AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE
Wydział Fizyki i Informatyki Stosowanej

Praca inżynierska

Marcin Fabrykowski

kierunek studiów: **informatyka stosowana**

System IPS oparty o iptables

Opiekun: **dr inż. Krzysztof Rzecki**

Kraków, styczeń 2013

Oświadczam, świadomy(-a) odpowiedzialności karnej za poświadczenie nieprawdy, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

.....
(czytelny podpis)

Na kolejnych dwóch stronach proszę dołączyć kolejno recenzje pracy popełnione przez Opiekuna oraz Recenzenta (wydrukowane z systemu MISIO i podpisane przez odpowiednio Opiekuna i Recenzenta pracy). Papierową wersję pracy (zawierającą podpisane recenzje) proszę złożyć w dziekanacie celem rejestracji.

Spis treści

Wstęp

W dobie coraz większego przenikania internetu do życia codziennego, przestępstwa komputerowe stają się coraz bardziej powszechne. Wykradane informacje stanowią taką samą wartość, a nawet większą, jak fizycznie ukradzione przedmioty. Ponadto odcięcie dostępu do portalu powoduje znaczne straty dla firmy oraz możliwość przejęcia części rynku przez konkurencję. Dlatego tak ważnym tematem jest obrona przez atakami sieciowymi.
W swojej pracy omówię podstawowe ataki odmowy dostępu oraz skanowania hosta, które może być pierwszym krokiem podczas próby ataku, a następnie postaram się przedstawić sposoby obrony przed nimi. Postaram się przedstawić metody ataku, możliwe konsekwencje z nick wynikające oraz sposoby obrony przed nimi.

Jako system operacyjny dla opracowywanego systemu IPS może posłużyć dowolna dystrybucja GNU/Linux. Systemy te dają dużą możliwość ingerencji i filtrowania ruchu sieciowego przy użyciu narzędzi Netfilter i Iptables. Używane narzędzie zostaną dokładniej opisane w rozdziałach 3 oraz 4.

Rozdział 5 poświęcony zostanie poświęcony atakom wykrywanym przez projektowany system IPS. Przedstawiony zostanie opis każdego ataku oraz możliwe konsekwencje każdego ataku. Następnie opisana metodologia obrony oraz przedstawienie konkretnego rozwiązania implementacyjnego.

Natomiast ostatni rozdział zostanie poświęcony metodzie zbierania informacji oraz zapisywania wykrytych zdarzeń do bazy danych.

Rozdział 1

O IPS

1.1 Co to jest IPS

System IPS (ang. Intrusion Prevention System) jest to system wykrywania i blokowania ataków sieciowych. Jego zadanie polega na analizie ruchu sieciowego wchodzącego do oraz przechodzącego przez niego oraz odpowiednie reagowanie w przypadku wykrycia nienormalnych zachowań sieci.

1.2 Schemat działania

Opracowany system IPS działa według poniższego schematu działania:

1. Analiza ruchu sieciowego
2. Wykrycie zachowań pasujących do zdefiniowanych reguł bezpieczeństwa
3. Reakcja na wykryte niebezpieczne zachowanie
4. Poinformowanie administratora o próbie ataku oraz podjętych działaniach
5. Zapisanie danych o ataku oraz podjętych działaniach do bazy danych
6. Udostępnienie administratorowi wglądu w historię ataków

Poszczególne etapy schematu zostały opisane w kolejnych rozdziałach.

Rozdział 2

Protokoły

2.1 IP

Protokół internetowy (ang. *Internet Protocol*), jest protokołem działającym w trzeciej warstwie sieciowej modelu OSI - warstwie sieciowej. Odpowiedzialny jest za dostarczanie pakietów do hostów. Hosty używające protokołu IP są adresowane przy użyciu adresów IP.

Aktualnie najpowszechniejszą wersją protokołu IP jest wersja IPv4, w której używany jest 32 bitowy adres. Drugą wersją, która zaczyna wypierać IPv4 jest wersja IPv6, w której adresy mają długość 128 bitów.

Jeżeli wysyłane dane w pakiecie są większe niż MTU (*MAximum Transfer Unit*), pakiet zostaje pofragmentowany, czyli podzielony na mniejsze porcje danych, tak aby ich rozmiar wraz z nagłówkiem były mniejsze niż najmniejsze MTU na trasie na której będzie przesyłany pakiet. W przypadku fragmentaryzacji pakietu, zostają ustawione odpowiednie flagi w pakiecie - szczegóły poniżej.

2.1.1 Nagłówek IP

Każdy pakiet IP zostaje opakowany w ramkę IP. Schemat takiej ramki został pokazany na rys. 2.1. Znaczenie poszczenie poszczególnych pól jest następujące:

Version, 4bit

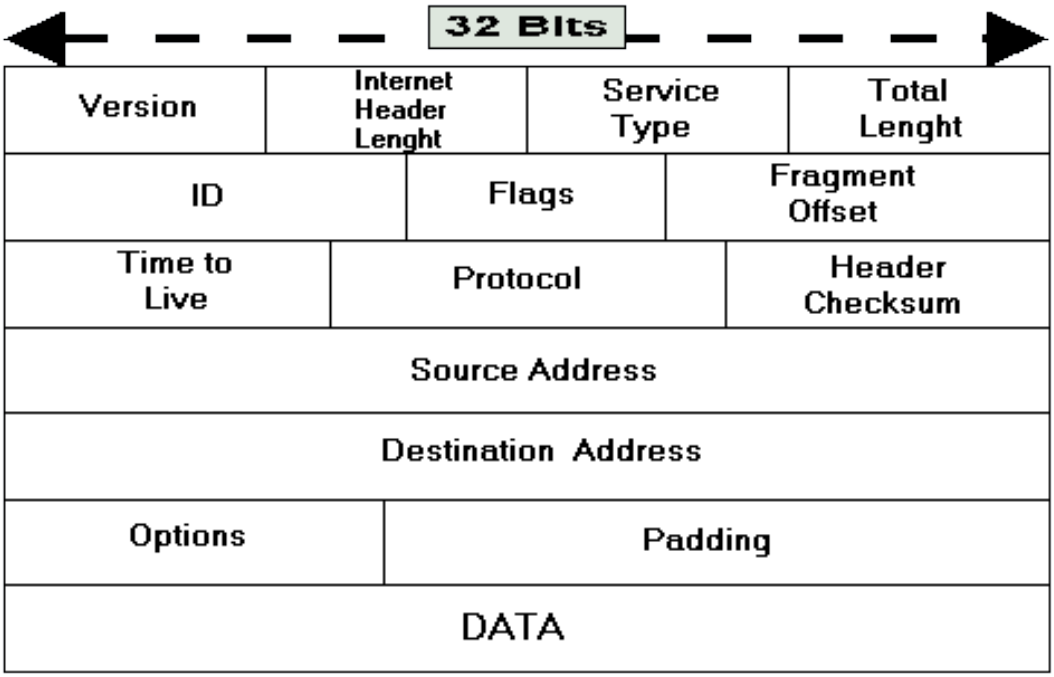
Determinuje która wersja protokołu IP będzie używana.

Internet Header Length, 4bit

Przechowuje informację o długości nagłówka IP. Wartość ta jest zawsze wielokrotnością 32 bitów. Aby nagłówek mógł być wielokrotnością 32 bitów, dodawana jest wartość pusta na końcu nagłówka, aby dopełnić jego długość do wielokrotności 32 bitów.

Service Type, 8bit

Przetrzymuje informacje o typie przesyłanych danych. Wartość ta była dawniej używa-



Rysunek 2.1: Ramka pakietu IP, źródło: [4]

na do profilowania ruchu (*QoS*), jednak aktualnie jest ona najczęściej ignorowana przez routery.

Total Length, 16bit

Zawiera długość całego pakietu - nagłówka oraz danych.

ID, 16bit

Numer identyfikacyjny pakietu.

Flags, 3bit

Flagi pakietu.

