Zastosowanie sieci Bayesa w wykrywaniu ataków DoS

Marcin Żurakowski, Przemysław Kazienko
Zakład Systemów Informacyjnych, Wydział Informatyki i Zarządzania, Politechnika Wrocławska
mzurakowski@esolution.pl, kazienko@pwr.wroc.pl

Wstęp

Powstające w latach 70'tych sieci ARPANET, NFSNET dały początek Internetowi, czyli powszechnie znanej i wykorzystywanej globalnej pajęczynie. W zamysłach projektantów sieć miała służyć przede wszystkim celom wojskowym lub naukowym: zapewniać łączność, wspomagać przeprowadzanie badań i wymiany informacji pomiędzy ośrodkami. Zapewne żaden z jej twórców nie spodziewał się, że kiedyś Internet będzie siecią oplatającą cały świat i jednym z ważniejszych filarów światowej gospodarki. Dziś Internet łączy miliony ludzi a w jego wirtualnej rzeczywistości możemy znaleźć coraz to więcej usług pochodzących ze świata rzeczywistego (e-handel, e-gospodarka, e-wybory, e-rozrywka).

Internet ma też swoją ciemną stronę: włamania do systemów, blokowanie usług, wirusy, spam to tylko kilka zagrożeń czyhających na ludzi używających Internetu. Błędy popełnione we wstępnych latach podczas projektowania protokołów i usług wraz z niedoskonałym i dziurawym oprogramowaniem, spowodowały, że te niedoskonałości zamieniły się w prawdziwe plagi.

Jednym z takich zagrożeń są ataki typu DoS/DDoS (*Denial of Sernice / Distributed Denial of Service*), czyli działania mające na celu zablokowanie wybranej usługi i uniemożliwienie korzystania z niej uprawnionym użytkownikom.

Niniejszy referat koncentruje się na podgrupie tych ataków występujących w warstwie sieciowej i transportowej modelu OSI [12]. Zaproponowany system BIDS (*Bayesian Intrusion Detection System*) został stworzony, jako praktyczna część pracy [17]. Jego wersja prototypowa została zaimplementowana, wdrożona i przetestowana w warunkach zbliżonych do rzeczywistych.

1. Ataki typu DoS/DDoS

Ataki typu DoS (*Denial of Service*) czyli odmowy usługi [9] mają na celu uniemożliwienie uprawnionym użytkownikom skorzystania z zasobów lub usług systemu komputerowego. Istnieje bardzo wiele odmian tego ataku - od przecięcia kabla zasilającego po skoordynowane uderzenie pakietami sieciowymi z tysięcy komputerów jednocześnie [5].

Urządzenia komputerowe do sprawnego działania potrzebują kilku podstawowych zasobów, takich jak czas procesora, powierzchnia dyskowa, pamięć RAM, przepustowość łącza. Ataki DoS starają się doprowadzić do zużycia bądź zablokowania tych zasobów, wykorzystując przy tym wiele z codziennie wykonywanych przez użytkowników czynności. Przepełnione konto pocztowe i zgubione listy do prosty przykład ataku DoS o nazwie Mail Bombing [8].

Można również utworzyć bardzo wiele małych plików, co np. w systemie Linux może doprowadzić do wyczerpania maksymalnej liczby węzłów (*INODE*)

przechowujących informacje o plikach i w konsekwencji do niemożności stworzenia jakiegokolwiek następnego pliku. Kolejną formą ataku DoS jest zajęcie całego czasu procesora przez procesy jednego użytkownika (*fork bomb*), co uniemożliwia korzystanie z zasobów przez inne osoby.

Osobna grupa ataków DoS polega na zniszczeniu lub modyfikacji danych o konfiguracji programów i urządzeń. Intruz może zdalnie lub lokalnie zniszczyć pliki konfiguracyjne, blokując dostęp uprawnionym użytkownikom do danego programu (usługi) lub uniemożliwiając jego prawidłowe działanie. Na przykład modyfikując tablicę routingu może on zablokować całą sieć, a zmieniając wartość rejestru systemowego Windows - doprowadzić do wyłączenia niektórych usług lub do bardzo dziwnego, uniemożliwiającego pracę, zachowania się systemu. Istotnym środkiem obrony przed atakami DoS jest fizyczne zabezpieczenie sprzętu gdyż unieruchomienie kluczowego routera może skutecznie zablokować całą sieć na długi czas [2].

Ataki sieciowe DoS nie wymagają lokalnego bezpośredniego dostępu do atakowanej maszyny, wykorzystują one luki w oprogramowaniu sieciowym oraz niedoskonałości w protokołach komunikacyjnych. Aplikacje sieciowe po otrzymaniu niepoprawnych, często specjalnie spreparowanych danych, przerywają działanie lub zajmują cały czas procesora, a nawet restartują maszynę. Bardzo często ataki DoS wykorzystują błędy w implementacji protokołów w systemach operacyjnych, np. tworzenie niewłaściwej tablicy ARP, nie sprawdzanie długości pakietów przed ich przetworzeniem (ataki *Land*, *Teardrop*, *Ping of Death*) czy nieprawidłowa obsługa składania pofragmentowanych pakietów IP.

DDoS (*Distributed Denial of Service*) jest udoskonaloną a przede wszystkim rozproszoną wersja ataku DoS. Znacznemu zmodyfikowaniu uległy głównie skuteczność oraz "bezpieczeństwo" agresora. O ile w metodzie DoS atak odbywa się z komputera agresora, o tyle atak DDoS przeprowadzany jest w sposób rozproszony, tzn. z wielu przejętych wcześniej komputerów jednocześnie. Komputery te znajdują się w różnych lokalizacjach, a ich użytkownicy nie są świadomi tego, iż właśnie biorą udział w ataku na serwer internetowy. Aby komputer taki mógł wziąć udział w ataku musi być wcześniej „zarażony” odpowiednim programem złośliwym. Służą do tego różnego rodzaje konie trojańskie, które dopiero na wyraźny sygnał od agresora uaktywniają się i rozpoczynają proces destrukcji. Wykrycie takiego programu złośliwego jest stosunkowo trudne ze względu na to, iż aktywuje się on tylko i wyłącznie w momencie ataku, po czym znów przechodzi w stan uśpienia. Takie niepożądane programy mogą być bardzo inteligentne - po przeprowadzonym ataku starannie zacierają ślady swojej obecności w systemie nieświadomego użytkownika samoczynnie się deinstalując i kasując.

Dokładne omówienie kilkunastu rodzajów ataków, ich anatomii oraz proponowanych metod obrony przed nimi zostało zawarte w pracy [17].

1.1. Fazy ataku

W zależności od rodzaju ataku, jego przebieg może być za każdym razem inny. Wyróżnijmy jednak kilka faz ataku, które są wspólne dla większości z nich [5].

- Faza I – Poszukiwanie błędów

Do przeprowadzenia skomasowanego i skutecznego ataku typu DDoS agresor potrzebuje armii maszyn i systemów operacyjnych, nad którymi będzie mógł

przejąć kontrolę i wykorzystać do własnych celów. Dla dużych firm, portali i innych popularnych systemów internetowych nieprzerwana widoczność w sieci to ich być albo nie być. Ataki DoS/DDoS są dla nich dużym zagrożeniem, dlatego inwestują środki w dobrą infrastrukturę sieciową:

- Równoległe łącza do wielu dostawców często o przepustowościach setek Mbit/s.
- Wydajne routery, które są zaprojektowane do obsługi o wiele większego ruchu niż aktualnie występujący. Routery te często odpowiadają za balansowanie ruchu na wszystkich dostępnych łączach. Zaatakowanie jego z nich powoduje przeniesienie użytkowników na inne.
- Wydajne zapory ogniowe renomowanych firm.
- Systemy IDS (*Intrusion Detection Systems*), czyli systemami wykrywania włamań [4] [6] [7].
- Właściwa konfiguracja wszystkich elementów sieci oraz zaktualizowane oprogramowanie.
- Całodobowy monitoring, ze strony własnego personelu technicznego, jak również ze strony administratorów dostawców internetowych ISP.

Atak na takie instytucje wymaga sieci DDoS o liczbie setek tysięcy maszyn. Podczas pierwszych ataków typu DDoS, sieć agresora była budowana z maszyn pracujących pod kontrolą systemów klasy Unix. Jednak ich stosunkowo mała liczba oraz dbałość administratorów (w większości przypadków) o ich bezpieczeństwo powodowało, że zbudowanie wystarczająco dużej sieci nie było zadaniem prostym. Wtedy agresorzy zwrócili się w kierunku systemów biurkowych firmy Microsoft (Windows 95, Windows 98, Windows Me, Windows 2000, Windows XP, Windows 2003). W stosunku do maszyn pracujących pod systemem Unix, mają one więcej zalet w punktu widzenia agresora:

- Popularność. Sytemu klasy Windows są zainstalowane na ponad 95% [16] komputerów PC, zaś liczba maszyn działających pod tym systemem i podłączonych do sieci Internet jest ogromna.
- Niewiedza użytkowników. Przeciętny użytkownik tego systemu posiada nikłą wiedze na temat bezpieczeństwa sieciowego. Dodatkowo, użytkownicy, często z lenistwa, nie aktualizują swoich systemów, nie instalują prywatnych zapór ogniowych ani innych elementów zwiększających bezpieczeństwo.
- Duża liczba krytycznych błędów. Statystyki błędów pokazują, że w każdym kwartale jest wykrywanych średnio po kilka błędów, które mają status krytyczny i umożliwiają przejęcie zdalnej kontroli.
- Słabe bezpieczeństwo. Domyślnie w tej klasie systemów zwykły użytkownik ma prawa administratora i może zrobić wszystko ze swoim komputerem. Uruchamiane przez niego programy również nie podlegają ograniczeniom. Takie środowisko jest wprost wymarzone dla wirusów i robaków.

