z/TPF APAR Download Commands 1.1

NOTE: Before using this information and the product it supports, read
the general information under "NOTICES" in this document.

CONTENTS

This file includes the following information:

CHANGE HISTORY

03/20/2012 testcase-yellow.boulder.ibm.com changed to transfer.boulder.ibm.com
04/21/2013 update aparDownload for repository move
from http://tpf.ncsa.illinois.edu
to ftp://software.linux.ibm.com/pub/linuxpatch-submission/TPF
- add quotes around download package name to support special
characters in the URL names

```
           - remove ${HOST} from the wget retrieval
09/24/2013 update aparList, aparDownload, aparMerge to correct a processing
           error when the starting APAR is a cancelled APAR and has no link
           information in the first column
10/06/2015 update aparList, aparDownload, aparMerge to remove references
           to the critical and severity columns in the CSV file
01/09/2017 add aparListOpen
02/16/2017 Update aparDownload command to add csrfToken support
03/07/2017 Update aparDownload and aparMerge command for z/TPF JRE download
09/20/2017 Add aparPrereq command
10/23/2017 Add option to skip opensource package download
12/04/2018 Update download URLs in aparList, aparListOpen, aparPrereq
           for the new download server URL:
           from: http://www.ibm.com/software/htp/tpf/maint/files
             to: https://public.dhe.ibm.com/software/htp/tpf/maint/files
12/11/2018 Update aparDownload to handle TPF hosted opensource download files
12/24/2018 Correct unpack issue with opensource package to omit
           the APAR number folder from the unpack target folder
01/10/2019 Updated for "close group" nomenclature
09/23/2019 Updated for formatting changes in the TPF hosted opensource APAR
           download pages
10/02/2019 Updated for "Contact TPFQA" and no longer available
           opensource packages
07/08/2021 Add aparPkgcheck command
07/06/2022 Add -a to greps against the APAR CSV content to allow
           for non-printable characters that can cause grep to
           think the ascii file is binary in the following scripts:
           aparDownload, aparList, aparMerge, aparPkgcheck
12/05/2022 Updated aparPkgcheck and aparDownload to modify csrfToken
           extraction; download server software was updated
01/17/2023 Update workdir pathname to append a unique process child folder
           when user specifies -w option
```

1.0 Introduction

This README contains information about the following commands that run in the Linux environment:

- aparList
- aparDownload
- aparMerge
- aparChgperm

These commands can be used to query, download and unpack available (closed) z/TPF product APARs (including opensource APARs) and then merge them into a common directory.

- aparListOpen

This command can be used to query open z/TPF product APARs.

- aparPrereq

This command can be used to list prerequisite z/TPF product APARs for
a specified (closed) z/TPF APAR.

- aparPkgcheck

This command can be used to query whether an APAR tar package for an
APAR released before a given date was updated after that date, indicating
the package had been replaced on the server since that date. This can
be useful to identify if any APAR tar packages that were downloaded
before a given date, were updated after that date.

---

aparList

This command will:
- Download the z/TPF APAR Data CSV file from the z/TPF maintenance website.
- Optionally generate a report on a set of APARs in that file and send
  that report to specified email addresses and or save the report to a file.
- Optionally generate a log file that contains the first and last APAR
  processed and the date the aparList command was run.

The set of APARs processed can be defined by:
- The user and can include:
  - Closed APARs between starting and ending APAR numbers
  - Closed APARs between starting and ending close group numbers
    where a close group is a 1 or 2 digit number:
     1-15 for PUT1-PUT15
    19-99 for APARs closed in year 20xx
  - Closed APARs between a combination of APAR and close group
    numbers
- The log file generated by an earlier run.
  - When this is done, the set of APARs begins with the next APAR
    after the last APAR stored in the log and continues through
    the most recently closed APAR.

---

aparDownload

This command will download and unpack a set of APAR tar files.

