

Bezpieczeństwo Sieci Komputerowych

Laboratorium 5: Bezpieczeństwo infrastruktury sieciowej – v1.1

Instrukcja do instrukcji

Kilka informacji wstępnych:

W porównaniu z wcześniejszymi zajęciami, w instrukcji do tych opisane jest bardzo dużo, ale i tak nie wszystko, poza tym jest parę błędów i nieścisłości. W tym opisie powinny znajdować się wszystkie komendy do wykonania na sprzęcie sieciowym potrzebne w czasie zajęć i wyjaśnienia do najistotniejszych z nich, polecam pracę z tym tekstem jako uzupełnienie instrukcji. Dodatkowo podałem tak jak zawsze jakąś informację praktyczną.

Większość podanych komend na urządzeniach Cisco trzeba wykonywać w trybie uprzywilejowanym, część z nich w różnych trybach konfiguracji. Według konwencji, której trzymałem się w opisie, przed komendą podany jest pełny ciąg znaków zachęty z wyjątkiem nazwy hosta, przykładowo:

> – taką komendę można wykonać w trybie zwykłego użytkownika

– taką komendę należy wykonać bezpośrednio w trybie uprzywilejowanym (*enable*)

(*config*)# – taką komendę trzeba wykonać w trybie konfiguracji globalnej (*config t*)

(*config-if*)# – taką komendę trzeba wykonać w trybie konfiguracji interfejsu (np. *int f0/1*)

Jeżeli jesteśmy w trybie konfiguracji i chcemy wykonać jakąś komendę wywoływaną bezpośrednio w trybie uprzywilejowanym (np. *show run*), możemy poprzedzić ją komendą *do*:

```
(config)# do show run
```

W ćwiczeniu będą potrzebne dwa komputery, jeden z Windowsem 7 (trzeba będzie zainstalować na nim Caina), drugi jak wolicie – Windows, Linux. W niektórych punktach potrzebny będzie też Wireshark.

Punkt 1.

Ćwiczenie zaczynamy od konfiguracji przełącznika (switcha). Ustawiamy nazwę urządzenia (po numerze grupy można jeszcze w sumie dopisać nazwiska):

```
(config)# hostname S<nr_grupy>
```

Poniższe ustawienie nie jest konieczne, ale warto je skonfigurować, aby komunikaty systemowe na konsoli przełącznika nie nachodziły na wpisywane polecenia:

```
(config)# line con 0
```

```
(config-line)# logging synchronous
```

Teraz możemy przejść do właściwego zadania, czyli skonfigurowania mechanizmu bezpieczeństwa na porcie przełącznika. Ma być dozwolony jeden adres MAC w trybie lepkim (*sticky*), w razie naruszenia port ma być wyłączany (*shutdown*) – przykład dla portu *fa0/2*:

```
(config)# int f0/2
```

```
(config-if)# switchport mode access
```

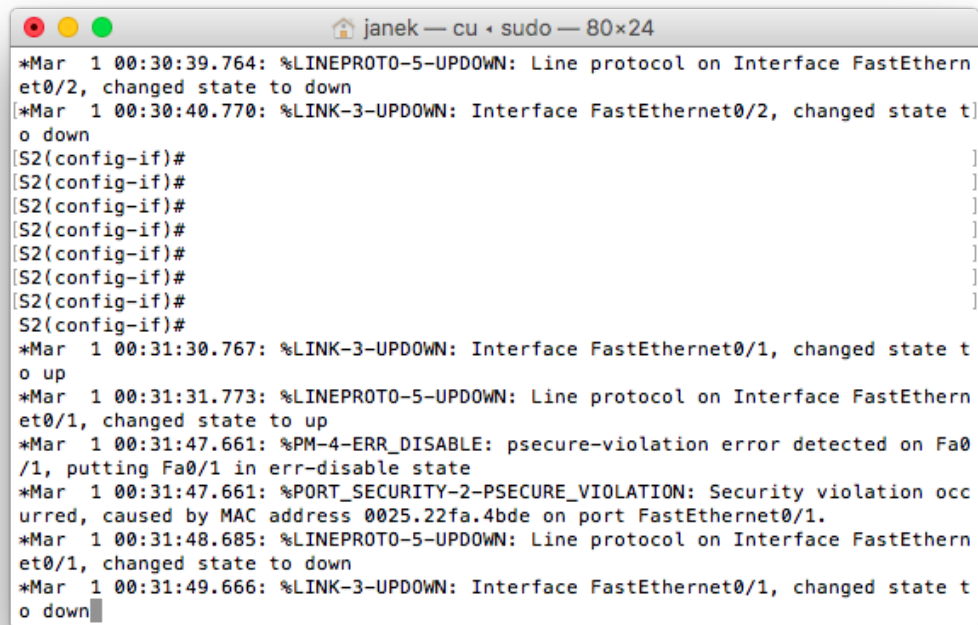
```
(config-if)# switchport port-security
```

```
(config-if)# switchport port-security mac-address sticky
```

