Badania podatności usługi DNS na wybrane zagrożenia

Zbigniew SUSKI

Instytut Teleinformatyki i Automatyki WAT
ul. Gen. S. Kaliskiego 2, 00-908 Warszawa
zsuski@wat.edu.pl

STRESZCZENIE: W artykule zostały przedstawione wyniki eksperymentów, których celem było sprawdzenie podatności najbardziej popularnych serwerów DNS na wybrane zagrożenia.

SŁOWA KLUCZOWE: testy penetracyjne, zagrożenia bezpieczeństwa, DNS

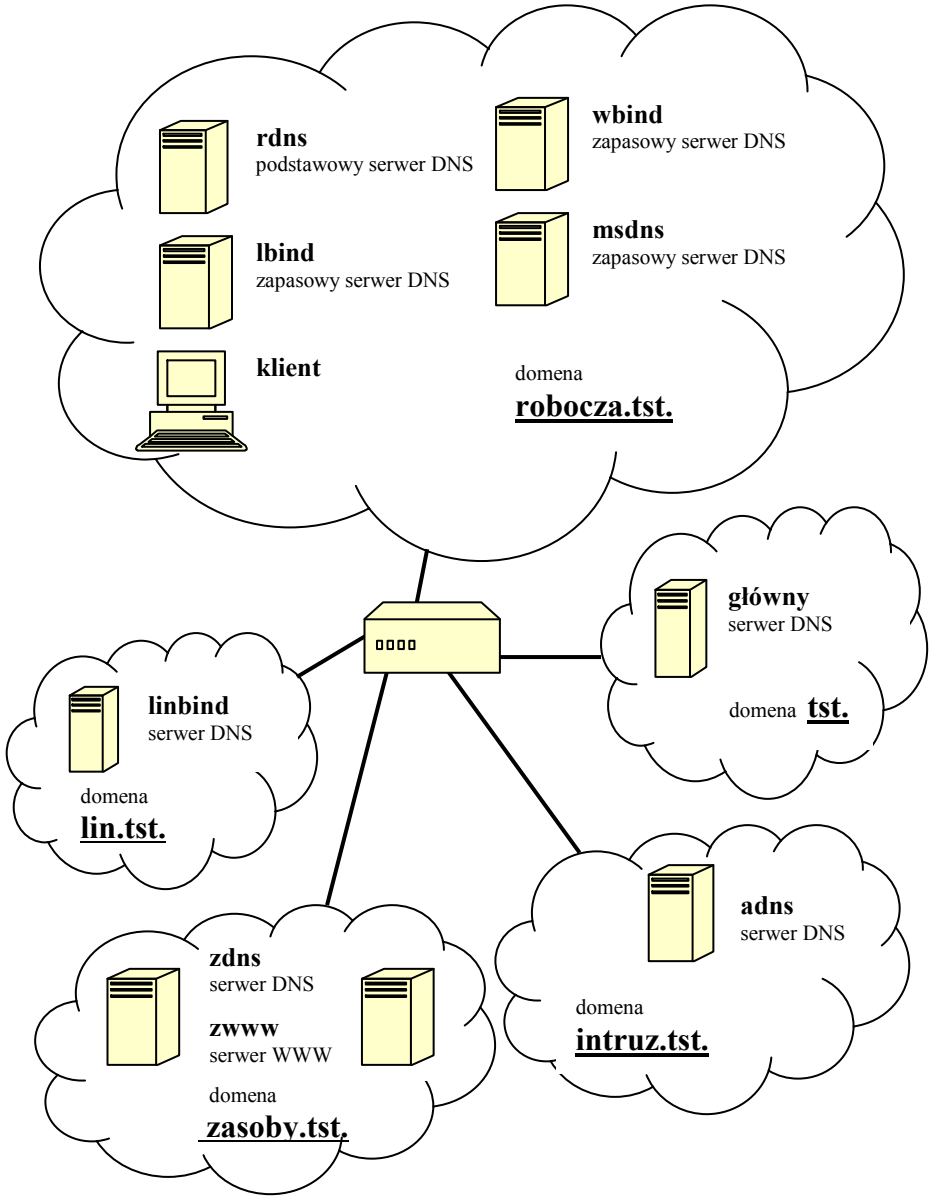
1. Wstęp

Celem przeprowadzonych i opisanych w niniejszym artykule badań, była eksperymentalna weryfikacja podatności wybranych serwerów DNS na zagrożenia opisane w [8]. W trakcie badań wykorzystywano elementy sieci przedstawione na rys. 1. Ich szczegółowa specyfikacja została przedstawiona w tab. 1. Są to głównie serwery DNS posadowione na różnych platformach systemowych obsługujące domeny wymienione na rys. 1 i w tab. 1

2. Transfer strefy

Transfer strefy związany jest z funkcją serwerów DNS polegającą na przesyłaniu rekordów bazy danych o strefie przez serwer główny do serwerów podrzędnych. Zagrożenia związane z tym mechanizmem przedstawiono w [8]. W przypadku niestarannie skonfigurowanego serwera DNS informacje znajdujące się w pliku strefy można uzyskać z dowolnego komputera, za pomocą programu *nslookup* lub *dig*. Na rys. 2 przedstawiono przykład pozyskiwania takiej informacji za pomocą programu *nslookup*. Na rys. 3

przedstawiono podobny przypadek ale z użyciem programu *dig*. W obu przypadkach programy uruchomiono w sieci testowej na komputerze atak.intruz.tst. Dane dotyczące domeny robocza.tst uzyskano z serwerów rdns.robocza.tst i lbind.robocza.tst. Można więc zauważyć, że podatność na przedstawione działanie wykazuje zarówno serwer *msdns* jak i *bind*.



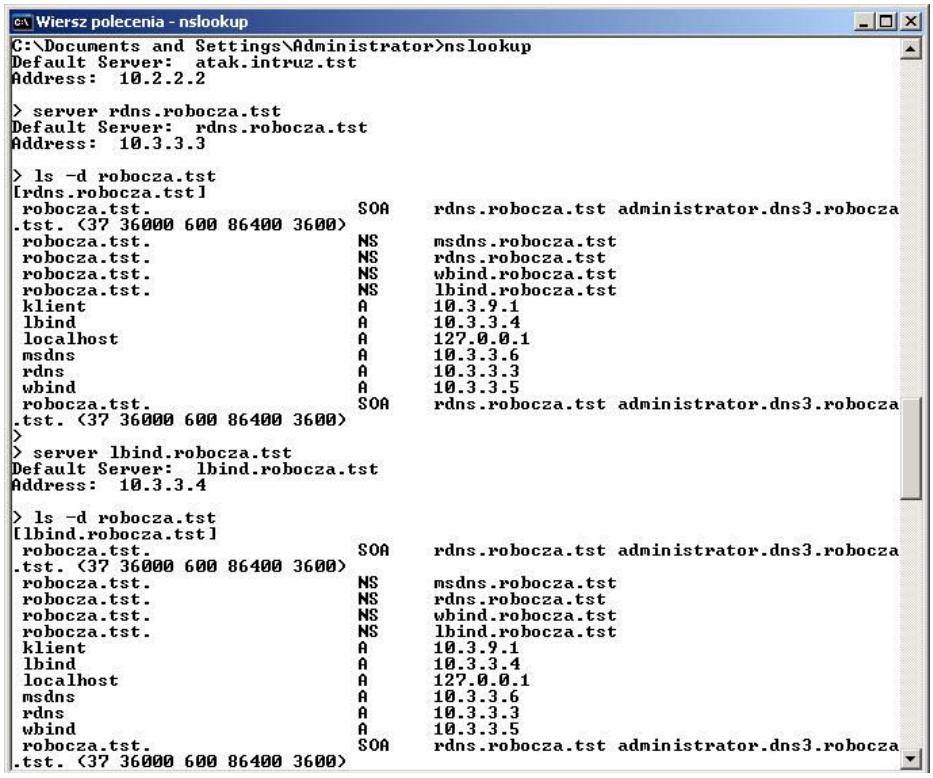
Rys. 1. Struktura sieci służącej do przeprowadzenia badań

Tab. 1. Charakterystyka systemów wykorzystywanych podczas eksperymentów

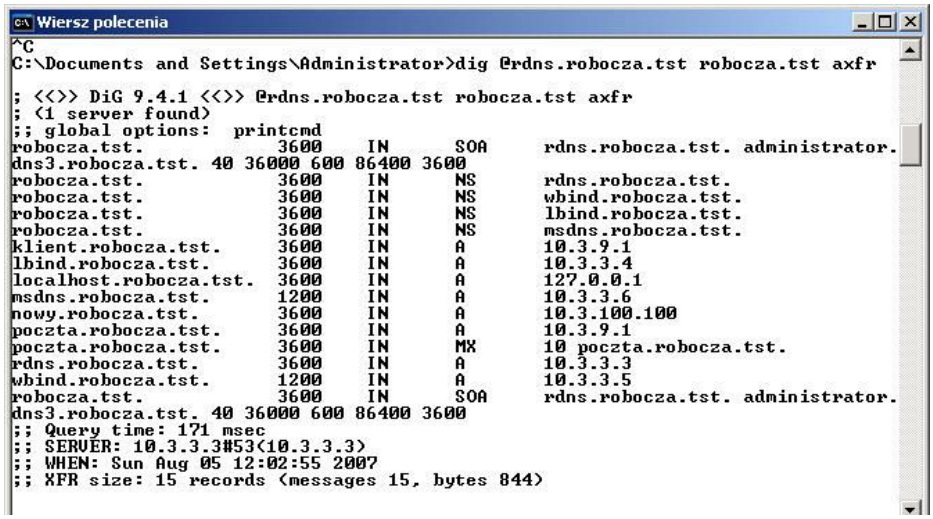
Maszyna wirtualna	FQDN domena odwr.	System operacyjny	Adres IP	Serwer DNS	Inne
1	glowny.tst 1.10.in-addr.arpa.	Windows 2003	10.1.1.1	MS DNS	
A_srv	atak.intruz.tst	Windows 2003	10.2.2.2	MS DNS	WWW
A_lin	lin.intruz.tst	Debian 4.0	10.2.2.33	brak	
R_dns	rdns.robocza.tst 3.10.in-addr.arpa.	Windows 2003	10.3.3.3	MS DNS	podstawowy
R_lbind	lbind.robocza.tst	Debian 4.0	10.3.3.4	BIND 9.4.1	zapasowy
R_wbind	wbind.robocza.tst	Windows 2003	10.3.3.5	BIND 9.4.1	zapasowy
R_msdns	msdns.robocza.tst 3.10.in-addr.arpa.	Windows 2003	10.3.3.6	MS DNS	zapasowy
R_kl	klient.robocza.tst	Windows 2003	10.3.9.1		
Z_www	zwww.zasoby.tst	Windows 2003	10.7.7.77		
Z_dns	zdns.zasoby.tst	Windows 2003	10.7.7.7	MS DNS	podstawowy
Lin	linbind.lin.tst	Windows 2003	10.4.4.4	BIND 9.4.1	podstawowy

Należy jednak zauważyć, że wyniki przedstawione na rys. 2 i 3 można uzyskać tylko w przypadku niestarannie skonfigurowanego serwera.