The set of APARs processed can be defined by:
- The user and can include:
  - Closed APARs between starting and ending APAR numbers
  - Closed APARs between starting and ending close group numbers
    where a close group is a 1 or 2 digit number:
     1-15 for PUT1-PUT15
    19-99 for APARs closed in year 20xx
  - Closed APARs between a combination of APAR and close group
    numbers
- The log file generated by the aparList run.
  - When this is done, the set of APARs is defined by the first
    and last APAR listed in the log file.

This command requires the CSV File retrieved by the aparList command.

When a log file is used, if all APARs are successfully downloaded, the
log file is updated to include the date aparDownload command was run.

_____

aparMerge

_____

This command will merge a set of APAR packages in closure sequence
order under a user specified directory.

The set of APARs processed can be defined by:
- The user and can include:
  - Closed APARs between starting and ending APAR numbers
  - Closed APARs between starting and ending close group numbers
    where a close group is a 1 or 2 digit number:
     1-15 for PUT1-PUT15
    19-99 for APARs closed in year 20xx
  - Closed APARs between a combination of APAR and close group
    numbers
- The log file generated by the aparList run.
  - When this is done, the set of APARs is defined by the first
    and last APAR listed in the log file.

This command requires the CSV File retrieved by the aparList command
and the directory of unpacked APARs created by the aparDownload
command.

The APEDIT (PJnnnnn.txt) files found in the untar directory are NOT
moved to the merge directory during processing.

_____

aparChgPerm

_____

This command will update the group ownership and permissions on
files and directories in a merged APAR directory.

The user can define:
- The group name to apply to all files and directories.
- The user read/write permissions to be applied on all files
  and directories.
- The group read/write permissions to be applied on all files
  and directories.
- The other read/write permissions to be applied on all files
  and directories.

Execute permissions on directories and files is managed by the tool:
- for directories, if the permission specified for u,g,o
  is read or write, execute is also added to that level.
- for files, if the permission specified for u,g,o
  is read or write, execute is also added to that level only
  if it is for an executable file, like a tpftools script.

_____

aparListOpen

_____

This command will:
- Download the z/TPF APAR Data CSV file from the z/TPF maintenance website
  and generate a report on the set of open APARs in that file.
- Optionally include which APARs have been opened or closed since the
  last run.
- Optionally send that report to specified email addresses and or save
  the report to a file.
- Optionally generate a log file that contains the list of open APARs and
  the date the report was last run.

_____

aparPrereq

_____

This command will:
- Download the z/TPF APAR Data and Segment Prerequisite Data CSV files
  from the z/TPF maintenance website and generate the prereq information
  for the specified APAR. Two files are produced:
  - prereqApars.txt
    This file contains the list of prereq APARs and whether the APAR is
    applied, missing, or partially applied in the specified TPF_ROOT.
  - prereqApars_Files.txt
    This file contains the list of source files, object code only (OCO)
    files, and optionally, object and shared object files that are

found in all prereq APARs and whether the APAR update to each is applied or missing.

---

aparPkgcheck

This command will:
- Download the z/TPF APAR Data CSV file.
- Download the list of APAR tar packages available for download and the package date, which includes the current year and the four prior years.
- Given a user specified date, identify any APAR that was made available on or before that date, for which the corresponding tar file has a timestamp newer than that date.
- Issue a report to standard output, indicating if any APARs match the criteria.
  - If yes, both the tar file date and the availability date are shown and the script exits with RC 4.
  - If no, a message stating not APARs were found and the script exits with RC 0.

2.0 Installing the z/TPF APAR Download Tools

2.1 Prerequisites

To successfully run the z/TPF APAR Download Tools, you must have a valid z/TPF maintenance id and password.

These commands use the "wget" and "curl" commands to access web pages and retrieve files.

The aparPrereq command requires the tpfzdmap command, available for download on the TPF Family Products: Tools for z/TPF 1.1 & z/TPFDF 1.1 website (http://www.ibm.com/support/docview.wss?uid=swg27049596).

