PowerCampus[01]

LPAR-Tool 1.4.2.x

What's New

Version: 1.4.2.0

https://www.powercampus.de

https://www.powercampus01.com

Foreword

Introduction

This handbook describes new features available in the LPAR tool starting with version 1.4.x. The manual presupposes the following:

- basic knowledge of working on the command line of a UNIX system
- basic understanding of virtualization concepts and features of POWER virtualization

The user manual can be downloaded from the download area on the PowerCampus01 website:

- *https://www.powercampus.de* or *https://www.powercampus01.com*

Additional Information

More information about the LPAR tool is available in the Tools section of the PowerCampus01 website:

- *https://www.powercampus.de* or *https://www.powercampus01.com*

Help with Problems

If the LPAR tool malfunctions, PowerCampus01 technical support can be contacted. The following URL will open a software call for the LPAR tool:

- https://www.powercampus.de/tools/lpar-tool/software-call

Support can be reached via the following e-mail address:

- E-mail: *support@powercampus.de*

Software updates of the LPAR tool can be downloaded from the download area on the PowerCampus01 website:

- *https://www.powercampus.de* or *https://www.powercampus01.com*

1. What's New in 1.4.2.x

1. Integrated Test License

Version 1.4.2.0 of the LPAR-Tool, available in the download section of our website, has an integrated test license, which is valid until 30th june 2020.

The integrated test license supports up to HMCs, up to 100 managed systems and a up to 1000 LPARs.

A test license can also be requested by simply writing an email to info@powercampus.de.

2. Displaying the Model Name

The "*ms show*" command has been expanded to include the *MODEL_NAME* output column. In this column the IBM model names of the displayed managed systems are displayed:

```
$ ms show
NAME  SERIAL_NUM  TYPE_MODEL  HMCS          MODEL_NAME
ms01  XXXXXXX     8203-E4A    hmc01,hmc02   IBM Power 520 Express
ms02  XXXXXXX     9133-55A    hmc01,hmc02   IBM System p5 550Q
ms05  XXXXXXX     8233-E8B    hmc03,hmc04   IBM Power 750 Express
ms06  XXXXXXX     8406-71Y    hmc03,hmc04   IBM Power BladeCenter PS701 Express
ms11  XXXXXXX     8284-22A    hmc05,hmc06   IBM Power System S822
ms13  XXXXXXX     8286-41A    hmc05,hmc06   IBM Power System S814
ms14  XXXXXXX     8205-E6B    hmc05,hmc06   IBM Power 740 Express
ms16  XXXXXXX     9117-MMC    hmc01,hmc02   IBM Power 770
ms19  XXXXXXX     9009-22A    hmc07,hmc08   IBM Power System S922
...
$
```

3. Administration of Authorized Keys

Setting up and managing authorized keys is now very easy with the *hmc* command.

The public keys in your own *authorized_keys2* on an HMC can be listed using "*hmc lsauthkeys*":

```
$ hmc lsauthkeys hmc01
LINE
ssh-rsa AAAAB3…Jvtw== user01

$ hmc lsauthkeys hmc01 user04
LINE
ssh-rsa AAAAB3…jhvhw== user04

$
```

A public key can be added to the *authorized_keys2* on the HMC with the command "*hmc addauthkeys*":

```
$ hmc addauthkeys hmc01 "ssh-rsa AAAAkasjkjaksjf testuser"

$ hmc lsauthkeys hmc01
LINE
```

```
ssh-rsa AAAAB3…Jvtw==
ssh-rsa XXYYYZZZ
ssh-rsa AAAAkasjkjaksjf testuser

$
```

Of course, a public key can also be removed easily, using the command "*hmc rmauthkeys*":

```
$ hmc rmauthkeys hmc01 "ssh-rsa XXYYYZZZ"
$ hmc lsauthkeys hmc01
LINE
ssh-rsa AAAAB3…Jvtw==
ssh-rsa AAAAkasjkjaksjf testuser

$ hmc rmauthkeys hmc01 testuser
$ hmc lsauthkeys hmc01
LINE
ssh-rsa AAAAB3…Jvtw==

$
```

4. Initiating a System Dump

A system dump with subsequent restart can now be initiated using the "*lpar dumprestart*" command:

```
$ lpar dumprestart lpar01
$
```