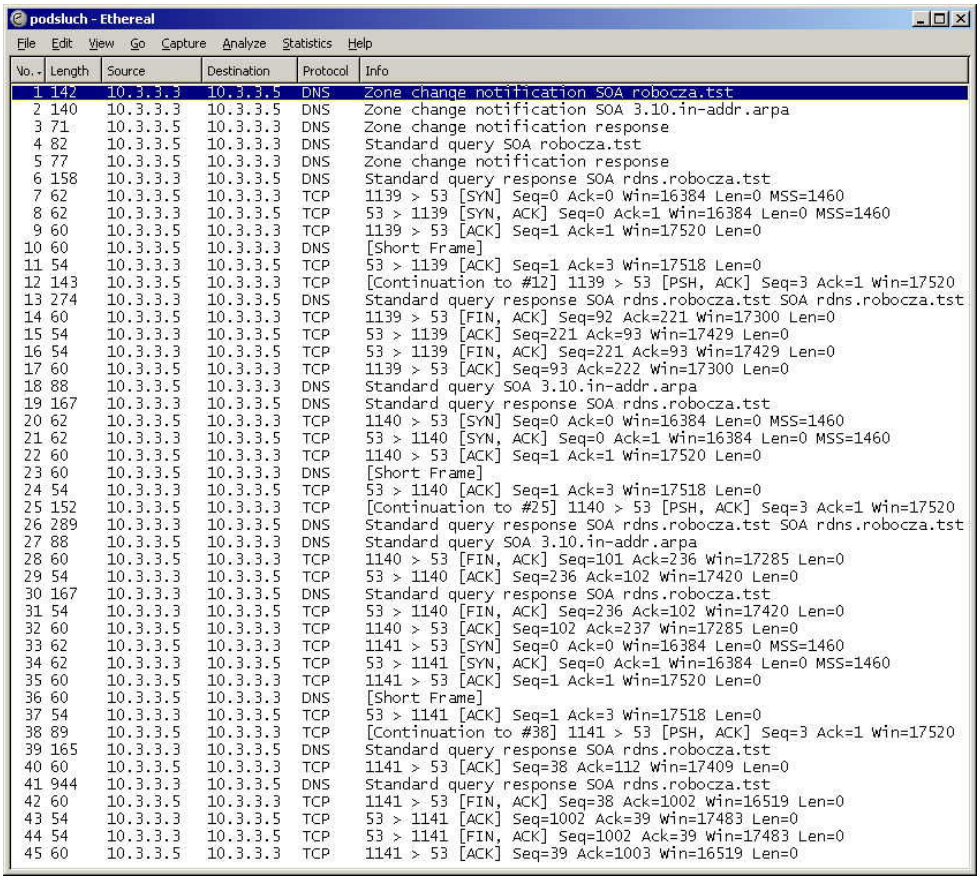
Następny eksperyment da efekt zadawalający dla intruza nawet w przypadku poprawnie skonfigurowanego serwera DNS. Jedynym warunkiem, który musi być spełniony jest możliwość prowadzenia podsłuchania ruchu sieciowego. Na rys. 4÷7 przedstawiono obraz ruchu sieciowego uzyskany za pomocą snifera zainstalowanego na komputerze intruza w sieci testowej. Zaobserwowany przepływ pakietów spowodowany został modyfikacją bazy serwera rdns.robocza.tst. Modyfikacja ta polegała na wprowadzeniu rekordu A i PTR dla nowego komputera w domenie robocza.tst. Komputer o adresie 10.3.100.100 został zarejestrowany pod nazwą nowy.robocza.tst.



Rys. 2. Wyniki pobrania informacji o strefie za pomocą programu *nslookup*



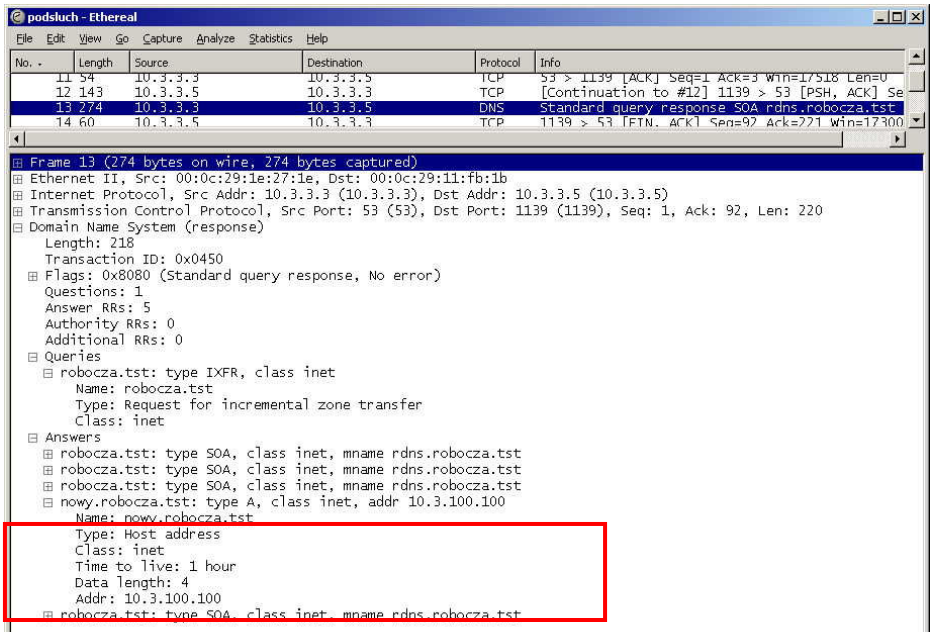
Rys. 3. Wyniki pobrania informacji o strefie za pomocą programu *dig*



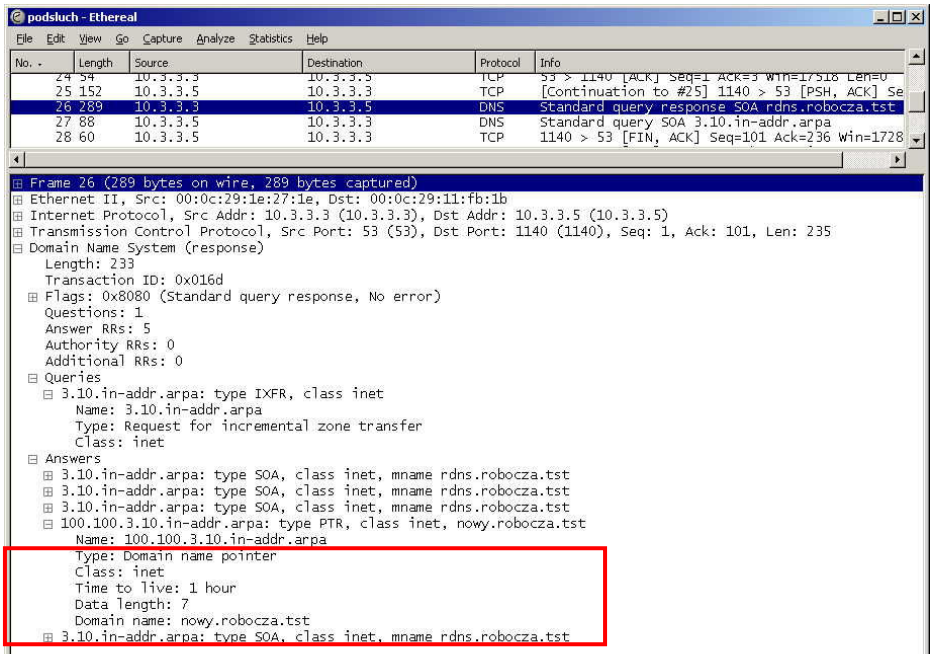
Rys. 4. Wymiana informacji pomiędzy serwerami DNS spowodowana zarejestrowaniem nowego komputera na serwerze podstawowym

Na rys. 4 pakiety 1÷5 dotyczą powiadamiania serwera pomocniczego o fakcie wprowadzenia zmian. Pozostałe pakiety są związane z przekazywaniem szczegółów wprowadzonej zmiany. Na rys. 5 możemy obejrzeć zawartość pakietu 13. Za jego pomocą serwer główny przekazuje zawartość nowego rekordu typu A. Na rys. 6 przedstawiono zawartość pakietu 26. Za jego pomocą przekazywana jest treść nowego rekordu PTR.

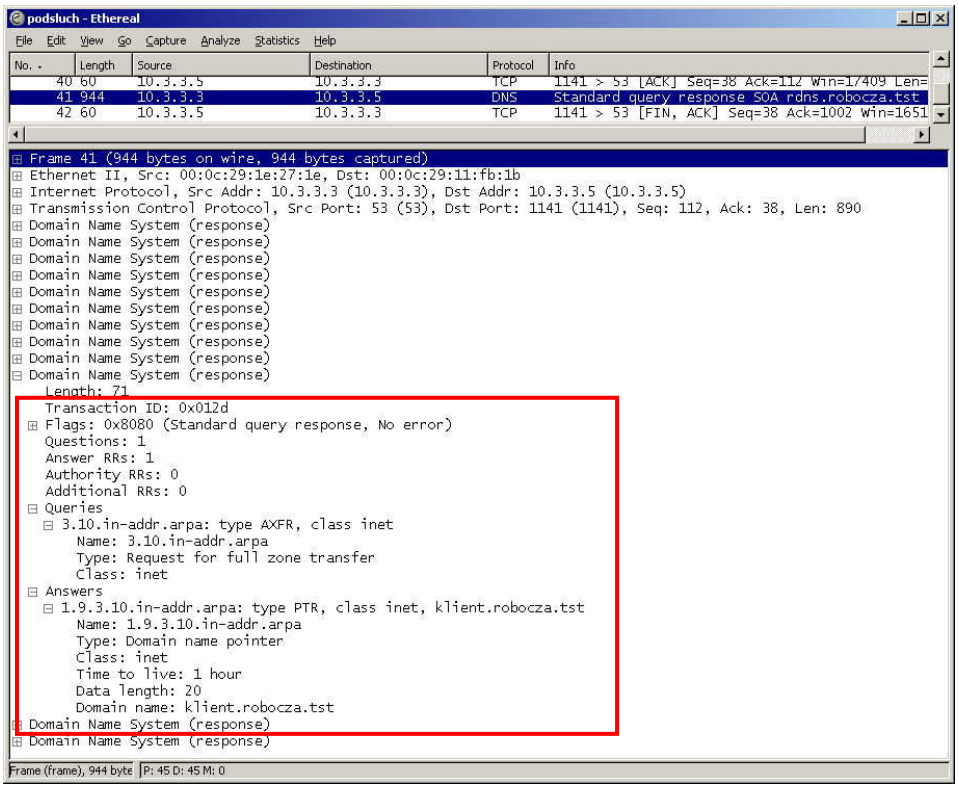
Na rys. 7 przedstawiono fragment zawartości rekordu 41. Jest to odpowiedź serwera głównego na żądanie całościowego transferu strefy. Na rysunku ramką zaznaczono rozwiniętą treść jednego z rekordów. Jest to rekord PTR komputera o nazwie klient.robocza.tst.



Rys. 5. Przekazanie treści nowo utworzonego rekordu typu A



Rys. 6. Przekazanie treści nowo utworzonego rekordu typu PTR

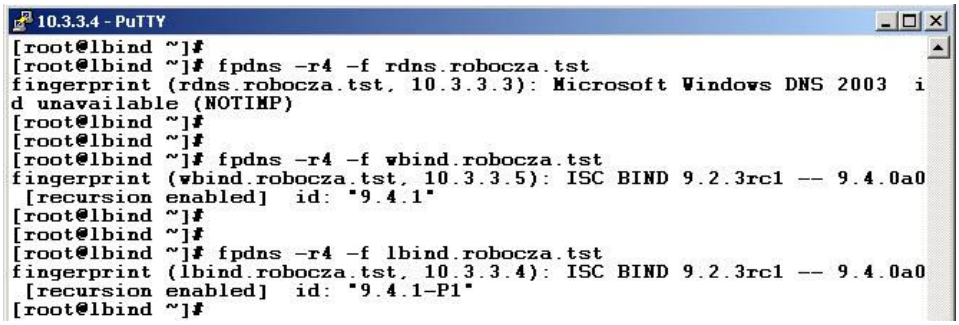


Rys. 7. Fragment odpowiedzi serwera głównego na żądanie całościowego transferu strefy

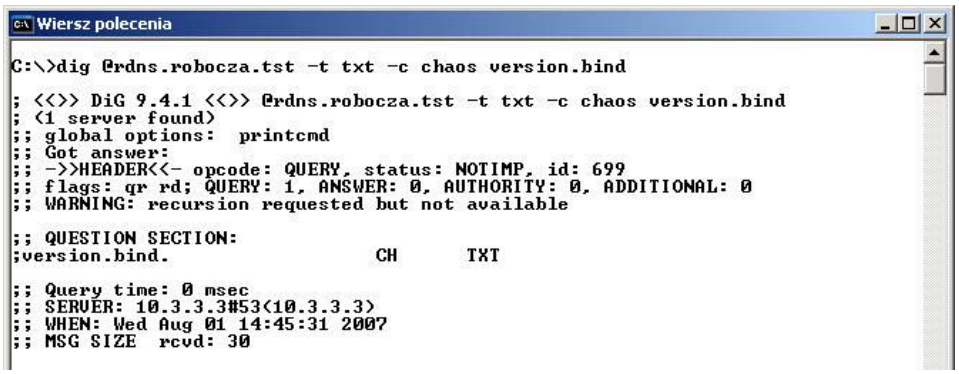
3. Sprawdzanie wersji

Do sprawdzania numeru wersji wykorzystać można dostępny w Internecie program ***fpdns***. Jak można zauważyć na rys. 8, w czasie przeprowadzanych eksperymentów prawidłowo rozpoznawał on zainstalowane oprogramowanie serwerów DNS. Do określania numeru wersji wykorzystać można program ***dig***. Jednak w przypadku serwera MS DNS nie uzyskuje się zadawalających rezultatów, co można zauważyć na rys. 9. Program ten prawidłowo określa natomiast wersję serwera ***bind*** i to bez względu na system operacyjny, na którym został on osadzony.