2.2 Unpack the tar file

To unpack aparDownload.tar.gz:

```
    cd /home/userid/<dirname>
    tar -xvzf APARDownload.tar.gz

  Once unpacked, the following files should be present:
    aparChgperm
    aparDownload
    aparList
    aparMerge
    aparListOpen
    aparPrereq
    aparPrereq_curl_setup
    sample_aparList.log
    sample_aparList.note
```

2.3 Install the commands

To install the APAR commands:

1. Place (copy, move) the aparDownload, aparList and aparMerge files into a directory included in the PATH.
2. Make sure execute permission is granted on each file.
3. Optionally, customize the aparPrereq_curl_setup file to define any environment variables, https or http proxy settings, or curl options that are specific to your installation. Options coded in the aparPrereq_curl_opts variable will be added to each curl command issued by the aparPrereq script.

2.4 Add an entry for the APAR server to the .netrc file

For each userid that will be running the commands:

1. Update the .netrc file for that user to include an entry for the z/TPF APAR server, setting the login userid and password to the z/TPF maintenance userid (ztpfmnt) and password received from the z/TPF Customer Support team.

   An example .netrc entry is shown below:

   ```
   machine transfer.boulder.ibm.com login ztpfmnt password yyyyyyyy
   ```

   Notes:
   - You MUST remember to update your .netrc file each time a the maintenance password is reset. If you do not have a z/TPF maintenance userid, send a note to TPFQA@us.ibm.com.
   - Access to the .netrc file must be restricted to read-only for the owning userid. To set read-only access, issue:

```
       chmod 600 /home/userid/.netrc
```

3.0 Running the z/TPF APAR Download Tools

3.1 Displaying the help information

To display the help information for the commands, use the -h option:

```
  aparList -h
  aparDownload -h
  aparMerge -h
  aparChgperm -h
  aparListOpen -h
  aparPrereq -h
  aparPkgcheck -h
```

This will display the usage information and all options
available for use in each of the commands.

3.2 Downloading the z/TPF APAR CSV File, generating APAR reports and logs

---

1. To download the latest copy of the z/TPF APAR Data CSV File:

   ```
   aparList -c /target_dir/aparList.csv
   ```

2. To download the CSV File, send a report to userid1 and userid2, include
   all closed APARs starting with PUT 8 and continuing through to the
   present, save a log of the run, and save a copy of the note:

   ```
   aparList -c /target_dir/aparlist.csv -s PUT8 -m userid1@host -m userid2@host -l
/target_dir/aparList.log -n /target_dir/aparList.note
   ```

   See:
   - sample_aparList.note for an example of the note that is sent.
   - sample_aparList.log for an example of the log file that is generated.

3. To download the CSV File, send a report to userid1 and userid2, include all open APARs and save a log of the run:

```
aparListOpen -c /target_dir/aparlist.csv -m userid1@host -m userid2@host -l /target_dir/aparListOpen.log
```

3.3 Downloading APARs

1. To download a single APAR and delete the tar file after the unpack:

```
aparDownload -c /target_dir/aparlist.csv -s PJ38500 -e PJ38500 -d /target_dir/download -u /target_dir/untar -k NO
```

2. To download and unpack all APARs in PUT 7 only, keeping the tar files:

```
aparDownload -c /target_dir/aparlist.csv -s PUT7 -e PUT7 -d /target_dir/download -u /target_dir/untar
```

3. To download all APARs in PUT 8 to present:

```
aparDownload -c /target_dir/aparlist.csv -s PUT8 -d /target_dir/download -u /target_dir/untar -k NO
```

4. To download all APARs processed by the aparList command, using the log file as input:

```
aparDownload -c /target_dir/aparlist.csv -l /target_dir/aparlist.log -d /target_dir/download -u /target_dir/untar -k NO
```

5. To download and unpack APARs in PUT 13 to present, omitting opensource APARs:

```
aparDownload -c /target_dir/aparlist.csv -s PUT13 -d /target_dir/download -u /target_dir/untar -k NO -skipOS /target_dir/skipOS.report
```

The list of skipped opensource APARs will be written to the file /target_dir/skipOS.report so that it can
be used for future processing. This option can be useful when access the opensource APAR download site is restricted
by corporate policy, firewall limitations, etc.

