POLITECHNIKA WARSZAWSKA
WYDZIAŁ ELEKTRONIKI I TECHNIK INFORMACYJNYCH
INSTYTUT TELEKOMUNIKACJI

KRZYSZTOF STANISŁAW SZCZYPIORSKI

STEGANOGRAFIA SIECIOWA

CYKL PUBLIKACJI Z LAT 2008-2011
BĘDĄCY PRZEDMIOTEM ROZPRAWY HABILITACYJNEJ

WARSZAWA, KWIECIEŃ 2011

Spis treści

Wprowadzenie ..... ***I***

1. Krzysztof Szczypiorski
   A Performance Analysis of HICCUPS - a Steganographic System for WLAN
   W: Telecommunication Systems: Modelling, Analysis, Design and Management, Volume 49: 3-4 - March/April 2012, Springer US, Journal no. 11235 (wersja elektroniczna dostępna, wersja papierowa w druku) ***1***

2. Krzysztof Szczypiorski, Wojciech Mazurczyk
   Steganography in IEEE 802.11 OFDM Symbols
   W: International Journal of Security and Communication Networks, John Wiley & Sons, 2011 (wersja elektroniczna dostępna, wersja papierowa w druku) ***7***

3. Wojciech Mazurczyk, Krzysztof Szczypiorski
   Steganography of VoIP Streams
   W: Robert Meersman and Zahir Tari (Eds.): OTM 2008, Part II - Lecture Notes in Computer Science (LNCS) 5332, Springer-Verlag Berlin Heidelberg, Proc. of OnTheMove Federated Conferences and Workshops: The 3rd International Symposium on Information Security (IS'08), Monterrey, Mexico, November 9-14, 2008, pp. 1001-1018 ..... ***19***

4. Józef Lubacz, Wojciech Mazurczyk, Krzysztof Szczypiorski
   Vice over IP
   W: IEEE Spectrum, ISSN: 0018-9235, February 2010, pp. 42-47
   (artykuł także na stronie internetowej czasopisma pod nazwą: ***Vice Over IP: The VoIP Steganography Threat***) ***37***

5. Wojciech Mazurczyk, Krzysztof Szczypiorski
   Covert Channels in SIP for VoIP signalling
   W: Hamid Jahankhani, Kenneth Revett, and Dominic Palmer-Brown (Eds.): ICGeS 2008 - Communications in Computer and Information Science (CCIS) 12, Springer Verlag Berlin Heidelberg, Proc. of 4th International Conference on Global E-security 2008, London, United Kingdom, 23-25 June 2008, ISBN: 978-3-540-69402-1, pp. 65-72 ..... ***43***

6. Wojciech Mazurczyk, Krzysztof Cabaj, Krzysztof Szczypiorski
   What are suspicious VoIP delays?
   W: Multimedia Tools and Applications, 2010, Springer US, Journal no. 11042 (wersja elektroniczna dostępna, wersja papierowa w druku) ***51***

7. Wojciech Mazurczyk, Miłosz Smolarczyk, Krzysztof Szczypiorski
   RSTEG: Retransmission Steganography and Its Detection
   W: Soft Computing, Volume 15, Number 3, March 2011, pp. 505-515..... ***69***

8. Wojciech Mazurczyk, Krzysztof Szczypiorski
   Evaluation of steganographic methods for oversized IP packets
   W: Telecommunication Systems: Modelling, Analysis, Design and Management, Volume 49: 3-4 - March/April 2012, Springer US, Journal no. 11235 (wersja elektroniczna dostępna, wersja papierowa w druku) ***81***

9. Bartosz Jankowski, Wojciech Mazurczyk, Krzysztof Szczypiorski
   PadSteg: Introducing Inter-Protocol Steganography
   Zaakceptowane do: Telecommunication Systems: Modelling, Analysis, Design and Management, Springer US, Journal no. 11235 (dostępne też jako preprint: http://arxiv.org/abs/1104.0422)..... ***93***

10. Wojciech Frączek, Wojciech Mazurczyk, Krzysztof Szczypiorski
    Stream Control Transmission Protocol Steganography
    W: Proc. of: The 2010 International Conference on Multimedia Information Networking and Security (MINES 2010), Nanjing, China, November 4-6, 2010, pp. 829-834 ..... ***103***

Istotą **steganografii** jest przekazywanie informacji w taki sposób, by nie ujawniać osobom postronnym faktu ich istnienia, ani samego aktu ukrytej komunikacji. Słowo steganografia pochodzi z języka greckiego (στεγανογραφία) i oznacza dosłownie *osłonięte, zakryte pisanie*. Steganografia jest odmienna od kryptografii (*κρυπτογραφία – ukryte, tajne pisanie*), której celem jest ochrona treści przesyłanej wiadomości przed jej odczytaniem przez osoby nieuprawnione, przy czym sam fakt komunikacji może być znany.

Metody steganograficzne ewoluują wraz z rozwojem nowych form komunikacji międzyludzkiej. Współczesne rozwiązania i prace badawcze w dziedzinie steganografii koncentrują się głównie na ukrywaniu informacji w treściach multimedialnych (cyfrowych obrazach, plikach dźwiękowych, filmach wideo, przesyłanym tekście) [12] oraz w sieciowych protokołach komunikacyjnych. W pierwszym przypadku istotą rozwiązań steganograficznych jest ukrycie danych w taki sposób, aby były one niewykrywalne przez zmysły człowieka (wzrok, słuch). W przypadku steganografii wykorzystującej jako nośnik protokoły sieciowe, modyfikacji podlegają właściwości protokołów, takie jak zawartość pól opcjonalnych, sekwencje wysyłanych wiadomości itp. Stąd metody steganograficzne, wykorzystujące jako nośnik ukrytych informacji jednostki danych lub sposób ich wymiany w sieciach telekomunikacyjnych, określa się mianem **steganografii sieciowej**. Termin ten został zaproponowany przeze mnie w 2003 roku [25].

Przedmiotem przedstawionej rozprawy habilitacyjnej są rezultaty badań w zakresie steganografii sieciowej. Badania zostały przeprowadzone zgodnie z zaproponowaną przeze mnie w 2003 roku, w pracy [24], ideą wykorzystywania „naturalnych" niedoskonałości w funkcjonowaniu sieci do stworzenia ukrytej komunikacji. Zgodnie z tą ideą protokoły służące do ukrywania informacji „symulują" wadliwe działanie sieci, którego objawem może być na przykład zwiększenie stopy błędów transmisyjnych lub zwiększenie opóźnień w przekazywanych danych. Dla zewnętrznego obserwatora takie działanie może być uznane jako „normalne", tj. wynikłe z nieidealności funkcjonowania zasobów transmisyjnych, czy też komutacyjnych sieci, a w związku z tym trudne do zdekonspirowania. Stały wzrost złożoności protokołów komunikacyjnych poszerza możliwość stosowania metod steganograficznych opartych na manipulowaniu protokołami i usługami sieciowymi.

Pierwszym systemem, który otworzył zainteresowanie tą ideą był zaproponowany przeze mnie w pracy [24] system HICCUPS (*Hidden Communication System for Corrupted Networks*). W grudniu 2009 Urząd Patentowy RP przyznał patent na system HICCUPS (na podstawie wniosku patentowego z kwietnia 2003).

Przedstawiony jako rozprawa habilitacyjna jednorodny cykl publikacji z lat 2008-11 składa się z 10 artykułów stanowiących, w moim przekonaniu, istotny wkład w rozwój ochrony informacji w sieciach teleinformatycznych. Dwie pierwsze publikacje ([1], [2]) dotyczą steganografii w sieciach WLAN (*Wireless Local Area Network*), cztery kolejne ([3],[4],[5],[6]) telefonii internetowej VoIP (*Voice over IP*), a pozostałe wykorzystaniu retransmisji w protokole TCP (*Transmission Control Protocol*) [7], datagramów IP (*Internet Protocol*) [8], dopełnień w warstwie drugiej modelu odniesienia OSI (*Open System Interconnection*) [9] oraz protokołu SCTP (*Stream Control Transmission Protocol*) [10].

Ogólnie rzecz biorąc, ukrywanie informacji w sieciach jest poważnym zagrożeniem dla bezpieczeństwa we wszystkich warstwach modelu odniesienia OSI, w szczególności w warstwie pierwszej [2], w drugiej ([1], [9]), w trzeciej [8] i w czwartej ([7], [10]). Systemy steganograficzne z warstwy trzeciej i wyższej mają większy geograficzny zasięg działania niż systemy z warstw 1-2, zatem mogą być zastosowane w sieciach rozległych takich jak Internet. Stąd też rodzina protokołów związanych z VoIP ([3], [4], [5], [6]), związana z warstwami 4-7, stanowi jeden z newralgicznych punktów w bezpieczeństwie współczesnej telekomunikacji.

Zbadaniu właściwości metod steganograficznych w dwóch najniższych warstwach OSI są poświęcone prace [1] i [2]. W pierwszej z nich, do analizy wydajności systemu HICCUPS, wykorzystano model matematyczny będący hybrydą autorskiego modelu sieci IEEE 802.11 i przekształcenia geometrycznego ustalającego punkt pracy systemu. W pracy [2] model sieci 802.11 został użyty do oszacowania własności kanału bazującego na mechanizmie dopełnień w OFDM (*Orthogonal Frequency-Division Multiplexing*); wykazano, że kanał ten może mieć bardzo wysoką przepustowość (ok. 1,5 Mbit/s).

Pomysł dotyczący dopełnień ([2]) został rozwinięty w pracy [9], tym razem na poziomie warstwy drugiej OSI. Zaprezentowana technika, przeznaczona dla sieci Ethernet (IEEE 802.3), wykorzystuje relacje między co najmniej dwoma protokołami różnych warstw. W pracy zaproponowano mechanizm „skakania" po protokołach-nośnikach powodujących modyfikację protokołu sterującego występowaniem dopełnień w ramkach. Technika ta tworzy nową klasę w steganografii sieciowej – tzw. steganografię międzyprotokołową.

Kolejnym osiągnięciem wnoszonym przez przedstawiane prace jest wykazanie możliwości ukrywania informacji w obsłudze datagramów IP o zbyt dużym rozmiarze [8], zarówno w przypadku fragmentacji, jak i zastosowaniu metod odkrywania maksymalnej wielkości datagramów. Ten sposób ukrywania komunikacji w warstwie sieciowej jest dostępny zarówno dla protokołu IPv4, jak i IPv6, podczas nawiązywania łączności pomiędzy routerami.

Ideą zaprezentowanego w pracy [7] systemu wykorzystującego protokół TCP jest użycie retransmisji do przesyłania steganogramów. Podobnie jak w systemie HICCUPS, stacje retransmitujące segmenty TCP w celach steganograficznych symulują uszkodzenie sieci powodujące celowe unikanie potwierdzeń. Użycie retransmisji w TCP otwiera nową klasę w steganografii sieciowej – steganografię hybrydową.

Istotnym zagadnieniem jest wszechstronne zbadanie metod ukrywania informacji w protokole SCTP dokonane w pracy [10]. Protokół SCTP jest używany m.in. do przenoszenia ruchu sygnalizacyjnego w sieciach konwergentnych bazujących na IP (SIGTRAN – *signaling transport*) i jest postrzegany jako potencjalny następca protokołów TCP i UDP. W pracy [10] zawarto analizę siedemnastu nowych metod steganograficznych dla protokołu SCTP i zaprezentowano wnioski zwiększające jego bezpieczeństwo.

Osiągnięciem prezentowanych prac jest spójne przedstawienie zagadnień steganografii w telefonii VoIP ([3], [4], [5], [6]). W szczególności praca [4] zawiera usystematyzowany przegląd metod w tej dziedzinie. Nowatorskim rozwiązaniem jest zaproponowany w [3]

system wykorzystujący celowe opóźnienia pakietów z głosem, którego praktyczne zastosowanie zostało potwierdzone eksperymentami ([5]). Praca [6] skupia się na analizie steganograficznego bezpieczeństwa protokołu SIP. W pracach [3], [4], [5] i [6] wykazano możliwość tworzenia ukrytych kanałów na każdym etapie łączności pomiędzy podmiotami zaangażowanymi w ustanawianie, utrzymywanie i zakończenie rozmowy, co umożliwia wyciek informacji z miejsc uznawanych do tej pory za bezpieczne np. z sieci korporacyjnych chronionych za pomocą standardowych systemów ochrony informacji.

Reasumując, wkład zaprezentowanych w rozprawie publikacji w dziedzinę bezpieczeństwa sieciowego to:

- propozycja nowych skutecznych metod steganograficznych dla różnych protokołów i usług, w tym IEEE 802.11 ([1], [2]), Ethernet [9], IPv4/IPv6 [8], TCP [7], SCTP [10], VoIP ([3], [4], [5], [6]),
- stworzenie spójnej klasyfikacji metod steganograficznych [10],
- wprowadzenie dwóch nowych klas metod steganograficznych: metod hybrydowych [7] i metod międzyprotokołowych [10],
- opracowanie nowych metod analitycznych ([1], [2]) i doświadczalnych [5],
- oraz popularyzacja steganografii sieciowej na świecie [4].

W dalszej części każdy z artykułów wchodzący w skład cyklu jest scharakteryzowany z zaznaczeniem istotnego wkładu do ochrony informacji w sieciach.

A Performance Analysis of HICCUPS – a Steganographic System for WLAN [1]

Artykuł został opublikowany w 2010 roku w czasopiśmie z listy filadelfijskiej *Telecommunication Systems: Modelling, Analysis, Design and Management* wydawnictwa Springer US, był także prezentowany na konferencji *International Conference on Multimedia Information NEtworking and Security* (MINES 2009) w Wuhan (Chiny) [23]. Przedstawiona w artykule analiza wybranych właściwości systemu steganograficznego HICCUPS skupia się na ocenie jego wydajności i kosztu działania. Do badań został użyty oryginalny model 802.11 CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*; por. [19], [20], [21]) oparty na łańcuchach Markowa. Koszt użycia systemu HICCUPS jest rozumiany jako utrata przepustowości użytkowej w sieci 802.11 wynikająca z działania systemu HICCUPS w trybie uszkodzonych ramek. Efektywność systemu HICCUPS jest rozumiana jako przepustowość systemu HICCUPS w trybie uszkodzonych ramek. Badania zostały przeprowadzone dla przypadku skrajnego tj. w stanie, gdy każda ze stacji posiada niepustą kolejkę z oczekującymi do wysłania ramkami. Miarą efektywnej przepustowości sieci w stanie nasycenia jest ruch przenoszony w takich nasycenia. W pracy wykazano, że dla ustalonej liczby stacji i długości ramki, koszt istotnie zależy od poziomu ramkowej stopy błędów wnoszonej do sieci użytkowej przez działanie systemu HICCUPS. Wykazano, że dla ustalonej liczby stacji i długości ramki efektywność zależy wyłącznie od poziomu ramkowej stopy błędów wnoszonej do sieci użytkowej przez działanie systemu HICCUPS.

Istotnym wkładem w dziedzinę ochrony informacji w sieciach jest:

- użycie własnego modelu sieci 802.11 do modelowania wydajności systemu steganograficznego,
- pełne zbadanie własności systemu HICCUPS z wykorzystaniem aparatu matematycznego,
- stworzenie przekształcenia geometrycznego pozwalającego na dokładne określenie punktu pracy systemu HICCUPS w zależności od liczby stacji i bitowej stopy błędów.

Steganography in IEEE 802.11 OFDM Symbols [2]

Artykuł został opublikowany w czasopiśmie z listy filadelfijskiej *International Journal of Security and Communication Networks* wydawnictwa John Wiley & Sons w 2011 i jest rozszerzoną wersją publikacji [22] wygłoszonej na konferencji *International Conference on Multimedia Information NEtworking and Security* (MINES 2010) w Nanjing (Chiny).

W artykule zaprezentowano i przenalizowano nową metodę ukrywania informacji bazującą na bitowym dopełnieniu w symbolach OFDM warstwy fizycznej sieci IEEE 802.11 nazwaną WiPad (*Wireless Padding*). Ze względu na strukturę ramki aż 210 bitów/ramkę może zostać użyte do ukrytej komunikacji. Analiza przeprowadzona przy użyciu modelu bazującego na łańcuchach Markowa zaproponowanego i zwalidowanego w [19], [20], [21] dla sieci IEEE 802.11g (54 Mbit/s) wykazała, że maksymalna przepływność steganograficzna dla WiPad wynosi 1,1 Mbit/s przy wykorzystaniu do celów steganograficznych ramek z danymi oraz 0,44 Mbit/s, gdy wykorzystywane są ramki z potwierdzeniami. Daje to w sumie całkowitą przepływność steganograficzną ok. 1,5 Mbit/s, co według wiedzy autorów jest jednym z największych znanych kanałów steganograficznych.

Istotnym wkładem w dziedzinę ochrony informacji w sieciach jest:

- propozycja nowej metody WiPad wykorzystującej do ukrywania informacji OFDM w 802.11,
- pełne zbadanie własności metody WiPad z wykorzystaniem aparatu matematycznego,
- stworzenie systemu steganograficznego o największej znanej przepustowości.

System WiPad został omówiony w czasopiśmie *Technology Review* wydawanym w *Massachusetts Institute of Technology* (MIT)[1], a także w *IEEE Spectrum*[2], w związku z użyciem steganografii przez rosyjskich szpiegów schwytanych w czerwcu 2010 w USA.

Moim wkładem własnym w artykule była koncepcja systemu WiPad, określenie własności systemu, które są interesujące do zbadania, przeprowadzenie obliczeń na bazie

[1] http://www.technologyreview.com/blog/mimssbits/25455/

[2] http://spectrum.ieee.org/tech-talk/computing/networks/russian-spies-thwarted-by-old-technology

własnego modelu sieci 802.11 i wyciągnięcie wniosków. Byłem także autorem kontaktowym podczas pracy nad ostateczną wersją z wydawnictwem John Wiley & Sons.

Steganography of VoIP Streams [3]

Artykuł został opublikowany jako rozdział w monografii pt. *OTM 2008, Part II* w serii *Lecture Notes in Computer Science* (LNCS), wydawnictwa Springer-Verlag i został wygłoszony na konferencji *OnTheMove Federated Conferences and Workshops: The 3rd International Symposium on Information Security (IS'08),* w Monterrey w Meksyku w 2008 roku. Artykuł był cytowany dziewięć razy (z wyłączeniem autocytowań), w tym w dwóch znanych książkach z dziedziny steganografii ([12], [26]).

Artykuł prezentuje dostępne techniki steganograficzne, które mogą być użyte do tworzenia ukrytych kanałów w strumieniach VoIP. Oprócz usystematyzowanego przedstawienia stanu sztuki, w pracy zaproponowano dwie nowe techniki steganograficzne: pierwszą opartą na protokołach RTP (*Real-Time Transport Protocol*) i RTCP (*Real-Time Control Protocol*), bazującą na wolnych lub opcjonalnych polach oraz drugą – LACK (*Lost Audio Packets Steganography*) wykorzystującą celowe opóźnienia pakietów z głosem.

Dla protokołów RTP oraz RTCP dokonano w artykule wszechstronnej analizy pól nagłówków i wyrażono analitycznie przepustowość dostępną dla ukrytych kanałów. Podobna analiza przepustowości została przeprowadzana dla LACK, jak i też pozostałych metod ukrywania informacji, w tym techniki znaku wodnego (*watermarking*).

W ramach prac dokonano eksperymentu, na podstawie którego wykazano, że w typowej rozmowie VoIP można uzyskać strumień ukrytych danych 2,5 kbit/s, o następujących właściwościach: ponad 96% z tego pasma jest uzyskane za pomocą steganografii bazującej na RTP i RCTP, a także IP/UDP, 2,6% przy wykorzystaniu metody LACK, a 1,2% za pomocą innych metod, w tym watermarkingu.

Istotnym wkładem w dziedzinę ochrony informacji w sieciach jest:

- usystematyzowanie zagadnień związanych z ukrywaniem informacji w VoIP,
- analiza protokołów RTP i RCTP pod kątem ukrytych kanałów,
- propozycja systemu LACK,
- oszacowanie wielkości łącznego strumienia ukrytych informacji podczas rozmowy VoIP.

LACK został zgłoszony do Urzędu Patentowego RP jako wynalazek (zgłoszenie numer P-384940 z 15 kwietnia 2008 na rzecz Politechniki Warszawskiej). Artykuł na temat metody LACK zamieściło prestiżowe czasopismo *New Scientist* w numerze z 31 maja 2008 roku[3].

[3] http://www.newscientist.com/article/mg19826586.000-secret-messages-could-be-hidden-in-net-phone-calls.html

W tym samym czasie metody ukrywania informacji w telefonii IP otrzymały miano **steganofonii**.

Moim wkładem własnym w artykule była praca nad koncepcją systemu LACK, analiza protokołów RTP i RTCP, określenie obszaru interesującego do analizy, przeprowadzenie badań, jak i wyciągnięcie wniosków. Byłem także autorem kontaktowym podczas pracy nad ostateczną wersją, a także wygłaszałem artykuł na konferencji.

Vice over IP [4]

Artykuł został opublikowany w czasopiśmie z listy filadelfijskiej *IEEE Spectrum* w lutym 2010. Wersja elektroniczna jest dostępna na stronie internetowej czasopisma pod nazwą: *Vice Over IP: The VoIP Steganography Threat*[4]. Artykuł jest rozszerzoną i zmienioną wersją artykułu [15] zaprezentowanego w 2008 roku na *26th Army Science Conference* (ASC 2008) w Orlando (USA).

Celem artykułu było stworzenie usystematyzowanego przeglądu metod steganografii sieciowej, w szczególności technik powiązanych z VoIP. Steganografia w VoIP została omówiona w kontekście realnych zagrożeń, takich jak wyciek informacji firmowych oraz komunikacja pomiędzy grupami przestępczymi (w tym pomiędzy terrorystami lub pedofilami). W pracy przedstawiono ponad 2500 letnią historię steganografii ze wskazaniem punktów przełomowych i najbardziej znanych sposobów ukrywania informacji. Ponadto przedstawiono steganografię sieciową w kontekście sieci IP, a następnie techniki VoIP. Zaprezentowano osiągnięcia autorów w omawianej dziedzinie, w tym systemy HICCUPS i LACK. Pracę uzupełniają obrazowe przykłady uzmysławiające wielkość ukrytych kanałów (np. w pojedynczym 6 minutowym pliku audio MP3 o wielkości 30 MB można ukryć tekst dowolnej sztuki Williama Szekspira).

Istotnym wkładem w dziedzinę ochrony informacji w sieciach jest:

- omówienie steganografii z wyróżnieniem punktów przełomowych (także historycznych),
- usystematyzowanie współczesnej steganografii,
- popularyzacja steganografii sieciowej (w tym systemów LACK i HICCUPS) we flagowym czasopiśmie IEEE o największym zasięgu czytelniczym (385 tys. czytelników).

W czasopiśmie IEEE Security & Privacy został opublikowany przez Liam M. Mayron artykuł pt. *Secure Multimedia Communications* [14], który, w odniesieniu do stanu sztuki we współczesnej ochronie informacji w multimediach, referuje 9 pozycji, w tym artykuł *Vice Over IP,* jako jeden z dwóch z zakresu steganografii.

Moim wkładem własnym w artykule była praca nad jego koncepcją, nad syntezą współczesnej steganografii, wyszukanie materiałów faktograficznych związanych z historią

[4] http://spectrum.ieee.org/telecom/internet/vice-over-ip-the-voip-steganography-threat

ukrywania informacji. Byłem także autorem kontaktowym podczas całego procesu pracy nad artykułem.

Covert Channels in SIP for VoIP signalling [5]

Artykuł został opublikowany jako rozdział w monografii pt. *ICGeS 2008* w serii *Communications in Computer and Information Science* (CCIS), wydawnictwa Springer-Verlag i został wygłoszony na konferencji *4th International Conference on Global E-security,* w Londynie w Wielkiej Brytanii w 2008 roku.

W artykule przeanalizowano metody ukrywania informacji w protokole SIP (*Session Initiation Protocol*), który jest obecnie najpopularniejszym protokołem sygnalizacyjnym dla usługi VoIP. Usystematyzowano ukryte kanały na poziomie parametrów, znaczników i pól opcjonalnych SIP, zbadano wykorzystanie pól używanych przez mechanizmy zabezpieczeń oraz zawartości przenoszonej przez protokół SDP (*Session Description Protocol*). Przedstawiono też analitycznie wielkość kanału opartego na SIP (2,4 kbit podczas inicjacji połączenia).

Istotnym wkładem w dziedzinę ochrony informacji w sieciach jest:

- analiza protokołu SIP pod kątem ukrytych kanałów w zastosowaniach związanych z VoIP,
- oszacowanie wielkości strumienia ukrytych informacji zbudowanego na SIP.

Moim wkładem własnym w artykule była idea, praca nad analizą protokołów RTP i RTCP, określenie obszaru interesującego do zbadania, przeprowadzenie badań, jak i wyciągnięcie wniosków. Byłem także autorem kontaktowym podczas pracy nad ostateczną wersją, a także wygłaszałem artykuł na konferencji.

What are suspicious VoIP delays? [6]

Artykuł został opublikowany w czasopiśmie z listy filadelfijskiej *Multimedia Tools and Applications* wydanym w 2010 przez wydawnictwo Springer US.

Badania opisane w artykule dotyczyły odpowiedzi na tytułowe pytanie: jakiego typu opóźnienia w komunikacji VoIP są podejrzane, a jakie można uznać za normalne. Zjawisko opóźnień w VoIP przeanalizowano także w kontekście strat, które są konsekwencją zarówno zaginięć pakietów w kanale (fizyczne straty), jak i nieakceptowalnych opóźnień, które prowadzą do odrzucenia przez bufor odbiorczy. W trakcie badań dokonano eksperymentu łącząc w sieci Internet hosty w dwóch lokalizacjach – w Warszawie i w Oxford w Wielkiej Brytanii. Zmieniano wielkość bufora (od 20 do 120 ms z krokiem 20 ms) i badano jakość połączenia za pomocą obiektywnej metody oceny jakości głosu E-model, opracowanej przez ITU-T. Mając wyznaczone charakterystyki jakości kanałów dla różnej wielkości bufora zbadano możliwość użycia steganografii opartej na zmianie zależności czasowych między pakietami strumienia RTP

(w tym w metodzie LACK). Badania wykazały, że jedynie część metod, w tym LACK, nadaje się do praktycznego wykorzystania.

Istotnym wkładem w dziedzinę ochrony informacji w sieciach jest:

- stworzenie środowiska do badań opóźnień w VoIP,
- przeprowadzenie badań jakości głosu oraz opóźnień pakietów w strumieniach RTP dla usługi VoIP w sieci Internet, pod kątem ukrywania informacji,
- wykazanie praktycznej możliwości stworzenia systemów steganograficznych opartych na zmianie zależności czasowych między pakietami.

Wyniki opublikowane w artykule zostały omówione w czasopiśmie *Technology Review* wydawanym w MIT[5].

Moim wkładem własnym w artykule była praca nad koncepcją, zdefiniowanie problemu badawczego, praca nad środowiskiem testowym, przeprowadzenie eksperymentów i wyciągnięcie wniosków.

RSTEG: Retransmission Steganography and Its Detection [7]

Artykuł został opublikowany w czasopiśmie z listy filadelfijskiej *Soft Computing – A Fusion of Foundations, Methodologies and Applications* wydawnictwa Springer Verlag w wersji elektronicznej w 2009 roku (w wersji papierowej w 2011). Artykuł jest rozszerzoną wersją publikacji [17], przedstawionej na konferencji MINES 2009 w Wuhan (Chiny).

Artykuł przedstawia system RSTEG (*Retransmission Steganography*), którego główną ideą jest celowe aktywowanie retransmisji i przesłanie steganogramu w polu danych retransmitowanej wiadomości. W pracy przedstawiono klasyfikację systemów steganografii sieciowej, która, obok znanych wcześniej klas (modyfikacja struktury pakietów i modyfikacja strumienia pakietów), wprowadza trzecią klasę – systemy hybrydowe. RSTEG, podobnie jak LACK, jest systemem hybrydowym, a więc wpływającym na protokół zarówno w zakresie o zawartości jednostek danych, jak i w zakresie zależności czasowych pomiędzy nimi. W pracy przedstawiono wyniki wszechstronnych badań nad systemem RSTEG w kontekście protokołu TCP, przedstawiając różne scenariusze jego działania, a także odmiany wynikające z różnych wariantów retransmisji w TCP (RTO – *retransmission time-outs*, FR/R – *fast retransmit/recovery*, SACK – *selective acknowledgment*). Ocenę jakości systemu RSTEG oparto na symulacjach w środowisku *ns-2*, które potwierdziły wysoką efektywność.
W artykule dokonano też analizy bezpieczeństwa systemu w kontekście steganografii wskazując, że największa niewykrywalność jest osiągana dla mechanizmów typu RTO, a największa wydajność dla mechanizmów typu SACK.

[5] http://www.technologyreview.com/blog/arxiv/24855/

Istotnym wkładem w dziedzinę ochrony informacji w sieciach jest:

- klasyfikacja metod steganografii sieciowej – wyróżnienie nowej klasy: metody hybrydowe,
- propozycja nowej metody – RSTEG, wykorzystującej do ukrywania informacji retransmisje,
- pełne zbadanie własności metody RSTEG dla kliku wariantów retransmisji w TCP za pomocą technik symulacyjnych.

Podobnie jak LACK, rozwiązanie to cieszyło się dużą popularnością medialną, m.in. opis tej metody steganograficznej w *New Scientist* z 26 maja 2009 roku[6]. Rozszerzona o implementację wersja artykułu została zaakceptowana do publikacji w czasopiśmie z listy filadelfijskiej *Telecommunication Systems: Modelling, Analysis, Design and Management* wydawnictwa Springer US [16].

Moim wkładem własnym w artykule była praca nad koncepcją systemu RSTEG i jego wariantami, zdefiniowanie problemu badawczego, nadzór nad tworzeniem środowiska badawczego i nad przeprowadzonymi symulacjami, a także wyciągnięcie wniosków.

Evaluation of steganographic methods for oversized IP packets [8]

Artykuł został opublikowany w 2010 roku, w czasopiśmie z listy filadelfijskiej *Telecommunication Systems: Modelling, Analysis, Design and Management* wydawnictwa Springer US. Jest to rozszerzona wersja artykułu prezentowanego na konferencji MINES 2009 w Wuhan (Chiny) [18].

W artykule przedstawiono zagadnienia związane z ukrywaniem informacji w protokołach, które wykorzystują mechanizmy służące do obsługi pakietów IP o zbyt dużych rozmiarach: fragmentację, PMTUD (*Path MTU Discovery*) oraz PLPMTUD (*Packetization Layer Path MTU Discovery*). Po przeanalizowaniu tych mechanizmów, zaproponowano dwie nowe metody, a także trzy rozszerzenia już istniejących.

Pierwsza z nowych metod znajduje zastosowanie we fragmentacji pakietów IP i bazuje na liczbie podzielonych fragmentów. Drugi ze sposobów, dla protokołu PMTUD, polega na sztucznym obniżaniu maksymalnej wielkości pakietu, który może zostać przesłany. Dla PLPMTUD, jako odpornego na ataki steganograficzne omówione dla PMTUD, zaproponowano użycie systemu RSTEG.

Istotnym wkładem w dziedzinę ochrony informacji w sieciach jest:

- analiza pod kątem ukrytych kanałów protokołów wykorzystujących mechanizmy służące do obsługi pakietów IP o zbyt dużych rozmiarach,
- zaproponowanie dwóch nowych metod steganograficznych,

[6] http://www.newscientist.com/article/mg20227096.200-fake-web-traffic-can-hide-secret-chat.html

- oszacowanie wielkości strumienia ukrytych informacji w tych metodach.

Moim wkładem własnym w artykule była praca nad koncepcją, zdefiniowanie problemu badawczego, nadzór nad stworzeniem środowiska badawczego i nad przeprowadzonymi eksperymentami, a także wyciągnięcie wniosków.

PadSteg: Introducing Inter-Protocol Steganography [9][7]

Artykuł został zaakceptowany do czasopisma z listy filadelfijskiej *Telecommunication Systems: Modelling, Analysis, Design and Management* wydawnictwa Springer US. Jest rozszerzoną wersją publikacji [13] wygłoszonej na *14th International Telecommunications Network Strategy and Planning Symposium (Networks 2010)* w Warszawie.

W artykule zaproponowano nowy system steganograficzny PadSteg (*Padding Steganography*), który do przesyłania ukrytych informacji w sieciach LAN wykorzystuje niepoprawnie dopełniane ramki ethernetowe. Dotychczasowe rozwiązania steganografii sieciowej wykorzystywały jedynie modyfikacje w odniesieniu do jednego protokołu (zawartości jego jednostek danych lub relacji czasowych pomiędzy nimi). PadSteg jest pierwszym rozwiązaniem, które do funkcjonowania wykorzystuje relacje między co najmniej dwoma protokołami różnych warstw modelu odniesienia OSI. Nowa klasa tego typu rozwiązań została nazwana steganografią międzyprotokołową (*Interprotocol Steganography*). Dodatkowo zaproponowano mechanizm skakania po protokołach-nośnikach (*carrier-protocol hopping*), który pozwala na zmianę protokołu powodującego występowanie dopełnienia w ramkach ethernetowych (TCP/ARP/ICMP/UDP), co znacznie utrudnia detekcję. Na bazie wykonanego eksperymentu oszacowano przepływność steganograficzną zaproponowanego system (27 bit/s) oraz jego niewykrywalność. Ramki zawierające steganograficzne dane imitują ramki rzeczywistych protokołów (TCP/ARP/ICMP/UDP), dlatego metody detekcji są znacznie utrudnione.

Istotnym wkładem w dziedzinę ochrony informacji w sieciach jest:

- propozycja nowej metody PadSteg, wykorzystującej do ukrywania informacji dopełnienie w warstwie 2 modelu OSI,
- zaproponowanie mechanizmu skakania po protokołach nośnikach,
- zidentyfikowanie nowej klasy protokołów steganograficznych, tzw. steganografii międzyprotokołowej,
- zbadanie własności metody za pomocą praktycznych doświadczeń.

Moim wkładem własnym w artykule była praca nad koncepcją systemu PadSteg, zdefiniowanie problemu badawczego, nadzór nad stworzeniem środowiska eksperymentalnego i nad przeprowadzonymi badaniami, a także wyciągnięcie wniosków.

[7] Artykuł dostępny też jako preprint: http://arxiv.org/abs/1104.0422

Byłem także autorem kontaktowym przy pracy nad ostateczną wersją z wydawnictwem Springer-Verlag.

Stream Control Transmission Protocol Steganography [10]

Artykuł został zaprezentowany na konferencji MINES 2010 w Nanjing (Chiny). Rozszerzona wersja [11] została wysłana do czasopisma z listy filadelfijskiej *Computer Communications* wydawnictwa Elsevier i jest obecnie w recenzji.

Protokół SCTP uważany jest za potencjalnego następcę najpopularniejszych obecnie protokołów warstwy transportowej, czyli TCP i UDP. W artykule opisano metody steganograficzne dla protokołu SCTP, które mogą stanowić zagrożenie dla bezpieczeństwa sieciowego, w tym 17 nowych metod. Zaproponowane metody wykorzystują nowe, charakterystyczne dla tego protokołu cechy, takie jak obsługa *multi-homingu* czy wielostrumieniowość. Przedstawione zagrożenia, a w szczególności sugerowane sposoby zapobiegania im, mogą być potraktowane jako suplement do dokumentu RFC 5062, w którym opisano podatności SCTP na ataki sieciowe.

Istotnym wkładem w dziedzinę ochrony informacji w sieciach jest:

- wnikliwa analiza protokołu SCTP pod kątem ukrytych kanałów,
- propozycja 17 metod ukrywania informacji,
- sformułowanie istotnych uwag do protokołu zwiększających istotnie jego bezpieczeństwo.

Moim wkładem własnym w artykule była praca nad jego koncepcją, zdefiniowanie problemu badawczego, nadzór nad stworzeniem środowiska badawczego i nad przeprowadzonymi eksperymentami, a także wyciągnięcie wniosków. Byłem także autorem kontaktowym podczas pracy nad ostateczną wersją, a także wygłaszałem artykuł na konferencji.

Cykl publikacji

[1] Szczypiorski, K.: *A Performance Analysis of HICCUPS – a Steganographic System for WLAN*. W: Telecommunication Systems: Modelling, Analysis, Design and Management, Volume 49: 3-4 – March/April 2012, Springer US, Journal no. 11235 (wersja elektroniczna dostępna, wersja papierowa w druku)

[2] Szczypiorski, K., Mazurczyk, W.: *Steganography in IEEE 802.11 OFDM Symbols*. W: International Journal of Security and Communication Networks, 2011, John Wiley & Sons (wersja elektroniczna dostępna, wersja papierowa w druku)

[3] Mazurczyk, W., Szczypiorski, K.: *Steganography of VoIP Streams*. W: Robert Meersman and Zahir Tari (Eds.): OTM 2008, Part II – Lecture Notes in Computer Science (LNCS) 5332, Springer-Verlag Berlin Heidelberg, Proc. of OnTheMove Federated Conferences and Workshops: The 3rd International Symposium on Information Security (IS'08), Monterrey, Mexico, November 9-14, 2008, str. 1001-1018

[4] Lubacz, J., Mazurczyk, W., Szczypiorski K.: *Vice over IP*. W: IEEE Spectrum, Volume: 47 Issue: 2, February 2010, str. 42-47

[5] Mazurczyk, W., Szczypiorski, K.: *Covert Channels in SIP for VoIP signaling*. W: Hamid Jahankhani, Kenneth Revett, and Dominic Palmer-Brown (Eds.): ICGeS 2008 – Communications in Computer and Information Science (CCIS) 12, Springer Verlag Berlin Heidelberg, Proc. of 4th International Conference on Global E-security 2008, London, United Kingdom, 23-25 June 2008, str. 65-72

[6] Mazurczyk, W., Cabaj, K., Szczypiorski, K.: *What are suspicious VoIP delays?* W: Multimedia Tools and Applications, 2010, Springer US, Journal no. 11042 (wersja elektroniczna dostępna, wersja papierowa w druku)

[7] Mazurczyk, W., Smolarczyk, M., Szczypiorski K.: *RSTEG: Retransmission Steganography and Its Detection*. W: Soft Computing – A Fusion of Foundations, Methodologies and Applications, Springer Verlag, Volume 15, Number 3, March 2011, str. 505-515

[8] Mazurczyk, W., Szczypiorski, K.: *Evaluation of steganographic methods for oversized IP packets*. W: Telecommunication Systems: Modelling, Analysis, Design and Management, Volume 49: 3-4 – March/April 2012, Springer US, Journal no. 11235 (wersja elektroniczna dostępna, wersja papierowa w druku)

[9] Jankowski, B., Mazurczyk, W., Szczypiorski, K.: *PadSteg: Introducing Inter-Protocol Steganography*. Zaakceptowane do: Telecommunication Systems: Modelling, Analysis, Design and Management, Springer US, Journal no. 11235

[10] Frączek, W., Mazurczyk, W., Szczypiorski, K.: *Stream Control Transmission Protocol Steganography*. W: The 2010 International Conference on Multimedia Information Networking and Security (MINES 2010) – Second International Workshop on Network Steganography (IWNS 2010), Nanjing, China, November 4-6, 2010, str. 829-834

Literatura uzupełniająca

[11] Frączek, W., Mazurczyk, W., Szczypiorski, K.: *Hiding Information in Stream Control Transmission Protoco*l. W recenzji: Computer Communications, Elsevier

[12] Fridrich, J.: *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press; First edition (12 Nov 2009)

[13] Jankowski, B., Mazurczyk, W., Szczypiorski, K.: *Information Hiding Using Improper Frame Padding*. Materiały: 2010 14th International Telecommunications Networks Strategy and Planning Symposium (NETWORKS), ISBN 978-1-4244-6703-7, 27-30 September 2010, str. 77-82

[14] Mayron, L.M.: *Secure Multimedia Communications*. W: IEEE Security & Privacy, Nov.-Dec. 2010, Volume: 8 Issue:6, str. 76-79

[15] Mazurczyk, W., Lubacz, J., Szczypiorski, K.: *Hiding Data in VoIP*. Materiały: The 26th Army Science Conference (ASC 2008), Orlando, Florida, USA, December 1-4, 2008

[16] Mazurczyk, W., Smolarczyk, M., Szczypiorski K.: *On Information Hiding in Retransmissions*. Zaakceptowane do: Telecommunication Systems: Modelling, Analysis, Design and Management, Volume 49: 3-4 – March/April 2012, Springer US, Journal no. 11235
[17] Mazurczyk, W., Smolarczyk, M., Szczypiorski, K.: *Retransmission Steganography Applied*. W: The 2010 International Conference on Multimedia Information Networking and Security (MINES 2010) – Second International Workshop on Network Steganography (IWNS 2010), Nanjing, China, November 4-6, 2010, str. 846-850
[18] Mazurczyk, W., Szczypiorski, K.: *Steganography in Handling Oversized IP Packets*. Materiały: 2009 International Conference on Multimedia Information NEtworking and Security (MINES 2009) – First International Workshop on Network Steganography (IWNS'09), Wuhan, Hubei, China, 18-20 November, 2009, Vol. I, str. 559-564
[19] Szczypiorski, K., Lubacz, J.: *Performance Analysis of IEEE 802.11 DCF Networks*. W: Journal of Zhejiang University – Science A, ISSN 1673-565X (print version) 1862-1775 (electronic version), Zhejiang University Press, co-published with Springer-Verlag GmbH, Vol. 9, No. 10, October 2008. str. 1309-1317
[20] Szczypiorski, K., Lubacz, J.: *Performance Evaluation of IEEE 802.11 DCF Networks*. W: Lorne Mason, Tadeusz Drwiega, and James Yan (Eds.) – Managing Traffic Performance in Converged Networks – Lecture Notes in Computer Science (LNCS) 4516, Springer-Verlag Berlin Heidelberg, Proc. of 20th International Teletraffic Congress – ITC-20, Ottawa, Canada, June 17-21, 2007. str. 1084-1095
[21] Szczypiorski, K., Lubacz, J.: *Saturation Throughput Analysis of IEEE 802.11g (ERP-OFDM) Networks*. Telecommunication Systems: Modelling, Analysis, Design and Management, ISSN: 1018-4864 (print version), ISSN: 1572-9451 (electronic version), Springer US, Journal no. 11235 Vol. 38, Numbers 1-2, June, 2008. str. 45-52
[22] Szczypiorski, K., Mazurczyk, W.: *Hiding Data in OFDM Symbols of IEEE 802.11 Networks*. Materiały: The 2010 International Conference on Multimedia Information Networking and Security (MINES 2010) – Second International Workshop on Network Steganography (IWNS 2010), Nanjing, China, November 4-6, 2010, str. 835-840
[23] Szczypiorski, K.: *A Performance Analysis of HICCUPS – a Steganographic System for WLAN*. Materiały: 2009 International Conference on Multimedia Information NEtworking and Security (MINES 2009) – First International Workshop on Network Steganography (IWNS'09), Wuhan, Hubei, China, 18-20 November, 2009, Vol. I, str. 569-572
[24] Szczypiorski, K.: *HICCUPS: Hidden Communication System for Corrupted Networks*. Materiały: The Tenth International Multi-Conference on Advanced Computer Systems ACS'2003, October 22-24, 2003 Międzyzdroje, str. 31-40
[25] Szczypiorski, K.: *Steganography in TCP/IP Networks. State of the Art and a Proposal of a New System – HICCUPS*. Prezentacja na seminarium w Instytucie Telekomunikacji, Politechnika Warszawska, 4 listopada 2003 [WWW: http://ksz.tele.pw.edu.pl/pdf/steg-seminar-2003.pdf]
[26] Wayner, P.: *Disappearing Cryptography*. Morgan Kaufmann; Third Edition (December 17, 2008)

Telecommun Syst
DOI 10.1007/s11235-010-9363-6

A performance analysis of HICCUPS—a steganographic system for WLAN

Krzysztof Szczypiorski

Abstract The paper presents an analysis of performance features of the HICCUPS (*HIdden Communication system for CorrUPted networkS*) including the efficiency and the cost of the system in WLANs (*Wireless Local Area Networks*). The analysis relies on the original CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*) 802.11 Markov chain-based model and proves that the HICCUPS is the efficient steganographic method with the reasonable cost.

Keywords Steganography · Network security · Wireless LAN · IEEE 802.11

1 Introduction

The HICCUPS (*HIdden Communication system for CorrUPted networkS*), introduced by the author in [6], is a steganographic system for WLANs (*Wireless Local Area Networks*). The main innovation of the system is usage of frames with intentionally wrong checksums to establish covert communication. The HICCUPS was recognized [1] as the first steganographic system for WLAN.

The analysis presented in this paper focuses on some performance features of the HICCUPS, including the efficiency and the cost of the system usage in WLAN. For the purpose of this analysis the Markov chain-based model was used which is dedicated for 802.11 CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*; [2–5, 7]). The cost of system usage (κ) is defined as a decline of WLAN throughput that results from the HICCUPS operating in the corrupted frame mode [6]. The efficiency of the system (ε) is defined as a throughput of the system in the corrupted frame mode.

The evaluation was performed for the saturated condition i.e. when all stations involved in communications have no empty queues. Saturation throughout (S) is an efficiency measure of maximum load in saturated conditions.

2 The analysis of saturation throughput for the corrupted frame mode—S_H

2.1 Calculation of S_H

First we evaluate the saturation throughput for the HICCUPS in the corrupted frame mode (S_H). The analysis is similar to effort done for the 802.11 CSMA/CA networks in [2–5, 7].

Figure 1 illustrates four states of the channel that could occur during the corrupted frame mode. In this mode all 802.11 frames have incorrect value of CRC-32 code deliberately set in the FCS field (*Frame Checksum Control*). Thus, there are no positive acknowledgments through ACK (*ACKnowledgment*) frames, and therefore "*ACK error*" state is omitted [2–5, 7]. The "*success*" of the transmission in the HICCUPS, not defined in the same way as for the 802.11 network, means that during transmission there were no collisions and no data errors. The mechanism of frame integrity for the HICCUPS is separate from 802.11 FCS.

The duration of four states are as following (Fig. 1):

T_{I_H}—idle slot,

T_{S_H}—successful transmission,

This work is based on the author's PhD thesis [7].

K. Szczypiorski (✉)
Institute of Telecommunications, Warsaw University of Technology, Warsaw, Poland
e-mail: ksz@tele.pw.edu.pl

Published online: 29 June 2010

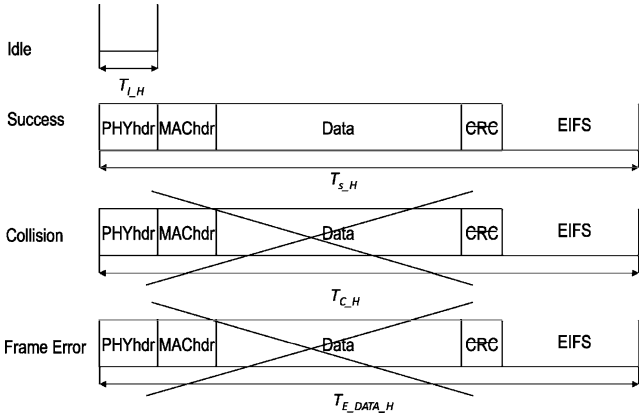


Fig. 1 States of the channel

T_{C_H}—transmission with collision,

$T_{E\_DATA\_H}$—unsuccessful transmission with data frame error.

So we have:

$$\begin{cases} T_{I\_H} = \sigma \\ T_{S\_H} = T_{PHYhdr} + T_{DATA} + \delta + T_{EIFS} \\ T_{C\_H} = T_{S\_H} \\ T_{E\_DATA\_H} = T_{S\_H} \end{cases} \tag{1}$$

Probabilities corresponding to states of the channel are denoted as follows:

P_{I_H}—probability of idle slot,

P_{S_H}—probability of successful transmission,

P_{C_H}—probability of collision,

$P_{E\_DATA\_H}$—probability of unsuccessful transmission due to data frame error.

Let τ_H be a probability of frame transmission in the corrupted frame mode, p_{e_data} a probability of data frame error (see the formula (22) in [3]). These are related to channel state probabilities as follows (see (12) in [3]):

$$\begin{cases} P_{I\_H} = (1-\tau_H)^n \\ P_{S\_H} = n\tau_H(1-\tau_H)^{n-1}(1-p_{e\_data}) \\ P_{C\_H} = 1-(1-\tau_H)^n - n\tau(1-\tau_H)^{n-1} \\ P_{E\_DATA\_H} = n\tau_H(1-\tau_H)^{n-1}p_{e\_data} \end{cases} \tag{2}$$

We use the same assumptions as stated in Chap. 2.1 of [3] so we could express S_H (similar to (6) in [3]):

$$S_H = \frac{P_{S\_H}L_{pld}}{T_{I\_H}P_{I\_H} + T_{S\_H}P_{S\_H} + T_{C\_H}P_{C\_H} + T_{E\_DATA\_H}P_{E\_DATA\_H}}, \tag{3}$$

where L_{pld} is a length of data in frame with FCS field, expressed in bps. S_H could be normalized to R—the rate of the 802.11 network (see formula (7) in [3]):

$$\overline{S}_H = \frac{S_H}{R} \tag{4}$$

2.2 Probability of frame transmission in the corrupted frame mode—τ_H

Based on the model presented and evaluated in [2–5, 7] let us consider a model of the 802.11 CSMA/CA backoff procedure in corrupted frame mode. From a WLAN perspective of the HICCUPS, communication always fails, because of absence of proper checksums. Hence transmission of steganograms is performed in every step of the backoff procedure, so we could describe the HICCUPS behaviour with the Markov chain-based model as presented in [2–5, 7] with probability of the failure $p_f = 1$ (means "always failure").

The state of the two-dimensional process $(s(t), b(t))$ will be denoted as (i, k) [2–5, 7], $b_{i,k}$ is a probability of this state. The one-step conditional state transition probabilities will be denoted by $P = (\cdot, \cdot|\cdot, \cdot)$.

Non-full transition probabilities are determined as follows:

$$\begin{cases} P(i,k|i,k+1) = 1 - p_{coll}, \\ \quad 0 \le i \le m, 0 \le k \le W_i - 2 \\ P(i,k|i,k) = p_{coll}, \\ \quad 0 \le i \le m, 1 \le k \le W_i - 1 \\ P(i,k|i-1,0) = 1/W_i, \\ \quad 0 \le i \le m, 0 \le k \le W_i - 1 \\ P(0,k|m,0) = 1/W_0, \\ \quad 0 \le k \le W_0 - 1 \end{cases} \tag{5}$$

where p_{coll} is a probability of collision, W_0 is an initial size of th contention window and m' is a maximum number by which the contention window may be doubled; m' may be both greater and smaller than m and also equal to m. W_i is the maximum value of a backoff timer at the i backoff stage:

$$W_i = \begin{cases} 2^i W_0, & i \le m' \\ 2^{m'} W_0 = W_m, & i > m' \end{cases} \tag{6}$$

With transition probabilities as above (5) and justifications as in [2, 3, 7], Markov chain transitions is presented in Fig. 2. Let us notice that differences between this diagram and the 802.11 CSMA/CA diagram [2–5, 7] of returns to states $(0, k)$ for $0 \le k \le W_0 - 1$ and $(i, 0)$ for $0 \le i \le m - 1$—this is a graphical interpretation of "always failure" from the perspective of WLAN.

For $0 \le i \le m$ we have:

$$b_{i,k} = \begin{cases} \frac{W_i - k}{W_i(1-p_{coll})} b_{0,0}, & 0 < k \le W_i - 1 \\ b_{0,0}, & k = 0 \end{cases} \tag{7}$$

Because

$$\sum_{i=0}^{m} b_{i,0} = b_{0,0}(m+1) \tag{8}$$

and (7) we get:

$$1 = \sum_{i=0}^{m} \sum_{k=1}^{W_i - 1} b_{i,k} + \sum_{i=0}^{m} b_{i,0}$$
$$= \frac{b_{0,0}}{1 - p_{coll}} \sum_{i=0}^{m} \frac{W_i - 1}{2} + b_{0,0}(m+1) \tag{9}$$

and

$$b_{0,0}^{-1} = \begin{cases} \frac{W_0(2^{m+1}-1)-(m+1)}{2(1-p_{coll})} + (m+1), \\ \quad m \le m' \\ \frac{W_0(2^{m'+1}-1)-(m+1)+(m-m')W_0 2^{m'}}{2(1-p_{coll})} + (m+1), \\ \quad m > m' \end{cases} \tag{10}$$

Having $b_{0,0}$ we may calculate (similar to [2–5, 7]) probability of frame transmission in the corrupted frame mode:

$$\tau_H = \sum_{i=0}^{m} b_{i,0}$$
$$= \begin{cases} (\frac{W_0(2^{m+1}-1)-(m+1)}{2(1-p_{coll})} + (m+1))^{-1}(m+1), \\ \quad m \le m' \\ (\frac{W_0(2^{m'+1}-1)-(m+1)+(m-m')W_0 2^{m'}}{2(1-p_{coll})} \\ \quad + (m+1))^{-1}(m+1), \quad m > m' \end{cases} \tag{11}$$

Probability p_{coll}, similar to the formula (25) in [3] is:

$$p_{coll} = 1 - (1 - \tau_H)^{n-1}. \tag{12}$$

Equations (10) and (11) form a system with two unknown variables τ_H and p_{coll} which may be solved numerically.

