Konfiguracja serwera FreeRADIUS

Uruchamianie i zatrzymywanie serwera

W trakcie laboratorium zaleca się uruchamianie serwera w trybie debug. W trybie tym serwer działa na pierwszym planie i wypisuje na ekranie wszystkie komunikaty dotyczące swojej pracy.
Aby uruchomić serwer w trybie debug należy użyć polecenia:
radiusd –X
Aby zatrzymać serwer pracujący w tym trybie, należy posłużyć się kombinacją klawiszy:
Ctrl+C

Lokalizacja plików serwera

Wszystkie pliki konfiguracyjne serwera FreeRADIUS zlokalizowane są w folderze:
/etc/raddb lub **/etc/freeradius**

Plik: clients.conf

W pliku tym umieszczamy adresy wszystkich klientów (na przykład: punktów dostępowych lub innych serwerów RADIUS), które będą wymieniać informacje z naszym serwerem RADIUS. Serwer będzie przyjmował żądania wyłączenie od klientów zdefiniowanych w tym pliku.
Plik składa się z sekcji stworzonych wg następującego wzorca:

```
client <hostname|ip-address|ip-network> {
          secret          = <wartość>
          shortname       = <wartość>
          <opcja>         = <wartość>
}
```

W miejscu ***<hostname | ip-address | ip-network>*** umieszczamy nazwę lub adres klienta, którego serwer ma obsługiwać. Można tu użyć:

- Nazwy, którą serwer jest w stanie przetłumaczyć na adres IP.
- Adresu IP.
- Adresu sieci w formacie CIDR, czyli ***<adres sieci>/<maska>***.

Wewnątrz tak zdefiniowanej sekcji, można umieścić opcje konfiguracyjne, z których dwie: ***secret*** i ***shortname*** są wymagane:

- secret – określa hasło używane do zabezpieczania komunikacji pomiędzy klientem i serwerem RADIUS. Musi zostać ustawione na tą samą wartość na kliencie i serwerze.
- shortname – umowna nazwa klienta. Może zostać ustawiona dowolnie.

Przykłady:

Klient o adresie 192.168.44.253, posługujący się hasłem „haslo”.

```
client 192.168.44.253 {
        secret          = haslo
        shortname       = CiscoAP350
}
```

Dowolny klient (lub wielu klientów) o adresach od 192.168.44.1 do 192.168.44.254, posługujący się hasłem „haslo354”.

```
client 192.168.44.0/24 {
        secret          = haslo354
        shortname       = GrupaKlientow
}
```

Plik: users

Plik ten jest jedną z baz użytkowników, z którymi może współpracować serwer FreeRADIUS.
Wpisy w nim umieszczone składają się z:

- **pierwszej linii** zawierającej nazwę użytkownika, oraz listę parametrów pozwalających na jego uwierzytelnienie (np. hasło) lub polecenie nakazujące bezwarunkowe dopuszczenie/odrzucenie użytkownika,
- **(opcjonalnie) dalszych linii**, zaczynających się znakiem tabulatora, zawierających listę **opcji które serwer ma przesłać klientowi** w celu konfiguracji chcącego się podłączyć użytkownika.

Serwer porównuje powyższą nazwę użytkownika z tą przysłaną przez klienta, a następnie sprawdza dalszy ciąg pierwszej linii wpisu co należy dalej zrobić.
Możemy nakazać np:

- bezwarunkowe zaakceptowanie użytkownika:
  `Auth-Type := Accept`
- bezwarunkowe odrzucenie użytkownika:
  `Auth-Type := Reject`
- sprawdzenie hasła:
  `Cleartext-password := “haslo”`
- sprawdzenie użytkownika w bazie danych systemu Linux:
  `Auth-Type := System`
- użycie określonej wartości **realm** (zastosowanie tego polecenia podano przy opisie pliku ***proxy.conf***):
  `Proxy-To-Realm := LOCAL`

UWAGA: Plik przeszukiwany jest do momentu znalezienia pierwszego pasującego wpisu i decyduje on o dalszych działaniach. Kolejne wpisy nie są przeglądane.

