Central European Journal of Computer Science • 1-30
The original publication is available at www.springerlink.com

Covert Channels in TCP/IP Protocol Stack - extended version-

Survey (Review) Article

Aleksandra Mileva*, Boris Panajotov †

University Goce Delčev, Faculty of Computer Science,
Goce Delčev 89, 2000, Štip

Abstract: We give a survey of different techniques for hiding data in several protocols from the TCP/IP protocol stack. Techniques are organized according to affected layer and protocol. For most of the covert channels its data bandwidth is given.

Keywords: Network steganography • Data hiding • Internet layer • Transport layer • Application layer

1. Introduction

An *overt channel* is a communication channel within a computer system or network, designed for the authorized transfer of data. A *covert channel* (first introduced by Lampson [62]), on the other hand, is any communication channel that can be exploited by a process to transfer information in a manner that violates the systems security policy [26]. Any shared resource can potentially be used as a covert channel. Covert channels can be divided primarily in *storage* and *timing channels*. In a case of the storage channels, usually one process writes (directly or indirectly) to a shared resource, while another process reads from it. Timing channel is essentially any technique that conveys information by the timing of events, in which case the receiving process needs a clock. Special timing covert channel is *counting channel* which carries data by counting the occurrences of certain events [44]. Additionally, timing channels can be *active* if they generate additional traffic or *passive* if they manipulate the timing of existing traffic. As any other communication channel, covert channel can be *noisy* or *noiseless*. According to the number of information flows between the sender and the receiver - several or one, there are *aggregated* and *non-aggregated* covert channels [36]. According to the presence or absence of the intermediate node in the communication, covert channels can be *indirect* or *direct*. *Payload tunnel* is a covert channel, where one protocol is tunnelled in the payload of the other protocol. Covert channels are studied as a part of the science

* *E-mail: aleksandra.mileva@ugd.edu.mk*
† *E-mail: boris.panajotov@ugd.edu.mk*

steganography, and different steganographic methods used in telecommunication networks are known as *network steganography*.

The adversary model is based on the Simmons prisoner problem [105]: two parties want to communicate confidentially and undetected over an insecure channel, the warden. The warden can be passive - which monitor traffic and report when some unauthorized traffic is detected, or active - which can modify the content of the messages with the purpose of eliminating any form of hidden communication. Simmons introduced the term *subliminal channel*, a variant of covert channel which uses cryptographic algorithm or protocol for hiding messages. A composition of covert channel with subliminal channel is the *hybrid channel*.

Covert channels can be analysed by the total number of steganogram bits transmitted during a second (Raw Bit Rate - RBR), or by the total number of steganogram bits transmitted per PDU (for example, Packet Raw Bit Rate- PRBR) [78]. In ideal case, if no packets are lost, capacity of some storage channels can be expressed as $C = PRBR \times N$ bps, where N is number of packets send in one second.

Network protocols are ideal for hiding data in them. Most of the network steganographic techniques for storage channels utilizes the fact that there exist redundant fields in protocol headers, which can be used for hiding data and creating covert channels. Wolf [124] proposed the reserved fields, pad field or undefined fields from the frames of IEEE 802.2, 3, 4, and 5 networks to be used for this intention. Similar, Handel and Sandford [46] proposed the reserved and unused fields from other protocol headers for same purpose. Another type are the fields filled with "random" data, such as the IP *Identification* or TCP *Initial Sequence Number (ISN)* fields. But naively filling them with data is detectable by passive warden [85], because these fields naturally exhibit sufficient structure and non-uniformity to be efficiently and reliably differentiated from unmodified PDU.

The work of Rowland [98] is first proof of concept of the existence and the exploitation of covert channels in TCP/IP protocol suite, by their concrete implementation. From the attacker's point of view, network storage channels are preferable than timing channels, because of the synchronization issues present in timing channels, their complexity, noisiness, and their significantly lower bandwidth in comparison to storage channels. Network-based covert channels can be used to coordinate distributed denial of service attacks, spreading of computer viruses and worms, for secret communication between terrorists and criminals, but also for secure network management communication [35], for bypassing the organization firewall, for transmitting authentication data [24] (like port knocking), circumventing the limitation in using Internet in some countries (Infranet [33], Collage [16], FreeWave [50]), for improving the security [76] of or providing QoS [75] for VoIP traffic, for watermarking of network flows (RAINBOW [48], SWIRL [49]), for tracing encrypted attack traffic or tracking anonymous peer-to-peer VoIP calls [121, 122], etc. Jones et al [54] propose a novel way to locate the entry point of an IP flow into a given network domain based on a marking method using the IPv4 header *TTL* field as a covert channel to carry the information. Recent P2P services are not immune to network steganography, so one can hide messages in Skype communication (SkyDe, [74]) by replacing the encrypted silence with secret data bits (about 2 kbps steganographic bandwidth), or in BitTorrent traffic (StegTorrent, [59]) by using the fact that, in BitTorrent there are usually many-to-one transmissions, and that for its protocol μTP, the header provides means for packets' numbering and retrieving their original sequence (about 270 bps steganographic bandwidth). FreeWave [50] circumvents censorship by modulating Internet traffic into the acoustic signals carried over VoIP connections, and they present one prototype using Skype. One can hide messages using the Google Suggest service as a carrier, by StegSuggest [12], and up to 100 bits can be inserted per suggestions list sent by this service. Even Web counters can be used to create covert storage channel (WebShare, [68]). New trends in network steganography are given in [129].

One very good and comprehensive survey of network covert channels before 2007, classified according to used techniques, is given in [126]. Other surveys are older, and present only several techniques [65, 108], or techniques for TCP, IP and IPSec only [4]. Most of the covert channels have their implementation, and one survey of several covert channel's implementations are given in [107]. Our paper offers an up-to-date comprehensive survey of the covert channels in TCP/IP protocol stack classified by affected layer and affected protocol, with included advantages, disadvantages and defence mechanisms if exist.

Some techniques are protocol independent, like Perkins [93] covert channel using sum of all bits in a message. At the beginning, sender and receiver need to agree about the maximum possible sum S and the number N_i of intervals in $[0, S]$. If the bitwise summation belongs to the particular interval, this correspondents to transfer of particular group of bits. This channel can send $log_2 N_i$ bits per packet. Some network covert channels are inter-protocol steganography solutions, which use the relation between two or more protocols from the TCP/IP stack to enable secret communication. Jankowski et al [51] proposed PadSteg, which utilizes ARP and TCP protocols together with an Etherleak vulnerability (improper Ethernet frame padding) to facilitate secret communication for hidden groups in LANs. Dong et al [28] suggest packet classification, where one carrier (for example, IP packet) is chosen, and several features (several fields from different protocols or timing features) are modulated, so the message can be mapped first in the sorted vector, and then in the selected features.

2. Steganography in Internet layer

Internet Protocol (IP) is the primary protocol in the TCP/IP protocol stack, that operates in the Internet layer. IP encapsulates obtained segments from Transport layer in packets with IP header, and delivers them from a given source to a given destination using IP addresses. It enables internetworking, offering connectionless datagram service. IP comes in two versions - version 4 (IPv4) and version 6 (IPv6). Table 1, 2 and 3 summarize the covert channels in the Internet Protocol, together with their advantages, disadvantages and defence mechanisms. Additionally, for storage channels their RBR or PRBR are given.

One group of of steganographic technics for IP uses fields from the IP header, that have some redundancy or normally are not used during the transmissions, such as *Identification, Flags, Fragment Offset* and *Options* in IPv4. Main drawback to all these channels is easy elimination by traffic normalizers, and main advantage is big covert rate. Rowland [98] uses the 16-bits long *Identification* field from IPv4 header, in which he places the character' ASCII value multiplied by 256. If fragmentation occurs, the receiver will receive the same information for every new fragment of the datagram. One can use the redundancy in the IP's fragmentation strategy [2, 3]. An unfragmented datagram has all zero fragmentation information (*More Fragment* (MF) = 0, *Do Not Fragment* (DF) = 0, *Fragment Offset*=0). If communication parties know the MTU (Maximum Transmission Unit) of their network, they can use the DF bit for sending 1-bit data per packet, or combination of DF bit and *Identification* field for sending 17-bit data per packet, by sending packets with sizes below the MTU. If communication parties do not now the MTU, they can send 8-bit per packet, by filling high 8 bits of the *Identification* field (the low 8 bits are generated randomly) with result of xoring of the first fixed 8-bits of the IP header and 8-bit data. Only condition is that the packet must not contain options in the IP header. Cauich et al [20] use the *Identification* and *Fragment Offset* fields for hiding messages. Their method provide 29 bits in every datagram that is not fragmented, but this can work only for two neighbouring nodes. First, they check if the receiving datagram has fragmentation ($MF = 1$). In negative case, they use some of the three reserved bits in the header as indicator does the datagram carries a message or not, and then they put the data in the datagram's *Identification* and *Fragment Offset* fields.

Some of the steganographic techniques, and presented in Mazurczyk and Szczypiorski paper[79], also deploy the IP fragmentation process. The authors suggest:

- dividing the original IP packet into predefined number of fragments (for example, even number will be binary 0, and odd number will be binary 1) - 1 bit per packet is send;
- modulating the values that are inserted into *Fragment Offset* field (for example, even value will be binary 1, and odd value will be binary 0) - $N_F - 1$ bits per packet are send, where N_F is the number of fragments for that packet;
- using legitimate fragment with steganogram inserted into payload - $N_F \cdot F_S$ bits per packet are send, where N_F is the number of fragments for that packet and the F_S is the size of the fragment. It has big covert rate and uses legitimate fragments, so it is harder to detect. Authors even suggested a method for the problem of differentiating the covert fragments by applying hash function;
- using different rates for packet fragmentation (for example, one rate will be binary 1, and other will be binary 0) - $log_2 h$ bits per packet are send, where h is the number of packets generation rates.

Covert channels can be created using fields in protocol's header that are changing during transmission, like 1-bit-per-packet noisy covert channel using *Time To Live* (TTL) field, suggested by Qu et al [95]. Zander et al [125] proposed an improved 1-bit-per-packet covert channel encoding in the TTL field, analysing initial TTL values and normal TTL occurring in networks. They suggest using two different starting values of TTL in packets, the typical initial value as High-TTL (binary 1) and High-TTL -1 as Low-TTL (binary 0).

