Warszawa, maj 2018 r.

Politechnika Warszawska
Wydział Elektroniki i Technik Informacyjnych

Instrukcja do zajęć laboratoryjnych z przedmiotu Bezpieczeństwo Systemów i Sieci nt. Sieciowe systemy wykrywania włamań (*network-based IDS*)

Opracował: dr inż. Krzysztof Cabaj

Uzupełniali: mgr inż. Paweł Radziszewski i dr inż. Jacek Wytrębowicz

Spis treści

1 Systemy wykrywania włamań

Zwiększenie się liczby oraz roli komputerów podłączonych do sieci Internet spowodowało wzrost wymagań związanych z bezpieczeństwem. Pierwsze zmiany polegały na uruchomieniu najprostszych mechanizmów systemowych, takich jak dostęp do wybranych usług na hasło czy tylko z wybranych adresów. Kolejnym krokiem było wprowadzanie list kontroli dostępu na routerach czy specjalizowanych systemów zapór sieciowych. Oba mechanizmy służą do filtracji ruchu w taki sposób, aby ruch niechciany nie dochodził do chronionych maszyn. Wymienione mechanizmy zabezpieczają użytkowników przed pewnymi niepożądanymi zachowaniami. Jednak jeśli osoba czy maszyna uznawana za zaufaną, mająca prawo komunikować się lub wykorzystywać pewne zasoby, będzie próbowała dokonać niepowołanych działań, to nie zostanie to zaobserwowane ani zablokowane. W systemach komputerowych pojawiło się zapotrzebowanie na oprogramowanie monitorujące działanie sieci i systemów komputerowych w celu wykrywania tego typu działań, czyli systemów wykrywania włamań.

Systemy wykrywania włamań (*Instrusion Detection Systems*, IDS) są oprogramowaniem umożliwiającym wykrywanie niepożądanych działań w systemach komputerowych. Oprogramowanie tego typu rozwijane jest od początku lat 80-tych, kiedy pojawiły się pierwsze prace zwracające uwagę na ryzyko, jakim mogą być przejęcia systemów komputerowych przez nieuprawnione osoby. Jednak dopiero ostatnie lata są związane z wzmożonym zainteresowaniem i rozwijaniem tego typu oprogramowania.

Głównym zadaniem systemów wykrywania włamań jest monitorowanie tego, co dzieje się w chronionym systemie. IDS może zarówno chronić pojedynczy system komputerowy, jak i korporacyjną sieć, składającą się z tysięcy stacji roboczych i serwerów. Na podstawie informacji docierających ze środowiska zewnętrznego, oraz informacji na temat zakazanych zdarzeń, generowane są alarmy. Niektóre systemy umożliwiają także automatyczną reakcję na wykryte zdarzenie np. zakończenie podejrzanego połączenia, czy zmiana reguł zapory sieciowej – przez niektórych autorów systemy tego typu kwalifikowane są do oddzielnej klasy systemów zapobiegania włamaniom (*Intrusion Prevention Systems*, IPS). Kolejnym ważnym zadaniem systemów IDS jest zbieranie jak największej liczby informacji na temat działalności włamywacza. W tym zbieranie informacji, które mogą zidentyfikować atakującego (adres IP maszyny, czas, wykorzystana technika) oraz sprawdzić, jaką działalność podjął w zaatakowanym systemie (jakich zasobów używał, jakie zmiany wprowadził do systemu). Uzyskane informacje mogą posłużyć ujęciu atakującego, pomóc w ewentualnym postępowaniu sądowym, oraz zorientować się jakie są słabe punkty systemu i jakie dane dostały się w ręce atakującego.

Przedstawiony powyżej opis jest bardzo ogólny. Systemy IDS można podzielić na różne klasy ze względu na wiele cech, jakie posiadają. Najważniejsze podziały dotyczą rodzaju danych używanych do wykrywania niepowołanych czy też podejrzanych zdarzeń oraz sposobu podejmowania decyzji świadczących o wystąpieniu naruszenia.

1.1 Dane wykorzystywane do wykrywania naruszeń

Systemy IDS mogą pobierać do analizy dane z dwóch głównych źródeł:

1) systemu operacyjnego – systemy stacyjne zwane żargonowo hostowymi (*Host-based IDS*, HIDS);
2) całego, surowego ruchu sieciowego – systemy sieciowe (*Network-based IDS*, NIDS).

Systemy HIDS bazują na zdarzeniach systemowych, np. logowanie użytkownika, uruchomienie określonego programu czy dostęp do pewnego pliku. Dane do sensora mogą być dostarczane bezpośrednio z systemu operacyjnego czy monitorowanego programu lub pośrednio poprzez analizę logów systemowych. Na podstawie dostarczanych zdarzeń podejmowana jest decyzja: czy jest to dozwolone zachowanie, czy trzeba zgłosić naruszenie. Pierwsze systemy IDS działały właśnie w ten sposób. Jednym z pierwszych tego typu systemów opisanych w literaturze był IDES.

Systemy NIDS analizują ruch sieciowy, dochodzący do danej maszyny. Obserwują, jakie dane są transmitowane przez sieć i w razie wykrycia podejrzanego zachowania, np. niepoprawnego pakietu czy określonego ciągu znaków wymienianego pomiędzy maszynami, generują alarm. Systemy tego typu rozwinęły się wraz z rozpowszechnieniem się sieci komputerowych. Dużą zaletą tego typu systemów jest to, że pojedynczy sensor umieszczony na przełączniku lub routerze może obserwować/chronić wiele maszyn. Nie ma potrzeby instalowania na każdej maszynie oddzielnego sensora, jak ma to miejsce w przypadku systemów HIDS. Przykładem tego typu systemu jest snort.

W pewnym momencie uważano, że systemy NIDS to rozwiązanie idealne. Jednak zmiany w technologiach sieciowych, takie jak zwiększenie szybkości oraz stosowanie sieci przełączanych, utrudniły stosowanie jednego sensora sieciowego do chronienia całej podsieci. Innym ograniczeniem dla systemów NIDS jest wprowadzenia szyfrowanych protokołów, np. SSH czy HTTPS. Zastosowanie tego typu zabezpieczenia przed podsłuchem uniemożliwia systemowi IDS analizowanie przesyłanych danych w poszukiwaniu niepożądanych zachowań.

Oba rodzaje systemów IDS posiadają swoje słabe i mocne strony. Dlatego kolejnym etapem ich rozwoju było zbudowanie rozwiązań hybrydowych – tj. produktów, które łączyły w sobie sensory stacyjne jak i sieciowe, współpracujące ze sobą. Tak działa większość systemów komercyjnych, m.in. ISS RealSecure czy Enterasys Dragon, jak i darmowy Prelude.