Przykłady:

```
user1           Auth-Type := Reject
```

Nakazuje bezwarunkowe odrzucenie użytkownika **user1**.

```
"John Doe"      Cleartext-Password := "hello"
```

Dopuszcza użytkownika **John Doe**, jeśli podał hasło **hello**.

```
user2           Auth-Type := Accept
```

Nakazuje bezwarunkowe zaakceptowanie użytkownika **user2**.

```
user3           Cleartext-Password := "hello"
                Framed-Protocol = PPP,
                Framed-Compression = Van-Jacobsen-TCP-IP
```

Dopuszcza użytkownika **user3**, oraz ustawia dla niego 2 opcje: “Framed-Protocol” oraz “Framed-Compression” na wartości odpowiednio: „PPP” i „Van-Jacobsen-TCP-IP”.

Plik: radiusd.conf

Jest głównym plikiem konfiguracyjnym serwera FreeRadius i umożliwia daleko idącą zmianę sposobu pracy serwera. Podczas laboratorium nie przewidujemy potrzeby tak daleko idących zmian, a zatem i edycji pliku **radiusd.conf**.

Plik: eap.conf

Plik eap.conf odpowiada za ustawienia protokołu EAP, który umożliwia użytkownikom uwierzytelnianie z użyciem jednego z wielu możliwych mechanizmów oraz przesyłanie kluczy szyfrujących.
Szczególnie interesującym nas przypadkiem jest odmiana protokołu EAP zwana Protected EAP (PEAP). W jej przypadku komunikacja odbywa się szyfrowanym tunelem TLS, oraz możliwe jest bezpieczne przesłanie przez serwer kluczy szyfrujących WEP/WPA do podłączającego się urządzenia bezprzewodowego.
Przykładową strukturę pliku eap.conf przedstawiono poniżej:

```
eap {
      tls {
            private_key_password = whatever
            private_key_file = ${raddbdir}/certs/cert-srv.pem
            certificate_file = ${raddbdir}/certs/cert-srv.pem
            CA_file = ${raddbdir}/certs/demoCA/cacert.pem
            dh_file = ${raddbdir}/certs/dh
            random_file = ${raddbdir}/certs/random
            fragment_size = 1024
            include_length = yes
      }
      peap {
            default_eap_type = mschapv2
      }
      mschapv2 {
      }
}
```

Pierwsza sekcja (tls), odpowiada za ustawienia szyfrowanego tunelu. Zostały tu wyszczególnione, między innymi, pliki zawierające:

- ***klucz prywatny serwera*** (private_key_file) oraz hasło dostępu do tego klucza (private_key_password),
- ***certyfikat serwera*** zawierający klucz publiczny (certificate_file),
- informacje (w tym certyfikat) ***wystawcy certyfikatu*** serwera (CA_file).

Klienci mogą, z ich użyciem, ustalić klucz szyfrujący wymianę danych protokołu PEAP oraz zweryfikować tożsamość serwera z którym się połączyli (zapobiega to atakowi Man-In-The-Middle).

Obecność **sekcji drugiej (peap)** powoduje włączenie obsługi ***protokołu uwierzytelniania*** PEAP. Sekcja w nawiasach klamrowych zawiera informacje konfiguracyjne tego protokołu, a w szczególności listę ***metod uwierzytelniania***, którymi może posłużyć się klient. Należy tu pamiętać, iż protokół PEAP umożliwia posłużenie się wieloma ***metodami uwierzytelniania***, samemu stanowiąc dla nich tylko „platformę działania”.

Ponieważ podczas laboratorium posługiwać będziemy się klientami Windows XP, ustawiamy domyślną metodę uwierzytelniania dla protokołu PEAP na **mschapv2**.

Sekcja trzecia (mschapv2) zawiera informacje o konkretnej ***metodzie uwierzytelniania***, stosowanej przez protokoły uwierzytelniania. Podobnie jak w przypadku protokołu PEAP, w nawiasach klamrowych można umieszczać opcje konfiguracyjne, które jednakże nie są w tym przypadku wymagane.
Sama sekcja natomiast musi występować, gdyż jej brak spowoduje wyłączenie obsługi tej metody uwierzytelniania na serwerze RADIUS. Stanie się tak pomimo, że skonfigurowaliśmy protokół PEAP do użycia wyłącznie tej metody – w efekcie nie można się będzie uwierzytelnić protokołem PEAP.

Plik: proxy.conf

Plik proxy.conf pozwala na przekierowanie zgłoszeń odebranych przez nasz serwer do innych serwerów RADIUS. Może to służyć, na przykład, do:

- rozłożenia obciążenia na wiele serwerów RADIUS,
- zapewnienia bezawaryjności, dzięki zdefiniowaniu serwerów zapasowych,
- delegowania zadań zarządzania siecią.

