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_chpid_status

```
Check css_ccw_chpid_status (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_device_availability

```
Check css_ccw_device_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_device_usage

```
Check css_ccw_device_usage (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_device_usage

```
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".
```

css_ccw_driver_association

```
Check css_ccw_driver_association (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)
```

fs_boot_zipl_bootmap

```
Check fs_boot_zipl_bootmap (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.
```

fs_boot_zipl_bootmap

```
  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)
```

fs_fstab_dasd_devnodes

```
Check fs_fstab_dasd_devnodes (active)
=====================================
Title:
  Identify standard DASD device nodes in the fstab file

Description:
  The DASD device driver creates standard device nodes for DASDs that are based
  on the order in which DASDs are set online. When you add or remove disks, the
  device name of a disk might change across a reboot. To preserve the mapping
  between standard device nodes and the associated physical disk space across
  reboot, use device nodes that are based on unique properties of a DASD and so
  identify a particular device.

Exceptions:
  volatile_devnodes_used=medium (active)
```

fs_fstab_fsck_order

```
Check fs_fstab_fsck_order (active)
==================================
Title:
  Check if filesystems are skipped by filesystem check (fsck)

Description:
  This check examines if the filesystems are skipped by filesystem check (fsck)
  while boot. If filesystems are not checked for consistency it might lead to
  filesystem corruption or drive might not even boot the system.

Exceptions:
  filesystem_not_checked=medium (active)
  root_low_prio_check=medium (active)
  root_not_checked=high (active)
```

fs_fstab_fsck_order

```
Parameters:
  exclude=none
        A list of filesystems, separated by colons (:). The filesystems mounted
        at the specified mount points are to be excluded from the consistency
        check. Special filesystems like /proc, /sys etc need not be checked for
        consistency.

        Example:

        /proc:/sys

        Default value is "none".

  mount=
        A list of filesystems, separated by colons (:). The filesystems mounted
        at the specified mount points are to be checked for consistency. If the
        list is empty, all mount points of /etc/fstab except in exclude list are
        checked.

        Example:

        /:/home

        Default value is "".
```

fs_inode_usage

```
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

```
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".
```

fs_mount_option_ro

```
Check fs_mount_option_ro (active)
=================================
Title:
  Check for read-only filesystems

Description:
  This check examines if filesystems have been mounted as read-only. If it has
  been mounted as read-only it would inhibit any kind of filesystem operations
  like editing, deleting files/folders etc.

Exceptions:
  read_only_filesystem=high (active)

Parameters:
  mount_points=/home:/proc:/tmp:/var/log:/sys
        A list of mount points, separated by colons (:). The filesystems mounted
        at the specified mount points are to be checked for read-only. If the
        list is empty, all mounted filesystems are checked.

        Example:

        /home:/proc

        Default value is "/home:/proc:/tmp:/var/log:/sys".
```

fs_tmp_cleanup

```
Check fs_tmp_cleanup (active)
=============================
Title:
  Verify that temporary files are deleted at regular intervals.

Description:
  Linux instances have one or more directories, for example, /tmp, to store
  temporary files. If temporary files are not deleted at regular intervals, they
  can fill up file systems and cause the Linux instance to run out of disk
  space.An exception message is issued unless a program is configured to clear
  directories with temporary files at regular intervals.

Exceptions:
  max_days_not_set=low (active)
  no_cron_job=low (active)
  temp_dir_miss=low (active)
  tmp_watch=low (active)

Parameters:
  temp_dir=/tmp
        A blank-separated list of directories that contain temporary files. The
        check verifies that temporary files in the specified directories are
        deleted at regular intervals.

        Default value is "/tmp".
```

fs_usage

```
Check fs_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_usage

```
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".

  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".
```

fw_callhome

```
Check fw_callhome (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:
  inactive=low (active)
  not_available=low (active)
```

fw_cpi

```
Check fw_cpi (active)
=====================
Title:
  Check if control program identification can display meaningful Linux instance
  names

Description:
  You can use the control program identification (CPI) to assign names to your
  Linux instances. The names are used to identify Linux instances, for example,
  on the Hardware Management Console (HMC) or the service element (SE).