The current progress of the dump can be displayed using "*lpar lsrefcode*":

```
$ lpar lsrefcode lpar01
05/08/2020 10:50:08  00cb              03400000  Dump Init:6%
05/08/2020 10:50:07  00cb              03400000  Dump Init:5%
05/08/2020 10:50:06  00cb              03400000  Dump Init:5%
05/08/2020 10:50:06  00cb              03400000  Dump Init:4%
05/08/2020 10:50:06  00cb              03400000  Dump Init:4%
05/08/2020 10:50:06  00cb              03400000  Dump Init:3%
05/08/2020 10:50:05  00cb              03400000  Dump Init:3%
05/08/2020 10:50:05  00cb              03400000  Dump Init:2%
05/08/2020 10:50:05  00cb              03400000  Dump Init:2%
05/08/2020 10:49:57  00cb              03400000  Dump Init:1%
05/08/2020 10:49:57  00cb              03400000  Dump Init:1%
05/08/2020 10:49:57  00cb              03400000  Dump Init:0%
05/08/2020 10:49:57  00cb              03400000  -
05/08/2020 10:49:57  -                 03400000  -
05/08/2020 10:49:57  D200A200          03400000  -
…
$
```

2. What's New in 1.4.0.x

Up to version 1.2 of the LPAR tool, it was written in the C programming language. To better test the program code, we decided to rewrite the commands step-by-step in C++. In version 1.3.x the command *hmc* was already implemented in C ++. With version 1.4.x the commands *ms* and *lpar* were also implemented in C++. The functionality is retained or expanded at the same time. In the present version *1.4.0.x* the so-called unit tests have been extended to about 400. All features of the previous versions have been adopted. In addition, some useful feature and sub-commands have been added, which will be briefly described below. Detailed information can be found in the user manual.

1. Integrated Test License

To test the previous versions, it was necessary to request a so-called trial license, this was very easy by filling out a form on our web page. In version 1.3.0.0 and later, a test license is included, so that you can try the software immediately after download and installation without having to request a trial license. The integrated test license has a limited validity of 2 months from the build date of the version. On our download page it is noted how long the integrated test license is valid.

The integrated test license supports up to HMCs, up to 100 managed systems and a up to 1000 LPARs.

The test license of version 1.4.0.1 is valid until 31 october 2019.

For more extensive testing, it is still possible to request a trial license, which will support up to 4 HMCs and any number of Managed Systems and LPARs.

2. Support of different output formats

When outputting information, the output in stanza format, JSON and YAML are supported in addition to the previous standard output format. The output format will be briefly shown here using the example of the command "*hmc lshmcusr*". First the standard output format:

```
$ hmc lshmcusr hmc01
NAME     DESCRIPTION       TASKROLE       RESOURCEROLE
root     root              hmcsuperadmin  ALL:
as       Armin Schmidt     hmcsuperadmin  ALL:
hscroot  HMC Super User    hmcsuperadmin  ALL:
$
```

For output in stanza format, the option "*-f*" or "*-o stanza*" can be used:

```
$ hmc lshmcusr -f hmc01
root:
     authentication_type = local
     description = root
     disabled = 0
     idle_timeout = 0
     inactivity_expiration = 0
     min_pwage = 0
     name = root
     password_encryption = sha512
```

```
     pwage = 99999
     remote_ssh_access = 1
     remote_webui_access = 1
     resourcerole = ALL:
     resources = <ResourceID = ALL:><UserDefinedName = AllSystemResources>
     session_timeout = 0
     taskrole = hmcsuperadmin
     verify_timeout = 15
as:
     authentication_type = local
     description = Armin Schmidt
     disabled = 0
     idle_timeout = 0
     inactivity_expiration = 0
     min_pwage = 0
     name = as
     password_encryption = sha512
     pwage = 99999
     remote_ssh_access = 1
     remote_webui_access = 1
     resourcerole = ALL:
     resources = <ResourceID = ALL:><UserDefinedName = AllSystemResources>
     session_timeout = 0
     taskrole = hmcsuperadmin
     verify_timeout = 15
hscroot:
     authentication_type = local
     description = HMC Super User
     disabled = 0
     idle_timeout = 0
     inactivity_expiration = 0
     min_pwage = 0
     name = hscroot
     password_encryption = sha512
     pwage = 99999
     remote_ssh_access = 1
     remote_webui_access = 1
     resourcerole = ALL:
     resources = <ResourceID = ALL:><UserDefinedName = AllSystemResources>
     session_timeout = 0
     taskrole = hmcsuperadmin
     verify_timeout = 15
$
```

In contrast to the standard output format, all available attributes are displayed here. The attribute names are the names used by default on the HMC.