Table 1. **Covert storage channels for IPv4.**

Paper	Year	Method	PRBR	Advantages	Disadvantages	Defence
Rowland [98]	1997	*Identification* field	16	Big covert rate	Easy elimination	Traffic normalizer and active warden
B0CK [120]	2000	*Source address* + IGMP	32	Easy deployment	Easy elimination	Egress filtering
Abad [1]	2001	*Header checksum*	16	Deployment in IP, TCP, UDP	Need to know original *TTL*	
Ahsan et al [2]	2002	*DF* field	1		Easy elimination	Traffic normalizer and active warden
Ahsan [3]	2002	*DF* and *Identification* fields	17	Big covert rate	Easy elimination	Traffic normalizer and active warden
Ahsan [3]	2002	*Version, IHL* and *Identification* fields	8	Big covert rate	Packet without options	Traffic normalizer and active warden
Cauich et al [20]	2002	*Identification* and *Fragment Offset* fields	29	Big covert rate	Only between neighbouring nodes and not fragmented packet	Traffic normalizer and active warden
Qu et al [95]	2004	*TTL*	1	Hard detection	Without normal initial TTL	Active warden [125]
Zander et al [125]	2006	*TTL*	1	Normal initial TTL	Noisy channel	Active warden [125]
Trabelsi et al [114, 115]	2007	Record route options	320	Big covert rate		
Mazurczyk et al [79]	2009	Predefined number of fragments	1	Big covert rate	Detectable statistical patterns	Statistical tests based on number of fragments, or reassemble in intermediate node [79, 80]
Mazurczyk et al [79]	2009	Modulating the values in *Fragment Offset* field	$N_F - 1$	Big covert rate	Detectable statistical patterns	Statistical tests based on fragments sizes, or reassemble in intermediate node [79, 80]
Mazurczyk et al [79]	2009	Steganogram in fragment payload	$N_F \cdot F_s$	Use of legitimate fragments	Differentiating covert fragments	Reassemble in intermediate node [79, 80]
Mazurczyk et al [79, 80]	2009	Probe messages in PMTUD	up to MTU	Big covert rate	Need to know MTU	Comparing probe messages [79, 80]

Servetto et al [103] introduces intentional losses in numbered stream of packets for creating a covert channel using phantom packets. They skip one sequence number at the sender so no user data is lost. Loss that occurred during fixed time interval is equal to sending one bit. The authors of [79, 80] use exactly the same technique for creating phantom fragments, by skipping one *Fragment Offset* value. For this to work correctly a modified receiver is needed.

Ahsan and Kundur [2, 3] showed how to create a covert timing channel using packet sorting. If one send n packets and if the network guarantee proper sequencing for packet delivery, then by reordering the packets, one can send $log_2 n!$ bits. For this to be possible, a reference to relate sorted packet numbers to their actual natural order is needed. This can be found in a 32-bit *Sequence number* field of the Authentication Header (AH) and *Option Data Length* and *Option Data* fields used in IPSec. In [2] an algorithm for best sequence estimation in resorting, applicable in real networks which do not guarantee the order of delivered packets, is given. The authors of [79, 80] use exactly the same reordering technique but for sequence of fragments of a given packet, with obtained

PRBR of $log_2 n!$, where n is the number of fragments. Another reordering scheme, robust and resilient to external reordering effects and with built-in error detection/correction capabilities is given in [31]. By using only a subset of the permutations, the channel is equipped with error detection and correction capabilities, and is extremely robust and resilient to external reordering effects. PRBR is higher than 2 bits per packet. Galatenko et al [37] suggested statistical covert channel through a PROXY server that transfers information by reordering packets so that destination addresses in a series of subsequent packets are ordered.

IPv4 and IPv6 have mechanisms for discovering Path MTU (PMTU) - the smallest, acceptable MTU along the entire end-to-end path. They are PMTUD (Path MTU Discovery) for IPv4 and PLPMTUD (Packetization Layer Path MTU Discovery) for IPv6. The first one use probe messages with DF flag set and ICMP for receiving notifications. The second one learns about PMTU by starting with packets of the relatively small size and when they get through, the progressively larger ones are send. Probe messages are validated at the transport layer, without help of the ICMP. One can utilize the probe messages in PMTUD (already knowing the MTU) to carry steganogram and invoke sending intentional fake ICMP messages by receiver [79, 80]. Covert probe messages can be distinguished by use of hash function, and fake ICMP messages can be distinguished by modifying TTL. RBR can be expressed as $\sum_{i=1}^{i=n} P_i/T$ bps, where n denotes number of probes sent from sender to receiver, P_i probe payload size and T connection duration. The authors also suggest use of RSTEG (Retransmission Steganography) [81] for steganographic PLPMTUD, which uses intentional retransmissions to sent steganograms.

Abad [1] demonstrated how a fundamental flaw in the design of the Internet checksum can allow a malicious user to embed covert channel data in the 16-bit *Header checksum* field itself using a hash collision. To obtain hidden message, the receiver need to know and include original TTL value before calculating the checksum.

Padlipsky et al [90] suggested a covert timing channel, in which the sender send or not send an IP packet in an arranged time interval. If a second is divided in N intervals, the RPR of this channel is N bps. Cabuk et al [17] implemented this idea. Client program listens on a given port for arriving of the first PDU. Additionally, the server send not only data bits, but also synchronization and error-correcting bits. One advantage of this channel is that a loss of packet does not affect the synchronization. Big disadvantage is creating of IPDs pattern if same time interval is used.

Berk et al [11] use the inter-packet delays (IPDs) of consecutive packets for encoding the covert information. In their system all the delays are stored and an average is calculated every time a new delay comes in. Every delay above the mean is decoded as binary one, and every delay below the mean is decoded as binary zero. For ex-filtrating the typed data, it is enough for the attacker to put some device between the keyboard and the victim computer (Keyboard JitterBug [104]). In some applications (SSH, Telnet, etc) each keypress corresponds to a packet being sent out on the network, and by introducing small delays in keypresses, a passive covert timing channel can be created. In JitterBug, to encode a binary 0, the added delay is multiply of the timing window w, and to encode binary 1, the added delay is multiply of $[\frac{w}{2}]$. Cabuk [18] presents improved timing channel based on replay attack, which uses saved real IPDs samples divided in two groups. Binary 1 is send by randomly

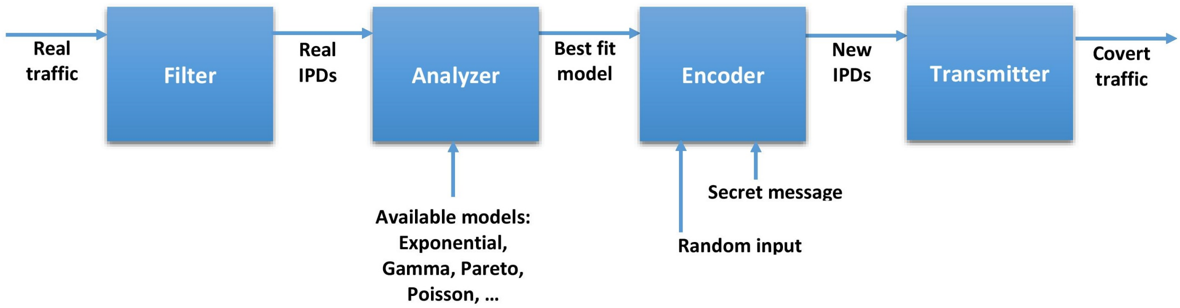

Figure 1. Gianvecchio et al [40] framework for building model-based covert timing channels with inter-packet delays.

replaying IPD from one group, and binary 0 by randomly replaying IPD from other group.

Timing channels, suggested in [40] and [102] model and mimic the statistical properties of inter-packet delays of legitimate traffic, and they are computationally non-detectable if IPDs are independent identically distributed (iid). For example, this holds for real Telnet traffic [91]. Gianvecchio et al [40] offer a framework, with filter, analyzer, encoder, and transmitter (Figure 1). The filter profiles the real traffic, and the analyzer fits the real traffic behaviour to a model (Exponential, Pareto, Piosson, etc). Then, based on the model, the encoder chooses the appropriate distribution functions from statistical tools and traffic generation libraries to create covert timing channels, and finally, the transmitter generates covert traffic, blending it with real traffic. The channel from [102] uses intentional IPDs of consecutive packets for encoding L-bit binary strings in a sequence of n packet inter-transmission times $T_1, T_2, \ldots, T_n$, and masking the secret message with random numbers obtained by CSPRNG using one-time pad encryption technique. One disadvantages of the last two channels is that they require accessible sequence numbers in the legitimate traffic for synchronisation, which is usually available in TCP traffic, but not always in UDP traffic. Another timing channel with reduced robustness against network jitter as drawback is given in [127]. This channel has better synchronisation obtained from using keyed hash function instead of CSPRNG for obtaining random numbers, and is stealthier because it encodes covert bits only into the least significant part of IPDs.

In [114, 115] a novel covert channel is suggested based on the use of the "traceroute" command and IP header Record route options, which can have length up to 40B. Trabelsi and Jawhar [116] give a novel covert file transfer protocol (CFTP) based on the IP Record route option. The proposed protocol is based on a novel session-oriented mechanism that offers TCP-like features embedded inside the IP option field.

Allix [4] gives the example of the following timing covert channel: let the attacker has the control of two machines A and B, each one having a connection to the same server C. If the machine A sends a packet and then the machine B sends two packets, this can be interpreted like a 0. If the machine A send two packets and then the machine B one, this can be interpreted like a 1. Allix suggests a counting covert channel also, with a chosen arbitrary number λ. During a connection if the number of transferred data is inferior to λ, this codes a binary

Table 2. **Covert timing channels for IP.**

Paper	Year	Method	Advantages	Disadvantages	Defence
Padlipsky et al [90]	1978	Send or not IP packet in interval	Good synchronization	Abnormal shape and regularity	ϵ-similarity test [17] EN and CCE test [41]
Servetto et al [103]	2001	Phantom packets		Need of synchronization	
Ahsan et al [2] Ahsan [3]	2002	Packet sorting		External reordering effects	
Galatenko et al [37]	2005	Packet sorting with ordered dest. addresses	Different error rate with sequence length	External reordering effects	
Shah et al [104]	2006	IPDs	Similar shape and regularity as normal traffic	Needs access to the link between keyboard and computer	EN test [41]
Cabuk [18]	2006	IPDs	Real saved IPDs samples	Abnormal regularity	CCE test [41]
Allix [4]	2007	Sending more or less data than λ	Easy deployment		
Gianvecchio et al [40]	2008	IPDs	Similar shape and regularity as normal traffic	Bad synchronization and easy detection for non iid normal IPDs	Auto-correlation test [127] CCE test [41]
Sellke et al [102]	2009	IPDs	Non-detectable if normal IPDs are iid	Bad synchronization and easy detection for non iid normal IPDs	Auto-correlation test [127]
Mazurczyk et al [79]	2009	Different rates for fragments	No need of synchronization		Reassemble in intermediate node [79, 80]
Mazurczyk et al [79]	2009	Phantom fragments	No need of synchronization	Modified receiver	
Mazurczyk et al [79]	2009	Fragment sorting	No need of synchronization	External reordering effects	
El-Atawy et al [31]	2009	Packet sorting	Error detection and correction		
Zander et al [127]	2011	IPDs	Better synchronisation and stealthiness than [40] and [102]	Reduced robustness against network jitter	

zero, if the number of transferred data is superior or equal to λ, this codes a binary one.

