PowerCampus[01]

LPAR-Tool 1.4.2.x

Benutzerhandbuch

Version: 1.4.2.0

https://www.powercampus.de

https://www.powercampus01.com

Vorwort

Einführung

Dieses Benutzerhandbuch ist für Administratoren und Benutzer gedacht, die das *LPAR-Tool* zur Administration und Konfiguration der POWER Virtualisierung einsetzen. Das Handbuch setzt folgendes voraus:

- grundlegende Kenntnis zum Arbeiten auf der Kommandozeile eines UNIX Systems
- grundlegendes Verständnis der Virtualisierungskonzepte und Funktionen der POWER Virtualisierung

Das Benutzerhandbuch kann über den Download-Bereich auf der PowerCampus^{01} Webseite heruntergeladen werden:

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

Weitere Informationen

Weitere Informationen zum *LPAR-Tool* sind im Tool-Bereich auf der PowerCampus^{01} Webseite verfügbar:

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

Hilfe bei Problemen

Bei fehlerhaftem Funktionieren des *LPAR-Tool*s kann der technische Support von PowerCampus^{01} kontaktiert werden. Über die folgende URL kann ein Software-Call für das *LPAR-Tool* eröffnet werden:

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

Der Support ist über die nachfolgende Email-Adresse erreichbar:

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

Software Updates des *LPAR-Tool*s können über den Download-Bereich auf der PowerCampus^{01} Webseite heruntergeladen werden:

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

1. Einführung

Das *LPAR-Tool* besteht im wesentlichen aus den 4 Programmen *hmc*, *ms*, *lpar* und *vios*. Diese Programme lassen sich von der Kommandozeile benutzen um eine POWER-Virtualisierungsumgebung zu administrieren. Die meisten Aufgaben, die ansonsten über die HMC-GUI durchgeführt werden, lassen sich mit diesen Kommandos bequem und effizient von der Kommandozeile aus durchführen.

1. Voraussetzungen

Das *LPAR-Tool* gibt es in Versionen für AIX, Linux und MacOS.

Notwendige Voraussetzung für die Funktionalität des *LPAR-Tool*s ist eine SSH-Verbindung zu der oder den HMCs. Per Default benutzt das LPAR-Tool den Account des eingeloggten Benutzers als HMC-User. Ist der HMC-User ein anderer, kann dies aber angegeben werden.

Der SSH-Login auf die HMC sollte ohne Eingabe eines Passwortes oder eines Passphrases möglich sein, was z.B. durch Verwendung des *ssh-agent* erreicht werden kann. D.h. der Public Key des Users muss auf der HMC zuvor abgelegt werden. Wird kein *ssh-agent* verwendet, ist eine Benutzung des *LPAR-Tool*s trotzdem möglich, es erfolgt aber dann bei jedem Aufruf des Tools eine Abfrage des Passworts oder des Passphrases.

Für die Benutzung des *LPAR-Tool*s ist eine gültige Lizenz notwendig.

2. Installation

Für die Installation des LPAR-Tools gibt es für jedes unterstützte Betriebssystem zwei verschiedene Möglichkeiten: Installation eines Paketes oder Installation eines *tar*-Files.

Empfohlen ist die Installation des Paketes für das entsprechende Betriebssystem. Dies setzt Administrationsrechte auf dem zu installierenden System voraus. Das LPAR-Tool steht damit prinzipiell allen Nutzern auf dem System zur Verfügung. Natürlich braucht ein Nutzer nach wie vor einen Login auf der oder den HMCs.

Die Möglichkeit ein *tar*-File für die Installation zu verwenden, ist für den Fall gedacht, das der Nutzer keine Administrationsrechte besitzt. Das *tar*-File kann der Nutzer dann in seinem Home-Verzeichnis auspacken. Das LPAR-Tool steht dann nur diesem Nutzer zur Verfügung. Natürlich können andere Nutzer auf dem gleichen System das LPAR-Tool ebenfalls per *tar* in ihrem Home-Verzeichnis installieren und nutzen.

3. Installation unter AIX

Für die Installation unter AIX steht ein Paket zur Verfügung: *pwrcmps.lpar.rte.1.X.X.X.bff*. Dieses enthält das Fileset *pwrcmps.lpar.rte*. Das Paket kann von der Website *https://powercampus.de/download* heruntergeladen werden.

Auf dem AIX-System muss die Installation mit *root*-Rechten ausgeführt werden:

```
# installp -acXYd pwrcmps.lpar.rte.1.4.0.1.bff all
+-----------------------------------------------------------------------------+
                    Pre-installation Verification...
+-----------------------------------------------------------------------------+
```

```
Verifying selections...done
Verifying requisites...done
Results...

SUCCESSES
---------
  Filesets listed in this section passed pre-installation verification
  and will be installed.

  Selected Filesets
  -----------------
  pwrcmps.lpar.rte 1.4.0.1                  # LPAR tool

  << End of Success Section >>

+-----------------------------------------------------------------------------+
                    BUILDDATE Verification ...
+-----------------------------------------------------------------------------+
Verifying build dates...done
FILESET STATISTICS
------------------
    1  Selected to be installed, of which:
        1  Passed pre-installation verification
  ----
    1  Total to be installed

+-----------------------------------------------------------------------------+
                         Installing Software...
+-----------------------------------------------------------------------------+

installp: APPLYING software for:
        pwrcmps.lpar.rte 1.4.0.1

Finished processing all filesets.  (Total time:  3 secs).

+-----------------------------------------------------------------------------+
                                Summaries:
+-----------------------------------------------------------------------------+

Installation Summary
--------------------
Name                        Level           Part        Event       Result
-------------------------------------------------------------------------------
pwrcmps.lpar.rte            1.4.0.1         USR         APPLY       SUCCESS
#
```

Die Dateien des Fileset werden unter */opt/pwrcmps* installiert. Die Programme (*hmc*, *ms*, *lpar* und *vios*) befinden sich unter */opt/pwrcmps/bin:*

```
# ls -l /opt/pwrcmps/bin
total 104256
-rwxr-xr-x    1 root     system     14065602 Sep  2 16:30 hmc
-rwxr-xr-x    1 root     system     15241792 Sep  2 16:30 lpar
-rwxr-xr-x    1 root     system     14832670 Sep  2 16:30 ms
-rwxr-xr-x    1 root     system      9230400 Sep  2 16:30 vios
#
```

*E*ine Beispiel-Konfigurationsdatei (*sample.lpar.cfg*) findet sich unter */opt/pwrcmps/etc*, sowie die Dateien *Copyright* und *LicenseAgreement*.

```
# ls -l /opt/pwrcmps/etc
total 24
-rw-r--r--    1 root     system          108 Aug 30 13:27 Copyright
-rw-r--r--    1 root     system          273 Aug 30 13:27 LicenseAgreement
-rw-r--r--    1 root     system          542 Aug 30 13:27 sample.lpar.cfg
#
```

Der Suchpfad der Shell (*PATH*-Variable) sollte um */opt/pwrcmps/bin* erweitert.

4. Installation unter Linux

Für die Installation unter Linux steht ein RPM-Pakete zur Verfügung: lpar-rte-1.X-X.rpm. Das RPM-Paket kann von der Website *https://powercampus.de/download* heruntergeladen werden.

```
# rpm -iv pwrcmps-lpar-rte-1.4.0-1.x86_64.rpm
Preparing packages...
pwrcmps-lpar-rte-1.4.0-1.x86_64
#
```

Die Dateien des Fileset werden unter */opt/pwrcmps* installiert. Die Programme (*hmc*, *ms*, *lpar* und *vios*) befinden sich unter */opt/pwrcmps/bin:*

```
# ls -l /opt/pwrcmps/bin
total 104256
-rwxr-xr-x    1 root     system     14065602 Sep  2 16:30 hmc
-rwxr-xr-x    1 root     system     15241792 Sep  2 16:30 lpar
-rwxr-xr-x    1 root     system     14832670 Sep  2 16:30 ms
-rwxr-xr-x    1 root     system      9230400 Sep  2 16:30 vios
#
```

*E*ine Beispiel-Konfigurationsdatei (*sample.lpar.cfg*) findet sich unter */opt/pwrcmps/etc*, sowie die Dateien *Copyright* und *LicenseAgreement*.

```
# ls -l /opt/pwrcmps/etc
total 24
-rw-r--r--    1 root     system          108 Aug 30 13:27 Copyright
-rw-r--r--    1 root     system          273 Aug 30 13:27 LicenseAgreement
-rw-r--r--    1 root     system          542 Aug 30 13:27 sample.lpar.cfg
#
```

Der Suchpfad der Shell (*PATH*-Variable) sollte um */opt/pwrcmps/bin* erweitert.

5. Installation unter MacOS

Die Installation unter MacOS ist aktuell nur über ein *tar*-File möglich.

6. Installation *tar*-File

Ist eine Installation der Pakete für das entsprechende OS-Derivat nicht möglich oder nicht gewünscht, dann kann das LPAR-Tool auch durch Entpacken eines *tar*-Files installiert werden.

Das notwendige *tar*-File kann von der Website *https://powercampus.de/download* heruntergeladen werden. Das *tar*-File sollte im Home-Verzeichnis ausgepackt werden.

```
$ pwd
/home/user01
$ tar xvf pwrcmps.lpar.rte.1.4.0.1.aix.tar
x Copyright, 108 bytes, 1 tape blocks
x LicenseAgreement, 273 bytes, 1 tape blocks
x hmc, 14065602 bytes, 27472 tape blocks
x ms, 14832670 bytes, 28971 tape blocks
x lpar, 15241792 bytes, 29770 tape blocks
x vios, 9230400 bytes, 18029 tape blocks
x sample.lpar.cfg, 542 bytes, 2 tape blocks

$
```

Die Binaries (*hmc*, *ms*, *lpar* und *vios*), sowie eine Beispiel-Konfigurationsdatei (sample.*lpar.cfg*) und eine Beispiel-Lizenz-Datei (sample.*lpar.lic*) sind dann direkt im Home-Verzeichnis. Die Binaries können anschließend an eine beliebige andere Stelle verschoben werden, z.B. *~/bin*. Die Konfigurationsdatei *.lpar.cfg* muss im Home-Verzeichnis stehen. Die Lizenz-Datei kann prinzipiell an beliebiger Stelle stehen, der Default ist aber das Home-Verzeichnis.

7. Konfiguration des LPAR-Tools

Bei der Installation des LPAR-Tools als Paket unter */opt/pwrcmps* wird eine Beispiel-Konfigurationsdatei */opt/pwrcmps/etc/sample.lpar.cfg* mit dem folgenden Inhalt angelegt:

```
$ cat /opt/pwrcmps/etc/lpar.cfg
# Directory where the files hmc.list, ms.list and lpar.list are stored
# Default: ~
#ConfigDirectory ~

# Where to find the license file. Default: /opt/pwrcmps/etc/lpar.lic
#LicenseFile /opt/pwrcmps/etc/lpar.lic

# How long (minutes) inactive master connection should persist. Default: 600
#ControlPersist 600

# Time interval within server is expected to send alive message.
# Default: 10 seconds
#ServerAliveInterval 10

# Maximum number of outstanding alive messages from master.
# Default: 2
#ServerAliveCountMax 2

# The lowest virtual slot for a client adapter.
# Default: 10
#LowestVirtualClientSlot 10

# The lowest virtual slot for a server adapter.
# Default: 20
# LowestVirtualServerSlot 20
```

Soll die Konfiguration des LPAR-Tools systemweit angepasst werden, kann die Beispiel-Konfigurationsdatei nach /*opt/pwrcmps/etc/lpar.cfg* kopiert und angepasst werden. In vielen Fällen ist die Default-Konfiguration aber schon passend. Jeder Nutzer des LPAR-Tools kann über die Datei *.lpar.cfg* in seinem Home-Verzeichnis eine abweichende Konfiguration haben. Die Benutzer-spezifische Konfigurationsdatei überschreibt die Einstellungen aus der System-weiten Konfigurationsdatei.

Das LPAR-Tool speichert die registrierten HMCs, Managed Systems und LPARs in den 3 Dateien *hmc.list*, *ms.list* und *lpar.list* ab. Über den Parameter *ConfigDirectory* kann konfiguriert werden, wo diese Dateien abgelegt werden sollen. Die Default-Einstellung ist „~“, das jeweilige Home-Verzeichnis.

Das LPAR-Tool benötigt einen gültigen Lizenz-Schlüssel. Über den Parameter LicenseFile kann angegeben werden, in welcher Datei die Lizenz eingetragen ist. Der Default ist */opt/pwrcmps/etc/lpar.lic* bzw. *~/.lpar.lic* (Home-Verzeichnis des Benutzers) und sollte nicht überschrieben werden.

Das LPAR-Tool verwendet SSH-Master-Connections, mit dem Parameter *ControlPersist* kann angegeben werden wie lange diese Connection (in Sekunden) bestehen bleiben soll. Der Default-Wert beträgt 5 Minuten.

8. Installation der Lizenz

Für die Benutzung des LPAR-Tools ist eine Lizenz erforderlich. Eine 30 Tage Trial-Lizenz ist über *info@powercampus.de* erhältlich.

Wurde das LPAR-Tool über ein Paket installiert, dann sollte die Lizenz in der Datei */opt/pwrcmps/etc/lpar.lic* eingetragen werden. Die Lizenz kann auch im Home-Verzeichnis der Nutzer unter *~/.lpar.lic* eingetragen werden.

Wurde das LPAR-Tool als *tar*-File installiert, dann kann die Lizenz im Prinzip an beliebiger Stelle installiert werden, der Pfad muss dann aber über das Konfigurations-File *.lpar.cfg* im Home-Verzeichnis des Benutzers konfiguriert werden.

Für die Generierung einer Lizenz werden die Seriennummern der verwendeten HMCs benötigt. Die Seriennummer erhält man z.B. durch Einloggen auf der HMC mittels *ssh* und dem Kommando „*lshmc -v*“:

```
$ ssh hmc01
hmc01> lshmc -v
…
*SE 123ABCD
…
hmc01> exit
$
```

(Die Seriennummer steht in der Zeile die mit „**SE*“ anfängt.)

Der Lizenz-Key muss über *PowerCampus*[01] angefordert werden. In der Regel wird der Lizenz-Key dann per Mail verschickt. Der Lizenz-Key aus der Mail muß dann in die Lizenz-Datei eingetragen werden:

```
# vi /opt/pwrcmps/etc/lpar.lic

LicenseId: 1
LicenseVersion: 1
LicenseType: trial
HMCs: 1234567,2345678
HMCs: 3456789,456789A
ExpirationDate: 01.09.2019
```

```
LicenseKey: 5cdb63906b09e352a781007eadc92dbf

#
```

Bzw. im Falle der Installation in ein Home-Verzeichnis:

```
$ vi ~/.lpar.lic

LicenseId: 1
LicenseVersion: 1
LicenseType: trial
HMCs: 1234567,2345678
HMCs: 3456789,456789A
ExpirationDate: 01.09.2019
LicenseKey: 5cdb63906b09e352a781007eadc92dbf

$
```

2. Benutzen des LPAR-Tools

1. Konfiguration von OpenSSH

Das LPAR-Tool greift mittels *ssh* auf die registrierten HMCs zu. Damit das LPAR-Tool funktionieren kann, wird auf jeder HMC ein Account benötigt, der mittels *ssh* erreichbar ist. Per Default wird angenommen das der Account auf den HMCs den gleichen Benutzernamen verwendet wie auf dem lokalen System. Jeder andere Benutzername ist aber auch möglich. Damit nicht ständig Passwörter oder Passphrases eingegeben werden müssen, sollte ein Public Key generiert werden und auf der HMC abgespeichert werden:

```
$ cat .ssh/id_rsa.pub
ssh-rsa XYS…

$ ssh HMC
…
HMC> mkauthkeys -a „ssh-rsa XYS…“
```

Nachdem nun der Public Key auf der HMC abgespeichert ist, sollte nun ein *ssh-agent* gestartet werden:

```
$ eval $( ssh-agent )
$ ssh-add
<Passphrase>
```

2. Registrieren einer HMC

Managed Systems und LPARs, die mit dem LPAR-Tool administriert werden sollen, müssen dem LPAR-Tool bekannt sein. Hierzu ist es notwendig die zugehörige(n) HMC(s) zu registrieren. Das Kommando „*hmc add*“ registriert die angegebene HMC, sowie alle an die HMC angebundenen Managed Systems und deren LPARs.

```
$ hmc add hmc01
hmc01:
   ms04
    > aix03
    > aix05
    > lpar17
    > ms04-vio2
    > ms04-vio1
   ms02
    > aix04
    > aix06
    > lpar18
…
$
```

Die Ausgabe zeigt die neu registrierten Managed Systems. Die registrierten HMCs lassen sich mit dem Kommando „*hmc list*“ oder „*hmc show*“ auflisten:

```
$ hmc list
```

```
hmc01
$ hmc show
NAME     SERIAL_NUM  TYPE_MODEL
hmc01    XXXXXXX     7042-CR9
$
```

Stimmt der Benutzername auf dem lokalen System nicht mit dem Benutzernamen auf der HMC überein, so muß der HMC-Benutzername beim Registrieren angegeben werden:

```
$ hmc add hscroot@hmc02
hmc02:
   ms04
    > aix03
    > aix05
    > lpar17
    > ms04-vio2
    > ms04-vio1
   ms02
    > aix04
    > aix06
    > lpar18
...
$
```

Auf diese Weise können alle HMCs registriert werden, zumindest insoweit es die Lizenzierung zuläßt.

Ist ein Managed System an zwei HMCs angebunden, das dürfte in der Praxis in der Regel der Fall sein, so sollte auch jeweils die zweite HMC registriert werden. Dies hat den Vorteil, das bei Nichtverfügbarkeit einer HMC, automatisch die andere HMC vom LPAR-Tool verwendet wird. Dies ist für den Benutzer transparent.

Natürlich kann man eine HMC auch wieder de-registrieren, hierzu gibt es das zu oben analoge Kommando „*hmc remove*“:

```
$ hmc show
NAME     SERIAL_NUM  TYPE_MODEL
hmc01    XXXXXXX     7042-CR9
hmc02    XXXXXXX     7042-CR9
$ hmc remove hmc01
$ hmc show
NAME     SERIAL_NUM  TYPE_MODEL
hmc02    XXXXXXX     7042-CR9
$
```

Die Informationen welche LPAR zu welchem Managed System und welches Managed System zu welcher HMC gehört, ist in 3 Dateien abgelegt: *hmc.list*, *ms.list* und *lpar.lis*t. Das LPAR-Tool liest diese Dateien ein, um die Zuordnung zu kennen. Wird nun eine neue LPAR über das HMC-GUI angelegt, oder wird eine LPAR per LPM über das HMC-GUI auf ein anderes Managed System verschoben, dann sind die gespeicherten Zuordnungen nicht vollständig und korrekt. In diesem Fall kann man die Zuordnungen neu einlesen lassen, indem alle HMCs kontaktiert und die Liste der Managed Systems und LPARs abgefragt wird. Hierzu dient das Kommando „*hmc rescan*“:

```
$ hmc rescan
hmc02:
   ms04
    > aix03
```

```
    > aix05
    > lpar17
    > ms04-vio2
    > ms04-vio1
   ms02
    > aix04
    > aix06
    > lpar18
...
$
```

Nach einem Durchlauf von „*hmc rescan*“ sind die Zuordnungen wieder korrekt.

3. Überblick über die Kommandos

Die 4 Kommandos des LPAR-Tools funktionieren auf ähnliche Weise. Jedes der Kommandos erwartet die Angabe eines Keywords. Das Keyword entscheidet über die auszuführende Funktionalität. Vor und nach dem Keyword können unterstützte Optionen angegeben werden. Den Abschluß bilden Argumente für die auszuführende Funktion:

```
command [options] keyword [options] arguments
```

Jedes der 4 Kommandos unterstützt die Keywords „*list*“ und „*show*“. Mit diesen beiden Keywords wird nur auf die internen Zuordnungsdateien zugegriffen, es erfolgt kein Zugriff per SSH auf die HMCs. Mit „*list*“ werden nur die Namen von registrierten HMCs, Managed Systems, LPARs oder Virtual-I/O-Server aufgelistet. Mit dem Keyword „*show*“ werden einige zusätzliche Informationen ausgegeben. Im folgenden einige Beispiele:

```
$ hmc list
hmc01
hmc02
$ ms show
NAME  SERIAL_NUM  TYPE_MODEL  HMCS          MODEL_NAME
ms02  XXXXXXX     9009-22A    hmc01,hmc02   IBM Power S922
ms04  XXXXXXX     9009-22A    hmc01,hmc02   IBM Power S922
…
$ lpar list
lpar1
lpar2
lpar3
…
$ vios show
VIOS        ID   SERIAL    TYPE        MS
ms02-vio1   1    XXXXXXXX  vioserver   ms02
ms02-vio2   2    XXXXXXXX  vioserver   ms02
ms04-vio1   1    XXXXXXXX  vioserver   ms04
…
$
```

In allen Fällen kann man über angegebene Argumente die Ausgabe eingrenzen:

```
$ hmc list hmc01 hmc02
hmc01
hmc02
$ hmc show hmc02
NAME     SERIAL_NUM  TYPE_MODEL
```

```
hmc02    XXXXXXX     7042-CR9
$
```

Die Kommandos *lpar* und *vios* unterstützen noch die Option *-m*, mit dieser kann ein Managed System angegeben werden und die Option *-h*, mit der eine HMC angegeben werden kann. Es werden dann nur die LPARs bzw. Virtual-I/O-Server des angegebenen Managed Systems bzw. der angegebenen HMC ausgegeben:

```
$ lpar -m ms02 list
lpar1
lpar2
ms02-vio1
ms02-vio2
$ vios -m ms01 show
VIOS       ID   SERIAL    TYPE       MS
ms02-vio1  1    XXXXXXXX  vioserver  ms02
ms02-vio2  2    XXXXXXXX  vioserver  ms02
$
```

Möchte man die möglichen Keywords für eines der Kommandos sehen, kann man das Kommando mit dem Keyword *help* und Argument *usage* aufrufen:

```
$ lpar help usage
USAGE:
   lpar [<option> ...] <keyword> [<option> ...] [<argument> ...]
   lpar -V

Recognized keywords:
   activate - Activate AIX, Linux, IBM i or virtual I/O server partition
   actvnicbkdev - Make virtual NIC backing device active
   addeth - Add virtual ethernet adapter
…
$
```

Optionen können generell entweder vor oder nach dem Keyword angegeben werden, die folgenden Kommando-Aufrufe sind daher äquivalent:

```
$ lpar -p standard -b sms activate lpar1
oder
$ lpar -p standard activate -b sms lpar1
oder
$ lpar activate -p standard -b sms lpar1
oder
$ lpar -b sms activate -p standard lpar1
$
```

Alle Keywords (bis auf „*help*“, „*list*“ und „*show*“) unterstützen die Option „*-v*“. Diese Option steht für *verbose-only*. Damit ist gemeint das die angegebene Funktion nicht ausgeführt wird, sondern die Kommandos angezeigt werden, die auf der HMC ausgeführt werden würden:

```
$ lpar -v -c addfc lpar1 11 ms01-vio1 105
hmc01: chhwres -m ms01 -r virtualio –rsubtype fc -o a -p lpar1 -s 11 -a
adapter_type=client,remote_lpar_name=ms01-vio1,remote_slot_num=105
hmc01: lssyscfg -m ms01 -r lpar –filter lpar_names=lpar1 -F curr_profile
```

```
hmc01: chsyscfg -m ms01 -r prof -i
lpar_name=lpar1,name=standard,“virtual_fc_adapters+=„“11/client//ms01-vio1/105//0“““
$
```

Neben den HMC-CLI Kommandos wird auch die HMC ausgegeben, auf welcher das oder die Kommandos durchgeführt wurden. Dies ist ganz nützlich, wenn man einmal sehen möchte welche Kommandos auf der HMC Kommandozeile letztlich ausgeführt werden.

Die Version des installierten LPAR-Tools lässt sich durch Angabe der Option „*-V*“ mit jedem der 4 Kommandos anzeigen:

```
$ ms -V
Version: 1.4.0.1 (20190827)
Copyright (c) 2018-2019 by PowerCampus 01 GmbH
Copyright (c) 2006-2018 Dr. Armin Schmidt
$
```

Die Version des LPAR-Tool Kommandos *ms* ist *1.4.0.1* und das Build-Datum ist der 27.08.2019.

4. Verwendung des Keywords *help*

Alle LPAR-Tool Kommandos liefern bei Aufruf ohne Argumente eine Usage-Meldung zurück. Diese enthält eine Übersicht aller verfügbaren Keywords:

```
$ lpar
USAGE:
   lpar [<option> ...] <keyword> [<option> ...] [<argument> ...]
   lpar -V

Recognized keywords:
   activate - Activate AIX, Linux, IBM i or virtual I/O server partition
   actvnicbkdev - Make virtual NIC backing device active
   addeth - Add virtual ethernet adapter
…
$
```

Wird eines der Kommandos mit dem Keyword ‚*help*‘ aufgerufen, wird die Funktionalität der Hilfe ausgegeben.

```
$ lpar help
Help is available for the following categories:

   lpar help eth fc io led lpm mem memory
   lpar help power proc processor prof profile scsi serial
   lpar help sriov vnic

Specific help is available for each of the supported keywords:

   lpar help <keyword>

For a complete list of all keywords try:

   lpar help usage
$
```

Gibt man als Argument für das Keyword ‚*help*' eines der angegebene Themen an, werden alle Keywords die in Zusammenhang mit diesem Keyword stehen aufgelistet. Damit lässt sich sofort erkennen, welche Kommandos für einen Bereich verfügbar sind. Möchte man beispielsweise wissen, welche Keywords im Zusammenhang mit Power existieren, kann ‚*lpar help power*' verwendet werden:

```
$ lpar help power
USAGE: lpar [<option> ...] <keyword> [<option> ...] [<argument> ...]

Recognized keywords for topic 'power' are:
   [-h <hmc>] [-m <ms>] [-p <profile>] activate [-b <bootmode>] [-c] [-k <keylock>] [-v]
<lpar>
   [-h <hmc>] [-m <ms>] shutdown [-c] [-f] [-i] [-o <operation>] [-r] [-v] <lpar>
$
```

Möchte man genauere Informationen zu Optionen oder Argumenten eines Keywords, so kann das entsprechende Keyword als Argument für *help* angegeben werden, hier z.B. das Keyword *shutdown* des Kommandos *lpar*:

```
$ lpar help shutdown
USAGE:
   lpar [-h <hmc>] [-m <ms>] shutdown [-c] [-f] [-i] [-o <operation>] [-r] [-v] <lpar>

DESCRIPTION

Performs a shutdown of the specified LPAR.

   -c : open a console session
   -f : force shutdown
   -i : shutdown immediately
   -o : operation to perform
      shutdown - shuts down LPAR
      osshutdown - issue OS shutdown (default)
      dumprestart - initiate dump and then restart
      retrydump - retry dump and restart (only valid for IBM i)
   -r : restart LPAR

EXAMPLES:

Shutdown LPAR aix02:
   lpar shutdown aix02

Shutdown and then restart LPAR aix02:
   lpar shutdown -r aix02
$
```

Neben den Optionen und Argumenten werden auch eventuell verfügbare Attribute aufgelistet und erläutert. Zusätzlich werden einige Beispiele der Nutzung des Keywords gezeigt.

