# 100 sposobów na sieci bezprzewodowe


Autor: Rob Flickenger
Tłumaczenie: Witold Zioło
ISBN: 83-7361-391-9
Tytuł oryginału: Wireless Hacks
100 Industrial-Strength Tips & Tools
Format: B5, stron: 288

Świat opleciony jest kablami. Pomimo tego wielu ludzi uważa, że najlepiej łączyć się z siecią bez ich użycia. Technologia bezprzewodowa – niezależnie od tego, czy stosowana jest w celu zastąpienia kabli czy zapewnienia łączności z internetem – zmienia sposób komunikowania się urządzeń i ludzi.

Książka „100 sposobów na sieci bezprzewodowe" zawiera 100 porad przygotowanych przez ekspertów na podstawie codziennych doświadczeń a dotyczących sieci bezprzewodowych. Każda porada, choć można ją przeczytać w zaledwie kilka minut, pozwoli zaoszczędzić wielogodzinnych poszukiwań.

W książce można znaleźć opisy przydatnych rozwiązań, które pozwolą wykorzystać wszystkie możliwości technologii bezprzewodowej.
Opisano między innymi:

- różne wersje standardu 802.11 oraz kryteria wyboru technologii najlepszej w danych warunkach,
- zastosowanie technologii bezprzewodowych, takich jak Bluetooth, przenośnych urządzeń radiowych, telefonów komórkowych, technologii Wi-Fi, hotspots i innych, również mniej popularnych, metod łączenia się z siecią,
- wdrożenie praktycznych metod wykrywania, analizy oraz monitorowania sieci bezprzewodowych – własnych oraz publicznych,
- poszerzanie zasięgu sieci oraz jak najlepsze wykorzystanie dostępnego widma radiowego,
- projektowanie i budowę własnych anten oraz punktów dostępowych,
- planowanie i zestawianie łączy na dużych odległościach,
- zagadnienia bezpieczeństwa sieci bezprzewodowych oraz zabezpieczania zasobów przed nieautoryzowanym dostępem oraz podsłuchem.

Książka przeznaczona jest dla osób zaawansowanych i średnio zaawansowanych. Niezależnie od tego czy sieć bezprzewodowa obejmuje tylko pomieszczenia biura, czy też ma sięgać drugiego końca miasta, ten zbiór technik będzie pomocny przy jej budowie.

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

# Spis treści

ROZDZIAŁ DRUGI

# Bluetooth i urządzenia przenośne

## Sposoby 13. – 19.

W przemyśle telekomunikacyjnym mówi się wiele o zapewnianiu łączności „na ostatnim kilometrze". Technologię Bluetooth należy traktować jako metodę zapewniającą łączność „na ostatnich metrach". Jawi się ona jako łatwy sposób wyeliminowania nieporęcznych kabli takich urządzeń, jak słuchawki, urządzenia sterujące, asystenty osobiste (PDA) i inne małe przyrządy. Celem technologii Bluetooth jest wyeliminowanie potrzeby noszenia wszędzie ze sobą metrowej długości kabli (oczywiście każdego z inną końcówką), służących do podłączenia danego urządzenia do laptopa. Urządzenia wykorzystujące technologię Bluetooth mogą komunikować się z laptopami lub z komputerami biurkowymi lub nawet komunikować się między sobą, wymieniając z łatwością informacje. Na rynku dostępne są już wykorzystujące technologię Bluetooth urządzenia wejściowe, takie jak myszy czy klawiatury. Mimo że Bluetooth powoduje zwiększenie uzależnienia od baterii, to jednak jest na najlepszej drodze do „wycięcia w pień" wszystkich kabli wychodzących z komputera. W tym rozdziale pokazane zostaną niektóre świetne pomysły zastosowania technologii Bluetooth.

W rozdziale przedstawionych zostanie także kilka sposobów wykorzystania sieci komórkowych **[Sposób 8.]**. Sieci te stają się szczególnie przydatne tam, gdzie sieci Wi-Fi lub inne środki łączności są niedostępne. Urządzenia wyposażone w Bluetooth, telefon komórkowy, pamięć, możliwości odtwarzania dźwięków, a nawet w aparat fotograficzny od jakiegoś czasu pojawiają się na rynku. Urządzenia te rozpoczynają proces nieuchronnej konwergencji produktów konsumenckich z komputerami i Internetem, udostępniając przeciętnemu użytkownikowi bezprecedensowe dotychczas możliwości komunikowania się. Dalej przedstawione zostaną rozwiązania, które rozwijają koncepcję hiperłączności jeszcze bardziej.

SPOSÓB 13.

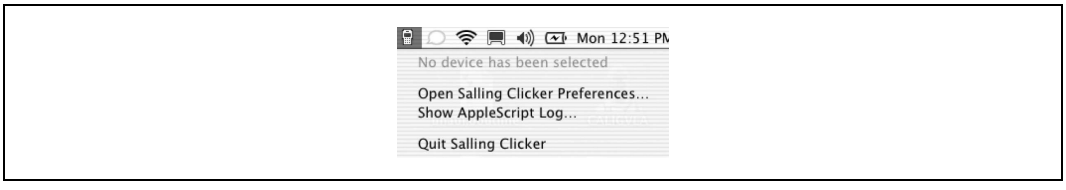
### Zdalne sterowanie systemem OS X z telefonu Sony Ericsson

Telefon komórkowy można wykorzystać do sterowania prezentacjami lub programem iTunes albo do wykonywania innych operacji, które da się zapisać w języku AppleScript.

Program Salling Clicker jest jednym z najlepszych programów wykorzystujących technologię Bluetooth. Zamienia on telefon Sony Ericsson w kolorowy, programowalny „pilot"

do systemu OS X. Można za jego pomocą uruchamiać programy, sterować prezentacjami, a nawet używać go jako myszy. Program ten działa na wielu telefonach Sony Ericsson, takich jak T39m, R520m, T68, T68i oraz T610. Można go pobrać z serwisu VersionTracker lub bezpośrednio ze strony *http://homepage.mac.com/jonassalling/Shareware/Clicker/*.

Program instaluje się jako nowy panel sterujący i jest automatycznie uruchamiany. Należy kliknąć małą ikonę telefonu znajdującą się na pasku menu (rysunek 2.1), wybrać *Open Salling Clicker Preferences…*, a następnie kliknąć *Select Device*. Wcześniej należy w systemie OS X uaktywnić Bluetooth i położyć telefon w pobliżu komputera.



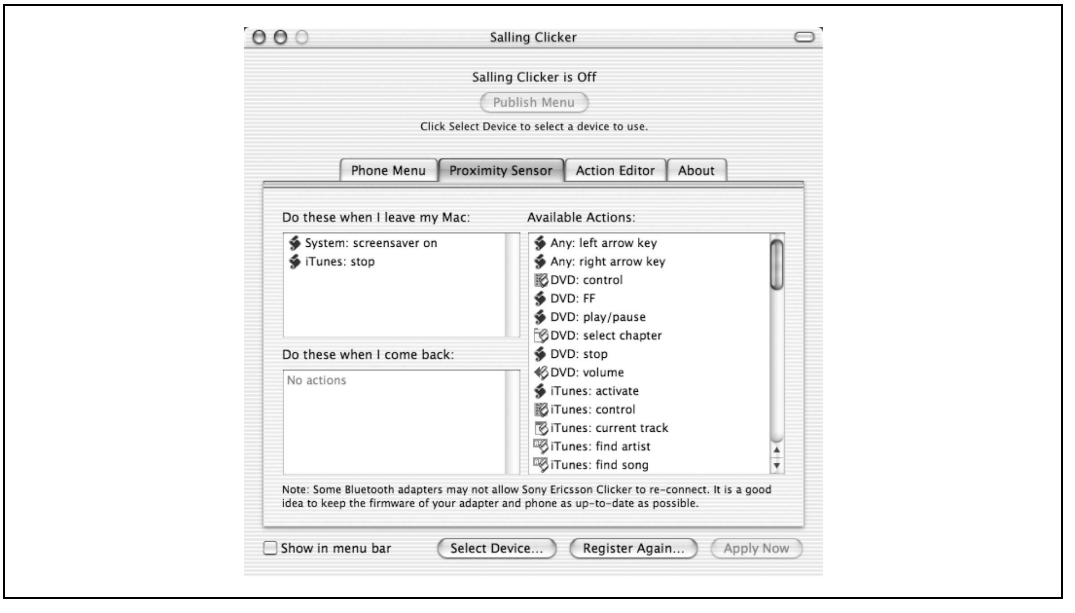

*Rysunek 2.1. Ikony paska menu programu Clicker*

Z listy dostępnych telefonów należy wybrać używany aparat i następnie zapisać zmiany. Od tej pory za pomocą telefonu można sterować systemem OS X oraz tworzyć swoje menu i przesyłać je do telefonu. Na zakładce *Phone Menu* można tworzyć własne menu zawierające dowolne polecenia, a następnie przesyłać je do telefonu. Systemem OS X steruje się, nawigując w telefonie pośród przygotowanych menu i wybierając czynności do wykonania, takie jak na przykład uruchomienie aplikacji czy wybór następnej ścieżki odtwarzanej przez program iTunes. Niektóre telefony (na przykład T68 i T68i) umożliwiają zamianę aparatu telefonicznego w mysz i sterowanie dowolnymi aplikacjami. Wystarczy wybrać *System → Mouse mode*, a niewielki manipulator telefonu będzie służył do przemieszczania wskaźnika myszy.

