PACKET FILTER FIREWALLS

Av Evan Saboo – saboo@kth.se

PACKET FILTER FIREWALLS

Av Evan Saboo – saboo@kth.se

Uppgift 0 – Setting Up Containers.

I förberedelse uppgiften satte jag upp containers och tilldelade olika IP adresser till dem. Den första jag satte upp var brandväggen med två nätverksenheter, där den första var ”10.0.10.1” vilket var kopplat till externa nätverket och ”10.0.20.1” till interna nätverket. Brandväggen var en gateway mellan interna och externa nätverken. De två noderna i interna nätet var inside-host med IP adress ”10.0.20.3” och attackerare med IP adressen ”10.0.20.3”. Den sista noden som sattes upp var outside-host med IP adress ”10.0.10.2”, vilket tillhörde externa nätverket.

Uppgift 1 – Building a Firewall.

1.1 Default Policy:

För att brandväggen ska blockera alla packet som skulle komma in, gå ut och dirigeras till des destination behövde man använda kommandon ”ufw default deny outgoing”, ”ufw default deny incoming” och ”ufw default deny routed”. Dessa kommandon gäller för alla paket förutom ICMP paket eftersom ufw kan bara hantera enkla konfigurationer medan iptables har full kontroll över brandväggen (även ICMP).

1.2 Network Premissions:

I andra uppgiften satte jag in två regler i brandväggen för att kunna:

- Tillåta all trafik från det interna nätverket gå till brandväggens interna enhet, med hjälp av kommandot ”sudo ufw allow in from 10.0.20.0/24 to 10.0.20.1”. Kommandot betyder att alla paket får komma in i brandväggen om ursprungsnoden tillhör en av noderna i interna subnätet.
- Tillåta all trafik från brandväggens interna enhet gå till alla noder i interna subnätet, med kommandot ”sudo ufw allow out from 10.0.20.1 to 10.0.20.0/24”. Kommandot betyder att alla paket får skickas till noderna i interna subnätet om ursprungsnoden är brandväggens interna IP adress.

Svar till fråga 2:

"deny" gör så att alla inkommande paket slängs bort utan att avsändaren vet om det. Fördelen med ”deny” är att en attackerare inte får någon information om vad som händer med alla skickade paket

och hur brandväggen är uppbyggd. Nackdelen är att en vanlig användare/avsändare vet inte om paketet hen skickade har nått fram till destinationsmålet.

"reject" gör samma sak som "drop" förutom att ett brandväggen skickar tillbaka ett "error" paket som säger att avsändarens paket inte lyckades komma fram till destinationsmålet. Fördelen är att en avsändare (t.ex. program) kan hantera misslyckat skickat paket då man får tillbaka felmeddelande. Nackdelen är att en attackerare kan få lite information om hur brandväggen fungerar då den vet att paketen inte kommer fram till destinationsmålet.

REJECT:
Jag testade med netcat för att kommunicera mellan brandväggen och inside-host. Resultatet blev att inside-host kopplades bort efter 1–2 sekunder från kommunikationen eftersom den fick i bakgrunden "error" paket från brandväggen om att kommunikationen misslyckades.

---

root@firewall:/home/student# ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), deny (outgoing), deny (routed)
New profiles: skip

To	Action	From
10.0.20.1	REJECT IN	10.0.20.2
10.0.20.1	ALLOW IN	10.0.20.0/24
10.0.20.2	REJECT OUT	10.0.20.1
10.0.20.0/24	ALLOW OUT	10.0.20.1

DROP:
Jag testade istället med att blockera kommunikationen mellan brandväggen och inside-host. När jag försökte koppla från inside-host till brandväggen troddes man att kopplingen lyckades men när försökte skicka paket till brandväggen så kom inget fram, men det såg ut som att inside-host hade fortfarande aktiv förbindelse med brandväggen.

