PUBLIC

Document Version: 2.0 – 2017-04-27

SAP HANA Hardware Configuration Check Tool 1.0

SAP HANA Platform

Typographic Conventions

Type Style	Description
Example	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Textual cross-references to other documents.
Example	Emphasized words or expressions.
EXAMPLE	Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.
`Example`	Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
`Example`	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
`<Example>`	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
EXAMPLE	Keys on the keyboard, for example, F2 or ENTER.

Contents

1 About the SAP HANA Hardware Configuration Check Tool

The SAP HANA Hardware Configuration Check Tool (HWCCT) is a framework that provides tests and reports for new single node appliances and scale-out systems to determine if the hardware you intend to use meets the minimum performance criteria required to run SAP HANA in production use.

This guide describes these tests and reports and contains examples of test configurations that you can use. For the latest version of this document, see SAP Note 1943937.

> ### Caution
>
> The test should only be used before going into production. It should only be used on production machines when this has first been requested by SAP Support.

1.1 General Prerequisites

Before using the tool, ensure that the following prerequisites are met:

- When using the SAP HANA Hardware Configuration Check Tool to evaluate a distributed landscape, the binaries should be available from a shared directory so every server can execute it.
- Ensure that you have already exchanged SSH keys between the different hosts so that inter-host communication can take place without passwords.
- Ensure that all servers can resolve hostnames with full qualified domain names.
- SAPCAR must be installed to extract the binaries.

1.2 Downloading the Tool

You can download the latest version of the tool on the SAP Support Portal:

1. Open https://support.sap.com in a browser.
2. Choose *Download Software*.
3. On the *Support Packages & Patches* tab, choose one of the following:
    - For SAP HANA Platform Edition 1.0, choose *By Alphabetical Index (A-Z)* -> *H* -> *SAP HANA PLATFORM EDITION* -> *Downloads* tab -> *SAP HANA PLATFORM EDIT. 1.0* -> *Downloads* tab -> *HANA CONFIG CHECK*-> *Downloads* -> *SAP HANA HW CONFIG CHECK 1.0*.

Choose *Linux on Power BE 64bit*, *Linux on Power LE 64bit*, or *Linux on x86_64 64bit* depending on your requirements.

- For SAP HANA Platform Edition 2.0, choose *By Alphabetical Index (A-Z)* -> *H* -> *SAP HANA PLATFORM EDITION* -> *Downloads* tab -> *SAP HANA PLATFORM EDITION 2.0 -> Downloads* tab -> *SAP HANA HW CONFIG CHECK 1.0*-> *Downloads* ->*SAP HANA HW CONFIG CHECK 1.0*.
  Choose *Linux on Power BE 64bit*, *Linux on Power LE 64bit*, or *Linux on x86_64 64bit* depending on your requirements.

1.3 Installing the Tool

We recommend that you put the binaries in a shared location, for example in parallel to your main SAP HANA installation directory, for example, "`/hana/shared/`" This helps to avoid potential problems when sharing the binaries for distributed tests.

1. Copy the SAR file "HWCCT.SAR" to the Linux system hosting your SAP HANA database.
2. Install the tool by executing this command:
   ```
   SAPCAR -xf HWCCT.SAR hwcct
   ```

A new directory `/hana/shared/hwcct` is created.

2 Test Modules

This section includes a description of the available modules and what they do. The current version of the tool has the following modules:

- Landscape Test
- File System Test
- Network Test

2.1 Landscape Test Module

2.1.1 About the Landscape Test Module

The landscape test module collects detailed system information and logs from the SAP HANA system landscape. This information is used to check that the OS configuration is valid and it checks the consistency of the landscape.

The module also helps to reduce support resolution time. Information gathering modules can disclose sensitive data. If this is an issue, you will need to prune the sensitive parts before sending the test results to SAP.

The system configuration will be checked based on the SAP HANA reference architecture and the operating system governance and configuration notes.

New findings will be included in tool updates.