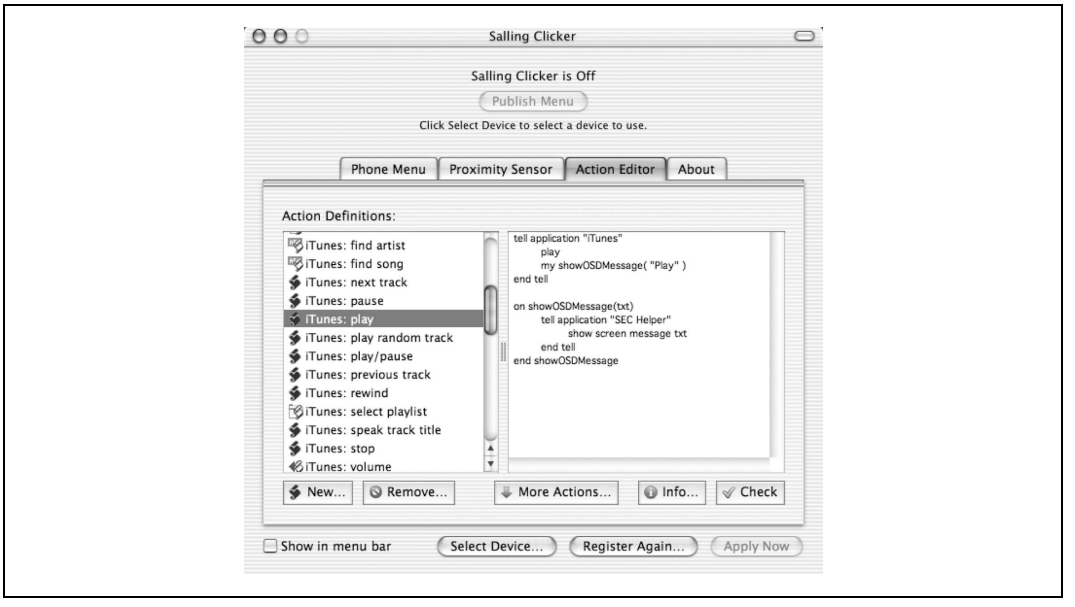
Ponieważ zasięg urządzeń Bluetooth wynosi nie więcej niż 10 metrów, możliwe jest wskazanie programowi Clicker akcji, którą ma podjąć, gdy telefon opuści zasięg Bluetooth oraz akcji, którą ma wykonać, gdy telefon powróci w obszar zasięgu. Funkcję tę można kontrolować na zakładce *Proximity Sensor* (rysunek 2.2). Można na przykład zdecydować, że w razie odsunięcia telefonu od komputera program Clicker zatrzyma odtwarzanie muzyki w programie iTunes i włączy wygaszacz ekranu. Obsługa tej funkcji jest bardzo prosta — wystarczy przeciągnąć akcję, która ma zostać wykonana do odpowiedniego pola i już można odejść.

Gdy akcje dostępne dla funkcji *Proximity Sensor* są niewystarczające, można zawsze utworzyć swoje własne. Akcje są skryptami napisanymi w języku AppleScript, więc wszystko, co da się zapisać za pomocą tego języka, może być uruchamiane za pomocą telefonu. W panelu sterowania programu Clicker na zakładce *Action Editor* (rysunek 2.3) można poddawać istniejące akcje edycji lub tworzyć swoje własne akcje.

Program Clicker posiada wygodną funkcję zdalnego sterowania pokazem slajdów programów PowerPoint lub Keynote, ale ponieważ może on symulować naciśnięcie dowolnego przycisku, można go stosować w praktycznie wszystkich programach. Szczególnie przydaje




*Rysunek 2.2. Funkcji Proximity Sensor można przypisać dowolną akcję*




*Rysunek 2.3. Jeżeli wbudowane akcje są niewystarczające, można napisać swoje własne*

się to przy prowadzeniu prezentacji, gdyż praktycznie zawsze ma się wtedy telefon ze sobą (i najprawdopodobniej jest on naładowany). Sam kupiłem kiedyś specjalnie do tego celu podłączany do portu USB pilot na podczerwień, ale w wielu sytuacjach, w których go potrzebowałem, nie miałem go akurat przy sobie, a nawet gdy go miałem, to nie było żadnej gwarancji, że jego baterie były naładowane, gdyż nie używałem go zbyt często.

Natomiast telefon komórkowy jest *zawsze* ze mną i jest praktycznie zawsze naładowany. Według mnie każdy program, który wykorzystuje moje przyzwyczajenia i lenistwo jest wart, aby za niego zapłacić.

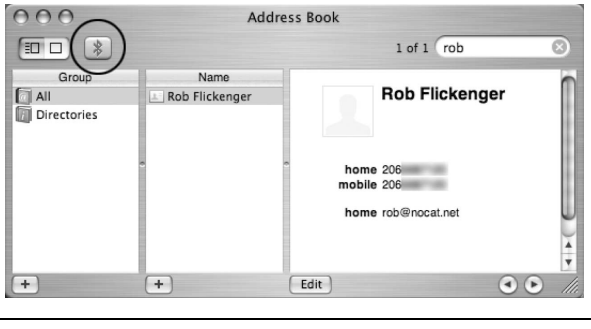
Większość ludzi uważa, że technologia Bluetooth służy jedynie do przesyłania danych albo głosu. Program Clicker jest aplikacją, która technologię Bluetooth wykorzystuje do czegoś więcej niż tylko do wyeliminowania kabli. Posiadacze telefonu Sony Ericsson używający systemu OS X mogą odkryć jeszcze wiele innych zastosowań tego programu. Jest on bardzo łatwy w użytkowaniu i może wykonać wszystko to, co można zapisać w postaci skryptu. Clicker jest aplikacją, która aż prosi się, żeby jej używać.

## SPOSÓB 14. Wpisywanie wiadomości SMS na klawiaturze komputera

Pora przestać marnować czas i zacząć pisać wiadomości SMS w laptopie.

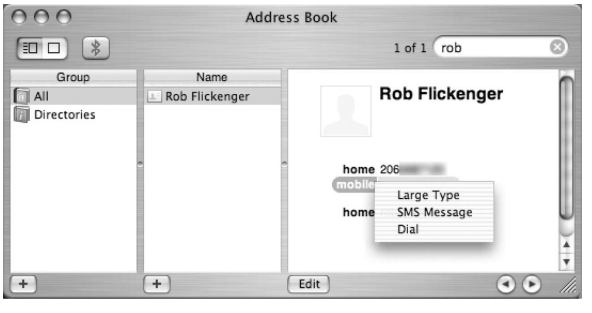
Wiadomości *SMS* (ang. *Short Message Service*), inaczej krótkie wiadomości tekstowe, stały się niezwykle popularne w wielu częściach świata (szczególnie w Japonii, na Filipinach i w większości krajów Europy), ale w Stanach Zjednoczonych z jakichś powodów zostały przyjęte mniej niż entuzjastycznie. W przypadku wielu osób jednym z powodów tej niechęci może być żmudny sposób wpisywania tekstu na klawiaturze telefonu. Presja na produkowanie coraz to mniejszych telefonów pozbawiła nas praktycznie nadziei na zastosowanie wygodnych, zintegrowanych z telefonem klawiatur. I chociaż technologie uzupełniające wpisywany tekst, takie jak *T9*, powodują, że w rezultacie wpisuje się mniejszą liczbę znaków, to interfejs telefonu nadal pozostaje mało intuicyjny. Wiele osób, próbując wyrazić swoje myśli, obsesyjnie naciska przyciski numeryczne, popełniając przy tym wiele błędów w pisowni. Natomiast wpisywanie znaków interpunkcyjnych jest tak niewygodne, że wiele osób nie zawraca sobie nimi głowy.

Istnieje jednak nadzieja dla posiadaczy telefonów wyposażonych w Bluetooth. System OS X bardzo dobrze integruje się z tymi urządzeniami oraz umożliwia obsługę wiadomości SMS. Najpierw trzeba się upewnić, że Bluetooth jest uaktywniony i laptop jest skonfigurowany do użycia odpowiedniego telefonu. Po uruchomieniu programu Address Book, gdy Bluetooth jest aktywny, w lewym górnym narożniku okna pojawia się ikona Bluetooth (rysunek 2.4). Po jej kliknięciu następuje integracja Bluetooth z programem Address Book.



