Wyobraźmy sobie, że jesteśmy w zasięgu sieci WLAN. Nie, nie żadnej prywatnej, w tym artykule nie będziemy łamać szyfrowania WEP ;) . Publicznej, ale płatnej sieci bezprzewodowej. Przy próbie otwarcia dowolnej strony pojawia się stronka dostawcy, na której możemy zapłacić.

I tu mamy lukę... aby udostępnić tą stronkę www, ISP w większości przypadków udostępnia także serwer DNS, przez który, jeśli mamy szczęście, możemy pobierać dane z innych serwerów DNS.

I właśnie postawimy serwer DNS, tylko że zamiast pełnić zwyczajną funkcję, będzie przekazywał dane.

Co potrzebne:

- serwer - komputer z publicznym adresem IP, który będzie włączony gdy będziemy chcieli skorzystać z bezpłatnego dostępu do płatnej sieci. Nie może to być serwer na którym jednocześnie działa "zwykły" serwer DNS (np. bind)

Przygotowania DNS-ów:

1. Sprawdzamy, pod jakim adresem IP jest z zewnątrz widziany komputer - serwer, np. na stronie whatismyip - przyjmijmy, że mamy IP 1.2.3.4.
2. Ustawiamy domenę z rekordem NS - w celu założenia darmowej korzystamy z serwisów dot.tk czy też bee.pl - załóżmy że naszadomena.bee.pl to rekord NS wskazujący na naszadomena-1.bee.pl, a naszadomena-1.bee.pl to rekord A wskazujący na 1.2.3.4. Takie komplikacje biorą się z technicznych ograniczeń, mianowicie rekord NS musi wskazywać na nazwę, nie na adres IP.

Mamy domenę. Kolejne kroki zależą od wybranego oprogramowania do przekazywania (serwer i klient pseudo-DNS). Dość niezawodnym i prostym w obsłudze pakietem jest iodine , który znajduje się w repo Debiana, Archa i zapewne innych dystrybucji GNU/Linuksa. Można też ściągnąć binarkę pod win32, ale NIE TESTOWAŁEM.

Uruchomienie iodine jest dziecinnie proste (prościej się nie da). Najpierw przedstawię komendę uruchamiającą serwer (uruchamiamy z roota):

{codecitation class="brush:bash"}

iodined -n 1.2.3.4 -P naszehaslo 172.16.0.1 naszadomena.bee.pl

{/codecitation}

Przyjęliśmy tutaj, że będziemy stosować podsieć prywatną 172.16.x.x. Wybrałem tą, ponieważ praktycznie jest bardzo rzadko stosowana - unikniemy kolizji. Parametr -n nie jest konieczny, lecz jeśli serwer nie jest podłączony bezpośrednio do sieci, ale przez NAT (w takim przypadku będzie konieczne przekierowanie portów), to będzie myślał że ma adres IP np. 192.168.1.12 - bo taki ma, tyle że prywatny. Parametr -n wymusza adres publiczny. Parametr -P to hasło dostępu do przekaźnika, takie samo ustawimy u klienta. Domenę na końcu linii komend także należy zmienić.

Wspomniałem że uruchomienie iodine jest bardzo proste. Ale musimy jeszcze zmienić tablice routingu tak, aby przekazywały ruch na świat przez nasz serwer DNS. Zajmuje się tym skrypt klienta (uruchamiamy go z roota): tutaj go zobacz

Na początku skryptu znajdziemy konfigurację - zmieniamy ją na własną. Zmiennej TUN_GW zazwyczaj nie będziemy musieli ruszać, o ile nie okaże się że jednak mamy kolizję z istniejącą podsiecią. W takim wypadku zmieniamy również odpowiedni adres w komendzie serwera.

A oto krótka ściąga co należy zmienić:

- 1.2.3.4 - adres IP serwera
- naszehaslo - zmieniamy na dowolne hasło
- naszadomena.bee.pl - domena z rekordem NS

Całość wyłączamy przez zrestartowanie całego połączenia... a może ktoś się pokusi o stworzenie skryptu przywracającego pierwotne ustawienia routingu? :)

Szczegóły techniczne zmian w tablicach routingu - TYLKO DLA DOCIEKLIWYCH:

Skrypt klienta rekonfiguruje tablice routingu tak, aby prawie cały ruch był przekazywany przez interfejs dns0 - czyli emulowane przez iodine urządzenie, przekazujące pakiety. Właśnie - prawie cały. Gdybyśmy przekazali naprawdę cały, to którędy szły by zapytania do serwerów DNS?

Dlatego zanim zmienimy wpis bramy domyślnej (sieć 0.0.0.0, maska 0.0.0.0), do tablic routingu dodajemy reguły każące przekazywać ruch do DNS-ów (wziętych z pliku resolv.conf). Dzięki temu nie odetniemy się od sieci. Na wypadek gdyby firewall ISP przepuszczał ruch DNS do dowolnych komputerów, dodajemy również regułę każącą przekazywać bez kombinowania ruch do serwera (będzie szybciej niż z pośredniczącymi DNSami). Czasami może być to niepożądane, w takim wypadku wykomentuj linijkę "$R add -host $NS_ADDR gw $local_gw dev $local_if".

Rozwiązywanie problemów:

Q: Marudzi o niezgodność wersji przy łączeniu, co robić?

A: Problem występuje np. gdy wykorzystujemy różne distra GNU/Linuksa. Należy wówczas na obu komputerach skompilować i zainstalować z tych samych źródeł.

Q: Pingowanie działa, ale połączenie TCP nie chce się nawiązać.

A: Prace trwają... czasami zadziała dodanie przełącznika -m 500 do komendy serwera. Możemy zmniejszyć tą liczbę. Jest to maksymalna wielkość pakietu który możemy wysłać łączem. Jeśli serwer DNS ma niski limit to domyślna wartość okazuje się za duża.

Q: Jak sprawdzić czy to w ogóle działa?

A: Przez wireshark, jeśli podczas pingowania czegokolwiek (poza lokalną podsiecią) po "fizycznym" interfejsie latają pakiety DNS, a nie ICMP echo, to znaczy że działa.

Legalność

Nie łamiemy zabezpieczeń, więc nie musimy obawiać się policji czy sądu. Naruszenie prawa wystąpi jedynie, gdy w regulaminie usługi napisane jest wprost, że takie łączenie jest zabronione.