Utility "xskulker" V1.0

The tool "xskulker" has been designed as a replacement for the skulker script to remove files and also old empty directories from a USS directory structure. The routine is written in REXX completely and exploits USS APIs. This provides a better performance over the original script. It can be used efficiently in a z/OS UNIX sysplex sharing environment and only if the file system(s) containing the files is/are owned locally are searched for candidates to get deleted. Note also that NFS file systems are not allowed by default.

The routine must run in USS. A user starting the utility should be a permanent superuper or have read access to **BPX.SUPEPRUSER** in class **FACILITY**. In order to control that the REXX does not use too much CPU resources you should set a jobname of **XSKULKER** during runtime and define a WLM rule to run an USS process with jobname XSKULKER with no high service class. If you want you can define to use just BATCHLOW.

To achieve jobname XSKULKER is used you can set the environment variable **_BPX_JOBNAME=XSKULKER** and the user running the utility must be a superuser or have read access to **BPX.JOBNAME** in class **FACILITY**.

You can also reduce the service class dynamically by overwriting the service class shown on the SDSF DA panel for xskulker or you use the system command "reset" directly based on the message shown in the syslog/operlog.

```
BPXF024I (HERING) XSKUL001I Running with PID 196906 ...  196
Use "RESET XSKULKER,SRVCLASS=BATCHLOW,A=0075"
to reset XSKULKER to service class BATCHLOW, for example.
```

The utility is provided in ASCII (ccsid 819 or code page ISO8851-1) in a ZIP file. To install it perform the following steps.

- Transfer the file binary to your home directory in USS, for example.
- Choose a good location for placing the utility within the USS file system structure like directory "/usr/local/bin".
- Run the following commands to complete the installation.

  ```
  $> su
  #> iconv -f819 -t1047 $HOME/xskulker > /usr/local/bin/xskulker
  #> chmod 755 /usr/local/bin/xskulker
  #> extattr -s /usr/local/bin/xskulker
  #> exit
  $>
  ```

- Now you are ready to use xskulker.

The syntax for using "xskulker"

The syntax for using "xskulker" is shown next via calling it from z/OS UNIX without parameters.

```
$> xskulker
Syntax: xskulker [ options ] dir_name days_old

-l logfile -- USS file to write all messages and output data to
-f -- Delete or list files to be deleted
-d -- Delete or list directories to be deleted if empty
-s nn -- Only search nn directory sub-levels; 0 means the start directory only
-x -- Only search within the initial file system
-delete -- Delete entries that are not accessed since more than "days_old" days
dir_name -- Directory name to start searching
days_old -- Number of days with no access to an entry

For more details see the documentation as provided.
$>
```

Important notes...

- If option "-f" is used and files have not been accessed for more than "days_old" days they are deleted if option "-delete" has been specified as well. Otherwise they are shown as candidates for deletion.
- Directories are deleted based on their modification time stamp not the access time stamp. Otherwise they would hardly get deleted. And they can only get deleted when they are empty.
- If you are using USS file system sharing, you must start the routine on that system that owns the corresponding file system. Otherwise you would see a message saying that a file system managed by another system is being skipped.
- If option "-l" with a valid logname is used all STDOUT and STDERR data is written to that logfile as well.

Using "xskulker" from within a batch job (according to ccsid 1141)

Following a BPXBATSL based job that can be nicely used to run "xskulker"...

```
//XSKULKER EXEC PGM=BPXBATSL,PARM='PGM /bin/sh -c $STDIN!$ICONV!sh'
//STDIN    DD DATA,DLM=##
xskulker -l /u/hering/skulker.log -f -d -x -delete \
/u/hering/test.access 100
##
//STDENV   DD DATA,DLM=##
STDIN=/bin/cat //dd:STDIN
ICONV=/bin/iconv -f1141 -t1047
_BPX_SHAREAS=YES
_BPX_JOBNAME=XSKULKER
PATH=/usr/local/bin:/bin
##
//STDOUT   DD SYSOUT=*,LRECL=2048,RECFM=VB
//STDERR   DD SYSOUT=*,LRECL=2048,RECFM=VB
```

Notes...

- The exclamation marks "!" in the EXEC card are the replacement characters for the 1047 pipe sign "|". You must use the right character for your ccsid used in TSO/ISPF.
- Based on the ICONV specification in STDENV you can use all the characters in STDIN as if you were editing according to ccsid 1047. You must specify the right ccsid for your environment.
- If you do not know it, go to ISPF "Dialog Test" and select "Variables". Then enter command "I ztermcp5" and you should see the actual value as known by ISPF.

Possible output in STDOUT...

```
Deleted directory /u/hering/test.access/newdir
Directory not empty; cannot delete /u/hering/test.access/testdir
Directory not empty; cannot delete /u/hering/test.access/test.convert
Directory not empty; cannot delete /u/hering/test.access/subdir
Deleted file /u/hering/test.access/subdir/testfile
Deleted file /u/hering/test.access/test.convert/testfile
Deleted file /u/hering/test.access/testdir/testfile
```

Possible output in STDERR...

```
Parameter list: "-l" "/u/hering/skulker.log" "-f" "-d" "-x" "-delete"
```

```
"/u/hering/test.access" "100"
2020-09-08 12:43:38 Starting ...
2020-09-08 12:43:38 Processing SC70 ZFS file system HERING.TEST.ZFS ...
2020-09-08 12:43:38 Stopping ...
xskulker on SC70    : Processing ended.
```

Using "xskulker" from within a crontab entry

Following a suggested command to use xskulker in a crontab entry...

```
_BPX_JOBNAME=XSKULKER /usr/local/bin/xskulker -l /u/hering/skulker.log -f -d -x
-delete /u/hering/test.access 100 >/dev/null 2>&1
```

Notes...

- The command shown should be placed completely within one long line in a crontab together with needed header information to define when the command is to be executed.
- All the output data is written only to the logname as specified.