2.1.2 Configuring the Landscape Test Module

The landscape test is the most basic test in the framework and measures currently relevant aspects of the operating system. The following test is included in this module: `EvalOs Test`.

2.1.2.1 EvalOS Test

The test class EvalOS belongs to the landscape test package.

```
{
            "package": "LandscapeTest",
            "test_timeout": 0,
            "id": 1,
            "config": { },
```

```
        "class": "EvalOs"
    }
```

2.2 File System Test Module

Caution

The test should only be used before going into production. It should only be used on production machines when this has first been requested by SAP Support.

2.2.1 About the File System Test Module

The purpose of this test module is to measure data throughput and latency of the IO subsystem. The process of measurement is related to the ioengine of the SAP HANA database, to ensure that the projected results meet the requirements of SAP HANA. The ioengine of SAP HANA uses the AIO interface, some operations are not well-represented in some file systems, like for example, `ext4`. This means that certain types of file-enlarging writes are completely unsupported and block any other activity for the entire duration of the operation.

To test different possible behaviors of the IO subsystem, the general procedure used for testing is as follows: create a file and enlarge it to a certain size, overwrite the entire file and finally read the file. All these operations are performed with blocks of different sizes:

-- Data volume: 4KB, 16KB, 64KB, 1MB, 16MB, 64MB

-- Log volume: 4KB, 16KB, 1MB

On the storage device, those blocks are written to a data file with a size between 1GB and 32GB.

2.2.2 Configuring the File System Test Module

The file size used for each test iteration can be determined with the test module parameter “duration”.

The need to test larger file sizes depends on the cache used in the IO subsystem.

You can use the configuration parameter ”duration” to control the data file size as shown in the table below. For an example, see the sample “test_config.json” file on lines 41 and 51 of the code listing:

Parameter	Description
`"duration":"short"`	Data file size will be 5GB max

Parameter	Description
`"duration":"long"`	Data file size will be 16GB max
`"duration":"verylong"`	Data file size will be 32GB max

We recommend that you first start testing with this parameter set to `"duration":"short"`.
Repeat the test with `"duration":"long"` and `"duration":"verylong"` to exclude possible caching effects.

You can configure this test to examine your data.

```
      {
            "package": "FilesystemTest",
            "test_timeout": 0,
            "id": 2,
            "config": {"mount":{"<hostname>":["<mountpoint_of_data_volume>"]
                              },
"parameter":{"parameter1 ":"value1",
"parameter2":"value2"},

                        "duration":"short"
                    },
            "class": "DataVolumeIO"
       },
      {
            "package": "FilesystemTest",
            "test_timeout": 0,
            "id": 3,
            "config": {"mount":{"<hostname>":["<mountpoint_of_log_volume>"]
                              },
"parameter":{"parameter1 ":"value1",
       "parameter2":"value2"},
                        "duration":"short"
                    },
            "class": "LogVolumeIO"
       }
```

2.2.3 KPIs for Data Throughput and Latency for Production SAP HANA Systems

The KPIs in the table below are the minimum values required for successful certification of production SAP HANA systems.

Volume	Block Sizes	Test File Size*	KPIs			
			Initial Write (MB/s)	Overwrite (MB/s)	Read (MB/s)	Latency (µs)
Log	4KB	5GB	n.a.	30	n.a.	1000
	16KB	16GB	n.a.	120	n.a.	1000
	1MB	16GB	n.a.	250	250	n.a.
Data	4KB	5GB	n.a.	n.a.	n.a.	n.a.
	16KB	16GB	40	100	n.a.	n.a.
	64KB	16GB	100	150	250	n.a.
	1MB	16GB	150	200	300	n.a.
	16MB	16GB	200	250	400	n.a.
	64MB	16GB	200	250	400	n.a.

*) Refers to size of the file created by the test if configured with `"duration":"long"`. For successful certification of your hardware, SAP asks you to fill in results of a test run with `"duration":"long"`.

