Network Covert Channels: Review of Current State and Analysis of Viability of the use of X.509 Certificates for Covert Communications

Carlos Scott

Technical Report
RHUL-MA-2008-11
15 January 2008

Department of Mathematics
Roal Holloway, University of London
Egham, Surrey TW20 0EX, England
http://www.rhul.ac.uk/mathematics/techreports

NETWORK COVERT CHANNELS: REVIEW OF CURRENT STATE AND ANALYSIS OF VIABILITY OF THE USE OF X.509 CERTIFICATES FOR COVERT COMMUNICATIONS

Author: Carlos Scott

Supervisor: Dr Chez Ciechanowicz

Submitted as part of the requirements for the award of the MSc. in Information Security at Royal Holloway, University of London.

September the 7th, 2007

ACKNOWLEDGEMENTS

Thank you Mum and Dad for paving the road on which I walk, for your unconditional support and infinite love.

Thank you Chez and everybody at the ISG for sharing your knowledge and wisdom. Being part of this course truly has been a wonderful experience.

CONTENTS

1 – EXECUTIVE SUMMARY

The popularity of computer-based smuggling has increased as a result of organizations taking measures to prevent traditional means of data exfiltration. Most organizations depend on broad and heterogeneous communication networks, which provide numerous possibilities for malicious users to smuggle sensitive private information out of their boundaries. They can achieve that objective with the use of network covert channels, that apart from carrying the data outside of the organization, hide the fact that the communication is taking place. This study provides a comprehensive, up to date review of the current state of research in the field of network covert channels: hidden communication channels that abuse legitimate network communication channels. It also presents a novel technique to establish such channels based on the use Digital Certificates, along with an informal framework to exfiltrate data making use of the technique. It involves the use of the Transport Secure Layer protocol, a network protocol normally used to provide confidentiality and integrity services to applications.

Several detection and prevention mechanisms and methodologies exist or have been proposed to counter the threats posed by this hidden communication channels. They are also identified and discussed in this work, explaining their applicability and limitations.

2 – INTRODUCTION

In any modern organization, the prospect of leakage of confidential information ranks among the highest fears of any executive. As most of these organizations depend on broad and heterogeneous communication networks, the possibilities for a malicious entity to smuggle sensitive private information out of the organization are numerous, and the detection of such event a challenging problem. It is rarely unambiguous to conclude which information is leaving the organization legitimately, and which communications are unauthorized data exfiltrations. It may even be impossible to determine if a communication is occurring at all.

Furthermore, commercially available computer systems and networks are rarely strictly regulated. However, data transmitted on such systems and networks may still be relatively sensitive. As such, the availability of tools employed to inspect outbound e-mail, web traffic and other forms of network communications has increased significantly, to prevent and identify leakage of confidential data.

At the moment, the most common means of information leakage are employees discussing proprietary information outside their employment context, reproduction of hard copies of classified documents and copying of confidential information on portable media. However, most organizations take measures to prevent them, which leads to an increased popularity of computer based data smuggling.

Network communication channels can be abused to implement covert communication channels. The reliability, speed and robustness of communication protocols allow for the implementation of such channels over networks. Other than data exfiltration, covert channels can be used for a variety of purposes, depending on the goals the user is trying to achieve. Some of these pose serious threats:

- Intelligence agencies and criminals can use them for covert communications. The use of encryption hides the content of communications, but it does not prevent adversaries to detect communication patterns, which is often sufficient

to discover onset anomalous activities. This makes them particularly useful in information warfare scenarios.

- Hackers that have compromised systems normally use them as launching points for subsequent attacks. Covert channels can be used to send instructions to these systems, as if this traffic is not covert it would alert the systems administrator, who would easily discover the compromised systems. They can also be used as backdoors, as the intruders cannot rely on the initial exploitation vector remaining available. [1]
- They can be used to circumvent measures taken by governments and private organizations to limit the freedom of speech and civilian use of strong encryption.

A good indication of the threat they pose can be derived from the highly publicized Distributed Denial of Service (DDoS) attacks conducted against popular internet websites such as Yahoo!, CNN, E-bay, E-trade and Buy.com which occurred in 1999. These were automated using thousands of distributed agents, which communicated with each other through covert channels in network protocols. Another good example that exposes their risk is the accounts surfaced in 2002 of suspected secret, hidden conveyance of plans or instructions through the Internet to terrorist groups operating within the United States. It is believed that many of these messages were transmitted using covert channels, encrypted and embedded within innocent-looking files [2].

A common analogy employed for discussing the dynamics of covert communications is one known as the “prisoner’s problem”[3]. It involves two prisoners, called here for simplicity Alice and Bob, who need to communicate with each other in order to devise an escape plan. It also involves a warden, Wendy, who oversees all inter-prisoner communications, and can monitor them in one of two ways:

- She can examine all messages, and let them pass or deny them based on what she sees. This is a passive approach.
- She can modify the message slightly to make sure it is not precisely what was sent, without changing the meaning of the message. By modifying the

message it is assumed that she might frustrate any attempt of embedding a secret message in the communication. This is an active approach.

Ideally, the prisoners find a way to communicate which doesn't raise suspicion from the warden. But the warden must accept that there is a risk that some covert communication may be attempted, and pose and hypothesis of how it might function.

The main objective of a covert channel is to hide the fact that a communication is taking place. Cryptography is different, as there is no intention to conceal the communication, but rather the objective is to make the information being communicated available solely to the intended receiver.

Covert channels and steganography (Greek for covered writing) are closely related and often confused. Both represent data-hiding techniques and involve piggybacking of messages on legitimate communication channels. Examples of steganography are the manipulation of bits in image or audio files to conceal information. Thus it can be said that steganography avails covert channels to transfer information secretly [4]. Actually, a variety of things can act as a conduit for steganographic communication. A Greek historian tells that a messenger's head was shaved and tattooed with a secret message calling for revolt against the Persians. Later, the messenger travelled to the location of the intended receiver after the hair had re-grown. The head was shaved again and the message revealed to the receiver [5]. While steganography requires some form of content as cover, network covert channels require a network protocol as a carrier. The ubuiquitous presence of some network protocols (e.g. the Internet Protocol) makes covert channels highly available and usable even in situations where steganography cannot be applied. [6]

They are the result of protocol specifications often being vulnerable to 'misuse' in unintended or unanticipated ways. This is due mainly to language peculiarities, lack of use of formal methods to define them, and the complexity of unambiguously postulating mind concepts that form the base of a protocol. They often include extendable and optional elements, which are normally not disallowed explicitly in situations where they have no use. Differences in implementations are also normally allowed, which reduces consistency. Covert channels adhering to protocol

specifications may be established in these cases and their detection is extremely difficult, as the resulting traffic cannot be considered anomalous.

Analysis of covert channels is a long-standing academic discipline. Generally speaking covert channels can be classified by the type of resource that allows for the communication between two entities. Common shared resources comprise timing and storage channels. Timing channels exist when it is possible to influence the response time of a system. On the other hand, storage channels make used of shared storage medium to transmit information. Research on covert channels traditionally focused on storage and timing channels within single systems. However during the last decade the scope expanded to network based covert channels, as systems became interconnected through networks and inter-networks.

The remainder of this study focuses on network based covert channels, as they pose a more obvious risk in information security. This is due in part to the comparative communication bandwidth and the greater availability of tools that enable the exploitation of this type of channels. Storage and Timing channels theory and concepts will be discussed in section 3, underlying the principles of covert communications.

In section 4 an up to date review of the use of common network protocols as covert channel carriers is conducted. It is by no means a definitive guide, as it is included to give an idea of the different techniques available for transmitting data covertly.

A set of novel covert channels is presented in section 5. They allow for covert communications based on the use X.509 Digital Certificates. An informal framework to exfiltrate data using this covert channels using the Transport Layer Secuity protocol is discussed in section 6. The decision to analyze these protocols was based on their property to provide functionalities to multiple applications and application layer protocols, which makes them omnipresent in different types of high security networks and systems.

Detection and prevention of the different types of communication channels described in this study pose a number of serious challenges, which is discussed in section 7.

How the research community has tackled these challenges is also discussed in this section.

Lastly, the conclusions that can be drawn from the results obtained in this study are exposed in section 8.

3 – COVERT CHANNELS THEORETICAL CONCEPTS

The concept of covert channel is difficult to define precisely, and often leads to discussions on the meaning of “covertness”. No formal definition of covertness exists in the field of computer science, although research on detection of covert communications is very active. Informally, an operation is “more” covert if it is difficult to detect without the use of special tools that look for it explicitly [7].

It is generally accepted that the concept of covert channels was first mentioned in the paper “A Note on the Confinement Problem” by B.W. Lampson [8], presented during the communications of the Association for Computer Machinery in 1973. Although the term was introduced in that paper, it restricted its use to a subclass of leakage channels that did not include storage channels and legitimate channels. According to Lampson, “a communication channel is covert if it is neither designed nor intended to transfer information at all”.

Later work defines a covert channel as “a communication channel that allows a process to transfer information in a manner that violates the system’s security policy”[9]. This definition is now more commonly accepted.

For the purpose of this study, a covert channel is defined as a communication channel that is not designed and/nor intended to exist and that can be used to transfer information in a manner that violates the existing security policy.