W tej pierwszej fazie, agresor albo intensywnie poszukuje błędów w systemach operacyjnych, albo czeka spokojnie, … aż inni je znajdą.

- Faza II – Kodowanie

Gdy odpowiedni błąd zostanie znaleziony, agresor musi stworzyć właściwy program (tzw. exploit), umożliwiający jego wykorzystanie. Do jego kodu, następnie dodawanie są kolejne moduły, z których najczęściej możemy spotkać:

- Moduł do skanowania sieci w poszukiwaniu następnych ofiar
- Moduł ukrywający narzędzie w systemie
- Moduły do infekcji innymi drogami (poczta elektroniczna, dyskietki, itp.)
- Moduł atakujący DDoS
- Moduł autodestrukcji
- Moduł komunikacji z innymi agentami i agresorem

Mniej pracowici wandale, zwykle składają robaka z gotowych klocków napisanych przez innych.

- Faza III – Budowanie sieci

Gdy narzędzie jest gotowe, wystarczy że agresor zarazi kilka komputerów na początek, a te z kolei zainfekują następne. W zależności od sposobu infekowania, poszukiwania innych maszyn oraz oczywiście popularności błędu, liczba maszyn w sieci DDoS może w ciągu kilku godzin osiągnąć nawet kilka milionów.

- Faza IV – Atak

Atak na wybrany cel odbywa się albo o ściśle określonym czasie, już wybranym podczas kodowania, albo agenci komunikują się między sobą i bezpośrednio, lub pośrednio otrzymują komendę od agresora.

2. System wykrywania ataków DOS/DDOS – BIDS

BIDS (*Bayesian Intrusion Detection System*), jest aplikacją, której celem jest wykrywanie ataków typu DoS/Dos [17]. Aplikacja ta koncentruje się na wykrywaniu anomalii w natężeniu ruchu. Szczególnie czuła jest na zwiększony ruch w wybranych protokołach. Takie założenie projektowe spowodowało, że system BIDS nie wykryje ataku DoS/DDoS, który polega na wysłaniu pojedynczego, specjalnie przygotowanego pakietu. W takich przypadkach bardzo dobrze sprawują się tradycyjne systemy IDS [6] [7], np. Snort [3], które posiadają swoją bazę sygnatur ataków. BIDS nie jest konkurencją dla nich, a jedynie uzupełnieniem ich funkcjonalności. Jest to więc system wykrywający anomalie (*anomalny detection*), a nie wzorce ataków (*signature detection*).

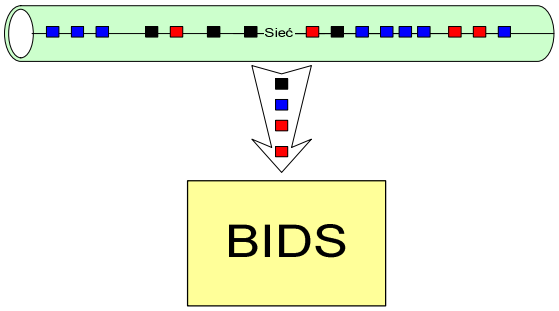
Założeniem projektowym było nie tylko wykrywanie kilku popularnych ataków DoS/DDoS, ale także ostrzeganie o wzroście podejrzanego ruchu, który nie został zidentyfikowany. Wybrane charakterystyczne wykrywane ataki to: powódź *ICMP Redirect*, *ICMP Smurf*, powódź *ICMP Unreach*, powódź *UDP Chargen*, atak na serwer DNS, *SYN Flood*, atak na serwer SMTP, atak na serwer WWW.

Inne ataki, które nie pojawiły się na powyższej liście, zostaną zakwalifikowane jako: powódź pakietów ICMP, powódź pakietów UDP, powódź pakietów TCP.

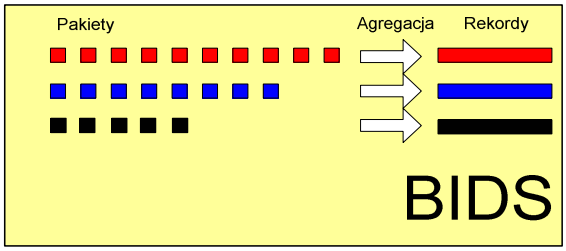
System BIDS nie tylko wykrywa ataki, które są dokonywane na sieć chronioną, ale również ostrzega on o anomaliach ruchu wychodzącego z tej sieci. Przez to możemy zostać poinformowani, że pośredniczymy, na przykład, w ataku typu SMURF.

2.1. Koncepcja działania systemu

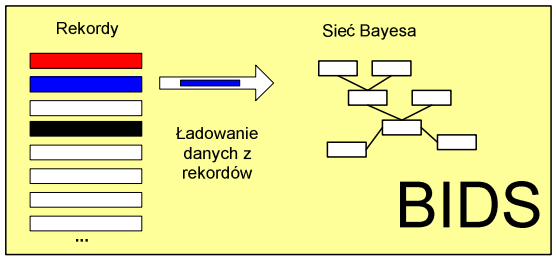
Aplikacja BIDS przełącza interfejs sieciowy w tryb nasłuchiwania pełnego (*promiscuous*). W tym trybie karta sieciowa potrafi odbierać wszystkie pakiety, które fizycznie do niej docierają, mimo, że nie są przeznaczone dla niej (także te, z innym adresem docelowym MAC). System BIDS umożliwia zdefiniowanie przez użytkownika dodatkowych filtrów, które ograniczą ruch brany pod uwagę przez samą aplikację.



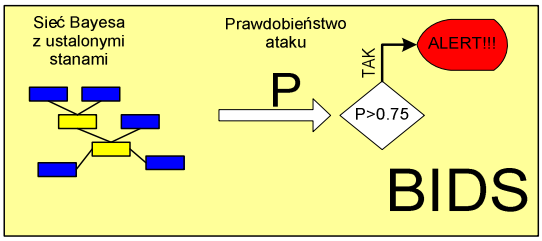
Przechwytywanie pakietów przez BIDS



Agregacja pakietów



Przekazanie zagregowanych danych do sieci Bayesa



Analiza danych przez sieć Bayesa

Przechwycone pakiety są **klasyfikowane** na dwie kategorie: ruch do chronionej sieci i ruch od chronionej sieci. Pozostałe pakiety są odrzucane. W następnych kroku dane z docierających pakietów (np. rozmiar, ilość) są sumowane z innymi. W ten sposób następuje **agregacja danych**. Zagregowane dane są następnie zapisywane do pamięci operacyjnej i czekają na zanalizowanie. Dane te tworzą rekordy. Rekord może odpowiadać pojedynczemu połączeniu (adres źródłowy IP, port źródłowy, adres docelowy IP, port docelowy), jak i może przedstawiać ruch na daną usługę w sieci chronionej (wszystkie połączenia do serwera WWW).

Każdy rekord jest **analizowany cyklicznie**, co jakiś czas. Długość tego okresu jest zależny od aktualnego obciążenia sytemu, jednak nie jest mniejszy od 4 s (domyślnie). Następnie system wyszukuje w pamięci te rekordy, które zostały zmodyfikowane od ostatniej analizy. Jeśli jakiś rekord spełnia te warunki, to **dane z niego są przepisywane do węzłów sieci Bayesa** – wartości oznaczają stan węzła. Stany wszystkich węzłów mają charakter dyskretny. Dla niektórych wartości, jak ilość pakietów na sekundę system stosuję specjalne funkcje dyskretyzujące.

W dalszej kolejności **sieć Bayesa oblicza prawdopodobieństwa** nieustalonych węzłów. Jednym z nich (centralnym) jest węzeł oznaczający typ ataku. W ten sposób zostaje obliczone prawdopodobieństwo ataku P. W przypadku, gdy jest większe od zdefiniowanego progu (0.75 - domyślnie), **na ekran jest wypisywane ostrzeżenie**. Przykładowy alert ma postać:

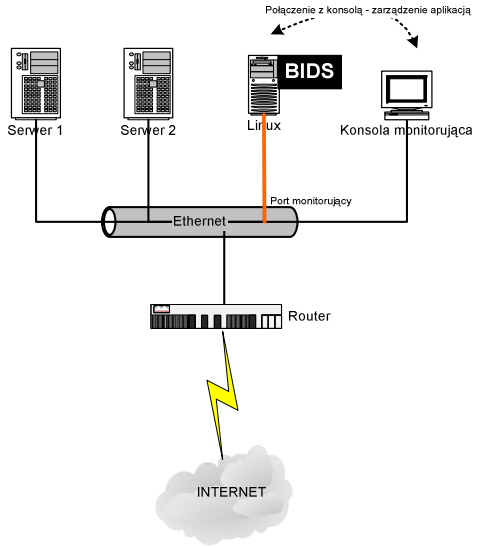
```
Atak 'tcp_syn_flood' z prawdopodobienstwem 0.987 z
192.168.10.10:0 -> 192.168.0.1:70
```

Każdy alert zawiera nazwę rozpoznanego ataku, jego prawdopodobieństwo, źródłowy adres IP, port źródłowy, docelowy adres IP, port docelowy.