3.4 Merging APARs

1. To generate a merged directory of all APARs from PUT 8 to present:

   aparDownload -c /target_dir/aparlist.csv -m /target_dir/merge -u /target_dir/untar -s PUT8

3.5 Updating permissions on a merged APAR directory

1. To generate change the group assignment on a merged directory of all APARs to "tpfusers", give the user(owner) read/write to the files, the group read, and others no access:

   aparChgperm -m /target_dir/merge -g tpfusers -u w -g r -o n

3.6 Sample automated download procedure

First, the base APAR download level must be set in the log file. To do this, run the aparList command to generate a log file with the most recently installed APAR. For example, if all of the PUT 7 apars have been installed, but none after that, generate the log file to include APARs through the end of PUT7:

```
aparList -c /target_dir/aparList.csv -l /target_dir/aparList.log -s PUT7 -e PUT7
```

Once the log file is generated, the following steps can be used to keep up to date with APARs:

1. Run aparList to get a list of all APARs issued after the EndAPAR listed in the log file. In this example, we also send an email to userid1 to let them know what has been added since the last run.

   aparList -c /target_dir/aparList.csv -l /target_dir/download.log -m userid1@host

   If no new APARs have been closed since the EndAPAR in the log, the log file will not be updated and the note sent to the user will indicate no new APARs were posted.

2. Download the APARs found by step 1:

   aparDownload -c /target_dir/aparList.csv -l /target_dir/download.log -d /target_dir/download -u /target_dir/untar

   This can be run even if no new APARs were found. The log file will keep a

record of whether the download had already been run and if so, this script will exit with nothing to be done.

3. Merge the APARs downloaded by step 2:

```
   aparMerge -c /target_dir/aparList.csv -l /target_dir/download.log -u
/target_dir/untar -m /target_dir/merge
```

   As with the download step, this step can be run even if no new APARs were found. The log file will keep a record of whether the merge had already been run and if so, this script will exit with nothing to be done.

By repeating steps 1-3 above, on a periodic basis (daily, weekly), you can keep a growing merge directory of all closed APARs. And to fully automate the process, you can create your own script that performs all three steps and then install that script as a cron job.

For example, your script might contain:

```
aparList -c /target_dir/aparList.csv -l /target_dir/download.log -m email@host
if [[ $? -eq 0 ]] ; then
   aparDownload -c /target_dir/aparList.csv -l /target_dir/download.log -d
/target_dir/download -u /target_dir/untar -k NO
   if [[ $? -eq 0 ]] ; then
      aparMerge -c /target_dir/aparList.csv -l /target_dir/download.log -u
/target_dir/untar -m /target_dir/merge
   fi
fi
```

3.7 Generating an APAR prerequisite list

1. To generate a list of prerequisite APARs for an APAR (PJ44874) through a known applied APAR (PJ44464) and check whether the prereq APARs found have been applied to a z/TPF root folder (/ztpf/prod):

```
      aparPrereq -a PJ44874 -e PJ44464 -t /ztpf/prod
```

   The above will produce output similar to the following:

```
      Begin aparPrereq processing ...

      Prereq APAR Report:  /home/user/mywork/aparPrereq_test/PrereqApars.txt
      Applied File Report: /home/user/mywork/aparPrereq_test/PrereqApars_Files.txt

      Applied: 0
      Missing: 1
```

```
        Partial: 0

        End aparPrereq processing.

     And the output files will contain:

        PrereqApars.txt

           Applied ClsSeq APARNum(PT) Ans H M Abstract
           MISSING 003988 PJ44874(14) PER N N Unable to IPL a non-BSS subsystem with
an SDA of C5xx

        PrereqApars_Files.txt

           PJ44874 MISSING  /ztpf/prod/base/ol/ibf1.cpy
```