Previous work is known on the topic of analyzing and modelling covert channels. However, there is no well-defined ontology. Two types of channels are presented on [9]: covert storage channels and covert timing channels. Moreover, the notion of measuring the threat level of a covert channel by looking for covert channels mechanisms with bandwidth that may exceed a certain amount of bits per second is also introduced. According to [9], “covert storage channels include all vehicles that allow the direct or indirect writing of a storage location by one process and the direct

or indirect reading of it by another". Furthermore, "covert timing channels include all vehicles that allow one process to signal information to another process by modulating its own use of the system resources in such a way that the change in response time observed by the second process would provide information".

A more recently published classification model presented in [10] is also based on the dimension in which data is encoded (time, space), but captures the character-encoding paradigm as well (value-based, transition-based). The classification consist of four types of channels:

- **Value-based spatial channel**: This type of channel encodes data within a spatial container (e.g.: using the corresponding representation of a letter from a Unicode character set to covertly communicate the letter through a TCP Initial Sequence Number).
- **Transition-based spatial channel**: This type of channel encodes data by representing it as changes between spatial containers (e.g.: covertly communicating a letter by crafting a TCP packet with a specific source port, and then crafting a second packet with a different source port, the transition between source ports representing the letter).
- **Value-based temporal channel**: This class encodes data by modulating the occurrence of events (time dimension, e.g.: using network packet arrival times to implement a binary channel).
- **Transition-based temporal channel**: This class encodes data by modulating intermediate events on the occurrence of events (e.g.: using network jitter to implement a binary channel).

Other terms of characterization are also commonly used for network covert channels, although on a more informal level [11]. Some of these are:

- **Behaviour**: Indicates how the carrier protocol is used to send data.
    - ***Active:*** The covert channel generates it's own traffic (e.g.: via a purposely developed application).

- ***Passive:*** The covert channel piggybacks on legitimate traffic generated by another process. This characteristic makes it dependent on external events in order to communicate data.

- **Path**: Indicates the logical route that lies between the sender and receiver.
  - ***Direct:*** The sender communicates directly to the receiver.
  - ***Indirect:*** The sender communicates with the receiver through intermediate forwarding or bouncing nodes.
  - ***Spread:*** The sender splits data and sends it to multiple intermediate nodes, which converge the data and send it to the intended receiver.

- **Efficiency**: Indicates the amount of data that can be sent per carrier unit.
  - ***Space***: Normally expressed as the number of bits/bytes per packet (spatial channels) or the number of packets per bit/byte (temporal channels) that can be transmitted through the channel.
  - ***Time***: Normally expressed as the number of bits/bytes per second (spatial channels) or the number of seconds per bit/byte (temporal channels).

Various distinct parameters to characterize covert channels were introduced on [12], the most relevant being*: Noise*, *Bandwidth, Capacity Synchronization* and *Aggregation.*

According to [12], a covert channel is noiseless if the receiver decodes correctly any bit transmitted by a sender with probability 1. Bandwidth is used to denote the rate at which information is transmitted through a channel. In a covert channel context, bandwidth is represented in bits/second, and is also related to the notion of *capacity*, the maximum possible error-free information communication rate in bits per second. Because error-correcting codes help turn a noisy channel into a noise-less one, these two parameters are closely linked. The resulting channel will, however, have a lower bandwidth than the similar noise-free channel [12]. Synchronization between the communicating entities allows them to notify the other when they have completed reading or writing data. Communications channels may be aggregated serially, in

parallel, or in combinations of serial and parallel aggregation to yield optimal bandwidth. All these parameters are further explained in the following section.

It is also possible to characterize channels by their multiplexing/de-multiplexing possibilities and the difference between control and data channels, as described in section 3.1.

3.1 COMMUNICATION CONCEPTS FOR NETWORK COVERT CHANNELS

As network covert channels are communication channels that are not designed nor intended to exist, the communication streams must be embedded inside authorized channels. They may be based on existing protocols from OSI low layers (e.g.: IP, TCP, UDP) to OSI high layers (e.g.: HTTP, SMTP). However the carrier protocol must be authorized by the Network Access Control System (NACS), and trade-offs regarding reliability and covertness must be accepted. Several communication concepts apply to the communication of data through covert channels.

3.1.1 – Control / Data Channels

Covert channels may consist, from an abstract perspective, either of a control channel and a data channel, or both [10]. Although neither of these terms have an academic definition, we may state that control channels carry the information necessary to handle the flows of information from one entity to another: establishing communication flows and managing resources and parameters such as bandwidth, latency and covertness; and that data channels are reliable communication channels that can be used to transfer information from one entity to another [13].

The control operations can normally be completed with short amounts of information. Specific communication establishment procedures may be sufficient to handle data channels for a certain period of time without having to establish a control channel. Theses procedures may involve different parameters such as the type of compression/

encryption to be used during the communication or what to do when specific attributes (e.g.: bandwidth, stealthness) reach a certain level.

The control channel has to be as stealthy as possible. Any unusual activity regarding the standard behaviour (e.g.: sessions, large amounts of data/ packets) should be avoided. Furthermore, the control channel is intended to transfer relatively short amounts of data, so bandwidth is of lesser importance and focus should be placed on stealth and latency parameters.

3.1.2 – Aggregation

As covert channels are based on legitimate communication channels permitted by Network Access Control Systems (NACS) , it is often possible to use more than one of these channels, and even different types of these channels, as simultaneous carriers for the covert channels. Each one of these channels has it's own requirements in terms of quality attributes. For example, different communication channels over HTTP, ICMP and SMTP protocols may be used simultaneously with the objective of increasing the stealthiness of a control channel. Further actions may help achieve this objective, such as changing the carrier protocol frequently or randomly, or after the stealthier protocol has been identified in a particular scenario.

However, the use multiple communication channels might be counter-productive in terms of stealthiness, as it may alert NACS administrators of the existence of ongoing covert communications. Other possibility is the aggregation of multiple data and/or control channels over a single communication channel. This is most useful when the NACS in place has a very restricted number of permitted communication channels. As this approach involves multiplexing of several channels over a single one, bandwidth usage of the latter may increase significantly, which may lead to the detection of the covert communications.

3.1.3 – Communication architectures

The primary goal of covert channel communication architectures is to remain covered long enough to transfer information before the cover time of the information elapses. Standard client/ server and peer-to-peer architectures may be used to achieve this purpose [13].
The client / server architecture differentiates the setup and operating modes of client and servers. Generally, the client(s) send request(s) to the server(s), which process the requests and responds with the results. This architecture is scalable as long as at least one server is available to the client(s).
In peer-to-peer architectures, all nodes involved operate as clients and servers, the main advantage being the independence of the availability of a limited number of servers.

Various communication models may be used to implement covert channels over these architectures. The most simplistic model consists of the direct communication of the client and the server through a point-to-point connection. In this model a server component or process is running on a system on an external network and the client component, located in the internal network establishes a communication channel through the NACS. This model is simple to implement in practice, although it provides restricted possibilities, as the server and client components can only operate in the manner they were designed, which limits their flexibility. In many situations, however, this is enough to achieve the goal of covert communication.

Another valid communication is the “proxy model”, which may allow the use of a covert channel for different types of data. It involves an additional component, a Proxy, which accepts data streams from clients and servers and acts as an intermediary. This model enables the use of distinct applications while using one or multiple types of covert channels. It is one of the most popular models among covert channel practical implementations and tunnelling implementations.

In scenarios where it is desirable for the NACS protected services to open the communication channels, a reverse communication model can be implemented, in which the server components themselves initiate the communication channels from

the internal network and wait for requests. This model applies to the client/server and proxy models.

4 – COVERT CHANNELS ON COMMON NETWORK PROTOCOLS

In the following sections different covert channels that have been identified on several commonly used network protocols are described. Most of them involve the manipulation of protocol features and the embedding of data on header fields in order to transfer information covertly.

4.1 – Internet Protocol (IP)

The Internet Protocol is the network layer protocol that drives the Internet, providing for the transfer of data in the form of packets across the network. It is responsible for routing, the process of selecting a path a network. It is an enabler for wide area networking, which means that its scope is global and can cover disparate network subsystems. This makes it a very popular target for data hiding.

Internet Protocol version 4 (IPv4) header tunnelling was one of the first instances of covert channels on the network layer. Each packet contains a protocol header that consists of 23 fields, used for a variety of purposes such as the carrying of routing information, Quality of Service and fragmentation. However this variety results in an inherent risk of them to be used to transmit data instead of network managing information. Data can be transferred covertly between networks, by compressing data to a form that can be embedded in the header. Although these headers are open to inspection, the embedded value is considered legitimate and is not considered anomalous.

The 16 bit IP Identification field is the most eligible choice. It can be used for byte-to-byte covert communication. The protocol specification states that it is used to identify individual packets in case packet fragmentation occurs in the network. Packets with

the same ID refer to the same original datagram; this way the destination host can re-establish the complete datagram.