Kiedy dany rekord został już zanalizowany, jego wartości są zerowane (ilość pakietów, sumaryczny rozmiar itp). Jeżeli przyjdą kolejne pakiety, które będą pasować do danego rekordu, to dane o nich zostaną do niego dodane. System BIDS dba o optymalne wykorzystanie pamięci, dlatego rekordy, które nie były przez dłuższy czas używane (domyślnie 60 s), są usuwane z pamięci.

2.2. Miejsce instalacji systemu

System BIDS ma charakter pasywny. Obserwuje on tylko ruch przechodzący przez sieci, ale nie wpływa na niego. Aplikację można zainstalować w kilku miejscach:



Umiejscowienie systemu BIDS

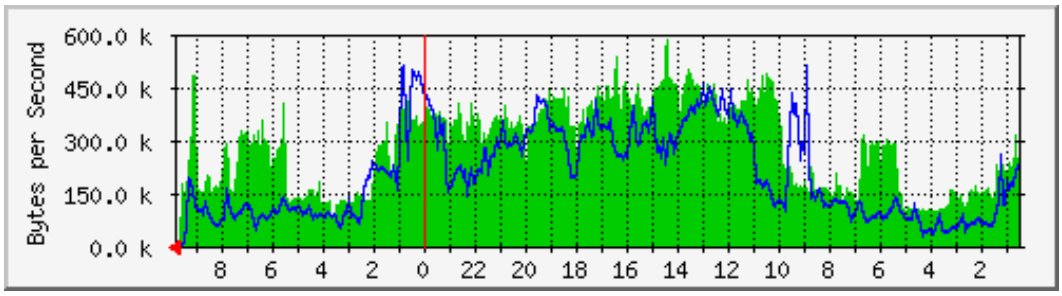
- Brama sieciowa – jest to możliwe, o ile funkcję tą pełni komputer klasy PC z zainstalowanym systemem Linux.
- Wewnątrz chronionej sieci – zalecana konfiguracja, jeśli nasz przełącznik w sieci lokalnej posiada port monitorujący, na który będzie kierowany cały ruch wejściowy i wyjściowy.
- Instalacja na jednym z serwerów – system w takiej konfiguracji będzie chronił tylko ten jeden komputer pod warunkiem, że będzie on pracował pod system Linux.

2.3. Obserwacja parametrów sieci

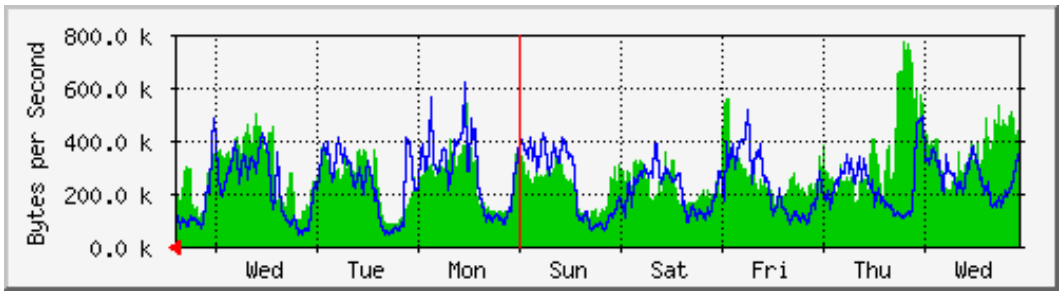
W każdej rzeczywistej sieci ilość, wielkość oraz natężenie przesyłanych danych jest inne. Dodatkowo wartości te zmieniają się w czasie na przestrzeni doby (duży ruch w dzień, mały w nocy), oraz w dłuższym okresie (zwiększanie ilości klientów – powolny wzrost ruchu). Parametry te zależą od bardzo wielu czynników, z których główne to:

- Liczba serwerów w chronionej sieci
- Konfiguracja sieci (zapory ogniowe, filtry i ograniczniki ruchu)
- Rodzaj udostępnianych usług (protokoły, wielkości przesyłanych danych)
- Przepustowość łącz danych do dostawców ISP
- Liczba klientów i ich zachowania
- Kultura pracy w firmie (godziny pracy)

Przykładowe wykresy obciążenia łączą zostały pobrane od jednego z dostawców dostępu do sieci Internet (ISP).



Przykładowy wykres ilości odebranych i wysłanych danych (okres doby)



Przykładowy wykres ilości odebranych i wysłanych danych (okres tygodnia)

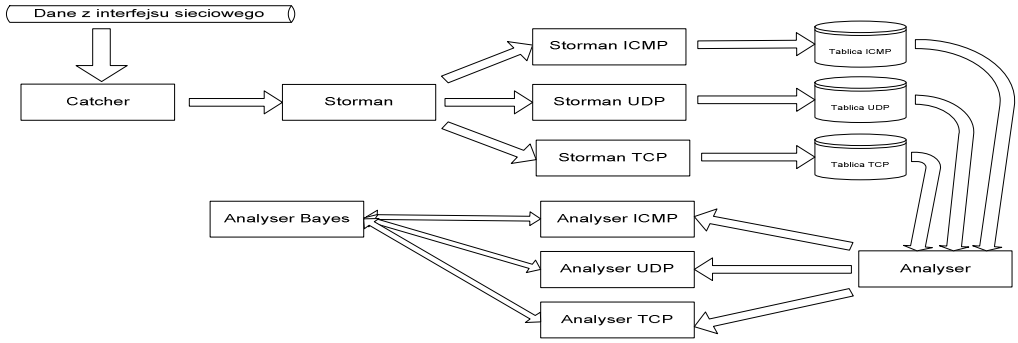
System BIDS potrzebuje do pracy średnich wartości przesyłanych danych. Wyznaczenie tego w sposób automatyczny jest zadaniem trudnym i może okazać się, że otrzymane wyniki będą błędne. Dodatkowo podczas automatycznej obserwacji sieci może zdarzyć się wiele zaburzeń, które zniekształcą wyliczone dane:

- Atak typu DoS/DDoS (nagły wzrost ruchu)
- Awaria łącza (prawie całkowity zanik ruchu)
- Prace administracyjne (wykonywana kopia zapasowa na inny serwer, rekonfiguracja sieci itp)

Z tego względu system BIDS wymaga od administratora ręcznego uruchomienia w specjalnym trybie, jak i również ręcznego zatrzymania. Podczas tego okresu będą mierzone średnie parametry przesyłanych danych. Takie działanie ma kilka zalet. Po pierwsze administrator może wybrać okres mierzenia parametrów (kilka godzin, doba, tydzień). Zmierzenie średniego ruchu w dzień, spowoduje, że system będzie prawdopodobnie za mało czuły w nocy (mniejszy ruch), natomiast uruchomienie tego trybu w nocy może spowodować, że w dzień jego czułość będzie za duża. To pozwoli administratorowi dobrać optymalny okres (np. doby). Drugim ważnym plusem jest to, że w przypadku niespodziewanego zdarzenia, które spowoduje zmniejszenie, lub zwiększenie ruchu pomiar może być przerwany i powtórzony jeszcze raz. Dlatego podczas tego trybu pracy zaleca się dodatkowo obserwację obciążenia sieci, na przykład przy pomocy takich narzędzi jak MRTG.

2.4. Moduły wewnętrzne

W celu funkcjonalnego podziału i lepszego zarządzania kodem, system BIDS składa się z kilkunastu modułów. Główne moduły wraz z kierunkiem przepływu danych zostały przedstawione na poniższym rysunku.



Moduły systemu BIDS i kierunki przepływu danych

- Moduł Catcher – odpowiedzialny za przechwytywanie pakietów z interfejsu sieciowego, klasyfikowaniu ich ze względu na protokoły oraz przekazywanie danych do dalszych modułów.
- Moduł Storman - odpowiedzialny za przechowywanie i zarządzanie informacjami o połączeniach (zarządzenie pamięcią)
- Moduł Analyser – odpowiedzialny za analizę zgromadzonych danych przechowywanych w pamięci operacyjnej. Moduł działa w postaci wątku, osobno od reszty modułów. Celem takiej implementacji było uniezależnienie go od reszty programu i dowolne dostosowanie tempa i czasu przetwarzania.
- Moduł Analyser Bayes - mózgiem całego systemu. Odpowiada on za załadowanie zgromadzonych danych do sieci Bayesa i odczytanie prawdopodobieństwa.