Graf [43] suggests using of IPv6 Destination Options header for hiding the message which is first TLV encoded. Lucena et al [66] analysed several covert storage channels in headers of IPv6 and for some of them, the sender must compute the *ICV* including the covert data. Manipulation of the IP header can be done in several ways:

- by setting false traffic in the 8-bit *Traffic class*,
- by setting false flow in the 20-bit *Flow Label*,
- by setting false source address in the 128-bit *Source Address*,
- by setting an initial *Hop Limit* value and manipulating the value of subsequent packets. A drawback of this channel is that packets do not necessarily travel the same route, so the number of intermediate hops may vary. By modifying n packets, $n-1$ bits are send,

Table 3. **Additional covert channels for IPv6.**

Paper	Year	Method	Type	PRBR	Advantages	Disadvantages	Defence
Graf [43]	2003	*Options* in Destination Options header	storage	max field length	Big covert rate	Easy elimination	Traffic normalizer and active warden
Lucena et al [66]	2005	*Traffic class*	storage	8	Big covert rate	Intermediate node can change the field	Stateful Active Warden [66]
Lucena et al [66]	2005	*Flow Label*	storage	20	Big covert rate	Many flow labels between two hosts	Stateful Active Warden [66]
Lucena et al [66]	2005	*Source Address*	storage	128	Big covert rate	Easy elimination	Egress filtering
Lucena et al [66]	2005	*Hop limit*	storage		Easy deployment	Noise channel	Stateful Active Warden [66]
Lucena et al [66]	2005	*Payload Length*	storage	up to $(2^{16} - \lvert LegP \rvert)$B	Legitimate packet	Failure in the ICV calculation	
Lucena et al [66]	2005	*Option Data Length* and *Option Data* in the Hop-by-hop options header	storage	up to 2038B	Very high rate	Failure in the ICV calculation	Stateless Active Warden [66]
Lucena et al [66]	2005	*Reserved* in Routing header	storage	4B	Big covert rate	Failure in the ICV calculation	Stateless Active Warden [66]
Lucena et al [66]	2005	fabricating addresses in Routing header	storage	up to 2048B	Very high rate	Many different routing headers between two nodes	Stateful Active Warden [66]
Lucena et al [66]	2005	*Option Data Length* and *Option Data* in the Destination options header	storage	up to 2038B	Very high rate	Failure in the ICV calculation	
Lucena et al [66]	2005	False padding in the Destination options header	storage	up to 256B	Very high rate	Failure in the ICV calculation	Stateful Active Warden [66]
Lucena et al [66]	2005	Two *Reserved* fields in Fragment header	storage	$8 + 2$	Big covert rate	Easy elimination	Stateless Active Warden [66]
Lucena et al [66]	2005	*Next Header* field in Fragment header	storage	8	Big covert rate	Different Next Header values among fragments of same packet	Stateful Active Warden [66]
Lucena et al [66]	2005	fake fragment	storage	up to 64KB/ fragment	Very high rate	Failure in the ICV calculation	Anomalies of single, unrelated fragments [79, 80]
Lucena et al [66]	2005	*Reserved* in Authentication header	storage	2B	Big covert rate	Easy elimination	Stateless Active Warden [66]
Lucena et al [66]	2005	Fake Authentication header	storage	up to 1022B	Very high rate	Sequence number values do not increase monotonically	Stateful Active Warden [66]
Lucena et al [66]	2005	Fake ESP header	storage	up to 1022B	Very high rate	Sequence number values do not increase monotonically	Stateful Active Warden [66]
Lucena et al [66]	2005	False padding value in ESP header	storage	up to 255B	Very high rate		
Mazurczyk et al [79, 80]	2009	PLPMTUD using RSTEG	storage	up to MTU	Very high rate	Use of normal network retransmission rate	Comparing probe messages [79, 80]

- by setting a valid value to add an extra extension header in the 8-bit *Next Header* field, or by increasing value of the *Payload Length* and append extra data at the end of the packet,
- by modification of the *Option Data Length* and *Option Data* fields in the the Hop-by-hop options header. For false router alert, PRBR is 2B, for false padding value PRBR is up to 256B, and for fabrication of one or more options, PRBR is up to 2038B,
- by 4-bytes *Reserved* field or by fabricating addresses up to 2048 bytes per packet in Routing header with routing Type 0
- by using 8-bit and 2-bit *Reserved* bits, setting false *Next Header* or inserting entire false fragment in the Fragment header. In the last case, authors propose two solutions to avoid this fragment to be included in the reassembly process: by inserting an invalid value in *Identification* field in Fragment header that causes fragment to be dropped, and by inserting overlapping *Fragment Offset* value that causes data to be overwritten during reassembly,
- by manipulating *Option Data Length* and *Option Data* fields with fabricating one or more options (PRBR up to 2038B) or setting false padding values (PRBR up to 256B) in the Destination options header,
- by using 2-bytes *Reserved* field or by creating an entire fake header up to 1022 bytes per packet in the Authentication header,
- by creating entire fake header up to 1022 bytes per packet or by setting false padding value up to 255 bytes per packet in the ESP header.

Some of these channels can make failure in the ICV calculation, and trigger immediate detection, so communication parties need to be careful. Sender need to calculate ICV after inserting covert data, and receiver need to intercept the packet before it reaches its destination.

Internet Control Message Protocol (ICMP) is another connectionless protocol on Internet layer, used to transfer error messages and other information between the nodes. ICMP messages are send encapsulated in IP packets. There are 14 (and 16 deprecated) different types of ICMP messages, which have common only first 4 byte of the 8-byte ICMP header. Project Loki [22, 23] demonstrated a covert channel by putting arbitrary information tunnelling in the payload of ICMP Echo Request and ICMP Echo Reply packets. Additionally, the Loki client allows a remote attacker to wrap and transmit commands in ICMP payloads and the Loki server, unwraps and executes the commands, sending the results back wrapped in ICMP packets. This channel will work for any network device which does not filter the contents of ICMP Echo traffic, and is very simple to deploy.

Other implementations of IP-over-ICMP tunnels are ICMPTX [113], Skeeve [128], ICMP-Chat [84], etc. Their PRBR depends on operating system and implementation, and can goes up to 24B, 56B or more. In Skeeve, hidden sender send an ICMP Echo Request packet to the bounce server with an address of the receiver as a source IP.

Table 4. **Covert channels for ICMP.**

Paper/tool	Year	Method	Type	PRBR	Advantages	Disadvantages	Defence
Loki [22, 23]	1996	Payload of ICMP Echo Request and ICMP Echo Reply packets	storage	up to 24B, 56B or more	IP-over-ICMP tunnel and command execution	Easy elimination	Blocking and [106] stateless model
Ahsan [3]	2002	*Reserved* field in ICMP Router Solicitation Message	storage	32	Easy deployment	Easy elimination	Traffic normalizer and active warden
ICMP-Chat [84]	2003	Payload of ICMP Echo Request and ICMP Echo Reply packets	storage	up to 24B, 56B or more	IP-over-ICMP tunnel, encryption	Easy elimination	Blocking and [106] stateless model
Skeeve [128]	2004	Payload of ICMP Echo Request and ICMP Echo Reply packets	storage	up to 24B, 56B or more	IP-over-ICMP tunnel and command execution	Easy elimination	Egress filtering, blocking and [106] stateless model
ICMPTX [113]	2005	Payload of ICMP Echo Request and ICMP Echo Reply packets	storage	up to 24B, 56B or more	IP-over-ICMP tunnel and command execution	Easy elimination and bad implementation	Blocking and [106] stateless model
Ping tunnel [110]	2005	Payload of ICMP Echo Request and ICMP Echo Reply packets	storage	up to 24B, 56B or more	TCP-over-ICMP tunnel, reliable	Easy elimination	Blocking and [106] stateless model
V00d00n3t [86]	2006	Payload of ICMP Echo Request and ICMP Echo Reply packets	storage		IPv6-over-ICMPv6 tunnel and different options		

The bounce server can replay this packet and forward it to the receiver. ICMP-Chat is a simple console-based chat that uses ICMP packets for communication and AES for data encryption. It offers password protection with SHA-256 and ability to change usage of ICMP codes within the application. These payload tunnels are further examined in [111] for resiliency with modern security approaches. Ping tunnel [110] allows to reliably tunnel TCP connections to a remote host using ICMP Echo Request and Echo Response packets. The tool V00d00n3t [86] can create covert channels using IPv6 and ICMPv6. It requires 4 digit PIN for sender and receiver, allowing multiple streams. Ray and Mishra [96] demonstrate that it is possible to transmit large amounts of data covertly with sophisticated support such as security and reliability, by using ICMP Echo Request packets.

For hiding data in ICMP one can use 32-bit reserved field in ICMP Router Solicitation Message [3].

Internet Group Management Protocol (IGMP) is a protocol for establishing multicast group memberships on IPv4 networks. First covert channel for IGMP is suggested in [3]. Scott [101] suggests using of 8-bit and 16-bit *Reserved* fields in the IGMPv3 Membership Report, which are usually set to zeroes and ignored by the receiver. B0CK [120] is one implementation of IGMP/IPv4 covert channel, which uses the *Source address* field for hiding data. While B0CK packets have the IP Protocol field set to IGMP, they contain no encapsulated IGMP header; the packet is padded with 124 bytes of zeros after the 20-byte IP header.

Dynamic Host Configuration Protocol (DHCP) is another protocol on the Internet layer, that enables a server to automatically assign an IP address to a computer from a defined range configured for a given network.

Rios et al [97] suggested and implemented the following reliable covert channels in the tool HIDE_DHCP:

- by setting the 32-bit *XID* field, which is randomly generated by the client. This is stealthier, but with limited bandwidth,
- by using the *SECS* field, similarly like [42], for transmitting one bit,
- by using the last 10B of the *CHADDR* field, when 48-bits Ethernet MAC address is used,
- 64-byte *SNAME* and 128-byte *FILE* fields consist of null-terminated strings, thus hidden data might be included after this character without negatively impacting other clients or servers. They are set to null by the operating system when they are not carrying their own data, but in some situations they might contain information belonging to the *Options* field, when option 52 (Overload) is included.
- by using variable-length *Option Value* field, and by the number of options used or the way options are ordered. The size of the resulting DHCP messages can be suspicious.

Address Resolution Protocol (ARP) is a protocol for resolution of IP address into a physical address such as an Ethernet address. Ji et al [53] suggest using of the last 8 bits of the *Target protocol address* field for carrying the secret message.

3. Steganography in Transport layer

Transmission Control Protocol (TCP) and **User Datagram Protocol (UDP)** are two protocols that operates on Transport layer. TCP offers a reliable end-to-end connection-oriented service (see Table 5 for covert channel), and UDP offers a connectionless oriented service (see Table 6 for covert channel).

Rowland [98] implements Covert_TCP by using 32-bit *Initial Sequence Number (ISN)* and *Acknowledge Sequence Number* fields from TCP header. Rutkowska [99] has created a passive covert channel NUSHU (based on Covert_TCP), which does not generate any additional traffic, but uses data of the existing one. NUSHU sender modifies the *Initial Sequence Number (ISN)* and *Acknowledge Sequence Number* fields generated by OS. Murdoch and Lewis [85] developed a robust scheme, Lathra, which generates *ISNs* for OpenBSD and Linux, that are almost indistinguishable from those generated by a genuine TCP stack, except by wardens with knowledge of a shared secret key.

