Received March 27, 2022, accepted April 13, 2022, date of publication April 18, 2022, date of current version April 22, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3168018

Analysis of Reversible Network Covert Channels

PRZEMYSŁAW SZARY[1], **WOJCIECH MAZURCZYK**[1], **(Senior Member, IEEE), STEFFEN WENDZEL**[2], **AND LUCA CAVIGLIONE**[3]

[1]Warsaw University of Technology, 00-661 Warsaw, Poland
[2]Worms University of Applied Science, 67549 Worms, Germany
[3]National Research Council of Italy, 16149 Genova, Italy

Corresponding author: Wojciech Mazurczyk (wojciech.mazurczyk@pw.edu.pl)

This work was supported in part by the European Union's Horizon 2020 Research and Innovation Program through the SIMARGL Project under Grant 833042.

ABSTRACT In the last years, the utilization of information hiding techniques for empowering modern strains of malware has become a serious concern for security experts. Such an approach allows attackers to act in a stealthy manner, for instance, to covertly exfiltrate confidential data or retrieve additional command & control payloads for the operation of malware. Therefore, the deep understanding of data hiding mechanisms is a core requirement, as it allows designing effective countermeasures. Unfortunately, the most recent evolution of information-hiding-capable threats enjoys *reversible* properties, i.e., the abused network flow is restored to its original form. Hence, detection approaches based on the comparison of different traffic samples may not work anymore. In this paper, we further investigate various methods for performing reversible data hiding for network covert channels. Specifically, we extend our previous research by considering different scenarios focusing on IPv4 traffic and HTTP conversations. The results confirm that reversibility can be used in various network conditions and is not impaired by middleboxes. In addition, engineering countermeasures or mitigation techniques could be difficult, thus requiring to consider reversible mechanisms already in the early design stages of a protocol/deployment.

INDEX TERMS Covert channels, information hiding, network security, network steganography, reversible data hiding.

I. INTRODUCTION

Modern malware is often endowed with advanced capabilities to make its detection harder and to improve its ability of infecting hosts and devices [1]. To this aim, prime approaches exploit fileless implementations, multi-stage loading architectures, encryption and anti-forensics countermeasures, or code obfuscation mechanisms. An emerging trend concerns the use of information hiding techniques for several malicious purposes [2]. For instance, information hiding can be used to elude security schemes enforced via sandboxes, to cloak payloads in digital media and infect computing equipment of users, or to create covert channels, i.e., parasitic communications paths between processes [3]. To implement such a cloaked communication scheme, the malware injects information in a suitable carrier, which is then used to encode or move the secret data. Among the various carriers that can be used to conceal information, network traffic has quickly become the most preferred one [4]. As a consequence, the attacker can establish a path from the infected host towards a controlled machine (e.g., a Command & Control server) by creating a network covert channel nested within legitimate overt traffic. In general, this requires that the covert endpoints parasitize a licit network conversation: in principle, the overt sender and the overt receiver may be not aware that their data exchange is utilized for other purposes [5]. Due to their ephemeral nature, network covert channels may provide a significant advantage over the defenders, especially making forensics investigations and threat analysis attempts harder [3], [5].

Therefore, understanding how to detect a network covert channel is of utmost importance. Despite the literature already proposes various countermeasures (see, e.g., [5]–[7] and the references therein), attackers improve their hiding schemes on a continuous basis, especially to bypass blockages and inspection tools as well as to postpone the detection as long as possible [1], [3]. As a consequence, the most sophisticated network covert channels are difficult to detect, especially without paying considerable costs in terms of traffic processing, disruption of the Quality of Experience (QoE), and impact on the privacy of the users [1], [3], [8]. To boost the undetectability of cloaked data transfers, a major

The associate editor coordinating the review of this manuscript and approving it for publication was Xujie Li.

approach exploits some form of *reversibility*, which is a technique for reverting the carrier manipulated by the attacker to its original form as soon as the hidden communication is completed. Hence, a defensive system residing in the network may not be able to spot malicious communication attempts, mainly due to the lack of visible signatures or the impossibility of finding discrepancies among traffic samples collected in different portions of the network [6], [7]. For the case of network traffic, reversibility is achieved through injection/extraction mechanisms that are able to restore the flows and datagrams to their initial condition [9], [10]. Such an approach, borrowed from the digital media domain, has proven to be effective, thus researchers require its detailed understanding to fully assess the security of modern network scenarios [10].

In this perspective, this work investigates Reversible Data Hiding (RDH) schemes for network covert channels. This paper is an extended version of our preliminary research published in [10] and has the following enhancements:

- the proof-of-concept implementation and the testbed have been improved to handle the presence of an HTTP proxy. Thus, this work also investigates the impact of middleboxes on the end-to-end semantics of the covert channel as well as on the proposed RDH schemes;
- alterations in the QoE of the overt HTTP conversations implementing the covert channels have been explicitly considered, e.g., page loading times;
- an in-depth analysis of HTTP communications after/before the injection process has been done, especially to understand whether delays can reveal the presence of a secret or void the RDH approach;
- reversibility has been further investigated to update the data hiding taxonomy progressively used and accepted by the research community dealing with network covert channels and information hiding [11];
- the discussion on the design and engineering of countermeasures have been improved and extended.

The rest of the paper is structured as follows. Section II reviews previous works dealing with network covert channels, with emphasis on reversible hiding mechanisms, as well as possible countermeasures. Section III introduces reversible network covert channels and the design of approaches targeting both IPv4 and HTTP traffic, while Section IV portrays the testbed and the used methodology. Section V showcases the numerical results, and Section VI deals with the design of countermeasures. Finally, Section VII concludes the paper and outlines possible future developments.

II. RELATED WORK

In this section, we initially summarize the current trends in the design of network covert channels. Afterwards, we focus on describing existing countermeasures. Lastly, we present the most recent works considering RDH-like mechanisms applied to the case of remote cloaked data transfers.

Lastly, the heterogeneity of modern network and computing scenarios, jointly with the availability of an almost unlimited amount of hardware and software entities that can be used to conceal the secret flow of information render the detection, elimination, and limitation of network covert channels challenging goals, which are also scarcely generalizable.

A. NETWORK COVERT CHANNELS

Network covert channels represent a double-edged tool, since they can be used both for licit and malicious purposes. For instance, a covert channel allows preserving privacy and guarantee anonymity in investigative journalism, as well as to hide the communicating endpoints to bypass censorship [22]. They can be also used to share intelligence reports or conceal mission-critical communications, e.g., to prevent industrial espionage attempts [23]. Alas, the major utilization of hidden network communications regards empowering malware and advanced persistent threats with the ability of exchanging commands, move stolen data, and drop a payload beyond secure perimeters enforced with firewalls, intrusion detection systems and personal security tools [2], [3].

Concerning the evolution of network covert channels, early techniques already arose back in the 80s [12], [14] and 90s [15], [16]. With the ubiquitous diffusion of always-connected devices and appliances, they became prime offense mechanisms. In general, two main groups of network covert channels can be distinguished [7], depending on the steganographic method used to embed the secret information:

- a *storage network covert channel* is created if the embedding methods directly inject secrets within the carrier. For network traffic, this could be the case of an attacker cloaking information inside a field in the header of a packet or in the data used for padding purposes;
- a *timing network covert channel* is created when the manipulation focuses on the timing of events or the temporal evolution of the carrier [5]. For network traffic, this could be an attacker cloaking information by altering the throughput of a flow or by modulating the inter-packet time of a burst of datagrams.

We point out that a more fine-grained classification scheme has been proposed, especially to uniform discrepancies between different research works dealing with network steganography and covert channels, even not limited to the network-wide case [11], [24]. Unfortunately, information hiding techniques are proliferating in almost every new protocol or network/computing framework [17], [18], IoT environment [19], [25], smart vehicle [26], or industrial control system [27], [28], just to mention the most popular or emerging scenarios. Concerning recent examples, [20] analyzes many opportunities that an attacker can exploit for creating a covert channel within the Network Time Protocol, whereas [13] investigates whether timing channels can be used to implement long-range communications across different continents. Since covert channels are often used to implement exfiltration or for sending commands to an infected host (e.g., to operate a backdoor), they mainly rely upon a half-duplex model. Yet, [21] evaluates the use of

TABLE 1. Assessment of the reversibility potential for some popular and emerging techniques used to implement network covert channels.