---

root@firewall:/home/student# ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), deny (outgoing), deny (routed)
New profiles: skip

To	Action	From

10.0.20.1	DENY IN	10.0.20.2
10.0.20.1	ALLOW IN	10.0.20.0/24
10.0.20.2	DENY OUT	10.0.20.1
10.0.20.0/24	ALLOW OUT	10.0.20.1

1.3 Permitting a Service

För att tillåta en nod från externa nätverket koppla till brandväggen med hjälp av SSH lade jag till regeln ”sudo ufw allow in from any to 10.0.10.1 port 22 proto tcp”. Regeln betyder att alla noder får skicka packet till 10.0.10.1 med protokoll TCP genom port 22.

- Jag lyckades koppla från andra noder till brandväggen med kommandot ”sudo ssh student@10.0.10.1”.
- Jag prövade att koppla från outside-host till inside-host med kommandot ”sudo ssh student@10.0.20.2” men det gick inte.
- Jag lyckades koppla från externa noden till interna noden genom att först koppla till brandväggen och sedan koppla till inside-host via brandväggen.

Svar till fråga 3:
Fördelar:
SSH möjliggör kryptering av data så att de onda attackerna inte kan komma åt din användarinformation och lösenord. SSH möjliggör också tunnling av andra protokoll som FTP.

Nackdel:
Den största nackdelen är att personer kan logga in till datorns administratör över SSH. Detta kan göras enkelt med brute-force om lösenordet är enkelt.

---

root@firewall:/home/student# ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), deny (outgoing), deny (routed)
New profiles: skip

To	Action	From
--	------	----
10.0.10.1 22/tcp	ALLOW IN	Anywhere
10.0.20.1	ALLOW IN	10.0.20.0/24
10.0.20.0/24	ALLOW OUT	10.0.20.1

1.4 Stateful Filtering

För att tillåta interna nätverket koppla till externa nätverket lade jag till regeln "sudo ufw route allow from 10.0.20.0/24 to any". För att interna nätverket ska komma ut måste man tillåta brandväggen dirigera paket från interna subnätet till externa nätverket, vilket gör så att brandväggen ska bete sig som en gateway. Då default dirigering är "deny" och jag har inte specifikt lagt till någon regel som tillåter externa nätet skicka paket till interna nätet kommer dessa paket att alltid blockeras av brandväggen.

Svar till fråga 4:

Vi kan verifiera vår nya tillagda regel med hjälp av netcat:

TCP:
Outside-host (Server): nc -l 1024
Inside-host (Klient): nc 10.0.10.2 1024
== Kopplingen lyckas

Inside-host (Server): nc -l 1024
Outside-host (Klient): nc 10.0.20.2 1024
== Kopplingen misslyckas

UDP:
Outside-host (Server): nc -u -l 1024
Inside-host (Klient): nc -u 10.0.10.2 1024
== Kopplingen lyckas

Inside-host (Server): nc -u -l 1024
Outside-host (Klient): nc -u 10.0.20.2 1024
== Kopplingen misslyckas

ICMP:

Vi kan verifiera detta med ICMP pings men både outside-host och inside-host kommer kunna pinga varandra eftersom ufw har inte kontroll över ICMP regler.
För att blockera ICMP pings från outside till inside måste man lägga en regel som säger att alla icmp routade echo-requests till 10.0.20.0/24 ska blockeras. Detta kan göras med iptables eller lägga till regeln i filen "before.rules".
Iptables kommandot blir " iptables -I FORWARD -d 10.0.20.0/24 -p icmp --icmp-type echo-request -j DROP".