- bit 0 - zarezerwowany
- bit 1 - DF - *Don't fragment*, określa czy pakiet może być fragmentowany
- bit 2 - MF - *More fragments*, określa czy będą następne fragmenty, czy ten jest ostatni

Fragment Offset, 13bit

Określa w miejsce w datagramie w którym znajduje się dany fragment

Time to live, 8bit

Przechowuje wartość TTL, określającą długość życia pakietu. Wartość ta jest mierzona w ilości ruterów jakie może minąć pakiet, ponieważ na każdym ruterze wartość ta jest zmniejszana o 1. Jeżeli wartość spadnie do 0, router kasuje pakiet i odsyła do nadawcy informacje o przekroczeniu czasu życia pakietu.

Protocol, 8bit

Określa jaki protokół transportowy jest wykorzystywany przy połączeniu. Najpopularniejszymi protokołami są:

- 0x04 - enkapsulacja IPv4
- 0x06 - TCP
- 0x01 - ICMP
- 0x11 - UDP

Header Checksum, 16bit

Suma kontrolna nagłówka. Pozwala ona na kontrolowanie, czy nagłówek nie uległ zmianie przy przesyłaniu. Wartość ta jest sprawdzana i przeliczana na każdym routerze, ponieważ wartość TTL zostaje zmniejszana.

Source adres, 32bit

Zawiera adres źródłowy pakietu.

Destination address, 32bit

Zawiera adres docelowy pakietu.

Options

Zestaw różnych opcji pakietów IP. Ich opis jest poza zakresem tej pracy.

Padding

Dopełnienie nagłówka do pełnej wielokrotności 32 bitów. Wynika to z faktu, że wartość Internet Header Length zawiera długość nagłówkach w 32 bitowych słowach. Aby wartość ta miała prawo bytu, nagłówek musi być długości wielokrotności słowa.

2.2 TCP

TCP (ang. *Transmission Control Protocol*) jest połączeniowym protokołem transportowym. Odpowiada on za nawiązywanie połączenia oraz jego kontrolowanie.

Protokół TCP często jest protokołem transportowym dla IP. Razem tworzą najpowszechniejszą parę TCP/IP. Protokół IP używany jest jedynie do określania nadawcy pakietu, natomiast określanie usług docelowych leży po stronie TCP.

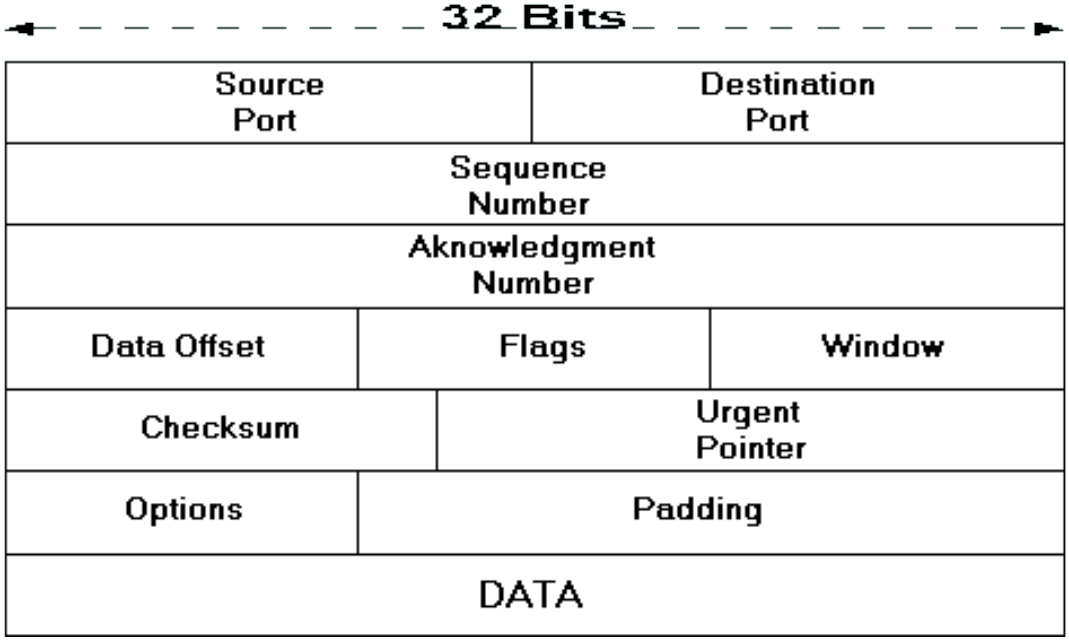
Dodatkowo, protokół TCP oddziela różne równoległe strumienie danych pomiędzy dwa hostami. Wprowadza on, obok adresów IP źródłowych i docelowych z protokołu IP, porty - źródłowy i docelowy. Używając kwartetu parametrów, TCP jest wstanie jednoznacznie sprecyzować, do którego połączenia należy pakiet.

Połączenie zwykle nawiązywane jest z losowego portu źródłowego (zwykle numer portu źródłowego jest większy niż 1024) na znany port docelowy. Numery portów poniżej 1024 są zarezerwowane dla tzw. *Well known ports* czyli usług o zwyczajowo przypisanych stałych portach. Najważniejszymi dobrze znanymi portami są:

- 80 - WWW
- 110 - POP3
- 25 - SMTP
- 22 - SSH
- 21 - FTP
- 20 - FTP-DATA
- 23 - TELNET

2.2.1 Nagłówek TCP

Każdy pakiet TCP zostaje opakowany w ramkę TCP. Schemat takiej ramki został pokazany na rys. 2.2. Znaczenie poszczenie poszczególnych pól jest następujące:



Rysunek 2.2: Ramka pakietu TCP, źródło: [4]

Source Port, 16bit

Numer portu źródłowego.

Destination Port, 16bit

Numer portu docelowego.

Sequence Number, 32bit

Numer sekwencyjny pakietu.

Acknowledgment Number, 32bit

Numer sekwencyjny odpowiedzi.

Data Offset, 4bit

Odległość od początku pakietu w którym zaczynają się dane. Wartość wyrażona w 32 bitowych słowach.

Flags, 9bit

Zawiera flagi cechujące dany pakiet TCP. Najważniejsze używane flagi to:

- SYN - oznacza pakiety przeprowadzające synchronizację numerów sekwencyjnych
- ACK - oznacza pakiety potwierdzające otrzymanie pakietów
- FIN - oznacza pakiet żądanie zakończenia połączenia
- RST - oznacza natychmiastowe resetowanie połączenia.

Window, 16bit

Rozmiar okna. Mechanizm okna pozwala na kontrolowanie przepływu danych oraz przyśpieszenie wysyłania danych.

Checksum, 16bit

Suma kontrolna nagłówka. Pozwala sprawdzić czy nagłówek nie uległ modyfikacji podczas przesyłania.

Options

Inne dodatkowe parametry cechujące pakiet

Padding

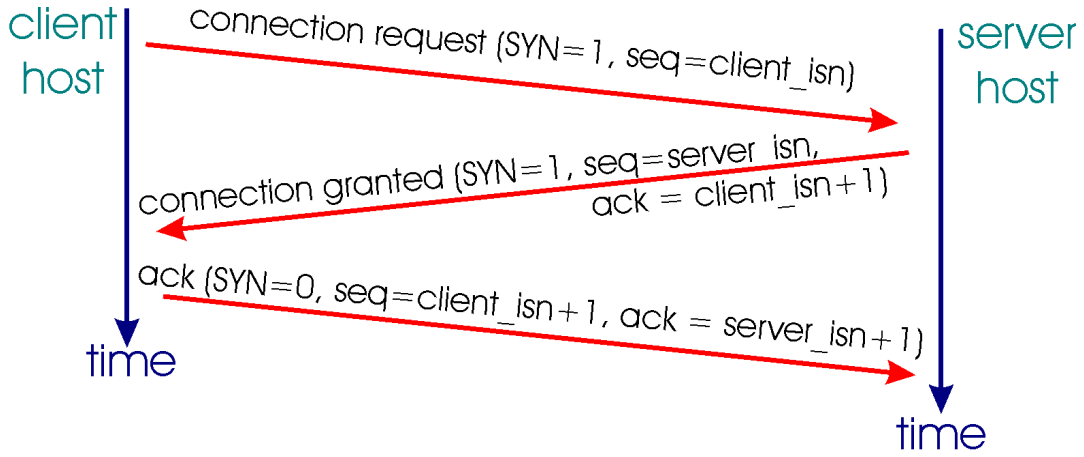
Dopełnienie nagłówka zerami do rozmiaru będącego wielokrotnością słowa.