2.5. Zastosowanie sieci Bayesa

U podstaw koncepcja sieci Bayesa leży twierdzenie podane przez angielskiego matematyka Thomas Bayes (1702–61). W rzeczywistym świecie jest wiele sytuacji, w których wystąpienie jakiegoś zdarzenia ściśle zależy od innego zdarzania. Zastosowanie sieci Bayesa pozwala na uniknięcie obliczeń o dużej złożoności – obliczenie jednego prawdopodobieństwa a posteriori łączy się z uprzednim obliczeniem wykorzystywanych prawdopodobieństw. Ogromną zaletą sieci Bayesa jest przedstawianie wiedzy niepewnej, zdarzeń, co, do których nie ma pewności, że zaszły. Klasyczna sieć Bayesa składa się z węzłów, które reprezentują zmienne oraz łuków definiujących powiązania pomiędzy węzłami. Graficznie przedstawienie sieci przybiera postać acyklicznego grafu.

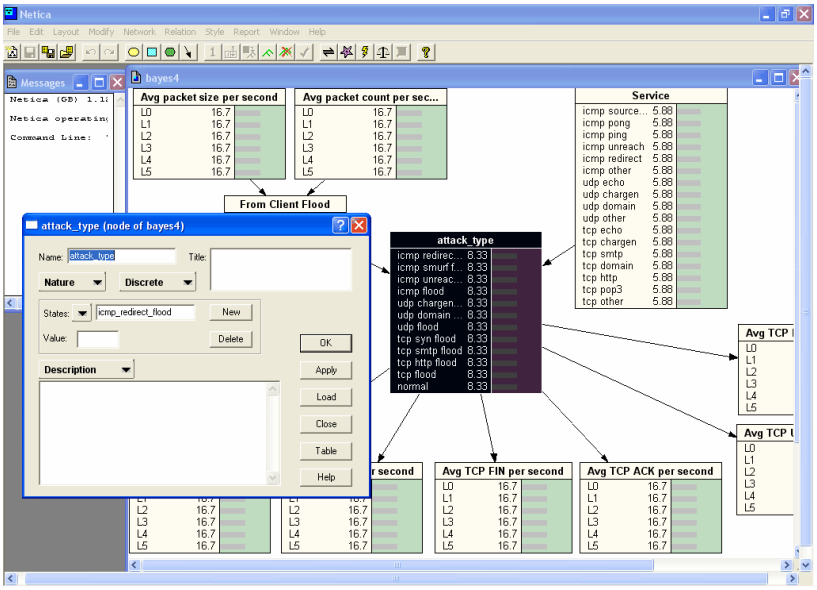
W dziedzinie wykrywania ataków sieć Bayesa posiada kilka ważnych zalet:

- Bardzo duża szybkość obliczeń – w przypadku pracy sytemu BIDS na bardzo obciążonej sieci, może być potrzeba setek analiz na sekundę. Obliczanie prawdopodobieństw wewnątrz sieci Bayesa jest proste i szybkie. System nie może pozwolić sobie na zajęcie czasu procesora na poziomie 100%, gdyż sam stałby się narzędziem DoS.
- Możliwość uczenia się – sieci Bayesa pozwalają na szybkie uczenie się. Jedynie, co trzeba im dostarczyć to stany zmiennych z rzeczywistych zdarzeń.

- Przetwarzanie wiedzy niepewnej – fakt zaistnienia ataku jest określany z pewnym prawdopodobieństwem o wartościach ciągłych.

Jako silnik implementujący sieć Bayesa została wybrany pakiet Netica firmy Norsys [13]. Przyczynami takiego wyboru były:

- Wersja graficzna pakietu pod system Windows
- Biblioteki C oraz Java zarówno na platformę Linux jak i Windows
- Dobrze zaprojektowane API
- Stabilność pakietu
- Darmowa wersja limitowana (limitowanie dotyczy ilości węzłów)

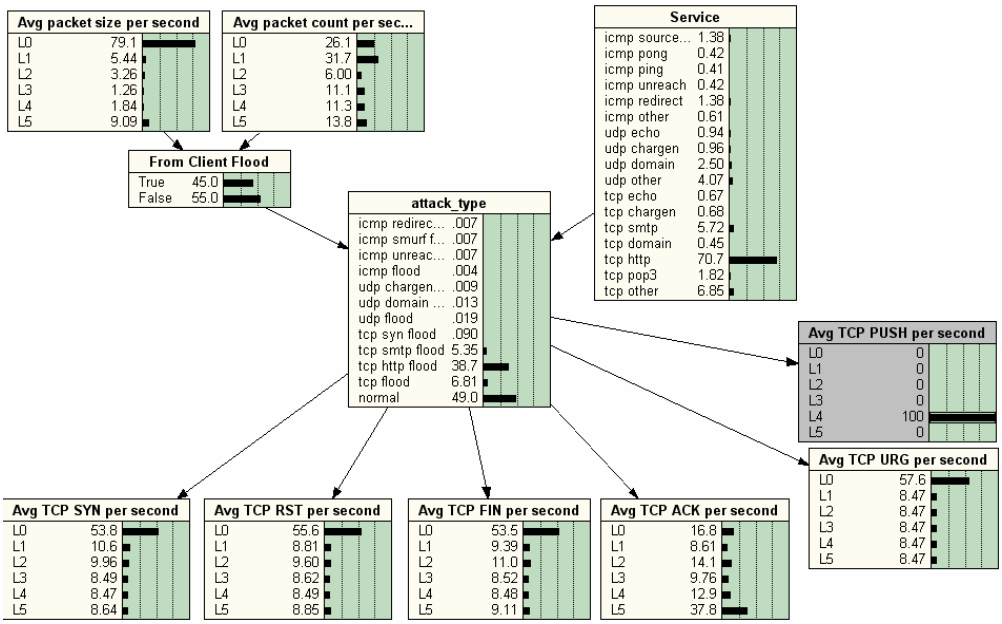


Graficzny moduł pakietu Netica (Windows)

Projekt sieci Bayesa został wykonany po przeanalizowaniu kilkunastu innych sieci z uwzględnieniem zależności pomiędzy gromadzonymi wartościami. Stany wszystkich węzłów mają charakter dyskretny. Ze względu na gromadzone dane, zostały zdefiniowane następujące węzły w sieci Bayesa:

- apsps (*Avg packet size per second*) - średnia ilość przesłanych danych w skali od L0....L5 (na sekundę).
- apcps (Avg packet count per second) – średnia ilość przesłanych pakietów w skali od L0...L5 (na sekundę).
- fcf (*From Client Flood*) – węzeł wskazujący, czy następuje powódź pakietów.
- srv (*Service*) – rodzaj usługi w sieci chronionej do jakiej kierowane są pakiety. Wybrano popularne usługi, które działają na protokołach: ICMP, UDP, TCP. Reszta usług będzie pokrywana przez stany węzła: icmp_other, udp_other i tcp_other.
- atsps (*Avg TCP SYN*) – średnia ilość pakietów protokołu TCP mających zapaloną flagę SYN w stali L0...L5 (dla ICMP i UDP zawsze L0).
- atrps (*Avg TCP RST*) – średnia ilość pakietów protokołu TCP mających zapaloną flagę RST w stali L0...L5 (dla ICMP i UDP zawsze L0).
- atfps (*Avg TCP FIN*) – średnia ilość pakietów protokołu TCP mających zapaloną flagę FIN w stali L0...L5 (dla ICMP i UDP zawsze L0).

- ataps (*Avg TCP ACK*) – średnia ilość pakietów protokołu TCP mających zapaloną flagę ACK w stali L0...L5 (dla ICMP i UDP zawsze L0).
- atpps (*Avg TCP PUSH*) – średnia ilość pakietów protokołu TCP mających zapaloną flagę PUSH w stali L0...L5 (dla ICMP i UDP zawsze L0).
- atups (*Avg TCP URG per second*) – średnia ilość pakietów protokołu TCP mających zapaloną flagę URG w stali L0...L5 (dla ICMP i UDP zawsze L0).
- attack_type – węzeł, który jako swoje stany posiada nazwy ataków. Ostatni stan o nazwie 'normal', oznacza brak ataku.



Schemat sieci Bayesa

2.6. Ważniejsze algorytmy - funkcje dyskretyzacji

Dane o ilości pakietów oraz ich rozmiarze, które są przechowywane w pamięci mają charakter ciągły – są to liczby rzeczywiste. Kluczowym fragmentem analizy jest ich zamiana na specjalną skalę 6-stopniową L0...L5, która jest wymagana przez sieć Bayesa. Do tego celu są wykorzystywane 3 funkcje dyskretyzujące, każda w innej sytuacji.