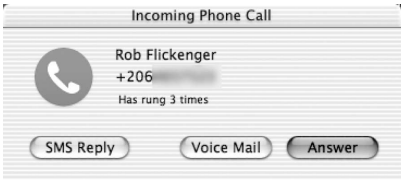

*Rysunek 2.4. Kliknięcie przycisku Bluetooth w programie Address Book powoduje integrację książki adresowej z Bluetooth*

Włączenie Bluetooth uaktywnia wiele przydatnych funkcji książki adresowej. Poza możliwością wybierania numerów telefonów bezpośrednio z książki adresowej uzyskuje się też możliwość wysyłania za jej pomocą wiadomości SMS. W tym celu należy kliknąć etykietkę znajdującą się z lewej strony numeru telefonu, do którego ma zostać wysłana wiadomość (rysunek 2.5), a następnie wybrać *SMS Message*. Pojawi się niewielkie okno tekstowe, w którym należy wpisać treść wiadomości SMS. Wreszcie można pisać wiadomości SMS na normalnej klawiaturze!




*Rysunek 2.5. Kliknięcie numeru telefonu w książce adresowej umożliwia wybranie numeru telefonu lub napisanie wiadomości SMS na klawiaturze komputera, a następnie jej wysłanie*

Program Address Book najlepiej implementuje także funkcję identyfikacji dzwoniącego. Gdy telefon dzwoni, program Address Book otwiera okno zawierające dane oraz numer telefonu osoby dzwoniącej (rysunek 2.6). Można w nim wybrać, czy chce się odebrać telefon, odesłać dzwoniącego do poczty głosowej lub wysłać mu wiadomość SMS.




*Rysunek 2.6. Dzwoniącą osobę można odesłać do poczty głosowej lub wysłać jej wiadomość SMS*

Naciśnięcie przycisku *SMS Reply* odsyła dzwoniącego do poczty głosowej, ale jednocześnie otwiera okno, w którym można napisać do niego wiadomość SMS. Przez cały czas, gdy uruchomiony jest program Address Book, nadchodzące wiadomości SMS są przez program automatycznie pokazywane i można na nie odpowiadać bezpośrednio z programu. Chociaż układ taki nie jest tak przenośny, jak sam telefon, użycie zwykłej klawiatury do wpisywania wiadomości SMS umożliwia szybsze i lepsze wyrażanie swoich myśli.

Przy okazji warto podkreślić, że wiadomości SMS przydają się szczególnie w miejscach, gdzie pokrycie siecią komórkową jest na tyle słabe, że uniemożliwia jednej lub obu stronom na prowadzenie rozmowy. W miejscach, w których połączenia głosowe są często

przerywane lub nawet są niemożliwe, próby wysłania wiadomości SMS będą podejmowane aż do skutku. Wiadomości SMS idealnie nadają się do wysłania krótkiej informacji komuś, z kimś nie można się połączyć w inny sposób. Wykorzystujący niewielką część pasma komunikat, który w jakikolwiek sposób może zostać przekazany, może być nieskończenie bardziej przydatny niż potrzebująca większej części pasma wiadomość głosowa, która nie może trafić do odbiorcy. W wielu sytuacjach wiadomości SMS, pomimo swoich ograniczeń, mogą być znakomitym sposobem komunikacji.

SPOSÓB 15.

## Automatyczne tworzenie blogów fotograficznych za pomocą telefonu Nokia 3650

Wykonywane zdjęcia można od razu publikować, bez potrzeby logowania się.

Cyfrowe fotografowanie ma swoje wady i zalety. Przyjemność obejrzenia zdjęcia od razu po jego zrobieniu w wielu przypadkach zależy od dwóch drobnostek — od tego, czy ma się aparat ze sobą i czy jego baterie są naładowane. Mój aparat jest stanowczo zbyt duży i zbyt delikatny, by nosić go stale ze sobą, a na dodatek nie cierpię mieć jeszcze jednego urządzenia, na które muszę zwracać uwagę i które muszę ładować w nocy. Z tych powodów większość moich zdjęć to zdjęcia okolicznościowe, a tylko na niewielu z nich znajdują się niezaimprowizowane scenki „z życia". Zbyt często zdarza się, że zanim przygotuję aparat, zdarzenie, które chciałem utrwalić — mija.

Kilku producentów doszło do wniosku, że istnieje takie urządzenie, które ludzie zwykle zawsze noszą ze sobą i prawie zawsze ma ono naładowaną baterię — jest nim telefon komórkowy. Firma Nokia poradziła sobie z umieszczeniem w jednym telefonie Nokia 3650 wielu znakomitych technologii — Bluetooth, GPRS, GSM oraz oczywiście aparatu cyfrowego. Daje to wiele różnego rodzaju możliwości. Za pomocą interfejsu Bluetooth zdjęcia można łatwo przesłać do laptopa, a ich wysłanie w postaci wiadomości e-mail jest również nadzwyczaj proste. Jakby mało jeszcze było spamu wysyłanego do komunikatorów, ludzie pracują teraz nad sposobem rozsyłania spamu zawierającego filmy wideo.

Zamiast rozsyłać zdjęcia znajomym i rodzinie, znacznie lepiej jest opublikować je w internecie w postaci albumu, a następnie wysłać odbiorcom jedynie łącze do tego albumu.

### Kod programu

Pierwszą czynnością, jaką należy wykonać, jest przygotowanie skryptu, który będzie odbierał wiadomości e-mail. Łatwo to zrobić za pomocą programu procmail. Poniższe wiersze należy dodać do pliku *.procmailrc* serwera pocztowego:

```
:0
* ^TO incub@helion.pl
| /home/username/bin/phonecam.sh
```

Oczywiście adres *incub@helion.pl* należy zamienić na adres email, który będzie wykorzystywany przez serwer obsługujący fotografie. Należy też zmienić ścieżkę na rzeczywistą, prowadzącą do miejsca, w którym został zapisany skrypt. Adres e-mail musi pozostać

poufny, ponieważ każde zdjęcie wysłane pod ten adres zostanie automatycznie opublikowane! Jeżeli serwer pocztowy nie używa programu procmail, należy zwrócić się o pomoc do zaprzyjaźnionego administratora sieci.

Następnie poniższy kod należy zapisać[1] w pliku o nazwie *phonecam.sh*, umieszczonym w katalogu wymienionym w pliku *.procmailrc*. Cały skrypt można pobrać ze strony *http://freenetworks.org/~mattw/badsoftware/phonecam/* (niżej zamieszczono jego nieco skróconą wersję). Zmienne występujące na początku skryptu należy dostosować do używanego systemu operacyjnego.

```
#!/bin/sh
#phonecam.sh
filepath="/home/incub/public_html/phonecam"
imgdir="img"
html="html"
time=`date +%s`
baseref="http://helion.pl/~incub/phonecam"
title="Phonecam wersja 3"
arcdate=`date +%D |sed '''s/\//./g'''`
perpage="16"

umask 062

if [ ! -f $filepath/count ]; then
  echo "0" > $filepath/count
fi

if [ ! -f $filepath/arc.txt ]; then
  touch $filepath/arc.txt
fi

if [ ! -d $filepath/archive ]; then
  mkdir $filepath/archive
fi

if [ ! -d $filepath/$html ]; then
  mkdir $filepath/$html
fi

if [ ! -d $filepath/$imgdir ]; then
  mkdir $filepath/$imgdir
fi

count=`head -1 $filepath/count`

mkdir ~/.$$
cd ~/.$$
munpack

for i in *.jpg; do
  a=`basename $i .jpg`
  mv $i $filepath/$imgdir/$time.jpg
  convert -resize 320x240 \
    $filepath/$imgdir/$time.jpg $filepath/$imgdir/$time.thumb.jpg

  convert -resize 150x90 $filepath/$imgdir/$time.jpg $filepath/latest.jpg
```

[1] Dla wygody Czytelników kody źródłowe wszystkich programów znajdujących się w tej książce zostały umieszczone w pliku *ftp://ftp.helion.pl/przyklady/100sie.zip — przyp. tłum.*

```
  # tworzenie nowej strony
  cat $filepath/new.txt > $filepath/new.tmp
  echo "<a href=\"$baseref/$html/$time.html\">
    <img src=\"$baseref/$imgdir/$time.thumb.jpg\"
    width=\"320\" height=\"240\" border=0></a>"
      > $filepath/new.txt

  cat $filepath/new.tmp >> $filepath/new.txt
  rm $filepath/new.tmp

  # tworzenie stron poszczególnych zdjęć
  echo "<html>
  <head><title>$title</title></head><body bgcolor=000000>
  <center><img src=\"$baseref/$imgdir/$time.jpg\" border=1></center><p>"
    > $filepath/$html/$time.html

  cat $a.desc >> $filepath/$html/$time.html

  echo "</body></html>" >> $filepath/$html/$time.html

  count=`expr $count + 1`
done

echo $count > $filepath/count

if [ $count = 1 ]; then
  echo "W kolejce znajduje się jedno zdjęcie" > $filepath/notify
else
  echo "W kolejce znajduje się $count zdjęć" > $filepath/notify
fi

if [ $count = $perpage ]; then
  echo "<html><head><title>$title</title></head><body bgcolor=000000><center>"
    > $filepath/archive/$time.html

  cat $filepath/index.txt >> $filepath/archive/$time.html
  cp $filepath/new.txt $filepath/index.txt
  rm $filepath/count
  rm $filepath/new.txt
  cat $filepath/arc.txt > $filepath/arc.tmp
  echo "<li><a href=\"$baseref/archive/$time.html\">$arcdate</a></li>"
    >> $filepath/arcn.txt

  cat $filepath/arc.tmp >> $filepath/arcn.txt
  rm $filepath/arc.tmp
  mv $filepath/arcn.txt $filepath/arc.txt

  echo "W kolejce nie ma żadnych zdjęć" > $filepath/notify
fi

rm -rf ~/.$$
```

Poza tym skryptem potrzebne będą jeszcze programy *munpack* (dekodujący zawartość MIME załączników) oraz *convert* (należący do pakietu Image Magick). Narzędzia te znajdują się we wszystkich standardowych dystrybucjach Linuks.