2.2.2 Nawiązywanie połączenia

Aby nawiązać połączenie przy użyciu protokołu TCP, należy przeprowadzić procedurę połączeniową, zwaną *3 way handshake*, ponieważ, aby nawiązać połączenie, potrzeba wymienić trzy pakiety.

Początkowo, host inicjujący połączenie, wysyła pakiet z ustawioną flagą SYN, oraz specjalnie wygenerowaną losową wartością pola SEQ. Host odbierający pakiet SYN, odpowiada pakietem

z ustawionymi flagami SYN oraz ACK, oraz specjalnie wygenerowaną wartością losową SEQ, natomiast wartość pola ACK zawiera wartość SEQ z pakietu SYN powiększoną o 1. W odpowiedzi host inicjujący połączenie wysyła pakiet z flagą ACK oraz wartościami SEQ równej pierwszej wartości SEQ powiększonej o jeden, oraz ACK zawierająca wartość SEQ z pakietu SYN+ACK powiększoną o 1.
Po wymienieniu tych trzech pakietów, połączenie zostaje nawiązane. Powyższy schemat wymiany został przedstawiony na rys. 2.3.



Rysunek 2.3: Nawiązywanie połączenia TCP, źródło: [5]

2.2.3 Kontrola transmisji

Protokół TCP kontroluje ciągłość transmisji poprzez numery sekwencyjne. Po nawiązaniu połączenia, każda ze stron połączenia ustaliła swoje numery sekwencyjne którymi będzie się posługiwać.

Wysyłając pakiet o numerze sekwencyjnym 10, host odbierający wysyła pakiet ACK, z ustawionym numerem ACK równym numerowi sekwencyjnemu pakietu opowiadającemu temu potwierdzeniu. Host wysyłający, po otrzymaniu odpowiedzi ACK z numerem sekwencyjnym pakietu który wysłał, uznaje ten pakiet za dostarczony.
W przypadku nie otrzymania odpowiedzi pewnym okresie czasu, uznaje się, że pakiet nie dotarł do adresata i następuje jego ponowne wysłanie.

Przy wysyłaniu kolejnych pakietów, numery sekwencyjne zwiększane są o długość poprzedniego pakietu. Dla przykładu:
Wysyłając pakiet o numerze sekwencyjnym 1000, oraz długości 100, otrzymamy odpowiedź z numerem ack równym 1000. Natomiast następny wysyłany pakiet będzie miał numer sekwencyjny 1100.

2.2.4 Mechanizm okna

Mechanizm przesuwnego okna pozwala na dopasowanie optymalnej prędkości przesyłu danych, przy zachowaniu ciągłości transmisji.

Gdyby host wysyłający, po wysłaniu pakietu oczekiwał na potwierdzenie jego odebrania przez host docelowy, a następnie wysyłał kolejną porcję danych, otrzymalibyśmy w pełni stabilne połączenie, jednak odstępy pomiędzy wysyłaniem kolejnych pakietów były znaczne, ponieważ równe by były czasowi dotarcia pakietu do hosta docelowego oraz czasowi powrotu potwierdzenia.

Aby tego uniknąć, host wysyłający mógłby wysyłać pakiety w serii, bez oczekiwania na potwierdzenia. Jednak, jeżeli host odbierający charakteryzuje się mniejszą mocą obliczeniową niż host wysyłający, nie będzie w stanie obsłużyć wszystkich dochodzących pakietów. Spowoduje to utratę tych pakietów i potrzeba ich retransmisji. Prowadzić to będzie do wielokrotnego wysyłania tych samych danych, co z efekcie spowoduje spadek prędkości połączenia.

Używając mechanizmu okna, host wysyła pakiety w serii, tylko w rozmiarze nieprzekraczającym rozmiaru okna. Następnie zaprzestaje wysyłania pakietów, do chwili otrzymania potwierdzenia otrzymania pierwszego pakietu. Wtedy następuje przesunięcie okna o jedną pozycję i wysłanie kolejnego pakietu.
Host docelowy odpowiada na odbierane pakiety, pakietami z wartościami ack równymi wartościom seq. W przypadku gdy host docelowy dostaje pakiet, którego parametr seq, jest większy niż wartość seq poprzedniego pakietu zsumowana z długością ostatniego pakietu, uznaje się, że w transmisji został zgubiony pakiet. Wtedy, host docelowy odpowiada wartością ack ostatniego poprawnego pakietu. Jeżeli zgubionym pakietem jest pierwszy pakiet w oknie nadawcy, okno nie zostaje przesunięte. W przypadku gdy host docelowy dostaje pakiet, którego parametr seq, jest większy niż wartość seq poprzedniego pakietu zsumowana z długością ostatniego pakietu, uznaje się, że w transmisji został zgubiony pakiet. Wtedy, host docelowy odpowiada wartością ack ostatniego poprawnego pakietu. Jeżeli zgubionym pakietem jest pierwszy pakiet w oknie nadawcy, okno nie zostaje przesunięte. Do hosta docelowego dostarczane są kolejne wysłane pakiety, na które odpowiedzią są pakiety z ack ostatniego poprawnego pakietu. Gdy nastąpi retransmisja zgubionego pakietu i retransmitowany pakiet zostanie dostarczony do hosta docelowego, ten odpowiada pakietem o ack ostatniego pakietu w serii. Dla przykładu:
wielkość okna: 800

- Host A → Host B: seq 1000, len 100
- Host A → Host B: seq 1100, len 100 (zagubiony)
- Host A → Host B: seq 1200, len 200
- Host A → Host B: seq 1400, len 100
- Host A → Host B: seq 1500, len 200
- Host A → Host B: seq 1700, len 100
- Host A ← Host B: ack 1000 (odpowiedź na seq 1000)

- Host A → Host B: seq 1800, len 100
- Host A ← Host B: ack 1000 (odpowiedź na seq 1200)
- Host A → Host B: seq 1100, len 100 (retransmisja)
- Host A ← Host B: ack 1000 (odpowiedź na seq 1400)
- Host A ← Host B: ack 1500 (odpowiedź na seq 1500)
- Host A → Host B: seq 1900, len 200
- Host A → Host B: seq 2100, len 100

Rozmiar okna jest ustalany dynamicznie w zależności o parametrów sieci i aktualnego obciążenia. Algorytmy ustalania jego wielkości są poza zakresem tej pracy.

2.2.5 Kończenie połączenia

Aby zakończyć połączenie, host wysyła pakiet z ustawioną flagą FIN oraz numerem sekwencyjnym równym SEQ. Host docelowy odpowiada pakietem ACK z wartością ack równą SEQ+1 otrzymanego pakietu. Następnie również wysyła pakiet FIN do hosta kończącego połączenie. Host kończący odpowiada pakietem ACK w wartością ack równą SEQ+1 pakietu FIN. Po wysłaniu ACK, host kończący uznaje połączenie za zakończone, natomiast host docelowy uznaje połączenie za zakończone po otrzymaniu ACK.

2.2.6 Nieoczekiwane zachowania

Protokół TCP reaguje na nieoczekiwane zachowania wysyłaniem pakietu z ustawioną flagą RST. Takimi nieoczekiwanymi sytuacjami mogą być:

- Wysłanie pakietu SYN na port na którym nie słucha żadna usługa.
  Jeżeli TCP otrzymuje pakiet SYN na porcie na którym nie słucha żadna usługa, uznaje, że taki pakiet nie jest oczekiwanym pakietem i odpowiada hostowi wysyłającemu pakietem z ustawionymi flagami RST i ACK. Jest to informacja że na tym porcie nie są oczekiwane połączenia.

- Wysłanie pakietu ACK.
  Jeżeli host otrzymuje nieoczekiwany pakiet ACK, wygenerowana zostaje informacja do hosta nadającego o zaprzestanie transmisji. Informacja taka jest przekazywana poprzez pakiet z ustawioną flagą RST.