An output in JSON format can be obtained with the option "*-j*" or "*-o json*":

```
$ hmc lshmcusr -o json hmc01
{
     "authentication_type": "local",
     "description": "root",
     "disabled": "0",
     "idle_timeout": "0",
     "inactivity_expiration": "0",
     "min_pwage": "0",
     "name": "root",
     "password_encryption": "sha512",
     "pwage": "99999",
     "remote_ssh_access": "1",
     "remote_webui_access": "1",
```

```
    "resourcerole": "ALL:",
    "resources": "<ResourceID = ALL:><UserDefinedName = AllSystemResources>",
    "session_timeout": "0",
    "taskrole": "hmcsuperadmin",
    "verify_timeout": "15"
}
{
    "authentication_type": "local",
    "description": "Armin Schmidt",
    "disabled": "0",
    "idle_timeout": "0",
    "inactivity_expiration": "0",
    "min_pwage": "0",
    "name": "as",
    "password_encryption": "sha512",
    "pwage": "99999",
    "remote_ssh_access": "1",
    "remote_webui_access": "1",
    "resourcerole": "ALL:",
    "resources": "<ResourceID = ALL:><UserDefinedName = AllSystemResources>",
    "session_timeout": "0",
    "taskrole": "hmcsuperadmin",
    "verify_timeout": "15"
}
{
    "authentication_type": "local",
    "description": "HMC Super User",
    "disabled": "0",
    "idle_timeout": "0",
    "inactivity_expiration": "0",
    "min_pwage": "0",
    "name": "hscroot",
    "password_encryption": "sha512",
    "pwage": "99999",
    "remote_ssh_access": "1",
    "remote_webui_access": "1",
    "resourcerole": "ALL:",
    "resources": "<ResourceID = ALL:><UserDefinedName = AllSystemResources>",
    "session_timeout": "0",
    "taskrole": "hmcsuperadmin",
    "verify_timeout": "15"
}
$
```


If required, output can also be in YAML format using the option "*-y*" or "*-o yaml*":


```
$ hmc lshmcusr -y hmc01
---
  authentication_type: local
  description: root
  disabled: 0
  idle_timeout: 0
  inactivity_expiration: 0
  min_pwage: 0
  name: root
  password_encryption: sha512
  pwage: 99999
  remote_ssh_access: 1
  remote_webui_access: 1
  resourcerole: ALL:
  resources: <ResourceID = ALL:><UserDefinedName = AllSystemResources>
  session_timeout: 0
  taskrole: hmcsuperadmin
  verify_timeout: 15
```

```
---
  authentication_type: local
  description: Armin Schmidt
  disabled: 0
  idle_timeout: 0
  inactivity_expiration: 0
  min_pwage: 0
  name: as
  password_encryption: sha512
  pwage: 99999
  remote_ssh_access: 1
  remote_webui_access: 1
  resourcerole: ALL:
  resources: <ResourceID = ALL:><UserDefinedName = AllSystemResources>
  session_timeout: 0
  taskrole: hmcsuperadmin
  verify_timeout: 15
---
  authentication_type: local
  description: HMC Super User
  disabled: 0
  idle_timeout: 0
  inactivity_expiration: 0
  min_pwage: 0
  name: hscroot
  password_encryption: sha512
  pwage: 99999
  remote_ssh_access: 1
  remote_webui_access: 1
  resourcerole: ALL:
  resources: <ResourceID = ALL:><UserDefinedName = AllSystemResources>
  session_timeout: 0
  taskrole: hmcsuperadmin
  verify_timeout: 15
$
```

Currently, these output formats are available for the *hmc*, *ms* and *lpar* command. A corresponding extension of the *vios* command will be available shortly.

3. Selection of the Output Format

Which data records are to be output for output commands can be determined by means of selections. You can use string comparisons, regular expressions or numeric comparisons.

