

Projekt OINS

Michał Kocon

Mateusz Chomiczewski

Zespół 5

STRESZCZENIE

Dokument jest poświęcony opracowaniu metody steganografii sieciowej. Do realizacji tego zadania została wykorzystana biblioteka WinDivert oraz język programowania c++, w celu stworzenia aplikacji umożliwiających przesył i odbiór danych w ukrytym kanale transmisyjnym oraz dodatkowa aplikacja pozwalająca przeanalizować ruch sieciowy pod kątem obecności takiego kanału.

Steganografia została zrealizowana korzystając z niedoskonałości modelu TCP/IP.

Spis treści

1. Wstęp	4
2. Techniki steganografii sieciowej w modelu TCP/IP	5
2.1. Warstwa dostępu do sieci	5
2.2. Warstwa Internetu	5
2.3. Warstwa transportowa	7
2.4. Warstwa aplikacji	8
3. Realizacja algorytmu steganograficznego	9
3.1. Koncepcja przesyłania niejawnych wiadomości	9
3.2. Biblioteka WinDivert	10
3.3. Aplikacja do przesyłu danych - TrafficModifier	11
3.4. Aplikacja do odbioru danych - NinjaReader	12
3.5. Aplikacja do analizy ruchu sieciowego - MessageFinder	12
4. Testy algorytmu i aplikacji	13
4.1. Scenariusz testowy	13
4.2. Analiza przeprowadzonych transmisji	13
4.2.1. Przepływność	13
4.2.2. Elastyczność	14
4.2.3. Wykrywalność	14
5. Podsumowanie	15
6. Bibliografia	16
7. Spis rysunków	17
8. Spis tabel	17

1. Wstęp

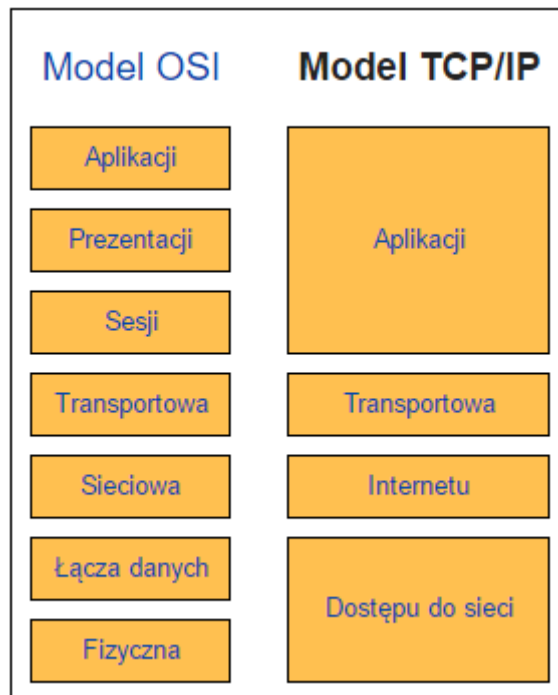
Steganografia jest techniką pozwalającą na przesył komunikatów w sposób niejawnym. W steganografii komputerowej wykorzystywane jest wiele technik umożliwiających ukrycie wiadomości. Bardzo popularną formą steganografii komputerowej jest ukrywanie wiadomości w obrazach czy plikach dźwiękowych, wykorzystując do tego celu nadmiarowość bitową wykorzystywaną przy ich zapisie. Modyfikując najmniej znaczące bity piksela lub próbki dźwięku można ukryć wiadomość, nie wpływając przy tym znacząco na jakość danych pierwotnych.

Innym sposobem ukrycia wiadomości jest steganografia sieciowa. Jest ona ściśle powiązana z protokołami sieciowymi. Obecnie za pomocą sieci przesyłane są bardzo duże ilości danych. Ze względu na problem z kontrolą takiej ilości informacji, komunikacja sieciowa stała się dużym obszarem, gdzie można wykorzystać metody steganograficzne.

W ramach niniejszego projektu, wykorzystywana jest niedoskonałość protokołu TCP/IP, pozwalająca na przesył danych w niejawnym kanale transmisyjnym, zrealizowanym równolegle do głównego kanału transmisyjnego.