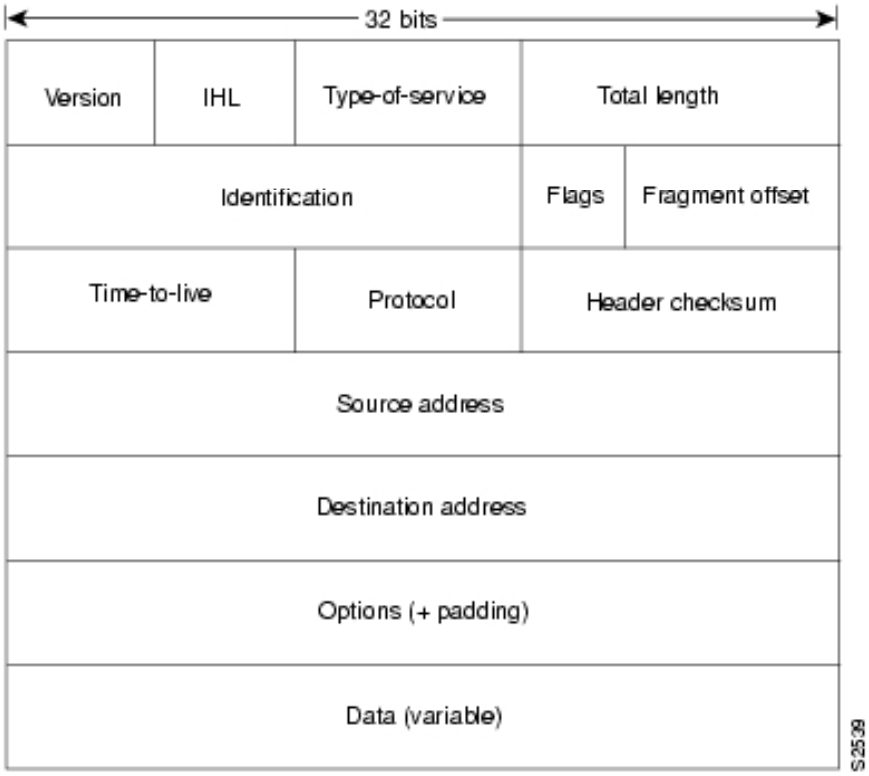

Figure 1 - Internet Protocol Header [33]

The value of this field should be a number chosen randomly; although operating systems differ in the way they generate this number. In general, it is increased by one upon transmission of a packet. It may also contain a non-random value without disrupting the IP mechanism, which means 16 bits of data can concealed in this field by crafting packets that use a predetermined value. A covert channel can be created by encoding data in separate 16 bit values and transmitting them in the IP ID field, then decoding them on the other end.

Another IP header field suitable for the transmission of data is the 3 bits Flags field, specifically the *Don't Fragment* (DF) flag. It may be considered a redundant bit and thus its value doesn't influence the normal IP operations. The downside of using this field for covert communications is its low size. It can only hold one bit per IP packet.

The IP header also contains a 24-bit Options field. Its use is optional and intended to provide for control functions that are useful in some situations, but unnecessary in most common communication scenarios. In some cases it can also be used to transfer data without interrupting the IP protocol service. A novel covert channel is proposed in [14] which allows for the sending of short messages encoded in the Record Route options field. A practical implementation that exploits this channel has been developed, and is based on a common application that makes use of this field, the *traceroute* command [14].

Concealment of data on the 8-bit Padding field is also possible. It is normally used to pad the Options field to align it to a 32-bit block. This field should only contain zeroes, but can be used by malicious users to transfer data as well.

The possibility of establishing a covert channel through the Time To Live (TTL) field has also been studied [6]. In this case, the sender manipulates the TTLs of subsequent packets, transmitting information to the receiver covertly. However, an encoding scheme is also necessary to achieve this, since network elements along the path also modify the TTL, creating a natural variation of its value. The use of any other TTL values apart from the few normal initial TTLs used in common systems would appear highly suspicious. The encoding scheme proposed consists of the use of two symbols: *low*-TTL signals and *high*-TTL signals. This allows for the traffic to look as legitimate traffic, although provides a low efficiency of one bit per packet. This however can be optimized employing some of the communication models presented earlier.

A new enhanced version of the Internet Protocol, known as the *Internet Protocol version 6* (IPv6), is intended to replace the current used version in the coming years. It provides improved reliability, broader address space and better security than its predecessor. However, it is also significantly more complex, which leads to an increased vulnerability to being used for covert communications.

A particularly interesting feature of the IPv6 protocol in the context of covert channels is the Extension Headers, which are legitimately used for a variety of purposes. These headers have 16 bits to specify the Next Header type, 8 bits to

indicate the header length, and a variable length options field. The two high order bits of the options field indicate the action that must be taken in case the option type is not recognized. A covert channel could possibly be established by generating a destination options extension header, and a proof-of-concept messaging application employing this mechanism has been developed [15]. Other interesting field of the IPv6 header that may possibly be used a carrier for a covert channel is the 8-bit Reserved Field. According to [11] it might be possible to hide 8 bits of data in this field. The modulation of destination addresses to communicate data covertly may also become a viable approach, given the large size of the IPv6 address space [11].

4.2 – Transmission Control Protocol (TCP)

The Transmission Control Protocol (TCP) is the transport-layer protocol that provides reliable data transmission on the Internet. The same reasons that make the Internet Protocol an evident carrier target for covert communications apply to TCP.

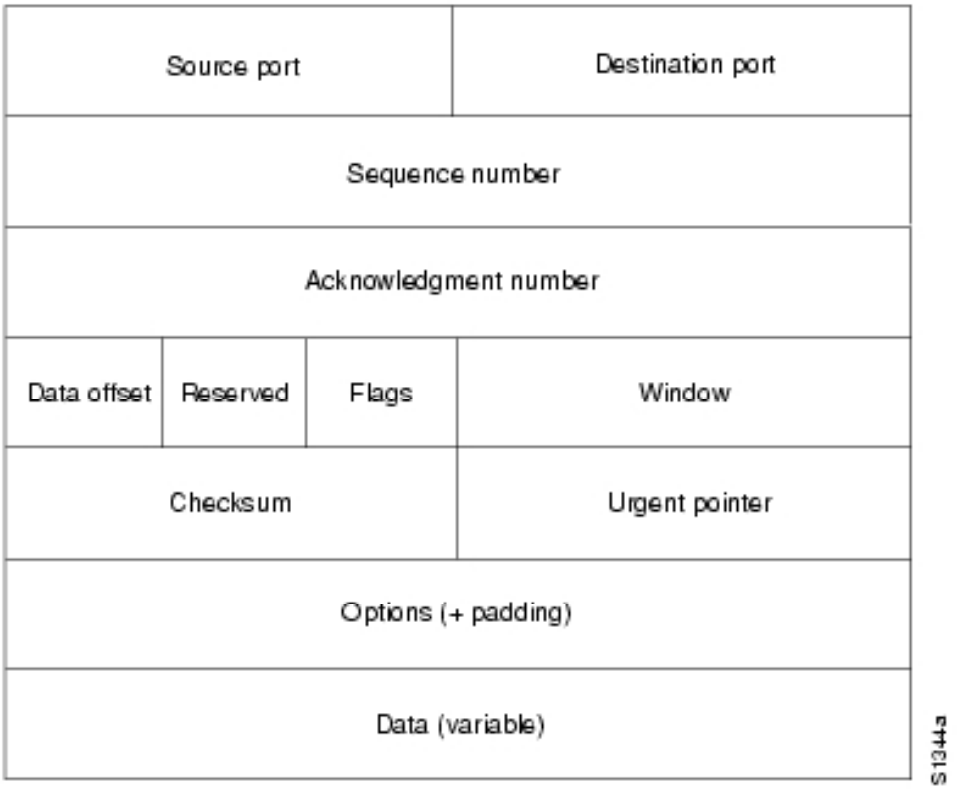

Figure 2 - Transport Control Protocol header [33]

One of the main purposes of the TCP protocol is the provision of packet reordering on arrival at the receiver and the provision of a retransmission service that allows the receiver to request the retransmission of particular segments. This is achieved with the use of sequence numbers, which are stored in the 32-bit TCP *sequence number* field. Initially, an *Initial Sequence Number* (ISN) is generated randomly, which is used in the first segment of a TCP session (SYN segment). The receiving host normally responds acknowledging it received the first segment (SYN / ACK segment), and uses the ISN + 1 as the sequence number (acknowledgment number). However, the use of a non-random value in the sequence number field doesn't disrupt the TCP protocol. This implies that a malicious user can use this field to transmit 32 bits of arbitrary data per segment. Furthermore, because random values are normally expected in this field, covert channels using it as carrier are particularly hard to detect.

TCP also provides a mechanism to acknowledge the proper receipt of packets to the source. This is achieved using the 32-bit acknowledgement number field, which contains the values of the sequence number of the originator incremented by 1. It has been proved that the sender IP address of a TCP segment can be spoofed, causing the receiver to acknowledge to an arbitrary host with the incremented value transmitted through this field. This technique makes use of a *bouncing host*.

4.3 – Internet Control Message Protocol (ICMP)

The Internet Control Message Protocol is a network layer protocol designed to troubleshoot IP-based network communications through informational, error and test messages. ICMP packets are encapsulated in IP datagrams. A detailed description of the operation of the ICMP protocol can be found in [16].

