



www.photocase.de

Wznawianie pracy komputera

Pobudka

Włączanie i wyłączenie w odpowiednim czasie komputerów, które nie muszą pracować przez całą dobę, może w znaczący sposób obniżyć zużycie energii. W niniejszym artykule omówimy trzy różne metody programowania czasu wznawiania pracy komputera pracującego pod kontrolą systemu Linux i to bez konieczności każdorazowej ingerencji w ustawienia czasu w BIOS komputera.

MIRKO DÖLLE

Serwery, systemy RAID i podobne muszą pracować 24 godziny na dobę przez 7 dni w tygodniu. Komputery typu desktop często zostawiamy włączone, gdy wychodzimy wieczorem z pracy, mimo że nie są one potrzebne dłużej niż 10 godzin na dobę. Oczywiście czasem zdarza się, że musimy wysłać automatycznie faks przy pomocy serwera faksowania o godzinie drugiej w nocy, ale czyż nie lepiej po prostu włączyć komputer kilka minut przed drugą i wyłączyć go zaraz po wykonaniu zadanej czynności, tak aby w tym stanie czekał do kolejnego nocnego zadania?

Zastosowanie rozwiązań tego typu jest całkiem proste. Potrzebujemy tylko wyłącznika zegarowego za kilkanaście złotych i już będziemy mogli włączać i wyłączać komputer o dowolnej godzinie. Oczywiście jest to bardzo siermiężne rozwiązanie, nie nadające się do zastosowania w firmie.

BIOS, znajdujący się na płytach głównych komputerów, zwykle umożliwia wznawianie pracy systemu bez konieczności stosowania dodatkowych urządzeń. Nowsze komputery posiadają na ogół także możliwość uruchamiania komputera o określonym czasie. Warto pamiętać także o dodatkowych funkcjach typu „Wake on LAN” i „Wake on Ring”, które umożliwiają wznawienie pracy serwera na żądanie.

Problemem jest zmiana czasu dla funkcji sterowania włączaniem komputera z poziomu Linuxa – różni producenci i wersje systemu BIOS oznaczają różne rozwiązania zastosowane na płytach głównych. Przykładowo, na wie-

lu płytach głównych wznawianie systemu poprzez ACPI nie działa prawidłowo ze względu na różne implementacje tego standardu. Z kolei wznawianie systemu z wykorzystaniem pamięci NVRAM (pamięć trwała o dostępie bezpośrednim), czyli pamięci, w której BIOS przechowuje informacje o systemie, pozwala na bezpośrednią edycję czasu wznawiania komputera. Oczywiście przy założeniu, że odzyskamy odpowiednią komórkę pamięci, a BIOS „zauważy” dokonane zmiany wartości. Trzecia metoda, *settime*, wykorzystuje pewną sztuczkę – BIOS (komputer) jest zawsze uruchamiany tego samego dnia, o tej samej godzinie, a zegar komputera jest ustawiany na konkretną datę podczas wyłączania komputera.

ACPI – tajna broń?

Wykorzystanie standardu ACPI to prawdopodobnie najprostszy sposób uruchomienia komputera o określonym czasie, oczywiście przy założeniu, że jądro systemu obsługuje ACPI. Jeżeli tak, wystarczy zapisać czas uruchomienia komputera w `/proc/acpi/alarm`:

```
echo 2004-08-02 20:15:00 >/proc/
/acpi/alarm
```

Jądro systemu przekazuje czas bezpośrednio do zegara czasu rzeczywistego komputera – nie przekazuje jednak daty. Dlatego komputer będzie uruchamiany o tej samej godzinie każdego dnia, a nie tylko wybranego, ustalonego dnia.

Problemem jest obecny standard ACPI,

w którym zstandardyzowano przekazywanie czasu do zegara czasu rzeczywistego, ale nie istnieje taki standard dla przekazywania daty uruchomienia komputera. Dlatego funkcja `acpi_system_write_alarm` z `drivers/acpi/system.c`, będąca częścią jądra 2.4, lub `drivers/acpi/sleep/proc.c` w przypadku jądra 2.6, potrafi obsługiwać wyłącznie czas włączenia alarmu.

Prehistoria

Twórcy implementacji ACPI w jądrze systemu, Andy Grover i Paul Diefenbaugh, przygotowali już rozwiązanie zapewniające pełną funkcjonalność, lecz wymagana jest jeszcze właściwa tabela ACPI (FACP). Czytając komentarze w kodzie jądra zauważymy, że obecne tabele FACP dla płyt głównych są bezużyteczne, co prowadzi do blokowania kodu i jego wyłączenia.