2. Techniki steganografii sieciowej w modelu TCP/IP

Model TCP/IP jest zbiorem protokołów służących od transmisji pakietów w sieciach komputerowych. Upraszcza on siedmiowarstwowy model OSI do modelu czterowarstwowego co zostało zaprezentowane na rysunku 2.1. Większość sieci jest oparta o model TCP/IP.



Rysunek 2.1. Porównanie warstw modelu OSI oraz TCP/IP. [3]

2.1. Warstwa dostępu do sieci

Najniższa z warstw modelu TCP/IP realizuje przekazywanie danych przez fizyczne połączenia pomiędzy urządzeniami. Najczęściej jest ona realizowana przez karty sieciowe lub odpowiednie modemy. Warstwa ta zazwyczaj realizuje bezpośrednie połączenia punkt-punkt. Z tego powodu przesył danych za pomocą tej warstwy w przypadku wielu urządzeń pośredniczących jest niemożliwy, ponieważ nagłówek zostaje nadpisany za każdym razem, gdy dotrze do routera.

2.2. Warstwa Internetu

Warstwa Internetu może zostać zrealizowana z pomocą dwóch protokołów – IPv4 oraz IPv6, jednak nadal najczęściej używana jest wersja czwarta, dlatego zostanie ona zanalizowana pod kątem możliwości ukrycia danych.

Na rysunku 2.2 został przedstawiony nagłówek protokołu IPv4. Na żółto zostały oznaczone potencjalne pola, w których można zrealizować transmisję steganograficzną.

Bity 0-3	4-7	8-15	16-18	19-23	24-31
Wersja	IHL	Typ usługi	Długość całkowita		
Identyfikator			Flagi	Przemieszczenie fragmentacji	
TTL		Protokół	Suma kontrolna nagłówka		
Adres źródłowy					
Adres docelowy					
Opcje					Dopełnienie

Rysunek 2.2. Nagłówek protokołu IPv4. [1]

Pole typu usługi (Type of Services) jest polem, które nie posiada jednoznacznej budowy i może się różnić w zależności od implementacji protokołu IP. Często to pole, zwłaszcza w małych sieciach, nie jest wykorzystywane przez co jest ono puste i może zostać wykorzystane w celu ukrycia wiadomości. W nowoczesnych sieciach może ono zostać wykorzystane do poprawy jakości działania połączenia i jest zastąpione polem DSCP. W zależności od pożądanej przepływności w metodzie steganograficznej, możliwe jest wykorzystanie różnych bitów pola DSCP. Przy analizie przesyłanych danych w polu TOS, można zidentyfikować jego nieprawidłowe użycie. Dodatkowo istnieje możliwość, że zostanie ono wymazane lub usunięte co sprawia, że umieszczony w nim niejawny kanał komunikacyjny nie będzie niezawodny.

Pole identyfikatora (ID) jest unikalne dla każdej przesyłanej ramki za wyjątkiem danych, które podlegają fragmentacji. W takim przypadku każdy z fragmentów posiada taki sam numer identyfikacyjny. Pole ID jest potencjalnym miejscem do wykorzystania w steganografii. Najważniejszym warunkiem jest zapewnienie unikalnych numerów identyfikacyjnych dla każdej z ramek. Dopóki preparowane numery są różne dla różnych ramek, transmisja przebiega prawidłowo. Sposób generowania identyfikatora zależy od wielu czynników jak system operacyjny czy sprzęt wysyłający pakiety. Z tego powodu bez dokładnej znajomości sposobu generowania wartości pola, znalezienie ukrytych wiadomości jest trudnym zadaniem.

Pole flag określa sposób obsługi ramki w przypadku jego defragmentacji. Zastosowanie tego pola nie jest bezpieczne z punktu widzenia ukrycia transmisji ze względu na możliwość łatwego wykrycia niezgodności tego pola.