1.2 Sposób wykrywania naruszeń

W poprzednim punkcie zostały opisane sposoby uzyskiwania danych dostarczanych do analizy. Mając dane należy podjąć decyzję o zakwalifikowaniu ich do zachowań dozwolonych lub niedozwolonych. Na podstawie sposobu podjęcia tej decyzji systemy dzielimy na wykrywające anomalie i wykrywające nadużycia (czasami zwane systemami z sygnaturami czy systemami sygnaturowymi).

Systemy oparte o wykrywanie anomalii generują alarm, jeśli jeden z obserwowanych parametrów zaczyna odbiegać od normy. Przykładem tego typu zachowania jest np. zwiększenie liczby następujących po sobie nieudanych prób logowania na jedno z kont, czy nagły wzrost ruchu sieciowego pewnego protokołu. Dużą zaletą tego typu systemów jest ich samoczynne dostosowywanie się do nowych warunków. Systemy tego typu umożliwiają wykrywanie ataków nawet jeśli atakujący zastosują drobne modyfikacje w sposobie przeprowadzania ataków. Jednak zaleta ta może być wykorzystana przez atakującego, który powoli zmienia swoje zachowanie w taki sposób, że zostaje to niezauważone przez system. Na zasadzie wykrywania anomalii działał system IDES.

Systemy wykrywające naruszenia działają na zasadzie porównywania obserwowanych zdarzeń z bazą zawierającą niedozwolone zachowania. Jeśli zostanie wykryte zdarzenie zakwalifikowane wcześniej do naruszeń, to zostaje wygenerowany alarm. Poważną wadą tych systemów jest konieczność stworzenia i uaktualniania dużej bazy sygnatur. System nie wykryje nowego ataku, nawet w wypadku, gdy jest on modyfikacją wcześniejszego, bez dodatkowej sygnatury. Zaletą systemów działających w ten sposób jest szybkość oraz możliwość zastosowania rozwiązań sprzętowych (sygnatury w systemach sieciowych często sprowadzają się do porównywania pewnych atrybutów pakietów i ich zawartości).

2 Wprowadzenie do narzędzi

Celem niniejszych zajęć laboratoryjnych jest zapoznanie się z sieciowymi systemami wykrywania włamań. Jakakolwiek praca z tego typu oprogramowaniem nie może być wykonywana bez znajomości działania mechanizmów sieciowych, protokołów komunikacyjnych oraz najczęściej obserwowanych schematów wymiany komunikatów. Dlatego w kolejnym rozdziale rozpoczniemy od prezentacji programu `tcpdump`, umożliwiającego obserwację i analizę ruchu sieciowego. Zostaną też przypomniane informacje z zakresu sieci komputerowych, które mogą być przydane przy analizie bezpieczeństwa sieciowego. Będą omówione filtry `tcpdump` umożliwiające obserwację jedynie wybranego, interesującego nas ruchu sieciowego. Przy pomocy filtrów można dokonać ręcznej analizy, czy nie doszło do naruszeń w chronionej sieci. W następnych rozdziałach przedstawione będą programy `nmap` i `snort`. Pierwszy z nich pozwala zdobywać informacje na temat interesujących nas maszyn, zaś drugi jest jednym z najpopularniejszych systemów wykrywania włamań.

2.1 tcpdump

`Tcpdump` jest programem umożliwiającym obserwację i analizę ruchu sieciowego [Tcpdump]. Program po uruchomieniu przełącza interfejs sieciowy w tryb *promiscuous*, w którym możliwe jest odbieranie wszystkich ramek zaobserwowanych przez interfejs. W normalnym trybie pracy interfejsu sieciowego do systemu operacyjnego przekazywane są jedynie ramki skierowane do niego (na jego adres MAC), ramki rozgłoszeniowe i ramki *multicast* adresowane do grup, do których interfejs jest zapisany. Dlatego program `tcpdump` można uruchomić jedynie na koncie z uprawnieniami administratora. Jeśli jednak komputery są

połączone poprzez przełącznik, to nawet po przełączeniu interfejsu w tryb *promiscuous* możliwa będzie obserwacja jedynie ruchu skierowanego do danej stacji.

Po uruchomieniu programu i odebraniu kilku ramek, na ekran powinny zostać wypisane napisy podobne do poniższych:

```
(1)  12:12:54.962219 arp who-has 172.16.0.55 tell 172.16.0.1
(2)  12:12:54.962237 arp reply 172.16.0.55 is-at 0:a:e4:44:30:30
(3)  12:13:33.234537 172.16.0.1.netbios-ns > 172.16.255.255.netbios-ns: NBT UDP
     PACKET (137): QUERY; REQUEST; BROADCAST
(4)  12:13:33.982369 172.16.0.1.netbios-ns > 172.16.255.255.netbios-ns:NBT UDP
     PACKET (137): QUERY; REQUEST; BROADCAST
(5)  12:13:34.733372 172.16.0.1.netbios-ns > 172.16.255.255.netbios-ns:NBT UDP
     PACKET (137): QUERY; REQUEST; BROADCAST
(6)  12:14:00.545415 172.16.0.1 > 172.16.0.55: icmp: echo request
(7)  12:14:00.545460 172.16.0.55 > 172.16.0.1: icmp: echo reply
(8)  12:14:02.772596 172.16.0.55.32769 > 172.32.0.2.domain: 21540+ A?host1.net.pl.
     (30) (DF)
(9)  12:14:08.010114 172.16.0.1.1775 > 172.16.0.55.http: S 5220202:5220202 (0) win
     65535 <mss 1460,nop,nop,sackOK> (DF)
(10)   12:14:08.010174 172.16.0.55.http > 172.16.0.1.1775: S 2822615422:2822615422
     (0) ack 5220203 win 5840 <mss1460,nop,nop,sackOK> (DF)
(11)   12:14:08.010324 172.16.0.1.1775 > 172.16.0.55.http: . ack 1 win 65535 (DF)
(12) 12:14:08.032110 172.16.0.1.1775 > 172.16.0.55.http: P 1:400(399) ack1 win 65535
     (DF)
(13) 12:14:08.032138 172.16.0.55.http > 172.16.0.1.1775: . ack 400 win6432 (DF)
(14)   12:14:08.033191 172.16.0.55.http > 172.16.0.1.1775: . 1:1461(1460) ack 400
     win 6432 (DF)
(15)   12:14:08.033209 172.16.0.55.http > 172.16.0.1.1775: . 1461:2921(1460) ack 400
     win 6432 (DF)
(16)   12:14:41.608525 172.16.0.55.32771 > 172.30.0.15.33435: udp 10 [ttl 1]
(17) 12:14:46.604826 172.16.0.55.32771 > 172.30.0.15.33436: udp 10 [ttl 1]
(18) 12:14:51.604819 172.16.0.55.32771 > 172.30.0.15.33437: udp 10 [ttl 1]
(19) 12:14:56.604843 172.16.0.55.32771 > 172.30.0.15.33438: udp 10
```

Każda linijka reprezentuje jedną odebraną ramkę (uwaga: niektóre linijki są „zawinięte” i kończą się w następnej linii.). Na początku znajduje się czas odebrania ramki, następnie tekst dokładniej opisujący ramkę. W zależności od protokołu dalsza część linijki i prezentowane informacje różnią się. Poniżej przedstawione są krótkie interpretacje wybranych linijek, objaśniające jakie informacje można uzyskać dla wybranych najpopularniejszych protokołów.

