The Clock Mini-HOWTO

Ron Bean, *mc864mclink.it* <mailto:rbean@execpc.com> Dec. 1996

번역: 이 상 훈 *sanghoon_rhee@geocities.com* <mailto:sanghoon_rhee@geocities.com> 2000년 1월

How to set and keep your computer clock on time. 컴퓨터의 시계를 어떻게 맞출 것이며 계속 정확하게 유지할 것인가.

목 차

제 1 절 1. 소개

개인용 컴퓨터의 마더 보드에 달려 있는 실시간 시계 (Real-time Clock, RTC) 은 부정확 하기로 유명 한데, 이건 비싼 워크스테이션이라고 해서 예외가 아닙니다. 리눅스에서는 이런 상황에서도, 심지어는 특별한 하드웨어나 외부에 있는 타임 서버의 도움이 없을 때도 컴퓨터의 시계를 *매우* 정확하게 맞출 수 있는 소프트웨어적인 방법을 제공합니다. 대부분의 사람들은 이 점을 잘 모르고 있는데 이건 아마 다음과 같은 이유때문이라고 생각합니다:

1. 이 방법은 흔히 볼 수 있는 "리눅스 설치 방법" 같은 문서에는 잘 나오지 않고, 설치 할 때 한 번에 정확한 시간을 맞추는 것은 어렵기 때문이다. (물론 모뎀이나 네트워크 카드라도 달려 있다면 전혀 불가능 한 것은 아닙니다)
2. "man clock" 명령을 입력 해 보면 clock(3)에 해당하는 페이지가 나오는데, 이것은 clock 커맨드에 대한 설명이 아니다. ("man 8 clock" 이라고 해야 clock 커맨드에 대한 설명이 나옵니다)
3. 대개 사람들은 정확한 시간에 그다지 신경쓰지 않는다.
4. 일부 거기에 신경쓰는 사람들은 louie.udel.edu 에서 xntpd 패키지를 가져와서 네트워크 타임 서버나 (표준 방송 시보를 수신 할 수 있는) 라디오 시계 하드웨어를 이용해서 시간을 맞춘다.

이 mini-HOWTO 문서는 기술적으로 수준이 낮은 방법을 설명하는 것입니다. 이런것이 자신이 원하는 것이 아니라고 생각하는 분들은 <*www.eecis.udel.edu/~ntp/* <http://www.eecis.udel.edu/~ntp/>> 를 방문

해서 훑어 보십시오. 이 사이트는 xntpd에 대한 모든 정보와 NIST(주1) 및 USNO(주2) 로의 링크등 많은 흥미로운 자료를 제공하고 있습니다. (이 문서 끝에 xntpd 에 대한 약간의 설명을 포함 해 두었습니다).

주의 하나이상의 운영체제를 사용하는 기계에서라면, 각각의 운영체제 서로가 서로를 혼동시키는 일이 없도록 하나의 운영체제에서만 시간을 맞추어야 합니다. 같은 기계에서 리눅스와 윈도즈를 정기으로 교대하여 사용하는 경우라면 윈도즈를 위한 쉐어웨어 버젼의 시계 동기화 프로그램을 쓰는 것을 생각 해 볼수도 있습니다. (위에 언급한 URL 참조)

제 2 절 "clock" 프로그램의 사용법

사실 필요한 정보는 clock(8) 매뉴얼 페이지에서 다 설명하고 있지만, 여기서는 차근차근 전체 과정을 설명해 보겠습니다.

주의 "clock" 프로그램이나 기타 커널시계나 CMOS시계에 영향을 줄 프로그램들은 반드시 루트권한으로 실행해야 효과가 있습니다.

2.1 설치상태를 확인

시스템의 시동 스크립트 파일들 중 "clock -a" 나 "clock -ua" 라는 줄을 포함하는 것이 있는지 확인합니다. 시스템에 설치된 배포판에 따라서 이 내용은 /etc/rc.local 에 있기도 하고 /etc/rc.d/rc.sysinit 나 다른 비슷한 파일에 들어있을 수도 있습니다.

