HØGSKOLEN I BERGEN

TOD120
Nettverk og windows og sikkerhet og datamaskiner og servere og sånn

Øving 12

Deltakere

Joachim Tingvold

Rutingtabell m/ statisk ruting

```
hege#sh ip ro
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route

Gateway of last resort is not set

     10.0.0.0/23 is subnetted, 1 subnets
S       10.0.0.0 [1/0] via 192.168.2.1
C    192.168.2.0/24 is directly connected, Serial0
C    192.168.3.0/24 is directly connected, Ethernet0

anne#sh ip ro
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR

Gateway of last resort is not set

C    10.0.0.0/8 is directly connected, Ethernet0
C    192.168.2.0/24 is directly connected, Serial0
S    192.168.3.0/24 [1/0] via 192.168.2.2
```

Rutingtabell m/ RIP

```
hege#sh ip ro
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route

Gateway of last resort is not set

R    10.0.0.0/8 [120/1] via 192.168.2.1, 00:00:08, Serial0
C    192.168.2.0/24 is directly connected, Serial0
C    192.168.3.0/24 is directly connected, Ethernet0

anne#sh ip ro
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR

Gateway of last resort is not set

C    10.0.0.0/8 is directly connected, Ethernet0
C    192.168.2.0/24 is directly connected, Serial0
R    192.168.3.0/24 [120/1] via 192.168.2.2, 00:00:10, Serial0
```

Her ser vi en forskjell. Siden RIPv1 er classless, vil 10.0.0.0/25-nettet vises som 10.0.0.0/8 (siden dette nettet er class A). Bruker vi statisk ruting (eller en stateful rutingprotokoll), vil vi selvsagt kunne rute bare /25-nettet.

Trafikkfiltrering

Her er hovedregelen at når man bruker standard aksesslister, så skal ACLen settes i utgående retning på det interfacet som er nærmest den plassen man ønsker å hindre tilgang til. Dette skyldes at standard aksesslister **kun** kan inneholde source host/network.

I oppgaveteksten ønsker vi å hindre tilgang til 10.0.0.0/23 og 192.168.2.0/24 fra maskin B. Siden 10.0.0.0/23 ligger bak 192.168.2.0/24, klarer vi oss med én ACL. Hadde vi nå hatt et ekstra nett (utenom 10.0.0.0/23, f.eks. 10.1.0.0/24), så hadde ikke dette vært mulig ved å bruke standard aksessliste, siden disse bare baserer seg på source host/network.

Så, for å oppnå det vi ønsker, lager vi følgende standard ACL på Hege. Hvorfor Hege? Som nevnt må vi altså sette ACLen i utgående retning på det interfacet som er nærmest plassen man ønsker å hindre tilgang til. Siden vi vil hindre tilgang til 192.168.2.0/24, så må vi gjøre det på Hege. Hadde vi gjort det på Anne, så hadde maskin B fått tilgang til 192.168.2.0/24.

```
hege(config)#access-list 1 deny host 192.168.3.100
hege(config)#end
hege#sh access-lists
Standard IP access list 1
    deny   192.168.3.100
```

Videre kan vi nå aktivere denne ACLen i utgående retning på interfacet som er nærmest 192.168.2.0/24-nettet. Her er det viktig å passe på. Man ville kanskje tro at det «nærmeste» interfacet er Serial0, men det er feil. Hadde vi aktivert denne ACLen på Serial0, ville vi fortsatt kunne nådd Serial0s IP-adresse (192.168.2.2). Vi må altså aktivere ACLen i utgående retning på Eth0.

Men vent? Har vi husket på alt? Nei. ACLer har en implisitt «deny any» i slutten. Det vil da si at selv om vi har nektet tilgang fra maskin B, så vil alle andre også bli hindret tilgang. Dette må vi fikse.

```
hege(config)#access-list 1 permit any
hege(config)#end
hege#sh access-lists
Standard IP access list 1
    deny   192.168.3.100
    permit any
```

Dette har med at Ciscos IOS prosesserer ACLene top-down, systematisk. Hvis en pakke matcher et kriterie i en ACL, utføres handlingen, og prosesseringen av ACLen stopper. Dvs. at om en pakke fra en spesifik host blir nektet tilgang (deny 192.168.2.100), så forkastes pakken, selv om neste entry tillater all trafikk (permit any).