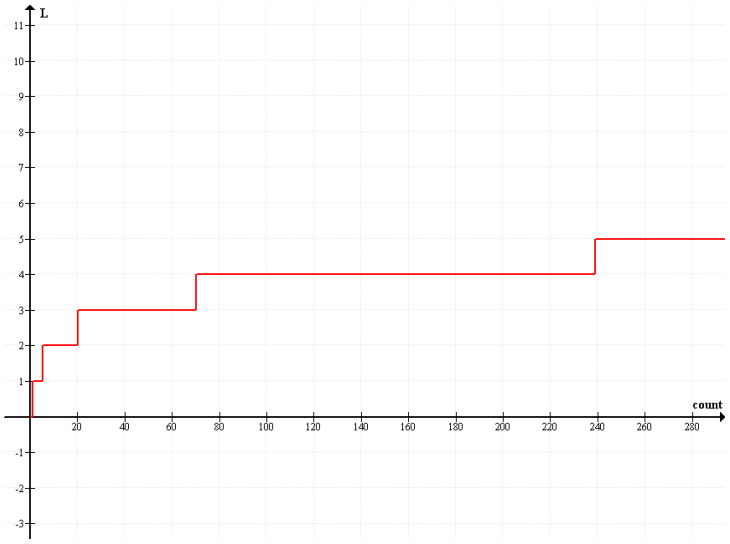
- Pierwsza z nich stosowana jest do zamiany ilości pakietów. Jest stosowana do wszystkich protokołów. Funkcja ma charakter logarytmiczny, rośnie szybko na początku, łagodniej dla większych argumentów i dlatego została wybrana do przeliczania liczby pakietów.

$$L = round(\log_{(nc+1)^{\frac{1}{MID\_LEVEL}}}(count+1)),$$

$$if \;\; L > MAX\_LEVEL \;\; then \;\; L = MAX\_LEVEL$$

gdzie:
L – obliczony poziom MIN_LEVEL...MAX_LEVEL
nc – współczynnik odczytany z pliku `net_stat.conf`
MID_LEVEL – średni poziom zdefiniowany w pliku konfiguracyjnym `config.h`
count – ilość pakietów w tym rekordzie danych (przechwyconych)
MAX_LEVEL – maksymalny poziom zdefiniowany w pliku `config.h`.

Dla: *nc*=20, *MID_LEVEL*=2.5, *MAX_LEVEL*=5 wykres funkcji *L*(*count*) jest następujący:

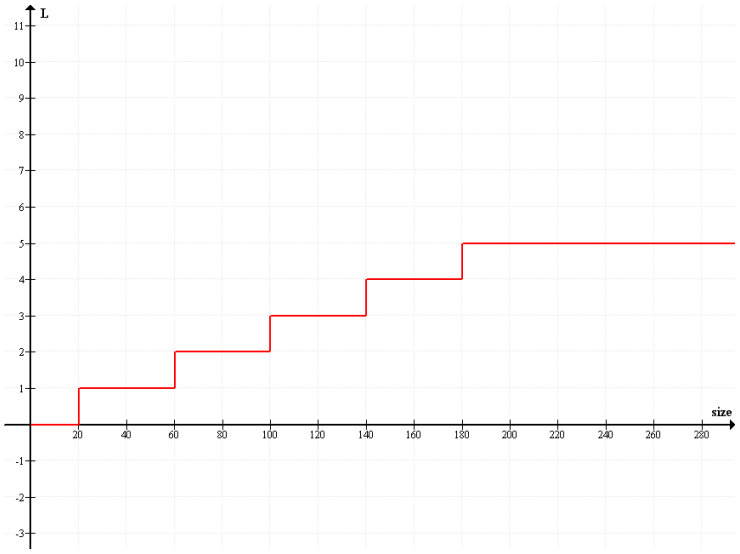


- Druga z nich, również jest stosowana do wszystkich protokołów i przelicza rozmiar zgromadzonych danych na poziom L0...L5. Ma charakter liniowy.

$$L = round(\frac{size * MID\_LEVEL}{ps}),$$
$$if \quad L > MAX\_LEVEL \quad then \quad L = MAX\_LEVEL$$

gdzie: *size* – rozmiar przechwyconych pakietów, *ps* – współczynnik odczytany z pliku `net_stat.conf`

Dla *ps*=100, *MID_LEVEL*=2.5, *MAX_LEVEL*=5 wykres funkcji *L*(*size*) jest następujący:



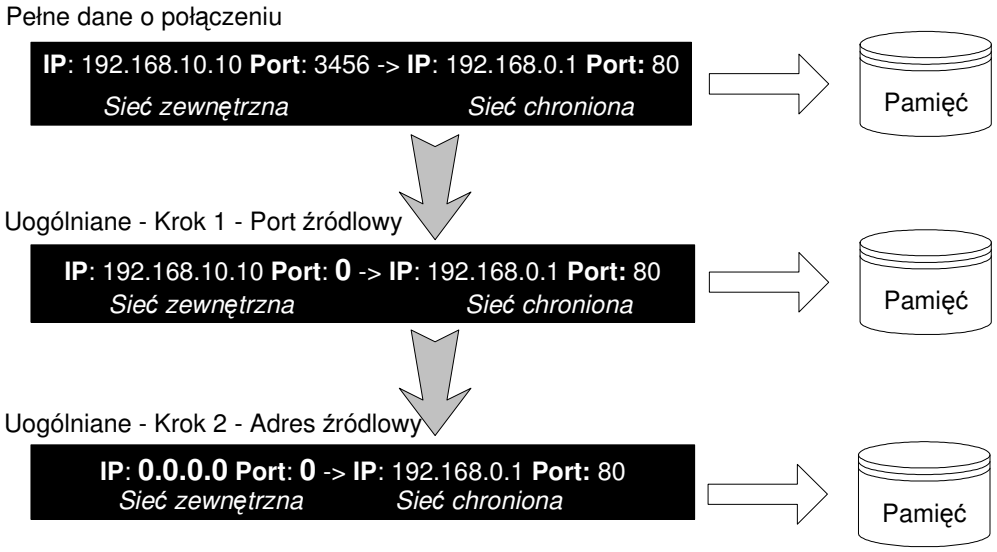
- Ostatnią funkcja ma za zadanie przeliczyć występowanie flag w pakietach TCP (SYN, ACK itp), na odpowiednie poziomy.

$$L = round(\frac{flag * MAX\_LEVEL}{count})$$

gdzie: *flag* – ilość pakietów z zapaloną daną flagą (przechwyconych)

2.7. Ważniejsze algorytmy – uogólnianie danych

Nie można poddawać analizie tylko rekordy zawierające dane o połączeniach komputer-komputer, gdyż w ten sposób atak na cała sieć został by przeoczony. Do dalszej analizy dane muszą być uogólniane. Postępowanie polega na odrzucaniu po kolei portu oraz adres nadawcy (z poza sieci chronionej) i następnie zsumowaniu danych z innymi połączeniami. Pozwoli to na wykrywanie ataków rozproszonych DDoS, oraz ataków na usługę w sieci chronionej, gdzie często adres i port źródłowy są losowane.



Schemat postępowania przy uogólnianiu danych

Powyższy przykład dotyczy protokołu UDP i TCP. Dla protokołu ICMP schemat postępowania jest podobny.

2.8. Ważniejsze algorytmy – uczenie sieci Bayesa

Sieć Bayesa składa się z węzłów, które reprezentują zmienne oraz łuków – prawdopodobnych zależności pomiędzy węzłami. Sieć charakteryzuje się tym, że poprzednik węzła bezpośrednio powoduje stan zmiennej skojarzonej z tym węzłem. Dla każdego węzła określa się rozkład prawdopodobieństwa warunkowego albo podając je wprost albo ucząc sieć na przykładach. Dla wartości dyskretnych funkcję prawdopodobieństwa warunkowego można zapisać w tabeli przedstawiającej prawdopodobieństwa przyjęcia poszczególnych wartości (własnych) przez węzeł dziecko, w zależności od kolejnych z możliwych wartości rodzica.

apsps	apcps	True	False
L0	L0	0.0286	99.971
L0	L1	0.0434	99.957
L0	L2	5.248	94.752
L0	L3	99.797	0.203
L0	L4	99.180	0.820
L0	L5	99.569	0.431
L1	L0	0.0861	99.914
L1	L1	0.0101	99.990
L1	L2	2.752	97.248
L1	L3	75.929	24.071
L1	L4	99.870	0.130
L1	L5	99.716	0.284
L2	L0	4.167	95.833
L2	L1	14.269	85.731
L2	L2	1.343	98.657

Przykładowy rozkład prawdopodobieństw w węźle sieci Bayesa

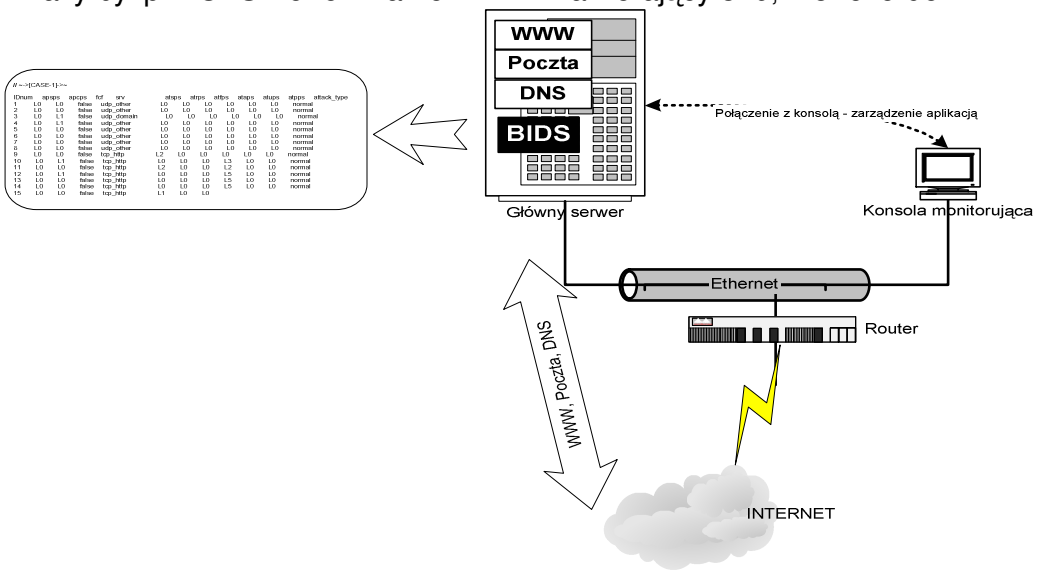
Podczas tworzenia systemu BIDS przyjęto, że sieć zostanie nauczona rozpoznawać ataki typu DoS/DDoS i prawdopodobieństwa nie będą ręcznie poprawiane. Przygotowanie odpowiednich pików uczących pozwoliło zrealizować to założenie.