Technique	Reversibility	Discussion/Issue	Refs.
ICMP, TCP, and HTTP	Medium/High	Reversibility is highly protocol-dependent. High potential for HTTP	[7]
Packet/Frame Losses	Very Low	Such a macroscopic, network-wide behavior cannot be reversed	[7]
Timing	Very Low	Traffic shaping cannot be arbitrary enforced in end nodes	[12], [13]
ACK and TCP initial sequence number	Low	End-to-end congestion control may prevent restoring the flow	[14]
Unused/Reserved TCP bits	High	Could be voided: unused bits are often "sanitized" by default	[15], [16]
Tags and Data Chunks	Medium	Difficult to achieve without impacting on protocol semantic	[17]
Various VoIP feature	Low/Medium	Complex to implement without degrading the perceived QoE	[18]
MQTT	Medium/High	Implementing reversibility in resource-constrained nodes is very challenging	[19]
NTP	Low	The various NTP "strata" make implementing reversibility hard	[20]
LTE	Very Low	Radio protocols are fragile, with a local scope, and difficult to manipulate	[21]

VoIP over LTE connectivity for the creation of bidirectional channels. Table 1 provides a compact assessment of issues and candidate features that should be considered to endow with some form of reversibility the most popular and emerging techniques used to implement network covert channels.

B. COUNTERMEASURES

The heterogeneity of modern network and computing scenarios, jointly with the availability of an almost unlimited amount of hardware and software entities that can be used to conceal the secret flow of information, make the detection, elimination, and limitation of network covert channels very challenging goals, which are also scarcely generalizable. In more detail, counteracting abusive communication attempts happening through a network covert channel is usually done via a third-party observer called a *warden* [5], [29]. The latter is an intermediate node in charge of capturing the overt traffic and performing inspections to reveal the presence of secret data. Usually, this is done by finding deviations from the "clean" behavior expected for the given flow or traffic aggregate. The literature abounds of metrics, but the most popular indicators that can be used to spot the presence of a parasitic conversation are: *i*) alterations of the throughput, timing statistics, anomalous volumes of retransmissions or malformed packets, as well as an uncommon variability in the sizes of protocol data units, *ii*) anomalous statistical distributions of values in specific fields contained in the protocol header, for instance in bits composing the padding or in unused fields [5], [30], [31].

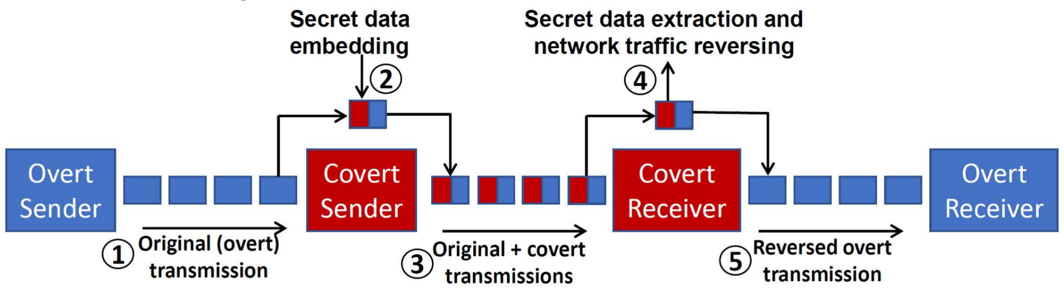
Additionally, a warden can manipulate the traffic by disrupting the covert channel. This process, often called sanitization, can be done via a variety of traffic processing and engineering techniques. As an example, buffering a network flow could remove the information encoded in its time statistics (e.g., the amount of delay between two adjacent packets). Similarly, overwriting some fields in the header can destroy the steganographic mechanisms at the basis of the storage channel [5]. Unfortunately, such operations may introduce additional delays and losses to the traffic, thus it is important to develop optimizations to not penalize flows in an indiscriminate manner as well as to pursue scalability properties [5], [32].

Luckily, the majority of methods used to inject information within a traffic flow cause persistent alterations [24], [33]. This trait is often at the basis of mechanisms and algorithms for revealing the presence of cloaked data transfers [24], [30]. Specifically, permanent alterations of traffic flows can be exploited by deploying a *distributed warden*, i.e., a security tool able to gather data in different points of the network and perform comparisons of the obtained samples. Distributed wardens are considered promising, since they demonstrated to be effective even in the presence of uncommon techniques used in advanced persistent threats [6], [34]. Lastly, a possible approach to recover the lack of signatures in the traffic could take advantage of the computational overhead caused by the additional processing needed to cloak/restore information within the flow (see, e.g., [35] for the case of channels heavily relying upon hashing mechanisms).

C. REVERSIBLE DATA HIDING

As hinted, RDH or some form of reversibility has been already investigated for digital media, especially for images. Hence, the literature abounds with ideas for enforcing reversible features for such a class of carriers. In more detail, the works in [36] and [37] are two recent examples of steganographic techniques not causing permanent alterations to the images used for cloaking purposes. Always referring to digital media, a thorough overview can be found in [38]. Specifically, RDH-capable algorithms can be engineered by exploiting a wide range of approaches, such as lossless compression, difference expansion, histogram shifting, prediction-error expansion, integer-to-integer transformation and pixel value ordering.

On the contrary, to the best of our knowledge, only the work in [9] explicitly addressed RDH and reversibility applied to network covert channels. The only notable exception is the work in [39], which discusses a method to further compress the speech samples of an IP telephony service to free space for embedding data. To prevent the



Secret data embedding
Secret data extraction and network traffic reversing
2
4
Overt Sender
Covert Sender
Covert Receiver
Overt Receiver
1 Original (overt) transmission
3 Original + covert transmissions
5 Reversed overt transmission

FIGURE 1. **The concept of reversible network covert channels.**

disruption of the vocal flow and void the stealthiness of the channel due to a decay in the audio quality, the transcoded voice samples are restored to the previous format. We point out that the RDH property is achieved as a "corollary" rather than as a design choice. Obtaining reversibility for hidden communication attempts in an intrinsic manner is outside the scope of this paper and has been already partially investigated in [9].

III. REVERSIBLE NETWORK COVERT CHANNELS

As discussed, creating a network covert channel requires embedding secret data within a feature of the targeted overt traffic. This may be achieved in different manners, e.g., by encoding secrets using a suitable modulation of delays between packets, or by overwriting a certain field within the header (without disrupting the expected functionality). Despite the used approach, the manipulation should not be "visible" to a third-party observer or being perceived as anomalous. At the same time, the presence of a cloaked flow of data should not interfere with the overt conversation or lead to observable signatures. According to [9], [10], RDH approaches can have various degrees of reversibility. For example, a field in the header could be completely restored to its initial value, thus achieving full reversibility, or only "guessed" based on the historical data. In this case, the process is said to be quasi-reversible.

Figure 1 illustrates how RDH approaches can be exploited in networks when considering a Man-in-the-Middle (MitM) covert communication scenario. Specifically:

(1) the secret transmission relies on the overt traffic generated by the overt sender;
(2) when the overt traffic reaches the covert sender, the flow is modified to inject the secret data: the detailed process depends on the utilized information hiding technique. We recall that this work considers mechanisms targeting IPv4 traffic and HTTP conversations;
(3) the modified network traffic is exchanged between the covert sender and the covert receiver, thus enabling the covert communication;
(4) when traffic is acquired by the covert receiver, the secret data is extracted and the characteristics of network traffic are restored completely (in case of full reversibility) or partially (in case of quasi-reversibility);
(5) the "reversed" traffic reaches the overt sender. Owing to the reversibility mechanism, discovering that the traffic has been used for covert communication purposes turns out to be more challenging, especially if compared to classical approaches not further adjusting the exploited packets/traffic features.

In the following, we provide a detailed description of the reversible network covert channels considered in this work. Section III-A discusses channels targeting IPv4, whereas Section III-B addresses those manipulating HTTP.

A. REVERSIBLE COVERT CHANNELS IN IPV4

To showcase the feasibility of creating reversible covert channels within IPv4 traffic, in this work we considered a hiding approach targeting the Type of Service (`ToS`) field within the IPv4 header, which is 8 bit long. It should be noted that in RFC 3260 the function of this field has been refreshed, and the `ToS` is now divided into two distinct fragments. The first is the Differentiated Service Code Point (`DSCP`), which is 6 bit long, and it is utilized to manage differentiated services. The second is the Explicit Congestion Notification (`ECN`), which is 2 bit long and conveys information related to end-to-end congestion events affecting a route. In the remaining of this paper, except when doubts can arise, with Differentiated Service (`DS`), we describe the concatenation of the two fields (i.e., `DSCP` and `ECN`).

As discussed, for creating an effective covert communication, the embedding of secret data should not result in an easily observable anomaly. Moreover, the chosen carrier needs to ensure enough capacity to contain the secret. To understand if such properties hold, we examined, as a preliminary step, how the `DS` field is utilized in legitimate traffic observed in realistic conditions. To this aim, we inspected traffic captures made available by the *Center for Applied Internet Data Analysis*[1] (CAIDA). Specifically, we investigated the popularity of traffic managed via differentiated services and the adoption of congestion notification mechanisms in real communication networks. We point out that, an in-depth study on the utilization of both `DSCP` and `ECN` to create stealthier injection mechanisms (not limited to the RDH case) is currently part of our ongoing research activity.

