FreeBSD 對潤秒的支援

目

1. 說明 1
2. FreeBSD預設的潤秒處理 1
3. 警告 1
4. 測試 2
5. 結論 2

1. 說明

潤秒是為了同步地球自轉，與原子鐘所做的特定一秒的修正。本文描述FreeBSD 如何處理潤秒。

本文寫作時，下一個潤秒會發生在2015年6月30日23:59:60
CST。下一次潤秒會發生在南北美洲和亞太地區的工作日。

潤秒是由 IERS 在 Bulletin C所發表。

標準的潤秒行為描述在RFC 7164.。也可見 time2posix(3)。

2. FreeBSD預設的潤秒處理

最簡單的處理潤秒方法使用FreeBSD預設的 POSIX 時間規則，並使用 NTP。如果 ntpd(8)
在執行，而且時間和上游正確處理潤秒的 NTP
伺服器同步，潤秒會使系統時間自動重複當天的最後一秒。不需要其他調整。

如果上游的 NTP 伺服器無法正確地處理潤秒，ntpd(8)
會在錯誤的上游伺服器發現錯誤並跳一秒後，跟著把時間跳一秒。

如果未使用 NTP ，將需要在潤秒過後，手動調整系統時鐘。

3. 警告

潤秒的插入在全世界是在同一個瞬間： UTC
午夜。在日本，是在上午九點，在太平洋，是正午，在美洲，是傍晚，在歐洲，是晚上。

我們相信和預期，如果提供正確和穩定的 NTP
服務，FreeBSD會如設計地在這次潤秒正確運作，就像在之前遇到潤秒時一樣。

然而我們要警告，實務上沒有應用程式曾經要求核心關於潤秒的事。我們的經驗是，如同設計，潤秒本質上是潤秒前一秒的重播，這對大部份應用程式設計師來說是意想不到的事。

其他作業系統或電腦可能會或可能不會像FreeBSD用同樣方法處理潤秒，沒有正確和穩定 NTP
服務的系統一點也不會知道潤秒的發生。

電腦因為潤秒而當機並不是沒有聽聞，經驗上也顯示，有大量公用的 NTP
伺服器沒有正確地處理和公告潤秒。

請試著確定不會因為潤秒而發生任何可怕的事情。

4. 測試

測試是否有使用潤秒是有可能的。由於 NTP
的性質，測試可能要運作到潤秒前24小時。有些主要的參考時鐘來源只在潤秒前一個小時公告。詢問 NTP
行程：

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

包含leap_add_sec 的輸出指出對於潤秒的支援。潤秒前24小時，或是潤秒已經過了，會顯示leap_none。

5. 結論

實務上，FreeBSD
的潤秒通常不是個問題。我們希望這篇概述能幫助釐清預期會遇到什麼狀況，如何使潤秒事件進行的更順利
。