Two of the most common ICMP messages are the *echo-request* and *echo-reply* messages, normally used to diagnose network problems. Normal operation consists of the sending of an *echo-request* message to a particular host, who answers with an *echo-response* message. The protocol also allows the returning of variable length data

to the sender in the Optional Data field. Some IP options such as *router alert*, *record route* and *timestamp* can be used when encapsulating ICMP *echo-request* messages, which provides a channel for covert communications between the communicating entities [17]. Currently, most Network Access Control Systems restrict the use of these messages. However, they are allowed in many situations for use in network diagnosis, and the contents of the messages are usually not filtered.

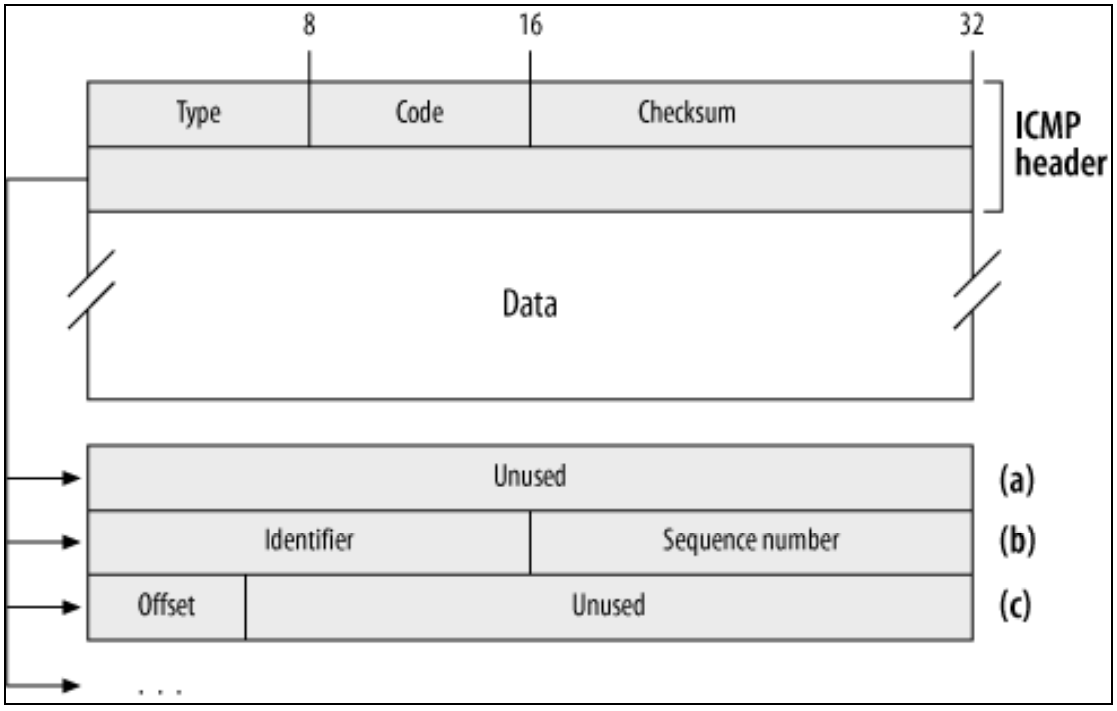

Figure 3 - Internet Control Message Protocol header [34]

Another ICMP message that can be used to establish a covert channel is the *address mask* request [17]. The request normally fills the 32-bit address mask field with zeroes, but can be used to transmit data instead. However the scope of a covert channel based in this technique would be very limited, as these messages are sent from a host to a router on the same Local Area Network (LAN).

Several practical implementations exist that make use of ICMP messages for covert communications. These will be discussed in section 4.

4.4 – Internet Group Management Protocol (IGMP)

IP multicasting allows the transmission of data to a subset of host computers, which can be spread across different physical networks across the Internet. A given subset is known as a Multicast Group. Multicast routers and hosts communicate group membership information via the Internet Group Management Protocol. The protocol is based on messages, specifically *report messages*, sent from a host to router to communicate the joining of a group, membership continuation or the leaving of a group; and *query messages*, sent by to routers to hosts for group monitoring purposes. IGMP packets are encapsulated in IP datagrams for transmission, where the destination address is a multicast address.

The IGMP header contains unused fields which can be used for covert communications through proper embedding / extraction processes at the communicating ends. Report messages provide an 8-bit unused field, which is actually set to zeroes and ignored by the receiver, while query messages provide 16 unused bits, which are also set to zeroes and ignored by the receiver. These can be combined with covert channels techniques in the IP header (described in section 4.1) to increase the bandwidth of the covert channel. [17]

The IGMP protocol, however, is designed for communications between hosts and routers on the same network. The TTL field in the IP header for IGMP messages is normally set to 1, which makes this messages unroutable outside the originator's local network. This limits the scope of covert communications using this protocol to embed secret data.

4.5 – Hyper-Text Transport Protocol (HTTP)

The Hyper-Text Transport Protocol is an application-layer protocol designed to transfer information over networks. Its operation consists of synchronous communication using request-response message pairs, the most common being *GET* and *POST* pairs. The protocol specification [18], however, contains six request types.

Almost all organizations allow the use of the HTTP protocol, as the World Wide Web is the primary information resource. It is widely implemented and used across different types of networks, which makes it an interesting target for carrying covert channels. Lower layer protocols (IP, TCP, ICMP) present numerous limitations such as limited capacity and modification of packets at intermediate nodes. For this reason HTTP has become the most used protocol for covert communications, and several different mechanisms to transfer data covertly using this protocol have been proposed and implemented.

HTTP request messages may contain multiple headers, some common examples being "*User-agent*" and "*Referer*". Malicious users can exploit this by using headers to transmit arbitrary data. A particularly interesting feature of the HTTP protocol is the *Entity-body*. It is normally only used in HTTP POST requests is the, because it has no real use for other types of requests. However, the protocol specification doesn't state explicitly that this should not be present in other request types, which enables the transmission of arbitrary data by a malicious user in any request type.

Just like HTTP request messages, HTTP response messages may contain multiple headers as well. Malicious users may exploit this by using headers to transmit arbitrary data in the same as request headers. When this field is combined with one of mentioned fields in the HTTP request messages a synchronous covert channel can be established.

4.6 – Domain Name System (DNS)

The Domain Name System (DNS) is a transport-layer protocol used for storing and querying information of domain names in a distributed database [11]. It is a well known and widely deployed protocol and provides for the resolution of domain names into their corresponding IP addresses and resolution of a domain's mail server through the use of *Mail Exchange* (MX) records.

Just like the HTTP protocol, DNS is a synchronous protocol based on request and response pairs, communication taking place with the use of messages. These

messages contain a message header, which is present on both request and response messages.

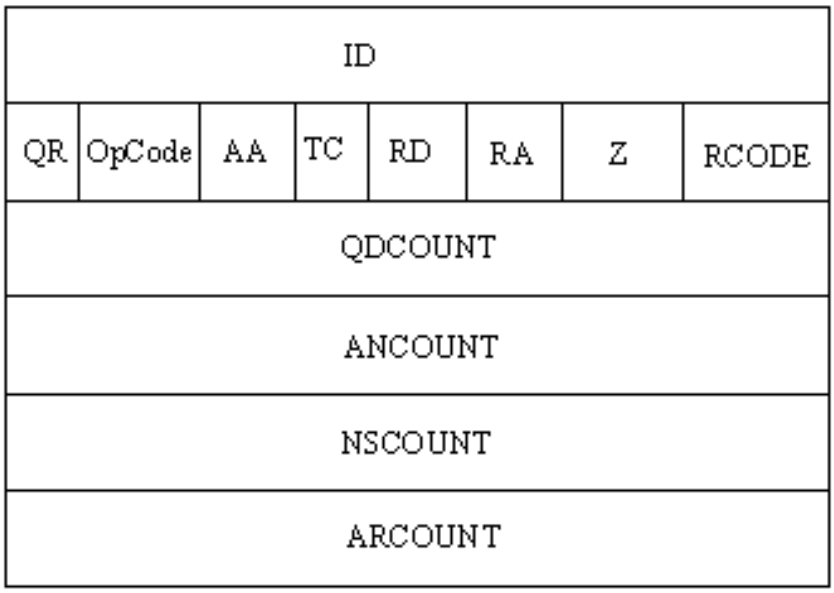

Figure 4 - Domain Name System message header [35]

The 16-bit identification (ID) field is used for keeping track of queries. A malicious user could design an algorithm that uses a shorter space to identify queries, and use the residual space to hide arbitrary data. [11]

Other particularly interesting fields in the DNS header are the *QDCOUNT*, *ANCOUNT*, *NSCOUNT* and *ARCOUNT* fields. Their length is 16 bits, and they serve different purposes for the operation of the DNS protocol. The QDCOUNT field is used to state the number of entries that follow in the question section that comes after this header. The ANCOUNT field is used to state the number of resource records in the answer section that appears after this header. The NSCOUNT field is used to state the number of name server resource records entries in the answer section that follows this header. The ARCOUNT field is used to state the number of entries in the question section that follows this header. [19] If a malicious user has control over a rogue DNS server, he could use these fields to covertly transfer up to 16 bits of data in each one of them. However, to avoid disruption of the protocol, he must add the correct amount of queries and answers in the messages after this header as stated in

the corresponding fields. This could be achieved with the use of a purposely-designed algorithm that calculates the relative number containing both the data being transferred and any legitimate headers that follow.