---
root@firewall:/home/student# ufw status verbose
Status: active

Logging: on (low)
Default: deny (incoming), deny (outgoing), deny (routed)
New profiles: skip

To	Action	From
--	------	----
10.0.10.1 22/tcp	ALLOW IN	Anywhere
10.0.20.1	ALLOW IN	10.0.20.0/24
10.0.20.0/24	ALLOW OUT	10.0.20.1
Anywhere	ALLOW FWD	10.0.20.0/24

1.5 Opening Ports

För att tillåta externa nätverket kunna koppla till interna nätverket genom port 9000 via TCP och UDP måste man lägga till två regler i brandväggen. Den första är ”sudo ufw route allow from any to 10.0.20.0/24 port 9000 proto tcp” och andra är ”sudo ufw route allow from any to 10.0.20.0/24 port 9000 proto udp”.

Svar till fråga 5:
Vi kan testa förbindelsen genom att använda netcat TCP ”nc -l 10.0.20.2 9000” eller UDP ”nc -u -l 10.0.20.2 9000” för att koppla från outside host till inside host, vilket lyckas. Om man testar med en annan port än 9000 kommer det misslyckas. För att testa att andra tjänster inte kan komma in kan vi testa SSH som använder port 22 och protokoll TCP från outside till inside, vilket misslyckas eftersom ”routed” är default deny. Andra tjänster brukar använda andra standardportar.

---
root@firewall:/home/student# ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), deny (outgoing), deny (routed)
New profiles: skip

To	Action	From
--	------	----
10.0.10.1 22/tcp	ALLOW IN	Anywhere
10.0.20.1	ALLOW IN	10.0.20.0/24
10.0.20.0/24	ALLOW OUT	10.0.20.1
Anywhere	ALLOW FWD	10.0.20.0/24
10.0.20.0/24 9000/udp	ALLOW FWD	Anywhere

10.0.20.0/24 9000/tcp ALLOW FWD Anywhere

1.6 Blocking Ports

För att blockera interna nätverket från att använda port 135 för att skicka paket till externa nätverket måste man lägga regeln ”sudo ufw route insert 1 reject from 10.0.20.0/24 to any port 135”. Jag blockerade bara dirigering via port 135 eftersom interna nätverket kan bara skicka paket till externa bara med ”routing”. Då behövde jag inte blockera utgående och inkommande paket i interna subnätet. Man måste vara försiktig i vilken position man ska lägga regeln eftersom när brandväggen kontrollerar ett paket med reglerna så börjar den med den första regeln i listan. Om paketet accepteras av första regeln kommer brandväggen sluta kolla igenom reglerna under och acceptera paketet. Därför lade jag min regel längst upp i regellistan, så att den ligger före regeln som tillåter alla kopplingar från interna till externa nätverket.

Svar till fråga 6:
Man kan använda netcat igen för att koppla från inside-host till outside-host. Man kan testa både via port 135 (vilket misslyckas) och andra portar, t.ex. 136, 777 eller 1337. Alla portar fungerar att använda netcat på förutom port 135.

---
root@firewall:/home/student# ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), deny (outgoing), deny (routed)
New profiles: skip

To	Action	From
--	------	----
10.0.10.1 22/tcp	ALLOW IN	Anywhere
10.0.20.1	ALLOW IN	10.0.20.0/24
10.0.20.0/24	ALLOW OUT	10.0.20.1
135	REJECT FWD	10.0.20.0/24
Anywhere	ALLOW FWD	10.0.20.0/24
10.0.20.0/24 9000/udp	ALLOW FWD	Anywhere
10.0.20.0/24 9000/tcp	ALLOW FWD	Anywhere

1.7 Blocking Ports

Jag använde tjänsterna netcat, ping och ssh för att kontrollera om alla regler följdes och allt annat blockerades.

Den slutgiltiga ufw regellistan blev:

---

root@firewall:/home/student# ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), deny (outgoing), deny (routed)
New profiles: skip

To	Action	From
--	------	----
10.0.10.1 22/tcp	ALLOW IN	Anywhere
10.0.20.1	ALLOW IN	10.0.20.0/24
10.0.20.0/24	ALLOW OUT	10.0.20.1
135	REJECT FWD	10.0.20.0/24
Anywhere	ALLOW FWD	10.0.20.0/24
10.0.20.0/24 9000/udp	ALLOW FWD	Anywhere
10.0.20.0/24 9000/tcp	ALLOW FWD	Anywhere

‘

Uppgift 2 – SSH Brute-force Attacks.