2.1.1 ARP (*Address Resolution Protocol*)

Protokół ARP umożliwia poznanie adresu MAC interfejsu sieciowego maszyny podłączonej do sieci lokalnej, której adres IP jest znany. Linie (1) i (2) pokazują, jak wygląda zapytanie oraz odpowiedź w tym protokole. W linii (1) maszyna o adresie IP 172.16.0.1 wysyła zapytanie o adres MAC maszyny o adresie IP 172.16.0.55. W linii (2) znajduje się odpowiedź – maszyna o adresie 172.16.0.55 podaje swój adres MAC. Obserwując wymianę komunikatów protokołu ARP można poznać adresy maszyn podłączonych do obserwowanej sieci lokalnej.

2.1.2 ICMP (*Internet Control Message Protocol*)

Protokół ICMP jest odpowiedzialny za dostarczenie informacji diagnostycznych dla stosu protokołów TCP/IP. W liniach (6) i (7) zaprezentowana jest przykładowa wymiana komunikatów. Pierwsze informacje (znajdujące się po czasie odebrania ramki) dotyczą adresów maszyn, które się ze sobą komunikują. W pierwszym komunikacie maszyna 172.16.0.1 wysyła zapytanie do maszyny o adresie 172.16.0.55. W dalszej części tej linii widać, że jest to pakiet ICMP typu *echo request*. W kolejnej linii widać odpowiedź maszyny 172.16.0.55 – pakiet ICMP typu *echo reply*. W zależności od typu pakietów wypisywany jest jego typ i ewentualnie kod. Więcej informacji na temat protokołu ICMP można znaleźć w dokumencie RFC 792. Zaprezentowana wymiana komunikatów w liniach (6) i (7) została spowodowana użyciem programu `ping`.

2.1.3 UDP (*User Datagram Protocol*)

UDP jest jednym z protokołów warstwy transportowej. W przeciwieństwie do TCP, jest on bardzo prosty, umożliwia przesyłanie danych bez możliwości potwierdzeń, odbudowywania kolejności danych w strumieniu, itp. Tak jak w przypadku protokołu ICMP, po czasie zarejestrowania pakietu, prezentowane są adresy IP. W linii (16) widać, że pakiety są wysyłane z maszyny o adresie 172.16.0.55 do maszyny posiadającej adres 172.30.0.15. Dodatkowo po każdym adresie pojawia się po kropce liczba, reprezentująca odpowiednio port źródłowy i docelowy. W niektórych przypadkach, jeżeli port jest jednoznacznie identyfikowany z usługą (*Well Known Ports*), to wypisywana jest symboliczna nazwa. Taką interpretację widzimy dla pakietów z linii (3)-(5) (`netbios-ns`) czy pakietu z linii (8) (`domain` – datagram protokołu DNS). Po danych dotyczących adresów i portów znajdują się informacje o tym, że jest to pakiet protokołu UDP oraz dodatkowe informacje. W prezentowanej linii (16) jest to wielkość pakietu i wartość niesiona w polu TTL, czyli czas życia pakietu. Pakiety zaprezentowane w liniach od (16) do (19) zostały wysłane przez program `traceroute`, służący do znajdowania trasy pakietów przesyłanych w sieci IP. Więcej informacji na temat protokołu UDP można znaleźć w dokumencie RFC 768.

2.1.4 TCP (*Transport Control Protocol*)

TCP to protokół warstwy transportowej. Jednak w porównaniu z UDP jest dużo bardziej skomplikowany. Spowodowane to jest tym, że TCP zapewnia stworzenia wirtualnego połączenia pomiędzy dwiema maszynami (dokładniej – dwoma programami działającymi na tych maszynach). Połączenie to zapewnia potwierdzenia odbioru, odbudowę kolejności strumienia danych i sterowanie przepływem (tj. ochronę przed gubieniem pakietów w efekcie przepełnienie bufora). Z tego powodu ilość danych prezentowana przez program `tcpdump` związana z tym protokołem jest dużo większa niż opisywanych wcześniej. Po dacie zarejestrowania pakietu następują informacje identyfikujące adresy i porty komunikujących się programów (konwencja jest identyczna jak w przypadku protokołu UDP). Następnie znajduje się informacja o ustawionych znacznikach. Może tutaj znaleźć się znak '.' oznaczający brak ustawionych znaczników, albo dowolna kombinacja liter S, F, P, R wskazujących ustawione znaczniki. Przykładowo, pakiet w linii (9), nazywany segmentem

protokołu TCP, ma ustawiony znacznik SYN (S), a w linii (12) znacznik PUSH (P). W dalszej części linii znajdują się wartości numerów sekwencyjnych bajtów przenoszonych przez dany segment (numer pierwszego bajtu i ostatniego powiększony o 1) oddzielone dwukropkiem i podaną w nawiasach liczbą tych bajtów. Numery sekwencyjne mogą być podawane na dwa sposoby: wartości bezwzględne tak jak są przesyłane w segmentach – linie (9) i (10), albo wartości względne liczone od 0 – jak w dalszych liniach. Z numerami sekwencyjnymi wiąże się także numer potwierdzający odebranie porcji danych. Numer potwierdzonego bajtu jest podawany po napisie ack. I tak, dla segmentu z linii (14) widać, że maszyna o adresie 172.16.0.55 z aplikacji o porcie 80 (HTTP) przesyła 1460 bajtów do maszyny 172.16.0.1 i jej aplikacji skojarzonej z portem 1775, oraz potwierdza odebranie 400 bajtów. Po napisie „win" znajduje się aktualna wielkość okna odbioru strony wysyłającej dane. W nawiasach trójkątnych znajdują się różne opcje związane z protokołem TCP. Opcje te są zwykle przesyłane we wstępnej fazie połączenia. Na ich podstawie można próbować identyfikować, jaki system operacyjny jest używany na zdalnej maszynie.