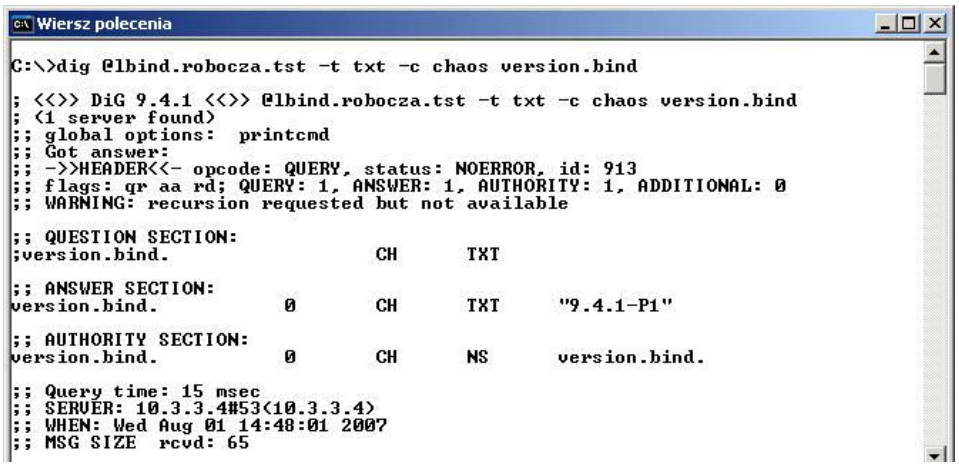
Wyniki dla programu ***bind*** zainstalowanego w systemie Linuks (Debian) przedstawia rys. 10. Na rys. 11 zaprezentowano raport uzyskany w przypadku tego serwera DNS zainstalowanego w systemie MS Windows 2003.



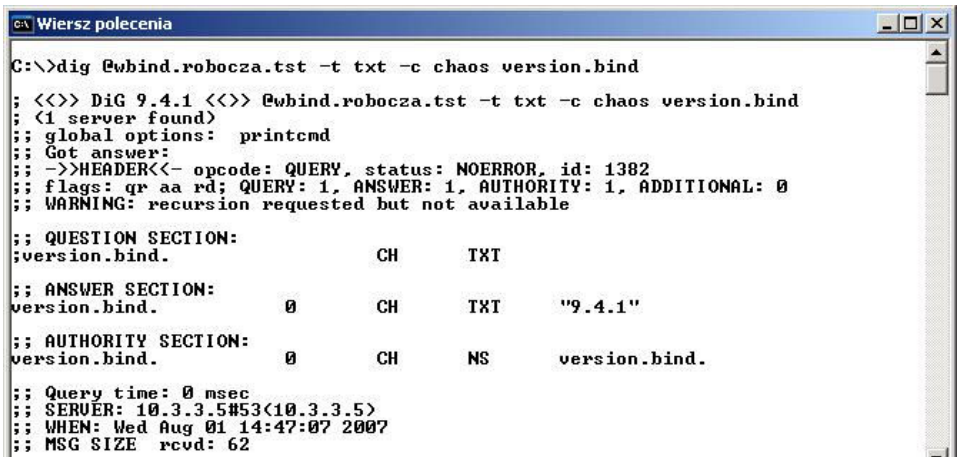
Rys. 8. Wyniki rozpoznania wersji uzyskane za pomocą programu *fpdns*



Rys. 9. Wyniki rozpoznania wersji uzyskane za pomocą programu *dig* w stosunku do serwera MS DNS



Rys. 10. Wyniki rozpoznania wersji uzyskane za pomocą programu *dig* w stosunku do serwera *bind* zainstalowanego w systemie Linuks



Rys. 11. Wyniki rozpoznania wersji uzyskane za pomocą programu *dig* w stosunku do serwera *bind* zainstalowanego w systemie MS Windows 2003

4. Ataki typu odmowa usługi

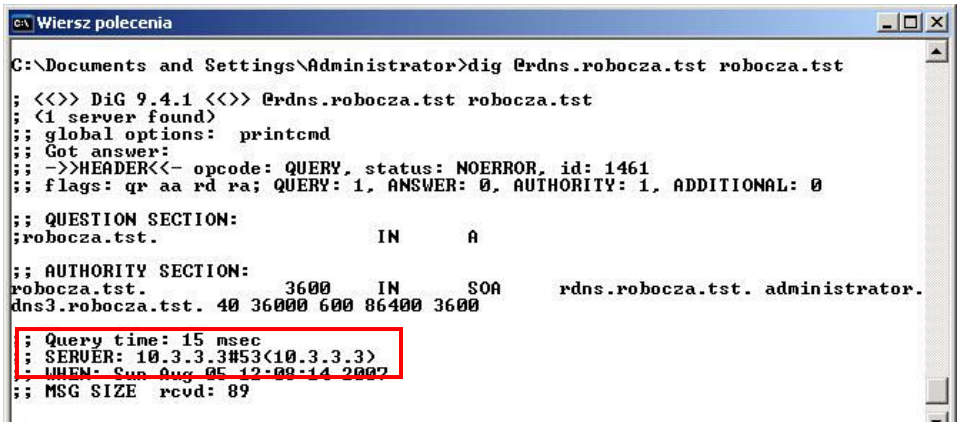
4.1. Atak SYNflood

W niniejszym punkcie przedstawiono wyniki eksperymentu polegającego na przeprowadzeniu ataku *SYNflood* na serwer DNS. Dla potrzeb tego eksperymentu opracowano program (*eksploit*) o nazwie SYNFLOOD.

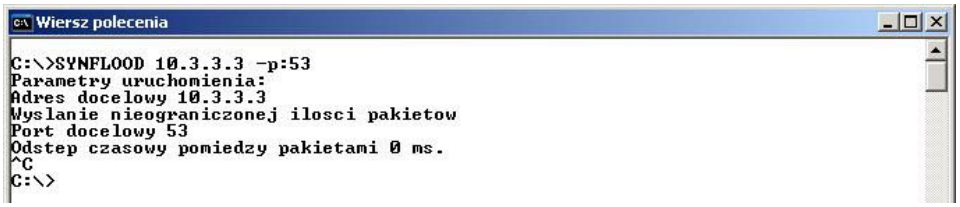
Na rys. 12 przedstawiono wyniki badania czasu odpowiedzi serwera rdns.robocza.tst w normalnych warunkach eksploatacji sieci testowej. Interesujący fragment raportu zaznaczono ramką. Można zauważyć, że czas odpowiedzi wynosi 15 ms.

Następnie zainicjowano atak *SYNflood*. Na rys. 13 przedstawiono raport z uruchomienia programu SYNFLOOD.

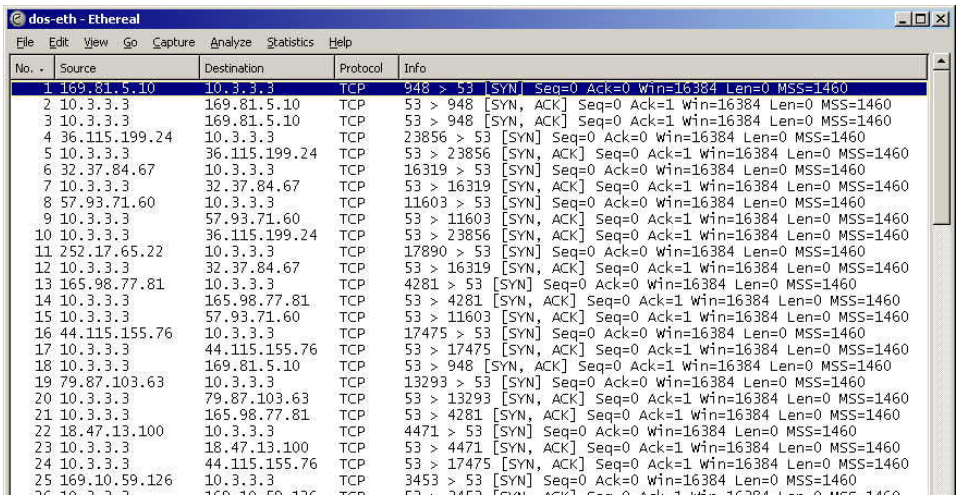
Rys. 14 prezentuje obraz ruchu sieciowego podczas ataku. Pakiety SYN kierowane są do atakowanego komputera o adresie 10.3.3.3 na port 53 TCP (DNS). Akceptuje on kolejne prośby o połączenie odpowiadając pakietami SYN/ACK. Ponieważ adresy źródłowe w pakietach SYN zostały sfałszowane, serwer DNS nie uzyskuje potwierdzenia odbioru dla swoich pakietów SYN, ACK. Można zaobserwować, że realizuje on wobec tego powtórzenie transmisji. W innych systemach operacyjnych, np. w Linuksie, tych prób powtórzeń byłoby więcej.



Rys. 12. Wyniki badania czasu odpowiedzi serwera rdns.robocza.tst w normalnych warunkach eksploatacji sieci testowej

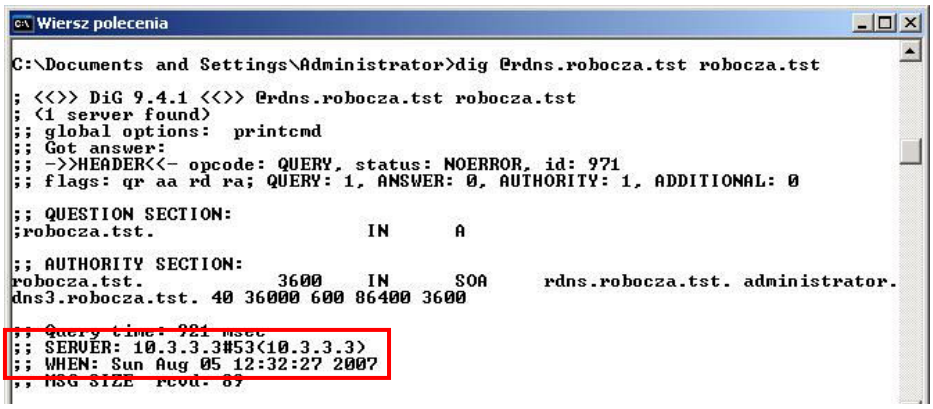


Rys. 13. Raport z uruchomienia programu SYNFLOOD

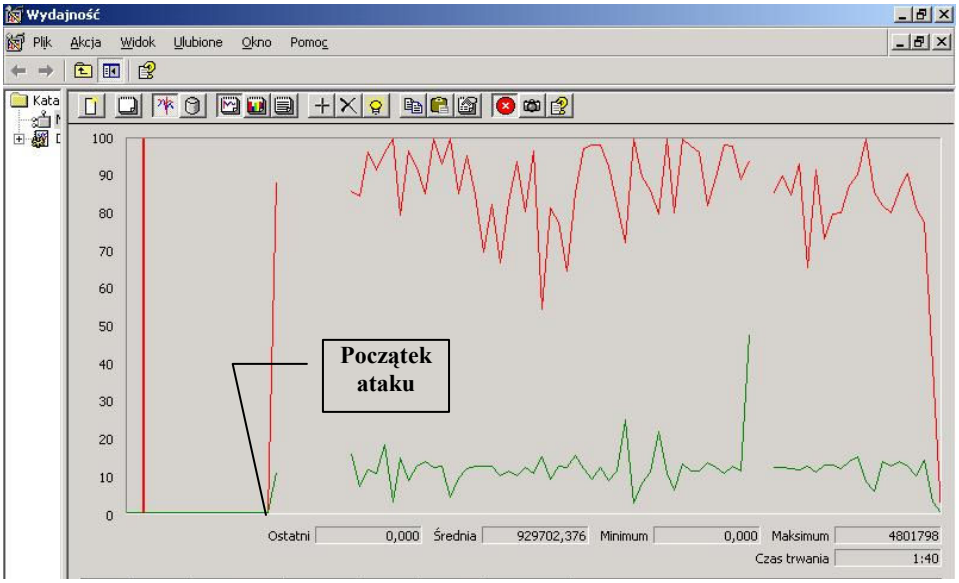


Rys. 14. Obraz ruchu sieciowego podczas ataku *SYNflood* na serwer DNS

Jakie były skutki ataku po stronie serwera DNS? Daje się zauważyć znaczne wydłużenie czasu odpowiedzi udzielanej przez serwer podczas ataku. W obserwowanym przypadku wyniósł on 921 ms – patrz ramka na rys. 15. Jednocześnie mierzono obciążenie procesora i karty sieciowej atakowanego komputera. Na rys. 16 można zauważyć, że przed atakiem obciążenie tych zasobów było znikome, wręcz niezauważalne. Podczas ataku obciążenie procesora oscylowało w przedziale 70÷100 %. Obciążenie to przedstawia górna krzywa. Znacznie wzrosło też obciążenie karty sieciowej – krzywa dolna.