[1]https://www.caida.org/

To evaluate the behavior of realistic traffic, we used the CAIDA Anonymized Internet Traces 2019 dataset,[2] leading to an analysis of about 17 million IP datagrams. The study revealed that, in the majority of packets, the `DS` field is limited to two values, i.e., 0×0 (77.26% of the datagrams) and 0×2 (18.59% of all cases). There are also several remaining values but with rare occurrences, for example, 0×38 in 0.48%, 0×1 in 0.33%, or `2E` in only 0.05% of the packets, respectively. Based on the above results, embedding secret information into the `DS` field of IPv4 header can still be acknowledged as a valid covert communication solution, but at the price of limiting the amount of secret data sent per time unit to not make it visible due to too diverse values for the `DS`. Concerning the utilization of the `DS` field to create a reversible covert channel in controlled setups (e.g., in a LAN), this has several important advantages, since it mainly requires modifying the unused parts of the header. However, the limited availability of values for this specific field may make the detection easier, thus the number of steganographically modified packets must be carefully adjusted, or a convenient encoding scheme should be utilized. Nevertheless, the usage of `DSCP` and `ECN` fields evolved over time (as proposed in RFC 2475 and RFC 3168, just to mention the main reference documents) and this may impede the successful deployment of the `DS`-based channel for Internet-scale schemes. In more detail, the security policies enforced at ingress routers may discard packets with `DSCP` values not considered correct or compliant with QoS/QoE schemes. In result, the covert communication may be impaired or identified due to anomalous values causing losses (due to dropping of packets in routers) or unexpected performance (due to wrong QoS guarantees specified by the manipulated `DSCP` fields). In case of invalid `ECN` values, a similar behavior is expected. Therefore, in the rest of this work, we do not focus on such impairments as our main aim is to study RDH capabilities. Yet, this supports the importance of restoring the original state of the carrier (i.e., the fields) before the network traffic reaches "critical" routers to avoid the abrupt interruption of the covert communication. Besides, the attacker should carefully plan the span of the channel as well as implement some "multi-hop" strategy to prevent disruption due to local quality policies.

1) RDH ENCODING SCHEMES

To implement reversible covert channels within the `DS` fields of IPv4 traffic, we considered the following embedding schemes:

- *Implicit*: the secret data is completely embedded into the `DS` field and the covert receiver restores the traffic by using historical inspection of the overt flow. To ensure enough data for the covert receiver to correctly "reverse" packets, the covert sender transmits a number of steganographically-unmodified packets. Owing to this, the covert receiver can create a list of typical `DS` values characterizing the traffic. In our prototype, the required information is inferred from 100 unmodified packets, i.e., no additional data is required to be exchanged between the two covert endpoints.
- *Explicit - Variant 1*: in this case, the "explicit" nature of the scheme is given by the need of extra information for restoring the field to its original state. To achieve this, the covert information is injected into the 4 least significant bits of the `DS` field. The remaining bits are then allocated to notify the covert receiver about the original value of the `DS` field (denoted as the *state*). Hence, the following transmission discipline has been implemented: *[state, secret], [state, secret], [state, secret], . . .* . Note that a fragment of the carrier is utilized to store the state, making the available steganographic bandwidth smaller. On the other hand, this is partially rewarded by the increased undetectability due to the restoration of the `DS` field to its original value.
- *Explicit - Variant 2*: like in the previous case, a fraction of the bandwidth of the covert channel is allocated to transmit control information to the secret receiver. Such an explicit exchange of information is achieved by using an interleaving scheme and without sharing the `DS` field. Then, a burst of unmodified IPv4 packets is sent, followed by another trail with secret data embedded. Note that in our implementation, both bursts were 3 packet long. In the result, the considered transmission sequence is: *[state], [state], [state], [secret], [secret], [secret]*. Such a pattern is then continuously repeated until all secret data is sent. The length of the burst is an important design parameter for the performance of the covert channel: thus, it needs to be carefully evaluated to reach a suitable trade-off between the available steganographic bandwidth and the undetectability of the exchange.

Both the implicit and explicit variants consider the manipulation of a value within the header of the IP protocol. In other words, they perform the so-called *state/value modulation*, which is addressed by the pattern *EN4. State/Value Modulation* of the taxonomy presented in [11]. As it will be discussed later, precisely understanding the exploited features is mandatory to elaborate defensive/detection strategies or to engineer more secure protocols.

B. REVERSIBLE COVERT CHANNELS BASED ON HTTP

Concerning the use of HTTP conversations to implement covert channels, such a process is more complex than the IP counterpart, since even minimal alterations of the header may cause macroscopic functional errors or visible artifacts. Thus, to create a prototypical reversible covert channel exploiting the HTTP traffic, we decided to use the `Accept-Language` header parameter.

Alas, if we directly embed information in the `Accept-Language` header, the presence of the channel will become evident. For instance, if we overwrite the `Accept-Language` value with the arbitrary content

[2]https://catalog.caida.org/details/dataset/passive_merged_pcap

01110 representing the secret, the information exchanged via HTTP could not be correctly displayed since the language will not be anymore the desired one, e.g., `en-US`. Therefore, also in this case, an approach similar to the one used for the `DS`-based covert channel presented in Section III-A should be pursued. The "natural" behavior of the targeted HTTP traffic should be known, mainly to not make the modified conversation be perceived as anomalous. Unfortunately, this is not a simple task, since HTTP partially depends on the behaviors of users or demographic features, such as the language or the geographical location of communicating parties. Thus, an appropriate network traffic study needs to be conducted before such covert channels can be deployed in real-world scenarios. Note that such analysis is a part of our ongoing research. At the same time, the adoption of a reversible approach would allow again to partially compensate the lack of a precise understanding of the exploited traffic.

1) RDH ENCODING SCHEMES

As previously mentioned, to create the reversible covert channel relying on the HTTP protocol, we modulate the `Accept-Language` header. The covert data is embedded using four values for the `Accept-Language` header, i.e., `de`, `fr`, `pl`, and `en`. Each value is mapped into a specific secret data binary sequence as follows:

- `en-US`,**de**;q = 0.5 = "01"
- `en-US`,**fr**;q = 0.5 = "10"
- `en-US`,**pl**;q = 0.5 = "00"
- `en-US`,**en**;q = 0.5 = "11"

When HTTP messages are received by the covert receiver, each `Accept-Language` header is analyzed and its value is extracted and decoded into secret data bits. To endow the covert channel with RDH capabilities, we propose the following embedding approaches:

- *Implicit*: the covert sender directly embeds the secret messages into the `Accept-Language` header. To obtain reversibility, the covert receiver reconstructs the overt traffic only according to historical observations. As in the case of the `DS`-based covert channel, we utilized a set of 100 clean HTTP messages to restore the carrier to its original form with a suitable degree of confidence. Note that we can obtain a higher accuracy for this scheme via continuous network traffic monitoring, but this requires to sacrifice some fraction of the available steganographic bandwidth.
- *Explicit - Variant 1*: similarly to the case of the `DS`-based covert channel, the information needed to restore the state of the various `Accept-Language` headers is sent by using suitable transmission patterns. In this case, the sequence *[state, secret], [state, secret], [state, secret], …* is considered. This further explains why the `Accept-Language` field has been partitioned in two, tightly-related parts, e.g., **en**-US,**en**;q=0.5. As a paradigmatic example, if the `Accept-Language` is `fr-CH`, **X**;q=0.9, we can apply this information to signal the original value of **X** containing the secret and restore it to `fr`.
- *Explicit - Variant 2*: the secret data is injected with a scheme analogous to the aforementioned one, but the information required by the covert receiver to reinstate the `Accept-Language` field is transmitted in bursts: *[state], [state], [state], [secret], [secret], [secret]*. During our experiments, we chose the length of such a pattern as 3 headers per type.

It is worth mentioning that the applied encoding scheme is a critical factor to consider, since it impacts on the available steganographic bandwidth as well as on the detectability of the covert channel. Therefore, to make our discussion more complete, we also propose an encoding variant, which utilizes the relative quality factor `q` and indicates the language preferred by the user in a $0-1$ range. The proposed mapping is presented below:

- `en-US`,en;**q=0.1** = "01"
- `en-US`,en;**q=0.2** = "10"
- `en-US`,en;**q=0.3** = "11"
- `en-US`,en;**q=0.4** = "00"

All methods used to embed information within the HTTP traffic can be brought back to a pattern-based behavior [24]. Specifically, the template is *EN4. State/Value Modulation*, as certain values (e.g., q=0.**x**, and **en**-US;**en**) are manipulated for data hiding purposes. It must be noted that the proposed approach does neither alter the position of the HTTP features nor how they are enumerated with the aim of encoding arbitrary information. As a consequence, alternative embedding patterns of [11] are not applied here. However, this would change if, for instance, the order of elements would be combined with a value modulation. One could then let the `Accept-Language` parameter appear before or after some other header parameter to indicate some binary state, while the `Accept-Language` parameter itself could carry the secret message. This witnesses the complexity of knowing in advance where the information hiding attempt happens, as well as the difficulties of designing protocols without ambiguities that can be exploited for carrying arbitrary data.

