

# FreeBSD 对闰秒的支持

## 目录

1. 介绍 .....	1
2. FreeBSD 对闰秒的默认处理方式 .....	1
3. 注意 .....	1
4. 测试 .....	2
5. 结论 .....	2

## 1. 介绍

闰秒是为了同步地球自转，而对原子钟标准时间所做的特定修正。本文描述了 FreeBSD 如何处理闰秒。

截至本文完稿时，下一个闰秒将会发生在2015年6月30日23:59:60 UTC。这个闰秒将会发生在南北美洲和亚太地区的一个工作日里。

闰秒是由 [IERS](#) 在 [Bulletin C](#) 上宣布的。

[RFC 7164](#) 描述了闰秒的标准行为。也可参见 [time2posix\(3\)](#)。

## 2. FreeBSD 对闰秒的默认处理方式

处理闰秒最简单的方法是使用 FreeBSD 预设的 POSIX 时间规则, 以及 [NTP](#)。如果 [ntpd\(8\)](#) 正在运行，并且时间和正确处理闰秒的上游 NTP 服务器同步，闰秒将使系统时间自动重复当天的最后一秒。不需要进行其它调整。

如果上游的 NTP 服务器没有正确处理闰秒，[ntpd\(8\)](#) 会在错误的上游服务器发现错误并修正后，跟着加上一秒。

如果未使用 NTP，将需要在闰秒过后手动调整系统时钟。

## 3. 注意

闰秒在全世界的同一瞬间插入：UTC 午夜。日本在上午，太平洋在正午，美洲在傍晚，而欧洲在晚上。

我们相信并预期，如果提供了正确和稳定的 NTP 服务，FreeBSD 会在闰秒时按设计运作，正如在之前遇到闰秒时一样。

然而我们要警告，事实上没有应用程序会向内核询问关于闰秒的事。我们的经验是，闰秒正如设计的一样，本质上是闰秒前一秒的重播，这对大部分应用程序开发者来说是意想不到的事。

其它操作系统和电脑可能会也可能不会像 FreeBSD 一样处理闰秒，没有正确和稳定 NTP 服务的系统一点也不知道闰秒的发生。

电脑因为闰秒而崩溃并非闻所未闻，经验显示，大量的公共 NTP 服务器可能会错误处理和公告闰秒。

请试着确认不会因为闰秒而发生任何可怕的事情。

## 4. 测试

测试是否将使用闰秒是可行的。由于 NTP 的性质，测试可能要运行到闰秒前24小时。有些主要的参考时钟来源只在闰秒事件前一小时公告。查询 NTP 守护进程：

```
% ntpq -c 'rv 0 leap'
```

包含 `leap_add_sec` 的输出表明了对于闰秒的正确支持。`leap_none` 会在闰秒前24小时或闰秒过后显示。

## 5. 结论

在实践中，FreeBSD 中的闰秒通常不是个问题。我们希望这篇文章能解释清楚这方面可能出现的状况，以及如何使闰秒事件进行得更顺利。