2.2.4 File System Test Fsperf

The SAP HANA Hardware Configuration Check Tool contains the tool `Fsperf`, which shows the latest changes in the FileIO layer.

The test driver is available in the `/lib` directory of the HWCCT folder.

Analogous to the `FilesystemTest`, the purpose of `Fsperf` is to measure throughput and latency for SAP HANA FileIO operations and to ensure a non-blocking IO configuration. Therefore, all of the block sizes mentioned in the KPI table above have to be tested for each of the respective volumes.

To ensure that IOs are non-blocking for SAP HANA, the "`Ratio trigger time to I/O time`" must be smaller than "`0.1`". This ratio is shown for every IO operation: file extends, overwrites and reads.

If the default SAP HANA configuration leads to a higher ratio, you can change the basic parameters to bring the ratio below the threshold.

Basic Parameters

Parameter Name	Description	Values
async_read_submit	Controls whether read requests are submitted asynchronously.	- off (Default) - on
async_write_submit_active	Controls whether write requests can be submitted asynchronously.	- off - auto (decide based on underlying file system, Default) - on
async_write_submit_blocks	Controls which blocks are written asynchronously. Only relevant if async_write_submit_active is 'on' or 'auto' and file system is flagged as requiring asynchronous write submits.	- none - new (Default) - all

We recommend to be relatively conservative regarding changes of asynchronously submitted requests from the client to the SAP HANA fileIO layer, if this is not required to achieve a “ratio” below 0.1.

For throughput and latency tuning, we provide optional parameters to modify queue depth and IO batch sizes. Some IO subsystems, or file systems may gain or lose performance from changes to the provided optional parameters. We recommend that an IO educated performance engineer tunes the system on the operating system and SAP HANA level.

Optional Parameters

Parameter Name	Description	Values
size_kernel_io_queue	Size of the kernel I/O queue per completion queue.	Value must be in [128, 2048]. Default is 512.
max_parallel_io_requests	Maximum number of parallel I/O requests per completion queue.	Must be in [1, size_kernel_io_queue], Default is 64.
min_submit_batch_size:	Minimum batch size for asynchronous submits.	Must be in [1, max_submit_batch_size[, Default is 16.
max_submit_batch_size:	Maximum batch size for asynchronous submits.	Must be in]min_submit_batch_size, max_parallel_io_requests], Default is 64.

Examples of Fsperf usage:

Displaying Help: ./fsperf --help

Run all measurements in a sequential IO pattern with 1MB blocks and 16GB file size:
./fsperf -m all -b 1M -f 16G -i sequential /hana/log/

Run all measurements in a sequential IO pattern with 1MB blocks and 16GB file size and two additional basic options:

./fsperf -m all -b 1M -f 16G -i random
--param async_write_submit_active=off --param async_read_submit=on /hana/data/

For an example output, see the appendix.

Fsperf integration in FilesystemTest:

Fsperf is now integrated into the FilesystemTest utility of HWCCT. The configuration as shown in the previous chapter has been extended be able to use the fsperf parameters.

You can use the configuration parameter: "parameter" to add the fsperf parameters from the basic and optional section to the test. For an example, see the following configuration:

```
      {
           "package": "FilesystemTest",
           "test_timeout": 0,
           "id": 2,
           "config": {"mount":{"<hostname>":["<mountpoint_of_data_volume>"]
                                 },
"parameter":{"parameter1 ":"value1",
       "parameter2":"value2"},

                        "duration":"short"
                      },
           "class": "DataVolumeIO"
        },
      {
           "package": "FilesystemTest",
           "test_timeout": 0,
           "id": 3,
           "config": {"mount":{"<hostname>":["<mountpoint_of_log_volume>"]
                                 },
```

```
"parameter":{"parameter1 ":"value1",
       "parameter2":"value2"},
                      "duration":"short"
                    },
           "class": "LogVolumeIO"
        }
```

2.3 Network Test Module

> **Caution**
>
> The test should only be used before going into production. It should only be used on production machines when this has first been requested by SAP Support.