IV. TESTBED AND METHODOLOGY

In this section, we introduce the used testbed, the considered performance metrics, and the applied experimental methodology.

A. EXPERIMENTAL TESTBED

To evaluate the network covert channels endowed with RDH capabilities discussed in Section III-A and Section III-B, we prepared the experimental setup depicted in Figure 2. To implement the communicating endpoints and the needed network functionalities, we used the Graphical Network Simulator 3.[3] In this way, it was possible to perform trials in an isolated environment while reducing the impact of uncontrollable uncertainties caused by packet losses

[3]Graphical Network Simulator 3, version 2.1.21, available online: https://www.gns3.com.

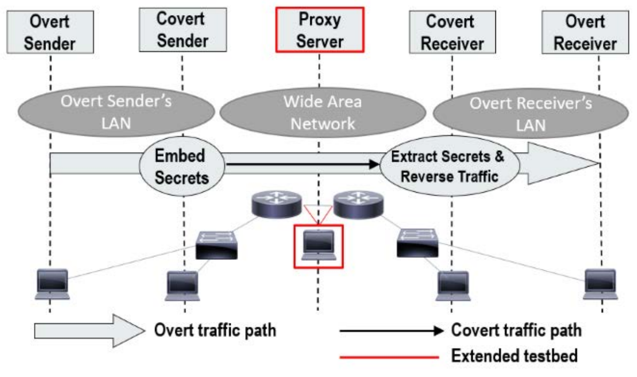


FIGURE 2. Experimental testbed for evaluating RDH-capable covert channels.

and delays of the Internet. As depicted in the figure, we considered a MitM attack template. Specifically:

- two overt communicating nodes, denoted as the Overt Sender (OS) and Overt Receiver (OR), respectively, operate on different local networks connected through a wide-area infrastructure, i.e., the Internet;
- two peers wanting to secretly exchange data via a network covert channel, denoted as the Covert Sender (CS) and Covert Receiver (CR), respectively, capture the overt traffic for embedding secrets, extracting information, and performing the restoration of the various carriers.

The covert endpoints have been implemented in Python by using *Scapy* (version 2.4.3) and *NetfilterQueue* libraries (version 0.8.1) and ran on Kali Linux. The overt endpoints ran on Ubuntu Linux and generate the needed traffic flows (i.e., HTTP conversations). To implement the MitM injection scheme, we used the *arpspoof* and *nfqueue* tools. The produced traffic has been captured and analyzed with *tshark* and ad-hoc scripts. To understand if the presence of an entity breaking the end-to-end semantics will play a major role when HTTP traffic is used to create RDH-based covert channels, we also performed trials with a proxy. In this case, we utilized *Squid* (version 3.5.27[4]) running on Ubuntu 18.04 LTS.

Concerning the overt traffic flows, we utilized IPv4 traffic traces produced by exchanging files between the two considered overt endpoints (i.e., OS and OR). Instead, for the case of HTTP traffic, we considered conversations generated when the OR retrieves content from a blog created with WordPress acting as the OS. To have a realistic condition, the overt HTTP conversations needed to create the `Accept-Language` covert channel have been produced via an ad-hoc script mimicking a user browsing the WordPress blog prepared for tests. To gather metrics about the perceived QoE of users, we used an instrumented version of Google Chrome and the obtained data has been processed offline with Python scripts. For all trials, to create the hidden information to be exchanged, we used a text file containing the book entitled "Forefathers' Eve" by Adam Mickiewicz, accounting for a secret communication of 20 KBytes.

To quantify the robustness of various covert channels, including the RDH-based injection mechanisms, we considered the impact of several network conditions. To this aim, we added delays to the overt traffic containing secret data with a per-packet granularity. The used values were equal to 0.01s and 0.1s for emulating low-latency wireless loops as well as impaired links, e.g., scenarios with a high *bandwidth·delay* product [40]. In the following, such test conditions are denoted as `No_Delay`, `Delay_0.01s`, and `Delay_0.1s`, respectively.

Lastly, to obtain a sufficient statistical relevance, each experiment has been repeated 10 times and, in the rest of the work, average results are presented.

B. PERFORMANCE INDEXES

To evaluate the proposed covert channels, we designed a comprehensive set of performance indexes. First, to quantify the behavior of the RDH-based methodology targeting the `DS` field, we introduced the following metrics, which have been partially borrowed from [10]:

- *Reversibility Index*: defines the ratio between correctly-restored packets and the overall number of packets used to transmit secret data;
- *Covert Transmission Length*: expresses the duration of the secret transmission between the two covert endpoints;
- *Covert Transmission Reliability*: defines whether the received secret information can be decoded or if it contains errors. It has been computed as the ratio between the amount of incorrect information and all received secret data.

Unfortunately, the proposed indicators may fail to capture the real impact of the reversible data hiding mechanism on the quality experienced by users. In the case of the `Accept-Language` methodology, the presence of two covert endpoints manipulating the traffic could lead to alterations of the QoE perceived by the overt endpoints. Thus, we introduced more suitable indicators, i.e., those made available in the cURL tool and discussed in [41]. In more detail:

- *Page Loading Time*: it is the time in seconds between the page request and when the content begins to be displayed in the browser of the user, even if not completed;
- *Header Size*: it is the total number of bytes sent to the OR to transfer the various headers;
- *Connection Time*: it is the time defined in seconds needed to establish a connection between the OS and the OR. In this work, it can represent both the time needed to contact the remote server (i.e., the OS) or the proxy;
- *Total Time*: it is the time expressed in seconds (including the time needed to establish the connection) needed for the complete retrieval of the page, i.e., the main object containing the HTML hypertext as well as companion inline objects.

[4]http://www.squid-cache.org

The two sets of metrics could partially overlap, since manipulating network traffic (e.g., IP datagrams or TCP segments) could impact on the behavior of the application layer. We underline that, due to the carrier-specific nature of the various hiding mechanisms, there is not a one-fits-all metric [30].

C. EXPERIMENTAL METHODOLOGY

To fully evaluate the proposed reversible covert channels, we performed several runs of tests. In the following, with `DS` or `HTTP` we denote trials containing results for the direct injection of data in the IPv4 header (see, Section III-A) or in `Accept-Language` messages (see, Section III-B). Concerning the RDH encoding schemes, with `IMPLICT` we denote the use of the implicit reversing policy, whereas with `EXPLICIT_V1` and `EXPLICIT_V2` we identify the different variants used to restore the traffic.

To evaluate the impact of the network architecture and its complexity, we considered two different scenarios. In the first case, we investigated the "basic" condition of [10], i.e., no proxy has been considered. Collected results will be presented in Section V-A. Instead, the second scenario studied the presence of a proxy (see Figure 2). Since its presence solely accounts for additional delays, which have a negligible impact both on the IP traffic and the covert channel nested within the `DS` field, Section V-B will present results limited to the case of channels targeting the `Accept-Language` header. Lastly, to have a reference baseline, we also repeated all trials by only considering plain, legitimate overt transmissions, i.e., without CS/CR embedding/extracting hidden information. In the following, such reference cases will be denoted as `NOSTEG_BASIC` and `NOSTEG_PROXY`, respectively.

V. NUMERICAL RESULTS

In this section, we present the numerical results. First, we discuss the performance of reversible channels in a "basic" network scenario (Section V-A), then we showcase the impact of a middlebox (Section V-B).

A. BASIC NETWORK SCENARIO

At first, we address the performance of covert channels in terms of the indicators presented in Section IV-B. Specifically, Figure 3 showcases the behavior of the Reversibility Index for `DS`-based covert channels. To avoid burdening the figure, we omitted the `Accept-Language` case, for which we were always able to reverse the HTTP headers to their original form, i.e., its Reversibility Index was always equal to 100%. Instead, for the case of `DS`-based channels, this metric is heavily influenced by delays. In more detail, when they account for TCP retransmissions (e.g., due to timeouts in the reception of acknowledgments) the transport layer tends to synchronize the burst of packets, thus impairing the NetfilterQueue module. As a consequence, the overall flow experiences additional packet losses, which impact on the Reversibility Index.