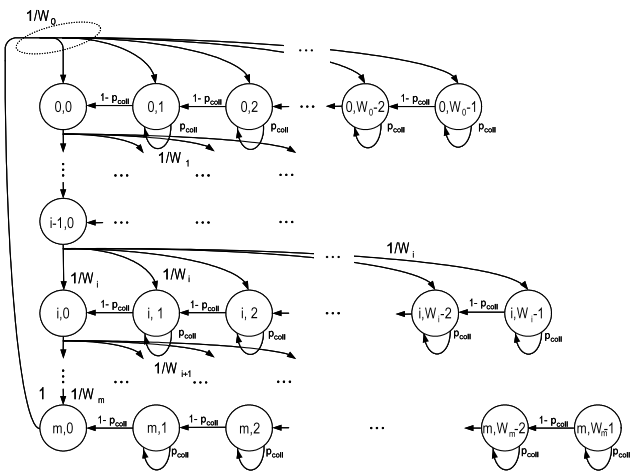


Fig. 2 Markov chain transitions

3 The cost—κ

According to the definition of the cost (κ), introduced in the first part of this paper, the cost is the difference between S, for frame error rate without the HICCUPS, and S, with frame error rate as a result of the HICCUPS in the corrupted frame mode. In other words κ is a decline of WLAN throughput grabbed by HICCUPS hidden channels.

Let us assume that the HICCUPS increases frame error rate by the constant value ΔFER (Fig. 3) and frame error rate of the networks without the HICCUPS equals FER'. We could notice that $0 \le \Delta FER \le 1 - FER'$. So we could express the cost as:

$$\kappa = S(FER') - S(FER' + \Delta FER) \tag{13}$$

and normalized to R:

$$\overline{\kappa} = \frac{\kappa}{R}. \tag{14}$$

The curves of the cost are based on $S(FER)$ and they look almost linear [7], so for small values of ΔFER we could use the following approximation formula (Fig. 4):

$$\kappa \approx \frac{\Delta FER}{1 - FER'} RPN_{WLAN}(FER'). \tag{15}$$

In Tables 1 and 2 the values of the cost κ for $n = 5$ and $n = 10$ are presented for IEEE 802.11g (ERP-OFDM) 54 Mbps—[4, 5]. These results, for $L = 1000$ bytes, come from (15), and were calculated for $FER' \in \{0; 0.0769; 0.5507\}$

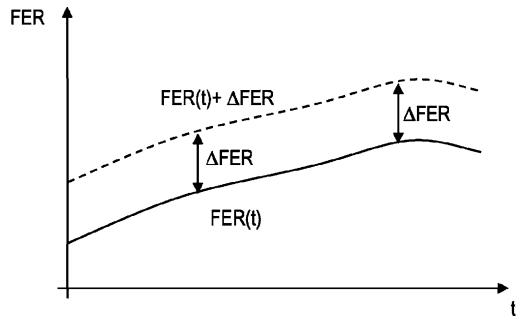


Fig. 3 Interpretation of ΔFER

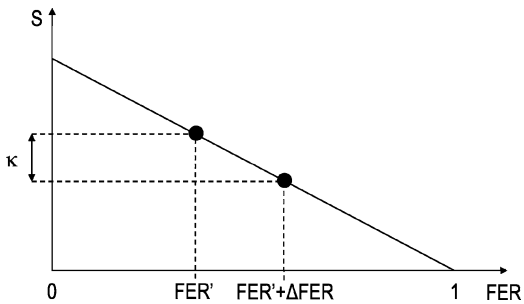


Fig. 4 Graphical presentation of the cost (κ)

Table 1 Normalized values of the cost κ (in brackets expressed in Mbps)—$N = 5$ and $L = 1000$ bytes

FER'	ΔFER				
	0.01	0.02	0.03	0.04	0.05
0	0.0048	0.0097	0.0145	0.0194	0.0242
	(0.26)	(0.52)	(0.78)	(1.05)	(1.31)
0.0769	0.0049	0.0097	0.0146	0.0194	0.0243
	(0.26)	(0.52)	(0.79)	(1.05)	(1.31)
0.5507	0.0047	0.0093	0.0140	0.0186	0.0233
	(0.25)	(0.50)	(0.75)	(1.01)	(1.26)

Table 2 Normalized values of the cost κ (in brackets expressed in Mbps)—$N = 10$ and $L = 1000$ bytes

FER'	ΔFER				
	0.01	0.02	0.03	0.04	0.05
0	0.0046	0.0092	0.0138	0.0184	0.0230
	(0.25)	(0.50)	(0.75)	(1.00)	(1.24)
0.0769	0.0046	0.0093	0.0139	0.0186	0.0232
	(0.25)	(0.50)	(0.75)	(1.00)	(1.25)
0.5507	0.0047	0.0095	0.0142	0.0190	0.0237
	(0.26)	(0.51)	(0.77)	(1.02)	(1.28)

(that corresponds to three bit error rates: $BER \in \{0, 10^{-5}, 10^{-4}\}$). For these conditions five typical values of ΔFER were taken into account (0.01; 0.02; 0.03; 0.04; 0.05).

4 The efficiency—ε

According to the definition of the efficiency (ε), as stated in the introduction, the efficiency is the S_H in conditions that result from physical channel (especially its BER) and amount of frames used by the HICCUPS in the corrupted frame mode. These conditions enable different view on frame error rate from the HICCUPS perspective: the proper frames for the HICCUPS are corrupted for WLAN, and of course the good ones for WLAN in the meaning of the HICCUPS are wrong. So we will use FER_H to emboss this difference, and define ε as follows:

$$\varepsilon = S_H(FER_H) \tag{16}$$

S_H, evaluated in the first part of the paper, allows to calculate the upper boundary of HICCUPS throughput. In the normal use of the HICCUPS the corrupted frame mode occurs very rarely.

To estimate efficiency we might consider two scenarios. In the first scenario: all stations are in the corrupted frame mode only (the HICCUPS is always on): S in the function of FER equals 0 (because $S(1) = 0$), and S_H in the function of FER equals $S_H(FER')$. Because $0 \le \Delta FER \le 1 - FER'$, $\Delta FER = 1 - FER'$. In the second scenario: the HICCUPS is off ($\Delta FER = 0$, only normal transmission is performed, so $S_H = 0$ (because $S_H(1) = 0$), S equals $S(FER')$.

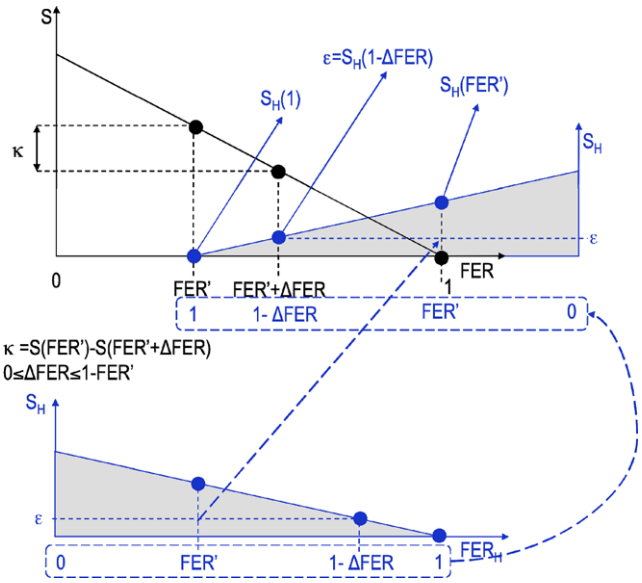


Fig. 5 Graphical interpretation of the efficiency ε

On the base of the two scenarios presented above we could estimate the hypothetic point of the HICCUPS operation for $(FER' + \Delta FER)$ as combination of the translation and the reflection (Fig. 5). The S_H curve is reflected and then translated in FER domain to keep $S(1) = 0$ and $S_H(FER')$ together as well as $S(FER')$ and $S_H(1) = 0$. After this operations we could observe that $FER_H = 1 - \Delta FER$. Finally:

$$\varepsilon = S_H(1 - \Delta FER) \tag{17}$$

and could be normalized to R:

$$\overline{\varepsilon} = \frac{\varepsilon}{R}. \tag{18}$$

Similarly to the analysis of the cost we consider an IEEE 802.11g (ERP-OFDM) 54 Mbps network with 1000 bytes frames, $n \in \{5, 10\}$, and the same values of ΔFER (0.01; 0.02; 0.03; 0.04; 0.05). The results are presented in the Table 3.

5 Conclusions and future work

The analysis presented in this paper focuses on the performance features of the HICCUPS including the efficiency and the cost of system usage in WLAN. The analysis relies on the original Markov chain-based model. The cost depends on the frame error rate, and the efficiency depends

Table 3 Normalized values of the efficiency ε (in brackets expressed in Mbps)—$N \in \{5, 10\}$ and $L = 1000$ bytes

n	ΔFER				
	0.01	0.02	0.03	0.04	0.05
5	0.0042 (0.23)	0.0085 (0.46)	0.0127 (0.69)	0.0169 (0.91)	0.0212 (1.14)
10	0.0047 (0.25)	0.0094 (0.51)	0.0141 (0.76)	0.0188 (1.01)	0.0235 (1.27)

only on ΔFER. As an example for an IEEE 802.11g (ERP-OFDM) 54 Mbps network with 10 stations and $\Delta FER = 0.05$, the efficiency ε equals 1.27 Mbps and the cost κ is 1.28 Mbps. The analysis proves that the HICCUPS is the efficient steganographic method with the reasonable cost.

Future work will focus on the simulation analysis of the HICCUPS to evaluate features of the systems in different scenarios and cover a versatile assessment of the HICCUPS security.

References

1. Krätzer, C., Dittmann, J., Lang, A., & Kühne, T. (2006). WLAN steganography: a first practical review. In *Proc. of 8th ACM Multimedia and Security Workshop*, Geneve (Switzerland), 26–27 September 2006.
2. Szczypiorski, K., & Lubacz, J. (2008). Performance analysis of IEEE 802.11 DCF networks. *Journal of Zhejiang University—Science A*, *9*(10), 1309–1317.
3. Szczypiorski, K., & Lubacz, J. (2007). Performance evaluation of IEEE 802.11 DCF networks. In L. Mason, T. Drwiega, & J. Yan (Eds.), *Lecture notes in computer science (LNCS): Vol. 4516. Managing traffic performance in converged networks* (pp. 1084–1095). Proc. of 20th international teletraffic congress—ITC-20, Ottawa, Canada, 17–21 June 2007. Berlin: Springer.
4. Szczypiorski, K., & Lubacz, J. (2007). Saturation throughput analysis of IEEE 802.11g (ERP-OFDM) networks. In R. Bestak, B. Simak, & E. Kozlowska (Eds.), *IFIP: Vol. 245. Personal wireless communications* (pp. 196–205). Proc. of 12th IFIP international conference on personal wireless communications—PWC'07, Prague, Czech Republic, 12–14 September 2007. Boston: Springer.
5. Szczypiorski, K., & Lubacz, J. (2008). Saturation throughput analysis of IEEE 802.11g (ERP-OFDM) networks. *Telecommunication Systems: Modelling, Analysis, Design and Management*, *38*(12), 45–52.
6. Szczypiorski, K. (2004). HICCUPS: hidden communication system for corrupted networks. In *Proc. the tenth international multiconference on advanced computer systems*. ACS'2003, Międzyzdroje, 22–24 October 2004 (pp. 31–40).
7. Szczypiorski, K. (2006). *Steganography in wireless local area networks*. PhD thesis, Warsaw, September 2006, Warsaw University of Technology (in Polish).

Krzysztof Szczypiorski holds an M.Sc. (1997) and a Ph.D. (2007) in telecommunications both with honours from the Faculty of Electronics and Information Technology, Warsaw University of Technology (WUT), and is an Assistant Professor at WUT. He is the founder and head of the International Telecommunication Union Internet Training Centre (ITU-ITC), established in 2003. He is also a research leader of the Network Security Group at WUT (secgroup.pl). His research interests include network security, steganography and wireless networks. He is the author or co-author of over 110 publications including 65 papers, two patent applications, and 35 invited talks.

SECURITY AND COMMUNICATION NETWORKS
Security Comm. Networks (2011)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/sec.306

SPECIAL ISSUE PAPER

Steganography in IEEE 802.11 OFDM symbols†

Krzysztof Szczypiorski* and Wojciech Mazurczyk

Warsaw University of Technology, Institute of Telecommunications, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland

ABSTRACT

This paper presents a new steganographic method called wireless padding (WiPad). It is based on the insertion of hidden data into the padding of frames at the physical layer of wireless local area networks (WLANs). A performance analysis based on a Markov model, previously introduced and validated by the authors, is provided for the method in relation to the IEEE 802.11 a/g standards. Its results prove that maximum steganographic bandwidth for WiPad is as high as 1.1 Mbit/s for data frames and 0.44 Mbit/s for acknowledgment frames. To the authors' best knowledge this is the most capacious of all the known steganographic network channels. Copyright © 2011 John Wiley & Sons, Ltd.

KEYWORDS

WLAN; IEEE 802.11; information hiding; OFDM; physical layer

***Correspondence**

Krzysztof Szczypiorski, Warsaw University of Technology, Institute of Telecommunications, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland.
E-mail: ksz@tele.pw.edu.pl

1. INTRODUCTION

Network steganography is currently recognized as a new threat to network security that may be used, among others, to enable data exfiltration or also as the way of performing network attacks. Wireless Local Area Networks (WLANs) described in IEEE 802.11 standards were not recognized as a serious area for data hiding especially because of a limited range (for 802.11a/b/g the range is 30 m indoors and 100 m outdoors, for 802.11n the range is doubled). However, IEEE 802.11 was used to transmit secret data among Russian spies hunted down in the USA in June 2010 [1]. From military perspective WLAN is also one of the several ways of communications among soldiers in a battlefield.

In this paper we present and evaluate a new information hiding method based on bit padding of Orthogonal Frequency Division Multiplexing (OFDM) symbols at the physical layer (PHY) of IEEE 802.11 networks. Depending on the transmission data rate at the PHY layer the number of encoded bits per symbol spans from 24 up to 216, therefore as many as 27 octets can be embedded in each OFDM symbol. Due to the specific structure of a frame (described in detail in Section 3) up to 210 bits per frame (26¼ octets/frame) can be allocated for hidden communication. We named this steganographic method utilizing the principle of frame padding in the PHY of WLANs with the acronym Wireless Padding (WiPad).

This paper provides an evaluation of throughput for this method with the aid of our general, Markov-based model introduced and validated in Ref. [2]. This model is in line with the extensions of Bianchi's basic model [3] proposed in Refs. [2,4]. The essential difference with respect to the latter two is the consideration of the effect of freezing of the stations' backoff timer, as well as the limitation of the number of retransmissions and the maximum size of the contention window, and the impact of transmission errors. Results presented in Ref. [2] proved good accuracy of our model in the case of both: error-free and error-prone channels. In either case the proposed model is more accurate than other models presented in literature with which it was compared (including Refs. [2–4]), most notably, when large numbers of stations are under consideration.

This paper is organized as follows. Next section provides an overview of the state of the art with regard to information hiding techniques that utilize padding in

†This is the extended version of the authors' paper entitled ***Hiding Data in OFDM Symbols of IEEE 802.11 Networks*** presented at Second International Workshop on Network Steganography (IWNS 2010) co-located with The 2010 International Conference on Multimedia Information Networking and Security (MINES 2010), Nanjing, China, 4–6 November, 2010.

Table I. Parameters of 802.11 a/g OFDM PHY.

Rate R [Mbit/s]	Modulation	Code rate	Number of bits per symbol – N_{BpS}	Factorization of N_{BpS} into primes
6	BPSK	½	24	$2^3 3$
9	BPSK	¾	36	$2^2 3^2$
12	QPSK	½	48	$2^4 3$
18	QPSK	¾	72	$2^3 3^2$
24	16-QAM	½	96	$2^4 3$
36	16-QAM	¾	144	$2^4 3^2$
48	64-QAM	⅔	192	$2^6 3$
54	64-QAM	¾	216	$2^3 3^3$

WLANs. Section 3 contains a description of our method. Section 4 is a brief overview of the model presented in Ref. [2] and introduces a performance metric for the proposed method. Section 5 presents a performance analysis of the method based on the given model. Finally, Section 6 contains conclusions and suggestions for future work.

2. STATE-OF-THE-ART

Data padding can be found at any layer of the Open System Interconnection Reference Model (OSI RM), but it is typically exploited for covert communications only in the data link, network and transport layers. Wolf proposed in Ref. [5] a steganographic method utilizing padding of 802.3 frames. Its achievable steganographic capacity was maximally 45 bytes/frame. Fisk *et al*. [6] presented padding of the IP and transmission control protocol (TCP) headers in the context of active wardens. Each of these fields offers up to 31 bits/packet for covert communication. Jankowski *et al*. [7] developed a steganographic system, PadSteg, which is based on Ethernet frames' padding and is used in conjunction with address resolution protocol (ARP) and TCP. Padding of IPv6 packets as means for information hiding was described by Lucena *et al*. [8] – offers a couple of channels with a steganographic bandwidth reaching 256 bytes/packet.

Steganography for IEEE 802.11 was proposed by Szczypiorski [9], who postulated the usage of frames with intentionally corrupted checksums to establish covert communication. The system was evaluated by Szczypiorski [10]. Krätzer *et al.* [11] developed a storage channel based scenario (employing header embedding) and a time channel based scenario for IEEE 802.11. Krätzer *et al.* [12] reconsidered the approach presented in Ref. [11].

3. THE METHOD

IEEE 802.11 a/g standards exploit OFDM at the PHY. 802.11 network's PHY layer consists of two sublayers: PHY Layer Convergance Procedure (PLCP) and PHY Medium-Dependent. Selection of a specific transmission data rate at the PHY layer implies functioning with a predefined number of bits corresponding to each OFDM symbol. The number of bits per symbol may vary from 24, for 6 Mbps, up to 216, for 54 Mbps (Table I). Three fields are liable to padding: SERVICE, Physical layer Service Data Unit (PSDU), TAIL (Figure 1). The lengths of SERVICE and TAIL are constant (16 and 6 bits, respectively), while the PSDU is a medium access control (MAC) frame and its length varies depending on user data, ciphers and network operation mode (ad hoc vs. infrastructure).

For each rate R, the number of bits per symbol can be factorized into primes (Table I) and then, using this knowledge, a least common multiple can be calculated as $2^6\ 3^3 = 1728$. This means that the maximum number of padding bytes (octets) that may be used for all rates is:

$$L_\alpha = \frac{2^6 3^3}{8}\alpha - 2 = 216\alpha - 2 \qquad (1)$$

where α is a positive integer.

Please note that padding is present in all frames, therefore frames that are more frequently exchanged, like ACKs may become an interesting target for covert communication.

Typically all padding bits are set to zero [13], but in this paper we assume that all of them could be used for steganographic purposes.

4. THE MODEL

4.1. Assumptions

We considered saturation conditions: stations have non-empty queues and there is always a frame to be sent. The number of stations competing for medium access is n (for $n = 1$ there is one station sending frames to another station which may only reply with an ACK frame). Errors in the transmission medium are fully randomly distributed; this is the worst-case scenario in terms of *frame error rate* (FER). All stations experience the same bit error rate (BER) and all are within each other's transmission range and there are no hidden terminals. Stations communicate in ad hoc mode (basic service set) with basic access method. Every station employs the same PHY. The transmission data rate R is the same and constant for all stations. All frames are of constant length L. The

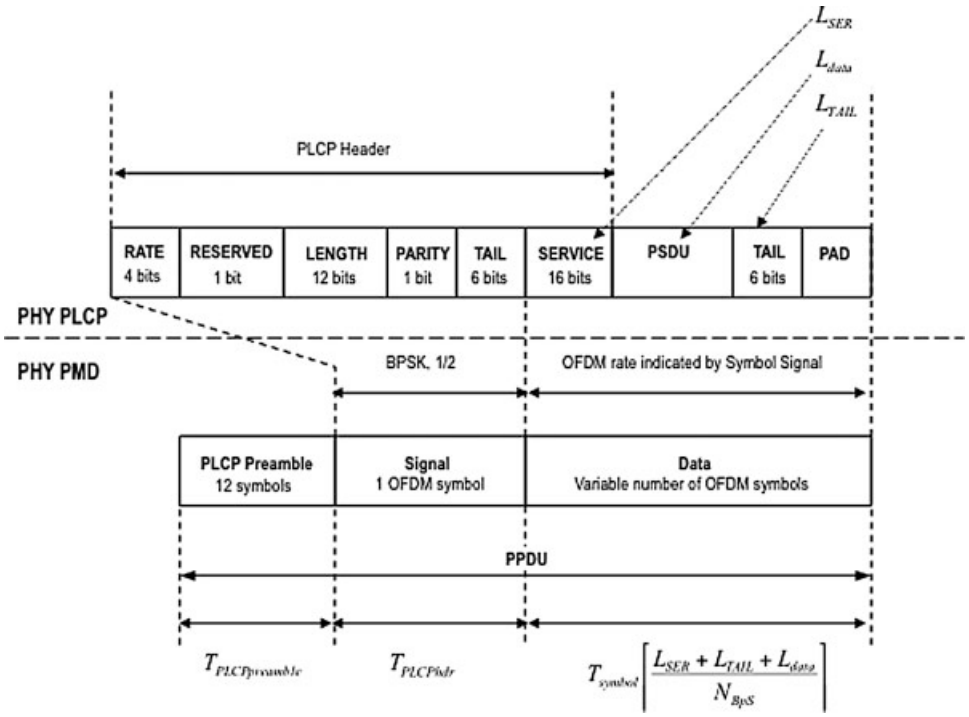


Figure 1. The structure of 802.11a/g PPDU for ERP-OFDM networks.

only frames that are exchanged are data frames and ACK frames. Collided frames are discarded – the capture effect [14] is not considered.

4.2. Saturation throughput *S* expressed through characteristics of the physical channel

The saturation throughput S is defined as in Ref. [2]:

$$S = \frac{E[\text{DATA}]}{E[T]} \tag{2}$$

where E[DATA] is the mean value of successfully transmitted payload, and $E[T]$ is the mean value of the duration of the following *channel states*:

T_{I} – idle slot,
T_{S} – successful transmission,
T_{C} – transmission with collision,
$T_{\text{E\_DATA}}$ – unsuccessful transmission with data frame error,
$T_{\text{E\_ACK}}$ – unsuccessful transmission with acknowledgement (ACK) error.

Figure 2 illustrates dependence of the above channel states on: T_{PHYhdr} – duration of a PLCP preamble and a PLCP header,

T_{DATA} – data frame transmission time,
T_{ACK} – ACK frame duration,
T_{SIFS} – duration of SIFS (short interframe space),
T_{DIFS} – duration of DIFS (DCF interframe space),
T_{EIFS} – duration of EIFS (extended interframe space).

The relation of the saturation throughput to physical channel characteristics is calculated similarly as in Ref. [4]:

$$\begin{cases} T_{\text{I}} = \sigma \\ T_{\text{S}} = 2T_{\text{PHYhdr}} + T_{\text{DATA}} + 2\delta + T_{\text{SIFS}} + T_{\text{ACK}} + T_{\text{DIFS}} \\ T_{\text{C}} = T_{\text{PHYhdr}} + T_{\text{DATA}} + \delta + T_{\text{EIFS}} \\ T_{\text{E\_DATA}} = T_{\text{PHYhdr}} + \delta + T_{\text{DATA}} + T_{\text{EIFS}} \\ T_{\text{E\_ACK}} = T_{\text{S}} \end{cases} \tag{3}$$

where σ is the duration of an idle slot (aSlotTime [13]) and δ is the propagation delay.

For 802.11a/g OFDM PHY (Figure 1):

$$T_{\text{ACK}} = T_{\text{symbol}} \left\lceil \frac{L_{\text{SER}} + L_{\text{TAIL}} + L_{\text{ACK}}}{N_{\text{BpS}}} \right\rceil \tag{4}$$

$$T_{\text{DATA}} = T_{\text{symbol}} \left\lceil \frac{L_{\text{SER}} + L_{\text{TAIL}} + L_{\text{DATA}}}{N_{\text{BpS}}} \right\rceil \tag{5}$$

where:

T_{symbol} – duration of a transmission symbol,
L_{SER} – OFDM PHY layer SERVICE field size,
L_{TAIL} – OFDM PHY layer TAIL field size,
N_{BpS} – number of encoded bits per symbol,
L_{ACK} – size of an ACK frame,
L_{DATA} – size of a data frame.

Values of σ, T_{PHYhdr}, T_{SIFS}, T_{DIFS}, T_{EIFS}, T_{symbol}, N_{BpS}, L_{SER}, and L_{TAIL} are defined in accordance with the 802.11 standard [13].

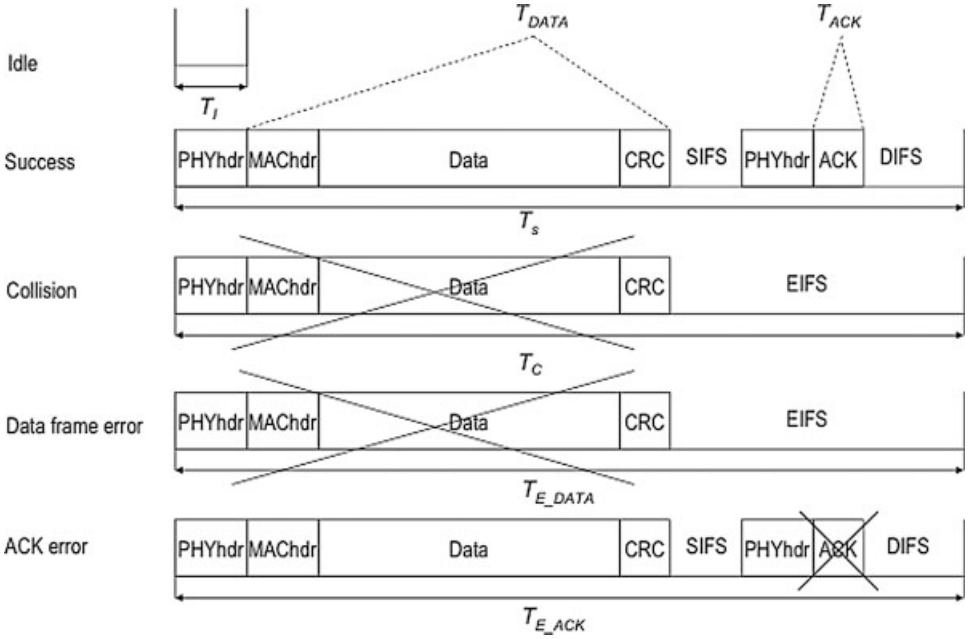


Figure 2. Channel states.

Probabilities corresponding to the states of the channel are denoted as follows:

P_I – probability of an idle slot,
P_S – probability of successful transmission,
P_C – probability of collision,
P_{E_DATA} – probability of unsuccessful transmission due to data frame error,
P_{E_ACK} – probability of unsuccessful transmission due to ACK error.

Let τ be the probability of frame transmission, p_{e_data} the probability of data frame error, and p_{e_ACK} the probability of an ACK error. These are related to channel state probabilities as follows:

$$\begin{cases} P_I = (1-\tau)^n \\ P_S = n\tau(1-\tau)^{n-1}(1-p_{e\_data})(1-p_{e\_ACK}) \\ P_C = 1-(1-\tau)^n - n\tau(1-\tau)^{n-1} \\ P_{E\_DATA} = n\tau(1-\tau)^{n-1}p_{e\_data} \\ P_{E\_ACK} = n\tau(1-\tau)^{n-1}(1-p_{e\_data})p_{e\_ACK} \end{cases} \quad (6)$$

The saturation throughput S equals:

$$S = \frac{P_S L_{pld}}{T_I P_I + T_S P_S + T_C P_C + T_{E\_DATA} P_{E\_DATA} + T_{E\_ACK} P_{E\_ACK}} \quad (7)$$

where L_{pld} is MAC payload size and $L_{pld} = L - L_{MAChdr}$, where L_{MAChdr} is the size of the MAC header plus the size of a frame checksum sequence.

The data rate R is defined as:

$$R = \frac{N_{BpS}}{T_{symbol}} \quad (8)$$

As a result, saturation throughput S is expressed as a function of τ, p_{e_data} and p_{e_ACK}. In the following sections these probabilities are evaluated.

4.3. Probability of frame transmission τ

Let $s(t)$ be a random variable describing DCF backoff stage at time t, with values from set $\{0, 1, 2, \ldots, m\}$. Let $b(t)$ be a random variable describing the value of the backoff timer at time t, with values from the set $\{0, 1, 2, \ldots, W_i - 1\}$. These random variables are correlated because the maximum value of the backoff timer depends on the backoff stage:

$$W_i = \begin{cases} 2^i W_0, & i \le m' \\ 2^{m'} W_0 = W_m, & i > m' \end{cases} \quad (9)$$

where W_0 is the initial size of the contention window (CW) and m' is (the boundary stage above which the contention widow will not be enlarged further); m' can be either greater, smaller or m. W_0 and $W_{m'}$ depend on CW_{min} and CW_{max} [13]:

$$W_0 = CW_{min} + 1 \quad (10)$$

$$W_{m'} = CW_{max} + 1 = 2^{m'} W_0 \quad (11)$$

The two-dimensional process $(s(t), b(t))$ will be analyzed with the aid of an embedded Markov chain (steady state probabilities), whose states correspond to the time instants at which the channel state changes. Let (i,k) denote the current state of this process. The conditional, one-step, state transition probabilities will be denoted by $P = (\cdot,\cdot|\cdot,\cdot)$.

Let p_f be the probability of transmission failure and p_{coll} the probability of collision. The non-null transition probabilities are determined as follows:

$$\begin{array}{ll}
\text{(a) } P(i,k|i,k+1) = 1-p_{coll}, & 0 \le i \le m,\ 0 \le k \le W_i-2 \\
\text{(b) } P(i,k|i,k) = p_{coll}, & 0 \le i \le m,\ 1 \le k \le W_i-1 \\
\text{(c) } P(0,k|i,0) = (1-p_f)/W_0, & 0 \le i \le m-1,\ 0 \le k \le W_0-1 \\
\text{(d) } P(i,k|i-1,0) = p_f/W_i, & 1 \le i \le m,\ 0 \le k \le W_i-1 \\
\text{(e) } P(0,k|m,0) = 1/W_0, & 0 \le k \le W_0-1
\end{array} \tag{12}$$

Ad (a): The station's backoff timer is decremented from $k+1$ to k at a fixed, i-th backoff stage, i.e., the station has detected an idle slot. The probability of this event *Pr{channel is idle}* = 1 − *Pr{one or more stations are transmitting}*. We consider saturation conditions, so *Pr{one or more stations are transmitting}* equals p_{coll}.

Ad (b): The station's backoff timer is frozen at a fixed, i-th backoff stage, i.e., the channel is busy. *Pr{channel is busy}* = *Pr{one or more stations are transmitting}* = p_{coll}.

Ad (c): The station's backoff timer is changed from 0 to k and the backoff stage reinitialized from i to 0. The probability of this event equals: *Pr{transmission is successful and number k was randomly chosen to initiate the backoff timer at stage 0}* = *Pr{transmission is successful}* · *Pr{number k was randomly chosen to initiate the backoff timer at stage 0}*. The probability of successful transmission is equal to $1-p_f$ and the probability that number k was randomly chosen to initiate the backoff timer at stage 0 equals $1/W_0$.

Ad (d): The station's backoff timer is changed from 0 to k and the backoff stage is increased from $i-1$ to i. Probability of this event equals: *Pr{transmission is unsuccessful and number k was randomly chosen to initiate the backoff timer at stage i}* = *Pr{transmission is unsuccessful}* · *Pr{number k was randomly chosen to initiate the backoff timer at stage i}*. The probability of unsuccessful transmission equals p_f and the probability that number k was randomly chosen to initiate the backoff timer at stage i equals $1/W_i$.

Ad (e): The station's backoff timer is changed from 0 to k and the backoff stage is changed from m to 0, i.e., the station has reached the maximum retransmission count. The probability of this event equals the probability that number k was randomly chosen to initiate the backoff timer at stage 0, i.e., $1/W_0$.

Let $b_{i,k}$ be the steady-state occupancy probability of state (i,k). It can be shown that:

$$b_{i,0} = p_f \cdot b_{i-1,0} \tag{13}$$

$$b_{i,0} = p_f^i \cdot b_{0,0} \tag{14}$$

and

$$b_{i,k} = \begin{cases} \frac{W_i-k}{W_i(1-p_{coll})} p_f^i \cdot b_{0,0}, & 0 < k \le W_i-1 \\ p_f^i \cdot b_{0,0}, & k = 0 \end{cases} \tag{15}$$

From the normalization condition:

$$\sum_{i-0}^{m} \sum_{k=0}^{W_i-1} b_{i,k} = 1 \tag{16}$$

and

$$\sum_{i=0}^{m} b_{i,0} = b_{0,0} \frac{1-p_f^{m+1}}{1-p_f} \tag{17}$$

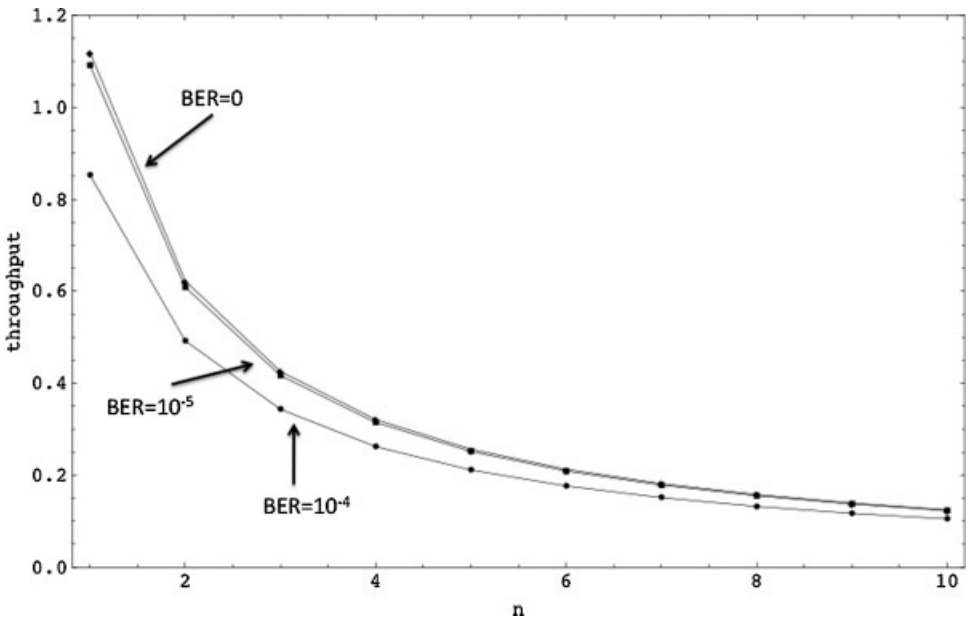


Figure 3. S_{DATA} as a function of n – for $L = 214$ octets and different values of BER.

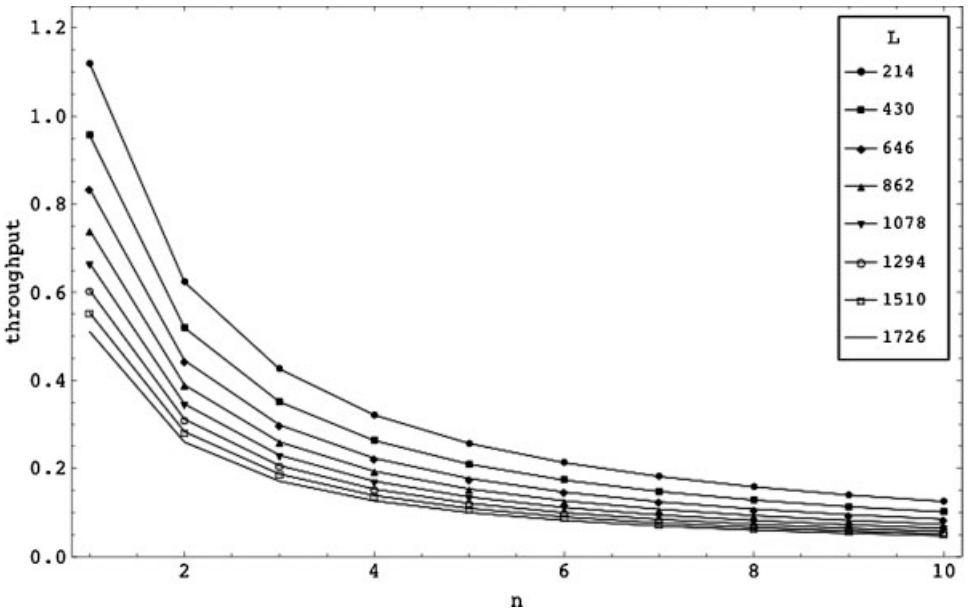


Figure 4. S_{DATA} as a function of n – for different values of frame length and $BER = 0$.

we get:

$$b_{0,0}^{-1} = \begin{cases} \frac{(1-p_{\mathrm{f}})W_0(1-(2p_{\mathrm{f}})^{m+1})-(1-2p_{\mathrm{f}})(1-p_{\mathrm{f}}^{m+1})}{2(1-2p_{\mathrm{f}})(1-p_{\mathrm{f}})(1-p_{\mathrm{coll}})} + \frac{1-p_{\mathrm{f}}^{m+1}}{1-p_{\mathrm{f}}}, & m \le m' \\ \frac{\psi}{2(1-2p_{\mathrm{f}})(1-p_{\mathrm{f}})(1-p_{\mathrm{coll}})} + \frac{1-p_{\mathrm{f}}^{m+1}}{1-p_{\mathrm{f}}}, & m > m' \end{cases} \tag{18}$$

where

$$\psi = (1-p_{\mathrm{f}})W_0(1-(2p_{\mathrm{f}})^{m'+1})-(1-2p_{\mathrm{f}})(1-p_{\mathrm{f}}^{m+1}) + W_0 2^{m'} p_{\mathrm{f}}^{m'+1}(1-2p_{\mathrm{f}})(1-p_{\mathrm{f}}^{m-m'}) \tag{19}$$

The probability of frame transmission τ is equal to *Pr{backoff timer equals* 0*}* and thus:

$$\tau = \sum_{i=0}^{m} b_{i,0} = \begin{cases} \left(\frac{(1-p_f)W_0(1-(2p_f)^{m+1})-(1-2p_f)(1-p_f^{m+1})}{2(1-2p_f)(1-p_f)(1-p_{coll})} + \frac{1-p_f^{m+1}}{1-p_f}\right)^{-1} \frac{1-p_f^{m+1}}{1-p_f}, & m \le m' \\ \left(\frac{\psi}{2(1-2p_f)(1-p_f)(1-p_{coll})} + \frac{1-p_f^{m+1}}{1-p_f}\right)^{-1} \frac{1-p_f^{m+1}}{1-p_f}, & m > m' \end{cases} \tag{20}$$

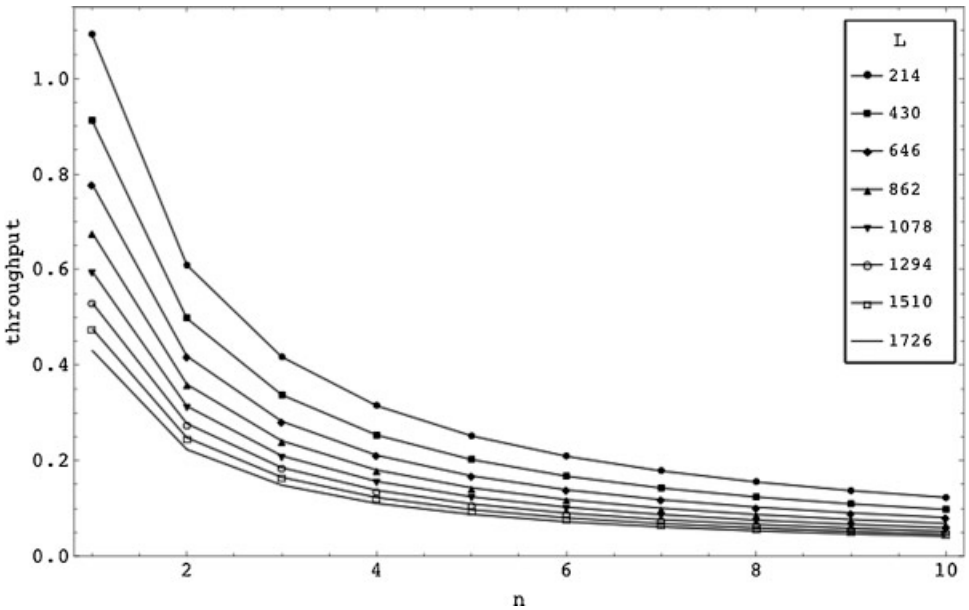


Figure 5. S_{DATA} as a function of n – for different values of frame length and $BER = 10^{-5}$.

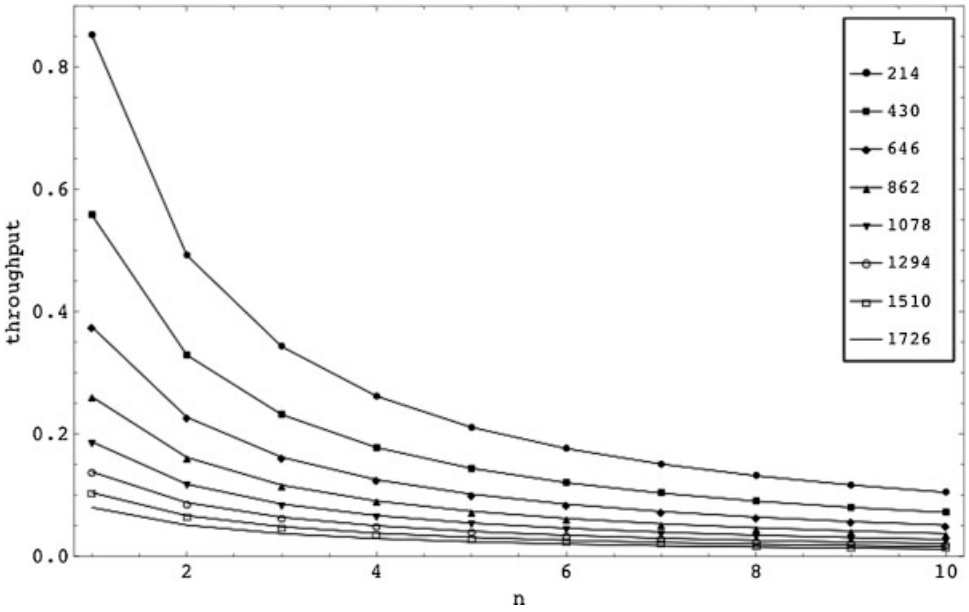


Figure 6. S_{DATA} as a function of n – for different values of frame length and $BER = 10^{-4}$.

For $p_{coll} = 0$ the above solution is the same as presented in Ref. [4].

4.4. Probability of transmission failure p_f and probability of collision p_{coll}

We use a channel model with random distribution of errors, i.e., without grouping of errors. The probability of transmission failure

$$p_f = 1-(1-p_{coll})(1-p_e) \tag{21}$$

where p_e is the frame error probability:

$$p_e = 1-(1-p_{e\_data})(1-p_{e\_ACK}) \tag{22}$$

where p_{e_data} is FER for data frames and p_{e_ACK} is FER for ACK frames. p_{e_data} and p_{e_ACK} can be calculated from bit error probability (i.e., BER), p_b:

$$p_{e\_data} = 1-(1-p_b)^{L_{data}} \tag{23}$$

$$p_{e\_ACK} = 1-(1-p_b)^{L_{ACK}} \tag{24}$$

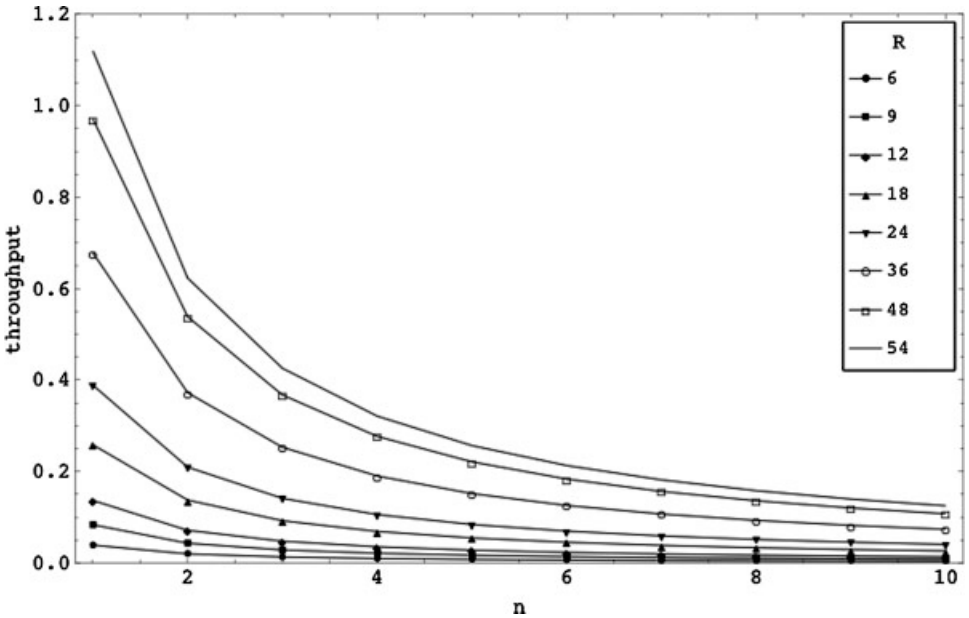


Figure 7. S_{DATA} as a function of n – for $L = 214$ octets, BER = 0 and different values of R.

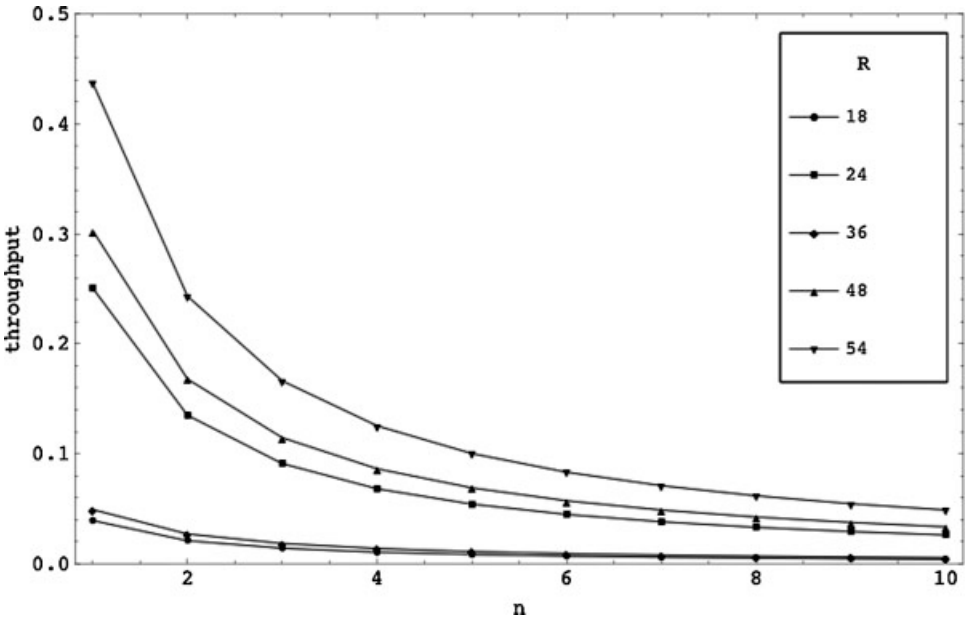


Figure 8. S_{ACK} as a function of n – for $L = 214$ octets, BER = 0 and different values of R.

The probability of collision:

$$p_{coll} = 1-(1-\tau)^{n-1} \tag{25}$$

Finally

$$p_f = 1-(1-p_{coll})(1-p_e) = 1-(1-\tau)^{n-1}(1-p_e) \tag{26}$$

Equations (20) and (26) form a nonlinear system with two unknown variables τ and p_f, which can be solved numerically.

4.5. Capacity and saturation throughput of steganographic channels

Let the capacity of a steganographic channel based on data frames be:

$$C_{DATA} = N_{BpS}\left|\frac{L_{SER} + L_{TAIL} + L_{DATA}}{N_{BpS}}\right| - (L_{SER} + L_{TAIL} + L_{DATA}) \tag{27}$$

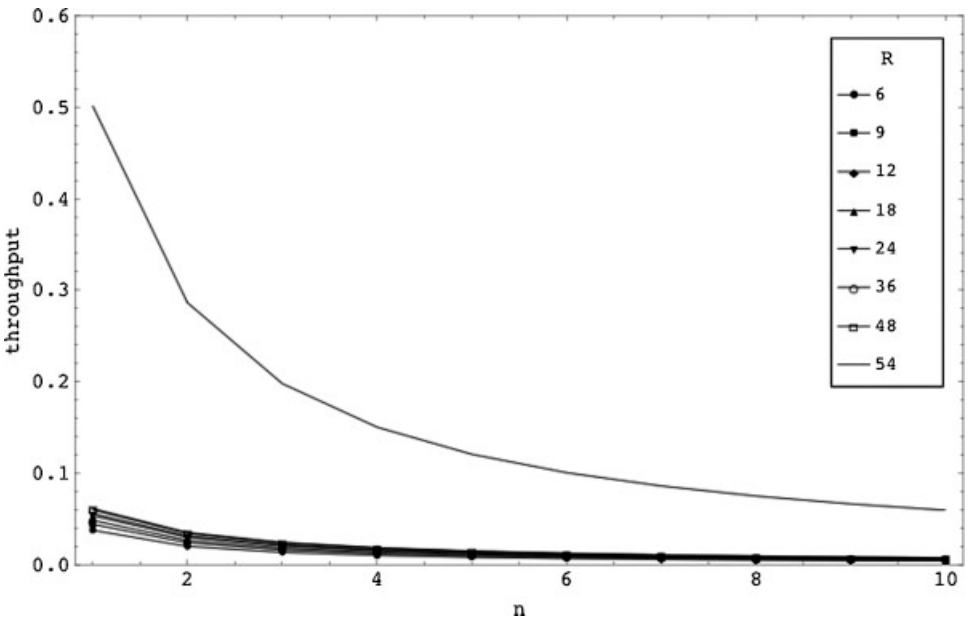


Figure 9. S_{DATA} as a function of n – for $L = 68$ octets, BER = 0 and different values of R.

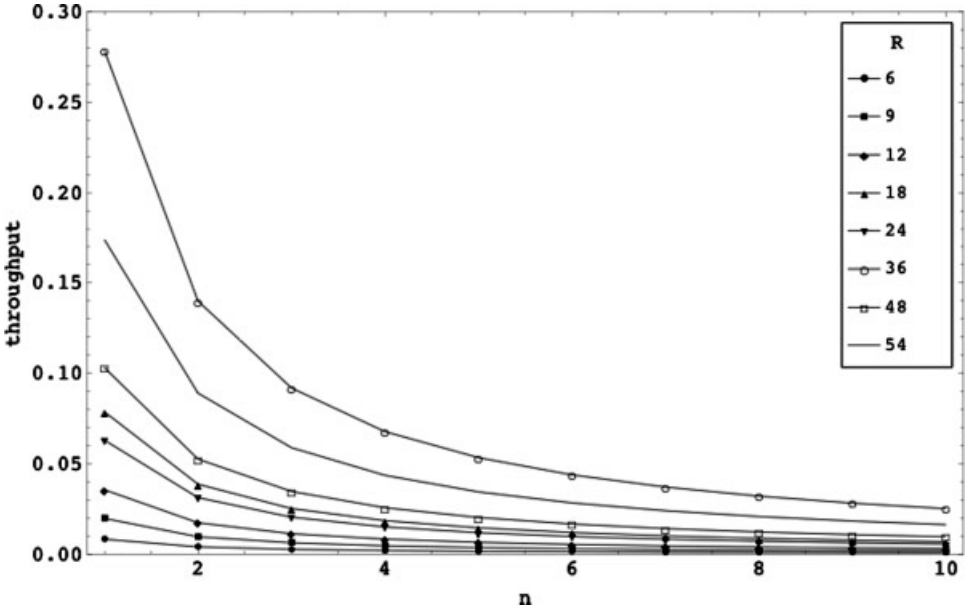


Figure 10. S_{DATA} as a function of n – for $L = 1528$ octets, BER = 0 and different values of R.