Na koniec należy przygotować plik *index.shtml* i umieścić go w katalogu głównym dokumentów serwera WWW. Plik ten powinien zawierać wiersz:

```
<!--#include virtual="index.txt"-->
```

Bardziej zaawansowany przykład pliku *index.shtml* znajduje się na stronie: *http://freenetworks.org/~mattw/badsoftware/phonecam/index.shtml.txt*

## Uruchomienie programu

Jeśli mamy już wszystko przygotowane, wystarczy wysłać zdjęcie jako wiadomość e-mail pod skonfigurowany adres. Skrypt automatycznie zdekoduje wiadomość e-mail, utworzy miniaturkę zdjęcia i umieści zdjęcie w kolejce. Gdy w kolejce znajdzie się określona przez parametr *perpage* liczba zdjęć, utworzona zostanie z nich nowa, pełna strona, a poprzednia zostanie przeniesiona do archiwum. Najnowsze zdjęcie zawsze znajduje się pod adresem *http://server/~incub/phonecam/latest.jpg*, a zdjęcia oczekujące w kolejce znajdują się pod adresem *http://server/~incub/phonecam/new.txt*. Skrypt zarządza kolejką i archiwami bez potrzeby interwencji użytkownika, a nawet może umieszczać opisy publikowanych zdjęć. W tym celu wystarczy umieścić opis zdjęcia w treści wiadomości e-mail.

Skrypt ten najprawdopodobniej można jeszcze uprościć lub rozwinąć, ale w tej prostej postaci może działać praktycznie na każdym serwerze. Tworzy prosty, ale efektywny interfejs WWW, który łatwo można zintegrować z blogiem lub inną istniejącą stroną WWW. A poza tym radość, jaką dają zdjęcia cyfrowe, dotyczy teraz także możliwości ich natychmiastowej publikacji.

## Inne źródła informacji

- Projekt phonecam Matta Westervelta (*http://freenetworks.org/~mattw/badsoftware/phonecam/*).
- Image Magick (*http://www.imagemagick.org/*).

SPOSÓB 16.

# Bluetooth w systemie Linux

W Linuksie 2.4 można szybko uruchomić obsługę Bluetooth.

Jak się tego można spodziewać, uruchomienie obsługi interfejsu Bluetooth w Linuksie wymaga trochę więcej pracy niż w innych systemach operacyjnych. Najpierw wypada zauważyć, że obecnie istnieją trzy różne stosy protokołu Bluetooth przeznaczone dla Linuksa — Affix, OpenBT oraz BlueZ. Każdy z nich w inny sposób obsługuje różne adaptery Bluetooth i każdy dysponuje innymi środkami konfiguracyjnymi. „Oficjalnym" stosem protokołu Bluetooth w Linuksie został wybrany BlueZ i to on zostanie dalej omówiony.

Najpierw należy sprawdzić, czy posiadany adapter Bluetooth jest obsługiwany przez BlueZ. Dość aktualną listę obsługiwanych urządzeń można znaleźć pod adresem *http://www.holtmann.org/linux/bluetooth/devices.html*.

Następnie należy sprawdzić, czy w jądrze systemu jest uruchomiona obsługa Bluetooth. Jądra dostarczane w dystrybucjach Red Hat 9.0 oraz Debian „Sarge" zawierają już obsługę Bluetooth. Żeby sprawdzić, czy jądro obsługuje Bluetooth, należy jako użytkownik root wydać polecenia `modprobe rfcomm`. Jeżeli uruchomienie programu modprobe nie powiedzie się, należy przebudować jądro.

W przypadku niepomyślnego uruchomienia programu modprobe należy zbudować i zainstalować czystą kopię jądra Linuksa w wersji 2.4.21 lub nowszej (lub wersję 2.4.20 z łatką –mh6). Konfigurując jądro, należy zaznaczyć, żeby wszystkie opcje pozycji „Bluetooth support" zostały zbudowane jako oddzielne moduły, przy czym opcja „USB Bluetooth support" w „USB support" powinna być wyłączona, gdyż w przeciwnym razie do obsługi UART zastosowany zostanie stos protokołu OpenBT, który będzie przeszkadzać w działaniu stosu BlueZ. Nowsze wersje jądra po wybraniu BlueZ automatycznie wyłączają tę ostatnią opcję.

Trzeba poczynić jeszcze jedną uwagę. Jeżeli używamy nowszych laptopów firm Toshiba lub Sony i chcemy wykorzystać wbudowany adapter Bluetooth, należy uaktywnić w sekcji „Processor type and features" oraz „Character devices" opcje jądra właściwe dla produktów tych firm. Do włączenia obsługi Bluetooth w tych komputerach potrzebne będą też specjalne narzędzia użytkownika. Omówienie tych narzędzi wykracza poza zakres tej książki, a więcej informacji na ich temat można znaleźć na liście urządzeń obsługiwanych przez BlueZ, o której wspomniano wcześniej w tym podrozdziale.

Następnie do pliku */etc/modules.conf* należy dodać następujące wiersze:

```
alias net-pf-31 bluez
alias bt-proto-0 l2cap
alias bt-proto-2 sco
alias bt-proto-3 rfcomm
alias bt-proto-4 bnep
alias tty-ldisc-15 hci_uart
alias bluetooth off
```

Teraz jako użytkownik root należy wykonać polecenie `/sbin/depmod -a`.

Opcje te informują jądro, które moduły mają zostać załadowane, gdy żądana jest obsługa Bluetooth. Ostatnia opcja — `alias bluetooth off` — informuje program modprobe, by ten nie ładował modułu OpenBT UART, gdyby moduł ten został zainstalowany przez przypadek.

Teraz potrzebne są narzędzia użytkownika BlueZ. W systemie Red Hat 9.0 wszystkie te narzędzia są już zainstalowane. Ich kod źródłowy, a także moduły RPM i pakiety *.deb* można pobrać ze strony głównej projektu BlueZ pod adresem *http://bluez.sourceforge.net*. Należy zbudować i (albo) zainstalować pakiety bluez-libs, bluez-utils, bluez-sdp oraz bluez-hcidump. W systemie Debian wystarczy po prostu uruchomić narzędzie apt-get; w tym systemie biblioteka „bluez-libs" nazywa się „libbluetooth1", a „libsdp2" umieszczona jest w oddzielnym pakiecie.

Teraz kilka słów o konfiguracji urządzeń opartych na układzie UART (czyli nie używających USB). W przypadku użycia adaptera Bluetooth USB można ten akapit opuścić. Urządzenia typu szeregowego, do których należą konwertery szeregowe i karty PCMCIA, muszą zostać jawnie „dołączone" do interfejsu kontrolera Bluetooth za pomocą programu `hciattach`. Po podłączeniu urządzenia odpowiedni sterownik jądra może załadować się automatycznie, czemu towarzyszyć będzie odpowiedni zapis w dzienniku */var/log/messages*. Do urządzeń opartych na układzie UART istnieją odniesienia w postaci */dev/ttySn*, gdzie *n* oznacza jakąś liczbę całkowitą. Urządzenie dołącza się do kontrolera Bluetooth,

wydając polecenie `/sbin/hciattach /dev/ttySn`. Tak, jak w przypadku dobrych narzędzi uniksowych, jeżeli nie pojawi się żaden komunikat, to oznacza to, że program `hciattach` zadziałał prawidłowo. W przeciwnym razie należy sprawdzić, czy użyte zostało odpowiednie urządzenie oraz odwołać się do dokumentacji `man`.

Po prawidłowym uruchomieniu programu `hciattach` w pliku */etc/bluetooth/uart* należy umieścić odwołanie do urządzenia — po to, by urządzenie zostało dołączone do kontrolera Bluetooth w czasie uruchamiania systemu. Jeżeli plik ten nie istnieje, należy go utworzyć. W pliku należy umieścić pojedynczy wiersz „*/dev/ttySn any*" i zastąpić „*n*" odpowiednim numerem urządzenia szeregowego. Jeżeli używamy adaptera USB, oczywiście nie trzeba wykonywać tej czynności.