  To assign meaningful names to your Linux instances, the CPI needs names for
  your Linux system and sysplex.

Exceptions:
  no_sysplex_name=low (inactive)
  no_system_name=medium (active)
```

log_syslog_rotate

```
Check log_syslog_rotate (active)
================================
Title:
  Verify that syslog files are rotated

Description:
  Syslog files contain the messages that are generated by the system components.
  If the size of the syslog files is not controlled, they might completely fill
  up your file system, and cause the Linux instance to run out of disk space.

  logrotate is a tool that monitors, rotates, compresses, or truncates syslog
  files to save disk space and to limit the syslog file size. This health check
  verifies that logrotate runs at regular intervals on your system and checks
  your logrotate configuration settings.

Exceptions:
  log_size_exceeded=medium (active)
  no_cron=high (active)
  no_logrotate=high (active)

Parameters:
  max_log_size=1MB
        Maximum syslog file size, specified in KB, MB, or GB.

        Default value is "1MB".
```

mem_swap_availability

```
Check mem_swap_availability (active)
====================================
Title:
  Check if swap space is available

Description:
  This check examines if swap space is available. For systems having memory
  constraints if swap space is not available it might lead to out-of-memory
  situations as also a system crash. Swapping allows the system to use more
  memory than is physically available.

Exceptions:
  no_swap_space=high (active)
```

mem_usage

```
Check mem_usage (active)
========================
Title:
  Ensure memory usage is within the threshold

Description:
  The check examines the RAM usage of the system. If not enough RAM is available
  system will slow down and become much more unresponsive and difficult or even
  impossible to use.

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

Parameters:
  critical_limit=90
        Memory usage (in percent) at which to raise a critical exception. Valid
        values are integers in the range of 1 to 100

        Default value is "90".

  warn_limit=80
        Memory usage (in percent) at which to raise a warning exception. Valid
        values are integers in the range of 1 to 100

        Default value is "80".
```

net_bond_ineffective

```
Check net_bond_ineffective (active)
===================================
Title:
  Identify bonding interfaces that are configured with single network interfaces

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

Exceptions:
  single_slave=medium (active)
```

net_bond_qeth_ineffective

```
Check net_bond_qeth_ineffective (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:
  single_chpid=medium (active)
```

net_dns_settings

```
Check net_dns_settings (active)
===============================
Title:
  Ensure nameserver is listed with correct address

Description:
  Nameserver helps to resolve names to numerical ip addresses. If nameserver is
  not listed or has an incorrect address it would prevent accessing any of the
  systems in the network by their name.

Exceptions:
  incorrect_nameserver=medium (active)
  no_nameserver=medium (active)
```

net_hsi_outbound_errors

```
Check net_hsi_outbound_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_errors

```
Check net_inbound_errors (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:
  limits_exceeded=medium (active)
```

net_inbound_errors

```
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

```
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)

Parameters:
  recommended_buffercount=<=500MB:16,<=900MB:32,<=1900MB:64,>1900MB: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.
```

net_qeth_buffercount

```
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). Note that this number is compared against the
  amount of available memory which may be lower than the total memory
  assigned to a Linux system due to kernel internal overhead.

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

net_qeth_buffercount

```
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,<=900MB:32,<=1900MB:64,>1900MB:128".
```

net_services_insecure

```
Check net_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".
```

proc_cpu_usage

```
Check proc_cpu_usage (active)
=============================
Title:
  Ensure processes do not hog cpu time

Description:
  This check ensures that processes do not end up hogging cpu time. If certain
  processes start hogging cpu time then other processes would be deprived of cpu
  time which might cause applications to slow down and the system might even
  become unresponsive.

Exceptions:
  process_hogs_cpu=high (active)

Parameters:
  cpu_time=300
        Per process accumulated cpu time in seconds which must be exceeded
        before an exception is reported.

        Valid values are integers starting with 1.

        Default value is "300".
```

proc_cpu_usage

```
cpu_usage=80
      Per process cpu usage at which to raise a high-severity exception. The
      cpu usage represents the percentage of time that a process spent running
      during its lifetime.

      Valid values are integers in the range 1 to 100.

      Default value is "80".

processes=
      A list of processes separated by comma (,) that are expected to consume
      high cpu time and which need not be reported by this check. If the list
      is empty, all the processes consuming high cpu time are reported.

      Example:

      firefox, apache2

      Default value is "".
```