DNS query messages are followed by a set of headers. The QNAME field is used to store the string entered in the query subject to translation, and according to the DNS specification it should be in the form a *Fully Qualified Domain Name* (FQDN). This filed is limited to the maximum length permitted for a FQDN, which is 255 bytes. A malicious user could use up to the 255 bytes permitted to transfer arbitrary data, although it must format it to carry 63 bytes per label to comply with the FQDN specifications. However, different implementations of the DNS protocol vary and in some cases these limits could be ignored and larger packets could be used [11]

Answers to a DNS query consist of the message and query headers described above, plus a set of answer headers. The NAME field is used to identify the FQDN to which a resource record belongs [19]. Information can be transferred covertly using this field in the same way as described for the QNAME field in the query message. As in the previous case, the contents of the NAME filed must also comply with the FQDN rules.

4.7 – Voice Over IP (VoIP)

Recently, covert channels to transfer information through various VoIP protocol specifications such as control traffic (i.e. SIP, H.323, RTCP) or data transport protocols (i.e. RTP) have been discovered. They present a significant risk; emerging threats such as VoIP SPAM or Botnet may work in tandem to transfer control signals or binary executables through VoIP covert channels. [20]

A proof of concept attack demonstrating this new VoIP threat has been developed (Vo^2IP). It allows for the establishment of a hidden conversation by embedding further compressed voice data into regular PCM-based voice traffic (i.e. G.711 codec). Therefore an eavesdropper who is not aware of the use of the covert channel can’t decode the conversation properly. [20]

In this particular attack, the input audio is compressed using the G.729 codec, and then embedded into a G.711 VoIP channel using the least significant bit (LSB) coding. G.729 is an audio data compression algorithm used to encode voice audio, and the standard codec operates at 8 Kbps. G.711 is a codec primarily used in telephony since the 1970s, and is also of the ITU-T standard and remains a "must support codec" in today's VoIP world. It operates at 64kbps.

By replacing a single bit of each sample in the G.711 audio data, it is possible to inject 8kb of arbitrary information for each second. Vo^{2}IP uses the least significant bit of each sample in order to minimize the changes in audio quality. The transfer of information at a faster rate is feasible by using more bits on each audio sample at the cost of reduced cover audio quality.

4.8 – Ad-Hoc On-Demand Distance Vector Routing Protocol (AODV)

AODV is a routing protocol designed to reduce the amount of control traffic on ad-hoc wireless networks by using on-demand route queries. An originator broadcasts a route request when he needs to send a message, which contains destination identifiers, destination sequence numbers, source identifiers, source sequence numbers, and hop count. He then receives route replies, which contain the request data, a destination identifier and a route lifetime that states for how long the route should be stored in the routing table.

A recent study has identified four potential covert channels in the AODV route discovery protocol [21]. The first is a timing covert channel, which can be realised by timing the requests at a previously agreed time interval. The second identified channel consists of the manipulation of the source sequence number. This number is normally used to aid in the establishment of routes from intermediate nodes back to the source. However, arbitrary data or specific increments of the value can be embedded in this field, thus creating a hidden communication channel. Arbitrary data can also be embedded in the lifetime field, as well as in the destination identifier. A message of

up to log2 (N -1) bits can be embedded in the destination identifier in a network with N nodes [21].

This protocol is susceptible to these threats because the sending of requests is at discretion of the sender (on-demand). The source will be able to transmit hidden information with the condition that source and destination agree a covert communication protocol. A downside of these channels is that they tend to be noisy because of the nature of the mediums used in ad-hoc networks (wireless), and would probably require the use of synchronization mechanisms.

Although this particular research focus is on the AODV protocol, other protocols used in ad-hoc networks might be also vulnerable to these threats.

5 – COVERT COMMUNICATIONS USING X.509 DIGITAL CERTIFICATES

Users of a public key require confidence that the right remote person or system with which they will use an encryption or digital signature mechanism owns the associated private key. This confidence is obtained through the use of digital certificates, also referred to as public-key certificates. They are data structures that bind a public key to a subject, and incorporate a digital signature by a Certificate Authority (CA) to assert this binding. A certificate has a limited valid lifetime, stated in its contents. A wide range of applications may use public-key certificates, covering a broad array of interoperability objectives and an even broader array of operational and assurance requirements [RFC]. Their users are people and processes that use client software, and include electronic mail readers and writers, World Wide Web clients and servers and IPSec processes among many others; they operate in an ample range of environments. Different certifications schemes exist. A certificate might be generated by the user and signed by the CA, or the CA might generate the certificate and sign it on behalf of the user.

ITU-T X.509 or ISO/IEC 9594-8 defines a standard certificate format (X.509). It was first published in 1988, defining the version 1 format, and it was later revised in 1993

to define the version 2 format. Experience gained while attempting to implement and deploy the standard highlighted the deficiency of the version 1 and 2 formats, which lead to the development of the X.509 version 3 certificate format (X.509 v3), standarised in 1996. It constitutes an extension of the version 2 format, achieved by providing additional extension fields.

```
Certificate:
   Data:
       Version: 1 (0x0)
       Serial Number: 7829 (0x1e95)
       Signature Algorithm: md5WithRSAEncryption
       Issuer: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
               OU=Certification Services Division,
               CN=Thawte Server CA/emailAddress=server-certs@thawte.com
       Validity
           Not Before: Jul  9 16:04:02 1998 GMT
           Not After : Jul  9 16:04:02 1999 GMT
       Subject: C=US, ST=Maryland, L=Pasadena, O=Brent Baccala,
                OU=FreeSoft, CN=www.freesoft.org/emailAddress=baccala@freesoft.org
       Subject Public Key Info:
           Public Key Algorithm: rsaEncryption
           RSA Public Key: (1024 bit)
               Modulus (1024 bit):
                   00:b4:31:98:0a:c4:bc:62:c1:88:aa:dc:b0:c8:bb:
                   33:35:19:d5:0c:64:b9:3d:41:b2:96:fc:f3:31:e1:
                   66:36:d0:8e:56:12:44:ba:75:eb:e8:1c:9c:5b:66:
                   70:33:52:14:c9:ec:4f:91:51:70:39:de:53:85:17:
                   16:94:6e:ee:f4:d5:6f:d5:ca:b3:47:5e:1b:0c:7b:
                   c5:cc:2b:6b:c1:90:c3:16:31:0d:bf:7a:c7:47:77:
                   8f:a0:21:c7:4c:d0:16:65:00:c1:0f:d7:b8:80:e3:
                   d2:75:6b:c1:ea:9e:5c:5c:ea:7d:c1:a1:10:bc:b8:
                   e8:35:1c:9e:27:52:7e:41:8f
               Exponent: 65537 (0x10001)
   Signature Algorithm: md5WithRSAEncryption
       93:5f:8f:5f:c5:af:bf:0a:ab:a5:6d:fb:24:5f:b6:59:5d:9d:
       92:2e:4a:1b:8b:ac:7d:99:17:5d:cd:19:f6:ad:ef:63:2f:92:
       ab:2f:4b:cf:0a:13:90:ee:2c:0e:43:03:be:f6:ea:8e:9c:67:
       d0:a2:40:03:f7:ef:6a:15:09:79:a9:46:ed:b7:16:1b:41:72:
       0d:19:aa:ad:dd:9a:df:ab:97:50:65:f5:5e:85:a6:ef:19:d1:
       5a:de:9d:ea:63:cd:cb:cc:6d:5d:01:85:b5:6d:c8:f3:d9:f7:
       8f:0e:fc:ba:1f:34:e9:96:6e:6c:cf:f2:ef:9b:bf:de:b5:22:
       68:9f
```

Figure 5 - Contents of a sample X.509 certificate [37]

The modification of any value in a digital certificate can be detected by computing the digital signature of the certificate with the algorithm specified and comparing it to the

signature, which must be decrypted using the CA's public key first. Thus, the use of certificates to exfiltrate data might seem like an impossible task. However, an attacker trying to exfiltrate data from a private system is not really concerned in verifying the validity of the key contained in the certificate, as he is using it for other purposes, so he might accomplish his objective even without completing the signature verification process. In some extreme cases the data could even be encoded in the signature or the public key.

It is evident that this approach is a very naïve one, as legitimate users of the system will detect that the certificate has been modified when they try to verify the certificate's signature. This means an attacker has to find a way to send modified certificates only to his client(s), and the private system's rightful certificate to legitimate clients. If the attacker is trying to exfiltrate data, it is sound to assume that he has some control over the private system. He could choose to send the secret data embedded in the certificate only in certain cases, based on the source or destination address of the underlying packets or on the occurrence of particular events controlled by him (i.e. specially crafted requests to a server).

A network based detection system, which can be even based on simple traffic analysis, could detect suspicious values on the different certificate fields used by the attacker (i.e.: odd validity dates). This means the attacker has to focus on encoding the data in a way that makes it look similar to that of a genuine certificate. A network based detection system based on more complex techniques can execute the signature verification process and discover that the certificate has been modified, although this would require that the detection system perform this for every certificate request, which can prove costly in terms of computing resources.

