HMC

Links

ezh (no longer maintained, see eezh below)

eezh

System

List managed frames

```
lssyscfg -r sys -F name:type_model:serial_num
```

List free processor/memory/adapters on specific frame/lpar

Proc

```
lshwres -r proc -m <framename> --level sys
lshwres -r proc -m <framename> --level lpar
```

Memory

```
lshwres -r mem -m <framename> --level sys
lshwres -r mem -m <framename> --level lpar
```

Adapters

```
lshwres -r hca -m <framename> --level sys
lshwres -r hca -m <framename> --level lpar
```

List io resources in a frame

```
lshwres -r io --rsubtype unit -m <frame>
lshwres -r io --rsubtype bus -m <frame>
lshwres -r io --rsubtype slot -m <frame>
```

Update firmware from the HMC

```
updlic -m <frame> -o a -t sys -l latest -r mountpoint -d /home/hscroot/fw_update
```

LPAR

List managed lpars attached to a frame

```
lssyscfg -m <frame> -r lpar -F name
```

List profiles from a specific lpar

```
lssyscfg -r prof -m <frame> --filter "lpar_names=<lpar>"
```

List the state of an LPAR

```
lssyscfg -m <frame> -r lpar -F name,state
lssyscfg -m <frame> -r lpar -F name,state --filter "lpar_names=<lpar>"
```

Shutdown/restart lpar from HMC

Shutdown using the shutdown command on the client operating system

```
chsysstate -r lpar -o osshutdown –n <lpar> -m `<frame>
```

Delayed partition shut down

```
chsysstate -r lpar -o shutdown –n <lpar> -m <frame>
```

Immediate partition shutdown

```
chsysstate -r lpar -o shutdown --immed –n <lpar> -m <frame>
```

Immediate partition restart

```
chsysstate -r lpar -o shutdown --immed --restart –n <lpar> -m <frame>
```

Restart of a partition after initiating a dump

```
chsysstate -r lpar -o dumprestart –n <lpar> -m <frame>
```

Start lpar from HMC

Boot lpar

```
chsysstate -r lpar -m <frame> -o on -n <lpar> -f <profile>
```

Boot into normal mode

```
chsysstate -r lpar -m <frame> -o on -n <lpar> -b norm
```

Boot into SMS mode

```
chsysstate -r lpar -m <frame> -o on -n <lpar> -b sms -f <profile>
```

Boot into diagnostic with default bootlist

```
chsysstate -r lpar -m <frame> -o on -n <lpar> -b dd -f <profile>
```

Boot into diagnostic with stored bootlist

```
chsysstate -r lpar -m <frame> -o on -n <lpar> -b ds -f <profile>
```

Boot into open firmware OK prompt

```
chsysstate -r lpar -m <frame> -o on -n <lpar> -b of -f <profile>
```

List WWPN login status information of virtual fiber channel adapters

```
lsnportlogin -m <frame> --filter "profile_names=<profile>"
```

WWPN status possible values:

- `0` - WWPN is not activated
- `1` - WWPN is activated
- `2` - WWPN status is unknown

Login/Logout NPIV on a virtual fiber channel adapter

Login

```
chnportlogin -m <frame> -o login -p <lpar_name> -n <profile>
```

Logout

```
chnportlogin -m <frame> -o logout -p <partition_name>
```

List client LPAR WWPN's

```
lssyscfg -r prof -m <frame> -F virtual_fc_adapters --filter lpar_names=<client>
```

List all client WWPNs on a frame

```
lshwres -r virtualio --rsubtype fc -m <frame> --level lpar -F lpar_name,slot_num,wwpns --
header | grep -v null
```

List PowerVM/VIOS Physical WWPN's

```
lshwres -r io --rsubtype slotchildren -m <frame> -F wwpn --filter lpar_names=<lpar>
```

DLPAR client virtual fibre channel adapter with WWPNs

```
`chhwres -m <frame> -r virtualio -o a -p <lpar_name> --rsubtype fc -s 44 -
a "adapter_type=client,remote_lpar_name=<vios1>,remote_slot_num=210,\"wwpns=c050760
```

Add client virtual fibre channel adapter to profile (LPAR offline)

```
chsyscfg -m <frame> -r prof -i lpar_name=<lpar_name>,name=Normal,
\"virtual_fc_adapters=\"\"41/client/1/<vios1>/37/c050760a833b007c,c050760a833b007d/
0\"\",\"\"42/client/1/<vios1>/38/c050760a833b007e,c050760a833b007f/0\"\",\"\"61/client/2/
<vios2>/38/c050760a833b007a,c050760a833b007b/0\"\",\"\"62/client/2/<vios2>/37/
c050760a833b0078,c050760a833b0079/0\"\"\"
```

Rename LPAR

```
chsyscfg -r lpar -m <frame> -i "name=<lpar_name>,new_name=<new_lpar_name>"
```