2.1.5 Inne protokoły

Program `tcpdump` umożliwia dokładną analizę także protokołów warstw wyższych. Przykładowo w liniach (3)-(5) widać analizę pakietów protokołu NetBIOS, za pomocą którego komunikują się maszyny z systemem Microsoft Windows. W linii (8) widać informacje, jakie program `tcpdump` odczytał z pakietu protokołu DNS. W tym przypadku jest to zapytanie o adres maszyny o nazwie `host1` w domenie `net.pl`. Zapytanie jest przekazywane do serwera DNS działającego pod adresem 172.32.0.2.

2.1.6 Przydatne przełączniki tcpdump

W zaprezentowanym powyżej raporcie `tcpdump` adresy IP były podane w formie dziesiętnokropkowej. Jeśli to możliwe, program `tcpdump` stara się wypisać tutaj nazwę hosta. Przełącznik –n powoduje, że `tcpdump` nie będzie pobierał nazwy hosta i zawsze adresy podawał będzie w notacji dziesiętnokropkowej. Przełącznik –nn ma analogiczne znaczenie w stosunku do nazw usług powiązanych z określonymi portami. Bez tego przełącznika, jeśli to możliwe, zamiast numeru portu wypisywana jest jego dobrze znana nazwa z pliku `/etc/services`.

Uruchomienie programu bez podawania żadnych przełączników powoduje wypisanie na ekranie wszystkich odebranych ramek. W większości wypadków okazuję się, że ilość danych jest tak duża, że niemożliwe jest ich analizowanie na bieżąco. Dlatego program `tcpdump` umożliwia zapisywanie i późniejszą analizę danych. Służą do tego opcje –w i –r. Pierwsza powoduje zapisanie, druga odczytanie danych z pliku. Inne przydatne opcje to m.in:

-l	(*line-buffered*) natychmiastowe wypisywanie każdej linii, bez buforowania;
-v, -vv, -vvv	(*verbosity*) poziomy szczegółowości wyjścia;
>	przekierowanie standardowego wyjścia przez program powłoki (nie mylić z binarnym zapisem ruchu (-w)).

2.1.7 Filtry

Najważniejszą zaletą programu `tcpdump` jest możliwość zastosowania filtracji obserwowanego czy zapisanego wcześniej ruchu [Pcap]. Idea filtrów polega na budowaniu wyrażeń logicznych dotyczących zawartości ramek. Jeśli dla danej ramki wyrażenie jest prawdziwe, to jest ona wypisywana na ekran lub zapisywana do pliku. Do budowy wyrażeń logicznych używane są specjalne „prymitywy", służące do porównywania interesujących cech pakietów. Jeśli interesują nas wybrane protokoły, możemy wyspecyfikować np. arp, icmp, tcp czy udp. Można także dokonać bardziej szczegółowego wyboru, przykładowo, jeśli chcemy słuchać jedynie ruchu związanego z określoną maszyną możemy skorzystać z prymitywów „dst host", „src host" czy „host". Wyrażenie „dst host 172.15.0.1" wypisze tylko takie pakiety które są wysyłane do maszyny o adresie 172.15.0.1. Wyrażenie „host" będzie prawdziwe, gdy jeden z adresów będzie odpowiadał podanemu. W podobny sposób można wyspecyfikować rejestrowane pakiety na podstawie numerów portów. Przykładowe wyrażenia to „src port 53" czy „dst port 80". Pierwsze wypisze tylko pakiety, które są wysłane z serwera DNS, a drugie dotyczące ruchu WWW. Ponieważ są to wyrażenia, można używać operatorów negacji (! lub not), iloczynu logicznego (&& lub and) lub sumy logicznej (|| lub or). Korzystając z nich można budować bardziej rozbudowane wyrażenia, np:

```
„icmp or (udp and dst port 53) or (tcp and port http and host www.elka.pw.edu.pl)"
```

Filtr ten wypisuje pakiety protokołu ICMP, wysyłane zapytania DNS oraz transmisje protokołu HTTP do maszyny o adresie www.elka.pw.edu.pl.

Więcej informacji na temat budowy zaawansowanych filtrów można znaleźć w dokumentacji do programu `tcpdump`. W ramach przygotowania do laboratorium należy zapoznać się ze stosowną jej częścią.

Za pomocą nagranego ruchu oraz filtrów programu `tcpdump` można dokonać prostego sprawdzenia, czy w obserwowanej sieci nie doszło do pewnych naruszeń. Bardzo często do różnego typu skanowań czy komunikacji pomiędzy skompromitowanymi maszynami używany jest protokół DNS. W normalnie działającej sieci cały ruch DNS powinien być kierowany do lokalnego serwera DNS. Aby wykryć za pomocą `tcpdump` sytuację gdy pojawia się zapytanie do zewnętrznego serwera DNS, można nagrać ruch i przefiltrować go za pomocą wyrażenia:

```
udp and port 53 and (not host lokalny.server.dns)
```

Wszystkie wybrane pakiety są podejrzane i wymagają dalszej analizy. W bezpieczeństwie sieciowym zawsze należy sprawdzić potencjalne zgłoszone naruszenia, choć może okazać się, że są one spowodowane legalnym działaniem. W takiej sytuacji mówilibyśmy o alarmie typu *false positive* – czyli legalnej działalności użytkownika, która wywołała alarm, np. mogło się okazać, że na maszynie, która została wykryta, ktoś omyłkowo zmienił adres serwera DNS.

Oczywiście nie jest to najprostsza i najelastyczniejsza metoda chronienia sieci, jednak umożliwia ona wyłapywanie pewnych potencjalnie podejrzanych przypadków. Jeden z systemów IDS – Shadow działa właśnie w ten sposób. Ruch z chronionej sieci zostaje

zapisany za pomocą programu `tcpdump`. Później jest on analizowany za pomocą filtrów pod kątem różnego typu naruszeń.