Ahsan [3] presented covert storage channels by redundancy present in some combination of six flag bits (URG, ACK, PSH, RST, SYN, FIN). From 64 possible combinations, 29 are valid. If the URG bit is not set, one can use the TCP *Urgent Pointer* field for creating covert channel with 16 bits per packet [3, 45]. One can use *Reserved* field for sending 4 bits per segment [4].

Giffin et al [42] presented a covert timing channel that uses TCP timestamps, by the modification of their low order bit. This method slows the TCP stream so that the timestamps on segments are valid when they are sent.

Table 5. Covert channels for TCP.

Paper/tool	Year	Method	Type	PRBR per segment	Advantages	Disadvantages	Defence
Covert_TCP [98]	1997	*ISN* and *Acknowledge Sequence Number* fields	storage	64	Easy deployment	Differentiation from normal ISNs	SVM [109] and anomaly tests [85]
Ahsan [3]							
Hintz [45]	2002	*Urgent Pointer* field	storage	16			
Abad [1]	2001	*Header checksum*	storage	16	Deployment in IP, TCP, UDP	Need to know original *TTL*	
Giffin et al [42]	2002	TCP timestamps	timing	1	No need of synchronisation	Some timestamps are skipped	Ratio of used different timestamps and total number of timestamps [45]
NUSHU [99]	2004	*ISN*	storage	32	Passive channel	Differentiation from normal ISNs	Anomaly tests [85] and neural networks [117]
Lantra [85]	2005	*ISN*	storage	32	Indistinguishable ISNs		
Chakinala et al [21]	2006	Segment reordering	timing	$log_2 n!$			
Allix [4]	2007	*Reserved*	storage	4	Easy deployment	Easy elimination	Traffic normalizer and active warden
Cloak [69]	2007	N packets over X TCP flows	timing		Reliable	Restricts the decoder location	
TCP-Script [70]	2008	TCP bursts	timing		Robust against network jitter, packet loss and reordering	Differentiable from normal traffic	Deviation score for the burst size, entropy of burst size, and inter ACK-data delay [70]
CLACK [71]	2009	*Acknowledge Sequence Number*	storage	32	Reliable and resilience to adverse network conditions		
RSTEG [81, 82]	2010	retransmissions	storage	max IP payload length	Big covert rate	Use of normal network retransmission rate	Passive warden [81]
ACKLeaks [72]	2011	TCP ACK packets	storage		Do not modify any packets	Errors from ACK losses	

1-bit-per-segment covert channel can be obtained by comparing the low order bit of every TCP timestamp with the current message bit. If they match the segment is sent immediately with generated TCP timestamp, otherwise it is delayed for one timestamp tick and TCP timestamp is incremented. On slow network this channel is hard to detect because the low order bit of the timestamp appears randomly distributed.

Chakinala et al [21] extend the reordering scheme proposed in [2, 3] and create a timing covert channel by reordering of TCP segments and using the *Sequence Number* field and suitably defined mathematical model. Luo et al [71] implemented storage covert channel Clack by modification of the *Acknowledge Sequence Number* field. Luo et al [72] implemented ACKLeaks covert channel which embeds covert messages into pure TCP ACK packets from single or multiple TCP connections, using the combinatorial approach. ACKLeaks can evade content-based detection methods and can be implemented by exploiting the existing TCP connections. Cloak [69] is a

Table 6. **Covert channels for UDP.**

Paper/tool	Year	Method	Type	PRBR per segment	Advantages	Disadvantages	Defence
Abad [1]	2001	*Header checksum*	storage	16	Deployment in IP, TCP, UDP	Need to know original *TTL*	
Fisk et al [34]	2002	Presence/absence of *Checksum*	storage	1	Easy deployment	Easy detection	Active warden [34]
Thyer [112]	2008	*Source address*, *Length* and *Checksum*	storage	up to 6B	Easy deployment	Easy elimination	Active warden and egress filtering [34]

new class of reliable timing covert channels, which encodes a message by a unique distribution of N packets over X TCP flows with ten different encoding and decoding methods, and carefully crafted to mimic the normal TCP flows. Sender can establish HTTP session with a remote server which consists of several TCP flows, or with multiple servers. The warden could not detect Cloak simply based on the presence of multiple TCP flows to the same server, because it is not uncommon to have multiple TCP flows in an HTTP session. In the second case, the decoder should be located on the common routing path. TCP-Script [70] is another implementation of timing covert channel, which embeds the message in the normal TCP data bursts, and exploits TCP's feedback and reliability service to increase the decoding accuracy. The message is represented by an array of positive integers m_i, where $m_i \in [1, M]$ and M is a constant pre-agreed by the encoder and decoder. Each m_i is encoded by a burst of m_i back-to-back TCP data segments, send in the encoding period T_E. Two adjacent data bursts are separated by an appropriate time interval. This channel is robust against network jitter, packet loss and reordering, but has very low capacity and it is easy to differentiate its traffic from legitimate traffic.

Mazurczyk et al [81, 82] apply RSTEG method to TCP by using all TCP retransmission mechanisms: RTO (Retransmission Timeout), FR/R (Fast Retransmit/Recovery) and SACK (Selective ACK). The main idea behind RSTEG is to not acknowledge a successfully received TCP segment in order to intentionally invoke retransmission. The retransmitted segment carries a steganogram instead of user data in the payload field. The intentional retransmissions due to RSTEG should be kept near normal network retransmission rate, to avoid detection. Authors use hash function for marking covert segments. One limitation is the need of using normal network retransmission rates for harder detection.

One covert channel in UDP can be created by presence or absence of the *Checksum* field in the datagram [34], because this field is optional in UDP. This results in PRBR of 1 bit per datagram. Abad [1] method can be applied to TCP and UDP *Checksum* field also. UDP offers 3 fields [112] for carrying secret data up to 6B: *Source address*, *Length* and *Checksum*.

Secure Sockets Layer (SSL) (versions 1.0, 2.0 and 3.0) and its successor **Transport Layer Security (TLS)** (versions 1.0, 1.1 and 1.2) are secure communications protocols for encryption/decryption of data in transfer between two sides. A hybrid covert channel as a combination of simple network covert channel in TCP and subliminal channel in SSL/TLS is given in [6, 7].

Table 7. Covert channels for HTTP.

Paper/tool	Year	Method	Type	PRBR	Advantages	Disadvantages	Defence
Reverse WWW Shell tool [118]	1999	HTTP request/response	storage	varies	Easy deployment	Pre-configured master	Analysing HTTP traffic [14, 88]
Corkscrew [89]	2001	HTTP request/response	storage	varies	SSH over HTTP tunnel	Suspicious SSH traffic	Analysing HTTP traffic [14, 88]
Bowyer [15]	2002	URL parameters or body of GET request	storage	varies	Communication through firewalls	Easy detection if data hide in GET body	Analysing HTTP traffic [14, 88]
Dyatlov et al [30]	2003	HTTP request/response	storage	varies	Very high rate	Easy detection if data hide in GET body	Analysing HTTP traffic [14, 88]
Muted Posthorn [10]	2003	Redirection	storage	up to 1024B	Very high rate		Analysing HTTP traffic [14, 88]
Muted Posthorn [10]	2003	*Referer*	storage	up to 1024B	Very high rate		Analysing HTTP traffic [14, 88]
Muted Posthorn [10]	2003	*Set-Cookie*	storage	up to 4096B	Very high rate		Analysing HTTP traffic [14, 88]
Muted Posthorn [10]	2003	HTML elements and Active contents	storage				Analysing HTTP traffic [14, 88]
Alman [5]	2003	CONNECT method	storage		Establishing VPN		
Eßer et al [32]	2005	Delaying or not a response	timing	1			
HTTunnel [63]	2005		storage		TCP and UDP over HTTP tunnel		Analysing HTTP traffic [14, 88]
Kwecka [61]	2006	consequent linear white space characters (CLRF, SP, HT])	storage	up to 8190B	Very high rate	Easy detection	Inline Filtering Agent [61]
Wondjina [119]	2006	HTTP Entity tags	storage				
Wondjina [119]	2006	*ContentMD5* header	storage	128			
Castro [19]	2006	cookies	storage				
Dunkan et al [29]	2010	*Access-Control-Allow-Origin* and *Content-Location* header	storage				
Dunkan et al [29]	2010	*Date* and *Last-Modified*	storage			*Date* can be rewritten	StegoProxy [13]

4. Steganography in Application layer

Several protocols at the Application layer of the TCP/IP protocol stack can be used for creating covert channels, like HTTP, FTP, DNS, RTP, RTCP, SIP, SDP, etc.

HyperText Transfer Protocol (HTTP). Even most restrictive organizations usually allow the HTTP traffic. Dyatlov et al [30] suggest covert storage channels using header and/or body of the HTTP request/response. There is no limits from the protocol itself in the size of the HTTP header or the body. But the size of all HTTP headers together depends on platform - Apache servers accept headers with size up to 8KB, IIS up to 8KB or 16KB depending on the version. The Reverse WWW Shell tool [118] demonstrates how effective is HTTP in hidden messages delivery. In practice, the client slave application contacts the pre-configured master via an outgoing HTTP Request. The master application send shell commands as HTTP Response packets, and output from commands return from the slave as cgi script HTTP GETs. Bowyer [15] uses HTTP for secret communication with Trojans behind firewalls, with the secret messages encoded as URL parameters or after GET request.

Bauer [10] suggests a protocol "Muted Posthorn" that allows to create an anonymous overlay network by exploiting the web browsing activities of regular users. the protocol uses five HTTP/HTML mechanisms: redirects, cookies, Referer headers, HTML elements and Active contents.

Kwecka [61] hides data in HTTP using the fact that HTTP treats any amount of consequent linear white space characters (optional line feed [CLRF], spaces [SP] and tabs [HT]) present in the header, in the same way as a single space character. For example, [HT] can be a binary one and [SP] can be a binary zero. Headers come in no specified order, so it is possible to embed data in the ordering of the headers. Header names are case-insensitive, so using the different capitalisation of the header values can be used for covert channel. Alman [5] showed that due to a weakness in the CONNECT method in the HTTP protocol, arbitrary connection can be made through a HTTP proxy server and even a VPN can be established. Van Horenbeeck [119] implemented a tool Wondjina that creates a tunnel using the HTTP Entity tags (*ETag* and *If-None-Match* headers), which allows a client to verify whether its locally cached copy is still current. Even a *Content-MD5* header can be used for transferring up to 128 bits of data per HTTP message. Similar ideas are used in [29], together with *Access-Control-Allow-Origin* and *Content-Location* header. Another approach involves modulating the least significant bits of the date-based fields such as *Date* and *Last-Modified*.

Eßer and Freiling [32] suggest covert timing channel using HTTP, in which a web server sends covert data to a client by delaying a response (binary 1) or responding immediately (binary 0). Castro [19] suggest using cookies for creating covert channels in HTTP.