Rozdział 3

Analiza ruchu sieciowego

3.1 Narzędzie Netfilter

Jako systemu analizującego ruch sieciowy wykorzystam pakiet Netfilter konfigurowany za pomocą iptables.

3.2 Ogólny zarys

Netfilter jest oprogramowaniem pozwalającym na filtrowanie pakietów, ich translacje (NAT) oraz inne manipulację. Od wersji jądra 2.4.x, pakiet netfilter jest umieszczony wewnątrz niego. Potrafi on dopasowywać analizowane pakiety ze względu na szeroką gamę kryteriów, jak również przeprowadzić szereg operacji na danych pakietach.

3.3 Zasada działania

Netfilter posiada 4 zdefiniowane tablice: raw, mangle, nat, filter, oraz 5 łańcuchów: PREROUTING, INPUT, FORWARD, OUTPUT, POSTROUTING. Kolejność przechodzenia pakietu przez tabele i łańcuchy obrazuje rys. 3.1.
Gdy pakiet dochodzi do komputera na którym jest netfilter, zostaje on przekazany do łańcucha PREROUTING. Następnie, następuje decyzja o routowaniu pakietu, i w zależności czy pakiet jest kierowany do lokalnego komputera, jest kierowany do łańcucha INPUT, bądź w przypadku gdy ma zostać przekazany dalej, do łańcucha FORWARD oraz POSTROUTING.
W przypadku, gdy pakiet jest generowany przez nasz komputer, zostaje on przejęty przez łańcuch OUTPUT, a następnie POSTROUTING.

Pierwszą tablicą przez którą przechodzi pakiet, jest tablica *raw*. W tym miejscu możemy oznaczyć pakiet targetem TRACE, który spowoduje logowanie każdej reguły do której pakiet będzie pasował. Wykorzystywane jest to przy debugowaniu sieci bądź firewalla. Inna opcją która może pojawić się jedynie w tablicy raw, jest target NOTRACK. Spowoduje on nieozna-

Rysunek 3.1: Przepływ pakietów w netfilter, źrodło [6]

czenie pakietu przez moduł *conntrack*, który rejestruje pakiety i rozpoznaje je jako istniejące połączenia.

Następna tablicą do której zostaje przekazany pakiet, jest tablica *mangle*. W tablicy tej możemy manipulować wartościami TTL.

Kolejną tablicą przetwarzającą pakiet, jest tablica *nat*. W tablicy tej możemy manipulować adresami pakietowymi. W zależności, czy naszym celem jest SNAT cz DNAT, używamy do tego odpowiednich łańcuchów PREROUTING i POSTROUTING

Ostatnią tablicą, która przetwarza pakiety, jest tablica *filter*. Jest to tablica w której powinno się umieszczać wszystkie reguły dotyczące filtrowania pakietów. Jeżeli nie zostanie jawnie podana tablica, domyślną tablicą jest tablica *filter*.

Jeżeli pakiet zostanie przetworzony przez całą ścieżkę w firewallu i nie zostanie podjęta decyzja o zaakceptowaniu bądź odrzuceniu pakietu, zostaje zastosowana polityka odpowiedniego dla tego pakietu łańcucha głównego, tj. INPUT, OUTPUT, FORWARD. Politykami mogą być TARGET-y terminujące, tj. ACCEPT, REJECT, DROP.

3.4 Najważniejsze kryteria dopasowania

Netfilter posiada bogaty wachlarz możliwości dopasowywania pakietów

--source, --src, -s ***<adres>***
dopasowuje adres źródłowy pakietu do podanego jako *adres*

--destination, --dst, -d ***<adres>***
dopasowuje adres docelowy pakietu do podanego jako *adres*

--protocol, -p ***<protocol >***
dopasowuje protokół używany przez pakiet.
Najczęściej używane protokołu to: *tcp*,*udp*,*icmp*

--source-port, --sport ***<port >***
dopasowuje port źródłowy pakietu.
Aby użyć tego dopasowania należy zdefiniować protokół.

--destination-port, --dport ***<port >***
dopasowuje port docelowy pakietu.
Aby użyć tego dopasowania należy zdefiniować protokół.

--tcp-flags ***<maska ><flagi >***
dopasowuje pakiet, jeżeli wszystkie *flagi* są ustawione oraz wszystkie flagi niewymienione w *flagi* a uwzględnione w *maska* są nieustawione

--mac-source ***<mac-adres >***
dopasowuje pakiet na podstawie źródłowego adresu sieciowego. Adres podawany jest w formacie: AA:BB:CC:DD:EE:FF.

--in-interface, -i ***<interface>***
dopasowuje pakiet ze względu na interface sieciowy na którym dany pakiet się pojawił

--out-interface, -o ***<interface>***
dopasowuje pakiet ze względu na interface sieciowy przez który pakiet będzie wysyłany

--limit ***<ilość>/<czas>***
dopasowuje pakiety, jeśli ich ilość nie przekracza *ilość* w okresie *czas*.
Czas może przyjmować wartości: second, minute, hour, day.

3.5 Najważniejsze działania

Jeżeli pakiet zostanie dopasowany, netfilter może wykonać jedną ze zdefiniowanych akcji a pakiecie. W przypadku niezdefiniowania akcji, pakiet przechodzi do kolejnych reguł w firewallu

a w firewallu zostaje zwiększony licznik dopasowanych pakietów dla danej reguły.
Dopasowany pakiet zostaje wysłany do wybranego targetu poprzez opcję -j, np: -j ACCEPT.

ACCEPT

dany pakiet zostaje zaakceptowany i nie przechodzi przez dalszą filtrację

REJECT

odrzucenie pakietu z wysłaniem informacji do adresata. Domyślna wartość odpowiedzi to icmp-port-unreachable.
Istnieje możliwość ustawienia odpowiedzi wysyłanej do adresata poprzez atrybut --reject-with <*type*>, gdzie *type* jest jednym z zdefiniowanych komunikatów:

- icmp-net-unreachable
- icmp-host-unreachable
- icmp-port-unreachable
- icmp-prot-unreachable
- icmp-net-prohibited
- icmp-host-prohibited
- icmp-admin-prohibited.

DROP

odrzucenie pakietu, bez wysyłani informacji zwrotnej do adresata. Pakiet zostaje "upuszczony".

LOG

pakiet zostaje wysłany do systemu logowania jądra. Jest to nieterminatorowy target, to znaczy, że po dopasowaniu pakietu, zostaje on zalogowany, a następnie przechodzi przez dalszą część firewalla.
Najczęściej jest on stosowany wraz z opcją --log-prefix, który dodaje prefix w systemie logowania. Pozwala on na późniejsze odróżnienie poszczególnych logów od siebie.

TTL

pozwala na manipulację wartościami TTL. Zalecane jest niezmienianie tej wartości, jednak w praktyce, są sytuacje w których zmiana wartości TTL jest potrzebna.
Możliwe opcje przekazywane do tego targetu:

--ttl-set <***wartość***>

ustawia wartość ttl na podaną

--ttl-inc <***wartość***>

zwiększa wartość ttl o podaną wartość

--ttl-dec *<wartość>*

zmniejsza wartość ttl o podaną wartość

REDIRECT

target ten przekierowuje pakiet na siebie. Istnieje również możliwość możliwość zmiany portu w danym pakiecie poprzez opcję: --to-ports *<port>*. Aby użyć opcji --to-ports, należy określić protokół -p *<protokół>*

SNAT

pozwala zmienić adres źródłowy dopasowanego pakietu. Określenie nowego adresu źródłowego dokonujemy za pomocą opji:

--to-source *<adres>* [:*port*[-*port*]]

gdzie *port* jest opcjonalnym parametrem określającym port, bądź zakres portów. Aby zdefiniować port(y) należy zdefiniować protokół.

DNAT

pozwala zmienić adres docelowy oraz port dopasowanego pakietu. Określenie nowego adresu docelowego dokonujemy za pomocą opji:

--to-source *<adres>* [:*port*]

Aby móc zdefiniować port, należy określić protokół.