Change default profile

```
chsyscfg -m <frame> -r lpar -i "name=<lpar>,default_profile=Normal"
```

List and find a specific LPAR MAC address

```
for FRAME in $(lssyscfg -r sys -F name); do lshwres -m "${FRAME}" -r virtualio --
rsubtype eth --level lpar -F lpar_name,mac_addr; done | grep -i D2FF6B2EC518
```

List all LPAR's with configured VLAN

```
for FRAME in $(lssyscfg -r sys -F name); do lshwres -m "${FRAME}" -r virtualio --
rsubtype eth --level lpar -F lpar_name,port_vlan_id; done | grep <vlan>
```

Miscellaneous

Change user password

```
chhmcusr -u <user> -t passwd
```

CoD trials

```
lscod -m <frame> -t cap -r proc -c trial
lscod -m <frame> -t cap -r mem -c trial
```

List DLPAR history

```
lssvcevents -t console -d 300 | grep <hostname>
```

Reclaim cores/memory from shutdown LPAR

```
chhwres -r proc -m <frame> -o s -p <lpar> --procunits 0
```

Reclaim all cores/memory from Not Activated LPARs on a frame

```
for FRAME in <frame>; do LPARS=$(lssyscfg -m ${FRAME} -r lpar -F name,state |
 grep "Not Activated" | cut -d',' -f1); for LPAR in ${LPARS}; do chhwres -r proc -m $
{FRAME} -o s -p ${LPAR} --procunits 0; done; done
```

Run command on every VIOS

```
for sys in $(lssyscfg -r sys -F name); do for vio in $(lssyscfg -r lpar -m "${sys}" -
F name,lpar_env | grep vioserver | cut -f 1 -d, | sort); do echo "${vio}"; viosvrcmd -m "$
{sys}" -p "${vio}" -c ioslevel; echo ""; done; done
```

Change padmin user attributes

```
command=$(printf "chuser -attr maxage=0 maxexpired=-1 padmin")
viosvrcmd -m <frame> -p <lpar_name> -c "${command}"
```

Scripts

lparwwpn

Script to list out an LPAR's WWPNs. Due to the nature of the restricted shell, you need to first source in the functions to use it. This takes functions from Brian Smith's ezh script, which I also recommend using.

For example, if you copy the script into your home directory as `lparwwpn`.

```
. lparwwpn
lparwwpn <lpar_name>
```

```
lparframelookup () {
	while read system; do
		while read lpar; do
			if [ "$lpar" = "$*" ]; then
				echo $system;
			fi
		done < <(lssyscfg -m "$system" -r lpar -F name)
	done < <(lssyscfg -r sys -F "name,state" | egrep "Standby|Operating" | cut -d, -f 1) |
tail -n 1
}

checklpar () {
	count=0;
	if [ -n "$1" ]; then
		while read system; do
			while read l; do
				if [ "$l" = "$1" ] ; then
					count=$((count +1))
				fi
			done < <(lssyscfg -m "$system" -r lpar -F name)
		done < <(lssyscfg -r sys -F "name,state" | egrep "Standby|Operating" | cut -d, -f 1)
		]=""
		if [ $count -eq 0 ]; then
			echo "ERROR:  LPAR not found: $1"
			return 1
		fi
		if [ $count -gt 1 ]; then
			echo "ERROR:  Multiple LPAR's with same name $1"
			return 2
		fi
	else
		unset input
		unset lpararray
		echo "Select LPAR: "
		while read system; do
```

```
                while read l; do
                        count=$((count +1))
                        printf "%5s.  %-20s %-20s\n" $count "$l" "`lssyscfg -r lpar -m
\"$system\" -F state --filter lpar_names=\"$l\"`"
                        lpararray[$count]="$l"
                done < <(lssyscfg -m "$system" -r lpar -F name)
        done < <(lssyscfg -r sys -F "name,state" | egrep "Standby|Operating" | cut -d, -f 1)
        echo
        while [ -z "${lpararray[$input]}" ]; do
                printf "Enter LPAR number (1-$count, \"q\" to quit): ";
                read input
                if [ "$input" = "q" -o "$input" = "Q" ]; then return 1; fi
        done
        lpar="${lpararray[$input]}"
        checklpar "$lpar" || return 2
    fi
}
runandecho () {
    printf "Running:  "; echo "$*" | sed 's/eval//'
    if [ "$EZHDEBUG" == "TRUE" ]; then
        echo "#EZHDEBUG=TRUE, command not run"
    else
        $*
    fi
}

lparwwpn () {
    lpar="$1"; checklpar "$lpar" && runandecho eval  lshwres -m \"`lparframelookup
$lpar`\" -r virtualio --rsubtype fc --level lpar --filter \"lpar_names=$lpar\" | cut -d '=' -f11 |
sed 's/\"//';
}
```