System został zainstalowany na głównym serwerze jednego z dostawców usług internetowych. Serwer ten ma wiele funkcji, z których główne to:

- Serwer WWW
- Serwer poczty (SMTP, POP3, IMAP)
- Serwer DNS

System ten jest średnio obciążony podczas dnia roboczego na poziomie: 200 kbit/s ruchu wchodzącego do serwera i 1.5 Mbit/.s ruchu wychodzącego z serwera. Przed nauką system BIDS został uruchomiony w trybie obserwacji sieci, aby zebrać dane dotyczące średniego ruchu. Sam proces uczenia następował w dwóch fazach:

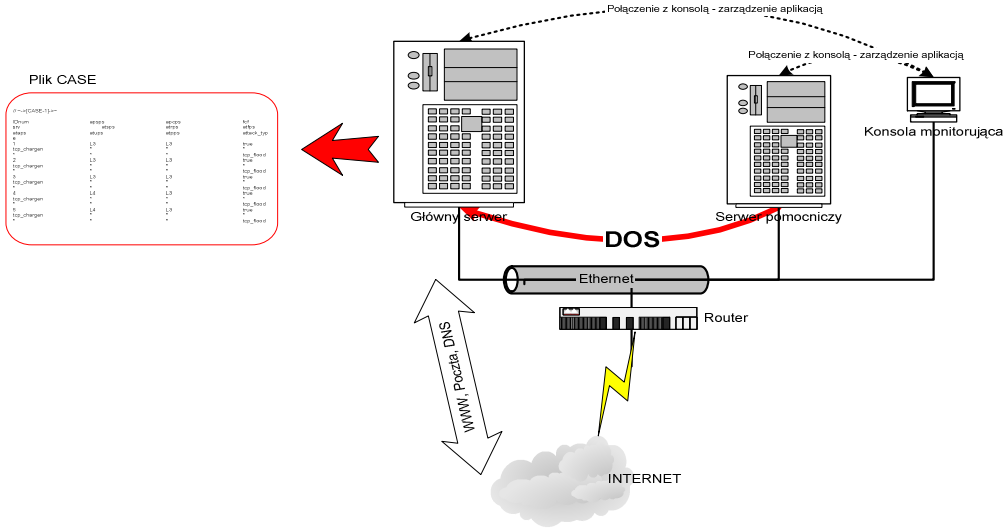
- W pierwszej fazie system BIDS został uruchomiony z opcją generowania pliku CASE. System w tym trybie obserwuje cały wychodzący i wchodzący ruch, agreguje dane i wypisuje wyniki na ekran. Wypisywanie wiersze mają tzw. stan normalny, co oznacza, że zostały na sztywno zakwalifikowane jako ruch prawidłowy (nie atak). Jednocześnie cały czas ruch wychodzący i wchodzący do serwera był obserwowany innym narzędziem (iptraf, MRTG), aby sprawdzić czy nie następuje próba ataku. Jeśli nastąpiłby prawdziwy atak, został by on przez system BIDS w tym trybie zakwalifikowany jako prawidłowy ruch. W rezultacie sieć dostała by błędne dane. Rezultatem tej fazy był plik CASE o rozmiarze 47MB i zawierający 319,748 rekordów.



Schemat sieci podczas pierwszej fazy uczenia

- Druga faza miała na celu nauczenie sieci Bayesa wykrywania ataków DoS/DDoS. Do nauki został użyty drugi serwer znajdujący się w sieci lokalnej. Z niego, przy wykorzystywaniu specjalnych narzędzi były symulowane ataki DoS oraz DDoS. Do części ataków stosowany był program hping2 [10], do innych zostały napisane specjalne skrypty w języku PERL i biblioteki Net::RawIP. Symulowane były kolejne ataki z różną siłą tj. zmianie ulegało natężenie pakietów oraz ich rozmiar. Najlepiej, aby nauka

nie odbywała się w sieci lokalnej, jednak nie było możliwości jej przeprowadzenia z komputera zdalnego, gdyż zostaliby odcięci klienci obu sieci. Rezultatem tej fazy były 4 pliki CASE: dla protokołu ICMP, UDP, TCP oraz osobny plik z wykrytymi atakami typu SYN Flood.

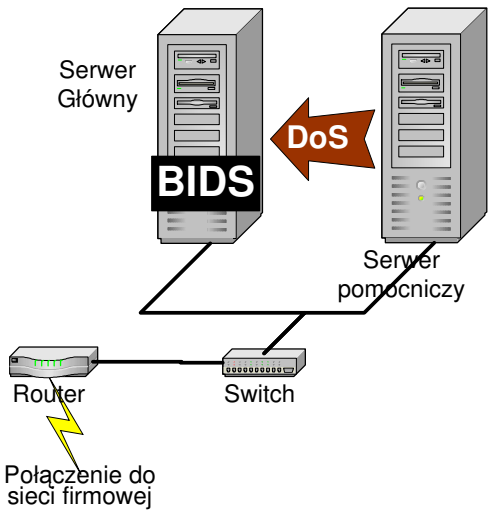


Schemat sieci podczas drugiej fazy uczenia

Wszystkie pliki z obu faz zostały następnie wczytane do pakietu NETICA, który dokonał ich przetworzenia. Podczas importu pakiet umożliwia wybranie tzw. stopnia importu. Jest to liczba całkowita, większa od 0. Wartość ta determinuje, ale razy każdy wiersz ma być przetwarzany. Jeśli plik CASE ma 10 przypadków, a podany współczynnik jest równy 5, wtedy każda linia 5 razy wpływa na zmianę prawdopodobieństw. Pliki CASE z atakami zawierały o wiele mniej danych niż plik CASE z prawidłowym ruchem, dlatego każdy z nich został zaimportowany z współczynnikiem 5. Więcej informacje o algorytmach uczenia sieci Bayesa można znaleźć w pomocy pakietu Netica [13] oraz w publikacjach [1] [14] [15].

3. Analiza działania systemu BIDS

Środowiskiem testowym stała się sieć lokalna jednej z firm działających w branży IT. Sieć ta była odseparowana przy pomocy bramy (gateway) od produkcyjnych systemów tej firmy. Brak zakłóceń w funkcjonowaniu tychże systemów był głównym warunkiem udostępnienia infrastruktury. Środowisko testowe składało się z następujących elementów:



Schemat środowiska testowego

- Główny serwer testowy, na którym została zainstalowana aplikacja BIDS. Parametry komputera: Pentium II 400 MHz, 128 MB RAM, Dysk IDE 10 GB.
- Pomocniczy serwer testowy, który będzie symulował ataki DoS/DDoS na serwer główny. Parametry

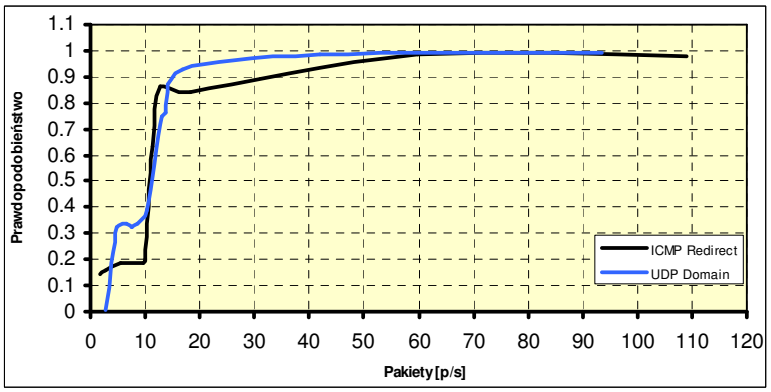
komputera: AMD XP 1400, 256 MB RAM, Dysk 40 GB.