Self-signed certificates constitute a particular case. The signature on a self-signed certificate is generated with the private key associated with the certificate's subject public key, which proves that the issuer, who in many situations is the user, possesses both the public and private keys. Their use presents a much more interesting scenario for the malicious user trying to smuggle data. There is a possibility that the compromise of the private system might lead to the compromise of its private key, which is unfeasible in scenarios where trusted third parties (TTP) are used to sign the

certificates, as compromise of the TTP's private key is extremely unlikely to say the least. In cases where the detection systems only verify the signature of the certificate, and do not perform analysis on the certificate itself, the transmission of secret data on the certificate can prove undetectable. The use of self-signed certificates is not uncommon, in particular in Public Key Infrastructures (PKI) based in the *Web of Trust* scheme. One could argue that if the attacker has compromised the system to the extent of compromising the user's private key and the ability to generate a digital certificate with secret data, he could find an easier way to exfiltrate it than having to embed it in a public-key certificate. It is important to remember, however, that the objective pursued by using covert channels is not to hide the data being exfiltrated, but to hide the fact that the transmission is taking place, making it a appear as regular traffic.

The basic syntax of a X.509 v3 certificate, as defined in [22] is as follows. It is specified in Abstract Syntax Notation 1 (ASN.1), and it is important to note that the data to be signed is encoded using the ASN.1 Distinguished Encoding Rules (DER).

```
Certificate  ::=  SEQUENCE  {
        tbsCertificate       TBSCertificate,
        signatureAlgorithm   AlgorithmIdentifier,
        signatureValue       BIT STRING  }

   TBSCertificate  ::=  SEQUENCE  {
        version         [0]  EXPLICIT Version DEFAULT v1,
        serialNumber         CertificateSerialNumber,
        signature            AlgorithmIdentifier,
        issuer               Name,
        validity             Validity,
        subject              Name,
        subjectPublicKeyInfo SubjectPublicKeyInfo,
        issuerUniqueID  [1]  IMPLICIT UniqueIdentifier OPTIONAL,
                             -- If present, version MUST be v2 or v3
        subjectUniqueID [2]  IMPLICIT UniqueIdentifier OPTIONAL,
                             -- If present, version MUST be v2 or v3
        extensions      [3]  EXPLICIT Extensions OPTIONAL
                             -- If present, version MUST be v3
        }
```

```
Version  ::=  INTEGER  {  v1(0), v2(1), v3(2)  }

CertificateSerialNumber  ::=  INTEGER

Validity ::= SEQUENCE {
     notBefore      Time,
     notAfter       Time }

Time ::= CHOICE {
     utcTime        UTCTime,
     generalTime    GeneralizedTime }

UniqueIdentifier  ::=  BIT STRING

SubjectPublicKeyInfo  ::=  SEQUENCE  {
     algorithm            AlgorithmIdentifier,
     subjectPublicKey     BIT STRING  }

Extensions  ::=  SEQUENCE SIZE (1..MAX) OF Extension

Extension  ::=  SEQUENCE  {
     extnID      OBJECT IDENTIFIER,
     critical    BOOLEAN DEFAULT FALSE,
     extnValue   OCTET STRING  }
```

The attributes that can be potentially used to transfer data covertly have been highlighted, and are described in further detail in the next subsections.

5.1 – serialNumber

The serial number of a certificate is a value assigned by the CA used to uniquely identify a certificate. The X.509 specification [22] states that it must be a positive integer and that given its uniqueness requirements, users must be able to handle serialNumber values of up to 20 octects. The encoding of embedded data in this number can be used as a covert channel, although its detection is rather trivial as

legitimate values are not random, but generated in a specific way, depending on the CA. It may remain undetected by very simple detection systems and naïve analysts.

5.2 – Validity

The certificate validity period is the time interval during which the CA guarantees that the certificate information is accurate. The field consists of a sequence of two dates: the date on which the validity period begins (notBefore) and the date which marks the expiration of the certificate (notAfter). Both values may be encoded as UTCTime or GeneralizedTime.

The universal time type, UTCTime, is a standard type designed for the representation of time and dates. It allows the specification of time to the precision of one minute or one second and the year is specified through the two low order digits. In the context of public-key certificates UTCTime values must include seconds even when the number of seconds is zero. As the year is represented in two digits, when the value YY is greater than or equal to 50, it shall be interpreted as 19YY, and when it is less than 50, it shall be interpreted as 20YY.

The generalized time type, GeneralizedTime is standard type designed for the variable representation of time. It must also include seconds, but not fractional seconds, representing time as YYYYMMDDHHMMSSZ.

The specification states that conforming validity dates through the year 2049 must be encoded as UTCTime, and dates in the year 2050 or later must be encoded as GeneralizedTime.

A covert channel can be established by specially encoding data in the difference of these values. According to the specification, the minimum year value allowed for the notBefore date would be 1900, which would be represented by the digits 00 in the two low order digits. The notAfter data could have a theoretical maximum year value of 9999, represented with four digits. Both notBefore and notAfter time values must be specified with a precision of seconds. This means that these values allow for the

following number of differences, which is equivalent to the number of seconds that exist between 00:00:00 of the year 1900 to 23:59:59 of the year 9999:

$$(9999 - 1900) * 365 * 3600 = \mathbf{10.642.086.000}$$

If a variation of one second is used to represent a message or symbol, this means that **10.642.086.000** symbols could be encoded using this differential encoding scheme, roughly equivalent to 2^{34} messages. Thus, up to 34 bits of data can be transferred covertly using this time-differential encoding scheme in a single digital certificate by manipulating its time validity.

A value of the notAfter field earlier than the current date and time would be highly suspicious, as it would mean that the certificate has expired. That would reduce the amount of possible symbols that can be encoded using this scheme. Furthermore, the use of border values allowed for the validity of a certificate might also raise suspicion, as their use in legitimate situations is questionable, as it would generate in practical terms a certificate that never expires, or a certificate which use is allowed even before the invention of modern computer systems. The range should be chosen carefully in order to look innocuous, taking into consideration the current date and the standard validity periods used in different scenarios. Certificates with expiration dates of 3 or more years from the issuing date are not uncommon on the Internet, although those used in mid to high security systems it's more likely to be a few days, hours or even minutes.

It might be possible to adapt optimized differential encoding schemes for use in this scenario. They are likely to provide a more efficient use of the channel than the encoding scheme presented. This is left for future work.

5.3 – Unique Identifiers

The design objective of having subject and issuer unique identifiers is to allow for the possibility to reuse of subject and/or issuer names over time. However it is

recommended that names are not reused for different entities, and the use of these unique identifiers is not recommended in Internet certificates [22].

They are of the type BIT STRING that is an arbitrary length string of bits. This makes them particularly attractive to transfer data. These fields must only appear if the version of the certificate is 2 or 3. They must not appear in version 1 certificates.
A significant limitation in using these fields for covert channels is that the Internet Engineering Task Force (IETF) has deprecated their use. This means that the mere presence of this field might seem suspicious by well-trained analysts and detection systems.

6 – COVERT COMMUNICATIONS USING THE TRANSPORT SECURE LAYER PROTOCOL AND X.509 DIGITAL CERTIFICATES

The Transport Secure Layer (TLS) allows applications to communicate preserving privacy and data integrity in order to prevent eavesdropping, tampering and message forgery, among other threats. For example, it is used on the Internet to secure traffic between HTTP clients and web servers. On a typical scenario, only the server is authenticated. It is the successor to the Secure Socket Layer 3.0 protocol (SSL3). Both protocols are very similar, and share the same vulnerabilities [23].

TLS consists of two protocols, the record protocol and the handshake protocol, that normally run on top of a reliable transport protocol such as TCP. The record protocol provides a private connection service based on the use of symmetric encryption and Message Authentication Codes (MACs), while the handshake protocol provides services of peer authentication and shared secret negotiation. This section focuses on the TLS handshake protocol, specifically on the possibility of using covert channels through the manipulation of the TLS protocol headers to establish the parameters of an exfiltration covert channel using the methods described in the previous section.

Message Type	Parameters
hello_request	Null
client_hello	Version, random, session id, cipher suite, compression
server_hello	Version, random, session id, cipher suite, compression
certificate	Chain of X.509v3 certificates
server_key_exchange	Parameters, signature
certificate_request	Type, authorities
server_done	Null
certificate_verify	Signature
client_key_exchange	Parameters, signature
finished	Hash value

Table 1 - TLS Handshake Protocol Message types

The handshake protocol is the most complex part of TLS. It is used before any application data is transmitted. It consists of a series of messages exchanged by the client and the server. The different message types are presented in Table 1. A very simplified description of the protocol, which does not include client authentication, is presented next.

The first phase of the protocol is used to initiate a logical connection and to establish the security capabilities that will be associated with it. It is initiated by the client, with the sending of a **client_hello** message. This message is responded by the server with a **server_hello** message that contains the same parameters as the **client_hello** message.

The second phase begins with the server sending its certificate; this message contains one or a chain of X.509 certificates. A certificate message is required for any agreed on key exchange method. The final message of this phase is the **server_done** message, to indicate the end of this phase.