Rozdział 4

Iptables

4.1 Zarys ogólny

Iptables jest konsolowym interfejsem dla netfilter-a. Pozwala on na tworzenie łańcuchów, dodawanie oraz usuwanie reguł, oraz wyświetlanie statystyk. Często nazwa iptables błędnie używana jest wymiennie z netfilter. Wynika to z faktu braku innych interfejsów do obsługi netfilter.

4.2 Polecenia iptables

Najczęściej wykorzystywane polecenia iptables to:

-t *<tablica>*

opcjonalny parametr, który możemy przekazać do każdej poniżej opisanej opcji, definiujący tablicę na której będziemy wykonywać operacje. Jeżeli ten parametr nie zostanie zdefiniowany, domyślną tablicą jest tablica *filter*.

-A *<łańcuch>* *<reguła>*

dodawanie nowej reguły na koniec łańcucha.

-I *<łańcuch>* **[nr]** *<reguła>*

wstawienie nowej reguły do łańcucha. Jeżeli zostanie podany parametr *nr*, reguła zostaje wstawiona na pozycję *nr*. Jeżeli parametr nie zostanie podany, domyślną wartością jest 1, czyli początek łańcucha.

-D *<łańcuch>* *<reguła>*

usuwa z łańcucha regułę podaną przez specyfikację.

-D *<łańcuch>* *<nr>*

usuwa z łańcucha regułę podaną przez numer porządkowy, liczony od 1.

-N <*łańcuch*>

tworzy nowy łańcuch o nazwie *łańcuch*.

-F [*łańcuch*]

usuwa wszystkie reguły z zadanego łańcucha. Jeżeli nie zostanie podany łańcuch, wyczyszczone zostaną wszystkie łańcuchy.

-X <*łańcuch*>

usuwa zadany łańcuch. Aby móc usunąć łańcuch, musi on być wcześniej wyczyszczony.

-P <*łańcuch*> <*polityka*>

ustawia politykę dla łańcucha.

-L [*łańcuch*]

wypisuje reguły w łańcuchu. Jeżeli wartość *łańcuch* nie zostanie podana, zostają wypisany wszystkie łańcuchy.

Często używane opcje polecenia -L, to:

-n

nie zamienia adresów ip na nazwy domenowe - często przyśpiesza wypisywanie wyników, gdyż nie oczekujemy na odpowiedzi od revdns-a.

-v

tryb gadatliwy. Wypisuje statystyki ilościowe i objętościowe dla wypisywanych reguł

Rozdział 5

Wykrywanie zagrożeń

5.1 SSH BruteForce

Atak polega na ciągłej próbie połączenia się z atakowanym komputerem za pomocą protokołu SSH, używając za każdym innego hasła z listy haseł. Jeżeli hasło użytkownika, znajduje się na liście haseł atakującego, istnieje duże prawdopodobieństwo, że atakującemu uda się przy którejś próbie połączyć z atakowanym komputerem.

5.1.1 Obrona

Jako obronę na ten typ ataku, zastosuję ograniczenie liczby połączeń z usługą SSH do 5 na minutę. Po wykryciu większej ilości połączeń, wygenerowany zostanie komunikat z ostrzeżeniem.

5.1.2 Implementacja

```
iptables -N SSH_BRUTE_DROPLOG
iptables -A SSH_BRUTE_DROPLOG
iptables -A SSH_BRUTE_DROPLOG -m hashlimit --hashlimit-upto 1/minute --hashlimit-name burute_log \
        --hashlimit-burst 1 -j LOG --log-prefix "IPS: SSH_BRUTEFORCE "
iptables -A SSH_BRUTE_DROPLOG -j REJECT

iptables -N SSH_BRUTE
iptables -A SSH_BRUTE -m hashlimit --hashlimit-above 5/minute --hashlimit-mode srcip \
        --hashlimit-name ssh_brute -j SSH_BRUTE_DROPLOG

iptables -A IPS -p tcp --dport 22 -j SSH_BRUTE
```

5.2 SYN Flood

Atak ten jest jednym z ataków DoS (Denial of Service) i polega wysyłaniu dużej ilości pakietów SYN do atakowanego hosta w celu nieumożliwienia pozostałym użytkownikom sieci

skorzystanie z atakowanej usługi.

Atak ten wykorzystuje specyfikę protokołu TCP i sposobu w jaki protokół ten nawiązuje połączenie. Aby nawiązać połączenie komputer łączący się wysyła pakiet SYN do serwera.
Serwer po otrzymaniu takiego pakietu, tworzy w tablicy połączeń wpis ze stanem "SYN RECEIVED" oraz odpowiada na to żądanie pakietem SYN+ACK sygnalizując swoją gotowość do nawiązania połączenia.
Następnie komputer łączący się wysyła pakiet ACK. Po takim nawiązaniu połączenia, nazywanym *three-way handshake*, oba hosty przeszły w stan połączeniowy (ESTABLISHED) i mogą zacząć wysyłać dane.

Atak ten polega na wysyłaniu dużej ilości spreparowanych pakietów SYN, z podmienionymi adresami źródłowymi, do atakowanego hosta. W efekcie atakowany host odpowiada pakietami SYN+ACK na adres podany jako adres źródłowy. Jeżeli jako adres źródłowy został podany aktywny host w sieci i otrzyma on nieoczekiwaną odpowiedź SYN+ACT, odpowie na nią pakietem RST. Po tej odpowiedzi atakowany host usunie wpis o połączeniu ze swojej tablicy.
Jeżeli natomiast jako adres źródłowy podamy nieaktywnego hosta w sieci, atakowany host odpowie do niego pakietem SYN+ACT i nie dostanie odpowiedzi, gdyż host jest nieaktywny. Spowoduje to, że atakowany host będzie czekał ustalony jako TIMEOUT czas, aż uzna że taki host nie istnieje i wtedy usunie wpis z tablicy połączeń. Przez ten czas, wpis jest obecny w tablicy połączeń. Jeżeli wysłana zostanie duża ilość pakietów SYN ze zmienionymi adresami źródłowymi, połączenia w stanie SYN RECEIVED wypełnią całą tablicę i nie będą przyjmowane kolejne połączenia.
W takim przypadku, próby połączenia się z atakowanym hostem przez zwykłych użytkowników zakończą się niepowodzeniem, gdyż host nie będzie przyjmował nowych połączeń z powodu przepełnienia tablicy połączeń. Jednocześnie, tablica ta nie będzie wolna, gdyż wysyłana w sposób ciągły fala pakietów SYN przez atakującego, wypełnia wolne miejsca po połączeniach, które zostały już usunięte z powodu przekroczenia TIMEOUT.

Nowsze wersje jądra Linux-a posiada wbudowaną obronę przez SYN Floodem. Aby zasymulować starszą wersję systemu, wyłączymy mechanizm syn_cookies:

```
echo "0" > /pros/sys/net/ipv4/tcp_syncoockies
```

5.2.1 Obrona

Aby zabezpieczyć się przed tego typu atakami, należy limitować ilość przychodzących pakietów SYN od jednego odbiorcy.
Jako domyślne wartości przyjąłem 1 połączenie przychodzące na sekundę, aktywowane po 5 pakietach.

5.2.2 Implementacja

```
iptables -N SYN_FLOOD_DROPLOG
```

```
iptables -A SYN_FLOOD_DROPLOG
iptables -A SYN_FLOOD_DROPLOG -m hashlimit --hashlimit-upto 1/minute --hashlimit-name syn_flood_log \
        --hashlimit-burst 1 -j LOG --log-prefix "IPS: SYN_FLOOD "
iptables -A SYN_FLOOD_DROPLOG -j REJECT

iptables -N SYN_FLOOD
iptables -A SYN_FLOOD -m hashlimit --hashlimit-above 1/second --hashlimit-mode srcip \
        --hashlimit-name syn_flood --hashlimit-burst 5 -j SYN_FLOOD_DROPLOG

iptables -A IPS -p tcp --tcp-flags ALL SYN -j SYN_FLOOD
```

5.3 ICMP Flood

ICMP Flood to najczęstszy z ataków mających na celu całkowite odcięcie dostępu do atakowanego hosta. Polega on na wysyłaniu bardzo dużej ilości danych. Ilość tych danych musi być większa niż przepustowość łącza atakowanego hosta. Następuje wtedy nasycenie pasma, i żądania od zwykłych klientów nie są dostarczane do hosta. Następuje odmowa dostępu.