Pole opcji jest polem opcjonalnym w nagłówku i ma zmienną długość. Implementacja tego pola pozwala na dodawanie zdefiniowanych opcji w dokumentacji RFC oraz pól użytkownika. W pierwszym przypadku dane mogą zostać usunięte jeżeli trafiają do odpowiednio skonfigurowanych routerów. W drugim przypadku dane pozostają w nagłówku. [1]

Dodatkowym polem do wykorzystania w steganografii może być pole sumy kontrolnej. W przypadku, gdy pole to nie jest wykorzystywane do sprawdzenia poprawności przesyłanego nagłówka, jego wartość może być dowolna. Istnieje również możliwość dodania odpowiedniego pola opcji, które spowoduje, że zmodyfikowana suma kontrolna będzie prawidłowa. [2]

2.3. Warstwa transportowa

Warstwa transportowa opiera się na dwóch protokołach. Protokół TCP pozwala na transmisję w trybie połączeniowym natomiast UDP jest protokołem bezpołączeniowym. W dalszej części podrozdziału przeanalizowany zostanie protokół TCP pod kątem możliwości wykorzystania technik steganograficznych. Nagłówek protokołu TCP znajduje się na rysunku 2.3. Kolorem żółtym zostały oznaczone pola w których istnieje możliwość stworzenia ukrytego kanału transmisyjnego.

Opis nagłówka TCP				
	Bit 0-3	4-7	8-15	16-31
0	Port nadawcy			Port odbiorcy
32	Numer sekwencyjny			
64	Numer potwierdzenia			
96	Długość nagłówka	Zarezerwowane	Flagi	Szerokość okna
128	Suma kontrolna			Wskaźnik priorytetu
160	Opcje (opcjonalnie)			
160/192+	Dane			

Rysunek 2.3. Nagłówek protokołu TCP. [1]

Numer sekwencyjny jest polem wybieranym podczas nawiązywania połączenia i wysyłany w pierwszym pakiecie TCP. Jego wartość jest losowa. Może ono zostać wykorzystane tylko raz podczas przesyłania nawiązywania połączenia. Wykrywalność jak i ilość danych możliwych do przesłania z pomocą tego pola jest bardzo mała.

Zarezerwowane pole pomiędzy długością nagłówka a flagami jest polem, które według dokumentacji RFC musi być wyzerowane. Zazwyczaj jego zmiana nie wpływa na komunikację, a zawartość jest ignorowana. Ze względu na ustaloną z góry wartość pola, wykrycie nieprawidłowości związanych ze steganografią jest bardzo prostym zadaniem.

Wskaźnik priorytetu jest obecnie rzadko używanym polem i ma przeznaczenie jedynie w przypadku ustawienia flagi URG. Przy jej wyzerowaniu i zapisaniu wskaźnika priorytetu możliwe jest przesłanie ukrytych wiadomości. Wykrycie takiej transmisji jest stosunkowo łatwe ze względu na wartość wskaźnika zazwyczaj równą zeru w przypadku braku ustawionej flagi URG.

Pole opcji podobnie jak w przypadku protokołu IP, pozwala na zdefiniowanie własnych opcji w których mogą zostać ukryte wiadomości, jednak w odróżnieniu od niego jest zawsze przesyłane. [1]

Pole sumy kontrolnej może być wykorzystane podobnie jak w protokole IP.

2.4. Warstwa aplikacji

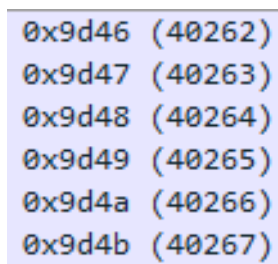
Metody steganograficzne działające w warstwie aplikacji najczęściej modyfikują protokoły komunikacji wyższych warstw jak HTTP czy FTP. Wymaga to jednak korzystania z określonego protokołu podczas gdy TCP oraz IP są używane znacznie częściej, ponieważ są podstawowym nośnikiem danych w sieciach.

3. Realizacja algorytmu steganograficznego

3.1. Koncepcja przesyłania niejawnych wiadomości

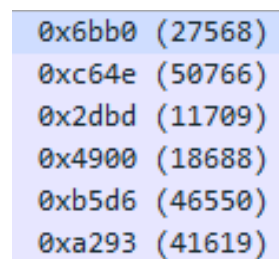
Do przesyłu wiadomości w niejawnym kanale komunikacyjnym zostało wykorzystane pole identyfikacyjne protokołu IPv4.