2.2 nmap

`Nmap` jest skanerem, czyli programem umożliwiającym dokonywanie przeglądu jakie usługi działają na zdalnej maszynie [Nmap]. Głównym zadaniem tego programu jest przeprowadzanie skanowania działających maszyn w określonej podsieci oraz dokładne sprawdzenie, które porty są na nich otwarte. Program pozwala skanować porty TCP oraz UDP. Do wyboru mamy kilka metod skanowania: od prostych skanowań połączeniowych, poprzez skanowania wykorzystujące specyfikę nawiązywania połączeń, aż po skanowania utajnione. Ponadto program umożliwia podjęcie próby rozpoznania systemu operacyjnego na zdalnej maszynie. W tym celu stosowana jest technika *fingerprinting*. Polega ona na wysyłaniu niestandardowych pakietów i analizowaniu odpowiedzi. Ponieważ w dokumentach RFC, w których opisane są protokoły sieciowe, zostawiano dużą swobodę dla ich implementacji, rozpoznawanie systemów może być precyzyjne. Na przykład w przypadku systemu Linux podawane są nawet prawdopodobne numery jąder systemu operacyjnego. Funkcja ta pozwala bardzo szybko zinwentaryzować aktualnie działające w sieci systemy. Program początkowo powstał w środowisku uniksowym. Aktualnie dostępny jest także w systemach MS Windows. Dostępne są też wersje z graficznym interfejsem pozwalającym skonfigurować opcje skanowania. Dostępny jest także serwis, umożliwiający wykonywanie skanowań cyklicznych lub o określonej porze. Opcja ta pozwala w bardzo prosty sposób kontrolować zmiany na wybranych maszynach.

Nie zawsze zachodzi potrzeba skanowania wszystkich portów maszyny. Do wyspecyfikowania tylko wybranych portów można użyć przełącznika -p, wpisując interesujące numery portów czy ich zakresy, np.: „-p 1-1024,2043,60000-65535”.

Za pomocą przełączników -s... można wybrać różne metody skanowania (np. -sF, -sU). Metody te wykorzystują różne techniki, najczęściej w celu utrudnienia wykrycia procesu skanowania czy obejścia źle skonfigurowanych urządzeń sieciowych.

2.3 snort

Program `snort` jest sieciowym sygnaturowym systemem wykrywania włamań [Snort]. Może być on również używany jako *sniffer*, czy jako program logujący nasłuchiwane dane. Jednak te dwie funkcje nie są z naszego punktu widzenia interesujące – podobne funkcje ma program `tcpdump`. Po uruchomieniu programu w trybie systemu wykrywania włamań, zostają wczytane sygnatury opisujące ruch, który zdaniem użytkownika jest niepożądany. W tej kategorii nie muszą znajdować się tylko i wyłącznie ataki. Można wykrywać także inne działania, które nie są atakami, ale z punktu widzenia polityki bezpieczeństwa danej instytucji są niepożądane. Dla przykładu, można za pomocą własnych sygnatur wykrywać próby dostępu do stron zawierających niepożądane treści, wysyłanie pewnych materiałów na zewnątrz, czy używanie niepożądanych programów.

2.3.1 Sygnatury

`Snort` jest systemem wykrywającym naruszenia bezpieczeństwa, co oznacza że wszystkie możliwe do wykrycia nadużycia muszą być w jakiś sposób opisane. Każde potencjalne nadużycie w systemie jest opisywana przez sygnaturę. System zapisu reguł jest bardzo elastyczny i dość popularny. Część komercyjnych systemów wykrywania włamań jest w stanie wczytać sygnatury zapisane w ten sposób. Przykładem jest moduł `Trons` do systemów firmy ISS. Standardowa dystrybucja zawiera zestaw kilku tysięcy sygnatur wykrywających znane ataki. Oprócz samych ataków znajdują się tutaj sygnatury rozpoznające np. podejrzane zachowanie, które może świadczyć o próbach zdobywania pewnych informacji na temat chronionej sieci. Jednak największą zaletą `snort` jest możliwość tworzenia własnych sygnatur. Sygnaturę `snort` można porównać do filtru nakładanego na ruch obserwowany przez `tcpdump`. Jeśli w sieci zostanie zaobserwowany pakiet, który pasuje do sygnatury, to zostaje generowany i logowany alarm. Reguły `snorta` mogą być dużo bardziej skomplikowane niż filtry `tcpdump`. Przykładowo można sprawdzać zawartość pakietów w poszukiwaniu pewnych ciągów świadczących o wystąpieniu ataku. Przed dokonywaniem porównania może być wykonanych także kilka innych operacji, mających zapobiec np. atakom polegającym na próbie oszukania systemu wykrywania włamań. Przykładowo program odbudowuje strumień TCP, czy dokonuje złożenia sfragmentowanych pakietów. Dzięki temu nie uda się próba przeprowadzania znanego ataku w taki sposób, aby informacje świadczące o ataku zostały wysłane w innej kolejności i niezaobserwowane bezpośrednio w sieci. Więcej informacji na ten temat można znaleźć w pracy [Ptacek].

Sygnaturę można podzielić na trzy najważniejsze części:

```
alert  ip any any -> 10.0.1.23 any  (msg:”Skompromitowna maszyna”;sid:7877;rev:1)
```

Pierwsza część sygnatury określa, co należy zrobić, jeśli nadchodzące dane zostaną dopasowane do danej sygnatury. W prezentowanym przypadku wyraz `alert` oznacza, że taki pakiet ma wygenerować alarm. Inne możliwości to zapisanie danego pakietu, przerwanie przetwarzania dalszych sygnatur dla danego pakietu, czy włączenie specjalnej reguły dynamicznej służącej do logowania pewnego ruchu po wystąpieniu tego zdarzenia. Podczas ćwiczenia wykorzystywane będą jedynie sygnatury ze słowem kluczowym `alert`.

Druga część sygnatury służy do wybrania ruchu, który ma podlegać dalszej analizie, na podstawie protokołu, adresów i portów. Można wybrać protokoły IP, ICMP, UDP, TCP i przy specjalnej kompilacji – WiFi (wówczas możliwe jest pisanie sygnatur obejmujących ramki protokołu 802.11). Następnie wyspecyfikowany jest adres nadawcy i odbiorcy. Zapis jest podobny do zapisu adresów i portów w programie `tcpdump`. W najprostszej formie obejmuje on adresy oraz porty. Pozwala jednak na większą elastyczność, możliwe jest używanie słowa kluczowego `any`, powodującego dopasowanie do każdego adresu IP lub portu. W zaprezentowanym przykładzie dopasowany ruch pochodzi z dowolnego adresu źródłowego i dowolnego portu, a jest skierowany na dowolny port maszyny o adresie 10.0.1.23. W miejscu adresu IP można podać zakres adresów w postaci „numer sieci/długość maski”. Można także w nawiasach kwadratowych podawać kilka podsieci lub adresów, albo za

pomocą znaku „!” dokonać negacji podanych adresów. Zakresy można podawać także w przypadku portów. Tym razem zastosowana notacja to „port początkowy: port końcowy”, przy czym można pominąć jeden zakres w celu wybrania wszystkich portów mniejszych lub większych niż podany. Określić należy tu także kierunek działania sygnatury. Sygnatura może być sprawdzana dla ruchu tylko w jednym kierunku, wtedy adresy rozdzielone są znakami „->”), lub bez względu na kierunek, wtedy rozdzielone są znakami „<>”.