Vi er nå klar til å applye denne ACLen på Eth0.

```
hege(config)#int eth0
hege(config-if)#ip access-group 1 out
```

Voilá. Vi er ferdig.

Hvis vi nå ønsker å bruke utvidet aksessliste, så blir praksisen egentlig stikk motsatt fra den vi har ved bruk av standard aksessliste. Siden vi med utvidet aksessliste kan matche på port, protokoll, source og destination, ønsker vi å plassere slike ACLer nærmest mulig de nettene/maskinene vi ønsker å hindre tilgang **fra**. Dette er for å forhindre at uønsket trafikk bruker båndbredde. Siden oppgaveteksten ber om å lage én ACL på Anne, vil nevnte praksis ikke bli fulgt.

Vi ønsker at 10.0.0.0/23 skal få koble til via TCP mot 192.168.2.0/24 og 192.168.3.0/24, men ikke motsatt vei. Videre ønsker vi at alle maskiner, utenom 10.0.0.2, i 10.0.0.0/23-nettet skal kunne pinge 192.168.2.0/24 og 192.168.3.0/24. Alle maskiner i 192.168.2.0/24 og 192.168.3.0/24 skal få pinge alle maskiner i 10.0.0.0/23, utenom 10.0.0.2. Det skal heller ikke tillates UDP-trafikk.

Siden vi her skal angi både source og destination, så kan det fort bli endel regler. For å korte ned på disse, kan man ofte slå sammen flere nett ved å bruke wildcard. I vårt tilfelle kan vi slå sammen 192.168.2.0/24 og 192.168.3.0/24 til 192.168.2.0/23. I wildcard-format skrives dette som 0.0.1.255.

```
anne(config)#ip access ext LOLZ
anne(config-ext-nacl)#permit tcp 10.0.0.0 0.0.1.255 192.168.2.0 0.0.1.255
anne(config-ext-nacl)#deny tcp 192.168.2.0 0.0.1.255 10.0.0.0 0.0.1.255
anne(config-ext-nacl)#deny icmp host 10.0.0.2 192.168.2.0 0.0.1.255
anne(config-ext-nacl)#deny icmp 192.168.2.0 0.0.1.255 host 10.0.0.2
anne(config-ext-nacl)#permit icmp 10.0.0.0 0.0.1.255 192.168.2.0 0.0.1.255
anne(config-ext-nacl)#permit icmp 192.168.2.0 0.0.1.255 10.0.0.0 0.0.1.255
anne(config-ext-nacl)#deny udp any any
anne(config-ext-nacl)#end
anne#sh access-lists
Extended IP access list LOLZ
    permit tcp 10.0.0.0 0.0.1.255 192.168.2.0 0.0.1.255
    deny tcp 192.168.2.0 0.0.1.255 10.0.0.0 0.0.1.255
    deny icmp host 10.0.0.2 192.168.2.0 0.0.1.255
    deny icmp 192.168.2.0 0.0.1.255 host 10.0.0.2
    permit icmp 10.0.0.0 0.0.1.255 192.168.2.0 0.0.1.255
    permit icmp 192.168.2.0 0.0.1.255 10.0.0.0 0.0.1.255
    deny udp any any
```

Vi er nå ferdig med selve ACLen. Da gjenstår det bare å aktivere den på et interface. Siden vi har laget **én** ACL, må den aktiveres i begge retninger. Dette gjøres ved å bruke både «in» og «out» (som to kommandoer).

```
anne(config)#int eth 0
anne(config-if)#ip access-group LOLZ out
anne(config-if)#ip access-group LOLZ in
```

Notis 1: Teksten sier ingenting om hva som skal være tillatt i forhold til aksessering av ruterne. Ved ovennevnte ACL, så vil man måtte bruke 192.168.2.1 for å kunne få tilgang til Anne fra 10.0.0.0/23-nettet (dette skyldes at ingen av entryene i ACLen matcher «fra 10.0.0.0/23 til host 10.0.0.3», og blir derfor implisitt denyet i slutten av ACLen).

Notis 2: I teksten står det at UDP skal sperres. Her har jeg skrevet «deny udp any any», men denne kunne vi strengt bare droppet, siden ACLen har implisitt deny any i slutten. Dette er ikke tilfellet dersom tanken var at «alt annet som ikke er nevnt i oppgaven, skal være tillat» (altså «permit any any» i slutten). Dersom dette er tanken, må selvsagt «deny udp any any» være med i ACLen (før «permit any any», selvsagt).