Upon the receipt of the **server_done** message, the client initiates phase 3, after verifying that the server provided a valid certificate. He sends a **client_key_exchange** message, which contains the key material to be used for the key generation. The specific contents of this message depend on the key exchange method agreed in phase 1.

The fourth and final phase consists of the client sending a **finished_message**, which verifies that the key exchange and authentication processes were successful. It is responded by a server **finished_message**. At this point the handshake is complete and the client and server may start to exchange application data.

The vulnerability of the TLS handshake protocol to covert channels has been studied [23], although on a slightly different context. Different fields from the messages involved have been identified as potential carriers for covert channels, the most relevant for this study being the *cipher_suite* and *random* fields. According to the authors, the *cipher suite* field sent by the client in the client_hello message can provide a few bits for covert communications, and the *random* field can provide 224 bits for the same purpose.

An interesting scenario would be to take advantage of these covert channels to request the exfiltration of data by a client attacker from a compromised server. As discussed in the previous section, an attacker trying to exfiltrate data in X.509 certificates might need to send modified certificates only to his client(s), while still being able to send the rightful certificate to legitimate clients. This might be accomplished by the sending of instructions through data encoded and embedded in the aforementioned TLS fields of client_hello messages. There must be processes on the server capable of interpreting and executing such instructions.

These covert channels in the TLS protocol could also prove useful to establish different parameters to be used for the exfiltration of data via X.509 certificates. For example, the attacker could instruct a process on the compromised system which data to transmit, and in which specific fields and positions of the X.509 certificate to embed the data. Other useful attributes that can be specified by the attacker can be which type of encoding to use, or instruct the compromised server process to encrypt

the data using a specific algorithm. If the malicious user is employing the validity fields of the X.509 certificate using the encoding technique presented in the previous section, he could use the TLS covert channel to establish the notBefore or notAfter values, chosen according to the scenario.

As the cipher suite and random fields are also present on server_hello messages, they could be used to acknowledge the reception of the malicious user messages, or to communicate back to the attacker any relevant information.

A process on the compromised server would then proceed to generate a X.509 certificate congaing the data to be exfiltrated, and send it to the attacker as part of phase 2 of the TLS handshake protocol, complying to the parameters previously specified.

7 – PREVENTION AND DETECTION OF NETWORK COVERT CHANNELS

The occurrence of covert channels cannot be completely eliminated, although it can be considerably reduced by careful design and analysis. A portion of the bandwidth of a legitimate communication channel that can be sidetracked, to be used as a carrier for covert communications, will always be present.

The detection of this type of threats can be approached in different ways. One consists of the monitoring and detection of traffic that exceeds specific thresholds established previously at the network and/or transport layers. A signature based detection approach is also valid, in which case the traffic is monitored in search for characteristic patterns of specific tools used for the establishment of covert channels. The detection of protocol anomalies generated by some tools is also an indicator of the presence of covert communications. Another approach is to learn the “network behaviour” and the use statistical methods to establish if the current monitored network traffic corresponds to that recorded correct behaviour.

Standard IDS technologies are also commonly used to aid in the detection of covert communications, along with the deployment of Network Security Monitoring (NSM) models. According to [24], NSM involves “to collect, analyze and increase indications and warnings to detect and respond to intrusions". [24]The same study states that "within the context of NSM indicators are outputs from products which are created by IDS and are usually referred to as alerts. Trained people who may be referred to as analysts should be engaged in interpreting intrusions. The interpretation of indicators results in warnings. Warnings are human conclusions which indicate to decision makers that a network may have been compromised."

It is critical for any detection approach to limit the amount false positives, which is raising alerts for the possible presence of covert communications while the monitored traffic is completely legitimate. It is also important to keep the ratio of false negatives to the lowest, which is not raising alerts in the presence of suspicious traffic.

A covert communication channel might be safe if the detection systems in place are signature-based and have no specific rule to look for it explicitly or if the set-up of such rule is too costly, either in terms of system resources, money or false-positives.

Furthermore, it is possible to increase the difficulty of detection of covert channels with several methods. A plausible strategy is to create confusion by using multiple sources and destinations.

Several techniques to increase the number of destinations are described in [25]. One consists of the use of transit nodes that will forward data to the final destination. A second method involves the sending of traffic to legitimate destinations each time the channel is used. A solution to use legitimate third-party systems to store and retrieve information is also described [13].

The number of sources can also be multiplied in order to increase confusion of analysts and avoid detection. Distinct sources can be used alternatively to send data through a NACS. Some sources may send legitimate data as well to puzzle the detection team, or to increase the volume of traffic that needs to be analysed. [13]

There is also a possibility of using nonexistent sources and / or destinations. If it is known that the recipient is on the communication path to another network, messages could be sent to this other network and intercepted in transit by the intended receiver. Even the destinations themselves could encode the secret information to be transferred. [13]

Research on detection targeted specifically on network covert channels is still in its early stages. Most of it discusses the theoretical possibilities of different approaches and techniques. However, after extensive research on identifying covert channels, their detection is a rapidly growing field of research. Different approaches, techniques and counter-measures have been proposed and demonstrated. The most relevant are briefly discussed in the following sections.

7.1 – Simple Traffic Analysis

Several covert "naïve" covert channels can be easily detected by conducting a traffic analysis at packet level. Different protocol stack implementations (e.g. different operating systems) normally exhibit well-defined characteristics in generated header fields. These can be used to establish a trusted baseline and identify anomalies that may be a sign of the use of protocol header manipulation for covert communications. For instance, a detailed description of the TCP/IP ISN and IP ID generation schemes in Linux and OpenBSD can be found in [26]. Similar studies have been conducted for several other protocols. An analyst could detect the manipulation of these protocol's headers because the values generated by attacker's tools can be differentiated easily from those generated by a genuine protocol stack. This same principle can be applied in many of the cases presented in section 4. However, if the attacker knows the method used by the victims' system to generate these values, he could encode his data in an identical way, thus creating an undetectable covert channel [26].

7.2 – Statistical traffic analysis

Many of the fields normally chosen by attackers in protocol headers to carry data covertly are designed to contain random values. Actually most of these fields are not completely random, or are only random in a limited space [27]. When using these fields, covert channels normally try to preserve the randomness to avoid detection, for example by using encryption algorithms before embedding the data. However the space covered by the generated random values is different to the one covered by the ones generated by the legitimate systems' algorithms, and thus can be detected. In some cases too much randomness can be suspicious.

7.3 – Process Query Systems (PQS)

Process query systems are a new retrieval information technology, in which queries are described as process descriptions [PQS]. They allow for a user to identify which process has generated which events from a given set of events. PQS have a broad range of applications, examples being the analysis of social networks and the tracking of fish and humans on video [27]. They can also be used to monitor networks, which is the most relevant application for the purpose of this discussion.
PQS are used in [28] to detect timing channels based on binary codes. The authors warn that some models can return false-positives. The use of these systems for the detection of covert channels is considered excessive by some studies [27]

7.4 – Neural Networks (NN)

Artificial Neural Networks, normally referred to simply as Neural Networks, consist of a network of simple processing elements (neurons), which can exhibit complex global behaviour, determined by the connections between the processing elements and element parameters. The technique was originally inspired and intends to imitate the functioning of the central nervous system and the neurons. In a Neural Network model, simple nodes, which constitute the processing elements, are inter-connected to form a network. They have several different applications, one of the most relevant for the information security field being the recognition of complex patterns and

sequences. They can be trained to recognize traffic characteristics that could be exploited by covert channels, and are very powerful in the sense that they can generalize [27]. NN can be trained to detect many different types of covert channels, but it is difficult to find the optimal number of neurons needed.
Other major drawbacks are that their training can be time consuming, and can trigger false positives. [27]

7.5 – Support Vector Machines (SVM)

Support Vector Machines are somewhat similar to Neural Networks, in the sense that they have learning capabilities that can be used to identify patterns. It is a concept that uses mathematical and optimization techniques. The use of SVM to detect covert channels has been demonstrated in [29]. However, the covert channels used in the tests conducted in this study could also be detected by other simpler detection techniques such simple or statistical traffic analysis and are not really stealth.

7.6 – Active Wardens

An active warden, also referred to as a traffic normalizer, is a network application that intercepts all the traffic on a network and enforces its compliance with the pre-defined security policies. Its operation is similar to that of firewalls, but unlike them, wardens detect, remove or modify any likely carriers of covert channels on all network layers, with the objective of removing ambiguities from network protocols. For example, IP ID fields are set to zero when there is no fragmentation, and when there is, the IP ID of the fragments are changed and randomized [27].

In order for this approach to be viable, wardens must alter and distort the traffic to an extent that it does not affect the quality of the data received by the user, but eliminating potential sources of covert communication. This modification has been conceptualized as Minimal Requisite Fidelity (MRF) [30]. In theory, active wardens could be used in combination with one of the detection techniques described in previous sub-sections.