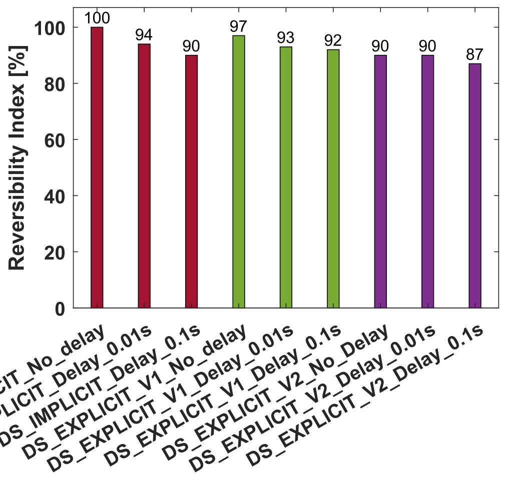


FIGURE 3. Reversibility index for covert channels targeting the `DS` field.

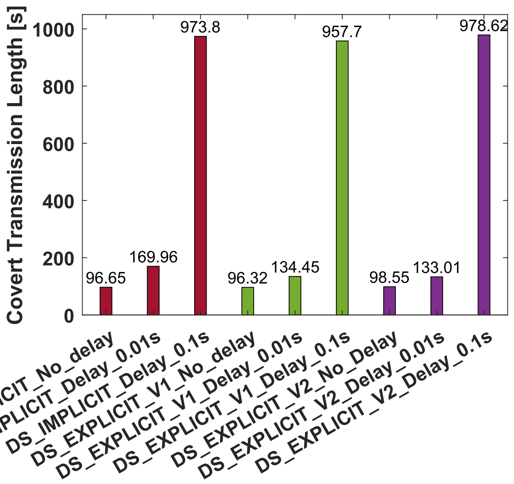


FIGURE 4. Covert transmission length for covert channels targeting the `DS` field.

As hinted, the Covert Transmission Length also plays a major role, i.e., the longer the time needed to exfiltrate information, the higher the probability of being able to spot the parasitic utilization of the targeted traffic flow. Figure 4 depicts the Covert Transmission Length for the case of `DS`-based covert channels. For the sake of clarity, the results for the case of `Accept-Language` have been omitted again. As shown, this parameter is heavily influenced by the presence of network delays. More specifically, the greater delay, the slower the exchange of the secrets. When a delay of 0.1s is present, transferring the secret information required a frame in the range of $950 - 970 \;\; s$, depending on the used RDH encoding scheme. This should be taken into account, as a longer secret transmission would require to manipulate several packets of the overt traffic, thus leading to potential fingerprints in the network traffic. Thus, from the attacker's point of view, a possible approach concerns trading some steganographic bandwidth for undetectability: when operating in networks with high delays, the volume of secret data should be kept as small as possible or sent via

TABLE 2. Overall results obtained for the considered metrics for `DS`-based covert channels.

Variant	Reversibility Index [%]		Covert Trans. Length [s]		Covert Trans. Reliability [%]	
	Val.	Std. dev.	Val.	Std. dev.	Val.	Std. dev.
DS_IMPLICIT_No_delay	100	0	96.65	1.73	100	0
DS_IMPLICIT_Delay_0.01s	94	0.448	169.96	2.08	100	0
DS_IMPLICIT_Delay_0.1s	90	0.727	973.80	3.01	100	0
DS_EXPLICIT_V1_No_delay	97	0.554	96.32	1.69	58	0.74
DS_EXPLICIT_V1_Delay_0.01s	93	0.505	134.45	2.86	100	0
DS_EXPLICIT_V1_Delay_0.1s	92	0.930	957.70	4.67	100	0
DS_EXPLICIT_V2_No_Delay	90	1.097	98.55	2.12	40	1.16
DS_EXPLICIT_V2_Delay_0.01s	90	1.626	133.01	2.26	100	0
DS_EXPLICIT_V2_Delay_0.1s	87	0.940	978.62	4.71	100	0

TABLE 3. Overall results obtained for the considered metrics for `HTTP`-based covert channels.

Variant	Reversibility Index [%]		Covert Trans. Length [s]		Covert Trans. Reliability [%]	
	Val.	Std. dev.	Val.	Std. dev.	Val.	Std. dev.
HTTP_IMPLICIT_No_Delay	100	0	49.00	0.025	100	0
HTTP_IMPLICIT_Delay_0.01s	100	0	49.06	0.034	100	0
HTTP_IMPLICIT_Delay_0.1s	100	0	49.20	0.064	100	0
HTTP_EXPLICIT_V1_No_delay	100	0	50.11	0.065	100	0
HTTP_EXPLICIT_V1_Delay_0.01s	100	0	50.30	0.046	100	0
HTTP_EXPLICIT_V1_Delay_0.1s	100	0	50.70	0.056	100	0
HTTP_EXPLICIT_V2_No_Delay	100	0	49.12	0.034	100	0
HTTP_EXPLICIT_V2_Delay_0.01s	100	0	49.19	0.025	100	0
HTTP_EXPLICIT_V2_Delay_0.1s	100	0	49.21	0.043	100	0

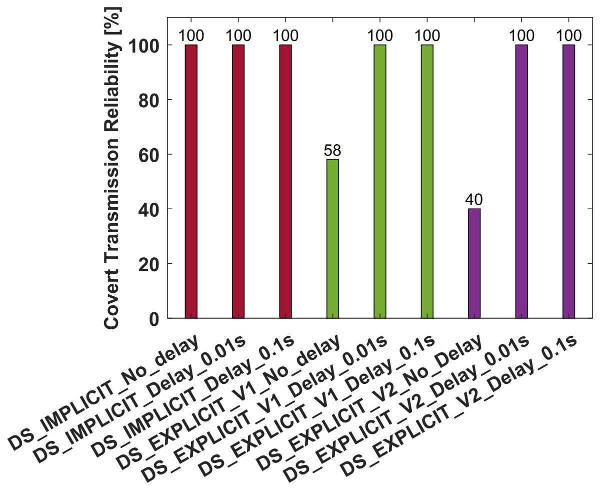


FIGURE 5. Covert transmission reliability for covert channels targeting the `DS` field.

multiple channels to partially recover the need of abusing long-lasting conversations.

Lastly, Figure 5 showcases results for the reliability of the various channels. For the same reason of the Reversible Index, values for the methods using the `Accept-Language` header have been omitted, i.e., the Covert Transmission Reliability observed throughout all trials has been always equal to 100%. A similar behavior has been experienced also for `DS`-based channels only when using the `IMPLICIT` variant for restoring the traffic to its prime form, despite the delays. Surprisingly, when in the presence of "optimal" network conditions (i.e., without delays), both explicit encoding strategies performed poorly (i.e., `EXPLICIT_V1` and `EXPLICIT_V2` achieved a Covert Transmission Reliability of 58% and 40%, respectively). As a consequence, we carried out further investigations. It turned out that the lack of delay "overwhelms" the NetfilterQueue module, i.e., the volume of incoming packets causes losses within the CR. On the contrary, impairments reduce the throughput of the overt traffic, thus the incoming packets offered to the NetfilterQueue have more time to be processed.

To summarize, Table 2 reports a comprehensive overview of the collected results, including standard deviations. As shown, the method should be considered as effective in terms of performance of the adopted indicators. Yet, the malicious entity wanting to exploit `DS`-based channels should carefully evaluate the presence of intermediate nodes enforcing quality of service guarantees as well as exploit incorrect implementations resetting the value of the `DSCP` field (see, e.g., [42] for a comprehensive discussion of some pathological behaviors of routers when handling the `DS` field of IPv4).

Instead, Table 3 presents the values obtained for the case of reversible channels hiding information in the `Accept-Language` field of HTTP conversations. As hinted before, both the Reversibility Index and the Covert Transmission Length always reach 100% (i.e., standard deviation equals zero). This must be ascribed to the presence of HTTP/TCP protocols, which can mitigate the impairments caused by the network or RDH injection mechanisms. Analogously, the Covert Transmission Length remains

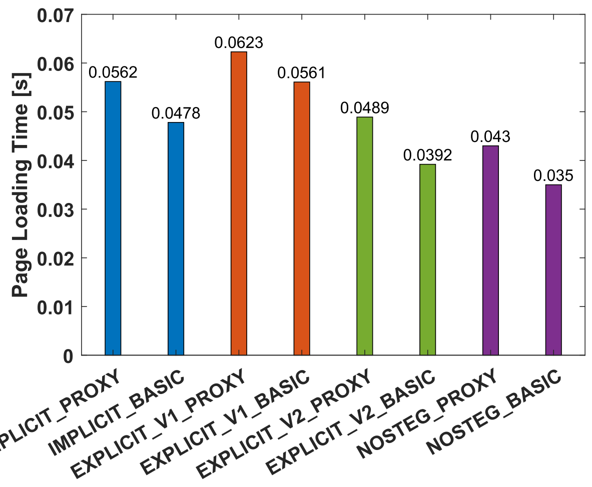


FIGURE 6. Page loading time for covert channels targeting HTTP traffic.