Pierwsza znajdująca się w pliku sekcja, to sekcja **proxy**:

```
proxy server {
        synchronous = no
        retry_delay = 5
        retry_count = 3
        dead_time = 120
        default_fallback = yes
        post_proxy_authorize = yes
}
```

Zawiera ona ogólne ustawienia mechanizmów przekierowywania zgłoszeń. Dla naszych celów nie jest konieczne dokonywanie tu żadnych modyfikacji.

Dalej może znajdować się pewna liczba sekcji **realm**. Każda z nich definiuje grupę zgłoszeń, które będą przekazywane na jakiś serwer RADIUS.
Czy któraś z nich zostanie użyta, decydują 2 mechanizmy:

- w standardowej konfiguracji serwera, podanie przy logowaniu nazwy użytkownika w formie: ***user@realm*** lub ***user%realm*** spowoduje użycie sekcji realm o nazwie podanej po znaku **@** lub **%**,
- zdefiniowanie w pliku ***users*** działania ***Proxy-To-Realm:=<nazwa>*** dla danego użytkownika, spowoduje użycie sekcji realm o podanej nazwie.

Każda z tych metod działa niezależnie i wystarczy zastosowanie jednej z nich.

Sekcje realm mają następującą składnię::

```
realm <nazwa> {
     type            = radius
     authhost        = <adres lub nazwa serwera RADIUS>:<port>
     secret          = <haslo do serwera>
}
```

`type = radius` – określa że przekierowujemy zgłoszenie na serwer typu RADIUS.

authhost = <adres lub nazwa>:<port> – podajemy tu adres lub nazwę, a po dwukropku port UDP serwera, na który chcemy przekierować zgłoszenie,
secret = <haslo> – podajemy tu hasło pozwalające na połączenie się z serwerem na który przekierowujemy zgłoszenie.

Opcja nostrip

Możliwe jest też dodanie opcji **nostrip**, która powoduje, że do serwera na który przekierowujemy zgłoszenie zostanie przesłana pełna nazwa użytkownika (w formacie ***user@realm*** lub ***user%realm***).

```
realm <nazwa> {
     type            = radius
     authhost        = <adres lub nazwa serwera RADIUS>:<port>
     secret          = <haslo do serwera>
     nostrip
}
```

W przypadku braku tej opcji, ciąg po znaku **@** lub **%** zostanie usunięty przed przekierowaniem zgłoszenia – czyli zostanie przesłana tylko część user.

UWAGA: W przypadku realizacji uwierzytelniania z użyciem protokołu PEAP, zastosowanie opcji **nostrip** jest konieczne. W przeciwnym przypadku nazwa użytkownika przesyłana przez serwer przekierowujący zgłoszenie, nie będzie zgodna z tą zawartą wewnątrz zaszyfrowanych pakietów PEAP. Niezgodność ta spowoduje odrzucenie zgłoszenia jako nieprawidłowego przez serwer na który dokonano przekierowania.

Przykład:
bez opcji nostrip: **user1@realm1.net** przekierujemy dalej jako **user1**.
z opcją nostrip: **user1@realm1.net** przekierujemy dalej jako **user1@realm1.net**.

Przykład:
Zdefiniowanie następującej sekcji realm spowoduje, że wszystkie zgłoszenia z nazwami użytkowników posiadającymi przyrostek „***@serwer2***" lub „***/serwer2***" (czyli np. „user1@serwer2" lub „test/serwer2") zostaną przekierowane do serwera RADIUS o adresie ***192.168.1.55 na port 1812*** w celu rozstrzygnięcia czy użytkownik ma prawo dostępu. W takim wypadku serwer RADIUS na którym zdefiniowano to przekierowanie nie bierze udziału w decyzji czy przyznać danemu użytkownikowi dostęp.
Natomiast zgłoszenie nie posiadające wyżej wymienionych przyrostków, będą rozstrzygane przez obecny serwer (chyba że zdefiniowano więcej realms i któreś z nich zostanie użyte).
Należy tu również zauważyć, że zdefiniowano opcję **nostrip**, a więc nazwy użytkownika zostaną przesłane serwerowi 192.168.1.55 w nie zmienionej formie, np. „***user1@serwer2***", „***test/serwer2***". Bez tej opcji przesłano by je w formie „***user1***" i „***test***".

```
realm serwer2 {
     type            = radius
     authhost        = 192.168.1.55:1812
     secret          = haslo987
     nostrip
}
```