Domyślna liczba dozwolonych adresów MAC to 1, a domyślną akcją w razie naruszenia jest *shutdown* – więc nie musimy tego konfigurować jawnie. Po podłączeniu do tego portu jednego z komputerów (nie musimy na razie konfigurować adresów IP), adres MAC jego karty sieciowej zostanie automatycznie zapamiętany przez przełącznik. Konfigurację i stan zabezpieczeń portu możemy wyświetlić komendą (poza trybem konfiguracji – jak napisałem na początku, jeżeli nie chcemy z niego wychodzić – możemy poprzedzić ją komendą *do*):

```
# show port-security int f0/2
```

Po podłączeniu drugiego komputera na konsoli przełącznika pojawi się komunikat o naruszeniu bezpieczeństwa i port oczywiście zostanie wyłączony:

A screenshot of a terminal window titled 'janeK — cu — sudo — 80x24'. The terminal displays a series of system messages related to port security on a switch. The messages indicate that the line protocol on interface FastEthernet0/2 was shut down, followed by a link shutdown. Then, after several configuration attempts, the line protocol on FastEthernet0/1 was brought up, but it was subsequently shut down due to a security violation on port FastEthernet0/1. The violation was caused by a MAC address (0025.22fa.4bde) that was not in the allowed list. The terminal output is as follows:

```
*Mar 1 00:30:39.764: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/2, changed state to down
*Mar 1 00:30:40.770: %LINK-3-UPDOWN: Interface FastEthernet0/2, changed state to down
[S2(config-if)#
[S2(config-if)#
[S2(config-if)#
[S2(config-if)#
[S2(config-if)#
[S2(config-if)#
[S2(config-if)#
[S2(config-if)#
*Mar 1 00:31:30.767: %LINK-3-UPDOWN: Interface FastEthernet0/1, changed state to up
*Mar 1 00:31:31.773: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to up
*Mar 1 00:31:47.661: %PM-4-ERR_DISABLE: psecure-violation error detected on Fa0/1, putting Fa0/1 in err-disable state
*Mar 1 00:31:47.661: %PORT_SECURITY-2-PSECURE_VIOLATION: Security violation occurred, caused by MAC address 0025.22fa.4bde on port FastEthernet0/1.
*Mar 1 00:31:48.685: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to down
*Mar 1 00:31:49.666: %LINK-3-UPDOWN: Interface FastEthernet0/1, changed state to down
```

Na przełączniku będzie to sygnalizowane zaświeceniem pomarańczowej diody (świecącej się nawet po odłączeniu kabla). Żeby zresetować port po podłączeniu z powrotem do pierwszego komputera, musimy go ręcznie wyłączyć i włączyć z powrotem:

```
(config)# int f0/2
(config-if)# shutdown
(config-if)# no shutdown
```

Jeżeli z jakiegoś powodu port zostałby mimo wszystko wyłączony znowu, możemy po prostu wyłączyć zabezpieczenia albo przełączyć komputer do innego portu – na przełączniku i tak już nic istotnego nie będzie trzeba robić.

Punkt 2.

W tym punkcie będziemy konfigurować prostą sieć i router, wykorzystujemy adresy IP z zakresu 192.168.x.0/24, gdzie *x* to nr naszej grupy. Łączymy sieć według schematu i zaczynamy konfigurację routera – ustawiamy nazwę hosta (możemy dopisać też nazwiska):

```
(config)# hostname R<nr_grupy>
```

Dodajemy 2 konta użytkowników zabezpieczone zwykłym hasłem:

```
(config)# username nazwisko_1 password haslo_1
(config)# username nazwisko_2 password haslo_2
```

Konfigurujemy dostęp do routera przez telnet:

```
(config)# line vty 0 4
(config-line)# transport input telnet
(config-line)# password haslo_telnet
(config-line)# login
```

W instrukcji podana jest komenda *secret* (hash MD5) zamiast *password* (niezbyt bezpieczne szyfrowanie Type 7), ale nie działa ona z jakiegoś powodu przy konfiguracji hasła dla telnetu. Następnie konfigurujemy adres IP – założmy, że przełącznik podłączyliśmy do portu *ge0/1*:

```
(config)# int g0/1
(config-if)# ip address 192.168.x.1 255.255.255.0
(config-if)# no shut
```

Po skonfigurowaniu adresów IP na komputerach (jakieś dowolne z zakresu, jako adres bramy oczywiście ustawiamy adres routera) powinna być możliwa komunikacja między wszystkimi urządzeniami. Nie konfigurujemy adresu IP na przełączniku, nie będzie to nam do niczego potrzebne.