Gerade am Anfang der Nutzung des LPAR-Tools kann die große Anzahl an Funktionen und Keywords den Benutzer überraschen und ein Auffinden der gewünschten Funktionalität erschweren. Die Möglichkeit eine themenspezifische Auflistung, wie oben gezeigt (Beispiel Power) kann hier enorm hilfreich sein.

5. Auswahl der HMC, des Managed Systems oder der LPAR

Bei den meisten Sub-Kommandos muß ein Objekt oder mehrere Objekte angegeben werden, auf welche sich das Sub-Kommando dann bezieht. Sub-Kommandos die Änderungen bewirken erwarten in der Regel immer genau ein Objekt. Werden mehrere Objekte angegeben oder ist die Auswahl nicht eindeutig dann wird das Kommando nicht ausgeführt und eine Fehlermeldung weißt darauf hin das genau ein Objekt angegeben werden muß:

```
$ lpar activate lpar1
ERROR:
…
$
```

In einem solchen Fall müssen die Optionen *-h* und/oder *-m* verwendet werden, um das Objekt eindeutig anzugeben. Gibt es nur eine LPAR namens *lpar1* die über die HMC *hmc01* erreichbar ist, dann kann Eindeutigkeit über die Option ‚*-h hmc01*' erreicht werden:

```
$ lpar -h hmc01 activate lpar1
$
```

Alternativ kann aber auch das Managed System der LPAR angegeben werden:

```
$ lpar -m ms01 activate lpar1
$
```

Gibt es das Managed System *ms01* mehrfach (eher unwahrscheinlich), dann muß sowohl die HMC als auch das Managed System angegeben werden:

```
$ lpar -h hmc01 -m ms01 activate lpar1
$
```

Sind die Namen aller LPARs, Managed Systems und HMCs eindeutig (also nur jeweils einmal vergeben), dann sind die Optionen *-h* und *-m* nicht notwendig. Dies erspart dann auch einiges an Tip-Arbeit!

Bei Kommandos die ein oder mehrere Objekte akzeptieren ist Eindeutigkeit nicht notwendig. Gibt es beispielsweise 2 LPARs mit gleichem Namen auf verschiedenen Managed Systems, dann bezieht sich das Kommando einfach auf beide LPARs:

```
$ lpar show lpar1
…
```

Das LPAR-Tool erlaubt Wildcards bei der Angabe von Objekten. Damit lassen sich beispielsweise sehr schnell alle LPARs auflisten, die mit „aix" beginnen:

```
$ lpar show aix*
NAME   ID  SERIAL   LPAR_ENV  MS    HMCS
aix04  3   XXXXXXX  aixlinux  ms06  hmc01,hmc02
aix05  4   XXXXXXX  aixlinux  ms03  hmc01,hmc02
…
$
```

Da die Shell aber zuerst Wildcards interpretiert und auswertet kann das zu ungewollten Argumenten führen, hier gezeigt am Beispiel von **db**. Der Wildcard passt in diesem Fall auf Dateien im aktuellen Verzeichnis und wird somit von der Shell durch diese Liste der Dateien im aktuellen Verzeichnis ersetzt. Das LPAR-Tool wird dann mit dieser Liste aufgerufen:

```
$ lpar show *db*
ERROR:  no matching LPAR 'audit_db2.tar' found
USAGE:
   lpar [-h <hmc>] [-m <ms>] show [{-o <format>|-f|-j|-y}] [-F <fields>] [-s <selections>]
[<lpar> …]
$
```

Dies läßt sich verhindern indem man entweder das Argument in Anführungszeichen oder Apostrophe setzt, oder einen Backslash voranstellt:

```
$ lpar show „*db*“
NAME     ID  SERIAL   LPAR_ENV  MS    HMCS
aixdb01  14  XXXXXXX  aixlinux  ms05  hmc01,hmc02
aixdb02  9   XXXXXXX  aixlinux  ms01  hmc01,hmc02
…
$ lpar show ‚*db*‘
NAME     ID  SERIAL   LPAR_ENV  MS    HMCS
aixdb01  14  XXXXXXX  aixlinux  ms05  hmc01,hmc02
aixdb02  9   XXXXXXX  aixlinux  ms01  hmc01,hmc02
…

$
```

6. Auswahl von HMCs

Eine HMC kann entweder über ihren Namen oder ihre Seriennummer angegeben werden:

```
$ hmc show hmc01
NAME   SERIAL_NUM  TYPE_MODEL
hmc01  1234567     7042-CR9
$ hmc show 1234567
NAME   SERIAL_NUM  TYPE_MODEL
hmc01  1234567     7042-CR9
$
```

In beiden Fällen sind Wildcards erlaubt:

```
$ hmc show hmc*
NAME   SERIAL_NUM  TYPE_MODEL
hmc01  1234567     7042-CR9
hmc02  1238351     7042-CR9
…
$ hmc show 123*
NAME   SERIAL_NUM  TYPE_MODEL
hmc01  1234567     7042-CR9
hmc01  1238351     7042-CR9
$
```

Außerdem können HMCs auch über Typ und Modell ausgewählt werden, z.B. alle HMCs vom Typ 7042-CR9:

```
$ hmc show 7042-CR9
NAME   SERIAL_NUM  TYPE_MODEL
hmc01  1234567     7042-CR9
hmc02  1238351     7042-CR9
…
$
```

Auch hier sind Wildcards erlaubt:

```
$ hmc show 7042*
NAME   SERIAL_NUM  TYPE_MODEL
hmc01  1234567     7042-CR9
hmc02  1238351     7042-CR9
…
$ hmc show *CR8
NAME   SERIAL_NUM  TYPE_MODEL
hmc05  XXXXXXX     7042-CR8
hmc06  XXXXXXX     7042-CR8
$
```

7. Auswahl von Managed Systems

Ein Managed System kann entweder über den Namen oder die Seriennummer angegeben werden:

```
$ ms show ms05
NAME  SERIAL_NUM  TYPE_MODEL  HMCS
ms05  0123456     8205-E6B    hmc01,hmc02
$ ms show 0123456
NAME  SERIAL_NUM  TYPE_MODEL  HMCS
ms05  0123456     8205-E6B    hmc01,hmc02
$
```

In beiden Fällen können auch Wildcards verwendet werden:

```
$ ms show ms0*
NAME  SERIAL_NUM  TYPE_MODEL  HMCS
ms01  XXXXXXX     8205-E6B    hmc01,hmc02
ms02  XXXXXXX     8205-E6B    hmc01,hmc02
…
$ ms show 0123*
NAME  SERIAL_NUM  TYPE_MODEL  HMCS
ms05  XXXXXXX     8205-E6B    hmc01,hmc02
ms06  XXXXXXX     8205-E6B    hmc01,hmc02
$
```

Außerdem können Managed Systems auch über Typ und Model oder nur den Typ ausgewählt werden:

```
$ ms show 8205-E6D
NAME  SERIAL_NUM  TYPE_MODEL  HMCS
ms01  XXXXXXX     8205-E6B    hmc01,hmc02
ms02  XXXXXXX     8205-E6B    hmc01,hmc02
…
```

```
$ ms show 8205
NAME  SERIAL_NUM  TYPE_MODEL  HMCS
ms01  XXXXXXX     8205-E6B    hmc01,hmc02
ms02  XXXXXXX     8205-E6B    hmc01,hmc02
…
$
```

Auch hier sind wieder Wildcards erlaubt:

```
$ ms show 820*
…
$ ms show 8205-E*
…
$
```

Bei allen obigen Beispielen wird aus allen Managed Systems die dem LPAR-Tool bekannt sind ausgewählt. Wird mit der Option ‚*-h*‘ eine HMC angegeben, dann wird die Auswahl beschränkt auf Managed Systems die an diese HMC angebunden sind. Damit lassen sich Beispielsweise leicht alle Managed Systems vom Typ 8205 an der HMC *hmc01* auflisten:

```
$ ms -h hmc03 show 8205
NAME  SERIAL_NUM  TYPE_MODEL  HMCS
ms08  XXXXXXX     8205-E6B    hmc03,hmc04
ms09  XXXXXXX     8205-E6B    hmc03,hmc04
$
```

Bei der Option selbst können keine Wildcards verwendet werden, die HMC muß entweder in Form des Namens oder der Seriennummer angegeben werden!

8. Auswahl von LPARs

Auch LPARs können entweder über ihren Namen oder ihre Seriennummer ausgewählt werden:

```
$ lpar show aix01
NAME   ID  SERIAL    LPAR_ENV  MS    HMCS
aix01  3   01234567  aixlinux  ms05  hmc01,hmc02
$ lpar show 01234567
NAME   ID  SERIAL    LPAR_ENV  MS    HMCS
aix01  3   01234567  aixlinux  ms05  hmc01,hmc02
$
```

Auch hier sind in beiden Fällen Wildcards erlaubt:

```
$ lpar show aix*
…
$ lpar show 0123*
…
$
```

Die Auswahl erfolgt aus allen dem LPAR-Tool bekannten LPARs. Über die Option *-h* kann dies auf LPARs eingeschränkt werden, die über eine bestimmte HMC angebunden sind. Über die Option *-m* kann dies auf LPARs eingeschränkt werden, die auf einem bestimmten Managed System laufen. Z.B. alle LPARs mit „*tsm*“ im Namen die auf dem Managed System *ms03* laufen:

```
$ lpar -m ms03 show *tsm*
NAME      ID  SERIAL    LPAR_ENV  MS    HMCS
aixtsm01  3   XXXXXXX3  aixlinux  ms03  hmc01,hmc02
aixtsm02  8   XXXXXXX8  aixlinux  ms03  hmc01,hmc02
$
```

9. Wahl des Ausgabeformats

Bei allen Kommandos die Informationen ausgeben, wurde versucht die Informationen möglichst gut lesbar darzustellen. In der Regel werden dabei einige Informationen nicht ausgegeben, um die Menge der Ausgaben übersichtlich zu halten. Das Format dieser Standard-Ausgabe kann sich von einer Version zur nächsten Version des LPAR-Tools ändern. Sollen die Ausgaben des LPAR-Tools von einem Programm weiterverarbeitet werden, bietet das LPAR-Tool die Möglichkeit an Ausgaben im JSON-Format, Stanza-Format oder YAML-Format auszugeben. Diese Formate sind für Programme leichter zu Parsen. Außerdem werden bei Verwendung dieser Formate standardmäßig alle Informationen auch ausgegeben. Aktuell bieten die Kommandos *hmc*, *ms* und *lpar* die Möglichkeit diese Ausgabe-Formate zu generieren. Für das Kommando *vios* wird dies in der nächsten Version (ab 1.5.X) realisiert werden.

Das Ausgabe-Format kann über die Optionen *-j* (JSON), *-f* (Stanza) oder *-y* (YAML) ausgewählt. Alternativ kann aber auch die Option *-o* mit einem der Argumente *json*, *stanza* oder *yaml* verwendet werden:

```
$ hmc show
NAME   SERIAL_NUM  TYPE_MODEL
hmc01  XXXXXXX     7042-CR8
hmc02  XXXXXXX     7042-CR8
$ hmc show -j
{
     "name": "hmc01",
     "serial_num": "XXXXXXX",
     "type_model": "7042-CR8"
}
{
     "name": "hmc02",
     "serial_num": "XXXXXXX",
     "type_model": "7042-CR8"
}
$ hmc show -f
hmc01:
     name = hmc01
     serial_num = XXXXXXX
     type_model = 7042-CR8
hmc01:
     name = hmc02
     serial_num = XXXXXXX
     type_model = 7042-CR8
$ hmc show -y
---
  name: hmc01
  serial_num: XXXXXXX
  type_model: 7042-CR8
```

```
---
  name: hmc02
  serial_num: XXXXXXX
  type_model: 7042-CR8
$
```

Dies funktioniert für alle anderen Sub-Kommandos die Ausgaben produzieren in gleicher Weise, das Kommando „*hmc show*“ ist hier lediglich ein Beispiel mit übersichtlichen Ausgaben.

10. Auswahl der auszugebenden Datensätze

Das LPAR-Tool bietet die Möglichkeit aus der Vielzahl von Datensätzen vor der Ausgabe diejenigen auszuwählen, die von Interesse sind. Wir demonstrieren dies am Beispiel der Kommandos „*lpar status*“ und „*lpar lsmem*“. Dies geht aber genauso bei allen anderen Kommandos die Ausgaben produzieren (Ausnahme: das Kommando *vios*).

Zunächst zeigen wir die Auswahl der Datensätze anhand eines Vergleiches eines Attributes mit einer vorgegebenen Zeichenkette. Das Kommando „*lpar status*“ liefert den Status von einer oder mehreren LPARs. Wir möchten die Ausgabe beschränken auf LPARs die gerade aktiv sind (*state = Running*). Hierzu kann die Option *-s* verwendet werden:

```
$ lpar status -s state=Running
NAME                LPAR_ID  LPAR_ENV   STATE    PROFILE   SYNC  RMC       PROCS
PROC_UNITS  MEM    OS_VERSION
aixauditdbi01          9        aixlinux   Running  standard  0     active    2
0.2         24576  AIX 7.1 7100-05-02-1810
aixauditdbp01          27       aixlinux   Running  standard  0     active    1
0.4         16384  AIX 7.1 7100-05-02-1810
aixauditdbt01          39       aixlinux   Running  standard  0     active    2
0.4         24576  AIX 7.1 7100-05-02-1810
…
$
```

Um alle LPARs auszuwählen die nicht im Zustand Running sind, kann der Vergleich *!=* verwendet werden:

```
$ lpar status -s state!=Running
NAME     LPAR_ID  LPAR_ENV  STATE          PROFILE   SYNC  RMC       PROCS  PROC_UNITS
MEM   OS_VERSION
aix14    18       aixlinux  Not Activated  -         0     inactive  0      -
0     Unknown
aix15    26       aixlinux  Not Activated  standard  0     inactive  1      0.4
1024  Unknown
lpar12   26       aixlinux  Not Activated  standard  0     inactive  0      0.0
0     Unknown
…
$
```

Es ist auch möglich mehrere Vergleiche durchzuführen. Hierzu wollen wir eine Liste von LPARs generieren, die im Zustand Running sind (*state = Running*), bei denen aber die RMC-Verbindung aktuell nicht funktioniert (*rmc_state != active*):

```
$ lpar status -s state=Running,rmc_state!=active
```

```
NAME       LPAR_ID  LPAR_ENV  STATE    PROFILE   SYNC  RMC       PROCS  PROC_UNITS  MEM
OS_VERSION
tsm01      53       aixlinux  Running  standard  0     inactive  5      1.5         81920
Unknown
tsm02      53       aixlinux  Running  standard  0     inactive  5      1.0         81920
Unknown
…
$
```

Werden Vergleiche mit Komma separiert, dann werden diese intern mit einem logischen UND verbunden, d.h. alle aufgelisteten Vergleiche müssen erfüllt sein. Möchte man mehrere Vergleiche mit logischem ODER kombinieren, dann muss man für jeden Vergleich eine eigene *-s* Option verwenden, also z.B. „*-s state=Running -s rmc_state!=active*“. In diesem Fall werden alle LPARs aufgelistet, die entweder im Zustand *Running* sind, oder eine aktive RMC-Verbindung haben (dies geht aber ohnehin nur wenn die LPAR aktiv ist).

Neben den Zeichenketten Vergleichen unterstützt das LPAR-Tool auch Reguläre Ausdrücke. Wir zeigen dies, indem wir diejenigen LPARs auflisten, die unter AIX 7.1 laufen. Das „*lpar status*“ Kommando gibt die OS-Version über das Attribut *os_version* aus. Die Major-Version wird dabei 4-stellig (z.B. *7100*) ausgegeben. Dies nutzen wir indem wir das Feld *os_version* gegen den regulären Ausdruck *7100* matchen (*os_version ~ 7100*):

```
$ lpar status -s os_version~7100-04
NAME                LPAR_ID  LPAR_ENV   STATE    PROFILE   SYNC  RMC       PROCS
PROC_UNITS  MEM    OS_VERSION
aixauditdbi01       9        aixlinux   Running  standard  0     active    2
0.2         24576  AIX 7.1 7100-05-02-1810
aixauditdbp01       27       aixlinux   Running  standard  0     active    1
0.4         16384  AIX 7.1 7100-05-02-1810
…
$
```

Oder hier eine Variante mit der alle LPARs aufgelistet werden, die nicht unter AIX 7.1 laufen:

```
$ lpar status -s os_version\!~7100-05
NAME          LPAR_ID  LPAR_ENV   STATE    PROFILE   SYNC  RMC       PROCS  PROC_UNITS
MEM    OS_VERSION
aix01         19       aixlinux   Running  standard  0     active    3      0.9
81920  AIX 7.1 7100-04-04-1717
aix02         19       aixlinux   Running  standard  0     active    3      0.9
81920  AIX 7.1 7100-04-04-1717
…
ms02-vio1     1        vioserver  Running  standard  0     active    6      2.4
6144   VIOS 2.2.5.10
ms02-vio2     2        vioserver  Running  standard  0     active    6      1.8
6144   VIOS 2.2.5.10
$
```

Leider werden jetzt auch die Virtual-I/O-Server mit aufgelistet. Dies lässt sich aber leicht verhindern, indem man zusätzlich verlangt das nur LPARs vom Typ *aixlinux* aufgelistet werden sollen:

```
$ lpar status -s lpar_env=aixlinux,os_version\!~7100-05
NAME          LPAR_ID  LPAR_ENV   STATE    PROFILE   SYNC  RMC       PROCS  PROC_UNITS
MEM    OS_VERSION
aix01         19       aixlinux   Running  standard  0     active    3      0.9
81920  AIX 7.1 7100-04-04-1717
```

```
aix02          19       aixlinux   Running  standard  0     active    3      0.9
81920  AIX 7.1 7100-04-04-1717
…
$
```

Neben Vergleichen mit Zeichenketten oder Regulären Ausdrücken, können auch numerische Vergleiche durchgeführt werden. Als Beispiel listen wir zunächst alle LPARs auf, die mehr als 32 GB (32768 MB) RAM besitzen:

```
$ lpar lsmem -s curr_mem:gt:32768
               MEMORY           MEMORY             HUGE_PAGES
LPAR_NAME     MODE   AME   MIN    CURR    MAX      MIN   CURR   MAX
tsm01         ded    1.0   1024   81920   163840   0     0      0
tsm02         ded    1.0   1024   81920   163840   0     0      0
…
$
```

Die möglichen numerischen Vergleiche sind dabei:

- *:gt:* - greater than
- *:ge:* - greater or equal
- *:eq:* - equal
- *:ne:* - not equal
- *:le:* - less or equal
- *:lt:* - less than

Natürlich können alle Vergleiche auch miteinander kombiniert werden!

11. Auswahl der auszugebenden Datenfelder

Eine weitere Möglichkeit des LPAR-Tools die Ausgabe individuell zu gestalten besteht in der Auswahl der gewünschten Felder. Bei einigen Ausgabe-Kommandos bekommt das LPAR-Tool von der/den HMC(s) Datensätze mit 50 oder mehr Feldern. Beim Standard-Ausgabeformat können aber lediglich einige ausgewählte Felder ausgegeben werden, damit die Ausgabe für den Benutzer noch lesbar bleibt. Natürlich kann es dabei passieren das interessante Felder gar nicht ausgegeben werden. Mittels der Option *-F* können die auszugebenden Felder ausgewählt werden. Es können beliebige Trennzeichen verwendet werden:

```
$ lpar lsmem -F lpar_name:curr_mem
aix01:0
aix02:1024
aix03:0
…
$
```

Im Beispiel wird nur der LPAR-Name und die aktuelle Hauptspeicher-Größe (getrennt durch einen Doppelpunkt) ausgegeben. Es kann aber auch ein beliebiges anderes Trennzeichen verwendet werden, hier z.B. ein Leerzeichen:

```
$ lpar lsmem -F „lpar_name curr_mem“
aix01 0
aix02 1024
aix03 0
…
$
```

Dies lässt sich natürlich mit allen verfügbaren Ausgabe Formaten und den oben angesprochenen Selektionen kombinieren:

```
$ lpar lsmem -s curr_mem:lt:4096 -y -F lpar_name:curr_mem
---
  curr_mem: 0
  lpar_name: aix01
---
  curr_mem: 1024
  lpar_name: aix02
---
  curr_mem: 0
  lpar_name: aix03
…
$
```

3. Administrieren von LPARs

Zunächst zeigen wir einige einfachere, aber häufig wiederkehrende, Operationen auf LPARs mit Hilfe des LPAR-Tools:

- Status einer LPAR
- Konfiguration einer LPAR
- Aktivieren einer LPAR
- Runterfahren einer LPAR
- Konsole für eine LPAR öffnen

1. Status einer LPAR

Für die Anzeige des Status einer LPAR gibt es das Kommando „*lpar status*“. Dieses kann auch den Status von mehreren oder sogar allen LPARs mit einem Kommando auflisten.

Status einer LPAR auflisten:

```
$ lpar status lpar1
NAME   LPAR_ID  LPAR_ENV  STATE    PROFILE   SYNC  RMC     PROCS  PROC_UNITS  MEM
OS_VERSION
lpar1  3        aixlinux  Running  standard  0     active  4      0.4         33792  AIX
7.1 7100-04-05-1720
$
```

Status mehrerer LPARs auflisten:

```
$ lpar status lpar1 lpar2 lpar3
NAME   LPAR_ID  LPAR_ENV  STATE    PROFILE   SYNC  RMC     PROCS  PROC_UNITS  MEM
OS_VERSION
lpar1  3        aixlinux  Running  standard  0     active  4      0.4         33792  AIX
7.1 7100-04-05-1720
lpar2  16       aixlinux  Running  standard  0     active  1      0.2         16384  AIX
7.1 7100-04-05-1720
lpar3  13       aixlinux  Running  standard  0     active  2      0.3         32768  AIX
7.1 7100-04-05-1720
$
```

Status aller LPARs eines Managed Systems auflisten:

```
$ lpar -m ms01 status
NAME   LPAR_ID  LPAR_ENV  STATE          PROFILE   SYNC  RMC       PROCS  PROC_UNITS
MEM    OS_VERSION
lpar1  3        aixlinux  Running        standard  0     active    4      0.4
33792  AIX 7.1 7100-04-05-1720
lpar4  7        aixlinux  Not Activated  standard  0     inactive  4      0.4
33792  AIX 7.1 7100-04-05-1720
…
```

```
$
```

Status aller LPARs anzeigen:

```
$ lpar status
NAME   LPAR_ID  LPAR_ENV  STATE          PROFILE   SYNC  RMC       PROCS  PROC_UNITS
MEM    OS_VERSION
lpar1  3        aixlinux  Running        standard  0     active    4      0.4
33792  AIX 7.1 7100-04-05-1720
lpar2  16       aixlinux  Running        standard  0     active    1      0.2
16384  AIX 7.1 7100-04-05-1720
lpar3  13       aixlinux  Running        standard  0     active    2      0.3
32768  AIX 7.1 7100-04-05-1720
lpar4  7        aixlinux  Not Activated  standard  0     inactive  4      0.4
33792  AIX 7.1 7100-04-05-1720
…
$
```

Neben dem Status der LPAR wird auch der Status der RMC-Verbindung und andere Informationen angezeigt.

2. Eigenschaften einer LPAR

Die aktuellen Eigenschaften (Attribute) einer LPAR lassen sich am einfachsten mit dem Kommando „*lpar lsattr*“ anzeigen:

```
$ lpar lsattr lpar1
NAME   DEFAULT_PROFILE  ALLOW_PERF_COLLECTION  TIME_REF  SUSPEND_CAPABLE
REMOTE_RESTART_CAPABLE  SIMPLIFIED_REMOTE_RESTART_CAPABLE
lpar1  standard         0                      0         0
0                       -
$
```

Einige Eigenschaften werden nicht über Profile festgelegt, sondern gelten für eine LPAR unabhängig vom verwendeten Profil. Beispiele solcher Eigenschaften sind der LPAR-Name, der Name des Default-Profils oder die Eigenschaft *remote_restart_capable*. Die Eigenschaften lassen sich über das Kommando „*lpar chattr*“ ändern, hier gezeigt am Beispiel der Eigenschaft *new_name*:

```
$ lpar chattr lpar1 new_name=lpar100
$
```

Hiermit wurde die LPAR umbenannt. Allerdings ist die Namensänderung nicht in den lokalen Zuordnungsdateien erfolgt, daher ist die LPAR noch unter dem alten Namen bekannt, kann aber über diesen nicht mehr angesprochen werden:

```
$ lpar lsmem lpar1
            MEMORY         MEMORY         HUGE_PAGES
LPAR_NAME   MODE   AME   MIN   CURR   MAX   MIN   CURR   MAX
$
```

Ein „*hmc rescan*“ aktualisiert die Zuordnungsdateien wieder und anschließend kann die umbenannte LPAR auch über Ihren neuen Namen angesprochen werden.