Padgett [89] developed a tool Corkscrew for tunneling SSH over HTTP proxy, and LeBoutillier [63] implemented a tool HTTunnel for tunneling TCP or UDP over HTTP.

Infranet [33] is a framework which uses covert channels in HTTP to circumvent censorship. Infranet's web servers receive covert requests for censured web pages encoded as a sequence of HTTP requests to harmless web pages and return their content hidden inside harmless images using steganography.

File Transfer Protocol (FTP). Zou et al [130] suggested two covert channels into the FTP. The first one encodes covert bits directly into the FTP commands, so if there are N commands, every command will represents $log_2 N$ bits. The second one varies the number of FTP NOOP commands send during idle periods. The number of send NOOP commands is equal to the integer value of the covert data. Before sending a new value, an ABOR command needs to be sent. For FTP, it is normal to send NOOP or ABOR continually to prevent the control connections from entering idle status.

Domain Name System (DNS). DNS is very suitable for creating covert channels for tunnelling other protocols, for example IP, TCP or UDP over DNS. Specially interested are NS, CNAME and TXT records with length up to 255B, and experimental NULL record with length up to 65536B (300B-1200B in implementations).

Two IPv4-over-DNS tools implemented in C, are Nameserver Transfer Protocol - NSTX [100] and Iodine [60].

Table 8. **Covert channels for DNS.**

Paper/tool	Year	Method	Type	PRBR	Advantages	Disadvantages	Defence
NSTX [100]	2002	TXT records	storage	up to 255B	IPv4-over-DNS	Splits IP packets	Blocking by forbidding non-compliant names in queries [87]
DNSCat [94]	2004	CNAME records	storage	up to 255B	IPv4-over-DNS	Splits IP packets	Limit the DNS bandwidth
OzymanDNS [55]	2004	TXT records	storage	up to 255B	TCP or SSH-over-DNS and supports of EDNS0	Must provide a reliable communication	Blocking rarely used records or extensions [87]
Anonymous [8]	2005	Negative caching	timing	1	No need to maintain or run a DNS server	Choosing good subdomains	
DNS2TCP [25]	2008	TXT records	storage	up to 255B	TCP or SSH-over-DNS	Must provide a reliable communication	Blocking by forbidding non-compliant names in queries [87]
TUNS [87]	2009	CNAME records	storage	up to 255B	IPv4-over-DNS, doesn‘t split IP packets	Slower for perfect network	Limit the DNS bandwidth
Iodine [60]	2010	Download - NULL records (or TXT, SRV, MX, CNAME, A records), upload - *Domain Name*	storage	Download - up to 1200B, upload - up to 255B	IPv4-over-DNS and supports of EDNS0	Splits IP packets	Blocking rarely used records or extensions [87]

Both split IP packets into several chunks, send them separately, then reassemble the IP packets at the other endpoint. For encoding data into queries, NSTX uses a non-compliant Base64 encoding, and replies are carried with TXT records. Iodine has support for a DNS extension EDNS0, which allows to use DNS packets longer than the initially chosen 512-byte limit. For download traffic, Iodine uses NULL records (or TXT, SRV, MX, CNAME, A records), and upload traffic is gzziped and Base32 or non-compliant Base64 encoded in the *Domain Name* field from DNS Resource record. DNSCat [94] is a Java IPv4-over-DNS tool which uses CNAME records for download traffic. TUNS [87] is another IPv4-over-DNS tool (in Ruby), which doesn‘t split IP packets. It uses only CNAME records with Base32 encoding for sending data, which are frequently used in normal DNS traffic, so TUNS traffic is harder to filter. TUNS uses caching to resolve the problem with duplicated DNS queries and experiments showed that for perfect network conditions it is the slowest implementation, but in presence of packet loss, performs much better.

OzymanDNS [55] and DNS2TCP [25] are two tools for tunnelling TCP or SSH over DNS, which use TXT records. Their main drawback is that they must provide a reliable communication channel over an unreliable protocol, and thus deal with losses, retransmissions, reordering and duplication of DNS packets.

Covert timing channel for DNS, which uses DNS negative caching, i.e. caching NXDOMAIN answer for non-existent domains, is given in [8]. By querying a previously agreed set of subdomains of one non-existent domain, two hosts can covertly exchange messages, treating each cached subdomain as a binary one, and non-cached

Table 9. **Covert channels for RTP and RTCP.**

Paper/tool	Year	Method	Type	PRBR	Advantages	Disadvantages	Defence
Mazurczyk et al [78]	2008	*Padding* field	storage	8		Easy elimination	Traffic normalizer and active warden
Mazurczyk et al [78]	2008	Extension header	storage	varies	Big covert rate	Easy elimination	Traffic normalizer and active warden
Mazurczyk et al [78]	2008	*Sequence Number* field	storage	16	Hard detection	Only for first RTP packet	
Mazurczyk et al [78]	2008	*Timestamp* field	storage	32	Hard detection	Only for first RTP packet	
Mazurczyk et al [78]	2008	*Timestamp* field	timing	1	No need of synchronisation	Some timestamps are skipped	Ratio of used different timestamps and total number of timestamps [45]
Mazurczyk et al [78]	2008	*NTP Timestamp* in RTCP	timing	1	No need of synchronisation	Some timestamps are skipped	Ratio of used different timestamps and total number of timestamps [45]
Mazurczyk et al [78]	2008	Report blocks in RR and SR in RTCP	storage	up to 160	Big covert rate		
Mazurczyk et al [78]	2008	*authentication tag*	storage	80	Hard detection		Stripping this field [78]
LACK [78]	2008	intentionally delayed packets	timing		Receiver discard packets	Do not exceed accepted level of packet loss	Active warden [78]
Bai et al [9]	2008	*interarrival jitter* in RTCP	storage	32	Use of jitter statistics		
Lizhi et al [64]	2012	Send RTP or (RTP, RTCP) packet	storage	32	Robust with many 0s		

domain as binary zero. There is no need hosts to maintain or run a DNS server, and no need to know each other addresses.

Another interesting area for deployment of covert channels are real-time applications over IP, as for example Voice over IP (VoIP), on-line multi-player games, streaming live A/V, etc. In these applications, usually audio and/or video is transmitted using separated streams by Real-Time Transport Protocol (RTP), supported by its companion RTP Control Protocol (RTCP). One part of the RTP works in the Transport layer above UDP, and second in the Application layer. On the other hand, VoIP has signalling phase at the beginning (before audio transfer using RTP), which is carrying by some signalling protocol, like Session Initiation Protocol (SIP). SIP usually is accompanied by some protocol for description of multimedia sessions, like Session Description Protocol (SDP).

Real-Time Transport Protocol (RTP) and **RTP Control Protocol (RTCP)**. Mazurczyk and Szczypiorski [78] explained how to create covert channels in RTP, using 8-bits *Padding* field, variable-length *Extension header*, randomly generated initial values of the 16-bit *Sequence Number* and 32-bit *Timestamp* fields in the first RTP packet, or applying here Giffin et al [42] method with low order bit of *Timestamp* for creating one-bit-per-RTP packet covert channel (Table 9). The last method can be used also in *NTP Timestamp* field in RTCP, but even more, up to 160 bits per packet covert channel can be obtain from report blocks in Receiver Report (RR) and Sender Report (SR) in RTCP. One can use also security mechanisms' fields in Secure RTP or in RTCP, like up to 80-bit *authentication tag*. In the same paper, authors suggest one method called LACK (Lost Audio Packets Steganographic Method) for creating covert channel, using intentionally delayed (and in consequence lost) packets payloads. The payload of the intentionally delayed packets is used to transmit secret information to receivers aware of the procedure. If the delay of such packets at the receiver is considered excessive, the packets are discarded by a receiver. Additionally, communication parties must consider the accepted level of packet loss for IP telephony and do not exceed it. First practical evaluation of this method is given in [83]. Other RTP payload based covert channels are not of our interests.

Bai et al [9] used the 32-bit *interarrival jitter* field of the RTCP header for creating a covert channel. They used two phases: in the first phase, statistics of the value of the jitter field in the current network are calculated. In the second phase, the secret message is modulated into the *interarrival jitter* field according to the previously calculated parameters. Lizhi et al [64] suggest novel covert timing channel, which utilizes Run Length Code and Multi-Zero Code (to avoid frequently sending RTCP packets) to improve imperceptibility and robustness. The basic idea is very simple, if the current stegobit is the same as previous, an RTP packet is sent, otherwise an RTCP packet and an RTP packet are both sent.

Session Initiation Protocol (SIP) and **Session Description Protocol (SDP)**. Mazurczyk and Szczypiorski [77] suggest to use some tokens and fields in SIP for hiding data, like randomly generated *tag* in *From* field (which forms SIP dialog identifier), *branch* in *Via* field (which forms transaction identifier), *Call-ID* field (which uniquely identifies a call), first part of *CSeq* field (initial sequence number that serves as a way to identify and order transactions), *Max-Forwards* fields and several other fields. They also suggest to hide data in SDP in the fields v (version field ignored by SIP), o (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). Another covert channel 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. For example, *Call-ID* field after *CSeq* field is binary 1, and opposite is binary 0. Fields are not case sensitive, so one can create channel by using upcase for binary 1 and lowercase for binary 0.

The authors of [77] suggest also creating of covert channels in SIP and SDP using security mechanisms for providing authentication and confidentiality. SDP content embedded into the SIP INVITE message, may be encrypted and signed using S/MIME, and the secured parts of the message are divided from themselves using boundary randomly generated value, so this is first possible covert channel. Second possibility is to use the signature bits inside the boundary values (*application/pkcs7-signature*) to transfer covert data.

Secure Shell (SSH) is a cryptographic client-server protocol for secure data exchange, remote command execution, and other secure network services between two computers that connects, via a secure channel over an insecure network. Lucena et al [67] suggest the *MAC* field for carrying messages up to 160 bits per SSH PDU.

Table 10. **Covert channels for SSH.**

Paper/tool	Year	Method	Type	PRBR	Advantages	Disadvantages	Defence
Lucena et al [67]	2004	*MAC* field	storage	up to 160	Randomness simulation	Non reliable	Wrong recomputed MAC value
Lucena et al [67]	2004	Beginning of payload	storage	up to 20B	Big covert rate	Different packet length distribution	Packets length analysis
Perkins [67]	2005	*Random padding* field	storage	up to 255B	Very high rate	Not so random values	Statistical tests

To simulate the randomness of the *MAC*, the embedded messages are previously compressed and then encrypted (Table 10). Another way is intermediate nodes to intercept the SSH traffic and inserts an additional encrypted message (up to 20B) at the beginning of the already encrypted payload. A 4 byte "magic" number at the beginning marks the presence of a hidden message. Hidden message can be carried in the *Random padding* field [93] also, with length up to 255B.