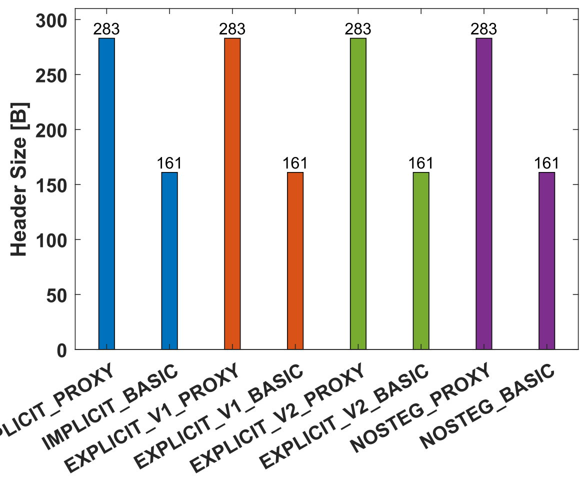


FIGURE 7. Header size for covert channels targeting HTTP traffic.

almost constant, i.e., the multiple-connection flavor of HTTP allows to "fill" the allotted bandwidth pipe provided by the transport layer. Lastly, we point out that the EXPLICIT_V2 scheme used to implement reversibility is slightly faster than the EXPLICIT_V1 one. This is due to the architecture of the algorithm and, in our trials, turns out mainly due to the use of the Scapy library, which causes major overheads.

B. HTTP PROXY SCENARIO

Initially, we wanted to determine the impact of the proxy on the Reversibility Index. It turned out that the obtained results are the same as in the previous case, i.e., the CR can reverse the altered traffic in the 100% of trials. Similarly, the Covert Transmission Reliability was always equal to 100%, thus making the hidden communication path highly robust even when traversing a middlebox. In other words, the proxy server has no noticeable impact on such metrics.

Recalling that the Covert Transmission Length may not capture the impact of RDH-capable channels nested within HTTP conversations, in the following we use the second set of metrics presented in Section IV-B. We point out that, for the sake of comparison, the results obtained for the Page Loading Time, Header Size, Connection Time, and Total Time even without the presence of the proxy have been reported here rather than in Section V-A.

In more detail, Figure 6 compares the Page Loading Time for all the RDH encoding schemes considered. As expected, the proxy accounts for a time overhead due to the "additional" hop in the path between the communicating endpoints, i.e., the OS and the OR. According to our trials, this is mainly caused by the increased number of three-way handshakes of the TCP, which is partially compensated by the pipelining architecture of HTTP. However, it is worth noting the impact of the method to restore the carrier, i.e., the various Accept-Language headers. The EXPLICIT_V1 variant, when jointly used with the proxy, exhibits the longest Page Loading Time, thus causing a decay on the QoE experienced by the user. Instead, the EXPLICIT_V2 variant is faster in

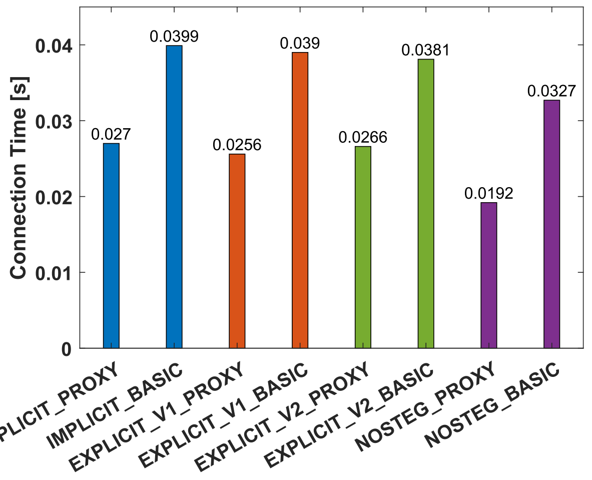


FIGURE 8. Connection time for covert channels targeting HTTP traffic.

the scenario with the proxy server, again due to bottlenecks of the Scapy library. Only the EXPLICIT_V2 RDH encoding scheme exhibits values similar to those experienced in a "clean" web browsing, which has been reported in the figure with the NOSTEG label. According to results, the proxy server accounts for an additional delay in the Page Loading Time of $\sim$8 ms, despite the used encoding scheme.

Next, Figure 7 showcases how the amount of data exchanged among the various communicating peers is influenced by the presence of the covert channel. As expected, the presence of a proxy inflates data volumes related to the Header Size, since additional messages have to be exchanged. However, despite the used RDH-encoding, no noticeable differences can be observed, mainly since the additional information hidden within Accept-Language (if any) is "masked" by the overheads of the HTTP. Hence, detection techniques leveraging discrepancies in the volume of the HTTP traffic or the overheads caused by headers should not be considered effective for the engineering of countermeasures.

Then, Figure 8 provides insights on the Connection Time observed during our trials. We point out that, from the

TABLE 4. Overall results obtained for the considered metrics for HTTP-based covert channels in the BASIC and PROXY scenarios.

Variant & Scenario	Page Loading Time [s]		Header Size [B]		Connection Time [s]		Total Time [s]	
	Val.	Std. dev.	Val.	Std. dev.	Val.	Std. dev.	Val.	Std. dev.
IMPLICIT_PROXY	0.0562	0.0035	283	0	0.0270	0.0065	0.0956	0.0166
IMPLICIT_BASIC	0.0478	0.0063	161	0	0.0399	0.0094	0.1066	0.0168
EXPLICIT_V1_PROXY	0.0623	0.0035	283	0	0.0256	0.0044	0.1009	0.0316
EXPLICIT_V1_BASIC	0.0561	0.0072	161	0	0.0390	0.0063	0.1125	0.0275
EXPLICIT_V2_PROXY	0.0489	0.0040	283	0	0.0266	0.0058	0.0926	0.0144
EXPLICIT_V2_BASIC	0.0392	0.0055	161	0	0.0381	0.0083	0.0968	0.0130
NOSTEG_PROXY	0.0430	0.0058	283	0	0.0192	0.0061	0.0795	0.0107
NOSTEG_BASIC	0.0350	0.0063	161	0	0.0327	0.0065	0.0908	0.0113

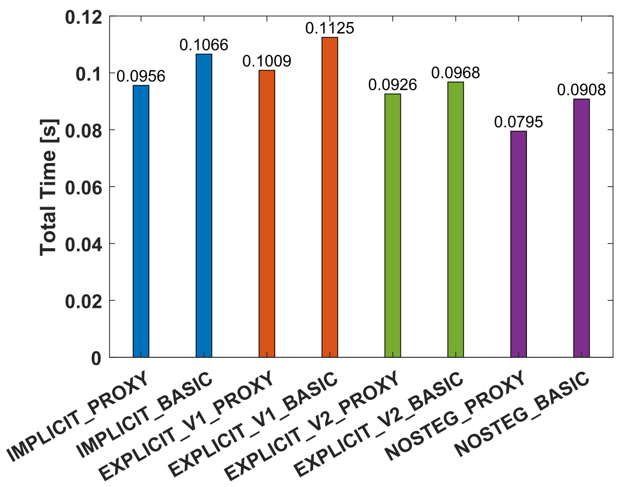


FIGURE 9. Total time for covert channels targeting HTTP traffic.

perspective of the browser, the experienced Connection Time is the one affecting the first "hop". Hence, when the proxy is present, the delays are almost halved. In our trials, it turned out that the EXPLICIT_V2 scheme is the fastest, and it achieves results similar to the case of "clean" HTTP conversations (denoted in the figure as NOSTEG). In some extent, the proxy contributes to mask the presence of the channel. Specifically, the RDH-based approach turns out to be stealthier (in terms of additional delays experienced by the end user) if the overt conversations are inspected between the OR and the proxy.

Furthermore, Figure 9 depicts results for the Total Time. Recalling that this is the time needed to complete the retrieval of the web content, the collected values indicate that the EXPLICIT_V1 RDH mechanism is the one causing higher delays. This is mainly due to the sharing of the Accept-Language header both for hiding secrets and "signal" the information needed to reverse data. Hence, the reduced bandwidth of the covert channel reflects in needing more messages to complete the cloaked transfer between the OR and OS, leading to inflated times.

Lastly, Table 4 provides an overview of the four considered performance indexes. As shown, the Header Size changes its value only when a middlebox is present. This trait should be considered as normal, mainly due to the additional information added by the proxy to plain HTTP headers. Moreover, the implemented covert channels do not alter the size of the various headers, as confirmed by the collected results.