Zum Umbenennen einer LPAR gibt es aber den einfacheren Weg über das Kommando „*lpar rename*“, welches auch die Zuordnungsdateien aktualisiert:

```
$ lpar rename lpar100 lpar1
$
```

Als weiteres Beispiel zeigen wir die Änderung der Eigenschaft *sync_curr_profile*, diese gibt an, ob die aktuelle Konfiguration einer LPAR automatisch mit dem aktuellen aktiven Profil synchronisiert werden soll. Ist dies aktiviert, dann werden Änderungen an der LPAR automatisch auch im Profil durchgeführt. Aktuelle Konfiguration und Profil sind dann immer synchron.

```
$ lpar chattr lpar1 sync_curr_profile=1
$
```

Gültige Werte für die Property *sync_curr_profile* sind:

```
0 - Synchronisierung deaktivieren
1 - Synchronisierung aktivieren
2 - Synchronisierung vorübergehend aussetzen (suspend), bis das nächste Mal ein Profil
aktiviert oder angewendet wird
```

Ist die Synchronisierung aktiviert, kann das aktuell aktive Profil nicht mehr geändert werden:

```
$ lpar -p standard chmem lpar1 mem_expansion=1.4
hmc01: chsyscfg -r prof -m ms09 -i 'lpar_name=lpar1,name=standard,mem_expansion=1.4'
ERROR: remote HMC command returned an error (1)
StdErr: An error occurred while changing the partition profile named standard.
StdErr: This profile is synchronized with the partition's current configuration.  To update
this profile, you can specify the force option on this command, or you can turn off current
profile synchronization for the partition.  If you specify the force option on this
command, this profile will be updated and synchronization of this profile will be suspended
until the next time this profile is activated or applied.
$
```

3. Aktivieren einer LPAR

Ist eine LPAR im Zustand „*Not Activated*“ kann sie mit dem Kommando „*lpar activate*“ wieder aktiviert werden. Im einfachsten Fall geht dies wie folgt:

```
$ lpar activate lpar1
$
```

Die LPAR wird mit der letzten aktuellen Konfiguration aktiviert. Je nach Konfiguration des Profils wird die LPAR automatisch gebootet. Welches das aktuelle Profil ist, lässt sich mit Hilfe von „*lpar status*“ ermitteln:

```
$ lpar status lpar1
```

```
NAME   LPAR_ID  LPAR_ENV  STATE          PROFILE   SYNC  RMC       PROCS  PROC_UNITS  MEM
OS_VERSION
lpar1  4        aixlinux  Not Activated  standard  1     inactive  1      -           2048
Unknown
$
```

Beim ersten Aktivieren einer LPAR muss zwingend ein Profil angegeben werden, die LPAR besitzt dann noch keine letzte aktuelle Konfiguration:

```
$ lpar -m ms09 create
.
    > lpar6
$ lpar status lpar6
NAME   LPAR_ID  LPAR_ENV  STATE          PROFILE   SYNC  RMC       PROCS  PROC_UNITS  MEM
OS_VERSION
lpar6  7        aixlinux  Not Activated  -         1     inactive  1      -           2048
Unknown
$
```

Versucht man die LPAR trotzdem, ohne Angabe eines Profils, zu starten, erhält man die folgende Fehlermeldung:

```
$ lpar activate lpar6
ERROR: lparActivate(): remote HMC command returned an error (1)
CMD on hmc01: chsysstate -m ms09 -r lpar -o on -n lpar6
StdErr: HSCL3680 Partition lpar6 cannot be activated due to insufficient resources in its
current configuration.  Please activate the partition with a profile.
$
```

Die verfügbaren Profile einer LPAR können mit dem Kommando „*lpar lsprof*“ angezeigt werden:

```
$ lpar lsprof lpar6
NAME             PROFILES
lpar6            standard
$
```

Wir aktivieren die obige LPAR jetzt mit dem Profil ‚*standard*‘:

```
$ lpar activate -p standard lpar6
$
```

Jetzt ist auch ein aktuelles Profil für die LPAR hinterlegt :

```
$ lpar status lpar6
NAME   LPAR_ID  LPAR_ENV  STATE    PROFILE   SYNC  RMC       PROCS  PROC_UNITS  MEM
OS_VERSION
lpar6  7        aixlinux  Running  standard  1     inactive  1      -           2048
Unknown
$
```

Soll die LPAR nicht automatisch gebootet werden, so kann der gewünschte Bootmode auch angegeben werden:

```
$ lpar activate -b sms lpar6
$
```

Die LPAR wird damit aktiviert und bootet ins SMS-Menü. Alternativ kann auch „*norm*“ für normalen Bootmode oder „*of*“ für OpenFirmware angegeben werden.

Relativ häufig wird man nach dem Aktivieren einer LPAR im Anschluß eine Konsole öffnen. Durch Benutzung der Option „*-c*“ kann „*lpar activate*“ die Konsole gleich mitöffnen, das Kommando „*lpar console*“ muss dann nicht extra gestartet werden:

```
$ lpar activate -c lpar6

 Open in progress

 Open completed.

IBM IBM IBM IBM IBM IBM IBM IBM IBM IBM IBM IBM IBM IBM IBM IBM
…
```

Weitere Möglichkeiten können über die Hilfe angezeigt werden:

```
$ lpar help activate
USAGE:
   lpar [-h <hmc>] [-m <ms>] [-p <profile>] activate [-b <bootmode>] [-c] [-k <keylock>] [-
v] <lpar>

   -b : the boot mode when activating an AIX, Linux, or Virtual I/O Server partition
     norm - normal boot
     dd - diagnostic with default boot list
     ds - diagnostic with stored boot list
     of - Open Firmware
     sms - System Management Services
   -c : open a console session
   -k : the keylock position to set
     manual or norm
   -p : the partition profile to use for activation

DESCRIPTION

Activates a AIX, Linux, IBM i or Virtual I/O Server partition. If no profile is specified
(Option -p), the partition is activated with its current configuration. In this case only
the boot mode can be specified.
Since in a lot of cases, after activating a partition, a console session is needed, this
can be achieved by using the option '-c', no separate command is needed in this case.

EXAMPLES:

Activate LPAR aix02 with its current configuration:
   lpar activate aix02

Activate LPAR aix02 with its current configuration and open a console session:
   lpar activate -c aix02

Activate LPAR aix02 in SMS mode using the partition profile 'standard':
   lpar -p standard activate -b sms aix02

$
```

4. Runterfahren einer LPAR

Eine LPAR sollte normalerweise vom Betriebssystem der LPAR aus heruntergefahren werden, im Falle von AIX z.B. mit dem Kommando *shutdown*. Alternativ kann dies aber auch mit dem LPAR-Tool durchgeführt werden:

```
$ lpar osshutdown lpar1
$
```

Voraussetzung für das Kommando „*lpar osshutdown*“ ist eine funktionierende RMC-Verbindung. Das Kommando initiiert einen regulären Shutdown des Betriebssystems. Das Kommando unterstützt, unter anderem, die beiden Optionen „*-i*“ und „*-r*“, welche einen Einfluß auf den Ablauf des Betriebssystem-Shutdowns haben.

Handelt es sich um eine AIX-LPAR, so werden die folgenden Kommandos ausgeführt:

-i (immediate)	-r (restart)	Ausgeführtes OS Kommando
nein	nein	shutdown
nein	ja	shutdown -r
ja	nein	shutdown -F
ja	ja	shutdown -F -r

Handelt es sich um eine Linux-LPAR, dann werden die folgenden Kommandos ausgeführt:

-i (immediate)	-r (restart)	Ausgeführtes OS Kommando
nein	nein	shutdown -h +1
nein	ja	shutdown -r +1
ja	nein	shutdown -h now
ja	ja	shutdown -r now

Bei einem Virtual-I/O-Server werden die folgenden Kommandos ausgeführt:

-i (immediate)	-r (restart)	Ausgeführtes OS Kommando
nein	nein	shutdown
nein	ja	shutdown -restart
ja	nein	shutdown -force
ja	ja	shutdown -force -restart

Sollte die LPAR oder das Betriebssystem hängen, ist das Kommando „*lpar osshutdown*“ in der Regel nicht erfolgreich. Es kann dann noch versucht werden zusätzlich die Option „*-f*“ (force) zu verwenden, um ein Herunterfahren zu erzwingen, in vielen Fällen bleibt dies aber ohne Erfolg.

In einem solchen Fall, oder falls das Betriebssystem gar nicht aktiv ist (z.B. *SMS*-Mode), kann das Kommando „*lpar shutdown*“ verwendet werden. Hier ist keine RMC-Verbindung notwendig. Es wird dann ein sogenannter Delayed Shutdown durchgeführt, d.h. der LPAR wird über den Hypervisor ein Shutdown signalisiert (im Falle von AIX wird ein *SIGPWR* Signal gesendet, woraufhin AIX einen Shutdown durchführt), die LPAR bekommt dann noch Zeit ausstehende I/Os abzuschließen und wird dann schließlich ausgeschaltet.

Auch hier werden die Option „*-i*“ und „*-r*“ unterstützt und haben die folgende Bedeutung:

-i (immediate)	-r (restart)	Bedeutung
nein	nein	Ein Delayed Shutdown wird durchgeführt.
nein	ja	Ein System-Dump mit Restart wird initiiert (Operator Panel Funktion 22). Äquivalent dazu ist „lpar dumprestart“.
ja	nein	Ein sofortiger Shutdown wird ausgelöst (Operator Panel Funktion 8).
ja	ja	Ein sofortiger Restart wird ausgeführt (Operator Panel Funktion 3).

Nach Möglichkeit sollte bei einem laufenden Betriebssystem immer ein regulärer Shutdown durchgeführt werden („*lpar osshutdown*“ oder direkt über das Betriebssystem).

5. Initiieren eines System-Dumps

In manchen Situationen ist ein reguläres Herunterfahren des Betriebssystems nicht mehr möglich, z.B. weil das Betriebssystem aufgrund eines Fehlers hängt und nicht mehr, oder nicht mehr korrekt reagiert. Möchte man für ein solches System einen Call bei IBM aufmachen, dann empfiehlt es sich das hängende System nicht einfach auszuschalten, sondern einen System-Dump zu erzeugen. Der System-Dump kann dann IBM zur Verfügung gestellt werden und kann helfen die Ursache für das Fehlverhalten oder Hängen herauszufinden. Ohne einen solchen Dump ist es für den IBM Support äußerst schwierig die Fehlerursache zu ermitteln!

Ein System-Dump kann mit Hilfe des LPAR-Tools sehr einfach initiiert werden, Kommando „*lpar dumprestart*“:

```
$ lpar dumprestart lpar01
$
```

Ein System-Dump kann bei einem größeren System mit viel Hauptspeicher eine Weile in Anspruch nehmen. Den Fortschritt des Dumps kann man mit Hilfe des Kommandos „*lpar lsrefcode*“ überwachen:

```
$ 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  -
…
$
```

Außerdem empfiehlt es sich eine Konsolen-Sitzung zu öffnen, um den anschließenden Boot-Vorgang auf etwaige Fehler zu überwachen.

6. Konsole für eine LPAR

Eine häufig genutzte Funktion des LPAR-Tools dürfte die Möglichkeit sein, jederzeit eine Konsole für eine LPAR zu starten:

```
$ lpar console lpar1

 Open in progress

 Open completed.

 PowerPC Firmware
 Version AL720_121
 SMS 1.7 © Copyright IBM Corp. 2000,2008 All rights reserved.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
 Main Menu
 1.   Select Language
 2.   Setup Remote IPL (Initial Program Load)
 3.   Change SCSI Settings
 4.   Select Console
 5.   Select Boot Options

…
```

Natürlich kann für eine LPAR nur eine Konsole gleichzeitig offen sein. Sollte schon eine Konsole geöffnet sein, bekommt man die folgende Fehlermeldung:

```
$ lpar console lpar1

 A terminal session is already open for this partition.
 Only one open session is allowed for a partition.
 Exiting….    Received end of file, Exiting.
               Shared connection to hmc01 closed.
$
```

Durch Verwenden der Option „*-f*“ (*force*) kann eine Konsole erzwungen werden, die schon geöffnete Konsole wird dabei beendet:

```
$ lpar console -f lpar1


 Open in progress

 Open completed.

…
```

Soll die Konsole geschlossen werden, dann muss die Escape-Sequence „~.“ verwendet werden:

```
 …
~.

 Terminate session? [y/n] y
```

```
Shared connection to hmc01 closed.
$
```

Soll nur eine hängende Konsolensitzung beendet werden, kann dies mit dem Kommando „lpar rmconsole“ gemacht werden:

```
$ lpar rmconsole lpar3
/bin/stty: standard input: Inappropriate ioctl for device
$
```

In der Konsolensitzung die beendet wird, erscheint die folgende Meldung:

```
Connection has closed


This session is no longer connected. Please close this window.
```

Ob es offene Konsolen gibt, kann indirekt überprüft werden. Hierzu kann man mit dem Kommando „*hmc lslogon*“ auf den angeschlossenen HMCs die laufenden Sessions auflisten:

```
$ hmc lslogon hmc01
USER_NAME   TTY_ID  LOGON_TIME        ACCESS_LOCATION
   TASK_NAME  TTY_ID  START_TIME  USER_NAME  PID
-          -       -                -
$
$ hmc lslogon hmc02
USER_NAME   TTY_ID  LOGON_TIME        ACCESS_LOCATION
   TASK_NAME  TTY_ID  START_TIME           USER_NAME  PID
user01      pts/1   2018-10-10 09:38  172.20.132.167
   mkvterm    pts/1   Oct 10 09:38:18 2018  root       24583
$
```

Eine Konsolensitzung ist über den Task-Namen *mkvterm* zu erkennen.

7. Live Partition Mobility (LPM)

Hier zeigen wir anhand von einer LPAR (*lpar2*), wie mit Hilfe des LPAR-Tools LPM durchgeführt werden kann.

Zunächst überprüfen wir den Status der LPAR und den Status der RMC-Verbindung. Das geht am einfachsten mit dem Kommando „*lpar status*“:

```
$ lpar status lpar2
NAME   LPAR_ID  LPAR_ENV  STATE    PROFILE   SYNC  RMC     PROCS  PROC_UNITS  MEM
OS_VERSION
lpar2  39       aixlinux  Running  standard  0     active  1      0.1         4096  AIX 7.2
7200-03-02-1846
$
```

Die LPAR ist im Status *Running* und die RMC-Verbindung ist *active*. Die LPAR ist mit einem Prozessor Core und *0.1* Prozessor Units aktiv, sie hat außerdem 4 GB Hauptspeicher.

Als nächstes überprüfen wir noch den Prozessor Compatibility Mode, hierzu müssen alle Attribute der LPAR aufgelistet werden:

```
$ lpar lsattr -f lpar2
lpar2:
            affinity_group_id = none
            allow_perf_collection = 0
            auto_start = 0
            boot_mode = norm
            curr_lpar_proc_compat_mode = POWER7
...
$
```

Die LPAR läuft aktuell im POWER7 Mode, d.h. die Zielplattform muß mindestens POWER7 unterstützen. Als letztes schauen wir uns noch an welche VLAN und VSwitches die LPAR verwendet:

```
$ lpar lsvslot lpar2
SLOT  REQ  ADAPTER_TYPE   STATE  DATA
0     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
1     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
5     No   eth            1      PVID=1234 VLANS= ETHERNET0 XXXXXXXXXXXX
10    No   fc/client      1      remote: ms01-vio1(1)/103 c050760XXXXX0052,c050760XXXXX0053
20    No   fc/client      1      remote: ms01-vio2(2)/203 c050760XXXXX0056,c050760XXXXX0057
$
```

Es wird das VLAN *1234* am VSwitch *ETHERNET0* verwendet.

Nun überprüfen wir das Ziel-Managed System *ms03*. Es handelt sich dabei um eine S824, also ein POWER8 System. Die verfügbaren Prozessor Units lassen sich mit „*ms lsproc*“ anzeigen:

```
$ ms lsproc ms03
NAME  INSTALLED  CONFIGURABLE  AVAIL  MAX_SHARED_PROC_POOLS
ms03  20.0       20.0          6.5    64
$
```

Der verfügbare RAM lässt sich mit „*ms lsmem*“ anzeigen:

```
$ ms lsmem ms03
NAME  INSTALLED  FIRMWARE  CONFIGURABLE  AVAIL   MEM_REGION_SIZE
ms03  1048576    17920     1048576       647680  256
$
```

Es sind also noch ausreichend Resourcen verfügbar.

Auch das VLAN *1234* am VSwitch *ETHERNET0* ist verfügbar, wie das folgende Kommando zeigt:

```
$ ms lsvswitch ms03
MS          VSWITCH           SWITCH_MODE  VLAN_IDS
ms03        ETHBLB            VEB          10,12,14
ms03        ETHERNET0(Default)  VEB        20,21,22,1234
```

```
$
```

Ein Verschieben der LPAR sollte also möglich sein.

Die manuellen Überprüfungen die wir oben durchgeführt haben, werden aber auch beim Verschieben der LPAR von der HMC durchgeführt. Man kann auch einige Zeit for dem Verschieben der LPAR eine sogenannte Validierung durchführen, dabei wird überprüft ob alle Voraussetzungen für eine Verschiebung erfüllt sind, ohne die LPAR wirklich zu verschieben.

Wir führen jetzt eine solche Validierung mit Hilfe des LPAR-Tools durch:

```
$ lpar validate lpar2 ms03

…

Warnings:
HSCLA4CD The management console cannot maintain the source Virtual I/O Server (VIOS) slot
number 203 for virtual fibre channel adapter 20 on the destination VIOS partition 1*MMMM-
TTT*SSSSSSS.
Shared connection to hmc01 closed.
$
```

Der Exit Status der Validierung ist trotz all der Meldungen *0*:

```
$ echo $?
0
$
```

Das bedeutet das sich die LPAR verschieben läßt. Wir führen nun die Verschiebung durch:

```
$ lpar migrate lpar2 ms03

…

Warnings:
HSCLA4CD The management console cannot maintain the source Virtual I/O Server (VIOS) slot
number 203 for virtual fibre channel adapter 20 on the destination VIOS partition 1*MMMM-
TTT*SSSSSSS.
Shared connection to hmc01 closed.
$
```

Es wurden lediglich Warnungen ausgegeben, d.h. die Verschiebung war erfolgreich. Wir schauen uns kurz an wo die LPAR nun gerade läuft:

```
$ lpar show lpar2
NAME   ID  SERIAL     LPAR_ENV  MS    HMCS
lpar2  40  XXXXXXXXX  aixlinux  ms03  hmc01,hmc02
$
```

Dies bestätigt das die Verschiebung erfolgreich war.

Man kann auch eine inaktive LPAR auf ein anderes Managed System verschieben. Dies probieren wir jetzt für die LPAR *lpar3* aus:

```
$ lpar status lpar3
NAME   LPAR_ID  LPAR_ENV  STATE          PROFILE   SYNC  RMC       PROCS  PROC_UNITS  MEM
OS_VERSION
lpar3  7        aixlinux  Not Activated  standard  0     inactive  1      -           2048
Unknown
$
```

Die LPAR verwendet ebenfalls das VLAN *1234*. Dieses Mal machen wir keine Validierung, sondern führen die Veschiebung sofort durch:

```
 $ lpar migrate lpar3 ms05

...

Warnings:
HSCLA295 As part of the migration process, the management console will create a new
migration profile containing the partition's current state.  The default is to use the
current profile, which will replace the existing definition of this profile.  While this
works for most scenarios, other options are possible.  You may specify a different existing
profile, which would be replaced with the current partition definition, or you may specify
a new profile to save the current partition state.
HSCLB505 The partition cannot use hardware-accelerated encryption on the destination
managed system because the destination managed system does not support hardware-accelerated
encryption.
HSCLB504 The migrating partition cannot use hardware-accelerated Active Memory Expansion on
the destination managed system because the destination managed system does not support
hardware-accelerated Active Memory Expansion.
HSCLA4CD The management console cannot maintain the source Virtual I/O Server (VIOS) slot
number 44 for virtual fibre channel adapter 10 on the destination VIOS partition 1*MMMM-
TTT*SSSSSSS.
Shared connection to hmc01 closed.
$
```

Die LPAR wurde nach ms05 verschoben:

```
$ lpar show lpar3
NAME   ID  SERIAL     LPAR_ENV  MS    HMCS
lpar3  39  XXXXXXXXX  aixlinux  ms05  hmc01,hmc02
$
```

4. Erzeugen von LPARs

Mit dem LPAR-Tool können auf sehr einfache Weise neue LPARs angelegt werden.

1. Erzeugen einer neuen LPAR

Eine neue LPAR kann mit dem Kommando „*lpar create*“ erzeugt werden. Es muss das Managed System angegeben werden, auf welchem die LPAR angelegt werden soll:

```
$ lpar -m ms01 create
.
    > lpar1
$
```

Die LPAR wird ohne physikalische und virtuelle Adapter angelegt. Da kein Profil-Name angegeben wurde, wird der Default „*standard*“ verwendet. Als Name der LPAR wird *lparN* verwendet, wobei *N* von *1* beginnend hoch gezählt wird. Profil-Name und/oder Name der LPAR können aber natürlich auch angegeben werden:

```
$ lpar -m ms01 -p myprofile create mylpar01
.
    > mylpar01
$
```

Der Status der neu angelegten LPAR kann mit dem Kommando „*lpar status*“ angezeigt werden:

```
$ lpar status lpar1
NAME   LPAR_ID  LPAR_ENV  STATE          PROFILE  SYNC  RMC       PROCS  PROC_UNITS  MEM
OS_VERSION
lpar1  4        aixlinux  Not Activated  -        0     inactive  0      -           0
Unknown
$
```

Die neue LPAR hat den Typ *aixlinux* und ist nicht aktiviert. Detaillierte Informationen zu einer LPAR können mit dem Kommando „*lpar display*“ angezeigt werden.

```
$ lpar display lpar1
NAME                          : lpar1
LPAR_ID                       : 4
LPAR_ENV                      : aixlinux
SERIAL                        : XXXXXXXXX
MS                            : ms01
HMCS                          : hmc01,hmc02
STATE                         : Not Activated
RESOURCE_CONFIG               : 0
OS_VERSION                    : Unknown
PROC_COMPAT_MODE              : desired=default,curr=POWER7
PROC_MODE                     : curr=ded,pend=ded
SHARED_PROC_POOL              : curr=-,pend=-
SHARING_MODE                  : curr=share_idle_procs,pend=share_idle_procs
UNCAP_WEIGHT                  : curr=-,pend=-
PROCS                         : min=0,desired=0,max=0
PROC_UNITS                    : min=-,desired=-,max=-
MEM_MODE                      : ded
MEM_EXPANSION                 : curr=0.0,pend=0.0
```

```
HPT_RATIO                       : curr=1:64,pend=-
MEMORY                          : min=0,desired=0,max=0
HUGE_PAGES                      : min=0,desired=0,max=0
PROFILE                         : default=standard,curr=
SYNC_CURR_PROFILE               : 0
RMC_STATE                       : inactive
RMC_IPADDR                      :
ALLOW_PERF_COLLECTION           : 0
AFFINITY_GROUP_ID               : none
AUTO_START                      : 0
BOOT_MODE                       : norm
LPAR_AVAIL_PRIORITY             : 127
LPAR_KEYLOCK                    : norm
REDUNDANT_ERR_PATH_REPORTING    : 0
TIME_REF                        : 0
VTPM_ENABLED                    : 0
WORK_GROUP_ID                   : none
POWER_CTRL_LPAR_IDS             : none
SUSPEND_CAPABLE                 : 0
REMOTE_RESTART_CAPABLE          : 0
$
```

Die Ausgabe zeigt das die LPAR aktuell nicht aktiviert ist (*state=Not Activated*) und aktuell auch keine Resourcen belegt (*resource_config=0*). Die gewünschte Konfiguration für eine LPAR wird in einem sogenannten Profil abgelegt. Das Profil bestimmt unter anderem die Prozessor-, Speicher- und I/O-Konfiguration einer LPAR.

Das Profil der neu angelegten LPAR kann mit den folgenden Kommandos angezeigt werden:

- Allgemeine Eigenschaften, Prozessor- und Speicher-Konfiguration: *lpar –p <profile> display*

- Prozessor Konfiguration: *lpar –p <profile> lsproc*

- Speicher Konfiguration: *lpar –p <profile> lsmem*

- Virtuelles I/O: *lpar –p <profile> lsvslot*

- Physikalisches I/O: *lpar –p <profile> lsslot*

Einen detaillierten Überblick eines Profils, mit einer Vielzahl von Informationen, erhält man mit dem Kommando „*lpar display*“. Das gewünschte Profil kann mit der Option „*-p*“ angegeben werden:

```
$ lpar -p standard display lpar1
NAME                            : standard
LPAR_NAME                       : lpar1
LPAR_ID                         : 4
LPAR_ENV                        : aixlinux
ALL_RESOURCES                   : 0
AFFINITY_GROUP_ID               : none
MAX_VIRTUAL_SLOTS               : 6
LPAR_IO_POOL_IDS                : none
PROC_COMPAT_MODE                : default
PROC_MODE                       : ded
SHARED_PROC_POOL                : name=-,id=-
SHARING_MODE                    : share_idle_procs
UNCAP_WEIGHT                    : -
PROCS                           : min=1,desired=1,max=1
PROC_UNITS                      : min=-,desired=-,max=-
MEM_MODE                        : ded
```

```
MEM_EXPANSION                  : 0.0
HPT_RATIO                      : 1:64
MEMORY                         : min=1024,desired=2048,max=8192
HUGE_PAGES                     : min=null,desired=null,max=null
AUTO_START                     : 0
BOOT_MODE                      : norm
CONN_MONITORING                : 1
REDUNDANT_ERR_PATH_REPORTING   : 0
BSR_ARRAYS                     : 0
WORK_GROUP_ID                  : none
POWER_CTRL_LPAR_IDS            : none
ELECTRONIC_ERR_REPORTING       : null
$
```

Ist man nur an der Prozessor-Konfiguration interessiert, dann kann das Subkommando „*lpar lsproc*“ verwendet werden:

```
$ lpar -p standard lsproc lpar1
            PROC          PROCS            PROC_UNITS                          UNCAP   PROC
LPAR_NAME   MODE   MIN   DESIRED   MAX   MIN   DESIRED   MAX   SHARING_MODE      WEIGHT  POOL
lpar1       ded    1     1         1     -     -         -     share_idle_procs  -       -
$
```

Das Subkommando „*lpar lsproc*“ erlaubt die Angabe beliebig vieler LPARs, z.B. alle LPARs die an eine bestimmte HMC angebunden sind:

```
$ lpar -p standard -m ms01 lsproc
                  PROC          PROCS             PROC_UNITS
UNCAP      PROC
LPAR_NAME         MODE     MIN   DESIRED   MAX   MIN   DESIRED   MAX   SHARING_MODE      WEIGHT
POOL
lpar1             ded      1     1         1     -     -         -     share_idle_procs  -       -
mylpar01          ded      1     1         1     -     -         -     share_idle_procs  -       -
aix01             shared   1     1         10    0.1   0.1       1.0   uncap             5
DefaultPool
aix02             shared   1     1         40    0.1   0.1       40.0  uncap             50
DefaultPool
ms01-vio1         shared   2     2         4     0.4   1.2       4.0   uncap             255
DefaultPool
ms01-vio2         shared   2     2         4     0.4   1.2       4.0   uncap             255
DefaultPool
$
```

Analog kann man sich die Speicher-Konfiguration mit dem Subkommando „*lpar lsmem*“ anschauen, auch hier können mehrere LPARs gleichzeitig angeschaut werden:

```
$ lpar -p standard lsmem lpar1
             MEMORY           MEMORY              HUGE_PAGES
LPAR_NAME   MODE   AME   MIN    DESIRED   MAX     MIN    DESIRED   MAX
lpar1       ded    0.0   1024   2048      8192    null   null      null
$
```

Ist eine LPAR aktiv und man möchte die Werte der aktiven LPAR sehen, so lässt man einfach die Profil-Angabe weg. Die Kommandos beziehen sich dann auf die aktive Konfiguration:

```
$ lpar lsmem lpar1
            MEMORY         MEMORY         HUGE_PAGES
LPAR_NAME   MODE   AME   MIN   CURR   MAX   MIN   CURR   MAX
lpar1       ded    0.0   0     0      0     0     0      0
$
```

Da die eben neu angelegte LPAR *lpar1* nicht aktiv ist, benötigt sie auch aktuell keinen Speicher. Für LPARs die gerade aktiv sind, sieht das natürlich anders aus:

```
$ lpar -m ms01 lsmem
                  MEMORY           MEMORY            HUGE_PAGES
LPAR_NAME         MODE   AME   MIN    CURR    MAX      MIN   CURR   MAX
lpar1             ded    0.0   1024   2048    8192     0     0      0
mylpar01          ded    0.0   1024   4196    8192     0     0      0
aix01             ded    0.0   1024   8192    16384    0     0      0
aix02             ded    0.0   1024   8192    16384    0     0      0
ms01-vio1         ded    0.0   1024   6144    8192     -     -      -
ms01-vio2         ded    0.0   1024   6144    8192     -     -      -
$
```

Für die Anzeige der Virtual-I/O-Konfiguration gibt es das Kommando „*lpar lsvslot*“. Hier kann allerdings im Unterschied zu den Kommandos oben immer nur eine LPAR angezeigt werden, da die Ausgabe deutlich länger ist:

```
$ lpar -p standard lsvslot lpar1
SLOT  REQ  ADAPTER_TYPE   DATA
0     Yes  serial/server  remote: (any)/any connect_status= hmc=1
1     Yes  serial/server  remote: (any)/any connect_status= hmc=1
$
```

Wie erwartet besitzt die neu-angelegte LPAR nur die beiden standardmäßig angelegten seriellen Adapter für die Konsole. Lässt man die Profil-Angabe weg, wird auch hier wieder die gerade aktive Konfiguration gezeigt. Die physikalische-I/O-Konfiguration kann mit dem Kommando „*lpar lsslot*“ angezeigt werden. Da die LPAR *lpar1* keine physikalischen I/O Slots besitzt, wird hier kein Ausgabe gezeigt.