Trzecia część sygnatury zawiera tzw. opcje. Jeżeli pakiet lub odbudowany strumień TCP spełnia wcześniejsze wymagania, to zostaje poddany dalszej analizie. Opcje można porównać do różnego typu testów nakładanych na obserwowane dane. Najprostsze sprawdzają poszczególne, istotne z punktu widzenia bezpieczeństwa pola, np. ustawione bity w segmencie TCP, typ i kod komunikatu ICMP, wielkość pakietu UDP, czy ustawienie znacznika DF i MF w pakiecie IP. Bardziej skomplikowane opcje związane są z wyszukiwaniem pewnych wzorców. Możliwe jest wyszukiwanie wskazanego ciągu bajtów, wyszukiwanie tekstu bez względu na wielkość liter, czy nawet zastosowanie wyrażeń regularnych. Opcje wpisywane są kolejno i oddzielone jedna od drugiej średnikiem. Każda sygnatura musi posiadać co najmniej:

- opcję `msg`, podającą tekst logowany gdy dana sygnatura zostanie dopasowana do obserwowanych danych;
- identyfikator (*signature ID* - `sid`), powinien być on unikalny – przy tworzeniu własnej sygnatury należy to zapewnić np. przeszukując narzędziem `grep` wszystkie sygnatury w katalogu;
- wersję (*revision number* - `rev`).

Przykładowe reguły

```
alert udp 172.16.0.0/24 :1024 -> any 53 (msg:”Duzy pakiet DNS, z uprzywilejowanego
    portu”;dsize:>512; sid:7878;rev:1;)
alert tcp 172.16.0.192/26 any <> ! 172.16.0.123 80 (msg:”Dostep do niechcianych
    treści“;content:”sex”;nocase; sid:7879;rev:1;)
```

Reguła pierwsza alarmuje nas o wysłaniu z podsieci pakietu UDP, skierowanego do serwera DNS (dokładniej: portu najczęściej związanego z tym protokołem). Dodatkowo, aby sygnatura była wzbudzona, pakiet musi mieć port źródłowy mniejszy od 1024 i wielkość większą niż 512 bajtów.

Druga reguła poszukuje w stronach oglądanych z maszyn o adresach z zakresu 172.16.0.192-172.16.0.255 słowa „sex”, bez względu na wielkość liter. Alarm jest podnoszony, jeśli taki wyraz zostanie wykryty w zapytaniu do (lub stronie zwróconej z) innego serwera niż 172.16.0.123.

Spis opcji wraz z przykładami znajduje się w dokumentacji programu snort, w rozdziale „Writing Snort Rules” [SnortUm].

2.3.2 Sposób logowania danych

Głównym zadaniem programu `snort` jest wykrywanie i logowanie zdarzeń, które można uważać za naruszenia. Po dopasowaniu obserwowanych danych do sygnatury zostaje wygenerowany alarm. Może on być zgłaszany na kilka sposobów, np. przekazywany do standardowego programu logującego `syslog`, zapisywany w bazie danych czy przekazywany do jakiegoś innego programu za pomocą gniazd dziedziny uniksa lub bezpośrednio zapisany przez program `snort` w celu dalszej analizy.

Logi mogą być zapisywane w formie tekstowej i binarnej. Wybór formatu danych jest zależny od wielu czynników. Standardowo program `snort` loguje dane w formie tekstowej, wygodnej do czytania przez człowieka. W takim przypadku w katalogu `/var/log/snort` generowany jest plik `alert`. Znajdują się w nim informacje o wykrytych naruszeniach. Częścią wpisu o wykryciu naruszenia jest m.in. nazwa sygnatury, czyli opis w opcji `msg`. Plik tekstowy może być tworzony na dwa sposoby, w zależności od wartości opcji `-A`, z jaką uruchomiony zostanie program: `fast` zapisuje dane jednolinijkowe, `full` oprócz nazwy alarmu i daty tekstowo całą zawartość nagłówka pakietu. Bez względy na wybór opcji w katalogu zostaną stworzone podkatalogi odpowiadające adresom IP pakietów lub strumieni TCP, które wygenerowały alarmy. W tych podkatalogach znajdują się dokładne dane na temat zdarzenia, które wygenerowało alarm: adresy IP i porty, znaczniki pakietów itp.

Takie logowanie przy dużej ilości danych może być jednak bardzo niewydajne. Dlatego można uruchomić program `snort` z opcją `-b`, powodującą zapisywanie binarne pakietów, które wygenerowały alarm. W takim wypadku dane zostaną zapisane w pliku o formacie identycznym z plikami zapisywanymi przez program `tcpdump`. W celu późniejszej analizy można na innej maszynie uruchomić program `snort` w trybie, w którym do analizy pobierane są dane z pliku (opcja `-r`).

3 Przygotowanie do ćwiczenia

W ramach przygotowania do ćwiczenia należy zapoznać się z niniejszą instrukcją i dokumentacją programów narzędziowych: `tcpdump`, `nmap` i `snort`. Pomocne będzie zastanowienie się nad sposobem rozwiązania zadań podanych w dalszej części instrukcji. Warto przygotować sobie listę wywołań powyższych programów, potrzebnych do wykonania kolejnych zadań!

Ćwiczenie to polega na interpretowaniu danych zbieranych wyżej wymienionymi narzędziami. Aby dane te były użyteczne, narzędzia muszą być właściwie używane. W przypadku `tcpdump` ważne jest sprawne pisanie filtrów. Konfiguracja narzędzi realizowana jest licznymi opcjami wywołania. W trakcie realizacji ćwiczenia przydatne są podane poniżej opcje.

1. `tcpdump` : -c, -l, -n i wywołanie z „trójnikiem”: `tcpdump -l | tee dat`;
2. `nmap` : -sU, -sO, -sV, -O, -A, T5;
3. `snort` : -d, -h, -l, -c, -v, -r, -A fast.

Ćwiczenie realizowane jest na laboratoryjnych stacjach roboczych z systemem operacyjnym Linux-Debian bez środowiska graficznego. Przydatne będzie posiadanie niniejszej instrukcji na niezależnym nośniku (np. wydruk, tablet, notebook). Należy sprawnie posługiwać się podstawowymi poleceniami systemu Unix służącymi do nawigacji i zarządzania plikami, m.in.: `man`, `info`, `pwd`, `cd`, `ls -l`, `cp`, `mv`, `rm`, `mc`, `|`, `>`, `ssh`, `wget`, `grep`, `tar`, `scp`, `sftp`. Przydatne będą uniksowe polecenia do rejestracji sesji `script` oraz polecenia diagnostyki sieci `ping`, `traceroute`, `nslookup, dig`.