VI. COUNTERMEASURES

In general, developing suitable countermeasures against network covert channels is a challenging goal, especially due to the tight coupling between the injection mechanism and the targeted protocol. As a consequence, several stages concurring to the mitigation of hidden communication attempts require to address poorly generalizable tasks or to inspect multiple traffic features at once [4]. Indeed, the availability of RDH-capable mechanisms is expected to exacerbate the challenge and contribute to the creation of stealthier offensive mechanisms or further empower advanced persistent threats [1]. In this vein, a primary approach should aim at engineering protocols/frameworks with a reduced amount of ambiguity or arbitrary behaviors that can be exploited for embedding secret data. This requires a precise modeling of the various protocols as well as of the selected steganographic approaches. In this sense, the development of a taxonomy or a common "knowledge base" of recurrent techniques for manipulating network traffic is a key step towards a more information-hiding-resistant design [4], [11].

Concerning the creation of mitigation techniques, the use of active wardens or traffic engineering to disrupt the covert channels should be considered a first path to explore. In fact, reversibility may be hard to achieve when in the presence of sophisticated embedding schemes, thus the majority of RDH-capable channels are expected to be more fragile and simpler. As a consequence, deploying ubiquitous techniques like traffic shaping, buffering, or random packet dropping, should be considered an effective first line of defense. Moreover, the need of deploying "signaling" strategies to exchange information for restoring the traffic, or the sharing of fields for delivering both the hidden data and the control information, can be treated as an important weakness of RDH-based channels. In this case, countermeasures aiming at adding noise to traffic (e.g., via discarding/altering packets using some random distributions) or overwriting header fields appear as valid. Unfortunately, these remedies suffer from several flaws. For instance, they may cause the reduction of the quality perceived by users [5], and they are not general enough. Thus, each network covert channel technique or a small subgroup of methods could require a dedicated approach, leading to a high number of defensive mechanisms to work in parallel.

Even if reversibility should be considered a core weaponization tool to prevent spotting the channel via distributed network probes (i.e., to search for discrepancies in the overt flow and recognize the presence of intermediate manipulations), enhancing the ability of sampling the network in more portions remain a prime defensive mechanism. To this aim, solutions using code layering or augmentation of the kernel to gain visibility over different behaviors of network traffic are effective tools to support the ''arm race'' between attackers and defenders [43].

As regards the design of countermeasures for specific channels considered in this work, discrepancies in the DS field collected with a per-packet granularity could be hard to compute. Similarly, alterations of the perceived QoE can surely aid the detection, but could be also challenging to obtain for a single user/application. Instead, probes can be placed in border routers for investigating requests in terms of QoS guarantees and concur in the definition of high-level markers for recognizing RDH-capable embedding techniques. For the case of channels targeting HTTP, the impacts on the user experience are easier to evaluate, especially using suitable proxies. In this case, deviations from the average value of the page loading time or the total time needed to retrieve a page can be used to drive further inspections.

Lastly, the various glitches caused by Scapy (e.g., additional losses or delays) as well as the burdening of the NetFilterQueue module should be further investigated in the vein of developing countermeasures based on signatures or general indicators. Even if it is highly unlikely to have real-world attacks implemented with such development tools (in general, malware is optimized or engineered with low-level techniques [1]), the need of implementing reversibility could challenge the malicious network routines, thus leading to some traffic signatures useful for detection purposes.

VII. CONCLUSION AND FUTURE WORK

In this paper, we presented the design and analysis of network covert channels enjoying reversibility properties. Specifically, we considered two different hiding mechanisms (one targeting IPv4 traffic and one HTTP conversations), each one using three approaches to implement reversibility. To assess the behavior of RDH when applied to network covert channels, we performed a thorough set of trials by considering various setups. Results indicate the feasibility of endowing with reversibility features both types of covert channels. Moreover, it turned out that the presence of an HTTP proxy has a negligible influence on the reversibility of the covert channel. Concerning the design of countermeasures, the results have shown that reversible HTTP channels can be spotted by considering discrepancies in the page loading time.

Future work aims at conducting measurements in production quality settings to refine our implementation. For instance, we are working towards acquiring the data needed to engineer and design suitable schemes for preventing that the overt traffic flow will be discarded by routers due to inconsistent differentiated service requirements or congestion flags. In addition, investigating the impact of other types of middleboxes on the performance of the channels is part of our ongoing research. Lastly, the creation of suitable countermeasures is another important topic that will be addressed in the near future.

REFERENCES

[1] L. Caviglione, M. Choras, I. Corona, A. Janicki, W. Mazurczyk, M. Pawlicki, and K. Wasielewska, ''Tight arms race: Overview of current malware threats and trends in their detection,'' *IEEE Access*, vol. 9, pp. 5371–5396, 2021.

[2] K. Cabaj, L. Caviglione, W. Mazurczyk, S. Wendzel, A. Woodward, and S. Zander, ''The new threats of information hiding: The road ahead,'' *IT Prof.*, vol. 20, no. 3, pp. 31–39, May 2018.

[3] W. Mazurczyk and L. Caviglione, ''Information hiding as a challenge for malware detection,'' *IEEE Secur. Privacy*, vol. 13, no. 2, pp. 89–93, Mar. 2015.

[4] L. Caviglione, ''Trends and challenges in network covert channels countermeasures,'' *Appl. Sci.*, vol. 11, no. 4, p. 1641, Feb. 2021.

[5] W. Mazurczyk, S. Wendzel, S. Zander, A. Houmansadr, and K. Szczypiorski, *Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures*, vol. 7. Hoboken, NJ, USA: Wiley, 2016.

[6] W. Mazurczyk, S. Wendzel, M. Chourib, and J. Keller, ''Countering adaptive network covert communication with dynamic wardens,'' *Future Gener. Comput. Syst.*, vol. 94, pp. 712–725, May 2019.

[7] S. Zander, G. Armitage, and P. Branch, ''A survey of covert channels and countermeasures in computer network Protocols,'' *IEEE Commun. Surveys Tuts.*, vol. 9, no. 3, pp. 44–57, 3rd Quart., 2007.

[8] W. Mazurczyk and S. Wendzel, ''Information hiding: Challenges for forensic experts,'' *Commun. ACM*, vol. 61, no. 1, pp. 86–94, 2017.

[9] W. Mazurczyk, P. Szary, S. Wendzel, and L. Caviglione, ''Towards reversible storage network covert channels,'' in *Proc. 14th Int. Conf. Availability, Rel. Secur.*, Aug. 2019, pp. 1–8.

[10] P. Szary, W. Mazurczyk, S. Wendzel, and L. Caviglione, ''Design and performance evaluation of reversible network covert channels,'' in *Proc. 15th Int. Conf. Availability, Rel. Secur.*, Aug. 2020, pp. 1–8.

[11] S. Wendzel, L. Caviglione, W. Mazurczyk, A. Mileva, J. Dittmann, C. Krätzer, K. Lamshöft, C. Vielhauer, L. Hartmann, J. Keller, and T. Neubert, ''A revised taxonomy of steganography embedding patterns,'' in *Proc. 16th Int. Conf. Availability, Rel. Secur.*, Aug. 2021, pp. 1–12.

[12] C. G. Girling, ''Covert channels in LAN's,'' *IEEE Trans. Softw. Eng.*, vol. SE-13, no. 2, pp. 292–296, Feb. 1987.

[13] J. O. Seo, S. Manoharan, and U. Speidel, ''Feasibility evaluation of long-distance network timing-based covert channels,'' in *Proc. Int. Conf. Electr., Commun., Comput. Eng. (ICECCE)*, Jun. 2021, pp. 1–5.

[14] M. Wolf, ''Covert channels in LAN protocols,'' in *Proc. LANSEC*, 1989, pp. 89–101.

[15] T. G. Handel and M. T. Sandford, ''Hiding data in the OSI network model,'' in *Proc. 1st Int. Workshop Inf. Hiding*. Berlin, Germany: Springer-Verlag, 1996, pp. 23–38.

[16] C. H. Rowland, ''Covert channels in the TCP/IP protocol suite,'' *First Monday*, vol. 2, no. 5, May 1997.

[17] W. Frączek, W. Mazurczyk, and K. Szczypiorski, ''Hiding information in a stream control transmission protocol,'' *Comput. Commun.*, vol. 35, no. 2, pp. 159–169, Jan. 2012.

[18] W. Mazurczyk, ''VoIP steganography and its detection—A survey,'' *ACM Comput. Surv.*, vol. 46, no. 2, pp. 1–21, Nov. 2013.

[19] A. Mileva, A. Velinov, L. Hartmann, S. Wendzel, and W. Mazurczyk, ''Comprehensive analysis of MQTT 5.0 susceptibility to network covert channels,'' *Comput. Secur.*, vol. 104, May 2021, Art. no. 102207.