W pierwotnej wersji, w pole identyfikacyjne było zastąpione bezpośrednio strumieniem z pliku Antygona, dodatkowo zaszyfrowanym kluczem publicznym algorytmu RSA. Pozwalało to na przesył 2 bajtów na jedną ramkę. Jednak po przeprowadzeniu analizy przesyłanych pakietów zmodyfikowanych i niezmodyfikowanych, można było zauważyć zdecydowane odstępstwa między formą przesyłanych identyfikatorów IP. W przypadku danych niezaszyfrowanych kolejne pakiety posiadały pole identyfikacyjne inkrementowane o 1 z każdym kolejnym pakietem (rysunek 3.1). Rysunek 3.2 przedstawia zmodyfikowane pole identyfikatora przez strumień danych.



0x9d46	(40262)
0x9d47	(40263)
0x9d48	(40264)
0x9d49	(40265)
0x9d4a	(40266)
0x9d4b	(40267)

Rysunek 3.1. Pola identyfikacyjne IP bez ukrytej wiadomości



0x6bb0	(27568)
0xc64e	(50766)
0x2dbd	(11709)
0x4900	(18688)
0xb5d6	(46550)
0xa293	(41619)

Rysunek 3.2. Pola identyfikacyjne IP z ukrytą wiadomością

Z tego powodu została opracowana druga koncepcja przesyłu danych. W celu wysłania informacji, pole identyfikacyjne jest zwiększane o odpowiednią liczbę. Ze strumienia danych pobierane są kolejne bity. W przypadku pobrania bitu o wartości 0, pole identyfikacyjne jest inkrementowane o 1, tak jak w przypadku kolejnych pakietów bez ukrytej wiadomości. W przypadku pobrania bitu o wartości 1, pole identyfikacyjne jest zwiększane o 2. Takie podejście pozwala na znacznie mniej widoczną modyfikację ramki. Przykład takiej transmisji został zaprezentowany na rysunku 3.3.

0x487d (18557)	
0x487e (18558)	0
0x487f (18559)	0
0x4880 (18560)	0
0x4881 (18561)	0
0x4883 (18563)	1
0x4884 (18564)	0
0x4885 (18565)	0
0x4887 (18567)	1
0x4888 (18568)	0
0x4889 (18569)	0
0x488a (18570)	0

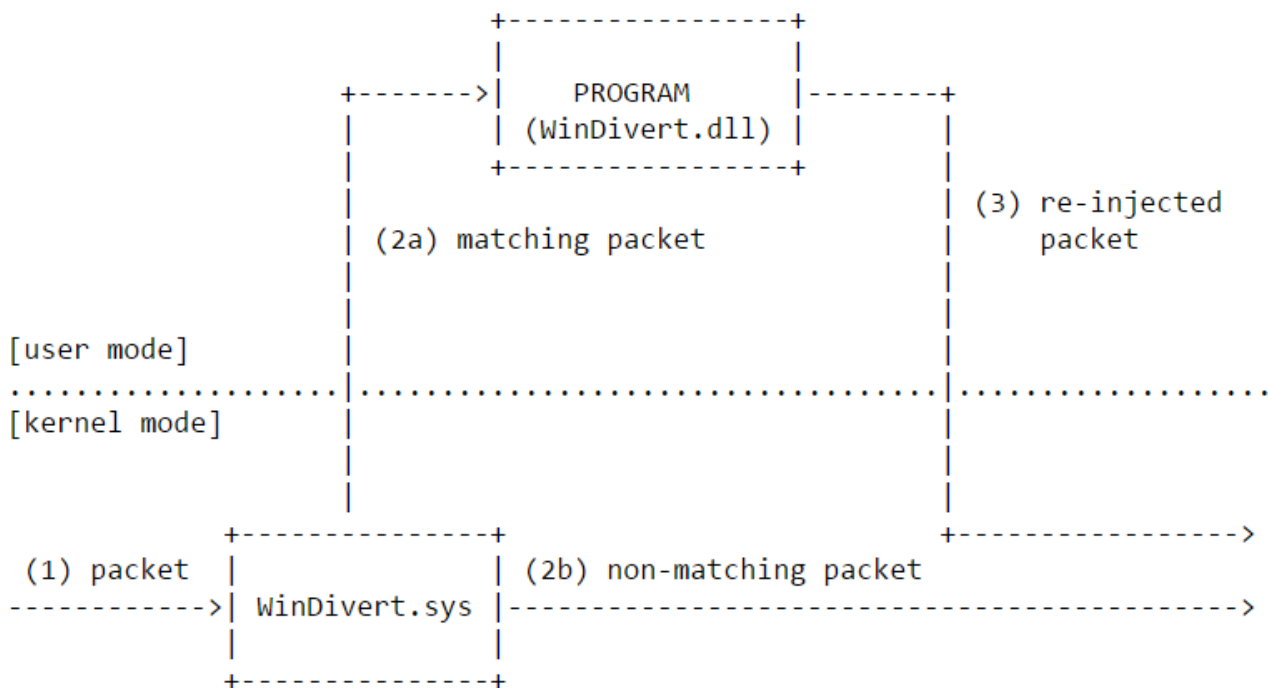
Rysunek 3.3. Pola identyfikacyjne z ukrytą wiadomością oraz jej zdekodowanie