Let the capacity of a steganographic channel based on ACK frames be:

$$C_{\mathrm{ACK}} = N_{\mathrm{BpS}} \left| \frac{L_{\mathrm{SER}} + L_{\mathrm{TAIL}} + L_{\mathrm{ACK}}}{N_{\mathrm{BpS}}} \right| - (L_{\mathrm{SER}} + L_{\mathrm{TAIL}} + L_{\mathrm{ACK}}) \tag{28}$$

Therefore the saturation throughput of a steganographic channel based on data frames may be defined as:

$$S_{\mathrm{DATA}} = \frac{C_{\mathrm{DATA}} \cdot S}{n \cdot L_{\mathrm{pld}}} \tag{29}$$

And, finally, the saturation throughput of a steganographic channel based on ACK frames equals:

$$S_{\mathrm{ACK}} = \frac{C_{\mathrm{ACK}} \cdot S}{n \cdot L_{\mathrm{pld}}} \tag{30}$$

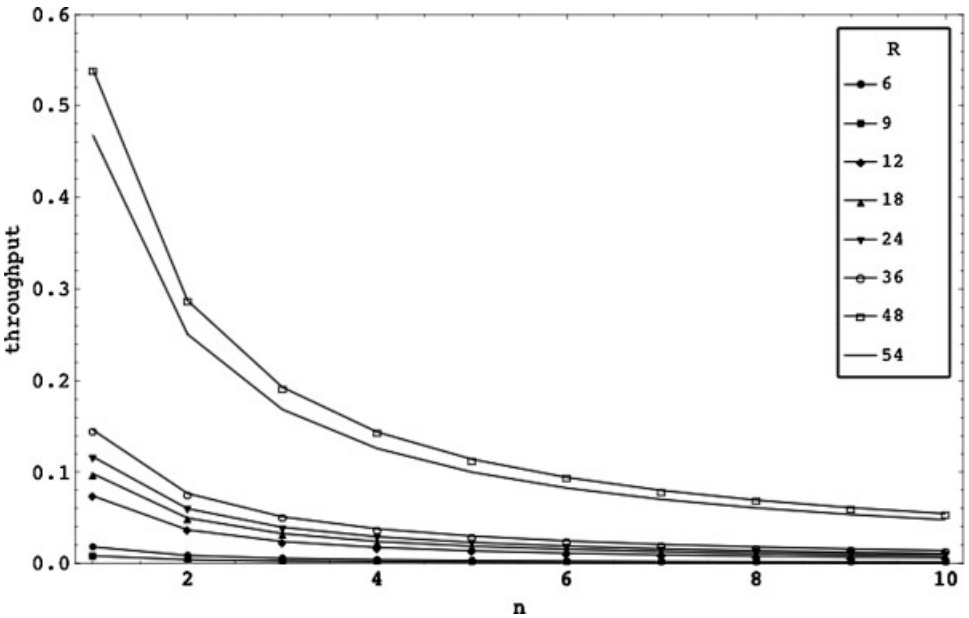


Figure 11. S_{DATA} as a function of n – for $L = 604$ octets, BER = 0 and different values of R.

Security Comm. Networks (2011) © 2011 John Wiley & Sons, Ltd.
DOI: 10.1002/sec

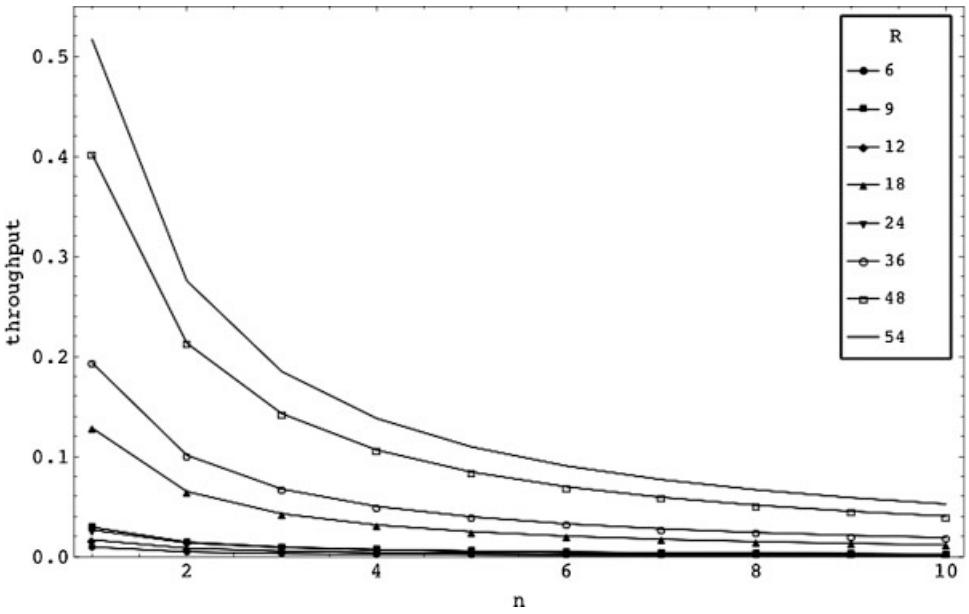


Figure 12. S_{DATA} as a function of n – for $L = 656$ octets, BER = 0 and different values of R.

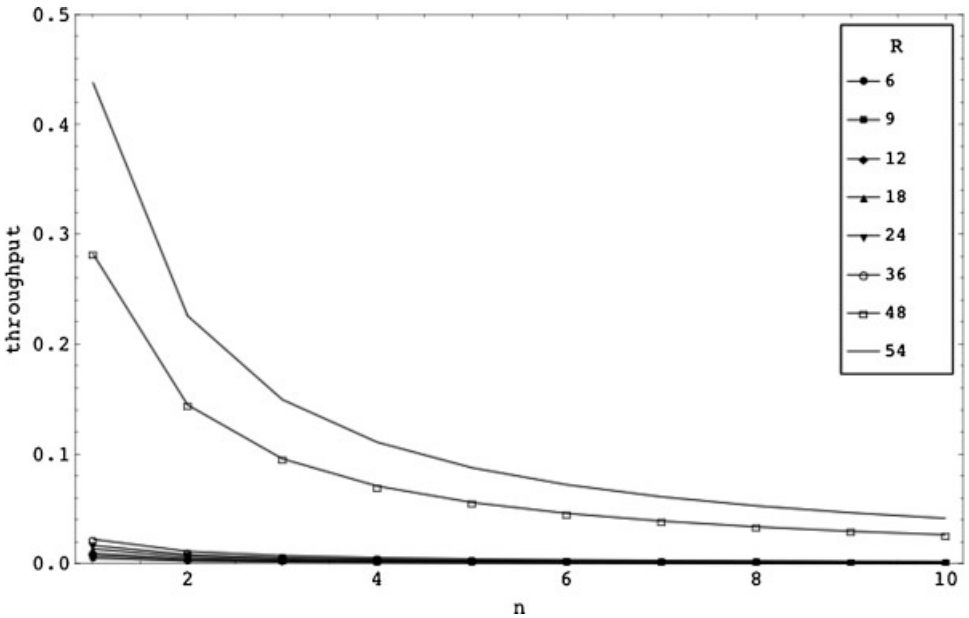


Figure 13. S_{DATA} as a function of n – for $L = 1328$ octets, BER = 0 and different values of R.

5. ANALYSIS

5.1. Frames with a maximum number of padding octets

All diagrams presented in this section display values of the saturation throughput of the proposed steganographic method (WiPad) based on the data frame variant. All calculations were made for $n \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$. For $L = 214$ octet frames ($\alpha = 1$; 186 bytes at IP layer) the following values of BER were used $\{10^{-4}, 10^{-5}, 0\}$, and for $L \in \{214, 430, 646, 862, 1078, 1294, 1510\}$ octet frames ($\alpha \in \{1, 2, \ldots, 7\}$) the correspondent BER$\in \{10^{-4}, 10^{-5}, 0\}$. We considered the IEEE 802.11g – ERP-OFDM i.e., 'g'-only mode and a data rate of $R = 54$ Mbps (with an exception for the last diagram, which provides an evaluation of the impact of R on S_{DATA}).

Figure 3 presents S_{DATA} as a function of n for $L = 214$ octet frames and different values of BER. Along with an

increasing value of BER the steganographic throughput, S_{DATA}, declines. The maximum value reaches 1.12 Mbps for BER = 0 and $n = 1$. Along with an increasing value of BER the presented curves flatten out. For a given BER, the decrease of S_{DATA} together with an increase of n is related to a growing number of collisions in the medium. The observed decline in the value of S_{DATA} between BER = 0 and BER = 10^{-5} is very small.

Figure 4 presents S_{DATA} as a function of n for different values of frame length and BER = 0. For a given n, an increasing frame length leads to a fall in the attainable S_{DATA}.

Figure 5 represents the correlation between S_{DATA} and n, for different values of frame length and BER = 10^{-5}, while Figure 6 displays S_{DATA} as a function of n for different frame lengths and BER = 10^{-4}. Compared to the values obtained for BER = 0, we observe a reduction in the value of S_{DATA} due to the influence of channel errors.

Finally we evaluate (Figure 7) S_{DATA} as a function of n for different IEEE 802.11g data rates $R \in \{6, 9, 12, 18, 24, 36, 48, 54\}$ Mbps.

5.2. ACK frames

We evaluate (Figure 8) S_{ACK} as a function of n for different IEEE 802.11g data rates $R \in \{18, 24, 36, 48, 54\}$ Mbps. For $n = 1$ and $R = 54$, $S_{ACK} = 0.44$ Mbps (82 bits serve as a hidden channel). The throughput for 24 Mbps networks is higher than for 36 Mbps because of the different capacity of the hidden channel: 58 and 10 bits, respectively.

5.3. Typical IP packet sizes

The Refs. [15,16] show that most typical sizes for IP packets are 40 and 1500 bytes, and then 576, 628, and 1300 bytes. These values are in line with $L \in \{68, 1528, 604, 656, 1328\}$ octet frames. We consider $R \in \{6, 9, 12, 18, 24, 36, 48, 54\}$ and BER = 0.

For $L = 68$ octets (Figure 9), $n = 1$ for and $R = 54$ S_{DATA} is 0.50 Mbps (capacity of the hidden channel: 82 bits). For $R \in \{6, 9, 12, 18, 24, 36, 48\}$ the capacity of the hidden channel is only 10 bits and for $n = 1$ S_{DATA} is low (<0.06 Mbps).

For $L = 1528$ octets (Figure 10), $n = 1$ for and $R = 36$ S_{DATA} is 0.28 Mbps (capacity of the hidden channel: 138 bits) and for $R = 54$ S_{DATA} is 0.17 Mbps (66 bits). For other values of R S_{DATA} is from 0.01 to 0.1.

For $L = 604$ octets (Figure 11), $n = 1$ for and $R = 48$ S_{DATA} is 0.54 Mbps (capacity of the hidden channel: 138 bits), and for $R = 54$ S_{DATA} is 0.47 Mbps (114 bits). For other values of R S_{DATA} is from 0.01 to 0.15.

For $L = 656$ octets (Figure 12), $n = 1$ for and $R = 54$ S_{DATA} is 0.52 Mbps (capacity of the hidden channel: 130 bits). For $R = 48$ S_{DATA} is 0.40 Mbps (106 bits), for $R = 36$ S_{DATA} is 0.19 Mbps (58 bits), and $R = 18$ S_{DATA} is 0.13 Mbps (58 bits). For other values of R S_{DATA} is from 0.01 to 0.03.

Finally, for $L = 1328$ octets (Figure 13), $n = 1$ for and $R = 54$ S_{DATA} is 0.48 Mbps (capacity of the hidden channel: 154 bits) and for $R = 48$ S_{DATA} is 0.28 Mbps (106 bits). For other values of R S_{DATA} is from 0.01 to 0.2.

For evaluated lengths of IP packets the highest throughput is generally for 54 and 48 Mbps IEEE 802.11 networks. For 40, 576, 628, and 1300 bytes packets the maximal value of S_{DATA} is around 0.50 Mbps. For 1500 bytes IP packet S_{DATA} is below 0.3 Mbps.

6. CONCLUSIONS AND FUTURE WORK

In this paper we evaluated a new steganographic method called WiPad intended for IEEE 802.11 OFDM networks, whose functioning bases on insertion of bits into the padding of transmission symbols. The analysis for IEEE 802.11g 54 Mbps networks revealed that the capacity of WiPad equals 1.1 Mbit/s for data frames and 0.44 Mbit/s for ACK frames, which gives a total of almost 1.54 Mbit/s. To the authors' best knowledge this is the most capacious of all the known steganographic network channels.

Future work will include WiPad the estimation of achievable steganographic bandwidth in case of the IEEE 802.11n standard also with channel model with grouping of errors. Further studies should also involve pinpointing potential detection mechanisms of the proposed communication system. Experimental implementation as a proof-of-concept will be made similar to Ref. [17] in MATLAB and Simulink with Communication Toolbox.

REFERENCES

1. BBC News. *FBI allegations against 'Russian spies' in US*. http://www.bbc.co.uk/news/10442869 [29 June 2010].
2. Szczypiorski K, Lubacz J. *Performance Evaluation of IEEE 802.11 DCF Networks*. In *20th International Teleraffic Congress (ITC-20), Ottawa, Canada*, June 2007; Lecture Notes in Computer Science (LNCS) 4516, Springer-Verlag Berlin Heidelberg, 2007; 1082–1093.
3. Bianchi G. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications* **18**(3): 2000; 535–547.
4. Ni Q, Li T, Turletti T, Xiao Y. Saturation throughput analysis of error-prone 802.11 wireless networks. *Wiley Journal of Wireless Communications and Mobile Computing (JWCMC)* **5**(8): 2005; 945–956.
5. Wolf M. *Covert Channels in LAN Protocols*. In *Proceedings of the Workshop of Local Area Network Security (LANSEC)*, 1989; 91–101.
6. Fisk G, Fisk M, Papadopoulos C, Neil J. Eliminating Steganography in Internet Traffic with Active Wardens. In *Proceedings of the 5th International Workshop on Information Hiding, Lecture Notes in Computer Science: 2578*, October 7–9, 2002, Noordwijkerhout, The Netherlands, Springer-Verlag: Heidelberg, Germany, 2003; 18–35.

7. Jankowski B, Mazurczyk W, Szczypiorski K. *Information hiding using improper frame padding*. In *Proceedings of the 14th International Telecommunications Network Strategy and Planning Symposium – Networks 2010*, September 2010, Warsaw, Poland.
8. Lucena NB, Lewandowski G, Chapin SJ. *Covert channels in IPv6*. In *Proceedings of the Privacy Enhancing Technologies (PET)*, May 2005; 147–166.
9. Szczypiorski K. *HICCUPS: hidden communication system for coruppted networks*. In *Proceedings of the Tenth International Multi-Conference on Advanced Computer Systems ACS'2003. Miedzyzdroje*, October 2003; 31–40.
10. Szczypiorski K. *A performance analysis of HICCUPS – a steganographic system for WLAN*. In *Proceedings of the 2009 International Conference on Multimedia Information NEtworking and Security (MINES 2009) – First International Workshop on Network Steganography (IWNS'09)*, Vol. I, Wuhan, Hubei, China, November 2009; 569–572.
11. Krätzer C, Dittmann J, Lang A, Kuhne T. *WLAN steganography: a first practical review*. In *Proceedings of the 8th ACM Multimedia and Security Workshop*. Geneve (Switzerland), September 2006.
12. Krätzer C, Dittmann J, Merkel R. *WLAN steganography revisited*. In *Proceedings of the SPIE Electronic Imaging 2008*, San Jose, CA, 2008.
13. IEEE 802.11, 2007 Edition, IEEE Standard for Information Technology – Telecommunications and Information Exchange between Systems – Local and Metropolitan Area Network – Specific Requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications (2007).
14. Kochut A, Vasan A, Shankar A, Agrawala A. *Sniffing out the correct physical layer capture model in 802.11b*. In *Proceedings of the 12th IEEE International Conference on Network Protocols (ICNP 2004)*, Berlin 2004.
15. John W, Tafvelin S. *Analysis of Internet backbone traffic and header anomalies observed*. In *Proceedings of the Internet Measurement Conference IMC'07*, San Diego, CA, August 2007.
16. Sinha R, Papadopoulos C, Heidemann J. Internet Packet Size Distributions: Some Observations. University of Southern California: Los Angeles, CA, USA, (web page released October 5 2005 republished as ISI-TR-2007-643 May 2007).
17. Odor M, Babak N, Salmanian M, Mason P, Martin M, Liscano R. *A frame handler module for a side-channel in mobile ad hoc networks*. In *Proceedings of the 5th LCN Workshop on Security in Communications Networks (SICK 2009)*, Zürich 2009; 930–936.

Steganography of VoIP Streams

Wojciech Mazurczyk and Krzysztof Szczypiorski

Warsaw University of Technology,
Faculty of Electronics and Information Technology,
Institute of Telecommunications,
15/19 Nowowiejska Str., 00-665 Warsaw, Poland
{W.Mazurczyk, K.Szczypiorski}@tele.pw.edu.pl

Abstract. The paper concerns available steganographic techniques that can be used for creating covert channels for VoIP (Voice over Internet Protocol) streams. Apart from characterizing existing steganographic methods we provide new insights by presenting two new techniques. The first one is network steganography solution which exploits free/unused protocols' fields and is known for IP, UDP or TCP protocols but has never been applied to RTP (Real-Time Transport Protocol) and RTCP (Real-Time Control Protocol) which are characteristic for VoIP. The second method, called LACK (Lost Audio Packets Steganography), provides hybrid storage-timing covert channel by utilizing delayed audio packets. The results of the experiment, that was performed to estimate a total amount of data that can be covertly transferred during typical VoIP conversation phase, regardless of steganalysis, are also included in this paper.

Keywords: VoIP, information hiding, steganography.

1 Introduction

VoIP is one of the most popular services in IP networks and it stormed into the telecom market and changed it entirely. As it is used worldwide more and more willingly, the traffic volume that it generates is still increasing. That is why VoIP is suitable to enable hidden communication throughout IP networks. Applications of the VoIP covert channels differ as they can pose a threat to the network communication or may be used to improve the functioning of VoIP (e.g. security like in [12] or quality of service like in [13]). The first application of the covert channel is more dangerous as it may lead to the confidential information leakage. It is hard to assess what bandwidth of covert channel poses a serious threat – it depends on the security policy that is implemented in the network. For example, US Department of Defense specifies in [24] that any covert channel with bandwidth higher than 100 bps must be considered insecure for average security requirements. Moreover for high security requirements it should not exceed 1 bps.

In this paper we present available covert channels that may be applied for VoIP during conversation phase. A detailed review of steganographic methods that may be applied during signalling phase of the call can be found in [14]. Here, we introduce two new steganographic methods that, to our best knowledge, were not described earlier.

R. Meersman and Z. Tari (Eds.): OTM 2008, Part II, LNCS 5332, pp. 1001–1018, 2008.

Next, for each of these methods we estimate potential bandwidth to evaluate experimentally how much information may be transferred in the typical IP telephony call.

The paper is organized as follows. In Section 2 we circumscribe the VoIP traffic and the communication flow. In Section 3, we describe available steganographic methods that can be utilized to create covert channels in VoIP streams. Then in Section 4 we present results of the experiment that was performed. Finally, Section 5 concludes our work.

2 VoIP Communication Flow

VoIP is a real-time service that enables voice conversations through IP networks. It is possible to offer IP telephony due to four main groups of protocols:

a. *Signalling protocols* that allow to create, modify, and terminate connections between the calling parties – currently the most popular are SIP [18], H.323 [8], and H.248/Megaco [4],
b. *Transport protocols* – the most important is RTP [19], which provides end-to-end network transport functions suitable for applications transmitting real-time audio. RTP is usually used in conjunction with UDP (or rarely TCP) for transport of digital voice stream,
c. *Speech codecs* e.g. G.711, G.729, G.723.1 that allow to compress/decompress digitalized human voice and prepare it for transmitting in IP networks.
d. Other *supplementary protocols* like RTCP [19], SDP, or RSVP etc. that complete VoIP functionality. For purposes of this paper we explain the role of RTCP protocol: RTCP is a control protocol for RTP and it is designed to monitor the Quality of Service parameters and to convey information about the participants in an ongoing session.

Generally, IP telephony connection consists of two phases: a *signalling phase* and a *conversation phase*. In both phases certain types of traffic are exchanged between calling parties. In this paper we present a scenario with SIP as a signalling protocol

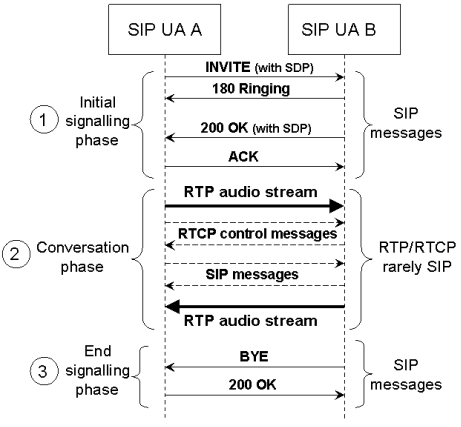


Fig. 1. VoIP call setup based on SIP/SDP/RTP/RTCP protocols (based on [9])

and RTP (with RTCP as control protocol) for audio stream transport. That means that during the signalling phase of the call certain SIP messages are exchanged between SIP endpoints (called: SIP User Agents). SIP messages usually traverse through SIP network servers: proxies or redirects that allow end-users to locate and reach each other. After this phase, the conversation phase begins, where audio (RTP) streams flow bi-directly between a caller and a callee. VoIP traffic flow described above and distinguished phases of the call are presented in Fig. 1. For more clarity we omitted the SIP network server in this diagram. Also potential security mechanisms in traffic exchanges were ignored.

3 Covert Channels in VoIP Streams Overview and New Insights

Besides characterizing IP telephony traffic flow Fig. 1 also illustrates steganographic model used in this paper for VoIP steganography evaluation. The proposed model is as follows. Two users *A* and *B* are performing VoIP conversation while simultaneously utilizing it to send steganograms by means of all possible steganographic methods that can be applied to IP telephony protocols. We assume that both users control their end-points (transmitting and receiving equipment) thus they are able to modify and inspect the packets that are generated and received. After modifications at calling endpoint, packets are transmitted through communication channel which may introduce negative effects e.g. delays, packet losses or jitter. Moreover, while traveling through network packets can be inspected and modified by an active warden [5]. Active wardens act like a semantic and syntax proxy between communication sides. They are able to modify and normalize exchanged traffic in such a way that it does not break, disrupt or limit any legal network communication or its functionality. Thus, active wardens can inspect all the packets sent and modify them slightly during the VoIP call. It must be emphasized however that they may not erase or alter data that can be potentially useful for VoIP non-steganographic (overt) users. This assumption forms important active wardens' rule although sometimes elimination of the covert channel due to this rule may be difficult.

To later, in section 4, practically evaluate covert channels that can be used for VoIP transmission we must first define three important measures that characterizes them and which must be taken into consideration during VoIP streams covert channels analysis. These measures are:

- *Bandwidth* that may be characterized with *RBR* (Raw Bit Rate) that describes how many bits may be sent during one time unit [bps] with the use of all steganographic techniques applied to VoIP stream (with no overheads included) or *PRBR* (Packet Raw Bit Rate) that circumscribe how much information may be covertly sent in one packet [bits/packet],
- *Total amount of covert data* [bits] transferred during the call that may be sent in one direction with the use of all applied covert channels methods for typical VoIP call. It means that, regardless of steganalysis, we want to know how much covert information can be sent during typical VoIP call,
- *Covert data flow distribution during the call* – how much data has been transferred in a certain moment of the call.

We will be referencing to abovementioned measures during the following sections while presenting available steganographic methods for VoIP communication and later during the experiment description and results characterization.

In this section we will provide an overview of existing steganographic techniques used for creation of covert channels in VoIP streams and present new solutions. As described earlier during the conversation phase audio (RTP) streams are exchanged in both directions and additionally, RTCP messages may be sent. That is why the available steganographic techniques for this phase of the call include:

- *IP/UDP/TCP/RTP* protocols steganography in network and transport layer of TCP/IP stack,
- *RTCP* protocol steganography in application layer of TCP/IP stack,
- *Audio watermarking* (e.g. LSB, QIM, DSSS, FHSS, Echo hiding) in application layer of TCP/IP stack,
- *Codec SID frames* steganography in application layer of TCP/IP stack,
- *Intentionally delayed audio packets* steganography in application layer of TCP/IP stack,
- *Medium dependent* steganographic techniques like HICCUPS [22] for VoWLAN (Voice over Wireless LAN) specific environment in data link layer of TCP/IP stack.

Our contribution in the field of VoIP steganography includes the following:

- Describing RTP/RTCP protocols' fields that can be potentially utilized for hidden communication,
- Proposing *security mechanisms fields steganography* for RTP/RTCP protocols,
- Proposing intentionally delayed audio packets steganographic method called LACK (Lost Audio Packets Steganographic Method).

3.1 IP/TCP/UDP Protocols Steganography

TCP/UDP/IP protocols steganography utilizes the fact that only few fields of headers in the packet are changed during the communication process ([15], [1], [17]). Covert data is usually inserted into redundant fields (provided, but often unneeded) for abovementioned protocols and then transferred to the receiving side. In TCP/IP stack, there is a number of methods available, whereby covert channels can be established and data can be exchanged between communication parties secretly. An analysis of the headers of TCP/IP protocols e.g. IP, UDP, TCP results in fields that are either unused or optional [15], [25]. This reveals many possibilities where data may be hidden and transmitted. As described in [15] the IP header possesses fields that are available to be used as covert channels. Notice, that this steganographic method plays an important role for VoIP communication because protocols mentioned above are present in every packet (regardless, if it is a signalling message, audio packet, or control message). For this type of steganographic method as well as for other protocols in this paper (RTP and RTCP steganography) achieved steganographic bandwidth can be expressed as follows:

$$PRBR_{NS} = \frac{\left(SB_0 + \sum_{j=1}^{l} SB_j \right)}{l+1} \ [bits / packet] \tag{1}$$

where:

$PRBR_{NS}$ (Packet Raw Bit Rate) denotes bandwidth of the covert channel created by IP/TCP/UDP steganography [bits/packet],
SB_0 is total amount of bits for IP/TCP/UDP protocols that can be covertly send in the fields of the first packet. This value differs from the value achieved for the following packets because in the first packet initial values of certain fields can be used (e.g. sequence number for TCP protocol),
SB_j denotes total amount of bits for IP/TCP/UDP protocols that can be covertly sent in the fields of the following packets,
l is number of packets send besides first packet.

3.2 RTP Protocols Steganography

3.2.1 RTP Free/Unused Fields Steganography

In conversation phase of the call when the voice stream is transmitted, besides protocols presented in section 3.1 also the fields of RTP protocol may be used as a covert channel. Fig. 2 presents the RTP header.

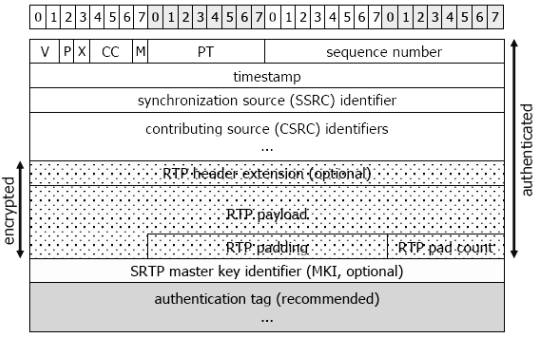


Fig. 2. RTP header with marked sections that are encrypted and authenticated

RTP provides the following opportunities for covert communication:

- *Padding* field may be needed by some encryption algorithms. If the padding bit (P) is set, the packet contains one or more additional padding octets at the end of header which are not a part of the payload. The number of the data that can be added after the header is defined in the last octet of the padding as it contains a count of how many padding octets should be ignored, including itself,
- *Extension header* (when X bit is set) – similar situation as with the padding mechanism, a variable-length header extension may be used,
- Initial values of the *Sequence Number* and *Timestamp* fields – because both initial values of these fields must be random, the first RTP packet of the audio stream may be utilized for covert communication,
- Least significant bits of the *Timestamp* field can be utilized in a similar way as proposed in [6].

It must be emphasized however that steganography based on free/unused/optional fields for RTP protocol (as well as for protocols mentioned in section 3.1) may be potentially eliminated or greatly limited by active wardens. Normalization of RTP headers' fields values (e.g. applied to *Timestamps*) or small modifications applied may be enough to limit covert bandwidth. On the other hand it is worth noting that so far no documented active warden implementation exists.

3.2.2 RTP Security Mechanisms Steganography

There is also another way to create high-bandwidth covert channel for RTP protocol. In Fig. 5 one can see what parts of RTP packet is secured by using encryption (payload and optionally header extension if used) and authentication (*authentication tag*). For steganographic purposes we may utilize security mechanisms' fields. The main idea is to use *authentication tag* to transfer data in a covert manner. In SRTP (Secure RTP) standard [2] it is recommended that this field should be 80 bits long but lower values are also acceptable (e.g. 32 bits). Similar steganographic method that utilizes security mechanism fields was proposed for e.g. IPv6 in [11]. By altering content of fields like *authentication tag* with steganographic data it is possible to create covert channel because data in these fields is almost random due to the cryptographic mechanism operations. That is why it is hard to detect whether they carry real security data or hidden information. Only receiving calling party, as he is in possession of pre-shared key (*auth_key*) is able to determine that. For overt users wrong authentication data in packet will mean dropping it. But because receiving user is controlling its VoIP equipment, when *authentication tag* fields are utilized as covert channel, he is still able to extract steganograms in spite of invalid authentication result.

Thus, most of steganalysis methods will fail to uncover this type of secret communication. The only solution is to strip off/erase such fields from the packets but this is a serious limitation for providing security services for overt users. Moreover it will be violation of the active warden rule (that no protocol's semantic or syntax will be disrupted).

Because the number of RTP packets per one second is rather high (depends on the voice frame generation interval) exploiting this tag provides a covert channel that bandwidth can be expressed as follows:

$$RBR_{SRTP} = SB_{AT} \cdot \frac{1000}{I_p} \quad [bits/s] \tag{2}$$

where:

RBR_{SRTP} (Raw Bit Rate) denotes bandwidth of the covert channel created by RTP security mechanism steganography (in bits/s),
SB_{AT} is total amount of bits in *authentication tag* for SRTP protocol (typically 80 or 32 bits),
I_p describes voice packet generation interval, in miliseconds (typically from 10 to 60 ms).

For example, consider a scenario in which *authentication tag* is 32 bits long and audio packet is generated each 20 ms. Based on equation 2 we can calculate that

RBR_{SRTP} = 1.6 kbit/s which is considerably high result for bandwidths of covert channel presented in this paper.

3.3 RTCP Protocol Steganography

3.3.1 RCTP Free/Unused Fields Steganography

To our best knowledge this is the first proposal to use RTCP protocol messages as a covert channel. RTCP exchange is based on the periodic transmission of control packets to all participants in the session. Generally it operates on two types of packets (reports) called: Receiver Report (RR) and Sender Report (SR). Certain parameters that are enclosed inside those reports may be used to estimate network status. Moreover all RTCP messages must be sent in compound packet that consists of at least two individual types of RTCP reports. Fig. 3 presents headers of SR and RR reports of the RTCP protocol.

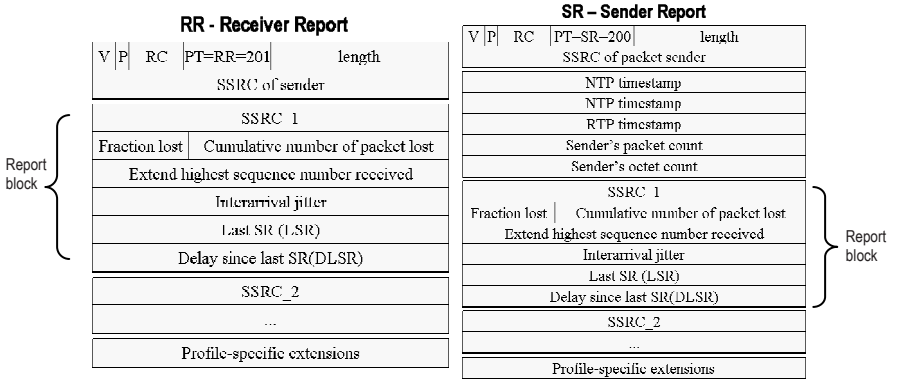


Fig. 3. RTCP Receiver Report (RR) and Sender Report (SR)

For sessions with small number of the participants the interval between the RTCP messages is 5 seconds and moreover sending RTCP communication (with overhead) should not exceed 5% of the session's available bandwidth. For creating covert channels report blocks in SR and RR reports (marked in Fig. 6) may be utilized. Values of the parameters transferred inside those reports (besides SSRC_1 which is the source ID) may be altered, so the amount of information that may be transferred in each packet is 160 bits. It is clear, that if we use this type of steganographic technique, we lose some (or all) of RTCP functionality (it is a cost to use this solution). Other free/unused fields in these reports may be also used in the similar way. For example *NTP Timestamp* may be utilized in a similar way as proposed in [6].

Other RTCP packet types include: SDES, APP or BYE. They can also be used in the same way as SR and RR reports. So the total PRBR for this steganographic technique is as follows:

$$PRBR_{RTCP} = S_{CP} \cdot N_{RB} \cdot S_{RB} \ [bits/packet] \quad (3)$$

where:

$PRBR_{RTCP}$ (Packet Raw Bit Rate) denotes bandwidth of the covert channel created with RCTP Free/Unused Fields Steganography (in bits/packet),
S_{CP} denotes size of the compound RTCP packet (the number of RTCP packet types inside the compound one),
N_{RB} is number of report blocks inside each RTCP packet type,
S_{RB} is the number of bits that can be covertly send in one RTCP report block.

It is also worth noting that RTCP messages are based on IP/UDP protocols, so additionally, for one RTCP packet, both protocols can be used for covert transmission.

To improve capacity of this covert channel one may send RTCP packets more frequently then each 5 seconds (which is default value proposed in standard) although it will be easier to uncover. Steganalysis of this method is not so straightforward as in case of security mechanism fields steganography. Active warden can be used to eliminate or greatly limit the fields in which hidden communication can take place although it will be serious limitation of RTCP functionality for overt users.

3.3.2 RTCP Security Mechanisms Steganography

Analogously as for RTP protocol the same steganographic method that uses SRTP security mechanism may be utilized for RTCP and the achieved RBR_{RTCP} rate is as follows:

$$RBR_{SRTCP} = \frac{SB_{AT} \cdot l}{T} \quad [bits/s] \tag{4}$$

where:

RBR_{SRTCP} (Raw Bit Rate) denotes bandwidth of the covert channel created with SRTP security mechanism steganography [in bps],
SB_{AT} is total amount of bits in *authentication tag* for SRTP protocol,
T denotes duration of the call (in seconds),
l is number of RTCP messages exchanged during the call of length T.

3.4 Audio Watermarking

The primary application of audio watermarking was to preserve copyrights and/or intellectual properties called DRM (Digital Right Management). However, this technique can be also used to create effective covert channel inside a digital content. Currently there is a number of audio watermarking algorithms available. The most popular methods that can be utilized in real-time communication for VoIP service, include: *LSB* (Least Significant Bit), *QIM* (Quantization Index Modulation), *Echo Hiding*, *DSSS* (Direct Sequence Spread Spectrum), and *FHSS* (Frequency Hopping Spread Spectrum) [3]. For these algorithms the bandwidth of available covert channels depends mainly on the sampling rate and the type of audio material being encoded. Moreover, if covert data rate is too high it may cause voice quality deterioration and increased risk of detection. In Table 1 examples of digital watermarking data rates are presented under conditions that they do not excessively affect quality of the conversation and limit probability of disclosure. Based on those results one can clearly see that, besides *LSB* watermarking, other audio watermarking algorithms covert channels' bandwidth range from few to tens bits per second.

Table 1. Audio watermarking algorithms and their experimentally calculated RBRs

Audio watermarking algorithm	**Covert bandwidth RBR** (based on [21])	**Covert bandwidth RBR** (based on [1])
LSB	1 kbps / 1 kHz (of sampling rate)	4 kbps
DSSS	4 bps	22.5 bps
FHSS	-	20.2 bps
Echo Hiding	16 bps	22.3 bps

Thus, we must consider that each audio watermarking algorithm affects perceived quality of the call. That means that there is a necessary tradeoff between the amount of data to be embedded and the degradation in users' conversation. On the other hand by using audio watermarking techniques we gain an effective steganographic method: because of the audio stream flow the achieved bandwidth of the covert channel is constant. Thus, although the bit rate of audio watermarking algorithms is usually not very high, it still may play important role for VoIP streams covert channels.

Steganalysis of audio watermarking methods (besides for LSB algorithm which is easy to eliminate) is rather difficult and must be adjusted to watermarking algorithm used. It must be emphasized however that if hidden data embedding rate is chosen reasonably then detecting of the audio watermarking is hard but possible and in most cases erasing steganogram means great deterioration of voice quality.

3.5 Speech Codec Silence Insertion Description (SID) Frames Steganography

Speech codecs may have built-in or implement mechanisms like Discontinuous Transmission (DTX)/VAD (Voice Activity Detection)/CNG (Comfort Noise Generation) for network resources (e.g. bandwidth) savings. Such mechanisms are able to determine if voice is present in the input signal. If it is present, voice would be coded with the speech codec in other case, only a special frame called Silence Insertion Description (SID) is sent. If there is a silence, in stead of sending large voice packets that do not contain conversation data only small amount of bits are transmitted. Moreover, during silence periods, SID frames may not be transferred periodically, but only when the background noise level changes. The size of this frame depends on the speech codec used e.g. for G.729AB it is 10 bits per frame while for G.723.1 it is 24 bits per frame. Thus, when DTX/VAD/CNG is utilized, during the silence periods SID frames can be used to covertly transfer data by altering information of background noise with steganogram. In this case no new packets are generated and the covert bandwidth depends on the speech codec used. It is also possible to provide higher bandwidth of the covert channel by influencing rate at which SID frames are issued. In general, the more of these frames are sent the higher the bandwidth of the covert channel. It must be however noted that the covert bandwidth for this steganographic is rather low. What is important, for this steganographic method steganalysis is simple to perform. Active warden that is able to modify some of the bits in SID frames (e.g. least significant) can eliminate or greatly reduce the bandwidth of this method.

3.6 LACK: Intentionally Delayed Audio Packets Steganography

To our best knowledge this is the first proposal of using intentionally delayed (and in consequence lost) packets payloads as a covert channel for VoIP service. Although

there was an attempt how to use channel erasures at the sender side for covert communication [20] but this solution characterizes low bandwidth especially if we use it for VoIP connection (where the packet loss value must be limited). It is natural for IP networks that some packets can be lost due to e.g. congestion. For IP telephony, we consider a packet lost when:

- It does not reach the destination point,
- It is delayed excessive amount of time (so it is no longer valid), and that is why it may not be used for current voice reconstruction in the receiver at the arrival time.

Thus, for VoIP service when highly delayed packet reaches the receiver it is recognized as lost and then discarded. We can use this feature to create new steganographic technique. We called this method LACK (Lost Audio Packets Steganographic Method). In general, the method is intended for a broad class of multimedia, real-time applications. The proposed method utilizes the fact that for usual multimedia communication protocols like RTP excessively delayed packets are not used for reconstruction of transmitted data at the receiver (the packets are considered useless and discarded). The main idea of LACK is as follows: at the transmitter, some selected audio packets are intentionally delayed before transmitting. If the delay of such packets at the receiver is considered excessive, the packets are discarded by a receiver not aware of the steganographic procedure. The payload of the intentionally delayed packets is used to transmit secret information to receivers aware of the procedure. For unaware receivers the hidden data is "invisible".

Thus, if we are able to add enough delay to the certain packets at the transmitter side they will not be used for conversation reconstruction. Because we are using legitimate VoIP packets we must realize that in this way we may influence conversation quality. That is why we must consider the accepted level of packet loss for IP telephony and do not exceed it. This parameter is different for various speech codecs as researched in [16] e.g. 1% for G.723.1, 2% for G.729A, 3% for G.711 (if no additional mechanism is used to cope with this problem) or even up to 5% if mechanisms like PLC (Packet Loss Concealment) is used. So the number of packets that can be utilized for proposed steganographic method is limited. If we exceed packet loss threshold for chosen codec then there will be significant decrease in voice quality.

Let us consider RTP (audio) stream (S) that consists of n packets (a_n):

$$\boldsymbol{S} = (a_1, a_2, a_3, \ldots, a_n) \text{ and } n = T / I_f \tag{5}$$

where:

S denotes RTP (audio) stream,
a_n is n-th packet in the audio stream S,
n a number of packets in audio stream.

For every packet (a_n) at the transmitter output total delay (d_T) is equal to:

$$d_T(a_n) = \sum_{m=1}^{3} d_m \tag{6}$$

where:

d_1 is speech codec processing delay,
d_2 is codec algorithm delay,
d_3 is packetization delay.

Now, from stream S we choose i-th packet a_i with a probability (p_i):

$$p_i < p_{Lmax} \tag{7}$$

where:

$p_{Lmax} \in \{1\%, 5\%\}$ where 1% packet loss ratio is for VoIP without PLC mechanism and 5% packet loss ratio is for VoIP with PLC mechanism.

To be sure that the RTP packet will be recognized as lost at the receiver, as mentioned earlier, we have to delay it by certain value. For the proposed steganographic method two important parameters must be considered and set to the right value: amount of time by which the chosen packet is delayed (d_4), to ensure that it will be considered as lost at the receiver side and the packet loss probability (p_i) for this steganographic method, to ensure that in combination with p_{Lmax} probability will not degrade perceived quality of the conversation. To properly choose a delay value, we must consider capacity of the receiver's de-jitter buffer. The de-jitter buffer is used to alleviate the jitter effect (variations in packets arrival time caused by queuing, contention and serialization in the network). Its value (usually between 30-70 ms) is important for the end-to-end delay budget (which should not exceed 150 ms). That is why we must add d_4 delay (de-jitter buffer delay) to the d_T value for the chosen packet (a_i). If we ensure that d_4 value is equal or greater than de-jitter buffer delay at the receiver side the packet will be considered lost. So the total delay (d_T) for a_i packets with additional d_4 delay looks as follows (8):

$$d_T(a_i) = \sum_{m=1}^{4} d_m \tag{8}$$

where d_4 is de-jitter buffer delay.

Now that we are certain that the chosen packet (a_i) is considered lost at the receiver, we can use this packet's payload as a covert channel.

As mentioned earlier, the second important measure for proposed steganographic method is a probability p_i. To properly calculate its value we must consider the following simplified packet loss model:

$$p_T = 1 - (1 - p_N)(1 - p_i) \tag{9}$$

where:

p_T denotes total packet loss probability in the IP network that offers VoIP service with the utilizing of delayed audio packets,
p_N is a probability of packet loss in the IP network that offers VoIP service without the utilizing delayed audio packets (network packet loss probability),
p_i denotes a maximum probability of the packet loss for delayed audio packets.

When we transform (9) to calculate p_i we obtain:

$$p_i \le \frac{p_T - p_N}{1 - p_N} \tag{10}$$

From (10) one can see that probability p_i must be adjusted to the network conditions. Information about network packet loss probability may be gained e.g. from the

RTCP reports during the transmission. So, based on earlier description, we gain a covert channel with *PRBR* (Packet Raw Bit Rate) that can be expressed as follows:

$$PRBR = r \cdot \frac{I_f}{1000} \cdot p_i \quad [bits / packet] \tag{11}$$

where r is the speech codec rate.

And available bandwidth expressed in *RBR* (Raw Bit Rate) can be described with the following equation (12):

$$RBR = r \cdot p_i \quad [bits / s] \tag{12}$$

For example, consider a scenario with G.711 speech codec where: speech codec rate: r = 64 kbit/s and p_i = 0.5% and I_f = 20 ms. For these exemplary values RBR is 320 b/s and PRBR is 6.4 bits/packet. One can see that the available bandwidth of this covert channel is proportional to the speech codec frame rate, the higher the rate, the higher the bandwidth. So the total amount of information (I_T) that can be covertly transmitted during the call of length d (in seconds) is:

$$I_T = d \cdot RBR = d \cdot r \cdot p \quad [bits] \tag{13}$$

Proposed steganographic method has certain advantages. Most of all, although it is an application layer steganography technique, it is less complex than e.g. most audio steganography algorithms and the achieved bandwidth is comparable or even higher.

Steganalysis of LACK is harder than in case of other steganographic methods that are presented in this paper. This is mainly because it is common for IP networks to introduce losses. If the amount of the lost packets used for LACK is kept reasonable then it may be difficult to uncover hidden communication. Potential steganalysis methods include:

- Statistical analysis of the lost packets for calls in certain network. This may be done by passive warden (or other network node) e.g. based on RTCP reports (Cumulative number of packets lost field) or by observing RTP streams flow (packets' sequence numbers). If for some of the observed calls the number of lost packets is higher than it can indicate potential usage of the LACK method,
- Active warden which analyses all RTP streams in the network. Based on the SSRC identifier and fields: S*equence number* and *Timestamp* from RTP header it can identify packets that are already too late to be used for voice reconstruction. Then active warden may erase their payloads fields or simply drop them. One problem with this steganalysis method is how greatly the packets' identifying numbers must differ from other packets in the stream to be discarded without eliminating really delayed packets that may be still used for conversation. The size of jitter buffer at the receiver is not fixed (and may be not constant) and its size is unknown to active warden. If active warden drops all delayed packets then it could remove packets that still will be usable for voice reconstruction. In effect, due to active warden operations quality of conversation may deteriorate.

Further in-depth steganalysis for LACK is surely required and is considered as future work.

3.7 Medium Dependent Steganography

Medium dependent steganography typically uses layer 1 or layer 2 of ISO OSI RM. For VoIP e.g. in homogenous WLAN environment data link layer methods that depend on available medium like HICCUPS [22] system can be utilized. Exemplary, the data rate for this system is 216 kbit/s (IEEE 802.11g 54 Mbit/s, changing of frame error rate from 1.5% into 2.5%, bandwidth usage 40%).

It must be emphasized however that this steganographic method is difficult to implement as it require modification to network cards. Moreover, steganalysis for HICCUPS is difficult too as it necessary to analyze frames in physical layer of OSI RM model.

4 Experimental Evaluation of VoIP Streams Covert Channels Bandwidth

Total achieved covert channel bandwidth (B_T) for the whole VoIP transmission is a sum of each, particular bandwidth of each steganographic methods that are used during voice transmission (each steganographic subchannel). It can be expressed as follows:

$$B_T = \sum_{j=1}^{k} B_j \tag{14}$$

where:

B_T denotes a total bandwidth for the whole VoIP voice transmission (may be expressed in RBR or PRBR),
B_j describes a bandwidth of the covert channel created by each steganographic method used during VoIP call (may be expressed in RBR or PRBR),
k is a number of steganographic techniques used for VoIP call.

The value of B_T is not constant and depends on the following factors:

- The number of *steganographic techniques* applied to the VoIP call,
- The *choice of the speech codec* used. Three important aspects must be considered here: *compression rate* (e.g. G.711 achieves 64 kbit/s while G729AB only 8 kbit/s), *size of the voice frame* that is inserted into each packet and *voice packet generation interval.* Compression rate influences the available bandwidth of the steganographic methods that relay on it. The size of the voice frame (typically from 10 to 30 ms) and voice packet generation interval influence the number of packets in audio stream.
- If the mechanisms like *VAD/CNG/DTX* are used. Some of the speech codecs have those mechanisms built-in, for some of them they must be additionally implemented. These solutions influence the number of packets that are generated during VoIP call. The lower number of packets are transmitted the lower total covert channel bandwidth B_T value.
- The *probability value of the packet loss* in IP network. Firstly, if this value is high we lose certain number of packets that are sent into the network, so the information covertly transferred within them is also lost. Secondly, while using delayed audio packets steganography we must adjust the probability of the intentionally lost

packets to the level that exists inside the network to be sure that the perceived quality of the call is not degenerated.
- Less important steganographic methods specific conditions like: how often are RTCP reports are sent to the receiving party or if security mechanisms for communication are used.

To evaluate measures presented at the beginning of Section 3 the following test scenario, as depicted in Fig. 4, has been setup. Two SIP User Agents were used to make a call – the signalling messages were exchanged with SIP proxy and the audio streams flowed directly between endpoints. Moreover RTCP protocol was used to convey information about network performance. Audio was coded with ITU-T G.711 A-law PCM codec (20 ms of voice per packet, 160 bytes of payload). The ACD (Average Call Duration) for this experiment was chosen based on duration of the call for Skype service [21] and for other VoIP providers. In [7] results obtained that ACD for Skype is about 13 minutes, while VoIP providers typically uses a value between 7 and 11 minutes. That is why we chose ACD for the experiment at 9 minutes. There were 30 calls performed and all diagrams show average results.

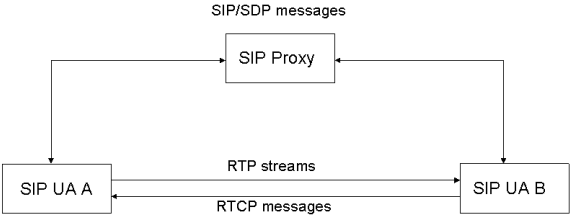


Fig. 4. VoIP steganography experimental test setup

The calls were initiated by *SIP UA A* and the incoming traffic was sniffed at *SIP UA B*. This way we were able to measure covert channel behavior for only one direction traffic flow. Based on the analysis of the available steganographic methods in section 3 the following steganographic techniques were used during the test (and the amount of data that were covertly transferred) as presented in Table 2.

Table 2. Steganographic methods used for experiment and their PRBR

Steganographic method	Chosen PRBR
IP/UDP protocol steg.	32 bits/packet
RTP protocol steg.	16 bits/packet
RTCP steg.	192 bits/packet
LACK	1280 bits/packet (used 0.1% of all RTP packets)
QIM (audio watermarking)	0.6 bits/packet

We chose these steganographic methods for the experiment because they are easy to implement and/or they are our contribution. Besides they are the most natural choice for VoIP communication (based on the analysis' results from section 3) and,

additionally, they represent different layers steganography. It is also important to note that assumed PRBR values for these methods were chosen to be reasonable in steganalysis context. We are interested however only in estimating a total amount of data that can be covertly transferred during the typical conversation phase of the VoIP call, and not how hard is to perform steganalysis. We want to see if the threat posed by steganography applied to VoIP is serious or not.

Achieved results of the experiment are presented below. First in Table 3 traffic flow characteristics, that were captured during performed VoIP calls are presented.

Table 3. Types of traffic distribution average results

Type of traffic	Percent [%]
SIP messages	0.016
RTP packets	99.899
RTCP reports	0.085

From Table 3 can be concluded that the steganographic methods that that utilizes RTP packets have the most impact on VoIP steganography as they cover 99.9% of the whole VoIP traffic. Next in Fig. 5 and Fig. 6 averaged results of the covert data flow distribution (RBR and PRBR respectively) during the average call are presented.