Niektóre płyty główne nie przekazują prawidłowo ustawionego czasu do zegara czasu rzeczywistego, przechowując tylko godzinę ostatniego uruchomienia komputera zamiast ustawionego czasu uruchomienia. Inne płyty główne powodują zawieszenie systemu przy próbie zapisania informacji w `/proc/acpi/alarm`.

Wznawianie pracy systemu przy pomocy ACPI działa prawidłowo z wieloma obecnie produkowanymi płytami głównymi. Na stronie dystrybucji LinVDR (mini-magnetowid cyfrowy) w części Distribution znajdziemy listę płyt głównych kompatybilnych ze standardem ABPI [1].

Do góry nogami

Jeżeli do uruchamiania komputera chcemy korzystać z ACPI, musimy w pierwszej kolejności sprawdzić ustawienia systemu BIOS. Od wielu lat większość komputerów posiada funkcje Wake on Timer, Resume on Alarm, RTC Alarm Resume lub podobne. W ten sposób użytkownicy muszą ustawiać czas uruchomienia komputera w BIOS. Wznawianie pracy systemu przy pomocy ACPI działa na podobnej zasadzie, lecz nie korzysta z funkcji systemu BIOS. Dlatego w przypadku większości płyt głównych należy wyłączyć (disable) funkcje uruchamiania systemu z BIOS. Z tego powodu, czas kolejnego uruchomienia komputera nie będzie już widoczny z poziomu BIOS.

Kolejna przeszkoda polega na tym, że na niektórych płytach głównych, np. na Asus A7V133, nie działa prawidłowo skrypt wyłączający komputer. Wykorzystuje on polecenie hwclock -w do synchronizacji zegara czasu rzeczywistego z czasem systemowym po ustawieniu czasu ponownego uruchomienia komputera. Komputer po prostu nie uruchamia się ponownie o określonej godzinie. Podczas wywoływania polecenia hwclock należy użyć parametru --directisa lub też ustawić czas ponownego uruchomienia komputera dopiero po wykonaniu polecenia hwclock.

Pobudka...

Jak już zauważyliśmy wcześniej, wznawianie systemu ACPI powoduje codzienne uruchamianie komputera o określonej godzinie i nie ogranicza to w żaden sposób przydatności tego standardu. Budzenie przez ACPI może być bardzo przydatne dla serwerów faksowania, które automatycznie przetwarzają wiadomości faksowe każdej nocy przed ponownym wyłączeniem komputera. Funkcja jest również przydatna dla magnetowidów opartych na PC (PVR), jak np. VDR.

Jednakże uruchamianie serwera drukowania o dziewiątej rano i wyłączenie go o siódmej wieczorem oznaczałoby niepotrzebne zużycie prądu przez 10 godzin każdego weekendowego dnia. Jednym z rozwiązań jest napisanie dodatkowych skryptów startowych, które rozpoznawałyby weekendy i święta państwowe i wyłączałyby komputer w te dni, lecz jest to tylko rozwiązanie przejściowe. Gdyby jednak z jakiegos powodu trzeba było skorzystać z takiego serwera drukowania w sobotni wieczór, zostałaby on natychmiast wyłączony, nawet gdyby został uruchomiony ręcznie.

Lepszym rozwiązaniem jest napisanie skryptu, który sprawdzałby dni wolne i święta

przy wyłączaniu komputera, programując jednocześnie datę i czas wznawienia pracy systemu. Najlepszym rozwiązaniem będzie tutaj wznawianie pracy systemu z wykorzystaniem pamięci NVRAM oraz metody setttime.

Budzenie z NVRAM

Wznawianie pracy systemu przy pomocy NVRAM wykorzystuje ustawienia BIOS przechowywane w pamięci trwałej RAM (NVRAM). Moduł jądra nvrाम umożliwi Linuxowi uzyskanie dostępu do obszaru pamięci trwałej o maksymalnym rozmiarze 128 bajtów.