Wir legen auf einem zweiten Managed System eine weitere LPAR mit dem gleichen Namen *lpar1* an:

```
$ lpar create -m ms02 lpar1
.
    > lpar1
$
```

Auf einem Managed System müssen die Namen der LPARs allerdings eindeutig sein:

```
$ lpar create -m ms02 lpar1
hmc01: mksyscfg -r lpar -m ms02 -i
'desired_mem=2048,lpar_env=aixlinux,max_mem=8192,min_mem=1024,name=lpar1,profile_name=stand
ard'
ERROR: remote HMC command returned an error (1)
StdErr: An error occurred while creating the partition named lpar1.
StdErr: HSCL05DE A partition in the managed system already uses the name lpar1. Provide
another name for this partition.
$
```

Der Versuch eine zweite LPAR mit dem gleichen Namen anzulegen scheitert, mit dem Hinweis das es eine LPAR diesen Namens auf dem Managed System schon gibt. Im allgemeinen sollten LPAR-Namen in der ganzen Umgebung eindeutig sein, damit immer klar ist, welche LPAR gemeint ist. Auch die Benutzung des LPAR-Tools erschwert sich geringfügig, wenn mehrere LPARs den gleichen Namen haben. Möchte man beispielsweise die LPAR aktivieren, dann reicht die Angabe des LPAR Namens nicht mehr aus:

```
$ lpar activate lpar1
ERROR: more than one LPAR matches
USAGE:
   lpar [-h <hmc>] [-m <ms>] [-p <profile>] activate [-b <bootmode>] [-c] [-k <keylock>] [-
v] <lpar>
$
```

Es gibt hier zwei Kandidaten und es ist nicht klar, welche der beiden LPARs hier gemeint ist. Durch Angabe des Managed Systems kann dies natürlich einfach geklärt werden:

```
$ lpar -m ms01 activate lpar1
$
```

Bisher haben wir LPARs einfach mit Default-Werten angelegt, das wollen wir jetzt ändern. Über Attribute können verschiedenste LPAR-Konfiguration erreicht werden. Wenn man nicht alle möglichen Attribute auswendig weiß, kann die Hilfe zum Subkommando „*lpar create*“ nützlich sein, da diese neben der allgemeinen Benutzung auch die wichtigsten Attribute auflistet:

```
$ lpar help create
USAGE:
   lpar [-h <hmc>] [-m <ms>] [-p <profile>] create [-v] [<lpar>] [<attributes> ...]

DESCRIPTION

Create a new LPAR on a managed system.

Valid attributes:
   name : name for the LPAR
   lpar_id : the ID of the LPAR
   profile_name : name of the default profile
   lpar_env : type of LPAR
      aixlinux - AIX or Linux (default)
      os400 - IBM i
      vioserver - virtual I/O server
   min_mem : minimum amount of memory in MB
   desired_mem : desired amount of memory in MB
   max_mem : maximum amount of memory in MB
   mem_expansion : Active Memory Expansion
      0 - disable AME
      1.00-10.00 - expansion factor
   proc_mode : processor mode
      ded - dedicated processors
      shared - shared processors
   min_procs : minimum number of processors
   desired_procs : desired number of processors
   max_procs : maximum number of processors
   min_proc_units : minimum number of processor units
   desired_proc_units : desired number of processor units
   max_proc_units : maximum number of processor units
   sharing_mode : when processors are shared
      keep_idle_procs - never share processors
      share_idle_procs - share processors only when LPAR is inactive
```

```
      share_idle_procs_active - share processors only when LPAR is active
      share_idle_procs_always - share processors always
      cap - cap processing units
      uncap - uncap processing units
   uncap_weight : weight priority when competting for processors
   shared_proc_pool_name : shared processor pool
   shared_proc_pool_id : shared processor pool ID
   max_virtual_slots : maximum number of virtual slots
   (for additional attributes see the IBM documentation)

EXAMPLES

Create a new LPAR on managed system ms01 with default attributes:
   lpar -m ms01 create

Create IBM i LPAR with name testlpar on managed system ms01:
   lpar -m ms01 create testlpar lpar_env=os400   # or
   lpar -m ms01 create name=testlpar lpar_env=os400

Create dedicated processor AIX LPAR with 2 dedicated processors (min=1, max=4):
   lpar -m ms01 create proc_mode=ded min_procs=1 desired_procs=2 max_procs=4
$
```

Der Typ der LPAR kann über das Attribut *lpar_env* angegeben werden. Möchte man z.B. eine IBM i Partition anlegen, so kann dies durch Angabe des Typs *os400* getan werden:

```
$ lpar -m ms01 create sys1 lpar_env=os400
hmc01: mksyscfg -r lpar -m ms01 -i
'desired_mem=2048,lpar_env=os400,max_mem=8192,min_mem=1024,name=sys1,profile_name=standard'
ERROR: remote HMC command returned an error (1)
StdErr: An error occurred while creating the partition named sys1.
StdErr: One or more required attributes are missing.  The missing attributes are
console_slot.  Please correct the configuration data and retry the command.
$
```

Das Kommando schlägt fehl, zum Anlegen einer IBM i Partition muss das Attribut *console_slot* angegeben werden. Wir probieren es noch einmal und geben *hmc* als *console_slot* an:

```
$ lpar -m ms01 create sys1 lpar_env=os400 console_slot=hmc
.
    > sys1
$
```

Alle bisher angelegten LPARs waren Dedicated Prozessor LPARs (*proc_mode=ded*):

```
$ lpar lsproc sys1 lpar1
           PROC         PROCS            PROC_UNITS                          UNCAP   PROC
LPAR_NAME  MODE  MIN  DESIRED  MAX  MIN  DESIRED  MAX  CURR_SHARING_MODE     WEIGHT  POOL
lpar1      ded   0    0        0    -    -        -    share_idle_procs      -       -
sys1       ded   0    0        0    -    -        -    share_idle_procs      -       -
$
```

Natürlich lassen sich aber auch Shared Prozessor LPARs erzeugen, hierzu muss lediglich das Attribut *proc_mode* mit dem Wert *shared* verwendet werden:

```
$ lpar -m ms01 create proc_mode=shared
.
```

```
    > lpar2
$
```

Da keine weiteren Attribute angegeben wurden, werden für die Prozessor- und Speicher-Konfiguration Default-Werte verwendet:

```
$ lpar lsproc -p standard lpar2
            PROC          PROCS            PROC_UNITS                    UNCAP      PROC
LPAR_NAME   MODE    MIN   DESIRED  MAX   MIN   DESIRED   MAX   SHARING_MODE  WEIGHT   POOL
lpar2       shared  1     2        4     0.1   0.2       0.4   uncap         64       DefaultPool
$
```

Abschließend zeigen wir noch ein Beispiel in dem Prozessor- und Speicher-Konfiguration explizit angegeben werden:

```
$ lpar -m ms01 create proc_mode=shared min_procs=1 desired_procs=3 max_procs=5
min_proc_units=0.1 desired_proc_units=0.7 max_proc_units=1.5 uncap_weight=20
.
    > lpar3
$
```

2. Löschen einer LPAR

Auch das Löschen einer LPAR lässt sich leicht mit dem LPAR-Tool durchführen. Hierzu gibt es das Subkommando „*lpar delete*“:

```
$ lpar -m ms01 delete mylpar01
Deleting LPAR mylpar01
ERROR: lparDelete(): remote HMC command returned an error (1)
CMD on hmc01: rmsyscfg -m ms01 -r lpar -n mylpar01
StdErr: An error occurred while deleting the partition named mylpar01.
StdErr: HSCL05E6 Partition mylpar01 delete failed. Cannot delete a partition when its state
is not in the Not Activated state. Perform a shutdown operation then delete the partition.
$
```

Die zu löschende LPAR darf natürlich nicht aktiv sein. Dies lässt sich mit „*lpar status*“ vor dem Löschen überprüfen:

```
$ lpar status mylpar01
NAME     LPAR_ID  LPAR_ENV  STATE          PROFILE    SYNC  RMC       PROCS  PROC_UNITS
MEM   OS_VERSION
mylpar01 3        aixlinux  Running        myprofile  0     active2   1      0.2
4096  Unknown
$
```

Die LPAR kann über das OS heruntergefahren und ausgeschaltet werden, oder auch mit dem Subkommando „*lpar shutdown*“. Das Kommando „*lpar shutdown*“ löst über die HMC ein geordnetes Herunterfahren der LPAR aus:

```
$ lpar shutdown -m ms01 mylpar01
$
```

Wird der Status „*Not Activated*“ angezeigt, so kann die LPAR gelöscht werden:

```
$ lpar status mylpar01
NAME     LPAR_ID  LPAR_ENV  STATE          PROFILE    SYNC  RMC       PROCS  PROC_UNITS
MEM   OS_VERSION
mylpar01 3        aixlinux  Not Activated  myprofile  0     inactive  2      0.2
4096  Unknown

$
```

Jetzt sollte das Löschen der LPAR erfolgreich sein:

```
$ lpar delete mylpar01
$
```

5. DLPAR-Operationen

Physikalische und virtuelle Ressourcen können zur Laufzeit einer LPAR dynamisch während laufendem AIX oder Linux hinzugefügt oder auch wieder weggenommen werden. Einzige Voraussetzung hierfür ist eine funktionierende RMC-Verbindung zur HMC. Mit Hilfe des LPAR-Tools lassen sich DLPAR-Operationen komfortabel durchführen. Bei allen DLPAR-Operationen mit dem LPAR-Tool wird per default auch das aktuelle Profil angepaßt. Auf der GUI muss dies manuell durchgeführt werden und wird gerne einmal vergessen, was zu Problemen nach einer neuen Aktivierung führt. Optional kann man eine DLPAR-Operation aber auch nur zur Laufzeit, ohne Anpassung des aktuellen Profils, durchführen. Hierzu dient die Option „*-d*“. Es ist auch möglich die Änderung nur im Profil vorzunehmen, wenn z.B. die LPAR deaktiviert ist, sich im SMS-Menü befindet oder es keine aktive RMC-Verbindung zwischen LPAR und HMC gibt.

In den folgenden Unterkapiteln wird jeweils gezeigt, wie DLPAR-Operationen für verschiedene Ressourcen durchgeführt werden können.

1. Arbeitsspeicher einer LPAR ändern

Eine häufig Aufgabe bei der Verwaltung von LPARs ist die Erweiterung des Hauptspeichers. Im Profil einer LPAR wird angegeben innerhalb welcher Grenzen sich der Hauptspeicher dynamisch vergrößern und verkleinern lässt. Diese Grenzen lassen sich auf verschiedene Weisen anzeigen:

```
$ lpar lsmem lpar1
             MEMORY          MEMORY          HUGE_PAGES
LPAR_NAME   MODE   AME   MIN    CURR   MAX    MIN   CURR   MAX
lpar1       ded    0.0   1024   4096   8192   0     0      0
$
```

Es wird die aktuelle Konfiguration der LPAR angezeigt. Möchte man die Konfiguration die sich aus einem Profil ergibt anschauen, so muss die Option „*-p <profile>*“ verwendet werden:

```
$ lpar -p standard mem lpar1
             MEMORY          MEMORY             HUGE_PAGES
LPAR_NAME   MODE   AME   MIN    DESIRED   MAX    MIN   DESIRED   MAX
lpar1       ded    0.0   1024   4096      8192   0     0         0
$
```

Bei unserer Beispiel-LPAR *lpar1* sind die Speichergrenzen mindestens 1024 MB und 8192 MB konfiguriert. Die aktuelle Speichergröße ist 4096 MB. Im Profil ist ebenfalls 4096 MB als gewünschter Wert konfiguriert.

Um den Speicher dynamisch zu ändern, wird eine aktive RMC-Verbindung benötigt. Mit dem Kommando „*lpar status*“ lässt sich dies leicht herausfinden:

```
$ lpar status lpar1
NAME   LPAR_ID  LPAR_ENV  STATE    PROFILE   SYNC  RMC     PROCS  PROC_UNITS  MEM
OS_VERSION
lpar1  39       aixlinux  Running  standard  0     active  1      0.1         4096  AIX 7.2
7200-03-02-1846
$
```

Der Hauptspeicher der LPAR *lpar1* soll um 1024 MB erweitert werden, dynamisch als auch im aktuellen Profil. Wir überprüfen noch schnell ob auf dem Managed System noch 1024 MB RAM verfügbar ist. Hierzu benötigen wir das zugehörige Managed System:

```
$ lpar show lpar1
NAME   ID  SERIAL     LPAR_ENV  MS    HMCS
lpar1  39  XXXXXXXXX  aixlinux  ms01  hmc01,hmc02
$
```

Die LPAR *lpar1* befindet sich auf dem Managed System *ms01*.

Das Managed System *ms01* verfügt noch über ausreichend freie RAM Kapazitäten:

```
$ ms lsmem ms01
NAME  INSTALLED  FIRMWARE  CONFIGURABLE  AVAIL   MEM_REGION_SIZE
ms01  1048576    17920     1048576       647680  256
$
```

Wir führen nun die RAM-Erweiterung von *lpar1* durch:

```
$ lpar addmem lpar1 1024
$
```

Der zusätzliche Speicher steht der LPAR sofort zur Verfügung. Das Profil der LPAR ist automatisch entsprechend abgeändert werden, so daß bei der nächsten Aktivierung mit dem aktuellen Profil die gleiche Hauptspeichergröße verwendet wird.

Soll die Speichergröße nur temporär erhöht werden, so kann die Option „*-d*“ (*dynamic only*) verwendet werden. Im Profil bleibt dann der alte Wert erhalten und die Erweiterung wird nur dynamisch durchgeführt:

```
$ lpar -d addmem lpar1 1024
$
```

Wird die LPAR ausgeschaltet und später wieder aktiviert, dann wird sie wieder mit 2048 MB RAM starten.

Möchte man die Speichergröße nur im Profil ändern, wird anstelle der Option „*-d*“ die Option „*-p <profile>*“ verwendet:

```
$ lpar -p standard addmem lpar1 1024
$
```

Soll der Speicher einer LPAR verkleinert werden, dann gibt es analog das Kommando „*lpar rmmem*“ mit den gleichen Optionen wie „*lpar addmem*“.

2. Hauptspeichergrenzen im Profil ändern

Die minimale und maximale RAM Größe einer LPAR lässt sich nur im Profil ändern. Damit sich die Änderung auswirkt, muss die LPAR gestoppt werden und anschließend unter Verwendung des geänderten Profils neu gestartet werden.

Soll der Speicher der *lpar1* oben auf 16384 MB erhöht werden, ist dies aufgrund der Grenze maximaler Speicher von 8192 MB nicht zur Laufzeit möglich. Hierzu muss das Profil angepasst werden, und dies Grenze auf einen Wert von mindestens 16384 MB erhöht werden. Nur dann kann die gewünschte Speichergröße von 16384 MB konfiguriert werden. Eine Änderung der Speichergrenzen im Profil kann mit dem Kommando „*lpar chmem*“ durchgeführt werden:

```
$ lpar -p standard chmem lpar1 min_mem=1024 desired_mem=16384 max_mem=32768
$
```

Im Beispiel wurde der minimal Wert auf 1024 MB belassen, die gewünschte Speichergröße auf 16384 MB gesetzt und der maximale Wert auf 32768 MB. Dies lässt Puffer für weitere Erhöhungen später, die dann bis zu einer Größe von 32768 MB dann wieder dynamisch durchgeführt werden können.

Die LPAR muss nach dem Herunterfahren und Ausschalten mit dem geänderten Profil wieder aktiviert werden:

```
$ lpar -p standard activate lpar1
$
```

3. Anzahl Prozessor Cores und Prozessor Units ändern

Bei Performance Engpässen kann die Anzahl der Prozessor Cores, sowie die Anzahl der Prozessor Units geändert werden. Im Profil einer LPAR wird angegeben innerhalb welcher Grenzen sich diese dynamisch vergrößern und verkleinern lassen. Diese Grenzen lassen sich auf verschiedene Weisen anzeigen:

```
$ lpar lsproc lpar1
            PROC          PROCS            PROC_UNITS                              UNCAP     PROC
LPAR_NAME   MODE     MIN  DESIRED  MAX   MIN  DESIRED  MAX  CURR_SHARING_MODE   WEIGHT   POOL
lpar1       shared   1    1        2     0.1  0.1      0.4  uncap                10
DefaultPool
$
```

Es wird die aktuelle Konfiguration der LPAR angezeigt. Möchte man die Konfiguration die sich aus einem Profil ergibt anschauen, so muss die Option „*-p <profile>*“ verwendet werden:

```
$ lpar -p standard lsproc lpar1
             PROC          PROCS            PROC_UNITS                        UNCAP       PROC
LPAR_NAME   MODE     MIN  DESIRED  MAX   MIN  DESIRED  MAX  SHARING_MODE   WEIGHT   POOL
lpar1       shared   1    1        2     0.1  0.1      0.4  uncap          10       DefaultPool
$
```

Bei unserer Beispiel-LPAR *lpar1* sind die Grenzen bei der Anzahl Prozessor Cores mindestens 1 und maximal 2. Die aktuelle Anzahl Prozessor Cores ist 1. Im Profil ist ebenfalls 1 als gewünschter Wert konfiguriert. Die Grenzen bei

den Prozessor Units sind mindestens 0.1 und maximal 0.4. Der aktuelle Wert der Prozessor Units ist 0.1. Auch im Profil ist dies als gewünschter Wert konfiguriert.

Um den Speicher dynamisch zu ändern, wird eine aktive RMC-Verbindung benötigt. Mit dem Kommando „*lpar status*“ lässt sich dies leicht herausfinden.

Die Anzahl Prozessor Cores soll um 1, die Anzahl Prozessor Units um 0.1 erweitert werden, dynamisch als auch im aktuellen Profil. Wir überprüfen noch schnell ob auf dem Managed System noch Prozessor Units verfügbar sind. Hierzu benötigen wir das zugehörige Managed System:

```
$ lpar show lpar1
NAME   ID  SERIAL     LPAR_ENV  MS    HMCS
lpar1  39  XXXXXXXXX  aixlinux  ms01  hmc01,hmc02
$
```

Die LPAR *lpar1* befindet sich auf dem Managed System *ms01*.

Das Managed System *ms01* verfügt noch über ausreichend nicht zugewiesene Prozessor Units:

```
$ ms lsproc ms01
NAME  INSTALLED  CONFIGURABLE  AVAIL  MAX_SHARED_PROC_POOLS
ms01  20.0       20.0          6.5    64
$
```

Wir führen nun die Prozessor Erweiterung von *lpar1* durch:

```
$ lpar addprocs lpar1 1
$ lpar addprocunits lpar1 0.1
$
```

Der zusätzliche Core und die zusätzlichen Prozessor Units stehen der LPAR sofort zur Verfügung. Das Profil der LPAR ist automatisch entsprechend abgeändert werden, so daß bei der nächsten Aktivierung mit dem aktuellen Profil die gleiche Konfiguration verwendet wird.

Sollen die Prozessor Cores und Prozessor Units nur temporär erhöht werden, so kann die Option „*-d*“ (*dynamic only*) verwendet werden. Im Profil bleibt dann der alte Wert erhalten und die Erweiterung wird nur dynamisch durchgeführt:

```
$ lpar -d addprocs lpar1 1
$ lpar -d addprocunits lpar1 0.1
$
```

Wird die LPAR ausgeschaltet und später wieder aktiviert, dann wird sie wieder mit 2 Cores und 0.2 Processor Units starten.

Möchte man die Prozessor Cores nur im Profil ändern, wird anstelle der Option „*-d*“ die Option „*-p <profile>*“ verwendet:

```
$ lpar -p standard addprocs lpar1 1
```

```
$ lpar -p standard addprocunits lpar 0.1
$
```

Sollen Prozessor Cores und/oder Prozessor Units reduziert werden, dann gibt es analog das Kommando „*lpar rmprocs*“ bzw. „*lpar rmprocunits*“ mit den gleichen Optionen wie „*lpar addprocs*“ und „*lpar addprocunits*“.

4. Prozessor Grenzen im Profil ändern

Die minimale und maximale Anzahl Cores sowie Prozessor Units einer LPAR lässt sich nur im Profil ändern. Damit sich die Änderung auswirkt, muss die LPAR gestoppt werden und anschließend unter Verwendung des geänderten Profils neu gestartet werden.

Soll die Anzahl Cores der *lpar1* oben auf 5 erhöht werden, ist dies aufgrund der Grenze maximale Anzahl Cores von 4 nicht zur Laufzeit möglich. Hierzu muss das Profil angepasst werden, und die Grenze auf einen Wert von mindestens 5 erhöht werden. Nur dann kann die gewünschte Anzahl Cores von 5 konfiguriert werden. Eine Änderung der Prozessor Grenzen im Profil kann mit dem Kommando „*lpar chproc*“ durchgeführt werden:

```
$ lpar -p standard chproc lpar1 min_procs=1 desired_procs=5 max_procs=8
$ lpar -p standard chproc lpar1 min_proc_units=0.1 desired_proc_units=0.5
max_proc_units=2.0
$
```

Im Beispiel wurde der minimal Wert der Cores bei 1 belassen, die gewünschte Anzahl Cores auf 5 gesetzt und der maximale Wert auf 8. Für die Prozessor Units wurde als minimaler Wert 0.1, für den maximalen Wert 2.0 und den gewünschten Wert 0.5 gesetzt. Dies lässt Puffer für weitere Erhöhungen später, die dann bis zu einer Größe von 8 Cores und 2.0 Prozessor Units dann wieder dynamisch durchgeführt werden können.

Die LPAR muss nach dem Herunterfahren und Ausschalten mit dem geänderten Profil wieder aktiviert werden:

```
$ lpar -p standard activate lpar1
$
```

5. Physikalische Slots konfigurieren

Auch physikalische Slots lassen sich per DLPAR-Operation im laufenden Betrieb in eine LPAR herein und heraus konfigurieren.

Welche physikalischen Slots einer LPAR aktuell schon zugeordnet sind, läßt sich mit dem Kommando „*lpar lsslot*“ anzeigen:

```
$ lpar lsslot ms01-vio1
DRC_NAME                    DRC_INDEX  IOPOOL  DESCRIPTION
U78C0.001.XXXXXXX-P2-C3     21010203   none    Quad 8 Gigabit Fibre Channel Adapter
U78C0.001.XXXXXXX-P2-C2     21010204   none    Unknown
U78C0.001.XXXXXXX-P2-C1     21010205   none    10 Gigabit Ethernet-SFP+ SR PCI-E adapter
U78C0.001.XXXXXXX-P2-C9-T1  21010208   none    PCI-E SAS Controller
U78C0.001.XXXXXXX-P2-C9-T2  21010209   none    PCI-E SAS Controller
U78C0.001.XXXXXXX-P2-C2     21010244   none    Quad 8 Gigabit Fibre Channel Adapter
U78C0.001.XXXXXXX-P2-C1     21010245   none    10 Gigabit Ethernet-SFP+ SR PCI-E adapter
```

```
U78C0.001.XXXXXXX-P2-C5     2101024D   none    4-Port 10/100/1000 Base-TX PCI Express
Adapter
$
```

Die Konfiguration im Profil kann natürlich abweichen, läßt sich aber ebenso leicht mit der Option „*-p <profile>*“ anzeigen:

```
$ lpar -p standard lsslot ms01-vio1
DRC_NAME                    DRC_INDEX  REQ  IOPOOL  DESCRIPTION
U78C0.001.XXXXXXX-P2-C2     21010204   No   none    Unknown
U78C0.001.XXXXXXX-P2-C1     21010205   No   none    10 Gigabit Ethernet-SFP+ SR PCI-E
adapter
U78C0.001.XXXXXXX-P2-C9-T1  21010208   No   none    PCI-E SAS Controller
U78C0.001.XXXXXXX-P2-C9-T2  21010209   No   none    PCI-E SAS Controller
U78C0.001.XXXXXXX-P2-C2     21010244   No   none    Quad 8 Gigabit Fibre Channel Adapter
U78C0.001.XXXXXXX-P2-C1     21010245   No   none    10 Gigabit Ethernet-SFP+ SR PCI-E
adapter
U78C0.001.XXXXXXX-P2-C5     2101024D   No   none    4-Port 10/100/1000 Base-TX PCI Express
Adapter
$
```

Soll ein weiterer Slot in die LPAR hereinkonfiguriert werden, empfiehlt es sich zunächst die verfügbaren physikalischen Slots des Managed Systems aufzulisten:

```
$ ms lsslot ms01
DRC_NAME                    DRC_INDEX  IOPOOL  LPAR_NAME  DESCRIPTION
U78C0.001.XXXXXXX-P2-T3     21010200   none    -          RAID Controller
U78C0.001.XXXXXXX-P2-C8-T7  21010201   none    -          Generic XT-Compatible Serial
Controller
U78C0.001.XXXXXXX-P2-C4     21010202   none    -          Empty slot
U78C0.001.XXXXXXX-P2-C3     21010203   none    ms01-vio1  Quad 8 Gigabit Fibre Channel
Adapter
U78C0.001.XXXXXXX-P2-C2     21010204   none    ms01-vio1  Unknown
U78C0.001.XXXXXXX-P2-C1     21010205   none    ms01-vio1  10 Gigabit Ethernet-SFP+ SR PCI-E
adapter
U78C0.001.XXXXXXX-P2-C9-T1  21010208   none    ms01-vio1  PCI-E SAS Controller
U78C0.001.XXXXXXX-P2-C9-T2  21010209   none    ms01-vio1  PCI-E SAS Controller
U78C0.001.XXXXXXX-P2-C8-T5  2101020A   none    -          Universal
…
$
```

Die Ausgabe zeigt alle physikalischen Slots des Managed Systems, sowie die aktuelle Zuordnung zu LPARs. Um einen dieser Slots zuzuweisen, darf dieser aktuell nicht einer anderen LPAR zugeordnet sein. Technisch muss beim hinzufügen der DRC-Index des Slots angegeben werden, der beim „*ms lsslot*“ Kommando in einer eigenen Spalte ausgegeben wird. Mit Hilfe des DRC-Index und dem Kommando „*lpar addslot*“ kann die Zuordnung erfolgen:

```
$ lpar addslot lpar1 2101000A
$
```

Die Änderung wird wieder dynamisch und im Profil durchgeführt. Die neue Hardware muss im Betriebssystem dann noch konfiguriert werden, was durch Starten des *cfgmgr* unter AIX als root durchgeführt werden kann:

```
lpar1 # cfgmgr
lpar1 #
```

Soll der Slot nur temporär dynamisch hinzugefügt werden, kann wieder die Option „*-d*“ verwendet werden:

```
$ lpar -d addslot lpar1 2101000A
$
```

Änderungen nur am Profil können wieder mit der Option „*-p <profile>*“ erreicht werden:

```
$ lpar -p standard addslot lpar1 2101000A
$
```

Das Herauskonfigurieren von physikalischen Slots funktioniert analog mit dem Kommando „*lpar rmslot*“. Zu beachten ist hier aber, das alle Devices des Slots im Betriebssystem entfernt werden müssen (*rmdev*)!