만약 위 파일들에 "clock -s" 나 "clock -us" 라고 적혀 있다면 "s"를 "a" 로 변경하고 /etc/adjtime 이라는 파일이 있는지 확인합니다. 이 파일은 대략 다음과 같은 내용이 한줄에 들어 있을 것입니다:

0.000000 842214901 0.000000

각각의 숫자를 설명 하자면, 첫번째 숫자는 '교정계수' (하루에 몇 초씩 시계가 틀려 지는가), 두번째 숫자는 마지막으로 시계교정을 한 때 (1970년 1월 1일 0시 이후 흘러간 초를 적습니다), 세번째 숫자는 마지막 시계교정때 초단위 이하 값을 얼마나 조정 했는가 하는 값입니다. 만약 /etc/adjtime에 이런 내용이 없다면 루트로 로긴 한 다음 다음과 같이 세 값 모두 0 이 되게 해서 적어 줍니다:

0.0 0 0.0

그 다음 "clock -a" 나 "clock -ua" 를 직접 쉘에서 입력하면 위의 2번째 숫자가 변경됩니다. 시스템 시계가 국제표준시 (UTC 혹은 GMT)를 기록한다면 "u"를 주고 그렇지 않으면 그냥 "a"만 옵션으로 줍니다.

2.2 시스템 시계의 유동률 (drift rate)를 측정한다.

먼저, 시간에 대해서 좀 더 배울 필요가 있겠죠 :-) ? 우리가 현재 파악하고 있는 지역 시간은 정확할 수도 있고 정확하지 않을 수도 있습니다. 필자는 (303)499-7111 번에 전화를 해서 WWV (주3) 의 음성 안내를 이용하는 것을 권장합니다 (이 번호는 장거리 전화입니다). 네트워크 타임 서버를 사용할 수 있다면 xntpd 패키지에 포함되어 있는 ntpdate 프로그램을 사용하면 간단합니다. (ntpdate를 사용할 때는 -b 옵션을 주지 않으면 커널이 CMOS 시계를 잘못 건드릴 수도 있습니다).

아니면 "date -s hh:mm:ss" 명령을 직접 입력해서 커널 시계를 맞출 수도 있는데 이렇게 하고 나서 "clock -w" 명령을 주면 커널 시계를 기준으로 CMOS 시계를 맞추게 됩니다. 마지막으로 시간을 바꾼 때가 언제인지 기억 해 두어야 하니 바꾸고 나서 바로 시간을 확인해서 적어 둡니다. ntpdate를 실행한 경우라면 "date +%s" 명령을 실행해서 나오는 숫자를 적어 둡니다. (이 값은 1970년 1월 1일 0시 이후 흘러간 초입니다)

그런 다음 몇 일이나 몇 주동안 잊고 지내다가 다시 돌아와서 시간이 얼마나 틀려졌는지 확인합니다. ntpdate를 사용하지 않고 수작업으로 시간을 맞췄었다면 최소한 2 주 이상 지난 다음에 확인하는 것이 좋습니다. 이 경우 교정계수는 0.1초 단위에서 반올림 해서 적습니다. 몇 달이 지난 다음에 측정 한다면 교정계수를 0.01초 까지 정확하게 측정할 수 있습니다. (어떤 사람은 이것보다 더 정밀한 값도 신뢰할 수 있다고 합

니다만, 약간 보수적으로 생각하는 것이 좋겠지요). ntpdate를 사용하는 경우라면 그렇게 오래 기다릴 필요는 없지만, 어떤 방법을 사용했건 간에 일정 시간 뒤에 한번 더 시간을 조정하는 일은 꼭 필요합니다.

cron을 이용하면 "clock -a"를 주기적으로 실행해서 커널 시계를 CMOS시계에 일치 시키도록 할 수 있습니다. 사실 이 명령은 시동 스크립트 어딘가에 들어 있기 때문에 매 번 시스템이 부팅 할 때 마다 실행됩니다. 따라서 부팅을 자주 한다면 그것 만으로도 CMOS 시계와 커널 시계를 어느 정도 선에서 일치 시키는 효과를 볼 수 있습니다. (부팅을 자주 하는 사람도 있죠?)

어떤 프로그램들은 커널 시계가 한번에 1초 이상 변경되거나 커널 시계가 뒤로 가면 경고 메시지를 보내기도 합니다. 이런 문제가 있을 때는 xntpd나 ntpdate프로그램을 사용해서, 커널시계가 보다 점진적으로 CMOS 시계와 일치 되게 설정 할 수도 있습니다.

2.3 예

2.3.1 시간 맞추기

먼저, 루트로 로긴 합니다. (303)499-7111로 (주3) 전화를 걸어서 시보 안내를 듣습니다. 그리고 다음과 같이 입력하고 (엔터키이를 치지 않은 채로) 기다립니다:

date -s hh:mm:ss

시보안내에서 '삐'소리가 나면 엔터키이를 누릅니다. (여기서 "date"가 아니라 "ntpdate"를 사용하는 사람은 전화걸고 '삐' 소리를 기다리고 하는 일 같은 건 할 필요가 없습니다). 이렇게 하면 "커널 시계"가 새로운 값으로 조정됩니다. 그 다음에 다음과 같이 입력합니다;

clock -w

이렇게 하면 CMOS 시계를 커널 시계에 맞춥니다. 그 다음에 다음과 같이 입력합니다:

date +%j

그리고 출력되는 숫자를 다음 과정을 위해서 어딘가 적어 둡니다. (만약 "date"대신 "ntpdate"를 사용했던 사람이라면 "date +%s"를 실행합니다)

2.3.2 시간을 초기화 하고 시간 유동률 (drift rate) 을 확인한다.