5. Defence Mechanisms

Covert channels need first to be identified or detected, before trying to defeat them. Identification tries to uncover a shared resource in the design phase, that can potentially be used as a covert channel variable, while detection tries to detect an actively running covert channel by examining its output events. There are several formal methods for identifying network covert channels: applying Shared Resource Matrix (SRM) method [56] in [27, 57], Covert Flow Tree (CFT) method [58], etc. Once identified or detected, the covert channel can be eliminated, limited, audited and/or documented [126].

Detecting covert timing channels is a challenging task and is usually based on some statistical tests for differentiating the covert from legitimate traffic. There are two basic classes of detection tests: shape test and regularity test. The first class uses first-order statistics (mean, variance, distribution), while the second class uses second or higher-order statistics. The regularity test is based on the fact that the variance of the legitimate IPDs changes over time, whereas for some timing channels the variance of the IPDs remains relatively constant.

Peng et al [92] uses Kolmogorov-Smirnov test for detection of watermarked IPDs in [121, 122]. Cabuk et al [17] suggest the use of ϵ-similarity test for detecting [17, 90] timing channel, which has a feature of creating clusters of similar IPDs at multiples of some time intervals. This and some other existing detection methods [11, 38] targets only specific timing channels, or can not perform well for high variation of network traffic. The authors of [127] using auto-correlation test showed that [40, 102] timing channels are easy to detect with 80% accuracy and a false positive rate of 0.5%, when IPDs of legitimate traffic are not iid. Detection accuracy of the same test for [127] channel is reduced down to 9%.

The most promising method for detecting covert timing channels based on IPDs is given in [39, 41] and is based on the observation that covert timing channels have effects on the entropy of the original process. The first order entropy (EN) is useful for comparing the shape of distributions of random variables, and the entropy rate estimated by minimum corrected conditional entropy (CCE) is useful for comparing the regularity of time series. Experiments showed that several timing channels can be detected, like: [90] channel (with abnormal shape and regularity), [17] (with similar shape and abnormal regularity), [40] (with similar shape and regularity as normal traffic) and [104] (with similar shape and regularity as normal traffic).

Timestamp channels [42] and [97] with *SECS* field, can be detected by computing the ratio of the number of used different timestamps and total number of possible timestamps. For legitimate connection this ratio is close to 1, and for covert channel is around 0.5 [45], because many timestamps are skipped.

Detection of fragmentation methods given in [79, 80] can be performed by different statistical test based on number of fragments, fragment size, presence of unrelated fragments, and fragment rates. The best method [79, 80] for their elimination is to reassembly original IP packet in the intermediate node responsible for detecting steganographic communication, then refragment it randomly and send to the receiver.

Most of the covert storage channels can be eliminated effectively through protocol scrubbers [73], traffic normalizers [47] and active wardens [34]. Protocol scrubbers and traffic normalizers are focused on eliminating ambiguities found in the traffic stream created by an attacker. Active wardens and traffic normalizers modify incoming and outgoing packets by setting unused, and reserved fields to zero, setting paddings to zero and removing extensions. They can be part of end host or network devices. Using traffic normalization, header fields that are used sometimes, also can be set to zero or rewritten under some conditions. For example, TTL channels [95, 125] can be eliminated by active warden which sets TTLs of all packets of a flow to the same value [125].

Lucena et al [66] suggest three types of active warden: stateless, stateful, and network-aware, for IPv6 channels. Stateless active warden has no recollection of previous packets nor previously encountered semantic conditions. Stateful active warden can also registers already-observed semantic conditions, and applies that knowledge in subsequent monitoring sessions. Network-aware active warden additionally has knowledge of the topology of the surrounding networks.

Several covert channels can be eliminate by blocking protocols/ports by firewalls (Loki [22, 23], ICMPTX [113], Skeeve [128], ICMP-Chat [84]) or ingress/egress filtering (B0CK [120], Skeeve [128], false IPv6 *Source Address* [66]). Another defence mechanisms against ICMP tunnelling are partially disabling ICMP traffic, limiting the size of ICMP packets, preserving ICMP packet for later investigation, and Singh et al [106] stateless model, which zeros out the entire payload of the ICMP packet.

Covert−TCP [98] can be detected using a Support Vector Machine (SVM) [109], and together with NUSHU [99] by [85] anomaly tests, because covert headers are easily distinguished from those generated by a genuine TCP/IP stack.

To fight RSTEG [81], a passive warden with large memory can be used which monitor TCP traffic and store segments. On retransmissions, passive warden compares originally sent segment with retransmitted one and if the payload differs RSTEG is detected and the segment is dropped.

To counter both UDP covert channels [34, 112] anomaly detection and/or egress filtering is needed [34].
By using behaviour profiles of traffic flows Pack et al [88] suggest the method for detecting HTTP tunnels. Their method contains the local level analysis module, which searches for local features of individual session packets; the session level analysis module, which examines the average activities for an entire session up to the current time; and the verification module, which extracts and considers the contents of packets to verify the existence of HTTP tunnel. Borders et al [14] tool Web Tap detects HTTP tunnels, by analysis of HTTP traffic over a training period, design of filters to help detect anomalies in outbound HTTP traffic, and using metrics such as request regularity, bandwidth usage, inter request delay time, and transaction size. Kwecka [**?**] implemented Inline Filtering Agent which detects its channel and channels created by modification of HTTP header names and order. [13] proposed a framework for sanitizing incoming and outgoing HTTP connections for tunnels, with a tool StegoProxy, which can be used for eliminating [29] channel also.
Tunnels over DNS can be detected and/or eliminated by different techniques [87]. NSTX [100] and DNS2TCP [25] can be blocked by forbidding non-compliant names in queries. They, together with Iodine and OzymanDNS and be blocked by not serving queries for rarely used DNS records (TXT, NULL) or extensions (EDNS0). If filtering of CNAME records is not applicable, one can limit DNS covert channel by limiting the DNS bandwidth to several requests per second.
SSH [67] MAC channel can be detected by wrong recomputed MAC value at the receiver end, and other [67] channel can be detected if sender do not care about packet length distribution in the normal SSH traffic.
Several solutions are suggested for limiting the capacity of the covert channels, like: introduction of random noise for covert channel masking, insertion of dummy packets, delaying packets, use of fixed packet rates [52], introduction of noise to all clocks [123], etc.

6. Conclusions

Recent research directions are towards designing of novel covert channels in Internet services, like IP telephony, P2P services (Skype and BitTorrent), social networks (Facebook), new network protocols, cloud computing environment, etc, and their detection, and possible elimination or limitation.
Because covert channels usually have malicious application and represent a serious network threat, this field is very important to network security. Many network covert channels are identified and detected. This is continuous race between hackers and security experts. Surveys of this type are necessary for following trends in this field.

References

[1] Abad, C., IP checksum covert channels and selected hash collision, Technical report, University of California, 2001

[2] Ahsan, K., Kundur, D., Practical data hiding in TCP/IP, In proceedings of: Multimedia and Security Workshop at ACM Multimedia 2002, Juan-les-Pins, France, December 1-6, 2002

[3] Ahsan, K., Covert channel analysis and data hiding in TCP/IP, Master thesis, University of Toronto, 2002

[4] Allix, P., Covert channels analysis in TCP/IP networks, IFIPS School of Engineering, University of Paris-Sud XI, Orsay, France, 2007

[5] Alman, D., HTTP Tunnels Though Proxies, SANS Institute, 2003

[6] Anjan, K., Abraham, J., Behavioral Analysis of Transport Layer Based Hybrid Covert Channel, In proceedings of: Third International Conference of Recent Trends in Network Security nd Applications (CNSA 2010), Chennai, India, July 23-25, 2010, Springer Berlin Heidelberg, 83–92

[7] Anjan, K., Abraham, J., Jadhav, M.: Design of Transport Layer Based Hybrid Covert Channel Detection Engine, arXiv:1101.0104

[8] Anonymous, DNS Covert Channels and Bouncing Techniques, Available at: http://seclists.org/fulldisclosure/2005/Jul/att-452/p63_dns_worm_covert_channel.txt, 2005

[9] Bai, L. Y., Huang, Y., Hou, G., Xiao, B., Covert Channels Based on Jitter Field of the RTCP Header. In: Pan, J.-S., Niu, X., Huang, H.-C., Jain, L. C. (Eds.) Proceedings of the Fourth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, IEEE Computer Society, 2008, 1388–1391

[10] Bauer, M., New Covert Channels in HTTP: Adding Unwitting Web Browsers to Anonymity Sets. In Proceedings of Workshop on Privacy Electronic Society (WPES 2003), Washington, DC, USA, October 30, 2003, 72–78

[11] Berk, V., Giani, A., Cybenko, G., Detection of Covert Channel Encoding in Network Packet Delays, Technical Report TR2005-536, Department of Computer Science, Dartmouth College, 2005

[12] Bialczak, P., Mazurczyk, W., Szczypiorski, K., Sending Hidden Data via Google Suggest, arXiv:1107.4062

[13] Blascoa, J., Hernandez-Castrob, J. C., de Fuentes, J. M., Ramosa, B, A framework for avoiding steganography usage over HTTP, Journal of Network and Computer Applications 35 (1), 491–501, 2012

[14] Borders, K., Prakash, A. Web Tap: Detecting Covert Web Traffic, in proceedings of: 11th ACM Conf. Computer and Communications Security (CCS), 110-120, October 2004

[15] Bowyer, L., Firewall Bypass via protocol Steganography, Network Penetration, 2002

[16] Burnett, S., Feamster, N., Vempala, S., Chipping Away at Censorship Firewalls with User-Generated Content, In proceedings of: the 19th USENIX conference on Security (USENIX Security'10), Washington, DC, August 11-13, 2010

[17] Cabuk, S., Brodley, C. E., Shields, C., IP covert timing channels: Design and detection, In Proceedings of: the 11th ACM Conference on Computer and Communications Security, Washington DC, USA, 2004, ACM Press, 178–187

[18] Cabuk, S., Network covert channels: design, analysis, detection and elimination, PhD thesis, Purdue University, 2006

[19] Castro, S., Gray World Team: Cooking Channels, hakin9 Magazine, May 2006, 50–57

[20] Cauich, E., Gómez, R., Watanabe, R., Data Hiding in Identification and Offset IP Fields, In Ramos, F. F., Rosillo, V. L., Unger, H. (Eds.) Proceedings of 5th International School and Symposium of Advanced Distributed Systems (ISSADS) 2005, LNCS vol. 3563, Springer, 118–125

[21] Chakinala, R., Kumarasubramanian, A., Manokaran, R., Noubir, G., Pandu Rangan, C., Sundaram, R., Steganographic communication in ordered channels, In proceedings of: the 8th International Conference on Information Hiding Workshop, Alexandria, VA, USA, July 10-12, 2006, Springer-Verlag Berlin, Heidelberg, 42–57

[22] daemon9, AKA, and route, Project Loki, Phrack Magazine, 49, article 6, 1996

[23] daemon9, Loki2 (the implementation), Phrack Magazine, 51, article 6, 1997

[24] deGraaf, R., Aycock J., Jacobson Jr., M., Improved Port Knocking with Strong Authentication, In proceedings of: the 21st Annual Computer Security Applications Conference (ACSAC 2005), Tucson, AZ, December 5-9, 2005

[25] Dembour, O., Collignon, C., DNS2TCP. Available at: http://hsc.fr/ressources/outils/dns2tcp/, 2008

[26] Department of Defence, Department of defence trusted computer system evaluation criteria, Technical Report DOD 5200.28-ST., Supersedes CSC-STD-001-83, December 1985

[27] Donaldson, A. L., McHugh, J., Nyberg, K. A., Covert Channels in Trusted LANs, in proceedings of: 11th NBS/NCSC National Computer Security Conference, October 1988, 226–232.