6. Virtuelle Ethernet Slots

Virtuelle Ethernet Slots können normalen LPARs oder Virtual-I/O-Servern hinzugefügt werden. Wir schauen uns zunächst nur normale LPARs an, für diese können keine sogenannten Trunking Adapter angelegt werden.

Um einen virtuellen Ethernet Adapter anzulegen benötigt man zunächst Informationen über die verfügbaren VSwitches und VLANs auf dem System. Diese Informationen kann man z.B. mit dem Kommando „*ms lsvswitch*“ abfragen:

```
$ ms lsvswitch ms01
MS   VSWITCH              SWITCH_MODE  VLAN_IDS
ms01  ETHPROD             VEB          720,735,437
ms01  ETHERNET0(Default)  VEB          100,102,105,107
ms01  ETHMGMT             VEB          1400,1600
$
```

Wir legen zunächst einen Virtuellen Ethernet Adapter für das VLAN 100 (Default Switch ETHERNET0) an. Als Slot-Nummer auf der LPAR wählen wir den Slot 5 (dieser darf natürlich noch nicht benutzt sein):

```
$ lpar addeth lpar1 5 100
$
```

Dies funktioniert natürlich nur wenn es schon einen Ethernet-Adapter in der LPAR mit aktiver RMC Verbindung gibt. Ansonsten bleibt nur der Weg über eine Änderung des Profils und neues Aktivieren der LPAR.

Wir legen einen weiteren Virtuellen Ethernet Adapter an, diese Mal im VLAN 720 (VSwitch *ETHPROD*). Da der Adapter dieses Mal nicht an den Default VSwitch geht, muss der VSwitch explizit angegeben werden:

```
$ lpar addeth -s ETHPROD lpar1 6 720
$
```

Bei beiden Adapter wurden dynamisch konfiguriert (DLPAR-Operation) und es wurde jeweils das aktuelle Profil um die Adapter erweitert. Die nach der Slot-Nummer angegebene VLAN-ID ist eine sogenannte PVID (Port-VLAN-ID), das bedeutet das nicht getaggte Ethernet-Pakete mit der angegebenen PVID getaggt werden. Möchte man weitere VLANs auf dem gleichen Adapter verwenden, dann ist dies auch möglich. Hierzu muss dann die Option „*-i*“ (IEEE) verwendet werden, es sind dann bis zu 20 weitere VLANs auf dem Adapter möglich:

```
$ lpar addeth -i lpar1 7 102 105,107
$
```

Die weiteren VLANs werden als kommaseparierte Liste angegeben.

Das Resultat läßt sich wiederum mit dem Kommando „*lpar lsvslot*“ anzeigen und zwar einmal die aktive Konfiguration:

```
$ lpar lsvslot lpar1

SLOT  REQ  ADAPTER_TYPE   STATE  DATA
…
2     No   eth            1      PVID=1400 VLANS= ETHMGMT XXXXXXXXXXXX
5     No   eth            1      PVID=100 VLANS= ETHERNET0 XXXXXXXXXXXX
…
$
```

Und die virtuellen Slots im Profil:

```
$ lpar -p standard lsslot lpar1
SLOT  REQ  ADAPTER_TYPE   DATA
…
2     No   eth            PVID=1400 VLANS= ETHMGMT
5     No   eth            PVID=100 VLANS= ETHERNET0
…
$
```

Wie bisher lassen sich mit der Option „*-d*“ Adapter nur dynamisch konfigurieren, oder mit der Option „*-p <profile>*“ nur ins Profil eintragen!

Möchte man Virtuelle Ethernet Adapter wieder wegnehmen, gibt es hierfür das Kommando „*lpar rmeth*“:

```
$ lpar rmeth lpar1 7
$
```

Neben der LPAR muss nur die Slot-Nummer angegeben werden.

7. Virtuelle SCSI Adapter

Bei virtuellen SCSI Adaptern muss zum einen ein virtueller SCSI Client Adapter in einer LPAR angelegt werden. Zusätzlich muss für diesen Adapter auf einem Virtual-I/O-Server des Managed Systems ein virtueller SCSI Server Adapter angelegt werden. Das LPAR-Tool legt standardmäßig sowohl den Client- als auch den Server-Adapter an. I/Os in der Client LPAR werden dann über den virtuellen SCSI Client Adapter mit Hilfe des Hypervisors an den

virtuellen SCSI Server Adapter eines Virtual-I/O-Servers weitergeleitet, dieser fungiert dann quasi als Storage System.

Einen virtuellen SCSI Client Adapter kann man mit dem Kommando „*lpar addscsi*“ anlegen:

```
$ lpar addscsi lpar1 11 ms01-vio1 56
lpar1: slot 11 -> ms01-vio1/56 added by DLPAR operation
lpar1: slot 11 -> ms01-vio1/56 added to current profile (standard)
ms01-vio1: slot 56 -> lpar1/11 added by DLPAR operation
$
```

Hierbei ist *11* die Slot-Nummer für den VSCSI-Client-Adapter auf der der Client LPAR *lpar1* und *56* die zu verwendende Slot-Nummer für den VSCSI-Server-Adapter auf dem Virtual-I/O-Server *ms01-vio1*. Das Kommando „*lpar addscsi*“ legt sowohl den Client-Adapter als auch den Server-Adapter an. Außerdem werden Client- und Server-Adapter auch in die jeweiligen Profile eingetragen.

Schauen wir uns kurz die virtuellen Adapter der LPAR *lpar1* an:

```
$ lpar lsvslot lpar1
SLOT  REQ  ADAPTER_TYPE   STATE  DATA
0     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
1     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
5     No   eth            1      PVID=1234 VLANS= ETHERNET0 XXXXXXXXXXXX
6     No   vnic           -      PVID=1200 VLANS=none XXXXXXXXXXXX failover sriov/ms01-
vio1/1/3/0/2700c003/2.0/2.0/20/100.0/100.0
10    No   fc/client      1      remote: ms01-vio1(1)/47 c050760XXXXX0016,c050760XXXXX0017
11    No   scsi/client    1      remote: ms01-vio1(1)/56
20    No   fc/client      1      remote: ms01-vio2(2)/25 c050760XXXXX0018,c050760XXXXX0019
$
```

Natürlich kann jeder Slot nur einmal verwendet werden! Wir schauen kurz an was passiert wenn man versucht im Slot *11* einen zweiten VSCSI-Client-Adapter anzulegen:

```
$ lpar addscsi lpar1 11 ms01-vio1 57
hmc01: chhwres -m ms01 -r virtualio --rsubtype scsi -o a -p lpar1 -s 11 -a
'adapter_type=client,remote_lpar_name=ms01-vio1,remote_slot_num=57'
ERROR: remote HMC command returned an error (1)
StdErr: HSCL294C Dynamic add of virtual I/O resources failed:
StdErr: A Virtual I/O device already exists at slot 11.
$
```

Die Fehlermeldung sagt eindeutig das es schon ein Device in Slot *11* gibt.

Wir verwenden auf der Client-Seite den unbenutzten Slot *12*, auf Seiten des Virtual-I/O-Servers aber den schon oben benutzten Slot *56*:

```
$ lpar addscsi lpar1 12 ms01-vio1 56
lpar1: slot 12 -> ms01-vio1/56 added by DLPAR operation
lpar1: slot 12 -> ms01-vio1/56 added to current profile (standard)
hmc01: chhwres -m ms01 -r virtual –rsubtype scsi -o a -p ms01-vio1 -s 56 -a
adapter_type=server,remote_lpar_name=lpar1,remote_slot_num=12
ERROR: remote HMC command returned an error (1)
StdErr: HSCL294C Dynamic add of virtual I/O resource failed:
StdErr: A Virtual I/O device already exists at slot 56.
$
```

Auch hier bekommt man eine Fehlermeldung. Dieses Mal wird der schon benutzte Slot *56* auf dem Virtual-I/O-Server angemahnt. Allerdings wurde der Client-Adapter schon angelegt, wie die Ausgabe von „*lpar vslots*“ zeigt:

```
$ lpar lsvslot lpar1
SLOT  REQ  ADAPTER_TYPE   STATE  DATA
0     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
1     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
5     No   eth            1      PVID=1234 VLANS= ETHERNET0 XXXXXXXXXXXX
6     No   vnic           -      PVID=1200 VLANS=none XXXXXXXXXXXX failover sriov/ms01-
vio1/1/3/0/2700c003/2.0/2.0/20/100.0/100.0
10    No   fc/client      1      remote: ms01-vio1(1)/47 c050760XXXXX0016,c050760XXXXX0017
11    No   scsi/client    1      remote: ms01-vio1(1)/56
12    No   scsi/client    1      remote: ms01-vio1(1)/56
20    No   fc/client      1      remote: ms01-vio2(2)/25 c050760XXXXX0018,c050760XXXXX0019
$
```

Bevor wir den VSCSI-Client-Adapter in Slot *12* der Client-LPAR *lpar1* wieder löschen, schauen wir kurz auf den Virtual-I/O-Server *ms01-vio1*:

```
$ lpar lsvslot ms01-vio1 | grep lpar1
47    No   fc/server      1      remote: lpar1(39)/10
56    No   scsi/server    1      remote: lpar1(39)/11
$
```

Der VSCSI-Server Adapter in Slot *56* ist nach wie vor mit Slot *11* der Client-LPAR *lpar1* verbunden.

Wie beim Anlegen von VSCSI-Adaptern werden auch beim Löschen die zugehörigen VSCSI-Server Adapter auf dem Virtual-I/O-Server mit entfernt.

Wir löschen nun den VSCSI-Client Adapter in Slot *12* der Client-LPAR *lpar1*:

```
$ lpar rmscsi lpar1 12
lpar1: slot 12 -> ms01-vio1/56 removed by DLPAR operation
lpar1: slot 12 -> ms01-vio1/56 removed from current profile (standard)
$
```

Das LPAR-Tool erkennt das der VSCSI-Server-Adapter in Slot *56* des Virtual-I/O-Servers zu einem anderen Client-Adapter gehört und entfernt diesen daher nicht, stattdessen wird ein entsprechender Hinweis ausgegeben. (Dieser erscheint zwei mal, da der Adapter einmal dynamisch und einmal im Profil entfernt werden muss.)

Es ist nicht zwingend notwendig Slot-Nummern beim Anlegen von VSCSI-Adaptern anzugeben. Man kann die Auswahl der Slot-Nummern auch dem LPAR-Tool überlassen. Das LPAR-Tool ermittelt dann zunächst freie Slot-Nummern und verwendet diese dann für das Anlegen der VSCSI-Adapter.

Wir löschen zunächst den VSCSI-Client-Adapter in Slot *11* (samt VSCSI-Server-Adapter):

```
$ lpar rmscsi lpar1 11
lpar1: slot 11 -> ms01-vio1/56 removed by DLPAR operation
lpar1: slot 11 -> ms01-vio1/56 removed from current profile (standard)
ms01-vio1: slot 56 -> lpar1/11 removed by DLPAR operation
```

```
$
```

Nun legen wir erneut einen VSCSI-Client-Adapter an, geben aber die Slot-Nummer auf dem Virtual-I/O-Server nicht an:

```
$ lpar addscsi lpar1 11 ms01-vio1
lpar1: slot 11 -> ms01-vio1/21 added by DLPAR operation
lpar1: slot 11 -> ms01-vio1/21 added to current profile (standard)
ms01-vio1: slot 21 -> lpar1/11 added by DLPAR operation
$
```

Auf dem Virtual-I/O-Server wurde die Slot-Nummer *21* ausgewählt! Auf eine Überprüfung mit Hilfe von „*lpar lsvslot lpar1*“ und „*lpar lsvslot ms01-vio1*“ verzichten wir an dieser Stelle.

Ein VSCSI-Client-Adapter kann auch nur dynamisch einer LPAR hinzugefügt werden, hierzu dient die Option „*-d*“:

```
$ lpar -d addscsi lpar1 12 ms01-vio1
lapr1: slot 12 -> ms01-vio1/22 added by DLPAR operation
ms01-vio1: slot 22 -> lpar1/12 added by DLPAR operation
$
```

Die Slot-Nummer *22* für den VSCSI-Server-Adapter wurde wieder vom LPAR-Tool ermittelt, kann aber natürlich auch angegeben werden.

Der VSCSI-Client-Adapter wurde nicht ins Profil eingetragen:

```
$ lpar -p standard lsvslot lpar1
SLOT  REQ  ADAPTER_TYPE   DATA
0     Yes  serial/server  remote: (any)/any connect_status= hmc=1
1     Yes  serial/server  remote: (any)/any connect_status= hmc=1
5     No   eth            PVID=1234 VLANS= ETHERNET0
6     Yes  vnic           PVID=1200 VLANS=none XXXXXXXXXXXX failover sriov/ms01-
vio1/1/3/0/2.0/10/100.0
10    No   fc/client      remote: ms01-vio1(1)/47 c050760XXXXX0016,c050760XXXXX0017
11    No   scsi/client    remote: ms01-vio1(1)/56
20    No   fc/client      remote: ms01-vio2(2)/25 c050760XXXXX0018,c050760XXXXX0019
$
```

Auf dem Virtual-I/O-Server wird die Änderung immer dynamisch und im gerade aktiven Profil durchgeführt!

Soll ein VSCSI-Client-Adapter nur in einem Profil konfiguriert werden, so geht dies mit der Option „*-p <profile>*“:

```
$ lpar -p standard -r addscsi lpar1 13 ms01-vio1
lpar1: slot 13 -> ms01-vio1/23 added to current profile (standard)
ms01-vio1: slot 23 -> lpar1/13 added by DLPAR operation
$
```

Zusätzlich haben wir hier noch die Option ‚*-r*‘ für *required* angegeben. Die LPAR kann dann mit diesem Profil nur dann aktiviert werden, wenn der VSCSI-Client-Adapter in Slot *13* auch verfügbar ist.

Auf dem Virtual-I/O-Server wird der zugehörige VSCSI-Server-Adapter wie immer dynamisch und im Profil angelegt!

In seltenen Fällen ist das automatische Anlegen der VSCSI-Server-Adapter auf einem Virtual-I/O-Server nicht gewünscht (z.B. weil dieser schon existiert). Über die Option ‚*-c*' für *client-only* kann dem LPAR-Tool mitgeteilt werden, dass nur der Client-Adapter angelegt werden soll:

```
$ lpar -c addscsi lpar1 ms01-vio2
lpar1: slot 4 -> ms01-vio1/24 added by DLPAR operation
lpar1: slot 4 -> ms01-vio1/24 added to current profile (standard)
$
```

(Hier wurde auf dem Client der Slot *4* und auf dem Virtual-I/O-Server der Slot *24* vom LPAR-Tool ausgewählt, die Slot-Nummern können aber natürlich auch angegeben werden.)

Auf dem Virtual-I/O-Server *ms01-vio2* wurde kein Adapter angelegt:

```
$ lpar lsvslot ms01-vio2 | grep lpar1
25    No   fc/server      1      remote: lpar1(39)/20
$
```

Die Option ‚*-c*' kann mit den Optionen ‚*-d*' und ‚*-p*' kombiniert werden.

Auch beim Löschen eines VSCSI-Client-Adapters kann angegeben werden, dass nur der Client-Adapter gelöscht werden soll:

```
$ lpar -c rmscsi lpar1 4
lpar1: slot 4 -> ms01-vio1/24 removed by DLPAR operation
lpar1: slot 4 -> ms01-vio1/24 removed from current profile (standard)
$
```

8. Virtuelle FC Adapter

Auch bei virtuellen FC Adaptern gibt es einen Client Adapter in der LPAR und einen zugehörigen Server Adapter auf einem Virtual-I/O-Server. Auch hier wird vom LPAR-Tool sowohl der Client- als auch der Server-Adapter standardmäßig angelegt. Das Anlegen geht daher analog zu den virtuellen SCSI Adaptern:

```
$ lpar addfc lpar1 21 ms01-vio2 55
lpar1: slot 21 -> ms01-vio2/55 added by DLPAR operation
lpar1: slot 21 -> ms01-vio2/55 added to current profile (standard)
ms01-vio2: slot 55 -> lpar1/21 added by DLPAR operation
$
```

Hierbei ist *21* die Slot-Nummer für den virtuellen FC-Client-Adapter auf der Client LPAR *lpar1* und *55* die zu verwendende Slot-Nummer für den virtuellen FC-Server-Adapter auf dem Virtual-I/O-Server *ms01-vio2*. Das Kommando „*lpar addfc*" legt sowohl den Client-Adapter als auch den Server-Adapter an. Außerdem werden Client- und Server-Adapter auch in die jeweiligen Profile eingetragen.

Schauen wir uns kurz die virtuellen Adapter der LPAR *lpar1* an:

```
$ lpar lsvslot lpar1
SLOT  REQ  ADAPTER_TYPE   STATE  DATA
0     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
1     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
5     No   eth            1      PVID=1234 VLANS= ETHERNET0 XXXXXXXXXXXX
6     No   vnic           -      PVID=1200 VLANS=none XXXXXXXXXXXX failover sriov/ms01-
vio1/1/3/0/2700c003/2.0/2.0/20/100.0/100.0
10    No   fc/client      1      remote: ms01-vio1(1)/47 c050760XXXXX0016,c050760XXXXX0017
20    No   fc/client      1      remote: ms01-vio2(2)/25 c050760XXXXX0018,c050760XXXXX0019
21    No   fc/client      1      remote: ms01-vio2(2)/55 c050760XXXXX004c,c050760XXXXX004d
$
```

Der virtuelle FC Client Adapter bekommt automatisch 2 WWPNs zugewiesen, die erste wird aktiv verwendet, sobald dem zugehörigen virtuellen FC Server Adapter ein physikalischer FC Port zugewiesen wird. Die zweite WWPN wird für Live Partition Mobility verwendet.

Auf dem Virtual-I/O-Server *ms01-vio2* wurde ein entsprechender Server-Adapter angelegt:

```
$ lpar lsvslot ms01-vio2 | grep lpar1
25    No   fc/server      1      remote: lpar1(39)/20
55    No   fc/server      1      remote: lpar1(39)/21
$
```

Wie schon bei den VSCSI-Adaptern, müssen Slot-Nummern nicht zwingend angegeben werden, sondern können vom LPAR-Tool ausgewählt werden. Wir legen einen weiteren virtuellen FC-Client Adapter an und lassen das LPAR-Tool die Slot-Nummer auf dem Virtual-I/O-Server bestimmen:

```
$ lpar addfc lpar1 22 ms01-vio2
lpar1: slot 22 -> ms01-vio2/20 added by DLPAR operation
lpar1: slot 22 -> ms01-vio2/20 added to current profile (standard)
ms01-vio2: slot 20 -> lpar1/22 added by DLPAR operation
$
$ lpar lsvslot lpar1
SLOT  REQ  ADAPTER_TYPE   STATE  DATA
0     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
1     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
5     No   eth            1      PVID=1234 VLANS= ETHERNET0 XXXXXXXXXXXX
6     No   vnic           -      PVID=1200 VLANS=none XXXXXXXXXXXX failover sriov/ms01-
vio1/1/3/0/2700c003/2.0/2.0/20/100.0/100.0
10    No   fc/client      1      remote: ms01-vio1(1)/47 c050760XXXXX0016,c050760XXXXX0017
20    No   fc/client      1      remote: ms01-vio2(2)/25 c050760XXXXX0018,c050760XXXXX0019
21    No   fc/client      1      remote: ms01-vio2(2)/55 c050760XXXXX004c,c050760XXXXX004d
22    No   fc/client      1      remote: ms01-vio2(2)/20 c050760XXXXX004e,c050760XXXXX004f
$
```

Hier wurde die Slot-Nummer *20* auf dem Virtual-I/O-Server *ms01-vio2* vom LPAR-Tool ausgewählt.

Alle weiteren Optionen funktionieren für FC-Adapter ganz genauso wie bei VSCSI-Adaptern beschrieben und sollen hier nicht erneut beschrieben werden.

Das Entfernen eines virtuellen FC-Adapters kann mit dem Kommando „*lpar rmfc*“ analog zum Kommando „*lpar rmscsi*“ durchgeführt werden. Wir zeigen das Entfernen hier für den zuletzt angelegten Adapter in Slot *22*:

```
$ lpar rmfc lpar1 22
lpar1: slot 22 -> ms01-vio2/20 removed by DLPAR operation
lpar1: slot 22 -> ms01-vio2/20 removed from current profile (standard)
ms01-vio2: slot 20 -> lpar1/22 removed by DLPAR operation
$
```

Neben dem Client-Adapter werden auch hier wieder die Server-Adapter auf dem Virtual-I/O-Server mit entfernt.

Hat man einen Client-Adapter versehentlich gelöscht, braucht diesen aber doch noch, kann man nicht einfach einen neuen Adapter anlegen. Der neue Adapter bekommt ein zwei neue (bisher noch nicht benutzte) WWPNs. Auf LUNs die auf die alten WWPNs gemappt sind, kann mit den neuen WWPNs nicht zugegriffen werden (außer die Storage-Konfiguration und SAN-Zoning wird entsprechend abgeändert). Es besteht aber die Möglichkeit einen Client-Adapter anzulegen und dabei die gewünschten WWPNs anzugeben. Wir verwenden im Beispiel die WWPNs *c050760XXXXX004e* und *c050760XXXXX004f* die vom Adapter im Slot *22* verwendet wurden:

```
$ lpar addfc lpar1 22 ms01-vio2 20 c050760XXXXX004e,c050760XXXXX004f
lpar1: slot 22 -> ms01-vio2/20 added by DLPAR operation
lpar1: slot 22 -> ms01-vio2/20 added to current profile (standard)
ms01-vio2: slot 20 -> lpar1/22 added by DLPAR operation
$
```

Der virtuelle FC-Adapter in Slot *22* der Client-LPAR *lpar1* hat die angegebenen WWPNs:

```
$ lpar lsvslot lpar1
SLOT  REQ  ADAPTER_TYPE   STATE  DATA
0     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
1     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
5     No   eth            1      PVID=1234 VLANS= ETHERNET0 XXXXXXXXXXXX
6     No   vnic           -      PVID=1200 VLANS=none XXXXXXXXXXXX failover sriov/ms01-
vio1/1/3/0/2700c003/2.0/2.0/20/100.0/100.0
10    No   fc/client      1      remote: ms01-vio1(1)/47 c050760XXXXX0016,c050760XXXXX0017
20    No   fc/client      1      remote: ms01-vio2(2)/25 c050760XXXXX0018,c050760XXXXX0019
21    No   fc/client      1      remote: ms01-vio2(2)/55 c050760XXXXX004c,c050760XXXXX004d
22    No   fc/client      1      remote: ms01-vio2(2)/20 c050760XXXXX004e,c050760XXXXX004f
$
```

Damit kann auf die alten LUNs wieder zugegriffen werden.

9. SR-IOV

Um SR-IOV benutzen zu können, benötigt man zum einen ein Managed System, welches SR-IOV unterstützt, und zum anderen einen SR-IOV fähigen PCI-Adapter. Ob ein Managed System SR-IOV unterstützt und SR-IOV fähige Adapter besitzt, lässt sich relativ einfach mit dem Kommando „*ms lssriov*“ feststellen. Hier ein Beispiel für ein nicht SR-IOV fähiges Managed System:

```
$ ms lssriov ms05
hmc01: lshwres -r sriov --rsubtype adapter -m ms05
ERROR: remote HMC command returned an error (1)
StdErr: HSCL1237 The managed system does not support SR-IOV.
$
```

Ein Beispiel für ein SR-IOV fähiges System ohne SR-IOV fähige Adapter zeigt die folgende Ausgabe:

```
$ ms lssriov ms15
PHYS_LOC  SLOT_ID  CONFIG_STATE  SRIOV_STATUS  ADAPTER_ID  PHYS_PORTS  LOGICAL_PORTS
$
```

Interessanter ist natürlich ein System das SR-IOV fähige Adapter besitzt:

```
$ ms lssriov ms22
PHYS_LOC                  SLOT_ID   CONFIG_STATE  SRIOV_STATUS  ADAPTER_ID  PHYS_PORTS
LOGICAL_PORTS
U78D3.001.XXXXXXX-P1-C6   2102001b  sriov         running       1           4
48
U78D3.001.XXXXXXX-P1-C4   21010020  dedicated     null          null        null
null
U78D3.001.XXXXXXX-P1-C9   21010010  dedicated     null          null        null
null
U78D3.001.XXXXXXX-P1-C11  21020013  sriov         running       2           4
48
$
```

Die Ausgabe zeigt 4 SR-IOV fähige Adapter. Die Adapter in den Slots P1-C9 und P1-C4 haben den *config_state dedicated*. Das bedeutet das diese Adapter aktuell klassisch ohne SR-IOV Funktionalität genutzt werden. Die Adapter können nur einer LPAR zugewiesen und nur von dieser direkt genutzt werden. Die Adapter in den Slots P1-C11 und P1-C6 haben den *config_state sriov*. Das bedeutet das diese Adapter von mehreren LPARs gleichzeitig benutzt werden können. Die Adapter selbst werden in diesem Falle keiner LPAR zugewiesen, sondern es werden sogenannte Logische Ports erzeugt, die unterschiedlichen LPARs zugewiesen werden können. Die Adapter können von mehreren LPARs gemeinsam direkt benutzt werden. Dazu werden sogenannte Virtual Functions verwendet, die in Form von Logical Ports den LPARs zugewiesen werden.