Teraz, kiedy wszystko jest już zainstalowane, należy podłączyć adapter Bluetooth i jako użytkownik root wydać polecenie `etc/rc.d/init.d/bluetooth start`. W systemie Debian należy wydać polecenie `/etc/init.d/bluez-utils start; /etc/init.d/bluez-sdp start`. W pliku dziennika */var/log/messages* powinny pojawić się odpowiednie komunikaty stanu. Jeżeli instalacja BlueZ nie zawierała skryptu */etc/rc.d/init.d/bluetooth*, odpowiednią jego wersję można przekopiować z podkatalogu *scripts/* pakietu bluez-utils. Zakładając, że wszystko działa poprawnie, za pomocą polecenia `chkconfig` lub za pomocą dowiązania ręcznego można dodać skrypt Bluetooth do odpowiedniego katalogu *rc.d* domyślnego poziomu startowego.

Następnie należy uruchomić program `hciconfig`. Powinny się pojawić następujące informacje:

```
hci0:   Type: USB
        BD Address: 00:11:22:33:44:55 ACL MTU: 192:8  SCO MTU: 64:8
        UP RUNNING PSCAN ISCAN
        RX bytes:99 acl:0 sco:0 events:13 errors:0
        TX bytes:296 acl:0 sco:0 commands:12 errors:0
```

Jeżeli nic się takiego nie pojawiło, należy sprawdzić, czy uruchomiony jest program `hcid` i czy w dzienniku zdarzeń */var/log/messages* nie ma żadnych komunikatów o błędach. Widoczny wyżej adres BD (*BD Address*) jest niepowtarzalnym identyfikatorem adaptera Bluetooth, podobnym do adresu MAC urządzeń Ethernet.

Teraz w polu działania adaptera Bluetooth należy umieścić inne urządzenie Bluetooth i sprawdzić, czy jest ono wykrywane przy skanowaniu. Następnie należy wydać polecenie `hcitool scan`. Skanowanie może zająć od 15 do 20 sekund, po których powinny pojawić się informacje typu:

```
$ hcitool scan
Scanning ...
        00:99:88:77:66:55       Nokia3650
```

Teraz wydając polecenie `sdptool browse 00:99:88:77:66:55`, można przetestować urządzenie i sprawdzić, jakiego rodzaju usługi świadczy. Powinna pojawić się długa lista oferowanych usług, zawierająca informacje, które można wykorzystać przy konfiguracji dostępu do tych usług.

*— Schuyler Erle*

# SPOSÓB 17. Korzystanie w Linuksie z połączeń GPRS poprzez Bluetooth

Tam, gdzie nie ma sieci Wi-Fi, telefonu z Bluetooth można użyć jako modemu.

Bez wątpienia sama możliwość skanowania najbliższych urządzeń Bluetooth z komputera z systemem Linux szybko przestanie być atrakcyjna i przyjdzie pora na *wykorzystanie* nowego połączenia Bluetooth do jakiegoś konkretnego celu. Czyż nie byłoby interesujące móc używać telefonu komórkowego jako modemu wszędzie tam, gdzie nie ma sieci Wi-Fi?

Bluetooth obsługuje kilka „profili" definiujących sposób, w jaki urządzenia Bluetooth komunikują się ze sobą. W tym przypadku należy skorzystać z profilu połączeń komutowanych *DUN* (ang. *Dial-up Networking*), wykorzystującego protokół RFCOMM, emulujący łącze szeregowe pomiędzy dwoma urządzeniami. Za pomocą protokołu RFCOMM można połączyć komputer z telefonem, a następnie — uruchamiając program *pppd* — uzyskać dostęp do Internetu. Program ten działa na łączach GPRS, a nawet na zwyczajnych łączach komutowanych.

Jeżeli obsługa Bluetooth jest już skonfigurowana **[Sposób 16.],** należy umieścić telefon w zasięgu komputera i rozpocząć jego skanowanie za pomocą programu `hcitool`. Załóżmy, że skanowanie się powiodło i że program `hcitool` poinformował, że adresem BD telefonu jest *00:11:22:33:44:55*.

To, czy w zasięgu znajduje się urządzenie obsługujące profil DUN, można sprawdzić też za pomocą polecenia `sdptool`:

```
$ sdptool search DUN
Inquiring ...
Searching for DUN on 00:11:22:33:44:55 ...
Service Name: Dial-up Networking
Service RecHandle: 0x10001
Service Class ID List:
  "Dialup Networking" (0x1103)
  "Generic Networking" (0x1201)
Protocol Descriptor List:
  "L2CAP" (0x0100)
  "RFCOMM" (0x0003)
    Channel: 1
```

Numer kanału należy zanotować — przyda się później. Jak widać, programy `hcitool` i `sdptool` oferują dużo przydatnych funkcji diagnostycznych Bluetooth, o których można się dowiedzieć z odpowiednich stron dokumentacji `man`.

Jednak przed połączeniem się z telefonem należy pomiędzy Linuksem a telefonem skonfigurować tak zwane *parowanie urządzeń* (ang. *device pairing*), za sprawą którego telefon będzie „wiedział", że ma umożliwić dostęp komputerowi do swoich usług i (być może) na odwrót. Kod PIN komputera znajduje się w pliku */etc/bluetooth/pin* i powinno się go zmienić na inny — poufny.

Większość telefonów ma PIN Bluetooth, który można skonfigurować w telefonie. W skład pakietu BlueZ wchodzi niewielki, napisany w języku Python program `bluepin`, który w razie potrzeby uruchamia okno dialogowe GTk+ i prosi o podanie numeru PIN. Ten

program najwyraźniej jednak nie działa we wszystkich dystrybucjach Linuksa. Co więcej, kto chciałby być za każdym razem proszony o kod PIN? Kody PIN różnych urządzeń Bluetooth można zapisać w poniższym skrypcie napisanym w języku Perl, który należy umieścić w pliku */etc/bluetooth/pindb*:

```
#!/usr/bin/perl
while ( ) {
    print "PIN:$1\n" if /^$ARGV[1]\s+(\w+)/o;
}
__DATA__
# Poniżej należy wpisać kody PIN Bluetooth w parach - adres_BD kod_PIN
# - oddzielone białym znakiem, po jednej parze w każdym wierszu.
#
# ## itd.

00:11:22:33:44:55       11111
```

Właścicielem pliku */etc/bluetooth/pindb* powinien być użytkownik root, a do pliku należy nadać uprawnienia dostępu *chmod 0700* — wszystko po to, by zwykli użytkownicy nie mogli podejrzeć kodów PIN urządzeń Bluetooth. Odpowiednia sekcja opcji pliku */etc/bluetooth/hcid.conf* powinna wyglądać tak, jak przedstawiono poniżej:

```
options {
        autoinit yes;
        security auto;
        pairing multi;
        pin_helper /etc/bluetooth/pindb;
}
```

W ten sposób urządzenia HCI są konfigurowane w momencie uruchamiana się systemu operacyjnego. Możliwe jest parowanie urządzeń, a *hcid* będzie zgłaszał programowi *pindb* żądanie podania numeru PIN w przypadku każdego urządzenia. Po dokonaniu zmian w pliku */etc/bluetooth/hcid.conf* należy ponownie uruchomić program *hcid* za pomocą polecenia `/etc/rc.d/init.d/bluetooth restart`.

Teraz, kiedy komputer jest skonfigurowany do parowania, podobnie należy skonfigurować telefon. W tym celu należy zapoznać się z instrukcją obsługi telefonu. Proces konfiguracji telefonu często wymaga, by telefon mógł przeskanować adapter Bluetooth komputera, wobec czego komputer musi znaleźć się w zasięgu telefonu, a adapter Bluetooth musi być włączony. Jego interfejs otrzyma prawdopodobnie nazwę „BlueZ(0)" lub podobnie, chyba że w pliku *hcid.conf* zmieniono opcję „name". Parowanie w telefonie należy skonfigurować jako „trusted" (zaufane) lub jego odpowiednik — po to, by za każdym razem, gdy wykonywane będzie połączenie z systemu Linux, użytkownik telefonu nie był proszony o jego zweryfikowanie.

Teraz, kiedy w pobliżu znajduje się oferujące połączenie komutowane urządzenie, z którym ustanowiono parowanie, następną czynnością jest powiązanie z tym urządzeniem interfejsu RFCOMM. Najpierw za pomocą polecenia `ls -l /dev/rfcomm*` należy sprawdzić, czy w katalogu */dev* znajdują się pozycje RFCOMM. Jeżeli polecenie `ls` nie znajdzie poszukiwanych plików („No such file or directory"), należy utworzyć 64 pozycje urządzeń RFCOMM, przełączając się na konto superużytkownika i wykonując polecenia:

```
# for n in `seq 0 63`; do mknod -m 660 /dev/rfcomm$n c 216 $n; done
# chown root:uucp /dev/rfcomm*
```

W przypadku systemu Debian za pomocą polecenia `chown` należy przypisać urządzenia RFCOMM do grupy *dialout*, a nie *uucp*.