- Przełącznik sieciowy 10/100 MBit 3COM
- Router Cisco, jego zadaniem było uniemożliwienie przedostania się generowanego ruchu do głównej sieci firmowej.
- System BIDS uruchomiony z logowaniem na poziomie 1 i 2 do pliku

Do generowania ruchu o pożądanym natężeniu i wielkości pakietów zostało wykorzystane narzędzie hping2 [10] oraz własne skrypty napisane w języku PERL i biblioteki Net::RawIP.

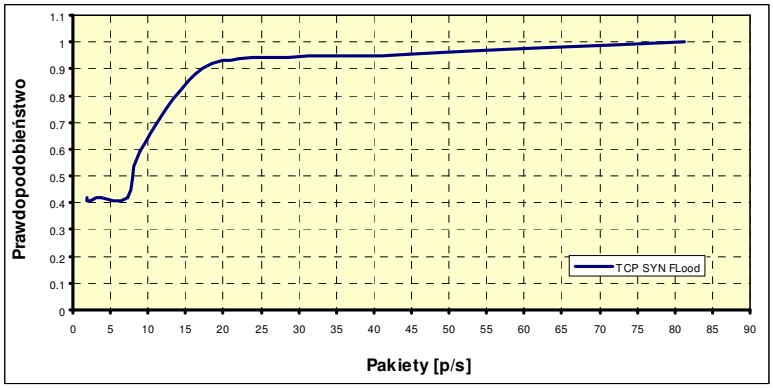
3.1. Testy skuteczności

Celem tych testów było zmierzenie skuteczności systemu, czyli jego zdolności do wykrywania ataków typu DoS/DDoS. W tym celu zasymulowano 3 rodzaje ataków, po jednym z każdego protokołu (ICMP, UDP, TCP). Siła ataku będzie regulowana przez zwiększanie dwóch parametrów:

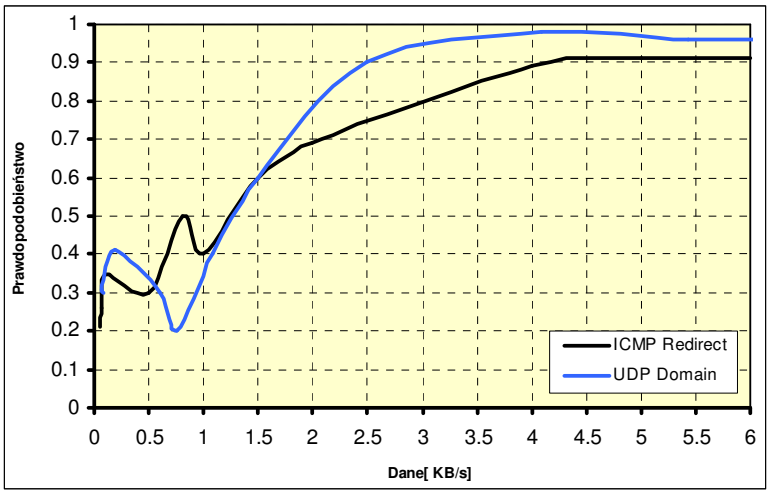


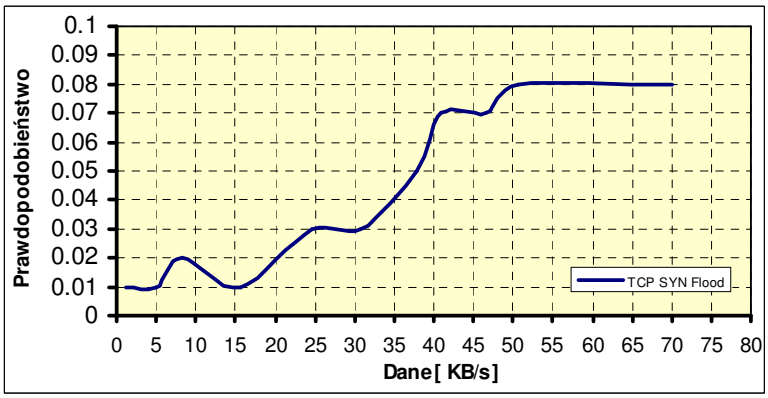
- Natężenia ilości pakietów [ilość/s]
- Natężenia ilości danych [KB/s]

Testy miały dać odpowiedź, jak system BIDS ocenia i klasyfikuje zwiększony ruch. Pierwsze trzy wykresy (dwa rysunki) przedstawiają zmianę prawdopodobieństwa w zależności od liczby pakietów. Na wszystkich z nich można zauważyć, że przy małych wartościach liczby pakietów (ok. 10-12) gwałtownie rośnie prawdopodobieństwo, osiągając szybko poziom 0,8-0,9. Wpływ na takie zachowanie systemu mają dwa czynniki. Po pierwsze średnia ilość pakietów zawarta w pliku konfiguracyjnym dla protokołów ICMP i UDP wynosi około 10-13. W tym właśnie przedziale zmienna przekazywana do sieci Bayesa (ilość pakietów/s) zmienia się z L2 na L3. Drugim czynnikiem, jest natomiast zawartość danych uczących. Dane oznaczone jako normalne miały wspomniany czynnik w większości ustawiony na wartości L0..L2, natomiast dane uczące ataków L3...L5. Ten gwałtowny wzrost nie obniża skuteczności systemu, o ile będzie on występował przy odpowiednich przedziałach. Administrator ma możliwość zmiany tych wartości.



W drugiej fazie tych testów, liczba pakietów była ustalana na wartość średnią dla danego protokołu, a zmianie ulegała ich wielkość, czyli wzrastała ilość danych docierających do serwera testowego. W przypadku pierwszych dwóch ataków (ICMP Redirect i UDP Domain Flood) wzrost natężenia danych pociągał za sobą wzrost prawdopodobieństwa. Można było zaobserwować, że wzrost ten nie jest jednostajny, są wartości, dla których prawdopodobieństwo powinno rosnąć, a mimo to maleje. Przyczyną tego zachowania jest nie do końca dobre ustalenie tablic prawdopodobieństw w sieci Bayesa, które nastąpiło w wyniku samodzielnego uczenia się. Trzeci z wykresów, dotyczący ataku TCP SYN Flood, znacząco różni się od pozostałych dwóch. Widać na nim, że wzrost natężenia danych prawie w ogóle nie wpływa na wzrost prawdopodobieństwa. Przy maksymalnej wartości, prawdopodobieństwo wynosi zaledwie 8%. Jest to jak najbardziej prawidłowa wartość, gdyż przy tego rodzaju ataku, agresorowi zależy na przesłaniu jak największej ilości pakietów i ich rozmiar powinien być jak najmniejszy.





Testy wykazały, że pomimo pewnych drobnych niedociągnięć system BIDS dobrze rozpoznaje ataki i może być skutecznych i przydatnych narzędziem do walki z agresorami. Uzyskanie lepszych wyników wiązałoby się prawdopodobnie z udoskonaleniem procesu uczenia (bardziej reprezentatywne próbki) oraz zmodyfikowaniem sieci Bayesa.

3.2. Testy wydajnościowe

Celem tych testów było zmierzenie obciążenia systemu, jakie powoduje system BIDS podczas największej powodzi, jaką uda się zasymulować. W tym teście będą badane dwa kluczowe parametry:

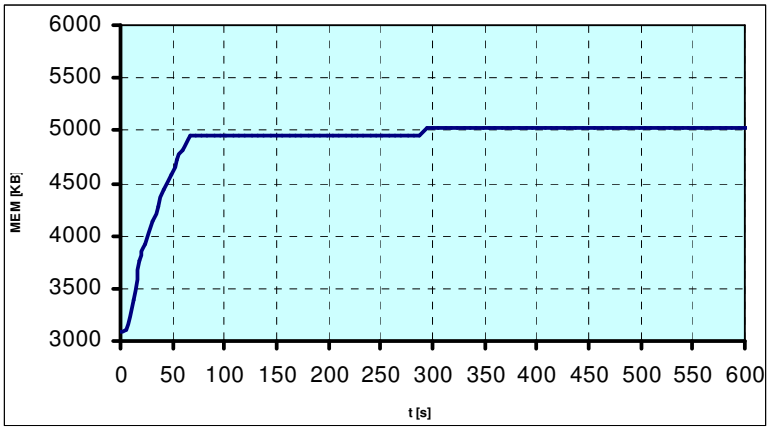
- Ilość zużytej pamięci [w kB] - CPU.
- Zajętość czasu procesora [%] - MEM.

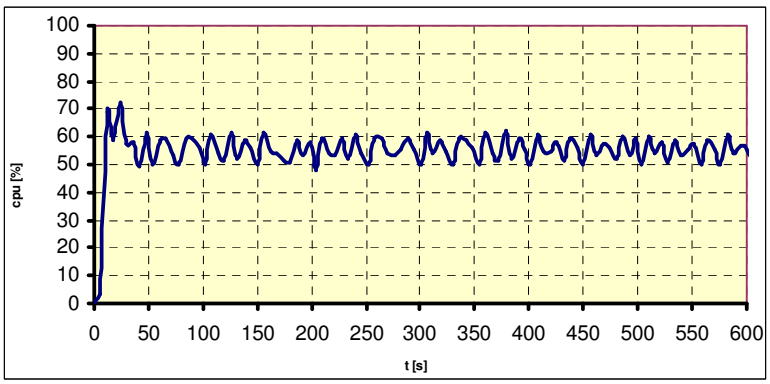
Działanie innych usług na testowanym serwerze została ograniczona do minimum, aby nie mogły one wpłynąć na wyniki. W tym teście serwer będzie zalewany pakietami TCP na przypadkowe usługi z przypadkowym adresem źródłowym oraz portem źródłowym. Czas pomiaru dla każdego z protokołów do 10 minut każdy. Wykresy obciążenia procesora i zużycia pamięci dotyczą typu procesu systemowego BIDS.