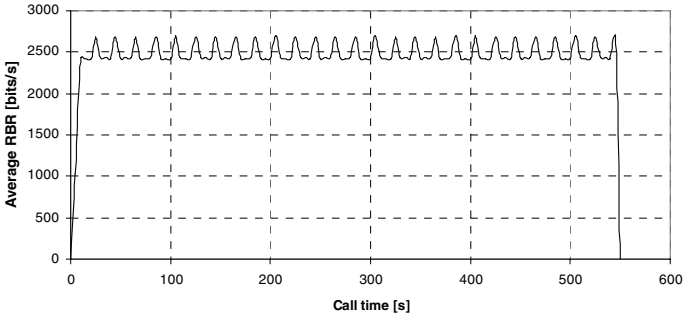


Fig. 5. Covert transmission data flow distribution for the experimental setup

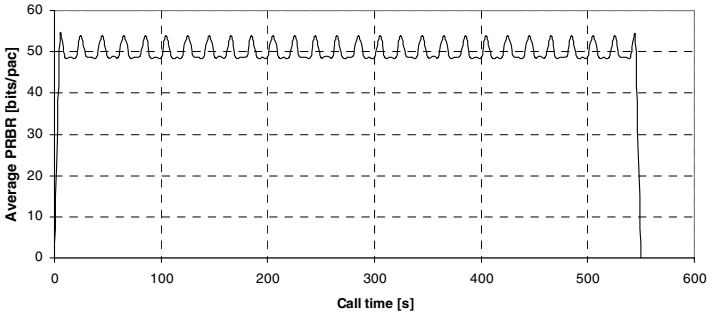


Fig. 6. PRBR during the average call

As one can see VoIP covert channels bandwidth expressed in RBR and PRBR changes in rather constant range during the call (between 2450 and 2600 bits/s for RBR and between 48 and 53 bits/packet for PRBR). The periodic peaks for curves presented in both figures are caused by steganographic bandwidth provided by LACK method. In every certain period of time packets are selected to be intentionally delayed and their payloads carry steganograms. For instants when these packets reach receiver the steganographic bandwidth increases. For this experiment the following average values were obtained and were presented in Table 4:

Table 4. Experimental results for typical call (for one direction flow only)

Measure	Value	Standard Deviation
Average total amount of covert data	1364170 [bits]	4018.711
Average RBR	2487,80 [bits/s]	4.025
Average PRBR	50,04 [bits/packet]	2.258

From the Table 4 we see that during the typical call one can transfer more than 1.3 Mbits (170 KB) of data in one direction with RBR value at about 2.5 kbit/s (50 bits/packet for PRBR).

Table 5. Types of traffic and theirs covert bandwidth fraction

Type of traffic	Bandwidth fraction [%]	Bandwidth fraction [%] per steganographic method	
RTP packets	99.646	IP/UDP	64.11
		RTP	32.055
		Delayed audio packets	2.633
		Audio watermarking	1.202
RTCP reports	0.354	-	

As results from Table 5 show vast part of covert channels' bandwidth for VoIP is provided by network steganography (for protocols IP/UDP it is about 64% and for RTP 32%). Next steganographic method is delayed audio packets steganography (about 2.6%) and audio watermarking (about 1.2%). RTCP steganography provides only minor bandwidth if we compare it with other methods.

5 Conclusions

In this paper we have introduced two new steganographic methods: one of them is RTP and RTCP protocols steganography and the second is intentionally delayed audio packets steganography (LACK). We also briefly described other existing steganographic

methods for VoIP streams. Next, for chosen steganographic method the experiment was performed. Obtained results showed that during typical VoIP call we are able to send covertly more than *1.3 Mbits* of data in one direction.

Moreover, the next conclusion is that the most important steganographic method in VoIP communication experiment is IP/UDP/RTP protocols steganography, while it provides over 96% of achieved covert bandwidth value. Other methods that contribute significantly are delayed audio packets steganography (about 2.6%) and audio watermarking techniques (about 1.2%).

Based on the achieved results we can conclude that total covert bandwidth for typical VoIP call is high and it is worth noting that not all steganographic methods were chosen to the experiment. Steganalysis may limit achieved bandwidth of the covert channels to some extent. But two things must be emphasized. Firstly, currently there is no documented active warden implementation thus there are no real counter measurements applied in IP networks so all the steganographic methods can be used for this moment. Secondly, analyzing each VoIP packet in active warden for every type of steganography described here can potentially lead to loss in quality due to additional delays – this would require further study in future. So, whether we treat VoIP covert channels as a potential threat to network security or as a mean to improve VoIP functionality we must accept the fact that the number of information that we can covertly transfer is significant.

References

1. Ahsan, K., Kundur, D.: Practical Data Hiding in TCP/IP. In: Proc. of: Workshop on Multimedia Security at ACM Multimedia 2002, Juan-les-Pins, France (2002)
2. Baugher, M., McGrew, D., Naslund, M., Carrara, E., Norrman, K.: The Secure Real-time Transport Protocol (SRTP), IETF, RFC 3711 (2004)
3. Bender, W., Gruhl, D., Morimoto, N., Lu, A.: Techniques for Data Hiding. IBM. System Journal. 35(3,4), 313–336 (1996)
4. Cuervo, F., Greene, N., Rayhan, A., Huitema, C., Rosen, B., Segers, J.: Megaco Protocol Version 1.0. IETF, RFC 3015 (2000)
5. Fisk, G., Fisk, M., Papadopoulos, C., Neil, J.: Eliminating Steganography in Internet Traffic with Active Wardens. In: Petitcolas, F.A.P. (ed.) IH 2002. LNCS, vol. 2578, pp. 18–35. Springer, Heidelberg (2003)
6. Giffin, J., Greenstadt, R., Litwack, P.: Covert Messaging Through TCP Timestamps. In: Proc. of: Privacy Enhancing Technologies Workshop (PET), pp. 194–208 (2002)
7. Guha, S., Daswani, N., Jain, R.: An Experimental Study of the Skype Peer-to-Peer VoIP System. In: Proc. of: IPTPS – Sixth International Workshop on Peer-to-Peer Systems (2006)
8. ITU-T Recommendation H.323: Infrastructure of Audiovisual Services – Packet-Based Multimedia Communications Systems Version 6, ITU-T (2006)
9. Johnston, A., Donovan, S., Sparks, R., Cunningham, C., Summers, K.: Session Initiation Protocol (SIP) Basic Call Flow Examples. IETF, RFC 3665 (2003)
10. Korjik, V., Morales-Luna, G.: Information Hiding through Noisy Channels. In: Proc. of: 4th International Information Hiding Workshop, Pittsburgh, PA, USA, pp. 42–50 (2001)
11. Lucena, N., Lewandowski, G., Chapin, S.: Covert Channels in IPv6. In: Danezis, G., Martin, D. (eds.) PET 2005. LNCS, vol. 3856, pp. 147–166. Springer, Heidelberg (2006)

12. Mazurczyk, W., Kotulski, Z.: New Security and Control Protocol for VoIP Based on Steganography and Digital Watermarking. In: Proc. of: IBIZA 2006, Kazimierz Dolny, Poland (2006)
13. Mazurczyk, W., Kotulski, Z.: New VoIP Traffic Security Scheme with Digital Watermarking. In: Górski, J. (ed.) SAFECOMP 2006. LNCS, vol. 4166, pp. 170–181. Springer, Heidelberg (2006)
14. Mazurczyk, W., Szczypiorski, K.: Covert Channels in SIP for VoIP Signalling. In: Jahankhani, H., Revett, K., Palmer-Brown, D. (eds.) ICGeS 2008. CCIS, vol. 12, pp. 65–72. Springer, Heidelberg (2008)
15. Murdoch, S., Lewis, S.: Embedding Covert Channels into TCP/IP. Information Hiding, 247–266 (2005)
16. Na, S., Yoo, S.: Allowable Propagation Delay for VoIP Calls of Acceptable Quality. In: Chang, W. (ed.) AISA 2002. LNCS, vol. 2402, pp. 469–480. Springer, Heidelberg (2002)
17. Petitcolas, F., Anderson, R., Kuhn, M.: Information Hiding – A Survey. IEEE Special Issue on Protection of Multimedia Content (1999)
18. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A.: SIP: Session Initiation Protocol. IETF, RFC 3261 (2002)
19. Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V.: RTP: A Transport Protocol for Real-Time Applications, IETF, RFC 3550 (2003)
20. Servetto, S.D., Vetterli, M.: Communication Using Phantoms: Covert Channels in the Internet. In: Proc. of IEEE International Symposium on Information Theory (2001)
21. Skype, http://www.skype.com
22. Szczypiorski, K.: HICCUPS: Hidden Communication System for Corrupted Networks. In: Proc. of ACS 2003, Międzyzdroje, Poland, October 22-24, 2003, pp. 31–40 (2003)
23. Takahashi, T., Lee, W.: An Assessment of VoIP Covert Channel Threats. In: Proc. of 3rd International Conference on Security and Privacy in Communication Networks (SecureComm 2007), Nice, France (2007)
24. US Department of Defense – Department of Defense Trusted Computer System Evaluation Criteria, DOD 5200.28-STD (The Orange Book) (1985)
25. Zander, S., Armitage, G., Branch, P.: A Survey of Covert Channels and Countermeasures in Computer Network Protocols. IEEE Communications Surveys & Tutorials, 3rd Quarter 2007 9(3), 44–57 (2007)

VICE OVER IP

A growing cadre of criminals is hiding secret messages in voice data

By JÓZEF LUBACZ, WOJCIECH MAZURCZYK & KRZYSZTOF SZCZYPIORSKI

MICK WIGGINS

7:00 P.M., SHANGHAI

An employee of an electronic equipment factory uploads a music file to an online file-sharing site. Hidden in the MP3 file (Michael Jackson's album *Thriller*) are schematics of a new mobile phone that will carry the brand of a large American company. Once the employee's Taiwanese collaborators download the file, they start manufacturing counterfeit mobile phones essentially identical to the original—even before the American company can get its version into stores.

3:30 P.M., SOMEWHERE IN AFGHANISTAN

A terrorist hunted by the U.S. Federal Bureau of Investigation posts an excerpt from the motion picture *High School Musical Three: Senior Year* on Facebook. Inside are hidden instructions for a bomb attack on a commuter rail line in southern Europe. Later that day, terrorists based in Athens follow the instructions to plan a rush hour attack that kills hundreds of people.

4:00 A.M., MALIBU, CALIF.

A very famous actor (VFA) has a brief conversation with a well-known director (WKD) over Skype, an application that lets them make free voice calls over the Internet. They discuss the medical problems of VFA's cat in great detail. When the conversation is over, WKD's computer has a sleazy new addition—in a folder on his desktop, there is a picture of a nude teenager, along with her mobile number and the date and time at which WKD will meet her at VFA's pool party for a photo session.

WHAT ALL these scenarios have in common is an information-smuggling technique called steganography—the communication of secret messages inside a perfectly innocent carrier. Think of steganography as meta-encryption: While encryption protects messages from being read by unauthorized parties, steganography lets the sender conceal the fact that he has even sent a message. After the 11 September attacks in 2001, rumors flew that they had been carried out with some help from steganography. A 2001 *New York Times* article described fake eBay listings in which routinely altered pictures of a sewing machine contained malevolent cargo. The link to 9/11 was never proved or disproved, but after those reports, the interest in steganographic techniques and their detection greatly increased.

Steganography use is on the rise, and not just among criminals, hackers, child pornographers, and terrorists. Persecuted citizens and dissidents under authoritarian regimes use it to evade government censorship, and journalists can use it to conceal sources. Investigators even use it on occasion to bait and trap people involved in industrial espionage: In the 1980s, to trace press leaks of cabinet documents, British Prime Minister Margaret Thatcher had government word processors altered to encode a specific user identity in the spaces between words. When leaked material was recovered, the identity of the leaker could be established by analyzing the pattern of those spaces.

Steganography is evolving alongside technology. A few years ago the cutting edge in steganographic tools involved hiding messages inside digital images or sound files, known as carriers, like that *Thriller* MP3. The technique quickly evolved to include video files, which are relatively large and can therefore conceal longer messages.

Now steganography has entered a new era, with stupendously greater potential for mischief. With the latest techniques, the limitations on the length of the message have basically been removed. Consider our example involving the use of Skype. Whereas the first two examples each required a carrier—an MP3 song and a video—there was no such requirement for the transmission of that nude photo. The data were secreted among the bits of a digital Voice over Internet Protocol conversation. In this new era of steganography, the mule that coconspirators are using is not the carrier itself but the communication protocols that govern the carrier's path through the Internet. Here's the advantage: The longer the communicators talk, the longer the secret message (or more detailed the secret image) they can send. Most strikingly, the concealment occurs within data whose inherent ephemerality makes the hidden payload nearly impossible to detect, let alone thwart.

We call this new technique network steganography. In our research at the Network Security Group at Warsaw University of Technology, we are studying the ever-evolving spectrum of carrier technologies, the increasing difficulty of detection as more sophisticated carriers leave fewer traces, and the implications of both for law enforcement and homeland security. Our work at Warsaw is literally self-defeating: We figure out the most advanced ways of doing network steganography and then design methods to detect them.

NETWORK STEGANOGRAPHY is a modern version of an old idea. You could argue that steganography helped spark the first major conflict between Greece and the Persian Empire. A classic use of steganography took place in 494 B.C., when Histiaeus, the ruler of Miletus, tried to instigate an Ionian revolt against the Persians. He shaved his favorite slave's head, tattooed it with a message, and waited for the slave's hair to grow back and obscure the tattoo. Then he sent the slave to his destination, where the intended recipient shaved the slave's head and read the message. The ensuing Ionian revolution lasted for half a century. In the 19th and 20th centuries, rapidly evolving warfare and espionage brought many innovations in steganography: Invisible ink, microdots, and Thatcher's word-processor trick are only a few among many.

With today's technology, information can be smuggled in essentially any type of digital file, including JPEGs or bitmaps, MP3s or WAV files, and MPEG movies. More than a hundred such steganographic applications are freely available on the Internet. Many of these programs are slick packages whose use requires no significant technical skills whatsoever. Typically, one mouse click selects the carrier, a second selects the secret information to be sent, and a third sends the message and its secret cargo. All the recipient needs is the same program the sender used; it typically extracts the hidden information within seconds.

CARRIER EVOLUTION

Steganography has been used for at least 2500 years to disguise secret messages. In its earliest forms, the carriers were physical, but as technology evolved, so did carriers.

494 B.C.
HEAD TATTOO

Histiaeus tattoos a secret message onto a slave's shaved head, waits for the hair to regrow, and sends the slave to the intended recipient, who shaves off the hair to read the message.

480 B.C.
BEESWAX

Demaratus writes a secret message on a wooden tablet to warn the Greeks of Persian attack, and then covers it with many coats of wax.

1558
EGGS

Italian scientist Giambattista della Porta discovers how to hide a message inside a hard-boiled egg: Write on the shell using an ink made from a mixture of alum and vinegar. The solution leaves no trace on the surface,

ANOUCHKA/ISTOCKPHOTO

A SINGLE 6-MINUTE MP3 OCCUPIES 30 MB, ENOUGH TO CONCEAL EVERY PLAY SHAKESPEARE EVER WROTE

Any binary file can be concealed—for instance, pictures in unusual formats, software (a nasty virus, say), or blueprints. The favored carrier files are the most common ones, like JPEGs or MP3s. This emphasis on popular file formats increases the anonymity of the entire transaction, because these file types are so commonplace that they don't stick out.

The one limitation that steganographers have traditionally faced is file size. The rule of thumb is that you can use 10 percent of a carrier file's size to smuggle data. For an ambitious steganographer, that could be a problem: Imagine an electronic equipment factory employee trying to explain to the IT department why he has to send his mother a 100-megabyte picture of the family dog. For that reason, steganographers soon turned to audio and video files. A single 6-minute song, in the MP3 compression format, occupies 30 MB; it's enough to conceal every play Shakespeare ever wrote.

And yet, even with these precautions, conventional steganography still has an Achilles' heel: It leaves a trail. Pictures and other e-mail attachments stored on a company's outgoing e-mail servers retain the offending document. Anything sent has to bounce through some kind of relay and can therefore be captured, in theory.

Steganography poses serious threats to network security mainly by enabling confidential information leakage. The new crop of programs leaves almost no trail. Because they do not hide information inside digital files, instead using the protocol itself, detecting their existence is nearly impossible.

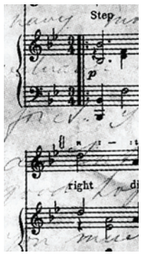
ALL THE new methods manipulate the Internet Protocol (IP), which is a fundamental part of any communication, voice or text based, that takes place on the Internet. The IP specifies how information travels through a network. Like postal service address standards, IP is mainly in charge of making sure that sender and destination addresses are valid, that parcels reach their destinations, and that those parcels conform to certain guidelines. (You can't send e-mail to an Internet address that does not use a 32-bit or 128bit number, for example.)

but the message is retrieved by removing the shell and reading the egg.

1800s
NEWSPAPER CODE

During the Victorian era, lovers send secret letters by punching holes above certain letters. When the marked letters are combined, the message can be read.

1915
INVISIBLE INK



During World War I, entertainer and German spy Courtney de Rysbach performs in shows all over Britain as a cover for gathering information. Using invisible ink, Rysbach encodes secret messages by writing them in invisible ink on sheets of music.

1941
MICRODOTS

During World War II, German agents photographically shrink a page of text down to a 1-millimeter dot. The microdot is then hidden on top of a period in an otherwise unremarkable letter.

All traffic, be it e-mail or streaming video, travels via a method called packet switching, which parcels out digital data into small chunks, or packets, and sends them over a network shared by countless users. IP also contains the standards for packaging those packets.

Let's say you're sending an e-mail. After you hit the Send button, the packets travel easily through the network, from router to router, to the recipient's in-box. Once these packets reach the recipient, they are reconstituted into the full e-mail.

The important thing is that the packets don't need to reach their destination in any particular order. IP is a "connectionless protocol," which means that one node is free to send packets to another without setting up a prior connection, or circuit. This is a departure from previous methods, such as making a phone call in a public switched telephone network, which first requires synchronization between the two communicating nodes to set up a dedicated and exclusive circuit. Within reason, it doesn't matter when packets arrive or whether they arrive in order.

As you can imagine, this method works better for order-insensitive data like e-mail and static Web pages than it does for voice and video data. Whereas the quality of an e-mail message is immune to traffic obstructions, a network delay of even 20 milliseconds can very much degrade a second or two of video.

To cope with this challenge, network specialists came up with the Voice over Internet Protocol (VoIP). It governs the way voice data is broken up for transmission the same way IP manages messages that are less time sensitive. VoIP enables data packets representing a voice call to be split up and routed over the Internet.

The connection of a VoIP call consists of two phases: the signaling phase, followed by the voice-transport phase. The first phase establishes how the call will be encoded between the sending and receiving computers. During the second phase, data are sent in both directions in streams of packets. Each packet, which covers about 20 milliseconds of conversation, usually contains 20 to 160 bytes of voice data. The connection typically conveys between 20 and 50 such packets per second.

Telephone calls must occur in real time, and significant data delays would make for an awkward conversation. So to ferry a telephone call over the Internet, which was not originally intended for voice communications, VoIP makes use of two more communications protocols, which had to be layered on top of IP: The Real-Time Transport Protocol (RTP) and the User Datagram Protocol (UDP). The RTP gets time-sensitive video and audio data to its destination fast and so has been heavily adopted in much of streaming media, such as telephony, video teleconference applications, and Web-based push-to-talk features. To do that, it relies in turn on the UDP.

Because voice traffic is so time critical, UDP does not bother to check whether the data are reliable, intact, or even in order. So in a VoIP call, packets are sometimes stuck in out

ALL THREE STEGANOGRAPHIC IDEAS WE'VE OUTLINED HERE ARE SO SIMPLE, WE'RE CERTAIN THAT REAL-LIFE APPLICATIONS ARE ALREADY OUT THERE

of sequence. But that's not a big deal because the occasional misplaced packet won't significantly affect the quality of the phone call. The upshot of UDP is that the protocol opens a direct connection between computers with no mediation, harking back to the era of circuit switching: Applications can send data packets to other computers on a connection without previously setting up any special transmission channels or data paths. That means it's completely private.

Compared to old-fashioned telephony, IP is unreliable. That unreliability may result in several classes of error, including data corruption and lost data packets. Steganography exploits those errors.

Because these secret data packets, or "steganograms," are interspersed among many IP packets and don't linger anywhere except in the recipient's computer, there is no easy way for an investigator—who could download a suspect image or analyze an audio file at his convenience—to detect them.

TO BETTER UNDERSTAND what security officials will soon have to deal with, we designed and developed three flavors of network steganography, all of which manipulate IP. The three methods we developed are Lost Audio Packet Steganography, or LACK; Hidden Communication System for Corrupted Networks (HICCUPS); and Protocol Steganography for VoIP application. As their names imply, these techniques exploit lost packets, corrupted packets, and hidden or unused data fields in the VoIP transmission protocol. LACK hides information in packet delays, HICCUPS disguises information as natural "distortion" or noise, and Protocol Steganography hides information in unused data fields.

In regular VoIP telephony, excessively delayed packets containing voice samples are considered useless by the receiver and thus discarded. LACK exploits this mechanism to transmit hidden data. Some of the sender's packets are intentionally delayed, and the steganograms are stowed away inside those delayed packets. To any node that is not "in the know"—that is, a nearby computer that does not have the steganography program installed—they appear useless and are ignored. But if the receiver has the proper software to understand the steganography, it will not discard the excessively delayed packets. It will know that these contain the hidden data [see diagram, "Hidden in the Network"].

The transmission capacity for this scheme depends on the system used to encode the voice and on the quality of the network—specifically, how well it handles packet loss and delays. Using a standard 32-bit-per-second codec, and accounting for a 3 percent packet loss introduced by the network and a 0.5 percent packet loss introduced by LACK itself, a smuggler could transmit about 160 bits per second. At that rate you might be able to transmit a medium-size, 13-kilobyte image or a 2000-word text file during a typical 9- to 13-minute VoIP conversation.

LACK's main selling points are that it is simple to use and hard to detect. The only way it could be detected is if the user tried to hide too many secret packets. In that case, the number of intentionally delayed packets—and therefore the introduced delay—would create a suspiciously abnormal voice connection that might attract the attention of any security officials monitoring the line. If the call was completed before those officials could intercept the packets, however, there would be nothing they could do to try to uncover and assemble the steganograms.

Where LACK relies on lost packets to smuggle steganograms, HICCUPS takes advantage of corrupted packets. HICCUPS is fast. Let's say you have an IEEE 802.11g network with a transmission capacity of 54 megabits per second, with 10 terminals and a 5 percent rate of corrupted frames. Over such a network, you could send hidden data at a rate higher than 200 kilobits per second. That's almost as fast as the ISDN lines that were all the rage in the 1990s.

HICCUPS works on wireless local area networks, such as plain old coffee shop Wi-Fi. In such a wireless environment, data are transmitted by a method called broadcasting, which shuttles data in groups called frames. Like many courier services, broadcasting doesn't concern itself with the contents of the data or whether the data contain errors. When a wireless network detects an error in a frame, the computer simply drops that corrupted frame. The responsibility for detecting dropped frames (and retransmitting them if necessary) is left to the origin and destination terminals.

So in a wireless local-area network, all the user terminals (laptops, for the most part) must have a way of differentiating good packets from corrupted ones. This error-checking mechanism is called the checksum, a kind of signature against which the integrity of the packets can be confirmed. The checksum is a numerical value assigned to a data packet based on the number of bits in that packet. A checksum program uses that value to authenticate that the data hasn't been corrupted.

When the receiver's computer gets a packet, it checks for errors using that packet's checksum. Normally, if the checksum is wrong, the computer discards that packet. But if a terminal has

1980s
WATERMARKING

AP PHOTO

In the 1980s, to trace press leaks of cabinet documents, British Prime Minister Margaret Thatcher has government word processors altered to encode a specific user identity in the spaces between words.

1990s
DIGITAL STEGANOGRAPHY

Researchers develop methods to secretly embed a signature in digital pictures and audio, exploiting the human visual system's varying sensitivity to contrast.

2003
STREAMING VIDEO

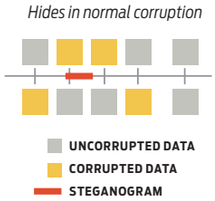
Video steganography is similar to image steganography, but more information may be transported in a stream of images.

2007
NETWORK STEGANOGRAPHY

New methods focus on using free or unused fields in a protocol's headers.

HIDDEN IN THE NETWORK

HICCUPS
(CORRUPTED PACKETS)

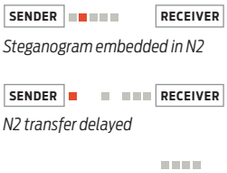


Highest information density HICCUPS [red] hides in the "noise" of natural distortion [orange] in an otherwise normal VoIP telephone call [gray].

Difficult to use Because this method requires hardware that can generate wrong checksums, it is difficult to use.

200 kilobits per second are transmitted during a typical 9–13 minute VoIP call.

LACK
(LOST AUDIO PACKETS)

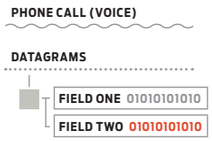


Lowest information density Excessively delayed packets are dropped by the receiver. LACK delays packets on purpose, encodes the hidden data, and decodes the steganograms when they arrive.

Hardest to detect Used carefully, LACK delays only a small percentage of packets.

160 bits per second are transmitted during a typical call.

PROTOCOL STEGANOGRAPHY
(HIDDEN FIELDS)



VoIP breaks voice data into datagram packets. Each packet contains frequently changing fields, such as authentication.

Easiest to use Each bit (phone-call data) contains data fields. Some fields contain frequently changing data, which can be wholly or partially replaced with a steganogram.

Hard to detect By replacing the authentication field, the user sacrifices security.

1–300 bits per second are transmitted during a typical call.

the right steganography program installed, it won't discard these intentionally wrong checksums—instead, it will know that these are precisely the data packets to scan for steganograms.

HICCUPS is more difficult to pull off than LACK. That's because this method requires a wireless card that can control frame checksums (good luck finding one of those at RadioShack). Network cards create checksums at the hardware level. We have applied for a patent in Poland for a HICCUPS-enabled card that can control checksums, but so far we haven't built our own card. Detecting HICCUPS wouldn't be easy. You'd need some way of observing the number of frames with incorrect checksums. If the number of those frames is statistically anomalous, then you might suspect the transmission of hidden information. Another way of detecting HICCUPS would analyze the content of those dropped—and therefore retransmitted—frames in order to detect the differences between the dropped and retransmitted frames. Major differences in these frames would constitute an obvious clue to nefarious goings-on.

Any of these detection methods, of course, would require not only that an investigator be aware that a transmission was about to take place but also that he be equipped with the right equipment, ready to monitor the conversation and intercept bits. Such a situation would be unlikely, to put it mildly.

The third method, Protocol Steganography, is a common name for a group of methods that use another aspect of IP: packet header fields. These fields are like sophisticated address labels that identify the contents of data packets to the recipient. Steganograms can be hidden inside unused, optional, or partial fields, because any data in these fields can be replaced without affecting the connection. Some of the more ham-fisted steganography techniques simply replace the content of the unused or optional fields with steganograms. But that would be relatively easy to detect and even jam.

So, to evade detection by simple analysis, the more sophisticated variant of Protocol Steganography uses fields in which the content changes frequently. For example, some of the more esoteric VoIP fields carry security data for authentication purposes. That little authentication subfield changes frequently during the course of a normal call. A steganogram smuggled inside one of its many randomly changing packets would be extremely hard to detect. Of course, there is a trade-off: The user would also sacrifice security, meaning that his or her conversation could be intercepted more easily.

Minimizing the threat of evolving steganography methods requires an in-depth understanding of how network protocols function and how they can be exploited to hide data. The problem is, however, the complexity of today's network protocols. All three steganographic ideas we've outlined here are so simple, we're certain that real-life applications are sure to come, if they aren't already out there. In fact, much more sophisticated methods will appear as Internet communication evolves from VoIP to other real-time media communications, such as video chat and conferencing.

THE ANONYMITY OF STEGANOGRAPHY might be good for privacy, but it also multiplies the threats to individuals, societies, and states. The trade-off between the benefits and threats involves many complex ethical, legal, and technological issues. We'll leave them for other thinkers and other articles.

What we're trying to do is understand what kind of potential contemporary communication networks have for enabling steganography, and in effect, create new techniques so that we can figure out how to thwart them. Some readers may object to our detailed descriptions of how these methods can be harnessed. But we would counter that unless someone shows how easy all this is, researchers won't understand the urgency and be inspired to develop protective measures. Not only can VoIP steganography be implemented in telephony tools that require a laptop or PC (like Skype), it can also be used in hard phones, such as the Android VoIP-enabled mobile phones that are starting to proliferate. Steganography on a phone is more difficult, because it requires access to the device's operating system, but no one should doubt that committed individuals will have no trouble rising to the challenge. As George Orwell once wrote, "On the whole human beings want to be good, but not too good, and not quite all the time." ❑

AP PHOTO

Covert Channels in SIP for VoIP Signalling

Wojciech Mazurczyk and Krzysztof Szczypiorski

Warsaw University of Technology, Faculty of Electronics and Information Technology, Institute of Telecommunications, 15/19 Nowowiejska Str.
00-665 Warsaw, Poland
{W.Mazurczyk,K.Szczypiorski}@tele.pw.edu.pl

Abstract. In this paper, we evaluate available steganographic techniques for SIP (Session Initiation Protocol) that can be used for creating covert channels during signaling phase of VoIP (Voice over IP) call. Apart from characterizing existing steganographic methods we provide new insights by introducing new techniques. We also estimate amount of data that can be transferred in signalling messages for typical IP telephony call.

Keywords: VoIP, SIP, information hiding, steganography.

1 Introduction

Steganography is a process of hiding secret data inside other, normally transmitted data. Usually, it means hiding of a secret message within an ordinary message and its extraction at the destination point. In an ideal situation, anyone scanning this information will fail to know whether it contains covert data or not. A covert channel [9] is one of the most popular steganographic techniques that can be applied in the networks. The covert channel offers an opportunity to "manipulate certain properties of the communications medium in an unexpected, unconventional, or unforeseen way, in order to transmit information through the medium without detection by anyone other than the entities operating the covert channel" [17].

Nowadays, VoIP is one of the most popular services in IP networks. It stormed into the telecom market and changed it entirely. As it is used worldwide more willingly, the traffic volume that it generates is still increasing. That is why VoIP traffic may be used to enable hidden communication throughout IP networks. Applications of the VoIP covert channels differ as they can pose a threat to the network communication or can be used to improve the functioning of VoIP (e.g. security like in [11] or quality of service like in [10]). The first application of the covert channel is more dangerous as it can lead to the confidential information leakage. It is hard to assess what bandwidth of a covert channel poses a serious threat, it depends on the security policy that is implemented in the network. For example: The US Department of Defense specifies in [16] that any covert channel with bandwidth higher than 100 bps must be considered insecure for average security requirements. Moreover for high security requirements it should not exceed 1 bps.

In this paper we present available covert channels that may be utilized for hidden communication for SIP protocol used as a signalling protocol for VoIP service.

H. Jahankhani, K. Revett, and D. Palmer-Brown (Eds.): ICGeS 2008, CCIS 12, pp. 65–72, 2008.

Moreover, we introduce new steganographic methods that, to our best knowledge, were not described earlier. For each of these methods we estimate potential bandwidth to later evaluate how much information may be transferred in a typical IP telephony call.

The paper is organized as follows. In Section 2 we circumscribe the types of VoIP traffic and a general communication flow for IP telephony call. In Section 3, we describe available steganographic methods that may be used to create covert channels for signalling messages. Then, in Section 4, we estimate a total amount of data that may be transferred with use of the SIP protocol. Finally, Section 5 concludes our work.

2 VoIP Communication Flow

VoIP is a real-time service that enables voice conversations through IP networks. Protocols that are used for creating IP telephony may be divided into four following groups:

a. *Signalling protocols* which allow to create, modify, and terminate connections between the calling parties. Nowadays the most popular are SIP [14], H.323 [6], and H.248/Megaco [3],
b. *Transport protocols,* from which the most important one is RTP [15], which provides end-to-end network transport functions suitable for applications transmitting real-time audio. RTP is used in conjunction with UDP (or rarely TCP) for transport of digital voice stream,
c. *Speech codecs* e.g. G.711, G.729, G.723.1 that allow to compress/decompress digitalized human voice and prepare it for transmitting in IP networks,
d. Other *supplementary protocols* like RTCP [15], SDP [5], or RSVP etc. that complete VoIP functionality. For purposes of this paper we explain the role of SDP protocol, which is used with SIP messages to describe multimedia sessions and to negotiate their parameters.

IP telephony connection may be divided into two phases: a *signalling phase* and a *conversation phase*. In both of these phases certain types of traffic are exchanged between calling parties. In this paper we consider VoIP service based on the SIP signaling protocol (with SDP) and RTP (with RTCP as control protocol) for audio stream transport. It means that during the signalling phase of the call certain SIP messages are exchanged between SIP endpoints (called: SIP User Agents). SIP messages usually traverse through SIP network servers: proxies or redirects that help end-users to locate and reach each other. After this phase, a conversation phase begins, where audio (RTP) streams flow bi-directly between a caller and a callee. VoIP traffic flow described above and distinguished phases of the call are presented in Fig. 1. For more clarity, we omitted the SIP network servers in this diagram (as they interpret the signalling messages and can modify only a few fields of SIP message which we will not use for steganographic purposes). Also potential security mechanisms in traffic exchanges were ignored.

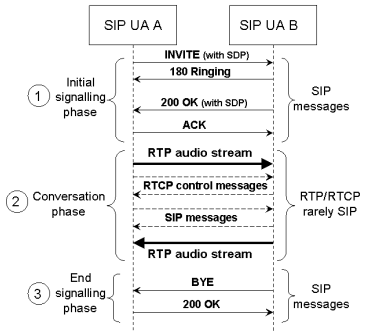


Fig. 1. VoIP call setup based on SIP/SDP/RTP/RTCP protocols (based on [7])

3 VoIP Signalling Covert Channels Overview and New Insights

In this section we will provide an overview of existing and new steganographic techniques used for creation of covert channels for VoIP that may be used during signalling phase of the call. To calculate potential amount of information that may be exchanged between calling parties we define *total amount of covert data* (B_T) that refers to information transferred (in bits) in SIP signalling messages (in one direction) with the use of all described steganographic methods. It can be expressed as:

$$B_T = \sum_{j=1}^{k} B_j \tag{1}$$

where: B_j describes amount of covert data transferred with use of the covert channel created by each steganographic method used during VoIP signalling and k is a number of steganographic techniques used for VoIP signalling.

Traffic generated during the signalling phase of the call is provided from SIP signalling messages that are exchanged between both endpoints. That is why, we can point out the following steganographic methods to create covert channels:

- *TCP/UDP/IP* steganography in transport and network layers of TCP/IP stack,
- *SIP/SDP* protocols steganography in application layer of TCP/IP stack.

3.1 IP/TCP/UDP Protocols Steganography

TCP/UDP/IP protocols steganography utilizes the fact that only few fields of headers in the packet are changed during communication process ([12], [1], [13]). Covert data is usually inserted into redundant fields (provided, but often unneeded) for abovementioned protocols and then transferred to the receiving side. In TCP/IP stack, there are a number of methods available, whereby covert channels may be established and data can be exchanged between communication parties secretly. An analysis of the headers of typical TCP/IP protocols e.g. IP, UDP, TCP, but also e.g. HTTP (Hypertext Transfer Protocol) or ICMP (Internet Control Message Protocol) results in fields that are either unused or optional [1], [18]. This reveals many possibilities where data may be

stored and transmitted. As described in [12] the IP header possesses fields that are available to be used as covert channels. The total capacity of those fields is rather high (as for the steganographic technique) and may exceed 32 bits per packet and there are also fields of TCP and UDP protocols that can be also used for this purpose. Notice that this steganographic method plays an important role for VoIP communication because protocols mentioned above are present in every packet (regardless, if it is a signalling message, audio packet, or control message).

3.2 SIP/SDP Protocols Steganography

To our best knowledge little research effort has been made to use SIP messages as a covert channel. For example in [2] authors have shown how the bouncing mechanism is used for SIP messages to secretly transfer data. The interest of research in SIP/SDP protocols steganography is rather low because the signalling phase is rather short and only few messages are exchanged during this phase. In spite of this observation we want to perform an analysis of covert channels that may be utilized for SIP signalling protocol to show how much information may be transferred in VoIP signalling messages – as mentioned in Section 1 transferring even 1 bps may be considered as a threat. When call setup begins, certain SIP signalling messages are exchanged between calling parties as depicted in Fig. 1 (marked as 1). Exemplary SIP message (with SDP session description) looks as presented in Fig. 2.

```
(1)   INVITE sip:bob@biloxi.example.com SIP/2.0
(2)   Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
(3)   Max-Forwards: 70
(4)   From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
(5)   To: Bob <sip:bob@biloxi.example.com>
(6)   Call-ID: 3848276298220188511@atlanta.example.com
(7)   CSeq: 12345 INVITE
(8)   Contact: AliceM <sip:alice@client.atlanta.example.com;transport=tcp>
(9)   Content-Type: application/sdp
(10)  Content-Length: 151

(11)     v=0
(12)     o=alice 2890844526 2890844526 IN IP4 client.atlanta.example.com
(13)     s=-
(14)     c=IN IP4 192.0.2.101
(15)     t=0 0
(16)     k=clear:9123123kjnhdasdoq12e31021n2e4
(17)     m=audio 49172 RTP/AVP 0
(18)     a=rtpmap:0 PCMU/8000
```

Fig. 2. Exemplary SIP INVITE signalling message with SDP session description (bolded are fields and tokens that can be used for covert transmission)

First part of the message in Fig. 2 (signalling message header – marked with grey filling) is a SIP INVITE message (which initiates a call), the second part is an SDP session description (body of the message – marked with white filling).

3.2.1 SIP Parameters, Tokens and Fields Steganography

In SIP signalling messages there are certain tokens, like *tag* (in *From* field line 4, that forms SIP dialog identifier) or *branch* (in *Via* field line 2 that forms transaction identifier). They consist of random strings generated by user's endpoint when the connection is initiated. Also the fields: *Call-ID* (line 6, which uniquely identifies a call) and first part of *CSeq* field (line 7, initial sequence number that serves as a way to identify

and order transactions) must be generated randomly. All abovementioned fields and tokens can be straightforwardly utilized as a low-bandwidth, one direction covert channel. However, for tag token [14] it stands that "when a tag is generated (…) it must be globally unique and cryptographically random with at least 32 bits of randomness…" – that means that the inserted secret value must be chosen appropriately. For value of a branch token the situation is similar, it must begin with the characters "z9hG4bK" (called magic cookie) to ensure that previous, older SIP version's implementation would not pick such a value. The rest of branch content is implementation-defined. Next, *Call-ID* is generated by the combination of a random string and the endpoint's host name or IP address (*random_string@host_name*). Moreover *CSeq* field consists of a sequence number and a method name; sequence number value, which is chosen arbitrarily, may be used for covert transmission. The only requirement for this number is that it must be expressible as a 32-bit unsigned integer and must be less than 2^{31}. For all of the mentioned tokens and fields there are no rules inside a SIP standard (besides for *CSeq*) that specify their length, so we can increase the bandwidth of the covert channel by choosing appropriate length of those values. There is also a field *Max-Forwards* (line 3), that is used for loop detection. It may be also used as a covert channel, if the value applies to certain rules: SIP standard defines only that the initial value of *Max-Forwards* should be 70, however other values are also allowed. Eventually, we can also utilize strings in SIP messages e.g. in *Contact* field (line 8) – a string AliceM. Such string values have no direct impact on the communication itself. Fields that can be exploited in the same way as *Contact* include (more rarely, not mandatory) fields like: *Subject, Call-Info, Organization, Reply-To, Timestamp, User-Agent*, and other.

3.2.2 SIP Security Mechanisms Steganography

For SIP/SDP protocols steganography we can also utilize security mechanisms that are executed to provide security services like authentication and confidentiality for signalling messages. Especially end-to-end mechanisms are important for our purposes as they allow to transfer data directly between end users. In this article we will present how to use end-to-end SIP security mechanism S/MIME (Secure MIME) [4] to create covert channel. Fig. 3 presents how the SDP content, embedded into the SIP INVITE message, may be encrypted and signed using S/MIME. The secured parts of the message are divided from themselves using boundary value (*992d915fef419824* value in Fig. 3). It is the first value that can be utilized as a covert channel as its length and value is chosen randomly. Next, the first part between the boundary values is the *application/pkcs7-mime* binary *envelopedData* structure that encapsulates encrypted SDP session description. The second part between the boundary values is a signature of the payload (*application/pkcs7-signature*).

The second possibility for hidden communication is to use the signature bits inside the boundary values (*application/pkcs7-signature*) to transfer covert data. Therefore, we resign from signature verification (it is the cost of using this method), but instead, we gain an opportunity to send additional covert data. The amount of data that can be transferred covertly depends on what hash function is used and must be matched properly.

```
(1)    INVITE sip: bob@biloxi.example.com SIP/2.0
(2)    Via: SIP/2.0/UDP 160.85.170.139:5060;branch=z9hG4bK4129d28b8904
(3)    To: Bob <sip: bob@biloxi.example.com>
(4)    From: Alice <sip: alice@atlanta.example.com>;tag=daa21162
(5)    Call-ID: 392c3f2b568e92a8eb37d448886edd1a@160.85.170.139
(6)    CSeq: 1 INVITE
(7)    Max-Forwards: 70
(8)    Contact: <sip:alice@client.atlanta.example.com:5060>
(9)    Content-Type: multipart/signed;boundary=992d915fef419824;
(11)   micalg=sha1;protocol=application/pkcs7-signature
(12)   Content-Length: 3088
(13)   --992d915fef419824
(14)   Content-Type: application/pkcs7-mime;
(15)   smime-type=envelopeddata; name=smime.p7m
(16)   Content-Disposition: attachment;handling=required;filename=smime.p7m
(17)   Content-Transfer-Encoding: binary
(18)   <envelopedData object encapsulating encrypted SDP attachment not shown>
(19)   --992d915fef419824
(20)   Content-Type: application/pkcs7-signature; name=smime.p7s
(21)   Content-Transfer-Encoding: base64
(22)   Content-Disposition: attachment; filename=smime.p7s;
(23)   handling=required
(24)
(25)   ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfHfYT6
(26)     QpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbB9HGTrfvbnj
(27)   n8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHhHUujpfyF4
(28)           7GhIGfHfYT64VQbnj756
(29)
(30)   --992d915fef419824--
```

Fig. 3. Example of SIP INVITE signalling message secured with S/MIME mechanism

3.2.3 SDP Protocol Steganography

For SDP protocol available covert channels are similar to those presented for SIP. In Fig. 1 SDP session description is enclosed in two SIP messages (INVITE from SIP UA A to SIP UA B and in 200 OK response in the reverse direction). It is possible to use session description fields in SDP protocol, some of them do not carry important information and other are ignored (but must be present in SIP/SDP message in order to be compliant with SDP). This includes bolded fields in Fig. 2 (second part with white filling): *v* (version – field ignored by SIP), *o* (owner/creator) – there is a randomly generated session identifier (*2890844526*), and the name of the owner/creator, *s* (session name – field ignored by SIP), *t* (time session is active – field ignored by SIP) and *k* (potential encryption key if the secure communication is used).

To summarize: for SIP/SDP protocols steganography creation of covert channels is possible because in specifications of these protocols there are no strict rules how to generate tokens and parameters and what is their desired length.

3.2.4 Other SIP/SDP Protocol Steganography Possibilities

For both protocols other steganographic methods may be utilized. For example, like in [8] we can use nonprintable characters (like spaces [SP] or tabs [HT]) or their sequences after the SIP header fields. Described situation is presented in Fig. 4.

The next method from [8] exploits the fact that the order of headers in the SIP/SDP message depends on implementation, thus reordering of headers is possible as a mean to covertly send data. If we consider exemplary signalling message form Fig. 4, if field *Call-ID* is after *CSeq* it can denote that binary "1" was sent, while if the order is reversed the value is "0". The last method exploits case modification (upper and lower cases), because names of the field are case-insensitive (so e.g. ***FROM*** header means "1" while ***to*** header "0"), but this technique is rather easy to uncover.

While call lasts, some signaling messages may also be exchanged to influence certain parameters of the session (e.g. codec). Bandwidth and steganographic techniques for SIP/SDP remain the same as described in the signalling phase. Moreover, during the conversation phase, we can also utilize SIP message like OPTIONS, which is used for sharing capabilities of the endpoints, e.g. to be able to support additional services. Such messages may be intentionally invoked (to some extent) to increase the covert channel bandwidth for these steganographic techniques. It is also worth noting that the SIP signalling messages are exchanged after the conversation phase is finished (marked on Fig. 1 with 3).

```
(1)    INVITE[SP]sip:bob@biloxi.example.com[SP]SIP/2.0[SP][SP][HT][SP][HT]
(2)    From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl[HT][SP][HT]
(3)    To: Bob <sip:bob@biloxi.example.com>[HT][SP][HT][HT][SP][HT][SP][SP]
(4)    Call-ID: 3848276298220188511@atlanta.example.com[SP][HT][SP][SP]
(5)    CSeq: 12345 INVITE
```

Fig. 4. Example of usage of nonprintable characters as a covert channel for SIP

4 Evaluation of Total Covert Data Transferred in VoIP Signalling

Let us consider a scenario from Fig. 1 and based on that we will try to estimate how much information one may hide in signalling messages during the VoIP call. From Fig. 1 we can conclude that about 5 signalling messages may be sent in one direction between end users (two during initial signalling phase, two during the conversation e.g. OPTIONS message and one to end the call). Moreover, let us assume that two of these messages will carry also SDP body and that:

- IP/TCP/UDP protocols steganography provides covert transmission at the rate of 16 bits/message,
- SIP parameters, tokens and fields steganography gives about 60 characters for the first SIP message that is total of 480 bits (usage of initial values),
- SIP security mechanisms steganography which provides 160 bits per SIP message,
- SDP protocol steganography that gives 60 characters for each SDP body (we assumed two SDP bodies) that result in total of 960 bits,
- Other SIP/SDP protocol steganography possibilities we assumed about 8 bits/message.

For the considered scenario from Fig. 1 and equation 1 we can easily calculate that B_T = 2.36 kbits. Therefore, we see that even for only five SIP messages exchanged during VoIP call we can covertly transfer, in one direction, more than two thousand bits. That is why for high security requirements networks we may consider SIP steganography as a potential threat to information security.

5 Conclusions

In this paper we have described existing and introduced new steganographic methods for SIP/SDP protocols. All new solutions are based on network steganography as they

utilized free or unused fields in abovementioned protocols. Total amount of information that may be transferred with use of proposed solutions is more than 2000 bits in one direction for each performed VoIP call. Although, this amount of information may be considered as low (as not many SIP/SDP messages are exchanged between end users), sometimes even this amount of data may be sufficient to cause serious information leakage.

References

1. Ahsan, K., Kundur, D.: Practical Data Hiding in TCP/IP. In: Proc. of Workshop on Multimedia Security at ACM Multimedia 2002, Juan-les-Pins, France (2002)
2. Bidou, R., Raynal, F.: Covert channels, http://www.radware.com/WorkArea/downloadasset.aspx?id=3928
3. Cuervo, F., Greene, N., Rayhan, A., Huitema, C., Rosen, B., Segers, J.: Megaco Protocol Version 1.0. IETF, RFC 3015 (2000)
4. Galvin, J., Murphy, S., Crocker, S., Freed, N.: Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted. IETF, RFC 1847 (1995)
5. Handley, M., Jacobson, V., Perkins, C.: SDP: Session Description Protocol. IETF, RFC 4566 (2006)
6. ITU-T Recommendation H.323: Packet-based Multimedia Communications Systems Ver. 6. ITU (2006)
7. Johnston, A., Donovan, S., Sparks, R., Cunningham, C., Summers, K.: Session Initiation Protocol (SIP) Basic Call Flow Examples. IETF, RFC 3665 (2003)
8. Kwecka, Z.: Application Layer Covert Channel Analysis and Detection. Napier University Edinburgh, Technical Report (2006), http://www.buchananweb.co.uk/zk.pdf
9. Lampson, B.: A Note on the Confinement Problem. Comm. ACM 16(10), 613–615 (1973)
10. Mazurczyk, W., Kotulski, Z.: New Security and Control Protocol for VoIP Based on Steganography and Digital Watermarking. Annales UMCS, Informatica, AI 5, 417–426 (2006) ISNN 1732-1360
11. Mazurczyk, W., Kotulski, Z.: New VoIP Traffic Security Scheme with Digital Watermarking. In: Górski, J. (ed.) SAFECOMP 2006. LNCS, vol. 4166, pp. 170–181. Springer, Heidelberg (2006)
12. Murdoch, S.J., Lewis, S.: Embedding Covert Channels into TCP/IP. In: Barni, M., Herrera-Joancomartí, J., Katzenbeisser, S., Pérez-González, F. (eds.) IH 2005. LNCS, vol. 3727, pp. 247–261. Springer, Heidelberg (2005)
13. Petitcolas, F., Anderson, R., Kuhn, M.: Information Hiding – A Survey. IEEE Special Issue on Protection of Multimedia Content (1999)
14. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A.: SIP: Session Initiation Protocol. IETF, RFC 3261 (2002)
15. Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V.: RTP: A Transport Protocol for Real-Time Applications. IETF, RFC 3550 (2003)
16. US Department of Defense – Trusted Computer System Evaluation Criteria. DOD 5200.28-STD. The Orange Book (1985)
17. Wikipedia, http://en.wikipedia.org/wiki/Covert_channel
18. Zander, S., Armitage, G., Branch, P.: A Survey of Covert Channels and Countermeasures in Computer Network Protocols. IEEE Communications Surveys & Tutorials, 3rd Quarter 2007 9(3), 44–57 (2007)

Multimed Tools Appl
DOI 10.1007/s11042-010-0532-0

What are suspicious VoIP delays?

Wojciech Mazurczyk · Krzysztof Cabaj · Krzysztof Szczypiorski

Abstract Voice over IP (VoIP) is unquestionably the most popular real-time service in IP networks today. Recent studies have shown that it is also a suitable carrier for information hiding. Hidden communication may pose security concerns as it can lead to confidential information leakage. In VoIP, RTP (Real-time Transport Protocol) in particular, which provides the means for the successful transport of voice packets through IP networks, is suitable for steganographic purposes. It is characterised by a high packet rate compared to other protocols used in IP telephony, resulting in a potentially high steganographic bandwidth. The modification of an RTP packet stream provides many opportunities for hidden communication as the packets may be delayed, reordered or intentionally lost. In this paper, to enable the detection of steganographic exchanges in VoIP, we examined real RTP traffic traces to answer the questions, what do the "normal" delays in RTP packet streams look like? and, is it possible to detect the use of known RTP steganographic methods based on this knowledge?

Keywords IP telephony · VoIP delays · LACK · Information hiding · Network steganography

1 Introduction

Steganography has been used for ages, dating back as far as ancient Greece [19]. Steganographic methods allow for hiding the very existence of the communication, so a third-party observer will not suspect anything if they are unaware of the steganographic exchange. Steganography encompasses information hiding techniques that embed a secret

W. Mazurczyk (✉) · K. Cabaj · K. Szczypiorski
Faculty of Electronics and Information, Warsaw University of Technology, Technology, 15/19 Nowowiejska Str., 00-665 Warsaw, Poland
e-mail: W.Mazurczyk@elka.pw.edu.pl

K. Cabaj
e-mail: K.Cabaj@elka.pw.edu.pl

K. Szczypiorski
e-mail: K.Szczypiorski@elka.pw.edu.pl

Published online: 22 April 2010

message (steganogram) into the carrier. The carrier is suitable for steganographic purposes if it fulfils two conditions: it is commonly used and the carrier modification caused by the embedding of the steganogram must not be "noticeable" to anyone. The form of the carrier has evolved over time—historical carriers were wax tablets, human skin or letters [19]—now it is instead a digital picture, audio or text.

Recently, a new type of steganography was identified, called *network steganography*. This includes information hiding techniques that utilise, as a carrier, data units and/or their exchange in a telecommunication network. Network steganography can pose a threat to network security, as the current security systems and mechanisms do not provide sufficient countermeasures and are in fact useless against this type of threat. Using steganography for malicious purposes can lead, for example, to confidential information leakage or serve as tools for the distribution of worms and viruses in planning and conducting DDoS (Distributed Denial of Service) attacks [21]. Thus, it is important to answer the question, what real impact may steganographic methods have on network security? The answer may be found through careful evaluation of a particular methods' potential steganographic bandwidth and its possibilities for detection (steganalysis).