Algorytm może być odpowiednio modyfikowany w celu zwiększenia przepływności/zwiększenia odporności na wykrycie. W przypadku gdy potrzebna jest większa przepustowość, ilość bitów wczytywanych ze strumienia może być większa niż 1. Przykładowo pobierane jest 2 bity ze strumienia i są one dodawane do obecnego identyfikatora. W przypadku pobrania bitów 00, strumień jest zwiększany standardowo o 1, natomiast w przypadku 11 o 5, ponieważ system operacyjny numeruje każdy kolejny identyfikator zwiększając go o 1. W celu zmniejszenia wykrywalności, identyfikator może być modyfikowany co pewną ustaloną liczbę pakietów. Wpłynie to jednak znacząco na przepustowość ukrytego kanału.

Ramki z ukrytą wiadomością są przesyłane jedynie na określonym porcie protokołu TCP do wyznaczonego adresu IP, jednak mogłyby być wysyłane również na wskazanym porcie protokołu UDP lub na dowolnym porcie do wskazanego adresu IP.

3.2. Biblioteka WinDivert

W implementacji algorytmu steganograficznego, wykorzystana została biblioteka WinDivert dla systemu Windows [4]. Pozwala ona na przechwytywanie pakietów wysyłanych przez aplikacje sieciowe, a następnie ich modyfikację i retransmisję. Posiada ona również tryb podsłuchiwania przesyłanych pakietów bez potrzeby przejęcia i retransmisji. Architektura biblioteki została przedstawiona na rysunku 3.4. Działa ona na poziomie jądra poprzez odpowiedni sterownik, poniżej warstwy sieciowej. Z tego powodu konieczne jest uruchomienie aplikacji jej używających z prawami administratora.



Rysunek 3.4. Architektura WinDivert. [4]

3.3. Aplikacja do przesyłu danych - TrafficModifier

Aplikacja do przesyłu danych (TrafficModifier) została napisana w języku c++. Za pomocą biblioteki WinDivert przechwytyuje ona pakiety o następującym filtrze:

```
"outbound && ip && tcp.SrcPort == 8000",
```

co wymusza kolejno: pakiety wychodzące, wykorzystujące IPv4, na porcie TCP równym 8000. Pozwala to na zmniejszenie narzutu przez aplikację na zainfekowany system, ponieważ nie ma potrzeby retransmisji pakietów, które nie spełniają warunków filtru. Dodatkowo może zostać dodany warunek o przechwytywaniu pakietów jedynie w przypadku konkretnego adresu docelowego, jednak ze względu na dydaktyczną formę programu, jako adres docelowy jest brany pierwszy IP klienta, który wysłał żądanie nawiązania połączenia na porcie 8000, co pozwala na jego prostsze wykorzystanie.