위 과정에서 적어둔 날짜를 확인합니다. 그 다음 루트로 로긴 해서 다음과 같이 입력합니다:

clock -a 전화를 걸어서 시보 안내를 들으면서, 다음과 같이 입력한다: 이렇게 하면 커널시계를 현재 CMOS 시계에 맞춥니다. (303)499-7111 (주3) 로 전화를 걸어서 시보 안내를 들으면서, 다음과 같이 입력합니다 (아직은 엔터키이를 누르지 않습니다):

date

'삐' 소리를 기다리는 동안 시보에서 음성으로 알려주는 시간을 어디 적어 둡니다. 그리고 '삐' 소리가 나면 엔터를 칩니다. 잠깐, 아직은 전화를 끊으면 안됩니다. 이렇게 해서 실제시간 (시보에서 '삐' 소리를 내는 때)과 커널 시계 (date 명령의 결과로 출력된 시간)가 얼마나 차이가 나는지 확인 할 수 있습니다. 자, 이제 다음과 같이 입력합니다:

date hh:mm:00

분 단위의 값은 지금 시보가 계속 떠들고 있는 숫자보다 1분 빠른 값을 적는 것을 잊지 마십시오. 분이 바뀌는 순간 '삐' 소리가 나면 엔터키를 칩니다. (그 다음에는 전화를 끊어도 됩니다). hh부분에는 지역시를 적습니다. 이 과정은 "커널 시계"를 맞추는 것입니다. 그 다음에 다음과 같이 입력합니다:

clock -w 이렇게 하면 새로운 (보다 정확한) 커널 시계값으로 CMOS 시계를 맞춥니다. 그 다음에는

date +%j

라고 입력합니다. (만약 이전에 "date +%s" 를 썼다면 여기서도 역시 같은 명령을 씁니다).

이렇게 하면 유동률 (drift rate)을 계산하기 위해 필요한 3개의 측정치 (측정한 날짜값 2개와 측정된 초단위의 시차)를 모두 확보한 게 됩니다.

2.3.3 교정 계수 (correction factor) 를 계산한다.

"date"를 실행해서 시간을 맞출 때, 시스템 시계는 느려졌던가요? 아니면 빨라 졌던가요? 시간이 빨라졌다면 초 단위 값에서 얼마간 빼 주어야 하므로 교정계수는 음수가 됩니다. 시간이 느려져 있다면 초 단위 값에서 얼마간 더 해 주어야 하므로 교정계수는 양수가 됩니다.

앞에서 말한 2개의 날짜 값을 비교합니다. "date +%j"를 사용한 경우 라면 적은 값은 그날이 1년 중 몇 번째 날이었는가 하는 것을 알려 줍니다. (평년일 경우 1 365 사이의 값이 나오고 윤년인 경우 1 366 사이의 값이 나옵니다). 이전에 시간을 조정한 것이 올해 1월 1일 이전이었다면 두 번째 날짜값 (오늘의 날짜값)에 365나 366을 더해서 비교 해야합니다.

"date +%s"를 사용한 경우라면 결과는 1970년을 기준으로 해서 초 단위로 나오므로 그냥 두 개의 값을 서고 빼고 나서 86400으로 나누어 주면 두 번의 측정 사이에 흘러간 날짜가 자연스럽게 나옵니다.

/etc/adjtime에 이미 교정계수가 들어 있는 경우라면 이 시점에서 이전의 교정계수 값을 감안 합니다. 이전 교정계수(의 절대치)가 너무 컸다면 지금 구한 교정 계수 값은 이전에 구한 값과 서로 부호가 다를 것입니다. 그렇지 않다면 같은 부호가 되겠지요. 이전의 교정계수를 지나간 날의 수에 곱하고, 새로 계산한 초단위 값을 더한 값을 구합니다. (이 때 부호를 무시 하지 마십시오. 부호가 서로 다르면 더한 값의 절대 값은 더 작아져야 합니다. 그렇지 않다면 새로 구한 값은 더 큰 값이 되지요)

이렇게 구한 '경과된 초'를 경과된 날로 나누면 새로운 교정계수가 나옵니다. 이 값을 /etc/adjtime의 자기 자리에 적어 넣습니다. 새로운 날짜 (오늘)를 (초 단위나 날 단위로)어딘가 적어 두어 다음 교정때 사용합니다.