[28] Dong, P., Qian, H., Lu, Z., Lan, S., A Network Covert Channel Based on Packet Classification, International Journal of Network Security, 2012, 14(2), 109–116

[29] Duncan, R., Martina, J.E., Steganographic Message Broadcasting using Web Protocols, In proceedings of: Simposio Brasilerio de Seguranca (SBSeg 2010), Fortaleza, Brasil, 2010

[30] Dyatlov, A., Castro, S., Exploitation of data streams authorized by a network access control system for arbitrary data transfers: tunneling and covert channels over the HTTP protocol, Gray-world, USA, June, 2003

[31] El-Atawy, A., Al-Shaer, E., Building Covert Channels over the Packet Reordering Phenomenon, in proceedings of: IEEE INFOCOM 2009, 2186–2194

[32] Eßer, H.-G., Freiling, F. C., Kapazittsmessung eines verdeckten Zeitkanals ber HTTP, Technical Report TR-2005-10, Universitt Mannheim, 2005

[33] Feamster, N., Balazinska, M., Harfst, G., Balakrishnan, H., Karger, D., Infranet: Circumventing Web Censorship and Surveillance, In proceedings of: the 11th USENIX Security Symposium, San Francisco, CA, August 8-12, 2002, 247–262

[34] Fisk, G., Fisk, M., Papadopoulos, C., Neil, J., Eliminating steganography in Internet traffic with active wardens. In Petitcolas, F. A. P. (Ed.) 5th International Workshop on Information Hiding (IH), 2002, LNCS vol. 2578, 18–35

[35] Forte, D. V., SecSyslog: An Approach to Secure Logging Based on Covert Channels, In proceedings of: First International Workshop of Systematic Approaches to Digital Forensic Engineering (SADFE 2005), Taipei, Taiwan, November 7-10, 2005, 248–263

[36] Gallagher, P. R., A Guide to Understanding Covert Channel Analysis of Trusted Systems, National Computer Security Center, USA, 1993

[37] Galatenko, A., Grusho, A., Kniazev, A., Timonina, E., Statistical Covert Channels Through PROXY Server, In proceedings of: Third International Workshop on Mathematical Methods, Models, and Architectures for Computer Network Security, St. Petersburg, Russia, 2005, LNCS vol. 3685, 424–429

[38] Giani, A., Berk, V., Cybenko, G., Covert channel detection using process query systems, in proceedings of: FLoCon 2005

[39] Gianvecchio, S., Wang, H., Detecting Covert Timing Channels: An Entropy-Based Approach, in proceedings of: ACM CCS 2007, Alexandria, USA, October 28-31, 2007

[40] Gianvecchio, S., Wang, H., Wijesekera, D., Jajodia, S., Model-based covert timing channels: Automated modeling and evasion, in: Lippmann, R., Kirda, E., Trachtenberg, A. (eds.) RAID 2008, LNCS, vol. 5230, Springer, 2008, 211–230

[41] Gianvecchio, S., Wang, H., An Entropy-Based Approach to Detecting Covert Timing Channels, IEEE Transactions on Dependable and Secure Computing, vol.8, issue 6, 2011, 785–797

[42] Giffin, J., Greenstadt, R., Litwack, P., Tibbetts, R., Covert Messaging Through TCP Timestamps, In Dingledine, R., Syverson, P. (Eds.) Proceedings of the 2nd international conference on Privacy enhancing technologies (PET 2002), San Francisco, CA, April 14-15, 2002, 194–208

[43] Graf, T., Messaging over IPv6 Destination Options, Swiss Unix User Group, 2003

[44] Gray, J., Countermeasures and tradeoffs for a class of covert timing channels, Available at: http://repository.ust.hk/dspace/bitstream/1783.1/25/1/tr94-18.pdf, 1994

[45] Hintz, A., Covert Channels in TCP and IP Headers, DefCon 10, Las Vegas, Nevada, August 2 - 4, 2003

[46] Handel, T., Sandford, M., Hiding data in the OSI network model, In Anderson R. (ed.) Information Hiding, Cambridge, U.K., May 30 - June 1, 1996, LNCS vol. 1174, 23–38

[47] Handley, M., Paxson, V., Network intrusion detection: Evasion, traffic normalization, and end-to-end protocol semantics. In proceedings of: the 10th USENIX Security Symposium, Washington, DC, August 13-17, 2001

[48] Houmansadr, A., Kiyavash, N., Borisov, N., RAINBOW: A Robust And Invisible Non-Blind Watermark for Network Flows, In proceedings of: the 16th NDSS Symposium 2009, San Diego, USA, February 8-11, 2009

[49] Houmansadr, A., Borisov, N., SWIRL: A scalable watermark to detect correlated network flows, In proceedings of: the 18th NDSS Symposium 2009, San Diego, USA, February 6-9, 2011

[50] Houmansadr, A., Riedl, T., Borisov, N., Singer, E.,I want my voice to be heard: IP over Voice-over-IP for unobservable censorship circumvention, In proceedings of: the 20th NDSS Symposium 2013, San Diego,

USA, February 24-27, 2013

[51] Jankowski, B., Mazurczyk, W., Szczypiorski, K., PadSteg: Introducing Inter-Protocol Steganography, Telecommunication Systems, 2013, 52(2), 1101–1111

[52] Jeng, A. B., Abrams, M. D., On Network Covert Channel Analysis, in proceedings of: 3rd Aerospace Computer Security Conference, 1987

[53] Ji, L., Fan, Y., Ma, C., Covert channel for local area network, In proceedings of: IEEE International Conference of Wireless Communications, Networking and Information Security (WCNIS 2010), 2010, 316 – 319

[54] Jones, E., Le Moigne,, O., Robert, J.-M., IP Traceback Solutions Based on Time to Live Covert Channel, In proceedings of: 12th IEEE International Conference on Networks (ICON 2004), November 16-19, 2004, 451–457

[55] Kaminsky, D., OzymanDNS, Available at: http://dankaminsky.com/2004/07/29/51/, 2004

[56] Kemmerer, R. A., Shared Resource Matrix Methodology: an Approach to Identifying Storage and Timing Channels, ACM Transactions on Computer Systems (TOCS), vol. 1, no. 3, Aug. 1983, 256–77.

[57] Kemmerer, R. A., A Practical Approach to Identifying Storage and Timing Channels: Twenty Years Later, in proceedings of: Annual Computer Security Applications Conference (ACSAC), Dec. 2002, 109–18.

[58] Kemmerer, R. A., Porras, P., Covert Flow Trees: A Visual Approach to Analysing Covert Storage Channels, IEEE Trans. Software Eng., vol. SE-17, no. 11, Nov. 1991, 1166–1185.

[59] Kopiczko, P., Mazurczyk, W., Szczypiorski, K., StegTorrent: a steganographic method for P2P files sharing service, in proceedings of: IEEE Symposium on Security and Privacy Workshops 2013,San Francisco, CA, May 23-24, 2013, 151–157

[60] Kryo, iodine, Available at: http://code.kryo.se/iodine/, 2010

[61] Kwecka, Z., Application Layer Covert Channel Analysis and Detection, Technical Report, Napier University Edinburgh, 2006

[62] Lampson, B., A Note on the Confinement Problem, Communications of the ACM, 1973, 16(10), 613–615

[63] LeBoutillier, P., HTTunnel, Available at: http://sourceforge.net/projects/httunnel/, 2005

[64] Lizhi, Y., Yongfeng, H., Jian, Y., Bai, L. Y., A Novel Covert Timing Channel Based on RTP/RTCP, Chinese Journal of Electronics, 2012, 21(4), 711–714

[65] Llamas, D., Allison, C., Miller, A., Covert Channels in Internet Protocols: A Survey, In proceedings of: the 6th Annual Postgraduate Symposium about the Convergence of Telecommunications, Networking and Broadcasting (PGNET), Liverpool, UK, June 27-28, 2005

[66] Lucena, N. B., Lewandowski, G., Chapin, S. J., Covert Channels in IPv6, In Danezis, G., Martin, D. (Eds.) Proceedings of: the 5th International Workshop Privacy Enhancing Technologies (PET 2005), Dubrovnik, May 30-June 1, 2005, 147–166

[67] Lucena, N., Pease, J., Yadollahpour, P., Chapin, S. J., Syntax and Semantics-Preserving Application-Layer

Protocol Steganography, In proceedings of: 6th Information Hiding Workshop, Toronto, Canada, May 23-25, 2004, LNCS vol. 3200, 164–179

[68] Luo, X., Chan, E., Chang, R., Crafting Web Counters into Covert Channels, in proceedings of: IFIP 2007, Sandton, South Africa, 2007, 337–348

[69] Luo, X., Chan, E., Chang, R., Cloak: A ten-fold way for reliable covert communications, in proceedings of: ESORICS 2007, Dresden, Germany, September 24 - 26, 2007, LNCS vol. 4734, 283–298

[70] Luo, X., Chan, E., Chang, R., TCP Covert Timing Channels: Design and Detection, In proceedings of: IEEE International Conference on Dependable Systems and Networks With FTCS and DCC, Anchorage, Alaska, June 24-27, 2008, 420 – 429

[71] Luo, X., Chan, E., Chang, R., CLACK: A network covert channel based on partial acknowledgement encoding, In proceedings of: IEEE International Conference on Communications, Dresden, Germany, June 14-18, 2009, 1–5

[72] Luo, X., Zhou, P., Chan, E., Chang, R., Lee, W., A Combinatorial Approach to Network Covert Communications with Applications in Web Leaks, In proceedings of: IEEE/IFIP 41st International Conference on Dependable Systems and Networks (DSN 2011), Hong Kong, June 27-30, 2011, 474 – 485

[73] Malan, G.R., Watson, D., Jahanian, F., Howell, P., Transport and application protocol scrubbing, in proceedings of: the IEEE INFOCOM 2002 Conference, 1381–1390, Tel-Aviv, Israel, 2000

[74] Mazurczyk, W., Karas, M., Szczypiorski, K., SkyDe: a Skype-based Steganographic Method, International Journal of Computers, Communications and Control, 2013, 8(3), 389–400

[75] Mazurczyk, W., Kotulski, Z., New Security and Control Protocol for VoIP Based on Steganography and Digital Watermarking, Annales UMCS Informatica AI, 2006, 5, 417–426