För att förhindra SSH brute-force eller DDoS attacker måste vi blockera alla externa noder som försöker koppla till brandväggen för många gånger. Det gör man med ufw kommandot "sudo ufw insert 1 limit from any to 10.0.10.1 port 22 proto tcp". Regeln säger att noden som försöker koppla (skicka paket) med TCP till 10.0.10.1 via port 22 för många gånger (6 eller mer) inom 30 sekunder kommer nekas.

Jag testade genom att försöka koppla till brandväggen från outside-host:

root@outside-host:/home/student# ssh student@10.0.10.1
student@10.0.10.1's password:
Permission denied, please try again.
student@10.0.10.1's password:
Permission denied, please try again.
student@10.0.10.1's password:
Permission denied (publickey,password).
root@outside-host:/home/student# ssh student@10.0.10.1
student@10.0.10.1's password:
Permission denied, please try again.
student@10.0.10.1's password:
Permission denied, please try again.
student@10.0.10.1's password:
Permission denied (publickey,password).
root@outside-host:/home/student# ssh student@10.0.10.1
student@10.0.10.1's password:
Permission denied, please try again.
student@10.0.10.1's password:
Permission denied, please try again.
student@10.0.10.1's password:
Permission denied (publickey,password).
root@outside-host:/home/student# ssh student@10.0.10.1
student@10.0.10.1's password:
Permission denied, please try again.
student@10.0.10.1's password:
Permission denied, please try again.
student@10.0.10.1's password:
Permission denied (publickey,password).
root@outside-host:/home/student# ssh student@10.0.10.1
student@10.0.10.1's password:
Permission denied, please try again.

student@10.0.10.1's password:
Permission denied, please try again.
student@10.0.10.1's password:
Permission denied (publickey,password).
root@outside-host:/home/student# ssh student@10.0.10.1
ssh: connect to host 10.0.10.1 port 22: Connection refused
root@outside-host:/home/student# ssh student@10.0.10.1
ssh: connect to host 10.0.10.1 port 22: Connection refused
root@outside-host:/home/student#

Man kan se att SSH tillåter 3 lösenordsförsök under varje koppling. Efter 5 kopplingsförsök blev outside-host nekad av brandväggen.

Uppgift 3 – Ping and ICMP.

Den sista uppgiften handlar om blockering av Internet Control Message Protocol och hur man går tillvägas för att skydda sig mot ping attacker.
Eftersom ufw inte kan hantera ICMP måste man använda iptables istället.

När man pingar från outside-host till inside-host ser man i outside terminalen varje sekund kommer en ny rad med info vart echo-reply kom från, hur stor paketet är, vilken icmp sekvensnummer den har, hur lång tid det tog att utföra en ping, och ttl (time to live). I wireshark ser man varje sekund en echo request från 10.0.20.2 till 10.0.10.2 och en echo reply tillbaka från 10.0.10.2 till 10.0.20.2.

Blocking ICMP Echo Requests

För att blockera ICMP echo requests från externa till interna nätverket använde jag iptables kommandot ”sudo iptables -I FORWARD -d 10.0.20.0/24 -p icmp --icmp-type echo-request -j DROP”. Jag behövde bara blockera icmp echo requests som routades till interna subnätet eftersom alla externa packet måste dirigeras via brandväggen för att komma till interna nätverket.
Eftersom vi använde ”DROP” så blockeras alla externa paket utan att brandväggen skickar någon error paket tillbaka till avsändaren:

---

root@outside-host:/home/student# ping 10.0.20.2
PING 10.0.20.2 (10.0.20.2) 56(84) bytes of data.
^C
--- 10.0.20.2 ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4017ms

---

Man såg också i Wireshark att ingen echo request eller reply visades eftersom alla paket dumpades före de gick igenom brandväggen där Wireshark lyssnade från.