Rys. 15. Wyniki badania czasu odpowiedzi serwera rdns.robocza.tst podczas ataku *SYNflood*



Rys. 16. Obciążenie procesora i karty sieciowej przed i podczas ataku *SYNflood*

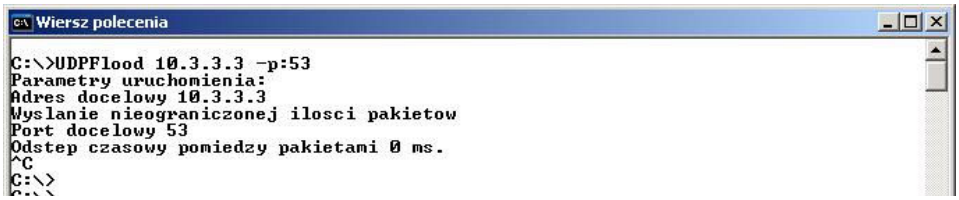
Jeszcze większe obciążenie komputera atakowanego można byłoby uzyskać przeprowadzając atak typu DDOS [8]. W testowanym przypadku do praktycznie całkowitego zablokowania serwera DNS wystarczyłoby przeprowadzenie ataku z 2÷3 komputerów.

4.2. Atak UDPflood

W niniejszym punkcie przedstawiono wyniki eksperymentu polegającego na przeprowadzeniu ataku *UDPflood* na serwer DNS. Dla potrzeb tego eksperymentu opracowano program (*eksploit*) o nazwie UDPFLOOD.

Przed rozpoczęciem ataku zbadano czas odpowiedzi serwera rdns.robocza.tst w normalnych warunkach eksploatacji sieci testowej. Wyniki uzyskano identyczne jak przed atakiem *SYNFlood* (rys. 12).

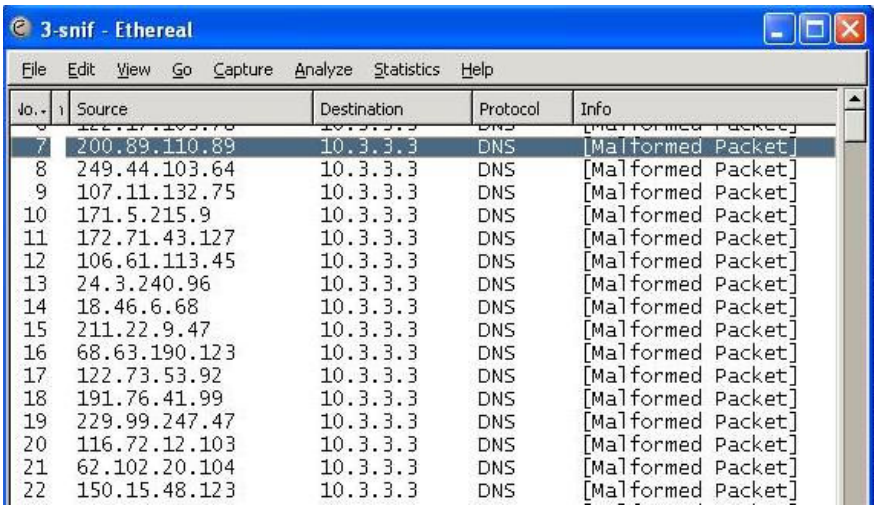
Następnie zainicjowano atak *UDPflood*. Na rys. 17 przedstawiono raport z uruchomienia programu UDPFLOOD.



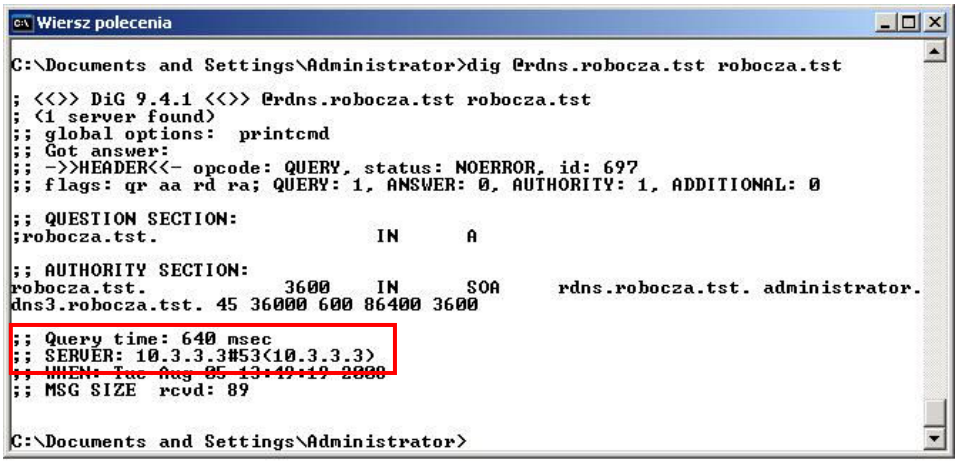
Rys. 17. Raport z uruchomienia programu UDPFLOOD

Rys. 18 prezentuje obraz ruchu sieciowego podczas ataku. Pakiety kierowane są do atakowanego komputera o adresie 10.3.3.3 na port 53 UDP(DNS). Ponieważ pakiety zawierają jedynie nagłówki IP i UDP – pole danych nie występuje, więc są one rozpoznawane jako pakiety uszkodzone (*malformed*). Podobnie jak podczas ataku *SYNflood* adresy źródłowe w zostały sfałszowane.

Po stronie serwera DNS daje się zauważyć wydłużenie czasu odpowiedzi udzielanej przez serwer podczas ataku. W obserwowanym przypadku wyniósł on 640 ms – patrz ramka na rys. 19. Jednocześnie mierzono obciążenie procesora i karty sieciowej atakowanego komputera. Na rys. 20 można zauważyć, że przed atakiem obciążenie tych zasobów było znikome, wręcz niezauważalne. Podczas ataku obciążenie procesora oscylowało w przedziale 50÷100 %. Obciążenie to przedstawia górna krzywa. Znacznie wzrosło też obciążenie karty sieciowej – krzywa dolna.



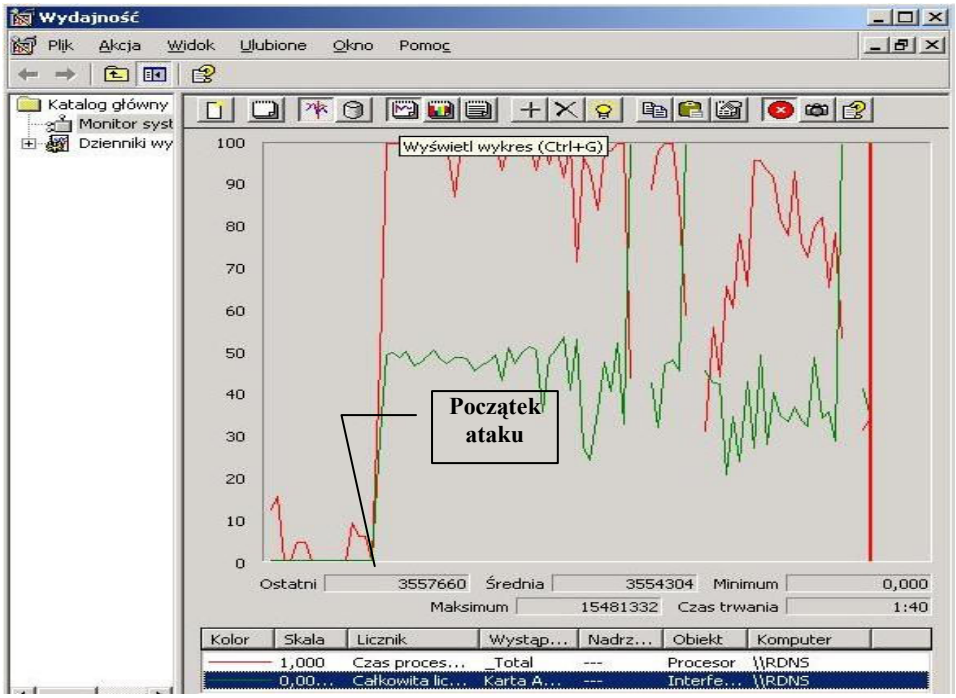
Rys. 18. Obraz ruchu sieciowego podczas ataku *UDPflood* na serwer DNS



Rys. 19. Wyniki badania czasu odpowiedzi serwera rdns.robocza.tst podczas ataku *UDPflood*

4.3. Atak DNSflood

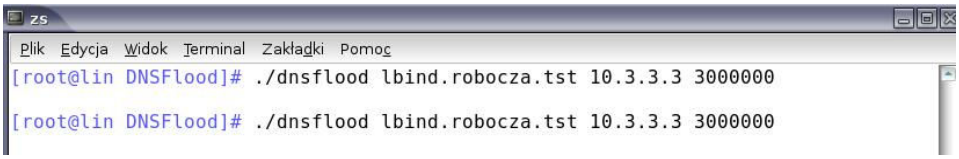
Atak tego typu polega na wysyłaniu dużej liczby poprawnie sformatowanych zapytań DNS. W niniejszym punkcie przedstawiono wyniki eksperymentu polegającego na przeprowadzeniu ataku *DNSflood* na serwer DNS. Dla potrzeb tego eksperymentu opracowano program (*eksploit*) o nazwie DNSFLOOD.



Rys. 20. Obciążenie procesora i karty sieciowej przed i podczas ataku *UDPflood*

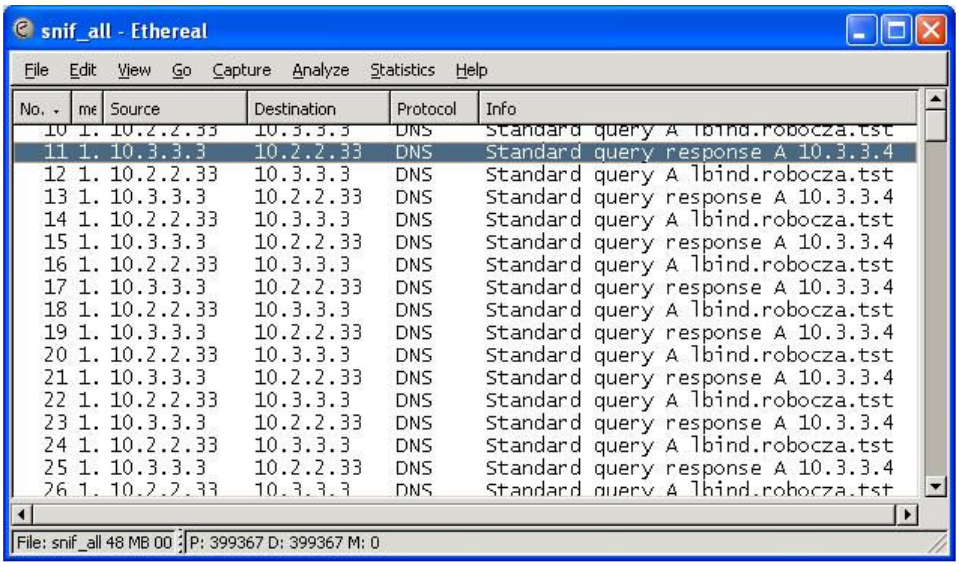
Podobnie jak w poprzednich eksperymentach, przed rozpoczęciem ataku zbadano czas odpowiedzi serwera rdns.robocza.tst w normalnych warunkach eksploatacji sieci testowej. Wyniki uzyskano identyczne jak przed atakiem *SYNFlood* (rys. 12).

Następnie zainicjowano atak *DNSflood*. Na rys. 21 przedstawiono raport z uruchomienia programu DNSFLOOD.