Teraz jako superużytkownik należy — za pomocą polecenia `rfcomm` pakietu bluez-utils — powiązać urządzenie */dev/rfcomm0* z telefonem na kanale, którego numer dla profilu DUN uzyskano wcześniej za pomocą programu `sdptool`:

```
# rfcomm bind /dev/rfcomm0 00:11:22:33:44:55:66 1
```

Jeżeli powiązanie przebiegło pomyślnie, to — jak przystało na dobre polecenie uniksowe — program `rfcomm` nie odpowie żadnym komunikatem. Że faktycznie operacja została wykonana poprawnie, można dowiedzieć się, wydając polecenie `rfcomm` bez argumentów:

```
# rfcomm
rfcomm0: 00:11:22:33:44:55 channel 1 clean
```

Teraz to urządzenie szeregowe można traktować tak, jak zwykły modem. Żeby to udowodnić, można — jako użytkownik root — uruchomić program `minicom` i przełączyć urządzenie szeregowe na */dev/rfcomm0*. Gdy program terminala uruchomi się, należy wpisać **`AT`** i nacisnąć Enter. Jeżeli telefon odpowie komunikatem `OK`, można sobie pogratulować — ustanowiono połączenie z telefonem komórkowym poprzez Bluetooth.

Zanim wykonana zostanie następna czynność, w pliku */etc/bluetooth/rfcomm.conf* należy wpisać poniższe wiersze, dzięki którym urządzenie RFCOMM będzie konfigurowane podczas uruchamiania Bluetooth,:

```
rfcomm0 {
        # Automatycznie wiąże urządzenie w momencie uruchamiania
        bind yes;
        device 00:11:22:33:44:55;
        channel 1;
        comment "Moj telefon";
}
```

Jeżeli używamy narzędzi bluez-utils 2.3 lub wcześniejszych w dystrybucjach innych niż oparte na systemie Debian, należy do sekcji `start()` pliku */etc/rc.d/init.d/bluetooth* dodać wiersz zawierający polecenie `rfcomm bind all`.

Do połączenia się z internetem pozostało już tylko wykonanie niewielkiego kroku. W pliku */etc/ppp/peers/gprs* należy umieścić następujące wiersze:

```
/dev/rfcomm0

connect '/usr/sbin/chat -v -f /etc/ppp/peers/gprs.chat'
noauth
defaultroute
usepeerdns
lcp-echo-interval 65535
debug
```

Natomiast poniższe wiersze należy zapisać w pliku */etc/ppp/peers/gprs.chat*:

```
TIMEOUT         15
ECHO            ON
HANGUP          ON
''              AT
OK              ATZ
OK              ATD*99#
```

W przypadku korzystania z programu `wvdial` do pliku */etc/wvdial.conf* należy dodać:

```
[Dialer gprs]
Modem       = /dev/rfcomm0
Phone       = *99#
Username    = foo
Password    = bar
```

Europejscy dostawcy usług przeważnie nadają użytkownikowi jego nazwę oraz hasło; w Stanach Zjednoczonych programowi `wvdial` ciągle jeszcze podaje się wartości fikcyjne. Tego, co dokładnie należy wpisać, można dowiedzieć się na stronie WWW dostawcy usług. Połączenie GPRS jest uwierzytelnione przez samo działanie telefonu w sieci komórkowej, więc do użycia protokołu PPP nie jest potrzebne dodatkowe uwierzytelnienie. Podany numer telefonu jest standardowym numerem dostępowym GPRS, który — gdy telefon jest skonfigurowany poprawnie — powinien zapewnić natychmiastowe połączenie. Jednak większość telefonów GSM obsługuje kilka punktów dostępowych GPRS, wobec czego w przypadku, gdy domyślne ustawienia telefonu nie są odpowiednie, należy w programie minicom wpisać polecenie **`AT+CGDCONT?`** i nacisnąć klawisz Enter. Telefon wyświetli listę dostępnych profili PDP (ang. *Packet Data Protocol*). Z listy tej należy wybrać profil, który wydaje się najbardziej odpowiedni, a następnie zmienić numer telefonu GPRS w pliku */etc/wvdial.conf* na `*99***`*`n`*`#`, zastępując *`n`* numerem profilu PDP, który ma zostać zastosowany. Gdyby i ten sposób zakończył się niepowodzeniem, należy zwrócić się o pomoc do dostawcy usług.

Konfigurację tę można przetestować jako użytkownik root, wydając — w zależności od konfiguracji — polecenie `pppd call gprs` lub `wvdial gprs` i jednocześnie obserwując w drugim oknie dziennik zdarzeń */var/log/messages*. Jedyną niedogodnością tej konfiguracji jest to, że w pliku */etc/resolv.conf* nie są domyślnie umieszczane serwery nazw. Sposobem na to jest umieszczenie — w przypadku systemu Red Hat — w pliku */etc/sysconfig/network-scripts/ifcfg-ppp0* (lub, gdy trzeba, *ppp1*, *ppp2* i tak dalej) następujących wierszy:

```
# Gdy używa się programu wvdial, należy za pomocą znaku komentarza
# wyłączyć zmienną CHATSCRIPT, natomiast włączyć zmienną WVDIALSECT.
DEVICE=ppp0
MODEMPORT=/dev/rfcomm0
CHATSCRIPT=/etc/ppp/peers/gprs.chat
# WVDIALSECT=gprs
```

W ten sposób połączenie można włączać i wyłączać za pomocą poleceń `ifup ppp0` i `ifdown ppp0`. Żeby to samo uzyskać w systemie Debian, należy zastosować pokazaną konfigurację demona *pppd* i do pliku */etc/network/interfaces* dodać wiersze:

```
iface ppp0 inet ppp
    provider gprs
```

Żeby DNS działał prawidłowo w przypadku dystrybucji innych niż Red Hat i Debian, do pliku */etc/ppp/peers/gprs* trzeba dodać poniższe wiersze; połączenie należy wówczas włączać i wyłączać za pomocą poleceń `pppd call gprs` i `killall pppd`:

```
welcome 'cp -b /etc/ppp/resolv.conf /etc/resolv.conf'
disconnect 'mv /etc/resolv.conf~ /etc/resolv.conf'
```

W zasadzie to wszystko, co trzeba zrobić, by połączyć się z internetem z dowolnego miejsca, w którym dostępne są usługi GSM. Nie należy się spodziewać jakiś nadzwyczajnych prędkości — obecnie prędkości w GPRS wahają się od 5 do 20 Kbps, w zależności od usługi, co według obecnych standardów nie jest dużą prędkością, ale i tak można ją uznać za dobrą tam, gdzie nie dysponuje się niczym innym.

Jako dodatek przedstawiony zostanie skrypt *iptables*, umożliwiający udostępnianie połączenia GPRS innym użytkownikom w zasięgu sieci Wi-Fi. Skrypt należy zapisać i uruchamiać z pliku */etc/ppp/ip-up.local*:

```
# Włączenie przekazywania IP oraz filtra rp_filter
# (w celu uniemożliwiania podszywania się pod adres IP).
echo "1" > /proc/sys/net/ipv4/ip_forward
echo "1" > /proc/sys/net/ipv4/conf/all/rp_filter

# Jeżeli trzeba, ładowane są odpowiednie moduły jądra.
for i in ip_tables ipt_MASQUERADE iptable_nat
    ip_conntrack ip_conntrack_ftp ip_conntrack_irc \
    ip_nat_irc ip_nat_ftp; do
    modprobe $i 2>/dev/null;
done

# Maskowanie wszystkiego, co przychodzi spoza interfejsu PPP
# (np. ethernet, Wi-Fi itp.).
iptables -t nat -A POSTROUTING -o ppp+ -j MASQUERADE
```

A co ze zwykłymi połączeniami komutowanymi? Z wysyłaniem faksów? Okazuje się, że wystarczy zamienić numer dostępowy GPRS na zwykły numer telefonu i (w większości telefonów) uzyska się z tym numerem połączenie o prędkości 9600 bodów. W tej sytuacji konfiguracja programów *efax* lub *mgetty-sendfax* do użycia połączenia Bluetooth do wysyłania faksów z telefonu GSM zostanie pozostawiona jako ćwiczenie dla Czytelnika.

*— Schuyler Erle*

SPOSÓB 18.

## Przesyłanie plików połączeniami Bluetooth w Linuksie

Komputer z systemem Linux i urządzenie Bluetooth mogą wymieniać między sobą dane.

Połączenie się z internetem **[Sposób 17.]** z dowolnego miejsca przez telefon komórkowy jest świetnym pomysłem, ale telefon najprawdopodobniej ma jeszcze inne funkcje. Może ktoś przesłał do telefonu zdjęcie z przyjęcia rodzinnego i teraz trzeba je przekopiować do laptopa lub trzeba zainstalować w nowym telefonie kilka programów.