Rejecting ICMP Echo Requests

Istället för att blockera externa icmp request paket använde jag kommandot “iptables -I FORWARD -d 10.0.20.0/24 -p icmp --icmp-type echo-request -j REJECT” för att avvisa dem. Den ända skillnaden mellan REJECT och DROP är att REJECT skickar nu tillbaka error paket till avsändaren efter varje avvisat paket.

---

root@outside-host:/home/student# ping 10.0.20.2
PING 10.0.20.2 (10.0.20.2) 56(84) bytes of data.
From 10.0.10.1 icmp_seq=1 Destination Port Unreachable
From 10.0.10.1 icmp_seq=2 Destination Port Unreachable
From 10.0.10.1 icmp_seq=3 Destination Port Unreachable
From 10.0.10.1 icmp_seq=4 Destination Port Unreachable

^C
--- 10.0.20.2 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 2997ms
---

Man kan se att outside host får tillbaka error paket från brandväggen, vilket säger att destinationsadressen går inte att komma åt.
Man såg heller ingen echo request och reply i Wireshark, likadant som ”DROP” testet.

Jag har verifierat att:

- Jag kan pinga från inside-host till outside-host.
- Outside-host kan inte pinga inside-host
- Det går att ping inside-host från brandväggen men av någon konstig anledning hade default reglerna i ufw blockerat icmp pings från brandväggen till outside-host. Min kompis hade samma problem och vi försökte också testa med att acceptera ufw default outgoing, vilket gjorde så att ping fungerade. Vi kom fram till slut att man var tvungen lägga till en regel med iptables som tillät brandväggen pinga till alla noder i externa nätverket. Kommandot var ”sudo iptables -I OUTPUT -s 10.0.10.1 -p icmp --icmp-type echo-request -j ACCEPT”.

Svar till fråga 9:

Min brandvägg har ingen iptables regel som säger att jag inte kan pinga till brandväggens interna IP adress. Ja, detta kan påverka säkerhetsimplikationer. En attackerare kan DDoS:a brandväggen med hjälp av ping flooding, d.vs. flera externa noder skickar icmp echo requests till brandväggen vilket leder till överbelastning. Som attackerare kan man också skicka farliga ping paket som är större än vad brandväggen ”ska kunna hantera”, då det leder till buffer overflow vilket kan göra så att systemet kraschar eller farlig kod kommer in i systemet. Den tredje attacken som kan ske är Smurf attack, där ett spoofat icmp ping request skickas till flera noder med brandväggens IP adress som avsändaradress. När noderna får ping förfrågan skickar de tillbaka ping svar som går till brandväggen, vilket leder till att för många echo-reply tas emot av brandväggen och den blir överbelastad. I alla dessa fall kan det hända att brandväggen går ner vilket leder till oskyddad internt subnät.

Blocking ICMP Echo Replies

För att attackeraren ska spoof-pinga outside-host med inside-hosts IP adress som source adress använde jag netwox kommandot ”netwox 50 -I 10.0.20.2 -i 10.0.10.2 -E 00:16:3e:84:31:c3 -e 00:16:3e:f9:02:01”. Netwox 50 skickar icmp packet (även spoofade) konstant med -I <source IP adress>, -i <destination IP adress>, -E <source MAC adress> och -e <destination MAC adress>. Source MAC adressen är inside-hosts MAC adress för att outside-host ska kunna ksicka tillbaka echo-reply via brandväggen. Destinations MAC adressen tillhör brandväggen eftersom man måste först skicka alla paket till gateway för att de ska dirigeras till outside-host.