Aplikacja składa się z dwóch części. Pierwszą z nich jest pętla służąca do przechwytywania odpowiednich pakietów. Odczytuje ona odpowiednio kolejne pakiety spełniające podany na początku filtr, modyfikuje je i wykonuje ich retransmisję.

Drugą częścią jest funkcja modyfikująca strumień pakietów. Dane do zakodowania są pobierane z pliku w formie binarnej. Do każdego z pakietów wpisywany jest jeden bit zgodnie z algorytmem opisanym w rozdziale 3.1. Został on zaimplementowany za pomocą wewnętrznego licznika, który jest inkrementowany z każdym wczytanym bitem 1. Następnie licznik jest dodawany do każdej odczytanej ramki. Pozwala to zachować unikalne wartości pola identyfikatora.

3.4. Aplikacja do odbioru danych - NinjaReader

Aplikacja do odczytywania ukrytych danych (NinjaReader) została napisana w języku c++. Przychodzące pakiety są filtrowane podobnym filtrem do tego zastosowanym w aplikacji modyfikującej ruch sieciowy.

```
"inbound && ip && tcp.SrcPort == 8000"
```

co kolejno oznacza: ruch przychodzący, wykorzystujący IPv4, na porcie TCP 8000. Istnieje możliwość dodania do filtra konkretnego źródłowego adresu IP. Ze względów praktycznych wykorzystany został adres IP od którego przyjdzie pierwsza wiadomość spełniająca podany filtr.

Ze względu na brak potrzeby przechwytywania ramek, główna pętla programu jedynie odczytuje kolejne ramki spełniające wcześniej zdefiniowany filtr. Każda ramka jest analizowana przez funkcję która śledzi stan obecnego i poprzedniego pakietu w celu określenia wartości skoku pola identyfikatora. W zależności od tego, czy skok wynosił 1 czy 2, wpisywany jest kolejno bit 0 lub 1 do ośmiobitowej zmiennej. Gdy odczytane zostanie 8 bitów, zawartość zmiennej jest wypisywana na strumień standardowy oraz do pliku.

3.5. Aplikacja do analizy ruchu sieciowego - MessageFinder

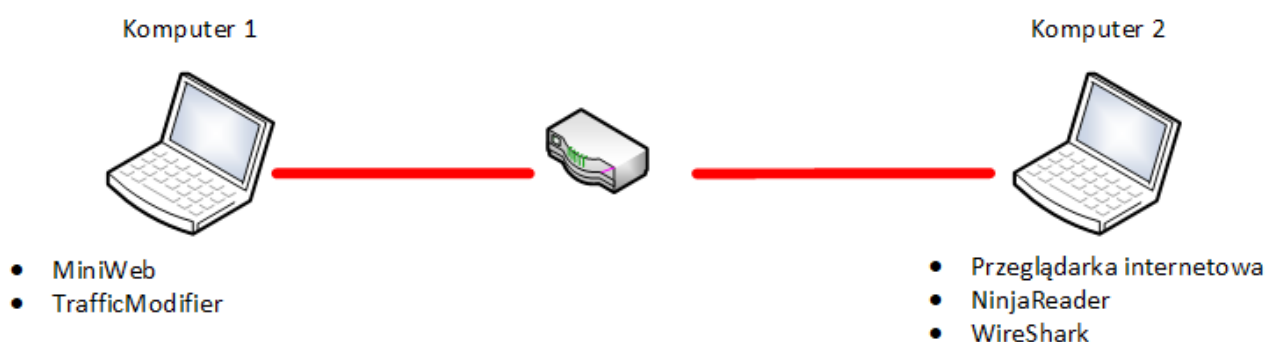
Ostatnią aplikacją (MessageFinder) jest program umożliwiający odczytywanie wcześniej przygotowanych plików ruchu sieciowego, rozpoznający czy dany zrzut aktywności sieciowej zawiera ukryte wiadomości czy też nie. Została tutaj użyta biblioteka libpcap [5], umożliwiająca operacje na plikach ruchu sieciowego zapisanych w programie Wireshark [7]. Zrzuty do programu zostały odpowiednio odfiltrowane aby nie zawierały informacji nadmiarowych.