Aby wszystko działało sprawnie, musimy skompilować moduł nvrाम, a także utworzyć urządzenia znakowe /dev/nvrाम o identyfikatorze głównym 10 i pobocznym 144, /dev/rtc o identyfikatorze głównym 10 i pobocznym 135 oraz /dev/mem o identyfikatorze głównym 1 i pobocznym 1. Ponadto należy zmodyfikować konfigurację modułu w /etc/modules.conf w taki sposób, aby moduł NVRAM był ładowany do pamięci przy próbie uzyskania dostępu do /dev/nvrाम lub wraz z uruchamianiem systemu.

Korzystanie z funkcji BIOS

Kompilacja programu z kodu źródłowego (dostępnego pod adresem [2]) powinna przebiegać bez zakłóceń. Domyślną ścieżką instalacji jest /usr/local, ale administratorzy systemu mogą w razie konieczności zmienić plik Makefile i ustawić ścieżkę na /usr. Jeżeli w tym momencie żadne z urządzeń /dev/nvrाम, /dev/rtc i /dev/mem nie istnieje, wykonujemy polecenie make devices, co spowoduje ich automatyczne utworzenie.

W wersji 0.96 instrukcje tr w liniach 62 i 75 skryptu pomocy guess-helper.sh nie działają:

```
answers=~$echo $answers | >
tr [:lower:] [:upper:]`
```



Rysunek 1: Większość systemów BIOS obsługuje funkcję uruchamiania systemu: **RTC Alarm Resume** lub o podobnej nazwie.

Na szczęście, po dodaniu apostrofów skrypt pracował prawidłowo

```
answers=~$echo $answers | >
tr '[:lower:]' '[:upper:]'`
```

Większość producentów traktuje sposób alokacji pamięci NVRAM jak wielką tajemnicę – ze względu na brak dokumentacji możemy tutaj napotkać trudności z odnalezieniem miejsca przechowywania godziny i daty. Co gorsza, lokalizacja pamięci zmienia się po każdej aktualizacji systemu BIOS.

Kod źródłowy „budzenia NVRAM” zawiera wykaz obsługiwanych płyt głównych. Informacje te uzyskano dzięki skryptowi guess-helper.sh. Uruchomienie nvrाम-wakeup z parametrem -D spowoduje uruchomienie programu w trybie debugowania i dzięki temu dowiemy się, czy nasza płyta główna jest rozpoznawana przez program. Jeżeli nvrाम-wakeup wyświetla następujący komunikat:

```
nvrाम-wakeup: Your motherboard >
is currently not supported.
```

mimo to uruchamiamy guess-helper.sh. Skrypt guess-helper.sh podejmie próbę zlokalizowania odpowiednich miejsc w pamięci NVRAM. Do wykonania pełnej procedury będziemy musieli uruchamiać nasz komputer przynajmniej cztery razy. Za pierwszym razem musimy wyłączyć funkcję wznawiania pracy systemu z BIOS, ustawić datę na 31 bieżącego miesiąca, sekundę przed północą i uruchomić guess-helper.sh. Następnie ustawiamy datę uruchomienia systemu na 11 bieżącego miesiąca, a czas na 12:13:14, potem ustawiamy datę na 1 bieżącego miesiąca o północy i w końcu ponownie wyłączamy funkcję wznawiania pracy systemu. Podczas każdej próby wywołania wznawiania pracy systemu skrypt guess-helper.sh porównuje zawartość pamięci NVRAM z innymi wywołaniami, aby na tej podstawie ustalić położenie punktów w pamięci odpowiedzialnych za wznawianie pracy systemu. guess-helper.sh wykorzystuje do tego celu dwie metody: dostęp poprzez /dev/nvrाम oraz dostęp bezpośredni do adresów I/O. Dzięki temu otrzymujemy dwa pliki konfiguracyjne nvrाम-wakeup.conf w katalogach guess-nvrाम-module i guess-directisa w katalogu głównym root.

Wyszukiwanie informacji

Rzut oka na plik konfiguracyjny wskazuje, w którym miejscu przechowywane są informa-

cje o płycie głównej i adresach. Plik `/root/guess-nvram-module` dla płyty Elito Epox 8K5A2+ był, poza kilkoma komentarzami, pusty, za to plik `/root/guess-directisa` zawierał adresy pokazane na Listingu 1.

Jeżeli oba pliki konfiguracyjne będą puste, należy założyć, że skrypt `guess-helper.sh` nie potrafił zlokalizować szukanych adresów.