Damit ein Adapter für SR-IOV verwendet werden kann, muss er zunächst vom klassischen dedicated Mode auf den sogenannten SR-IOV Mode umgeschaltet werden. Dies kann mit dem Kommando „*ms chsriov*“ durchgeführt werden:

```
$ ms chsriov ms22 21010010 shared
$
```

Ist der Adapter aktuell einer LPAR zugeordnet, kann der Mode nicht umgeschaltet werden, wie das folgende Beispiel zeigt:

```
$ ms chsriov ms22 21010010 shared
hmc01: chhwres -r sriov --rsubtype adapter -m ms22 -o a -a 'slot_id=21010010'
ERROR: remote HMC command returned an error (1)
StdErr: HSCL1232 The command failed because the adapter is in use by partitions ms22-vio2.
Remove the adapter's resources from the specified partitions and try the operation again.
$
```

Die Ausgabe von „*ms lssriov*“ zeigt, dass der Adapter nun den *config_state sriov* hat. Außerdem wird nun die Adapter ID, die Anzahl der physikalischen Ports und die maximale Anzahl der logischen Ports angezeigt:

```
$ ms lssriov ms22
```

```
PHYS_LOC                  SLOT_ID   CONFIG_STATE  SRIOV_STATUS  ADAPTER_ID  PHYS_PORTS
LOGICAL_PORTS
U78D3.001.XXXXXXX-P1-C6   2102001b  sriov         running       1           4
48
U78D3.001.XXXXXXX-P1-C4   21010020  dedicated     null          null        null
null
U78D3.001.XXXXXXX-P1-C9   21010010  sriov         running       15          4
    48
U78D3.001.XXXXXXX-P1-C11  21020013  sriov         running       2           4
48
$
```

Der SR-IOV Adapter hat die Adapter ID 15 bekommen, hat 4 physikalische Ports und es können maximal 48 logische Ports konfiguriert werden. Soll eine bestimmte Adapter ID verwendet werden, so kann dies beim Umkonfigurieren angegeben werden. Wie, zeigt uns die Hilfe zum Kommando:

```
$ ms help chsriov
USAGE:
…

Attributes when switching an adapter to shared mode:
   adapter_id - 1-32, default: assign next available adapter ID

Note: An SR-IOV adapter can only be changed to shared mode, if it is not assigned
to an LPAR!

EXAMPLES:

…
Switch adapter 21010010 of managed system ms02 to shared mode using
adapter ID 3:
   ms chsriov ms01 21010010 shared adapter_id=3
$
```

Wir probieren dies aus und vergeben für den Adapter oben die Adapter ID 3:

```
$ ms chsriov ms22 21010010 dedicated
$ ms chsriov ms22 21010010 shared adapter_id=3
$
```

Eine schon existierende Adapter ID führt natürlich zu einem Fehler:

```
$ ms  chsriov ms22 21010010 shared adapter_id=2
hmc01: chhwres -r sriov --rsubtype adapter -m ms22 -o a -a adapter_id=2,slot_id=21010010
ERROR: remote HMC command returned an error (1)
StdErr: HSCL1299 The operation to switch the adapter in slot 21010010 to shared mode failed
with the following errors:
StdErr:
StdErr: HSCL1301 Valid adapter IDs are from 1 to 32 and must be unique on the managed
system. The adapter ID cannot be changed while the adapter is in SR-IOV mode.
$
```

Die physikalischen Ports der Adapter lassen sich mit Hilfe der Option ‚*-p*' (physical port) anzeigen:

```
$ ms lssriov -p ms22
```

```
PHYS_PORT_LOC                LABEL  TYPE  ADAPTER  PPORT  USED  MAX  CONN_SPEED  MTU
U78D3.001.XXXXXXX-P1-C6-T3   -      eth   1        2      0     4    0           9000
U78D3.001.XXXXXXX-P1-C6-T4   -      eth   1        3      0     4    0           9000
U78D3.001.XXXXXXX-P1-C4-T3   -      eth   2        2      1     4    1000        9000
U78D3.001.XXXXXXX-P1-C4-T4   -      eth   2        3      0     4    0           9000
U78D3.001.XXXXXXX-P1-C9-T3   -      eth   4        2      1     4    1000        9000
U78D3.001.XXXXXXX-P1-C9-T4   -      eth   4        3      0     4    0           9000
U78D3.001.XXXXXXX-P1-C11-T3  -      eth   3        2      0     4    0           9000
U78D3.001.XXXXXXX-P1-C11-T4  -      eth   3        3      0     4    0           9000
U78D3.001.XXXXXXX-P1-C6-T1   -      ethc  1        0      1     20   10000       9000
U78D3.001.XXXXXXX-P1-C6-T2   -      ethc  1        1      1     20   0           9000
U78D3.001.XXXXXXX-P1-C4-T1   -      ethc  2        0      1     20   10000       9000
U78D3.001.XXXXXXX-P1-C4-T2   -      ethc  2        1      1     20   10000       9000
U78D3.001.XXXXXXX-P1-C9-T1   -      ethc  4        0      1     20   10000       9000
U78D3.001.XXXXXXX-P1-C9-T2   -      ethc  4        1      2     20   10000       9000
U78D3.001.XXXXXXX-P1-C11-T1  -      ethc  3        0      1     20   10000       9000
U78D3.001.XXXXXXX-P1-C11-T2  -      ethc  3        1      1     20   0           9000
$
```

Die Spalte USED gibt an wieviele logische Ports aktuell konfiguriert sind, die Spalte MAX gibt an wieviel maximal konfiguriert werden können.

Einige Attribute eines physikalischen Ports lassen sich mit dem Kommando „*ms chsriov*“ ändern. Die Hilfe zum Kommando listet die meisten der änderbaren Attribute auf:

```
$ ms help chsriov
USAGE:
   ms [-h <hmc>] chsriov [-v] <ms> {<slot_id> {dedicated|shared} | <adapter_id>
<phys_port_id> <attributes>} [<attributes> ...]

DESCRIPTION:

Switches an SR-IOV adapter in a managed system either to dedicated or shared mode,
or sets attributes for an SR-IOV physical port.

Attributes when switching an adapter to shared mode:
   adapter_id - 1-32, default: assign next available adapter ID

Attributes for an SR-IOV physical port:
   conn_speed - ethernet speed
      auto : autonegotation
      10 : 10 Mbps
      100 : 100 Mbps
      1000 : 1 Gbps
      10000 : 10 Gbps
      40000 : 40 Gbps
      100000 : 100 Gbps
   max_recv_packet_size - MTU
      1500 - 1500 byts
      9000 - 9000 bytes (jumbo frames)
   phys_port_label - label for the physical port
      1-16 characters
      none - to clear the label
   phys_port_sub_label - sublabel for the physical port
      1-8 characters
      none - to clear the sublabel
   recv_flow_control
      0 - disable
      1 - enable
   trans_flow_control
      0 - disable
```

```
      1 - enable
   veb_mode
      0 - disable virtual ethernet bridge mode
      1 - enable virtual ethernet bridge mode
   vepa_mode
      0 - disable virtual ethernet port aggregator mode
      1 - enable virtual ethernet port aggregator mode
   (see the IBM documentation for additional attributes)
…
$
```

Wir konfigurieren Jumbo Frames und vergeben einen Namen für den Port 0 des Adapters 3:

```
$ ms chsriov ms22 3 0 phys_port_label=myport max_recv_packet_size=9000
$ ms lssriov -p -s adapter_id=3,phys_port_id=0 ms22
PHYS_PORT_LOC                LABEL   TYPE  ADAPTER  PPORT  USED  MAX  CONN_SPEED  MTU
U78D3.001.XXXXXXX-P1-C11-T1  myport  ethc  3        0      1     20   10000       9000
$
```

Nicht alle Eigenschaften eines physikalischen Ports werden standardmäßig angezeigt. Möchte man wirklich alle Eigenschaften sehen, oder möchte man die Ausgabe in Skripten verwenden, dann empfiehlt sich die Ausgabe im JSON, Stanza oder YAML Format:

```
$ ms lssriov -p -s adapter_id=3,phys_port_id=0 -y ms22
---
  adapter_id: 3
  capabilities:
[pvid_priority_capable,port_vlan_id_capable,mac_vlan_consistency_capable,clear_phys_port_st
at_capable,clear_logical_port_stat_capable,disable_enable_logical_ports_capable]
  config_conn_speed: auto
  config_logical_ports: 1
  config_max_recv_packet_size: 9000
  config_recv_flow_control: 0
  config_trans_flow_control: 0
  conn_speed: 10000
  curr_eth_logical_ports: 1
  max_diag_ports: 1
  max_eth_logical_ports: 20
  max_promisc_ports: 1
  max_recv_packet_size: 9000
  phys_port_id: 0
  phys_port_label: myport
  phys_port_loc: U78D3.001.XXXXXXX-P1-C11-T1
  phys_port_max_logical_ports: 20
  phys_port_sub_label: null
  phys_port_type: ethc
  priority_flow_control_active: 0
  recv_flow_control: 0
  state: 1
  supported_max_eth_logical_ports: 20
  trans_flow_control: 0
  veb_mode: 1
  vepa_mode: 0
$
```

Nachdem nun der Adapter im SR-IOV Mode ist und die physikalischen Ports konfiguriert wurden, können nun logische Ports für LPARs erzeugt werden. Wir starten mit einer unter AIX 7.2 laufenden LPAR, die bisher keine logischen SR-IOV Ports besitzt:

```
$ lpar status aix02
NAME   LPAR_ID  LPAR_ENV  STATE    PROFILE   RMC     PROCS  PROC_UNITS  MEMORY
OS_VERSION
aix02  39       aixlinux  Running  standard  active  1      0.7         4096    AIX 7.2
7200-03-02-1846
$
$ lpar lssriov aix02
LPORT  REQ  ADAPTER  PPORT  CONFIG_ID  CAPACITY  MAX_CAPACITY  PVID  VLANS  CURR_MAC_ADDR
$
$ lpar lssriov -p standard aix02
LPORT  ADAPTER  PPORT  CONFIG_ID  CAPACITY  MAX_CAPACITY  PVID  VLANS  MAC_ADDR
ALLOWED_OS_MAC_ADDRS
$
```

Wir erzeugen für diese LPAR einen logischen Port auf Adapter 3 physikalischer Port 0. Der logische Port soll die PVID 1200 haben:

```
$ lpar addsriov aix02 3 0 port_vlan_id=1200
$
```

Das Kommando fügt den Port per DLPAR-Operation hinzu und aktualisiert das aktuelle Profil um sicherzustellen das bei einer neuen Aktivierung der LPAR der logische SR-IOV Port verfügbar ist. Die logischen Ports lassen sich mittels „*lpar lssriov*“ auflisten:

```
$ lpar lssriov aix02
LPORT     REQ  ADAPTER  PPORT  CONFIG_ID  CAPACITY  MAX_CAPACITY  PVID  VLANS
CURR_MAC_ADDR
2700c004  1    3        0      0          2.0       100.0         1200  all    xxxxxx945f00
$
$ lpar lssriov -p standard aix02
LPORT     ADAPTER  PPORT  CONFIG_ID  CAPACITY  MAX_CAPACITY  PVID  VLANS  MAC_ADDR
ALLOWED_OS_MAC_ADDRS
2700c004  3        0      0          2.0       100.0         1200  all    -
all
$
```

Die Ausgabe zeigt das der logische Port standardmäßig mit einer garantierten Kapazität von 2% (Spalte *CAPACITY*) und einer maximalen Kapazität von 100% angelegt wurde. Dies kann man natürlich beim Erzeugen des logischen Ports angegeben, Attribute *capacity* bzw. *max_capacity*. Der logische SR-IOV Port ist unter AIX im Zustand *Defined* sichtbar und kann nach einem *cfgmgr*-Lauf benutzt werden!

```
aix02:/root> lscfg -l ent*
  ent0             U9009.22A.YYYYYYY-V39-C5-T1     Virtual I/O Ethernet Adapter (l-lan)
  ent1             U78D3.001.XXXXXXX-P1-C11-T1-S5  PCIe3 10GbE SFP+ SR 4-port Converged
Network Adapter VF (df1028e214100f04)
aix02:/root>
```

Das es sich hier um einen logischen SR-IOV Port handelt kann man der Beschreibung des Adapter-Typs in der Ausgabe von *lscfg* entnehmen. Dort steht unter anderem „… *Network Adapter VF* …“. Die Bezeichnung VF für Virtual Function ist ein Hinweis für SR-IOV. Der gemeinsame Zugriff auf die Hardware der gleichen PCI-Karte wird bei SR-IOV über Virtual Functions realisiert, dabei bekommt jedes zugreifende System (LPAR) eine solche Virtual Function zugewiesen.

Die Konfiguration des resultierenden Ethernet Adapters unter AIX funktioniert so wie immer. Das es sich hier um einen logischen SR-IOV Port handelt ist anhand des Device-Namens *ent1* nicht ersichtlich.

Die Attribute eines logischen SR-IOV Ports können mit dem Kommando „*lpar chsriov*“ geändert werden. Mögliche Attribute kann man der Online Hilfe entnehmen:

```
$ lpar help chsriov
USAGE:
   lpar [-h <hmc>] [-m <ms>] [-p <profile>] chsriov [-d] [-f] [-l <detail_level>] [-w
<wait_time>] [-v] <lpar> <config_id>|<logical_port_id> <attributes> ...

DESCRIPTION:

Changes attributes of an SR-IOV logical port.

   allowed_os_mac_addrs - allowed MAC addresses
   allowed_priorities - allowed QoS priorities
      none : no priorities allowed (default)
      0-7: comma-separated list of integers
      all : all priorities are allowed
   allowed_vlan_ids - allowed VLANs
      none : no VLAN IDs allowed
      VLAN_IDs: comma-separated list of VLAN IDs
      all : all VLAN IDs are allowed (default)
   diag_mode - diagnostics mode
      0 : disable (default)
      1 : enable
   port_vlan_id - VLAN-ID for untagged packets or 0 to disable VLAN tagging
   pvid_priority - priority of the PVID

EXAMPLES:

Set the PVID of logical port with config_id 1 of LPAR aix01 to 1200:
   lpar chsriov aix01 1 port_vlan_id=1200

Set the PVID and PVID priority of logical port 2700c006 of LPAR aix01 dynamically only:
   lpar chsriov -d aix01 2700c006 port_vlan_id=1200 pvid_priority=0

Set the PVID priority of logical port with config_id 1 in the profile 'standard'
of LPAR aix01 to 2:
   lpar -p standard chsriov aix01 1 pvid_priority=2

$
```

Wir zeigen hier das Ändern der Attribute *port_vlan_id* und *pvid_priority*:

```
$ lpar chsriov aix02 0 port_vlan_id=1200 pvid_priority=2
$
```

Anstelle der *config_id* (hier *0*) kann auch die *logical_port_id* (hier *2700c004*) als Argument angegeben werden, um den logischen SR-IOV Port auszuwählen.

Zum Abschluß soll noch gezeigt werden, wie bei Bedarf ein logischer SR-IOV Port wieder entfernt werden kann. Voraussetzung dazu ist, das der Port im Betriebssystem nicht mehr benutzt wird und mit *rmdev* entfernt wurde!

```
$ lpar rmsriov  aix02 2700c004
$
```

Die Gesamtheit der logischen SR-IOV Ports eines Managed Systems, kann mit „*ms lssriov –l*“ aufgelistet werden:

```
$ ms lssriov -l ms22
LOCATION_CODE                  ADAPTER_ID  PHYS_PORT_ID  LOGICAL_PORT_ID  LPAR_NAME
CAPACITY  CURR_MAC_ADDR
U78D3.001.XXXXXXX-P1-C6-T1-S1  1           0             27004001         ms22-vio2
50.0      xxxxxxf72200
U78D3.001.XXXXXXX-P1-C6-T2-S2  1           1             27004002         ms22-vio2
50.0      xxxxxxf72201
U78D3.001.XXXXXXX-P1-C4-T1-S1  2           0             27008001         ms22-vio2
50.0      xxxxxxf72202
U78D3.001.XXXXXXX-P1-C4-T2-S2  2           1             27008002         ms22-vio2
10.0      xxxxxxf72203
U78D3.001.XXXXXXX-P1-C4-T3-S3  2           2             27008003         ms22-vio2
50.0      xxxxxxc01a04
U78D3.001.XXXXXXX-P1-C9-T1-S1  4           0             27010001         ms22-vio1
50.0      xxxxxx58de02
U78D3.001.XXXXXXX-P1-C9-T2-S2  4           1             27010002         ms22-vio1
50.0      xxxxxx58de03
U78D3.001.XXXXXXX-P1-C9-T3-S3  4           2             27010003         ms22-vio1
50.0      xxxxxx37e504
U78D3.001.XXXXXXX-P1-C9-T2-S4  4           1             27010004         aix01
10.0      xxxxxxdff700
U78D3.001.XXXXXXX-P1-C11-T1-S1 3           0             2700c001         ms22-vio1
50.0      xxxxxx07fe00
U78D3.001.XXXXXXX-P1-C11-T2-S2 3           1             2700c002         ms22-vio1
50.0      xxxxxx58de01
U78D3.001.XXXXXXX-P1-C11-T1-S5 3           0             2700c004         aix02
4.0       xxxxxx945f09
$
```

6. Virtual-I/O-Server

Eine Vielzahl von Operationen auf einem Virtual-I/O-Server lässt sich bequem über das LPAR-Tool durchführen, ohne das man sich direkt auf dem Virtual-I/O-Server einloggen muss.

1. Virtual Media Repository

Wir zeigen hier die Administration einer Virtual Media Repository auf einem Virtual-I/O-Server mit Hilfe des LPAR-Tools.

Zunächst legen wir eine Repository mit einer Größe von 20 GB an:

```
$ vios mkrep ms04-vio2 20g
Virtual Media Repository Created
Repository created within "VMLibrary" logical volume
$
```

Details zur Repository lassen sich mit dem Kommando "*vios lsrep*" anzeigen:

```
$ vios lsrep ms04-vio2
Size(mb) Free(mb) Parent Pool         Parent Size      Parent Free
   20396    20396 rootvg                  279552           172544
$
```

Um die Virtual Optical Library nutzen zu können, benötigt eine LPAR ein virtuelles CD-Laufwerk. Ein virtuelles CD-Laufwerk ist ein virtuelles SCSI Device, d.h. die LPAR benötigt einen virtuellen SCSI Adapter. Unsere Beispiel-LPAR verfügt schon über einen solchen Adapter:

```
$ lpar lsvslot lpar6
SLOT  REQ  ADAPTER_TYPE   STATE  DATA
0     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
1     Yes  serial/server  1      remote: (any)/any connect_status=unavailable hmc=1
5     No   eth            1      PVID=1234 VLANS= ETHERNET0 XXXXXXXXXXXX
10    No   fc/client      1      remote: ms04-vio1(1)/47 c050760XXXXX0016,c050760XXXXX0017
20    No   fc/client      1      remote: ms04-vio2(2)/25 c050760XXXXX0018,c050760XXXXX0019
21    No   scsi/client    1      remote: ms04-vio2(1)/21
$
```

Um ein virtuelles CD-Laufwerk (File Backed Optical) anlegen zu können, benötigen wir noch den *vhost*-Adapter auf dem Virtual-I/O-Server für den VSCSI Client Adapter:

```
$ vios vscsi ms04-vio2
VIOS        SLOT   NAME        CLIENT                LUNS
ms04-vio2   C19    vhost0      0x00000006            0
ms04-vio2   C21    vhost1      0x00000001            0
$
```

Der richtige Adapter ist *vhost1* (Slot *C21*). Nun können wir das virtuelle CD-Laufwerk anlegen:

```
$ vios mkfbo ms04-vio2 vhost1
```

```
vtopt0 Available
$
```

Auf Wunsch kann dem virtuellen optischen Device auch ein beschreibender Name gegeben:

```
$ vios mkfbo ms04-vio2 vhost1 lpar6_cd
lpar6_cd Available
$
```

Die Devices lassen sich mit dem Kommando „*vios lsvopt*“ anzeigen:

```
$ vios lsvopt ms04-vio2
VTD             Media                                           Size(mb)
lpar6_cd        No Media                                             n/a
vtopt0          No Media                                             n/a
$
```

Wir haben einige Medien in der Virtual Optical Library angelegt (noch nicht vom LPAR-Tool unterstützt):

```
$ vios lsrep ms04-vio2
Size(mb) Free(mb) Parent Pool         Parent Size      Parent Free
   51703    45887 rootvg                  571392           286720

Name                                             File Size Optical         Access
aix610702-mksysb                                      1752 None            ro
aix710103-mksysb                                      1643 None            ro
aix710104-mksysb                                      1653 None            ro
blank                                                  768 None            rw
$
```

Ein virtuelles optisches Medium kann mit dem Kommando „*vios loadopt*“ in ein virtuelles CD-Laufwerk eingelegt werden:

```
$ vios loadopt ms04-vio2 lpar6_cd blank
$
```

Welches Medium gerade in ein virtuelles Laufwerk eingelegt ist, kann man wie folgt sehen:

```
$ vios lsrep ms04-vio2
Size(mb) Free(mb) Parent Pool         Parent Size      Parent Free
    1015      915 rootvg                  139776            99072

Name                                             File Size Optical         Access
blank                                                  100 lpar6_cd        rw
```

Das virtuelle Medium steht damit in der Client LPAR zur Verfügung.

Um ein Medium wieder „*auszuwerfen*“, kann „*vios unloadopt*“ verwendet werden:

```
$ vios unloadopt ms04-vio2 lpar6_cd
$
```

Sollte die Kapazität der Repository ausgeschöpft sein, kann sie relativ einfach vergrößert werden:

```
$ vios chrep ms04-vio2 10g
$
```

Dies geht natürlich nur solange noch Platz in der *rootvg* ist.

Soll die Virtual Media Repository gelöscht werden, so ist dies auch mit dem LPAR-Tool möglich:

```
$ vios rmrep ms04-vio2
ERROR: viosRmrep(): remote HMC command returned an error (1)
CMD on hmc01: viosvrcmd -m ms04 -p ms04-vio2 -c "rmrep"
StdErr: HSCL2970 The IOServer command has failed because of the following reason:
StdErr: DVD repository contains file backed DVD media. Use -f to remove
StdErr:
StdErr: rc=4
$
```

Allerdings müssen die Medien vorher gelöscht werden. Alternativ kann man aber auch mit der Option „*-f*“ ein Löschen aller Medien beim Löschen der Virtual Media Repository erzwingen:

```
$ vios rmrep -f ms04-vio2
$
```

2. Administration von VSCSI

Auf dem Virtual-I/O-Server wird für jeden virtual SCSI Server Adapter ein *vhost*-Device angelegt:

```
$ vios vscsi ms01-vio1
VIOS       SLOT  NAME    CLIENT  LUNS
ms01-vio1  C35   vhost0  0x04    18
ms01-vio1  C80   vhost1  0x05    7
ms01-vio1  C48   vhost2  0x03    0
$
```

Zum Slot *48* gehört das Device *vhost2* und es sind bisher keine virtuellen Target Devices auf dieses *vhost*-Device zugeordnet.

Nehmen wir an, das auf dem Virtual-I/O-Server die Platte *hdisk15* noch nicht in Verwendung ist und der Client LPAR *lpar1* zugeordnet werden soll, z.B. als Platte für die *rootvg*, dann kann man die Zuordnung mit dem folgenden Kommando durchführen:

```
$ vios map ms01-vio1 vhost2 hdisk15
$
```

Eine kurze Überprüfung zeigt, das jetzt eine LUN dem *vhost2*-Device zugeordnet ist:

```
$ vios vscsi ms01-vio1
VIOS        SLOT  NAME    CLIENT  LUNS
ms01-vio1   C35   vhost0  0x04    18
ms01-vio1   C80   vhost1  0x05    7
ms01-vio1   C48   vhost2  0x03    1
$
```

Genauere Informationen bekommt man, wenn man als zusätzliches Argument das *vhost*-Device angibt:

```
$ vios vscsi ms01-vio1 vhost2
VIOS        VHOST   DISK     VTD
ms01-vio1   vhost2  hdisk15  vtscsi2
$
```

Der Name *vtscsi2* ist etwas unschön, man kann diesem nicht ansehen wofür dieses benutzt wird. Viele Administratoren vergeben Namen die den Verwendungszweck und die Client-LPAR andeuten, z.B. *lpar1_hd0* um anzudeuten das dieses Device in der LPAR *lpar1* als *hdisk0* verwendet wird. Natürlich kann man beim Mappen mit dem LPAR-Tool auch einen beliebigen Namen für das virtuelle Target Device angeben:

```
$ vios map ms01-vio1 vhost2 hdisk16 lpar1_hd1
$
```

Ein solcher Name ist gleich viel informativer, insbesondere wenn sehr viele Platten mit VSCSI verwendet werden:

```
$ vios vscsi ms01-vio1 vhost2
VIOS        VHOST   DISK     VTD
ms01-vio1   vhost2  hdisk15  vtscsi2
ms01-vio1   vhost2  hdisk16  lpar1_hd1
$
```

Selbstverständlich kann ein solches Mapping auch wieder entfernt werden. Dazu sollte als erstes die zugehörige hdisk in der Client LPAR aus dem Betriebssystem herauskonfiguriert werden. Um das Mapping dann auf dem Virtual-I/O-Server zu löschen, verwendet man das Kommando „*vios unmap*“:

```
$ vios unmap ms01-vio1 vhost2 lpar1_hd1
$
```

3. Administration von NPIV

Auf dem Virtual-I/O-Server wird für den virtuellen FC Server Adapter ein *vfchost*-Device erzeugt. Dieses können wir mit dem Kommando „*vios npiv*“ auflisten:

```
$ vios npiv ms01-vio2
VIOS        ADAPT  NAME      CLIENT  OS   ADAPT  STATUS        PORTS
ms01-vio2   fcs3   vfchost1  lpar2   AIX  fcs1   LOGGED_IN       7
ms01-vio2   fcs1   vfchost2  lpar3   AIX  fcs1   LOGGED_IN       5
ms01-vio2          vfchost3  lpar1   AIX  fcs1   NOT_LOGGED_IN   0
$
```

Das *vfchost*-Device ist in diesem Fall *vfchost3*. Das *vfchost3*-Device ist noch auf keinen physikalischen Adapter zugeordnet (Spalte 2 *ADAPT*). Daher ist der Status noch *NOT_LOGGED_IN* (nicht in die Fabric eingeloggt) und die Anzahl der Ports ist *0*.