UWAGA – ponieważ systemy na stanowiskach są odpalane w wirtualkach, nie zareagują prawidłowo na ręczne ustawienie adresów IP (podobnie jak w przypadku DHCP na poprzednich laborkach). Żeby wymusić użycie nowych adresów, trzeba wyłączyć i włączyć kartę sieciową, dotyczy to oczywiście i Windowsa, i Linuksa.

Kolejnym krokiem jest wyświetlenie konfiguracji urządzenia (*show run*) i zwrócenie uwagi na hasła, wszystkie powinny być podane jawnym tekstem. Konfigurujemy proste szyfrowanie haseł (Type 7):

```
(config)# service password-encryption
```

Przy okazji warto ustawić hasło do trybu *enable* – w instrukcji nigdzie to wspomniane nie jest, ale bez niego nie wejdziemy do trybu administracyjnego przez połączenie telnet chwilę później:

```
(config)# enable secret haslo_enable
```

Ponownie wyświetlamy konfigurację urządzenia i tym razem hasła nie powinny już być widoczne wprost – natomiast zastosowany do ich zaszyfrowania algorytm Type 7 nie ma na celu zapewnienia poufności (ma zabezpieczać tylko przed podejrzeniem), więc bardzo łatwo można je odszyfrować. I będziemy tym nawet zajmować się później ;-)

Teraz łączymy się z routerem przez telnet z jednego z komputerów i jednocześnie przechwytyjemy sesję za pomocą Wiresharka. Na routerze logujemy się i wykonujemy komendę *show run*, a na komputerze wyświetlamy przechwycone w Wiresharku pakiety (*Follow TCP Stream*), oczywiście przebieg całej komunikacji z routerem będziemy mogli odczytać.

Tworzymy na routerze nowe konto użytkownika zabezpieczone hasłem zabezpieczonym MD5, konfigurujemy dostęp przez SSH (musi być do tego ustawiona nazwa hosta) i wyłączamy dotychczasowy dostęp przez telnet:

```
(config)# username nazwa_konta secret haslo
(config)# ip domain-name bsk.local
(config)# crypto key generate rsa
(config)# line vty 0 4
(config-line)# no transport input
(config-line)# transport input ssh
(config-line)# login local
(config-line)# exit
(config)# ip ssh authentication-retries 3
```

Łączymy się znowu z routerem (tym razem przez SSH) i przechwytujemy pakiety w Wiresharku, teraz oczywiście nie uda nam się odczytać przebiegu sesji. Podobne ćwiczenie pojawiało się też na trzecich laborkach pod Linuxem, więc nie ma sensu rozpisywać się tutaj więcej na ten temat ;-)
Po wyświetleniu konfiguracji routera za pomocą *show run* będzie widać, że hasło utworzonego właśnie konta jest zapisane inaczej (przechowywany jest jego hash MD5).

Punkt 3.

Tutaj będziemy konfigurować na routerze protokół RIP. Zaczynamy od podłączenia drugiego interfejsu routera do wspólnej sieci (switch wskazany przez doktora Markowskiego) i skonfigurowania na nim adresu w sieci 10.10.15.0/24 – przykład dla portu ge0/0:

```
(config)# int g0/0
(config-if)# ip address 10.10.15.x 255.255.255.0
```

W miejsce x wpisujemy nasz numer grupy (koniecznie) – jest to bardzo istotne, bo na tej podstawie doktor sprawdza później, czy udało nam się wykonać ćwiczenie. Następnie konfigurujemy protokół RIP i dopisujemy do niego obie sieci (naszą 192.168.x.0 i wspólną 10.10.15.0):

```
(config)# router rip
(config-router)# version 2
(config-router)# network 192.168.1.0
(config-router)# 10.10.15.0
```

W oryginalnej instrukcji przy komendzie *network* po adresach sieci były podane jeszcze ich maski, ale na urządzeniach Cisco nie podaje się ich przy konfiguracji protokołu RIP – i komendy z instrukcji powodowały błąd.