Musimy jeszcze wskazać programowi ścieżkę dostępu do pliku konfiguracyjnego – najlepiej za pomocą parametru `-C` podczas każdorazowego wywołania polecenia `nvram-wakeup`. Ponadto w naszym przypadku wymagany jest parametr `-A`, gdyż będziemy korzystać z bezpośredniego adresowania I/O, bez udziału urządzenia `/dev/nvram`. Jeżeli płyta główna jest obsługiwana przez `nvram-wakeup`, można oczywiście pominąć oba parametry.

Jazda próbna

Aby rozpocząć pierwsze testy z `nvram-wakeup`, należy jeszcze ustawić w BIOS czas wznawienia pracy systemu. Do kontroli poprawności użyjemy `nvram-wakeup`. W poniższym przykładzie ustawiliśmy w BIOS czas wznawienia pracy na 20:15 15 bieżącego miesiąca. Data ustalana dla `nvram-wakeup` musi uwzględniać przesunięcie czasowe pomiędzy czasem lokalnym a UTC, plus dodatkowe pięć minut na uruchomienie komputera. Na Rysunku 2 pokazano aktualne wartości dla `nvram-wakeup`. W górnej części Rysunku 2 pokazano dane zlokalizowane przez `nvram-wakeup` w pamięci NVRAM; wartości znajdujące się poniżej to dane wprowadzone przez nas, bez parametru ochrony przed zapisem `-N`. Wartości są takie same, więc możemy założyć, że skrypt `guess-helper.sh` prawidłowo zlokalizował odpowiednie komórki pamięci. Następnym krokiem jest przesunięcie czasu o 15 minut w przód i wyłączenie komputera:

```
nvram-wakeup -s `date -d „+15`
Minutes" +%s` -A -C /etc/nvram
```

```
-wakeup.conf
poweroff
```

Jeżeli komputer nie uruchomi się w określonym czasie, może być to spowodowane problemem z APM/ABPI albo konieczne jest ponowne uruchomienie komputera, aby wszystko działało jak należy.

Ponowne uruchomienie, a zamknięcie systemu

Aby przekonać się, czy mamy do czynienia z problemem ponownego uruchamiania komputera, wystarczy przeprowadzić prostą procedurę. Ustawiamy czas wznawienia pracy systemu na kolejny pełny kwadrans i wpisujemy polecenie `reboot`, aby ponownie uruchomić komputer. Następnie ustawiamy czas wznawienia pracy systemu na następny pełny kwadrans i tym razem wpisujemy polecenie `poweroff`, co spowoduje zamknięcie systemu i wyłączenie komputera.

Jeżeli BIOS wznowi pracę systemu po drugim kwadransie, oznacza to, że konieczne jest odczytywanie parametrów BIOS do korzystania z tej funkcji. W przypadku niektórych płyt głównych rozmiar obszaru pamięci dostępny dla `/dev/nvram` (jak zdefiniowano w module `nvram`) jest zbyt mały, aby uzyskać dostęp do całego obszaru NVRAM. Typowym przykładem tego problemu jest sytuacja, w której `guess-helper.sh` nie potrafi zlokalizować pozycji dla dnia, godziny, minut i sekund lub też znajduje tylko niektóre z nich. Źródła jądra definiują ten obszar w `drivers/char/nvram.c`:

```
#define NVRAM_BYTES 2
(128-NVRAM_FIRST_BYTE)
```

Musimy zatem rozszerzyć ten obszar do pełnych 128 bajtów:

```
#define NVRAM_BYTES 128
```

Listing 1: Plik `nvram-wakeup.conf`

```
##### ## - BIOS release: „04/07/2003”
motherboard autodetection
information:      addr_stat = 0xD2
##              shift_stat = 5
## - motherboard vendor: „”
## - motherboard type: „VT8367-
8235”
## - motherboard revision: „”
## - BIOS vendor: „Award Software
International, Inc.”
## - BIOS version: „6.00 PG”
upper_method = VT8235_37
```

Taka zmiana może pomóc skryptowi `guess-helper.sh` w prawidłowym zlokalizowaniu odpowiednich wartości.

Powrót do przeszłości

Skrypt `settime` to pomysłowa metoda z gwarancją prawidłowego działania na każdej płycie głównej. Wykorzystano tutaj ideę zaprogramowania w BIOS jednego, określonego czasu, np. 31 dnia miesiąca o godzinie 23:59:59. Podczas zamykania systemu skrypt oblicza przesunięcie czasowe pomiędzy wyłączeniem a wznawieniem pracy systemu, odejmuje to przesunięcie od daty 31 sierpnia 2004, 23:59:59 i ustawia czas systemowy i zegar czasu rzeczywistego na odpowiedni dzień i godzinę w sierpniu 2004. Po ponownym uruchomieniu komputera pozostanie już tylko ustawienie właściwej daty i aktualnego czasu.