I Wireshark såg man att varje echo-request paket skickades till outside-host och noden skickade tillbaka icmp echo-reply till inside-host (10.0.20.2). Det konstiga var att även om destinationsadressen var inside-host så skickades reply paketen ändå tillbaka till attackeraren. Allt såg rätt ut i wireshark och i brandväggen (undersöktes med kommandot ”tcpdump”) men paketen skickades ändå till attackeraren. Efter lite undersökning och hjälp från assistenten insåg man att det var VM:s switch (som kopplar ihop inside-host, attackerare och brandväggen) som skickade tillbaka paketen till fel destinationsadress eftersom den visste att det var attackeraren som skickade ping förfrågorna.
Det var svårt att komma fram till en lösning eftersom vi inte hade mycket tid på oss till deadline.

För att blockera icmp echo svar från outside-host använde jag iptables kommandot ”sudo iptables -I FORWARD -d 10.0.20.2 -p icmp --icmp-type echo-reply -j DROP”. Regeln säger att alla icmp echo-reply packet som vidarebefordras till inside-host ska blockeras. Då vi täcker bara scenariot ovan och ingen annan så kommer inside-host inte kunna få tillbaka echo svar vid andra scenarier.

Alla regler jag har just nu i brandväggen:

1. Iptables – Alla icmp echo-request paket som ska dirigeras till subnätet 10.0.20.0/24 ska avisas.
2. Iptables – Alla icmp echo-reply paket som ska dirigeras till IP adressen 10.0.20.2 ska blockeras.
3. Iptables – Alla utgående icmp echo-request paket från IP adressen 10.0.10.1 ska accepteras
4. UFW – Antalet anslutningar från externa nätverket till 10.0.10.1 via port 22 ska begränsas inom vis tid.
5. UFW – Alla paket skickat från subnätet 10.0.20.0/24 vår inte dirigeras via port 135.
6. UFW – Tillåt paket skickas ut från brandväggens IP adress ”10.0.20.1” till interna nätverket.
7. UFW – Tillåt paket från subnätet 10.0.20.0/24 gå till 10.0.20.1, dvs paketen får komma in i brandväggen.
8. UFW - Alla paket som kommer från subnätet 10.0.20.0/24 får vidarebefordras till vartsomhelst.
9. UFW – TCP Paket som ska skickas via port 9000 till en nod i subnätet 10.0.20.0/24 får dirigeras via brandväggen.
10. UFW – UDP Paket som ska skickas via port 9000 till en nod i subnätet 10.0.20.0/24 får dirigeras via brandväggen.
11. UFW – Default blockering av inkommande, utgående och dirigerade paket.

Limits

I den allra sista deluppgiften ska man begränsa antalet ICMP echo-reply paket per sekund, när de dirigeras till inside-host.
Eftersom brandväggen blockerar alla icmp echo-reply paket till inside-host så är det bara lägga en till regel som tillåter max antal paket dirigeras per sekund. Jag la till regeln ”iptables -I FORWARD -p icmp -d 10.0.20.2 --icmp-type echo-reply -m limit --limit 3/second -j ACCEPT” i iptables, vilket betyder att bara 3 paket får dirigeras till inside-host per sekund. Jag valde 3/sekund eftersom mottagande av 3 paket skulle inte resultera i DDoS attack. Jag kunde också ha valt 5 paket per sekund, vilket är fortfarande inom säkerhetsmarginalet. Jag testade regeln genom att skicka 5 spoofade icmp paket per sekund (med netwox 50 flaggan ”-m 200”) till

outside-host , som sedan skickade tillbaka echo-reply till inside-host. Man såg i Wireshark att från början fick varje echo-request en echo-reply men efter 11 echo-requests började brandväggen blockera var tredje echo-reply paket.
Jag testade också att pinga outside-host från inside-host samtidigt som attckeraren spoofpinga samma nod, då båda pingade varje halv sekund. I Wireshark såg man att var fjärde echo-reply till inside-host blockerades.

Hur skulle begränsning av echo-replies motstå DDoS attacker?
Om offret får begränsat antal paket sekund kommer den kunna hantera dem utan det blir överbelastning.