Podstawą mechanizmu przesyłania plików poprzez połączenia Bluetooth jest protokół OBEX (ang. *Object Exchange*), umożliwiający przesyłanie plików binarnych nie tylko przez połączenia Bluetooth, ale również przez połączenia wykorzystujące podczerwień, a nawet

za pomocą tradycyjnego protokołu TCP/IP. Najbardziej rozpowszechnioną implementacją *open source* protokołu OBEX jest projekt OpenOBEX (*http://openobex.sf.net/*). Pakiety *libopenobex* oraz *openobex-apps* można otrzymać z dystrybucją *sid*, a pakiety *openobex* dla systemu Red Hat można pobrać z serwera SourceForge lub poprzez *rpmfind.net*. Bluetooth obsługuje obecnie dwa różne profile przesyłania plików — OBEX Push, którego podstawowym zastosowaniem jest przesyłanie do urządzenia Bluetooth pojedynczych plików oraz OBEX File Transfer, oferujący bogatszy zestaw operacji wymiany plików.

Niestety technologia przesyłania plików przez Bluetooth w systemie Linux podlega jeszcze nieustannym zmianom. Pakiet *openobex-apps* zawiera program *obex_test*, realizujący dość elementarny sposób wysyłania plików do urządzeń Bluetooth. Najpierw należy się dowiedzieć, który numer kanału Bluetooth wykorzystuje do przesyłu OBEX File Transfer telefon lub inne urządzenie. W tym celu należy wpisać:

```
# sdptool search FTRN
Inquiring ...
Searching for FTRN on 00:11:22:33:44:55 ...
Service Name: OBEX File Transfer
Service RecHandle: 0x10003
Service Class ID List:
  "OBEX File Transfer" (0x1106)
Protocol Descriptor List:
  "L2CAP" (0x0100)
  "RFCOMM" (0x0003)
    Channel: 10
  "OBEX" (0x0008)
```

Teraz można spróbować połączyć się z urządzeniem za pomocą polecenia `obex_test` z opcją `-b` (Bluetooth), adresem BD urządzenia oraz numerem kanału FTRN:

```
$ obex_test -b 00:11:22:33:44:55 10
Using Bluetooth RFCOMM transport
OBEX Interactive test client/server.
> c
Connect OK!
Version: 0x10. Flags: 0x00
> x
PUSH filename> /home/incub/images/image.jpg
name=/home/incub/images/image.gif, size=7294
Going to send /home/incub/images/image.gif(opt21.gif), 7294 bytes
Filling stream!
```

`obex_test` pokazuje jeszcze kilka komunikatów informujących o wykonywanych czynnościach, po których następuje potwierdzenie:

```
Made some progress...
Made some progress...
Made some progress...
Filling stream!
PUT successful!
```

Plik powinien pojawić się w urządzeniu. Analogicznie, po skonfigurowaniu interfejsu Bluetooth do odpowiadania na żądania OBEX Push, program `obex_test` może odbierać pliki wysyłane z urządzenia. Interfejs Bluetooth należy skonfigurować za pomocą programu `sdptool`, podając mu ten sam numer kanału, którego urządzenie używa dla funkcji OBEX Push:

```
$ sdptool add --channel=10 OPUSH
$ obex_test -b 00:11:22:33:44:55 10
Using Bluetooth RFCOMM transport
OBEX Interactive test client/server.
> s
```

*obex_test* powinien wyświetlić pusty wiersz, a następnie zatrzymać się. Teraz można wysłać plik z telefonu. Pojawi się mnóstwo komunikatów, po czym *obex_test* poinformuje o zakończeniu przesyłania pliku. Przesłany plik trafi do katalogu */tmp*.

Problem z programem `obex_test` — oprócz tego, że nie istnieje jakakolwiek dokumentacja do tego programu — polega również na tym, że można go używać wyłącznie w sesji interaktywnej. A co w przypadku, gdy chce się sterować przesyłaniem plików za pomocą skryptów lub zautomatyzować odbieranie plików? Inny, chyba łatwiejszy sposób wysyłania plików z komputera polega na użyciu niewielkiego programu `ussp-push`, który można znaleźć pod adresem *http://www.unrooted.net/hacking/ussp-push.tgz*. Program `ussp-push` wykorzystuje część kodu stosu Affix, ale w rzeczywistości opiera się na OpenOBEX. W chwili obecnej nie można go też od razu skompilować, dlatego trzeba dostosować go do najnowszych wersji OpenOBEX za pomocą skryptu napisanego w języku Perl:

```
# tar xfz ussp-push.tgz
# cd ussp-push
# perl -pi -e 's/custfunc\.userdata/custfunc.customdata/g' obex_main.c
# make
...
# cp ussp-push /usr/local/bin
```

Program `ussp-push` wykorzystuje urządzenie szeregowe RFCOMM, powiązane z kanałem Bluetooth używanym w funkcji OBEX Push. Z tego powodu trzeba ponownie użyć programu `sdptool`, a następnie za pomocą programu `rfcomm` należy powiązać urządzenie z wymienionym kanałem:

```
# sdptool search OPUSH
Inquiring ...
Searching for OPUSH on 00:11:22:33:44:55 ...
Service Name: OBEX Object Push
Service RecHandle: 0x10004
Service Class ID List:
  "OBEX Object Push" (0x1105)
Protocol Descriptor List:
  "L2CAP" (0x0100)
  "RFCOMM" (0x0003)
    Channel: 9
  "OBEX" (0x0008)

# rfcomm bind /dev/rfcomm1 00:11:22:33:44:55 9
```

Teraz za pomocą programu *ussp-push* można wysyłać pliki do urządzenia.

```
$ ussp-push /dev/rfcomm1 /home/incub/images/image.jpg image.jpg
```

Programowi `ussp-push` podaje się trzy argumenty — nazwę urządzenia RFCOMM, nazwę pliku do przesłania oraz nazwę, pod jaką plik ma zostać zapisany po drugiej stronie. Po uruchomieniu programu pojawi się lawina komunikatów, kończąca się informacją

o przekopiowaniu pliku. Jeżeli dla nowego urządzenia RFCOMM utworzony zostanie zapis w pliku */etc/bluetooth/rfcomm.conf*, urządzenie to zostanie dowiązane podczas uruchamiania systemu. Poniższemu skryptowi należy nadać nazwę `btpush` i zapisać go w jakimś wygodnym miejscu (na przykład w swoim katalogu *~/bin*). Od tej pory można go używać do zautomatyzowanego wysyłania plików do telefonu:

```
#!/bin/bash

# btpush - wysyła pliki do urządzenia Bluetooth, wykorzystując program ussp-push.
# Skrypt uruchamia się poleceniem: btpush <nazwa pliku>

ussp-push /dev/rfcomm1 $1 `basename $1`
```

Jeżeli podczas używania programu `ussp-push` pojawiają się jakieś problemy z uprawnieniami, należy sprawdzić, czy urządzenia */dev/rfcomm** mogą być zapisywane przez grupę, której jest się członkiem i czy ich właścicielem jest ta grupa.

Pliki przesyłać można również za pomocą innego małego programu `obexserver`, dostępnego pod adresem *http://www.frasunek.com/sources/unix/obexserver.c*. Program *obexserver* należy zbudować łącznie z programami *openobex-apps*. Po zbudowaniu *openobex-apps* w najwyższym katalogu plików źródłowych należy wykonać polecenia:

```
# cd src
# wget http://www.frasunek.com/sources/unix/obexserver.c
# gcc -o obexserver obexserver.c libmisc.a -lopenobex
# cp obexserver /usr/local/bin
```

Żeby za pomocą tego programu móc odbierać pliki, należy najpierw skonfigurować w komputerze usługę OBEX Push, wydając polecenie `sdptool add --channel=10 OPUSH`, w którym numer kanału należy zmienić na taki, którego urządzenie używa w funkcji OBEX Push. Następnie program `obexserver` należy uruchomić bez argumentów i wysłać plik z telefonu. Program *obexserver* odbierze plik, zapisze go w katalogu */tmp*, a następnie zakończy działanie.

Możliwość wysyłania plików z telefonu do komputera i w drugą stronę jest bardzo miła, ale wysyłanie plików pojedynczo nie jest zbyt wygodne. Na szczęście gdy urządzenie Bluetooth wyposażone jest w system SymbianOS lub EPOC, jak to jest w przypadku telefonów Series 60 oraz Ericsson P800, można zamontować w komputerze system plików urządzenia za pomocą NFS. Pakiet, który dokonuje tego niewielkiego cudu, nazywa się *p3nfs* i można go pobrać ze strony *http://www.koeniglich.de/p3nfs.html*. Plikom źródłowym towarzyszą pliki binarne przeznaczone do uruchamiania w telefonie, przygotowane na wypadek, gdyby ktoś nie dysponował kompilotorem skrośnym (co zdarza się często). Program przeznaczony dla telefonu nazywa się `nfsapp`, a jego wersję binarną, odpowiednią dla danego rodzaju telefonu, można znaleźć w podkatalogu *bin/* — ma ona rozszerzenie *.sis*. Plik ten należy za pomocą jednej z opisanych wcześniej metod przekopiować do telefonu i zainstalować go tam. Następnie w tradycyjny sposób należy zbudować i zainstalować program *p3nfs*. Jeżeli użyte zostały pobrane ze strony *koeniglich.de* moduły RPM, pliki SIS trafią do katalogu */usr/share/doc/p3nfs-[wersja]*.