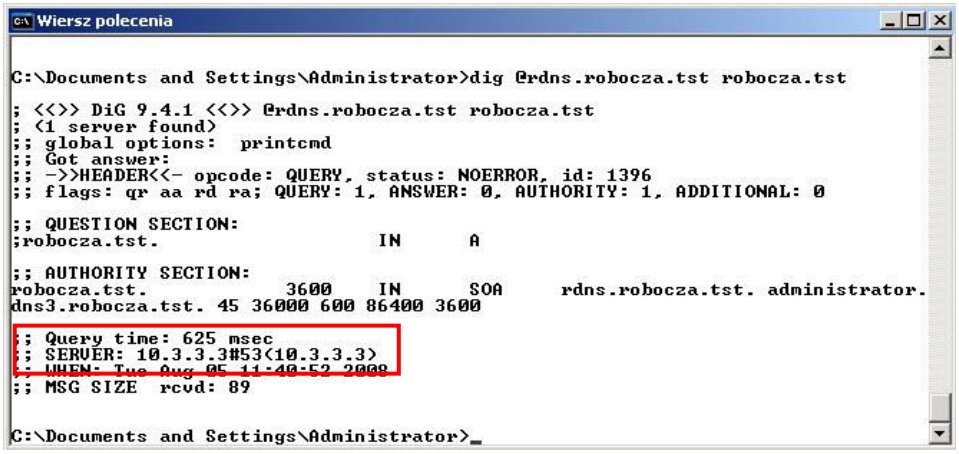
Rys. 21. Raport z uruchomienia programu DNSFLOOD

Rys. 22 prezentuje obraz ruchu sieciowego podczas ataku. Pakiety zawierające poprawne zapytania DNS kierowane są do atakowanego komputera o adresie 10.3.3.3 na port 53 TCP(DNS). Zwrotnie przekazywane są odpowiedzi serwera.

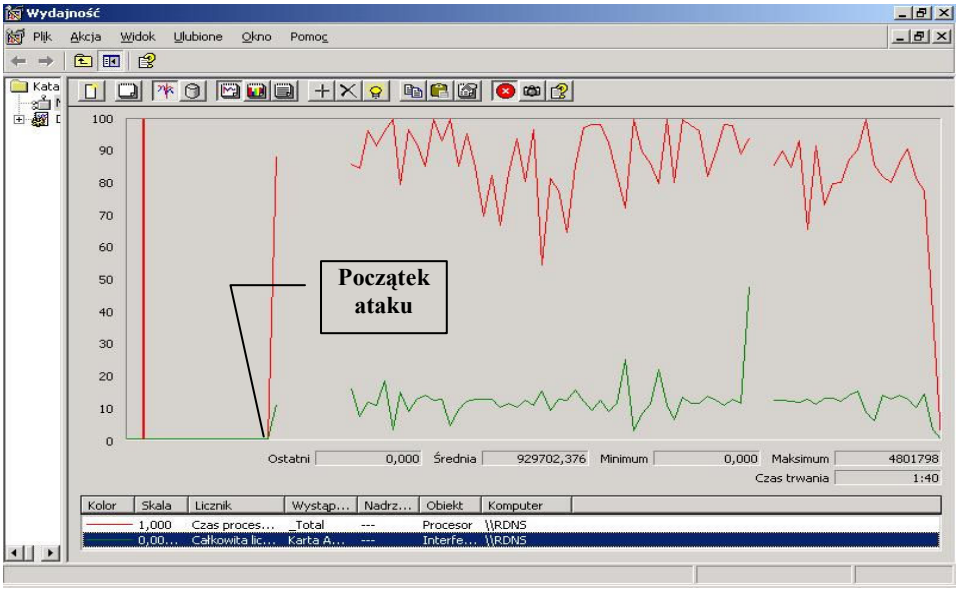


Rys. 22. Obraz ruchu sieciowego podczas ataku *DNSflood* na serwer DNS

Po stronie serwera DNS daje się zauważyć wydłużenie czasu odpowiedzi udzielanej przez serwer podczas ataku. W obserwowanym przypadku wyniósł on 625 ms – patrz ramka na rys. 23. Jednocześnie mierzono obciążenie procesora i karty sieciowej atakowanego komputera. Na rys. 24 można zauważyć, że przed atakiem obciążenie tych zasobów było znikome, wręcz niezauważalne. Podczas ataku obciążenie procesora oscylowało w przedziale 70÷100 %. Obciążenie to przedstawia górna krzywa. Znacznie wzrosło też obciążenie karty sieciowej – krzywa dolna.



Rys. 23. Wyniki badania czasu odpowiedzi serwera rdns.robocza.tst podczas ataku *DNSflood*



Rys. 24. Obciążenie procesora i karty sieciowej przed i podczas ataku *DNSflood*

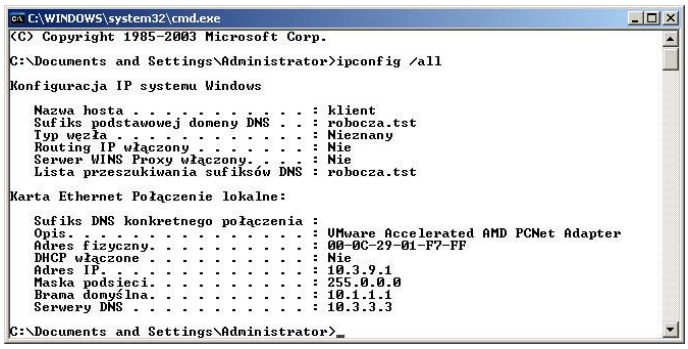
5. Zatruwanie pamięci podręcznej

Ofiarą ataku w trakcie eksperymentu polegającego na zatruwaniu pamięci podręcznej (*cache poisoning*) był serwer podstawowy (rdns) domeny robocza.tst zbudowany w oparciu o oprogramowanie MSDNS w systemie Windows 2003 Server.

Eksperyment rozpoczął się od sprawdzenia poprawności współdziałania komputera klient.robocza.tst z serwerem DNS rdns.robocza.tst. Sprawdzenie to polegało na weryfikacji konfiguracji interfejsu sieciowego klienta za pomocą programu *ipconfig* (rys. 25) oraz badaniu poprawności rozwiązywania nazwy i osiągalności komputera www.zasoby.tst (rys. 26). Na podstawie zamieszczonych obrazów można stwierdzić, że klient poprawnie współpracuje ze swoim serwerem DNS.

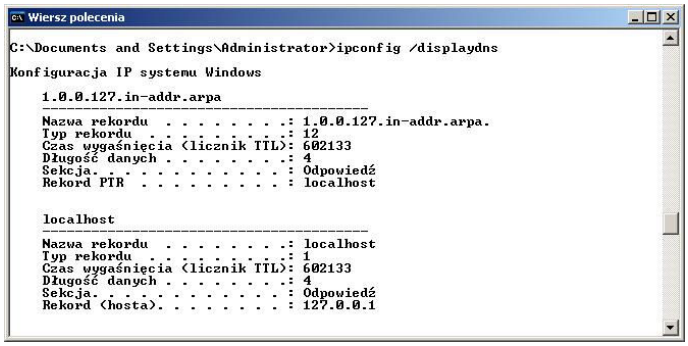
Po tych czynnościach wstępnych zrealizowano „czyszczenie" pamięci podręcznej *resolvera* systemu klient.robocza.tst. Poprawny wynik tej operacji dokumentuje rys. 27. W pamięci podręcznej *resolvera* nie ma żadnych „obcych" wpisów.

Ostatnim etapem weryfikacji wstępnej była próba pobrania strony www z serwera www.zasoby.tst. Wynik tej operacji przedstawiono na rys. 28. Można zauważyć, że wynik jest poprawny.



Rys. 25. Konfiguracja interfejsu sieciowego komputera klient.robocza.tst

Rys. 26. Wynik badania poprawności rozwiązywania nazwy i osiągalności komputera www.zasoby.tst

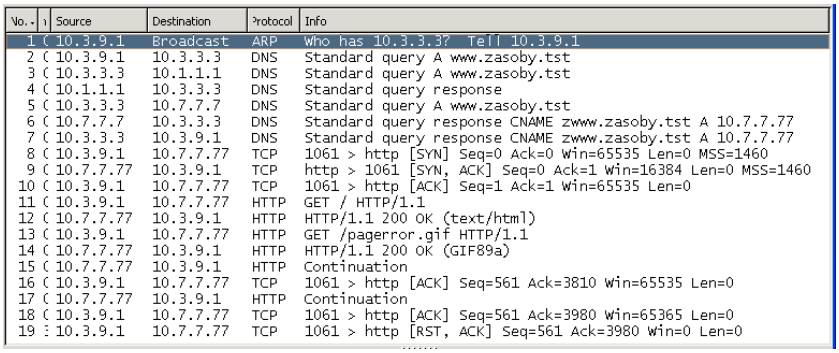


Rys. 27. Zawartość pamięci podręcznej *resolvera* systemu klient.robocza.tst po operacji czyszczenia tej pamięci

Na rys. 29 zamieszczono obraz ruchu sieciowego realizowanego podczas pobierania strony z serwisu www.zasoby.tst. Jego prześledzenie jest istotne ze względu na wychwycenie różnic, które będzie można zaobserwować po udanym ataku.

Rys. 28. Wynik próby pobrania przez klienta strony z serwera www.zasoby.tst

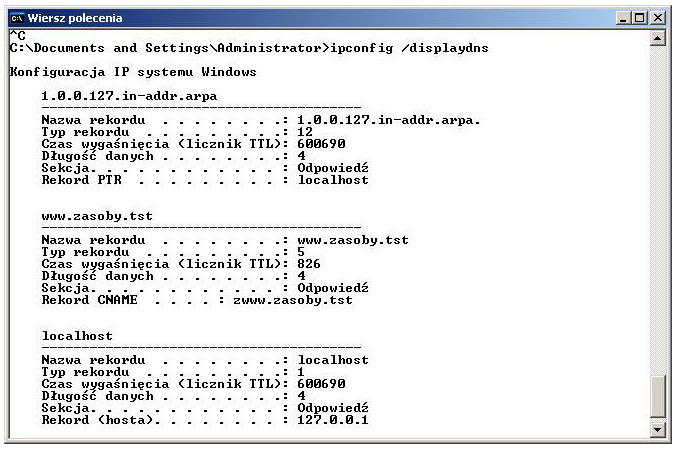
Na przedstawionym obrazie można zauważyć, że najpierw klient (10.3.9.1) zwraca się do swojego serwera DNS (10.3.3.3) z żądaniem rozwiązania nazwy www.zasoby.tst. Ponieważ nie jest to serwer autorytatywny domeny zasoby.tst, więc zwraca się on z takim żądaniem do serwera głównego (10.1.1.1). Otrzymuje od niego informację, że z takim żądaniem powinien się zwrócić do serwera 10.7.7.7, co też czyni. Uzyskuje informację, że system www.zasoby.tst ulokowany jest pod adresem 10.7.7.77 i taką informację przekazuje swojemu klientowi (10.3.9.1). Po otrzymaniu adresu systemu www.zasoby.tst, klient inicjuje z nim połączenie TCP na porcie 80 i realizowana jest sekwencja związana z przesłaniem żądania klienta i przesłaniem strony przez serwer www. Wreszcie następuje zamkniecie połączenia TCP.



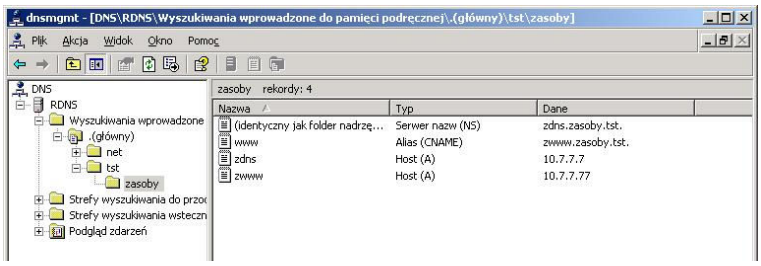
No.	Source	Destination	Protocol	Info
1	10.3.9.1	Broadcast	ARP	Who has 10.3.3.3? Tell 10.3.9.1
2	10.3.9.1	10.3.3.3	DNS	Standard query A www.zasoby.tst
3	10.3.3.3	10.1.1.1	DNS	Standard query A www.zasoby.tst
4	10.1.1.1	10.3.3.3	DNS	Standard query response
5	10.3.3.3	10.7.7.7	DNS	Standard query A www.zasoby.tst
6	10.7.7.7	10.3.3.3	DNS	Standard query response CNAME zwww.zasoby.tst A 10.7.7.77
7	10.3.3.3	10.3.9.1	DNS	Standard query response CNAME zwww.zasoby.tst A 10.7.7.77
8	10.3.9.1	10.7.7.77	TCP	1061 > http [SYN] Seq=0 Ack=0 Win=65535 Len=0 MSS=1460
9	10.7.7.77	10.3.9.1	TCP	http > 1061 [SYN, ACK] Seq=0 Ack=1 Win=16384 Len=0 MSS=1460
10	10.3.9.1	10.7.7.77	TCP	1061 > http [ACK] Seq=1 Ack=1 Win=65535 Len=0
11	10.3.9.1	10.7.7.77	HTTP	GET / HTTP/1.1
12	10.7.7.77	10.3.9.1	HTTP	HTTP/1.1 200 OK (text/html)
13	10.3.9.1	10.7.7.77	HTTP	GET /pagerror.gif HTTP/1.1
14	10.7.7.77	10.3.9.1	HTTP	HTTP/1.1 200 OK (GIF89a)
15	10.7.7.77	10.3.9.1	HTTP	Continuation
16	10.3.9.1	10.7.7.77	TCP	1061 > http [ACK] Seq=561 Ack=3810 Win=65535 Len=0
17	10.7.7.77	10.3.9.1	HTTP	Continuation
18	10.3.9.1	10.7.7.77	TCP	1061 > http [ACK] Seq=561 Ack=3980 Win=65365 Len=0
19	10.3.9.1	10.7.7.77	TCP	1061 > http [RST, ACK] Seq=561 Ack=3980 Win=0 Len=0

Rys. 29. Obraz ruchu sieciowego realizowanego podczas pobierania strony z serwera www.zasoby.tst

Dla porządku przedstawiono jeszcze zawartość pamięci podręcznej *resolvera* systemu klient.robocza.tst (rys. 30) i serwera rdns.robocza.tst (rys. 31).