W praktyce jednak metoda ta może być dość kłopotliwa. Musimy pamiętać o ustawianiu właściwego czasu i daty po każdym uruchomieniu komputera i to jeszcze przed wykonaniem polecenia `fsck` – zapobiegnie to sprawdzaniu za każdym razem dysku twardego. Jeżeli zdążymy poprawić czas i datę przed wykonaniem `fsck`, nie będziemy mogli tego nigdzie odnotować, gdyż będziemy mieć wyłącznie prawa do odczytu danych na dysku. Komputery dwusystemowe z systemem Windows lub innym systemem operacyjnym mogą sprawić także niemałe trudności, gdyż będą prawdopodobnie uruchamiać się w niewłaściwym czasie. Tak więc takie rozwiązanie ma sens, jeżeli uruchamiamy komputer przynajmniej raz na dwa miesiące.

Przesunięcie czasowe

Kolejne rozwiązania oferują dwa oddzielne skrypty: `settime` odlicza przesunięcie czasowe pomiędzy czasem rzeczywistym a czasem wznawienia pracy systemu, zapisuje różnicę w pliku o nazwie `/etc/timediff`, ustawia czas systemowy na datę w sierpniu 2004 i synchronizuje zegar czasu rzeczywistego [3]. Podczas



Rysunek 2: Aby pierwsza próba uruchomienia komputera była udana, wyświetlane przez `nvram-wakeup` wartości muszą być takie same. Czas wznawienia pracy systemu ustalony w BIOS pokazano na górze – wartość znajdująca się poniżej jest przetwarzana przez program.

uruchamiania systemu jako pierwszy skrypt proces `init` wywołuje `correcttime`. Odczytuje on przesunięcie czasowe, poprawia czas systemowy i synchronizuje z nim zegar czasu rzeczywistego. Dzięki temu pozostałe skrypty uruchamiane wraz ze startem systemu nawet nie zauważają przesunięcia czasowego.

Metoda wykorzystująca `settime` przyjmuje zasadę, że zegar czasu rzeczywistego będzie odmierzał aktualny czas. Dzięki temu nie potrzeba synchronizować czasu systemowego z innym systemem czy porównywać czasu z dokładnym zegarkiem podczas każdorazowego uruchamiania systemu. Jedyne czego potrzebujemy, to liczba sekund, które „zgubił” zegar w stosunku do czasu rzeczywistego.

Nie ma również konieczności przeprowadzania skomplikowanych obliczeń biorących pod uwagę lata przestępne – polecenie `date` posiada wszystkie funkcje, których potrzebujemy. Przykładowo:

```
date -s „+3600 seconds”
```

spowoduje ustawienie zegara „w przód” o jedną godzinę. Aby wrócić do przeszłości, wystarczy wpisać:

```
date -s „3600 seconds ago”
```

Ponadto `date` potrafi dokonywać konwersji dat zapisanych według międzynarodowego standardu, podawanych w sekundach od początku epoki:

```
debian:~# date -d „2004-08-31 23:59:59” +%s
1093993199
```

Reszta jest banalna. Na Listingu 2 pokazano fragment skryptu `settime`. W linii 3 dokonywana jest konwersja czasu wznawienia pracy systemu na sekundy, w linii 4 w ten sam sposób obliczana jest bieżąca data, a w linii 6 wyliczone wcześniej wartości pomagają obliczyć przesunięcie czasowe (w sekundach).

W linii 5 obliczany jest czas wznawienia pracy systemu ustawiany w BIOS, np. sekundę przed północą 31 sierpnia 2004, oczywiście wyrażony w sekundach, dzięki czemu w linii 7 możliwe jest obliczenie różnicy w czasie rzeczywistym i zapisanie wartości

Listing 2: Fragment skryptu `settime`

```
#!/bin/bash
BiosWakeup='2004-08-31 23:59:59'
Wakeup=`date -d '$1' +%s`
Now=`date +%s`
Bios=`date -u -d '${BiosWakeup}' +%s`
Diff=$((Wakeup-Now))
echo '$(Now-${Bios})' > /etc/timediff
date -u -s '${BiosWakeup} ${Diff} seconds ago' >/dev/null
hwclock -w --utc
```

w pliku `/etc/timediff`. Teraz, po ponownym uruchomieniu systemu, skrypt `correcttime` musi tylko dodać obliczoną różnicę do czasu systemowego, aby w ten sposób otrzymać czas rzeczywisty.