proc_load_avg

```
Check proc_load_avg (active)
============================
Title:
  Ensure the system is running with optimal load

Description:
  This check examines the CPU and IO load of the system. If a system is running
  with very high loads, then system could turn unresponsive and applications can
  take more time to react/complete.

Exceptions:
  high_load=medium (active)
  over_load=high (active)
```

proc_load_avg

```
Parameters:
  avgload=90
        System usage (in percent) at which to raise a exception. Valid values
        are integers in the range 1 to 100.

        Default value is "90".

  time=15
        Time for which load average needs to be checked. Valid values are 1, 5,
        15. You can specify more than one timestamp separated by comma (,).

        Example:

        1,15

        Default value is "15".
```

proc_mem_oom_triggered

```
Check proc_mem_oom_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)
```

proc_mem_usage

```
Check proc_mem_usage (active)
=============================
Title:
  Ensure processes do not hog memory

Description:
  This check ensures that processes do not end up hogging memory. If certain
  processes start hogging memory other processes would be deprived of memory and
  applications might slow down and system might even become unresponsive.

Exceptions:
  process_hogs_memory=high (active)
```

proc_mem_usage

```
Parameters:
  mem_usage=90
        Per process memory usage at which to raise a high-severity exception.
        Valid values are integers in the range 1 to 100.

        Default value is "90".

  processes=lnxhc
        A list of processes separated by comma (,) that are expected to consume
        high memory and which need not be reported by this check. If the list is
        empty, all the processes consuming high memory are reported.

        Example:

        firefox, apache2

        Default value is "lnxhc".
```

proc_priv_dump

```
Check proc_priv_dump (active)
=============================
Title:
  Ensure that privilege dump is switched off

Description:
  With the privilege dump set to non-zero value (1 or 2), a core-dump of
  set-user-id or otherwise protected/tainted binaries is created. For security
  reasons it is important to disable it by default.

Exceptions:
  debug_mode=high (active)
  suidsafe_mode=medium (active)
```

ras_dump_kdump_on_panic

```
Check ras_dump_kdump_on_panic (active)
======================================
Title:
  Ensure kdump is configured and running

Description:
  This check examines if kdump is configured and running on the system. If kdump
  is not running and a system crash occurs, crash dump will not be captured and
  system will not be available for use. Kdump allows the system to come back
  after crash along with crash dump which can be used for post-morterm analysis.

Exceptions:
  no_kdump=high (active)
  no_kdump_crash=high (active)
```

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)
```

ras_panic_on_oops

```
Check ras_panic_on_oops (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)
```

sec_tty_root_login

```
Check sec_tty_root_login (active)
=================================
Title:
  Confirm that root logins are enabled for but restricted to secure terminals

Description:
  The login program and the Linux Pluggable Authentication Modules (PAM)
  configuration restrict root logins to the terminals listed in /etc/securetty.

  This check verifies that root logins are enabled for all terminals that are
  considered secure. This check also verifies that no root logins are permitted
  on terminals that are considered insecure.

  Root logins on multiple terminals might be helpful in emergency situations.
  However, root logins on insecure terminals constitute a security exposure.

Exceptions:
  insecure_enabled=medium (active)
  secure_disabled=medium (active)
```

sec_tty_root_login

```
Parameters:
  insecure_ttys=
        A blank-separated list of terminals that are considered insecure, and
        for which root logins must not be permitted. When specifying terminals,
        omit the leading /dev/.

        An exception message is issued if any terminal here is also listed in
        /etc/securetty.

        Default value is "".

  secure_ttys=ttyS0 ttyS1 ttysclp0 sclp_line0 hvc0 hvc1 hvc2 hvc3 hvc4 hvc5 hvc6 hvc7
        A blank-separated list of terminals that are considered secure, and for
        which root logins should be permitted. When specifying terminals, omit
        the leading /dev/.

        An exception message is issued if any terminal listed here is missing in
        /etc/securetty.