VoIP (Voice over IP) is a real-time service which enables users to make phone calls through data networks that use an IP protocol. The popularity of this technology has caused a continuous rise in the volume of VoIP traffic. Thus, it may be increasingly targeted for steganographic purposes, as stated by Lubacz, Mazurczyk and Szczypiorski in [14], and it is therefore important to develop detection methods. To achieve this goal, we must first find an answer to the question, what does an anomaly caused by the use of steganography during a VoIP call look like?

RTP (Real-time Transport Protocol) [22] is the most promising carrier of steganograms in VoIP. RTP provides end-to-end network transport functions suitable for applications transmitting real-time audio. RTP is usually used together with UDP (or rarely TCP) for the transport of digital voice streams. During the conversation phase of the call audio (RTP) streams flow bidirectionally between a caller and a recipient. The rate at which RTP packets flow depends on the codec used, e.g., in the G.711 codec [9], each RTP packet carries 20 ms of voice using 160 bytes; in this case the RTP packet flow rate is 50 packets per second. Thus, even by hiding 1 bit in every RTP packet we gain the quite high steganographic bandwidth of 50 bit/s. In effect, this would allow the user to send about 5 kB of data during a typical VoIP call.

As the authors stated in [16], steganalysis methods must be developed for RTP transmission to enable the detection and/or elimination of hidden communication. To achieve this goal, a steganographic method for inspecting RTP transmission must be developed. Two broad groups of information hiding techniques exist that may affect RTP; the first group is based on modifying the RTP packet header and/or payload, while the second affects the RTP packet stream by modifying the time relation between them. In this study, we focus on the second group of steganographic solutions, because the first is easy to detect and eliminate. Methods for modifying an RTP stream to transmit bits of a steganogram can:

- Affect the sequence order of RTP packets [12] by assigning an agreed-upon order of packets during a predetermined period of time. For example, sending packets in ascending order could indicate a binary one, and descending order a binary zero
- Use different sending rates for the RTP stream [7]—in a simple case, one (the original) rate denotes a binary one, a second rate, achieved, e.g., by delaying RTP packets, means a binary zero

- Modify inter-packet delay [2]—e.g., where predetermined delays between two subsequent RTP packets are used to send single steganogram bit
- Introduce intentional losses [23] by skipping one sequence number while generating RTP packets. Detecting so called "phantom" loss during a predetermined period of time means sending one bit of the steganogram
- Use intentionally delayed packets (by the transmitter) from the RTP stream to carry a steganogram. An example of such a method is LACK (Lost Audio Packets Steganography) [16]. If the delay of the chosen packets at the receiver is considered excessive, the packets are discarded by a receiver not aware of the steganographic procedure. The payload of the intentionally delayed packets is used to transmit secret information to receivers aware of the procedure so no extra packets are generated; for unaware receivers the hidden data is "invisible". More detailed LACK description may be found in [16].

Steganographic methods described above have one common feature—all of them modify delays of the RTP packets. Thus, to evaluate the impact that they may have on network security, real RTP packet delays during VoIP calls should be investigated.

For VoIP, network delays and packets losses have already been thoroughly researched, e.g., in [3], [15] and [5], but not yet in the steganographic context. Moreover, the existing research has focused on measuring overall packet delay and losses rather than their detailed characterisation. Consequently, the main objective of this study was to describe what can happen to packets in an RTP stream while traversing the network based on real VoIP traffic captures. Our research focused on RTP packet delays and all scenarios that may lead to the loss of RTP packets (physically or by the receiver, e.g., jitter buffer). Using this knowledge, we were able to characterise delays that can be introduced into the network and to evaluate the threat which may be posed by steganographic methods that utilise RTP by estimating their steganographic bandwidths. This information will be also needed to develop effective countermeasures.

Thus, the goals of this study were to:

- Characterise the delays and losses for VoIP over the Internet, based on the experiment conducted for an average VoIP call (average duration, connection path length, typical codec, loss concealment method and jitter buffer sizes)
- Identify all scenarios for RTP packet losses, including physical losses and losses caused by jitter buffer (e.g., late packets dropped and buffer overflow), and present the corresponding results
- Evaluate the feasibility of RTP steganographic methods based on real VoIP traffic

The structure of the paper is as follows. Section 2 briefly describes the basics of RTP and the jitter buffer algorithms used in VoIP. In Section 3, the experimental results for VoIP delays are presented and analysed. Section 4 discusses how the knowledge of real RTP packet delays affects VoIP steganographic methods in use; Section 5 concludes our work.

2 RTP (real-time transport protocol) packets and VoIP jitter buffers

As mentioned in the introduction, RTP is a crucial protocol for VoIP during the transport of voice packets through IP networks. Usually, RTP packets are generated by the transmitter at a fixed rate, e.g., every 20 ms in the G.711 codec, and they are expected at the receiver at the same rate. However, while traversing the network voice packets may be subjected to

such impairments as delay, loss or jitter. Thus, the delays in the received packets can be different from the transmitted ones. This is why there is a need for a receiving buffer, called a jitter buffer. The size of the jitter buffer is crucial for limiting the so-called *mouth-to-ear* delay (which should not exceed 150 ms) and determines the quality of the conversation. If the buffer is too large, the *mouth-to-ear* delay is increased, causing a degradation of call quality. However, if the buffer is too small, overall packet losses are increased due to jitter buffer drops, which can also negatively affect call quality. Thus, the sizing of the jitter buffer always involves a trade-off between increasing the overall delay and minimising losses. Typical jitter buffers for VoIP are sized in the range of 40–80 ms.

Another important fact is associated with the type of jitter buffer used. There are two types of jitter buffers: fixed or adaptive. A fixed buffer has a constant size while an adaptive buffer changes size during the call or between subsequent calls based on information about delays and losses introduced by the network. Adaptive buffers change size in a defined range (e.g., from 40 to 100 ms). Various algorithms for jitter buffering exist and are described, e.g., in [20], [25] or [18]. However, the real problem is that only a few of these proposed algorithms are implemented in practice, and as Wu et al. [26] stated, most popular VoIP applications use fixed-size buffers or adapt to network conditions, but not optimally. In this paper we chose to simulate a simple fixed buffer as specified in [5]. This simple jitter buffer allows for the visualisation of problems that may occur in real RTP streams, and, as mentioned, such fixed-size buffers are still commonly implemented. The jitter buffer operates as follows: after the initiation of a call, before the receiver begins to play back the speech samples to the recipient it continues buffering the RTP packets until the buffer is filled to half capacity. Then, when the next packet above half capacity is received the speech samples are played back.

The next most important mechanism used to limit quality degradation due to packet loss consists of PLC (Packet Loss Concealment) methods. In the simplest scenario, these utilise repetition of the last received packet to substitute for a missing one [9], but more complex algorithms have also been developed [13]. In our implementation, to help preserve voice quality repetition of the last successfully received packet was used to conceal a physically lost or dropped one.

Despite the jitter buffer algorithm and PLC mechanism used in VoIP, packets may be lost; a packet is considered lost if:

- *It is discarded in the network* (Fig. 1, point b)—in this case it never reaches the receiver. Such a situation may be caused, e.g., by buffer overflow in some intermediate device caused by a bottleneck within a network. We refer to such losses as *physical losses.*
- *It is dropped by the jitter buffer* (Fig. 1, point c)—when an RTP packet is excessively delayed due to network latency it reaches the receiver but is useless as it cannot be used for voice reconstruction; thus, it is discarded and counted as lost. Moreover, due to so-called delay spikes, the jitter buffer, in addition to dropping late packets (drops caused by *buffer underflow*) may also drop subsequent RTP packets because they may all arrive simultaneously and the size of the jitter buffer may be insufficient to store them all (*buffer overflow*).

VoIP statistics regarding losses should distinguish between physical losses and losses caused by jitter buffer. Moreover, we need to know what realistic VoIP inter-packet delays are and what they look like. Do the delays and losses happen singly or in series, and if so, can these series be characterised? Another important consideration is whether any method may utilise an intentional reordering of RTP packets—is this realistic for IP telephony in today's Internet? We address these and other questions in the next sections of this paper.

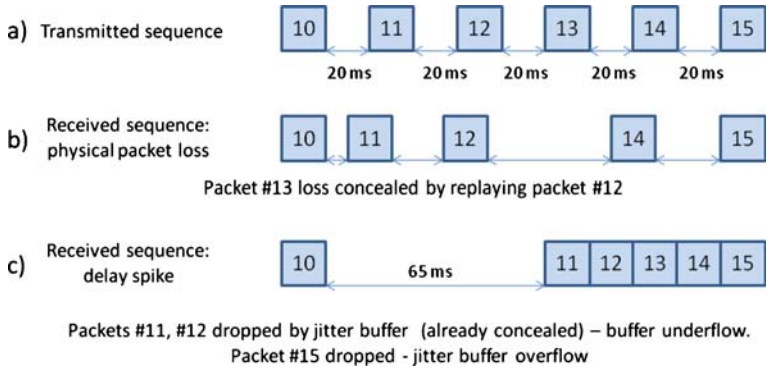


Fig. 1 Packet losses in VoIP

3 What can actually happen to RTP packets while traversing a network?

For a practical evaluation of the feasibility of steganographic methods utilising RTP as specified in the introduction, we assumed that the VoIP endpoints exchanging the RTP streams are also the sender and the receiver of the steganogram (but this is not the only possible scenario, as stated by Mazurczyk and Szczypiorski in [16]).

To evaluate the real VoIP delays of RTP packets, an experiment was conducted. VoIP calls were established from Warsaw, Poland to Cambridge, UK (see Fig. 2) through the Internet using the very popular free SIP-based softphone *X-lite* [28] (ver. 3.0 build 56125) and SIP proxy server (OnDo Brekeke SIP Server ver. 2.3.7.4) which was located in Warsaw. The distance between the cities is ~1,800 km. One hundred calls were captured using *Wireshark* (ver. 1.3.3) [27] between 27 October and 4 November, 2009 during working hours, which resulted in total number of 2,825,076 packets transmitted. The communication path between the cities represents typical, average Internet connection path of about 16 hops [1, 6, 29]. Audio was coded with the ITU-T G.711 A-law PCM codec (20 ms of voice per packet, 160 bytes of payload). The average call duration for the

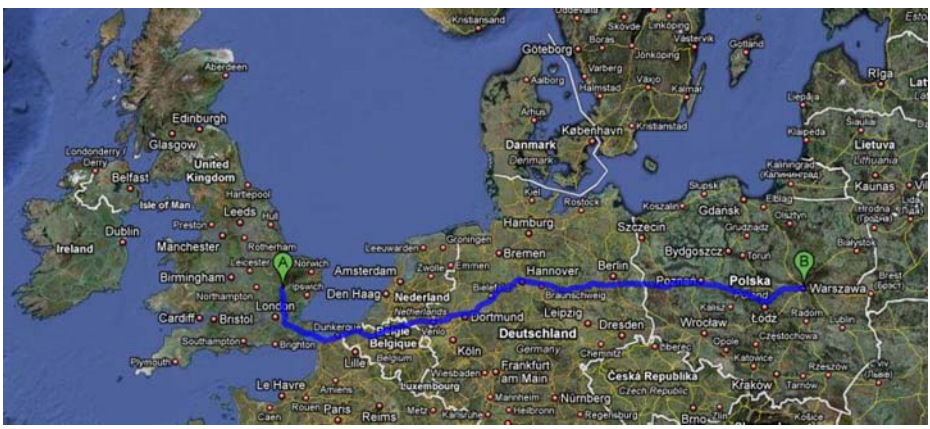


Fig. 2 VoIP experimental evaluation scenario—calls from Cambridge, UK **a** to Warsaw, Poland **b** (http://maps.google.com)

experiment was chosen based on the average duration of calls using Skype [24] and other VoIP service providers. In [8], the results obtained show that the average call duration on Skype was about 13 min, while for other VoIP providers it is typically between 7 and 11 min. Thus, we chose an experimental call duration of 9 min.

In the experiment, we developed custom software to analyse delays and losses occurring in the RTP stream of the captured call traces and to simulate different fixed buffer sizes to be able to evaluate the relationships between RTP packet delays, losses and jitter buffer size. The jitter buffer sizes used and the number of buffered packets after which the playback of the voice samples began are specified in Table 1. The description of the jitter buffer algorithm was described in detail in the previous section. For each experimental call we measured packets dropped by the jitter buffer, delayed packets and physical losses.

For each call quality was assessed using the ITU-T E-model [11], which is a quality objective assessment method for transmission planning. The E-model expresses call quality as an R factor which ranges in value from 0 (worst quality) to 100 (best quality). For real VoIP traffic, Cole and Rosenbluth [5] proposed a simplified formula for R calculation based on VoIP performance monitoring, which takes into account only impairments caused by losses and delays, as follows:

$$R = 94.2 - I_d - I_{ef} \tag{1}$$

where I_d denotes impairments caused by delays and I_{ef} impairments caused by losses. I_d was calculated based on mouth-to-ear delay (d) as proposed in [5]:

$$I_d = 0.024 + 0.11 \cdot (d - 177.3)H(d - 177.3) \tag{2}$$

where $H(x)$ is the Heaviside (or step) function defined as:

$$H(x) = \begin{cases} 0 \ if \times < 0 \\ 1 \ if \times \geq 0 \end{cases} \tag{3}$$

I_{ef} was calculated based on an equation given by [5] that was derived explicitly for the G.711 codec, additionally concerning random losses:

$$I_{ef} = 30 \cdot \ln(1 + 15p_L) \tag{4}$$

where p_L denotes the probability of RTP packet loss.

Based on the E-model and results from our experiments, an R factor was obtained. This was then converted into an MOS (Mean Opinion Score) score ranging from 1 (bad quality) to 5 (good quality) [10], which is typically used for expressing the quality of VoIP calls, using the known formula:

$$MOS = 1 + 0.035 \cdot R + 7 \cdot 10^{-6} \cdot R(R - 60)(100 - R) \tag{5}$$

For the experimental scenario and assumptions presented above the following results were obtained.

Call quality results in form of CDF (Cumulative Distribution Function) of MOS scores are presented in Fig. 3. It is often assumed that an MOS score equal to or greater than 3.6 is

Table 1 Jitter buffer characteristic parameters

Jitter buffer size [ms]	20	40	60	80	100	120
No. of initially buffered packets	1	1	2	2	3	3
Playback starts after receiving [packets]	2	2	3	3	4	4

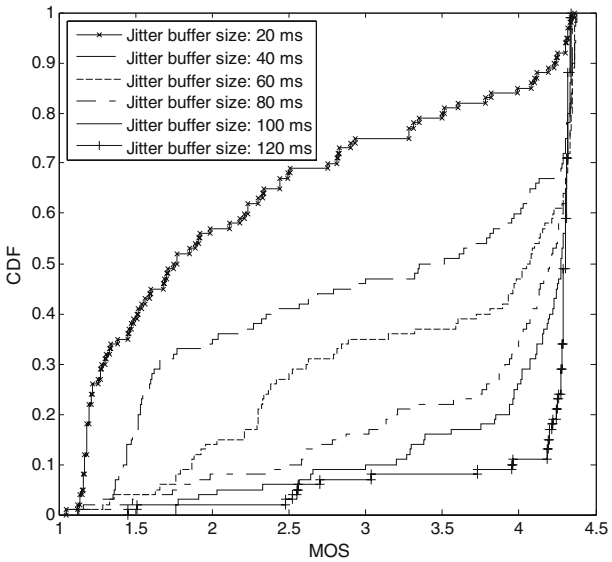


Fig. 3 CDF of MOS scores for different jitter buffer sizes

considered to be of comparable quality to traditional PSTN (Public Switched Telephone Network) calls [3]. By this standard, the quality of the experimental calls using the 80-, 100- and 120-ms buffer sizes can be judged as good, as less than ~20% of these calls were of a quality lower than 3.6. For the 60-ms jitter buffer about 30% of the calls were of lower quality than 3.6, while for the 20-ms jitter buffer it was about 75% of all calls.

It was of interest to plot the cumulative distributions of the two most important impairments in the experimental VoIP data: delay and loss. The results for physical packet losses are presented in Fig. 4.

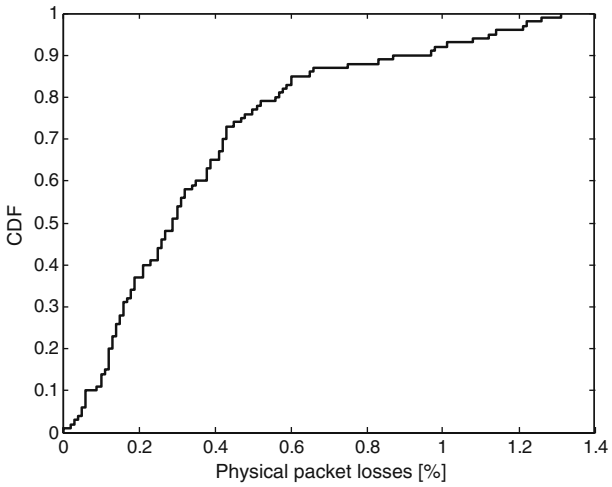


Fig. 4 CDF of physical packet losses for the experimental data

The above figure illustrates that losses caused by the network (packets that never reach the receiver) did not exceeded 0.5% of all RTP packets in more than 80% of the calls. These results somehow confirms earlier research in that area. For example, Borella et al. [4] analysed a month of Internet packet loss statistics for speech transmission and their findings are that physical packet losses for the three paths, all in the U.S., ranged from 0.4% to 3.5%.

What we want to explore next is the RTP inter-packet delays. Here, we wanted to know how many packets arrived late if we assumed a certain delay threshold. For our experiment, we chose threshold values from 20 to 100 ms with a step of 20 ms. The results obtained are presented in Fig. 5.

From the figures above it can be seen that there was a great difference in the number of delayed RTP packets between a packet delay of 20 ms and the remaining delay values. This was caused by the packet generation time interval in the transmitter—packets were sent each 20 ms. Thus, if there was any delay, even the slightest, in a packet's reception introduced by the network or by clock skew it was counted as delayed. It is obvious that the larger the assumed delay threshold the lower the number of delayed packets. For example, about 30% of the calls experienced 2% or more packets delayed more than 40 ms, while only about 5% of all calls had about 1% or more packets delayed by more than 80 ms.

Next, we compared how many of the delayed packets presented in the figure above resulted in losses caused by jitter buffer drop. First, in Fig. 6 we present packet drops caused by jitter buffer. As expected, with an increase in the size of the jitter buffer the number of packets dropped decreased. For example, more than 40% of the packets were dropped in 35% of the calls using a 40-ms jitter buffer, whereas it was about 5% of all VoIP calls for the 80-ms jitter buffer.

The results concerning the influence that delayed RTP packets have on packet losses caused by a too-small jitter buffer are presented in Fig. 7 (for 40-ms and 80-ms jitter buffers).

With the 40-ms jitter buffer almost 50% of all calls experienced ~10% or more packet drop, while ~10% or more of the packets were delayed by more than 40 ms for only 10% of the calls. For 80-ms jitter buffer, ~ 10% or more buffer drops were observed for about 25% of the calls, with only a small number of the packets delayed by more than 80 ms. Thus, the larger jitter buffer yielded a greater number of delayed packets that were not lost and could be used for voice reconstruction. Simultaneously, a larger buffer adds more delay to the conversation, which may affect call quality if *mouth-to-ear* delay exceeds 150 ms. Total

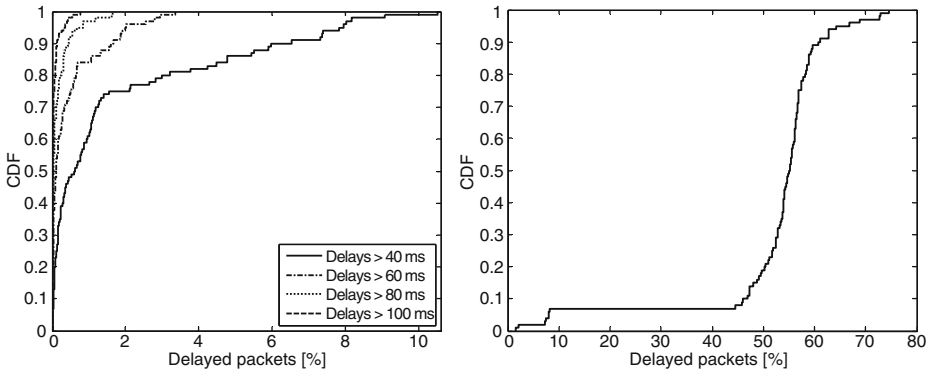


Fig. 5 CDF of RTP packet delays for different delays thresholds – 40, 60, 80, 100 ms (*left*), 20 ms (*right*)

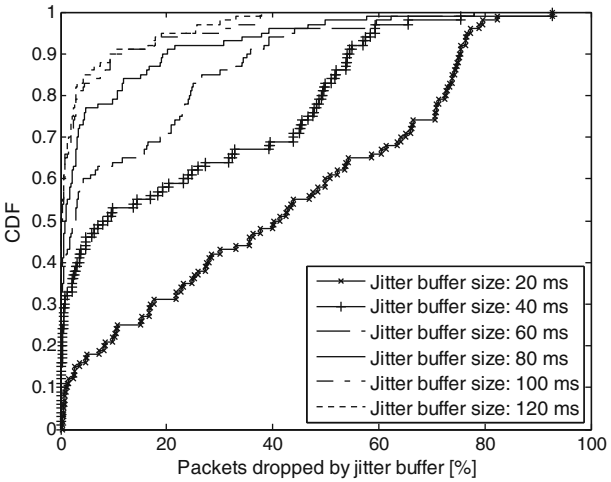


Fig. 6 CDF of RTP packet drops by jitter buffer for different buffer sizes

losses, including packets dropped by jitter buffer and physical losses, are presented in Fig. 8.

For the G.711 codec, which has PLC (Packet Loss Concealment) functionality, the maximum tolerable packet loss is 5% [17]. Thus, for our experimental data it would be best to use an 80-ms or larger jitter buffer to preserve conversation quality as for this size almost 80% of the calls experienced losses lower than 5%.

Next, we compared physical losses and losses caused by jitter buffer (Fig. 9).

The fraction of the physical losses was so small that the main factors determining total losses were those associated with buffer drops. As stated in the introduction, there are two types of jitter buffer losses: drops caused by jitter buffer overflow and those which are caused by late packets. The buffer overflows when it is full and the next received packet cannot be stored, thus it must be discarded (for logging purposes we noted this event as a D1 drop). The second type of jitter buffer drop is caused by RTP packets which are received too late, so they are not present at the receiver when they should be played to the recipient. In this case the PLC mechanism fills the gaps by replaying the last successfully received packet. When these packets finally reach the receiver, they cannot be used for

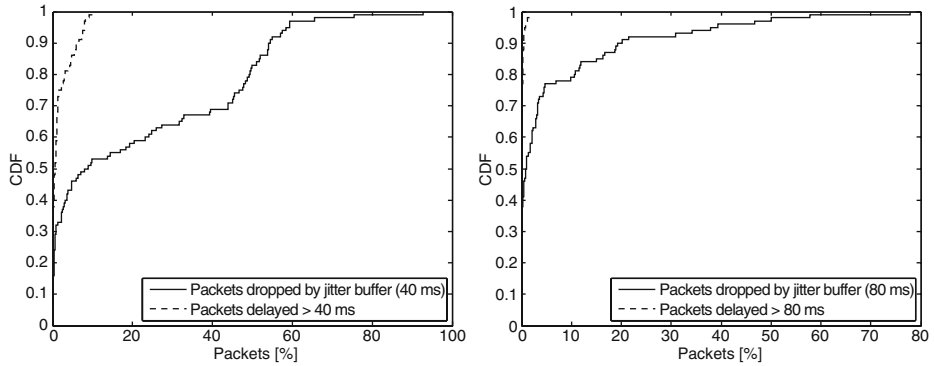


Fig. 7 Relationship between delayed and dropped packets for jitter buffers of 40 ms (*left*) and 80 ms (*right*)

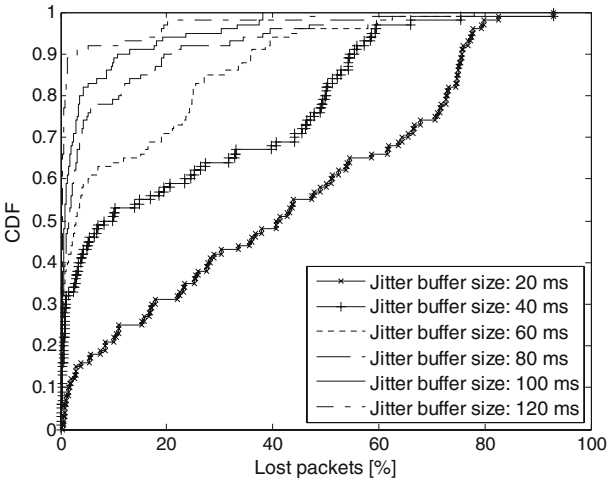


Fig. 8 Total packets lost due to physical losses and drops by the jitter buffer

voice reconstruction, and they are dropped as a result (for logging purposes we noted this event as a D2 drop). In both cases described, the dropped RTP packets physically reach the receiver but they are discarded by the jitter buffer and never used for voice reconstruction.

Late packet drops (D2) are usually caused by delay spikes, i.e., at some point in the connection there is a great increase in inter-packet delay which results in buffer underflow, and the late packets are not needed for voice reconstruction (because they have already been concealed) and are dropped. However, jitter buffer overflows (D1) happened more frequently for calls that experienced the following event: at the beginning of the call a burst of RTP packets came nearly simultaneously (i.e., the inter-packet delay was about zero). Such an event, especially for small jitter buffers, results in buffer overflow and influences the rest of the conversation as well by introducing subsequent drops whenever the inter-packet delay differs, even slightly, from the RTP packet generation time (20 ms).

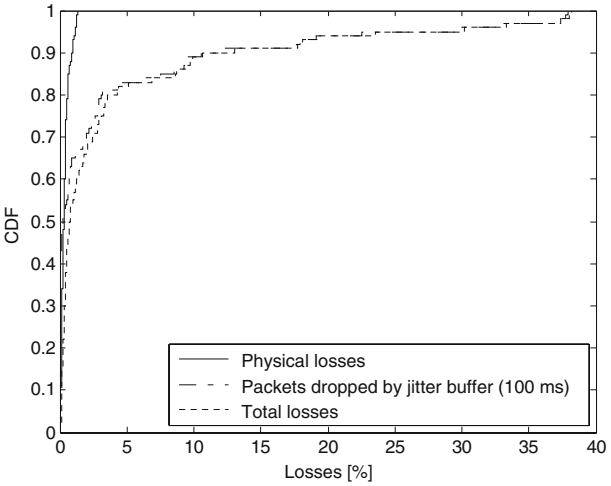


Fig. 9 Relationship between physical losses and losses caused by jitter buffer (100 ms)

The results from the application of the software tool developed for RTP stream analysis revealed two often-observed sequences of events that produced high levels of packet drop. Both situations are presented in Table 2, which contains sample sections of two logs from the developed research software. Here, only the sequence number of the RTP packets (the number after the *seq* string) and the event type (the string after the colon) are considered in the given excerpts; the two other numbers represent analysed RTP packet numbers from the beginning of the given RTP stream and from the beginning of the recorded *Wireshark* file (these values were used for testing and debugging purposes).

The left column in Table 2 presents drops due to buffer overflows (D1). Here, a received RTP packet is dropped due to the full jitter buffer. In effect, the PLC mechanism had to subsequently reconstruct the packet dropped earlier, leading to an R event. This situation can be observed, for example, in the first two lines, concerning the RTP packet with sequence number 8101.

The right column of Table 2 presents the second type of jitter buffer drops (D2). In contrast to the previous situation, in this case drops are associated with buffer underflow events (U). Because the jitter buffer is empty the PLC mechanism had to reconstruct a packet, and as a result a U event appears in log. Later, when the original late packet arrived, due to the previous reconstruction it had to be dropped. This sequence, concerning the packet with sequence number 5331, can be observed in first and last lines.

Not surprisingly, for both types of jitter buffer drops increasing the jitter buffer size caused a decrease in the total buffer losses. It should be also noted that drops caused by buffer overflows were more rapidly compensated for with increased buffer size (see Figs. 10 and 11). For smaller jitter buffer sizes, i.e., from 20 to 60 ms, losses due to jitter buffer overflows dominated, while for buffers larger than 60 ms losses caused by late packets took precedence (Fig. 12).

Because the jitter buffers sized at 20, 40 and 60, 80 and 100, and 120 ms start playing the voice samples after buffering the same number of RTP packets (1, 2 and 3, respectively), the curves representing losses due to late packet drops for these buffers were the same (hence, overlapping curves are not presented in Fig. 13).

We also observed that a large number of the experimental calls followed a pattern of only a single type of jitter buffer drop, i.e., if there were a lot of drops caused by buffer overflows, the level of late packet drops for the same call was rather low and *vice versa*.

Finally, it must be emphasised that during the performed experimental calls there *were no reordered RTP packets*. This means that while delays, even high delays, are possible for the RTP packets they do not lead to their reordering.

Table 2 Sample logs from the research software developed for RTP stream analysis (U refers to a buffer underflow event and R represents invocation of the PLC mechanism; the sequence number from RTP header is given after the *seq* string)

900[1861,seq 8101],10: D1	37[101,seq 5331],20: U
905[1871,seq 8101],10: R	37[101,seq 5328],20: D2
905[1871,seq 8106],10: D1	38[103,seq 5332],20: U
910[1881,seq 8106],10: R	38[103,seq 5329],20: D2
910[1881,seq 8111],10: D1	39[105,seq 5333],20: U
915[1891,seq 8111],10: R	39[105,seq 5330],20: D2
915[1891,seq 8116],10: D1	40[107,seq 5334],20: U
920[1901,seq 8116],10: R	40[107,seq 5331],20: D2

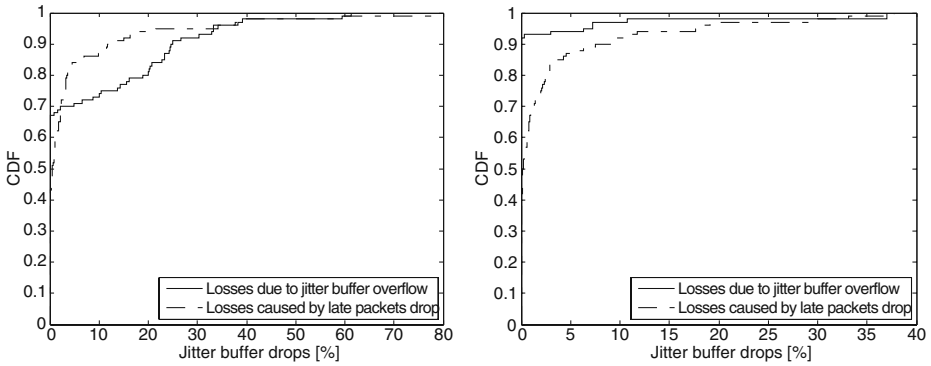


Fig. 10 Comparison of different types of jitter buffer drops for 60-ms (*left*) and 100-ms (*right*) jitter buffers

4 Feasibility of RTP steganographic methods based on real VoIP traffic

First, let us consider steganographic methods that affect the sequence of RTP packets. For a sequence of n RTP packets, the potential number of steganogram bits is $log_2(n!)$; thus, the steganographic bandwidth (S_B) may be expressed as:

$$S_B = \frac{i \cdot \log_2 n!}{T} [bits/s] \tag{6}$$

where T denotes VoIP call duration (in seconds) and i is the number of time intervals in which a steganogram will be detected. For example, if we assume that we try to send a steganogram using a sequence of 10 subsequent RTP packets (for G.711 it is interval of 0.2 s, so i=2,700), for the same call duration as the experimental ones (540 s) we achieve a steganographic bandwidth of about 100 bits/s. However, it must be noted that, as mentioned above, there were *no reordered RTP packets*, so applying such a method will be trivial to detect. Moreover, affecting the sequence of the RTP packets may lead to a deterioration of

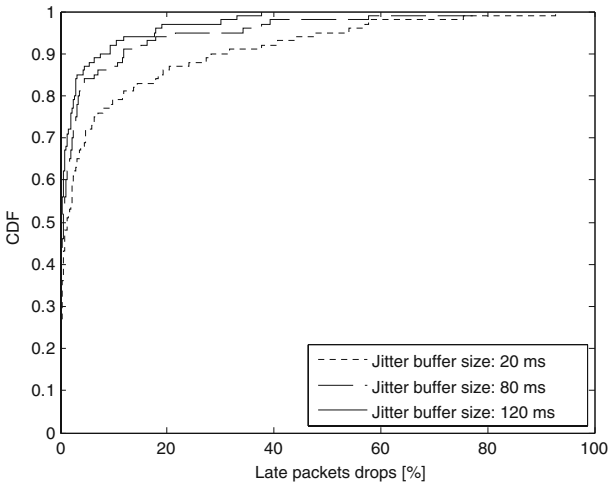


Fig. 11 Late packet drops by jitter buffers (20-, 80- and 120-ms)

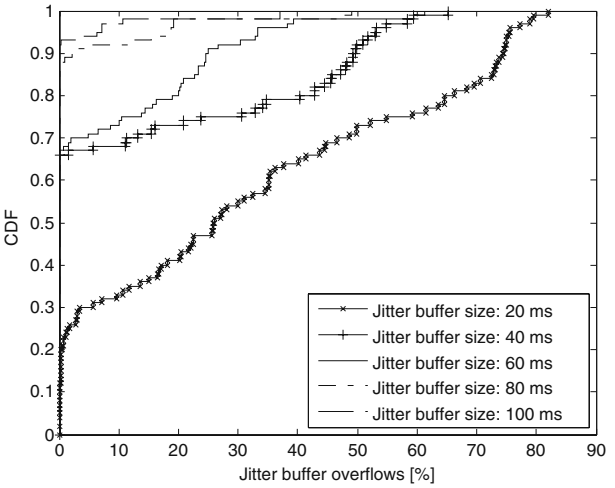


Fig. 12 Packets dropped due to jitter buffer overflow

conversation quality as the jitter buffer may be unable to compensate for intentional packet reordering.

Next, let us consider steganographic methods that utilise different RTP packet-sending rates. In the simplest case, the original generation rate of the RTP packets denotes sending a binary one and second rate is achieved, e.g., by delaying RTP packets, which means sending a binary zero. If h different methods of sending RTP packets are used it is possible to send $log_2 h$ bits of a steganogram. This may be expressed as:

$$S_B = \frac{i \cdot \log_2 h}{T} [bits/s] \tag{7}$$

For example, if $h=2$ and we assume a VoIP call duration of 9 min and a steganogram is sent each second then we achieve a steganographic bandwidth of about 1 bit/s. A similar method is based on modifying RTP inter-packet delay, where predetermined delays between two subsequent RTP packets are used to send one steganogram bit.

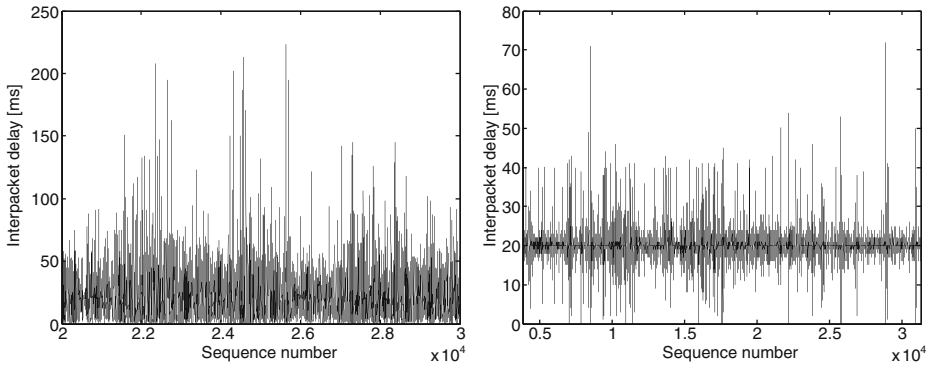


Fig. 13 High (*left*) and low (*right*) inter-packet delays of two selected experimental calls

For these two methods let us consider Fig. 13, showing inter-packet delay diagrams for two experimental calls that were chosen based on different delay statistics. The left diagram in Fig. 13 presents an experimental call that experienced high inter-packet delays during the call and the right diagram shows the opposite situation.

Note that the difference in inter-packet delay for these two diagrams is quite high and the delay spikes are distributed rather randomly. If we now assume a low inter-packet delay and if we apply the steganographic method utilising two different rates of RTP packet generation, the resulting diagram, analogous to those presented above, will be similar to that presented in Fig. 14.

If the RTP packet generation rate is intentionally modified in order to send a steganogram, a certain regularity in inter-packet delays may be observed. Thus, the detection of such method is easy. Moreover, if the RTP packets experience high inter-packet delays (Fig. 13, left diagram) then reception of the steganogram bits may be difficult. The same argument applies to the steganographic method that modifies inter-packet delays.

Let us focus on intentional losses by skipping one sequence number while generating RTP packets. Detecting such so-called "phantom" loss during a predetermined time period means sending one bit of the steganogram. As in the case of the method which modified inter-packet delays, the reception of the steganogram bits may be disrupted due to losses introduced by the network. For the experimental data the average packet loss was 0.37% (about 100 packets), which would make detection of steganogram bits difficult.

Moreover, from a practical point of view such a method is characterised by a rather low steganographic bandwidth, which may be expressed as:

$$S_B = \frac{i}{T}[bits/s] \tag{8}$$

For example, if we assume that intentional losses will be invoked every 5 s during the call, the steganographic bandwidth will be about 0.2 bits/s.

Now, consider a method that uses intentionally delayed packets in the transmitter of the RTP stream to carry a steganogram such as LACK (Lost Audio Packets Steganography). As

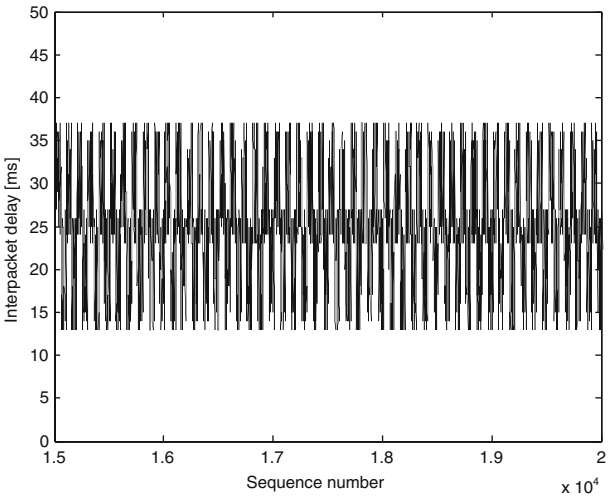


Fig. 14 Exemplary inter-packet delays for a steganographic method utilising two different rates for RTP packets

proven by the results discussed in Section 3, LACK can utilise both types of events which lead to packet dropping by jitter buffer (D1 and D2): delay spikes (by intentionally increasing inter-packet delay) and an RTP-packet burst at the beginning of the call, which can cause buffer underflows during the remaining part of the connection. The late packet drops (D2) occur almost twice as often as drops due to buffer overflows (D1), so it would be easy to explain that the probability of a packet being late is greater than its probability of arriving too soon. In a typical, nonsteganographic VoIP call, such events happen often enough to provide quite a good steganographic bandwidth, which can be expressed as:

$$S_B = r \cdot p_L[bits/s] \tag{9}$$

where r denotes the codec output rate (e.g., 64 kbit/s for G.711) and p_L is the probability of intentional RTP packet loss introduced by LACK. For example, if the G.711 codec is used and there is a 1% intentional loss the steganographic bandwidth achieved is about 640 bits/s.

Let us consider that a 100-ms jitter buffer is the size for which an acceptable voice quality was achieved (see Section 3). The average number of drops due to buffer overflow would then be about 300 during the whole connection (with a standard deviation of 1,490). Assuming that during the call about 150 intentionally invoked drops are introduced, the potential steganographic bandwidth achieved is about 350 bit/s. The average number of drops caused by delay spikes during the connection is about 750 (with a standard deviation of 1,882), resulting in steganographic bandwidth of about 900 bit/s if half of the average drops are invoked intentionally. Moreover, we can utilise a combine of these two types of drops during the same connection which results in an increased steganographic bandwidth. Because of the high standard deviations, this could be interpreted as making it extremely hard to predict the number of drops, thus the detection of LACK is not easy but is also very crucial. Of course, introducing jitter buffer losses must be carefully controlled to minimise the chance of detecting inserted data and to avoid excessive deterioration of voice quality. Additionally, packet losses introduced by the network must be carefully monitored. Because LACK uses legitimate RTP traffic, it thus increases overall packet losses. To ensure that the total packet loss introduced by the network and by LACK will not degrade the perceived quality of the conversation, the level of packet loss used for steganographic purposes must be controlled and dynamically adapted.

The high, potentially steganographic bandwidth of LACK makes it the most dangerous method among all those presented in this study that may influence an RTP stream. Thus, developing and implementing steganalysis methods for LACK is crucial.

5 Conclusions and future work

In this study delays and losses of voice (RTP) packets during real VoIP traffic were inspected in detail. Modifying the RTP packet stream potentially provides many of opportunities for hidden communication, as the packets may be delayed, reordered or intentionally lost. To assess whether RTP streams are suitable for steganographic purposes, an experiments was conducted, in which 100 average VoIP calls (of typical duration, connection path length, codec, loss concealment method and jitter buffer sizes) were performed. The experimental data was evaluated with respect to RTP packet losses including physical losses and losses caused by jitter buffer, where late packet drops and buffer overflows were distinguished, and the corresponding results for such losses were

presented. Most importantly, the results were analysed to evaluate the feasibility of implementing RTP steganographic methods based on real VoIP traffic.

Steganographic traffic is harder to detect, when its characteristic is similar to normal (innocent) traffic that can be observed in a network. The results obtained proved that some of the proposed methods may be quite easily detected, as, e.g., reordering was not present in the captured data, thus the feasibility of such methods is questionable. On the other hand, when steganographic method mimics some often-observed behaviour of the protocol, its detection may be hard. For example, LACK may mimic delay spikes, characteristic formation of packets which can lead to packet drops at the receiving end. In result, this method is quite feasible, and thus it may be considered as a threat to network security. LACK can use RTP packet sequences that will surely lead to jitter buffer losses by causing late packet drops or jitter buffer overflows. LACK may provide a potential steganographic bandwidth of hundreds of bits per second and be more difficult to detect than the other steganographic methods considered here. Further research concerning analysing VoIP traffic should identify often-observed protocols behaviours (packet exchanges) that can be utilized by potential new steganographic methods. Usage of such methods can lead to hiding of steganographic data that may be even more difficult to detect.

In future work, more VoIP data must be analysed to verify and confirm with greater accuracy the results obtained and presented in this paper. Moreover, it was shown that some steganographic methods utilising RTP can pose a serious threat to network security, hence detection solutions must be designed and developed.

References

1. Begtasevic F, Van Mieghem P (2001) Measurements of the hopcount in Internet. In: Proc. of the Passive and Active Measurement. 2001
2. Berk V, Giani A, Cybenko G (2005) Detection of covert channel encoding in network packet delays. Tech. Rep. TR2005-536, Department of Computer Science, Dartmouth College, November 2005, URL: http://www.ists.dartmouth.edu/library/149.pdf
3. Birke R, Mellia M, Petracca M, Rossi D (2007) Understanding VoIP from backbone measurements. In Proc. of 26th IEEE International Conference on Computer Communications (INFOCOM 2007), May 2007, pp. 2027-35, ISBN 1-4244-1047-9
4. Borella M, Swider D, Uludag S, Brewster G (1998) Internet packet loss: measurements and implications for End-to-End QoS. In Proc. of International Conference on Parallel Processing, August 1998
5. Cole RG, Rosenbluth JH (2001) Voice over IP performance monitoring. ACM SIGCOMM Computer Communication Review, vol. 31 no. 2, pp. 9–24, April 2001
6. Fei A, Pei G, Liu R, Zhang L (1998) Measurements on delay and hop-count of the internet. Proc. IEEE GLOBECOM'98, 1998
7. Girling CG (1987) Channels in LAN's. IEEE Trans Software Eng SE-13(2):292–296
8. Guha S, Daswani N, Jain R (2006) An experimental study of the skype peer-to-peer VoIP system. Sixth International Workshop on Peer-to-Peer Systems (IPTPS), February 2006
9. ITU-T Recommendation: G.711 (1988) Pulse code modulation (PCM) of voice frequencies, November 1988
10. ITU-T Recommendation: P.800 (1996) Methods for subjective determination of transmission quality, September 1996
11. ITU-T, Recommendation G.107 (2002) The E-model: a computational model for use in transmission planning, 2002
12. Kundur D, Ahsan K (2003) Practical internet steganography: data hiding in IP. Proceedings of the Texas Workshop on Security of Information Systems, April 2003

13. Liang YJ, Farber N, Girod B (2003) Adaptive playout scheduling and loss concealment for voice communications over IP networks. IEEE Transactions on Multimedia 5(4):532–543
14. Lubacz J, Mazurczyk W, Szczypiorski K (2010) Vice over IP In: IEEE Spectrum, ISSN: 0018-9235, February, pp. 40–45
15. Markopoulou AP, Tobagi FA, Karam MJ (2002) Assessment of VoIP quality over internet backbones. IEEE Infocom, New York
16. Mazurczyk W, Szczypiorski K (2008) Steganography of VoIP Streams, In: R. Meersman and Z. Tari (Eds.): OTM 2008, Part II—Lecture Notes in Computer Science (LNCS) 5332, Springer-Verlag Berlin Heidelberg, Proc. of The 3rd International Symposium on Information Security (IS'08), Monterrey, Mexico, November 2008, pp. 1001–1018
17. Na S, Yoo S (2002) Allowable propagation delay for VoIP calls of acceptable quality. In: Chang, W. (ed.) AISA 2002. LNCS, vol. 2402, pp. 469–480. Springer, Heidelberg, 2002
18. Narbutt M, Murphy L (2003) VoIP playout buffer adjustment using adaptive estimation of network delays. In Proceedings of 18th International Teletraffic Congress (ITC-18), 2003, pp. 1171–1180
19. Petitcolas F, Anderson R, Kuhn M (1999) Information hiding—a survey IEEE. Special Issue on Protection of Multimedia Content, July 1999
20. Ramjee R, Kurose J, Towsley D, Schulzrinne H (1994) Adaptive playout mechanisms for packetized audio applications in wide-area networks. In Proceedings of the IEEE INFOCOM 1994, 1994, pp. 680–688
21. Schechter SE, Smith MD (2003) Access for sale, ACM Workshop on Rapid Malcode (WORM'03). ACM SIGSAC, November 2003
22. Schulzrinne H, Caspkner S, Frederick R, Jacobson V (2003) RTP: a transport protocol for real-time applications. IETF, RFC 3550, July 2003
23. Servetto SD, Vetterli M (2001) Communication using phantoms: covert channels in the internet. In Proc. of IEEE International Symposium on Information Theory, June 2001
24. Skype–URL: http://www.skype.com
25. Sreenan CJ, Chen J, Agrawal P, Narendran B (2000) Delay reduction techniques for playout buffering. IEEE Transactions on Multimedia 2:88–100
26. Wu C, Chen K, Huang C, Lei C (2009) An empirical evaluation of VoIP playout buffer dimensioning in Skype Google Talk and MSN Messenger. In Proceedings of ACM NOSSDAV, 2009
27. Wireshark–URL: http://www.wireshark.org
28. X-lite–URL: http://www.counterpath.com/x-lite.html
29. Zhou X, Van Mieghem P (2005) Hopcount and E2E delay: IPv6 Versus IPv4. PAM2005, Boston

Wojciech Mazurczyk holds an M.Sc. (2004) and a Ph.D. (2009) in telecommunications from the Faculty of Electronics and Information Technology, Warsaw University of Technology (WUT, Poland) and is now an Assistant Professor at WUT and the author of over 40 scientific papers and over 25 invited talks on information security and telecommunications. His main research interests are information hiding techniques, network security and multimedia services, and he is also a research leader of the Network Security Group at WUT (secgroup.pl). Personal website: http://mazurczyk.com.

Krzysztof Cabaj holds an M.Sc (2004) and a Ph.D. (2009) in computer science from the Faculty of Electronics and Information Technology, Warsaw University of Technology (WUT), and is an Assistant Professor at WUT and a researcher in the Network Security Group formed at WUT. He has served as an Instructor of Cisco Academy courses: CCNA, CCNP and NS at the International Telecommunication Union Internet Training Centre (ITU-ITC). His research interests include network security, honeypots and data-mining techniques. He is the author or co-author of over 20 publications.

Krzysztof Szczypiorski holds an M.Sc. (1997) and a Ph.D. (2007) in telecommunications both with honours from the Faculty of Electronics and Information Technology, Warsaw University of Technology (WUT), and is an Assistant Professor at WUT. He is the founder and head of the International Telecommunication Union Internet Training Centre (ITU-ITC), established in 2003. He is also a research leader of the Network Security Group at WUT (secgroup.pl). His research interests include network security, steganography and wireless networks. He is the author or co-author of over 110 publications including 65 papers, two patent applications, and 35 invited talks.

Soft Comput (2011) 15:505–515
DOI 10.1007/s00500-009-0530-1

FOCUS

Retransmission steganography and its detection

Wojciech Mazurczyk · Miłosz Smolarczyk · Krzysztof Szczypiorski

Published online: 5 November 2009

Abstract The paper presents a new steganographic method called RSTEG (retransmission steganography), which is intended for a broad class of protocols that utilises retransmission mechanisms. The main innovation of RSTEG is to not acknowledge a successfully received packet in order to intentionally invoke retransmission. The retransmitted packet carries a steganogram instead of user data in the payload field. RSTEG is presented in the broad context of network steganography, and the utilisation of RSTEG for TCP (transmission control protocol) retransmission mechanisms is described in detail. Simulation results are also presented with the main aim of measuring and comparing the steganographic bandwidth of the proposed method for different TCP retransmission mechanisms, as well as to determine the influence of RSTEG on the network retransmission level.

Keywords RSTEG · Steganography · Retransmission mechanism

1 Introduction: network steganography and its classification

Communication network steganography is a method of hiding secret data in the normal data transmissions of users so that it ideally cannot be detected by third parties. Many new methods have been proposed and analysed, including those in Zander et al. (2007), Petitcolas et al. (1999) and Murdoch et al. (2005). Network steganography methods may be viewed as a threat to network security, as they may be used as a tool for confidential information leakage, for example. For this reason, it is important to identify possibilities for covert communication, as knowledge of information hiding procedures may be used to develop countermeasures. To detect the existence of hidden data inside the network, traffic, steganalysis methods are used. Steganalysis tools identify suspected network communication and try to determine whether or not it carries hidden information. If it is possible, they should also recover hidden information.

Network steganography may be classified (Mazurczyk et al. 2008) into three broad groups (Fig. 1):

- steganographic methods that modify packets (MP) including network protocol headers or payload fields;
- steganographic methods that modify the structure of packet streams (MS), for example, by affecting the order of packets, modifying inter-packet delay or introducing intentional losses;
- Hybrid steganographic methods (HB) that modify both the content of packets and their timing and ordering.

Examples of methods for each group and their characteristic features are described in Tables 1, 2 and 3.