W liniach 8 i 9 ustawiany jest czas systemowy i zegar czasu rzeczywistego do poprzedniej wartości. Podobnie jak w linii 5, czasy te zamieniane są do formatu UTC i zapisywane. Aby zamiast czasu UTC używać dla zegara czasu rzeczywistego z naszej strefy czasowej, musimy zmienić trzy linie.

Listing 3: Fragment skryptu `correcttime`

```
#!/bin/bash
if [ -r /etc/timediff ]; then
    Timediff=`date -r /etc/timediff +%s`
    Now=`date +%s`
    if [ '${Timediff}' -gt '${Now}' ]; then
        Diff=`cat /etc/timediff | head -n 1`
        date -s '+${Diff} seconds' >/dev/null
        hwclock -w --noadjfile --utc
    exit 0
fi
```

Korekta czasu podczas uruchamiania systemu

Skrypt `settime` jest uruchamiany tylko na żądanie, gdy pojawia się taka potrzeba przed zamknięciem systemu, a więc wtedy, gdy chcemy uruchomić komputer o określonej godzinie. Oznacza to, że skrypt `correcttime` musi dowiedzieć się, czy czas zegara czasu rzeczywistego jest faktycznie czasem rzeczywistym, czy też komputer pracuje obecnie z datą przeszłą. Na Listingu 3 pokazano fragment kodu skryptu `correcttime`.

Odwołanie do polecenia `date` w linii 3 sprawdza, kiedy ostatnio modyfikowany był plik `/etc/timediff`, podając wartość w sekundach. Jeżeli data modyfikacji jest datą przyszłą, czas systemowy należy cofnąć. Jeżeli na-

tomiast jest to data z przeszłości, oznacza to, że czas systemowy jest aktualnym czasem rzeczywistym.

W liniach 6, 7 i 8 skrypt `correcttime` odczytuje przesunięcie czasowe pomiędzy czasem systemowym a rzeczywistym na podstawie pliku `/etc/timediff` i dodaje je do aktualnego czasu systemowego, cofając w ten sposób nasz komputer w przeszłość. Czas rzeczywisty jest następnie przekazywany do zegara czasu rzeczywistego, gdyż nie wszystkie dystrybucje

dokonyują synchronizacji czasu rzeczywistego podczas uruchamiania komputera.

Oszczędzaj prąd – zarabiaj pieniądze!

Rzut oka na ostatni rachunek za prąd może przekonać, że elektryczność sporo kosztuje. Możemy jednak oszczędzać energię elektryczną – i to bez specjalnych wyrzeczeń. Pięć standardowych komputerów PC typu desktop (150 Watów każdy) pracujących non-stop 24 godziny na dobę, jak to się zdarza w wielu biurach i firmach, zużywa ponad 6500 kilowatogodzin. Serwery są jeszcze bardziej żarłoczne, gdyż zużycie 400 Watów przez jeden serwer nie jest wcale wyjątkiem. Pojedynczy serwer może zużyć ponad 3500 kilowatogodzin, to już sporo pieniędzy. Automatyczne wyłączanie komputerów o dziewiątej wieczorem i ponowne ich uruchamianie o szóstej rano oznacza oszczędności rzędu 3600 kilowatogodzin w przypadku komputerów typu desktop i około 2000 kilowatogodzin w przypadku serwerów.

Są to konkretne oszczędności rzędu kilkuset złotych, nie mówiąc już o zmniejszonym poziomie CO₂ czy innych korzyściach dla środowiska naturalnego.

Nawet w środowiskach klient-serwer, gdy nikt nie pracuje w weekend, można skonfigurować serwer, włączając opcję „Wake on LAN” – jeśli jednak ktoś przyjdzie to będzie mógł skorzystać z serwera. ■

INFO

- [1] Kompatybilność ACPI:
<http://linvdr.org/wiki/index.php?pagename=LinVDR-Mainboards>
- [2] Wznawianie pracy systemu z NVRAM:
<http://sourceforge.net/projects/nvram-wakeup/>
- [3] Skrypty `settime` i `correcttime`:
<http://www.linux-magazin.de/Service/Listings/2004/08/wakeup>