[76] Mazurczyk, W., Kotulski, Z., New VoIP Traffic Security Scheme with Digital Watermarking, In: Górski, J. (ed.) proceedings of: SafeComp 2006, Gdansk, Poland, September 26-29, 2006, LNCS vol. 4166, 170–181

[77] Mazurczyk, W., Szczypiorski, K., Covert Channels in SIP for VoIP Signalling, Communications in Computer and Information Science, 2008, 12, 65–72

[78] Mazurczyk, W., Szczypiorski, K., Steganography of VoIP Streams, In: Tari, Z. (ed.) On the Move to Meaningful Internet Systems (OTM 2008), Monterrey, Mexico, November 9-14, 2008, LNCS vol. 5332, 1001–1018

[79] Mazurczyk, W., Szczypiorski, K., Steganography in handling oversized IP packets, In proceedings of: First International Workshop on Network Steganography (IWNS 2009), Wuhan, China, November 18–20, 2009

[80] Mazurczyk, W., Szczypiorski, K., Evaluation of steganographic methods for oversized IP packets, Telecommunication Systems, 2012, 49, 207–217

[81] Mazurczyk, W., Smolarczyk, M., Szczypiorski, K., Retransmission Steganography Applied, In proceedings of: 2010 International Conference on Multimedia Information Networking and Security (MINES), Nanjing, China, November 4-6, 2010, 846–850

[82] Mazurczyk, W., Smolarczyk, M., Szczypiorski, K., On Information Hiding in Retransmissions, Telecommunication Systems: Modelling, Analysis, Design and Management, vol. 52, issue 2, 1113-1121, 2013

[83] Mazurczyk W., Lost Audio Packets Steganography: The First Practical Evaluation, Journal of Security and Communication Networks, 2012, 5(12), 1394–1403

[84] Muench, M., ICMP-Chat, Available at http://icmpchat.sourceforge.net/, 2003

[85] Murdoch, S. J., Lewis, S., Embedding Covert Channels into TCP/IP. In Barni, M., Herrera Joancomartí, J., Katzenbeisser, S., Pérez-González, F. (Eds.) proceedings of: 7th Information Hiding Workshop, Barcelona, Spain, June 6-8, 2005, LNCS vol. 3727, 247–261

[86] Murphy, R., V00d00n3t - Ipv6 / ICMPv6 Covert Channel, DefCon, Las Vegas, 2006

[87] Nussbaum, L., Neyron, P., Richard, O., On Robust Covert Channels Inside DNS, In proceedings of: IFIP Advances in Information and Communication Technology, 2009, 51–62

[88] Pack, P., Streilein, W., Webster, S., Cunningham, R., Detecting HTTP Tunneling Activities, in proceedings of: 3rd Annual Information Assurance Wksp., June 2002

[89] Padgett, P., Corkscrew, Available at http://www.agroman.net/corkscrew/, 2001

[90] Padlipsky, M. A., Snow, D. W., Karger, P. A., Limitations of End-to-End Encryption in Secure Computer Networks, Technical Report ESD-TR-78-158, Mitre Corporation, August 1978

[91] Paxson, V., Floyd, S., Wide-area traffic: the failure of Poisson modeling, IEEE/ACM Transactions on Networking, Vol. 3 No. 3, 1995, 226–244

[92] Peng, P., Ning, P., Reeves, D., On the secrecy of timing-based active watermarking trace-back techniques, in proceedings of: the 2006 IEEE Symposium on Security and Privacy, Oakland, USA, May 2006

[93] Perkins, M. C., Hiding out in Plaintext: Covert Messaging with Bitwise Summations, Master thesis, Iowa State University, 2005

[94] Pietraszek, T., DNSCat, Available at: http://tadek.pietraszek.org/projects/DNScat/, 2004

[95] Qu, H., Su, P., Feng, D., A Typical Noisy Covert Channel in the IP Protocol, In proceedings of: the 38th Annual International Carnahan Conference of Security Technology, October 11-14, 2004, 189–192

[96] Ray, B., Mishra, S., A Protocol for Building Secure and Reliable Covert Channel, In Korba, L., Marsh, S., Safavi-Naini, R. (Eds.) proceedings of: the Sixth Annual Conference on Privacy, Security and Trust (PST 2008), Fredericton, New Brunswick, Canada, October 1-3, 2008, 246 – 253

[97] Rios, R., Onieva, J. A., Lopez, J., HIDE_DHCP: Covert Communications Through Network Configuration Messages, In proceedings of: 27th IFIP TC 11 Information Security and Privacy Conference, Heraklion, Crete, Greece, June 4-6, 2012, IFIP Advances in Information and Communication Technology vol. 376, 162–173

[98] Rowland, C. H., Covert channels in the TCP/IP protocol suite, DoIS Documents in Information Science, May 1997

[99] Rutkowska, J., The Implementation of Passive Covert Channels in the Linux Kernel, Chaos Communication

Congress, 2004

[100] savannah.nongnu.org, NSTX, Available at http://savannah.nongnu.org/projects/nstx/, 2002

[101] Scott, C., Network Covert Channels: Review of Current State and Analysis of Viability of the use of X.509 Certificates for Covert Communications, Technical Report RHUL-MA-2008-11, Department of Mathematics, Roal Holloway, University of London, 2008

[102] Sellke, S. H., Wang, C.-C., Bagchi S., Shroff, N., TCP/IP Timing Channels: Theory to Implementation, In proceedings of: the 28th Conference on Computer Communications (IEEE INFOCOM), Rio de Janeiro, Brazil, April 19-25, 2009

[103] Servetto, S. D., Vetterli, M., Communication using phantoms: covert channels in the Internet. In proceedings of: IEEE International Symposium on Information Theory (ISIT), Washington, DC, June 24-29, 2001

[104] Shah, G., Molina, A., Blaze M., Keyboards and Covert Channels, In proceedings of: 15th USENIX Security Symposium, Vancouver, Canada, August 2006, 59–75

[105] Simmons, G. J., The prisoners problem and the subliminal channel, In: Chaum, D. (ed.) Crypto '83, Advances in Cryptography, 1984, Springer US, 51–67

[106] Singh, A., Lu, C., Nordstrom, O., dos Santos, A. L. M., Malicious ICMP Tunneling: Defense against the Vulnerability, In Mu, Y., Susilo, W., Seberry, J. (Eds.) proceedings of: the 8th Australasian Conference on Information Security and Privacy (ACISP), Wollongong, Australia, July 9-11, 2003, LNCS vol. 5107, Springer, 226–236

[107] Smith, J. C., Covert Shells, SANS GIAC Reading Room, 2000

[108] Smeets, M., Koot, M., Covert Channels, Research Report, University of Amsterdam, 2006

[109] Sohn, T., Seo, J., Moon, J., A study on the covert channel detection of TCP/IP header using support vector machine, in Perner, P., Qing, S., Gollmann, D., Zhou, J., (eds.) Information and Communications Security, LNCS vol. 2836, 313–324, Springer-Verlag, 2003

[110] Stdle, D., Ping Tunnel, Available at: http://www.cs.uit.no/~daniels/PingTunnel/, 2011

[111] Stokes, K., Yuan, B., Johnson, D., Lutz, P., ICMP covert channel resiliency, In Elleithy, K., Sobh, T., Iskander, M., Kapila, V., Karim, M.A., Mahmood, A. (Eds.) Technological Developments in Networking, Education and Automation, 2010, 503–506

[112] Thyer, J. S., Covert Data Storage Channel Using IP Packet Headers, SANS Institute,2008

[113] Thomer, G., IP-over-ICMP using ICMPTX, available at: http://thomer.com/icmptx/, 2005

[114] Trabelsi, Z., El-Sayed, H., Frikha, L., Rabie, T., Traceroute Based IP Channel for Sending Hidden Short Messages, In proceedings of: First International Workshop on Security Advances in Information and Computer Security (IWSEC 2006), Kyoto, Japan, October 23-24, 2006, LNCS vol. 4266, 421–436

[115] Trabelsi, Z., El-Sayed, H., Frikha, L., Rabie, T., A novel covert channel based on the IP header record route option, Journal International Journal of Advanced Media and Communication, 2007, 1(4), 328–350

[116] Trabelsi, Z., Jawhar, I., Covert File Transfer Protocol Based on the IP Record Route Option, Journal of

Information Assurance and Security, 2010, 5, 64–73

[117] Tumoian, E., Anikeev, M. Detecting NUSHU Covert Channels Using Neural Networks, Technical report, Taganrog State University of Radio Engineering, 2005

[118] vanHauser, Placing Backdoors through Firewalls, The Hacker's Choice, 1999

[119] Van Horenbeeck, M., Deception on the network: thinking differently about covert channels, In proceedings of; Australian Information Warfare and Security Conference, Perth, Western Australia, December 4-5, 2006, 174–184

[120] vecna, B0CK, Butchered From Inside, 2000, 7(2)

[121] Wang, X., Chen, S., Jajodia, S., Tracking anonymous peer-to-peer voip calls on the internet, in proceedings of: the 2005 ACM Conference on Computer and Communications Security, November 2005

[122] Wang, X., Reeves, D. S., Robust correlation of encrypted attack traffic through stepping stones by manipulation of inter packet delays, in proceedings of: the 2003 ACM Conference on Computer and Communications Security, October 2003

[123] Wei-Ming, H., Reducing Timing Channels with Fuzzy Time, in proceedings of: IEEE Computer Society Symposium Research in Security and Privacy, 8–20, 1991

[124] Wolf, M., Covert channels in LAN protocols, In: Berson, T. A., Beth, T. (Eds.) proceedings of: Workshop on Local Area Network Security (LANSEC 89), Karlsruhe, FRG, April 3-6, 1989, LNCS vol. 396, 91–102

[125] Zander, S., Armitage, G., Branch, P., Covert Channels in the IP Time To Live Field, In proceedings of: the Australian Telecommunication Networks and Applications Conference (ATNAC), Melbourne, December 4-6, 2006

[126] Zander, S., Armitage, G., Branch, P., A survey of covert channels and countermeasures in computer network protocols, IEEE Communications Surveys and Tutorials, 2007, 9(3), 44–57

[127] Zander, S., Armitage, G., Branch, P., Stealthier Inter-packet Timing Covert Channels, in proceedings of: 10th International IFIP TC 6 Networking Conference, Valencia, Spain, May 9-13, 2011, LNCS vol. 6640, 458–470

[128] Zelenchuk, L., Skeeve - ICMP Bounce Tunnel, available at: http://www.gray-world.net/poc_skeeve.shtml, 2004

[129] Zielinska, E., Mazurczyk, W., Szczypiorski , K., Development Trends in Steganography. Communications of the ACM 2014, 57(2) (in press)

[130] Zou, X., Li, Q., Sun, S.-H., Niu, X., The Research on Information Hiding Based on Command Sequence of FTP Protocol, In: Khosla, R., Howlett, R. J., Jain, L. C. (Eds.) proceedings of: the 9th International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES 2005), Melbourne, Australia, September 14-16, 2005, LNCS vol. 3683, 1079–1085