Rys. 30. Zawartość pamięci podręcznej *resolvera* systemu klient.robocza.tst po operacji pobrania strony www.zasoby.tst

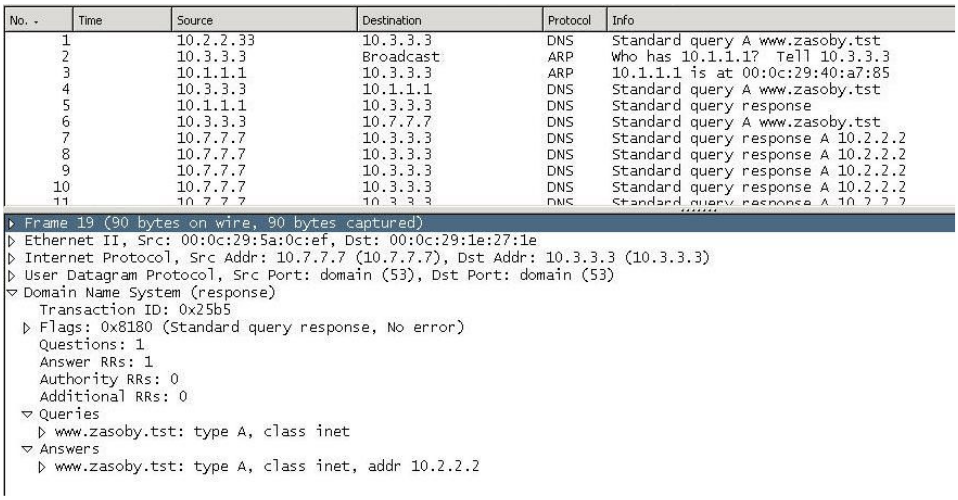


Rys. 31. Zawartość pamięci podręcznej serwera rdns.robocza.tst po operacji pobrania przez klienta strony www.zasoby.tst

Teraz przeprowadzany jest sam atak. Wykorzystano do tego skrypt opracowany i zamieszczony w [10]. Skrypt ten generuje zapytanie o określoną nazwę, a następnie wysyła do pytanego serwera DNS sfałszowane odpowiedzi [8]. Raport z uruchomienia skryptu zamieszczono na rys. 32. Obraz ruchu sieciowego generowanego przez ten skrypt przedstawiono na rys. 33. Należy zwrócić uwagę, że sfałszowane, wygenerowane przez skrypt odpowiedzi informują, że system www.zasoby.tst jest ulokowany pod adresem 10.2.2.2. Pod tym adresem intruz ulokował serwis www, który zamierza podstawić klientowi w miejsce serwisu www.zasoby.tst. Pole adresu źródłowego (10.7.7.7) wskazuje na komputer, który pełni rolę serwera DNS w domenie zasoby.tst. Ten komputer poprzednio udzielał odpowiedzi. W czasie przeprowadzonego ataku nie powinien być aktywny. W trakcie prawdziwego ataku zatruwania bufora oznacza to zwykle przeprowadzenie dowolnego ataku DoS skierowanego na ten komputer. W trakcie eksperymentu atak DoS zasymulowano wyłączając system 10.7.7.7.

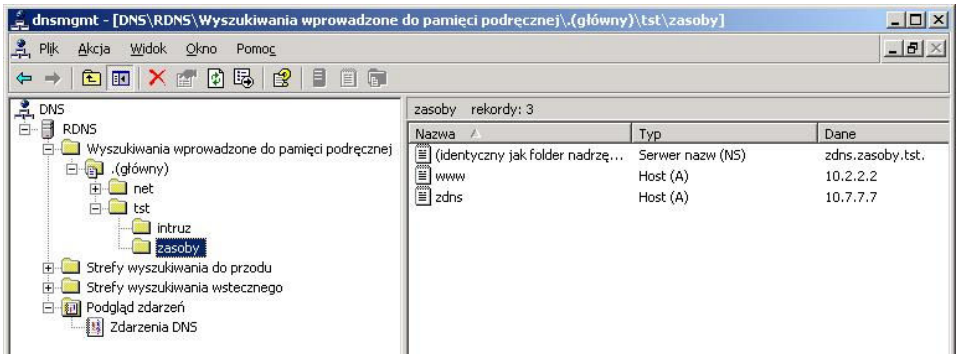
```
[root@lin exploit]# ./spoof.sh 10 2000 100

[root@lin exploit]#
```

Rys. 32. Raport z uruchomienia skryptu umożliwiającego przeprowadzenie ataku zatruwania bufora serwera DNS

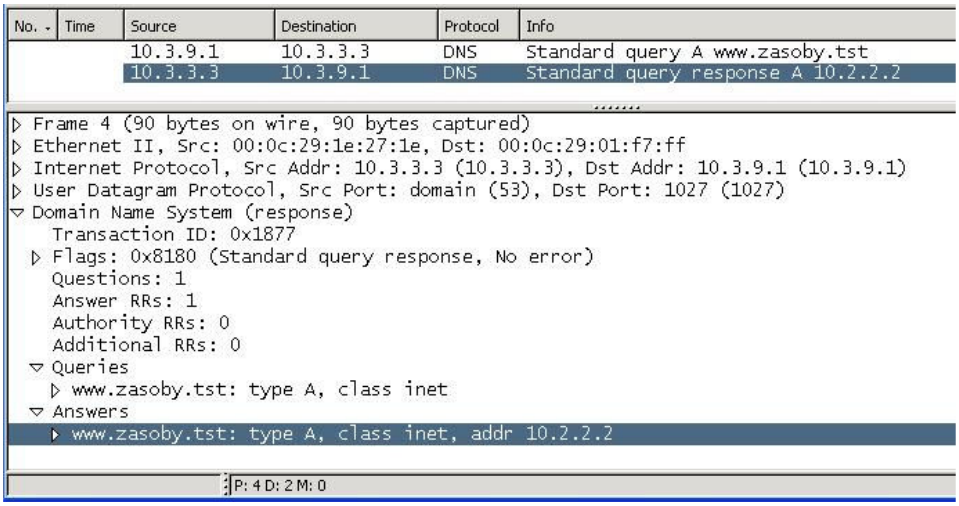


Rys. 33. Obraz ruchu sieciowego podczas ataku zatruwania bufora serwera DNS

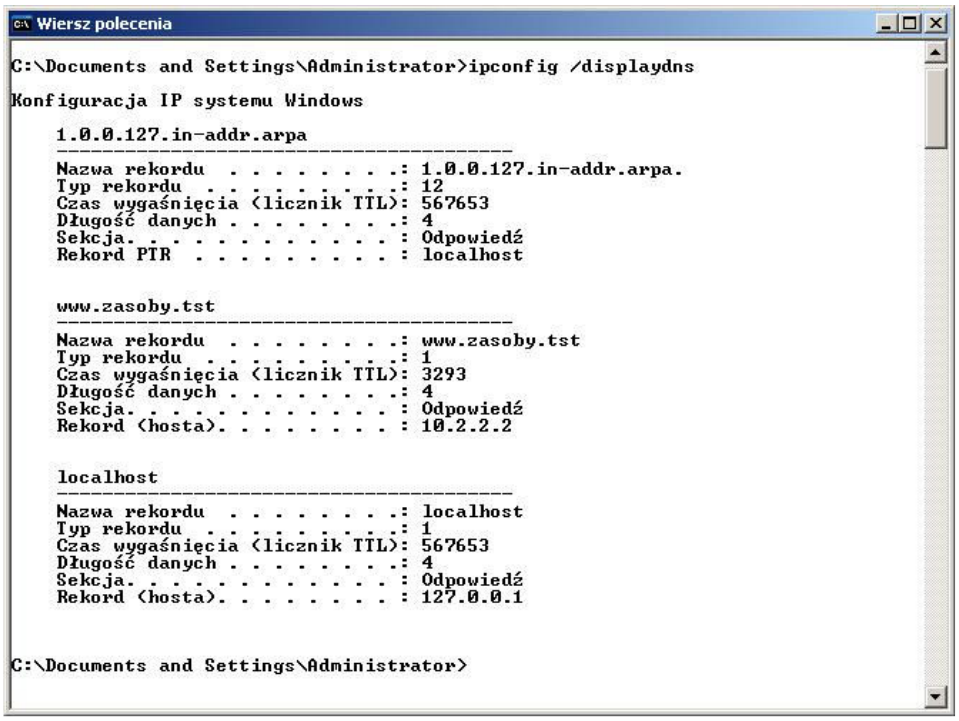
Po ataku, w buforze serwera DNS 10.3.3.3 znalazła się informacja, że system www.zasoby.tst jest ulokowany pod adresem 10.2.2.2 (rys. 34). Klient żądający rozwiązania nazwy www.zasoby.tst otrzyma nieprawdziwą odpowiedź, co obrazuje rys. 35. Zawartość bufora klienta przedstawiono na rys. 36.



Rys. 34. Zatruta zawartość bufora zaatakowanego serwera DNS

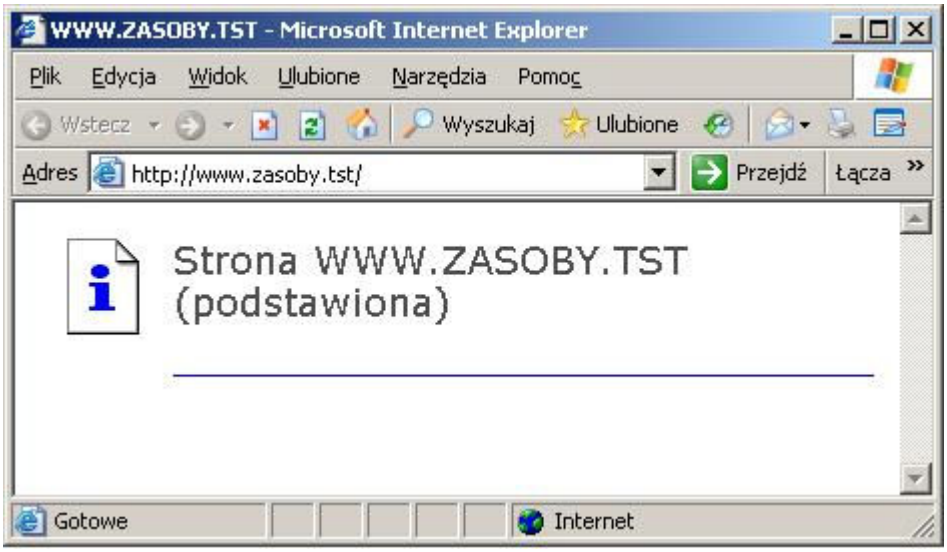


Rys. 35. Nieprawdziwa odpowiedź udzielona klientowi przez zaatakowany serwer DNS



Rys. 36. Zatruta zawartość bufora klienta zaatakowanego serwera DNS

W wyniku całego opisanego procesu, klient żądający dostępu do strony www.zasoby.tst, otrzyma stronę podstawioną przez intruza pod adresem 10.2.2.2. Ilustruje to rys. 37.