The network test module examines the network behavior with the same network properties and communication stack used in SAP HANA.

The test driver is available in the "`lib`" directory of the HWCCT folder.

SAP HANA Hardware Configuration Check Tool consists of a server and client component, very similar to Iperf. To perform a point-to-point network test, it is necessary to start the server component on one of the servers and connect it to the client component on another server in the SAP HANA cluster. To verify network performance, it is necessary to measure all network combinations between the servers in the SAP HANA cluster.

The KPI for a bonded 10Gbit Device is 9.0 Gbit/s for maximum throughput, with a minimum payload size of 10MB on 10 iterations.

Examples of TDINet usage:

Displaying Help: **`./TDINetClient -h`**

The server component can be started without any parameters:

`./TDINetServer`

The program will show on which interfaces it starts accepting requests. Example output is shown in the appendix.

To start measurement, it is necessary to specify the IP address and port to be able to connect to the right server, and the options for the test size and iterations.

> **Example**
>
> TDINetClient -m <host> <port> <size> <iterations> <verbose>
>
> ./TDINetClient -m 127.0.0.1 44644 10000000 10 n

For an example output, see the appendix.

3 Example: Test Configuration

Before implementing SAP HANA, it is necessary to test the hardware configuration. Your SAP contact will discuss with you which test is required to successfully validate the proposed configuration. The following section however, shows an example of a test configuration that can be used.

3.1 Example Configuration

You must create the file `"test_config.json"` in the directory `/hana/shared/hwcct/`

Enter the test configuration settings as follows:

1. In line 2 (`"report_id":"<report_id>"`) replace `<report_id>` with a meaningful name that will help you to identify the test reports later on.

   > **Note**
   >
   > All test reports are stored in the directory:
   > `/hana/shared/hwcct/report_<report_id>_<timestamp>`
   > The value entered for `<report_id>` must not contain any blanks, otherwise test execution will fail with an error message.

2. In line 4 (`"blades":["<externalhostname>"]`) replace `<hostname>` with the hostnames of your SAP HANA servers.

   > **Note**
   >
   > If you are not sure about the correct hostname, you can execute the command `"hostname"` on that server.

3. In line 39 (`"config": {"mount":{"<hostname>":["<mountpoint_of_data_volume>"]`) do the following:
   1. Replace `<hostname>` with the hostname of your SAP HANA server.
   2. Replace `<mountpoint_of_data_volume>` with the mountpoint of the data volume on your Enterprise Storage system.

      > **Note**
      >
      > Do not forget the trailing `"/"` character.

4. In line 49 (`"config": {"mount":{"<hostname>":["<mountpoint_of_log_volume>"]`) do the following:
   1. Replace `<hostname>` with the hostname of your SAP HANA server
   2. Replace `<mountpoint_of_log_volume>` with the mountpoint of the log volume on your Enterprise Storage system.

Note

Do not forget the trailing "/" character.