        Default value is "ttyS0 ttyS1 ttysclp0 sclp_line0 hvc0 hvc1 hvc2 hvc3
        hvc4 hvc5 hvc6 hvc7".
```

sec_users_uid_zero

```
Check sec_users_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".
```

storage_dasd_cdl_part

```
Check storage_dasd_cdl_part (active)
====================================
Title:
  Identify CDL-formatted DASD where the metadata area is used for storing data

Description:
  Compatible Disk Layout (CDL) formatted DASD should have a partition and the
  partition should not start before track 2. Otherwise data corruptions might
  occur. Also the metadata which is stored in tracks 0 and 1 can be corrupted.
  Metadata contains partition tables and volume labels that are required by
  other operating systems, for example, z/OS. If metadata is corrupted, other
  operating systems might no longer recognize the disk.

  On CDL formatted devices, the first blocks are formatted with a non-standard
  block size. And for what ever the data written, while reading it gives back
  only '0xE5'. The first two tracks of CDL DASDs contain meta-data such as
  volume labels and partition tables. The volume labels are required so that the
  disk can be recognized by other operating systems (e.g. z/OS). If these are
  overwritten, the disk contents will no longer be recognized by these operating
  systems.

Exceptions:
  invalid_partition_start=high (active)
  no_partition_found=medium (active)
```

storage_dasd_eckd_blksize

```
Check storage_dasd_eckd_blksize (active)
========================================
Title:
  Confirm 4K block size on ECKD DASD devices

Description:
  Verify the block size of low-level formatted ECKD DASD devices. If the block
  size is other than 4096 an exception is reported. A block size of 4096 maps to
  the default block size of file systems and typically have a good I/O
  throughput.

Exceptions:
  unexpected_eckd_block_size=medium (active)
```

storage_dasd_nopav_zvm

```
Check storage_dasd_nopav_zvm (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:
  ineffective_nopav=low (active)
```

storage_dasd_pav_aliases

```
Check storage_dasd_pav_aliases (active)
=======================================
Title:
  Identify active DASD alias devices without active base device

Description:
  Alias devices without active base device affect the system performance and
  indicate a configuration problem. Through the Parallel Access Volume (PAV)
  feature, storage systems can represent the same physical disk space as a base
  device and one or more alias devices. With IBM HyperPAV, an alias can be used
  for any base device within the same logical subsystem on the storage system.

Exceptions:
  orphaned_alias=medium (active)
```

storage_mp_ineffective

```
Check storage_mp_ineffective (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)
```

storage_mp_path_state

```
Check storage_mp_path_state (active)
====================================
Title:
  Identify multipath devices with too few available paths or too many failed
  paths

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 an insufficient number of available
  paths or an excessive number of failed paths might not meet these goals.

Exceptions:
  too_few_available_paths=high (active)
  too_many_failed_paths=medium (active)

Parameters:
  failed_path_limit=1
        Maximum number of failed hardware paths to be tolerated for a multipath
        device.

        Default value is "1".

  remaining_path_limit=2
        Minimum number of available hardware paths to be required for a
        multipath device.

        Default value is "2".
```

tty_console_getty

```
Check 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:
  getty_on_console=medium (active)
```

tty_console_log_level

```
Check tty_console_log_level (active)
====================================
Title:
  Check for current console_loglevel

Description:
  This check examines if appropriate console_loglevel is set. console_loglevel
  determines the severity of messages that needs to go to the console. If
  appropriate console_loglevel is not set then the user might miss some
  important messages.

Exceptions:
  low_loglevel=medium (active)
```

tty_console_log_level

```
Parameters:
  log_level=4
        Log level below which to raise an exception. Valid values are integers
        in the range 1 to 8

        List of log levels are:

        KERN_EMERG      0     /*system is unusable                */
        KERN_ALERT      1     /*action must be taken immediately     */
        KERN_CRIT       2     /*critical conditions                */
        KERN_ERR        3     /*error conditions                */
        KERN_WARNING    4     /*warning conditions                */
        KERN_NOTICE     5     /*normal but significant condition     */
        KERN_INFO       6     /*informational                   */
        KERN_DEBUG      7     /*debug-level messages                */

        Default value is "4".
```

tty_devnodes

```
Check tty_devnodes (active)
===========================
Title:
  Detect terminals with multiple device nodes

Description:
  This check detects terminals that are represented by more than one device
  node. If getty programs are configured for two device nodes that both map to
  the same terminal, the getty programs interfere with each other, so that users
  cannot log in.