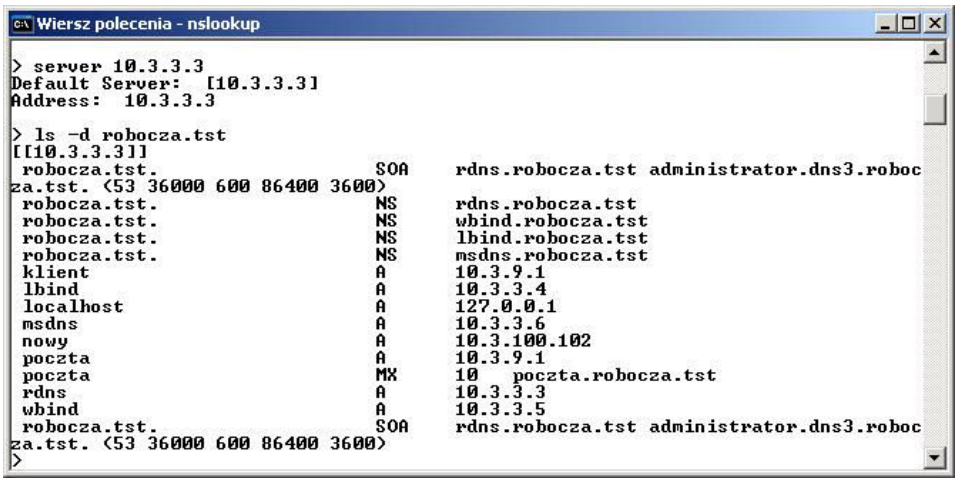


Rys. 37. Wynik próby pobrania przez klienta strony z domniemanego serwera www.zasoby.tst (faktycznie serwera podstawionego przez intruza)

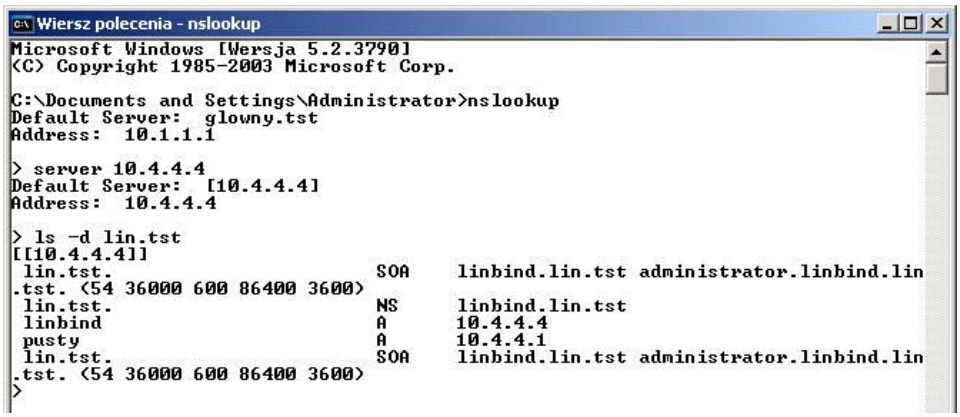
6. Dynamiczna aktualizacja

Funkcja dynamicznej aktualizacji może być wykorzystywana przez komputery w sieci lokalnej, które nie mają przypisanego stałego adresu IP. Podczas uruchamiania, system operacyjny stara się uaktualnić właściwy rekord w zasobach serwera DNS. To rozszerzenie dotyczące funkcjonowania DNS zostało zdefiniowane w RFC 2136 [16]. W dalszej części opisano eksperyment, którego celem było wprowadzenie do bazy serwera DNS rekordu przygotowanego przez potencjalnego intruza. Eksperyment przeprowadzono dla serwera *msdns* (rdns.robocza.tst) i bind 4.9.1 (linbind.lin.tst). Na rys. 38 i 39 przedstawiono wyniki pobierania informacji o strefie z obu serwerów przed rozpoczęciem eksperymentu.

Na rys. 40 i 41 pokazano zawartość plików strefowych rezydujących na obu serwerach (przed rozpoczęciem eksperymentu).



Rys. 38. Wyniki pobrania informacji o strefie utrzymywanej przez serwer rdns.robocza.tst za pomocą programu *nslookup*



Rys. 39. Wyniki pobrania informacji o strefie utrzymywanej przez serwer *bind* (linbind.lin.tst) za pomocą programu *nslookup*

Dodatkowo na rys. 42 przedstawiono obraz konsoli graficznej służącej do zarządzania serwerem *msdns* (przed rozpoczęciem eksperymentu).

Po opisanych czynnościach wstępnych uruchomiono opracowany program (*exploit*). Jego zadaniem było zrealizowanie procedury dynamicznej aktualizacji ze strony klienta. Można wykorzystać również program *nsupdate* dostępny w dystrybucji pakietu *bind*. Raport z uruchomienia opracowanego programu, dla obu opisywanych przypadków zamieszczono na rys. 43 i 44. Obraz ruchu sieciowego spowodowanego uruchomieniem tego programu przedstawiono na rys. 45 i 46. Ramkami zaznaczono najważniejsze pakiety zawierające dane

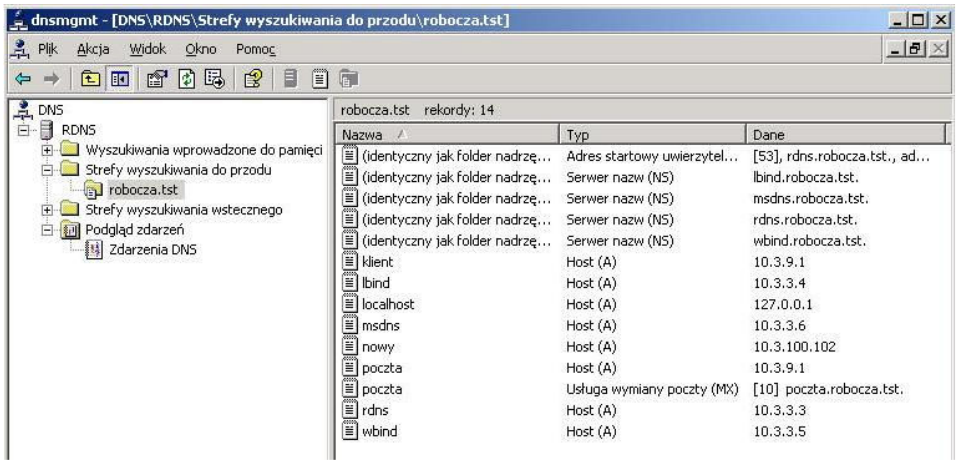
aktualizacyjne. Jak się można było spodziewać ruch sieciowy w obu przypadkach jest podobny, ale nie identyczny.

```
;  Database file robocza.tst.dns for robocza.tst zone.
;      Zone version:  53
@                                       IN    SOA  rdns.robocza.tst.
administrator.dns3.robocza.tst. (
                              53          ; serial number
                              36000       ; refresh
                              600         ; retry
                              86400       ; expire
                              3600      ) ; default TTL
;  Zone NS records
@                       NS    rdns.robocza.tst.
@                       NS    wbind.robocza.tst.
@                       NS    lbind.robocza.tst.
@                       NS    msdns.robocza.tst.
;  Zone records
klient                  1200  A     10.3.9.1
lbind                   A     10.3.3.4
localhost               A     127.0.0.1
msdns                   1200  A     10.3.3.6
nowy                    A     10.3.100.102
poczta                  A     10.3.9.1
                        MX    10    poczta.robocza.tst.
rdns                    A     10.3.3.3
wbind                   1200  A     10.3.3.5
```

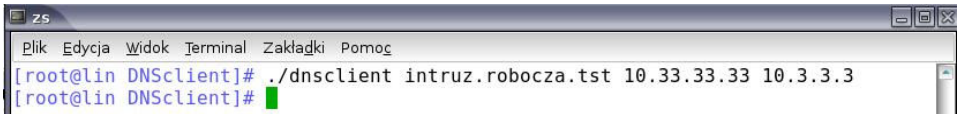
Rys. 40. Zawartość pliku *robocza.tst.dns* zawierającego informacje o strefie utrzymywanej przez serwer rdns.robocza.tst przed rozpoczęciem eksperymentu

```
$ORIGIN .
$TTL 3600   ; 1 hour
lin.tst                 IN SOA      linbind.lin.tst.
administrator.linbind.lin.tst. (
                        54          ; serial
                        36000       ; refresh (10 hours)
                        600         ; retry (10 minutes)
                        86400       ; expire (1 day)
                        3600        ; minimum (1 hour)
                        )
                NS      linbind.lin.tst.
$ORIGIN lin.tst.

$TTL 1200   ; 20 minutes
linbind                 A     10.4.4.4
pusty           A       10.4.4.1
```

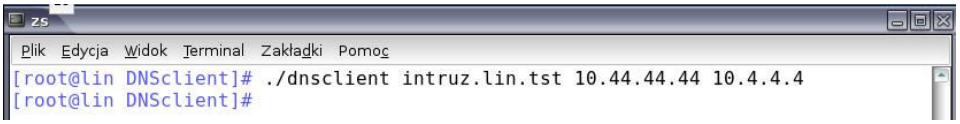
Rys. 41. Zawartość pliku *lin.tst.dns* zawierającego informacje o strefie utrzymywanej przez serwer *bind* (linbind.lin.tst) przed rozpoczęciem eksperymentu



Rys. 42. Obraz konsoli służącej do zarządzania serwerem rdns.robocza.tst przed rozpoczęciem eksperymentu

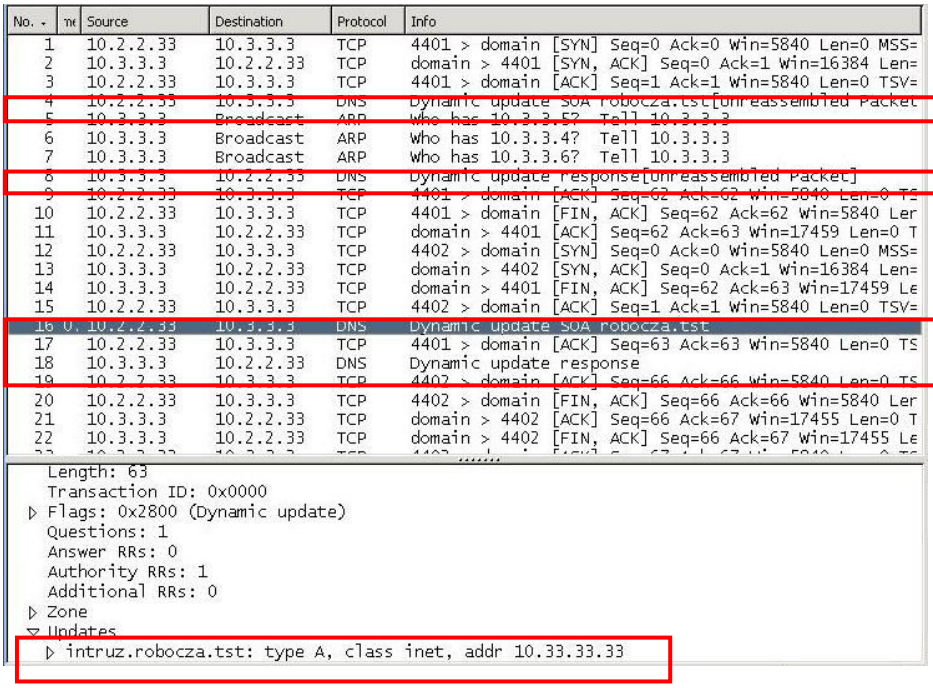


Rys. 43. Raport z uruchomienia programu umożliwiającego przeprowadzenie ataku dynamicznej aktualizacji serwera rdns.robocza.tst

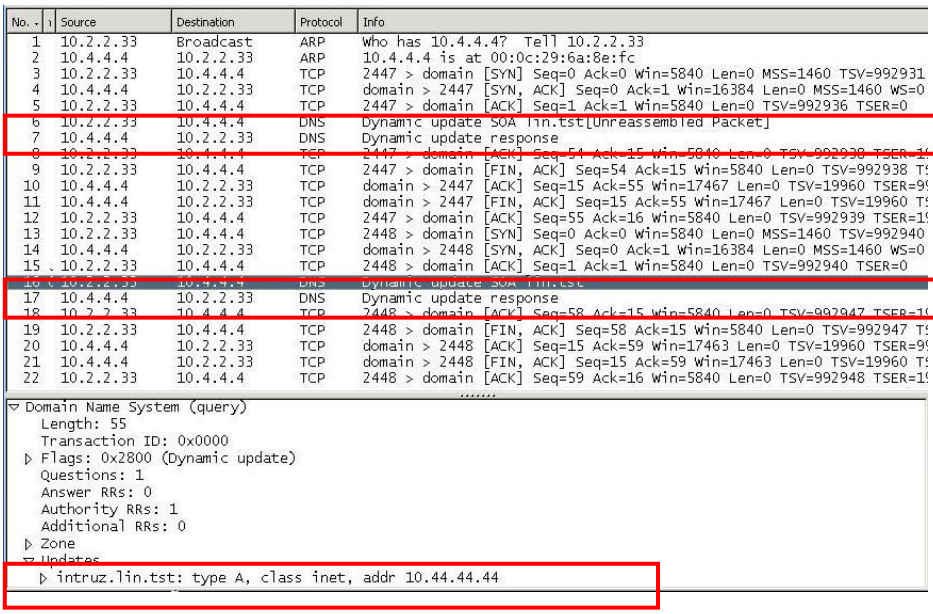


Rys. 44. Raport z uruchomienia programu umożliwiającego przeprowadzenie ataku dynamicznej aktualizacji serwera *bind* (linbind.lin.tst)