	ICMP	UDP	TCP
Czas pomiaru	576 s	546 s	711 s
Średnia ilość pakietów	571,77 p/s	420,23 p/s	370,72 p/s
Wartość średnia CPU	56,78 %	57,01 %	54,99 %
Wartość minimalna CPU	0 %	0 %	0 %
Wartość maksymalna CPU	77,88 %	74,86 %	75,96 %
Wartość średnia MEM	4603,05 KB	4561,89 KB	4899,42 KB
Wartość minimalna MEM	3072 KB	3084 KB	3088 KB
Wartość maksymalna MEM	4700 KB	4628 KB	5028 KB

Wykorzystanie zasobów komputera podczas testów wydajnościowych

Warunki przeprowadzonego testu miały na celu wywołanie maksymalnego obciążenia sytemu. W sytuacji, kiedy każdy z pakietu wysłanego do serwera miał losowy numer IP i port (źródłowy), BIDS był zmuszony do przydzielenia każdemu z nich pojedynczego rekordu w pamięci. Dlatego we wszystkich przypadkach w ciągu pierwszych 60 s następuję gwałtowny wzrost wykorzystanej pamięci. Po 60 s (czas życia rekordu), pierwsze alokowane rekordy straciły swoją ważność i zostały nadpisane przez nowe dane. Dynamika przydziału nowych bloków spadła praktycznie do zera i utrzymała się do końca testu. Ilość zaalokowanej pamięci wynosiła maksymalnie ok 2MB (5028 KB - 3088 KB = 1940 KB) w przypadku protokołu TCP i nieznacznie mniej w przypadku pozostałych (UDP, ICMP). Wynika to z prostej przyczyny, że pojedynczy rekord dla protokołu TCP zajmuje o kilkadziesiąt bajtów więcej niż rekordy dla ICMP i UDP.





Inaczej przedstawiało się wykorzystanie procesora. We wszystkich przypadkach, z początkowego zera (brak ataku), nastąpił gwałtowny wzrost i osiągnięcie maksymalnej wartości w ciągu pierwszych 10 s – ICMP 77,88%. Następnie wykorzystanie CPU spadło nieznaczne o około 15-20%. W dalszych fazach testu w przypadku wszystkich protokołów wykorzystanie procesora zmieniało się cyklicznie w czasie (raz malało, raz wzrastało), nie wychodząc jednak z widełek 50%-60%. Można przypuszczać, że przyczynami takiego zachowania mogło być zmiana strategii przydzielania czasu procesora przez system albo konieczność synchronizowania jednej z funkcji wewnątrz wątku. Ustalenie dokładnej przyczyny wymaga zastosowania profilera i jest dość trudne do przeprowadzenia przy takim obciążeniu.

Generalnie system przeszedł pozytywnie testy wydajnościowe. Pozytywnie należy ocenić bardzo niewielkie wykorzystanie pamięci (do 5 MB), co w przypadku dzisiejszych programów jest dobrą wielkością. Nieco gorzej aplikacja radzi sobie z obciążaniem procesora – jest ono trochę za wysokie, ale mieszczące się w granicach rozsądku. Z pewnością pomogłoby dokładne przeanalizowanie całego kodu aplikacji i jego optymalizacja. Również zastąpienie sekwencyjnego przeszukiwania tablicy rekordów poprzez przeszukiwanie uporządkowanego drzewa znacząco poprawiłoby wyniki. W dalszej kolejności można by rozważyć własną, wysoce zoptymalizowaną implementacje sieci Bayesa.

4. Podsumowanie

W ostatnich latach ataki typu odmowa usługi (DoS, DDoS) stały się prawdziwą plagą. Ofiarami ich stają się największe firmy z branży nowych technologii: Yahoo, eBay, Amazon (2000), Microsoft (2003 i 2004), SCO (2004). Pomimo, że firmy te dysponują ogromnymi środkami, nie powstrzymały zmasowanych ataków na ich serwery. Problem z atakami typu DDoS polega na tym, że nie ma „złotego środka" zaradczego. System wykrywania ataków DoS/DDoS - BIDS (*Bayesian Intrusion Detection System*) jest krokiem na przód w tej walce. Aplikacja ta koncentruje się na wykrywaniu anomalii w natężeniu ruchu i jest szczególnie czuła na zwiększony ruch w wybranych protokołach. Założeniem projektowym było, aby system wykrywał nie tylko kilka określonych ataków, ale również potrafił wykryć te nieznane. Poprzez odpowiednio dobrany proces uczenia sieci Bayesa cel został osiągnięty. Pewną obawą było, czy system BIDS będzie w stanie zanalizować w czasie rzeczywistym wszystkie dane docierające do niego podczas ataku. Testy wykazały jednak, że obciążenie procesora i wykorzystanie pamięci mieści się w bezpiecznych granicach. Zasadnicze testy dotyczące skuteczności wykrywania ataków, również wypadły dobrze – system poprawnie wykrywa zwiększone natężenie przesyłanych pakietów i odpowiednio klasyfikuje rodzaj ataku. Wykorzystanie sieci Bayesa, jako silnika do analizy przechwyconych okazało się bardzo dobrym pomysłem. Ogromną jej zaletą jest szybkość (kluczowy parametr) oraz możliwość uczenia się poprzez zadanie odpowiednich przykładów zachowania sieci.

5. LITERATURA

[1] *Bayesian Learning*, *http://www.andrew.cmu.edu/user/dgovinda/pdf/bayes.pdf*.

[2] BODZIANOWSKI Łukasz: Bezpieczeństwo systemów komputerowych, *http://www.bsk.konin.lm.pl/wstep.html*.

[3] CASWELL Brian, ROESCH Marty: *Snort. The Open Source Network Intrusion Detection System*, Sourcefire, INC. 2004, *http://www.snort.org/*.

[4] *Cisco - Cisco Intrusion Detection*, Cisco Systems, Inc, 2004 *http://www.cisco.com/warp/public/cc/pd/sqsw/sqidsz/*.

[5] CZARNOWSKI Aleksander: *DDoS - nowa postać starego ataku*, PC Kurier 5/2000, *http://www.pckurier.pl/archiwum/art0.asp?ID=3873*.

[6] DOROSZ Piotr., KAZIENKO Przemysław.: *Systemy wykrywania intruzów*. VI Krajowa Konferencja Zastosowań Kryptografii ENIGMA 2002, Warszawa 14-17 maja 2002 r., s. TIV 47-78.

[7] DOROSZ Piotr, KAZIENKO Przemysław: *IDS - systemy wykrywania włamań - cz. I. Rodzaje i objawy włamań. Zadania i budowa systemów IDS.* IT FAQ 12/2002, s. 21-27. *Cz. II. Klasyfikacja, metody i techniki.* IT FAQ 1/2003, s. 17-22. *Cz. 3. Reagowanie na włamania i kierunki rozwoju. Komercyjne systemy IDS.* IT FAQ, 2/2003, s. 30-34.

[8] *Email Bombing and Spamming*, CERT Coordination Center, 1999-2002 *http://www.cert.org/tech_tips/email_bombing_spamming.html*.

[9] FERGUSON Paul: *Denial of Service (DoS) Attack Resource Page*, DTS, 2000, *http://www.denialinfo.com/*.

[10] *HPING*, Salvatore Sanfilippo, 2004, *http://www.hping.org/*.

[11] *Internet Protocol*, University of Southern California, 1981, *http://www.ietf.org/rfc/rfc0791.txt*.

[12] *Open Systems Interconnection (OSI) Protocols*, Cisco Systems, Inc, 2003, *http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/osi_prot.htm*.

[13] Norsys Software Corp., 2004, *http://www.norsys.com/*.

[14] Radford Neal: *What is Bayesian Learning*, 2004, *http://www.faqs.org/faqs/ai-faq/neural-nets/part3/section-7.html*.

[15] ROTH Dan: *Bayesian Learning*, *http://l2r.cs.uiuc.edu/~danr/Teaching/CS346-04/Lectures/10-LecBayes-NB4.pdf*.

[16] *Systemy operacyjne*, Ranking.PL, Gemius SA, *http://www.ranking.pl/rank.php?stat=sysoperAL*.

[17] ŻURAKOWSKI Marcin: *Obrona przed Atakami typu odmowa usługi (DoS). Praca magisterska*, Wydział Informatyki i Zarządzania, Politechnika Wrocławska, 2004, *http://inet4u.esol.pl/opatou-dos.pdf*.