In the context of the above classification of network steganography methods, we propose a new hybrid method called RSTEG (retransmission steganography), which is intended for a broad class of protocols that utilise retransmission mechanisms. The main innovation of RSTEG is to not acknowledge a successfully received packet to intentionally invoke retransmission. The

W. Mazurczyk (✉) · M. Smolarczyk · K. Szczypiorski
Warsaw University of Technology, Institute of Telecommunications, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland
e-mail: w.mazurczyk@tele.pw.edu.pl

M. Smolarczyk
e-mail: milosz.smolarczyk@gmail.com

K. Szczypiorski
e-mail: ksz@tele.pw.edu.pl

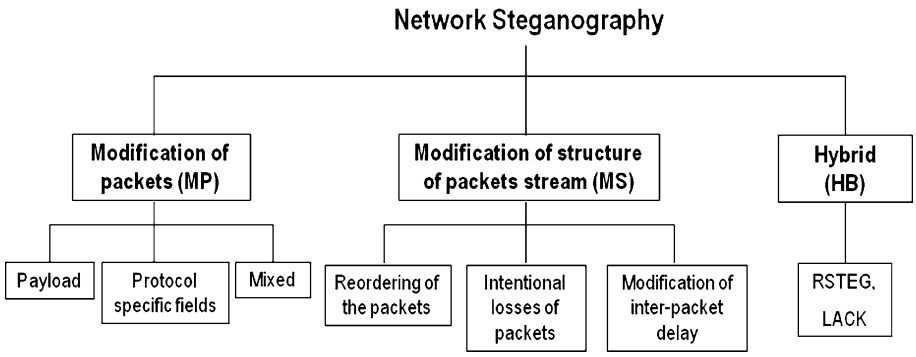


Fig. 1 A network steganography classification

Table 1 Examples and characteristic features of steganographic MP methods

MP methods	Examples of steganographic methods	Features
Methods that modify protocol-specific fields	Methods based on the modification of IP, TCP and UDP headers fields (Murdoch and Lewis 2005)	Yield relatively high steganographic capacity. Implementation and detection is relatively straightforward. Drawbacks include potential loss of protocol functionality
Methods that modify packet payload	Watermarking algorithms (Cox et al. 1997; Chen and Wornell 2001), speech coded steganographic techniques.	Generally yield lower steganographic capacity and are more difficult to implement and detect. Drawbacks include potential deterioration of transmission quality, e.g. if applied to VoIP (Voice over IP)
Mixed techniques	HICCUPS (hidden communication system for corrupted networks (Szczypiorski 2003)	Offer high steganographic capacity, but the implementation is more difficult than other methods due to the required low-level hardware access. For the same reason, steganalysis is more difficult to perform. Drawbacks include increased frame error rate

Table 2 Examples and characteristic features of steganographic MS methods

Examples of MS methods	Features
Methods that affect the sequence order of packets (Kundur and Ahsan 2003) Methods that modify inter-packet delay (Berk et al. 2005) Methods that introduce intentional losses by skipping sequence numbers at the sender (Servetto and Vetterli 2001)	• Sender–receiver synchronisation required • Lower steganographic capacity and more difficulty in detecting than methods that utilise protocol-specific fields • Straightforward implementation • Drawbacks include delays that may affect transmission quality

Table 3 Examples and characteristic features of steganographic HB methods

Examples of HB methods	Features
LACK (Lost Audio PaCKets Steganography) (Mazurczyk and Szczypiorski 2008) RSTEG (which is presented in detail in this paper)	• Modify both packets and their time dependencies • High steganographic capacity • Hard to detect • Sender–receiver synchronisation not required • Straightforward implementation • Drawbacks include a loss in connection quality

retransmitted packet of user data then carries a steganogram in the payload field.

2 Related work

Currently, there are few proposed steganographic methods that can incorporate retransmission mechanisms. Handel and Sandford (1996) proposed a steganographic method for Ethernet CSMA/CD (carrier sense multiple access/collision detection), which uses a retransmission mechanism after collisions. If frame collisions occur, then a jam signal is issued, and the senders back off for a random amount of time. To send a single hidden bit, a back-off delay of either zero or a maximum value is used so that the hidden data rate is one bit per frame. The receiver extracts a

steganogram by analysing the order of the frame arrivals after collisions.

Krätzer et al. (2006) proposed a steganographic method for the 802.11 protocol, as an extension of (Szczypiorski 2003), which transmits hidden information through the retransmission of frames. The sender encodes hidden data by duplicating frames transmitted to a receiver. The receiver decodes the hidden data by detecting the duplications.

The rest of the paper is dedicated to presenting the RSTEG steganographic method. Section 3 describes RSTEG in detail as well as communication scenarios in which it may be used. Performance issues involved in using the method are also discussed. In Sect. 4, results from an application of RSTEG to a TCP protocol simulation are presented. Section 5 concludes our work and indicates possible future research.

3 General idea of RSTEG and communication scenarios

Retransmission steganography can be used for all protocols that utilise retransmissions at different layers of OSI RM. A generic retransmission mechanism based on time-outs is presented in Fig. 2. RSTEG may be applied also to other retransmission mechanisms in TCP, such as FR/R (fast retransmit and recovery) (Stevens 1997) or SACK (selective acknowledgement) (Mathis et al. 1996).

In a simplified situation, a typical protocol that uses a retransmission mechanism based on time-outs obligates a receiver to acknowledge each received packet. When the packet is not successfully received, no acknowledgement is sent after the time-out expires, and so the packet is retransmitted (Fig. 2).

As mentioned in Sect. 1, RSTEG uses a retransmission mechanism to exchange steganograms. Both a sender and a receiver are aware of the steganographic procedure. They reliably exchange packets during their connection; that is, they transfer a file. At some point during the connection after successfully receiving a packet, the receiver intentionally does not issue an acknowledgement message. In a normal situation, a sender is obligated to retransmit the lost packet when the time frame within which packet acknowledgement should have been received expires. In the context of RSTEG, a sender replaces original payload with a steganogram instead of sending the same packet again. When the retransmitted packet reaches the receiver, he/she can then extract hidden information (Fig. 2).

Four possible hidden communication scenarios may be considered in the context of RSTEG (Fig. 3). Note that for few scenarios presented in Fig. 3, the packet sender and the packet receiver do not take part in hidden communication.

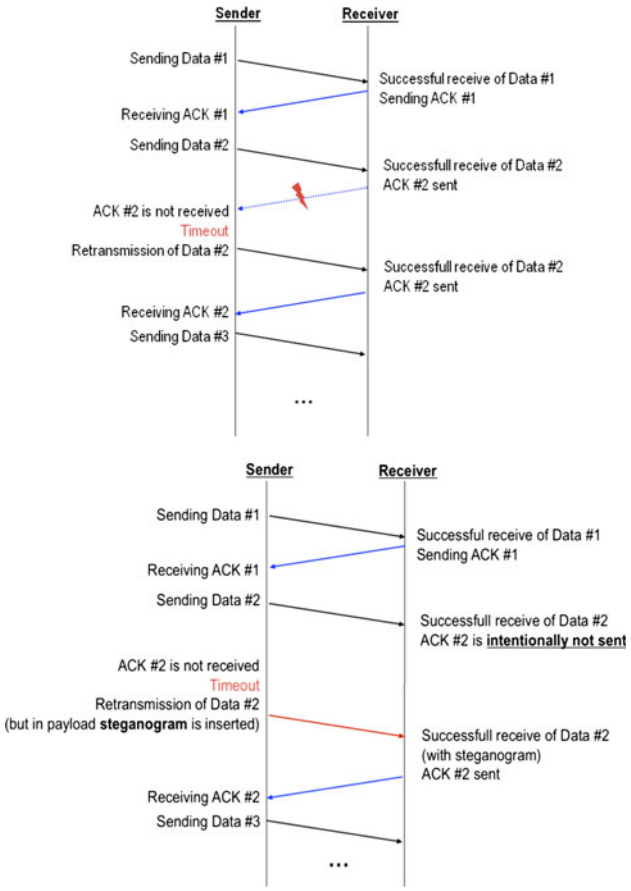


Fig. 2 Generic retransmission mechanism based on time-outs (*above*); RSTEG (*below*)

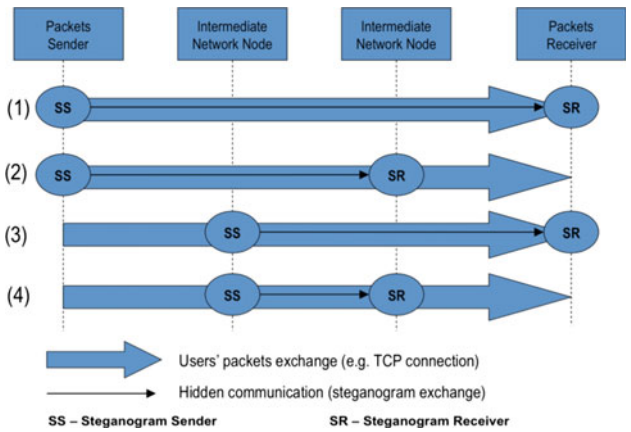


Fig. 3 Hidden communication scenarios for RSTEG

Only a part of their communication path is utilised by intermediate nodes, which are SS (steganogram sender) and SR (steganogram receiver).

Scenario (1) is most common: the sender, who is the steganogram sender (SS), and the receiver, who is the

steganogram receiver (SR), engage in a connection and simultaneously exchange steganograms. The conversation path is the same as the hidden data path. RSTEG for this scenario works as follows:

(1-1) End-to-end connection is established between sender and receiver, and the packets are exchanged.
(1-2) At some point, the receiver does not acknowledge a successfully acquired packet.
(1-3) After the retransmission timer expires, the packet is retransmitted and in its payload a steganogram is inserted.
(1-4) The receiver is able to distinguish a retransmitted packet, so when it reaches the receiver, he/she extracts a steganogram.

In the next three scenarios (2–4 in Fig. 3), only part of the connected end-to-end path is used for hidden communication as a result of actions undertaken by intermediate nodes; the sender and/or receiver are, in principle, unaware of the steganographic data exchange.

In scenario (2), one intermediate node is involved in hidden communication with the original packet sender (SS). The steganographic procedure for this scenario works as follows:

(2-1) Whilst the connection lasts, one packet is selected by the sender and is marked for hidden communication.
(2-2) When the modified packet reaches the SR, the SR copies a payload and drops the packet. Now, both the SS and SR know that the retransmission of this packet will be used for covert communication.
(2-3) When the retransmission time-out expires, the packet is retransmitted by the sender, and its original payload is replaced with a steganogram.
(2-4) When the modified retransmitted packet reaches the SR, the SR extracts a steganogram and inserts the original payload that was copied earlier and then sends it to the receiver.

In scenario (3), there is also one intermediate node involved in hidden communication (the SS), and the SR is located in the receiver. The steganographic procedure for this scenario works as follows:

(3-1) Whilst the connection lasts, one packet is selected by the intermediate node (SR) and is marked for hidden communication.
(3-2) When the packet successfully reaches the receiver (SR), the SR intentionally does not issue an acknowledgement.
(3-3) When the retransmission time-out expires, the packet is retransmitted by the sender.
(3-4) When the retransmitted packet reaches SS, its payload is replaced with a steganogram.
(3-5) When the modified, retransmitted packet reaches SR, the SR extracts a steganogram.

In scenario (4), two intermediate nodes are involved in hidden communication and utilise existing end-to-end connection between sender and receiver. RSTEG for this scenario works as follows:

(4-1) Whilst the connection lasts, one packet is selected by the SS and is marked for hidden communication.
(4-2) When the modified packet reaches the SR, the SR copies the payload and drops the packet. Now, both the SS and SR know that retransmission of this packet will be used for covert communication.
(4-3) When the retransmission time-out expires, the packet is retransmitted by the sender.
(4-4) When the retransmitted packet reaches the SS, its payload is replaced with the steganogram.
(4-5) When the modified retransmitted packet reaches the SR, the SR extracts the steganogram and inserts the original payload that was copied earlier and sends it to the receiver.

Of the above scenarios, scenario (1) is easiest to implement; scenarios (2)–(4) require control over the intermediate node used for hidden communication and that all packets traverse through it during connection. On the other hand, scenarios (2), (3) and, in particular, (4) are more difficult to detect than (1). The typical location of the node used for steganalysis is near the sender or receiver of the packets. Thus, in scenarios in which only part of the communication path is used, it may be more difficult to uncover.

The performance of RSTEG depends on many factors, such as the details of the communication procedure (in particular, the size of the packet payload, the rate at which segments are generated and so on). No real-world steganographic method is perfect; whatever the method, the hidden information can be potentially discovered. In general, the more hidden information is inserted into the data stream, the greater the chance that it will be detected, for example, by scanning the data flow or by some other steganalysis methods.

Moreover, the more packets that are used to send covert data, the higher will be the retransmission rate, which allows easier detection. That is why the procedure of inserting hidden data has to be carefully chosen and controlled to minimise the chance of detecting inserted data.

Additionally, packet losses introduced by the network must be carefully monitored. Because RSTEG uses legitimate traffic, it thus increases the overall packet losses. To ensure that the total packet loss introduced

by the network and by RSTEG is not too high when compared with other connections in the same network, the level of the retransmissions used for steganographic purposes must be controlled and dynamically adapted.

4 RSTEG in TCP: functioning, detection and experimental results

Applying RSTEG to TCP is the natural choice for IP networks, as a vast amount of Internet traffic (about 80–90%) is based on this protocol. For TCP, the following retransmission mechanisms are defined:

- RTO (retransmission time-outs) (Postel 1981) in which segment loss detection is based on RTO timer expiration. Results from Rewaskar et al. (2007) show that 60–88% of all retransmissions on the Internet were caused by RTO mechanism. In RTO, a segment is considered lost if the receiver does not receive an acknowledgement segment (ACK) after the specified period of time, after which it is retransmitted. The RTO timer value varies in TCP implementation across different operating systems, and it depends mainly on RTT (round trip time) and its variation. If the RTO timer is set to too low a value, it may cause too many spurious retransmissions; otherwise, the sender will wait too long to retransmit a lost segment, which may cause throughput decrease.
- FR/R (fast retransmit/recovery) is based on detecting duplicate ACKs (that is, ACKs with the same acknowledgement number). A receiver acknowledges all segments delivered in order. When segments arrive out of order, the receiver must not increase the acknowledgement number so as to avoid data gaps, but instead sends ACKs with unchanged acknowledgement number values, which are called duplicate ACKs (dupACKs). Usually, a segment is considered lost after the receipt of three duplicate ACKs. Issuing duplicate ACKs by the receiver is often a result of out-of-order segment delivery. If the number of duplicate ACKs that triggers retransmission is too small, it can cause too many retransmissions and can degrade network performance.
- SACK (selective acknowledgement) is based on fast retransmit/recovery. It uses an extended ACK option that contains blocks edges to deduce which received blocks of data are non-contiguous. When retransmission is triggered, only missing segments are retransmitted. This feature of SACK decreases network load.

4.1 RSTEG insertion and extracting procedures for TCP

The intentional retransmissions due to RSTEG should be kept at a reasonable level to avoid detection. To achieve this goal, it is necessary to determine the average number of natural retransmissions in TCP-based Internet traffic as well as to know how intentional retransmissions affect the network retransmission rate. Usually, network retransmissions are caused by network overload, excessive delays or reordering of packets (Rewaskar et al. 2007), and their number is estimated to account for up to 7% of all Internet traffic (Rewaskar et al. 2007; Internet Traffic Report (http://www.internettrafficreport.com/30day.htm); Chen et al. 2001).

RSTEG can be applied to all retransmission mechanisms presented above. It requires modification to both the sender and the receiver. A sender should control the insertion procedure and decide when a receiver should invoke a retransmission. The sender is also responsible for keeping the number of retransmissions at a non-suspicious level. The receiver's role is to detect when the sender indicates that intentional retransmission should be triggered. Then, when the retransmitted segment arrives, the receiver should be able to extract the steganogram.

The sender must be able to mark segments selected for hidden communication (that is, retransmission request segments), so the receiver would know for which segments retransmissions should be invoked and which segments contain steganograms. However, marked TCP segment should not differ from those sent during a connection. The following procedure for marking sender segments is proposed. Let us assume that the sender and receiver share a secret Steg-Key (SK). For each fragment chosen for steganographic communication, the following hash function (H) is used to calculate the identifying sequence (IS):

$$\mathrm{IS} = H(\mathrm{SK}||\text{Sequence Number}||\text{TCP Checksum}||\mathrm{CB}). \quad (1)$$

Note that Sequence Number and TCP Checksum denote values from the chosen TCP header fields in segments, || is the bits concatenation function, and CB is a control bit that allows the receiver to distinguish a retransmission request segment from a segment with a steganogram. For every TCP segment used for hidden communications, the resulting IS will have different value due to the variety of values in the Sequence Number and TCP Checksum header fields. All IS bits (or only selected ones) are distributed by the sender across a segment's payload field in a predefined manner. The receiver must analyse each incoming segment; based on SK and values from the TCP header, the receiver calculates two values of IS, namely, one with CB = 1 and one with CB = 0. Then the receiver

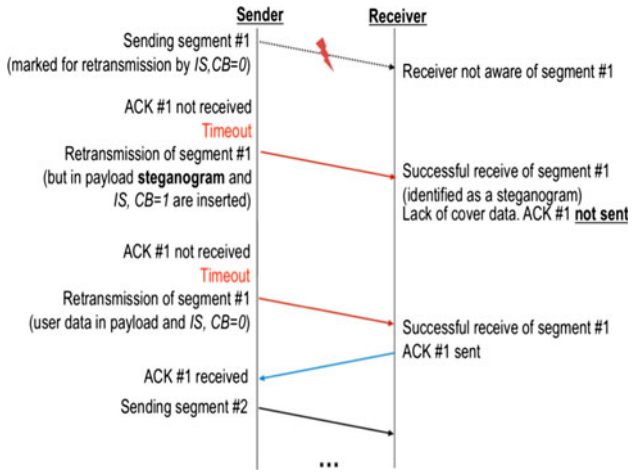


Fig. 4 RTO-based RSTEG segment recovery example

checks if and which IS is present inside the received segment.

Problems may arise when the segment that informs the receiver of a necessity to invoke an intentional retransmission (which contains user data together with the IS) is lost due to network conditions. In that case, a normal retransmission is triggered, and the receiver is not aware that the segment with hidden data will be sent. However, in this case, the sender believes that retransmission was invoked intentionally by the receiver, and so he/she issues the segment with steganogram and the IS. In this scenario, user data will be lost, and the cover connection may be disturbed.

To address the situation in which the receiver reads a segment with an unexpected steganogram, the receiver should not acknowledge reception of this segment until he/she receives the segment with user data. When the ACK is not sent to the sender, another retransmission is invoked. The sender is aware of the data delivery failure, but he/she does not know which segment to retransmit, so he/she first issues a segment with user data. If delivery confirmation is still missing, then the segment with the steganogram is sent. The situation continues until the sender receives the correct ACK. This mechanism of correcting steganogram network losses is illustrated in Fig. 4.

For example, consider the scenario in which we invoke 0.5% of intentional retransmissions. If 5% is lost, it means that the above-described mechanism will take place only for 0.025% of steganogram segments, and thus it will be used rarely.

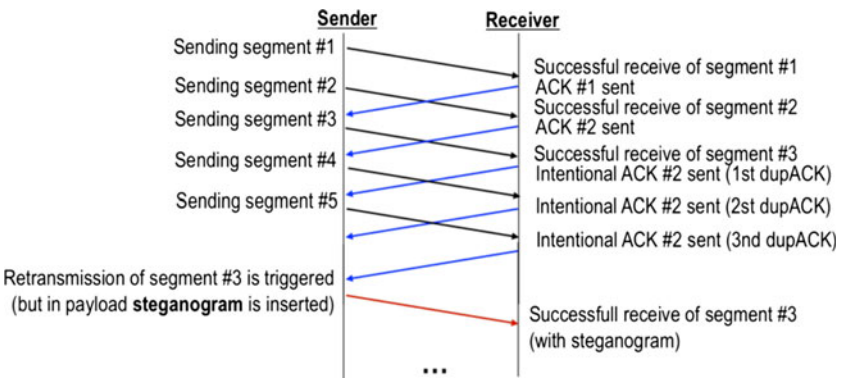
The above RSTEG may be applied to the retransmission mechanisms presented above as follows:

- RTO-based RSTEG: the sender marks a segment selected for hidden communication by distributing the IS across its payload. After successful segment delivery, the receiver does not issue an ACK message. When the RTO timer expires, the sender sends a steganogram inside the retransmitted segment's payload (see Fig. 2). The receiver extracts the steganogram and sends the appropriate acknowledgement.
- FR/R-based RSTEG: the sender marks the segment selected for hidden communication by distributing the IS across its payload. After successful segment delivery, the receiver starts to issue duplicate ACKs to trigger retransmission. When the ACK counter at the sender side exceeds the specified value, the segment is retransmitted (see Fig. 5). Payload of the retransmitted segment contains a steganogram. The receiver extracts the steganogram and sends an appropriate acknowledgement.
- SACK-based RSTEG: the scenario is exactly the same as FR/R, but in the case of SACK, it is possible that many segments are retransmitted because of potential non-contiguous data delivery.

4.2 An experimental evaluation of the influence of RSTEG on TCP connections

Simulations were generated using ns-2 Simulator ver. 2.33 (The Network Simulator Webpage (http://www.isi.edu/nsnam/ns/ns-build.html)) with the following modifications. The adaptation of ns-2 Simulator to RSTEG required

Fig. 5 FR/R-based RSTEG



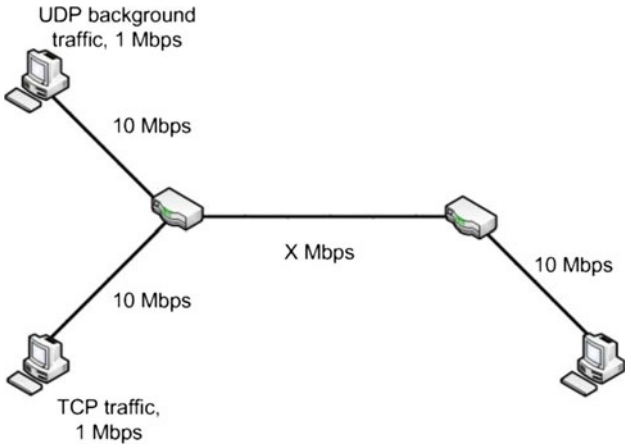


Fig. 6 RSTEG simulation scenario

Table 4 The chosen bandwidth for bottleneck link (X) for different TCP retransmission mechanisms to achieve 3 and 5% NR_P

NR_P/TCP retrans.	RTO (Mbps)	FR/R (Mbps)	SACK (Mbps)
3%	1.985	1.985	1.985
5%	1.8	1.8	1.9

only modifications of the receiver. The receiving functionality of the segments was modified to intentionally not issue ACKs (in the case of RTO) or to not increase the acknowledgement number (in the cases of FR/R and SACK). The decision regarding which segment would be treated as lost is made randomly according to a parameter that specifies the intentional retransmissions frequency.

The network topology was matched to fit Internet traffic retransmission statistics. The simulation scenario consists of two traffic sources (TCP and UDP) and the bottleneck link between intermediate devices such as routers (see Fig. 6). Each traffic source is connected with a 10-Mbps link to the intermediate device. The receiver is also connected to its router with a 10-Mbps link. The UDP traffic source and the bandwidth of the link between intermediate devices (X) are chosen to introduce certain network retransmission probabilities (NR_P); due to network overload, NR_P is about 3 or 5%. Table 4 summarises the bandwidths of the bottleneck links that are used for simulation purposes.

The simulation results are based on comparing retransmissions for a network with RSTEG applied to TCP traffic as well as for a network without RSTEG retransmissions. Network traffic was measured for 9 min, starting at 1 min after the beginning of simulation. The RSTEG intentional retransmission probability (IR_P) was changed from 0 to 5% with intermediary steps at 0, 0.5, 1, 2, 3, 4 and 5%.

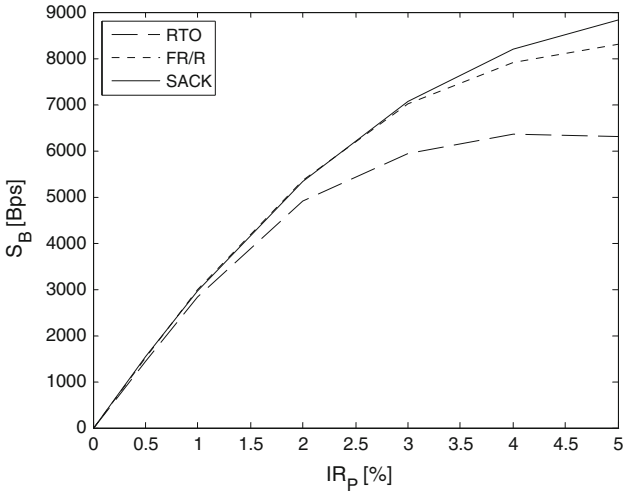


Fig. 7 S_B for TCP retransmission mechanisms when $NR_P = 3\%$ and IR_P varies

In the above simulation scenario, two parameters were measured for RSTEG:

- Steganographic bandwidth (S_B) is defined as the amount of the steganogram transmitted using RSTEG during 1 s (Bps). For different retransmission mechanisms in the TCP protocol, this parameter can be used to estimate which mechanism yields the highest S_B and is most suitable from a RSTEG utilisation point of view. S_B depends mainly on the size of the segment and the number of intentional retransmissions invoked, and so it may be expressed as

$$S_B = \frac{N_S \cdot S_S}{T} \quad \text{(Bps)} \tag{2}$$

  where N_S is the number of segments used for hidden communication, S_S the size of segment payload, and T is the duration of the connection.
- Retransmissions difference (R_D) is defined as the difference between retransmissions in a network after applying RSTEG and in a network before applying RSTEG. This parameter can be used to estimate the influence that RSTEG has on the TCP retransmissions rate. Thus, it can illustrate how to choose the correct intentional retransmission probability to limit the risk of detection. For example, if the network retransmission probability is 5%, 1% of intentional retransmissions are introduced by RSTEG, which causes the overall retransmission rate to increase to 7%, with $R_D = 2\%$.

The results for TCP retransmission mechanisms when $NR_P = 3\%$ and $NR_P = 5\%$ are presented in Figs. 7, 8, 9 and 10.

Tables 5 and 6 summarise the simulation results.

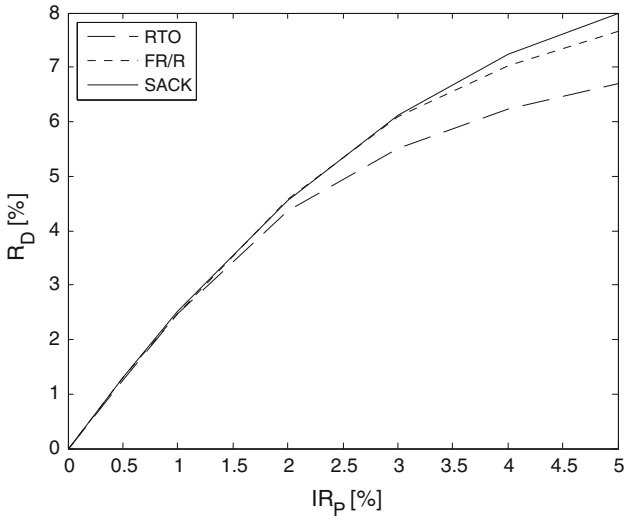


Fig. 8 R_D for TCP retransmission mechanisms when $NR_P = 3\%$ and IR_P varies

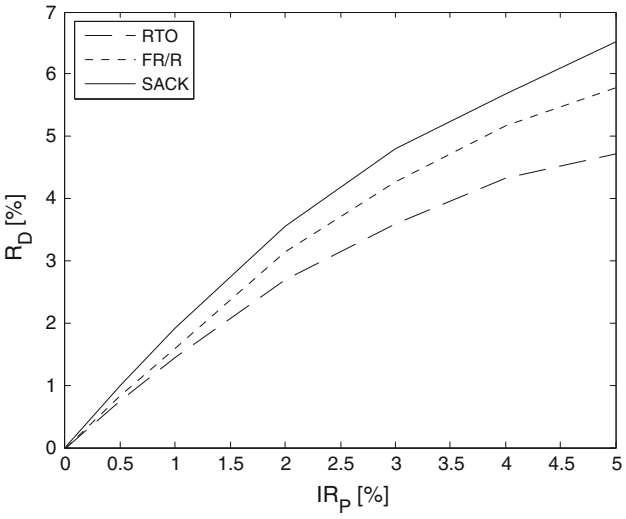


Fig. 10 R_D for TCP retransmission mechanisms when $NR_P = 5\%$ and IR_P varies

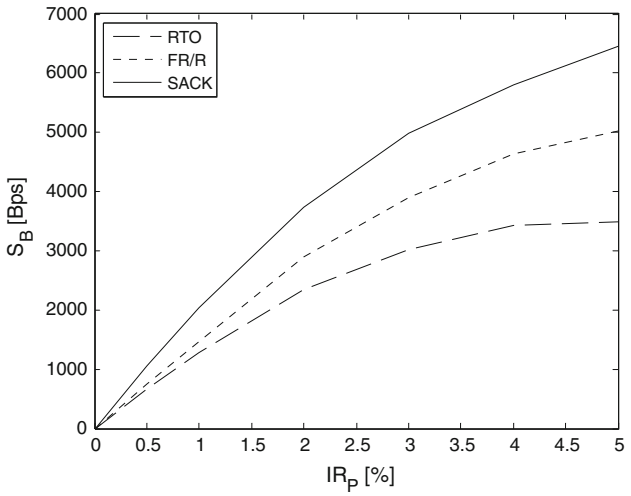


Fig. 9 S_B for TCP retransmission mechanisms when $NR_P = 5\%$ and IR_P varies

Based on the results presented above, one can conclude that for low intentional retransmission probability values (0–0.5% for $NR_P = 5\%$ and 0–1% for $NR_P = 5\%$), the resulting S_B values for all retransmission mechanisms are similar and, therefore, it is not important which of the retransmission mechanisms (that is, RTO, FR/R or SACK) is used. The higher the IR_P, the greater is the difference in the steganographic bandwidth. It is not surprising that RSTEG based on SACK and FR/R mechanisms yield higher steganographic bandwidth than RTO-based RSTEG, as the former are more effective retransmission mechanisms. That is, under the same IR_P, they achieve greater S_B. However, higher steganographic bandwidth for RSTEG based on SACK and FR/R mechanisms increases the retransmission difference values in comparison to RTO-based RSTEG. This may increase the likelihood of detection of RSTEG. Thus, retransmission mechanisms for which R_D values are lower are favourable in terms of steganalysis. RTO-based RSTEG achieved the lowest steganographic bandwidth, but simultaneously introduced the lowest R_D. Considering this analysis and knowing that RTO is the most frequent retransmission mechanism used for TCP on the Internet (60–88%) suggests that RTO-based RSTEG is a favourable choice for TCP protocol if the risk of disclosure must be minimised. If detection issues are omitted, SACK-based RSTEG should be chosen to maximise the amount of steganogram that is sent.

Regarding RTO-based RSTEG and its appropriateness based on TCP protocol, Figs. 11 and 12 present a comparison of S_B and R_D when $IR_P = 3\%$ and $IR_P = 5\%$.

Results from Figs. 11 and 12 show that an increase in the number of retransmissions introduced in a network lowers the influence that RSTEG has on network retransmissions. That is, they are more difficult to detect, although the steganographic bandwidth is lower. An increase in network retransmissions means that it is easier to hide intentional retransmissions amongst unintentional retransmissions.

4.3 RSTEG steganalysis possibilities

Retransmissions in IP networks are a ‘natural phenomenon’, and so intentional retransmissions introduced by RSTEG are not easy to detect if they are kept at a reasonable level. The experimental results presented here show that RTO-based RSTEG is a favourable TCP retransmission mechanism in terms of steganalysis.

Table 5 Simulation results when $NR_P = 3\%$

IR_P (%)	RTO				FR/R				SACK			
	S_B (Bps)	σ_{SB}	R_D (%)	σ_{RD}	S_B (Bps)	σ_{SB}	R_D (%)	σ_{RD}	S_B (Bps)	σ_{SB}	R_D (%)	σ_{RD}
0.5	1,454	112.5	1.25	0.0971	1,530	92.8	1.25	0.1292	1,530	92.8	1.29	0.0778
1.0	2,821	164.3	2.45	0.1356	2,999	141.1	2.54	0.1302	2,999	141.1	2.54	0.1183
2.0	4,802	183.4	4.26	0.1503	5,395	171.3	4.67	0.1773	5,395	171.3	4.62	0.1445
3.0	5,982	96.4	5.54	0.0754	7,113	106.6	6.12	0.1384	7,113	106.6	6.17	0.0896
4.0	6,306	100.7	6.21	0.0911	8,128	157.3	7.03	0.1119	8,128	157.3	7.18	0.1355
5	6,320	81.5	6.72	0.0800	8,865	62.0	7.73	0.0830	8,865	62.0	8.07	0.0754

Table 6 Simulation results when $NR_P = 5\%$

IR_P (%)	RTO				FR/R				SACK			
	S_B (Bps)	σ_{SB}	R_D (%)	σ_{RD}	S_B (Bps)	σ_{SB}	R_D (%)	σ_{RD}	S_B (Bps)	σ_{SB}	R_D (%)	σ_{RD}
0.5	5,457	677	0.77	0.0680	5,474	694	0.75	0.0666	6,119	1,020	0.96	0.0938
1.0	6,068	1,288	1.46	0.0929	6,277	1,497	1.61	0.0736	7,119	2,020	1.90	0.1019
2.0	7,169	2,389	2.75	0.1186	7,699	2,919	3.15	0.1473	8,726	3,627	3.44	0.1092
3.0	7,848	3,068	3.62	0.1014	8,692	3,912	4.28	0.0982	9,998	4,899	4.71	0.1323
4.0	8,173	3,393	4.24	0.0881	9,406	4,626	5.14	0.1390	10,886	5,787	5.67	0.1115
5	8,304	3,524	4.74	0.1216	9,863	5,083	5.81	0.1101	11,549	6,450	6.50	0.0929

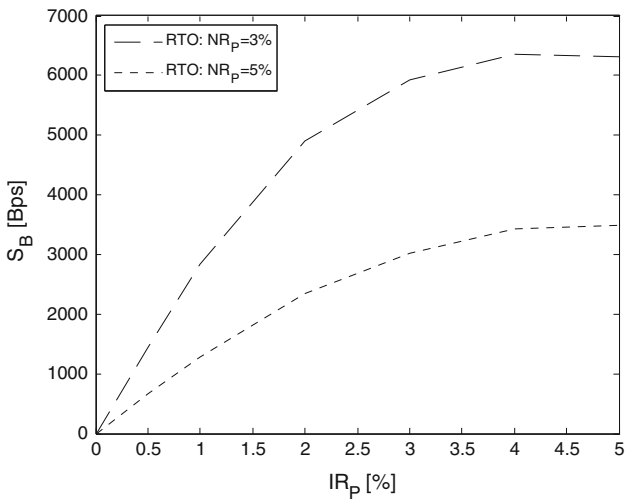


Fig. 11 S_B for RTO-based RSTEG as IR_P varies

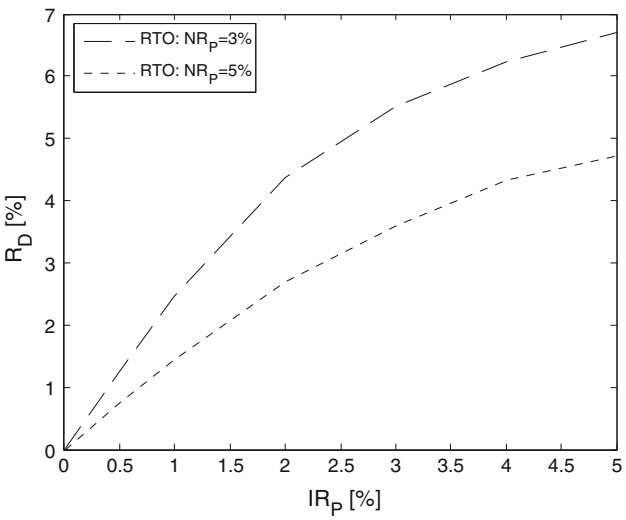


Fig. 12 R_D for RTO-based RSTEG as IR_P varies

Moreover, if the sender can observe the average retransmission rate in a network, then he/she can also choose an IR_P so as to limit the risk of detection.

One possible detection method is statistical steganalysis based on the network retransmission rate. If for certain TCP connections, the retransmission rate is significantly higher than for others, then potential usage of RSTEG may be detected. Such a steganalysis method involves the monitoring of TCP retransmission rates for all connections in a sub-network.

However, there is a solution that makes the steganalysis of RSTEG, as applied to TCP protocol, easier to perform. The proposed steganalysis method may be implemented with a passive warden (Fisk et al. 2002) (or some other network node responsible for steganography usage detection). Passive warden must be able to monitor all the TCP traffic and for each TCP connection it must store sent segments for the given period of time, which depends on the retransmission timer, i.e. passive warden must store the segment until it is acknowledged by the receiver, so the

retransmission is not possible any more. When there is a retransmission issued, passive warden compares originally sent segment with retransmitted one and if the payload differs, RSTEG is detected and the segment is dropped. However, it should be noted that there may be serious performance issues involved if passive warden monitors all the TCP connections and must store a large number of the segments.

On the other hand, it must be noted that based on results presented in Stone and Partridge (2000), up to 0.09% (1 in 1,100) of TCP segments may be corrupted due to network delivery. As a result, an imperfect copy of a segment may be sent to the receiver. After reception of the invalid segment, verification is performed based on the value in the TCP Checksum field, and the need to retransmit is signalled to the sender. Thus, in this scenario, the original segment and the retransmitted one will differ from each other. Occurrences of this effect in IP networks mask the use of RSTEG. Thus, the steganalysis methods described above may fail, because the warden will drop retransmitted segments when differences amongst segments are discovered and, as a result, user data will be lost.

It is worth noting that even for the low rates of intentional retransmission (0.09%) that are required to mask RSTEG, if we assume that the TCP segments are generated at a rate of 200 segments/s, with the connection lasting 5 min and the segment's payload size being 1,000 bytes, then this results in $S_B = 180$ Bps, which is a rather high bandwidth, considering the other steganographic methods presented in Sect. 1.

To summarise, measures to detect RSTEG have been proposed and can be utilised, but if the rate of intentional retransmissions is very low, then the detection of hidden communications may be difficult.

5 Conclusions and future work

Retransmission steganography is a hybrid network steganographic method based on the classification presented earlier in this paper. The steganographic bandwidth it can provide may be comparable for methods that modify packets only, and its bandwidth is higher than that of methods that only modify the structure of packet streams.

In this paper, we have focused on presenting the framework guiding this steganographic method and have showed how it may be applied and detected in the context of TCP protocol, which may be useful in developing detection measures. A more detailed evaluation of RSTEG performance for other protocols with retransmissions and in other layers of the TCP/IP stack is needed.

The simulation results show that to minimise the risk of detection, RTO-based retransmissions should be used by RSTEG, and intentional retransmissions should be kept to a reasonable level. However, to maximise the steganographic bandwidth, SACK-based RSTEG is more appropriate.

Application of RSTEG to TCP protocol is a logical choice for IP networks, but as shown in this paper, it can be detected, especially if intentional retransmissions are issued excessively. Nevertheless, RSTEG can be also used for other protocols that utilise retransmission mechanisms, in particular for wireless networks. We believe that RSTEG in this environment may be more difficult to detect; however, this claim requires a more detailed analysis. Analytical and experimental results concerning this issue will be presented by the authors in forthcoming papers.

References

Berk V, Giani A, Cybenko G (2005) Detection of covert channel encoding in network packet delays. Tech. Rep. TR2005-536, Department of Computer Science, Dartmouth College, URL: http://www.ists.dartmouth.edu/library/149.pdf

Chen B, Wornell G (2001) Quantization index modulation: a class of provably good methods for digital watermarking and information embedding. IEEE Trans Info Theory 47(4):1423–1443

Chen C, Mangrulkar M, Ramos N, Sarkar M (2001) Trends in TCP/IP retransmissions and resets. Technical Report. http://www-cse.ucsd.edu/classes/wi01/cse222/projects/reports/tcp-flags-13.pdf

Cox I, Kilian J, Leighton F, Shamoon T (1997) Secure spread spectrum watermarking for multimedia. IEEE Trans Image Process 6(12):1673–1687

Fisk G, Fisk M, Papadopoulos C, Neil J (2002) Eliminating steganography in Internet traffic with active wardens, 5th International Workshop on Information Hiding. Lect Notes Comput Sci 2578:18–35

Handel T, Sandford M (1996) Hiding data in the OSI network model. In: Proceedings of the 1st international workshop on information hiding, pp 23–38

Krätzer C, Dittmann J, Lang A, Kühne T (2006) WLAN steganography: a first practical review. In: Proceedings of the 8th ACM multimedia and security workshop

Kundur D, Ahsan K (2003) Practical Internet steganography: data hiding in IP. In: Proceedings of the Texas workshop on security of information systems

Mathis M, Mahdavi J, Floyd S, Romanow A (1996) TCP selective acknowledgment options. IETF RFC 2018

Mazurczyk W, Szczypiorski K (2008) Steganography of VoIP streams. In: Meersman R, Tari Z (eds) OTM 2008. Part II. Lecture notes in computer science (LNCS), vol 5332. Springer, Berlin. Proceedings of the 3rd international symposium on information security (IS'08), Monterrey, Mexico, November 10–11, pp 1001–1018

Mazurczyk W, Lubacz J, Szczypiorski K (2008) Hiding data in VoIP. In: Proceedings of the 26th army science conference (ASC 2008), Orlando, Florida, USA, December 1–4

Murdoch S, Lewis S (2005) Embedding covert channels into TCP/IP. In: Proceedings of the 7th international workshop on information hiding 2005, LNCS, vol 3727. Springer, Heidelberg, pp 247–261

Petitcolas F, Anderson R, Kuhn M (1999) Information hiding–a survey. IEEE Special Issue on Protection of Multimedia Content, vol 87, no. 7, pp 1062–1078

Postel J (1981) Transmission control protocol. IETF RFC 793
Rewaskar S, Kaur J, Smith F (2007) A performance study of loss detection/recovery in real-world TCP implementations. In: Proceedings of the IEEE international conference on network protocols, ICNP 2007, October 16–19, Beijing, China, ISBN 1-4244-1588-8, pp 256–265
Servetto S, Vetterli M (2001) Communication using phantoms: covert channels in the Internet. In: Proceedings of IEEE international symposium on information theory
Stevens W (1997) TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms. IETF RFC 2001
Stone J, Partridge C (2000) When the CRC and TCP checksum disagree. In: Proceedings of SIGCOMM 2000
Szczypiorski K (2003) HICCUPS: hidden communication system for corrupted networks. In: Proceedings of: ACS'2003, October 22–24, Miedzyzdroje, Poland, pp 31–40
Zander S, Armitage G, Branch P (2007) A survey of covert channels and countermeasures in computer network protocols. IEEE Commun Surv Tutorials 9(3):44–57 ISSN: 1553-877X

Telecommun Syst
DOI 10.1007/s11235-010-9362-7

Evaluation of steganographic methods for oversized IP packets

Wojciech Mazurczyk · Krzysztof Szczypiorski

Abstract This paper describes new network steganography methods that utilize mechanisms for handling oversized IP packets: IP fragmentation, PMTUD (Path MTU Discovery) and PLPMTUD (Packetization Layer Path MTU Discovery). In particular, for these mechanisms we propose two new steganographic methods and three extensions of existing ones. We present how mentioned mechanisms can be used to enable hidden communication for both versions of IP protocol: 4 and 6 and how they can be detected. Results for experimental evaluation of IP fragmentation steganographic methods are also enclosed in this paper.

Keywords Network steganography · IP fragmentation · PMTUD · PLPMTUD

1 Introduction

Steganographic methods hide secret data in users' normal data transmissions and in ideal situation hidden information and existence of hidden communication cannot be detected by third parties. Various steganographic methods have been proposed and analyzed, e.g. [1–4]. They may be seen as a threat to network security as they may be used as a tool to cause for example confidential information leakage. That is why it is important to identify potential possibilities for covert communication, because knowledge of the information hiding procedure can be used to develop countermeasures.

W. Mazurczyk (✉) · K. Szczypiorski
Institute of Telecommunications, Warsaw University of Technology, 15/19 Nowowiejska Str., 00-665 Warsaw, Poland
e-mail: wmazurczyk@tele.pw.edu.pl

K. Szczypiorski
e-mail: ksz@tele.pw.edu.pl

Published online: 18 June 2010

Both versions of IP protocol 4 [5] and 6 [9] were designed to be used on various transmission links. The maximum length of an IP packet is 64 kB but on most transmission links maximum packet length is smaller. This limited value characteristic for the specific link is called a MTU (Maximum Transmission Unit). MTU depends on the type of the transmission link e.g. for Ethernet—1500, wireless IEEE 802.11—2300 and PPP (Point to Point Protocol)—296 bytes.

There are two possibilities to transmit large IP packet through an end-to-end path that consists of links with different MTUs:

- Permit to divide oversized packet to smaller ones. To achieve this mechanism called IP fragmentation [5] has been standardized.
- Do not allow packet fragmentation and adjust IP packet size to so called PMTU (Path MTU)—the smallest, acceptable MTU along the entire end-to-end path. For this purpose two methods have been proposed PMTUD (Path MTU Discovery) [6] for IPv4 and [7] for IPv6 and PLPMTUD (Packetization Layer Path MTU Discovery) [8], which is enhancement of previous method for both versions of IP protocol.

Mechanisms for handling oversized packets like IP fragmentation, PMTUD or PLPMTUD are needed and used in network scenarios where in the end-to-end path intermediate links have smaller MTUs than the MTU of the end links. Below typical network scenarios that require dealing with oversized packets are listed:

- Usage of various tunneling protocols like GRE (Generic Routing Encapsulation), IPSec (IP Security), and L2TP (Layer Two Tunneling Protocol) which add headers and trailers thus reduce effective MTU.

- Using PPPoE (Point to Point Protocol over Ethernet) with ADSL (Asymmetric Digital Subscriber Line). PPPoE has 8 bytes header thus it reduces the effective MTU of the Ethernet to 1492 bytes.
- Using MPLS over Ethernet.
- Connections between endpoints in Token Ring or FDDI networks, which have an Ethernet link between them (with lower MTU) and other similar cases.

This work is extension of the previous authors' work [13]. The objectives of this paper are to:

- Describe mechanisms used to handle oversized packets in IPv4 and IPv6 networks.
- Present exiting network steganography methods that utilize these mechanisms.
- Propose two new steganographic methods and three extensions of existing ones All presented steganographic methods may be applied to both versions of IP protocol (4 and 6). Additionally, we show how IP fragmentation simplifies usage of methods that modify time relations between the packets.
- Present the experimental evaluation of steganographic bandwidth for IP fragmentation network steganography methods.

The rest of the paper is as follows. Section 2 describes existing mechanisms for handling oversized packets for IPv4 and IPv6 protocols. In Sect. 3 existing network steganography methods that utilize these mechanism are presented. Section 4 includes detailed description of new information hiding methods and their potential detection. Section 5 presents experimental results for IP fragmentation steganographic methods. Section 6 concludes our work.

2 Overview of mechanism for handling oversized IP packets

2.1 IP fragmentation

To accommodate MTU differences on links in end-to-end path in IP fragmentation, intermediate nodes are allowed to fragment oversized packets to smaller ones. Then receiver or some other network node (e.g. router) is responsible for reassembling the fragments back into the original IP packet.

IP fragmentation mechanism involves using the following fields of the IPv4 header (Fig. 1): *Identification*, *Fragment Offset* fields, along with the MF (More Fragments) and DF (Don't fragment) flags. It also needs to adjust values in *Total Length* and *Header Checksum* fields for each fragment to represent correct values. The above header fields are used as follows:

- *Identification* (16 bits) is a value assigned by the sender to each IP packet to enable correct reassembling of the fragments (each fragment has the same *Identification* value).

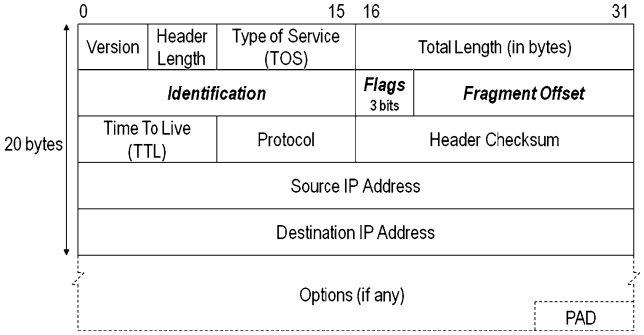


Fig. 1 The IPv4 protocol header (bolded are fields used by IP fragmentation)

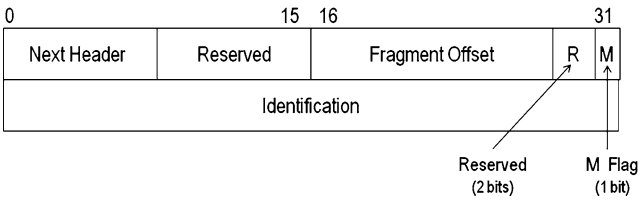


Fig. 2 IPv6 Fragment header extension

- *Fragment Offset* (13 bits) indicates which part of the original packet fragment carries.
- *Flags* field (3 bits) contains control flags. Bit '0' is reserved and is always set to 0. Bit '1' is the DF flag—if set to 0 fragmentation can occur; if set to 1 fragmentation is not possible. Bit '2' is the MF flag—if set to 0 and *Fragment Offset* is different from 0, this denotes the presence of last fragment and if set to 1 more fragments are expected to be received.

Similar mechanism is used in version 6 of IP protocol, where *Fragment extension header* (Fig. 2) is used to perform fragmentation. What differs IPv6 from IPv4 fragmentation is that it may only be performed by the sender and reassembly process have to take place only in the receiver and not in some intermediate node.

The example of the IP packet fragmentation for IPv4 is presented in Table 1. Original packet which size is 5140 bytes is divided into four fragments of maximum 1500 bytes.

There are several issues that make IP Fragmentation in IPv4 networks undesirable because it lowers the efficiency and reliability of communication. Fragmentation causes serious overhead for the receiver because while reassembling the fragments the receiver must allocate memory for the arriving fragments and after all of the fragments are received they are put back into original IP packet. While it is not an issue for a host as it has the time and memory resources to devote to this task, reassembly may be very inefficient on intermediate nodes (e.g. routers). Router is not able to determine the size of the original IP packet until the last fragment

Table 1 IP fragmentation example

Sequence	Identifier	Total length	DF	MF	Fragment offset
Original IP packet					
0	345	5140	0	0	0
IP Fragments					
0–0	345	1500	0	1	0
0–1	345	1500	0	1	185
0–2	345	1500	0	1	370
0–3	345	700	0	0	555

is received, so while reassembling it must assign a large receiving buffer.

Another fragmentation issue involves handling dropped fragments. If one fragment of an IP packet is dropped, then the entire original IP packet must be resent (all fragments).

Firewalls and NATs (Network Address Translation) may have trouble processing fragments correctly and in effect drop them. If the IP fragments are out of order, a firewall may block the non-initial fragments because they do not carry the information that would match the packet filter. This would mean that the original packet could not be reassembled by the receiving host. Similar problem may occur with NAT as it has problems with interpreting the IP fragment if it comes out of order.

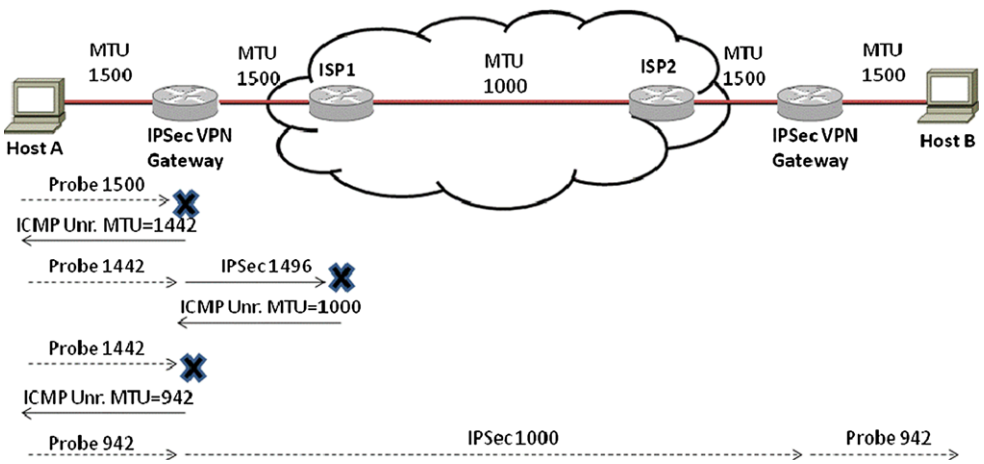
2.2 PMTUD (path MTU discovery)

PMTUD was standardized for IPv4 and published in 1990, but it did not become widely deployed for the next few years. Currently PMTUD is implemented in major operating systems (Windows, Unix, Linux)—in 2002 about 80–90% of endpoints on the Internet were using it. As mentioned in the introduction this mechanism was developed to avoid fragmentation in the path between the endpoints. Similar to IPv4 PMTUD mechanism was also developed and standardized for IPv6 [7].

PMTUD is used to dynamically determine the lowest MTU along the end-to-end path between packets sender and receiver. Instead of fragmenting packet, an endpoint determines the largest possible size of the packet that can be sent to a specific destination. An endpoint establishes the correct packet size associated with a specific path by sending packets with different sizes. Packets used by PTMUD are called probe messages and they have DF flag set in the IP protocol header. Their size is initially set to the senders link MTU. While sender generates probes he/she responds to possible ICMP (Internet Control Message Protocol) error reports that indicate a low MTU is present along the connection path. Sender receives a notification informing what packet size will be suitable. The notifications are requested by setting the DF flag in outgoing packets. For IPv4 the notifications arrive as ICMP messages known as "Fragmentation required, and DF flag set" (ICMP type 3, code 4), for IPv6 it is "Packet too big" message from ICMPv6 protocol [10]. PMTUD is working continually during connection because the path between sender and receiver can changed (e.g. because of link failure).