Exceptions:
  tty_has_multiple_nodes=medium (active)
```

tty_hvc_iucv

```
Check tty_hvc_iucv (active)
===========================
Title:
  Confirm that all available z/VM IUCV HVC terminals are enabled for logins

Description:
  The z/VM IUCV Hypervisor Console (HVC) device driver can manage up to eight
  HVC terminals that can be enabled for user logins. The number of HVC terminals
  is specified through a kernel parameter.

  HVC terminals that are not enabled for logins serve no purpose and cannot
  provide access to the Linux instance in emergencies.

  This check confirms that all available HVC terminals are enabled for user
  logins.

Exceptions:
  too_few_ttys=low (active)
  unused_ttys=medium (active)

Parameters:
  min_hvc_iucv=1
        Specifies the minimum number of HVC terminal devices that must be
        available. This in an integer number in the range from 1 to 8.

        Default value is "1".
```

tty_idle_terminals

```
Check tty_idle_terminals (active)
=================================
Title:
  Identify idle terminals

Description:
  Identify terminals on which users are logged in but are not active. Each
  logged-in user occupies a terminal that could be used by another user.

Exceptions:
  idle_ttys=low (active)
```

tty_idle_terminals

```
Parameters:
  idle_time=1d
        Specifies the maximum idle time to be tolerated. Valid values are
        positive integers followed by d, h, m, or s for days, hours, minutes, or
        seconds.

        If a user exceeds this idle time, an exception message is issued.

        Default value is "1d".

  tty=
        A blank-separated list of terminals. The check identifies idle users who
        are logged in through the specified terminals. If the list is empty, all
        terminals are checked.

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

        Default value is "".
```

tty_idle_users

```
Check tty_idle_users (active)
=============================
Title:
  Identify idle users

Description:
  Identify users who are logged in but are not active. Each logged-in user
  occupies a terminal that could otherwise be used by another user.

Exceptions:
  idle_users=low (active)
```

tty_idle_users

```
Parameters:
  idle_time=1d
        Specifies the maximum idle time to be tolerated. Valid values are
        positive integers followed by d, h, m, or s for days, hours, minutes, or
        seconds.

        If a user exceeds this idle time, an exception message is issued.

        Default value is "1d".

  tty=
        A blank-separated list of terminals for which idle users are identified.
        Terminals are specified by their device node without the leading /dev/.
        If the list is empty, all terminals are checked.

        Default value is "".

  users=root
        A blank-separated list of user IDs for which the idle times are checked.
        If the list is empty, all users are checked.

        Default value is "root".
```

tty_usage

```
Check 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".
```

zvm_priv_class

```
Check zvm_priv_class (active)
=============================
Title:
  Check the privilege classes of the z/VM guest virtual machine on which the
  Linux instance runs

Description:
  This check examines the z/VM privilege classes of the current z/VM guest
  virtual machine and compares them with the permitted privilege classes. The
  permitted privilege classes are provided by the permitted_privclass parameter.

  Higher privilege classes than the permitted ones might allow operations which
  can inadvertently or maliciously affect the security and availability of other
  z/VM guest virtual machines running in the same z/VM instance. Generally,
  higher privilege classes should be assigned sparingly and only to trusted z/VM
  user IDs.

Exceptions:
  default_privileges_exceeded=medium (active)
  running_privileges_exceeded=medium (active)
```

zvm_priv_class

```
Parameters:
  check_for=Currently, Directory
        Privilege classes to check: privilege classes effective at run-time
        (currently), privilege classes permanently defined in the user directory
        (directory), or both (currently, directory).

        Default value is "Currently, Directory".

  permitted_privclass=G
        Privilege classes permitted for z/VM guest virtual machines. Valid
        values are lists of letters in the range A to Z and integers in the
        range 1 to 6.

        Example:

        ABCD12

        Default value is "G".
```