POSIX and Olson time zone formats

Setting the right time zone values.

Introduction

A time zone can be specified in two formats: *POSIX* and *Olson*. This article helps you to understand the Olson and POSIX time zone formats and helps you set the required time zone values in UNIX.

POSIX format specification

The POSIX time zone format is the traditionally used format for AIX systems and provides a slight performance advantage over the Olson time zone format. Example of a POSIX format is `EST5EDT`.

The advantage of POSIX is that you can easily and explicitly specify the time zone and daylight saving time (DST) details manually, however you wish. The performance of applications that call time functions will be faster than using Olson specification. And whenever a nation's government decides to change its DST rules, the POSIX format is simpler because we can simply change the variable definition. There is no need to install any new patch to update time database files, as Olson requires.

The disadvantage of this approach is that it cannot track the history of timezone-related changes and it is not easy to read as it looks cryptic. When a government changes the rules and you update your time zone (`TZ`) variable, it is assumed to be the same DST rule for all years past and future.

Olson format specification

This method maintains a database for each time zone under the */usr/share/lib/zoneinfo* directory in AIX. The advantage with this approach is that the time zone name is easy to specify and Olson maintains a history of time zone and DST-related changes.

This format uses known names of cities or regions. We can specify the time zone name in a simple, easy-to-understand format, such as `"America/Sao_Paulo"` rather than specifying the more complex `TZ=GRNLNDST3GRNLNDDT,M10.3.0/00:00:00,M2.4.0/00:00:00.`

The disadvantage of this approach is that Olson has to load the database file for each time zone specified, and then parse it to find the time zone and DST details. This process can have a modest performance penalty compared to the POSIX format. Additionally, when a government changes a time zone or the DST rule, a new patch becomes necessary for the Olson time database file.

Understanding the POSIX format

The TZ variable specified in POSIX format contains all the information required to identify the time zone, specify when to switch DST on and off, and specify the offset from Coordinated Universal Time (UTC). The system internally does everything in UTC time, and any display of time to the users is a computed offset, depending on the time zone and DST rules specified.

The format is `TZ = local_timezone,date/time,date/time.`

Here, date is in the `Mm.n.d` format, where:

- Mm (1-12) for 12 months
- n (1-5) 1 for the first week and 5 for the last week in the month
- d (0-6) 0 for Sunday and 6 for Saturday

For example:

```
TZ=CST6CDT,M3.2.0/2:00:00,M11.1.0/2:00:00
```

This would effect a change to daylight saving time at 2:00 AM on the second Sunday in March and change back at 2:00 AM on the first Sunday in November, and keep 6 hours time offset from Greenwich mean time (GMT) every year. The breakdown of the string is:

- `CST6CDT` is the name of the time zone
- `CST` is the abbreviation used when DST is *off*
- `6` hours is the time difference from GMT
- `CDT` is the abbreviation used when DST is *on*
- `,M3` is the third month
- `.2` is the second occurrence of the day in the month
- `.0` is Sunday
- `/2:00:00` is the time
- `,M11` is the eleventh month
- `.1` is the first occurrence of the day in the month
- `.0` is Sunday
- `/2:00:00` is the time

Customizing DST using the POSIX format time zone

The POSIX time format can be customized to override the default daylight savings time values. The customized override values for DST can be used when the default POSIX time zone change is not expected on the US dates.

If the DST option is enabled, the default in AIX is for the system time to move forward by 1 hour (to DST) at 2:00 a.m. the second Sunday in March, and move back one hour (to Standard Time) at 2:00 a.m. on the first Sunday in November .However, the date and time at which the switch to DST occurs can be customized by root (global environment) or by users (user environment) by setting the `$TZ` environment variable.

For example, to change the DST change from March to April with daylight saving time of 30 minutes, you need to enter:

```
TZ=CET6CEST530,M4.5.0/02:00:00,M10.5.0/03:00:00
```

Here, the difference between 6 and 5:30 is calculated for daylight saving time.

The following examples represent some of the customized POSIX formats:

```
HAST10HADT,M4.2.0/03:0:0,M10.2.0/03:0:00
AST9ADT,M3.2.0,M11.1.0
AST9ADT,M3.2.0/03:0:0,M11.1.0/03:0:0
EST5EDT,M3.2.0/02:00:00,M11.1.0/02:00:00
GRNLNDST3GRNLNDDT,M10.3.0/00:00:00,M2.4.0/00:00:00
EST5EDT,M3.2.0/02:00:00,M11.1.0
EST5EDT,M3.2.0,M11.1.0/02:00:00
CST6CDT,M3.2.0/2:00:00,M11.1.0/2:00:00
MST7MDT,M3.2.0/2:00:00,M11.1.0/2:00:00
PST8PDT,M3.2.0/2:00:00,M11.1.0/2:00:00
```

Managing the TZ variable

The TZ environment variable is used to represent time zone information. The value of TZ can be specified either in the POSIX or Olson format.

To set the time zone using Olson-defined values, use the following System Management Interface Tool (SMIT) path:

```
System Environments > Change / Show Date, Time and Time Zone.
```

To set the Olson format in the command line, enter:

```
#TZ=America/New_York
```

If the TZ environment variable does not match a proper Olson time zone value, AIX then follows the POSIX specification rules.

To set the POSIX format in the command line, enter:

```
#TZ=EST5EDT
```

If the value set is invalid or unrecognized, the time zone defaults to UTC.

To determine the time zone format in the command line, enter:

```
#echo $TZ
```

Output of the `zdump` command

The output of the `zdump` command gives a clear picture of DST change timing for the particular time zone. For example, in Figure 1, the first two lines explain the scenario when daylight saving is enabled and last two lines explain the scenario when it is disabled.

Figure 1. zdump command output

```
# zdump -v America/New_York | grep 2012
America/New_York  Sun Mar 11 06:59:59 2012 UTC = Sun Mar 11 01:59:59 2012 EST isdst=0
America/New_York  Sun Mar 11 07:00:00 2012 UTC = Sun Mar 11 03:00:00 2012 EDT isdst=1
America/New_York  Sun Nov  4 05:59:59 2012 UTC = Sun Nov  4 01:59:59 2012 EDT isdst=1
America/New_York  Sun Nov  4 06:00:00 2012 UTC = Sun Nov  4 01:00:00 2012 EST isdst=0
```

Line 1:

On Sunday, 11 March, exactly after 01:59:59 clock is expected to set forward by one hour. The abbreviation is EST at 01:59:59 and `isdst=0` specifies that DST is *off* at this time. The corresponding UTC time is 06:59:59.

Line 2:

One second after 01:59:59, time changes to 03:00:00, DST is *on* and the abbreviation changes to EDT. `isdst=1` specifies that DST is enabled. The corresponding UTC time is 07:00:00.

Line 3:

On Sunday, 4 November, exactly after 01:59:59, clock is expected to set back one hour. The abbreviation is EDT at 01:59:59 and `isdst=1` specifies that DST is *on* at this time. The corresponding UTC time is 05:59:59.

Line 4:

One second after 01:59:59, time changes to 01:00:00, DST is *off* and the abbreviation changes to EST. `isdst=0` specifies that DST is disabled. The corresponding UTC time is 06:00:00.