Es fehlt also noch die Zuordnung zu einem physikalischen Port. Dies kann leicht mit dem Kommando „*vios vfcmap*“ nachgeholt werden:

```
$ vios vfcmap ms01-vio2 vfchost3 fcs3
$
```

Status und Zuordnung des Adapters sollte sich nun geändert haben:

```
$ vios npiv ms01-vio2
VIOS       ADAPT  NAME      CLIENT  OS   ADAPT  STATUS       PORTS
ms01-vio2  fcs3   vfchost1  lpar2   AIX  fcs1   LOGGED_IN      7
ms01-vio2  fcs1   vfchost2  lpar3   AIX  fcs1   LOGGED_IN      5
ms01-vio2  fcs3   vfchost3  lpar1   AIX  fcs1   LOGGED_IN      1
$
```

Der Adapter ist jetzt im Status *LOGGED_IN* und es wird ein Port angezeigt. Die erste WWPN des virtuellen FC Client Adapters ist jetzt per NPIV zusätzlich auf den physikalischen Port *fcs3* gemappt. Diese WWPN ist in der Fabric eingeloggt und ist aktuell der einzige Port in seiner Zone.

Platten können nun direkt auf die WWPN der Client LPAR per Zoning konfiguriert werden. Die Platten müssen nicht mehr wie bei VSCSI mühsam auf den Virtual-I/O-Servern zugeordnet werden.

Um das Mapping des *vfchost*-Devices auf einen physikalischen Port wieder aufzulösen, gibt man beim „*vios vfcmap*“ Kommando einfach keinen physikalischen Port an:

```
$ vios vfcmap ms01-vio2 vfchost3
$
```

7. Administration von Managed Systems

1. Multiple Shared Processor Pools

Seit der Einführung von POWER6 Servern gibt es die Möglichkeit mehrere Prozessor Pools zu konfigurieren. Damit können LPARs in Prozessor Pools organisiert werden, um z.B. eine bessere Kontrolle über die Performance zu haben oder um Software-Lizenzen auf LPARs in einem Prozessor Pool einzuschränken.

Welche Prozessor Pools auf einem Managed System verfügbar sind, läßt sich mit dem Kommando „*ms lsprocpool*“ auflisten:

```
$ ms lsprocpool ms09
MS          PROCPOOL                  MAX   RESERVED
ms09        DefaultPool(0)              -          -
ms09        pool1(1)                  4.0        0.0
ms09        pool2(2)                  1.0        0.0
$
```

(Hinweis: Es können auch mehrere Managed Systems angegeben werden)

Die ID eines Prozessor Pools wird in Klammern hinter dem benutzerdefinierten Poolnamen angezeigt. Der Prozessor Pool *DefaultPool* mit der ID 0 existiert immer und ist auch immer aktiv. In der Spalte *MAX* wird die maximale Pool Kapazität aufgelistet, diese ist immer eine ganze Zahl und gibt an wieviele Procunits im Prozessor Pool maximal verfügbar sind. Die LPARs in einem Prozessor Pool können diese maximale Kapazität nicht überschreiten. Die Spalte *RESERVED* gibt an wieviele Procunits für LPARs im Prozessor Pool reserviert sind (zusätzlich zur Entitled Capacity der LPARs).

Die Ausgabe kann mit Hilfe der Optionen „*–f*“ und/oder „*–F*“ selbst definiert werden:

```
$ ms lsprocpool -f ms09
ms09:
   name=DefaultPool
   shared_proc_pool_id=0
   \"lpar_names=lpar2,lpar1,ms09-vio2,ms09-vio1,lpar3\"
   \"lpar_ids=4,3,2,1,7\"

ms09:
   name=pool1
   shared_proc_pool_id=1
   max_pool_proc_units=4.0
   curr_reserved_pool_proc_units=0.0
   pend_reserved_pool_proc_units=0.0
   lpar_ids=none

ms09:
   name=pool2
   shared_proc_pool_id=2
   max_pool_proc_units=1.0
   curr_reserved_pool_proc_units=0.0
   pend_reserved_pool_proc_units=0.0
   lpar_ids=none

$
```

```
$ ms lsprocpool -F name:max_pool_proc_units ms09
name:max_pool_proc_units
DefaultPool:-
pool1:4.0
pool2:1.0
$
```

Neben dem *DefaultPool* gibt es noch 63 weitere Shared Prozessor Pools. Per Default heißen diese *SharedPoolN*, wobei *N* die *ID* des Pools ist (*N*=1..63). Weitere Pools können nicht angelegt werden, es können nur die bestehenden Pools umkonfiguriert werden. Solange einem Pool keine Prozessor Ressourcen zugewiesen sind, wird er vom LPAR-Tool nicht angezeigt.

Wir weisen dem Pool *SharedPool6* einen Prozessor Core zu und benennen ihn gleichzeitig um in *mypool*:

```
$ ms chprocpool ms09 SharedPool06 new_name=mypool max_pool_proc_units=1
$
```

Jetzt wird der Pool bei „*ms lsprocpool*“ auch angezeigt:

```
$ ms lsprocpool ms09
MS         PROCPOOL                 MAX   RESERVED
ms09       DefaultPool(0)             -          -
ms09       pool1(1)                 4.0        0.0
ms09       pool2(2)                 1.0        0.0
ms09       mypool(6)                1.0        0.0
$
```

2. Administration von Virtuellen Switches

Auf jedem Managed System gibt es per Default immer den Virtuellen Ethernet Switch *ETHERNET0*. In der Regel werden für die LPARs dann virtuelle Ethernet Adapter erzeugt, die an diesen VSwitch angebunden sind. Der Netzwerk-Verkehr wird dann an einen sogenannten Trunking Port auf einem Virtual-I/O-Server weitergeleitet, der Bestandteil eines Shared-Ethernet Adapters (SEA) ist. Jeder SEA besitzt mindestens einen physikalischen Ethernet-Adapter über welchen dann der Netzwerk-Verkehr in ein physikalisches LAN weitergeleitet wird.

Wird nun ein weiterer Shared-Ethernet Adapter mit einem physikalisch getrennten LAN konfiguriert, dann sollte die Trennung der Netze auch im Managed System erhalten bleiben. Benutzt man allerdings den Default VSwitch *ETHERNET0*, der auch schon für den bisherigen SEA verwendet wird, dann werden die extern getrennten Netze über den VSwitch *ETHERNET0* verbunden. Dies ist nicht nur aus sicherheitstechnischen Aspekten bedenklich, sondern kann auch zu Überraschungen führen, wenn in den physikalisch getrennten Netzen gleiche VLAN-IDs benutzt werden.

Die vorhandenen virtuellen Ethernet Switches auf einem Managed System lassen sich mit „*ms lsvswitch*“ auflisten:

```
$ ms lsvswitch ms01
MS    VSWITCH              SWITCH_MODE  VLAN_IDS
ms01  ETHERNET0(Default)   VEB          100,200,300
$
```

Für jeden VSwitch werden auch die verfügbaren (konfigurierten) VLANs aufgelistet.

Um einen neuen VSwitch anzulegen dient das Kommando „*ms addvswitch*“, als Argument wird der Name des Managed Systems und der Name des neu anzulegenden VSwitches angegeben:

```
$ ms addvswitch ms01 ETHNAS
$
```

Der neue VSwitch heißt *ETHNAS* und wird ab jetzt natürlich auch aufgelistet:

```
$ ms lsvswitch ms01
MS    VSWITCH              SWITCH_MODE  VLAN_IDS
ms01  ETHERNET0(Default)   VEB          100,200,300
ms01  ETHNAS               VEB          none
$
```

Per Default wird ein neuer VSwitch im Mode VEB (Virtual Ethernet Bridge) angelegt. Der VSwitch funktioniert dann wie eine Ethernet Bridge, LPARs im gleichen Managed System können über den VSwitch direkt kommunizieren, ohne das Netzwerk-Pakete über einen SEA ins externe physikalische Netz gehen. Damit unterliegt der Netzwerk-Verkehr zwischen LPARs im gleichen Managed System, die sich im gleichen VLAN am gleichen VSwitch befinden keiner Firewall!

In der Betriebsart VEPA (Virtual Ethernet Port Aggregator) wird jeglicher Netzwerk-Verkehr ins externe Netz weitergeleitet. Ist das Ziel im gleichen Managed System wie der Sender, dann werden die Netzwerk-Pakete wieder zurück an das Managed System geleitet. LPARs im gleichen Managed System können damit nicht mehr direkt über den Hypervisor Netzpakete austauschen. Damit ist natürlich die Performance schlechter als in der Betriebsart VEB.

Beim Anlegen von virtuellen Ethernet Adaptern kann über die Option ‚*-s*‘ der gewünschte VSwitch angegeben werden.

Wird ein VSwitch nicht mehr benötigt, dann kann er mittels „*ms rmvswitch*“ wieder entfernt werden:

```
$ ms rmvswitch ms01 ETHNAS
$
```

Der VSwitch darf hierzu allerdings nicht mehr in Benutzung sein, ansonsten bekommt man die folgende Fehlermeldung:

```
$ ms rmvswitch ms01 ETHNAS
ERROR: msVswitch(): remote HMC command returned an error (1)
CMD on hmc01: chhwres -m ms01 -r virtualio –rsubtype vswitch -o r –vswitch ETHNAS
StdErr: HSCL3689 This virtual switch cannot be deleted since the following virtual networks
are using this virtual switch: VLAN100-ETHNAS,. All virtual networks using a virtual switch
must be deleted before the virtual switch can be deleted.
$
```

3. Verwalten von Partitions Daten

Die Partitions Profil-Daten können auf der HMC gesichert werden. Wurde ein LPAR gelöscht oder die Konfiguration einer LPAR geändert, dann kann mit Hilfe eines Backups die LPAR wieder hergestellt werden.

Um die Profil-Daten aller LPARs eines Managed Systems auf einer HMC zu sichern, dient das Kommando „*ms bkprofdata*“:

```
$ ms bkprofdata ms01 backup01
$
```

Als Argumente müssen das Managed System und ein Dateiname für das Backup angegeben werden. Wird ein relativer Pfad angegeben, dann wird das Backup auf der HMC unter */var/hsc/profiles/<serial-number>* abgespeichert, wobei *<serial-number>* die Seriennummer des Managed Systems ist.

Vorsicht: Es werden immer die Profil-Daten aller LPARs gesichert, das Backup kann nicht auf einzelne LPARs eingeschränkt werden!

Welche Backups von Profil-Daten es in dem genannten Standard-Verzeichnis schon gibt kann mit „*ms lsprofdata*“ aufgelistet werden:

```
$ ms lsprofdata ms01
backup
backup01
$
```

Auf dem Managed System *ms01* gibt es aktuell die folgenden LPARs, die auch alle in dem oben gemachten Backup enthalten sind:

```
$ lpar -m ms01 show
LPAR             ID  SERIAL    LPAR_ENV   MS    HMCs
lpar2            4   XXXXXXX4  aixlinux   ms01  hmc01,hmc02
lpar3            5   XXXXXXX5  aixlinux   ms01  hmc01,hmc02
ms01-vio1        2   XXXXXXX2  vioserver  ms01  hmc01,hmc02
ms01-vio2        3   XXXXXXX3  vioserver  ms01  hmc01,hmc02
$
```

Die LPAR *lpar3* wird nicht benötigt und ist auch aktuell nicht aktiv. Wir löschen diese, um anschließend einen Restore der Profil-Daten zu demonstrieren:

```
$ lpar delete lpar3
$ lpar -m ms01 show
LPAR             ID  SERIAL    LPAR_ENV   MS          HMCs
lpar2            4   XXXXXXX4  aixlinux   ms01  hmc01,hmc02
ms01-vio1        2   XXXXXXX2  vioserver  ms01  hmc01,hmc02
ms01-vio2        3   XXXXXXX3  vioserver  ms01  hmc01,hmc02
$
```

Nun wird das angelegte Backup verwendet, um die LPAR *lpar3* wieder zu restaurieren:

```
$ ms rstprofdata ms01 3 backup01
$
```

Es gibt die folgenden 4 Restore-Typen:

1 — voller Restore

2 — Merge der aktuellen Konfiguration mit dem Backup, bei Unterschieden wird das Backup verwendet

3 — Merge der aktuellen Konfiguration mit dem Backup, bei Unterschieden wird die aktuelle Konfiguration beibehalten

4 — Initializierung, alle Partitionen, Partitions-Profile und System-Profile werden gelöscht.

Durch den Restore wurde die LPAR *lpar3* wiederhergestellt, damit diese in den Zuordnungs-Dateien wieder aufgenommen wird, muss einmal das Kommando „*hmc rescan*“ gestartet werden:

```
$ hmc rescan
ms01
ms02
ms03
ms04
…
$
```

Die LPAR *lpar3* wurde wieder angelegt:

```
$ lpar -m ms01 show
LPAR            ID  SERIAL    LPAR_ENV   MS    HMCs
lpar2           4   XXXXXXX4  aixlinux   ms01  hmc01,hmc02
lpar3           5   XXXXXXX5  aixlinux   ms01  hmc01,hmc02
ms01-vio1       2   XXXXXXX2  vioserver  ms01  hmc01,hmc02
ms01-vio2       3   XXXXXXX3  vioserver  ms01  hmc01,hmc02
$
```

Wird ein Backup nicht mehr benötigt, kann es mittels „*ms rmprofdata*“ wieder gelöscht werden:

```
$ ms rmprofdata ms01 backup01
$
```

Vorsicht: Ein Restore kann sich auf alle LPARs auswirken, auch wenn dies im Beispiel oben nicht gezeigt wurde. In einem Backup sind in der Regel immer die Profil-Daten von mehreren LPARs enthalten.

8. HMC

Alle HMC Benutzer-Accounts können mit Hilfe des LPAR-Tools verwaltet werden. Jedem HMC Benutzer ist eine Resource Rolle und eine Task Rolle zugeordnet. Die Resource Rolle bestimmt welche Managed Systems und LPARs der Benutzer verwalten kann. Managed Systems und LPARs die seiner Resource Rolle nicht zugeordnet sind, sind für ihn nicht sichtbar. Dies trifft für die Kommandozeile (und damit auch das LPAR-Tool), als auch für das GUI zu. Die Task Rolle legt fest welche Operationen der Benutzer auf Managed Systems, LPARs und HMC durchführen darf.

1. Benutzer Accounts

Zunächst zeigen wir, wie mit Hilfe des LPAR-Tools Benutzer Accounts aufgelistet, angelegt, gelöscht und geändert werden können.

Eine Liste der aktuell angelegten Benutzer auf einer HMC bekommt man mit dem Kommando „*hmc lshmcusr*“:

```
$ hmc lshmcusr hmc01
NAME      DESCRIPTION     TASKROLE        RESOURCEROLE
hscroot   HMC Super User  hmcsuperadmin   ALL:
lpar2rrd  technical user  hmcviewer       ALL:
operator  Operators       firstlevel      ALL:
kmeier    Klaus Meier     hmcsuperadmin   ALL:
…
$
```

Die Resource Rolle „ALL:“ umfaßt alle Managed Systems und LPARs.

Ein neuer Benutzer kann mit dem Kommando „*hmc mkhmcusr*“ angelegt werden:

```
$ hmc mkhmcusr hmc01 testuser
Enter the new password: XXXXXXXX
Retype the new password: XXXXXXXX
$ hmc lshmcusr hmc01
NAME      DESCRIPTION     TASKROLE        RESOURCEROLE
hscroot   HMC Super User  hmcsuperadmin   ALL:
lpar2rrd  technical user  hmcviewer       ALL:
operator  Operators       firstlevel      ALL:
kmeier    Klaus Meier     hmcsuperadmin   ALL:
…
testuser                  hmcviewer       ALL:
$
```

Der neue Benutzer bekommt die Task Rolle „*hmcviewer*“ und die Resource Rolle „*ALL:*“ zugewiesen.

Die Attribute eines Benutzers, wie zugeordnete Task Rolle oder Resource Rolle können natürlich ebenfalls leicht mit dem LPAR-Tool geändert werden. Hierzu gibt es das Kommando „*hmc chhmcusr*“:

```
$ hmc chhmcusr hmc01 testuser taskrole=hmcsuperadmin
$ hmc lshmcusr hmc01
NAME      DESCRIPTION     TASKROLE        RESOURCEROLE
hscroot   HMC Super User  hmcsuperadmin   ALL:
lpar2rrd  technical user  hmcviewer       ALL:
operator  Operators       firstlevel      ALL:
kmeier    Klaus Meier     hmcsuperadmin   ALL:
```

```
…
testuser                  hmcsuperadmin  ALL:
$
```

Genauso einfach läßt sich auch die Resource Rolle eines Benutzers ändern:

```
$ hmc chhmcusr hmc01 testuser resourcerole=testonly
$ hmc lshmcusr hmc01
NAME      DESCRIPTION     TASKROLE        RESOURCEROLE
hscroot   HMC Super User  hmcsuperadmin   ALL:
lpar2rrd  technical user  hmcviewer       ALL:
operator  Operators       firstlevel      ALL:
kmeier    Klaus Meier     hmcsuperadmin   ALL:
…
testuser                  hmcsuperadmin   testonly
$
```

Natürlich muss die Resource Rolle „*testonly*“ auch existieren!

Sollte ein Benutzer Account nicht mehr benötigt werden, dann kann er gelöscht werden:

```
$ hmc rmhmcusr hmc01 testuser
$
```

2. Administration von Authorized Keys

Ab der Version 1.4.2.0 lassen sich Authorized Keys von HMC Benutzern ganz leicht mit dem *LPAR-Tool* verwalten.

Mit dem Kommando „*hmc lsauthkeys*“ lassen sich alle Public Keys aus der Datei *~/.ssh/authorized_keys2* anzeigen. Wird kein Benutzername angegeben, so wird die eigene *authorized_keys2* ausgegeben:

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

$
```

In der angezeigten *authorized_keys2* gibt es aktuell nur einen RSA-Key. Der Key selbst ist hier nicht vollständig dargestellt. Der Key hat den Kommentar „*user01*“. Soll ein weiterer Key hinzugefügt werden, kann das Kommando „*hmc addauthkeys*“ verwendet werden. Der zuzufügende Key muss in Anführungszeichen gesetzt werden, da er Leerzeichen enthält:

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

$
```

Es wurde ein weiterer RSA-Key, mit dem Kommentar „*testuser*“, hinzugefügt. Eine kurze Überprüfung mit „*hmc lsauthkeys*“ zeigt, das der Key wirklich eingetragen wurde:

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

```
$
```

Genauso einfach kann ein Key auch wieder entfernt werden! Hierzu gibt man beim Kommando „*hmc rmauthkeys*“ entweder den kompletten Key an (ganze Zeile) und natürlich wieder in Anführungszeichen oder, alternativ, kann auch der Kommentar des Keys angegeben werden:

```
$ hmc rmauthkeys hmc01 "ssh-rsa AAAAkasjkjaksjf testuser“
$
oder
$ hmc rmauthkeys hmc01 testuser
$
```

Eine Überprüfung zeigt, dass der Key nicht mehr vorhanden ist:

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

$
```

Hat der eigene Account ausreichende Berechtigungen (siehe Task Rollen), dann kann auch auf die *authorized_keys2* von anderen Benutzern zugegriffen werden. Der Benutzername wird einfach als zusätzliches Argument nach der HMC angegeben, hier z.B. der Benutzer *user15*:

```
$ hmc lsauthkeys hmc01 user15
LINE

$
```

Offensichtlich hat der Benutzer *user15* aktuell noch keinen Public Key eingetragen. Wir fügen einen Public Key hinzu:

```
$ hmc addauthkeys hmc01 user15 „ssh-rsa XXXXXXXXXXXXXX user15“
$
```

Der neue Key taucht in der *authorized_keys2* von Benutzer *user15* auf:

```
$ hmc lsauthkeys hmc01 user15
LINE
ssh-rsa XXXXXXXXXXXXXX user15

$
```

Allerdings kann nur der Benutzer selber einen Key wieder entfernen! Das Kommando „*hmc rmauthkeys*“ bietet aktuell nicht die Möglichkeit einen Benutzernamen anzugeben. (Dies liegt letztlich daran das dies auf der HMC CLI nicht unterstützt ist.)

3. Resource Rollen

Mit Hilfe von Resource Rollen kann eingeschränkt werden, welche Managed Systems und LPARs ein Benutzer administrieren darf. Managed Systems und LPARs die nicht zur Resource Rolle des Benutzers gehören, sind für den Benutzer nicht sichtbar. D.h. diese werden weder im GUI, noch auf der Kommandozeile angezeigt.

Standardmäßig existiert nur die fest-eingebaute und nicht veränderbare Resource Rolle „*ALL:*“. Diese kann zwar verwendet werden, ist aber für viele Kommandos nicht sichtbar. Weitere Resource Rollen können nach belieben angelegt, konfiguriert und Benutzern zugeordnet werden.

Eine Liste der existierenden Resource Rollen auf einer HMC bekommt man mit dem Kommando „*hmc lsresourceroles*“:

```
$ hmc lsresourceroles hmc01
NAME
role01
role02
ms01only
testonly
$
```

Die eingebaute Resource Rolle „*ALL:*“ wird wie oben schon angedeutet nicht angezeigt. Nur Resource Rollen die auch geändert werden können, werden angezeigt.

Möchte man die Ressourcen sehen, die sich hinter einer Resource Rolle verstecken, so ist das Kommando „*hmc lsresourcerole*“ zu verwenden. Die anzuzeigende Resource Rolle wird einfach als Argument nach der HMC angegeben:

```
$ hmc lsresourcerole hmc01 testonly
RESOURCEROLE: testonly
cec:ms01
lpar:all:ms01
lpar:testdb01
lpar:testdb02
$
```

Um eine neue Resource Rolle hinzuzufügen, kann das Kommando „*hmc mkresourcerole*“ verwendet werden:

```
$ hmc mkresourcerole hmc01 myrole
$
```

Die Resource Rolle ist nach dem Anlegen leer:

```
$ hmc lsresourcerole hmc01 myrole
RESOURCEROLE: myrole
$
```

Das Ändern einer bestehenden Resource Rolle kann mit dem Kommando „*hmc chresourcerole*“ durchgeführt werden. Hierbei gibt es eine Reihe von verschiedenen Möglichkeiten.

Hinzufügen eines Managed Systems:

```
$ hmc chresourcerole hmc01 myrole +cec:ms02
$
```

Ist der Name des Managed Systems eindeutig, d.h. nicht gleichzeitig auch der Name einer LPAR, dann kann auch die Kurzform:

```
$ hmc chresourcerole hmc01 myrole +ms02
$
```

verwendet werden.

Das hinzufügen einer LPAR funktioniert ähnlich:

```
$ hmc chresourcerole hmc01 myrole +lpar:lpar1
$
 oder kürzer
$ hmc chresourcerole hmc01 myrole +lpar1
$
```

Sollen alle LPARs eines Managed Systems angesprochen werden, geschieht dies mit:

```
$ hmc chresourcerole hmc01 myrole +lpar:all:ms02
$
```

Natürlich können analog, durch Voranstellen eines Minus-Zeichens, Managed Systems oder LPARs auch wieder aus einer Resource Rolle entfernt werden:

```
$ hmc chresourcerole hmc01 testrole -ms04
$ hmc chresourcerole hmc01 testrole -cec:ms04
$ hmc chresourcerole hmc01 testrole -lpar:lpar3
$ hmc chresourcerole hmc01 testrole -lpar3
$ hmc chresourcerole hmc01 testrole -lpar:all:ms03
```

Nicht mehr benötigte Resource Rollen können ganz einfach gelöscht werden und zwar mit „*hmc rmresourcerole*“:

```
$ hmc rmresourcerole hmc01 testrole
$
```

4. Task Rollen

Mit Task Rollen lässt sich konfigurieren welche Operationen ein Benutzer auf Managed Systems, LPARs oder HMC durchführen darf.

Standardmäßig sind schon einige Task Rollen auf jeder HMC vordefiniert. Es können beliebige weitere Task Rollen angelegt werden. Die aktuell existierenden Task Rollen können mit dem Kommando „*hmc lstaskroles*“ aufgelistet werden:

```
$ hmc lstaskroles hmc01
NAME           PARENT
hmcdev         Predefined
hmcsuperadmin  Predefined
hmcviewer      Predefined
$
```

Jede Task Rolle kann im Detail aufgelistet werden:

```
$ hmc lstaskrole hmc01 hmcviewer
taskrole: hmcviewer
parent: Predefined
resources:
  frame
    CheckPSN
    ListFrameProperty
  cec
    CaptureSystemTemplate
    CollectCECVPDInfo
    LSProfileSpace
    ListCECProperty
    ListCoDInformation
    ListPCIeTopology
    ListRioTopology
    ListSSP
    ListSystemProfileProperty
    ListUtilizationData
    ListVETInfo
    ManageSysProfile
    ViewDumps
    ViewPowerManagment
    ViewSSP
  lpar
    CapturePartitonTemplate
    ListLPARProperty
    ListProfileProperty
    ManageProfile
  HMCConsole
    ChangeLocale
    ChangeUserPasswords
    CollectVPDInfo
    ListCloudConnServiceSettings
    ListConnections
    ListHMCConfiguration
    ListHMCEncrTask
    ListSNMPServiceableEvents
    ListStorageMedia
    TemplateLibrary
    TipOfTheDay
    UserSettings
    ViewConsoleEvents
    ViewHMCFileSystems
$
```

Jede Task Rolle hat einen Parent. Die Ressourcen der Parent Task Rolle beschränken welche Ressourcen einer Resource Rolle zugeordnet werden können. Dies verhindert eine Eskalation der Privilegien.

Eine neue Task Rolle kann mit dem Kommando „*hmc mktaskrole*“ erzeugt werden:

```
$ hmc mktaskrole hmc01 trole1 hmcviewer
```

```
$ hmc lstaskrole hmc01 trole1
taskrole: trole1
parent: hmcviewer
resources:
$
```

Die neue Task Rolle ist zunächst leer. Es können nur Ressourcen (Operationen) zur neuen Task Rolle hinzugefügt werden, die in der Task Rolle *hmcviewer* enthalten sind.

Ressourcen können einer Task Rolle mit dem Kommando „*hmc chtaskrole*“ hinzugefügt werden:

```
$ hmc chtaskrole hmc01 trole1 +lpar:ListLPARProperty
$ hmc chtaskrole hmc01 trole1 +cec:ListCECProperty
$ hmc lstaskrole hmc01 trole1
taskrole: trole1
parent: hmcviewer
resources:
  cec
    ListCECProperty
  lpar
    ListLPARProperty
$
```

Natürlich kann man auch Ressourcen wieder aus einer Task Rolle entfernen:

```
$ hmc chtaskrole hmc01 trole1 -cec:ListCECProperty
$ hmc chtaskrole hmc01 trole1 -lpar:ListLPARProperty
$
```

Nicht benötigte Task Rollen können mit Hilfe des Kommandos „*hmc rmtaskrole*“ gelöscht werden:

```
$ hmc rmtaskrole hmc01 trole1
$
```

5. Auf der HMC eingeloggte Benutzer

Benutzer können entweder mittels GUI oder CLI auf der HMC arbeiten. Aktionen die von eingeloggten Benutzern durchgeführt werden, können gelegentlich hängen. Dies kommt insbesondere beim Arbeiten mit dem GUI gelegentlich vor.

Per CLI eingeloggte Benutzer können mit dem Kommando „*hmc lslogon*“ angezeigt werden, inklusive der sogenannten Tasks die diese gestartet haben:

```
$ hmc lslogon hmc01
USER_NAME  TTY_ID  LOGON_TIME        ACCESS_LOCATION
   TASK_NAME TTY_ID  START_TIME          USER_NAME   PID
kmeier     pts/1   2018-05-29 16:17  10.11.12.13
   bash      pts/1   May 29 16:17:25 2018 root        20513
$
```

Sollte der Prozeß hängen, oder der eingeloggte Benutzer ist längere Zeit nicht verfügbar, kann der Prozeß (Task) beendet werden:

```
$ hmc termtask hmc01 20513
$
```

Das angegebene Argument ist die PID des zu beendenden Prozesses.

Auch die per GUI eingeloggten Benutzer können mit „*hmc lslogon*“ aufgelistet werden, hierbei muß die Option „*-r webui*“ angegeben werden:

```
$ hmc lslogon -r webbi hmc01
USER_NAME  SESSION_ID  LOGON_TIME           LOGON_MODE
   TASK_ID  TASK_NAME  SESSION_ID  START_TIME           USER_NAME
kmeier     3           04/26/2018 12:54:28  Enhanced+
   156      ms01       3           04/26/2018 12:59:13  kmeier
$
```

Der Default für die Option *-r*, falls nicht angegeben, ist „*-r ssh*“.

Auch ein Task auf der GUI lässt sich mit dem Kommando „*hmc termtask*“ beenden, hierzu muss die angezeigte Task ID als Argument angegeben werden:

```
$ hmc termtask -r webui hmc01 156
$
```

9. Fortgeschrittene Virtualisierung

In diesem Kapitel werden einige fortgeschrittene Themen der *POWER* Virtualisierung beschrieben, wie z.B. *SR-IOV* und *vNIC*s.

1. Konfiguration von SR-IOV

In diesem Abschnitt soll die Konfiguration von *SR-IOV* mit dem *LPAR-Tool* gezeigt werden.

Bei der klassischen POWER Virtualisierung erfolgt die Virtualisierung über das Zusammenspiel von POWER Hypervisor und Virtual-I/O-Server. Über den POWER Hypervisor werden I/O- und Netzwerk-Zugriffe an den Virtual-I/O-Server weitergeleitet. Dieser besitzt alle notwendigen Hardware-Ressourcen und leitet die Zugriffe an physikalische Hardware weiter. Dieses Konzept ist sehr flexibel und hat sich über viele Jahre bewährt. Nachteile dieses Konzepts liegen aber auf der Hand:

- Alle Zugriffe gehen über einen Virtual-I/O-Server, was Ressourcen (CPU und RAM) kostet. Insbesondere Netzwerk-Traffic ist sehr CPU-intensiv, so daß einiges an CPU-Resourcen durch den oder die Virtual-I/O-Server verbraucht werden.

- Die Weiterleitung von Zugriffen an einen Virtual-I/O-Server bringt eine zusätzliche Verzögerung mit sich, so daß sich die Latenz von Zugriffen zwangsläufig erhöht.

Bei Verwendung von *SR-IOV* (Single Root I/O Virtualization) findet die Virtualisierung direkt auf der *PCI*-Karte statt. Ein Virtual-I/O-Server wird dann nicht mehr zwangsweise benötigt, was CPU- und RAM-Resourcen einspart. Außerdem reduziert sich die Latenz von Zugriffen, da keine Weiterleitung der Zugriffe an einen Virtual-I/O-Server erfolgt.

Im Folgenden zeigen wir die Konfiguration von *SR-IOV* an einem konkreten Beispiel. Unser Managed System *ms24* ist mit *SR-IOV* Karten bestückt. Die *SR-IOV* Karten eines Systems lassen sich mit dem Kommando „*ms lssriov*“ anzeigen:

```
$ ms lssriov ms24
PHYS_LOC                  SLOT_ID   CONFIG_STATE  SRIOV_STATUS  ADAPTER_ID  PHYS_PORTS
LOGICAL_PORTS
U78D3.001.XXXXXXX-P1-C4   21010020  dedicated     null          null        null
null
U78D3.001.XXXXXXX-P1-C6   2102001b  dedicated     null          null        null
null
U78D3.001.XXXXXXX-P1-C11  21020013  dedicated     null          null        null
null
U78D3.001.XXXXXXX-P1-C9   21010010  dedicated     null          null        null
null
$
```

Unser System ist mit 4 *SR-IOV* Karten bestückt, wie die obige Ausgabe zeigt. Die Karten befinden sich in den Slots *P1-C4*, *P1-C6*, *P1-C9* und *P1-C11*. Alle 4 Karten befinden sich noch im Konfigurations-Zustand (*CONFIG_STATE*) „*dedicated*“. Damit können die Karten jeweils genau einer LPAR zugewiesen werden, aber nicht für *SR-IOV* genutzt werden.

Die Karten müssen also zunächst einmal in den sogenannten „*Shared Mode*“ gebracht werden, welcher die Nutzung der Hardware-Virtualisierung überhaupt erst ermöglicht. Dies geschieht mit dem Kommando „*ms chsriov*“:

```
$ ms chsriov ms24 21020013 shared
$ ms chsriov ms24 2102001b shared
$
```

Als Argument wird die Slot-ID (*SLOT_ID*), sowie das Argument „*shared*“ oder „*dedicated*“ angegeben. Jeder *SR-IOV* Adapter bekommt eine eindeutige Adapter-ID (*ADAPTER_ID*) zugewiesen. Möchte man für einen bestimmten Adapter eine bestimmte Adapter-ID verwenden, dann kann die gewünschte Adapter-ID als zusätzliches Argument angegeben werden:

```
$ ms chsriov ms24 21010020 shared adapter_id=4
$ ms chsriov ms24 21010010 shared adapter_id=8
$
```

Wir schauen uns den Zustand der Karten kurz an:

```
$ ms lssriov ms24
PHYS_LOC                  SLOT_ID   CONFIG_STATE  SRIOV_STATUS  ADAPTER_ID   PHYS_PORTS
LOGICAL_PORTS
U78D3.001.XXXXXXX-P1-C4   21010020  sriov         initializing  unavailable  unavailable
unavailable
U78D3.001.XXXXXXX-P1-C6   2102001b  sriov         running       2            4
48
U78D3.001.XXXXXXX-P1-C11  21020013  sriov         initializing  unavailable  unavailable
unavailable
U78D3.001.XXXXXXX-P1-C9   21010010  sriov         missing       unavailable  unavailable
unavailable
$
```

Die Ausgabe zeigt, dass die Initialisierung einige Zeit dauern kann. Nachdem alle Karten initialisiert sind, ergibt sich die folgende Ausgabe:

```
$ ms lssriov ms24
PHYS_LOC                  SLOT_ID   CONFIG_STATE  SRIOV_STATUS  ADAPTER_ID  PHYS_PORTS
LOGICAL_PORTS
U78D3.001.XXXXXXX-P1-C11  21020013  sriov         running       1           4           48
U78D3.001.XXXXXXX-P1-C6   2102001b  sriov         running       2           4           48
U78D3.001.XXXXXXX-P1-C4   21010020  sriov         running       4           2           120
U78D3.001.XXXXXXX-P1-C9   21010010  sriov         running       8           2           120
$
```

Der *CONFIG_STATE* ist jetzt bei allen 4 Karten „*sriov*“ mit *SRIOV_STATUS* „*running*“. Die Adapter mit ID 1 und 2 haben jeweils 4 physikalische Ports, welche insgesamt 48 logische Ports unterstützen. Jeder logische Port kann genau einer LPAR zugewiesen werden. Die Adapter mit den IDs 4 und 8 haben nur 2 physikalische Ports, unterstützen aber 120 logische Ports. Wieviele logische Ports unterstützt werden, hängt letztlich immer von der verwendeten *SR-IOV* Adapter-Karte ab.

Als nächstes schauen wir uns die physikalischen Ports genauer an. Auch hier kann das Kommando „*ms lssriov*“ wieder verwendet werden, diesmal mit der Option „*-p*“ für physikalische Ports:

```
$ ms lssriov -p ms24
```

```
PHYS_PORT_LOC                 STATE  LABEL  TYPE  ADAPTER  PPORT  USED  MAX  CONN_SPEED  MTU
U78D3.001.XXXXXXX-P1-C11-T1   1      -      ethc  1        0      0     20   10000
9000
U78D3.001.XXXXXXX-P1-C11-T2   0      -      ethc  1        1      0     20   0
9000
U78D3.001.XXXXXXX-P1-C11-T3   0      -      eth   1        2      0     4    0
9000
U78D3.001.XXXXXXX-P1-C11-T4   0      -      eth   1        3      0     4    0
          9000
U78D3.001.XXXXXXX-P1-C6-T1    1      -      ethc  2        0      0     20   10000
9000
U78D3.001.XXXXXXX-P1-C6-T2    0      -      ethc  2        1      0     20   0
9000
U78D3.001.XXXXXXX-P1-C6-T3    0      -      eth   2        2      0     4    0
9000
U78D3.001.XXXXXXX-P1-C6-T4    0      -      eth   2        3      0     4    0
9000
U78D3.001.XXXXXXX-P1-C6-T4    0      -      eth   2        3      0     4    0
9000
U78D3.001.XXXXXXX-P1-C4-T1    0      -      roce  4        0      0     60   0
1500
U78D3.001.XXXXXXX-P1-C4-T2    0      -      roce  4        1      0     60   0
1500
U78D3.001.XXXXXXX-P1-C9-T1    0      -      roce  8        0      0     60   0
1500
U78D3.001.XXXXXXX-P1-C9-T2    0      -      roce  8        1      0     60   0
1500
$
```

Die physikalischen Ports eines Adapters sind jeweils mit 0 beginnend durchnummeriert. Einige der physikalischen Ports haben schon einen Link (*STATE*=1). In der Spalte *CONN_SPEED* findet man dann die Geschwindigkeit, hier jeweils 10 Gb/s (10.000 Mb/s), sowie die *MTU* Größe.

Man sollte den verwendeten physikalischen Ports einen Namen geben (Spalte *LABEL*). Dies ist wichtig, falls man später *vNIC* verwendet und eine Live Partition Mobility Operation durchführen möchte. Hierfür kann wiederum das Kommando „*ms chsriov*“ verwendet werden, es muss die Adapter-ID und physikalische Port-ID angegeben werden:

```
$ ms chsriov ms24 1 0 phys_port_label=ApplTier
$ ms chsriov ms24 2 0 phys_port_label=ApplTier
$
```

Wir haben für 2 der physikalischen Ports den Namen *ApplTier* vergeben, da diese im Application-Tier liegen:

```
$ ms lssriov -p ms24
PHYS_PORT_LOC                 STATE  LABEL     TYPE  ADAPTER  PPORT  USED  MAX  CONN_SPEED
MTU
U78D3.001.XXXXXXX-P1-C11-T1   1      ApplTier  ethc  1        0      0     20   10000
9000
U78D3.001.XXXXXXX-P1-C11-T2   0      -         ethc  1        1      0     20   0
9000
U78D3.001.XXXXXXX-P1-C11-T3   0      -         eth   1        2      0     4    0
9000
U78D3.001.XXXXXXX-P1-C11-T4   0      -         eth   1        3      0     4    0
9000
U78D3.001.XXXXXXX-P1-C6-T1    1      ApplTier  ethc  2        0      0     20   10000
9000
U78D3.001.XXXXXXX-P1-C6-T2    0      -         ethc  2        1      0     20   0
9000
U78D3.001.XXXXXXX-P1-C6-T3    0      -         eth   2        2      0     4    0
9000
```

```
U78D3.001.XXXXXXX-P1-C6-T4   0      -         eth   2        3      0     4    0
9000
U78D3.001.XXXXXXX-P1-C4-T1   0      -         roce  4        0      0     60   0
1500
U78D3.001.XXXXXXX-P1-C4-T2   0      -         roce  4        1      0     60   0
1500
U78D3.001.XXXXXXX-P1-C9-T1   0      -         roce  8        0      0     60   0
1500
U78D3.001.XXXXXXX-P1-C9-T2   0      -         roce  8        1      0     60   0
1500
$
```

Es gibt eine ganze Reihe weiterer Attribute die mittels „*ms chsriov*“ geändert werden. Diese kann man entweder der IBM Dokumentation oder der Online-Hilfe des Kommandos entnehmen:

```
$ ms help chsriov
USAGE:
   ms [-h <hmc>] chsriov [-v] <ms> {<slot_id> {dedicated|shared} | <adapter_id>
<phys_port_id> <attributes>} [<attributes> ...]

DESCRIPTION:

Switches an SR-IOV adapter in a managed system either to dedicated or shared mode,
or sets attributes for an SR-IOV physical port.

Attributes when switching an adapter to shared mode:
   adapter_id - 1-32, default: assign next available adapter ID

Attributes for an SR-IOV physical port:
   conn_speed - ethernet speed
      auto : autonegotation
      10 : 10 Mbps
      100 : 100 Mbps
      1000 : 1 Gbps
      10000 : 10 Gbps
      40000 : 40 Gbps
      100000 : 100 Gbps
   max_recv_packet_size - MTU
      1500 - 1500 byts
      9000 - 9000 bytes (jumbo frames)
   phys_port_label - label for the physical port
      1-16 characters
      none - to clear the label
   phys_port_sub_label - sublabel for the physical port
      1-8 characters
      none - to clear the sublabel
   recv_flow_control
      0 - disable
      1 - enable
   trans_flow_control
      0 - disable
      1 - enable
   veb_mode
      0 - disable virtual ethernet bridge mode
      1 - enable virtual ethernet bridge mode
   vepa_mode
      0 - disable virtual ethernet port aggregator mode
      1 - enable virtual ethernet port aggregator mode
   (see the IBM documentation for additional attributes)

Note: An SR-IOV adapter can only be changed to shared mode, if it is not assigned
to an LPAR!
```

```
EXAMPLES:

Switch adapter 21010010 of managed system ms01 to shared mode:
   ms chsriov ms01 21010010 shared

Switch adapter 21010010 of managed system ms01 back to dedicated mode:
   ms chsriov ms01 21010010 dedicated

Switch adapter 21010010 of managed system ms01 to shared mode using
adapter ID 3:
   ms chsriov ms01 21010010 shared adapter_id=3

$
```

Als nächstes können logische Ports erzeugt und den LPARs zugewiesen werden. Wir schauen uns zunächst einmal alle vorhandenen logischen Ports an. Hierfür gibt es die Option „*-l*“ (logischer Port) beim Kommando „*ms lssriov*“:

```
$ ms lssriov -l ms24
LOCATION_CODE  ADAPTER  PPORT  LPORT  LPAR  CAPACITY  CURR_MAC_ADDR  CLIENTS
$
```

Da die Adapter-Karten gerade erst auf *SR-IOV Shared*-Mode umgestellt wurden, gibt es natürlich noch keine logischen Ports. Ein logischer Port wird einer LPAR zugewiesen, daher muss das Kommando *lpar* verwendet werden. Das Hinzufügen eines logischen Ports geht mit dem Kommando „*lpar addsriov*“:

```
$ lpar addsriov aix01 2 0 capacity=50
$
```

Im Beispiel wurde der LPAR *aix01* ein logischer Port mit einer garantierten Kapazität von 50% der Netzwerk-Bandbreite zugewiesen. Der logische Port gehört zum Adapter mit der ID 2 und dort dem physikalischen Port mit der ID 0.

Eine kurze Überprüfung mit dem „*ms lssriov*“ Kommando zeigt den gerade erzeugten logischen Port:

```
$ ms lssriov -l ms24
LOCATION_CODE                  ADAPTER  PPORT  LPORT     LPAR   CAPACITY  CURR_MAC_ADDR
CLIENTS
U78D3.001.XXXXXXX-P1-C6-T1-S1  2        0      27008001  aix01  50.0      02606b8e4300   -
$
```

Natürlich kann man auch alle logischen *SR-IOV* Ports einer LPAR anzeigen lassen, da es um eine LPAR geht, ist hierfür das Kommando „*lpar lssriov*“ verfügbar:

```
$ lpar lssriov aix01
LPORT     REQ  ADAPTER  PPORT  CONFIG_ID  CAPACITY  MAX_CAPACITY  PVID  VLANS
CURR_MAC_ADDR  CLIENTS
27008001  Yes  2        0      0          50.0      100.0         0     all
02606b8e4300   -
$
```

Der logische *SR-IOV* Port wurde beim Hinzufügen zur LPAR auch im Profil der LPAR eingetragen. Das Kommando „*lpar lssriov*“ kann mit der Option „*-p*“ und einem Profil-Namen gestartet werden, um die Informationen aus dem Profil anzuzeigen:

```
$ lpar -p standard lssriov aix01
LPORT     ADAPTER  PPORT  CONFIG_ID  CAPACITY  MAX_CAPACITY  PVID  VLANS  MAC_ADDR
CLIENTS
27008001  2        0      0          50.0      100.0         0     all    02606b8e4300  -
$
```

Natürlich lassen sich auch bei einem logischen Port verschiedene Attribute, wie z.B. *PVID* oder zusätzliche *VLAN*s ändern. Hierfür kann das Kommando „*lpar chsriov*“ verwendet werden. Die möglichen änderbaren Attribute kann der Online-Hilfe zu dem Kommando oder der IBM Dokumentation entnommen werden:

```
$ lpar help chsriov
USAGE:
   lpar [-h <hmc>] [-m <ms>] [-p <profile>] chsriov [-d] [-f] [-l <detail_level>] [-w
<wait_time>] [-v] <lpar> <config_id>|<logical_port_id> <attributes> ...

DESCRIPTION:

Changes attributes of an SR-IOV logical port.

  allowed_os_mac_addrs - allowed MAC addresses
   allowed_priorities - allowed QoS priorities
      none : no priorities allowed (default)
      0-7: comma-separated list of integers
      all : all priorities are allowed
   allowed_vlan_ids - allowed VLANs
      none : no VLAN IDs allowed
      VLAN_IDs: comma-separated list of VLAN IDs
      all : all VLAN IDs are allowed (default)
   diag_mode - diagnostics mode
      0 : disable (default)
      1 : enable
   port_vlan_id - VLAN-ID for untagged packets or 0 to disable VLAN tagging
   pvid_priority - priority of the PVID

EXAMPLES:

Set the PVID of logical port with config_id 1 of LPAR aix01 to 1200:
   lpar chsriov aix01 1 port_vlan_id=1200

Set the PVID and PVID priority of logical port 2700c006 of LPAR aix01 dynamically only:
   lpar chsriov -d aix01 2700c006 port_vlan_id=1200 pvid_priority=0

Set the PVID priority of logical port with config_id 1 in the profile 'standard'
of LPAR aix01 to 2:
   lpar -p standard chsriov aix01 1 pvid_priority=2

$
```

2. Konfiguration von vNIC

to be done.

10. Troubleshooting

Sollten Fehlermeldungen bei der Benutzung des LPAR-Tools auftreten, dann stellt sich natürlich die Frage nach der Ursache des Fehlers. Im allgemeinen können Fehler einer der folgenden Klassen zugeordnet werden:

- Ein Kommando wurde nicht korrekt benutzt: z.B. falsche Schreibweise, falsche Syntax, falsche Option, fehlende Argumente oder ähnliches. In diesem Fall sollte ein Fehlermeldung ausgegeben werden. Das Kommando sollte korrigiert und dann noch einmal gestartet werden.
- Ein Kommando wurde korrekt benutzt, aber die ausführende HMC liefert einen Fehler zurück.
- Das LPAR-Tool funktioniert nicht korrekt und liefert eine Fehlermeldung.

Alle Aufrufe des LPAR-Tools werden in der Datei *~/lpar.log* protokolliert, bei Fehlern wird auch die Fehlermeldung in der Log-Datei festgehalten.

1. Inkorrekte Benutzung von Kommandos

Wird ein Kommando nicht korrekt benutzt, so wird vom LPAR-Tool eine Fehlermeldung zusammen mit einer Usage-Meldung ausgegeben.

```
$ lpar lsvslot
ERROR: argument missing
   Usage: lpar [option …] keyword [arg …]
          lpar -V
…
$
```

Das LPAR-Tool gibt eine Fehlermeldung aus:

„*ERROR: argument missing*“.

Häufig wird auch ein LPAR-Name oder Managed System Name falsch geschrieben:

```
$ lpar addprocs testlpar7 1
ERROR: LPAR ‚testlpar‘ not found
$
```

2. HMC liefert Fehlermeldung

Auch bei korrekter Benutzung von Kommandos können Fehler auftreten.

```
$ ms lssriov ms03
ERROR: remote HMC command returned an error (1)
CMD on hmc01: lshwres -m ms03 -r sriov —rsubtype adapter
StdErr: HSCL1237 The managed system does not support SR-IOV.
$
```

Das Kommando *ms* wurde korrekt verwendet, allerdings unterstützt das angegebene Managed System *ms03* kein SRIOV. Das LPAR-Tool gibt die Fehlermeldung

„*ERROR: remote HMC command returned an error (1)*“

aus. Auf der zugehörigen HMC (*hmc01*) wurde das Kommando

```
lshwres -m ms03 -r sriov —rsubtype adapter
```

mit korrekter Syntax abgesetzt. Das LPAR-Tool gibt die Fehlermeldung des HMC-Kommandos aus:

„*StdErr: HSCL1237 The managed system does not support SR-IOV.*“

Es liegt letztlich kein Fehler des LPAR-Tools vor, das angegebene Managed System unterstützt kein SRIOV.

Hier ein Beispiel für einen Fehler im LPAR-Tool (dieser ist allerdings in der aktuellen Version schon behoben):

```
$ lpar lsvslot lpar7
ERROR: remote HMC command returned an error(1)
CMD on hmc02: lshwres -m ms11 -r virtualio —rsubtype vnic —level lpar —filter
lpar_names=lpar7
StdErr: The command entered is either missing a required parameter or a parameter value is
invalid. The parameters that are missing or have an invalid value are —rsubtype. Please
check your entry and retry the command.
$
```

Auch hier wurde das Kommando korrekt benutzt (in diesem Fall *lpar*). Allerdings ist in diesem Fall das abgesetzte Kommando auf der HMC fehlerhaft und damit lassen sich die virtuellen Slots der LPAR *lpar7* nicht anzeigen. Das Problem ist in diesem Fall eine ältere HMC-Version, die keinen Support für vNIC hat und daher den Wert „*vnic*“ bei der Option „*—rsubtype*“ nicht kennt. Das Problem wurde durch Einbau eines Checks behoben.

Sollte also das LPAR-Tool ein HMC-Kommando erzeugen, welches nicht korrekt ist (ungültige Option oder Argument), dann liegt vermutlich ein Fehler im Programm-Code des LPAR-Tools vor. Das Problem kann dann an *support@powercampus.de* gemeldet werden. Die verwendete HMC-Version, sowie die Ausgabe des Kommandos inklusive Fehlermeldung sollte dann zugesendet werden.

3. Fehler des LPAR-Tools

Keine Software ist perfekt und fehlerfrei. Wir haben das LPAR-Tool intensiven Tests unterzogen, trotzdem lässt sich nicht ausschließen das es im LPAR-Tool Fehler gibt. Hier ein konstruierter Fehler, um zu demonstrieren wie sich ein solcher noch nicht bekannter Fehler äußern kann:

```
$ lpar lsrefcode lpar5
ERROR: tableGetRow(): index out of range
$
```

Um Abstürze der LPAR-Tool Kommandos zu vermeiden, überprüfen die meisten verwendeten Funktionen die Argumente mit denen sie aufgerufen wurden. Wird ein fehlerhaftes Argument entdeckt, bricht das Kommando mit einer Fehlermeldung und dem Hinweis auf die Funktion die den Fehler entdeckt hat ab:

„*ERROR: tableGetRow(): index out of range*“

Hier wurde eine Funktion namens *tableGetRow()* aufgerufen, diese hat bei der Überprüfung ihrer Argumente festgestellt, das der mitgegebene Index außerhalb des Gültigkeitsbereichs ist. Fehlermeldungen bei denen ein Funktionsname mitgegeben wird, weisen in der Regel auf einen Fehler in einem der LPAR-Tools hin. Auch in diesem Fall sollte der Fehler an *support@powercampus.de* zusammen mit der Bildschirmausgabe gemeldet werden.

A. Konfigurationsparameter

Über die Dateien */opt/pwrcmps/etc/lpar.cfg* und *~/.lpar.cfg* kann das LPAR-Tool konfiguriert werden. Die System-weite Konfigurationsdatei (falls vorhanden) wird dabei zuerst eingelesen. Die benutzerspezifische Konfigurationsdatei (falls vorhanden) wird im Anschluß eingelesen. Ist ein Parameter in beiden Dateien gesetzt, so gilt die zuletzt eingelesene benutzerspezifische Konfiguration.

ConfigDirectory:

Dieser Parameter legt fest, wo die Dateien *hmc.list*, *ms.list* und *lpar.list* angelegt werden.

Default: im Homeverzeichnis des Benutzers.

LicenseFile:

Der Parameter legt fest wo sich die Lizenz-Datei befindet und wie sie heißt.

Default: */opt/pwrcmps/etc/lpar.lic* und *~/.lpar.lic*.

ControlPersist:

Gibt an wieviele Minuten ein SSH-Master-Verbindung bei Inaktivität bestehen bleiben soll.

Default: 5 Minuten.

ServerAliveInterval:

Zeitintervall innerhalb dessen eine Alive-Meldung vom Server erwartet wird.

Default: 10 Sekunden.

ServerAliveCountMax:

Maximale Anzahl ausstehender Alive-Meldungen des Servers.

Default: 2.