3.8 Checking for updated APAR packages

1. To generate a list of APAR tar packages that have been updated after a specific date, for which the APAR was released before that date:

```
        aparPkgcheck 01/01/2021

     The above will produce output similar to the following:

        Begin aparPkgcheck processing ...
           Download z/TPF APAR Data file
           Download APAR tar package dates for: 2021
           Download APAR tar package dates for: 2020
           Download APAR tar package dates for: 2019
           Download APAR tar package dates for: put15
           Download APAR tar package dates for: put14

        The following APAR packages were released before and
        the corresponding tar file was updated after: 01/01/2021

        APAR Package   Tar Date   Release Date
        -------------- ---------- ------------
        PJ45959.tar.gz 04/01/2021 04/24/2020
        PJ45794.tar.gz 04/06/2021 11/22/2019
        PJ45310.tar.gz 03/17/2021 11/15/2018
        PI90312.tar.gz 03/11/2021 08/10/2018
        PI92286.tar.gz 03/11/2021 06/28/2018
        PJ45259.tar.gz 04/30/2021 05/14/2018
        PJ45295.tar.gz 04/30/2021 03/27/2018
        PJ45032.tar.gz 03/19/2021 02/20/2018
```

```
        End aparPkgcheck processing. RC=4
```

4.0 Known Problems and Workarounds

4.1 Downloading a large number of APARs

When downloading a large number of APARs from the server, the "robots" may temporarily suspend the downloads for your id, to prevent one job from consuming all of the resources. This typically happens after a large number of downloads (in our testing, this was typically around 50 APAR packages), but the suspension may also occur after a large number of bytes. The "time-out" usually lasts for 10 minutes or so, but it does not appear to be a fixed amount of time.

When a download fails, the aparDownload script is set up to go into a loop and repeatedly try the same download up to 30 times, waiting for 1 minute in between each download attempt. In testing, this usually allows the aparDownload job to complete, without having to stop and restart the job manually. However, if the maximum number of download attempts is reached, an error will be issued and the download will have to be restarted manually.

4.2 Downloading opensource APARs

The aparDownload command will attempt to download opensource APARs. As of January 2011, when these commands were first created, the opensource downloads were working successfully. However, as more opensource packages are added, the downloads might begin to fail if the website referenced for the opensource download contains more than one tar file and none of those tar files can be uniquely associated with the APAR number being processed. If that happens, the aparDownload command will issue an error and a manual download and unpack of the opensource APAR package will be required.

4.3 Customer modifications to APAR source files

After the APAR source files are merged, any customer modifications must be applied before builds can be run.

If the modified files are maintained in a local_mod directory, the files

that need to be investigated can be found by running the following command to compare the files in the merge directory against those in the local_mod directory under the production source code root directory:

```
cd /target_dir/merge ; find . -type f | grep -v '/local_mod/' | while read filename ; do
    if [[ -s /ztpf_prod_root/local_mod/$filename ]] ; then
       echo "A local_mod version found for: /target_dir/merge/${filename#./}"
       echo "See:  /ztpf_prod_root/local_mod/${filename#./}"
    fi
done
```

5.0 Notices

IBM may not offer the products, services, or features discussed in this information in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. IBM may have patents or pending patent applications covering subject matter in this information. The furnishing of this information does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department 830A
Mail Drop P131
2455 South Road
Poughkeepsie, NY 12601-5400

USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee. Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

5.1 Copyright

5.2 Trademarks and service marks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

5.3 Warranty

This file is provided on an "as is" basis. There are no warranties, express or implied, including the implied warranties of merchantability and fitness for a particular purpose. IBM has no obligation to provide service, defect correction, or any maintenance for the file. IBM has no obligation to supply any updates or enhancements for the file to you even if such are or later become available.