The PMTUD example is illustrated in Fig. 3. Host A sends packet to host B which size is set to 1500 bytes (default Ethernet MTU). The packet will be transmitted with use of IPSec tunnel, which begins at first router. Because the next link MTU is also 1500 bytes and IPSec adds 54 bytes overhead then total packet size exceeds admissible MTU. Thus the packet is dropped and ICMP message is sent back to the host A with suitable MTU for the next link. Then host A retries sending the packet by reducing its size to 1442 bytes to meet the limit, so packet can successfully traverse through first router. However, the link after next router has MTU of 1000 bytes so the packet is once again dropped and ICMP message is sent in host A direction but it is filtered out by first router. After the timeout expires host A retransmits the packet and receives ICMP message which

Fig. 3 PMTUD example



indicates necessity to decrease packet size to 942 bytes. This last MTU value is then used to successfully exchange data with host B.

It must be noted that there are security issues related with using PMTUD. In particular, sometimes network administrators treat all ICMP traffic as dangerous and block it, disabling possibility of using path MTU discovery. Other potential issues for TCP protocol are described in [11].

2.3 PLPMTUD (packetization layer path MTU discovery)

To alleviate issues related with using ICMP traffic for PMTUD, enhancement called PLPMTUD was developed and standardized in [8]. What differs PLPMTUD from PMTUD is that receiving probes messages are validated at the transport layer. It does not rely on ICMP or other messages from the network, instead it learns about correct MTU by starting with packets which size is relatively small and when they get through with progressively larger ones. In particular, PLPMTUD uses a searching technique to determine optimal PMTU. Each probe narrows the MTU search range. It may raise the lower limit on a successful probe receipt or lower the upper limit if probe fails. The isolated loss of a probe message is treated as an indication of an MTU limit and transport layer protocol is permitted to retransmit any missing data.

3 Related work

To authors best knowledge, there are no steganographic methods proposed for PMTUD and PLPMTUD mechanisms.

For IPv4 there are few existing methods that utilize IP fragmentation mechanism and fields in IP header related to it. Rowland [1] proposed multiplying each byte of the hidden data by 256 and inserts it directly into *Identification* header field. Cauich et al. [14] described how to use *Identification* and *Fragment Offset* fields to carry hidden data between intermediate nodes but under condition that the packet is not fragmented. Additionally, in selected packet reserved flag is used to mark packet so that the receiver can distinguish between real and covert fragments. Murdoch et al. [4] proposed transmitting hidden information by modulating the size of the fragments to match the hidden data inserted into *Fragment Offset* field. Ahsan and Kundur [12] proposed steganographic method that use IP fragmentation fields. It utilizes high eight bits of the *Identification* to transmit covert data and the low eight bits are generated randomly. The same authors in [17] described a method that uses DF flag as a covert data carrier. If the sender knows the correct MTU for the end-to-end path to the receiver and issues packets which size is less than MTU then DF can be set to arbitrary values.

For IPv6 protocol Lucena et al. [15] identified four network steganographic methods based on *Fragment header extension*. Two methods use reserved fields to carry steganogram and one next header field. Fourth steganographic method is based on fake fragments insertion. In this case all fields of the fragment header may be used for covert communication. To avoid having inserted fragment included in the reassembly process of the original IP packet, authors propose two solutions: first is based on inserting an invalid value in *Identification* field in *Fragment extension header*, thus the receiver will drop such fragment, second—inserting overlapping *Fragment Offset* value that causes data to be overwritten during reassembly. Fake fragments carry hidden data only in certain header fields.

4 Proposed methods: communication scenarios, functioning and detection

Every steganographic method should be analyzed in terms of steganographic bandwidth and risk of hidden communication disclosure. Steganographic bandwidth may be expressed by means of RBR (Raw Bit Rate), which is defined as a total number of steganogram bits transmitted during one time unit [bit/s] or equivalently by PRBR (Packet Raw Bit Rate) which is defined as a total number of steganogram bits transmitted in single packet used during the hidden communication process [bit/packet]. Some steganographic methods are trivial to detect (e.g. those which simply modifies header fields) but for others the steganalysis may be harder to perform. Thus, for each proposed steganographic solution potential detection methods must be analyzed.

In general, there are four communication scenarios possible for network steganographic exchange. The first scenario (1) in Fig. 4, is most common: the sender, who is also a Steganogram Sender (SS) and the receiver, who is also a Steganogram Receiver (SR) establish a connection while

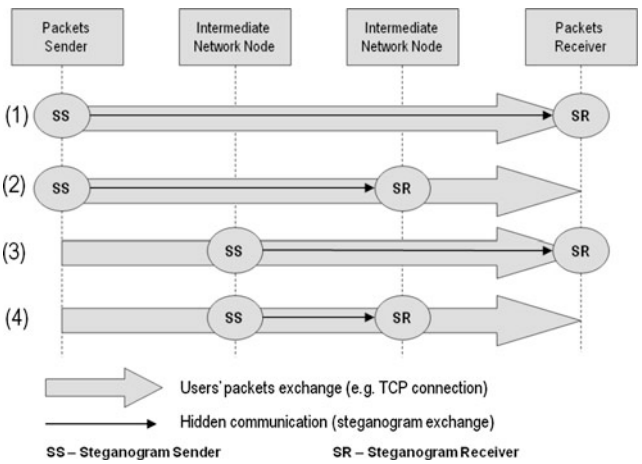


Fig. 4 Hidden communication scenarios

simultaneously exchanging steganograms. In the next three scenarios (marked 2–4 in Fig. 4) only a part of the end-to-end path is used for hidden communication as a result of actions undertaken by intermediate nodes; the sender and/or receiver are, in principle, unaware of the steganographic data exchange.

Hidden communication scenarios presented above differ in steganalysis, in particular, the scenario 4 is harder to detect, because the network node which analyses traffic for hidden communication called warden [20] is usually placed at the edge of source or destination endpoints (sub)network.

4.1 IP fragmentation

For IP fragmentation mechanism we propose new steganographic method (F1) and two enhancements of the previously proposed ones (F2 and F3). Moreover, we also show how IP fragmentation simplifies usage of existing steganographic methods that require transmitter-receiver synchronization (F4–F6). Steganographic methods that may be used for IP Fragmentation can be classified as presented in Fig. 5.

Each of presented methods may be utilized for IPv4 and IPv6 protocols for each scenario from Fig. 4. However, for IPv4 fragmentation, fragments reassembly may be performed by intermediate nodes as well as by the sender and/or receiver. This may limit the steganogram exchange only to the fragmenting and assembling nodes. For IPv6 there is no such limitation.

4.1.1 Steganographic method F1

In this method SS (Steganogram Sender) must be the source of the fragmentation. SS inserts single bit of hidden data by dividing original IP packet into the predefined number of fragments. For example, if the number of fragments is even then it means that binary "0" is transmitted and in other case binary "1" (Fig. 6).

After reception of the fragments SR uses the number of the fragments of each received IP packet to determine what hidden data was sent.

Potential steganographic bandwidth for this method is PRBR = 1 bit/packet.

Detection of this method may be hard to perform. Statistical steganalysis based on number of fragments can be performed to detect irregularities in number of the fragments. The best method to make hidden communication unavailable is to reassembly original IP packet in the intermediate node responsible for detecting steganographic communication (warden [20]), then refragment it randomly and send to the receiver.

After reception of the fragments SR uses the number of the fragments of each received IP packet to determine what hidden data was sent.

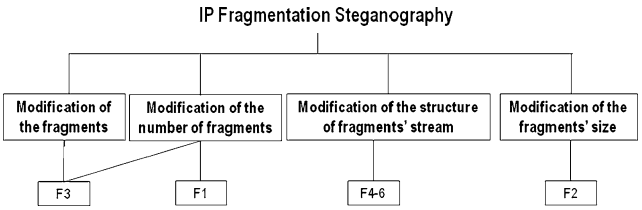


Fig. 5 Classification of IP Fragmentation steganographic methods

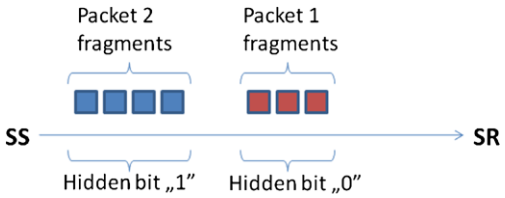


Fig. 6 F1 steganographic method example

Potential steganographic bandwidth for this method is PRBR = 1 bit/packet.

Detection of this method may be hard to perform. Statistical steganalysis based on number of fragments can be performed to detect irregularities in number of the fragments. The best method to make hidden communication unavailable is to reassembly original IP packet in the intermediate node responsible for detecting steganographic communication (warden [20]), then refragment it randomly and send to the receiver.

4.1.2 Steganographic method F2

The main idea of this method is to divide a packet into fragments and insert hidden information by modulating the values that are inserted into *Fragment Offset* field. As mentioned in Sect. 3, Murdoch et al. [4] proposed inserting steganogram directly into *Fragment Offset* field and modulate the size of the fragment to match this value. Such approach can cause high irregularities in fragments sizes which may be easily detected. We propose enhancement of this method which has lower steganographic bandwidth but is harder to detect.

F2 method works as follows. SS must be the source of the fragmentation. SS inserts single bit of hidden data by intentionally modulating the size of each fragment of the original packet in order to obtain fixed values in *Fragment Offset* field. For example, even offset means transmitting binary "1", odd offset—binary "0". Similar method may be used with total length of the packet as the sum of the digits of packet size may be modulated to be even or odd.

"Steganographic" fragmentation of the exemplary IP packet which was introduced in Table 1 is presented in Table 2.

After successful reception of the fragments SR extracts hidden data based on the values from *Fragment Offset* field.

Table 2 F2 steganographic method example

IP Fragments						
Seq.	Identifier	Total length	DF	MF	Fragment offset	Hidden data
0–0	345	1300	0	1	0	–
0–1	345	1340	0	1	160	1
0–2	345	1340	0	1	325	0
0–3	345	1220	0	0	490	1

Steganographic bandwidth for this method is $PRBR = N_F - 1$ [bit/packet], where N_F denotes number of fragments of the packet.

Steganalysis in case of F2 is harder than in case of method proposed by Murdoch, but hidden communication still can be uncovered, because usually all the fragments except last one have equal sizes (see Table 1). Thus, if there are any irregularities in fragments sizes, then steganographic communication may be uncovered. However, this method may be further improved, so the detection is more difficult to perform. We may influence the size of the fragments in such a manner that all fragments except last one would have the same length and the value in *Fragment Offset* field in last fragment is modulated to achieve even or odd value. In this case the hidden communication may not be detected at all as this fragmented packet will be similar to other ones.

Steganographic bandwidth for this improved method will be lower than for above method and will be equal PRBR = 1 bit/packet.

Detection of this method may be hard to perform. Statistical steganalysis based on fragments sizes can be performed to detect irregularities. The best method to make the hidden communication unavailable is the same as in case of method F1.

4.1.3 Steganographic method F3

Proposed method is enhancement of Lucena et al. [15] work for IPv6 fragmentation where they proposed to generate fake fragments. As mentioned in Sect. 3 two solutions to distinguish fake fragments from the legitimate were presented—first is based on inserting an invalid value in *Identification* field in *Fragment extension* header, second—inserting overlapping *Fragment Offset* value that causes data to be overwritten during reassembly. Fake fragments carry hidden data only in certain header fields. However, described methods may be easy to uncover because the warden can monitor all the fragments sent and determine potential anomalies like overlapping offsets or single, unrelated fragments. Our proposition is to use legitimate fragment with steganogram inserted into payload for higher steganographic bandwidth and harder detection.

F3 method works as follows. SS must be the source of the fragmentation. SS while dividing the packet, inserts steganogram instead of inserting user data into the payload of selected fragment. The problem with such approach is to properly mark fragments used for hidden communication so the receiver can extract it in a way that will not interfere with reassembly process. We propose the following procedure to make the selected fragments distinguishable from others yet hard to detect. Let us assume that sender and receiver share secret Steg-Key (*SK*). For each fragment chosen for steganographic communication the following hash function (H) is used to calculate Identifying Sequence (*IS*):

$$IS = H(SK||Fragment\ Offset||Identification) \tag{4.1}$$

where *Fragment Offset* and *Identification* denote values from these IP fragment header fields and || bits concatenation function. For every fragment used for hidden communications the resulting *IS* will have different value due to the changing values in *Fragment Offset*. All *IS* bits or only selected ones are distributed across payload field in predefined manner. Thus, for each fragment the receiver based on *SK* and values from the IP header can calculate appropriate *IS* and checks if it contains steganogram or user data. If the verification is successful then the rest of the payload is considered as hidden data and extracted. Then SR does not utilize this fragment in reassembly process of original IP packet.

Steganographic bandwidth for this method may be expressed as

$$PRBR = N_F \cdot F_S \text{ [bits/packet]} \tag{4.2}$$

where N_F denotes number of fragments and F_S the size of the fragment payload.

Figure 7 illustrates example of the proposed steganographic method. IP packet with ID 345 is divided into four fragments (F1–F4). Fragment F2 is used for steganographic purposes, so inside its payload steganogram is inserted together with correct *IS*. Values in *Fragment Offset* and *Identification* remain the same as in other legitimate fragments. While reassembling original packet, receiver merges payloads P1, P2 and P3, omits fragment F2 and use it only to extract steganogram.

Method F3 is hard to detect because legitimate fragments are used as hidden data carriers. The best method to make the hidden communication unavailable is the same as in case of methods F1 and F2.

4.1.4 Steganographic methods F4–F6

Fragments that are created during fragmentation process may be treated as numbered stream of the packets, because *Identification* and *Fragment Offset* fields uniquely identify

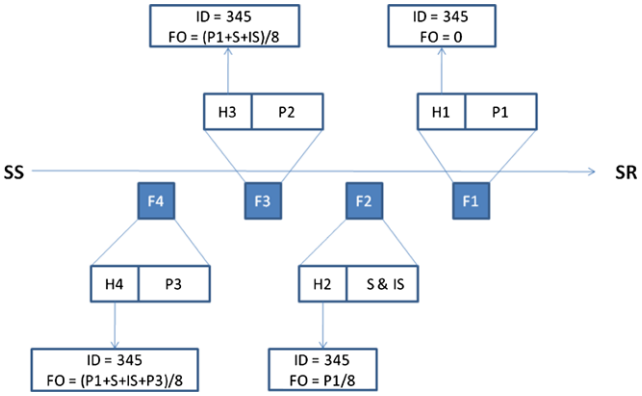


Fig. 7 F3 steganographic method example (H—header, P—payload)

each piece and allow their correct placement during reassembly process. That is why, for IP fragmentation mechanism existing network steganographic methods proposed for such numbered data may be utilized. These are: intentional changing sequence of the packets, modifying inter-packet delays and introducing intentional losses. What is common to these methods is sender-receiver synchronization requirement. We show that for fragmentation process this requirement is not longer valid, so the deployment of these methods is easier—synchronization is not needed because one packet fragmentation may be treated as one synchronization period. The lack of requirement for sender-receiver synchronization makes these methods easier to implement.

Intentional changing sequence of the packets for transmitting covert data was proposed in [16, 17]. These methods may be applied to IP Fragmentation (F4), especially if the number of fragments is high by sending fragments in a predefined fashion. In Table 1 four fragments were created and *Fragment Offset* values decide of their sequence. So sending fragments in the sequence 0, 1, 2, 3 may be interpreted as binary '1' and the reverse order as binary '0'.

In general, PRBR of such method depends on number of fragments (n) and may be expressed as

$$PRBR = \log_2 n! \text{ [bits/packet]} \tag{4.3}$$

Network steganography method that modifies inter-packet delay was presented in [18]. Such approach may be successfully utilized for IP fragmentation (F5) and for example work as follows. During fragmentation of one IP packet, fragments are generated at one rate (it may mean sending hidden binary '1') and while dividing another one with different rate (e.g. it means sending binary '0').

In general, PRBR of such method depends on number of packets generation rates (h) and may be expressed as

$$PRBR = \log_2 h \text{ [bits/packet]} \tag{4.4}$$

Method proposed by Servetto et al. [19] which introduces intentional losses in numbered stream of packets may be also utilized. This solution is implemented as skipping one sequence number at the sender so no user data is lost. Loss that occurred during fixed time interval is equal to sending one steganogram bit. This method is called *phantom packets*. The same method can be applied to IP fragmentation (F6). While sender generates fragments, it skips one *Fragment Offset* value and inserts the user data into next fragment. If the loss of fragment occurs it means sending binary '1' and if it is not present, binary '0'. To work correctly this method requires modified receiver which can reassembly original IP packet even though not all fragments reached the receiver. We named this modified version of existing method as *phantom fragments*.

For presented method steganographic bandwidth equals PRBR = 1 bit/packet.

4.2 PMTUD

The main idea for exchanging hidden data with PMTUD is simple—it involves sender to utilize probe messages to carry steganogram and invoke sending intentional fake ICMP messages by receiver. Detailed hidden information procedure is suitable for both IPv4 and IPv6 and is possible for all scenarios from Fig. 4.

Proposed steganographic method works as follows. SS knows from previous interactions with SR what the correct MTU for their communication path is. When SS wants to send steganogram then it sends a probe message that contains steganogram inserted into packet payload. The size of the packet is set to the maximum MTU allowed for path between SS and SR, thus SS is certain that this packet will reach the receiver.

To make the selected packet for steganographic purposes distinguishable from other yet hard to detect we propose similar procedure as it was presented for IP fragmentation mechanism. If we assume that sender and receiver share secret Steg-Key (SK), then for each packet chosen for hidden communication a hash function (H) is used to calculate Identifying Sequence (IS):

$$IS = H(SK||Identification||CB) \tag{4.5}$$

where *Identification* denotes values from that IP header field, CB is *Control Bit* and || is bits concatenation function. *Control Bit* is used to inform the receiver whether it should sent more fake ICMP messages or not ($CB = 1$ send more ICMP, $CB = 0$ do not send more ICMP). For every IP packet used for hidden communications the resulting IS will be different due to the changing values from *Identification* field. All IS bits or only selected ones are distributed across payload field in predefined manner.

After a probe message reaches the receiver, he/she calculates two ISs (one for $CB = 1$, second for $CB = 0$) based

on *SK* and value from the IP header and checks if it contains steganogram or user data. When steganogram is detected it is extracted from the packet payload. If *IS* calculation indicates that $CB = 1$ then receiver intentionally send ICMP message that indicate that the MTU of the path must be decreased and thus sender is obligated to send smaller probe message (which will also contain steganogram). In fake ICMP message source IP address must be spoofed to avoid trivial detection. In the payload of ICMP message IP header of the original packet and 64 bits of original data are present. Receiver must mark ICMP message to allow sender to distinguish real ICMP from fake one. To achieve this we propose to modify the TTL (Time To Live) field of the original IP packet header from the ICMP payload and change the *Total Length* and *Header Checksum* values accordingly. TTL is the only field in IP header (if IP fragmentation is not used) which may be modified during traversing the network. Thus comparing original packet sent with returned in ICMP message will not result in easy hidden communication detection. There are many possibilities of TTL modifications and, in particular, they include setting TTL to prearranged value or to even/odd one. Functioning of the described above steganographic method is also illustrated in Fig. 8. In this example, during the PMTUD exchange, about 3 kB of steganogram was sent from SS to SR.

For proposed method steganographic bandwidth can be expressed with as:

$$RBR_{PMTUD} = \frac{\sum_1^n P_n}{T} \text{ [bits/s]} \tag{4.6}$$

where n denotes number of probes sent from sender to receiver, P_n probe payload size and T connection duration.

During PMTUD exchange all probes messages may be used for steganographic purposes but in this case detection may be easier to perform. Because it is assumed that the earlier probes failed to reach the receiver, next ones should carry fragment of the same data. Thus, comparing each probe message sent with the first one issued may be used to detect steganograms. Only in case when the first probe is used to carry steganogram above steganographic method is hard to detect but then the steganographic bandwidth is limited.

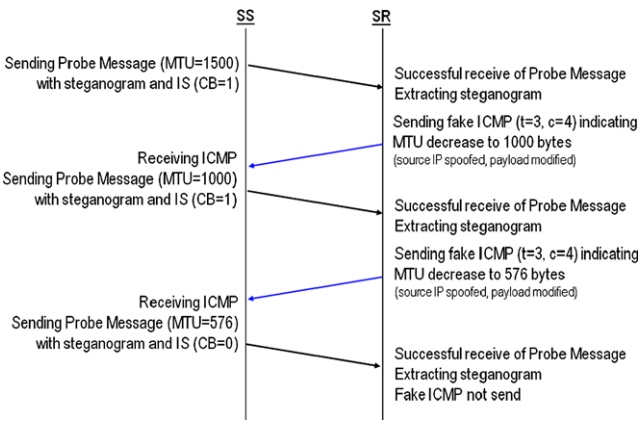


Fig. 8 PMTUD steganographic method

4.3 PLPMTUD

In PLPMTUD probes messages are validated at the transport layer and correct MTU is learned by starting with packets which size is relatively small and when they get through they proceed with progressively larger ones. The isolated loss of a probe packet is treated as an indication of an MTU limit and transport layer protocol is permitted to retransmit any missing data. Thus, steganographic method described for PMTUD is not applicable. Nevertheless, other possibilities for hidden communication may be utilized. One of them is RSTEG (Retransmission Steganography) method which is presented by authors in details in [21] and uses intentional retransmissions to sent steganograms. RSTEG main idea is to not acknowledge a successfully received packet in order to intentionally invoke retransmission. The retransmitted packet carries a steganogram instead of user data in the payload field. RSTEG may be used for IPv4 and IPv6 in all hidden communication scenarios from Fig. 4.

For PLPMTUD using RSTEG works as follows. SS knows from previous interactions with SR what the correct MTU for their communication path is. When the connection starts, SS sends probe message with prearranged MTU. After successfully receiving the packet, the receiver intentionally does not issue an acknowledgment message. In a normal situation, a sender is obligated to retransmit the lost packet when the timeframe within which packet acknowledgement should have been received expires. In the context of RSTEG, a sender replaces original payload with a steganogram instead of sending the same packet again. When the retransmitted packet reaches the receiver, he/she can then extract hidden information.

The detection method is similar to one presented for PMTUD and is based on comparing probes messages payload during MTU learning process.

5 Experimental evaluation for IP fragmentation steganography

To evaluate the steganographic bandwidth for methods presented in Sect. 4.1 for IP fragmentation, prototype application called *StegFrag* was implemented. It encloses steganographic methods F1–F5 except method F6 as it may interfere with other methods and decrease achievable steganographic bandwidth. As stated in Sect. 4.1 some of presented method may be easily detected if used alone. In *StegFrag* chosen steganographic methods were implemented to achieve

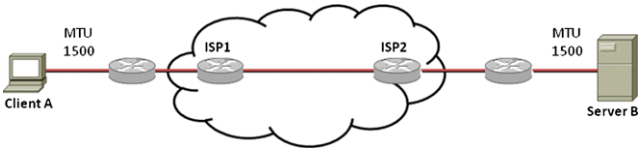


Fig. 9 Experimental IP fragmentation steganography setup

Table 3 Experimental connections characteristic features

Measure	Average	Standard deviation
Number of fragments	219698	142.7
Connection time [s]	792.6	7.23

Table 4 Chosen PRBR for steganographic methods used in experiment

Steganographic methods	F1	F2	F3	F4	F5
PRBR [bit/packet]	1	0.001	320	6	1

higher steganographic bandwidth yet limit the risk of detection.

Experimental client-server scenario was set up which is presented in Fig. 9.

In presented scenario, client A requests and downloads a 100 MB file from the server B. Both the sender and the receiver are on LAN, thus their MTU is 1500 bytes. Server B intentionally sends fragmented packets with MTU equals 740 bytes, thus each original 1500 bytes packet is divided into three fragments (740, 740 and 60 bytes respectively). The experiment was repeated 10 times and average results of these connections are presented in Table 3.

For each steganographic method implemented in *StegFrag*, following PRBR was used as presented in Table 4. For F3 method, if the fake fragment is generated it is always the third (with highest *Fragment Offset*) and its payload is used to carry steganogram (40 bytes, IS included).

Above mentioned steganographic methods were implemented to limit the risk of disclosure. Thus methods F1 and F3 depend on each other. Each original IP packet is fragmented into three pieces so without further modifications in functioning using method F1 is impossible. That is why when there is binary '0' in hidden data to send then the third fragment is assumed to be fake inserted one. Thus, for method F1 the number of "real" fragments sent is two—this allows to transmit additional bit of steganogram per one original IP packet. In other case three "real" fragments are present and method F3 is not used.

F5 is implemented as follows. Every 1000 packets there is slight change in next packets sizes to set *Fragment Offset* field in last fragment to even/odd value. This allows to embed one steganogram bit per 1000 original IP packets. Such rare changes were deliberately set to limit the risk of detection.

For example when there is binary '0' in hidden data to be sent, steganographic bandwidth provided by methods F1–F5 is a sum of each method steganographic bandwidth. When binary '1' must be sent steganographic bandwidth is much lower because it consists only of steganographic bandwidths from methods F1, F2, F4 and F5.

When fragments reach the at client A, it is extracted in predefined manner—presence of hidden bits from method F3 is checked first and extracted, then hidden bit from F1 and methods F4, F5. Last steganogram bit is extracted if it is possible from method F2.

The actual algorithms in pseudocode for embedding steganogram at server B and extracting it at client A are presented below.

Embedding algorithm at server B:

```
For each Original_IP_packet
{
If Steg_bit = 0 then
  {
  F3_Insert(Fake_fragment3);
  Generate(IS);
  InsertBits(IS) → Fake_fragment3;
  Steg_bit = NextStegBit;
    While Free_payload(Fake_fragment3) <> 0
      {
       InsertBits(Steg_bit) → Fake_fragment3;
       Steg_bit = NextStegBit;
      }
  }

  Steg_bit = NextStegBit;
  F4_SetFragSequence;

  Steg_bit = NextStegBit;
  F5_SetFragDelay;

  Steg_bit = NextStegBit;
  If NoOfPackets mod 1000 = 0 then
  {
  If Steg_bit = 0 then
    ChangeFragmentSize(Even_Last_FragmentOffset)
    else
ChangeFragmentSize(Odd_Last_FragmentOffset)
    }
  }
```

Extraction algorithm at client A:

```
For each IncomingFragment
{
If CheckIS(Fragment3) = 1 then
  {
  Insert(Fragment3) → ExtractedStegBits;
  FragNumEven → 1;
```

```
    }
  If FragNumEven = 1 then ExtractedStegBits ←
Insert(1);
  ExtractedStegBits ← Insert(F4_CheckFragSequence);
  ExtractedStegBits ← Insert(F5_CheckFragDelay);
    If NoOfPackets mod 1000 = 0 then
      {
       If Even(Last_FragmentOffset) = 1 then
ExtractedStegBits ← Insert(0)
       else ExtractedStegBits ← Insert(1);
      }
  }
```

The following experimental results were obtained (Table 5).

During the 100 MB file transfer, 1.54 MB of steganogram, on average, was secretly transferred during the single connection. It must be noted however, that usable bandwidth due to fake fragments detection with IS sequence is slightly lower and is about 1.25 MB. This is large amount of secret data sent during nearly 13.5 minutes connection with limited risk of detection. Figures 10 and 11 illustrate PRBR and cumulative total steganogram sent during the fragment of the exemplary connection respectively.

Due to F3 method functioning and its PRBR, average connection PRBR is changing dynamically during the connection (Fig. 10). The same cause is responsible for the shape of the total steganogram curve (Fig. 11).

In Table 6 fraction of the total steganographic bandwidth for each of implemented methods is presented. It can be seen that about 95% of total steganographic bandwidth is provided by method F3, which is not surprising considering their PRBRs.

6 Conclusions

In this paper we presented potential steganographic methods that can be used for mechanisms for handling oversized IP packets: IP fragmentation, PMTUD and PLPMTUD. In particular, we propose two new steganographic methods, three extensions of existing ones and we show how IP fragmentation simplifies utilizing steganographic solutions which require transmitter-receiver synchronization.

Proposed steganographic methods are characterized by different steganographic bandwidth and detection possibilities, thus they can have various impact on network security. Knowledge of these information hiding procedures can now be utilized to develop and implement countermeasures for network traffic monitoring. This may limit the risk of confidential information leakage or other threats caused by covert communication.

Table 5 Experimental results

Measure	Average	Standard deviation
Total amount of covert data sent [bits]	12302478	7991.62
RBR [bit/s]	15517.5	141.9

Table 6 Steganographic bandwidth fraction [%] per steganographic method

	F1	F2	F3	F4	F5
Steganographic bandwidth fraction [%]	0.6	0.0006	95.23	3.57	0.6

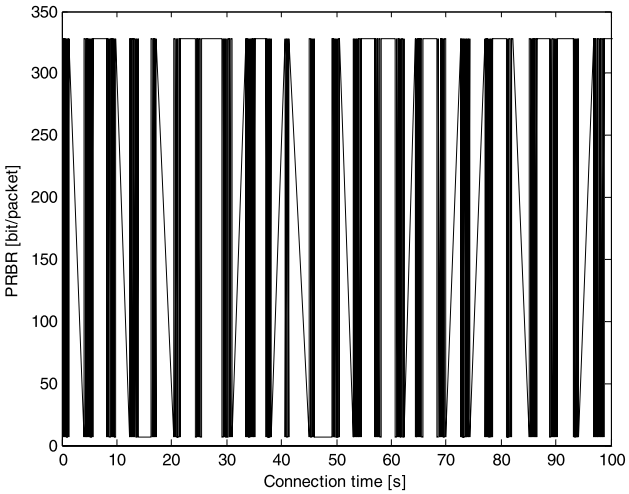


Fig. 10 PRBR for fragment of the exemplary connection

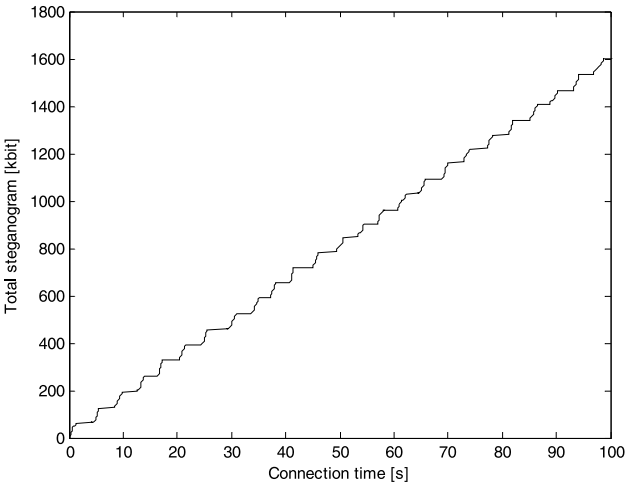


Fig. 11 Cumulative total steganogram sent during the fragment of the exemplary connection

Experimental results for IP fragmentation achieved with prototype application showed that, while downloading 100 MB file, in about 13 minutes connection, one is able to send more than 1 MB of hidden data with limited risk of detection. These results urge to develop and deploy suitable steganalysis tools in every network that should be secure.

Acknowledgements This research was partially supported by the Ministry of Science and Higher Education, Poland (Grant No. 0716/BT02/2009/37).

References

1. Rowland, C. (1997). Covert channels in the TCP/IP protocol suite, first Monday. *Peer Reviewed Journal on the Internet*, July 1997.
2. Zander, S., Armitage, G., & Branch, P. (2007). A survey of covert channels and countermeasures in computer network protocols. *IEEE Communications Surveys & Tutorials*, *9*(3), 44–57. ISSN: 1553-877X.
3. Petitcolas, F., Anderson, R., & Kuhn, M. (1999). Information hiding—a survey. *IEEE Special Issue on Protection of Multimedia Content*, July 1999.
4. Murdoch, S. J., & Lewis, S. (2005). Embedding covert channels into TCP/IP. *Information Hiding*, 247–260.
5. Postel, J. (1981). Internet protocol. *IETF RFC* 791, September 1981.
6. Mogul, J., & Deering, S. (1990). Path MTU discovery. *IETF RFC* 1191, November 1990.
7. McCann, J., Mogul, J., & Deering, S. (1996). Path MTU discovery for IP version 6. *IETF RFC* 1981, August 1996.
8. Mathis, M., & Heffner, J. (2007). Packetization layer path MTU discovery. *IETF RFC* 4821, March 2007.
9. Deering, S., & Hinden, R. (1998). Internet protocol, version 6 (IPv6) specification. *IETF RFC* 2460, December 1998.
10. Conta, A., Deering, S., & Gupta, M. (2006). Internet control message protocol (ICMPv6) for the Internet protocol version 6 (IPv6) specification. *IETF RFC* 4443, March 2006.
11. Lahey, K. (2000). TCP problems with path MTU discovery. *IETF RFC* 2923, September 2000.
12. Ahsan, K., & Kundur, D. (2002). Practical data hiding in TCP/IP. In *Proc. ACM wksp. multimedia security*, December 2002.
13. Mazurczyk, W., & Szczypiorski, K. (2009). Steganography in handling oversized IP packets. In *Proc. of first international workshop on network steganography (IWNS 2009)*, November 18–20, 2009, Wuhan, China.
14. Cauich, E., Gomez Cardenas, R., & Watanabe, R. (2005). Data hiding in identification and offset IP fields. In *Proc. 5th int'l. school and symp. advanced distributed systems (ISSADS)*, January 2005 (pp. 118–125).
15. Lucena, N. B., Lewandowski, G., & Chapin, S. J. (2005). Covert channels in IPv6. In *Proc. privacy enhancing technologies (PET)*, May 2005 (pp. 147–166).
16. Chakinala, R., Kumarasubramaniam, A., Manokaran, R., Noubir, G., Pandu Rangan, C., & Sundaram, R. (2006). Steganographic communication in ordered channels, materiały. In *Information hiding workshop, IHW 2006, LNCS 4437/2007* (pp. 42–57).
17. Kundur, D., & Ahsan, K. (2003). Practical Internet steganography: data hiding in IP. In *Proc. of Texas workshop: security of information systems*, April 2003.
18. Girling, C. G. (1987). Covert channels in LAN's. *IEEE Transactions on Software Engineering*, *SE-13*(2), 292–296.
19. Servetto, S. D., & Vetterli, M. (2001). Communication using phantoms: covert channels in the Internet. In *Proc. IEEE international symposium information theory (ISIT)*, June 2001.
20. Fisk, G., Fisk, M., Papadopoulos, C., & Neil, J. (2002). Eliminating steganography in Internet traffic with active wardens. In *Lecture notes in computer science: Vol. 2578. Proc. 5th international workshop on information hiding* (pp. 18–35). Berlin: Springer.
21. Mazurczyk, W., Smolarczyk, S., & Szczypiorski, K. (2009). Hiding information in retransmissions. In *Computing research repository (CoRR)*. arXiv:0905.0363 [abs].

Wojciech Mazurczyk holds an M.Sc. (2004) and a Ph.D. (2009) in telecommunications from the Faculty of Electronics and Information Technology, Warsaw University of Technology (WUT, Poland) and is now an Assistant Professor at WUT and the author of over 40 scientific papers and over 25 invited talks on information security and telecommunications. His main research interests are information hiding techniques, network security and multimedia services, and he is also a research leader of the Network Security Group at WUT (secgroup.pl). Personal website: http://mazurczyk.com.

Krzysztof Szczypiorski holds an M.Sc. (1997) and a Ph.D. (2007) in telecommunications both with honours from the Faculty of Electronics and Information Technology, Warsaw University of Technology (WUT), and is an Assistant Professor at WUT. He is the founder and head of the International Telecommunication Union Internet Training Centre (ITU-ITC), established in 2003. He is also a research leader of the Network Security Group at WUT (secgroup.pl). His research interests include network security, steganography and wireless networks. He is the author or co-author of over 110 publications including 65 papers, two patent applications, and 35 invited talks.

PadSteg: Introducing Inter-Protocol Steganography

Bartosz Jankowski, Wojciech Mazurczyk, Krzysztof Szczypiorski

Abstract — **Hiding information in network traffic may lead to leakage of confidential information. In this paper we introduce a new steganographic system: the *PadSteg* (Padding Steganography). To authors' best knowledge it is the first information hiding solution which represents *inter-protocol steganography* i.e. usage of relation between two or more protocols from the TCP/IP stack to enable secret communication. *PadSteg* utilizes ARP and TCP protocols together with an *Etherleak* vulnerability (improper Ethernet frame padding) to facilitate secret communication for hidden groups in LANs (Local Area Networks). Basing on real network traces we confirm that *PadSteg* is feasible in today's networks and we estimate what steganographic bandwidth is achievable while limiting the chance of disclosure. We also point at possible countermeasures against *PadSteg*.**

Keywords: steganography, ARP, frame padding, Etherleak

I. INTRODUCTION

Network steganography is currently seen as a rising threat to network security. Contrary to typical steganographic methods which utilize digital media (pictures, audio and video files) as a cover for hidden data (steganogram), network steganography utilizes communication protocols' control elements and their basic intrinsic functionality. As a result, such methods may be harder to detect and eliminate.

In order to minimize the potential threat to public security, identification of such methods is important as is the development of effective detection (steganalysis) methods. This requires both an in-depth understanding of the functionality of network protocols and the ways in which it can be used for steganography. Many methods had been proposed and analyzed so far – for the detailed review see Zander et al. [2] or Petitcolas et al. [3].

Typical network steganography method uses modification of a single network protocol. The classification of so such methods was introduced by Mazurczyk et al. in [15]. The protocol modification may be applied to the PDU (Protocol Data Unit) [1], [4], [5], time relations between exchanged PDUs [6], or both [14] (hybrid methods). This kind of network steganography can be called *intra-protocol* steganography.

As far as the authors are aware, *PadSteg* (Padding Steganography), presented in this paper, is the first steganographic system that utilizes what we have defined as *inter-protocol* steganography i.e. usage of relation between two or more different network protocols to enable secret communication – *PadSteg* utilizes Ethernet (IEEE 802.3), ARP, TCP and other protocols. This paper is an extension of the work introduced in [16].

Thus, classification introduced above may be further expanded to incorporate *inter-protocol* steganographic methods (Fig. 1).

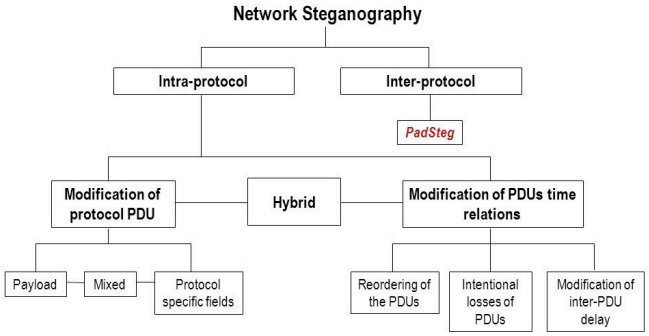


Figure 1. Network steganography classification

ARP (Address Resolution Protocol) [10] is a simple protocol which operates between the data link and network layers of the OSI (Open Systems Interconnection) model. In IP networks it is used mainly to determine the hardware MAC (Media Access Control) address when only a network protocol address (IP address) is known. ARP is vital for proper functioning of any switched LAN (Local Area Network) although it can raise security concerns e.g. it may be used to launch an ARP Poisoning attack.

In Ethernet, frame length is limited to a minimum of 64 octets, due to the CSMA/CD (Carrier Sense Multiple Access/ Collision Detection) mechanism, and a maximum of 1500 octets. Therefore, any frames whose length is less than 64 octets have to be padded with additional data. The minimal size of an Ethernet data field is 46 octets and can be filled with data originating from any upper layer protocol, without encapsulation via the LLC (Link Layer Control), because LLC (with its 8 octets header) is very rarely utilized in 802.3 networks.

However, due to ambiguous standardization (RFC 894 and RFC 1042), implementations of padding mechanism in current NICs (Network Interface Cards) drivers vary. Moreover, some drivers handle frame padding incorrectly and fail to fill it with zeros. As a result of memory leakage, Ethernet frame padding may contain portions of kernel memory. This vulnerability is discussed in *Atstake* report and is called *Etherleak* [9]. Data inserted in padding by *Etherleak* is considered unlikely to contain any valuable information; therefore it does not pose serious threat to network security as such. However, it creates a perfect candidate for a carrier of the steganograms, thus it may be used to compromise network defenses. Utilization of padding in Ethernet frames for steganographic purposes was originally proposed by Wolf [13]. If every frame has padding set to zeros (as stated

in standard), its usage will be easy to detect. With the aid of *Etherleak,* this information hiding scheme may become feasible as it will be hard to distinguish frames affected by *Etherleak* from those with steganogram.

In this paper we propose a new steganographic system *PadSteg*, which can be used in LANs and utilizes ARP and other protocols (like TCP or ICMP) together with an *Etherleak* vulnerability. We conduct a feasibility study for this information hiding system, taking into account the nature of todays' networks. We also suggest possible countermeasures against *PadSteg*.

The rest of the paper is structured as follows. Section 2 describes the *Etherleak* vulnerability and related work with regard to the application of padding for steganographic purposes. Section 3 includes a description of *PadSteg* components. Section 4 presents experimental results for real-life LAN traffic which permit for an evaluation of feasibility of the proposed solution. Section 5 discusses possible methods of detection and/or elimination of the proposed information hiding system. Finally, Section 6 concludes our work.

II. RELATED WORK

A. The Etherleak vulnerability

The aforementioned ambiguities within the standardization cause differences in implementation of the padding in Ethernet frames. Some systems have an implemented padding operation inside the NIC hardware (so called *auto padding*), others have it in the software device drivers or even in a separate layer 2 stack.

In the *Etherleak* report Arkin and Anderson [9] presented in details an Ethernet frame padding information leakage problem. They also listed almost 50 device drivers from Linux 2.4.18 kernel that are vulnerable.

Due to the inconsistency of padding content of short Ethernet frames (its bits should be set to zero but in many cases they are not), information hiding possibilities arise. That is why it is possible to use the padding bits as a carrier of steganograms.

Since Arkin and Anderson's report dates back to 2003, we performed an experiment in order to verify whether *Etherleak* is an issue in today's networks. The achieved results confirmed that many NICs are still vulnerable (see experimental results in Section 4).

B. Data hiding using padding

Padding can be found at any layer of the OSI RM, but typically it is exploited for covert communications only in the data link, network and transport layers.

Wolf in [13], proposed a steganographic method which utilizes padding of 802.3 frames. Its achievable steganographic bandwidth is up to 45 bytes/frame.

Fisk et al. [7] presented padding of the IP and TCP headers in the context of active wardens. Each of these fields offers up to 31 bits/packet for steganographic communication.

Padding of IPv6 packets for information hiding was described by Lucena et al. in [8] and offers a couple of channels with a steganographic bandwidth up to 256 bytes/packet.

III. IMPROPER ETHERNET FRAME PADDING IN REAL-LIFE NETWORKS

Real network traffic was captured to verify whether described in 2003 *Etherleak* vulnerability is still feasible in current LANs. It will also be used to evaluate the proposed in Section IV steganographic system – its steganographic bandwidth and detectability.

The experiment was conducted at the Institute of Telecommunications at Warsaw University of Technology between 15 and 19 of March 2010 (from Monday to Friday). It resulted in about 37 million packets captured, which corresponds, daily, to 7.43 million frames on average (with a standard deviation 1.2 million frames) – for details see Table 1. The traffic was captured with the aid of *Dumpcap* which is part of the *Wireshark* sniffer ver. 1.3.3 (www.wireshark.org). The sources of traffic were ordinary computer devices placed in several university laboratories and employees' ones but also peripherals, servers and network equipment. To analyze the captured traffic and calculate statistics *TShark* (which is also part of *Wireshark*) was utilized. Statistics were calculated per day, and average results are presented.

TABLE I. THE NUMBER OF CAPTURED FRAMES PER DAY

Date	*Monday*	*Tuesday*	*Wednesday*	*Thursday*	*Friday*
No. of frames	7,205,904	7,027,170	5,761,723	8,241,832	8,945,403

The captured traffic classification by upper layer protocol is presented in Fig. 2. Three quarters of the traffic was HTTP. Together with SSH, UDP and SSL protocols it sums up to about 93% of the traffic.

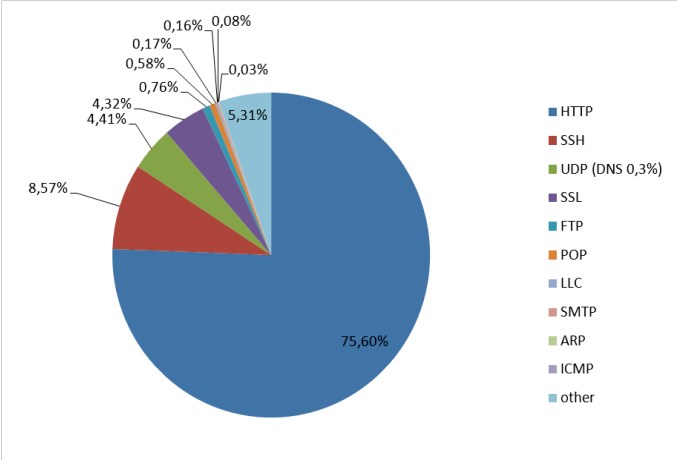


Figure 2. Captured traffic characteristics

Almost 22% (with a standard deviation of 7.7%) of all daily traffic had padding bits added (~8 million frames). It is obvious that not all of the frames were affected since padding is added only to small-sized packets.

Table 2 shows for which network protocols frames were mostly improperly padded.

TABLE II. UPPER LAYER PROTCOLS AFFECTED WITH ETHERNET FRAME IMPROPER PADDING IN EXPERIMENTAL DATA AND EXEMPLARY PID ASSIGMENT

Affected protocol	*TCP*	*ARP*	*ICMP*	*UDP*	*Others*
[%]	92.82	4.17	2.31	0.54	0.16
PID	1	2	3	4	-

However, it is important to note, that almost 22% of the padded frames experienced improper padding (~1.8 million frames). These frames were generated by about 15% of hosts in the inspected network (their NICs were produced among others by some US leading vendors). We considered Ethernet frame padding improper if the padding bits were not set to zeros.

TCP segments with an ACK flag set (which have no payload) result in frames that have to be padded, thus, it is no surprise that ~93% of improperly padded traffic is TCP. Nearly all of this traffic consists of ACK segments. Other frames that had improper padding were caused by ARP and ICMP messages – *Echo Request* and *Echo Reply* (~6.5%). It is also worth noting that there is also padding potential in UDP datagrams as UDP-based applications often generate small-sized frames (e.g. voice packets in IP telephony). However, padding was only present in 0.5% of all padded frames.

For *PadSteg* ARP protocol plays important role (see Section IV for details), thus our aim was also to find out ARP statistics i.e. what are the most frequently used ARP messages, what is their distribution and how many of them have improper padding. The results are presented in Fig. 3.

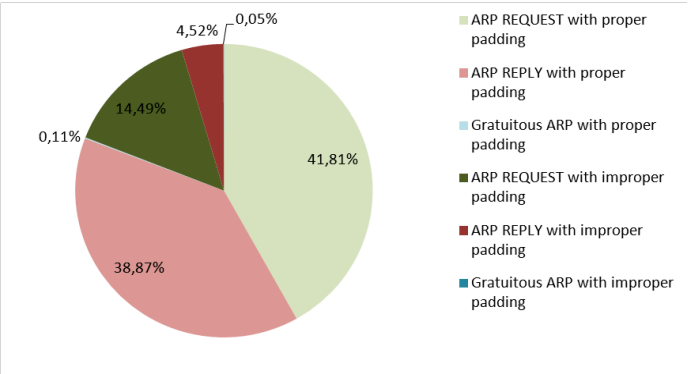


Figure 3. Captured ARP characteristics

Not surprisingly, the most frequently sent ARP messages were ARP Request (~56.3%) and Reply (~43.4%), while Gratuitous ARP messages are in minority (~0.2%). Out of all ARP messages almost 20% had improper padding.

IV. COMPONENTS OF THE PROPOSED STEGANOGRAPHIC SYSTEM

PadSteg enables secret communication in a hidden group in a LAN environment. In such group, each host willing to exchange steganograms should be able to locate and identify other hidden hosts. To provide this functionality certain mechanisms must be specified. In our proposal, ARP protocol, together with improper Ethernet frame padding are used to provide localization and identification of the members of a hidden group. To exchange steganograms improper Ethernet frame padding is utilized in frames that in upper layer use TCP, ARP or ICMP (or other network protocols that cause Ethernet frames to be padded). These protocols will be called carrier-protocols as they enable transfer of steganograms throughout the network.

Moreover, while the secret communication takes place, hidden nodes can switch between carrier-protocols to minimize the risk of disclosure. We called such mechanism *carrier-protocol hopping* and it will be described in details later.

In this section we first describe ARP protocol, and then we focus on proposed steganographic system operations.

A. Overview of ARP Protocol

ARP returns the layer 2 (data link) address for a given layer 3 address (network layer). This functionality is realized with two ARP messages: Request and Reply. The ARP header is presented in Fig. 4.

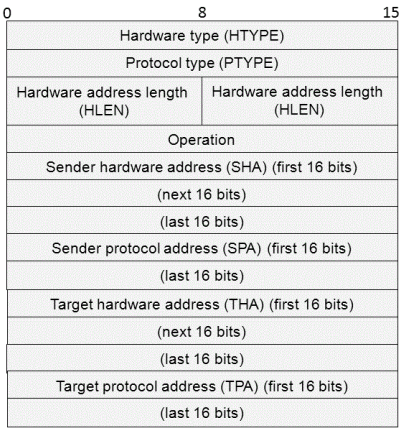


Figure 4. ARP header format

ARP header fields have the following functions:

- HTYPE (Hardware Type) – type of data link protocol used by sender (1 is inserted if it is Ethernet).
- PTYPE (Protocol Type) – type of network protocol in network layer (0800h is inserted if IP is used).
- HLEN (Hardware Length) – length of hardware address fields: SHA, THA (in bytes).

- PLEN (Protocol Length) – length of protocol address fields: SPA, THA (in bytes).
- OPER (Operation) – defines, whether the frame is an ARP REQUEST (1) or REPLY (2) message.
- SHA (Sender Hardware Address) – sender data link layer address (MAC address for Ethernet).
- SPA (Sender Protocol Address) – sender network layer address.
- THA (Target Hardware Address) – data link layer address of the target. This field contains zeros whenever a REQUEST ARP message is sent.
- TPA (Target Protocol Address) – network layer address of the target. This field contains zeros if REQUEST ARP message is sent.

An example of ARP communication with Request/Reply exchange, captured with the *Wireshark* sniffer (www.wireshark.org), is presented in Fig. 5. First, ARP Request is issued (1), which is used by the host with IP address 10.7.6.29 to ask other stations (by means of broadcast): 'Who has IP 10.7.56.47?'. In order to send a frame intended for everyone in a broadcast domain, Ethernet header destination address must be set to FF:FF:FF:FF:FF:FF (2). Next, host with IP address 10.7.56.47 replies directly to 10.7.6.29 using unicast ARP Reply (3) with its MAC address.

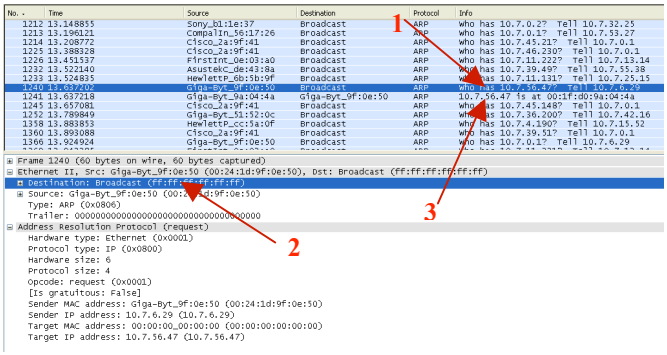


Figure 5. ARP exchange captured with *Wireshark*

Basing on the proposed description of ARP protocol, it can be concluded that ARP header is rather of fixed content and presents little possibilities for information hiding. One opportunity is to modulate address fields like it was proposed in [11] or [8]. However, this solution provides limited steganographic bandwidth if certain level of undetectability is to be achieved. Moreover, it may result in improper IP and MAC address advertisements which may make this method more prone to detection.