Informacje o trasach na routerze możemy wyświetlić komendą:

```
# show ip route
```

W tym momencie nie będzie tam trasy do sieci 172.16.16.0, więc komunikacja z serwerem nie będzie oczywiście możliwa. Powinny natomiast pojawić się trasy do sieci innych grup, które też skonfigurowały już routing. Pakiety RIP wysyłane będą na wszystkie interfejsy, więc można przechwycić je i zobaczyć w Wiresharku na komputerze. Aby temu zapobiec, interfejs po stronie naszej sieci lokalnej należy ustawić jako pasywny:

```
(config)# router rip
(config-router)# passive-interface g0/1
```

W praktyce – generalnie zasada jest taka, że nie należy wysyłać pakietów z trasami tam, gdzie nie są potrzebne (czyli tam, gdzie nie ma żadnych wykorzystujących je routerów, np. do sieci ze zwykłymi komputerami), oprócz poprawy bezpieczeństwa zmniejsza to też niepotrzebne obciążenie łączy.

Na koniec tego punktu mamy skonfigurować bezpieczną wersję RIP z hasłem *bsk_lab5*:

```
(config)# key chain KLUCZ_RIP
(config-keychain)# key 1
(config-keychain)# key-string bsk_lab5
(config-keychain)# int g0/0
(config-if)# ip rip authentication mode md5
(config-if)# ip rip authentication key-chain KLUCZ_RIP
```

W tym momencie będzie już możliwa komunikacja z serwerem 172.16.16.200, wśród tras na routerze powinna pojawić się również trasa do jego sieci, powinny też być widoczne trasy do sieci innych grup, które skonfigurowały już bezpieczną wersję RIP. Jeżeli trasa do sieci 172.16.16.0 nie pojawiła się od razu, należy spróbować sprawdzić tablicę routingu jeszcze raz po pewnym czasie. Trasy w protokole RIP i innych protokołach routingu dynamicznego ogłaszane są co określony czas (rzędu mniej więcej minuty), więc brak tras nie musi od razu oznaczać, że skonfigurowaliśmy router nieprawidłowo.

Punkt 4.

Tu należy wykonać tylko skrypt zabezpieczający router – komenda:

```
# auto secure
```

I porównać porty otwarte przed i po jego wykonaniu – w moim przypadku niczego to nie zmieniło, ale też żadnych dziwnych portów wcześniej nie miałem otwartych (tylko SSH i telnet). Do wyświetlenia listy otwartych portów służy komenda:

```
# show control-plane host open-ports
```

Punkt 5.

Ostatnim zadaniem na zajęciach jest włamanie się na router *internetowy* (czyli zapewniający routing do sieci 172.16.200.0), wykorzystując konto użytkownika, o którym wiemy, że stosuje słabe hasła – generalnie takie samo hasło jak login i wszędzie to samo. Najpierw musimy ustalić adres IP routera – wyświetlamy tablicę routingu i szukamy routera obsługującego wspomnianą przed chwilą sieć:

```
# show ip route
```

Jak już mamy adres routera, musimy znaleźć login pechowego użytkownika. Jest on w tym momencie zalogowany na routerze, więc użyjemy do tego usługi *finger* podającej informacje o zalogowanych użytkownikach. Na naszym routerze wykonujemy polecenie:

```
> telnet <adres_ip_routera> finger
```

Zalogowanych może akurat być kilku użytkowników, ale bez problemu skojarzymy o którego tutaj chodzi. W tym momencie łączymy się z komputera za pomocą SSH ze zdalnym routerem i logujemy jako odkryty przed chwilą użytkownik, wchodzimy do trybu enable (pisałem wcześniej regułę co do haseł), wyświetlamy konfigurację (*show run*) i interesujemy się przede wszystkim

hasłami innych użytkowników. Instalujemy na komputerze program Cain & Abel i po kolei je łamiemy.

Większość haseł jest zaszyfrowana algorytmem Type 7, aby je odczytać wchodzimy w Cainie w *Tools > Cisco Type-7 Password Decoder* i sprawdzamy kolejne hasła. Jedno hasło ma postać hasha MD5, aby je złamać przechodzimy w Cainie na zakładkę *Cracker*, wybieramy *Cisco IOS-MD5 Hashes*, dodajemy hasło do listy (prawoklik > *Add to list*) i łamiemy metodą brutalną. Nie zajmie to dużo czasu, hasło jest bardzo „lipne” ;-).

Na koniec logujemy się na jedno z kont, do którego przed chwilą złamaliśmy hasło.

I to wszystko, powodzenia i wykonania wszystkich punktów!

Jan Potocki

Wrocław, 06.12.2018