[20] J. Hielscher, K. Lamshöft, C. Krätzer, and J. Dittmann, ''A systematic analysis of covert channels in the network time protocol,'' in *Proc. 16th Int. Conf. Availability, Rel. Secur.*, Aug. 2021, pp. 1–11.

[21] X. Zhang, L. Guo, Y. Xue, and Q. Zhang, "A two-way VoLTE covert channel with feedback adaptive to mobile network environment," *IEEE Access*, vol. 7, pp. 122214–122223, 2019.

[22] S. Schmidt, W. Mazurczyk, R. Kulesza, J. Keller, and L. Caviglione, "Exploiting IP telephony with silence suppression for hidden data transfers," *Comput. Secur.*, vol. 79, pp. 17–32, Nov. 2018.

[23] J. Saenger, W. Mazurczyk, J. Keller, and L. Caviglione, "VoIP network covert channels to enhance privacy and information sharing," *Future Gener. Comput. Syst.*, vol. 111, pp. 96–106, Oct. 2020.

[24] S. Wendzel, S. Zander, B. Fechner, and C. Herdin, "Pattern-based survey and categorization of network covert channel techniques," *ACM Comput. Surv.*, vol. 47, no. 3, pp. 1–26, Apr. 2015.

[25] A. Velinov, A. Mileva, and D. Stojanov, "Power consumption analysis of the new covert channels in coap," *Int. J. Adv. Secur.*, vol. 12, no. 1, pp. 42–52, 2019.

[26] X. Ying, G. Bernieri, M. Conti, and R. Poovendran, "TACAN: Transmitter authentication through covert channels in controller area networks," in *Proc. 10th ACM/IEEE Int. Conf. Cyber-Phys. Syst.*, 2019, pp. 23–34.

[27] W. Lucia and A. Youssef, "Covert channels in stochastic cyber-physical systems," *IET Cyber-Phys. Syst., Theory Appl.*, vol. 6, no. 4, pp. 228–237, Dec. 2021.

[28] M. Hildebrandt, K. Lamshöft, J. Dittmann, T. Neubert, and C. Vielhauer, "Information hiding in industrial control systems: An OPC UA based supply chain attack and its detection," in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, 2020, pp. 115–120.

[29] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding—A survey," *Proc. IEEE*, vol. 87, no. 7, pp. 1062–1078, Jul. 1999.

[30] W. Mazurczyk and L. Caviglione, "Steganography in modern smartphones and mitigation techniques," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 334–357, 1st Quart., 2015.

[31] L. Caviglione, S. Wendzel, and W. Mazurczyk, "The future of digital forensics: Challenges and the road ahead," *IEEE Security Privacy*, vol. 15, no. 6, pp. 12–17, Nov./Dec. 2017.

[32] J. Xing, Q. Kang, and A. Chen, "NetWarden: Mitigating network covert channels while preserving performance," in *Proc. 29th USENIX Secur. Symp. (USENIX Secur.)* Berkeley, CA, USA: USENIX Assoc., 2020, pp. 2039–2056.

[33] L. Caviglione, M. Podolski, W. Mazurczyk, and M. Ianigro, "Covert channels in personal cloud storage services: The case of dropbox," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 1921–1931, Aug. 2017.

[34] G. Lewandowski, N. B. Lucena, and S. J. Chapin, "Analyzing network-aware active wardens in IPv6," in *Proc. 8th Int. Workshop Inf. Hiding (IH)*, 2006, pp. 58–77.

[35] T. Schmidbauer and S. Wendzel, "Hunting shadows: Towards packet runtime-based detection of computational intensive reversible covert channels," in *Proc. 16th Int. Conf. Availability, Rel. Secur.*, Aug. 2021, pp. 1–10.

[36] C. Song, Y. Zhang, and G. Lu, "Reversible data hiding in encrypted images based on image partition and spatial correlation," in *Proc. Int. Workshop Digit. Watermarking*. Berlin, Germany: Springer-Verlag, 2018, pp. 180–194.

[37] B. Ma, X. Wang, B. Li, and Y.-Q. Shi, "A multiple linear regression based high-accuracy error prediction algorithm for reversible data hiding," in *Proc. Int. Workshop Digit. Watermarking*. Berlin, Germany: Springer-Verlag, 2018, pp. 195–205.

[38] B. Ou, X. Li, W. Li, and Y.-Q. Shi, "Pixel-value-ordering based reversible data hiding with adaptive texture classification and modification," in *Proc. Int. Workshop Digit. Watermarking*. Berlin, Germany: Springer-Verlag, 2018, pp. 169–179.

[39] W. Mazurczyk, P. Szaga, and K. Szczypiorski, "Using transcoding for hidden communication in IP telephony," *Multimedia Tools Appl.*, vol. 70, no. 3, pp. 2139–2165, Jun. 2014.

[40] L. Caviglione, "Can satellites face trends? The case of web 2.0," in *Proc. Int. Workshop Satell. Space Commun.*, Sep. 2009, pp. 446–450.

[41] S. Baraković and L. Skorin-Kapov, "Survey of research on quality of experience modelling for web browsing," *Qual. User Exper.*, vol. 2, no. 1, pp. 1–31, Dec. 2017.

[42] A. Custura, R. Secchi, and G. Fairhurst, "Exploring DSCP modification pathologies in the internet," *Comput. Commun.*, vol. 127, pp. 86–94, Sep. 2018.

[43] L. Caviglione, W. Mazurczyk, M. Repetto, A. Schaffhauser, and M. Zuppelli, "Kernel-level tracing for detecting stegomalware and covert channels in Linux environments," *Comput. Netw.*, vol. 191, May 2021, Art. no. 108010.

PRZEMYSŁAW SZARY received the B.Sc. degree in automatic control and robotics and the M.Sc. degree in telecommunications from the Warsaw University of Technology (WUT), in 2018 and 2020, respectively, where he is currently pursuing the Ph.D. degree in computer science. He currently works as an IT Security Analyst in the energy sector enterprise. His research interests include information hiding, networks security, and ICS security.

WOJCIECH MAZURCZYK (Senior Member, IEEE) received the B.Sc., M.Sc., Ph.D. (Hons.), and D.Sc. (Habilitation) degrees in telecommunications from the Warsaw University of Technology (WUT), Warsaw, Poland, in 2003, 2004, 2009, and 2014, respectively. He is currently a Professor with the Institute of Computer Science, WUT. He also works as a Researcher with the Parallelism and VLSI Group, Faculty of Mathematics and Computer Science, FernUniversität in Hagen, Germany. His research interests include bio-inspired cybersecurity and networking, information hiding, and networks security. He is involved in the technical program committee of many international conferences and also serves as a reviewer for major international magazines and journals. Since 2016, he has been the Editor-in-Chief of an open access *Journal of Cyber Security and Mobility*. He has been serving as an Associate Editor for the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and as a Mobile Communications and Networks Series Editor for *IEEE Communications Magazine*, since 2018.

STEFFEN WENDZEL received the Ph.D. and Habilitation degrees from the Faculty of Mathematics and Computer Science, FernUniversität in Hagen, Hagen, Germany, in 2013 and 2020, respectively. He is a Professor of information security and computer networks with Hochschule Worms, Germany, where he is also the Scientific Director of the Center for Technology and Transfer. In addition, he is a Lecturer with the Faculty of Mathematics and Computer Science, FernUniversität in Hagen. Before joining Hochschule Worms, he led a smart building security research team at Fraunhofer FKIE, Bonn, Germany. He has (co)authored more than 170 publications and (co-)organized several conferences and workshops (e.g., EICC'21; Sicherheit'16; and IWSMR'19, '20, and '21) and special issues for major journals, such as IEEE SECURITY AND PRIVACY (S&P), *Future Generation Computer Systems* (FGCS) (Elsevier), *Journal of Universal Computer Science* (J.UCS), *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications* (JoWUA), and IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS (TII). He is an Editorial Board Member of *J.UCS*, *Journal of Cyber Security and Mobility* (JCSM), and *Frontiers in Computer Science*. For more details, visit his personal website (https://www.wendzel.de).

LUCA CAVIGLIONE received the Ph.D. degree in electronics and computer engineering from the University of Genoa, Genoa, Italy. He is currently a Senior Research Scientist with the Institute for Applied Mathematics and Information Technologies, National Research Council of Italy. He has been involved in research projects funded by ESA, EU, and MIUR. He is a Work Group Leader of the Italian IPv6 Task Force, a Contract Professor, and a Professional Engineer. His current research interests include networks security and information hiding, cloud architectures, and optimization of large-scale computing systems. He is involved in the technical program committee of international conferences and serves as a reviewer for international journals. He is the author or coauthor of over 150 reviewed scientific publications and holds several patents in the field of peer-to-peer computing and energy efficiency of datacenters.