The concept of active warden has been discussed in the research community for over two decades. However no full practical implementation has been developed. A constraint of the design of active wardens is the context of a high-bandwidth, real-time firewall, as the practical capabilities of an active warden vary extensively from those of a theoretical one. It is expected that the implementation of active wardens will improve the understanding on the extent of the viability of this approach. [30]

7.7 – Quantized pumps

The quantized pump system is and improvement of traditional one-way communication systems such as the store-and-forward protocol, the pump and upwards channel. Its operation is limited to one-way communications. Its objective is to minimize the amount of information that could be transmitted using covert channels. The bandwidth of such channels can be controlled precisely using this method. Its main use is to enforce the Bell-LaPadula security model. This model has been studied extensively in [31]

7.8 – Risk Management Approach

It is possible to compare covert channels to physical smuggling. Risk management approaches to smuggling are often based on the measurement of the probability of an incident taking place. This probability is normally derived from the implemented security measures and the existing threats. Related research has been conducted in the field of Nuclear Material trafficking [32].

Covert channels, however, correspond to threats that that take place in an unexpected form. Furthermore, they are generally part of a broader incident, and their nature makes them difficult to detect unless they lead to further incidents. This makes them very difficult to predict. These issues makes the assessment of covert channels through the use of probabilistic risk management very complex, and impacts on the ability of efficiently planning for the mitigation of such threats.

Security management standards, such as ISO 17799 and ISO 27001 do not treat covert channels expressly, but assume they are managed with broad network segregation and network connection controls. A framework that allows for the holistic identification of network covert channels is yet to be developed.

8 - CONCLUSIONS

This study makes important contributions to the research field of covert communications and data hiding. Little previous work has focused on the study of Digital Certificates as possible carriers for the transmission of secret data. Much of its importance lies in the fact that public-key certificates are considered the cornerstone of Public Key Infrastructures. The implementation and deployment of these infrastructures has been plagued by obstacles, which include the complexity of the task, poor understanding of them and numerous interoperability issues. The fact that they can be used for the purposes discussed in this study provides another matter of concern for their adoption and viability.

In order to understand the implications of this contribution, this work also provides a comprehensive, up to date overview of the current state of covert communications research, including many important techniques and methodologies, some of which will be the focus of discussions in the future. As the research community has directed extensive efforts on identifying covert channels, it is now looking to broaden and diversify the research. In particular, many of the detection and prevention approaches summarized in this study are likely to be the central focus of research in the following years.

It must also be said that the mechanism and methodologies proposed in this study are just a few in an incredibly large universe of possibilities. It can be safely asserted that the possibilities for establishing covert communications through network covert channels are almost endless. This is good reason to recommend that future research should direct its efforts in the development of generic detection and prevention mechanisms to counter the threat posed by these channels.

9 – BIBLIOGRAPHY

[1] Beijtlich, Richard, Powell, G.. The Tao of Network Security Monitoring. Addison Wesley. 2004

[2] Maney, Kevin. “Bin Laden’s Messages Could Be Hiding In Plain Sight.” USA Today December 19, 2001
http://www.usatoday.com/life/cyber/ccarch/2001/12/19/maney.htm

[3] Simmons, Gustavus J. Prisoners’ Problem and the Subliminal Channel (The), CRYPTO83 - Advances in Cryptology, August 22-24. 1984. pp. 51-67.

[4] Pukhraj, Singh. Whispers on the Wire, Network Based Covert Channels, Whitepaper, http://gray-world.net/papers/pukhrajsingh_covert.doc

[5] Petitcolas, Fabien A., Ross J. Anderson and Markus G. Kahn, Information hiding - a Survey. part of IEEE special issue on protection of multimedia content 7/99 http://www.cl.cam.ac.uk/~fapp2/publications/ieee99-infohiding.pdf

[6] Zander, Sebastian. Grenville, Armitage, Philip Branch. Covert Channels in the IP Time To Live field. Centre for Advanced Internet Architectures (CAIA), Swinburne University of Technology, Melbourne, Australia

[7] Annarita Giani, Vincent H. Berk, George V. Cybenko. Data Exfiltration and Covert Channels. Thayer School of Engineering, Dartmouth College, Hanover, USA

[8] Lampson, W. A note on the confinement problem. Communications of the ACM, Volume 16, Issue 10. 1973

[9] U.S. Department of Defense. Trusted Computer System Evaluation “The Orange Book”. Publication DoD 5200.28-STD. Washington: GPO 1985 http://www.radium.ncsc.mil/tpep/library/rainbow/5200.28-STD.html

[10] Wang, Zhenghong: New Constructive Approach to Covert Channel Modeling and Channel Capacity Estimation, 2005, Department of Electrical Engineering, Princeton University, Princeton, NJ, USA. http://palms.ee.princeton.edu/PALMSopen/ISC05_w_cit.pdf

[11] Marc Smeets, Matthijs Koot. Research Report: Covert Channels. University of Amsterdam, MSc in System and Network Engineering, 2006

[12] A guide to understanding Covert Channel Analysis of Trusted Systems , National Computer Security Center, Maryland, USA. 1993. http://www.fas.org/irp/nsa/rainbow/tg030.htm

[13] Gray-World.net Team: Covert channels through the looking glass. 2005. http://gray-world.net/pro jects/papers/cc.txt

[14] Zouheir Trabelsi, Hesham El-Sayed, Lilia Frikha, Tamer Rabie. A novel covert channel based on the IP header record route option. International Journal of Advanced Media and Communication (IJAMC), Vol. 1, No. 4, 2007.

[15] Graf, Thomas. Messaging over IPv6 Destination Options, http://net.suug.ch/articles/2003/07/06/ip6msg.html

[16] J. Postel, INTERNET CONTROL MESSAGE PROTOCOL, DARPA INTERNET PROGRAM PROTOCOL SPECIFICATION, 1981. http://www.faqs.org/rfcs/rfc792.html

[17] Ahsan, Kamran. Covert Channel Analysis and Data Hiding in TCP/IP (master’s thesis). University of Toronto. 2002. http://gray-world.net/papers/ahsan02.pdf

[18] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee. Hypertext Transfer Protocol -- HTTP/1.1, RFC 2616. 1999.
http://www.faqs.org/rfcs/rfc2616.html

[19] P. Mockapetris. Domain Names – Implementation and Specification. RFC 1035, 1987.
http://www.faqs.org/rfcs/rfc1035.html

[20] Vo2IP Project, Georgia Tech Information Security Center, GA, USA
http://www.voipcc.gtisc.gatech.edu/vo2ip.php

[21] Song Li; Epliremides, A. A network layer covert channel in ad-hoc wireless networks. Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on
Volume , Issue , 4-7 Oct. 2004 Page(s): 88 – 96.

[22] R. Houley, W. Ford, W. Polk, D. Solo., Internet X.509 Public Key Infrastructure Certificate and CRL Profile, RFC 2459, 1999.
http://www.ietf.org/rfc/rfc2459.txt

[23] Eu-Jin Goh , Dan Boneh , Benny Pinkas, and Philippe Golle. The Design and Implementation of Protocol-Based Hidden Key Recovery. Stanford University. 2002.

[24] Bejtlich, Richard. Integrating the Network Security Monitoring Model. Sysadmin Magazine. April 2004.

[25] Dyatlov, Alex. Castro, Simon. Exploitation of data streams authorized by a network access control system for arbitrary data transfers: tunneling and covert channels over the HTTP protocol. Whitepaper. 2003.
http://gray-world.net/projects/papers/covert_paper.txt

[26] Murdoch, Steven J., Lewis, Stephen. Embedding Covert Channels into TCP / IP. University of Cambridge, Computer Laboratory. 2005.

[27] Allix, Pierre. Covert Channels Analysis in TCP / IP networks. IFIPS School of Engineering, University of Paris-Sud XI, Orsay, France. 2007

[28] Vincent Berk, Annarita Giani, George Cybenko, Covert Channel Detection Using Process Query Systems, 2005.

[29] Taeshik Sohn, JungTaek Seo, and Jongsub Moon, A Study on the Covert Channel Detection of TCP/IP Header Using Support Vector Machine, 2003.

[30] Gina Fisk, Mike Fisk, Christos Papadopoulos, and Josh Neil. Eliminating Steganography in Internet Traffic with Active Wardens. Los Alamos National Laboratory, University of Southern California.

[31] Ogurtsov, N.; Orman, H.; Schroeppel, R.; Oapos;Malley, S.; Spatscheck, O. Experimental results of covert channel limitation in one-waycommunication systems. Network and Distributed System Security, 1997. Proceedings., 1997 Symposium on Volume , Issue , 10-11 Feb 1997 Page(s):2 – 15.

[32] Scott, B. (2002) Decision-based model development for nuclear material, theft, smuggling and illicit use. Proceedings of international conference on physical protection (NUMAT). Salzburg: University of Salzburg. http://www.numat.at/list%20of%20papers/scott.pdf

[33] Cisco Systems Product Documentation http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/ip.htm#wp2468

[34] Benvenutti, Christian. Understanding Linux Network internals, O' Reilly, 2005. http://safari.oreilly.com/0596002556/understandlni-CHP-25-SECT-1

[35] Unknown. Teach yourself TCP / IP in 14 days, Second Edition. http://www4.dogus.edu.tr/bim/bil_kay/network/tcpip/

[37] Wikipedia. http://en.wikipedia.org/wiki/X.509#Sample_X.509_certificates