Thus, in the proposed steganographic system *PadSteg*, we utilize ARP Request messages, broadcasted throughout LAN, to make other members of the hidden group become aware of the presence of a new member.

B. Steganographic system operation

PadSteg is designed for LANs only because it utilizes improper Ethernet frame padding in Ethernet. It allows members of the hidden groups to secretly exchange data (Fig. 6).

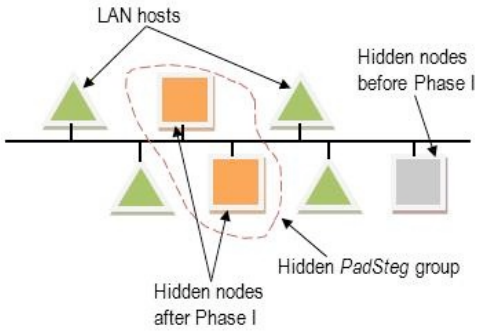


Figure 6. *PadSteg* hidden group

Every member from the hidden group is obligated to fill each short Ethernet frame it sends with non-zero padding to make detection harder – such node must mimic *Etherleak* vulnerability. *PadSteg* also uses protocols like ARP, TCP or ICMP to control hidden group and to transfer steganograms.

PadSteg operation can be split into two phases:

- Phase I: Advertisement of the hidden node and a carrier-protocol.
- Phase II: Hidden data exchange with optional carrier-protocol change.

Phase I

This phase is based on the exchange of ARP Request messages with improper Ethernet frame padding (Fig. 7).

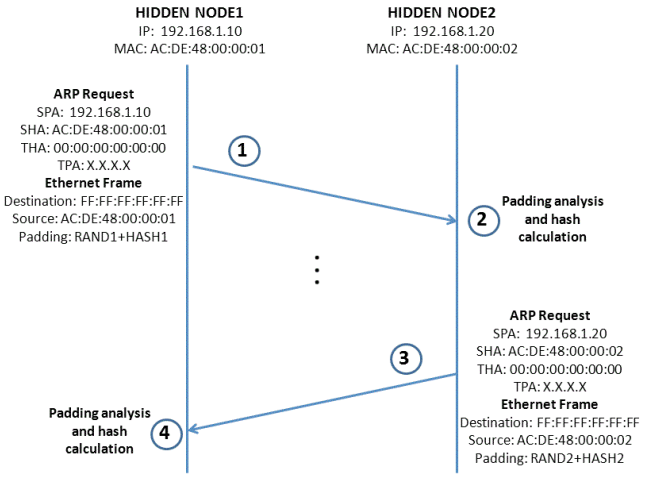


Figure 7. Hidden node and its carrier-protocol advertisement phase

Hidden node that wants to advertise itself to others in the group, broadcasts an ARP Request message (1) and inserts *advertising sequence* into the padding bits. It consists of: a random number *RD* (different from 0), and hash R_H which is calculated based on *RD*, carrier-protocol identifier *PID* and source MAC address (see eq. 4-1). Incorporating *RD* ensures that frame padding will be random. *PID* is an identifier of the upper layer carrier-protocol for the steganograms transfer

and may have been assigned exemplary values like in Table II. *PID* is used to advertise hidden node preference for the secret data transfer and may be used during steganograms exchange by carrier-protocol hopping mechanism.

An example of the padding bits format (which for ARP is 144 bits long), assuming usage of MD5 hash function, is presented in Fig. 8.

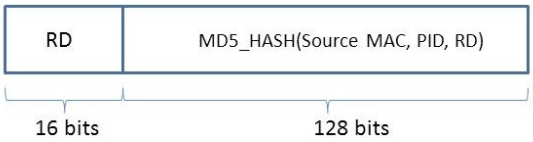


Figure 8. Padding format of ARP Request messages for the activation phase

All the hidden nodes are obligated to analyze the padding of all received ARP Requests. If an ARP Request is received with padding that is not all zeros, it is analyzed by extracting the random number and calculating corresponding hashes (2) as follows

$$R_H(PID) = H(PID \| RD \| SR\_MAC) \qquad (4\text{–}1)$$

For each extracted hash, receiver computes hashes with different *PID*. The order of the *PID* values for hashes calculation should correspond to traffic characteristics i.e. more likely carrier-protocols should be checked first. For example, based on *PID* values in Table II, $R_H(1)$ will be computed first, then $R_H(2)$ etc. because padding will more likely occur for TCP protocol than ARP and others. Such approach will limit unnecessary hashes calculation. Finally, if the received and calculated hashes are the same it means that a new hidden node is available for steganographic exchange and the carrier-protocol for this node is established. It means that if any hidden node receives frames from this new hidden node, only these corresponding to extracted *PID* value carry steganogram and will be analyzed.

Each hidden node stores a list of nodes from which it has received advertisements with their advertised carrier-protocol. Every hidden node should also reissue ARP Requests at certain time intervals to inform other hidden nodes about its existence. To limit the chance of detection, sending of ARP Requests may not happen too often (3, 4). In ARP, if an entry in host ARP cache is not refreshed within 1 to 20 minutes (implementation dependent) it expires and is removed. Thus, hidden nodes should mimic such behavior to imitate the sending of ARP Requests caused by ARP cache expiration.

Adaptation of ARP messages for identification of new hidden nodes has two advantages:

- The broadcast messages will be received by all hosts in LAN.
- The ARP traffic totals to about 0.1% of all traffic (see next Section for details), so this choice is also beneficial from the performance perspective. Each hidden node does not have to analyze all of the received traffic but only ARP Requests.

Phase II

After the identification of a new hidden node and its carrier-protocol, other hidden nodes analyze each short Ethernet frame's padding sent from that MAC address that in upper layers has chosen carrier-protocol. The received frames' padding contains steganogram bits.

The bidirectional transmission is performed as presented in Fig. 9. Two hidden nodes make e.g. an overt TCP connection – they transfer a file (1). During the connection TCP ACK segments are issued with improper Ethernet frame padding (2 and 4). Received TCP segments are analyzed for improper Ethernet padding presence and secret data is extracted (3 and 5). For third party observer such communication looks like usual data transfer.

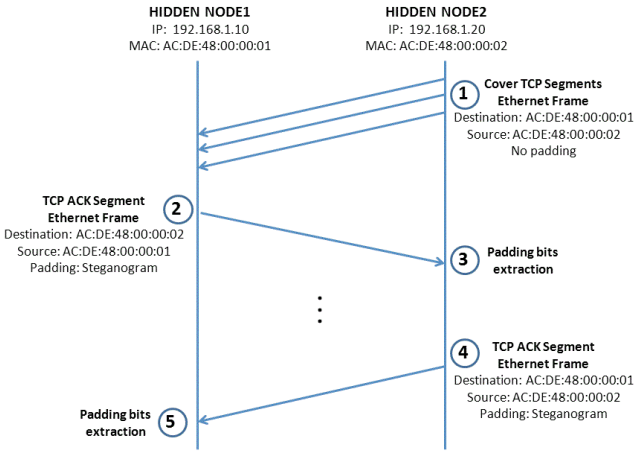


Figure 9. Hidden group steganograms exchange phase

During the exchange of steganograms or between two consecutive connections between two hidden nodes changing of carrier-protocol may occur. Hidden nodes may achieve this with use of *carrier-protocol hopping* mechanism. Let assume that there are two hidden nodes HN1 and HN2 and they want to change their carrier-protocols. To achieve it they do as follows (see Fig. 10):

- When HN1 wants to change its carrier-protocol it issues ARP Request which contains different from previous *PID* included in the hash inserted into the padding of this frame (see Fig. 8). ARP Request has TPA field set to IP address of the HN2 (1).
- After receiving ARP Request HN2 updates its list of hidden nodes and their carrier-protocols based on calculated hash analysis and PID (2). Then HN2 issues ARP Reply directly to HN1, which in padding contains its carrier-protocol preference (3).
- When HN1 receives ARP Reply it updates its list of hidden nodes and their carrier-protocols and is ready to use different carrier-protocol for HN2 i.e. it will analyze padding from all the short frames that in upper layers has chosen carrier-protocol (4).

Note that steganogram exchange does not necessarily must be symmetrical i.e. hidden nodes do not have to use the

same carrier-protocols which performing hidden data transfer.

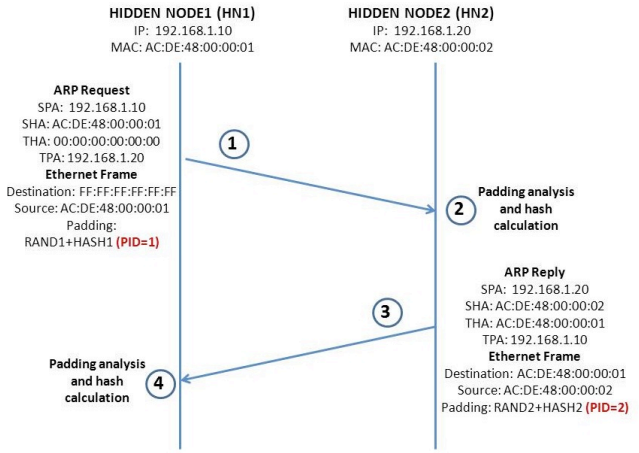


Figure 10. Carrier-protocol hopping mechanism example

V. PADSTEG EVALUATION

A. Padding content analysis

Table III presents hexadecimal values of frame padding, written in regular expression standard. Depending on day of observation padding contained different values, therefore we cannot state which value occurred most or least often. However, values bolded did not change in consecutive days. Some values were constant and other completely random. Therefore, we can make an assumption that padding content pattern changes with reboot of the device. Results confirm that memory leakage value in padding show some patterns that are very difficult to predict. That is why, we suggest that the proposed system should sacrifice few bits of the padding to generate some pattern in every message in order to increase undetectability.

TABLE III. FRAME PADDING CONTENT VARIETY (HEXADECIMAL VALUES)

Padding Length	6B	18B
Regex	**00{2}[0-F]{4}**	**80fca7a0[0-F]{14}**
	80[0-F]{5}	**a96f[0-F]{16}**
	c0[0-F]{5}	00{14} [0-F]{4}
	20{6}	[0-F]+00{3}[0-F]*
	474554202f[0-F]{1}	80fca7a0ffffffffffff[0-F]{8}
	0101050a74b6	80fca7a080fe88e0ffffffff0 012179cfd53
	[0-F]{6} (random)	[0-F]{18} (random)

B. Steganographic bandwidth estimation

Let us try to estimate *PadSteg* steganographic bandwidth for a single hidden node transmitting in a hidden group.

Because, currently, there are no tools for steganography detection, in real-life networks, every member of a hidden group can exchange almost unlimited number of steganograms and remain undiscovered. However, if the network traffic is consequently monitored, a naive use of *PadSteg* – that is: excessive generation of Ethernet frames with improper padding may be easily detected.

This leads to conclusion that it is important to evaluate what is the realistic steganographic bandwidth under the assumption that the secret data exchange will not differ from other hosts' traffic burdened with the *Etherleak* vulnerability. To achieve this goal steganographic user's network activity must mimic behavior of other users in terms of sending Ethernet frames with improper padding.

We calculated the steganographic bandwidth of the proposed system based on the average, daily number of TCP, ARP, ICMP, UDP messages with improper Ethernet padding per susceptible host (see Table IV).

Because each TCP and ICMP messages padding is 6 bytes long, ARP message padding 18 bytes, the average steganographic bandwidth is about 32 bit/s (with a daily standard deviation of about 14 bit/s). Therefore, if the hidden node generates Ethernet frames with improper padding that fall within the average range, for the inspected LAN network, steganographic communication may remain undetected.

TABLE IV. THE NUMBER OF FRAMES WITH IMPROPER PADDING PER HOST

Prot.	*Monday*	*Tuesday*	*Wednesday*	*Thursday*	*Friday*
TCP	25,379	53,469	31,014	79,981	52,940
ARP	1,036	250	2,116	2,828	1,825
ICMP	618	1,330	1,154	1,660	9
UDP	31	117	65	1,773	77

TABLE V. ESTIMATED STEGANOGRAPHIC BANDWIDTH

[bit/s]	*TCP*	*ARP*	*ICMP*	*Sum*
Average steg. bandwidth	26.98	3.43	1.90	32.31
Standard deviation	12.03	1.15	0.66	13.84
Confidence Interval (95%)	5.41	0.52	0.30	6.23

C. PadSteg prototype

PadSteg prototype – *StegTalk* – was implemented in C/C++ programming language with use of WinPcap 4.1.1 library (www.winpcap.org) for Windows XP OS. *StegTalk* is limited in functioning to ARP protocol only, so the PID value (see Fig. 8) is constant and equal 2. Application allows sending and receiving content from *.txt files between program instances running on different hosts.

StegTalk behavior is not deterministic in time. Messages containing steganograms are sent every ~60 seconds (depending on initial command line arguments) and initialization messages every 180 seconds, imitating host with Windows XP OS behavior. The ~60 seconds interval

was estimated in the following way. Based on experimental results presented in Table V maximum steganographic throughput that sustains high undetectability level, using ARP protocol is ~4 bit/s. It means that a single ARP message is issued every ~45 seconds. However, because initialization ARP messages are sent every 180 seconds, therefore, messages containing actual data should be sent every ~60 seconds.

Exemplary *StegTalk* output and functioning is presented in Fig. 10. Hidden host received ARP message and discovered new hidden node (1). Then host sent its own advertisement ARP message with steganographic capabilities (2). Every ARP message that hash was not successfully recognized is ignored (3). Each ARP message which is received from known hidden node is verified and hidden data is extracted ("topsecretmessage") (4).

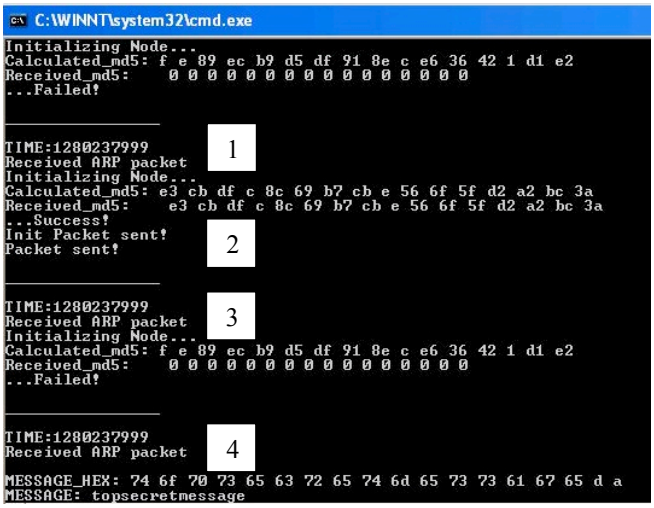


Figure 10. StegTalk application functioning

StegTalk tests were conducted on two virtual PC's with use of VMware Server 2.0 (www.vmware.com). Fixed-size text was sent from one host to another three times for each application mode (maximizing undetectability --*slow* or throughput --*fast,* see Fig. 11), in order to measure the time needed to receive the full text. Measured goodput (application level throughput) was approx. 2.3 bit/s and depending on program initial command line arguments it varied between 1.7 bit/s and 2.5 bit/s (standard deviation approx. 0.2 bit/s).

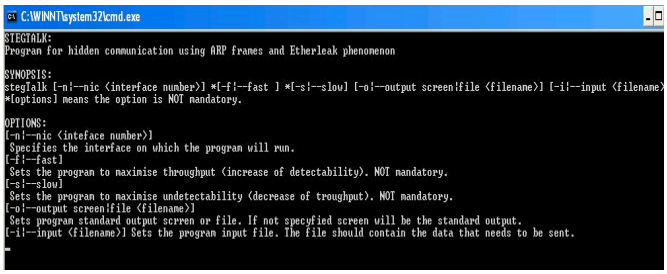


Figure 11. StegTalk application arguments

Having tested *StegTalk* behavior, in order to estimate application undetectability, sample host's network traffic had to be profiled – Fig. 12. Generally, application generates significantly fewer messages than the host during each 24h period. It is worth noting that the total amount of ARP messages will be a sum of those generated by host and *StegTalk*. Editing Windows OS registry keys may decrease the amount of ARP messages send by host and would increase *StegTalk* undetectability.

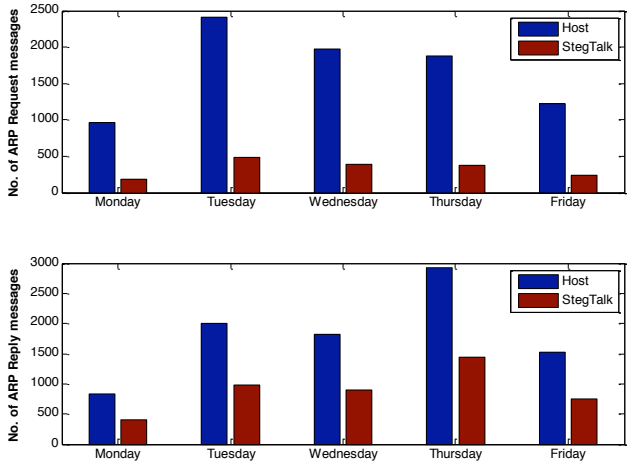


Figure 12. No. of ARP messages generated each day by an exemplary host and *StegTalk* application

VI. POSSIBLE COUNTERMEASURES

Our proposal of the new steganographic system, *PadSteg*, proves that such phenomenon like *inter-protocol steganography* is possible and may pose a threat to network security.

In today's LANs, with security measures they provide, *PadSteg* will be hard to detect. The main reason for this is that current IDS/IPS (Intrusion Detection/Prevention System) systems are rarely used to analyze all traffic generated in a LAN as this would be hard to achieve from the performance point of view. Moreover, usually IDSs/IPSs operate on signatures, therefore they require continuous signatures updates of the previously unknown steganographic methods, especially, if the information hiding process is distributed over more than one network protocol (as it is in *PadSteg*).

Thus, the best steps we can take to alleviate *PadSteg* in LANs are to:

- Ensure that there are no NICs with *Etherleak* vulnerability in the LAN.
- Enhance IDS/IPS rules to include *PadSteg* and deploy them in LANs.
- Improve access devices (e.g. switches) by adding active warden functionality [7] i.e. ability to modify (set to zeros) Ethernet frame padding if an improper one is encountered.

Implementation of the specified countermeasures greatly minimizes the risk of successful *PadSteg* utilization.

VII. CONCLUSIONS

In this paper we presented new steganographic system - *PadSteg* – which is the first information hiding solution based on *inter-protocol steganography*.

It may be deployed in LANs and it utilizes two protocols to enable secret data exchange: Ethernet and ARP/TCP. A steganogram is inserted into Ethernet frame padding but one must always "look" at the other layer protocol (ARP or TCP) to determine whether it contains secret data or not. Based on the results of conducted experiment the average steganographic bandwidth of *PadSteg* was roughly estimated to be 32 bit/s. It is a quite significant number considering other known steganographic methods.

In order to minimize the potential threat of *inter-protocol steganography* to public security identification of such methods is important. Equally crucial is the development of effective countermeasures. This requires an in-depth understanding of the functionality of network protocols and the ways in which they can be used for steganography.

However, considering the complexity of network protocols being currently used, there is not much hope that a universal and effective steganalysis method can be developed. Thus, after each new steganographic method is identified, security systems must be adapted to the new, potential threat.

As a future work larger volumes of traffic from different LANs should be analyzed in order to pinpoint more accurately *PadSteg* feasibility and calculate its steganographic bandwidth.

ACKNOWLEDGMENT

This work was partially supported by the Polish Ministry of Science and Higher Education under Grant: N517 071637.

REFERENCES

[1] Rowland C., Covert Channels in the TCP/IP Protocol Suite, First Monday, Peer Reviewed Journal on the Internet, July 1997

[2] Zander S., Armitage G., Branch P., A Survey of Covert Channels and Countermeasures in Computer Network Protocols, IEEE Communications Surveys & Tutorials, 3rd Quarter 2007, Volume: 9, Issue: 3, pp. 44-57, ISSN: 1553-877X

[3] Petitcolas F., Anderson R., Kuhn M., Information Hiding – A Survey: IEEE Special Issue on Protection of Multimedia Content, July 1999

[4] Murdoch S.J., Lewis S., Embedding Covert Channels into TCP/IP, Information Hiding (2005), pp. 247-26

[5] Ahsan, K. and Kundur, D.: Practical Data Hiding in TCP/IP, Proc. ACM Wksp. Multimedia Security, December 2002.

[6] Kundur D. and Ahsan K.: Practical Internet Steganography: Data Hiding in IP, Proc. Texas Wksp. Security of Information Systems, April 2003.

[7] Fisk, G., Fisk, M., Papadopoulos, C., Neil, J.: Eliminating Steganography in Internet Traffic with Active Wardens, In Proc: 5th International Workshop on Information Hiding, Lecture Notes in Computer Science: 2578, 2002, str. 18–35

[8] Lucena N. B., Lewandowski G., Chapin S. J., Covert Channels in IPv6, Proc. Privacy Enhancing Technologies (PET), May 2005, pp. 147–66.

[9] Arkin O., Anderson J., Ethernet frame padding information leakage, Atstake report, 2003

http://packetstorm.codar.com.br/advisories/atstake/atstake_etherleak_report.pdf

[10] Plummer D. C., An Ethernet Address Resolution Protocol, RFC 826, November 1982

[11] Girling C. G., Covert Channels in LAN's, IEEE Trans. Software Engineering, vol. SE-13, no. 2, Feb. 1987, pp. 292–96.

[12] Handel T., Sandford M.. Hiding Data in the OSI Network Model. In Proceedings of the First International Workshop on Information Hiding, pages 23-38, 1996.

[13] Wolf M, "Covert Channels in LAN Protocols," Proc. Wksp. Local Area Network Security (LANSEC), 1989, pp. 91–101.

[14] Mazurczyk W, Szczypiorski K (2008) Steganography of VoIP Streams, In: R. Meersman and Z. Tari (Eds.): OTM 2008, Part II - Lecture Notes in Computer Science (LNCS) 5332, Springer-Verlag Berlin Heidelberg, Proc. of The 3rd International Symposium on Information Security (IS'08), Monterrey, Mexico, November 2008, pp. 1001-1018

[15] Mazurczyk W., Smolarczyk M., Szczypiorski K.: Retransmission steganography and its detection, Soft Computing, ISSN: 1432-7643 (print version), ISSN: 1433-7479 (electronic version), Journal no. 500 Springer, November 2009

[16] B. Jankowski, W. Mazurczyk, K. Szczypiorski, Information Hiding Using Improper Frame Padding, Submitted to 14th International Telecommunications Network Strategy and Planning Symposium (Networks 2010), 27-30.09.2010, Warsaw, Poland

Bartosz Jankowski studies telecommunication at Warsaw University of Technology (WUT, Poland) since 2007. His main areas of interest are network security, information hiding techniques and recently project management. Member of the Network Security Group at WUT (secgroup.pl) and coauthor of first inter-protocol steganographic system *PadSteg*. He is regarded as goal-oriented person with a strong drive to learn. He is a co-author of 3 publications and 1 invited talk.

Wojciech Mazurczyk holds an M.Sc. (2004) and a Ph.D. (2009) in telecommunications from the Faculty of Electronics and Information Technology, Warsaw University of Technology (WUT, Poland) and is now an Assistant Professor at WUT and the author of over 50 scientific papers and over 25 invited talks on information security and telecommunications. His main

research interests are information hiding techniques, network security and multimedia services. He is also a research co-leader of the Network Security Group at WUT (secgroup.pl). Personal website: http://mazurczyk.com.

Krzysztof Szczypiorski holds an M.Sc. (1997) and a Ph.D. (2007) in telecommunications both with honours from the Faculty of Electronics and Information Technology, Warsaw University of Technology (WUT), and is an Assistant Professor at WUT. He is the founder and head of the International Telecommunication Union Internet Training Centre (ITU-ITC), established in 2003. He is also a research leader of the Network Security Group at WUT (secgroup.pl). His research interests include network security, steganography and wireless networks. He is the author or co-author of over 110 publications including 65 papers, two patent applications, and 35 invited talks.

2010 International Conference on Multimedia Information Networking and Security

Stream Control Transmission Protocol Steganography

Wojciech Frączek, Wojciech Mazurczyk, Krzysztof Szczypiorski
Institute of Telecommunications
Warsaw University of Technology
Warsaw, Poland
e-mail: wfraczek@gmail.com,{wmazurczyk, ksz}@tele.pw.edu.pl

Abstract— **Stream Control Transmission Protocol (SCTP) is a new transport layer protocol that is due to replace TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) protocols in future IP networks. Currently, it is implemented in such operating systems like BSD, Linux, HP-UX or Sun Solaris. It is also supported in Cisco network devices operating system (Cisco IOS) and may be used in Windows. This paper describes potential steganographic methods that may be applied to SCTP and may pose a threat to network security. Proposed methods utilize new, characteristic SCTP features like multi-homing and multi-streaming. Identified new threats and suggested countermeasures may be used as a supplement to RFC 5062, which describes security attacks in SCTP protocol and can induce further standard modifications.**

Keywords: steganography, SCTP

I. INTRODUCTION

Steganographic techniques have been used for ages and dates back to the ancient Greece [4]. The aim of the steganographic communication back then and now, in modern applications, is the same: hide secret data (steganogram) in innocent looking cover and send it to the proper recipient which is aware of the information hiding procedure. In ideal situation the existence of hidden communication cannot be detected by third parties. What distinguishes historical steganographic methods from modern ones is, in fact, only the form of the cover (carrier) for secret data. Historical methods used human skin, wax tables or letters etc., nowadays rather digital media like pictures, audio, video which are transmitted using telecommunication networks were often used. Recent trend in steganography is utilization of the network protocols as a steganogram carrier by modifying content of the packets they use, time relations between these packets or hybrid solutions. All of the information hiding methods that may be used to exchange steganograms in telecommunication networks is described by the term network steganography which was originally introduced by Szczypiorski in 2003 [8]. Many steganographic methods have been proposed and analyzed, e.g. [1]-[4]. They should be treated as a threat to network security, because they may cause e.g. confidential information leakage. Steganography as a network threat was marginalized for few years but now not only security staff but even business and consulting firms are becoming continuously aware of the potential danger and possibilities it creates [10].

Knowledge of the information hiding procedure is helpful to develop countermeasures therefore, it is important to identify potential, previously unknown possibilities for covert communication. It is especially important when it comes to new network protocols that are forecasted to be widely deployed in future networks. For example, the detailed analysis of information hiding methods in IPv6 protocol header was presented by Lucena et al. [9]. The same case is with Stream Control Transmission Protocol (SCTP) [5] which is a transport layer protocol and its main role is similar to both popular protocols Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). It provides some of the same service features of both, ensuring reliable, in-sequence transport of messages with congestion control. Nevertheless, there are certain advantages which make SCTP a candidate for a transport protocol in future IP networks – the main are that it is multi-streaming and multi-homing.

To authors' best knowledge, there are no steganographic methods proposed for SCTP protocol. However, information hiding methods that have been proposed for TCP and UDP protocols (e.g. utilizing free/unused or not strictly standard-defined fields) may be utilized as well due to several similarities between these transport layer protocols and SCTP. Steganographic methods for TCP and UDP protocols were described by Rowland [1] and Murdoch and Lewis [2] and very good surveys on hidden communication can be found in Zander et al. [3] and Petitcolas et al. [4].

The popularity of the SCTP is still growing as it has been already deployed in many important operating systems like BSD, Linux (the most popular is lksctp [13]), HP-UX or Sun Solaris and is supported Cisco network devices operating system (Cisco IOS) and even in Windows if the proper library is installed [11].

This paper can be treated as a supplement to RFC 5062 [12], which describes security attacks in SCTP protocol and current countermeasures. However, it does not include any information about steganography-based attacks and ways to prevent them. That is why, in this paper we identify new attack opportunities to network security for SCTP and propose detection and/or elimination techniques.

The rest of the paper is structured as follows. Section 2 gives brief overview of SCTP protocol. In Section 3 network steganography methods that are characteristic for SCTP protocol are presented. Section 4 provides possible detection and elimination solutions for proposed methods. Finally, Section 5 concludes our work.

978-0-7695-4258-4/10 $26.00 © 2010 IEEE
DOI 10.1109/MINES.2010.176

II. OVERVIEW OF SCTP PROTOCOL

SCTP [5] was defined by the IETF Signaling Transport (SIGTRAN) working group in 2000, and is maintained by the IETF Transport Area (TSVWG) working group. It was being developed for one specific reason - transportation of telephony signaling over IP-based networks. However, its features make it capable of being general purpose transport layer protocol ([5], [6]).

SCTP, like TCP, provides reliable, in-sequence data transport with congestion control, but it also eliminate limitations of TCP, which are more and more onerous in many applications. SCTP allows also to set order-of-arrival delivery of the data, which means that the data is delivered to the upper layer as soon as it is received (a sequence number is of no significance). Unordered transmission can be set for all messages or only for part of the messages depending on application need.

The SCTP Partial Reliability Extension, defined in [7], is a mechanism which allows to send not all data if it is not necessary, i.e. data, which were not correctly received but got out-of-date. Decision not to transmit some data is made by sender. He/she has to inform a receiver that some data will not be sent and receiver should treat this data like correctly received and acknowledged. Partial Reliability Extension and order-of-arrival delivery enable to use SCTP in many applications which are using UDP now.

In TCP all data is sent as a stream of bits with no boundaries between messages. This behavior requires that TCP-based applications have to do message framing and provide a buffer for incomplete messages from TCP agent. In SCTP, data is sent as separate messages passed by the upper layer. This feature makes SCTP-based applications easier to develop than TCP-based ones.

Each SCTP connection (which is called association in SCTP) can use one or more streams, which are unidirectional logical channels between SCTP endpoints. Order-of-transmission or order-of-arrival delivery of data is performed within each stream separately, not globally. If one of the streams is blocked (i.e. a packet is lost and receiver is waiting for it), it does not affect other streams. Benefit of using multiple streams is illustrated in Fig. 1.

User X sends four messages (A, B, C, D) to user Y. There are two requirements concerning delivery order of these messages. Message A must be delivered before message B, and message C must be delivered before message D. In TCP messages are sent in following order: A, B, C, D (1). If message A is lost (2), other messages, in spite of the correct reception, cannot be dispatched to the upper layer until message A is retransmitted and successfully received by user Y (3). In SCTP, using multi-streaming, messages can be divided into two streams. Messages A and B can be sent within stream 1, and messages C and D can be sent within stream 2 (4). If message A is lost (5), only message B cannot be passed to the upper layer until message A is received. Messages C and D can be delivered to the upper layer, since they are sent within different stream than messages A and B (6).

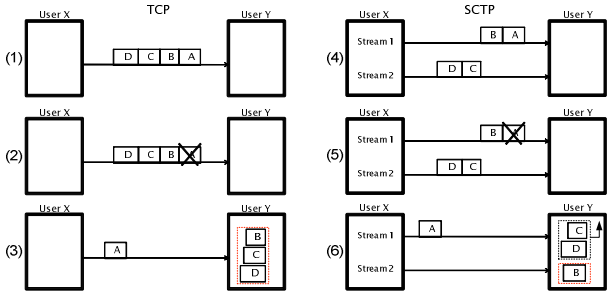


Figure 1. Comparison of TCP and SCTP data transport using multiple streams

Another SCTP feature is provision for protocol extensibility. Each SCTP packet consists of main header and one or more chunks (Fig. 2). There are two types of chunks: data chunks, which contain user data and control chunks, which are used to control data transfer. Each chunk consists of fields and parameters specific to chunk type (Fig. 3). Fields are mandatory, and parameters can be either mandatory or optional. SCTP packet structure allows defining not only new chunk types but also broadening functionality of the existing chunk types through defining new parameters.

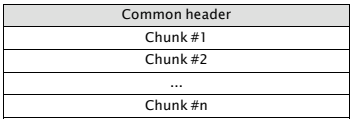


Figure 2. SCTP packet format

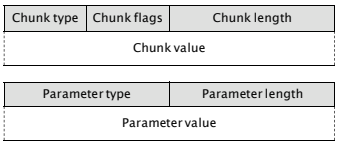


Figure 3. SCTP chunks and parameters format

SCTP supports multi-homing i.e. host ability to be visible in the network through more than one IP address, for instance if host is equipped with a few NICs (Network Interface Cards). Multi-homing in SCTP is used to provide more reliable data transfer. If there are no packets losses, all messages are transmitted using one source address and one destination address (primary path). If chunk is retransmitted, it should be sent using different path (different source and destination addresses) than primary path. Another advantage of SCTP multi-homing in SCTP is ability to failover data transfer if primary path is down.

SCTP uses a four-way handshake with cookie (Fig. 4), which provides protection against synchronization attack (type of Denial of Service attack) known from TCP. In SCTP, user initiates an association with INIT chunk. In response he/she receives INIT ACK chunk with cookie

(containing information that identifies proposed connection). Then he/she replies with a COOKIE ECHO with copy of received cookie. Reception of this chunk is acknowledged with COOKIE ACK chunk. After successful reception of COOKIE ACK association is established. Afterwards connected users can send data using DATA chunks and acknowledge reception of them with SACK chunks.

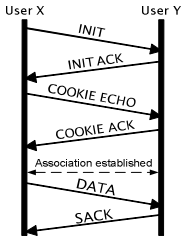


Figure 4. SCTP association establishment

Aside from described features, SCTP provide also built-in path MTU discovery, data fragmentation mechanism and, in general, it is considered more secure than TCP.

III. SCTP-SPECIFIC STEGANOGRAPHIC METHODS AND DETECTION POSSIBILITIES

SCTP-specific steganographic methods can be divided in three groups:

- Methods that modify content of SCTP packets.
- Methods that modify how SCTP packets are exchanged.
- Methods that modify both content of SCTP and the way they are exchanged – hybrid methods.

A. Methods that modify content of SCTP packets

As mentioned before, each SCTP packet consists of chunks and each chunk can contain variable parameters. We propose 13 new steganographic methods which modify content of SCTP packets in the following chunks and parameters:

- INIT and INIT ACK chunks – used during initialization of SCTP association (methods I1, I2),
- DATA chunks – which contain user data (methods D1, D2),
- SACK chunks – used to acknowledge received DATA chunks (methods S1, S2),
- AUTH chunk – used to authenticate chunks (method A1),
- PAD chunk – used to pad packets (method P1),
- Variable parameters – used in specific chunks. (methods VP1-5).

Steganographic methods listed above are explained below.

INIT and INIT ACK chunks

(I1) *Initiate Tag* is a 32 bits value of *Verification Tag* field. It must be inserted into each SCTP packet, which is sent to the originator of INIT or INIT ACK chunks within this association. The *Initiate Tag* can be any value except 0, thus it may be used for steganographic purposes. Maximum bandwidth of this channel is 32 bits/chunk (fewer bits of this field should be used in order to limit chance of detection).

(I2) *Number of Inbound Streams* is a 16 bits field which define the maximum number of inbound streams that sender of the INIT or INIT ACK can handle within this association. In most cases, using more than one hundred streams is unlikely, thus at least a few the most significant bits can be used to insert hidden data. To limit the risk of detection not only the most significant bits may be used. Potential bandwidth of this method is 8 bits/chunk.

DATA chunks

(D1) *Stream Sequence Number* (SSN) is a 16 bits sequence number within each stream. If order-of-arrival delivery of data is set, there are no requirements concerning SSN. This feature makes it possible to use SSN to send steganograms. Maximum bandwidth of this channel is 16 bits/chunk. Presented method can be utilized only if unordered transmission is set for all data within a stream.

(D2) *Payload Protocol Identifier* is a 32 bits field which represents an upper layer protocol identifier. This field is not used by SCTP agent, it is for purposes of upper layer protocols. Value 0 indicates no identifier, other values should be standardized with IANA. SCTP does not verify this value, so it can be used to send secret data. Maximum bandwidth of this channel is 32 bits/chunk.

SACK chunks

(S1) *Advertised Receiver Window Credit* is a 32 bits field which indicates current size of the SACK sender's receiver buffer. A few least significant bits of this field can be utilized for steganographic purposes. Potential bandwidth of this method is 3-4 bits/chunk. It cannot be higher since it may affect flow control.

(S2) Duplicate TSNs, which are part of the SACK chunk, are sequence numbers of the duplicate chunks which has been received. This mechanism may enable hidden communication through adding not duplicating chunks TSNs to the list of duplicate TSNs. In spite of 32 bits length of TSN, potential steganographic bandwidth is few bits per chunk. This is because adding very different TSNs from recently sent is easy to detect. Presented method is harder to detect if it is used by multi-homed hosts since it should be considered to send SACK chunks with duplicates to other address than source address of DATA chunks.

AUTH chunks

(A1) *Shared Key Identifier* is a 16 bits field that indicates which pair of shared keys is used in this chunk. This field can be used for covert communication because receiver of the packet can authenticate sender through checking all previously exchanged shared keys. Potential steganographic bandwidth of this channel is 1-4 bits/chunk since, in most cases, there will be not many shared keys available. Detection of this method is quite hard because shared keys are established outside SCTP protocol.

PAD chunks

(P1) *Padding Data* is a field which length depends on padding needs. There are no requirements concerning value of this field, so it can be used for covert communication. Thus, steganographic bandwidth of this channel depends on size of padding data.

Variable Parameters

(VP1) IPv4 Address in *IPv4 Address Parameter* and IPv6 Address in *IPv6 Address Parameter* contain addresses of the sending endpoints. These parameters are used for multi-homed hosts and can be attached to INIT, INIT ACK and ASCONF (used to dynamic address reconfiguration) chunks. Each address in these parameters is considered as unconfirmed until its reachability is not checked. This behavior allows using these parameters for steganographic purposes by sending secret data instead of IP address. Maximum bandwidth is 32 bits/parameter for IPv4 address and 128 bits/parameter for IPv6 address.

(VP2) *Heartbeat Info Parameter* is used in HEARTBEAT chunk, which is exploited to verify reachability of the destination addresses. *Heartbeat Info Parameter* contains *Sender-Specific Heartbeat Info* field, which content is not defined, so it can be used a steganogram carrier. In Linux Kernel Stream Control Transmission Protocol (lksctp-2.6.28-1.0.10) implementation of SCTP, *Sender-Specific Heartbeat Info* field has 40 bytes, thus steganographic bandwidth for this methods is about 320 bits/chunk.

TABLE I. SUMMARY OF METHODS' POTENTIAL STEGANOGRAPHIC BANDWIDTH

Steganographic method	*Steganographic bandwidth*	*Units*
I1	32	bits/chunk
I2	8	bits/chunk
D1	16	bits/chunk
D2	32	bits/chunk
S1	3-4	bits/chunk
S2	3-4	bits/chunk
A1	1-4	bits/chunk
P1	varies	n/a
VP1	32	bits/par.
VP2	320	bits/chunk
VP3	32	bits/chunk
VP4	32	bits/par.
VP5	varies	n/a

(VP3) *Random Number* in *Random Parameter* also can be used for covert communication. Steganographic bandwidth of this method depends on purpose of the number. If it is used in authentication process, random number has 32 bits and it is sent in INIT or INIT ACK chunks. That is why the maximum steganographic bandwidth is 32 bits/chunk.

(VP4) *ASCONF-Request Correlation ID* in *Add IP Address Parameter*, *Delete IP Address Parameter* and *Set Primary Address Parameter* is 32 bits field which identifies each request. The only requirement concerning its value is to be unique for each request, thus it may be used to transfer steganograms. The maximum steganographic bandwidth of this method is 32 bits/parameter.

(VP5) *Padding Data* in *Padding Parameter* can be exploited for covert communication in the same way as *Padding Data* in *Padding* chunk (see method P1). *Padding Parameter* can be used only in the INIT chunk.

B. Methods that modify how SCTP packets are exchanged

MULTI-HOMING

SCTP multi-homing feature can be utilized to perform hidden communication. The main idea of the proposed steganographic method is presented in Fig. 4. Two users establish SCTP association (User 1 and User 2), each of them is equipped with more than one NIC. The primary path for the users' communication is through interfaces A and X (1). If n_1 denotes the number of the alternative sender NIC addresses (in Fig. 4 they are 2), and n_2 represents the number of alternative receiver NIC addresses (in Fig. 4 also 2) then each address can be used to represent one steganogram bit (or a sequence of bits). Possible alternative paths for communication between these users are: BY, BZ, CY and CZ. User 1's B interface IP address represents binary '0', C interface IP address binary '1' (similar situation is for User 2). Assigning the bits or sequence of bits to the users' NICs may depend on the IP addresses value i.e. available NICs addresses can be sorted from lowest to highest and then consecutive values (bit sequences) can be assigned to them.

If User 1 wants to send steganogram, he/she waits for the transmission error on primary path to occur and then retransmits chunk through appropriate path. For example, in Fig. 4, if User 1 wants to send steganogram which consists of the sequence '01', he/she waits for the transmission error on primary path to occur (1) and sends retransmitted packets through path BZ (2). Before sending steganogram it should be established which retransmitted chunks carry hidden data. Users can assume that all retransmissions carry bits of steganogram or should mark beginning of hidden communication, for example, with an initiation sequence (a sequence of retransmitted chunks through previously agreed paths).

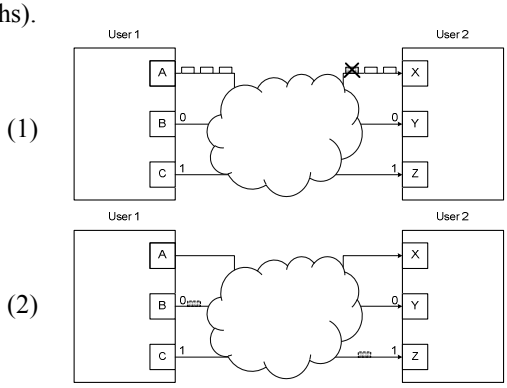


Figure 4. Multi-homing based steganographic method

Steganographic bandwidth $S_{B\text{-}MH}$ for this method can be expressed as

$$S_{B-MH} = \log_2(n_1) + \log_2(n_2) \quad [bits/chunk] \qquad (3\text{-}1)$$

For example in Fig. 4, if SCTP packets rate is 250 packets/s, assuming that each packet contains only single data chunk and the retransmission rate is 2% (retransmission rate in Internet is up to 5%), then achieved steganographic bandwidth is 10 bits/s.

MULTI-STREAMING

In SCTP, multi-streaming (for ordered delivery) is realized by utilizing two identifiers: Stream Identifier (SI), to uniquely mark stream and Stream Sequence Number (SSN) to ensure correct order of packets at the receiver. Despite these two identifiers each DATA chunk contains also Transmission Sequence Number (TSN) that is assigned independently to each chunk.

Steganographic method that adopts multi-streaming is based on determined assignment of TSNs for every chunk distributed along different streams. SIs in subsequent DATA chunks will represent hidden data bits. The example for this method is presented in Fig. 5.

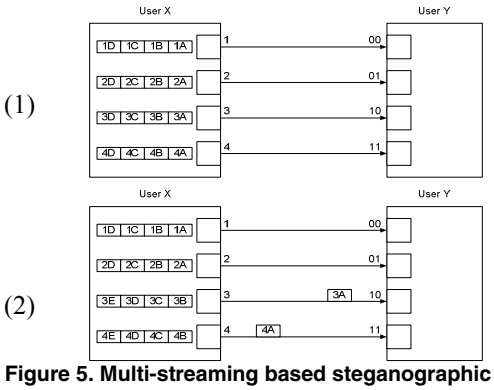


Figure 5. Multi-streaming based steganographic method

At initialization phase of the SCTP association users negotiate a number of utilized streams (in the example there are 4 streams). Each stream is assigned with binary sequences (1) – from '00' to '11'. Sending the data through certain stream depends on the steganogram bits. Therefore, if User X wants to secretly transfer '1011' bits sequence he/she first sends data through stream 3, then through stream 4 (2). If *s* denotes the number of available streams, then maximum steganographic bandwidth $S_{B\text{-}MS}$ for this method may be expressed as

$$S_{B-MS} = \log_2(s) \quad [bits/chunk] \qquad (3\text{-}2)$$

For example, if we assume that the overt communication rate is 250 packets/s, each packet has only one chunk with data and 4 streams are used then the steganographic bandwidth is 500 bits/s.

C. Hybrid method

For SCTP partial reliability extension was also proposed by Stewart et al. [7]. It allows not retransmitting certain data despite the fact it was not successfully received. It is possible through the FORWARD TSN (FT) chunk, where new acknowledge TSN is inserted. After receiving such message receiving side treats missing chunks with equal or lower TSNs as they were properly delivered. This functionality may be adopted for steganographic purposes. The idea of the proposed method is similar in concept to LACK which was developed for real-time multimedia services by Mazurczyk and Szczypiorski [14].

The main idea of the proposed method is presented in Fig. 6.

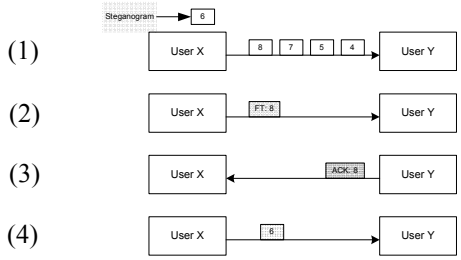


Figure 6. Multi-streaming based steganographic method

From the User X data sent chunk with TSN 6 is skipped and to this chunk steganogram is inserted (1). Next, User X sends FT chunk to signal new acknowledged TSN (2). After successful reception of FT chunk, User Y issues SACK chunk with new acknowledged TSN (3). When User X receives SACK chunk, he/she can send omitted DATA chunk with steganogram (4).

If we assume that the overt communication rate is 250 packets/s, each packet has only one chunk with payload size being 1000 bytes and we use 0.01% of packets to insert steganogram then the potential steganographic bandwidth is 200 bits/s.

IV. DETECTION POSSIBILITIES

For each of the groups of steganographic methods proposed in Section 3 detection or elimination solutions are sketched. The main aim of this Section is to point out potential enhancements that may be applied to SCTP standard to alleviate steganography utilization, ideally, at the standard development stage. Therefore, proposed countermeasures should be treated as guidelines for standard improvements.

A. Methods that modify content of SCTP packets

For steganographic methods that utilize modification to the SCTP packets content possible detection techniques and proposed countermeasures are depicted in Table II.

B. Methods that modify how SCTP packets are exchanged

MULTI-HOMING

It is worth noting that steganographic methods that utilize multi-homing are generally harder to detect than single-homing ones, because to detect covert communication it requires observing traffic on few, different communication paths.

Resistance to detection for method proposed in Section 3 depends on how future typical SCTP implementations will behave. If alternative paths for retransmitted chunks will often change proposed steganographic method that utilizes multi-homing will be harder to detect. But if retransmitted chunks will be send through only one alternative path then other behavior will be treated as anomaly. Thus, requirement

that states that retransmitted chunks should be sent through only one alternative path should be enclosed in SCTP standard.

TABLE II. POSSIBLE STANDARD IMPROVEMENTS TO NEUTRALIZE STEGANOGRAPHIC METHODS THAT MODUFY CONTENT OF SCTP PACKETS

Steg. method	*Detection technique*	*Countermeasure (proposed standard change)*
I1	Analysis of Verification Tags values.	–
I2	Comparison between values of Maximum Inbound Streams sent by "normal" users (users who do not use steganography) and suspicious user.	Limit possible values of Maximum Inbound Streams, i.e. only powers of 2 may be allowed.
D1	Comparison between values of Stream Sequence Number sent by "normal" users (users who do not use steganography) and suspicious user.	For unordered transmission, Stream Sequence Number must be set to 0.
D2	Checking value of Payload Stream Identifier.	Only standardized values must be allowed.
S1	Analysis of a_rwnd values and sizes of received chunks.	–
S2	Analysis of average number of duplicated chunks.	–
A1	Analysis of Shared Key Identifier values.	Limit the number of shared keys for association to 1 or set one pair of shared keys for time slot, i.e. 10 minutes.
P1	Analysis of Padding Data.	All bits of Padding Data must be set to 0.
VP1	Checking the existence of IP addresses that are sent in these parameters.	Remove these parameters. Replace them with new chunk type, which will be sent from each user's address in order to add it to association.
VP2	Comparison between values of Heartbeat Info Parameter sent by normal user (user who do not use steganography) and suspicious user.	Define value of Heartbeat Info Parameter.
VP3	Analysis of Random Number.	–
VP4	Comparison between values of ASCONF-Request Correlation ID sent by normal user (user who do not use steganography) and suspicious user.	ASCONF-Request Correlation ID must be a sequence number.
VP5	Analysis of Padding Data.	All bits of Padding Data must be set to 0.

Whatever the implementation, statistical analysis of NIC addresses used for retransmitted chunks may help to detect hidden communication.

Elimination of proposed steganographic method is possible by changing source and destination addresses of randomly chosen packet that contains retransmitted chunks.

MULTI-STREAMING

Similarly to the multi-homing based steganographic method detection of multi-streaming method may be hard to perform and depends on the concrete application were SCTP will be utilized. If the pattern of streams usage is established, then statistical SCTP traffic analysis may reveal hidden communication.

Elimination of the proposed steganographic method may be achieved by changing TSNs by an intermediate node e.g. edge router with steganography detection functionality. Such operation may successfully interrupt proper exchange of hidden data.

C. Hybrid method

If the number of intentionally omitted chunks is kept to the reasonable level then detection of such method is hard – statistical analysis of the frequency of moving acknowledged TSNs may be helpful.

Elimination of such method is possible by a specialized intermediate node which will be responsible for detection and dropping of chunks that have been already acknowledged by the receiver.

V. CONCLUSIONS

In this paper we presented sixteen different steganographic methods that can be used in SCTP protocol. All of these methods may lead to confidential information leakage and should be treated as a threat to network security. A lot of them may be evaded by changing SCTP standard – where it is possible certain improvements were proposed.

This analysis emphasizes how important it is to further inspect other network protocols that are to be utilized in future networks to avoid hidden communication as early as possible, ideally, still at the standard development stage.

REFERENCES

[1] C. Rowland, "Covert Channels in the TCP/IP Protocol Suite", First Monday, Peer Reviewed Journal on the Internet, July 1997

[2] Murdoch S.J., Lewis S., *Embedding Covert Channels into TCP/IP,* Information Hiding (2005), pp. 247-26

[3] S. Zander, G. Armitage, P. Branch, "A Survey of Covert Channels and Countermeasures in Computer Network Protocols", IEEE Communications Surveys & Tutorials, 3rd Quarter 2007, Volume: 9, Issue: 3, pp. 44-57, ISSN: 1553-877X

[4] Petitcolas F., Anderson R., Kuhn M., Information Hiding – A Survey: IEEE Special Issue on Protection of Multimedia Content, July 1999

[5] Stewart R.: Stream Control Transmission Protocol. RFC 4960, September 2007.

[6] Stewart R., Xie Q.: Stream Control Transmission Protocol (SCTP): A Reference Guide. Addison-Wesley, 2002.

[7] Stewart R., Ramalho M., Xie Q., Tuexen M., Conrad P.: Stream Control Transmission Protocol (SCTP) Partial Reliability Extension. RFC 3758, May 2004

[8] Szczypiorski K., Steganography in TCP/IP Networks. State of the Art and a Proposal of a New System – HICCUPS, Institute of Telecommunications' seminar, Warsaw University of Technology, Poland, November 2003

URL:http://krzysiek.tele.pw.edu.pl/pdf/steg-seminar-2003.pdf

[9] N. B. Lucena, G. Lewandowski, and S. J. Chapin, Covert Channels in IPv6, Proc. Privacy Enhancing Technologies (PET), May 2005, pp. 147–66.

[10] Frost & Sullivan, Steganography: Future of Information Hiding, Technical Insights Deliverable, December 2009

http://www.frost.com/prod/servlet/report-toc.pag?repid=D1D9-01-00-00-00

[11] SCTP library (sctplib): URL: http://www.sctp.de/sctp-download.html

[12] R. Stewart, M. Tuexen, G. Camarillo, Security Attacks Found Against the Stream Control Transmission Protocol (SCTP) and Current Countermeasures, RFC 5062, September 2007

[13] The Linux Kernel Stream Control Transmission Protocol (lksctp) project: http://lksctp.sourceforge.net/

[14] Mazurczyk W, Szczypiorski K (2008) Steganography of VoIP Streams, In: R. Meersman and Z. Tari (Eds.): OTM 2008, Part II - Lecture Notes in Computer Science (LNCS) 5332, Springer-Verlag Berlin Heidelberg, Proc. of The 3rd International Symposium on Information Security (IS'08), Monterrey, Mexico, November 2008, pp. 1001-1018