Studenci nie mający wprawy w posługiwaniu się powyższymi poleceniami powinni je przećwiczyć przed przystąpieniem do zajęć!

4 Potencjalne pytania sprawdzające

1. Wymień zalety i wady systemów HIDS/NIDS.
2. Wymień zalety i wady sygnaturowych systemów wykrywania naruszeń bezpieczeństwa.
3. Wymień zalety i wady systemów wykrywania naruszeń bezpieczeństwa analizujących anomalie.
4. Na czym polega przełączenie interfejsu sieciowego w tryb *promiscuous*?
5. Na czym polega (do czego służy i jak działa) tzw. technika *fingerprinting* stosowana przy skanowaniu maszyn podłączonych do sieci?
6. Z jakich trzech zasadniczych części składa się sygnatura *snorta*?
7. W jakiej sytuacji wskazane jest zapisywanie przez *snort* pakietów w postaci binarnej a nie tekstowej?
8. Jakie jest przeznaczenie programów *tcpdump*, *nmap* i *snort*?
9. Jaką funkcję w sieci lokalnej pełni brama domyślna?
10. Jakie jest przeznaczenie programów *ping*, *traceroute*, *nslookup, dig*?
11. Jakim poleceniem można skierować zapytanie do wybranego serwera DNS?
12. Do czego służą preprocesory snorta? Podaj przykładowe funkcje.
13. Jaka jest zaleta zdefiniowania adresu sieci lokalnej (*-h home-net*) w wywołaniu *snort*?
14. Napisać filtr *tcpdump* rejestrujący pakiety ssh do mion.elka.pw.edu.pl.
15. Napisać regułę *snorta* wykrywającą komunikaty http zawierające łańcuch znaków "ATAK".

5 Przebieg laboratorium

Ćwiczenia rozpoczniemy od obserwacji i analizy ruchu sieciowego przy użyciu programu `tcpdump`. Będziemy rozpoznawać protokoły generujące ruch i maszyny widoczne w sieci. Stymulując ruch w sieci i korzystając z filtrów wykrywać będziemy wybrane serwery usługowe.

W drugiej części ćwiczenia używany jest program `nmap` – skaner sieciowy. Program ten służy do zdobywania informacji na temat interesujących maszyn. Przy pomocy wcześniej omówionego programu `tcpdump`, będzie można obserwować w jaki sposób przebiega proces zdobywania informacji.

Trzecia część ćwiczenia poświęcona jest jednemu z najpopularniejszych systemów wykrywania włamań – programowi `snort`. Korzystając z wcześniej poznanych narzędzi

zostanie zaprezentowane działanie tego programu, reakcja na skanowanie i inne próby zdobywania informacji o chronionym systemie. Ponieważ `snort` jest systemem sygnaturowym, poznanie jego składni jest bardzo ważne. Ostatnim zadaniem na laboratorium będzie poznanie wybranych standardowo dostarczanych sygnatur i napisanie własnych zgodnie z zaleceniami prowadzących.

Uruchamiając stację roboczą, należy wybrać w menu wyboru systemu pozycję: „`Linux instalacja specjalna na zajęcia BSS`". Hasło do konta użytkownika `root` zostanie przekazane przez prowadzącego. Proszę upewnić się, że poprawne jest ustawienie zegara i ewentualnie je skorygować (polecenie `date`).

Sugerowane jest realizowanie ćwiczenia w 2-3 konsolach tekstowych (np. w jednej narzędzie monitorujące, w drugiej generacja ruchu, w trzeciej obserwacja pojawiających się plików logów). Przełączanie pomiędzy konsolami odbywa się za pomocą skrótów klawiszowych: Ctrl-Alt-F1, Ctrl-Alt-F2, Ctrl-Alt-F3 ... .

Sprawdź adres IP i jego maskę (polecenie `ifconfig`) używanego interfejsu sieciowego i zanotuj w celu podania w sprawozdaniu. Sprawdź w pliku `/etc/resolv.conf` adres serwera DNS.

5.1.1 Zadania dotyczące tcpdump

1. Uruchomić program `tcpdump` bez żadnych filtrów. Zaobserwować, jaki ruch dochodzi do maszyny. Co można na jego podstawie powiedzieć o sieci, do której podłączony jest dany komputer? Jakie protokoły są widoczne? Jaka jest ich główna funkcja procesów powodujących ten ruch?
   Podaj w sprawozdaniu wydane polecenia i odpowiedzi na powyższe pytania.
2. Napisać filtry do programu `tcpdump`, tak aby wyświetlany był ruch:
   - wybranego protokołu,
   - z wybranych adresów.

   Podaj w sprawozdaniu wydane polecenia i stosowne linie zebranego ruchu.
3. Za pomocą narzędzia `tcpdump` (poprzez monitorowanie odpowiednio wygenerowanego ruchu, a nie analizę plików konfiguracyjnych czy obserwację tylko samego wyniku działania poleceń diagnostycznych) znaleźć:
   - adres serwera DNS,
   - bramy domyślnej,
   - adresy kilku innych maszyn podłączonych do danej podsieci.

   Podaj w sprawozdaniu wydane polecenia dla zbierania i generowania ruchu oraz stosowne linie zebranego śladu. Znalezione adresy podaj w reprezentacji dziesiętno-kropkowej i nazw domenowych.

5.1.2 Zadania dotyczące nmap

Uwaga!!! Ponieważ skanowanie może być uważane za pierwszą fazę ataku, dopuszczalne jest skanowanie jedynie maszyn w sali laboratoryjnej (i ewentualnie maszyny podanej przez prowadzącego) oraz jedynie w czasie trwania ćwiczenia. Próby skanowania innych maszyn poza wskazanymi (umyślne lub nie) będą skutkowały usunięciem z zajęć z oceną 0.

4. Przeskanuj wybraną metodą maszynę o adresie podanym przez prowadzącego. Jakie informacje zostały w ten sposób uzyskane?

5. Zaobserwuj (za pomocą programu `tcpdump`) ruch podczas wykonywania skanowania. Na czym polega wybrany tryb skanowania? Jak to jest widoczne na śladzie `tcpdump`?
6. W jaki sposób przeprowadzany jest *fingerprinting* systemu operacyjnego? Przeprowadź go w odniesieniu do wskazanej przez prowadzącego maszyny (opcja `-T4` lub `-T5` przyspiesza to zadanie). Jaki system został wykryty?