5. Save the "test_config.json" file.

Code Listing: test_config.json

```
1    {
2    "report_id":"<report_id>",
3    "use_hdb":false,
4    "blades":["externalhostname1","externalhostname2",
     "externalhostname3","externalhostname4","externalhostname5"],
5    "tests": [{
6                "package": "LandscapeTest",
7                "test_timeout": 0,
8                "id": 1,
9                "config": { },
10               "class": "EvalOs"
11           },
             {
12               "package": "NetworkTest",
13               "test_timeout": 0,
14               "id": 2,
15               "config":
     {"InternHosts":{"externalhostname1":"internodehostname1",
16                       "externalhostname2":"internodehostname2",
17                       "externalhostname3":"internodehostname3",
18                       "externalhostname4":"internodehostname4",
19                       "externalhostname5":"internodehostname5"}
20           },
21               "class": "NetworkSingleStream"
22           },
23           {
24               "package": "NetworkTest",
25               "test_timeout": 0,
26               "id": 3,
```

```
27              "config":
    {"InternHosts":{"externalhostname1":"internodehostname1",
28                  "externalhostname2":"internodehostname2",
29                  "externalhostname3":"internodehostname3",
30                  "externalhostname4":"internodehostname4",
31                  "externalhostname5":"internodehostname5"}
32                },
33               "class": "NetworkBidirectStream"
34          },
35         {
36              "package": "FilesystemTest",
37              "test_timeout": 0,
38              "id": 4,
39              "config":
    {"mount":{"<hostname>":["<mountpoint_of_data_volume>"]
40                                  },
41                           "duration":"short"
42                       },
43              "class": "DataVolumeIO"
44          },
45         {
46              "package": "FilesystemTest",
47              "test_timeout": 0,
48              "id": 5,
49              "config":
    {"mount":{"<hostname>":["<mountpoint_of_log_volume>"]
50                                  },
51                           "duration":"short"
52                       },
53              "class": "LogVolumeIO"
54          }
55  ]
56  }
```

3.2 Running the Test

Make sure you have configured `test_config.json` with the values as described in 3.1 Example Configuration.

In the directory `/hana/shared/hwcct`, execute the command:

```
source envprofile.sh
./hwval -f test_config.json
```

Result

The test results directory is located under:

`/hana/shared/hwcct/report_<report_id>_<timestamp>`

> **Note**
>
> The value entered for `<report_id>` must not contain any blanks; otherwise test execution will fail with an error message.

4 Appendix

4.1 fsperf example output

The tests will be executed with the following settings:

```
  Test mode:.....all
  Measurements.....all
  Program output:.....short
  Block size:.....1MB
  File size:.....16GB
  I/O access order:.....sequential
  Directory path:.....//hana/data/
  Additional program parameters:
    None


Results of Initial Write Test
==============================

Throughput Test:
----------------

  Trigger time:..... 0.000165 s   (Throughput: 6.20606e+06 MB/s)
  Asynchronous submit time:..... 0.031103 s   (Throughput:    32922.8 MB/s)
  Synchronous submit time:.....       0 s   (Throughput:          0 MB/s)
  I/O time:..... 1.93177 s   (Throughput:    530.083 MB/s)
  Ratio trigger time to I/O time:.8.54138e-05

Latency Test:
-------------

  I/O time:..... 1.96958 s   (Throughput:    519.907 MB/s)
  Latency:.....    1923 us


Results of Overwrite Test
```

==========

Throughput Test:

Trigger time:..... 0.005631 s (Throughput: 181850 MB/s)
Asynchronous submit time:..... 0 s (Throughput: 0 MB/s)
Synchronous submit time:..... 0.008885 s (Throughput: 115250 MB/s)
I/O time:..... 2.08097 s (Throughput: 492.077 MB/s)
Ratio trigger time to I/O time:.0.00270594

Latency Test:

I/O time:..... 2.70103 s (Throughput: 379.114 MB/s)
Latency:..... 2637 us

Results of Read Test

==========

Throughput Test:

Trigger time:..... 0.004618 s (Throughput: 221741 MB/s)
Asynchronous submit time:..... 0 s (Throughput: 0 MB/s)
Synchronous submit time:..... 0.003754 s (Throughput: 272775 MB/s)
I/O time:..... 1.00937 s (Throughput: 1014.48 MB/s)
Ratio trigger time to I/O time:.0.0045751

Latency Test:

I/O time:..... 1.97919 s (Throughput: 517.381 MB/s)
Latency:..... 1932 us

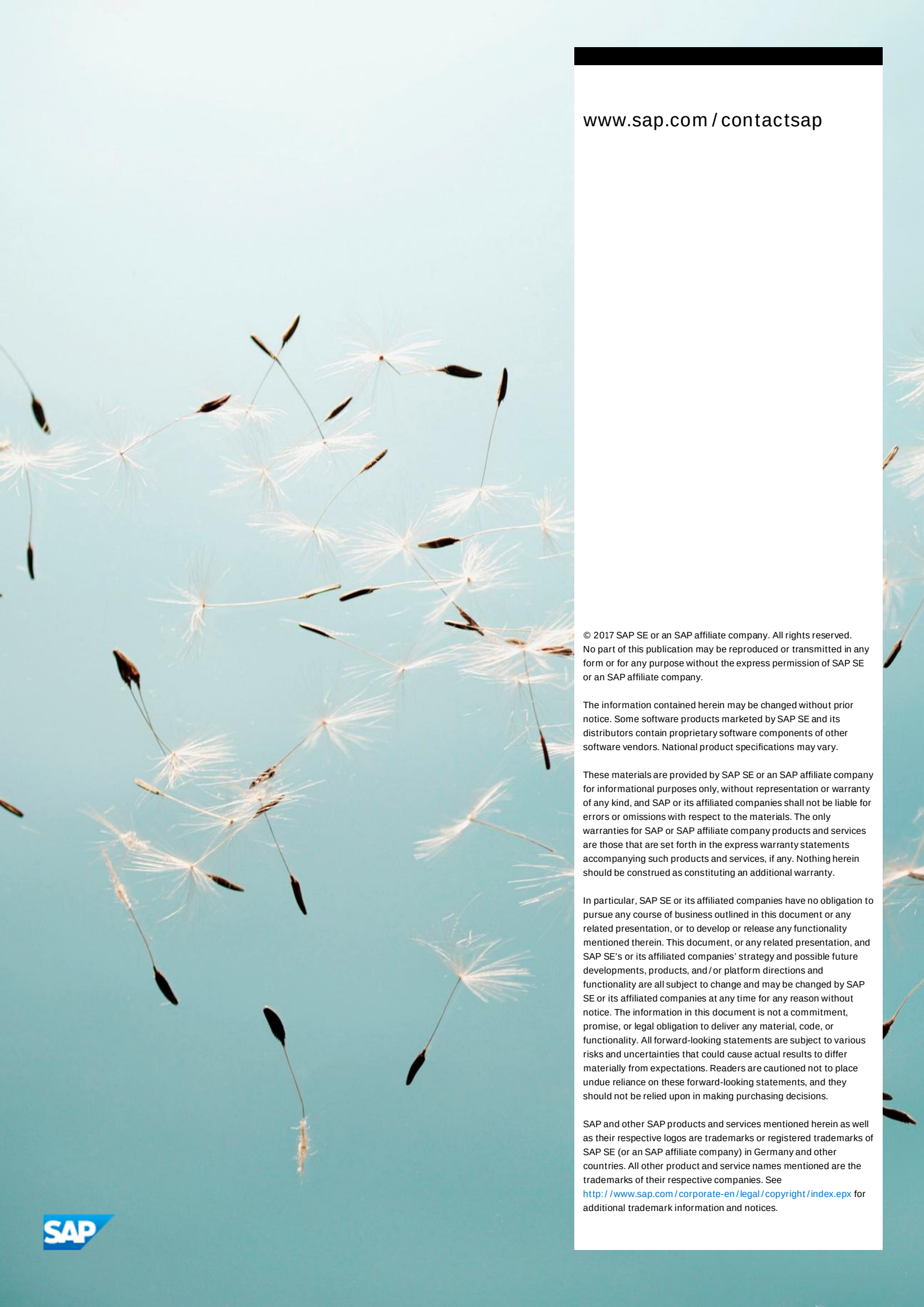
4.2 TDINet example output

./TDINetServer

TDI Server starting
accepting requests at 127.0.0.1:44644;

./TDINetClient -m 127.0.0.1 44644 10000000 10 n
sss
Checking TDI Server...
 TDI Server ok

Test Size 100000000
  Maximal Throughput (MBit/s): 9324
  Average Throughput (MBit/s): 9048



www.sap.com/contactsap