W systemie plików należy przygotować punkt zamontowania udziału NFS. Domyślnie `p3nfs` używa katalogu */mnt/psion*. Program `p3nfs` wymaga dodatkowo swojego własnego urządzenia RFCOMM, które trzeba skonfigurować. W przypadku telefonów Nokia najprawdopodobniej trzeba dowiązać je do kanału 4., a w przypadku innych telefonów — do kanału 3.

```
# mkdir /mnt/phone
# rfcomm bind /dev/rfcomm2 00:11:22:33:44:55 4
```

Teraz w telefonie należy uruchomić program `nfsapp`. Nie trzeba się martwić, gdy początkowo wybierze się zły kanał — program `nfsapp` informuje, na którym kanale nasłuchuje. Jeżeli *rfcomm* został związany ze złym kanałem, wystarczy użyć polecenia `rfcomm unbind /dev/rfcomm2`, a następnie spróbować poprawnego kanału. Domyślnie program *nfsapp* nasłuchuje na porcie podczerwieni. Klikając dżojstik lub naciskając przycisk `p`, można przełączać się pomiędzy portami IR, Bluetooth oraz TCP. Po wybraniu Bluetooth program *nfsapp* czeka 30 sekund na uruchomienie w komputerze programu `p3nfs`. Zakładając, że wszystko inne zostało skonfigurowane poprawnie, uruchomienie programu `p3nfs` powinno być możliwe w następujący sposób:

```
# p3nfsd -series60 -tty /dev/rfcomm2 -dir /mnt/phone -user incub
p3nfsd: version 5.13a, using /dev/rfcomm2 (115200), mounting on /mnt/phone
p3nfsd: to stop the server do "ls /mnt/phone/exit". (pid 3274)
```

Montowanie udziału zajmie programowi `p3nfs` kilka sekund. Szybki rzut oka na program `lsmod` potwierdzi fakt, że rzeczywiście używa on obsługi NFS jądra. Gdy używany telefon, który nie należy do rodziny Series 60, należy opcję `-series60` zastąpić opcją `-UIQ` lub inną. Listę dostępnych opcji można uzyskać, wydając polecenie `p3nfsd` bez żadnych opcji. Jeżeli trzeba, można też zmienić opcje `-tty`, `-dir` oraz `-user`. Opcja `-user` nie jest bezwzględnie potrzebna, ale polecenie `p3nfsd` montuje udział z uprawnieniami do odczytu i do wykonania pliku nadanymi tylko podanemu użytkownikowi. Zatem jeżeli opcja ta nie jest używana, to korzystanie z udziału jest możliwe tylko dla użytkownika root. Teraz można — za pomocą polecenia `cd` — przejść do katalogu */mnt/phone* i wykonać wszystkie operacje, które da się wykonać w systemie plików Unix, w katalogach zamontowanych przez NFS. Żeby odmontować udział, należy wydać polecenie `ls /mnt/phone/exit`, co jest trochę dziwnym rozwiązaniem. Odmontowanie udziału następuje też po kilku minutach braku aktywności. Czas ten można określić za pomocą opcji `-timeout`.

Przedstawiono tu kilka sposobów wymiany danych pomiędzy urządzeniami Bluetooth. Oczywiście techniki te można zastosować także do wymiany plików pomiędzy dwoma komputerami wyposażonymi w urządzenia Bluetooth i systemy Linux. Co więcej, te same metody działają tak samo dobrze podczas wymiany plików przez porty wykorzystujące podczerwień. W takim przypadku konieczne jest dokonanie niewielkich zmian, które jednak wykraczają poza zakres tego podrozdziału. I jeszcze jedna uwaga na koniec — żeby nowe, skonfigurowane urządzenia RFCOMM działały też po ponownym uruchomieniu systemu, nie należy zapomnieć o umieszczeniu odnoszących się do nich zapisów w pliku */etc/bluetooth/rfcomm.conf*. Teraz można już zacząć instalować programy!

*— Schuyler Erle*

SPOSÓB 19.

# Sterowanie programem XMMS poprzez Bluetooth

Za pomocą urządzenia Bluetooth można zdalnie sterować muzyką odtwarzaną w systemie Linux.

Za pomocą przenośnego urządzenia Bluetooth można sterować programem XMMS działającym w Linuksie. Istnieje kilka programów, które używają interfejsów typu WAP w telefonach serii T (takich jak T68i i T39m) firmy Ericsson i wykorzystują je jako urządzenia do zdalnego sterowania programem XMMS.

Jednym z tych programów jest samodzielny, napisany w języku Ruby program o nazwie `bluexmms`, który można pobrać ze strony *http://linuxbrit.co.uk/bluexmms/*. Telefon należy sparować (**[Sposób 16.]**) z interfejsem Bluetooth komputera. Po zainstalowaniu programu *bluexmms* należy — za pomocą programu `rfcomm` — powiązać urządzenie RFCOMM, w przypadku telefonu T68i, z kanałem 2., który w telefonach T68 pełni funkcję „ogólnej usługi telefonicznej" (dziwne!).

Następnie należy wydać polecenie `bluexmms /dev/rfcomm1`, przekazując w nim odpowiednią nazwę utworzonego urządzenia RFCOMM. Od tej chwili powinno być możliwe wybranie z menu telefonu polecenia *Accessories/XMMS Remote*!

Drugie, ale bardzo podobne rozwiązanie wykorzystuje moduł rozszerzający XMMS o nazwie `btexmms`, który można pobrać ze strony *http://www.lyola.com/bte/*. Po zbudowaniu i zainstalowaniu modułu należy — w sposób opisany wcześniej — utworzyć urządzenie RFCOMM wykorzystujące kanał 2. Następnie należy przejść do menu ustawień XMMS i po wybraniu *Effects → General Plugins* włączyć i skonfigurować moduł rozszerzający BTE Control. Następnie należy skonfigurować użycie utworzonego urządzenia RFCOMM i zapisać zmiany. Od tej pory funkcję zdalnego sterowania można uruchamiać z menu telefonu *Accessories/XMMS Remote*.

Jeżeli nie dysponujemy telefonem Ericsson z serii T, możemy użyć programu Bemused, który działa na urządzeniach z systemem SymbianOS, w który wyposażone są takie telefony, jak Nokia 3650/7650 czy Ericsson P800. W przeciwieństwie do właśnie omówionego programu dla T68, w przypadku którego to komputer ustanawia połączenie z telefonem, program Bemused wykorzystuje klienta inicjującego z telefonu połączenie z serwerem działającym w komputerze.

Serwer i klienta programu Bemused można pobrać ze strony *http://www.compsoc.man.ac.uk/~ashley/bemused/*. Po rozpakowaniu pliku *bemused.zip* należy przekopiować do telefonu i zainstalować w nim odpowiedni plik *.sis*. Następnie należy pobrać plik *bemusedlinuxserver.tar.gz* — pliki serwera należy zbudować i zainstalować w komputerze. Następnie za pomocą polecenia `sdptool add --channel=10 SP` należy ogłosić usługi szeregowe portu Bluetooth komputera oraz otworzyć do edycji i odpowiednio skonfigurować plik */etc/bemused.conf*. Plik README programu Bemused sugeruje użycie kanału 10. komputera, ale równie dobrze można użyć dowolnego nieużywanego kanału. Teraz należy włączyć X11 (jeśli wcześniej nie były uruchomione). Następnie w wierszu poleceń należy uruchomić program `bemusedserverlinux`. W tym momencie powinno być możliwe uruchomienie w telefonie programu Bemused — możemy już cieszyć się potęgą XMMS, przemieszczając się po pomieszczeniu.

Nie trzeba się martwić, jeśli nie ma się żadnego z tych telefonów — praktycznie wszystkie współczesne urządzenia Bluetooth mają zaimplementowaną jakąś warstwę komunikacji szeregowej. Posługując się przykładami opisanych tu projektów, można do sterowania programem XMMS użyć innego telefonu lub urządzenia PDA. Możliwości sterowania tego rodzaju aplikacjami przez Bluetooth jest dość wiele.

Jeżeli ktoś dotarł do tego miejsca, to prawdopodobnie wpadł na pomysł, że dysponując zdalnie sterowanym programem XMMS, można by podłączyć do wzmacniacza stereo wyspecjalizowany linuksowy serwer MP3 i nie musieć nigdy podłączać do niego ani monitora, ani klawiatury. Można by też podłączyć do karty dźwiękowej nadajnik FM o niewielkiej mocy i słuchać w domu muzyki z dowolnego radia. To prawda — wszystkie te pomysły i na pewno także wiele innych można łatwo zrealizować za pomocą Bluetooth.

*— Schuyler Erle*