Ataki tego tupu noszą nazwę DoS (Denial of Service) gdyż powodują odmowę dostępu do usługi. Jednakże, aby wykonać taki atak, atakujący musi dysponować łączem o większej przepustowości niż atakowany host.

Istnieje również odmiana ataków DoS poprzez flooda nie wymagająca większego łącza. Są to ataki DDoS (Distributed Denial of Service). Działają one w myśl tej samej idei wysycania łącza atakowanego hosta, jednak uzyskiwane jest to przy użyciu wielu komputerów atakujących. W takim przypadku suma przepustowości wszystkich atakujących komputerów musi być większa niż pasmo atakowanego hosta.

Do takich ataków najczęściej wykorzystywane są tzw. *botnet-y*, czyli sieci komputerów zainfekowanych wirusami, które przez większą część swojego życia na zainfekowanym komputerze nie wykazują żadnej aktywności. W momencie kiedy właściciel takiego botneta chce go wykorzystać, rozsyłana jest informacja do wszystkich zainfekowanych komputerów o celu ataku i przeprowadzany jest atak.

5.3.1 Obrona

Obrona przed atakami wykorzystującymi wysycanie łącza jest niemożliwa przy użyciu netfilter.

Wynika to z faktu, że gdy host jest atakowany dużą ilością pakietów, które wysycają łącze, netfilter może odrzucać pakiety agresora dopiero gdy docierają one do firewalla, czyli gdy już wysycą łącze. Nie ma możliwości przy użyciu narzędzia netfilter na zapobieganiu dostarczania pakietów do naszego komputera.

5.4 ICMP Timestamp Request

Żądanie ICMP Timestamp Request o numerze 13, jest badaniem podania czasu serwera. Nie jest ono samo w sobie atakiem, ale poznanie dokładnego czasu atakowanego hosta, może ułatwić złamanie algorytmów bazujących na generatorach liczb pseudolosowych opartych o aktualny czas.

Do serwera wysyłane jest zapytanie o numerze typu 13, a odpowiedź jest odsyłana w ICMP Timestamp Reply o numerze 14. Wiele nowoczesnych systemów w domyślnej konfiguracji blokuje pakiety Timestamp Request.

5.4.1 Obrona

W przypadku kiedy nie używamy synchronizacji czasu z serwerem za pomocą ICMP, możemy blokować zapytania tego typu.

5.4.2 Implementacja

```
iptables -N TIMESTAMP_DROPLOG
iptables -A TIMESTAMP_DROPLOG
iptables -A TIMESTAMP_DROPLOG -m hashlimit --hashlimit-upto 1/minute --hashlimit-name timestamp_log \
        --hashlimit-burst 1 -j LOG --log-prefix "IPS: ICMP_TIMESTAMP_REQUEST "
iptables -A TIMESTAMP_DROPLOG -j REJECT

iptables -A IPS -p icmp --icmp-type 13 -j TIMESTAMP_DROPLOG
```

5.5 Skanowanie portów pakietami SYN

Skanowanie portów pozwala na zbadanie atakowanego hosta pod kątem udostępnianych przez niego usług. Znając działające usługi na hoście, można dobierać odpowiednie metody.

Wysyłając pakiet SYN na skanowany port, host może zareagować na 4 sposoby:

odpowiedź SYN+ACK

oznacza ona, że port jest otwarty i nasłuchuje na nim jakaś usługa. Odpowiedź SYN+ACK jest drugim pakietem wymienianym w *3-way handshake*, co pokazuje nam, że została rozpoczęta procedura nawiązywania połączenia.

odpowiedź RST+ACK

oznacza ona, że port jest zamknięty. Jeżeli host otrzymuje żądanie SYN na port na którym nie jest spodziewane nawiązywanie połączenia, odpowiada pakietem z ustawioną flagą reset, która informuje o zerwaniu połączenia (w tym przypadku o zerwaniu próby połączenia z portem)

odpowiedź ICMP error message

oznacza ona, że port jest filtrowany na firewallu przez reguły REJECT, które generują odpowiedź do klienta o niemożności połączenia się z portem. Taka odpowiedź daje nam informacje, że na skanowanym hoście jest aktywny firewall, natomiast nie daje nam wiedzy czy na danym porcie działa jakaś usługa - połączenia mogą być filtrowane ze względu na IP źródłowe.

brak odpowiedzi

może być oznaką zarówno błędów sieci i zgubienia pakietów (w przypadku ograniczenia liczby retransmisji), bądź obecności firewalla i filtrowania pakietów metodą DROP. Pakiety takie zostają upuszczone i nie zostaje wysyła odpowiedź do hosta skanującego. Scenariusz błędów sieci jest zwykle mniej prawdopodobny, dlatego brak odpowiedzi najczęściej świadczy o obecności firewalla na skanowanym hoście.

Skanowanie portów metodą pakietów SYN jest bardzo podobne do metody ataku SYN Flood. Istnieją jednak pewne aspekty odróżniające te dwa działania, a mianowicie:

- ilość wysyłanych pakietów - w SYN Flood wysyłamy bardzo dużo pakietów, aby zapełnić tablicę połączeń, w skanowaniu wystarczy wysłać po jednym pakiecie na każdy badany port
- adres źródłowy pakietów - w SYN Flood podawaliśmy nieistniejący adres źródłowy, aby połączenie trwało w oczekiwaniu na odpowiedź, natomiast w skanowaniu portów, podajemy swój adres jako adres źródłowy, abyśmy mogli odebrać i zinterpretować odpowiedź serwera.

5.5.1 Obrona

Metoda obrony przed skanowaniem portów metodą SYN jest taka sama jak w przypadku SYN Flood, gdyż tak samo otrzymujemy dużą ilość pakietów SYN.

5.5.2 Implementacja

Patrz: 5.2.2: Wykrywanie zagrożeń/SYN Flood/Implementacja na stronie 24.

5.6 Skanowanie portów funkcjami systemowymi

Skanowanie portów funkcjami systemowymi, wykorzystuje oferowaną przez system operacyjny obsługę połączeń sieciowych. Nie dają one możliwości preparowania pakietów, dlatego w przypadku natrafienia na otwarty port przeprowadzana jest kompletna procedura *3-way-handshake* jak również nie ma możliwości zmiany adresów źródłowych ani innych wartości

w ramkach pakietu. Jednak, jest to jedyna metoda która może być wykonana na komputerze bez dostępu do konta administratora.

5.6.1 Obrona

Funkcje systemowe aby nawiązać połączenie wysyłają pakiety SYN, dlatego obrona przed tego typu skanowaniem jest taka sama jak przy wysyłaniu spreparowanych pakietów SYN.

5.6.2 Implementacja

Patrz: 5.5.2: Wykrywanie zagrożeń/Skanowanie portów pakietami SYN/Implementacja na stronie 27.

5.7 Skanowanie portów pakietami ACK

Skanowanie portów pakietami ACK wykorzystuje specyfikację protokołu TCP, który na nieoczekiwany pakiet z flagą ACK odpowiada pakietem RST.

Wysyłając spreparowany pakiet TCP z ustawioną flagą ACK, sprawiamy że system operacyjny na atakowanym hoście, nie jest w stanie go dopasować do żadnego istniejącego połączenia i odsyła pakiet z ustawioną flagą RST do atakującego hosta informując w ten sposób o nieprawidłowościach.
Po otrzymaniu odpowiedzi RST mamy informację, że atakowany nie filtruje na firewallu danego portu.

Metoda ta nie daje informacji czy na danym porcie jest uruchomiona jakaś usługa, a jedynie czy dany port jest filtrowany na firewallu. W przypadku filtrowania portu, nie dostaniemy żadnej odpowiedzi, gdyż pakiet zostanie upuszczone. W przypadku braku firewalla, wysłana zostanie odpowiedź RST.