Comparison with a string:

```
$ lpar status -s state=Running
NAME                  LPAR_ID  LPAR_ENV   STATE    PROFILE   SYNC  RMC       PROCS
PROC_UNITS  MEM    OS_VERSION
aix01                    9        aixlinux   Running  standard  0     active    2
0.2         24576  AIX 7.1 7100-05-02-1810
aix03                    22       aixlinux   Running  standard  0     active    2
0.2         24576  AIX 7.1 7100-05-02-1810
…
$
```

Or LPARs not in the *Running* state:

```
$ lpar status -s state!=Running
NAME                 LPAR_ID  LPAR_ENV  STATE          PROFILE   SYNC  RMC       PROCS
PROC_UNITS  MEM   OS_VERSION
aix02                18       aixlinux  Not Activated  -         0     inactive  0      -
          0     Unknown
aix04                26       aixlinux  Not Activated  standard  0     inactive  1
0.4         1024  Unknown
…
$
```

Regular expressions:

```
$ lpar status -s os_version~7100-04
NAME          LPAR_ID  LPAR_ENV  STATE    PROFILE   SYNC  RMC     PROCS  PROC_UNITS
MEM    OS_VERSION
lpar11        19       aixlinux  Running  standard  0     active  3      0.9
81920  AIX 7.1 7100-04-04-1717
lpar13        19       aixlinux  Running  standard  0     active  3      0.9
81920  AIX 7.1 7100-04-04-1717
…
$
```

Numerical comparisons are also possible:

```
$ lpar lsmem -s curr_mem:gt:32768
              MEMORY          MEMORY           HUGE_PAGES
LPAR_NAME     MODE  AME  MIN   CURR   MAX     MIN  CURR  MAX
lpar11        ded   1.0  1024  81920  163840  0    0     0
lpar13        ded   1.0  1024  81920  163840  0    0     0
…
$
```

Besides *gt* (greater than), *ge* (greater or equal), *eq* (equal), *ne* (not equal), *lt* (less than) and *le* (less or equal) are supported. The chosen comparison is placed between 2 colons.

Of course, this can be combined with all supported output formats.

4. Selection of the Output Fields

Which data fields (or attributes) are to be shown for output commands can be determined with the help of the new *-F* option. This makes it easy to implement your own output formats.

```
$ lpar lsmem -F lpar_name:curr_mem
aix11:0
lpar9:1024
lpar11:81920
…
$
```

This can of course be combined with selections and all supported output formats:

```
$ lpar lsmem -s curr_mem:lt:4096 -y -F lpar_name:curr_mem
---
  curr_mem: 0
  lpar_name: aix11
---
```

```
  curr_mem: 1024
  lpar_name: lpar9
---
  curr_mem: 0
  lpar_name: aix02
…
$
```

The selection of data fields can be combined with all output formats and the selection of data sets.

5. Support for SR-IOV

The administration of SR-IOV is supported by the following new sub-commands:

- *ms chsriov* - Change the attributes of SR-IOV adapters and physical ports
- *ms lssriov* - List attributes of SR-IOV adapters, physical and logical ports
- *lpar addsriov* - Add an SR-IOV logical port to an LPAR
- *lpar chsriov* - Change the attributes of an SR-IOV logical port
- *lpar lssriov* - Display attributes of SR-IOV logical ports of an LPAR
- *lpar rmsriov* - Remove one SR-IOV logical port from an LPAR R

6. Support for vNICs

The administration of vNICs is supported by the following new sub-commands:

- *lpar addvnic* - Add a vNIC adapter to an LPAR
- *lpar addvnicbkdev* - Add a vNIC backing device
- *lpar rmvnicbkdev* - Removing a vNIC BackingDevices
- *lpar rmvnic* - removing a vNIC adapter

7. Administration of VLAN-IDs

The administration of VLAN IDs of virtual Ethernet adapters has been simplified. VLAN IDs can now be added or removed with the following sub-commands:

- *lpar addvlan* - Add VLANs to a virtual adapter
- *lpar rmvlan* - Removing VLANs from a virtual adapter

8. Performance-Optimization for some Sub-Commands

For many sub-commands that display information for multiple managed systems or LPARs, performance has increased by a factor of 10 in some cases.