Sprawdzenie czy zrzut posiada transmisję steganograficzną odbywa się na zasadzie analizy czy występują skoki pola identyfikacyjnego o 2. Jeżeli wystąpi ich pewna określona liczba to wypisywany jest komunikat o znalezieniu ukrytego kanału. Aplikacja odczytuje pierwsze dwa tysiące pakietów aby nie sprawdzać całego ze względu na znaczne czasy analizy takiego zrzutu.

4. Testy algorytmu i aplikacji

4.1. Scenariusz testowy

W celu przetestowania aplikacji zostały wykorzystane dwa komputery połączone siecią ethernet za pomocą routera. Na jednym z nich został uruchomiony serwer HTTP za pomocą aplikacji MiniWeb [6]. W celu umieszczenia niejawnego kanału komunikacyjnego była dodatkowo włączana aplikacja TrafficModifier. Drugi z komputerów pełnił funkcję klienta otwierającego w przeglądarce stronę udostępnianą przez serwer MiniWeb. Podczas transmisji uruchomiona była aplikacja NinjaReader w celu odbioru ukrytej wiadomości. Do zapisywania ruchu wykorzystany został program WireShark. Całość została przedstawiona na rysunku 4.1.



Rysunek 4.1. Schemat sieci i lista programów uruchomionych na poszczególnych komputerach.

Jako strony testowe, wykorzystane zostały zapisane strony portali internetowych. Były one odświeżane tyle razy, by w ruchu znajdowało się minimum 10kB ukrytych danych.

4.2. Analiza przeprowadzonych transmisji

4.2.1. Przepływność

W tabeli 4.1 zostały zestawione parametry poszczególnych transmisji z różnymi stronami internetowymi udostępnianymi przez serwer.

Numer transmisji	Rozmiar ukrytych danych [bit]	Rozmiar ukrytych danych [bajt]	Czas ~ [s]	Przepływność stenograficzna [bit/s]
1	102000	12750	217	470
2	120000	15000	238	504,2
3	127000	15875	211	601,9
4	90000	11250	269	334,6
5	106000	13250	209	507,2
6	91000	11375	221	411,8
7	99000	12375	232	426,7

Tabela 4.1. Parametry transmisji

Szacowana średnia przepustowość steganograficzna wynosi około 465 [bit/s], jednak będzie ona zależała od przepływności łącza. W naszym przypadku przepływność była ograniczona szybkością wysyłania danych przez serwer.

4.2.2. Elastyczność

Aplikacja działa pomimo zerwania połączenia, nie wpływa to na odbierany zaszyty tekst. Zerwanie połączenia z zainfekowanym serwerem spowoduje zatrzymanie transmisji wiadomości steganograficznej w pewnym punkcie i po ponownym wznowieniu będzie kontynuowana transmisja, od owego punktu, pod warunkiem jeżeli obie aplikacje będą włączone. W przypadku utraty jakiegokolwiek pakietu może wprowadzić błąd jedynie jednego bitu, przez co metoda powinna być odporna na nieoczekiwane błędy transmisji.

Ukrywanie wiadomości w protokole IP pozwala na znaczne rozszerzenie możliwości aplikacji, ponieważ transmisja nie musi być uzależniona od protokołów warstw wyższych.

4.2.3. Wykrywalność

Dla zwykłego użytkownika zmiana którą wprowadza aplikacja TrafficModifier jest niezauważalna ze względu na brak modyfikacji danych dostarczanych użytkownikowi. Posługując się nawet programami do obserwacji ruchu w sieci, doświadczony użytkownik może mieć problemy z zauważeniem różnicy w transmisji ze względu na niewielkie zmiany w modyfikowanych pakietach. Wykorzystana metoda nie zapewnia dużych przepływności lecz powinna gwarantować niewykrywalność i możliwość przesłania wiadomości w przypadku używania protokołu IPv4, niezależnie od protokołów wyższych warstw. Pole identyfikacyjne protokołu IP może się zmieniać w bardzo różny sposób w zależności od systemu, przez co wymagana jest wiedza jak wyglądają niezmodyfikowane i spreparowane pakiety.