Podaj w sprawozdaniu wydane polecenia oraz wybrane linie zebranych wyników uzupełniając je odpowiedziami na powyższe pytania.

5.1.3 Zadania dotyczące snort

Program `snort` podczas uruchomienia przeszukuje katalog bieżący w poszukiwaniu pliku z konfiguracją `snort.conf`. Istnieje możliwość wymuszenia wczytania konfiguracji z innego pliku za pomocą opcji `-c` (np. `/etc/snort/snort.conf`). W pliku tym znajdują się wszystkie parametry związane z działaniem programu.

Początkowa zawartość pliku konfiguracyjnego dotyczy uruchomienia tzw. *preprocesorów*. Są to zewnętrzne moduły dokonujące pewnych działań na danych, np. odbudowy strumieni TCP czy łączenia sfragmentowanych pakietów IP. W ramach laboratorium należy używać standardowych ustawień. Zmianie mogą podlegać ostatnie linie zaczynające się od słowa `include`, decydujące o wczytywaniu zewnętrznych plików zawierających sygnatury. Alternatywnie do tworzenia własnych plików z sygnaturami, można same sygnatury dopisywać do pliku `/etc/snotr/rules/local.rules`.

Program `snort` może działać jako demon; w takim wypadku należy go uruchomić z opcją `-D`. Podobnie jak w programie `tcpdump`, przy standardowym uruchomieniu analizie podlegają wszystkie pakiety. Stosując filtry identyczne z tymi stosowanymi przez program `tcpdump` można wyspecyfikować ruch podlegający analizie. W tym przypadku sprawdzeniu danych z sygnaturami będzie podlegał jedynie ruch, który już spełnia warunki nałożonego filtru.

Działanie `snorta` warto obserwować nie tylko na jego `stdout` ale również w logach w katalogu `/var/log/snort/`, szczególnie ciekawym jest plik `/var/log/snort/alert`.

7. Uruchom program `snort` ze wszystkimi dostępnymi regułami. Przeanalizuj logi po wykonaniu różnych skanowań danej maszyny przez program `nmap` (należy skanować obcą maszynę lub poddać własną maszynę skanowaniu przez kolegę), użycia programu `ping` itp. Jakie informacje można wyczytać z logów? Podaj w sprawozdaniu przykłady stosownych linii z logów.
8. Zaprojektuj własne reguły i przetestuj ich działanie.
   Wymagane reguły:
   - Sygnatura wykrywająca komunikacje protokołu DNS do innych serwerów niż przewidziane dla danej podsieci.

   Opcjonalne reguły:
   - Sygnatura wykrywająca ruch do wybranego serwera FTP.
   - Sygnatura wykrywająca ruch do wybranego serwera WWW.
   - Sygnatura wykrywająca ruch SMTP.

   Podaj w sprawozdaniu wydane polecenia i stosowne linie z logów (alerty i ewentualnie linie rejestrowanego ruchu).

6 Opracowanie wyników

Zebrane podczas ćwiczenia logi należy zebrać w jednym katalogu, spakować za pomocą narzędzia `tar` i skopiować za pomocą narzędzi `sftp` lub `scp` na zdalną maszynę (np. serwer `mion` lub `galera`), a później pobrać je do środowiska graficznego, np. narzędziami `sftp`, `scp`, `gftp`, `WinSCP`. Na koniec ćwiczenia należy odtworzyć pierwotny stan systemu usuwając:

1) zawartość katalogu roboczego,
2) plik z własną regułą programu `snort`,
3) dyrektywę włączającą tę regułę do pliku `snort.conf`,
4) zawartość katalogu logów (`/var/log/snort`).

Sprawozdanie, opracowywane po zakończeniu ćwiczenia, powinno mieć formę pojedynczego pliku w formacie PDF, o nazwie `Nazwisko.Imie.pdf` (bez polskich znaków), dołączonego do listu elektronicznego, zawierającego w temacie jeden z oznaczników: [BSS pn08], [BSS pn10], [BSS pn12], [BSS wt16], [BSS wt18], [BSS cz08], [BSS cz10], [BSS cz12], stosownie do zespołu ćwiczeniowego. W sprawozdaniu należy podać numer stanowiska i zespół ćwiczeniowy. Ewentualna realizacja ćwiczenia z innym niż nominalny zespołem może mieć miejsce jedynie w wyjątkowych przypadkach, powinna być uzgodniona z prowadzącym przynajmniej w przeddzień ćwiczenia i być wyraźnie zaznaczona w sprawozdaniu.

Sprawozdanie powinno być opisem i przekonywującym dowodem samodzielnego i świadomego wykonania wszystkich kroków ćwiczenia. Powinno zawierać odniesienie się do poszczególnych kroków, odpowiedzi na zadane pytania oraz reprezentatywne fragmenty logów zarejestrowanych podczas realizacji ćwiczenia. Logi powinny być zawarte w tekście sprawozdania, a nie np. dołączane do listu. Podawane fragmenty logów powinny ilustrować formułowane w sprawozdaniu wnioski. Umieszczanie pełnych logów będzie skutkowało obniżeniem oceny.

Oceniane będzie: poprawne interpretowanie zebranych danych, właściwe ilustrowanie wniosków, oraz kompletność wykonanych zadań.

Każda wykryta próba preparowania logów będzie skutkowała wystawieniem oceny 0 pkt!

7 Bibliografia

Dokumentacje do wykorzystywanych programów

[Tcpdump] Van Jacobson, Craig Leres and Steven McCanne, "man pages section 1: Tcpdump", 2 February 2017.
www.tcpdump.org/tcpdump_man.html

[Pcap] "man pages section 7: pcap-filter", 3 August, 2015.
www.tcpdump.org/manpages/pcap-filter.7.html

[Nmap] Gordon Fyodor Lyon: "Nmap Network Scanning", Nmap Project, January 1, 2009. Chapter 15. Nmap Reference Guide.
nmap.org/book/man.html

[Snort] Martin Roesch: "man pages section 1M: System Administration Commands", February 2009.
docs.oracle.com/cd/E36784_01/html/E36871/snort-1m.html

[SnortUm] Martin Roesch, Chris Green: "SNORT Users Manual 2.9.9", The Snort Project, November 14, 2016.
manual-snort-org.s3-website-us-east-1.amazonaws.com/snort_manual.html

Materiały uzupełniające

[Ptacek98] T. H. Ptacek, T.N. Newsham.: "Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection", January 1998.

List of TCP and UDP port numbers
en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers

Command-line network packet crafting and injection utility
github.com/troglobit/nemesis

Suricata – a free and open source IDS, IPS, network security monitoring and offline pcap processing. suricata-ids.org