제 시스템의 /etc/adjtime의 내용은 다음과 같습니다:

-9.600000 845082716 -0.250655

(하루에 9.6초라는 것은 한달이면 거의 5분의 오차를 의미합니다!)

제 3 절 xntpd에 대해서

컴퓨터에는 실제로는 2개의 시계가 있습니다. 하나는 충전지로 가동되는 "실시간 시계(real time clock)" 이라는 것인데 시스템이 꺼져 있을 때도 계속 돌아가면서 시간을 기억하고 있습니다. (다른 말로는 "CMOS 시계", "하드웨어 시계", "RTC" 라고도 합니다) 두번째는 "커널 시계" 인데 이것은 처음 부팅 시에 CMOS 시계에 의해서 맞춰 진 다음 타이머 인터럽트에 의해서 갱신됩니다. (이것은 "소프트웨어 시계" 또는 "시스템 시계" 라고도 합니다). 이 두 개는 둘 다 조금씩, 서로 다른 정도로 틀리기 때문에 시간이 지나면 두 개 모두 정확한 시간과 차이가 커 질 뿐만 아니라 두 개 간에도 서로 조금 씩 차이가 커지게 됩니다.

xntp 문서에서 "시계 (the clock)" 이라고 한 것은 모두 "커널 시계"를 의미하고 있습니다. xntp나 timed (또는 adjtimex() 시스템 콜을 사용하는 다른 프로그램) 를 사용하는 경우, 커널 시계가 CMOS 시계 보다 더 정확하다고 가정하고 매 11분 마다 CMOS 시계를 다시 맞춥니다. (시스템을 리붓 하기 전까지 계속 됩니다). 다시 말해서 "clock" 명령을 사용할 때 CMOS 시계가 언제 리셋 되었는지 알 수가 없고, /etc/adjtime에 적을 교정 계수를 계산 할 수가 없게 된다는 것입니다.

Xntpd는 여러가지 하드웨어 '라디오 시계 (radio clock)' 장치를 지원 하므로, 주기적으로 NIST 등에서 제공하는 시보 서비스에 전화로 접속하여 시간을 맞출 수 있게 되어 있습니다. (전화를 거는 시간 간격을 지정할 때는 전화 요금에도 신경을 쓰십시오.) 이것 (xntpd)는 또한 일정 시간 이상 (NIST 서버등의) 외부 시보를 참조 할 수 없는 상황이 생기더라도 교정 계수를 이용해서 커널 시계를 조정 할 수 있게 되어 있습니다.

대개 '라디오 시계' 장치는 3000 4000 달러 정도 가격이지만, 간단한 "기계장치 (gadget box)" (실제로는 300

baud의 모뎀) 를 시스템과 캐나다의 CHU 시보국에 다이얼을 맞춰 둔 단파 라디오에 연결해서 사용해 볼 수도 있습니다. (<ftp://ftp.udel.edu/pub/ntp/gadget.tar.Z>를 참조 하십시오). Heathkit WWV 수신기 ("가장 정확한 시계" 라고 광고 하던 제품) 도 사용할 수 있는데 이건 약 400 500 달러 정도 합니다. GPS 신호도 시간 정보를 포함하고 잇고 어떤 GPS 수신기는 컴퓨터의 직렬 포트에 연결할 수 있게 되어 있습니다. 가까운 장래에 이러한 GPS 수신기를 싼 가격에 구입할 수 있게 될 지도 모릅니다.

이론대로 하자면, NIST의 전화 시보 서비스에 자동으로 접속해서 교정계수를 자동으로 계산하고 CMOS 시계와 커널 시계를 자동으로 교정 해 주는 프로그램을 만드는 것도 가능합니다. 아직까지는 이런 기능을 가진 프로그램을 보지는 못했지만, 만약 만들려고 한다면 xntpd의 소스만 참조 해도 대부분의 기능을 구현할 수 있을 것입니다.

역주1) NIST : National Institutes of Standards and Technology, 미국 표준기술국

역주2) USNO : U.S. Naval Observatory 미 해군 관측소

역주3) WWV : NBS (미국 표준국)에서 관리하는, 표준 시보 방송국 전화를 통한 시보 안내 서비스도 제공하고 있슴. 한국의 경우 116으로 전화를걸면 시보 안내가 나옵니다.

–이상–