5. Podsumowanie

W ramach projektu opracowano metodę steganograficzną oraz dwie aplikacje pozwalające na modyfikację i odczyt ruchu sieciowego jak również dodatkowy program do analizy ruchu sieciowego zapisanego do pliku pcap.

Podczas realizacji przeanalizowany został stos protokołów modelu TCP/IP. Na jego podstawie opracowany został algorytm steganograficzny wykorzystujący pole identyfikacyjne protokołu IPv4. W celu jego realizacji zostały stworzone trzy aplikacje. Pierwsza pozwala na modyfikację ruchu jednego z komputerów w celu umieszczenia w nim niejawnego kanału transmisyjnego. Druga odpowiada za podsłuch oraz interpretację zakodowanych informacji. Trzeci program umożliwia stwierdzenie obecności ukrytej transmisji na podstawie dołączonych zrzutów ruchu sieciowego.

Zaprojektowane aplikacje zostały przebadane na podstawie transmisji 7 różnych stron za pomocą protokołu HTTP pod kątem przepływności, elastyczności oraz niewykrywalności ukrytej transmisji.

Istotnym ulepszeniem byłoby dostosowanie przepływności poprzez przesyłanie większej ilości bitów w każdym pakiecie, w zależności od natężenia ruchu zainfekowanej maszyny, dzięki czemu byłaby możliwość znacznie szybszego przesyłu ukrytych danych nieznacznie zwiększając ryzyko jej wykrycia.

Podczas działania programów zaobserwowano problem z wypisywaniem odpowiednich znaków, jeżeli NinjaReader został uruchomiony po przesłaniu przez TrafficModifier liczby bitów nie będących wielokrotnością ósemki. Jest to spowodowane przesunięciem strumienia bitowego przez bity występujące na początku, które pochodzą z częściowo przesłanego wcześniej bajtu.

6. Bibliografia

- [1] Steganografia w protokole TCP/IP 1 [PDF]; Dostęp na: <http://img1.oferia.pl/8b4b3cdce386d8290c1b14414902d295.pdf> [Dostęp: 12.04.2017]
- [2] A Survey of Covert Channels and Countermeasures in Computer Network Protocols; Dostęp na: <http://caia.swin.edu.au/cv/szander/publications/szander-ieee-comst07.pdf> [Dostęp: 12.04.2017]
- [3] Porównanie warstw modelu TCP/IP i modelu OSI; Dostęp na: https://pl.wikipedia.org/wiki/Model_TCP/IP [Dostęp: 26.04.2017]
- [4] WinDivert; Dostęp na: <https://reqrypt.org/windivert.html> [Dostęp: 8.04.2017]
- [5] libpcap; Dostęp na <http://www.tcpdump.org> [Dostęp: 16.04.2017]
- [6] MiniWeb; Dostęp na: <http://miniweb.sourceforge.net/> [Dostęp: 8.04.2017]
- [7] WireShark; Dostęp na: <https://www.wireshark.org> [Dostęp: 1.03.2017]

7. Spis rysunków

Rysunek 2.1. Porównanie warstw modelu OSI oraz TCP/IP. [3].....	5
Rysunek 2.2. Nagłówek protokołu IPv4. [1].....	6
Rysunek 2.3. Nagłówek protokołu TCP. [1].....	7
Rysunek 3.1. Pola identyfikacyjne IP bez ukrytej wiadomości.....	9
Rysunek 3.2. Pola identyfikacyjne IP z ukrytą wiadomością.....	9
Rysunek 3.3. Pola identyfikacyjne z ukrytą wiadomością oraz jej zdekodowanie.....	10
Rysunek 3.4. Architektura WinDivert. [4].....	11
Rysunek 4.1. Schemat sieci i lista programów uruchomionych na poszczególnych komputerach.....	13

8. Spis tabel

Tabela 4.1. Parametry transmisji.....	13
---------------------------------------	----