5.7.1 Obrona

Aby obronić się przed takim atakiem, powinniśmy blokować wszystkie pakiety mające ustawioną flagą ACK oraz będące interpretowane jako nowe połączenia zamiast ustanowione.

5.7.2 Implementacja

```
iptables -N ACK_SCAN_DROPLOG
iptables -A ACK_SCAN_DROPLOG
iptables -A ACK_SCAN_DROPLOG -m hashlimit --hashlimit-upto 1/minute --hashlimit-name ack_scan_log \
        --hashlimit-burst 1 -j LOG --log-prefix "IPS: ACK_SCAN "
iptables -A ACK_SCAN_DROPLOG -j REJECT
```

```
iptables -N ACK_SCAN
iptables -A ACK_SCAN -j ACK_SCAN_DROPLOG

iptables -A IPS -p tcp -m state --state NEW --tcp-flags ALL ACK -j ACK_SCAN
```

5.8 Spoofing z sieci wewnętrznej

Ataki wykorzystujące spoofing polegają na podszywaniu się pod innego użytkownika sieci. Mają one na celu zafałszowanie informacji o atakującym.
Wykonując atak SYN FLOOD (patrz: 5.2), używaliśmy spoofingu w celu opóźnienia usuwania wpisów z tablicy. Jednak spoofingu można użyć np: do wykonania podejrzanego skanowania portów jako inny użytkownik sieci. Sprawi to, że administrator sieci, bazując na zapisach adresu IP z którego zostało wykonane skanowanie portów bądź inna próba ataku, skupi się na jego analizie, co może przysporzyć nieprzyjemności podszywanemu użytkownikowi sieci.
Innym celem takiego ataku, może być wywołaniu kilku oczywistych ataków z komputerów ofiar, tak, aby skupić uwagę administratora na tamtych wydarzeniach i w tym czasie wykonać cichy atak na serwer.
Wyróżniamy dwa główne typu spoofingu:

IP Spoofing

Polega on na preparowaniu pakietów podmieniając adres źródłowy IP. Efektem takiego ataku jest interpretowanie takich pakietów przez serwer jako pakietów wysyłany przez komputer ofiary. Minusem tej metody jest to, że odpowiedzi generowane przez serwer odsyłane są do komputera ofiary, ponieważ jej komputer odpowiada na zapytania ARP od serwera.

Ominięciem tej niedogodności jest ustawienie ręczne adresu IP identycznego z adresem ofiary. Wtedy mamy możliwość nawiązywania pełnych połączeń z serwerem jako komputer ofiary.

ARP Spoofing

Atak ten polega na podmianie wpisów w pamięci podręcznej protokołu ARP.

Protokół ARP(ang. Address Resolution Protocol) jest protokołem pozwalającym na dopasowanie adresów MAC do adresów IP. Komunikacja wykorzystująca adresy IP wykonywana jest w 3 warstwie modelu OSI - warstwie sieciowej. Jednakże, przełączniki działają w warstwie 2 - warstwie łącza danych, dlatego wysyłając pakiet IP, należy go opakować w ramkę łącza danych zawierającą adres MAC odbiorcy i nadawcy. Aby użytkownik nie musiał podawać tej wartości, opracowany został protokół ARP.

Zasada działania protokołu ARP:

Jeżeli system operacyjny chce wysłać pakiet do odbiorcy o podanym adresie IP, sprawdza pamięć podręczną w poszukiwaniu adresu MAC odbiory. W sytuacji w której adres

ten zostanie tam znaleziony, zostaje on wykorzystany. W przeciwnym przypadku system wysyła do wszystkich komputerów w sieci zapytanie o adres MAC komputera o poszukiwanym IP. Na tą prośbę odpowiada poszukiwany komputer, odpowiadając pakietem podpisanym swoim adresem MAC. Po otrzymaniu takiego pakietu, adres MAC zostaje zapisany w pamięci podręcznej i wykorzystany do wysłania pierwotnej wiadomości.

Podatnością tego protokołu jest interpretacja odpowiedzi ARP bez wcześniejszego zgłaszania zapotrzebowania na adres. Atak ARP Spoofing polega na wysłaniu do atakowanego hosta informacji ARP Reply zawierającej adres źródłowy podszywanego komputera oraz MAC adres atakującego. Atakowany host po otrzymaniu takiej informacji zapisze ją w pamięci podręcznej, bądź, jeśli już istniał wpis dla takiego adres IP - nadpisze starą informację nowa. Spowoduje to, że wysyłane pakiety z serwera do podszywanego komputera będą podpisywane adresem MAC atakującego, gdyż jego MAC znajduje się w pamięci podręcznej atakowanego hosta. Atakujący host powinien mieć ustawioną opcję ip_forward aby pakiet docelowo wysłany do podszywanego komputera, docierający do atakującego, mógł zostać wysłany dalej aby mógł osiągnąć swój cel. Jednak przechodząc przez komputer atakującego, daje on możliwość przechwycenia znajdujących się w nim informacji a nawet ingerencję w jego zawartość.

5.8.1 Obrona

Najlepszą metodą obrony przed spoofingiem jest zbudowanie bazy znanych hostów sieci wewnętrznej i umieszczenie jej w kontenerze IPSet. Najlepiej nadającym do się do tego tupu zadań kontenerem jest kontener typu *MACIPMAP*. Przechowuje on pary adres MAC-adres IP. Następnie dla każdego pakietu przychodzącego do serwera należy sprawdzić czy jego para IP--MAC znajduje się w bazie. Jeżeli nie znajduje się, może to oznaczać jedną z dwóch możliwości:

- jest to nowy host w sieci
- nastąpiła próba spoofingu

Aby wykluczyć pierwszą możliwość, administrator może sprawdzić czy MAC adres pakietu znajduje się w bazie. Jeżeli znajduje się, to oznacza to, że pakiet został spreparowany i prawdopodobnie wysłany w celu przeprowadzenia ataku.

5.8.2 Implementacja

```
ipset -N clients macipmap --network 192.168.240.0/24

ipset -A clients 192.168.241.41%00:21:85:c1:7f:83
ipset -A clients 192.168.242.219%00:0a:e6:86:0f:75
ipset -A clients 192.168.242.111%00:26:b6:66:d5:18
```

```
iptables -N WEW_SPOOF_DROPLOG
iptables -A WEW_SPOOF_DROPLOG -m hashlimit --hashlimit-upto 1/minute --hashlimit-name wew_spoof_log \
        --hashlimit-burst 1 -j LOG --log-prefix "IPS: WEW_SPOOF "
iptables -A WEW_SPOOF_DROPLOG -j REJECT


iptables -A IPS -i eth0 -m set ! --match-set clients src -J WEW_SPOOF_DROPLOG
```

Zakładamy że interface eth0 jest naszym interfejsem do sieci wewnętrznej a eth1 do sieci zewnętrznej

5.9 Spoofing z zewnątrz

Ataki tego typu polegają na podszywaniu się pod hosty z sieci wewnętrznej przy połączeniach z sieci zewnętrznej. Może to prowadzić do prowadzić do wykonania ataku DDoS przy użyciu adresu broadcast. Wysyłając pakiety z ustawionym adresem źródłowym na IP atakowanego hosta, IP docelowym na broadcast oraz docelowym adresem MAC na adres serwera, zostaną one przekazane przez serwer na adres broadcast, po czym każdy host w sieci odpowie na taki pakiet do hosta wskazanego przez adres źródłowy. Efektem tego może być kilkudziesięciokrotne zwiększenie liczby otrzymywanych przez atakowanego hosta pakietów do liczby wysyłanych pakietów przez atakującego.

5.9.1 Obrona

Aby obronić się przed tego typu atakami, należy filtrować wszystkie pakiety przychodzące na interfejs zewnętrzny posiadające adresy źródłowe należące do klas prywatnych.