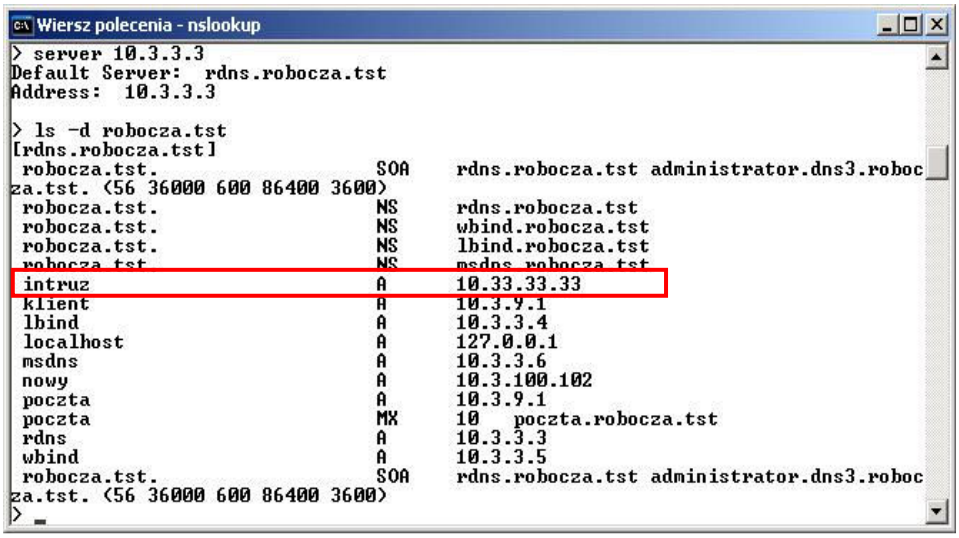
Skutki ataków można obejrzeć na rys. 47-51. Na wszystkich tych rysunkach ramkami zaznaczono elementy, które są skutkiem przeprowadzonych ataków. Na rys. 47 i 48 przedstawiono wyniki pobierania informacji o strefie z obu serwerów po przeprowadzonym ataku. Na rys. 49 i 50 pokazano zawartość plików strefowych rezydujących na obu serwerach. Dodatkowo na rys. 51 przedstawiono obraz konsoli graficznej służącej do zarządzania serwerem rdns.robocza.tst (po ataku).



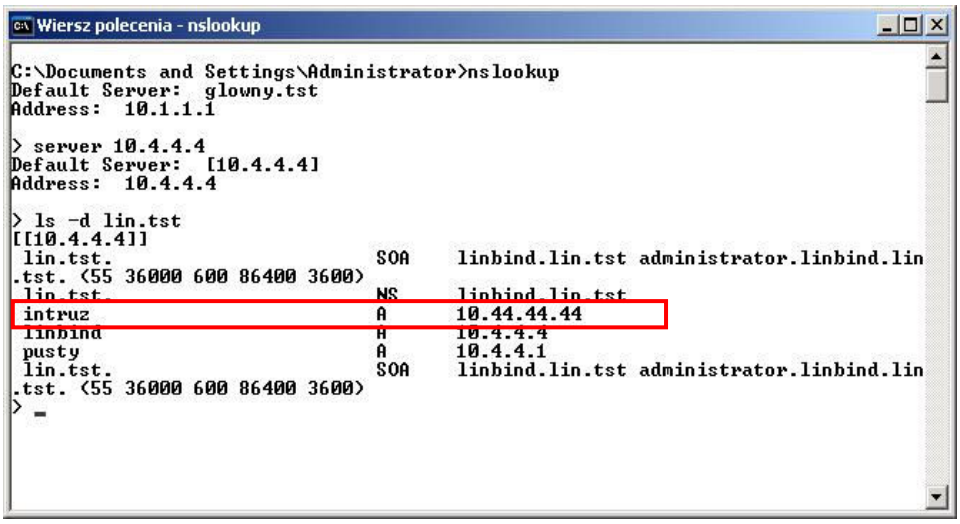
Rys. 45. Obraz ruchu sieciowego podczas ataku dynamicznej aktualizacji serwera rdns.robocza.tst



Rys. 46. Obraz ruchu sieciowego podczas ataku dynamicznej aktualizacji serwera *bind* (linbind.lin.tst)



Rys. 47. Wyniki pobrania informacji o strefie utrzymywanej przez serwer rdns.robocza.tst za pomocą programu *nslookup* – po ataku



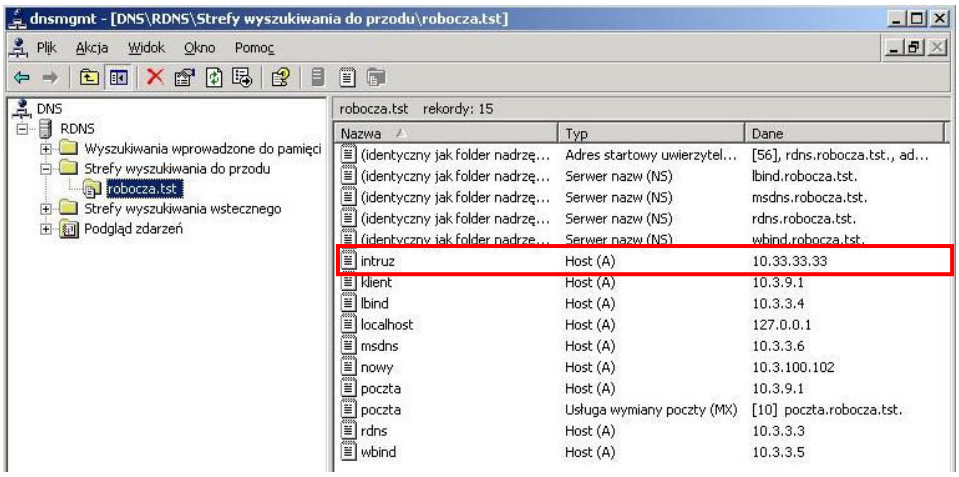
Rys. 48. Wyniki informacji o strefie utrzymywanej przez serwer *bind* (linbind.lin.tst) za pomocą programu *nslookup* – po ataku

```
;  Database file robocza.tst.dns for robocza.tst zone.
;      Zone version:  56
@                                       IN    SOA  rdns.robocza.tst.
administrator.dns3.robocza.tst. (
                              56           ; serial number
                              36000        ; refresh
                              600          ; retry
                              86400        ; expire
                              3600       ) ; default TTL
;  Zone NS records
@                         NS    rdns.robocza.tst.
@                         NS    wbind.robocza.tst.
@                         NS    lbind.robocza.tst.
@                         NS    msdns.robocza.tst.
;  Zone records
intruz                    300   A     10.33.33.33
klient                    1200  A     10.3.9.1
lbind                     A     10.3.3.4
localhost                 A     127.0.0.1
msdns                     1200  A     10.3.3.6
nowy                      A     10.3.100.102
poczta                    A     10.3.9.1
                          MX    10    poczta.robocza.tst.
rdns                      A     10.3.3.3
wbind                     1200  A     10.3.3.5
```

Rys. 49. Zawartość pliku *robocza.tst.dns* zawierającego informacje o strefie utrzymywanej przez serwer rdns.robocza.tst po ataku

```
$ORIGIN .
$TTL 3600   ; 1 hour
lin.tst                   IN SOA      linbind.lin.tst.
administrator.linbind.lin.tst. (
                          55          ; serial
                          36000       ; refresh (10 hours)
                          600         ; retry (10 minutes)
                          86400       ; expire (1 day)
                          3600        ; minimum (1 hour)
                          )
                    NS    linbind.lin.tst.
$ORIGIN lin.tst.
$TTL 300    ; 5 minutes
intruz                    A     10.44.44.44
$TTL 1200   ; 20 minutes
linbind                   A     10.4.4.4
pusty               A     10.4.4.1
```

Rys. 50. Zawartość pliku *lin.tst.dns* zawierającego informacje o strefie utrzymywanej przez serwer *bind* (linbind.lin.tst) po ataku



Rys. 51. Obraz konsoli służącej do zarządzania serwerem rdns.robocza.tst po ataku

7. Podsumowanie

W artykule zostały przedstawione wyniki eksperymentów, których celem było sprawdzenie podatności wybranych serwerów DNS na ataki. Można się zorientować, że przeprowadzenie wielu ataków jest bardzo proste i możliwe do wykonania nawet przez niezbyt zaawansowanego napastnika. Prezentacje zamieszczone w niniejszym artykule być może spowodują wzrost świadomości wśród administratorów systemów i przyczynią się do wzrostu bezpieczeństwa systemów, którymi się opiekują.

Dla części opisanych luk występujących w najbardziej popularnych implementacjach serwerów DNS opracowano już łaty programowe. Należy jednak zdawać sobie sprawę, że opracowanie łaty nie rozwiązuje problemu. Jest dopiero pierwszym krokiem na drodze do zbudowania bezpiecznego serwera. Taką łatę należy jeszcze zainstalować. Wydaje się to truizmem. Niestety należy o tym przypominać, gdyż większość użytkowników nie realizuje żadnych zabiegów pielęgnacyjnych w stosunku do używanego przez siebie oprogramowania.

Literatura:

[1] ALBITZ P., LIU C., *DNS and BIND. Edition 5*, O'Reilly Media Inc., Sebastopol, 2006.

[2] ARENDS R., AUSTEIN R., LARSON M., MASSEY D., ROSE S., *DNS Security Introduction and Requirements*, *RFC 4033*. IETF 2005.

[3] ARENDS R., AUSTEIN R., LARSON M., MASSEY D., ROSE S., *Resource Records for the DNS Security Extensions*, *RFC 4034*. IETF 2005.

[4] ARENDS R., AUSTEIN R., LARSON M., MASSEY D., ROSE S., *Protocol Modifications for the DNS Security Extensions*, *RFC 4035*. IETF 2005.

[5] ATKINS D., AUSTEIN R., *Threat Analysis of the Domain Name System (DNS)*, *RFC 3833*. IETF 2004.

[6] BIRKHOLZ E. P., *Operacje specjalne – Bezpieczeństwo komputerów i sieci Microsoft UNIX, ORACLE*, Translator, Warszawa, 2003.

[7] BORZYM M., „*Weryfikacja bezpieczeństwa serwerów DNS*", praca magisterska, WAT, Warszawa, 2006.

[8] BORZYM M., SUSKI Z., *Zagrożenia usługi DNS*, Biuletyn IAiR 25/2008, WAT, Warszawa, 2008.

[9] DANIELS A., KNIEF H., GRAHAM J., ABELL R., *Windows 2000 DNS*, Helion, Gliwice, 2001.

[10] HAŁAJKO G., „*Bezpieczeństwo serwerów DNS*", praca magisterska, PJWSTK, Warszawa, 2007.

[11] HATCH B., LEE J., KURTZ G., *Hakerzy w Linuksie. Sekrety zabezpieczeń sieci komputerowych*, Translator, Warszawa, 2003.

[12] LIU C., LARSON M., ALLEN R., *DNS on Windows Server 2003*, *Edition 3* O'Reilly Media Inc., Sebastopol, 2003.

[13] MOCKAPETRIS P., *Domain Names – Concepts And Facilities*, *RFC 1034*. IETF 1987.

[14] MOCKAPETRIS P., *Domain Names – Implementation And Specification*, *RFC 1035*. IETF 1987.

[15] TOMASZEWSKI M., *Pharming-Ataki DNS cache poisoning*, Hakin9 4/2005, str. 14-22.

[16] VIXIE P., THOMSON S., REKHTER Y., BOUND J., *Dynamic Updates in the Domain Name System (DNS UPDATE)*, *RFC 2136*. IETF 1997.

The researches of DNS susceptibility to selected threats

ABSTRACT: The paper presents results of penetrative tests. The goal of the tests was a verification of DNS servers susceptibility to chosen threats.

KEYWORDS: penetrative tests, security threats, DNS

Praca wpłynęła do redakcji: 15.09.2010.