5.9.2 Implementacja

```
iptables -N ZEW_SPOOF_DROPLOG
iptables -A ZEW_SPOOF_DROPLOG -m hashlimit --hashlimit-upto 1/minute --hashlimit-name wew_spoof_log \
        --hashlimit-burst 1 -j LOG --log-prefix "IPS: ZEW_SPOOF "
iptables -A ZEW_SPOOF_DROPLOG -j REJECT

iptables -N ZEW_SPOOF
iptables -A ZEW_SPOOF -s 192.168.0.0/16 -j ZEW_SPOOF_DROPLOG
iptables -A ZEW_SPOOF -s 172.16.0.0/12 -j ZEW_SPOOF_DROPLOG
iptables -A ZEW_SPOOF -s 127.0.0.0/8 -j ZEW_SPOOF_DROPLOG
iptables -A ZEW_SPOOF -s 10.0.0.0/8 -j ZEW_SPOOF_DROPLOG

iptables -A IPS -i eth1 -j ZEW_SPOOF
```

Zakładamy że interface eth0 jest naszym interfejsem do sieci wewnętrznej a eth1 do sieci zewnętrznej

5.10 Złośliwe oprogramowanie

Obecność wirusów w sieci może prowadzić do infekowania kolejnych komputerów, a w przypadku utworzenia tzw. *botnet*-u, przeprowadzenie ataku. Aktywność większości wirusów charakteryzuje się wysyłaniem dużej ilości poczty elektronicznej.

5.10.1 Obrona

Należy wychwytywać i blokować zbyt duży ruch poczty elektronicznej. Jednakże, należy tutaj uważać, na tzw. *false-positive* dopasowania, czyli uznanie normalnego ruchu pocztowego jako ruchu spamowego. Sytuacja taka może nastąpić podczas wysyłania aplikacji do pracy, gdzie następuje wysyłanie dużej ilości wiadomości do pracodawców.

Trzeba jednak pamiętać, że filtrowanie pakietów pocztowych nie zapewnia całkowitego zabezpieczenia. Należy jeszcze stosować dodatkowe filtry działające w warstwach wyższych.

5.10.2 Implementacja

```
iptables -N WIRUSY_DROPLOG
iptables -A WIRUSY_DROPLOG
iptables -A WIRUSY_DROPLOG -m hashlimit --hashlimit-upto 1/minute --hashlimit-name wirusy_log \
        --hashlimit-burst 1 -j LOG --log-prefix "IPS: WIRUSY "
iptables -A WIRUSY_DROPLOG -j REJECT

iptables -N WIRUSY
iptables -A WIRUSY -m recent --set --name wirusy
iptables -A WIRUSY -m recent --update --seconds 60 --hitcount 5 --name wirusy -j WIRUSY_DROPLOG

iptables -A IPS -p tcp --dport 25 -j WIRUSY
```

Rozdział 6

Raportowanie

6.1 Logowanie do pliku

Logowanie pakietów przy użyciu opcji *-j LOG*, w domyślnej konfiguracji zapisuje pakiety do pliku */var/log/messages*. W tym pliku zapisywane są wszystkie zdarzenia logowane na komputerze. Mnogość logowanych zdarzeń może utrudniać filtrowanie interesujących nas informacji z systemu IPS. Dlatego też, należy przekierować zapisywanie zapisów do osobnego pliku. W systemie Ubuntu 10.4, domyślnym systemem logowania jest *rsyslog*, dlatego przedstawiona zostanie konfiguracja dla tego systemu.

Cechą wspólną wszystkich zapisów tworzonych przez opisywany system IPS, jest występowanie frazy "IPS:". Wszystkie wpisy zawierające powyższą frazę traktujemy jako wpis systemu i przekierowujemy do osobnego pliku. Aby to osiągnąć, umieszczamy plik konfiguracyjny:

Listing 6.1: /etc/rsyslog.d/ips.conf

```
1 :msg, contains, "IPS: " -/var/log/ips.log
2 & ~
```

6.2 Logowanie do bazy danych

Ponieważ operowanie na plikach tekstowych jest zarówno niewygodne jak i mniej optymalne niż operowanie na rekordach bazy danych, zdarzenia będą zapisywane do bazy danych.

Zapisywania będziemy dokonywać, uruchamiając co minutę skrypt napisany w języku Python. Skrypt ten będzie filtrować plik */var/log/ips.log*, zawierający zdarzenia z systemu IPS, tak, aby uzyskać tylko zdarzenia zarejestrowane w poprzedniej minucie. Następnie przeprowadzi analizę wpisów zdarzeń i dokona wpisu do bazy danych.

Każdy rekord w bazie danych, reprezentujący zdarzenie, będzie zawierał podstawowe parametry zdarzenia, tj:

- czas w którym nastąpiło wykrycie zdarzenia

- typ ataku
- adres źródłowy ataku
- adres docelowy ataku
- port źródłowy
- port docelowy
- użyty protokół
- dokładny wpis wygenerowany przez iptables

Skrypt realizujący parsowanie pliku dziennika wygląda następująco:

Listing 6.2: log_events.py

```
#!/usr/bin/env python
import os,sys
os.chdir(os.path.dirname(sys.argv[0]))
os.environ['DJANGO_SETTINGS_MODULE']='settings'
from django.core.management import setup_environ
import settings
from logs.models import *
setup_environ(settings)

import datetime

t=datetime.datetime.now()-datetime.timedelta(minutes=1)
czas="%d %02d:%02d:"%(t.day,t.hour,t.minute)
plik=open("/var/log/ips.log")
lines=plik.readlines()
for line in lines:
  if czas in line:
    data=line.split()
    logczas="%d-%02d-%02d %s"%(t.year,t.month,t.day,data[2])
    data=data[7:]
    typ="Brak"
    src="Brak"
    dst="Brak"
    sport="Brak"
    dport="Brak"
    proto="Brak"
    typ=data[0]
    for d in data:
      if d.startswith("SRC="):
        src=d[4:]
      elif d.startswith("DST="):
```

```
            dst=d[4:]
        elif d.startswith("DPT="):
            dport=d[4:]
        elif d.startswith("SPT="):
            sport=d[4:]
        elif d.startswith("PROTO="):
            proto=d[6:]
    Event(czas=logczas,typ=typ,src=src,dst=dst,sport=sport,\
        dport=dport,proto=proto,log=line).save()
```

Aby ograniczyć zbytni przyrost objętości plików dziennika, który może powodować zapełnienie przestrzeni dyskowej oraz spowolnienie analizy pliku, zastosowano ograniczenie zapisywanie zdarzeń do jednego na minutę. Daje to dobre wyważenie pomiędzy ograniczeniem wielkości danych a zachowaniem wszystkich ważnych informacji.

Podsumowanie

Przedstawione metody ataku stanowią duże zagrożenie dla systemów informatycznych, jednak jak udało się pokazać, obrona przed większością z nich nie jest trudna. Większość przedstawionych ataków sieciowych można zablokować wykorzystując jedną regułę zapory ogniowej, co nie niesie dużego narzutu czasu procesora, a jest w stanie uchronić broniony serwer przed atakami a właściciela przed stratami.
Opracowany w niniejszej pracy system IPS jest w stanie zabezpieczyć serwer przed omówionymi atakami. Oczywiście trzeba mieć na uwadze, że przedstawione ataki są atakami głównie w trzeciej warstwie modelu OSI, dlatego przedstawiony system w żadnym wypadku nie chroni serwera przed atakami z warstwy aplikacji - takimi jak aktywność wirusów bądź błędy w skryptach PHP, jednakże pełni on ważną rolę w obronie systemu.

Bibliografia

[1] http://ipset.netfilter.org/iptables.man.html.

[2] http://newartisans.com/2007/09/neat-tricks-with-iptables.

[3] http://en.wikipedia.org/wiki/List_of_IP_protocol_numbers.

[4] http://www.netguru.net/ntc/NTCC8.htm.

[5] http://www.bilgiagi.web.tr/wp-content/uploads/2011/03/tcp_3way.gif.

[6] http://www.shorewall.net/images/Netfilter.png.

[7] K. Krysiak, *Sieci komputerowe: kompendium*, Helion, 2005.

[8] A. Mroczko and A. Bojczuk, *Internet: agresja i ochrona*, Wydawnictwo Robomatic, 1998.

[9] K. Siyan, T. Parker, and P. Koronkiewicz, *Tcp/ip*, Księga Eksperta, Helion, 2002.

Spis rysunków